CUSTOMER PROFILING

IE 7275 - Data Mining (Spring 2023)
Prof. Sagar Kamarthi
Group 63

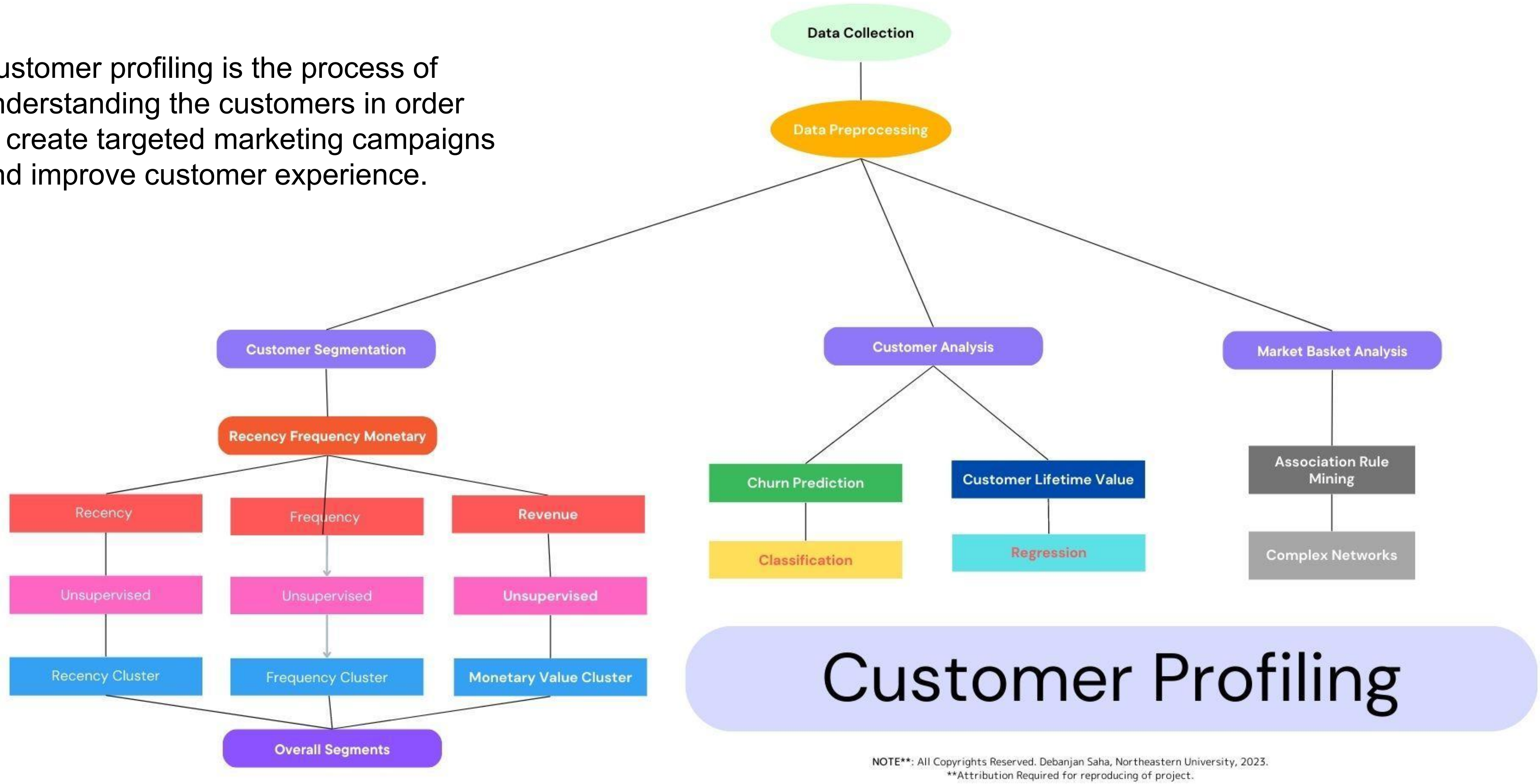
Debanjan Saha & Ritika Rao

AGENDA

- Introduction
- Data Overview
- Exploratory Data Analysis
- Data Preprocessing
- Clustering
- Classification
- Regression
- Model Selection
- Model Evaluation
- Conclusion

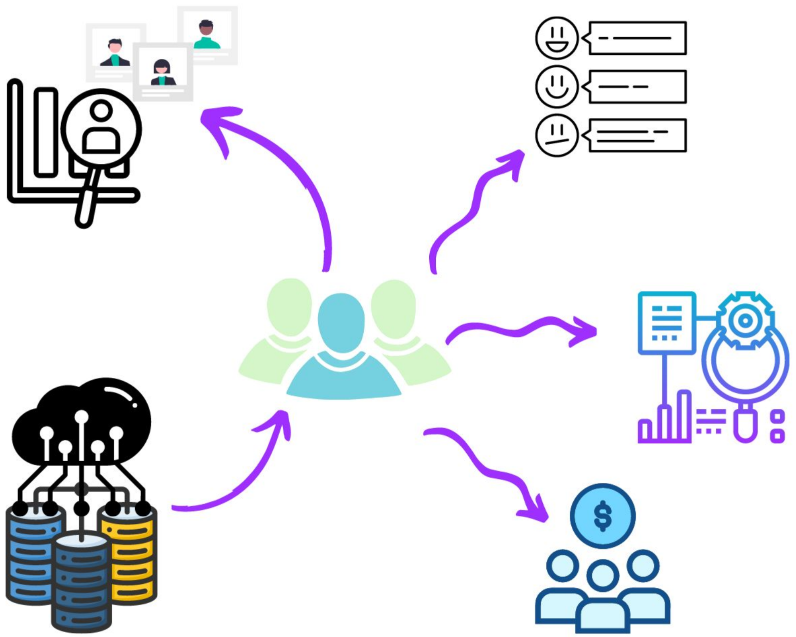
INTRODUCTION TO CUSTOMER PROFILING

Customer profiling is the process of understanding the customers in order to create targeted marketing campaigns and improve customer experience.



INTRODUCTION

- **Customer Segmentation**: Customers can be divided into various groups based on different factors.
- **Customer Lifetime Value (CLV)**: Net profit attributed to the entire future relationship with a customer
- **Customer Churn**: Customers who are likely to cancel their subscription or stop doing business with a company
- **Customer Relationship Management (CRM)**: Both CLV and churn prediction are the key elements used to inform strategic decision making in areas such as marketing, sales, and customer service



DATA OVERVIEW

Domain: Retail Sales

Data Source: [Kaggle]

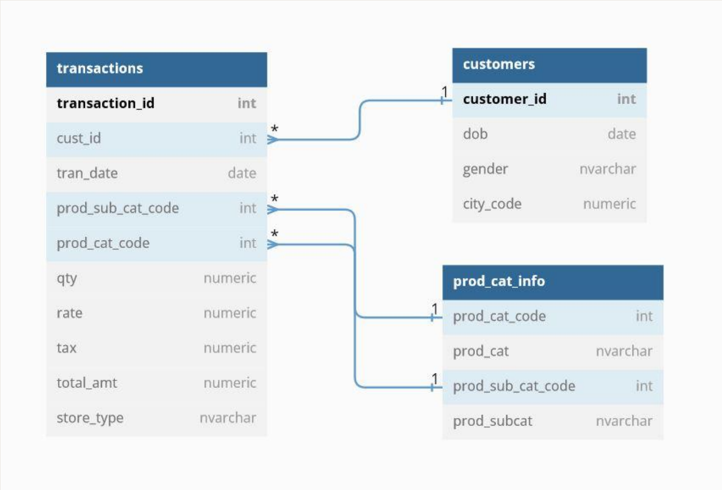
Data Dictionary

Transactions		
Attribute	Data Type	Description
transaction_id	Numeric	Identifies transactions uniquely
cust_id	Numeric	Identifies customers uniquely
tran_date	Date	Date on which transaction took place
prod_subcat_code	Numeric	Identifies the sub category of the product
prod_cat_code	Numeric	Identifies the category of the product
Qty	Numeric	Count of the product bought
Rate	Numeric	Cost of product per unit
Tax	Numeric	Tax applied on the transaction
total_amt	Numeric	Total cost of transaction
Store_type	Catgorical	Type of store receiving order(online/in-person)

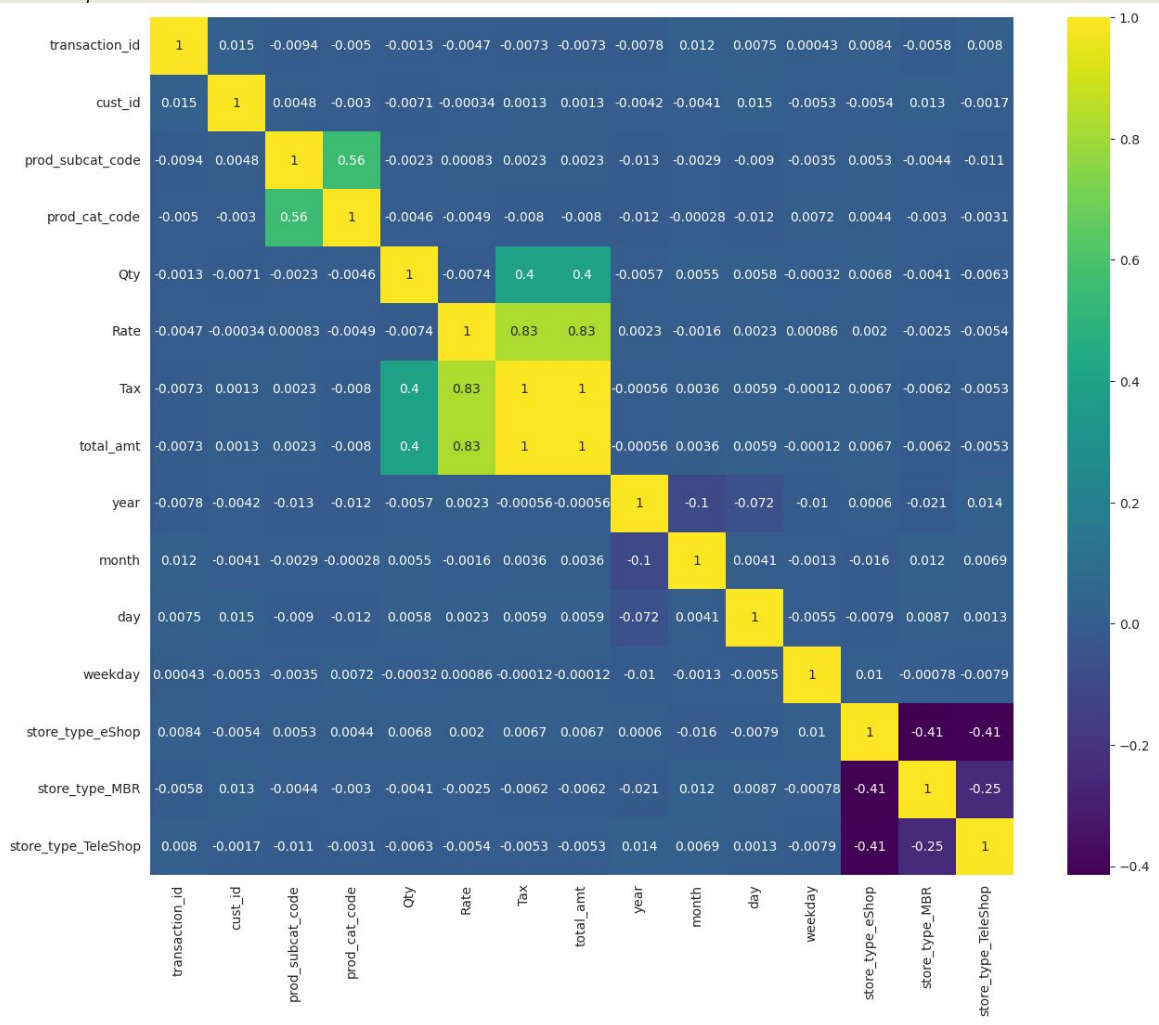
Customer		
Attribute	Data Type	Description
customer	Numeric	Identifies customers uniquely
DOB	Date	Date of birth of the customer
Gender	Categorical	Gender of the customer
city_code	Numeric	City that the customer lives in

prod_cat_info		
Attribute	Data Type	Description
prod_cat_code	Numeric	Identifies category of product
prod_cat	Categorical	Name of the category
prod_sub_cat_code	Numeric	Identifies sub category of product
prod_subcat	Categorical	Name of the sub category

Entity Relationship Diagram (ERD)



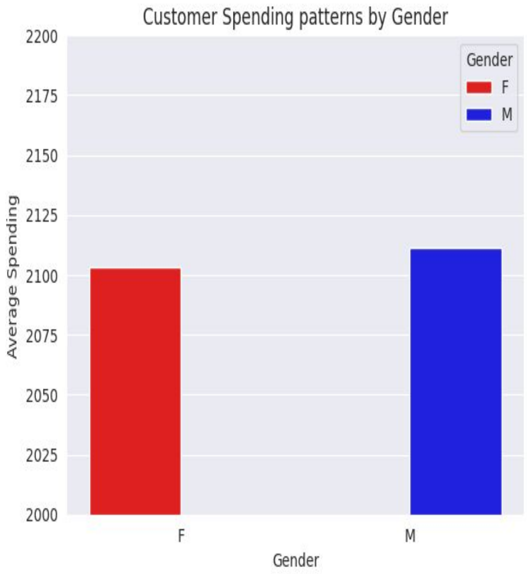
EXPLORATORY DATA ANALYSIS (EDA)



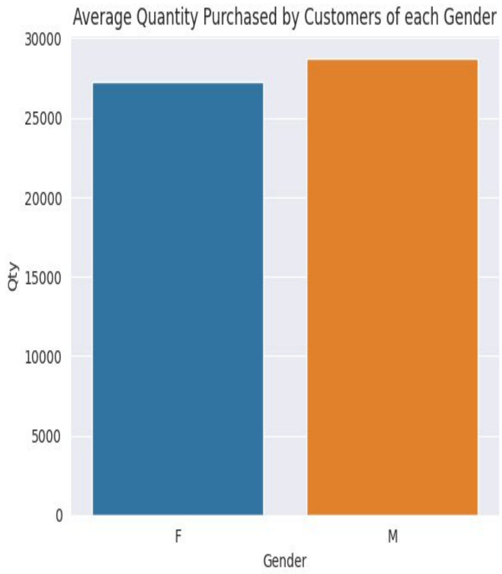
Correlation Heatmap

- The correlation heatmap indicates a strong positive correlation between *'Tax'*, *'Rate'* and *'total_amount'* as well as *'Qty'* and *'total_amount'*
- This is because *'total_amount'* can be written as : ***Qty * Rate + Tax***
- We drop *'Tax'* and *'Rate'* columns

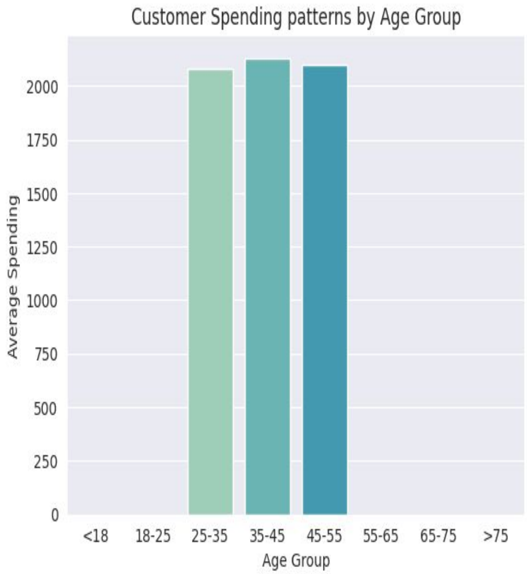
Customer Spending by Gender



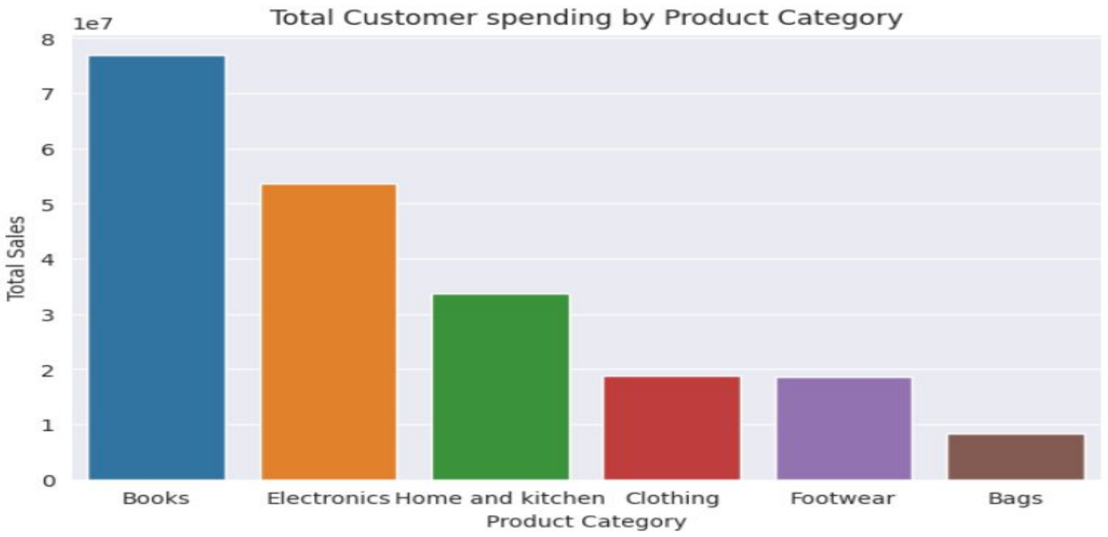
Quantity Ordered by Gender



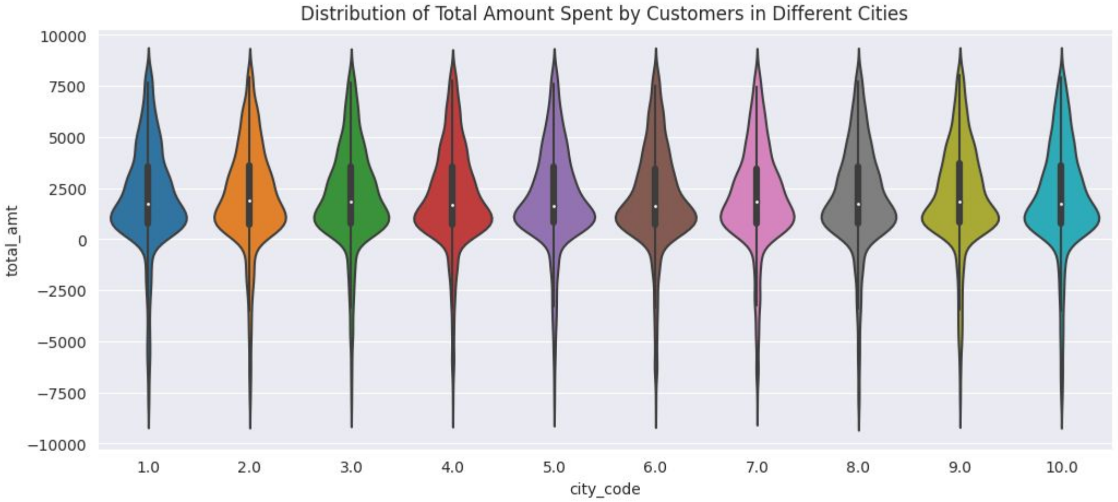
Average Spending by Age Groups



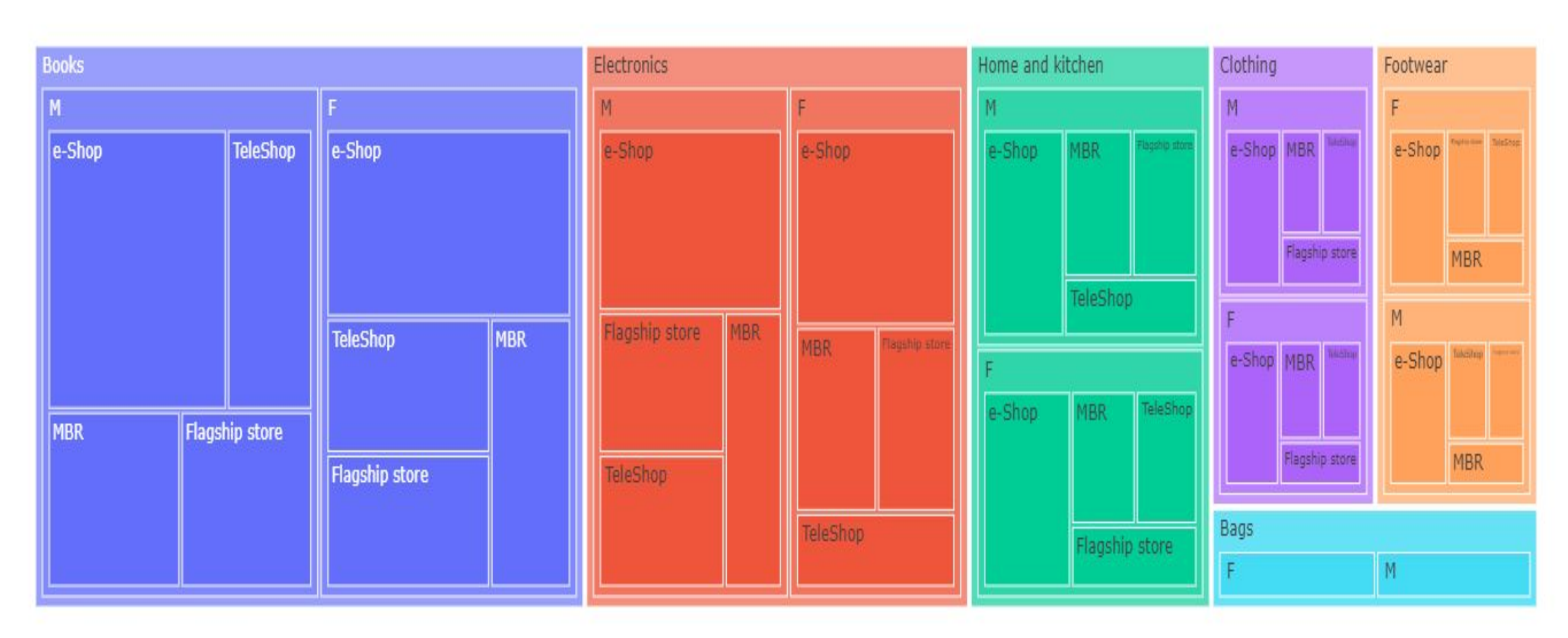
Customer Spending across Product Categories



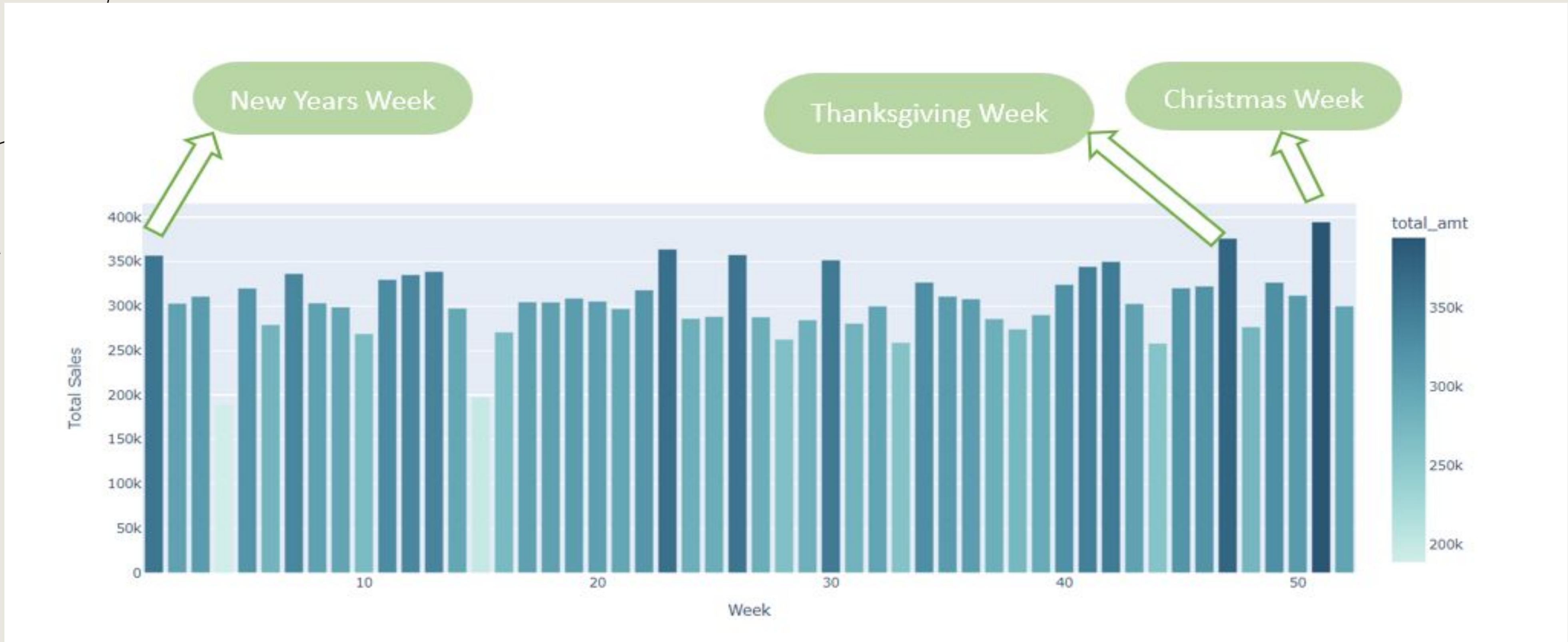
Customer Spending based on different cities

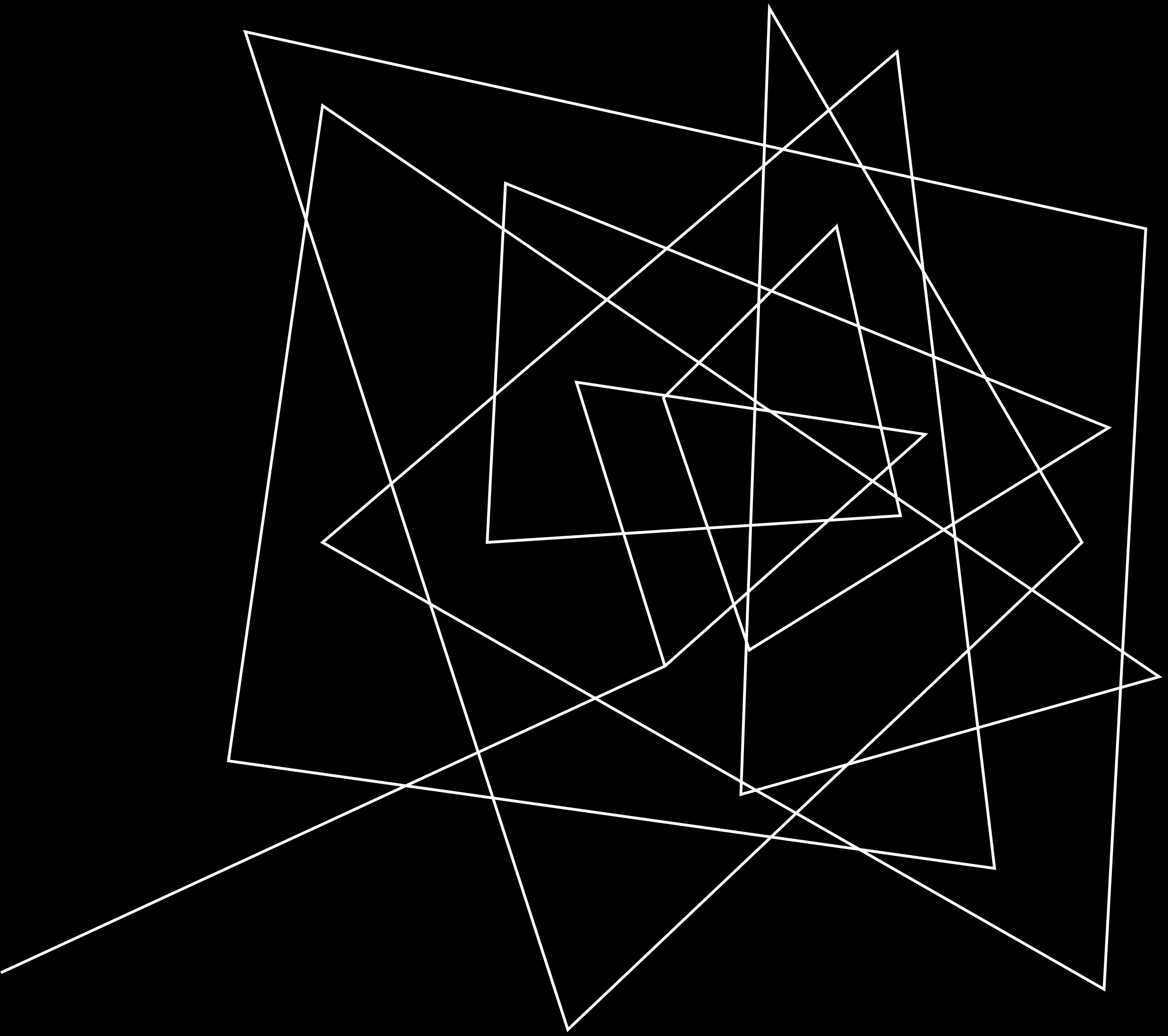


Product Category Sales Trend by Gender and Store Type



Weekly Sales Trends for the year 2012



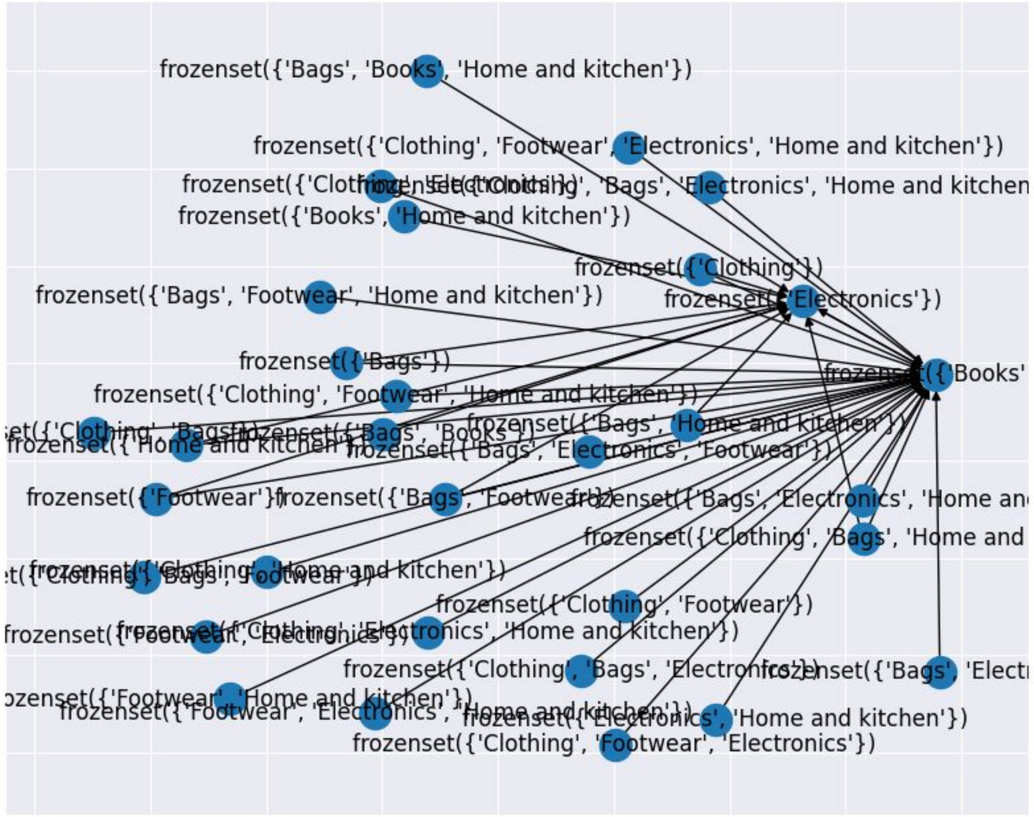


MARKET BASKET ANALYSIS

- Market basket analysis is a technique used to gain insights into customer behavior by examining the products customers tend to purchase together.
- Association rule mining is a popular method in market basket analysis that identifies relationships between items in a transactional database.
- The relationships are used to generate rules that can be used to predict future purchases and create targeted marketing campaigns.
- Complex networks can be used to visualize the relationships and identify key products that drive customer behavior.

ASSOCIATION RULE MINING

antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
(Clothing, Bags)	(Books)	0.087541	0.589720	0.052307	0.597510	1.013210	0.000682	1.019355
(Clothing, Bags, Footwear)	(Books)	0.029059	0.589720	0.017254	0.593750	1.006833	0.000117	1.009919
(Clothing, Bags, Electronics)	(Books)	0.042317	0.589720	0.024882	0.587983	0.997054	-0.000074	0.995783
(Bags, Home and kitchen)	(Electronics)	0.110243	0.528696	0.057210	0.518946	0.981558	-0.001075	0.979731
(Clothing)	(Books)	0.353433	0.589720	0.204504	0.578623	0.981182	-0.003922	0.973664
(Footwear)	(Books)	0.357065	0.589720	0.205412	0.575280	0.975513	-0.005156	0.966000
(Books, Home and kitchen)	(Electronics)	0.263712	0.528696	0.135852	0.515152	0.974381	-0.003572	0.972065
(Electronics, Home and kitchen)	(Books)	0.236469	0.589720	0.135852	0.574501	0.974192	-0.003599	0.964231
(Bags, Books, Home and kitchen)	(Electronics)	0.061751	0.528696	0.031784	0.514706	0.973539	-0.000864	0.971172
(Home and kitchen)	(Electronics)	0.459499	0.528696	0.236469	0.514625	0.973385	-0.006466	0.971009
(Home and kitchen)	(Books)	0.459499	0.589720	0.263712	0.573913	0.973195	-0.007263	0.962901
(Bags)	(Electronics)	0.251907	0.528696	0.128769	0.511175	0.966860	-0.004414	0.964157
(Bags, Footwear)	(Electronics)	0.083908	0.528696	0.042862	0.510823	0.966193	-0.001500	0.963462
(Electronics)	(Books)	0.528696	0.589720	0.301126	0.569564	0.965820	-0.010657	0.953172
(Books)	(Electronics)	0.589720	0.528696	0.301126	0.510625	0.965820	-0.010657	0.963074
(Clothing, Footwear)	(Books)	0.118416	0.589720	0.067381	0.569018	0.964895	-0.002451	0.951966



DATA PREPROCESSING

- We first formatted the datetime field to desired format & extracted features such as the year, month, day, weekday
- There was some problems with the values for ‘Qty’ and ‘Rate’ so they were negated, to follow the rule that Qty * Rate = Total Amount
- ‘store_type’ was a categorical variable which was one hot encoded to convert into numerical variable
- The data contained multiple refund records for a single transaction, which needed to be cleaned
- The demographics data contained DOB which was used to find the Age of the customers and successively binned into different Age Groups in intervals of 10s

DATA PREPROCESSING CODE SNIPPETS

Extracting features from Date column

```
# Convert tran_date to datetime
transactions['tran_date'] = pd.to_datetime(transactions['tran_date'],
                                           infer_datetime_format=True)

# Extract month from tran_date
transactions['month'] = transactions['tran_date'].dt.month

# Extract week from tran_date
transactions['week'] = transactions['tran_date'].dt.isocalendar().week

#Extract week and day of week from tran_date
transactions['day_of_week'] = transactions['tran_date'].dt.dayofweek
```

Removing multiple transactions for returns or transactions which did not go through like Credit Card Declined, etc.

```
# Find the rows where neg_count > 1
mult_negs = unq_dups.loc[unq_dups['neg_count'] > 1, :].index

# Filter the rows to keep only the first negative total_amt for each affected transaction
rows_to_drop = pd.concat([
    pd.Series(df_neg.iloc[1:].index) for (cust_id, transaction_id), df_neg in
    txn.loc[txn['total_amt'] < 0].groupby(['cust_id', 'transaction_id']) if len(df_neg) > 1
]).reset_index(drop=True)

print(f'Dropping {rows_to_drop.shape[0]} duplicate records')
# Drop the selected rows from the main dataframe
txn = txn.drop(index=rows_to_drop)
print('Number of Transactions After Drop = ', txn.shape[0])
```

Extracting Age and binning from Date of Birth (DOB)

```
# We need to convert the Customers DOB column to datetime format
customers['DOB'] = pd.to_datetime(customers['DOB'])

# Calculate Age of customers based on their DOB
now = pd.to_datetime('now').year
customers['Age'] = now - customers['DOB'].dt.year

# Create age group
customers['Age_Group'] = pd.cut(customers['Age'], bins=[0, 18, 25, 35, 45, 55, 65, 75, 100],
                                labels=['<18', '18-25', '25-35', '35-45', '45-55', '55-65', '65-75', '>75'])
```

FEATURE SELECTION

- We tried to select most important features from the data using recursive feature elimination with Cross Validation
- Different types of models (lasso, tree-based, boosting) predict different set of features in the order of importance

FEATURE SELECTION CODE & RESULTS

Recursive Feature Elimination Code

```python
def recursive_feature_elimination(estimator):
    # Define the recursive feature elimination object and fit on training data
    selector = RFECV(estimator, step=1, cv=5)
    selector.fit(X_train, y_train)

    # Print the ranking of each feature
    print(f"\nModel: {str(estimator)[:-2]} \nRankings:")
    ranked_features = sorted(zip(X_train.columns, selector.ranking_), key=lambda x: x[1])
    for feature in ranked_features:
        print(f"Rank: {feature[1]} \t Feature: {feature[0]}")
```

Results

Lasso Regression

```
Rankings:
Rank: 1          Feature: Qty
Rank: 2          Feature: store_type_TeleShop
Rank: 3          Feature: store_type_MBR
Rank: 4          Feature: prod_cat_code
Rank: 5          Feature: prod_subcat_code
Rank: 6          Feature: store_type_eShop
Rank: 7          Feature: weekday
Rank: 8          Feature: day
Rank: 9          Feature: month
Rank: 10         Feature: year
Rank: 11         Feature: cust_id
Rank: 12         Feature: transaction_id
```

Random Forest Regression

```
Rankings:
Rank: 1          Feature: transaction_id
Rank: 1          Feature: cust_id
Rank: 1          Feature: prod_subcat_code
Rank: 1          Feature: prod_cat_code
Rank: 1          Feature: Qty
Rank: 1          Feature: year
Rank: 1          Feature: month
Rank: 1          Feature: day
Rank: 1          Feature: weekday
Rank: 2          Feature: store_type_eShop
Rank: 3          Feature: store_type_MBR
Rank: 4          Feature: store_type_TeleShop
```

XGBoost Regression

```
Rankings:
Rank: 1          Feature: Qty
Rank: 2          Feature: cust_id
Rank: 3          Feature: weekday
Rank: 4          Feature: day
Rank: 5          Feature: month
Rank: 6          Feature: year
Rank: 7          Feature: transaction_id
Rank: 8          Feature: store_type_eShop
Rank: 9          Feature: prod_cat_code
Rank: 10         Feature: store_type_TeleShop
Rank: 11         Feature: store_type_MBR
Rank: 12         Feature: prod_subcat_code
```

FEATURE SELECTION RESULTS (CONT.)

So, we repeated the experiments but removing transaction_id and cust_id as they are unique identifiers and do not contribute much

Results

Lasso Regression

```
Rankings:
Rank: 1         Feature: Qty
Rank: 2         Feature: store_type_TeleShop
Rank: 3         Feature: store_type_MBR
Rank: 4         Feature: prod_cat_code
Rank: 5         Feature: prod_subcat_code
Rank: 6         Feature: store_type_eShop
Rank: 7         Feature: weekday
Rank: 8         Feature: day
Rank: 9         Feature: month
Rank: 10        Feature: year
Rank: 11        Feature: cust_id
```

```
Rankings:
Rank: 1         Feature: Qty
Rank: 2         Feature: store_type_TeleShop
Rank: 3         Feature: store_type_MBR
Rank: 4         Feature: prod_cat_code
Rank: 5         Feature: prod_subcat_code
Rank: 6         Feature: store_type_eShop
Rank: 7         Feature: weekday
Rank: 8         Feature: day
Rank: 9         Feature: month
Rank: 10        Feature: year
```

Random Forest Regression

```
Rankings:
Rank: 1         Feature: cust_id
Rank: 1         Feature: prod_subcat_code
Rank: 1         Feature: prod_cat_code
Rank: 1         Feature: Qty
Rank: 1         Feature: year
Rank: 1         Feature: month
Rank: 1         Feature: day
Rank: 1         Feature: weekday
Rank: 1         Feature: store_type_eShop
Rank: 1         Feature: store_type_MBR
Rank: 2         Feature: store_type_TeleShop
```

```
Rankings:
Rank: 1         Feature: prod_subcat_code
Rank: 1         Feature: prod_cat_code
Rank: 1         Feature: Qty
Rank: 1         Feature: year
Rank: 1         Feature: month
Rank: 1         Feature: day
Rank: 1         Feature: weekday
Rank: 1         Feature: store_type_eShop
Rank: 1         Feature: store_type_MBR
Rank: 2         Feature: store_type_TeleShop
```

XGBoost Regression

```
Rankings:
Rank: 1         Feature: Qty
Rank: 2         Feature: cust_id
Rank: 3         Feature: weekday
Rank: 4         Feature: day
Rank: 5         Feature: store_type_MBR
Rank: 6         Feature: month
Rank: 7         Feature: store_type_TeleShop
Rank: 8         Feature: year
Rank: 9         Feature: store_type_eShop
Rank: 10        Feature: prod_cat_code
Rank: 11        Feature: prod_subcat_code
```

```
Rankings:
Rank: 1         Feature: Qty
Rank: 2         Feature: year
Rank: 3         Feature: month
Rank: 4         Feature: store_type_eShop
Rank: 5         Feature: weekday
Rank: 6         Feature: day
Rank: 7         Feature: store_type_MBR
Rank: 8         Feature: prod_subcat_code
Rank: 9         Feature: store_type_TeleShop
Rank: 10        Feature: prod_cat_code
```

MODELING OBJECTIVES

- Customer Segmentation (RFM Analysis)
- Customer Churn Analysis
- Customer Lifetime Value

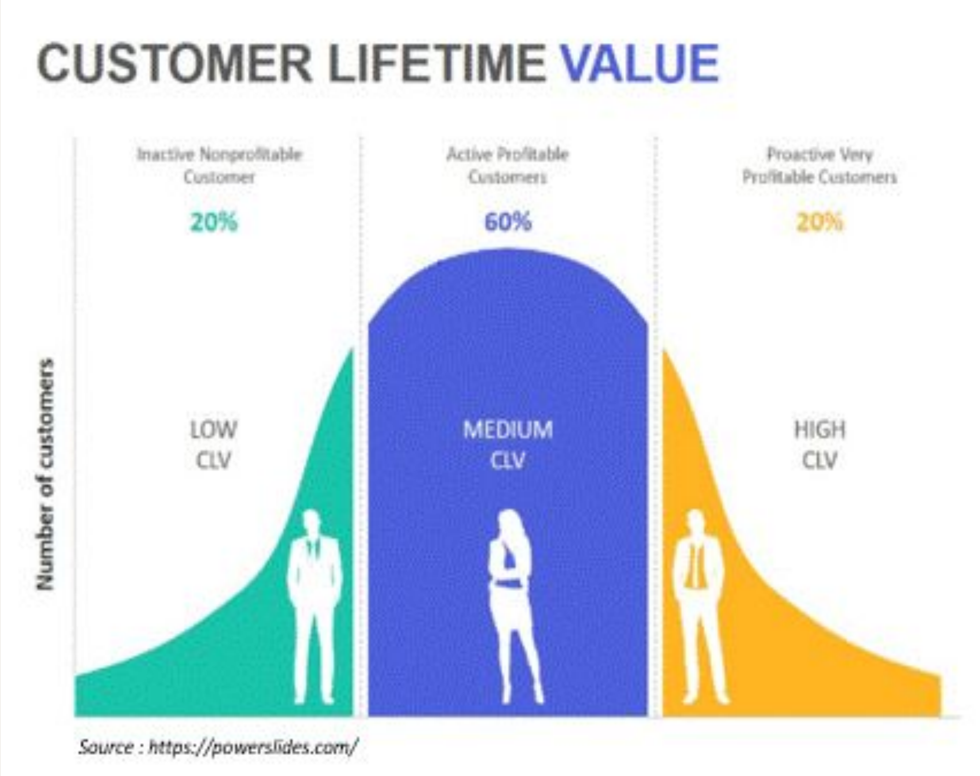
CLUSTERING TECHNIQUES FOR CUSTOMER PROFILING

- Clustering is a popular technique used in customer profiling to group customers with similar characteristics together.
- There are several clustering algorithms such as **K-Means**, **Hierarchical Clustering**, and **DBSCAN** that can be used to analyze customer data.
- K-Means is a simple and effective algorithm that partitions customers into k clusters based on their similarity.
- Hierarchical Clustering creates a tree-like structure of clusters, where each cluster contains sub-clusters.
- DBSCAN is a density-based algorithm that groups customers based on their proximity to each other.

RFM ANALYSIS FOR CUSTOMER SEGMENTATION

- RFM analysis is a powerful tool used in customer segmentation.
- It identifies high-value customers and predicts their future behavior.
- It involves analyzing three key metrics: **Recency**, **Frequency**, and **Monetary Value**.
- **Recency** refers to how recently a customer made a purchase.
- **Frequency** refers to how often they make purchases.
- **Monetary** Value refers to how much they spend.
- By segmenting customers based on these metrics, businesses can create targeted marketing campaigns.
- It helps in improving customer retention.

CUSTOMER LIFETIME VALUE PREDICTION



- Customer lifetime value (CLV) is the estimated amount of revenue a customer will generate over the course of their relationship with a business.
- Predicting CLV helps businesses identify high-value customers and allocate resources accordingly.
- Machine learning algorithms such as Linear Regression, Boosted Trees, and other regression models can be used to predict CLV.
- They use customer data such as purchase history, demographics, and customer behavior to predict CLV.
- This information can be used to improve customer acquisition and retention strategies.

CHURN PREDICTION

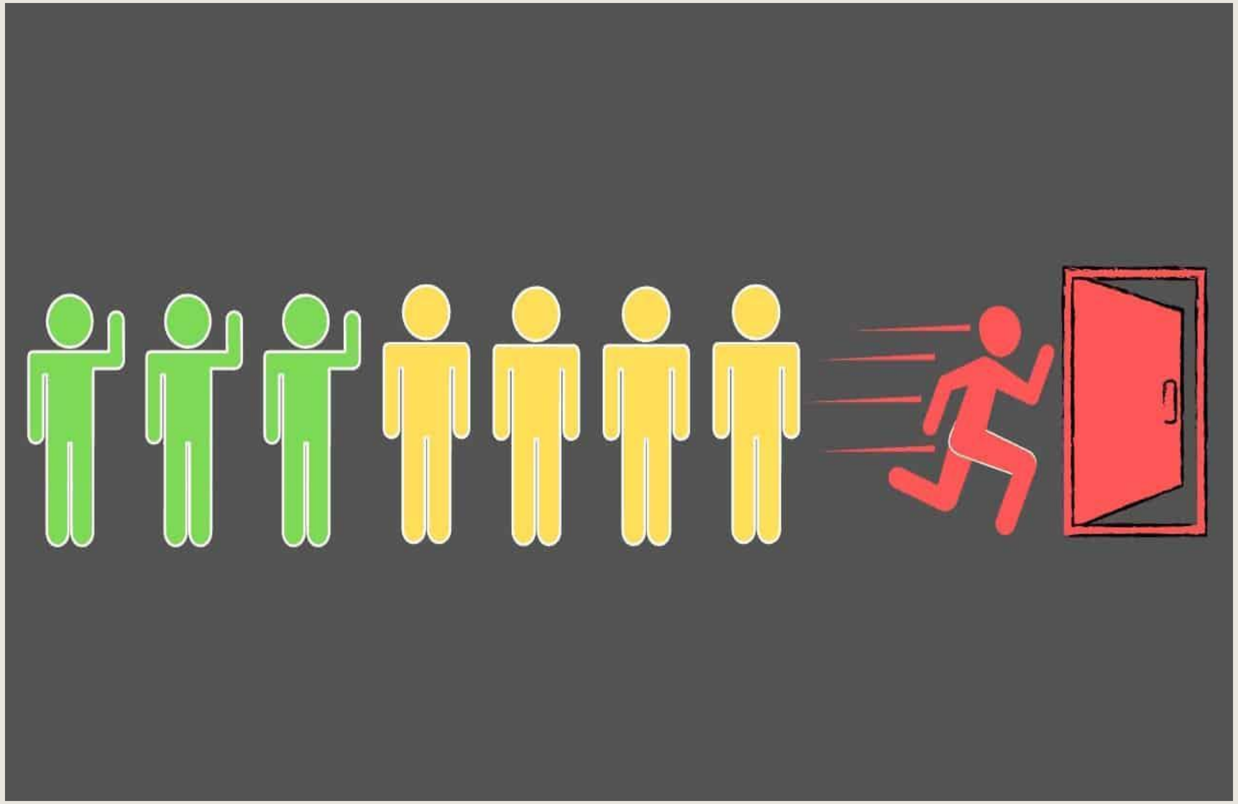
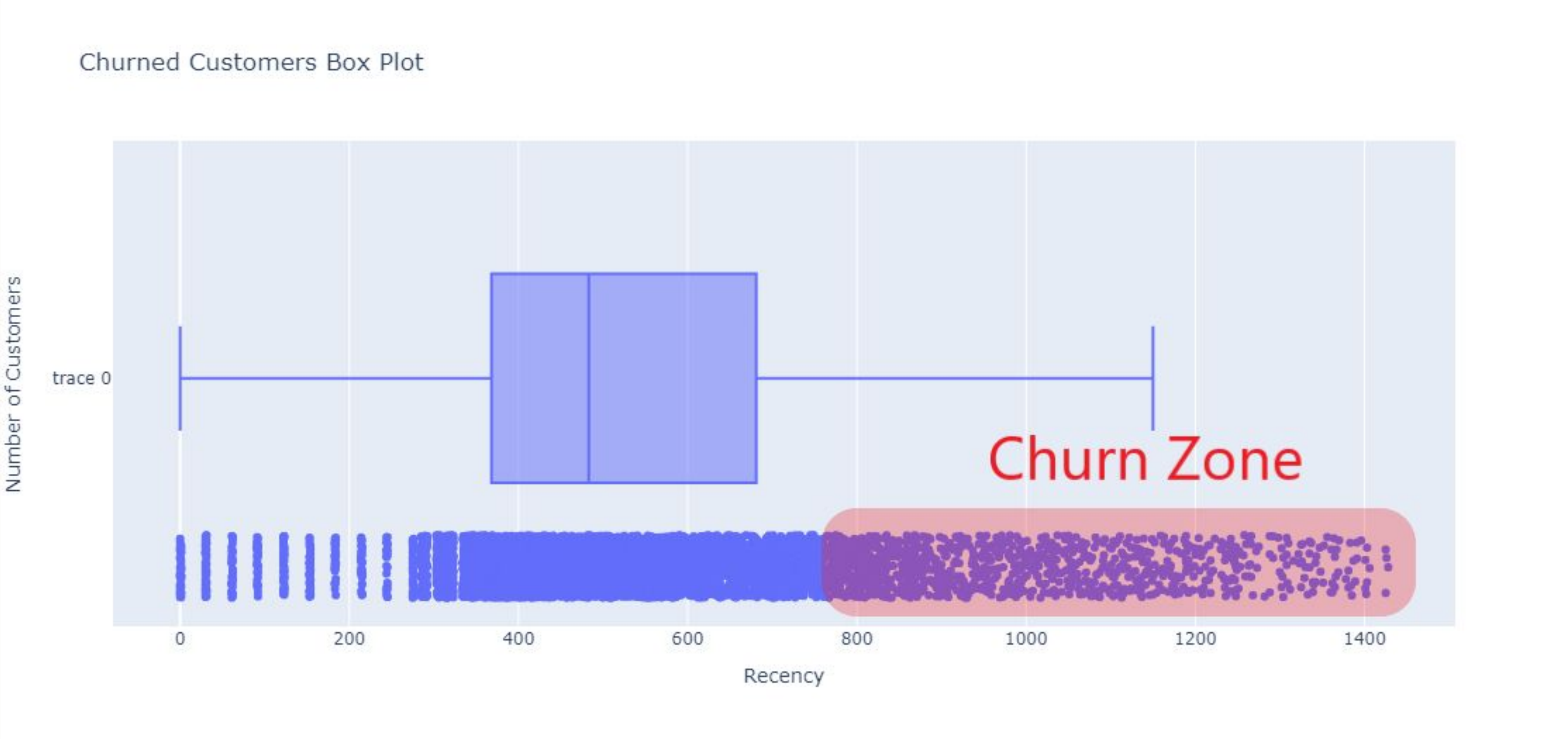


Image Source: https://userguiding.com/blog/how-to-reduce-churn/

- Churn prediction is the process of identifying customers who are likely to stop using a product or service.
- It analyzes customer data such as purchase history, customer support interactions, and demographic information.
- **Classification** algorithms can predict which customers are at risk of churning.
- Businesses can use this information to develop targeted retention strategies and prevent customer churn.
- We will use data mining methods for churn prediction such as **Logistic Regression**, **Random Forest**, and **XGBoost**.

CHURN ZONE


Churned Customers Box Plot
Number of Customers
trace 0
Churn Zone
0
200
400
600
800
1000
1200
1400
Recency

MODEL SELECTION & EVALUATION

- We implemented and evaluated various models using performance metrics such as accuracy, precision for classification, RMS for regression and Dunn Index for Clustering
- It was an iterative process of improving models through feature selection & hyperparameter tuning
- A final model was selected on best performance in terms of accuracy and other relevant metrics.

PERFORMANCE EVALUATION – CLUSTERING

Clustering Models used:

1. **KMeans++**
2. **Agglomerative (Hierarchical)**
3. **DBScan**

Initial Observations, as evidenced by the tables:

1. DBScan clusters appeared to be least separated based on the mean of each cluster
2. Agglomerative and Kmeans++ had relatively separated clusters

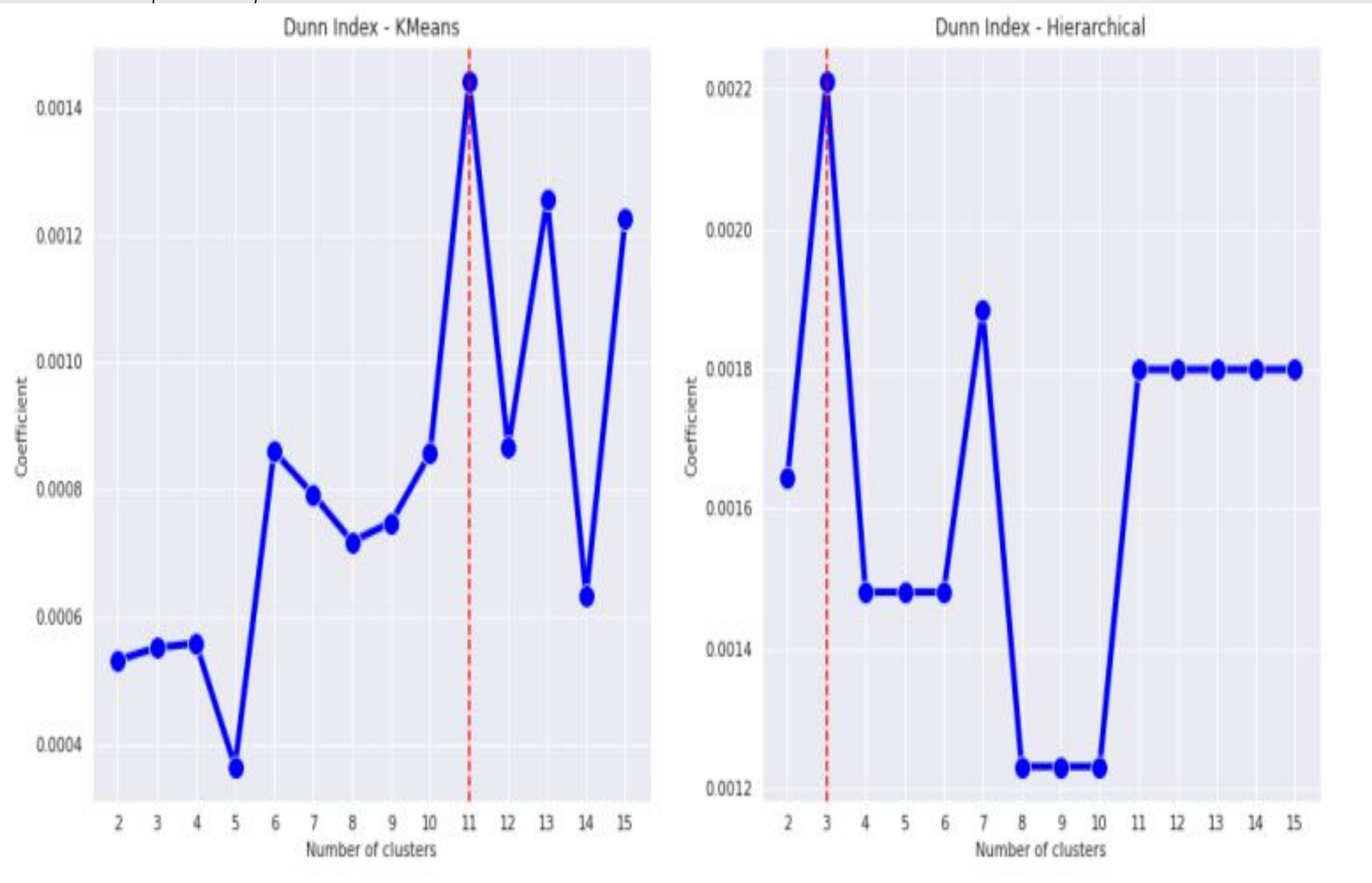
Revenue Clusters for DBScan

RevenueCluster	count	mean
-1	5242.0	9062.006136
0	59.0	4817.387966
1	62.0	0.000000
2	36.0	7228.296111
3	35.0	8215.517143
4	37.0	7400.065541
5	35.0	8352.379286

Revenue Clusters for KMeans++

RevenueCluster	count	mean
0	1780.0	2938.688525
1	481.0	21232.671486
2	1938.0	8002.091842
3	1307.0	13733.224916

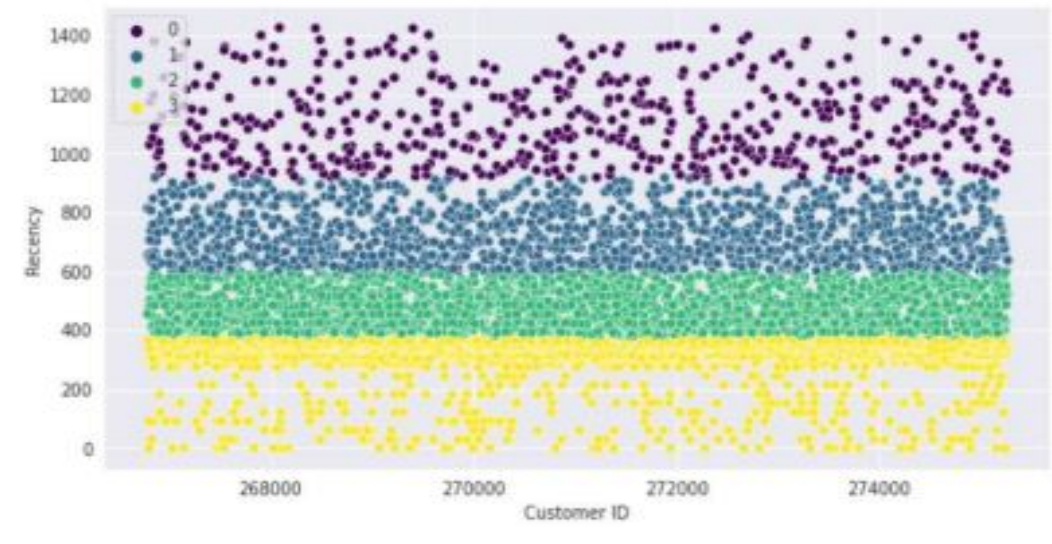
PERFORMANCE EVALUATION – CLUSTERING



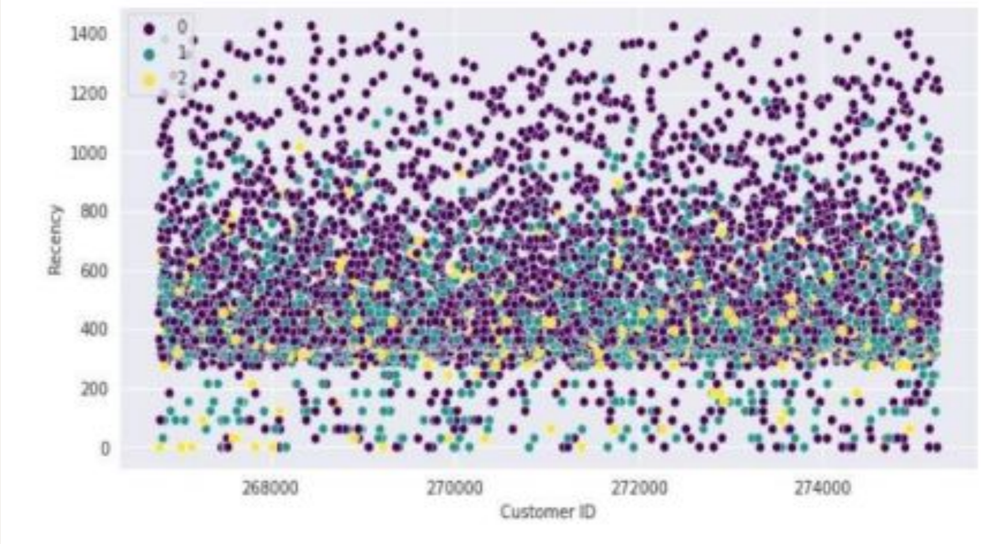
- Dunn Index for Kmeans++ and Agglomerative model
- Dunn Index is a slightly better for agglomerative clustering
- Dunn Index can be misleading in cases where intercluster distance is small
- Thus, visual inspection of the clusters was performed to further verify the performance in terms of cluster separation and compactness

PERFORMANCE EVALUATION – CLUSTERING

Recency Cluster for Kmeans++



Recency Cluster for Agglomerative



- On visualzing the clusters it was clear that Kmean++ out performed the Agglomerative clusters
- Thus, **Kmeans++** was picked as the preferred model for this task

PERFORMANCE EVALUATION - CLASSIFICATION

- Classification Models were used to predict churn, which is a simple indicator of **recency** of purchases
- Since the data was imbalanced, we performed both **oversampling** (**SMOTE**) and **undersampling** to balance the data
- In both cases, Logistic Regression and Random Forest were misclassifying a particular data point
- In both cases, when we switched to **XGBoost** Classifier we got perfect classification

UnderSampling	Logistic Regression	Random Forest Classifier	XGBoost Classifier
Accuracy	0.9985	0.9985	1
Precision	0.9971	0.9971	1
Recall	1	1	1
F1 Score	0.9985	0.9985	1
MCC	0.9971	0.9971	1

OverSampling	Logistic Regression	Random Forest Classifier	XGBoost Classifier
Accuracy	0.9994	0.9994	1
Precision	0.9971	0.9971	1
Recall	1	1	1
F1 Score	0.9986	0.9986	1
MCC	0.9982	0.9982	1

PERFORMANCE EVALUATION CUSTOMER LIFETIME VALUE

As a part of this section the following tasks were performed:

1. Multiple regression models were employed
2. Hyperparameter tuning conducted for model improvement and selection
3. Results of the analysis are presented in the following slides

PERFORMANCE EVALUATION – REGRESSION

Stochastic Gradient Descent (SGD)

- We utilized GridSearchCV for hyperparameter tuning
- Using the best parameters, we were able to achieve a significant reduction in error
- Prior to tuning: RMSE = 219672554947157.9062
- After tuning: RMSE = 7323.9049
- Not the best RMSE recorded, but noteworthy improvement

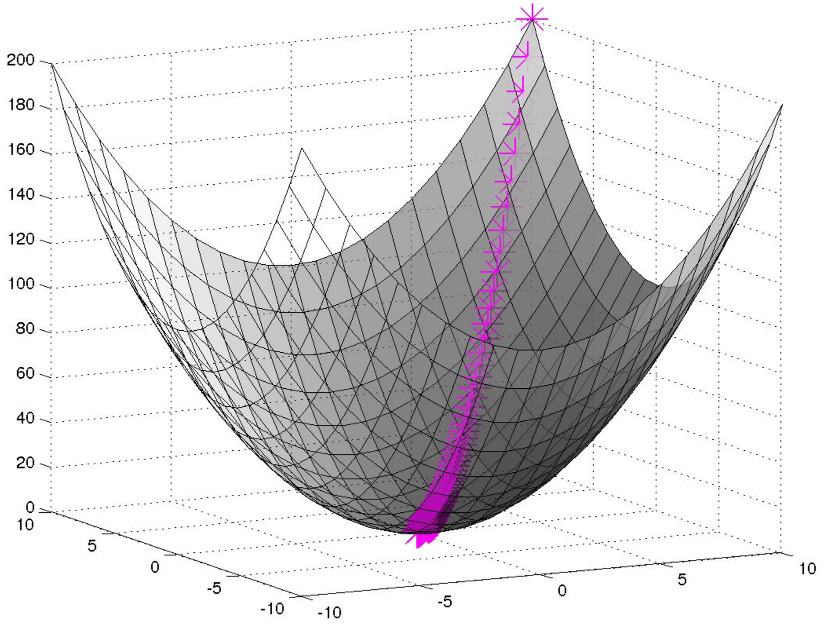


Image Source: simar (2023). Gradient Descent Visualization (https://www.mathworks.com/matlabcentral/fileexchange/35389-gradient-descent-visualization)

PERFORMANCE EVALUATION – REGRESSION

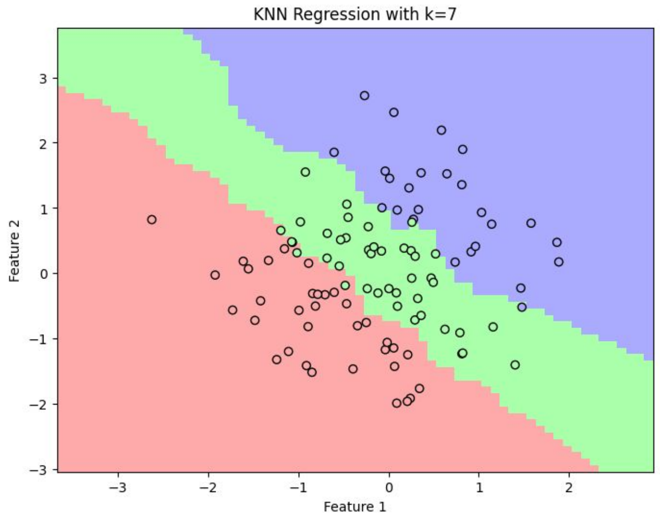
KNN Regressor

- Optimal value for k determined by iterating over various k values and picking the one with lowest validation error
- Lowest validation error observed where k was between **6** to **10**
- Observed RMSE values for various distances:

  - Chebyshev = 4214.4135

  - Euclidean = 4128.3531

  - Manhattan = 4162.6955



PERFORMANCE EVALUATION – REGRESSION

LASSO & RIDGE REGRESSION

- For Lasso Regression, R2 score remained approximately the same for all alpha values ranging [0.1,2]
- Similar trend was observed for Ridge Regression
- Thus, a regularization term did not improve the model fit

LIGHT GBM REGRESSOR

- We utilized GridSearchCV for hyperparameter tuning
- Using the best parameters, we were able to achieve a very minor reduction in error
- Prior to tuning: RMSE = 3628.4738
- After tuning: RMSE = 3625.6553

PERFORMANCE EVALUATION – REGRESSION

	Mean Squared Error	Root Mean Squared Error	Mean Absolute Error	Median Absolute Error	R2 Score	Adjusted R2 score	Spearman R
Linear Regression	1.241822e+07	3523.949986	2757.498644	2234.880818	0.626068	0.625614	0.773154
LARS	1.241822e+07	3523.949986	2757.498644	2234.880818	0.626068	0.625614	0.773154
Ridge Regression	1.241824e+07	3523.952081	2757.518378	2234.937541	0.626067	0.625614	0.773154
Ridge CV	1.241824e+07	3523.952083	2757.518397	2234.938026	0.626067	0.625614	0.773154
Lasso Regression	1.241825e+07	3523.953112	2757.527935	2234.965593	0.626067	0.625613	0.773154
Lasso LARS	1.241825e+07	3523.953113	2757.528010	2234.969106	0.626067	0.625613	0.773154
Huber Regressor	1.246329e+07	3530.338033	2740.223968	2235.713070	0.624711	0.624256	0.775285
Gradient Boost	1.278357e+07	3575.411579	2792.125569	2258.861644	0.615067	0.614600	0.778191
LGBM Regressor	1.314538e+07	3625.655263	2825.868031	2289.598039	0.604172	0.603692	0.776509
Transformed Regressor	1.314761e+07	3625.962975	2794.066173	2224.662416	0.604105	0.603625	0.777276
Poisson Regressor	1.500837e+07	3874.063487	2965.871912	2523.399445	0.559321	0.558787	0.773154
XGBoost	1.487274e+07	3856.518532	2983.600619	2401.597246	0.552158	0.551615	0.756483
Adaboost	1.496392e+07	3868.322111	3106.317376	2617.383133	0.549413	0.548866	0.778335
KNN Regressor	1.704330e+07	4128.353143	3261.965853	2757.113125	0.486799	0.486177	0.696809
Random Forest Regressor	1.770414e+07	4207.628994	3273.035789	2735.976632	0.466900	0.466254	0.710306
SGD Regressor	5.350386e+07	7314.633191	5661.549076	4425.410492	-0.611086	-0.613040	-0.210775

PERFORMANCE EVALUATION – REGRESSION

Based on the evaluation metrics in previous slide, Linear Regression was chosen as the optimal choice for this task

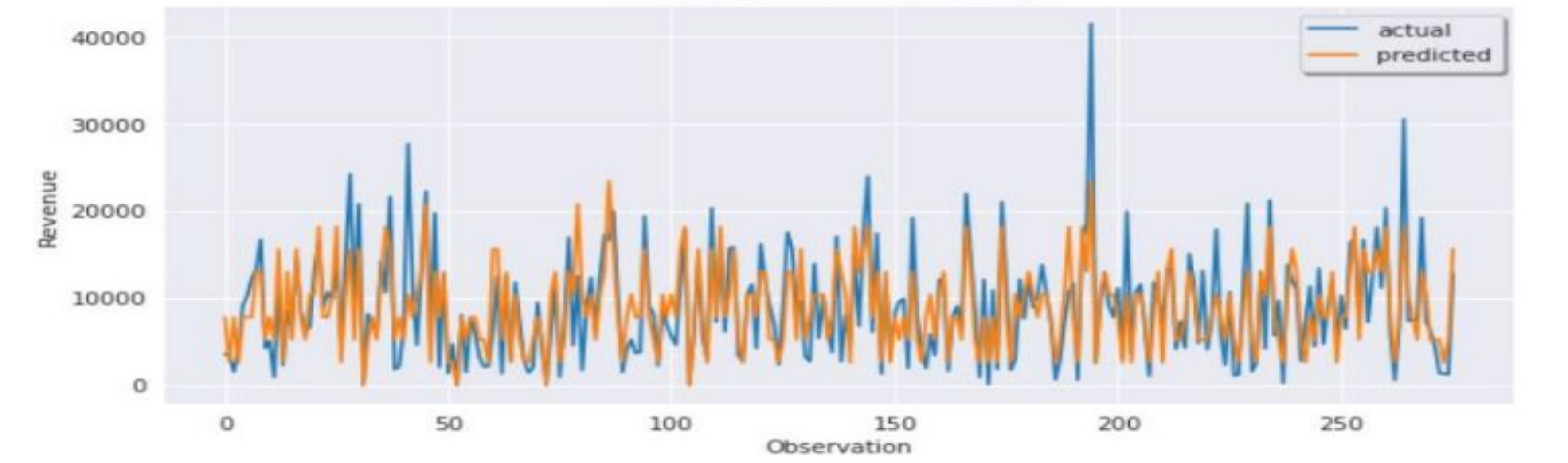
Thus, we visualized the actuals v/s predicted for Linear Regression, and following observations were noted:

While the model accuracy/fit wasn’t the highest, it predicts the trend of customer spending relatively accurately

This can help us identify customers that are more likely to spend

PERFORMANCE EVALUATION – REGRESSION

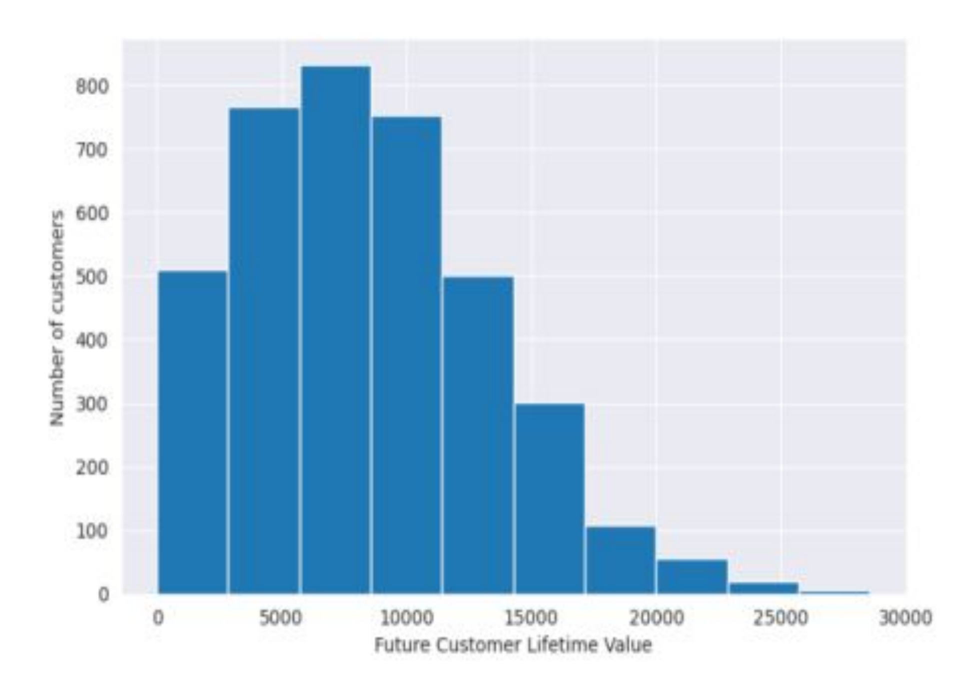
Actuals vs Predicted for Linear Regression



CUSTOMER PROFILE - FUTURE CUSTOMER LIFETIME VALUE

- We used the linear regression model we trained to predict revenue for each customer in the next 365 days
- We then combined results from Churn Prediction, CLTV and customer demographics to create customer profiles and derive actionable metrics

FUTURE LIFETIME VALUE DISTRIBUTION



Histogram of future customer lifetime value vs number of customers

The distribution is right skewed with majority customers have a spending capacity between 5000-10000

CUSTOMER PROFILE - TOP 10

	Customer ID	Recency_x	Frequency_x	Revenue	Churn	CLTV	Age	Age_Group	Gender	city_code
1248	270803	405	11.0	22162.985	0	28571.428683	36	35-45	F	4.0
936	272741	369	11.0	29264.820	0	28570.149207	50	45-55	F	7.0
384	270535	319	11.0	31969.860	0	28568.372157	35	25-35	F	7.0
1139	272354	487	10.0	33954.440	0	25976.351293	43	35-45	M	10.0
1509	272518	432	10.0	28142.140	0	25974.396538	51	45-55	F	9.0
1977	267346	393	10.0	13313.040	0	25973.010439	52	45-55	M	7.0
358	271565	317	10.0	21086.715	0	25970.309323	48	45-55	M	8.0
2311	270540	550	9.0	17383.860	0	23380.598624	43	35-45	F	1.0
1319	271834	412	9.0	41510.430	0	23375.693966	43	35-45	M	9.0
1276	273290	408	9.0	11094.200	0	23375.551802	33	25-35	M	3.0

- Combined data from all previous analysis to create holistic **customer profile**
- Analyzed top 10 customers with highest future CLV to identify patterns and behavior
- Top three customers with **highest future CLV** are female, from different age groups, and have buying frequency of 11, indicating loyalty

SUMMARY

- **Market Basket Analysis** - Association Rule Mining
- **RFM Clustering** - Unsupervised (KMeans++)
- **Churn Prediction** - Classification (XGBoost)
- **Customer Lifetime Value** - Linear Regression

THANK YOU

Debanjan Saha

MS in Data Analytics Engineering,
College of Engineering, Boston, MA

saha.deb@northeastern.edu

Ritika Rao

MS in Data Analytics Engineering,
College of Engineering, Boston, MA

rao.rit@northeastern.edu