KU LEUVEN

FACULTEIT INGENIEURSWETENSCHAPPEN

Energieverbruik voorspellen en clusteren met Gaussiaanse processen

Christiaan Leysen

Thesis voorgedragen tot het behalen van de graad van Master of Science in de ingenieurswetenschappen: computerwetenschappen, hoofdspecialisatie Artificiële intelligentie

Promotoren:
Prof. dr. Luc De Raedt
Dr. Tom Tourwé

Assessoren:
Dr. Raoul Strackx
Dr. ir. Wannes Meert

Begeleiders:
Dr. ir. Wannes Meert
Dr. Mathias Verbeke
Pierre Dagnely

Academiejaar 2015 – 2016

Voorwoord

Ik denk niet dat er de voorbije maanden een uur voorbij ging zonder dat ik aan Gaussiaanse processen heb gedacht... Na 930 werkuren is het eindelijk zover! Mijn laatste werk als student ingenieur zit erop en mijn ingenieurscarrière kan beginnen. Deze thesis is een werk dat ik met veel trots voorstel. Het heeft me veel bijgeleerd over hoe onderzoek wordt gedaan en hoe het wordt gerapporteerd. Vooral de zoektocht naar een generaal model voor verschillende tijdreeksen gebruikmakend van Gaussiaanse processen was een fantastisch leuke en soms frustrerende uitdaging. De publicatie hiervan was zeker mijn favoriete moment van heel de thesis. Zonder de hulp van een groep mensen was dit natuurlijk niet mogelijk geweest en ik zou hen dan ook hartelijk willen bedanken voor hun hulp:
Mijn begeleiders, Wannes Meert, Mathias Verbeke en Pierre Dagnely. Zij hebben me fantastisch bijgestaan en gepusht tot het afleveren van een goed werk.
Marion Neumann voor haar expertise in Gaussiaanse processen.
Clara Verhelst en 3E voor de samenwerking, de geleverde feedback en de data.
Mijn promotor Tom Tourwé en Sirris voor de samenwerking en de feedback.
Mijn promotor Luc De Raedt voor de feedback.
Mijn vriendin Stephanie voor de nodige steun en haar geduld terwijl ik steeds aan mijn thesis werkte.
Mijn familie: mama, Bieke, Simon en Johan voor de steun en speciaal Jef Roefs voor het nalezen, de hulp en de steun.
Mijn vrienden: Joris Gielen voor de hulp bij de afbeeldingen en het nalezen, Dario Incalza en de Tielense vrienden voor de nodige ontspanning.
De voltallige jury voor de tijd en moeite te nemen om mijn werk te bestuderen.

Na deze grote bedankingsspeech, die in een Oscaruitreiking niet had misstaan, eindig ik hoe dat ik begonnen ben. Ik begon deze thesis onwetend welke technieken ik ging bestuderen maar ik ben ontzettend blij dat het Gaussiaanse processen zijn geworden want ik vond het een razend interessant onderwerp. Het feit dat ik kon afwisselen tussen de voorspelling en de clustering hield me op dreef en ik heb dan ook met veel plezier aan deze thesis gewerkt. Ik hoop deze tekst dit reflecteert en dat het voor de lezer even interessant is.

Arré, zoewaait zemme dan ok wéral.

Christiaan Leysen

Inhoudsopgave

Samenvatting

Vandaag de dag is elektriciteit een basisbehoefte. Doordat de elektriciteitsvraag elk jaar sterk stijgt, moet ook de hoeveelheid opgewekte energie elk jaar opgedreven worden. Dit gebeurt meer en meer op een duurzame manier. Het nadeel hiervan is echter dat de productie op deze manier zeer sterk kan fluctueren, afhankelijk van de weersomstandigheden. Energiebedrijven hebben daarom een goed zicht nodig op de consumptie van elektrische energie en doen hiervoor vaak beroep op voorspellings- en/of clustermethoden. In deze context stelt dit werk een voorspellings- en clustermethode voor, die gebaseerd zijn op Gaussiaanse processen.

Deze thesis is opgedeeld in een voorspellings- en een clustergedeelte. In het voorspellingsgedeelte bespreken we hoe we de ruwe data verwerken tot input voor de Gaussiaanse proces regressie en focussen we ons op een voorspelling voor de volgende twee dagen per uur.
Het clustergedeelte van de thesis stelt een nieuwe clustermethode voor, die gebaseerd is op Gaussiaanse proces regressie (GPRC), en passen we toe op het consumptiegedrag van huishoudens om er inzichten in te ontdekken. Dit doen we door de weekprofielen (tijdreeksen) van de huishoudens te beschouwen. Om deze te clusteren zal de methode gebruik maken van een algemeen model dat geleerd wordt op een set van tijdreeksen, gebaseerd op hun waarschijnlijkheid. Het voordeel van de voorgestelde techniek is dat ze geen paarsgewijze vergelijking van de tijdreeksen nodig heeft, in tegenstelling tot vele andere clustermethoden voor tijdreeksen.

Deze methoden worden geëvalueerd op een *real-life* dataset van 71 huishoudens, die historische consumptie en meteo-data van één jaar bevat. De voorspellingsmethode wordt geëvalueerd en vergeleken met lineaire regressie, *support vector regressie* en een baseline methode die de waarde van een week geleden teruggeeft als voorspelling.
De clustermethode wordt vergeleken met *k-medoids* met *dynamic time warping* en hiërarchisch agglomeratief clusteren met *dynamic time warping*. Er wordt enerzijds aangetoond dat GPRC een betere schaalbaarheid heeft en anderzijds dat de resultaten ervan nuttig zijn in het beslissingsproces van een bedrijf uit de energiesector.

Lijst van figuren en tabellen

Lijst van figuren

Lijst van tabellen

Lijst van afkortingen en symbolen

Afkortingen

AR	Auto-regressie
CDF	Cumulatieve distributiefunctie
CV	Cross-validation
DTW	Dynamic time warping
GP	Gaussiaans proces
GPR	Gaussiaanse proces regressie
GPRC	Clustering gebaseerd op Gaussiaanse processen regressie
HAC	Hiërarchisch agglomeratief clusteren
MAE	Gemiddelde absolute fout
MAP	Maximum a posteriori
MRE	Gemiddelde relatieve fout
OLS	Lineaire regressie gebaseerd op de kleinste kwadratenmethode
PDF	Kansdichtheidsfunctie
RMSE	Root mean square error
SVM	Support vector machines
SVR	Support vector regressie

Hoofdstuk 1

Inleiding

Deze thesis onderzoekt het gebruik van Gaussiaanse processen voor het voorspellen van energieverbruik van huishoudens en het clusteren van hun gebruiksprofielen. De methoden zullen geëvalueerd worden op een reële dataset, geleverd door 3E[1], in samenwerking met Sirris[2], 3E en KU Leuven. Voor het voorspellen maken we gebruik van de bestaande Gaussiaanse proces regressie en vergelijken we onze resultaten met de best bevonden methode uit een voorgaande thesis rond hetzelfde onderwerp, *Machine Learning Techniques for Forecasting of Building Energy Consumption* [1]. Voor het clusteren van de gebruiksprofielen stellen we een nieuwe methode voor, die steunt op de Gaussiaanse proces regressie. We zullen deze methode theoretisch beschrijven, aantonen dat deze methode competitief is met andere clustermethoden en praktisch bruikbaar wordt geacht door een bedrijf als 3E.

In dit eerste hoofdstuk geven we de lezer een introductie van de thesis. Sectie 1.1 beschrijft de relevantie van het onderwerp en de motivatie tot deze thesis. Vervolgens zal Sectie 1.2 de belangrijkste onderzoeksvragen behandelen. Sectie 1.3 beschrijft de geleverde bijdrage en tot slot zal Sectie 1.4 de structuur van dit werk toelichten.

1.1 Motivatie

Vandaag de dag is elektriciteit een basisbehoefte. Doordat de elektriciteitsvraag elk jaar sterk stijgt, moet ook de hoeveelheid opgewekte energie elk jaar opgedreven worden. De laatste jaren is er echter een verschuiving te merken van traditionele opwekking naar opwekking door middel van duurzame energie.
Met traditionele opwekking bedoelen we opwekking die steunt op het gebruik van fossiele brandstoffen of kernenergie. Duurzame energie is energie waarover de mensheid voor onbeperkte tijd kan beschikken en waarbij, door het gebruik ervan, het leefmilieu en de mogelijkheden voor toekomstige generaties niet worden benadeeld. De belangrijkste voorbeelden van duurzame energiebronnen zijn bio-energie, windenergie, waterenergie en zonne-energie.

[1] `http://www.3e.eu`
[2] `http://www.sirris.be`

Volgens het internationale energie agentschap[3] werd er in 2013 wereldwijd ongeveer 22% van de energie geproduceerd door hernieuwbare energiebronnen. Dit cijfer stijgt elk jaar en het lijkt dus dat de toekomst van de energieopwekking in de goede richting evolueert.
Een nadeel van deze productiemethoden is echter dat de productie zeer sterk kan fluctueren, afhankelijk van de weersomstandigheden. Het huidige energienet is niet ontworpen met dit fluctuerend karakter in het achterhoofd en het is dus een uitdaging deze productiemethoden hierin te integreren. Er zijn twee manieren waarop dit nadeel kan beperkt worden. Men kan enerzijds een elektrische opslagcapaciteit voorzien in het energienet, om overbelasting van het netwerk te voorkomen of men heeft een zeer goed beeld van de consumptiezijde nodig. Aangezien de huidige technologie niet in staat is om elektrische energie op te slaan op deze grote schaal, is het dus van groot belang de consumptie van energie zo goed mogelijk in kaart te brengen.
In de praktijk wordt dit vertaald in marktregels. De dag voor de consumptie zullen de elektriciteitsleveranciers de geschatte energievraag doorgeven aan de energieproducenten. Wanneer de gevraagde hoeveelheid niet overeenstemt met de productiehoeveelheid zal de onbalans moeten weggewerkt worden door de productiehoeveelheden aan te passen. Deze productie-aanpassingen zijn uiteraard zeer duur en vormen dus ook een significante kost voor de transmissiesysteemoperator. Om deze onbalanskosten te reduceren, is het dus belangrijk dat de energieleveranciers een zo goed mogelijke schatting van de consumptie van de klanten kennen. Het is daarom van uiterst belang een goede korte termijn schatting van de elektriciteitsvraag te hebben [2].
Een bijkomende moeilijkheid voor het energienet is de toename van decentrale opwekking van energie. Het bekendste voorbeeld hiervan is het plaatsen van zonnepanelen bij particulieren. Op dit moment wordt er zelden gebruik gemaakt van energieopslag in huis. De opgewekte elektrische energie van de zonnepanelen die niet wordt verbruikt door het gezin wordt hierdoor dus naar elektrisch energienet geleid. Op deze manier wordt de particulier zowel consument als producent. Dit heeft een grote invloed op de schatting van de hoeveelheid benodigde energie. Niet enkel heeft het een grote invloed maar het maakt de schatting ook moeilijker doordat er relatief weinig communicatie is tussen de particulier en de traditionele leveranciers. Een mogelijke oplossing die naar de toekomst toe wordt voorgesteld is de invoer van een *smart grid*. Dit is een elektriciteitssysteem dat constante communicatie tussen gebruiker en leverancier voorziet om te komen tot een schoon, veilig, betrouwbaar, efficiënt en duurzaam elektriciteitssysteem [3]. Praktisch komt het er dan op neer dat er intelligente energiemeters worden geïnstalleerd bij de klanten thuis die informatie doorsturen en ontvangen van en naar de leveranciers en we dus zo een tweerichtingsverkeer creëren tussen producent en consument. Op deze manier krijgen we accurate data van het energieverbruik. Deze data kan dan gebruikt worden in een voorspellingsmodel zodat de energieproducenten een zo goed mogelijke schatting van het energieverbruik verkrijgen. Tevens geven de voorspellingen ook een beter zicht op het energiegebruik van de klant zelf [4].
We opteren om Gaussiaanse proces regressie te onderzoeken omdat deze methode

[3] `http://www.iea.org`

uit eerder onderzoek zeer geschikt blijkt voor het voorspellen van energieconsumptie (Hoofdstuk 3).
Niet enkel de voorspelling maar ook de manier waarop een klant zijn energie gebruikt, kan van commercieel belang zijn voor de energieleveranciers. Dit hangt vaak af van verschillende factoren (b.v. aantal gezinsleden, aanwezigheid) en varieert vaak. Deze variabelen zijn vaak onbekend of het is onduidelijk welke variabelen men moet opmeten en onderzoeken. In deze context stellen we dus naast een voorspellingsmethode ook een methode voor die de huishoudens kan groeperen (clusteren) volgens gelijkaardig gedrag. Deze resultaten kunnen door leveranciers gebruikt worden om de voorspelling te verbeteren, anomalieën te detecteren, om gebruiksprofielen af te leiden uit de consumptiedata van huishoudens of om energiepakketten voor te stellen. Onze clustermethode biedt het voordeel dat deze methode geen parameter instellingen van de gebruiker vereist voor het leren van het model alsook geen voorafgaand onderzoek en selectie van de te onderzoeken variabelen.
Het doel van deze thesis is dus enerzijds een zo goed mogelijke voorspelling proberen te bekomen zodanig dat zowel de energie-efficiëntie van de klant en de kosten voor de producenten verbeterd kunnen worden. Anderzijds willen we aantonen dat de voorgestelde clustermethode extra inzichten in het energieverbruik kan geven en dat ze praktisch nut heeft in een industriële context.

1.2 Onderzoeksvragen

In deze thesis willen we een antwoord vinden op onderstaande onderzoeksvragen:

- Voorspelling:
  - Wat is het effect van de features (gekozen factoren die we onderzoeken) op een auto-regressieve vs niet-auto-regressieve methode?
  - Hoe goed is de voorspelling bekomen door de Gaussiaanse proces regressie in vergelijking met andere technieken?
- Clustering:
  - Hoe kunnen we de Gaussiaanse proces regressie gebruiken om tijdreeksen (data doorheen de tijd verzameld) te clusteren?
  - Hoe goed is de clustering in vergelijking met andere technieken?
  - Welke inzichten kunnen we detecteren met deze clustermethode?
  - Hoe kan deze clustering gebruikt worden in de praktijk?

Om de vragen in verband met de voorspellingsmethode te beantwoorden, zullen we een diepgaand onderzoek naar Gaussiaanse proces regressie voeren. Daarnaast zal de methode vergeleken worden met andere gebruikte technieken in deze context. De kwaliteit van de ontwikkelde methode zal experimenteel vergeleken worden met andere technieken op een reële dataset en we bespreken hoe deze data gebruikt wordt om tot een zo goed mogelijke voorspelling van de energieconsumptie te komen. De

data bestaat uit historische energiedata van 71 verbruiksprofielen van huishoudens en overeenkomstige historistische meteorologische data.
Om de onderzoeksvragen in verband met de clustering te beantwoorden wordt eerst een uiteenzetting gegeven over hoe we GPR gebruiken en aanpassen zodat we een model kunnen leren over een set van verschillende tijdreeksen in plaats van voor één tijdreeks. In de bijhorende experimenten zullen zowel de resultaten als de complexiteit van deze nieuwe methode vergeleken worden met twee bekende methoden voor het clusteren van temporele data. Ten slotte zullen we kort bespreken wat het praktisch nut van deze methode is voor een bedrijf als 3E.

1.3 Bijdrage

Het eerste deel van deze thesis biedt een uitgebreide uiteenzetting over het voorspellen van tijdreeksen met Gaussiaanse proces regressie. Waar de voorgaande thesis rond hetzelfde onderwerp [1] niet-auto-regressieve methoden onderzocht zal dit werk dus een auto-regressieve methode onderzoeken en vergelijken. De besproken niet-auto-regressieve methoden zullen een regressie uitvoeren om een verband te vinden tussen de verschillende onderzochte features zonder het tijdsaspect intrinsiek in rekening te brengen. Een auto-regressieve methode zal het tijdsaspect van deze features wel intrinsiek in rekening brengen door een verband te onderzoeken tussen de features en vorige waarden van deze features. Deze informatie moest in de voorgaande thesis expliciet toegevoegd worden door zelf te onderzoeken welke waarden uit het verleden gecorreleerd zijn met de huidige waarde. Deze informatie moest daarna expliciet toegevoegd worden aan de feature set. (In onze context bv. het energieverbruik van een week geleden.)
Een tweede deel van dit werk zal een nieuwe modelgebaseerde clustermethode voorstellen die steunt op Gaussiaanse proces regressie. Dit is de eerste keer dat GPR gebruikt en aangepast wordt voor het leren over een set van tijdreeksen. We tonen aan dat deze methode beter schaalbaar is dan andere bekende clustermethoden voor tijdreeksen, dat ze praktisch nut heeft en dat ze eenvoudig in gebruik is. Deze aanpassing zullen we ook voorstellen aan de auteurs van de gebruikte toolbox (pyGP) zodat ze deze kunnen toevoegen.

1.4 Structuur van de thesis

Eerst wordt in Hoofdstuk 2 de nodige achtergrond besproken die verwacht wordt voor deze thesis. Dan zal Hoofdstuk 3 een overzicht tonen van de onderzochte literatuur. Hoofdstuk 4 zal de aanpak bespreken voor het voorspellen met Gaussiaanse Processen. Vervolgens zal Hoofdstuk 5 de theoretische uitwerking beschrijven voor het clusteren met Gaussiaanse processen. Hoofdstuk 6 zal de uitgevoerde experimenten en hun resultaten bespreken en tenslotte overlopen we in Hoofdstuk 7 nog eens de belangrijkste conclusies en resultaten.

Hoofdstuk 2

Achtergrond

In deze thesis werd onderzoek gevoerd naar twee verschillende probleemstellingen. Enerzijds het voorspellen van het energieverbruik van huishoudens met Gaussiaanse processen. Anderzijds het gebruik van deze Gaussiaanse processen voor het clusteren (groeperen) van huishoudens met gelijkaardige verbruikspatronen. Dit hoofdstuk bespreekt de nodig achtergrond voor deze thesis en de concepten zijn onderverdeeld volgens de twee probleemstellingen. Sectie 2.1 bespreekt de achtergrond die nodig is voor het voorspellingsgedeelte van dit werk. Hierin wordt een uiteenzetting gegeven van tijdreeksanalyse en de technieken die we in deze thesis zullen gebruiken: lineaire regressie, support vector regressie en Gaussiaanse proces regressie. Sectie 2.2 zal vervolgens de nodige achtergrond voor het clustergedeelte bespreken. Hierin worden *dynamic time warping* (DTW), *K-medoids met DTW*, en hiërarchisch clusteren met DTW toegelicht.

2.1 Voorspelling

Eerst zal Sectie 2.1.1 tijdreeksanalyse beschrijven. Vervolgens zullen in Secties 2.1.2, 2.1.3, 2.1.4 en 2.1.5 de gebruikte methoden toegelicht worden.

2.1.1 Tijdreeksanalyse en decompositie

Definitie 1. *Een tijdreeksanalyse is een analyse van data die sequentieel verzameld is in de loop van een bepaalde tijdspanne, genoteerd als:*

$$y = y_1, y_2, y_3, \ldots \tag{2.1}$$

waarbij y_i waarden zijn op verschillende tijdspunten, afkomstig van bv. een sensor.

Tijdreeksanalyse omvat methoden voor het analyseren van tijdreeksdata en om zinnige statistieken en andere karakteristieken te beschrijven. Wanneer we een beschrijvende analyse van een tijdreeks geven, kunnen we drie soorten patronen ontdekken: trends, seizoensgebondenheid of cyclussen. Deze componenten kunnen we als volgt definiëren [5]:

- Een trend is een langdurig merkbare stijging of daling in data.
- Seizoensgebondenheid is een patroon dat ontstaat wanneer de tijdreeks beïnvloed wordt door seizoensfactoren zoals maanden of dagen van de week. Seizoensgebondenheid gaat altijd over een vaste periode.
- Een cyclus ontstaat wanneer de data stijgingen en dalingen vertoont die zich niet over een vaste periode manifesteren.

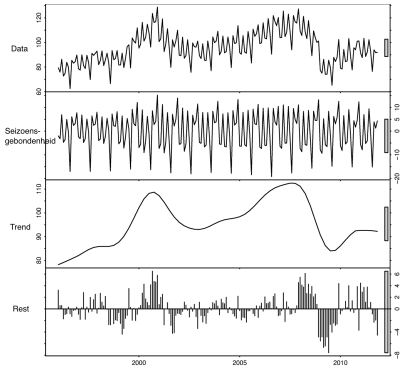
Tevens is het ook mogelijk dat er een combinatie van de verschillende componenten zichtbaar is.

We kunnen een tijdreeks y_t beschouwen als een combinatie van drie componenten zoals in Figuur 2.1. Twee componenten bestaan uit voorgaand beschreven patronen nl. een seizoensgebonden component en een trend-cycluscomponent. En een derde component is een restcomponent die het resterend deel van de tijdreeks bevat. We kunnen dit formeel beschrijven als een additief- of multiplicatief model [5]:

$$y_t = S_t + T_t + R_t \tag{2.2}$$

$$y_t = S_t \times T_t \times R_t \tag{2.3}$$

waarbij y_t de data op moment t is, S_t de seizoensgebonden component op moment t is, T_t de trend-cyclus-component op moment t en R_t de restcomponent op moment t is. Het additief model is het meest geschikt als de grootte van seizoensgebonden fluctuaties of de variatie rond de trend-cyclus niet varieert met data van de tijdreeks zelf [5].
Een andere soort van data-analyse binnen tijdreeksanalyse is het voorspellen van data gebruikmakend van voorspellingsmodellen. Dit noemen we regressie-analyse (of kortweg regressie), en is een statistische techniek voor het analyseren van gegevens waarin (mogelijk) sprake is van een specifieke samenhang. In de volgende secties zullen we enkele regressiemethoden bespreken.



Figuur 2.1: De tijdreeksdecompositie met bovenaan de data, daaronder respectievelijk de seizoenseffecten, de trend en de rest [5].

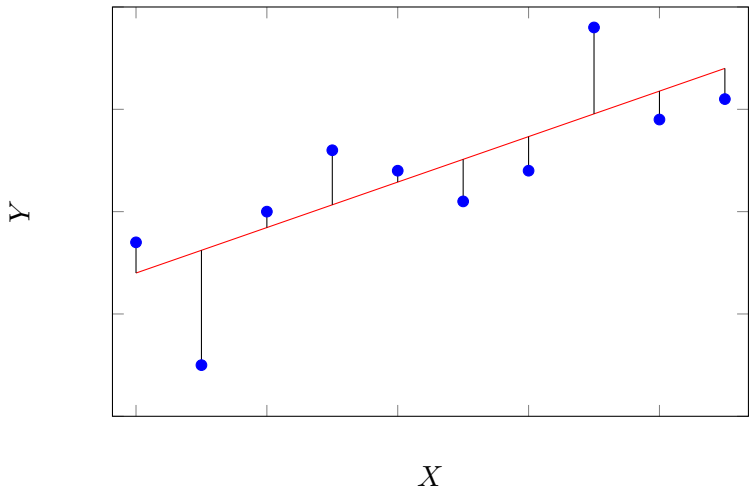
2.1.2 Lineaire regressie gebaseerd op de kleinste kwadratenmethode

Voor de toelichting van lineaire regressie gebaseerd op de kleinste kwadratenmethode (OLS) volgen we de uiteenzetting van De Somer en Kutz [1].
Lineaire regressie is één van de eenvoudigste technieken die gebruikt worden voor het voorspellen van tijdreeksen. Het doel is de beste benaderende functie te vinden doorheen de beschikbare datapunten. De variabele die hierbij voorspeld moet worden (doelvariabele) wordt gemodelleerd als een lineaire combinatie van de inputvariabelen. Een multi-regressiemodel met meer dan één variabele wordt geschreven als:

$$y = w_0 + w_1 x_1 + \ldots + w_p x_p \tag{2.4}$$

waarbij y de voorspelde waarde voorstelt, x_i de input variabele en w_i de coëfficiënten. Voor het schatten van de onbekende coëfficiënten van dit lineaire regressiemodel kan de kleinste kwadraten benadering gebruikt worden. Het schat de coëfficiënten $w = (w_1, \ldots, w_p)$, zodanig dat de som van de kwadraten tussen de verwachte waarde van de dataset en het doel, voorspeld door de lineaire benadering, geminimaliseerd wordt (zie Figuur 2.2). Ofwel, het minimaliseert de som van de kwadraten van de verticale afstanden tussen elk datapunt en de regressielijn. Hoe kleiner deze afstanden, hoe beter de fitting van het model voor deze dataset. Wiskundig lossen we dus een probleem op van de vorm:

$$\min_w ||Xw - y||^2 \tag{2.5}$$



Figuur 2.2: Lineaire regressie gebaseerd op de kleinste kwadratenmethode.

2.1.3 Support vector regressie

Voor de toelichting van *support vector regressie* volgen we deels de uiteenzetting van De Somer en Kutz [1].
Support Vector Machines (SVM) werden origineel geïntroduceerd voor classificatieproblemen. In 1996 werd echter door H. Drucker et al. [6] *Support Vector Machines* voor regressiedoeleinden voorgesteld.
Het basisidee voor het oplossen van een niet-lineair regressieprobleem met SVMs is een oplossing gebaseerd op een transformatie die de input variabele transformeert naar een hoger dimensionale ruimte waarin het probleem gereduceerd wordt tot een lineair probleem. De functie die dan benaderd moet worden kan dan gedefinieerd worden als:

$$y = w^T * \phi(x) + b \tag{2.6}$$

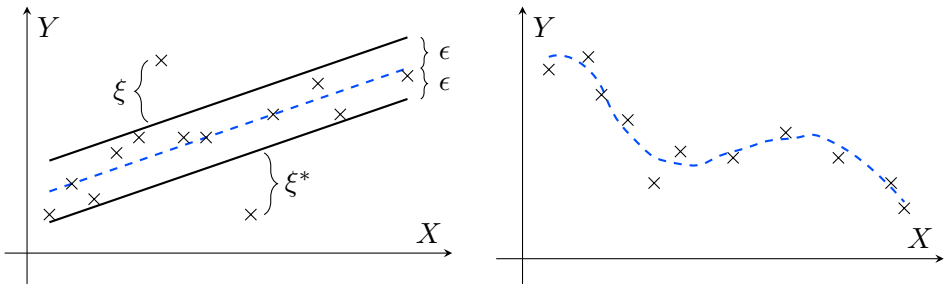
waarbij $\phi(x)$ op de transformatie naar de geïnduceerde ruimte duidt en w de gewichtsvector is. Daarnaast wordt er ook een fout-ongevoelige ϵ-zone gespecificeerd, zodanig dat alle data elementen in deze zone behandeld worden alsof ze perfect benaderd zijn door de resulterende functie. Enkel voor elementen buiten de ϵ-zone worden fouten geregistreerd in termen van zogenaamde *slack*-variabelen ξ. Voor elk element worden twee *slack*-variabelen geïntroduceerd om het verschil tussen de twee zijden van de zone aan te duiden (Figuur 2.3 (links)). Het afgeleide optimalisatieprobleem is vervolgens:

$$\min_{w,b,\xi_i,\xi_i^*} \frac{1}{2} w^T w + C \sum_{i=1}^{N} (\xi_i + \xi_i^*) \tag{2.7}$$

rekening houdend met

$$\begin{aligned} y_i - w^T \phi(x_i) - b &\leq \epsilon + \xi_i, i = 1, \ldots, N \\ w^T \phi(x_i) + b - y_i &\leq \epsilon + \xi_i^*, i = 1, \ldots, N \\ \xi_i, \xi_i^* &\geq 0, i = 1, \ldots, N \end{aligned}$$

De eerste term in vergelijking 2.7 is een regularisatieterm die de resulterende functie zo simpel mogelijk maakt. Wanneer we een gewicht op nul zetten, betekent dit dat we een bepaald data element voor de constructie van de functiebenadering gaan negeren. Dit resulteert in een vlakkere niet-lineaire functie wanneer we de terugtransformatie naar de originele inputruimte uitvoeren. Alle datapunten waarbij de oplossing een gewicht bevat dat niet nul is, worden *support vectors* genoemd. De sanctieparameter C beïnvloedt de trade-off tussen een kleine benaderingsfout en een lage functiecomplexiteit.
Het uitdagende deel van de SVM is de transformatie naar een hoog-dimensionale ruimte, waarbij het niet-lineaire probleem wordt gereduceerd tot een lineair probleem. Vaak wordt deze transformatie niet rechtstreeks beschreven door de functie $\phi(x)$ maar door de correlaties $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ tussen de elementen van de ruimte. $K(x_i, x_j)$ wordt ook de *kernel*-functie genoemd en is in de meeste gevallen lineair



(a) (b)

Figuur 2.3: (a) SVM voor regressie in de hoog-dimensionale ruimte is een lineair regressiemodel met foutongevoelige ϵ-zone en *slack*-variabelen ξ en ξ^*. (b) Het terug transformeren van de oplossing naar de originele input ruimte resulteert in een niet-lineaire functiebenadering [1].

of exponentieel. Vooral de lineaire en radiale-basis kernelfunctie zijn populaire kernelfuncties, resprectievelijk:

$$K_{LIN}(x_i, x_j) = x_i \cdot x_j \tag{2.8}$$

$$K_{RBF}(x_i, x_j) = \exp\left(-\gamma||x_i - x_j||_2^2\right) \tag{2.9}$$

met $\gamma = \frac{1}{2\sigma^2}$ de kernelparameter en σ de signaalvariantie. Wanneer de oplossing terug getransformeerd wordt naar de originele inputruimte, wordt een niet-lineaire functiebenadering bekomen, zoals op Figuur 2.3 (rechts) kan worden waargenomen.

2.1.4 Auto-regressie

Het auto-regressieve (AR) model is een model uit de tijdreeksanalyse dat gebruikt wordt om toekomstige waarden te voorspellen [7]. In tegenstelling tot een regressief model, waarbij voorspelling van de output een lineaire combinatie is van de voorspellende variabelen, specificeert een AR model de output als een lineaire combinatie van voorgaande waarden van de voorspelde variabele en een stochastische term. Het model kan dus formeel beschreven worden door volgende stochastische differentievergelijking van orde p [5]:

$$y_t = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \epsilon_t \tag{2.10}$$

waarbij c een constante en ϵ witte ruis is en $\phi_1, \ldots, \phi_p$ de parameters van het model zijn.
Dit is als een meervoudige regressie maar dan met vertraagde waarden van y_t als voorspellers. We noemen dit het AR(p) model [5]. Binnen deze thesis wordt gebruik gemaakt van een concrete auto-regressieve methode, Gaussiaanse processen.

2.1.5 Gaussiaanse proces regressie

Deze sectie volgt grotendeels de uiteenzetting van het werk van Rasmussen & Williams [8] (en Samarasinghe et al. [9]).

Cumulatieve distributiefunctie

Voor elk reëel getal x gaande van $-\infty$ tot $+\infty$, wordt de cumulatieve distributiefunctie (CDF) van een willekeurige variabele X gegeven door:

$$F_X(x) = P\{X \leq x\} \tag{2.11}$$

Het is de kans dat de waarde van een bepaalde willekeurige variabele X kleiner dan of gelijk is aan de beschouwde waarde x.

Kansdichtheidsfunctie

De kansdichtheidsfunctie (PDF) is de afgeleide van de cumulatieve distributiefunctie (CDF) en kan dus worden beschreven via de volgende vergelijking:

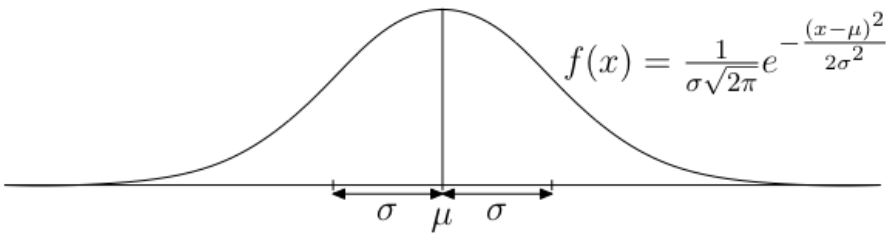
$$f_X(x) = \frac{dF_X(x)}{dx} \tag{2.12}$$

Gaussiaanse verdeling

Een Gaussiaanse of normaalverdeelde willekeurige variabele X wordt aangeduid door $X \sim \mathcal{N}(\mu, \sigma^2)$. Waarbij μ het gemiddelde aanduidt en σ de standaard afwijking. De kansdichtheidsfunctie (PDF) van een normaalverdeling wordt gegeven door:

$$fx(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{2.13}$$

De grafiek van de PDF heeft de typische klokcurve, zoals te zien op Figuur 2.4. Deze curve is symmetrisch rond het gemiddelde μ en de curve is platter naarmate de variantie σ^2 vergroot, wat een grotere afwijking van het gemiddelde voorstelt. Merk op dat de oppervlakte onder deze curve steeds gelijk is aan één.



Figuur 2.4: De normale kansdichtheidsfunctie

Gaussiaanse processen

Gaussiaanse processen vinden hun oorsprong in statistiek. Tijdreeksanalyse met GP werd al in 1880 gebruikt door de astronoom T.N. Thiele en in de late veertiger jaren van de vorige eeuw werden ze gebruikt om trajecten van militaire doelwitten te bepalen in het werk van Wiener-Kolmogorov [10]. Rond het begin van de jaren 1990 vonden GP hun intrede in het veld van Machine Learning door het werk van Williams en Rasmussen [8] en momenteel worden ze als een brede oplossingsmethode aangezien. Hoewel GP dus al lange tijd gebruikt worden, worden ze binnen het veld van voorspellingsmethoden minder uitgebreid gebruikt dan andere bekende methoden zoals bijvoorbeeld artificiële neurale netwerken. Het laatste decennium stellen we echter een opmars van de GPs binnen dit veld. [11].
Een GP is een veralgemening van de normaalverdeling uit sectie 2.1.5. Waar een kansverdeling willekeurige variabelen beschrijft die scalars of vectoren (voor multivariate verdelingen) zijn, zal een stochastisch proces (zoals GP) de eigenschappen van een functie beschrijven. Intuïtief, kunnen we een functie ruwweg voorstellen als een lange vector waarbij elk element van de vector een functiewaarde $f(x)$ voorstelt voor een bepaalde inputwaarde x. Dit alles kunnen we formeel samenvatten in volgende definitie [8]:

Definitie 2. *Een Gaussiaans proces is een collectie van willekeurige variabelen, waarbij elke eindige lineaire combinatie van deze willekeurige variabelen een multivariate normale verdeling heeft.*

De notatie van een Gaussiaans proces kunnen we als volgt weergeven:

$$f(x) \sim \ \mathcal{GP}(m(x), k(x, x')) \tag{2.14}$$

waarbij $m(x)$ de gemiddelde functie is en $k(x, x')$ de covariantiefunctie. Het gemiddelde en de variantie beschrijven volledig een normale verdeling. De gemiddeldefunctie en covariantiefunctie beschrijven op hun beurt volledig een Gaussiaans proces.

Gemiddelde functie

De gemiddelde functie $m(x)$ wordt berekend als de verwachtingswaarde van $f(x)$:

$$m(x) = \mathbb{E}[f(x)] \tag{2.15}$$

In de meeste gevallen wordt deze functie nul beschouwd zonder verlies van generaliteit. In dit geval wordt de gemiddelde waarde van de functies bij elke x in de Gaussiaanse prior nul. Een prior duidt de veronderstelling van de data aan, voordat we naar de observaties kijken. Dat deze functie meestal nul wordt beschouwd, is echter geen nodige voorwaarde zoals besproken in Rasmussen en Williams [8].

Covariantiefunctie

De covariantiefunctie $k(x, x')$ is de meest besproken functie binnen GP. Deze functie wordt ook wel de kernelfunctie van het GP genoemd. Deze kernelgebaseerde non-parametrische aard van GP maakt het een zeer flexibel model.
De covariantie tussen twee willekeurige variabelen $f(x)$ en $f(x')$ kunnen we via volgende formule berekenen:

$$k(x, x') = \mathbb{E}[(f(x) - m(x))(f(x') - m(x'))] \tag{2.16}$$

wanneer we $m(x) = m(x') = 0$ beschouwen, verkrijgen we volgende formule:

$$k(x, x') = \mathbb{E}[(f(x))(f(x'))] \tag{2.17}$$

Als notatie voor de covariantie $k(x, x')$ wordt gebruikt, wordt de covariantie berekend tussen de functiewaarden $f(x)$ en $f(x')$. Het is een waarde voor de correlatie tussen twee willekeurige variabelen. Zoals vermeld is deze functie zeer belangrijk en de keuze van de kernel zal dan ook een groot effect op de nauwkeurigheid van de voorspelling hebben. Deze geselecteerde functie moet bepaalde eigenschappen bevatten [8]. De belangrijkste nodige voorwaarde is, dat de geselecteerde functie positief definiet en symmetrisch is ($k(x, x') = k(x', x)$).
De gebruikte kernelfuncties in dit werk zijn de lineaire kernel,

$$k_{LIN}(x, x') = \sigma_0^2 x \cdot x' \tag{2.18}$$

met als parameter de signaalvariantie σ_0^2 en de kwadratisch exponentiële kernel (of de radiale basisfunctie) met ruisterm,

$$k_{RBF}(x, x') = \sigma_f^2 \exp\left(-\frac{(x - x')^2}{2l^2}\right) + \sigma_n^2 \delta(x, x') \tag{2.19}$$

met als parameters de signaalvariantie σ_f^2, de ruisvariantie σ_n^2 en de lengteschaal l.

Gaussiaanse proces regressie

In deze sectie bespreken we hoe we GP kunnen gebruiken voor regressiedoeleinden (GPR).
Kort samengevat gaat het als volgt in zijn werk. Ten eerste veronderstellen we bij GPR een onderliggende Gaussiaanse prior functie en selecteren we een gepaste covariantiefunctie, die overeenstemt met de karakteristieken van de gebruikte data. Daarna zal het GP, dat gespecificeerd is met de geselecteerde kernelfuctie, een distributie van willekeurige variabelen produceren. Wanneer we de trainingsdata introduceren, zal het de best passende functies van de distributie selecteren. M.a.w. de functies die de data het best fitten. Op deze manier vindt het proces de beste set van functies uitgaande van de Gaussiaanse prior. Voor een goede voorspelling zijn er dus een

voldoende aantal trainingsvoorbeelden nodig.
De selectie van de beste functies hangt dus af van de keuze van de kernelfunctie en haar parameters. Deze parameters worden de hyperparameters genoemd en worden besproken in de sectie 2.1.5. Eerst bespreken we in meer detail hoe het regressieproces formeel in zijn werk gaat.

Stel dat we de volgende trainingsdataset $D = (x_i, y_i)|i = 1, 2, \ldots, n$ hebben met observaties die onderhevig zijn aan ruis. De data waarvoor we een voorspelling willen doen, noemen we de testdata. We willen een bepaalde doelwaarde y_* voorspellen voor een nieuwe inputwaarde x_*. Het probleem is dat we een functie moeten leren van de dataset, waarvoor we een aangenomen Gaussiaanse prior van functies moeten gebruiken. Maar de observaties en de onderliggende functiewaarden f zijn niet identiek doordat er ruis op de observaties zit. Hierdoor kunnen de doelwaarden voorgesteld worden als:

$$y = f(x) + \epsilon \tag{2.20}$$

Hierbij veronderstellen we dat het Gaussian ruismodel voorgesteld wordt door $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. De onderliggende functie f wordt benaderd door een GP met gemiddelde de nulfunctie en een covariantiefunctie. De meest gebruikte covariantiefunctie is de kwadratisch exponentiële covariantiefunctie. Gebruikmakend van deze functie, kunnen we vergelijking 2.14 herschrijven als:

$$f(x) \sim \ \mathcal{GP}(0, k(x, x')) \tag{2.21}$$

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{(x - x')^2}{2l^2}\right) \tag{2.22}$$

Wanneer we vergelijking 2.20 beschouwen, kunnen de eigenlijke observaties gespecificeerd worden door het ruismodel op te tellen bij onderliggende functie die gedefinieerd wordt door vergelijking 2.21. Dan kunnen we de covariantiefunctie bijhorend bij de doelwaarden y, noteren als:

$$cov(x, x') = k(x, x') + \sigma_n^2(\delta(x, x')) \tag{2.23}$$

waarbij $\delta(x, x')$ de Kronecker delta functie is die gelijk is aan: $\delta(x, x') = \begin{cases} 1 \ als \ x = x' \\ 0 \ als \ x \neq x' \end{cases}$
Gebruikmakend van de kernelfunctie, kunnen we de correlatie tussen alle trainingsdata bepalen. De matrix met als elementen al deze covarianties, noemen we de covariantiematrix en wordt genoteerd door K.

$$K = \begin{bmatrix} cov(x_1, x_1) & cov(x_1, x_2) & \cdots & cov(x_1, x_n) \\ cov(x_2, x_1) & cov(x_2, x_2) & \cdots & cov(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ cov(x_n, x_1) & cov(x_n, x_2) & \cdots & cov(x_n, x_n) \end{bmatrix} \tag{2.24}$$

De overeenkomstige covariantiematrix van de trainingsdata en de testdata wordt dan gegeven door K_*:

$$K_* = \begin{bmatrix} cov(x_*, x_1) & cov(x_*, x_2) & \cdots & cov(x_*, x_n) \end{bmatrix} \tag{2.25}$$

K_{**} specificeert de covariantiematrix van de testdata zelf.

$$K_{**} = cov(x_*, x_*). \tag{2.26}$$

De simultane verdeling van de observaties y en de voorspellingen y_*, is de multivariate normaalverdeling:

$$\begin{bmatrix} y \\ y_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K & K_*^T \\ K_* & K_{**} \end{bmatrix}\right) \tag{2.27}$$

Met K_*^T de transpose van de covariantiematrix van de trainingsdata en de testdata. De eigenlijke voorspelling wordt nu gegeven door de conditionele verdeling van y_* gegeven y.

$$y_*|y \sim \mathcal{N}\left(K_* K^{-1} y, K_{**} - K_* K^{-1} K_*^T\right) \tag{2.28}$$

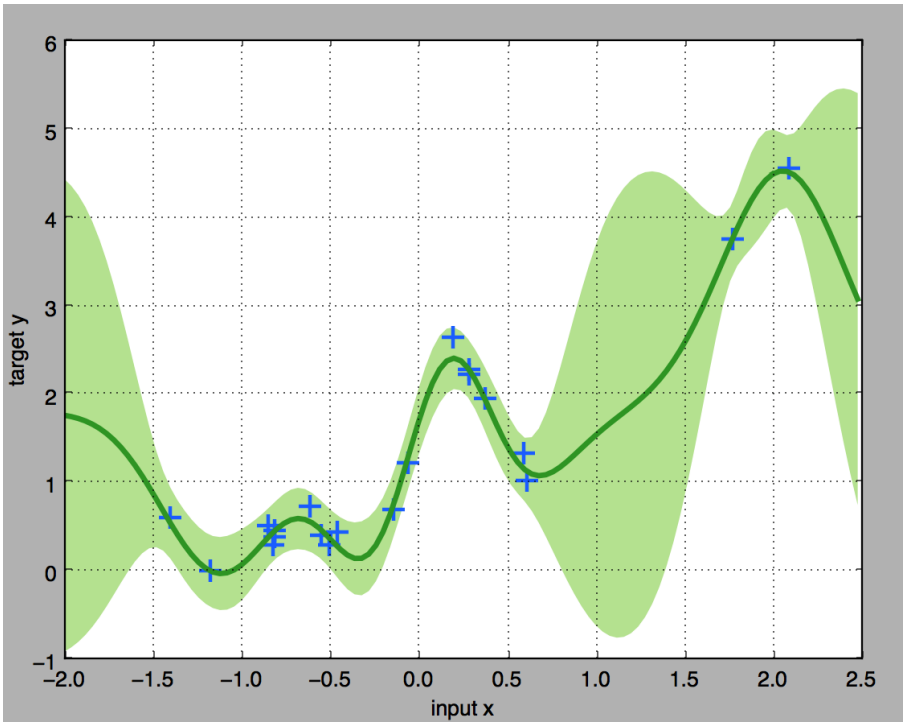
met K^{-1} de inverse van matrix K. Deze verdeling wordt de Gaussiaanse posterior verdeling genoemd. Volgens Rasmussen en Williams [8], wordt voor elke Gaussiaanse posterior het gemiddelde van de posterior verdeling als beste voorspelling van de variabele beschouwd. Dit noemt men de *maximum a posteriori* (MAP). In voorgaande vergelijking 2.28, is de eigenlijke voorspelling y_*, haar eigen gemiddelde. Dit is de MAP voorspelling:

$$\bar{y}_* = K_* K^{-1} y \tag{2.29}$$

De variantie van de voorspelling wordt gegeven door:

$$\mathbb{V}[y_*] = K_{**} - K_* K^{-1} K_*^T \tag{2.30}$$

Met deze variantie kunnen we het 95% betrouwbaarheidsinterval berekenen, via de formule $\pm 1.96\sqrt{\mathbb{V}[y_*]}$, dit is ongeveer dus tweemaal de standaarddeviatie van de posterior distributie. Een voorbeeld van het resultaat van de GPR is weergegeven in Figuur 2.5. Hierbij stellen de blauwe kruisjes de data punten voor, de volle groene lijn het posterior gemiddelde en de lichtgroene zone daarrond is het 95% betrouwbaarheidsinterval van de voorspelling. Merk op dat het betrouwbaarheidsinterval (en dus ook de variantie) groter zijn in de gebieden waar weinig datapunten bekend zijn.



Figuur 2.5: Gaussiaanse proces regressie: blauwe kruisjes zijn de datapunten, de groene volle lijn is het posterior gemiddelde en de lichtgroene zone is het 95% betrouwbaarheidsinterval van de voorspelling.

Om deze sectie te verduidelijken, zullen we een klein voorbeeld uitwerken. Stel dat we de doelwaarde y_* voor de nieuwe input $x_* = 5$ willen voorspellen (Tabel 2.1). We veronderstellen een radiale basisfunctie als covariantiefunctie met $\sigma_f = 3{,}7$, $l = 2{,}8$ en $\sigma_n = 0{,}1$.

Tabel 2.1: Voorbeeld oefening

x	1	2	3	4	5
y	3	4,5	6	6,25	?

We berekenen eerst de covariantiematrix van de trainingsdata K a.d.h.v. Vergelijkingen 2.23 en 2.24:

$$K = \begin{bmatrix} 13{,}70 & 12{,}84 & 10{,}61 & 7{,}71 \\ 12{,}84 & 13{,}70 & 12{,}84 & 10{,}61 \\ 10{,}61 & 12{,}84 & 13{,}70 & 12{,}84 \\ 7{,}71 & 10{,}61 & 12{,}84 & 13{,}70 \end{bmatrix} \tag{2.31}$$

De overeenkomstige covariantiematrix van de trainingsdata en de testdata wordt dan berekend via Vergelijking 2.25:

$$K_* = \begin{bmatrix} 4{,}93 & 7{,}71 & 10{,}61 & 12{,}84 \end{bmatrix} \tag{2.32}$$

Vervolgens berekenen we de covariantiematrix van de testdata via vergelijking 2.26:

$$K_{**} = \Big[13{,}70\Big] \tag{2.33}$$

Gebruikmakend van deze resultaten kunnen we nu de nieuwe doelwaarde $\overline{y}_*$ en haar variantie ($\mathbb{V}[y_*]$) berekenen, respectievelijk via Vergelijking 2.29 en 2.30:

$$\overline{y}_* = K_* K^{-1} y = 5{,}52 \tag{2.34}$$

$$\mathbb{V}[y_*] = K_{**} - K_* K^{-1} K_*^T = 0.31 \tag{2.35}$$

Selectie van de hyperparameters

Zoals al werd aangehaald zijn de parameters van de kernelfunctie van uiterst belang voor een goede fitting van de data. Deze parameters worden de hyperparameters van het GP genoemd. Het leren van een model komt dan praktisch ook overeen met leren van de geschikte hyperparameters. Wanneer we een radiale basisfunctie als kernelfunctie veronderstellen kunnen we deze hyperparameters voorstellen als de vector θ,

$$\theta = \{l, \sigma_f^2, \sigma_n^2\} \tag{2.36}$$

waarbij l de karakteristieke lengteschaal, σ_f^2 de signaalvariantie en σ_n^2 de ruisvariantie is.

Bij GPR worden deze parameters geleerd uit de trainingsdata. Algemeen wordt een Bayesiaans model toegepast om deze af te leiden, namelijk de *marginal likelihood maximization*-methode. Volgens de regel van Bayes kunnen we de posterior waarschijnlijkheid van de parameters als volgt berekenen:

$$p(\theta|X, y) = \frac{p(y|X, \theta)p(\theta)}{p(y|X)} \tag{2.37}$$

$p(y|X, \theta)$ is de marginale waarschijnlijkheid die gemaximaliseerd moet worden en $p(\theta)$ is de prior waarschijnlijkheid van de parameters.

De marginale waarschijnlijkheid kunnen we berekenen door het marginaliseren van de integraal van de waarschijnlijkheid en de Gaussiaanse prior, over de latente functie f.

$$p(y|X, \theta) = \int p(y|f, X)p(f|X)df \tag{2.38}$$

waarin zowel $p(y|f, X)$ en $p(f|X)$ normaal verdeeld zijn. Het logaritme van de marginale waarschijnlijkheid geeft de parameters zoals in Rasmussen en Williams [8]:

$$\log\ p(y|X, \theta) = -\frac{1}{2} y^T K^{-1} y - \frac{1}{2} \log |K| - \frac{n}{2} \log 2\pi \tag{2.39}$$

De eerste term $-\frac{1}{2} y^T K^{-1} y$ stelt de data-fit voor, de tweede term $-\frac{1}{2} \log |K|$ is een complexiteitsstraf en de laatste term $-\frac{n}{2} \log 2\pi$ is een normalisatieconstante.

Algoritme

Ter afsluiting van deze sectie over Gaussian process regressie geven we nog het volledige gebruikte Algoritme 1. Dit algoritme maakt gebruik van de Cholesky decompositie, zodat de kostelijke berekening voor het inverteren van de matrix K niet moet worden uitgevoerd. Deze methode is dus sneller en tevens ook numeriek stabieler dan het berekenen van de inverse matrix K^{-1}. De complexiteit van de Cholesky decompositie is $n^3/6$ en de complexiteit van het oplossen van het triangulaire systeem op lijn 2 en voor elk testgeval op lijn 4 is $n^2/2$ [8]. Merk op dat de signaal variantie (σ_f^2) en de lengteschaal l niet expliciet gebruikt worden in dit algoritme maar vervat zit in de covariantiefunctie k.

Algoritme 1 Gaussiaanse proces regeressie algoritme

INPUT: X (*inputs*), y (*targets*), k (*covariance function*),
σ_n^2 (*noise level*), σ_f^2 (*signal variance*), l (*length scale*),
x_* (*test input*), $m(X)$ (*mean function*)

$L \leftarrow cholesky(K + \sigma_n^2 I)$
$\alpha \leftarrow L^T \setminus (L \setminus y)$
} Posterior berekenen

$\overline{y}_* \leftarrow m(X) + k_*^T \alpha$
$v \leftarrow L \setminus k_*$
$\mathbb{V}[y_*] \leftarrow k(x_*, x_*) - v^T v$
$\log\ p(y|X) \leftarrow -\frac{1}{2} y^T \alpha - \sum_i logL_{ii} - \frac{n}{2} \log 2\pi$

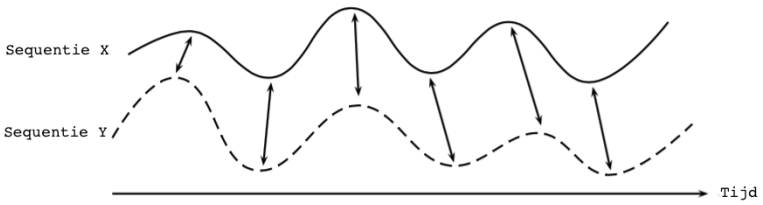
RETURN: $\overline{y}_*$ (*mean*), $\mathbb{V}[y_*]$ (*variance*),
$\log p(y|X)$ (*log marginal likelihood*)

2.2 Clustering

Omdat onze voorgestelde clustermethode gebaseerd op Gaussiaanse proces regressie (GPRC) ontworpen is voor het clusteren van tijdreeksen zullen we ons vooral richten op de achtergrond van clustertechnieken voor tijdreeksen.

2.2.1 Dynamic time warping

Binnen het domein van tijdreeksanalyse is dynamic time warping (DTW) een populaire techniek voor het meten van de gelijkaardigheid van twee tijdreeksen [12]. Voor de beschrijving van deze techniek volgen we de uiteenzetting van Müller [13]. DTW heeft als doelstelling het vergelijken van twee temporele sequenties $X = \{x_1, x_2, \ldots, x_N\}$ van lengte N en $Y = \{y_1, y_2, \ldots, y_M\}$ van lengte M (Figuur 2.6). De intuïtie achter DTW is dat het in de meeste gevallen niet correct is om de punten van dezelfde tijdstempels te vergelijken, maar dat deze eerst gealiëneerd moeten worden.



Figuur 2.6: DTW alignatie van twee tijdsafhankelijke sequenties

Door het evalueren van de lokale kost voor elk paar van elementen van de sequenties X en Y, bekomen we de kostmatrix $C \in \mathbb{R}^{N \times M}$ gedefinieerd door $C(i, j) = c(x_i, y_j)$. In deze matrix kunnen we het pad vinden dat de totale kost minimaliseert en de beste alignatie van de twee tijdreeksen beschrijft. Dit pad wordt het optimale *warping* pad tussen sequentie X en Y genoemd. Formeel:

Definitie 3. Een (N, M)-*warping* pad (of kortweg het *warping* pad) is een sequentie $p = (p_1, \ldots, p_L)$ met $p_l = (n_l, m_l) \in [1 : N] \times [1 : M]$ *voor* $l \in [1 : L]$ waarbij volgende drie condities zijn voldaan:

1. Grensconditie: $p_1 = (1, 1)$ *en* $p_L = (N, M)$.
2. Monotoniciteitsconditie: $n_1 \leqslant n_2 \leqslant \ldots \leqslant n_L$ *en* $m_1 \leqslant m_2 \leqslant \ldots \leqslant m_L$.
3. Stapgrootte-conditie: $p_{l+1} - p_l \in \{(1, 0), (0, 1), (1, 1)\}$ *voor* $l \in [1 : L - 1]$.

De totale kost van een *warping* pad tussen X en Y rekening houdend met de lokale kost c wordt gedefinieerd als

$$c_p(X, Y) = \sum_{l=1}^{L} c(x_{n_l}, y_{m_l}). \tag{2.40}$$

Een optimaal *warping* pad tussen X en Y is een pad p^* met minimale kost.
De DTW afstand, DTW(X,Y) tussen X en Y is dan gedefinieerd als de totale kost van p^*.

$$\begin{aligned} DTW(X,Y) &= c_{p*}(X,Y) \\ &= min\{c_p(X,Y) \mid p \textit{ is een } (N,M)\textit{-warping pad}\} \end{aligned} \tag{2.41}$$

Deze DTW afstand kunnen we gebruiken als een maatstaf voor de gelijkaardigheid van tijdreeksen. Praktisch is dit algoritme gebaseerd op *dynamic programming*. De $N \times M$ geaccumuleerde kostmatrix D ($D(n,m) = DTW(X(1:n), Y(1:m))$ wordt via volgende regels berekend:

- $D(n,1) = \sum_{k=1}^{n} c(x_k, y_1) \mid n \in [1:N]$,
- $D(1,m) = \sum_{k=1}^{m} c(x_1, y_k) \mid m \in [1:M]$,
- $D(n,m) = c(x_n, y_m) + min\{D(n-1,m-1), D(n-1,m), D(n,m-1)\} | 1 < n \leq N \textit{ en } 1 < m \leq M$

Als kostfunctie c wordt meestal de Euclidische kost gebruikt. De complexiteit van deze methode is $O(N \times M)$ ofwel $O(N^2)$ als we twee tijdreeksen van gelijke lengte beschouwen. Wanneer we S tijdreeksen van gelijke lengte N willen vergelijken, zullen $S \times S/2 - S = S(S-2)/2$ unieke paren onderzocht moeten worden. Dit resulteert in een totale complexiteit van $O(S^2N^2)$. Sectie A.2 toont de code van het volledige algoritme. Hierin is w een parameter die de *window*-grootte verkleint. De *window*-grootte is een extra globale beperking voor de toelaatbare *warping* paden. Grafisch kunnen we dit voorstellen als een toelaatbare band rond de diagonaal van de kostmatrix, waarin het optimale *warping*-pad gezocht kan worden. Hoewel dit een versnelling is voor het DTW-algoritme, kan het zijn dat door deze beperking het meest optimale *warping* pad niet gevonden wordt (omdat het gedeeltelijk buiten deze band valt). We zullen deze beperking dan ook niet gebruiken in deze thesis ($w = 0$). Voor meer informatie verwijzen we u naar [13, 14]. Deze methode zal door de clustermethoden in de volgende secties gebruikt worden als afstandsfunctie (maatstaf voor de gelijkvormigheid) voor de te clusteren tijdreeksen.

2.2.2 K-medoids clusteren met DTW

K-medoids is een populaire vlakke clustertechniek die een dataset van S datapunten verdeelt in k clusters. Met een vlakke clustering bedoelen we een clustering die de gegevensverzameling onderverdeelt in clusters zonder een expliciete structuur te tonen van hoe de clusters gerelateerd zijn [15].
K-medoids is zeer gelijkaardig aan de bekende *k-means* methode maar kan in tegenstelling tot *k-means* met arbitraire afstandsfuncties werken. Intuïtief is *k-means* een iteratieve clustermethode om S observaties te verdelen in k clusters, waarbij k observaties worden gekozen waar rond de meest gelijkaardige buren worden gegroepeerd. Merk op dat in dit werk de datapunten van de dataset tijdreeksen zijn. Om deze reden is het belangrijk dat gebruikte clustermethode compatibel is met de DTW

afstandsfunctie.
De *Partitioning Around Medoids* (PAM) [16] implementatie van *k-medoids* is één van de meest gebruikte en wordt ook in dit werk gebruikt. PAM gaat algemeen als volgt te werk:

1. Initialiseer k centrums (*medoids*) door willekeurig k datapunten uit de totale dataset van S datapunten te selecteren.

2. Associeer elk datapunt met de dichtst bijzijnde *medoid*, gebruikmakend van de gekozen afstandsfunctie.

3. Zolang de kost van de configuratie verkleint:

    a) Voor elke *medoids* m en voor elk datapunt o

        i. Wissel m en o om en herbereken de kost. (som van de afstanden van de datapunten tot hun *medoids*.)

        ii. Als de totale kost van de configuratie is vergroot in de vorige stap, wordt de omwisseling ongedaan gemaakt.

Een mogelijk nadeel van deze techniek is dat het aantal clusters (k) a priori moet gespecificeerd worden en de niet-deterministische aard ervan omdat de centra initieel willekeurig worden gekozen. De complexiteit van het PAM algoritme is $O(K(S-K)^2I)$ met K het aantal clusters, S het aantal datapunten (tijdreeksen) en I het aantal iteraties tot convergentie [17]. Gebruikmakend van de DTW afstandsfunctie wordt de totale complexiteit $O(S^2N^2)$ in functie van de tijdreekslengte N en het aantal te clusteren tijdreeksen S.

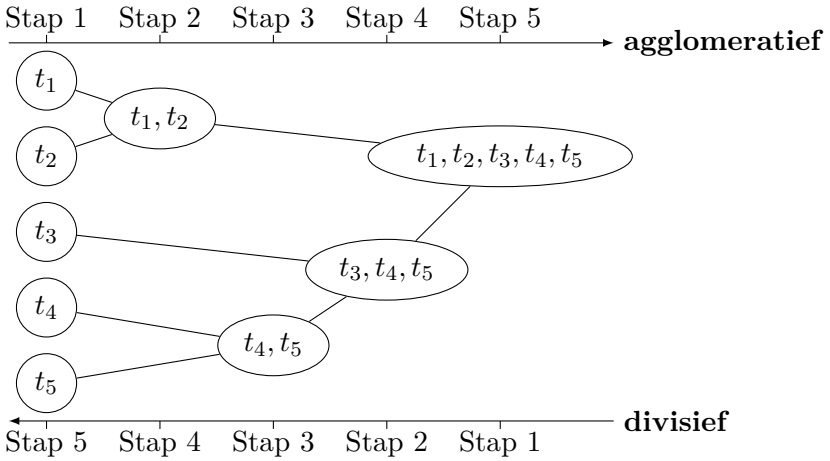
2.2.3 Hiërarchisch clusteren met DTW

Hiërarchisch clustering is een clustertechniek die een hiërarchie als clusterresultaat geeft. D.w.z dat het resultaat een groep is die onderverdeeld is in subgroepen die op hun beurt ook weer in groepen zijn onderverdeeld. Doordat deze techniek de vorming van de clusters toont in de structuur van het resultaat, zal ze meer informatie bevatten dan een resultaat dat bekomen is door een vlakke clustering. Daarnaast heeft dit algoritme ook geen nood aan het vooraf specificeren van het aantal clusters die gevormd moeten worden.
Hiërarchisch clusteren kan op een *top-down* or *bottom-up* manier gebeuren (Figuur 2.7). Bij een *top-down* aanpak zal het algoritme geïnitialiseerd worden met de totale dataset en zal deze recursief opgesplitst worden totdat groepen van individuele elementen bekomen worden. Anderzijds zal een *bottom-up* aanpak beginnen met de individuele elementen en zal ze deze elementen (of agglomeraties van elementen) samenvoegen totdat één grote cluster gevormd wordt, die alle elementen bevat [15]. Deze laatste aanpak wordt hiërarchisch agglomeratief clusteren (HAC) genoemd en wordt in deze thesis gebruikt. Een eenvoudige beschrijving van dit algoritme maakt gebruik van S data-elementen die geclusterd moeten worden en een $S \times S$ afstands- (of gelijkaardigheids) matrix. Een afstandsmatrix is een matrix die paarsgewijze

afstanden tussen alle elementen bevat. In dit werk zal de DTW afstand (Sectie 2.2.1) gebruikt worden om deze matrix op te stellen. Het volledige algoritme wordt beschreven door:

1. Wijs elk element aan een eigen cluster toe zodat S clusters bekomen worden. De afstand tussen deze clusters kan opgezocht worden in de afstandsmatrix.
2. Vind het meest gelijkaardige paar van clusters en voeg deze samen tot een nieuwe cluster.
3. Bereken de afstand (gelijkaardigheid) tussen de nieuwe cluster en elke andere oude cluster.
4. Herhaal stap 2 en 3 totdat alle elementen geclusterd zijn tot één enkele cluster van grootte S.

Merk op dat in stap 2 van het algoritme de afstand tussen twee clusters moet bepaald worden. Wanneer er meerdere elementen in een cluster zitten, moet er een criterium zijn voor hoe we deze afstand gaan bepalen. Dit criterium is het link criterium. De twee meest gebruikte zijn de *complete-link* en de *single-link*. Bij *complete-link* clustering zal de gelijkaardigheid van twee clusters bepaald worden door de gelijkaardigheid van hun meest ongelijkaardige leden. Bij *single-link* wordt de gelijkaardigheid bepaald door de gelijkaardigheid van hun meest gelijkaardige leden [15]. Dit laatste wordt toegepast in dit werk.
De complexiteit van deze methode is $O(S^2)$ waarbij S het aantal te clusteren elementen (tijdreeksen) zijn [18, 19]. Het berekenen van de afstand met DTW heeft een complexiteit van $O(N^2)$ met N de lengte van de tijdreeksen. De totale complexiteit van deze clustermethode voor tijdreeksen is dus $O(S^2N^2)$.



Figuur 2.7: Hiërarchisch clusteren: divisieve of agglomeratieve aanpak

Hoofdstuk 3

Relevante literatuur

3.1 Voorspelling

In deze sectie geven we een overzicht van de bestudeerde werken m.b.t. voorspellingsmethoden. De nadruk ligt hierbij op het voorspellen van energieconsumptie. Tabel 3.1 toont de opdeling van de onderzochte methoden. Het hoofdstuk is dan ook opgedeeld volgens deze structuur in twee secties: Sectie 3.1.1 bespreekt het onderzoek gaande over niet-auto-regressieve methoden en Sectie 3.1.2 geeft een overzicht van het onderzoek m.b.t. de auto-regressieve methoden. De vergelijking van de verschillende methoden onderling is vooral gebaseerd op hun gemiddelde fouten. In de laatste sectie 3.1.3 wordt nog een samenvatting gegeven waarbij de twee beste methoden per werk worden opgelijst.

Voorspellingsmethoden	
niet-auto-regressieve methoden	**auto-regressieve methoden**
Lineaire regressie (LR)	*Autoregr. integrated moving average* (ARIMA)
Support Vector Regressie (SVR)	Gaussiaanse proces regressie (GPR)
Random Forest Regressie (RFR)	
Artificial Neural Networks (ANN): Multilayer Perceptron (MLP) Radiale Basisfuncties (RBF)	

Tabel 3.1: Overzicht van de belangrijkste voorspellingsmethoden in de literatuurstudie

3.1.1 Niet-auto-regressieve methoden

In deze sectie worden de niet-auto-regressieve methoden besproken die gebruikt worden voor het voorspellen van energieconsumptie, dit zijn vooral methoden die gebaseerd zijn op regressie.

Lineaire regressie

Volgens het overzicht van de voorspellingstechnieken voor elektriciteitsconsumptie van A. K. Singh [20], is lineaire regressie (LR) nog steeds de meest gebruikte statistische methode. Dit komt vooral door haar relatief eenvoudige implementatie. Een veel gebruikte regressiemethode is de kleinste kwadraten methode. Zoals besproken in 2.1.2 is het doel van deze methode de beste benaderende lijn te vinden doorheen de beschikbare datapunten, zodat de kwadratische fout minimaal is. In een groot deel van de literatuur wordt deze methode als een referentiemethode gekozen om de geteste methode mee te vergelijken. In een overzicht van B. Yan et al. [21] wordt deze methode als referentie gebruikt voor o.a. support vector regressie (SVR) en Gaussiaanse proces regressie (GPR) voor het voorspellen van dagelijkse energieconsumptie. Hierbij merken we dat LR in dit geval gemiddeld iets minder goed voorspelt dan SVR en GPR. Ook in het onderzoek van K. Kandananond [22] wordt LR als referentiemethode gebruikt. Het onderzoek vergelijkt het gebruik van artificial neural networks (ANN) en *autoregressive integrated moving average* (ARIMA) met LR m.b.t. het voorspellen van de elektriciteitsconsumptie in Thailand. Uit de resultaten blijkt dat ANN de beste techniek is, gevolgd door ARIMA en als laatste LR.
Uit het onderzoek van de voorgaande thesis [1] bleek dat voor beide casestudies LR de mist accurate methode was voor het voorspellen van de elektrische energieconsumptie van kantoren. In het werk van H. Wei-chiang [23] wordt LR nogmaals vergeleken met de SVR voor energieconsumptievoorspelling. In dit werk wordt ook geconcludeerd dat LR inferieur is aan de SVR in deze gevallen. Toch kan het zijn dat LR in bepaalde gevallen beter voorspelt dan complexere methoden, dit is namelijk zo in het werk van Z. Jianwu et al [24]. Dit werk handelt over het voorspellen van zonne-energie gebruikmakend van RBFN (radiale basisfuncties netwerk). Hierin is de beste voorspellingsmethode RBFN maar op de voet gevolgd door LR, dat het beter doet dan een AR-model (auto-regressief model).

Support vector regressie

Vapnik [25] introduceerde als eerste support vector machines (SVM), dit is een krachtige methode uit *machine learning* die gebaseerd is op statistisch leren. Het idee erachter is dat input variabelen naar een hogere dimensie worden getild waarin vervolgens lineaire regressie wordt uitgevoerd. Hierna wordt het resultaat terug getransformeerd naar de beginruimte, waardoor we een niet-lineaire functie bekomen. Initieel werden SVM voor classificatie gebruikt maar later ook voor regressie (SVR) [6].
Support vector machines behoren tot de top drie van meest gebruikte methoden voor energievoorspellingen volgens het werk van K. Kandananond [26] en in een onderzoek van P. F. Pai [27] wordt geconcludeerd dat SVM beter presteert dan ARIMA of ANN voor het voorspellen van het regionale elektriciteitsverbruik in Taiwan. In het werk van B. Yan et al. [21], dat energievoorspelling uitvoert op gebouwen van de Harvard campus, wordt als groot voordeel van de methode de aanpasbaarheid van de parameters beschreven, maar als nadeel wordt de grote

rekenkost vermeld. Nochtans wordt in dit werk bevonden dat de SVR de tweede beste methode is, na GP en voor LR en RBF. In het onderzoek van Y. Yongquan [28] naar het voorspellen van de consumptie van een webserver, werd SVR vergeleken met ARIMA en multilayer perceptron (MLP-ANN), en was het de minst accurate methode van deze drie. Uit het onderzoek in voorgaande thesis rond hetzelfde onderwerp [1] bleek dat voor beide casestudies SVR de beste methode was voor het voorspellen van de elektrische energieconsumptie van kantoorgebouwen, in deze studie werden Gaussiaanse processen niet vergeleken.

Artificial neural networks: multilayer perceptron (MLP) en radial basis function (RBF)

In deze sectie worden twee artificial neural network methoden besproken. Enerzijds de Multilayer perceptron (MLP) en anderzijds het radiale basisfunctie netwerk (RBFN). Een MLP is een feedforward artificial neural network dat een set van inputdata mapt op een set van geschikte outputs. Om een RBFN te bekomen wordt in een MLP de verborgen tussenlaag vervangen door Radiale basis functies. Dit geeft als voordeel dat een RBFN minder gevoelig is voor locale minima ten opzichte van een MLP.
In het werk van K. Kandananond [26], waarin de elektriciteitsconsumptie van Thailand wordt voorspeld, worden MLP, ARIMA en LR methoden vergeleken. Het werk vergelijkt de nauwkeurigheid van de verschillende methoden en concludeert dat de MLP iets beter presteert dan het ARIMA model en dat MLP significant beter presteert dan LR.
Het werk van Z. Jiangwu [24] stelt een RBFN voor als voorspellingsmethode voor het voorspellen van zonne-energie. De prestatie van deze methode wordt vergeleken met deze van een Auto-regressief (AR) model en een lineair regressie (LR). Het RFBN overtroeft LR licht en AR zwaar.

Random forest regressie

Als laatste deel van de niet-auto-regressieve methoden wordt Random forest regressie besproken, dit is een data-mining methode die gebruikt wordt voor zowel classificatie als regressie en werd voor het eerst geïntroduceerd door Breiman [29]. Terwijl een klassieke classificatieboom gebruik maakt van de beste splitsing tussen alle variabelen bij elke knoop, zal een random forest algoritme de beste splitsing uitvoeren tussen een deelset van willekeurig gekozen variabelen bij een knoop [1]. Deze methode is dus gebaseerd op het combineren van de resultaten van verschillende beslissingsbomen. Omdat in deze thesis deze methode maar kort vergeleken wordt in de literatuurstudie, gaan we hier niet dieper op in.
Het werk van B. Yan [21], dat energievoorspelling uitvoert op gebouwen van de Harvard campus, besluit dat GP veruit het best presteren. Hoewel RFR een stevige gedeelde tweede plaats heeft met SVR, moet dit resultaat toch sceptisch bekeken worden. Voor de RFR werd een veel grotere trainingsset en een kleinere testset gebruikt dan voor de GP en SVM. Uit het onderzoek van voorgaande thesis [1] bleek dat voor de eerste casestudies RFR de tweede minst accurate methode uit vier voor

het voorspellen van de elektrische energieconsumptie was. Voor de tweede casestudie was RFR de tweede beste methode, wat dus een gemiddeld resultaat betekent. Verder is het merkbaar dat het aantal werken waarin RFR wordt gebruikt relatief schaars zijn t.o.v. de andere besproken methoden.

3.1.2 Auto-regressieve methoden

In deze sectie wordt een tweede groep voorspellingsmethoden besproken. Deze methoden zijn auto-regressieve methoden die uiterst geschikt zijn om tijdreeksanalyse uit te voeren.

Autoregressive integrated moving average (ARIMA)

Binnen tijdreeksanalyse, is de *Autoregressive integrated moving average*-methode (ARIMA), een stochastische differentievergelijking [30]. De graad van het ARIMA model wordt aangeduid door het drietal (p, d, q); p is de graad van het auto-regressief model, d is de differentiegraad en q is de graad van het *moving-average*-model. Omdat in deze thesis ook deze methode maar kort vergeleken wordt in de literatuurstudie, gaan we hier niet dieper op in.
In het onderzoek van K. Kandananond [22], waarin de energieconsumptie binnen een land wordt voorspeld, wordt ARIMA overtroefd door twee ANN architecturen (MLP en RBF). Er wordt wel geconcludeerd dat de rekentijd dubbel zo groot is voor de ANN-modellen dan voor de ARIMA maar dat de ARIMA-methode eenvoudiger is. Binnen dit werk is ARIMA wel significant beter dan LR. Het werk van P. Chujai et al. [31] toont aan dat het gebruik van deze methode in het kader van huishoudelijke elektrische consumptie ook nuttig en accuraat kan zijn, spijtig wordt er in dit werk geen vergelijking gemaakt met de andere besproken methoden. Uit een laatste werk van Y. Yongquan et al. [28], dat handelt over de voorspelling van consumptie van webservers, wordt besloten dat ARIMA iets beter scoort dan SVM en significant beter scoort dan ANN.

Gaussiaanse proces regressie

De theoretische achtergrond van Gaussiaanse proces regressie werd eerder al besproken in sectie 2.1.5. In een recent werk van M. Blum [32] wordt de voorspellingsprestatie van GP getest op gesimuleerde data van elektriciteitsconsumptie en wordt het resultaat vergeleken met een *baseline* methode die het resultaat van de voorgaande week geeft als voorspelling voor deze week. De GP methode houdt hierbij rekening met arbitraire features zoals de toestand van het weer en beschouwt hierbij dagelijkse en wekelijkse periodiciteit in de data. De GP methode doet een significant betere voorspelling in het werk dan de *baseline* methode.
In een volgend werk van H. Noh [33] wordt GP vergeleken met een adaptief auto-regressief model (AAR). In dit werk wordt ook onderzoek gedaan naar het voorspellen van reële energieconsumptie van een gebouw van de Stanford Universiteit. Hierbij werd vastgesteld dat de AAR methode beter geschikt was voor zeer korte termijn voorspellingen, zoals 15 minuten of een uur op voorhand. GP is beter geschikt voor

voorspellingen die een dag in de toekomst kijken.
Reële data van een Japanse energiemaatschappij is gebruikt in het werk van M. Hiroyuki [34]. Hierin wordt een voorspelling voor de volgende dag bepaald door middel van GP en wordt deze vergeleken met SVR en MLP neural networks. De gemiddelde fout van GP is merkbaar kleiner dan deze van MLP en SVR. MLP scoort nog iets beter dan SVR maar dit verschil is gering.
In een laatste onderzoek van B. Yan [21] is reële data van een gebouw op de Harvard campus gebruikt. Er worden drie verschillende voorspellingen gedaan: de voorspelling van consumptie van elektriciteit, koud water en warm water; en verschillende voorspellingsmethoden worden vergeleken op basis van hun beschouwde features en hun voor- en nadelen. In dit werk wordt GP vergeleken met LR, SVR, RFR en k-Nearest neighbors (k-NN). K-NN scoort veel slechter dan de rest, dus wordt buiten beschouwing gelaten en wordt verder ook niet besproken in deze thesis. Het is zeer duidelijk dat GP in twee van de drie voorspellingen de meest accurate voorspelling bekomt. In het derde geval is RFR iets beter maar hierbij moet opgemerkt worden dat RFR een veel grotere trainingsset ter beschikking had. GP presteerde in dit werk zelfs goed op korte termijn voorspellingen (van een uur op voorhand). Een aangehaald voordeel van deze methode is dat ze zelf de vorm van de tijdreeks kan leren. Een aangehaald nadeel is echter dat de rekenkost relatief hoog oploopt.

3.1.3 Samenvattend overzicht

Nu alle methoden uit literatuurstudie vergeleken zijn met elkaar, kunnen we een algemeen overzicht geven van de best presterende methoden. Dit overzicht is te vinden in Tabel 3.2. We merken dat een groot deel van de literatuur Gaussiaanse proces regressie als beste methode beschouwt, het is dan ook de moeite waard om deze methode in detail uit te testen op de dataset van deze thesis. Verder zien we dat ook de neural network methoden, ARIMA en Support vector regressie goede kandidaten zijn voor de voorspellingsmethoden.

Referentie literatuur	**Beste Methode**	**Tweede beste methode**
B. Yan et al. [21]	Gaussiaanse proces regressie	support vector regressie
Z. JianWu et al. [24]	RBF neural network	Lineaire regressie
H. Noh et al. [33]	Gaussiaanse proces regressie	(adaptief autoregressief model)
M. Blum et al. [32]	Gaussiaanse proces regressie	(baseline methode)
Y. Yongquan et al. [28]	ARIMA	Support vector regressie
T. Koskela et al. [35]	Recurrent neural network	MLP Neural network
M. Hiroyuki et al. [34]	Gaussiaanse proces regressie	Support vector regressie
K. Kandananond et al. [26]	MLP neural network	ARIMA
K. Kandananond et al. [36]	Support vector regressie	MLP & RBF neural network

Tabel 3.2: Overzicht van de beste besproken methoden in de literatuur

3.2 Clustering

In deze sectie zullen we een beschrijving geven van de (recente) werken die handelen over het clusteren van tijdreeksen. We zullen beginnen met enerzijds te bespreken hoe Gaussiaanse processen al gebruikt worden en anderzijds aantonen dat het geleverd werk verschilt met het werk in deze thesis. Daarnaast bespreken we nog de werken die handelen over andere technieken voor het clusteren van tijdreeksen.

Gaussiaanse processen

Gaussiaanse processen (GP) zijn al eerder gebruikt voor clustering. Echter in deze thesis, is het de eerste keer dat ze gebruikt worden voor het leren van een model over een set van functies (tijdreeksen) om het temporele gedrag van een cluster te bepalen. Pimentel et al. [37] leert een GP nadat de set van tijdreeksen zijn geaggregeerd in één tijdreeks. Dit werk gebruikt GP voor het herkennen van (ab)normale vitale functies van patiënten.
Kim et al. [38] stelt een clustermethode voor die gebruik maakt van de variantiefunctie van Gaussiaanse processen regeressie in combinatie met een gereduceerde complete graaf strategie. Maar deze methode clustert (niet-temporele) feature vectoren in plaats van tijdreeksen.
Kumar et al. [39] stelt een afstandsfunctie voor die gebaseerd is op de assumptie van onafhankelijke Gaussiaanse modellen van data fouten. In dit werk wordt gebruik gemaakt van een hiërarchische clustermethode voor het groeperen van sequenties met bepaalde seizoenseffecten in een gewenst aantal clusters.
Het werk van Duvenaud [40] geeft een uitgebreid onderzoek van Gaussiaanse processen voor het automatisch construeren, visualiseren en beschrijven van een grote groep modellen, die bruikbaar zijn voor het voorspellen en vinden van structuren binnen tijdreeksen. De onderzochte clustermethoden in dit werk zijn ook weer enkel gericht op het gebruik van feature vectoren, niet op het gebruik van tijdreeksen.

Andere technieken voor het clusteren van tijdreeksen

Naast Gaussiaanse processen is er nog een variëteit aan andere technieken die toegepast worden voor het clusteren van tijdreeksen.
Liao et al. [41] geeft een overzicht van ruwe-data gebaseerde, feature-gebaseerde en modelgebaseerde clustertechnieken voor tijdreeksen. Elke techniek is besproken op gebied van het clusteralgoritme, de gelijkaardigheidsfunctie en het evaluatiecriterium. Het overzicht concludeert dat tot nu toe de focus lag op technieken die gebaseerd zijn op statistische (bv ARIMA) en probabilitische methode (bv Dynamische Bayesiaanse netwerken). Gaussiaanse processen worden enkel kort aangehaald, wanneer de modelgebaseerde technieken besproken worden.
Ook uit het werk van M. Espinoza et al. [42] blijkt dat voor het clusteren van

energiedata vooral statistische methoden toegepast worden. Het onderzoek van Espinoza gebeurde in samenwerking met de Belgische nationale elektriciteitsnetwerk operator ELIA en in tegenstelling tot hun werk hebben Gaussiaanse processen (zoals gebruikt in deze thesis) het voordeel dat er geen domeinkennis nodig is voor het bepalen van de optimale hyperparameters.
Een tweede overzicht van recente clustertechnieken voor tijdreeksen wordt gegeven door Rani et al [43]. Onder andere *k-means* clusteren wordt besproken met verschillende gelijkaardigheidsfuncties zoals de Euclidische afstandsfunctie. Gebruikmakend van de L^*-normen (met L^2 de Euclidische norm) is *k-means* een algemeen bruikbare methode maar bij hoog dimensionale input vectoren kan ze falen in het geven van betekinsvolle resultaten (*curse of dimentionality* [44]). De auteur toont positieve resultaten voor het clusteren van vectoren die metingen van 24 uur lang voorstellen met een intervalwaarde van 15 minuten. Er moet wel opgemerkt worden dat de trends van de tijdreeksen steeds vrij eenvoudig zijn om te onderscheiden.
Zoals besproken in Sectie 2.2.1 is dynamic time warping (DTW) [12] een populaire methode voor het vergelijken van tijdreeksen. Wanneer deze afstandsfunctie echter gecombineerd wordt met het *k-means* clusteralgoritme faalt dit algoritme in het geven van betekenisvolle resultaten. Dit komt omdat *k-means* ontworpen is om de variantie te minimaliseren en niet een arbitraire afstandsfunctie als DTW. *K-means* zal dus een gemiddelde van de tijdreeksen proberen te nemen maar omdat ze gedeeltelijk verschoven kunnen zijn zal dit meestal geen goed resultaat geven [45].
Een recente methode *k-shape*, voorgesteld door Paparrizos et al. [46] is een zwaartepuntgebaseerde clustermethode. Het gebruikt kruis-correlatie als gelijkaardigheidsfunctie. Het overtreft alle (niet-)schaalbare partitionerende-, hiërarchische en spectrale clustermethoden op gebied van nauwkeurigheid, met als uitzondering *k-medoids* met DTW. Deze methode geeft gelijkaardige resultaten. Volgens de auteur is *k-shape* wel beter schaalbaar.
Uit de literatuur kunnen we besluiten dat beste vorm-gebaseerde clustermethoden, partitionerende methoden zijn, die compatibel zijn met een schaal- en schuif-invariante afstandsfunctie. Uit deze soort methoden is *k-medoids* de meest populaire omdat het eenvoudig toelaat een vorm-gebaseerde afstandsfunctie te gebruiken.

Hoofdstuk 4

Voorspellen met Gaussiaanse processen

Dit hoofdstuk geeft de lezer een overzicht van de stappen die we doorlopen wanneer we Gaussiaanse processen gebruiken voor het voorspellen van de energieconsumptie. Omdat het stappenplan gelijkaardig is aan werk van O. De Somer en T. Kutz [1], zijn bepaalde secties van dit hoofdstuk gebaseerd op hun werk. Sectie 4.1 beschrijft hoe we de data onderzoeken, Sectie 4.2 beschrijft hoe we de data gaan manipuleren voor het voorspellen en Sectie 4.3 bespreekt de eigenlijk voorspelling en evaluatie-criteria.

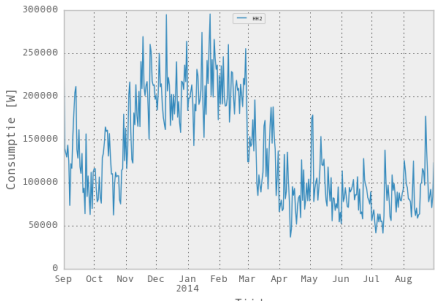
4.1 Data exploratie

Een eerste belangrijke stap in een data-analyse is het bestuderen van de data. Het is de bedoeling om een duidelijk zicht te krijgen welke features belangrijk kunnen zijn voor de voorspelling. Daarnaast kunnen we ook al trends, cyclussen en andere interessante observaties ontdekken, zodanig dat we later hiernaar kunnen terugkoppelen wanneer we vreemde voorspellingen waarnemen. De gebruikte data is *real-life* ruwe data, die we moeilijk volledig kunnen doorlopen zonder gebruik te maken van enkele hulpmiddelen. We starten met maken van grafieken van de data waardoor we nuttige inzichten verwerven die we in de volgende stappen kunnen gebruiken.

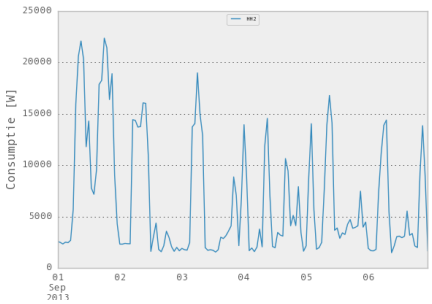
4.1.1 Tijdgrafiek

Omdat deze thesis tijdreeksen behandelt, gaan we tijdgrafieken creëren van de data. Op deze manier kunnen we dag-, week- en jaarpatronen detecteren. Zoals in Sectie 2.1.1 besproken is, zijn we op zoek naar trends, seizoenen en cyclussen. Bij het selecteren van de voorspellingsmethode, is het belangrijk dat het model deze patronen kan leren. Figuren 4.1 en 4.2 tonen voorbeelden van tijdgrafieken van de elektrische consumptie gedurende verschillende perioden en *sample frequenties*. De grafiek in Figuur 4.1 toont het verbruik gedurende één jaar. Hieruit kunnen we bijvoorbeeld al leren dat de consumptie gedurende de wintermaanden hoger ligt, dus een temperatuur feature zal zeker nuttig zijn. De grafiek in Figuur 4.2 toont het verbruik gedurende

een week per uur, door het terugkerend patroon kunnen we hieruit leren dat een tijd feature dat het uur aanduidt, ook nuttig zal zijn.



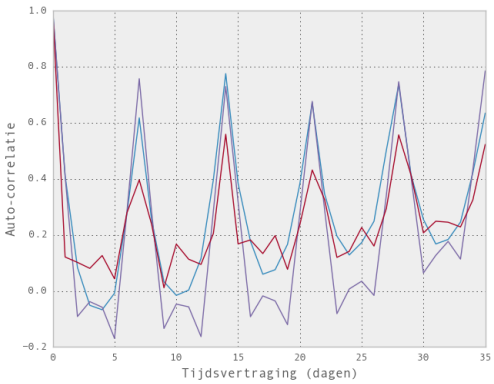
Figuur 4.1: Tijdgrafiek van consumptie data van één huishouden gedurende één jaar per dag.



Figuur 4.2: Tijdgrafiek van consumptie data van één huishouden gedurende één week per uur.

4.1.2 Auto-correlatiegrafiek

Naast deze tijdgrafieken maken we ook gebruik van auto-correlatiegrafieken. Deze worden vooral gebruikt voor het vinden van de periode voor de vertraagde variabelen. Omdat in deze thesis gebruik wordt gemaakt van de auto-regressieve GP methode hoeven we dit niet expliciet zelf te onderzoeken. Toch zullen we het effect van deze variabele onderzoeken en er conclusies uit trekken. Figuur 4.3 toont enkele auto-correlatiegrafieken waarbij de x-waarde van een piek overeen komt met de vertragingen van de consumptievariabele waarvoor de auto-correlatie groot is. Zoals we zien is er bv. een sterke correlatie tussen de huidige energieconsumptie en die van een week geleden.



Figuur 4.3: Auto-correlatiegrafiek van de consumptie van de toestellen voor enkele huishoudens.

4.2 Data preprocessing

Data preprocessing is een belangrijke voorbereidende stap wanneer we ruwe data willen gebruiken voor het maken van voorspellingen. De ruwe data zal getransformeerd worden naar een handelbaar formaat. *Real-life* data is vaak incompleet, inconsistent en bevat vaak fouten. De belangrijkste stappen tijdens de preprocessing zijn dan ook het opschonen, integreren en transformeren van de data en tenslotte de feature selectie.

4.2.1 Data cleaning

Data cleaning houdt zich bezig met ontbrekende data en vreemde waarden in de dataset. Hoe we deze gevallen behandelen hangt af van de hoeveelheid beschikbare data. Wanneer we voldoende data hebben, is de eenvoudigste oplossing de observaties met ontbrekende datapunten te negeren of te verwijderen. Een andere techniek is het interpoleren van de ontbrekende datapunten. In de data van deze thesis waren er weinig ontbrekende datapunten, zodat we de interpolatie techniek gemakkelijk konden toepassen.

4.2.2 Data integratie

In praktijk is de data vaak afkomstig van verschillende bronnen. Data-integratie is het samenvoegen van deze data op een coherente manier. Hierbij moet rekening worden gehouden met de mogelijke verschillende *sample frequenties*. Dit is het aantal keer dat er per tijdseenheid een meting wordt uitgevoerd. In deze thesis moeten we de consumptiedata die per kwartier wordt gemeten, samenvoegen met de temperatuurdata die per uur werd opgemeten. Het verwijderen van alle extra metingen zal een vertekend beeld geven van de consumptie omdat we dan $\frac{3}{4}$ van de data zouden weggooien. Het aggregeren van deze data is hiervoor een oplossing en werd ook gebruikt in deze thesis.
Interpolatie of het opvullen van ontbrekende waarden zijn hier nog andere technieken voor maar worden niet verder besproken in de thesis omdat deze ook niet gebruikt werden in dit werk.

4.2.3 Schalen van features

Voor bepaalde voorspellingsmethoden wordt het aangeraden de data te manipuleren zodat alle features gelijkaardige schaal hebben. Op deze manier verzekeren we dat alle features evenveel kunnen bijdragen tot het leren van het model. Omdat de features meestal in vectorvorm aangeboden worden aan het voorspellingsalgoritme zou een feature dat een grootte-orde groter is dan al de andere, dominant kunnen zijn. Om dit op te lossen kunnen we een herschaling of een normalisering van de data doorvoeren.
De eerste methode impliceert een herschaling van de features zodat deze allemaal in hetzelfde gebied liggen, meestal $[0, 1]$ of $[-1, 1]$. Vergelijking 4.1 toont hoe we de

features herschalen naar $[0, 1]$.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{4.1}$$

Normalisatie zal de features transformeren zodat elke feature een nul-gemiddelde en een standaard variantie heeft. Dit gebeurt door het gemiddelde van de feature af te trekken en deze uitkomst te delen door de standaard afwijking van deze feature:

$$x' = \frac{x - \mu}{\sigma} \tag{4.2}$$

met μ het gemiddelde en σ de standaardafwijking van de overeenkomstige feature. Dit noemt men ook de *Z-Score* transformatie en wordt gebruikt in deze thesis.

4.2.4 Feature selectie

Feature selectie is het defiëren van de set van features die belangrijk zijn voor de voorspelling. Deze feature set of feature vector zal als input dienen voor het voorspellingsalgoritme. Initieel starten we van een set van opgemeten waarden die een verband hebben met te voorspellen waarden. In dit werk zullen we gebruik maken van exogene data zoals de hoeveelheid zonneschijn (irradiatie) en de buitentemperatuur. Doordat we gebruikmaken van een auto-regressieve methode (GP) zullen we niet zoals in het vorige werk [1] expliciet gebruik maken van een afgeleide vertraagde variabele zoals de waarden van de elektrische energie consumptie (EEC) van enkele dagen geleden (enkel ter vergelijking). Algemeen is het GP in staat deze informatie zelf te leren maar deze informatie kan de methode extra helpen. Dit is een belangrijk verschil met het werk van O. De Somer en T. Kutz [1].
Naast deze informatie is het belangrijk dat we de temporele informatie in de feature set verwerken. Dit doen we door variabelen zoals uur van de dag, dag van de week, feestdagen e.d. toe te voegen. Tabel 4.1 toont een voorbeeld van een feature set. De eerste kolom bevat de tijdstempel van elke observatie. De tweede kolom bevat de EEC of doelvariabele, dit is de waarde die we willen voorspellen. De volgende vijf kolommen bevatten de afgeleide voorspellers en de laatste twee kolommen de exogene voorspellers.

Tabel 4.1: Voorbeeld van een feature vector

	Doel var.	Afgeleide voorspellers					exogene voorspellers	
TIJD	EEC[W]	EEC_{-1D}	Dag_{Jaar}	Dag_{Week}	Nr_{week}	Uur_{Dag}	T [°C]	IR $[Wb/m^2]$
4/01/2014 08:00	1500	NaN	4	5	1	8	4	50
4/01/2014 09:00	2000	1500	4	5	1	9	5	60
4/01/2014 10:00	1800	2000	4	5	1	10	6	70

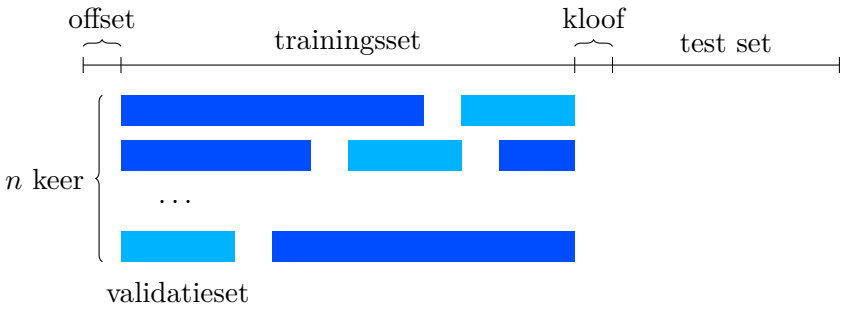
Deze tabel bevat alle features die we beschouwen. We kunnen deze nog inkorten door een deelset uit deze feature vector te nemen. Afhankelijk van de gebruikte methode zal het weglaten van features al dan niet een betere voorspelling geven, dit kan best empirisch getest worden. Merk wel op dat een vermindering van het aantal features ook de looptijd van het algoritme zal verminderen.

4.3 Consumptievoorspelling

Na de preprocessing van de dataset, gaan we de consumptie voorspellen gebruikmakend van een voorspellingsmethode. In dit werk focussen we ons op Gaussiaanse proces regressie. Support Vector regressie is een tweede methode die aan bod komt ter vergelijking. De volgende secties bepreken de trainings- en evaluatiemethoden.

4.3.1 Trainings- en testset

De meeste voorspellingsmethoden binnen statistiek of *machine learning* vereisen training a.d.h.v. historische data om een model uit te leren, voordat ze gebruikt kunnen worden. Een gangbare praktijk is het splitsen van de dataset in een trainingsset en een onafhankelijke testset. Zo kunnen de modellen die getraind zijn op de trainingsset gevalideerd worden op de testset. Op deze manier kan onderzocht worden hoe het model de patronen in de trainingsset kan generaliseren. Hierbij is het belangrijk dat er geen *overfitting* is, dit is het geval waarbij het model goede resultaten bekomt bij trainingsset maar niet bij nieuwe data. De basisassumptie voor deze procedure is dat de trainings- en testset identiek verdeeld en onafhankelijk zijn. Op deze manier lekt er geen informatie van de trainingsset in de testset.
Omdat we met tijdreeksen werken, zullen de datasets temporeel zijn i.p.v. onafhankelijke samples. Bijvoorbeeld in een dataset van vier weken beschouwen we de eerste drie weken als trainingsset en de laatste week als testset. Echter, zoals besproken in Sectie 4.2 kan het zijn dat we gebruik maken van een vertraagde variabele feature. Stel dat het feature een vertraging van twee dagen ($V = 2$) heeft, dan moeten we opletten dat de trainings- en testset niet afhankelijk van elkaar worden. Deze afhankelijkheid kunnen we vermijden door een kloof ter grootte van de (grootste) vertraagde periode $V = 2$ te voorzien tussen beide. De eerste V elementen van de trainingsset zullen geen waarde hebben voor de vertraagde variabele omdat deze data niet beschikbaar is, daarom wordt er een offset gebruikt zodat deze overgeslagen worden. Figuur 4.4 toont de splitsing van de dataset.



Figuur 4.4: Splitsing van de dataset in de trainings- en testset. De trainingsset wordt n keer gesplitst in een subtrainings- en validatieset voor het optimaliseren van de hyperparameters [1]

4.3.2 Hyperparameter optimalisatie

Het trainen van een model komt overeen met het leren van de parameters. In dit werk maken SVR en GP gebruik van hyperparameters die geleerd moeten worden uit de trainingsdata (respectievelijk Sectie 2.1.3 en 2.1.5). Omdat niet alle toolboxen die gebruikt worden in dit werk automatische hyperparameteroptimalisatie voorzien, maken we gebruik van *n-fold cross-validation* waarbij de trainingsset n keer opnieuw opgesplitst wordt in een validatieset en een subtrainingsset (Figuur 4.4). Ook hier zullen we weer rekening moeten houden met de onafhankelijkheidseis tussen trainings - en testset, daarom zullen ook hier weer kloven worden gebruikt (Sectie 4.3.1). Deze methode wordt de *modified* of *h-block* kruis-validatie genoemd [47, 48]. De modellen zullen daarna vergeleken en geselecteerd worden op basis van hun gemiddelde performantie door bijvoorbeeld hun gemiddelde absolute fout (Sectie 4.3.3) van de validatiesets te vergelijken.
Een nadeel van de beschreven kruis-validatie methode is echter dat we veel trainingsdata verliezen als de vertraagde periode groot is. Omdat de beschikbare dataset in deze thesis beperkt is zullen we de vertraagde periode tot twee dagen beperken.

4.3.3 Evaluatie van de predictie

Na het optimaliseren van de hyperparameters van het model, is de voorspellingsmethode in staat om ongeziene data te voorspellen. Een evaluatie hiervan kan gedaan worden door het model toe te passen op de onafhankelijke testset en de afwijking van de voorspelling t.o.v. de verwachte waarden op te meten. In deze laatste sectie definiëren we de metrieken die we in dit werk gebruiken om de kwaliteit van de voorspelling te evalueren.
We maken hiervoor gebruik van de relatieve gemiddelde fout. Deze is gebaseerd op de gemiddelde absolute fout, gedefinieerd in Vergelijking 4.3 met p_i de voorspelling en y_i de verwachte waarden op tijdstip i.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|p_i - y_i| \tag{4.3}$$

Om de relatieve fout te berekenen maken we gebruik van Vergelijking 4.4. De gemiddelde absolute fout wordt gedeeld door de gemiddelde verwachte waarde en wordt omgezet naar een percentage. Via deze metriek kunnen vergelijkingen worden gedaan over verschillende datasets heen.

$$MRE = \frac{MAE}{\overline{y}} * 100\% \tag{4.4}$$

In Sectie 6.1 zal de volledige voorspellingsmethode toegepast worden op een dataset van 71 huishoudens. Hierbij zullen alle stappen van dit hoofdstuk in de praktijk worden toegepast en het resultaat hiervan zal geëvalueerd worden aan de hand van de metrieken besproken in deze sectie.

Hoofdstuk 5

Clusteren met Gaussiaanse processen

In dit hoofdstuk tonen we de lezer hoe we Gaussiaanse proces regressie (GPR) kunnen gebruiken voor het clusteren van tijdreeksen. Hiervoor beschrijven we eerst in Sectie 5.2 hoe we GPR kunnen aanpassen om een algemeen model over verschillende tijdreeksen te leren. Vervolgens tonen we in Sectie 5.3 hoe we gebruikmakend van dit model een clustermethode kunnen construeren. Ter afsluiting bespreken we nog de complexiteit in Sectie 5.4.

5.1 Set van functies naar één functie

Vooraleer we onze unieke GPR-gebaseerde clustermethode (GPRC) kunnen voorstellen moeten we een belangrijke ontbrekende stap uitleggen. Dit deel is dan ook één van de belangrijkste bijdragen van deze thesis. Gaussiaanse proces regressie (Sectie 2.1.5) heeft voor onze toepassing als grote tekortkoming dat het enkel een model kan leren voor één enkele gegeven functie of tijdreeks. Dit werk stelt daarom een niet-triviale aanpassing voor, steunend op GPR, die het mogelijk maakt om een algemeen model te leren dat een set van tijdreeksen beschrijft.

Voor we deze methode hebben ontworpen, hebben we reeds enkele andere mogelijkheden getest, zonder degelijk resultaat. We zullen deze eerst bespreken omdat ze een inzicht geven in het gestelde probleem.

- **Identificatie-parameter toevoegen:** In een eerste geteste aanpak voor het leren over verschillende tijdreeksen, voegden we een extra identificatie-parameter toe aan de feature-vector van de tijdreeks. Algemeen bestaat deze feature-vector uit een tijdsaanduiding van de tijdreeks, waarvan we de functiewaarden leren en later voorspellen (in ons geval de energieconsumptie van dat moment). Het probleem met deze parameter is echter dat ze geen temporele waarde bevat. Het toevoegen ervan creëert tevens ook het probleem dat we bij het

voorspellen niet weten welke parameter we de te voorspellen tijdreeks moeten meegeven. Bij het testen van deze methode merkten we dat de voorspelling bijna uitsluitend rekening hield met de informatie van de tijdreeks waarvan de parameter het dichtst bij de ingegeven parameter van de te voorspellen tijdreeks lag.

- **Tijdreeksen aaneenschakelen:** Een tweede manier om een algemeen model te creëren is het aaneenschakelen van verschillende tijdreeksen. Het probleem met deze techniek is echter dat we kunstmatige overgangen gaan creëren die er eigenlijk niet zijn, maar wel in rekening worden gebracht door het model. Dit zorgt er ook voor dat we lange tijdreeksen moeten gebruiken, willen we voldoende informatie per tijdreeks in deze aaneenschakeling steken. Hierdoor moeten er meer berekeningen gebeuren, GP heeft namelijk een complexiteit van $O(N^3)$ met N de lengte van de tijdreeks.

Na deze inzichten lijkt het misschien dat GP weinig voordelen opleveren om gebruikt te worden om een algemeen model te creëren. We kunnen alvast concluderen dat bovenstaande voorstellen geen goede oplossingen zijn. Maar zoals al eerder aangehaald is, zorgt de niet-parametrische aard van GP ervoor dat het een algemene eenvoudig bruikbare methode is.

5.2 Leren over een set van functies

Nu de moeilijkheden duidelijk zijn, stellen we onze aanpak voor die steunt op het gebruik van de waarschijnlijkheid (*likelihood*) die berekend wordt binnen GPR (Vergelijking 2.39). We willen een algemeen model leren dat de totale waarschijnlijkheid van een set van tijdreeksen maximaliseert. M.a.w. we willen de hyperparameter van het algemeen model optimaliseren. Om dit mogelijk te maken, introduceren we een nieuwe functie die de totale waarschijnlijkheid van alle tijdreeksen optelt (Algoritme 2). Merk op dat hierin K de covariantiematrix is die afhangt van de gebruikte kernel-functie en haar hyperparameters, en de inputwaarden.

Algoritme 2 Overall likelihood over all time series.

INPUT: TSS ($TimeSeriesSet:$ *inputs and targets set*), k (*covariance function*), σ_n^2 (*noise level*), σ_f^2 (*signal variance*), l (*lengthscale*)

1: $likelihood_{overall} \leftarrow 0$
2: ***for*** (X, y) ***in*** TSS ***do***
3: $L \leftarrow cholesky(K + \sigma_n^2 I)$
4: $\alpha \leftarrow L^T \setminus (L \setminus y)$
5: $-\log\ p(y|X) \leftarrow \frac{1}{2} y^T \alpha + \sum_i \log L_{ii} + \frac{n}{2} \log 2\pi$
6: $likelihood_{overall} \leftarrow likelihood_{overall} + (-\log\ p(y|X))$

RETURN: $likelihood_{overall}$ (*overall likelihood*)

Nu we via Algoritme 2 de totale waarschijnlijkheid van een set van tijdreeksen kunnen bepalen, gaan we op zoek naar het model waarvoor deze totale waarschijnlijkheid maximaal is. Omdat Algoritme 2 de negatieve totale waarschijnlijkheid berekent, zullen we dus op zoek moeten gaan naar de hyperparameters waarbij deze functie minimaal is (Algoritme 3). We gebruiken de notatie $TSS.X$ en $TSS.y$ om respectievelijk de set van tijdstempels en de set van doelwaarden (consumptiewaarden) van een enkele tijdreeks voor te stellen. Door gebruik te maken van de optimale hyperparameters ($\sigma^2_{n*}, \sigma^2_{f*}$ en l_*) en de gewogen gemiddelde doelwaarden $\overline{y}$ kunnen we de optimale L en α berekenen (als gewichten voor het gewogen gemiddelde nemen we de relatieve *likelihoods* van de verschillende modellen). M.a.w. we berekenen de posterior gebruikmakend van de algemene optimale hyperparameters en het gewogen gemiddelde (gebruikmakend van de relatieve likelihoods). Deze posterior en het gewogen gemiddelde $\overline{y}$ kunnen op hun beurt dan gebruikt worden om het algemeen model te creëren door een voorspelling te doen voor dezelfde periode (zelfde set van tijdstempels). Merk op dat we veronderstellen dat alle tijdreeksen van de set TSS over dezelfde periode ($X_i | i \in (1, 2, .., N)$) zijn genomen. Daarom kunnen we een willekeurige rij (periode) uit de tijdreeksenset TSS kiezen, bijvoorbeeld de eerste (lijn 3 Algoritme 3). Merk ook op dat voor het clusteren met GPR de trainings- en testset dezelfde is. Op deze manier beschouwen we enkel de onzekerheid van het clusteren.
Wanneer we tijdreeksen willen vergelijken van verschillende momenten zullen we de tijden moeten synchroniseren of gebruikmaken van een relatieve tijdstempeltoewijzing. Dit laatste wordt in de experimenten van deze thesis gebruikt.

Algoritme 3 Algemeen model berekenen

INPUT: TSS (*TimeSeriesSet : inputs and targets set*),
k (*covariance function*), σ^2_{ni} (*initial noise level*),
σ^2_{fi} (*initial signal variance*), l_i (*initial length scale*)

1: $(\alpha, L) \leftarrow minimum(likelihood_{overall}(\ TSS, k, \sigma^2_{ni}, \sigma^2_{fi}, l_i))$
2: $\overline{y} \leftarrow mean_w(TSS.y)$
3: $x_* \leftarrow$ *firstRow*$(TSS.X)$
4: $\overline{y}_* \leftarrow \overline{y} + k_*^T \alpha$
5: $v \leftarrow L \setminus k_*$
6: $\mathbb{V}[y_*] \leftarrow k(x_*, x_*) - v^T v$

RETURN: $\overline{y}_*$ (*mean*), $\mathbb{V}[y_*]$ (*variance*)

5.3 Recursieve clustering

Voor de eigenlijke clustering gebruiken we een recursieve clusteraanpak. Kort samengevat werkt ze als volgt: We creëren een algemeen model die de totale set van tijdreeksen beschrijft, daarna zullen we de fout berekenen tussen het generaal model en elke tijdreeks. Op basis van deze foutenwaarden maken we dan een beslissing om een bepaald aantal tijdreeksen af te splitsen. Op die manier hebben we twee nieuwe clusters waarop we deze techniek nog eens toepassen. Dit herhalen we tot bepaalde

voorwaarden voldaan zijn. De totale methode staat beschreven in Algoritme 5 en zullen we nu in detail uitleggen.

Foutenwaarde bereken

Voor het berekenen van de foutenwaarden maakt de methode gebruik van de *root mean square error* (RMSE) tussen de geobserveerde tijdreeksen en het algemene model. De RMSE is de vierkantswortel van het gemiddelde van de som van de kwadraten van de fouten tussen de voorspelling (v) en de observatie (o) (Vgl. 5.1). De RMSE-waarde wordt vaak gebruikt en is een excellente universele foutenmaatstaf. In tegenstelling tot de gelijkaardige gemiddelde absolute fout (MAE), zal RMSE grote fouten extra in rekening brengen.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(v_i - o_i)^2} \tag{5.1}$$

Deze waarde wordt gebruikt als de gelijkaardigheidsmaatstaf tussen de verschillende tijdreeksen.

Clusterparameters

De clustermethode maakt gebruik van drie parameters:

1. s_{min}: minimum clustergrootte; minimum gewenste aantal tijdreeksen in een cluster.

2. t_{sim}: minimum gelijkaardigheidsdrempel; gewenste minimum gemiddelde gelijkaardigheid in een gevormde cluster.

3. r: splitsingsratio; de ratio van het aantal tijdreeksen die afgesplitst worden wanneer de splitsingscriteria voldaan zijn.

Het gebruik van de splitsingscriteria is vervat in Algoritme 4. Zolang de minimum clustergrootte en de minimum gelijkaardigheidsdrempel niet bereikt zijn zal het algoritme nieuwe clusters vormen gebruikmakend van de splitsingsratio. Wanneer we een cluster niet verder kunnen clusteren zal die originele cluster teruggegeven worden. Een niveau hoger (Algoritme 5) zal dan gecheckt worden of de gevormde cluster verschillend is van de originele inputlijst van tijdseries (merk op dat dit een recursieve methode is). Wanneer dit niet het geval is geven we de originele cluster terug als resultaat van deze recursiestap (lijn 6-7, Alg. 5). Merk op dat lijn 2 van Algoritme 4 een normalisatiestap is van de RMSE waarden van de cluster en dat de tijdreeksenset (TSS) omgekeerd geordend is volgens de RMSE waarden van de tijdreeksen (lijn 4, Alg. 5). De reden hiervoor is dat we op deze manier de tijdreeksen met de grootste RMSE foutenwaarden afsplitsen, gebruikmakend van de splitsingsratio (r) (lijn 4, Alg. 4). De clustering is deterministisch, d.w.z. dat vaste parameters steeds dezelfde clustering zullen produceren. Daarnaast laat de hiërarchische aard van de clustering toe om een structureel beeld te vormen van de (on)gelijkheid van de tijdreeksen.

Door de visualisatie van het resultaat kunnen we dus de afstand van de clusters in de boom dan ook interpreteren als afstandsfunctie voor de gelijkheid.
Op deze manier hebben we het volledige algoritme besproken en rest ons enkel nog een evaluatie van de performantie van het algoritme in de volgende sectie.

Algoritme 4 Splitsen van een cluster

INPUT: TSS (*TimeSeriesSet : inputs and targets set*),
E_r (*list of RMSE values*), t_{sim} (*similarity threshold*),
s_{min} (*minimum cluster size*), r (*split ratio*)

if $size(TSS) > s_{min}$ **then**
  $E'_r \leftarrow E_r / max(E_r)$
  if $mean(E'_r) < t_{sim}$ **then**
    $(C_1, C_2) \leftarrow split(TSS, E'_r, r)$
    return(C_1, C_2)
  else
    return TSS
else
  return TSS

RETURN: C (*Clustering*)

Algoritme 5 Totaal clusteralgoritme voor tijdreeksen

INPUT: TSS (*TimeSeriesSet : inputs and targets set*),
k (*covariance function*), σ^2_{ni} (*noise level*),
σ^2_{fi} (*signal variance*), l_i (*length scale*), t_{sim} (*similarity threshold*),
s_{min} (*minimum cluster size*), r (*split ratio*)

$(\overline{f}_*, \mathbb{V}[f_*]) \leftarrow findGeneralModel(TSS, k, \sigma^2_{ni}, \sigma^2_{fi}, l_i)$
for (X, y) ***in*** TSS ***do***
  $E_r \leftarrow RMSE(y, \overline{f}_*)$
$TSS \leftarrow reverseOrderBy(TSS, E_r)$
$(C_1, C_2) \leftarrow divide(TSS, E_r, t_{sim}, s_{min}, r)$
if $C_1 == TSS$ *or* $C_2 == TSS$ **then**
  return TSS
else
  $cluster(C_1, k, \sigma^2_{ni}, \sigma^2_{fi}, l_i, t_{sim}, s_{min}, r)$
  $cluster(C_2, k, \sigma^2_{ni}, \sigma^2_{fi}, l_i, t_{sim}, s_{min}, r)$

RETURN: C (*Clustering*)

5.4 Complexiteit

Na de voorstelling van de nieuwe clustermethode is een evaluatie van de performantie gewenst. Als we veronderstellen dat de tijdseries bestaan uit N (equidistant) waarden, zal het leergedeelte van een Gaussiaanse processen regressie een complexiteit hebben van $O(N^3)$, de voorspelling heeft een complexiteit van $O(N^2)$ [8]. In totaal is de kost dus $O(N^3)$. Als we S aantal tijdreeksen beschouwen, zullen we dit S moeten herhalen voor het creëren van een algemeen model (Alg. 3). De totale kost is hierdoor dus $O(SN^3)$ plus de kost van de minimalisatie ($O(S)$, empirisch vastgesteld). Na het leren van het algemeen model volgt ons enkel nog een vergelijking van de RMSE waarden, dus de totale kost van Algoritme 5 blijft $O(SN^3)$.
In de meeste clustersituaties is N een vaste waarde maar variëert het aantal te clusteren elementen (tijdreeksen). Wanneer we de complexiteit van dit algoritme vergelijken met paarsgewijze clustermethoden als *k-medoids* (Sectie 2.2.2) of HAC (Sectie 2.2.3) reduceert GPRC de complexiteit van $O(S^2)$ tot $O(S)$, wat resulteert in een betere schaalbaarheid.

Hoofdstuk 6

Experimenten en resultaten

Dit hoofdstuk geeft de lezer een overzicht van de opzet en de resultaten van de experimenten. Sectie 6.1 bespreekt de experimenten omtrent de voorspelling en Sectie 6.2 omtrent de clustering.

6.1 Voorspelling

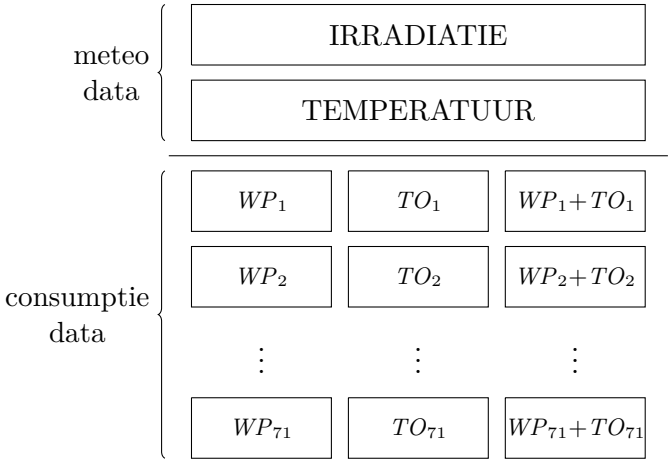
In deze sectie bespreken we de experimentele opzet (Sectie 6.1.1) en de resultaten (Sectie 6.1.2) van de voorspellingsexperimenten. De resultaten van GPR (bekomen via twee verschillende pakketten) zullen vergeleken worden met die van *support vector regressie* (SVR), lineaire regressie (OLS) en een baseline (BL) methode die de waarde van 7 dagen geleden teruggeeft als voorspelling.

6.1.1 Experimentele opzet

Data

De gebruikte data werd beschikbaar gesteld door 3E [1], een bedrijf gespecialiseerd in duurzame energie consulting, onderzoek en software. De geleverde data is verzameld in de context van het FLEXIPAC project [49] gedurende één jaar (september 2013 t.e.m. augustus 2014) en bestaat uit de consumptie van alle toestellen en de warmtepomp van 71 verschillende huishoudens. Daarnaast is er ook nog meteo data beschikbaar gesteld. De structuur van de data wordt getoond in Figuur 6.1.

[1] `http://www.3e.eu`

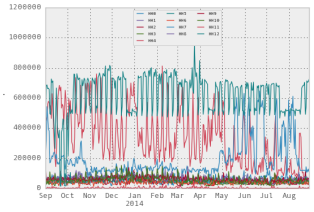


Figuur 6.1: De structuur van de geleverde data. WP_x is de consumptie van de warmtepomp, TO_x is de consumptie van de toestellen en $WP_x + TO_x$ is de som van beide. Het subschrift x duidt het nummer van het huishouden aan.

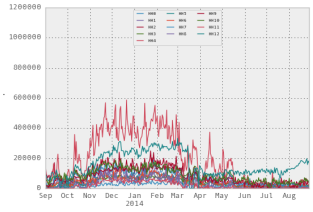
Stap 1: Data exploratie

Tijdgrafieken

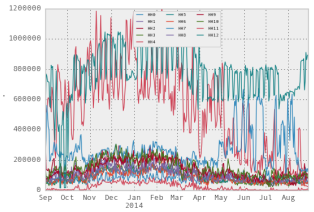
De tijdgrafieken geven al enkele interessante inzichten. Uit Figuur 6.2 leiden we af dat het verbruik van de toestellen doorheen het jaar redelijk constant blijft voor de meeste huishoudens en voor enkele huishoudens het afhankelijk is van het seizoen. Op Figuur 6.3 en 6.4 zien we daarentegen dat het verbruik van de warmtepomp voor de meeste huishoudens sterk afhankelijk is van het seizoen met het grootste verbruik in de wintermaanden. Figuren 6.5 en 6.6 tonen respectievelijk de hoeveelheid zonneschijn en de temperatuur doorheen het jaar. Wanneer we deze vergelijken met het verbruik van de warmtepomp 6.3 zien we een duidelijk verband, deze metingen zullen dus zeker opgenomen worden in de feature vector.



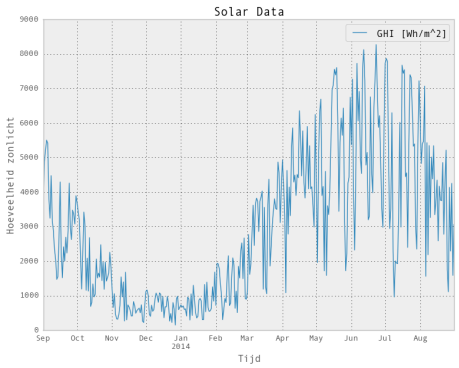
Figuur 6.2: Jaarpatroon consumptie toestellen



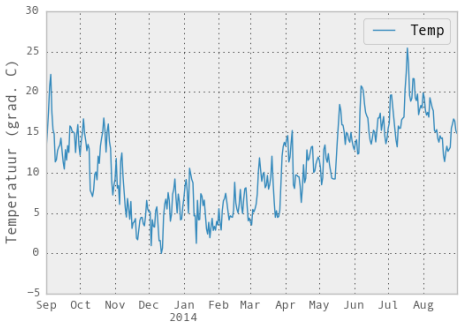
Figuur 6.3: Jaarpatroon consumptie warmtepomp



Figuur 6.4: Jaarpatroon consumptie toestellen + warmtepomp

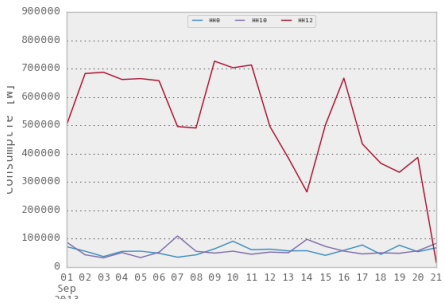


Figuur 6.5: Jaarpatroon zonneschijn

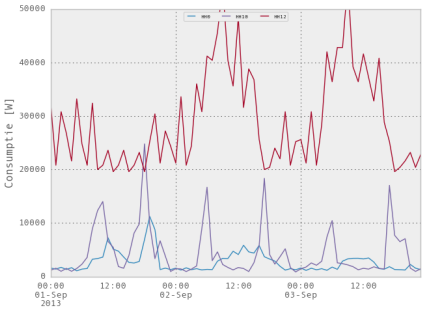


Figuur 6.6: Jaarpatroon temperatuur

Omdat de beschikbare data een tijdspanne van één jaar heeft, is het helaas zeer moeilijk om effecten doorheen het jaar te leren. In plaats daarvan focussen we ons op het voorspellen per maand. We zullen dus inzoomen op een kortere tijdspanne en een voorspelling per uur proberen te bekomen. Figuur 6.8 toont de consumptie per uur gedurende 3 dagen. Hieruit blijkt dat het uur van de dag een belangrijke factor is die we zeker moeten toevoegen aan de feature vector. Figuur 6.7 toont het verbruik per dag gedurende een maand. Hierop merken we al een terugkerend patroon per week. We zullen dit verder onderzoeken in de auto-correlatiegrafieken.



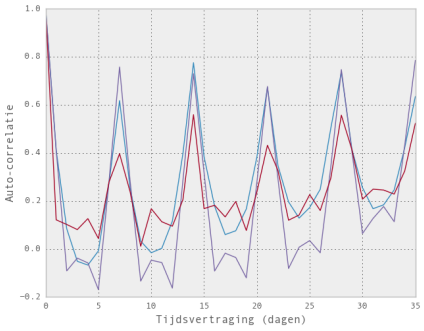
Figuur 6.7: Maandpatroon van de consumptie van de toestellen + warmtepomp



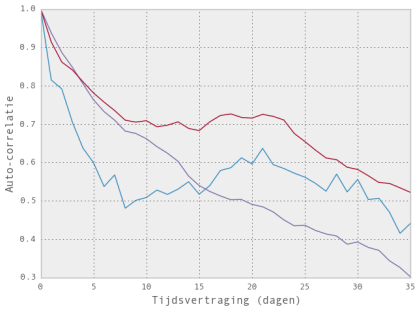
Figuur 6.8: Dagpatroon van de consumptie van de toestellen + wamtepomp 3 dagen

Auto-correlatiegrafieken

Uit Figuur 6.9 kunnen we afleiden dat er een sterke correlatie is tussen de consumptie van de toestellen van de huidige dag en deze van 7, 14, 21,... dagen geleden en een negatieve correlatie tussen het verbruik van de huidige dag en dat van 2 dagen geleden. Uit Figuur 6.10 leiden we af dat de auto-correlatie van het verbruik van de warmtepomp sterk verschilt, afhankelijk van het beschouwde huishouden. Dit zal waarschijnlijk te maken hebben met de instelling van de warmtepomp.



Figuur 6.9: Auto-correlatie van de consumptie van de toestellen voor enkele huishoudens. (Dit is het geval voor het grootste deel van de huishoudens.)



Figuur 6.10: Auto-correlatie van de consumptie van de warmtepomp voor enkele huishoudens. (Dit is zeer afhankelijk van de geselecteerde huishoudens.)

Stap 2: Data preprocessing

Data cleaning

We testten de data op ontbrekende waarden, dit kwam maar een tiental keer voor en deze waarden werden geïnterpoleerd.

Data integratie

Zoals uit Figuur 6.1 blijkt is de data afkomstig van verschillende datasets. De meteo data heeft een sample-frequentie van een uur en de consumptiedata heeft een sample-frequentie van een kwartier. We hebben de sample-frequentie van consumptiedata herschaald naar een uur door de consumptiewaarden van de vier kwartieren per uur op te tellen. Vervolgens hebben we deze meteo- en consumptiedata gebundeld samen met de tijdsfeatures. Als tijdsfeatures nemen we:

- *Dag van het jaar*: $[0, 1, 2, \ldots, 365]$
- *Dag van de week*: $[0, 1, 2, \ldots, 6]$ met $0 =$ maandag, $1 =$ dinsdag,$\ldots$, $6 =$ zondag
- *Weeknummer*: $[0, 1, \ldots, 51]$
- *Uur van de dag*: $[0, 1, \ldots, 23]$

Tenslotte worden feestdagen ook gemarkeerd als een zondag (*Dag van de week*:6), omdat de meeste huishoudens dan gelijkaardig consumptiegedrag vertonen.

Schalen van features

Alle input features worden genormaliseerd door middel van de *Z-Score* transformatie die besproken is in Sectie 4.2.3.

Feature selectie

Voor het selecteren van de features hebben we twee combinaties getest die interessant zijn om te onderzoeken in het kader van de auto-regressieve GP methode. We hebben echter ook enkele combinaties getest door het weglaten van de meteo data en verschillende tijdsfeatures maar deze resultaten waren echter steeds slechter, we gaan hier niet verder op in. Tabellen 6.1 en 6.2 tonen de twee feature vectoren die gebruikt worden. We zullen uittesten wat het effect van de vertraagde variabele is. We selecteren hiervoor de consumptie van twee dagen geleden en zullen de negatieve auto-correlatie effecten hiervan onderzoeken.

Doelvariabele	Afgeleide voorspellers					exogene voorspellers	
EEC[W]	EEC_{-2D}	Dag_{Jaar}	Dag_{Week}	Nr_{week}	Uur_{Dag}	Temp [°C]	IR $[Wb/m^2]$

Tabel 6.1: Feature vector met de vertraagde variabele: consumptie van twee dagen geleden.

Doelvariabele	Afgeleide voorspellers					exogene voorspellers	
EEC[W]	/////	Dag_{Jaar}	Dag_{Week}	Nr_{week}	Uur_{Dag}	Temp [°C]	IR $[Wb/m^2]$

Tabel 6.2: Feature vector zonder de vertraagde variabele.

Stap 3: consumptievoorspelling

Trainings- en testset

Omdat de beschikbare data beperkt is tot één jaar, zullen we ons richten op het leren van een model per maand. Concreet nemen we de eerste 25 dagen als trainingsset en zullen dag 27 en 28 als testset gebruikt worden. We zullen dus een voorspelling voor de komende twee dagen per uur willen testen. We nemen een kloof van 2 dagen omdat de consumptie van twee dagen geleden als feature gebruikt wordt. De onafhankelijkheidseis van trainings- en testset moet gerespecteerd worden (Sectie 4.3.1).

Gebruikte methoden en hyperparameter optimalisatie

Voor het voorspellen met GPR maken we gebruik van pyGP en SKlearn GP. PyGp voorziet automatische hyperparameter optimalisatie gebaseerd op *maximum likelihood estimation*. Voor de optimalisatie bij SKLearn GP maken we gebruik van de *grid search CV*-methode van de SKLearn toolkit in combinatie met een *8-fold cross-validation* met kloven van twéé dagen die besproken is in Sectie 4.3.2.
Ter vergelijking gebruiken we een baseline methode die als voorspelling de waarde van 7 dagen geleden geeft. Dit is gebaseerd op de resultaten van de auto-correlatiegrafieken (Sectie 6.1.1).
Daarnaast gebruiken we ter vergelijking uit SKLearn ook nog de lineaire regressie, gebaseerd op kleinste kwadraten benadering (OLS) (Sectie 2.1.2) en *support vector*

Methode	Hyperparameters	Waarden
SVR	C: sanctieparameter	$10^{-2}, \cdots, 10^{5}$
	ϵ: breedte van de fout-ongevoelige ϵ-tube	$10^{-3} \cdots 10^{3}$
SKLearn GP	σ_0: signaalvariantie	$10^{-4}, \cdots, 10^{1}$
	σ_n: ruisvariantie	$10^{-3}, \cdots, 10^{-1}$

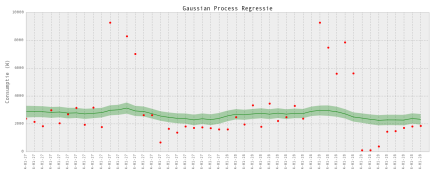
Tabel 6.3: Zoekruimte hyperparameters.

regressie (SVR) (Sectie 2.1.3). Voor het optimaliseren van de hyperparameters van SVR gebruiken we dezelfde methode als voor SKLearn GP. Tabel 6.3 toont de zoekruimte van hyperparameters voor SKLearn GP en SVR, beide met een lineaire kernel.

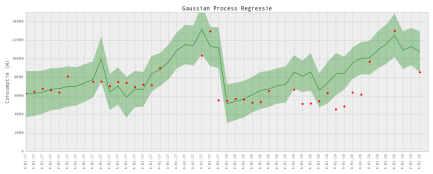
6.1.2 Resultaten

Algemene evaluatie voorspelling

Ter inleiding van de resultaten tonen we Figuren 6.11 en 6.12. Deze tonen GPR voorspelling voor het verbruik van de warmtepomp + toestellen voor de komende twee dagen per uur. We zien duidelijk dat het eerste huishouden (HH12) slechter voorspeld is dan het tweede (HH10). Wanneer we de voorspelling voor alle 71 huishoudens uitvoeren, verkrijgen we de resultaten uit Tabel 6.4 en 6.5.



Figuur 6.11: Twee-daagse voorspelling per uur van het verbruik warmtepomp + toestellen voor huishouden 12. Dit huishouden is relatief moeilijk te voorspellen.



Figuur 6.12: Twee-daagse voorspelling per uur van het verbruik warmtepomp + toestellen voor huishouden 10. Dit huishouden is relatief makkelijk te voorspellen.

Deze tabellen tonen de gemiddelde fouten van de 71 huishoudens. Wanneer we de volledige feature vector gebruiken, concluderen we dat *Support vector regressie* en de baseline methode de beste methoden zijn voor het voorspellen van de gemiddelde consumptie van de warmtepomp van de huishoudens afgaande op de gemiddelde MRE resultaten, op de voet gevolgd door GPR. GPR is de beste methode voor het voorspellen van de consumptie van de toestellen en de consumptie van de toestellen + warmtepomp (Tabel 6.4).
Wanneer de vertraagde variabele niet wordt gebruikt merken we dat SVR de beste methode is voor de voorspelling van het verbruik van de warmtepomp en het verbruik van toestellen afgaande op de gemiddelde MRE. GPR blijft de beste methode voor het

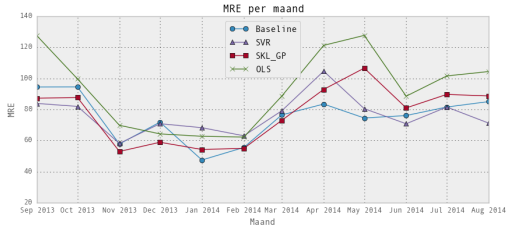
voorspellen van het verbruik van de warmtepomp + toestellen (Tabel 6.5). Wanneer we echter op de gemiddelde MAE afgaan, is GPR op elk gebied de best presterende methode. Omdat echter MRE een maatstaf is die makkelijke vergelijking toelaat, focussen we ons op de MRE waarden.
Een belangrijke eerste conclusie is dat het gebruik van de vertraagde variabele de performantie van GPR verbetert en die van de andere methoden verslechtert. Dit is te wijten aan de slechte auto-correlatie tussen de energieconsumptie van de huidige dag en deze van twee dagen geleden (Sectie 6.1.1). GPR is een auto-regressieve methode die zelf seizoenseffecten detecteert en dus extra relevante informatie zal meestal een betere voorspelling opleveren omdat we nu een referentiepunt meegeven. De andere methoden gebruiken regressie om verbanden tussen de features te vinden, zonder zelf een auto-correlatief effect te onderzoeken. Hierdoor zal het toevoegen van een feature met een negatieve auto-correlatie (t.o.v. het te voorspellen verbruik), ook een negatieve impact hebben op de voorspelling.
De baseline methode geeft de waarde van vorige week terug als voorspelling, uit de auto-correlatiegrafieken 6.9 en 6.10 bleek hiertussen een sterk verband. Zoals besproken in Sectie 6.1.1 is dit effect sterker bij het verbruik van de toestellen en zo blijkt ook de voorspelling van de toestellen de beste. Een derde methode waar we mee vergelijken is de OLS. Deze methode doet het relatief slecht, zeker voor de voorspelling van de warmtepomp.
Een laatste belangrijke vergelijking is de vergelijking van de twee gebruikte pakketten voor het berekenen van de GPR. We merken dat SKlearn GP het beduidend beter doet dan pyGP maar dat de performantie in beide gevallen verbetert wanneer we gebruik maken van de vertraagde variabele. Het verschil tussen beide pakketten is de manier waarop de hyperparameters geleerd worden. pyGP maakt gebruikt van de *maximum likelihood estimation* methode (MLE) voor de *marginal likelihood maximization* 2.1.5 en SKLearn GP van de *cross-validation* methode (CV) 4.3.2. Uit de resultaten blijkt SKLearn duidelijk beter te zijn. Dit komt omdat MLE gevoeliger is voor overfitting [50]. Voor meer informatie hieromtrent verwijzen we u naar het werk van Cawley et al. [51] [52].
SVR en SKlearn GP gebruiken dezelfde methode (CV) voor het optimaliseren van de hyperparameters en SKLearn GP bekomt de beste resultaten voor GPR. Om deze redenen zullen we de resultaten van het SKLearn pakket als verdere referentie voor de voorspellingsresultaten van GPR gebruiken.
Merk wel op dat we in het clustergedeelte van deze thesis gebruikmaken van pyGP omdat deze implementatie de nodige aanpassingen beter ondersteunt en we contact hadden met de schrijvers van dit pakket.

	pyGP (lin+rbf)	pyGP (lin)	SKLearn GP (lin)	SVR (lin)	Baseline -7D	OLS
HP	84,90 (1467)	84,77 (1465)	77,34 (1422)	76,19 (1602)	76,00 (1521)	93,21(1639)
Ot	56,39 (1408)	57,22 (1426)	54,80 (1350)	63,12 (1773)	58.60 (1484)	62,94 (1590)
Ot+HP	51,13 (2331)	51,46 (2343)	48,73 (2215)	67,41 (3383)	54.07 (2509)	55,97 (2574)

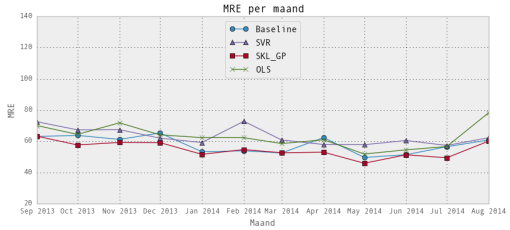
Tabel 6.4: Overzicht van de gemiddelde MRE (en MAE) fouten van de 71 huishoudens, gebruikmakend van de vertraagde variabele die de consumptie van 2 dagen geleden bevat. Tussen haakjes achter de methode staat de gebruikte kernelfunctie.

	pyGP (lin+rbf)	pyGP (lin)	SKLearn GP (lin)	SVR (lin)	Baseline -7D	OLS
HP	94,07 (1620)	94,07 (1608)	78,93 (1414)	67.50 (1471)	76,00 (1521)	84,25 (1596)
Ot	62,40 (1554)	60,64 (1516)	55,39 (1361)	52.70 (1450)	58.60 (1484)	59,37 (1495)
Ot+HP	56,32 (2559)	54,98 (2514)	49,01 (2213)	50.66 (2454)	54.07 (2508.76)	52,76 (2413)

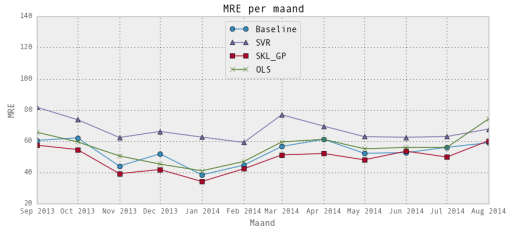
Tabel 6.5: Overzicht van de gemiddelde MRE (en MAE) fouten van de 71 huishoudens, zonder de vertraagde variabele die de consumptie van 2 dagen geleden bevat. Tussen haakjes achter de methode staat de gebruikte kernelfunctie.



(a)



(b)



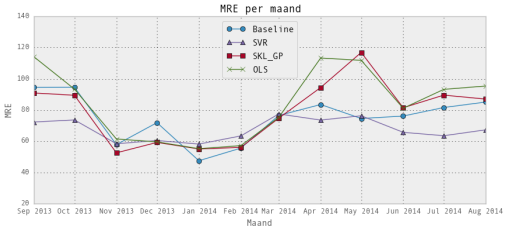
(c)

Figuur 6.13: Gemiddelde relatieve fout per maand (consumptie van 2 dagen geleden in feature vector). (a) Warmtepomp. (b) Toestellen. (c) Warmtepomp + toestellen.

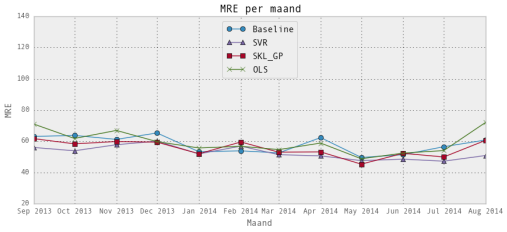
Evaluatie per maand

Figuren 6.13 en 6.14 tonen de MRE foutenwaarden per maand doorheen het jaar. Op Figuur 6.13a en Figuur 6.14a merken we dat de foutenwaarden van de warmtepomp het grootst zijn in de lentemaanden, dit kunnen we mogelijk verklaren doordat de huishoudens hun verwarming gaan afbouwen naar de zomer toe en meer willekeur in het gebruik creëren. In de herfst zien we dit effect ook, maar in mindere mate.
De MRE voorspellingsfout van het verbruik van de toestellen is daarentegen relatief constant (Figuren 6.13b en 6.14b). De totale MRE van de toestellen en de warmte-

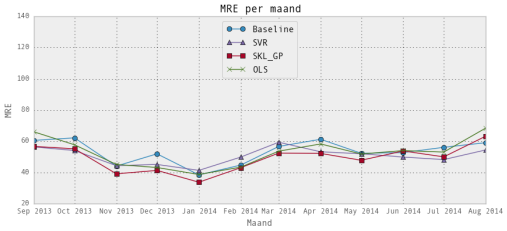
pomp heeft de kleinste fout doorheen de wintermaanden en is doorheen de rest van het jaar relatief constant (Figuren 6.13c en 6.14c). Dit komt omdat het grootste deel van het verbruik doorheen de wintermaanden naar verwarming gaat en dit werd met een relatief kleine fout voorspeld.



(a)



(b)



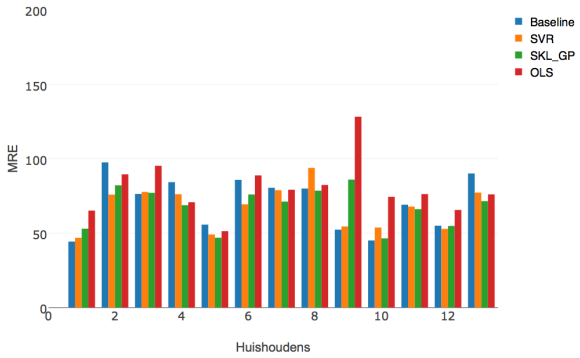
(c)

Figuur 6.14: Gemiddelde relatieve fout per maand (consumptie van 2 dagen geleden niet in feature vector). (a) Warmtepomp. (b) Toestellen. (c) Warmtepomp + toestellen.

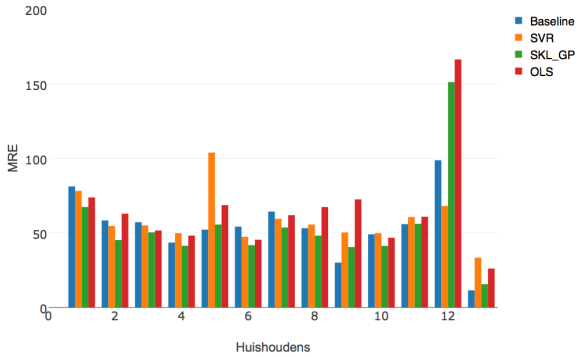
Evaluatie per huishouden

Bijlage B.1 bevat de MRE waarden per huishouden. Figuren 6.15 en 6.16 tonen een selectie van de eerste 13 huishoudens. Hierop zien we duidelijk dat het toevoegen van de vertraagde variabele een gunstig effect heeft op GPR en een negatief effect op de andere methoden. We concluderen dat GPR meestal de beste methode is voor het voorspellen van de consumptie van toestellen en de consumptie van de toestellen + warmtepomp. Voor de warmtepomp alleen is SVR algemeen meestal de beste methode. Merk op dat voor bv. huishouden 12 GPR een slechte voorspelling bekomt (consistent met Figuren 6.11 en 6.12).
Merk ook op dat de meeste methoden voor dezelfde huishoudens relatief slechte/ goede voorspellingen bekomen. Wanneer we moeilijk te voorspellen huishoudens

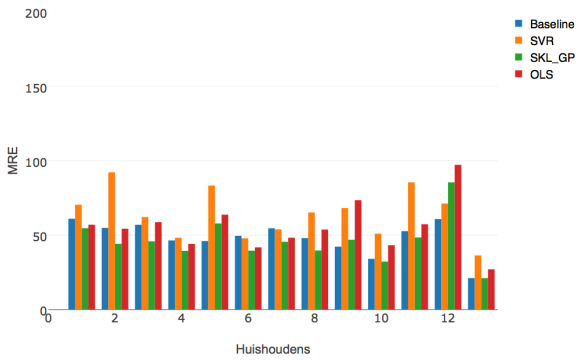
op voorhand zouden kunnen detecteren, kunnen we een nauwkeuriger beeld van de voorspelling verkrijgen. Clustering zou bijvoorbeeld hiervoor gebruikt kunnen worden.



(a)

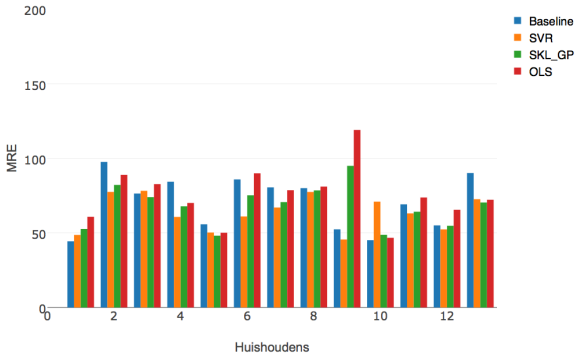


(b)

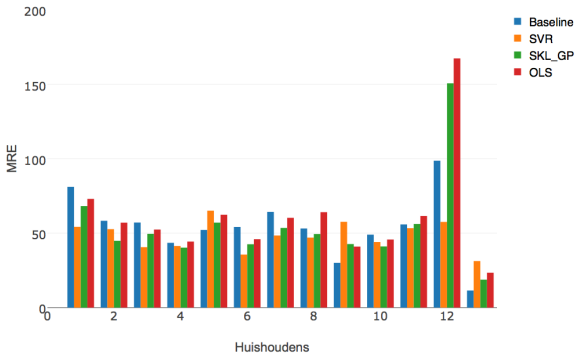


(c)

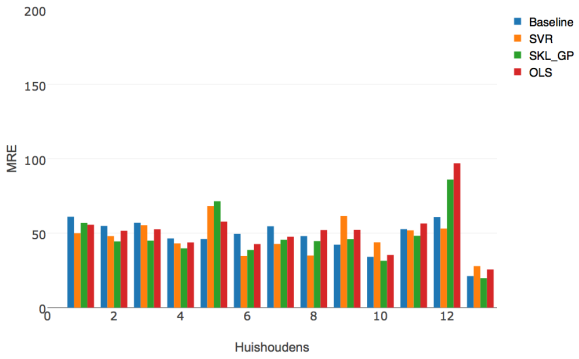
Figuur 6.15: Gemiddelde relatieve fout per huishouden (consumptie van 2 dagen geleden in feature vector). (a) Warmtepomp. (b) Toestellen. (c) Warmtepomp + toestellen.



(a)



(b)



(c)

Figuur 6.16: Gemiddelde relatieve fout per huishouden (consumptie van 2 dagen geleden niet in feature vector). (a) Warmtepomp. (b) Toestellen. (c) Warmtepomp + toestellen.

6.2 Clustering

In deze sectie bespreken we de experimentele opzet (Sectie 6.2.1) en de resultaten (Sectie 6.2.2) van de clusterexperimenten. De twee experimenten tonen aan dat de voorgestelde GPRC clustermethode interessante resultaten bekomt en beter schaalbaar is dan *k-means* en Hiërarchisch agglomeratief clusteren met DTW.

6.2.1 Experimentele opzet

Data

In de experimenten clusteren we weekpatronen van de elektrische consumptie van 71 huishoudens. De gebruikte dataset is de totale consumptie van de huishoudens (toestellen + warmtepomp) en is een onderdeel van de data die ook gebruikt wordt voor de voorspelling (Sectie 6.1.1). We *resamplen* de data zodat we tijdreeksen met metingen per uur bekomen en gebruiken een relatieve tijdstempel die bestaat uit de dag van de week en het uur van de dag. Zo kunnen we ook tijdreeksen van verschillende weken vergelijken. Ook normaliseren we de data (Sectie 4.2.3) om de focus op de vormen van de tijdreeksen te leggen.
Als conventie starten we de week op zaterdag zodat we duidelijk de overgang tussen het weekend en de werkweek kunnen visualiseren. Experimenteel bleek deze overgang interessanter dan de overgang tussen werkweek en weekend. In totaal bestaat een tijdreeks uit $N = 24 * 7 = 168$ samples per week en hebben we $S = 71$ tijdreeksen. Om huishoudens onderling te vergelijken, selecteren we dezelfde week zodat de weersomstandigheden gelijkaardig zijn. Om de veranderingen in de gebruikspatronen te analyseren selecteren we verschillende weken van dezelfde huishoudens.

PyGPs uitbreiden voor het leren over meerdere functies

Voor de Gaussiaanse proces regressie, maken we gebruik van het pyGP pakket [53]. We hebben dit pakket uitgebreid voor het leren optimaliseren van de (hyper)parameters over meerdere functies (Alg. 2) en het berekenen van de predicties (Alg. 3). Voor het vinden van het model met de maximale waarschijnlijkheid over de set van tijdreeksen (Alg. 3) gebruiken we het *L-BFGS-B* algoritme uit de *Scipy* toolbox [54]. De gebruikte kernelfunctie is een som van de radiale basisfunctie en de lineaire kernelfunctie. Deze combinatie geeft de beste nauwkeurigheid voor de voorspellingen. De startwaarden van de hyperparameters zijn $\sigma_{fi(LIN)} = \sigma_{fi(EXP)} = l_i = 10^{-7}$ *en* $\sigma_{ni} = e^{(2\log(0.1))}$.

Clustermethoden

Clustering gebaseerd op Gaussiaanse processen:
Onze GPRC clustermethode (Sectie 5.3) maakt gebruik van drie parameters: de minimum clustergrootte (s_{min}), de minimum gelijkaardigheidsdrempel (t_{sim}) en de splitsingsratio (r). Het is belangrijk een goede minimum gelijkaardigheidsdrempel te kiezen. Indien deze te hoog is, zal de gemiddelde RMSE waarde per cluster steeds kleiner zijn en zullen de clusters blijven opsplitsen totdat de gewenste minimum

clustergrootte wordt bereikt. Indien de gelijkaardigheidsdrempel te klein wordt gekozen zal de gemiddelde gelijkaardigheid initieel al voldoende zijn en zal er geen opsplitsing uitgevoerd worden.
Voor het eerste experiment gebruiken we $s_{min} = 1$ (om unieke profielen te kunnen ontdekken), $t_{sim} = 0.99$ en $r = 0.2$ om de looptijd op te meten. Respectievelijk gebruiken we voor het clusteren van de huishoudprofielen 1, 0.99 en 0.25. De gekozen weken lopen van 12-07-2014 t.e.m. 18-07-2014.
In het tweede experiment gebruiken we $s_{min} = 1$, $t_{sim} = 0.98$ en $r = 0.2$ voor het detecteren van (in)consistente huishoudprofielen. We gebruiken voor dit laatste een lagere gelijkaardigheidsdrempel om een zwakkere eis te stellen omdat we tijdreeksen van verschillende perioden gebruiken. De splitratio is in dit experiment ook kleiner omdat we minder tijdreeksen clusteren. We selecteren hiervoor een week per seizoen: 02-11-2013 t.e.m. 08-11-2013; 01-02-2014 t.e.m. 07-02-2014; 03-05-2014 t.e.m. 09-05-2014; 02-08-2014 t.e.m. 08-08-2014. Voor de overzichtelijkheid, gebruiken we drie huishoudens maar men kan een arbitrair aantal huishoudens toevoegen.
We zullen hetzelfde experiment herhalen voor de eerste vier weken van februari voor drie huishoudens. Op deze manier onderzoeken we de gelijkaardigheid van de weken binnen één maand. Als parameters gebruiken we $s_{min} = 1$, $t_{sim} = 0.99$ en $r = 0.2$. We gebruiken terug een hogere gelijkaardigheidsdrempel omdat de onderzochte perioden dichter bij elkaar liggen.

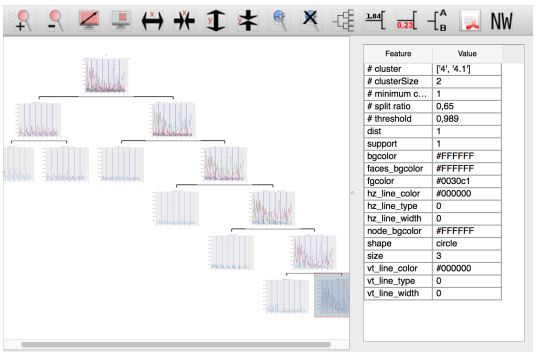
***K-medoids met DTW*:**
k-medoids veronderstelt dat we het aantal clusters a priori specificeren. In ons opzet is dit nadelig omdat we niet op voorhand weten hoeveel clusters wenselijk zijn om een inzicht te krijgen in de huishoudens. Voor het clusteren van de 71 huishoudens en het testen van de looptijd selecteren we $k = 15$ (en $k = 5$ voor het testen van 10 huishoudens), dit is gebaseerd op het aantal clusters die onze clustering vond tijdens het complexiteitsexperiment.

Hiërarchisch agglomeratief clusteren met DTW:
Voor de HAC methode moeten we geen parameters instellen. We gebruiken het *single-link* criteria (Sectie 2.2.3).

Interactieve visualisatie

Omdat onze data geen labels heeft, is het belangrijk dat we de clusterresultaten op een overzichtelijke manier kunnen visualiseren. Om deze resultaten te inspecteren hebben we de *ETE toolkit* [55] aangepast, zodat de visualisatie op een interactieve manier kan gebeuren. Deze toolkit biedt de functionaliteit aan om boomstructuren te visualiseren en maakt het de ideale toolkit voor het visualiseren van de vorming van de clusters door onze GPRC methode. Ook de HAC methode kan op deze manier geïnspecteerd worden. De resultaten van de *k-means* zijn iets minder interessant om via deze structuur te visualiseren omdat het een vlakke clustering is, toch blijft het ook hiervoor een handige manier om de gevonden clusters te inspecteren.
Omdat deze tool maar enkele formaten kan inlezen, converteren we de gevonden

clustering van Algoritme 5 naar het *Newick tree format* [56]. Voor elk van de gevonden clusters (kinderen van de boom), creëren we een afbeelding van de grafieken van de tijdreeksen in die cluster. Voor elke interne ouderknoop, creëren we een afbeelding van de grafieken van de unie van de tijdreeksen van de onderliggende kinderen en/of ouders. Deze afbeeldingen worden geplaatst in de visualisatie van de boom. De visualisatie is volledig interactief, elke knoop in de clustering selecteerbaar zodat extra informatie (Fig. 6.17) ervan getoond kan worden. Op deze manier kan de gebruiker op een zeer eenvoudige manier de resulterende clustering inspecteren.



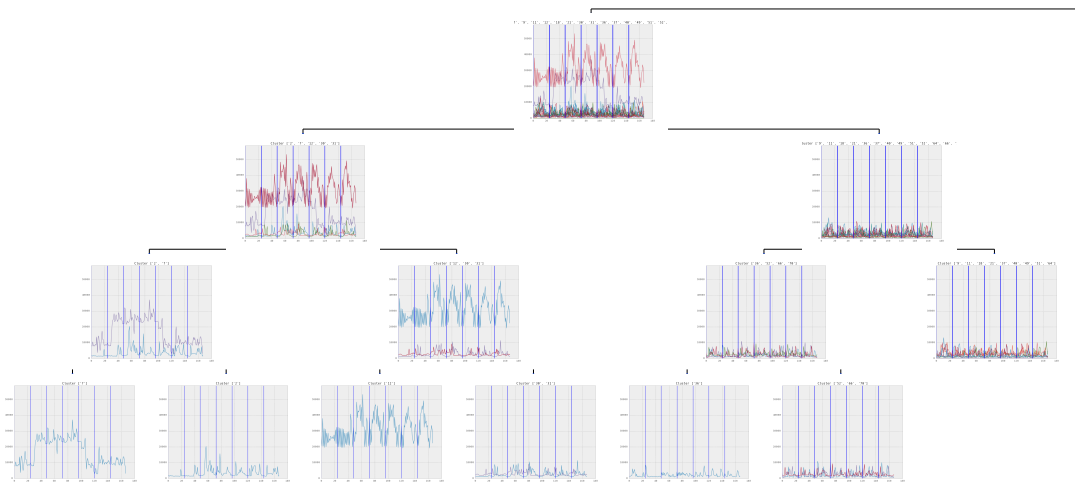
Figuur 6.17: Schermafbeelding van de interactieve visualisatie gebaseerd op de *ETE toolkit*.

6.2.2 Resultaten

Experiment 1: Huishoudens met gelijkaardig consumptiegedrag

Gevormde clusters

Wanneer we de GPRC methode toepasten op de 71 huishoudprofielen verkregen we het resultaat dat getoond wordt op Figuur B.7. Uit een selectie 6.18 hiervan kunnen we al enkele interessante resultaten ontdekken. De linkse tak bevat namelijk huishoudens met een weekendverbruik dat sterk verschilt van dat van de werkweek. In de rechtse tak en de takken onder de eerste linkerknoop, stellen we clusters van gelijkaardige dag/nacht-patronen vast. De resultaten van onze clustermethode zijn gevalideerd door een domeinexpert van 3E. Helaas hebben we geen metriek om de kwaliteit van de clustering objectief te vergelijken met *k-medoids*+DTW of HAC+DTW. De resulterende clustering van *k-medoids* en HAC is echter bijgesloten in B.8.



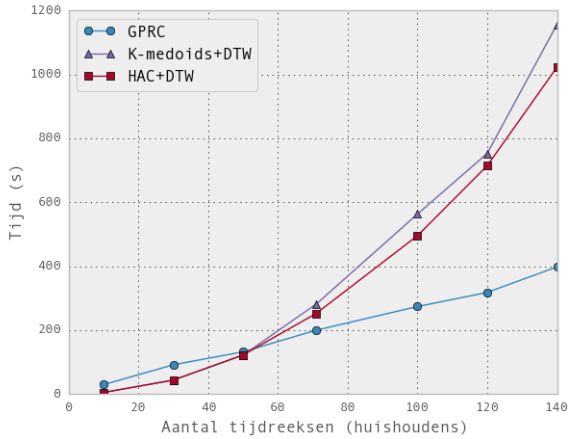
Figuur 6.18: Selectie uit de volledige clustering bekomen door de GPRC methode (Volledige versie in Fig. B.7).

Looptijden:

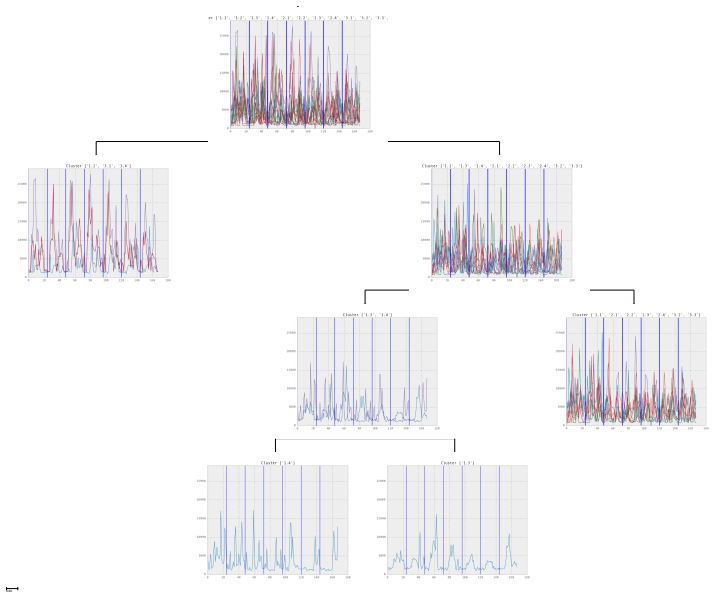
Ondanks dat we geen metriek hebben voor de clusterkwaliteit kunnen we wel objectief de schaalbaarheid bepalen door de looptijden van de algoritmen te vergelijken. De gunstige tijdscomplexiteit van de GPRC methode t.o.v. *k-medoids* en HAC wordt bewezen door Figuur 6.19. De experimenten bevestigen de lineaire trend in functie van het aantal huishoudens voor GPRC en een kwadratische voor *k-medoids* en HAC. Voor minder dan 50 huishoudens zal *k-medoids* en HAC sneller zijn omdat het leren van het model bij GPRC meer tijd vraagt om te berekenen. Vanaf 50 huishoudens is de GPRC methode sneller omdat er geen paarsgewijze vergelijking van de tijdreeksen meer nodig is. Dit bewijst de schaalbaarheid van onze GPRC methode.

Experiment 2: Huishoudens met (in)consistent consumptiegedrag

Wanneer we de spreiding van de weekprofielen over de cluster bekijken in Figuur 6.20, zien we dat alle weken van huishouden 2 in dezelfde cluster gegroepeerd zijn. De weekprofielen van huishouden 3 en 1 zijn verspreid over respectievelijk twee en vier clusters. Dit betekent dat huishouden 2 het meest consistente consumptiegedrag vertoont en huishouden 1 het meest inconsistente. Een visuele check van de tijdreeksen via de interactieve tool bevestigt dit. De tijdreeksen van huishouden X benoemen we $X.1$, $X.2$, $X.3$ en $X.4$ voor respectievelijk de winter-, lente-, zomer- en herfstweek. Hierdoor kunnen we besluiten dat huishouden 3 een gelijkaardig herfst- en wintergedrag vertoont omdat deze gegroepeerd zijn. Dit geldt ook voor het lente- en zomergedrag.



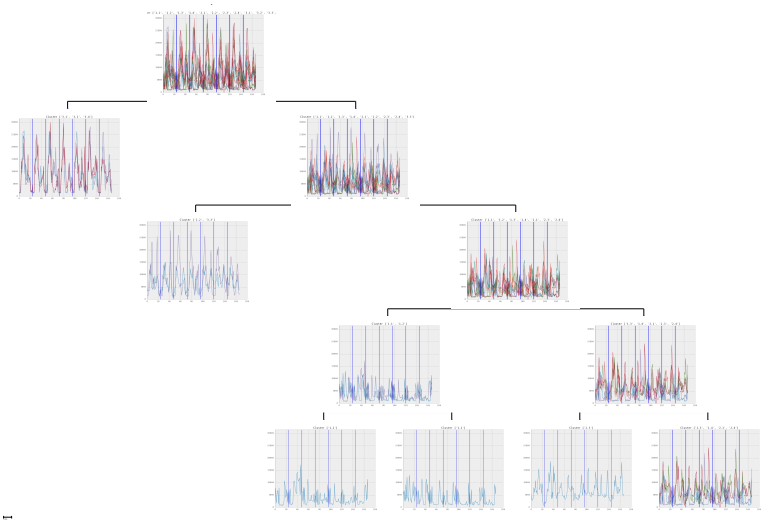
Figuur 6.19: Vergelijking van de looptijden van de verschillende clustermethoden. Meerdere weken per huishouden werden gebruikt om tot 140 tijdreeksen te komen.



Figuur 6.20: Clustering van weekprofielen van de vier seizoenen van drie huishoudens. We gebruiken $X.1, X.2, X.3$ en $X.4$ om de weekprofielen van huishouden X aan te duiden.

Figuur 6.21 toont het clusterresultaat van de vier weken van februari van 3 huishoudens. Hierin zien we dat huishouden 1 verdeeld is over drie clusters, net als huishouden 2. Huishouden 3 is verdeeld over twee. Daarnaast concluderen we dat $\frac{3}{4}$ van de weken van huishouden 3 in één cluster zitten en dat de vierde week zich in de dichtstbijzijnde cluster bevindt. Met dicht/ver bedoelen we de afstand tussen de bladeren (clusters) in de boomstructuur. Op deze manier zien we ook dat drie

weken van huishouden 2 lang bij elkaar geclusterd blijven (buiten week 2), terwijl de weken van huishouden 1 sneller afgesplitst worden van elkaar (zie 1.1 en 1.2). Wanneer we dit resultaat terugkoppelen naar de voorspellingsresultaten, zien we dat deze drie huishoudens respectievelijk een MRE foutwaarde van 58,41, 40,31 en 29,17 hebben wanneer we de consumptie voor deze huishoudens voorspellen. We kunnen dus besluiten dat we via de clustering een beter zicht krijgen op welke huishoudens relatief moeilijk of makkelijk te voorspellen zijn.



Figuur 6.21: Clustering van weekprofielen van de eerste vier weken van februari van drie huishoudens. We gebruiken $X.1, X.2, X.3$ en $X.4$ om de weekprofielen van huishouden X aan te duiden.

6.2.3 Praktisch gebruik voor bedrijven in de energiesector

Na overleg met een domeinexpert van 3E kunnen we besluiten dat de voorgestelde GPRC methode een bedrijf in de energiesector op verschillende manieren bijstand kan bieden om essentiële taken efficiënter te maken.

- Het tweede experiment toont hoe we de consistente huishoudens kunnen detecteren. Voor deze huishoudens zullen de voorspellingen nauwkeuriger zijn. Door hier rekening mee te houden biedt de clustering een sterke ondersteuning voor de voorspelling, waardoor deze beter uitgevoerd zal kunnen worden. Daarnaast kan deze informatie ook nuttig zijn om energiepakketten voor te stellen aan de huishoudens.

- Het eerste experiment toont hoe we huishoudens kunnen clusteren in functie van hun elektriciteitsgebruik. Dit stelt ons in staat het huishouden te vergelijken met gelijkaardige huishoudens. Dit leidt tot een eerlijke *benchmark* van de elektrische consumptie per cluster.

- Zowel het eerste als het tweede experiment kan helpen bij anomalie-detectie. Bijvoorbeeld, wanneer een warmtepomp grote repetitieve pieken vertoont, is dit slecht voor de energie performantie, de warmtepomp zelf en het elektriciteitsnet. Zulke inefficiënties kunnen gedetecteerd worden in een opzet als het eerste experiment door de huishoudens in de aparte clusters te controleren. Of in experiment twee door te checken of een bepaalde week plots apart geclusterd wordt.

Daarnaast is onze methode schaalbaar en is er geen parameter tuning nodig door de gebruiker voor het leren van het model. Dit vergroot het praktisch nut.

Hoofdstuk 7

Besluit

In dit laatste hoofdstuk zullen we de thesis afsluiten met een conclusie van het geleverde werk. We begonnen met de nodige achtergrond te bespreken in Hoofdstuk 2. In Hoofdstuk 3 toonden we een literatuurstudie, gevolgd door de uiteenzetting van de voorspellingsmethodiek met Gaussiaanse processen in Hoofdstuk 4. Hoofdstuk 5 besprak de theoretische achtergrond voor het clusteren met Gaussiaanse proces regressie (GPRC) en in Hoofdstuk 6 werden de experimenten en hun resultaten getoond.

7.1 Resultaten

Het doel van dit werk is een uitgebreid onderzoek voeren naar het gebruik van Gaussiaanse processen voor het voorspellen en het clusteren van energieconsumptie. Hiervoor hebben we de thesis onderverdeeld in een voorspellings- en een clustergedeelte en enkele onderzoeksvragen vooropgesteld. We hebben de methoden getest op een dataset van het energieverbruik van 71 huishoudens en zullen de belangrijkste resultaten en de antwoorden op de onderzoeksvragen nu expliciet overlopen.

Voorspelling

Voor het voorspellingsgedeelte hebben we volgende onderzoeksvragen opgesteld:

1. Wat is het effect van de features op een auto-regressieve vs niet-auto-regressieve methode?

2. Hoe goed is de voorspelling bekomen door de Gaussiaanse proces regressie in vergelijking met andere technieken?

Om onderzoeksvraag 1 te beantwoorden voegen we de consumptie van twee dagen geleden toe als feature (vertraagde variabele). Deze heeft een negatieve auto-correlatie met de te voorspellen consumptie. We concluderen dat de voorspelling van onze auto-regressieve Gaussiaanse proces regressie methode verbetert. De voorspelling van de niet-auto-regressieve methoden (lineaire regressie en *support vector regressie*) wordt minder accuraat. Dit verklaren we doordat de auto-regressieve methode zelf

de seizoenseffecten zal ontdekken, terwijl de regressiemethoden voor de extra feature ook een verband zullen zoeken dat er niet expliciet is voor die bepaalde momenten. Het antwoord op onderzoeksvraag 2 vinden we door de voorspellingen van de verschillende methoden te evalueren aan de hand van hun gemiddelde relatieve (en absolute) fout. We hebben de voorspellingen opgedeeld in de voorspellingen van de warmtepomp, de toestellen en de som van deze twee. Wanneer we de vertraagde variabele gebruiken in de feature vector concluderen we dat *Support vector regressie* (SVR) en de baseline methode de beste methoden zijn voor het voorspellen van de gemiddelde consumptie van de warmtepomp van de huishoudens afgaande op de gemiddelde relatieve fout, op de voet gevolgd door GPR. Maar GPR is de beste methode voor het voorspellen van de consumptie van de toestellen en de consumptie van de toestellen + warmtepomp (Tabel 6.4). Wanneer de vertraagde variabele niet wordt gebruikt, merken we dat SVR de beste methode is voor de voorspelling van het verbruik van de warmtepomp en het verbruik van toestellen. GPR blijft de beste methode voor het voorspellen van het verbruik van de warmtepomp + toestellen (Tabel 6.5). Verder hebben we nog een analyse van de fouten per maand en per huishouden uitgevoerd (Sectie 6.1.2).

Clustering

Voor het clustergedeelte hebben we volgende onderzoeksvragen opgesteld:

1. Hoe kunnen we de Gaussiaanse proces regressie gebruiken om tijdreeksen (data doorheen de tijd verzameld) te clusteren?

2. Hoe goed is de clustering in vergelijking met andere technieken?

3. Welke inzichten kunnen we detecteren met deze clustermethode?

4. Hoe kan deze clustering gebruikt worden in de praktijk?

Onderzoeksvraag 1 wordt uitgebreid behandeld in hoofdstuk 5. Hierin bespreken we hoe we GPR aanpassen door de som te nemen van de waarschijnlijkheden van de individuele tijdreeksen, waarna we de hyperparameters zoeken die deze som maximaliseerden. Op deze manier bekomen we een algemeen model voor een set van tijdreeksen. Door de foutenwaarden van elke tijdreeks t.o.v. het algemeen model te berekenen verkrijgen we een gelijkvormigheidsmaatstaf waarmee we een recursieve clustering kunnen uitvoeren.
Om onderzoeksvraag 2 te beantwoorden, hebben we onze GPRC methode vergeleken met *k-medoids* + DTW en hiërarchisch agglomeratief clusteren met DTW (HAC + DTW). Omdat de geleverde data geen labels hebben, hebben we geen objectieve maatstaf om de kwaliteit van de clusters te vergelijken. Wel hebben we de looptijden van de algoritmen vergeleken. GPRC heeft een lineaire looptijd, wat een verbetering is t.o.v. de kwadratische looptijd van *k-medoids* + DTW en HAC + DTW.
Om te onderzoeken welke inzichten we kunnen detecteren (onderzoeksvraag 3) maken we gebruik van een interactieve visualisatie die gebaseerd is op de *ETE toolkit* [55].

We concluderen dat de clustermethode in staat is om voor de 71 huishoudens verschillende verbruikspatronen te onderscheiden en te groeperen. Hieruit kunnen we o.a. leren dat een aantal huishoudens een geheel ander verbruik doorheen het weekend vertonen. In een volgend experiment onderzoeken we de (in)consistentie van enkele weken van enkele huishoudens. Hieruit blijkt dat de huishoudens, waarvoor de weken voornamelijk in dezelfde cluster gegroepeerd worden, een consistent energieverbruik hebben, en andersom. We kunnen dit ook terugkoppelen naar de predictie, waarbij we merken dat de huishoudens met een consistent verbruik overeenkomen met de huishoudens waarvoor we een goede voorspelling bekomen.
Als laatste punt onderzoeken we nog het praktisch nut van onze methode (onderzoeksvraag 4). Hiervoor hebben we beroep gedaan op een domeinexpert van het bedrijf 3E. Ten eerste zal de clustering een goede ondersteuning zijn voor de predictie. Ten tweede kunnen we huishoudens eerlijker vergelijken met andere huishoudens, door te vergelijken met huishoudens van dezelfde cluster. Ten derde laten de groeperingen een bedrijf ook toen om anomalie detectie uit te voeren door eigenaardige clusters te inspecteren. Tenslotte kunnen bedrijven door deze groeperingen ook gepersonaliseerde energiepakketten voorstellen aan de huishoudens. Naast al deze zaken is onze methode schaalbaar en is er geen parameter tuning nodig door de gebruiker voor het leren van het model. Dit vergroot het praktisch nut.

7.2 Toekomstig werk

Mogelijk toekomstig werk voor het voorspellen zou zich kunnen richten naar het voorspellen van de energieconsumptie doorheen het jaar. Moesten we data van meerdere jaren hebben, zouden we kunnen toetsen of de seizoenseffecten (en trends) doorheen het jaar geleerd kunnen worden. Hierbij aansluitend zou men een online predictie-algoritme kunnen ontwikkelen in deze context, waarbij het geleerd model na een bepaalde tijd wordt aangepast wanneer er nieuwe data beschikbaar is. Er zou ook verder onderzoek naar andere features kunnen gevoerd worden voor GPR.
Voor de clustering is er ook nog ruimte voor toekomstig werk. Men zou de clustering nog kunnen aanpassen zodanig dat de parameters optimaler gebruikt worden of een complexer gelijkaardigheidscriterium dan de RMSE waarden onderzoeken.
Er zouden ook meer verbanden tussen de clustering en de voorspelling kunnen gezocht worden en hoe we deze kunnen integreren in één systeem om de beste voorspelling en ondersteuning voor bedrijven te creëren. Men zou automatische conclusies kunnen leren uit de combinatie van de predictie en de clustering (o.a. anomalie detectie).
Naast bovenstaande mogelijkheden is het gebruik van het algemene model nog niet onderzocht buiten deze thesis. Hiervoor kunnen dus nog verschillende toepassingen onderzocht worden, zoals bijvoorbeeld het gebruik van dit model in een predictie context.

Bijlagen

Bijlage A

Broncode

In deze bijlagen vindt men de broncode terug die nuttig kan zijn voor de lezer, maar die niet essentieel zijn om het betoog in de normale tekst te kunnen volgen.

A.1 Volledige broncode thesis

De experimenten en algoritmen die besproken zijn in deze thesis zijn geïmplementeerd in Python en kunnen teruggevonden worden in [57]. Daarnaast kunnen ze opgevraagd worden via de begeleiders of de auteur.

A.2 Dynamic time warping

Listing A.1: DTW-algoritme

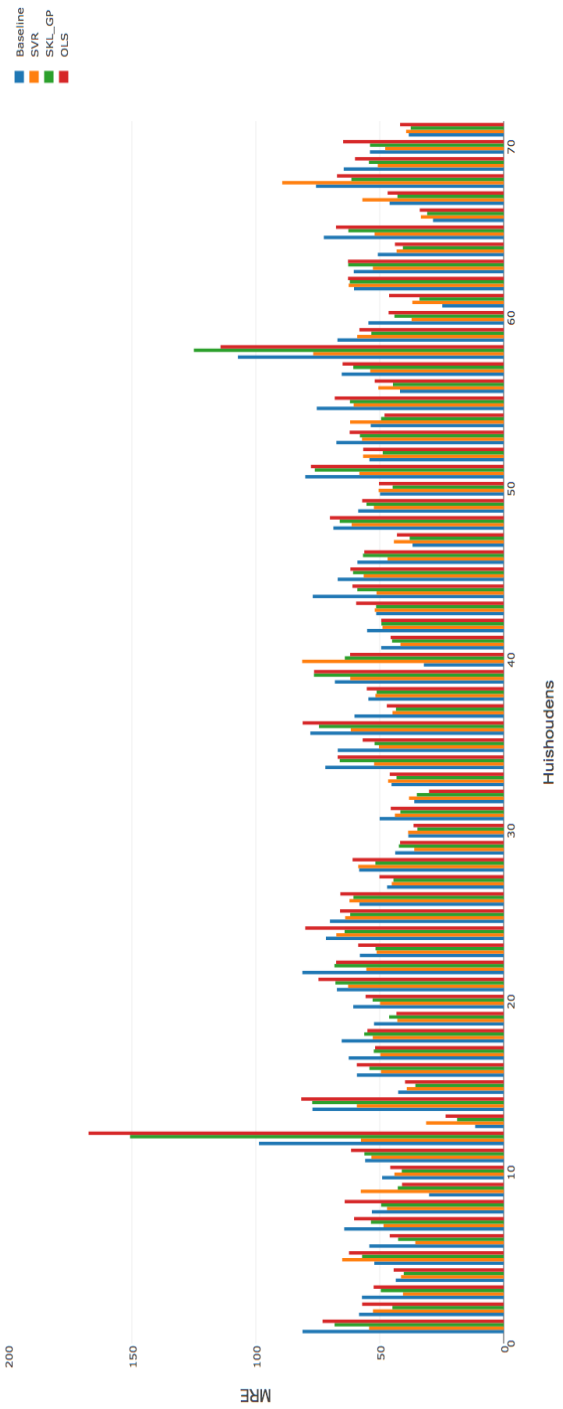
```
int DTWDistance(s: array [1..n], t: array [1..m], w: int) {

    DTW := array [0..n, 0..m]
    w := max(w, abs(n-m)) // adapt window size (*)

    for i := 0 to n
        for j:= 0 to m
            DTW[i, j] := infinity
    DTW[0, 0] := 0

    for i := 1 to n
        for j := max(1, i-w) to min(m, i+w)
            cost := d(s[i], t[j])
            DTW[i, j] := cost + minimum(DTW[i-1, j  ],    // insertion
                                        DTW[i, j-1],    // deletion
                                        DTW[i-1, j-1])    // match
    return DTW[n, m]
}
```

Bijlage B

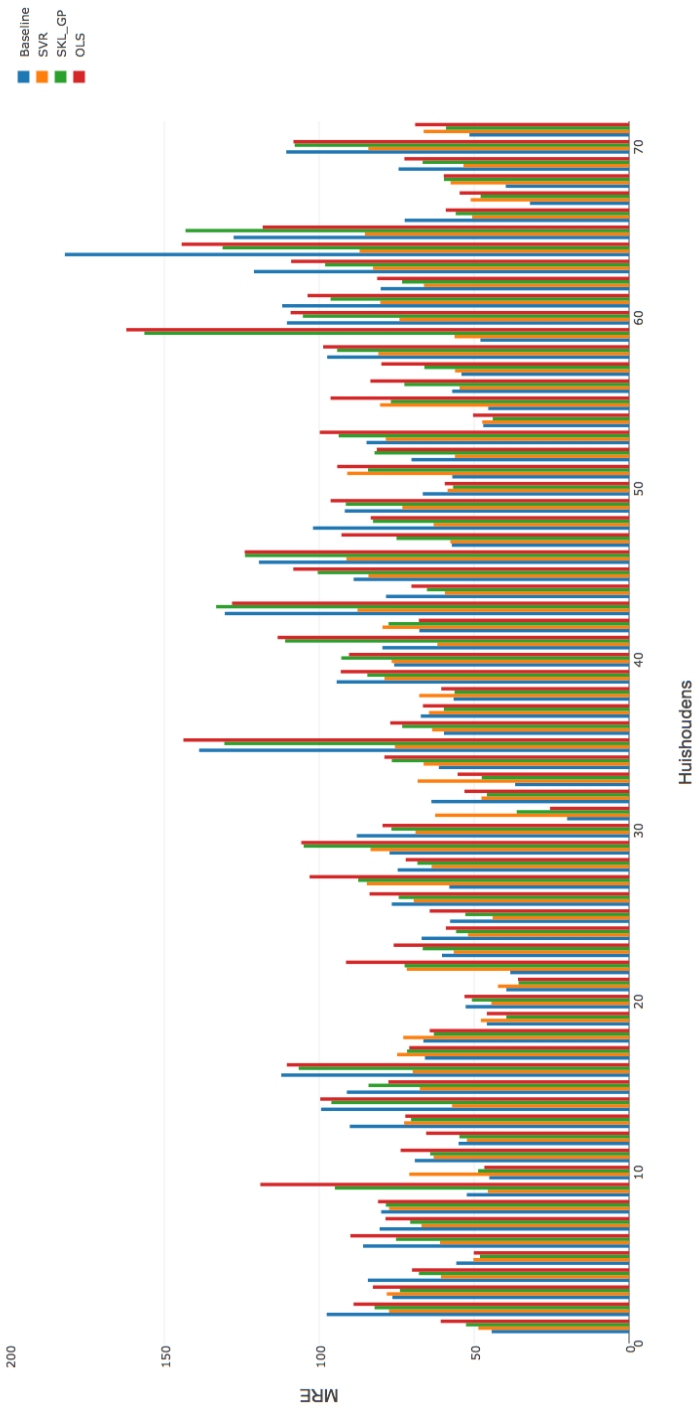
Resultaten

Deze bijlage bevat resultaten van de uitgevoerde voorspellingen en clusteringen.

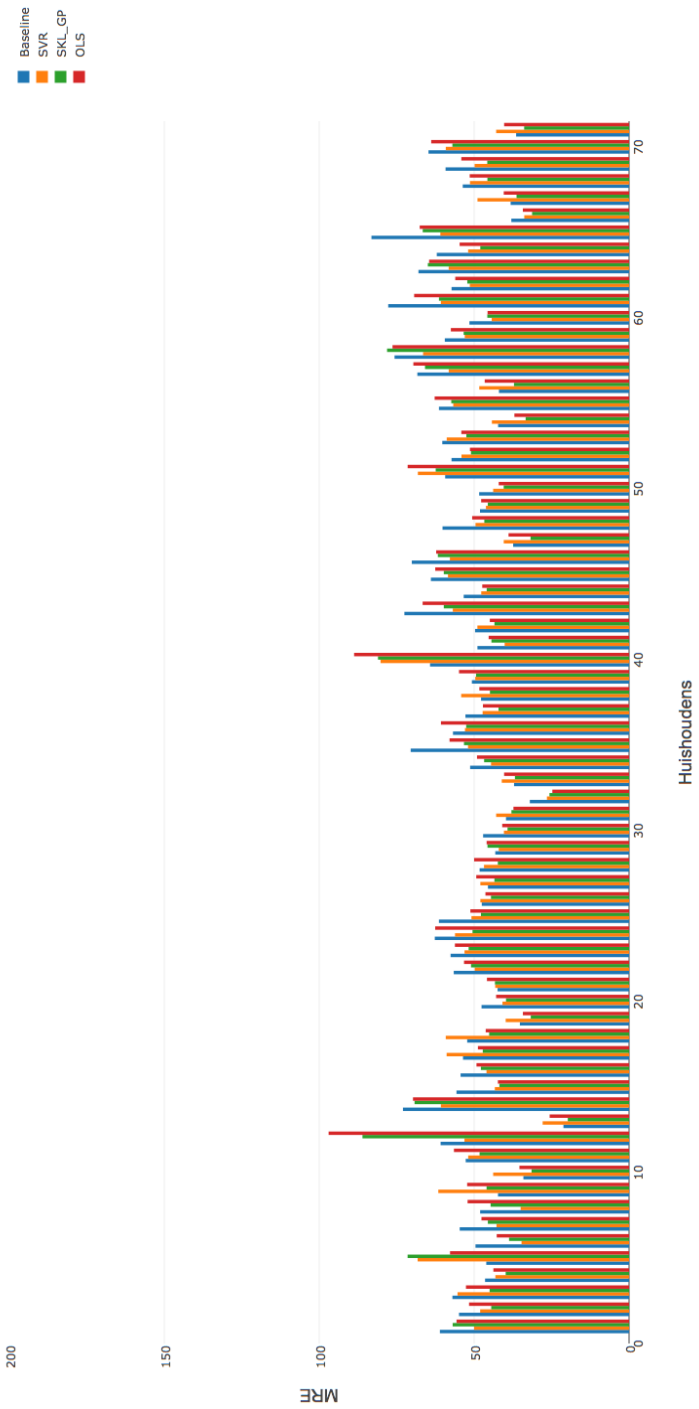
B.1 Voorspellingsresultaten



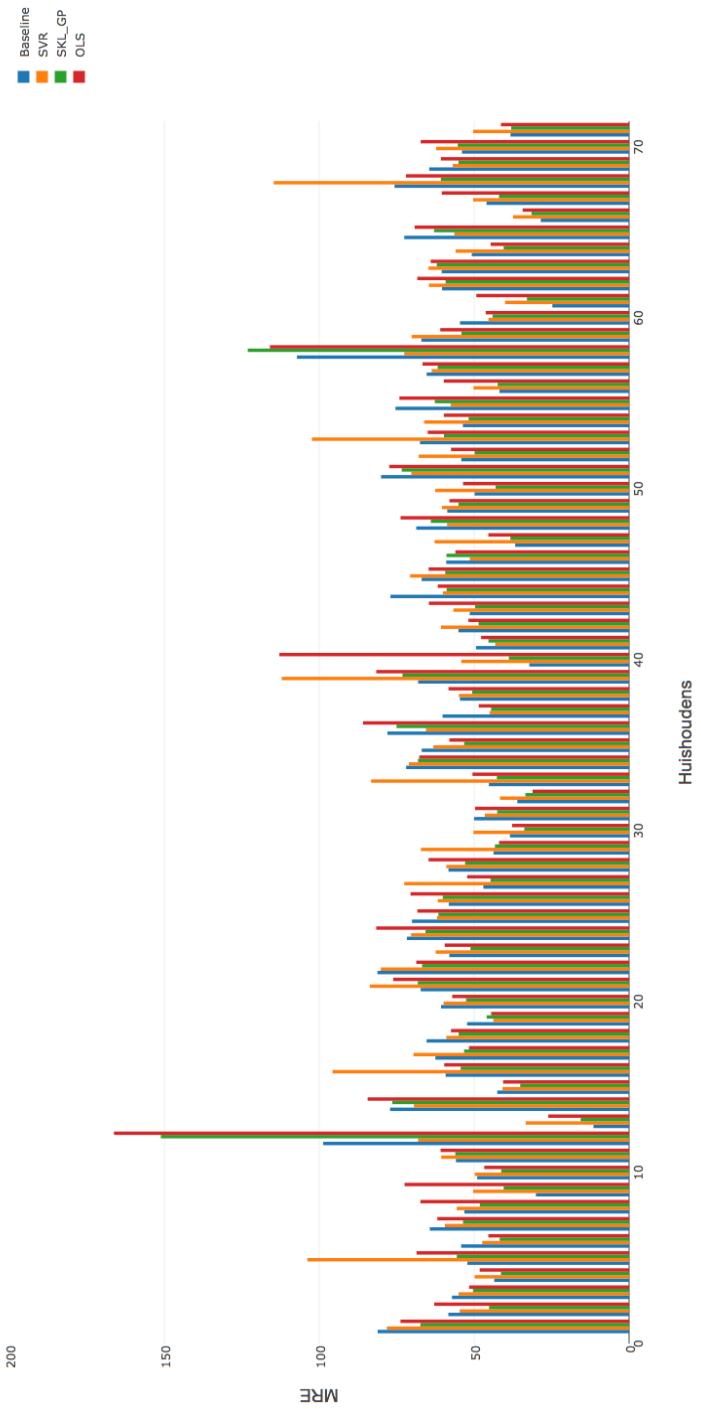
Figuur B.1: Gemiddelde MRE voorspellingsfout van de consumptie van de toestellen per huishouden (consumptie van twee dagen geleden niet in featurevector)



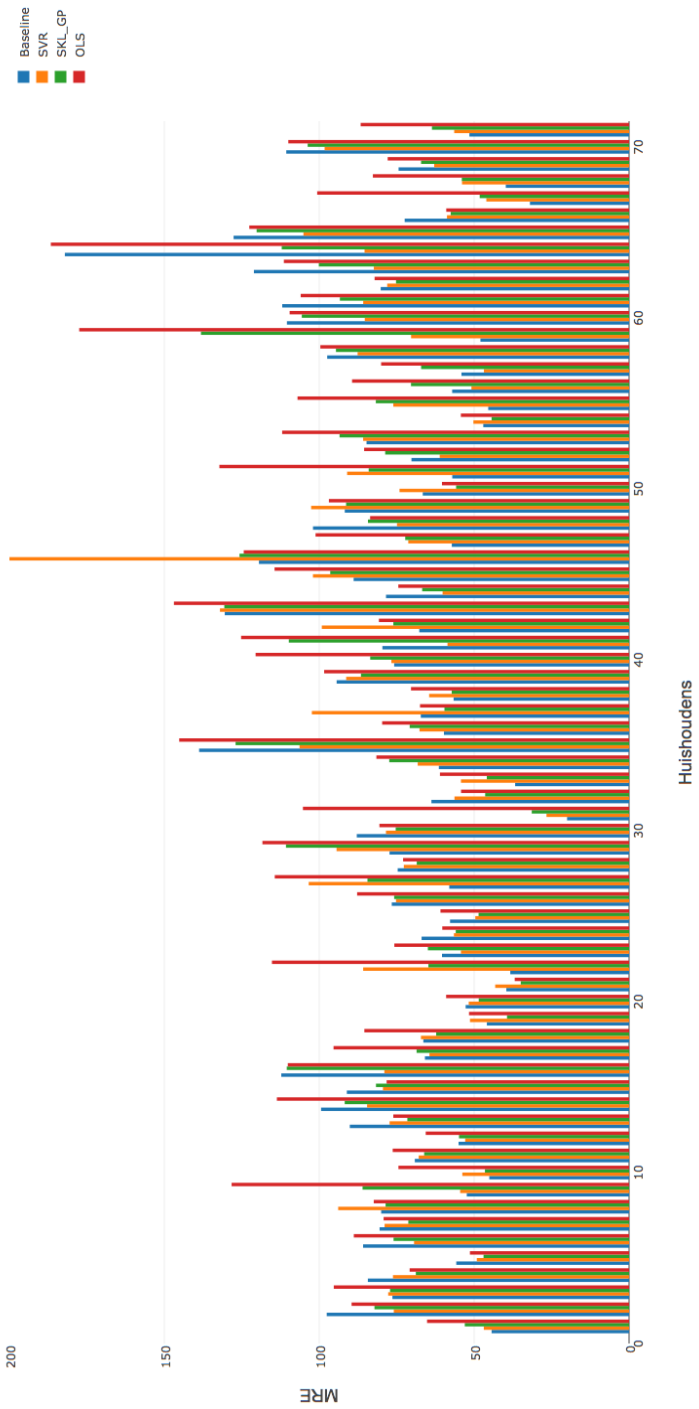
Figuur B.2: Gemiddelde MRE voorspellingsfout van de consumptie van de warmtepomp per huishouden (consumptie van twee dagen geleden niet in featurevector)



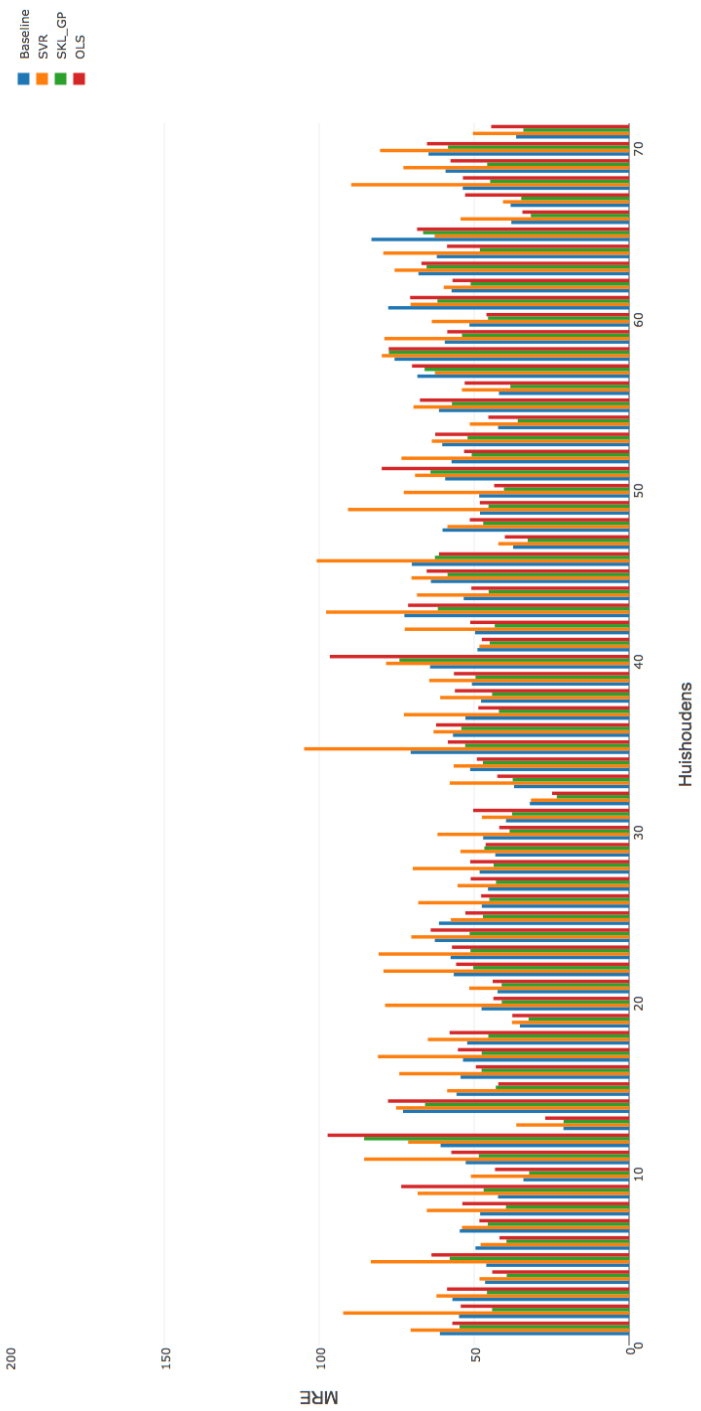
Figuur B.3: Gemiddelde MRE voorspellingsfout van de consumptie van de toestellen + warmtepomp per huishouden (consumptie van twee dagen geleden niet in featurevector)



Figuur B.4: Gemiddelde MRE voorspellingsfout van de consumptie van de toestellen per huishouden (consumptie van twee dagen geleden in featurevector)



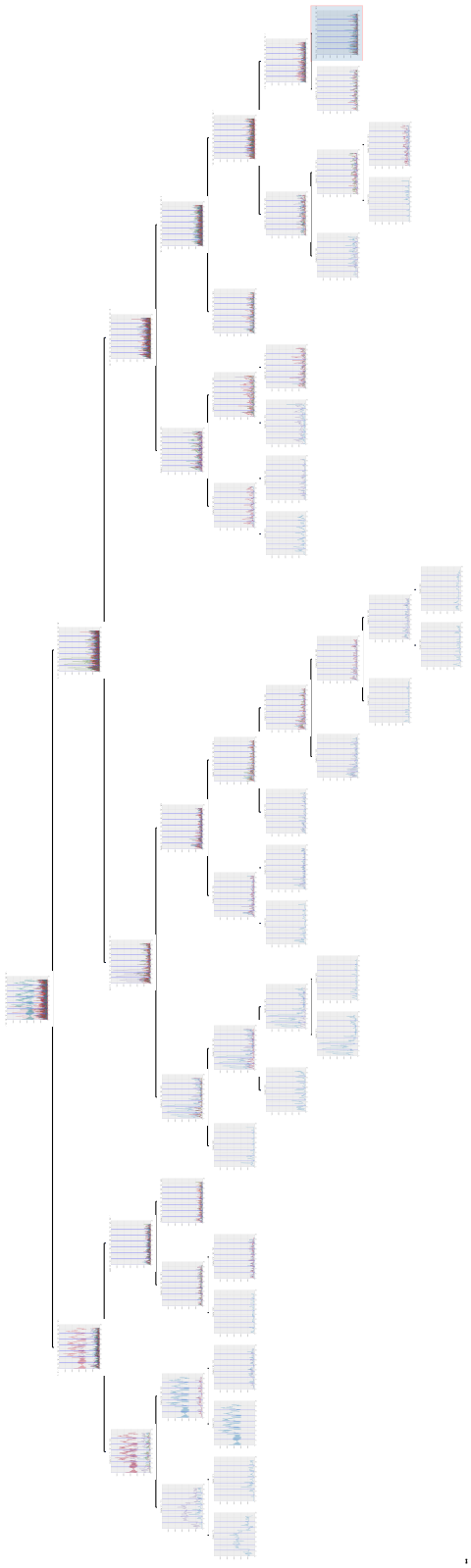
Figuur B.5: Gemiddelde MRE voorspellingsfout van de consumptie van de warmtepomp per huishouden (consumptie van twee dagen geleden in featurevector)



Figuur B.6: Gemiddelde MRE voorspellingsfout van de consumptie van de toestellen + warmtepomp per huishouden (consumptie van twee dagen geleden in featurevector)

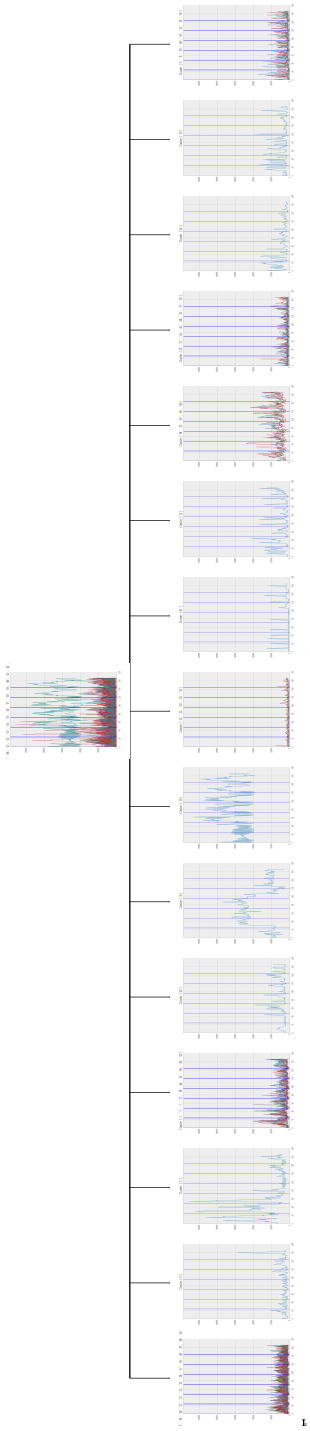
B.2 Clusterresultaten

B.2.1 GPRC methode



Figuur B.7: Clustering bekomen via de GPRC methode.

B.2.2 *k-medoids* met DTW



Figuur B.8: Clustering bekomen via *k-means* met DTW.

B.2.3 *Hiërarchisch agglomeratief clusteren* met DTW

Figuur B.9: Selectie van de clustering bekomen via hiërarchisch agglomeratief clusteren met DTW.

Bijlage C

Wetenschappelijk artikel

Energy consumption profiling using Gaussian Processes

Christiaan Leysen*, Mathias Verbeke†, Pierre Dagnely†, Wannes Meert*
*Dept. Computer Science, KU Leuven, Belgium
†Data Innovation Team, Sirris, Belgium

***Abstract*—We present a novel clustering approach for time series based on Gaussian process regression in order to discover insights in the spending habits of households. The advantage of the proposed method is that it avoids the pairwise comparison of time series, employed by many existing methods. To this end, it learns a generalized model on several time series at once, based on their likelihood. We have validated our method using a real-world energy consumption dataset of 71 households and compared it with K-medoids and agglomerative clustering, using dynamic time warping. We not only show that our method is superior in terms of scalability but also that the produced results are useful in the decision making process of a company.**

***Keywords*—Gaussian process regression; Clustering; Time series; Energy consumption profiling**

I. Introduction

Accurate prediction of household energy consumption is of great value to energy companies as it is crucial for estimating the aggregated energy that needs to be bought by these companies. The spending behaviour of a household depends on a number of factors (e.g., family size, presence of airconditioning) and contains a lot of variation. Information about these variables leads to a better understanding of the spending habits of an energy company's clients. Often, however, these variables are not known or it is unclear what variables should be investigated and measured. It is in this context that we propose an algorithm for clustering household energy consumption data. These clusters can be used to derive spending profiles of the households or to recommend households certain energy packages. Not only the relations between households but also the relationship between consumption periods are interesting to investigate, e.g. to detect anomalies. In addition to energy companies, this information is also of interest to the households themselves, since it gives insights in their spending behaviour throughout the year.

Energy consumption is naturally represented as a time series. The households that end up in the same cluster do not necessarily have an identical consumption but their spending behaviour follows the same temporal patterns. A model of this behaviour can be learned by auto-regression methods to predict future values. In this work we will compare the learned patterns to group households together rather than to predict future values. Since we do not know upfront what time periods are relevant or how many patterns are superimposed, a method that learns this model based on the available data is preferred. For this reason and the fact that there is no need for parameter tuning by the user, we selected Gaussian Processes (GPs).

Out-of-the-box GP implementations requires a single function as input while we consider a set of functions, as we want to learn a model for multiple households together. To overcome this, we present a new approach to jointly learn a GP model and optimize the hyperparameters over a set of functions, which is the first contribution of this paper. Second, we embed this in a hierarchical clustering method for time series. Finally, we present an interactive application and show how this can be used by energy providers to analyse and predict customer behaviour.

In order to test our approach we propose two experiments. In our main experiment we cluster the energy consumption of 71 distinct households to investigate which households have similar spending habits. Next we compare the time complexity of our method with K-means and agglomerative clustering, both using dynamic time warping as a similarity measure. In a second experiment, we test our method on data from four different weeks per households (for three households) in order to find households with steady spending habits.

II. Related work

Gaussian Processes (GPs) have been used previously for clustering. This work, however, is the first to learn over a set of functions to characterize the temporal behaviour of a cluster. Pimentel et al. [1] learn a GP only after aggregating time series. Kim et al. [2] employ a clustering method based on the variance function of Gaussian process regression in combination with a reduced complete graph strategy. However, this method clusters feature vectors instead of time series. Kumar et al. [3] propose a distance function based on the assumed independent Gaussian models of data errors and used a hierarchical clustering method to group seasonality sequences into a desirable number of clusters. The work of Duvenaud [4] presents a thorough investigation of GPs for automatically constructing, visualizing and describing a large class of models, useful for forecasting and finding structure inside time series using GPs. The clustering methods investigated in this work are also for feature vectors, not time series.

In addition to GPs, a variety of other techniques have been applied to cluster time series. Liao et al. [5] provides an overview of raw-data-based, feature-based and model-based clustering techniques. This survey concludes that until now the focus was on techniques based on statistical (e.g. ARIMA) and probabilistic methods (e.g. Dynamic Bayesian nets). This work contributes GPs, which fit within the class of model-based techniques. Mostly statistical methods have been applied on energy data [6]. The advantage of GPs as used in this work is that these do not require domain knowledge to set hyperparameters.

Rani et al. [7] provide a survey of recent techniques to cluster time series and among others describe the use of K-means clustering with different similarity measurements. Using L*-norms with K-means is a general method but one has to consider the curse of dimensionality. Recent work of Lavin et al. [8] uses K-means with L*-norms to group and identify patterns in $k = 9$ energy profiles of a number of customers. The authors show positive results for vectors representing 24 hour periods with 15 minute intervals. When using Dynamic Time Warping (DTW) [9] as a distance metric [10], however, K-means fails to give correct results because it averages the shape of the time series that may be partially shifted [11].

Another method, named k-shape proposed by Paparrizos et al. [12] is a novel centroid-based clustering algorithm that uses the cross-correlation as similarity measurement. It outperforms all scalable and non-scalable partitional, hierarchical, and spectral methods in terms of accuracy. With as only exception K-medoids with DTW which achieves similar results. However, K-medoids needs to compute the similarity matrix, which makes it harder to scale. The best performing shape-based approaches from the literature are partitional methods combined with scale- and shift-invariant distance measures. Among partitional methods, K-medoids [13] is the most popular method as it enables the easy adoption of any shape-based distance measure.

III. Background

A. Gaussian process regression

We will first describe Gaussian processes, following the description given by Rasmussen and Williams [14]. A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution, and can be completely specified by its mean $m(x)$ and covariance function $k(x, x')$:

$$m(x) = \mathbb{E}[f(x)], \tag{1}$$

$$k(x, x') = \mathbb{E}[(f(x)m(x))(f(x')m(x'))], \tag{2}$$

The Gaussian process can then be written as

$$f(x) \sim GP(m(x), k(x, x')). \tag{3}$$

The covariance or kernel function can be seen as a similarity metric. It maps a pair of inputs (x, x') into $\mathbb{R}$. This Gaussian process model can be used for regression purposes. Suppose we have a training set $D = (x_i, y_i)|i = 1, 2, ..., n$ with observations subjected to some noise. If we want to predict a new target y_{new} given some new input data x_{new} we have to learn the underlying function which describes the training input data, assuming a Gaussian prior. The observation and the underlying function values are not identical as typically there is noise ϵ. Thus, we describe the target values as follows:

$$y = f(x) + \epsilon \tag{4}$$

with the assumption that the Gaussian noise model can be described as $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$.This noise assumption together with the model directly gives rise to the likelihood, i.e., the probability density of the observations given the parameters, which is factored over cases in the training set (because of the independence assumption) to give

$$p(y|X, w) = \mathcal{N}(X^T w, \sigma_n^2 I) \tag{5}$$

In Bayesian formalism there is need for a prior to express the beliefs about the parameters before the observations are observed. We put a zero mean Gaussian prior with covariance matrix Σ_p on the weights $w \sim N(0, \Sigma_p)$. Inference in the Bayesian linear model is based on the posterior distribution over the weights, computed by Bayes rule. The marginal likelihood (normalizing constant) is the integral of the likelihood times the prior. The posterior combines the likelihood and the prior, and captures everything we know about the parameters.

Specifically, we introduce the function $\phi(x)$ which maps a d-dimensional input vector x into an n dimensional feature space. Further, let the matrix $\Phi(X)$ be the aggregation of columns $\phi(x)$ for all cases in the training set and we define the covariance matrix $K = \Phi^T \Sigma_p \Phi = K(X, X)$ which also depends on the used kernel function and the σ_f^2 signal variance. Note that we shorten $\Phi(X)$ to Φ and $K(X, X)$ to K to simplify notation. Stating this, we can now outline the algorithm used for Gaussian process regression (Alg. 1).

Algorithm 1 Predictions and log marginal likelihood for Gaussian process regression.

INPUT: $X\ (inputs)$, $y\ (targets)$, $k\ (covariance\ function)$, $\sigma_n^2\ (noise\ level)$, $\sigma_f^2\ (signal\ variance)$, $l\ (length\ scale)$, $x_*\ (test\ input)$, $m(X)\ (mean\ function)$

1: $L \leftarrow cholesky(K + \sigma_n^2 I)$
2: $\alpha \leftarrow L^T \setminus (L \setminus y)$
3: $\overline{f}_* \leftarrow m(X) + k_*^T \alpha$
4: $v \leftarrow L \setminus k_*$
5: $\mathbb{V}[f_*] \leftarrow k(x_*, x_*) - v^T v$
6: $log\ p(y|X) \leftarrow -\frac{1}{2} y^T \alpha - \sum_i logL_{ii} - \frac{n}{2} log 2\pi$

RETURN: $\overline{f}_*\ (mean), \mathbb{V}[f_*]\ (variance)$, $logp(y|X)\ (log\ marginal\ likelihood)$

The computational complexity is $n^3/6$ for the Cholesky decomposition in line 1, and $n^2/2$ for solving triangular systems in line 2 and (for each test case) in line 4 [14]. The algorithm uses the Cholesky decomposition instead of calculating the inverting the matrix directly, since it is faster and numerically more stable.

B. Hyperparameters

The covariance or kernel function typically has some parameters that need to be set, i.e., the hyperparameters of the GP. These hyperparameters are important for fitting the given data. The kernel functions used in this work are the linear kernel function and the squared-exponential kernel function. The linear kernel function has only one parameter, the signal variance σ_f^2. The squared-exponential kernel function has as parameters the signal variance σ_f^2, the noise variance σ_n^2 and the length scale l. These parameters can be learned from the data [14]..

Algorithm 2 Overall likelihood over all time series.

INPUT: TSS $(TimeSeriesSet:$ $inputs\ and\ targets\ set)$, k $(covariance\ function)$, σ_n^2 $(noise\ level)$, σ_f^2 $(signal\ variance)$, l $(lengthscale)$

1: $likelihood_{overall} \leftarrow 0$
2: ***for*** (X, y) ***in*** TSS ***do***
3: $L \leftarrow cholesky(K + \sigma_n^2 I)$
4: $\alpha \leftarrow L^T \setminus (L \setminus y)$
5: $-\log\ p(y|X) \leftarrow \frac{1}{2} y^T \alpha + \sum_i \log L_{ii} + \frac{n}{2} \log 2\pi$
6: $likelihood_{overall} \leftarrow likelihood_{overall} + (-log\ p(y|X))$

RETURN: $likelihood_{overall}$ $(overall\ likelihood)$

IV. Method

A. Generalized model

The method we propose for clustering time series data makes use of the Gaussian process regression method. Instead of learning a model, describing only one time series, we create a model which describes a set of time series. This is a non-trivial step because the GP accepts a *single* function, whereas we want to consider *multiple* time series of the same time. In order to create a generalized model, that describes multiple time series, we make use of the likelihood which is calculated when we perform Gaussian process regression. In order to learn the generalized model which maximizes the overall likelihood of the set of time series, we optimize the hyperparameters of the generalized model. In order to do so, we introduce a new function (Alg. 2). Note that K is the covariance matrix, which depends on the input values and the used kernel function (which contains the hyperparameters σ_f^2, σ_n^2 and l).

As generalized model, we want to find the model for which the likelihood is maximal. As Alg. 2 returns the negative log likelihood, we thus need to find the hyperparameters for which the function, described in Alg. 3, is minimal. We use $TSS.X$ and $TSS.y$ to denote the set of timestamps and the set of target values of the time series set TSS respectively. By making use of the optimal hyperparameters $(\sigma_{n*}^2, \sigma_{f*}^2, l_*)$, we can calculate the optimal L and α. Using these results and the means $\overline{y}$ of the set of target values $TSS.y$, we can calculate the generalized model by making a prediction for the same period. Note that we assume that all time series X_i of the set $TSS.X$ run over the same period, so we just need to pick a random row from this set, e.g., the first one (line 3, Alg. 3).

B. Clustering

For the subsequent clustering we employ a recursive clustering approach. The method, as outlined in Alg. 5, uses the root mean square error (RMSE) between the observed time series and the generalized model. This is used as a measurement of the similarity between the time series. The clustering method makes use of three parameters: 1) a minimum cluster size (s_{min}) if we would like to have a minimum number of time series inside a cluster, 2) a similarity threshold (t_{sim}) to check if the mean similarity of the time series inside the cluster is higher or equal to the similarity threshold, and 3) a split ratio (r) used for the number of samples that need to be split into another cluster when above conditions are met (using the approach described in Alg. 4). Else we return the original cluster and check whether the newly formed cluster is different from the input list of time series. When this is not the case we return the original time series list (line 6-7, Alg. 5). Note that line 2 of Alg. 4 is a normalization step of the RMSE values of the cluster. Also note that the time series set (TSS) is reversely ordered by the corresponding RMSE values of the time series (line 4, Alg. 5). The reason we reversely order the time series set by the RMSE value is that we remove the time series with the biggest RMSE value first, based on the split ratio (r) (line 4, Alg. 4).

Algorithm 3 Find generalized model

INPUT: TSS $(TimeSeriesSet:$ $inputs\ and\ targets\ set)$, k $(covariance\ function)$, σ_{ni}^2 $(initial\ noise\ level)$, σ_{fi}^2 $(initial\ signal\ variance)$, l_i $(initial\ length\ scale)$

1: $(\alpha, L) \leftarrow minimum(likelihood_{overall}(TSS, k, \sigma_{ni}^2, \sigma_{fi}^2, l_i))$
2: $\overline{y} \leftarrow mean(TSS.y)$
3: $x_* \leftarrow$ $firstRow(TSS.X)$
4: $\overline{f}_* \leftarrow \overline{y} + k_*^T \alpha$
5: $v \leftarrow L \setminus k_*$
6: $\mathbb{V}[f_*] \leftarrow k(x_*, x_*) - v^T v$

RETURN: $\overline{f}_*$ $(mean), \mathbb{V}[f_*]$ $(variance)$

C. Complexity

If we assume that one time series consists of N (equidistant) samples, the learning part of the Gaussian process regression of N samples has a computational cost of $O(N^3)$, while for predicting this is $O(N^2)$ [14]. When we have S time series, this needs to be repeated S times (Alg. 3). The resulting overall cost is $O(SN^3)$ plus the cost of the minimization $O(S)$. For the clustering presented here, N is fixed whereas the number of examples or series varies. Compared to pairwise approaches such as dynamic time warping, this reduces the complexity of $O(S^2)$ to $O(S)$, resulting in a better scalability. Other methods and their complexities can be found in Section V-D.

V. Experimental setup

We conducted two experiments that are useful to energy providers. In our main experiment we cluster distinct house-

Algorithm 4 Dividing a cluster

INPUT: TSS $(TimeSeriesSet:$ $inputs\ and\ targets\ set)$, E_r $(list\ of\ RMSE\ values)$, t_{sim} $(similarity\ threshold)$, s_{min} $(minimum\ cluster\ size)$, r $(split\ ratio)$

1: **if** $size(TSS) > s_{min}$ **then**
2: $E_r' \leftarrow E_r / max(E_r)$
3: **if** $mean(E_r') < t_{sim}$ **then**
4: $(C_1, C_2) \leftarrow split(TSS, E_r', r)$
5: **return**(C_1, C_2)
6: **else**
7: **return** TSS
8: **else**
9: **return** TSS

RETURN: C $(Clustering)$

Algorithm 5 Clustering of time series

INPUT: TSS $(TimeSeriesSet:\ inputs\ and\ targets\ set)$, k $(covariance\ function)$, σ_{ni}^2 $(noise\ level)$, σ_{fi}^2 $(signal\ variance)$, l_i $(length\ scale)$, t_{sim} $(similarity\ thres$ s_{min} $(minimum\ cluster\ size)$, r $(split\ ratio)$

$(\overline{f}_*, \mathbb{V}[f_*]) \leftarrow findGeneralModel(TSS, k, \sigma_{ni}^2, \sigma_{fi}^2, l_i)$
for (X, y) **in** TSS **do**
  $E_r \leftarrow RMSE(y, \overline{f}_*)$
$TSS \leftarrow reverseOrderBy(TSS, E_r)$
$(C_1, C_2) \leftarrow divide(TSS, E_r, t_{sim}, s_{min}, r)$
if $C_1 == TSS$ *or* $C_2 == TSS$ **then**
  return TSS
else
  $cluster(C_1, k, \sigma_{ni}^2, \sigma_{fi}^2, l_i, t_{sim}, s_{min}, r)$
  $cluster(C_2, k, \sigma_{ni}^2, \sigma_{fi}^2, l_i, t_{sim}, s_{min}, r)$

RETURN: C $(Clustering)$

holds based on their time series from a particular week in order to investigate which households have similar spending habits. Next we compare the time complexity of our method with two other methods. In a second experiment, we test our method on data from four different weeks per household (for three households) in order to find households with steady spending habits.

A. Energy Consumption Data

Our main experiment uses historical electrical consumption data of 71 distinct households. This data was provided by 3E[1], a company operating in sustainable energy consulting, research and software. The provided data, gathered in the context of the FLEXIPAC project [15], spans over one year and is comprised of the consumption of all the appliances and the heat pump of the households. A resampling to hourly intervals was applied over the time and the consumption was normalized to focus on shape and patterns.

To compare across different weeks, the timestamp was split into 'day of week' and 'hour of day'. As a convention, the week starts on Saturday to clearly visualize the weekend behaviour and to show the transition from weekend to workweek. The target variable is the aggregated consumption data. This results in $N = 168$ samples per week and $S = 71$ time series. To compare households, the same week was selected such that the weather conditions are similar. To analyse changes in usage patterns, different weeks of the same households were selected.

B. Extending pyGPs to learn over multiple functions

For the Gaussian process regression we employed the pyGPs package [16]. We extended this package to learn and optimize (hyper)parameters over multiple functions (Alg. 2) and compute predictions (Alg. 3). To find the maximized likelihood of the set of time series (Alg. 3) we relied on the L-BFGS-B algorithm implemented in SciPy [17]. As kernel function we used an addition of the squared-exponential kernel function and a linear kernel function. This combination gave the best accuracy

[1]http://www.3e.eu

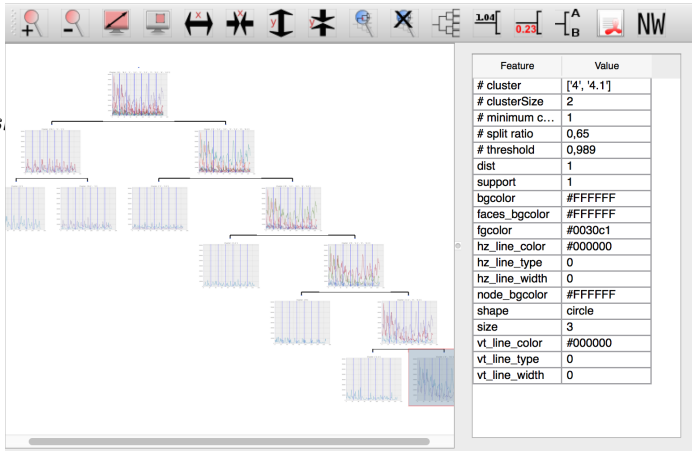


Fig. 1. Screenshot of the interactive visualization based on the ETE toolkit

when performing prediction. The initial hyperparameters used are: $\sigma_{fi(LIN)} = \sigma_{fi(EXP)} = l_i = 10^{-7}; \sigma_{ni} = e^{(2*log(0.1))}$.

C. Interactive Visualization

In order to inspect the clustering we adapted the ETE toolkit [18] to visualize the resulting clustering of the data. We converted the clustering result from Alg. 5 to the Newick tree format [19]. For each of the clusters (leaves of the tree), we create an image of the plot of the time series of the cluster. For the internal nodes, we create images of the union of the sets of time series of the underlying nodes. These images are placed in the visualization of the tree. The visualization is fully interactive and each node of the clustering is clickable and provides extra information about it (Fig. 1). This allows a user to easily inspect the resulting clustering.

D. Clustering approaches

We selected a number of clustering approaches to cluster the consumption time series of 71 distinct households.

1) Clustering using Gaussian process regression: Gaussian process regression models were learned on a set of time series. The predictive quality of the overall model with respect to each individual time series is used to select and split the set of time series to achieve a recursive clustering approach (Sec. IV). It is important to choose a good similarity threshold. When the similarity threshold is too large, the mean of the normalized RMSE values will be smaller than the similarity threshold and the clusters will continue to divide until the minimum cluster size is reached. When the threshold is chosen too small, the mean of the normalized RMSE values will always be greater and no new clusters will be formed because the algorithm rules that all clusters have already a high enough similarity. For the complexity experiment, we used as minimum cluster size 1 (to be able to detect unique profiles), a similarity threshold of 0.99 and a split ratio of 0.2. For the clustering of households with similar consumption we used 1, 0.99, 0.25 respectively and for checking the (in)consistent consumption behaviour we used 1, 0.98 and 0.2 respectively. For the latter we used a smaller cluster similarity to slightly loosen the similarity demand because the time series are from different weeks. Also,

the split ratio is smaller to accommodate the smaller number of households. Note that our clustering method is deterministic, so fixed parameters will always produce the same clustering.

2) K-medoids clustering using dynamic time warping: The Partitioning Around Medoids (PAM) [13] variant of K-medoids is an existing method for clustering time series. As similarity measurement it uses DTW [9]. Its time complexity is $O(K(S-K)^2I)$ with K number of clusters, S number of time series and I number of iterations to converge [20]. To compare two temporal sequences of length N and M respectively, DTW evaluates the local cost measure for each pair of elements, resulting in a cost matrix. The time complexity of this method is $O(N^2)$ if we assume time series of the same length [10]. Calculating the similarity matrix for K-medoids using DTW, has a time complexity of $O(N^2S^2)$, thus quadratic in the number of time series. For our experiment we used the PAM algorithm described by [21] in combination with the standard DTW algorithm using the Euclidean norm as distance. A big disadvantage of this method is that the number of clusters needs to be specified a priori, which is hard, especially in our case where we do not know exactly how many clusters are desirable for getting an insight in the households. We used $K = 15$ (and $K = 5$ for the test of 10 households) based on several results of our own method. Note that we did not use the K-means algorithm because it is discouraged to use this algorithm in combination with DTW (Sec. II).

3) Hierarchical clustering using dynamic time warping: We use bottom-up hierarchical clustering or agglomerative clustering using single-linkage with a complexity of $O(S^2)$ where S is the number of time series that need to be clustered [22], [23]. The calculation of the distance matrix, with complexity $O(N^2)$, can be done separately so the overall complexity is $O(S^2N^2)$.

VI. Results

A. Households with similar consumption behaviour

We select the same week for every household and apply the selected clustering algorithms. Selecting the same week allows us to assume that these households experience similar weather conditions due to their geographical distribution. When applying GP clustering we obtain the results shown in Fig. 2. From Fig. 2, it can be seen that the left branch contains households with different spending habits during the weekend (the weeks start at Saturday in this plot). In the right branch and the branches beneath the first left node, we find that households with the same day/night habits are clustered together. The result of our method was validated by a domain expert. We could not compare its quality with the clusterings of the other methods due to the lack of a suitable measure. A hierarchical clustering is preferred because it gives a more structured view on what households are (dis)similar. The visualization allows to interpret distance in the tree as a proxy for the distance between two spending profiles. An advantage of GP clustering is the beneficial time complexity. Compared to K-medoids DTW and agglomerative DTW clustering, GP is linear instead of quadratic in the number of households as it employs a model based approach (Fig. 3). For datasets less than about 50 time series K-medoids and agglomerative clustering will be faster because the model learning part of our approach takes more time than the calculation of the DTW similarity for the other methods. If more than 50 time series need to be clustered, our approach is faster, because we do not need to do a pairwise comparison of all the time series. This shows the scalability of our approach.

B. Households with (in)consistent consumption behavior

For our second experiment we investigate the spending habits of households during the year. This is achieved by selecting multiple weeks of the same household. For ease of presentation, we only show the clustering for three distinct households and one week in every season (Fig 4) but one can add arbitrarily many households. By inspecting the distribution of a household over different clusters, it is clear that all weeks of household 2 are in the same cluster, and the weeks of household 3 and 1 are spread across two and four clusters respectively. This means that household 2 is the household with most consistent spending habits and household 1 is inconsistent. A visual check of the time series using our interactive tool confirms this. The time series of household X are named $X.1$, $X.2$, $X.3$, and $X.4$ for a Winter, Spring, Summer and Autumn week respectively. We can also conclude that household 3 has similar Autumn and Winter spending habits because they are clustered together. This is also true for the Spring and Summer spending habits.

VII. Practical use in decision support

The proposed method can support companies operating in the energy sector, e.g., 3E, in tackling a number of common tasks more efficiently. The first experiment showed how to cluster households in function of the occupancy (intermittent versus permanent). This allows one to benchmark the electricity consumption per cluster, and only compare households to peers which have a similar consumption behaviour. Another use of the first experiment is to identify inefficient or problematic heating systems. For example, a heat pump with repetitive large peaks in consumption is bad for energy performance, heat pump life time and the electricity grid. Such inefficiencies can be detected by looking for households that are alone in a cluster. The second experiment shows how to detect if the household exhibits a smooth and stable behaviour over time. For such a household, forecasting techniques to predict the consumption day-ahead are more accurate. Furthermore, it

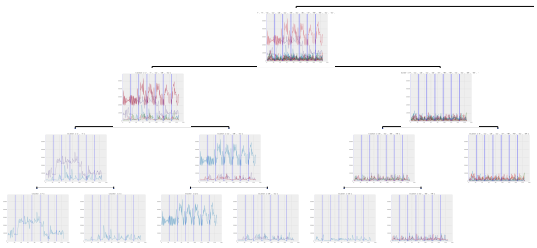


Fig. 2. Cutout of the GPR clustering (full version in appendix).

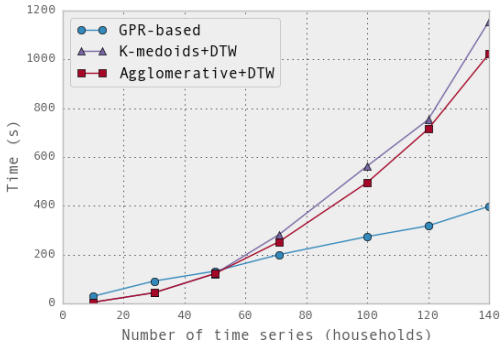


Fig. 3. Run time comparison of the different clustering methods. Multiple weeks were used to obtain up to 140 time series.

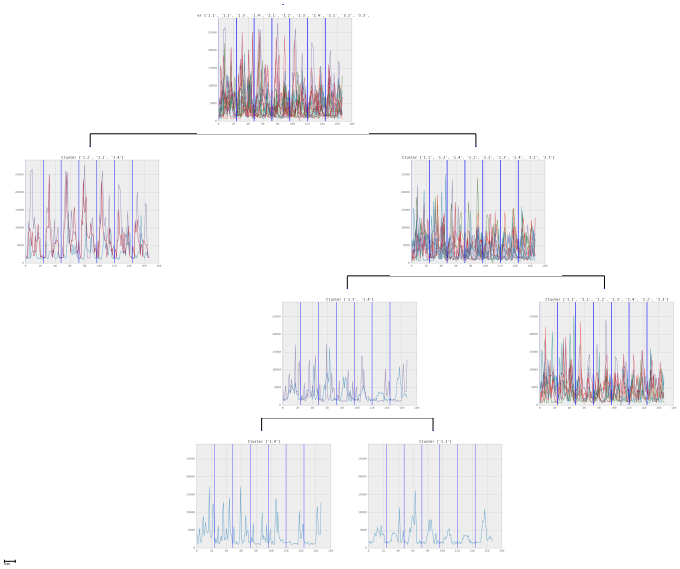


Fig. 4. Resulted clustering of three households using data from four different times. Note that we use $X.1, X.2, X.3, X.4$ to indicate the time series of household number X.

can be useful information when recommending certain energy packages. In addition, our method is scalable and it does not require any parameter tuning by the user to create a model. This supports its practical usefulness.

VIII. Conclusion

We proposed an approach for clustering energy consumption of households based on Gaussian process regression. The clusters retrieved by our approach can be used to create spending profiles and give companies specific insights in the spending behaviour. In our first experiment we clustered 71 households showed that our algorithm has a linear time complexity in comparison to the quadratic time complexity of K-medoids and agglomerative clustering methods using dynamic time warping and discussed our results. In a second experiment we showed that clustering time series data from different weeks per household makes it possible to cluster households based on their steady spending habits during the year. Lastly we outlined the practical use of our resulted clustering in the decision process of a company operating in the energy sector.

Acknowledgments

The authors would like to thank Clara Verhelst (iLab, 3E), Marion Neumann, Roman Garnett, Luc De Raedt, Tom Tourwé for their feedback.

References

[1] M. A. Pimentel, D. A. Clifton, and L. Tarassenko, "Gaussian process clustering for the functional characterisation of vital-sign trajectories," in *Machine Learning for Signal Processing, 2013 IEEE International Workshop on*, 2013, pp. 1–6.

[2] H.-C. Kim and J. Lee, "Clustering based on Gaussian processes," *Neural computation*, vol. 19, no. 11, pp. 3088–3107, 2007.

[3] M. Kumar, N. R. Patel, and J. Woo, "Clustering seasonality patterns in the presence of errors," in *Proceedings of the Eighth ACM International Conference on Knowledge Discovery and Data Mining*, NY, USA, 2002, pp. 557–563.

[4] D. Duvenaud, "Automatic model construction with Gaussian processes," Ph.D. dissertation, Computational and biological learning laboratory, University of Cambridge, 2014.

[5] T. Warren Liao, "Clustering of time series data - a survey," *Pattern Recogn.*, vol. 38, no. 11, pp. 1857–1874, Nov. 2005.

[6] M. Espinoza, C. Joye, R. Belmans, and B. D. Moor, "Short-term load forecasting, profile identification, and customer segmentation: a methodology based on periodic time series," *Power Systems, IEEE Transactions on*, vol. 20, no. 3, pp. 1622–1630, 2005.

[7] S. Rani and G. Sikka, "Article: Recent techniques of clustering of time series data: A survey," *International Journal of Computer Applications*, vol. 52, no. 15, pp. 1–9, August 2012.

[8] A. Lavin and D. Klabjan, "Clustering time-series energy data from smart meters," *Energy Efficiency*, vol. 8, no. 4, pp. 681–689, 2014.

[9] C. S. Myers, "A comparative study of several dynamic time warping algorithms for speech recognition," Ph.D. dissertation, MIT, 1980.

[10] H. Izakian, W. Pedrycz, and I. Jamal, "Fuzzy clustering of time series data using dynamic time warping distance," *Engineering Applications of Artificial Intelligence*, vol. 39, pp. 235–244, 2015.

[11] V. Niennattrakul and C. A. Ratanamahatana, "On clustering multimedia time series data using k-means and dynamic time warping," in *2007 International Conference on Multimedia and Ubiquitous Engineering*, April 2007, pp. 733–738.

[12] J. Paparrizos and L. Gravano, "k-Shape: Efficient and Accurate Clustering of Time Series," in *Proceedings of the 2015 ACM SIGMOD*, NY, USA, 2015, pp. 1855–1870.

[13] L. Kaufman and P. J. Rousseeuw, *Finding groups in data : an introduction to cluster analysis*, ser. Wiley series in probability and mathematical statistics. New York: Wiley, 1990.

[14] C. E. Rasmussen and C. K. I. Williams, "Gaussian processes for machine learning," 2006.

[15] "FLEXIPAC: Valorisation de la flexibilité des pompes à chaleurs," Research project financed by the Walloon Region, Belgium, 2013-2015. [Online]. Available: http://www.flexipac.ulg.ac.be

[16] M. Neumann, S. Huang, D. E. Marthaler, and K. Kersting, "pyGPs–a python library for Gaussian process regression and classification," *Journal of Machine Learning Research*, vol. 16, pp. 2611–2616, 2015.

[17] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001. [Online]. Available: http://www.scipy.org/

[18] J. Huerta-Cepas, F. Serra, and P. Bork, "Ete 3: Reconstruction, analysis, and visualization of phylogenomic data," *Molecular biology and evolution*, p. msw046, 2016.

[19] D. R. Heckendorn. (2012) Newick tree formats. [Online]. Available: http://marvin.cs.uidaho.edu/Teaching/CS515/newickFormat.html

[20] G. W. Claude Sammut, *Encyclopedia of Machine Learning*. Springer-Verlag,N.Y., 2007.

[21] Q. Zhang and I. Couloigner, *Computational science and its applications – International Conference, Singapore, May, Part III*. Springer Berlin Heidelberg, 2005, pp. 181–189.

[22] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1.

[23] B. Everitt, S. Landau, and M. Leese, "Cluster analysis." *Arnold, London*, 2001.

APPENDIX A
CLUSTER RESULT

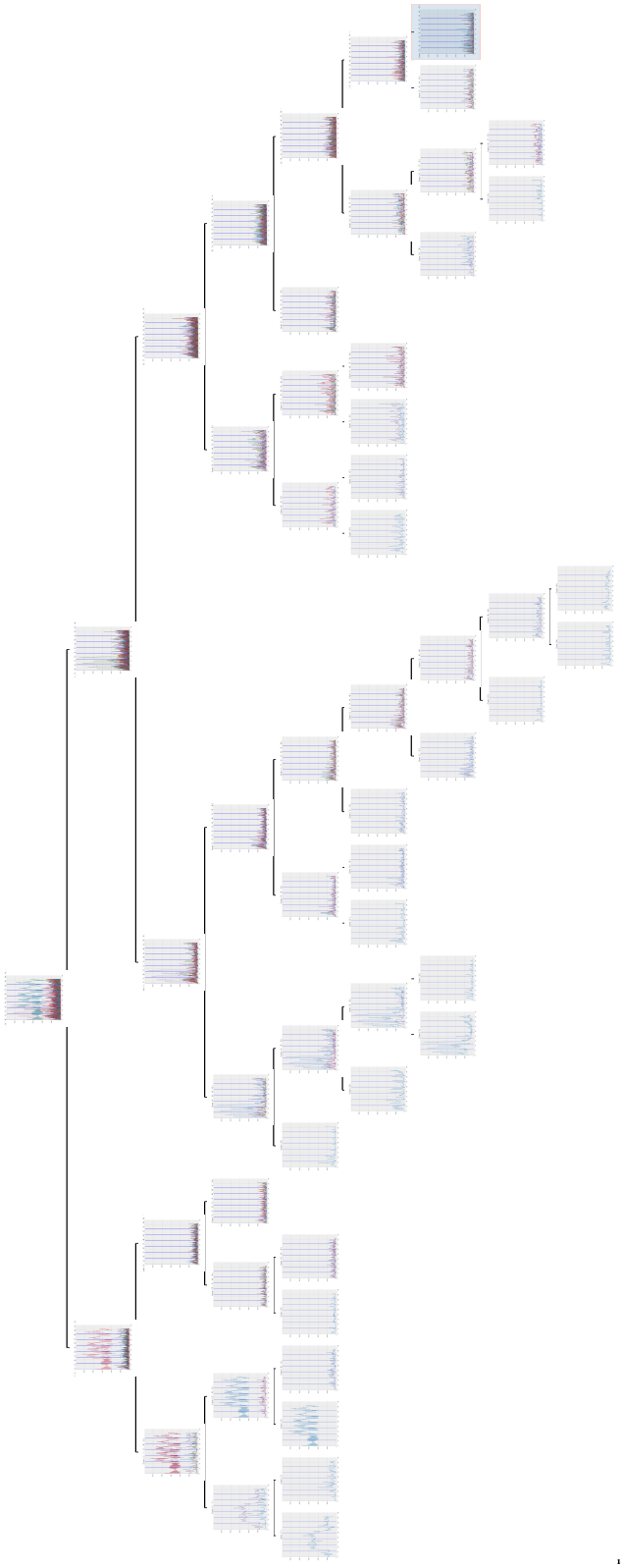


Fig. 5. Resulted clustering of our Gaussian process regression based method.

Bijlage D

Poster

Energieverbruik voorspellen en clusteren met Gaussiaanse processen

Christiaan Leysen

Promotoren: Prof. Dr. Luc De Raedt, Dr. Tom Tourwé
Begeleiders: Dr. Wannes Meert, Dr. Mathias Verbeke, Pierre Dagnely

Introductie

Probleemstelling: Hoe kunnen we energieconsumptie voorspellen en clusteren gebaseerd op historische data?

Input: Historische energieconsumptie en meteorologische data
Middel: Gaussiaanse processen regressie (GPR)
Output: **1)** 2 daagse Predictie en van energieverbruik per uur
2) Clustering van weekprofielen gebaseerd op GPR

Een Gaussiaans proces (GP) is een collectie van willekeurige variabelen, waarbij elke eindige lineaire combinatie van deze willekeurige variabelen een multivariate normale verdeling heeft:

$$f(x) \sim GP(m(x), k(x,x'))$$

- Kernelfunctie: correlatie tussen alle trainingsdata bepalen.

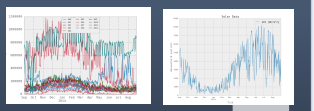
$$k(x,x') = \sigma_f^2 \exp\left[\frac{(x-x')^2}{2l^2}\right] + \sigma_n^2 \delta(x,x')$$

- Hyperparameters kernel -> maximum likelihood schatting

Predictie

Exploratie

- consumptie
- meteo



Preprocessing

- Cleaning
- Resampling
- Feature engineering

$Y = predict(X_1, X_2, X_3, X_4, X_5, X_6, X_7)$

Y: Energy consumption
X_1: consumption lagged
x_1: Temperature
x_2: Solar data
X_3: WeekOfTheYear
X_4: DayOfTheWeek
X_5: HourOfDay
X_6: DayOthTheYear

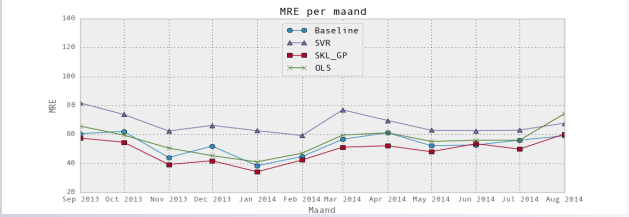
Predictie

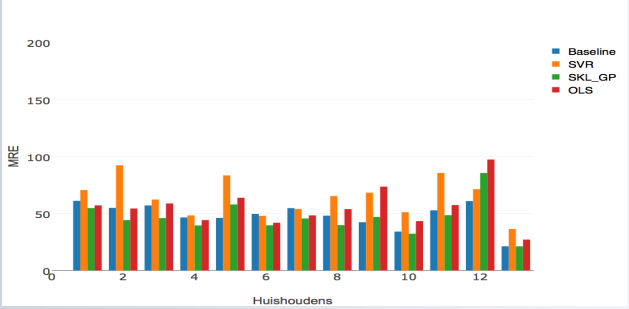
- Modaal huishouden
- Grillig huishouden

Predictie resultaten

MRE voor 71 Huishoudens (HH):

	pyGPs(lin+SE)	pyGPs (lin)	SKL_GP(lin)	SVR (lin)	Baseline -7D	OLS
HeatPump	84,90	84,77	77,34	76,19	76,00	93,21
Other	56,39	57,22	54,80	63,12	58,60	62,94
Ot+HP	51,13	51,46	48,73	67,41	54,07	55,97





Clustering

- Predictie geeft info over individuele huishoudens
- Extra info dataset → Relatieve temporele info HHs

GPR gebaseerd tijdreeksclusteren

+ Auto parameter tuning, schaalbaar,
+ Geen feature engineering nodig
- Leert model voor maar één functie (tijdreeks)

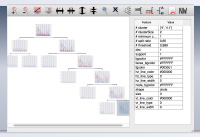
GPR Generalized: (Model leren door hyperparameter optimalisatie over meerdere tijdreeksen)

1) For every Sample (X_n,Y_n)
   likelihood = getPosterior(X_n,Y_n)
   addToTotal(likelihood)
   Return total_likelihood
2) Find Hyperparams for which total_likelihood is maximum

Clustering (S_{min}= min. clusterSize, t_{sim}=similarity threshold,$mean_{sim}$= mean similarity):

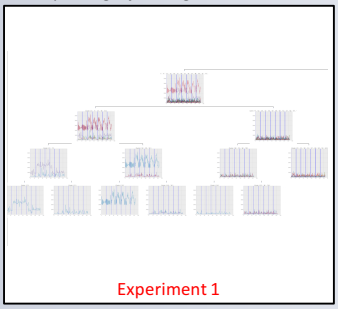
1) Calculate General model
2) Calculate RMSE between generalised prediction and HH observations
3) If (cluster size > S_{min} & $mean_{sim} < t_{sim}$): Divide cluster using split ratio
4) Cluster again recursively

Interactieve visualisatie:



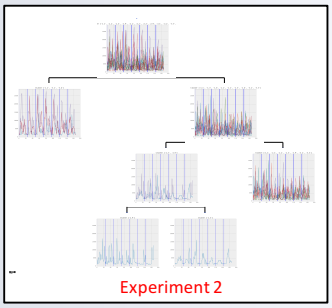
Clustering resultaten

Groeperen gelijkaardige huishoudens:



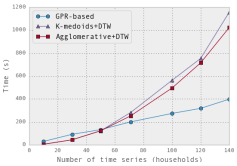
Experiment 1

Consistentie doorheen het jaar:



Experiment 2

Complexiteit:



Praktisch gebruik:

- Experiment 1:
  - -Energiepakketten aanbevelen
  - Vergelijk huishoudens met gelijkaardige gebruikspatronen
  - Detecteer inefficiënties
- Experiment 2:
  - Stabiele huishoudens detectie →Geschikte kandidaten voor nauwkeurige predictie

Contactinformatie: christiaan.leysen@student.kuleuven.be

Bibliografie

[1] O. D. Somer and T. Kutz, "Machine learning techniques for forecasting of building energy consumption," Ph.D. dissertation, KU Leuven, 2015.

[2] R. W. T. Borg, "Electricity load modelling using computational intelligence." Ph.D. dissertation, TU Delft, 2005.

[3] K. S. Weranga K. S. K and C. D. P., *Smart Metering Design and Applications*, 2014. [Online]. Available: http://link.springer.com/10.1007/978-981-4451-82-6

[4] K. Li and H. Su, "Forecasting building energy consumption with hybrid genetic algorithm-hierarchical adaptive network-based fuzzy inference system," *Energy and Buildings*, vol. 42, no. 11, pp. 2070 – 2076, 2010.

[5] R. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice:*. OTexts, 2014.

[6] H. Drucker, C. J. Burges, L. Kaufman, C. J. C, B. L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," 1996.

[7] (2016) Auto-regressief model. [Online]. Available: https://en.wikipedia.org/wiki/Autoregressive_model

[8] C. E. Rasmussen and C. K. I. Williams, "Gaussian processes for machine learning," 2006.

[9] M. Samarasinghe and W. Al-Hawani, "Short-term forecasting of electricity consumption using Gaussian processes," Master's thesis, 2012.

[10] D. J. C. MacKay, *Information Theory, Inference & Learning Algorithms*. New York, NY, USA: Cambridge University Press, 2002.

[11] N. P. Hadi Asheri, Hamid Reza Rabiee and M. H. Rohban, "A gaussian process regression framework for spatial error concealment with adaptive kernels," in *The 20th International Conference on Pattern Recognition*. Istanbul, Turkey: IEEE, 2010.

[12] C. S. Myers, "A comparative study of several dynamic time warping algorithms for speech recognition," Ph.D. dissertation, MIT, 1980.

[13] M. Müller, *Information retrieval for music and motion*. Springer, 2007, vol. 2.

[14] Q. Chen, G. Hu, F. Gu, and P. Xiang, "Learning optimal warping window size of dtw for time series classification," in *Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on*, 2012, pp. 1272–1277.

[15] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[16] L. Kaufman and P. J. Rousseeuw, *Finding groups in data : an introduction to cluster analysis*, ser. Wiley series in probability and mathematical statistics. New York: Wiley, 1990.

[17] G. W. Claude Sammut, *Encyclopedia of Machine Learning*. Springer-Verlag,N.Y., 2007.

[18] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1.

[19] B. Everitt, S. Landau, and M. Leese, "Cluster analysis." *Arnold, London*, 2001.

[20] S. K. Arunesh Kumar Singh, Ibraheem and M. Muazzam, "An overview of electricity demand forecasting techniques," in *Network and Complex Systems*, vol. 3, Nov 2013, pp. 38–48.

[21] C. W. Bin Yan and W. Xie, "Prediction of buildings energy consumption," *Neural Netw.*, vol. 4, no. 2, Mar. 2013. [Online]. Available: http://cs109-energy.github.io

[22] K. Kandananond, "Forecasting electricity demand in thailand with an artificial neural network approach," *Energies*, vol. 4, no. 8, pp. 1246–1257, 2011.

[23] W.-C. Hong, "Electric load forecasting by support vector model," *Applied Mathematical Modelling*, vol. 33, no. 5, pp. 2444 – 2454, 2009.

[24] J. Zeng and W. Qiao, "Short-term solar power prediction using an rbf neural network," in *2011 IEEE Power and Energy Society General Meeting*, July 2011, pp. 1–8.

[25] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, USA: Springer-Verlag, Inc., 1995.

[26] K. Kandananond, "Forecasting electricity demand in thailand with an artificial neural network approach," *Energies*, vol. 4, no. 8, p. 1246, 2011.

[27] P.-F. Pai and W.-C. Hong, "Forecasting regional electricity load based on recurrent support vector machines with genetic algorithms," *Electric Power Systems Research*, vol. 74, no. 3, pp. 417 – 425, 2005.

[28] Y. Yan, P. Guo, and L. Liu, "A novel hybridization of artificial neural networks and arima models for forecasting resource consumption in an iis web server," in *Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on*, Nov 2014, pp. 437–442.

[29] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[30] G. E. Box and G. M. Jenkins, "Time series analysis forecasting and control," San Francisco, 1970. [Online]. Available: http://opac.inria.fr/record=b1108766

[31] N. K. Pasapitch Chujai and K. Kerdprasop, "Time series analysis of household electric consumption with arima and arma models," *Lecture Notes in Engineering and Computer Science*, 2013.

[32] M. Blum and M. Riedmiller, "Electricity demand forecasting using gaussian processes," in *The AAAI-13 Workshop on Trading Agent Design and Analysis*, 2013.

[33] H. Y. Noh and R. Rajagopal, "Data-driven forecasting algorithms for building energy consumption," in *Conference on Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, San Diego, CA, USA, 2013.

[34] H. Mori and M. Ohmi, "Probabilistic short-term load forecasting with gaussian processes," in *Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems, IEEE*, Nov 2005.

[35] T. Koskela, M. Lehtokangas, J. Saarinen, and K. Kaski, "Time series prediction with multilayer perceptron, fir and elman neural networks," in *In Proceedings of the World Congress on Neural Networks*. Press, 1996, pp. 491–496.

[36] t. . A. Karin Kandananond, 2012.

[37] M. A. Pimentel, D. A. Clifton, and L. Tarassenko, "Gaussian process clustering for the functional characterisation of vital-sign trajectories," in *Machine Learning for Signal Processing, 2013 IEEE International Workshop on*, 2013, pp. 1–6.

[38] H.-C. Kim and J. Lee, "Clustering based on Gaussian processes," *Neural computation*, vol. 19, no. 11, pp. 3088–3107, 2007.

[39] M. Kumar, N. R. Patel, and J. Woo, "Clustering seasonality patterns in the presence of errors," in *Proceedings of the Eighth ACM International Conference on Knowledge Discovery and Data Mining*, NY, USA, 2002, pp. 557–563.

[40] D. Duvenaud, "Automatic model construction with Gaussian processes," Ph.D. dissertation, Computational and biological learning laboratory, University of Cambridge, 2014.

[41] T. Warren Liao, "Clustering of time series data - a survey," *Pattern Recogn.*, vol. 38, no. 11, pp. 1857–1874, Nov. 2005.

[42] M. Espinoza, C. Joye, R. Belmans, and B. D. Moor, "Short-term load forecasting, profile identification, and customer segmentation: a methodology based on periodic time series," *Power Systems, IEEE Transactions on*, vol. 20, no. 3, pp. 1622–1630, 2005.

[43] S. Rani and G. Sikka, "Article: Recent techniques of clustering of time series data: A survey," *International Journal of Computer Applications*, vol. 52, no. 15, pp. 1–9, August 2012.

[44] R. Bellman, "Adaptive control processes: A guided tour. (A RAND Corporation Research Study)." Princeton, N. J.: Princeton University Press, XVI, 255 p. (1961)., 1961.

[45] V. Niennattrakul and C. A. Ratanamahatana, "On clustering multimedia time series data using k-means and dynamic time warping," in *2007 International Conference on Multimedia and Ubiquitous Engineering*, April 2007, pp. 733–738.

[46] J. Paparrizos and L. Gravano, "k-Shape: Efficient and Accurate Clustering of Time Series," in *Proceedings of the 2015 ACM SIGMOD*, NY, USA, 2015, pp. 1855–1870.

[47] C.-K. Chu and J. S. Marron, "Comparison of two bandwidth selectors with dependent errors," *The Annals of Statistics*, vol. 19, no. 4, pp. pp. 1906–1918, 1991.

[48] P. Burman, E. Chow, and D. Nolan, "A cross-validatory method for dependent data," *Biometrika*, vol. 81, no. 2, pp. pp. 351–358, 1994.

[49] "FLEXIPAC: Valorisation de la flexibilité des pompes à chaleurs," Research project financed by the Walloon Region, Belgium, 2013-2015. [Online]. Available: http://www.flexipac.ulg.ac.be

[50] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[51] G. C. Cawley and N. L. C. Talbot, "Preventing over-fitting during model selection via bayesian regularisation of the hyper-parameters," *J. Mach. Learn. Res.*, vol. 8, pp. 841–861, 2007.

[52] G. C. Cawley and N. L. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *J. Mach. Learn. Res.*, vol. 11, pp. 2079–2107, 2010.

[53] M. Neumann, S. Huang, D. E. Marthaler, and K. Kersting, "pyGPs–a python library for Gaussian process regression and classification," *Journal of Machine Learning Research*, vol. 16, pp. 2611–2616, 2015.

[54] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001. [Online]. Available: http://www.scipy.org/

[55] J. Huerta-Cepas, F. Serra, and P. Bork, "Ete 3: Reconstruction, analysis, and visualization of phylogenomic data," *Molecular biology and evolution*, p. msw046, 2016.

[56] D. R. Heckendorn. (2012) Newick tree formats. [Online]. Available: http://marvin.cs.uidaho.edu/Teaching/CS515/newickFormat.html

[57] C. Leysen. Implementatie: Energieverbruik voorspellen en clusteren met gaussiaanse processen. [Online]. Available: https://www.dropbox.com/sh/mn0bfn9mjsp24au/AAC_D9s8SvJkZooNrq3Mr8Iga?dl=0

Fiche masterproef

Student: Christiaan Leysen

Titel: Energieverbruik voorspellen en clusteren met Gaussiaanse processen

Engelse titel: Energy consumption prediction and clustering using Gaussian processes

UDC: 621.3

Korte inhoud:
Energiebedrijven hebben een goed zicht nodig op de consumptie van elektrische energie en doen hiervoor vaak beroep op voorspellings- en/of clustermethoden. In deze context stelt dit werk een voorspellings- en clustermethode voor, die gebaseerd zijn op Gaussiaanse processen.
Deze thesis is opgedeeld in een voorspellings- en een clustergedeelte. In het voorspellingsgedeelte bespreken we hoe we de ruwe data verwerken tot input voor de Gaussiaanse proces regressie en focussen we ons op een voorspelling voor de volgende twee dagen per uur.
Het clustergedeelte van de thesis stelt een nieuwe clustermethode voor, die gebaseerd is op Gaussiaanse proces regressie (GPRC). Deze methode passen we toe op het consumptiegedrag van huishoudens door hun weekprofielen (tijdreeksen) te beschouwen. Om deze te clusteren zal de methode gebruik maken van een algemeen model dat geleerd wordt op een set van tijdreeksen, gebaseerd op hun waarschijnlijkheid. Het voordeel van de voorgestelde techniek is dat ze geen paarsgewijze vergelijking van de tijdreeksen nodig heeft, in tegenstelling tot vele andere clustermethoden voor tijdreeksen.
Evaluatie gebeurt a.d.h.v. een *real-life* dataset van 71 huishoudens. De voorspellingsmethode wordt geëvalueerd en vergeleken met lineaire regressie, *support vector regressie* en een baseline methode die de waarde van een week geleden teruggeeft als voorspelling. De clustermethode wordt vergeleken met *k-medoids* met *dynamic time warping* en hiërarchisch agglomeratief clusteren met *dynamic time warping*. Er wordt enerzijds aangetoond dat GPRC een betere schaalbaarheid heeft en anderzijds dat de resultaten ervan nuttig zijn in het beslissingsproces van een bedrijf uit de energiesector.

Thesis voorgedragen tot het behalen van de graad van Master of Science in de ingenieurswetenschappen: computerwetenschappen, hoofdspecialisatie Artificiële intelligentie

Promotoren: Prof. dr. Luc De Raedt
Dr. Tom Tourwé

Assessoren: Dr. Raoul Strackx
Dr. ir. Wannes Meert

Begeleiders: Dr. ir. Wannes Meert
Dr. Mathias Verbeke
Pierre Dagnely