Ryan Hsu

CS677 – Final Project

4/27/2022

**Feature Selection and Classification Performance**

This project attempts to dive into the effect feature selection makes on different prediction models. In particular, we will dive into what is more important in feature selection, the amount of features or the best correlated features. As well, we will use two different selector tools SelectKBest and RFE selection from the sklearn library to help select the ‘best’ features.

To visualize this, this project attempts to find the most important features that lead to a student doing well in an academic course. The dataset is from the UCI Machine Learning Repository and can be found in the link below:

UCI Machine Learning https://archive.ics.uci.edu/ml/datasets/Higher+Education+Students+Performance+Evaluation+Dataset

**Data Processing**

The dataset from UCI has 32 features and one of the features was the Course Id for the class the student was taking. As we are just finding out performance of a student, this column was removed from the dataset and the course they were taking was not taken into account. Though, interestingly, this feature had the highest impact on predicting data. This could possibly be pointing more towards how easy a course was. But, since we are more interested in demographics outside of the difficulty of a course it has been removed for this project. Also, the true labels were a gradient from fail to ‘A’, because we just want to predict performance, this was compressed to be just pass or fail. That is a grade of ‘C’ or higher is pass, otherwise it is considered a failing grade.

**Python Code**

***Instructions to run code:***

***To run this project, simply ensure you have tools.py, featureselection.py, modelDataFit.py and DATA.csv in the same directly. Then run modelDataFit.py.***

This project has three Python files, featureselection.py, modelDataFit.py and tools.py

featureselection.py handles the feature selection using SelectKBest and RFE selection tools from sklearn. The two methods kbest\_select() and rfe\_select() both take a dataframe and the amount of features you want to get back. Interestingly, SelectKBest does a calculation over all features and to get the number of features you want back you just slice that list. So instead of returning a sliced amount each time, kbest\_select() will just return the whole list and it can be sliced before use. This saves the code from having to run the selector tool over and over again.

modelDataFit.py handles the actual prediction of the data. In the code, there is a ‘for-loop’ that will step through feature counts (5 to 31). This can obviously be changed to match whatever number of features you want to see. At each feature, SelectKBest and RFE will return their best ‘x’ number of features. A subset of the dataframe with these features will be created and that dataframe will be fitted using four different models, MultinomialNB, DecisionTree, KNeighborClassification and Polynomical SVM.

The code will first plot two plots, one for SelectKBest and the other for RFE selection. Each describing the accuracy over the number of features for the four different models. Then the code will plot another four plots, this time describing the difference of the selector tools for each model as a function of accuracy versus feature selection.

Tools.py is used to put the data into a dataframe and to grab a subset of the dataframe when needed.

**Findings**

The best feature selector, feature amount and classifier for this dataset was the RFE Selection tool from sklearn with 5 features, classifying with KNeighborClassifier with 5 neighbors.

The best 5 features were:

1. Age
2. Sex
3. Salary
4. Impact of Project
5. GPA from last semester

The most important feature was the GPA, then the Impact of Project and thirdly the Salary. These had the most impact on the prediction of the student’s performance.

