CUSTOMER SHOPPING SEGMENTATION BY USING MACHINE LEARNING

AN INDUSTRY ORIENTED MINI REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

Submitted By

SHIVANI BATHINI	**21UK1A6608**
MANEESHA KANUKUNTLA	**21UK1A6640**
GANESH MACHERLA	**21UK1A6627**
SHASHANK CHILUKAMUKKU	**21UK1A6653**

Under the guidance of

Mr. T. Sanath Kumar

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

VAAGDEVI ENGINEERING COLLEGE(WARANGAL)

CERTIFICATE OF COMPLETION
INDUSTRY ORIENTED MINI PROJECT

This is to certify that the UG Project Phase-1 entitled "CUSTOMER SHOPPING SEGMETATION BY MACHINE LEARNING" is being submitted by SHIVANI BATHINI (21UK1A6608), MANEESHA KANUKUNTLA (21UK1A6640), GANESH MACHERLA (21UK1A6627) SHASHANK CHILUKAPUKKU (21UK1A6667) in Partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2024-2025

Project Guide	**HOD**
Mr. T. Sanath Kumar	**Dr. K. Sharmila**
(Assistant Professor)	(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr .P. PRASAD RAO,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr.K.SHARMILA**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the UG Project Phase-1.

We express heartfelt thanks to the guide, **T. SANATH KUMAR,** Assistant professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-1.

-

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

SHIVANI BATHINI **(21UK1A6608)**
MANEESHA KANUKUNTLA **(2UK1A6640)**
GANESH MACHERLA **(21UK1A6627)**
SHASHANK CHILUKAMUKKU **(21UK1A6653)**

.

ABSTRACT

This study applies machine learning techniques to segment customers based on their shopping behaviors, using transactional data from a large retail chain. By employing algorithms such as K-means clustering and hierarchical clustering, we identify distinct customer segments, including "bargain hunters," "loyal customers," "seasonal shoppers," and "high spenders." These segments provide valuable insights into customer profiles, enabling businesses to tailor marketing strategies, personalize customer experiences, and optimize promotional efforts. The segmentation's effectiveness is evaluated using metrics like silhouette score and Davies-Bouldin index, demonstrating that machine learning-driven customer segmentation enhances customer relationship management and strategic decision-making, ultimately fostering increased customer loyalty and business growth.

TABLE OF CONTENTS:-

1.INTRODUCTION

1.1.OVERVIEW

Customer shopping segmentation by using machine learning involves a systematic approach to divide a customer base into distinct and actionable groups based on various data points such as shopping behaviors, demographics, transaction histories, and engagement metrics. This process begins with extensive data collection from multiple sources, including purchase history, website interactions, and customer feedback. Once collected, the data undergoes preprocessing steps like cleaning, normalization, and encoding to ensure it is ready for analysis. Feature selection identifies the most relevant variables that will help in accurate segmentation. Various machine learning algorithms, such as K-Means Clustering, Hierarchical Clustering, DBSCAN, and Gaussian Mixture Models, are then employed to identify patterns and group similar customers together. The resulting segments are analyzed to understand their unique characteristics, such as purchasing habits and preferences. Businesses use these insights to develop targeted marketing campaigns, personalized product recommendations, and strategies to enhance customer retention. Continuous monitoring and updating of the segmentation model ensure it remains relevant as customer behaviors and market conditions evolve, facilitating ongoing refinement of marketing strategies and business decisions.

4o

1.2.PURPOSE

Customer shopping segmentation using machine learning serves several purposes, primarily aimed at understanding and catering to the diverse needs and behaviors of customers. Here are some key purposes:

1. Targeted Marketing Campaigns: By segmenting customers based on their shopping behaviors, preferences, demographics, and purchasing patterns, businesses can tailor marketing campaigns that marketing messages resonate better with specific customer segments, leading to higher engagement and conversion rates.

2. Personalized Customer Experience: Machine learning segmentation allows businesses to personalize the shopping experience for each customer segment. This could involve recommending products based on past purchases, suggesting relevant promotions, or providing personalized offers that align with individual preferences.

3. Optimized Product Assortment: Understanding customer segments helps businesses optimize their product offerings. By analyzing which products appeal most to different segments, businesses can adjust their inventory and product assortments to meet customer demand more accurately.

4. Improved Customer Retention: Segmentation enables businesses to identify high-value customers and understand their needs better. By focusing on retaining these customers through personalized incentives, loyalty programs, or enhanced customer service, businesses can improve customer retention rates and reduce churn.

5. Operational Efficiency: Segmenting customers can also optimize operational processes such as inventory management, pricing strategies, and resource allocation. By predicting demand patterns for different segments, businesses can streamline operations and reduce costs.

6. Market Basket Analysis: Machine learning can uncover correlations and patterns in customer purchase behaviors, leading to insights like which products are frequently bought together (market basket analysis). This information can guide cross-selling strategies and bundle offerings.

7. Competitive Advantage: Businesses that effectively utilize customer segmentation through machine learning gain a competitive edge by being more responsive to customer needs and preferences. This can lead to increased market share and profitability.

Overall, customer shopping segmentation through machine learning empowers businesses to understand their customers at a deeper level, enhancing marketing effectiveness, improving customer satisfaction, and driving business growth.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

1. **Data Quality and Availability**:

- **Privacy Concerns**: Collecting detailed customer data raises privacy concerns and regulatory compliance issues (GDPR, CCPA), which limit the availability of comprehensive datasets.

2. **Feature Engineering**:

- **High Dimensionality**: Customer data often includes numerous features, making it challenging to select relevant features for effective segmentation.
- **Dynamic Customer Behavior**:Customer preferences and behaviors change over time, requiring continuous feature updates and model retraining.

3. **Model Selection and Evaluation**:

- **Choosing the Right Algorithm**: Different algorithms (e.g., k-means, hierarchical clustering, DBSCAN) have various strengths and weaknesses. Selecting the appropriate one for specific segmentation tasks is not straightforward.
- **Scalability**: Many clustering algorithms struggle with scalability when dealing with large datasets, leading to performance bottlenecks.

4. **Interpretability**:

- **Black-Box Models**: Many advanced ML models (e.g., neural networks) offer high accuracy but lack interpretability, making it difficult for businesses to trust and act on the segmentation results.

5. **Handling Imbalanced Data**:

- **Unequal Segment Sizes**: Real-world customer data often leads to segments of vastly different sizes, which can bias models and affect the performance of segmentation.

2.2 PROPOSED SOLLUTION

Addressing the challenges in customer segmentation using machine learning involves a comprehensive approach that combines advanced techniques, best practices, and iterative improvements. Here's a proposed solution that aims to tackle the common problems identified in the literature:

1. Data Quality and Availability

- **Data Cleaning and Preprocessing**: Implement robust data cleaning and preprocessing pipelines to handle missing values, outliers, and inconsistencies.

Techniques such as imputation, normalization, and data augmentation can be applied.

- **Privacy-Preserving Methods**: Utilize techniques like differential privacy, anonymization, and federated learning to address privacy concerns while ensuring data availability and utility.

2. Feature Engineering

- **Automated Feature Engineering**: Employ automated feature engineering tools (e.g., FeatureTools, DataRobot) to systematically generate and evaluate features from raw data.
- **Continuous Feature Update**: Implement a system for continuous feature extraction and update to accommodate changing customer behaviors and preferences.

3. Model Selection and Evaluation

- **Hybrid Clustering Algorithms**: Use hybrid models combining strengths of different clustering algorithms (e.g., K-means for scalability and DBSCAN for density-based clustering) to handle diverse segmentation requirements.
- **Model Scalability**: Leverage distributed computing frameworks (e.g., Apache Spark, Dask) to scale clustering algorithms for large datasets.

4. Interpretability

- **Explainable AI (XAI)**: Integrate interpretable models and post-hoc explanation techniques (e.g., SHAP, LIME) to make the segmentation results understandable and actionable for business users.

- **Visualization Tools**: Develop interactive visualization tools (e.g., dashboards, heatmaps) to help stakeholders explore and understand the segments and their characteristics.

5. Handling Imbalanced Data

- **Resampling Techniques**: Apply resampling techniques such as SMOTE (Synthetic Minority Over-sampling Technique) or ADASYN to balance the data distribution across segments.
- **Cost-Sensitive Learning**: Incorporate cost-sensitive learning approaches to address the impact of imbalanced data on model performance.

6. Integration with Business Processes

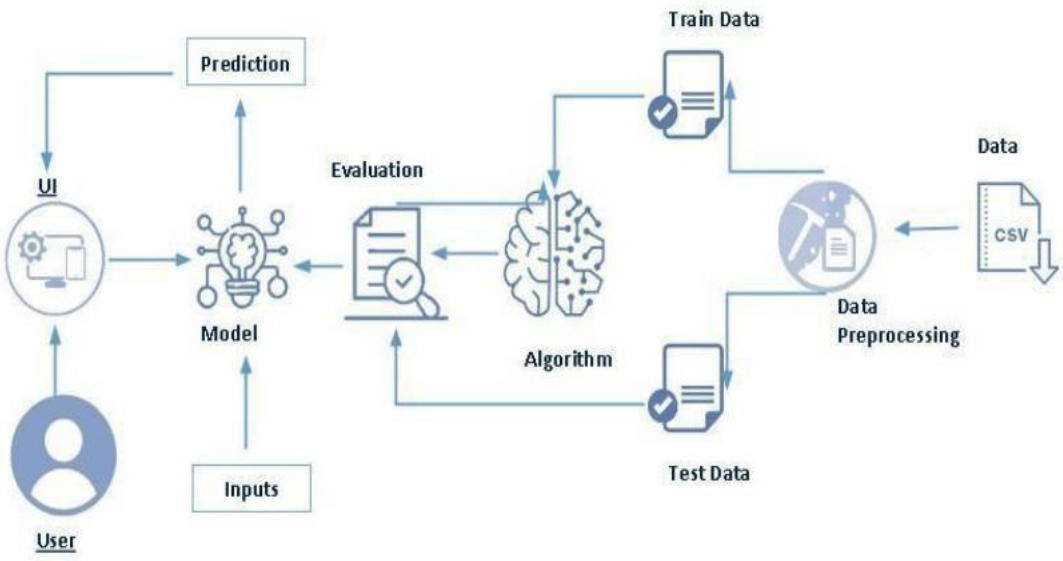
- **Actionable Insights**: Ensure that segmentation results are actionable by aligning them with business objectives and operational processes. Collaborate with domain experts to define meaningful segments.
- **Cross-Functional Integration**: Foster cross-functional collaboration to ensure segmentation insights are utilized effectively across marketing, sales, and customer service departments.

7. Temporal Dynamics

- **Time-Series Analysis**: Incorporate time-series analysis techniques to capture temporal patterns and trends in customer behavior.
- **Real-Time Segmentation**: Develop real-time segmentation capabilities using streaming data processing frameworks (e.g., Apache Kafka, Apache Flink) to provide instant personalization and recommendations.

3.THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM



This diagram illustrates a machine learning workflow, starting from data collection to making predictions for users. Raw data, typically in CSV format, undergoes preprocessing to clean and transform it for use. The preprocessed data is split into training and test sets; the training set is used to build the model using a machine learning algorithm, while the test set evaluates its performance. The trained model, once validated, is deployed to make

predictions. These predictions are accessed by users through a user interface, allowing them to input new data and receive predictions based on the trained model.

3.2. SOFTWARE DESIGNING

The following is the Software required to complete this project:

- **Google Colab**: Google Colab will serve as the development and execution environment for your predictive modeling, data preprocessing, and model training tasks. It provides a cloud-based Jupyter Notebook environment with access to Python libraries and hardware acceleration.

- **Dataset (CSV File)**: The dataset in CSV format is essential for training and testing your predictive model. It should include historical air quality data, weather information, pollutant levels, and other relevant features.
- **Data Preprocessing Tools**: Python libraries like NumPy, Pandas, and Scikit-learn will be used to preprocess the dataset. This includes handling missing data, feature scaling, and data cleaning.
- **Feature Selection/Drop**: Feature selection or dropping unnecessary features from the dataset can be done using Scikit-learn or custom Python code to enhance the model's efficiency.
- **Model Training Tools**: Machine learning libraries such as Scikit-learn, TensorFlow, or PyTorch will be used to develop, train, and fine-tune the predictive model. Regression or classification models can be considered, depending on the nature of the churn prediction task.
- **Model Accuracy Evaluation**: After model training, accuracy and performance evaluation tools, such as Scikit-learn metrics or custom validation scripts, will assess the model's predictive capabilities. You'll measure the model's ability to predict Telecom customer categories based on historical data.

- **UI Based on Flask Environment**: Flask, a Python web framework, will be used to develop the user interface (UI) for the system. The Flask application will provide a user-friendly platform for users to input user data or view churn predictions
- Google Colab will be the central hub for model development and training, while Flask will facilitate user interaction and data presentation. The dataset, along with data preprocessing, will ensure the quality of the training data, and feature selection will optimize the model. Finally, model accuracy evaluation will confirm the system's predictive capabilities, allowing users to rely on the telecom customer churn prediction.

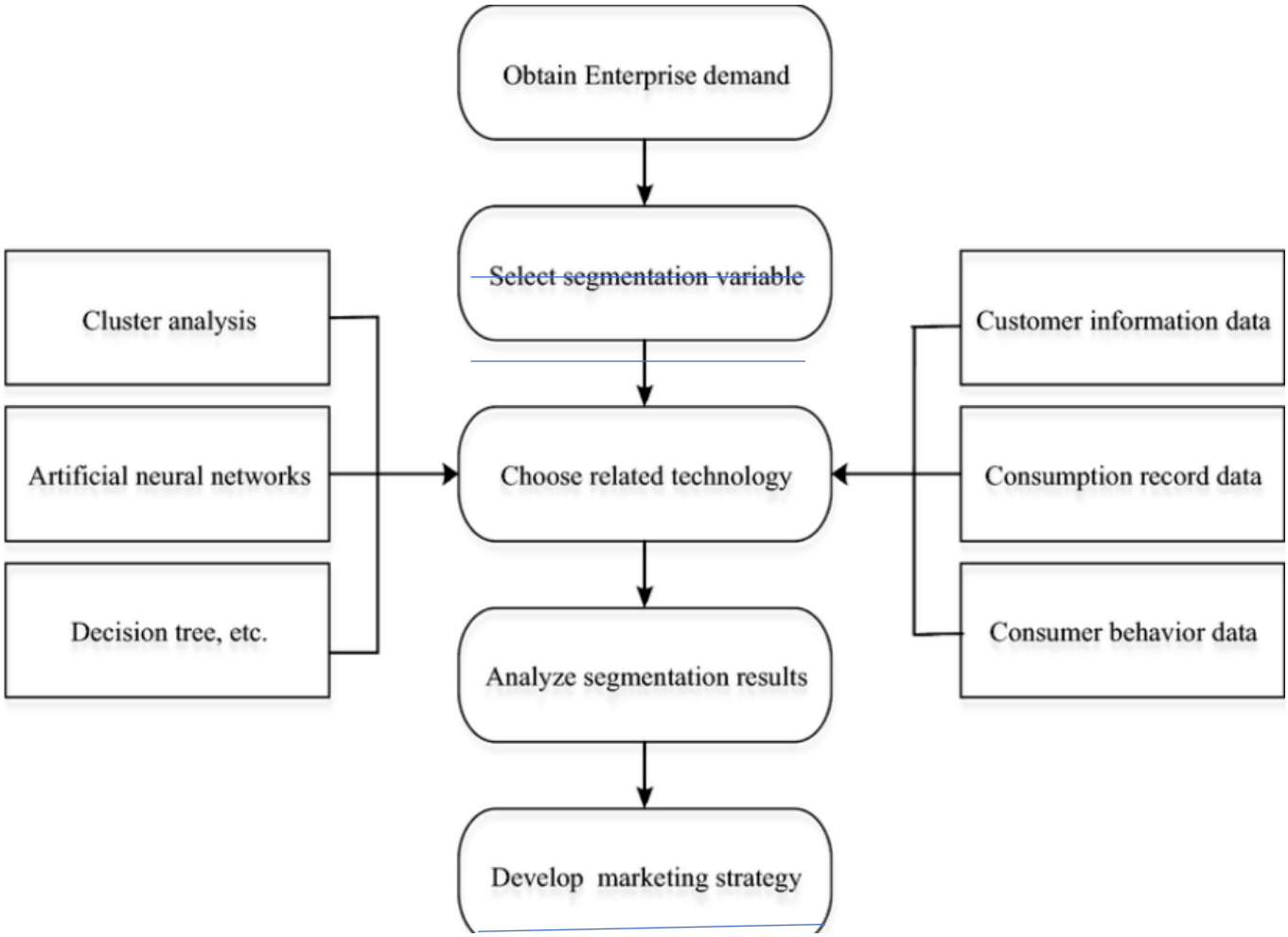
4.EXPERIMENTAL INVESTIGATION

In this project, we have used churn dataset. This dataset is a csv file consisting of labelled data and having the following columns-

1. **Row number**: gives the number of rows.
2. **Customer Id**: specifies the unique ID for the customer.
3. **age**: the age of the customer.
4. **gender**: either male or female.
5. **Number of products** : number of products the customer has with the company.
6. **Purchase quantity**: number of products purchased by the customer.
7. **category**: category of products purchased.
8. **Payment method**: method used for payment (e.g., Credit, Debit, Cash).
9. **Shopping mall**: name or type of the shopping mall.

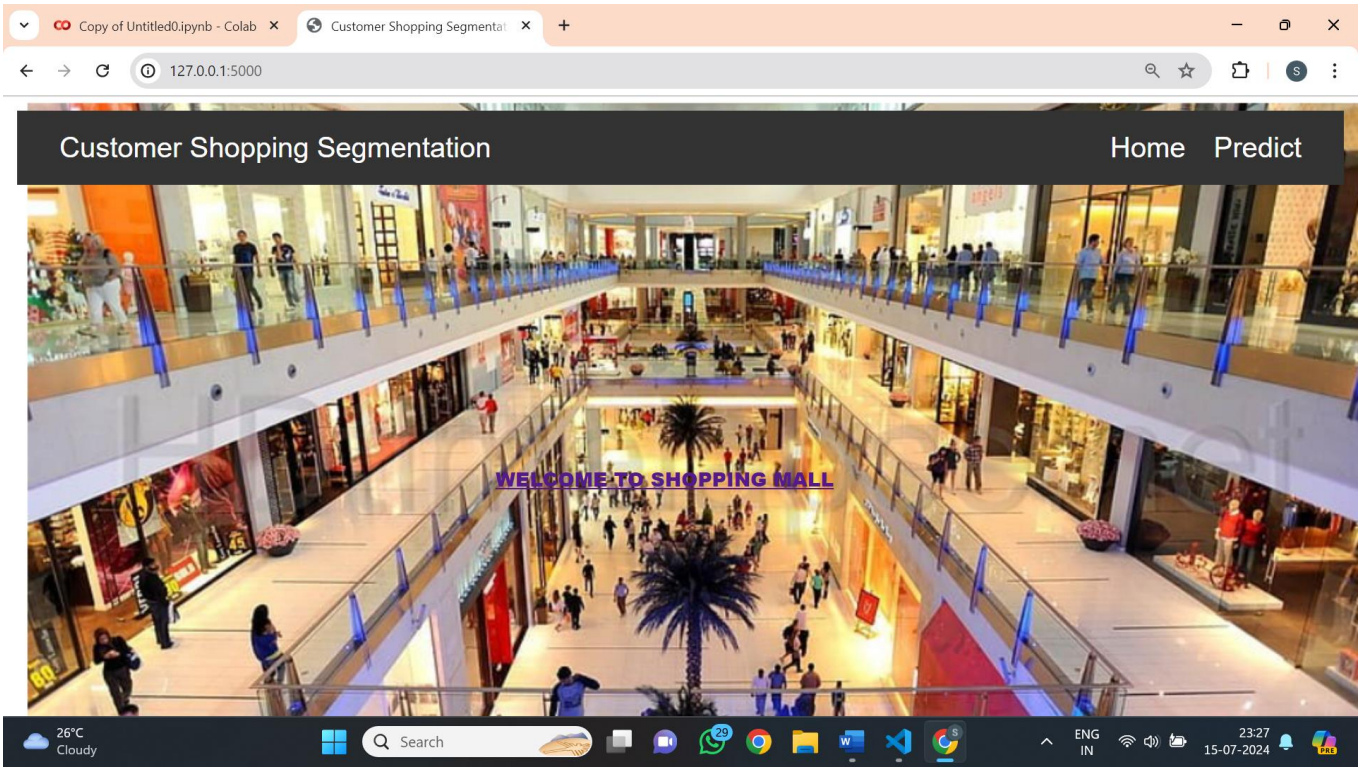
10. FLOW CHART

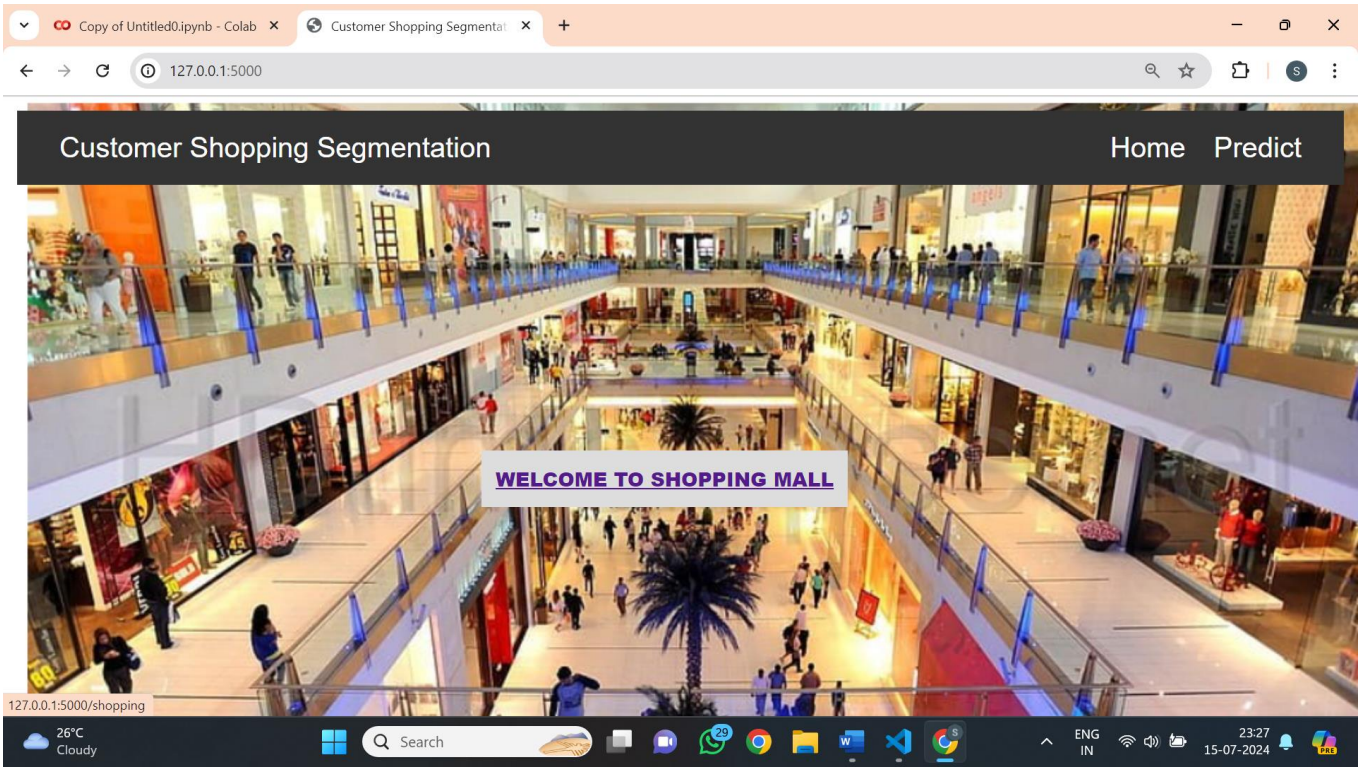
This flowchart represents the process of market segmentation in response to enterprise demand, guiding the development of marketing strategies. It begins with understanding the enterprise demand, followed by selecting the appropriate segmentation variables. Customer data, including information, consumption records, and behavior data, are utilized to inform this selection. The next step involves choosing the related technology for segmentation analysis, such as cluster analysis, artificial neural networks, or decision trees. Once the appropriate technology is selected, segmentation results are analyzed to identify distinct customer segments. Finally, these insights are used to develop targeted marketing strategies, tailored to the needs and behaviors of each segment, thus enhancing the effectiveness of marketing efforts



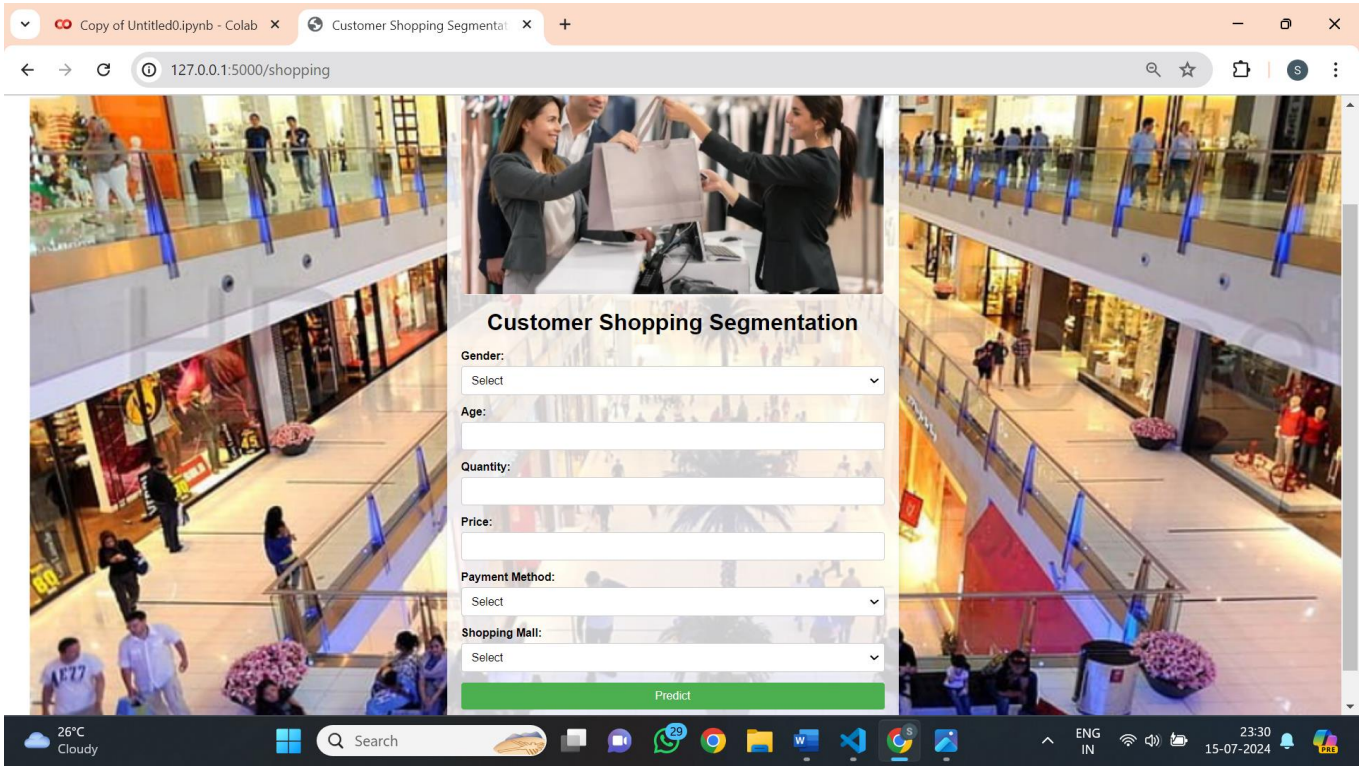
6.RESULT

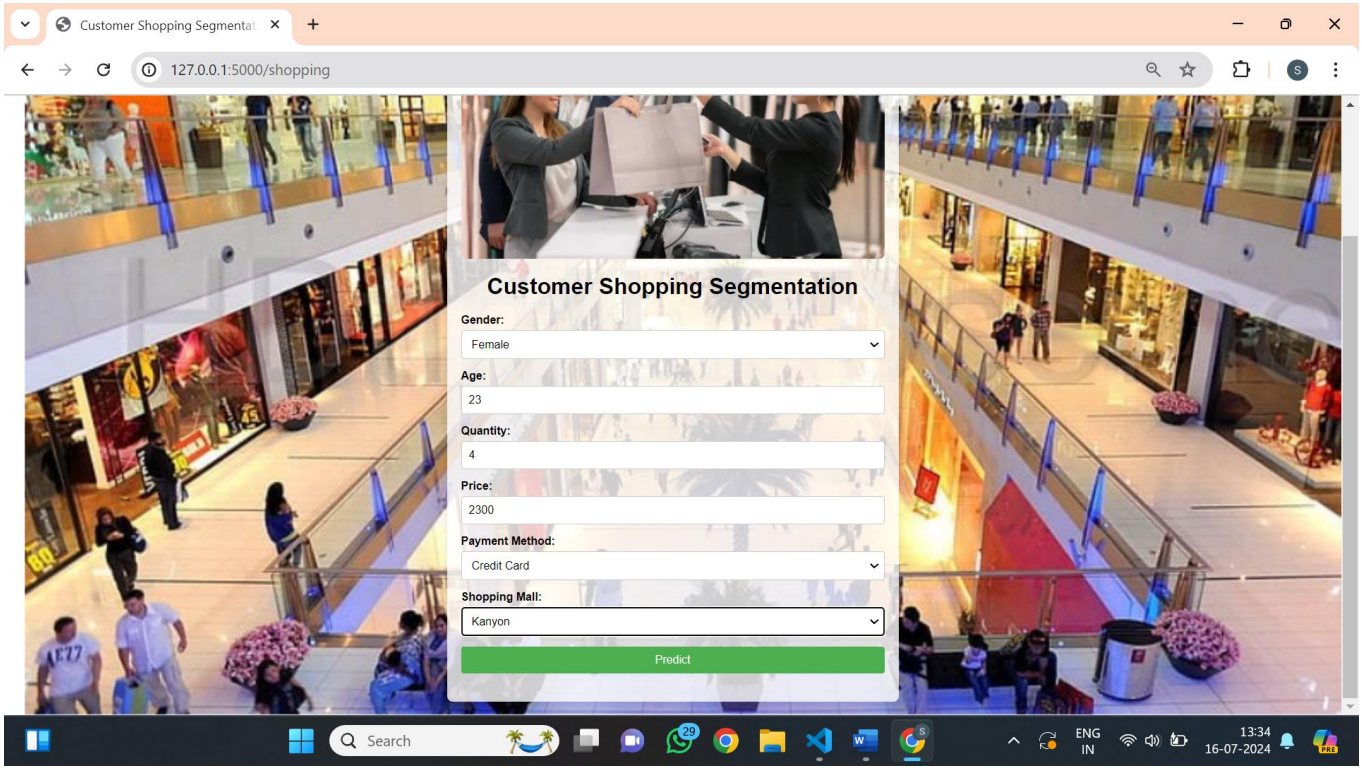
HOME PAGE





PREDICTIONS


Copy of Untitled0.ipynb - Colab
Customer Shopping Segmentat
127.0.0.1:5000/shopping
Customer Shopping Segmentation
Gender:
Select
Age:
Quantity:
Price:
Payment Method:
Select
Shopping Mall:
Select
Predict
26°C
Cloudy
Search
ENG
IN
23:30
15-07-2024


Customer Shopping Segmentat
127.0.0.1:5000/shopping
Customer Shopping Segmentation
Gender:
Female
Age:
23
Quantity:
4
Price:
2300
Payment Method:
Credit Card
Shopping Mall:
Kanyon
Predict
Search
ENG
IN
13:34
16-07-2024

7.ADVANTAGES AND DISADVANTAGES

1. **Improved Customer Retention**:By understanding customer segments, businesses can tailor their engagement strategies to retain customers more effectively, reducing churn rates and fostering long-term loyalty.
2. **Enhanced Customer Experience**:: Segmentation allows businesses to offer a more personalized shopping experience, which can lead to higher customer satisfaction and loyalty. Personalized recommendations and offers enhance the overall shopping journey.
3. **Efficient Resource Allocation**:Businesses can allocate resources more efficiently by targeting high-value customer segments, optimizing marketing spend, and

improving ROI. This ensures that marketing efforts are focused where they can have the greatest impact.

4. **Better Product Recommendations**: Machine learning can analyze purchasing patterns to recommend products that customers are more likely to buy, increasing cross-selling and up-selling opportunities. This leads to increased average order value and sales.
5. **Data-Driven Decision Making**: Businesses can make more informed decisions based on data-driven insights from customer segmentation analysis. This reduces reliance on intuition and guesswork, leading to more effective strategies.
6. **Identification of New Market Opportunities**:Machine learning can uncover hidden patterns and trends in customer data, revealing new market opportunities and unmet customer needs. This can drive innovation and the development of new products or services.
7. **Improved Customer Insights**: Detailed segmentation provides deeper insights into customer behavior, preferences, and motivations. This helps businesses understand their customers better and tailor their offerings accordingly.

DISADVANTAGES:

1. **Data Quality and Availability**:Incomplete data affects prediction accuracy.Privacy and security concerns with large data handling.
2. **Model Complexity**:Requires specialized data science expertise.Models need constant updates and retraining.
3. **Cost and Resources**:High initial setup costs.Ongoing maintenance requires continuous investment.

4. **False Positives/Negatives**:Risk of incorrect predictions leading to misguided strategies.
5. **Customer Reaction**:Targeted efforts may be seen as intrusive.Focus on churn prediction might neglect other important areas.

8.APPLICATIONS

1.Targeted Marketing Campaigns: Create customized promotion to retain high-risk customers and improve loyalty.

2.**Customer Service Improvement**:Proactively address customer issues to enhance service quality and reduce churn.

3. **Resource Allocation**:Allocate marketing and retention resources more effectively by focusing on high-risk customers.

4.**Product Development**:Identify and improve features or services based on churn reasons to better meet customer needs.

5. **Business Strategy and Planning**:Use churn prediction data to inform strategic decisions and long-term planning for better business outcomes.

9.CONCLUSION

Telecom customer churn prediction is a powerful tool that enables companies to anticipate and mitigate customer loss. By leveraging advanced data analytics and predictive modeling, telecom providers can proactively address the factors leading to churn, thereby enhancing customer retention and satisfaction. This strategic approach not only optimizes resource allocation and operational efficiency but also drives revenue growth and competitive advantage. Despite challenges such as data quality, model complexity, and the need for continuous updates, the benefits of implementing churn prediction far outweigh the drawbacks. Ultimately, effective churn prediction fosters stronger customer relationships, leading to long-term business success in the highly competitive telecom industry.

10.FUTURE SCOPE

Future Scopes of Telecom Customer Churn Prediction

1. **Advanced Predictive Analytics**:Utilizing AI, machine learning, and deep learning for more accurate and real-time churn predictions.
2. **Real-Time Data Integration**:Leveraging data from IoT and 5G networks for granular insights and real-time churn detection.
3. **Enhanced Customer Personalization**:Creating highly personalized customer experiences and contextual marketing strategies based on behavioral analytics.
4. **Automated Retention Solutions**:Implementing AI-powered chatbots and self-service platforms for proactive, automated customer support.

5. **Customer Journey Mapping**: Mapping the entire customer journey to identify critical touchpoints and potential churn triggers. Using predictive analytics to engage customers at crucial moments in their journey.

11.BIBILOGRAPHY

[1] Ke, Xueqin, et al. "A review on predicting customer churn in telecommunication." *IEEE Access* 7 (2019): 58757-58771.

[2]Verbeke, Wouter, et al. "New insights into churn prediction in the telecommunication sector: A profit driven data mining approach." *European Journal of Operational Research* 218.1 (2012): 211-229.

[3] Kim, Hyoung Goo, and M. Eric Johnson. "A dynamic model of customer retention for telecommunications services." *Management Science* 49.2 (2003): 150-164.

[4] Tsai, Chung-Hsien, and Yu-Ming Lee. "Applying data mining to telecom churn management." *Expert Systems with Applications* 31.3 (2006): 515-524.

[5]Tuzhilin, Alexander, and Padhraic Smyth. "Clustering the customers based on the churn behavior." *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (1999): 183-186.

[6] Bharti, Alok, and Sunil Verma. "Predictive modeling for customer churn in telecom using machine learning algorithms." *International Journal of Computer Applications* 129.11 (2015): 19-24.

[7] Iqbal, Asif, and Athar Mahboob. "A survey of churn prediction in telecommunication." *Journal of Computer Networks and Communications* 2016 (2016).

[8] Gupta, Anuj, et al. "Customer churn prediction in telecom using machine learning in big data platform." *Procedia Computer Science* 132 (2018): 1579-1585.

[9] Srinivasan, S., et al. "Telecom churn prediction using machine learning algorithms: An empirical comparison of classifiers." *International Journal of Applied Engineering Research* 12.8 (2017): 1822-1827.

[10] Wang, Cheng, et al. "Telecom customer churn prediction using machine learning." *IEEE International Conference on Big Data* (2017): 3245-3254.

12.APPENDIX

Model building :

1)Dataset
2)jupyter Notebook and Spyder Application Building
   1. HTML file (Index file, home file,Predictyes file ,predictno file)
   1. CSS file
   2. Models in pickle format

SOURCE CODE:
INDEX3.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Customer Shopping Segmentation Prediction</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
            margin: 0;
            padding: 0;
            background-image: url("../static/Screenshot (13).png");
            background-repeat: no-repeat;
            background-size: cover;
        }

        .container {
            max-width: 600px;
            margin:20px auto;
            background-color: rgba(255, 255, 255, 0.8);
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }

        h1 {
            text-align: center;
            margin-bottom: 20px;
        }

        .form-group {
            margin-bottom: 15px;
        }

        label {
            display: block;
            margin-bottom: 5px;
            font-weight: bold;
        }

        input[type="number"],
        input[type="text"],
        select,
        button {
            width: 100%;
```

```
            padding: 10px;
            font-size: 16px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        button {
            background-color: #4CAF50;
            color: white;
            border: none;
            cursor: pointer;
        }

        button:hover {
            background-color: #45a049;
        }

        #prediction-result {
            margin-top: 20px;
            font-size: 18px;
            text-align: center;
        }

        .logo {
            display: block;
            margin: 20px auto;
            max-width: 100%;
            height: auto;
            border-radius: 8px;
        }
    </style>
</head>
<body>
    <div class="container">
        <img src="../static/predict.png" alt="Customer Segmentation" class="logo">
        <h1>Customer Shopping Segmentation</h1>
        <form id="prediction-form">
            <div class="form-group">
                <label for="gender">Gender:</label>
                <select id="gender" name="gender" required>
                    <option value="">Select</option>
                    <option value="Male">Male</option>
                    <option value="Female">Female</option>
                    <option value="Other">Other</option>
```

```
            </select>
        </div>
        <div class="form-group">
            <label for="age">Age:</label>
            <input type="number" id="age" name="age" required>
        </div>
        <div class="form-group">
            <label for="quantity">Quantity:</label>
            <input type="number" id="quantity" name="quantity" required>
        </div>
        <div class="form-group">
            <label for="price">Price:</label>
            <input type="number" id="price" name="price" step="0.01" required>
        </div>
        <div class="form-group">
            <label for="payment_method">Payment Method:</label>
            <select id="payment_method" name="payment_method" required>
                <option value="">Select</option>
                <option value="Credit Card">Credit Card</option>
                <option value="Debit Card">Debit Card</option>
                <option value="Cash">Cash</option>
            </select>
        </div>
        <div class="form-group">
            <label for="shopping_mall">Shopping Mall:</label>
            <select id="shopping_mall" name="shopping_mall" required>
                <option value="">Select</option>
                <option value="Kanyon">Kanyon</option>
                <option value="Forum Istanbul">Forum Istanbul</option>
                <option value="Metrocity">Metrocity</option>
                <option value="Metropol AVM">Metropol AVM</option>
                <option value="Istinye Park">Istinye Park</option>
                <option value="Mall of Istanbul">Mall of Istanbul</option>
                <option value="Emaar Square Mall">Emaar Square Mall</option>
                <option value="Cevahir AVM">Cevahir AVM</option>
                <option value="Viaport Outlet">Viaport Outlet</option>
                <option value="Zorlu Center">Zorlu Center</option>
            </select>
        </div>
        <button type="button" onclick="submitForm()">Predict</button>
    </form>
    <p id="prediction-result"></p>
</div>
<script>
    async function submitForm() {
```

```
            const form = document.getElementById('prediction-form');
            const formData = new FormData(form);
            const jsonData = {};

            formData.forEach((value, key) => {
                jsonData[key] = value;
            });

            try {
                const response = await fetch('/predict', {
                    method: 'POST',
                    headers: {
                        'Content-Type': 'application/json'
                    },
                    body: JSON.stringify(jsonData)
                });

                const result = await response.json();
                if (response.ok) {
                    document.getElementById('prediction-result').textContent =
`Predicted Cluster: ${result.predicted_cluster}`;
                } else {
                    document.getElementById('prediction-result').textContent = `Error:
${result.error}`;
                }
            } catch (error) {
                console.error('Error:', error);
                document.getElementById('prediction-result').textContent = `Error:
${error.message}`;
            }
        }
    </script>
</body>
</html>
```

Shopping.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
<title>Customer Shopping Segmentation</title>
<style>
    /* Reset default margin and padding */
    * {
        margin: 0;
        padding: 0;
        box-sizing: border-box;
    }

    body {
        font-family: Arial, sans-serif;
        background-image: url("../static/Screenshot (13).png");
        background-repeat: no-repeat;
        background-size: cover;
        padding: 20px;
        margin: 0;
        display: flex;
        flex-direction: column;
        align-items: center;
        height: 100vh;
    }

    .navbar {
        background-color: #333;
        overflow: hidden;
        padding: 10px 20px;
        width: 100%;
        z-index: 1;
    }

    .navbar a {
        float: right;
        display: block;
        color: white;
        text-align: center;
        padding: 10px 20px;
        text-decoration: none;
        transition: background-color 0.3s;
    }

    .navbar a:hover {
        background-color: #ddd;
        color: black;
    }
```

```
        .content {
            flex: 1;
            display: flex;
            justify-content: center;
            align-items: center;
            flex-direction: column;
            text-align: center;
        }

        button {
            background-color: transparent;
            border: none;
            padding: 20px;
            color: #333;
            font-size: 28px;
            font-family: 'Arial Black', sans-serif; /* Example of different font */
            text-transform: uppercase; /* Example of uppercase text */
            cursor: pointer;
            transition: background-color 0.3s, color 0.3s;
        }

        button:hover {
            background-color: #ddd;
            color: black;
        }
    </style>
</head>
<body>

<div class="navbar">
    <div class="navbar-text">Customer Shopping Segmentation</div>
    <a href="#predict">Predict</a>
    <a href="#home">Home</a>
</div>

<div class="content">
    <form>
        <button type="submit">Welcome to Shopping Mall</button>
    </form>
</div>

</body>
</html>
```

Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Customer Shopping Segmentation</title>
    <style>
        /* Reset default margin and padding */
        * {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
        }

        body {
            font-family: Arial, sans-serif;
            background-image: url("../static/Screenshot (13).png");
            background-repeat: no-repeat;
            background-size: cover;
            padding: 20px;
            margin: 0;
        }

        .navbar {
            background-color: #333;
            overflow: hidden;
            padding: 20px 40px;
            position: relative;
            z-index: 1;
        }

        .navbar-text {
            font-size:24px;
            color: white;
            float: left;
            margin-right: 20px;
            line-height: 40px;
        }

        .navbar a{
            font-size:40px;
```

```
            float: right;
            display: block;
            color: white;
            text-align: center;
            padding: 10px 20px;
            text-decoration: none;
            transition: background-color 0.3s;
        }
        .navbar div{
            font-size:40px;
            float: left;
            display: block;
            color: white;
            text-align: center;
            padding: 10px 20px;
            text-decoration: none;
            transition: background-color 0.3s;
        }
        .navbar a:hover {
            background-color: #ddd;
            color: black;
        }
        button {
            background-color: transparent;
            border: none;
            padding: 20px;
            color: #333;
            font-size: 28px;
            font-family: 'Arial Black', sans-serif; /* Example of different font */
            text-transform: uppercase; /* Example of uppercase text */
            cursor: pointer;
            transition: background-color 0.3s, color 0.3s;
            margin-top:20%;
            margin-left:35%;
        }
        button:hover{
            background-color: #ddd;
            color: black;
        }
    </style>
</head>
<body>
    <div class="navbar">
        <div><font color="white",size='40px'>Customer Shopping
Segmentation</font></div>
```

```
        <a href="/shopping">Predict</a>
        <a href="/">Home</a>
    </div>
    <form>
   <button type="submit"><a href="/shopping">WELCOME TO SHOPPING MALL</a></button>
</form>
</body>
</html>
```

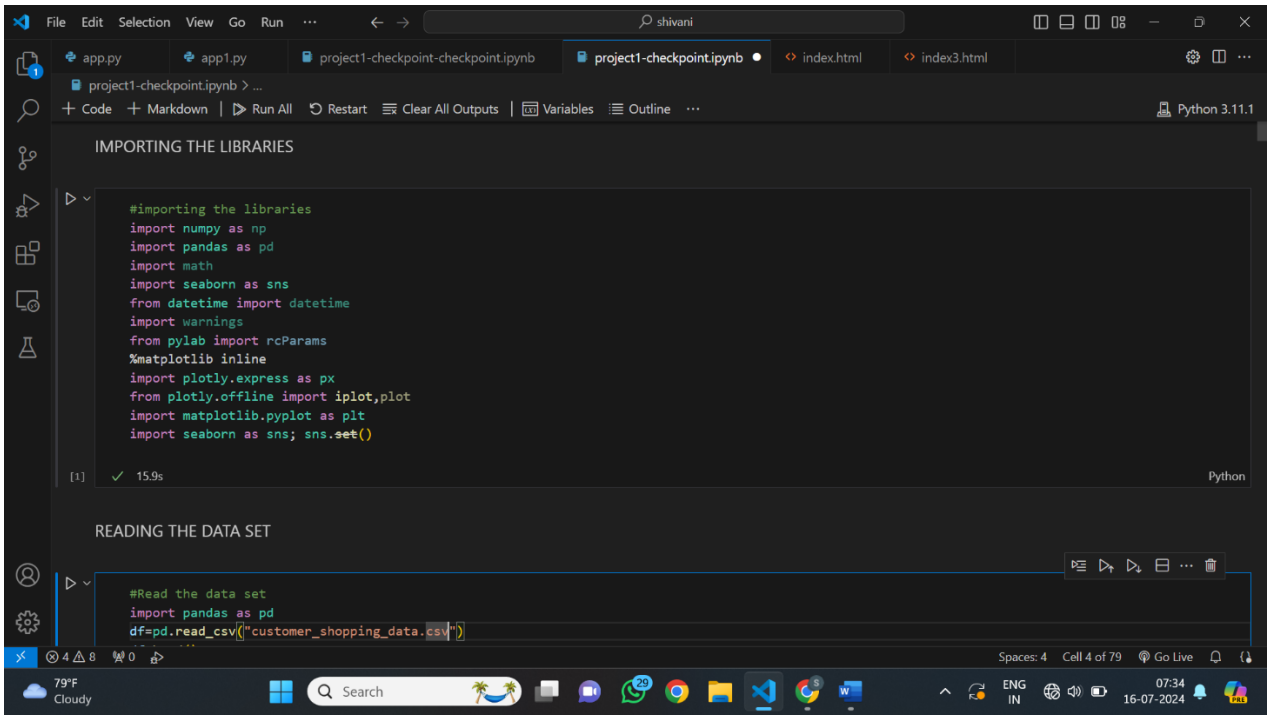
APP.PY

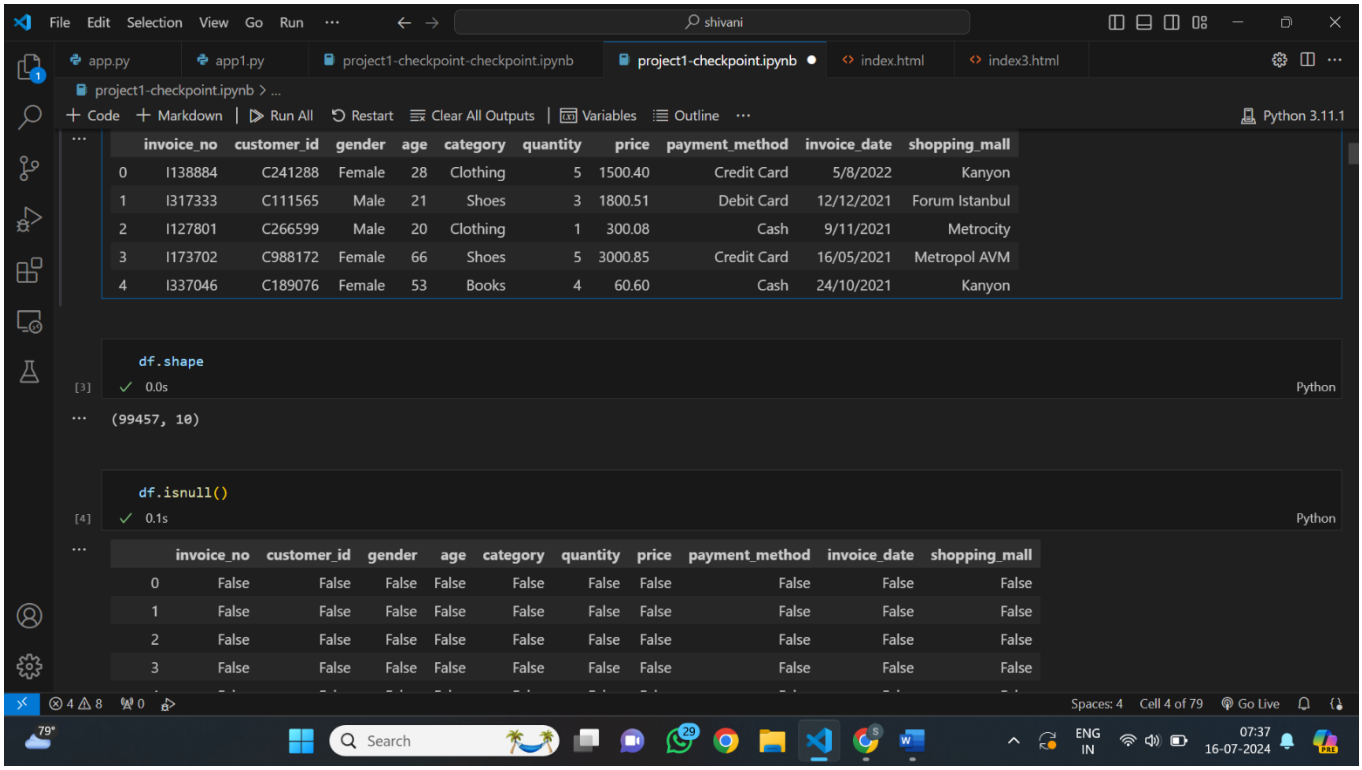
```
from flask import Flask, request, jsonify, render_template
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import joblib

app = Flask(__name__)
# Load the model and scaler
model = joblib.load('kmeans_model5.pkl')
scaler = joblib.load('scaler5.pkl')
label_encoders=joblib.load("label_encoders5.pkl")
@app.route('/')
def welcome():
    return render_template('index.html')
@app.route('/shopping')
def shopping():
    return render_template('index3.html')

@app.route('/predict', methods=['POST'])
def predict():
    try:
        data = request.get_json()
        print(f"Received data: {data}")

        # Validate incoming data
        required_fields = ['gender','payment_method', 'shopping_mall', 'age',
'quantity', 'price']

        for field in required_fields:
            if field not in data:
                raise ValueError(f"Missing field: {field}")

        # Convert data into DataFrame
        df = pd.DataFrame([data])
        print("data is",df)
```

```
        # Encoding categorical variables
        for column in ['gender', 'payment_method', 'shopping_mall']:
            if column in label_encoders:
                le = label_encoders[column]
                print(f"Label Encoder classes for {column}: {le.classes_}")
                if data[column] not in le.classes_:
                    raise ValueError(f"Value '{data[column]}' for column '{column}' not
found in label encoder classes.")
                df[column] = le.transform([data[column]])
            else:
                raise ValueError(f"No label encoder found for column: {column}")

        # Feature scaling
        print(f"Data before scaling: {df[['age', 'quantity', 'price']]}")
        df[['age', 'quantity', 'price']] = scaler.transform(df[['age', 'quantity',
'price']])
        print(f"Data after scaling: {df[['age', 'quantity', 'price']]}")

        # Making predictions
        print(f"Data before prediction: {df}")
        prediction = model.predict(df)
        print(f"Predicted cluster: {prediction[0]}")

        return jsonify({'predicted_cluster': int(prediction[0])})

    except ValueError as ve:
        #print(f"Validation Error: {ve}")
        return jsonify({'error': str(ve)}), 400

    except Exception as e:
        #print(f"Error: {e}")
        return jsonify({'error': str(e)}), 400

if __name__ == '__main__':
    app.run(debug=True)
```

CODE SNIPPETS

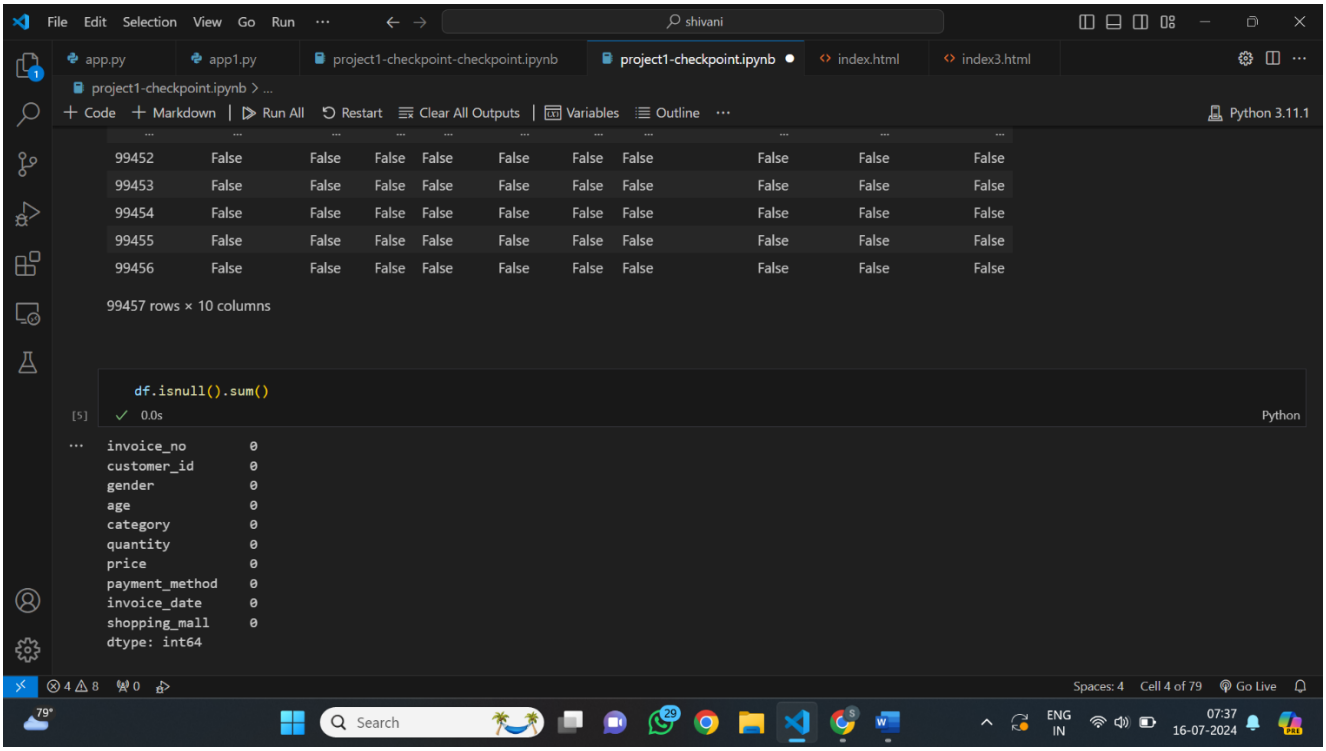
MODEL BUILDING

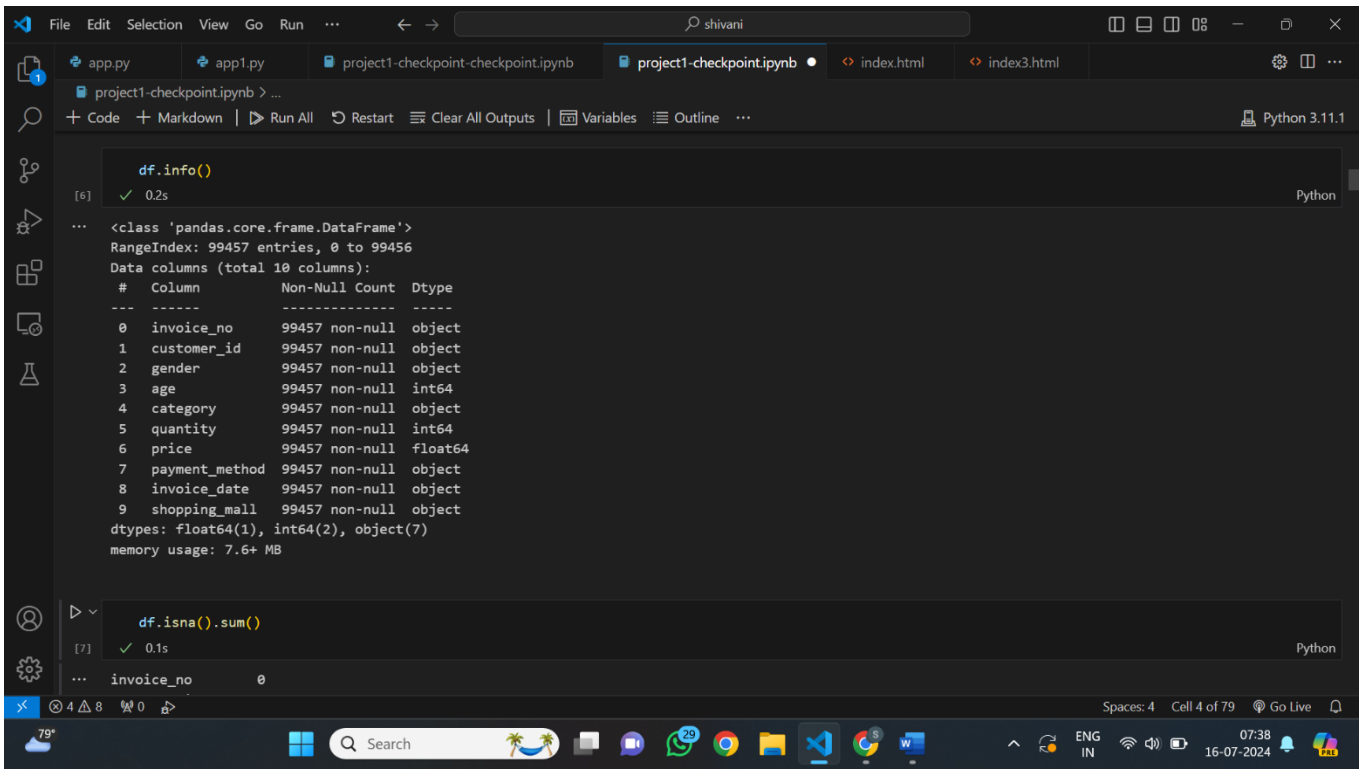


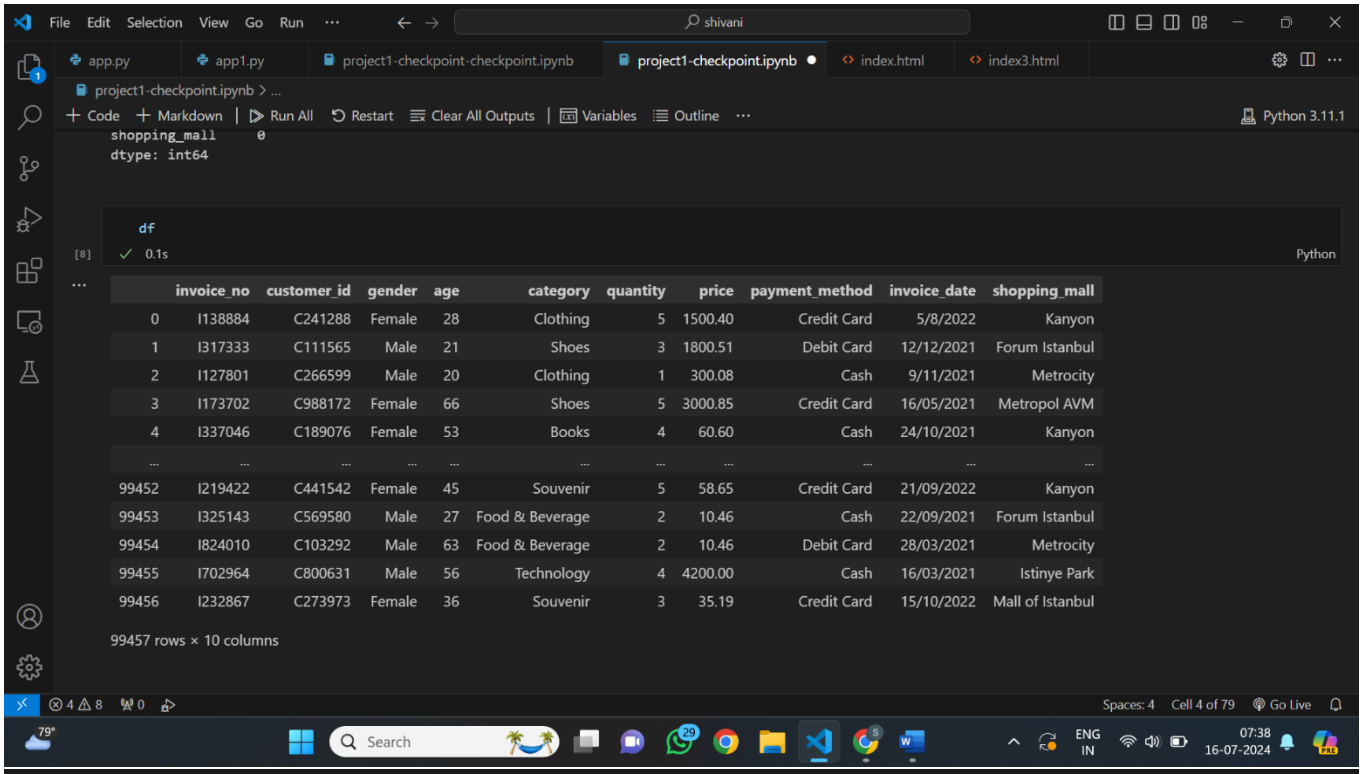


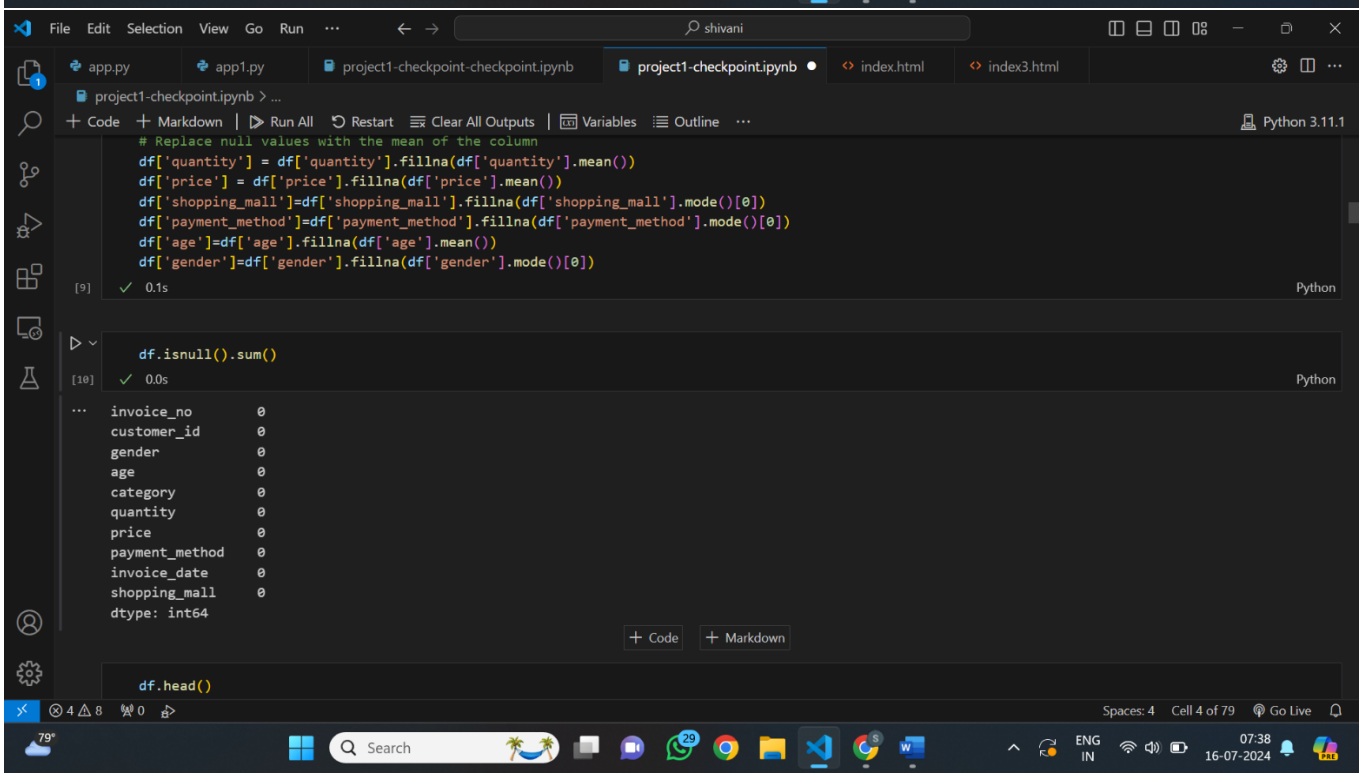
	invoice_no	customer_id	gender	age	category	quantity	price	payment_method	invoice_date	shopping_mall
0	I138884	C241288	Female	28	Clothing	5	1500.40	Credit Card	5/8/2022	Kanyon
1	I317333	C111565	Male	21	Shoes	3	1800.51	Debit Card	12/12/2021	Forum Istanbul
2	I127801	C266599	Male	20	Clothing	1	300.08	Cash	9/11/2021	Metrocity
3	I173702	C988172	Female	66	Shoes	5	3000.85	Credit Card	16/05/2021	Metropol AVM
4	I337046	C189076	Female	53	Books	4	60.60	Cash	24/10/2021	Kanyon

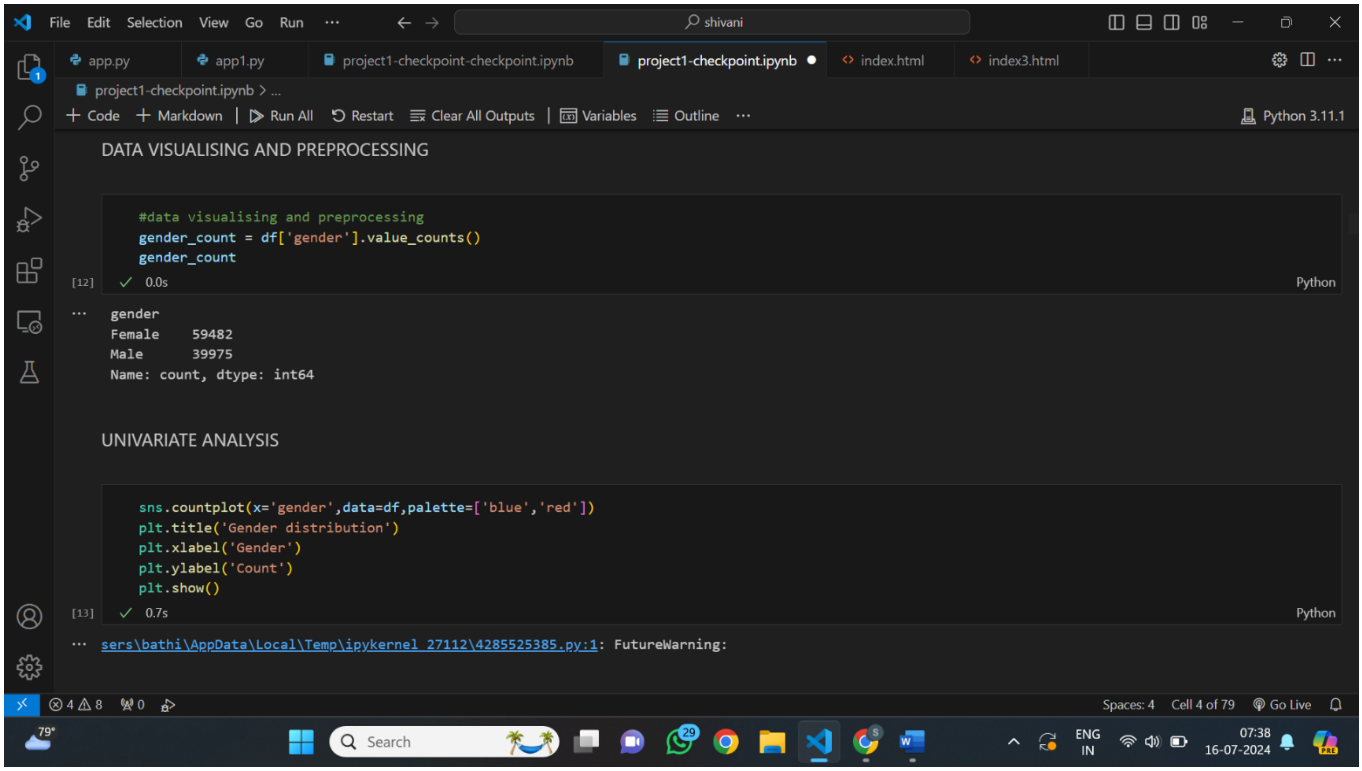
	invoice_no	customer_id	gender	age	category	quantity	price	payment_method	invoice_date	shopping_mall
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False

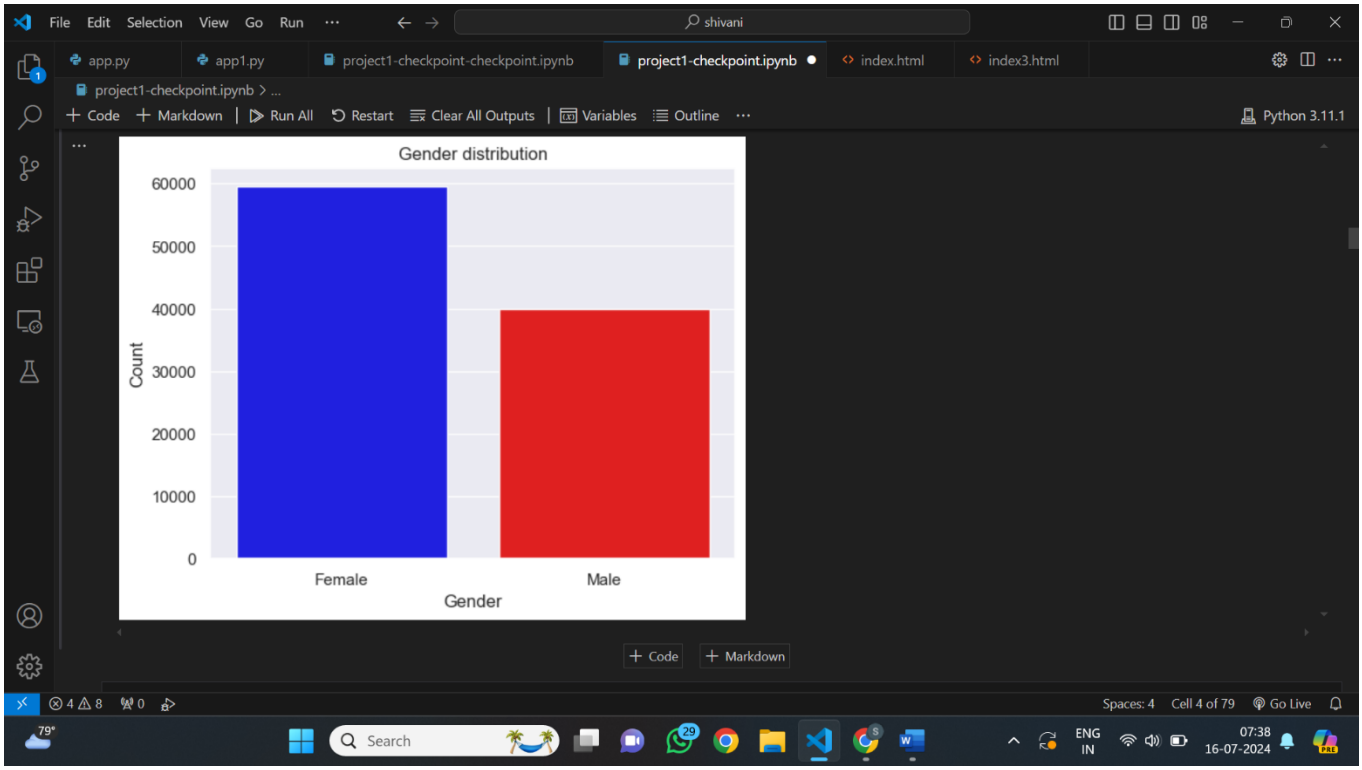

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
99452 False False False False False False False False False False
99453 False False False False False False False False False False
99454 False False False False False False False False False False
99455 False False False False False False False False False False
99456 False False False False False False False False False False
99457 rows × 10 columns
df.isnull().sum()
[5] 0.0s
Python
invoice_no 0
customer_id 0
gender 0
age 0
category 0
quantity 0
price 0
payment_method 0
invoice_date 0
shopping_mall 0
dtype: int64
Spaces: 4 Cell 4 of 79 Go Live
Search
ENG IN
07:37
16-07-2024

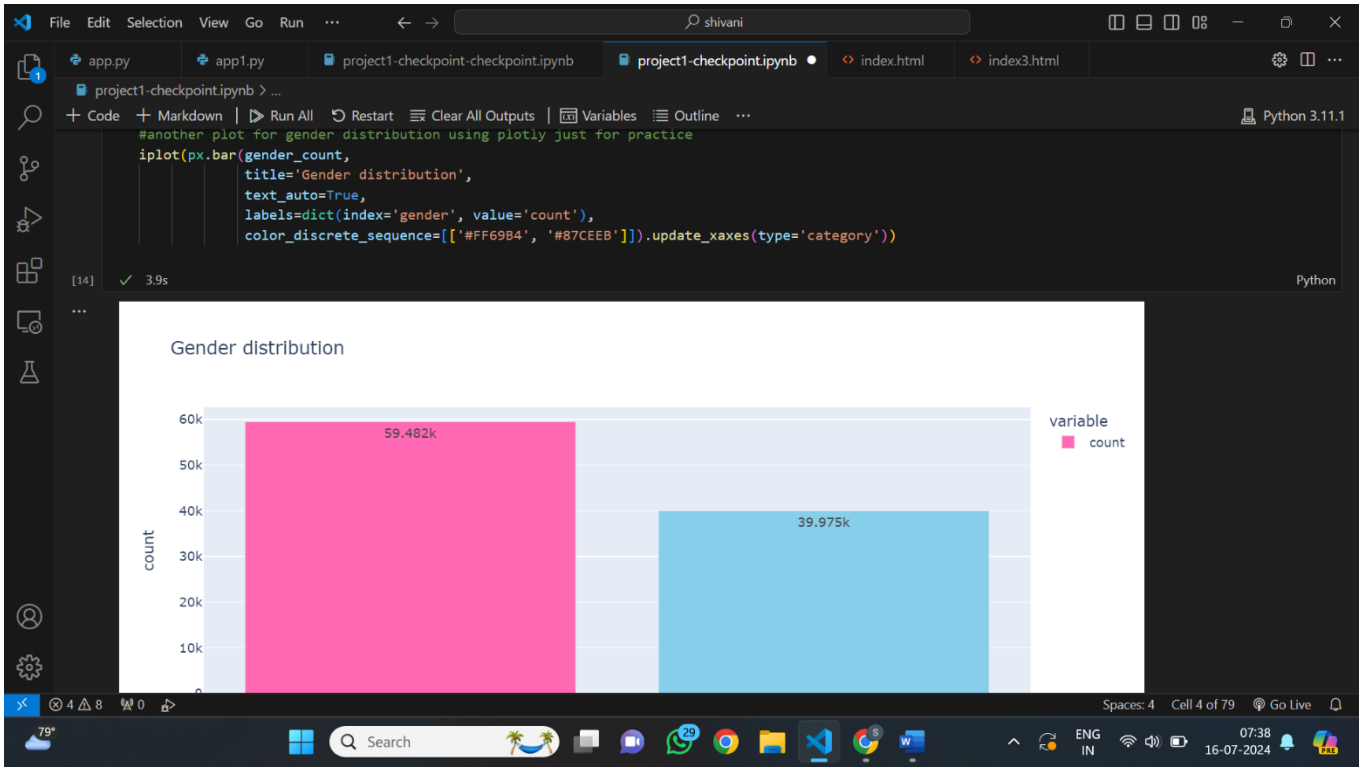

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
df.info()
[6] 0.2s
Python
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99457 entries, 0 to 99456
Data columns (total 10 columns):
Column Non-Null Count Dtype
--- ------ -------------- -----
0 invoice_no 99457 non-null object
1 customer_id 99457 non-null object
2 gender 99457 non-null object
3 age 99457 non-null int64
4 category 99457 non-null object
5 quantity 99457 non-null int64
6 price 99457 non-null float64
7 payment_method 99457 non-null object
8 invoice_date 99457 non-null object
9 shopping_mall 99457 non-null object
dtypes: float64(1), int64(2), object(7)
memory usage: 7.6+ MB
df.isna().sum()
[7] 0.1s
Python
invoice_no 0
Spaces: 4 Cell 4 of 79 Go Live
Search
ENG IN
07:38
16-07-2024

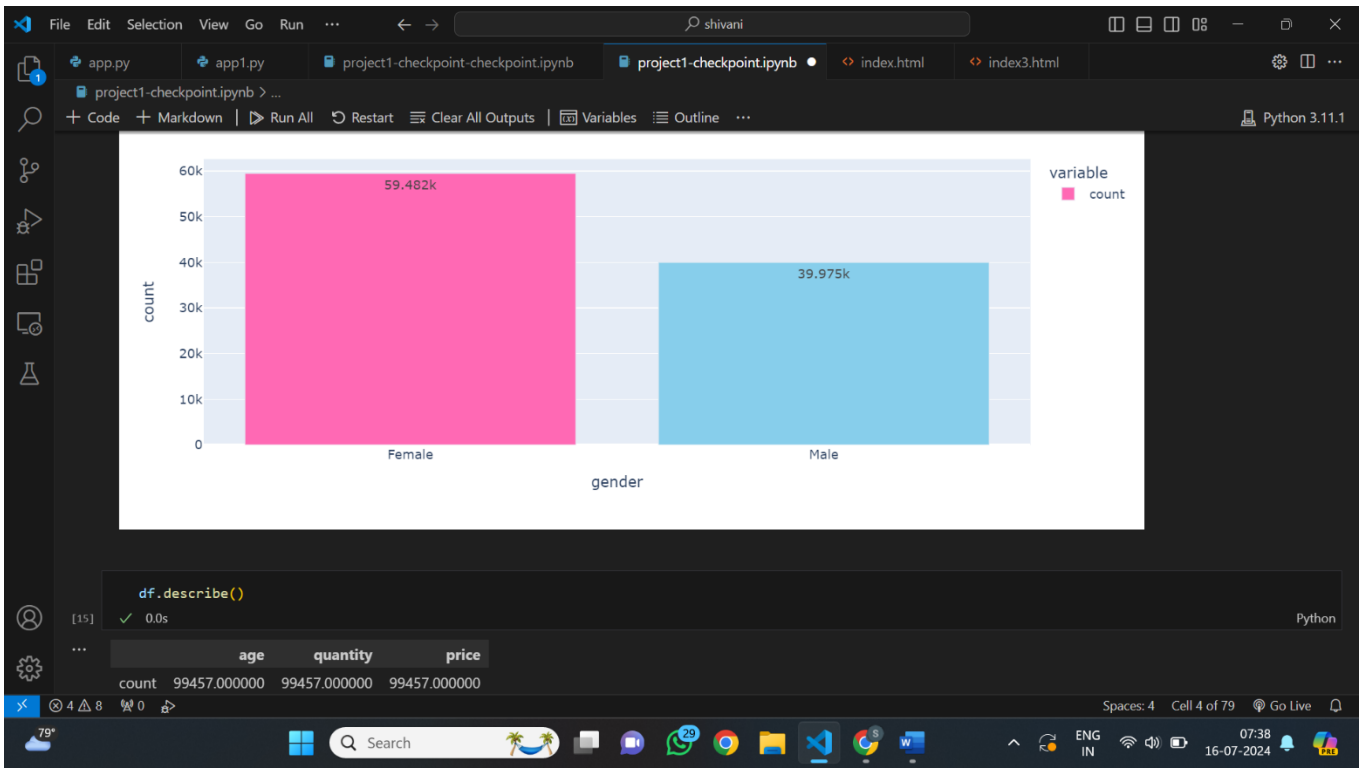

shopping_mall 0
dtype: int64
df
[8] 0.1s
Python
invoice_no customer_id gender age category quantity price payment_method invoice_date shopping_mall
0 I138884 C241288 Female 28 Clothing 5 1500.40 Credit Card 5/8/2022 Kanyon
1 I317333 C111565 Male 21 Shoes 3 1800.51 Debit Card 12/12/2021 Forum Istanbul
2 I127801 C266599 Male 20 Clothing 1 300.08 Cash 9/11/2021 Metrocity
3 I173702 C988172 Female 66 Shoes 5 3000.85 Credit Card 16/05/2021 Metropol AVM
4 I337046 C189076 Female 53 Books 4 60.60 Cash 24/10/2021 Kanyon
99452 I219422 C441542 Female 45 Souvenir 5 58.65 Credit Card 21/09/2022 Kanyon
99453 I325143 C569580 Male 27 Food & Beverage 2 10.46 Cash 22/09/2021 Forum Istanbul
99454 I824010 C103292 Male 63 Food & Beverage 2 10.46 Debit Card 28/03/2021 Metrocity
99455 I702964 C800631 Male 56 Technology 4 4200.00 Cash 16/03/2021 Istinye Park
99456 I232867 C273973 Female 36 Souvenir 3 35.19 Credit Card 15/10/2022 Mall of Istanbul
99457 rows × 10 columns

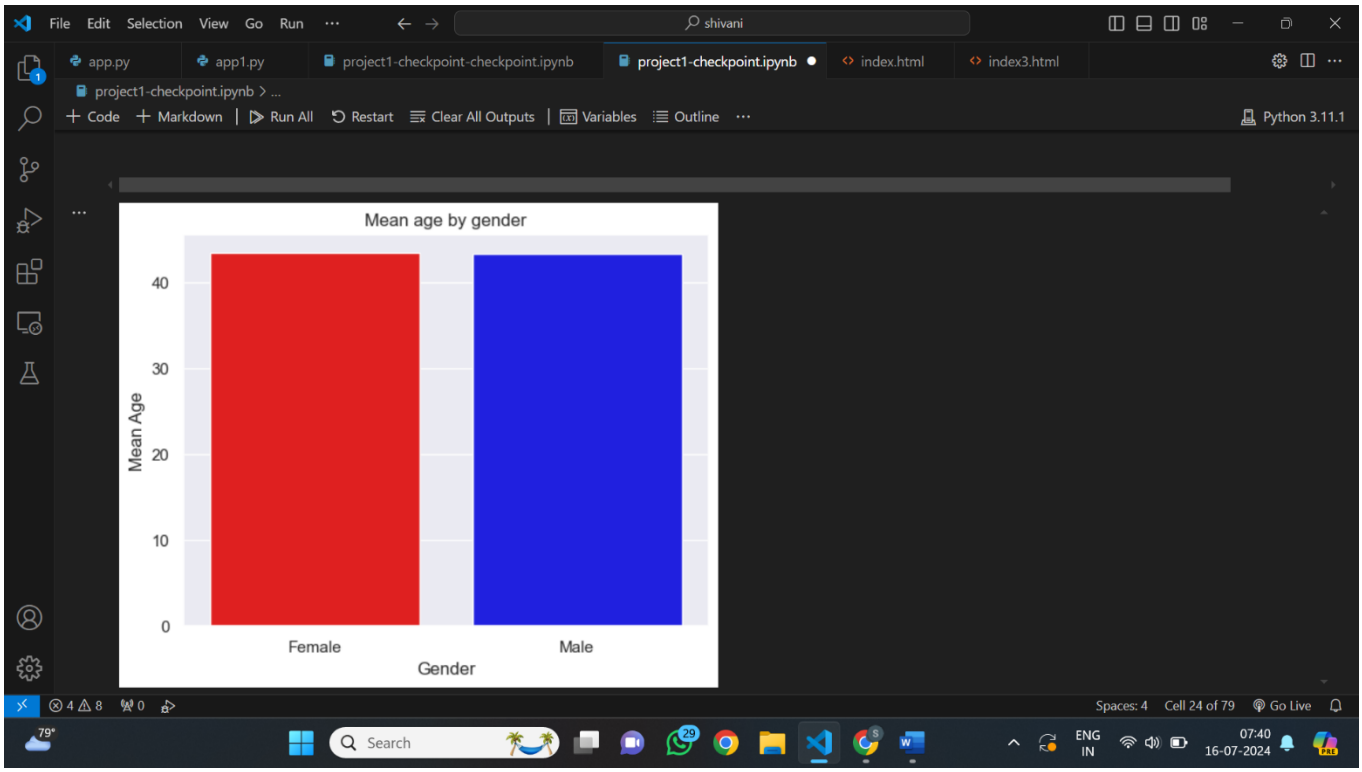

Replace null values with the mean of the column
df['quantity'] = df['quantity'].fillna(df['quantity'].mean())
df['price'] = df['price'].fillna(df['price'].mean())
df['shopping_mall']=df['shopping_mall'].fillna(df['shopping_mall'].mode()[0])
df['payment_method']=df['payment_method'].fillna(df['payment_method'].mode()[0])
df['age']=df['age'].fillna(df['age'].mean())
df['gender']=df['gender'].fillna(df['gender'].mode()[0])
[9] 0.1s
Python
df.isnull().sum()
[10] 0.0s
Python
invoice_no 0
customer_id 0
gender 0
age 0
category 0
quantity 0
price 0
payment_method 0
invoice_date 0
shopping_mall 0
dtype: int64
+ Code + Markdown
df.head()

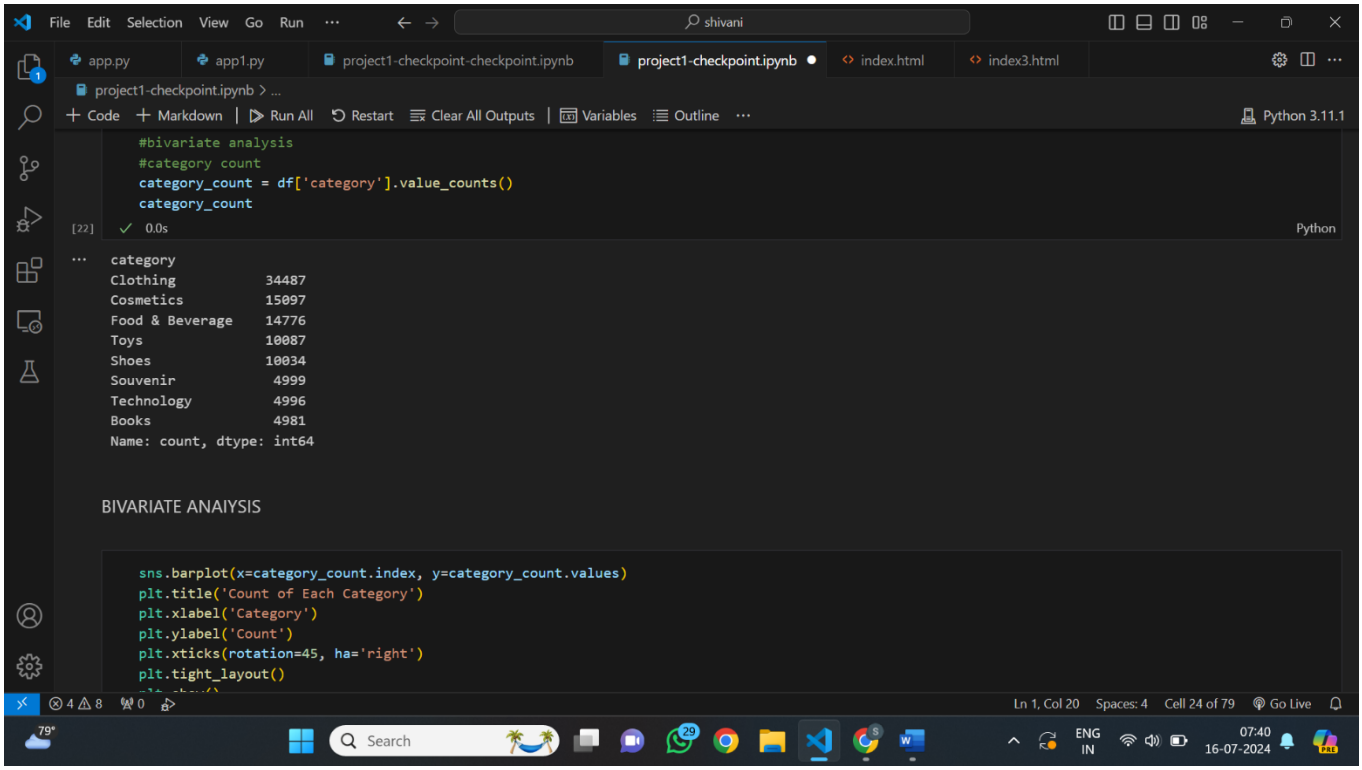

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
DATA VISUALISING AND PREPROCESSING
#data visualising and preprocessing
gender_count = df['gender'].value_counts()
gender_count
[12] 0.0s
Python
gender
Female 59482
Male 39975
Name: count, dtype: int64
UNIVARIATE ANALYSIS
sns.countplot(x='gender',data=df,palette=['blue','red'])
plt.title('Gender distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
[13] 0.7s
Python
sers\bathi\AppData\Local\Temp\ipykernel_27112\4285525385.py:1: FutureWarning:
Spaces: 4 Cell 4 of 79 Go Live
Search
ENG IN
07:38 16-07-2024

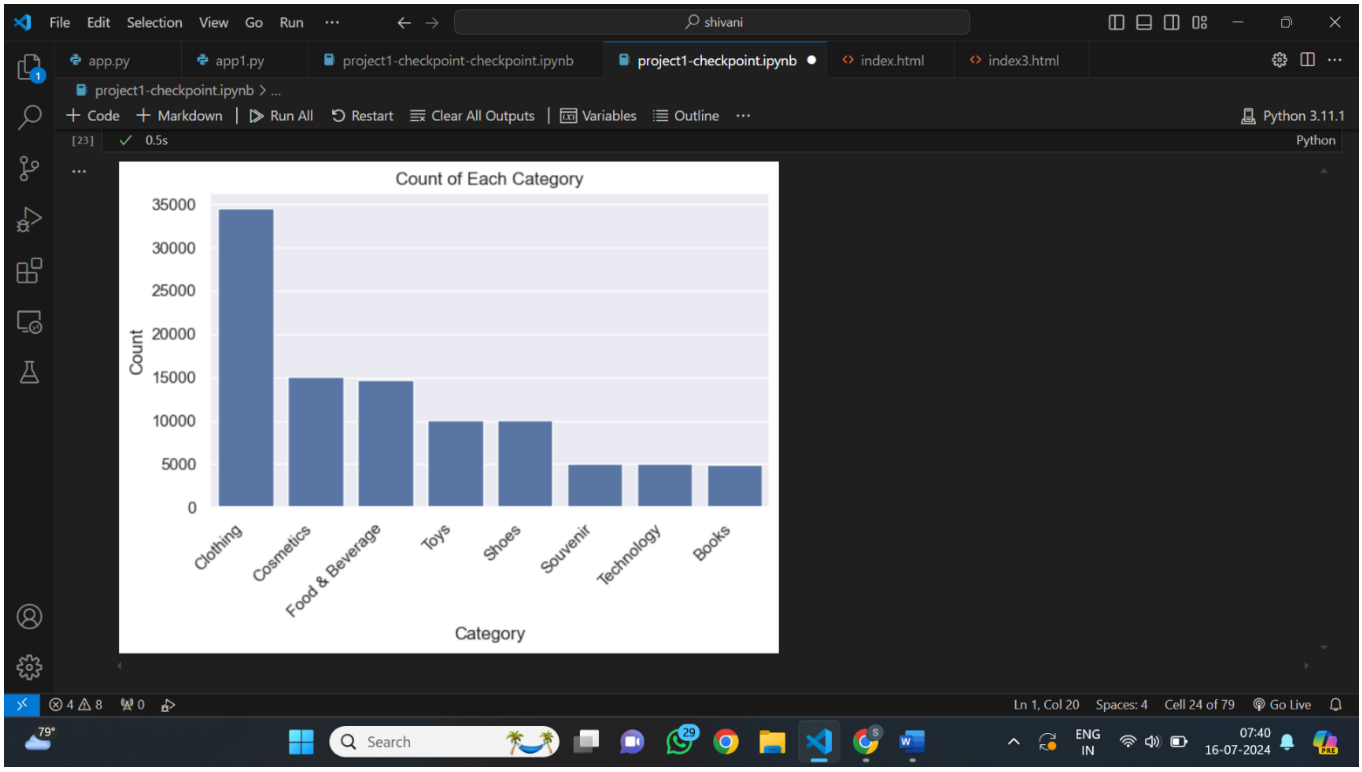

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
Gender distribution
60000
50000
40000
30000
20000
10000
0
Count
Female
Male
Gender
Code Markdown
Spaces: 4 Cell 4 of 79 Go Live
Search
ENG IN
07:38 16-07-2024

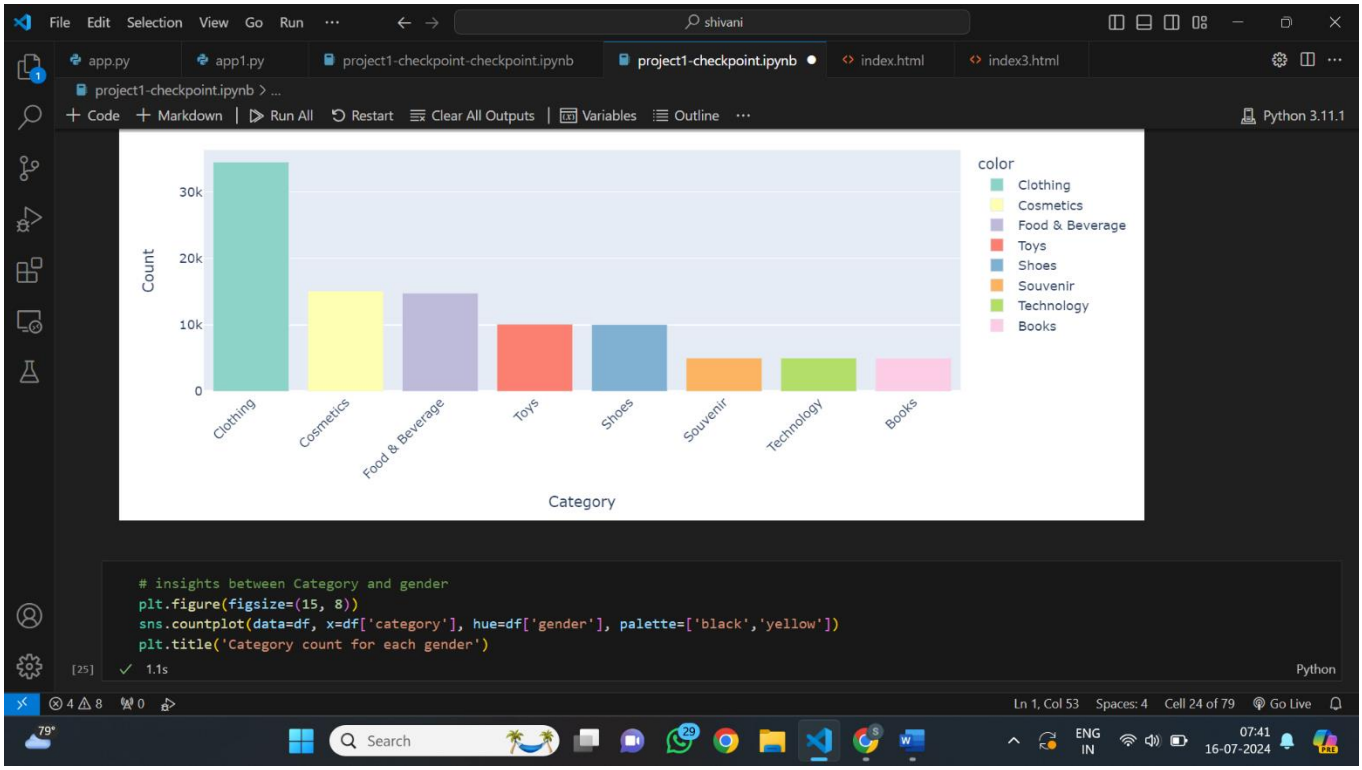

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
#another plot for gender distribution using plotly just for practice
iplot(px.bar(gender_count,
title='Gender distribution',
text_auto=True,
labels=dict(index='gender', value='count'),
color_discrete_sequence=[['#FF69B4', '#87CEEB']]).update_xaxes(type='category'))
[14] 3.9s
Python
Gender distribution
60k
50k
40k
30k
20k
10k
count
59.482k
39.975k
variable
count
Spaces: 4 Cell 4 of 79 Go Live
Search
ENG
IN
07:38
16-07-2024

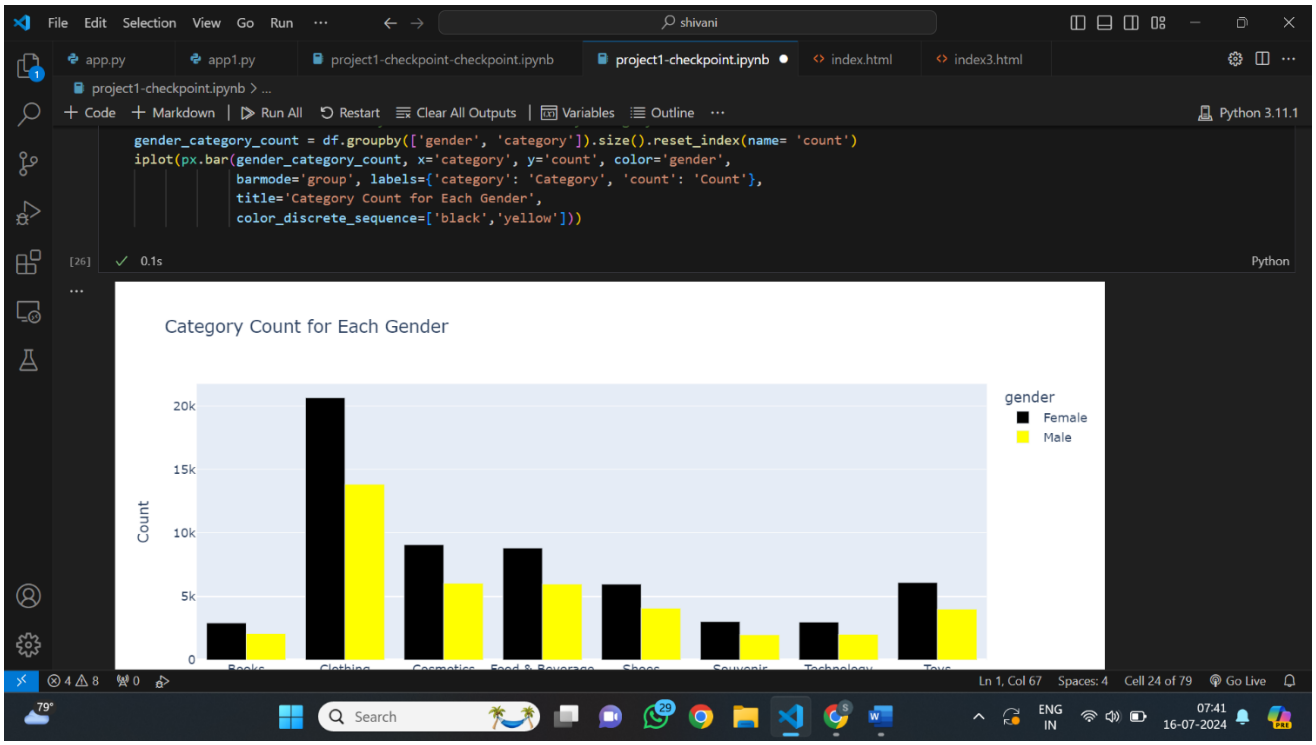

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
60k
50k
40k
30k
20k
10k
0
count
59.482k
39.975k
Female
Male
gender
variable
count
df.describe()
[15] 0.0s
Python
age quantity price
count 99457.000000 99457.000000 99457.000000
Spaces: 4 Cell 4 of 79 Go Live
Search
ENG
IN
07:38
16-07-2024

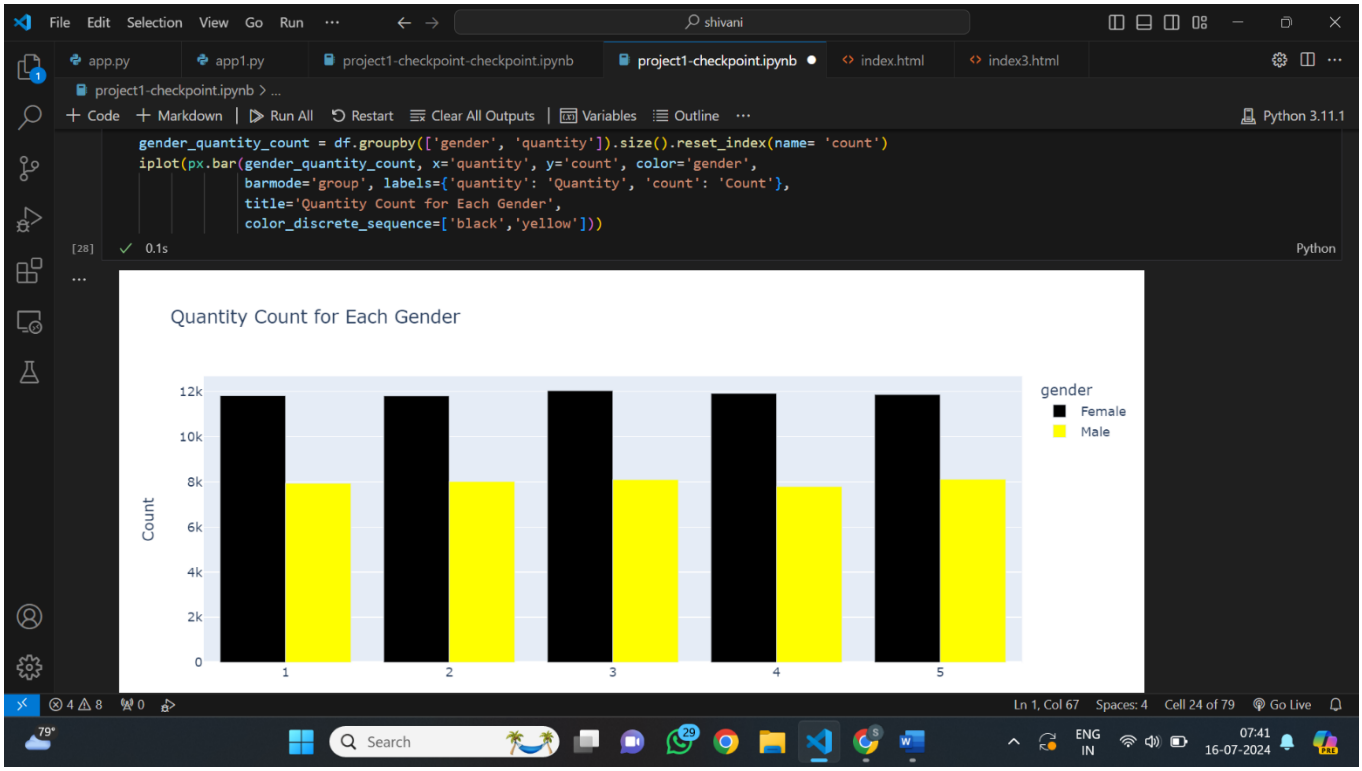

Mean age by gender
Mean Age
40
30
20
10
0
Female
Male
Gender

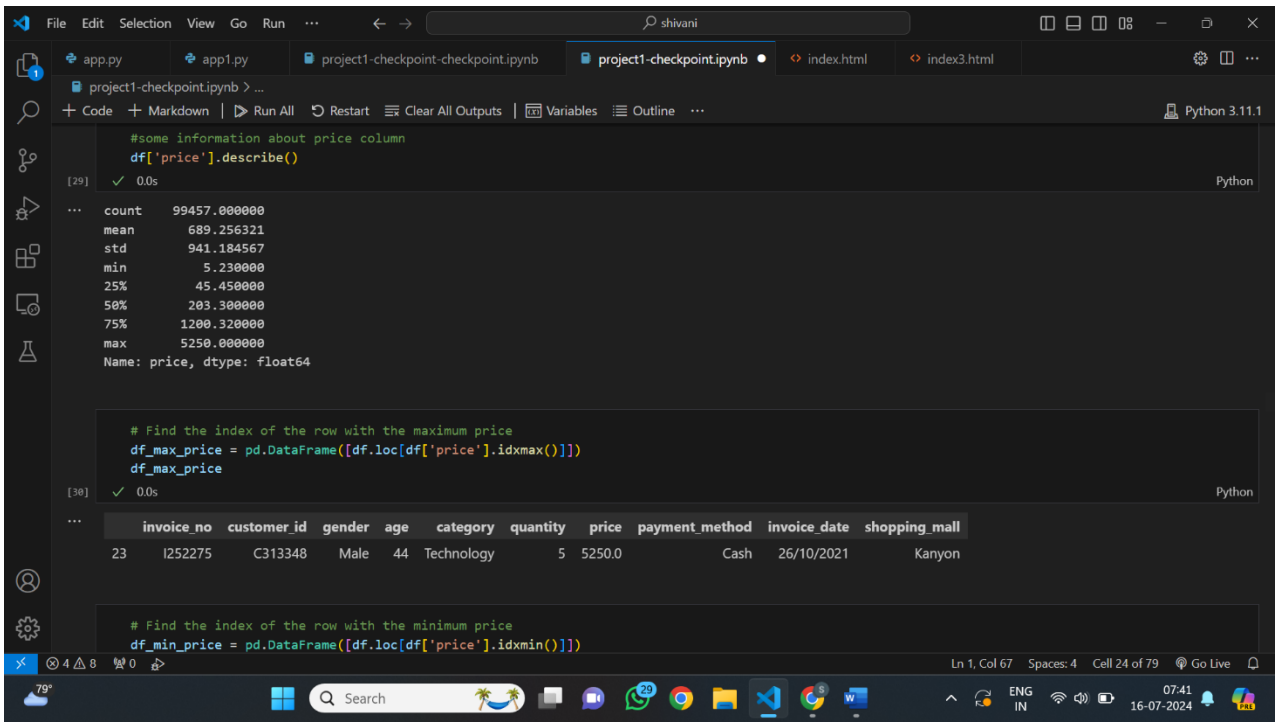

#bivariate analysis
#category count
category_count = df['category'].value_counts()
category_count
[22] 0.0s
Python
category
Clothing 34487
Cosmetics 15097
Food & Beverage 14776
Toys 10087
Shoes 10034
Souvenir 4999
Technology 4996
Books 4981
Name: count, dtype: int64
BIVARIATE ANAIYSIS
sns.barplot(x=category_count.index, y=category_count.values)
plt.title('Count of Each Category')
plt.xlabel('Category')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()

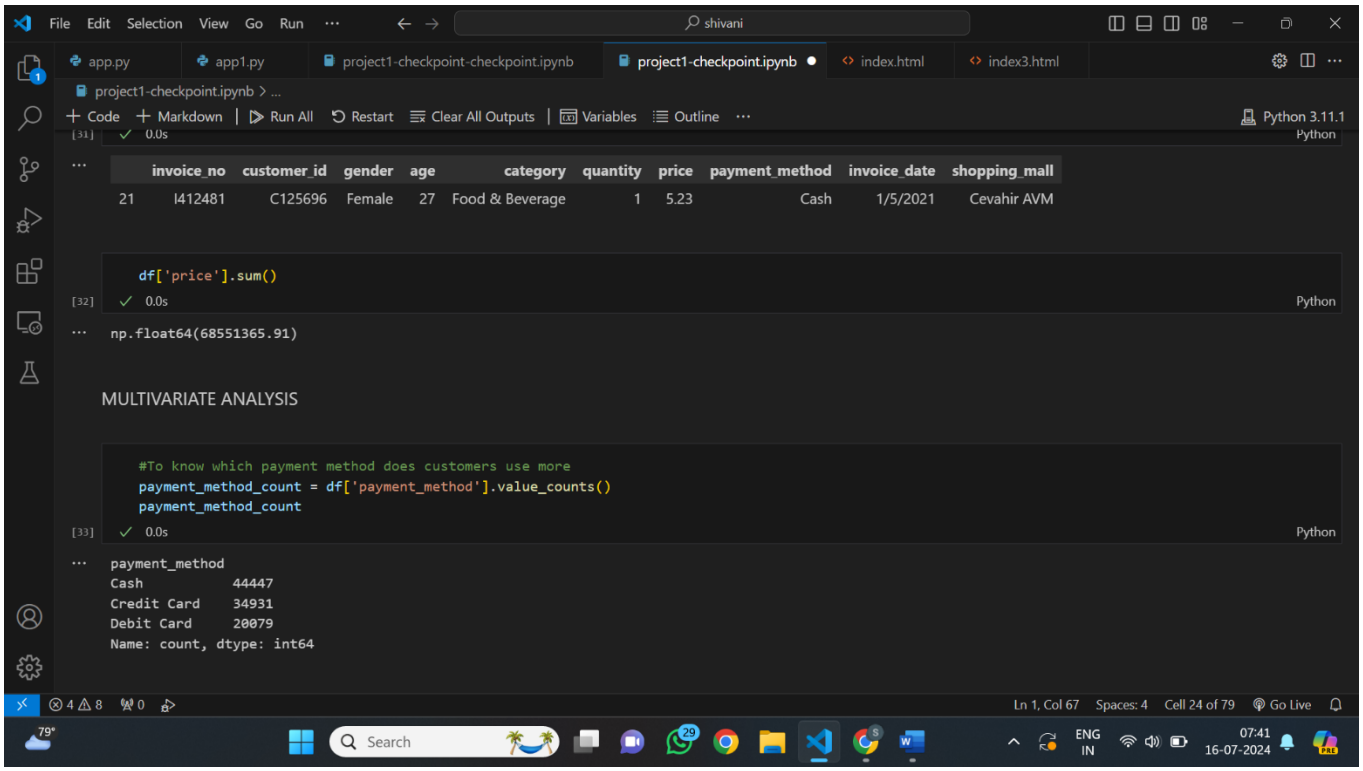

Count of Each Category
Count
Category
Clothing
Cosmetics
Food & Beverage
Toys
Shoes
Souvenir
Technology
Books

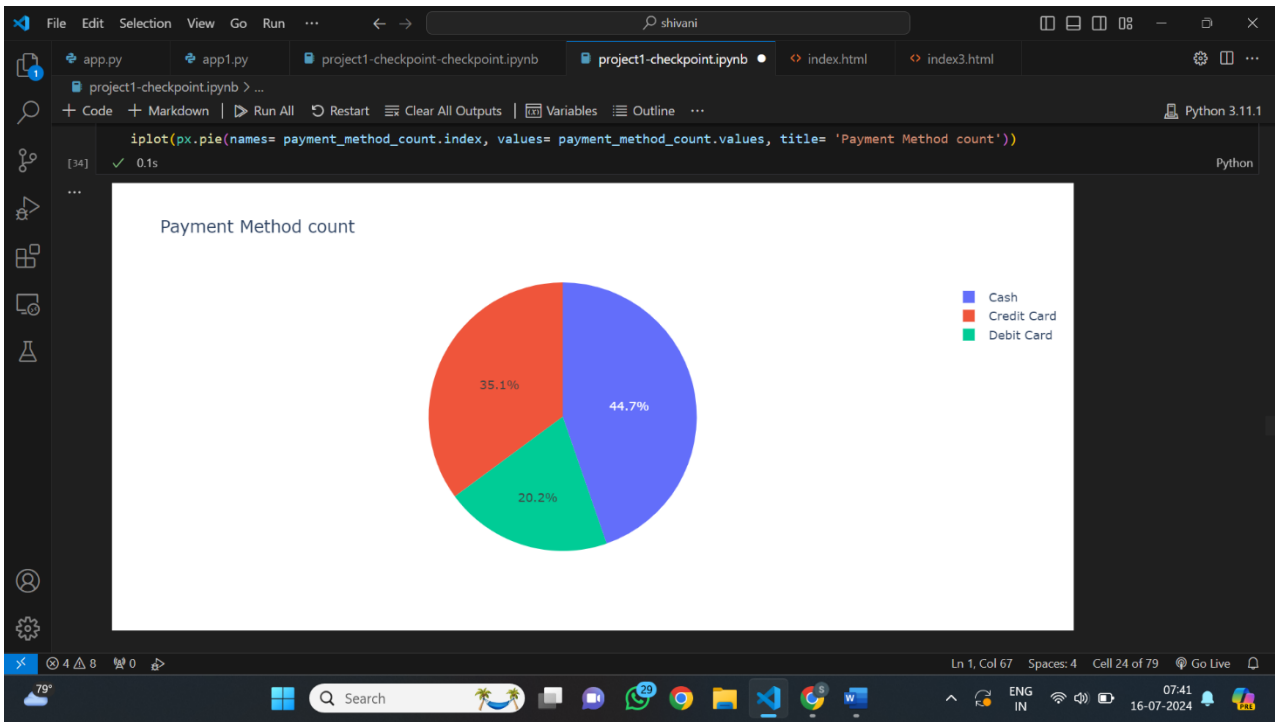

color
Clothing
Cosmetics
Food & Beverage
Toys
Shoes
Souvenir
Technology
Books
Count
Category
# insights between Category and gender
plt.figure(figsize=(15, 8))
sns.countplot(data=df, x=df['category'], hue=df['gender'], palette=['black','yellow'])
plt.title('Category count for each gender')

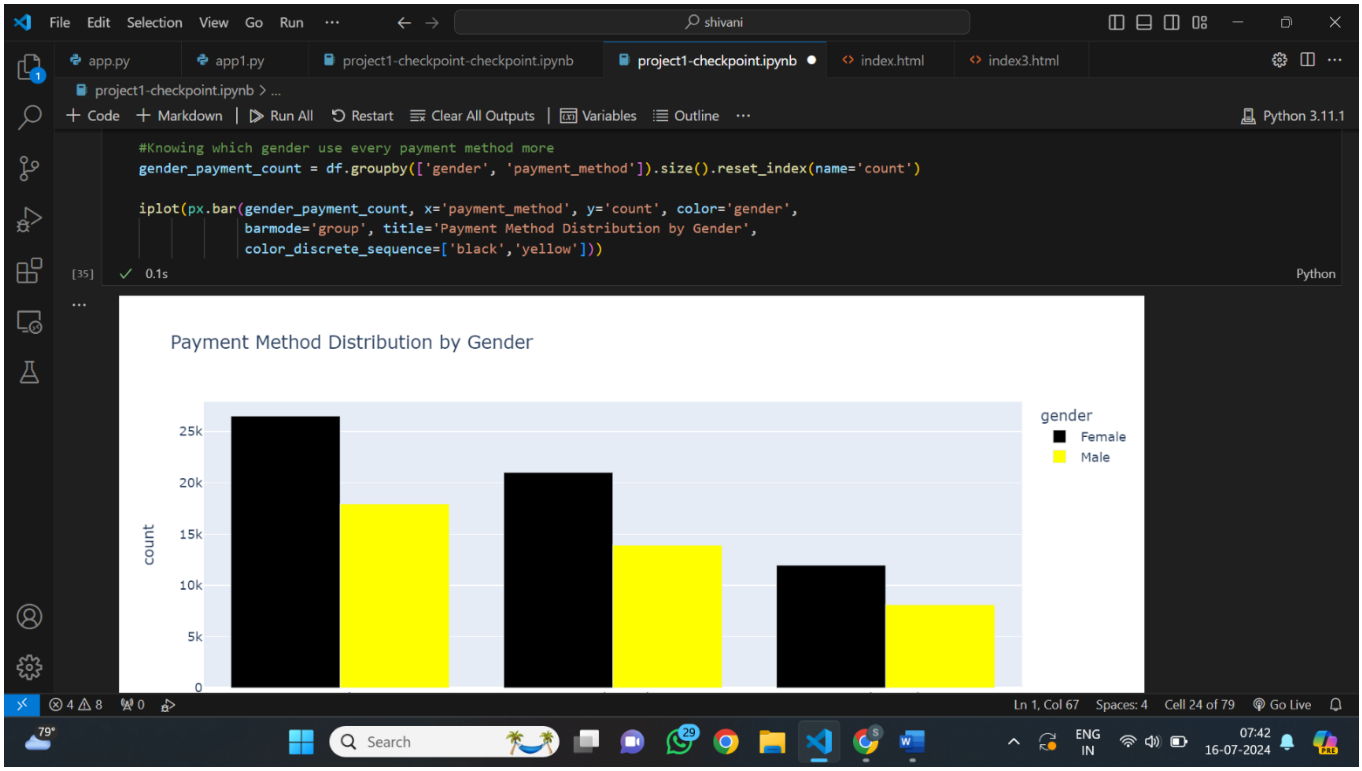

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
gender_category_count = df.groupby(['gender', 'category']).size().reset_index(name= 'count')
iplot(px.bar(gender_category_count, x='category', y='count', color='gender',
barmode='group', labels={'category': 'Category', 'count': 'Count'},
title='Category Count for Each Gender',
color_discrete_sequence=['black','yellow']))
[26] 0.1s
Python
Category Count for Each Gender
Count
20k
15k
10k
5k
0
gender
Female
Male
Books
Clothing
Cosmetics
Food & Beverage
Shoes
Souvenirs
Technology
Toys
Ln 1, Col 67 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG
IN
07:41
16-07-2024

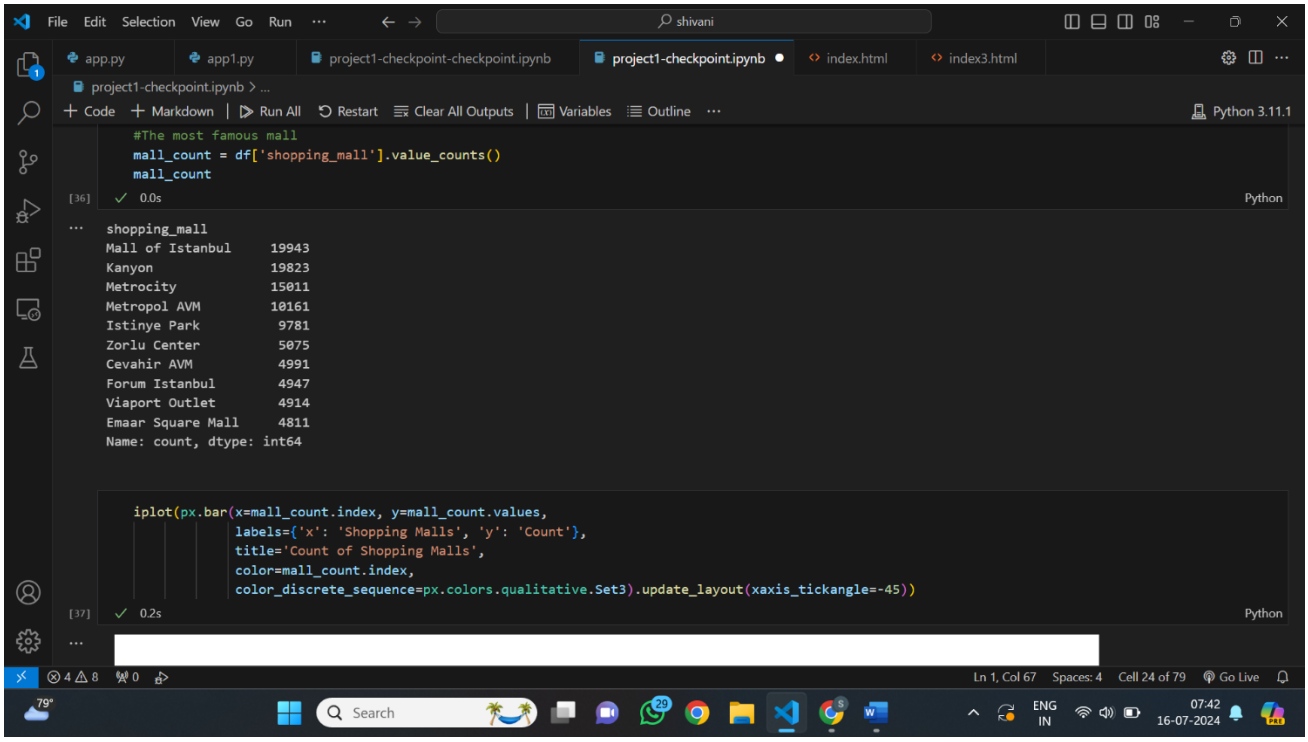

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
gender_quantity_count = df.groupby(['gender', 'quantity']).size().reset_index(name= 'count')
iplot(px.bar(gender_quantity_count, x='quantity', y='count', color='gender',
barmode='group', labels={'quantity': 'Quantity', 'count': 'Count'},
title='Quantity Count for Each Gender',
color_discrete_sequence=['black','yellow']))
[28] 0.1s
Python
Quantity Count for Each Gender
Count
12k
10k
8k
6k
4k
2k
0
1
2
3
4
5
gender
Female
Male
Ln 1, Col 67 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG
IN
07:41
16-07-2024

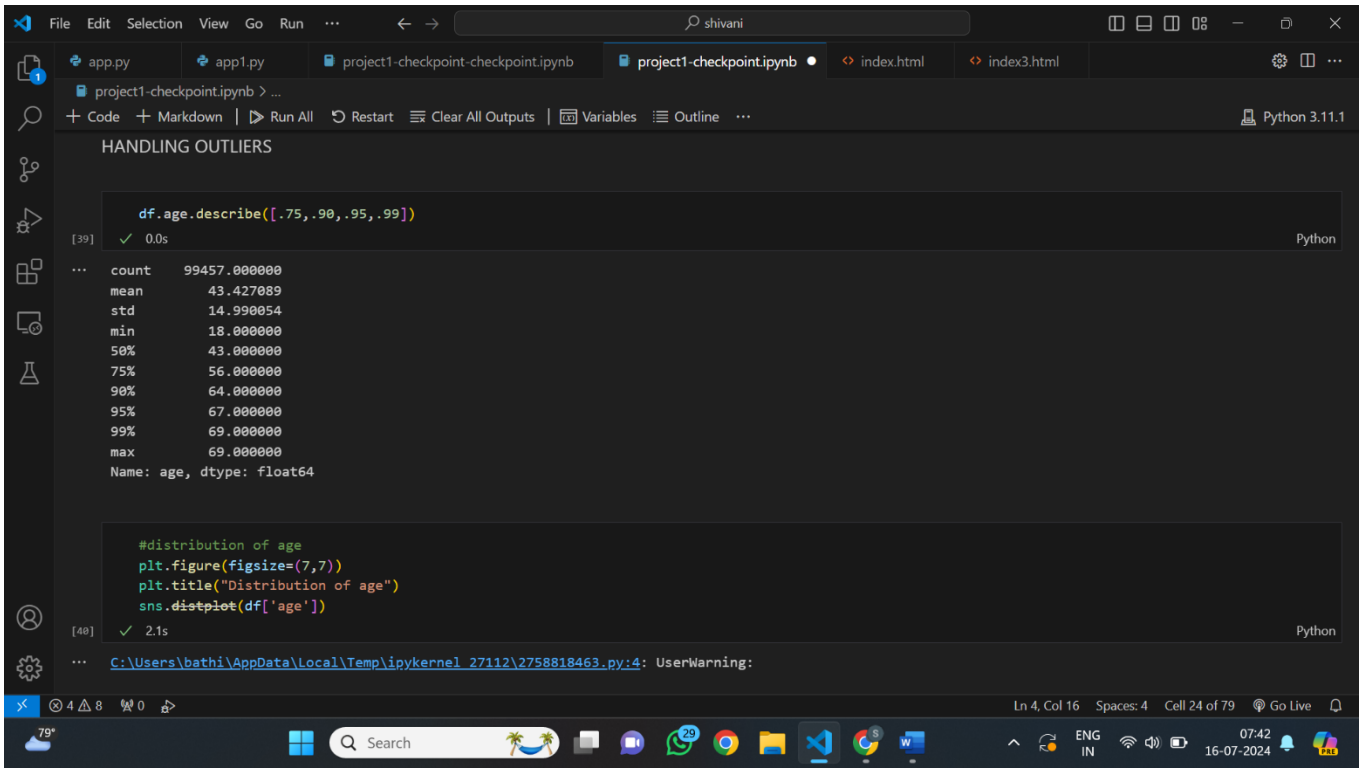

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
#some information about price column
df['price'].describe()
[29] 0.0s
Python
count 99457.000000
mean 689.256321
std 941.184567
min 5.230000
25% 45.450000
50% 203.300000
75% 1200.320000
max 5250.000000
Name: price, dtype: float64
# Find the index of the row with the maximum price
df_max_price = pd.DataFrame([df.loc[df['price'].idxmax()]])
df_max_price
[30] 0.0s
Python
invoice_no customer_id gender age category quantity price payment_method invoice_date shopping_mall
23 I252275 C313348 Male 44 Technology 5 5250.0 Cash 26/10/2021 Kanyon
# Find the index of the row with the minimum price
df_min_price = pd.DataFrame([df.loc[df['price'].idxmin()]])
Ln 1, Col 67 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:41 16-07-2024

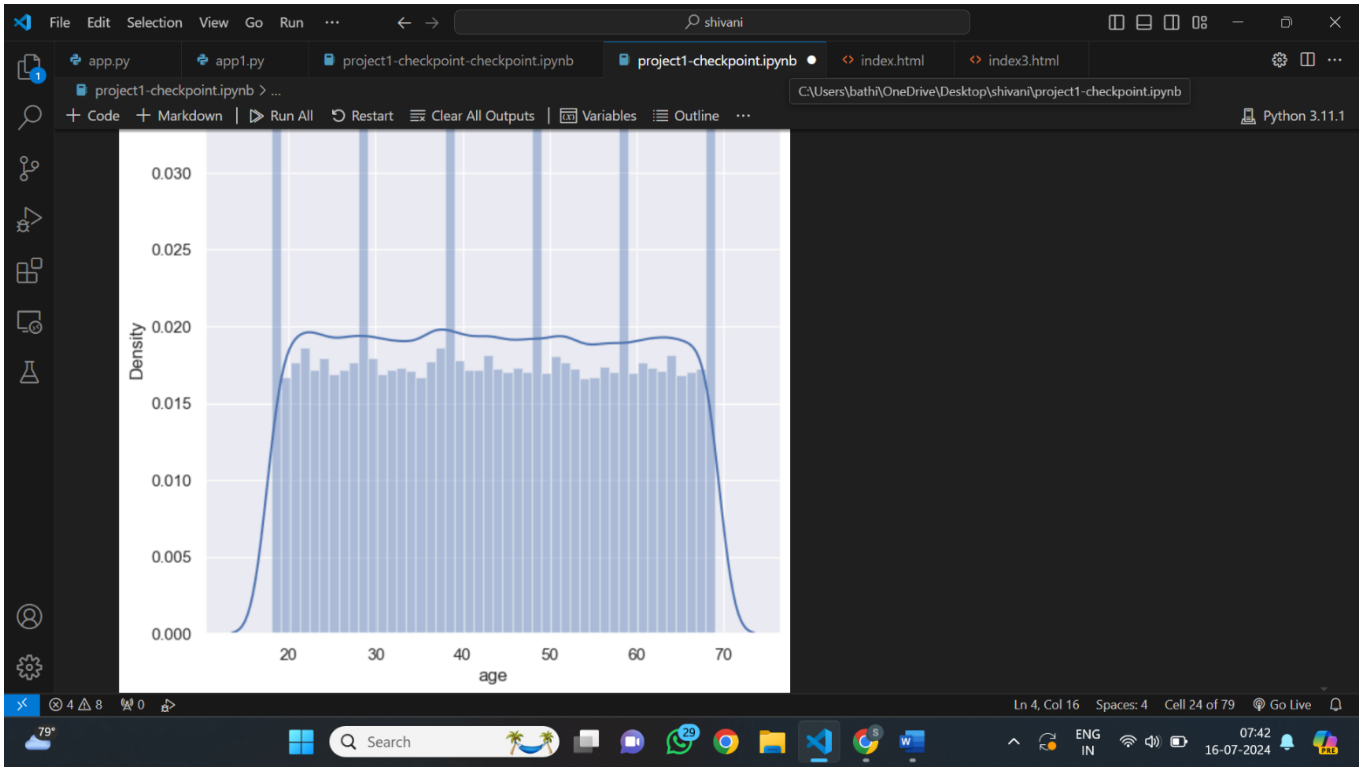

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
[31] 0.0s
Python
invoice_no customer_id gender age category quantity price payment_method invoice_date shopping_mall
21 I412481 C125696 Female 27 Food & Beverage 1 5.23 Cash 1/5/2021 Cevahir AVM
df['price'].sum()
[32] 0.0s
Python
np.float64(68551365.91)
MULTIVARIATE ANALYSIS
#To know which payment method does customers use more
payment_method_count = df['payment_method'].value_counts()
payment_method_count
[33] 0.0s
Python
payment_method
Cash 44447
Credit Card 34931
Debit Card 20079
Name: count, dtype: int64
Ln 1, Col 67 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:41 16-07-2024

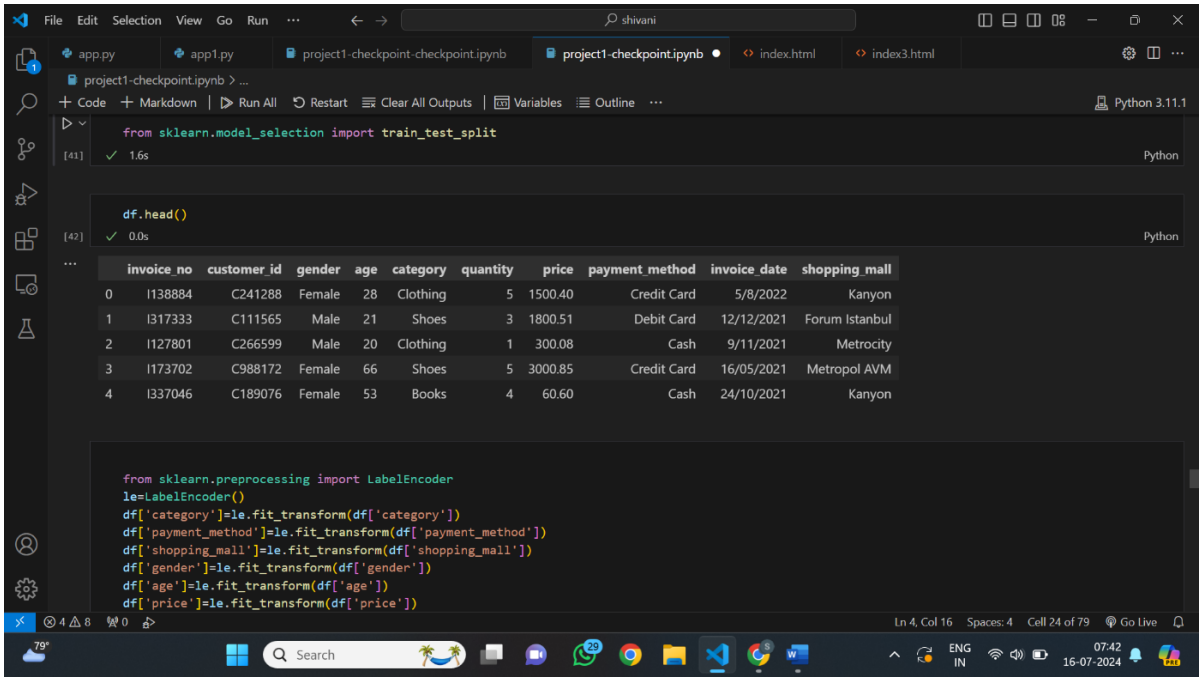

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
iplot(px.pie(names= payment_method_count.index, values= payment_method_count.values, title= 'Payment Method count'))
[34] 0.1s
Python
Payment Method count
Cash
Credit Card
Debit Card
35.1%
44.7%
20.2%
Ln 1, Col 67 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:41 16-07-2024

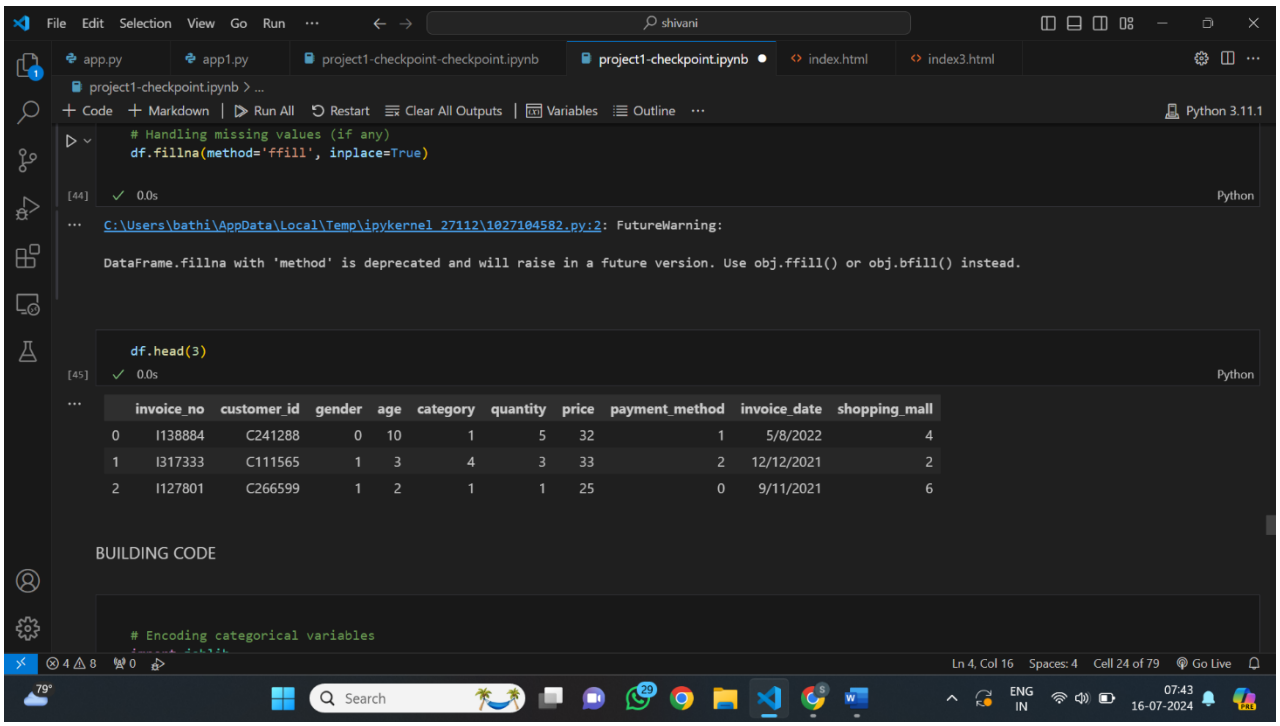

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
#Knowing which gender use every payment method more
gender_payment_count = df.groupby(['gender', 'payment_method']).size().reset_index(name='count')
iplot(px.bar(gender_payment_count, x='payment_method', y='count', color='gender',
barmode='group', title='Payment Method Distribution by Gender',
color_discrete_sequence=['black','yellow']))
[35] 0.1s
Python
Payment Method Distribution by Gender
gender
Female
Male
count
25k
20k
15k
10k
5k
0
Ln 1, Col 67 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:42 16-07-2024

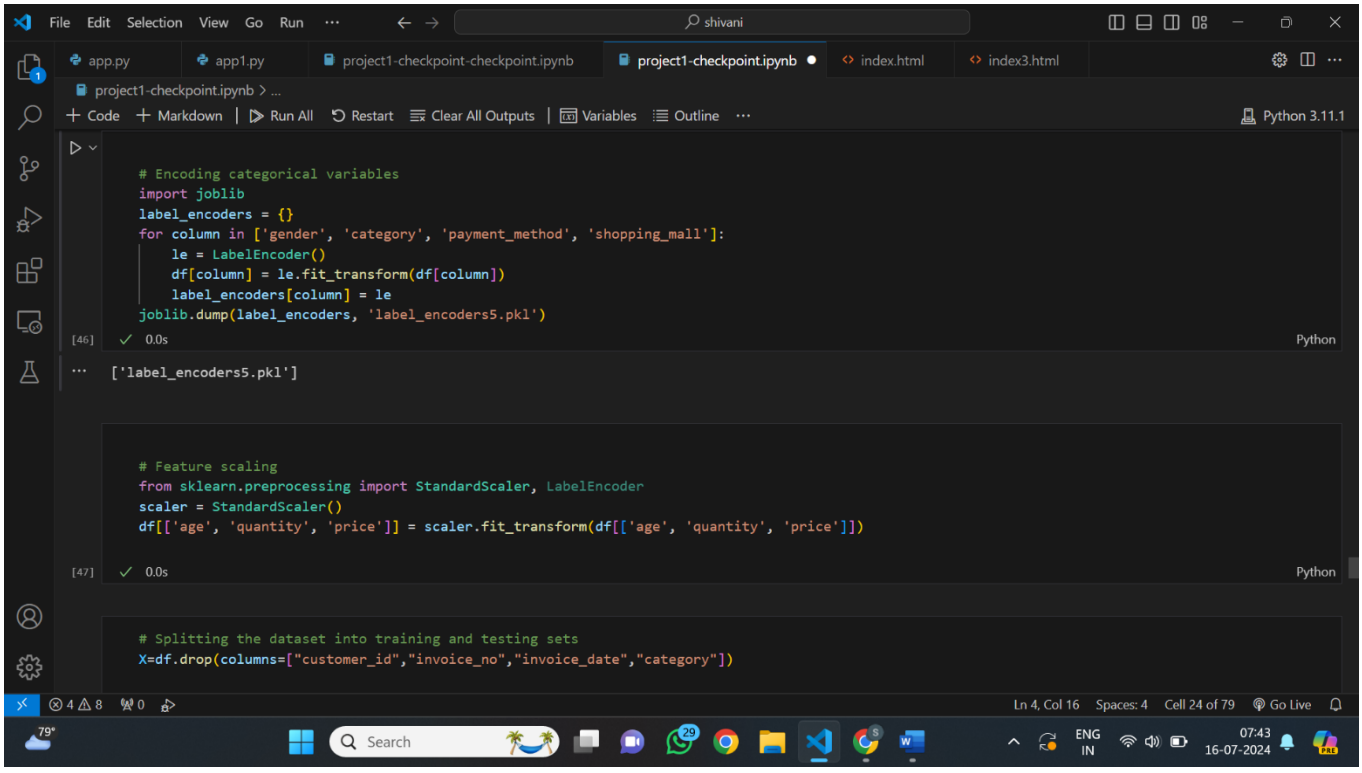

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
#The most famous mall
mall_count = df['shopping_mall'].value_counts()
mall_count
[36] 0.0s
Python
shopping_mall
Mall of Istanbul 19943
Kanyon 19823
Metrocity 15011
Metropol AVM 10161
Istinye Park 9781
Zorlu Center 5075
Cevahir AVM 4991
Forum Istanbul 4947
Viaport Outlet 4914
Emaar Square Mall 4811
Name: count, dtype: int64
iplot(px.bar(x=mall_count.index, y=mall_count.values,
labels={'x': 'Shopping Malls', 'y': 'Count'},
title='Count of Shopping Malls',
color=mall_count.index,
color_discrete_sequence=px.colors.qualitative.Set3).update_layout(xaxis_tickangle=-45))
[37] 0.2s
Python
Ln 1, Col 67 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:42 16-07-2024

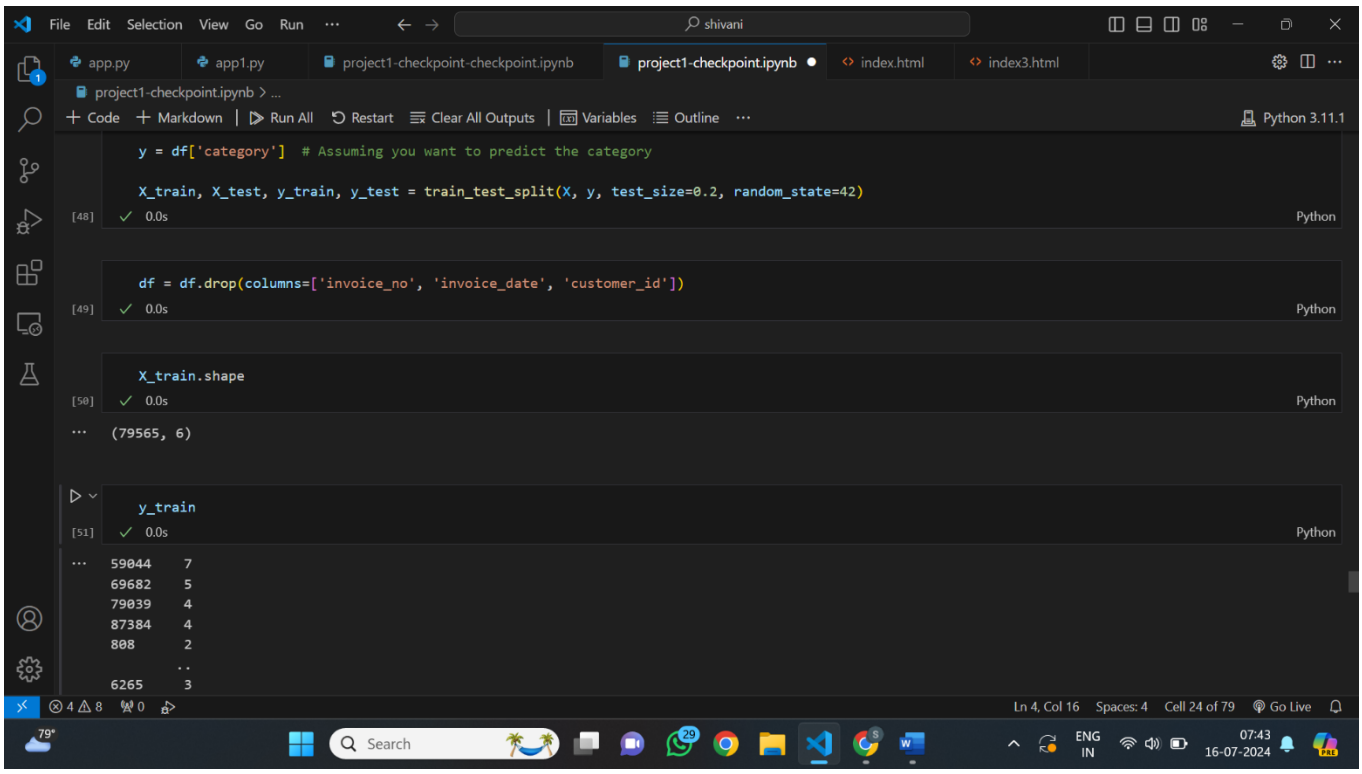

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
HANDLING OUTLIERS
df.age.describe([.75,.90,.95,.99])
[39] 0.0s
Python
count 99457.000000
mean 43.427089
std 14.990054
min 18.000000
50% 43.000000
75% 56.000000
90% 64.000000
95% 67.000000
99% 69.000000
max 69.000000
Name: age, dtype: float64
#distribution of age
plt.figure(figsize=(7,7))
plt.title("Distribution of age")
sns.distplot(df['age'])
[40] 2.1s
Python
C:\Users\bathi\AppData\Local\Temp\ipykernel_27112\2758818463.py:4: UserWarning:
Ln 4, Col 16 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:42 16-07-2024

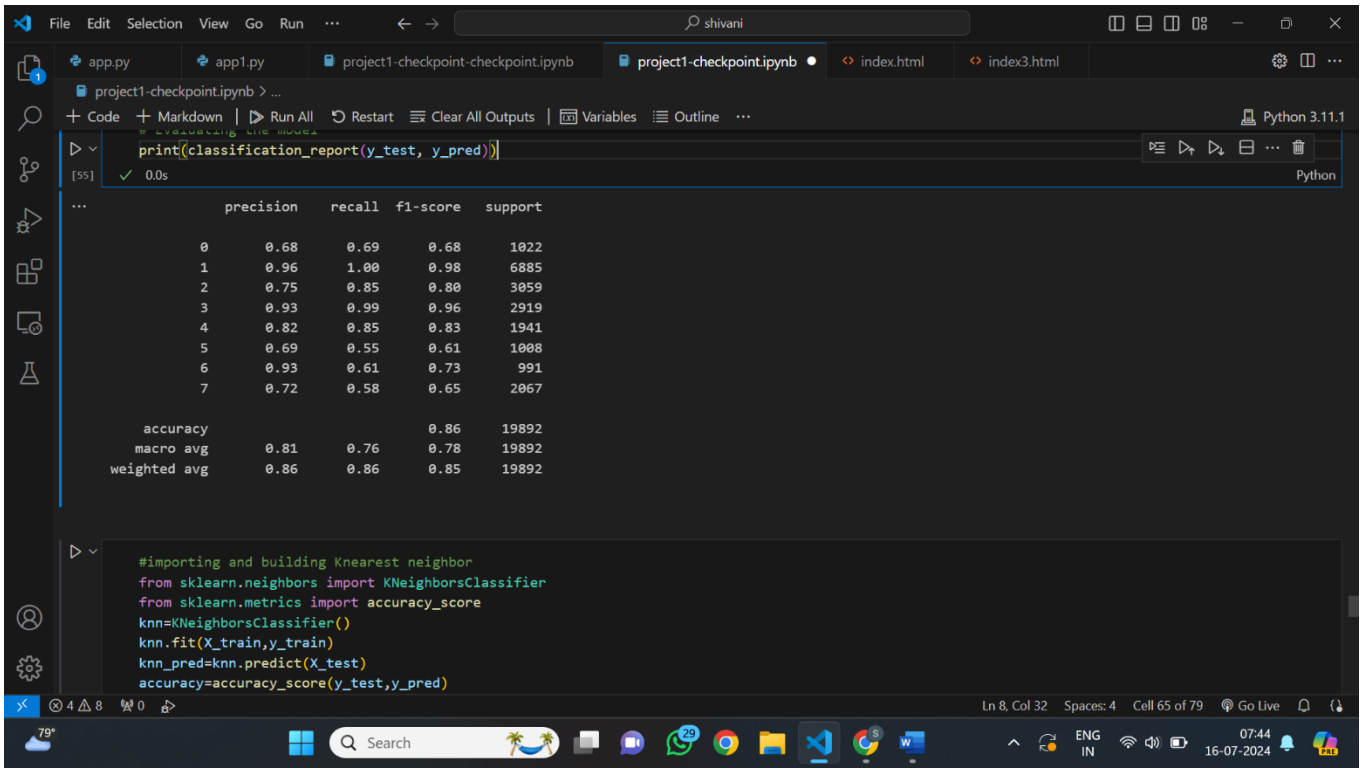

File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
C:\Users\bathi\OneDrive\Desktop\shivani\project1-checkpoint.ipynb
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
Density
age
0.030
0.025
0.020
0.015
0.010
0.005
0.000
20
30
40
50
60
70
Ln 4, Col 16 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:42 16-07-2024


File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
from sklearn.model_selection import train_test_split
[41] 1.6s
Python
df.head()
[42] 0.0s
Python
invoice_no customer_id gender age category quantity price payment_method invoice_date shopping_mall
0 I138884 C241288 Female 28 Clothing 5 1500.40 Credit Card 5/8/2022 Kanyon
1 I317333 C111565 Male 21 Shoes 3 1800.51 Debit Card 12/12/2021 Forum Istanbul
2 I127801 C266599 Male 20 Clothing 1 300.08 Cash 9/11/2021 Metrocity
3 I173702 C988172 Female 66 Shoes 5 3000.85 Credit Card 16/05/2021 Metropol AVM
4 I337046 C189076 Female 53 Books 4 60.60 Cash 24/10/2021 Kanyon
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['category']=le.fit_transform(df['category'])
df['payment_method']=le.fit_transform(df['payment_method'])
df['shopping_mall']=le.fit_transform(df['shopping_mall'])
df['gender']=le.fit_transform(df['gender'])
df['age']=le.fit_transform(df['age'])
df['price']=le.fit_transform(df['price'])
Ln 4, Col 16 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:42 16-07-2024


File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
# Handling missing values (if any)
df.fillna(method='ffill', inplace=True)
[44] 0.0s
Python
C:\Users\bathi\AppData\Local\Temp\ipykernel_27112\1027104582.py:2: FutureWarning:
DataFrame.fillna with 'method' is deprecated and will raise in a future version. Use obj.ffill() or obj.bfill() instead.
df.head(3)
[45] 0.0s
Python
invoice_no customer_id gender age category quantity price payment_method invoice_date shopping_mall
0 I138884 C241288 0 10 1 5 32 1 5/8/2022 4
1 I317333 C111565 1 3 4 3 33 2 12/12/2021 2
2 I127801 C266599 1 2 1 1 25 0 9/11/2021 6
BUILDING CODE
Encoding categorical variables
Ln 4, Col 16 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:43 16-07-2024


File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
# Encoding categorical variables
import joblib
label_encoders = {}
for column in ['gender', 'category', 'payment_method', 'shopping_mall']:
le = LabelEncoder()
df[column] = le.fit_transform(df[column])
label_encoders[column] = le
joblib.dump(label_encoders, 'label_encoders5.pkl')
[46] 0.0s
Python
['label_encoders5.pkl']
# Feature scaling
from sklearn.preprocessing import StandardScaler, LabelEncoder
scaler = StandardScaler()
df[['age', 'quantity', 'price']] = scaler.fit_transform(df[['age', 'quantity', 'price']])
[47] 0.0s
Python
# Splitting the dataset into training and testing sets
X=df.drop(columns=["customer_id","invoice_no","invoice_date","category"])
Ln 4, Col 16 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:43 16-07-2024


File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
y = df['category'] # Assuming you want to predict the category
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
[48] 0.0s Python
df = df.drop(columns=['invoice_no', 'invoice_date', 'customer_id'])
[49] 0.0s Python
X_train.shape
[50] 0.0s Python
(79565, 6)
y_train
[51] 0.0s Python
59044 7
69682 5
79039 4
87384 4
808 2
..
6265 3
Ln 4, Col 16 Spaces: 4 Cell 24 of 79 Go Live
Search
ENG IN
07:43 16-07-2024


File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
print(classification_report(y_test, y_pred))
[55] 0.0s Python
precision recall f1-score support
0 0.68 0.69 0.68 1022
1 0.96 1.00 0.98 6885
2 0.75 0.85 0.80 3059
3 0.93 0.99 0.96 2919
4 0.82 0.85 0.83 1941
5 0.69 0.55 0.61 1008
6 0.93 0.61 0.73 991
7 0.72 0.58 0.65 2067
accuracy 0.86 19892
macro avg 0.81 0.76 0.78 19892
weighted avg 0.86 0.86 0.85 19892
#importing and building Knearest neighbor
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
knn=KNeighborsClassifier()
knn.fit(X_train,y_train)
knn_pred=knn.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
Ln 8, Col 32 Spaces: 4 Cell 65 of 79 Go Live
Search
ENG IN
07:44 16-07-2024

```
0.8581339231852001
```

```
#importing and building KMeans
from sklearn.cluster import KMeans
km=KMeans()
km.fit(X_train,y_train)
km_pred=km.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
accuracy
```

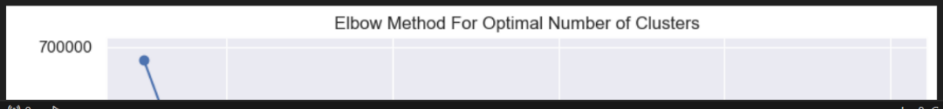
```
0.8581339231852001
```

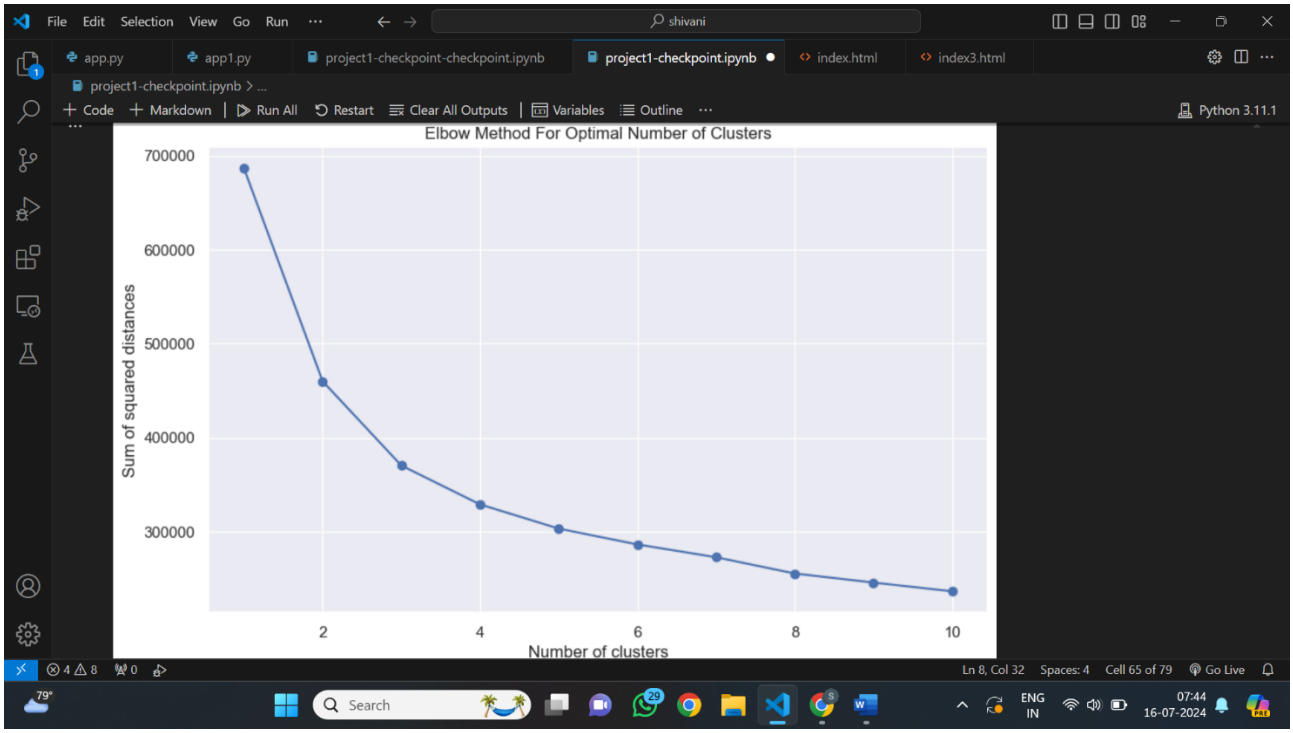
```
#import and building decision tree
from sklearn.tree import DecisionTreeClassifier
dt=DecisionTreeClassifier()
dt.fit(X_train,y_train)
y_pred=dt.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
accuracy
```

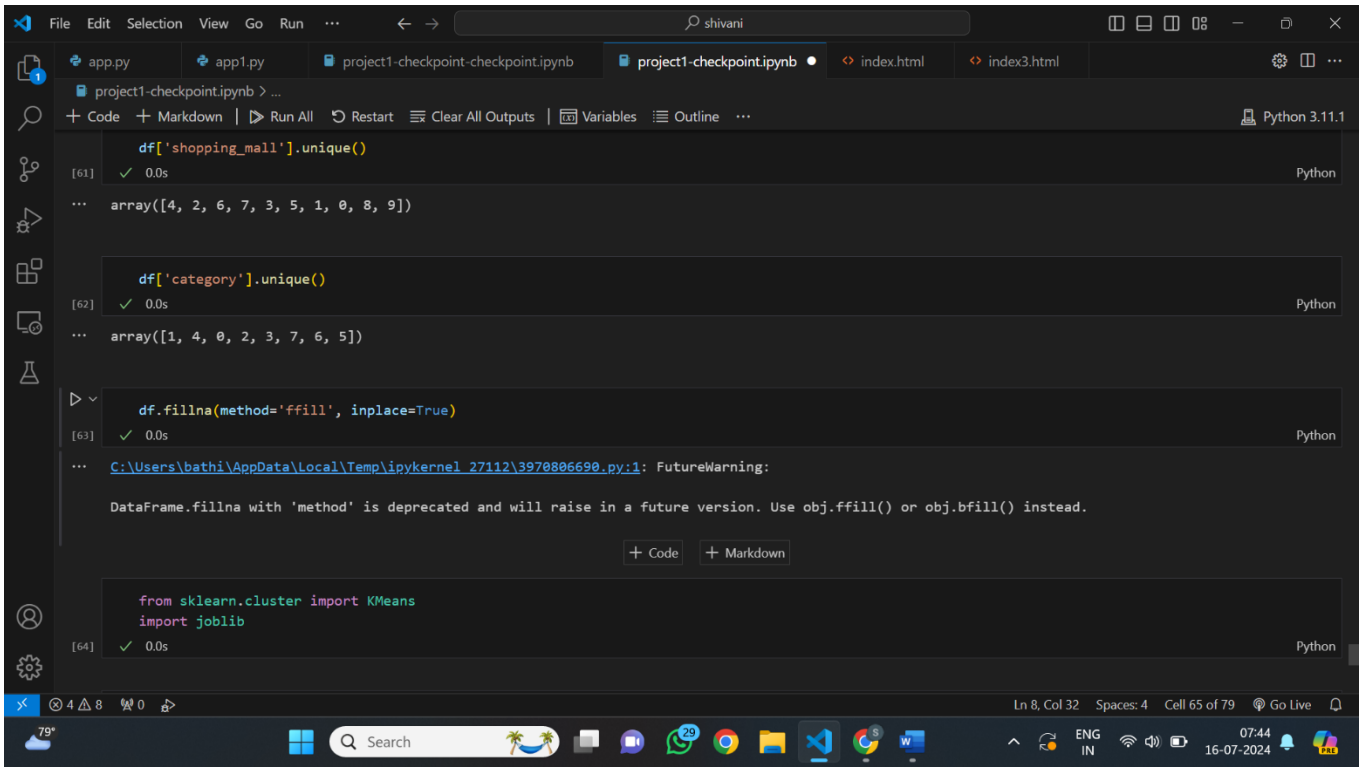
```
1.0
```

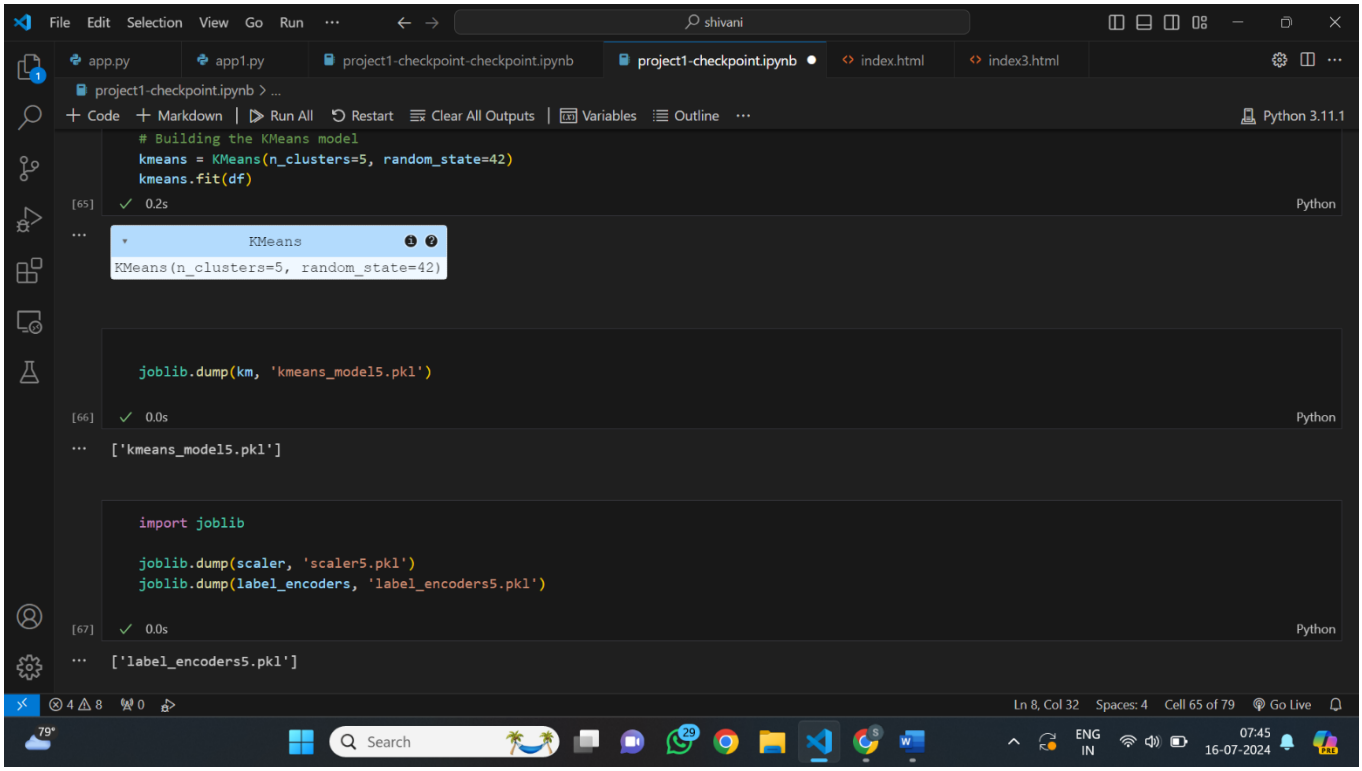
```
# Using the Elbow method to find the optimal number of clusters
sse = []
k_values = range(1, 11)
for k in k_values:
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_train)
    sse.append(kmeans.inertia_)
```

```
# Plotting the results
plt.figure(figsize=(10, 6))
plt.plot(k_values, sse, marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Sum of squared distances')
plt.title('Elbow Method For Optimal Number of Clusters')
plt.grid(True)
plt.show()
```




Elbow Method For Optimal Number of Clusters
Sum of squared distances
Number of clusters
700000
600000
500000
400000
300000
2
4
6
8
10


df['shopping_mall'].unique()
array([4, 2, 6, 7, 3, 5, 1, 0, 8, 9])
df['category'].unique()
array([1, 4, 0, 2, 3, 7, 6, 5])
df.fillna(method='ffill', inplace=True)
C:\Users\bathi\AppData\Local\Temp\ipykernel_27112\3970806690.py:1: FutureWarning:
DataFrame.fillna with 'method' is deprecated and will raise in a future version. Use obj.ffill() or obj.bfill() instead.
from sklearn.cluster import KMeans
import joblib


File Edit Selection View Go Run
shivani
app.py
app1.py
project1-checkpoint-checkpoint.ipynb
project1-checkpoint.ipynb
index.html
index3.html
project1-checkpoint.ipynb > ...
Code Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.11.1
# Building the KMeans model
kmeans = KMeans(n_clusters=5, random_state=42)
kmeans.fit(df)
[65] 0.2s
Python
KMeans
KMeans(n_clusters=5, random_state=42)
joblib.dump(km, 'kmeans_model5.pkl')
[66] 0.0s
Python
['kmeans_model5.pkl']
import joblib
joblib.dump(scaler, 'scaler5.pkl')
joblib.dump(label_encoders, 'label_encoders5.pkl')
[67] 0.0s
Python
['label_encoders5.pkl']
Ln 8, Col 32 Spaces: 4 Cell 65 of 79 Go Live
Search
ENG IN
07:45 16-07-2024