Contents

Synopsis

Project Title: Data Analytics with Stock Market Data

Introduction:

Data Analytics with Stock Market Data is a crucial aspect of the automotive industry that helps stakeholders make informed decisions. With the increasing demand for stocks-efficient vehicles globally, it has become crucial to understand the dynamics of Stock Market Data toremain competitive.

Problem Statement:

The problem with Stock Market Data analysis is that it can be complex and time-consuming. The datasets are often large and unstructured, making it difficult to extract meaningful insights. Therefore, there is a need for an effective methodology that can simplify the process and provide accurate results.

Literature Survey:

Previous studies have explored different techniques for Stock Market Data analysis. Some researchers have used machine learning algorithms such as regression analysis and decision trees to predict stock markets. Others have used statistical methods like descriptive statistics and correlation analysis to understand the relationship between different variables and stock prediction. However, there is still a need for more research in this field, especially with the increasing amount of data available.

Objective of the Project Work:

The primary objective of this project is to develop a comprehensive data analysis methodology that can handle large and unstructured Stock Market Datasets to provide insights into the dynamics of stock prediction.

To achieve this objective, the proposed methodology will involve several steps. The first step will be data preprocessing, which involves cleaning and transforming the raw data into a structured format that can be analyzed. This step will ensure that the data is of high quality and free of errors, which is essential for accurate analysis.

The second step will be exploratory data analysis, which involves using graphical and statistical techniques to identify patterns, trends, and outliers in the data. This step will help to uncover any hidden relationships between variables and provide insights into the dynamics of stock prediction.

The third step will be feature selection, which involves identifying the most important variables that influence stock prediction. This step will help to determine which variableshave the most significant impact on stock prediction and how they are related.

Lastly, data visualization will be performed using Python programming and data analysis libraries like Matplotlib to create visualizations of the Stock Market Data. These visualizations will help to identify trends and patterns in the data and present the results in an easy-to-understand format.

By developing a data analysis methodology that can handle unstructured Stock Market Data and provide insights into stock prediction dynamics, this project will enable stakeholders in the automotive industry to make informed decisions.

Proposed Methodology:

The proposed methodology involves data preprocessing, exploratory data analysis, feature selection, and data visualization using Python programming and data analysis libraries like Pandas, NumPy, and Matplotlib. The data preprocessing step involves cleaning and transforming the raw Stock Market Data into a structured format that can be used for analysis. The exploratory data analysis step involves using graphical and statistical techniques to identify patterns, trends, and outliers in the data. The feature selection step involves identifying the most important variables that influence stock prediction. Finally, data visualization is performed to create visualizations of the Stock Market Data using Matplotlib.

Summary:

In summary, this project aims to develop an effective methodology for Stock Market Data analysis using Python programming and data analysis libraries. By predicting stock predictionaccurately, identifying the key drivers of stock prediction, and providing insights that can inform business decisions, the methodology can help stakeholders in the automotive industry make informed decisions.

1. Introduction

Welcome to the exciting world of data analytics, where we use data to uncover insights that can transform the way we do business. In this particular case, we will be focusing on analyzing Stock Market Data, an important aspect of the automotive industry.

The demand for stock market has increased significantly in recent years due to rising stock costs and environmental concerns. As a result, understanding Stock Market Data is crucial for car manufacturers, dealers, and policymakers to make informed decisions. In this project, we will use Python for data visualization to analyze Stock Market Data and uncover insights that can inform business decisions. We will explore the relationships between different variables, such as engine size, transmission type, and stocks efficiency, using graphical and statistical techniques.

The primary objective of this project is to develop a comprehensive data analysis methodology that can handle large and unstructured Stock Market Datasets to provide insights into the dynamics of stock efficiency. By identifying the key factors that influence stocks efficiency, making accurate forecasts, and providing useful information for decision-making, this methodology can help manufacturers, dealers, investors, and policymakers develop effective strategies to increase stocks efficiency and remain competitive.

In conclusion, data analysis of Stock Market Data using Python for data visualization is an essential tool for businesses and policymakers in the automotive industry. By leveraging the insights gained from this analysis, they can make data-driven decisions that can help them stay ahead of the competition, maximize their profitability, and contribute to a sustainable future.

2. Requirement analysis

Requirement analysis is a crucial step in the development of any project, including the Data Analytics with Stock Market Data project. The goal of requirement analysis is to identify the specific goals, objectives, and requirements of the project to ensure that it can be developed in a way that meets these needs. Here are some of the key requirements that must be identified for this project:

Data Sources:

The first requirement for the Data Analytics with Stock Market Data project is to identify the data sources that will be used for the analysis. These may include Stock Market Data from the EPA's website, vehicle make and model data, and other relevant sources.

Data Quality:

The quality of the data used for the analysis is critical to the success of the project. Therefore, the requirement analysis should identify the quality of the data sources and any limitations or issues that may impact the analysis.

Goals and Objectives:

The requirement analysis should identify the goals and objectives of the Data Analytics with Stock Market Data project. These may include identifying the most stock prices, analyzing the relationship between stocks efficiency and vehicle characteristics, and other related goals.

Key Performance Indicators:

The requirement analysis should also identify the key performance indicators (KPIs) that will be used to measure the success of the project. These may include metrics companies, and other related KPIs.

Analysis Techniques:

The requirement analysis should identify the analysis techniques that will be used for the project, such as data visualization using Python libraries like Matplotlib and Seaborn.

Reporting Requirements:

The requirement analysis should identify the reporting requirements for the project. This may include the frequency of reporting, the format of the reports, and the specific information that should be included in the reports.

Timeline and Budget: Finally, the requirement analysis should identify the timeline and budget for the project. This will help to ensure that the project is completed on time and within budget.

In conclusion, requirement analysis is a critical process for the Data Analytics with Stock Market Data project. By identifying the specific goals, objectives, and requirements ofthe project, businesses can ensure that the project is developed in a way that meets their needs and delivers the desired results.

3. Software requirement specification

The software requirements for a data analytics project with Stock Market Data using Pythonfor data visualization will depend on the specific goals and objectives of the project, as well as the analysis techniques that will be used. However, here are some of the key software requirements that are typically needed for such a project:

Data Collection and Storage:
To perform a data analysis project with Stock Market Data, businesses need to collect and store large amounts of data. Therefore, they will need software tools for data collection, storage, and management, such as SQL databases, cloud storage services, or data warehouses.

Data Cleaning and Pre-processing:
Before analysis can begin, the data must be cleaned and pre-processed to remove any errors, inconsistencies, or duplicates. Python libraries such as Pandas, NumPy, and Scikit-learn can be used for data cleaning and pre-processing.

Data Visualization:
Data visualization tools play a crucial role in a data analysis project as they enable businesses to present the results of their analysis in a visually appealing and easy-to-understand manner. Python libraries such as Matplotlib, Seaborn, and Plotly can be used for data visualization.

Statistical Analysis:
Statistical analysis involves applying mathematical and statistical techniques to Stock Market Data to uncover meaningful patterns, trends, and relationships. Python libraries such as Scikit-learn, Statsmodels, and SciPy can be used for statistical analysis.

Machine Learning:
Machine learning techniques can be used to build predictive models that can help businesses understand and forecast stock prediction trends. Python libraries such as Scikit-learn, TensorFlow, and Keras can be used for machine learning.

Collaboration and Version Control:

Collaboration and version control are essential for any data analytics project to ensure that team members can work together efficiently and keep track of changes. Tools such as GitHub, GitLab, and Bitbucket can be used for collaboration and version control.

In conclusion, the software requirements for a data analytics project with Stock Market Data using Python for data visualization will depend on the specific needs and goals of the project. However, having the right software tools for data collection, storage, cleaning, visualization, statistical analysis, machine learning, and collaboration is essential for a successful data analytics project.

Users are used to this project they have to make installation of the Twitter Sentiment System on their system which is developed by using the technologies needed and this technologies software implementation is hidden in form of the service they used.

1) User Requirements

Users who are used to this project have to make installation of the following softwares:

- Backend Technology: Python (NumPy, Pandas, Matplotlib, Seaborn)
- IDE: Jupiter notebook.
- Software set-up installation

2) Software Requirement

- Backend Technology: Python (Pandas, Matplotlib)
- IDE: Jupiter notebook.
- Data Set (Stock prediction)

4. Analysis and design

The analysis and design of a "Data Analytics with Stock Market Data" project using python for data visualization involves several steps to ensure that the data is accurately analyzed and interpreted. Here are the main steps involved in the analysis and design of this project:

Data Collection:

The first step in any data analysis project is to collect relevant data. In the case of the "Data Analytics with Stock Market Data" project, businesses need to collect Stock Market Data from various sources, such as the Environmental Protection Agency (EPA) and other public sources. The data collected should be relevant to the project's goals and objectives.

Data Cleaning:

Once the data is collected, it needs to be cleaned and organized to ensure that it is accurate and consistent. This involves removing duplicates, filling in missing values, and correcting any errors in the data.

Data Transformation:

The data collected may be in different formats or structures, so it needs to be transformed into a consistent format to enable analysis. This involves converting data into a common format, such as CSV or Excel, and creating a data dictionary to document the data types and formats.

Data Analysis:

After the data is cleaned and transformed, the project can start analyzing the data. This involves using statistical techniques, such as regression analysis, correlation analysis, and time-series analysis, to identify patterns and relationships in the data. The analysis should be performed in a way that is relevant to the project's goals and objectives.

Data Visualization:

Data visualization is an essential component of any data analysis project. It involves creating charts, graphs, and other visual aids to help stakeholders understand the insights and findings from the data analysis. In the case of the "Data Analytics with Stock Market Data" project, python libraries such as Matplotlib, Seaborn, and Plotly can be used to create visualizations.

Reporting:
The final step in the analysis and design of the "Data Analytics with Stock Market Data" project is reporting. Businesses need to document their findings and insights in a report that can be shared with stakeholders. The report should be clear, concise, and relevant to the project's goals and objectives.

In conclusion, the analysis and design of the "Data Analytics with Stock Market Data" project using python for data visualization involve several steps, including data collection, cleaning, transformation, analysis, visualization, and reporting. By following these steps, businesses can gain valuable insights into stock prediction performance and make data-drivendecisions to improve their operations.

5. Data understanding

Data understanding is a crucial step in any data analytics project, including the analysis of Stock Market Data using Python for data visualization. Data understanding involves exploring and analyzing the data to gain a comprehensive understanding of the variables, data types, and patterns present in the data. Here are the main steps involved in data understanding for a data analytics project using Stock Market Data:

Data Collection:

The first step in data understanding is to collect all relevant data from various sources such as government websites and vehicle manufacturers. This data should include information on stock prediction ratings, vehicle make and model, engine size, and transmission type.

Problem that we are trying to solve:

1. Are more models using alternative sources of stocks? By how much?
2. How much have vehicle classes improved in stock prediction?
3. What are the characteristics of SmartWay vehicles?
4. What features are associated with better stock prediction?
5. For all of the models that were produced in 2008 that are still being produced in 2018, how much has the mpg improved and which vehicle improved the most?

Data Exploration:

Once the data is collected, it needs to be explored to understand the different variables and data types present in the data. This involves performing descriptive statistics such as mean, median, and standard deviation to understand the distribution of the data.

Data Visualization with Python:

Visualization is an essential component of data understanding as it enables businesses to identify patterns and trends that may not be immediately apparent from the raw data. Python offers several powerful libraries such as Matplotlib and Seaborn to create various charts and graphs such as scatter plots, histograms, and box plots to explore the data visually.

Data Cleaning:

Data cleaning is an important step in data understanding as it involves removing any outliers or errors in the data. This step is essential to ensure that the data is accurate and consistent and that any insights derived from the data are reliable.

Feature Engineering:

Feature engineering involves creating new features from the existing data to improve the performance of the analysis. For example, businesses can create new features such as stock prediction trends over time or stock prediction by vehicle class.

Data Preparation:

Once the data is cleaned and feature engineered, it needs to be prepared for analysis. This involves transforming the data into a format suitable for analysis, such as converting categorical data to numerical data. By following these steps and using Python for data visualization, businesses can gain a comprehensive understanding of their Stock Market Data, identify patterns and trends, and prepare the data for analysis. This understanding enables businesses to make data-driven decisions to improve their stock prediction performance.

6. Implementation and results

To carry out data analysis for the Stock Market Data, a range of techniques must be applied to the dataset to address the research questions identified in the requirement analysis stage. This involves using Python's data analysis and visualization libraries, such as Pandas, NumPy, and Matplotlib, to manipulate and explore the data and generate visualizations and statistical models. Through these techniques, we can gain insights into stock prediction trends, identify factors that influence stocks efficiency, and evaluate the effectiveness of government policies aimed at reducing stocks consumption and emissions. Overall, the process of data analysis for the Stock Market Dataset involves applying a range of techniques and tools to the data in order to gain insights and make data-driven decisions.

6.1 Data Acquisition

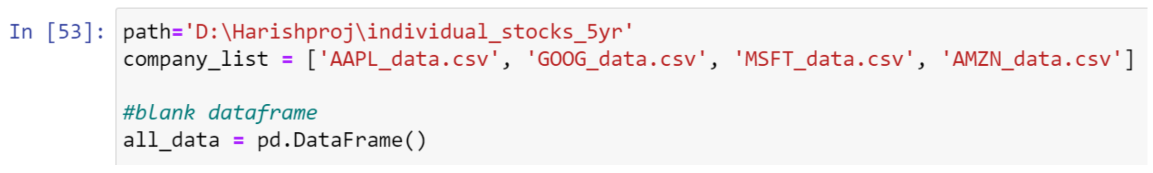
a) This information is provided by the U.S. Environmental Protection Agency, Office of Mobile Sources, National Vehicle and Stocks Emissions Laboratory.

Data acquisition is a critical step in the "Data Analytics with Stock Market Data" project, as it involves obtaining the necessary raw data for analysis. In the context of the Stock Market Dataset, the raw data can be obtained from the U.S. Department of Energy's website, which provides access to the data in various file formats, including CSV and Excel.

Once the dataset has been downloaded, it can be imported into Python using the Pandas library. Pandas provides a range of functions that can be used to manipulate and explore the data, as well as generate visualizations and statistical models.

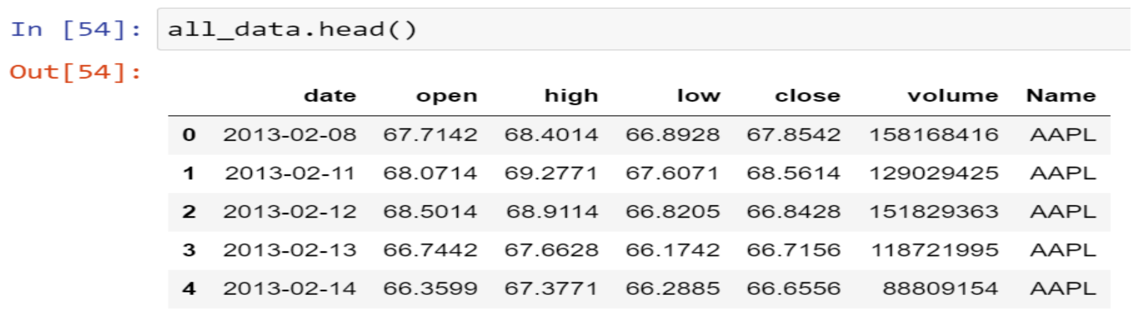
Overall, the process of data acquisition for the Stock Market Dataset involves accessing the raw data from the U.S. Department of Energy's website and importing it into Python using the Pandas library, which can then be used to explore and analyze the data in greater detail.

b) Reading the Data:



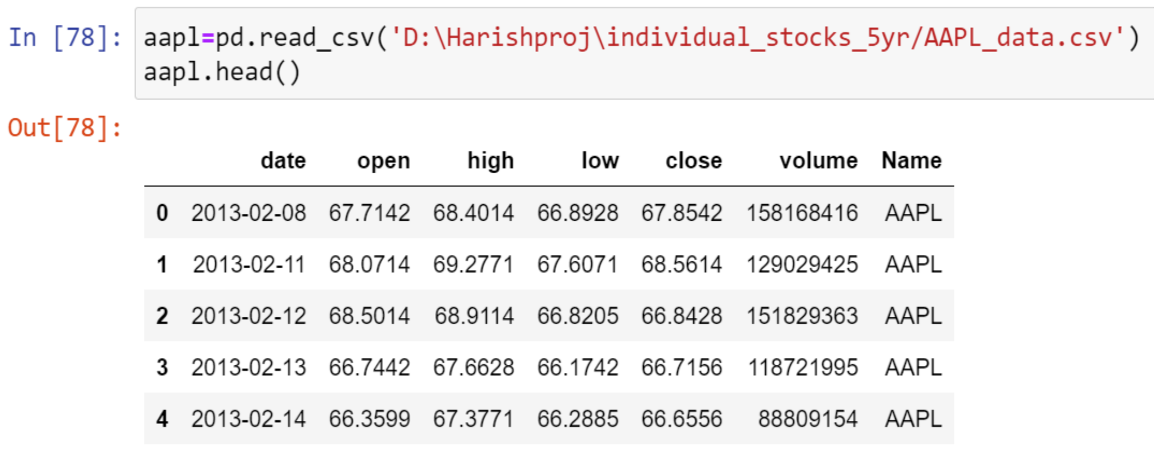
```
In [53]: path='D:\Harishproj\individual_stocks_5yr'
         company_list = ['AAPL_data.csv', 'GOOG_data.csv', 'MSFT_data.csv', 'AMZN_data.csv']

         #blank dataframe
         all_data = pd.DataFrame()
```

c) Display first 5 and last 5 records:



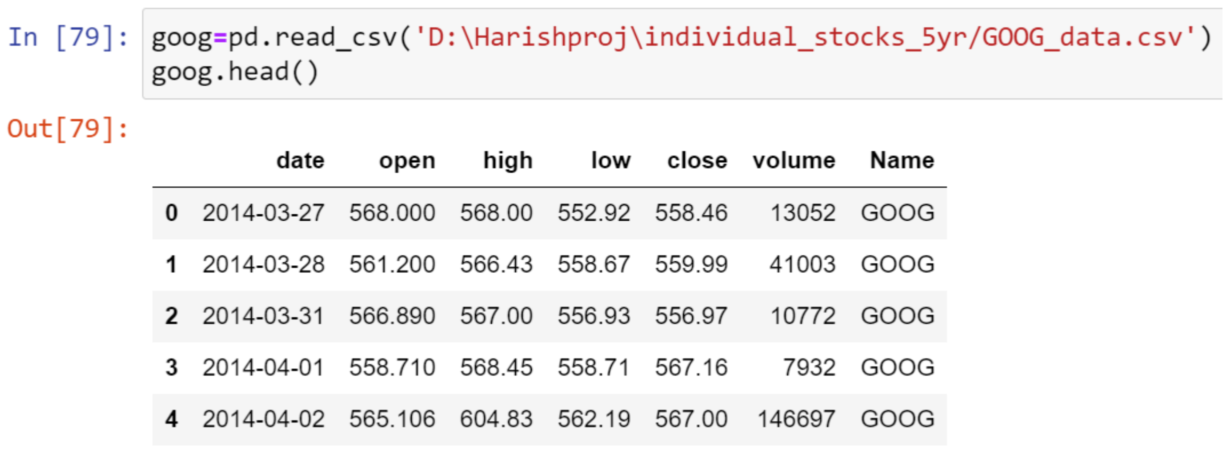
```
In [54]: all_data.head()
Out[54]:
```

	date	open	high	low	close	volume	Name
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL



```
In [78]: aapl=pd.read_csv('D:\Harishproj\individual_stocks_5yr/AAPL_data.csv')
         aapl.head()
Out[78]:
```

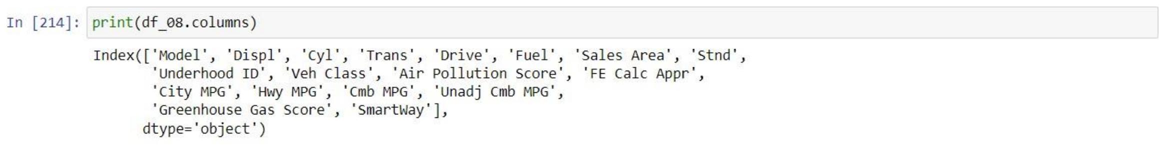
	date	open	high	low	close	volume	Name
0	2013-02-08	67.7142	68.4014	66.8928	67.8542	158168416	AAPL
1	2013-02-11	68.0714	69.2771	67.6071	68.5614	129029425	AAPL
2	2013-02-12	68.5014	68.9114	66.8205	66.8428	151829363	AAPL
3	2013-02-13	66.7442	67.6628	66.1742	66.7156	118721995	AAPL
4	2013-02-14	66.3599	67.3771	66.2885	66.6556	88809154	AAPL



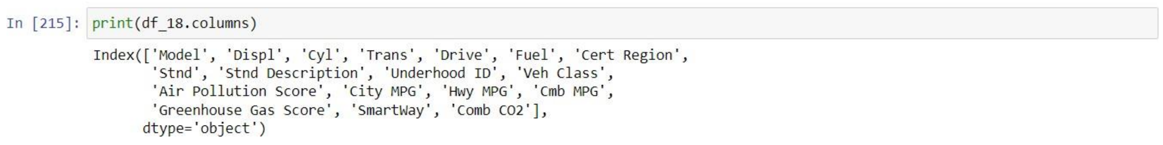
```
In [79]: goog=pd.read_csv('D:\Harishproj\individual_stocks_5yr/GOOG_data.csv')
         goog.head()
Out[79]:
```

	date	open	high	low	close	volume	Name
0	2014-03-27	568.000	568.00	552.92	558.46	13052	GOOG
1	2014-03-28	561.200	566.43	558.67	559.99	41003	GOOG
2	2014-03-31	566.890	567.00	556.93	556.97	10772	GOOG
3	2014-04-01	558.710	568.45	558.71	567.16	7932	GOOG
4	2014-04-02	565.106	604.83	562.19	567.00	146697	GOOG

d) Display column headings:

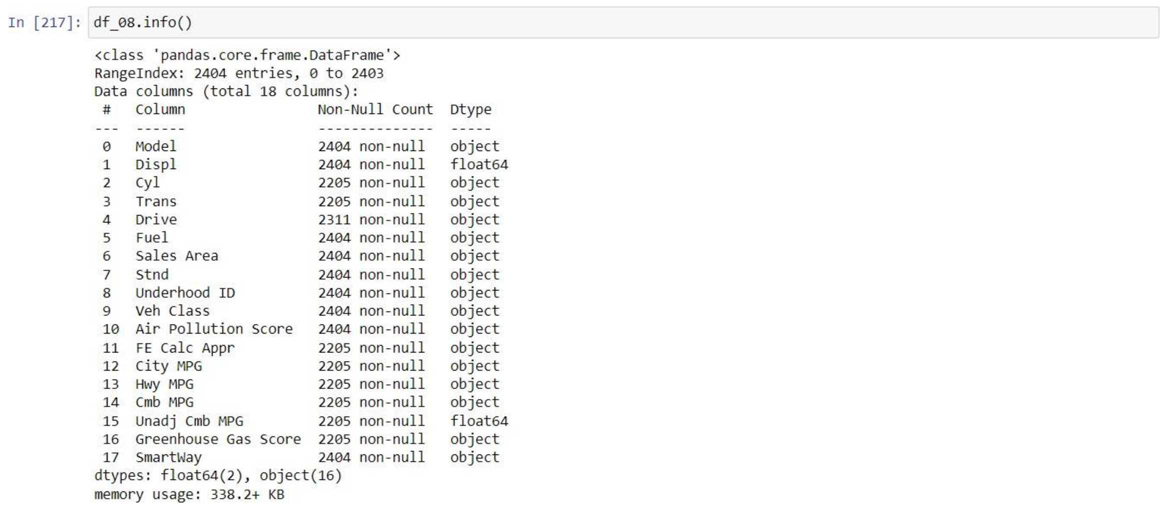


```
In [214]: print(df_08.columns)

Index(['Model', 'Displ', 'Cyl', 'Trans', 'Drive', 'Fuel', 'Sales Area', 'Stnd',
       'Underhood ID', 'Veh Class', 'Air Pollution Score', 'FE Calc Appr',
       'City MPG', 'Hwy MPG', 'Cmb MPG', 'Unadj Cmb MPG',
       'Greenhouse Gas Score', 'SmartWay'],
      dtype='object')
```

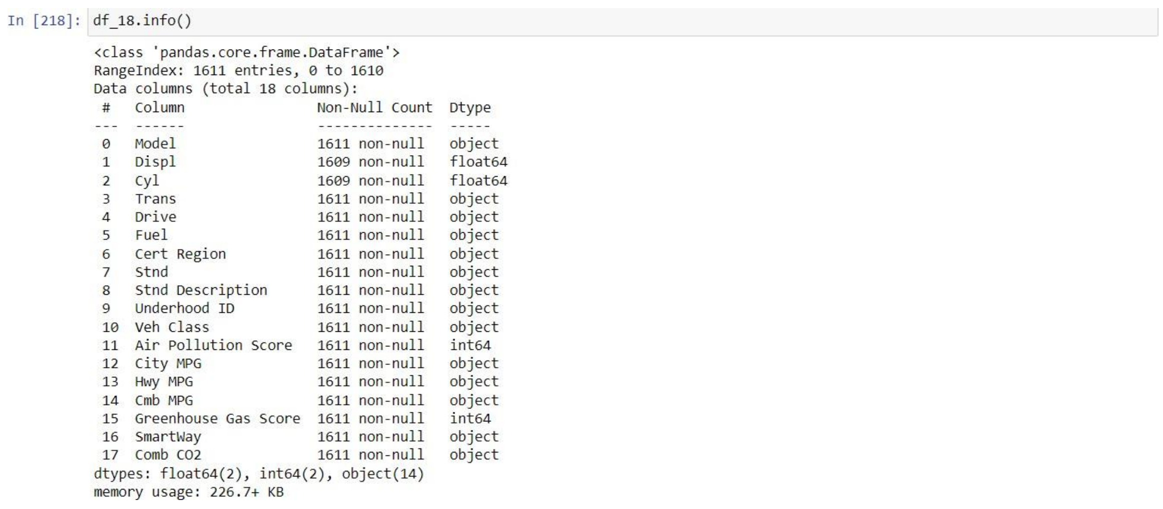


```
In [215]: print(df_18.columns)

Index(['Model', 'Displ', 'Cyl', 'Trans', 'Drive', 'Fuel', 'Cert Region',
       'Stnd', 'Stnd Description', 'Underhood ID', 'Veh Class',
       'Air Pollution Score', 'City MPG', 'Hwy MPG', 'Cmb MPG',
       'Greenhouse Gas Score', 'SmartWay', 'Comb CO2'],
      dtype='object')
```

e) Display Statistical information:

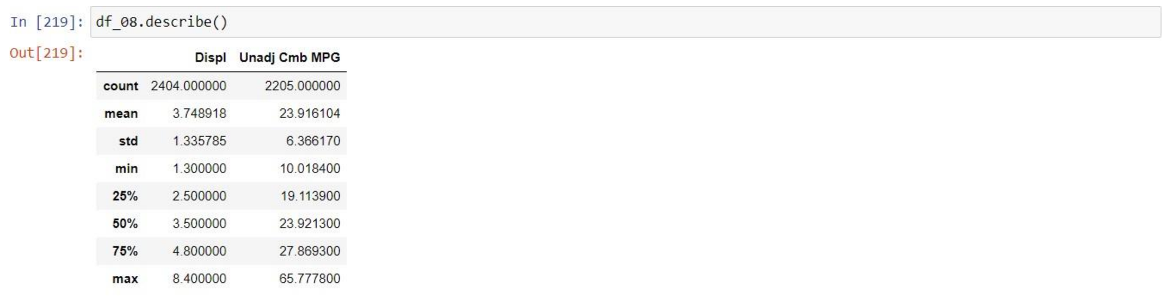


```
In [217]: df_08.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2404 entries, 0 to 2403
Data columns (total 18 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Model                 2404 non-null   object
 1   Displ                 2404 non-null   float64
 2   Cyl                   2205 non-null   object
 3   Trans                 2205 non-null   object
 4   Drive                 2311 non-null   object
 5   Fuel                  2404 non-null   object
 6   Sales Area            2404 non-null   object
 7   Stnd                  2404 non-null   object
 8   Underhood ID          2404 non-null   object
 9   Veh Class             2404 non-null   object
 10  Air Pollution Score   2404 non-null   object
 11  FE Calc Appr          2205 non-null   object
 12  City MPG              2205 non-null   object
 13  Hwy MPG               2205 non-null   object
 14  Cmb MPG               2205 non-null   object
 15  Unadj Cmb MPG         2205 non-null   float64
 16  Greenhouse Gas Score  2205 non-null   object
 17  SmartWay              2404 non-null   object
dtypes: float64(2), object(16)
memory usage: 338.2+ KB
```



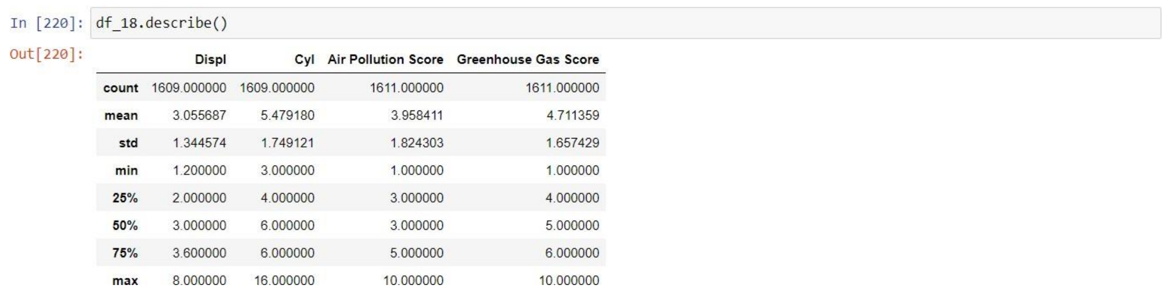
```
In [218]: df_18.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1611 entries, 0 to 1610
Data columns (total 18 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Model                 1611 non-null   object
 1   Displ                 1609 non-null   float64
 2   Cyl                   1609 non-null   float64
 3   Trans                 1611 non-null   object
 4   Drive                 1611 non-null   object
 5   Fuel                  1611 non-null   object
 6   Cert Region           1611 non-null   object
 7   Stnd                  1611 non-null   object
 8   Stnd Description      1611 non-null   object
 9   Underhood ID          1611 non-null   object
 10  Veh Class             1611 non-null   object
 11  Air Pollution Score   1611 non-null   int64
 12  City MPG              1611 non-null   object
 13  Hwy MPG               1611 non-null   object
 14  Cmb MPG               1611 non-null   object
 15  Greenhouse Gas Score  1611 non-null   int64
 16  SmartWay              1611 non-null   object
 17  Comb CO2              1611 non-null   object
dtypes: float64(2), int64(2), object(14)
memory usage: 226.7+ KB
```

f) Display Description of data:



```
In [219]: df_08.describe()
Out[219]:
```

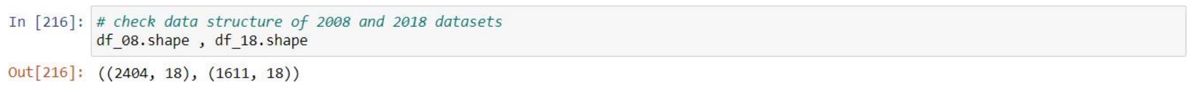
	Displ	Unadj Cmb MPG
count	2404.000000	2205.000000
mean	3.748918	23.916104
std	1.335785	6.366170
min	1.300000	10.018400
25%	2.500000	19.113900
50%	3.500000	23.921300
75%	4.800000	27.869300
max	8.400000	65.777800



```
In [220]: df_18.describe()
Out[220]:
```

	Displ	Cyl	Air Pollution Score	Greenhouse Gas Score
count	1609.000000	1609.000000	1611.000000	1611.000000
mean	3.055687	5.479180	3.958411	4.711359
std	1.344574	1.749121	1.824303	1.657429
min	1.200000	3.000000	1.000000	1.000000
25%	2.000000	4.000000	3.000000	4.000000
50%	3.000000	6.000000	3.000000	5.000000
75%	3.600000	6.000000	5.000000	6.000000
max	8.000000	16.000000	10.000000	10.000000

g) Data Structure



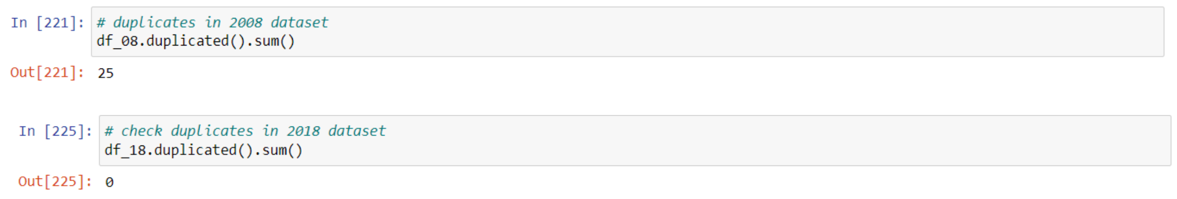
```
In [216]: # check data structure of 2008 and 2018 datasets
          df_08.shape , df_18.shape
Out[216]: ((2404, 18), (1611, 18))
```

6.2 Data Preparation

The data preparation phase in the context of the Stock Market Data project involves refining and structuring the data to ensure it is suitable for analysis using Python's data analysis and visualization libraries such as Pandas, NumPy, and Matplotlib. This phase may include handling missing or inaccurate data, addressing anomalies and outliers, and converting the data into a format suitable for analysis, such as transforming categorical data into numerical values or merging multiple datasets for a comprehensive analysis.

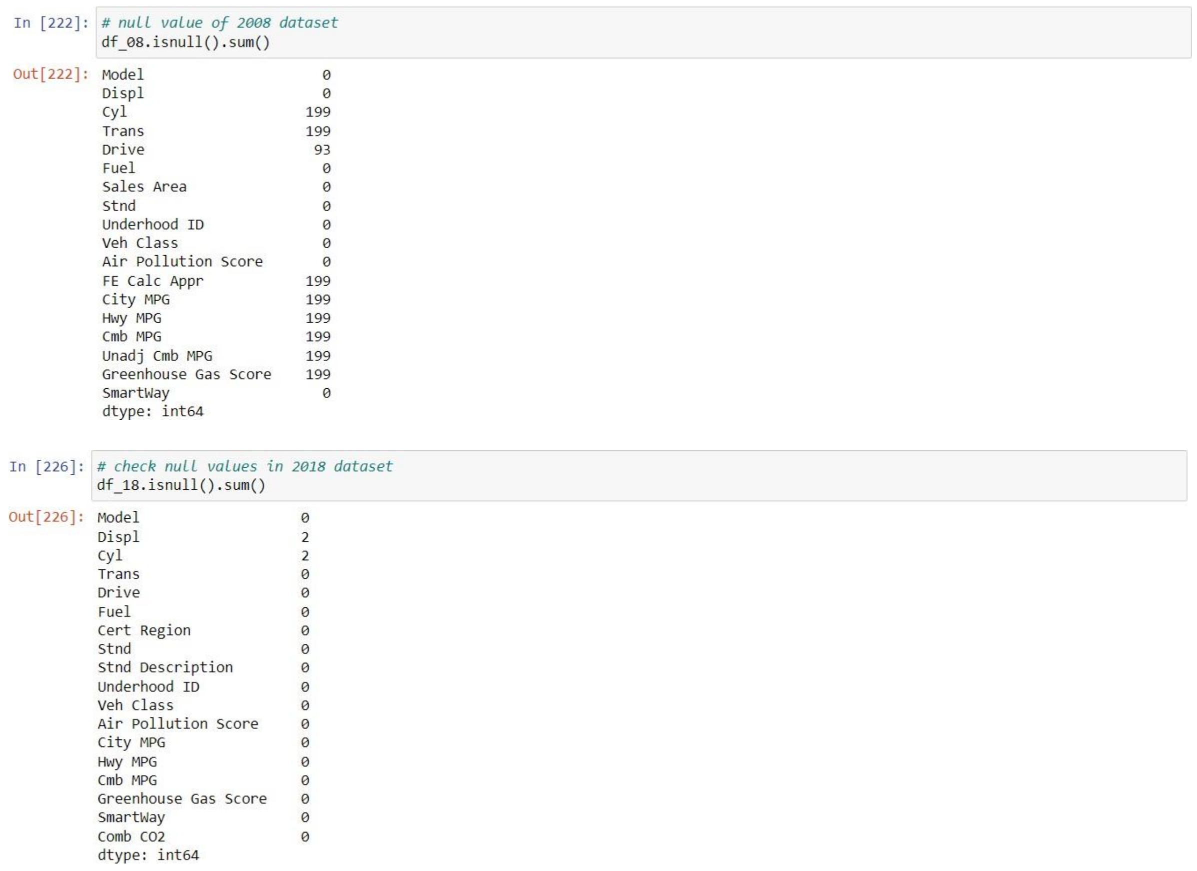
For example, converting the stock prediction column from miles per gallon to liters per 100 kilometers could simplify analysis for international comparisons. This stage is critical because it ensures the data is reliable and accurate, which is essential for generating meaningful insights and drawing valid conclusions. Data preparation can be a time-consuming process, but it is necessary to ensure the analysis is accurate, relevant, and useful for the project's objectives.

a) Duplicate data:



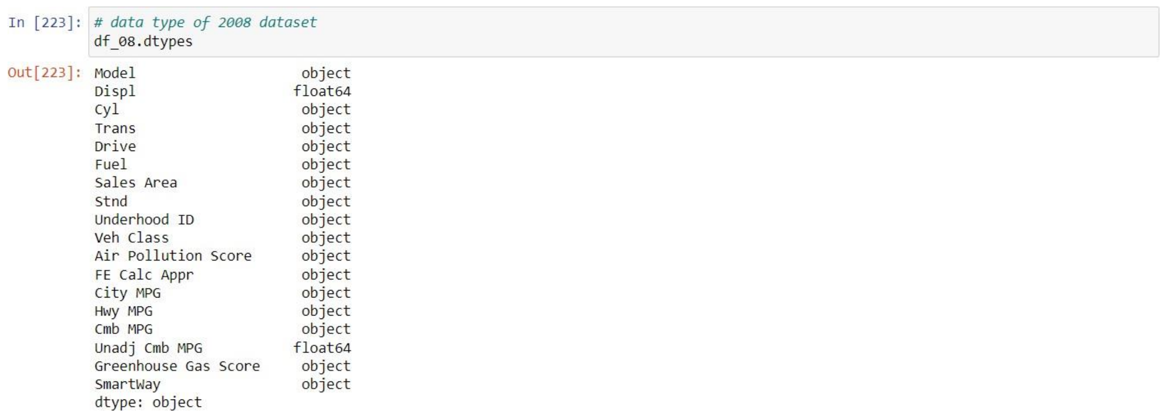
```
In [221]: # duplicates in 2008 dataset
          df_08.duplicated().sum()

Out[221]: 25

In [225]: # check duplicates in 2018 dataset
          df_18.duplicated().sum()

Out[225]: 0
```

b) Missing data:

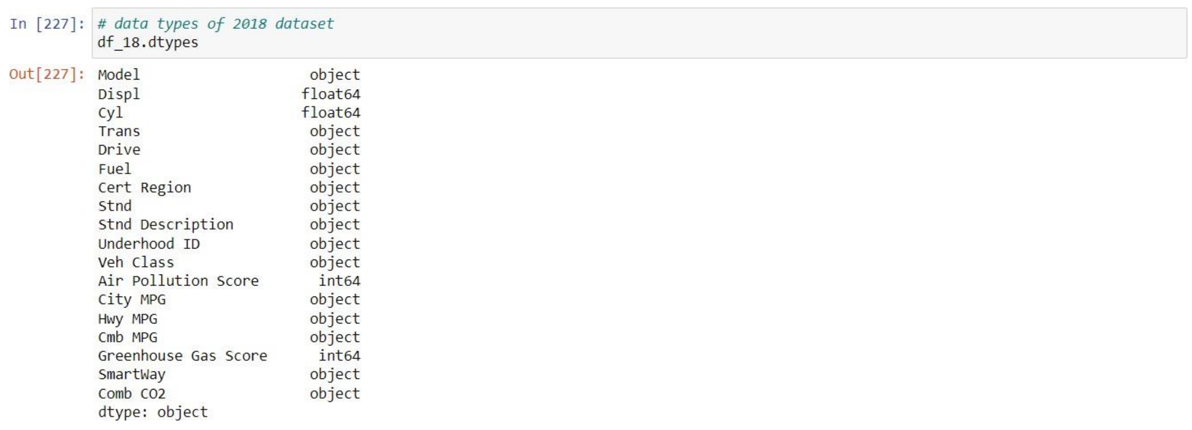


```
In [222]: # null value of 2008 dataset
          df_08.isnull().sum()

Out[222]: Model                     0
          Displ                     0
          Cyl                     199
          Trans                   199
          Drive                    93
          Fuel                      0
          Sales Area                0
          Stnd                      0
          Underhood ID              0
          Veh Class                 0
          Air Pollution Score       0
          FE Calc Appr            199
          City MPG                199
          Hwy MPG                 199
          Cmb MPG                 199
          Unadj Cmb MPG           199
          Greenhouse Gas Score    199
          SmartWay                  0
          dtype: int64

In [226]: # check null values in 2018 dataset
          df_18.isnull().sum()

Out[226]: Model                   0
          Displ                   2
          Cyl                     2
          Trans                   0
          Drive                   0
          Fuel                    0
          Cert Region             0
          Stnd                    0
          Stnd Description        0
          Underhood ID            0
          Veh Class               0
          Air Pollution Score     0
          City MPG                0
          Hwy MPG                 0
          Cmb MPG                 0
          Greenhouse Gas Score    0
          SmartWay                0
          Comb CO2                0
          dtype: int64
```

c) Data types:

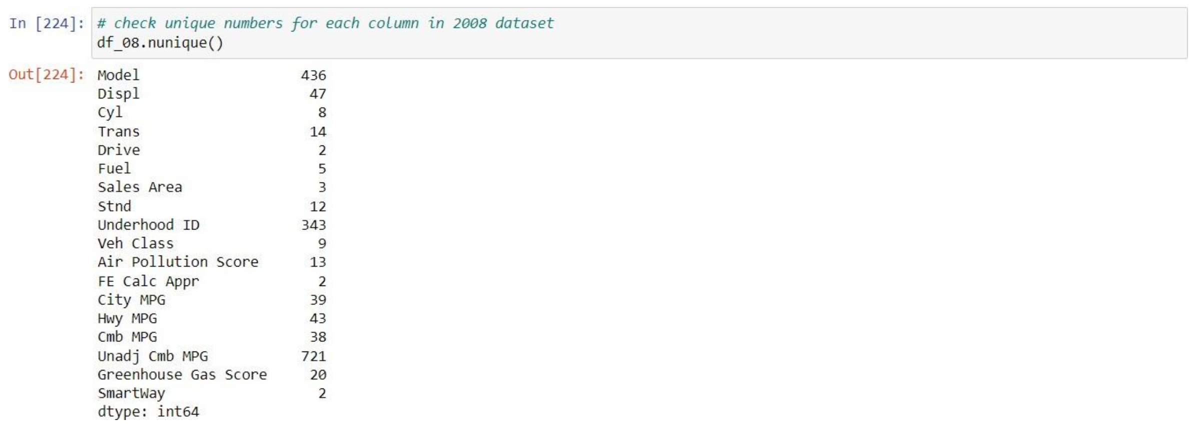


```
In [223]: # data type of 2008 dataset
          df_08.dtypes

Out[223]: Model                    object
          Displ                   float64
          Cyl                      object
          Trans                    object
          Drive                    object
          Fuel                     object
          Sales Area               object
          Stnd                     object
          Underhood ID             object
          Veh Class                object
          Air Pollution Score      object
          FE Calc Appr             object
          City MPG                 object
          Hwy MPG                  object
          Cmb MPG                  object
          Unadj Cmb MPG           float64
          Greenhouse Gas Score     object
          SmartWay                 object
          dtype: object
```

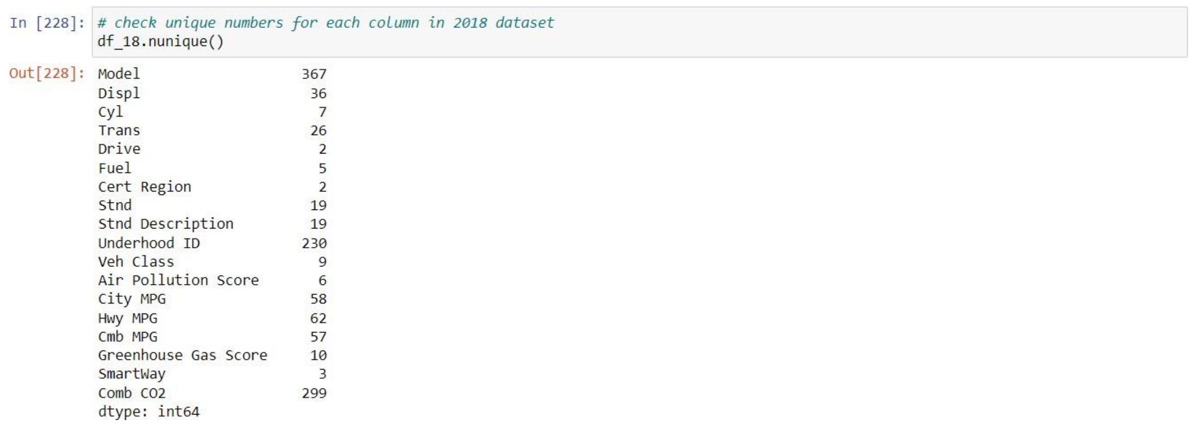


```
In [227]: # data types of 2018 dataset
          df_18.dtypes

Out[227]: Model                    object
          Displl                  float64
          Cyl                     float64
          Trans                    object
          Drive                    object
          Fuel                     object
          Cert Region              object
          Stnd                     object
          Stnd Description         object
          Underhood ID             object
          Veh Class                object
          Air Pollution Score       int64
          City MPG                 object
          Hwy MPG                  object
          Cmb MPG                  object
          Greenhouse Gas Score      int64
          SmartWay                 object
          Comb CO2                 object
          dtype: object
```

d) Unique data:



```
In [224]: # check unique numbers for each column in 2008 dataset
          df_08.nunique()

Out[224]: Model                   436
          Displ                    47
          Cyl                       8
          Trans                    14
          Drive                     2
          Fuel                      5
          Sales Area                3
          Stnd                     12
          Underhood ID            343
          Veh Class                 9
          Air Pollution Score      13
          FE Calc Appr              2
          City MPG                 39
          Hwy MPG                  43
          Cmb MPG                  38
          Unadj Cmb MPG           721
          Greenhouse Gas Score     20
          SmartWay                  2
          dtype: int64
```



```
In [228]: # check unique numbers for each column in 2018 dataset
          df_18.nunique()

Out[228]: Model                   367
          Displ                    36
          Cyl                       7
          Trans                    26
          Drive                     2
          Fuel                      5
          Cert Region               2
          Stnd                     19
          Stnd Description         19
          Underhood ID            230
          Veh Class                 9
          Air Pollution Score       6
          City MPG                 58
          Hwy MPG                  62
          Cmb MPG                  57
          Greenhouse Gas Score     10
          SmartWay                  3
          Comb CO2                299
          dtype: int64
```

6.3 Data exploration and Visualization

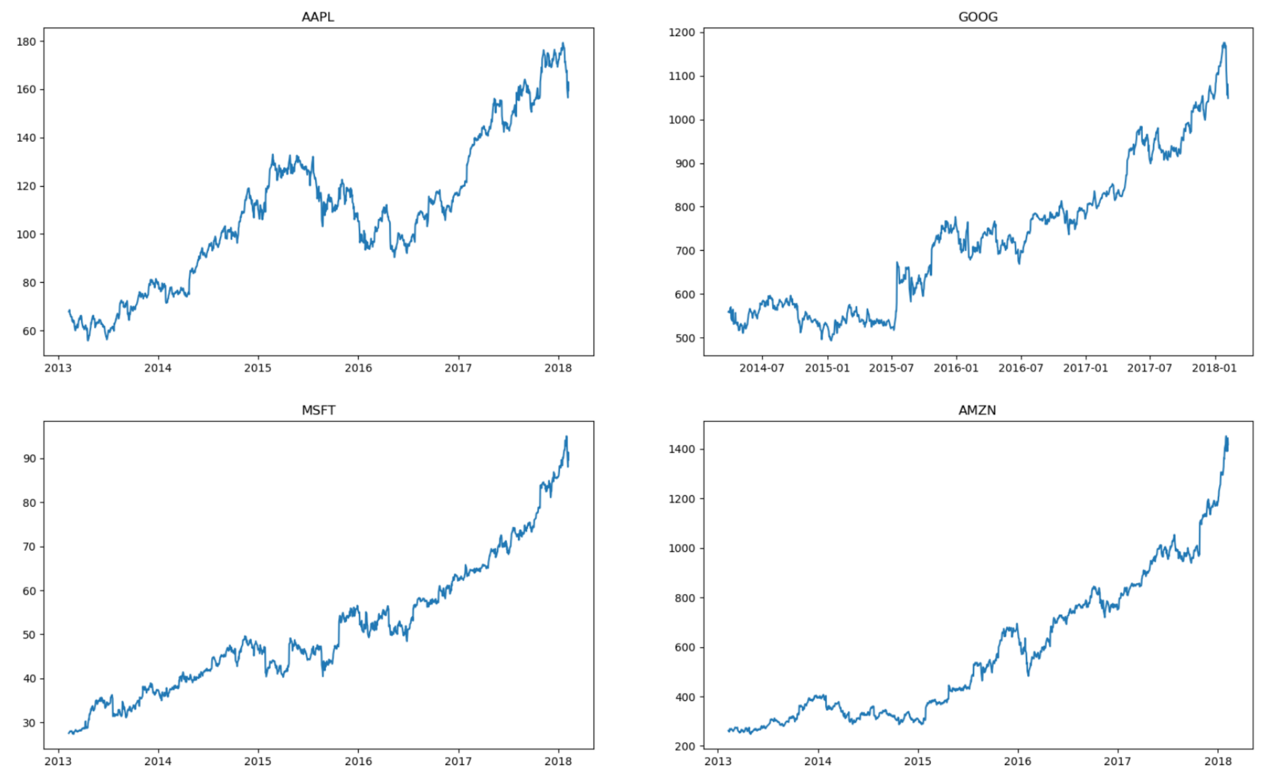
In the context of analyzing car sales data using the "Data Analytics with Stock Market Data" Python project, data exploration involves utilizing various data visualization and analysis techniques to uncover patterns, trends, and correlations within the data. This may include using histograms, scatterplots, and other visualization tools to identify potential relationships between car models and attributes such as mileage or price.

For example, data visualization techniques could be utilized to identify potential connections between stock prediction and car model, year, or manufacturer. Additionally, data exploration techniques could be used to identify any outliers or anomalies within the data that may need to be addressed in further analysis. Overall, data exploration is a critical step in analyzing car sales data using the "Data Analytics with Stock Market Data" Python project to gain insights and inform decision-making.

Using Different Plots for Stock Market Prediction

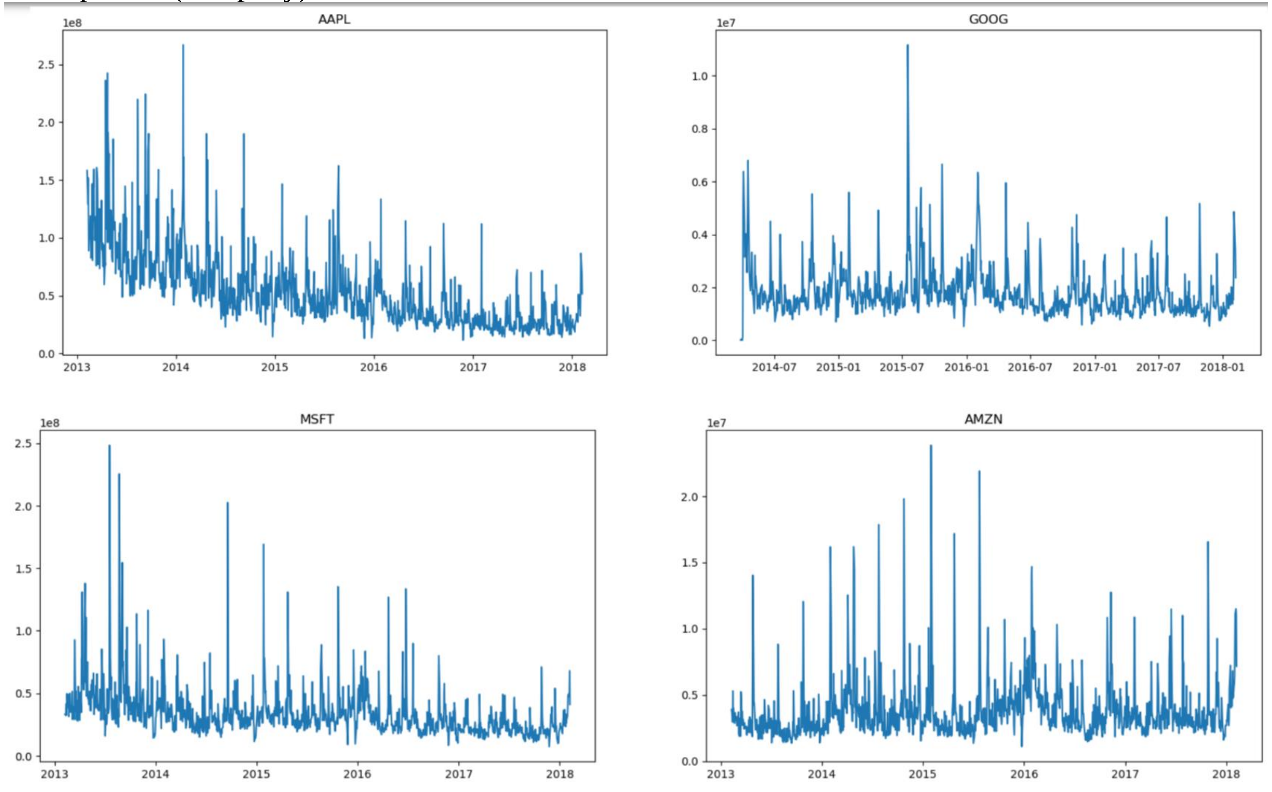
Line Plots

```
plt.figure(figsize=(20,12))
for i, company in enumerate(tech_list,1):
    plt.subplot(2, 2, i)
    df=all_data[all_data['Name']==company]
    plt.plot(df['date'],df['close'])
    plt.title(company)
```



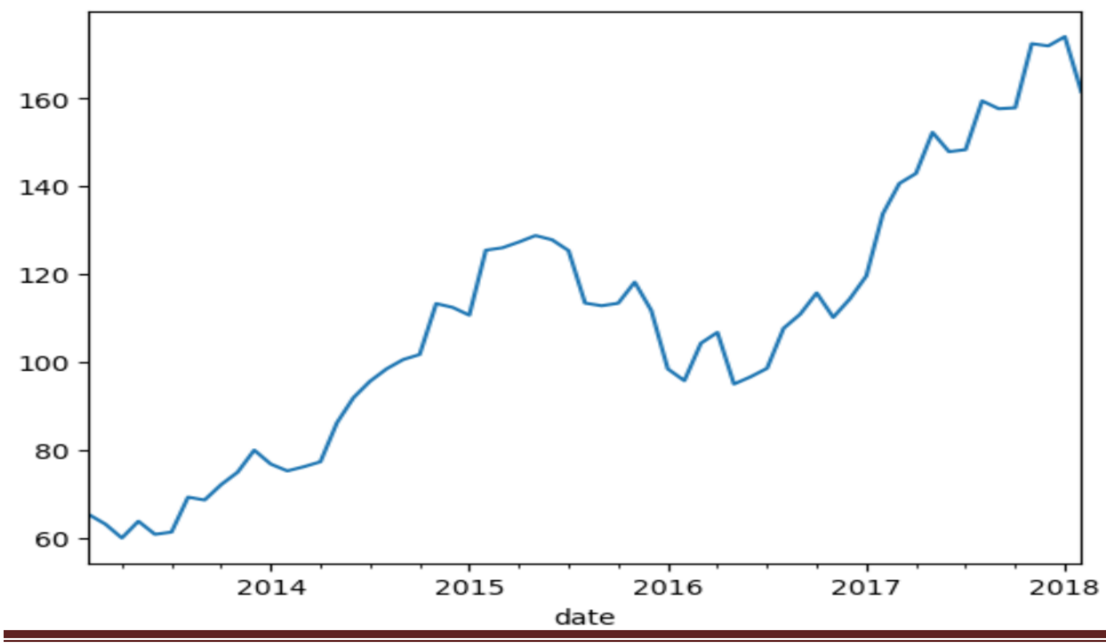
Sub plot on single categorical variable.

```
plt.figure(figsize=(20,12))
for i, company in enumerate(tech_list,1):
    plt.subplot(2, 2, i)
    df=all_data[all_data['Name']==company]
    plt.plot(df['date'],df['volume'])
    plt.title(company)
```



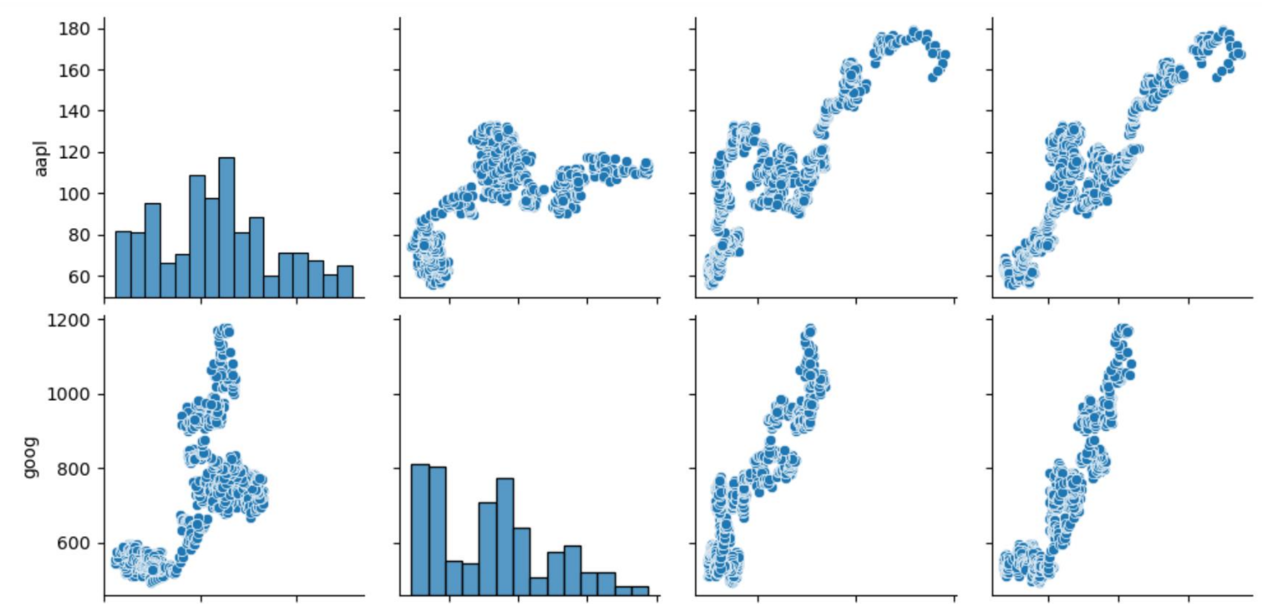
Resampling plots

```
df2['close'].resample('M').mean().plot()
```



Pairplot using seaborn

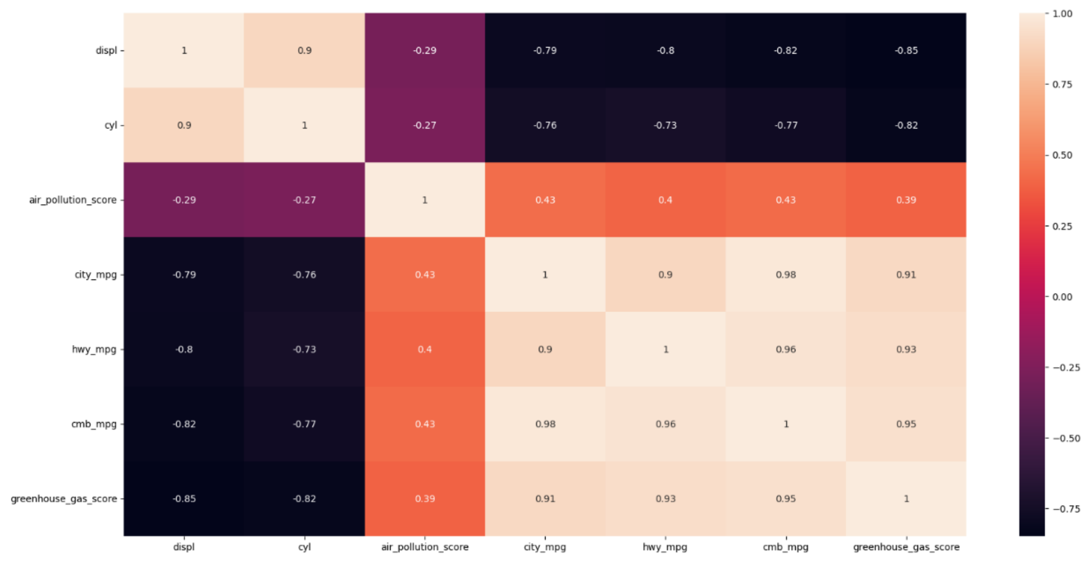
```
sns.pairplot(data=close)
```



Heatmap on each Dataset.

```
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
plt.figure(figsize=(20,10))
sns.heatmap(df_08.corr(),annot=True)
```

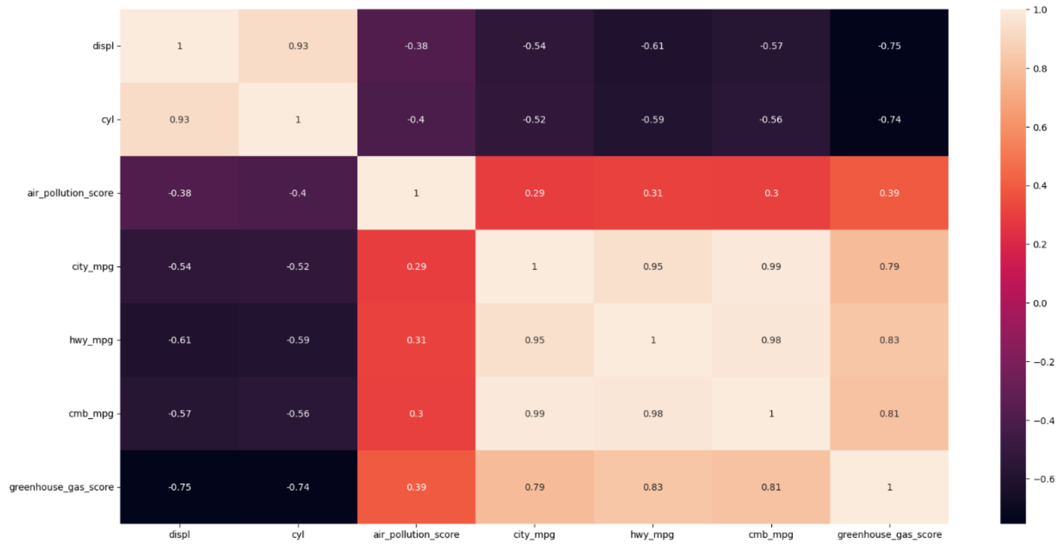
```
Out[303]: <Axes: >
```



	displ	cyl	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score
displ	1	0.9	-0.29	-0.79	-0.8	-0.82	-0.85
cyl	0.9	1	-0.27	-0.76	-0.73	-0.77	-0.82
air_pollution_score	-0.29	-0.27	1	0.43	0.4	0.43	0.39
city_mpg	-0.79	-0.76	0.43	1	0.9	0.98	0.91
hwy_mpg	-0.8	-0.73	0.4	0.9	1	0.96	0.93
cmb_mpg	-0.82	-0.77	0.43	0.98	0.96	1	0.95
greenhouse_gas_score	-0.85	-0.82	0.39	0.91	0.93	0.95	1

```
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
plt.figure(figsize=(20,10))
sns.heatmap(df_18.corr(),annot=True)
```

```
Out[304]: <Axes: >
```

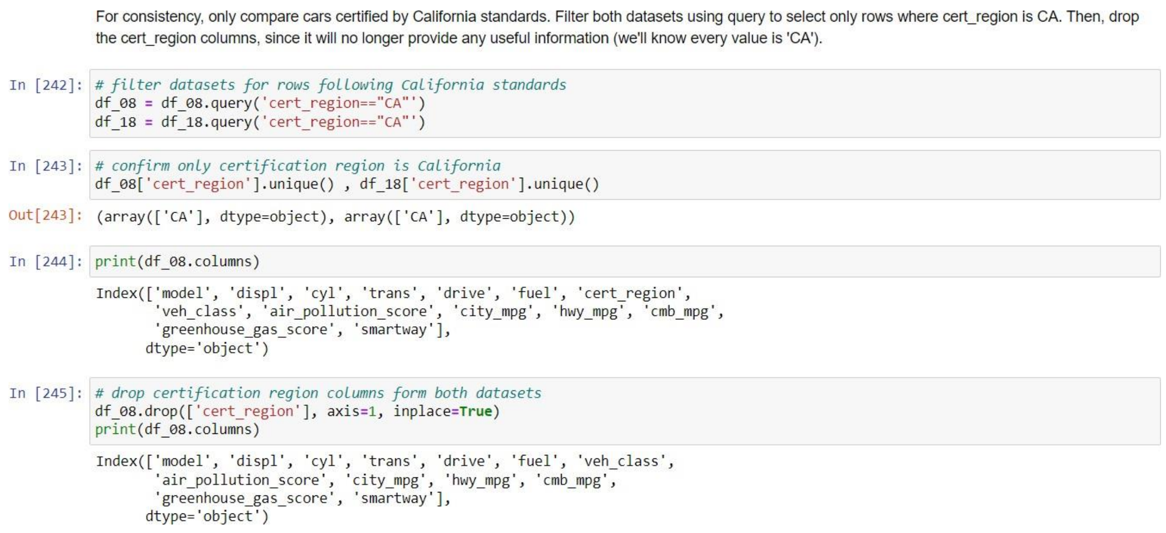


6.4 Data wrangling

In the context of analyzing car sales data using the "Data Analytics with Stock Market Data" Python project, data wrangling refers to the process of transforming and restructuring the data to prepare it for analysis and visualization. This may include selecting relevant attributes from the dataset, removing any unnecessary information, and transforming the data into a format suitable for analysis.

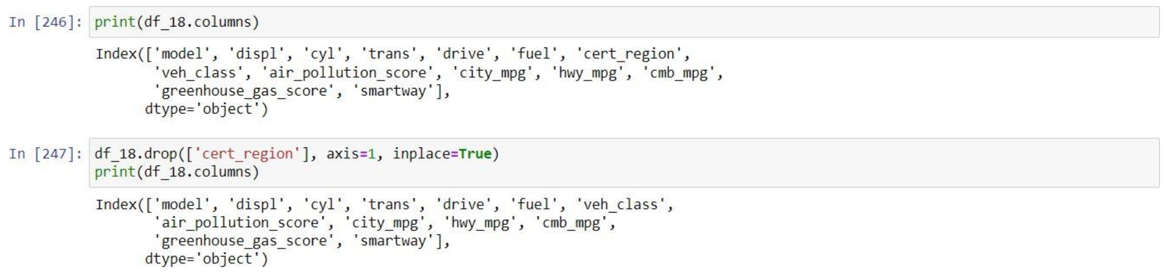
For example, data wrangling could involve filtering the Stock Market Data to include only the most relevant attributes for analysis, such as car make, model, year, and stock prediction metrics. Additionally, data may need to be cleaned, merged, or transformed to create a more complete and usable dataset for visualization and analysis.

The purpose of data wrangling is to make the data more manageable and structured for effective analysis and visualization, which can lead to improved accuracy and insights. By preparing the data for analysis, the project can identify patterns and trends that may not be immediately visible, and create visualizations that help to convey these insights in a clear and concise manner.

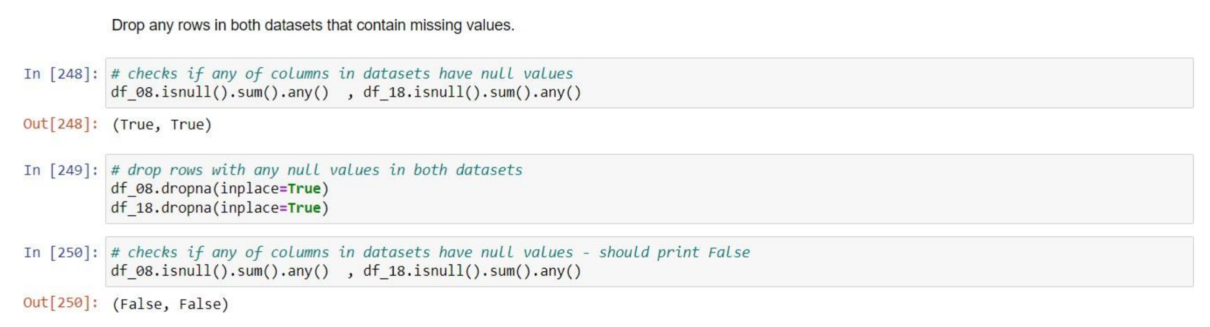


For consistency, only compare cars certified by California standards. Filter both datasets using query to select only rows where cert_region is CA. Then, drop the cert_region columns, since it will no longer provide any useful information (we'll know every value is 'CA').

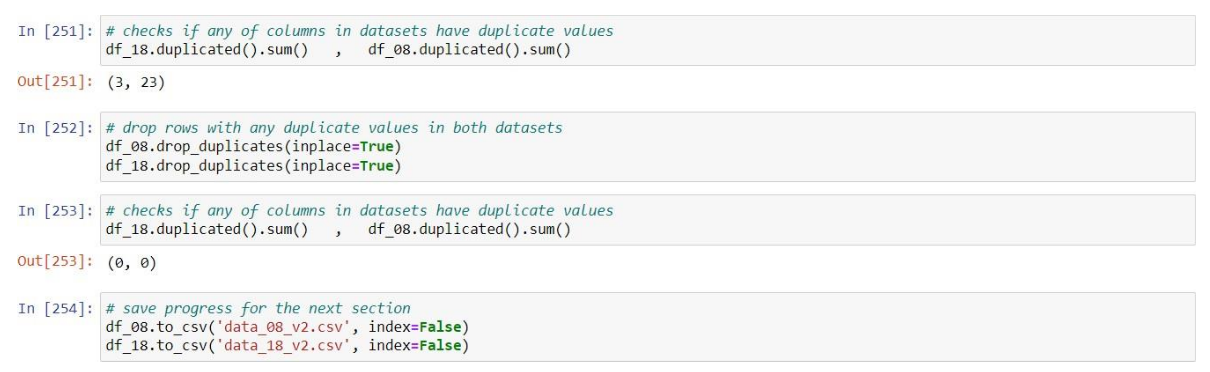
```
In [242]: # filter datasets for rows following California standards
          df_08 = df_08.query('cert_region=="CA"')
          df_18 = df_18.query('cert_region=="CA"')

In [243]: # confirm only certification region is California
          df_08['cert_region'].unique() , df_18['cert_region'].unique()

Out[243]: (array(['CA'], dtype=object), array(['CA'], dtype=object))

In [244]: print(df_08.columns)

          Index(['model', 'displ', 'cyl', 'trans', 'drive', 'fuel', 'cert_region',
                 'veh_class', 'air_pollution_score', 'city_mpg', 'hwy_mpg', 'cmb_mpg',
                 'greenhouse_gas_score', 'smartway'],
                dtype='object')

In [245]: # drop certification region columns form both datasets
          df_08.drop(['cert_region'], axis=1, inplace=True)
          print(df_08.columns)

          Index(['model', 'displ', 'cyl', 'trans', 'drive', 'fuel', 'veh_class',
                 'air_pollution_score', 'city_mpg', 'hwy_mpg', 'cmb_mpg',
                 'greenhouse_gas_score', 'smartway'],
                dtype='object')
```



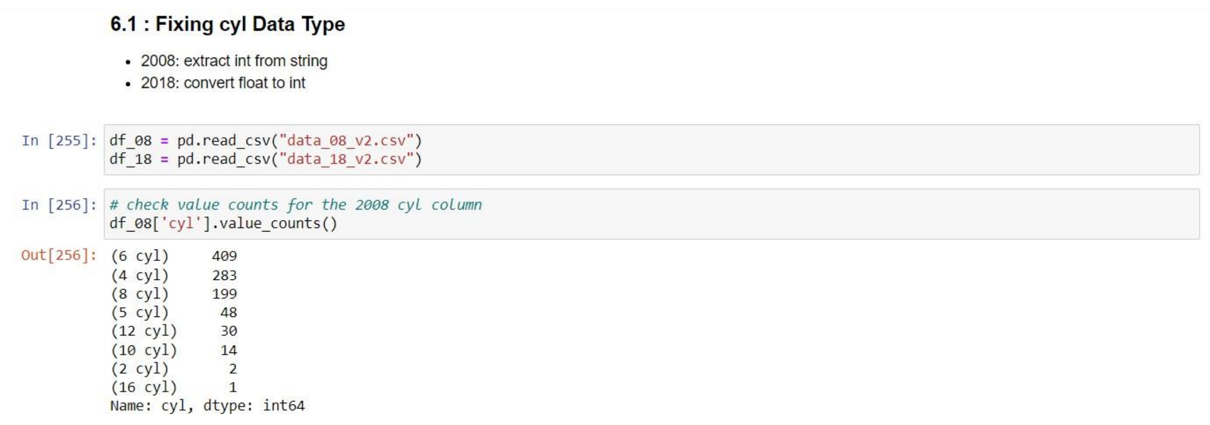
```
In [246]: print(df_18.columns)

          Index(['model', 'displ', 'cyl', 'trans', 'drive', 'fuel', 'cert_region',
                 'veh_class', 'air_pollution_score', 'city_mpg', 'hwy_mpg', 'cmb_mpg',
                 'greenhouse_gas_score', 'smartway'],
                dtype='object')

In [247]: df_18.drop(['cert_region'], axis=1, inplace=True)
          print(df_18.columns)

          Index(['model', 'displ', 'cyl', 'trans', 'drive', 'fuel', 'veh_class',
                 'air_pollution_score', 'city_mpg', 'hwy_mpg', 'cmb_mpg',
                 'greenhouse_gas_score', 'smartway'],
                dtype='object')
```


Drop any rows in both datasets that contain missing values.

```
In [248]: # checks if any of columns in datasets have null values
          df_08.isnull().sum().any()  , df_18.isnull().sum().any()

Out[248]: (True, True)

In [249]: # drop rows with any null values in both datasets
          df_08.dropna(inplace=True)
          df_18.dropna(inplace=True)

In [250]: # checks if any of columns in datasets have null values - should print False
          df_08.isnull().sum().any()  , df_18.isnull().sum().any()

Out[250]: (False, False)
```

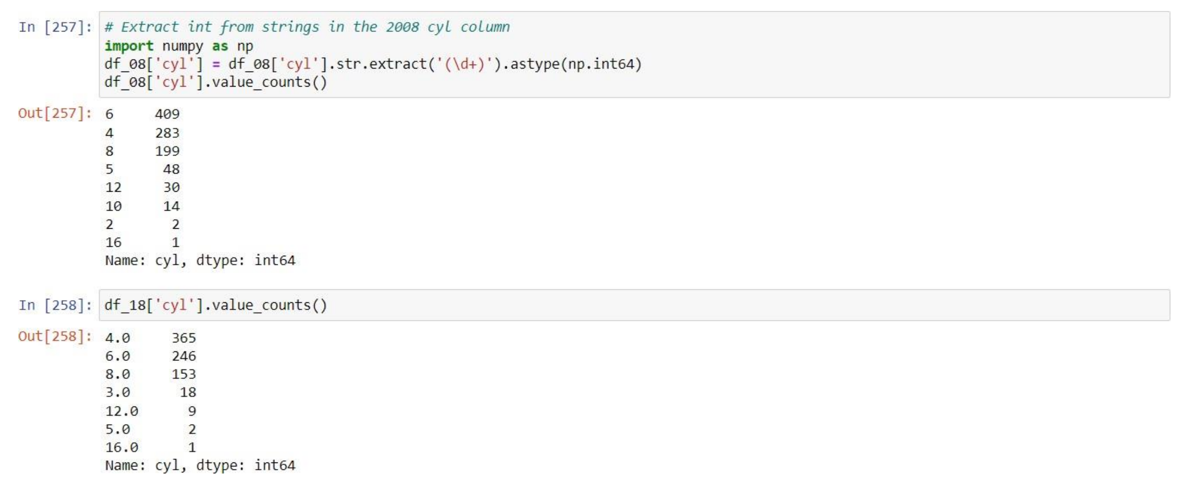


```
In [251]: # checks if any of columns in datasets have duplicate values
          df_18.duplicated().sum()   ,   df_08.duplicated().sum()

Out[251]: (3, 23)

In [252]: # drop rows with any duplicate values in both datasets
          df_08.drop_duplicates(inplace=True)
          df_18.drop_duplicates(inplace=True)

In [253]: # checks if any of columns in datasets have duplicate values
          df_18.duplicated().sum()   ,   df_08.duplicated().sum()

Out[253]: (0, 0)

In [254]: # save progress for the next section
          df_08.to_csv('data_08_v2.csv', index=False)
          df_18.to_csv('data_18_v2.csv', index=False)
```


6.1 : Fixing cyl Data Type

- 2008: extract int from string
- 2018: convert float to int

```
In [255]: df_08 = pd.read_csv("data_08_v2.csv")
          df_18 = pd.read_csv("data_18_v2.csv")

In [256]: # check value counts for the 2008 cyl column
          df_08['cyl'].value_counts()

Out[256]: (6 cyl)     409
          (4 cyl)     283
          (8 cyl)     199
          (5 cyl)      48
          (12 cyl)     30
          (10 cyl)     14
          (2 cyl)       2
          (16 cyl)      1
          Name: cyl, dtype: int64
```



```
In [257]: # Extract int from strings in the 2008 cyl column
          import numpy as np
          df_08['cyl'] = df_08['cyl'].str.extract('(\d+)').astype(np.int64)
          df_08['cyl'].value_counts()

Out[257]: 6     409
          4     283
          8     199
          5      48
          12     30
          10     14
          2       2
          16      1
          Name: cyl, dtype: int64

In [258]: df_18['cyl'].value_counts()

Out[258]: 4.0     365
          6.0     246
          8.0     153
          3.0      18
          12.0      9
          5.0       2
          16.0      1
          Name: cyl, dtype: int64
```

Drop any rows in both datasets that contain missing values.

```
In [248]: # checks if any of columns in datasets have null values
          df_08.isnull().sum().any()  , df_18.isnull().sum().any()

Out[248]: (True, True)
```

```
In [249]: # drop rows with any null values in both datasets
          df_08.dropna(inplace=True)
          df_18.dropna(inplace=True)
```

```
In [250]: # checks if any of columns in datasets have null values - should print False
          df_08.isnull().sum().any()  , df_18.isnull().sum().any()

Out[250]: (False, False)
```

6.2 Fixing air_pollution_score Data Type

- 2008: convert string to float
- 2018: convert int to float

```
In [261]: # load datasets
          import pandas as pd
          df_08 = pd.read_csv('data_08_v3.csv')
          df_18 = pd.read_csv('data_18_v3.csv')
```

```
In [262]: df_08.iloc[582]    # This is used to retrieve a specific row from the Data Set

Out[262]: model                    MERCEDES-BENZ C300
          displ                                   3.0
          cyl                                       6
          trans                               Auto-L7
          drive                                   2WD
          fuel                            ethanol/gas
          veh_class                         small car
          air_pollution_score                     6/4
          city_mpg                              13/18
          hwy_mpg                               19/25
          cmb_mpg                               15/21
          greenhouse_gas_score                    7/6
          smartway                                 no
          Name: 582, dtype: object
```

```
In [263]: # First, let's get all the hybrids in 2008
          hb_08 = df_08[df_08['fuel'].str.contains('/')]
          hb_08
```

Out[263]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway
582	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	ethanol/gas	small car	6/4	13/18	19/25	15/21	7/6	no

```
In [264]: # hybrids in 2018
          hb_18 = df_18[df_18['fuel'].str.contains('/')]
          hb_18
```

Out[264]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartw
52	BMW 330e	2.0	4	SemiAuto-8	2WD	Gasoline/Electricity	small car	3	28/66	34/78	30/71	10	
78	BMW 530e	2.0	4	SemiAuto-8	2WD	Gasoline/Electricity	small car	7	27/70	31/75	29/72	10	E
79	BMW 530e	2.0	4	SemiAuto-8	4WD	Gasoline/Electricity	small car	7	27/66	31/68	28/67	10	E
92	BMW 740e	2.0	4	SemiAuto-8	4WD	Gasoline/Electricity	large car	3	25/62	29/68	27/64	9	
189	CHEVROLET Impala	3.6	6	SemiAuto-6	2WD	Ethanol/Gas	large car	5	14/18	20/28	16/22	4	
195	CHEVROLET Silverado 15	4.3	6	Auto-6	2WD	Ethanol/Gas	pickup	5	12/18	16/24	14/20	4	
196	CHEVROLET Silverado 15	4.3	6	Auto-6	4WD	Ethanol/Gas	pickup	5	12/17	15/22	13/19	3	

In [265]:
```
# create two copies of the 2008 hybrids dataframe
df1 = hb_08.copy()  # data on first fuel type of each hybrid vehicle
df2 = hb_08.copy()  # data on second fuel type of each hybrid vehicle

# Each one should look like this
df2
```

Out[265]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway
582	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	ethanol/gas	small car	6/4	13/18	19/25	15/21	7/6	no

In [266]:
```
# columns to split by "/"
split_columns = ['fuel', 'air_pollution_score', 'city_mpg', 'hwy_mpg', 'cmb_mpg', 'greenhouse_gas_score']

# apply split function to each column of each dataframe copy
for c in split_columns:
    df1[c] = df1[c].apply(lambda x: x.split("/")[0])
    df2[c] = df2[c].apply(lambda x: x.split("/")[1])
#A lambda function can take any number of arguments, but can only have one expression.
#Ex: Add 10 to argument a, and return the result:
#x = lambda a : a + 10
#print(x(5))
# O/P
#15
```

In [267]:
```
# this dataframe holds info for the FIRST fuel type of the hybrid
# aka the values before the "/"s
df1
```

Out[267]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway
582	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	ethanol	small car	6	13	19	15	7	no

In [268]:
```
# this dataframe holds info for the SECOND fuel type of the hybrid
# aka the values after the "/"s
df2
```

Out[268]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway
582	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	gas	small car	4	18	25	21	6	no

In [269]:
```
# combine dataframes to add to the original dataframe
new_rows = pd.concat([df1,df2])

# now we have separate rows for each fuel type of each vehicle!
new_rows
```

Out[269]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway
582	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	ethanol	small car	6	13	19	15	7	no
582	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	gas	small car	4	18	25	21	6	no

In [270]:
```
# drop the original hybrid rows
df_08.drop(hb_08.index, inplace=True)

# add in our newly separated rows
df_08 = pd.concat([df_08,new_rows], ignore_index=True)
```

In [271]:
```
# check that all the original hybrid rows with "/"s are gone
df_08[df_08['fuel'].str.contains('/')]
```

Out[271]:

model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway

In [272]:
```
df_08.shape
```

Out[272]:
```
(987, 13)
```

Repeat the process for dataset 2018

In [273]:
```
# create two copies of the 2018 hybrids dataframe, hb_18
df1 = hb_18.copy()
df2 = hb_18.copy()
# df1
```

```
In [274]: # apply split function to each column of each dataframe copy

          # list of columns to split
          split_columns = ['fuel','city_mpg', 'hwy_mpg','cmb_mpg']

          # apply split function to each column of each dataframe copy
          for c in split_columns:
              df1[c] = df1[c].apply(lambda x: x.split("/")[0])
              df2[c] = df2[c].apply(lambda x: x.split("/")[1])
          # df1  no more /
          # df2
```

```
In [275]: # append the two dataframes
          new_rows = pd.concat([df1,df2])

          # drop each hybrid row from the original 2018 dataframe
          # do this by using Pandas drop function with hb_18's index
          df_18.drop(hb_18.index, inplace=True) # drop original one

          # append new_rows to df_18
          df_18 = pd.concat([df_18,new_rows],ignore_index=True)
```

```
In [276]: # check that they're gone
          df_18[df_18['fuel'].str.contains('/')]
```

Out[276]:

model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway

```
In [277]: df_18.shape
```

6.3 Fix city_mpg, hwy_mpg, cmb_mpg datatypes

2008 and 2018: convert string to float

```
In [282]: # convert mpg columns to floats
          mpg_columns = ['city_mpg', 'hwy_mpg', 'cmb_mpg']
          for c in mpg_columns:
              df_18[c] = df_18[c].astype(float)
              df_08[c] = df_08[c].astype(float)
```

```
In [283]: df_08.dtypes
```

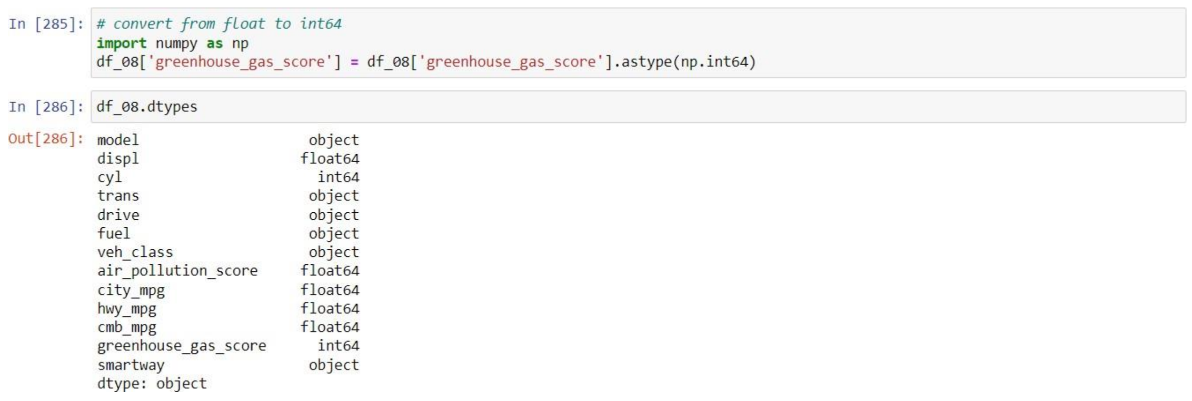
```
Out[283]: model                    object
          displ                   float64
          cyl                       int64
          trans                    object
          drive                    object
          fuel                     object
          veh_class                object
          air_pollution_score     float64
          city_mpg                float64
          hwy_mpg                 float64
          cmb_mpg                 float64
          greenhouse_gas_score      int64
          smartway                 object
          dtype: object
```

```
In [284]: df_18.dtypes
```

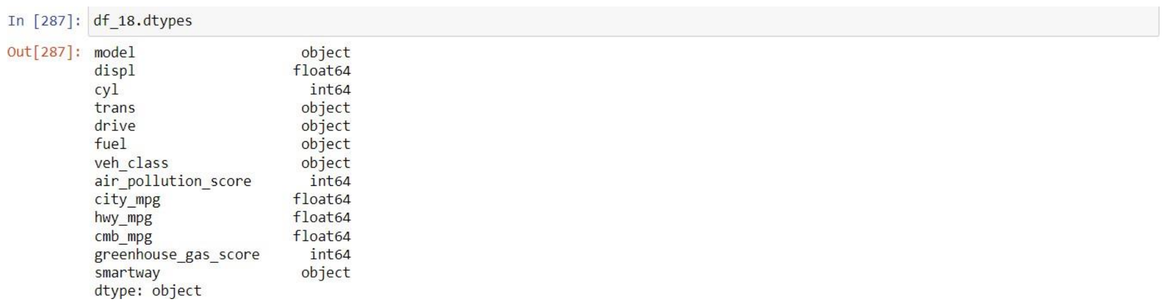
```
Out[284]: model                    object
          displ                   float64
          cyl                       int64
          trans                    object
          drive                    object
          fuel                     object
          veh_class                object
          air_pollution_score       int64
          city_mpg                float64
          hwy_mpg                 float64
          cmb_mpg                 float64
          greenhouse_gas_score      int64
          smartway                 object
          dtype: object
```

6.4 Fix greenhouse_gas_score datatype

2008: convert from float to int



```
In [285]: # convert from float to int64
          import numpy as np
          df_08['greenhouse_gas_score'] = df_08['greenhouse_gas_score'].astype(np.int64)

In [286]: df_08.dtypes

Out[286]: model                    object
          displ                   float64
          cyl                       int64
          trans                    object
          drive                    object
          fuel                     object
          veh_class                object
          air_pollution_score     float64
          city_mpg                float64
          hwy_mpg                 float64
          cmb_mpg                 float64
          greenhouse_gas_score      int64
          smartway                 object
          dtype: object
```



```
In [287]: df_18.dtypes

Out[287]: model                    object
          displ                   float64
          cyl                       int64
          trans                    object
          drive                    object
          fuel                     object
          veh_class                object
          air_pollution_score       int64
          city_mpg                float64
          hwy_mpg                 float64
          cmb_mpg                 float64
          greenhouse_gas_score      int64
          smartway                 object
          dtype: object
```

7.Conclusion

To achieve successful data analysis outcomes in the context of car sales using the "Data Analytics with Stock Market Data" Python project, it is essential to first identify the research question or problem to be addressed. This will guide the selection of appropriate data acquisition, preparation, exploration, and wrangling techniques that are relevant to the specific dataset and research question.

Data acquisition may involve gathering relevant car sales data from various sources, such as sales records, customer feedback, or online databases. Data preparation will then involve cleaning, transforming, and formatting the data to ensure that it is in a suitable format for analysis and visualization. Data exploration techniques, such as statistical analysis and data visualization, can then be used to identify patterns, trends, and outliers in the data that are relevant to the research question.

Finally, data wrangling techniques can be applied to merge, manipulate, and reshape the data to prepare it for further analysis. This may include selecting relevant attributes, removing irrelevant information, and transforming the data into a more manageable and structured format.

By applying appropriate data analysis techniques, such as regression analysis, clustering, or decision trees, and effectively reporting the findings, valuable insights can be extracted that can guide decision-making and enhance the overall business value of the analysis. For example, insights may include identifying factors that impact car sales, such as price, location, or car features, or identifying customer preferences or behaviors that can inform marketing strategies.

Throughout the data analysis process, numerous observations and insights can be derived from the dataset. These insights can be effectively communicated through data visualizations, such as scatterplots, histograms, and other visualization tools, which can help to convey the findings in a clear and actionable manner. Ultimately, the success of data analysis in the context of car sales using the "Data Analytics with Stock Market Data" Python project depends on the effective implementation of all these steps and the ability to communicate the insights obtained in a clear and actionable manner.

8.Future Enhancements

There are several potential future enhancements for Stock Market Data analysis that could further improve its effectiveness and value for businesses in the automotive industry. Some of these include:

1. Integration of unstructured data: In addition to structured data, such as vehicle characteristics and stock prediction ratings, unstructured data such as customer reviews and social media feeds can also provide valuable insights into customer preferences and behaviors. Integrating unstructured data sources into Stock Market Data analysis could provide a more comprehensive understanding of customer sentiment and preferences.
2. Real-time analysis: Real-time data analysis can provide businesses with up-to-the-minute insights into stock prediction trends, customer behavior, and other key metrics. By leveraging technologies such as artificial intelligence and machine learning, Stock Market Data analysis could provide even more powerful insights and support decision-making in real- time.
3. Improved predictive modeling: Predictive modeling can be used to forecast future stock prediction trends and identify potential opportunities and risks. Future enhancements to predictive modeling could include the integration of external factors such as weather data, traffic patterns, and stocks prices to improve accuracy.
4. Personalization: By leveraging customer data and machine learning, Stock Market Data analysis could provide personalized recommendations to customers based on their vehicle preferences and behavior. This could help businesses increase customer engagement and loyalty.
5. Integration of blockchain technology: The use of blockchain technology could provide a more secure and transparent way of storing and sharing Stock Market Data, improving data quality and reliability while reducing the risk of fraud.

Overall, these enhancements and others hold great potential to further improve the effectiveness and value of Stock Market Data analysis for businesses in the automotive industry.

9.Bibliography

Aggarwal, R., & Wu, G. (2006). Stock market manipulations. Journal of Business, 79(4), 1915–1954. Aitken, M. J. in The Australian. (2012 July, 17). Smaller stocks make ASX innefficient says academic. Retrieved from http://www.theaustralian.com.au/ business/wealth/smaller-stocks-make-asx-inefficient-says-academic/storye6frgac6-1226427548117.

Accessed on July 28, 2012. Aitken, M. J., & Siow, A. S. (2003), "Ranking world equity markets on the basis of market efficiency and integrity. In H. Skeete (Ed.), The HP handbook of world stock, derivative & commodity exchanges 2003 (pp. xlix–lv). New York: Mondo VisioneLtd. Aitken, M. J., Harris, F. H., & Ji, S. (2009a).

Trade-Based manipulation and market efficiency: A cross-market comparison. Paper presented at 22nd Australasian Finance and Banking Conference, November 18, 2009, Sydney, Australia. Retrieved from http://www.cmcrc.com/index.php/rd/academicpapers/academic-papers. Accessed on September 30, 2011. Aitken, M. J., Rowan, M. C., deB. Harris, F. H., & McInish, T. H. (2009b). Market design and execution cost for matched securities worldwide. Working Paper Babcock School, Wake Forest University. Aktas, R., & Doganay, M. M. (2006).

Stock-Price manipulation in the ˘ Istanbul stock exchange. Eurasian Review of Economics and Finance, 2(1), 21–28. Allen, F., & Gale, D. (1992). Stock price manipulation. Review of Financial Studies, 5(3), 503–529