Machine Learning Datakwaliteit NDW

Afstudeerverslag

Özcan Seker
v3.0 Definitieve versie
19-10-2021

1 Inhoud

2 Versiebeheer

Versie	**Datum**	Aanpassingen
0.1	15-3-2021	Initiële opzet van document: - Inleiding - Doel - Requirements - Onderzoek - Eerste resultaten onderzoek
0.2	05-04-2021	- Vooronderzoek - Resultaten toegevoegd - Adviesrapport document aangemaakt toegevoegd - Scrum documenten aangemaakt en toegevoegd
1.0 Definitieve versie	01-06-2021	- Dashboard resultaten - Conclusie discussie - Adviesrapport - Evaluatie requirements
2.0 Herkansing versie	9-6-2021	Verslag opnieuw gestructureerd en herschreven - Korter methoden uitgelegd - Termen lijst toegevoegd - Bewijs van competenties toegevoegd
2.1 Herkansing versie	27-8-2021	Verwerken feedback bedrijfsbegeleider en afstudeerbegeleider.
3.0	19-10-2021	Herstructurering van het verslag om er een samenhangend verslag van te maken. Verslag geherstructureerd met behulp van commentaar en hulp van Sander ten Hoor.

3 Betrokkenen

Naam	**Functie**	**Rol**
Özcan Seker	Student HBO-ICT	Afstudeerder
Sander ten Hoor	Software Engineer Quintor	Bedrijfsbegeleider
Leo de Vries	Manager Quintor	Peoplemanager
Dick Heijink	Docent Saxion	Afstudeerbegeleider
Arjen Wassink	Manager Quintor	Opdrachtgever
Willem van Loon	Data analist datakwaliteit NDW	Opdrachtgever / Contactpersoon NDW

4 Begrippenlijst

4.1 Afkortingen

NDW	Nationaal Dataportaal Wegverkeer
POC	Proof of concept
ML	Machine Learning
STL	Seasonal-Trend decomposition using LOESS
ETS	Exponential smoothing
ARIMA	Autoregressive integrated moving average
IQR	Interquartile Range (Interkwartielafstand)
MAD	Median absolute deviation
GESD	Generalized Extreme Studentized Deviate Test for Outliers

4.2 Begrippen

(ML) Model	Een ML model. Kan aan de hand van inputs, een output geven. Het model kan getraind worden om voor een set aan inputs, de gewenste output te geven.
Methode	In de context van dit verslag wordt de term ‘methode’ gebruikt om te verwijzen naar een manier om foutieve datapunten te vinden.
Gelabelde data	Data waarbij de gewenste output voor een set aan inputs is aangegeven. *Voorbeeld: Verkeersdata waar foutieve datapunten zijn gemarkeerd als fout.*
Datapunt	In de context van dit verslag een verkeersmeting. Om de zoveel tijd wordt de intensiteit en snelheid op de weg gemeten. Elke meting is een apart datapunt.
Meetpunt/Meetlocatie	Locatie waar verkeersdata wordt gemeten. Elke locatie heeft een eigen id in de data.
Verkeersintensiteit	Het aantal auto’s die op dat moment over de weg rijden.
Tijdreeks	Dataset waarbij dezelfde variabelen op verschillende tijdstippen zijn gemeten.

Trend	De algemene daling of stijging van een tijdreeks die over de tijd plaatsvindt.
Seizoenaliteit	Het terugkerend patroon in een tijdreeks over een vast periode. *Voorbeeld: is het terugkerend spitsuur in verkeersdata. De spits zal elke dag (vaste periode) terugkomen op hetzelfde tijdstip (terugkerend patroon).*
Cyclus	De cyclus in een tijdreeks is ook een terugkerend patroon, alleen vindt de cyclus niet plaats over een vaste periode.
Overige waarden / remainder	De delen van een waarde die niet door de trend en seizoenaliteit uitgelegd kunnen worden.
Error	De fout die het model maakt. De error is de voorspelde waarden min de gemeten waarden.
Signaal	De output data van een meetlocatie, de verkeersgegevens.
Technisch foute data	Een fout in de data welke ontstaat omdat een meetpunt defect functioneert.
Verkeerskundige situatie	Een ongewone situatie in de data, wat niet fout is. Een file is een verkeerssituatie.
Correlatie	Samenhang tussen twee variabelen. Wanneer de correlatie hoog is dan zullen de twee variabelen sterk afhankelijk zijn van elkaar. *Voorbeeld: Het gewicht schatten van een mens met behulp van zijn lengte.*
Autocorrelatie	Correlatie van de data met zijn eigen verleden waarden. Als de correlatie hoog is dan zal de huidige waarde sterk afhankelijk zijn met de vorige waarde.*Voorbeeld: Het gewicht schatten van een persoon met behulp van zijn gewicht gisteren.*
Model parameters	Aanpasbare delen van het model waardoor deze anders werkt. Er zijn twee soorten parameters: - De parameters die door een gebruiker aangepast kunnen worden. - De parameters die door het trainingsalgoritme aangepast worden. Het trainingsalgoritme zal de parameters zo aanpassen dat aan de hand van de gegeven inputs de gewenste output zo goed mogelijk wordt teruggegeven.

5 Samenvatting

Het doel van deze opdracht was het vinden van een Machine Learning oplossing om verkeersdata te valideren. De Machine Learning oplossing moest in staat zijn om:

- Technisch foute data in verkeersdata te markeren.
- Aan te geven waarom de fouten gemarkeerd zijn als fout.
- Fouten vinden die niet door een mens gevonden kunnen worden.
- Leren van de nieuw gevonden fouten zodat deze ook gevonden kunnen worden in de toekomst.

Het bewijs dat de gevonden ML oplossing werkt, moest komen in de vorm van een proof of concept. Voordat er aan de POC is begonnen, is er onderzoek gedaan. Het onderzoek bevatte literatuuronderzoek om ML methode te vinden om verkeersdata te valideren. Hierna kwam er dataonderzoek. Als laatst werden er prototypes gemaakt van de gevonden ML methoden, om te bewijzen dat deze methoden ook echt werken. Met de informatie uit het onderzoek moest er een POC ontworpen en gebouwd worden.

Er zijn uiteindelijk twee methoden gevonden in de literatuur:

- STL en toets
- Voorspellend model en afstand

Van deze methoden zijn ook prototypes gebouwd. Het bleek lastiger dan verwacht om te bewijzen dat deze methoden ook technisch foute data kunnen detecteren. Tijdens de prototyping waren de methodes maar gedeeltelijk geïmplementeerd en getest. Er is ook verkeers domeinkennis nodig om te bepalen of een gedetecteerde fout ook daadwerkelijk een datafout is en niet een verkeerskundige situatie(file, minder verkeer feestdag). De afstudeer kon niet zonder verkeerskennis bepalen of de methoden ook echt technische fouten detecteren.

Er is gekozen om het project te focussen op het bewijzen dat de ML methoden correct werken. De uiteindelijke POC was daarom ook een applicatie die helpt met het bewijzen dat de gevonden ML oplossingen werken. Er is gekozen om een dashboard als POC te implementeren.

Een dashboard maakt het ook makkelijk om resultaten te creëren. Hierdoor kan een domeinexpert zelf resultaten creëren met verschillende opties zonder te programmeren. Een dashboard is ook visueel waardoor het duidelijk aantoont hoe de ML methoden werken. Hiernaast kost het maken van een dashboard ook ontwikkel en programmeerwerk, wat weer aansluit bij de HBO-i competenties.

Uiteindelijk is de conclusie van het onderzoek dat de methode voorspellend model en afstand potentie heeft om technisch foute data te detecteren in verkeersdata.

De onderwerpen in dit verslag zijn lastig als de lezer niet bekend is in het veld van ML en tijdreeksanalyses. Het bevat veel concepten die niet standaard terugkomen in het HBO-ICT curriculum. Tegelijkertijd is er een limiet aan hoe groot het verslag mag zijn. De balans tussen een uitgebreide uitleg en ruimtegebrek is een uitdaging geweest.

6 Inleiding

Dit is het afstudeerverslag voor de opdracht Machine Learning Datakwaliteit NDW uitgevoerd door Özcan Seker. De inhoud van verschillende documenten worden in dit verslag samengevat. De originele documenten zijn meegeleverd als bijlage in een aparte bestanden.

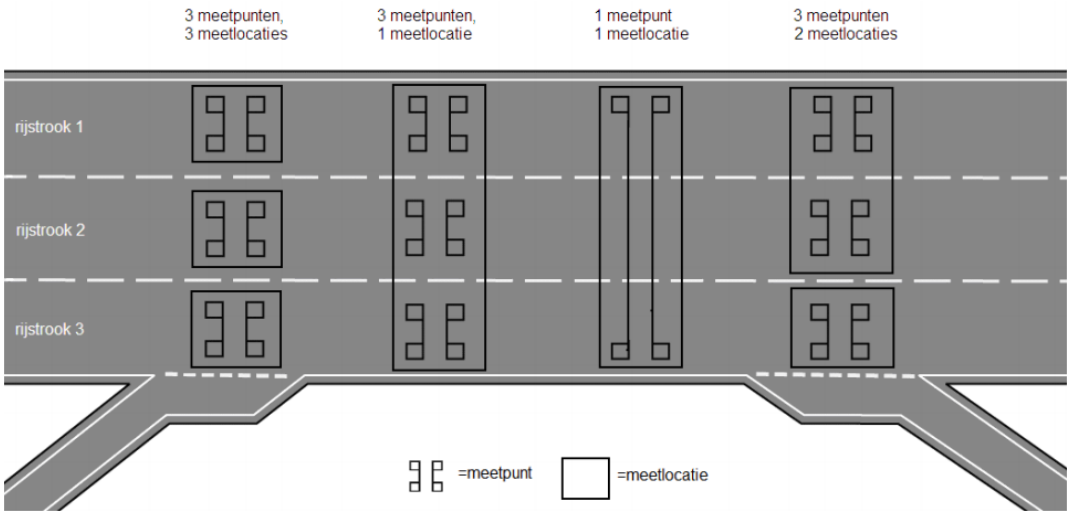
De opdracht wordt uitgevoerd bij Quintor B.V. Quintor wordt ingeschakeld door klanten om projectmatig software te ontwikkelen. Dit doen ze voor top 250 organisaties en overheden. Werknemers van Quintor worden als teams ingezet bij verschillende klanten. Quintor bestaat uit 5 vestigingen met totaal 178 werknemers (Quintor, 2021).

Nationaal Dataportaal Wegverkeer(NDW) is één van de opdrachtgevers van Quintor. In NDW werken Nederlandse overheden samen aan het inwinnen, combineren, opslaan en distribueren van mobiliteitsdata (NDW, 2021). De opdracht beschreven en uitgevoerd in dit verslag is afkomstig van NDW. NDW heeft deze opdracht in het verleden bedacht. Deze opdracht is al een keer uitgevoerd door het bedrijf Vortech.

De opdracht van NDW werd voorgesteld tijdens de sollicitatieprocedure. Deze opdracht is verbonden aan een bedrijfscase, wat niet geldt voor all Quintor opdrachten. Hierom is samen met de afstudeerder en Quintor besloten om deze opdracht uit te voeren. Hierna is NDW erbij betrokken om de opdracht nog een keer uit te voeren als afstudeeropdracht. NDW deed mee omdat het altijd op zoek is naar nieuwe methoden om de kwaliteit te waarborgen.

6.1 Aanleiding opdracht

NDW verzamelt verkeersdata die gekocht wordt van leveranciers. De verkeersdata is afkomstig van meetlussen (Zie afbeelding 1).



Afbeelding 1: Meetlocaties/meetpunten op de weg (NDW, 2010, p. 17)

NDW stelt kwaliteitseisen aan deze gegevens, die de leveranciers zelf moeten controleren en rapporteren. De fouten in de data moeten aangegeven zijn door de leverancier. Deze

eisen blijken lastig te handhaven zonder eigen analyses en handvatten op de data. Met de analyses kunnen de leveranciers ingelicht worden, wanneer ze niet voldoen aan de eisen. Daarnaast willen gebruikers van de data alleen juiste gegevens gebruiken.

Er zijn validatieregels ingezet om de data te valideren. Deze validatieregels komen voort uit veel voorkomende verstoringen. Een aantal voorbeelden van invalide data zijn:

- De data bestaat alleen uit nullen
- De data bestaat uit slechts een zeer beperkt aantal waardes in een uur
- De indeling in voertuigcategorieën bevat te weinig personenauto's of te veel motoren

De ingezette validatieregels kunnen niet alle fouten opvangen. Er zijn ook complexere fouten die voortkomen uit interacties tussen verschillende variabelen (NDW, 2019, p. 3 t/m 6). NDW is niet in staat om complexere validatieregels te schrijven. Deze kosten relatief veel geld/tijd om te implementeren en detecteren alleen fouten die door een beheerder gevonden zijn. De validatieregels worden ook snel te complex vanwege uitzonderingen. (Vries & NDW, 2019, p. 2).

NDW zoekt naar een mogelijkheid om onjuiste gegevens te scheiden van juiste gegevens, zonder dat op voorhand in detail bekend is wat goede en wat foute gegevens zijn. NDW wil onderzoeken of er met behulp van Machine Learning (ML) juiste gegevens te onderscheiden zijn van onjuiste gegevens (NDW, 2010, p. 3 t/m 6).

6.2 Probleemstelling

NDW wil de data van leveranciers valideren. NDW is op dit moment niet in staat om complexe datavalidatie toe te passen zonder veel geld en tijd te investeren. Er moet een oplossing komen dat in staat is om fouten te detecteren zonder dat op voorhand in detail bekend is wat goede en wat foute gegevens zijn.

6.3 Opdracht

De data is afkomstig van verschillende meetlocaties en is tijdsgebonden data. Er wordt elke minuut data verzameld(NDW, 2010, p. 3 t/m 6).

NDW zoekt naar technisch onjuiste data oftewel slechte datakwaliteit. Er zijn ook verkeerskundig afwijkende situaties. Zoals het boerenprotest waar wel verkeerskundig afwijkende data te zien is, maar die technisch wel van goede kwaliteit kan zijn(NDW, 2010, p. 3 t/m 6).

NDW wil dat er op twee tijdsniveaus fouten worden aangegeven, op minuut en etmaalniveau. Op etmaalniveau kan worden aangegeven of een meetpunt defect is. Op minuut niveau kunnen individuele datapunten gemarkeerd worden als incorrect. De eerste prioriteit van NDW is om op etmaalniveau defecte meetlocaties te vinden. Indien het mogelijk is, wil NDW ook de foutieve datapunten markeren op minuutniveau(NDW, 2010, p. 3 t/m 6).

NDW kan de gevonden afwijkingen melden bij de leverancier. De onjuiste gegevens kunnen ook gemarkeerd worden zodat gebruikers de data kunnen filteren(NDW, 2010, p. 3 t/m 6).

Er worden eisen gesteld aan een oplossing met ML. De nadruk wordt gelegd op zekerheid. De ML methode moet onjuiste gevallen markeren wanneer het zeker is. De onzekere gevallen kunnen worden voorgelegd aan een beheerder. Ook wil het dat het model leert van de nieuwgevonden fouten. Hierdoor kan de methode in de toekomst beter classificeren en worden het aantal onzekere gevallen verminderd. De ML methode moet ook aangeven waarom een datapunt fout is geclassificeerd(NDW, 2010, p. 3 t/m 6).

De validatieregels zijn efficiënt. De nieuwe methoden moet niet dezelfde fouten terug kunnen vinden want de validatieregels kunnen dit al snel doen. Het is de bedoeling dat het nieuwe fouten kan vinden. Het hoeft niet per se de validatieregels te vervangen. Het kan ook als een extra validatieregel worden ingezet(19-3-2021 Willem van Loon, Arjen Wassink).

NDW is ook onderzoek aan het doen naar MLOps(A. Wassink, persoonlijke communicatie, 08-03-2021). Dit is DevOps voor Machine Learning oplossingen. Met MLOps kan een opzichzelfstaande ML methode omgezet worden naar een ML systeem. Dit systeem is in staat om het model automatisch te trainen, testen en in productie te brengen(Google, 2021). NDW wil ook dat dit wordt meegenomen in het onderzoek(A. Wassink, persoonlijke communicatie, 08-03-2021).

NDW is nog niet op zoek naar een eindoplossing maar een basis waarop zij zelf kunnen voortborduren. Ze zijn op zoek naar een proof of concept (POC). Dit wordt een onderzoekend project. Dit project zal aangeven of Machine Learning toegevoegde waarde biedt aan de huidige validatie methode(NDW, 2019, p. 3 t/m 6).

Er is al een poging gewaagd om dit probleem op te lossen met ML. Deze poging is gedaan door het bedrijf VORtech. Ze hebben drie modellen getraind met de data van de NDW: twee cluster modellen en een random forest model. Een random forest model is een soort van beslissingsboom. De cluster modellen proberen groepen in de data te identificeren (VORtech, 2020). De conclusie van Vortech was dat er geen toegevoegde waarde is aan het implementeren van deze methoden om data te valideren (Bijlage gesprek: 28-5-2021 Willem van Loon). De opdracht is nog een keer opgezet door Quintor als opdracht voor de afstudeerder. NDW doet mee omdat het altijd op zoek is naar nieuwe methoden om de kwaliteit te waarborgen en er zijn nog methodes die VORtech niet heeft geprobeerd (Bijlage: Gesprek Willem 2-9-2021).

7 Project

Op basis van de opdrachtomschrijving kan er een doel met subdoelen geformuleerd worden. Aan de hand van het doel en de beroepsproducten kan er een proces om een POC te bouwen opgezet worden.

7.1 Doel

Het doel van dit project is om te onderzoeken of er een Machine Learning methode bestaat die in staat is, om in de NDW data op etmaalniveau aan te geven of een meetpunt defect is. Het bewijs moet in de vorm van een proof of concept komen.

Met de volgende subdoelen:

- De POC moet fouten detecteren zonder dat op voorhand in detail bekend is wat goede en wat foute gegevens zijn
- De POC moet alleen technisch foute data detecteren.
- De POC moet onzekere gevallen markeren als onzeker.
- De POC moet aangeven waarom het punten als fout markeert.
- De POC moet leren van de door de beheerder gevonden fouten. Hierdoor kan het in de toekomst minder onzekere gevallen detecteert.
- Als het mogelijk is moet de POC naast etmaalniveau ook op minuutniveau foutieve datapunten kunnen markeren.
- De POC moet MLOps functionaliteit hebben, waarbij continuous integration, continuous deployment en continuous training is geimplementeerd.

7.2 Beroepsproducten

Om aan te tonen dat de stage succesvol is afgerond en de doelen bereikt zijn, worden de volgende beroepsproducten opgeleverd aan Quintor & het NDW:

- Dit afstudeerverslag wat beschrijft hoe het onderzoek is uitgevoerd.
- Adviesrapport voor verdere aanpak van dit probleem.
- Technische en Functioneel documentatie met software specificaties.
- Een proof of concept applicatie die voldoet aan de gestelde eisen.
- Vooronderzoek: Geeft meer achtergrondinformatie over de gevonden ML methoden en de experimenten uitgevoerd tijdens dit project.

Het Saxion verwacht de volgende opleveringen:

- Afstudeerdossier met de volgende inhoud:
    - Zelfreflectie document
    - Eindbeoordeling
    - Beoordeling bedrijfsbegeleider
    - Beroepsproducten opgeleverd bij Quintor

7.3 Proces

Het project zal bestaan uit verschillende fasen.

7.3.1 Voorbereiding

In deze fase wordt het plan van aanpak opgezet. Dit gebeurt in de eerste vijf weken van het project. Er is veel wachttijd in deze fase. Het plan van aanpak zal meerdere keren worden beoordeeld door de afstudeer en bedrijfsbegeleider. In deze fase wordt er zoveel mogelijk onderzoek gedaan tijdens het wachten.

Het is belangrijk dat de datatoegang zo snel mogelijk geregeld wordt. Hierdoor kan er in deze fase al gestart worden aan het dataonderzoek. Naast het dataonderzoek kan er ook gestart worden aan het literatuuronderzoek om ML methoden te vinden die foutieve datapunten kunnen detecteren.

Hier moet ook het eerste contact gelegd worden met opdrachtgever.

7.3.2 Onderzoeksfase

Hier wordt het onderzoek dat gestart is in de vorige fase voortgezet. Voordat er een POC gebouwd wordt, wordt er onderzocht welke ML methoden beschikbaar zijn om het gegeven probleem op te lossen. Naast het onderzoek doen naar de ML methodes moet ook de data onderzocht worden om de structuur van de data te bepalen.

Naast het literatuur en data onderzoek worden er ook prototypes gemaakt van de ML modellen. De prototypes zullen bepalen of deze methoden goed verkeersdata kunnen valideren, zodat deze ingezet kunnen worden in een POC.

Deze fase zal twee weken duren.

7.3.3 Requirementsfase

Hier worden de requirements voor de ML oplossing en de POC gedefinieerd.

Om een indicatie te krijgen van wat voor soort aspecten belangrijk kunnen zijn, wordt ISO 25010 gebruikt. Relevante sub kwaliteitsaspecten worden uit het ISO 25010 model geselecteerd. Aan de subaspecten worden requirements verbonden (ISO 25000, 2020). De requirements worden hierna geprioriteerd met de MoSCoW methode.

Dit zal één week duren.

7.3.4 Realisatiefase

In deze fase wordt de POC geïmplementeerd. Er wordt gewerkt met Scrum. De realisatiefase is zes weken. In zes weken kunnen drie sprints van twee weken gepland worden.

Elke iteratie van de Scrum methode wordt de POC verder ontworpen, geprogrammeerd en getest. Voor elke userstory wordt opgesteld hoe deze gedemonstreerd kan worden, oftewel de *how to demo*. Ook wordt er aangegeven hoe deze getest moeten worden, oftewel de *how to test*. Definition of Done is wanneer de userstories aan beide “how to”s voldoen.

7.3.5 Afrondingsfase

Deze fase duurt vier weken. Na de eerste twee weken van deze fase moet de conceptversie van dit verslag ingeleverd worden op Saxion.

In de eerste twee weken wordt het project afgerond en de hoofdvraag beantwoord. Hier worden de conclusie en discussie van het verslag geschreven. Hier wordt ook de POC geëvalueerd of deze voldoet aan de opgestelde requirements.

Als laatst wordt er een adviesrapport opgesteld waarin beschreven staat of het implementeren van een oplossing met ML om verkeersdata te valideren uitvoerbaar is. Er wordt ook een advies uitgebracht over het implementeren van MLOps.

Nadat de conceptversie is ingeleverd kunnen nog ontbrekende delen in het verslag en POC aangevuld worden.

7.4 Tooling

Er zijn verschillende tools gebruikt tijdens het afstuderen. Een aantal van deze tools worden gebruikt door Quintor:

- Project management:
    - Jira
- Communicatie:
    - Slack
    - Whatsapp
    - Jitsi
- Code version control
    - Gitlab

De rest van de tooling is gekozen door de afstudeerder zelf:

- Ontwerp:
    - UML met draw.io
    - Figma voor mockups
- Documentatie:
    - Google docs
- Ontwikkeltools :
    - R
    - Python
    - Dash als dashboarding framework
    - Pytest voor unittesten
    - Pylint voor codekwaliteit en code smells
    - Pycharm als Python IDE
    - RStudio als R IDE

8 Onderzoek

In dit hoofdstuk worden de opzet van het onderzoek, de onderzoeksvragen en antwoorden op deze onderzoeksvragen beschreven. Aan de hand van dit onderzoek kan er een POC gebouwd worden.

8.1 Onderzoeksvragen

Deze hoofdvraag is afgeleid van het doel. De hoofdvraag van dit onderzoek is:

- Welke ML methoden zijn in staat, om per etmaal in de NDW data aan te geven of een meetpunt juiste data verschaft?

Om deze hoofdvraag te beantwoorden zullen de volgende stappen worden ondernomen. Het zal beginnen met het onderzoek doen naar het soort data dat NDW heeft. Als bekend is hoe de data is opgebouwd, kunnen er ML methoden gevonden worden die werken op dit soort data. Ook is het handig om te onderzoeken wat voor soort fouten al gevonden kunnen worden door de NDW. Hierdoor is er ook een inzicht in wat NDW al kan.

Voor de POC moeten requirements opgezet worden. Aan de hand van de requirements kan er een POC ontworpen en gebouwd worden. Wanneer de POC gebouwd is kan er een conclusie voor het project worden getrokken. Ook wil de NDW een kijk nemen naar MLOps en dit wordt ook onderzocht.

Dit kunnen we omzetten naar de volgende deelvragen:

- Hoe is de data van NDW gestructureerd?
- Wat voor soort datafouten bevinden zich in de data?
- Welke eisen worden gesteld aan de ML methode vanuit NDW?
- Welke ML methoden zijn er die in staat zijn in de NDW data onjuiste gegevens te markeren?
- Kunnen de ML methoden foutieve datapunten detecteren?

8.2 Onderzoeksmethodiek

Het plan voor het onderzoeken was om te beginnen met dataonderzoek. In realiteit was er geen datatoegang voor de eerste vijf weken van het project. Er was wel sample data beschikbaar. De sample data is beperkt maar toont aan hoe de NDW data is gestructureerd. De deelvraag dat eerst onderzocht wordt, is *“Welke ML methoden zijn er die in staat zijn in de NDW data onjuiste gegevens te markeren?”*. Dit wordt onderzocht m.b.v literatuuronderzoek.

De sample data toont aan dat de NDW verkeersdata tijdreeksdata is. Het boek *Forecasting: Principles and Practice (3rd ed)* legt verschillende aspecten van tijdreeksen uit en hoe de lezer met tijdreeksen kan werken. Methodes uit dit boek worden gebruikt om de NDW data te plotten en te onderzoeken. De deelvragen *“Hoe is de data van NDW gestructureerd?”* en *“Wat voor soort onjuiste gegevens bevinden zich in de data?”* worden beantwoord. Het doel is om een aspect van de data te vinden, welke foutieve en correcte datapunten scheidt.

Met de informatie uit het literatuuronderzoek en de data worden er prototypes van de ML methode gemaakt om de deelvraag *“Kunnen de ML methode foutieve datapunten*

detecteren?" te beantwoorden. De prototypes moeten aantonen of de ML methode foutieve datapunten vinden en wat de implementatie details zijn van de ML methode.

Om de POC te bouwen en ontwerpen moeten requirements opgesteld worden. Hiermee wordt de deelvraag *"Welke eisen worden gesteld aan de ML methode vanuit NDW?"* beantwoord. Dit wordt gedaan door interviews te houden met de opdrachtgever. Ook beschrijft de opdrachtomschrijving eisen waar de ML methode aan moet voldoen.

Als laatste kan MLOps onderzocht worden. Dit komt niet terug in de deelvragen. Dit wordt of literatuuronderzoek of experimenteel. Als er tijd over is kunnen aspecten van MLOps in de POC geïmplementeerd worden. Anders wordt er in het adviesrapport aangegeven hoe een MLOps omgeving opgezet kan worden.

Met de informatie opgedaan in het onderzoek kan er een POC gebouwd worden die aantoont dat de ML methode wel of niet implementeerbaar zijn.

8.3 Deelvragen

In dit gedeelte worden de deelvragen beantwoord. Als laatst wordt de hoofdvraag van het onderzoek beantwoord. Met de resultaten van dit onderzoek wordt er een POC ontworpen en gebouwd. Een uitgebreidere versie van het literatuuronderzoek, dataonderzoek en prototypering is beschikbaar als *6.Vooronderzoek* in de bijlagen map.

8.3.1 Welke ML methoden zijn er die in staat zijn in de NDW data onjuiste gegevens te markeren?

Eerst moet er duidelijk zijn welke oplossingen er zijn om verkeersdata te valideren met Machine Learning. Dat wordt beantwoord met deze deelvraag.

8.3.1.1 Machine Learning

Het doel van dit onderzoek is om met Machine Learning(ML) technisch foute data te vinden. Een Machine Learning model kan aan de hand van input data, outputs creëren. Tijdens het trainen leert het ML model de gewenste outputs creëren.

Er zijn twee basis aanpakken om een model te trainen: supervised en unsupervised. Supervised learning gebruikt gelabelde data om een model te trainen. Gelabelde data is data waarbij het gewenste resultaat in de data is aangegeven. Omdat bekend is wat het ML model moet teruggeven als output, kan het model geoptimaliseerd worden om het gewenste resultaat te vinden. Het kan datapunten opdelen in categorieën(*Bijvoorbeeld:Dieren indelen in groepen*) maar het kan ook numerieke waarden voorspellen(*Bijvoorbeeld:Gewicht aan de hand van lengte voorspellen*).

Unsupervised learning werkt niet met gelabelde data en daarom is het van tevoren niet bekend wat het gewenste resultaat is. Unsupervised learning probeert verborgen patronen in de data te vinden. Unsupervised learning wordt vooral gebruikt om data te clusteren(Delua, 2021).

Veel methodes die gebruikt kunnen worden om fouten te voorspellen vallen weg. Supervised learning methoden vallen weg omdat een eis van de NDW is dat de ML methode nieuwe fouten moet vinden die nog niet door een mens gevonden kunnen worden. De validatieregels vangen al de fouten op die door een mens gevonden kunnen worden. Een ML oplossing die dezelfde fouten kan vinden zal geen toevoeging hebben aan het validatiesysteem van de NDW.

NDW wil ook niet dat er van tevoren al bekend moet zijn welke gegevens fout en welke gegevens goed moeten zijn, wat met supervised learning wel het geval is.

Unsupervised methoden zijn vooral cluster methoden. Dit heeft VorTech al geprobeerd en ze hebben geconcludeerd dat het geen toegevoegde waarde had. Het is niet nodig om dit experiment te herhalen.

De supervised en unsupervised methoden kunnen niet direct worden gebruikt om de fouten in de data te vinden. Het Machine Learning model kan dus niet goed of fout als output geven. Wel kunnen ze gebruikt worden om de patronen in de data vast te leggen en kunnen afwijkingen van deze patronen gemarkeerd worden als fout. Er kan bijvoorbeeld wel een supervised ML model worden getraind om verkeer te voorspellen. Hierdoor kan een model het patroon leren en kunnen gemeten afwijkingen van dat patroon gemarkeerd worden als incorrect.

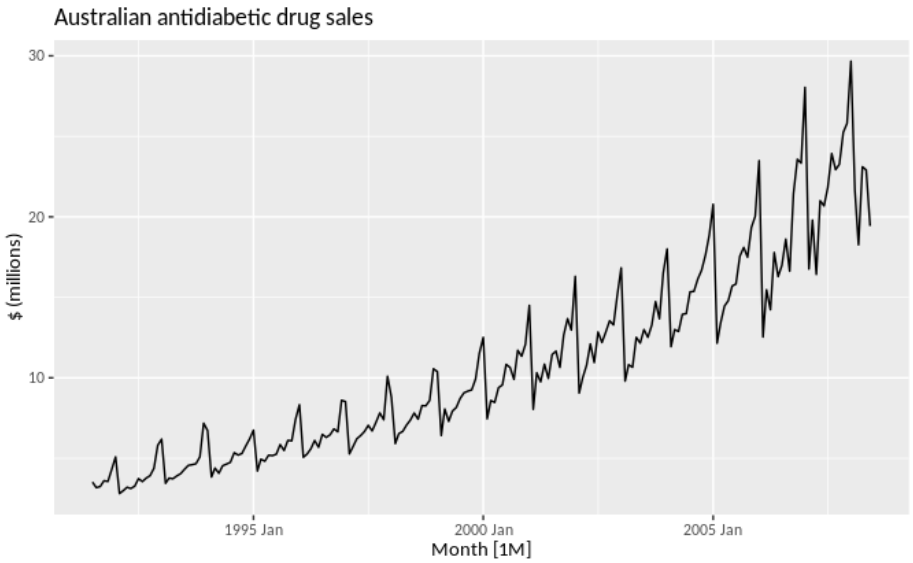
Uit de sample data is duidelijk geworden dat de NDW data tijdreeks data is. Elke minuut meet NDW een aantal variabelen over het verkeer. Tijdreeksen kunnen patronen bevatten en deze kunnen vastgelegd worden door een ML methode.

8.3.1.1 Tijdreeksen

Tijdreeksen ontstaan wanneer dezelfde variabelen op verschillende momenten gemeten worden(CBS, 2020). Tijdreeksen hebben 3 eigenschappen: trend, seizoen en cyclus.

- De trend is de algemene daling of stijging van een tijdreeks die over de tijd plaatsvindt.
  Voorbeeld trend: Er rijden meer auto's op een weg omdat de verbonden stad steeds meer inwoners krijgt. Het verkeersgedrag(spitsuur) blijft hetzelfde alleen in een hoger volume.
- De seizoenaliteit is het terugkerend patroon in een tijdreeks over een vast periode.
  Voorbeeld seizoenaliteit: Een seizoenaliteit in de verkeersdata is het terugkerend spitsuur. De spits zal elke dag (vaste periode) terugkomen op hetzelfde tijdstip (terugkerend patroon).
- De cyclus in een tijdreeks is ook een terugkerend patroon, alleen vindt de cyclus niet plaats over een vaste periode. De cyclus heeft ook een periode langer dan twee jaar, terwijl seizoenaliteit een periode heeft minder dan een jaar.
  Voorbeeld: Temperatuur op aarde heeft tussen ijstijden ongeveer hetzelfde patroon, maar er zit niet altijd evenveel tijd tussen ijstijden (Geen vaste periode).

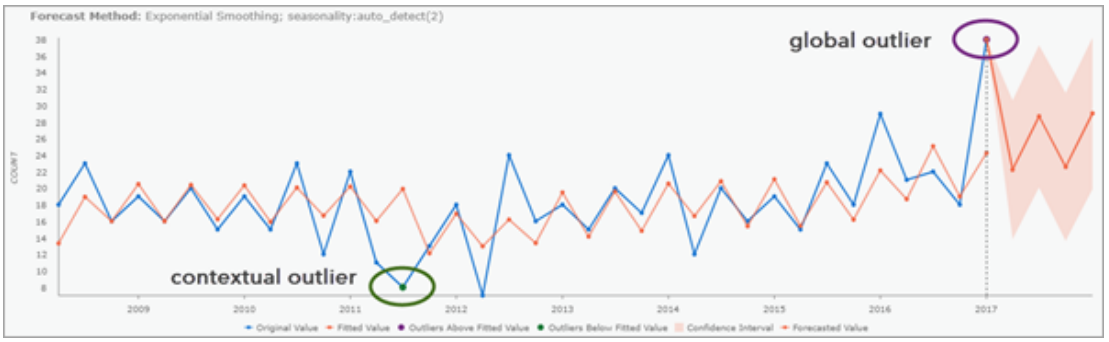
(Hyndman & Athanasopoulos, 2021, H2.3 Time series patterns).



Afbeelding 2: Tijdreeks met seizoenaliteit en trend (Hyndman & Athanasopoulos, 2021, H 2.2 Time plots)

Uitschieters binnen tijdreeksen zijn waarden die verschillen van de patronen en trends van de andere waarden in dezelfde tijdreeks (ArcGIS, z.d.). Er zijn verschillende soorten uitschieters(Cohen, z.d.):

- Globaal: Dit zijn uitschieters die buiten de hele reeks vallen
- Contextueel: Dit zijn uitschieters die anders zijn dan datapunten in dezelfde context. Bijvoorbeeld: De waarde van één dinsdag is anders dan alle andere dinsdagen.
- Collectief: Dit zijn uitschieters die niet op zichzelf globale of contextuele uitschieters zijn maar gegroepeerd wel een uitschieter zijn



Afbeelding 3: Voorbeeld contextuele en globale uitschieter (ArcGIS, z.d.)

Dit project hoeft niet expliciet te zoeken naar globale uitschieters. Als iets een globale uitschieter is dan is het ook een contextuele uitschieter. De onderscheid tussen deze twee soorten uitschieters kan belangrijk zijn voor andere projecten maar dit is voor dit project niet het geval. De contextuele uitschieters zijn wel interessant. Dit zijn afwijkingen van het normale seizoenspatroon.

Collectieve uitschieters zijn ook interessant. Dit zijn uitschieters die op zichzelf niet uitschieters zijn maar gegroepeerd wel.
Voorbeeld collectieve uitschieters: Er worden metrieken bijgehouden over het aantal bezoekers van een website en de cpu-gebruik van de server. Normaliter zal het cpu gebruik omhoog gaan als er veel verkeer is op de website. Als het aantal bezoekers wel omhoog

gaat maar niet het cpu gebruik, dan is er sprake van een collectieve uitschieter. Het aantal bezoekers dat omhoog gaat is normaal en een stabiel cpu-gebruik is ook normaal. Beide metingen gecombineerd is een uitschieter.

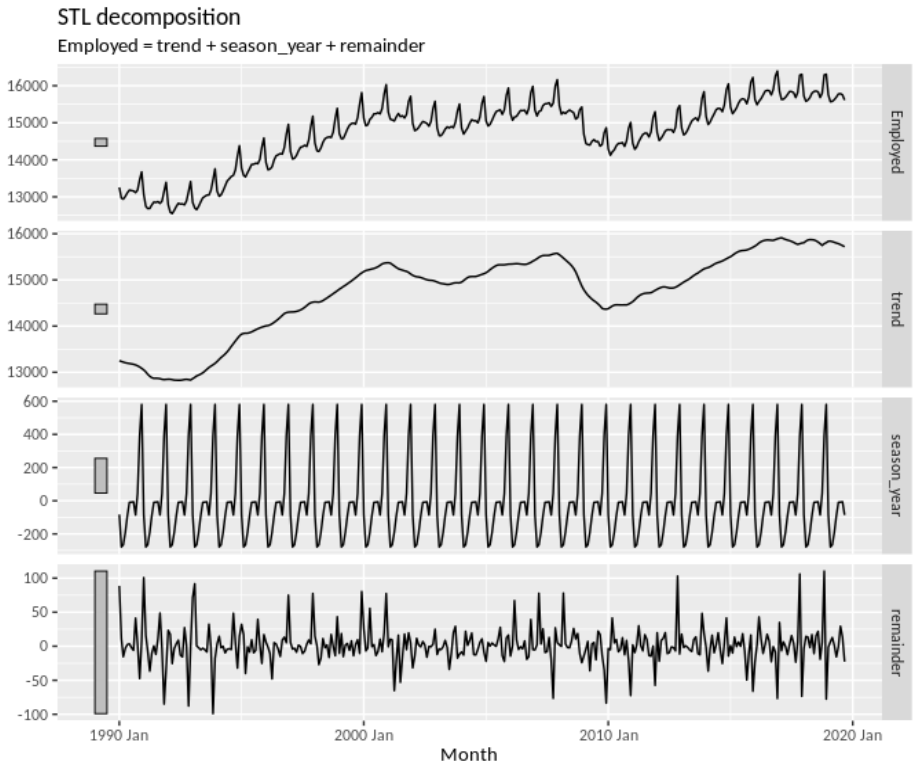
Er zijn methode om de patronen in tijdreeksdata vast te leggen. Dit kan met decompositie of door een voorspellend model te trainen. Hierna kunnen met toetsen afwijkingen van deze patronen gevonden worden. Tijdens dit onderzoek worden er twee methoden onderzocht om verkeersdata te valideren. Er worden maar twee methode onderzocht omdat het vinden en implementeren van machine learning modellen lang kan duren. Hiernaast moet er ook nog een POC gebouwd worden.

De eerste van de twee methoden onderzocht, is de STL met toets methode. De andere methode onderzocht in dit project is het voorspellen model en voorspellingsinterval methode. Deze twee methode zijn gekozen omdat deze in de praktijk voorkomen. Ook worden deze methoden niet gebruikt door NDW of zijn ze onderzocht door VORtech.

8.3.1.2 STL met toets methode

Eén van de methode om patronen in de data vast te leggen is decompositie. Met Seasonal and Trend decomposition using Loess(STL) kan de trend en seizoenaliteit van een tijdreeks vastgelegd worden(Hyndman & Athanasopoulos, 2021, H 3.6 STL decomposition). STL is een automatisch iteratief algoritme.

Afbeelding 4 toont decompositie van een tijdreeks. De bovenste grafiek is de originele data. STL is in staat om de trend van de bovenste grafiek vast te leggen in de tweede grafiek. Hierna kan het ook de seizoenaliteit van de bovenste grafiek vastleggen in de derde grafiek.



Afbeelding 4: Voorbeeld STL waarbij de data getoond in de bovenste grafiek is ontleed in zijn trend, jaarlijks seizoen en restwaarde. De restwaarden heeft pieken welke uitschieters kunnen zijn. (Hyndman & Athanasopoulos, 2021, H 3.6 STL decomposition)

Naast de seizoenaliteit en trend kan het ook de rest waarden vastleggen. De rest waarden zijn de waarden die overblijven wanneer de trend en seizoenaliteit van de originele data wordt afgetrokken(Hyndman & Athanasopoulos, 2021, H 3.2 3.2 Time series components).

Voorbeeld: STL vindt dat er op dinsdag 12:00 gewoonlijk 20 auto's rijden(seizoenaliteit) over een bepaald meetpunt. Omdat het nu de wintermaanden zijn rijden er op de weg 5 minder auto's(trend). Het meetpunt heeft deze dinsdag om 12:00 50 auto's geteld. De restwaarde is 50 - (20-5) = 30.

De rest waarden zijn spelingen in de originele data die niet uitgelegd kunnen worden door de trend of seizoenaliteit. De rest waarden zijn ook te zien in afbeelding 4. Dit is de onderste grafiek. De grote pieken die in de rest waarden te zien zijn, zijn grote afwijkingen van de trend en seizoenaliteit. Deze kunnen mogelijke uitschieters zijn.

Er moet een sterk terugkerend patroon in de data zijn wil STL goed werken. Als er geen sterk terugkerend patroon is dan zal het niet goed de seizoenaliteit kunnen vastleggen. Als het niet de seizoenaliteit kan vastleggen, zullen de restwaarden ook niet nauwkeurig zijn.

8.3.1.2.1 Toetsen

Deze pieken in de restwaarden kunnen gevonden worden met statistische toetsen. De toetsen die gebruikt worden in dit project zijn:

- Interkwartielafstand(IQR) toets: Bepaalt uitschieters aan de hand van de mediaan en interkwartielafstand(Khan Academy, 2016).
- Median absolute deviation(MAD) toets: Bepaalt uitschieters aan de hand van mediaan en de mediaan absolute deviatie(Lin, 2019).
- Generalized extreme Studentized(GESD) deviate toets: Bepaalt uitschieters aan de hand van de hand van een normaalverdeling(NIST, 2012, H 1.3.5.17.3. Generalized ESD Test for Outliers).

Alle drie toetsen werken ongeveer op dezelfde manier. Ze vinden eerst de mediaan. Hierna zullen ze op hun eigen wijze afwijkingen vinden van die mediaan. De strengheid van deze toetsen kunnen aangepast worden waardoor ze meer of minder uitschieters vinden.

8.3.1.2.2 Verantwoording keuzes STl en toets

STL en toets methode is gekozen omdat het al door verschillende libraries gebruikt wordt om uitschieters te vinden:

- De anaomalize package in R
- De AnomalyDetection van Twitter in R

Hiernaast wordt het ook in verschillende blogs aanbevolen:

- Bajaj, A. (2021), *Anomaly Detection in Time Series: 2021.*
- Tiunov, P. (2017, 6 8). *Time Series Anomaly Detection Algorithms*.

In het boek van Hyndman & Athanasopoulos *Forecasting: Principles and Practice (3rd ed) H13.9 Dealing with outliers and missing values* wordt de methode met STL en toets ook aangeraden om uitschieters te vinden in tijdreeksdata.

STL is niet de enige methode die decompositie kan uitvoeren op tijdreeksdata. STL kan een aantal dingen die andere methode niet kunnen. STL kan met uur en dag seizoenaliteit omgaan en ook meerdere seizoenaliteiten tegelijk uit de data halen. STL kan ook robuust gemaakt worden waardoor de trend en seizoenaliteit minder beïnvloed worden door uitschieters.

De gekozen toetsen zijn standaard statistische toetsen. Deze toetsen zijn gekozen omdat deze de mediaan gebruiken om uitschieters te vinden. De mediaan en afgeleide waarden van de mediaan worden niet beïnvloed door uitschieters. Met het gemiddelde is dit wel het geval.

8.3.1.3 Voorspellend model en voorspellingsinterval methode

De tweede methode om tijdreekspatronen vast te leggen zijn voorspellende modellen. Voorspellende modellen moeten de patronen(zoals de trend en seizoenaliteit) in de data leren willen ze het voorspellen.

Het verschil tussen een voorspellend model en STL is, dat STL de trend en seizoenaliteit vastlegt. Hierna doet STL er niets meer mee. Een voorspellend model leert de trend en seizoenaliteit gebruiken om voorspellingen te maken.

Als het model goed kan voorspellen dan is het bekend wat de verwachte waarde moeten zijn. Als een gemeten punt afwijkt van de verwachting dan kan dit een uitschieter zijn. Tijdens dit project worden de modellen ETS en ARMA gebruikt om verkeersdata te voorspellen. Dit zijn beide supervised methode om verkeer te voorspellen.

8.3.1.3.1 ETS

ETS staat voor exponential smoothing. ETS voorspellingen zijn gewogen gemiddelde van vorige waarden, waarbij de gewichten exponentieel afnemen naarmate de waarnemingen ouder worden. In andere woorden, de recentere waarnemingen hebben een hoger gewicht. ETS is in staat om snel betrouwbare voorspellingen te generen voor veel verschillende tijdreeksen. (Hyndman & Athanasopoulos, 2021, H Chapter 8 Exponential smoothing)

8.3.1.3.2 ARIMA

ARIMA staat voor autoregressive integrated moving average. ARIMA probeert toekomstige waarden te voorspellen aan de hand van de autocorrelatie in de data. De autocorrelatie is de samenhang van het huidige punt met de vorige waarden in de data(Hyndman & Athanasopoulos, 2021, H 9 ARIMA models).

Beide modellen vertrouwen erop dat er sterk terugkerende patroon te vinden zijn in de data. Ze vertrouwen beide op vorige waarden om data te voorspellen. Als er een terugkerend patroon is en het signaal niet willekeurig beweegt, dan zal het beter in staat zijn om te voorspellen.

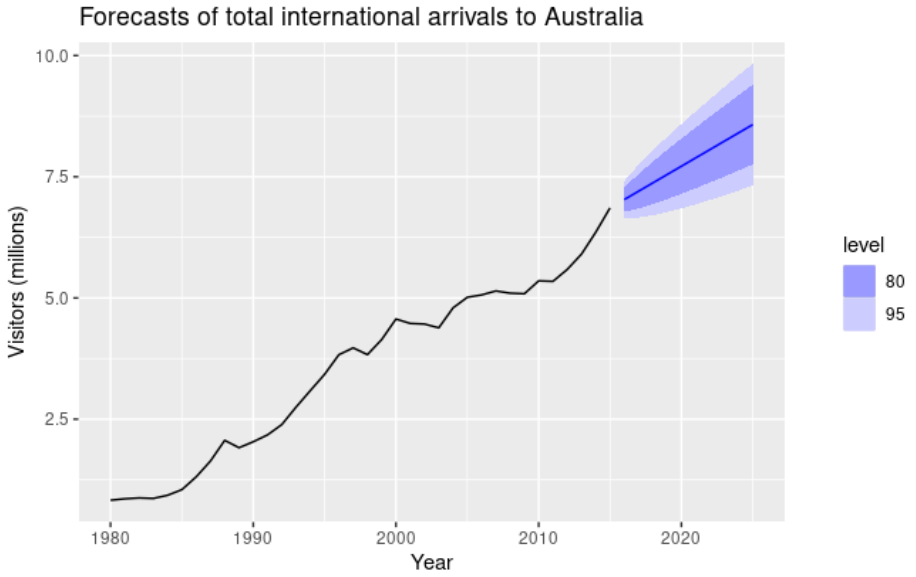
Training en testset
Voor het maken van een model wordt de data waarvan het model leert opgedeeld in twee datasets: De trainingsdata en testdata. De trainingset wordt gebruikt om het model te trainen. Hierbij worden de parameters van het model geschat. Hierna kan het model gebruikt

worden om de waarden in de training set te voorspellen. De error op de trainingset geeft aan hoe goed het model in staat was om de patronen in de trainingsdata te herkennen. Een model met een lagere error heeft het patroon beter herkend.

De testset is om het model te testen op data dat het nog niet gezien heeft. Het model probeert de waarden in de test set te schatten. De fout die het model maakt over de test geeft aan hoe goed het model werkt op data dat het nog niet heeft gezien. Een lagere error is beter (Hyndman & Athanasopoulos, 2021, H 5.8 Evaluating point forecast accuracy).
Om het beste soort model te vinden worden de test en training set van verschillende soorten modellen met elkaar vergeleken. Het model met de laagste error op beide sets, is het beste model.

8.3.1.3.3 Voorspellingsinterval

Wanneer een model voorspelt, kan er ook een voorspellingsinterval gegenereerd worden. Dit voorspellingsinterval geeft het interval waarin een volgend punt met een bepaalde zekerheid zal vallen. (Hyndman & Athanasopoulos, 2021, H 1.7 The statistical forecasting perspective)



Afbeelding 5 : Voorbeeld voorspellingsinterval met 80% zekerheid en 95% zekerheid (Hyndman & Athanasopoulos, 2021, H 1.7 The statistical forecasting perspective).

Wanneer een gemeten punt buiten dit voorspellingsinterval valt, betekent dat dit punt een onwaarschijnlijk punt is. In het boek Forecasting: Principles and Practice(3rd ed) worden de intervallen 80% en 95% gebruikt. In dit onderzoek wordt het 95% interval gebruikt omdat dit een strengere toets is en het zal minder onzekere gevallen teruggeven. Dit onwaarschijnlijk punt wordt met deze methode als uitschieter gemarkeerd.

8.3.1.3.4 Verantwoording keuzes voorspellend model en voorspellingsinterval

De methode voorspellend model en voorspellingsinterval wordt gebruikt op de minor Datascience op de Hogeschool van Amsterdam om afwijkende tijdreeks signalen te detecteren. Op de minor wordt vooral het ARIMA model gebruikt.

Google heeft deze methode ook gebruikt om fouten in webverkeer-tijdreeksen te vinden met een positief resultaat(Shipmon et al., n.d.).

Ook worden in blogs deze methoden aangeraden om uitschieters in tijdreeksdata te vinden:
- Bajaj, A. (2021), *Anomaly Detection in Time Series: 2021.*
- Tiunov, P. (2017, 6 8). *Time Series Anomaly Detection Algorithms*.

De ARIMA en ETS modellen zijn gekozen om voorspellingen te maken. Er zijn andere modellen om voorspellingen te maken. Deze twee modellen worden het meest gebruikt voor het voorspellen van tijdreeksen. Deze modellen zijn relatief eenvoudig, waardoor ze sneller getraind zijn. Meestal zijn deze modellen goed genoeg om een nauwkeurige voorspelling te maken(Hyndman & Athanasopoulos, 2021, H 9 ARIMA models).

8.3.1.4 Conclusie

Supervised en unsupervised methoden kunnen niet direct worden gebruikt om foute en correcte data punten te markeren. Wel kunnen deze methoden gebruikt worden om patronen in de data te leren. De afwijkingen van deze patronen zullen dan uitschieters zijn, welke weer mogelijk technisch foute data zijn.

De NDW data is tijdreeksdata. Tijdreeks data kan patronen bevatten. Er zijn twee methoden om deze patronen vast te leggen en afwijkingen van deze patronen te vinden:
- De STL en toets methode
- Voorspellend model en voorspellingsinterval methode

Met het dataonderzoek moet onderzocht worden of er een sterk terugkerend patroon(seizoenaliteit) aanwezig is in de data. Deze methoden zullen beter werken op data met sterk terugkerende patronen.

8.3.2 Hoe is de data van NDW gestructureerd?

Met deze deelvraag wordt beantwoord of er een sterk terugkerend patroon terug te vinden is in de verkeersdata. Hiernaast wordt er ook gekeken naar de data zelf en wat deze inhoudt.

8.3.2.1 Verkeersgegevens NDW

De verkeersdata dat onderzocht wordt zijn de actuele verkeersgegevens(NDW, 2021).
Dit onderzoek zal zich alleen focussen op de intensiteit en snelheid(Bijlage gesprek: 5-3-2021 Interview Willem van Loon).

De data bevat ongeveer 18.000 meetlocaties. Ongeveer 5000 daarvan worden gemonitord door NDW. De data bestaat uit vijf variabelen. Deze variabelen zijn:
- Tijd: Tijdstip waarop de variabelen zijn gemeten. Elke minuut wordt er gemeten.
- Id: Id van de meetlocatie.
- ndw_index : Een meetlocatie heeft meerdere ndw indexen. De ndw index geeft aan over welke rijstrook het gaat. Ook bevat de index informatie over de voertuigcategorie. De voertuigen zijn opgedeeld in categorieën qua lengte. *Voorbeeld ndw index is F6C, welke verwijst naar rijstrook 1 en voertuigcategorie alle voertuigen. Een andere index is F23C welke verwijst naar rijstrook 2 en alle voertuigen met een lengte langer dan 12 meter.*

- gem_intensiteit: Gemiddelde intensiteit(aantal auto's) op dit minuut tijdstip
- gem_snelheid: Gemiddelde snelheid op dit minuut tijdstip

8.3.2.2 Steekproef meetpunten

Het is onmogelijk om alle 5000 meetpunten te plotten en samen te vatten. Om het behapbaarder te maken zijn er een 30 willekeurige meetpunten gekozen om mee te onderzoeken. Hiernaast is er ook alleen gekeken naar data van 1 januari 2021 tot 1 maart 2021 om het nog makkelijker te maken.

Als laatst zijn de 6 meetpunten gekozen die defect waren in de periode 1 januari 2021 en 1 maart 2021, om onderzoek te doen naar deze meetpunten. Er is een bestand online gepubliceerd door de NDW, dat aangeeft welke meetpunten op welk moment defect zijn. Dit bestand is gebruikt om de meetpunten te selecteren.

8.3.2.3 Tijdsniveaus

Er is met twee tijdsniveaus geëxperimenteerd, minuut en uurniveau. De uurdata is een aggregatie van de minuutdata door het gemiddelde van de minuut intensiteit en snelheid binnen een uur te nemen.

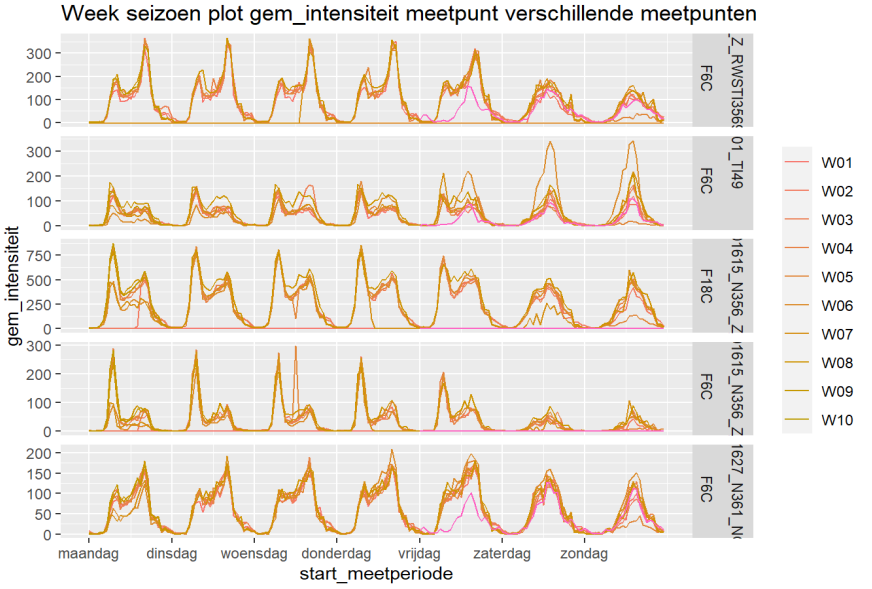
De minuutdata is erg druk. Er zit veel speling in de minuutdata. Het werken met minuutdata is ook traag, omdat computerprocessen een stuk langer duren met minuutdata(minuutdata duurt ong. 16 keer zo lang). Hiernaast is er ook geen terugkerend patroon gevonden binnen een uur op minuutniveau. De minuutdata is daarom niet verder meegenomen in het onderzoek. Er wordt ook niet met etmaaldata gewerkt omdat er over een dag ook nog een patroon te zien is. Met etmaaldata is deze informatie verloren.

8.3.2.4 Seizoensplot

De twee gevonden methoden in het literatuuronderzoek verwachten sterke terugkerende patronen in de data en afwijkingen van deze patronen zijn uitschieters. Om te testen of de tijdreeks wel patronen heeft, kan er een seizoensplot gemaakt worden.

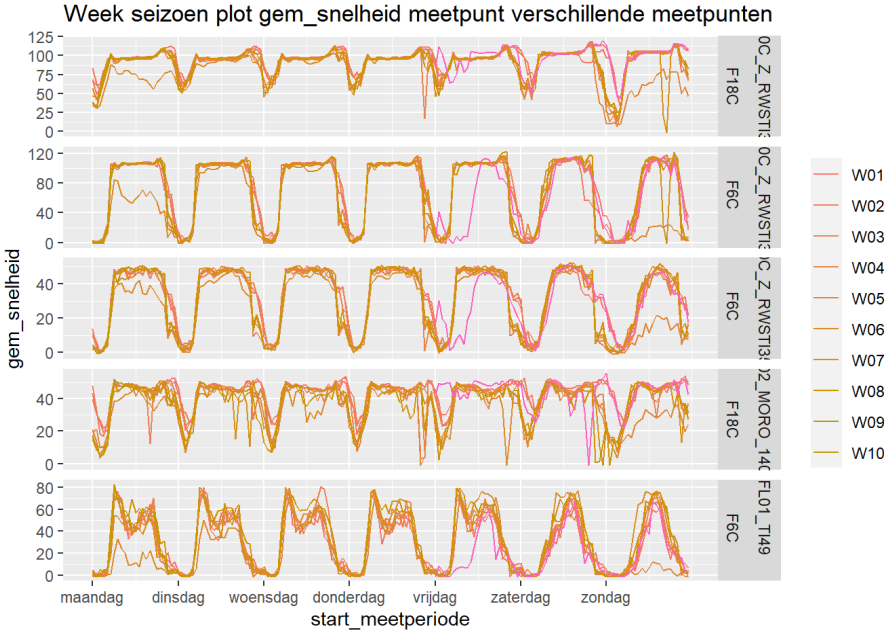
Een seizoensplot knipt het verkeerssignaal in stukken. De lengte van elk stuk staat gelijk aan de seizoenslengte. Als er een seizoenslengte is van 1 week (oftewel een patroon die elke week terugkomt) dan zal de seizoensplot het verkeerssignaal opdelen in stukken van 1 week. Het seizoensplot plot alle stukken van het signaal over elkaar heen. Als er een terugkerend patroon is dan zullen alle stukken ongeveer hetzelfde lopen. Als er geen patroon is dan zullen alle stukken kriskras door elkaar heen lopen.

Dit is gedaan in afbeelding 6 met de gem. intensiteit van meerdere meetlocaties en de index F6C(alle voertuigen, rijstrook 1). Er zijn 5 grafieken waarbij elke grafiek de gemiddelde intensiteit van één meetpunt toont. De lijnen zijn mooi in lijn en dit betekent dat elke week ongeveer hetzelfde patroon terugkomt voor de intensiteit. Er is dus een wekelijks patroon. Er zijn ook twee dagelijkse patronen. De eerste 5 pieken in elke grafiek zijn de weekdagen. Deze lijken allemaal op elkaar. Hierna komen de weekenddagen, welke ook op elkaar lijken.



Afbeelding 6: Seizoensgrafiek gemiddelde intensiteit van willekeurig meetpunten

Dit is nog eens gedaan met de gem. snelheid, waar ook een terugkerend patroon te zien is. Met de snelheid is er wel een kleiner verschil tussen de week en weekenddagen.



Afbeelding 7: Seizoen plot gemiddelde snelheid van willekeurig meetpunten

8.3.2.5 Conclusie

Tijdens dit onderzoek worden de snelheid en verkeersintensiteit variabelen van de AVG onderzocht. Er is met een siezoensplot getest of er een sterk terugkerend patroon(seizoenaliteit) te vinden is in de data. Dit blijkt voor beide variabelen waar te zijn.

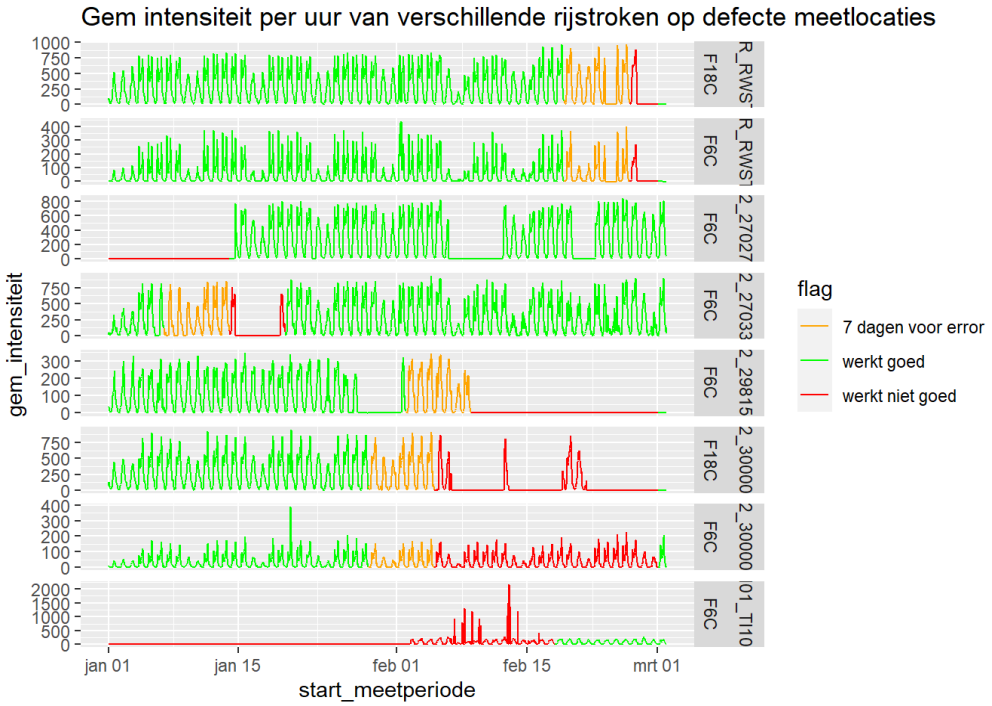
De NDW data heeft meerdere tijdsniveuas. De minuutdata is lastig om mee te werken. Hierom is er gekozen in overleg met de opdrachtgever om met de uurdata verder te gaan.

8.3.3 Wat voor soort datafouten bevinden zich in de data?

Om een idee te krijgen wat voor soort fouten NDW kan vinden zijn de defecte periode van verschillende meetpunten geplot. Dit zijn de 6 meetpunten die van 1 januari 2021 tot 1 maart 2021 defect waren, die ook terugkomen in de steekproef data.

8.3.3.1 Defecte meetpunten

In afbeelding 8 is de gemiddelde uurintensiteit van de eerste rijstrook van verschillende defecte meetpunten geplot. Elke grafiek is een ander meetpunt. Het rode stuk is de defect periode, en het oranje gedeelte 7 dagen ervoor. Het groene stuk is wanneer het meetpunt correct werkt. Er is te zien dat het signaal meestal platligt in de defecte periode. Het zou optimaal zijn als de twee methode van dit onderzoek foutieve punten konden signaleren in het oranje stuk. Hierdoor worden defecten vroegtijdig gevonden.



Afbeelding 8: gemiddelde intensiteit op defecte meetlocaties.

8.3.3.2 Conclusie

De periode welke NDW markeert als fout zijn meestal periodes wanneer het meetpunt helemaal niets meer doet. Deze informatie zal niet helpen in dit onderzoek. De defecte meetpunten kunnen wel meegenomen worden in de eindresultaten door de twee onderzochte validatiemethode op de data met defecte periodes te testen.

8.3.4 Welke eisen worden gesteld aan de ML methode vanuit NDW?

De volgende requirements zijn uit de opdrachtomschrijving gehaald. De opdrachtomschrijving bleek al een grote deel van de requirements te bevatten.

- NDW wil onderzoek doen naar ML oplossingen voor het valideren van data, om de huidige data validatie methoden te innoveren.
- NDW wil een ML oplossing die zelf data validatie regels kan aanmaken omdat zelf schrijven van data validatie methoden duur is.

- NDW wil een ML oplossing die zelf data validatie regels kan aanmaken zodat NDW niet complexe validatieregels hoeft te schrijven.
- NDW wil een ML oplossing die in staat is om fouten te detecteren zonder dat er op voorhand bekend is wat er fout is zodat NDW niet zelf validatieregels hoeft te schrijven.
- De leveranciers en gebruiker moeten met de informatie verschaft door de methoden kunnen bepalen, welke meetpunten niet werken op etmaalniveau.
- De leveranciers en gebruiker moeten met de informatie verschaft door de methoden kunnen bepalen, welke datapunten foutief zijn op minuutniveau.
- De leveranciers en gebruiker moeten met de informatie verschaft door de methoden kunnen bepalen waarom punten fout zijn gemarkeerd.
- De beheerder moet met de informatie verschaft door de methoden kunnen bepalen welke punten niet zeker zijn voor markering.
- Het systeem moet met een ML methoden data valideren.
- Het systeem moet defecte meetpunten op etmaal-niveau kunnen aangeven in de data.
- Het systeem moet foutieve datapunten op minuut-niveau kunnen aangeven in de data.
- Het systeem moet een onderscheid kunnen maken tussen zekere markeringen en onzekere markeringen.
- Het systeem moet het soort fout kunnen markeren.
- Het systeem moet kunnen leren van de nieuw gevonden fouten zodat het deze in de toekomst beter kan identificeren.

Tijdens het gesprek met de opdrachtgever zijn ook de volgende eisen erbij gekomen.

- NDW wil een ML oplossing die zelf data validatie regels kan aanmaken zodat NDW ook fouten kan detecteren die niet door een mens gevonden kunnen worden.
- NDW wil onderzoeken hoe MLOps geïmplementeerd kan worden in de praktijk zodat NDW ook een MLOps systeem kan opzetten
- Het model moet andere fouten dan de validatieregels kunnen detecteren.
- Het model moet binnen 1 uur 24 uur data kunnen valideren (Het moet aanzienlijk minder zijn dan een dag).
- Het model moet in staat zijn om 24 uur data te verwerken.

8.3.5 Kunnen de ML methoden foutieve datapunten detecteren?

Er zijn twee methoden gevonden in de literatuur om tijdreeksdata te valideren:

- STL met statistische toets
- Voorspellend model en voorspellingsinterval

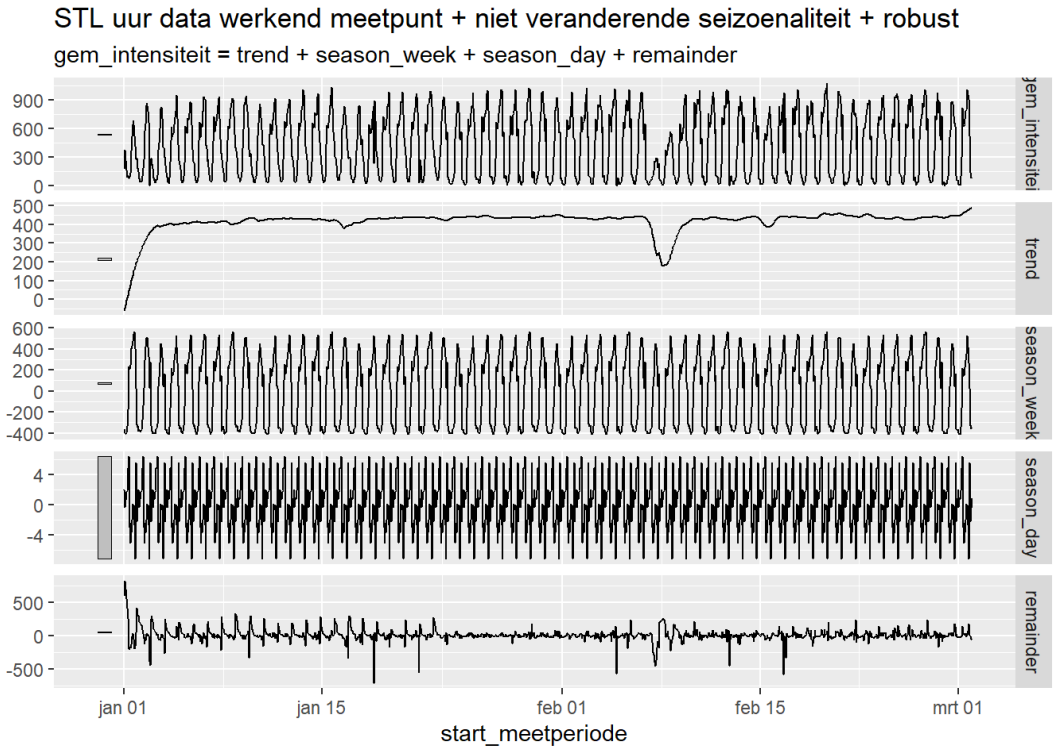
Deze twee methoden werken het beste op tijdreeksdata met sterk terugkerende patronen. Er is bevestigd dat de verkeersdata sterk terugkerende patronen heeft met de seizoenplots. Om te testen of deze twee methoden ook echt fouten kunnen vinden in de verkeersdata, worden er prototypes gemaakt.

Als de methoden ook echt verkeersfouten kunnen vinden, dan kan er een POC gebouwd worden met één of twee van deze methoden. Dit wordt onderzocht in deze deelvraag.

8.3.5.1 STL en toets prototyping

Eerst is er een prototype gemaakt van de STL en toets methode. De STL-methode is toegepast op de 36 gekozen meetpunten van het dataonderzoek.

Afbeelding 9 is alleen de STL methode toegepast op een meetpunt. Dit is STL uitgevoerd op de gemiddelde intensiteit van het meetpunt GRT02_MORO_1408_2 met index F6C (Rijstrook 1, alle voertuigen)



Afbeelding 9: STL uitgevoerd op de uurgemiddelde intensiteit van meetpunt GRT02_MORO_1408_2, index F6C

De bovenste grafiek zijn de originele waarden. De tweede grafiek is de trend. Hierna komen de wekelijkse en dagelijkse seizoenaliteit. De remainder zijn de waarden die niet uitgelegd kunnen worden door de trend of seizoenaliteit, oftewel afwijkingen van het standaard patroon. Op de overige waarden worden de toetsen uitgevoerd.

STL kan op verschillende manieren worden uitgevoerd door de gebruiker. Dit zorgt ervoor dat de resultaten van de decompositie ook net wat anders wordt. De manieren waarop de STL veranderd kan worden zijn:

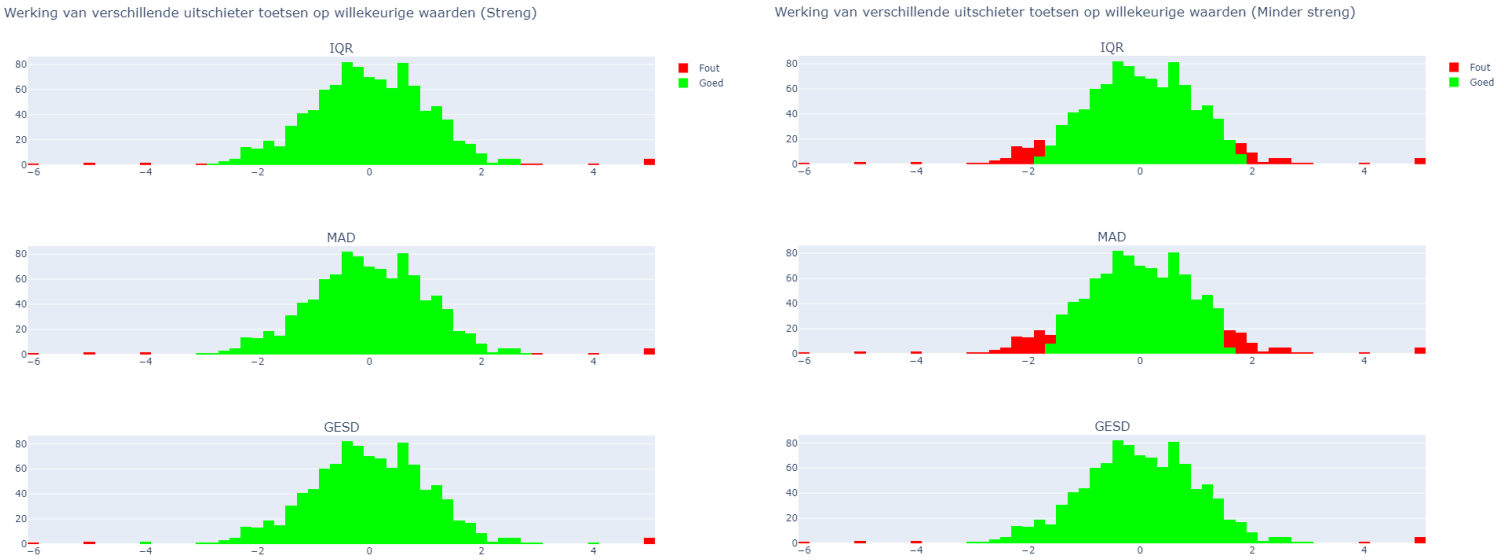
- De trend kan vast worden gezet, zodat deze een recht lijn wordt.
- De seizoenaliteit kan over de tijd veranderen
- De STL kan robust worden gemaakt, zodat uitschieters minder invloed hebben in de trend en seizoenaliteit en door worden gezet naar de rest waarden.

8.3.5.1.1 Toetsen

De IQR, MAD en GESD toetsen zijn geïmplementeerd en getest. In afbeelding 10 is te zien dat ze ongeveer hetzelfde werken. De toetsen detecteren uitschieters aan de uiteinden van de verdelingen. Hoe veel uitschieters deze detecteren is aanpasbaar door de strengheid van

de testen aan te passen. Afbeelding 10 toont dat ook. De linker grafiek toont aan wanneer de testen streng zijn gezet en de rechter grafiek wanneer de testen coulanter testen.

De GESD test is altijd streng en zal niet veel fouten detecteren. Dit is niet zozeer fout. De MAD en IQR lijken op elkaar. De MAD en IQR zijn beter dan de GESD omdat deze flexibeler zijn. De strengheid kan zo aangepast worden dat het in verschillende situaties kan functioneren.



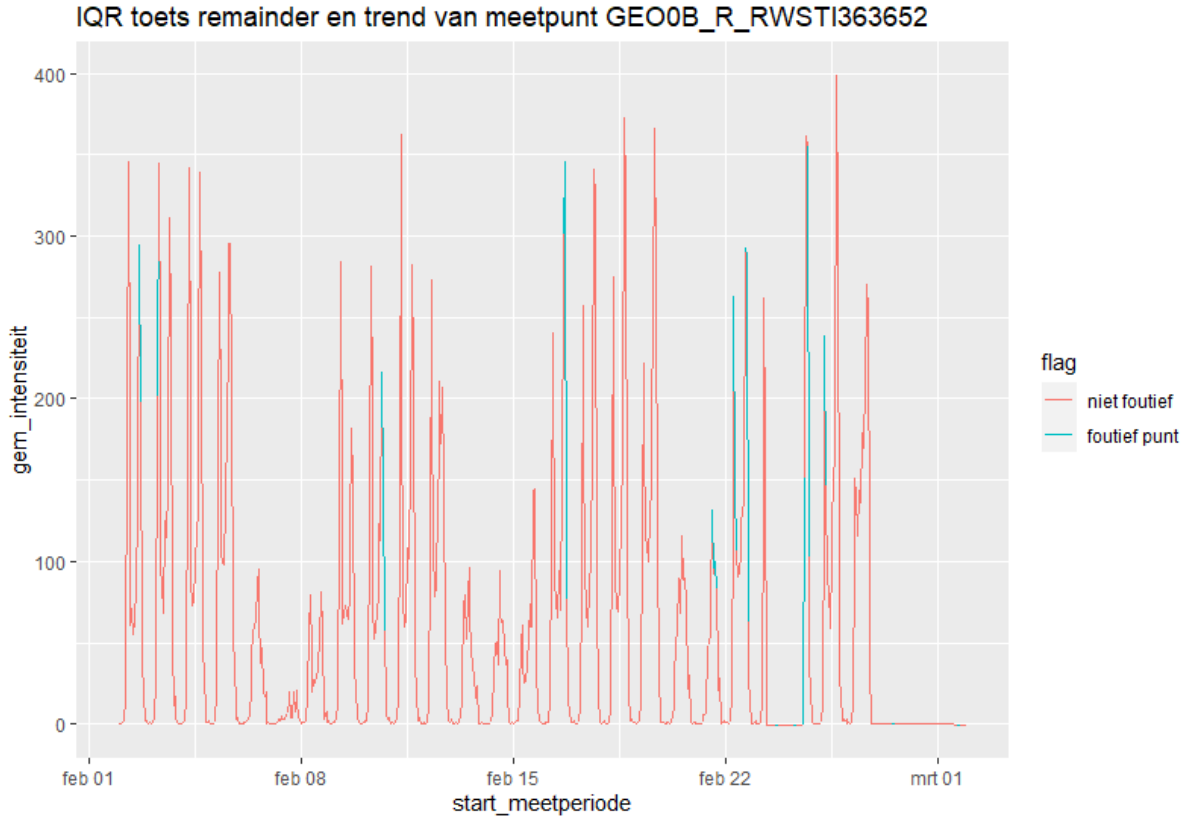
Afbeelding 10 : Vergelijking van verschillende toetsen op normaalverdelingen. De rode delen zijn delen wat de toetsen als fout markeren in een verdeling.

8.3.5.1.2 Toepassen op verkeersdata

Nu zijn de STL en de toetsen apart getest. Door eerst STL toe te passen op verkeersdata van een meetpunt en daarna de uitschieter toets toe te passen op de overige waarden, komen de volgende resultaten te zien in afbeelding 11. Het STL+toets is goed in staat om op feb 8 een periode te detecteren wat afwijkt.

Afbeelding 11: STL + IQR toegepast op meetpunt GRT02_MORO_1408_2 (werkend meetpunt)

Afbeelding 12 toont het resultaat van STL+IQR op een defect meetpunt dat defect ging rond 27 feb. De verwachting is dat feb 24 - maart 1 blauw wordt. Dit is niet het geval. Er worden veel andere plekken blauw gekleurd. Om dit resultaat te creëren moest de strengheid van toets ook veel strenger worden gezet dan de test in afbeelding 11, anders markeert het veel meer punten als foutief.



Afbeelding 12: STL + IQR toegepast op meetpunt GEO0B_R_RWSTI363652 (defect meetpunt)

Dit kan komen doordat STL alle punten meeneemt in het maken van de decompositie, dus ook de defecte periode. STL ziet de defecte periode ook als delen van het patroon. Hierdoor lijken deze “normaal”.

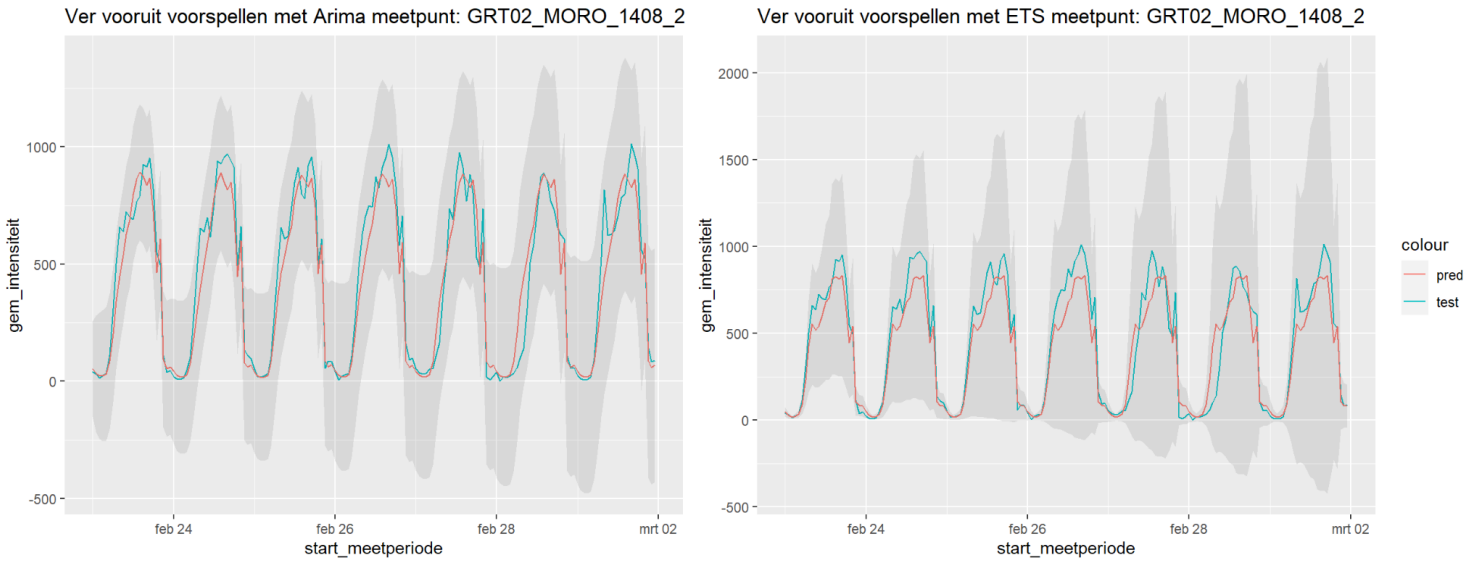
8.3.5.1.3 Conclusie STL en toets prototype

De STL werkt op de verkeersdata. Het is in staat om de verkeersgegevens te ontleden in trend, dag seizoenaliteit, week seizoenaliteit en de restwaarden. De toetsen werken allemaal maar IQR en MAD zijn flexibeler.

Wanneer de STL en toets samen worden toegepast werkt het soms wel en niet. Afbeelding 11 werkt het wel en het kan een afwijkende periode vinden. Afbeelding 12 kan het niet goed de defecte periode vinden. Ook moeten de parameters van STL en toets aangepast worden willen in verschillende scenarios fouten gevonden worden. Dit betekent dat er niet één set aan parameters is om fouten te vinden.

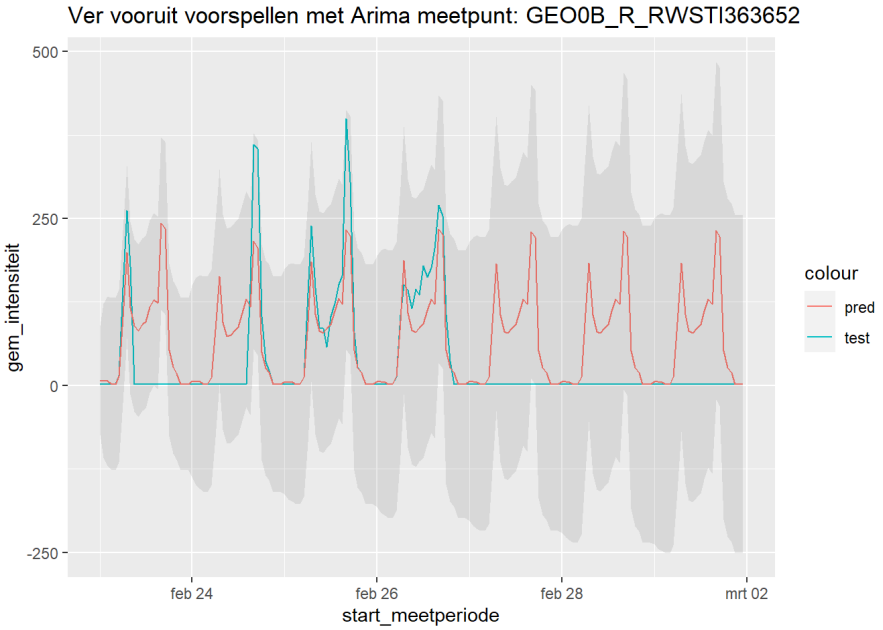
8.3.5.2 Model trainen en voorspellen prototyping

Hierna zijn er ook prototypes gemaakt van het ETS en ARIMA-model. Beide modellen zijn getraind op de trainingsdata van een meetpunt, waarna ze ze getest op de test set. De resultaten zijn te zien in afbeelding 13.



Afbeelding 13: Voorspellingen ARIMA en ETS model op hetzelfde meetpunt.

De ARIMA en ETS kunnen beide goed voorspellen. De rode lijn zijn de voorspelde waarden en de blauwe lijn zijn de gemeten waarden. In afbeelding 13 is te zien dat de voorspelde lijn de gemeten lijn nauw volgt. Het grijze gebied is het voorspellingsinterval. Het voorspellingsinterval geeft aan waar het volgend punt kan vallen met een bepaalde zekerheid. De zekerheid dat het binnen dat grijze gebied valt is 95%. Het grijze gebied van de ETS is veel groter dan die van ARIMA. Dit betekent dat er veel onzekerheid is in het voorspellen met ETS, wat ervoor zorgt dat het punt een grote uitschieter moet zijn om als fout te worden gemarkeerd.



Afbeelding 14: ARIMA toegepast op een defect meetpunt

Als het ARIMA model wordt toegepast op een defect meetpunt dan ziet het eruit als afbeelding 14. In deze afbeelding is te zien ARIMA het standaard signaal blijft voorspellen, wat goed is. Het probleem hier is het voorspellingsinterval. De gemeten waarde(blauwe lijn) vallen nauwelijks buiten het voorspellingsinterval. Hierom is het voorspellingsinterval niet te gebruiken. Wel is de afstand tussen de gemeten waarde en voorspelde waarde groot. Deze afstand kan gebruikt worden om te bepalen of iets een uitschieter is.

Een manier om de afstand te bepalen is om een procentuele waarde te nemen. Deze wordt berekend door de minwaarde in een tijdreeks af te trekken van de maxwaarde en hier een percentage van te nemen. Uit experimenten blijkt dat 25% van de max-min waarde een goede max afstand is om fouten te detecteren, zonder dat het veel vals positieve waarden teruggeeft.

8.3.5.2.1 Conclusie

Het ARIMA model is goed in staat om het signaal te recreëren. ETS is ook goed in staat om voorspellingen te maken.

Het voorspellingsinterval is niet te gebruiken. De afstand tussen het voorspelde punt en gemeten punt is misschien wel een indicatie om aan te geven of iets een uitschieter is.

8.3.5.3 Tussentijdse Conclusie prototyping

Er waren een aantal problemen tijdens het maken van de prototypes:

- Het tonen van de resultaten bij de belanghebbende is lastig. Op dit moment zijn er erg veel grafieken die bestudeerd moeten worden. Het is ook niet meteen duidelijk wat de grafieken tonen zonder dat iemand deze uitlegt.
- Het is niet altijd duidelijk of een datapunt daadwerkelijk een een technische fout of een verkeerskundige situatie is voordat een verkeersexpert er naar kijkt.
- Er zijn nog opties aan de ML modellen die veranderd kunnen worden waardoor de methoden iets anders werken.
- Heel veel handelingen worden tijdens het experimenteren handmatig uitgevoerd.
- Niet alles kon getest worden. In de experimenten hiervoor is er alleen onderzoek gedaan met gem intensiteit.

Er is nog veel onzekerheid over de methoden. Het onderzoek moet nog voortgezet worden wil er een duidelijk resultaat uit komen. Hierom moet er een POC gebouwd die help met het voortzetten van dit onderzoek. Met dit POC kan een verkeersexpert zelf ook makkelijk resultaten creëren en onderzoek doen. De verkeersexpert moet uiteindelijk beoordelen of deze methoden ook echt technische fouten vinden. De beste POC om deze problemen op te lossen is een dashboard.

Het onderzoeken of de twee methode van dit onderzoek ook echt fouten vinden in de verkeersdata wordt verder onderzocht. Er zijn wel een aantal veranderingen aan het onderzoek gemaakt aan de hand van de gevonden resultaten tijden het onderzoek. Deze zijn:

- Er komt een nieuw voorspellend model bij naast de ETS en ARIMA, namelijk het STL voorspellend model(STLVM). STL kan heel goed de patronen in de data vinden, alleen doet het niets met deze waarden. Door STL toe te passen op een periode

waarbij er geen fouten in de data zijn, zullen de foutieve datapunten niet het STL proces beïnvloeden. Met de gevonden STL waarden kunnen voorspellingen gemaakt worden. De STL en toets methode wordt nog verder onderzocht, alleen komt er ook nog een voorspellend model bij.
- ETS wordt niet verder meegenomen in het onderzoek, het voorspelt ongeveer hetzelfde als ARIMA en is hierom extra werk om te implementeren zonder toegevoegde waarde.
- In plaats van te kijken naar het voorspellingsinterval wordt er gekeken naar de afstand tussen het voorspelde punt en gemeten punt bij het voorspellend model en voorspellingsinterval methode(Vanaf dit moment wordt deze het voorspellend interval en afstand methode genoemd).
- Tijdens het experimenteren is er veel met de gemiddelde intensiteit gewerkt. De gemiddelde snelheid moet ook meegenomen worden in de eindresultaten. Beide variabelen moeten gecombineerd worden om een eindresultaat te geven. Er is gekozen om punten fout te markeren als deze op beide variabelen als foutief worden gedetecteerd. De variabelen kunnen ook nog eens individueel beoordeeld worden, wat de scope van dit project zal vergroten. Dit is niet gewenst.

8.3.5.4 Validatie modellen

Er worden dus twee voorspellingsmodellen meegenomen in de eindresultaten: ARIMA en STLVM. In de eindresultaten kan niet alles beschreven worden. Er is een max limiet aan paginas dat dit verslag kan hebben. Hierom moet er een keuze worden gemaakt welke voorspellend model er gebruikt wordt voor de eind resultaten. Om te testen welk model het beste is, is de *mean absolute scaled error(MASE)* berekend. De MASE geeft een indicatie hoe goed de modellen de verkeers variabelen kunnen voorspellen.

De MASE wordt berekend door de error van een model te vergelijken met een heel simpel voorspellend model. Het hele simpele model hier is de naïeve seasonal forecast. De naïeve seasonal forecast pakt het vorige punt op hetzelfde seizoen moment als voorspelling. *Voorbeeld: Vorige dinsdag reden er 10 auto's, dan zullen er deze dinsdag ook 10 auto's rijden.*

De errors van het getrainde model worden gedeeld door de errors van het simpele model. Hierna wordt het absoluut van deze waarden genomen en het gemiddelde berekend.

MASE = gemiddelde(| errors getraind model / errors simpel model |)

Door het delen wordt het een schaal vrije manier om te bepalen hoe goed een methode werkt. Een schaalvrije manier is gewenst omdat verschillende meetpunten verschillende schalen hebben, een meetpunt kan een gemiddelde intensiteit hebben van max 200 auto's en een andere van max 50 auto's omdat deze weg minder druk is. Door deze schaal weg te halen kunnen de methoden over verschillende wegen worden getest en samengevat hoe goed deze kunnen voorspellen.

Er zijn over de 30 meetpunten, die in het dataonderzoek willekeurig zijn geselecteerd, modellen getraind voor de index rijstrook 1 en alle voertuigen. Er zijn STLVM modellen en

ARIMA-modellen getraind. Hierna is er voor elk meetpunt een MASE berekend. De MASE van alle meetpunten is hierna samengevat door het gemiddelde te berekenen.

De resultaten zijn te zien in tabel 1. Een getal onder 1 betekent dat het model beter kan voorspellen dan het hele simpele model. Dit is nooit het geval. Het hele simpele model is niet meegenomen in het onderzoek omdat dit niet een ML methode is. Het is wel handig om dit in vervolgonderzoek mee te nemen omdat het heel goed kan voorspellen. Een getal van 2 zal betekenen dat de gemiddelde fout van het ML model 2 keer zo groot is.

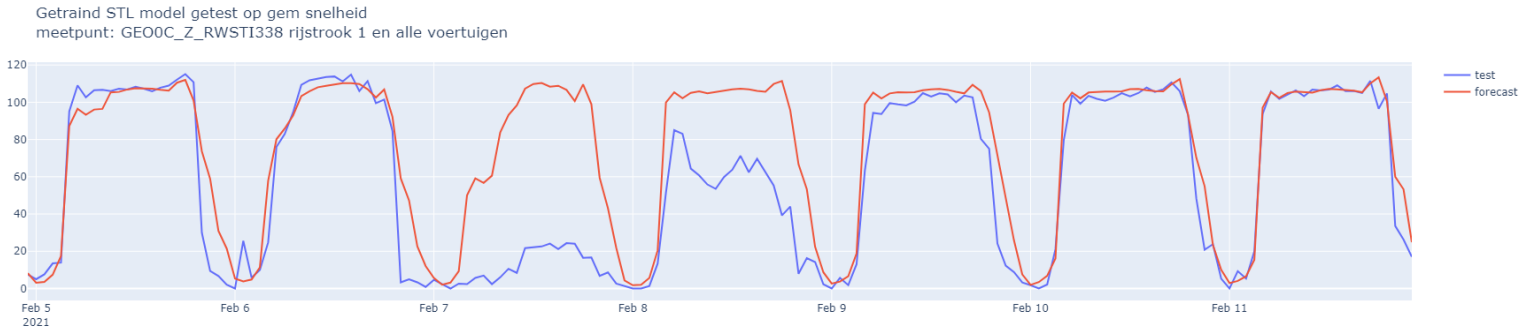
De STLVM is over de training sets slechter dan de ARIMA. Wel doet de STLVM het beter op de test sets. Dit betekent dat STLVM het beter doet op data dat het nog niet gezien heeft. Dit betekent ook dat STLVM het algemene patroon beter heeft weten vast te leggen.

Tabel 1 : Vergelijking ARIMA en STL voorspellend model met mase

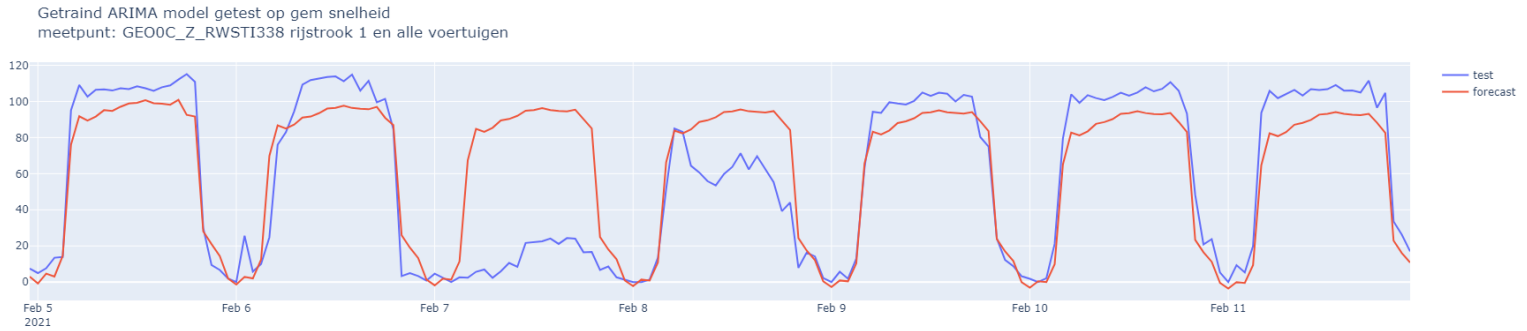
	ARIMA-model	STL voorspellend model
Gemiddelde train set MASE intensiteit	3.4	3.6
Gemiddelde test set MASE intensiteit	5.7	3.5
Gemiddelde train set MASE snelheid	4.5	177.7
Gemiddelde test set MASE snelheid	11.3	3.9
Gemiddelde trainingstijd intensiteit	11.6	6.3
Gemiddelde trainingstijd snelheid	7.6	6.3

De gemiddelde trainingsduur van de modellen is ook meegenomen. Het STLVM is over het algemeen ook sneller getraind (Hyndman & Athanasopoulos, 2021, H 5.8 Evaluating point forecast accuracy).

Afbeelding 15 en 16 tonen de vergelijking van de model voorspelling en de daadwerkelijke waarden. De blauwe lijnen zijn de daadwerkelijke waarden en de rode de voorspelde waarden. Rond feb 7 en 8 was er ijzel op de weg. Hierdoor was er minder verkeer. Beide modellen konden dit niet voorspellen. De rest voorspelde ze wel goed. Vooral het STLVM is heel goed in het voorspellen van de waarden.



Afbeelding 15: Resultaten STLVM voorspelling gem snelheid op de test data



Afbeelding 16: Resultaten ARIMA voorspelling gem snelheid op de test data

8.3.5.5 Eindresultaten

Dit subhoofdstuk bevat de resultaten die gebruikt worden om deze deelvraag en uiteindelijk de hoofdvraag te beantwoorden. De resultaten zijn gegenereerd met behulp van de POC.

Een eis was dat de methoden fouten konden detecteren zonder dat er van tevoren bekend was wat fout of goed is. Er is geen dataset beschikbaar waarin is aangegeven wat fout en goed is(gelabelde dataset). Doordat er geen gelabelde dataset is, kan er ook niet een metriek ontwikkeld worden om te testen hoe goed deze methoden werken. Er kan niet onderzocht worden hoeveel procent van de fouten de methoden uit de data kunnen halen, of hoeveel procent van de fouten de methoden niet kunnen vinden. Er moet toch besloten worden of de methoden wel of niet goed werken.

De resultaten zijn in de vorm van interessante observaties. Er is door de data gekamd voor interessant bevindingen. Deze bevindingen tonen hoe de methoden interacteren met de verkeersdata en waar het fouten detecteert. Er is naar twee soorten verkeersdata gekeken: verkeersdata van defecte meetpunten en willekeurige meetpunten.

De methoden zijn toegepast op de defecte meetpunten vlak voor en net in de periode waar het meetpunt als defect is gemarkeerd. Deze test toont aan hoe deze methoden interacteren met defecte meetpunten. De beste scenario is wanneer de methoden de defecten zien aankomen, oftewel een periode voor de defect al fouten aan beginnen te geven.

Er zijn ook willekeurige meetpunten geselecteerd waar de methoden op toegepast zijn. Deze tonen hoe de meetpunten interacteren met de meetpunten in het algemeen. De periode waarover de meetpunten zijn getest zijn ook willekeurig.

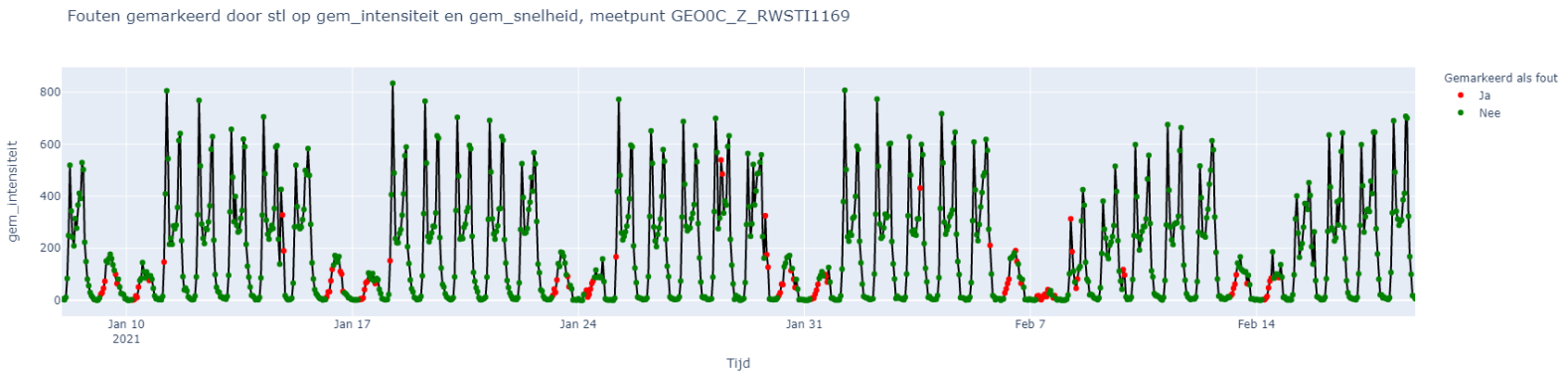
Er is ook alleen gewerkt met uurdata en rijstrook 1 alle voertuigen indexen. Op de meest rechter rijstrook zullen er meer auto's rijden en hebben deze hierom de beste patronen.

8.3.5.5.1 STL methode met toets

De volgende resultaten zijn de resultaten van de methode STL en toets. Voor alle resultaten in dit hoofdstuk wordt de IQR-toets gebruikt. Tijdens het experimenteren is ontdekt dat de toetsen ongeveer hetzelfde werken.

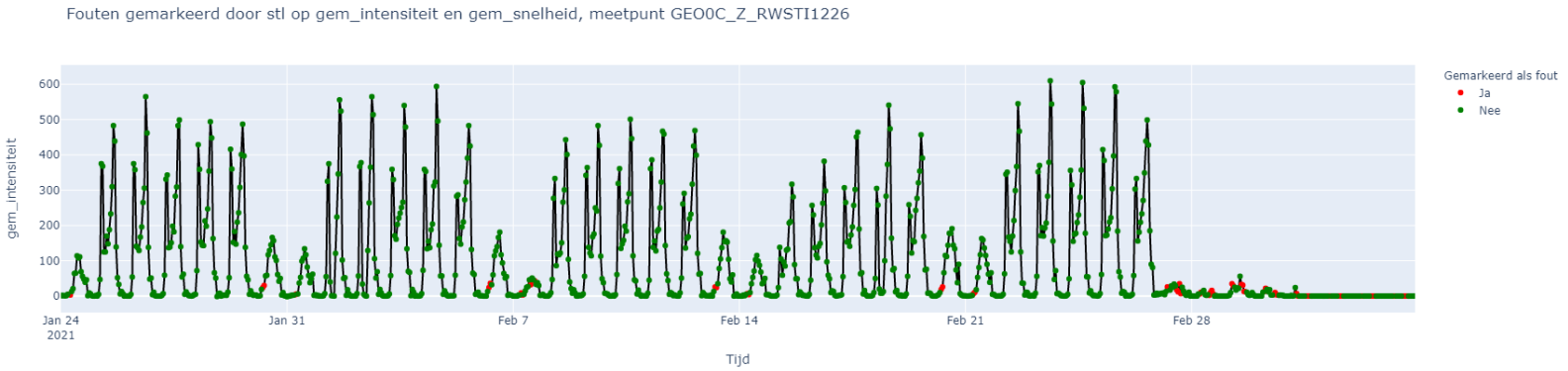
8.3.5.5.1.1 Defecte meetpunten STL en toets

In afbeelding 17 is de STL en toets methode getest op een defect meetpunt. Dit meetpunt is defect gemarkeerd op 12 februari. De verwachting is dat deze voor 12 februari geen fouten markeert en rond 12 februari begint te markeren. Dit is niet het geval. Het markeert vooral weekenden als fout. De STL-parameters en de toets strengheid kan niet zo aangepast worden dat het alleen een error geeft na 12 februari.



Afbeelding 17: STL + toets toegepast op een defect meetpunt

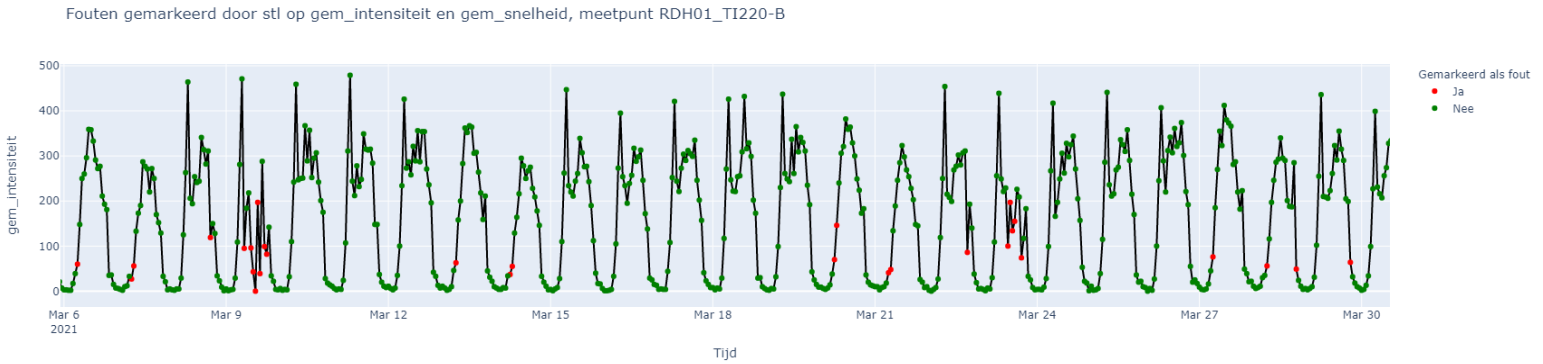
Afbeelding 18 is weer de STL en toets validatiemethode getest op een defect meetpunt. Na 26 februari vindt het defect plaats. De STL-methode pikt hier de foute periode op. Om de fout te vinden moesten de STL-parameters wel veranderd worden. Hier werd de trend vastgezet. Ook hier worden er in de niet defecte periode in het weekend punten foutief gemarkeerd.



Afbeelding 18: STL + uitschieter toets uitslag op een meetpunt net voor het defect

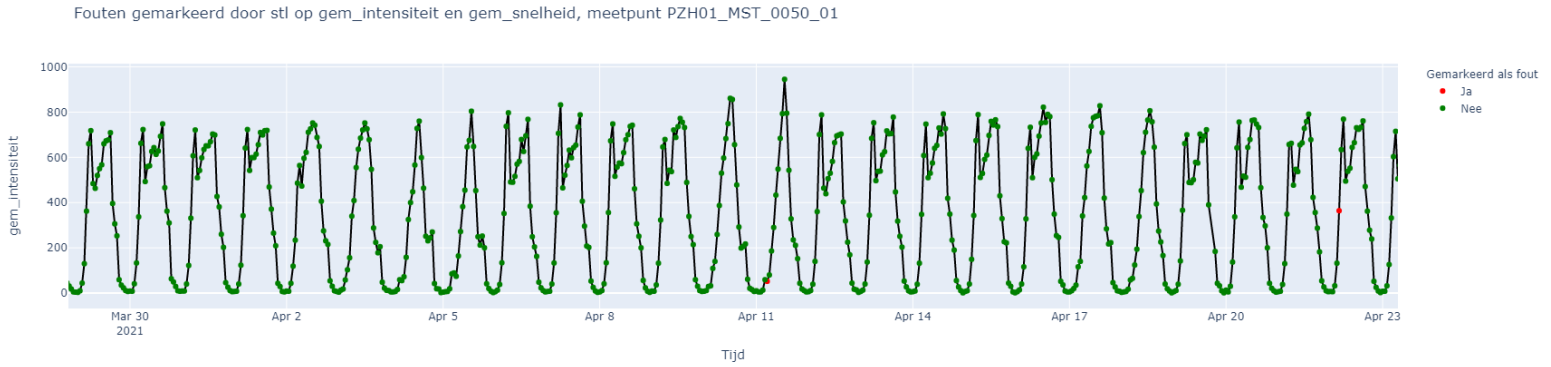
8.3.5.5.1.2 Willekeurig meetpunt STL+toets

Afbeelding 19 toont de werking van de STL en toets methode op een willekeurig meetpunt. Rond 10 maart en 23 maart is er een ongewoon signaal. STL markeert deze ongewone punten als fout. 6, 7, 13, 14 maart vallen in het weekend. Hier worden ook in het weekend fouten gemarkeerd.



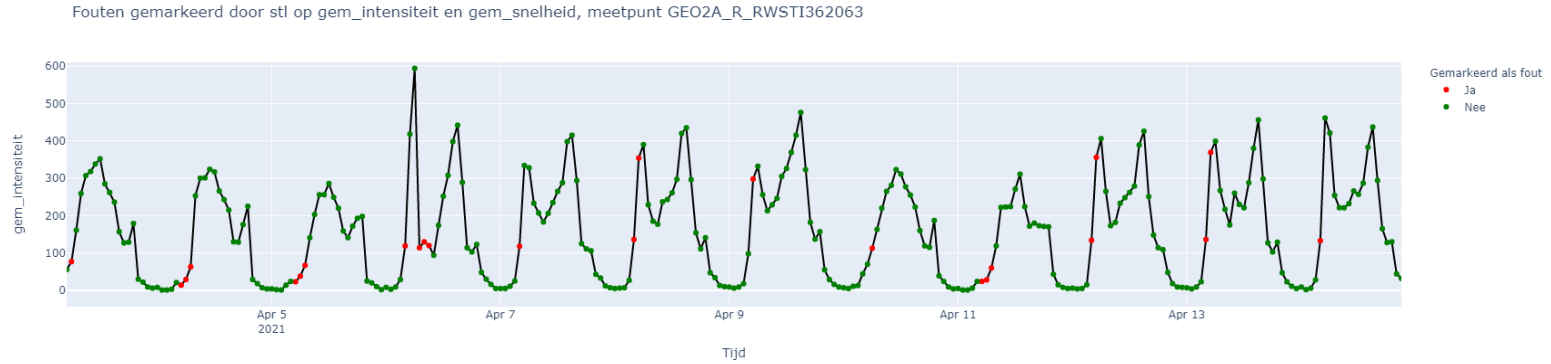
Afbeelding 19: STL + uitschieter toets uitslag willekeurig meetpunt

Afbeelding 20 is een andere willekeurige meetlocatie. Hier doet STL het wel goed. Het geeft weinig foutieve punten aan. Visueel zijn er ook geen afwijkingen te zien in het signaal.



Afbeelding 20: STL + uitschieter toets uitslag willekeurig meetpunt

Afbeelding 21 is een ander voorbeeld. 4, 5, 10, 11 april vallen allemaal in het weekend en hier worden fouten gemarkeerd. Door de weeks worden er op hetzelfde tijdstip ook steeds foutief gemarkeerd. Wel pikt het een ongewoon signaal op, op 6 april.



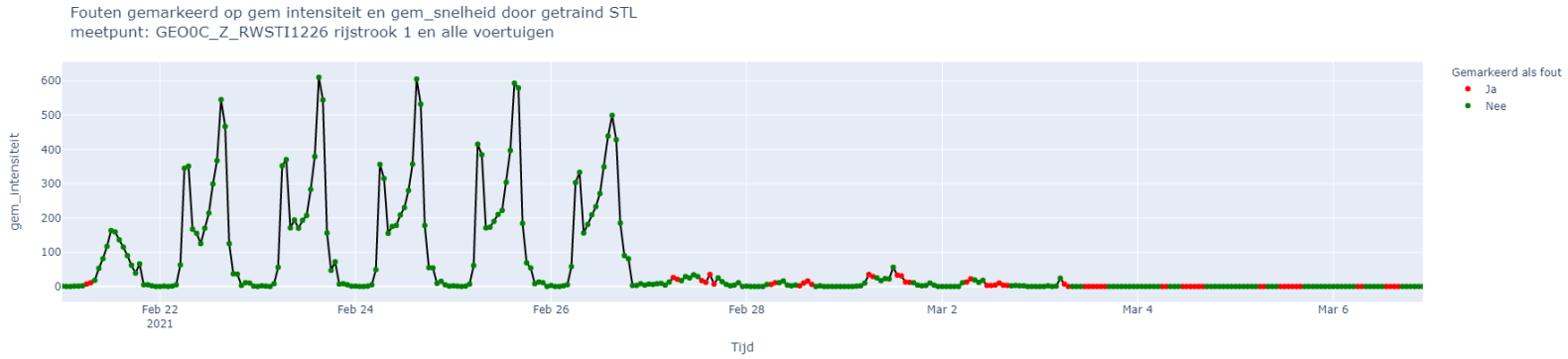
Afbeelding 21: STL + uitschieter toets uitslag willekeurig meetpunt

8.3.5.5.2 Resultaten voorspellend model en afstand methode

Er zijn ook observaties voor het voorspellend model en afstand methode. De resultaten van deze methoden zijn gecreëerd met het STL voorspellend model(STLVM). De STLVM methoden van voorspellen is nauwkeuriger dan de ARIMA-methode van voorspellen op data dat het nog niet gezien heeft. Het was dus beter in staat om het algemene patroon van de data op te leren. Hierom is alleen het STLVM model gebruikt om de datavalidatie toe te passen op de volgend resultaten.

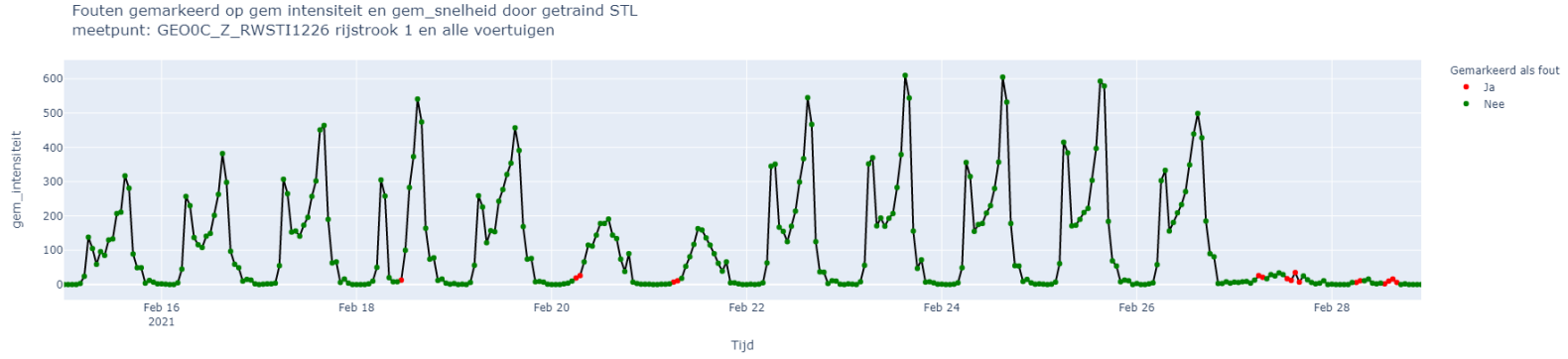
8.3.5.5.2.1 Defecte meetpunten

Afbeelding 22 toont de datavalidatie van het voorspellend model en afstand methode. De defecte periode is wel gemarkeerd als fout. Er is een wisseling tussen groen en fout in de defecte periode. Er rijden minder auto's in de nacht. Dat het defecte signaal plat ligt komt overeen met het rijgedrag op de weg. Hierom markeert deze methode punten in de defecte periode als juist.



Afbeelding 22: STLVM en afstand toets uitslag defect meetpunt

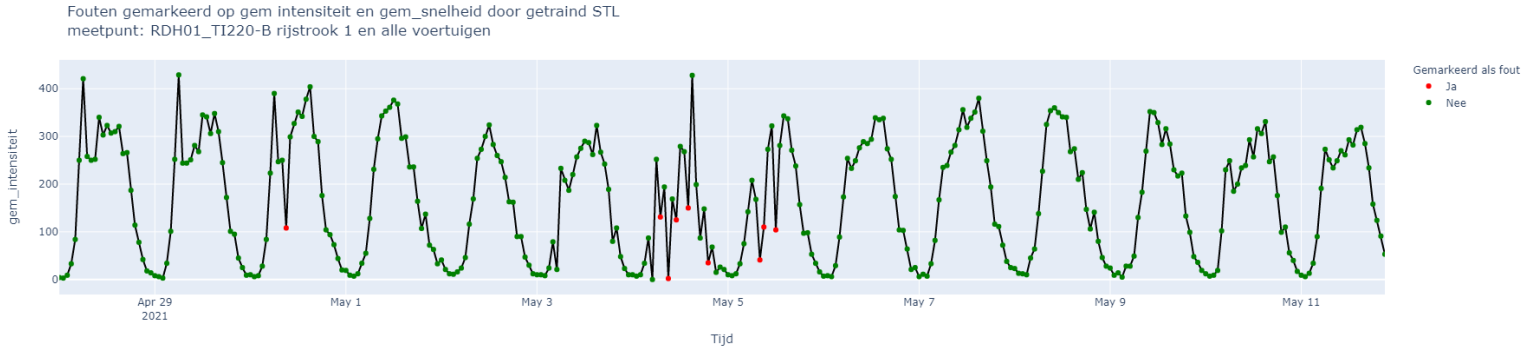
Afbeelding 23 is weer een test van deze methode op een defect meetpunt. Het meetpunt ging defect rond 28 februari. Het wordt opgepikt door deze methode. Rond 18/19 februari was er ook een raar patroon, welke deze methode heeft gedetecteerd. Rond 20 en 21 februari heeft deze methode ook een fout gedetecteerd, maar dit lijkt niet een fout te zijn. Dit is weer het weekend, waar STL een probleem mee heeft.



Afbeelding 23: STLVM en uitslag uitslag op defect meetpunt

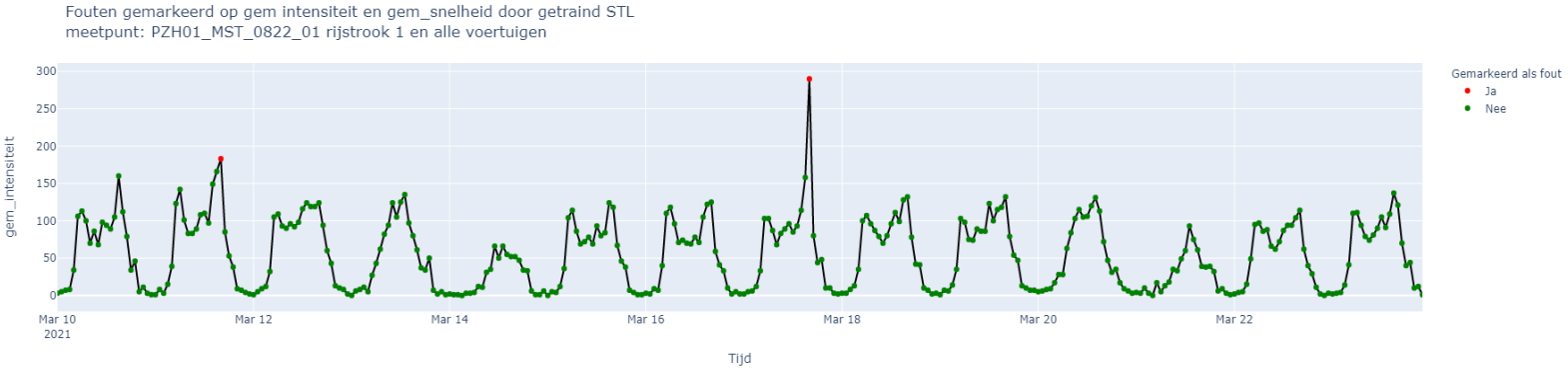
8.3.5.5.2.2 Willekeurige meetpunten

Afbeelding 24 is de voorspellend model en afstand methode toegepast op een willekeurig meetpunt. Op 4 mei en 6 mei was er een ongewoon signaal. Dit heeft deze methode opgepikt. Het is niet duidelijk of dit een defect is van het meetpunt of dat het ongewone verkeersomstandigheden zijn.



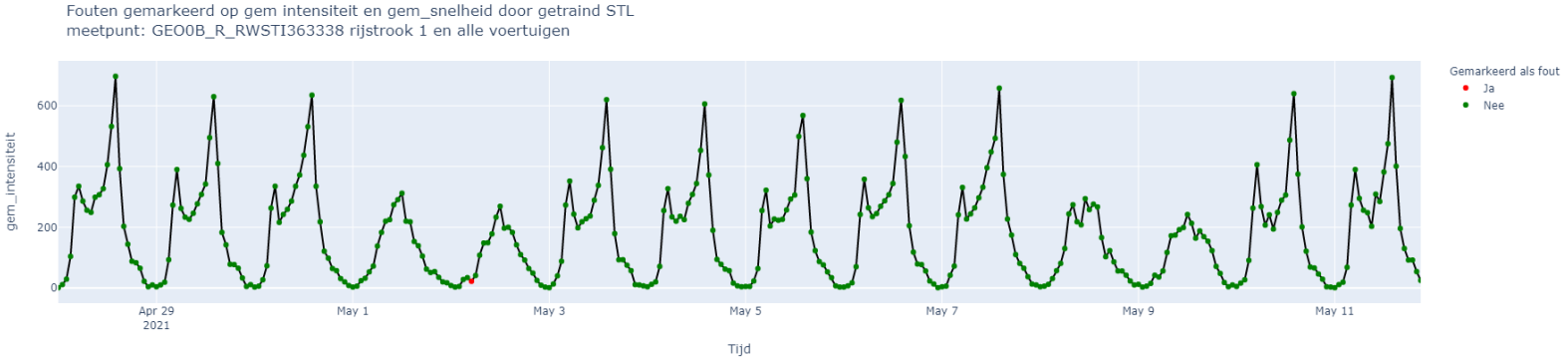
Afbeelding 24: STLVM en afstand toets uitslag willekeurig meetpunt

Afbeelding 25 is een ander voorbeeld waarbij deze methode is uitgevoerd op een willekeurig meetpunt. Hier was er op 17 maart een file. Deze heeft deze methode opgepakt.



Afbeelding 25: STLVM en afstand uitslag willekeurig meetpunt

Afbeelding 37 is deze methode weer toegepast op een willekeurig meetpunt. Dit lijkt op een normaal signaal maar er is 1 fout gedetecteerd rond 2 mei.



Afbeelding 26: STLVM en afstand uitslag op willekeurig meetpunt

8.3.6 Conclusie hoofdvraag

De NDW data is tijdreeksdata. Er zijn validatiemethoden om in tijdreeksdata afwijkingen te vinden:

- STL en toets
- Voorspellend model en afstand

Deze twee methoden werken beter op data met een sterk terugkerend patroon. De NDW verkeersdata heeft een sterk terugkerend patroon.

Uit de resultaten blijkt dat STL en toets methode veel vals positieve resultaten teruggeeft. Hierdoor zijn echte foutieve punten niet te onderscheiden van de vals positieve. STL en toets methode heeft ook veel parameters die de gebruiker kan veranderen. Voor elk signaal kan er een aparte set aan parameters gevonden worden om het resultaat te optimaliseren. Dit is niet gewenst. NDW monitort 5000 meetpunten. Er kan niet voor elk meetpunt een apart set aan parameters gevonden worden. Als er één set aan parameters gevonden kan worden voor de STL en toets methode dan is deze te gebruiken. Dit is niet gevonden in dit project en daarom ook niet aan te raden om te implementeren.

Het voorspellend model en afstand methode is goed in staat om ongewone situaties in de data te detecteren. Dit zijn niet zozeer technische fouten. Er moet een manier gevonden worden om de technische fouten van de ongewone situaties te scheiden. In de huidige staat is het wel een goede methode om ongewone situaties te detecteren.

Het voorspellend model en afstand methode is potentieel een oplossing die in staat is om op etmaalniveau te bepalen of een meetpunt defect is of niet. Het kan nu ongewone situaties vinden in uurdata. Er is nog werk vereist om dit werkend te krijgen om defecte meetpunten te vinden op etmaalniveau.

Dit is een conclusie getrokken door een iemand zonder verkeersexpertise. Een verkeersexpert kan deze resultaten bestuderen en zijn eigen conclusie trekken. Ook kan de verkeersexpert zelf experimenteren met de gebouwde POC. Hij/Zij is ook in staat om voor een langere periode de validatiemethode van dit onderzoek te monitoren op live verkeersdata.

9 Proof of concept

NDW heeft geen specifieke eisen voor de POC. Het heeft alleen eisen opgesteld voor de ML oplossing. Hierom was het voor een lange periode niet helder hoe het eindproduct eruit zou zien. Er is er besloten om een dashboard te bouwen. Het dashboard kan recente data binnenhalen van een NDW database. Hier kan het dashboard de methode van het onderzoek(STL en toets, Voorspellend model en afstand) toepassen op actuele verkeersdata.

Het dashboard is gekozen omdat er blijkt dat er verkeer expertise nodig is om de resultaten van de experimenten te beoordelen. Het is voor een onwetend persoon niet meteen duidelijk of bijvoorbeeld iets een verkeerskundige situatie iets of een technische fout. Om het makkelijk te maken voor een expert om zelf resultaten te genereren is er gekozen voor een dashboard. De verkeersexpert kan zelf nu de onderzochte validatiemethode toepassen op actuele verkeersdata met een druk op de knop. Het is dus een tool voor de verkeersexpert.

Deze POC zal uiteindelijk NDW helpen met het beslissen of de onderzochte validatiemethode het waard zijn om te implementeren. Het doel van dit onderzoek is om te onderzoeken of er methoden zijn die technisch foute data kunnen markeren. De beste POC is iets dat helpt met het beantwoorden van deze vraag, wat in dit geval het dashboard is.

NDW kan zelf experimenten uitvoeren zonder dat het zelf iets hoeft te implementeren. Het dashboard heeft verschillende opties waardoor de methode aangepast kunnen worden. Doordat een dashboard ook grafieken creëert kunnen deze weer gebruikt worden in bijvoorbeeld presentaties. Hiernaast kan NDW ook de methoden voor een lange tijd testen, waardoor ze ook door hebben hoe deze methoden presteren over een langere periode.

Naast dat een dashboard een goed POC is voor de NDW, heeft het ook geholpen met het beantwoorden van de hoofdvraag van het onderzoek in dit verslag. De eindresultaten zijn gecreëerd met de POC.

Het implementeren van een dashboard als POC was overlegd met de opdrachtgever en het is goedgekeurd.(Bijlage gesprek: 9-4-2021 Willem van Loon).

Het originele plan was om ook MLOps te implementeren in de POC. MLOps past niet in dit dashboard. MLOps is een ook heel project op zich en is niet zomaar geïmplementeerd. Hierom is het besloten om advies over de implementatie te geven en is het niet geïmplementeerd.

9.1 Requirements

Dit zijn de requirements van de ML methoden en het dashboard. De hoogste prioriteit is om te onderzoeken of deze methoden werken. Hierna kunnen aspecten zoals kosten en efficiëntie onderzocht worden (zie requirement 1.B02).

De doelen zoals “het model moet leren van de fouten” en “waarom een punt fout is gemarkeerd” zijn lager geprioriteerd. Dit zijn aspecten waar de validatiemethoden onderzocht in dit project minder goed in zijn. Zie 16 Discussie voor een uitgebreidere uitleg.

Termen
B = Business requirement
U = User requirement
TF = Technisch functioneel requirement
TNF = Technisch niet functioneel requirement
Gebruiker = eindgebruiker van de data
Leverancier = leverancier van de data
Beheerder = Persoon die uiteindelijk bepaalt of een punt fout is of niet
Methoden = De twee methoden onderzocht in dit project

Bronnen:
Opdrachtomschrijving (NDW, 2019)
Persoonlijke communicatie 8 maart 2021 Arjen Wassink
Gesprek over requirements en ontwerp (Bijlage gesprek: 19-3-2021 Willem van Loon, Arjen Wassink)

9.1.1 Requirements ML methode

Dit zijn de requirements voor de ML methode. Deze zijn afgeleid van de opgestelde doelen in 7 Doel en hierna bevestigd met de opdrachtgever(Bijlage gesprek: 19-3-2021 Willem van Loon, Arjen Wassink).

ID	Requirement	MoS CoW	Kwaliteitse igenschap
1.B01	NDW wil onderzoek doen naar ML oplossingen voor het valideren van data, om de huidige data validatie methoden te innoveren.	M	
1.B02	NDW wil een ML oplossing die zelf data validatie regels kan aanmaken omdat zelf schrijven van data validatie methoden duur is.	C	
1.B03	NDW wil een ML oplossing die zelf data validatie regels kan aanmaken zodat NDW niet complexe validatieregels hoeft te schrijven.	M	
1.B04	NDW wil een ML oplossing die zelf data validatie regels kan aanmaken zodat NDW ook fouten kan detecteren die niet door een mens gevonden kunnen worden.	M	
1.B05	NDW wil een ML oplossing die in staat is om fouten te detecteren zonder dat er op voorhand bekend is wat er fout is zodat NDW niet zelf validatieregels hoeft te schrijven.	M	
1.B06	NDW wil onderzoeken hoe MLOps geïmplementeerd kan worden in de praktijk zodat NDW ook een MLOps systeem kan opzetten	M	
1.U01	De leveranciers en gebruiker moeten met de informatie verschaft door de methoden kunnen bepalen, welke meetpunten niet werken op	M	

	etmaalniveau.		
1.U02	De leveranciers en gebruiker moeten met de informatie verschaft door de methoden kunnen bepalen, welke datapunten foutief zijn op minuutniveau.	C	
1.U03	De leveranciers en gebruiker moeten met de informatie verschaft door de methoden kunnen bepalen waarom punten fout zijn gemarkeerd.	S	
1.U04	De beheerder moet met de informatie verschaft door de methoden kunnen bepalen welke punten niet zeker zijn voor markering.	S	
1.TF01	Het systeem moet met een ML methoden data valideren.	M	
1.TF02	Het systeem moet defecte meetpunten op etmaal-niveau kunnen aangeven in de data.	M	
1.TF03	Het systeem moet foutieve datapunten op minuut-niveau kunnen aangeven in de data.	C	
1.TF04	Het systeem moet een onderscheid kunnen maken tussen zekere markeringen en onzekere markeringen.	S	
1.TF05	Het systeem moet het soort fout kunnen markeren.	S	
1.TF06	Het systeem moet kunnen leren van de nieuw gevonden fouten zodat het deze in de toekomst beter kan identificeren.	S	
1.TNF01	Het model moet binnen 1 uur 24 uur data kunnen valideren (Het moet aanzienlijk minder zijn dan een dag).	M	Time behaviour
1.TNF02	Het model moet andere fouten dan de validatieregels kunnen detecteren.	M	Functional correctness
1.TNF03	Het model moet in staat zijn om 24 uur data te verwerken.	M	Capacity

9.1.2 Requirements dashboard

Dit zijn de requirements voor het dashboard. Het implementeert de validatiemethoden beschreven in het onderzoek. Hiernaast kan de gebruiker data ophalen uit een live database. Ook wou de opdrachtgever dat de verkeersdata gevisualiseerd kon worden.

ID	**Requirement**	**MoS CoW**	**Kwaliteitse igenschap**
2.U01	De gebruiker moet data kunnen kiezen waarop deze datavalidatie wilt uitvoeren	M	
2.U02	De gebruiker moet de gekozen data kunnen visualiseren.	M	
2.U03	De gebruiker moet datavalidatie kunnen uitvoeren op de gekozen data, met STL + uitschieter toetsen	M	
2.U04	De gebruiker moet datavalidatie kunnen uitvoeren op de gekozen data, met STL voorspellend model + afstand tussen gemeten en voorspeld punt	M	

2.U05	De gebruiker moet datavalidatie kunnen uitvoeren op de gekozen data, met ARIMA model + afstand tussen gemeten en voorspeld punt	M	
2.TF01	Het systeem moet in staat zijn om de gebruiker meetpunten te laten kiezen waarop data validatie wordt uitgevoerd	M	
2.TF02	Het systeem moet in staat zijn om de gebruiker datum van data te laten kiezen waarop data validatie wordt uitgevoerd	M	
2.TF03	Het systeem moet in staat zijn om de gebruiker tijdsniveau (uur of minuut data) te laten kiezen waarop data validatie wordt uitgevoerd	M	
2.TF04	Het systeem moet in staat zijn om de gebruiker STL + uitschieter toets datavalidatie te laten uitvoeren op de gekozen data	M	
2.TF05	Het systeem moet in staat zijn om de gebruiker STL + uitschieter toets datavalidatie parameters aan te laten passen	M	
2.TF06	Het systeem moet in staat zijn om de gebruiker STL voorspellend model datavalidatie te laten uitvoeren op de gekozen data	M	
2.TF07	Het systeem moet in staat zijn om de gebruiker STL voorspellend model datavalidatie parameters aan te laten passen	M	
2.TF08	Het systeem moet in staat zijn om de gebruiker ARIMA-datavalidatie te laten uitvoeren op de gekozen data	M	
2.TF09	Het systeem moet in staat zijn om de gebruiker ARIMA-datavalidatie parameters aan te laten passen	M	
2.TF10	Het systeem moet in staat zijn om data te visualiseren	M	
2.TF11	Het systeem moet in staat zijn om meerdere meetpunten tegelijk te visualiseren (20 Bijlage gesprek: 15-4-2021 Willem van Loon)	M	
2.TF12	Het systeem moet data kunnen ophalen uit een database	M	
2.TF13	Het systeem moet de performance (runtime, memory) van de datavalidatie methoden kunnen tonen	M	
2.TF14	Het systeem moet de resultaten van STL-datavalidatie kunnen tonen	M	
2.TF15	Het systeem moet de resultaten van STL voorspellend model datavalidatie kunnen tonen	M	
2.TF16	Het systeem moet de resultaten van ARIMA-datavalidatie kunnen tonen	M	
2.TF17	Het systeem moet in staat zijn om de gebruiker willekeurig meetpunten te laten kiezen	M	
2.TNF1	Het systeem moet de database Azure Data Lake store kunnen benaderen	M	Interoperability

9.2 Functioneel ontwerp POC

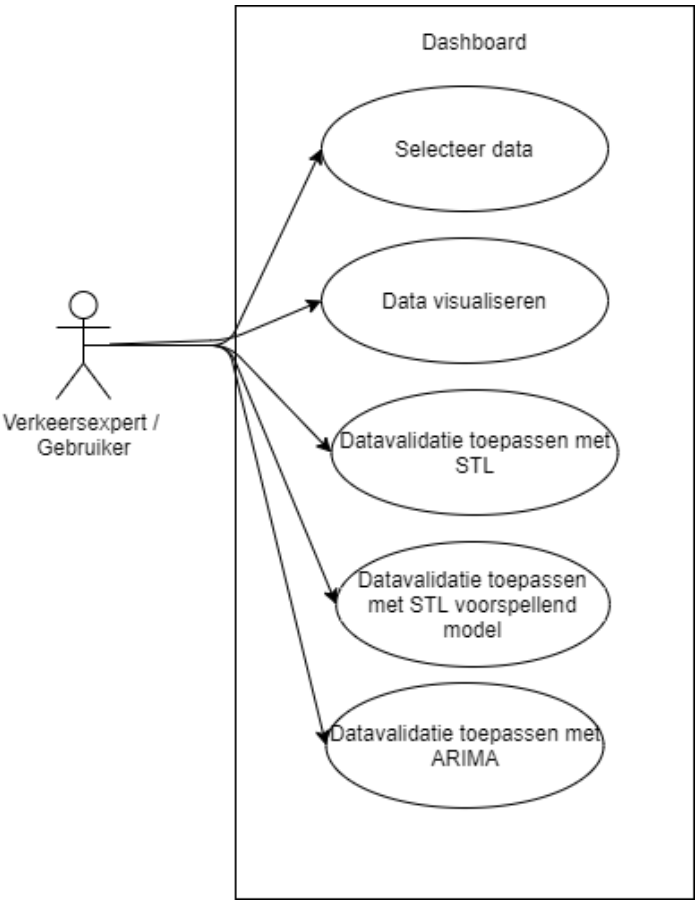
Het dashboard heeft de volgende use-cases:

Dataselectie
De gebruiker moet in staat om zelf data te selecteren. De gebruiker kan meetpunten, begin- en einddatum van de data selecteren.

Data visualiseren
Dit was een wens van de opdrachtgever. De data geselecteerd door de gebruiker kan gevisualiseerd worden zodat de gebruiker een idee kan krijgen hoe de data eruit ziet.

Datavalidatie
Datavalidatie met de verschillende methoden onderzocht in dit project:

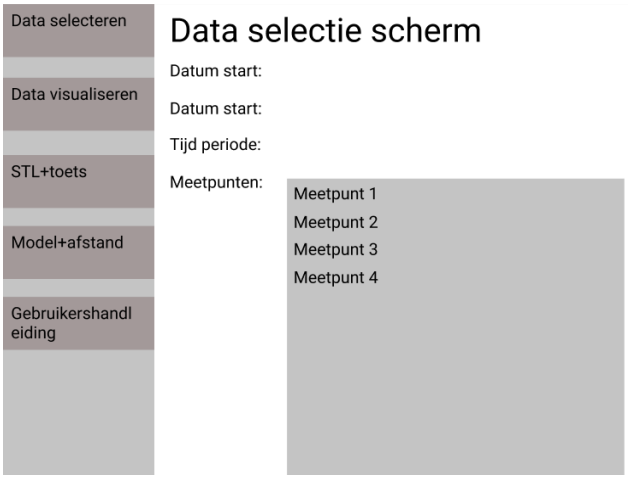
- STL en verschillende toetsen. De gebruiker kan ook parameters aanpassen(zoals strengheid toetsen)
- ARIMA en afstand
- STL voorspellend model(STLVM) en afstand



Afbeelding 27: Use case diagram dashboard

9.2.1 Mockups dashboard

Het plan is om een dashboard te maken met verschillende schermen, waar tussen gewisseld kan worden door middel van tabbladen. De tabbladen zijn zo geordend dat het de workflow van de gebruiker weergeeft. Het begint dus met dataselectie en visualisatie. Hierna komen de data validatie methoden. Deze mockups zijn screenshots van een figma prototype.



Afbeelding 28: Mock up dataselectiescherm

Afbeelding 29 : mock up STL+toets scherm

9.3 Dashboard

Dit hoofdstuk beschrijft de implementatiedetails en validatie van de POC.

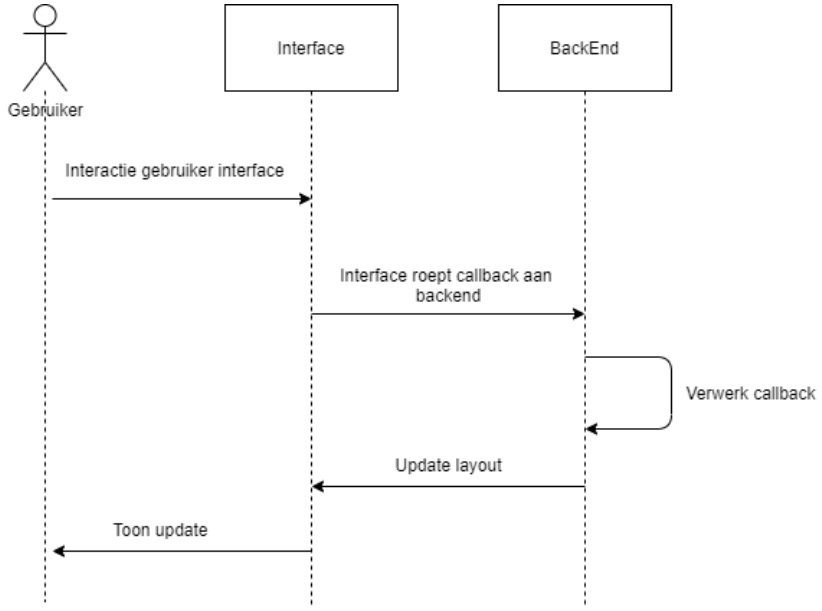
9.3.1 Technisch ontwerp

9.3.1.1 Dash

Het dashboard is gebouwd in Python met het dashboarding framework Dash. Het framework combineert de Python flask server met React.

Er zijn andere dashboarding tools die gebruikt kunnen worden, zoals Streamlit en Bokeh. De grootste reden om voor Dash te kiezen is omdat Dash de grootste gemeenschap op internet heeft. Hierdoor kunnen oplossingen voor obstakels tijdens programmeren makkelijker opgezocht worden. Hiernaast geeft Dash de mogelijkheid om het dashboard meer aan te passen door de gebruiker ook de css te laten veranderen. Dash is ook efficiënt. Dash berekent en past alleen delen van de layout aan welke geupdate moeten worden. Streamlit berekent en past alle elementen op de pagina opnieuw aan. Doordat Dash het meest aanpasbaar is, vergt het ook het meeste programmeerwerk, wat weer voldoet aan de HBO-i competenties (Plotly Dash, n.d.).

Het idee achter dit framework is dat er een layout en callbacks bestaan. De layout is hoe de applicatie in de browser eruit ziet en de interface waar de gebruiker mee interacteert. Wanneer de gebruiker met een layout element interacteert worden callback functies aangeroepen in de backend. De backend reageert en update de layout aan de hand van de interactie.



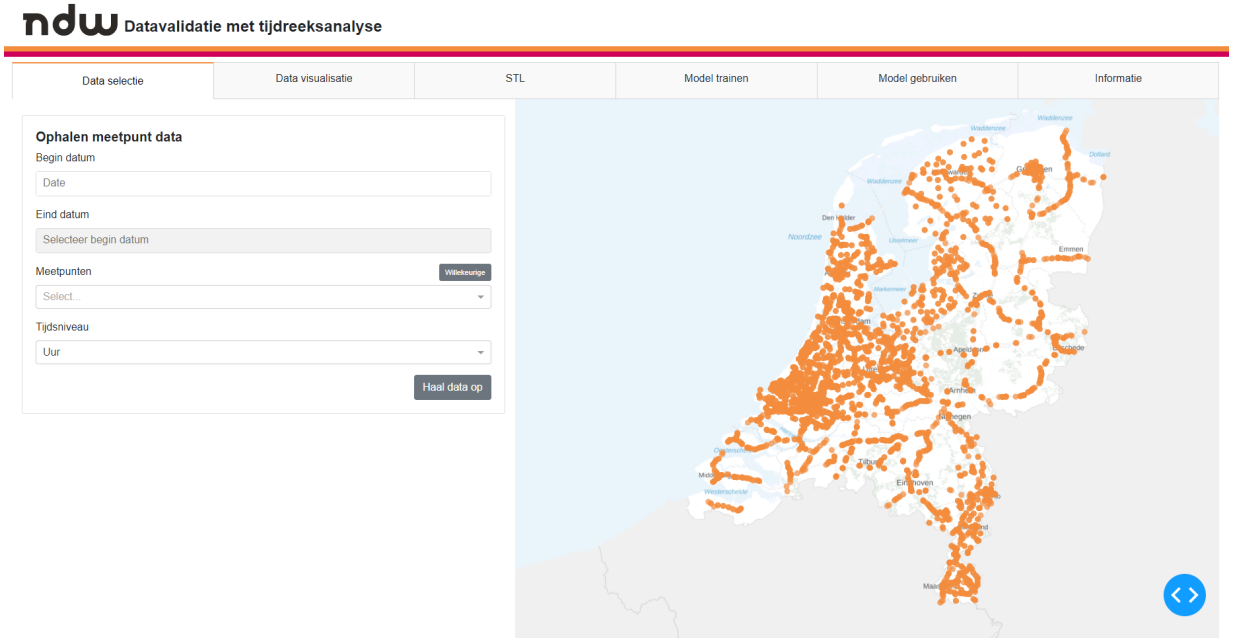
Afbeelding 30: Voorbeeld interactie interface en backend.

9.3.1.2 Layout

Afbeelding 31 toont een pagina van het geïmplementeerd dashboard. Dash is gebouwd in Python. De opbouw van de HTML paginas wordt ook geschreven in Python. De input velden

zijn React componenten die het framework zelf implementeert. Deze React componenten hebben verschillende eigenschappen, waarnaar geluisterd kan worden. Wanneer een eigenschap verandert, wordt de er een callback aangeroepen die naar dit eigenschap luistert.

De structuur van de paginas/tabbladen van het dashboard zijn aangepast met css.



Afbeelding 31: Pagina van het dashboard

9.3.1.3 Backend

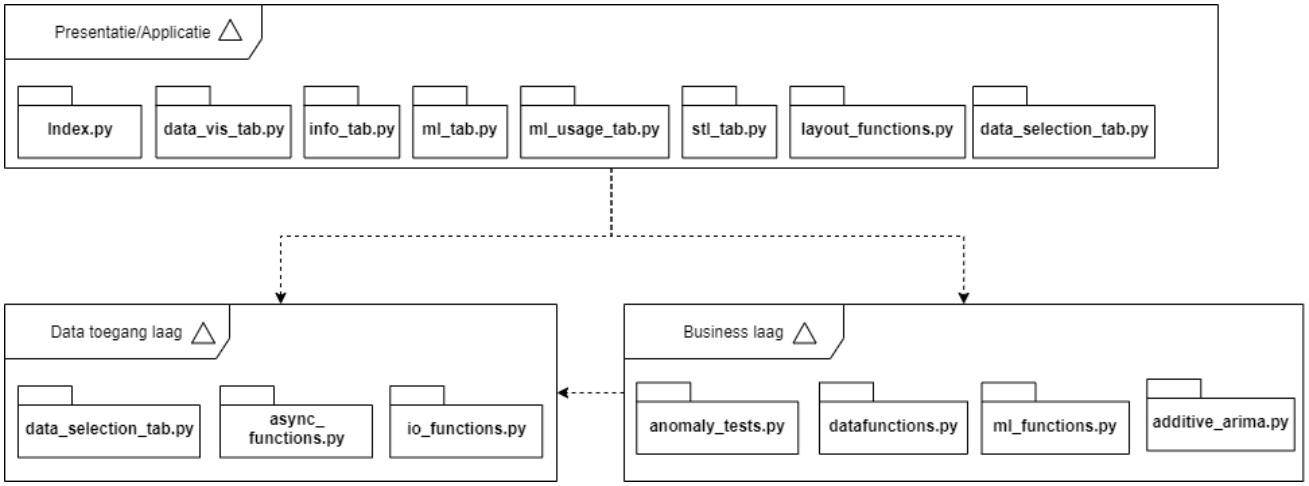
Er zijn verschillende modules die functies bevatten die verschillende taken verrichten. De modules zijn:

- anomaly_tests.py: Deze module bevat de geimplementeerde statisctische testen voor de STL+toets methode.
- async_functions.py: Bevat functies om delen concurrent te laten verlopen.
- datafunctions.py: Bevat functies om data te verwerken en te transformeren.
- io_functions.py: Bevat functie om data op te halen en weg te schrijven.
- layout_functions.py: Bevat een aantal helper functies die gebruikt kunnen worden door de layout. Dit zijn functies die op meerdere tabbladen gebruikt worden.
- ml_functions.py: Bevat functies om modellen te trainen.

De applicatie heeft ook meerdere tabblad modules. Deze modules bevatten de layout van verschillende tabbladen en de callbacks die luisteren naar veranderingen in de layout.

Het ontwerp van het dashboard is een n tier architectuur waarbij de lagen bestaan uit de.:

- Presentatielaag
- Applicatielaag
- Business laag
- Data toegang laag



Afbeelding 32: Modules van het dashboard

De presentatielaag is de laag die aan de gebruiker getoond wordt. Dit is de layout die in de browser wordt getoond. De applicatielaag bevat de callbacks die worden aangeroepen wanneer de gebruiker interacteert met de interface. De businesslaag is waar de dataverwerking plaatsvindt. Hier worden ook de ML modellen getraind. De data toegang laag handelt alles waarbij data wordt opgehaald en weggeschreven.

9.3.1.4 Modellen trainen

Er worden drie soorten modellen getraind:

- ARIMA voorspellend model
- STL-model
- STL voorspellend model

Het ARIMA model en STL model worden getraind met behulp van de Python library *statsmodels*. Deze library traint een ML model die het signaal zo goed mogelijk probeert te volgen. Het STL voorspellend model is zelf geïmplementeerd. Dit model gebruikt de *statsmodels* STL om een voorspellend model te maken.

9.3.1.5 Programmeer wijze

De applicatie bestaat grotendeels uit functies en het is niet gefocust op objectgeoriënteerd programmeren. De tabblad modules hebben callbacks die de vorm aannemen van een functie. De callbacks roepen weer andere functies aan in andere modules om het gewenste resultaat te creëren.

De informatie-uitwisseling tussen functies gaat grotendeels met behulp van dataframes. Dataframes zijn tabellen. Dataframes in dit project hebben de kolommen voor de gemiddelde intensiteit en gemiddelde snelheid van verschillende tijden. *Pandas* is een library die efficiënt kan omgaan met dataframes. Hierom is er gekozen om de data overdracht plaats te laten vinden met dataframes.

De klassen die voor deze applicatie gecreëerd zijn, zijn als volgt:

- STLModel : Dit is een implementatie van het STL voorspellend model. Dit model kan gebruikt worden om de standaardwaarde van een meetpunt over een week te leren. Hierna kan deze ook voorspellingen maken. Deze klas bevat een interface dat de gebruiker het model laat trainen en voorspelling kan laten doen.
- MetaData: Er wordt metadata opgeslagen over de data die is opgehaald van de meetpunten. Om de data makkelijk over te dragen is deze klas geïmplementeerd.
- ModelMetaData : Er wordt metadata opgeslagen over de modellen die zijn getraind. Deze klas is er om de overdracht van die data tussen functies makkelijker te maken.
- ARIMAException, STLModelException: Dit zijn exceptions die gegooid kunnen worden wanneer er iets fout gaat tijdens het trainen. Deze foutmeldingen worden opgevangen en er wordt een bericht aan de gebruiker getoond met informatie hoe het deze error kan oplossen.

9.3.1.6 Data ophalen en opslaan

De verkeersdata wordt opgehaald van een Azure Data Lake. Azure Data Lake is een database om gegevens van elke omvang en vorm op te slaan(Microsoft Azure, n.d.). Het downloaden van de data uit de Azure Data Lake duurt erg lang. Hierom wordt de Data Lake opgehaalde data lokaal opgeslagen, om vervolgens gebruikt te worden in het dashboard.

De data wordt opgeslagen in de map waar het dashboard is gestart. Daar maakt het dashboard een mapje aan genaamd “data_dir”. Hier wordt ook metadata over de opgehaalde data opgeslagen en de modellen die getraind worden.

Omdat het ophalen van de data veel tijd kost, is besloten om dit op een andere thread uit te voeren. Op deze manier loopt de backend niet vast terwijl het de data ophaalt.

Om toegang te krijgen tot de Azure Data Lake is een API key nodig. De API key is afkomstig van de NDW. De API key bevindt zich in de map genaamd *data_toegang* met een aantal libraries en functies die het mogelijk maken om te verbinden met de Azure Data Lake. Deze functies zijn geschreven en aangeleverd door NDW zelf.

Als deze NDW library met een geldige API key niet bestaat in de dashboard map, dan zal het dashboard draaien in demo modus. Hier haalt het dashboard niet meer de data op van de Azure Data Lake maar van een lokale bron. De lokale bron is de map genaamd *local_data* in de map van het dashboard. Deze data is data van de 36 meetpunten gebruikt in het experimenteren.

9.3.1.7 Productieomgeving opzet

Dit dashboard is gebouwd om lokaal op te zetten. In de huidige toestand zou het niet centraal op een server opgezet kunnen worden. De data van de gebruiker wordt altijd op dezelfde plek opgeslagen. Dit betekent dat voor meerdere gebruikers dezelfde data steeds wordt overgeschreven.

Als iemand deze applicatie centraal zou willen opzetten op een centrale server dan moeten gebruikers geauthenticeerd worden. Dit kan met een log in(wat mogelijk is in Dash) maar het kan ook met een simpele gebruikers id zijn.

Met deze authenticatie kunnen er voor verschillende gebruikers verschillende data opgeslagen worden. Dit kunnen verschillende mappen zijn op de harde schijf of zelfs een database.

9.3.2 Testen

Validatie dat de code correct werkt, is uitgevoerd door automatische testen te schrijven. Dit zijn unittesten van de functies en klassen gebruikt in dit dashboard. Hiernaast zijn er ook een aantal mock klassen gemaakt voor verschillende unittesten.

Sommige functies in de *io_functions.py* zijn niet automatisch getest. Deze bevat functies die bestanden wegschrijven en lezen op een specifieke locatie. Wanneer deze getest worden, worden de bestanden overschreven wat niet gewenst is.

Sommige functies in “datafunctions.py” zijn ook niet getest door hetzelfde probleem. Dit zijn vooral functies waarbij meerdere andere kleinere functies achter elkaar worden aangeroepen. Deze kleinere functies voeren een handeling uit. Deze zijn wel getest.

De async methoden zijn ook niet getest. Dit zijn korte functies die een async proces starten waarbij een methoden worden aangeroepen. Deze aangeroepen methoden zijn wel getest.

De layout van de Dash applicatie is niet live getest. Dash heeft het testen van de layout nog niet goed geïmplementeerd. De huidige stand van zaken is dat de gebruiker de applicatie moet herschrijven in een unittest om deze live te kunnen testen. Dit is niet gewenst, want hierdoor wordt code herhaald en wordt ook niet echt getest of de applicatie code werkt. Er is wel getest of verschillende callbacks de juiste layout genereren/teruggeven. De callbacks zijn wel te testen met unit tests (Dash plotly, n.d.).

De coverage van de unittest valt uiteindelijk op 82%, wat boven de eis van Quintor uitvalt namelijk 80%.

```
tests\test_9_layout_functions.py         54      4      6      1    92%   23, 28, 44, 85
-------------------------------------------------------------------------------------
TOTAL                                  2637    417    266     10    82%


Results (335.58s):
      93 passed
      45 skipped
```

Afbeelding 33: Coverage automatisch testen

9.3.3 Code kwaliteit

De geschreven code moet een bepaalde kwaliteit hebben zodat het leesbaar is. Hierdoor is het makkelijk om te interpreteren wat er gebeurt en kan er verder aan gewerkt worden. Dit is bereikt door de code style aan te passen aan de PEP 08 standaard. Pep 08 is de style guide van Python zelf (Python, 2001).

Dit is gevalideerd met de code linter Pylint. Pylint checkt of de code in de juiste stijl is en dat het goed is gedocumenteerd. Naast de code style detecteert Pylint ook code smells. Code smells zijn indicaties in de code die duiden op een dieper probleem (Fowler, 2006).

Pylint gaf de geschreven code een 8.

```
--------------------------------------------------------------------
Your code has been rated at 8.01/10 (previous run: 8.01/10, +0.00)
```

Afbeelding 34: Pylint beoordeling code kwaliteit

9.4 Evaluatie requirements

Het dashboard heeft niet alle requirements kunnen implementeren. Van de ML model requirements zijn er 13 van de 18 geïmplementeerd. De requirement die niet geïmplementeerd zijn:

1. (1.B02) NDW wil een ML oplossing die zelf data validatie regels kan aanmaken omdat zelf schrijven van data validatie methoden duur is.
    a. Deze requirement is niet meegenomen in dit onderzoek. Hier moet onderzocht worden of het financieel waard is om deze methoden te implementeren. Om het af te kaderen is dit niet meegenomen in het onderzoek.
2. (1.U02) De leveranciers en gebruiker moeten met de informatie verschaft door de methoden kunnen bepalen welke datapunten foutief zijn op minuutniveau.
    a. Minuut data was te lastig om mee te werken en is daarom niet meegenomen in het eindresultaat.
3. (1.TF03) Het systeem moet foutieve datapunten op minuut-niveau kunnen aangeven in de data.
    a. Zie reden 1.U02
4. (1.TF06) Het systeem moet kunnen leren van de nieuw gevonden fouten zodat het deze in de toekomst beter kan identificeren.
    a. Deze methoden zijn niet in staat om te leren van de nieuw gevonden data.
5. (1.TNF01) Het model moet binnen 15 minuten 24 uur data kunnen valideren (Het moet aanzienlijk minder zijn dan een dag).
    a. Dit is niet getest, vanwege tekort aan tijd en dit was laag op de priotisering. Er is dus niet getest hoe lang de methoden erover doen om 24 uur data van alle meetpunten te valideren.

Van de dashboard requirements zijn er 22 van de 23 geïmplementeerd:

1. (T.F13) Het systeem moet de performance (runtime, memory) van de datavalidatie methoden kunnen tonen
    a. Laag op de priotering en hierom ook niet aan toegekomen.

De requirement 1.B05 onderzoek MLOps is beschreven in het adviesrapport.

9.5 Sprint proces

Om het dashboard te implementeren is Scrum gebruikt. Het dashboard is gemaakt in 3 sprints van 2 weken. Er zijn 41 userstories opgesteld waarvan er 7 niet zijn geïmplementeerd. Deze zijn niet geïmplementeerd vanwege een tekort aan tijd.

In sprint 1 was de eerste prototyping uitgevoerd dat beschreven is in het onderzoek. In sprint 2,3 en 4 is er aan de code van het dashboard gewerkt. In sprint 2 en 3 zijn geen userstories afgerond. De *how to tests* van de userstories in sprint 2 en 3 waren niet af. Hierdoor bleven veel userstories in de *needs testing* rondhangen. Sprint 4 was gefocust op het schrijven van testen. De afbeelding hieronder laat de velocity chart zien en dat er in sprint 4 veel was afgerond omdat daar de testen zijn geschreven.

Tijdens het implementeren van het dashboard was het testen naar de achtergrond gedrukt. Het implementeren van functionaliteit kreeg voorrang. De functionaliteit werd wel getest alleen waren er geen automatische testen.

Afbeelding 35: velocity chart uitgevoerde sprints

10 Conclusie

De NDW data is tijdreeksdata. Er zijn validatiemethoden om in tijdreeksdata afwijkingen te vinden:

- STL en toets
- Voorspellend model en afstand

Deze twee methoden werken beter op data met een sterk terugkerend patroon. De NDW verkeersdata heeft een sterk terugkerend patroon. Uit onderzoek blijkt dat het mogelijk is om ongewone punten te vinden in verkeersdata met het voorspellend model en afstand methode. Het is nog niet in staat om alleen technisch foute data te detecteren. Er moet nog werk in de methode gestopt worden om technisch foute data te vinden. De STL en toets methode is niet te gebruiken als validatiemethode voor verkeersdata.

Er is besloten om een dashboard als POC te maken. De resultaten moeten nog een keer geïnterpreteerd worden door een verkeersexpert. Met het dashboard kan een verkeersexpert resultaten genereren met een druk op de knop. Een expert kan de onderzochte methoden in dit onderzoek ook voor een langere tijd testen m.b.v de POC.

Uiteindelijk vond de opdrachtgever jammer dat er geen model uit het onderzoek is gekomen maar vond de geven adviezen wel helder(Gesprek Willem 2-9-2021).

11 Discussie

ML methoden valideren
Het lastige aan dit project was bepalen hoe er een conclusie getrokken moest worden aan het onderzoek. Achteraf is het duidelijk dat hier aan het begin van het project meer tijd aan besteed moest worden. Van tevoren moest er al de methode gevonden worden om de onderzochte methoden te beoordelen. Door het van tevoren op te zetten, kon er ook naar een bepaald doel toegewerkt worden.

Het is nu gedaan met interessante observaties. Het pluspunt hier is dat het visueel toont hoe de methoden interacteren met de data.

Een andere methode die er nog bij gedaan kon worden is testen met een gelabelde dataset. Deze dataset kon verkeerskundige situaties, defecte periode en correcte periode gemarkeerd hebben. Hierdoor konden er kwantitatieve resultaten gegenereerd worden. Hiermee kon getest worden hoeveel procent van de fouten eruit gehaald worden, en hoeveel procent niet. Ook kon het testen geautomatiseerd worden.

Dit zou wel geholpen hebben om een meer overtuigendere conclusie te trekken. Hiernaast konden er ook prestatie indexen opgezet worden waar de methoden aan moesten voldoen om “werkend” te worden beoordeeld.

Er was wel een overweging om een gelabelde dataset te gebruiken. Er is een bestand met defecte meetperiode en de datums waarop ze defect waren. Dit bestand kon gebruikt

worden om defecte meetpunten te labelen. Daarnaast kon door de afstudeerder verschillende verkeerskundige situaties gemarkeerd worden.

Er was wel een idee achter het niet gebruiken van een gelabelde dataset. De ML oplossingen moeten zonder dat er van tevoren bekend is wat er fout en niet fout is fouten kunnen markeren. Als er een gelabelde dataset bestaat dan zal de onderzoeker willen of onwillende de dataset toch gebruiken om het “gelabeld” te trainen. De dataset zal een zekere invloed hebben op het onderzoeksproces.

Hierom is er gekozen voor een blind onderzoek. Als het niet bekend is wat er fout en niet fout is dan kan dat ook geen invloed hebben op het onderzoeksproces. Met deze aanpak blijkt het concluderen erg lastig te zijn. Volgend project is het wel aan te raden om een gelabelde dataset te gebruiken.

STL en toets methode
In de resultaten is te zien dat de STL en toets methode erg veel vals positieve resultaten teruggeeft. Het geeft vooral veel vals positieve resultaten terug op locaties waarbij het weekend signaal veel verschilt van het weekdag signaal. Het weekend signaal een weekdag signaal zijn verschillende maar beïnvloeden elkaar, tijdens het opstellen van de STL. Dit kan opgelost worden. Er kunnen verschillende STLs gedaan worden voor de week- en weekenddagen. Hierdoor wordt er een decompositie van de twee signalen apart gedaan.

De resultaten van de methode STL en toets zijn ook te beïnvloeden door de parameters van de methode te veranderen(bijv strengheid toets). Hierdoor worden de resultaten ook net wat anders. Er moet een set aan parameters gevonden worden die werkt voor elk meetpunt. Deze is er nu niet.

Een probleem met de STL en toets methode is ook dat het STL proces beïnvloed wordt door de foutieve metingen. De STL wordt gedaan over de correcte en de defecte periode. Hierdoor wordt het defecte signaal ook meegenomen om het standaard signaal te creëren. Hierdoor lijkt de defecte periode ook normaal.

In de huidige vorm is de STL en toets methode niet te gebruiken. Het geeft nog te veel vals positieve punten. Het STL heeft problemen met het weekend maar dit kan opgelost worden. Het heeft ook verschillende parameters die kunnen veranderen, wat weer voor extra complexiteit zorgt. Ook wordt het STL proces beïnvloed door de defecte periode.

Het STL voorspellend model is een goede methoden als er weinig tijdreeksen beoordeeld moeten worden. De verschillende parameters(soort toets, strengheid toets) kunnen met de hand aangepast worden om het beste resultaat te creëren. Er zijn teveel meetpunten in deze data om het handmatig de beste set aan parameters te vinden voor elk meetpunt.

Voorspellend model en afstand methode
De methode voorspellend model en afstand kan ongewone metingen uit de data halen. Het is nog niet in staat om technisch foute data te detecteren, en hierom is het ook nog niet in staat om defecte meetpunten te detecteren.

Defecte meetpunten zullen meetpunten zijn die constant een verkeerd signaal teruggeven. In het onderzoek is dit niet getest, maar de onderzoeker kan kijken hoe lang de validatiemethode een foutief signaal teruggeeft. Als het een lange periode een foutief signaal blijft teruggeven, dan is het waarschijnlijk een defect meetpunt. Als het na een tijdje weer terugspringt naar het gewoon signaal, dan was het een verkeerskundig situatie.

Uit experimenten blijkt dat het STL voorspellend model(STLVM) het beste model is om te voorspellen. STLVM is goed in staat om het patroon van de data te herkennen. STLVM wordt getraind op correcte data en hierna gebruikt om data te voorspellen.

STLVM heeft nog steeds wel moeite met het weekend zoals te zien in afbeelding 23 en 26. De methode markeert in het weekend correcte datapunten als incorrect. Het ARIMA model heeft ook moeite met het weekend. Het probleem met STLVM en weekenden kan opgelost worden door een apart model te trainen voor het weekend en weekdagen en deze hierna samen te voegen.

In afbeelding 15 is bijvoorbeeld te zien dat het temperatuur een invloed kan hebben op het signaal. In deze afbeelding in de periode rond 7 februari is te zien dat er minder auto's reden. De temperatuur en andere variabelen kunnen meegenomen worden in de voorspelling om het voorspellen nog nauwkeuriger te maken. Hierdoor komen er minder afwijkende situaties voor in de data omdat het voorspellend model deze afwijkingen zal verwachten. Voorbeelden van overige variabelen: Evenementen, feestdagen, wegonderhoud enz.

Een probleem met deze methode is wel dat er voor elk signaal een apart model getraind moet worden, dus voor de gemiddelde snelheid en intensiteit van elk meetpunt index moet een model komen. Dit zijn op z’n minst 10.000 modellen. Dit is op dit moment de grootste reden om deze methode niet te implementeren. Er zijn vast manieren om dit probleem op te lossen bijv. het groeperen van soortgelijke meetpunten, waarbij voor elke groep een apart model getraind wordt. Dit is niet verder onderzocht in dit project.

Het is handig om een simpel modellen te trainen. STL voorspellend model traint snel. Voor dit project zijn er ook simpele ARIMA-modellen getraind, welke ook snel trainen. Een complex ARIMA model, dat misschien beter dan beide simpele modellen kan voorspellen, kan heel lang duren om te trainen. Er moet voor elk meetpunt ook een nieuwe model getraind worden. Hierom zijn simpele modellen die goed genoeg kunnen voorspellen beter dan modellen die perfect kunnen voorspellen.

Terwijl deze methode wel goed in staat is om ongewone punten te vinden is het nog niet realistisch om in te zetten. Het aantal modellen is te veel. Ook zijn verkeerskundige situaties nog niet te onderscheiden van technisch foute data. Als deze twee problemen opgelost kunnen worden dan is er zeker potentie in deze methode.

Minuut data
Het werken met minuut data is een hele andere uitdaging op zich. Alles is veel lastiger om te valideren en alle processen duren ook langer(ca. factor 16), bijv. het trainen van een model. Hierom is er voor gekozen om minuut data niet mee te nemen in de eindresultaten.

Eisen ML methode
Deze model en afstand methode is in staat om andere fouten te detecteren dan de validatieregels(bijv afbeelding 24). Andere eis van NDW was dat de methode aan moest geven waarom een punt fout is. Dat is voor deze methoden altijd hetzelfde: Het signaal wijkt af van het standaard signaal.

Een eis was ook dat het onzekere gevallen moest melden bij een beheerder. Dit kan voor beide onderzochte methode geïmplementeerd worden door op verschillende afstanden naar fouten te zoeken. In plaats van een harde limiet te zetten, kunnen er meerdere limieten worden gezet van hele erge afwijkingen tot minder erge afwijkingen.

Beide onderzochte methoden zijn niet in staat om van de nieuwe gevonden fouten te leren. Het vindt alleen afwijkende signalen van het standaard signaal. Het zal nooit meer fouten vinden dan dat het nu al vindt.

Dashboard
Het dash framework is gebruikt om het dashboard te maken. Dash is niet geschikt om een snel prototype dashboard te maken. Dash is bedoeld voor dashboards die in productie zullen draaien. De programmeur is in staat om veel in het dashboard aan te passen maar dit betekent dat er ook veel geprogrammeerd moet worden. Dit kost tijd.

Voor dit project is dat niet erg omdat de afstudeerder een competentie had waar hij moest programmeren. Voor een volgend project is het handig om een ander dashboarding tool te gebruiken om een dashboard op te zetten voor experiment resultaten, zoals Streamlit.

Predictive maintenance
De ideale situatie is wanneer defecten gedecteerd kunnen worden het meetpunt stuk gaat. Dit wordt ook wel predictive maintenance genoemd. Als het meetpunt afwijkingen vertoond voordat het defect gaat, kunnen deze afwijkingen gebruikt worden om toekomstige defecten te detecteren. Dit is iets dat gedaan kan worden met de voorspellend model en afstand methode. Het probleem is dat de meetpunten geen afwijkend signaal tonen voordat het defect plaatsvindt. De meetpunten gaan meestal plotseling kapot zoals te zien in afbeelding 9.

12 Advies

Dit hoofdstuk zal het advies beschrijven. Als eerst wordt beschreven of het wel handig is om deze validatiemethoden te in te zetten. Als de NDW nog het onderzoek door wil zetten zijn er nog een aantal punten die onderzocht kunnen worden. Er wordt ook uitgelegd hoe het systeem eruit zal zien in productie. Als laatst wordt uitgelegd hoe MLOps geïmplementeerd kan worden. Door MLOps te implementeren wordt een experimentele oplossing omgezet naar een product dat functioneert in een productieomgeving.

Het volledig advies met een uitgebreidere uitleg is te vinden in de meegeleverde bijlagen map. Dit zijn de belangrijkste punten van dat bestand.

12.1 Aanbeveling

Is het waard om deze technieken in te zetten? Nee. Het doel van dit project was het vinden van een ML oplossing die automatisch fouten kon vinden. Om deze technieken werkend te krijgen moet er nog tijd en geld geïnvesteerd worden.

De STL en toets methode werkt niet consistent en heeft nog menselijk input nodig om fouten te vinden, door middel van de parameters te veranderen. De STL en toets is een hele geschikte methode om voor één tijdreeks fouten te vinden. Het is geen goede manier om voor veel meetpunten fouten te vinden op een automatische manier.

Hetzelfde probleem heeft het voorspellend model en afstand methode. Terwijl deze methode wel potentie heeft, is het niet handig om het te implementeren. Er moet een goede voorspellingsmethode komen welke consistent goed kan voorspellen, in verschillende situaties. Dit kost tijd, geld en vergt onderhoud. Dit wordt ook een heel ander probleemgebied dat opgelost moet worden. In plaats van de onderzoeken wat de beste methode is om data te valideren, moet er nu ook onderzocht worden wat de beste manier is om verkeersdata te voorspellen.

Het beste aan de validatieregels, is het feit dat ze simpeler zijn dan een ML oplossing. Hierdoor zijn ze makkelijk te onderhouden en is het ook duidelijk wat ze doen.

Terwijl het niet waard is om deze methoden in te zetten als validatieregels, is het misschien wel een interessant project voor de NDW om verkeer te voorspellen. De voorspellende modellen van dit onderzoek kunnen goed verkeer voorspellen omdat het verkeer terugkerende patronen heeft. Als NDW ooit plannen heeft om een verkeersdata te voorspellen dan zou de voorspelling output meegenomen kunnen worden in de datavalidatie.

12.2 Advies voor het uitbreiden van het onderzoek

Als er toch besloten wordt om verder onderzoek te doen naar deze methoden, dan zijn er de volgende mogelijkheden om te onderzoeken:

- De snelheid en intensiteit individueel valideren in plaats van te combineren.
- Verschillende voertuig/rijbaan indexen.
- Uitbreiden STL voorspellend model

- Met extra variabelen voorspellen
- Werking van de methoden op minuut data testen
- Groeperen van meetpunten

Deze worden veel uitgebreider uitgelegd in het adviesrapport in de meegeleverde map.

12.3 Advies systeem in productie

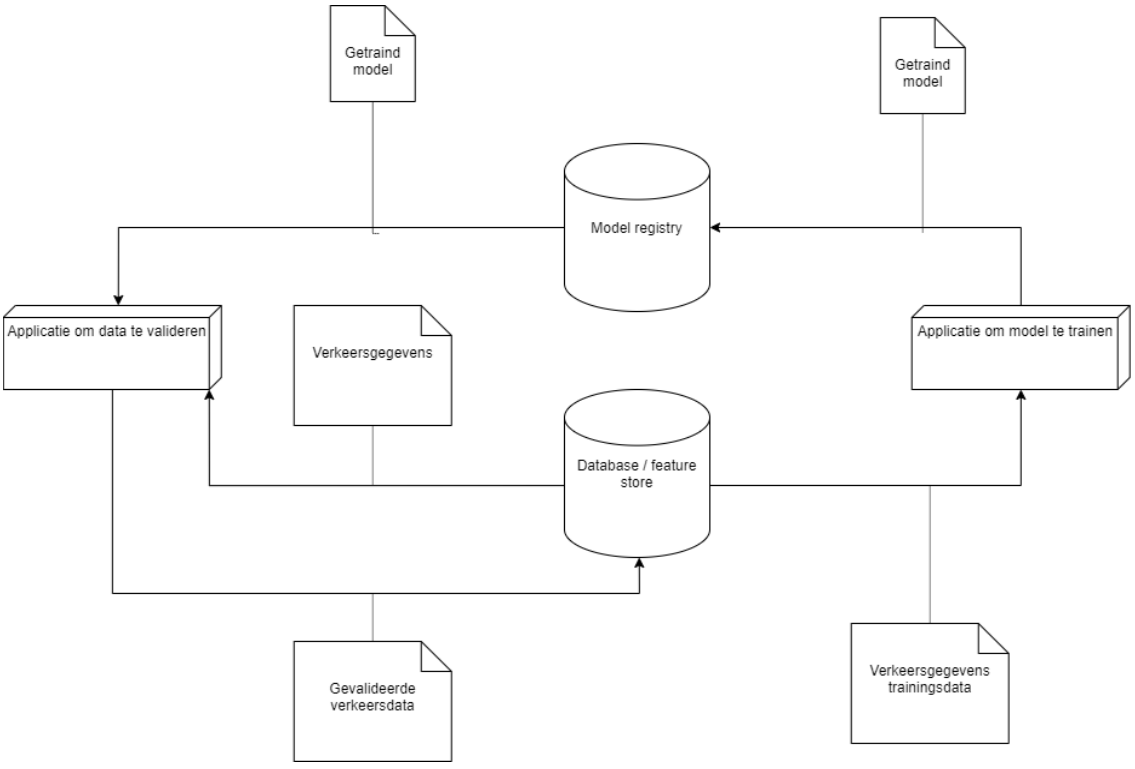
Om een validatiesysteem met Machine Learning op te zetten in een productieomgeving is er een systeem ontworpen. Het systeem zal bestaan uit twee applicaties:

- Een applicatie die in staat is om modellen te trainen
- Een applicatie die de modellen gebruikt om de data te valideren.

Het is handig om applicaties met verschillende functionaliteiten te scheiden. Hierdoor wordt het makkelijker om de verschillende applicaties te onderhouden.

En het systeem zal ook twee services hebben:

- Model registry
- Database/Feature store



Afbeelding 36: ML systeem met alle componenten in de NDW productie omgeving

12.3.1 Applicatie trainen modellen

Een mens kan een nieuw model trainen om de zoveel tijd maar dit is repetitief werk en er kunnen fouten gemaakt worden. Door een applicatie te ontwikkelen die in staat is om een model automatisch te trainen kan dit verholpen worden. Het moet in staat zijn om data op te halen, een model te trainen en ook om te valideren of dit model wel goed werkt.

Om te valideren of het model wel goed werkt kunnen metrieken ontwikkeld worden om het model te testen. Het model kan bijvoorbeeld getraind worden met een train/test split en getest worden op hoe goed deze presteert op deze twee datasets. Deze prestatie kan vergeleken worden met de prestaties van vorige modellen.

De prestaties van een model zullen na een tijdje minder worden. Het verkeerspatroon zal veranderen. Er kan een automatische manier opgezet worden om deze applicatie bijv. elke maand te draaien.

Het getrainde model wordt opgeslagen in de model registry.

12.3.2 Model registry

De model registry is de plek waar het getraind model opgeslagen wordt. De simpelste manier is om het model om te zetten naar geserialiseerd Python object en dit op te slaan is in een blob storage maar er zijn betere manieren om ML modellen op te slaan.

Er zijn ook model registries die nog extra functionaliteit bieden. Model registries kunnen de functionaliteit hebben dat het model ook geversioned wordt. Hiernaast kan ook metadata worden toegevoegd aan het model, zoals trainingsmetrieken. Hierdoor kan de engineer bepalen hoe goed het model is getraind(Rocco, 2021). De MLFlow Model registry is een voorbeeld model registry met extra functionaliteit(MLflow, z.d.). Er zijn geen andere model registries te vinden op het internet.

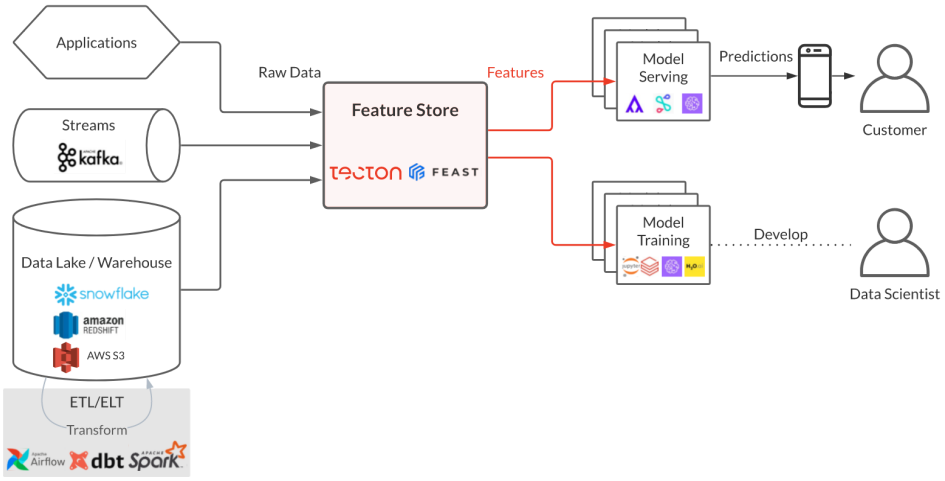
12.3.3 Feature store / database

De data moet ergens vandaan komen. De NDW gebruikt Azure blob storage en Azure Data Lake om verkeersgegevens op te slaan(persoonlijke communicatie e-mail Willem van Loon 10-3-2021 en 10-5-2021). Dit zijn databases waar ongestructureerde data opgeslagen kan worden(Microsoft, 2020). Deze databases kunnen gebruikt worden als databron voor het validatiemodel in productie.

Uit een gesprek met Arjen Wassink(08-03-2021) kwam tevoorschijn dat NDW ook van plan is om meer ML projecten op te zetten. Het is aan te raden om een feature store te implementeren wanneer NDW meerdere ML projecten wil opzetten.

De feature store houdt features bij. Een feature is input data van een ML model. Van variabelen kunnen afgeleide features gemaakt worden bijv. de uurintensiteit kan afgeleid worden van de minuut intensiteit. Door dit proces plaats te laten vinden in een feature store kunnen meerdere applicatie bij dezelfde features.

Een features store is ook een centrale plek waar data van verschillende bronnen samen wordt gevoegd. De feature store is in staat om de data op te slaan en te verwerken. De feature store kan data verlenen aan meerdere diensten(Balso, 2020).



Afbeelding 37: Feature store brengt data vanuit verschillende bronnen samen en verleent data aan verschillende diensten. (Balso, 2020)

De feature store zorgt dat de data gestandaardiseerd en het feature creatie proces geabstraheerd wordt. Het standaardiseren van de data zorgt er ook voor dat het ML model bij dezelfde data kan tijdens het trainen en wanneer het in productie draait.

Door het feature creatie proces te abstraheren kunnen er snel Machine Learning projecten opgezet worden. Nu hoeft elke data scientist niet zelf features te bedenken. Dezelfde dataverwerkings pipelines kunnen gebruikt worden voor verschillende projecten.

Door de modellen te ontkoppelen van de databron, zijn er ook minder afhankelijkheden. Het enige waar de modellen data vandaan halen is de feature store. De feature handelt de connectie met de databronnen en abstraheert dit(Balso, 2020).

Feast is een gratis open source feature store. Met Feast is het niet mogelijk om features te creëren. Er moet een andere applicatie komen die features creëert. Feast is voor het uitlenen, beheren en opslaan van data(Pienaar, 2021). Tecton is een andere feature store, die in staat om features te creëren, maar is niet gratis of open source(Balso, 2020).

12.3.4 Applicatie valideren verkeersdata

Er moet een applicatie komen die één keer per dag datavalidatie kan toepassen op de data.

De applicatie moet eerst controleren of er een nieuw getraind model beschikbaar is. Als dit het geval is moet het, het nieuwe model ophalen uit de model registry. Hierna moet de applicatie in staat zijn om verkeersgegevens op te halen uit een databron. De resultaten van de datavalidatie wordt geschreven in de databron.

Nu moet de applicatie zelf valideren dat het de laatste versie van het model heeft. Het model kan ook automatisch gedeployed worden naar deze applicatie met CI/CD.

12.3.5 Cloud oplossingen

Er zijn cloud oplossingen die dit van A t/m Z + extra's implementeren. Dit zijn alternatieven voor een on premise systeem.

Mogelijke cloud oplossingen:

- Amazon sage.
- Azure Machine Learning, NDW maakt al gebruik van Azure services.
- Google Cloud AI Platform.

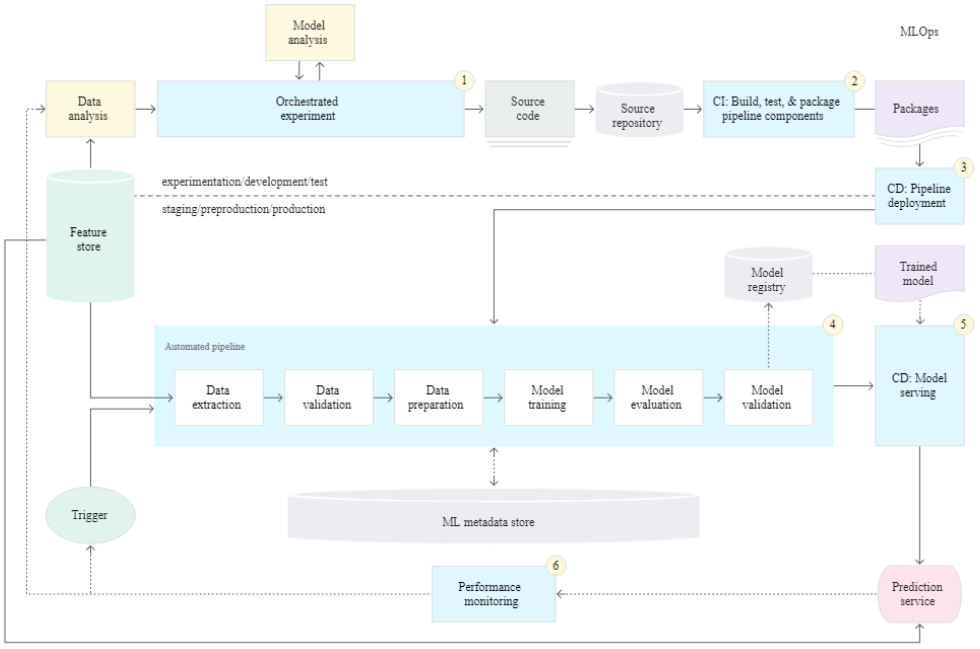
12.4 Advies MLOps

NDW is op zoek naar ML methoden om data te valideren. Een andere requirement was het onderzoeken van MLOps. *(1.B06) NDW wil onderzoeken hoe MLOps geïmplementeerd kan worden in de praktijk zodat NDW ook een ML systeem kan opzetten.* De informatie in dit hoofdstuk is beschreven door Google op de webpagina [MLOps: Continuous delivery and automation pipelines in Machine Learning].

Er zijn een aantal zaken welke fout kunnen gaan wanneer een iemand een ML oplossing implementeert in de praktijk. Een probleem is dat een model niet meer goed werkt omdat de onderliggende data veranderd. Het rijgedrag van de gebruikers op een weg kan veranderen. Als dit het geval is dan zal het getraind model niet meer werken. Om dit automatisch te detecteren en af te handelen kan MLOps geïmplementeerd worden.

MLOps is DevOps voor ML systemen. De grootste uitdaging is niet om een ML model te trainen die het goed doet. De uitdaging ligt in het creëren van een ML systeem dat in staat is om goed in productie te draaien. Door MLOps principes toe te passen, kunnen delen van het Machine Learning proces geautomatiseerd worden.

MLOps systemen implementeren continuous integration (CI), continuous deployment (CD), continuous monitoring (CM) en continuous training (CT). CM en CT zijn nieuwe aspecten die MLOps toevoegt aan het DevOps proces. CT is in staat om het ML model automatisch te hertrainen wanneer dat gewenst is. CM houdt de prestatie van het ML model bij.



Afbeelding 38: niveau 2 MLOps implementatie (Google, 2021)

Het begint met het opzetten van model-train-pipeline, die aangeeft hoe een ML model getraind moet worden. De pipeline begin met het verzamelen van data tot aan het model valideren.

Hierna wordt een ML model in productie gebracht en constant gemonitord. Door het monitoren kan het model-train-pipeline aangeroepen worden om automatisch een nieuw model te trainen, wanneer de prestatie van het productie model afzakt. Het nieuw getrainde model kan dan meteen in productie worden gebracht met CD(Google, 2021).

Er zijn al oplossingen die dit van A t/m Z implementeren, zoals de Google Cloud AI platform. Als NDW van plan is om zelf een systeem in elkaar te zetten kunnen de volgende tools gebruikt worden: (kelvins, 2021):

- Kubernettes om Kubeflow te draaien.
- Kubeflow als orchestrator van de automated pipeline en triggers. De automated pipeline heeft verschillende stadia, zoals data ophalen en model trainen. De orchestrator roept de ene stadia aan naar de ander.
- Tensorflow extended als framework voor de orchestrated en automated pipeline.
- Feast als feature store
- MLFlow model registry.
- Azure Data Lake als dataopslag

Er wordt sterk geadviseerd om MLOps te implementeren als NDW van plan is om nieuwe Machine Learning methode in te zetten. Door te vroegtijdig te investeren in het opzetten van een MLOps systeem zal een hoop ellende worden bespaard. Het MLOps systeem zal veel onderhoudstaken automatiseren. Hierdoor zullen ook projecten minder snel verlaten worden als deze ouder worden.

13 Reflectie

Dit project was een lastig project. Achteraf realiseer ik me dat ik heel wat dingen fout heb gedaan. Een grote blunder was niet goed van tevoren nadenken hoe deze validatiemethoden gevalideerd moeten worden. Een mogelijke manier om deze methoden te valideren is door middel van metrieken. Door creatief te denken en misschien wat meer tijd hier te investeren konden metrieken gevonden worden.

Door de metrieken te vinden kon er ook naar het realiseren van deze metriek gewerkt worden. Hierdoor zou ik een duidelijk doel hebben: implementeer een systeem die test of de methoden aan deze metrieken voldoen.

Het ging niet soepel. Het kon beter. Het hele idee van metrieken realiseerde ik me drie kwart door het project, wanneer ik al druk bezig was met de POC. Dit was nadelig voor het project maar goed voor mijn leerervaring. Ik heb expres voor dit soort project gekozen om tegen dit soort problemen aan te lopen. Nu weet ik voor een volgend project hoe het opgezet moet worden. Nu weet ik hoe een eindresultaat er ongeveer uit moet zien. Ik heb nieuwe dingen uitgeprobeerd, zoals het weglaten van een gelabelde dataset.

Ik heb geen spijt dat ik dit project hebt gekozen. Toen ik begon vond ik het ook eng maar ik denk niet dat ik dat gevoel zal hebben voor een tweede project. Al met al, ben ik blij met het resultaat en de ervaring die ik heb opgedaan.

Hier volgt nog een STARR reflecties:
S: Mij werd tijdens één van de demos verteld dat ik onduidelijk uitleg wat de methoden zijn van mijn aanpak en de resultaten ervan.
T: Mijn taak is om tijdens de demos uit te leggen wat ik ga doen en wat ik heb bereikt
A: Ik heb de volgende demo gepakt om mijn methoden opnieuw uit te leggen. Ik ben rustiger door mijn aanpak gelopen en heb gedetailleerder uitgelegd wat ik heb gedaan. Ik heb veel plaatjes in mijn uitleg meegenomen omdat ik het dan ook visueel kan uitleggen.
R: Ik kreeg het commentaar dat het nu een stuk beter was uitgelegd.
R: Ik merk dat het voor andere mensen lastig is om te volgen waar ik het over heb. Een constante kritiek die ik krijg is dat mensen niet snappen wat ik doe. Ik snap ook wel waarom mensen het soms niet volgen. Ik moet dan ook een keuze maken, moet ik er heel diep op ingaan waardoor ik hele lange presentaties hou, of moet ik het kort en bondig uitleggen waardoor mensen niet snappen wat er nou echt gebeurt. Het balans vinden is lastig. Ik denk wel, naarmate ik meer presentaties deed, dat ik er beter in ben geworden.

14 Bronnen

References

Anomalize Methods. (2020, 10 20). Anomalize. https://business-science.github.io/anomalize/articles/anomalize_methods.html

ArcGIS. (z.d.). *Understanding outliers in time series analysis*. esri. https://pro.arcgis.com/en/pro-app/latest/tool-reference/space-time-pattern-mining/understanding-outliers-in-time-series-analysis.htm

AWS. (z.d.). *Model Fit: Underfitting vs. Overfitting*. AWS. https://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html

CBS. (2020, 11 10). *Tijdreeksen*. Centraal Bureau voor de Statistiek. https://www.cbs.nl/nl-nl/onze-diensten/open-data/statline-als-open-data/tijdreeksen

Cleveland, R. B., Cleveland, W. S., McRae, J. E., & Terpenning, I. (1990). STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statisctics*, *6*, 33. http://www.wessa.net/download/stl.pdf

Cohen, I. (z.d.). *A Quick Guide to the Different Types of Outliers*. anadot. https://www.anodot.com/blog/quick-guide-different-types-outliers/

DABOTS. (2020, 11 5). *Linear Regression in Machine Learning*. DABOTS. https://dataanalyticsdots.com/ml-linear-regression/

Dash plotly. (n.d.). *Dash Testing | Dash for Python Documentation | Plotly*. https://dash.plotly.com/testing

Delua, J. (2021, maart 12). *Supervised vs. Unsupervised Learning: What's the Difference?* IBM. https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning

Fowler, M. (2006, 2 9). *CodeSmell*. Martin Fowler. https://martinfowler.com/bliki/CodeSmell.html

Galarnyk, M. (2018). *Different parts of a boxplot* [Afbeelding]. https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

ggplot2. (z.d.). *https://ggplot2.tidyverse.org/reference/geom_smooth.html*. Smoothed conditional means. https://ggplot2.tidyverse.org/reference/geom_smooth.html

Google. (2021, 28 05). *MLOps: Continuous delivery and automation pipelines in machine learning*. Google cloud. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

HAYES, A. (2020, 11 9). *T Distribution Definition*. Investopedia. https://www.investopedia.com/terms/t/tdistribution.asp

Heijst, L. v. (2020, 06 06). *Aannames bij statistische toetsen*. Scribbr. https://www.scribbr.nl/statistiek/aannames-statistiek/

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: principles and practice* (3rd ed.). OTexts: Melbourne, Australia. https://otexts.com/fpp3/index.html

Hyndman, R. J., & Khandakar, Y. (2008, 07). Automatic Time Series Forecasting: The forecast Package for R. *JSS Journal of Statistical Software*, *Volume 27, Issue 3*. 10.18637/jss.v027.i03

ISO 25000. (2020). *ISO/IEC 25010*. ISO 25000. https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

Khan Academy. (2016). *Judging outliers in a dataset* [Video]. https://www.khanacademy.org/math/statistics-probability/summarizing-quantitative-data/box-whisker-plots/v/judging-outliers-in-a-dataset

Lin, H. (2019, 9 7). *Use median absolute deviation instead of z-score to detect outliers*. Data science. https://hausetutorials.netlify.app/posts/2019-10-07-outlier-detection-with-median-absolute-deviation/

Microsoft Azure. (n.d.). *Data Lake*. Azure. https://azure.microsoft.com/nl-nl/solutions/data-lake/

Mitrani, A. (2020, 1 10). *Achieving Stationarity With Time Series Data*. Towards Data Science. https://towardsdatascience.com/achieving-stationarity-with-time-series-data-abd59fd8d5a0

NDW. (2010, 10 10). *NDW Interface beschrijving* [Afbeelding]. http://rotterdamopendata.nl/storage/f/2012-06-05T140030/ndw-interface-beschrijving-v1.pdf

NDW. (2019, 11 13). *Opdrachtbeschrijving Machine learning Datakwaliteit AVG* (V0.1).

NDW. (2021, 02 16). *NDW Docs: Inleiding Actuele Verkeersgegevens (AVG)*. NDW DOCS. https://docs.ndw.nu/nl/avg/index.html

NDW. (2021, 07 22). *Onze diensten*. Over NDW | Nationaal Dataportaal Wegverkeer. https://www.ndw.nu/over-ndw/onze-diensten

NIST. (2012). *NIST/SEMATECH e-Handbook of Statistical Methods*. https://doi.org/10.18434/M32189

Oracle. (z.d.). *Wat is machine learning?* Oracle. https://www.oracle.com/nl/data-science/machine-learning/what-is-machine-learning/

Plotly Dash. (n.d.). *Introduction | Dash for Python Documentation | Plotly*. https://dash.plotly.com/introduction

Python. (2001, 7 5). *PEP 8 -- Style Guide for Python Code*. Python. https://www.python.org/dev/peps/pep-0008/

Quintor. (2021, 05 02). *Wie we zijn*. Quintor. https://quintor.nl/over-ons/

Rocco, D. (2021, 06 10). *What is a Model Registry?* phData. https://www.phdata.io/blog/what-is-a-model-registry/

Rosenmai, P. (2013, 10 24). *Using the Median Absolute Deviation to Find Outliers*. Eureka statistics. https://eurekastatistics.com/using-the-median-absolute-deviation-to-find-outliers/

Shipmon, D. T., Gurevitch, J. M., Piselli, P. M., Edwards, S., & Google, Inc. (n.d.). Detection of Anomalous Drops with Limited Features and Sparse Examples in Noisy Highly

Periodic Data. https://storage.googleapis.com/pub-tools-public-publication-data/pdf/dfd834facc9460163438b94d53b36f51bb5ea952.pdf

Tiunov, P. (2017, 6 8). *Time Series Anomaly Detection Algorithms*. cube.js. https://blog.statsbot.co/time-series-anomaly-detection-algorithms-1cef5519aef2

VORtech. (2020, 2 20). *Detecting anomalous traffic data through Machine Learning* (0.1).

Vries, Y. d., & NDW. (2019, 10 21). *Toelichting offerteuitvraag Meetafwijkingdetectie met Machine Learning* [Powerpoint presentatie].

Wicklin, R. (2016, 10 17). *What is loess regression?* The DO Loop. https://blogs.sas.com/content/iml/2016/10/17/what-is-loess-regression.html

15 Bijlage gesprek: 5-3-2021 Interview Willem van Loon

Op vrijdag 5 maart vond een gesprek plaats met Willem van loon van de NDW.

De vragen die in het onderzoek staan zijn gesteld.

Een meetpunt is een lus op de grond. Dit kan verschillende dingen meten.

Er zijn er 17000. De locaties met rws01 kunnen eruit worden gefilterd om een sample data van 4000 te krijgen. Deze worden niet gemonitord door de NDW.

De opdracht is al een tijdje oud. Er zijn nu meerdere nieuwe validatieregels.

Vortech heeft het al een keertje geprobeerd. In dit project wordt er gekeken of er met tijdreeksen het ook kan worden opgelost. Dit is een andere invalshoek.

De nieuwe validatieregels zijn ook een stuk robuuster. Het is nog steeds met de hand gevalideerd. De validatieregels kunnen dingen tellen. Het is de vraag of de uitkomst van de validatieregels een goede indicator zijn voor het valideren van regels met ML.

Het is om te kijken of dit een mogelijke extra validatie/ vervanger is van sommige huidige validatieregels.

Misschien wordt er gekeken naar MLops.

De data kan op worden gehaald van dexter. Hier is een account voor nodig. Alleen kijken naar snelheid en intensiteit.

De data is soort van gelabeld. er is een link genaamd /exlusions waar alle fouten meetpunten zijn gedetecteerd.

Er is een shape file waar alle meetpunten in staan. Hier kan een sample van gemaakt worden. met deze shapefile staan alle meetpunten in. Met een Python lib kan er een call worden gemaakt naar de server om de data op te halen.

Als het goed is krijg ik ook nog een lijst met de uitslag van de validatieregels.

Liefst wel in Python als je die dingen wil gebruiken. Ik ga het denk ik doen in R eerst.

16 Bijlage gesprek: 19-3-2021 Willem van Loon, Arjen Wassink

Gesprek gehad over het gemaakte ontwerp en requirements. Door het ontwerp heengelopen. Nog niet helemaal zeker wat waar komt maar dat wordt volgende week onderzocht.

Het ontwerp mist de ML gedeelte.

Voor de requirements kijk in de casus. De requirement voor hoe snel het moet gaan is aanzienlijk minder dan een dag. kan ook 10 minuten duren om te duren Aanzienlijk minder dan 24 uur. fractie van een dag.

De ml oplossing moet niet wat de validatieregels al vinden detecteren. dit wordt al gedaan door de validatieregels en deze zijn efficiënter.

Het moet dingen die niet door de validatieregels worden opgevangen kunnen detecteren. Het maakt niet uit of het alles kan detecteren wat in de exclusions staat. Het moet nieuwe dingen detecteren.

De ML methoden moet de validatieregels uitbreiden in plaats van compleet vervangen.

Het moet dus focussen op unsupervised learning methods.

Niet trainen op basis van exclusions. De exclusions moeten gebruikt worden om de data te valideren.

Het liefst ook werken met minuut data. Met uurdata wordt er teveel geaggregeerd en gaan er teveel dingen verloren.

feature store onderzoeken kijken of dit wel nodig is.

Tip: neem de tijd te nemen om alle aspecten te onderzoeken. That will pay off in the longrun

bronnen voor onderzoek

Docker en kubernettes ook in quintor academy.
https://academy.quintor.nl/moodle/login/index.php

http://cloud-native-minor.s3-website.eu-central-1.amazonaws.com/lecture-1-3-4_Containers.html#/

http://cloud-native-minor.s3-website.eu-central-1.amazonaws.com/assignment-1-3-4_Containers.html

17 Bijlage gesprek: 9-4-2021 Willem van Loon

Resultaten
start datum wordt aan handmatig ingevuld aan de hand van visuele analyse. Bij exclusions.

hoge intensiteit kan ook file zijn. Worden alleen verkeerskundig situaties gedetecteerd. Hier moet je een oplossing voor vinden.

Kijken naar 5 minuten. kijken naar minuut data als het gedetecteerd is in uur data. Voortschrijdende gemiddelde, moving average kijken.

is een dashboard een goed eindproduct?

Ja, dat is prima

een dashboard maken die help met het maken van een keuze, Visualiseert, geeft statistieken + performance van de methoden.

Maak mock-ups van hoe het dashboard eruit ziet.

data ophalen uit met een API. Er zijn verschillende APIs. En goed ook kunnen testen. Liever niet met een csv tje .

Python of R? Liever Python
dash is iets om dashboards meet te maken in Python.

Taken
Afmaken model.
Mock-ups. Maandag
Onderzoeksvraag aanpassen.
Kleine vraag gewoon stellen.
Nieuwe meeting Donderdag.

18 Bijlage gesprek: 15-4-2021 Willem van Loon

Onderzoek
Verschil tussen voorspeld punt en het daadwerkelijk punt kunnen kijken in plaats van kijken naar alleen prediction interval.

Kan ook kijken naar beide. Waarbij je kijkt of een prediction interval breed is. Als je weet dat het niet breed is dan weet je dat de voorspelling goed is. Hierna kan je kijken naar de afstand.

Kan ook kijken naar stukken dat over een interval fout is in plaats van één punt.

Je kan ook proberen te voorspellen wanneer een punt defect is. Je moet dus door het verleden kunnen lopen en kijken wanneer de voorspellingen anders beginnen te worden. Dat is het punt wanneer het fout gaat.
- Kijken naar exclusions om te kijken of je het kan voorspellen.
- Past beter in het onderzoek

Dashboard
- Meerdere meetpunten in één grafiek doen.
- Het model kan je dan zien of het anders werkt op meerdere meetpunten.
- Evaluatie ook goed zoals hoe snel en hoeveel geheugen

workflow
je selecteert een meetpunt.
Je selecteert de time frame
je selecteert bron data en te voorspellen data.
selecteert zelf je train en test. met tijd ertussen zou kunnen.

Liever in Python
onderzoek doen naar een andere dashboarding. Ik hoor het ook graag als er andere alternatieven is.
Moet echt een interactief dashboard zijn.
Dash of bokeh kan ook echt in Python.

Database

Api is al beschikbaar om Python om data op te halen.

Toegang tot de data wordt geregeld.

Liever iets dat ze meteen kunnen gebruiken in plaats van iets waar ze nog aan moeten sleutelen

19 Bijlage gesprek: 17-5-2021 Willem van Loon

Uitleggen goed hoe je de applicatie hebt gemaakt

- Dan kunnen zij zelf er nog aan werken en dingen toevoegen
- Een conclusie trekken aan de hand van het dashboard
    - Dashboard is een tool om dingen te testen zonder zelf te hoeven programmeren, Dit is de reden waarom het gebouwd is

- Aangeven hoe je de data geaggregeerd en waarom je dingen niet hebt gedaan. Dat is ook zodat zij nog dingen zelf kunnen toevoegen.
    - gemiddelde snelheid is het harmonisch gemiddelde
    - Uitleggen hoe je de data bewerkingen hebt gedaan

- Willem ook uitnodigen voor eindpresentatie
    - Hij is op vakantie van 31 mei tot 13 juni

20 Bijlage gesprek: 28-5-2021 Willem van Loon

Met Willem door de applicatie gelopen:

- Hij was er tevreden mee
- Moeten nog instructies in de applicatie komen

vortech demo :

- Ze hadden een poc gemaakt maar zagen geen toegevoegde waarde om het te implementeren

21 Bijlage: Gesprek Willem 2-9-2021

opdrachtgever
vragen Willem:

- Voorspelt de NDW al verkeersdata?
    - Nee nog niet maar er is wordt wel een project een project gaande om het te doen.
- Waarom weer opgezet project?
    - NDW is altijd op zoek naar nieuwe methoden om de kwaliteit te waarborgen.
- Kan een meetpunt tijdelijk kapot zijn?
    - In theorie zou het wel kunnen, maar komt niet zo veel voor

Opmerkingen van Willem:
Het was handig geweest als de opdracht concreter was gemaakt. Dat of meer analyse model en model training of iets meer naar de ICT kant moest worden getrokken. (ICT kan, hoe je het model in productie brengt)

De opdrachtgever vindt de adviezen zijn helder, jammer dat er geen werkend model to stand is gekomen.

22 Bijlage: Inhoud afstudeerdossier

Dit is een beschrijving van de documenten in het afstudeerdossier

Dit afstudeerverslag
welke het probleem en aanpak hiervan beschrijft.

Eindbeoordeling Ik
Eindbeoordeling van mij, met de competenties die ik heb vervuld

Eindbeoordeling begeleiders
Eindbeoordeling bedrijfsbegeleiders

Scrum Document
Beschrijft het scrum document

Userstories
Een opsomming van de user stories en details

Adviesrapport
Vervolg advies verdere aanpak probleem
Ontwerp systeem om te voorspellen
MLOps implementatie ontwerp

Technische documentatie
Functioneel ontwerp dashboard
Technisch ontwerp dashboard
Testen samenvatting

Code
Code van het dashboard

Onderzoeksdocument
Document met alle literatuur, data en expirimenteelonderzoek

Resultaat document
De resultaten van hoofdstuk 8 volledig.

Met de volgende documenten
- Onderzoek.html
- STL.html
- ETS.html
- ARIMA
- ETS.html
- ARIMA.html