Classification_Report

Bowen Liu

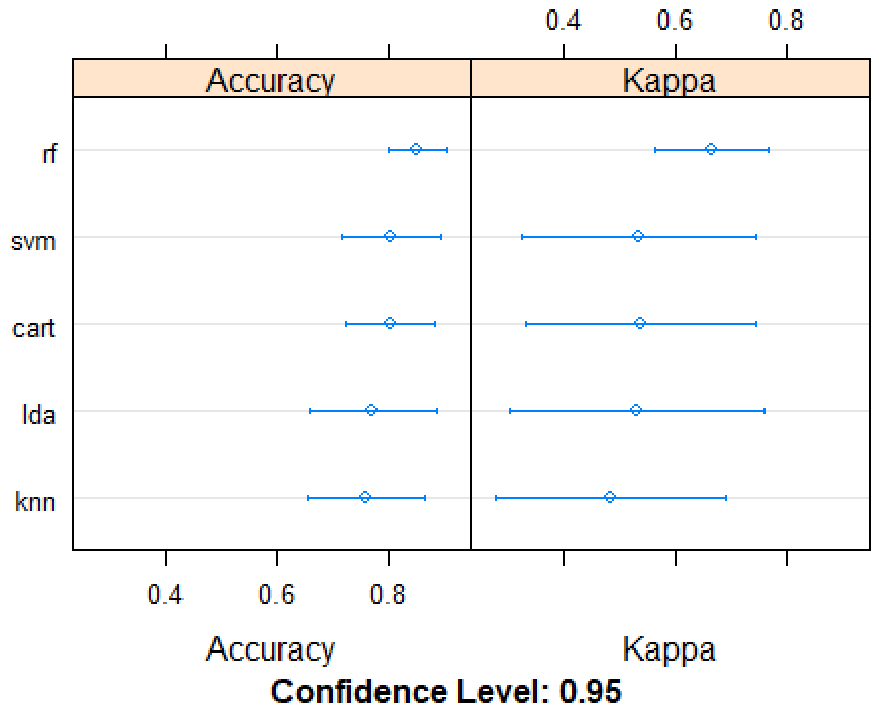
8/1/2019

Data Preprocessing and Model Selecting

In order to using the data set to predict category, we first cleaned and wrangled the data by removing the id number and change the target of our prediction, category, to a categorical variable. We can see that, the data has 88 observations and 310 features, which indicate a high dimentionailty:

```
## [1]  88 310
```

In general, as we discussed in the ISLR book, when dealing with classification problems, there are several powerful models, including Logistic Regression, LDA, Cart, KNN, SVM, and tree methods like random forest. We want to compare their performance on this specific data set. We used the methods of 10-fold cross validation to estimate their accuracies. Their performances are shown below:



As a result, as we can see in the plot above: if we use the full model, regarding the high dimensionality and few number of obervations, random froest has the best performance. Thus, we choose random forest as our model and improve it in further steps.

Model Tuning

For the number of trees, due to the high dimensionality, we want chose large number of trees to deal with the large possible variances. We tried 100, 200, 300, 400, and 500 trees. From the summary below, we can see that the best number of trees, given by random forest, is 400, as its median accuracy is high, median kappa is low, and variation is smaller.

```
##
## Call:
## summary.resamples(object = tree.num)
##
## Models: 100, 200, 300, 400, 500
## Number of resamples: 10
##
## Accuracy
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## 100 0.6000000 0.8020833 0.8888889 0.8433333 0.8888889 1.0000000    0
## 200 0.6250000 0.7777778 0.8375000 0.8161111 0.8888889 0.8888889    0
## 300 0.5555556 0.7777778 0.8750000 0.8316667 0.9750000 1.0000000    0
## 400 0.6250000 0.7777778 0.8750000 0.8386111 0.8888889 1.0000000    0
## 500 0.6666667 0.7777778 0.8819444 0.8636111 0.9750000 1.0000000    0
##
## Kappa
##            Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## 100  0.09090909 0.4785714 0.7272727 0.6299101 0.7587413 1.0000000    0
## 200  0.25000000 0.4166667 0.6298701 0.5685498 0.7272727 0.7272727    0
## 300 -0.20000000 0.5000000 0.7142857 0.6011180 0.9456522 1.0000000    0
## 400  0.00000000 0.4428571 0.7142857 0.6054545 0.7272727 1.0000000    0
## 500  0.18181818 0.4250000 0.7417582 0.6732559 0.9456522 1.0000000    0
```

Modeling Fitting and Model Interpretation

We then used the random forest model with number of trees of 400 to deal make our prediction. Our output, namely 'uploadl.csv', has a 0.94736 accuracy on Kaggle. Our model, however, has less interpretability, as it is basically putting together 400 trees and averge them to get the best best performace to overcome the problem of overfitting.

Discussion of the Model Performance

When it comes to the model performance, the accuracy is high. However, due to the limited number of observations, especially when the number of observations is few, the model could be overfitting to some extent. Although we used the random forest to partly overcome this problem, as we discussed above, this model could also be improved by using techniques like PCA to reduce the curse of dimensionality, if time permitted.

##reference - "R Random Forest Tutorial with Example",GURU99 Inc, 7/28/2019
https://www.guru99.com/r-random-forest-tutorial.html