REPORT_PART 2

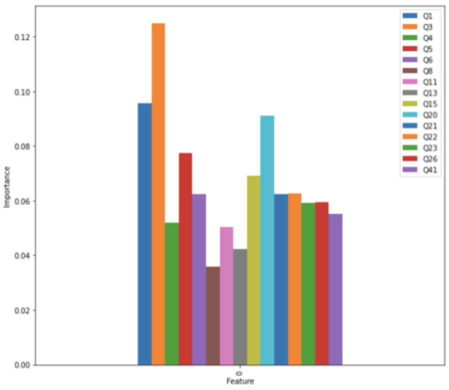
To study for the salary bucket of a survey respondent’s current yearly compensation, the ‘clean_kaggle_data_2021.csv’ generated from ‘2021 Kaggle Machine Learning & Data Science Survey’ dataset has been used, and the analysis has been broken down into five parts as per the report below, while codes for analysis is in the ipynb file.

Data cleaning:

The data provided in the contains missing values and categorical feature, so in this part, these have to be justified and encoded. Since the dataset contains 15391 participant results, columns with more than 10000 missing values are dropped first. Although some of these columns are multiple-choices columns which contains either selected or null value, there are about 2/3 participants did not select this choice in the multiple-choice part which might indicate that these choices are relatively not that important and are dropped as well. After dropping columns that have more than 10000 missing values, there are only 42 columns remaining. Among the remaining columns, there are some features that are categorical but not ordinal, and theses features are encoded using dummy variables. These features include all other columns except for Time from Start to Finish(seconds), Q1, Q2, Q3, Q4, Q5, Q6, Q8, Q11, Q13, Q15, Q20, Q21, Q22, Q23, Q25, Q26, Q41, Q25_Encoded and Q25_buckets. After encoded categorical but not ordinal in dummies, we look at missing values in the remaining columns. Q8, Q11, Q13 and Q15 all have 961 remaining columns, so participants with missing value in Q8 are dropped to see whether the same group of participants did not answer these four questions. And since after dropping participants with missing values in Q8, there are 0 remaining missing value in Q11, Q13 and Q15, therefore the same group of participants did not answer these four questions and they are dropped from the data frame. Since we have 42 columns now, participants with more than 10 questions unanswered among these 42 questions have been checked and dropped. Then there are only 204 missing values in Q26, and 1627 missing value in Q41. For Q26 about 38% participants chooses $0 option and no other option has the same large amount as this. Since there are only 204 missing values, which is not large, the mode is used to fill in these missing numbers. For Q41 nearly 40% of participants chose the option of 'Local development environments (RStudio, JupyterLab, etc.)'. Since the number of missing values is 1627, which is not a very large portion of the participants and thus the mode is used to reflect this group, therefore, missing values are filled in with mode. Since ‘Time from Start to Finish (seconds)’ is not relevant to compensation in each salary bucket, it is removed from the data frame. And also, since we need to encode the columns and ‘Q25_Encoded’ has the encoded ‘Q25_bucket’, we remove ‘Q25’ and ‘Q25_bucket’ and use the encoded column ‘Q25_Encoded’ as the encoded target variable. Then encoding for all remaining columns except for ‘Q25_Encoded’ and dummies need to be done. Q1, Q4, Q5, Q6, Q13, Q15, Q21, Q22, Q23, Q26 and Q41 are clearly ordinal, so ordinal encoding is applied to them. For Q2, Q3, Q8, Q11and Q20 although they are not clearly ordinal, their different options do have certain order when related to compensation. For example, for Q3, the country reside is relevant to salary bucket because developed countries may have higher salary comparing to developing countries, therefore, these columns are also encoded using ordinal encoding.

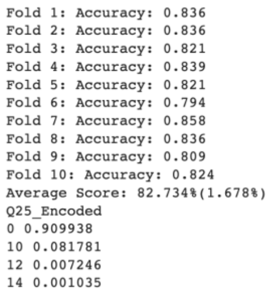
Exploratory data analysis and feature selection:

Correlation plot is used as below to help visualize the order of importance for exploratory data analysis. And since the correlation plot (Appendix 1) contains many features and from the plot, it is shown that some of the features have relatively low importance, therefore feature selection is needed to help select important features. The data is split to training set and test set with test set containing 33% of the data. Since it contains features which have been encoded using ordinal encoding and some features have relatively fewer ordering levels while others may contain many levels, stander scaler is applied to the training set and the test set to standardized features by removing the mean and scaling to unit variance to make the analysis more reasonable. Random forest classifier is used to perform feature selection. Random forest has sub-samples of the dataset, and many decision tree classifiers are fitted on them to help improve the accuracy of prediction and to control the problem of overfitting. This classification algorithm is used to help with feature importance and feature selection. The feature importance plot (Appendix 2) of all features shows that the features on the left has relatively much higher feature importance than the features on the right and many features on the right share similar importance. Random forest classifier is applied to help visualize the order of feature importance and top 15 important features among all remaining features have been manually selected based on the order provided by random forest and the plot of the order of these 15 features is shown above. The top 15 features are: Q1, Q3, Q4, Q5, Q6, Q8, Q11, Q13, Q15, Q20, Q21, Q22, Q23, Q26 and Q41. Feature engineering is a very useful tool in this assignment because encoding is used to help with the data cleaning and preparation process of many categorical variables and feature selection is useful for large number of features.

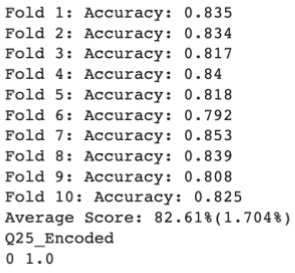


Model Implementation:

In order to implement ordinal logistic regression algorithm on the training data using 10-fold cross-validation, the OrderedModel from statsmodels is used with distribution set to logistic. Two values of the hyperparameter ‘method’ are used to compare the performance of the model. As we can see from the results below, the accuracy of each fold along with the average accuracy score, the standard deviation of the accuracy and the probability of belonging to each salary bucket have been presented when ‘method’ is chosen as ‘bfgs’ (left) and ‘nm’ (right), while other hyperparameters are kept the same. To compare the performance of model, based on bias-variance trade-off, relatively lower bias is preferred so that it won’t be very underfitting and relatively lower variance is preferred so that it won’t be very overfitting. Since model accuracy is involved in 10-fold cross-validation, relatively higher accuracy would be preferred because it relates to lower bias, and relatively lower standard deviation is preferred because it relates to lower variance. Therefore, based on the cross-validation accuracy results, when selecting the hyperparameter, ‘method’ as ‘bfgs’ would perform better. As stated in the section above, stander scaler is applied to the data to standardized features by removing the mean and scaling to unit variance so that it would help with the issue that after encoding, some features have relatively fewer ordering levels while others may contain many levels and the number for them vary, by using stander scaler it would help with this issue.



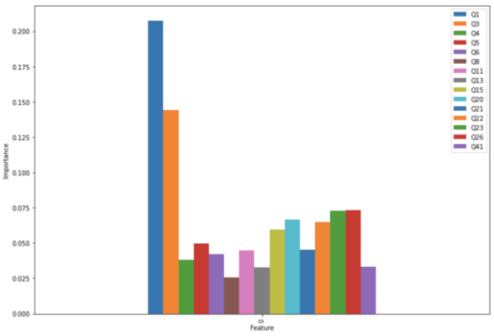
```
Fold 1: Accuracy: 0.836
Fold 2: Accuracy: 0.836
Fold 3: Accuracy: 0.821
Fold 4: Accuracy: 0.839
Fold 5: Accuracy: 0.821
Fold 6: Accuracy: 0.794
Fold 7: Accuracy: 0.858
Fold 8: Accuracy: 0.836
Fold 9: Accuracy: 0.809
Fold 10: Accuracy: 0.824
Average Score: 82.734%(1.678%)
Q25_Encoded
0 0.909938
10 0.081781
12 0.007246
14 0.001035
```



```
Fold 1: Accuracy: 0.835
Fold 2: Accuracy: 0.834
Fold 3: Accuracy: 0.817
Fold 4: Accuracy: 0.84
Fold 5: Accuracy: 0.818
Fold 6: Accuracy: 0.792
Fold 7: Accuracy: 0.853
Fold 8: Accuracy: 0.839
Fold 9: Accuracy: 0.808
Fold 10: Accuracy: 0.825
Average Score: 82.61%(1.704%)
Q25_Encoded
0 1.0
```

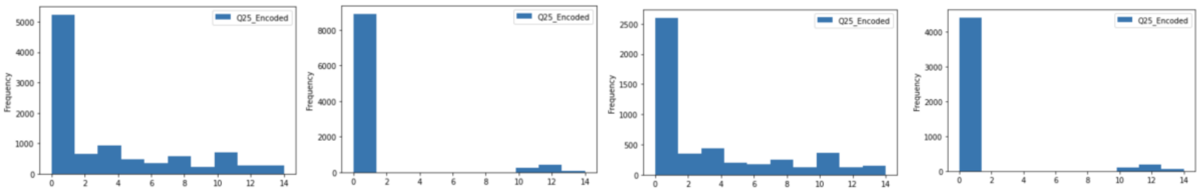
Model tuning:

For the model I used which is the OrderedModel from statsmodels with distribution set to logistic, and its hyperparameters includes ‘start_params’, ‘method’, ‘maxiter’, ‘full_output’, ‘disp’, ‘fargs’, ‘callback’, ‘retall’, ‘skip_hessian’ and ‘kwargs’. Two hyperparameters selected for model tuning are ‘method’ and ‘maxiter’. Grid search is applied by performing cross-validation on the combination of different ‘method’ and number of ‘maxiter’. For ‘maxiter’ it includes ‘newton’, ‘nm’, ‘bfgs’, ‘lbfgs’, ‘powell’, ‘cg’, ‘ncg’, ‘minimize’. And for ‘maxiter’ it includes 1, 5, 10, 20, 50, 100. Confusion matrix to test for accuracy is used to compare the performance of the models and the optimal model after grid search is the model with ‘maxiter’ set as 1 and ‘method chosen as ‘lbfgs’, and the accuracy is 82.764%, which indicates relatively high accuracy and quite good performance of the model. Feature importance plot is created to see which features were the most determining in model predictions. To compare with the feature importance graph obtained in Section 2, in section 2 plot, top five important features ranking from the most important to the least is Q3, Q1, Q20, Q5 and Q15, while in section 4 they are Q1, Q3, Q23, Q26 and Q20. Q1, Q3 and Q20 are remained in the top 5 important features, while there are some changes in the importance of other features in model prediction. In addition, in section 2, the most important feature Q3 has importance a little bit over 0.12, while in section 4, the most important feature Q1 has importance a little bit over 0.2, and the overall trend of the order of importance is still rather similar to that in section 2.



Testing & Discussion:

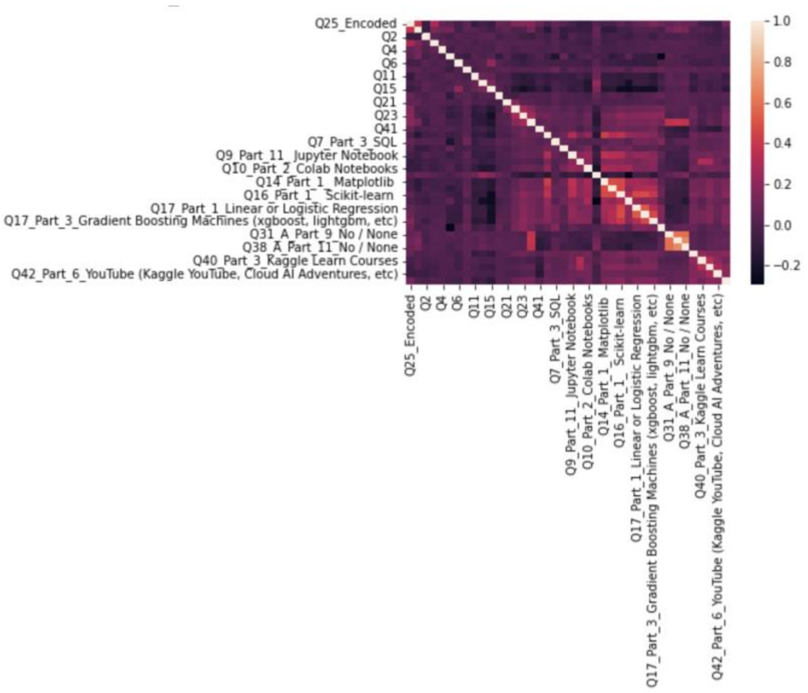
Optimal model is applied on the test set and the model has an accuracy of 79.84% on the test set. Compared with the 82.764% accuracy on the training set, the accuracy on the test set is a little bit lower but reasonable. The optimal model performs a quite proper fitting with relatively similar accuracy on the training and test set, which is not overfitting or underfitting. If more data is provided, the accuracy of the model might be able to increase due to a better model with more valid data. If much deeper model tuning has been performed based on more hyperparameters, it might increase the accuracy of the model. And to perform a different algorithm other than the algorithm used in this study would help to increase the accuracy. Below are the plots of the distribution of true target variable values and their prediction on both training and test set: (true on training, prediction on traing, true on test, prediction on test)



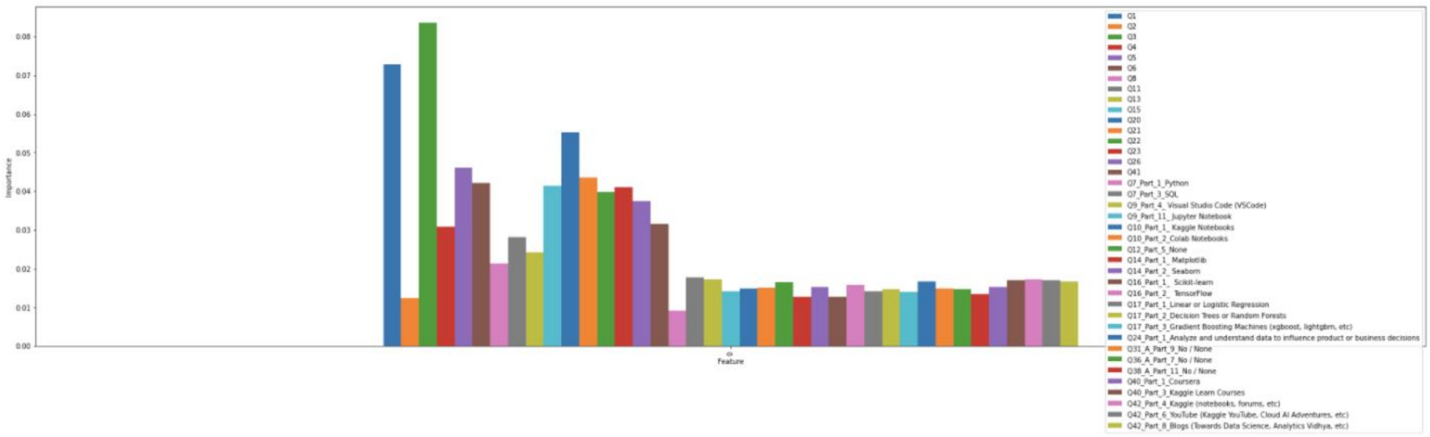
From the above plots, it shows that my trained model would prefer to highlight the difference between salary buckets with very high frequency and very low frequency. For example, from 0-1 encoded level which is $0-9999 and $10000-19999, these level ranges have the highest number of data points on true target variable, while in prediction, the number becomes even higher. However, some class with small number of data points in true target become less in prediction.

Appendix

Appendix 1:



Appendix 2:


Importance
Feature
Q1
Q2
Q3
Q4
Q5
Q6
Q8
Q11
Q13
Q15
Q20
Q21
Q22
Q23
Q26
Q41
Q7_Part_1_Python
Q7_Part_3_SQL
Q9_Part_4_ Visual Studio Code (VSCode)
Q9_Part_11_ Jupyter Notebook
Q10_Part_1_ Kaggle Notebooks
Q10_Part_2_Colab Notebooks
Q12_Part_5_None
Q14_Part_1_ Matplotlib
Q14_Part_2_ Seaborn
Q16_Part_1_ Scikit-learn
Q16_Part_2_ TensorFlow
Q17_Part_1_Linear or Logistic Regression
Q17_Part_2_Decision Trees or Random Forests
Q17_Part_3_Gradient Boosting Machines (xgboost, lightgbm, etc)
Q24_Part_1_Analyze and understand data to influence product or business decisions
Q31_A_Part_9_No / None
Q36_A_Part_7_No / None
Q38_A_Part_11_No / None
Q40_Part_1_Coursera
Q40_Part_3_Kaggle Learn Courses
Q42_Part_4_Kaggle (notebooks, forums, etc)
Q42_Part_6_YouTube (Kaggle YouTube, Cloud AI Adventures, etc)
Q42_Part_8_Blogs (Towards Data Science, Analytics Vidhya, etc)