A2: Ordinal Classification Model

Shuman Zhao, 2024.11.9

Q1. Data Cleaning

First, I clean the data for Single-column Responses.

I removed the feature “Q5” because all values are NO, which means all respondents are not students. It’s useless for the analytics. Then, the target features “Q29” and “Q29_buckets” are removed because we already have the encoded target variable “Q29_Encoded.” The feature “Duration (in seconds)” is also removed because the time a respondent took to finish the survey has to be revealed to his bucket.

I modified the special feature “Q8” to fix the meaningless characters contained in its values. To make the analytics result meaningful, I replace the characters with the symbol “”. Besides, features “Q9”, “Q22”, “Q32”, and “Q43” are removed because they have a large percentage of missing values. Forcedly using them (for example, by fulfilling NA) will make the analytics result untrustworthy. The cause of this situation might be that many respondents are far away from academia and not familiar with machine learning, cloud platforms, and PTU. Missing values in feature “Q30” are removed directly because they only occupy 0.61% of all values and will not reduce the number of observations a lot.

Q29_Encoded	Alldata	NaN Q16
0.0	0.375860	0.612573
1.0	0.097345	0.092105
2.0	0.075467	0.068713
3.0	0.057030	0.039474
4.0	0.051745	0.026316
5.0	0.044985	0.040936
6.0	0.039086	0.019006
7.0	0.035521	0.021930
8.0	0.027286	0.013158
9.0	0.024213	0.005848
10.0	0.049656	0.020468
11.0	0.033063	0.013158
12.0	0.042035	0.011696
13.0	0.028638	0.005848
14.0	0.018068	0.008772

I also want to fulfill the missing values in feature “Q16” because the NA percentage of 8.41% is neither big nor small, and this feature “Years respondent use ML methods” is important to the model. Before doing this, I compare the distribution of the target variable “Q29_Encoded” and the distribution of the target variable “Q29_Encoded” where the “Q16” is NaN. According to the result on the left, there are large differences between them, which means the best solution is to fulfill missing values with "Unknown".

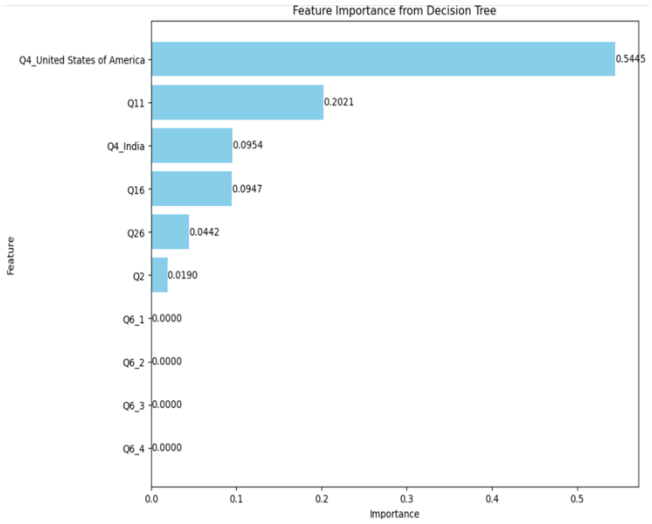
After that, I encode the ordinal features “Q2”, “Q11”, “Q16”, “Q25”, “Q26”, and “Q30” by replacing string values with numbers to maintain an ordered relationship between the values in the category. This is also because scikit-learn only takes numerical values as input in a NumPy array. For example, I replace the string value “55-59” with 55 for the feature “Q2”. Then I reduce the number of categorical values in feature “Q4” from 58 to 30 by masking the values (countries) that appear less than 50 times in the data. Finally, I create dummy variables (and drop one of each of the dummy variables because its value is implied) for each non-ordinal categorical feature: “Q3”, “Q4”, “Q8”, “Q23”, “Q24”, and “Q27”.

Second, I clean the data for Multi-column Responses.

For all features in this section, I replace missing values with 0 and others with 1. In this method, 1 means the respondent chose this option, and 0 means the respondent didn’t choose this option.

Q2. Exploratory data analysis and feature selection

First, for feature engineering, I generated a new feature “LanguageNumber” for both the training and test dataset by adding values of feature “Q12_1” to “Q12_15” together to reduce feature numbers for the model. The “LanguageNumber” counts the number of programming languages a respondent uses regularly instead of showing specific names of them.



Second, I use the Decision Tree method to show the order of feature importance. In Decision Tree, importance is calculated based on how much a feature contributes to reducing entropy across splits. According to the graph, the feature "Q4_United States of America" has the highest 0.5445 importance, which indicates that a respondent currently resides in the United States of America or does not have a high influence on the prediction of his salary. Besides, the feature "Q2", respondent's age, has only 0.0190 importance. This indicates that a respondent's age does not have a high influence on the prediction of his salary.

Third, I use Lasso regression combined with RFE (Recursive Feature Elimination) to select features. Lasso regression is a regularization technique that shrinks coefficients of less-important features to 0. RFE is a technique that selects features by recursively considering smaller and smaller sets of features until they reach the specified number. Combining these two techniques improves the modeling efficiency and reduces the risk of overfitting. In this case, I set the feature number to 10 and applied Lasso and RFE on the training data to select the 10 most important features for the model. The features I selected are: "Q11", "Q16", "Q26", "Q30", "Q4_Australia", "Q4_India", "Q4_United Kingdom of Great Britain and Northern Ireland", "Q4_United States of America", "Q23_Manager (Program, Project, Operations, Executive-level, etc)", and "Q24_Academics/Education". See the Appendix for distribution graphs of features "Q11" and "Q16".

Q3. Model Implementation

```
fold 1 : 0.3956723338485317
fold 2 : 0.41112828438948995
fold 3 : 0.39258114374034003
fold 4 : 0.41731066460587324
fold 5 : 0.43585780525502316
fold 6 : 0.36476043276661513
fold 7 : 0.3616692426584235
fold 8 : 0.44049459041731065
fold 9 : 0.43034055727554177
fold 10 : 0.4241486068111455
Average accuracy    :  0.40739636617682945
Variance of accuracy:  0.0007090576353510714
```

In this section, I fit the ordinal logistic regression with the training dataset. Then I perform a 10-fold cross-validation on the training data. Left is the model accuracy scores across the folds. The average accuracy is 40.74%, and the variance of accuracy is 0.00071. This means the model's performance accuracy does not vary a lot across the folds.

```
Bias-Variance Trade-off Analysis:
C=0.01: Avg Bias=62692.6141, Avg Variance=0.5434
C=0.1: Avg Bias=63445.9591, Avg Variance=0.6928
C=1: Avg Bias=63536.0338, Avg Variance=0.7121
C=10: Avg Bias=63547.7888, Avg Variance=0.7179
C=100: Avg Bias=63556.2853, Avg Variance=0.7135

Best C value in terms of bias-variance trade-off: 0.01
```

Then a Bias Variance Trade-off Analysis is performed by using a for loop to test different values of C. C is a hyperparameter in the Ordinal Logistic Regression model which is used to control the strength or the regularization applied to the model. A high value of C means less strength and a small value of C means large strength. According to the graph above, the strength of regularization decreases while the C value increases from 0.01 to 100. In this case, the average variance increases because the coefficients of the model are increasing, which leads the model to fit more to the training data. The average bias should decrease in this case based on the bias-variance trade-off. However, its value remains approximately stable when the average variance is increasing. This might be because the logistic model approaches its limits to predict the target variable accurately due to its limitations in capturing complex relationships. Besides, elements like noise and multicollinearity between features can also be the reason for this situation. The best C value for the model is 0.01, which leads to the best balance of bias-variance trade-off.

Scaling of features is also important for the task. This is because the ordinal logistic regression model is sensitive to the scale of features, especially when applying regularizations. Larger scale features will make their coefficients become larger to fit the data and will receive more penalties when applying regularization, which applies equally to all coefficients.

Q4. Model Tuning

Unfortunately, accuracy is not a suitable performance metric for the model. This is because the dataset has a class imbalance problem, which means for a feature, some classes are significantly more frequent than others. In this case, the accuracy can give a misleading result. According to the graphs "Distribution of 'Q16'" and "Distribution of 'Q29_Encoded'", there are large differences between the frequency of some of the classes. The significant difference can cause a misleading impression from accuracy. Therefore, to test model performance more properly, I use the F1-score instead of accuracy.

In this section, I use Grid Search to select the best C and solver for the model tuning. The selected best values of hyperparameters C and solver are 10 and "lbfgs". The reason I chose them is that C helps find the best balance between bias and variance, and Solver helps to find the best algorithm to minimize cost function (since different algorithms have different strengths and weaknesses. There are also other hyperparameters of the ordinal logistic regression model that I don't use for tuning. For example, "max_iter", which is the maximum number of iterations for the solver to converge; "fit_intercept", which means whether or not to include an intercept term in the model; "class_weight", which includes options balanced or custom weights to handle imbalanced datasets; and "penalty" which includes regularization type like L2, L1, and elasticnet.

Then, to figure out the order of feature importance of the model, I calculate the average of coefficients from each binary classifier. The larger the average, the more of the importance. According to the "Feature Importance for Ordinal Logistic Regression" graph, the most important feature is still "Q4_United States of America" with an unchanged importance value of 0.545. However, feature "Q16" replaces "Q11" to become the second important feature with an importance value of 0.202. My model relies on features "Q4_United States of America", "Q16", "Q11", "Q26", "Q30", and "Q4_Austrilia" a lot for prediction.

Q5. Testing and Discussion

```
Training Set Performance:
Accuracy: 0.4156
Precision: 0.2883
Recall: 0.4156
F1 Score: 0.2975

Test Set Performance:
Accuracy: 0.3999
Precision: 0.2449
Recall: 0.3999
F1 Score: 0.2790
```

On the left is the comparison model result between the training and test set. All metrics for training set performance are slightly larger than the test set, which indicates a slightly overfitting problem. To further improve the model performance, I'll try a higher penalty in the regularization process to either reduce the feature numbers or the value of coefficients.

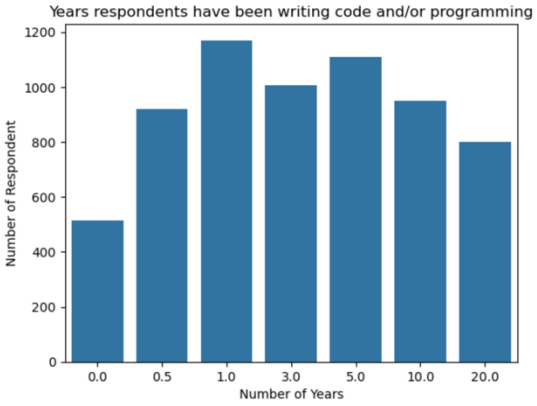
The last two graphs of prediction distribution for the training set and test set in the Appendix also support the testing result. We can see that the True Values and Prediction curves in the training set are a little bit closer to each other than in the test set.

In conclusion, the ordinal logistic regression model I created predicts a respondent's current yearly compensation bucket with a not-very-strong performance. The features of whether a respondent is in the US or not, years using ML methods and programming, and how much has been spent on ML plays a significant role in the prediction. The highly imbalanced dataset also causes challenges in model accuracy.

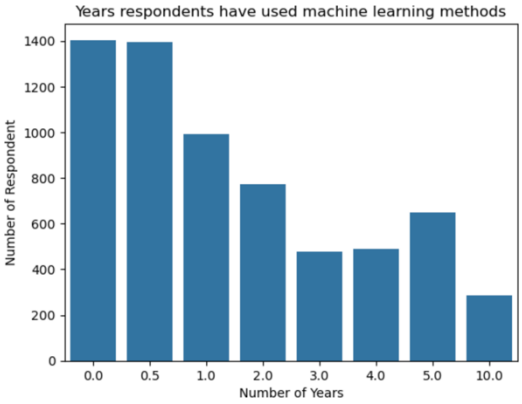
Appendix

Q2. Exploratory data analysis and feature selection

Distribution of "Q11"

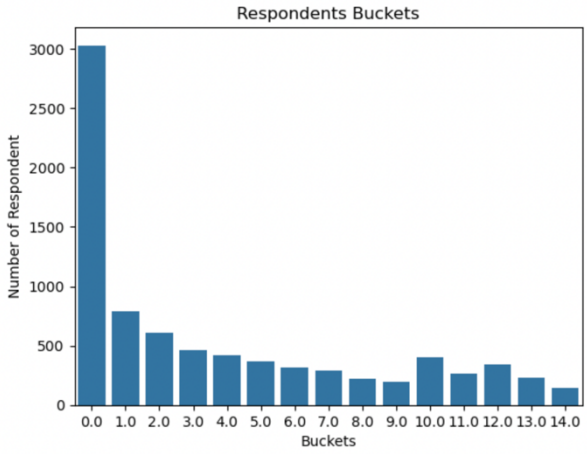


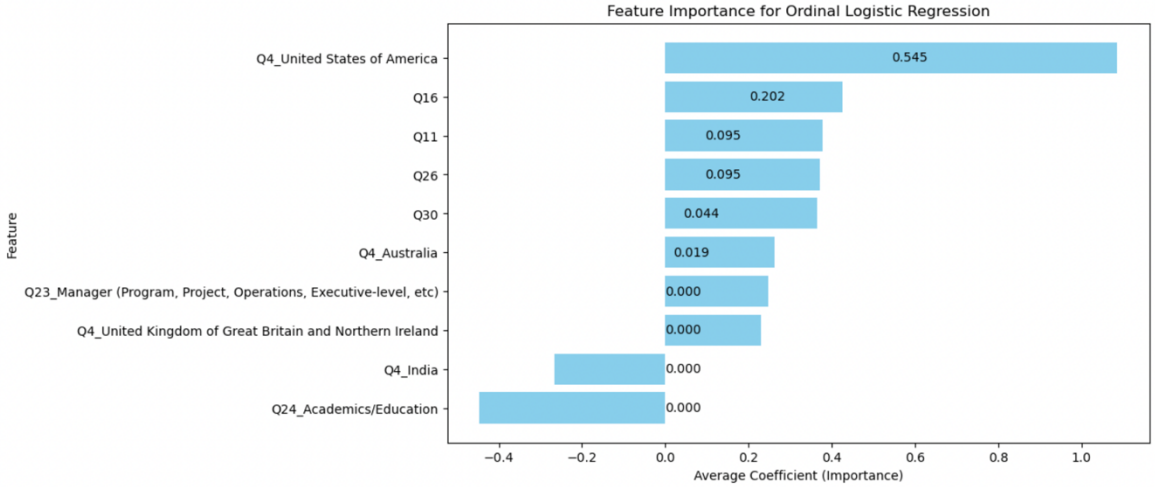
Distribution of "Q16"



Q4. Model Tuning

Distribution of "Q29_Encoded"





Q5. Testing and Discussion