Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Science and Engineering (Data Science)

Report on Mini Project
Machine Learning -I (DJ19DSC402)
AY: 2022-23

E-commerce Business Customer Segmentation, Churn Classification and Sales Prediction

NAME: Yash Thakar

SAP ID: 60009210205

Guided By: Dr. Kriti Srivastava

CHAPTER 1: INTRODUCTION

E-commerce has become a highly competitive business and Machine Learning can enable customer segmentation and sales prediction, enhance customer experience, predict sales, cluster customers to better understand consumer behaviour and drive revenue growth.

The objective is to develop a robust framework for segmenting customers based on their behaviour, preferences, and purchase history, and utilize this segmentation to accurately predict sales trends and patterns.

The dataset at hand is highly suitable for solving this problem as it comprises a comprehensive range of variables and features that are relevant to e-commerce customer behaviour and sales patterns.

It includes information on customer purchase history, product attributes, and sales data. Additionally, the dataset is large and diverse, providing a significant sample size for building and validating robust predictive models. With the help of this dataset, one can develop accurate customer segmentation and sales prediction models.

CHAPTER 2: DATA DESCRIPTION

The data relates to a Brazilian e-commerce public dataset of orders made at “Olist Store”. The aforementioned dataset possesses the potential to function as a representative sample for any e-commerce establishment, serving as a valuable tool for generating sophisticated analytical frameworks and models, which enable accurate prediction of KPIs and metrics that are essential for the seamless operation of an e-commerce enterprise. The models created from this can help with customer segmentation to predicting sales or predicting customer churn and satisfaction.

Domain: E-commerce

Important Datasets and Attributes from the same are:

Customers Dataset

This dataset has information about the customer and its location. Use it to identify unique customers in the orders dImportant Datasets and Attributes from the same are:

Customer_id: key to the orders dataset. Each order has a unique customer_id.
Customer_zip_code_prefix: first five digits of customer zip code
Customer_city: customer city name

Order Dataset
Important to link orders with customer, important attributes in the dataset
Order_id: order unique identifier
Customer_id: key to the customer dataset. Each order has a unique customer_id
Other attributes include order status and delivery informationomer_state: customer state

Order Items Dataset
This dataset includes data about the items purchased within each order.
Order_id: order unique identifier
Order_item_id: sequential number identifying number of items included in the same order.
Product_id: product unique identifier
Seller_id: seller unique identifier
Shipping_limit_date: Shows the seller shipping limit date for handling the order over to the logistic partner.
Price: item price
Freight_value: item freight value item

Products Dataset

This dataset includes data about the products sold by Olist.

Product_id: unique product identifier

Product_category_name: root category of product, in Portuguese. (Translation is a available in a separate csv)

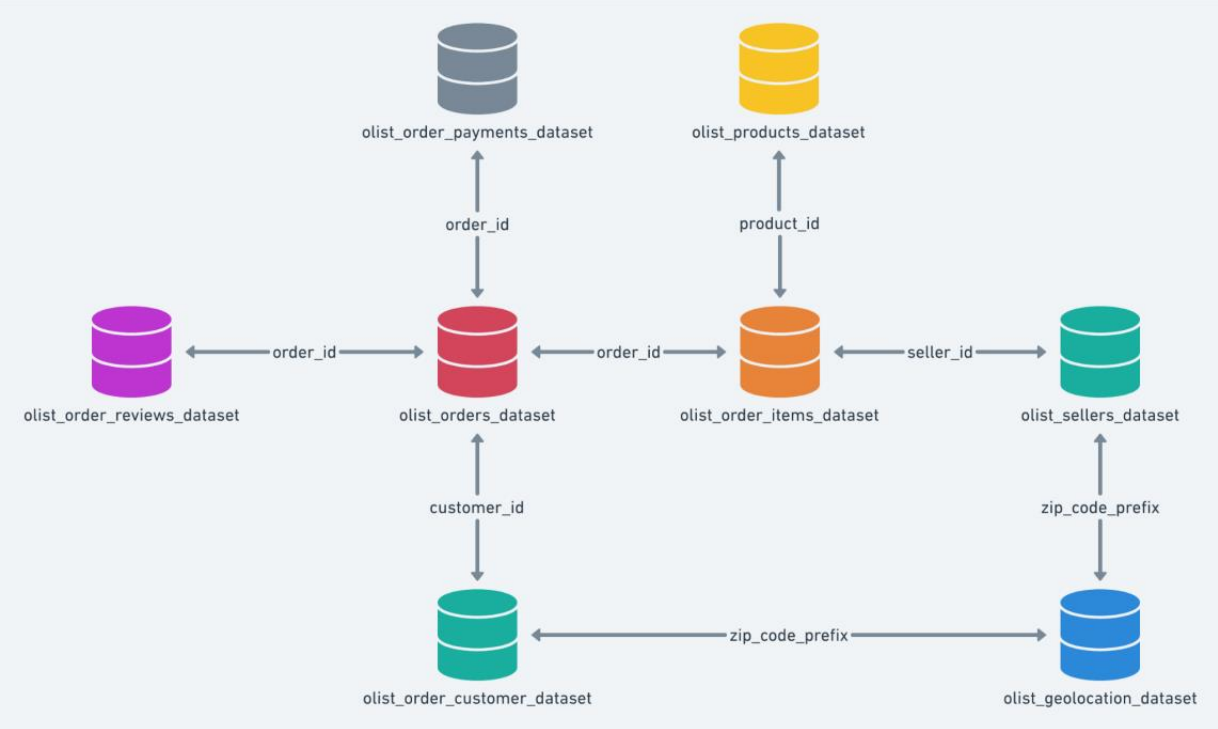
Product_weight_g: product weight measured in grams.

Product_length_cm: product length measured in centimeters.

Product_height_cm: product height measured in centimeters.

Product_width_cm: product width measured in centimeters.

Data Schema:

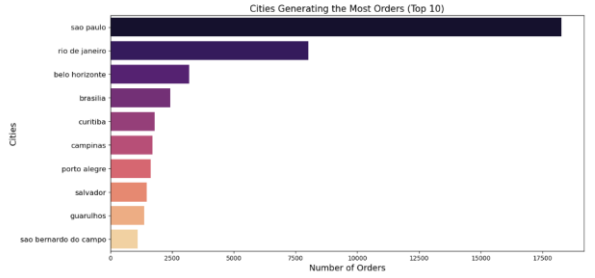


CHAPTER 3: DATA ANALYSIS

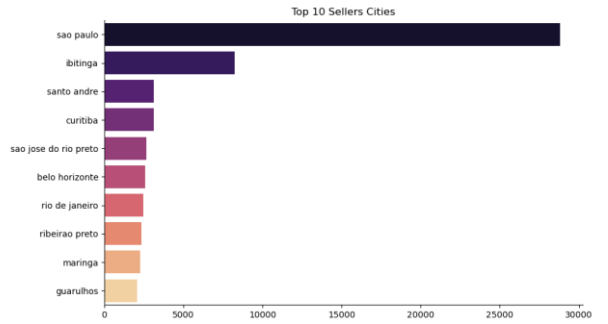
Knowing and understanding your major markets can be a huge deal for e-commerce businesses and thus we

plot the top 10 cities that are generating the highest orders

```
top_orders_cities = df.groupby("customer_city")["order_id"].count().reset_index().sort_values("order_id", ascending = False)
plt.figure(figsize = (14, 7))
sns.barplot(x = "order_id", y="customer_city", data = top_orders_cities[:10], palette = 'magma')
plt.xlabel("Number of Orders", fontsize = 14)
plt.ylabel("Cities", fontsize = 14)
plt.yticks(fontsize = 12)
plt.title("Cities Generating the Most Orders (Top 10)", fontsize = 15)
plt.show()
```



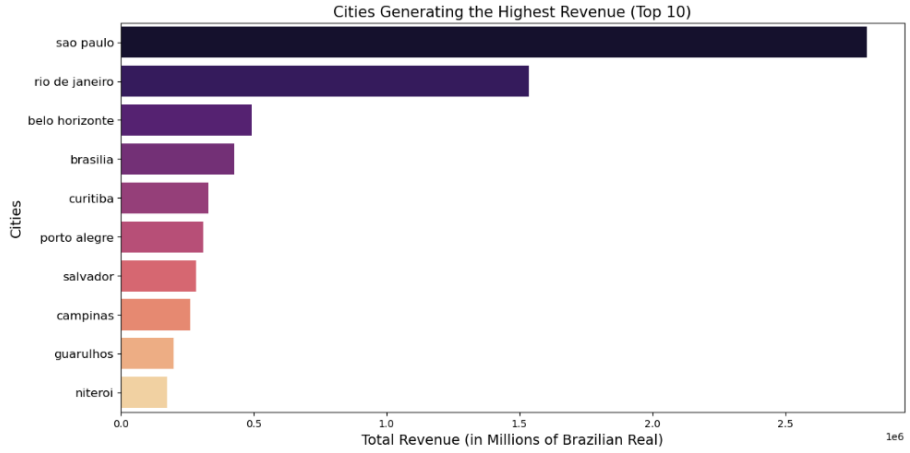

```
plt.figure(figsize=[10, 6])
sns.barplot(x = df.seller_city.value_counts().values[:10], y= df.seller_city.value_counts().index[:10], palette= 'magma')
plt.title('Top 10 Sellers Cities')
sns.despine()
```



We can clearly see that most of the orders are coming from Brazil's biggest metropolitan cities - Sao Paulo and Rio de Janerio, while most sellers are located in sao paulo and ibitinga, this can be a major insight to improve logistical operations.

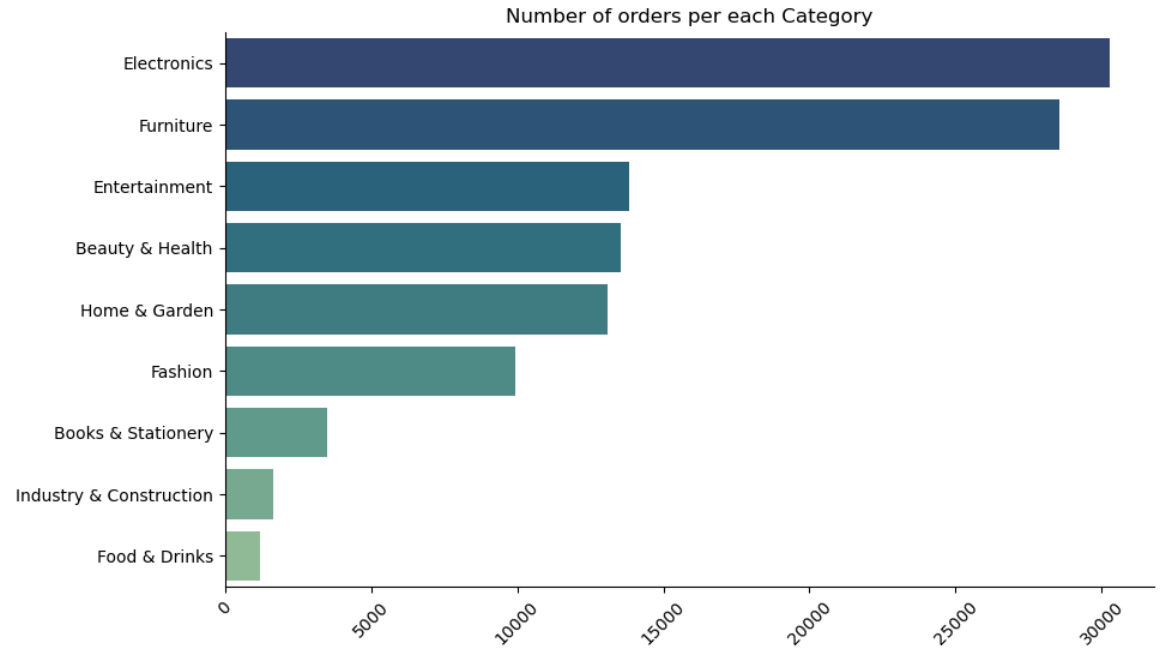
Similarly lets analyse the cities that generate most revenue

```
top_revenue_cities = df.groupby("customer_city")["payment_value"].sum().reset_index().sort_values("payment_value", ascending = Fa
plt.figure(figsize = (14, 7))
sns.barplot(x = "payment_value", y = "customer_city", data = top_revenue_cities[:10], palette = 'magma')
plt.xlabel("Total Revenue (in Millions of Brazilian Real)", fontsize = 14)
plt.ylabel("Cities", fontsize = 14)
plt.yticks(fontsize = 12)
plt.title("Cities Generating the Highest Revenue (Top 10)", fontsize = 15)
plt.show()
```



But since here many categories can be grouped into one major category to simplify our understanding we do so manually.

```
]: plt.figure(figsize=[10, 6])
   sns.barplot(x = df.product_category.value_counts().values, y = df.product_category.value_counts().index, palette= 'crest_r')
   plt.title('Number of orders per each Category')
   plt.xticks(rotation = 45)
   sns.despine()
```



Now we can observe in the below bar chart that the Electronics is the best performing category followed by Furniture while Foods and Drinks have the least amount of orders on the E-commerce platform.

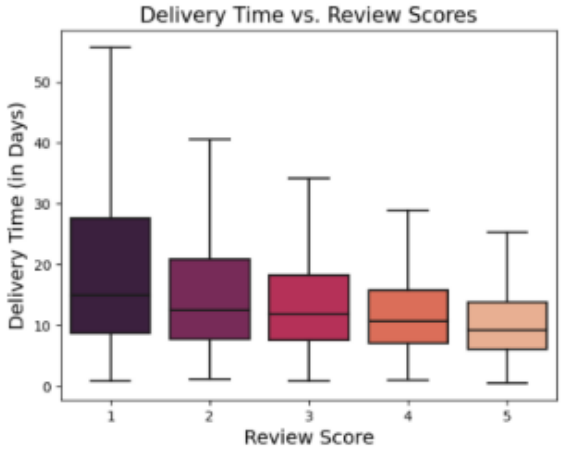
While we are on the matter of delivery lets take into account reviews and Delivery times, as these are some quantitative factors that affect customer satisfaction:

```
In [24]: # Distribution of review scores
plt.figure(figsize=(8, 6))
sns.countplot(x='review_score', data=df)
plt.title('Distribution of Review Scores', fontsize=16)
plt.xlabel('Review Score', fontsize=14)
plt.ylabel('Count', fontsize=14)
plt.tick_params(axis='both', which='major', labelsize=12)
sns.despine()
plt.show()
```

We can observe that customers are mostly satisfied as there are large number of 4 and 5 Star reviews

Now lets take a look at how Delivery time affect Review Scores:

```
In [33]: sns.boxplot(x = "review_score", y = "deliveryTime_Days", data = df, showfliers = False, palette = 'rocket')
         plt.xlabel("Review Score", fontsize = 14)
         plt.ylabel("Delivery Time (in Days)", fontsize = 14)
         plt.title("Delivery Time vs. Review Scores", fontsize = 15)
         plt.show()
```



As observed in the above Box Plot, there's a slight correlation between delivery times and review scores. The longer it takes for an order to be delivered, the more likely it is to receive a low review score.

To measure Delivery times,

```
In [28]: deliveryTime = (df["order_delivered_customer_date"] - df["order_purchase_timestamp"])
         deliveryTime_Seconds = deliveryTime.apply(lambda x: x.total_seconds())
         df['deliveryTime_Days'] = round(deliveryTime_Seconds/86400, 2)

In [29]: df['deliveryTime_Days'].describe()

Out[29]: count    113209.000000
         mean         12.442129
         std           9.356006
         min           0.530000
         25%           6.740000
         50%          10.190000
         75%          15.580000
         max         208.350000
         Name: deliveryTime_Days, dtype: float64

In [30]: sns.boxplot(df.deliveryTime_Days, orient = 'h', showfliers = True)
         plt.xlabel("Number of Days", fontsize = 14)
         plt.yticks([])
         plt.title('Distribution of Delivery Time Across All Orders', fontsize = 15)
         plt.show()
```

We can some to the conclusion from the Box and Whisker Plot, that on an average a delivery takes 12 days to reach the customer, which is significantly slower by modern standards.

We can also observe a huge amount of Outliers in the Box Plot pointing towards inconsistent Delivery Times

In [78]:
```
# Contingency table of order status vs. review score
cont_table = pd.crosstab(df['order_status'], df['review_score'])
plt.figure(figsize=(8, 6))
sns.heatmap(cont_table, cmap='Blues', annot=True, fmt='d')
plt.title('Order Status vs. Review Score', fontsize=16)
plt.xlabel('Review Score', fontsize=14)
plt.ylabel('Order Status', fontsize=14)
plt.tick_params(axis='both', which='major', labelsize=12)
sns.despine()
plt.show()
```

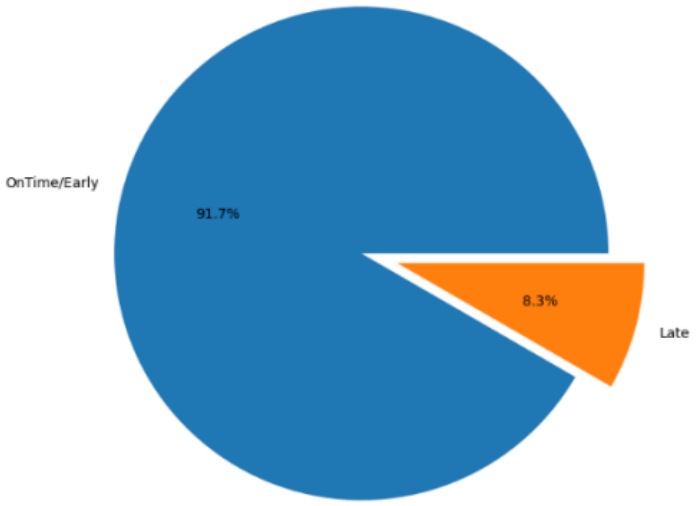
We can observe that the proportion of negative reviews for a cancelled order or a order that shows shipped/invoiced/processing as its order status is significantly high and is a sign of unsatisfied customers.

On classifying the delivers into On-time and Late we can observe that 91.7% of deliveries only make it on time, which is a number that needs improvement to improve customer retention:

In [34]:
```
df['seller_to_carrier_status'] = (df['shipping_limit_date'].dt.date - df['order_delivered_carrier_date'].dt.date).dt.days

# Now calssify the duration into 'OnTime/Early' & 'Late'
df['seller_to_carrier_status'] = df['seller_to_carrier_status'].apply(lambda x : 'OnTime/Early' if x >=0 else 'Late')
```

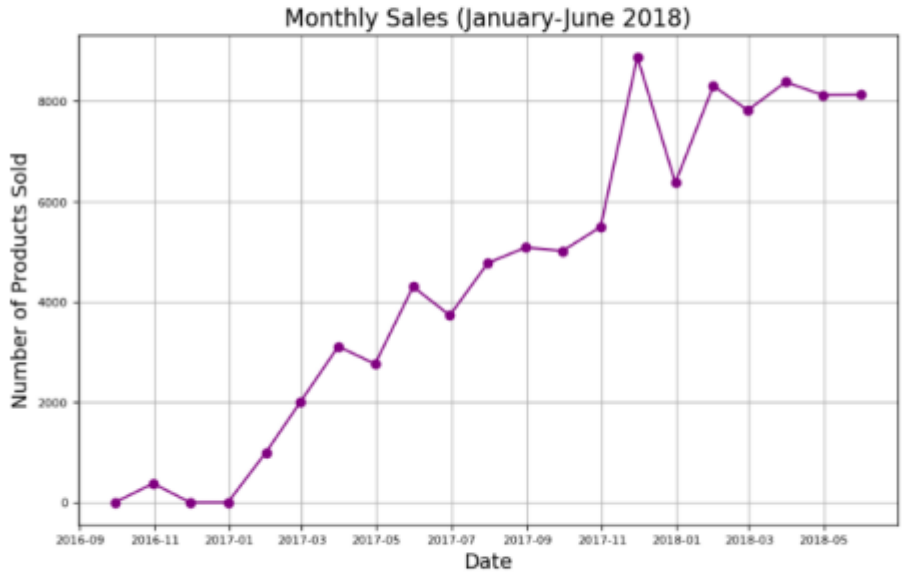
In [35]:
```
# First get difference between estimated delivery date and actual delivery date in days
df['arrival_status'] = (df['order_estimated_delivery_date'].dt.date - df['order_delivered_customer_date'].dt.date).dt.days

# Now Classify the duration in 'OnTime/Early' & 'Late'
df['arrival_status'] = df['arrival_status'].apply(lambda x : 'OnTime/Early' if x >=0 else 'Late')
```

In [36]:
```
plt.figure(figsize=[30,8])
Values = df.arrival_status.value_counts().values
Labels = df.arrival_status.value_counts().index
plt.pie(Values, explode=(0.05, 0.1), labels= ['OnTime/Early', 'Late'], autopct='%1.1f%%')
plt.show()
```



Analysing Revenue and Sales:

```
In [40]: monthly_sales = df.resample('M', on='order_purchase_timestamp')['order_item_id'].count()

         monthly_sales = monthly_sales['2016-01':'2018-05']


         plt.figure(figsize=(10,6))
         plt.plot(monthly_sales.index, monthly_sales.values, '-o', color='purple')

         plt.title('Sales', fontsize=16)
         plt.xlabel('Date', fontsize=14)
         plt.ylabel('Number of Products Sold', fontsize=14)

         plt.grid(True)
```

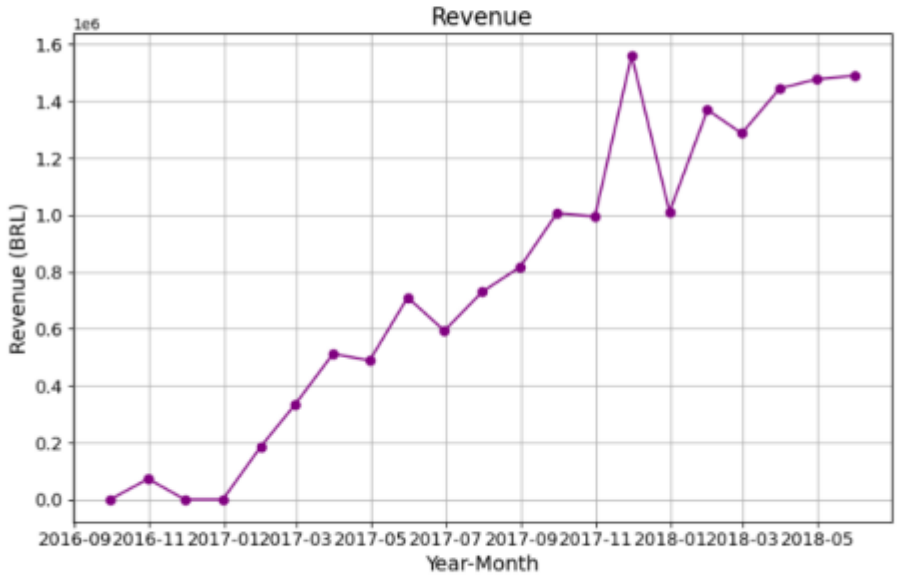


There was a huge growth in the sales from 2016 to 2017,hitting its peak in the end of 2017 after which the sales fell down and became inconsistent.

With a major hit in sales observed in the start of a 2018 right after the company made record sales.

A similar trend can be observed in the Revenue:

```
In [50]: monthly_revenue = df['payment_value'].resample('M').sum()

         monthly_revenue = monthly_revenue['2016-01':'2018-05']


         plt.figure(figsize=(10,6))
         plt.plot(monthly_revenue.index, monthly_revenue.values, '-o', color='purple')

         plt.title('Revenue', fontsize=16)
         plt.xlabel('Year-Month', fontsize=14)
         plt.ylabel('Revenue (BRL)', fontsize=14)

         plt.grid(True)
         plt.show()
```



Although Revenue is constantly growing after the dip in the beginning of 2018/04 to 2018/06 in spite of a sales being a little low throughout this period, this might indicate organizational efficiency and improved margins in the business.

CHAPTER 4: REASON TO SELECT MACHINE LEARNING MODEL

XGBoost (eXtreme Gradient Boosting) is a powerful machine learning algorithm that is widely used for regression and classification tasks. It is an optimized implementation of gradient boosting that has proven to be highly effective in a variety of data science competitions and real-world applications.

The main idea behind XGBoost is to iteratively train weak learners (decision trees) on the residual errors of the previous iteration, with the goal of minimizing a loss function. During each iteration, XGBoost assigns weights to the data points based on their error, so that the next weak learner focuses more on the harder-to-predict cases.

XGBoost provides several hyperparameters that allow you to fine-tune the model to your specific problem. Some of the most important hyperparameters include the learning rate, the number of trees, the maximum depth of each tree, and the regularization parameters.

One of the main advantages of XGBoost is its scalability and speed. It can handle large datasets with millions of rows and thousands of features, and it can be parallelized across multiple CPU cores and even across multiple machines.

Overall, XGBoost is a powerful tool that can be used for a wide range of machine learning tasks, including regression, classification, and ranking problems. However, it requires some tuning and parameter optimization to achieve optimal performance, and it may not be the best choice for very highdimensional or sparse datasets.

The XGBoost Model was selected for both Classification of Customer Churn Data and Regression for Sales Predicition.

XGBoost was selected for the classification task because:

XGBoost is also Robust to outliers, highly accurate and XGBoost includes regularization techniques that help to prevent overfitting, which is could be a problem in Churn Classification Problems. It uses a combination of L1 and L2 regularization to reduce the complexity of the model, resulting in more robust and accurate predictions

XGBoost was selected for the Regression task(Sales Prediction) because:

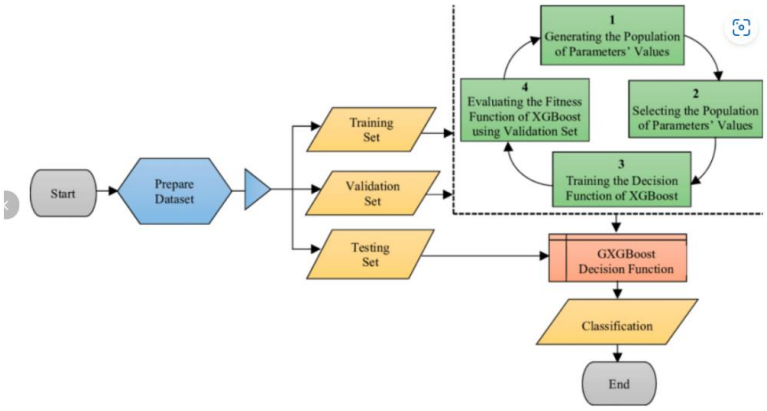
Applying Tree based Regressors as they have high explainability thus you can explain the decisions, identify possible events that might occur, and see potential outcomes. The analysis helps you determine what the best decision would be. High explainability can also be a huge advantage when working with models that will be making business decisions so as to better understand how and why certain decisions are made

XGBoost Regressor has the highest value of R-sqauare score and has the lowest value of Mean Absolute Error and Mean Squared Error This is possibly because, XGBoost is a Boosting method that uses several weak learners and improves on the weak learners in every iteration to improve its results.

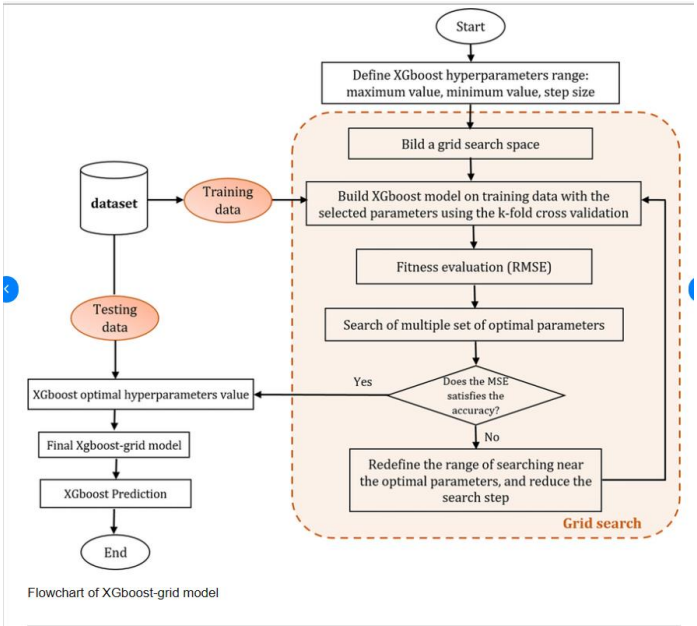
XGBoost is also Robust to outliers, highly accurate and XGBoost includes regularization techniques that help to prevent overfitting, which is could be a problem in Churn Classification Problems. It uses a combination of L1 and L2 regularization to reduce the complexity of the model, resulting in more robust and accurate predictions

CHAPTER 5: ALGORITHM

XGBoost Classifier: (used for modelling churn using Classification)



XGBoost Regressor: (used for Predicting Sales using Regression)



Flowchart of XGboost-grid model

CHAPTER 6: RESULT ANALYSIS

CHURN MODELLING (CLASSIFICATION PROBLEM)

Model	Accuracy	AUC of ROC
Logistic Regression	0.5946	0.63
Random Forest (Bagging)	0.7332	0.82
AdaBoost (Boosting)	0.5946	0.73
XGBoost (Boosting)	0.7574	0.85

From the above table we can conclude that the XGBoost Classification Model is the best choice as it has the highest accuracy score and the highest AUC of ROC . XGBoost is also Robust to outliers, highly accurate and XGBoost includes regularization techniques that help to prevent overfitting, which is could be a problem in Churn Classification Problems. It uses a combination of L1 and L2 regularization to reduce the complexity of the model, resulting in more robust and accurate predictions.

XGBoost

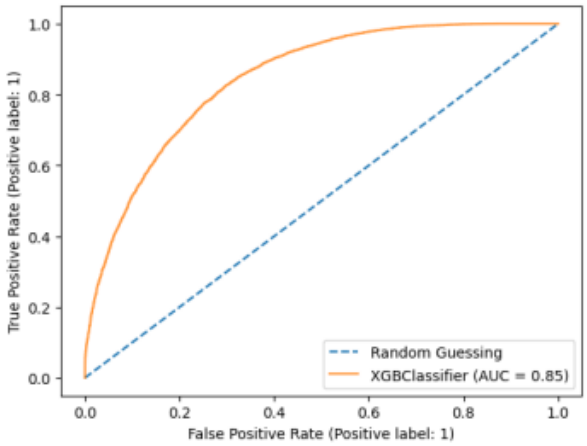
In [114]:
```
from sklearn.metrics import accuracy_score
from xgboost import XGBClassifier
model= XGBClassifier(n_estimators = 50,random_state=0)
model.fit(X_train,y_train)

# Predict on the test set and calculate accuracy
y_pred_xg = model.predict(X_test)
```

In [115]:
```
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import recall_score
cm_xg = confusion_matrix(y_test, y_pred_xg)
print('Confusion Matrix:\n', cm_xg)
print('Classification Report:\n', classification_report(y_test, y_pred_xg))
xg_score=model.score(X_test , y_test)
print(xg_score)
```

```
Confusion Matrix:
 [[7809 2292]
 [2149 6044]]
Classification Report:
              precision    recall  f1-score   support

           0       0.78      0.77      0.78     10101
           1       0.73      0.74      0.73      8193

    accuracy                           0.76     18294
   macro avg       0.75      0.76      0.75     18294
weighted avg       0.76      0.76      0.76     18294

0.75724281185088
```

ROC Curve:

In [116]:
```
from sklearn.metrics import RocCurveDisplay
ax = plt.gca()
plt.plot([0, 1], [0, 1], linestyle='--', label='Random Guessing')
xg_disp = RocCurveDisplay.from_estimator(model, X_test, y_test, ax=ax, alpha=0.8)
plt.show()
```



SALES PREDICTION (REGRESSION PROBLEM)

Model	R-squared value	MAE	MSE
Linear Regression	0.00542	39.14	68.93
DecisionTree Regression	0.8925	9.41	20.31
RandomForestRegressor	0.9136	9.02	20.32
XGBoostRegressor	0.9459	10.26	16.06

Applying Tree based Regressors as they have high explainability thus you can explain the decisions, identify possible events that might occur, and see potential outcomes. The analysis helps you determine what the best decision would be. High explainability can also be a huge advantage when working with models that will be making business decisions so as to better understand how and why certain decisions are made.

From the Above table we can observe that the XGBoost Regressor has the highest value of R-sqauare score and has the lowest value of Mean Absolute Error and Mean Squared Error
This is possibly because, XGBoost is a Boosting method that uses several weak learners and improves on the weak learners in every iteration to improve its results.

XGBoostRegressor

```
In [104]: from xgboost import XGBRegressor

In [109]: xgb=XGBRegressor(n_estimators=100,random_state=42, learning_rate=0.05)

In [110]: xgb.fit(x_train,y_train)

Out[110]: XGBRegressor(base_score=None, booster=None, callbacks=None,
                       colsample_bylevel=None, colsample_bynode=None,
                       colsample_bytree=None, early_stopping_rounds=None,
                       enable_categorical=False, eval_metric=None, feature_types=None,
                       gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
                       interaction_constraints=None, learning_rate=0.05, max_bin=None,
                       max_cat_threshold=None, max_cat_to_onehot=None,
                       max_delta_step=None, max_depth=None, max_leaves=None,
                       min_child_weight=None, missing=nan, monotone_constraints=None,
                       n_estimators=100, n_jobs=None, num_parallel_tree=None,
                       predictor=None, random_state=42, ...)

In [111]: y_pred_xgb = xgb.predict(x_test)

In [112]: print("R squared value",metrics.r2_score(y_test,y_pred_xgb))
          print("MAE",metrics.mean_absolute_error(y_test,y_pred_xgb))
          print("MSE",np.sqrt(metrics.mean_squared_error(y_test,y_pred_xgb)))

          R squared value 0.9459609561509599
          MAE 10.262075330105832
          MSE 16.068519977477678
```

CHAPTER 7: CONCLUSION AND FUTURE SCOPE

The Data Analysis and Modelling was done on the Brazilian ecommerce public dataset of orders made at Olist Store. The dataset has information of 100k orders from 2016 to 2018 made at multiple marketplaces in Brazil. Its features allows viewing an order from multiple dimensions: from order status, price, payment and freight performance to customer location, product attributes and finally reviews written by customers. We also released a geolocation dataset that relates Brazilian zip codes to lat/lng coordinates.

This is real commercial data, it has been anonymised, and references to the companies and partners in the review text have been replaced.

The Project worked on two problems:

- Customer Segmentation and Churn Analysis
- Sales Prediction

For Customer Segmentation we applied RFM (Recency-Frequency-Monetary) analysis by making calculating columns for RFM from customer data available. K-means algorithm was then applied to the RFM dataset obtained to group customers into clusters these clusters can then be analysed from the cluster centres can then be used to gain insights into each cluster from their RFM values.

Once the customers were segmented, we used the recency values to label whether each customer was churned or not. Once the labels were obtained we dropped the RFM values column and classified the customers based on customer data whether or not the customer had churned or not.

Churn rates are important because losing customers means losing revenue. So, the bottom line is that high churn could affect your bottom line.

Another reason it's critical to improve customer retention and reduce churn is that it's generally more expensive to find new customers than it is to keep existing ones.

For this classification problem XGBoost Classification Model is the best choice as it has the highest accuracy score and the highest AUC of ROC . XGBoost is also Robust to outliers, highly accurate and XGBoost includes regularization techniques that help to prevent overfitting, which is could be a problem in Churn Classification Problems. It uses a combination of L1 and L2 regularization to reduce the complexity of the model, resulting in more robust and accurate predictions. 75.74% accuracy was achieved.

For sales prediction we used the available product and order data available to extract important features that were not directly present but could be calculated, e.g. sales_amt, avg_review_score, avg_price, estimated_delivery, product_category. This data was then analysed to model the sales of each product on the store, this can useful for the store to analyse which product to push and which not to. This can also be useful as

- A sales forecast helps every business make better business decisions. It helps in overall business planning, budgeting, and risk management.
- Sales forecasting allows companies to efficiently allocate resources for future growth and manage its cash flow.
- Sales forecasting also helps businesses to estimate their costs and revenue accurately based on which they are able to predict their short-term and long-term performance.

For this Regression problem, Tree based Regressors were selected as they have high explainability thus you can explain the decisions, identify possible events that might occur, and see potential outcomes. The analysis helps you determine what the best decision would be. High explainability can also be a huge advantage when working with models that will be making business decisions so as to better understand how and why certain decisions are made.

From the Result analysis we can observe that the XGBoost Regressor has the highest value of R-sqauare score and has the lowest value of Mean Absolute Error and Mean Squared Error This is possibly because, XGBoost is a Boosting method that uses several weak learners and improves on the weak learners in every iteration to improve its results. A R-squared score of 0.9459 was achieved.

The future scope of this project is significant. Upon completion of this project I was able to analyse the customer retention with our Churn classification problem and I was able to forecast sales of a product in the store. Resulting in increased insights for the e-commerce store. The project allowed us to collect valuable customer data, which can be used to improve our products and personalize the stores marketing and sales efforts.
Beyond the methods we used in the future we can use density-base clustering methods to improve on the clusters obtained and find nested clusters in the customer data.
Also forecasting can be done by Time-series analysis, LSTM and other deep learning methods to gain better understanding of the data and improve our results.
Also further more customer satisfaction can be predicted through reviews data available.

PYTHON NOTEBOOKS:

EDA:

https://colab.research.google.com/drive/1XPme7jyd-Iy_GtXfdAjB1hYkZQKiwYPQ?usp=sharing

CHURN MODELLING (Classification):

https://colab.research.google.com/drive/1e4k24Q2WofuHN8ykUimO9jBuBHrqbgd8?usp=sharing

SALES PREDICTION (Regression):

https://colab.research.google.com/drive/1FtT0cTZpG4SUipMQbaPrn1kjfS8wiq02?usp=sharing