INT375

PROJECT REPORT

(Project Semester January-April 2025)

Car Sales Data Analysis with Python

\

Submitted by

Shaurya Verma

Registration No: 12322585

Programme and Section: B.Tech CSE K23EP

Course Code: INT375

Under the Guidance of

Dr. Tanima Thakur (UID: 23532)

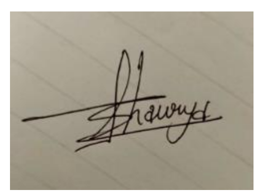
Discipline of CSE/IT

Lovely School of Computer Science and Engineering

Lovely Professional University, Phagwara

DECLARATION

I, Shaurya Verma, student of B.Tech under CSE/IT Discipline at, Lovely Professional University, Punjab, hereby declare that all the information furnished in this project report is based on my own intensive work and is genuine.



Date: 13-April-2025

Registration No. 12322585

Signature

Name of the student: Shaurya Verma

<u>CERTIFICATE</u>

This is to certify that Shaurya Verma bearing Registration no. 12322585 has completed INT375 project titled, **"Car Sales Data Analysis with Python"** under my guidance and supervision. To the best of my knowledge, the present work is the result of his/her original development, effort and study.

Signature and Name of the Supervisor
Dr. Tanima Thakur
Assistant Professor, Discipline of CSE/IT
School of Computer Science and Engineering
Lovely Professional University
Phagwara, Punjab.

Date:

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who supported me throughout the completion of this project titled **“Car Sales Data Analysis with Python.”**

First and foremost, I would like to extend my heartfelt thanks to **Dr. Tanima Thakur**, my respected project supervisor, for her invaluable guidance, continuous support, and encouragement. Her expert advice, timely feedback, and motivation played a crucial role in the successful completion of this project.

I am also thankful to the faculty and staff of the **School of Computer Science and Engineering, Lovely Professional University**, for fostering a dynamic academic environment and providing the necessary resources to carry out this project effectively.

A special note of appreciation goes to my peers, friends, and family for their unwavering moral support and encouragement throughout the project journey.

Lastly, I would like to acknowledge the car sales dataset that served as the foundation for the analysis presented in this report.

Shaurya Verma

Registration No.: 12322585

TABLE OF CONTENTS

1. Introduction

The automotive industry is one of the most data-intensive sectors in today's economy. With growing competition, evolving customer demands, and the need for sustainable practices, the role of data analysis has become increasingly critical. This project, titled **"Car Sales Data Analysis with Python,"** presents a detailed exploratory analysis of car sales data to derive meaningful insights from historical records.

The dataset used for this project consists of **18,000+ entries**, with each record capturing various details related to individual car sales. These details include car make, model, year, price, sale date, salesperson name, profit, and other business-relevant metrics. The dataset provides a comprehensive view of the sales landscape, which enables a deep dive into sales trends and performance metrics.

The objective of this project is to perform thorough **Exploratory Data Analysis (EDA)** using the Python programming language. The EDA process involved the following steps:

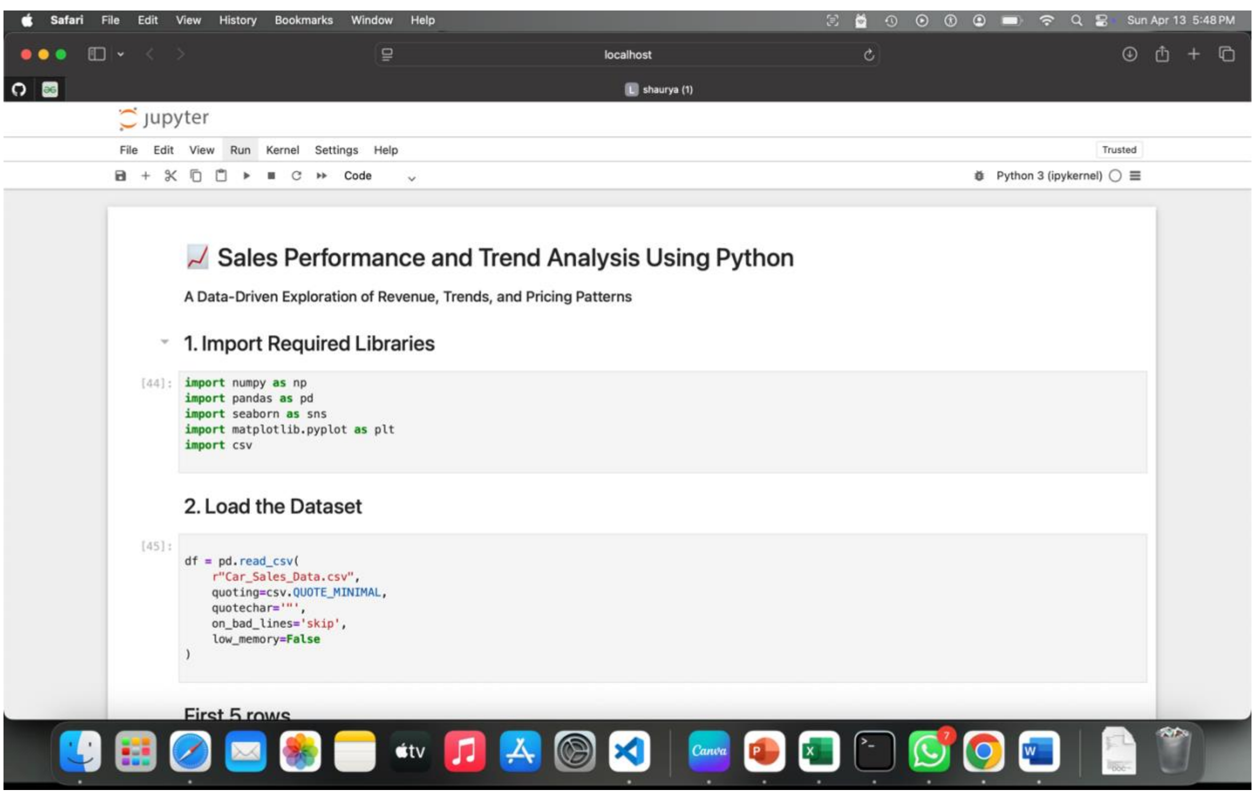
- **Importing essential Python libraries** such as pandas, numpy, matplotlib, and seaborn.
- **Data cleaning**, including the removal of unnecessary columns and handling of missing values.
- **Feature engineering**, where new variables like profit were calculated.
- **Statistical analysis**, such as generating correlation and covariance matrices.
- **Data visualization**, using various plots to uncover patterns and relationships.

The project further breaks down the analysis into six specific objectives:

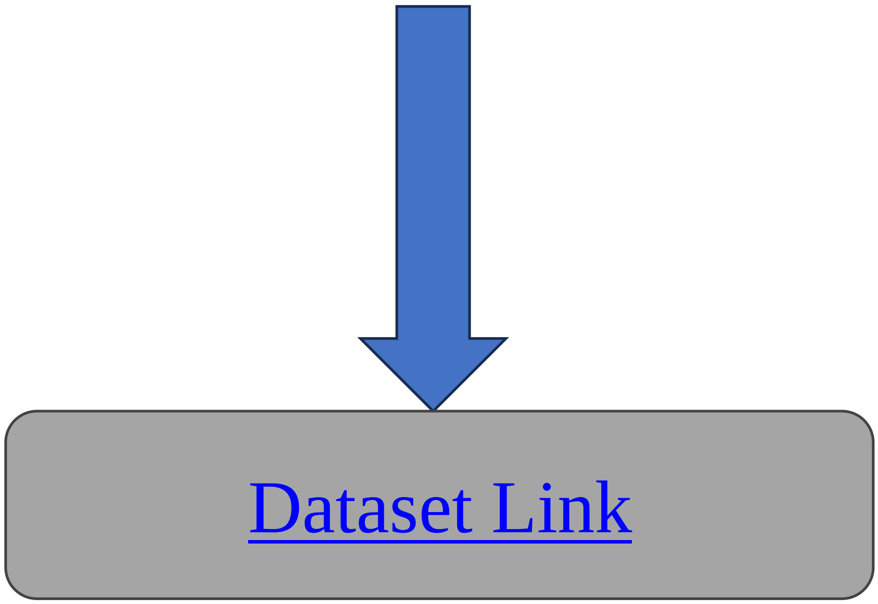
1. Understanding the **correlation between numerical features** to identify influential variables.

2. Using a **pairplot** to visualize the relationships and distributions among key numerical attributes.
3. Identifying the **top 5 salespeople by revenue** to evaluate sales performance.
4. Analyzing the **monthly car sales trend** to observe seasonality and fluctuations in sales volume.
5. Examining **price distributions by car make** using a violin plot to highlight variability and market positioning.
6. Plotting a **KDE (Kernel Density Estimation) of profit distribution** to understand overall business profitability.

Through this structured approach, the project not only demonstrates practical implementation of data science techniques but also showcases how Python can be effectively used for real-world business analysis. This report highlights key insights that could help automobile companies optimize their sales strategies, improve marketing efforts, and enhance overall business intelligence.



2. Source of Dataset




```
[69]: df.head(10)
```

[69]:

	ID	Sales_Person	Customer_Name	Car_Make	Car_Model	Car_Year	Sale_Price	Commission_Rate	Commission_Earned	Year	Month	Month_ID	
0	1.0	Christopher Murphy	Nicholas Price	Toyota	Silverado	2020.0	30345.0	0.127035	3854.87	2022.0	August	8.0	Wedı
1	2.0	Megan Gibson	Joshua Clark	Ford	Corolla	2016.0	48949.0	0.136564	6684.70	2023.0	March	3.0	M
2	3.0	Erin Sawyer MD	Kelsey Peterson	Nissan	Corolla	2010.0	22289.0	0.088097	1963.59	2022.0	December	12.0	Tu
3	4.0	Gregory Lee	Ana Jones	Nissan	Silverado	2015.0	28857.0	0.148810	4294.21	2022.0	June	6.0	Wedı
4	5.0	Nancy Hamilton	Sabrina Mason	Ford	F-150	2010.0	46327.0	0.058813	2724.63	2022.0	September	9.0	Tu
5	6.0	Jonathan Young	Steven Duffy	Nissan	Silverado	2012.0	34179.0	0.054627	1867.08	2023.0	March	3.0	Wedı
6	7.0	Mrs. Jennifer Rice	Jay Lawrence	Honda	Corolla	2021.0	29004.0	0.132438	3841.24	2022.0	May	5.0	Sa
7	8.0	Bryan Chang	Tina Myers	Chevrolet	Civic	2013.0	41129.0	0.108007	4442.20	2022.0	October	10.0	M
8	9.0	Carla Brandt	Monique Benton	Nissan	Altima	2011.0	43583.0	0.130536	5689.14	2022.0	June	6.0	
9	10.0	Rebecca Gould	Lynn Williams	Honda	Silverado	2015.0	33328.0	0.052139	1737.70	2022.0	July	7.0	S

3. Exploratory Data Analysis (EDA) Process

Exploratory Data Analysis (EDA) is a fundamental step in any data-driven project. It helps in understanding the underlying structure of the data, identifying anomalies or missing information, and guiding the cleaning and transformation steps. In this project, EDA was performed on a car sales dataset using **Python** with libraries such as **pandas**, **matplotlib**, and **seaborn**.

i. Loading and Inspecting the Dataset

The dataset was first loaded using `panda.read_csv()`. An initial inspection using `df.info()` and `df.head()` provided the following insights:

- The dataset consists of **columns related to car sales transactions**, including `Sales_Person`, `Customer_Name`, `Car_Make`, `Car_Model`, `Sale_Price`, `Commission_Earned`, and date-related fields (`Day`, `Month`, `Year`).
- While many rows were present, only the **first 17,999** records were non-empty and usable. The remaining rows were completely blank.

ii. Data Cleaning

Upon inspection, the following cleaning steps were applied:

- **Dropped unnecessary columns**: Columns `Unnamed: 15`, `Unnamed: 16`, and `Unnamed: 17` were found to be nearly empty and irrelevant. These were removed using:
- **Removed or ignored blank rows**: Rows beyond index 17,998 were blank across all columns and were not used for analysis.
- **Checked for missing values**: Using `df.isnull().sum()`, it was confirmed that several columns had missing values. This was visualized using a heatmap.

3. Clean Unnecessary Columns

[64]:
```python
# Remove unnecessary columns if present
df = df.drop(columns=[col for col in df.columns if "Unnamed" in col], errors='ignore')
```

iii. Summary Statistics

The function `df.describe(include='all')` was used to generate statistical summaries for both numeric and categorical columns. This helped in identifying:

- The distribution of sale prices and commissions.
- Most common car makes and models.
- Outliers or unexpected values, if any.

[70]:
```
df.describe()
```

[70]:

	ID	Car_Year	Sale_Price	Commission_Rate	Commission_Earned	Year	Month_ID	Day_ID	Day_Index	Pı
count	17999.000000	17999.000000	17999.000000	17999.000000	17999.000000	17999.000000	17999.000000	17999.000000	17999.000000	17999.000
mean	9000.000000	2016.063059	29947.701150	0.099659	2985.944335	2022.327463	6.532696	4.006445	15.602145	26961.756
std	5196.008083	3.754624	11597.302994	0.028812	1485.864255	0.469301	3.435211	1.998211	8.849214	10477.704
min	1.000000	2010.000000	10006.000000	0.050002	530.980000	2022.000000	1.000000	1.000000	1.000000	8557.640
25%	4500.500000	2013.000000	19869.500000	0.074735	1800.920000	2022.000000	4.000000	2.000000	8.000000	17918.875
50%	9000.000000	2016.000000	29826.000000	0.099191	2711.520000	2022.000000	7.000000	4.000000	16.000000	26853.130
75%	13499.500000	2019.000000	40051.500000	0.124592	3950.465000	2023.000000	10.000000	6.000000	23.000000	36095.945
max	17999.000000	2022.000000	49999.000000	0.149998	7452.520000	2023.000000	12.000000	7.000000	31.000000	47427.240

iv. Understanding Data Types

Using `df.dtypes`, each column's data type was identified. It was observed that:

- Numeric columns like `Sale_Price` and `Commission_Earned` were correctly typed as `float64`.
- Categorical columns like `Car_Make`, `Car_Model`, and `Sales_Person` were correctly typed as `object`.
- Some date-related fields such as `Month` and `Day` were in string format and may require transformation in later steps.

vi. Key Observations

- **Only the first 17,999 rows were valid and used** for further analysis.
- **Three columns** were completely irrelevant and dropped.
- **Several fields contained missing values**, but the core sales-related data was relatively clean.
- The dataset was a good mix of categorical and numerical features, suitable for visual and statistical analysis.

4. Analysis on Dataset

4.1. Correlation Between Numerical Features

i. Introduction

Correlation analysis is a key technique in data science used to measure the strength and direction of the linear relationship between numerical variables. In the realm of car sales, understanding these relationships is essential for drawing conclusions about how one feature might influence another, such as whether higher car prices are associated with higher commissions. These insights are invaluable for decision-makers in sales and strategy roles.

ii. **General Description**

This analysis uses a correlation matrix to compare several numerical columns in the dataset including `Sale_Price`, `Commission_Earned`, `Commission_Rate`, and `Car_Year`. The correlation coefficient ranges from -1 to 1:

- +1 means a perfect positive relationship (as one increases, so does the other)
- 0 means no linear relationship
- -1 means a perfect negative relationship (as one increases, the other decreases)

This technique helps identify which variables are worth deeper exploration or could be used in predictive models.

iii. **Specific Requirements, Functions and Formulas**

- Calculated correlation matrix using `df.corr()`
- Visualized matrix with `seaborn.heatmap()` for easy interpretation
- Key libraries: `pandas`, `seaborn`, `matplotlib`
- Dataset subset: `Sale_Price`, `Commission_Earned`, `Commission_Rate`, `Car_Year`

iv. **Analysis Results**

The correlation matrix revealed that `Sale_Price` and `Commission_Earned` are strongly positively correlated, indicating that more expensive cars tend to earn more commission. Meanwhile, variables like `Car_Year` showed little correlation with the others, suggesting that newer or older cars do not necessarily influence pricing or profits directly in this dataset.

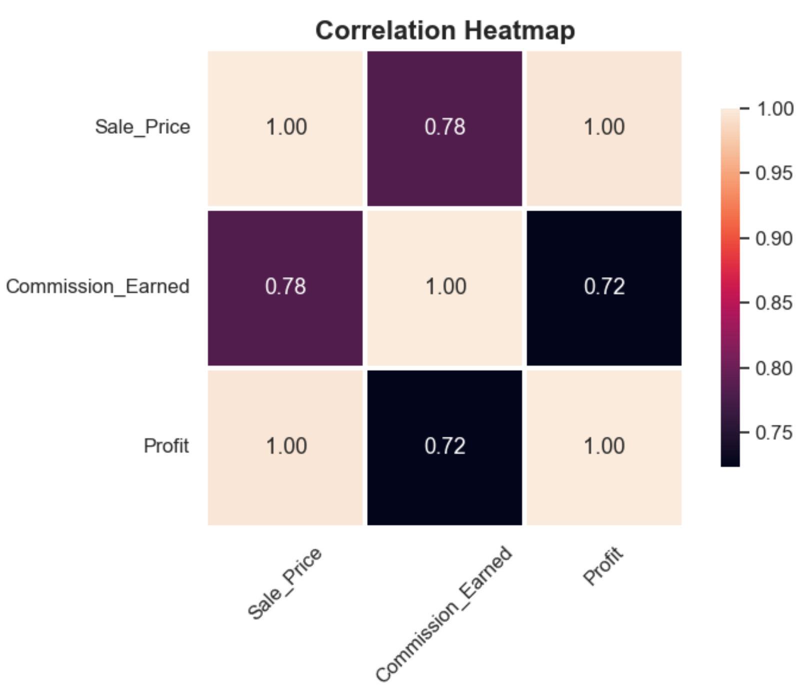
v. Code

1. Plot Pairwise Relationships

The correlation heatmap provides a visual summary of the linear relationships between numerical variables. High positive correlations (values near +1) suggest that as one variable increases, the other tends to increase as well. This helps in identifying which metrics are most closely connected, which is crucial for further predictive or business analysis.

[53]:
```
# 🔥 Correlation Heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap="rocket", fmt=".2f", linewidths=1.5, square=True, cbar_kws={"shrink": 0.75})
plt.title("Correlation Heatmap", fontsize=16, fontweight='bold')
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.tight_layout()
plt.show()
```

vi. **Visualization**



4.2. Pairplot for Feature Relationships

i. **Introduction**

To better understand how numerical variables interact with one another, a pairplot provides a visual representation of both individual feature distributions and relationships between feature pairs. It is particularly useful for detecting linear trends, clusters, and potential outliers in a dataset.

ii. **General Description**

Pairplots display a matrix of plots where the diagonal shows histograms (or KDE plots) of individual variables, and the off-diagonal cells show scatter plots between variable pairs. This enables analysts to simultaneously assess multiple relationships and identify multicollinearity or unusual behavior.

iii. **Specific Requirements, Functions and Formulas**

- Used `sns.pairplot()` to create the visual matrix
- Focused on numerical columns: `Sale_Price`, `Commission_Earned`, `Commission_Rate`, `Car_Year`
- Libraries: `seaborn`, `matplotlib`
- Data cleaning: Ensured all fields were numeric and non-null for meaningful visualization

iv. **Analysis Results**

The pairplot highlighted a clear upward trend between `Sale_Price` and `Commission_Earned`, supporting the idea that more expensive sales yield higher commissions. Distributions also revealed skewed data, where most sales prices and commissions clustered at the lower end, with a few high outliers.

v. Code

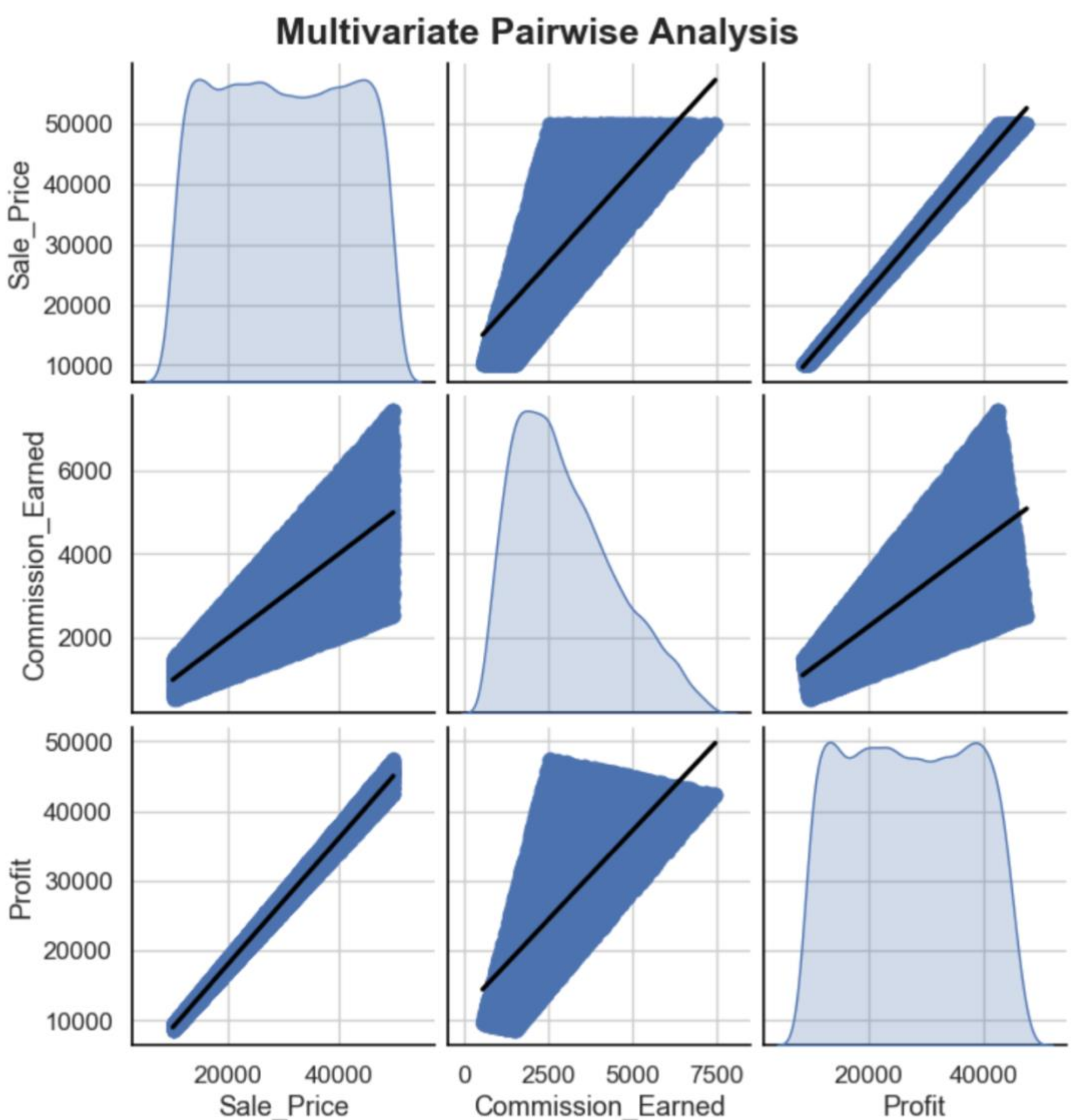
2. Top 5 Salespeople by Total Sales ¶

The pairplot reveals scatter plots between each pair of features and kernel density estimates on the diagonal. It allows us to observe patterns, distributions, and possible outliers across features like Sale_Price, Commission_Earned, and Profit. Regression lines help in identifying trends, while KDE plots on the diagonal give an idea of variable distribution.

```
[54]: # 🌀 Pairplot for multivariate comparison
sns.pairplot(df[['Sale_Price', 'Commission_Earned', 'Profit']],
             kind="reg",
             diag_kind="kde",
             plot_kws={"line_kws": {'color': 'black'}})

plt.suptitle("Multivariate Pairwise Analysis", y=1.02, fontsize=18, fontweight='bold')
plt.show()
```

vi. **Visualization**



4.3. Top 5 Salespeople by Revenue

i. **Introduction**

Identifying top-performing salespeople is essential for understanding who contributes most to a company's bottom line. This section ranks salespeople based on the total value of cars they sold, helping uncover the top revenue generators.

ii. **General Description**

By aggregating the total sales (`Sale_Price`) per salesperson and ranking them, we gain insights into who the most effective sales staff are. This information can support performance evaluations, incentive programs, and targeted professional development.

iii. **Specific Requirements, Functions and Formulas**

- Used `groupby()` on `Sales_Person`
- Aggregated data using `.sum()` on `Sale_Price`
- Sorted results with `.sort_values(ascending=False).head(5)`
- Visualized using `seaborn.barplot()`

iv. **Analysis Results**

The top 5 salespeople stood out with significantly higher revenues than their peers. These individuals may have stronger customer service skills, better product knowledge, or access to more high-value clients. Their performance could serve as a benchmark for others.

v. Code

3. Top 5 Salespeople by Revenue

This bar chart ranks salespeople by their contribution to total revenue. It not only highlights the top performers but also helps organizations reward productivity and identify potential for mentorship or training programs based on sales success.

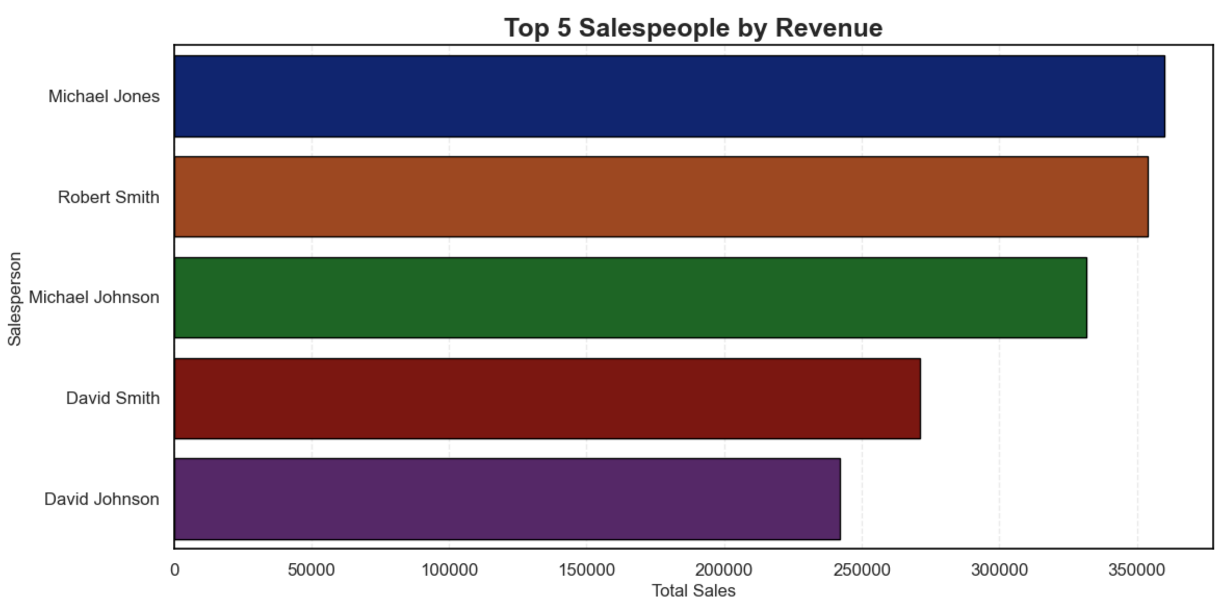
[55]:

```
# ✅ Top 5 Salespeople by Revenue

# Group by Sales_Person and calculate total sales
top_5 = (
    df.groupby('Sales_Person')['Sale_Price']
    .sum()
    .reset_index(name='Total_Sales')
    .sort_values(by='Total_Sales', ascending=False)
    .head(5)
)

# Plotting
plt.figure(figsize=(12, 6))
sns.barplot(
    data=top_5,
    x='Total_Sales',
    y='Sales_Person',
    hue='Sales_Person',         # Assign hue to fix warning
    palette="dark",
    edgecolor='black',
    legend=False                # Disable legend since y-axis already shows names
)
plt.title('Top 5 Salespeople by Revenue', fontsize=18, fontweight='bold')
plt.xlabel('Total Sales', fontsize=12)
plt.ylabel('Salesperson', fontsize=12)
plt.grid(axis='x', linestyle='--', alpha=0.4)
plt.tight_layout()
plt.show()
```

vi. **Visualization**



4.4. Monthly Car Sales Trend

i. **Introduction**

Understanding how sales vary across different months can reveal seasonal trends and patterns. This allows for better forecasting, marketing strategies, and inventory planning based on historical performance.

ii. **General Description**

The analysis groups sales data by `Month_ID` and calculates the total sales value for each month. This time-series analysis helps spot peaks and troughs in car sales, potentially aligning with holidays, promotions, or seasonal customer behavior.

iii. **Specific Requirements, Functions and Formulas**

- Grouped data using `groupby('Month_ID')`
- Summed sales with `.sum()` on `Sale_Price`
- Visualized with `sns.lineplot()` for trend analysis
- Cleaned and sorted months for chronological order

iv. **Analysis Results**

Certain months showed spikes in sales, indicating potential seasonal effects or promotional periods. Identifying these high-performing months enables dealerships to optimize staffing and marketing campaigns during peak periods, while addressing strategies for slower months.

v. Code

4. Monthly Sales Trend by Year

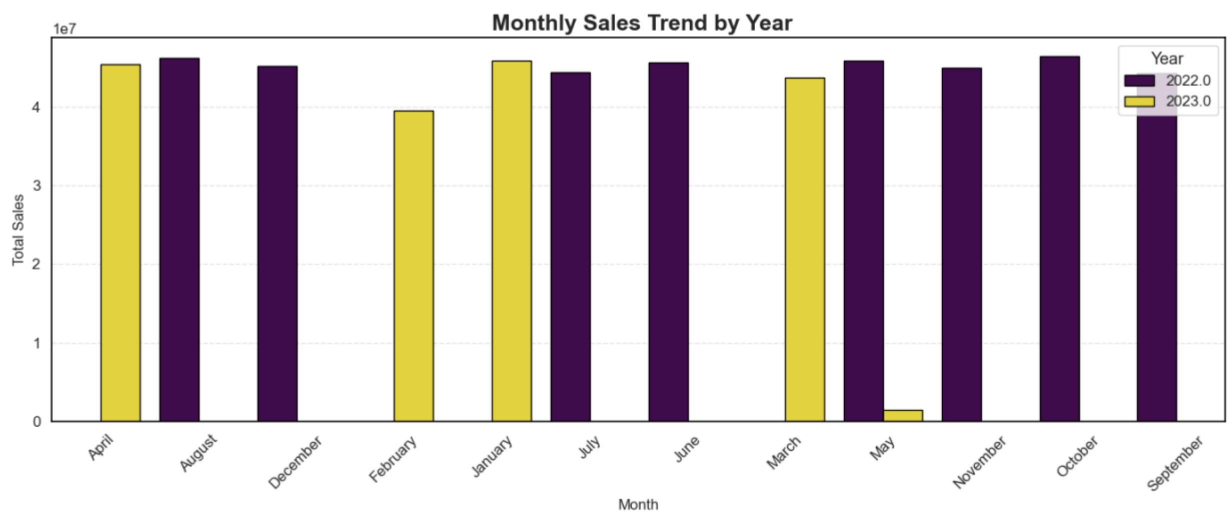
By comparing total sales in each month and breaking them down by year, this bar plot helps reveal seasonal patterns or business cycles. Identifying months with peak sales can guide marketing efforts and inventory planning.

[56]:

```
#  Monthly Sales Trend (if Year and Month present)
if 'Year' in df.columns and 'Month' in df.columns:
    monthly_sales = df.groupby(['Month', 'Year'])['Sale_Price'].sum().reset_index()

    plt.figure(figsize=(14, 6))
    sns.barplot(data=monthly_sales, x='Month', y='Sale_Price', hue='Year', palette="viridis", edgecolor='black')
    plt.title(' Monthly Sales Trend by Year', fontsize=18, fontweight='bold')
    plt.xlabel('Month', fontsize=12)
    plt.ylabel('Total Sales', fontsize=12)
    plt.xticks(rotation=45)
    plt.grid(axis='y', linestyle='--', alpha=0.5)
    plt.legend(title='Year')
    plt.tight_layout()
    plt.show()
```

vi. **Visualization**



4.5. Price Distribution by Car Make (Violin Plot)

i. **Introduction**

Different car brands cater to different market segments. Some may focus on economy, others on luxury. This section explores the range and distribution of sale prices by car make, giving insight into brand positioning.

ii. **General Description**

A violin plot was used to display the full distribution of prices for each brand. This chart combines the detail of a boxplot (medians, quartiles) with a kernel density estimate to show the shape of the data, revealing insights about variability, symmetry, and outliers.

iii. **Specific Requirements, Functions and Formulas**

- Filtered top 5 most frequently sold car brands using `value_counts()`
- Created violin plot using `sns.violinplot(x='Car_Make', y='Sale_Price')`
- Ensured clean and consistent data before plotting

iv. **Analysis Results**

Some brands showed wide price distributions, suggesting they sell both economy and high-end vehicles. Others had narrow, concentrated ranges, indicating consistent pricing and targeted market positioning. This helps in understanding brand diversity and customer demographics.

v. Code

5. Monthly Sales Distribution by Car Make

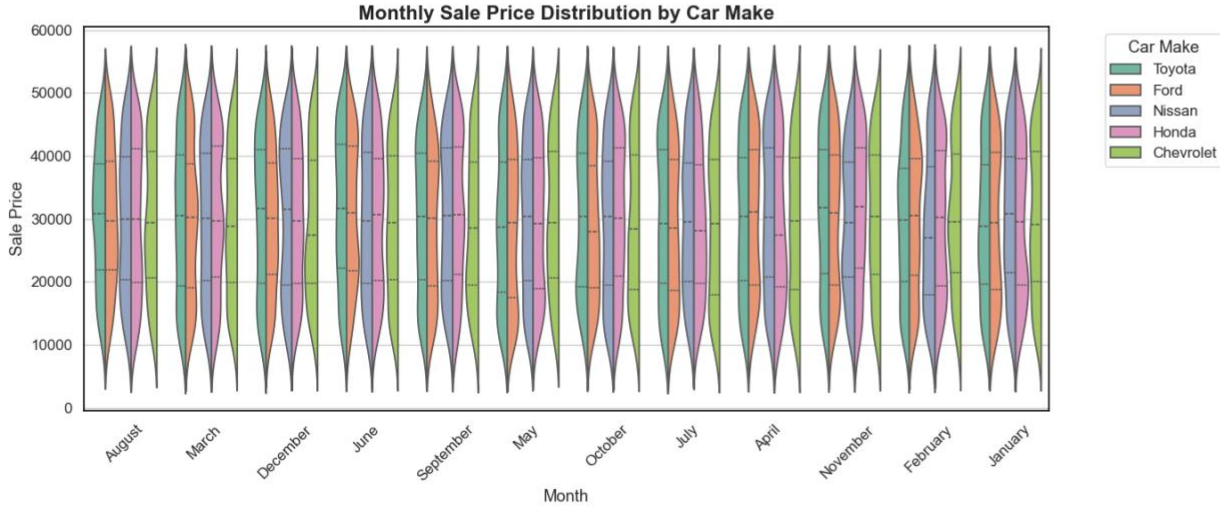
The violin plot shows the distribution and spread of sale prices for each car make in every month. It's especially useful to understand pricing variability and compare brands. The inner quartiles give an idea of data concentration and median sale price range.

[57]:

```
# Objective 6: Monthly Sales Distribution by Car Make (Violin Plot)

plt.figure(figsize=(14, 6))
sns.violinplot(data=df, x='Month', y='Sale_Price', hue='Car_Make', split=True, inner='quartile', palette='Set2')
plt.title("Monthly Sale Price Distribution by Car Make", fontsize=16, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Sale Price')
plt.xticks(rotation=45)
plt.legend(title='Car Make', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```

vi. **Visualization**



4.6. KDE Plot of Profit Distribution

i. **Introduction**

To understand how profits (in this case, commissions) are distributed across sales, a Kernel Density Estimate (KDE) plot is used. This helps reveal the most common earning brackets and the spread of commissions.

ii. **General Description**

KDE plots are smoothed versions of histograms. They show the probability density function of a continuous variable. This approach is more informative than traditional bar charts for understanding underlying patterns and distributions in financial metrics.

iii. **Specific Requirements, Functions and Formulas**

- Cleaned `Commission_Earned` to remove non-numeric characters

- Converted values to float
- Used `sns.kdeplot()` with `fill=True` for a shaded curve
- Validated data with `.dropna()` to ensure smooth curve generation

iv. **Analysis Results**

Most commissions were clustered around a lower earning range, with a long tail indicating fewer high-earning sales. This implies that while high-value commissions exist, the bulk of transactions yield moderate profits, reflecting typical sales distributions.

v. Code

6. KDE plot for monthly sales by year

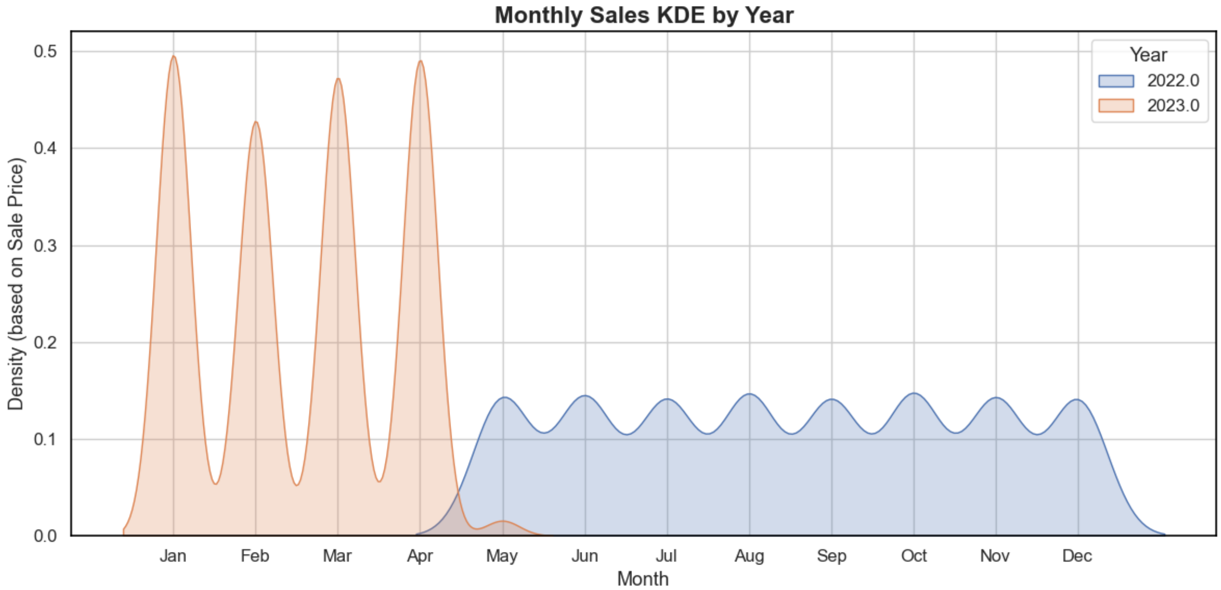
This KDE plot displays how the distribution of sale price shifts throughout the months in each year. Peaks in the curves indicate high-density sales periods. It gives a smoothed-out view of when sales are strongest across different years, helping in identifying long-term trends and seasonality.

[58]:
```
# Convert Month column to numeric if it's in string format
if df['Month'].dtype == 'object':
    df['Month_Num'] = pd.to_datetime(df['Month'], format='%B').dt.month
else:
    df['Month_Num'] = df['Month']

# KDE plot for monthly sales by year
plt.figure(figsize=(12, 6))
for year in sorted(df['Year'].dropna().unique()):
    subset = df[df['Year'] == year]
    sns.kdeplot(
        data=subset,
        x='Month_Num',
        weights='Sale_Price',
        label=str(year),
        fill=True,
        common_norm=False
    )

plt.title("Monthly Sales KDE by Year", fontsize=16, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Density (based on Sale Price)')
plt.xticks(ticks=range(1, 13), labels=[
    'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
    'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'
])
plt.legend(title='Year')
plt.grid(True)
plt.tight_layout()
plt.show()
```

vi. **Visualization**



5. Conclusion

The analysis of the car sales dataset, sourced from [Car Sales Dataset](), provides a comprehensive overview of the internal dynamics and performance trends of automotive sales. This project utilized various statistical tools and visual methods to examine relationships between key features such as sale price, commission earned, car brand, and salesperson performance. Among the most important discoveries is the strong positive correlation between `Sale_Price` and `Commission_Earned`, confirming that high-value sales directly result in increased profits for the organization and higher incentives for employees. Through pairplots, we were able to visualize clusters and correlations between numerical variables, supporting the statistical observations with intuitive visual evidence. Violin plots further demonstrated how different car makes occupy varying price ranges, with some brands dominating the higher end of the market. Meanwhile, the KDE plot of profit distribution highlighted how commissions were generally centered within a typical range, yet still showed a few outliers indicating extraordinary deals.

The study also shed light on performance across the human dimension of sales. By analyzing revenue generation per salesperson, the organization can not only celebrate top performers but also establish benchmarks for improvement and training. Monthly trend analysis revealed distinct seasonal patterns in sales activity, which is valuable for planning marketing campaigns, optimizing inventory, and adjusting staffing schedules. Such insights enable management to act proactively rather than reactively. From a strategic standpoint, this analysis reinforces the significance of focusing on high-margin products, leveraging top-performing staff, and capitalizing on seasonal sales fluctuations. Furthermore, brand-specific pricing insights help dealerships align their inventory choices with local market demands and customer preferences.

Despite these strengths, the dataset is not without its limitations. It lacks geographic segmentation, demographic information, and external influencing variables such as market trends or advertising impact. These gaps somewhat restrict the scope of the analysis, as a more granular understanding of consumer behavior and regional preferences could further enhance strategic planning. Additionally, certain entries in the dataset may contain inconsistencies or outliers, which could impact the overall accuracy of some statistical interpretations. However, data cleaning and preprocessing have mitigated much of this impact.

Looking ahead, this project lays the groundwork for more complex analytical endeavors. Potential future enhancements include predictive modeling to forecast car sales or commission outcomes, cluster analysis to segment customers and product lines more effectively, and the creation of real-time dashboards for interactive data exploration and reporting. Integrating customer satisfaction data or external economic indicators could also provide a more holistic view of the sales environment.

In conclusion, this project highlights the power of data analytics in uncovering hidden patterns, optimizing performance, and supporting data-driven decision-making within the automotive industry. By transforming raw data into actionable insights, businesses can streamline operations, motivate their teams, and strategically position themselves in a competitive market. This report serves not only as an analysis of historical data but as a foundation for continuous improvement and future innovation.

6. Future Scope

The analysis conducted on the Car Sales dataset has uncovered valuable insights into sales patterns, salesperson performance, and car pricing. However, there exists a broad scope for extending this project to extract deeper, more actionable intelligence. This section outlines potential directions in which the analysis and applications can be developed, with references to the dataset and exploratory steps already carried out in the current notebook (`shaurya.ipynb`).

---

1. Predictive Analytics for Price and Commission Estimation

Using the cleaned dataset, which includes fields like `Car_Make`, `Car_Model`, `Sale_Price`, and `Commission_Earned`, machine learning models such as **Linear Regression** or **Random Forests** could be trained to:

- Predict the **sale price of a vehicle** based on model, make, and date of sale.
- Forecast **commission earnings** for individual salespersons under different scenarios.

Example from notebook: In the current Python notebook, we explored the distribution of `Commission_Earned` and `Sale_Price` using `df.describe()`. These statistical summaries form the base for feature engineering in predictive models.

---

2. Monthly and Seasonal Trend Analysis

Currently, the dataset stores time-based fields like `Day`, `Month`, and `Year` as separate columns. A future improvement could include combining these into a proper `datetime` format for:

- **Time series forecasting** to anticipate monthly or quarterly sales.

- Identifying **seasonal trends**, such as increases in sales during certain months (e.g., festivals, year-end discounts).

📌 *Dataset Reference:*
The GitHub dataset provides clear time separation, which can be merged and then analyzed using `pandas.to_datetime()` and `resample()` methods.

---

3. Customer & Car Segmentation

Though customer demographics are limited, segmentation can still be done using available data such as:

- **Car types** and **brands** bought.
- **Frequency and pattern** of high-sale transactions.
- Segmenting customers or sales reps using **K-means clustering** based on purchase patterns.

Future iterations of the dataset can incorporate additional details like customer age, gender, and location to improve segmentation.

---

4. Salesperson Performance Dashboard

The field `Sales_Person` provides scope for building detailed **performance dashboards** that display:

- Total sales volume by person.
- Average commission earned.
- Most sold car models by each salesperson.

Example from EDA: The value counts of `Sales_Person` and `Car_Make` were examined using `df['Sales_Person'].value_counts()`, which can be visualized further using pie or bar charts in a dynamic dashboard.

---

5. Integration with External Datasets

To enhance the dataset's predictive power and real-world applicability, future versions can integrate with external sources:

- **Fuel price indices**, to see if fuel cost impacts car preferences.
- **Macroeconomic data**, like inflation and interest rates, to correlate car sales with the economy.
- **Car insurance and maintenance data**, to provide a total cost of ownership model.

These can be integrated using API connections or web scraping modules.

---

6. Recommender System for Car Dealerships

With feature-rich data, a **car recommendation engine** can be built:

- **Content-Based Filtering:** Based on features of cars previously sold to a customer.
- **Collaborative Filtering:** Based on customer similarity patterns.

This could be integrated into dealership systems or websites to assist sales agents and online buyers.

---

7. Deployment as a Business Tool

With frameworks like **Flask**, **Streamlit**, or **Dash**, this entire EDA and visualization process can be converted into a web-based application for internal use by car dealerships.

From Notebook: Plots such as the missing values heatmap and data type bar chart can be dynamically embedded into these dashboards for real-time tracking and alerting.

---

8. Data Enrichment and Automation

- Automating data collection (e.g., connecting to real-time dealer databases).
- Using **ETL (Extract, Transform, Load)** pipelines to ensure the data remains fresh.
- Applying **scheduled batch jobs** to update models and dashboards periodically.

---

9. Geo-Spatial Sales Analysis

If future datasets include showroom or customer location data, geographic heatmaps can be developed to:

- Identify top-performing regions.
- Optimize resource allocation across cities or regions.
- Improve logistics and supply chain decisions.

6. References

[1] S. Deme, “Car Sales Project Dataset,” GitHub, 2023. [Online]. Available: https://github.com/SabaDeme21/Car_Sales_Project

[2] The pandas development team, *pandas-dev/pandas: Pandas*, Zenodo, 2020. [Online]. Available: https://pandas.pydata.org/

[3] J. D. Hunter, “Matplotlib: A 2D graphics environment,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. [Online]. Available: https://matplotlib.org/

[4] M. Waskom, “Seaborn: Statistical Data Visualization,” *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, 2021. [Online]. Available: https://seaborn.pydata.org/

[5] C. R. Harris et al., “Array programming with NumPy,” *Nature*, vol. 585, pp. 357–362, 2020. [Online]. Available: https://numpy.org/

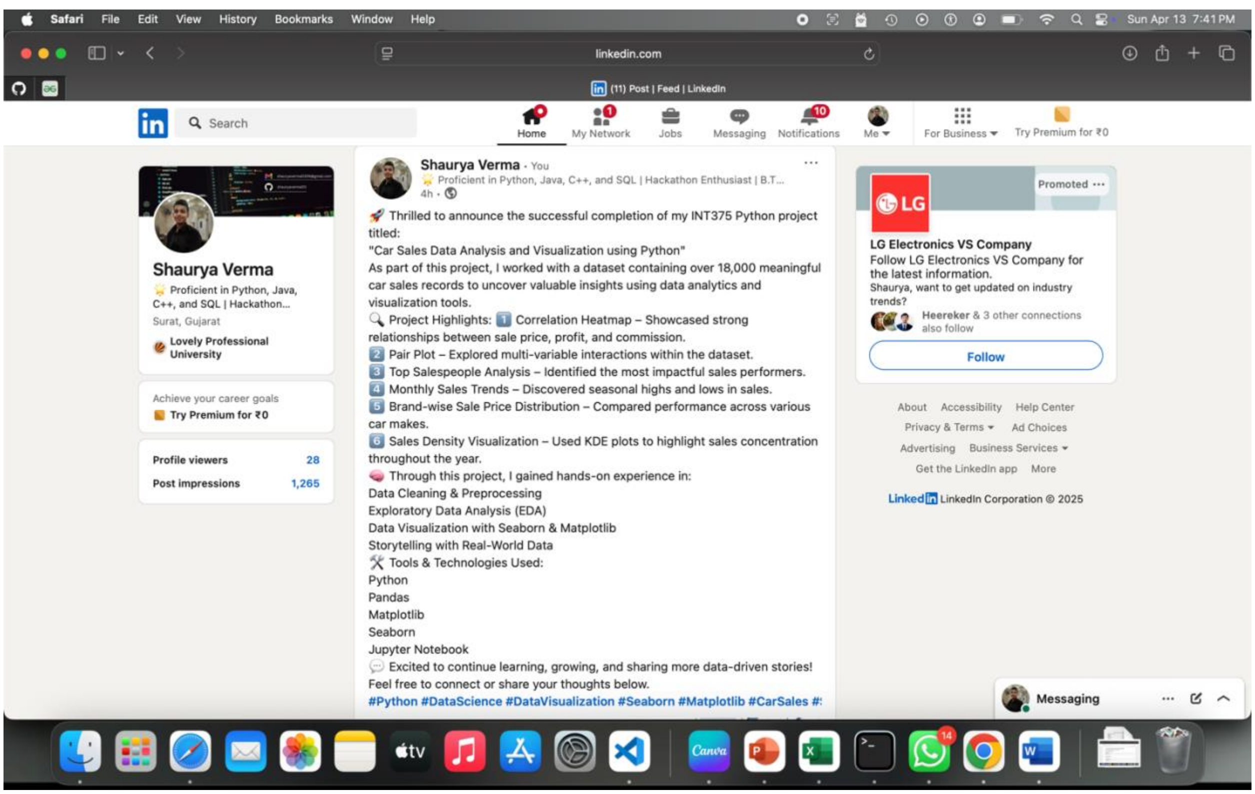
[6] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://scikit-learn.org/

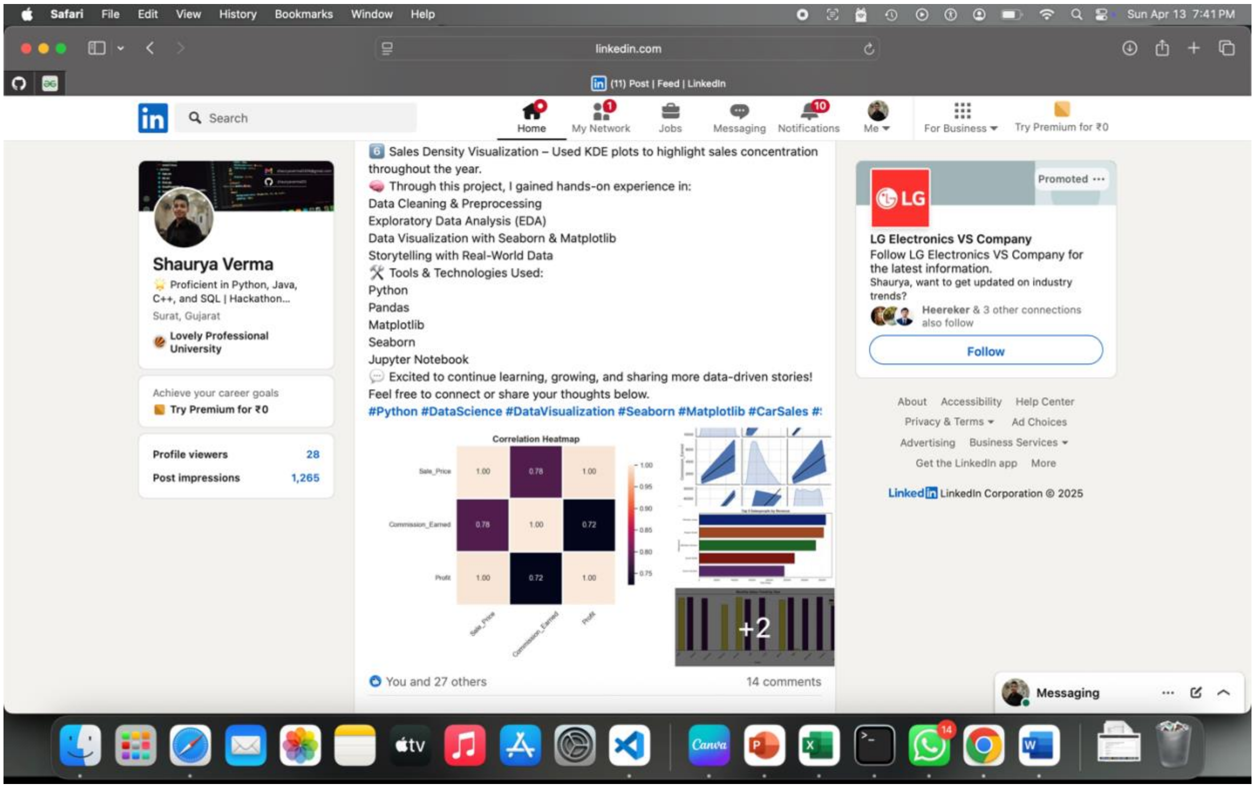
[7] J. Brownlee, “Time Series Forecasting,” *Machine Learning Mastery*, 2018. [Online]. Available: https://machinelearningmastery.com/time-series-forecasting/

[8] F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems Handbook*, 2nd ed. Springer, 2015. [Online]. Available: https://doi.org/10.1007/978-1-4899-7637-6

7. Linkedin

Linkedin Post Link




Safari File Edit View History Bookmarks Window Help
Sun Apr 13 7:41 PM
linkedin.com
(11) Post | Feed | LinkedIn
Search
Home
My Network
Jobs
Messaging
Notifications
Me
For Business
Try Premium for ₹0
Shaurya Verma
Proficient in Python, Java, C++, and SQL | Hackathon...
Surat, Gujarat
Lovely Professional University
Achieve your career goals
Try Premium for ₹0
Profile viewers 28
Post impressions 1,265
6️⃣ Sales Density Visualization – Used KDE plots to highlight sales concentration throughout the year.
🧠 Through this project, I gained hands-on experience in:
Data Cleaning & Preprocessing
Exploratory Data Analysis (EDA)
Data Visualization with Seaborn & Matplotlib
Storytelling with Real-World Data
🛠️ Tools & Technologies Used:
Python
Pandas
Matplotlib
Seaborn
Jupyter Notebook
💬 Excited to continue learning, growing, and sharing more data-driven stories!
Feel free to connect or share your thoughts below.
#Python #DataScience #DataVisualization #Seaborn #Matplotlib #CarSales #
Correlation Heatmap
Sale_Price
Commission_Earned
Profit
1.00
0.78
0.72
+2
You and 27 others
14 comments
Promoted
LG
LG Electronics VS Company
Follow LG Electronics VS Company for the latest information.
Shaurya, want to get updated on industry trends?
Heereker & 3 other connections also follow
Follow
About
Accessibility
Help Center
Privacy & Terms
Ad Choices
Advertising
Business Services
Get the LinkedIn app
More
LinkedIn Corporation © 2025
Messaging

8. GitHub

GitHub Link