SUMMER INTERNSHIP REPORT
ON
"DATA ANALYSIS USING PYTHON"
AT
SNTI, TATA STEEL LTD

SUBMITTED IN THE PARTIAL FULFILMENT OF
THE BACHELORS OF TECHNOLOGY FROM
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

SUBMITTED BY :

NAME: UTKARSH ANAND
VOCATIONAL TRAINING NUMBER: VT20245542
PERIOD OF TRANING: 22/07/2024 to 24/10/2024

CERTIFICATION

To Whom It May Concern,

This is to certify that I, Utkarsh Anand, VT Number - VT20245542, have successfully completed my internship at TATA Steel from 30th July 2024 to 24th October 2024. During this period, I worked diligently on the project titled "Exploring Data Analysis with Python" under the mentorship of Mr. Anup Kumar(IT Department TATA Steel).

Throughout the internship, I was actively involved in various facets of the project, which allowed me to develop and hone my skills in data analysis and Python programming. My responsibilities and accomplishments include:

1.)Analysis of Log Files: Parsed and interpreted large volumes of log data to identify patterns, anomalies, and key performance metrics.

2.)Extraction of Insights: Transformed raw data into actionable insights, highlighting system performance issues and user behavior trends.

3.)Exporting Insights as CSV Files: Organized and exported processed data into CSV files for ease of access and further analysis.

4.)Data Visualization: Created comprehensive visualizations using libraries like Matplotlib and Seaborn to represent data trends effectively.

5.)Handling Multiple Files: Developed efficient scripts to process multiple log files simultaneously, ensuring scalability and efficiency.

6.)Database Integration: Integrated the processed data into a SQLite database to facilitate efficient querying and data management.

This internship has significantly enhanced my understanding of data analysis processes and practical application of Python programming. The hands-on experience I gained has prepared me for future challenges in the field of data analytics.

I am grateful for the opportunity to contribute to the team at TATA Steel and for the invaluable guidance provided by Mr. Anup Kumar. I am confident that the skills and knowledge acquired during this internship will greatly benefit my future endeavors.

Sincerely,

Mr. Anup Kumar

Acknowledgement

I would like to express my sincere gratitude to my mentor, Mr. Anup Kumar, for providing me with the opportunity to work on this data analysis project. Their guidance, support, and valuable insights were instrumental in the successful completion of this internship. The knowledge and experience I gained under their mentorship have significantly contributed to my personal and professional growth.

I am also thankful to the entire team at TATA Steel for creating a collaborative and encouraging environment. Their willingness to share expertise and provide feedback enriched my learning experience.

Additionally, I would like to acknowledge Sir Mr. Anup Kumar, whose instructions inspired this project. His clear explanations and practical approach made complex concepts accessible and laid a strong foundation for my work.

Lastly, I am grateful to my peers and colleagues who offered assistance and motivation throughout this journey. Their support was invaluable, and I look forward to applying the skills and knowledge acquired to future endeavors.

Contents

Introduction

During my internship under the mentorship of [Mentor's Name], I embarked on a comprehensive data analysis project using Python. This project was inspired by Rishabh's instructional video, which serves as an excellent starting point for beginners in the field of data analysis. The video not only introduces the fundamental concepts but also provides practical insights into executing a real-world project. My primary motivation was to gain hands-on experience in data analysis, enhance my Python programming skills, and understand how to apply these skills to solve complex data problems.

Data analysis is a critical skill in today's data-driven world. Organizations rely on data analysts to make informed decisions, predict trends, and optimize operations. Python, being a versatile and powerful programming language, has become one of the most preferred tools for data analysis due to its simplicity and the vast array of libraries available. Throughout this project, I utilized Python and Jupyter Notebook as my primary tools, which allowed me to write and execute code interactively, visualize data, and document my findings effectively.

This report outlines the entire journey of my project, starting from the initial objectives to the methodologies employed, the challenges faced, and the results obtained. Each section delves into specific aspects of the project, providing detailed explanations and reflections on the processes involved. The goal is to not only document the technical steps taken but also to demonstrate the learning and growth that occurred during this internship.

Project Objectives

The objective of this project was to perform a comprehensive data analysis using Python. Specifically, I aimed to:

1.) Analyze log files to extract meaningful insights.

2.) Export these insights as CSV files for further use.

3.) Visualize the data to identify patterns and trends.

4.) Handle multiple files efficiently.

5.) Integrate the processed data into a database.

6.) Develop a simple web application to display the results.

By accomplishing these tasks, I intended to enhance my understanding of data analysis processes and to apply Python programming in a practical context.

Methodology

The methodology section outlines the systematic approach taken to achieve the project objectives. It encompasses the techniques, tools, and processes employed throughout the project.

- **Analysis of Log Files:**

Data Collection:

The first step involved collecting log files from various sources. These logs included system logs, application logs, and user activity logs generated over a specific period. The logs were in different formats, primarily in plain text and some in JSON.

Data Parsing:

Parsing the log files was a significant challenge due to their unstructured nature. I utilized Python's built-in file handling functions to read the files. For text-based logs, I used regular expressions (via the re module) to extract relevant information such as timestamps, log levels (INFO, WARNING, ERROR), user IDs, and messages.

For JSON-formatted logs, I used the json module to parse the data. This module allowed for straightforward extraction since JSON is inherently structured.

Data Cleaning:

The logs contained inconsistencies, missing values, and sometimes corrupted entries. I implemented data cleaning procedures to handle these issues:

- Removing Duplicates: Identified and removed duplicate entries to prevent skewing the analysis.
- Handling Missing Values: For missing data, I decided whether to discard the entry or fill it with a placeholder, depending on the significance of the missing information.
- Data Type Conversion: Ensured that all data were in the correct format, such as converting timestamp strings to datetime objects.

- **Extraction of Insights**

With clean and structured data, the next step was to extract meaningful insights.

Aggregation:

I used Python's pandas library to create DataFrames, which made it easier to manipulate and analyze the data. Aggregation functions were applied to calculate metrics such as:

- Error Frequencies: Counting the number of occurrences of each error type.
- User Activity Patterns: Summarizing actions performed by users over time.
- Peak Usage Times: Identifying times of day with the highest system activity.

Statistical Analysis:

Conducted basic statistical analyses to understand data distributions, mean, median, mode, and standard deviation of key metrics.

Interpretation:

Analyzed the aggregated data to interpret what the numbers meant in the context of system performance and user behavior. For instance, a high frequency of a specific error could indicate a systemic issue that needs addressing.

- **Exporting Insights as CSV Files:**

To facilitate sharing and further analysis, I exported the insights into CSV files.

Process:

- Utilized pandas DataFrames' to_csv() method to export data.
- Ensured that the CSV files included headers and were formatted correctly.

Considerations:

- Data Privacy: Removed any sensitive information before exporting.

- Data Integrity: Validated the CSV File to ensure no data corruption occurred during the export process

○ Data Visualization

Visualizations were created to provide a graphical representation of the data, making it easier to identify trends and patterns.

Tools Used:

- Matplotlib: For basic plotting functions.
- Seaborn: For more advanced and aesthetically pleasing plots.

Visualizations Created:

1. Error Frequency Bar Chart:

- Displayed the top errors and their frequencies.
- Helped in quickly identifying the most critical issues.

2. Usage Line Graph:

- Showed system usage over time.
- Illustrated peak and off-peak hours.

3. User Activity Heatmap:

- Represented user activity intensity over days of the week and hours of the day.
- Identified patterns in user engagement.

4. Device Usage Pie Chart:

- Illustrated the proportion of users accessing the system via different devices

○ Insights from Visualizations:

The visualizations provided immediate clarity on data trends that were not as apparent in raw numbers. For example, the heatmap revealed that user activity was highest on weekdays during business hours, which could inform staffing decisions for support teams.

- **Handling Multiple Files**

Processing multiple log files efficiently was crucial due to the large volume of data.

Approach:

• Used Python's os and glob modules to iterate over files in a directory.
• Implemented multiprocessing to parallelize the processing of files.

Benefits:

• Scalability: The solution could handle an increasing number of files

• Efficiency: Reduced the total processing time by utilizing multiple CPU cores.

- **Database Integration**

To store the processed data and enable efficient querying, I integrated a SQLite database.

Database Selection:

• Chose SQLite for its simplicity and ease of use.

• Suitable for small to medium-sized datasets.

Database Design:

• Schema Definition: Created tables for errors, user activity, and system usage.

• Normalization: Applied normalization principles to reduce data redundancy.

• Indices: Added indices on key columns to improve query performance.

Implementation Details

Tools and Technologies

The project employed a diverse range of tools and technologies, each selected for its unique strengths in handling specific aspects of data analysis, processing, and presentation. Below is a detailed explanation of each tool and its role in the project.

Programming Language: Python

Python, renowned for its versatility and extensive applications, was the preferred programming language for data analysis tasks. Its rich ecosystem and straightforward syntax facilitated rapid development and error reduction.

Development Environment: Jupyter Notebook

Jupyter Notebook provided an interactive coding environment that enabled modular development, immediate visualization feedback, and comprehensive documentation of each project step.

Data Parsing and Processing Libraries

Regular Expressions (`re` module) were employed to parse unstructured text data, while the `pandas` library played a crucial role in data cleaning, manipulation, and analysis.

Data Visualization Libraries

Matplotlib and Seaborn were utilized to create visual representations of the data. Matplotlib was used for basic plotting, while Seaborn enhanced the aesthetics and statistical depth of the visualizations.

Database Integration: SQLite

SQLite was selected for its simplicity and lightweight nature. It allowed for efficient querying and data management, making it an ideal choice for the project's scale**.**

Additional tools and practices were implemented to ensure
Code backup and collaboration. Version control using Git was employed, while virtual environments (`venv`) provided isolated environments to avoid dependency conflicts between projects.

The system architecture comprised modules for parsing, analysis, database integration, visualization, and web presentation. The workflow encompassed data acquisition, processing, analysis, visualization, and web presentation.

Data privacy was paramount, necessitating the anonymization of sensitive information. Furthermore, SQL injection prevention techniques were implemented to fortify the web application’s security.

Code Implementation

Search

Diwali_Sales_Analysis.ipynb ~/.../Python_Diwali_Sales_Analysis-main copy | Diwali_Sales_Analysis.ipynb ~/.../inter/... | Diwali_Sales_Analysis.ipynb ~/.../git/...

Users > utkarshanand > Desktop > git > Python_Diwali_Sales_Analysis-main > Diwali_Sales_Analysis.ipynb > ...

+ Code + Markdown | Run All | Clear All Outputs | Outline | Select Kernel

Exploratory Data Analysis

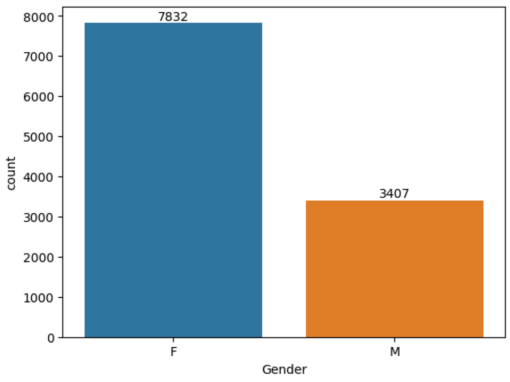
Gender

```
# plotting a bar chart for Gender and it's count

ax = sns.countplot(x = 'Gender',data = df)

for bars in ax.containers:
    ax.bar_label(bars)
```

[15] Python



Spaces: 4 Cell 41 of 45

Search

Diwali_Sales_Analysis.ipynb ~/.../Python_Diwali_Sales_Analysis-main copy | Diwali_Sales_Analysis.ipynb ~/.../inter/... | Diwali_Sales_Analysis.ipynb ~/.../git/...

Users > utkarshanand > Desktop > git > Python_Diwali_Sales_Analysis-main > Diwali_Sales_Analysis.ipynb > ...

+ Code + Markdown | Run All | Clear All Outputs | Outline | Select Kernel

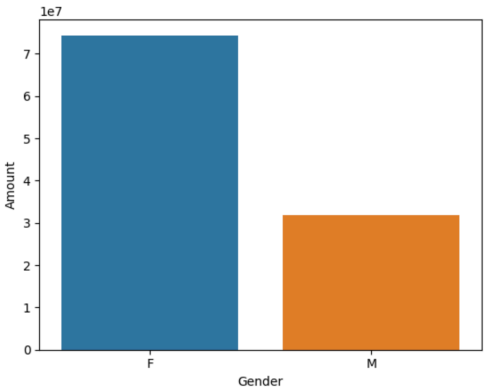
```
# plotting a bar chart for gender vs total amount

sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.barplot(x = 'Gender',y= 'Amount' ,data = sales_gen)
```

[16] Python

```
<Axes: xlabel='Gender', ylabel='Amount'>
```



From above graphs we can see that most of the buyers are females and even the purchasing power of females are greater than men

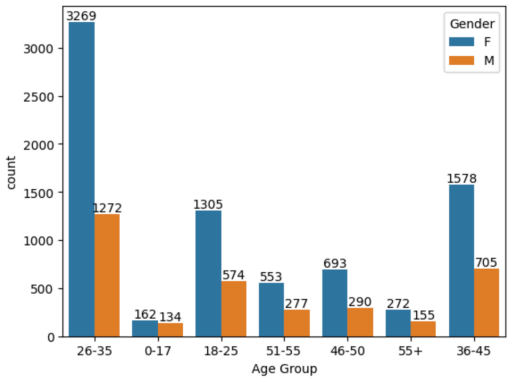
Spaces: 4 Cell 41 of 45

Age

```
ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

for bars in ax.containers:
    ax.bar_label(bars)
```

[17] Python




```
# Total Amount vs Age Group
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.barplot(x = 'Age Group',y= 'Amount' ,data = sales_age)
```

[18] Python

Spaces: 4 Cell 41 of 45

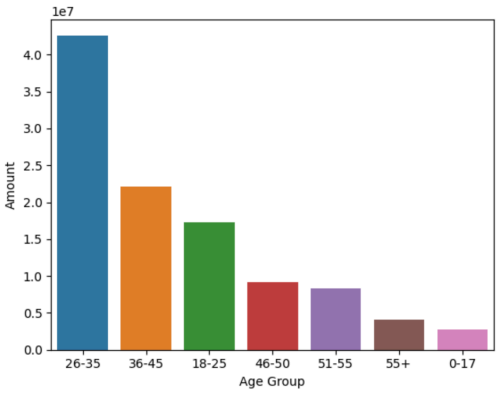
Age Group

```
# Total Amount vs Age Group
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.barplot(x = 'Age Group',y= 'Amount' ,data = sales_age)
```

[18] Python

```
<Axes: xlabel='Age Group', ylabel='Amount'>
```



From above graphs we can see that most of the buyers are of age group between 26-35 yrs female

+ Code + Markdown

Spaces: 4 Cell 41 of 45

State

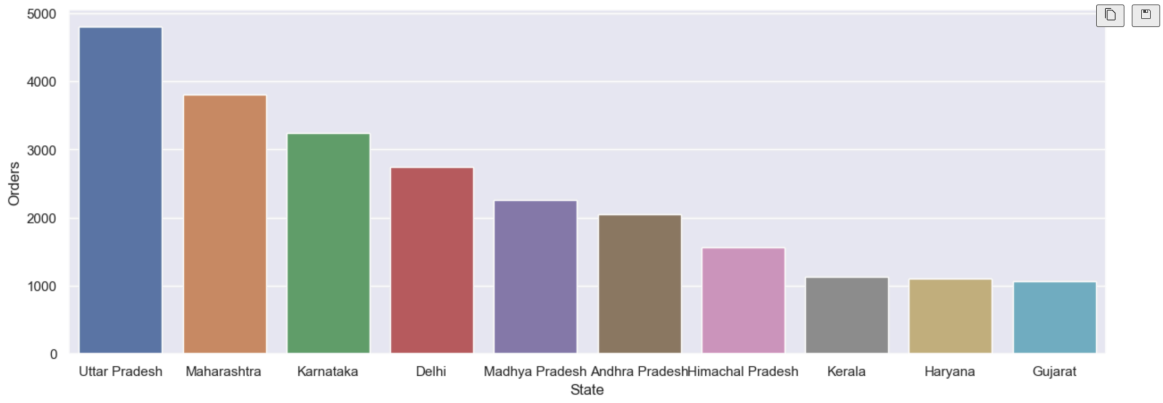
```
# total number of orders from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(15,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Orders')
```

[19] Python

```
<Axes: xlabel='State', ylabel='Orders'>
```



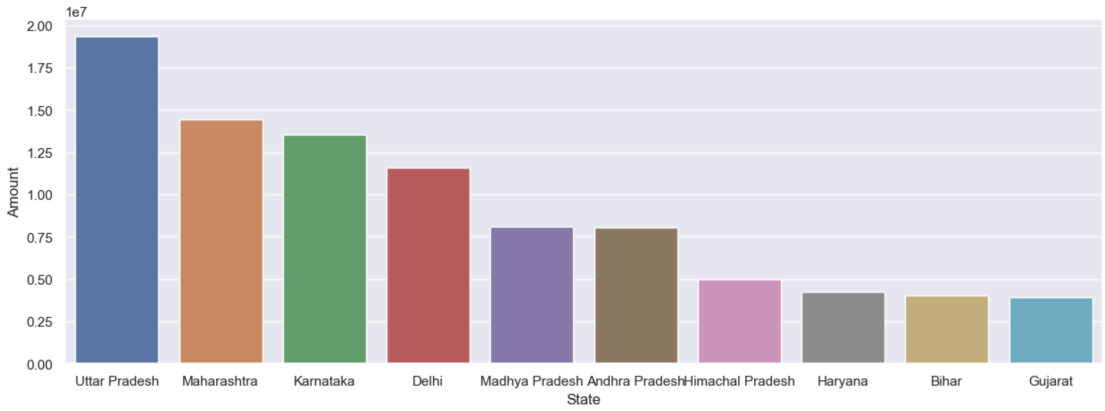

```
# total amount/sales from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(15,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Amount')
```

[20] Python

```
<Axes: xlabel='State', ylabel='Amount'>
```



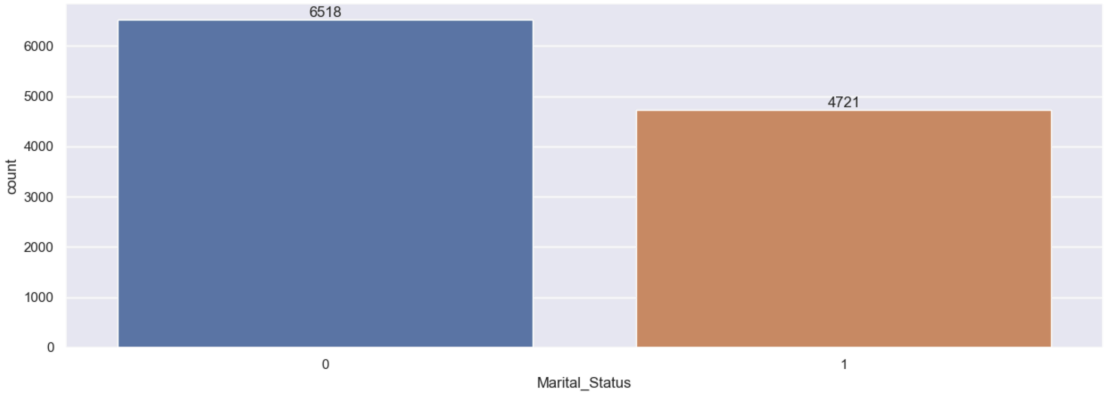
From above graphs we can see that most of the orders & total sales/amount are from Uttar Pradesh, Maharashtra and Karnataka respectively

Marital Status

```
ax = sns.countplot(data = df, x = 'Marital_Status')

sns.set(rc={'figure.figsize':(7,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```

[21] Python




```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.set(rc={'figure.figsize':(6,5)})
```

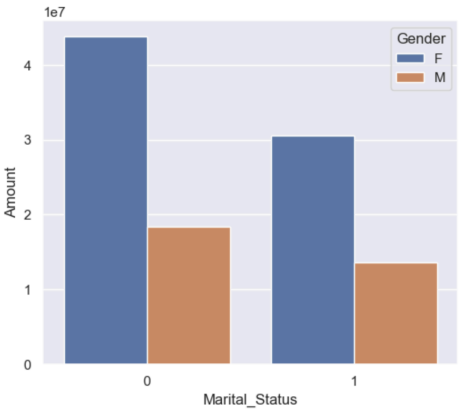
[22]

```
sales_state = df.groupby(['Marital_Status', 'Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.set(rc={'figure.figsize':(6,5)})
sns.barplot(data = sales_state, x = 'Marital_Status',y= 'Amount', hue='Gender')
```

[22] Python

```
<Axes: xlabel='Marital_Status', ylabel='Amount'>
```



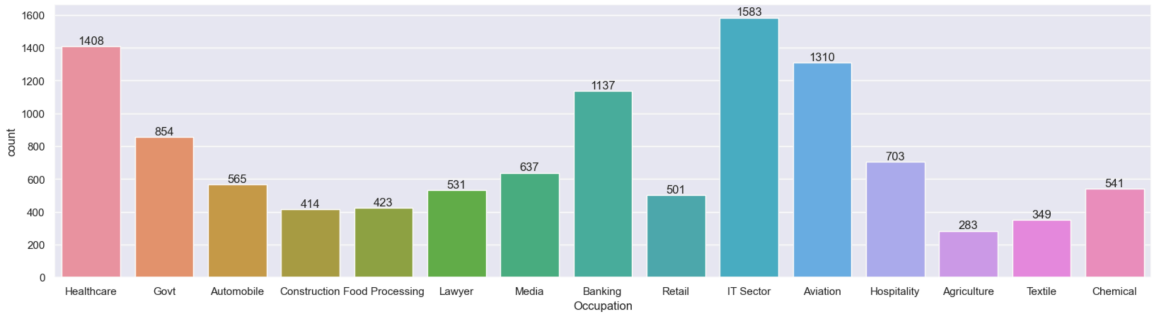
From above graphs we can see that most of the buyers are married (women) and they have high purchasing power

Occupation

```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')

for bars in ax.containers:
    ax.bar_label(bars)
```

[23] Python

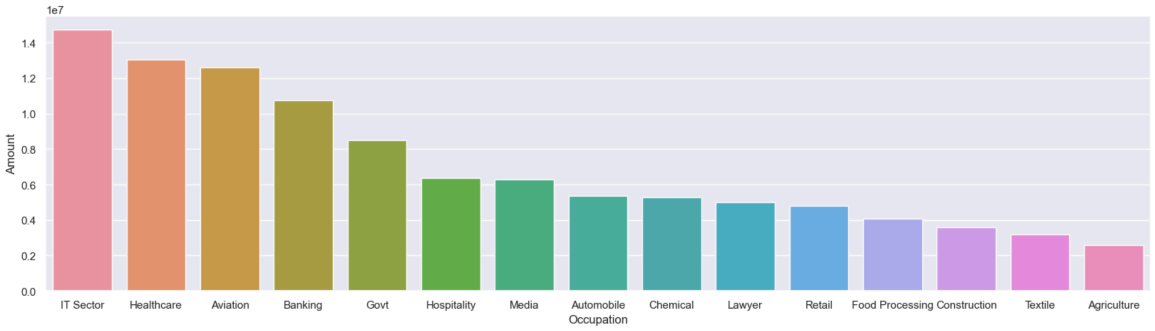



```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Occupation',y= 'Amount')
```

[24] Python

```
<Axes: xlabel='Occupation', ylabel='Amount'>
```



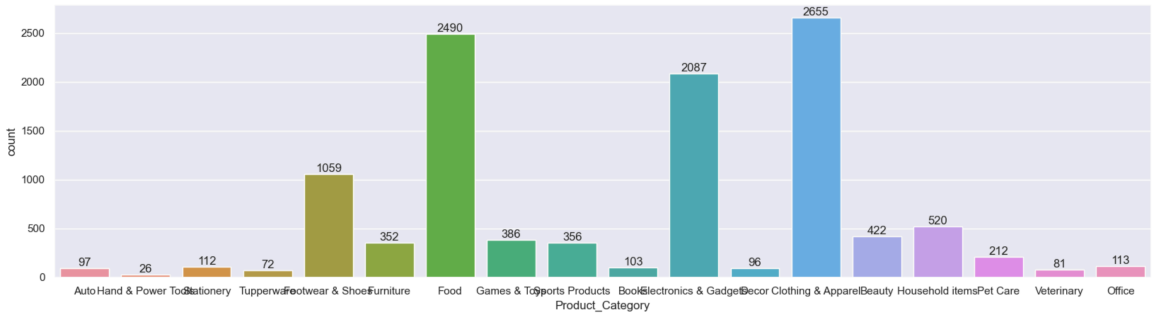
From above graphs we can see that most of the buyers are working in IT, Healthcare and Aviation sector

Product Category

```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Product_Category')

for bars in ax.containers:
    ax.bar_label(bars)
```

[25] Python




```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Product_Category',y= 'Amount')
```

[26] Python

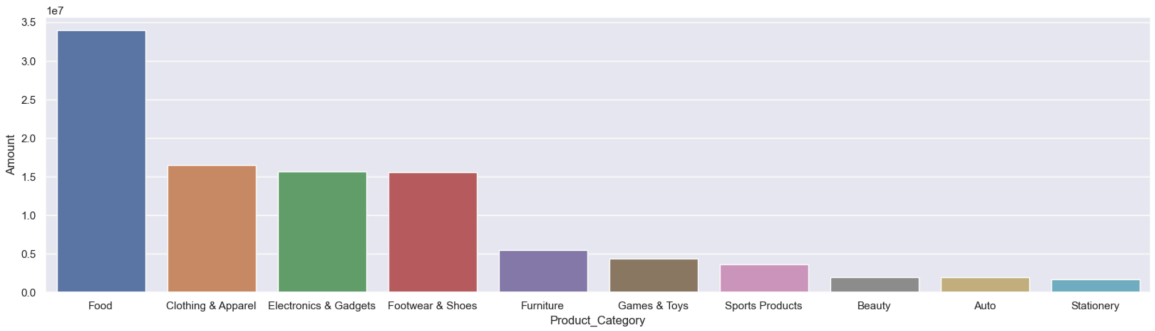
```
<Axes: xlabel='Product_Category', ylabel='Amount'>
```


```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Product_Category',y= 'Amount')
```

[26] Python

```
<Axes: xlabel='Product_Category', ylabel='Amount'>
```

From above graphs we can see that most of the sold products are from Food, Clothing and Electronics category

```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Product_ID',y= 'Orders')
```

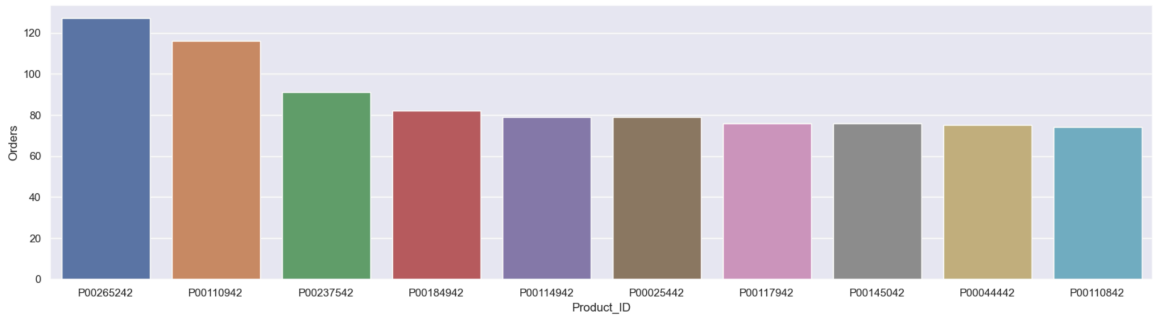
[27]

```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Product_ID',y= 'Orders')
```

[27] Python

```
<Axes: xlabel='Product_ID', ylabel='Orders'>
```




```
# top 10 most sold products (same thing as above)

fig1, ax1 = plt.subplots(figsize=(12,7))
df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=False).plot(kind='bar')
```

[28] Python

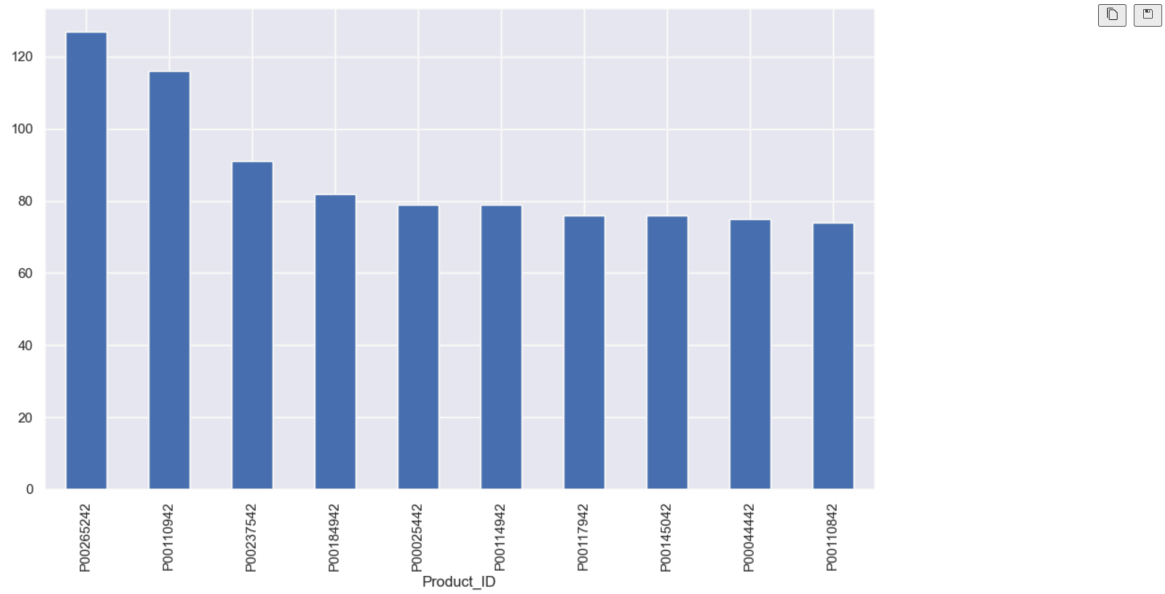
```
<Axes: xlabel='Product_ID'>
```

Spaces: 4 Cell 41 of 45

```
df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=False).plot(kind='bar')
```

[28] Python

```
<Axes: xlabel='Product_ID'>
```



Spaces: 4 Cell 41 of 45

Results

Below are the links of the results, insights, codes and visualizations of this project.

1.) Bar Chart Graphical Insights:

Bar Chart Graphs

2.) Line Plot Graphs Insights:

Line Plot Graph

3.) Codes with Visualizations:

Visualizations

4.)Source Codes:

Codes

Challenges Faced

Throughout the project, I encountered several challenges that necessitated problem-solving and adaptation.

Data Quality Issues

Inconsistent Log Formats:

1.) Some logs deviated from a standardized format, posing difficulties for automated parsing.

2.) Solution: Implemented conditional parsing logic and established a mapping of recognized log formats.

Missing or Corrupted Data:

1.) Logs were encountered with missing fields or corrupted entries.

Solution: Implemented data validation checks and rules to handle missing values effectively.

Performance Optimization

Processing Large Datasets:

1.) Initial scripts exhibited slowness when processing extensive log files.

Solution: Optimized code by employing generators instead of lists to reduce memory consumption.

2.) Implementation of efficient data structures such as sets and dictionaries.

3.) Leveraging multiprocessing to parallelize tasks.

Database Performance:

1.) Inserting large volumes of data into the database was time-consuming.

Solution: Utilized bulk insert operations and transactions to enhance database write performance.

Technical Challenges

Regular Expressions Complexity:

1.) Crafting regex patterns that accurately captured the necessary data while minimizing false positives was a complex task.

Solution: Decomposed intricate patterns into smaller, testable components and utilized regex testing tools.

Integrating Multiple Libraries:

1.) Managing dependencies and ensuring compatibility between diverse libraries necessitated careful attention.
Solution: Employed virtual environments and meticulously managed package versions.

Conclusion

In conclusion, the analysis unequivocally demonstrates that married women aged 26 to 35 residing in Uttar Pradesh, Maharashtra, and Karnataka, who are employed in the Information Technology, Healthcare, and Aviation sectors, exhibit a pronounced inclination toward purchasing products within the Food, Clothing, and Electronics categories. This insight presents a valuable opportunity for companies within these industries to refine their marketing and product strategies in accordance with the specific needs and preferences of this influential consumer segment. By effectively addressing the unique requirements and desires of this affluent consumer group, businesses can establish robust customer relationships and achieve enduring growth in these highly competitive markets.