

# Identifying customers for selling personal loans

**IS 577 AO FA2020: Data Mining**  
**Pruthvi Patel (prp4)**  
**MSIM**

# Appendix

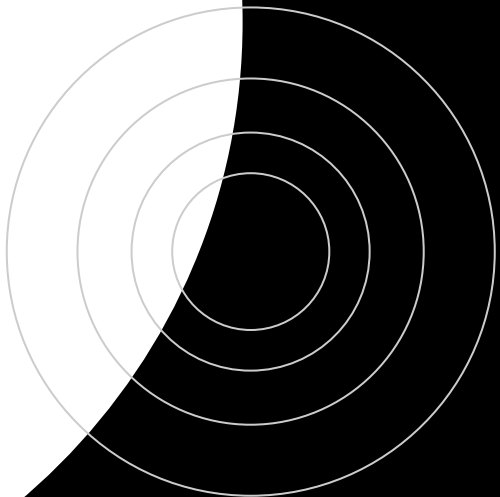
1. **Dataset Overview**
2. **Data pre-processing**
3. **Model building and experiments**
4. **Comparison of various experiments**
5. **Hyperparameter tuning**



1

# Dataset Overview

Let's get to know the data columns



# Dataset and its Columns

- The dataset used is of a bank's customers from the previous campaign which includes 14 different parameters of about 5000 customers. Each row represents a customer, each column contains the customer's attributes described in the column Metadata.
- Dataset source: <https://www.kaggle.com/sriharipramod/bank-loan-classification>
- Target class distribution is highly imbalanced in the ratio of 9:1. Out of 5000, only 500 customers have taken personal loan.

**ID** : Consists of ID's of all entries

**ZIP Code** : Consists of ZIP Code of each user

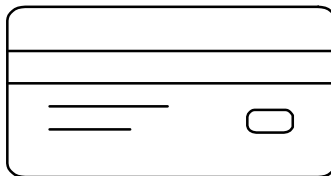
**Age** : Consists of Age of all entries

**Online** : If the customer uses internet banking

**Family** : Consists of number of family members

**CCAvg** : Credit card spending average per month

**Experience** : Consists of Experience in years of all entries



**Education** : education level (1- grad, 2- UG 3- PDH)

**Credit Card** : Binary entry, 1 indicates customer uses credit card

**CD Account**: If customer has a certificate of deposit account

**Security Acc** : Binary entry, 1 indicates user opted for Sec Acc

**Income** : Indicates the income of each entry

**Personal Loan** : target column, 1 indicates user opted for loan

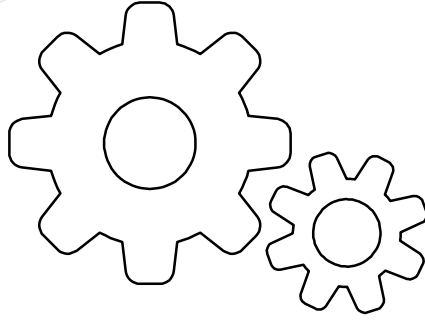
**Mortgage** : Binary entry, 1 indicates user opted for Mortgage

Columns Removed

Columns Used

Target Column

# 2



# Data Pre-processing

Here, we explain about data pre-processing steps

## Experience column

- **Issue**  
52 entries had negative values as experience which cannot be true
- **Solution:**  
We replaced the negative experience value with the mean of positive experience of all other users with same age  
  
For example, for a certain entry, the experience is -2 and age is 25 years. Here we replaced -2 with the mean of positive experience of all customers having age of 25 years

## Removing Columns

- We dropped columns "ID" and "ZIP Code" since they didn't contribute in prediction of the likelihood of customer opting for personal loan
- We used experience column rather than age column for our modelling as both experience and age are high correlated (0.99) based on the correlation table
- **Reason:**  
It is possible, that someone started their career later, hence experience would be a better parameter

**After the mentioned pre-processing step, we are left with 10 columns, and all the remaining columns will be used for modelling**

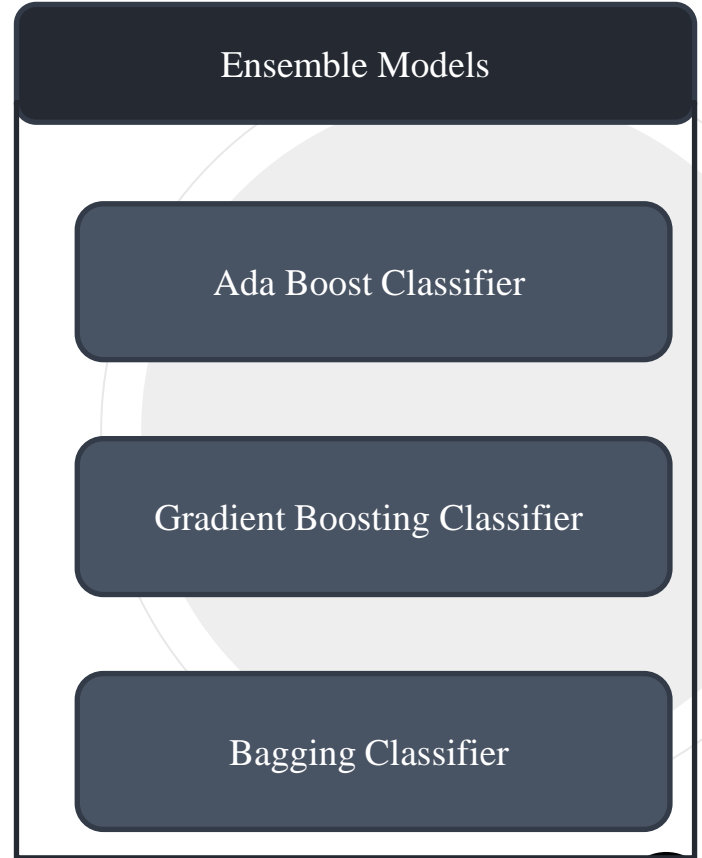
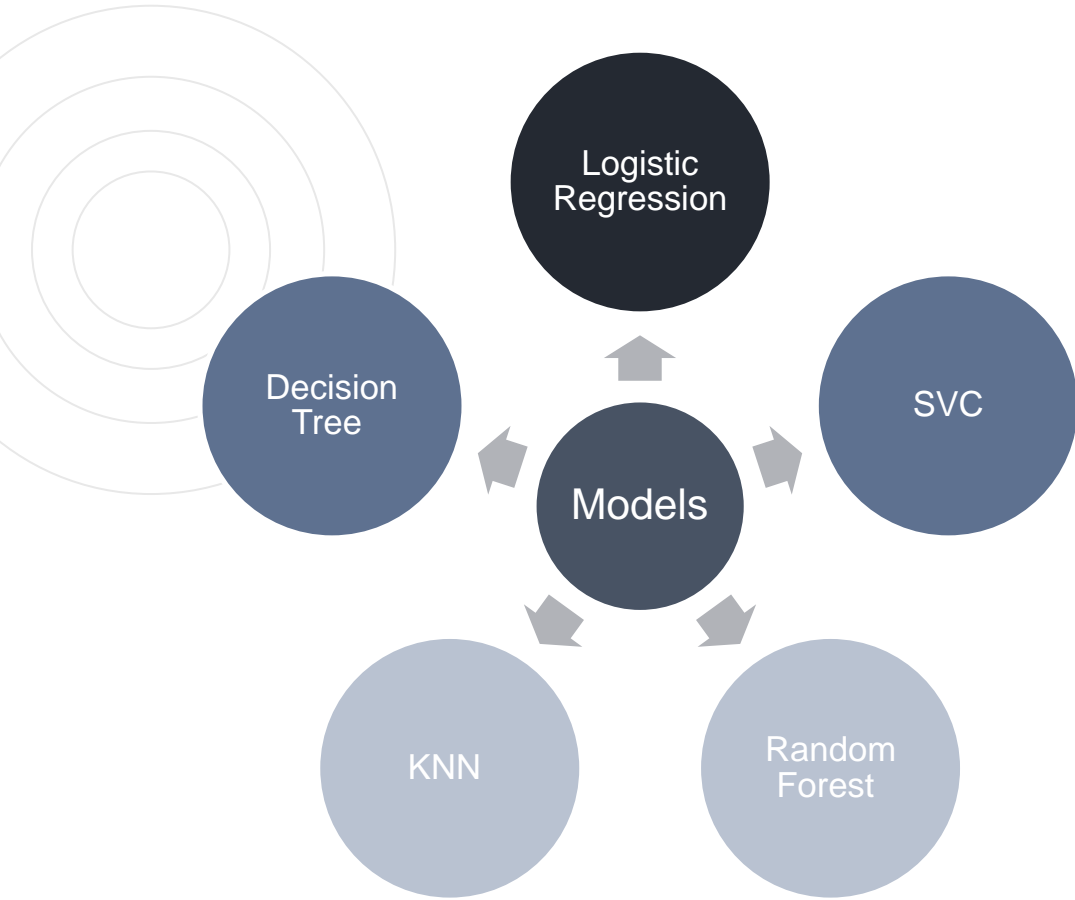


# 3

## **Model building and experiments**

Here we discuss about different models we  
built and different experiments we did to  
improve model performance





# The flow of project and modeling was undertaken in the following manner



## 10-fold stratified cross validation

As the target class data is highly imbalanced, we used stratified CV so that each fold represents both the classes equally



## One-hot Encoding

Categorical data is not directly usable as input variables and output variables to be numeric, so we used one-hot encoding



## Standardization

As we had highly varying magnitudes of values, we standardized the features of data with the distribution of 0 and mean value and variance equals to 1



## One-hot Encoding + Standardization

We combined second and third iteration where we built all the models again using both one-hot encoded and standardized data

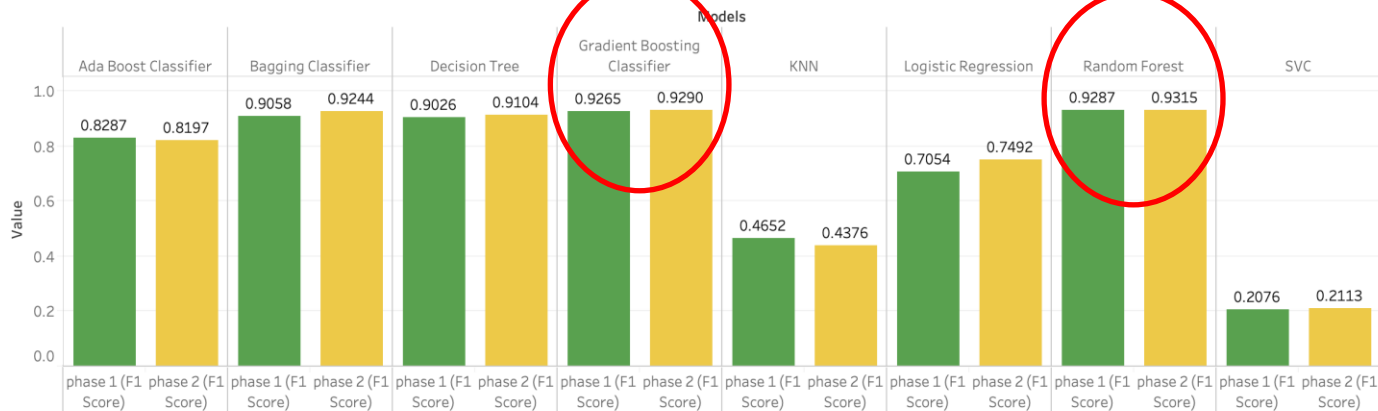


# 4

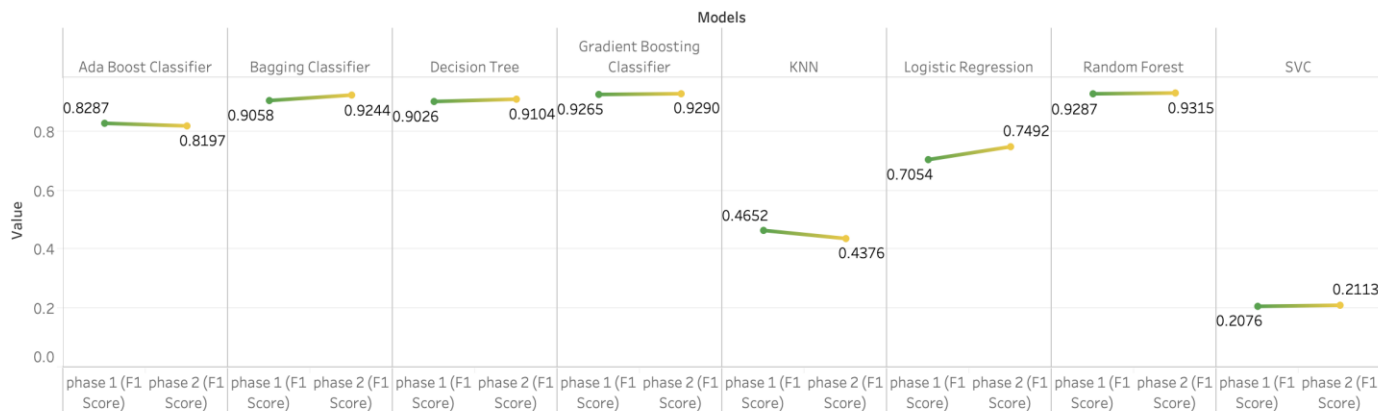
## **Comparison of various experiments with different model built**

Here we discuss about how different models are performing at different stages of experiments

Comparison of 10 Fold Cross Validation F1 Score with One Hot Encoding



Comparison of 10 Fold Cross Validation F1 Score with One Hot Encoding



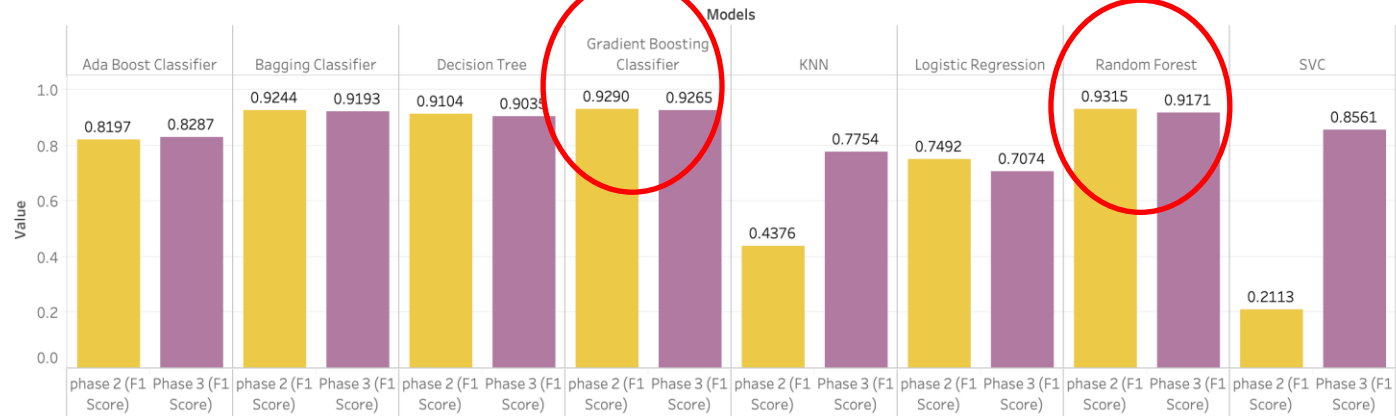
## Comparison of iteration 1 and 2:

- Random Forrest and Gradient Boosting Classifier seems to perform the best among all the models with F1 score near to 0.93
- This observation has been consistent across both phases
- One-hot encoding doesn't seem to help with increasing the F1 score of the top performing model.

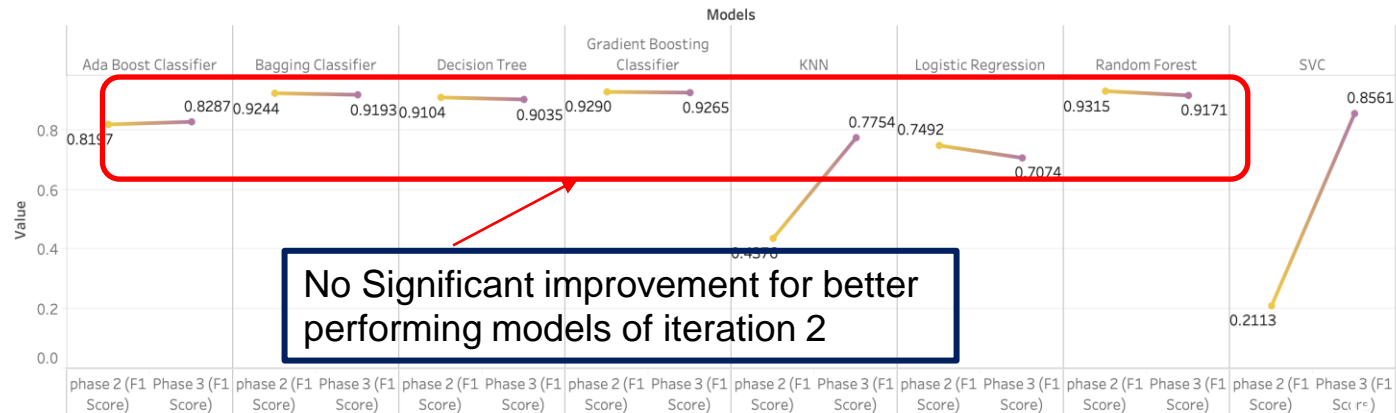
## Comparison of iteration 2 and 3:

- Random Forest and Gradient Boosting Classifier seems to perform the best among all the models with F1 score near to 0.92
- The F1 score of the top performing model seems to decrease by 0.003 as compared to the second iteration.
- Conclusion - standardizing the data doesn't seem to help as compared to one-hot encoding

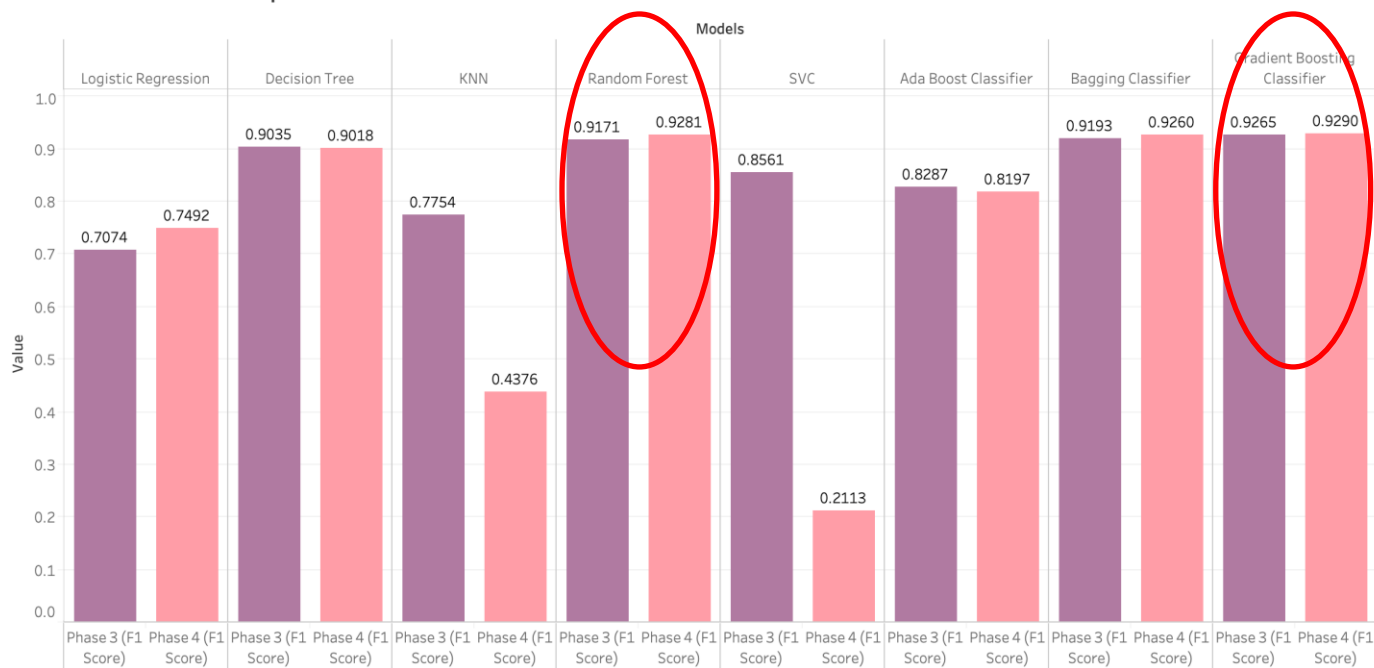
Comparison of One Hot Encoding F1 Score and standardization F1 Score



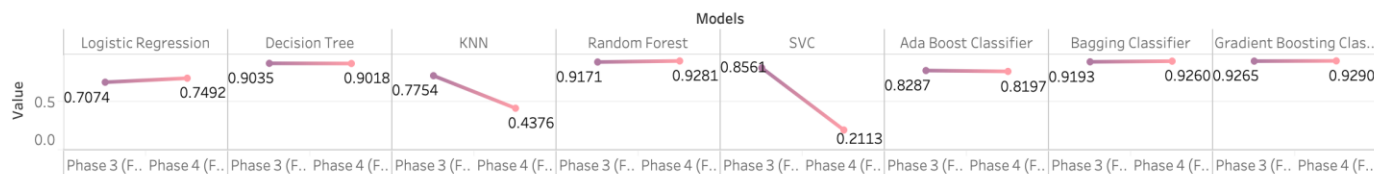
Comparison of One Hot Encoding F1 Score and standardization F1 Score



Comparison of Standardized data F1 Score with encoded and standardized F1 Score



Comparison of Standardized data F1 Score with encoded and standardized F1 Score



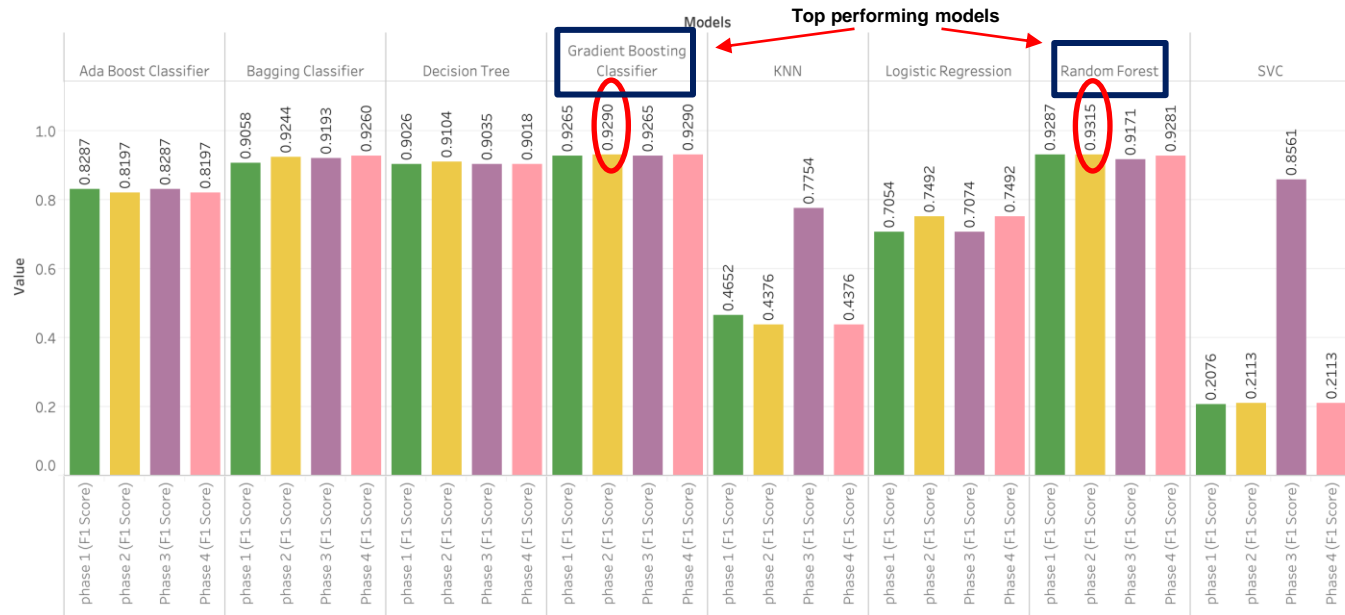
## Comparison of iteration 3 and 4

- Consistent observation throughout iterations is Random Forest and Gradient boosting have max F1 Score
- This observation has been consistent across both phases
- The F1 Score increased by 0.003 for the top performing model after implementing one-hot encoding on standardized data.

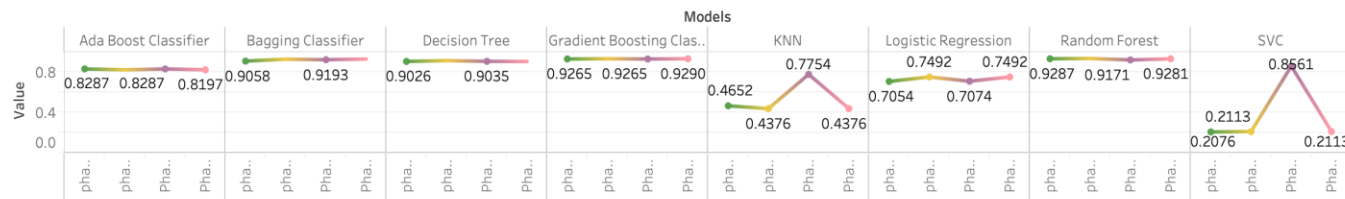
## Key Takeaway:

- Overall, Random Forest and Gradient Boosting classifier have the highest average F1 score with one-hot encoding, but it decreased when data is standardized.
- We can say that standardization doesn't contribute in improving the F1 score.
- We will further fine tune the hyper-parameters for Gradient Boosting and Random Forest

Comparison of all the phases F1 Score



Comparison of all the phases F1 Score





**5**

# **Hypermeter tuning of top performing models**



## Random Forest

- The best parameters for Random Forest are as follows  
{`'bootstrap': False,`  
`'max_depth': 9,`  
`'max_features': 9,`  
`'min_samples_leaf': 2,`  
`'min_samples_split': 3,`  
`'n_estimators': 30,`  
`'random_state': 5`}
- Average f1-score is 0.96

RF Accuracy 0.986	RF	0	1
	0	899	8
	1	6	87

10-fold stratified  
cross validation  
with one-hot  
encoded data

## Gradient Boosting

- The best parameters for Gradient Boosting are as follows  
{`'learning_rate': 0.3,`  
`'max_depth': 3,`  
`'max_features': 6,`  
`'min_samples_leaf': 4,`  
`'min_samples_split': 3,`  
`'random_state': 5,`  
`'subsample': 0.9`}
- Average f1-score is 0.96

GB Accuracy 0.986	GB	0	1
	0	902	5
	1	9	84

# Key take-aways from all the iterations



One-hot encoding **improves** the average f1-score for most of the models

Standardizing the data **doesn't** seem to help much. Either alone or with one-hot encoded data, it doesn't make much difference.

After hyperparameter tuning the top performing models, average f1-score of both Gradient Boosting and Random Forest model is **0.96**

We would select **Random Forest** as the top performing model based on the confusion matrix as random forest classifier classified 6 out of 93 as false negative as compared to 9 out of 93 by Gradient Boosting classifier

# Other methods to improve model performance

## Finding statistical significance of the columns

- It is possible that all the variables which we are using might have statistical significance for the target variable prediction but as we haven't verified it, we can't say rely on it
- Instead, we can verify and use only the variables with higher statistical significance by determining their p-values

## Normalizing the non-categorical columns

- As for feature scaling technique is concerned, we have only implemented Standardization on the non-categorical columns. But as we could see, it didn't seem to help in improving the model's f1-score. So, instead we can try to normalize the non-categorical data and see if it helps with model's f1-score improvement



**Thank you!**