Midterm Report

Huiyu Yang

March 2023

1 How do I subset the dataset?

The Python codes can be found in the file named **part1_clean_data.py**.

After I obtain the survey dataset, I check if there is empty cells at first. I find there are 12 observations contain empty cells, so I drop these observations. I then calculate the summary statistics of the dataset and find some observations have negative ages. I drop observations that have negative ages. Besides, I find some variables do not have summary statistics, such as personality2, personality4 and personality5. Further investigation revealed that these variables contained letters within the numbers, where "1" and "0" were incorrectly written as "I" and "O". These errors were corrected. Finally, I conducted checks on gender, region, horoscope, and game to ensure that no problematic observations were present. No issues were found.

To facilitate ease of data cleaning for future programs, I created a template program named **clean_data_template.py**.

2 How do I choose the model and the parameters?

The Python codes can be found in the files named **part1_try_models.py**, **part1_logistic_regression.py** and **part1_svm_linear.py**.

I mixed the process of variable selection and model selection together. In order to get a general idea of which models perform better, I included all variables in the regressions to obtain accuracy scores. Prior to this, I noticed that several variables were not numeric. Therefore, I converted "game" into

a categorical variable, and "gender", "region", and "horoscope" into dummy variables.

In **part1_try_models.py**, I tested several models including KNeighborsClassifier, logistic regression, linear SVM, poly SVM, decision tree, and random forest. KNeighborRegressor and linear regression were not included as these models cannot generate categorical predictions. Upon rough calculations, I found that the KNeighborsClassifier, poly SVM, and decision tree models had relatively lower accuracy scores between 0.68 and 0.79, while the logistic model, linear SVM model, and random forest had relatively higher accuracy scores between 0.9 and 0.95.

To check for overfitting in models with higher accuracy scores, I plotted the mean absolute error and accuracy scores for both the train and test data. I also plotted these graphs for models with lower accuracy scores for comparison.

After examining the accuracy score graphs of the random forest and decision tree, I discovered that they become overfitting when the max_depth surpasses 10. The gap between the train data's accuracy score and test data's accuracy score considerably increases beyond this point. Furthermore, when the max_depth goes beyond 15, the train data's accuracy score nearly reaches 1. So, I dropped the random forest.

As for the other high accuracy score models, the differences between the train data and test data are insignificant and do not change much when compared to the low accuracy score models.

Therefore, my attention is now focused on the logistic model and the linear SVM model.

I analyzed the feature importance of variables in the logistic and linear SVM models in **part1_svm_linear.py** and **part1_logistic_regression.py**. The feature importance was calculated by taking the average coefficients of each variable. Then, I ranked the features in order of their importance and calculated the accuracy scores for regressions that contained only the most important feature, the top two important features,... to all features.

In the logistic model, the highest accuracy score was achieved with the top 14 features. In the linear SVM model, the highest accuracy score was achieved with the top 21 features, which was much higher than the logistic model's accuracy score. To explore if other combinations of features could produce better accuracy scores, I dropped all horoscope dummies in the 21 features and found that the remaining features generated a higher accuracy score. Additionally, I used the top 14 features in the logistic regression to

run the linear SVM model and found that these 14 features produced an even higher accuracy score.

I also checked for overfitting problems after reducing the number of features and found that there were none. Ultimately, I selected the top 14 features from the logistic regression to train the machine using the linear SVM model.

3 Limitations of my Model

My exploration of the available data may not have encompassed all possible combinations of features that could lead to the highest accuracy score. However, I have done my best given the vast number of potential feature combinations.