ADV EXPERIMENT 1

Name : Adwait Purao
UID : 2021300101
Batch : BE COMPS D

Source Url : https://www.kaggle.com/datasets/utkarsharya/ecommerce-purchases/data

Description: This dataset is a collection of e-commerce transactions, where each row corresponds to an individual purchase made by a customer. The dataset captures a wide array of information about the transaction, the customer, and the product purchased. Below is a detailed description of the dataset, including the columns, potential tables if normalized into a relational database, and insights into the kind of information each column might represent.

Dataset Overview

- **Table Name**: `E-commerce Purchases`
- **Number of Rows**: Each row represents one transaction or purchase made by a customer.
- **Number of Columns**: 14 columns, each providing different details related to the transaction, customer, and product.

Column Descriptions

1. **`Address`**:
    - **Data Type**: String
    - **Description**: The physical address of the customer. This may include street address, city, state, and postal code.
    - **Example**: `"16629 Pace Camp Apt. 448\nAlexisborough, NE 77130-7478"`
2. **`Lot`**:
    - **Data Type**: String
    - **Description**: Possibly represents a unique identifier or characteristic associated with the transaction, product, or shipment. The meaning of "Lot" here is unclear without further context, but it may refer to a batch number or location identifier.
    - **Example**: `"46 in"`
3. **`AM or PM`**:
    - **Data Type**: String
    - **Description**: Indicates whether the transaction occurred in the AM or PM.
    - **Example**: `"PM"`
4. **`Browser Info`**:

- **Data Type**: String
- **Description**: The user agent string of the browser used to make the purchase. This can include information about the browser version, operating system, and device.
- **Example**: `"Opera/9.56.(X11; Linux x86_64; sl-SI) Presto/2.9.183 Version/12.00"`

5. **`Company`**:
   - **Data Type**: String
   - **Description**: The name of the company or organization associated with the transaction. This might be the customer's employer, a supplier, or another relevant entity.
   - **Example**: `"Martinez-Herman"`
6. **`Credit Card`**:
   - **Data Type**: Integer/String
   - **Description**: The credit card number used for the transaction. This is a sensitive piece of information, usually anonymized or masked in datasets.
   - **Example**: `"6011929061123406"`
7. **`CC Exp Date`**:
   - **Data Type**: String (Date)
   - **Description**: The expiration date of the credit card used, typically in the format MM/YY.
   - **Example**: `"02/20"`
8. **`CC Security Code`**:
   - **Data Type**: Integer
   - **Description**: The security code (usually 3-4 digits) associated with the credit card. This is also sensitive information.
   - **Example**: `"900"`
9. **`CC Provider`**:
   - **Data Type**: String
   - **Description**: The financial institution or card network that issued the credit card (e.g., Visa, Mastercard, JCB).
   - **Example**: `"JCB 16 digit"`
10. **`Email`**:
    - **Data Type**: String
    - **Description**: The email address of the customer.
    - **Example**: `"pdunlap@yahoo.com"`
11. **`Job`**:
    - **Data Type**: String
    - **Description**: The job title or profession of the customer. This provides insight into the demographics of the customer base.
    - **Example**: `"Scientist, product/process development"`
12. **`IP Address`**:

- **Data Type**: String
- **Description**: The IP address from which the transaction was made. This can provide information on the location of the customer or the device used.
- **Example**: `"149.146.147.205"`

13. **`Language`**:
    - **Data Type**: String
    - **Description**: The preferred language of the customer, which may influence how they interact with the e-commerce platform.
    - **Example**: `"el"`
14. **`Purchase Price`**:
    - **Data Type**: Float
    - **Description**: The total amount of money spent on the transaction. This is the final amount charged to the customer.
    - **Example**: `"98.14"`

Data Model Diagram

Star Schema Design

Fact Table: `Transactions`

The fact table will store the measurable, quantitative data about the transactions. This table will also contain foreign keys that link to the dimension tables.

Column Name	Data Type	Description
`TransactionID`	Integer	Unique identifier for each transaction (Primary Key)
`CustomerID`	Integer	Foreign key to the `Customers` dimension table
`ProductID`	Integer	Foreign key to the `Products` dimension table
`CreditCardID`	Integer	Foreign key to the `CreditCards` dimension table
`DateTimeID`	Integer	Foreign key to the `DateTime` dimension table
`PurchasePrice`	Float	The amount spent on the transaction

Dimension Table 1: `Customers`

This table will store information about the customers who made the purchases.

Column Name	Data Type	Description
CustomerID	Integer	Unique identifier for each customer (Primary Key)
Address	String	Physical address of the customer
Email	String	Email address of the customer
Job	String	Job title or profession of the customer
IP Address	String	IP address from which the purchase was made
Language	String	Preferred language of the customer

Dimension Table 2: `Products`

This table will store information about the products or services being purchased.

Column Name	Data Type	Description
ProductID	Integer	Unique identifier for each product (Primary Key)
Lot	String	Identifier for the lot or batch of the product
Company	String	Company associated with the product
Browser Info	String	Browser used to make the purchase

Dimension Table 3: `CreditCards`

This table will store information about the credit cards used for transactions.

Column Name	Data Type	Description
CreditCardID	Integer	Unique identifier for each credit card (Primary Key)
Credit Card	String	Credit card number (potentially anonymized)
CC Exp Date	String	Expiration date of the credit card

`CC Security Code`	Integer	Security code of the credit card
`CC Provider`	String	Provider of the credit card (e.g., Visa, Mastercard)

Dimension Table 4: `DateTime`

This table will store information about the date and time of the transaction.

Column Name	Data Type	Description
`DateTimeID`	Integer	Unique identifier for each date-time entry (Primary Key)
`AM or PM`	String	Indicates whether the transaction was in the AM or PM
`Hour`	Integer	Hour of the transaction (derived from `Lot`)

Schema Diagram

- **Fact Table: `Transactions`**
    1. `TransactionID` (PK)
    2. `CustomerID` (FK)
    3. `ProductID` (FK)
    4. `CreditCardID` (FK)
    5. `DateTimeID` (FK)
    6. `PurchasePrice`
- **Dimension Tables**:
    1. **`Customers`**: `CustomerID` (PK), `Address`, `Email`, `Job`, `IP Address`, `Language`
    2. **`Products`**: `ProductID` (PK), `Lot`, `Company`, `Browser Info`
    3. **`CreditCards`**: `CreditCardID` (PK), `Credit Card`, `CC Exp Date`, `CC Security Code`, `CC Provider`
    4. **`DateTime`**: `DateTimeID` (PK), `AM or PM`, `Hour`

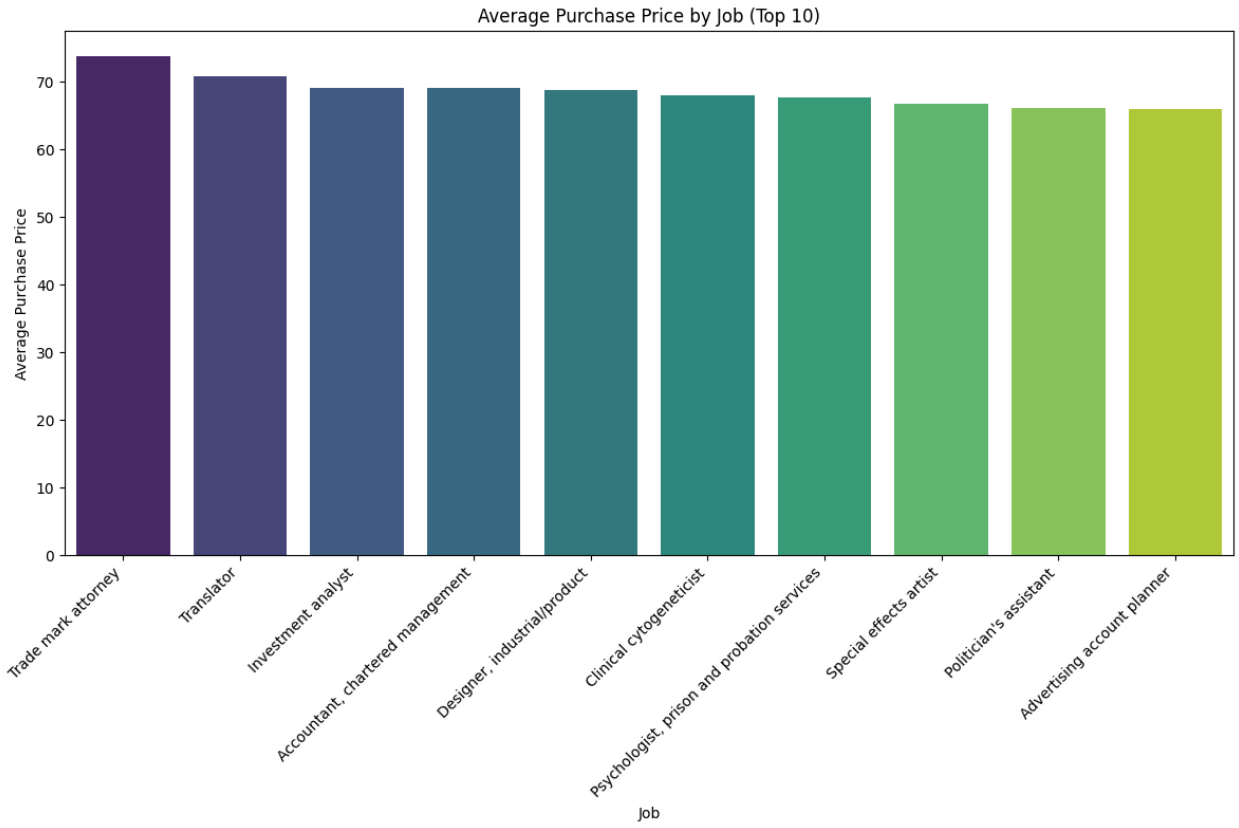
Visualizations:

1. Bar Chart - Average Purchase Price by Job

- **Description: This bar chart displays the average purchase price associated with each job title in the dataset. The x-axis represents different job titles, and the y-axis represents the average purchase price.**
- **Observation: The top 10 highest average purchase prices show a slightly decreasing trend.**
- **Questions Answered:**
    - **Which job titles are associated with higher purchase prices?**
    - **Are there specific professions that tend to make larger purchases?**

```
# 1. Bar Chart - Average Purchase Price by Job



import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns



# Load your DataFrame (assuming it's already done)



# 1. Bar Chart - Average Purchase Price by Job

job_sales = df.groupby('Job')['Purchase
Price'].mean().sort_values(ascending=False)



# Select the top N (e.g., top 10) jobs for better readability

top_n = 10

top_job_sales = job_sales.head(top_n)



plt.figure(figsize=(12, 8))

sns.barplot(x=top_job_sales.index, y=top_job_sales.values,
palette='viridis')
```

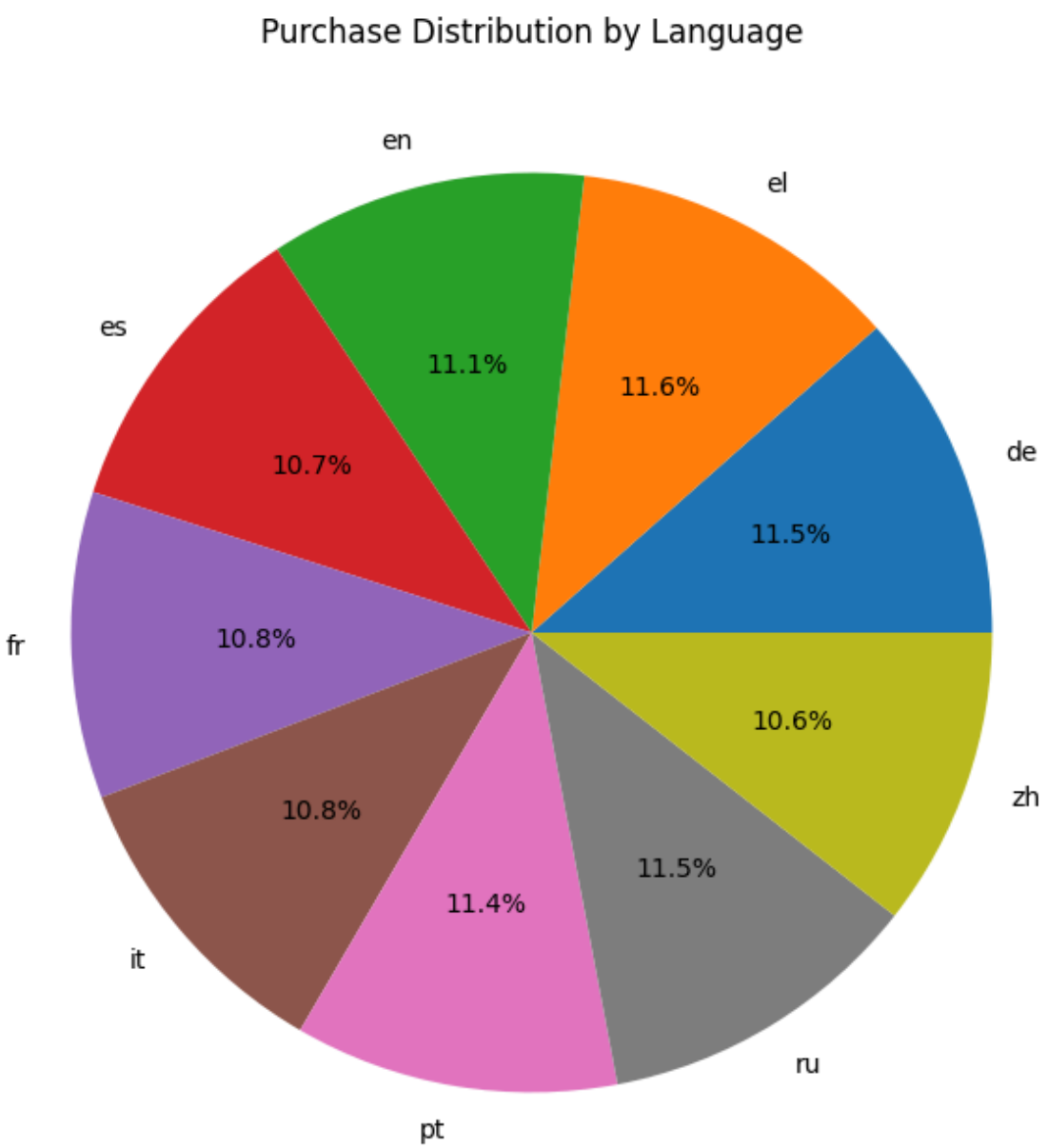
```
# Rotate labels for better readability
plt.xticks(rotation=45, ha='right', fontsize=10)


plt.title('Average Purchase Price by Job (Top 10)')
plt.xlabel('Job')
plt.ylabel('Average Purchase Price')
plt.tight_layout()


plt.show()
```



2. Pie Chart - Purchase Distribution by Language

- **Description: This pie chart shows the distribution of purchases made by customers based on their preferred language. Each slice of the pie represents a different language, with the size of the slice indicating the proportion of total purchases.**
- **Observation: The chart reveals that the customers are almost evenly distributed over all languages.**
- **Questions Answered:**
    - **What is the distribution of purchases among different language groups?**
    - **Is there a dominant language among customers that could guide marketing efforts?**

```
# 2. Pie Chart - Purchase Distribution by Language

language_sales = df.groupby('Language')['Purchase Price'].sum()

plt.figure(figsize=(10, 8))

plt.pie(language_sales, labels=language_sales.index, autopct='%1.1f%%')

plt.title('Purchase Distribution by Language')

plt.show()
```



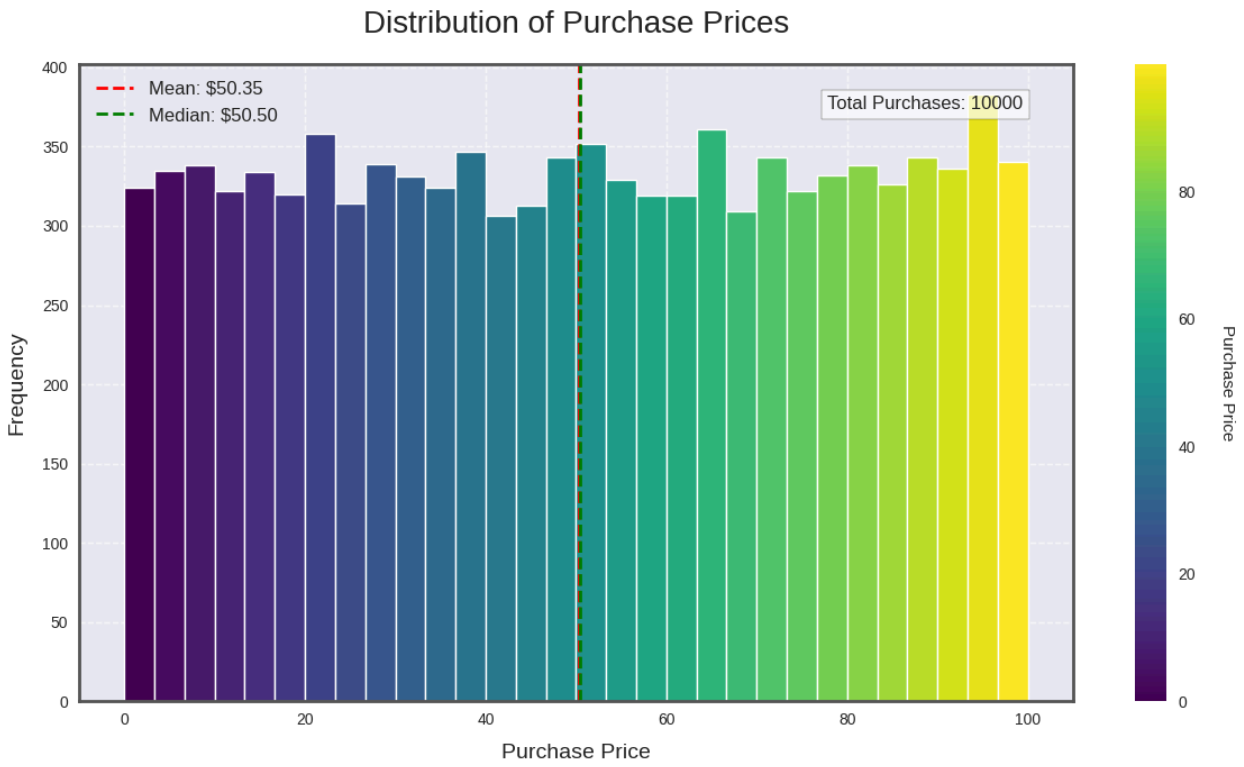
3. Histogram - Purchase Price Distribution

- **Description: The histogram illustrates the distribution of purchase prices across all transactions in the dataset. The x-axis represents the purchase price intervals, and the y-axis shows the number of transactions within each interval.**
- **Observation: Purchase prices are distributed over all ranges from 0 to 100 with some variation between them.**
- **Questions Answered:**
    - **What is the general distribution of purchase prices?**
    - **Are there any significant outliers or common price points in the data?**

```
# 3. Histogram - Purchase Price Distribution

# Set the style for a more modern look

plt.style.use('seaborn')



# Create the figure and axis objects

fig, ax = plt.subplots(figsize=(12, 7))



# Create the histogram

n, bins, patches = ax.hist(df['Purchase Price'], bins=30,
edgecolor='white', linewidth=1)



# Color gradient

cm = plt.cm.get_cmap('viridis')

bin_centers = 0.5 * (bins[:-1] + bins[1:])

col = bin_centers - min(bin_centers)

col /= max(col)



for c, p in zip(col, patches):

    plt.setp(p, 'facecolor', cm(c))



# Add a color bar

sm = plt.cm.ScalarMappable(cmap=cm,
norm=plt.Normalize(vmin=min(df['Purchase Price']), vmax=max(df['Purchase
Price'])))
```

```
sm.set_array([])

cbar = plt.colorbar(sm)

cbar.set_label('Purchase Price', rotation=270, labelpad=25)



# Customize the plot

ax.set_title('Distribution of Purchase Prices', fontsize=20, pad=20)

ax.set_xlabel('Purchase Price', fontsize=14, labelpad=10)

ax.set_ylabel('Frequency', fontsize=14, labelpad=10)



# Add grid lines

ax.grid(True, linestyle='--', alpha=0.7)



# Add a subtle box around the plot

for spine in ax.spines.values():

    spine.set_edgecolor('#555555')

    spine.set_linewidth(2)



# Add mean and median lines

mean_price = df['Purchase Price'].mean()

median_price = df['Purchase Price'].median()

ax.axvline(mean_price, color='red', linestyle='dashed', linewidth=2,
label=f'Mean: ${mean_price:.2f}')

ax.axvline(median_price, color='green', linestyle='dashed', linewidth=2,
label=f'Median: ${median_price:.2f}')
```

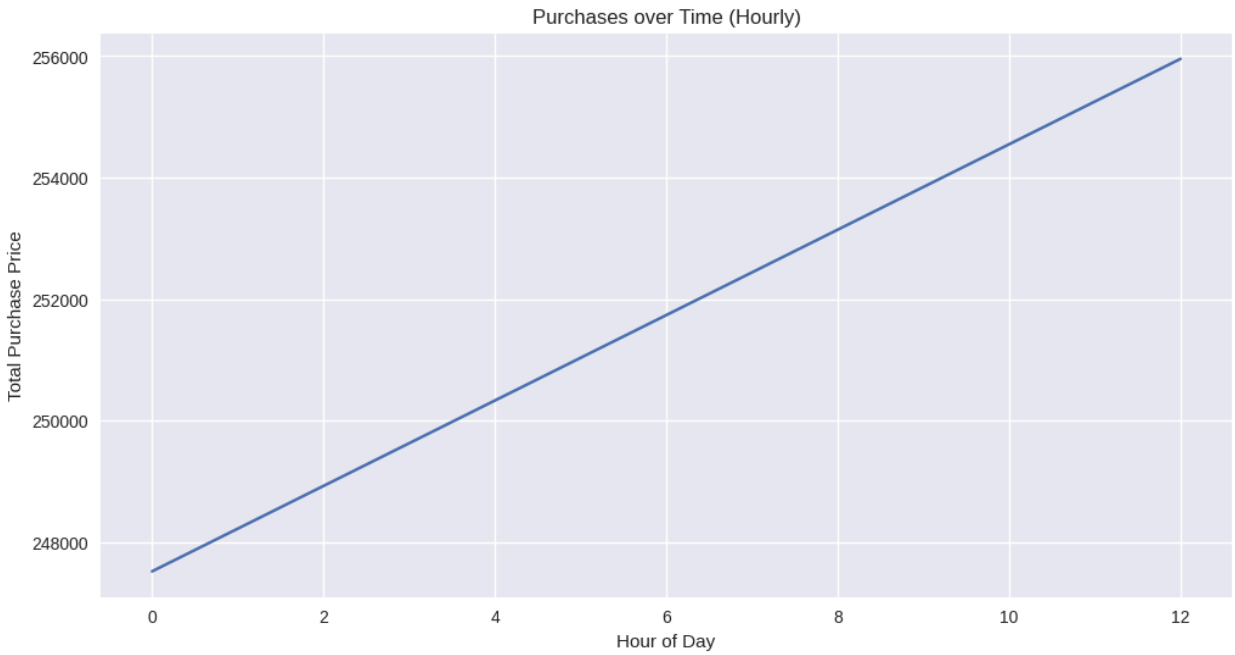
```
# Add legend
ax.legend(fontsize=12)


# Add text annotations
ax.text(0.95, 0.95, f'Total Purchases: {len(df)}',
        verticalalignment='top', horizontalalignment='right',
        transform=ax.transAxes, fontsize=12, bbox=dict(facecolor='white',
alpha=0.7))


plt.tight_layout()
plt.show()
```



4. Timeline Chart - Purchases over Time (Hourly)

- **Description: This timeline chart shows the number of purchases made during different hours of the day. The x-axis represents the hours, and the y-axis shows the number of purchases.**
- **Observation: The total purchase price goes on increasing linearly with each hour of the day.**
- **Questions Answered:**
    - **What are the peak hours for customer purchases?**
    - **Is there a pattern in purchase activity throughout the day?**

```
# 4. Timeline Chart - Purchases over Time (Hourly)

hourly_sales = df.groupby('Hour')['Purchase Price'].sum()

plt.figure(figsize=(12, 6))

hourly_sales.plot()

plt.title('Purchases over Time (Hourly)')

plt.xlabel('Hour of Day')

plt.ylabel('Total Purchase Price')

plt.show()
```

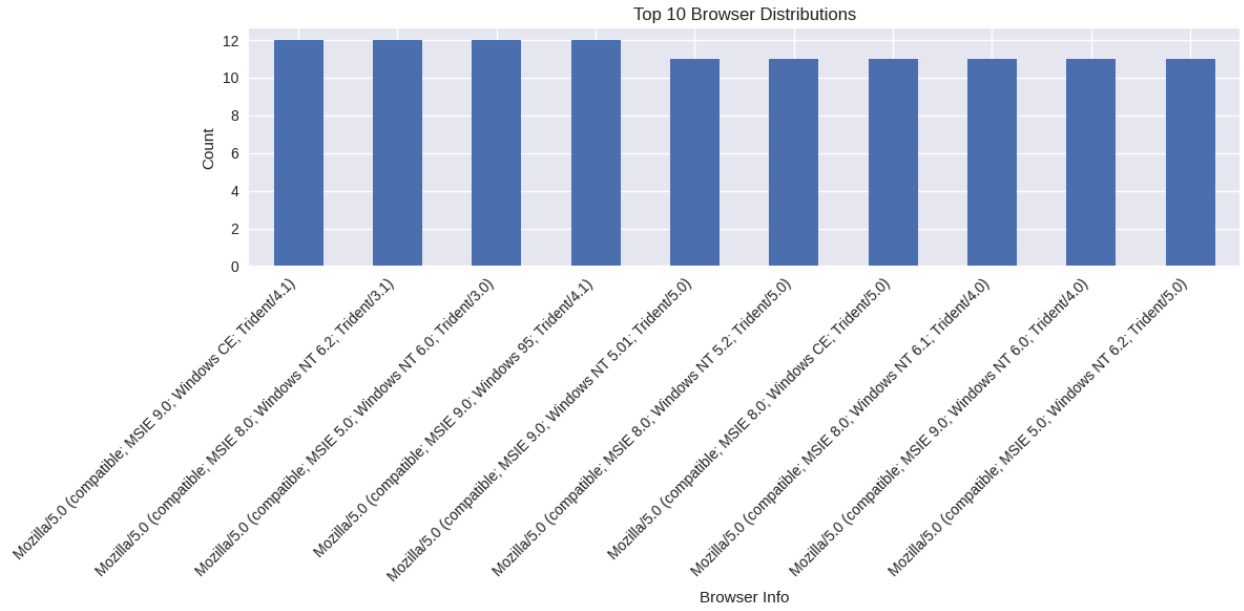


5. Bar Chart: Browser Info Distribution

- **Description: This bar chart shows the distribution of purchases based on the browser information used by customers. The x-axis represents different browsers, and the y-axis shows the number of purchases made using each browser.**
- **Observation: Firefox dominates the plot , with the type of machine changing for each browser.**
- **Questions Answered:**
    - **Which browsers are most commonly used by customers?**
    - **Does browser preference impact purchase behavior?**

```
# 5. Bar Chart: Browser Info Distribution

plt.figure(figsize=(12, 6))

df['Browser Info'].value_counts().head(10).plot(kind='bar')

plt.title('Top 10 Browser Distributions')

plt.xlabel('Browser Info')

plt.ylabel('Count')

plt.xticks(rotation=45, ha='right')

plt.tight_layout()
```

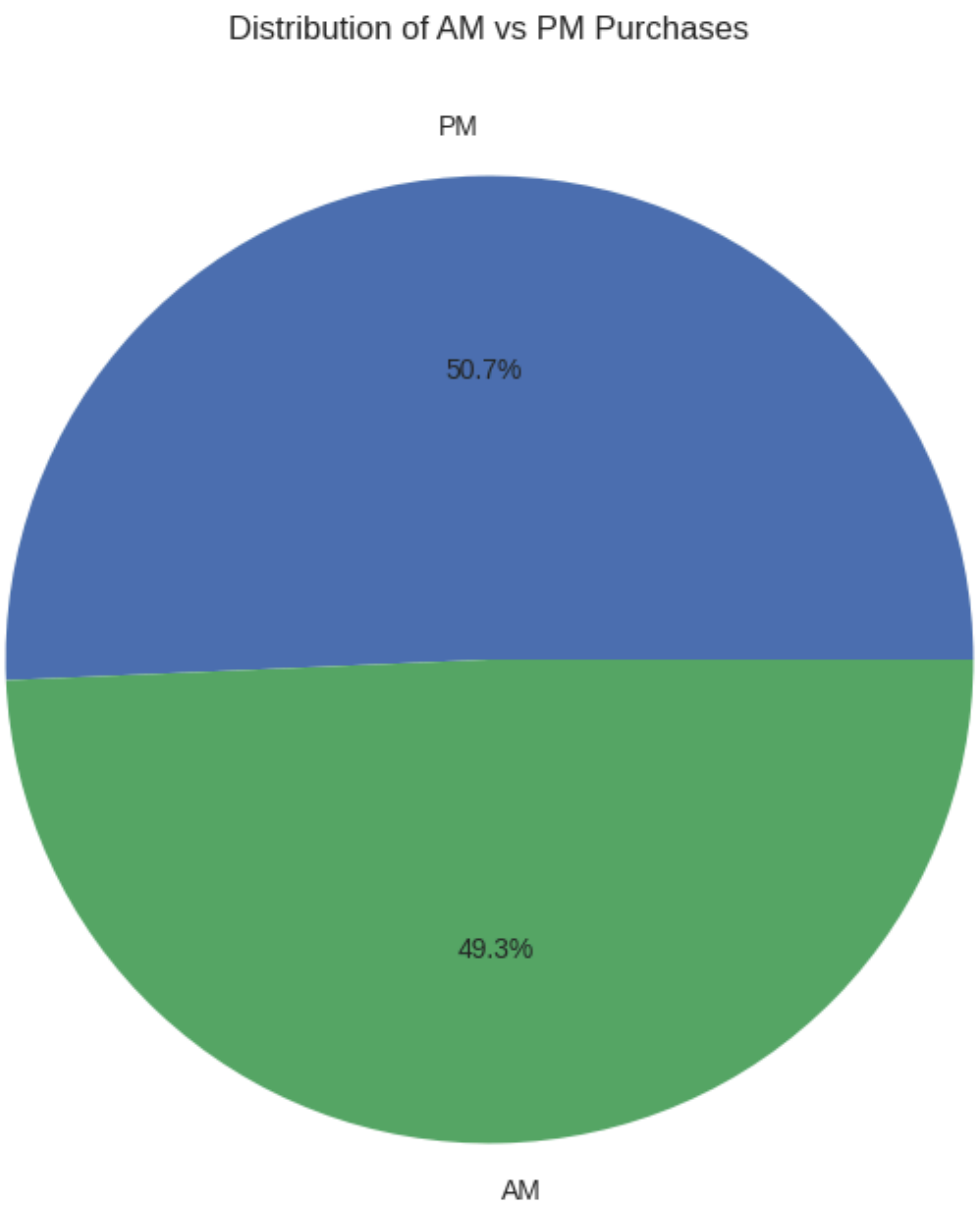
```
plt.show()
```



6. Pie Chart: AM/PM Distribution

- **Description: This pie chart illustrates the distribution of purchases made in the AM versus PM. Each slice of the pie represents either AM or PM, with the size indicating the proportion of purchases.**
- **Observation: The chart shows equal proportions of purchases made during AM and PM.**
- **Questions Answered:**
    - **Do customers prefer to shop in the morning or evening?**
    - **Is there a significant difference in purchase volume between AM and PM?**

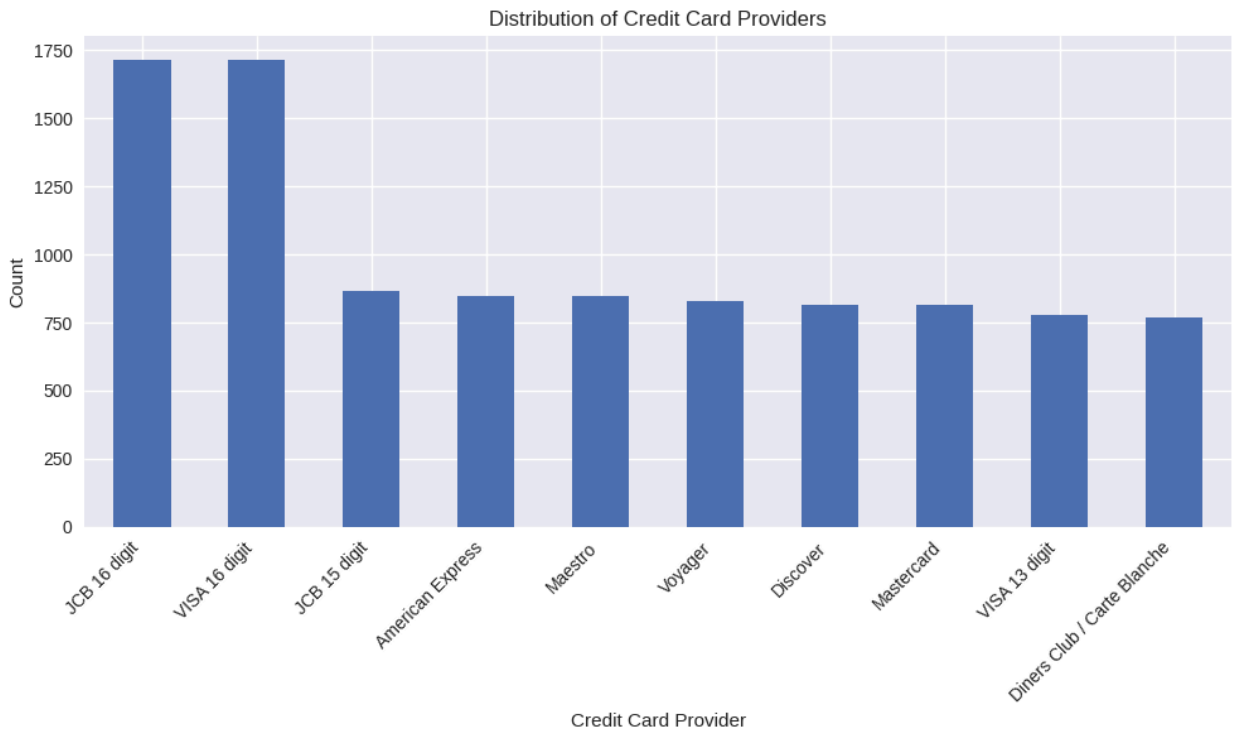
```
# 6. Pie Chart: AM/PM Distribution

plt.figure(figsize=(8, 8))

df['AM or PM'].value_counts().plot(kind='pie', autopct='%1.1f%%')

plt.title('Distribution of AM vs PM Purchases')

plt.ylabel('')

plt.show()
```



7. Bar Chart: Credit Card Provider Distribution

- **Description: This bar chart shows the distribution of purchases made with different credit card providers. The x-axis represents different providers, and the y-axis represents the number of purchases.**
- **Observation: JCB and Visa show a higher number of purchases, indicating their popularity among customers, rest all are evenly distributed.**
- **Questions Answered:**
    - **Which credit card providers are most frequently used by customers?**
    - **Does the choice of credit card provider correlate with purchase frequency?**

```
# 7. Bar Chart: Credit Card Provider Distribution

plt.figure(figsize=(10, 6))

df['CC Provider'].value_counts().plot(kind='bar')

plt.title('Distribution of Credit Card Providers')

plt.xlabel('Credit Card Provider')

plt.ylabel('Count')

plt.xticks(rotation=45, ha='right')

plt.tight_layout()

plt.show()
```

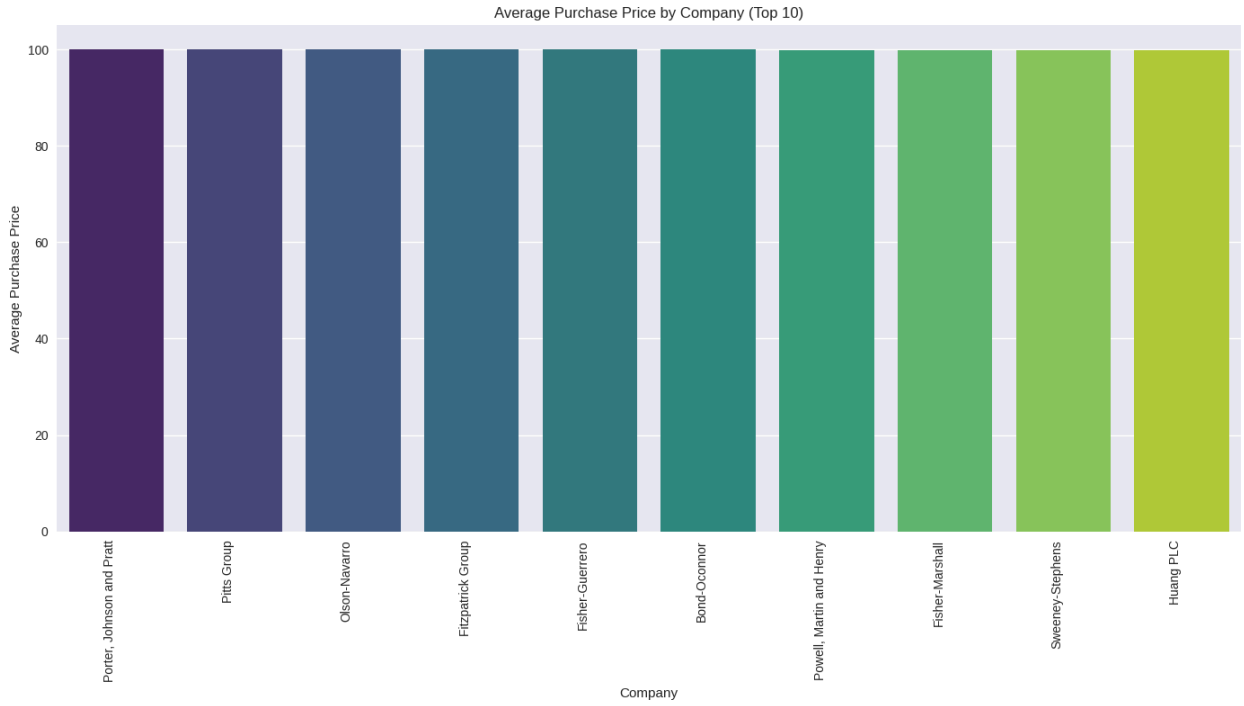


8. Bar Chart - Average Purchase Price by Company (Top 10)

- **Description: This bar chart displays the top 10 companies by average purchase price. The x-axis represents different companies, and the y-axis represents the average purchase price.**
- **Observation: The purchases made by all companies is almost the same.**
- **Questions Answered:**
    - **Which companies have the highest average purchase prices?**
    - **Are there specific companies associated with higher-value transactions?**

```
# 8. Bar Chart - Average Purchase Price by Company (Top 10)



# Calculate average purchase price by company

company_sales = df.groupby('Company')['Purchase
Price'].mean().sort_values(ascending=False)



# Select the top 10 companies

top_company_sales = company_sales.head(10)



plt.figure(figsize=(14, 8))

sns.barplot(x=top_company_sales.index, y=top_company_sales.values,
palette='viridis')



# Rotate labels for better readability

plt.xticks(rotation=90, ha='right', fontsize=10)



plt.title('Average Purchase Price by Company (Top 10)')

plt.xlabel('Company')

plt.ylabel('Average Purchase Price')

plt.tight_layout()
```

```
plt.show()
```

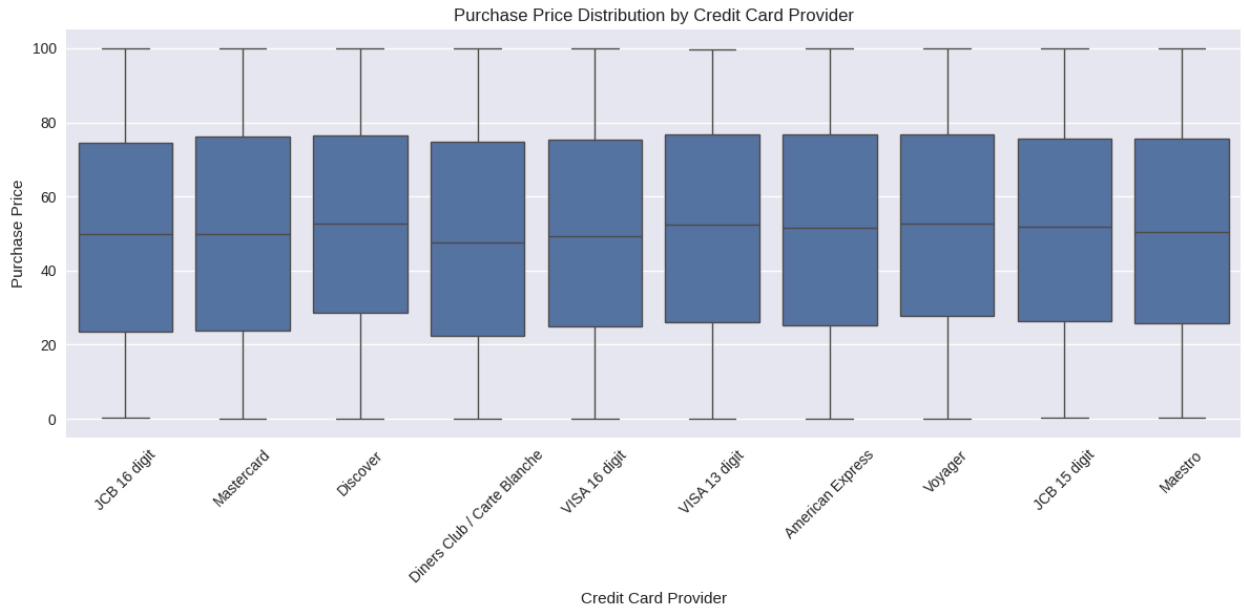


9. Boxplot: Purchase Price by Credit Card Provider

- **Description: This boxplot shows the distribution of purchase prices for each credit card provider. The x-axis represents different providers, and the y-axis represents the range of purchase prices.**
- **Observation: The boxplot shows that approximately the same money is transacted for each credit card provider.**
- **Questions Answered:**
    - **How does purchase price distribution vary by credit card provider?**
    - **Are certain providers associated with higher or more varied purchase prices?**

```
# 9. Boxplot: Purchase Price by Credit Card Provider

plt.figure(figsize=(12, 6))

sns.boxplot(x='CC Provider', y='Purchase Price', data=df)

plt.title('Purchase Price Distribution by Credit Card Provider')
```

```
plt.xlabel('Credit Card Provider')

plt.ylabel('Purchase Price')

plt.xticks(rotation=45)

plt.tight_layout()

plt.show()
```

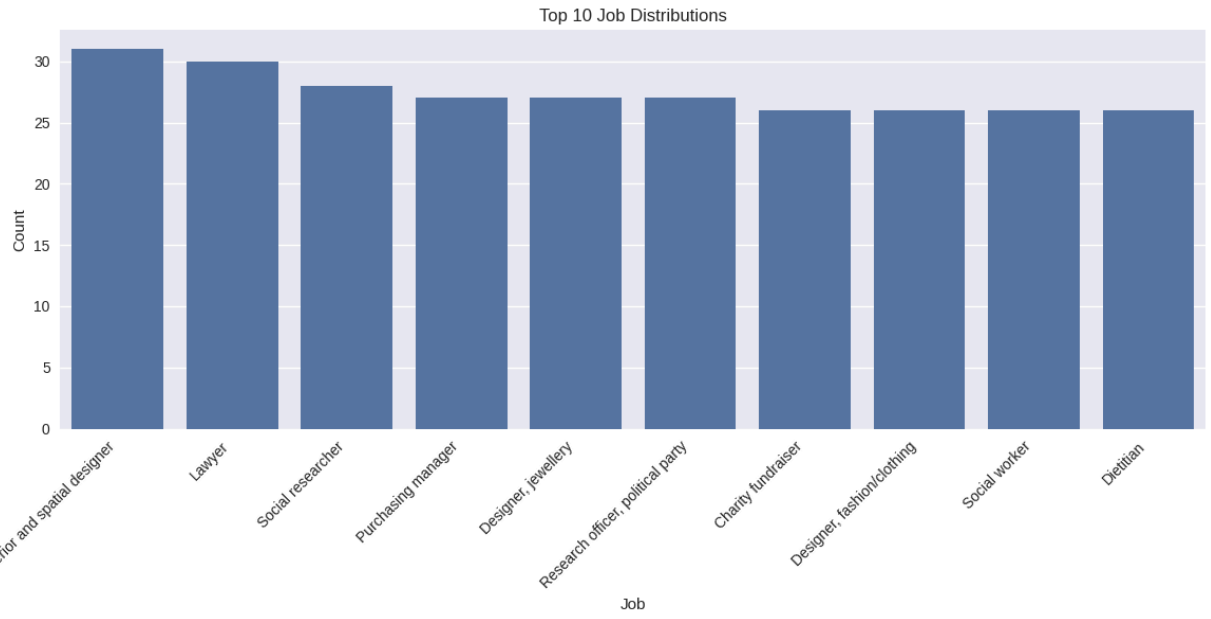


10. Countplot: Job Distribution

- **Description: This countplot shows the distribution of customers' job titles in the dataset. The x-axis represents different job titles, and the y-axis shows the count of customers in each job.**
- **Observation: Approximately all job titles have the same count.**
- **Questions Answered:**
    - **What is the distribution of job titles among customers?**
    - **Are certain professions more common among the customer base?**

```
# 10. Countplot: Job Distribution

plt.figure(figsize=(12, 6))

job_counts = df['Job'].value_counts().head(10)
```

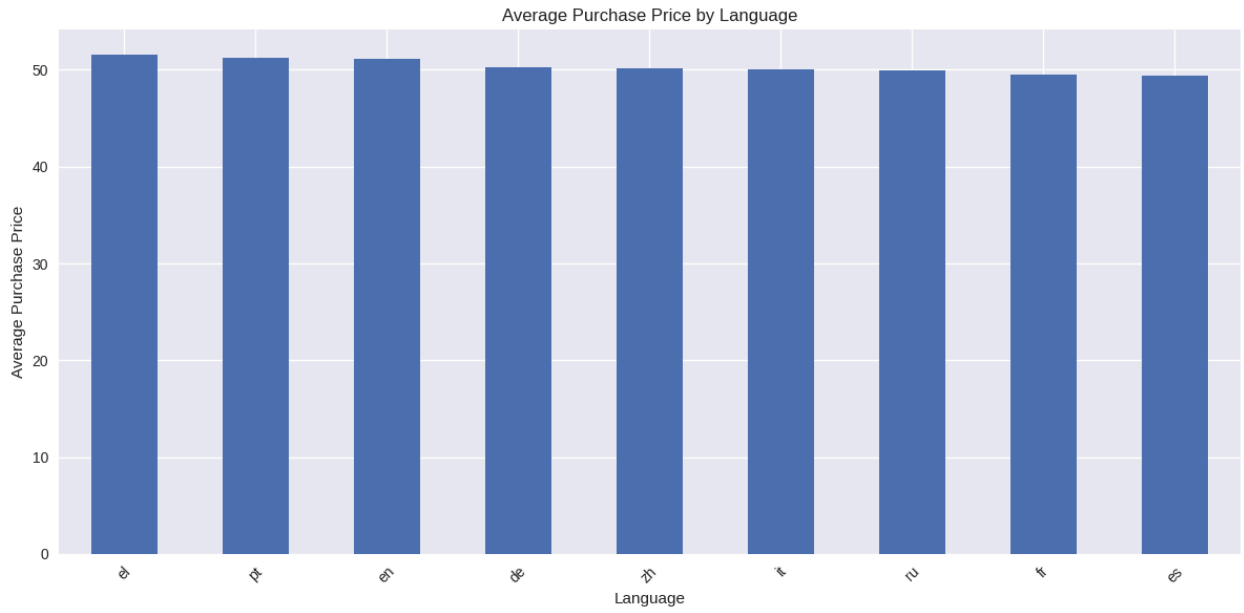
```
sns.barplot(x=job_counts.index, y=job_counts.values)

plt.title('Top 10 Job Distributions')

plt.xlabel('Job')

plt.ylabel('Count')

plt.xticks(rotation=45, ha='right')

plt.tight_layout()

plt.show()
```



11. Bar Chart: Average Purchase Price by Language

- **Description: This bar chart displays the average purchase price by the language preference of customers. The x-axis represents different languages, and the y-axis shows the average purchase price.**
- **Observation: Customers of all languages have approximately the same average purchase price.**
- **Questions Answered:**
    - **Which language groups are associated with higher average purchase prices?**
    - **Does language preference correlate with spending behavior?**

```
# 11. Bar Chart: Average Purchase Price by Language

avg_price_by_language = df.groupby('Language')['Purchase
Price'].mean().sort_values(ascending=False)

plt.figure(figsize=(12, 6))

avg_price_by_language.plot(kind='bar')

plt.title('Average Purchase Price by Language')

plt.xlabel('Language')

plt.ylabel('Average Purchase Price')

plt.xticks(rotation=45)

plt.tight_layout()

plt.show()
```



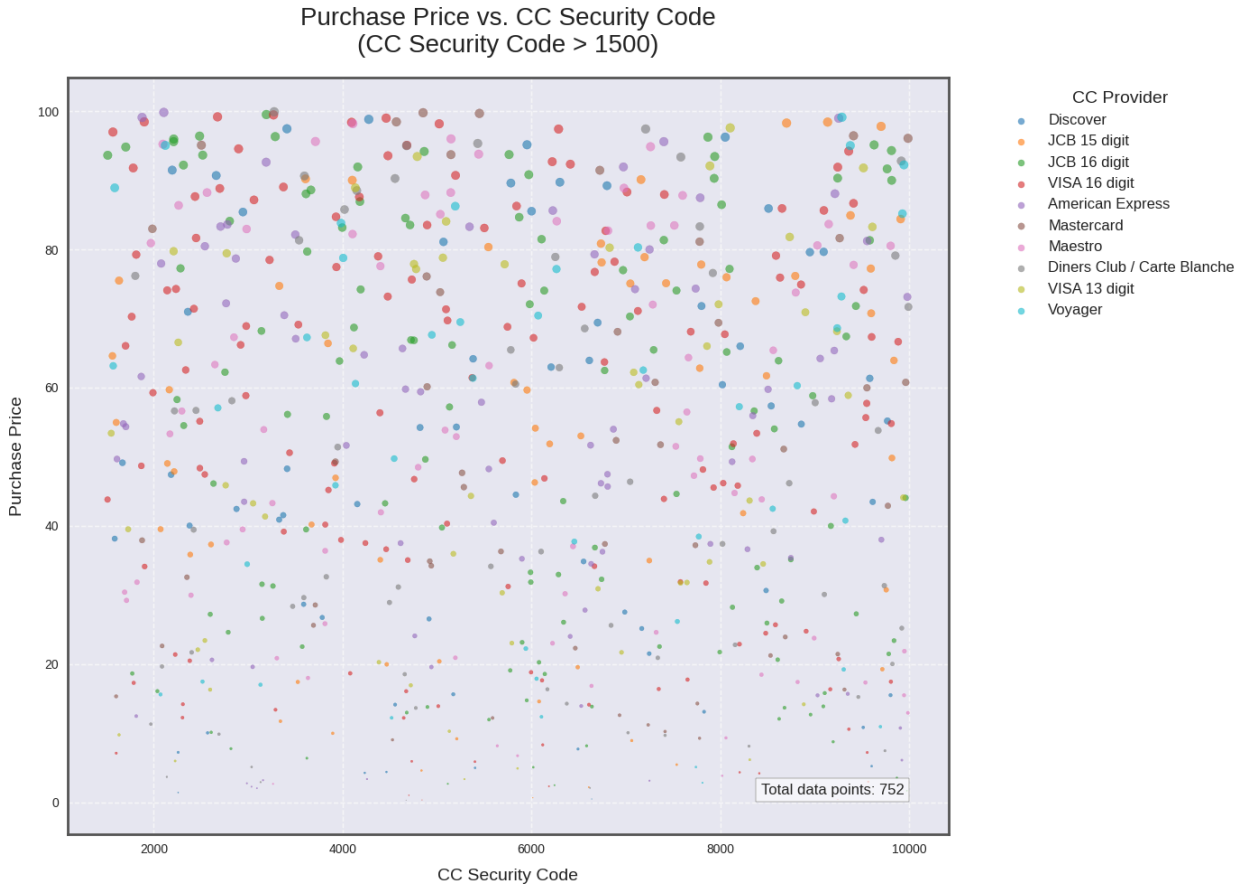
12. Bubble Plot: Purchase Price vs. CC Security Code, with CC Provider as color and Purchase Price as size

- **Description: This bubble plot visualizes the relationship between purchase price and CC security code, with the size of the bubbles representing purchase price and the color representing different CC providers.**
- **Observation: There are no clusters or patterns.**
- **Questions Answered:**
    - **Is there a relationship between CC security code and purchase price?**
    - **Do certain CC providers tend to be associated with higher purchase prices?**

```
# 12. Bubble Plot: Purchase Price vs. CC Security Code, with CC Provider
as color and Purchase Price as size

# Set the style for a more modern look

plt.style.use('seaborn')



# Create the figure and axis objects

fig, ax = plt.subplots(figsize=(14, 10))



# Filter the dataframe for CC Security Code > 1500

df_filtered = df[df['CC Security Code'] > 1500]



# Create a color map

color_map = plt.cm.get_cmap('tab10')

color_norm = plt.Normalize(vmin=0, vmax=len(df_filtered['CC
Provider'].unique()))



for i, provider in enumerate(df_filtered['CC Provider'].unique()):

    provider_data = df_filtered[df_filtered['CC Provider'] == provider]

    ax.scatter(provider_data['CC Security Code'],
```

```
                provider_data['Purchase Price'],
                s=provider_data['Purchase Price'] * 0.5,  # Size based on
Purchase Price
                alpha=0.6,
                label=provider,
                color=color_map(color_norm(i)))

# Customize the plot
ax.set_title('Purchase Price vs. CC Security Code\n(CC Security Code >
1500)', fontsize=20, pad=20)
ax.set_xlabel('CC Security Code', fontsize=14, labelpad=10)
ax.set_ylabel('Purchase Price', fontsize=14, labelpad=10)

# Add grid lines
ax.grid(True, linestyle='--', alpha=0.7)

# Add a subtle box around the plot
for spine in ax.spines.values():
    spine.set_edgecolor('#555555')
    spine.set_linewidth(2)

# Improve legend
ax.legend(title='CC Provider', bbox_to_anchor=(1.05, 1), loc='upper left',
fontsize=12, title_fontsize=14)
```

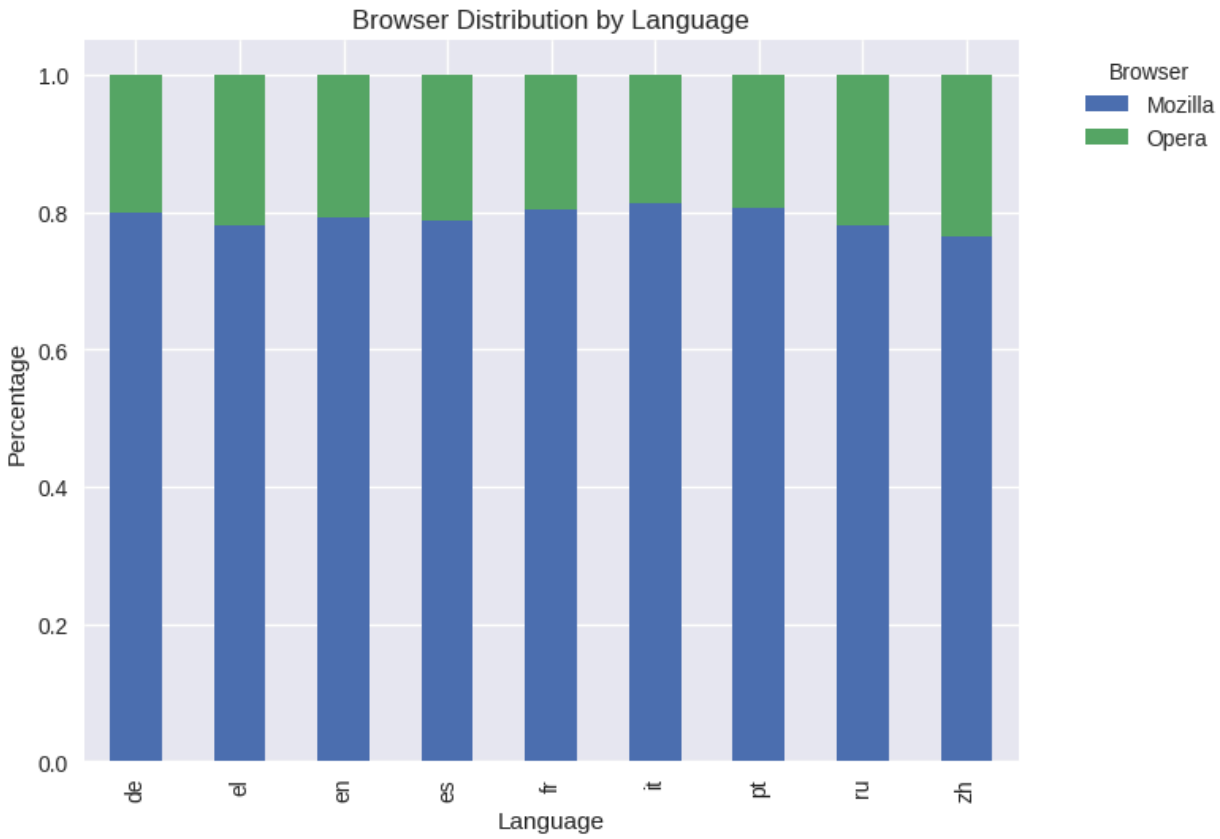
```
# Add text annotations
ax.text(0.95, 0.05, f'Total data points: {len(df_filtered)}',
        verticalalignment='bottom', horizontalalignment='right',
        transform=ax.transAxes, fontsize=12, bbox=dict(facecolor='white',
alpha=0.7))


plt.tight_layout()
plt.show()
```



13. Stacked Bar Chart: Browser Distribution by Language

- **Description: This stacked bar chart shows the distribution of browser usage among customers based on their language preference. The x-axis represents different browsers, and the y-axis represents the number of purchases, with stacks indicating different languages.**
- **Observation: The chart reveals that customers of all languages prefer mozilla..**
- **Questions Answered:**
    - **How does browser usage vary among different language groups?**
    - **Is there a correlation between language preference and browser choice?**

```
# 13. Stacked Bar Chart: Browser Distribution by Language
browser_lang = pd.crosstab(df['Language'], df['Browser
Info'].str.split('/').str[0])
browser_lang_pct = browser_lang.div(browser_lang.sum(axis=1), axis=0)
plt.figure(figsize=(12, 6))
browser_lang_pct.plot(kind='bar', stacked=True)
plt.title('Browser Distribution by Language')
plt.xlabel('Language')
plt.ylabel('Percentage')
plt.legend(title='Browser', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```


Browser Distribution by Language
Percentage
1.0
0.8
0.6
0.4
0.2
0.0
de
el
en
es
fr
it
pt
ru
zh
Language
Browser
Mozilla
Opera