whitehorses

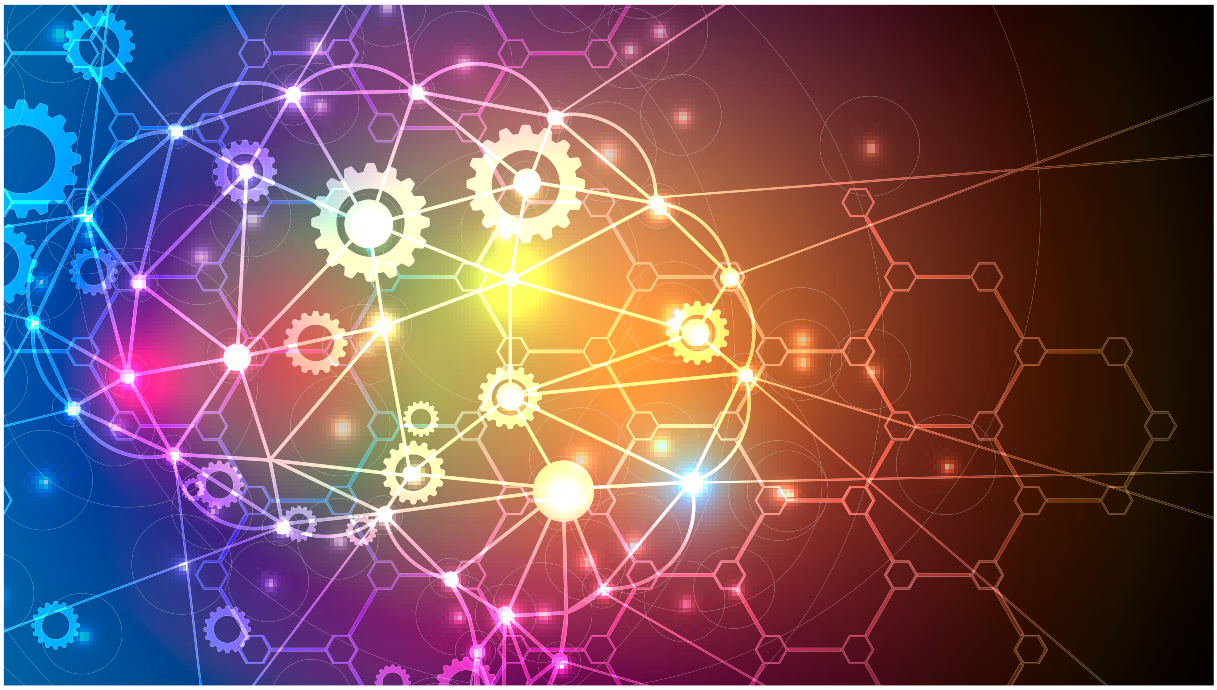
member of the human network

WHITEBOOK

Machine Learning made easy door Google

Februari 2020

Auteur:
Laurens van der Starre
INTEGRATIESPECIALIST



Introductie

Artificial Intelligence (AI) is een hot topic tegenwoordig. We zijn inmiddels al (ver) voorbij het toekomstbeeld dat werd geschetst in de science fiction films uit de 80s en 90s: *Skynet* is nog niet self-aware en er zijn voor zover ik weet nog geen *replicants* uit Blade Runner onder ons. Toch komt AI steeds vaker ter sprake in het dagelijkse leven. Door ons online bestaan is er geen plek te vinden waar geen *algoritmen* of andere *machine learning* wordt toegepast om onze digitale sporen en data te vergaren en verwerken. Data is het product. Data geeft power. Wij zijn het product.

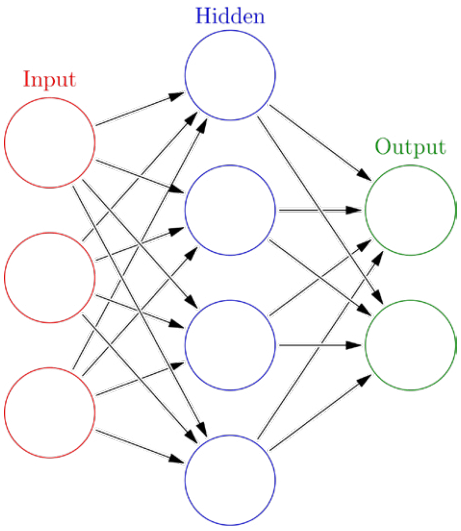
Machine learning is een onderdeel van AI. Op basis van trainingsdata bouwt het een wiskundig model (algoritme) op waarmee het beslissingen kan maken of voorspellingen kan doen. Dit allemaal zonder expliciet daarvoor geprogrammeerd te zijn. Er wordt onderscheid gemaakt in *supervised learning* en *unsupervised learning*.

In het eerste geval leert het algoritme op basis van (*trainings*)data dat zowel de input als de gewenste output bevat. Deze methode is uitermate geschikt voor het classificeren van data en het voorspellen van uitkomsten.

Aan de andere kant is er unsupervised learning. In dit geval is de gewenste output niet aanwezig in de trainingsdata. Deze methode wordt gebruikt om structuur in data te herkennen.

Machine learning met neurale netwerken

In dit Whitebook gaan we supervised learning toepassen met een neuraal netwerk. Een (kunstmatig) neuraal netwerk is losjes gebaseerd op een biologisch neuraal netwerk in het brein van dieren. Een neuraal netwerk bevat neuronen, die onderling connecties met elkaar hebben. In dit geval van supervised learning lopen de connecties van de input door één of meerdere lagen met neuronen naar uiteindelijk de output neuronen. De neuronen tussen de input - en de output neuronen worden ook wel verborgen (*hidden*) neuronen genoemd.

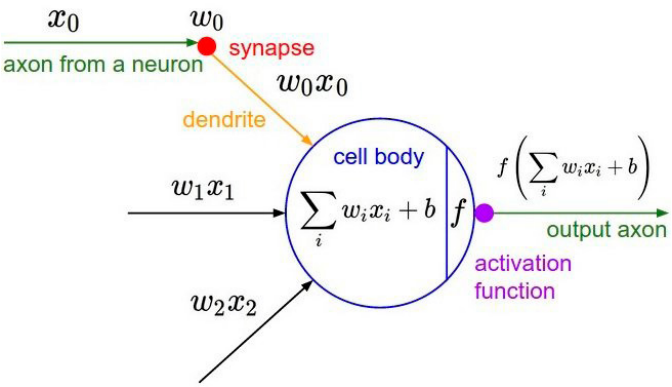


Figuur 1 Neuraal netwerk met verborgen neuronen

Neuronen hebben een state. Ze krijgen input (*features*). Een set van input features wordt daarom de *feature vector* genoemd. Neuronen kunnen door features geactiveerd worden als een bepaalde drempelwaarde wordt overschreden (de *threshold*). Een *activation* of *transfer function* manipuleert vervolgens de verkregen waarde van de input. De output van deze neuron wordt vervolgens via één of meerdere connectors doorgegeven aan de volgende neuron.

De eerdergenoemde connectors hebben een gewicht (*weight*) die meewegen in de output die over de connector wordt doorgegeven. Een hoger gewicht maakt die connector dus belangrijker, terwijl een laag gewicht de impact van de connector vermindert. Een weight is dus een soort van multiplier van de waarde die over de connector wordt doorgegeven.
De ontvangende neuron combineert uiteindelijk alle binnenkomende waarden en gewichten als zijn eigen input. Zie figuur Figuur 2 (ontleend uit [2] en [3]).

Er zijn verschillende types activation functions met elk specifieke eigenschappen. Ze bepalen de werking van het netwerk en elk heeft z'n eigen toepassingsgebied. De verschillende activation functions vallen buiten de scope van dit Whitebook.



Figuur 2 Kunstmatige neuron

Een neuraal netwerk leert door tijdens het trainen de gewichten en thresholds aan te passen m.b.v. een zgn. *cost function*. Het doel is de verwachte uitkomst tegenover de daadwerkelijke uitkomst van het netwerk zo dichtmogelijk naar elkaar te krijgen. De cost function is een wiskundig principe waarmee uiteindelijk het verschil tussen de uitkomst van het netwerk en de verwachte waarde (uit de trainingsdata) wordt geminimaliseerd.

Sommige neuronen hebben geen uitgaande connectors. Dit is dus de output neuron. Deze zgn. *output vector* bevat het resultaat van het netwerk en bestaat uit één of meerdere output neuronen.

Het trainen van een neuraal netwerk is het constant aanpassen van de gewichten van connectoren en de thresholds van neuronen, om zo tot een laag mogelijk foutpercentage te komen.

TenforFlow en AutoML

Het trainen van neurale netwerken is lastig. Niet alleen is het aantal lagen (en dus ook het daadwerkelijke aantal verborgen neuronen) van groot belang, maar ook de activation en cost function. En dan is er nog de input data voor de feature vector die wellicht nog wat moet worden bijgeschaafd. Het is letterlijk hogere wiskunde en vergt een grote rekenkracht.
Dat Google al tijden bezig is met machine learning en AI zal niemand verbazen. Uiteindelijk heeft dit geresulteerd in TenforFlow: een opensource bibliotheek dat gebruikt kan worden voor het maken, trainen en gebruiken van neurale netwerken. Google zal Google niet zijn als ze niet ook een zeer toegankelijke cloud oplossing voor AI zouden hebben: AutoML. Google heeft zelfs een Tensor Processing Unit (TPU) ASIC (Application Specific Integrated Circuit) ontwikkeld expliciet voor machine learning. Dit stukje elektronica kan razendsnel werken met TensorFlow models.

AutoML neemt heel veel werk uit handen. Het verbetert en prepareert de feature data, en traint het neurale netwerk (in werkelijkheid een TensorFlow *model*) op verschillende manieren met verschillende methoden op de gigantische TPU-cloud van de Googleplex. Zo vindt Google het beste model passend op de feature data. Het is bijna een AI die een neuraal netwerk creëert.

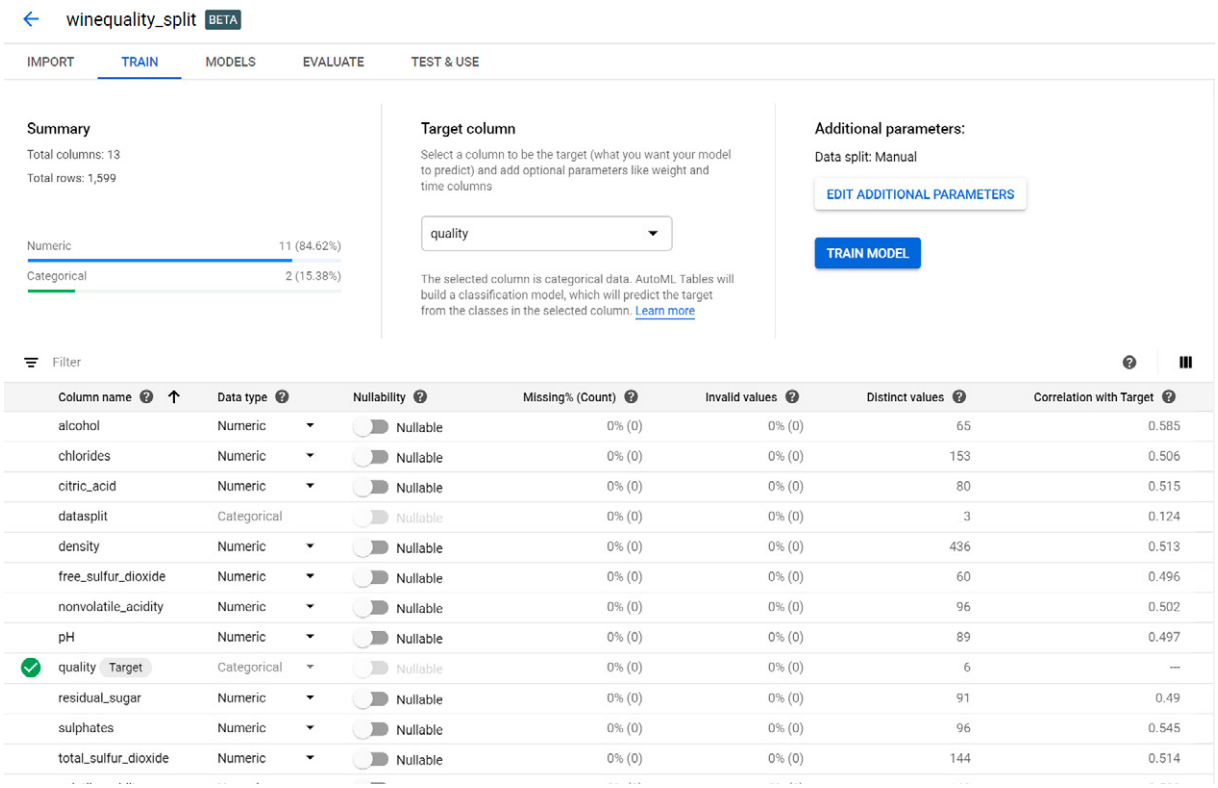
Casus

Stel er is een hypothetische wijnbottelaar voor rode wijnen die ingekochte rode wijn bottelt van diverse producenten. Er wordt een label op geplakt en de wijn wordt in een bepaalde prijsklasse gepositioneerd voor in de schappen van de welbekende (budget) supermarktketens. Je kan al de aangeleverde wijn proeven en vervolgens bepalen of het in een literpak moet worden verpakt, of in een mooie fles voor dubbel de prijs. Door dit een tijd te doen, en de samenstelling van de wijnen uiteen te zetten tegen de kwaliteitsklasse ontstaat er een mooie dataset [1]. Deze dataset bevat metingen van o.a. diverse zuren, suikers, pH-waarden, sulfaten en alcohol, en bevat uiteraard een classificatie van de kwaliteit (een geclassificeerde waarde van 3 t/m 8).

Google AutoML Tables

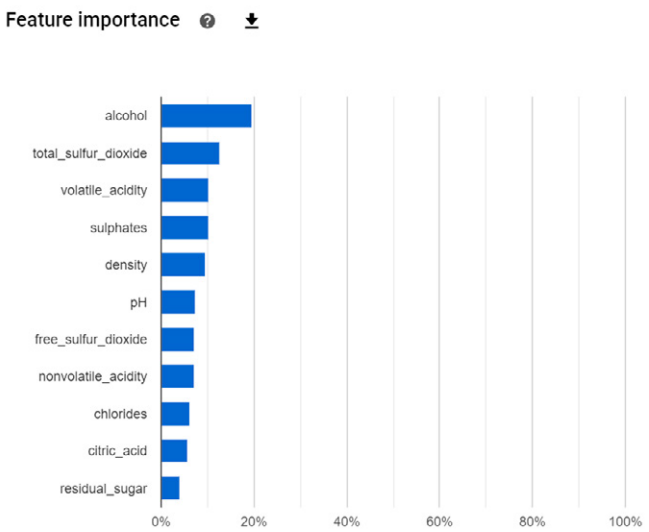
In Google's cloud kan deze data als een simpele comma seperated file (CSV) worden ingelezen. AutoML split deze data automatisch in meerdere datasets voor het trainen, valideren en testen van het te trainen model. Indien nodig kan deze datasplit ook zelf worden opgezet door per record in de CSV aan te geven tot welke dataset het record behoort. In het onderstaande voorbeeld is dit gedaan in de kolom datasplit. Op deze manier is handmatig de dataset opgedeeld in een set voor trainen, testen en valideren om zo een evenwichtige verdeling van de dataset te creëren.

Na het importeren van de dataset kan de target (output) kolom worden gekozen (in dit geval quality):



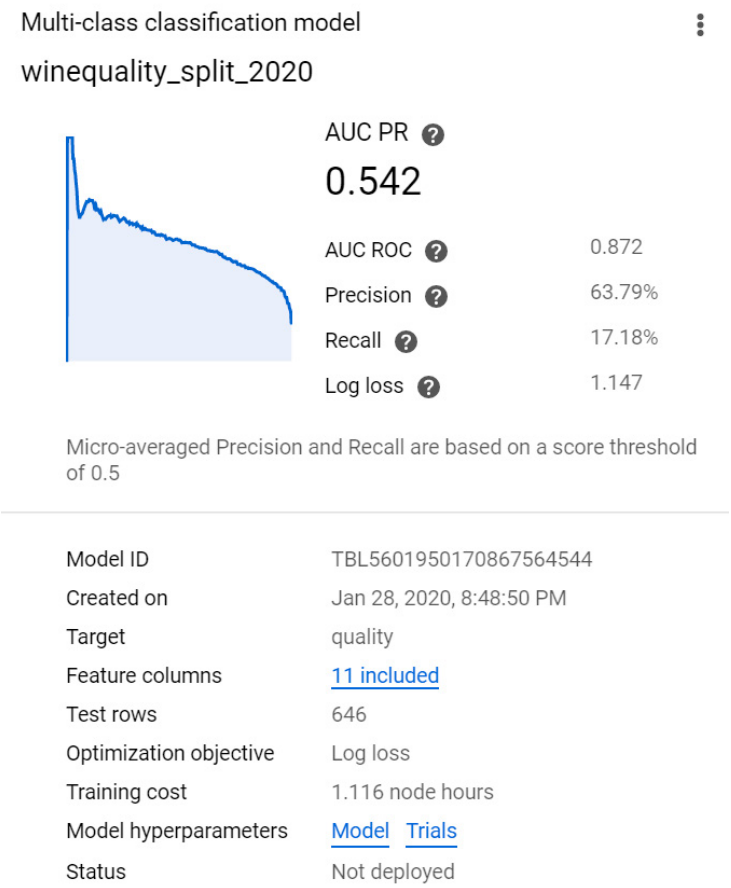
Figuur 3 Dataset

In de bovenstaande figuur is te zien dat de quality target kolom een gecategoriseerde waarde heeft. Tevens zijn de overige kolommen uiteengezet tegen de target en is te zien hoe deze van invloed is op het resultaat (Correlation with Target en Feature Importance).



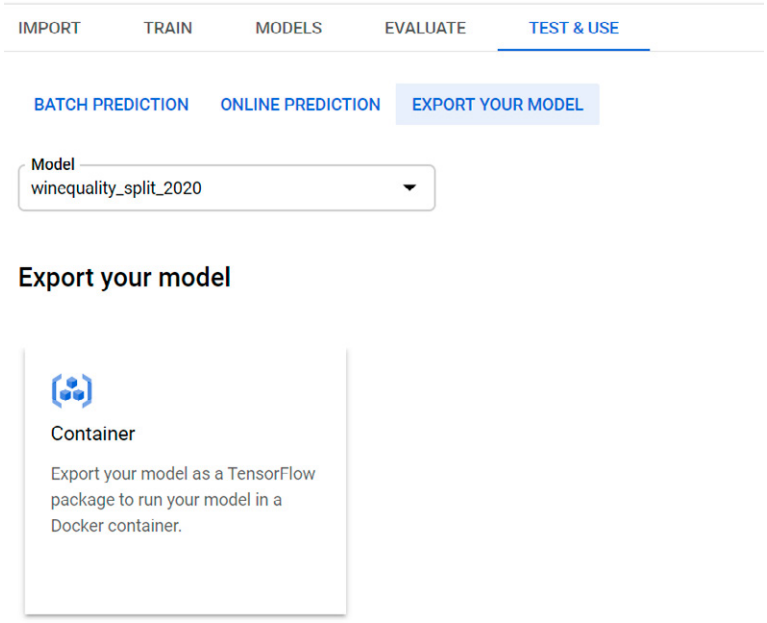
Figuur 4 Feature importance

Vervolgens kan AutoML het model gaan trainen. Het trainen duurde een kleine twee uur en kostte in totaal 1116 compute uren op de TPU compute nodes. Er zijn dus een dikke 500 TPU's bijna twee uur in de weer geweest. Het resultaat is een model, die met de (redelijk beperkte set van [1]), een precisie van 63% heeft.



Figuur 5 Model evaluation

Dit model zou beter kunnen presteren als er meer feature data zou zijn, en een betere verdeling van de training-, valideer- en testset zou worden gemaakt.
Het model dat uiteindelijk is getraind, kan worden deployed in de Google Cloud. Via een REST API call kunnen er één of meerdere feature data records worden aangeboden waarna de voorspelling van de target kolom wordt geretourneerd. Wat echter interessanter is, is dat het model kan worden geëxporteerd als TensorFlow model die in een Docker container kan worden gedraaid. Op deze manier kan je offline via REST het model aanspreken.



Figuur 6 Exporten van het model

Deze container is te downloaden uit de Cloud-storage via de Google cloud toolkit:

```
gsutil cp -r gs://target/* ./download_dir
```

(In de target locatie staat het geëxporteerde model, die vervolgens wordt gedownload naar de download_dir).

Vervolgens moeten we een Google AutoML TensorFlow Docker container image downloaden:

```
sudo docker pull gcr.io/cloud-automl-tables-public/model_server
```

In deze casus wordt met

```
docker run -v `pwd`/model-export/tbl/winequality:/models/default/0000001
-p 8080:8080 -it gcr.io/cloud-automl-tables-public/model_server
```

het model “winequality”, versie 00001, geladen. Tevens specificeren we de HTTP port 8080 voor het ontvangen van de REST calls.

Nu kunnen we het netwerk gebruiken. De feature vector zullen we als een JSON-payload aan bieden:

```
{
    “instances”: [{
            “nonvolatile_acidity”: 10,
            “volatile_acidity”: 0.5,
            “citric_acid”: 0,
            “residual_sugar”: 2.0,
            “chlorides”: 1,
            “free_sulfur_dioxide”: 15,
            “density”: 34,
            “pH”: 0.9978,
            “sulphates”: 1.9,
            “alcohol”: 0.75
        }
    ]
}
```

Kan met een simpel CURL-commando input deze “input.json” worden aangeboden

```
curl -X POST --data @input.json http://localhost:8080/predict
```

Om zo een voorspelling te doen van de wijnkwaliteit:

```
{
    “predictions”: [{
            “classes”: [“5”, “6”, “7”, “4”, “8”, “3”],
            “scores”: [0.5600000023841858, 0.1599999964237213, 0.0,
0.03999999910593033, 0.0, 0.23999999463558197]
        }
    ]
}
```

Het model voorspelt dat de rode wijn met de ingestuurde eigenschappen waarschijnlijk een kwaliteit van 5 heeft (56%) met als runner up een 3 (23%). Een gemiddeld wijntje dus.

Conclusie

Voor dit Whitebook heb ik mijn lesmateriaal van het vak Neural Networks van de Rijks*universiteit* Groningen er weer eens bij gepakt. Ik stond versteld van het feit dat ik er niks meer van snapte... Een Neuraal Netwerk is letterlijk hogere wiskunde. Om een netwerk goed op te zetten, en goed te trainen is het noodzakelijk dat je goed weet hoe zo'n netwerk werkt, wat de activation en cost functions doen, en wat de invloed is van de hidden neuronen. Handmatig een netwerk maken met TensorFlow of andere libraries (zoals Deeplearning4J) is niet iets wat je zo maar even oppakt en waarmee je een goed neuraal netwerk traint als je alleen maar de syntax kent. Google's AutoML is dus een uitkomst. Deze cloud dienst neemt heel veel werk uit handen, en is verrassend slim (en dus verrassend *scary*). Het lijkt wel een AI die je helpt een AI te maken. Google AutoML helpt je een goed TensorFlow model te trainen, echter ben je zelf nog niet overbodig. Het is essentieel goede feature data aan te leveren, en goed te werken met de analyse die wordt gemaakt van de feature data. Als bijvoorbeeld een feature niet bijdraagt aan het resultaat, dan is het wellicht beter deze niet mee te nemen in de training. Ook een eigen datasplit maken kan helpen om een goede verdeling te maken van de train, test en validate datasets, met name als de output vector geclassificeerde data bevat.

Al met al is AutoML een geweldige toolkit om je eigen neural netwerk te trainen en TensorFlow model te maken. De toepassingen zijn eindeloos.

Referenties:

1. Dataset: https://github.com/tirthajyoti/Machine-Learning-with-Python/blob/master/Datasets/winequality-red.csvÿÿÿÿÿl
2. https://mc.ai/everything-you-need-to-know-about-activation-functions-in-deep-learning-models/
3. https://www.researchgate.net/figure/Activation-function-of-neuron_fig1_329799984