Assignment 2 Report

The purpose of this assignment is to train, validate, and tune multi-class ordinary classification models that can classify, given a set of survey responses by a data scientist, what a survey respondent’s current yearly compensation bucket is.

First of all, data cleaning is performed in order to prepare training and test set. In total of 371 columns, there’re a lot of them contain large number of missing values, for example Q17 and Q38 columns. Therefore, we first remove columns which have more than 30% missing values of the sample size, and keep the remaining 21 columns in our data. In addition, by looking through the original survey questions, we find out that columns 'Q8', 'Q26', 'Q41','Q11','Q13','Time from Start to Finish (seconds)' are unrelated to our target variable, yearly compensation. For instance, ‘Q8’ and ‘Q11’ are questions about personal preferences, which will not be useful for us to predict yearly salary in our common sense. Thus, after further removing those 6 columns, there’re 15 columns remaining in the dataset. Next, we check how many NA’s each column has, and it turns out that only column ‘Q7_Part_1’ and ‘Q15’ have some missing values. As there’re 2924 NAs in column ‘Q7_Part_1’, it is inappropriate to remove corresponding rows containing NAs directly. We couldn’t replace them with the most common value in that column, because, there’re only one non-missing value, “Python.” The best way to deal with them is to assign a new value, ‘Other’, to the missing values, so that ‘Q7_Part_1’ column indicating whether the respondents use Python or not on a regular basis. On the other hand, since there’re only 961 missing values in ‘Q15’, which is relatively a very small portion to our sample size 15391, we drop those corresponding rows directly without losing to much information.

Furthermore, we notice that even though there’s no missing value in ‘Q3’, the number of unique values is very large. If we encode those country names, the size of our data will explode. Thus, we count number of responses for each country and select only top 20 of them for dimensional reduction purpose. It turns out that the number of responses reduce to 10341, and it is enough for us to train and test the model. Finally, we need to convert categorical data into numerical data by encoding. For column ‘Q1’, there’re 11 different age ranges, so one-hot-encoding is inappropriate. Therefore, we create a dictionary to map each range to a number. For example, assigning '18-21' to 0, '22-24' to 1 and increasing the integer value for increasing age range. Similarly, we map years of using machine learning methods to integers for column ‘Q15’ as it has 9 unique values. For the rest of columns, since they have under 5 unique values, we simply use one-hot-encoding method to convert them into numerical data.

After encoding, the dimension of data is 10341 rows × 82 columns. Then, we could perform ‘train_test_split’ to create our training and test set, and the proportion of test set is set to be 30%. Before select features, standardization is performed to prevent features with wider ranges from dominating the distance metric. Since we are going to use Lasso Regularization for feature selection, standardization is required: Lasso regressions place a penalty on the magnitude of the coefficients associated to each variable. And the scale of variables will affect how much penalty will be applied on their coefficients. Because coefficients of variables with large variance are small and thus less penalized. Next, we use Grid Search to find the optimal value of hyperparameter in Lasso Regularizer and it turns out that alpha=0.1 and Normalize= False is the optimal choice. After fitting lasso regression to our training data, we select top 26 highest coefficient estimates and print corresponding features. As higher coefficient means higher correlation between features and target, we keep those 26 features for the future model training. Moreover, the feature importance can be measure by correlation plot. It shows that whether the participant is from the US has the highest correlation, with a value of 0.538233, to the yearly compensation among all the features. So, the original attribute, 'Q3' in the data are most related to a survey respondent's yearly compensation. The second most related attribute is 'Q1' with a value of 0.418955 correlation. Thus, in our training data, age and country seem to be more closely related to respondent's yearly compensation. Although I could use PCA instead for feature selection, it will lose useful interpretations as it transfers columns in our original data to for principal components. So, it is not straightforward to see which original feature have the highest effect on the target based on their correlation.

The third step is to implement ordinal logistic regression algorithm on the training data using 10-fold cross-validation. To evaluate the performance of models, we choose f1-score instead of accuracy. By looking at the value counts of target variable, Q25_bucket, the classes are highly imbalanced, with more than half of the response fall onto the class $0-9,999. If we select model based on accuracy, we might end up with a model that learn to classify all datapoints into the major class. However, as this model does not identify the minority class well, we cannot interpret it a good model. After performing cross validation, the Fold 1 has the highest f1 Score, 0.5055, and Fold 2 has the lowest one, Fold 2 f1 Score: 0.4351. There's not much difference in f1 scores across 10 folds. The average and variance of f1 Score is 0.46394 and 0.0191. Next, we try different values of hyperparameter C, which representing regularization strength, to see the model performance. The default value is 1, and model with C=1 has a mean f1 score of 0.46394. As we decrease the value of C to 0.001, the model's f1 score decrease to 0.450955. Since smaller values specify stronger regularization, we learn that the ordinal model perform worse with very strong regularizer. As we increase C value to 100, the mean f1-score slightly increase to 0.46408. It seems that model performance doesn't improve much with loosing regularization. The optimal value of C is 0.1 because it has the highest f1-score 0.46422. According to the Bias and Variance theory, the stronger the regularization, the less variance but higher bias, and vice versa. So, in order to keep the error at minimum level, the value of C couldn't be too high or too low.

We know that hyperparameters in our model are penalty, dual, tol, C, fit_intercept, intercept_scaling, class_weight, solver, max_iter, multi_class, verbose, warm_start, n_jobs, and l1_ratio. Besides regularization strength, C, solver is another important hyperparameter. Solver indicates the algorithm to use in the optimization problem. For small datasets, 'liblinear' is a good choice, whereas 'sag' and 'saga' are faster for large ones. So, we select different combinations of choice of 'C' and 'solver' to tune our model using grid search. The result shows that the optimal value is C=0.1 and solver': 'sag', with the mean f1 Score= 0.46422 and standard deviation of this metric= 0.0193. Noticed that to evaluate the performance of models, we choose f1-score instead of accuracy: By looking at the value counts of target variable, Q25_bucket, the classes are highly imbalanced, with more than half of the response fall into the class $0-9,999. If we select model based on accuracy, we might end up with a model that learn to classify all datapoints into the major class. However, as this model does not identify the minority class well, we cannot interpret it a good model. Furthermore, we obtain feature importance based on predictions from optimal model. The result is very different from what we got in section 2. Using the predicted value of training set, whether the participant is a Man is the one with highest correlation, which is below 0.03. The strength of correlation between sex and predicted yearly compensation becomes much weaker as compared to section 2, and whether the participant comes from the US is no longer the one with the highest correlation. Moreover, some features demonstrate negative correlations to the prediction which never exists in section 2 when using the true value of the target.

Finally, we use our optimal model to make classifications on the test set. The f1 Score on test set is 0.44, which is lower than the score on training set. Since our ordinal logistic model perform bad on both training set and test set (the f1 Score both not exceed 0.5), there might exist underfitting problem. We can further investigate the problem by plotting the distribution of true target variable values and their predictions on both the training set and test set. The plots illustrates that our model is unable to capture the relationship between the input and output variables accurately. For instance, on the training data, there're 2000 cases fall onto the class 0-9999 in our predictions compared to the true values. This misclassification leads to the result that only a few cases are predicted on the other compensation class, which isn't true in our real data. Similarly, our model incorrectly classifies more data onto 0-9999 class, whereas very few data are classified correctly to classes 70000-79999, 30000-39999, and 10000-19999. In order to solve underfitting problem, we might need to consider other classification models. Alternatively, try different feature engineering method and increase the number of features to improve the accuracy of predictions.