Werkstuk AI - Kaggle Challenge Titanic

Academiejaar 2019-2020

Davy Van Keymeulen

Abstract

Deze paper gaat over het oplossen van de Titanic: Machine Learning from Disaster Kaggle challenge. Het doel hiervan is om te voorspellen of een passagier de titanic overleeft heeft of niet met behulp van machine learning. Via Kaggle kunnen we beschikken over 2 datasets. De train en de test dataset. De bedoeling hierbij is om een zo hoog mogelijke score te behalen. Deze kan je behalen door feature engineering (optimaliseren van de je dataset) en verschillende Methodes te gebruiken. Door verschillende Methodes te testen, is mij opgemerkt dat de scores niet ver uit elkaar liggen.

1. Introductie

Het doel van deze paper is om met behulp van machine learning te voorspellen of een passagier de titanic overleeft heeft, gebaseerd op informatie of features die kaggle me geeft. Het type van machine learning die hiervoor zal gebruikt worden is classification, specifiek binary classification. Dit is omdat een passagier het overleefd heeft ofwel niet.

Via Kaggle krijgen we twee datasets. De dataset waarmee we onze methode moeten trainen en de dataset waarmee we onze methode testen. Uiteindelijk is het dan de bedoeling dat onze methode zegt of de passagier het overleefd heeft of niet.

Voor een goed resultaat op deze Kaggle challenge is redelijk veel feature engineering nodig geweest.
Feature engineering is het maken van nieuwe features met behulp van de features die je al hebt. Ook is er heel wat normalisatie aan de pas geweest en ‘one Hot Encoding’. Deze heb ik beslist om zelf te doen, aangezien deze dan overzichtelijker is. ‘One Hot Encoding’ is een process dat ervoor zorgt dat 1 feature omgevormd wordt naar een andere vorm, zodat deze meer geoptimaliseerd is voor de machine learning Methode.

2. Data

De train-data en de test-data hebben dezelfde features buiten de survived. Dit moet dan ook voorspeld worden door de methodes.
De train-data en test-data had voor de feature engineering deze features

- PassengerId

Geen nuttige informatie.

- Survived -> niet in test-data

0 = not survived, 1 = Survived

- Pclass

Is de ticket class 1,2 en 3.
Ticket class 1 was het duurst, daaropvolgend ticket class 2 en als laatste ticket class 3 als goedkoopste.

- Name

Geen nuttige informatie, maar hier zijn met feature engineering titles uitgehaald.

- Sex

Man of vrouw.

- Age

Leeftijd

- SibSp

Het aantal broers,zussen en echtgenoot

- Parch

Het aantal ouders en kinderen dat je hebt

- Ticket -> ticketnummer

Geen nuttige informatie.

- Fare

Prijs van het ticket.

- Cabin

Welke plaats je had op de boot.
Maar te onnauwkeurig

- Embarked

plaats waar je bent opgestapt C,Q of S

Hiervan heb ik ook een aantal grafieken kunnen maken waaruit bleek dat.

- vrouwen hogere overlevingskansen hadden.
- passagiers met Pclass 1 hogere overlevingskansen hadden.
- als de passagier geen SibSp had de overlevingskans kleiner waren dan passagiers met 2 of meer SibSp
- passagiers met meer dan twee kinderen hadden een groter overlevingskans

Eerst heb ik de missing data ingevuld.Dit wil zeggen dat alleen de data waar niets instaat is veranderd.

- Age

Deze wordt pas gedaan na de feature engineering van Title

- Embarked

Ingevuld met ‘S’ omdat dit het meeste gebruikte resultaat is.

- Fare

Ingevuld met met median Fare voor elke Pclass

- Cabin

Deze wordt pas gedaan na de feature engineering van Deck

Dan heb ik als tweede stap een aantal nieuwe features geëngineerd.

- Name -> Title

Naam is omgezet naar Titles. voorbeeld Name = ‘Sir. Bruce’ wordt dan Title=’Sir’. omdat de naam onbelangrijk is maar titels hebben wel een bepaald sociaal statuut, daarom heb ik dit toegepast.
Hierna is ook de feature ‘Name’ verwijderd geweest.

- Parch & SibSp -> Familysize & Alone

Voor Familysize hebben we Parch & SibSp opgeteld en plus 1 gedaan (passagier zelf). En Alone hebben we gezien of deze persoon alleen was.

- Cabin -> Deck

Cabin is hier omgezet naar Deck omdat deze informatie beter past bij het zinken van de titanic. deze is met de achterkant naar beneden gezonken en is het dus beter om te werken met de Deck. De Deck kon je afleiden van de Cabin string. Dit is altijd het eerste letter van de Cabin

Dankzij een aantal nieuwe features kan ik nu wat missing data nauwkeuriger invullen

- Age

Ingevuld met het median Age waar de Sex, Pclass en de Title hetzelfde bij zijn.

- Deck

Ingevuld met U for unkown

Als laatste stap voor de data op te kuisen heb ik nog een aantal features verwijderd, omdat deze onbelangrijke informatie hadden.

- Ticket
- Name
- PassengerId

Hierna heb ik alles genormaliseerd en ‘One Hot Encoding’ toegepast. Dit zorgt ervoor dat je de dataset optimaliseert voor je machine learning methode.

Als laatste heb ik met behulp van RandomForestClassifier een grafiek gegenereerd die aantoont welke features het belangrijkste zijn. Hierop kan je zien

dat Fare en Age de belangrijkste features zijn.

3. Methode

Eens dat de data zo goed mogelijk was geoptimaliseerd, hebben we deze opgesplitst in de train en test data. Deze kunnen we verder gebruiken voor onze machine learning methods. Deze Kaggle challenge was gebaseerd op een classification oefening. Dus moesten mijn keuzes qua methodes voor classification werken.

De eerste machine learning method die we gebruikt hebben is de LogisticRegression Methode. LogisticRegression is een Methode die in een dataset 1 of meer onafhankelijke variabelen analyseert en zo een uitkomst gaat geven die maar 2 mogelijke uitkomsten kan hebben (0 of 1).

De tweede machine learning method is de RandomForestClassifier. Als derde hebben we de DecisionTreeClassifier. Na de DecisionTreeClassifier hebben we de KNeighborsClassifier. Deze hebben niet veel uitleg nodig, aangezien ze al veel tijdens de les besproken zijn geweest.

Als laatste hebben we een speciale machine learning method gebruikt de GradientBoostingClassifier. Deze heb ik dan ook gehouden omdat dit mij het beste resultaat gaf. De werking van deze methode is als volgt. Gradient boosting is een machine learning methode die een voorspelling gaat doen in de vorm van verschillende machine learning modellen.

4. Resultaten

Methode	Score
LogisticRegression	0.8227274967931975
RandomForest Classifier	0.8013913107621249
DecisionTreeClassifier	0.791234444303772
KNeighborsClassifier	0.6959921086238074
GradientBoosting Classifier	0.8238446024469223

5. Conclusie & Future work

Zoals je kan zien bij de resultaten heeft de GradientBoostingClassifier de hoogste score behaald met 0.8238446024469223. De scores van alle methodes staan redelijk dicht bij elkaar buiten de KNeighborsClassifier. Mijn allereerste inzending in Kaggle was alleen gebaseerd op gender. Bij deze inzending had ik een score van 0.77511. Uiteindelijk heb ik deze score kunnen verhogen door middel van feature engineering en andere Machine Learning methods te gebruiken.

Ik denk dat ik mijn score nog zou kunnen verhogen, indien ik in staat ben om een feature te maken met de exacte positie van elk persoon op het moment dat de Titanic botste met de ijsberg. Maar dit ging niet met de features die me gegeven waren van Kaggle.

6. Referenties

[1] https://www.kaggle.com/davyvankeymeulen/titanicproject/edit

[2] https://triangleinequality.wordpress.com/2013/09/08/basic-feature-engineering-with-the-titanic-data/

[3] https://www.kaggle.com/alexisbcook/titanic-tutorial

[4] https://medium.com/@praveen.orvakanti/this-will-help-you-score-95-percentile-in-the-kaggle-titanic-ml-competition-aa2b3fd1b79b

[5] https://github.com/porvakanti/Kaggle-Competition-TitanicSurvival/blob/master/Kaggle_Titanic_ML_from_Disasters.ipynb

[6] https://github.com/minsuk-heo/kaggle-titanic/blob/master/titanic-solution.ipynb

[7] https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f

[8] https://github.com/ahmedbesbes/How-to-score-0.8134-in-Titanic-Kaggle-Challenge/blob/master/article_1.ipynb

[9] https://www.kaggle.com/sashr07/kaggle-titanic-tutorial

[10] https://towardsdatascience.com/logistic-regression-for-dummies-a-detailed-explanation-9597f76edf46

[11] https://en.m.wikipedia.org/wiki/Gradient_boosting