COMP 309: Assignment 4

# Keiryn Hart

## 300428418

Part 1: Performance Metrics in Regression:

Data Pre-Processing:

After loading the diamonds dataset into python, I firstly checked to see if the dataset had any missing values, it did not so I did not have anything to worry about in that respect. I then used df. describe() to gain an understanding of the basic characteristics of the dataset and to check to see if the data seemed to be relatively normal. Since everything looked pretty good, I decided to continue into my exploratory analysis.

Exploratory analysis:

After running some exploratory analysis on the data, I noticed a few interesting things. Firstly, when looking at the categorical variables in comparison to price I managed to identify features that seemed to affect the price of diamonds more than others.

Chart, bar chart

Description automatically generatedChart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

The bar plots above show the effects that the cut, clarity and colour of the diamonds have on the price. We can see that ‘fair’ and ‘premium’ cuts appear to be the most expensive and the ‘J’ category for colour appears to be the most expensive which is quite strange as J is meant to be the worst colour. In terms of clarity we can see that S12 also appears to be the most expensive which again is slightly strange as it is also one of the worst categories.

Before doing anything else or splitting the data into training and testing sets, I decided to change the category variables from string objects to integers which will make training easier and allows for normalization if necessary.

After this I decided to test the correlations between all the variables to see for any strong correlations, this is now easier to see now that I have changed to categorical variables to numeric.

A picture containing chart

Description automatically generated

We can see that there seems to be a few relatively high correlations between x, y and z and also with carat and those 3 variables. Other than that, there are not many other strong correlations with a lot of the correlations ranging between 0.2 and -0.2. the strong correlations between x, y, z and also carat could be a result of x, y, z being the variables determine the size of the diamonds and carat also being a measurement of size in regard to weight. For this reason, I do not think that these stronger correlations will have negative effects on the performance of my models as the correlations are fair and accounted for.

Data preparation and model training:

After splitting the data into training and testing sets with 30% of the data being used for testing it was now time to train my models and analyze the results. I trained 10 different algorithms which were; linear regression, k-neighbors regression, Ridge regression, decision tree regression, random forest regression, gradient Boosting regression, SGD regression, support vector regression, linear SVR, and multi-layer perceptron regression.

The table below shows the various results and performance metrics that I got out of my models:

Table

Description automatically generated

As shown in the table above we can see the MSE, RMSE, r2 and MAE for all 10 of the algorithms that I have trained. When comparing the results, the first thing I do is look at the r2 values from this we can see that the 2 best models appeared to be the gradient boosting regression and k-neighbours regression with an r2 value of 0.997 which is extremely accurate we can also see with these regressors that their values for MSE, RMSE and MAE are substantially smaller than the other algorithms these lower value indicate that the these 2 models have a much better fit than the other models I tested.

The worst performing model by a landslide was the support vector regression which reported an r2 value of 0.499 and the other corresponding values followed a similar trend being much larger than the other models. I believe this could be due to the fact that the data was not standardized and therefore the data had an effect on how well the model could interprate it and make predictions. When I was using the linear SVR I decided to implement a standard scalar with the hopes that this may help and it did although it still was not one of the better models and was more or less towards the worse end out of the 10 models.

Part 2: Performance Metrics in Classification:

Data Pre-Processing:

Just like the diamonds dataset the first thing I did after loading 2 csv files in was checking for NA values, again both datasets had no missing values. After this I then decided to drop the variable fnlwgt as I thought that it was not hugely useful nor did it make a lot of sense to me.

After looking at the data a little bit I noticed that the occupation column and the work class column both had instanced where their values were ‘?’ I decided to change these to other just to make the data a bit more interpretable. After this I noticed a number of different things related to the data and consistency between the training and testing sets. The income column in the training and testing sets had slightly different values where in the testing set each value had a full stop at the end of it whereas the training did not, I thought this may effect the models so I made them the same in both datasets.

The capital gains and capital losses columns had a lot of instances of 0 which is probably accurate as a lot of people may not have other means of income and therefore may be getting no capital gains and/or capital losses.

On the topic of changing values, I decided to change the values for the income column to binary format where 1 indicated someone who earned over 50k and 0 indicating someone who earned under 50k. other values in columns that I elected to change was in the training set I changed ‘Holland-Netherlands’ to ‘other’ as it was not a value in the testing set and when it came to making dummy variables this was causing a lot of issues for me.

Exploratory analysis:

Firstly I began by looking at a correlation heat map, which is shown below, we can see that there are no strong correlations at all between any of the variables in the data, this is interesting but makes sense as there is not a lot of numerical values most of the information is categorical.

A picture containing qr code

Description automatically generated

A picture containing text

Description automatically generatedA picture containing text

Description automatically generated

After looking at the correlations I was interesting to know how balanced the data was, using .value\_counts() I looked at both the training and testing sets and saw that both were quite imbalanced where the amount of instances of ‘0’ was significantly more than that of ‘1’ shown above on the left is the training set and on the right is the testing set. Although there is quite a large imbalance, I think that with cross validation this imbalance shouldn’t have too much of a negative effect on the modelling.

Based on my exploratory analysis of the data and my look into the differences between variables I think that for a classification task it is important that I create dummy variables for each categorical variably which should hopefully assist the classification of the algorithms later on, to do this one new binary column is created for each individual category in each individual variable. After doing this and looking at the shape of the data, the number of columns increase from 14 variables to 107 which seems like a lot, but they all contain basic binary information which helps the classification perform better. This also means that I can normalize the data as before with string categories it would not have been possible to do so because they need to be numeric.

I thought that 107 variables is far too many for prediction although they are dummy variables I still felt that there wasn’t a need for that many, it could lead to the model being to specific and overfitting the training data which would mean the model would be useless with new data. Because of this I performed PCA on the training set and reflected it in the test set I gave the PCA a limit of 20 variables as after a bit of testing it seemed to be a good point that didn’t sacrifice accuracy.

Data preparation and model training:

After all of that analysis I decided It was time to prepare the training and testing sets and start to model the data. Once I had organized the training and testing sets I used standard scaler to standardize all the data, I thought this would be useful as there are some numerical variables that have varying ranges which could have an impact on the modelling, plus for an algorithm like kNN it will be necessary so that the distance measurement is not affected in any way by the different numerical categories.

Once this was done, I began training my models, these models were kNN, naive

Bayes, SVM, decision tree, random forest, AdaBoost, gradient Boosting, linear discriminant analysis, multi-layer perceptron, and logistic regression.

The table below shows the different performance results from each of the 10 classification models:

Table

Description automatically generated

When analyzing the results of the table above we can see that when looking at the f1 score which is a weighted average of both precision and recall that this metric state the two best classifiers were Multi-layer Perceptron and SVM we can also see that in all of the other performance metrics for these two classifiers that they appear to be overall the two best out of all 10 classifiers.

The worst performing classifiers appear to be a close battle between the Naïve Bayes and the decision tree, both these classifiers were still very accurate and by no means bad but the performed much poorly overall in comparison to the other classifiers.

The confusion matrix for the two best performing classifiers are shown below with MLP being on the left and SVM on the right.

A picture containing text

Description automatically generatedText

Description automatically generated

When thinking about accuracy as a performance metric to evaluate classification results it is hard to say that it is the best metric to assess the classifier. Obviously, accuracy is still a good way to analyze the classifier it should not be considered to be the only option or the best option for evaluation.

Accuracy in some cases may not be the best metric to evaluate the performance of a classifier, this is especially apparent in instances where there might be a large class imbalance in the dataset which is what has occurred with this adult dataset. There is a large imbalance and as a result this can lead to the classifier having a high accuracy score as it is correctly classifying most of the instances in the larger class but the smaller class may suffer as a result as the classifier has a better predictive power for the larger class meaning it might get a lot more of the smaller class wrong.

For this reason, Accuracy is probably not the best performance metric to evaluate a classifier, especially in this situation. It can still be used to help inform a decision, but it is better to take additional measures in order to be more confident about the performance of the classifier.

Part 3: Optimisation Methods:

a)

BGD+MSE:

Chart

Description automatically generated

Above is the path of Gradient Descent for BGD+MSE, this shows the change in iterations that the model made in order to minimize the cost function. When considering the difference between batch gradient descent as shown above and mini-batch gradient descent is that batch gradient descent approaches minimizing the cost function by using the whole training set in every iteration, in comparison to this mini-batch gradient descent uses a small portion of the training set in each iteration this could cause it to take a longer or shorter period of time depending on the batch sizes and number of iterations.

(I could not get mini batch to work)

b)

Table

Description automatically generated

Shown above are the results of the four learnt models based on the performance metrics on the test set. There was not a huge amount of difference between the performance metrics of the four learnt models really they were all very similar and the differences between the values was in the 0.01 range.

c)

Chart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generated

Displayed above are the scatter plot and regression lines of the two learnt models PSO+MSE (left) and PSO+MAE (right)

d)

computational times:

the computational time for the standard batch gradient descent was 0.108 seconds, in comparison to this, the time for PSO+MSE was 4.769 seconds and the time for PSO+MAE was 4.269 seconds.