Win Prediction Analytics

for IT Consulting Company Project Bids

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Outline

- Recap of Background, Motivation, Objectives
- Results
- Summary and Outlook

Recap

- Background
  - IT companies make bids for network services
- Motivation
  - More bids won -> more potential profit -> more success
    - How to increase chances of winning a proposal?
- Objectives

1. Grading system for false predictions
2. Determine prediction accuracy
3. Create the best win prediction model
4. Identify the key attributes
5. Identify the top 5 bid managers

Dataset

Overview of Dataset

8 Attributes (Independent Variables)

Outcome

	A	B	C	D	E	F	G	H	I
1	Client Category	Solution Type	Deal Date	Sector	Location	VP Name	Manager Name	Deal Cost	Deal Status Code
2	Telecom	Solution 7	27-Mar-12	Sector 24	L5	Ekta Zutshi	Gopa Trilochana	150000.00	Won
3	Telecom	Solution 7	25-Sep-12	Sector 24	L5	Ekta Zutshi	Gopa Trilochana	744705.88	Won
4	Internal	Solution 59	1-Aug-11	Sector 20	Others	Ekta Zutshi	Russell Dahlen	60000.00	Lost
5	Internal	Solution 59	28-Apr-11	Sector 20	Others	Ekta Zutshi	Russell Dahlen	60000.00	Lost
6	Internal	Solution 32	3-Jun-11	Sector 20	Others	Ekta Zutshi	Russell Dahlen	80882.35	Lost
7	Internal	Solution 32	24-May-11	Sector 20	Others	Ekta Zutshi	Russell Dahlen	80882.35	Lost
8	Internal	Solution 59	3-Nov-11	Sector 2	L10	Mervin Harwood	rahul sharma	526176.47	Won

- About the Dataset
  - Retrieved from undergraduate course (Prof. Ram Rohit Vannarath)
  - Free, large, no previous studies available

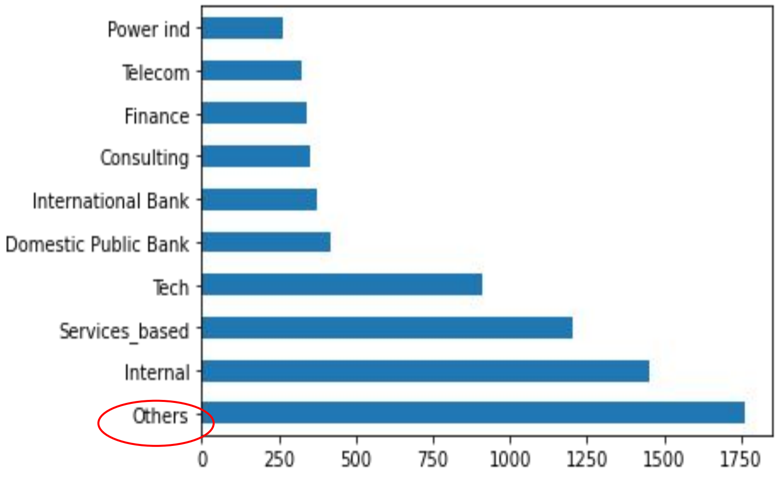
Dataset in Detail

- Data Preprocessing
  - 10,061 observations
  - Deal Status: 63% Lost, 37% Won
  - Deal Cost: range $0-37M, average $0.7M (exponential distribution)
  - Deal Date: range 2011-2019
  - Deal Location: range 1-13
  - Solution Type: range 1-67
  - VP Names: 43 unique names
  - Manager Names: 278 unique names
  - Sector: range 1-25
- Cleaning
  - 79 objects with missing data
  - Inconsistent naming of client category: “Energy” vs. “Energy ”

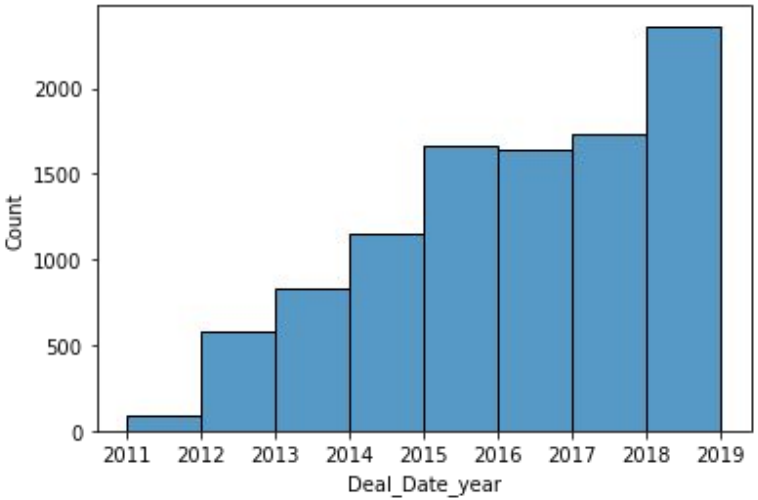
space

Dataset in Detail

Top 10 Client Categories (By Counts)

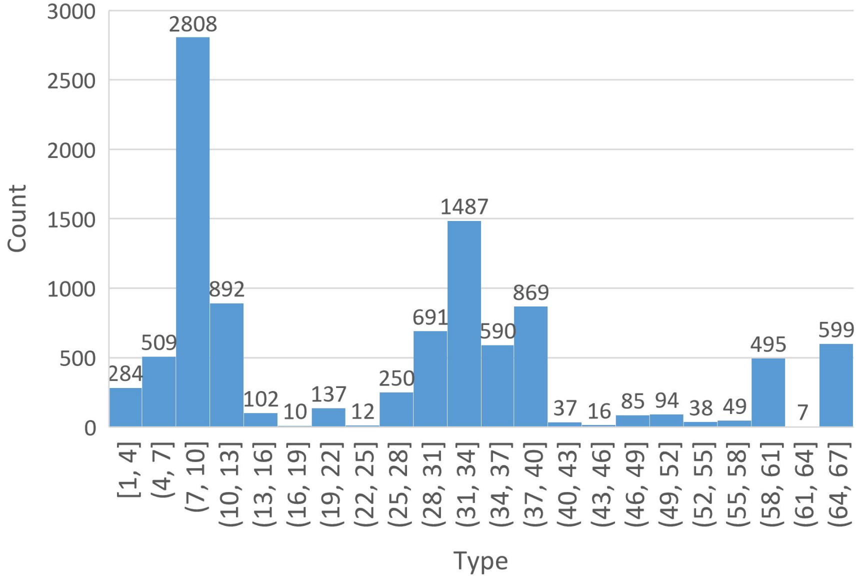


Deals Recorded per Year



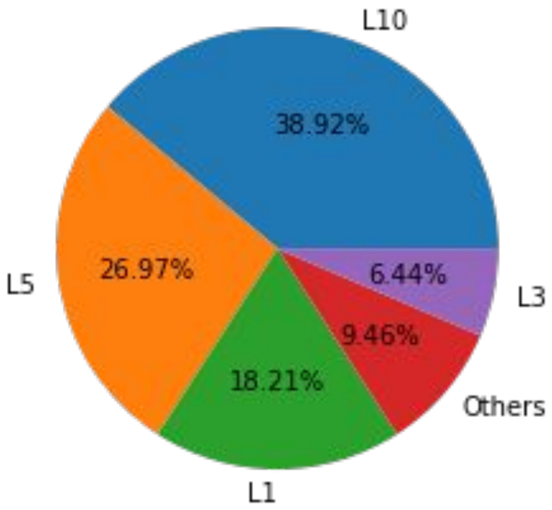
Dataset in Detail

Histogram of Solution Type Counts



(67 solution types)

Deal Location (% Counts)



Key Results

Objectives 1 & 2: Model Prediction Penalty & Accuracy

- Penalty Cost (PC) Matrix
  - False predictions <= 2 penalty levels
    1. FP -> wasted time (time is money)
    2. FN -> opportunity cost (potential profit)
  - Penalty cost determined for each model

    $$PC = 2*FN + 1*FP$$
- Model (Testing) Performance Measures
  - Accuracy

    $$A = (TP + TN)/ (TP + TN + FP + FN)$$
  - Specificity

    $\boldsymbol{S = TN / N}$, N <= *bid lost*
  - Normalized PC (NPC)
    - Min-max normalization [0,1]

Penalty Cost		Predicted Outcome	
		Lost	Won
Actual Outcome	Lost	0	1
	Won	2	0

Objective 3: Best Model

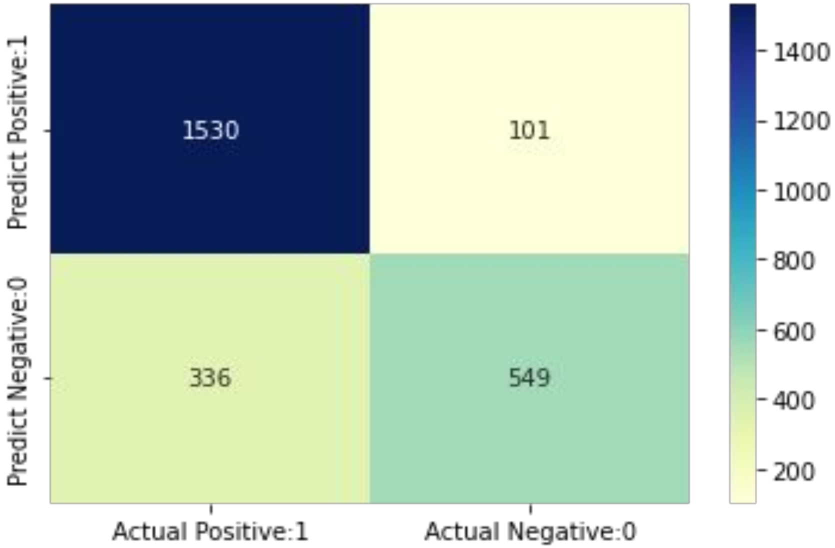
Rank	Model	Inputs	A	S	NPC	Avg*
1	Random Forest	Client category, Solution type, Deal date, Sector, Location, VP name, Manager name, Deal cost (All)	**0.83**	**0.86**	0.11	0.86
2	XGBoost	All	0.79	0.75	**0**	0.85
3	KNN	Deal date, Solution type (numeric), Deal cost	0.68	0.81	0.27	0.74
4	Logistic Regression	All; Deal date, Solution type and Deal cost as categorical	0.75	0.70	0.52	0.64
5	Naïve Bayes	All; Solution type and deal cost is numerical, all other categorical.	0.70	0.65	0.44	0.64
6	Decision Tree	All except Deal date; all as categorical	0.75	0.53	0.54	0.58

Avg = (⅓)[A + S + (1-NPC)]

RUTGERS
UNIVERSITY | NEW BRUNSWICK

In Detail: Random Forest

Confusion Matrix :

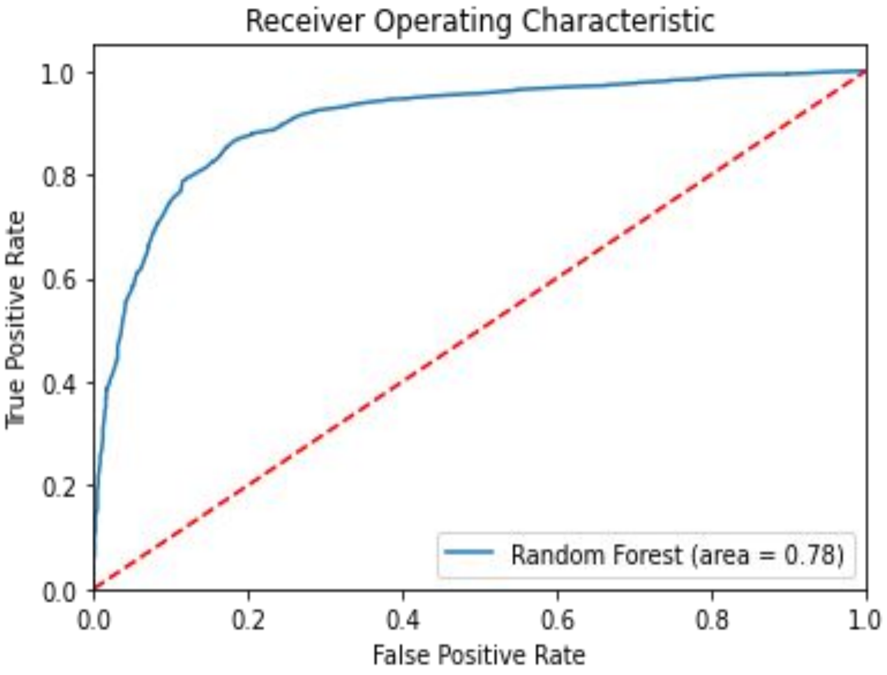


Testing Model Accuracy : 82.63%
Training Model Accuracy : 99.66%

Classification Report Analysis	Percentage Accuracy
Classification Accuracy	82.63%
Classification Error	17.37%
Precision	93.81%
Recall	81.99%
F-1 Score	88%

In Detail: Random Forest

AUC - ROC Curve



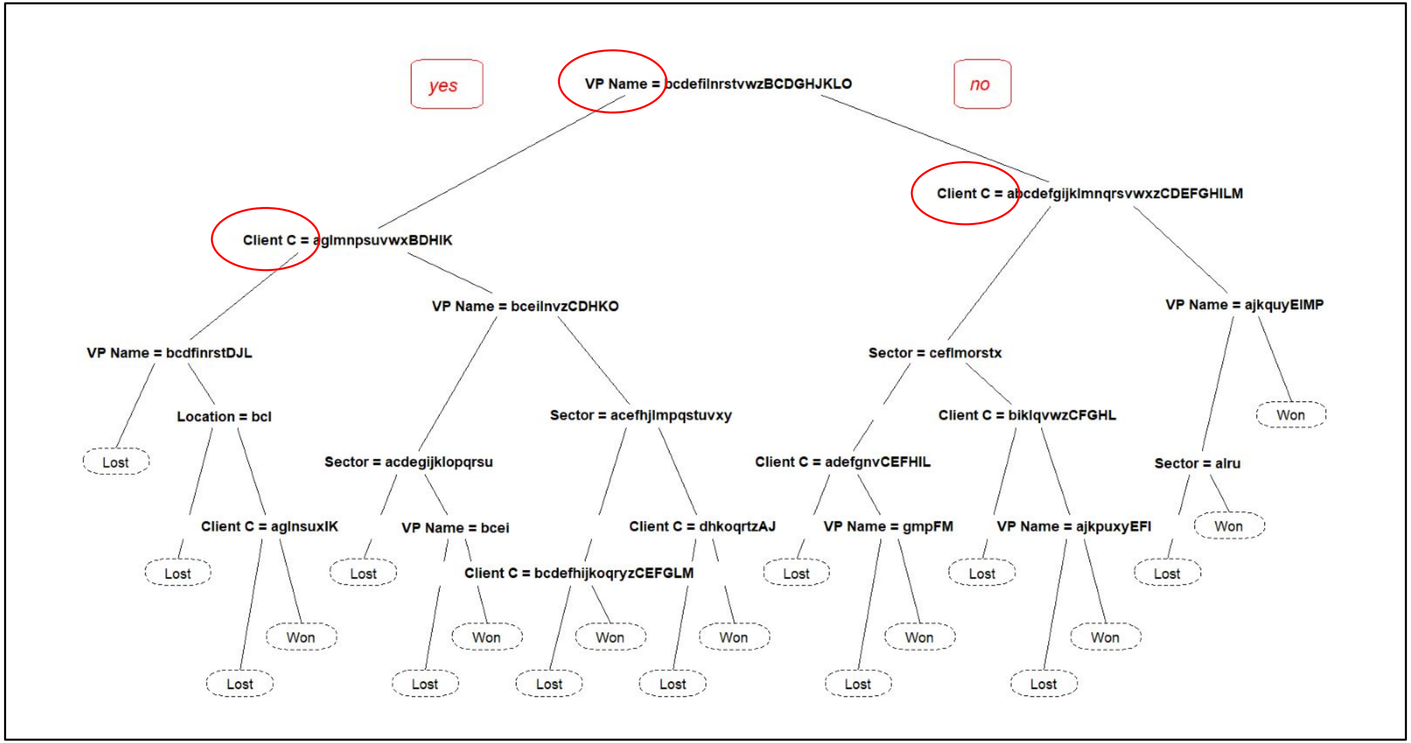
AUC Score : 78%

Objective 4: Key Attributes

- Key Attributes
  - Appear in topmost levels (1 & 2) of the Decision tree (highest information gain)
  - Tree node (top of tree) is automatically a key attribute
  - 2nd level nodes that appear in both trees are key attributes
- Decision tree #1
  - Solution type and Client Category as key attributes
- Decision tree #2
  - VP Name and Client Category as key attributes

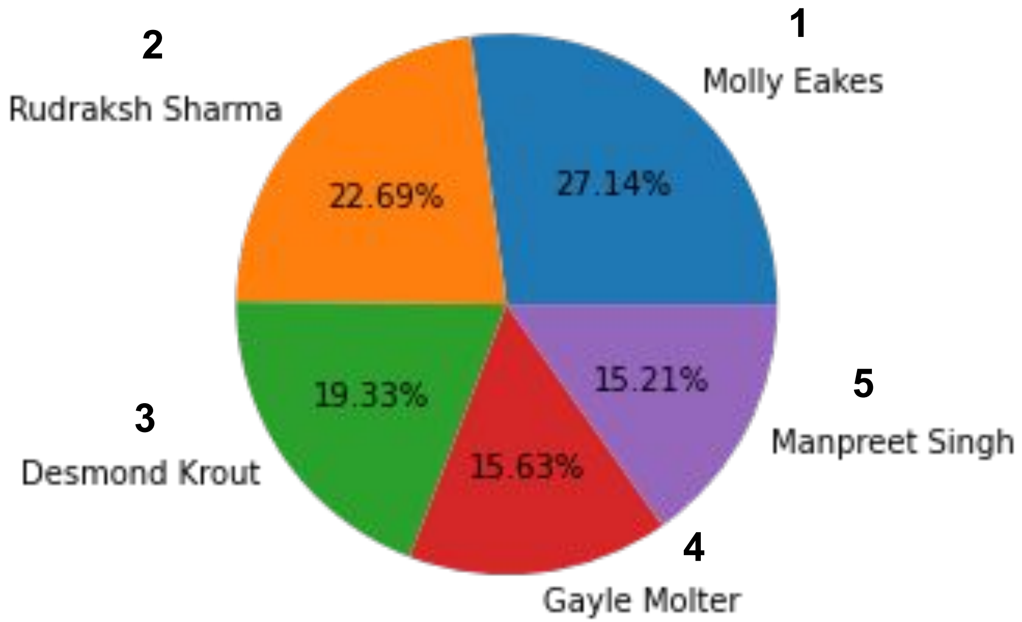
Decision Tree #1

Decision Tree #2



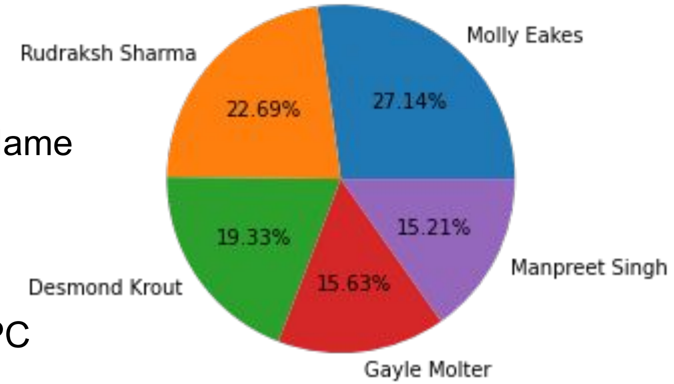
Objective 5: Top 5 Bid Managers

- Ranking
  - Based on bid winning percentage
  - Does not account for deal cost



Summary and Outlook

- Conclusions
  - Best model <= Random forest (83%)
  - Key attributes <= Solution type, VP Name, Manager Name
  - Top 5 bid managers (see pie chart)
  - False predictions <= 2 penalty levels
    1. FP -> wasted time (time is money)
    2. FN -> opportunity cost (potential profit)
  - Performance measured by accuracy, specificity, and PC
- Insight
  - Identification of key attributes
    - Evaluate management success
    - Reallocation of resources/efforts/investments
- Continued Study
  - Apply/extend same models to other bidding data
    - e.g., construction, grant proposals
  - Rank bid managers by % wins and deal cost
  - Correct overfitting for Random Forest model



Penalty Cost		Predicted Outcome	
		Lost	Won
Actual Outcome	Lost	0	1
	Won	2	0

Thank you! Questions?

Appendix

Model Performance Details (Avinash)

Model	Inputs	A	S	PC	TP	FP	TN	FN
Random Forest		82.63%	84.46%	773	1530	101	549	336
Decision Tree	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
XGBoost		79.49%	74.50%	652	1439	192	561	324
Logistic Regression		64.7%	33.33%	1770	1627	4	2	883
KNN		67.6%	54.23%	1257	1258	373	443	442
Naïve Bayes		64.94%	52.54%	1736	1603	28	31	854

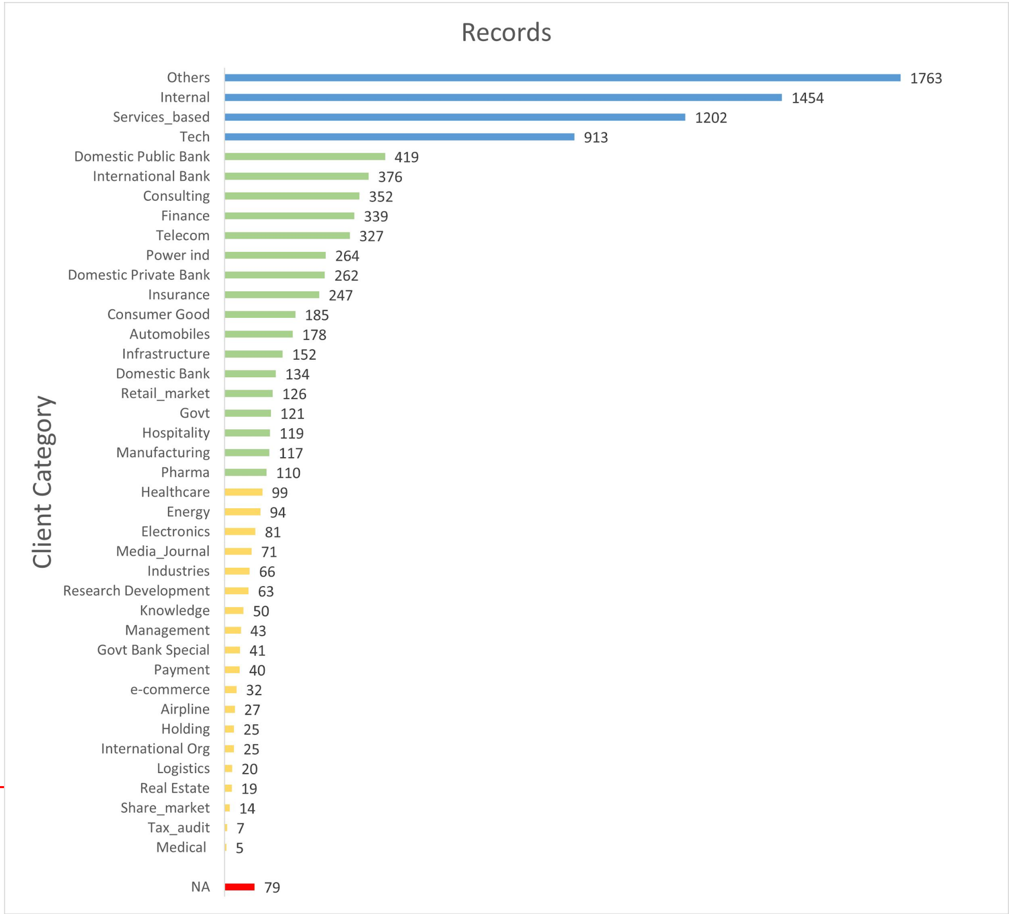
Model Performance Details (Peter)

Model	Inputs	A	S	PC	TP	FP	TN	FN
Random Forest	NA							
Decision Tree	Client Category, Sector, Location, VP Name	0.68	0.71	1148	363	238	1141	455
XG Boost	NA							
Logistic Regression (threshold = 0.42)	Deal Date, Solution Type (numeric), Deal Cost	0.60	0.62	1641	49	99	1278	771
KNN (k=5)	Deal Date, Solution Type (numeric), Deal Cost	0.68	0.81	959	373	447	1121	256
Naïve Bayes	NA							

Model Performance Details (Unmesh)

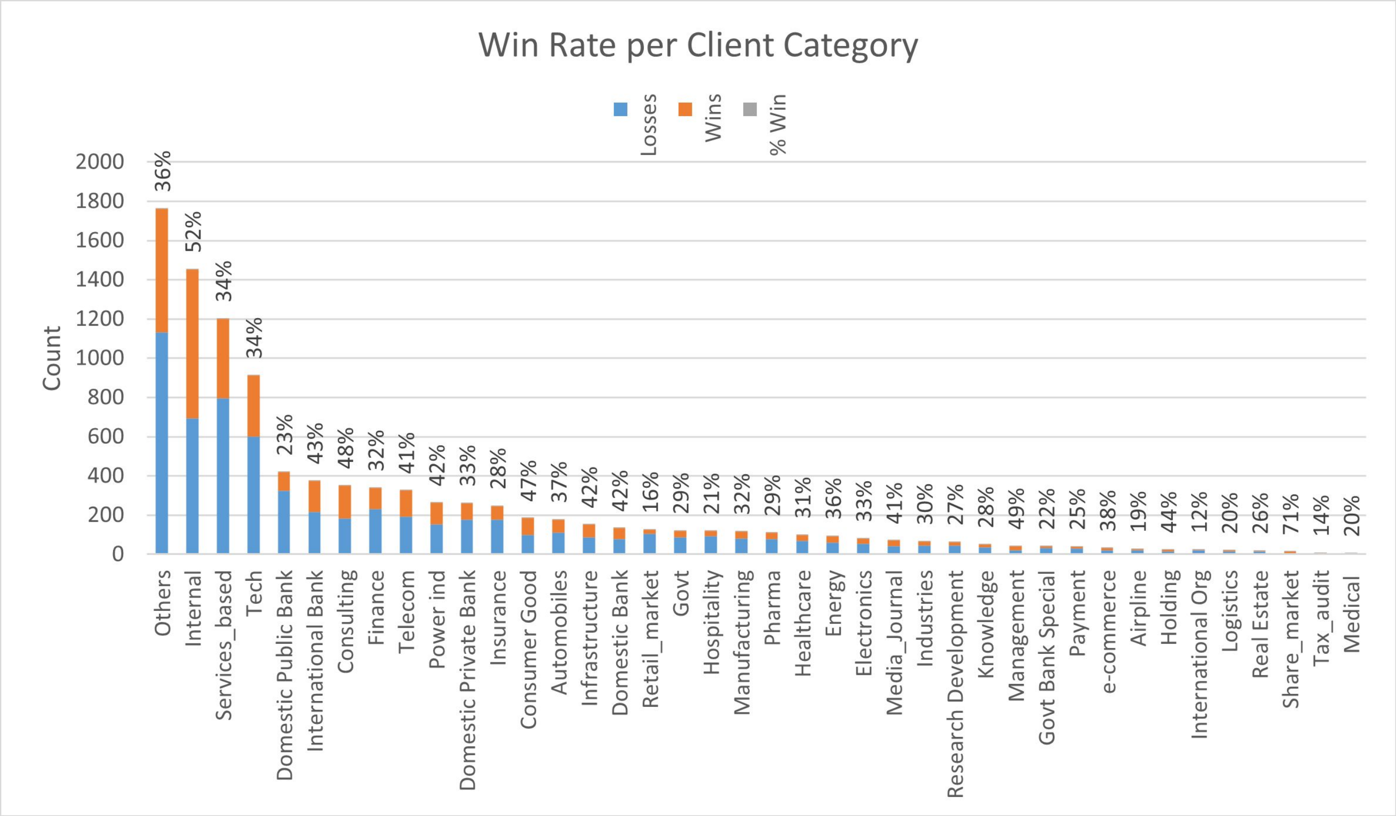
Model	Inputs	A	S	PC	TP	FP	TN	FN
Random Forest	Deal cost,Deal status code	74%	51%	1266	1573	541	569	184
Decision Tree	All attributes except deal date, all independent attributes are categorical	75.39	53%	1252	1658	518	590	216
XGBoost	NA							
Logistic Regression	All attributes except deal date,solution type and dealcost are numerical attributes.	75%	70%	1231	1611	263	624	484
KNN	NA							
Naïve Bayes	All attributes, solution type and deal cost is numerical, all other categorical.	70%	64.8%	1147	1623	251	463	645

Initial Data Analysis



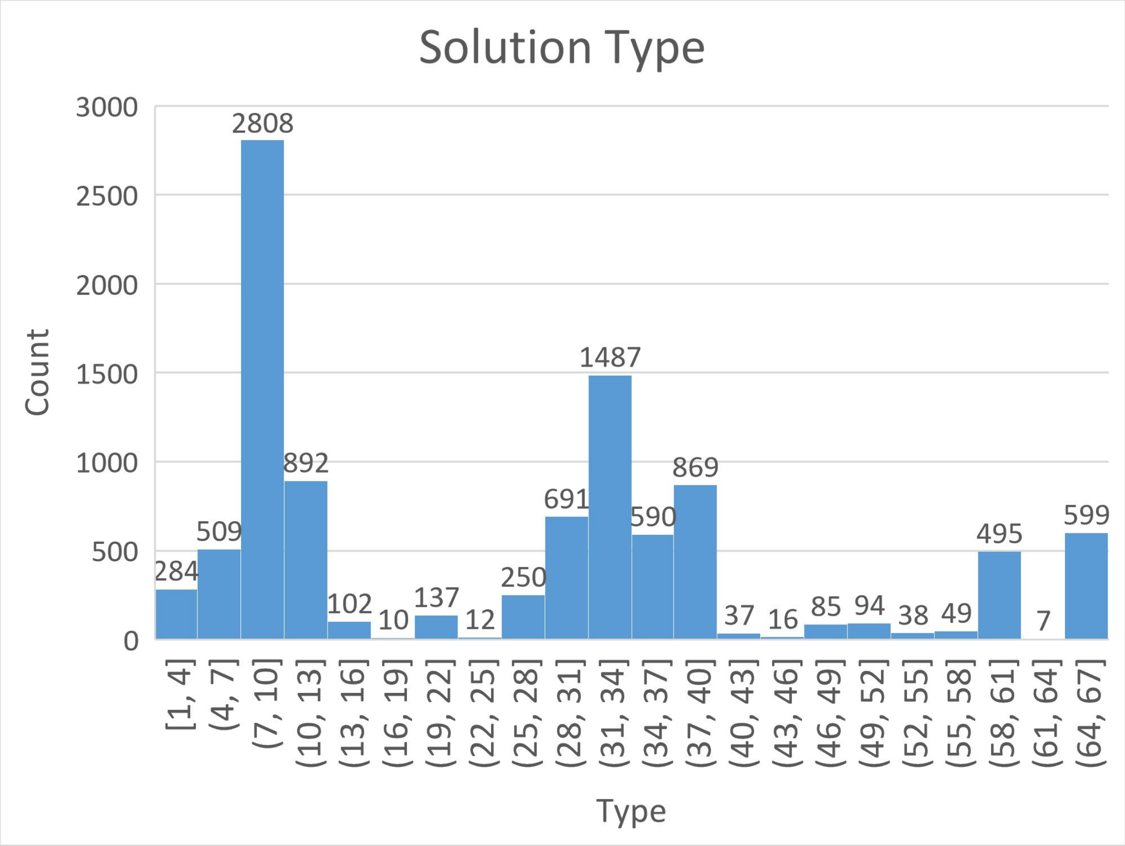
RUTGERS
UNIVERSITY | NEW BRUNSWICK

Initial Data Analysis



RUTGERS
UNIVERSITY | NEW BRUNSWICK

Initial Data Analysis



Initial Data Analysis

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10061 entries, 0 to 10060
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Client Category     9982 non-null   object
 1   Solution Type       10061 non-null  object
 2   Deal Date           10061 non-null  datetime64[ns]
 3   Sector              10061 non-null  object
 4   Location            10061 non-null  object
 5   VP Name             10061 non-null  object
 6   Manager Name        10061 non-null  object
 7   Deal Cost           10061 non-null  float64
 8   Deal Status Code    10061 non-null  object
dtypes: datetime64[ns](1), float64(1), object(7)
memory usage: 707.5+ KB
```

- Checking the type of data with the Dependent and Independent variables.
- Most of the variables are objects except for Deal Cost which is a character.

Initial Data Analysis

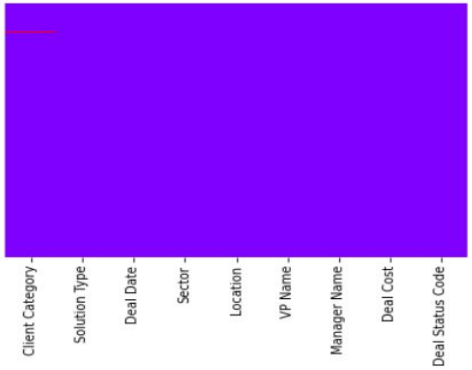
```
#Verifying the missing value
df.isnull().any()
#Client category is the only missing data
```

```
Client Category        True
Solution Type         False
Deal Date             False
Sector                False
Location              False
VP Name               False
Manager Name          False
Deal Cost             False
Deal Status Code      False
dtype: bool
```

```
#Calculating the number of missing values
df.isnull().sum()
#79 missing values is extremely less
```

```
Client Category      79
Solution Type         0
Deal Date             0
Sector                0
Location              0
VP Name               0
Manager Name          0
Deal Cost             0
Deal Status Code      0
dtype: int64
```

```
#Visualization of the missing values using a Heat Map
sns.heatmap(df.isnull(), yticklabels = False, cbar=False, cmap='rainbow')
plt.show()
#Missing data is with 'Client Category' as showing in the graph.
```



- From the analysis, it is clear that “Client Category” has 79 missing values in the dataset. Since, it is a very small percentage of missing values, it cannot be deleted.

Initial Data Analysis

```
#Handling the missing data
Client_Category=df['Client Category'].value_counts()
Client_Category
```

```
]: Others                  1763
   Internal                1454
   Services_based          1202
   Tech                     913
   Domestic Public Bank     419
   International Bank       376
   Consulting               352
   Finance                  339
   Telecom                  327
   Power ind                264
   Domestic Private Bank    262
   Insurance                247
   Consumer Good            185
   Automobiles              178
   Infrastructure           152
   Domestic Bank            134
   Retail_market            126
   Govt                     121
   Hospitality              119
   Manufacturing            117
   Pharma                   110
   Healthcare                99
   Electronics               81
   Media_Journal             71
   Industries                66
   Research Development      63
   Energy                    57
   Knowledge                 50
   Management                43
   Govt Bank Special         41
   Payment                   40
```

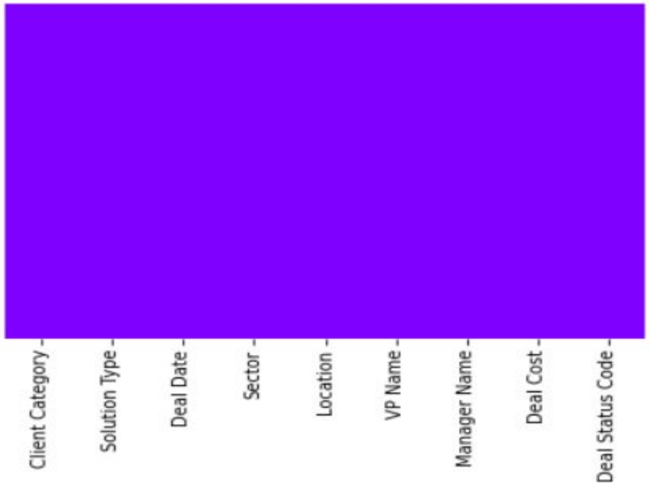
- From the table, it is observed that “Others” have the highest count of about 1763 and the “Payment” Client category has the lowest count of about 40.
- Since there are 79 missing values and “Others” have the most number of values, we attributed “Others” to the 79 missing values.
- We use the function “Mode” for this action because it gives us the maximum frequency.

RUTGERS
UNIVERSITY | NEW BRUNSWICK

Initial Data Analysis

```
df['Client Category'] = df['Client Category'].fillna('Others')
```

```
#Confirming if there are any missing values
sns.heatmap(df.isnull(), yticklabels = False, cbar=False, cmap='rainbow')
plt.show()
```



- Heat Map to confirm that there are No missing values in the data.

Initial Data Analysis

Summary of Categorical Variable

```
sumcat = df.describe(include='O')
# 'O' because Object Catgegory
```

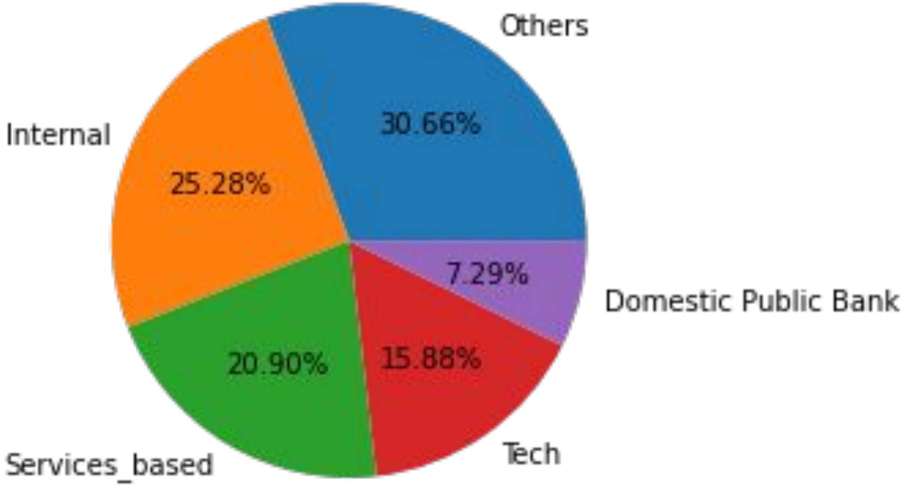
```
sumcat
#Observe no missing data
#This can help us with the analysing high frequency names
```

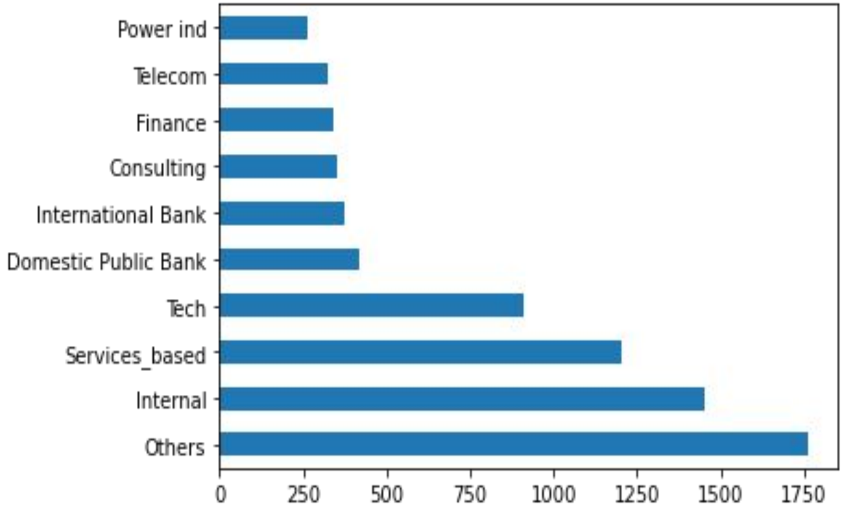
]:

	Client Category	Solution Type	Sector	Location	VP Name	Manager Name	Deal Status Code
count	10061	10061	10061	10061	10061	10061	10061
unique	41	67	25	13	43	278	2
top	Others	Solution 32	Sector 23	L10	Mervin Harwood	Molly Eakes	Lost
freq	1842	1439	2693	3360	1166	323	6306

Initial Data Analysis

Top five domains in Client Category

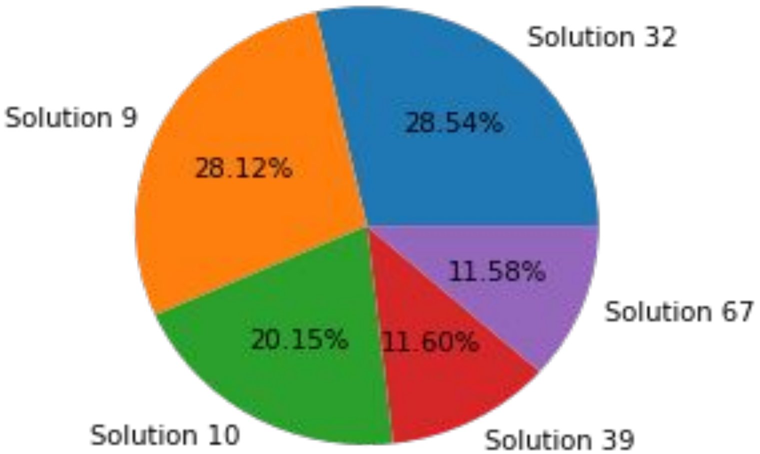


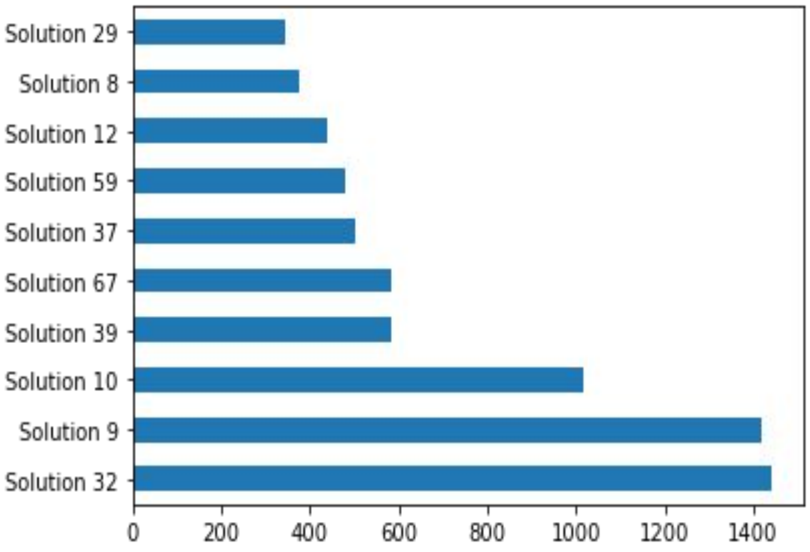


RUTGERS
UNIVERSITY | NEW BRUNSWICK

Initial Data Analysis

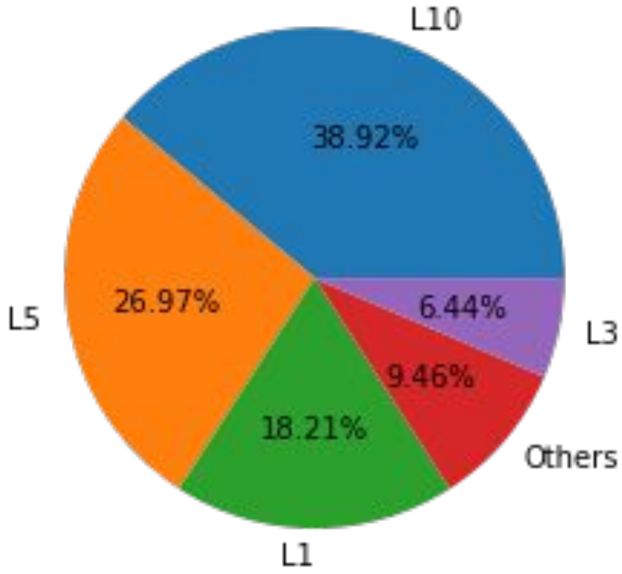
Top five domains in Solution Type

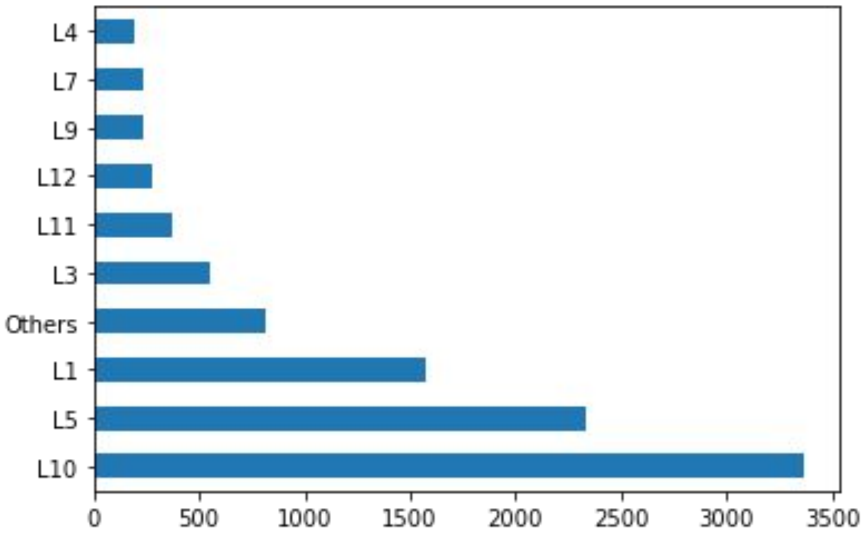




Initial Data Analysis

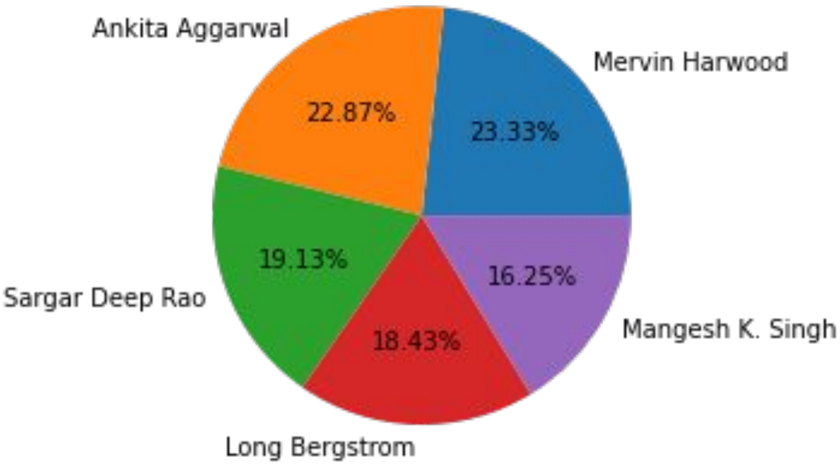
Top five domains in Locations of the deal

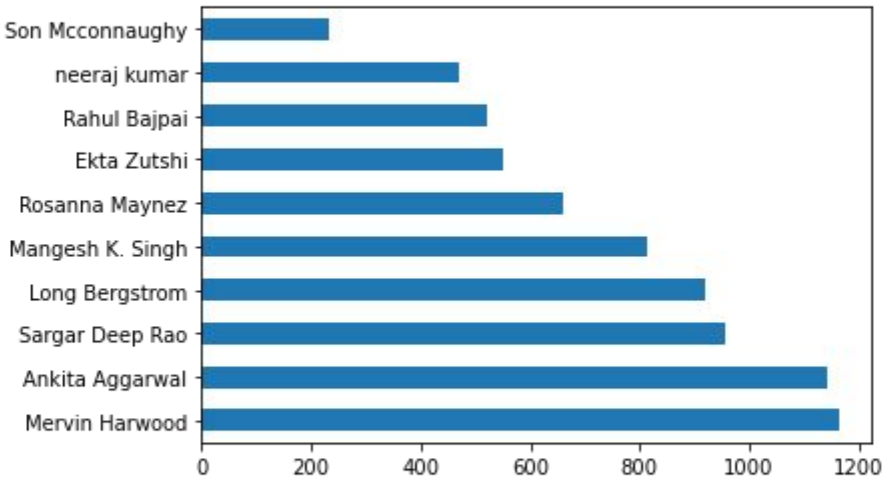




Initial Data Analysis

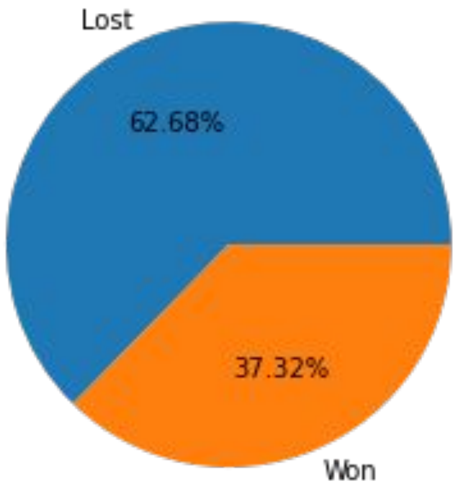
Top five VPs of the deal





Initial Data Analysis

Wins and Loses in the Deal Status Code:




```python
#Deal Status Codes
Deal_Status_Code = df['Deal Status Code'].value_counts()
Deal_Status_Code
#Balanced set
```

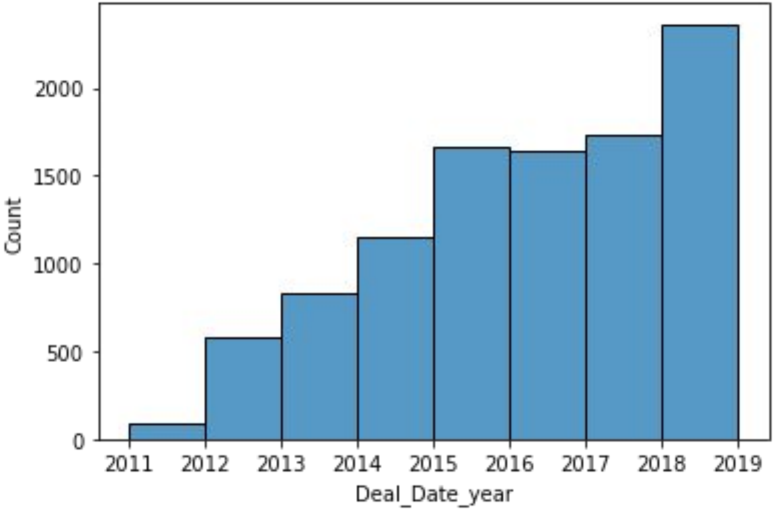
```
]: Lost    6306
   Won     3755
   Name: Deal Status Code, dtype: int64
```

```python
#Index Values
Deal_Index = df['Deal Status Code'].value_counts().index
Deal_Index
```

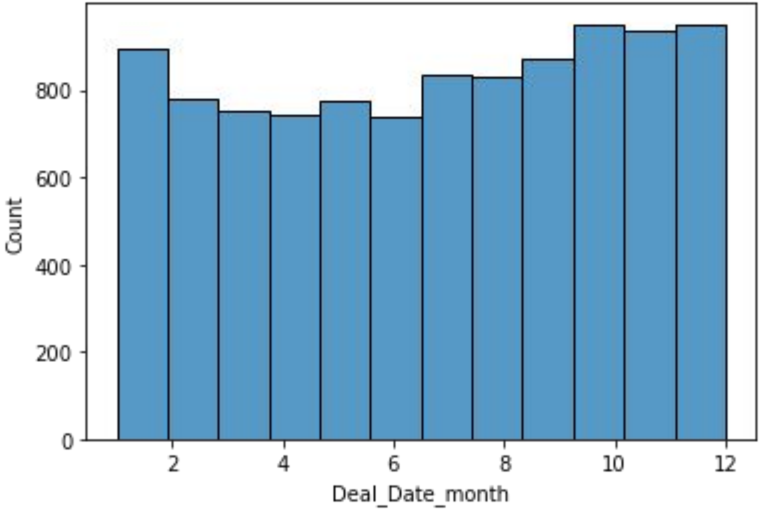
```
]: Index(['Lost', 'Won'], dtype='object')
```

Initial Data Analysis

Analysing the year with most deals:



Analysing the time of year with most deals:



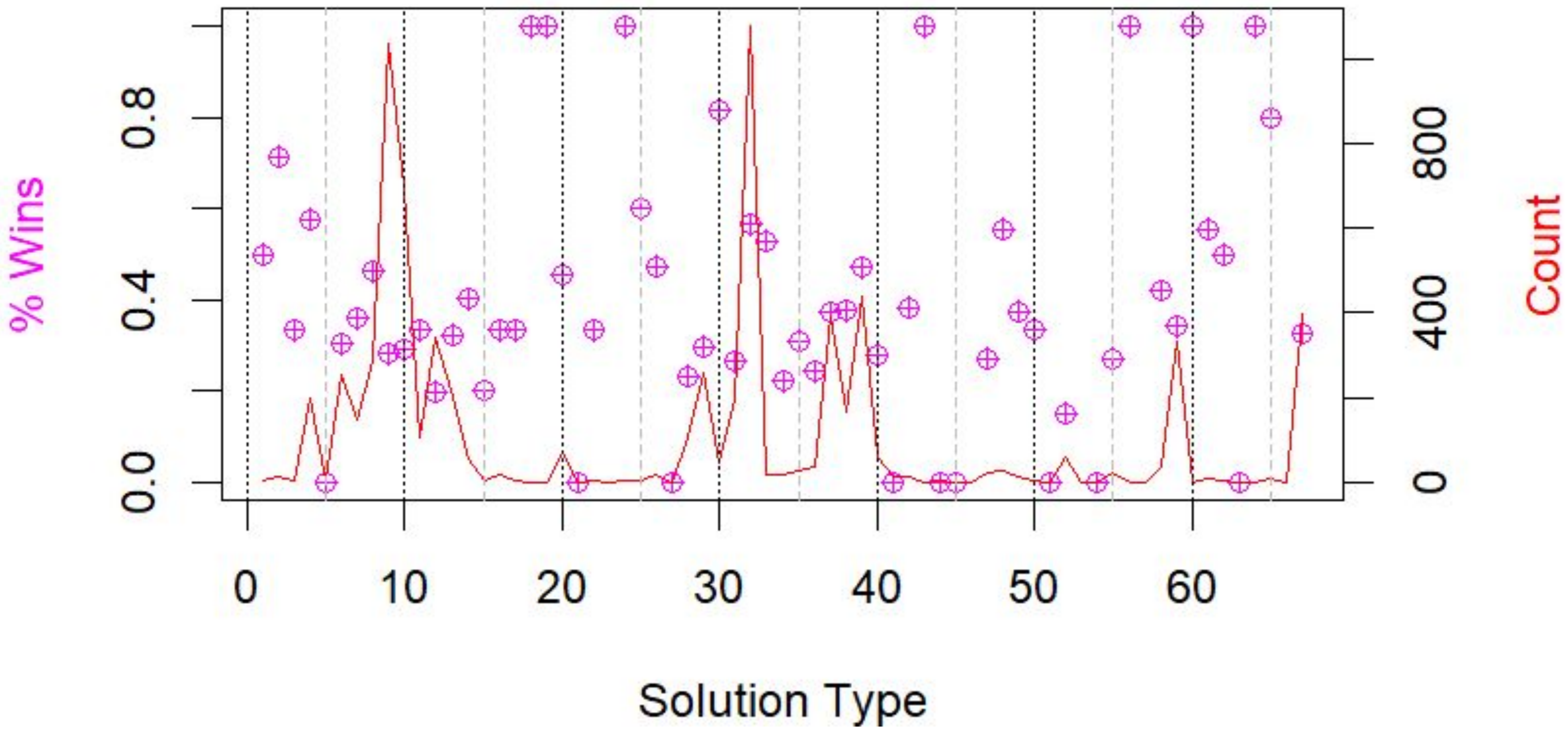
RUTGERS
UNIVERSITY | NEW BRUNSWICK

Initial Data Analysis

Encoding :

```
#The next step is encoding. We have to convert everything to a number to use it in a model
df['Client Category'] = df['Client Category'].astype('category')
df['Client Category'] = df['Client Category'].cat.codes

df['Solution Type'] = df['Solution Type'].astype('category')
df['Solution Type'] = df['Solution Type'].cat.codes

df['Sector'] = df['Sector'].astype('category')
df['Sector'] = df['Sector'].cat.codes

df['Location'] = df['Location'].astype('category')
df['Location'] = df['Location'].cat.codes

df['VP Name'] = df['VP Name'].astype('category')
df['VP Name'] = df['VP Name'].cat.codes

df['Manager Name'] = df['Manager Name'].astype('category')
df['Manager Name'] = df['Manager Name'].cat.codes

df['Deal Status Code'] = df['Deal Status Code'].astype('category')
df['Deal Status Code'] = df['Deal Status Code'].cat.codes
```

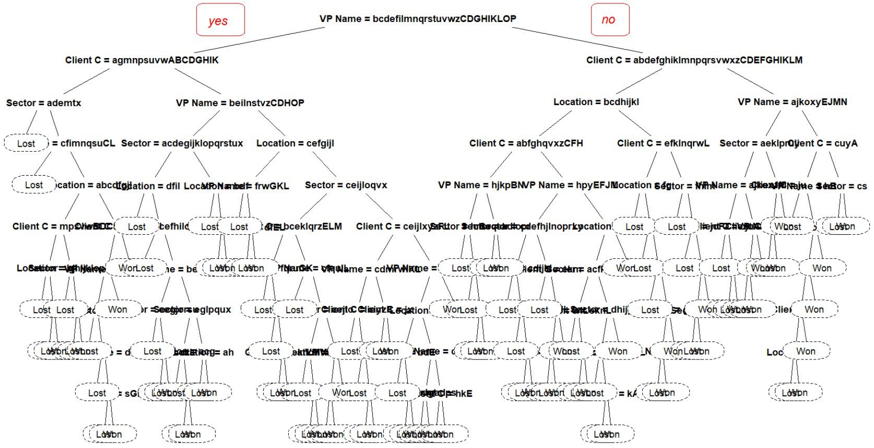
Initial Data Analysis: Solution Type Win Rates



RUTGERS
UNIVERSITY | NEW BRUNSWICK

In Detail: Decision Tree (Peter)

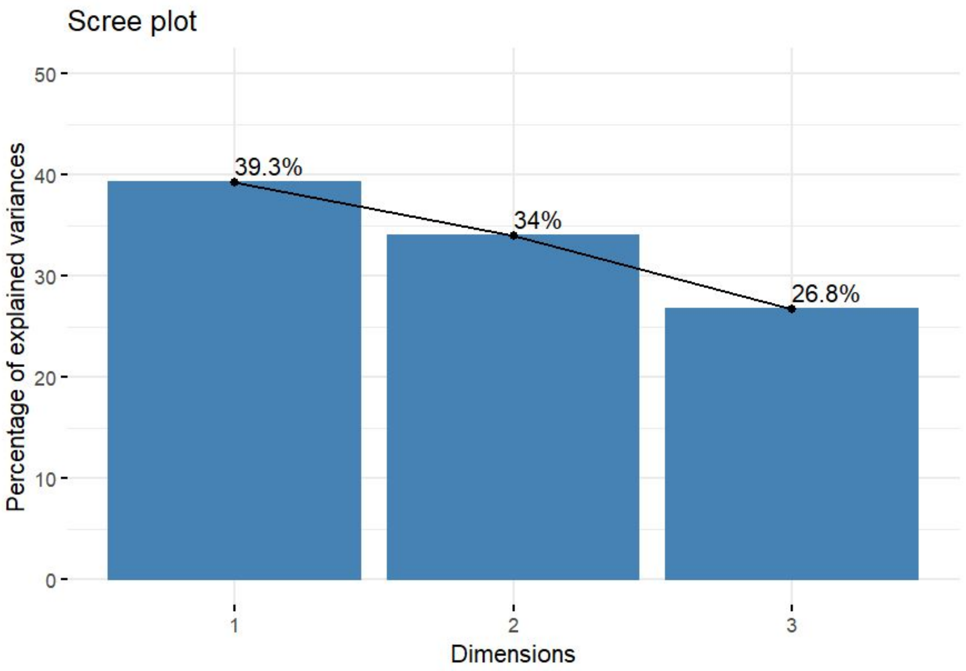
- Conclusions
  - Decent accuracy
    - Average accuracy = 68%
  - Plot is depth-limited
- Analytic Techniques (R)
  - Clean data
    - Remove incomplete or vague objects
    - Remove numerical data
    - Remove infrequent data
  - Split data 70/30
    - Random sample with stratification
  - Tree depth = 10

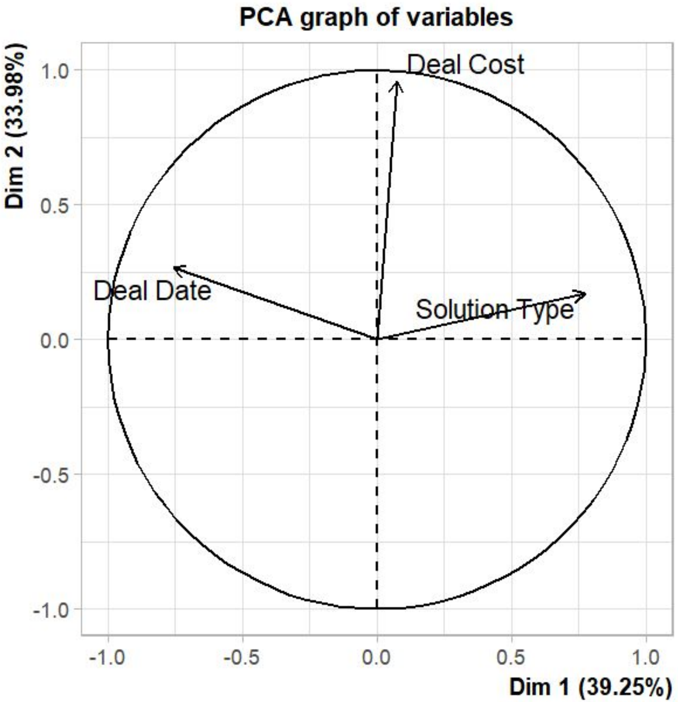


In Detail: Logistic Regression (Peter)

- Conclusion
  - Not a helpful model
    - Not enough correlated numerical data
- Analytic Techniques
  - Clean data
    - Remove incomplete or vague objects
    - Remove/convert categorical data
  - PCA
  - Split data 70/30
    - Random sample with stratification
  - Highest accuracy = 61%

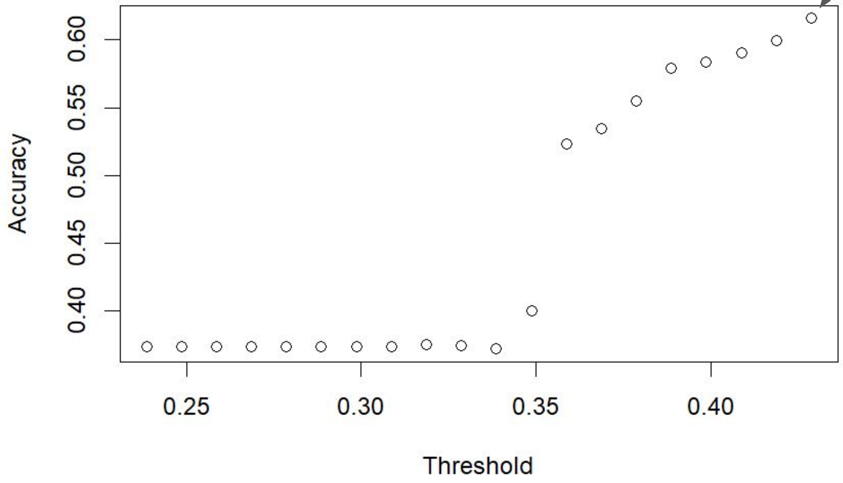
In Detail: Logistic Regression (Peter)

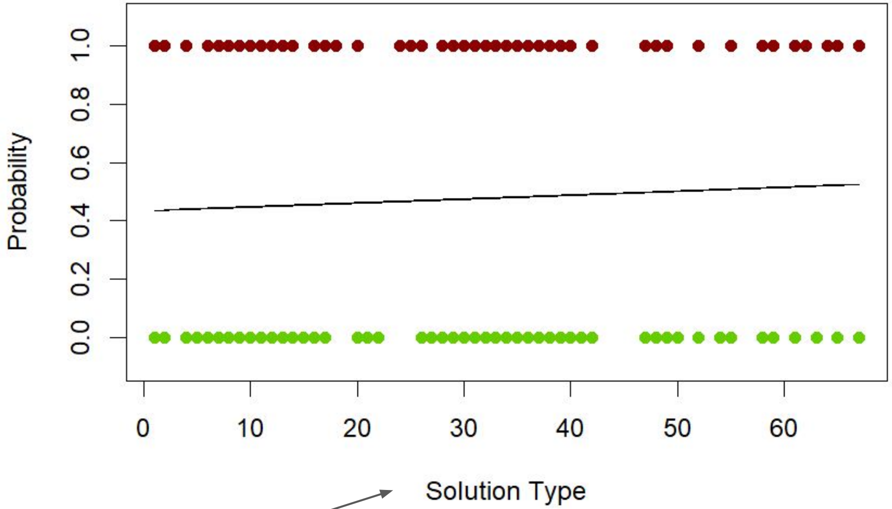




In Detail: Logistic Regression (Peter)

Largest probability during testing

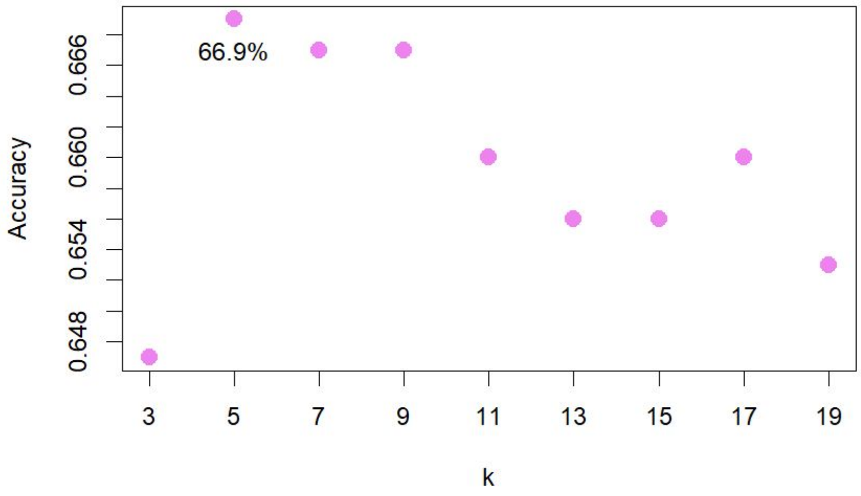




Only significant variable ($p < 0.001$)

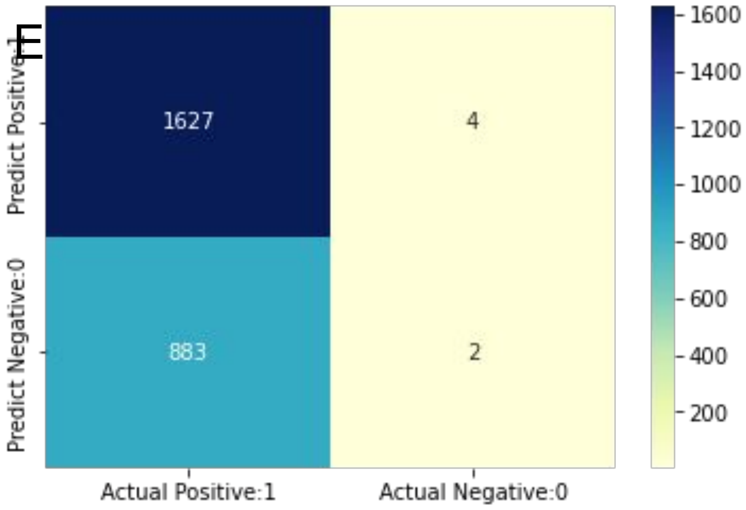
In Detail: KNN (Peter)

- Conclusion
  - Not a helpful model
    - Not enough correlated numerical data
- Analytic Techniques
  - Same data as logistic regression
    - Min-max normalization
  - Split data 70/30
    - Random sample with stratification
  - Highest accuracy = 66.9%
  - Average accuracy = 66.7%
    - (5 iterations of sampling, k=5)



In Detail: Logistic Regression (Avinash)

Confusion Matrix :



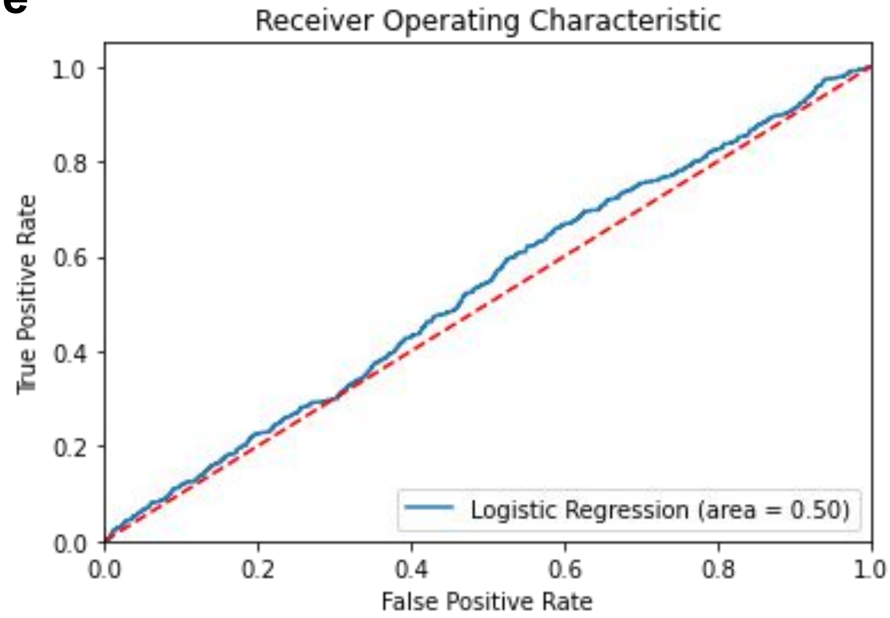
Testing Model Accuracy : 64.75%
Training Model Accuracy : 61.96%

Classification Report Analysis	Percentage Accuracy
Classification Accuracy	64.75%
Classification Error	35.25%
Precision	99.75%
Recall	64.82%
F-1 Score	0.79

RUTGERS
UNIVERSITY | NEW BRUNSWICK

In Detail: Logistic Regression (Avinash)

AUC - ROC Curve



AUC Score : 50%

In Detail: Logistic Regression (Avinash)

Cross Validation for Logistic Regression :

```
#Cross Validation for Logistic Regression
```

```
from sklearn.model_selection import cross_val_score
accuracy_train = cross_val_score(logit,x_train,y_train, cv = 15)
accuracy_test = cross_val_score(logit,x_test,y_test,cv = 15)
```

```
#Training Data
print(accuracy_train)
```

```
[0.62027833 0.62027833 0.61829026 0.6222664  0.61829026 0.62027833
 0.61630219 0.62027833 0.62027833 0.62425447 0.62027833 0.61829026
 0.62027833 0.61829026 0.61431412]
```

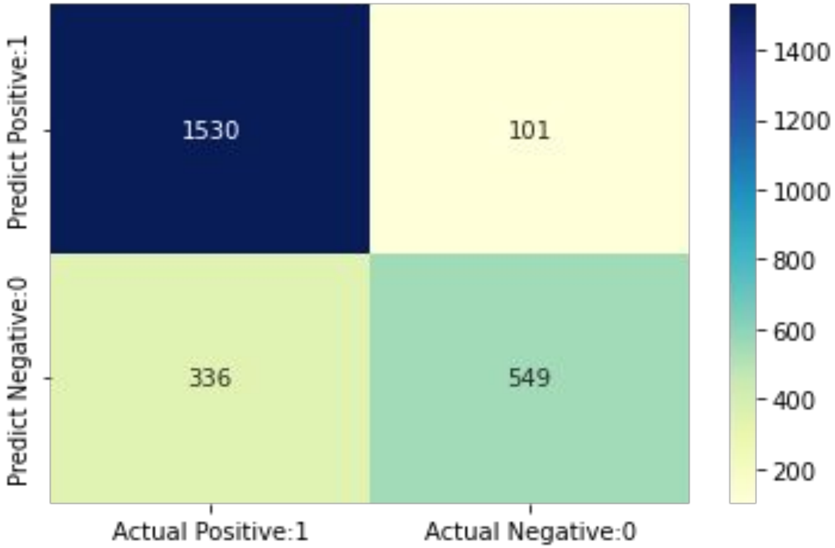
```
print('Average cross-validation score: {:.4f}'.format(accuracy_train.mean()))
```

```
Average cross-validation score: 0.6195
```

RUTGERS
UNIVERSITY | NEW BRUNSWICK

In Detail: Random Forest (Avinash)

Confusion Matrix :

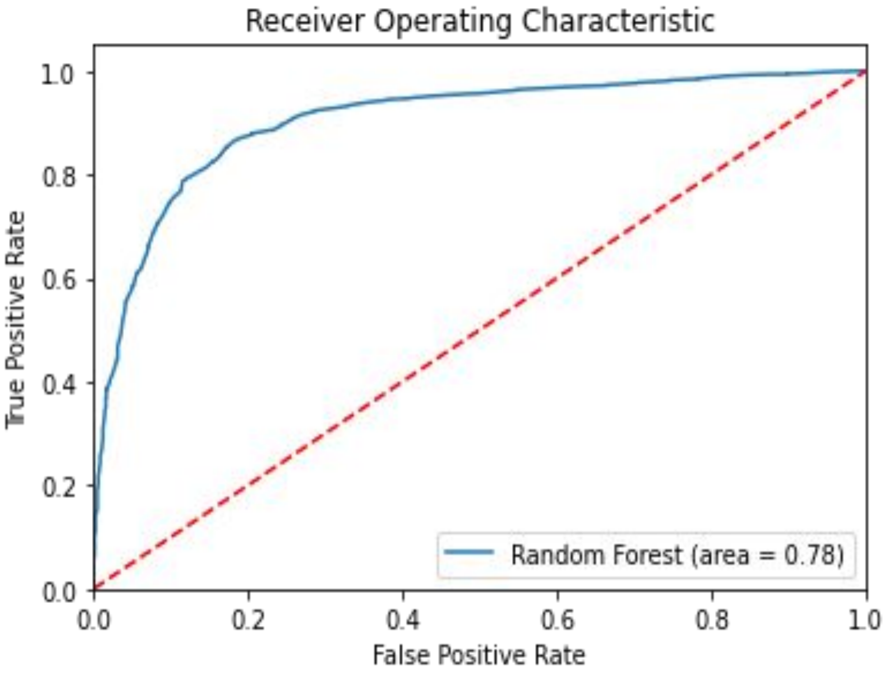


Testing Model Accuracy : 82.63%
Training Model Accuracy : 99.66%

Classification Report Analysis	Percentage Accuracy
Classification Accuracy	82.63%
Classification Error	17.37%
Precision	93.81%
Recall	81.99%
F-1 Score	88%

In Detail: Random Forest (Avinash)

AUC - ROC Curve



AUC Score : 51%

In Detail: Random Forest (Avinash)

Cross Validation for Random Forest :

```
#Cross Validation Method
from sklearn.model_selection import cross_val_score

accuracy_test_rf = cross_val_score(rf,x_test,y_test,cv = 20)

print(accuracy_test_rf)
```

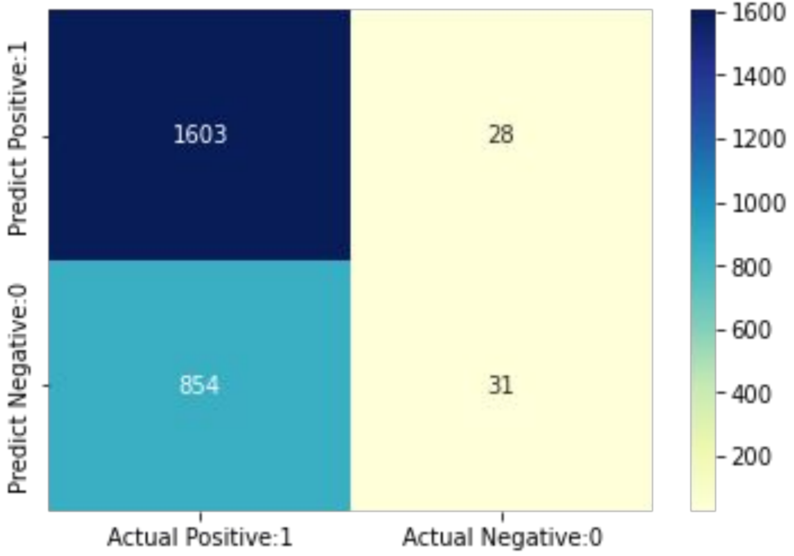
```
[0.66666667 0.66666667 0.6984127  0.72222222 0.6984127  0.74603175
 0.73809524 0.67460317 0.78571429 0.67460317 0.76190476 0.76190476
 0.76984127 0.70634921 0.67460317 0.76190476 0.712      0.712
 0.736      0.808     ]
```

```
print('Average cross-validation score: {:.4f}'.format(accuracy_test_rf.mean()))
```

```
Average cross-validation score: 0.7238
```

In Detail: Naive Bayes (Avinash)

Confusion Matrix :

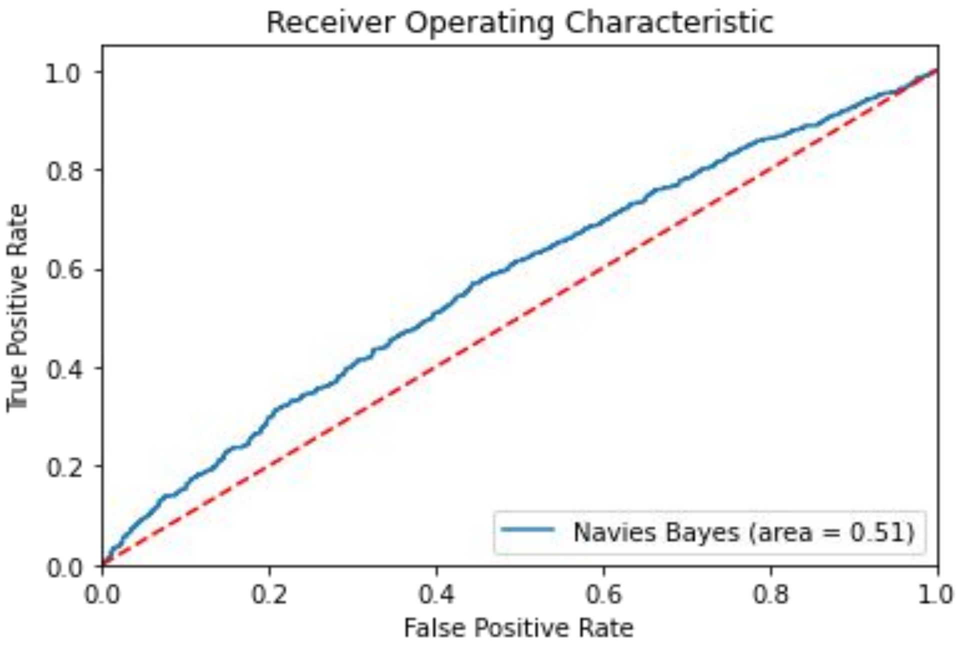


Testing Model Accuracy : 64.94%
Training Model Accuracy : 61.86%

Classification Report Analysis	**Percentage Accuracy**
Classification Accuracy	64.94%
Classification Error	35.06%
Precision	98.28%
Recall	65.24%
F-1 Score	78%

In Detail: Naive Bayes (Avinash)

AUC - ROC Curve



AUC Score : 51%

In Detail: Naive Bayes (Avinash)

Cross Validation for Naive Bayes :

```
#K Fold Cross Validation
```

```
from sklearn.model_selection import cross_val_score

scores = cross_val_score(gnb, x_train, y_train, cv = 10, scoring='accuracy')

print('Cross-validation scores:{}'.format(scores))
```

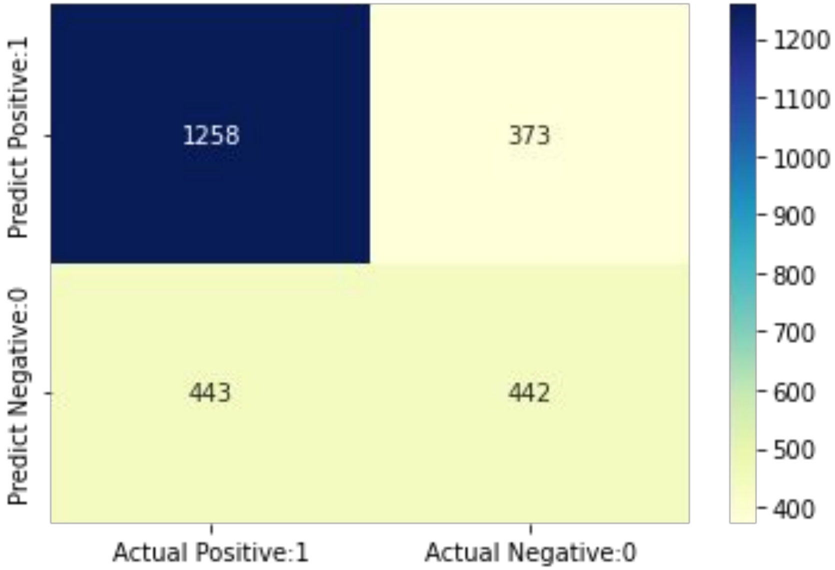
```
Cross-validation scores:[0.61986755 0.61324503 0.62119205 0.61589404 0.61986755 0.62068966
 0.61538462 0.62334218 0.61671088 0.6193634 ]
```

```
print('Average cross-validation score: {:.4f}'.format(scores.mean()))
```

```
Average cross-validation score: 0.6186
```

In Detail: K Nearest Neighbor (Avinash)

Confusion Matrix :

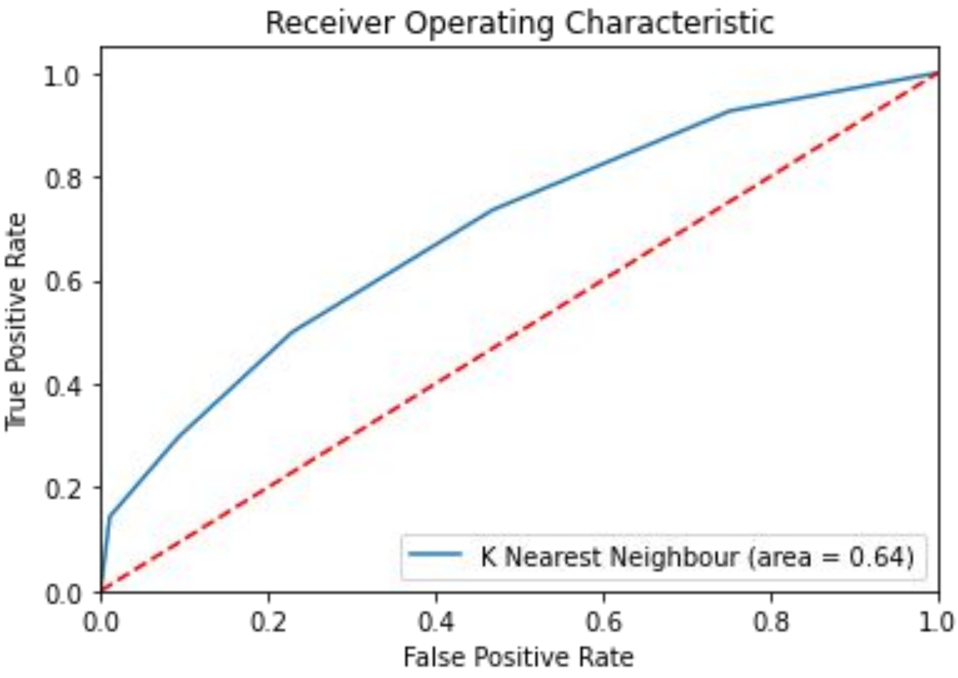


Testing Model Accuracy : 67.57%
Training Model Accuracy : 78.26%

Classification Report Analysis	Percentage Accuracy
Classification Accuracy	67.57%
Classification Error	32.43%
Precision	77.13%
Recall	73.96%
F-1 Score	76%

In Detail: K Nearest Neighbor (Avinash)

AUC - ROC Curve



AUC Score : 64%

In Detail: K Nearest Neighbor (Avinash)

Cross Validation for K Nearest Neighbor :

```
#Cross Validation Method
```

```
from sklearn.model_selection import cross_val_score

scores = cross_val_score(knn, x_train, y_train, cv = 10, scoring='accuracy')

print('Cross-validation scores:{}'.format(scores))
```

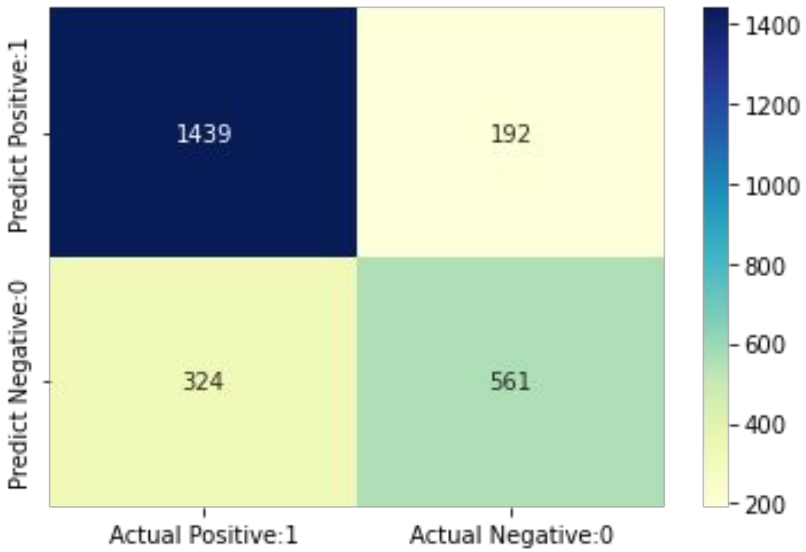
```
Cross-validation scores:[0.69403974 0.68609272 0.67682119 0.65430464 0.67152318 0.65119363
 0.64456233 0.68302387 0.67639257 0.66976127]
```

```
print('Average cross-validation score: {:.4f}'.format(scores.mean()))
```

```
Average cross-validation score: 0.6708
```

In Detail: XG Boost (Avinash)

Confusion Matrix :

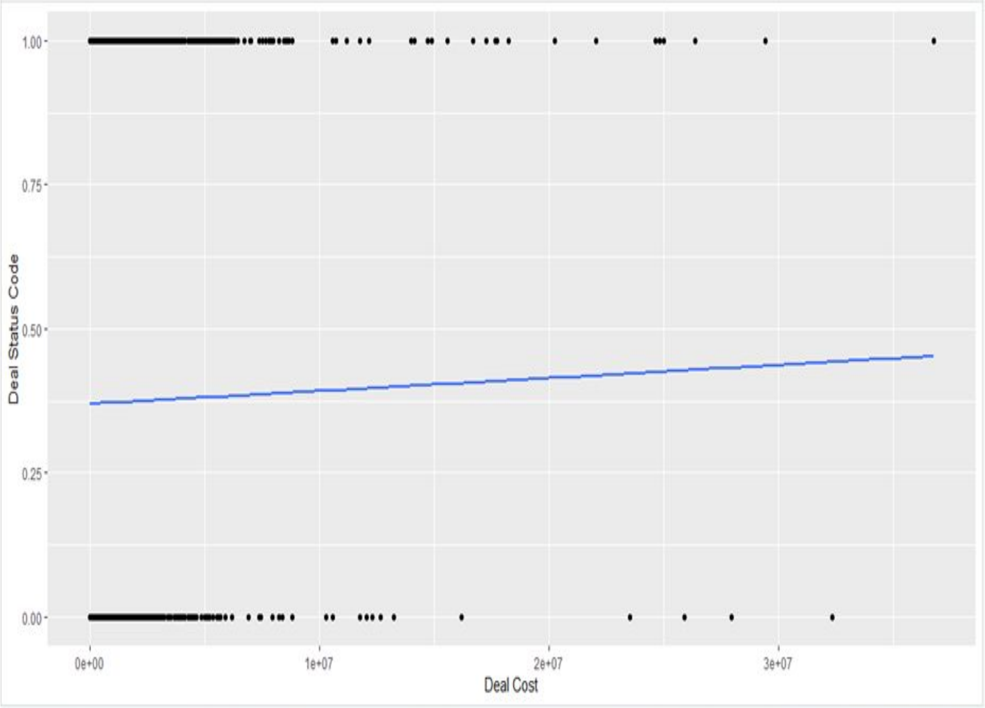


Testing Model Accuracy : 79.49%
Training Model Accuracy : 92.71%

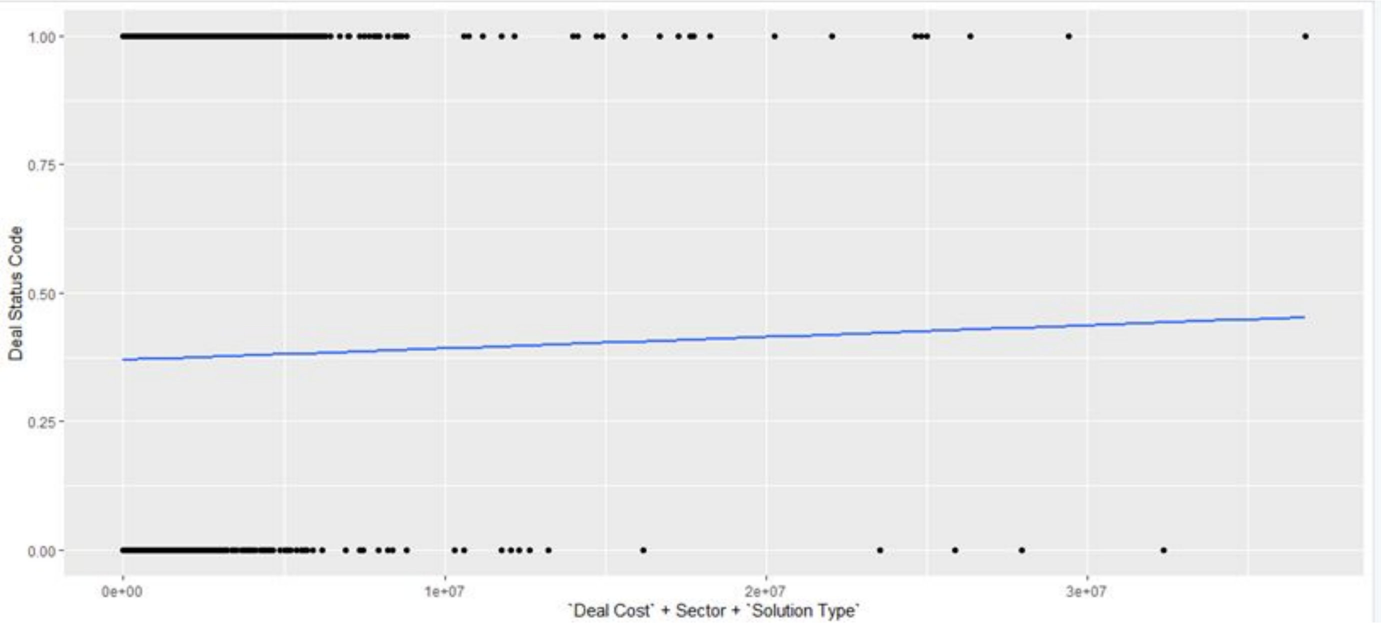
Classification Report Analysis	Percentage Accuracy
Classification Accuracy	67.57%
Classification Error	32.43%
Precision	77.13%
Recall	73.96%
F-1 Score	76%

Logistic Regression(Unmesh)

- Threshold of 0.375 gave highest accuracy.
- 1 indicates bid won.
- 0 indicates bid lost.



Deal Status Code Vs Deal Cost and Solution Type

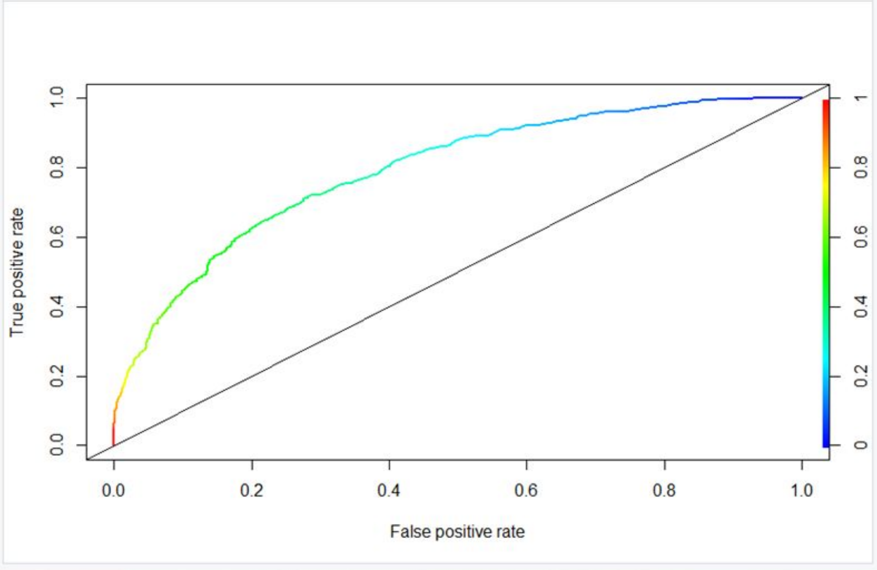


Accuracy =0.7494

Sensitivity=0.7689737

Specificity= 0.7034949

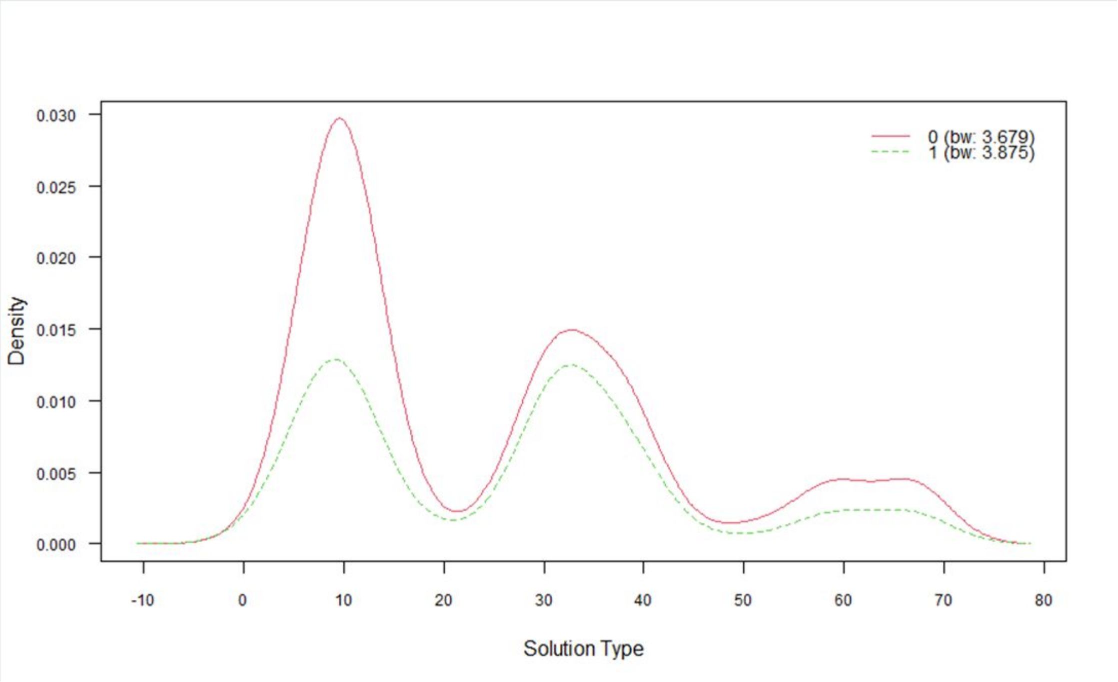
	FALSE	TRUE
Lost	1611	263
Won	484	624



Area under curve: 0.7919446

Naive-Bayes (Unmesh)

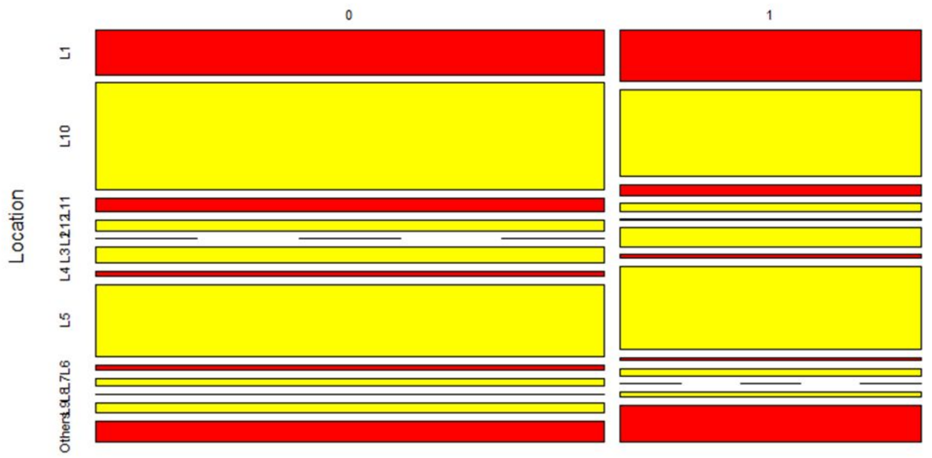
- Distribution of Solution types according to their win/loss status.



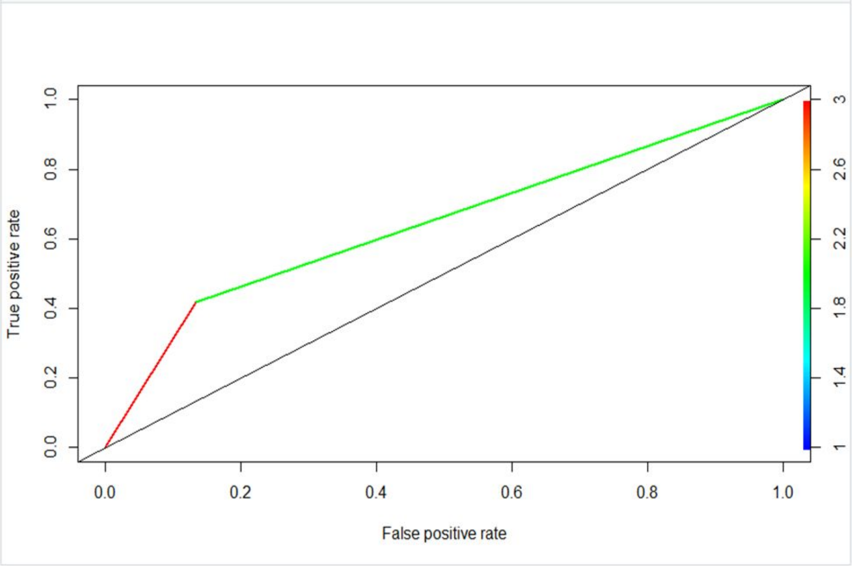
Naive-Bayes(Unmesh)

- Distribution of Won/Loss depending on Location.

Location	0	1
L1	0.1424759872	0.1624548736
L10	0.3425827108	0.2788808664
L11	0.0410885806	0.0333935018
L12	0.0352187834	0.0243682310
L2	0.0000000000	0.0009025271
L3	0.0496264674	0.0595667870
L4	0.0165421558	0.0108303249
L5	0.2315901814	0.2644404332
L6	0.0149413020	0.0090252708
L7	0.0256136606	0.0225631769
L8	0.0016008538	0.0000000000
L9	0.0314834578	0.0144404332
Others	0.0672358591	0.1191335740



Naive-Bayes (Unmesh)




```
[1] 0.641966
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1623  251
         1  645  463

               Accuracy : 0.6995
                 95% CI : (0.6827, 0.716)
    No Information Rate : 0.7606
    P-Value [Acc > NIR] : 1

                  Kappa : 0.3062

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.7156
            Specificity : 0.6485
         Pos Pred Value : 0.8661
         Neg Pred Value : 0.4179
             Prevalence : 0.7606
         Detection Rate : 0.5443
   Detection Prevalence : 0.6284
      Balanced Accuracy : 0.6820

       'Positive' Class : 0
```

Decision Tree (Unmesh)

- All attributes categorical.
- Unique observations in each of the attributes are eliminated.
- Important classifier to determine key attributes
- Pruned decision tree with a minimum split of 700 and

  mincriterion(depth of the tree) of 0.90

```
Confusion Matrix and Statistics

          Reference
Prediction Lost  Won
      Lost 1658  518
      Won   216  590

               Accuracy : 0.7539
                 95% CI : (0.738, 0.7692)
    No Information Rate : 0.6284
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4418

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8847
            Specificity : 0.5325
         Pos Pred Value : 0.7619
         Neg Pred Value : 0.7320
             Prevalence : 0.6284
         Detection Rate : 0.5560
   Detection Prevalence : 0.7297
      Balanced Accuracy : 0.7086

       'Positive' Class : Lost
```

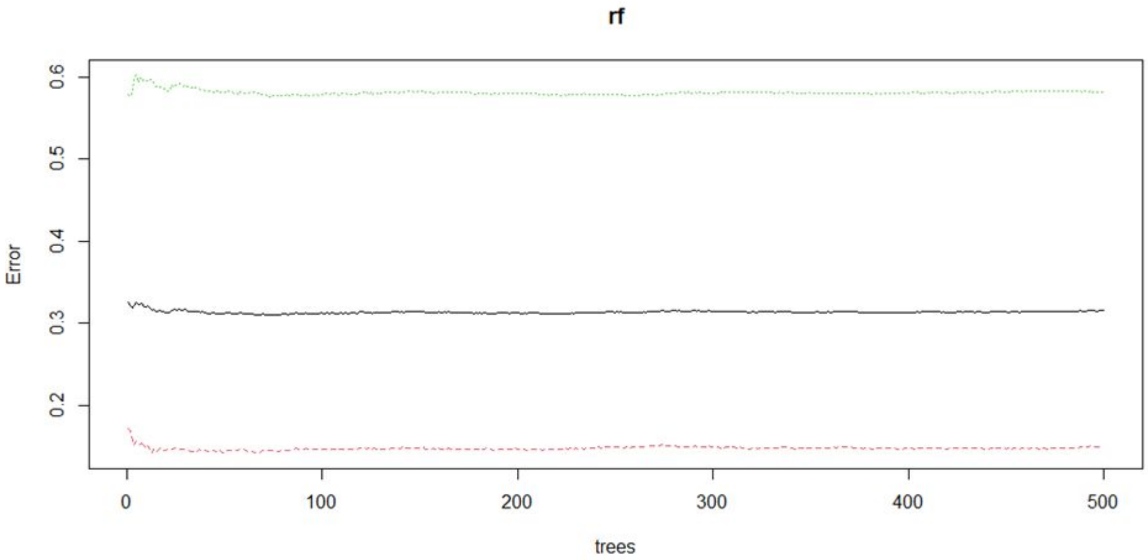
Random Forest(Unmesh)

```
Confusion Matrix and Statistics

          Reference
Prediction Lost  Won
      Lost 1573  541
      Won   184  569

               Accuracy : 0.7471
                 95% CI : (0.7308, 0.7629)
    No Information Rate : 0.6128
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.4336

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8953
            Specificity : 0.5126
         Pos Pred Value : 0.7441
         Neg Pred Value : 0.7556
             Prevalence : 0.6128
         Detection Rate : 0.5487
   Detection Prevalence : 0.7374
      Balanced Accuracy : 0.7039

       'Positive' Class : Lost
```

Random forest(Unmesh)

- Negligible error

  Variation after 500 tress.



Random Forest (Unmesh)

Area under curve: 0.7364111