CUSTOMER SHOPPING SEGMENTATION

Submitted to

JAWAHARLAL NEHURU TECHNOLOGICAL UNVIERSITY, HYDERABAD

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

MASTER OF COMPUTER APPLICATIONS

IN

COMPUTER SCIENCE AND ENGINEERING(MCA)

Submitted By

ADUNURI LAVANYA (23UK1F0011)

Under the guidance of

Mrs. B. Anusha

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING(MCA)

VAAGDDEVI ENGINEERING COLLEGE

(AUTONOMOUS)

Affiliated to JNTU, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) 506005

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (MCA)

VAAGDEVI ENGINEERING COLLEGE(AUTONOMOUS)

<u>CERTIFICATE OF COMPLETION OF PROJECT WORK REVIEW-1</u>

This is to certify that the PG project phrase -1 entitled **"CUSTOMER SHOPPING SEGMENTATION**" is being submitted by **ADUNURI LAVANYA (23UK1F0011)** in partial fulfillment of the requirements for the award of the degree of Master of computer applications in computer science & engineering to Jawaharlal Nehru Technological University Hyderabad during academic year 2023-2024

Project guide	**HOD**
Mrs. B. Anusha	**Dr. R. Naveen Kumar**
(Assistant professor)	(Professor)

External

ACKNOWLEDGEMENT

I wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. Syed Musthak Ahmed,** principle, Vaagdevi Engineering college for making us available all the required assistance and for his support and inspiration to carry out this PG Project Phrase-1 in the institute.

I extend my heartfelt thanks to **DR. R. NAVEEN KUMAR,** Head of the Department of CSE , Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the PG project phrase-1.

I express heartfelt thanks to the guide, **B**. **ANUSHA**, Assistant professor, Department of MCA for her constant support and giving necessary guidance for completion of this PG Project Phase-1.

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

ADUNURI LAVANYA (23UK1F0011)

ABSTRACT

- The Market Segmentation Analysis Using ML aims to analyze the spending behaviour of customers and identify opportunities for growth. The data set consists of spending (gender, age, category, quantity, payment_method, shopping _mall , price) of the customer.
- Using unsupervised machine learning techniques, specifically clustering algorithms, the project seeks to group customers with similar spending patterns together. By identifying customer segments with distinct spending behaviours , the project aims to provide insights on how w businesses can tailor their marketing strategies and product offerings to better serve each customer segment. The project also aims to identify opportunities for growth, such as which products or product categories are underrepresented among customers, and which segments may be receptive to new product offerings.
- Overall, the project seeks to provide valuable insights for wholesale businesses on how to optimize their operations and increase customer satisfaction and retention.

TABLE OF CONTENT

1. INTRTODUCTION

Customer shopping segmentation using machine learning (ML) is a data-driven approach to categorize customers into distinct groups based on their behaviors, preferences, and characteristics. This segmentation allows businesses to understand their customer base better and tailor their marketing strategies and product offerings accordingly.

1.1.PURPOSE

The purpose of customer shopping segmentation is multifaceted and serves several key objectives for businesses looking to optimize their marketing strategies, improve customer satisfaction, and drive profitability. Here are the primary purposes of customer shopping segmentation:

1. **Targeted Marketing Strategies**: By segmenting customers into distinct groups based on their behaviors, preferences, and demographics, businesses can tailor their marketing efforts more effectively. Each segment may respond differently to various marketing messages, promotions, and channels. Targeted marketing allows businesses to allocate resources more efficiently and achieve higher returns on investment (ROI).
2. **Personalized Customer Experiences**: Segmentation enables businesses to offer personalized experiences to their customers. By understanding the unique needs and preferences of each segment, businesses can customize product recommendations, pricing strategies, and customer service interactions. This personalization enhances customer satisfaction and loyalty, as customers feel understood and valued by the brand.
3. **Improved Customer Retention**: Segmentation helps businesses identify high-value customers and those at risk of churn. By focusing on retaining valuable customers through targeted retention strategies (such as loyalty programs, special offers, or personalized communications), businesses can reduce churn rates and increase customer lifetime value (CLV).
4. **Optimized Product Development**: Understanding customer segments can provide insights into which products or services are most appealing to different groups. This knowledge guides product development efforts, allowing businesses to innovate and introduce offerings that meet the specific needs and preferences of their target segments.
5. **Enhanced Operational Efficiency**: Segmentation facilitates more efficient resource allocation and inventory management. Businesses can prioritize inventory based on demand from different segments, optimize pricing strategies to maximize profitability across segments, and streamline operations to better meet customer expectations.

6. **Market Differentiation and Competitiveness**: Segmentation helps businesses identify niche markets and unique selling propositions (USPs) that differentiate them from competitors. By catering to specific customer segments with tailored offerings and experiences, businesses can position themselves as leaders in their respective markets and attract a loyal customer base
7. **Data-Driven Decision Making**: Customer shopping segmentation relies on data analytics and machine learning techniques to uncover meaningful insights from large datasets. By leveraging data-driven insights, businesses can make informed decisions about marketing investments, operational strategies, and customer engagement initiatives, minimizing risks and maximizing opportunities for growth.

2. Objectives

- The principle objective of this task is the field of the cluster analysis.
- Clustering the mall customer based on their common characteristic like gender, age, interest, and spending habits.
- One extremely normal AI calculation that is appropriate for client division issues is the k-means clustering algorithm.
- K-means clustering efficiency scores are used to evaluate the quality of the clustering result obtained from the k-means algorithm.

3. LITERATURE SURVEY

3.1 EXISTING PROBLEM

- Customer shopping segmentation is having too many segmentation parameters, each customer and product segmentation implementation should be unique to your current business needs. It may not be a business best interest to implement standard customer parameters that do impact their overall customer analysis.
- Integrating data from various sources such as transactional data, browsing history, demographic data, and customer feedback poses challenges due to differences in formats, quality, and completeness. Sparse data, especially in e-commerce where not all customers make frequent purchases, can affect the reliability of segmentation models.

- Customer shopping behaviors can vary significantly based on seasons, trends, promotions, and external factors (e.g., economic conditions). Models need to capture and adapt to these temporal variations effectively. Customer preferences can change over time, necessitating segmentation models that can adapt dynamically rather than relying on static assumptions.
- Customer shopping segmentation Adhering to data privacy regulations (e.g., GDPR, CCPA) while utilizing customer data for segmentation purposes poses legal and ethical challenges. Machine learning models can inadvertently introduce biases based on demographic factors, purchasing history, or other characteristics, leading to unfair treatment of certain customer segments.

3.2. PROPOSED SYSTEM

1. Data Collection and Integration

- **Data Sources**: Gather data from various sources such as transactional records, website interactions, customer demographics, and feedback.
- **Data Cleaning and Preprocessing**: Address data quality issues, handle missing values, and normalize data to ensure consistency and accuracy..

2. Feature Engineering

- **Behavioral Features**: Extract features related to shopping behavior (e.g., frequency of purchases, average order value, browsing history).
- **Demographic and Psychographic Features**: Include demographic information (age, gender, location) and psychographic attributes (lifestyle, interests).

3. Machine Learning Models

- **Clustering Algorithms**: Utilize clustering techniques such as K-means, hierarchical clustering, or DBSCAN to group customers based on similarities in their shopping behaviors and preferences.

- **Dimensionality Reduction**: Apply techniques like PCA or t-SNE to reduce the dimensionality **Temporal Analysis**: Incorporate temporal dynamics to capture seasonal trends, promotions, and changes in customer preferences over time.

4. Dynamic Segmentation

- **Real-time Data Processing**: Implement mechanisms for real-time data ingestion and processing to adapt segmentation models dynamically based on current customer interactions.
- **Temporal Analysis**: Incorporate temporal dynamics to capture seasonal trends, promotions, and changes in customer preferences over time.
-

5. Personalized Recommendations

- **Segment-specific Strategies**: Develop personalized marketing strategies tailored to each customer segment identified through clustering.

- **Content Personalization**: Customize product recommendations, promotional offers, and marketing messages based on the preferences and behaviors of each segment.

6. Evaluation and Optimization

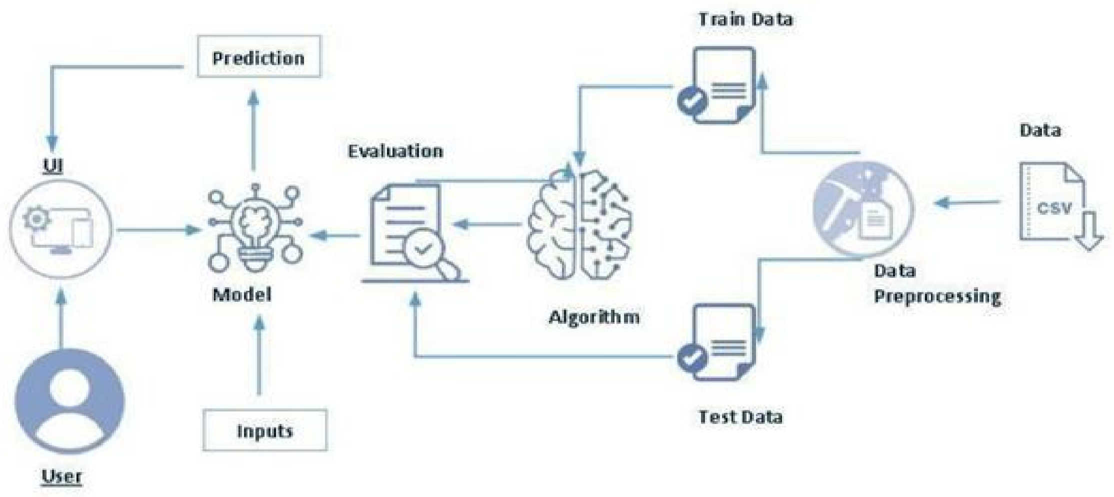
- **Performance Metrics**: Define key performance indicators (KPIs) such as conversion rates, retention rates, and customer lifetime value (CLV) to measure the effectiveness of segmentation.
- **A/B Testing**: Conduct experiments to validate the impact of segmentation strategies and iteratively optimize models based on results.

Benefits

- **Enhanced Customer Experience**: Deliver personalized shopping experiences that cater to the unique preferences of different customer segments.
- **Improved Marketing ROI**: Optimize marketing efforts by targeting specific customer segments with relevant offers and promotions.
- **Operational Efficiency**: Streamline decision-making processes with data-driven insights and automate repetitive tasks related to segmentation and marketing

4. THEORITICAL ANALYSIS

4.1. BLOCK DIAGRAM



4.2 SOFTWARE DESIGNING

- **VISUAL STUDIO:** Using Visual Studio Code (VS Code) for customer shopping segmentation involves leveraging its capabilities for coding, debugging, and managing projects related to data processing, machine learning model development, and user interface design. Here's a structured approach to using VS Code for this purpose
- **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 AQI 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 AQI 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 location data or view AQI predictions, health information, and recommended precautions.
- Visual studio code 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 AQI predictions and associated health information.

5. .EXPERIMENTAL INVESTIGATION

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

- **Invoice_no :** This column represents a unique identifier for each transaction or invoice. It helps in tracking individual purchases made by customers and is crucial for analyzing buying patterns and behaviors.
- **Customer_id :** This column identifies each customer uniquely within the dataset. Customer IDs allow businesses to link transactions to specific individuals, enabling analysis of individual customer behavior, preferences, and lifetime value.
- **Gender:** This column denotes the gender of the customer, which is a demographic variable. Gender can influence shopping preferences and product choices, making it relevant for segmenting customers based on their purchasing behavior.
- **age**: Age represents the age group or age range of the customer. It is another important demographic variable as shopping behaviors often vary across different age groups. Understanding age helps in targeted marketing and product offerings.
- **quantity**: This column indicates the quantity of each item purchased in a transaction. It provides insights into the volume of products customers buy, helping to identify trends in purchasing behavior and popular products.
- **payment_ method**: This column specifies the method used by the customer to pay for their purchase (e.g., credit card, cash, online payment, etc.). Payment method can influence shopping decisions and preferences, especially in terms of convenience and security.
- **invoice_ date**: This column records the date and time when the transaction took place. It is essential for analyzing seasonal trends, peak shopping periods, and the frequency of customer purchases over time.
- **Shopping_ mall:** This column indicates the shopping mall or store location where the purchase was made. It provides geographic context and can help understand shopping preferences based on location, store characteristics, or accessibility.
- **price**: Price represents the cost of each item purchased in the transaction. It is essential for analyzing customer spending patterns, average order value, and pricing sensitivity across different customer segments.

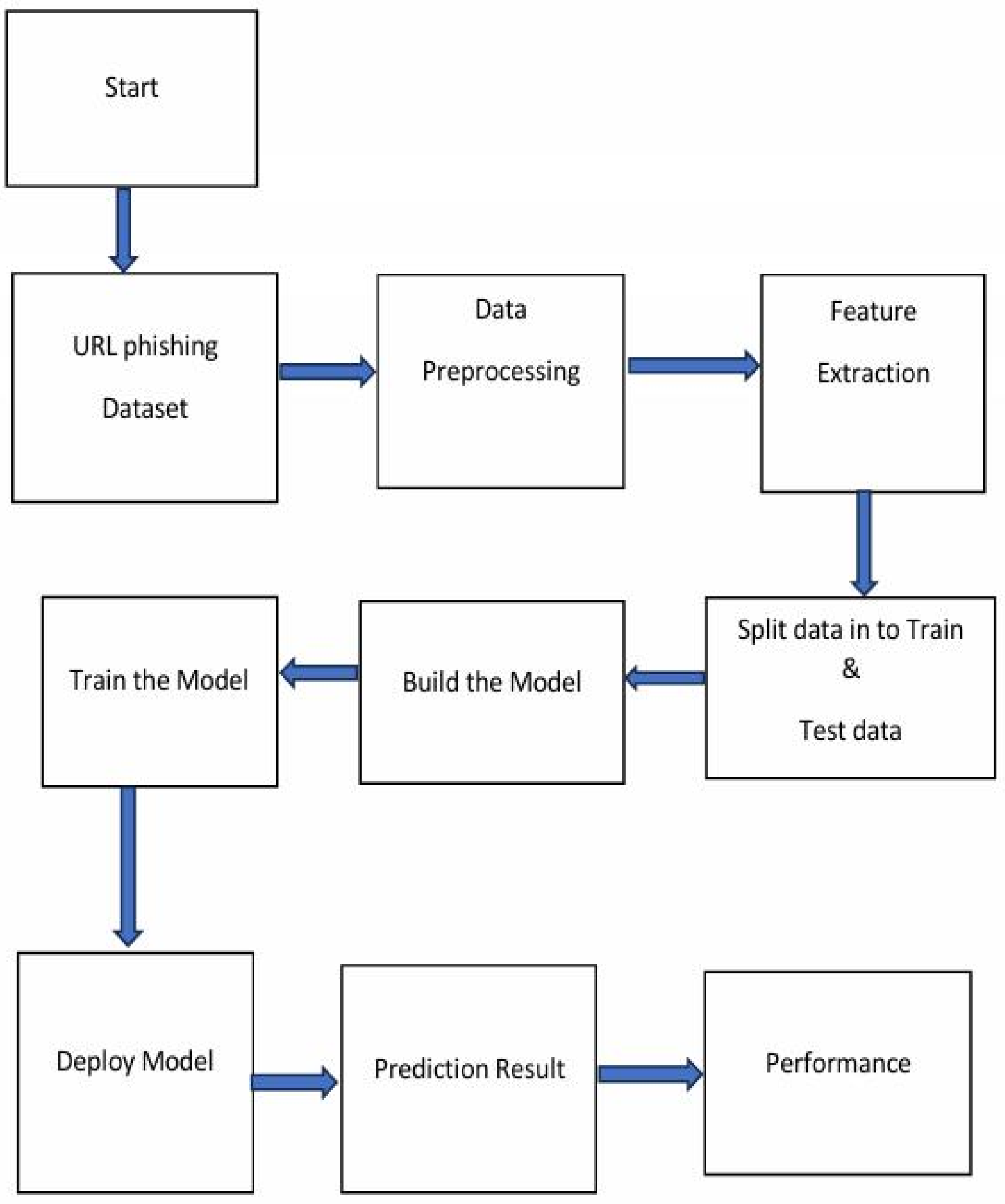
- **category**: This column categorizes the products purchased into different product categories (e.g., electronics, clothing, groceries, etc.). Categorization helps in understanding which types of products are popular among different customer segments and allows for targeted marketing strategies.

Each of these columns plays a crucial role in customer shopping segmentation by providing valuable information about customer demographics, behavior, purchasing patterns, and preferences. Analyzing these variables collectively enables businesses to create meaningful customer segments, tailor marketing strategies, optimize product offerings, and enhance overall customer satisfaction and loyalty.

For the dataset we selected, it consists of more than the columns we want to predict it . So, we have chosen the feature drop it contains the columns that we are going to predict.

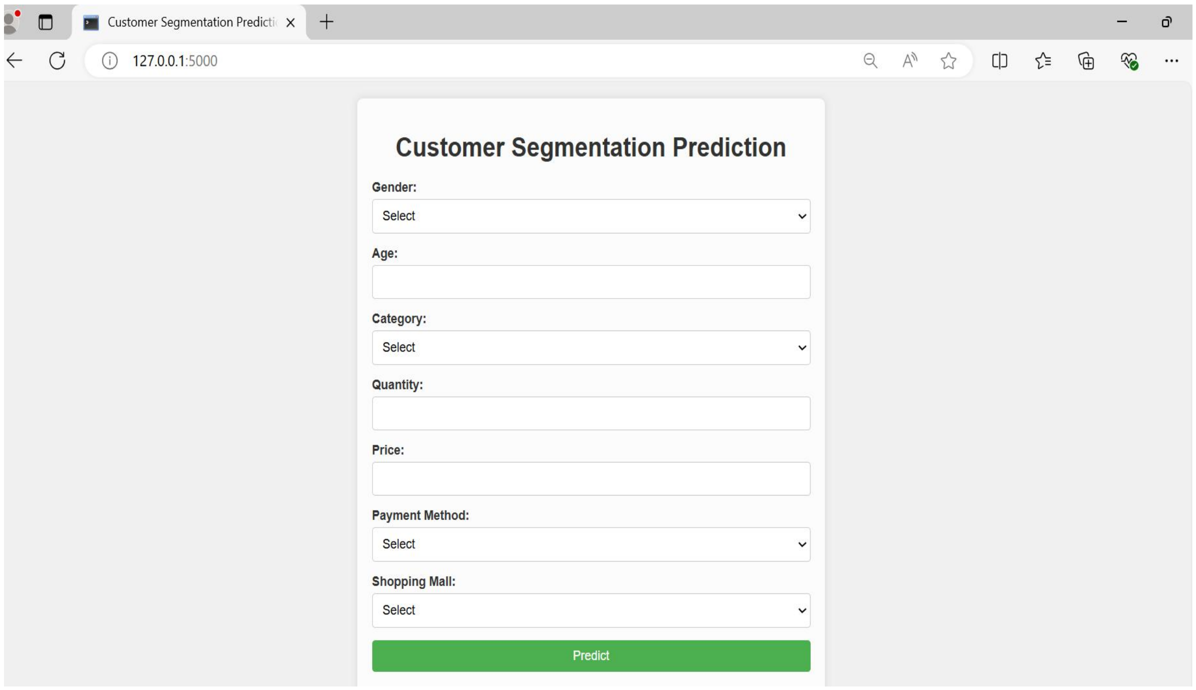
- Feature drop means it drops the columns that we don't want in our dataset.
- `df.drop(columns=["customer_id","invoice_no","invoice_date"])`

6. FLOW CHART


Start
URL phishing Dataset
Data Preprocessing
Feature Extraction
Split data in to Train & Test data
Build the Model
Train the Model
Deploy Model
Prediction Result
Performance

7. RESULT

HOME PAGE



Customer Segmentation Prediction

Gender: Select

Age:

Category: Select

Quantity:

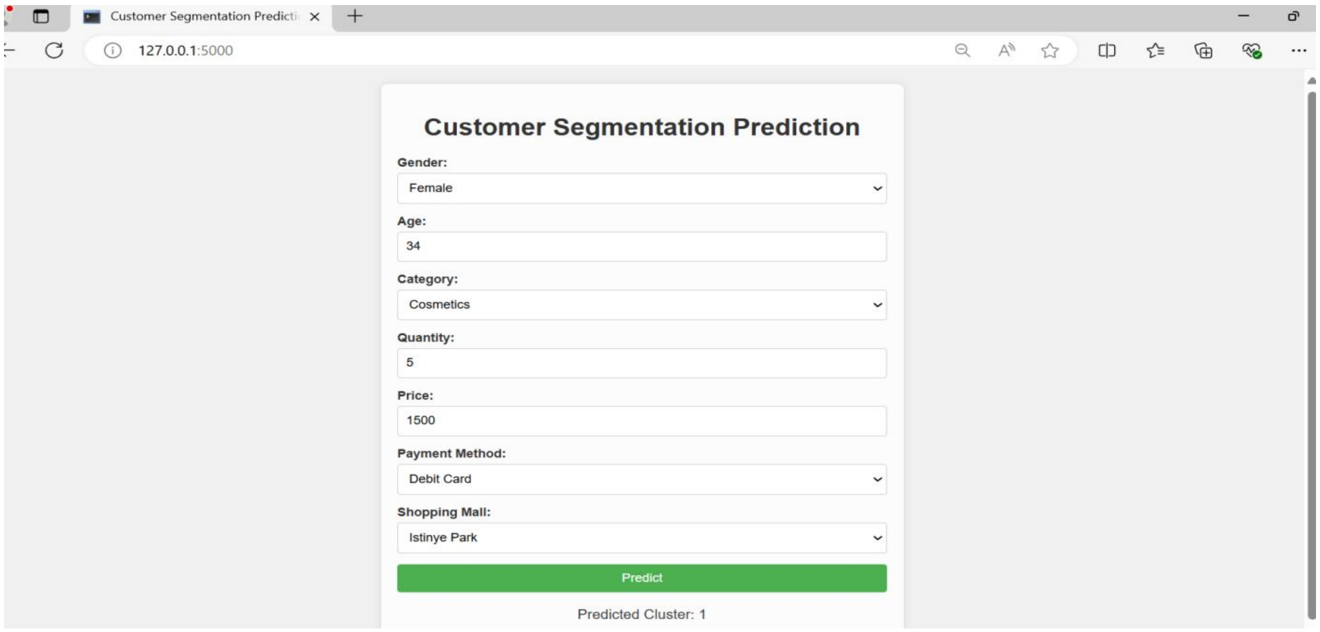
Price:

Payment Method: Select

Shopping Mall: Select

Predict

RESULT



Customer Segmentation Prediction

Gender: Female

Age: 34

Category: Cosmetics

Quantity: 5

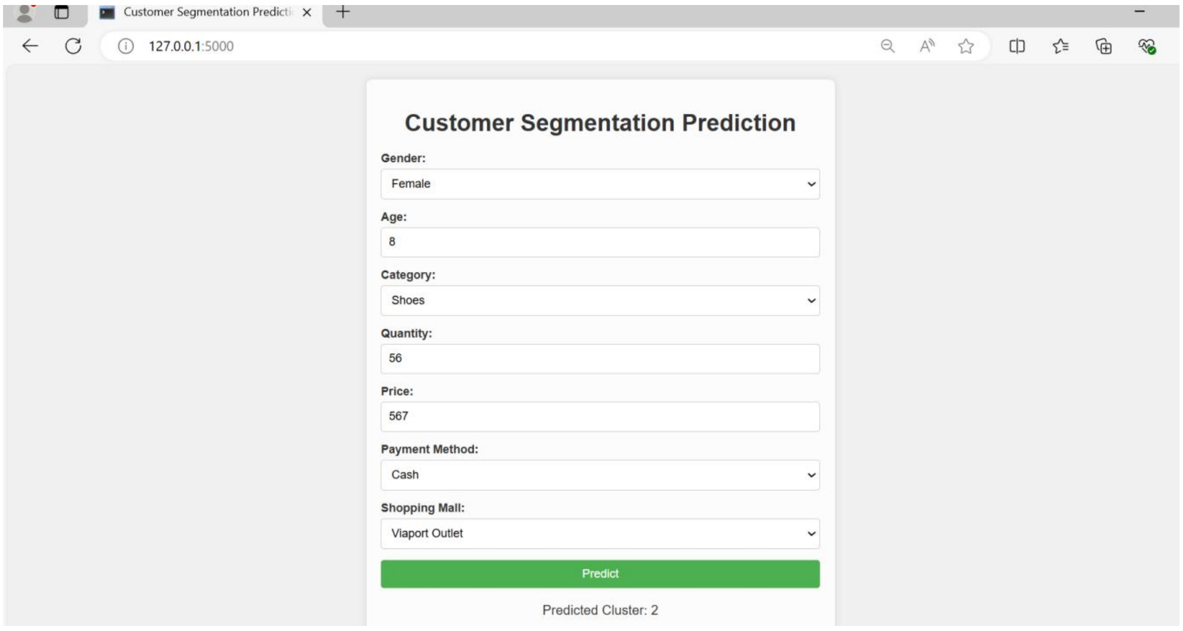
Price: 1500

Payment Method: Debit Card

Shopping Mall: Istinye Park

Predict

Predicted Cluster: 1



Customer Segmentation Prediction

Gender: Female

Age: 8

Category: Shoes

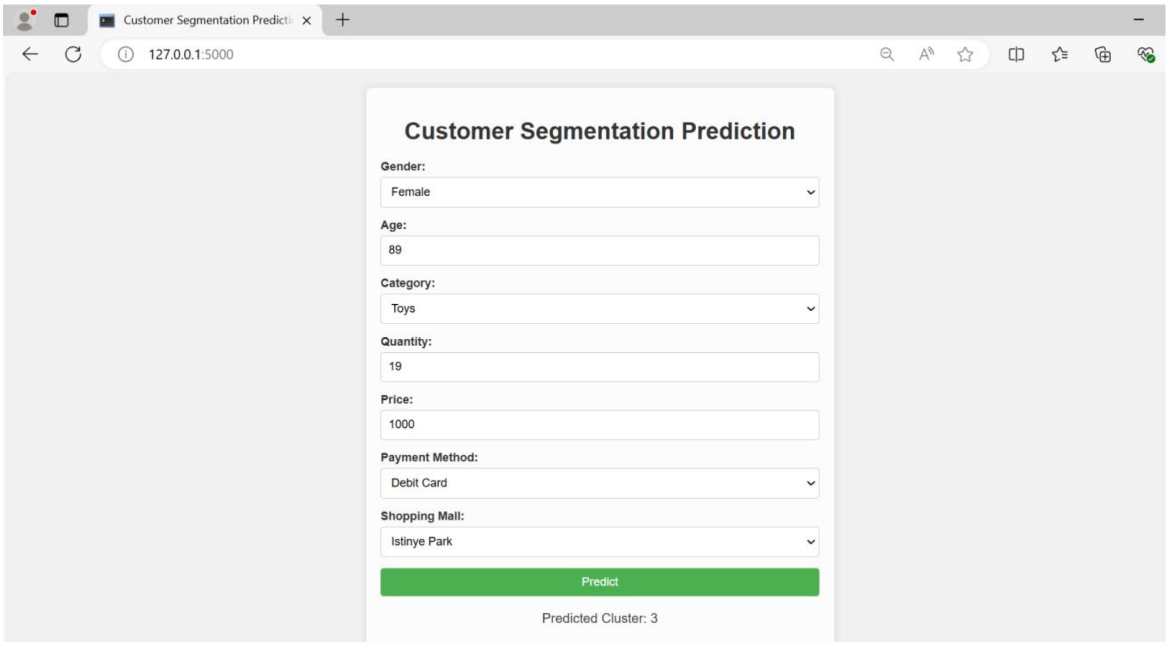
Quantity: 56

Price: 567

Payment Method: Cash

Shopping Mall: Viaport Outlet

Predicted Cluster: 2



Customer Segmentation Prediction

Gender: Female

Age: 89

Category: Toys

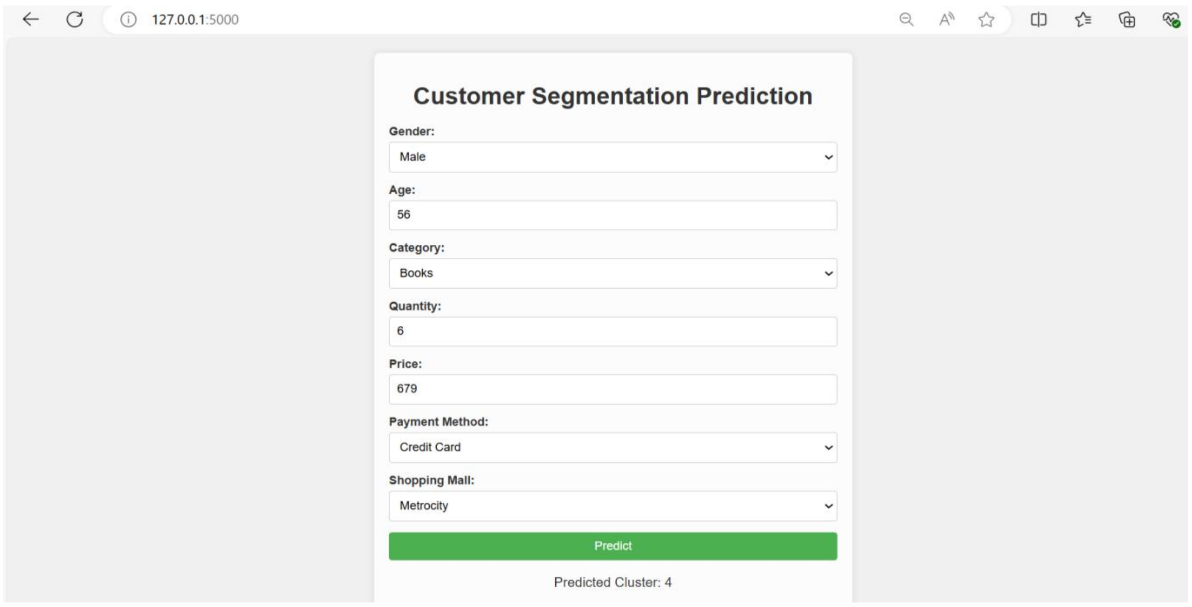
Quantity: 19

Price: 1000

Payment Method: Debit Card

Shopping Mall: Istinye Park

Predicted Cluster: 3



Customer Segmentation Prediction

Gender: Male

Age: 56

Category: Books

Quantity: 6

Price: 679

Payment Method: Credit Card

Shopping Mall: Metrocity

Predicted Cluster: 4

8.ADVANTAGES AND DISADVANTAGES

Advantages:

1. **Targeted Marketing**: Segmentation allows businesses to target specific customer groups with tailored marketing messages, promotions, and product offerings. This increases the relevance of marketing efforts and improves conversion rates.
2. **Improved Customer Satisfaction**: By understanding the unique needs and preferences of different customer segments, businesses can enhance the shopping experience. This leads to higher customer satisfaction and loyalty.
3. **Optimized Resource Allocation**: Segmentation helps allocate resources (such as marketing budgets and inventory) more efficiently. Businesses can focus investments where they are likely to yield the highest returns, minimizing wastage.
4. **Competitive Advantage**: Effective segmentation enables businesses to differentiate themselves from competitors by offering products and services that better meet the needs of specific customer groups.
5. **Insights for Product Development**: Understanding customer segments provides valuable insights for product development and innovation. Businesses can identify trends and preferences that guide new product offerings.

Disadvantages:

1. **Complexity and Cost**: Implementing and maintaining a segmentation strategy can be complex and costly. It requires expertise in data analysis, market research, and potentially technology (if using advanced segmentation methods).
2. **Segment Overlap**: There can be overlap between customer segments, making it challenging to create distinct marketing strategies for each group. This requires careful management to avoid confusion or inefficiencies.
3. **Changing Customer Behavior**: Customer preferences and behaviors can change over time, necessitating regular updates to segmentation strategies. Failure to adapt can lead to outdated marketing approaches.
4. **Privacy Concerns**: Gathering and using customer data for segmentation purposes raises privacy concerns. Businesses must comply with data protection regulations and handle customer data responsibly to maintain trust.
5. **Risk of Stereotyping**: Segmentation based on demographic or behavioral data runs the risk of stereotyping customers, leading to generalized assumptions that may not accurately reflect individual preferences.

In summary, while customer shopping segmentation offers significant benefits such as targeted marketing and improved customer satisfaction, businesses must navigate challenges related to complexity, cost, evolving customer behaviors, privacy, and the potential for oversimplification or stereotyping. Effective segmentation requires a balanced approach that considers both the opportunities and risks involved.

9.APPLICATIONS

Customer shopping segmentation can be used in various applications to help businesses better understand and cater to their customers. Some applications include:

1.**Targeted Marketing:** By segmenting customers based on their shopping behavior, preferences, and demographics, businesses can create targeted marketing campaigns. This helps in delivering personalized messages and offers to specific customer segments, increasing the likelihood of conversion.

2.Product Recommendations: Segmentation allows businesses to recommend products or services based on the preferences and past purchases of different customer segments. This personalized approach enhances the shopping experience and increases customer satisfaction.

3. Inventory Management: Understanding different customer segments can help businesses optimize their inventory by stocking products that are popular among specific segments. This reduces excess inventory and ensures that the right products are available to meet customer demand.

4. Customer Retention: Segmentation helps in identifying high-value customers who are more likely to make repeat purchases. By focusing on retaining these customers through loyalty programs or special offers, businesses can improve customer retention rates and increase customer lifetime value.

5. Pricing Strategies: Businesses can use segmentation to implement dynamic pricing strategies based on different customer segments. This allows for pricing adjustments to maximize revenue while remaining competitive in the market**.**

6. Store Layout and Design: Segmentation can also be used to optimize store layout and design to cater to the preferences of different customer segments. By creating a personalized shopping experience, businesses can enhance customer satisfaction and encourage repeat visits.

Overall, customer shopping segmentation is a valuable tool for businesses to enhance their marketing strategies, improve customer satisfaction, and drive growth.

10.CONCLUSION

Segmenting customers for targeted marketing strategies is crucial in today's competitive landscape. By understanding distinct shopping behaviors and preferences, businesses can personalize their approach, enhance customer satisfaction, and ultimately drive growth. Through effective segmentation, companies can allocate resources efficiently, tailor offerings to specific needs, and foster long-term customer loyalty. This strategic approach not only improves marketing ROI but also strengthens relationships with customers, positioning businesses for sustained success in the mark

11.FUTURE SCOPE

1. **Advanced Data Analytics**: As technology evolves, so does the ability to collect, process, and analyze vast amounts of data. Machine learning and AI algorithms will enable deeper insights into customer behavior, allowing for more precise segmentation based on real-time and predictive analytics.

2. **Personalization at Scale**: The focus will shift towards hyper-personalization, where segmentation goes beyond demographic and behavioral data to include individual preferences, interests, and even emotions. This will enable tailored marketing strategies that resonate on a deeply personalized level.

3. **Omni-Channel Integration**: Seamless integration across online and offline channels will become paramount. Customer segmentation will play a crucial role in delivering consistent and personalized experiences regardless of the touch point, enhancing overall customer satisfaction and loyalty.

4. **Dynamic Segmentation**: Segmentation models will become more dynamic and adaptive, continuously updating based on real-time interactions and feedback. This agility will allow businesses to respond quickly to changing market conditions and customer behaviors.

5. **Behavioral Economics Insights**: Integration of behavioral economics principles into segmentation strategies will provide deeper understanding of decision-making processes, enabling businesses to influence customer behavior more effectively.

12.APPENDIX

Model building

1) Dataset

2) Google colab and VS code Application Building

1. HTML file (Index file, Predict file)
2. CSS file
3. Models in pickle format

SOURCE CODE:

Home.html

```
# Example of generating HTML report
html_content = '''
<!DOCTYPE html>
<html>
<head>
    <title>Customer Segmentation Report</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        h1 {
            color: #333;
            text-align: center;
        }
        .cluster {
            border: 1px solid #ccc;
            padding: 10px;
            margin-bottom: 20px;
        }
        .cluster h2 {
            color: #555;
        }
        .cluster table {
            width: 100%;
            border-collapse: collapse;
        }
        .cluster th, .cluster td {
            border: 1px solid #ddd;
            padding: 8px;
            text-align: left;
        }
        .cluster th {
            background-color: #f2f2f2;
        }
    </style>
</head>
<body>
    <h1>Customer Segmentation Report</h1>
'''

# Add cluster-wise analysis to HTML
for cluster_id, cluster_data in data.groupby('Cluster'):
    html_content += f'''
    <div class="cluster">
        <h2>Cluster {cluster_id}</h2>
        <table>
            <tr>
                <th>Customer ID</th>
                <th>Feature 1</th>
                <th>Feature 2</th>
                <!-- Add more columns as needed -->
            </tr>
    '''
    for index, row in cluster_data.iterrows():
        html_content += f'''
            <tr>
                <td>{row['Customer ID']}</td>
                <td>{row['Feature1']}</td>
                <td>{row['Feature2']}</td>
                <!-- Add more columns as needed -->
            </tr>
        '''
    html_content += '''
        </table>
    </div>
    '''

# Close the HTML document
html_content += '''
</body>
</html>
'''

# Save HTML content to a file
```

```
with open('customer_segmentation_report.html', 'w') as file:
    file.write(html_content)

print('HTML report generated successfully!')
```

index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Customer Segmentation</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <div class="container">
        <h1>Customer Segmentation</h1>
        <form action="/segmentation" method="post">
            <input type="submit" value="Segment Customers">
        </form>
    </div>
</body>
</html>
```

Index.py

```
<!DOCTYPE html>
<html>
<head>
    <title>Customer Segmentation</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <div class="container">
        <h1>Customer Segmentation</h1>
        <form action="/segmentation" method="post">
            <input type="submit" value="Segment Customers">
        </form>
    </div>
</body>
</html>
```

Index2.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Customer Segmentation Prediction</title>
```

```
</head>
<body>
    <h1>Customer Segmentation Prediction</h1>
    <form id="prediction-form">
        <label for="customer_gender">Gender:</label>
        <input type="text" id="customer_gender" name="customer_gender"><br>
        <label for="age">Age:</label>
        <input type="number" id="age" name="age"><br>
        <label for="quantity">Quantity:</label>
        <input type="number" id="quantity" name="quantity"><br>
        <label for="price">Price:</label>
        <input type="number" id="price" name="price"><br>
        <label for="payment_method">Payment Method:</label>
        <input type="text" id="payment_method" name="payment_method"><br>
        <label for="shopping_mall">Shopping Mall:</label>
        <input type="text" id="shopping_mall" name="shopping_mall"><br>
        <input type="submit" value="Predict">
    </form>
    <p id="prediction-result"></p>

    <script>
        document.getElementById('prediction-form').addEventListener('submit',
function(event) {
            event.preventDefault();

            const formData = new FormData(event.target);
            const data = {};
            formData.forEach((value, key) => data[key] = value);

            fetch('/predict', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify(data)
            })
            .then(response => response.json())
            .then(result => {
                document.getElementById('prediction-result').innerText =
`Predicted Category: ${result.predicted_category}`;
            })
            .catch(error => console.error('Error:', error));
        });
    </script>
</body>
</html>
```

Index3.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Customer Segmentation Prediction</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
            margin: 0;
            padding: 0;
            /* Set background image */
            background-image:
url('D:\MiniProjects24\CSP\static\customer_segmentation.jpg');
            background-size: cover;
            background-position: center;
            /* Ensuring content is readable */
            color: #333;
        }

        .container {
            max-width: 600px;
            margin: 20px auto;
            background-color: rgba(255, 255, 255, 0.8); /* Semi-transparent white
background for readability */
            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="D:\MiniProjects24\CSP\static\D:\MiniProjects24\CSP\staticcustomer_segmentati
on.jpg" alt="Customer Segmentation" class="logo" -->
        <h1>Customer Segmentation Prediction</h1>
        <form id="prediction-form">
            <div class="form-group">
                <label for="gender">Gender:</label>
                <select id="gender" name="gender" required>
                    <option value="">Select</option>
```

```html
                <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="category">Category:</label>
            <select id="category" name="category" required>
                <option value="">Select</option>
                <option value="Clothing">Clothing</option>
                <option value="Shoes">Shoes</option>
                <option value="Books">Books</option>
                <option value="Cosmetics">Cosmetics</option>
                <option value="Food & Beverage">Food & Beverage</option>
                <option value="Toys">Toys</option>
                <option value="Technology">Technology</option>
                <option value="Souvenir">Souvenir</option>
            </select>
        </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="submit">Predict</button>
            </form>
            <p id="prediction-result"></p>
        </div>

        <script>
            document.getElementById('prediction-form').addEventListener('submit',
function(event) {
                event.preventDefault();

                const formData = new FormData(event.target);
                const data = {};
                formData.forEach((value, key) => data[key] = value);

                fetch('/predict', {
                    method: 'POST',
                    headers: {
                        'Content-Type': 'application/json'
                    },
                    body: JSON.stringify(data)
                })
                .then(response => response.json())
                .then(result => {
                    document.getElementById('prediction-result').innerText =
`Predicted Cluster: ${result.predicted_cluster}`;
                })
                .catch(error => console.error('Error:', error));
            });
        </script>
</body>
</html>
```

Segmentation_result

```
<!DOCTYPE html>
<html>
<head>
    <title>Segmentation Result</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <div class="container">
        <h1>Segmentation Result</h1>
        <h2>Segmented Customer Data</h2>
        {% if table %}
            {{ table | safe }}
        {% endif %}
        <h2>Segmentation Plot</h2>
        {% if plot_image %}
            <img src="{{ url_for('static', filename=plot_image) }}"
alt="Segmentation Plot">
        {% endif %}
    </div>
</body>
</html>
```

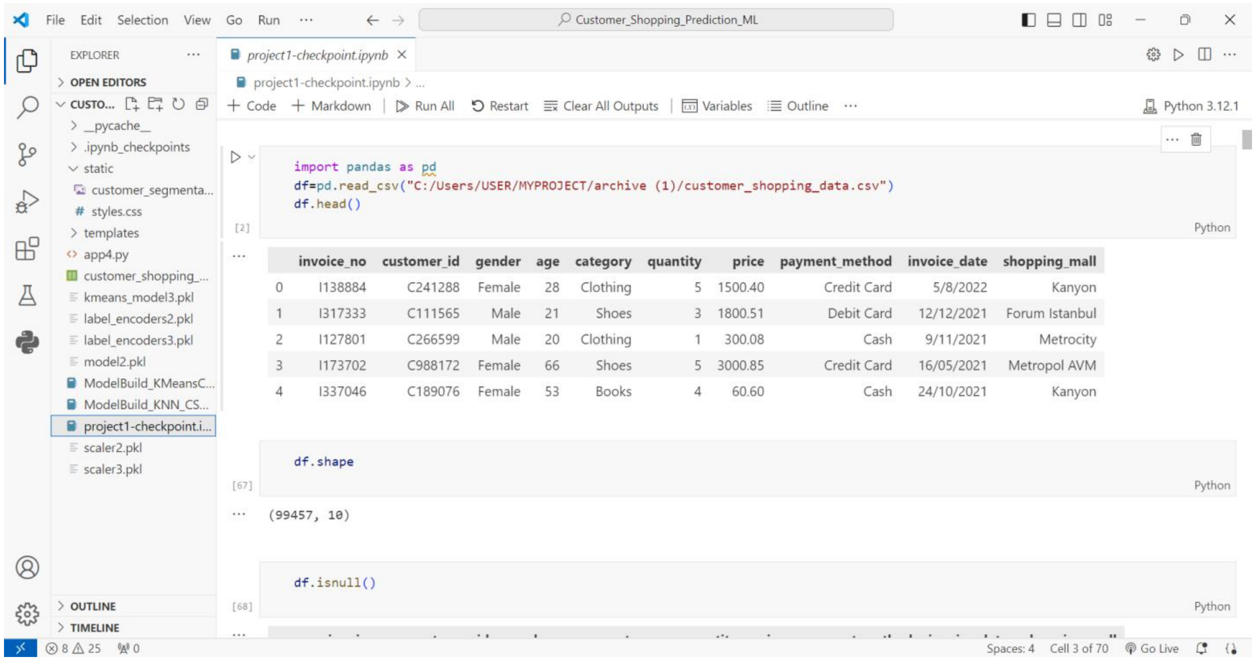
App.py

```
# -*- coding: utf-8 -*-
"""
Created on Wed Jul  3 20:02:21 2024

@author: madha
"""

from flask import Flask, request, jsonify, render_template
import pandas as pd
import joblib

app = Flask(__name__)

# Load the model, scaler, and label encoders
model = joblib.load('kmeans_model3.pkl')
scaler = joblib.load('scaler3.pkl')
label_encoders = joblib.load('label_encoders3.pkl')

@app.route('/')
def welcom():
    return render_template('index3.html')

@app.route('/predict', methods=['POST'])
def predict():
    data = request.json

    # Convert data into DataFrame
    df = pd.DataFrame(data, index=[0])
    # Encoding categorical variables
    for column in ['gender', 'category', 'payment_method', 'shopping_mall']:
        le = label_encoders[column]
        df[column] = le.transform(df[column])

    # Feature scaling
    df[['age', 'quantity', 'price']] = scaler.transform(df[['age', 'quantity',
'price']])

    # Making predictions
    prediction = model.predict(df)

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

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

IMPORTING LIBRARIES

```
import numpy as np
import pandas as pd
from numpy import math
import seaborn as sns
from datetime import datetime
import warnings
from pylab import rcParams
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
warnings.filterwarnings('ignore')
```

READ DATASET



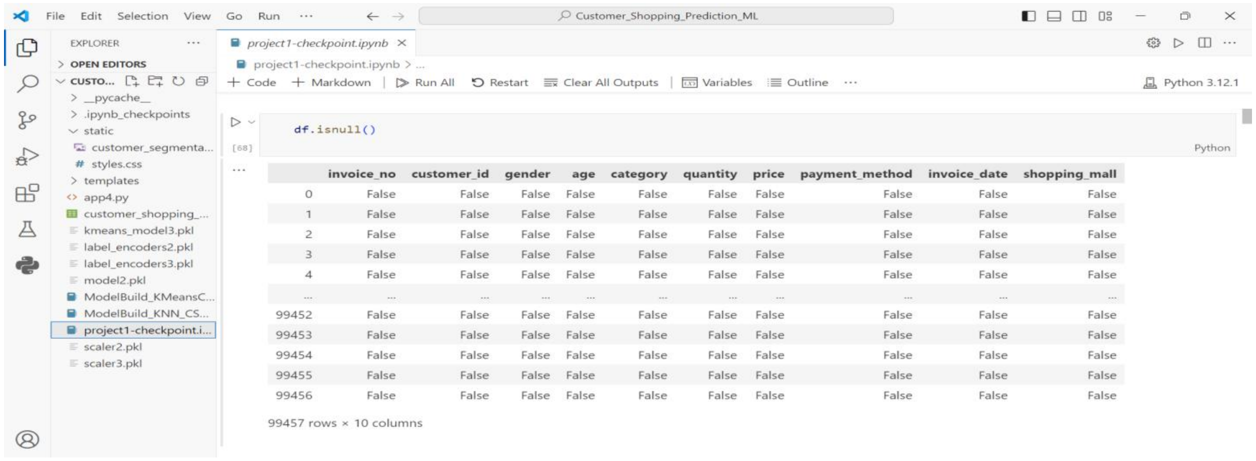
```
import pandas as pd
df=pd.read_csv("C:/Users/USER/MYPROJECT/archive (1)/customer_shopping_data.csv")
df.head()
```

```
   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
```

```
df.shape
```

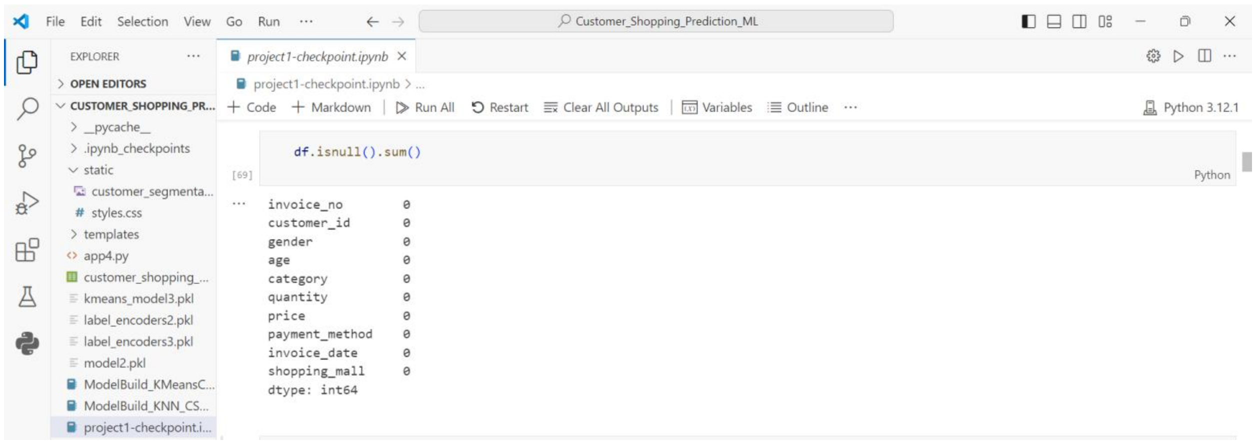
```
(99457, 10)
```

```
df.isnull()
```



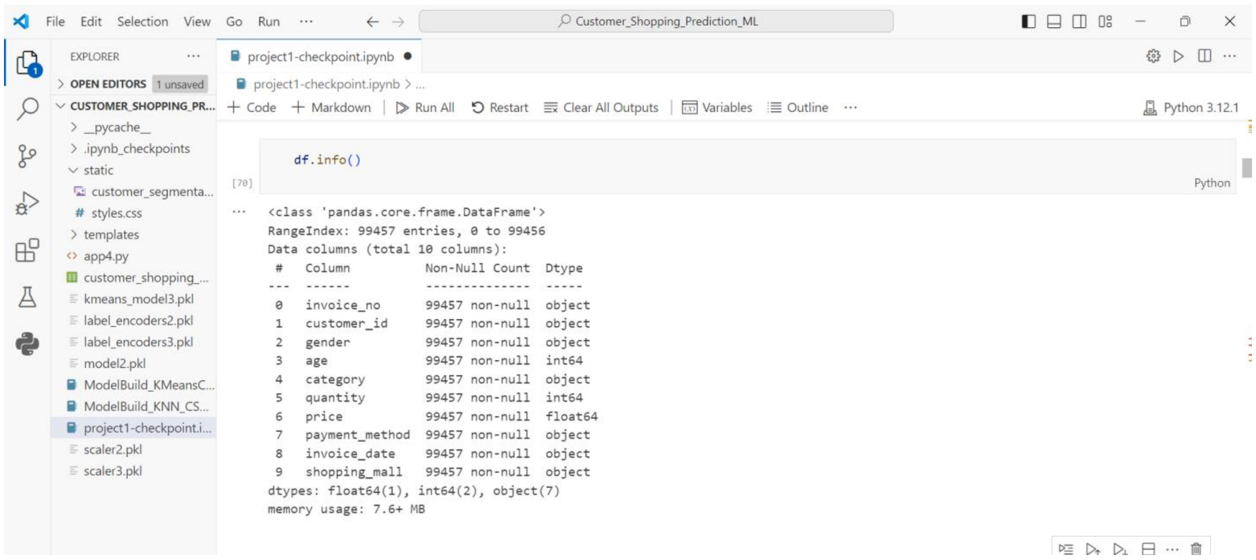
```
df.isnull()
```

```
       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
4           False        False   False  False     False     False  False           False         False          False
...           ...          ...     ...    ...       ...       ...    ...             ...           ...            ...
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
```



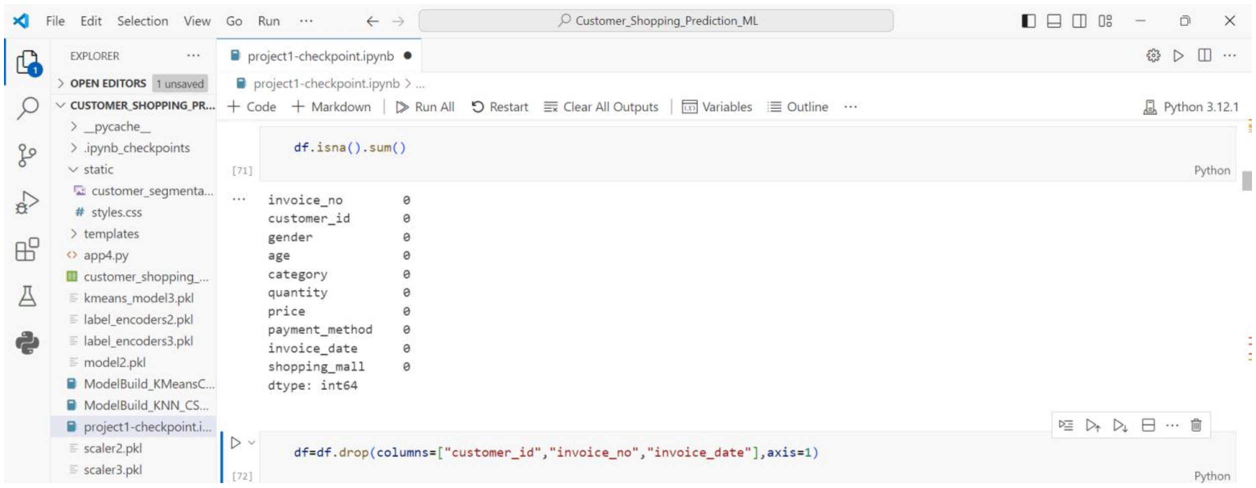
```python
df.isnull().sum()
```

```
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
```


File Edit Selection View Go Run
Customer_Shopping_Prediction_ML
OPEN EDITORS 1 unsaved
scaler2.pkl
scaler3.pkl
Python 3.12.1

```python
df.info()
```

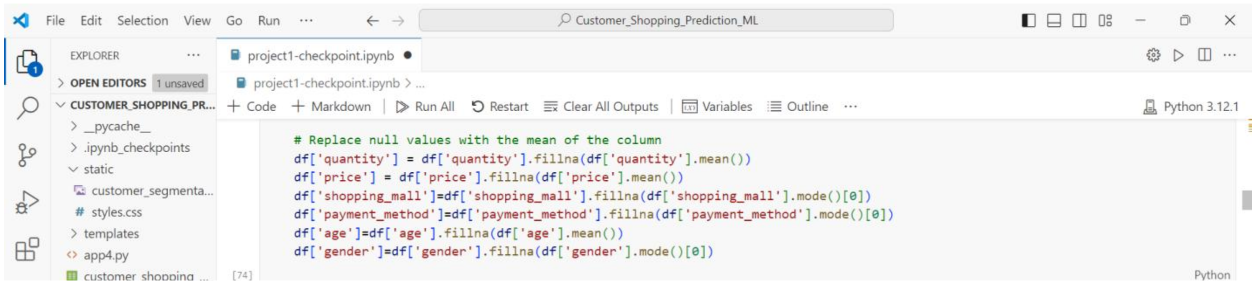
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99457 entries, 0 to 99456
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   invoice_no      99457 non-null  object 
 1   customer_id     99457 non-null  object 
 2   gender          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
```


File Edit Selection View Go Run
Customer_Shopping_Prediction_ML
OPEN EDITORS 1 unsaved
Python 3.12.1

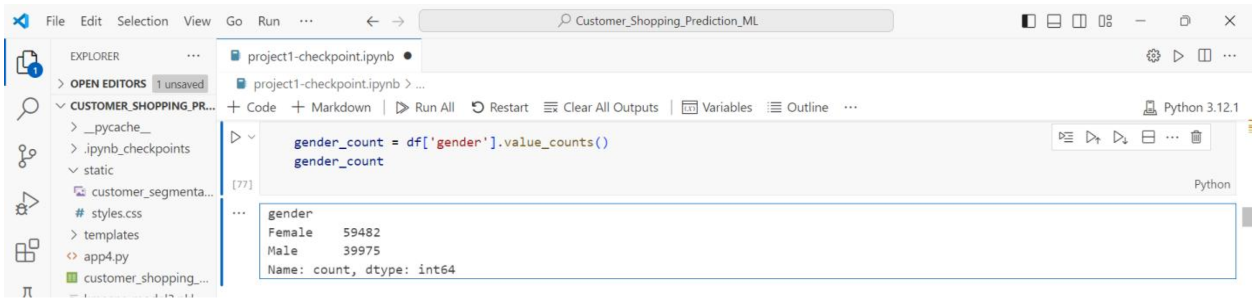
```python
df.isna().sum()
```

```
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
```

```python
df=df.drop(columns=["customer_id","invoice_no","invoice_date"],axis=1)
```

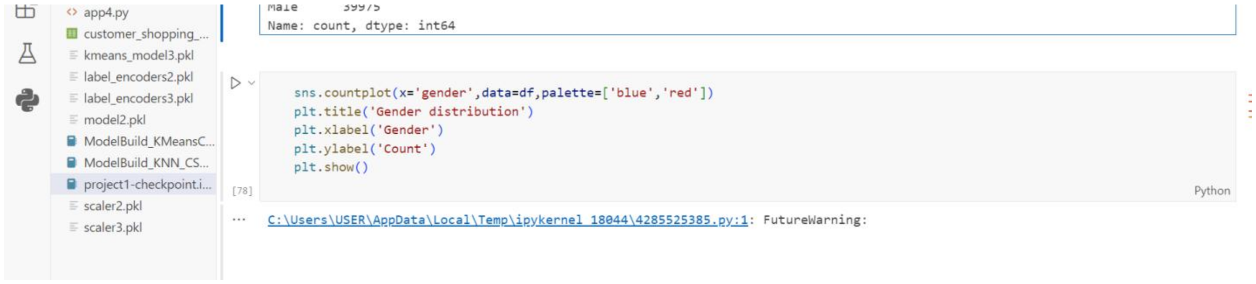


```python
# Replace null values with the mean of the column
df['quantity'] = df['quantity'].fillna(df['quantity'].mean())
df['price'] = df['price'].fillna(df['price'].mean())
df['shopping_mall']=df['shopping_mall'].fillna(df['shopping_mall'].mode()[0])
df['payment_method']=df['payment_method'].fillna(df['payment_method'].mode()[0])
df['age']=df['age'].fillna(df['age'].mean())
df['gender']=df['gender'].fillna(df['gender'].mode()[0])
```



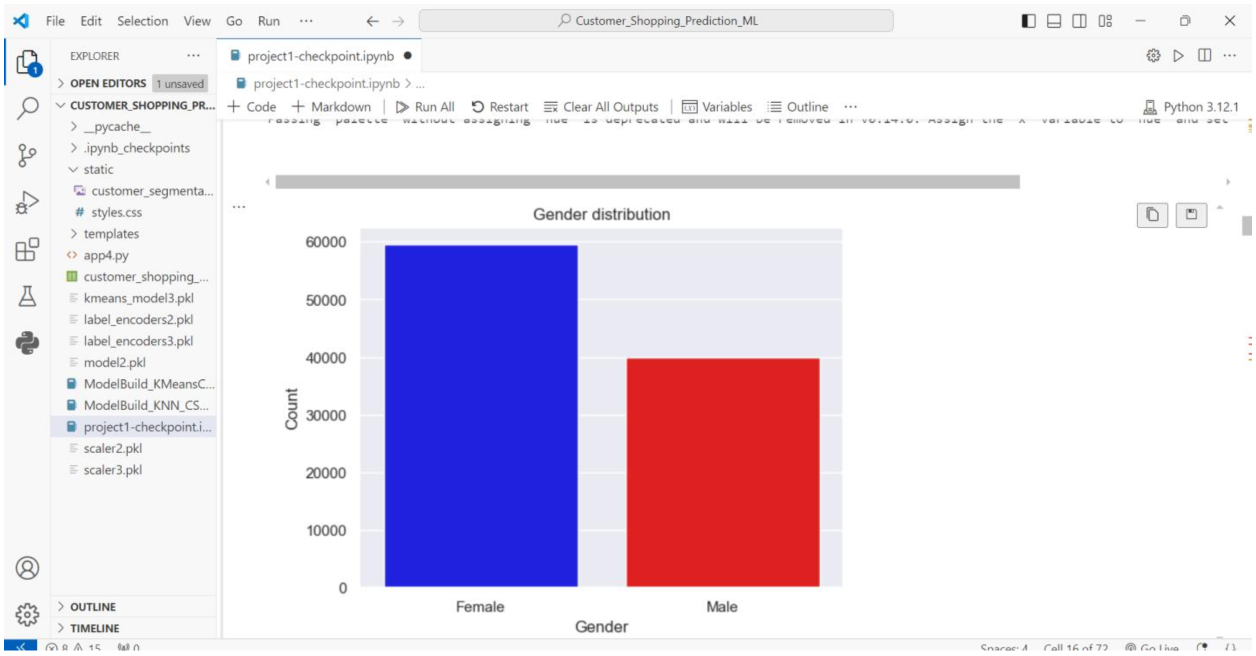
```python
gender_count = df['gender'].value_counts()
gender_count
```

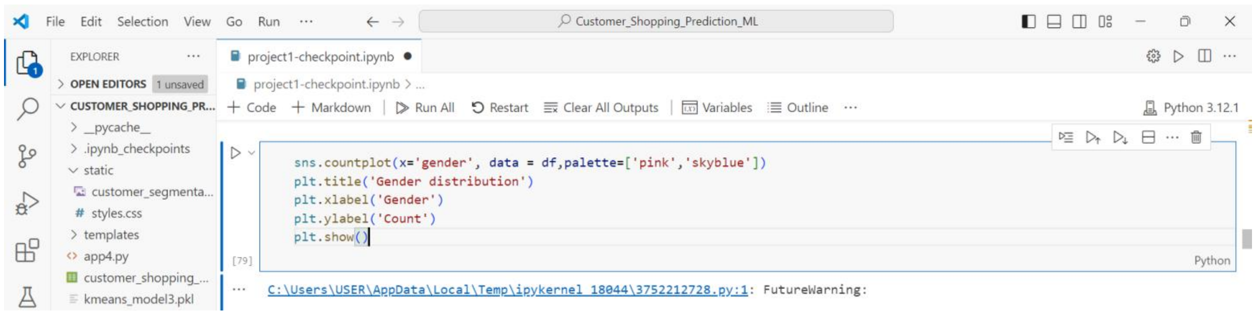
```
gender
Female    59482
Male      39975
Name: count, dtype: int64
```



```python
sns.countplot(x='gender',data=df,palette=['blue','red'])
plt.title('Gender distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```

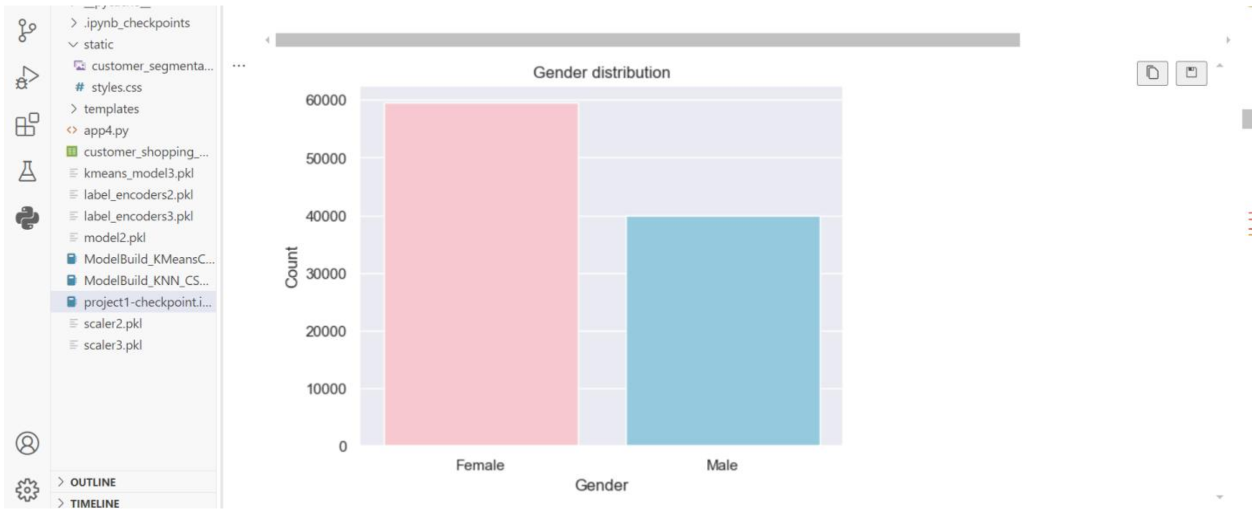
```
C:\Users\USER\AppData\Local\Temp\ipykernel_18044\4285525385.py:1: FutureWarning:
```

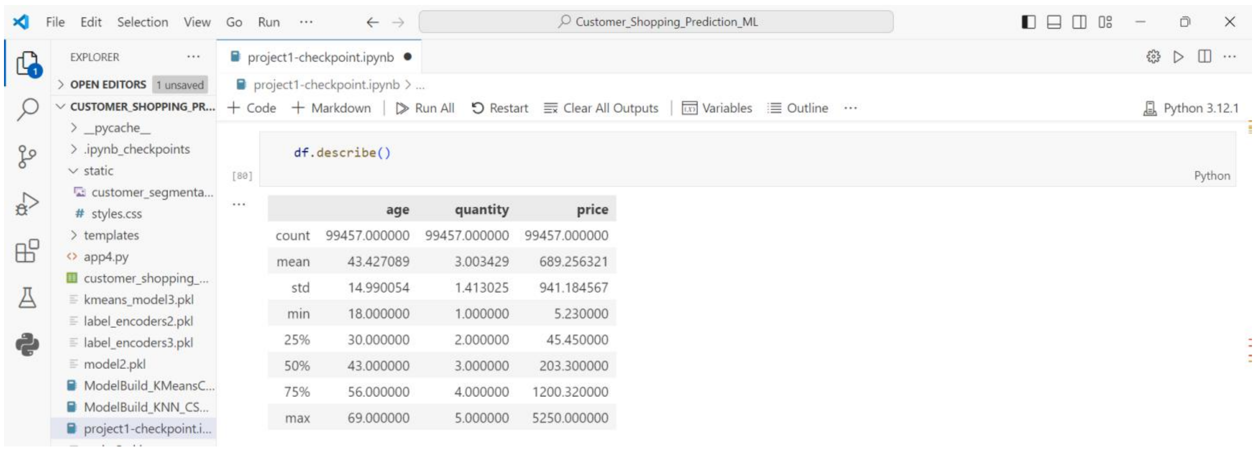




```
sns.countplot(x='gender', data = df,palette=['pink','skyblue'])
plt.title('Gender distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```

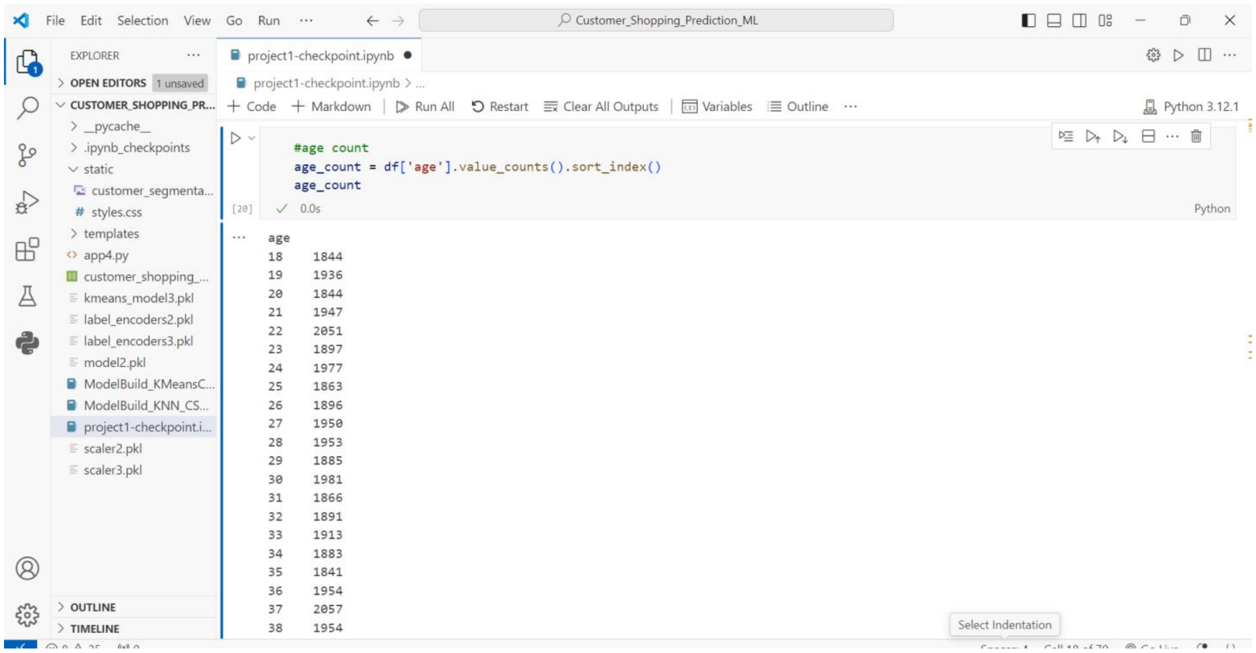
```
C:\Users\USER\AppData\Local\Temp\ipykernel_18044\3752212728.py:1: FutureWarning:
```





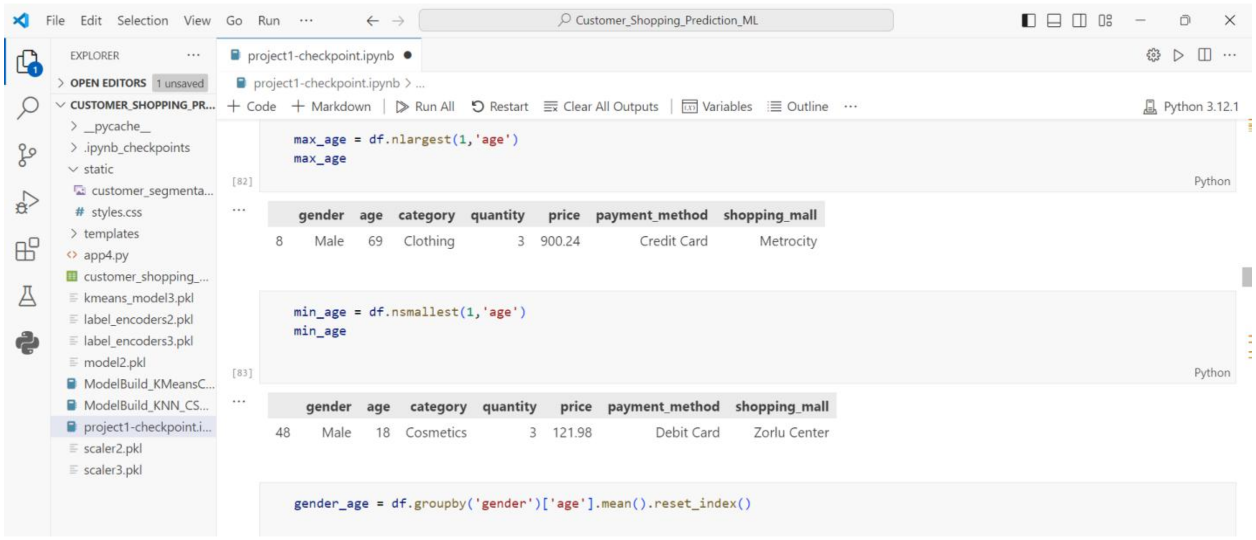
```
df.describe()
```

```
                age      quantity         price
count  99457.000000  99457.000000  99457.000000
mean      43.427089      3.003429    689.256321
std       14.990054      1.413025    941.184567
min       18.000000      1.000000      5.230000
25%       30.000000      2.000000     45.450000
50%       43.000000      3.000000    203.300000
75%       56.000000      4.000000   1200.320000
max       69.000000      5.000000   5250.000000
```



```
#age count
age_count = df['age'].value_counts().sort_index()
age_count
```

```
age
18    1844
19    1936
20    1844
21    1947
22    2051
23    1897
24    1977
25    1863
26    1896
27    1950
28    1953
29    1885
30    1981
31    1866
32    1891
33    1913
34    1883
35    1841
36    1954
37    2057
38    1954
```



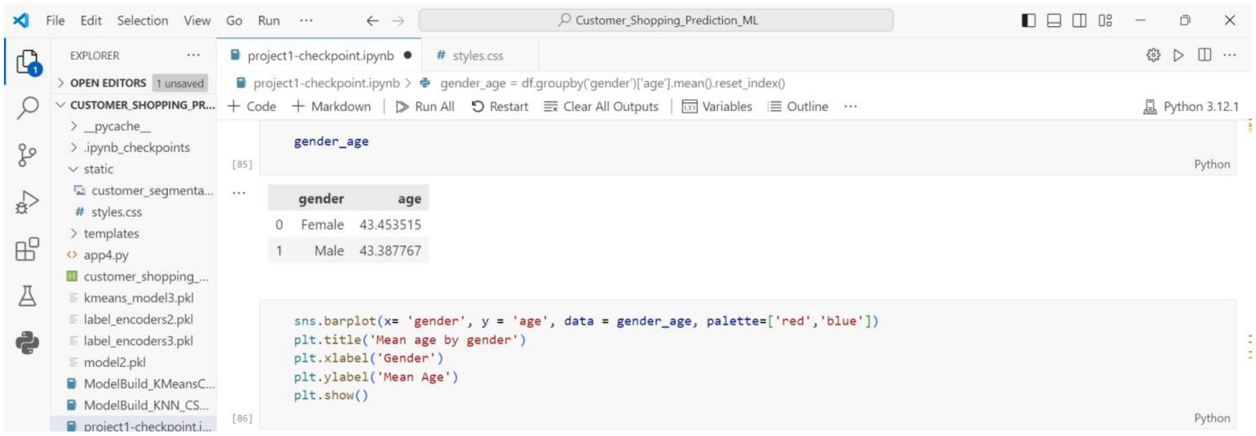
```
max_age = df.nlargest(1,'age')
max_age
```

	gender	age	category	quantity	price	payment_method	shopping_mall
8	Male	69	Clothing	3	900.24	Credit Card	Metrocity

```
min_age = df.nsmallest(1,'age')
min_age
```

	gender	age	category	quantity	price	payment_method	shopping_mall
48	Male	18	Cosmetics	3	121.98	Debit Card	Zorlu Center

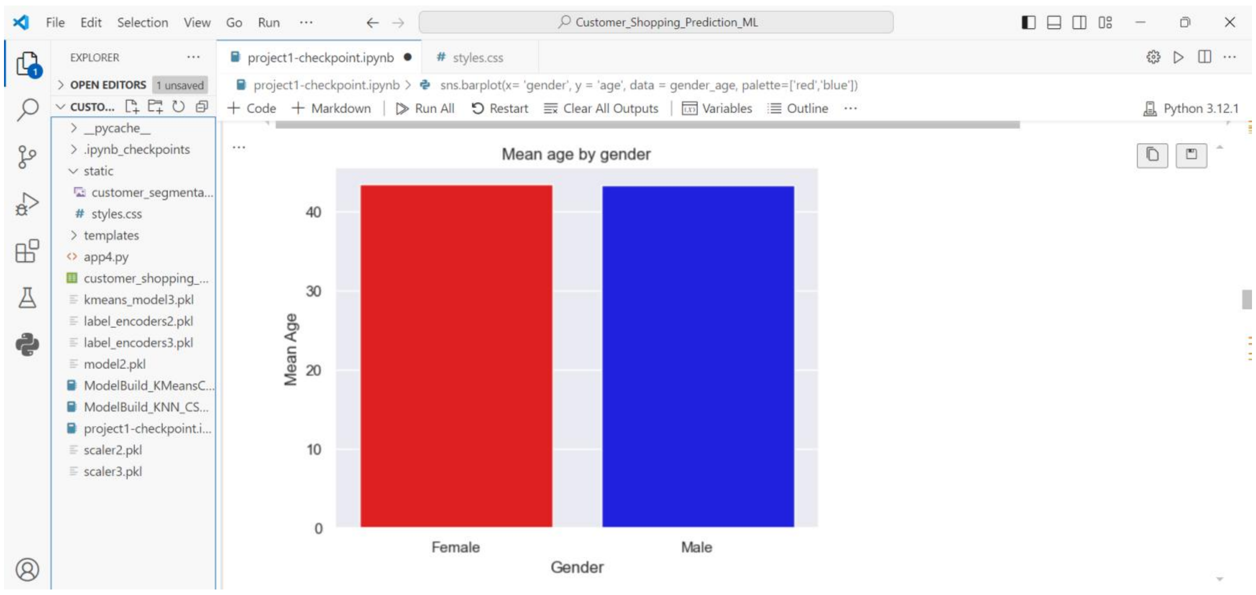
```
gender_age = df.groupby('gender')['age'].mean().reset_index()
```



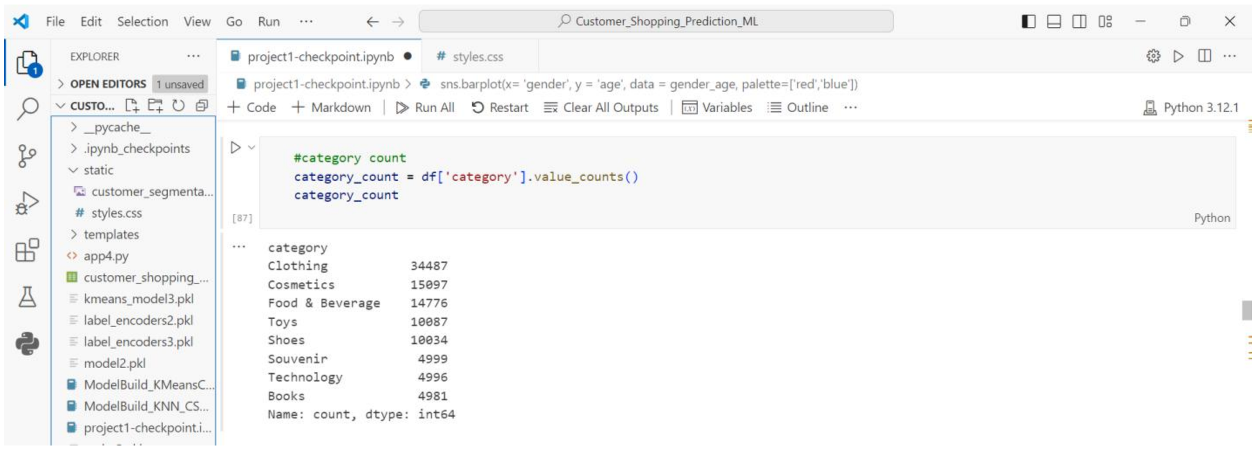
```
gender_age
```

	gender	age
0	Female	43.453515
1	Male	43.387767

```
sns.barplot(x= 'gender', y = 'age', data = gender_age, palette=['red','blue'])
plt.title('Mean age by gender')
plt.xlabel('Gender')
plt.ylabel('Mean Age')
plt.show()
```

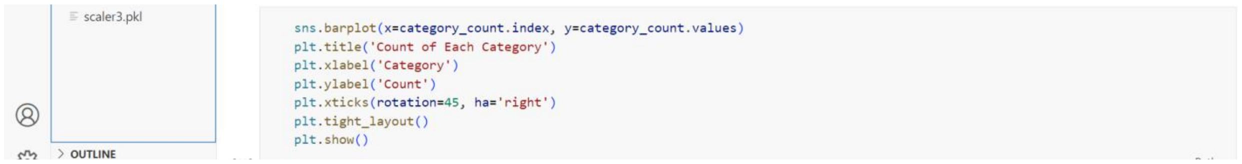


```
sns.barplot(x= 'gender', y = 'age', data = gender_age, palette=['red','blue'])
```

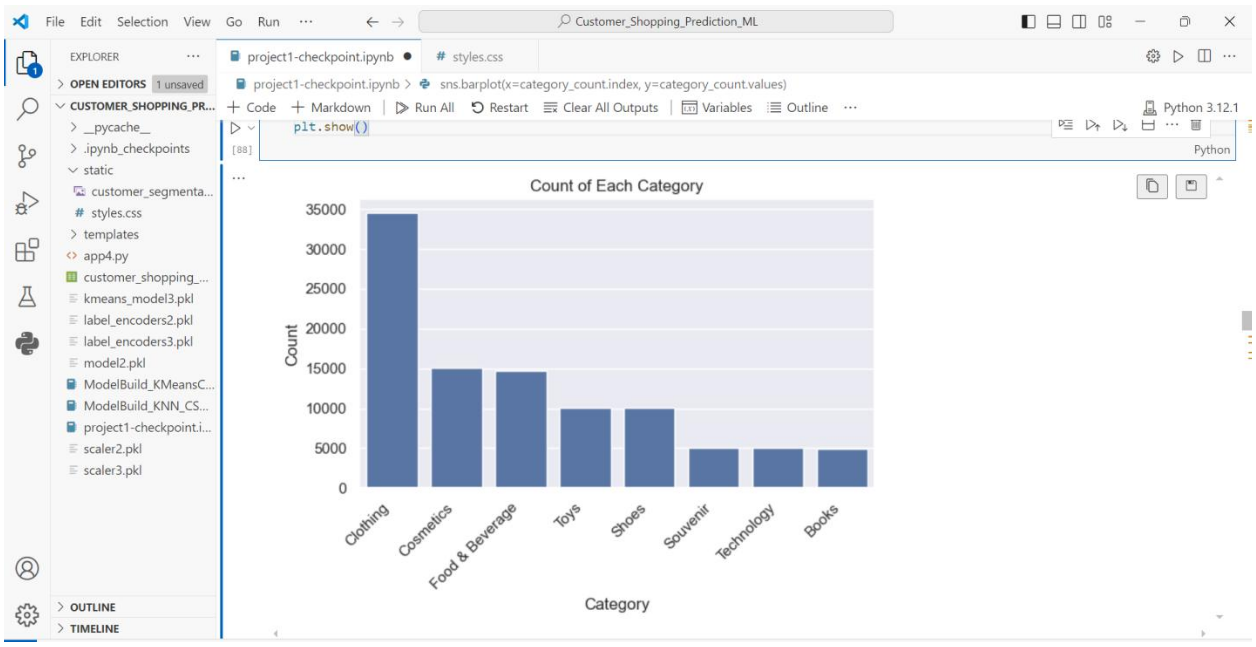


```
#category count
category_count = df['category'].value_counts()
category_count
```

```
category
Clothing          34487
Cosmetics         15097
Food & Beverage   14776
Toys              10087
Shoes             10034
Souvenir           4999
Technology         4996
Books              4981
Name: count, dtype: int64
```

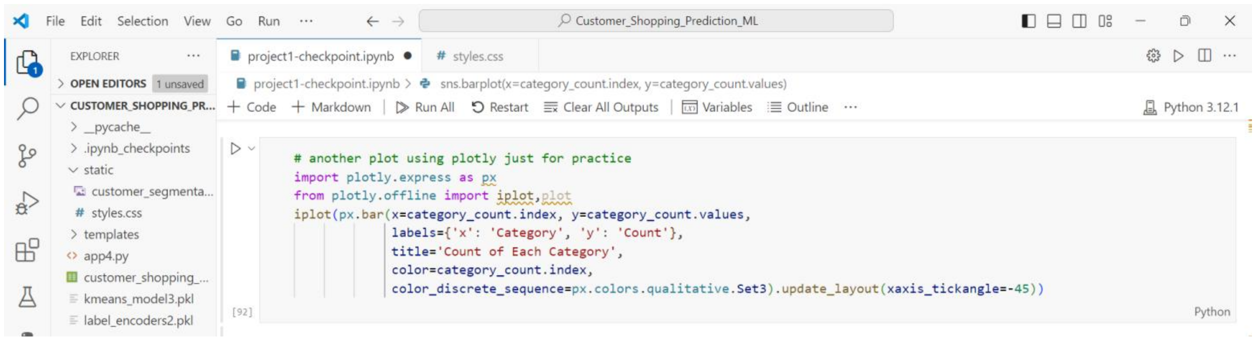


```
sns.barplot(x=category_count.index, y=category_count.values)
plt.title('Count of Each Category')
plt.xlabel('Category')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



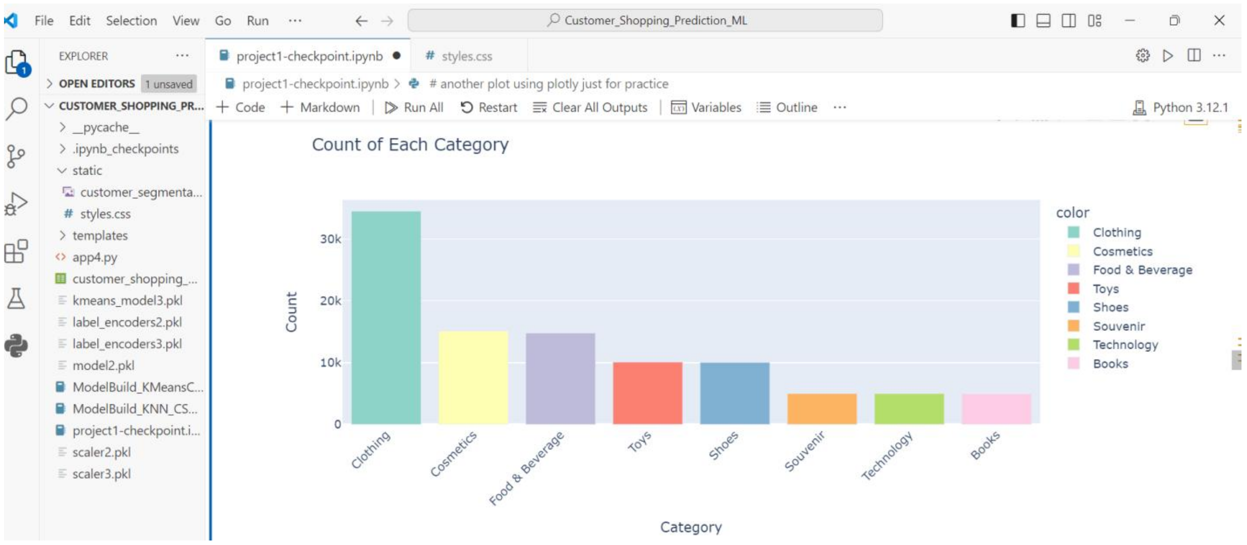
```
plt.show()
```

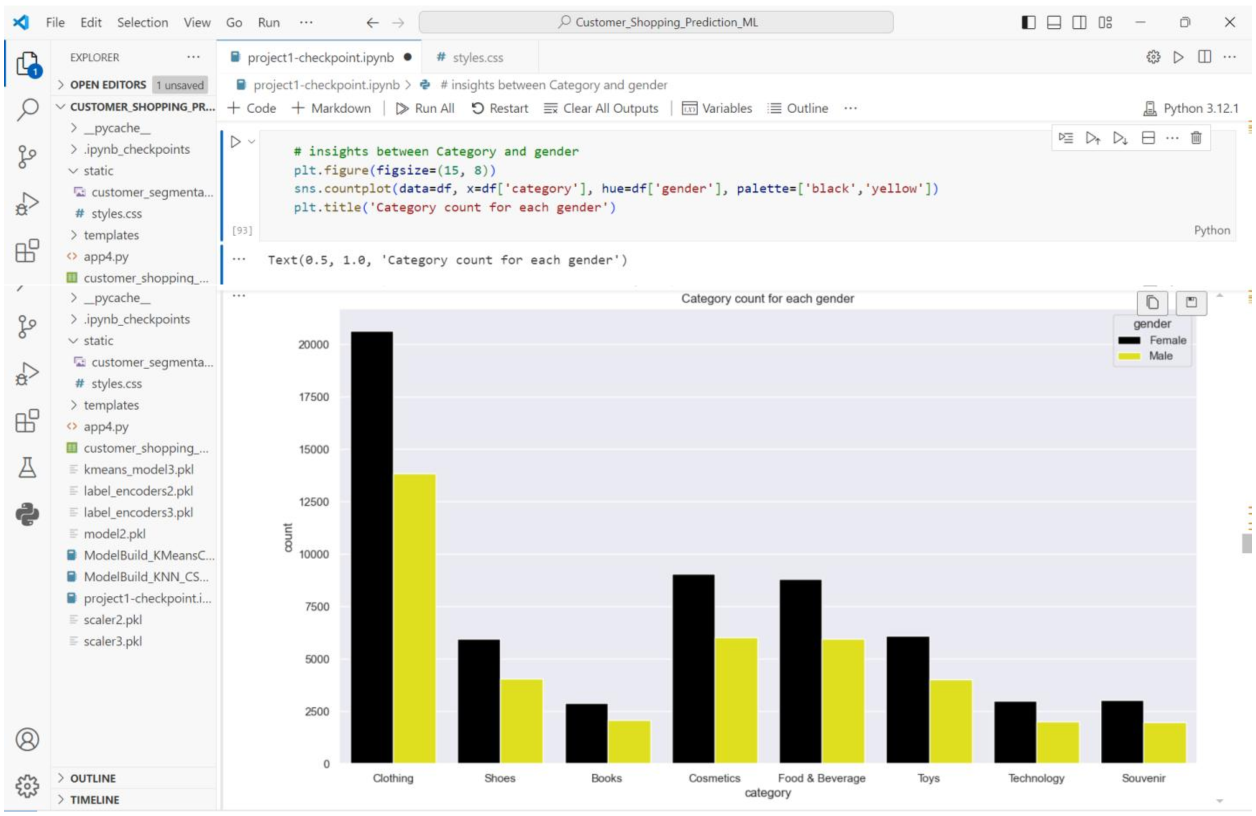
[88]


Customer_Shopping_Prediction_ML
project1-checkpoint.ipynb
styles.css
sns.barplot(x=category_count.index, y=category_count.values)
Python 3.12.1

```
# another plot using plotly just for practice
import plotly.express as px
from plotly.offline import iplot,plot
iplot(px.bar(x=category_count.index, y=category_count.values,
             labels={'x': 'Category', 'y': 'Count'},
             title='Count of Each Category',
             color=category_count.index,
             color_discrete_sequence=px.colors.qualitative.Set3).update_layout(xaxis_tickangle=-45))
```

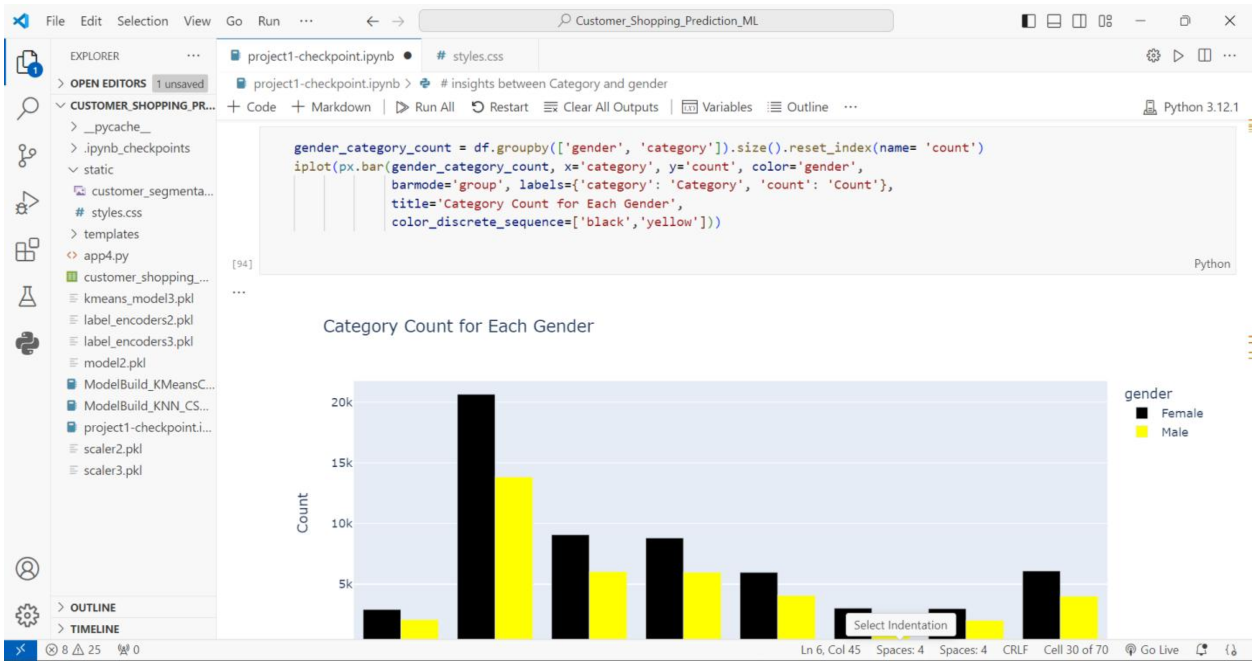
[92]





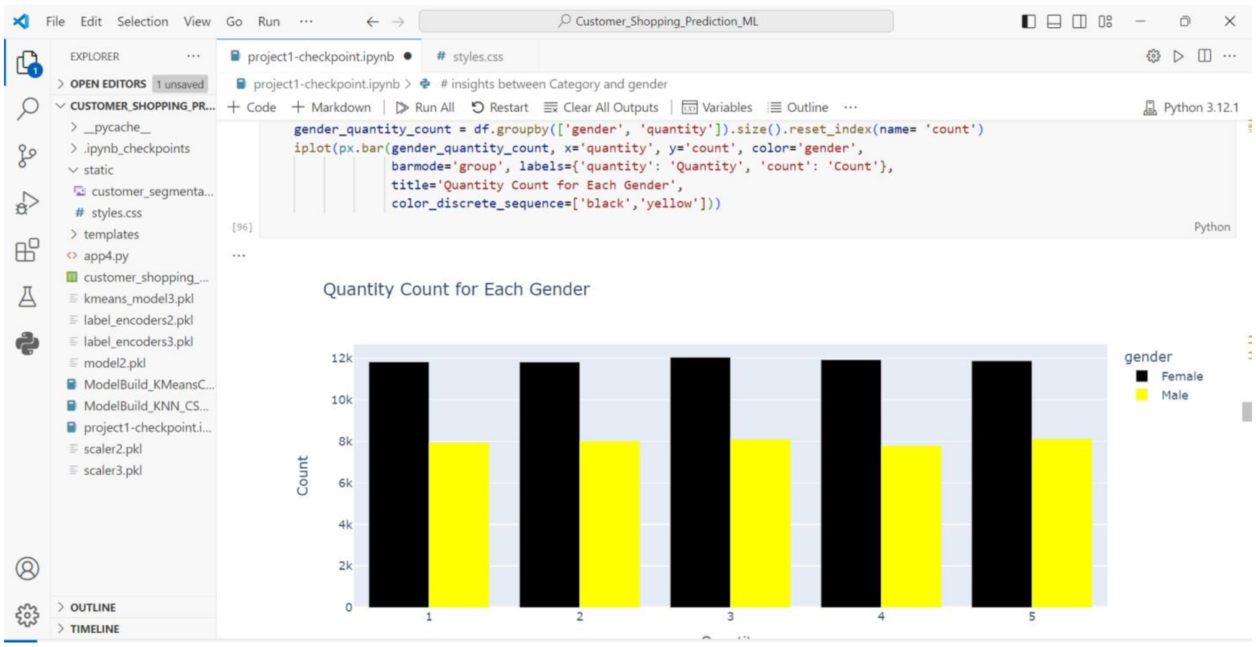
```
# insights between Category and gender
plt.figure(figsize=(15, 8))
sns.countplot(data=df, x=df['category'], hue=df['gender'], palette=['black','yellow'])
plt.title('Category count for each gender')
```

```
Text(0.5, 1.0, 'Category count for each gender')
```

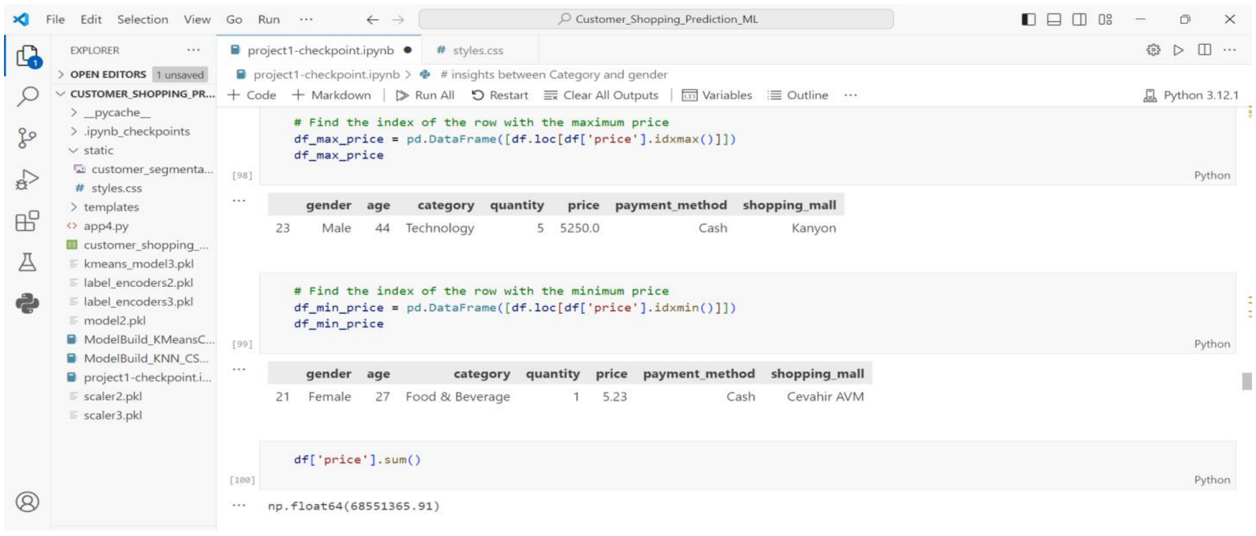


```
gender_category_count = df.groupby(['gender', 'category']).size().reset_index(name= 'count')
iplot(px.bar(gender_category_count, x='category', y='count', color='gender',
             barmode='group', labels={'category': 'Category', 'count': 'Count'},
             title='Category Count for Each Gender',
             color_discrete_sequence=['black','yellow']))
```

File Edit Selection View Go Run
Customer_Shopping_Prediction_ML
EXPLORER
OPEN EDITORS 1 unsaved
CUSTOMER_SHOPPING_PR...
project1-checkpoint.ipynb
styles.css
+ Code + Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.12.1
[94]
Python
Category Count for Each Gender
gender
Female
Male
Count
20k
15k
10k
5k
Select Indentation
Ln 6, Col 45 Spaces: 4 Spaces: 4 CRLF Cell 30 of 70 Go Live



```
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']))
```



```
# Find the index of the row with the maximum price
df_max_price = pd.DataFrame([df.loc[df['price'].idxmax()]])
df_max_price
```

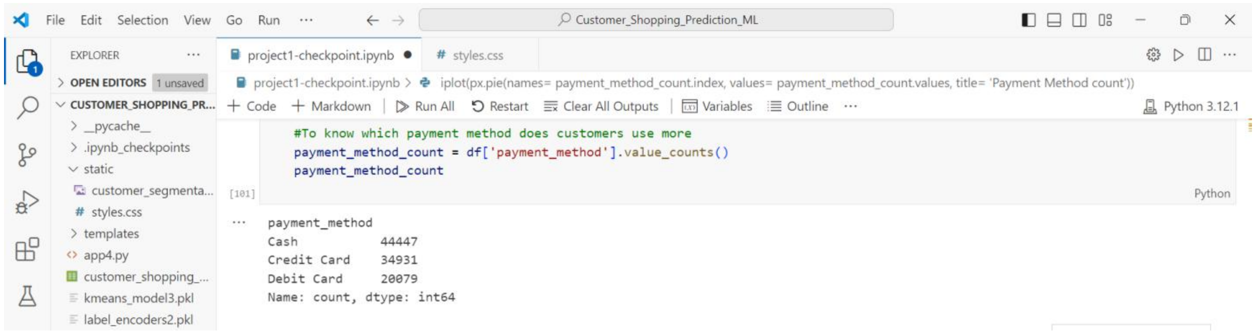
	gender	age	category	quantity	price	payment_method	shopping_mall
23	Male	44	Technology	5	5250.0	Cash	Kanyon

```
# Find the index of the row with the minimum price
df_min_price = pd.DataFrame([df.loc[df['price'].idxmin()]])
df_min_price
```

	gender	age	category	quantity	price	payment_method	shopping_mall
21	Female	27	Food & Beverage	1	5.23	Cash	Cevahir AVM

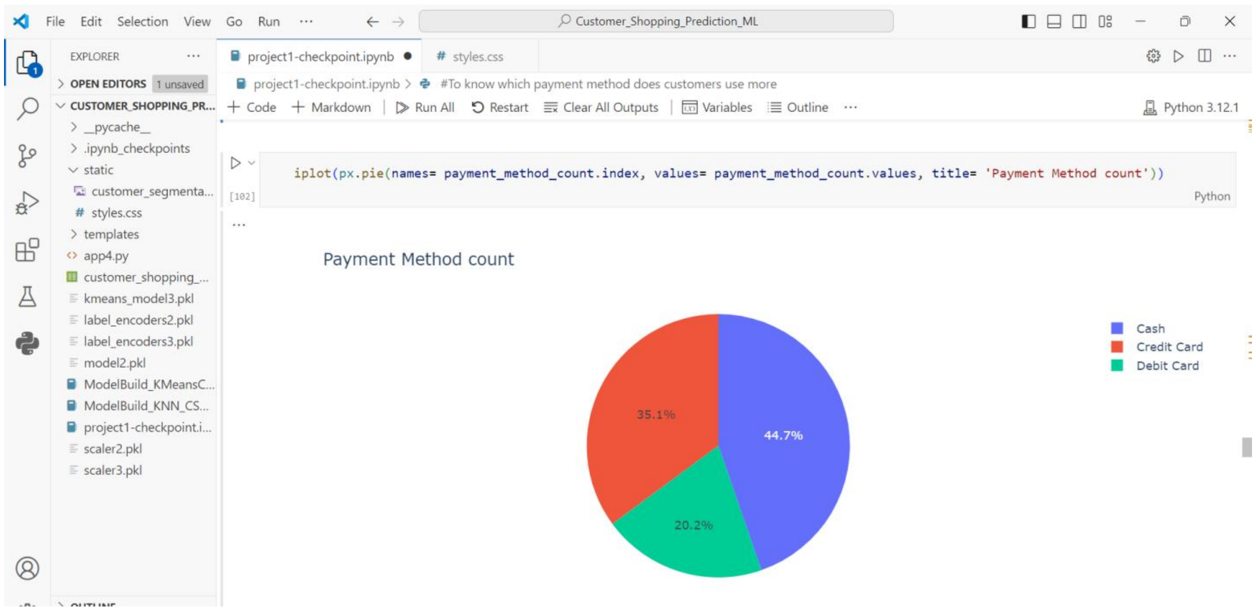
```
df['price'].sum()
```

```
np.float64(68551365.91)
```

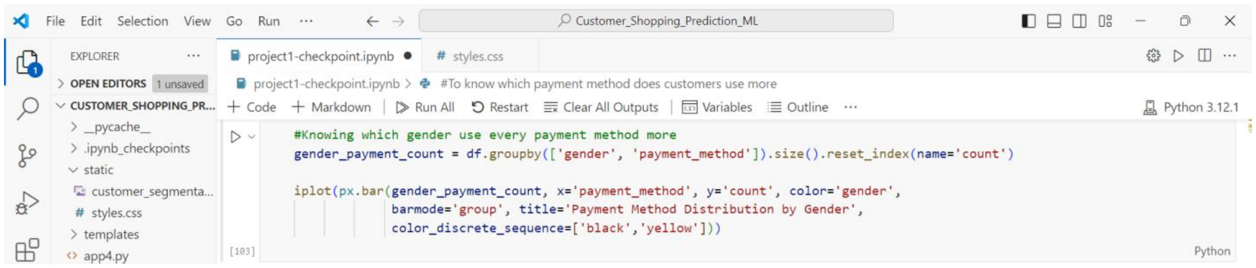


```
#To know which payment method does customers use more
payment_method_count = df['payment_method'].value_counts()
payment_method_count
```

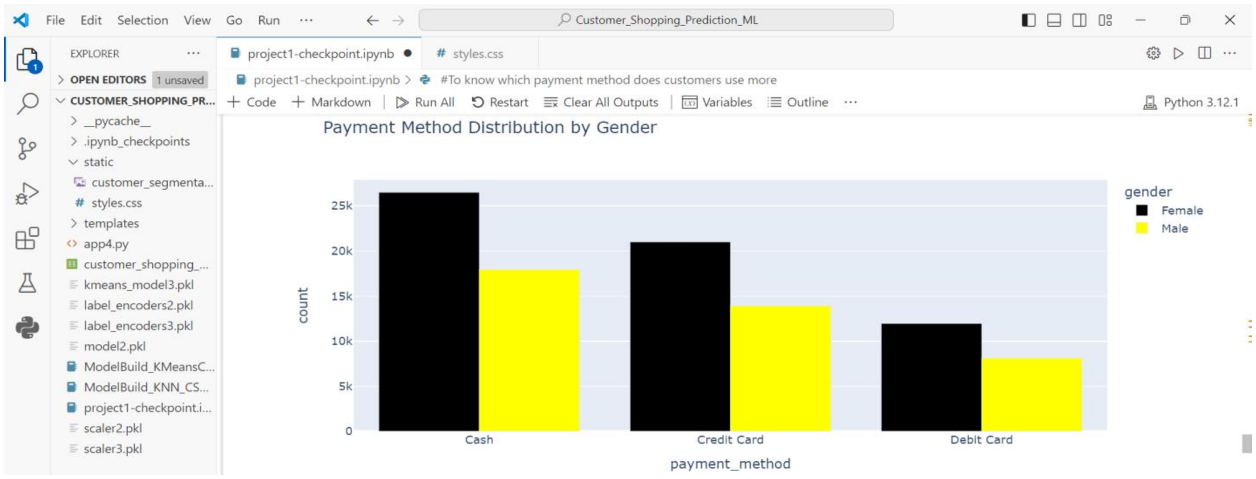
```
payment_method
Cash           44447
Credit Card    34931
Debit Card     20079
Name: count, dtype: int64
```

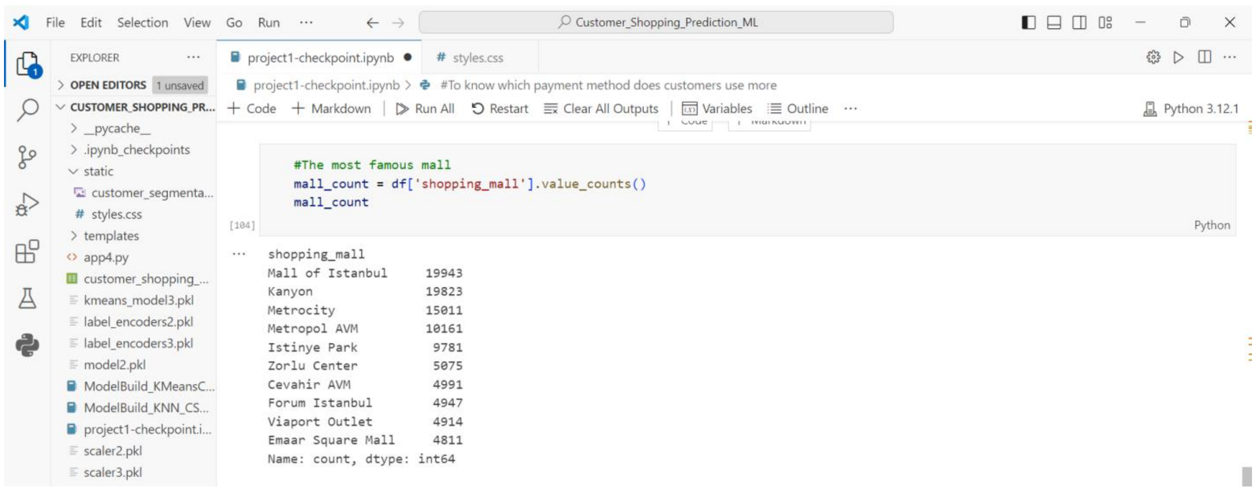


```
iplot(px.pie(names= payment_method_count.index, values= payment_method_count.values, title= 'Payment Method count'))
```


File Edit Selection View Go Run
Customer_Shopping_Prediction_ML
project1-checkpoint.ipynb
styles.css
#To know which payment method does customers use more
Python 3.12.1
[103]
Python

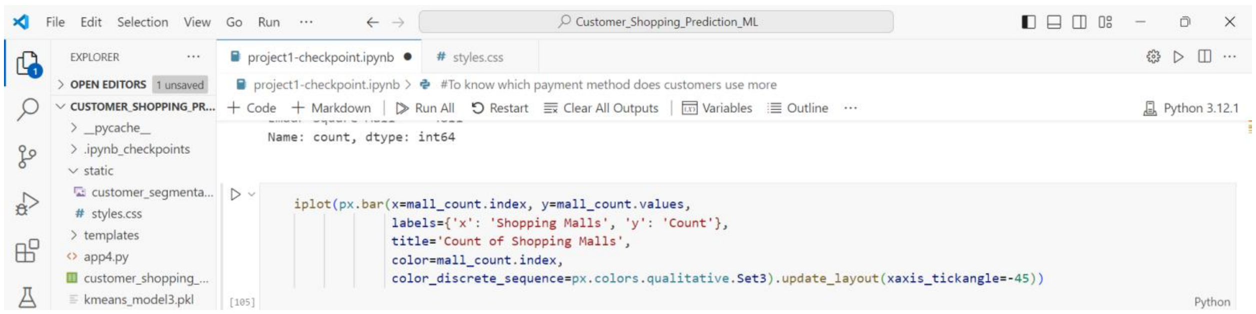
```
#Knowing which gender use every payment method more
gender_payment_count = df.groupby(['gender', 'payment_method']).size().reset_index(name='count')

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']))
```


File Edit Selection View Go Run
Customer_Shopping_Prediction_ML
project1-checkpoint.ipynb
styles.css
#To know which payment method does customers use more
Python 3.12.1
Payment Method Distribution by Gender
gender
Female
Male
count
25k
20k
15k
10k
5k
0
Cash
Credit Card
Debit Card
payment_method



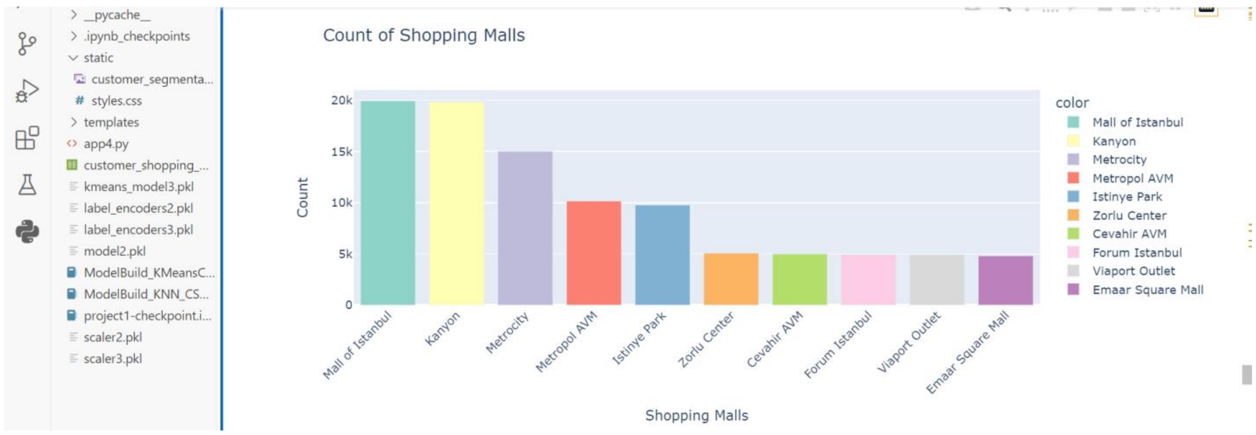
```
#The most famous mall
mall_count = df['shopping_mall'].value_counts()
mall_count
```

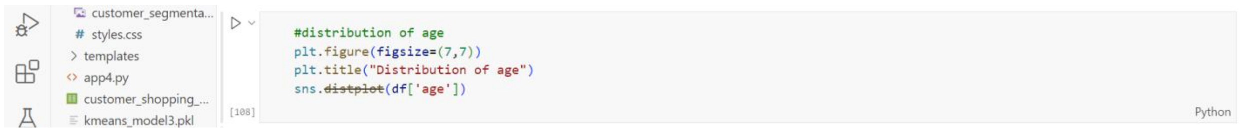
```
shopping_mall
Mall of Istanbul     19943
Kanyon               19823
Metrocity            15011
Metropol AVM         10161
Istinye Park          9781
Zorlu Center          5075
Cevahir AVM           4991
Forum Istanbul        4947
Viaport Outlet        4914
Emaar Square Mall     4811
Name: count, dtype: int64
```



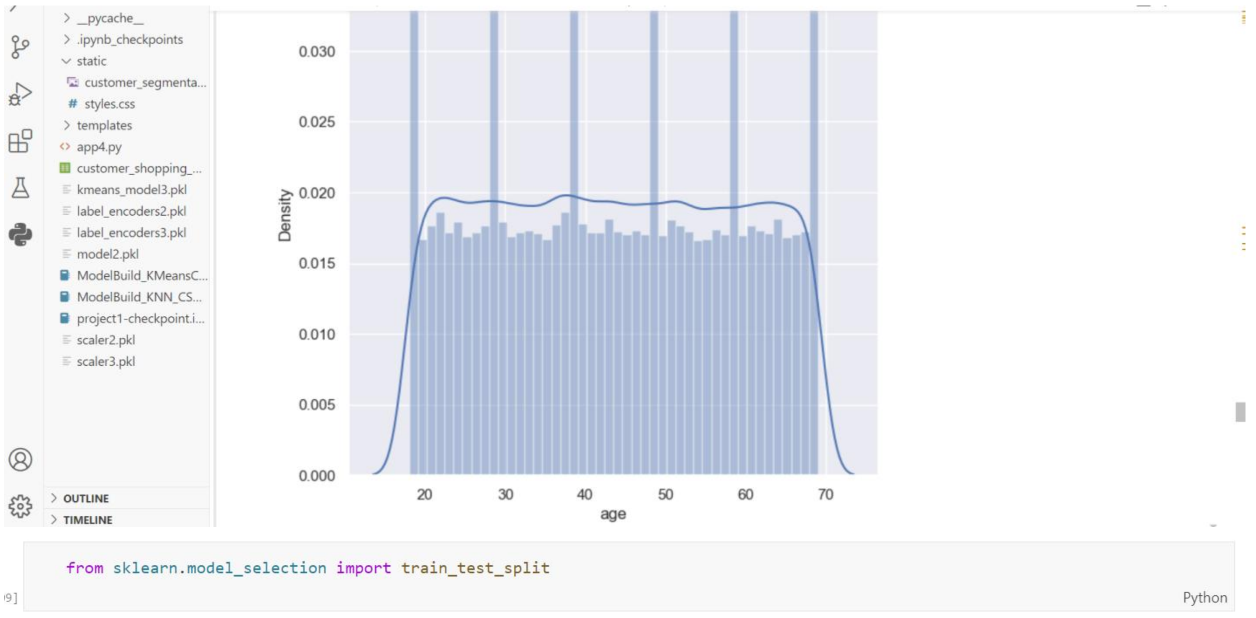
```
Name: count, dtype: int64
```

```
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))
```

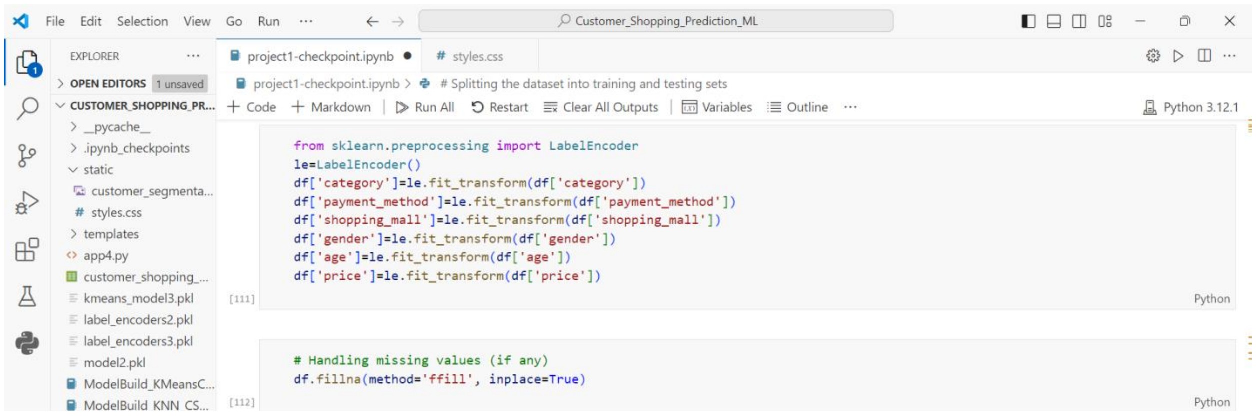




```
#distribution of age
plt.figure(figsize=(7,7))
plt.title("Distribution of age")
sns.~~distplot~~(df['age'])
```



```
from sklearn.model_selection import train_test_split
```



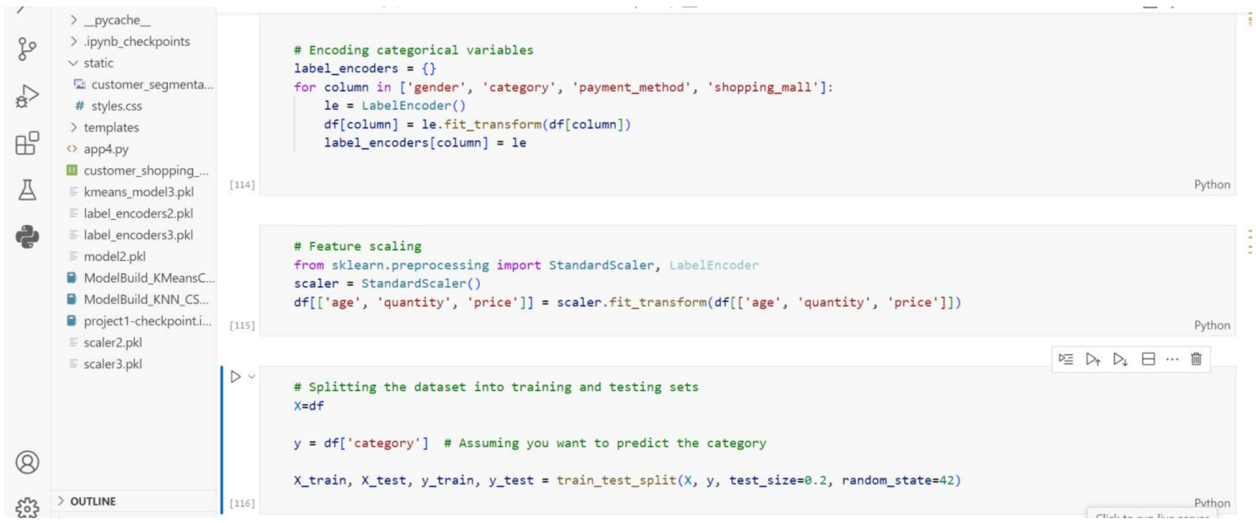
File Edit Selection View Go Run ···
Customer_Shopping_Prediction_ML
project1-checkpoint.ipynb
styles.css
EXPLORER
OPEN EDITORS 1 unsaved
CUSTOMER_SHOPPING_PR...
project1-checkpoint.ipynb > # Splitting the dataset into training and testing sets
+ Code + Markdown Run All Restart Clear All Outputs Variables Outline
Python 3.12.1
Python

```
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'])
```

[111]

```
# Handling missing values (if any)
df.fillna(method='ffill', inplace=True)
```

[112]



```
# Encoding categorical variables
label_encoders = {}
for column in ['gender', 'category', 'payment_method', 'shopping_mall']:
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column])
    label_encoders[column] = le
```

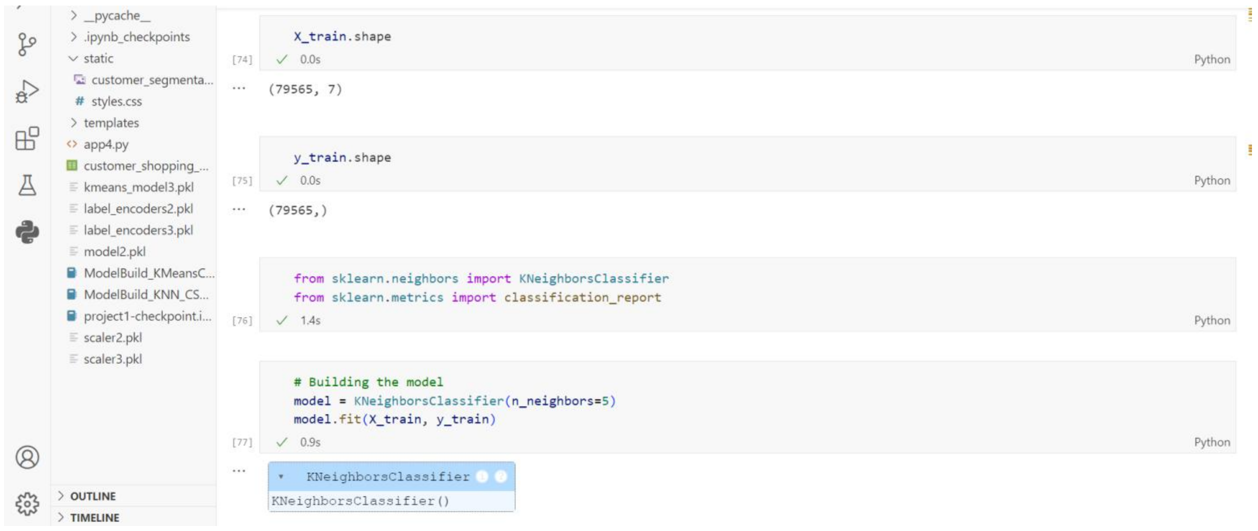
[114]

```
# Feature scaling
from sklearn.preprocessing import StandardScaler, LabelEncoder
scaler = StandardScaler()
df[['age', 'quantity', 'price']] = scaler.fit_transform(df[['age', 'quantity', 'price']])
```

[115]

```
# Splitting the dataset into training and testing sets
X=df

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)
```

[116]



```
X_train.shape
```
[74] ✓ 0.0s Python

```
(79565, 7)
```

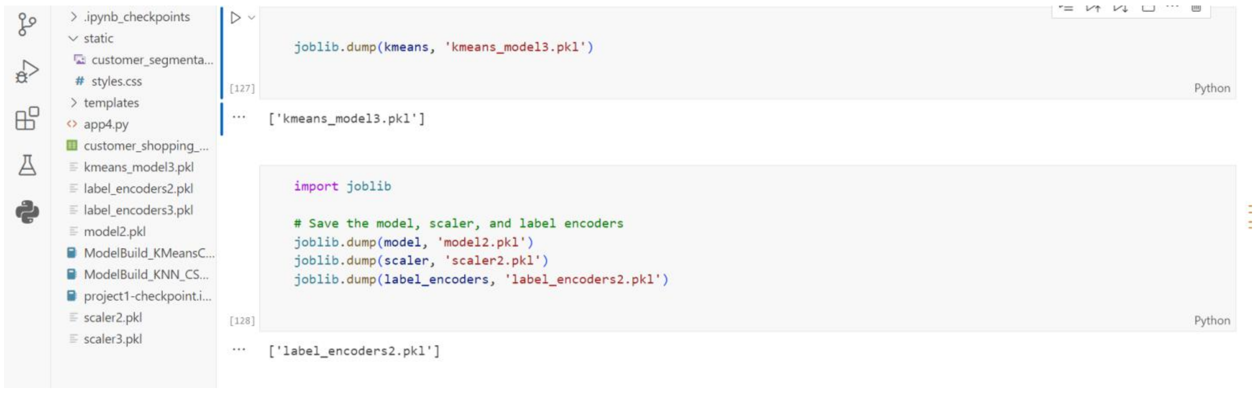
```
y_train.shape
```
[75] ✓ 0.0s Python

```
(79565,)
```

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
```
[76] ✓ 1.4s Python

```
# Building the model
model = KNeighborsClassifier(n_neighbors=5)
model.fit(X_train, y_train)
```
[77] ✓ 0.9s Python

```
KNeighborsClassifier
KNeighborsClassifier()
```

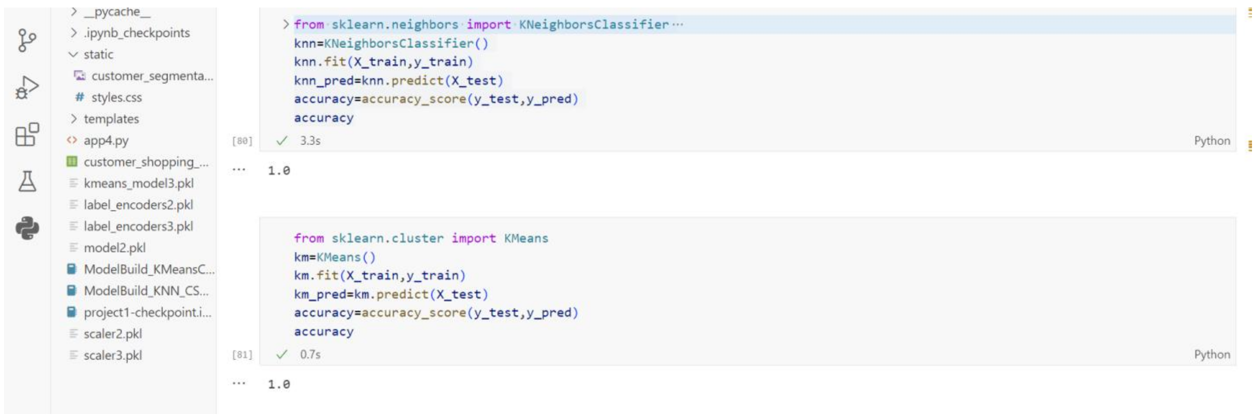


```
joblib.dump(kmeans, 'kmeans_model3.pkl')
```
[127] Python

```
['kmeans_model3.pkl']
```

```
import joblib

# Save the model, scaler, and label encoders
joblib.dump(model, 'model2.pkl')
joblib.dump(scaler, 'scaler2.pkl')
joblib.dump(label_encoders, 'label_encoders2.pkl')
```
[128] Python

```
['label_encoders2.pkl']
```



```
> from sklearn.neighbors import KNeighborsClassifier ...
knn=KNeighborsClassifier()
knn.fit(X_train,y_train)
knn_pred=knn.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
accuracy
```
[80] ✓ 3.3s Python

```
1.0
```

```
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
```
[81] ✓ 0.7s Python

```
1.0
```