Correcte algoritmes verkeerd gebruikt

Fraude-bijstand; software Rotterdam

Jan van Rongen

2021-12-26 14:12:16

Inleiding

De Rotterdamse fraude-software staat ter discussie, met name of bepaalde kenmerken mogen worden meegenomen in het model dat fraude tracht te voorspellen. Bijzonder is dat de code openbaar is gemaakt,. Niettemin heeft niemand de kwaliteit van de software zelf beoordeeld. De essentiële vraag: *levert deze software een bruikbaar (valide) model* is daarmee dan ook in het geheel niet gesteld, laat staan beantwoord

In dit memo stel ik dat de software weliswaaar op het eerste gezicht keurig de state-of-the-art volgt, maar dat er een serieuze en fundamentele (maar ook elementaire) fout is gemaakt waardoor de resultaten waardeloos zijn en geen enkele voorspellende waarde hebben.

Tevens vrees ik dat de software (mede) door een extern bedrijf is gemaakt en dat die deze zal proberen te verkopen aan andere gemeenten (met dezelfde schrijnende gevolgen voor de bijstandstrekkers). .

Karakteristieken van de code (= software)

De code is opgebouwd volgens een standaard *statistical learning* model: *supervised learning*. Het is geschreven in de (gespecialiseerde) programmeertaal R, een Open Source raamwerk voor het maken en doorrekenen van statistische modellen.

De code ziet er goed gestructureerd en leesbaar uit, het is makkelijk te volgen voor ingewijden met goede kennis van R, van statistiek en in dit geval ook nog van statistical learning. In totaal hebben zeker 5 medewerkers aan de code bijgedragen sinds 2016. Het lijkt me zeer onwaarschijnlijk dar de gemeente Rotterdam zoveel gespecialiseerde kennis in huis heeft.

Wat is *Supervised learning*?

De software is gebaseerd op het algemeen gehanteerde model van supervised learning. In dit geval gaat het tevens om classificatie: de vraag of een "object" tot een bepaalde klasse behoort.

"Objecten" zijn in dit geval gewoon records van personen in een bestand met heel veel van hen bekende of onbekende eigenschappen (die in het jargon "features" heten).

Van de meeste objecten is de classificatie onbekend, bij supervised learning neemt men een (historische) deelverzameling waarvan de klassificatie wel bekend is.

De leerfase: leren van representatieve gegevens.

In de "leerfase" moet een classificatie-model worden gemaakt waarmee de nieuwe "objecten" toch zo goed mogelijk kunnen worden geclassificeerd. Er zijn meer methodes om dat te doen, maar het gebruik van een test-set maakt het meest dudidelijk waar het aan schort.

Je kunt deze fase beschouwen als een lab met heel veel experimenten op een mini-populatie waarbij de supervisor er voor zorgt dat er niet te veel gespiekt wordt.

De verzameling objecten waarvan de classificatie wèl bekend is, wordt gebruikt. Die wordt (at random) verdeeld tussen een modelleur en een supervisor. Die sets heten in het jargon “train” en “test”. De modelleur kent de test-set niet.

De modelleur probeert classificatie-modellen uit m.b.v. de train set en laat ze beoordelen door de supervisor.

Die supervisor meldt dan in één getal wat de afwijking is tussen het model en de werkelijke classificatie. De maatstaf daarvoor is de zogeheten “loss function” die modelleur en supervisor onderling hebben afgesproken.

De modelleur kan duizenden modellen proberen: computers zijn geduldig en snel. Maar uiteindelijk wil men natuurlijk een model hebben dat ook goed scoort op alle werkelijke data. En daarom moet die train- en vooral de test-set qua samenstelling een zo getrouw mogelijke afspiegeling zijn van de werkelijkheid. Maar dat zijn ze hier in het geheel niett.

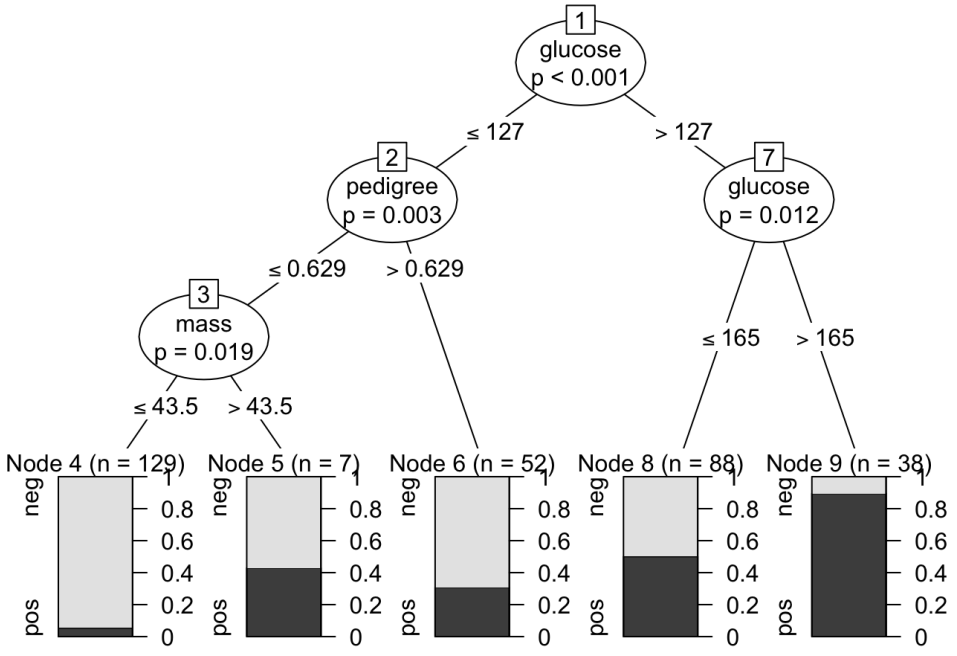
Wat men dus fout doet.

In deze Rotterdamse software is de data voor de modelleur (“train”) en die van de supervisor (“test”) afgeleid van vermoedens van oude fraude De data betreft onderzoeken waarvan ongeveer 60% fraudegevallen waren en de rest niet. De werkelijkheid is eerder 0,25% versus 99.75%. En dat betekent dus dat de supervisor niet goed kan werken, want hij begint al met een aanname van 60% kans op schuldig zijn in plaats van minder dan 1%. De gegevens zijn te beperkt m/b/t/ niet-fraudeurs en derhalve zijn alle modellen die op deze wijze worden gemaakt slecht in het herkennen van niet-fraudeurs.

Beslisbomen en de prior class probablity.

Nu wordt het even technisch, maar 4 van de 5 modellen die men heeft uitprobeerd om de beste te vinden zijn gebaseerd op (binaire) beslisbomen.

Wat is dat? Neem als voorbeeld de volgende beslisboom (afkomstig uit een voorbeeld in een van de R-paketten) naar de kans dat iemand diabetes heeft.



Er zijn dus drie “features”, glucose-gehalte, aanleg en BMI. Je begint bovenin en elk object valt uiteindelijk in een van de 5 mandjesonderine met ingeschatte kans op diabetes van 0.01 t/m o.9. Er staat zelfs bij hoeveel

van de objecten in die mandjes vallen. Dat is precies de situatie in een leerfase, waarin de prognose wordt vergeleken met wat we feitelijk weten van de testgroep: de "prior class probability" van 60%.

Omdat de testgroep een heel ander profiel heeft dan de feitelijke populatie, worden er dus ook heel andere beslisbomen goed gevonden dan wanneer er veel meer informatie beschijbaar zou zijn over die andere klasse en zou dus ook een veel verfijndere strutuur zijn ontstaan om om niet-fraudeurs te herkennen.

Samenvatting Conclusies

1. Trainen met data met een totaal andere prior class probability zonder verdere voorzorg is simpelweg heel erg fout, want gaat in tegen de gedachte achter supervised learning.
2. Men lijkt zich daar vaag van bewust, maar het wordt niet goed onderzocht en er is geen enkele stap richting een oplossing.

De gevolgen daarvan

Lopen we nog even door een ander document dan zien we dat de omvan van train 9934 objecten is en van test 2482. Dus van de totaal beschikbare al geklassificeerde data gaat 75% naar de modelleur. Omdat er toch ergens een vermoeden is dat er een onbalans is, is er ook nog een proef met een alternatieve splitsing van 11026 objecten in train en 1390 in test. Zie file **controle_model_ANON.html**. De tekst is wat onduidelijk, maar het lijkt er op dat men een gebalanceerde test set heeft proberen te maken.

En daarmee daalt de gekozen "loss functie" van accuraatheid onder de 50%. Voor diegenen die niet weten hoe dat werkt: het is slechter dan een random toewijzing.

Appendix

A. Wat is Statistical Learning

Klassiekers in de literatuur die deze methode beschrijven zijn The Elements of Statistical learning uit 2001 ("ESL") en Introduction to Statistical Learning ("ISL" uit 2009). Beide zijn herdrukt en de meest recente versies zijn gratis te downloaden. Kortom standaard-werken. ESL is geschrevcen voor statistici op ongeveer master-niveau, ISL is iets makkelijker - maare vraagt nog steeds de nodige statistische kennis op academisch niveau.

Beide boeken hebben als hoofdauteurs Trevor Hastie en Robert Tibshirani, hoogleraren in Stanford. Er is ook een gratis cursus bij ISR van een jaar of 7 geleden te vinden bij Stanford University.

"Statistical learning refers to a vast set of tools for understanding data. These tools can be classified as supervised or unsupervised. Broadly speaking, supervised statistical learning involves building a statistical model for pre- dicting, or estimating, an output based on one or more inputs. Problems of this nature occur in fields as diverse as business, medicine, astrophysics, and public policy." (ISR)

B. algoritmes

De software gebruikt 5 algoritmes (glm, gbm, xgbTree, rf en rpart) weaarvan alleen glm geen decision trees gebruikt. gbm en xgbTree worden op vrijwel identieke manier gebruikt. xgbTree is onderdeel van het xgboost pakket, veel moderner dan gbm. Het presteert ook vrijwel altijd beter.

Modernere ontwikkelingen zoals neurale netwerken en lightgbm van Yandex ontbreken.

C. De Loss functies

Het vergelijken van een voorspelling met de bekende werkelijkheid gaat met een zg. loss function. Die hoor je te kiezen aan de hand van het probleem, niet omdat de ene wat makkelijker is dan de andere. Dehier vooral gebruikte is de "Acuuracy" en dat is ook de meest simpele: het percentage goed voorspelde gevallen.

(Technisch gesproken is dat natuurlijk een winstfunctie. Maar wiskundigen zijn nu eenmaal nerds die dat geen enkel probleem vinden.) Een tweede probleem, en dat is wel iets serieuszer, is dat de voorspellingen waarschijnlijkheden zijn en dat er dus drempelwaarden moeten worden afgesproken.

Nauwkeurigheid ("accuracy") vertelt hoe goed de voorspelling is. Bij 1% fraude-gevallen is een nauwkeurigheiud van voorspellen van 99% heel makkelijk, want dan voorspel je gewoon dat niemand fraudeert. Bij 60% farude zoals in deze train/test, is een random beslissing goed voor 56% nauwkeurigheid. De modellen komen met een nauwkeuigheid van 63-65% dus dat is zeker niet heel veel beter. En ik zie die 56% nergens terug als anker voor de beslissingen.

"Accuracy" is m.i. geen goede maat voor beslissingen. Misschien wel voor de software, maar dan is AUC - de maat van "Area under the Curve" flexibeler - en vaker gebruikt. AUC en de daarvan afgeleide Kappa (= (AUC-0.5)/2) komen wel in sommige rapporten voor maar de software selecteert er niet op.

BCE wordt niet gebruikt. Het is (min of meer) de standaard voor neurale netten en zou ook in deze situatie goede inzichten kunnen geven. BCE staat voor Binary Cross Entropy.

Jan van Rongen