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

Prashanth Thirukkurungudi Sekar (prthiru)

November 5, 2018

1 Introduction

In this assignment we are given the following four data files:

1. a3-train.data
2. a3-train.labels
3. a3-test.data
4. a3-test.labels

There are 2,000 training rows and 600 test rows, each row has 500 features, and there are only two labels, -1 and 1.

2 Exploratory Data Analysis

1. **Sneek Peak at the datasets**
The first part of the EDA phase is to make sure that the data we have read has the correct dimensions and in the right format

2. **Summary statistics of the data**
Next, we check the summary statistics of the data to check if any feature is clearly an outlier with respect to the 5 summary statistics such as mean, median, standard deviation, min, max and quantile values
Inference: There doesn't seem to much variation among the features except for the standard deviation

3. **Verify the data types of features**
This step is to verify the data type of the features in order to give us an idea of how to proceed with data cleaning and feature engineering
Inference: There are only numerical features in both train and test data

4. **Checking for Missing values**
Next, we need to check for missing values in the data in order to evaluate methods to treat the missing values before proceeding with baseline models
Inference: Both train and test do not have any missing values

5. **Checking if there are single level features in the data sets**
If there are features with just 1 unique value then it's better to remove those features since they won't add any importance to the model
Inference: Both train and test data do not have any columns with just 1 value being duplicated across all records

6. **Checking the number of unique values in each feature to identify any possible categorical feature**
If we can identify any categorical feature, then during the feature engineering step, we can either encode these features using dummy coding or try Weight of Evidence techniques
Inference: There are few features such as '90','154','168','173' etc., with less unique values and we will explore these in the feature engineering step

7. **Checking for duplicate features in the data set**
It's always important to discard any duplicate columns in the data since there is no use having the same column twice
Inferece: There are no two columns which are completely identical to each other

8. **Checking for Unique Identifier features in the data set**
Identifying the unique identifier feature will help in feature engineering and also help us weed out the feature during the model building phase
Inferece: Since the maximum number of unique values in a column in 552, we can confirm that there are no unique identifier columns in the data

9. **Checking for skewness in the data**
This step allows us in identifying the features that are heavily skewed and transforming those features will help in capturing the variance better
Inference: Even after trying with a hard cutoff for checking the skewness in features, the features above the cut-off still seem to be normally distributed

10. **Checking for balance of the target variable**
Checking the balance of the target variable helps in deciding if we need to perform any sampling technique to balance the data if the target variable is imbalanced
Inference: The target variable is perfectly balanced

3 Baseline Models

I have picked Logistic Regression, K Nearest Neighbours Classifier, Random Forest Classifier and Gradient Boosting Classifier as the models for this analysis and AUC and Accuracy as the metrics to compare the performance of the baseline models.
With respect to both the evaluation metrics, Gradient Boosting performed the best followed by K Nearest, Random Forest and finally Logistic Regression.

Model	AUC Score	Accuracy
Logistic Regression	0.602	0.59
KNN Classifier	0.759	0.692
Random Forest Classifier	0.675	0.628
Gradient Boosting Classifier	0.834	0.744

Note: All these models were executed with default parameters and given the high sophisticated methods used in the building/working of Gradient Boosting, it isn't surprising to see that it performs the best among the chosen models.

4 Feature Engineering

1. **First, let's try removing highly correlated variables and execute our baseline models to check if the AUC/Accuracy score improves**
i. We use the Variance Inflation Factor method to remove highly correlated features. This technique recursively runs linear regression by considering each independent feature as the dependent feature during every execution and calculates how much of it can be explained by the other independent features and if they can explain a significant amount of variance (90-95%) then we can remove the variable of interest
ii. Using the above explained technique, highly correlated features were removed and the baseline model was executed for the remaining features alone

Model	AUC Score	Accuracy
Logistic Regression	0.6	0.574
KNN Classifier	0.769	0.694
Random Forest Classifier	0.685	0.628
Gradient Boosting Classifier	0.826	0.783

Inference: It might be a bit surprising that Logistic regression dint improve in terms of AUC or Accuracy after removing correlated features but other models slightly improved (Not by a significant amount though). Sometimes correlation does not lead to causality and maybe it's better to have these variables in the model and some more interesting feature engineering is required to improve the accuracies

2. **Next, Replacing the 'May-be' categorical features identified in step 6 of EDA with their information value**
i. Features with less than or equal to 10 levels were considered as categorical features
ii. For each level in the feature, weight of evidence (WOE) is calculated where WOE is the log ratios of the total number of 1's in the level to the total number of -1's in the level. Then information value for each level in the feature is computed which is WOE * (Distribution of Good - Distribution of Bad)

Note: Distribution of Good = Good of all goods (Distribution of 1's in this level when compared to the total number of 1's in the whole feature)
Distribution of Bad = Bad of all bads (Distribution of -1's in this level when compared to the total number of -1's in the whole feature)

iii. After replacing the levels in the features with their respective information value, the data was normalized and the baseline models were executed

Model	AUC Score	Accuracy
Logistic Regression	0.563	0.532
KNN Classifier	0.515	0.522
Random Forest Classifier	0.433	0.445
Gradient Boosting Classifier	0.563	0.547

Inference: The baseline models aren't performing better after replacing certain "might-be" categorical features with the information value of their respective levels. In fact, they are performed badly.

3. **Subsetting important features based on feature importance**
i. Based on the feature importance obtained from the baseline models (Random Forest and Gradient Boosting), both the importances were combined and sorted and the top 10 features were selected
ii. The baseline models were executed for the above obtained subset of features

Model	AUC Score	Accuracy
Logistic Regression	0.628	0.594
KNN Classifier	0.961	0.9
Random Forest Classifier	0.94	0.868
Gradient Boosting Classifier	0.887	0.818

Inference: Subsetting important features improved the baseline model performances significantly

4. **Creating new features by combining features from the feature importance plots obtained through the above method**
i. After plotting the feature importance plots for the Random Forest and Gradient Boosting classifiers, 6 new features were created by combining the features together
ii. The newly created features are : adding features 105 and 153, 105 and 378, 378 and 153, 442 and 153 ; multiplying features 378 and 153, 153 and 442

Model	AUC Score	Accuracy
Logistic Regression	0.64	0.587
KNN Classifier	0.96	0.9
Random Forest Classifier	0.94	0.867
Gradient Boosting Classifier	0.912	0.84

Inference: Creating new features improved the baseline model performances significantly especially for Logistic Regression and Gradient Boosting and the created features had high feature importance too.

5 Model Building

Logistic Regression, K Nearest Neighbours, Random Forest and Gradient Boosting classifiers are chosen for parameter tuning.

1. **Tuning the parameters of Logistic Regression Model**
i. In order to perform L2-Regularized regression, the lambda value is tuned using 5-fold Cross Validation
ii. The optimal lambda value obtained is :1000

2. **Tuning the parameters of K Nearest Neighbours Classifier**
i. The optimal K value is tuned using GridSearch through 3-fold Cross Validation
ii. The optimal K value obtained is : 6

3. **Tuning the parameters of Random Forest Classifier**
i. The optimal values for parameters such as number of estimators, depth of the tree, critireon for node split and critireon for selecting the maximum number of features for each tree is tuned using GridSearch through 3-fold Cross Validation

ii. The optimal parameter values are :

Parameter	Value
n_estimators	600
max_depth	20
criterion	gini
Max_features	auto

4. **Tuning the parameters of Gradient Boosting Classifier**
i. The optimal values for parameters such as learning rate, depth of the tree and number of estimators for each tree is tuned using GridSearch through 5-fold Cross Validation
ii. The optimal parameter values are :

Parameter	Value
n_estimators	500
max_depth	8
learning_Rate	0.1

Performance of the models after hyperparamete tuning were :

Model	AUC Score	Accuracy
Logistic Regression	0.627	0.594
KNN Classifier	0.964	0.908
Random Forest Classifier	0.959	0.89
Gradient Boosting Classifier	0.954	0.882

Inference: The model performances improved significantly for KNN, Random Forest and GBM after parameter tuning.

Note: Since Logistic Regression model after parameter tuning gave the least score in the train data set, it was not considered for ensemble and stacking methods.

5. **Ensemble of Model Results**
i. Each of the other 3 models were trained on the train data set and an ensemble or weighted average of the results were taken to capture the variance explained by individual models
ii. The AUC score after ensemble of the models was : 0.968

6. **Ensemble of Model Results using Validation dataset and stacking**
i. Each of the 3 models were trained on a training data set (90% of the actual train data) and validated on the validation data set (10% of the actual training data) and eventually tested on the test data
ii. Using the results obtained on the validation data set using the 3 models, a logsitic regression was trained using only these 3 results as the independent variable and tested using the predictions obtained by the models on the test data (Basic Stacking)
iii. Eventually, the above obtained results and the results obtained through step 5 were combined through a weighted average method to obtain the final prediction

Final AUC Score obtained is : 0.970

Final Accuracy score obtained with a threshold of is : 0.91

Final Confusion Matrix is :

Actuals/Predicted	-1	1
1	287	13
1	40	260

6 Discussion

1. **Performance of the models**
The models improved significantly after tuning their corresponding hyperparameters using cross validation and GridSearch since the parameters or the hyperparameters were fine tuned to fit this particular dataset and like we expected they did a pretty good job in comparision to the default parameters which we had utilized in the baseline models.
KNN performed slightly better than the other models after parameter tuning.

2. **Challenges and improvement of model performance**
i. Since the features were masked, it was difficult to identify and create new features without knowing the meaning of the features or the business context
ii. Choosing a particular evaluation metric was also difficult without knowing the business requirement since metrics are selected according to the business question. For example, we could use metrics such as Recall for targeted e-mail marketing campaign analysis.
iii. If the business requirement is provided, the methods mentioned in the above points coupled with more computing power to tune all the hyperparameters using Gridsearch can produce more accurate results.