Postgraduate Certificate in Software Design with Artificial Intelligence

Advanced Machine Learning and Neural Networks - (AL_KSAIG_9_1)

Minor Exercise 1

Student ID: A00267948

Student Name: Daniel Foth

GIT: https://github.com/DanielsHappyWorks/aml-data-discovery

Brief Description: This assignment aims to outline data assumptions made before applying models to dataset (e.g. increase in rooms, increase in price), the strengths and weakness to each model, the accuracy of each model using cross validation and a conclusion which will outline the best model for the chosen dataset and why.

Contents

Data

The data set utilised for this exercise is the "Adult" data set. The data was extracted from the census bureau database by Barry Becker and contributed to the UCI Repository by Ronny Kohavi and Barry Becker.
UCI Repository: https://archive.ics.uci.edu/ml/datasets/adult

The data contains 32561 rows and 15 features. Which include:

1. Age - Numeric
2. Workclass - Categorical
3. Fnlwgt - Numeric
4. Education - Categorical
5. education-num- Numeric
6. marital-status - Categorical
7. occupation - Categorical
8. relationship - Categorical
9. race - Categorical
10. sex - Categorical
11. capital-gain- Numeric
12. capital-loss- Numeric
13. hours-per-week- Numeric
14. native-country - Categorical
15. salary – Categorical (>50K or <=50K)

For this specific problem, we are trying to establish whether a person makes over 50K a year. The "Adult" data set has a few missing values. This data set has both numerical and categorical data. Below you can find a table describing each feature individually.

age	***workclass***	***fnlwgt***	***race***	***sex***
Min.:17.00	Private:22696	Min.:12285	Amer-Indian-Eskimo:311	Female:10771
1stQu.:28.00	Self-emp-not-inc:2541	1stQu.:117827	Asian-Pac-Islander:1039	Male:21790
Median:37.00	Local-gov:2093	Median:178356	Black:3124	
Mean:38.58	?:1836	Mean:189778	Other:271	
3rdQu.:48.00	State-gov:1298	3rdQu.:237051	White:27816	
Max.:90.00	Self-emp-inc:1116	Max.:1484705		
	(Other):981			
education.str	***education.num***	***marital.status***	***occupation***	***relationship***
HS-grad:10501	Min.:1.00	Divorced:4443	Prof-specialty:4140	Husband:13193
Some-college:7291	1stQu.:9.00	Married-AF-spouse:23	Craft-repair:4099	Not-in-family:8305
Bachelors:5355	Median:10.00	Married-civ-spouse:14976	Exec-managerial:4066	Other-relative:981
Masters:1723	Mean:10.08	Married-spouse-absent:418	Adm-clerical:3770	Own-child:5068
Assoc-voc:1382	3rdQu.:12.00	Never-married:10683	Sales:3650	Unmarried:3446
11th:1175	Max.:16.00	Separated:1025	Other-service:3295	Wife:1568
(Other):5134		Widowed:993	(Other):9541	
capital.loss	***hours.per.week***	***native.country***	***capital.gain***	***salary***
Min.:0.0	Min.:1.00	United-States:29170	Min.:0	<=50K:24720
1stQu.:0.0	1stQu.:40.00	Mexico:643	1stQu.:0	>50K:7841
Median:0.0	Median:40.00	?:583	Median:0	
Mean:87.3	Mean:40.44	Philippines:198	Mean:1078	
3rdQu.:0.0	3rdQu.:45.00	Germany:137	3rdQu.:0	
Max.:4356.0	Max.:99.00	Canada:121	Max.:99999	
		(Other):1709		

Assumptions

With this much data available it's hard to assume what will best affect the accuracy of a model. From looking at the fields available I expect that age, occupation, education and hours per week will affect the precision the most as those are always a good indicator as to how much a person may earn. To check this theory, I'll compare models that use only those or all features. The data set with less features will be referred to as the minimal dataset in this document and the code.

The data set is very much skewed in favour of people earning less than 50K as the ratio of data entries is 3.15:1. This could affect how well the models can deal with categorising over 50K pay. The output is also categorical so regression-based techniques will probably score low.

Regression

Model Accuracy

Model	Label Encoding	One Hot Encoding
All features - linear	rmse: 0.3666839926524649 r2: 0.2610695398089894	rmse: 0.33917416703233644 r2: 0.36778441071048706
All features – polynomial degree 2	0.33126675005462264 Poly (deg 2) r2: 0.3969193970395877	No results – too intensive (33 features * 21815 rows for training data)
minimal dataset - linear	rmse: 0.4024513314435926 r2: 0.10988441861235965	rmse: 0.37482691088114695 r2: 0.2278864149291321
minimal dataset – polynomial degree 2	rmse: 0.39495188950294485 r2: 0.1427488860609536	No results – too intensive (108 features * 21815 rows for training data)

Strengths & Weaknesses of Models

1. All the models performed nearly equally poorly. This is as expected because we have a category as the output for this dataset.
2. Regression isn't the best option for trying to figure out if a value is True or False. Its better for estimating numerical values that have a range.
3. Linear regression is very fast so it can handle the data set very well.
4. Polynomial regression slows down a bit especially when we try to encode all the categorical columns using one hot encoding. It was so intensive that the results weren't being generated on the machine I own.

Clustering

Model Accuracy

Model	Label Encoding	One Hot Encoding
kMeans – all features	Accuracy 0.3804551457264826	Accuracy 0.3805779920764104
kMeans – minimal features	Accuracy 0.3598476705260895	Accuracy 0.3597862473511256
kNN Regressor – all features	Regressor score 0.0876405148913636	Regressor Model score 0.08768039562392449
kNN Regressor – minimal features	Regressor Model score 0.17547645146352142	Regressor Model score 0.1775284159620465
kNN Classifier– all features	Classifier score 0.792573981016192	Classifier Model score 0.792573981016192
kNN Classifier – minimal features	Classifier Model score 0.7807556300018612	Classifier Model score 0.7795458775358273

Strengths & Weaknesses of Models

1. kMeans performed poorly with all the datasets.
2. The kNN Regressor performed even worse than kMeans in general as it is better at estimating numbers than a True/False value. With this kind of output, we would be better off guessing randomly.
3. The kNN Classifier performed very well getting nearly 80% with all the datasets. These are the first acceptable models found as it is capable of correctly classifying the output we are trying to predict.
4. All of these models are fast at processing large datasets with many features.

SVN

Model Accuracy

Model	All Features	Minimal Features
SVC – with min max scaler	CV score 0.7531445084682907 Accuracy 0.7614923384410394	CV score 0.7073965035283087 Accuracy 0.7021985343104596

SVC – without scaler	CV score 0.7600269739811287 Accuracy 0.7548301132578281	CV score 0.6551185840584122 Accuracy 0.6455696202531646
SVM	Accuracy 0.8238997573784589	Accuracy 0.7557814563434784
SVR	CV score 0.2628153255250669 Accuracy 0.24626707604271025	CV score -0.04764654751289381 Accuracy -0.04493932440605941

Strengths & Weaknesses of Models

1. More complex models get slower. For all of these I had to drop one hot encoding and for the SVC and SVM I had to limit the data set to 5000 entries.
2. The overall performance of the SVC & SVM seems very good (above 75% with all features) but it is using a limited number of rows which could be skewing the output.
3. SVR did very poorly. This is probably because it's a regression technique that doesn't perform classification of data very well. SVR was significantly faster than SVM and SVC overall.

Ensemble

Model Accuracy

Model	All Features	Minimal Features
Bagging Ensemble Regressor	Accuracy 0.28926298881160606	Accuracy 0.13732625950319055
Bagging Ensemble Clasifier	Accuracy -0.03392700160826245	Accuracy -0.24379619414680787
Random Forest Ensemble Regressor	Score 0.3766799320710287	Score 0.0742274599258933
Random Forest Ensemble Clasifier	Score 0.8470750806080147	Score 0.7690772301550745
Voting Ensemble	MLPRegressor -104.9674557274744 KNeighborsRegressor 0.09638129995444955 LinearRegression 0.24986482970320611 VotingRegressor -20.866295785713298	MLPRegressor 0.1359988897527048 KNeighborsRegressor 0.17458411947910601 LinearRegression 0.102173034957176 VotingRegressor 0.15683971051267986
Voting Ensemble	MLPClassifier -3.2813315277863264 KNeighborsClassifier -0.1855467856054973 GaussianNB -0.09560284594612067 VotingClassifier -0.09474623699698381	MLPClassifier -0.47251078356636556 KNeighborsClassifier -0.2712076805191892 GaussianNB -0.31146830112862456 VotingClassifier -0.28748325055279067

Note: I think I did something wrong with the bagging and voting ensembles as the outputs are weird. Feel free to send me an email if you see where its incorrect within the code.

Strengths & Weaknesses of Models

1. The bagging and voting ensembles did dab but I think its an issue with the way I configured them.
2. The Random Forest Regressor came out as expected based on all the other regressor models. Once gain it trying to estimate numbers which lowers the Score.
3. The Random Forest Classifier did very well as its capable of classifying True/False values.

Neural Network

Model Accuracy

Model	All Features	Minimal Features
NN regressor	Score 0.4295298170294163	Score 0.24381352330543427
NN classifier	Score 0.8552126516198373	Score 0.8010133578995855

Strengths & Weaknesses of Models

1. Neural Networks can be slower that other models for processing large quantities of data.
2. They also take a lot more tinkering to potentially get a good result.
3. Once again, the regressor performed poorly and classifier performed well.
4. The classifier model with all the features had the best score overall at 85% but its hard to tell how that result is achieved.

Conclusion

The data set used was very large. It consisted mostly of data that had categories, so it had to be encoded for most algorithms used. I used label and one hot encoding but quickly realised that for a dataset of this size one hot encoding was too much. When I tried using one hot encoding with SVC's I abandoned it as it was too slow yet produced similar results in Clustering and Regression.

As expected, the best performing models were ones that could categorise the data. Regression models tended to do poorly. Most classifier models achieved a score of over 70%. Which would be a pretty good guess. Regression models usually had an accuracy of sub 40%.

Lastly the features I expected to give good results usually did ok in comparison to using the whole dataset when the model was good. They tended to be up to 10% less then when using the full dataset which could be considered a significant performance decrease. The only selling point to using one of these is the speed boost from shoving less data into certain algorithms but the boost probably wouldn't be worth it in the long run.