Ecommerce Shipping

Delivery Prediction

-Dikshant Joshi

Agenda

- PROJECT OVERVIEW
- DATASET
- EDA
- MODELS
- INFERENCES
- CONCLUSIONS

Project Overview

- Problem: An international e-commerce company is aiming to enhance Customer Satisfaction by leveraging advanced machine learning techniques on their E-commerce Data to predict Ontime Delivery(Yes/No).
- Goal: Analyze key features influencing on-time delivery and implement predictive models for accurate delivery predictions, optimizing operational efficiency.

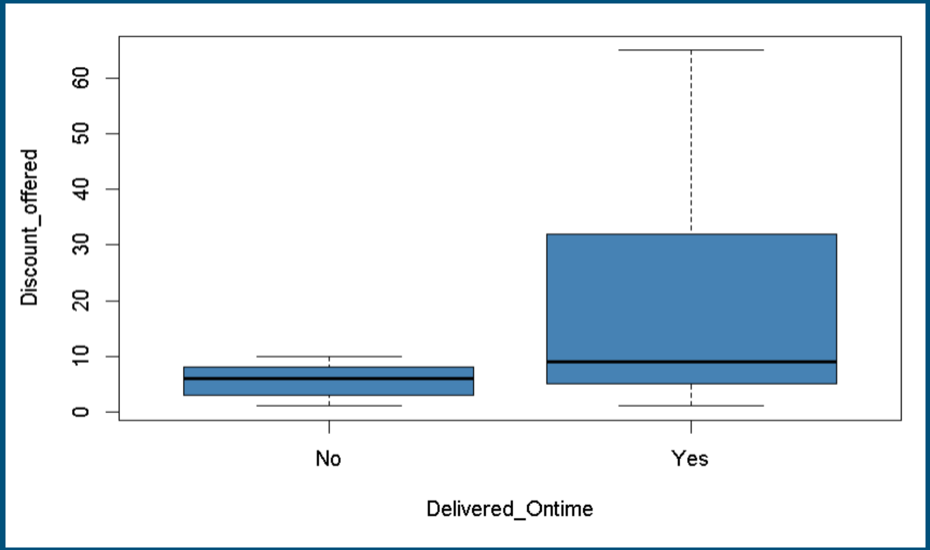
Dataset

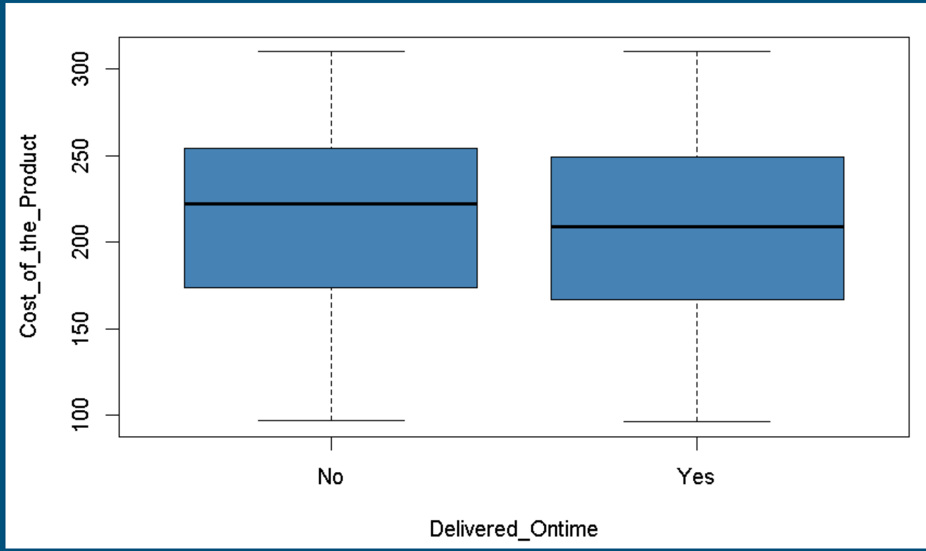
ID	Number of customer
Warehouse Block	The company has a big warehouse which is divided into blocks such as A,B,C,D,E
Mode of Shipment	The company ships the products in multiple ways such as Ship, Flight and Road
Customer Care Calls	The number of calls made for enquiry of the shipment
Customer Rating	The company has been rated by every customer. 1 is the lowest (Worst), 5 is the highest (Best)
Cost Of Product	Cost of the Product in US Dollars
Prior Purchases	The Number of Prior Purchase
Product Importance	The company has categorized the product in the various parameter such as low, medium, high
Gender	Male and Female
Discount Offered	Discount offered on that specific product
Weight in gms	It is the weight in grams
Reached on Time	It is the target variable, where 1 Indicates that the product has reached on time and 0 indicates it has not reached on time.

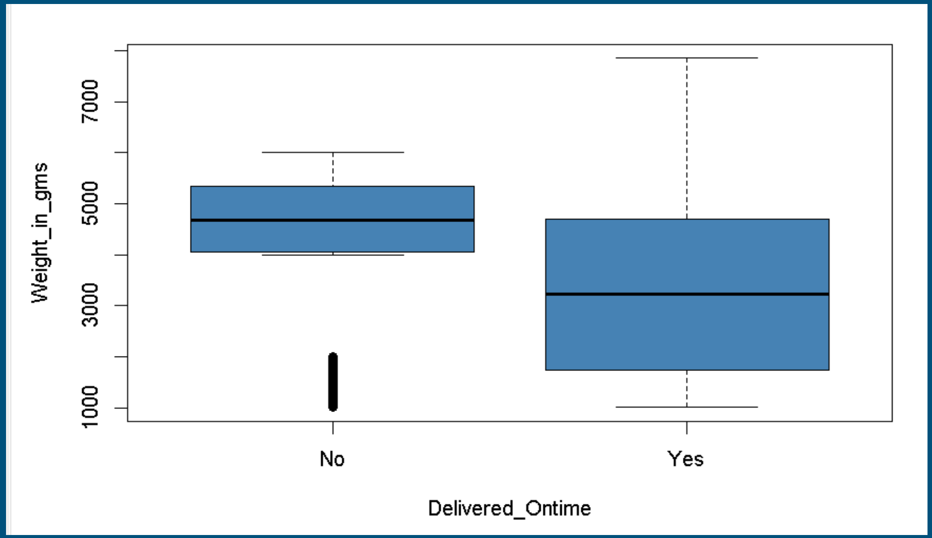
- E-Commerce Shipping Data
- The dataset consists of 10999 observations and 12 attributes.
- Multivariate dataset
- Binary Classification Problem

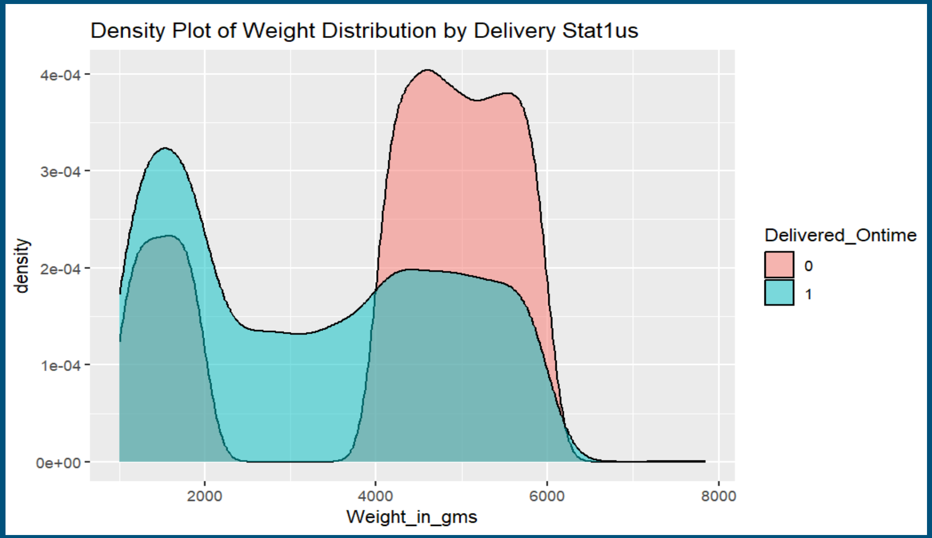
Reference: Kaggle Ecommerce Shipping Data

EDA - Numerical Variables

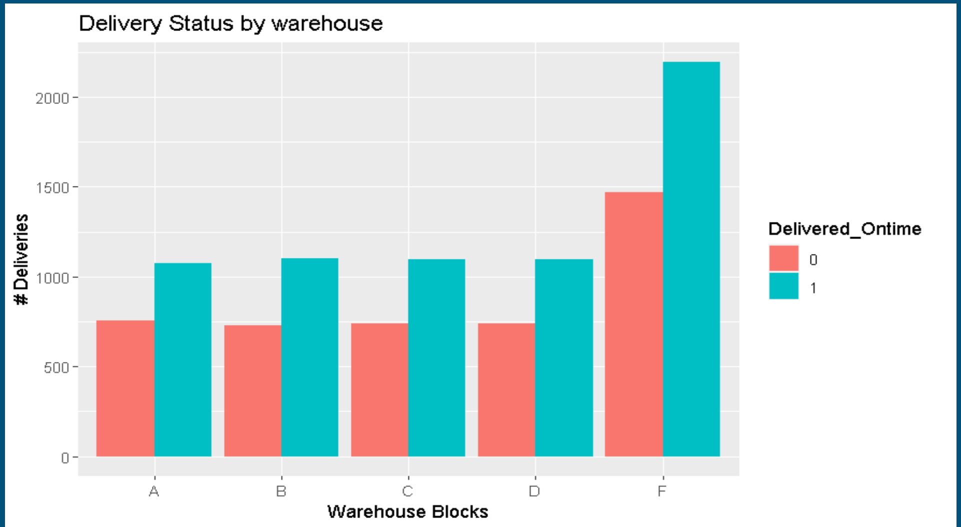


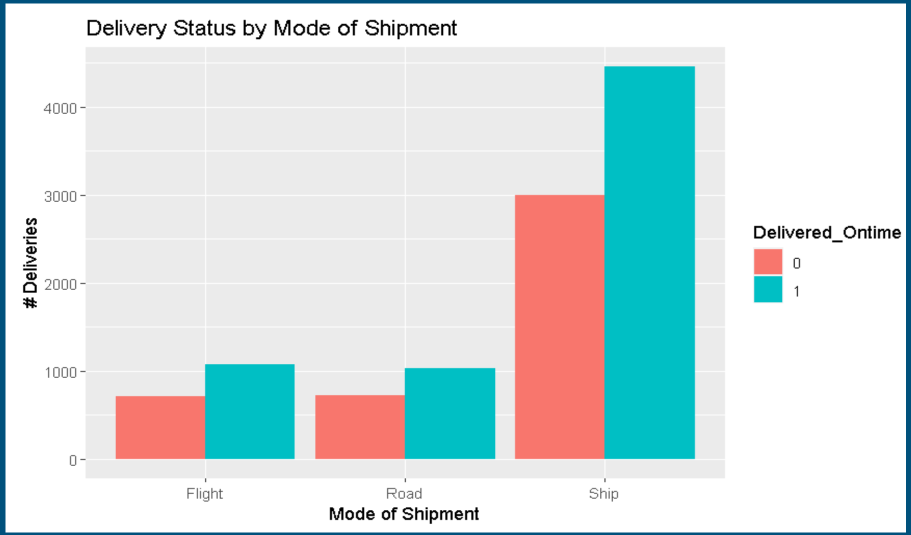


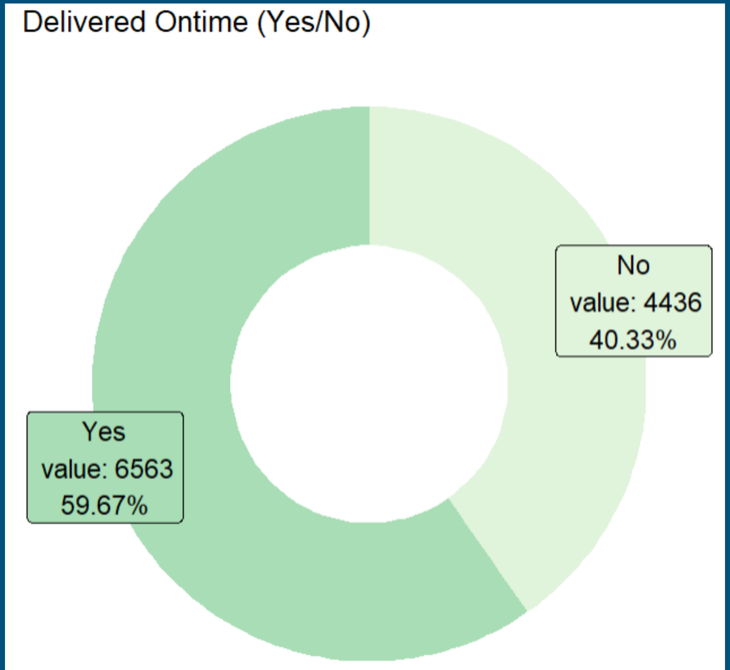


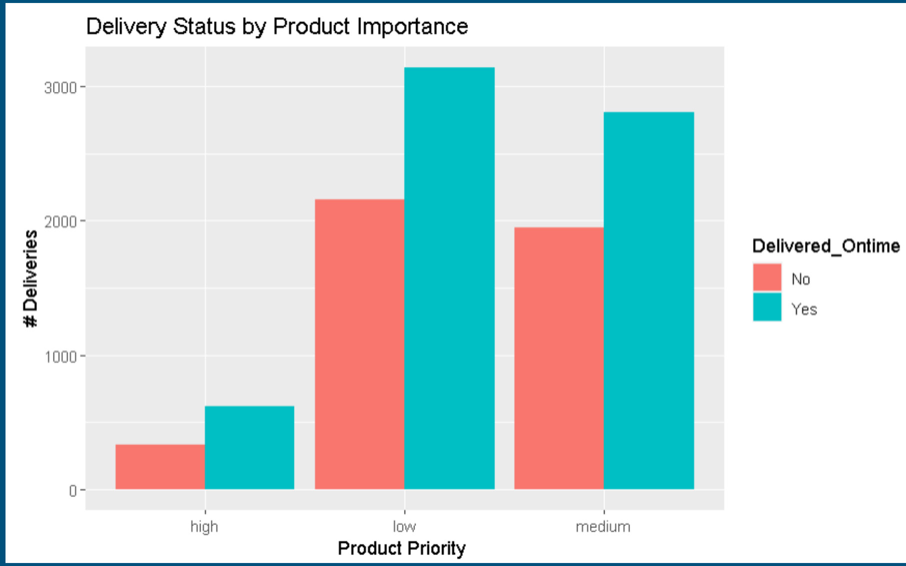


EDA - Categorical Variables









Logistic Regression

Logistic regression is like a statistical method used to analyze and model the relationship between a dependent variable and independent variables.

Confusion Matrix:

	Reference 0	1
Predicted 0	770	636
1	560	1332

```
Accuracy          : 63.74%
95% CI            : (0.6207, 0.6538)
No Info Rate      : 0.5967
P-Value           : 9.379e07
Sensitivity       : 0.5789
Specificity       : 0.6768
```

```
Call:
glm(formula = Delivered_Ontime ~ ., family = binomial, data = train_df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7320  -1.0720   0.1267   1.0889   1.9250

Coefficients:
                        Estimate Std. Error z value Pr(>|z|)
(Intercept)            1.627e+00  2.380e-01   6.835 8.20e-12 ***
Warehouse_blockB       1.106e-01  8.497e-02   1.302 0.192903
Warehouse_blockC       5.746e-02  8.444e-02   0.680 0.496217
Warehouse_blockD       9.169e-02  8.433e-02   1.087 0.276927
Warehouse_blockF       9.005e-02  7.360e-02   1.224 0.221132
Mode_of_ShipmentRoad  -9.017e-02  8.592e-02  -1.049 0.293962
Mode_of_ShipmentShip  -7.850e-02  6.764e-02  -1.161 0.245825
Customer_care_calls   -1.054e-01  2.408e-02  -4.377 1.20e-05 ***
Customer_rating        4.482e-02  1.734e-02   2.584 0.009762 **
Cost_of_the_Product   -2.119e-03  5.618e-04  -3.772 0.000162 ***
Prior_purchases       -7.230e-02  1.703e-02  -4.246 2.18e-05 ***
Product_importancelow -3.827e-01  9.329e-02  -4.103 4.09e-05 ***
Product_importancemedium -3.403e-01  9.350e-02  -3.640 0.000273 ***
GenderM                1.764e-02  4.888e-02   0.361 0.718256
Discount_offered       1.112e-01  4.954e-03  22.458  < 2e-16 ***
Weight_in_gms         -2.421e-04  1.803e-05 -13.428  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 11885.6  on 8798  degrees of freedom
Residual deviance:  9586.3  on 8783  degrees of freedom
AIC: 9618.3

Number of Fisher Scoring iterations: 6
```

k-nearest neighbour (KNN)

k-nearest neighbour (KNN) algorithm is a non - parametric supervised machine learning algorithm.

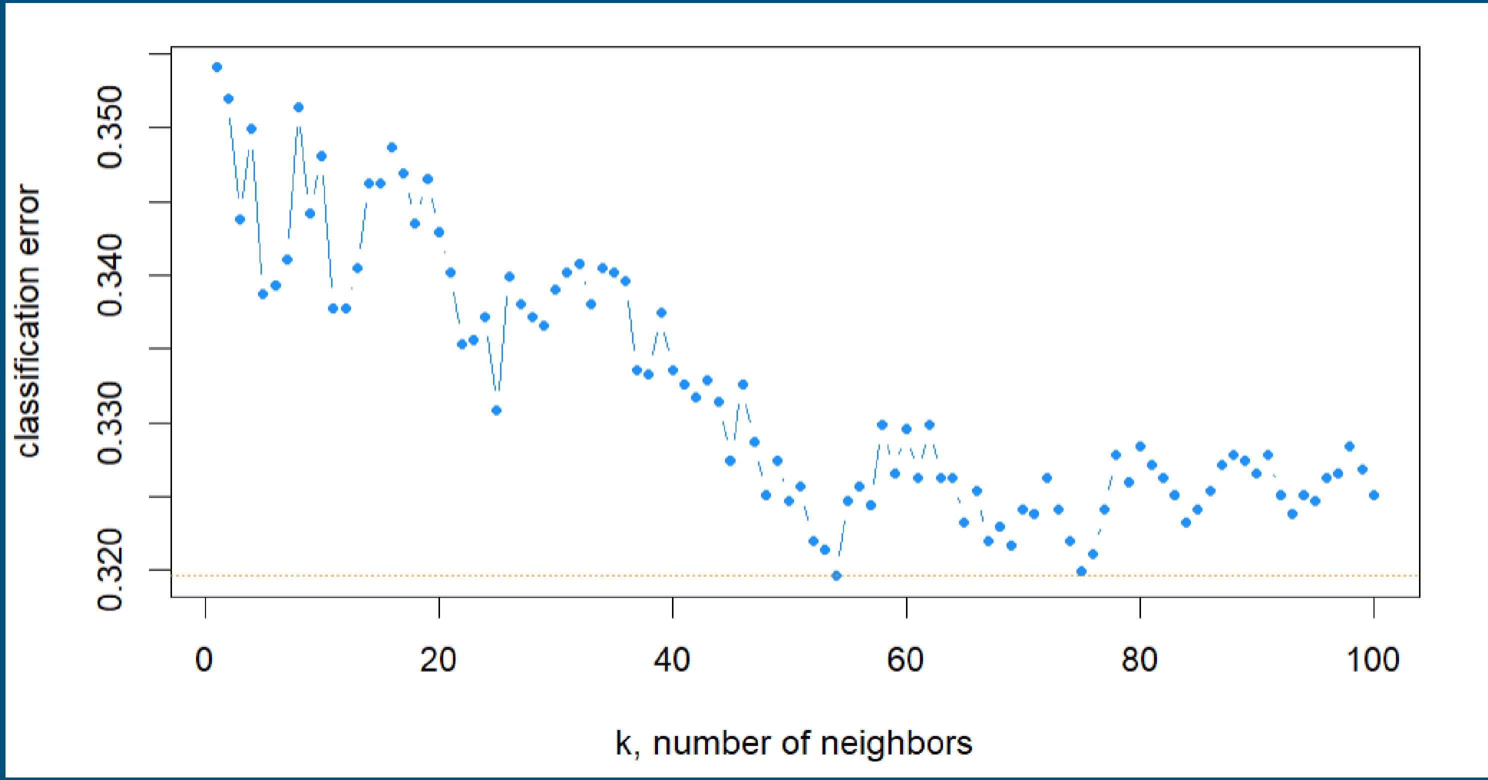
It uses proximity/distance to classify a data point to a group.

For our Model:

1. We converted all text data into numeric values and factored all categorical variables.
2. Chose k value by creating iteration for 100 k values and checking the lowest error value.

	k =12	**k=54 (lowest error %)**
Error %	33.9	32.5

k-nearest neighbour (KNN)



Code: k value for the lowest error

```
set.seed(1994)
k.grid=1:100
error=rep(0, length(k.grid))
knn.train=train_knn[,1:11]
knn.test=test_knn[,1:11]
for (i in seq_along(k.grid)) {
   pred = knn(train = knn.train,
                  test  = knn.test,
                  cl    = knn.trainLabels,
                  k     = k.grid[i])
    error[i] = mean(knn.testLabels !=pred)
}

which.min(error)
```

k-nearest neighbour (KNN)

For k = 12

Confusion Matrix

	Reference 0	1
Predicted 0	879	667
1	451	1301

Accuracy : **66.1%**
95% CI : (0.6446, 0.6772)
No Info Rate : 0.5967
P-Value : 1.682e-14
Sensitivity : 0.6609
Specificity : 0.6611

For k = 54

Confusion Matrix

	Reference 0	1
Predicted 0	1129	871
1	201	1097

Accuracy : **67.5%**
95% CI : (0.6587, 0.6909)
No Info Rate : 0.5967
P-Value : < 2.2e-16
Sensitivity : 0.8489
Specificity : 0.5574

Tree Based Model (Classification Tree)

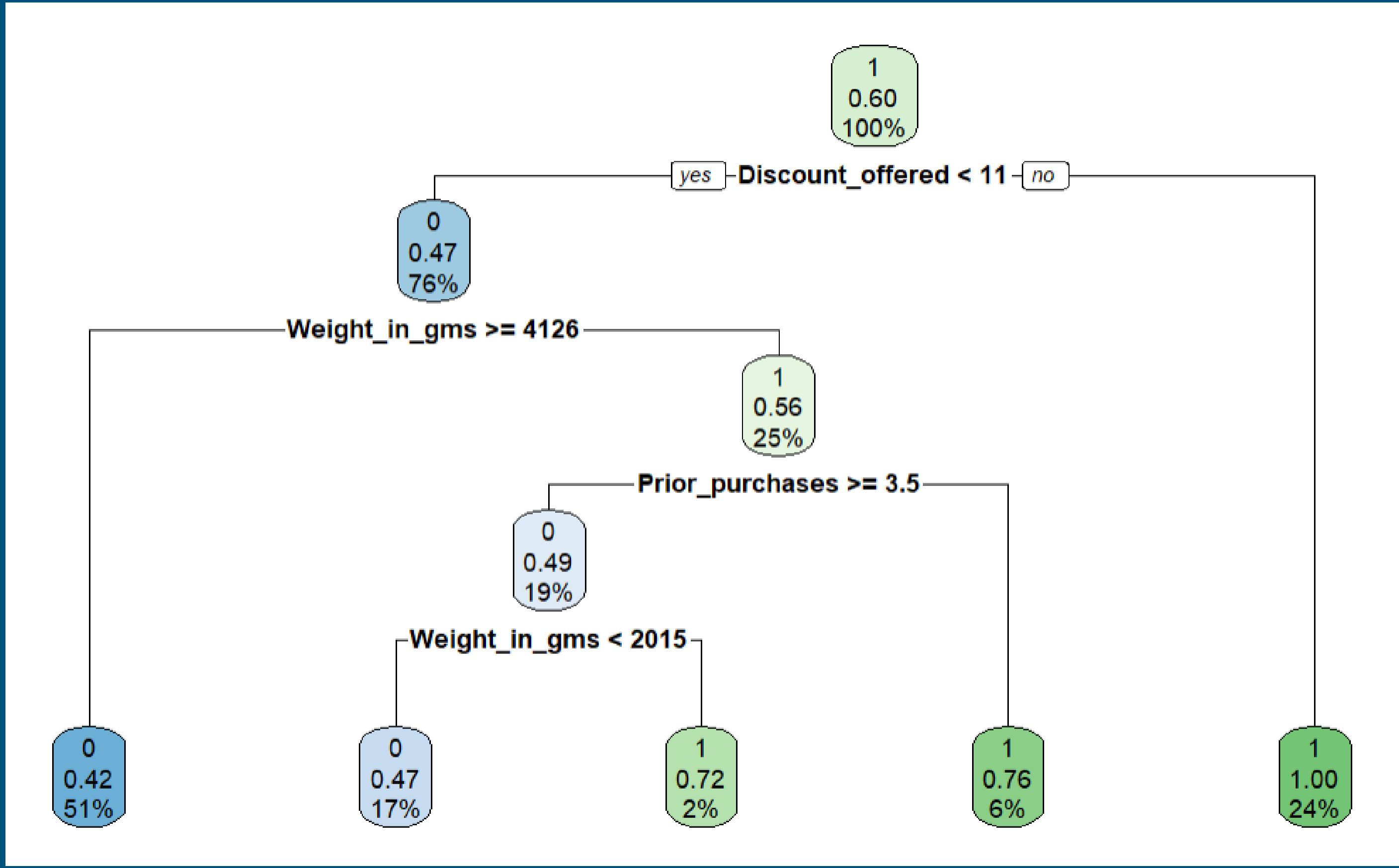
Tree-based models use a decision tree to represent how different input variables can be used to predict a target value.

We did Classification tree algorithm which is a structural mapping of binary decisions that lead to a decision in a class.

Confusion Matrix

	True 0	1
Predicted 0	1252	980
1	78	988

```
Accuracy        : 67.92%
95% CI          : (0.663, 0.6951)
No Info Rate    : 0.5967
P-Value         : < 2.2e-16
Sensitivity     : 0.9414
Specificity     : 0.5020
```



Tree Based Model (Random Forest)

Tree-based models use a decision tree to represent how different input variables can be used to predict a target value.

We did Classification tree algorithm which is a structural mapping of binary decisions that lead to a decision in a class.

Confusion Matrix

	True 0	1
Predicted 0	1065	819
1	265	1149

```
Call:
 randomForest(formula = Delivered_Ontime ~ ., data = train_df,      importance = TRUE,
proximity = TRUE, ntree = 500, mtry = 2,      nodesize = 10)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 32.32%
Confusion matrix:
     0    1 class.error
0 2497  609   0.1960721
1 1880 2715   0.4091404
```

```
Accuracy          : 67.13%
95% CI            : (0.655, 0.6873)
No Info Rate      : 0.5967
P-Value           : < 2.2e-16
Sensitivity       : 0.8008
Specificity       : 0.5838
```

Boosting Model (XG Boost)

Tree-based models use a decision tree to represent how different input variables can be used to predict a target value.

We did Classification tree algorithm which is a structural mapping of binary decisions that lead to a decision in a class.

Confusion Matrix

	True 0	1
Predicted 0	1232	950
1	98	1018

```
Accuracy          : 68.22%
95% CI            : (0.666, 0.6981)
No Info Rate      : 0.5967
P-Value           : < 2.2e-16
Sensitivity       : 0.9263
Specificity       : 0.5173
```

```
param <- list(objective = "binary:logistic",
              eval_metric = "logloss",
              eta = 0.1,
              max_depth = 3,
              alpha = 7,    # L1 regularization term
              gamma = 1   #Minimum reduction in the loss
)

xgb_model_f <- xgboost(params = param,
               data = xgboost_train,
               nrounds = 1000,
              early_stopping_rounds = 100)
```

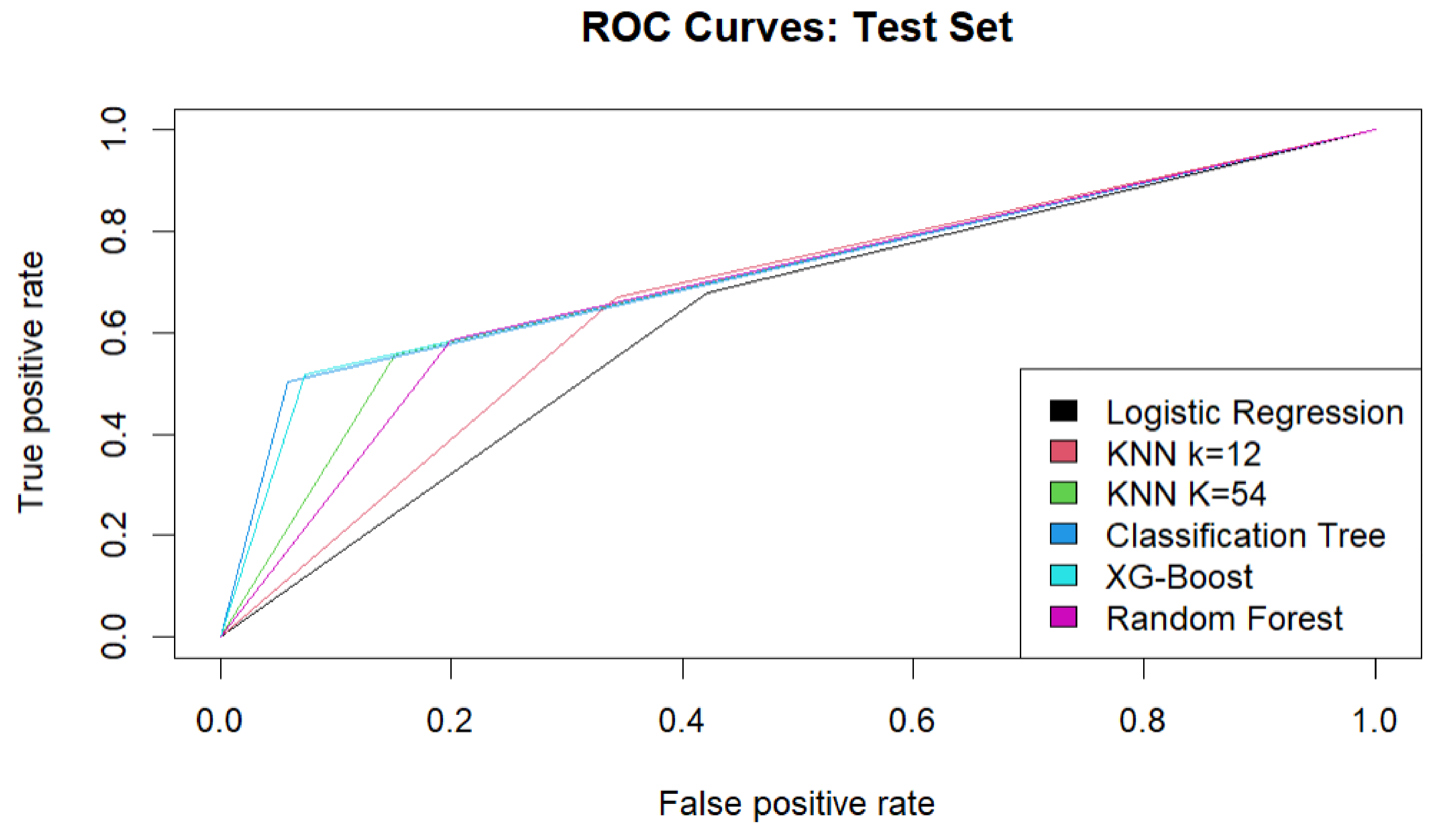
Metric Comparison – Sensitivity/Accuracy

We notice that the classification tree and XG Boost gives better accuracy and Sensitivity(TP Rate) compared to all models.

Models	Logistic Regression	kNN k=12	kNN k=54	Classification Tree	XG Boost	Random Forest
Accuracy %	63.89	66.1	67.8	67.53	68.22	66.03
Sensitivity	58.29	66.09	84.89	93.57	92.63	69.79

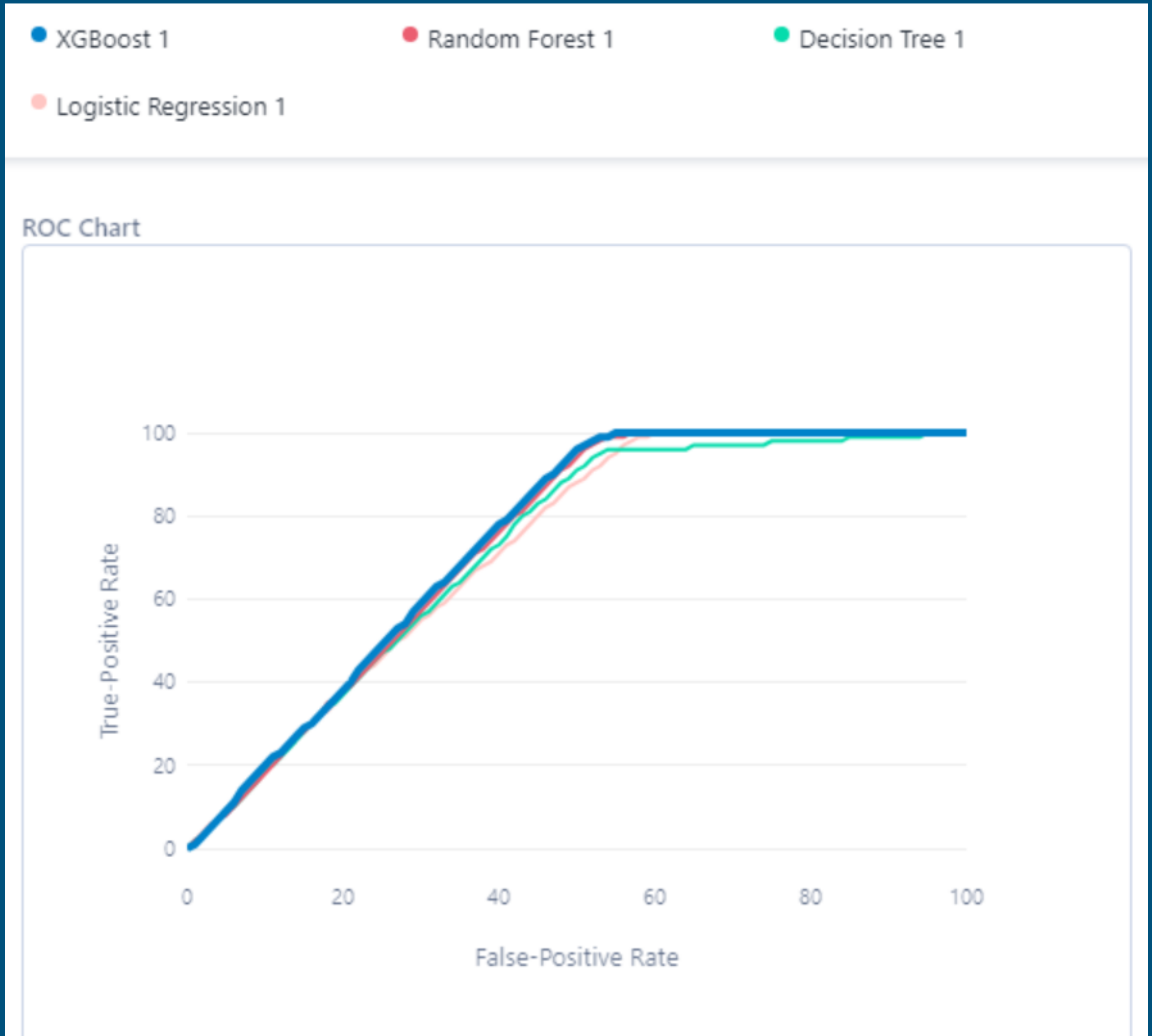
Metric Comparison – AUC

We notice that the classification tree and XG Boost gives better predictions on the basis of AUC compared to all models.

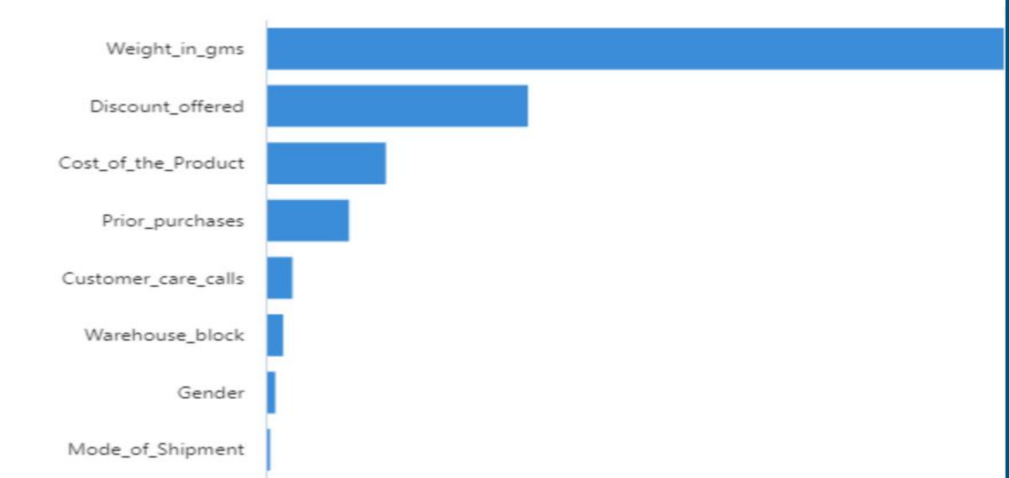


Models	AUC
Logistics	0.63
KNN K=12	0.66
KNN K=54	0.70
Classification Tree	0.72
XG Boost	0.72
Random Forest	0.69

Alteryx Assistive modeling



Model	Accuracy ↓	Balanced Accuracy	Log Loss	AUC
XGBoost 1	67.6%	70.8%	0.5	0.74
Random Forest 1	65.6%	64.9%	0.54	0.74
Decision Tree 1	64.7%	65.5%	1.76	0.71
Logistic Regression 1	63.6%	62.5%	0.55	0.72



Conclusions

- In order to predict and classify our data, we have investigated a number of machine learning models, including Logistics, KNN, Classification Tree, Random Forest, XG-Boost.
- Each model's performance was assessed, and we discovered that the Classification Tree and XG Boost delivered the most accurate outcomes and was able to give high Sensitivity thereby fulfilling the goal. In our investigation, its capability to handle non-linear connections and interactions between variables was beneficial.

	Logistic Regression	kNN k=12	kNN k=80	Classification Tree	XG Boost	Random Forest
Accuracy %	64.41	66.36	67.05	67.64	65.54	67.13
Sensitivity (Recall)	58.29	66.09	84.89	93.57	92.63	80.08
AUC	0.63	0.66	0.70	0.72	0.72	0.69

Quote

All models are wrong but some are useful !

- *George E.P. BOX*

So I propose Business to implement XG boost model to work on orders that are more likely to be delivered late prior to shipping using the predictions and increase Customer satisfaction

Future Actions

- Moving forward, I plan to conduct further analysis to refine the models. Exploring additional features, integrating real-time data, and incorporating customer feedback will be crucial in improving the accuracy and reliability of the predictions. Getting more data might also result in increasing the Specificity and thereby increasing the overall accuracy of the model making them more reliable.

- With more time and resources, I would explore advanced modeling techniques and collaborate closely with operational teams for deeper insights.

THANK YOU

QUESTIONS ?