Task 2C

Task 2a was aiming to find which classification algorithms perform better by splitting the data set into training set and test set, then train the model with training set and test the accuracy using the test set. Take only the data who has class label. Then split the data into 2/3 training set and 1/3 test set. For the missing values in both sets, impute and fill them with their corresponding data median (fill training set’s missing value with training set’s median, fill test set’s missing value with test set’s median). Normalized the data sets by removing the mean and scaling to unit variance. Train each model with the normalised training set then perform classification. Looking at the results of each algorithms, accuracy of decision tree is 78.689%, accuracy of k-nn when k=5 is 81.967% and accuracy of k-nn when k=10 is 86.885%. The k-nn when k=10 seems to have a better performance in the classification of this data set, and decision tree has a relatively low performance among these algorithms. It is clear that when the k value is bigger, the k-nn performs better on the data set, which is interpretable as more samples we look at that nearby the data we predict, we might get a better understanding of which cluster the particular data belongs to.

A close up of a map

Description automatically generatedTask 2b compares some feature selection methods. Beforehand, concat world and life file, remove those rows without class label. As well as fill every missing value with its corresponding column median. In order to find the cluster label, the number of clusters need to be determined at first. A graph is drawn to find the number of clusters on the data set as its silhouette score. The graph portrays that when the number of clusters between 2-4, it has the highest rating. However, since the class label has 3 groups, to match the class label of the data set, 3 clusters was picked. Afterwards, fit the data into Kmeans with cluster set to 3 to find the cluster labels, add the cluster label column to the ‘feature’ data frame. Aggregate the original 20 features to create the interaction between these features which form 190 new features. A data frame called feature\_data was made with the original 20 features plus cluster label as well as the 190 interaction pair features. Based on these 211 features, the train test split is performed where the class label is the column of life expectancy in csv life. Next the normalized mutual information between each of the 211 features and the class label was computed and placed in to feature\_NMI. In doing so, we can find out the amount of information about the class label we gain by knowing every feature. Then feature\_NMI was sorted in descending order with the NMI value. Took the top 4 features and their corresponding values for each country, so these 4 features were the most correlated features to life expectancy. Reset the value of X\_train and X\_test with only these 4 features. Normalise both of them by StandardScaler and lastly use k-nn to calculate its accuracy of the prediction of test set. Hence the accuracy of feature engineering is 77.049%.

After the implementation of feature engineering using interaction term pairs and clustering labels. We then use PCA to do feature engineering. To start with, the data set was split into 2/3 training set and 1/3 test set. Normalise the X\_train and X\_test as before. From sklearn library import PCA, set the number of components to be 4, apply k-nn to get the accuracy of the classification which is 75.410%. Lastly, take the first 4 features and split the data into training and test sets, normalised it and finally perform k-nn classification to obtain the classification accuracy, which appears to be 75.410%.

Through these steps, we got the accuracy score of classification using the same classification method (k-nn when k=5) but different methods of feature engineering. We find out that using interaction term pairs and clustering labels, we got the highest classification accuracy. This is reasonable, as we not only using the original 20 features but also many combination of features that gives us chance to explore some more strong features to do the classification as well as provides us with better understanding of the correlation between the features and class by using mutual information. Hence this method should get a higher precision than just simply pick the first 4 features at random, also it is logical to be more accurate than using the original 20 features using PCA.

There are serval ways to improve classification accuracy with this data. For example, increase the size of training set. With a larger portion of training set size, the model will be better so the prediction of the test set would be more accurate. Also, trying other classification algorithms is the ideal approach to achieve higher accuracy. Through incessant experiments, you might find that some algorithms are better suited to a particular type of data sets than others. Hence, we should apply all relevant models and check the performance. In the end, we can pick the algorithms that best fit the data set.

Overall, the method of feature engineering is useful in data classification as it filters out the less importance features and reduces its dimension which reduces the complexity of a model and makes it easier to interpret, either as graphs or some other representations. It also enables the classification algorithm to train faster as there are less features to look at. However, as we can observe from the result of task-2B, the accuracy of classifications after feature engineering is lower than using the full 20 features (82%). That is because we got less information from the data set as many features has been removed, although we already choose the most significant features to do the prediction. In this data set, because the total feature is 20 which is not really big, it is possible for us to keep them all and use them to built models. However, in some cases there might be hundred and thousands of features, there are also many excessive data, and it is impossible for us to use all of them. So, it become very important that we are able to use feature engineering and feature selections to exact the information we need. Therefore, sometimes feature engineering might have some disadvantages but it is generally a very useful technique if we do not care so much about the loss of some precision. Furthermore, to some extend the k-nn(k=5) is reliable as it simply finds the dominate group nearby the data point, which in most of the time it is correct. However, it also got some shortcomings. K-nn is sensitive to noisy data and outliers, that means imbalance data might cause a problem. It does not learn anything from the training data and simply uses it for classification. In addition, another issue with k-nn is that it is hard to find an optimal number of neighbours, which it is kind of important in k-nn classification as different k value can lead to a big difference in some situation. So, in this data set, k=5 might not be the best choice.