

Afbeelding met openbaar vervoer, transport, Spoorbaan, voertuig

Automatisch gegenereerde beschrijving

Inhoud

[1. ProRail 2](#_gjdgxs)

[1.1 Probleem 2](#_30j0zll)

[1.2 Doelstelling 2](#_1fob9te)

[1.3 Oplossing 3](#_3znysh7)

[2. Data 3](#_2et92p0)

[2.1 Beschrijving dataset 3](#_tyjcwt)

[2.2 Data Understanding 4](#_3dy6vkm)

[2.3 Data Preparation 5](#_1t3h5sf)

[2.4 Beschrijving opgeschoonde dataset 5](#_4d34og8)

[3. Model 6](#_2s8eyo1)

[3.1 Keuze voor bepaalde modellen 6](#_17dp8vu)

[3.2 Toelichting van hyperparameters 6](#_3rdcrjn)

[3.3 Resultaten van experimenten 6](#_26in1rg)

[3.4 Overzicht alle experimenten 7](#_lnxbz9)

[3.5 Evaluaties resultaten 8](#_35nkun2)

[4. Dashboard 8](#_1ksv4uv)

[4.1 Overzicht requirements 8](#_44sinio)

[4.2 Keuze ontwikkelomgeving 8](#_2jxsxqh)

[4.3 Resultaten 9](#_z337ya)

[5. Conclusie en aanbeveling 10](#_3j2qqm3)

[5.1 Conclusie 10](#_1y810tw)

[5.2 Aanbeveling 10](#_4i7ojhp)

# ProRail

## Probleem

ProRail is een bedrijf dat verantwoordelijkheid heeft voor alle sporen in Nederland voor de aanleg, het onderhoud, beheer en de veiligheid van het spoor. Dus moet ProRail problemen oplossen die te maken hebben met het spoor. Dit kunnen veel verschillende problemen zijn: dit kan iemand zijn die te dicht bij het spoor loopt of dat het spoor ergens kapot is. Het kost natuurlijk tijd om deze problemen te analyseren en op te lossen.

Nu wordt deze tijd ingeschat door de aannemers die het probleem gaan oplossen met vakmanschap. Deze prognose kan natuurlijk goed zijn, maar er wordt ook wel vaker een foute prognose gesteld. Volgens de gegevens in de database is de prognose van de aannemer 19% van de tijd goed.[[1]](#footnote-0)

## Doelstelling

Het doel van dit project is om de planners te ondersteunen bij het organiseren van de hervatting van het treinverkeer. Dit doen we door een nauwkeurige voorspelling te maken van hoe lang het duurt voordat de functie van het spoor hersteld is en de trainen weer over het spoor kunnen rijden. Momenteel komt het vaak voor dat het probleem al is opgelost, terwijl de inschatting was dat het nog een uur zou duren, waardoor er geen personeel en trein klaarstaat. Dit gebeurt omdat aannemers vaak te pessimistisch zijn in hun prognoses.

Ons doel is om een meer precieze tijdsindicatie te geven, zodat de organisatie op de juiste momenten personeel en treinen gereed heeft staan. We gaan dit bereiken door een model te trainen en een GUI te ontwikkelen die de informatie weergeeft. Er zijn daarbij enkele eisen gesteld door ProRail, zodat ze de tool goed kunnen gebruiken. We moeten een GUI maken waarop de betrouwbaarheid en nauwkeurigheid van onze inschatting zichtbaar zijn, evenals de geschatte tijd.

## Oplossing

Om het probleem op te lossen, gaan we een model maken dat betrouwbaarder en nauwkeuriger is dan de prognoses van de aannemer. Dit model zal laten zien hoe nauwkeurig het is en hoe betrouwbaar het kan zijn. We doen dit door te kijken naar de percentages en te analyseren hoe vaak het model naast de juiste inschatting zit wanneer het fout is.

*Figuur 1.1 Dit is het BPMN dus het proces van ProRail*

Afbeelding met tekst, diagram, schermopname

Automatisch gegenereerde beschrijving

Hier in kan je ook zien waar ons model in komt dus waar we onze schatting maken.

# Data

## Beschrijving dataset

We hebben een dataset gekregen met 139 kolommen. Veel van deze kolommen zijn gemarkeerd als n.v.t. of ?, wat betekent dat ze niet belangrijk zijn. Ook hebben we de regels met prioriteit 9 verwijderd. Dit komt omdat deze regels administratief zijn en geen functie herstel hebben. Ook dingen zoals kopieën, hoewel de dataset veel informatie bevat, is een groot deel hiervan niet bruikbaar voor onze modellen. Daarom zullen we veel kolommen verwijderen, wat we verder zullen uitleggen in de Data Preparation-fase.

## Data Understanding

Zoals al eerder gezegd, bevat deze dataset veel data. Door de data te onderzoeken, zijn we uitgekomen op een paar belangrijke kolommen. Deze kolommen kiezen we door overleg met PO. Daarnaast hebben we ook een correlatie tabel gemaakt, maar er waren geen sterke correlaties met de target variabele. Dit zijn de kolommen die de grootste invloed zullen hebben op het inschatten van de tijd. We hebben ook zelf een tabel toegevoegd aan de database; dit is de tijd die we gaan voorspellen. Dit betreft de tijd tussen het moment dat de aannemer bij het probleem aan komt en het moment dat er weer functie herstel is. De belangrijke kolommen kunt u vinden in het notebook met de bijbehorende uitleg over wat de tabellen doen. Hieronder kunt u de correlatie tabel zien.

*Figuur 2.1 Correlatie tabel.*

*Afbeelding met tekst, schermopname, patroon, steek

Automatisch gegenereerde beschrijving*

## Data Preparation

Er zijn verschillende keuzes gemaakt bij het opschonen van de database. Kolommen zijn uit de database verwijderd op basis van de volgende criteria: als ze beschrijvende informatie bevatten, te veel lege waarden hadden, of geen toegevoegde waarde boden voor ons model. Een precieze lijst van de verwijderde kolommen is te vinden in het notebook.

Daarnaast hebben we overwogen hoe we de tijd het beste konden voorspellen. Uiteindelijk hebben we gekozen om 12 bins te maken met een grofweg gelijk aantal datapunten. Dit zorgt ervoor dat we veelvoorkomende tijden beter kunnen inschatten. Deze aanpak bracht echter het probleem met zich mee dat sommige bins te klein of juist te groot werden. Daarom hebben we 12 bins gemaakt waarbij geen enkele bin een tijdsduur van meer dan 2 uur heeft. Kleinere bins kunnen we indien nodig samenvoegen. Hieronder kunt u een grafiek zien van hoe de bins eruit zien.

*Figuur 2.2 Bins verdeling tijd staat in minuten*Afbeelding met tekst, schermopname, Perceel, diagram

Automatisch gegenereerde beschrijving

## Beschrijving opgeschoonde dataset

Na het opschonen van de database blijven er ongeveer 15 van de 139 kolommen over, en 48.000 van de 89.000 rijen. De belangrijkste kolommen voor onze modellen zijn: 'stm\_geo\_mld', 'stm\_prioriteit', 'stm\_oorz\_code', 'stm\_contractgeb\_gst', en 'stm\_techn\_mld'. Uitleg voor deze tabellen kunt u vinden in het notebook.

# Model

## Keuze voor bepaalde modellen

Voor de modellen die we hebben onderzocht, moest het model een probabilistische output geven, omdat dit gebruikt zal worden om een betrouwbaarheidspercentage te berekenen. Vervolgens hebben we een reeks modellen getraind en zijn we verder gegaan met de modellen die de laagste log loss en de hoogste nauwkeurigheid hebben.

## Toelichting van hyperparameters

We hebben eerst geprobeerd de beste vijf modellen te brute-forcen, maar dit duurde te lang, dus zijn we daarmee gestopt. Vervolgens probeerden we het beste model aan te passen, maar ook dat kostte teveel tijd. Uiteindelijk hebben we ervoor gekozen om een DecisionTreeClassifier te gebruiken en die handmatig te optimaliseren die niet veel slechter was dan de beste.

De instellingen die we gebruikten waren:

* criterion: 'gini'
* splitter: 'best'
* max\_depth: 15
* min\_samples\_split: 2
* min\_samples\_leaf: 1
* max\_features: 'log2'
* class\_weight: None

Met deze instellingen behaalde het model een nauwkeurigheid van 0.22 en een log loss van 3.00.

## Resultaten van experimenten

Als resultaat van het DecisionTreeClassifier-model behalen we een accuracy van 20% en een log loss[[2]](#footnote-1) van 3. Dit betekent dat ons model wel heel zeker is van zijn inschattingen, hoewel de nauwkeurigheid eerder 27% was. Deze hogere waarde kwam echter door overfitting, wat we ook hebben onderzocht. Hieronder ziet u een grafiek die dit illustreert.

*Figuur 3.1: Een grafiek die toont wanneer het model overfit is.*

*Afbeelding met tekst, Perceel, lijn, diagram

Automatisch gegenereerde beschrijving*

## Overzicht alle experimenten

Daarnaast hebben we ook nog tien andere modellen getraind, maar hadden we niet genoeg tijd om deze verder te onderzoeken. Hieronder volgt een lijst van deze modellen met hun respectieve accuratesse en log loss:

| Model | Accuracy | Log Loss |
| --- | --- | --- |
| BeggingClassifier | 0.27 | 5.06 |
| RandomForestClassifier | 0.28 | 7.10 |
| ExtraTreesClassifier | 0.28 | 7.10 |
| GaussianNB | 0.02 | 35.30 |
| NearestNeighborsClassifier | 0.22 | 15.54 |
| MLPClassifier | 0.22 | 2.19 |
| LinearTreeClassifier[[3]](#footnote-2) | 0.21 | 28.44 |
| LinearBoostClassifier | 0.20 | 28.76 |
| VotingClassifier | 0.27 | 2.42 |
| DecisionTreeClassifier | 0.28 | 8.68 |

Zoals u kunt zien, waren er nog verschillende veelbelovende modellen die we wilden testen, maar helaas hadden we daar niet genoeg tijd voor. Dit biedt een interessant perspectief voor vervolgonderzoek, waarin meer modellen kunnen worden getest op overfitting en onderzocht of ze na hyperparameter-tuning beter presteren dan de DecisionTreeClassifier. Het is echter niet gegarandeerd dat deze modellen beter zullen presteren; het is ook mogelijk dat ze gaan overfitten en daardoor een lager prestatieniveau behalen dan de DecisionTreeClassifier.

## Evaluaties resultaten

Ons eindproduct is een DecisionTreeClassifier-model met een accuracy van 22% en een log loss van 3. Hoewel het model het slechts iets vaker goed heeft dan de aannemers, die 19% van de voorspellingen correct hebben, is de lage log loss een belangrijke eigenschap. Deze lage log loss betekent dat ons model, wanneer het een goede voorspelling maakt, veel zekerder is dat deze inschatting accuraat is.

# Dashboard

## Overzicht requirements

Het dashboard moet voldoen aan een aantal eisen. Ten eerste moet de gebruiker de feature variabelen kunnen invullen. Dit is van belang omdat de gebruiker anders niet een prediction kan krijgen.  
Ten tweede moet de dashboard een grafiek kunnen weergeven met informatie over de resultaten van het model. Deze informatie bestaat uit welke klassen het model voorspelt en hoe zeker hij is over die klassen. Dit is nodig zodat de gebruiker een geinformeerde beslissing kan maken over zijn keuze.

## Keuze ontwikkelomgeving

Er is gekozen om de interface in PyQT te maken. De reden hiervoor is omdat PyQT een breed scala aan customisability geeft in het maken van een GUI. Hiernaast is er ook ervaring hiermee in de ontwikkelteam, dit zorgt ervoor dat er gemakkelijk iets gemaakt kan worden. Ook had de PO aangegeven dat het niet uitmaakt wat voor GUI library gebruikt wordt.

## Resultaten

Met de requirements in gedachte zijn er 2 GUI’s gemaakt. De eerste is de GUI waar de gebruiken de feuture variabelen kan invullen en daar een voorspelling terug kan krijgen.

Afbeelding met tekst, schermopname, scherm, software

Automatisch gegenereerde beschrijving

Hiernaast is er een tweede GUI gemaakt. Hierop is te zien waar de storing zich bevindt, en ook informatie over de storing zelf. Op de kaart zijn alle spoorwegen inclusief stations te zien. De stations zijn aangegeven met een blauwe markering en de storingen met een roode markering.

De GUI is te zien op website die [hier te vinden is.](https://mightypiggie.github.io/data-driven-bussines-ai-2024/)

# Conclusie en aanbeveling

## Conclusie

Hoewel we verschillende benaderingen hebben getest om het model te optimaliseren, leverde geen van de pogingen de gewenste resultaten op binnen een redelijke tijd. Het brute-forcen van de beste modellen bleek te lang te duren, en ook het aanpassen van het best presterende model was niet efficiënt. Uiteindelijk hebben we besloten om een DecisionTreeClassifier te gebruiken, die we handmatig hebben getuned. Helaas kunnen we dit model niet beter maken dan een nauwkeurigheid van 0.22 en een log loss van 3.00.

## Aanbeveling

Gezien de lage prestaties van de DecisionTreeClassifier raden we aan om andere algoritmes te verkennen die mogelijk beter geschikt zijn voor deze taak. Modellen zoals de RandomForestClassifier en VotingClassifier zouden mogelijk betere resultaten kunnen opleveren. Of het model uiteindelijk gebruikt zal worden, hangt af van de PO (product owner), aangezien ons model wel wat beter presteert dan de aannemers. Het is echter mogelijk niet de beste beslissing om dit model in plaats van het advies van de aannemers te gebruiken. Wellicht kan ons model als tweede keuze dienen of als back-up, vooral als de aannemers zelf twijfels hebben over hun eigen voorspellingen.

1. Hiermee wordt de baseline model gebruikt om “goed” te classificeren. [↑](#footnote-ref-0)
2. https://en.m.wikipedia.org/wiki/Loss\_functions\_for\_classification [↑](#footnote-ref-1)
3. https://github.com/cerlymarco/linear-tree [↑](#footnote-ref-2)