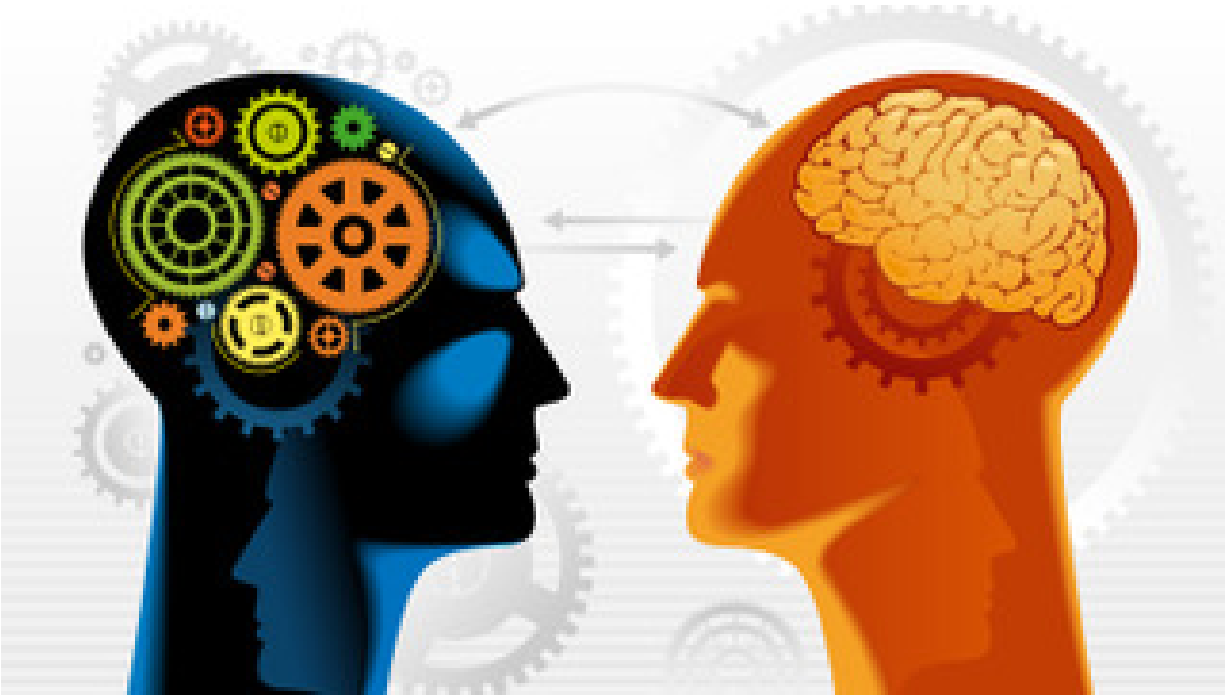
Opdracht 1: Classification



Tim Jongsma
Tom Schuitert

Opdrachtomschrijving

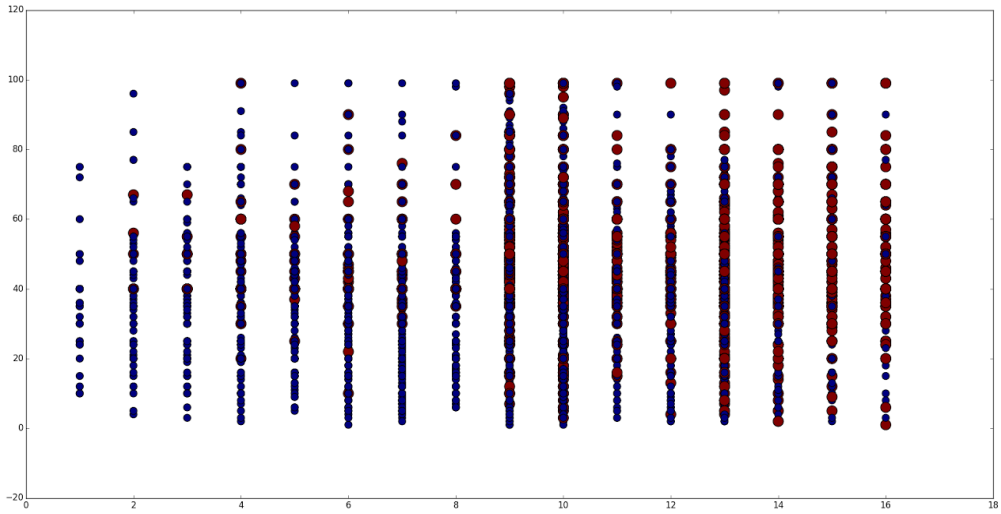
Voor deze opdracht hebben we een dataset gekregen met daarin verschillende features van een persoon. De opdracht is in eerste instantie om handmatig de verschillende features te analyseren en dan om in te schatten welke waarschijnlijk invloed hebben op het inkomen. Daarna moeten verschillende machine learning algoritmes uitgeprobeert worden. Deze algoritmes moeten aan de hand van een set gegevens inschatten of een persoon meer of minder dan vijftig duizend verdiend. We zullen de prestaties van de volgende algoritmes vergelijken: K-nearest neighbor, decision tree, naive bayes, support vector machine en random forest. Tenslotte geven we nog een kleine uitleg over random forest.

Features

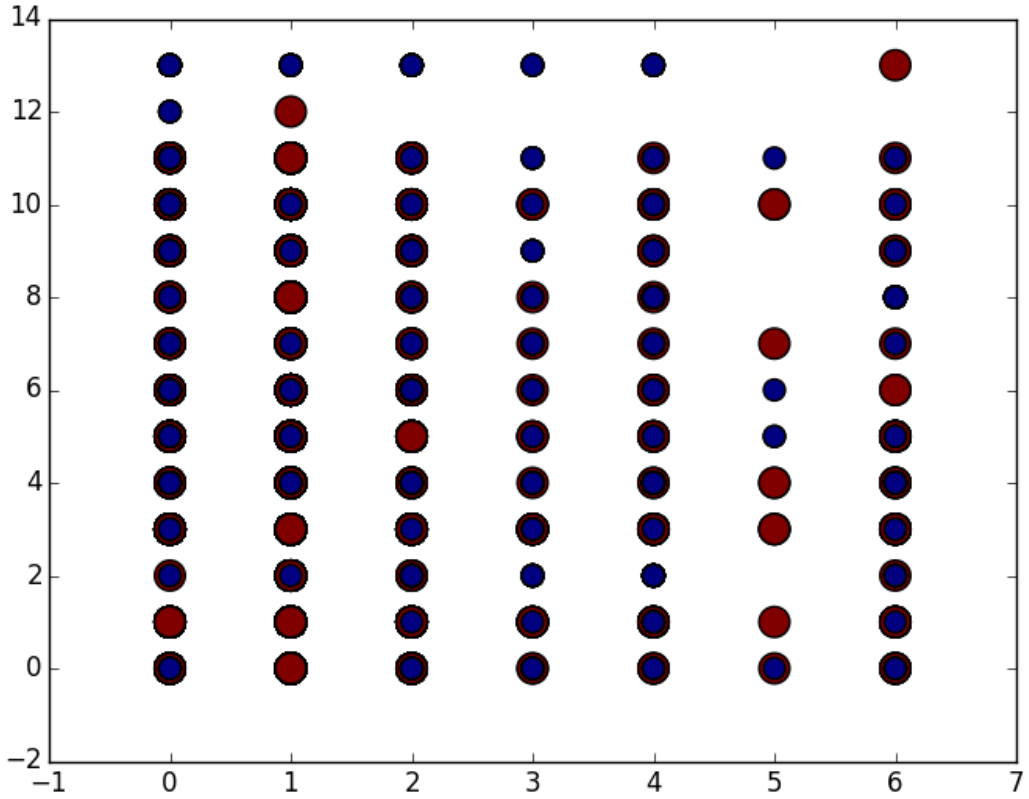
Naam	**Waarden**
Age	Continuous
Workclass	Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked
FNLWGT	Continuous
Education	Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool
Education-num	Continuous
Marital-status	Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouseabsent, Married-AF-spouse
Occupation	Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transportmoving, Priv-house-serv, Protective-serv, Armed-Forces
Relationship	Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried
Race	White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black
Sex	Male, Female
Capital-gain	Continuous
Capital-loss	Continuous
hours-per-week	Continuous
Native-country	United-States, Cambodia, England, Puerto-Rico, Canada, Germany, OutlyingUS(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands

Analyse features

Wij denken dat de features baan, uren per week, getrouwde status, ras en duur van de opleiding de features zijn waaruit kan worden afgeleid of een persoon meer dan 50.000 verdient of niet. Het profiel dat vaak terug kan worden gevonden bij een persoon met een salaris van boven de 50K is een getrouwde blanke man die meer dan 48 uur in de week werkt en een manager is.



Hierboven staat een grafiek met op de X as de studietijd en op de Y as het aantal uren werk in de week. Het is duidelijk te zien dat hoe hoger het aantal jaren studie en aantal uren werk in de week hoe vaker het voorkomt dat er meer als 50K wordt verdient.



X-as (martial-status):
0=Never-married, 1=Married-civ-spouse, 2=Divorced, 3=Married-spouse-absent, 4=Separated, 5=Married-AF-spouse, 6=Widowed

Y-as (Occupation): 0=Adm-clerical, 1=Exec-managerial, 2=Handlers-cleaners, 3=Prof-specialty, 4=Other-service, 5=Sales, 6=Transport-moving, 7=Farming-fishing, 8=Machine-op-inspct, 9=Tech-support, 10=Craft-repair, 11=Protective-serv, 12=Armed-Forces, 13=Priv-house-serv

In de bovenstaande grafiek is geen relatie te de getrouwde staat en de baan van een persoon. Wel is te zien dat personen met een de martial status Married-civ-spouse en Married-AF-spouse vaker meer als 50K verdienen als andere personen.

Implementatie

Er zijn 4 stappen die gebeuren :

1. data en testdata inlezen
2. data en testdata normalizeren
3. classifier aanmaken
4. classifier testen

Bij het inlezen van strings hebben we ervoor gekozen om per kolom de unieke waardes op te slaan in een array, de index in deze array is dan het getal dat we opslaan in de matrix.
De regels met kolommen waarvan de waarde niet bekent zijn hebben we niet verwerkt. Hier hebben we voor gekozen omdat er van uit zijn gegaan dat de waarde “onbekend” niks zegt over het salaris en daarom niet tot een positieve verbetering leidt.

Analyse algoritmes

Algoritme	**Parameters**	**Error rate**
KNN	Neighbours=3	0.18572377158034528
KNN	Neighbours=10	0.10126162018592297
KNN	Neighbours=50	0.10126162018592297
Support vector machine	Gamma=0.0	0.09375830013280212
Support vector machine	Gamma=0.5	0.12171314741035856
Support vector machine	Gamma=2.0	0.11115537848605578
Naive bayes	-	0.12337317397078353
Decision tree	Splitter=random	0.2681938911022576
Decision tree	Criterion=entropy	0.23399734395750332
Decision tree	Criterion=entropy min_samples_split=8	0.22901726427622843
Random Forest	-	0.12649402390438247
Random Forest	Estimators=30	0.11786188579017265

KNN
Het aantal buren waarnaar het algoritme kijkt blijkt daadwerkelijk invloed te hebben op de error rate van dit algoritme. Wat we echter ook zien is dat de invloed van het aantal buren waar het algoritme naar kijkt niet meer toeneemt vanaf een bepaald aantal.

SVM
De gamma waarde geeft de zachte scheiding tussen positieve en negatieve resultaten weer. Door deze zachte scheiding toe te voegen kan overfitting tegen worden gegaan. We kunnen zien dat een kleine gamma waarde bij deze dataset inderdaad een positieve invloed heeft.

Echter bij een hogere gamma waarde worden er teveel foutieve resultaten toegestaan wat een slechte invloed heeft op de error rate.

DT

Met verschillende parameters blijkt het niet mogelijk om met de gegeven dataset een error rate te creëren die in de buurt ligt van de andere algoritmes. Een mogelijke verklaring hiervoor is dat de verbanden tussen de verschillende features niet consequent genoeg zijn om herhalend tot hetzelfde resultaat te komen. Waardoor het lastig is één boom te maken die correcte voorspellingen doet.

Random forest

We hebben gezien dat een decission tree algoritme verreweg het slechtst presteert met de gegeven dataset. Dit komt doordat dit algoritme van nature gevoelig is voor overfitting waarbij er in de leerfase een boom wordt gecreëerd die te goed past bij de gegeven leer data. Wanneer er vervolgens met een andere set data getest gaat worden is de boom niet voor deze data geschikt wat resulteert in een hoge error-rate.
Een oplossing op dit probleem is het random forest algoritme. Zoals de naam al doet vermoeden wordt er hierbij een bos van decission trees gecreëerd. Dit is gebaseerd op het principe van bagging. Dit principe stelt dat de prestaties van een algoritme toenemen wanneer verschillende modellen worden gecombineerd.

Bij een random forest algoritme wordt dit op de volgende manier gedaan. Er worden een door de gebruiker gespecificeerd aantal decission trees aangemaakt. Echter een normale decission tree maakt in de leerfase gebruik van alle beschikbare data. Bij een random forest krijgt elke boom een random geselecteerd aantal regels uit de beschikbare leer data toegewezen. Hierdoor krijgen deze bomen allemaal een andere vorm. Binnen deze subset van de data kunnen namelijk andere features doorslaggevend zijn dan in een andere subset. Wanneer dit algoritme vervolgens gebruikt wordt geven alle bomen in het bos hun eigen antwoord op het vraagstuk. Vervolgens worden deze antwoorden gecombineerd. Bij een classificatie probleem worden de verschillende antwoorden geteld en vervolgens wordt het antwoord dat het vaakst voorkomt terug gegeven. Zo niet wordt het gemiddelde antwoord terug gegeven.

Door deze combinatie van verschillende bomen te gebruiken wordt het effect van overfitting wat gewone decission trees hebben afgevlakt. Hierdoor zal ook de error-rate lager zijn dan bij een decission tree algoritme dit valt ook te zien in de resultaten.