CS 412 Project Final Report

Yue liu (yueliu5)

Shiming Song (ssong38)

I. Algorithms:

- Adaboost: We first tested Adaboost by using external library, which gave us around 80% accuracy. We then implemented Adaboost by calling the DecisionTreeClassifier in Python Sklearn. Besides, we modified the original algorithm by introducing an additional parameter as a threshold for differentiating the two classes from the raw output of Adaboost.

- Random Forest: We used decision tree as basic classifier and do bagging to implement Random Forest.

- K-nearest Neighbor: We implemented KNN by calling scipy.sptial.KDTree to find the nearest neighbors. We also introduced an extra variable as a threshold to adjust the raw output from KNN. Although not working so well, KNN can be used in Adaboost as a basic classifier.

- Among all of them, it turned out that Random Forest (decision tree based) and Adboost gave the best results.

II. Features and Data Preprocessing:

- Feature Engineering
    - We used all of features in the file features.csv.
    - We used pandas.get_dummies() to convert categorical variables (here it is the 201st variable in features.csv) to numerical variables. Before this, we encoded the categorical variable to integers, but we do not think it is appropriate because when encoding, order as information will be introduced to the data set, which did not existed in the original data.
    - We used Sklearn.feature_selection and utilized the SelectKBest function to find the best k features for classification.
- Preprocessing
    - In training data part, we replaced the NAN values with their corresponding class means.
    - In testing data part, we replaced the NAN value with the mean of the feature.

- We also implemented randomized cross validation for parameter selection.

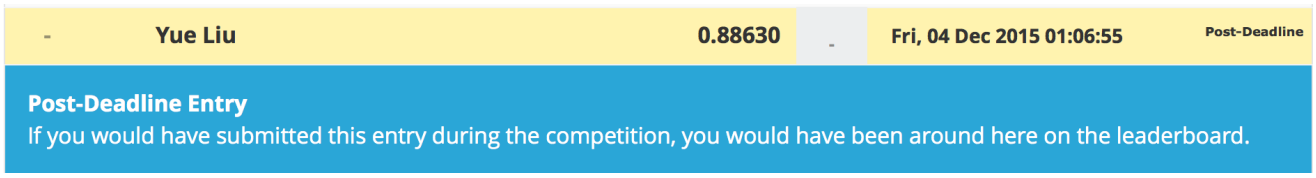
III. Work distribution:

- Yue Liu is responsible for data preprocessing and implementing Adaboost, Random Forest and Randomized Corss Validation.

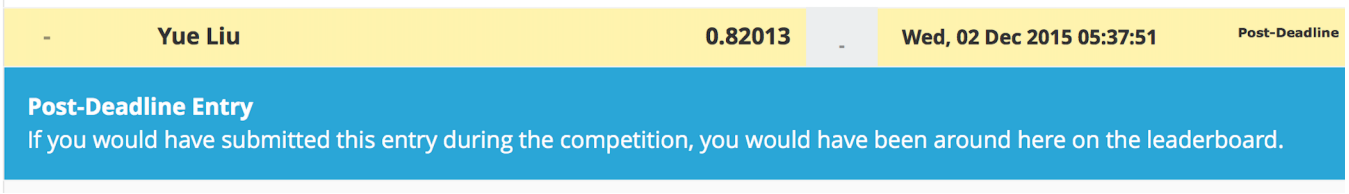
- Shiming Song is responsible for implementing K-Nearest Neighbor and writing the final report.

IV. Results

Accuracy for revised random forest:



Accuracy for revised Adaboost (Tree based):



V. Required Packages:

Numpy

Scipy for building KdTree

Sklearn.feature_selection for feature selection

Sklearn.cross_validation for building randomized cross validation

Sklearn.tree for building decision tree

Sklearn.ensemble, sklearn.neighbors for initial testing

VI. Discussion:

- Why algorithm like KNN does not work?

We think this is because positive instance is too sparse in the data, so after we build a traditional KNN classifier, it is very hard for it to get major votes of 1, so it can rarely output 1.

- What changes did we make to the classifiers?
  Mostly we added a threshold parameter into the classifier constructor or prediction function so that we can adjust the strictness of classifying. For example, in KNN, after we added the threshold, we can specify that as long as 10% of the K neighbors are Robots, we classify the querying point as Robot. This gave us a lot of flexibility.