**DATA PREDICTION MODEL**

**(BASED ON DATA SCIENCE)**

**MINOR PROJECT REPORT**

***Submitted in partial fulfillment of th*e *requirements for the award of the degree***

***Of***

**BACHELOR OF TECHNOLOGY**

***In***

**MECHANICAL AND AUTOMATION ENGINEERING**

***By***

**Mohd Faizan Gaurav Akshit Gupta Dhruv Parasher**

00696203617 41196203617 41296203617 41596203617

“F-16” “F-16” “F-16” “F16”

***Guided by***

***Dr. Deepak Bhardwaj***

 **(Dept. of Mechanical and Automation Engineering)**

**{Department of Mechanical and Automation Engineering}**

**DR. AKHILESH DAS GUPTA INSTITUTE OF TECHNOLOGY AND MANAGEMENT** (AFFILIATED TO GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY)

**NEW DELHI – 110078.**

**[AUG-DEC 2020]**

# ABSTRACT

Predictive analytics is a term mainly used in statistical and analytics techniques. This term is drawn from statistics, machine learning,   
 database techniques and optimization techniques. It predicts the future   
by analyzing current and historical data. The future events and behaviour of variables can be predicted using the models of predictive analytics.

Nowadays MNCs and Big Marts keep the track of their sales data of each and every individual item for predicting future demand of the customer and update the inventory management as well. These data stores basically contain a large number of customer data and individual item attributes in a data warehouse. Further, anomalies and frequent patterns are detected by mining the data store from the data warehouse. The resultant data can be used for predicting future sales volume with the help of different machine learning techniques for the MNCs. In this paper, we propose a predictive model using Multi-Linear Regression technique for predicting the sales of automobile company and forecast the estimated value of their respective models w.r.t year.



### Department of Mechanical & Automation Engineering

**Dr. Akhilesh Das Gupta Institute of Technology & Management Delhi-53**

**Certificate**

It is certified that the work contained in this report titled *“****DATA PREDICTION MODEL*** “is the original work done by Mohd. Faizan *(00696203617)*, Gaurav *(41196203617)*, Akshit Gupta *(41296203617)* Dhruv Parasher *(41596203617) and* has been carried out under our supervision.

***Dr. Deepak Bhardwaj Mr. Neeraj Kumar***

### (Project Supervisor) (Head of department)

**Signature of External Examiner: …………………**

**ACKNOWLEDGEMENT**

We express our deep gratitude to **Dr.Deepak Bhardwaj**, Associate Professor, Department of Mechanical and Automation Engineering for his valuable guidance and suggestion throughout our project work.

We would like to extend our sincere thanks to **Dr.Deepak Bhardwaj** for his time to time suggestion to complete our project work. We are also thankful to **Prof. (Dr.) Sanjay Kumar (Director)** for providing us the facilities to carry out our project work.

Sign:Sign:

**Mohd. Faizan Gaurav**

“00696203617” “41196203617”

Sign: Sign:

**Akshit Gupta Dhruv Parasher**

“41296203617” “41596203617”

**Table of Content**

* [Abstract](#_TOC_250022) 2
* [Certificate](#_TOC_250021) 3
* Acknowledgment 4
* Table of Content 5
* Introduction 6
* Objective 7
* Theory 7 -10
* Data Science
* Data Discovery
* Data Preparation
* Mathematical Model

### ****Getting Things in Action****

* Communication
* Data Science Components
* Methodology
* Program 11-37

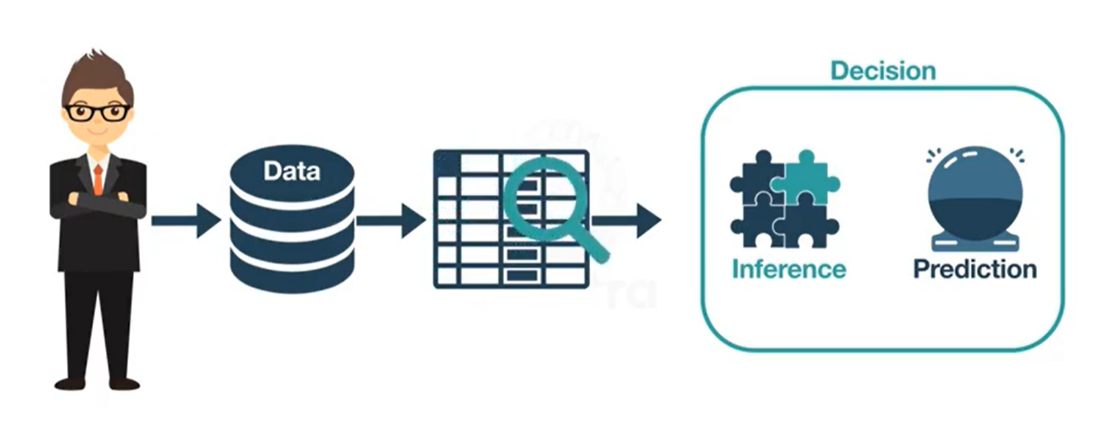
1. Cars Price Prediction
2. California Housing

* Advantages & Disadvantages 38
* Application 39
* Result 40
* Conclusion 41
* Future Scope 42
* References 43

# 

# INTRODUCTION

Data science is all about using data to solve problems. The problem could be decision making such as identifying which email is spam and which is not.



Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data. Data Science is primarily used to make decisions and predictions making use of predictive causal analytics, prescriptive analytics (predictive plus decision science) and machine learning.

* **Predictive causal analytics –**If you want a model that can predict the possibilities of a particular event in the future, you need to apply predictive causal analytics.
* **Prescriptive analytics –**If you want a model that has the intelligence of taking its own decisions and the ability to modify it with dynamic parameters, you certainly need prescriptive analytics for it. This relatively new field is all about providing advice.
* **Machine learning for making predictions** — If you have transactional data of a finance company and need to build a model to determine the future trend, then machine learning algorithms are the best bet.
* **Machine learning for pattern discovery** — If you don’t have the parameters based on which you can make predictions, then you need to find out the hidden patterns within the dataset to be able to make meaningful predictions. This is nothing but the unsupervised model as you don’t have any predefined labels for grouping.

# 

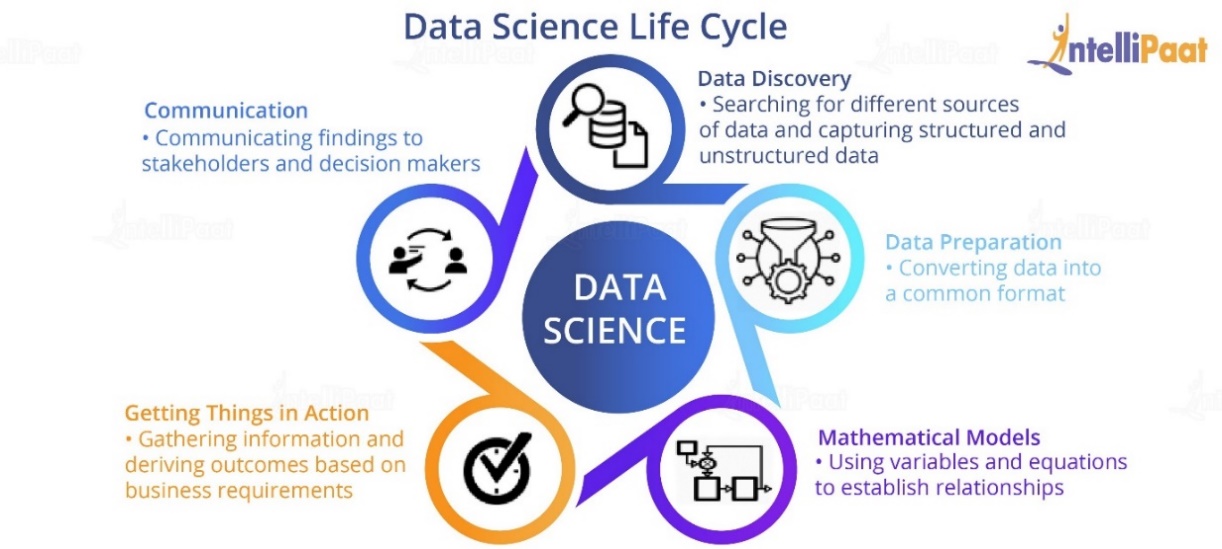
# Objective

# Predictive analytics is the use of data, statistical algorithms and machine learning techniques to identify the likelihood of future outcomes based on historical data. The goal is to go beyond knowing what has happened to providing a best assessment of what will happen in the future .The main objective of data prediction model is to use current and historical data to make predictions about future customer behavior, otherwise unknown events, risks, trends and opportunities.

# THEORY

# Data Science

# Data Science is a blend of various tools, algorithms, and machine learning principles with the goal to discover hidden patterns from the raw data.



* **Data Discovery**

The first phase in the Data Science life cycle is data discovery for any Data Science problem. It includes ways to discover data from various sources which could be in an unstructured format like videos or images or in a structured format like in text files, or it could be from relational database systems. Organizations are also peeping into customer social media data, and the like, to understand customer mindset better.

### ****Data Preparation****

Once the data discovery phase is completed, the next stage is data preparation. It includes converting disparate data into a common format in order to work with it seamlessly. This process involves collecting clean data subsets and inserting suitable defaults, and it can also involve more complex methods like identifying missing values by modelling, and so on. Once the data cleaning is done, the next step is to integrate and create a conclusion from the dataset for analysis. This involves the integration of data which includes **merging** two or more tables of the same objects, but storing different information, or summarizing fields in a table using **aggregation**. Here, we would also try to explore and understand what patterns and values our datasets have.

### ****Mathematical Models****

All [**Data Science projects**](https://intellipaat.com/blog/data-science-project-ideas/) have certain mathematical models driving them. These models are planned and built by the Data Scientists in order to suit the specific need of the business organization. This might involve various areas of the mathematical domain including statistics, logistic and linear regression, differential and integral calculus, etc. Various tools and apparatus used in this regard could be R statistical computing tools, [**Python programming language**](https://intellipaat.com/blog/tutorial/python-tutorial/what-is-python/), [**SAS advanced analytical tools**](https://intellipaat.com/blog/tutorial/sas-tutorial/introduction-to-sas/), [**SQL**](https://intellipaat.com/blog/tutorial/sql-tutorial/introduction-to-sql/), and various data visualization tools like [**Tableau**](https://intellipaat.com/blog/what-is-tableau/) and **[QlikView](https://intellipaat.com/blog/tutorial/qlikview-tutorial/introduction/" \t "_blank)**.

### ****Getting Things in Action****

### Once the data is prepared and the models are built, it is time to get these models working in order to achieve the desired results. There might be various discrepancies and a lot of troubleshooting that might be needed, and thus the model might have to be tweaked. Here, model evaluation explains the performance of the model.

* **Communication**

Communicating the findings is the last but not the least step in a Data Science endeavour. In this stage, the Data Scientist needs to be a liaison between various teams and should be able to seamlessly communicate his findings to key stakeholders and decision-makers in the organization so that actions can be taken based on the recommendations of the Data Scientist.

## ****Data Science Components****

Key components of Data Science, which are:

* **Data (and Its Various Types)**

The raw dataset is the foundation of Data Science, and it can be of various types like structured data (mostly in a tabular form) and unstructured data (images, videos, emails, PDF files, etc.)

* **Programming (Python and R)**

Data management and analysis is done by computer programming.      
In Data Science, two programming languages are most popular:   
Python and R.

* **Statistics and Probability**

Data is manipulated to extract information out of it. The mathematical foundation of Data Science is statistics and probability. Without having a clear knowledge of statistics and probability, there is a high possibility of misinterpreting data and reaching at incorrect conclusions. That’s the reason why statistics and probability play a crucial role in Data Science.

* **Machine Learning**

As a Data Scientist, every day, you will be using Machine Learning algorithms such as regression and classification methods. It is very important for a Data Scientist to know [Machine learning](https://intellipaat.com/blog/what-is-machine-learning/) as a part of their job so that they can predict valuable insights from available data.

# METHODOLOGY

# Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The goal of multiple linear regression (MLR) is to model the [linear relationship](https://www.investopedia.com/terms/l/linearrelationship.asp) between the explanatory (independent) variables and response (dependent) variable.

# Multiple Linear Regression - Formula

# where, for i=n observations:

# = dependent variable

# = explanatory variables

# ​ = y-intercept (constant term)

# ​= slope coefficients for each explanatory variable

# ϵ = the model’s error term (also known as the residuals)

# In a Data Science model, Multiple Linear Regression attempts to model the relationship between two or more features and a response by fitting a linear equation to observed data.

**PROGRAMS**

**1. Cars Price Prediction:-**

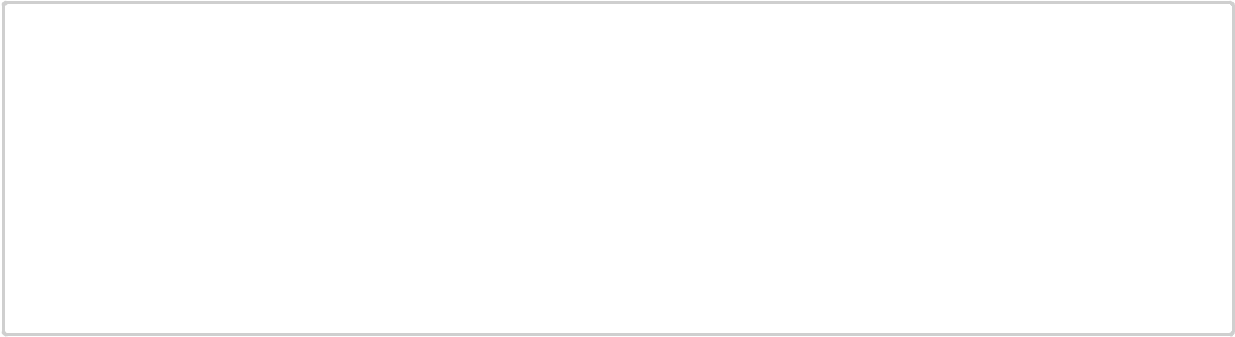
The goal of this notebook is to predict the prices of used cars in India. Three important steps involved in this notebook are: Data cleaning/ Feature engineering. Exploratory Data Analysis. Predicting the price of car using price using Machine Learning.

Source of Data - [www.kaggle.com](file:///F:\dhruv\DATA_SCIENCE%20PROJECT\www.kaggle.com)

Kaggle, a subsidiary of Google LLC, is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges

**Read dataset**

I'll read the dataset and get information about it



In [1]: **import pandas as pd**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**from matplotlib import** style

**import sklearn**

**from pandas import** Series

**from pandas import** DataFrame

**import matplotlib.pyplot as plt**



In [2]: data = pd.read\_csv("C:**\\**Users\mohdf\OneDrive\Desktop\Automobile.csv")



In [3]: data.head(5)

Out[3]:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **car** | **price** | **body** | **mileage** | **engV** | **engType** | **registration** | **year** | **model** | **drive** |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **0** | Ford | 15500.0 | crossover | 68 | 2.5 | Gas | yes | 2010 | Kuga | full |  |
| **1** | Mercedes- | 20500.0 | sedan | 173 | 1.8 | Gas | yes | 2011 | E- | rear |  |
| Benz | Class |  |
| **2** | Mercedes- | 35000.0 | other | 135 | 5.5 | Petrol | yes | 2008 | CL | rear |  |
| Benz | 550 |  |
| **3** | Mercedes- | 17800.0 | van | 162 | 1.8 | Diesel | yes | 2012 | B 180 | front |  |
| Benz |  |
| **4** | Mercedes- | 33000.0 | vagon | 91 | NaN | Other | yes | 2013 | E- | NaN |  |
| Benz | Class |  |

**CHECKING TO NUMBER OF UNIQUE VALUES**



|  |  |
| --- | --- |
| In [4]: data.nunique() |  |
| Out[4]: car | 87 |
| price | 1353 |
| body | 6 |
| mileage | 442 |
| engV | 117 |
| engType | 4 |
| registration | 2 |
| year | 56 |
| model | 888 |
| drive | 3 |
| dtype: int64 |  |

**CONVERTING TEXT DATA INTO NUMERIC FORM USING LABEL ENCODER AND ONEHOT ENCODER**



In [5]: **from sklearn import** reprocessing

le = reprocessing.LabelEncoder()

data['registration'] = le.fit\_transform(data['registration'])

print(le.classes\_)

['no' 'yes']

In [6]: data.columns

Out[6]: Index(['car', 'price', 'body', 'mileage', 'engV', 'engType', 'registration', 'year', 'model', 'drive'],

dtype='object')



In [7]: data = pd.get\_dummies(data,columns = ["body","engType","drive"])

In [8]: data.head(10)

Out[8]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **car** | **price** | **mileage** | **engV** | **registration** | **year** | **model** | **body\_crossover body\_hatch** | |  |
|  |  | |  |  |  |  |  |  |  |  |  |  |
| **0** | | | Ford | 15500.0 | 68 | 2.5 | 1 | 2010 | Kuga | 1 | 0 |  |
| **1** | | | Mercedes- | 20500.0 | 173 | 1.8 | 1 | 2011 | E-Class | 0 | 0 |  |
| Benz |  |
| **2** | | | Mercedes- | 35000.0 | 135 | 5.5 | 1 | 2008 | CL 550 | 0 | 0 |  |
| Benz |  |
| **3** | | | Mercedes- | 17800.0 | 162 | 1.8 | 1 | 2012 | B 180 | 0 | 0 |  |
| Benz |  |
| **4** | | | Mercedes- | 33000.0 | 91 | NaN | 1 | 2013 | E-Class | 0 | 0 |  |
| Benz |  |
| **5** | | | Nissan | 16600.0 | 83 | 2.0 | 1 | 2013 | X-Trail | 1 | 0 |  |
| **6** | | | Honda | 6500.0 | 199 | 2.0 | 1 | 2003 | Accord | 0 | 0 |  |
| **7** | | | Renault | 10500.0 | 185 | 1.5 | 1 | 2011 | Megane | 0 | 0 |  |
| **8** | | | Mercedes- | 21500.0 | 146 | 1.8 | 1 | 2012 | E-Class | 0 | 0 |  |
| Benz |  |
| **9** | | | Mercedes- | 22700.0 | 125 | 2.2 | 1 | 2010 | E-Class | 0 | 0 |  |
| Benz |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

**CHECKING TOTAL NUMBER OF NAN VALUES IN A GIVEN DATA SET**



In [9]: data.isnull().head(10)

Out[9]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **car** | **price** | **mileage** | **engV** | **registration** | | **year** | **model** | **body\_crossover body\_hatch body\_o** | | |
|  |  | |  |  |  |  |  | |  |  |  |  |  |
| **0** | | | False | False | False | False | False | | False | False | False | False | F |
| **1** | | | False | False | False | False | False | | False | False | False | False | F |
| **2** | | | False | False | False | False | False | | False | False | False | False | F |
| **3** | | | False | False | False | False | False | | False | False | False | False | F |
| **4** | | | False | False | False | True | False | | False | False | False | False | F |
| **5** | | | False | False | False | False | False | | False | False | False | False | F |
| **6** | | | False | False | False | False | False | | False | False | False | False | F |
| **7** | | | False | False | False | False | False | | False | False | False | False | F |
| **8** | | | False | False | False | False | False | | False | False | False | False | F |
| **9** | | | False | False | False | False | False | | False | False | False | False | F |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |





In [10]: data.isnull().sum()

|  |  |
| --- | --- |
| Out[10]: car | 0 |
| price | 0 |
| mileage | 0 |
| engV | 434 |
| registration | 0 |
| year | 0 |
| model | 0 |
| body\_crossover | 0 |
| body\_hatch | 0 |
| body\_other | 0 |
| body\_sedan | 0 |
| body\_vagon | 0 |
| body\_van | 0 |
| engType\_Diesel | 0 |
| engType\_Gas | 0 |
| engType\_Other | 0 |
| engType\_Petrol | 0 |
| drive\_front | 0 |
| drive\_full | 0 |
| drive\_rear | 0 |
| dtype: int64 |  |

**DROPPING ALL ROWS HAVING NAN VALUES**



In [11]: data =data.dropna()

**IMPORTING OUR CLEAN DATA SET**



In [12]: data32= data.to\_csv("C:**\\**Users\mohdf\OneDrive\Documents\data32.csv")



In [13]: data32

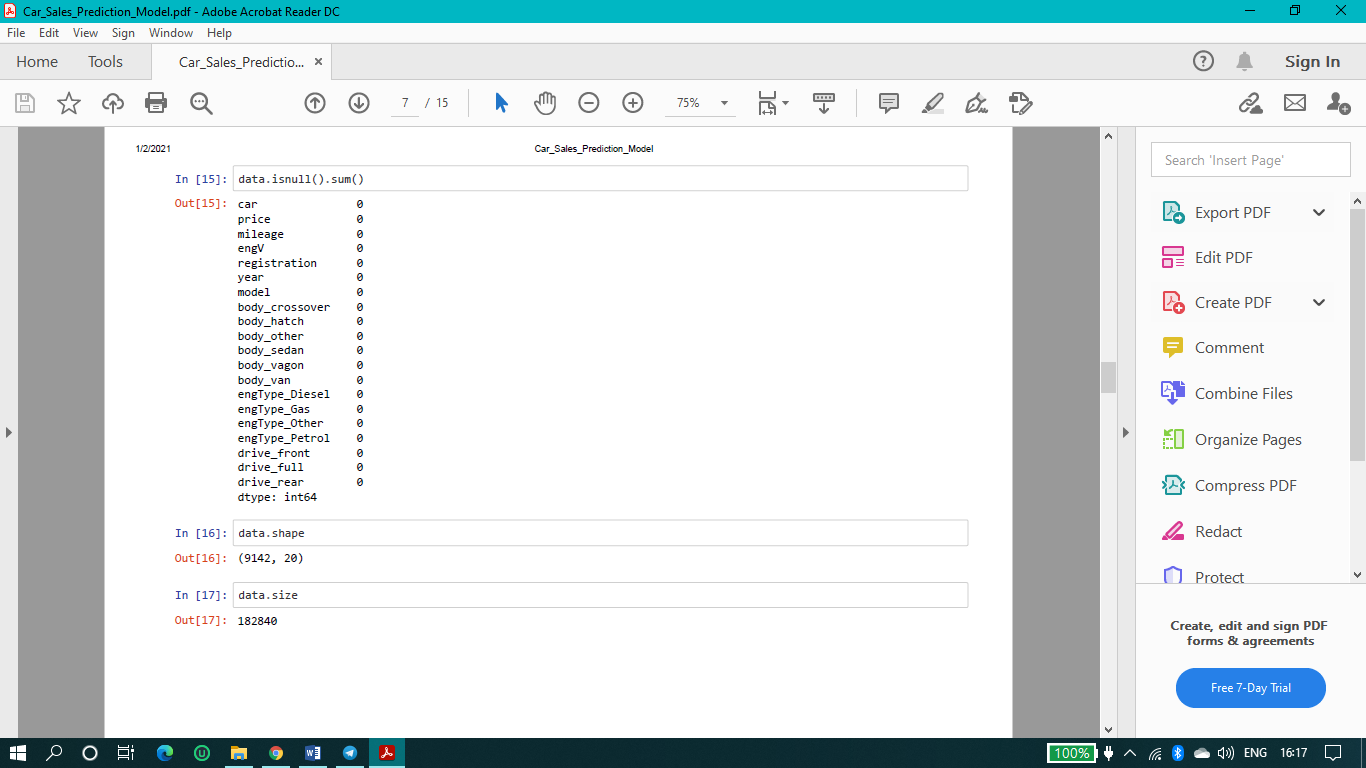
**GETTING MORE INFORMATION FROM OUR DATA SET**

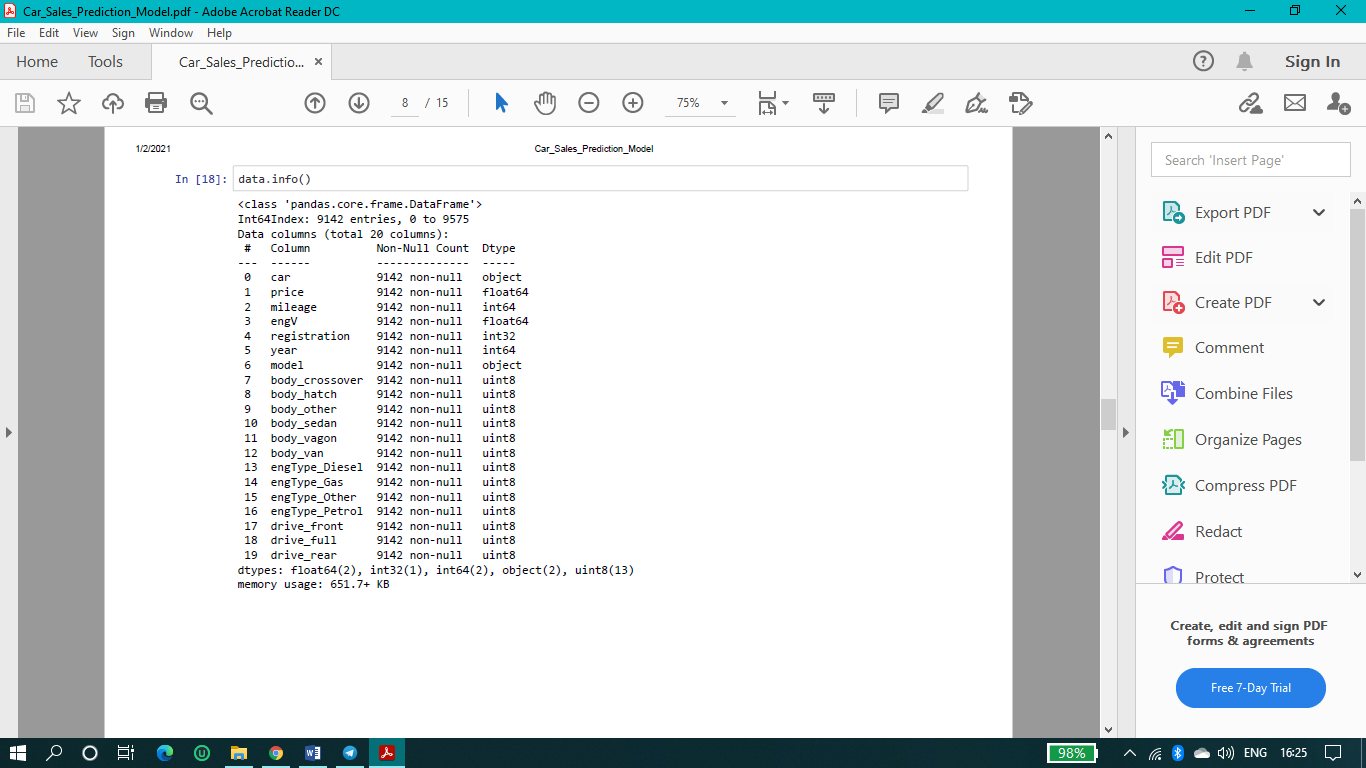
In [14]: data.head(10)

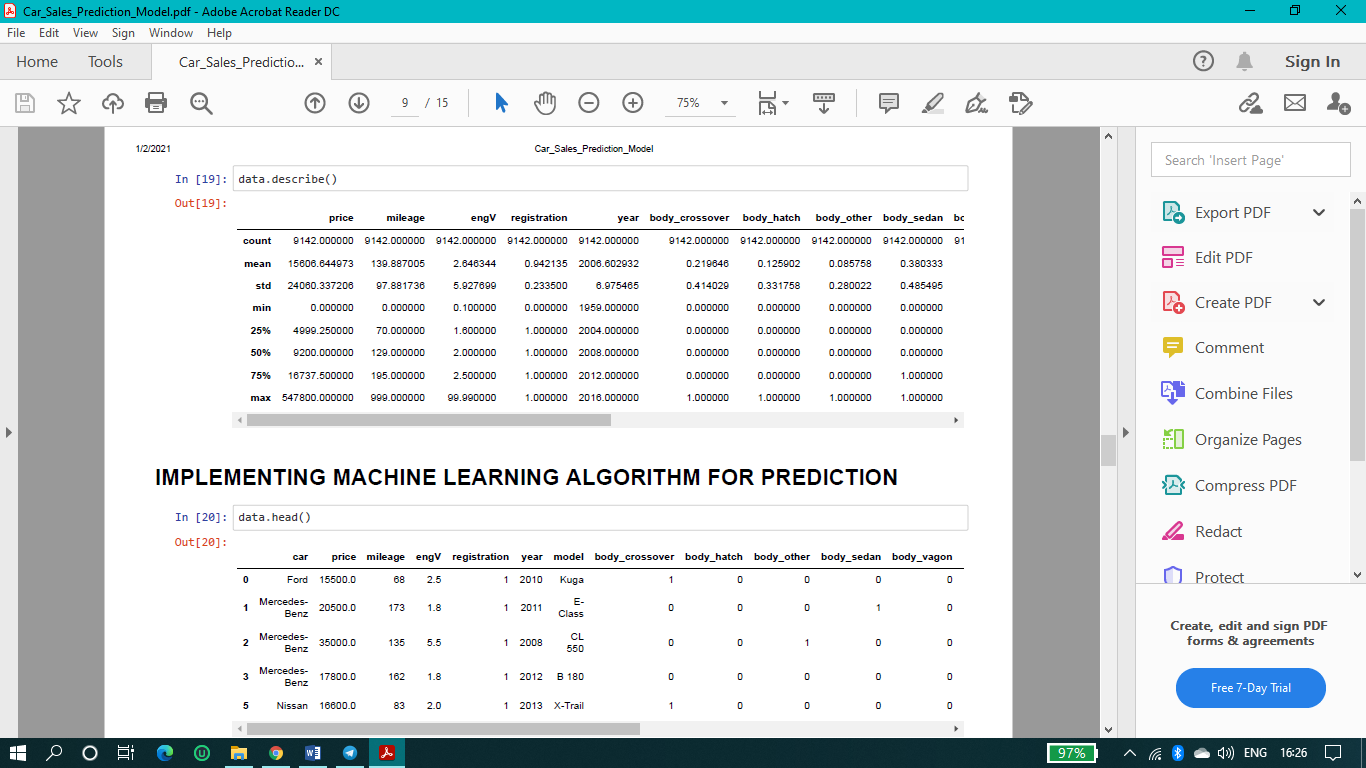
Out[14]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **car** | **price** | **mileage** | **engV** | | **registration** | **year** | **model** | **body\_crossover body\_hat** | |  |
|  |  | |  |  |  |  | |  |  |  |  |  |  |
| **0** | | | Ford | 15500.000 | 68 | 2.5 | | 1 | 2010 | Kuga | 1 |  |  |
| **1** | | | Mercedes- | 20500.000 | 173 | 1.8 | | 1 | 2011 | E-Class | 0 |  |  |
| Benz |  |  |
| **2** | | | Mercedes- | 35000.000 | 135 | 5.5 | | 1 | 2008 | CL 550 | 0 |  |  |
| Benz |  |  |
| **3** | | | Mercedes- | 17800.000 | 162 | 1.8 | | 1 | 2012 | B 180 | 0 |  |  |
| Benz |  |  |
| **5** | | | Nissan | 16600.000 | 83 | 2.0 | | 1 | 2013 | X-Trail | 1 |  |  |
| **6** | | | Honda | 6500.000 | 199 | 2.0 | | 1 | 2003 | Accord | 0 |  |  |
| **7** | | | Renault | 10500.000 | 185 | 1.5 | | 1 | 2011 | Megane | 0 |  |  |
| **8** | | | Mercedes- | 21500.000 | 146 | 1.8 | | 1 | 2012 | E-Class | 0 |  |  |
| Benz |  |  |
| **9** | | | Mercedes- | 22700.000 | 125 | 2.2 | | 1 | 2010 | E-Class | 0 |  |  |
| Benz |  |  |
| **10** | | | Nissan | 20447.154 | 0 | 1.2 | | 1 | 2016 | Qashqai | 1 |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |

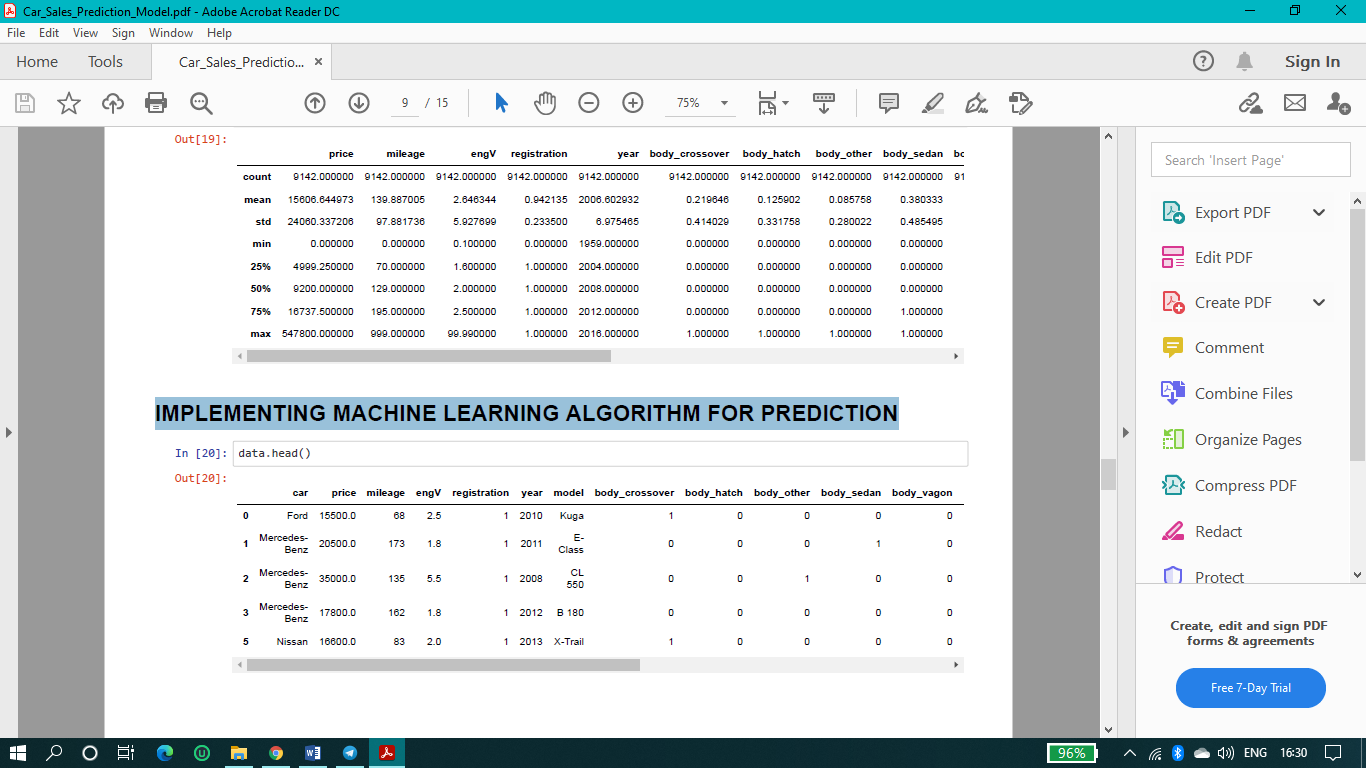


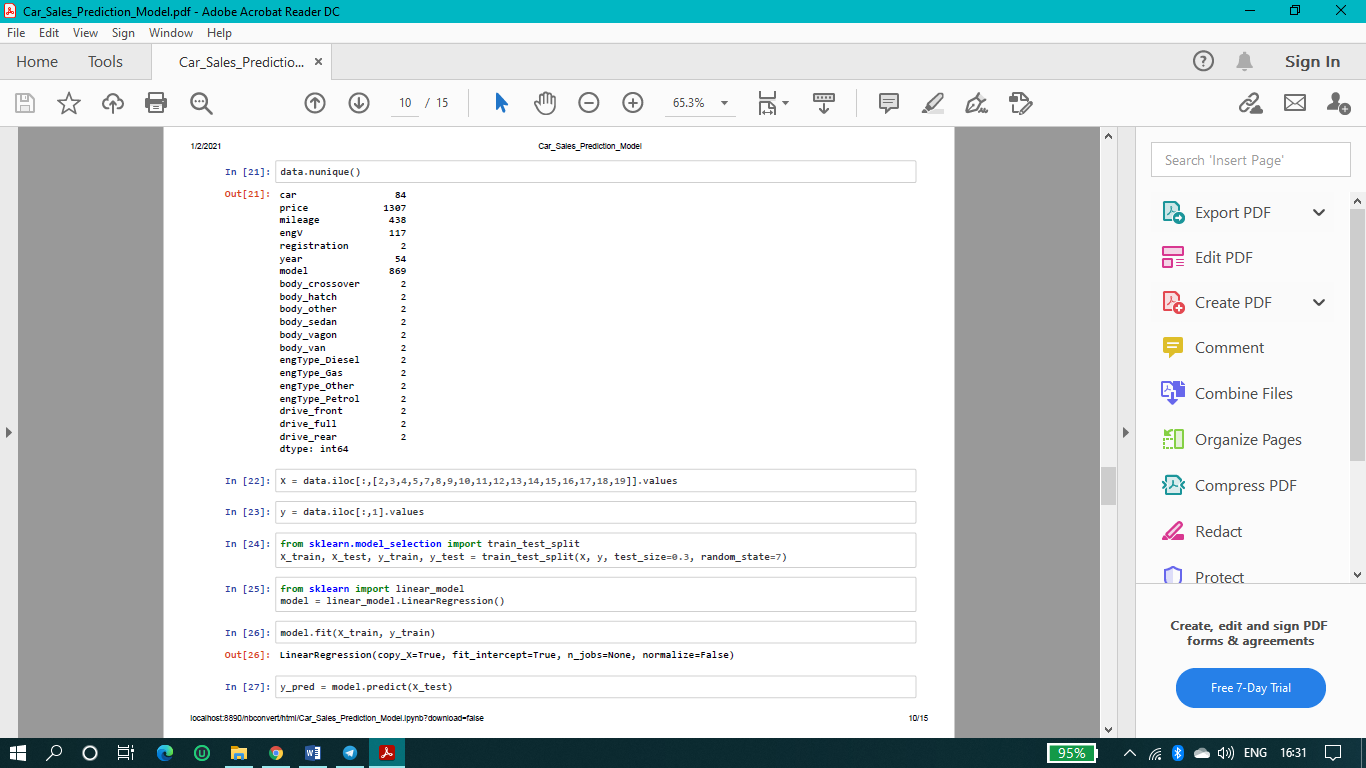


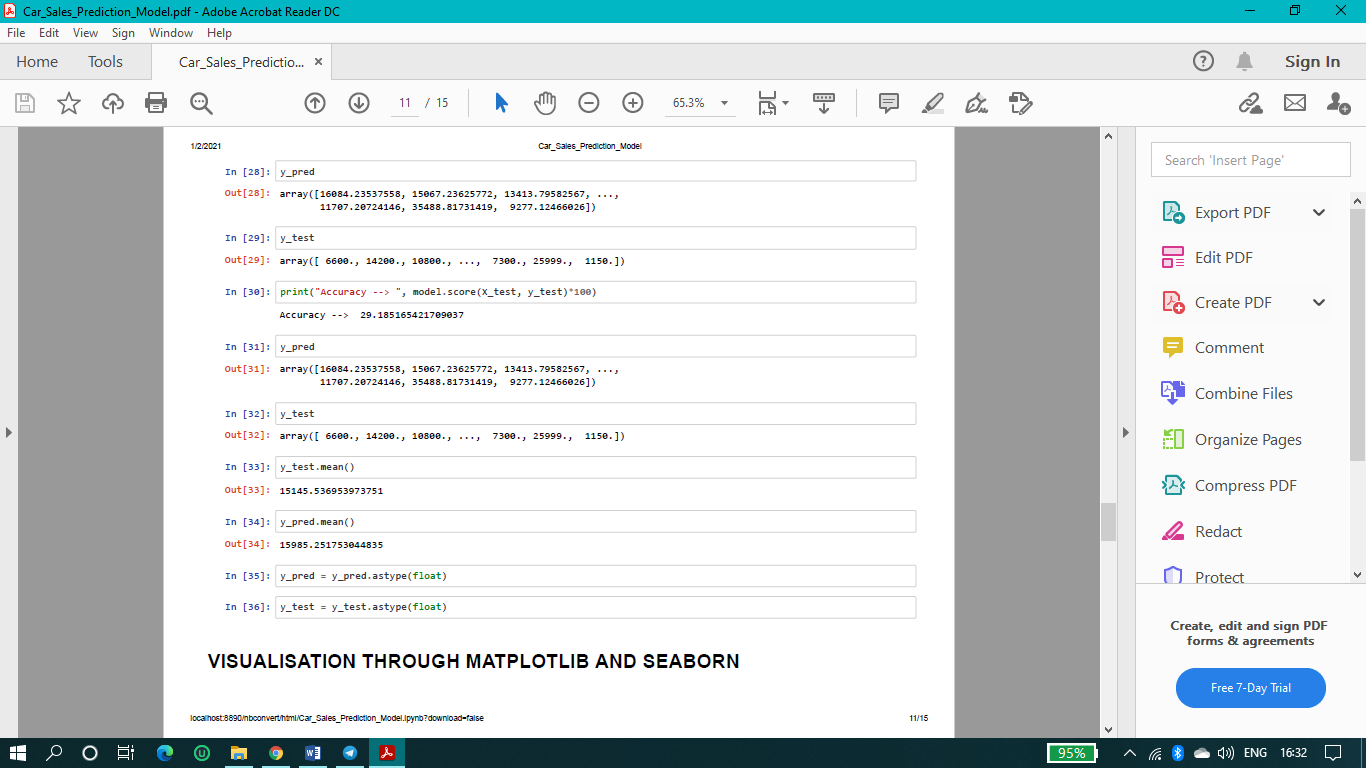




**IMPLEMENTING MACHINE LEARNING ALGORITHM FOR PREDICTION**







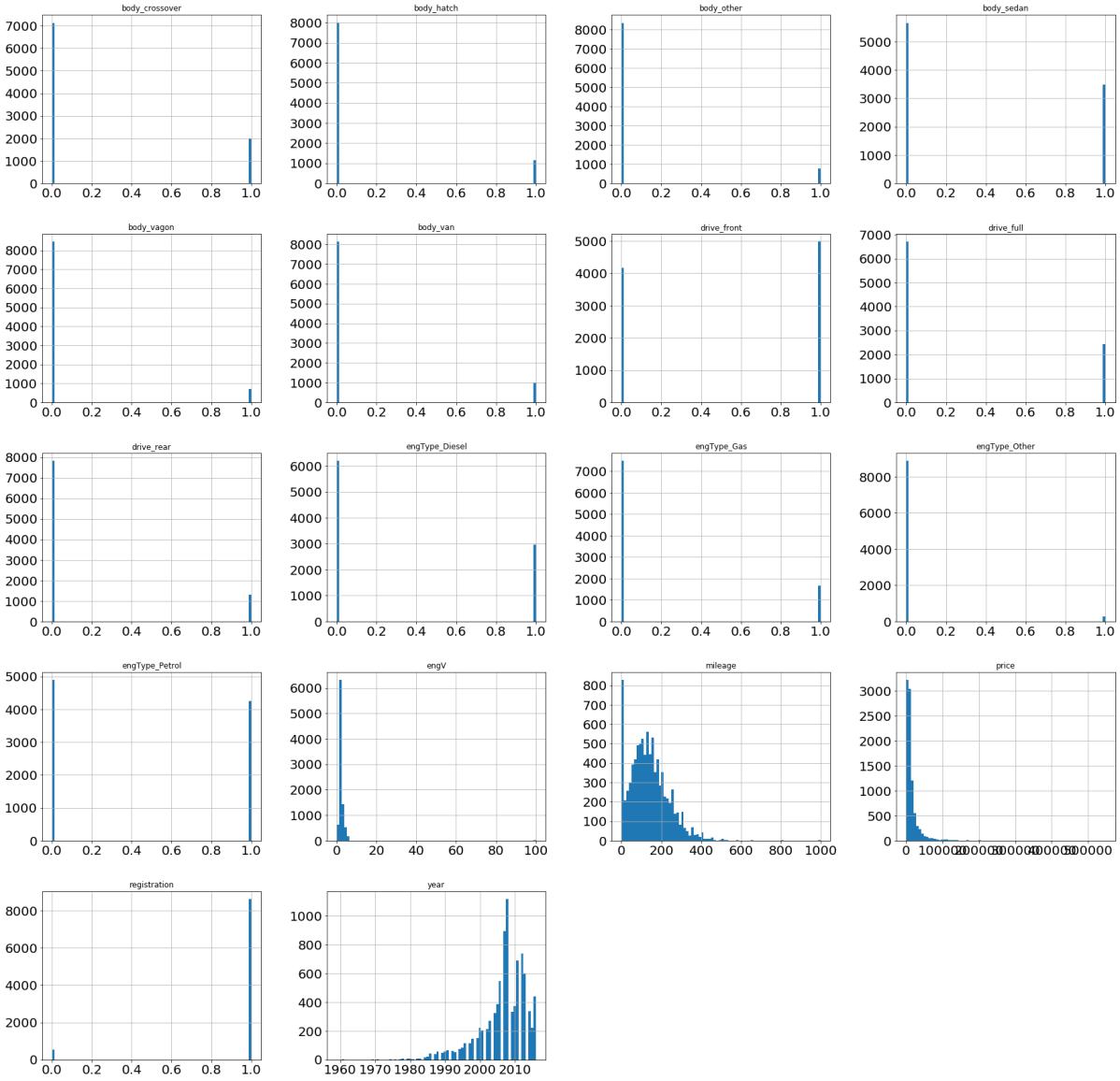
**VISUALISATION THROUGH MATPLOTLIB AND SEABORN**



In [37]: **import matplotlib.pyplot as plt**

data.hist(bins=80, figsize=(30,30),xlabelsize=20,ylabelsize=20,grid=**True**)

plt.show()



**Checking Correlation using heatmap**

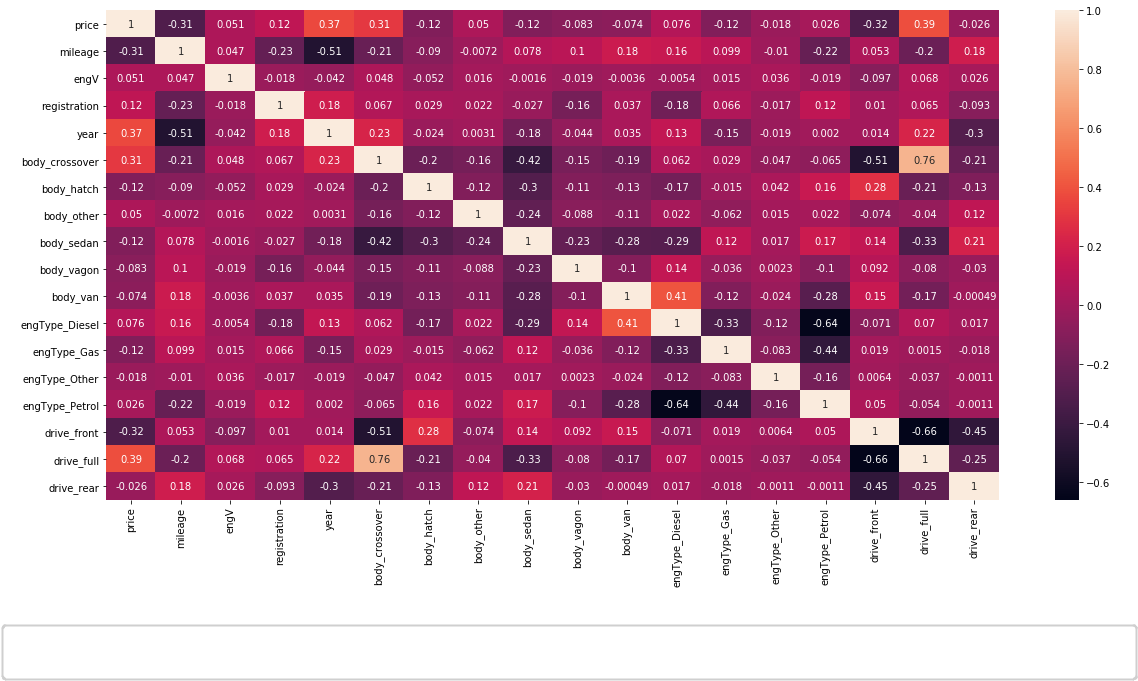


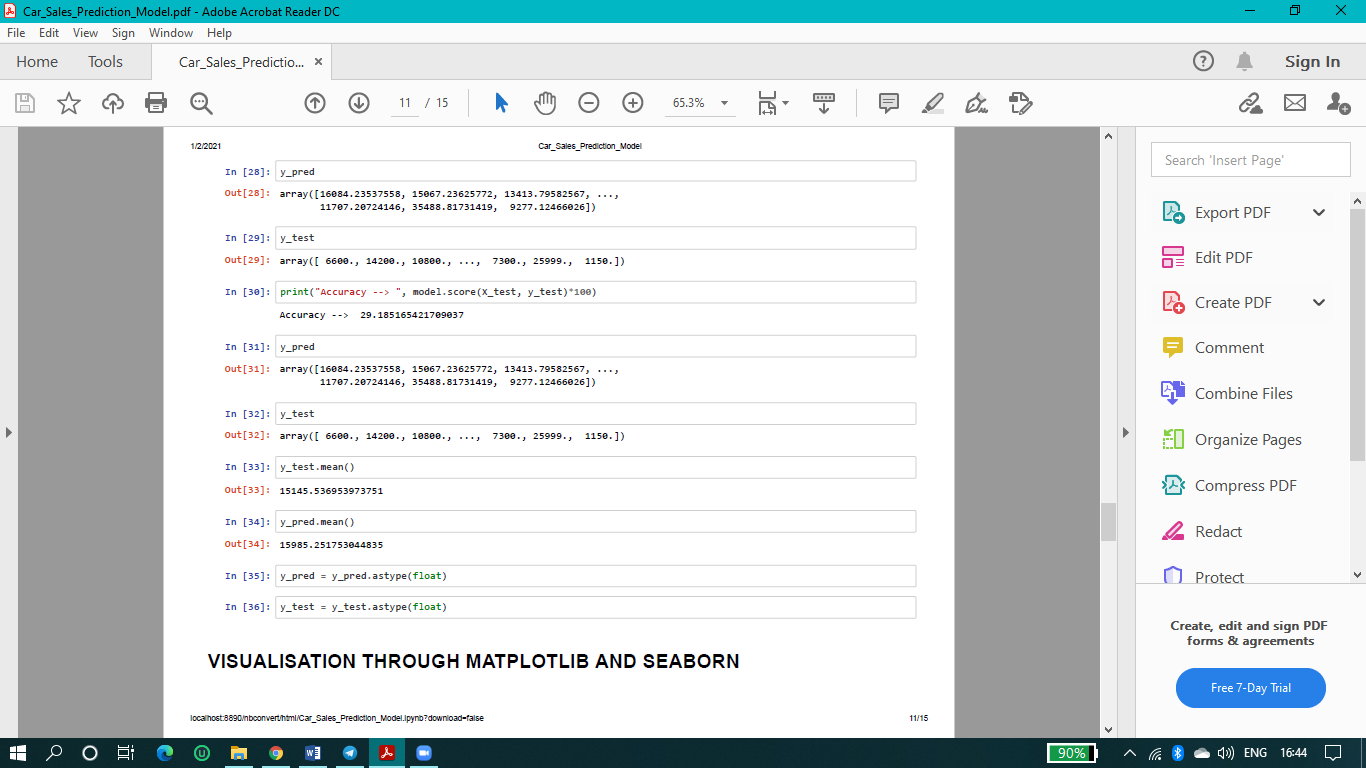
In [38]: corr = data.corr()

plt.subplots(figsize=(20,9))

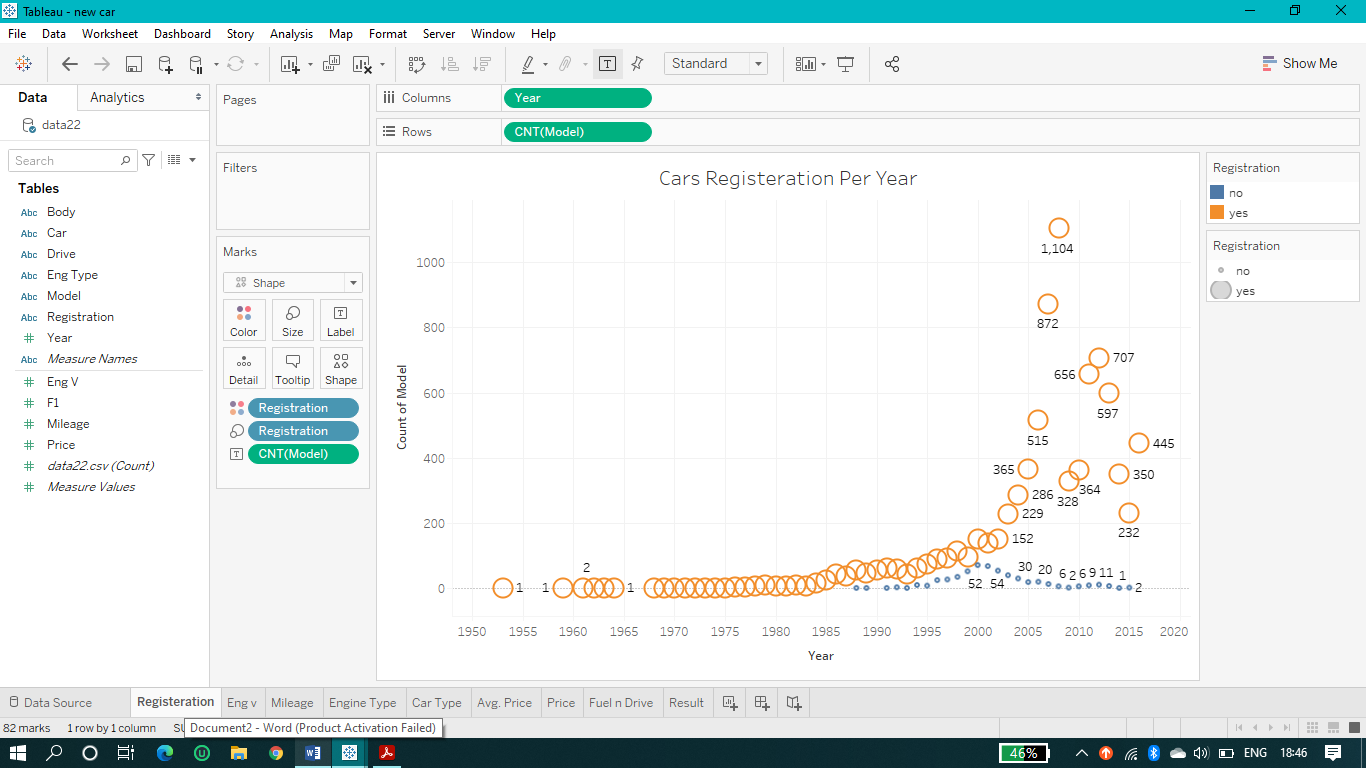
sns.heatmap(corr, annot=**True**)

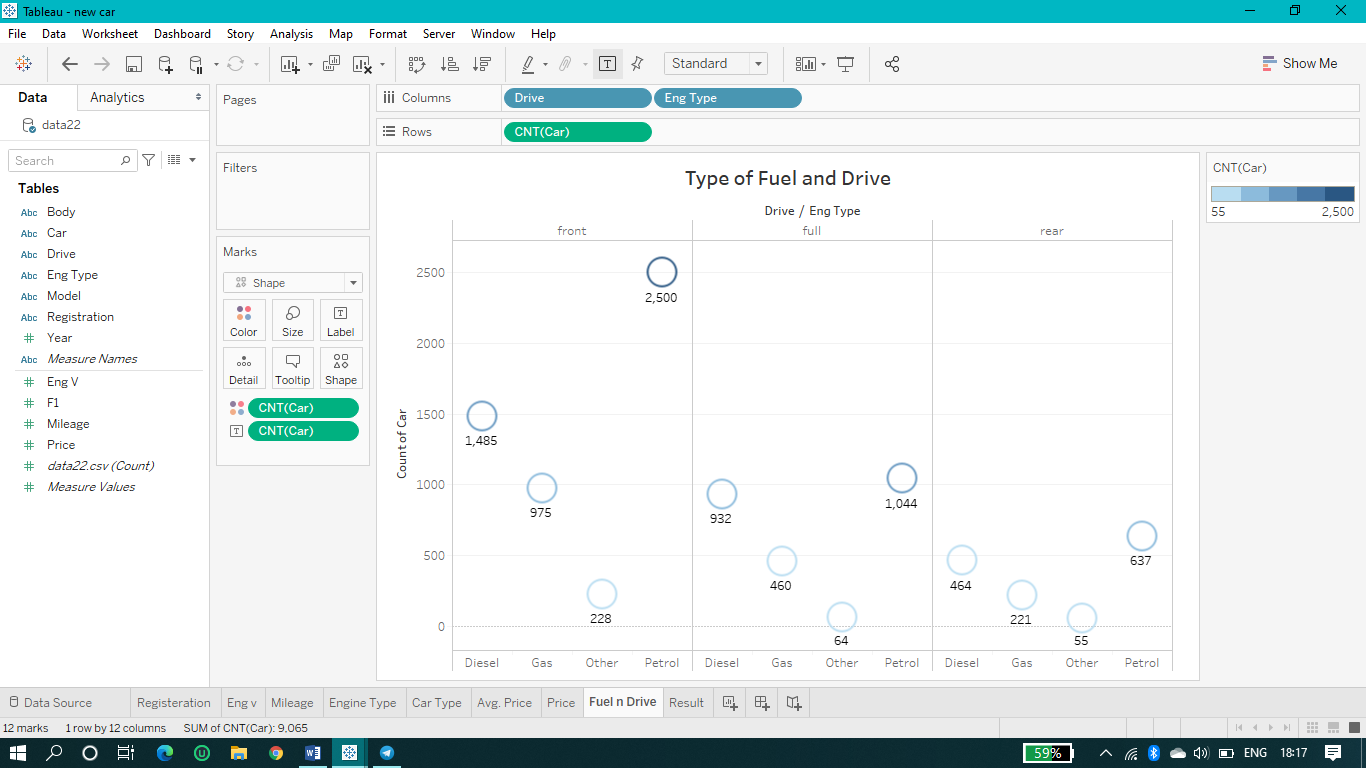
plt.show()

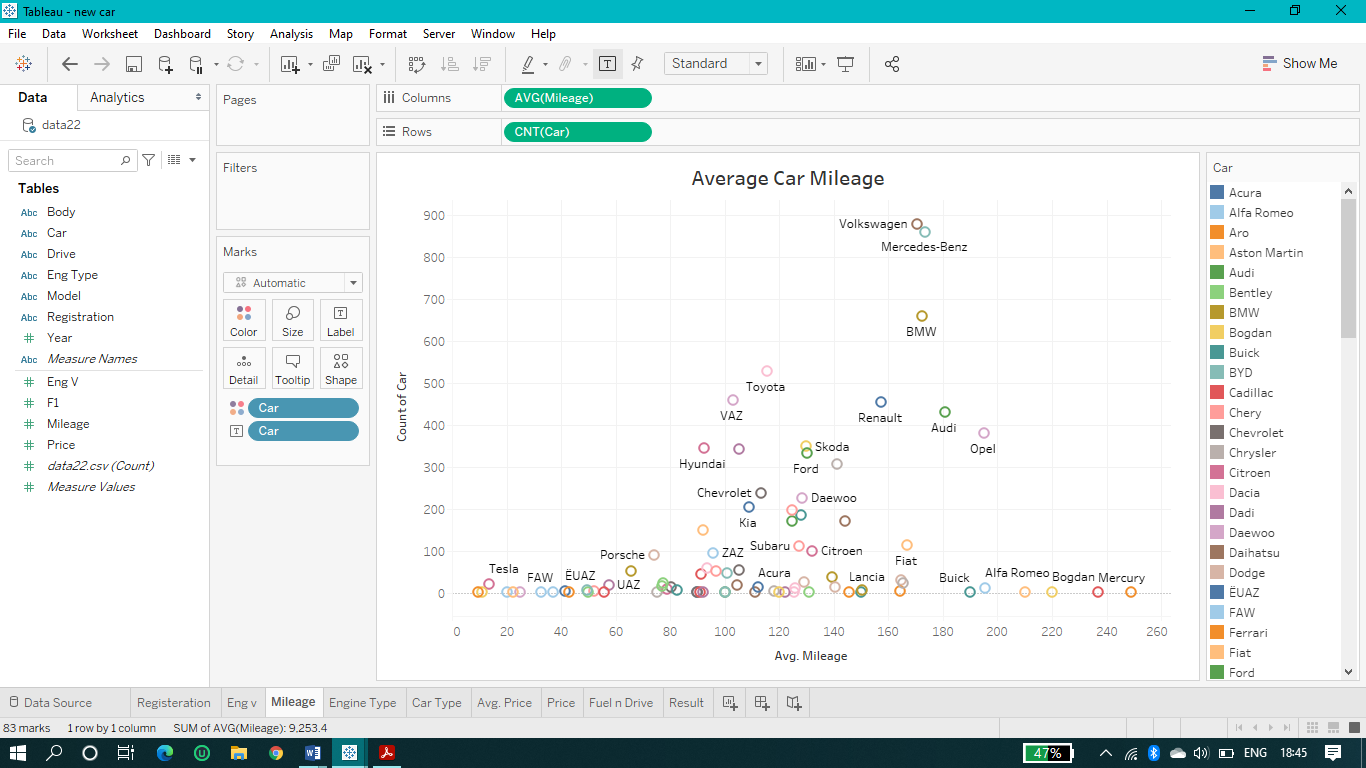


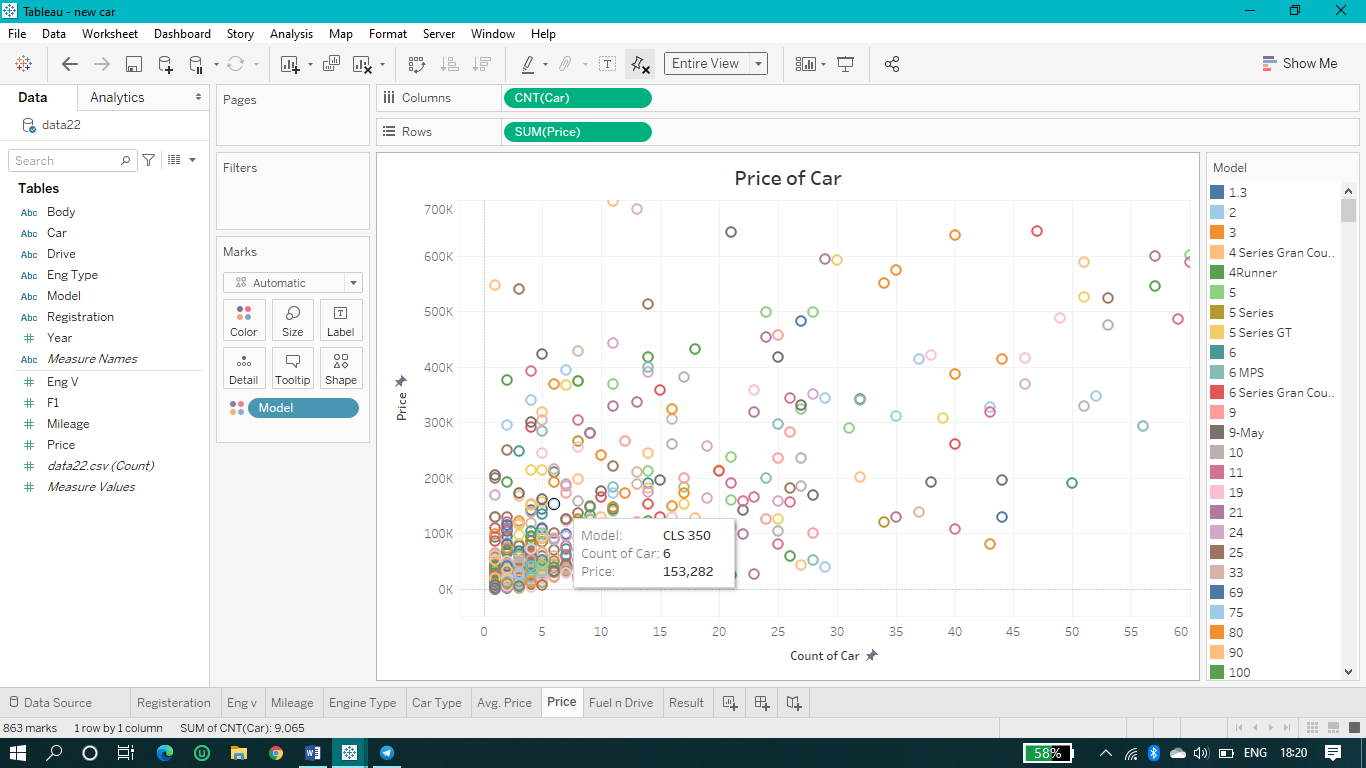


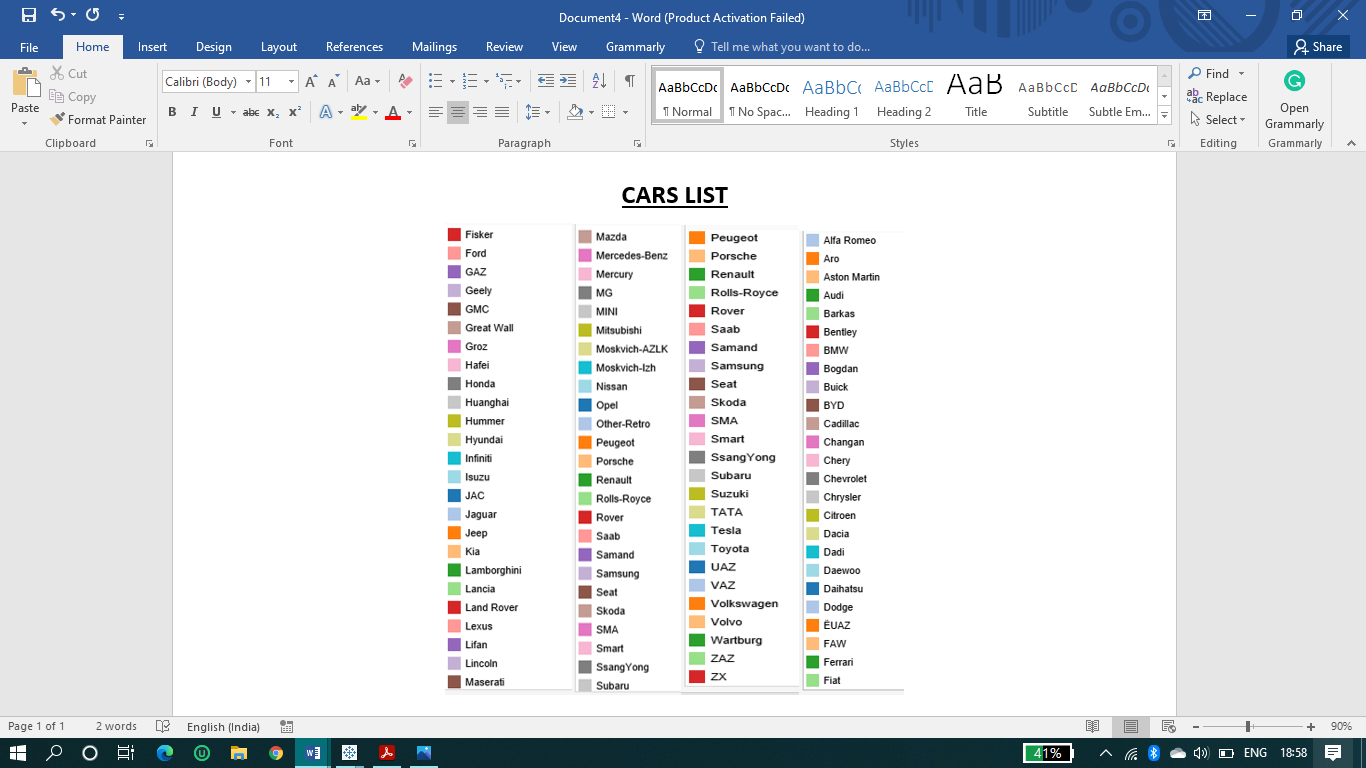
**VISUALIZATION ON TABLEAU**











* 1. **California Housing Prediction :-**

**Read dataset**

I'll read the dataset and get information about it



In [1]: **import pandas as pd**

**import numpy as np**

**from matplotlib import** pyplot **as** plt

**import seaborn as sns**

%**matplotlib** inline



In [2]: housing = pd.read\_csv("D:**\\**housing.csv")



In [3]: housing1 = housing.copy()



In [4]: housing.head()

Out[4]:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **longitude** | **latitude** | **housing\_median\_age total\_rooms** | | **total\_bedrooms population households** | | | | |
|  |  | |  |  |  |  |  | |  |  | |
| **0** | | | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | | 322.0 | 126.0 |
| **1** | | | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | | 2401.0 | 1138.0 | |
| **2** | | | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | | 496.0 | 177.0 | |
| **3** | | | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | | 558.0 | 219.0 | |
| **4** | | | -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | | 565.0 | 259.0 | |
|  |  |  |  |  |  |  |  |  |  |  | |
|  |  |  |  |  |  |  |  |  |  |  | |



**CHECKING TO NUMBER OF UNIQUE VALUES**



|  |  |
| --- | --- |
| In [5]: housing.nunique() |  |
| Out[5]: longitude | 844 |
| latitude | 862 |
| housing\_median\_age | 52 |
| total\_rooms | 5926 |
| total\_bedrooms | 1923 |
| population | 3888 |
| households | 1815 |
| median\_income | 12928 |
| median\_house\_value | 3842 |
| ocean\_proximity | 5 |
| dtype: int64 |  |



|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| In [6]: housing.info() | | | |  |  |  |  |  |  |  |
| <class 'pandas.core.frame.DataFrame'> | | | | | | |  |  |  |  |
| RangeIndex: 20640 entries, 0 to 20639 | | | | | | |  |  |  |  |
| Data | | columns (total 10 columns): | | | | |  |  |  |  |
| # | | Column | |  | Non-Null Count | | Dtype |  |  |  |
| --- | | ------ | |  | -------------- | | ----- |  |  |  |
| 0 | | longitude | |  | 20640 | non-null | float64 |  |  |  |
| 1 | | latitude | |  | 20640 | non-null | float64 |  |  |  |
| 2 | | housing\_median\_age | | | 20640 | non-null | float64 |  |  |  |
| 3 | | total\_rooms | | | 20640 | non-null | float64 |  |  |  |
| 4 | | total\_bedrooms | | | 20433 | non-null | float64 |  |  |  |
| 5 | | population | |  | 20640 | non-null | float64 |  |  |  |
| 6 | | households | |  | 20640 | non-null | float64 |  |  |  |
| 7 | | median\_income | | | 20640 | non-null | float64 |  |  |  |
| 8 | | median\_house\_value | | | 20640 | non-null | float64 |  |  |  |
| 9 | | ocean\_proximity | | | 20640 | non-null | object |  |  |  |
| dtypes: float64(9), object(1) | | | | | |  |  |  |  |  |
| memory usage: 1.6+ MB | | | | |  |  |  |  |  |  |
| In [7]: housing.isnull() | | | | |  |  |  |  |  |  |
| Out[7]: | |  | **longitude** | **latitude housing\_median\_age** | | | **total\_rooms total\_bedrooms population house** | | |  |
|  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  | **0** | False | False |  | False | False | False | False |  |
|  |  | **1** | False | False |  | False | False | False | False |  |
|  |  | **2** | False | False |  | False | False | False | False |  |
|  |  | **3** | False | False |  | False | False | False | False |  |
|  |  | **4** | False | False |  | False | False | False | False |  |
| **...** | | | ... | ... |  | ... | ... | ... | ... |  |
| **20635** | | | False | False |  | False | False | False | False |  |
| **20636** | | | False | False |  | False | False | False | False |  |
| **20637** | | | False | False |  | False | False | False | False |  |
| **20638** | | | False | False |  | False | False | False | False |  |
| **20639** | | | False | False |  | False | False | False | False |  |

20640 rows × 10 columns



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | |  |  |
| In [8]: housing.isnull | |  |  |  |  |  |
| Out[8]: <bound method DataFrame.isnull of | | | | longitude | latitude | housing\_median\_ |
| age | total\_rooms | total\_bedrooms |  | \ |  |  |
| 0 | -122.23 | 37.88 |  | 41.0 | 880.0 | 129.0 |
| 1 | -122.22 | 37.86 |  | 21.0 | 7099.0 | 1106.0 |
| 2 | -122.24 | 37.85 |  | 52.0 | 1467.0 | 190.0 |
| 3 | -122.25 | 37.85 |  | 52.0 | 1274.0 | 235.0 |
| 4 | -122.25 | 37.85 |  | 52.0 | 1627.0 | 280.0 |
| ... | ... | ... |  | ... | ... | ... |
| 20635 | -121.09 | 39.48 |  | 25.0 | 1665.0 | 374.0 |
| 20636 | -121.21 | 39.49 |  | 18.0 | 697.0 | 150.0 |
| 20637 | -121.22 | 39.43 |  | 17.0 | 2254.0 | 485.0 |
| 20638 | -121.32 | 39.43 |  | 18.0 | 1860.0 | 409.0 |
| 20639 | -121.24 | 39.37 |  | 16.0 | 2785.0 | 616.0 |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | population households median\_income median\_house\_value \ | | | |
| 0 | 322.0 | 126.0 | 8.3252 | 452600.0 |
| 1 | 2401.0 | 1138.0 | 8.3014 | 358500.0 |
| 2 | 496.0 | 177.0 | 7.2574 | 352100.0 |
| 3 | 558.0 | 219.0 | 5.6431 | 341300.0 |
| 4 | 565.0 | 259.0 | 3.8462 | 342200.0 |
| ... | ... | ... | ... | ... |
| 20635 | 845.0 | 330.0 | 1.5603 | 78100.0 |
| 20636 | 356.0 | 114.0 | 2.5568 | 77100.0 |
| 20637 | 1007.0 | 433.0 | 1.7000 | 92300.0 |
| 20638 | 741.0 | 349.0 | 1.8672 | 84700.0 |
| 20639 | 1387.0 | 530.0 | 2.3886 | 89400.0 |
|  | ocean\_proximity |  |  |  |
| 0 | NEAR BAY |  |  |  |
| 1 | NEAR BAY |  |  |  |
| 2 | NEAR BAY |  |  |  |
| 3 | NEAR BAY |  |  |  |
| 4 | NEAR BAY |  |  |  |
| ... | ... |  |  |  |
| 20635 | INLAND |  |  |  |
| 20636 | INLAND |  |  |  |
| 20637 | INLAND |  |  |  |
| 20638 | INLAND |  |  |  |
| 20639 | INLAND |  |  |  |

[20640 rows x 10 columns]>



|  |  |
| --- | --- |
|  |  |
| In [9]: housing.isnull().sum() | |
| Out[9]: longitude | 0 |
| latitude | 0 |
| housing\_median\_age | 0 |
| total\_rooms | 0 |
| total\_bedrooms | 207 |
| population | 0 |
| households | 0 |
| median\_income | 0 |
| median\_house\_value | 0 |
| ocean\_proximity | 0 |
| dtype: int64 |  |




In [10]: housing.columns

Out[10]: Index(['longitude', 'latitude', 'housing\_median\_age', 'total\_rooms', 'total\_bedrooms', 'population', 'households', 'median\_income', 'median\_house\_value', 'ocean\_proximity'],



|  |  |  |  |
| --- | --- | --- | --- |
|  | dtype='object') | | |
| In [11]: | housing["ocean\_proximity"].value\_counts() | | |
| Out[11]: <1H OCEAN | | | 9136 |
|  | INLAND |  | 6551 |
|  | NEAR OCEAN | | 2658 |
|  | NEAR BAY |  | 2290 |
|  | ISLAND |  | 5 |
|  | Name: ocean\_proximity, dtype: int64 | | |
| In [12]: | housing["housing\_median\_age"].value\_counts().head(5) | | |
| Out[12]: | 52.0 | 1273 |  |
|  | 36.0 | 862 |  |
|  | 35.0 | 824 |  |
|  | 16.0 | 771 |  |
|  | 17.0 | 698 |  |
|  | Name: housing\_median\_age, dtype: int64 | | |
| In [13]: | housing["housing\_median\_age"].value\_counts().head(5) | | |
| Out[13]: | 52.0 | 1273 |  |
|  | 36.0 | 862 |  |
|  | 35.0 | 824 |  |
|  | 16.0 | 771 |  |
|  | 17.0 | 698 |  |

Name: housing\_median\_age, dtype: int64



In [14]: housing.shape

Out[14]: (20640, 10)



In [15]: housing.size

Out[15]: 206400



In [16]: housing.describe()

Out[16]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **longitude** | **latitude** | **housing\_median\_age** | **total\_rooms** | | **total\_bedrooms** | **popul** |
|  |  | |  |  |  |  | |  |  |
|  | **count** | | 20640.000000 | 20640.000000 | 20640.000000 | 20640.000000 | | 20433.000000 | 20640.00 |
|  | **mean** | | -119.569704 | 35.631861 | 28.639486 | 2635.763081 | | 537.870553 | 1425.47 |
|  |  | **std** | 2.003532 | 2.135952 | 12.585558 | 2181.615252 | | 421.385070 | 1132.46 |
|  |  | **min** | -124.350000 | 32.540000 | 1.000000 | 2.000000 | | 1.000000 | 3.00 |
| **25%** | | | -121.800000 | 33.930000 | 18.000000 | 1447.750000 | | 296.000000 | 787.00 |
| **50%** | | | -118.490000 | 34.260000 | 29.000000 | 2127.000000 | | 435.000000 | 1166.00 |
| **75%** | | | -118.010000 | 37.710000 | 37.000000 | 3148.000000 | | 647.000000 | 1725.00 |
|  |  | **max** | -114.310000 | 41.950000 | 52.000000 | 39320.000000 | | 6445.000000 | 35682.00 |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

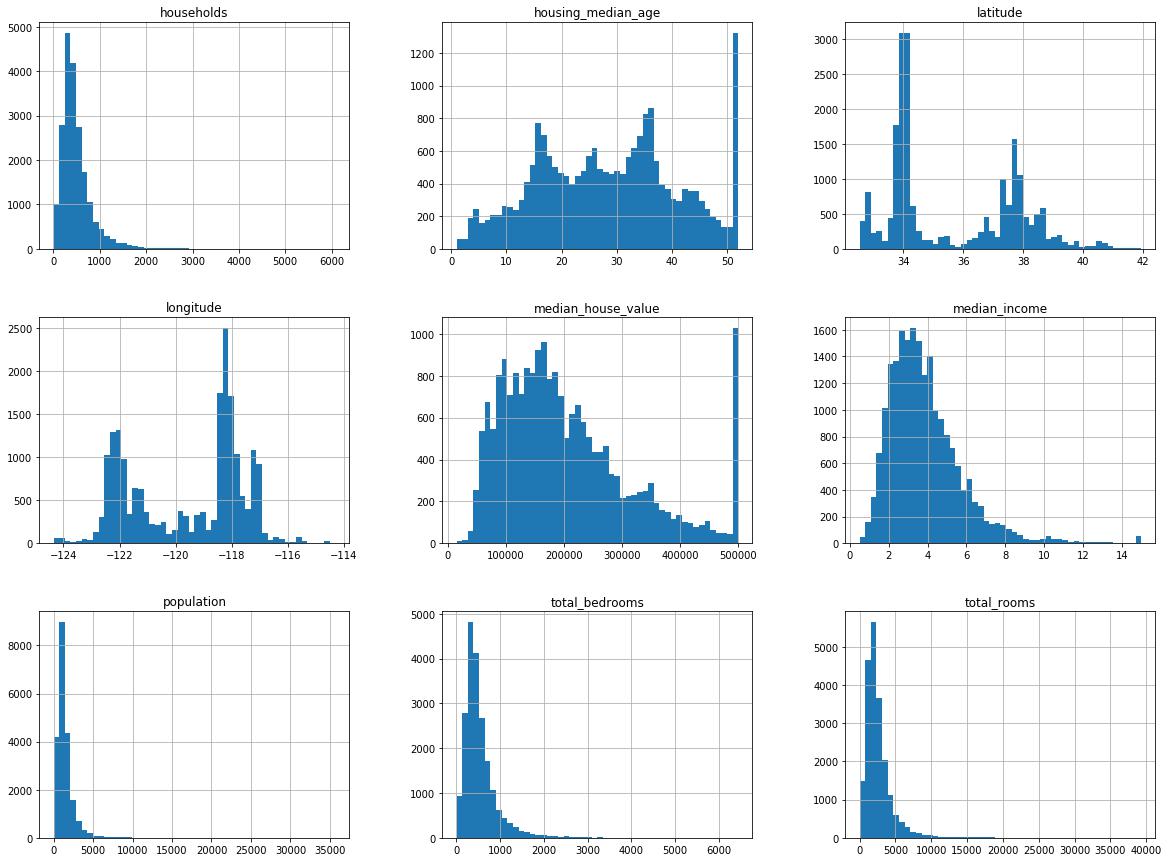




In [17]: **import matplotlib.pyplot as plt**

housing.hist(bins=50, figsize=(20,15))

plt.show()



In [18]: housing.head(5)

Out[18]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **longitude** | **latitude** | **housing\_median\_age total\_rooms** | | **total\_bedrooms population households** | | | |
|  |  | |  |  |  |  |  | |  |  |
| **0** | | | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | | 322.0 | 126.0 |
| **1** | | | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | | 2401.0 | 1138.0 |
| **2** | | | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | | 496.0 | 177.0 |
| **3** | | | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | | 558.0 | 219.0 |
| **4** | | | -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | | 565.0 | 259.0 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |





In [19]: housing["households"].value\_counts()

|  |  |
| --- | --- |
| Out[19]: 306.0 | 57 |
| 386.0 | 56 |
| 335.0 | 56 |
| 282.0 | 55 |
| 429.0 | 54 |
|  | .. |
| 1506.0 | 1 |
| 1765.0 | 1 |
| 1338.0 | 1 |
| 2333.0 | 1 |
| 1455.0 | 1 |

Name: households, Length: 1815, dtype: int64



In [20]: housing.size

Out[20]: 206400



In [21]: data = housing.dropna()



In [22]: data.size

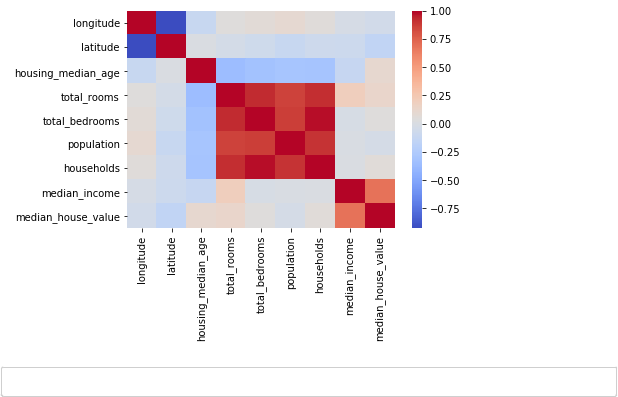
Out[22]: 204330

**Checking Correlation**



In [23]: sns.heatmap(data.corr(),cmap='coolwarm')

Out[23]: <matplotlib.axes.\_subplots.AxesSubplot at 0x1d337f43808>



In [24]: data.columns

Out[24]: Index(['longitude', 'latitude', 'housing\_median\_age', 'total\_rooms', 'total\_bedrooms', 'population', 'households', 'median\_income', 'median\_house\_value', 'ocean\_proximity'],

dtype='object')



In [25]: data.head(4)

Out[25]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **longitude** | **latitude** | **housing\_median\_age total\_rooms** | | **total\_bedrooms population households** | | | |
|  |  | |  |  |  |  |  | |  |  |
| **0** | | | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | | 322.0 | 126.0 |
| **1** | | | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | | 2401.0 | 1138.0 |
| **2** | | | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | | 496.0 | 177.0 |
| **3** | | | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | | 558.0 | 219.0 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |





In [26]: **del** housing['total\_bedrooms']



In [27]: housing.head(4)

Out[27]:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **longitude** | **latitude** | **housing\_median\_age total\_rooms population** | | | | **households** | **median\_income** |
|  |  | |  |  |  |  |  | |  |  |
| **0** | | | -122.23 | 37.88 | 41.0 | 880.0 | 322.0 | | 126.0 | 8.3252 |
| **1** | | | -122.22 | 37.86 | 21.0 | 7099.0 | 2401.0 | | 1138.0 | 8.3014 |
| **2** | | | -122.24 | 37.85 | 52.0 | 1467.0 | 496.0 | | 177.0 | 7.2574 |
| **3** | | | -122.25 | 37.85 | 52.0 | 1274.0 | 558.0 | | 219.0 | 5.6431 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |





In [28]: **del** housing['population']



In [29]: housing.head(4)

Out[29]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **longitude** | **latitude** | **housing\_median\_age total\_rooms households** | | | **median\_income median\_h** | |
|  |  | |  |  |  |  |  |  | |
| **0** | | | -122.23 | 37.88 | 41.0 | 880.0 | 126.0 | 8.3252 | |
| **1** | | | -122.22 | 37.86 | 21.0 | 7099.0 | 1138.0 | 8.3014 | |
| **2** | | | -122.24 | 37.85 | 52.0 | 1467.0 | 177.0 | 7.2574 | |
| **3** | | | -122.25 | 37.85 | 52.0 | 1274.0 | 219.0 | 5.6431 | |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |





In [30]: **del** housing['households']



|  |  |
| --- | --- |
| In [31]: housing1.nunique() |  |
| Out[31]: longitude | 844 |
| latitude | 862 |
| housing\_median\_age | 52 |
| total\_rooms | 5926 |
| total\_bedrooms | 1923 |
| population | 3888 |
| households | 1815 |
| median\_income | 12928 |
| median\_house\_value | 3842 |
| ocean\_proximity | 5 |
| dtype: int64 |  |



In [32]: housing.head()

Out[32]:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **longitude** | **latitude** | **housing\_median\_age total\_rooms median\_income** | | | **median\_house\_value o** | |
|  |  | |  |  |  |  |  |  | |
| **0** | | | -122.23 | 37.88 | 41.0 | 880.0 | 8.3252 | 452600.0 | |
| **1** | | | -122.22 | 37.86 | 21.0 | 7099.0 | 8.3014 | 358500.0 | |
| **2** | | | -122.24 | 37.85 | 52.0 | 1467.0 | 7.2574 | 352100.0 | |
| **3** | | | -122.25 | 37.85 | 52.0 | 1274.0 | 5.6431 | 341300.0 | |
| **4** | | | -122.25 | 37.85 | 52.0 | 1627.0 | 3.8462 | 342200.0 | |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |





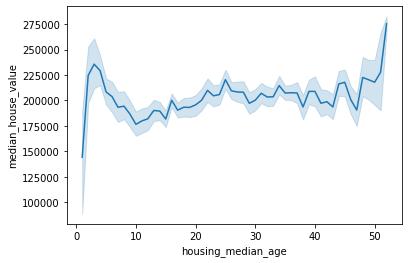
In [33]: **del** housing["ocean\_proximity"]

**Visualization**



In [34]: sns.lineplot(housing["housing\_median\_age"],housing["median\_house\_value"])

plt.show()

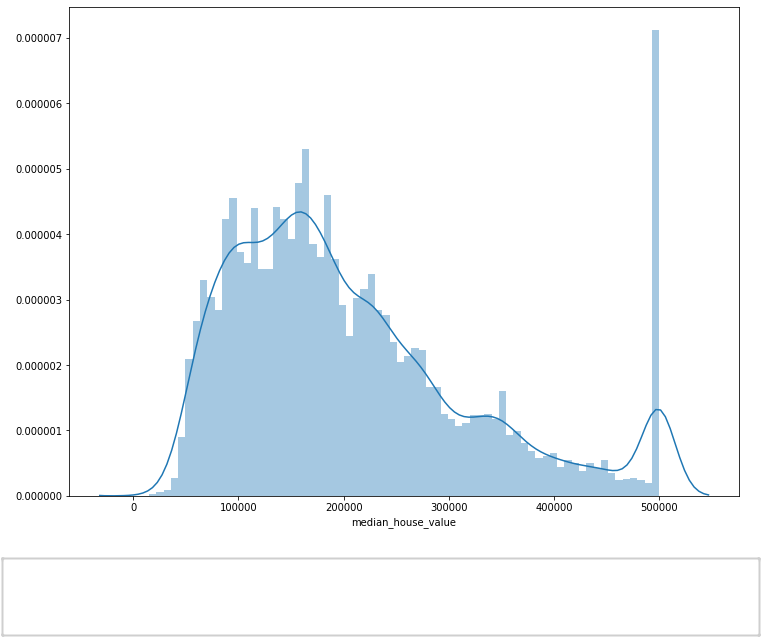




In [35]: plt.subplots(figsize=(12,9))

sns.distplot(housing['median\_house\_value'],bins = 70)

Out[35]: <matplotlib.axes.\_subplots.AxesSubplot at 0x1d3399c5a48>



In [36]: corr = housing.corr()

plt.subplots(figsize=(20,9))

sns.heatmap(corr, annot=**True**)

Out[36]: <matplotlib.axes.\_subplots.AxesSubplot at 0x1d338ceb6c8>





In [37]: y = housing["median\_house\_value"]



In [38]: **del** housing["median\_house\_value"]



In [39]: X = housing.values



In [40]: y = y.values



In [41]: **from sklearn.model\_selection import** train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, rando m\_state=7)



In [42]: **from sklearn import** linear\_model

model = linear\_model.LinearRegression()



In [43]: model.fit(X\_train, y\_train)

Out[43]: LinearRegression(copy\_X=True, fit\_intercept=True, n\_jobs=None, normalize=Fals e)



In [44]: y\_pred = model.predict(X\_test)

In [45]: print("Predict value " + str(model.predict([X\_test[600]])))

print("Real value " + str(y\_test[600]))

Predict value [312985.97547473]

Real value 410500.0



In [46]: y\_pred

Out[46]: array([153526.02420419, 244917.16207568, 249089.2341909 , ..., 202145.21045608, 237283.07582104, 397578.96788946])



In [47]: y\_test

Out[47]: array([360000., 336000., 269900., ..., 200000., 406500., 413700.])



In [48]: print("Accuracy --> ", model.score(X\_test, y\_test)\*100)

Accuracy --> 59.415290009731415



In [49]: df1 =pd.DataFrame({"y\_pred": y\_pred})



In [50]: df2 = pd.DataFrame({"y\_test": y\_test})



In [51]: df1 = df1.loc[0:500].astype(float)



In [52]: df2 = df2.loc[0:500].astype(float)

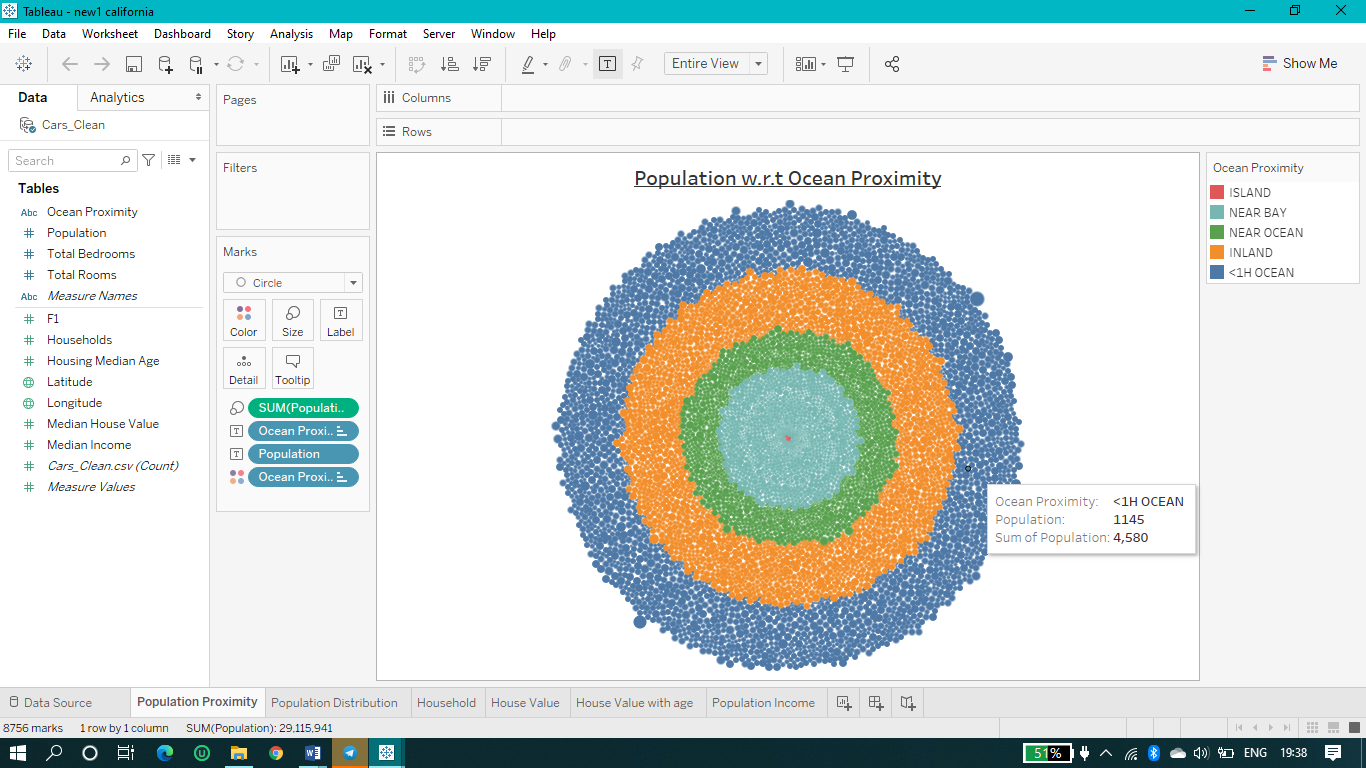


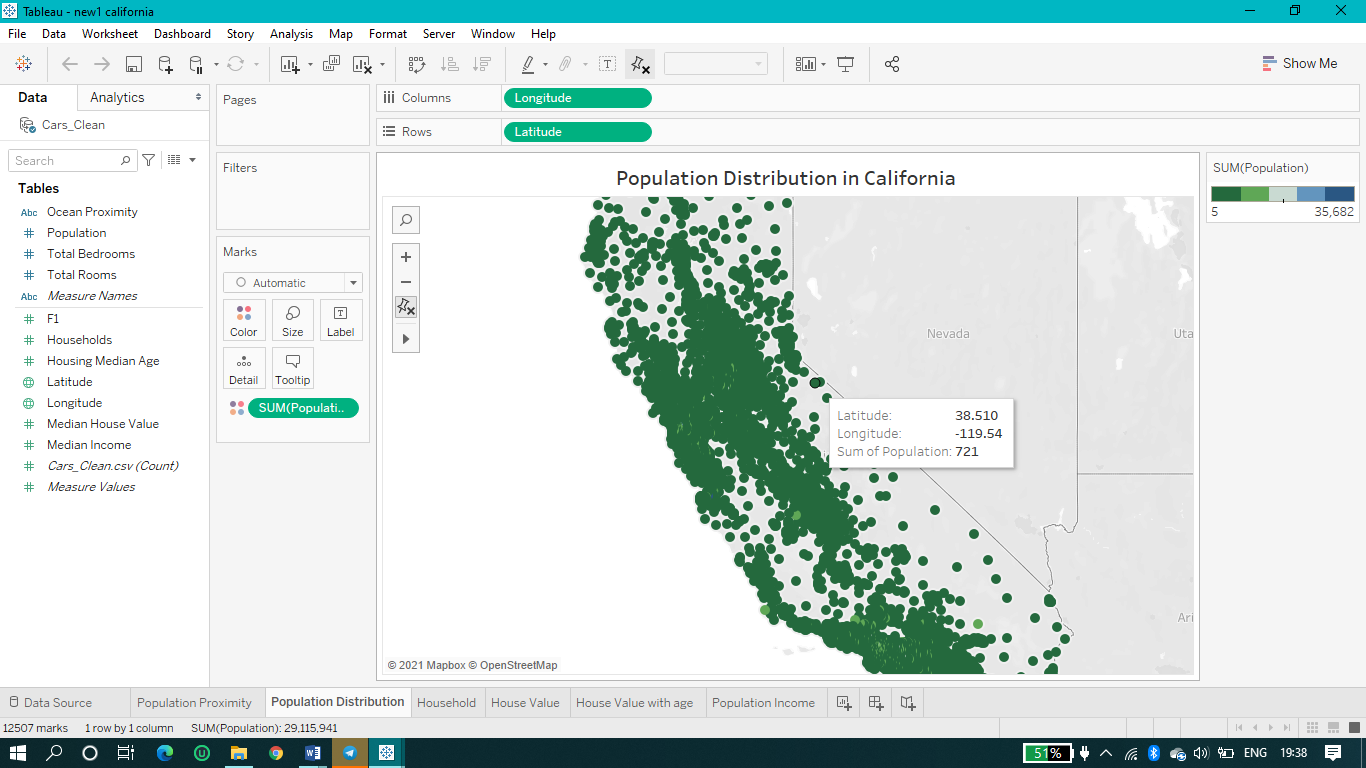
In [53]: df1.to\_csv("D:\y\_pred.csv")

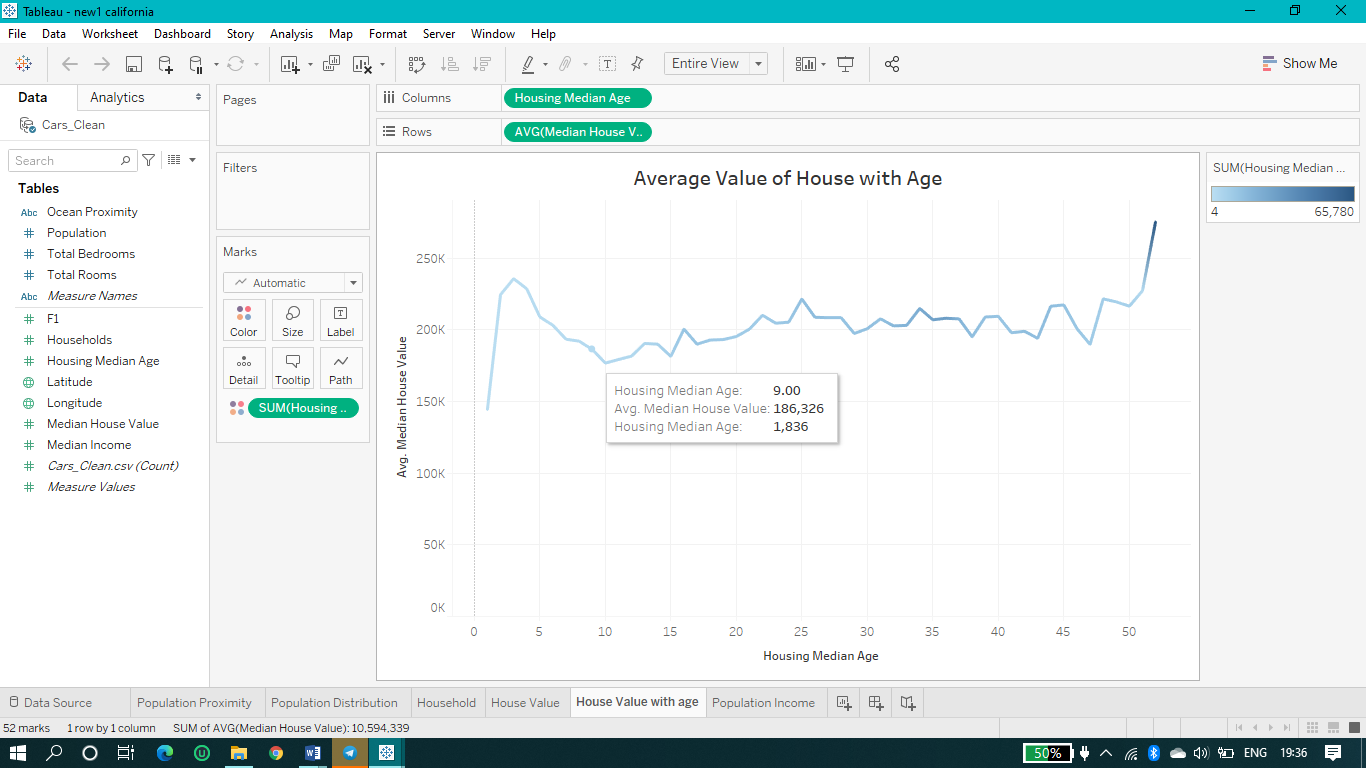


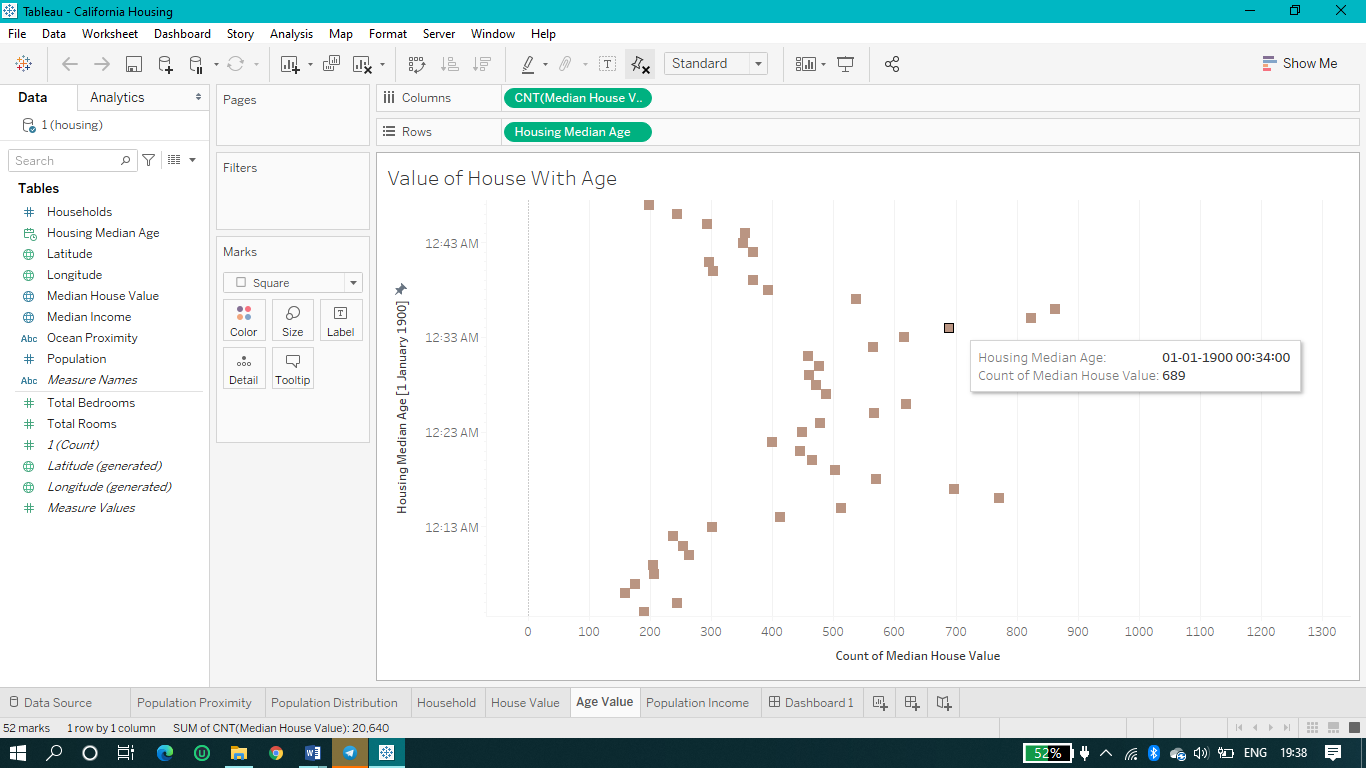
In [54]: df2.to\_csv("D:\y\_test.csv"

**Visualization on Tableau**









**Advantages and disadvantages**

## 

## 

## 

## Advantages

* Smarter detection
* Prioritize workloads
* Monitor progress
* Detect patterns to initiate action
* Aggregate and correlate information
* Optimize processes and performance
* Identity insights and relationships insights
* Catch suspicious trends before loss occurs
* Achieve improved collaboration and control
* Embed logic into case management systems

## 

## 

## Disadvantages

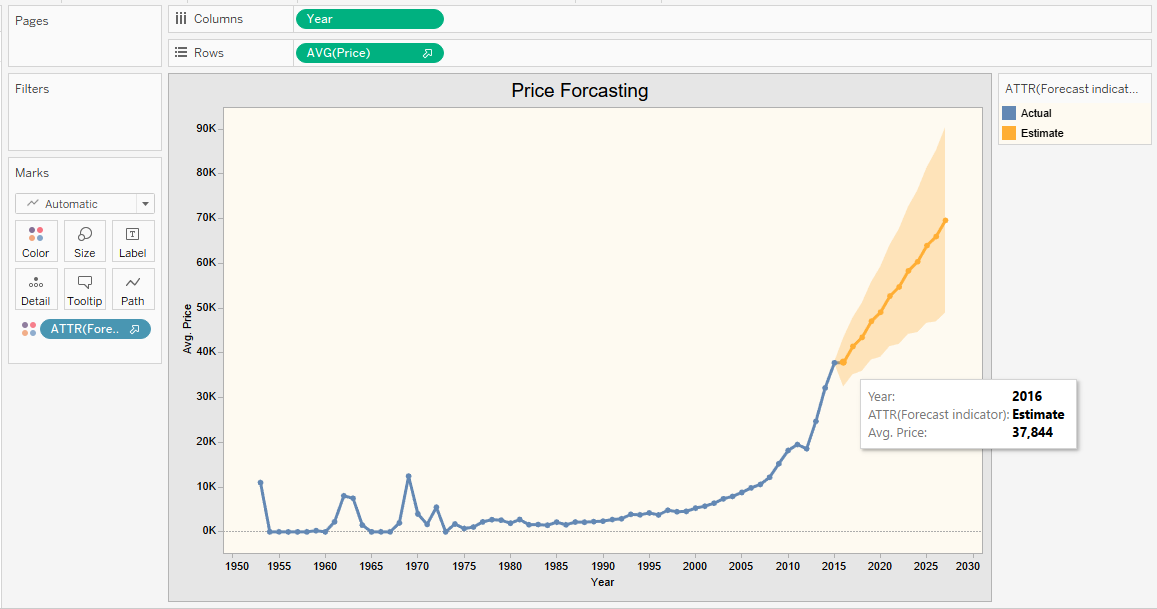
* The data could be incomplete. Missing values, even the lack of a section or a substantial part of the data, could limit its usability.
* If you’re using data from surveys, keep in mind that people don’t always provide accurate information.
* Data collected from different sources can vary in quality and format. Data collected from such diverse sources as surveys, e-mails, data-entry forms, and the company website will have different attributes and structures.

**APPLICATIONS OF DATA PREDICTION MODEL:**

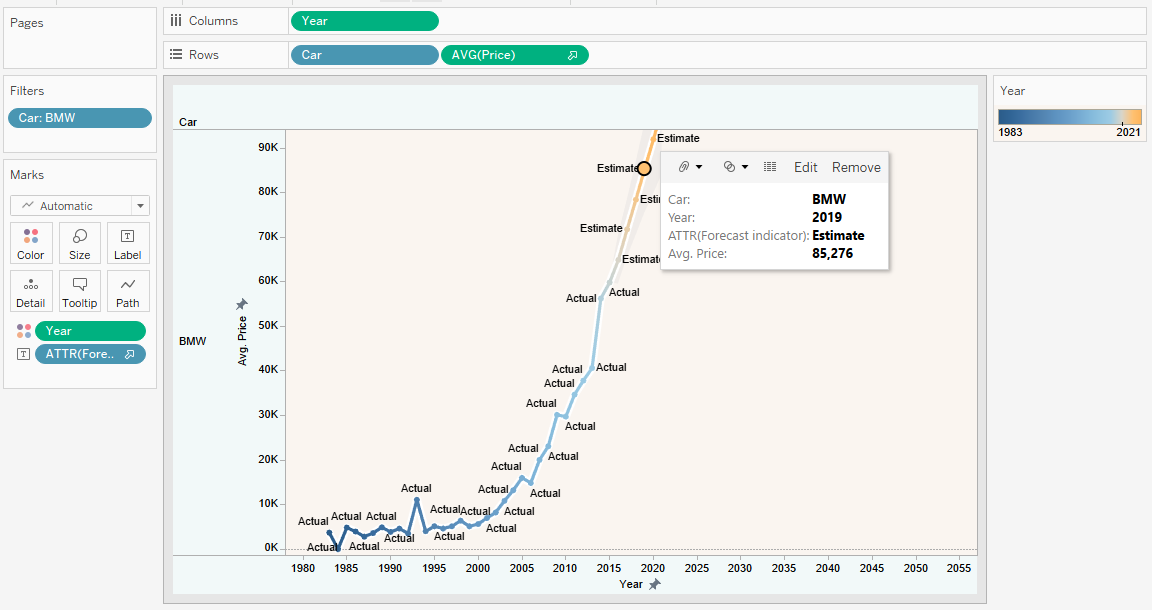
* Actuarial science
* Marketing
* Financial services
* Insurance
* Telecommunications
* Travel
* Healthcare
* Mobility

**Result**

***Forecasting***



(Year vs Average Price )



[Year vs Average Price (BMW)]

Forecasting the estimated price of the cars using our data prediction model based on Python and Machine Learning using raw data.

# CONCLUSION:

This paper has presented the Report of our Minor Project, i.e.. DATA PREDICTION MODEL. It has shown all the required information about the introduction, objective, plan, advantages and application of the project. It can make a positive contribution to society. Data science can give you some pretty super superpowers. One of them is reshaping industries like healthcare, business. The amount of data produced about patients and illnesses rises by the second, opening new opportunities for better structured and more informed healthcare. The challenge is to carefully analyze the data in order to be able to recognize problems quickly and accurately – like deepsense.ai did in diagnosing diabetic retinopathy with deep learning.

We made a predictive model using Multi-Linear Regression technique for predicting the sales of automobile company and forecast the estimated value of their respective models w.r.t year.

**FUTURE SCOPE**

The market is witnessing an unprecedented shift in business intelligence (BI), largely because of technological innovation and increasing business needs. The latest shift in the BI market is the move from traditional analytics to predictive analytics. Although predictive analytics belongs to the BI family, it is emerging as a new distinct software sector.  
  
Analytical tools enable greater transparency, and can find and analyze past and present trends, as well as the hidden nature of data. However, past and present insight and trend information are not enough to be competitive in business. Business organizations need to know more about the future, and in particular, about future trends, patterns, and customer behavior in order to understand the market better. To meet this demand, many BI vendors developed predictive analytics to forecast future trends in customer behavior, buying patterns, and who is coming into and leaving the market and why.  
  
Traditional analytical tools claim to have a real 360° view of the enterprise or business, but they analyze only historical data—data about what has already happened. Traditional analytics help gain insight for what was right and what went wrong in decision-making. Today's tools merely provide rear view analysis.  
However, one cannot change the past, but one can prepare better for the future and decision makers want to see the predictable future, control it, and take actions today to attain tomorrow's goals.

**REFERENCES**

* Aurélien Géron , “Hands-On Machine Learning with Scikit Learn and TensorFlow”.
* <https://www.python.org/doc/>
* <https://www.tableau.com/>
* <https://www.anaconda.com/>
* <https://seaborn.pydata.org/>
* <https://numpy.org/>
* <https://matplotlib.org/>
* <https://pandas.pydata.org/>
* [https://scikit-learn.org/stable/](https://scikit-learn.org/stable/%20)
* <https://www.kaggle.com>
* <https://www.kaggle.com/mohammadarshoddin/car-sales-prediction>
* Nyce, Charles (2007), [*Predictive Analytics White Paper*](https://www.the-digital-insurer.com/wp-content/uploads/2013/12/78-Predictive-Modeling-White-Paper.pdf) , American Institute for Chartered Property Casualty Underwriters/Insurance Institute of America, p. 1
* *Fletcher, Heather (March 2, 2011),*[*"The 7 Best Uses for Predictive Analytics in Multichannel Marketing"*](http://www.targetmarketingmag.com/article/7-best-uses-predictive-analytics-modeling-multichannel-marketing/1)*, Target Marketing*
* Gaeth, Andrae. ["Evaluating Predictive Analytics for Capacity Planning"](https://www.hisa.org.au/slides/tue/AndraeGaeth.pdf) (PDF). *www.hisa.org.au*. Retrieved 22 November 2018.
* Finlay, Steven (2014). *Predictive Analytics, Data Mining and Big Data. Myths, Misconceptions and Methods* (1st ed.). Basingstoke: Palgrave Macmillan. p. 237. [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)) [978-1137379276](https://en.wikipedia.org/wiki/Special:BookSources/978-1137379276).
* Siegel, Eric (2013). *Predictive Analytics: The Power to Predict Who Will Click, Buy, Lie, or Die* (1st ed.). Wiley. [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)) [978-1-1183-5685-2](https://en.wikipedia.org/wiki/Special:BookSources/978-1-1183-5685-2).
* Temple-Raston, Dina (Oct 8, 2012), [*Predicting The Future: Fantasy Or A Good Algorithm?*](https://www.npr.org/2012/10/08/162397787/predicting-the-future-fantasy-or-a-good-algorithm), NPR
* *Halper, Fern (November 1, 2011),*[*"The Top 5 Trends in Predictive Analytics"*](http://www.information-management.com/issues/21_6/the-top-5-trends-in-redictive-an-alytics-10021460-1.html)*, Information Management*