# CSCI-P556
# Fall 2018
# Assignment 3
# Due 11:59PM, Nov. 2, 2018

Vishal Singh (singhvis)

November 5, 2018

## 1 Introduction

A training dataset of 2000 samples and 500 features and a testing dataset of 600 samples and 500 features are provided. We after preliminary EDA, we are training baseline models on training data and checking score on testing data (which we are treating as validation). Next we perform feature engineering and tweaking the parameters of the models to improve accuracy on the test data. The main challenge on this dataset is feature engineering as the number of features are a lot and we do not know the physical significance of the features.

## 2 Exploratory Data Analysis

Findings about Features-

1. Number of features: 500
2. Number of continuous features:500
3. The density plot(Figure 1) of most of the features resemble a normal with mean and median being close to each other
4. Columns with NAs: None
5. Columns with negative/trash value: None
6. Number of features with less than 5 distinct values in train dataset: f91, f277, f424 (These features can be treated as continuous variables but for the following analysis I've treated them as continuous due to their values being comparable to values in other columns)
7. There are a lot of columns having outliers[1](Figure 2), I have decided not to treat then because of following reasons:
   (a) We do know the physical significance of the values, it might be that columns values might be an outliers according to the definition of outlier defined by us but might be a meaningful value
   (b) Treating outliers in all the column might decrease the bias in the data, leading to overfitting the training data
8. Columns with unique identifiers: None

9. On checking correlation between the features, we get **47 pairs** of features which have correlation of 0.5 or more. A new dataset has been created after removing one feature from the pairs.

Findings about Labels-

- Distinct Label: 2
- Values of Label: -1, 1
- Proportion of each label in train data: 50:50 (balanced)
- Proportion of each label in test data: 50:50 (balanced)
- Since, the labels are balance, we can use accuracy score to check accuracy of model. If the data was imbalanced then we would've checked f-score

Since the data does not have any NaNs, unique identifiers, we do not make any changes to the dataset before applying the baseline models

## 3 Baseline Models

Since the data has 2 distinct labels, I've chosen Logistic Regression, Random Forest Classifier, KNN Algorithm, and Support Vector Machines for predicting the output label for test set.
Baselines models are trained on normalized train dataset and then tested on test dataset.

Performance on baselines models is as follows:

| Model | Baseline Accuracy |
|---|---|
| Logistic Regression | 57.99% |
| Random Forest Classifier | 61.83% |
| KNN(Figure 6) | 62.5% (k=6) |
| SVM(linear) | 58.5% |

## 4 Feature Engineering

1. **Logistic Regression**
   Following Feature Engineering steps have been performed to increase the accuracy of Logistic Regression models

   (a) **Correlated Features** First, I've trained the model which is obtained after removing one of the correlated pairs. The accuracy obtained after performing this step is **56%**.

   (b) **KBest Features-** Next I've used the **sklearn.feature_selection.SelectKBest** library to search for best k features. These features chosen based on highest scores obtained for correlation with the output variable

   (c) **Principal Component Analysis-** Since the number of features are relatively large, we can do PCA to reduce the number of features. I have iterated through number of components obtained through PCA and trained a Logistic Regression Model using the principal components(Figure 4).

   (d) **Recursive Feature Elimination-** I've also tried a wrapper method for feature elimination. Running **Recursive Feature Elimination** for selecting the best 20 features. This is done using the library **sklearn.feature_selection.RFE**. Since, this process this computationally expensive, I've tried just for 20 features.

2. **Random Forest Classifier**
   Following Feature Engineering steps have been performed to increase the accuracy of Random Forest Classifier

(a) **Feature Importance: feature_importances_** method in **sklearn.ensemble.RandomForestClassifier** is used to find the n most important features. feature_importances_ returns the most important features which are used to classify the dataset using decision trees.

(b) The features obtained using **feature_importances_** have correlation between them(Figure 5). The ones which have high correlation (more than 0.7) are found and one of them is removed. The model is trained on the remaining features and the accuracy is obtained.

3. **KNN**

   (a) **Feature Importance:** The most important features obtained after running Random Forst Classifier are selected and then KNN is applied on it.

   (b) **KBest Features:** The 20 best features obtained using **sklearn.feature_selection.SelectKBest** are used and KNN is applied on it.

   (c) **Principal Components:** KNN is applied on the variable obtained using PCA. The number of components is varied from 1 to 50 to make the computationally feasible

4. **Support Vector Machines:**

   (a) **Feature Importance:** The most important features obtained after running Random Forst Classifier are selected and then SVM model is trained using those features.

# 5 Model Building

1. **Logistic Regression**
   Following Feature Engineering steps have been performed to increase the accuracy of Logistic Regression models

   (a) **Correlated Features** Since the number of features is still very high. I've trained the Logistic Regression model using **L2 Normalization** to avoid overfitting. The accuracy obtained after performing this step is **56%**.

   (b) **KBest Features**- Next I've used the **sklearn.feature_selection.SelectKBest** library to search for best k=[1-25] features then I've trained a Logistic Regression Model using **L2 Normalization** on the k best features obtained. The accuracy for each of these models have been calculated. The maximum accuracy of **62.83%**is obtained for **k=4**. (Refer Figure 3)

   (c) **Principal Component Analysis**- I've iterated through the number of principal components chosen from 1-500, trained a logistic regression model using L@ Regularization. The maximum accuracy of **59.83%** is obtained for **number of components = 1** even though then variance explained is only 0.014. (Refer Figure 4). Since the maximum accuracy is obtained for 1 principal component, I've avoided L2 in final model which is there in Jupyter notebook.

   (d) **Recursive Feature Elimination**- The accuracy obtained on test set after training a logistic regression model on 20 best features using RFE is **58.16%**

2. **Random Forest Classifier**
   Following Feature Engineering steps have been performed to increase the accuracy of Random Forest Classifier

   (a) **Feature Importance:** On iterating through different number of importatnt features obtained using Random Forest. Maximum accuracy was obtained when we took top 18 important features. Accuracy of **89.83%** was obtained. **n_estimators**, which is the number of trees build using RF is taken to be as 1000.

   (b) After removing one of the **highly correlated** features, we have trained a random forest classifier on the remaining features.

Parameters given: **n_estimators**=1000
Accuracy obtained: 87.5%

(c) **Grid Search Cross Validation:** Using **sklearn.model_selection.GridSearchCV** we can iterate through different combination of parameters specified for **sklearn.ensemble.RandomForestClassifie** Steps are shown in the Jupyter Notebook for iterating through one feature at a time, one iteration is done at a time as it is computationally expensive.
Parameters:

- max_depth: [5,10,15,25,50,100], best parameter: max_depth=10
- max_features: [1,3,6,7,9,10,15], best parameter: max_features=9
- n_estimators: [100:1100], best parameter: n_estimator=1000

Accuracy obtained: 88.33%
The accuracy is lower than the accuracy obtained when choosing original 18 important features. This can be because the best combination of parameters can be choosen by iterating over one parameter at a time.

3. **KNN**

(a) **Feature Importance:** On iterating through the n=[1:20] most important features from Random Forest Classifier and K=[1:25] max accuracy of **91.66%** is obtained
Best Parameters: **imp features**=13, **k=6**
Accuracy obtained: 91.66%

(b) **KBest Features:** On iterating through K=[1:25] for kbest features obtained from **SelectKBest** max accuracy of **79.5%** is obtained
Best Parameters: **K Best Features**=20, **k=15**
Accuracy obtained: 79.5%

(c) **Principal Components:**On iterating through the number of principal components=[1:50] from Principal Component Analysis and K=[1:number of components] maximum accuracy of **91.66%** is obtained
Best Parameters: **number of principal components**=13, **k=12**
Accuracy obtained: 60.66%

4. **Support Vector Machines:**

(a) **Types of SVM**:

- Polynomial SVM: Max Accuracy of 58.66% is obtained for a 2 degree polynomial SVM
- Gaussian SVM: Max Accuracy of 58.66%
- Sigmoid SVM: Max Accuracy of 58.33%

(b) **Feature Importance:** The 20 most important features obtained after running Random Forest Classifier are selected and then a Gaussian SVM model(since it gave the best accuracy) is trained using those features.
Accuracy obtained: 82.66%

## 6 Discussion

The best accuracy is always obtained on models after using the most important features obtained from Random Forest Classifier. Maximum accuracy is obtained when we use KNN on the best features obtained from Random Forest which is **91.66%**. Overall, Logistic Regression does not perform well on this data. Reducing number of features using PCA also does not improve accuracy for Logistic Regression, KNN. Reducing number of correlated features does not improve accuracy for Logistic Regression.

| Model | Best Accuracy |
|---|---|
| Logistic Regression | 62.833% |
| Random Forest Classifier | 89.83% |
| KNN | 91.66% |
| SVM(gaussian) | 82.5% |

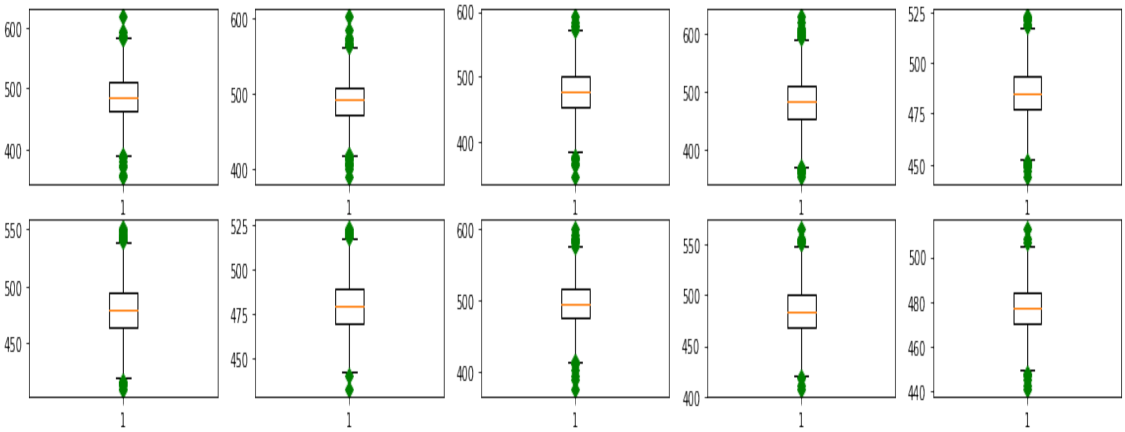

Figure 1: Boxplots of features 10-20

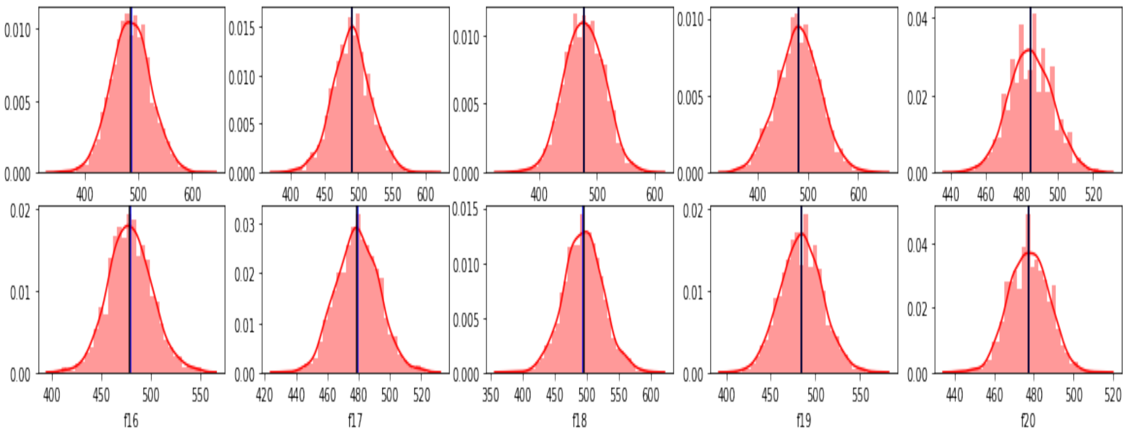

Figure 2: Density plot of features 10-20

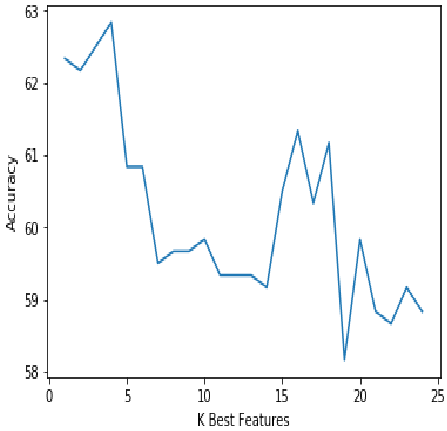

Figure 3: Accuracy of LR model trained on k best features

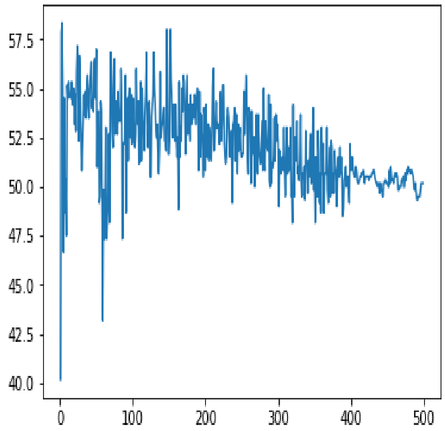

Figure 4: Accuracy of LR model trained by number of principal components

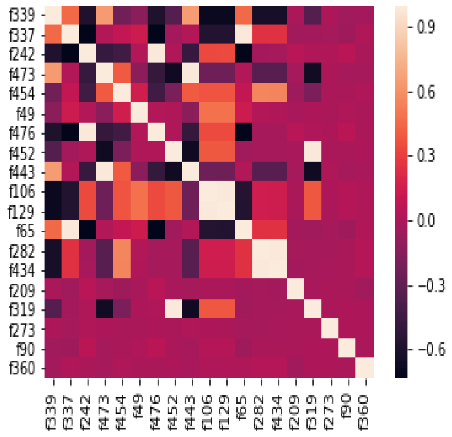

Figure 5: Correlation between important features obtained from RF

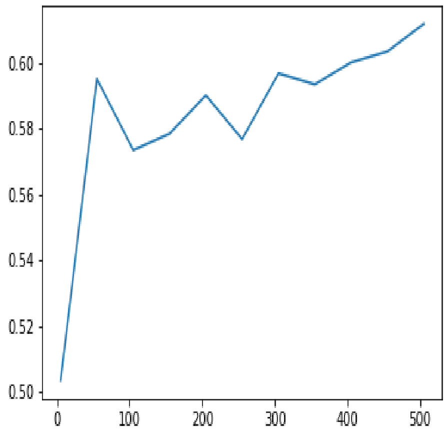

Figure 6: Baseline KNN model: k vs accuracy