Werkt het eigenlijk wel?

Rotterdanse Bijstandsfraude-software onder de loep

Jan van Rongen

2022-01-06

0. Inleiding

Op 18 december 2021 puliceerde Argos de code van de Bijstandsfraude-detectie software van de gemeente Rotterdam.

Omdat dit het soort software is dat ik tussen 2010 en 2018 zeer intensief heb gemaakt besloot ik er een blik op te werpen.

Er ging vrijwel meteen een alarmbel rinkelen: het model deugt niet. Dat was aanleiding om Argos te mailen en een begeleidend memo te maken met meer toelichting. [1].

Wat deugt er dan niet aan ?

1. Het model deugt zelfs al theoretisch niet

Het model wordt gemaakt met behulp van een historische data-set waarin 60% fraudeurs zijn geïdentificeerd en dus 40% niet-fraudeurs. In de feitelijke populatie waarop het model dan wordt losgelaten, is het daadwerkelijke percentage fraudeurs veel kleiner, zelfs 10% lijkt overdreven.

Ik noem dat maar even de `prevalentie-bias`: dat er wordt gewerkt met een data-set waarin één klasse zwaar oververtegenwoordigd is vergeleken met de "populatie". Ofwel het model dat wordt gemaaakt "weet" veel te weinig van de andere groep en daardoor zal die vaak verkeerd worden ingedeeld. In gewoon Nederlands: veel te veel mensen krijgen het label fraudeur terwijl ze dat niet zijn.

Hey heel simpel: als een model wordt getrained op basis van 60/40, zal het ook uitgaan van die verdeling - tenzij er in de software voor wordt gecorrigeeerd.

Er is in de door Argos opgevraagde documenten één aanwijzing dat iemand zich bewust was van dit probllem [2]. Echter, diens analyse is lang niet juist en daardoor is er ook geen opvolging gekomen: het probleem wordt niet duidelijk genoeg gemaakt.

De gebruikte algoritmes komen uit het domein van `statistical learning` [3] . Het probleem van klassificatie met ongebalanceerde data is uiteraard wel bekend. Dus zijn ook oplossingen voor bedacht, maar die komen niet voor in deze software.

Daarmee is de constatering gerechtvaardigd dat een op deze wijze geproduceerd model alleen al op theoretische gronden fout is.

2. De code

De software is geschreven in R, een statistische programmeertaal [7]. Binnen R zijn er veel pakketten beschikbaar om statistische modellen te maken en te te beoordelen. De hier gebruikte modellen komen allemaal uit de hoek van _Statistical learning___[3], Ook een specialisme dat de nodige studie vraagt.

Volgens de code hebben minimaal 5 medewerkers frequent aan deze software gewerkt. Het leek me onmogelijk dat er genoeg medewkers van de gemeente Rotterdam deze complexe materie zouden beheersen. Dat klopt, want de software is door Accenture gemaakt (volgens Argos).

Op het eerste gezicht is de software goed geschreven en gestructureerd, maar bij nadere inspectie valt gebrek aan ervaring met _Statisticall Learning___ toch wel op.

Van een externe partij mag je verwachten dat ze over voldoende kennis en ervaring beschikken, maar dat is hier toch echt de vraag. Vooral het niet corrigeren voor de prevalentie-bias is een blunder van de eerste orde die eigenlijk alleen maar door gebrek aan ervaring kan worden verklaard.

2.1 De modellen en ongebalanceerde data

In de code worden 5 soorten modellen getrained en beoordeeld om te kijken welke het beste model levert. Die modellen worden in de trainingsfse "getuned" al naar gelang de mogelijkheden van het soort model..

Deze zijn : `glmnet`, `gbm`, `xgboost`, `rf` (= Random Forest) en `rpart`. Het opmerkelijke is dat elk van die modeeln voorzieningen kent om met ongebalanceerde data om te gaan maar dat die in geen enkel geval zijn gebruikt.

`glmnet`, `gbm`, `rpart` hebben de mogelijkheid om met de `weights` parameter te proberen de prevalentie-bias weg te werken.

`rf` kent een expliciete parameter `classwt` voor de gewenste prevalentie.

`xgboost` kent beide manieren om de prevalentie-bias te mitigeren. Ook hier kan een gewicht per observatie worden meegegeven. Daarnaast is er de parameter `scale_pos_weight` die precies doet wat je zou willen: de werkelijke verdeling van de beide klassen gebruiken. Uit mijn simulaties blijkt hoe goed dat werkt.

Maar waarom hebben ze dat niet gebruikt? Deels, denk ik, omdat ze de tuning aan het pakket `caret` overlaten. Dat maakt het allemaal wat makkelijker om de software te maken maar helaas wordt het zo amateuristisch gebruikt dat het kind met het badwater wordt weggegooid. [4]

2.2 Trainen met het `caret` pakket

Het goed tunen van modellen is een langdurige zaak die je graag zoveel mogelijk zou willen over laten aan een sotware-pakket. En dat gebeurt: alle modellen worden getuned me`caret` - dat staat voor **Classification and Regression Training**.

Dat is mooi, want dan valt in een audit te zien hoe dat gebeurt, maar tegelijk is het een soort eenheidsworst waardoor veel van de mogelijkheden die fe afzonderlijke modellen hebben, niet bereikbaar zijn. [5]

Helaas worden alle parameters via de "grid" doorgegeven. Op die manierZo is het echter niet mogelijk de genoemde parameter `classwt` van RandomForest aan `caret`door te geven, noch de `scale_pos_weight` van xgboost. **weights** zijn er wel. Maar zijn niet gebruikt dus.

2.3 De merlwaardige tuning van xgboost

Xgboost kan redelijk goed getuned worden met `caret`. Maar dan moet je wel al een beetje weten wat je doet, Die kennisnis lijkt onvoldoende aanwezig. Of wilde men met opzet dit pakket niet mee laten doen?

In ieder geval wordt een merkwaardige combinatie van parameters uitgeprobeerd. `Gamma` =1 zorgt voor een moeilijker splisting van knooppunten en dat is tegelijkertijd in strijd met het werken met een beperkte diepte (van 1:3). Heeft men over het hoofd gezien dat de default voor gamma 0 is? En warom is`max_tree_depth` hooguit 3? Vrijwel altijd is een hogere waarde veel beter.

Die vragen kunnen alleen worden beantwoord met de originele test data, en die heb ik niet.

Maar niet gek dus dat xgboost niet als beste uit de bus kwam.

3. Is dit “proven technology”?

On de privacy impact assessment [6] wordt bij punt 16 gesteld dat er geen sprake is van nieuwe technolgie. Dat zou je formeel kunnen stellen:

(a) de belangrijkste algoritmes dateren uit de periode 2000-2010.

(b) de programmertaal R is ook al zo’n 20 jaar oud.

R is echter een programmeertaal die alleen door statistici werd gebruikt. Het vraagt heel sterke statistische kennis om er goed mee om te gaan. Een goede data-scientist heeft die kennis, maar de gemiddelde programmeur zeker niet.

Verder moet worden opgemerkt dat de algoritmen eerst in de wiskundige literatuur worden gepubliceerd en dat er daar nog volop nieuwe ontwikkelingen zijn. De software is vaak pas een paar jaar later “rijp” genoeg om makkelijk te gebruiken.

3.1 Hoe kunne we begrijpen wat het model doet?

Vier van de 5 pakettenen die in deze software worden vergeleken maken een model dat uit duizenden beslisbomen bestaat die op een bepaalde manier met elkaar samenhangen. Dat het werkt kunnen we meten (en er is wiskundig bewijs voor), maar wat er dan uit komt valt zelfs door kenners niet te interpreteren.

Datzelfde probleem speelt misschien nog wel sterker bij neurale netwerken.

Het gevolg van het niet kunnen begrijpen zijn foute interpretaties. Zo is er in Rotterdam besloten om bepaalde kenmerken niet te gebruiken omdat de disciminerend zouden zijn. Feit is dat in de “state of the art” van de Rotterdanmse software nog helemaal niet kon worden gezien of een kenmerk positief of negatief werd gebruikt. De software probeert immers gewoon de “objecten” zo goed mogelijk in een klasse te plaatsen en belangrijke kenmerken dragen vaak aan beide klassen bij.

Na 2017 waarin deze Rotterdamse software is gemaakt is er veel onderzoek gedaan om begrijpelijker verklaringen van modellen te maken.

Inmiddels is dat onderzoek vertaald naar software pakketten die sinds 2020 beschikbaar zijn. Bijvoorbeeld in R het pakket `shapr`.

4. Noten

[1] Toelichtend memo voor Argos: argos/comment01.pdf

[2] Achtergrond_Rotterdam/controle_model_ANON.html

[3] “The element of statistical learning” is het standaardwerk dat ook vrij beschikbaar is via de eerste auteur Trevor Hastie: https://hastie.su.domains/ElemStatLearn/

[4] `Caret` kent meerdere manieren om parameters door te geven. De documentatie is echter niet altijd even duidelijk en de foutmeldingen die `caret` geeft zijn vaak onbegrijpelijk. Er is een behoorlijke leercurve voordat alle ins and outs zijn doorgrond.

[5] `xgboost` kent ook andere parameters die je zou willen tunen maar die `caret` niet wil doen.. Bijvoorbeeld `alpha` en `lamba` die dezelfde functie hebben als in `glmnet` waar dat dan weer wel kan.

[6] PIA: Achtergrond_Rotterdam/2017020 Privacy Impact Assessment pilotfase Project Uitkeringsonrechtmatigheid.pdf

[7] https://cran.r-project.org/