MF 815 Final Project: Mutual Fund Style Classification from Prospectus

Sike Yang

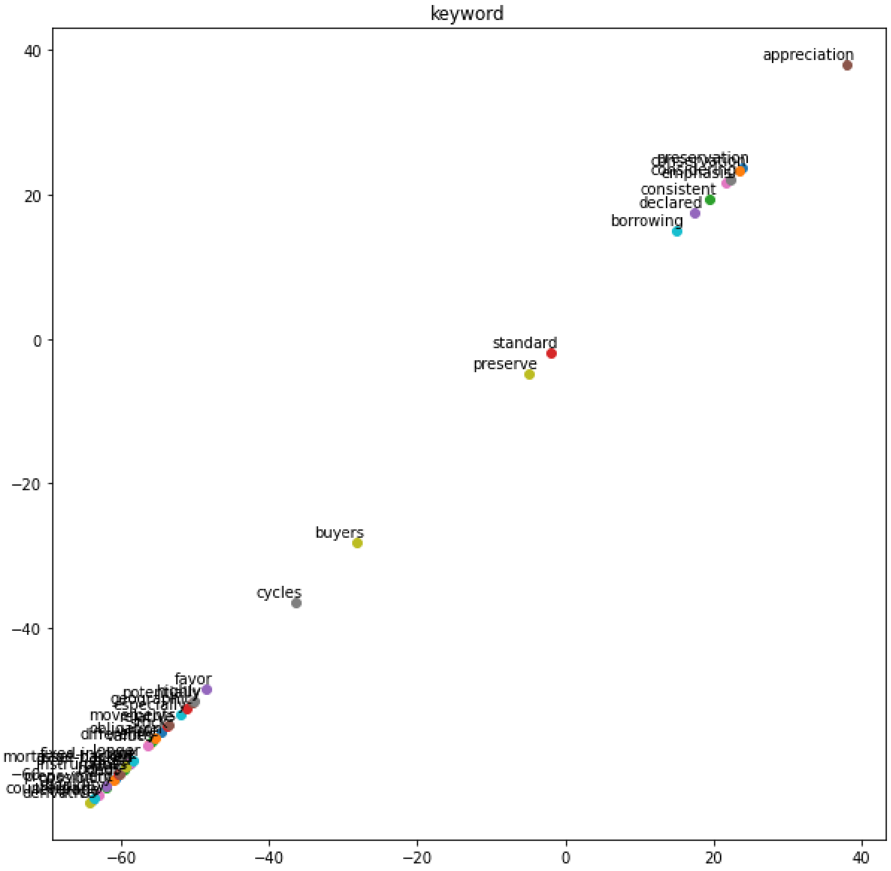
Summary

In this project, the aim is to use the mutual fund prospectus to classify the investment style of a mutual fund. The dataset contains a collection of mutual fund summaries, which will be used to build a word embedding dictionary by applying a skip-gram model, and a CSV file on 'Mutual Fund Labels.' To achieve the classification goal, the dataset will be split into training , validation and testing sets after sentence extraction, and techniques will be applied to deal with imbalanced data. Four classification methods will be selected, and their hyperparameters will be tuned. The performance of these models will be evaluated on the validation set, and the results will be used to identify the best model for classification. All these processes will be accomplished using Python.

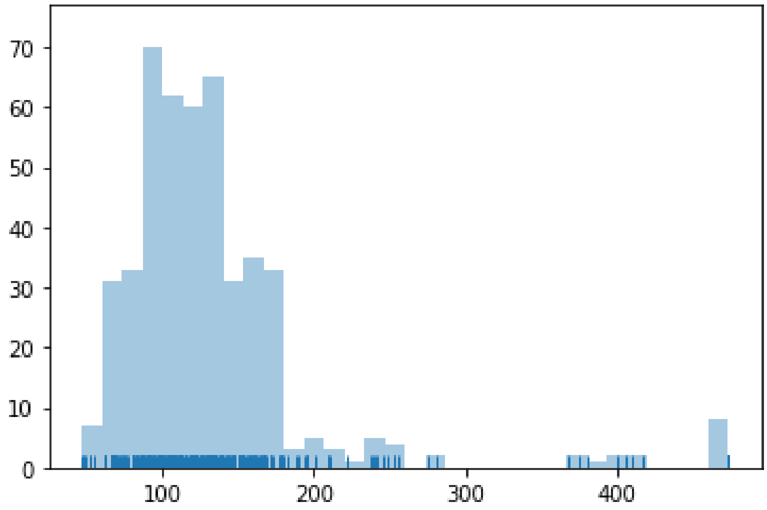
After model selection, it was found that the random forest had the highest accuracy (80%) and f1 score (0.86), indicating the best performance. After removing the "Long Short Funds (High Risk)" category and considering only the other three categories, the new classification algorithm performed similarly. Therefore, the random forest model with four categories was chosen to predict the test data, achieving 67% accuracy and 0.73 f1 score.

Data Processing

The dataset consists of 466 mutual funds and comprises two parts with no missing values: Mutual Fund Summaries and Mutual Fund Labels. To build a word embedding dictionary for the mutual fund summaries, I used a skip-gram model that considered only the 5000 most frequently occurring words and set the dimension of the word vectors to 50. After training the skip-gram model, I obtained a work vector and each word has its unique id. I then built a function that evaluates the distance between words to obtain the 10 closest words for each fund style. For balanced funds, I chose "preservation" as the central word to obtain neighbors based on its feature. For fixed income long-only funds, I selected "bonds" and "stocks" for Equity Long Only, and "leverage" for long-short funds. After obtaining these 40 neighbor words, I set them as the keyword list to build the knowledge base.



The knowledge base is created by taking the 5 closest neighbors of each keyword in the word vector. After obtaining the knowledge, to extract sentences related to the investment strategy, I build a scoring function that counts the number of words that are in the intersection of the knowledge base and the sentence. Then, each summary is turned into a document related to the fund style, but the model inputs should have the same dimension. To determine the appropriate length, I visualized the distribution of the number of words per document. Based on the figure, I set the length of the document series to 180.



The Mutual Fund Labels csv contains only the 'Investment Strategy code' column. The dataset includes 247 equity long-only funds, 130 fixed-income long-only funds, 84 balanced funds, 4 long-short funds, and 1 commodities fund. Since the classification is only for the first four categories, I excluded the commodities fund. I converted the categorical labels to numeric values: 0 for balanced funds, 1 for equity long-only, 2 for fixed-income long-only, and 3 for long-short funds.

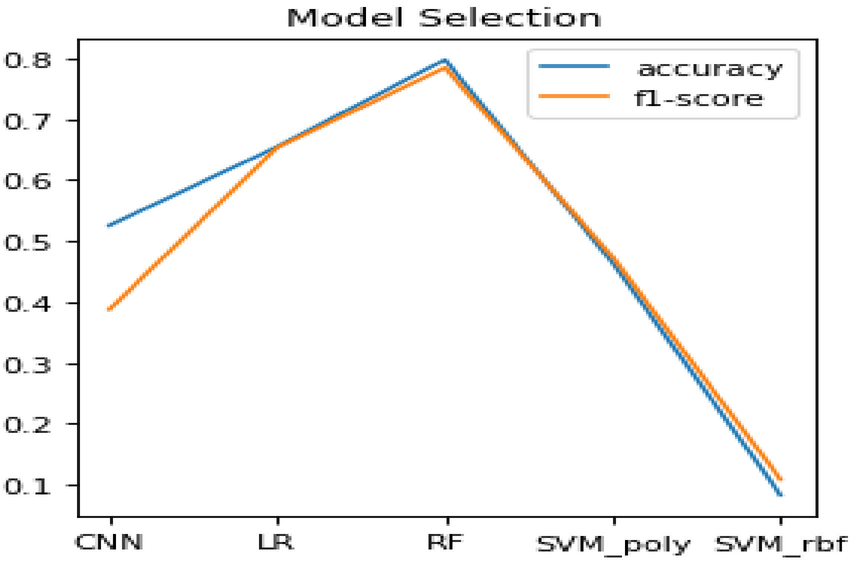
Given that there are only four "Long Short Funds (High Risk)" in the dataset, I split the training, testing, and validation datasets separately into low-risk and high-risk funds to ensure that each dataset had at least one fund in each category. I then merged the datasets to fit the model. The ratio for training, testing, and validation was 80%, 10%, and 10%, respectively.

Model Selection

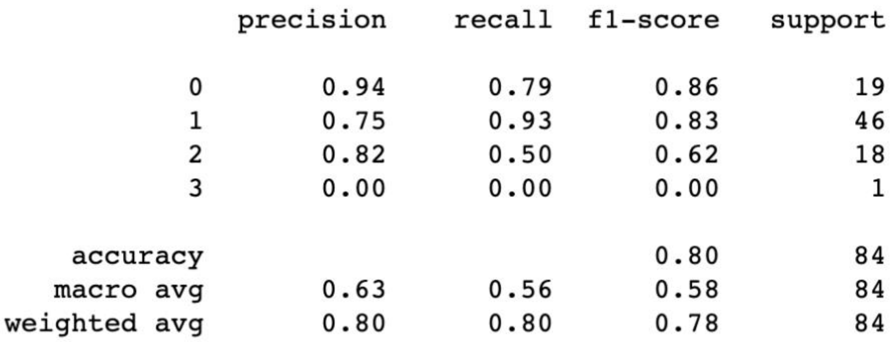
The dataset is high dimensional, and the target variable is multi-classification and imbalanced. Therefore, the model needs to have the ability to capture non-linear decision boundaries, handle high dimension classification tasks, and be adapted for text classification. In this study, I have chosen four models: Random Forest, Convolutional Neural Network, Logistic Regression, and Support Vector Machine, to perform the classification.

To improve the model performance, I use SMOTE to deal with the imbalanced data. To optimize the Random Forest and Logistic Regression models, I apply GridSearchCV from Scikit-Learn to obtain the best combination of hyperparameters. For the CNN model, I apply 10 epochs to ensure that the model converges to the highest accuracy. For SVM, I compare the poly kernel and rbf kernel to get the better one. And the best model selection is based on the accuracy and f1 score of validation dataset prediction.

Result



According to the figure, it can be found that random forest has the highest accuracy which is 80% and f1 score which is 0.86, so I choose random forest to do the predict. The accuracy for balanced fund prediction is the highest, then is equity only fund and fixed income fund, while the performance in long short fund prediction is poor. Considering the extremely low amount of long short fund might influence the model performance in a negative way, I remove the long, short funds and build a new classification algorithm which only focus on classification of the other 3 types, the model accuracy turned to 77%, which is not higher than the previous one.

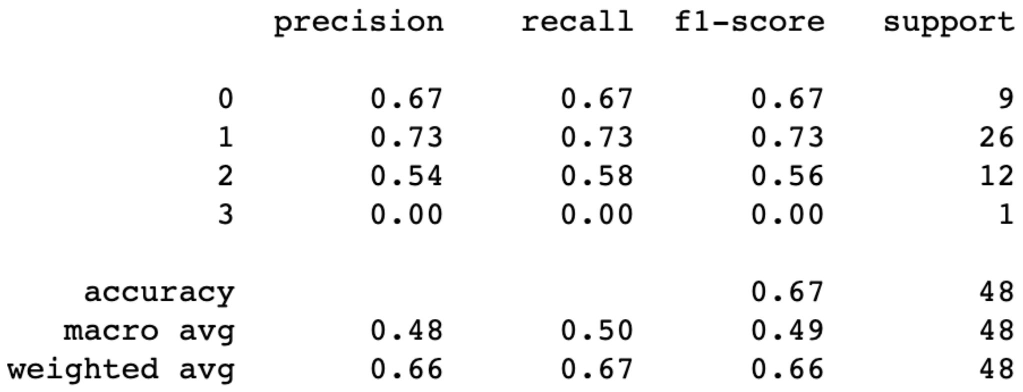


```
              precision    recall  f1-score   support

           0       0.94      0.79      0.86        19
           1       0.75      0.93      0.83        46
           2       0.82      0.50      0.62        18
           3       0.00      0.00      0.00         1

    accuracy                           0.80        84
   macro avg       0.63      0.56      0.58        84
weighted avg       0.80      0.80      0.78        84
```

The test data has 48 mutal funds, and the prediction results in 9 balanced fund, 26 equity only fund, 13 fixed incomes only fund and 0 long short only fund. The accuracy of the test data prediction is 67% and the f1 score is 0.67.



```
              precision    recall  f1-score   support

           0       0.67      0.67      0.67         9
           1       0.73      0.73      0.73        26
           2       0.54      0.58      0.56        12
           3       0.00      0.00      0.00         1

    accuracy                           0.67        48
   macro avg       0.48      0.50      0.49        48
weighted avg       0.66      0.67      0.66        48
```