4. Experimental setup:

4.1 K-Nearest Neighbors (KNN)

In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression[1]. In both cases, the input consists of the k closest training examples in the feature space. In this project, k-NN was used for classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). Thus, the k value is the most important parameter in this model.

To begin with, we read the preprocessed data using pandas. By ".head()" function, we found that the column named "Unnamed: 0" repeat the index of the .csv files which need to be dropped in both train and test data. Then, the last column is defined as 'labels' that indicates weather the disaster is real or not, using 1 and 0 separately. The rest of the columns became the features which is assigned to be x of train data.

After feature engineering, the KNeighborsClassifier from sklearn package was applied for KNN modeling. The prediction was created with the x of test data. We saved the result in a csv file and made it match the format for submission in Kaggle website. Besides, upload the result of KNN with default and get the score of accuracy as our base line.

The next step is to find the best k parameter for KNN. In this part, we used train data in the last step for both training and testing without submitting to Kaggle. The data was split into 5 folds using KFold package from sklearn. Each fold in 5 folds worked as a test data and the rest of folds became the train data every loop. We set the k range from 5 to 20 and all the 5 folds would have 15 iterations for different k value. Then we saved and compared the score of accuracy in each fold. Thus, we got 5 best k for 5 different folds and we picked the best k by majority rule.

Moreover, we run the model again and replace the default k with the best k we found. Then, the ROC curve and PR curve were used to evaluate our result.

4.2 Logistic Regression

In statistics, the logistic model is used to model the probability of a certain class or events would be assigned a probability between 0 and 1 and the sum adding to one. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression.[2] Here is the logistic curve relates the independent variable, X, to the rolling mean of the DV, P (Y).(Fig.1)

$$P = \frac{e^{a+bX}}{1+e^{a+bX}}$$

Figure 1. The equation of logistic curve.

The pre-processed and cleaned train data, just like data in KNN, was used to run the logistic regression model from sklearn. The prediction was submitted to Kaggle and we got the accuracy score with default as a baseline.

Furthermore, we tried to improve our accuracy score by using three solvers in L2 penalty, including liblinear, lbfgs and newton-cg. Besides, gradient iteration number was another parameter we take into account. The candidate iteration number contains 100, 150, 200, 250, 300. Thus, we used two loops to find the proper solver as well as the best iteration number. The original train data was split into 5 folders having both train and test groups. The result of parameter was saved as a csv file.

Additionally, we also calculated the class weight, then we run the logistic regression model again and replace the default setting with the best parameter we found. Then, the ROC curve and PR curve were used to evaluate our result.

5. Results

5.1 K-Nearest Neighbors (KNN)

5.1.1 The parameter in KNN

We compared the score of accuracy in each fold. Then, we got 5 best k for 5 different folds and we picked the best k by majority rule. Thus, the best k is 12 that displayed in Figure 3.

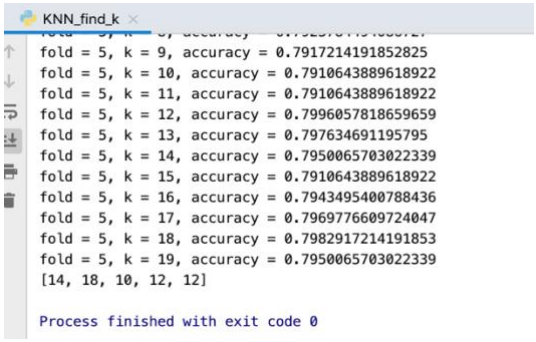

Figure 3. The result of best k value for KNN.

5.1.2 Using all train data to train the model

The score 0.77300 is the baseline of KNN's performance with default (Figure 2).

Figure 2. Kaggle Accuracy Score of KNN (with default).

By submitting the KNN result csv file to Kaggle, it returns the score and the screen shoot of it is shown as Figure 3. Compare the two scores, we find that the accuracy has been improved by 3%.

Figure 3. Kaggle Accuracy Score of KNN (k=12).

5.1.3 ROC curve and PR curve

For KNN model, AUC of ROC curve is 0.85. The true positive rate is calculated as the number of true positives divided by the sum of the number of true positives and the number of false negatives. The AUC describes how good the model is at predicting the positive class when the actual outcome is positive.

The AP for PR curve is 0.82. Compute average precision (AP) from prediction scores This score corresponds to the area under the precision-recall curve.

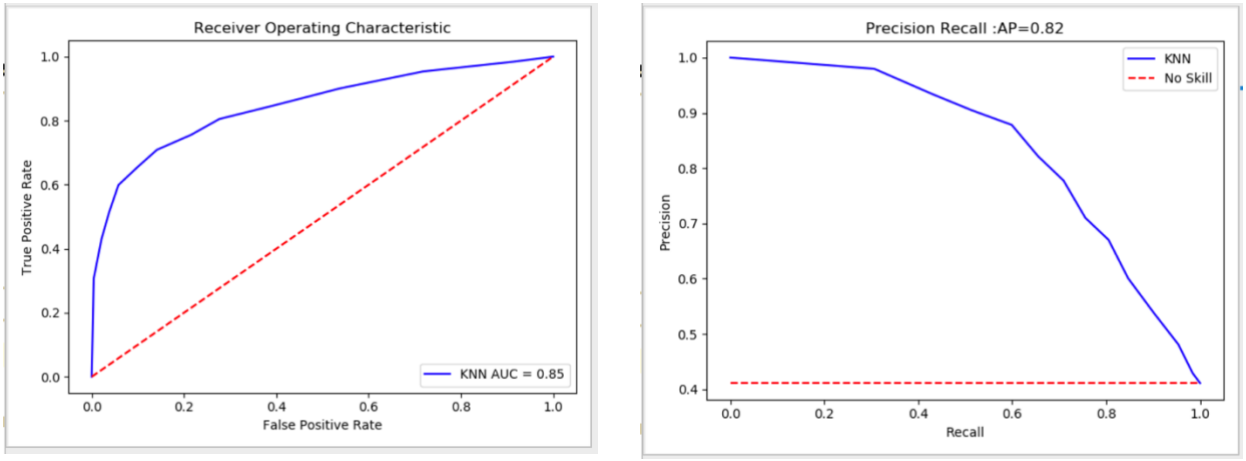

Figure 6. ROC Curve and P-R Curve of KNN Model

5.2 Logistic Regression

5.2.1 The parameter in Logistic Regression

Among “liblinear”, “lbfgs” and “newton-cg”, “lbfgs” showed the higher average accuracy within five folders. Besides, iteration number equals to 150 performed best among gradient iteration numbers.

```
fold = 5, iteration num = 200, solver = lbfgs, accuracy = 0.7910643889618922
fold = 5, iteration num = 200, solver = newton-cg, accuracy = 0.7917214191852825
fold = 5, iteration num = 250, solver = liblinear, accuracy = 0.7897503285151117
fold = 5, iteration num = 250, solver = lbfgs, accuracy = 0.7897503285151117
fold = 5, iteration num = 250, solver = newton-cg, accuracy = 0.7917214191852825
fold = 5, iteration num = 300, solver = liblinear, accuracy = 0.7897503285151117
fold = 5, iteration num = 300, solver = lbfgs, accuracy = 0.7890932982917214
fold = 5, iteration num = 300, solver = newton-cg, accuracy = 0.7917214191852825
iteration=100, average accuracy=0.7823003334187516
iteration=150, average accuracy=0.7826946090734882
iteration=200, average accuracy=0.7826068900022979
iteration=250, average accuracy=0.7824317682237808
iteration=300, average accuracy=0.7823441641939954
solver=liblinear, average accuracy=0.7824754551972686
solver=lbfgs, average accuracy=0.782606895754368
solver=newton-cg, average accuracy=0.7823443079957514

Process finished with exit code 0
```

Figure 4. The parameter for LR Model. “Iteration=150” and “solver= ldfgs” performed best among these parameters.

5.2.2 Using all train data to train the model

The score 0.80163 is the baseline of LR’s performance with default (Figure 5):

Figure 5. Kaggle Accuracy Score of Logistic Regression (with default).

By submitting the Logistic Regression test csv file to Kaggle, it returns the score and the screen shoot of it is shown as Figure 6. The score =0.80061 shows that the parameter changing didn't work well.

Figure 6. Kaggle Accuracy Score of Logistic Regression (iteration=150, solver= “ldfgs”)

5.2.2 ROC curve and PR curve

For Logistic Regression, AUC of ROC curve is 0.86. The true positive rate is calculated as the number of true positives divided by the sum of the number of true positives and the number of false negatives. The AUC describes how good the model is at predicting the positive class when the actual outcome is positive.

AP, here is 0.85, is a single number used to summarize the PR curve. Compute average precision (AP) from prediction scores This score corresponds to the area under the precision-recall curve.

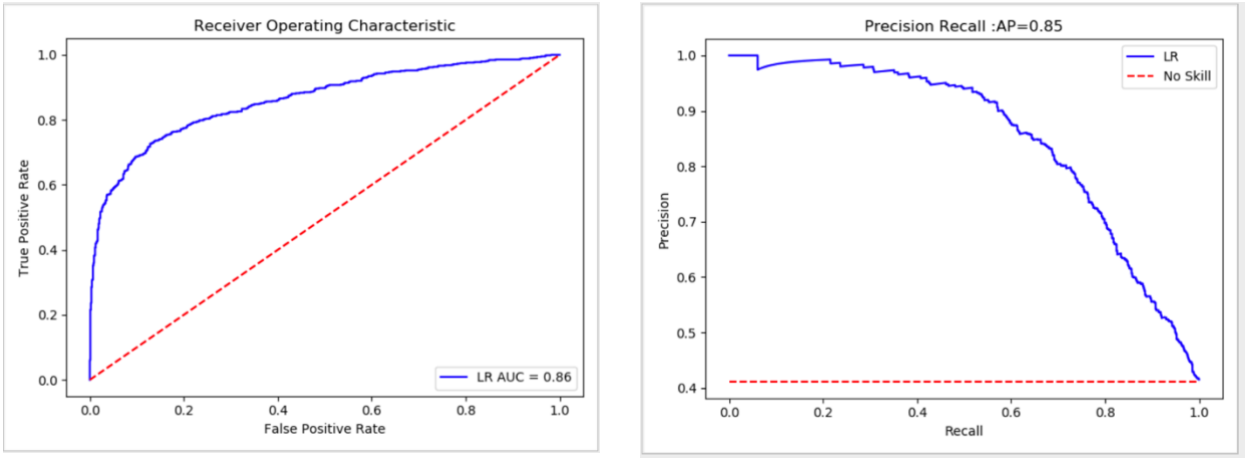

Figure 6. ROC Curve and P-R Curve of Logistic Regression Model

For KNN model, the accuracy score is 0.80470 which increased 3% when making k =12. Besides， both the ROC curve and PR curve display that KNN model performs well.

For LR model, unfortunately, the accuracy rate does not increase after changing the default setting, which may indicate two things. One is that the LR model overfitted for our data. It shows a better performance for fold 2 than other fold which will influence the average score and misleading us for parameters. The other is that the accuracy score of around 0.80 has shown all the potential for the pre-processed data we worked on.

References

[1] Altman, Naomi S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression" (PDF). The American Statistician. 46 (3): 175–185.
[2] Tolles, Juliana; Meurer, William J (2016). "Logistic Regression Relating Patient Characteristics to Outcomes". JAMA. 316 (5): 533–4.