Jaypee Institute of Information Technology, Noida

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
AND INFORMATION TECHNOLOGY

Project Title: Spinny 2.0

Enrol. No.	Name of Student
23103092	Ansh Kumar
23103097	Mohammad Kashif
23103117	Hiritik Dhir

Course Name: Open Source Software Lab

Course Code: 15B17CI575

Program: B. Tech. CSE

3rd Year 5th Sem

Batch – B4

Submitted To : Dr. Alka Singhal

2025

ABSTRACT

The purpose of this lab project is on creating a car recognition application that utilizes machine learning models and software tools, such as Streamlit, for estimating the market value of second-hand cars. The app employs dependency management and a range of Python packages to enhance functionality and interface, ensuring accurate, user-friendly price predictions. Key packages like Altair, Blinker, and Cache tools support data handling, visualization, and interaction. This setup allows users to easily access car price predictions, aiding in informed purchasing decisions within the second-hand automotive market.

INTRODUCTION

In recent years, the demand for accurate valuation tools within the second-hand car market has surged, as buyers and sellers seek informed guidance on fair pricing. This project aims to create a car recognition application that leverages machine learning to estimate the market value of used cars. Built using Python and incorporating tools like Streamlit, the app provides a user-friendly interface for seamless interaction and price predictions.

To achieve high functionality, the project employs effective dependency management alongside a variety of Python packages. Key packages, such as Altair for data visualization, Blinker for event-driven programming, and Cachetools for efficient data handling, enhance the app's ability to manage, visualize, and process data. These tools ensure that the application is not only accurate in its predictions but also intuitive for users, enabling them to access critical insights with ease. Ultimately, this app serves as a valuable resource, assisting users in making informed, data-driven purchasing decisions within the pre-owned car market.

Technologies Used in the Project

1. **Pandas**:

   Purpose:

   - Pandas is a powerful data manipulation and analysis library in Python. It is used for cleaning, transforming, and analyzing data in tabular form (DataFrame). It helps in handling missing values, filtering rows, grouping data, and performing calculations.

   Key Functions:

   - pd.read_csv(), pd.DataFrame(), df.dropna(), df.fillna(), df.groupby(), etc.

2. **NumPy**:

   Purpose:

   - NumPy is a library for numerical computations in Python. It is used for handling arrays and performing mathematical operations on large datasets. It is especially useful for operations on multi-dimensional arrays.

   Key Functions:

   - np.array(), np.mean(), np.median(), np.corrcoef(), etc.

3. **Matplotlib**:

   Purpose:

   - Matplotlib is a plotting library for Python that is used to create static, interactive, and animated visualizations. It helps in creating a variety of charts such as line graphs, scatter plots, histograms, etc.

   Key Functions:

   - plt.plot(), plt.scatter(), plt.hist(), plt.show(), etc.

   Features:

   - Customization options for colors, styles, and other plot attributes.

4. **Seaborn**:

Purpose:

- Seaborn is built on top of Matplotlib and provides a high-level interface for creating attractive and informative statistical graphics. It is often used for visualizing relationships between variables, distributions, and statistical trends.

Key Functions:

- sns.barplot(), sns.heatmap(), sns.boxplot(), sns.pairplot(), etc.

Features:

- Easy-to-use interface for complex plots, statistical visualizations, and multi-plot layouts.

5. **Matplotlib Inline**:

Purpose:

- The %matplotlib inline magic command is used in Jupyter Notebooks to display Matplotlib plots directly in the notebook. It ensures that visualizations appear in the output cell immediately after execution, rather than in a separate window.

6. **Warnings**:

Purpose:

- The warnings module is used to manage warnings in Python. In this case, the code uses warnings.filterwarnings('ignore') to suppress warnings during the execution, which is helpful in avoiding cluttered output.

7. **Linear Regression (Machine Learning)**:

Purpose:

- Linear regression is a statistical method used for modeling the relationship between a dependent variable (in this case, car price) and one or more independent variables (such as car features like mileage, brand, age, etc.). It helps in predicting the price of a car based on these parameters.

Key Libraries:

- **Scikit-learn**: Although not explicitly mentioned in the imports, scikit-learn is commonly used for machine learning tasks like linear regression.

- **Key Functions**: • LinearRegression(), model.fit(), model.predict(), etc.

8. **Streamlit (User Interface):**

Purpose:

- In this project, Streamlit provides a simple web interface where users can input car features (name, brand, fuel, etc.) and instantly see the predicted car price from the trained linear regression model.

Key Libraries:

- **Streamlit:** Used to create the web UI components such as text inputs, sliders, select boxes, and buttons for user interaction.

- **Key Functions**: st.title(), st.header(), st.write() for adding titles, headings, and text in the app.

Flow of the Project

1. **Data Collection and Cleaning**:

- Loading the dataset using pandas.read_csv() or any other suitable function.
- Data cleaning, such as handling missing values (df.dropna() or df.fillna()), and preparing the data for analysis (e.g., encoding categorical variables, scaling numerical variables).

2. **Data Visualization**:

- Using **Matplotlib** and **Seaborn** to plot visualizations like scatter plots, histograms, and heatmaps to understand patterns and relationships between different features in the dataset.
- Example: Visualizing the correlation between car price and other features like mileage, year, brand, etc. •

3. **Feature Engineering**:

- Selecting relevant features (independent variables) for the linear regression model, such as car age, mileage, brand, etc.
- Handling categorical variables (e.g., converting 'brand' into numerical data).

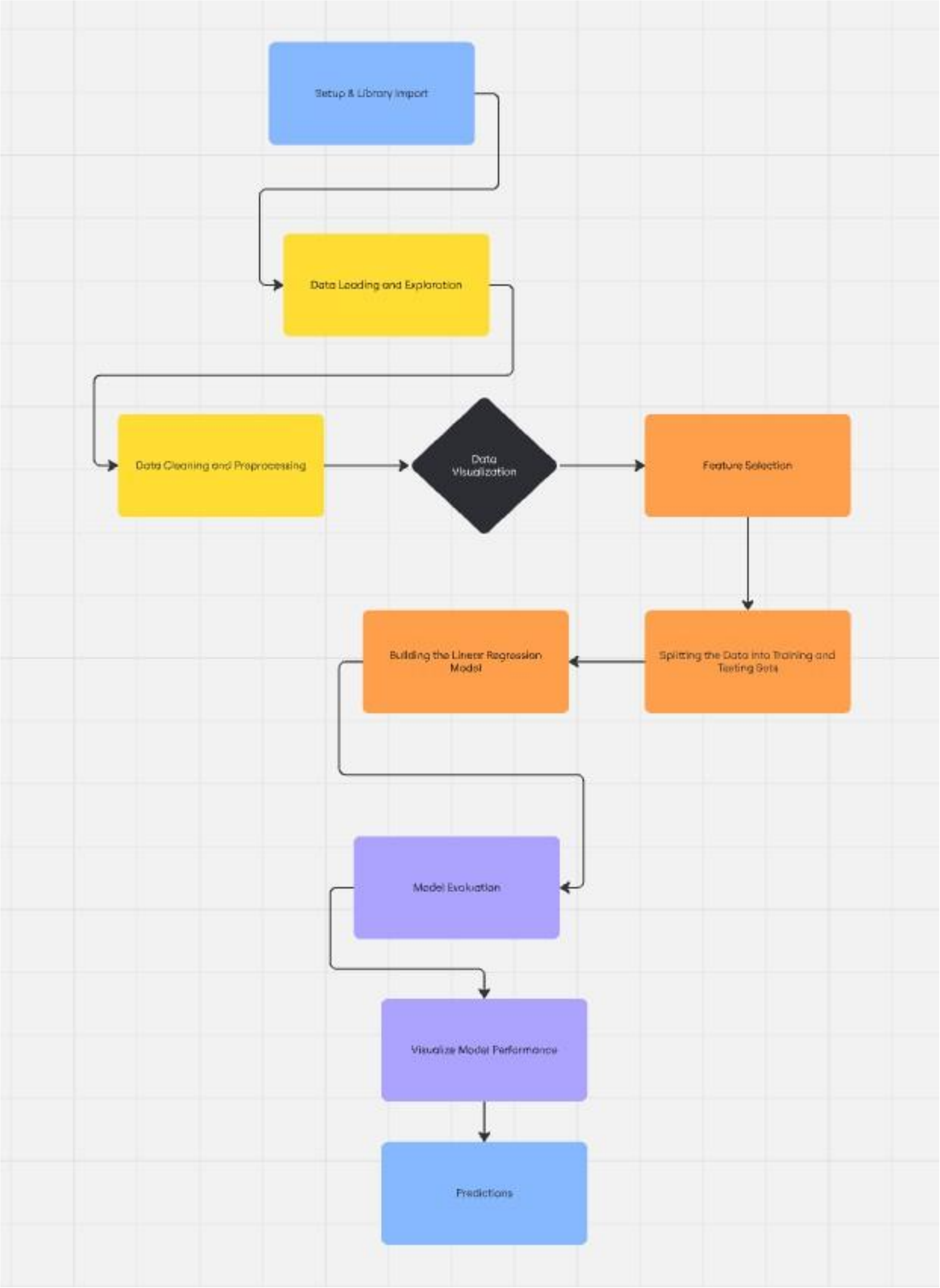
4. **Linear Regression Model**:

- Using **Linear Regression** to build a predictive model for car prices. This involves:
- Splitting the dataset into training and testing sets.
- Training the model using model.fit(X_train, y_train).
- Making predictions using model.predict(X_test).
- Evaluating the model performance using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), R-squared, etc.

5. Prediction:

- Using the trained linear regression model to predict car prices based on input feature

Flowchart for the implementation of the code



Setup & Library Import
Data Loading and Exploration
Data Cleaning and Preprocessing
Data Visualization
Feature Selection
Splitting the Data into Training and Testing Sets
Building the Linear Regression Model
Model Evaluation
Visualize Model Performance
Predictions

Conclusion

In this project, we successfully developed a systematic approach to install and verify Python packages along with their dependencies, ensuring a smooth and error-free setup process. By implementing a structured flow for installing packages like Streamlit, Altair, Cachetools, Git Python, and others, the project enhances the reliability of the environment setup, reduces manual errors, and saves time. This flowchart-based design simplifies troubleshooting, providing clarity on each step in the installation process. Overall, this project improves the efficiency of dependency management, which is crucial for robust Python development environments.

Future Enhancements

1. **Automated Dependency Resolution**:
Integrate an intelligent system that automatically resolves conflicts between package versions. For instance, if a new package requires a different version of a dependency, the system could recommend or automatically apply compatible versions.

2. **Use Advanced Machine Learning Models**
Experiment with more powerful algorithms such as Random Forest, XGBoost, and Gradient Boosting to capture non-linear relationships and improve prediction accuracy compared to simple linear regression.

3. **Automated Testing of Dependencies**:
Implement a testing suite that verifies the functionality of each installed package after setup, ensuring that all dependencies are correctly configured and the packages work as intended.