



UNIVERSITEIT VAN AMSTERDAM

LEREN EN BESLISSSEN

STOKHOS

Incident voorspellingen met behulp van machine learning

Auteurs:

Tessa VERMEULEN
Camille NIESSINK
Stijn HERING
Isa-Ali KIRCA

Begeleiders:

Faysal EL KAAOICHI
Bjarty GARCIA

31 januari 2021



Samenvatting

Het is van belang dat de reactietijd van ambulances zo kort mogelijk is, waardoor mensen in nood beter geholpen kunnen worden. In de meeste gevallen komen de ambulances op tijd bij de incidenten aan, echter is hier nog wel wat verbetering in mogelijk. Voor het verbeteren van de reactietijd is het van belang dat de ambulances zo efficiënt mogelijk worden verdeeld over de regio. Hiervoor heeft Stokhos een model ontwikkeld dat voorspellingen maakt over de kans dat een incident plaatsvindt op een bepaalde tijd en plaats.

Stokhos had voor dit project de opdracht gegeven om een uitbreiding te maken op het model dat zij ontwikkeld hebben, zodat er ook verschillende tijds patronen meegenomen kunnen worden.

Om dit doel te bereiken is de data getraind door middel van een *Extreme Gradient Boosting* (XGB) algoritme en een *Long Short-Term Memory* (LSTM) algoritme. Vervolgens is de accuratie van beide modellen met elkaar vergeleken en is er gekeken hoe toepasbaar de modellen zijn voor Stokhos. Het XGB algoritme en het LSTM hebben respectievelijk een accuratie behaald van 94 en 74 procent. Naast het feit dat het XGB algoritme een hogere accuratie heeft behaald, was het model ook beter toepasbaar. Daarnaast kon het XGB algoritme ook de plaats van een incident meenemen in de voorspellingen. Bij het LSTM werd de plaats niet meegenomen. Na het analyseren van beide modellen, kon dus geconcludeerd worden dat het XGB algoritme beter aan de eisen van de opdracht voldoet.

Inhoudsopgave

1	Introductie	3
1.1	Probleemstelling	3
1.2	Data	3
1.2.1	Details van de data	3
2	Methode	4
2.1	Data pre processing	4
2.2	Gekozen modellen	5
2.2.1	XGB Classifier	5
2.2.2	Implementatie XGB Classifier	5
2.2.3	LSTM	7
2.2.4	Implementatie LSTM	7
3	Resultaten	9
3.1	XGB Classifier	9
3.1.1	Model zonder locaties	9
3.1.2	Model met wijken	11
3.1.3	Model met gemeentes	12
3.2	LSTM	14
4	Conclusie	15
4.1	Bruikbaarheid	15
4.2	Mogelijke verbeteringen	15
4.2.1	Data pre-processing en analyse op basis van het gekozen model	16
4.2.2	Complexiteit van het LSTM model	16
4.2.3	Meer trainingsdata	16

1 Introductie

Dagelijks komen er meer dan honderd meldingen binnen bij de meldkamer van incidenten waar medische hulp bij nodig is. Het ene incident vereist meer spoed dan de ander en in sommige gevallen, wanneer een ambulance niet op tijd aankomt, kan dit tot ernstige gevolgen leiden. Tot nu toe komt in 7 procent van de spoedgevallen de ambulance te laat bij incidenten aan. Dit betekent dat er naar schatting 35.000 patiënten per jaar niet op tijd de juiste zorg ontvangen.

Dit percentage kan verbeterd worden door ervoor te zorgen dat de ambulances zo efficiënt mogelijk zijn verdeeld over de regio, zodat de reactietijd zo kort mogelijk is.

Stokhos is een bedrijf dat zich hiermee bezighoudt. Ze hebben namelijk een model ontwikkeld dat voorspellingen kan maken over de waarschijnlijkheid dat er een incident plaatsvindt op een bepaalde locatie en tijd. Dit kan ervoor zorgen dat de ambulances zich van te voren beter kunnen positioneren in de regio.

1.1 Probleemstelling

Het model dat Stokhos ontwikkeld heeft, voorspeld nu echter alleen nog op jaarbasis. Dit betekend dat alle voorspellingen voor elke dag hetzelfde zijn en wordt er dus nog geen rekening gehouden met de verschillen in dagen of seizoenen.

Stokhos heeft voor dit project dus de opdracht gegeven om een model te ontwikkelen dat ook verschillende tijdspatronen meeneemt bij het maken van deze voorspellingen. De uitdaging van deze opdracht is om te kijken of er met behulp van machine learning voorspeld kan worden wat de kans is dat een type incident plaats zou kunnen vinden op een bepaalde tijd voor elke locatie.

1.2 Data

De data die Stokhos beschikbaar heeft gesteld bestaat uit alle meldingen die zijn verwerkt over de periode van mei 2018 tot en met mei 2020 in de regio Zuid-Holland Zuid. Deze dataset bestaat uit 84.459 regels en elke regel bestaat uit informatie over een incident melding die op een bepaald tijdstip gemaakt is.

In totaal bevat de dataset 26 features. Deze features worden in sectie 1.2.1 verder uitgewerkt.

1.2.1 Details van de data

De feature ‘DTG Melding’ bevat de datum en tijd waarop alle meldingen binnen zijn gekomen in de meldkamer. ‘DTG TP’ geeft de datum en tijd weer van het moment dat het ambulancepersoneel bij het incident ter plaatse is. Bovendien staan in de dataset verschillende features over de plaats van het incident. De kolommen van deze features bevatten onder andere gegevens over de postcode, straat, huisnummer en gebied, maar ook de X en Y coördinaten van het rijksdriehoeksstelsel. Naast alle gegevens over de locatie van het incident staan er ook gegevens in de dataset over waar de ambulance na het incident naartoe gaat en welk type incident heeft plaatsgevonden.

Verder bevat ‘Voertuigsoort afk’ informatie over welk voertuig naar een incident wordt gestuurd en de feature ‘Prioriteit CPA’ geeft de prioriteit van het incident weer. De prioriteit zegt iets over de ernst van het incident. De categorieën van deze feature zijn A1, A2, B1 en B2. Een incident met prioriteit A1 vereist de hoogste spoed en B2 de minste spoed. De laatste feature ‘Rit soort afsluiting’ bevat informatie over het soort rit. ‘Overplaatsing’ is bijvoorbeeld een rit soort waarbij een patiënt overgeplaatst moet worden.

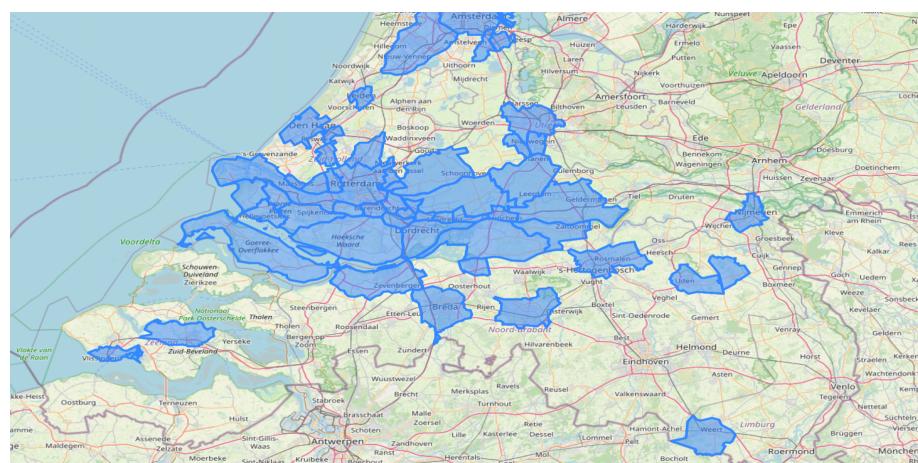
2 Methode

2.1 Data pre processing

Om verder te kunnen werken met de beschikbare data, moesten er een aantal stappen ondernomen worden. De eerste stap was het bepalen welke features het meest relevant waren. Aangezien het project voornamelijk gaat over het voorspellen op welke tijd de kans het grootst is dat er een incident plaatsvindt, is er gekozen voor de feature ‘DTG Melding’. Hiermee zijn middels *feature engineering* nieuwe features gecreëerd. Voorbeelden hiervan zijn dag, maand, uur,dagdeel en seizoen.

Daarnaast is er met behulp van de features ‘van_X_Coord’ en ‘van_Y_Coord’ ook een koppeling gemaakt met wijken en gemeentes in de regio. Deze coördinaten zijn namelijk omgezet naar GPS-coördinaten, waardoor het mogelijk werd om de locaties van incidenten in kaart te brengen. In figuur 1 zijn de gemeentes te zien waarin de incidenten plaats hebben gevonden.

De hoeveelheid meldingen in een aantal wijken en gemeentes was erg laag. Dit zou betekenen dat bij het trainen van de modellen de kans groot was dat er overfitting? zou plaatsvinden. Vandaar dat er eerst is gekeken naar het gemiddelde aantal meldingen dat gedaan is. De wijken en gemeentes waarvan het aantal meldingen onder dit gemiddelde zaten zijn uit de dataset verwijderd. Met alle nieuwe features voor tijd en plaats is een nieuwe dataset gemaakt waarmee de modellen zijn getraind.



Figuur 1: Gemeentes waar incidenten hebben plaatsgevonden.

2.2 Gekozen modellen

De essentie van het project is voornamelijk het classificeren met behulp van *time-series*. Algoritmes die hier mogelijkheden in hebben, zijn een *Extreme Gradient Boosting* classificatie algoritme (XGB Classifier) en een *Long Short-Term Memory* algoritme (LSTM).

2.2.1 XGB Classifier

Het XGB classificatie algoritme is een *boosting* algoritme dat gebaseerd is op een beslisboom met gradiënt versterking. Gradient boosting is een ensemble techniek waarbij nieuwe modellen worden toegevoegd om de fouten van vorige modellen te corrigeren. In elke iteratie van het algoritme worden modellen getraind en achtereenvolgens toegevoegd tot er geen verdere verbeteringen meer aangebracht kunnen worden. Het verschil met andere gradient boosting ensemble technieken is dat XGB een betere regularisatie techniek toepast om overfitting te verminderen.

Over het algemeen is het algoritme erg snel. Voornamelijk als het wordt vergeleken met andere implementaties van gradient boosting. De XGBClassifier is goed in het herkennen van patronen in timeseries data, daarnaast is het model erg robuust voor ruis in de data. De XGBClassifier kan hierdoor goed met eventuele *outliers* omgaan die in timeseries data voorkomen. De XGBClassifier is hierdoor praktisch en relevant voor de Stokhos case(Yang et al., 2019).

De XGBClassifier is een veelgebruikt model bij risico voorspellingen, dit omdat de XGBClassifier erg accuraat en betrouwbaar is in vergelijking met andere classificatie modellen. Daarnaast is het model ook goed in het identificeren van de belangrijkste features die het meeste impact hebben op de uiteindelijke kansen die het model produceert. Hierdoor helpt het model de gebruiker in de juiste richting te sturen, wat de sleutel is tot een betere modellering(Shi et al., 2019). Daarnaast is het algoritme goed te gebruiken bij time-series. Dit maakt de XGB Classifier erg praktisch voor dit project.

2.2.2 Implementatie XGB Classifier

Er zijn drie verschillende XGB Classifiers getraind en getest, namelijk een algemeen model zonder locatie, een model met wijken en een model met gemeentes.

Voordat de modellen getraind konden worden, werden er tijdsintervallen gekozen om vervolgens de meldingen te tellen die binnen de tijdsintervallen hadden plaatsgevonden. De tijdsintervallen zijn gekozen door het aantal meldingen per interval om te zetten naar binaire vorm. Dit wil zeggen dat de intervallen waarin geen meldingen hadden plaatsgevonden gelabeld werden met een nul en alle andere intervallen waarin wel meldingen hadden plaatsgevonden gelabeld werden met een één. Vervolgens werd er voor verschillende tijdsintervallen gekeken naar de verdeling van de ratio één en nul. Om de modellen zo goed mogelijk te kunnen trainen was het van belang dat de ratio ongeveer gelijk verdeeld was, zodat de bias in het model verminderd kon worden.

Na het proberen van een aantal verschillende tijdsintervallen was voor het model dat zonder locaties is getraind een tijdsinterval van dertig minuten genomen. Voor

de andere twee modellen met locatie gebonden features gaf een tijdsinterval van vier uur de beste verdeling.

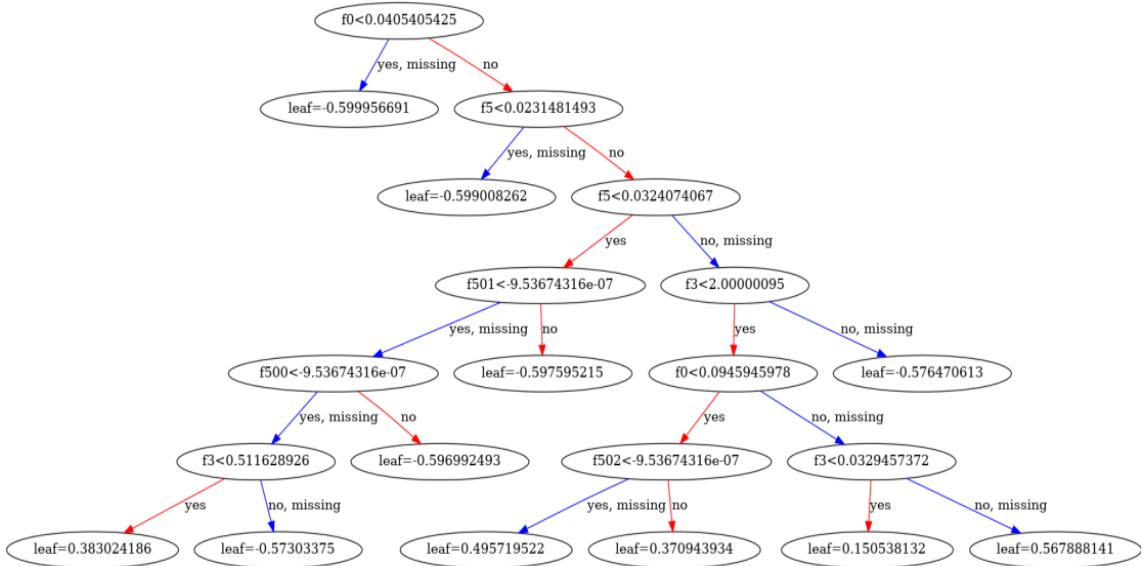
Voordat de modellen getraind konden worden op de data, werden tot slot nog de categorische variabelen omgezet naar binaire vectors. Hierbij werd gebruik gemaakt van *one-hot-encoding*. Deze stap was van belang, aangezien de XGB Classifier de categorische features niet kan meenemen in het model.

De XGB Classifier heeft een ingebouwde cross validatie functie die voorkomt dat het aantal benodigde boost-iteraties moet worden berekend. Bij het trainen van de modellen zijn er verschillende parameters die je kan meegeven, zoals *learning rate* en maximale diepte van de keuzenboom getest. Door middel van een *grid search* zijn alle verschillende combinaties van parameters getest, waarna het beste model wordt weergegeven. Het beste model heeft dus de hoogste cross validatie score. Deze score wordt weergegeven als de cross entropy loss en is een getal tussen de nul en de één. Voor deze score geldt hoe dichter bij één hoe beter de score. Een overzicht van de parameters is te vinden in tabel 1.

Tabel 1: Belangrijkste parameters voor XGB.

Parameter	Beschrijving
Gamma	Hoe groter deze waarde is, hoe conservatiever het model zal zijn.
Learning rate	Wordt gebruikt om overfitten te voorkomen.
Max depth	Hoe groter deze waarde is, hoe complexer het model wordt en hoe groter de kans is op overfitting.
Lambda	Door deze waarde te verhogen wordt het model conservatiever.
Alpha	Door deze waarde te verhogen wordt het model conservatiever.

Het XGB Classifier model produceert een beslisboom aan de hand van de input features. Dit wil zeggen dat er een tijd of een locatie in het model gestopt kan worden, waarna het model als output aan de hand van een beslisboom de kans geeft of er op die tijd of locatie een incident plaatsvindt. Deze kans wordt met behulp van een *logistic function* weergegeven als een 0 of een 1, waarbij 0 betekent dat er geen kans is op een incident en 1 betekent dat er wel een kans is. Ook is het mogelijk om alleen de probabiliteit als output te krijgen.



Figuur 2: Keuzeboom geproduceerd door XGB Classifier.

2.2.3 LSTM

Long short-term memory (LSTM) is een speciaal type *recurrent neural network* (RNN). Het is één van de meest voor de hand liggende algoritmes voor het gebruik van time-series.

Een LSTM-netwerk is een type RNN dat naast *standard units* ook *special units* gebruikt. LSTM-eenheden bevatten een *memory cell* die informatie gedurende lange tijd in het geheugen kan bewaren. Een set poorten wordt gebruikt om te bepalen wanneer informatie het geheugen binnentkomt, wanneer deze wordt uitgevoerd en wanneer de informatie wordt vergeten.

Een RNN is een uitbreiding van een conventioneel feedforward neuraal netwerk, dat in staat is om sequentie-invoer met variabele lengte te verwerken. De RNN verwerkt de reeks met variabele lengte door een terugkerende verborgen toestand (hidden state) te hebben waarvan de activering op elk moment afhankelijk is van die van de vorige keer. Deze neurale netwerken geven de output van de vorige stap als input mee aan de huidige stap. Zo hebben RNN's feedbackloops in de terugkerende laag (recurrent layer). Hierdoor kunnen ze in de loop van de tijd informatie in 'het geheugen' bewaren. Echter, is gebleken dat het moeilijk is om RNN's te trainen om langdurige afhankelijkheden vast te leggen, omdat de gradiënten ofwel verdwijnen (meestal) of exploderen (zelden, maar met ernstige gevolgen). Dit maakt de op een gradiënt gebaseerde optimalisatiemethode moeilijk, niet alleen vanwege de variaties in de gradiëntgroottes, maar omdat het effect van langetermijnafhankelijkheden verborgen is (exponentieel kleiner met betrekking tot de sequentiellengte) door het effect van kortetermijnafhankelijkheden (Chung et al., 2014).

2.2.4 Implementatie LSTM

Het LSTM-model is veelal door middel van trial en error geïmplementeerd om zo de optimale parameters te verkrijgen. Dergelijke RNN's worden getypeerd als complexe

modellen, waarbij de optimale parameters nooit op voorhand te bepalen zijn. De trial en error procedure bestond voornamelijk uit cycli waarbij telkens bepaalde parameters werden gekozen en het model met deze parameters werd getraind om steeds betere resultaten te verkrijgen. De parameters *epochs*, *lagged features*, *drop out ratio* en *batchsize* speelde de grootste rol in het optimaliseren van dit model. In tabel 2 wordt er verder uitleg gegeven over deze parameters.

Tabel 2: Belangrijkste parameters voor LSTM.

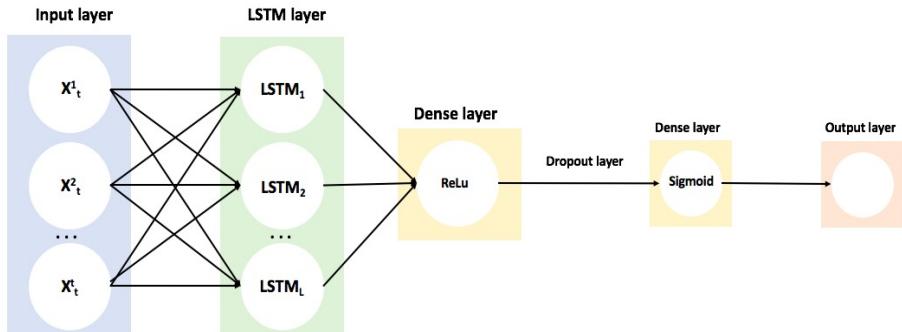
Parameter	Beschrijving
Epochs	Geeft het aantal keren aan dat het algoritme de volledige trainingsset doorwerkt. Minimaliseert de fout van het model.
Lagged features	Geeft aan hoeveel waarnemingen uit het verleden meegenomen worden voor de voorspellingen.
Dropout ratio	Zet een percentage willekeurige nodes op 0 om overfitting te voorkomen.
Batchsize	Definieert het aantal samples dat moet worden doorlopen voordat de interne modelparameters bijgewerkt worden.

Het LSTM-netwerk dat is getraind bestaat uit verschillende lagen, namelijk een input laag, een LSTM laag, twee dense lagen en een output laag.

Het LSTM laag bestaat uit 32 memory cellen. Deze cellen worden gebruikt om informatie van de lagged features mee te nemen in het creëren van een output. Er zijn verschillende aantallen lagged features gebruikt die meegenomen worden in de berekening van de output.

De volgende lagen zijn 2 dense lagen. Dit zijn lagen van nodes die diep verbonden zijn, wat betekent dat elk neuron in de dense laag de input ontvangt van alle neuronen in de vorige laag. In de dense lagen worden respectievelijk een ReLU en sigmoid als activatie functies gebruikt. De sigmoid geeft een waarde tussen de 0 en de 1. Tussen deze twee lagen in zit een ‘dropout’ laag, deze zet een percentage willekeurige nodes op 0 om overfitting te voorkomen en worden de overige nodes geforceerd beter te trainen.

Tot slot is er de output laag die aangeeft wat de kans is dat er een incident plaatsvindt in het komende half uur. Deze lagen zijn te zien in figuur 3.



Figuur 3: LSTM Netwerk.

De features die gebruikt zijn voor het LSTM-netwerk zijn gelijkwaardig aan de features van de XGB Classifier. Bovendien zijn ook de categorische features op dezelfde manier one-hot encoded. Er zijn in totaal twee LSTM-netwerken getraind. Het eerste netwerk is getraind met alle features en voor het tweede netwerk is een Chi-square test uitgevoerd om te bepalen welke van de features al dan niet afhankelijk van elkaar waren. De laagst scorende features zijn weggelaten bij het trainen van dit netwerk.

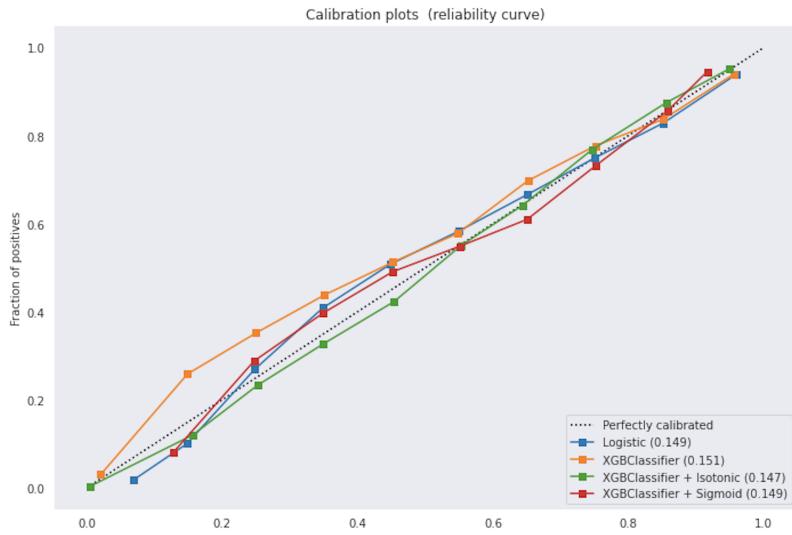
3 Resultaten

3.1 XGB Classifier

In totaal zijn er drie verschillende modellen gemaakt met de XGB Classifier. Om de modellen te kunnen analyseren is er gekeken naar een aantal aspecten, namelijk de betrouwbaarheid, de accuratie, de learning curve, de scalability en de performance van het model. Door de betrouwbaarheid van een model te analyseren kan er worden gekeken of de kansen die uit het model rollen ook daadwerkelijk eerlijk zijn. De learning curve geeft aan of het toevoegen van dezelfde soort data invloed heeft op de score van het model. De scalability en performance van het model geven de impact aan op het model als het aantal training samples groter wordt.

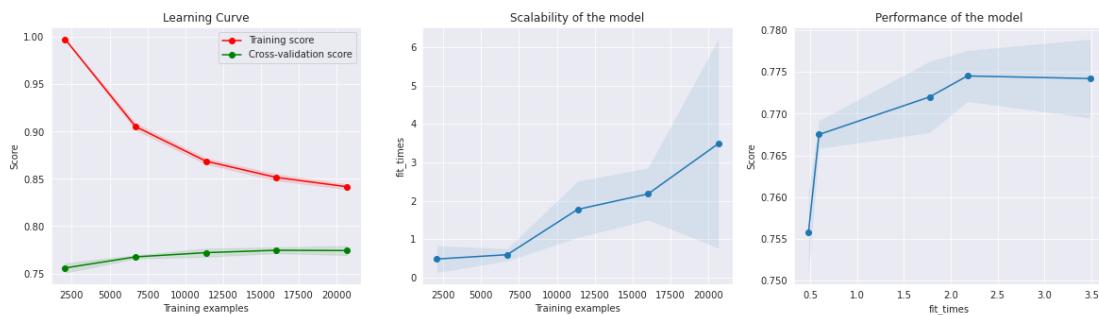
3.1.1 Model zonder locaties

Het algemene model met een tijdsinterval van 30 min is voor 83 procent accuraat op de training data en 77 procent op de test data. Om de betrouwbaarheid van de output te testen is er een kalibratie curve geplot, die te zien is in figuur 4.



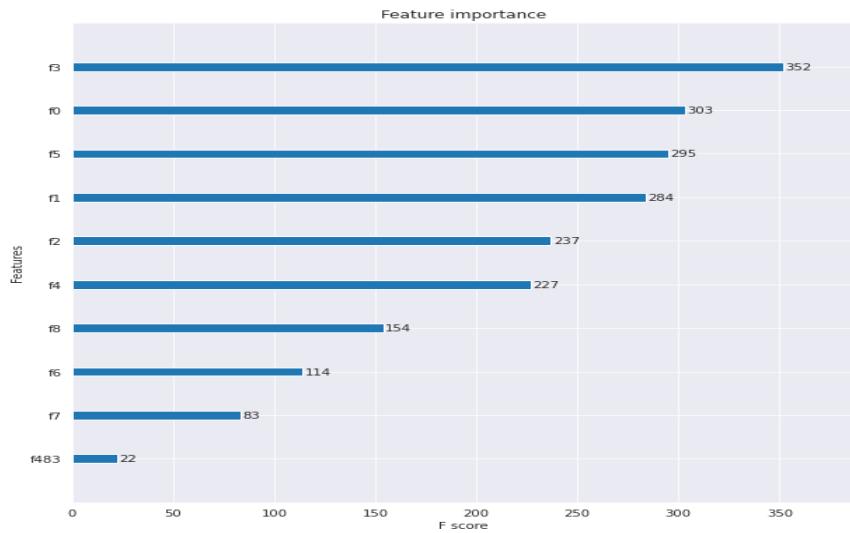
Figuur 4: Betrouwbaarheid model van het model zonder locaties.

Ook is er gekeken naar de prestaties van het algemene model, hierin is te zien dat het algemene model blijf hangen op een accuraatheid van 77 tot 79 procent. In figuur 5 zijn de prestaties te zien van het model. In de eerste grafiek zijn de training en cross validatie scores te zien over het aantal training data te zien. In de tweede grafiek is te zien hoe lang het model doet het training met als x as het aantal training data die word gebruik. In de laatste grafiek is te zien hoe het model scoort bij elke training iteratie.



Figuur 5: Prestaties van het model zonder locaties.

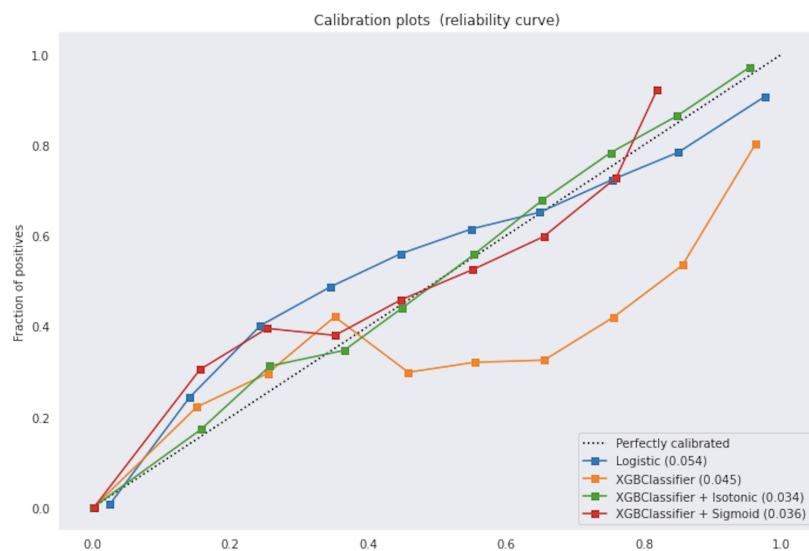
Tot slot is er gekeken naar welke features het meeste invloed hebben op de uiteindelijke kansen die het model produceert. In figuur 6 is te zien dat de features ‘lopend gemiddelde(2u)’, ‘dag van de week’, ‘ochtend’, ‘week 11 van het jaar’, ‘week 8 van het jaar’, ‘week 163 van het jaar’, ‘week 69 van het jaar’, ‘week 329 van het jaar’, ‘week 34 van het jaar’ en ‘week 32 van het jaar’ het meest invloed hebben op de uiteindelijke kansen die het algemene model produceert.



Figuur 6: Belangrijkste features van het model zonder locaties.

3.1.2 Model met wijken

Het model met wijken is bij een tijdsinterval van 4 uur voor 94 procent accuraat op de training data en 93 procent op de test data. De output van dit model is minder betrouwbaar, omdat het een grote bias heeft naar 0. Ondanks dat de wijken die een zeer klein aantal meldingen hebben uit het model zijn gelaten en er een groter interval is genomen was de ratio van 0 tot 1 nog steeds groot. Het model heeft hierdoor een bias aangeleerd die richting de 0 valt. In figuur 7 is de betrouwbaarheid te zien van de output kansen.



Figuur 7: Betrouwbaarheid van het model met wijken.

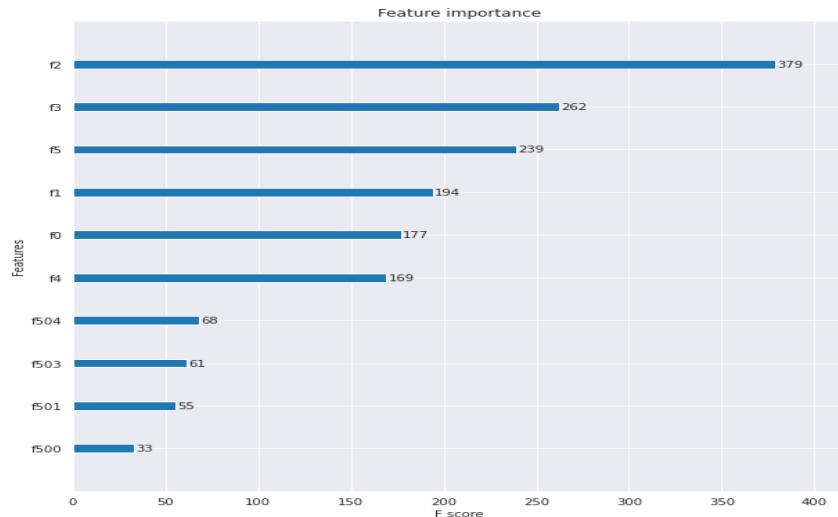
Wanneer er wordt gekeken naar de prestaties van dit model, is te zien dat het wijken model blijft hangen op een accuraatheid van 94 procent. Het gebruiken van meer

training data heeft blijkbaar niet zoveel effect op uiteindelijke accuraatheid van dit model. Het toevoegen van meer data heeft wel effect op de tijd die dit model erover doet om te trainen.



Figuur 8: Prestaties van het model met wijken.

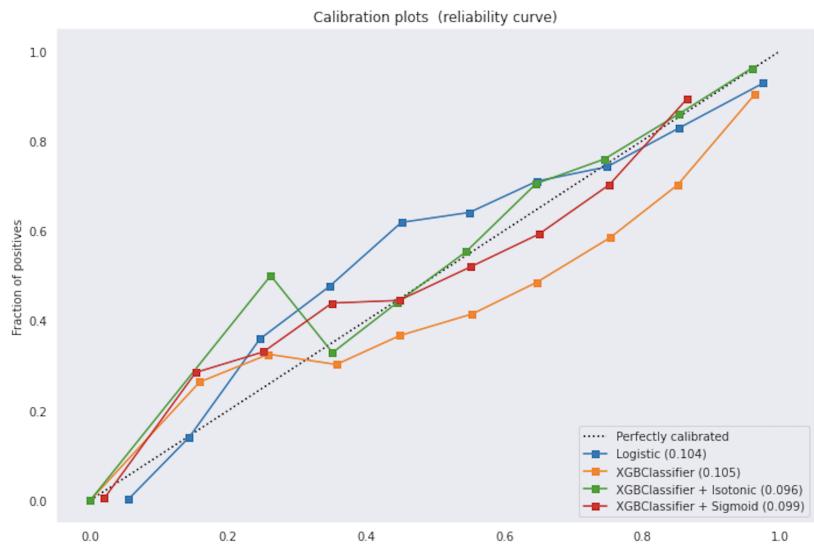
In figuur 9 is te zien dat de features ‘Lopend gemiddelde(12u)’, ‘Avond’, ‘Gemiddeld aantal meldingen;Week van het jaar, weekdag,dagdeel’, ‘middag’, ‘Werktijd’, ‘Werkdag’, ‘Week 19 van het jaar’, ‘Gemiddeld aantal meldingen;Week in het jaar, dag in de week,tijdens werktijd’, ‘Week 47 van het jaar’, ‘WK050510’(Dubbeldam, Dordrecht) het meeste invloed hebben op de uiteindelijke kansen die dit model produceert.



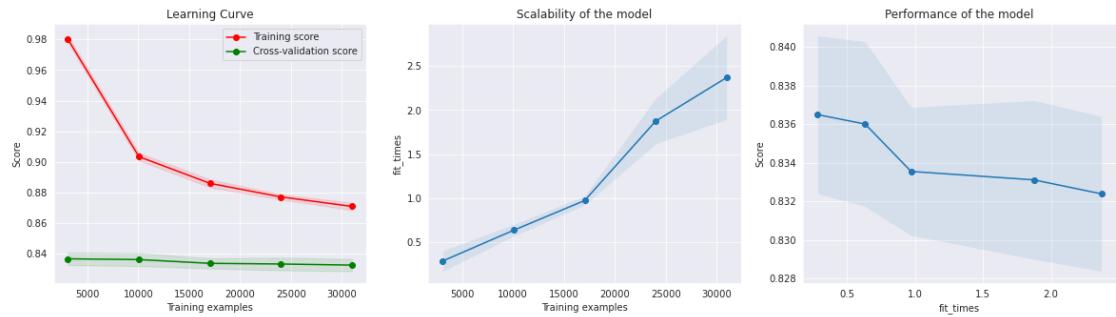
Figuur 9: Belangrijkste features van het model met wijken.

3.1.3 Model met gemeentes

Het model met de gemeentes is met een tijdsinterval van 4 uur voor 84 procent accuraat op de training data en 83 procent op de test data. Dit model heeft een veel minder grote bias richting de klasse 0 dan het model met wijk locaties, waardoor dit model robuuster is. De betrouwbaarheid van de output kansen zijn te vinden in de kalibratie plot in figuur 10.

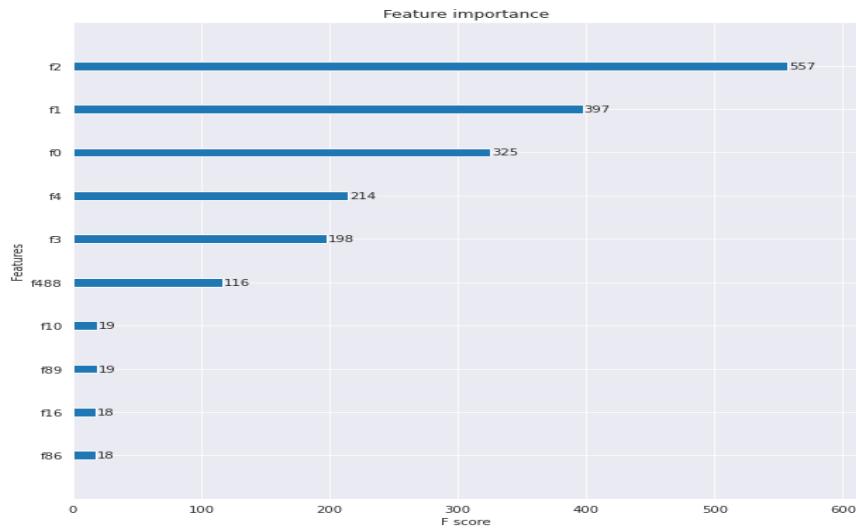


Figuur 10: Betrouwbaarheid van het model met gemeentes.



Figuur 11: Prestaties van het model met gemeentes.

De cross-validate score gaat langzaam omlaag naarmate er meer training samples worden gebruikt maar de accuraatheids score blijft redelijk stabiel. In figuur 12 is te zien dat de features ‘Lopend gemiddelde(dag)’, ‘Dag 9 van de maand’, ‘Week 33 van het jaar’, ‘Dag 11 van het jaar’, ‘maand 1 van het jaar’, ‘Dag 6 van de maand’, ‘Dag 18 van de maand’, ‘Dag 8 van het jaar’, ‘Week 21 van het jaar’ en ‘Dag 236 van het jaar’ het meeste invloed hebben op de uiteindelijke kansen die het dit model produceert.



Figuur 12: Belangrijkste features van het model met gemeentes.

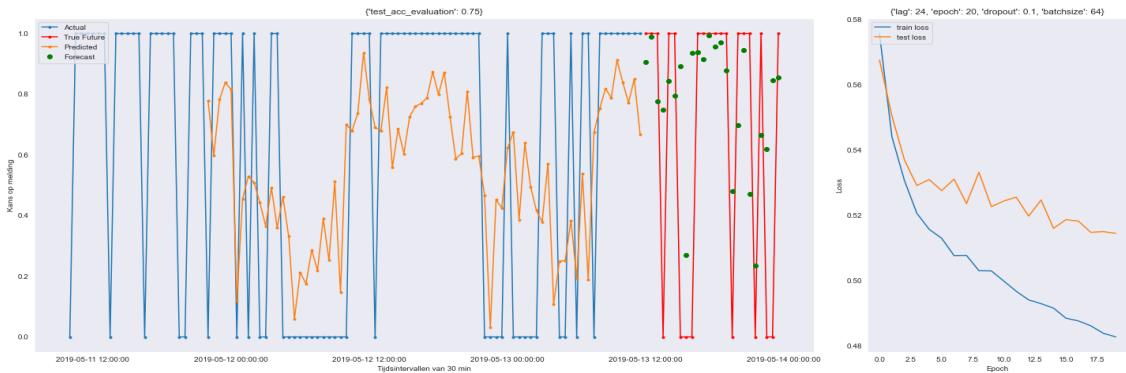
3.2 LSTM

Het LSTM-netwerk is getraind met twee modellen die gedefinieerd zijn in sectie 2.2.4. In tabel 3 zijn de verschillende scores met de verschillende parameters te zien, waarbij het model met de hoogste accuraatheid bovenaan staat. Het model met de beste scores, heeft een accuraatheid van 75 procent voor het tweede model. De hoogste scores die zijn behaald voor het eerste model lagen rond de 68 en 70 procent. Naast de accuraatheid zijn ook de *losses* van de train en testset te zien in tabel 3.

Tabel 3: Trial and error met verschillende parameters.

epoch	lag	dropout	batchsize	history-loss_-last	history-loss_-first	history-val_loss_-first	history-val_loss_-last	test_acc-evaluation
20	24	0.1	64	0.575798	0.482704	0.567696	0.514481	0.750000
20	46	0.2	256	0.592225	0.493878	0.578920	0.516336	0.748801
20	24	0.1	256	0.589105	0.490059	0.583276	0.518129	0.748113
15	24	0.1	64	0.573976	0.491433	0.559559	0.516949	0.747822
20	12	0.1	512	0.603304	0.500488	0.585254	0.519591	0.747605
...
15	12	0.2	256	0.590080	0.503330	0.579437	0.533799	0.733672
10	12	0.1	512	0.610772	0.526950	0.592797	0.538990	0.732801
10	12	0.1	256	0.587166	0.508847	0.573983	0.529590	0.732221
10	46	0.2	512	0.603215	0.530601	0.590833	0.544635	0.731222
15	12	0.2	64	0.581230	0.496452	0.564721	0.550443	0.728737
Total duration: 0 days 12:11:56.569318								

In figuur 13 zijn de voorspellingen over de laatste 124 observeringen in de testset en hoe de losses afnemen per epoch te vinden.



Figuur 13: Beste model

4 Conclusie

Het doel van dit project was om voorspellingen te maken over de kans dat er een incident plaatsvindt op een bepaalde tijd en/of een bepaalde locatie. Wanneer we de XGB Classifier en het LSTM met elkaar vergelijken is te zien dat het beste model van de XGB Classifier een accuraatheid heeft behaald van 94 procent en het LSTM heeft een accuraatheid behaald van 75 procent.

Het XGB Classifier model zonder de locaties kan aan de hand van datum en tijd een kans berekenen op een melding. Deze kans op een melding valt binnen een interval van 30 min. Door specifieker op locaties te filteren is de accuraatheid van het model toegenomen. Het toevoegen van gebieden zorgde voor een hogere accuraatheid dan het toevoegen van wijken. Echter zorgde het toevoegen van locatie er wel voor dat het tijdsinterval voor de voorspellingen groter werd.

Het LSTM maakt ook voorspellingen die gebaseerd zijn op een tijdsinterval van het komend half uur. Echter, kent dit model ook hoge losses die duiden op overfitting. Dit kan komen doordat er niet voldoende data beschikbaar was en/of dat het model te complex was voor de hoeveelheid data die beschikbaar was.

4.1 Bruikbaarheid

De voorgestelde en geïmplementeerde modellen kunnen bijdragen aan het oplossen van het probleem om tijds patronen mee te nemen in het model, omdat de XGBClassifier een goede en het LSTM een redelijke accuraatheid hebben behaald. Het laatste model, blijkt echter niet een optimaal te zijn vanwege de hoge losses die duiden op overfitting van het model. Ook worden in het LSTM geen locaties meegenomen bij het maken van de voorspellingen. Hierdoor wordt dit model ook als minder bruikbaar gezien ten opzichte van de XGBClassifier.

4.2 Mogelijke verbeteringen

Aangezien de resultaten die zijn weergegeven in dit rapport, zijn behaald in een tijdsbestek van vier weken, kan geconcludeerd worden dat mogelijke aanpassingen

een beter model kan opleveren. De volgende subsecties geven weer hoe dit model eventueel in de toekomst verbeterd kan worden.

4.2.1 Data pre-processing en analyse op basis van het gekozen model

Door in eerste instantie de data goed te analyseren en vervolgens nog een keer te gaan kijken naar welke features kunnen bijdragen aan een model, kan zowel de accuraatheid van het model verhoogd als de losses van het model gereduceerd worden. Het verschilt namelijk per model hoe en welke features een bijdrage kunnen leveren aan eerder genoemde statistieken. Zo werden er verschillen in score geconstateerd bij het testen van beide modellen met dezelfde features. Andere features voor bijvoorbeeld het LSTM leverden betere scores op dan wanneer dezelfde features werden gebruikt voor de XGBClassifier.

4.2.2 Complexiteit van het LSTM model

De complexiteit van het model beïnvloedt de eerder genoemde statistieken enorm. Zo kan in de toekomst gekeken worden of het toevoegen of verwijderen van de lagen de statistieken van het model opkrikken. Verder kan ook gekeken hoe ver er teruggekeken kan worden in de tijd met hoeveel features. Dit model was vrij complex doordat er 523 features werden meegeven aan het model, waarbij er naar 24 observeringen in het verleden werd gekeken om de kans op een melding van het komend half uur te voorspellen.

4.2.3 Meer trainingsdata

Door dezelfde modellen te trainen met meer data van bijvoorbeeld tien jaar, kunnen al snel betere resultaten behaald worden. De complexiteit van zowel het model als het gekozen algoritme hebben namelijk ook te maken met hoeveel data er beschikbaar is. Zo worden verschillende regels voorgesteld om bijvoorbeeld een x aantal data samples te hebben voor elke parameter en/of feature in het model. Daarnaast zou gekeken kunnen worden naar het toevoegen van andere soort data zoals weer, verkeer, gezondheid en welzijn gegevens van gebieden en evenementen data. Door het toevoegen van relevante data die impact kan hebben op kans van een melding binnen een gebied kan het model in theorie nog specifieker en preciezer worden.

Referenties

- Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.
- Shi, X., Wong, Y. D., Li, M. Z.-F., Palanisamy, C., & Chai, C. (2019”). A feature learning approach based on xgboost for driving assessment and risk prediction. *Accident Analysis Prevention*, 129:170 – 179.
- Yang, L., Li, Y., & Di, C. (2019). Application of xgboost in identification of power quality disturbance source of steady-state disturbance events. In *2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, pages 1–6.