Project Report

Predictive Modeling of Clickstream data for Online Shopping

Ruju Shah

Abstract:

In today's digital era, E-commerce giants strive to convert millions of browsing users into purchasers. Therefore, it's crucial for these companies to determine the value of each user interaction to maximize success. This will even help companies to optimize their offerings and tailor product recommendations to better serve the needs of their customers.

The goal of this analysis is to delve into patterns within the clickstream data and predict the potential purchasing amount or price for each user. Furthermore, the aim is to label each user interaction based on the potential purchase they might make in that ongoing session.

Dataset Selection and Preprocessing

- **Data Source:**
  Dataset Title: Clickstream data for online shopping sourced from the UCI dataset repository: [Link].

Data Description:

The clickstream data is collected from an online clothing store for pregnant women. It is collected by tracking each user's click on a digital product like a web application / mobile application. The data is from five months of 2008 and has 165,474 rows and 14 attributes. The original response variable is the price attribute, which is a continuous variable. However, we are interested in a discrete response variable that represents the potential purchase amount in that user session. Hence, we map the values of the continuous variable to discrete states and create a new variable called the price category. Therefore, it is a Classification Problem.

The following table summarizes the data attributes from the dataset.

Attribute	**Description**	**Data Type**	**Range of Values**
Year	Year when the data was collected.	Integer/Numeric	Just one - 2008
Month	Month when the data was collected.	Integer/Numeric	April (4) to August (8)
Day	Day when the data was collected.	Integer/Numeric	1 to 31
Order	Number of clicks during each session.	Integer/Numeric	1 to 195
Country	Country of origin for an IP Address.	Integer/Numeric	1 to 47
Session ID	Unique ID of each web session.	Integer/Numeric	1 to 24026

Page 1 (Main category)	Main product category.	Integer/Numeric	1 to 4: 1-trousers 2-skirts 3-blouses 4-sale
Page 2 (Clothing model)	Product code for each product under main category.	String	Alphanumeric: ` B4`. 217 Unique strings
Color	Color of each product.	Integer/Numeric	1 to 14: 1 – beige; 2 - black 3 – blue; 4 – brown 5 – burgundy; 6 - gray 7 – green; 8 - navy blue; 9 - of many colors; 10 - olive 11 – pink; 12 - red 13 – violet;14 - white
Location	Location of the picture w.r.t six frames in which the page is divided.	Integer/Numeric	1 to 6: 1 - top left 2 - top in the middle 3 - top right 4 - bottom left 5 - bottom in the middle 6 - bottom right
Model Photography	Indicates whether the photo is en-face / profile.	Integer/Numeric	1 to 2: en-face / profie
Price	Price of the product in USD.	Integer/Numeric	18 to 82
Price 2	Indicates whether the price of a specific product is higher than the entire product category.	Integer/Numeric	1 to 2: YES / No
Page	Page number of the shopping website.	Integer/Numeric	1 to 5
Price Category	Potential amount a user might spend in the current session.	String	0 - 25: Budget, 25 - 35: Value, 35 - 65: Average, 65 - 100: Premium

```
!wget https://rahul-public-datasets.s3.ap-south-1.amazonaws.com/ecommerce-clothing-data.gzip
```

The above command can be used to retrieve the data from S3 bucket(from cloud) while using the Colab instead od loading the data again andagain as the run time ends.

Data Cleaning:

- **Data Cleaning and Preprocessing**
  In preparation for analysis and modeling, the raw clickstream data underwent a series of cleaning and preprocessing steps. These steps aimed to enhance the quality and structure of the dataset, making it suitable for further analysis. The following procedures were carried out:

- **Renaming Columns**
  Column names were standardized to improve readability and consistency throughout the dataset. This involved renaming specific columns such as 'session ID' to 'session-id' and 'price 2' to 'price-higher-than-category', among others.

- **Encoding Categorical Variables**
  Categorical variables within the dataset were encoded to numerical format using a custom LabelEncoder class. This facilitated the processing of categorical data by machine learning algorithms. For instance, categorical variables like 'category' and 'model-photography' were encoded into numerical representations.

- **Bucketing Price Data**
  To simplify the analysis and modeling process, the continuous variable 'price' was bucketed into categorical price categories. This was achieved by defining price ranges and assigning each price value to a corresponding category such as 'budget', 'value', 'average', or 'premium'. Subsequently, a LabelEncoder was employed to encode the categorical price categories into numerical format.

- **Handling Missing Values**
  A thorough inspection of the dataset revealed no missing values in any of the columns, ensuring the completeness of the data for analysis.

- **Summary of Data Structure**
  After preprocessing, the dataset comprises various attributes including temporal information such as year, month, and day, as well as categorical variables like 'category', 'colour', and 'location'.

Data Exploration and Visualization:

Bar charts, Histograms, line graphs, Count plots, Choropleth maps have been plotted to check the price distribution, how the values of the columns are distributed, to visualize how the price of different products varies over time and to get insights on which countries are spending most on online shopping

Exploratory Data Analysis:

Univariate Analysis

Understanding the Years:

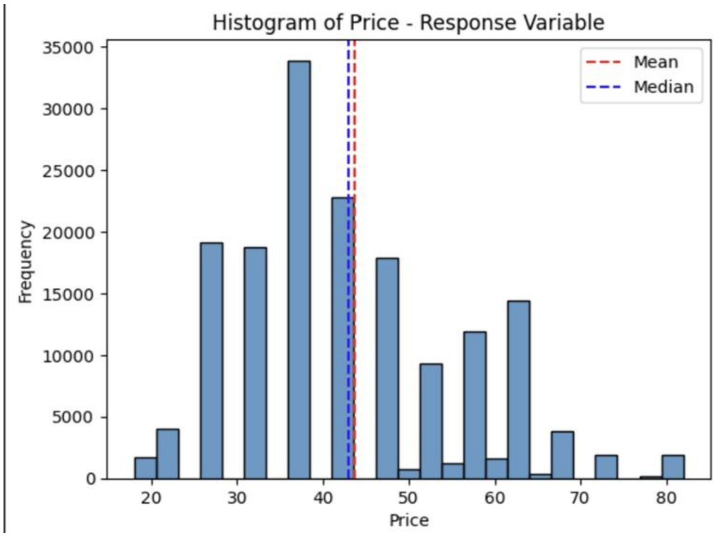
We checked the range of years present in our data and found that all the entries are from the year 2008. Since this doesn't provide us with any unique information about our response variable, we decided to drop the 'year' attribute.

Investigating Session IDs

We counted the number of unique session IDs, which are unique identifiers for each user session. However, since these IDs don't give us any insights into user behavior, we determined that they can be dropped as well.

Analyzing Price Distribution

We created a histogram to visualize the distribution of prices in our dataset. The histogram shows that most prices are clustered around $44. The red dashed line represents the mean price, which is approximately $44, and the blue dashed line represents the median price, which is about $43.



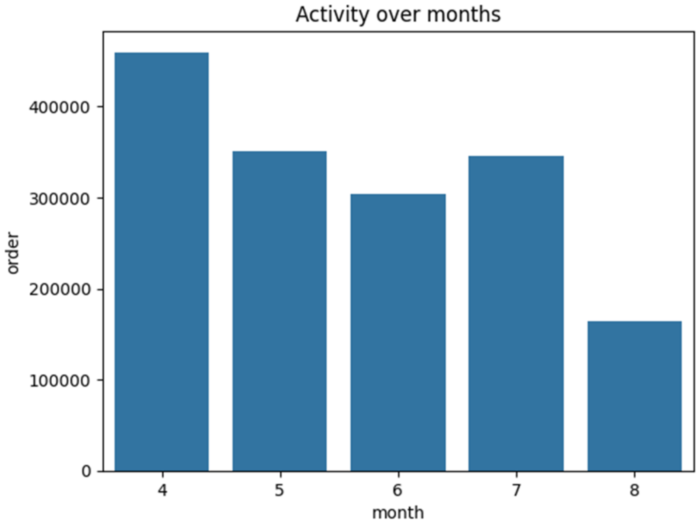
This analysis gives us a clear picture of the distribution of prices in our dataset, with the majority of prices centering around $44.

Monthly Activity Analysis

To understand the variation in orders across different months, we examined the monthly order activity

Identifying Peak and Low Months

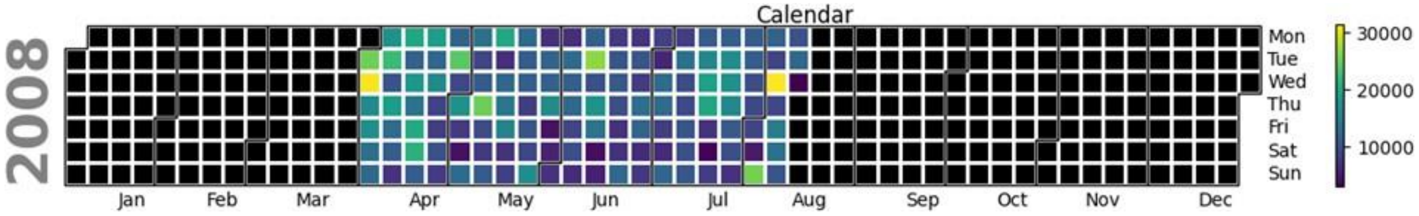
We grouped the data by month and calculated the total number of orders for each month. The resulting bar plot visualizes the monthly order activity, with each bar representing the total orders for a specific month.



Observations
Peak Month: April
Lowest Month: August

Calendar Visualization of Order Activity

We visualized the order activity over time using a calendar plot, which provides an intuitive overview of order trends throughout the year:



<u>Key Observations</u>:

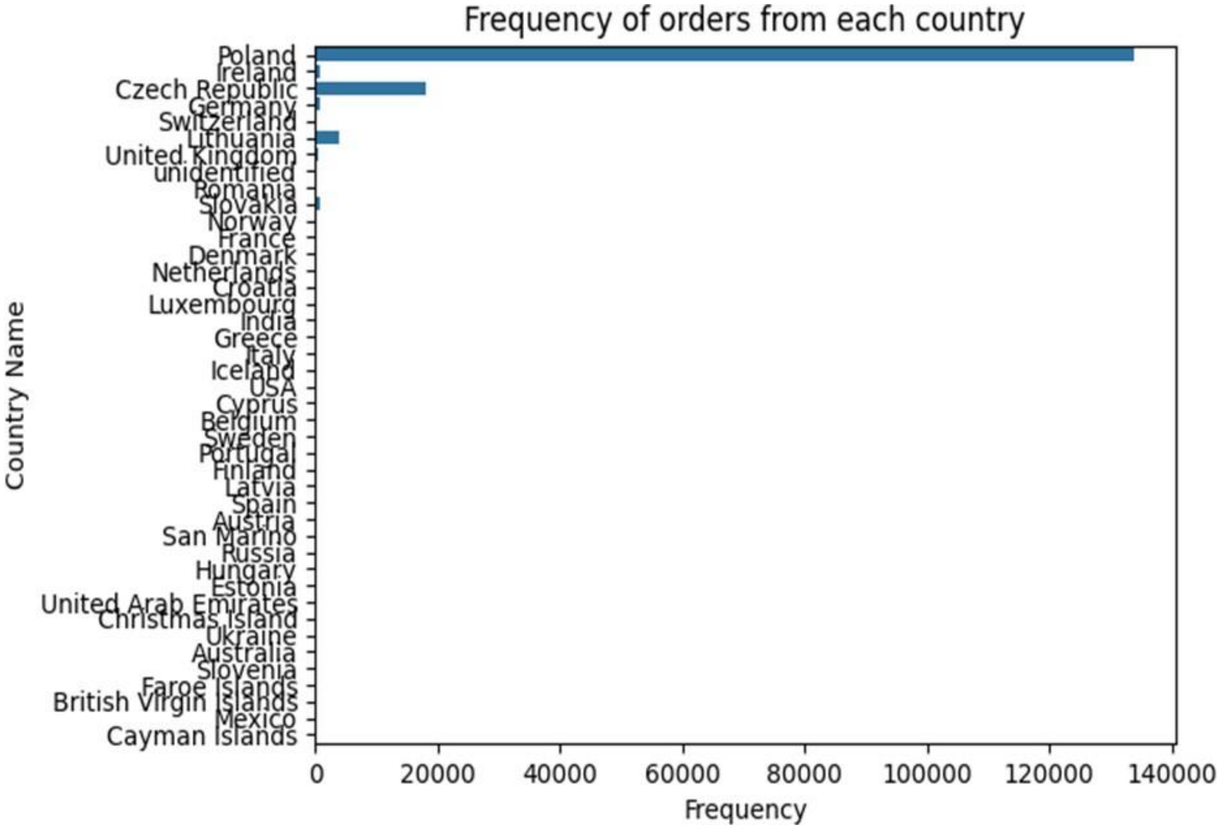
April Peaks: April stands out with the highest number of orders, indicated by lighter-colored cells on the calendar plot.
June Lows: Conversely, June exhibits the lowest order activity, with darker-colored cells indicating fewer orders.
May and July: These months show relatively average order activity, with similar numbers of orders received.
August Spike: August shows a notable spike in order activity on one particular day, highlighted by a very light-colored cell.

<u>Insights:</u>

The calendar visualization highlights the seasonal variations in order activity, with April emerging as a peak period for orders.
Understanding these patterns can help in planning inventory, marketing campaigns, and resource allocation to capitalize on peak periods and address slower months effectively.

Orders by Country

We looked at where our orders were coming from to see if any countries stood out:



What We Found:

Top Countries: Countries like Poland, Czech Republic, and Lithuania had the highest frequency of orders.
Chart Overview: The bar chart shows the number of orders from each country. Each bar represents a country, and the taller the bar, the more orders we received from that country.

Next Steps:

We noticed that a few countries had a really high number of orders. We removed those countries from our analysis to get a clearer picture of order frequency from other countries.

Countrywise Order Count

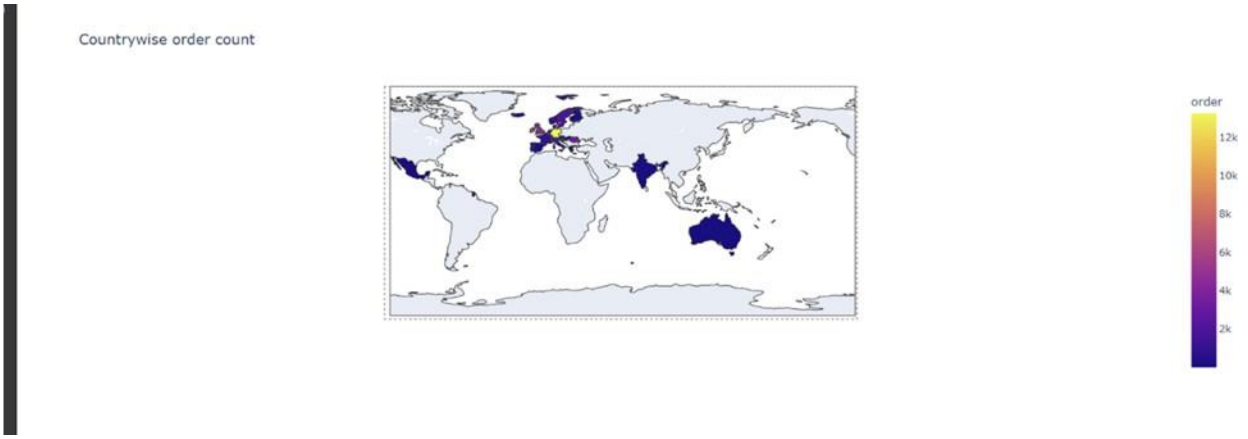
We visualized the number of orders from different countries using a map

What We Did:

Data Preparation: We combined our order data with country information, so we could see where the orders were coming from.

Filtering: We removed countries like Poland and Czech Republic with very high order counts to focus on other countries.

Visualization: The map shows the number of orders from each country. Darker colors represent more orders, while lighter colors represent fewer orders.



<u>Key Insights</u>:

High-Order Countries: Germany, UK, Ireland, Norway, and Romania had a significant number of orders compared to other countries.

This map helps us understand where our orders are coming from and which countries contribute the most to our business.

Price Contribution by Country

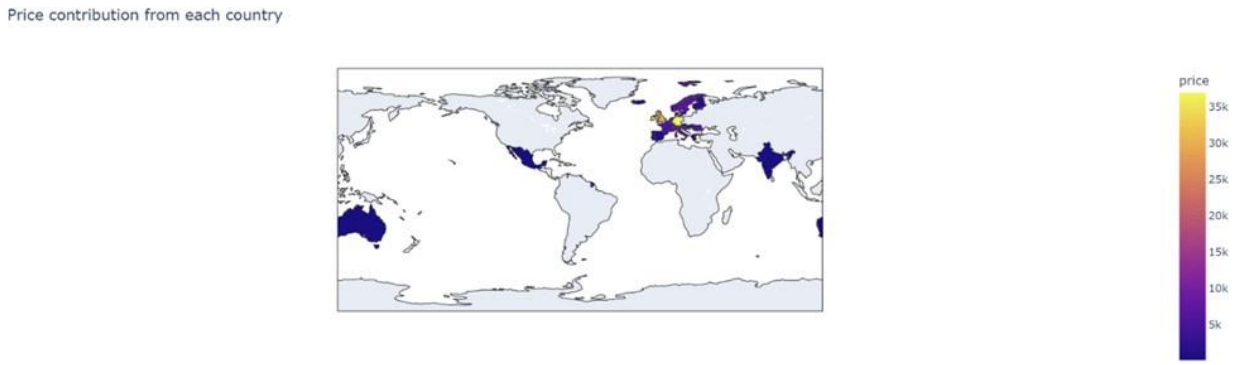
We visualized the contribution of different countries to the total order value using a map:

What We Did:

Data Selection: We selected the country and price data from our dataset to understand how much each country contributes to the total order value.

Filtering: We removed countries like Poland and Czech Republic with exceptionally high order values to focus on other countries' contributions.

Visualization: The map illustrates the total order value contributed by each country. Darker colors represent higher order values, while lighter colors represent lower values.



<u>Key Insights:</u>

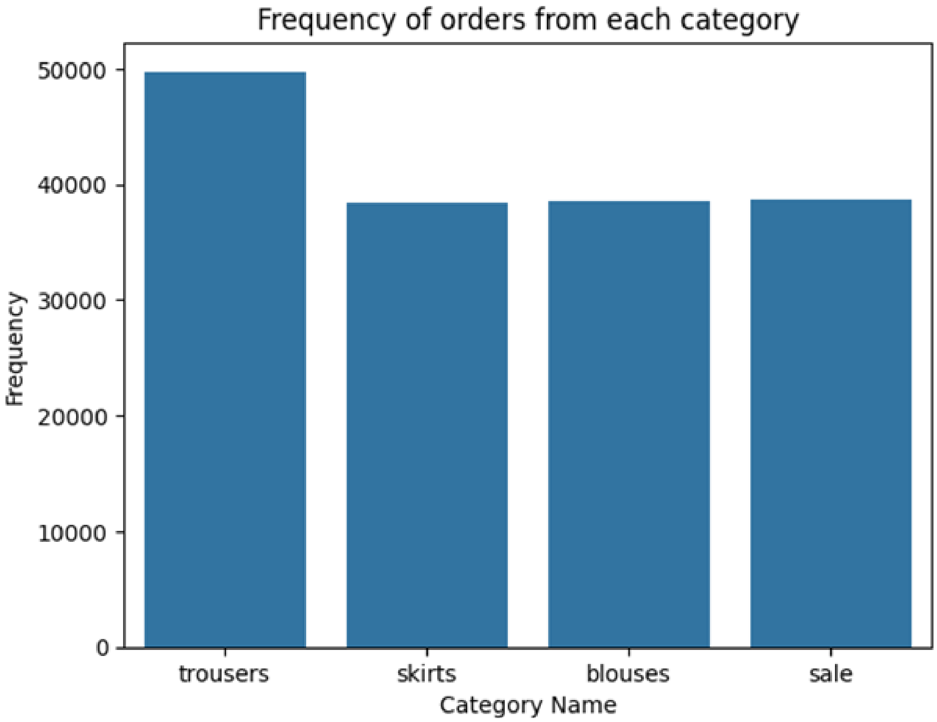
Top Contributing Countries: Germany, Ireland, UK, France, Sweden, Norway, and Romania emerged as significant contributors to the total order value, following closely behind Poland and Czech Republic.
This map helps us identify the countries that play a significant role in driving our business's revenue through high-value orders.

Product-Category wise Analysis

What We Did:

Data Selection: We focused on the product category, order count, and month from our dataset to understand the popularity of each category.

Visualization: The bar chart shows the frequency of orders from each product category. Each bar represents a category, and the height of the bar indicates how often that category was ordered.



<u>Key Insights</u>:

Trousers Top the List: Trousers were the most popular product category, with the highest number of orders compared to other categories.

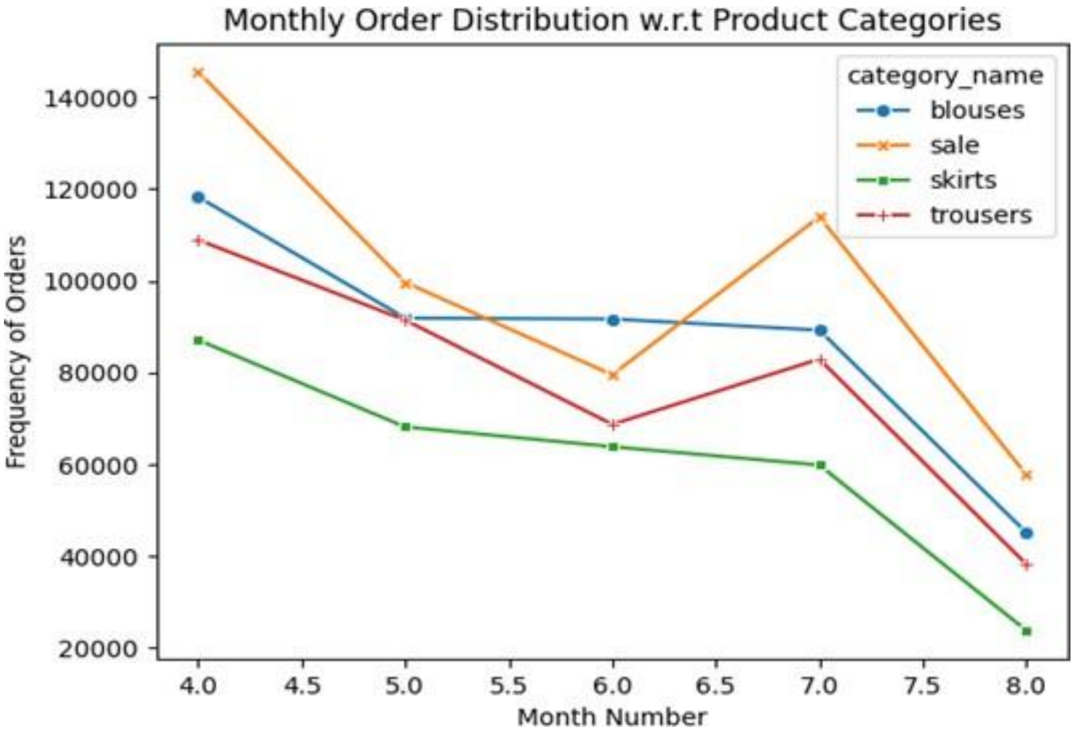
Equal Interest: Other categories such as skirts, blouses, and sale items received a similar number of orders, suggesting relatively equal interest among users.

Monthly Order Distribution by Product Categories

What We Did:

Data Analysis: We grouped the data by month and product category to understand the distribution of orders over time.

Visualization: The line plot shows the trend of order frequency for each product category over the months. Each line represents a product category, and the y-axis indicates the number of orders, while the x-axis represents the month.



Key Insights:

Sale Months: April and August stand out as months with increased order frequency, particularly for sale items.

Blouses Leading: Despite trousers having a higher total number of orders overall, blouses consistently have a higher number of orders each month.

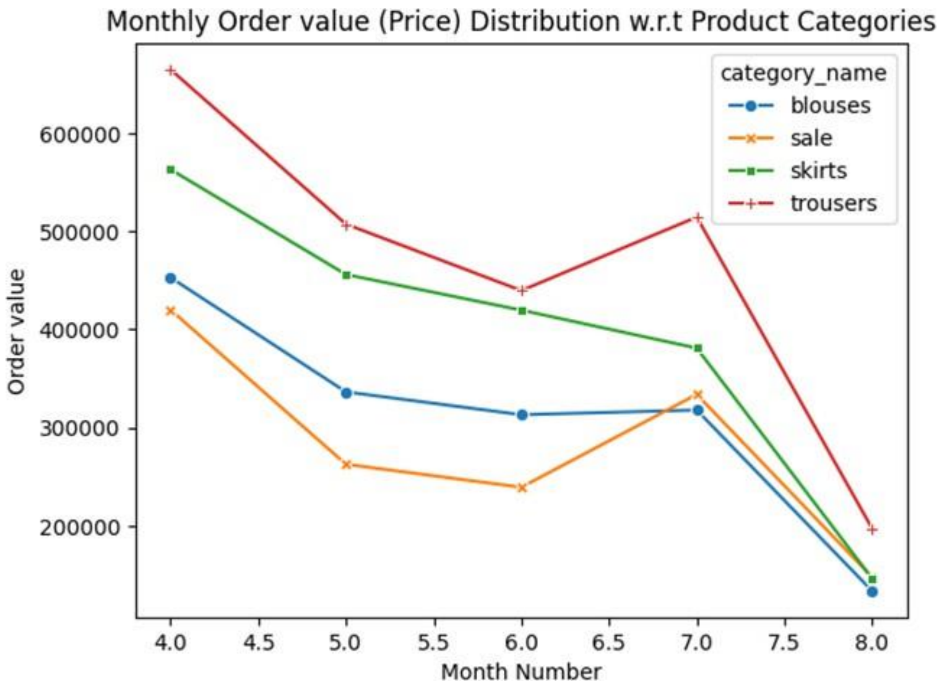
Skirts Trend: Skirts show a relatively lower frequency of orders compared to other categories, remaining consistent throughout the months.

Monthly Order Value Distribution by Product Categories

What We Did:

Data Analysis: We selected the product category, price, and month from our dataset to understand the distribution of order values over time.

Visualization: The line plot shows the trend of order value for each product category over the months. Each line represents a product category, and the y-axis indicates the total order value, while the x-axis represents the month.



<u>Key Insights</u>:

Trousers' Price Premium: Despite trousers being ordered less frequently, they have a higher price compared to other categories, indicating a premium pricing strategy for trousers.

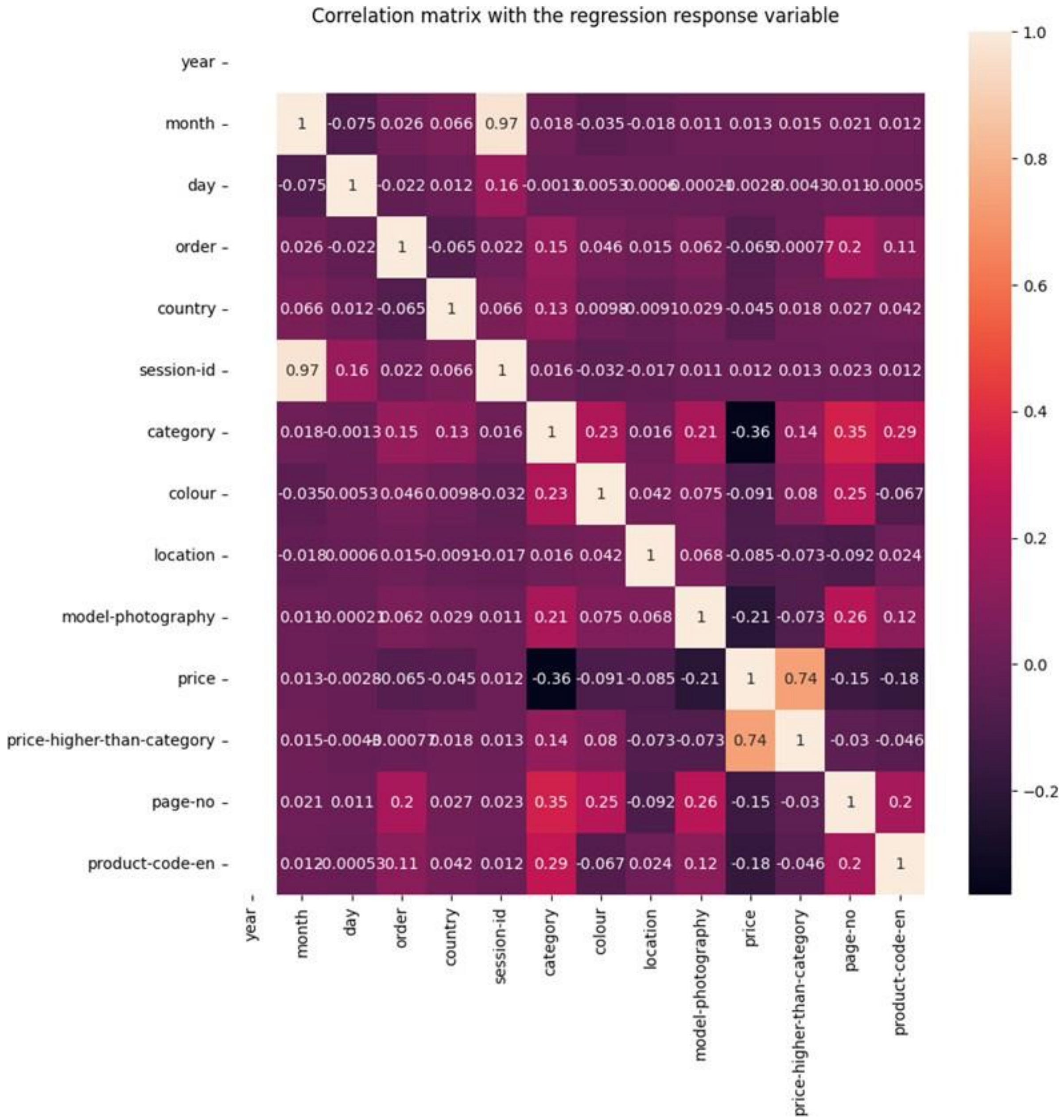
Blouses' Pricing Strategy: Blouses, despite having a higher number of orders, have a lower price compared to other categories. This suggests that blouses may have been priced lower, leading to higher order volume.

Price-Order Relationship: The price-order relationship varies across categories, with some categories having higher prices and lower orders, while others have lower prices and higher orders.

Bivariate Analysis

Correlation Analysis

We used Pearson's correlation to understand the relationships between different attributes:



Key Observations:

Year Unimportant: The 'year' attribute doesn't seem to have much impact on the analysis.

Month and Session ID: There's a strong correlation between 'month' and 'session-id'. This is because session IDs increase with each new month, but this doesn't provide unique information.

Price and Price-Higher-Than-Category: 'Price' and 'price-higher-than-category' have a strong positive correlation. We'll keep 'price' and remove 'price-higher-than-category'.

Regression Analysis

We constructed a regression model using Ordinary Least Squares (OLS) method to predict the price based on various attributes.

Model Evaluation: We examined the summary statistics of the regression model to understand the significance of each attribute in predicting the price.

Key Findings:

P-Values Analysis: Attributes like 'day', 'order', 'country', and 'page-no' have p-values higher than 0.05, indicating that their coefficients are not statistically significant in predicting the price.

Backward Elimination: We performed backward elimination to test the model's performance by removing these attributes. The adjusted R-squared values for different models (a, b, c, d) remained constant, indicating that removing these attributes didn't affect the model's ability to capture variance in the price.

Implications:

Attribute Importance: Attributes such as 'day', 'order', 'country', and 'page-no' don't seem to contribute significantly to predicting the price. Removing these attributes doesn't impact the model's performance.

Preprocessing like Standardization is initially done on Training data. And then applied to Validation data, to ensure that model

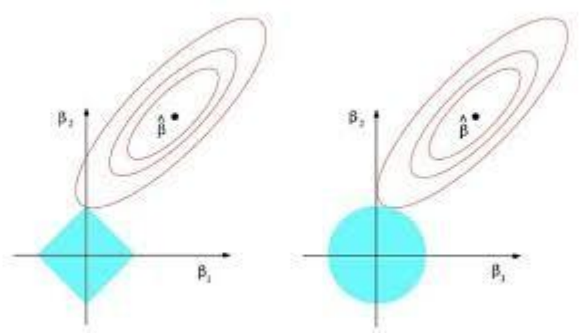
Rule of Thumb: 10 records out of entire data has been kept a side to test on how model performs on unseen new data

Model Building:

Linear : parametric approach where in we assume the relationship between the independent and dependent variables is assumed to be linear and we try to find the unknown values(parameters) by training our model

Ridge : Regularize the model to use certain data by **adding L2 penalty term**

LASSO (Least Absolute Shrinkage and Selective Operation): automatic feature selection technique where in the model decide all the features that are important and drop some of the weights of the features to zero most of the times by **adding L1 penalty term**



Neural Networks: which mimics the human neuron system, where in each neuron is connected to neurons in subsequent layers forming a network and can be used both for regression and classification

Soft max Regression (Multinomial Logistic Regression One Versus All Classification) : multiclass classification technique wherein we use multiple sigmoids with normalization and we tend to maximize the soft(normalized) probability

KNN(K Nearest Neighbors): can be used for both regression and classification, by assigning integer for ‘K’ (hyper parameter) and average of the K nearest observations and mode of K nearest observation are used in predicted the value and class label for regression and classification respectively

Gaussian Naive Bayes: is used for numeric data where in columns are independent and also values in the column are normally distributed

SVM (Street View Classification): technique that finds optimal classifier, among all possible classifiers by minimizing the distance between the point and the line implied by maximizing the margin and constructs a line exactly middle to both boundaries

HardMargin SVM: linear - doesn't allow mis classification)

SoftMargin SVM: linear - allows for mis classification

Model Evaluation:

Implemented Linear, Ridge Regression as we found that inverse of (X.T.X) is invertible by showing that Rank(X.T.X) is full rank and not a low rank and if closed form/analytical, normal equation does not exist for algorithms like LASSO Gradient descent have been performed.

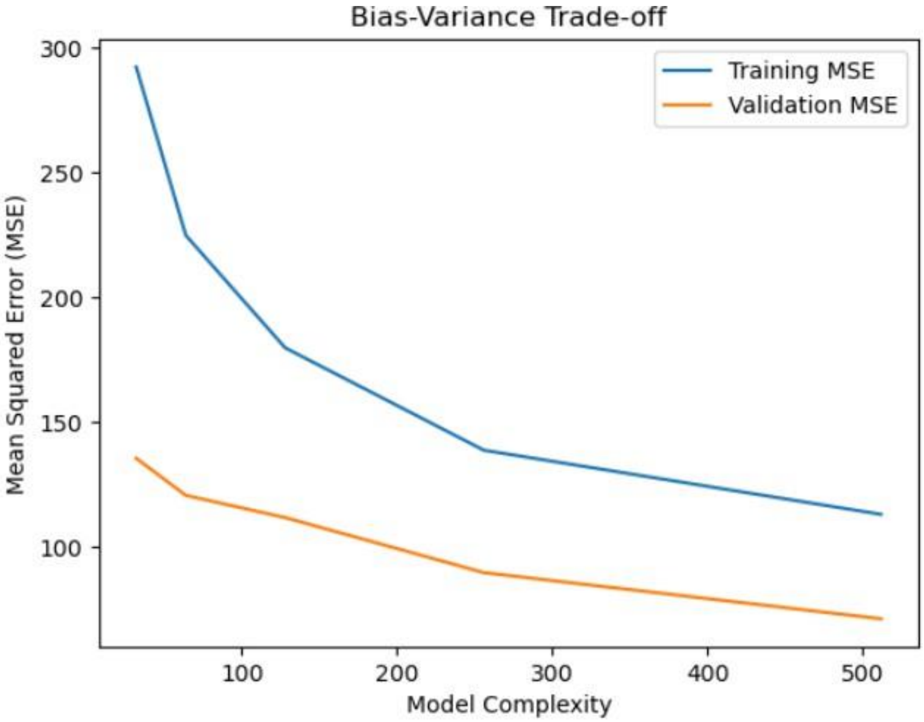
Regression				
Algorithm	**Model Fit - R^2_Score(NumPy)**	**Model Test - RMSE(NumPy)**	**Model Fit - R^2_Score(Sklearn)**	**Model Test - RMSE(SkLearn)**
Linear	0.16509	11.49163	0.16509	11.49163
Ridge	0.16509	11.49162	0.16509	11.49164
LASSO	0.16508	11.49169	12.57668	0
Neural Networks	N/A	N/A	0.64369	7.50707

Since we dropped all the fields that are correlated using the correlation matrix and selected the features on importance and there is no dependency or relation between the features we have chosen for classification, Gaussian Naive Bayes can be implemented

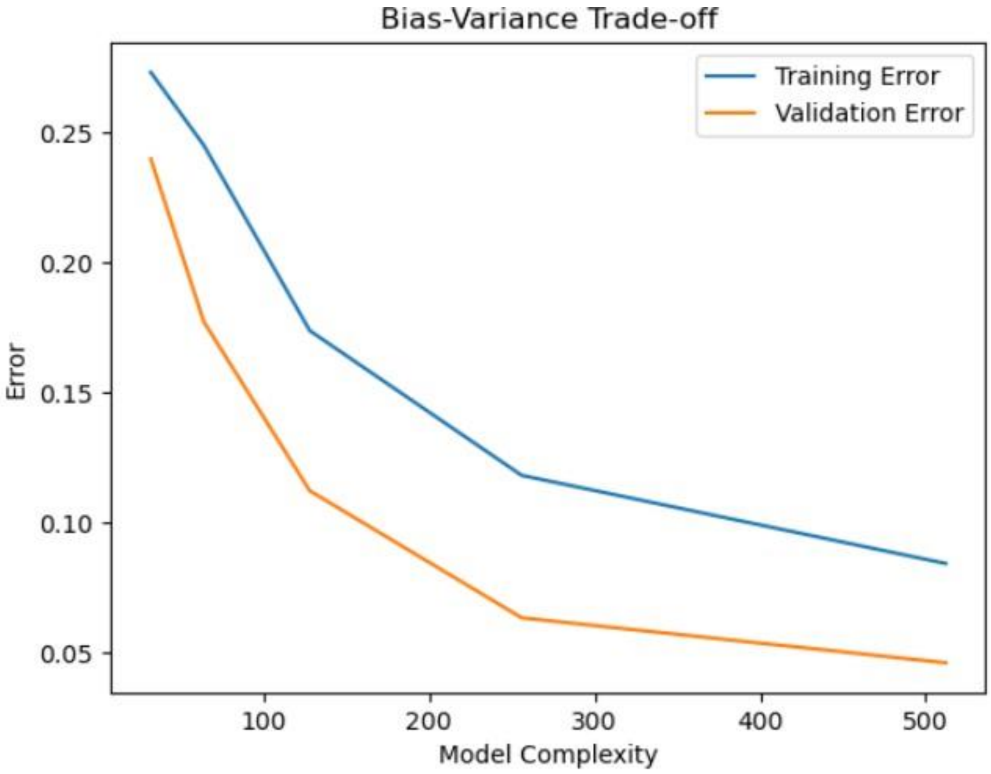
Classification		
Algorithm	**Accuracy - Numpy**	**Accuracy - SkLearn**
Softmax	69%	69%
KNN (K Nearest Neighbors)	Memory Error	100%
Gaussian NaïveBayes	65%	100%
HardMarginSVM	69%	Not Implemented
Neural Networks	Not Implemented	96%
SoftMarginSVM	Memory Error	Not Implemented

Bias Variance Trade-off:

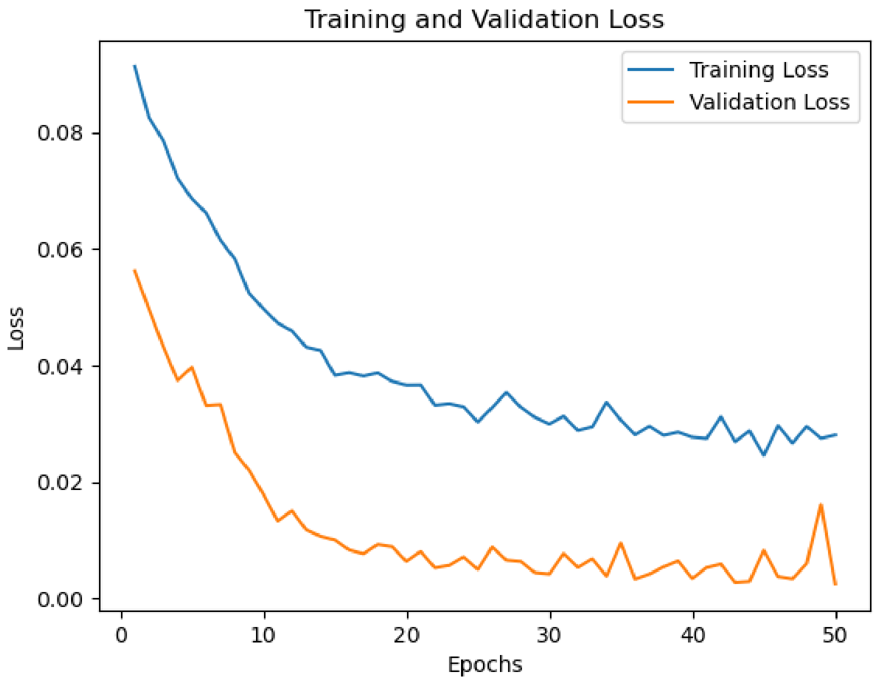
Regression:



Classification:



As the model complexity increases by varying the number of neurons in hidden layer, the training MSE and the validation MSE start to decrease. This is because the model is not too flexible so that it doesn't overfit by fitting the noisy data and also not too complex to be performing not good on both training and test data, hence balancing the model's complexity with the amount of available data is achieved.



The bias variance trade-off has also been performed by training the model for different number of epochs and after certain epochs(=50), the decrease in the train error and the validation error is constant and we have stopped fitting the model at the point of time as it reached the point of convergence. The entire goal of Bias-Variance trade off is to create a model that is complex enough to capture the important patterns in the data but simple enough to generalize well to new data. This involves not only selecting the right number of predictors but also ensuring that the data is sufficient to support the model's complexity.

Model Complexity and Challenges:

- KNN's fit on X_train_std = 1Lakh+ records gives Memory Limit Error, as it generates a pairwise distance matrix of 1Lakh+ x 1Lakh+
- We were not able to run the SVM, particularly Soft Margin SVM, is memory intensive and might require a significant amount of memory to store the training data and the computed support vectors.

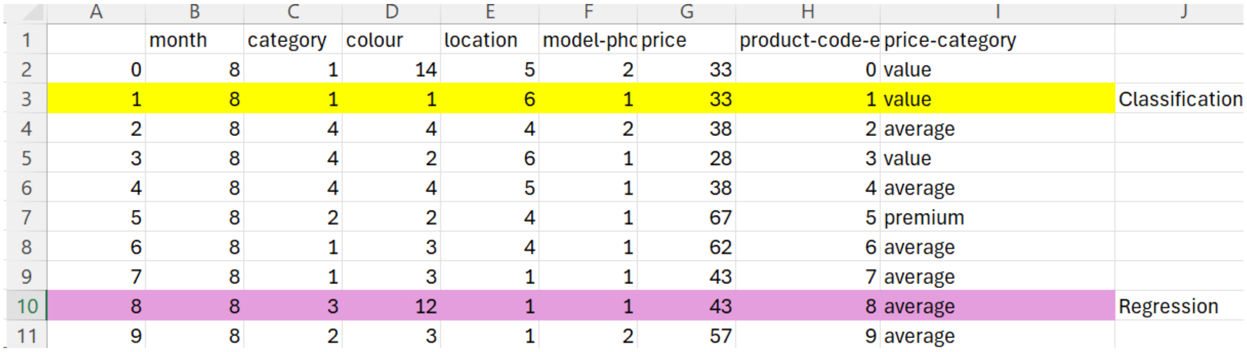
For both regression and classification it has been found that Neural Networks have outperformed of all the algorithms and hence provides accurate estimates on our data, we downloaded the Neural Network Regressoe and Classifier respectively for Model deployment.

Model Deployment:

The model has been deployed using AWS EC2, the best performing model for both regression and classification have been chosen

URL: http://18.191.32.75:5000/

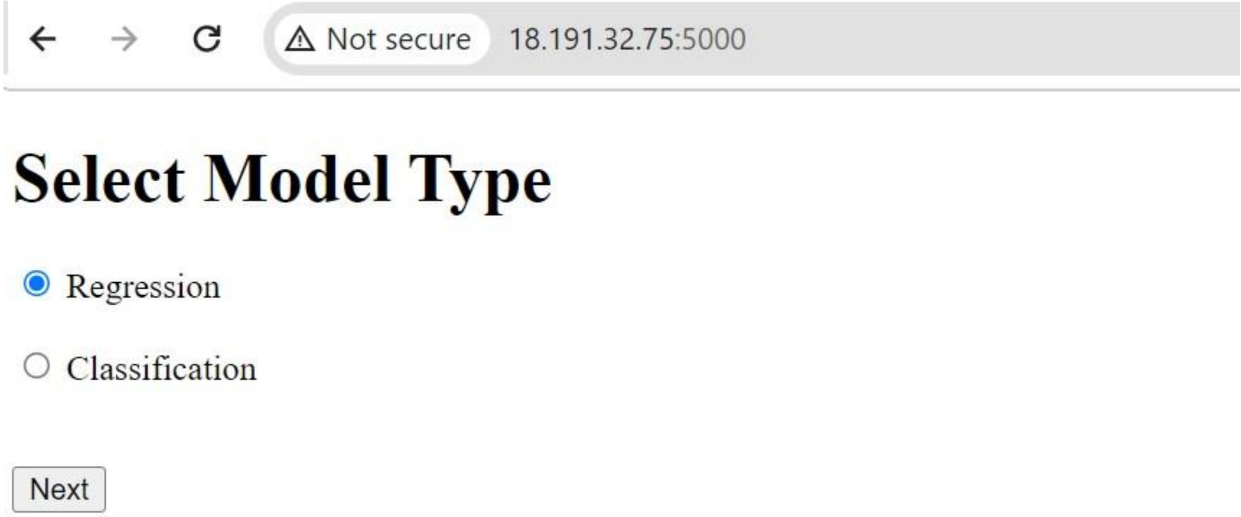
User Flow:



	A	B	C	D	E	F	G	H	I	J
1		month	category	colour	location	model-phc	price	product-code-e	price-category	
2	0	8	1	14	5	2	33	0	value	
3	1	8	1	1	6	1	33	1	value	Classification
4	2	8	4	4	4	2	38	2	average	
5	3	8	4	2	6	1	28	3	value	
6	4	8	4	4	5	1	38	4	average	
7	5	8	2	2	4	1	67	5	premium	
8	6	8	1	3	4	1	62	6	average	
9	7	8	1	3	1	1	43	7	average	
10	8	8	3	12	1	1	43	8	average	Regression
11	9	8	2	3	1	2	57	9	average	

The above two records form unseen data were chosen to test the model

Step-1

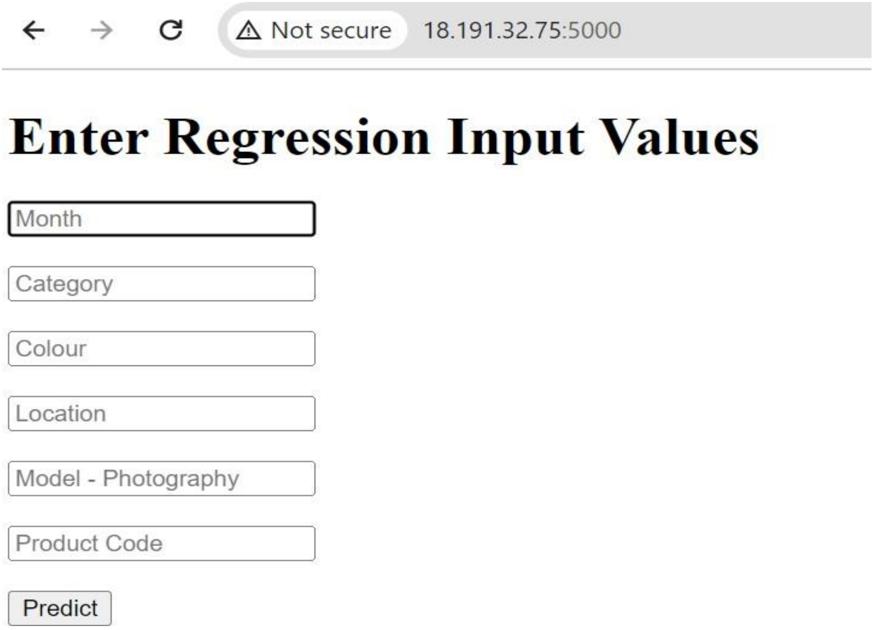


Select Model Type

● Regression

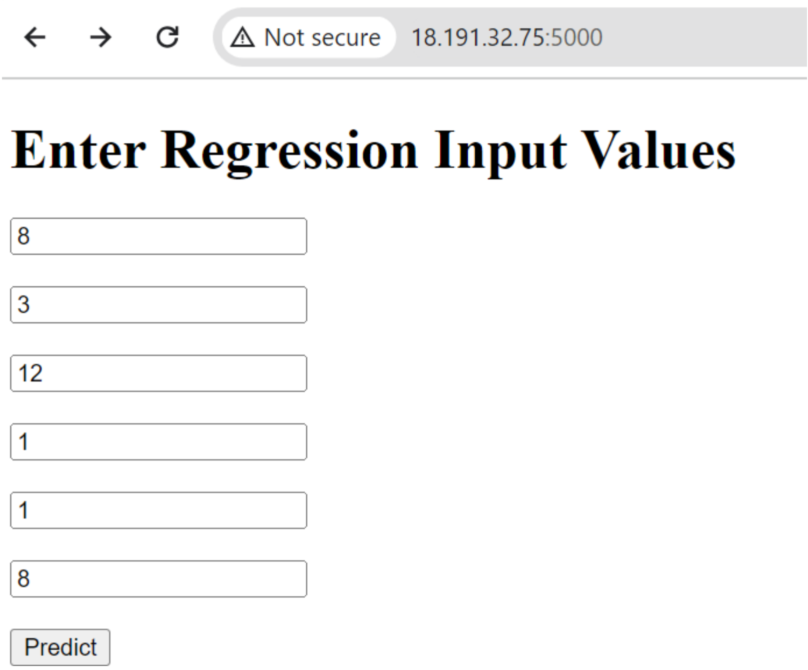
○ Classification

Step-2



Enter Regression Input Values

Step-3



Enter Regression Input Values

8

3

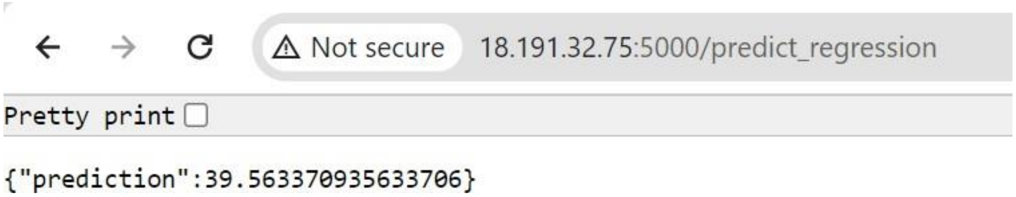
12

1

1

8

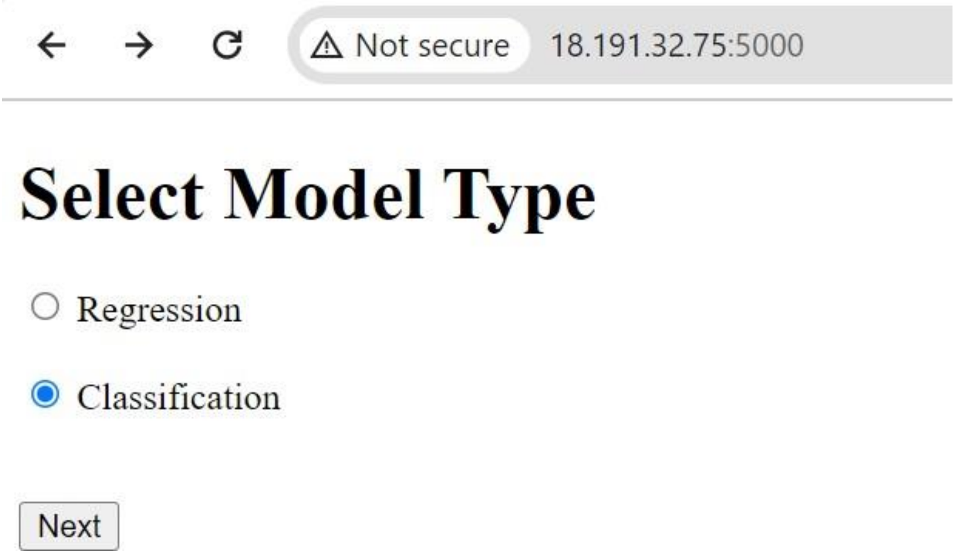
Step-4



```
{"prediction":39.563370935633706}
```

Classification Prediction

Step-1

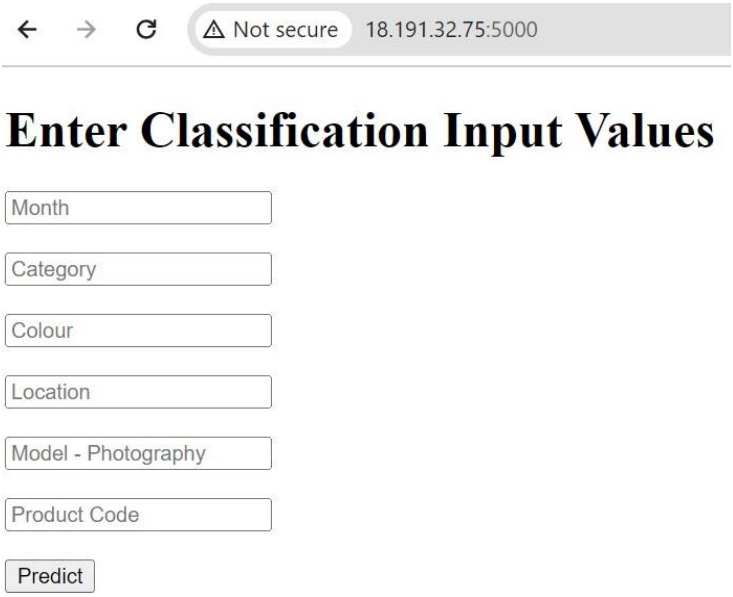


Select Model Type

○ Regression

● Classification

Step-2



Enter Classification Input Values

Month

Category

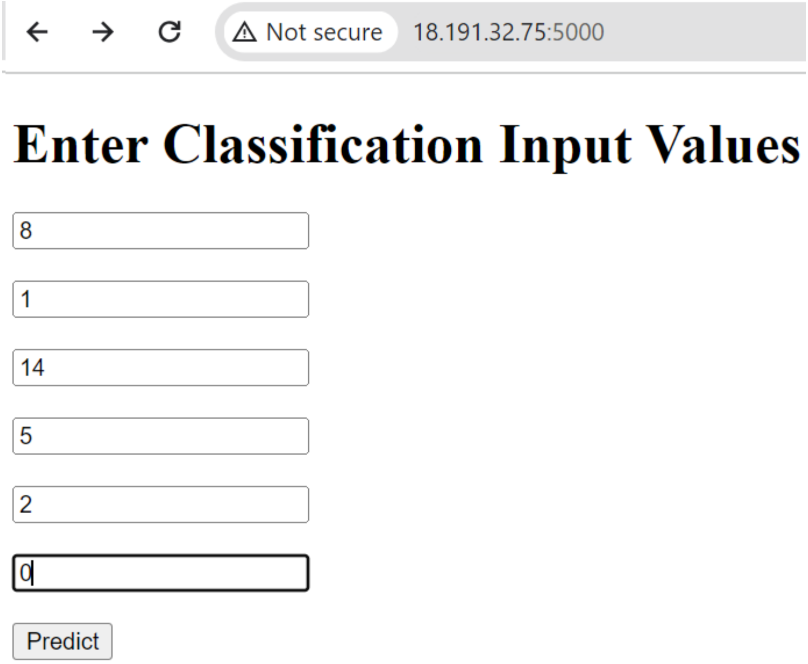
Colour

Location

Model - Photography

Product Code

Step-3



Enter Classification Input Values

8

1

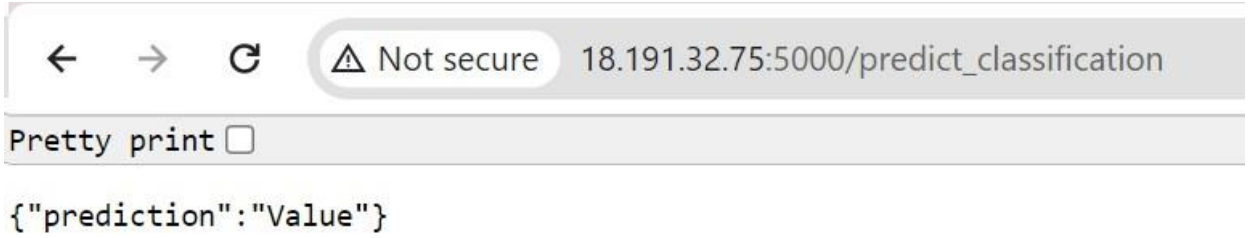
14

5

2

0

Step-4



```
{"prediction":"Value"}
```

Future Work:

- Feature Expansion – X^2, log(X) – automated feature engineering – for synergy/ interaction effect
- Enhance the User Experience for the GUI developed, i.e to decode the numericals
- Interactive Visualizations, as the data flows into the system
- Using AWS Lambda endpoint to have lifetime experience with API and GUI
- A/B Testing and Experimentation - to know about real world customer purchase pattern
- Further improve README.md – for more readability – HTML, CSS – clear picture

“No Free Lunch” theorem in machine learning states that no single machine learning algorithm is universally superior for all tasks