ZEAL EDUCATION SOCIETY's

ZEAL COLLEGE OF ENGINEERING AND RESEARCH, NARHE, PUNE

DEPARTMENT OF COMPUTER ENGINEERING

SEMESTER-II

[A.Y. : 2022 - 2023]

Data Science and Big Data Analytics (310251)
LABORATORY MANUAL

Department Vision and Mission

INSTITUTE VISION	To impart value added technological education through pursuit of academic excellence, research and entrepreneurial attitude.
INSTITUTE MISSION	**M1:** To achieve academic excellence through innovative teaching and learning process. **M2:** To imbibe the research culture for addressing industry and societal needs**.** **M3:** To provide conducive environment for building the entrepreneurial skills. **M4:** To produce competent and socially responsible professionals with core human values.

DEPARTMENT VISION	To emerge as a department of repute in Computer Engineering which produces competent professionals and entrepreneurs to lead technical and betterment of mankind.
DEPARTMENT MISSION	**M1:** To strengthen the theoretical and practical aspects of the learning process by teaching applications and hands on practices using modern tools and FOSS technologies. **M2:** To endavor innovative interdisciplinary research and entrepreneurship skills to serve the needs of Industry and Society. **M3:** To enhance industry academia dialog enabling students to inculcate professional skills. **M4:** To incorporate social and ethical awareness among the students to make them conscientious professionals.

Department
Program Educational Objectives (PEOs)

PEO1: To Impart fundamentals in science, mathematics and engineering to cater the needs of society and Industries.
PEO2: Encourage graduates to involve in research, higher studies, and/or to become entrepreneurs.
PEO3: To Work effectively as individuals and as team members in a multidisciplinary environment with high ethical values for the benefit of society.

Savitribai Phule Pune University Third Year of Computer Engineering (2019 Course) 310251: Data Science And Big Data Analytics		
Teaching Scheme: PR: 04 Hours/Week	**Credit** 02	**Examination Scheme:** TW: 25 Marks PR: 50 Marks

Course Objectives:

- To understand principles of Data Science for the analysis of real time problems
- To develop in depth understanding and implementation of the key technologies in Data Science and Big Data Analytics
- To analyze and demonstrate knowledge of statistical data analysis techniques for decision-making
- To gain practical, hands-on experience with statistics programming languages and Big Data tool

Course Outcomes:

On completion of the course, student will be able to-

CO1: Apply principles of Data Science for the analysis of real time problems

CO2: Implement data representation using statistical methods

CO3: Implement and evaluate data analytics algorithms

CO4: Perform text preprocessing

CO5: Implement data visualization techniques

CO6: Use cutting edge tools and technologies to analyze Big Data

List of Assignments

Sr. No.	Title
	Group A
01	**Data Wrangling, I** Perform the following operations using Python on any open source dataset (e.g., data.csv) 1. Import all the required Python Libraries. 2. Locate an open source data from the web (e.g., https://www.kaggle.com). Provide a clear description of the data and its source (i.e., URL of the web site). 3. Load the Dataset into pandas dataframe. 4. Data Preprocessing: check for missing values in the data using pandas isnull(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame. 5. Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions. 6. Turn categorical variables into quantitative variables in Python. In addition to the codes and outputs, explain every operation that you do in the above steps and explain everything that you do to import/read/scrape the data set.
02	**Data Wrangling II** Create an "Academic performance" dataset of students and perform the following operations using Python. 1. Scan all variables for missing values and inconsistencies. If there are missing values and/or inconsistencies, use any of the suitable techniques to deal with them. 2. Scan all numeric variables for outliers. If there are outliers, use any of the suitable techniques to deal with them. 3. Apply data transformations on at least one of the variables. The purpose of this transformation should be one of the following reasons: to change the scale for better understanding of the variable, to convert a non-linear relation into a linear one, or to decrease the skewness and convert the distribution into a normal distribution.
03	**Descriptive Statistics** - Measures of Central Tendency and variability Perform the following operations on any open source dataset (e.g., data.csv) 1. Provide summary statistics (mean, median, minimum, maximum, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorical) variable. For example, if your categorical variable is age groups and quantitative variable is income, then provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable. 2. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of 'Iris-setosa', 'Iris-versicolor' and 'Iris-versicolor' of iris.csv dataset. Provide the codes with outputs and explain everything that you do in this step.
04	**Data Analytics I** Create a Linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (https://www.kaggle.com/c/boston-housing). The Boston Housing dataset contains information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables in this dataset. The objective is to predict the value of prices of the house using the given features
05	**Data Analytics II** 1. Implement logistic regression using Python/R to perform classification on Social_Network_Ads.csv dataset. 2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

06	**Data Analytics III** 1. Implement Simple Naïve Bayes classification algorithm using Python/R on iris.csv dataset. 2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.
07	**Text Analytics** 1. Extract Sample document and apply following document preprocessing methods: Tokenization, POS Tagging, stop words removal, Stemming and Lemmatization. 2. Create representation of document by calculating Term Frequency and Inverse Document Frequency.
08	**Data Visualization I** 1. Use the inbuilt dataset 'titanic'. The dataset contains 891 rows and contains information about the passengers who boarded the unfortunate Titanic ship. Use the Seaborn library to see if we can find any patterns in the data. 2. Write a code to check how the price of the ticket (column name: 'fare') for each passenger is distributed by plotting a histogram.
09	**Data Visualization II** 1. Use the inbuilt dataset 'titanic' as used in the above problem. Plot a box plot for distribution of age with respect to each gender along with the information about whether they survived or not. (Column names : 'sex' and 'age') 2. Write observations on the inference from the above statistics
10	**Data Visualization III** Download the Iris flower dataset or any other dataset into a DataFrame. (e.g., https://archive.ics.uci.edu/ml/datasets/Iris). Scan the dataset and give the inference as: 1. List down the features and their types (e.g., numeric, nominal) available in the dataset. 2. Create a histogram for each feature in the dataset to illustrate the feature distributions. 3. Create a boxplot for each feature in the dataset. 4. Compare distributions and identify outliers.
Group B- Big Data Analytics – JAVA/SCALA (Any three)	
1	Write a code in JAVA for a simple WordCount application that counts the number of occurrences of each word in a given input set using the Hadoop MapReduce framework on local-standalone set-up.
2	Design a distributed application using MapReduce which processes a log file of a system.
3	Locate dataset (e.g., sample_weather.txt) for working on weather data which reads the text input files and finds average for temperature, dew point and wind speed.
4	Write a simple program in SCALA using Apache Spark framework

Mini-Projects/ Case Study	
1	Write a case study on Global Innovation Network and Analysis (GINA). Components of analytic plan are 1. Discovery business problem framed, 2. Data, 3. Model planning analytic technique and 4. Results and Key findings.
2	Use the following dataset and classify tweets into positive and negative tweets. https://www.kaggle.com/ruchi798/data-science-tweets
3	3. Develop a movie recommendation model using the scikit-learn library in python. Refer dataset https://github.com/rashida048/Some-NLP-Projects/blob/master/movie_dataset.csv
4	Use the following covid_vaccine_statewise.csv dataset and perform following analytics on the given dataset https://www.kaggle.com/sudalairajkumar/covid19-in-india?select=covid_vaccine_statewise.csv a.

	Describe the dataset b. Number of persons state wise vaccinated for first dose in India c. Number of persons state wise vaccinated for second dose in India d. Number of Males vaccinated d. Number of females vaccinated
5	Write a case study to process data driven for Digital Marketing OR Health care systems with Hadoop Ecosystem components as shown. (Mandatory) ● HDFS: Hadoop Distributed File System ● YARN: Yet Another Resource Negotiator ● MapReduce: Programming based Data Processing ● Spark: In-Memory data processing ● PIG, HIVE: Query based processing of data services ● HBase: NoSQL Database (Provides real-time reads and writes) ● Mahout, Spark MLLib: (Provides analytical tools) Machine Learning algorithm libraries ● Solar, Lucene: Searching and Indexing

Group A

Assignment No 1

Problem Statement:

Perform the following operations using Python on any open source dataset (e.g., data.csv) Import all the required Python Libraries.

1. Locate open source data from the web (e.g.https://www.kaggle.com).
2. Provide a clear description of the data and its source (i.e., URL of the website).
3. Load the Dataset into the pandas data frame.
4. Data Preprocessing: check for missing values in the data using pandas insult(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame.
5. Data Formatting and Data Normalization: Summarize the types of variables by checking the data types (i.e., character, numeric, integer, factor, and logical) of the variables in the data set. If variables are not in the correct data type, apply proper type conversions.
6. Turn categorical variables into quantitative variables in Python.

Objective: Able to perform the data wrangling operation using Python on any open source datasets

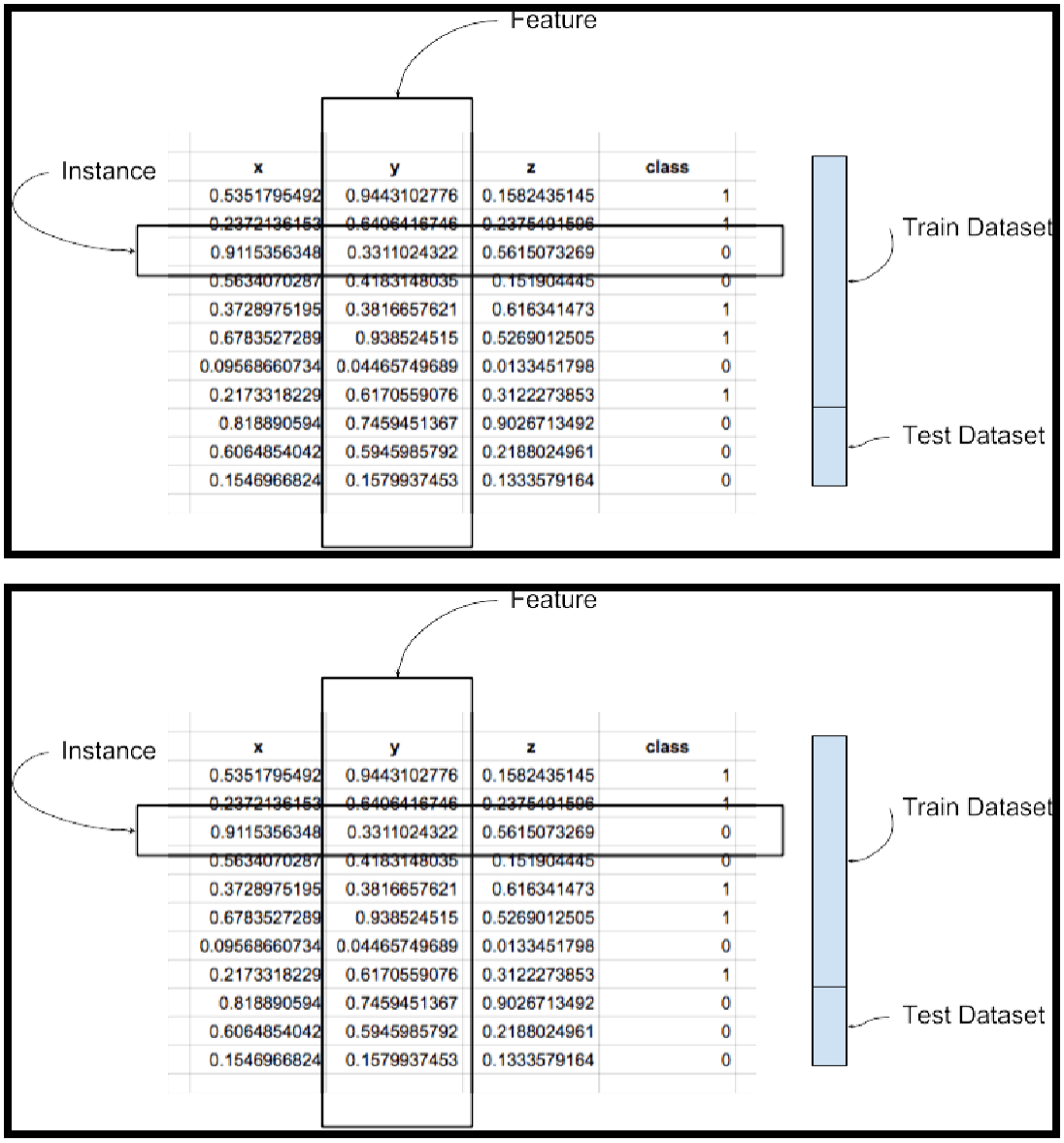
Outcome: To implement the basic concepts of preprocessing, perform normalization and turn categorical variables into quantitative variables in Python

CO Relevance: CO1

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

Introduction to Dataset A dataset is a collection of records, similar to a relational database table. Records are similar to table rows, but the columns can contain not only strings or numbers, but also nested data structures such as lists, maps, and other records.



Feature
Instance
x
y
z
class
0.5351795492 0.9443102776 0.1582435145 1
0.2372136153 0.6406416746 0.2375491596 1
0.9115356348 0.3311024322 0.5615073269 0
0.5634070287 0.4183148035 0.151904445 0
0.3728975195 0.3816657621 0.616341473 1
0.6783527289 0.938524515 0.5269012505 1
0.09568660734 0.04465749689 0.0133451798 0
0.2173318229 0.6170559076 0.3122273853 1
0.818890594 0.7459451367 0.9026713492 0
0.6064854042 0.5945985792 0.2188024961 0
0.1546966824 0.1579937453 0.1333579164 0
Train Dataset
Test Dataset

Instance: A single row of data is called an instance. It is an observation from the domain.

Feature: A single column of data is called a feature. It is a component of an observation and is also called an attribute of a data instance. Some features may be inputs to a model (the predictors) and others may be outputs or the features to be predicted.

Data Type: Features have a data type. They may be real or integer-valued or may have a categorical or

ordinal value. You can have strings, dates, times, and more complex types, but typically they are reduced to real or categorical values when working with traditional machine learning methods.

Datasets: A collection of instances is a dataset and when working with machine learning methods we typically need a few datasets for different purposes.

Training Dataset: A dataset that we feed into our machine learning algorithm to trainour model.

Testing Dataset: A dataset that we use to validate the accuracy of our model but is not used to train the model. It may be called the validation dataset.

Data Represented in a Table:

Data should be arranged in a two-dimensional space made up of rows and columns. This type of data structure makes it easy to understand the data and pinpoint any problems. An example of some raw

```
1., Avatar, 18-12-2009, 7.8
2., Titanic, 18-11-1997,
3., Avengers Infinity War, 27-04-2018, 8.5
```

data stored as a CSV (comma separated values).

The representation of the same data in a table is as follows:

Pandas Data Types

A data type is essentially an internal construct that a programming language uses to understand how to store and manipulate data.A possible confusing point about pandas data types is that there is some overlap between pandas, python and numpy. This table summarizes the key points:

Pandas dtype	Python type	NumPy type	Usage
object	str or mixed	string_, unicode_, mixed types	Text or mixed numeric and non-numeric values
int64	int	int_, int8, int16, int32, int64, uint8, uint16, uint32, uint64	Integer numbers
float64	float	float_, float16, float32, float64	Floating point numbers
bool	bool	bool_	True/False values
datetime64	NA	datetime64[ns]	Date and time values

timedelta[ns]	NA	NA	Differences between two datetimes
category	NA	NA	Finite list of text values

1. Python Libraries for DataScience

i. Pandas

Pandas is an open-source Python package that provides high-performance, easy-to-use data structures and data analysis tools for the labeled data in Python programming language.

Features of Pandas

i).Indexing, manipulating, renaming, sorting, merging dataframe

ii).Update, Add, Delete columns from a dataframe

iii).Impute missing files, handle missing data or NANs

iv).Plot data with histogram or box plot

ii. NumPy

One of the most fundamental packages in Python, NumPy is a general-purpose array-processing package. It provides high-performance multidimensional array objects and tools to work with the arrays. NumPy is an efficient container of generic multi-dimensional data.

Features of Numpy

Basic array operations: add, multiply, slice, flatten, reshape, indexarrays. Advanced array operations: stack arrays, split into sections, broadcast arrays

a).Work with Date Time or Linear Algebra

b).Basic Slicing and Advanced Indexing in NumPy Python

iii. Matplotlib

This is undoubtedly an essential Python library. You can create stories with the data visualized with Matplotlib. Another library from the SciPy Stack, Matplotlib plots 2D figures.

Features of Matplotlib

Histogram, bar plots, scatter plots, area plot to pie plot, Matplotlib can depict a wide range of visualizations. With a bit of effort and tint of visualization capabilities, with Matplotlib, you can create

just any visualizations: Scatter plots, Area plots, Bar charts and Histograms, Pie charts, Stem plots, Contour plots, Quiver plots and Spectrograms. Matplotlib also facilitates labels, grids, legends, and some more formatting entities with Matplotlib.

iv. Seaborn

So when you read the official documentation on Seaborn, it is defined as the data visualization library based on Matplotlib that provides a high-level interface for drawing attractive and informative statistical graphics. Putting it simply, seaborn is an extension of Matplotlib with advanced features.

Features of Seaborn

1. Determine relationships between multiple variables (correlation)
2. Observe categorical variables for aggregate statistics
3. Analyze univariate or bi-variate distributions and compare them between different data subsets
4. Plot linear regression models for dependent variables
5. Provide high-level abstractions, multi-plot grids.
6. Seaborn is a great second-hand for R visualization libraries like corrplot and ggplot.

v. Scikit Learn

S.No	Movie	Release Date	Ratings (IMDb)
1.	Avatar	18-12-2009	7.8
2.	Titanic	18-11-1997	*Na*
3.	Avengers Infinity War	27-04-2018	8.5

Introduced to the world as a Google Summer of Code project, Scikit Learn is a robust machine learning library for Python. It features ML algorithms like SVMs, random forests, k-means clustering, spectral clustering, mean shift, cross-validation and more... Even NumPy, SciPy and related scientific operations are supported by Scikit Learn with Scikit Learn being a part of the SciPy Stack.

Features of Scikit Learn

1. Classification: Spam detection, image recognition
2. Clustering: Drug response, Stock price
3. Regression: Customer segmentation, Grouping experiment outcomes
4. Dimensionality reduction: Visualization, Increasedefficiency
5. Model selection: Improved accuracy via parameter tuning

6. Pre-processing: Preparing input data as a text for processing with machine learning algorithms.

2. Description of Dataset:

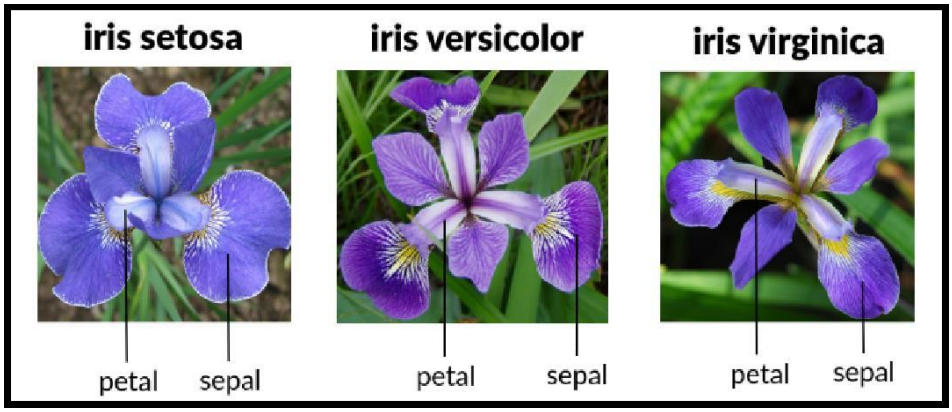
The Iris dataset was used in R.A. Fisher's classic 1936 paper, The Use of MultipleMeasurements in Taxonomic Problems, and can also be found on the UCI Machine Learning Repository.

It includes three iris species with 50 samples each as well as some properties about each flower. One flower species is linearly separable from the other two, but the other two are not linearly separable from each other.

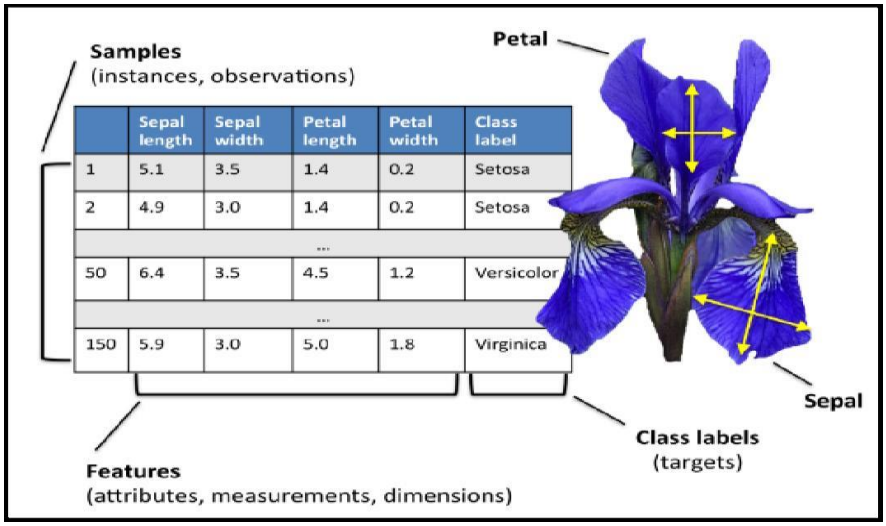
Total Sample- 150

The columns in this dataset are: Id, SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm

3 Different Types of Species each contain 50 Samples



Description of Dataset-



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

3. Panda Dataframe functions for LoadDataset

The columns of the resulting DataFrame have different dtypes. iris.dtypes

1. The dataset is downloaded from UCI repository.

   csv_url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'

2. Now Read CSV File as a Dataframe in Python from from path where you saved the same The Iris data set is stored in .csv format. '.csv' stands for comma separated values. It is easier to load .csv files in Pandas data frame and perform various analytical operations on it.

3. Load Iris.csv into a Pandas data frame —Syntax-iris = pd.read_csv(csv_url, header = None)

4. The csv file at the UCI repository does not contain the variable/column names. They are located in a separate file.

   col_names = ['Sepal_Length','Sepal_Width','Petal_Length','Petal_Width','Species']

5. read in the dataset from the UCI Machine Learning Repository link and specify column names to use iris = pd.read_csv(csv_url, names = col_names)

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

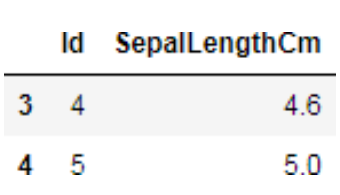
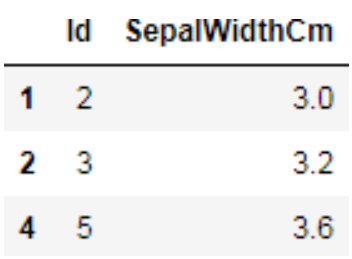
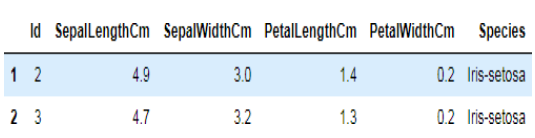
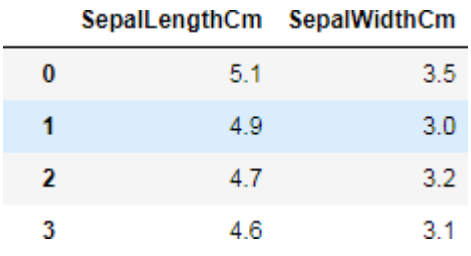
6. Panda Dataframe functions for Data Preprocessing :

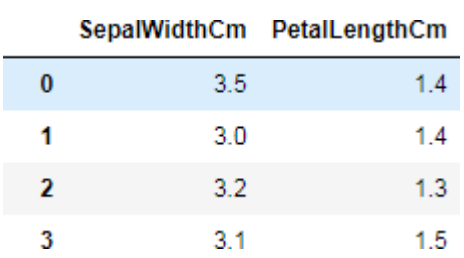
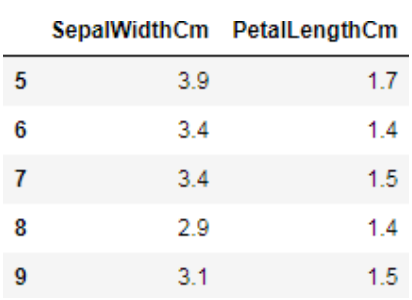
Sr. No	Data Frame Function	Description
1	**dataset.head(n=5)**	**Return the first n rows.**
2	**dataset.tail(n=5)**	**Return the last n rows.**
3	**dataset.index**	The index (row labels) of the Dataset.

4	**dataset.columns**	The column labels of the Dataset.
5	**dataset.shape**	Return a tuple representing the dimensionality of the Dataset.
6	**dataset.dtypes**	Return the dtypes in the Dataset.

7	**dataset.columns.values**	Return the columns values in the Dataset in array format
8	**dataset.describe(include='all')**	Generate descriptive statistics. to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values. Analyzes both numeric and object series, as well as Dataset column sets of mixed data types.
9	**dataset['Column name]**	Read the Data Column wise.
10	**dataset.sort_index(axis= 1, ascending=False)**	Sort object by labels (along an axis).
11	**dataset.sort_values(by="Col u mn name")**	Sort values by column name.
12	**dataset.iloc[5]**	Purely integer-location based indexing for selection by position.
13	**dataset[0:3]**	Selecting via [], which slices the rows.
14	**dataset.loc[:, ["Col_name1", "col_name2"]]**	Selection by label
15	**dataset.iloc[:n, :]**	a subset of the first n rows of the original data
16	**dataset.iloc[:, :n]**	a subset of the first n columns of the original data
17	**dataset.iloc[:m, :n]**	a subset of the first m rows and the first n columns

Few Examples of iLoc to slice data for iris Dataset

Sr. No	Data Frame Function	Description	Output
1	**dataset.iloc[3:5, 0:2]**	Slice the data	
2	**dataset.iloc[[1, 2, 4], [0, 2]]**	By lists of integer position locations, similar to the NumPy/Python style:	
3	**dataset.iloc[1:3, :]**	For slicing rows explicitly:	
4	**dataset.iloc[:, 1:3]**	For slicing Column explicitly:	
4	**dataset.iloc[1, 1]**	For getting a value explicitly:	4.9
5	**dataset['SepalLengthCm'].iloc[5]**	Accessing Column and Rows by position	5.4

6	**cols_2_4=dataset.columns[2:4]** **dataset[cols_2_4]**	Get Column Name then get data from column	
7	**dataset[dataset.columns[2:4]].iloc[5:10]**	in one Expression answer for the above two commands	

4. Checking of Missing Values in Dataset:

- **isnull()** is the function that is used to check missing values or null values in pandas python.
- **isna()** function is also used to get the count of missing values of column and row wise count of missing values
- The dataset considered for explanation is:

	Name	State	Gender	Score
0	George	Arizona	M	63.0
1	Andrea	Georgia	F	48.0
2	micheal	Newyork	M	56.0
3	maggie	Indiana	F	75.0
4	Ravi	Florida	M	NaN
5	Xien	California	M	77.0
6	Jalpa	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN

a. is there any missing values in dataframe as a whole

Function: DataFrame.isnull()

Output:

	Name	State	Gender	Score
0	False	False	False	False
1	False	False	False	False
2	False	False	False	False
3	False	False	False	False
4	False	False	False	True
5	False	False	False	False
6	False	True	True	True
7	True	True	True	True

b. is there any missing values across each column Function: DataFrame.isnull().

Output:

```
Name      True
State     True
Gender    True
Score     True
dtype: bool
```

c. count of missing values across each column using isna() and isnull()

In order to get the count of missing values of the entire dataframe isnull() function is used. sum() which does the column wise sum first and doing another sum() will get the count of missing values of the entire dataframe.

Function: dataframe.isnull().sum()

Output : 8

d. count row wise missing value using isnull() Function: dataframe.isnull().sum(axis = 1)

Output:

```
0    0
1    0
2    0
3    0
4    1
5    0
6    3
7    4
dtype: int64
```

e. count Column wise missing value using isnull() Method1:

Function: dataframe.isnull().sum()

```
Name      1
State     2
Gender    2
Score     3
dtype: int64
```

Output:

Method 2:

Function: dataframe.isna().sum()

```
Name      1
State     2
Gender    2
Score     3
dtype: int64
```

a. **count of missing values of a specific column. Function:**dataframe.col_name.isnull().sum()df1.Gender.isnull().sum()

b. **groupby count of missing values of a column.**

In order to get the count of missing values of the particular column by group in pandas we will be using isnull() and sum() function with apply() and groupby() which performs the group wise count of missing values as shown below.

Function: df1.groupby(['Gender'])['Score'].apply(lambdax: x.isnull().sum())

Panda functions for Data Formatting and Normalization

The Transforming data stage is about converting the data set into a format that can be analyzed or model effectively, and there are several techniques for this process.

5. **Data Formatting:** Ensuring all data formats are correct (e.g. object, text, floating number, integer, etc.) is another part of this initial 'cleaning' process. If you are working with dates in Pandas, they also need to be stored in the exact format to use special date-time functions.

Functions used for data formatting

Sr. No	Data Frame Function	Description	Output

1.	**df.dtypes**	To check the data type	`df.dtypes`
			`sepal length (cm) float64`
			`sepal width (cm) float64`
			`petal length (cm) float64`
			`petal width (cm) float64`
			`dtype: object`

2.	**df['petal length (cm)']= df['petal length (cm)'].astype("int ")**	To change the data type (data type of 'petal length (cm)'changed to int)	`df.dtypes`
			`sepal length (cm) float64`
			`sepal width (cm) float64`
			`petal length (cm) int64`
			`petal width (cm) float64`
			`dtype: object`

5. **Data normalization:** Mapping all the nominal data values onto a uniform scale (e.g.from0to1) is involved in data normalization. Making the ranges consistent across variables helps with statistical analysis and ensures better comparisons later on. It is also known as Min-Max scaling.

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing from sklearn import preprocessing

Step 2: Load the iris dataset in dataframe object df

Step 3: Print iris dataset. **df.head()**

Step 5: Create a minimum and maximum processor object

min_max_scaler = preprocessing.MinMaxScaler()

Step 6: Separate the feature from the class label

x=df.iloc[:,:4]

Step 7: Create an object to transform the data to fit min max processor

x_scaled = min_max_scaler.fit_transform(x)

Step 8: Run the normalizer on the dataframe

df_normalized = pd.DataFrame(x_scaled)

Step 8: View the dataframe

df_normalized

Output: After Step 3:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

	0	1	2	3
0	0.222222	0.625000	0.067797	0.041667
1	0.166667	0.416667	0.067797	0.041667
2	0.111111	0.500000	0.050847	0.041667
3	0.083333	0.458333	0.084746	0.041667
4	0.194444	0.666667	0.067797	0.041667

6. Panda Functions for handling categorical variables

Categorical variables have values that describe a 'quality' or 'characteristic' of a data unit, like 'what type' or 'which category'.

Categorical variables fall into mutually exclusive (in one category or in another) and exhaustive (include all possible options) categories. Therefore, categorical variables are qualitative variables and tend to be represented by a non-numeric value.

Categorical features refer to string type data and can be easily understood by human beings. But in case of a machine, it cannot interpret the categorical data directly. Therefore, the categorical data must be translated into numerical data that can be understood by machine.

There are many ways to convert categorical data into numerical data. Here the three most used methods are discussed.

Label Encoding: Label Encoding refers to **converting the labels into a numeric form** so as to convert

them into the machine-readable form.

Example: Suppose we have a column Height in some dataset. After applying label encoding , the Height column is converted into:

Height
0
1
2

Height
Tall
Medium
Short

where 0 is the label for tall, 1 is the label for medium, and 2 is a label for short height. **Label Encoding on iris dataset:** For iris dataset the target column which is Species. It contains three species Iris-setosa, Iris-versicolor, Iris-virginica.

Sklearn Functions for Label Encoding:

- **preprocessing.LabelEncoder :** It Encode labels with value between 0 and n_classes-1.
- **fit_transform(y):**

  Parameters: yarray-like of shape (n_samples,**) Target values.**

Returns: array-like of shape (n_samples,)

Encoded labels.

This transformer should be used to encode target values, and not the input.

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing from sklearn import preprocessing

Step 2: Load the iris dataset in dataframe object df

Step 3: Observe the unique values for the Species column. df['Species'].unique()

output: array(['Iris-setosa','Iris-versicolor', 'Iris-virginica'], dtype=object)

Step 4: define label_encoder object knows how to understand word labels.label_encoder = preprocessing.LabelEncoder()

Step 5: Encode labels in column 'species'. df['Species']= label_encoder.fit_transform(df['Species'])

Step 6: Observe the unique values for the Species column.

df['Species'].unique()

Output: array([0, 1, 2], dtype=int64)

Use LabelEncoder when there are only two possible values of a categorical feature. For example, features having value such as yes or no. Or, maybe, gender features when there are only two possible values including male or female.

Limitation: Label encoding converts the data in machine-readable form, but it assigns a **unique number(starting from 0) to each class of data**. This may lead to the generation of **priority issues in the data sets**. A label with a high value may be considered to have high priority than a label having a lower value.

One-Hot Encoding:

In one-hot encoding, we create a new set of dummy (binary) variables that is equal to the number of categories (k) in the variable. For example, let's say we have a categorical variable Color with three categories called "Red", "Green" and "Blue", we need to use three dummy variables to encode this variable using one-hot encoding. A dummy (binary) variable just takes the value 0 or 1 to indicate the exclusion or inclusion of a category.

In one-hot encoding,

"Red" color is encoded as **[1 0 0]** vector of size 3.

"Green" color is encoded as **[0 1 0]** vector of size 3.

"Blue" color is encoded as **[0 0 1]** vector of size 3.

One-hot encoding on iris dataset: For iris dataset the target column which is Species. It contains three species Iris-setosa, Iris-versicolor, Iris-virginian.

Sklearn Functions for One-hot Encoding:

- **sklearn.preprocessing.OneHotEncoder():** Encode categorical integer features using a one-hot aka one-of-K scheme

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing from sklearn import preprocessing

Step 2: Load the iris dataset in dataframe object df

Step 3: Observe the unique values for the Species column.

df['Species'].unique()

output:array(['Iris-setosa','Iris-versicolor', 'Iris-virginica'], dtype=object)

Step 4: Apply label_encoder object for label encoding the Observe the unique values for the Species column.

df['Species'].unique()

Output: array([0, 1, 2], dtype=int64)

Step 5: Remove the target variable fromdataset

features_df=df.drop(columns=['Species'])

Step 6: Apply one_hot encoder for Species column.

```
enc = preprocessing.OneHotEncoder()
enc_df=pd.DataFrame(enc.fit_transform(df[['Species']])).toarray()
```

Step 7: Join the encoded values with Features variable

df_encode = features_df.join(enc_df)

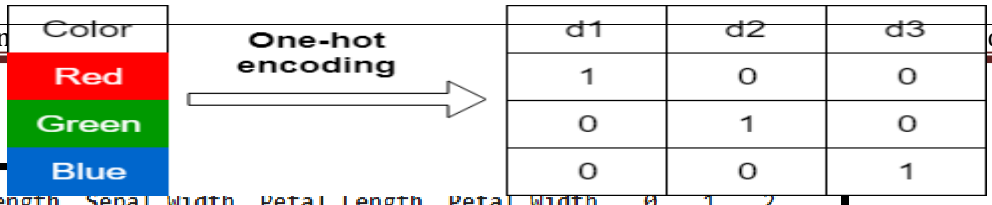
Step 8: Observe the merge dataframe df_encode

Step 9: Rename the newly encoded columns.

df_encode.rename(columns = {0:'Iris-Setosa',1:'Iris-Versicolor',2:'Iris-virginica'}, inplace = True)

Step 10: Observe the merge dataframe

df_encode



	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	0	1	2
0	5.1	3.5	1.4	0.2	1.0	0.0	0.0
1	4.9	3.0	1.4	0.2	1.0	0.0	0.0
2	4.7	3.2	1.3	0.2	1.0	0.0	0.0
3	4.6	3.1	1.5	0.2	1.0	0.0	0.0
4	5.0	3.6	1.4	0.2	1.0	0.0	0.0

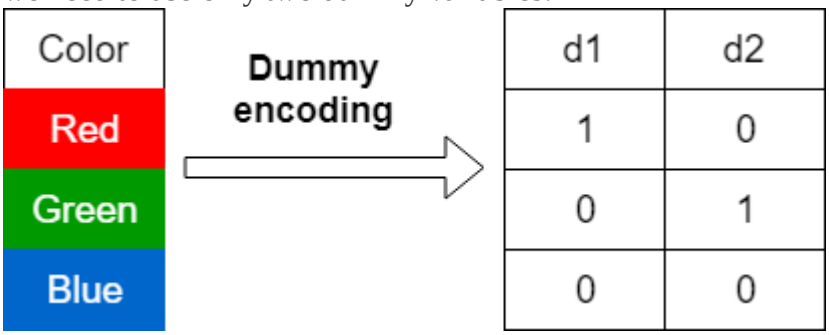
Output after Step 8

Output after Step 10:

	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Iris-Setosa	Iris-Versicolor	Iris-virginica
0	5.1	3.5	1.4	0.2	1.0	0.0	0.0
1	4.9	3.0	1.4	0.2	1.0	0.0	0.0
2	4.7	3.2	1.3	0.2	1.0	0.0	0.0
3	4.6	3.1	1.5	0.2	1.0	0.0	0.0
4	5.0	3.6	1.4	0.2	1.0	0.0	0.0

Dummy Variable Encoding

Dummy encoding also uses dummy (binary) variables. Instead of creating a number of dummy variables that is equal to the number of categories (k) in the variable, dummy encoding uses k-1 dummy variables. To encode the same Color variable with three categories using the dummy encoding, we need to use only two dummy variables.



In dummy encoding,

"Red" color is encoded as **[1 0]** vector of size 2. **"Green"** color is encoded as **[0 1]** vector of size 2.

"Blue" color is encoded as **[0 0]** vector of size 2.

Dummy encoding removes a duplicate category present in the one-hot encoding

Pandas Functions for One-hot Encoding with dummy variables:

pandas.get_dummies(data, prefix=None, prefix_sep='_', dummy_na=False, columns=None, sparse=False, drop_first=False, dtype=None): Convert categorical variable into dummy/indicator variables.

Parameters:

data: array-like, Series, or DataFrame Data of which to get dummy indicators.

prefixstr: list of str, or dict of str, default None String to append DataFrame column names.

prefix_sep: str, default '_'

If appending prefix, separator/delimiter to use. Or pass a list or dictionary as with prefix.

dummy_nabool: default False

Add a column to indicate NaNs, if False NaNs are ignored.

columns: list:like, default None

Column names in the DataFrame to be encoded. If column is None then all the columns with object or category type will be converted

Whether the dummy-encoded columns should be backed by a SparseArray (True) or a regular NumPy array (False).

drop_first:bool, default False

Whether to get k-1 dummies out of k categorical levels by removing the first level.

dtype: dtype, default np.uint8

Data type for new columns. Only a single dtype is allowed.

Return : DataFrame with Dummy-coded data.

Algorithm:

Step 1 : Import pandas and sklearn library for preprocessing from sklearn import preprocessing

Step 2: Load the iris dataset in dataframe object df

Step 3: Observe the unique values for the Species column.

df['Species'].unique()

output:array(['Iris-setosa','Iris-versicolor', 'Iris-virginica'], dtype=object)

Step 4: Apply label_encoder object for label encoding the Observe the unique values for the Species column.

df['Species'].unique()

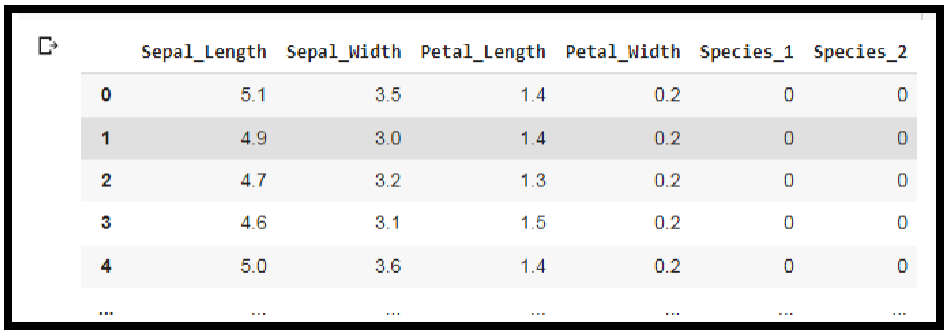
Output: array([0, 1, 2], dtype=int64)

Step 6: Apply one_hot encoder with dummy variables for Species column.

one_hot_df = pd.get_dummies(df, prefix="Species", columns=['Species'], drop_first=True)

Step 7:Observe the merge dataframe

one_hot_df



	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Species_1	Species_2
0	5.1	3.5	1.4	0.2	0	0
1	4.9	3.0	1.4	0.2	0	0
2	4.7	3.2	1.3	0.2	0	0
3	4.6	3.1	1.5	0.2	0	0
4	5.0	3.6	1.4	0.2	0	0
...	...	...	...	...	...	...

Conclusion-In this way we have explored the functions of the python library for Data Preprocessing, Data Wrangling Techniques and How to Handle missing values on Iris Dataset.

Viva Questions

1. Explain Data Frame with Suitable example.
2. What is the limitation of the label encoding method?
3. What is the need of data normalization?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

AssignmentNo:2

Title of the Assignment: Data Wrangling, II

Create an Academic performance dataset of students and perform the following operations using Python.

1. Scan all variables for missing values and inconsistencies. If there are missing values and/or inconsistencies, use any of the suitable techniques to deal with them.
2. Scan all numeric variables for outliers. If there are outliers, use any of the suitable techniques to deal with them.
3. Apply data transformations on at least one of the variables. The purpose of this transformation should be one of the following reasons: to change the scale for better understanding of the variable ,to convert a non-linear relation into a linear one to decrease the skewness and convert the distribution into a normal distribution.

   Reason and document your approach properly.

Objective of the Assignment: Students should be able to perform the data wrangling operation using Python on any open source dataset.

Outcome: To scan all variables for missing values and inconsistencies, Scan all numeric variables for outliers and apply data transformation to change the scale for better understanding of the variable.

Contents for Theory:

1. **Creation of Dataset using Microsoft Excel.**
2. **Identification and Handling of Null Values**
3. **Identification and Handling of Outliers**
4. **Data Transformation for the purpose of:**
   a. **To change the scale for better understanding**
   b. **To decrease the skewness and convert distribution into normal distribution**

Theory:

1. **Creation of Dataset using Microsoft Excel.**

   The dataset is created in "CSV" format.

   - The name of dataset is **Students Performance**
   - **The features of the dataset are:** Math _Score ,Reading_Score, Writing_Score, Placement_Score, Club_Join_Date .
   - **Number of Instance**s:30
   - **The response variable is**: Placement_ Offer _Count.
   - **Range of Values:**

     Math_Score [60-80], Reading_Score[75-,95], ,Writing_Score [60,80], Placement_Score[75-100], Club_Join_Date [2018-2021].
   - **The response variable is** the number o fplacement offers facilitated top articular students, which is largely depend on Placement_Score

To fill the values in the dataset the **RANDBETWEEN** is used. Returns a random integer number between the numbers you specify

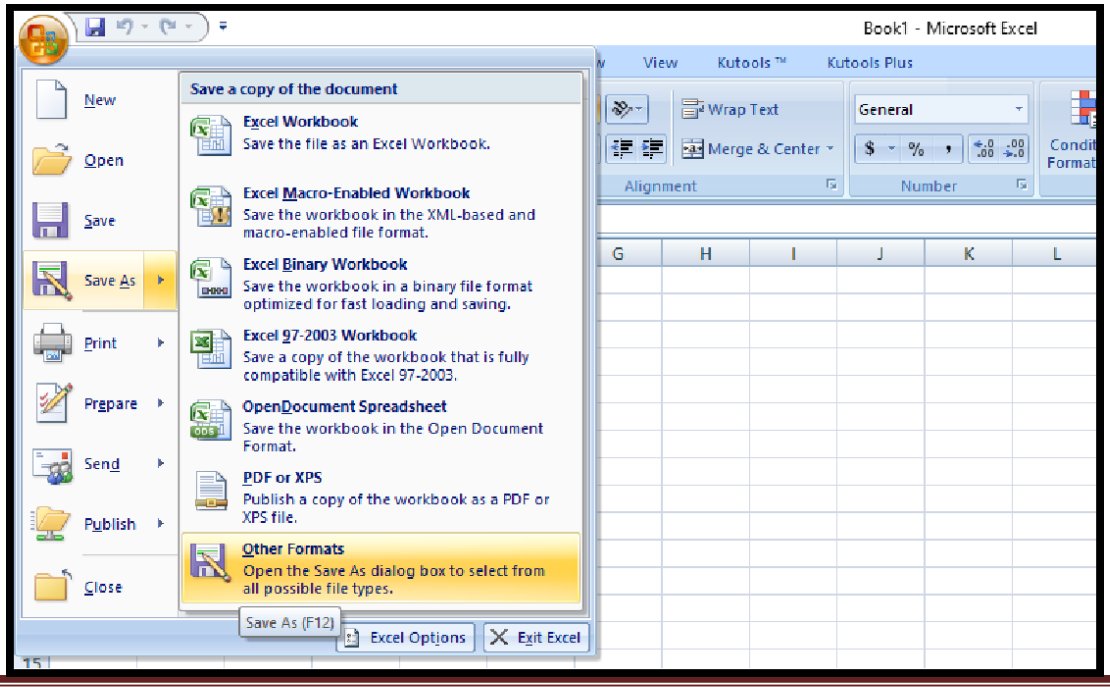
Syntax : RANDBETWEEN(bottom, top) Bottom The smallest integer and

Top The largest integer RANDBETWEEN will return.

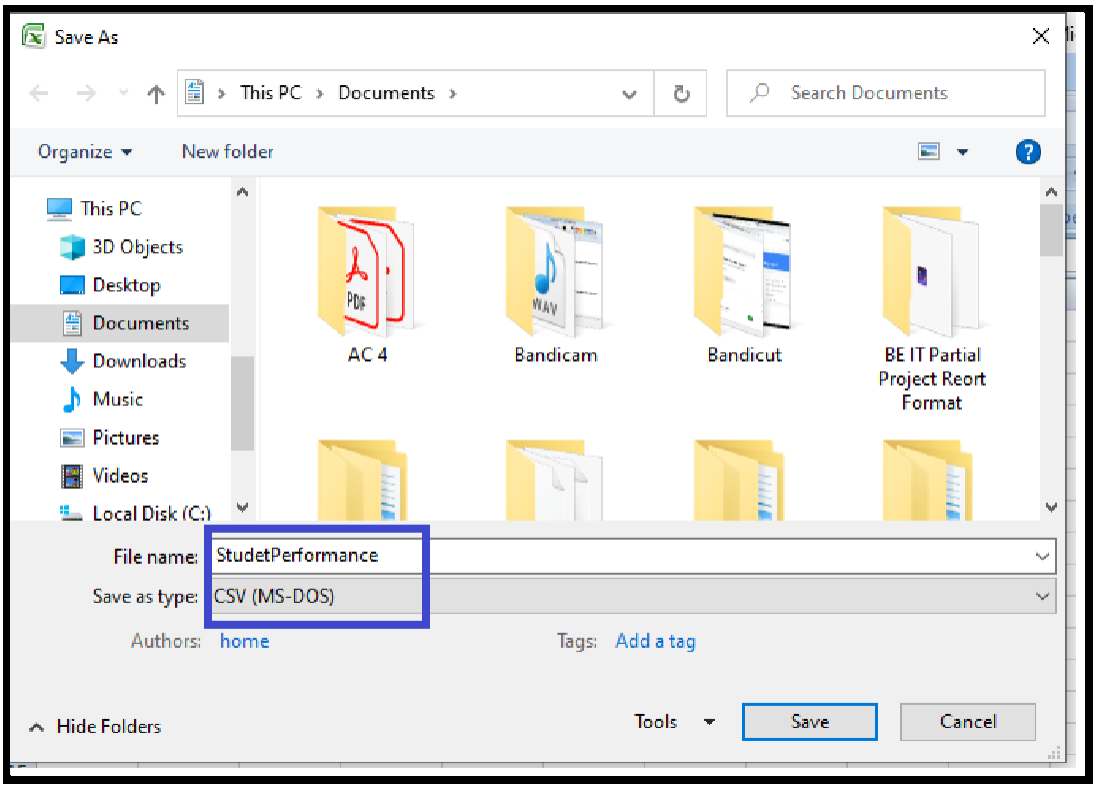
For better understanding and visualization, 20% impurities are added into each variable to the dataset.

The step to create the dataset are as follows:

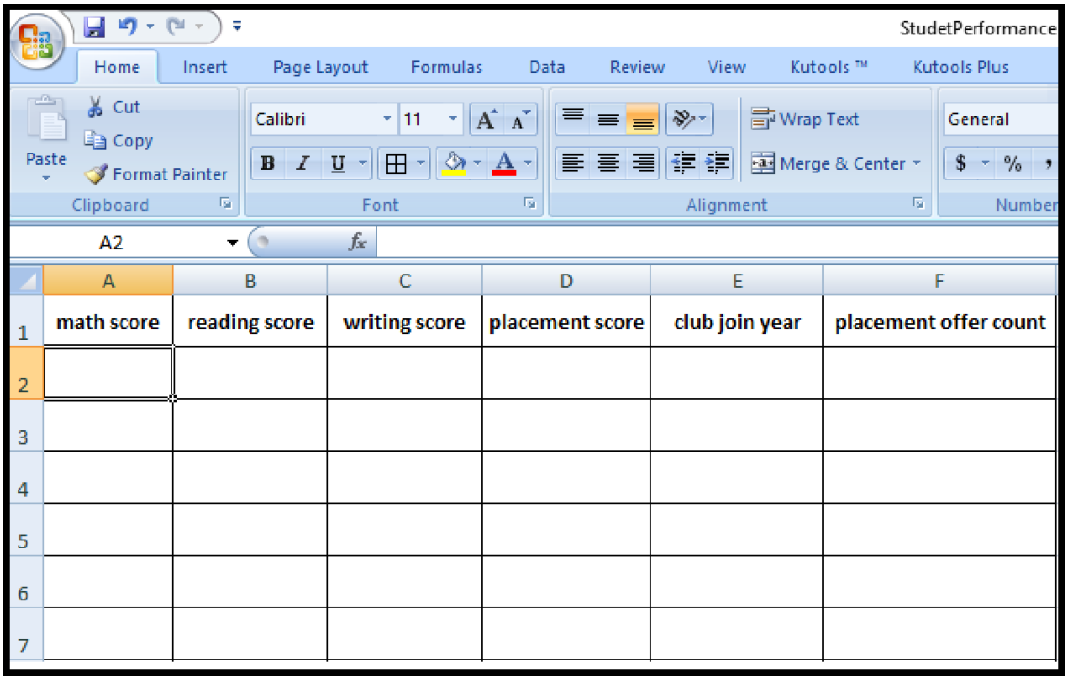
Step 1: Open Microsoft Excel and click on Save As. Select Other .Formats



Step 2: Enter the name of the dataset and Save the dataset as tye CSV(MS-DOS).

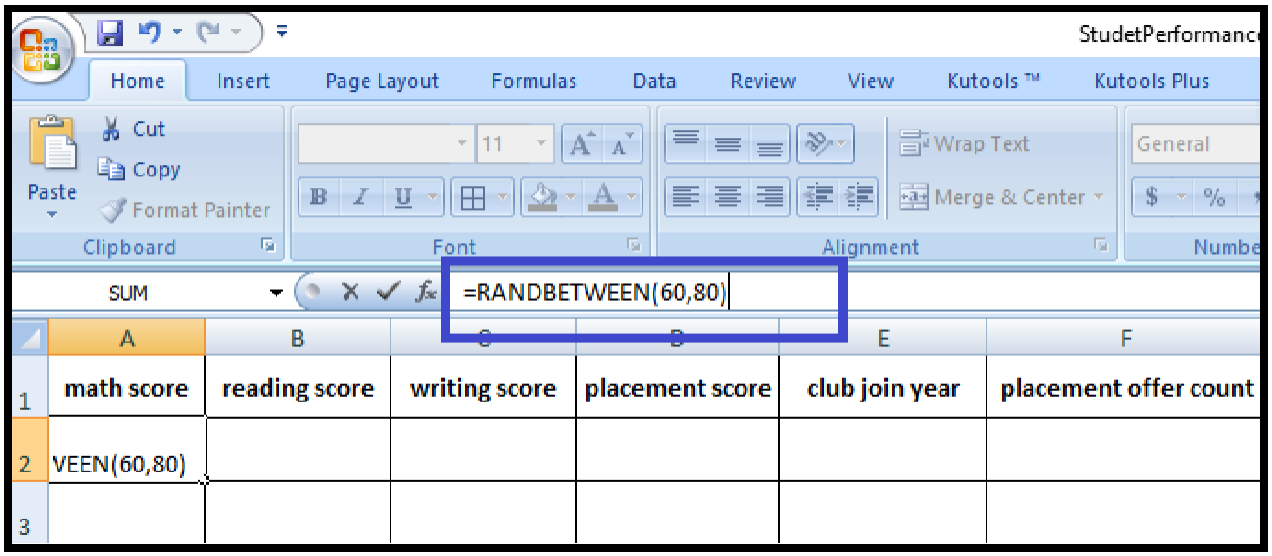


Step3: Enter the name of features as column header.



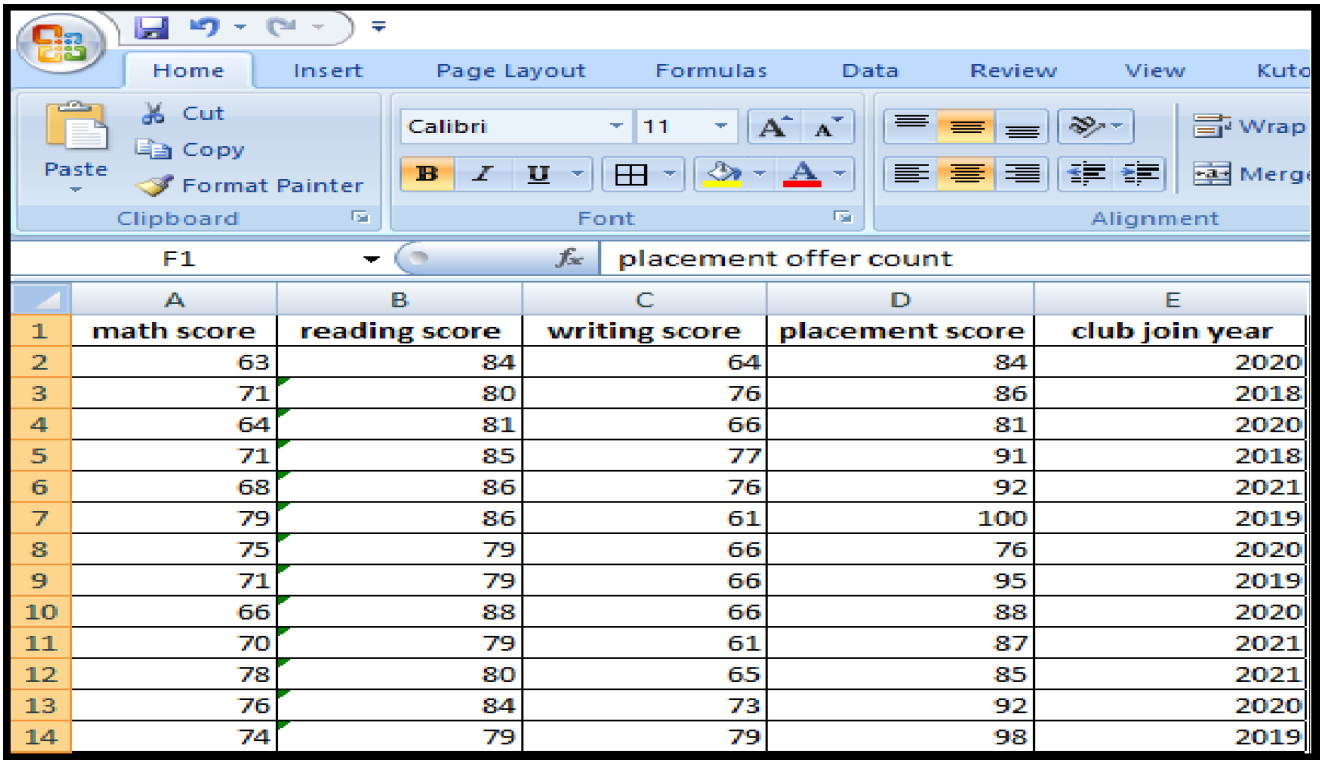
Step4: Fill the dara by using **RANDOMBETWEEN** function. For every feature, fill the data by considering above spectified range.

one example is given:



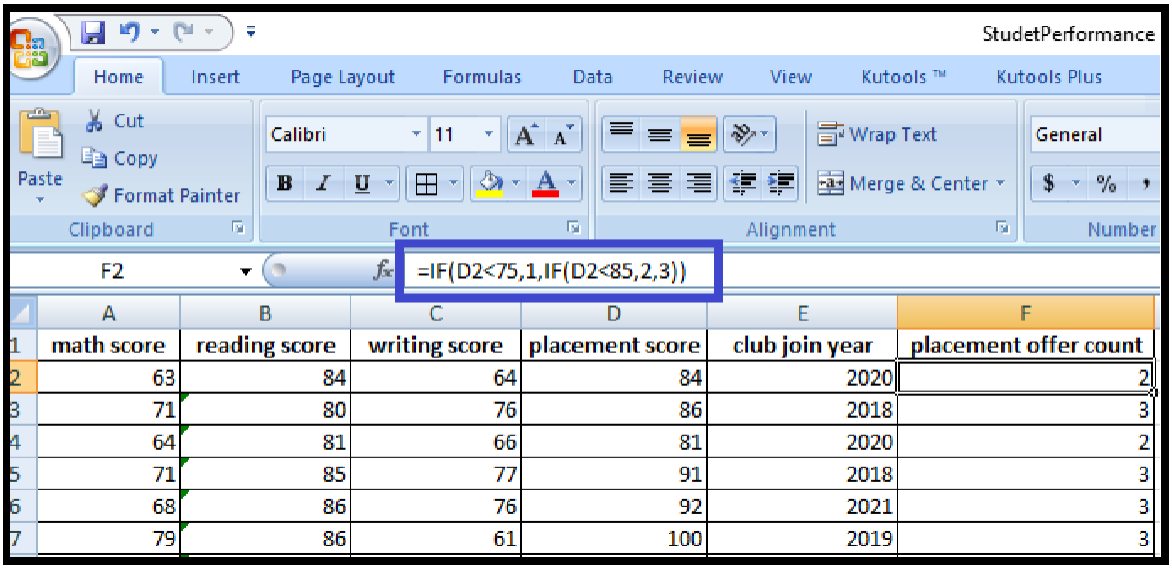
Scroll down the cursor for 30 rows to create 30 instances.

Repeat this for the features, Reading_Score, Writing_Score Placement_Score,Club_Join_Date.



	A	B	C	D	E
1	math score	reading score	writing score	placement score	club join year
2	63	84	64	84	2020
3	71	80	76	86	2018
4	64	81	66	81	2020
5	71	85	77	91	2018
6	68	86	76	92	2021
7	79	86	61	100	2019
8	75	79	66	76	2020
9	71	79	66	95	2019
10	66	88	66	88	2020
11	70	79	61	87	2021
12	78	80	65	85	2021
13	76	84	73	92	2020
14	74	79	79	98	2019

The placement count largely depends on the placement score. It is considered that if placement score <75,1 offer is facilitated; for placement score >75 , 2 offer is facilitated and for else (>85)3 offer is facilitated. Nested If formula is used for ease of data filling.



	A	B	C	D	E	F
1	math score	reading score	writing score	placement score	club join year	placement offer count
2	63	84	64	84	2020	2
3	71	80	76	86	2018	3
4	64	81	66	81	2020	2
5	71	85	77	91	2018	3
6	68	86	76	92	2021	3
7	79	86	61	100	2019	3

Step4: In 20% data ,fill the impurities. The range of math scoreis[60,80], updating a few instances values below 60 or above 80. Repeat this for Writing_Score [60,80], Placement_Score[75-100], Club_Join_Date [2018-2021].

	A	B	C	D	E
1	math score	reading score	writing score	placement score	club join year
2	68	94	64	90	2018
3	72	85	70	86	2018
4	94	90	64	91	2020

Step 5: To violate the rule of response variable, update few valus . If placement score is greater then 85, facilated only1 offer.

	A	B	C	D	E	F
1	math score	reading score	writing score	placement score	club join year	placement offer count
2	70	91	64	87	2019	3
3	77	75	67	81	2020	2
4	94	84	73	99	2019	3
5	78	84	77	96	2020	1

The dataset is created with the given description.

2. Identification and Handling of Null Values

Missing Data can occur when no information is provided for one or more items or for a whole unit. Missing Data is a very big problem in real-life scenarios. Missing Data can also refer to as NA(Not

Available) values in pandas. In Data Frame sometimes many datasets simply arrive with missing data, either because it exists and was not collected or it never existed. For Example, Suppose different users being surveyed may choose not to share their income, some users may choose not to share the address in this way many datasets went missing.

In Pandas missing data is represented by two value:

1. None: None is a Python singleton object that is often used for missing data in Python code.

2.NaN:NaN(an acronym for Not a Number), is a special floating-point value recognized by all systems that use the standard IEEE floating-point representation.

Pandas treat None and NaN as essentially interchangeable for indicating missing o rnull values.To facilitate this convention, there are several useful functions for detecting, removing, and replacing null values in Pandas DataFrame :

isnull()

notnull()

dropna()

fillna()

replace()

1. Checking for missing values using is null() and not null()

Checking for missing values using is null()

In order to check null values in Pandas DataFrame , isnull() function is used. This function return dataframe of Boolean values which are True for NaN values.

Algorithm:

Step 1 : Import pandas and numpy in order to check missing values in Pandas DataFrame

```
import pandas as pd
import numpy as np
```

Step 2: Load the dataset in dataframe object df

```
df=pd.read_csv("/content/StudentsPerformanceTest1.csv")
```

Step 3: Display the data frame

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
1	female	69	90	88.0	NaN	2	na
2	female	90	95	93.0	74.0	2	Nashik
3	male	47	57	NaN	78.0	1	Na
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik
7	male	NaN	65	67.0	49.0	1	Pune
8	male	5	77	89.0	55.0	0	NaN

Step 4: Use isnull() function to check null values in the dataset.

df.isnull()

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	False	False	False	False	False	False	False
1	False	False	False	False	True	False	False
2	False	False	False	False	False	False	False
3	False	False	False	True	False	False	False
4	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False
7	False	True	False	False	False	False	False
8	False	False	False	False	False	False	True

Step 5: To create a series true for NaN values for specific columns. for example math score in dataset and display data with only math score as NaN

series = pd.isnull(df["math score"])df[series]

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
7	male	NaN	65	67.0	49.0	1	Pune

Checking formissing values using notnull()

In order to check null values in Pandas Dataframe, notnull() function isused. This function return dataframe of Boolean values which are False for NaN values.

Algorithm:

Step 1 : Import pandas and numpy in order to check missing values in Pandas DataFrame

```
import pandas as pd
import numpy as np
```

Step 2: Load the dataset in dataframe object df

```
df=pd.read_csv("/content/StudentsPerformanceTest1.csv")
```

Step 3: Display the data frame **df**

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
1	female	69	90	88.0	NaN	2	na
2	female	90	95	93.0	74.0	2	Nashik
3	male	47	57	NaN	78.0	1	Na
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik
7	male	NaN	65	67.0	49.0	1	Pune
8	male	5	77	89.0	55.0	0	NaN

Step 4: Use notnull() function to check null values in the dataset.

```
df.notnull()
```

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	True	True	True	True	True	True	True
1	True	True	True	True	False	True	True
2	True	True	True	True	True	True	True
3	True	True	True	False	True	True	True
4	True	True	True	True	True	True	True
5	True	True	True	True	True	True	True
6	True	True	True	True	True	True	True
7	True	False	True	True	True	True	True
8	True	True	True	True	True	True	False

Step 5: To create a series true for NaN values for specific columns. for example emath score in

dataset and display data with only math score as NaN

series1 = pd.notnull (df["math score"])df[series1]

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
1	female	69	90	88.0	NaN	2	na
2	female	90	95	93.0	74.0	2	Nashik
3	male	47	57	NaN	78.0	1	Na
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik
8	male	5	77	89.0	55.0	0	NaN

See that there are also categorical values in the dataset, for this, you need to use Label Encoding or One Hot Encoding.

from sklearn.preprocessing import LabelEncoderle = LabelEncoder()

df['gender'] = le.fit_transform(df['gender'])newdf=df

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	0	72	72	74.0	78.0	1	Pune
1	0	69	90	88.0	NaN	2	na
2	0	90	95	93.0	74.0	2	Nashik
3	1	47	57	NaN	78.0	1	Na
4	1	na	78	75.0	81.0	3	Pune
5	0	71	Na	78.0	70.0	4	na
6	1	12	44	52.0	12.0	2	Nashik
7	1	NaN	65	67.0	49.0	1	Pune
8	1	5	77	89.0	55.0	0	NaN

Filling missing values using dropna(), fillna(), replace()

Inorder to fill null values in a datasets, fillna(), replace() functions are used. These functions replace NaN values with some value of their own. All these functions help in filling null values in datasets of a DataFrame.

For replacing null values with NaN
missing_values = ["Na", "na"]

df=pd.read_csv("StudentsPerformanceTest1.csv",na_values=missing_values)

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72.0	72.0	74.0	78.0	1	Pune
1	female	69.0	90.0	88.0	NaN	2	NaN
2	female	90.0	95.0	93.0	74.0	2	Nashik
3	male	47.0	57.0	NaN	78.0	1	NaN
4	male	NaN	78.0	75.0	81.0	3	Pune
5	female	71.0	NaN	78.0	70.0	4	NaN
6	male	12.0	44.0	52.0	12.0	2	Nashik
7	male	NaN	65.0	67.0	49.0	1	Pune
8	male	5.0	77.0	89.0	55.0	0	NaN

Filling null values with a single value

Step 1 : Import pandas and numpy in order to check missing values in Pandas DataFrame

```
import pandas as pd

import numpy as np
```

Step 2: Load the dataset in dataframe object df

```
df=pd.read_csv("/content/StudentsPerformanceTest1.csv")
```

Step 3: Display the data frame **df**

Step 4: filling missing value using fillna() ndf=dfndf.fillna(0)

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
1	female	69	90	88.0	0.0	2	na
2	female	90	95	93.0	74.0	2	Nashik
3	male	47	57	0.0	78.0	1	Na
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik
7	male	0	65	67.0	49.0	1	Pune
8	male	5	77	89.0	55.0	0	0

Step5:filling missing values using mean, median and standard deviation of that column.

```
data['math score'] = data['math score'].fillna(data['math score'].mean())

data["math score"]  = data["math score"].fillna(data["mathscore"].median())

data['math score"] = data["math score"].fillna(data["math score"].std())
```

replacing missing values in for enoon column with minimum/maximum number of that column

```
data["math score"] = data["math score"].fillna(data["math score"].min())

data["math score"] = data["math score"].fillna(data["math score"].max())
```

Filling null values in dataset

To fill null values in dataset using place=true

```
m_v=df['math score'].mean()

df['math score'].fillna(value=m_v, inplace=True)df
```

	gender	math score	reading score	writing score	Placement Score	placement offer count
0	female	72.000	72	74	78	1
1	female	69.000	90	88	70	2
2	female	90.000	95	93	74	2
3	male	47.000	57	44	78	1
4	male	11.000	78	75	81	3
5	female	71.000	83	78	70	4
6	male	12.000	44	52	12	2
7	male	47.125	65	67	49	1
8	male	5.000	77	89	55	0

Filling a null values using replace() method

Following line will replaceNan value in dataframe with value -99

```
ndf.replace(to_replace = np.nan, value = -99)
```

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
1	female	69	90	88.0	-99.0	2	na
2	female	90	95	93.0	74.0	2	Nashik
3	male	47	57	-99.0	78.0	1	Na
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik
7	male	-99	65	67.0	49.0	1	Pune
8	male	5	77	89.0	55.0	0	-99

Deleting null values using dropna() method

In order to drop null values from a dataframe, dropna()f unction is used. This function drops Rows/Columns of datasets with Null values in different ways.

1. Dropping rows with at least 1 null value
2. Dropping rows if all values in that row are missing
3. Dropping columns with at least 1 null value.
4. Dropping Rows with at least 1 null value in CSV file

Algorithm:

Step 1 : Import pandas and numpy in order to check missing values in Pandas DataFrame

```
import pandas as pd

import numpy as np
```

Step 2: Load the dataset in dataframe object df

```
df=pd.read_csv("/content/StudentsPerformanceTest1.csv")
```

Step 3: Display the data frame **df**

Step 4:To drop rows with at least 1 null value

ndf.dropna()

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
2	female	90	95	93.0	74.0	2	Nashik
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik

Step5: To Drop rows if all values in that row are missing

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
1	female	69	90	88.0	NaN	2	na
2	female	90	95	93.0	74.0	2	Nashik
3	male	47	57	NaN	78.0	1	Na
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik
7	male	NaN	65	67.0	49.0	1	Pune
8	male	5	77	89.0	55.0	0	NaN

ndf.dropna(how = 'all')

Step6: To Drop columns with atleast1 null value.

	gender	reading score	placement offer count
0	female	72	1
1	female	90	2
2	female	95	2
3	male	57	1
4	male	78	3
5	female	Na	4
6	male	44	2
7	male	65	1
8	male	77	0

ndf.dropna(axis = 1)

Step7: To drop rows with atleast 1 null value in CSV file. making new data frame with dropped NA values

new_data = ndf.dropna(axis = 0, how ='any')

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72	72	74.0	78.0	1	Pune
2	female	90	95	93.0	74.0	2	Nashik
4	male	na	78	75.0	81.0	3	Pune
5	female	71	Na	78.0	70.0	4	na
6	male	12	44	52.0	12.0	2	Nashik

new_data

Identification and Handling of Outliers

Identification of Outliers

One of the most important steps as part of data preprocessing is detecting and treating the outliers as they can negatively affect the statistical analysis and the training process of a machine learning algorithm resulting in lower accuracy.

What are Outliers?

An Outlier is an observation in a given dataset that lies far from the rest of the observations. That means an outlier is vastly larger or smaller than the remaining values in the set.

Why do they occur?

An outlier may occur due to the variability in the data, or due to experimental error/human error.

They may indicate an experimental error or heavy skewness in the data(heavy-tailed distribution).

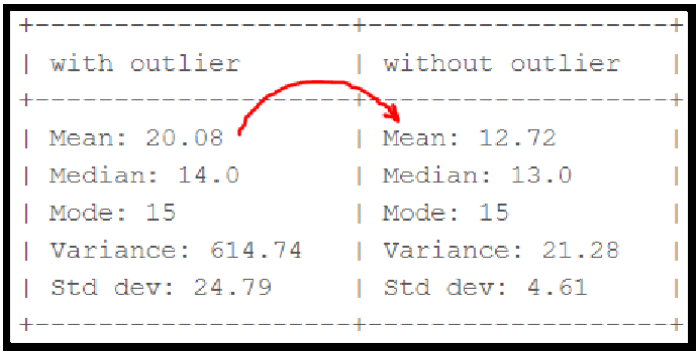
What do they affect?

In statistics, we have three measures of central tendency namely Mean, Median, and Mode. They help us describe the data.

Mean is the accurate measure to describe the data when we do not have any outliers present. **Median** is used if there is an outlier in the dataset.
Mode is used if there is an outlier AND about ½ or more of the data is the same.

Example:

Consider a small dataset, sample= [15, 101, 18, 7, 13, 16, 11, 21, 5, 15, 10, 9]. By looking at it, one can quickly say '101' is an outlier that is much larger than the other values.



with outlier	without outlier
Mean: 20.08	Mean: 12.72
Median: 14.0	Median: 13.0
Mode: 15	Mode: 15
Variance: 614.74	Variance: 21.28
Std dev: 24.79	Std dev: 4.61

fig. Computation with and without outlier

From the above calculations, we can clearly say the Mean is more affected than the Median.

Detecting Outliers

If our dataset is small, we can detect the outlier by just looking at the dataset. But what if we have a huge dataset, how do we identify the outliers then? We need to use visualization and mathematical techniques.

Below are some of the techniques of detecting outliers

Box plots, Scatter plots, Z-score, Inter Quintile Range (IQR) Detecting outliers using Box plot:

It captures the summary of the data effectively and efficiently with only a simple box and whiskers. Box plot summarizes sample data using 25th, 50th, and75th percentiles. One can just get insights(quartiles, median, and outliers) into the dataset by just looking at its boxplot.

Algorithm:

Step 1 : Import pandas and numpy libraries

```
import pandas as pd

import numpy as np
```

Step 2: Load the dataset in dataframe object df

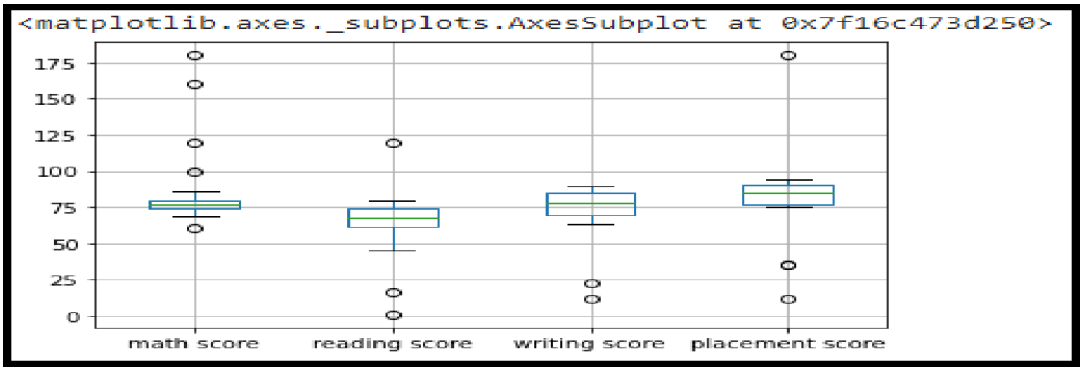
```
df=pd.read_csv("/content/demo.csv")
```

Step 3: Display the data frame **df**

	math score	reading score	writing score	placement score	placement offer count
0	80	68	70	89	3
1	71	61	85	91	3
2	79	16	87	77	2
3	61	77	74	76	2
4	78	71	67	90	3
5	73	68	90	80	2
6	77	62	70	35	2
7	74	45	80	12	1
8	76	60	79	77	2
9	75	65	85	87	3
10	160	67	12	83	2
11	79	72	88	180	2
12	80	80	78	94	3

Step 4:Select the columns for boxplot and draw the boxplot.

```
col = ['math score', 'reading score' , 'writingscore','placement score']

df.boxplot(col)
```



Step5: We can now print the outliers for each column with reference to the boxplot.

```
print(np.where(df['math score']>90))
print(np.where(df['reading
score']<25))print(np.where(df['writingscore']<30))
```

Detecting outliers using Scatter plot:

It is used when you have paired numerical data, or when your dependent variable has multiple values for each reading independent variable, or when trying to determine the relationship between the two variables. In the process of utilizing the scatter plot, one can also use it for outlier detection.

Placement score and Placement count features are used.

Algorithm:

Step 1 : Import pandas , num py and mat plot lib libraries

```
import pandas as pd import numpy as np

import mat plot lib.py plot as plt
```

Step 2: Load the dataset in dataframe object df

```
df=pd.read_csv("/content/demo.csv")
```

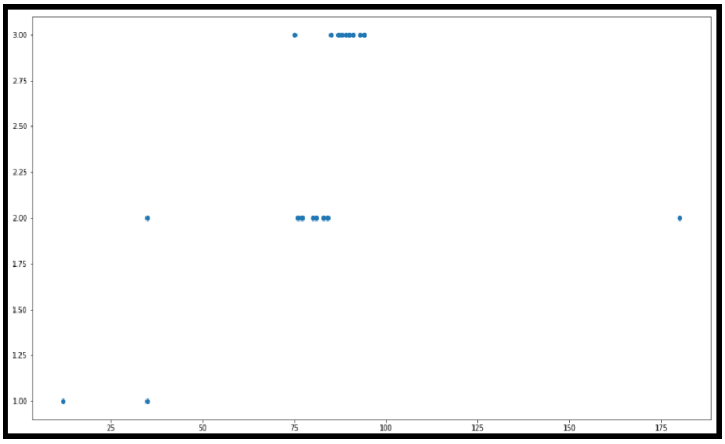
Step 3: Display the data frame **df**

Step4: Draw the scatter plot with placement score and placement offer count **fig, ax = plt.subplots(figsize = (18,10))ax.scatter(df['placement score'], df['placement offer count'])**

plt.show()

Labels to the axis can be assigned (Optional)

```
ax.set_xlabel('(Proportion non-retail businessacres)/(town)')

ax.set_ylabel('(Full-value property-tax rate)/($10,000)')
```



Step5: We can now print the outliers with reference to scatter plot.

```
print(np.where((df['placementscore']<50)&(df['placementoffer
count']>1)))print(np.where((df['placement score']>85)&(df['placement offer count']<3)))
```

Detecting outliers using Z-Score:

Z-Score is also called a standard score. This value/score helps to understand how far is the data point from the mean. And after setting up a threshold value one can utilize z score values of data points to define the outliers.

Zscore = (data_point -mean) / std. deviation

Algorithm:

Step 1 : Import num py and stats from scipy libraries

```
import numpy as np

from scipy import stat
```

Step 2: Calculate Z-Score for math score column

```
z = np.abs(stats.zscore(df['math score']))
```

Step3: PrintZ-ScoreValue.Itprintsthez-score values of each data item of the column

```
print(z)
```

```
[0.17564553 0.5282877  0.21482799 0.92011234 0.25401045 0.44992277
 0.29319292 0.41074031 0.33237538 0.37155785 2.95895157 0.21482799
 0.17564553 0.25401045 0.37155785 0.25401045 0.05944926 0.17564553
 0.37155785 0.0972806  0.60665263 0.60800375 0.48910524 0.41074031
 0.37155785 3.74260085 0.48910524 0.5282877  1.39165302]
```

Step 4: Now to define an outlier threshold value is chosen.

threshold = 0.18

Step 5: Display the sample outliers

```
sample_outliers = np.where(z <threshold)sample_outliers
```

```
(array([ 0, 12, 16, 17, 19]),)
```

Detecting outliers using Inter Quantile Range(IQR):

IQR (Inter Quartile Range) Inter Quartile Range approach to finding the outliers is the most commonly used and most trusted approach used in the research field.

IQR = Quartile3 – Quartile1

To define the outlier base value is defined above and below data sets normal range namely Upper and Lower bounds, define the upper and the lower bound (1.5*IQR value is considered) :

upper = Q3 +1.5*IQR lower=Q1–1.5*IQR

In the above formula as according to statistics, the 0.5 scale-up of IQR (new_IQR = IQR + 0.5*IQR) is taken.

Algorithm:

Step 1 : Import num py library

```
import num py as np
```

Step 2: Sort Reading Score feature and store it into sorted_rscore.

```
sorted_rscore= sorted(df['reading score'])
```

Step 3: Print sorted_rscore

```
sorted_rscore
```

Step 4: Calculate and print Quartile 1 and Quartile 3

```
q1 = np.percentile(sorted_rscore,25 ) q3 = np.percentile(sorted_rscore,75)
print(q1,q3)
```

```
62.0 74.0
```

Step 5: Calculate value of IQR (Inter Quartile Range)

```
IQR = q3-q1
```

Step 6: Calculate and print Upper and Lower Bound to define the outlier base value.

```
lwr_bound = q1-(1.5*IQR)upr_bound = q3+(1.5*IQR)print(lwr_bound, upr_bound)
```

```
44.0 92.0
```

Step 7: Print Outliers

```
r_outliers = []
for i in sorted_rscore:
```

```
if (i<lwr_bound or i>upr_bound):r_outliers.append(i)

print(r_outliers)
```

```
[1, 16, 120]
```

Handling of Outliers:

For removing the outlier, one must follow the same process of removing an entry from the dataset using its exact position in the dataset because in all the above methods of detecting the outliers end result is the list of all those data items that satisfy the outlier definition according to the method used. Below are some of the methods of treating the outliers

Trimming/removing the outlier

Quantile based flooring and capping

Mean/Median imputation

Trimming/removing the outlier:

In this technique, we remove the outliers from the dataset. Although it is not a good practice to follow.

```
new_df=df

for i in sample_ outliers: new _df .drop (i,inplace=True )
```

	math score	reading score	writing score	placement score	placement offer count
1	71	61	85	91	3
2	79	16	87	77	2
3	61	77	74	76	2
4	78	71	67	90	3
5	73	68	90	80	2
6	77	62	70	35	2
7	74	45	80	12	1
8	76	60	79	77	2
9	75	65	85	87	3
10	160	67	12	83	2
11	79	72	88	180	2
13	78	69	71	90	3
14	75	1	71	81	2
15	78	62	79	93	3
18	75	62	86	87	3

New df

	math score	reading score	writing score	placement score	placement offer count
1	71	61	85	91	3
2	79	16	87	77	2
3	61	77	74	76	2
4	78	71	67	90	3
5	73	68	90	80	2
6	77	62	70	35	2
7	74	45	80	12	1
8	76	60	79	77	2
9	75	65	85	87	3
10	160	67	12	83	2
11	79	72	88	180	2
13	78	69	71	90	3
14	75	1	71	81	2
15	78	62	79	93	3
18	75	62	86	87	3

Here Sample_ outliers arrays `(array([ 0, 12, 16, 17 ]),)` So instances with index 0,12 ,16 and 17 are deleted.

Quantile based flooring and capping:

In this technique, the outlier is capped at a certain value above the 90th percentile value or floored at a factor below the 10th percentile value **df = pd.read_csv("/demo.csv")**

df_stud=df ninetieth_percentile = np.percentile(df_stud['math score'], 90)

b = np.where(df_stud['math score']>ninetieth_percentile,ninetieth_percentile,

df_stud['math score'])

```
New array: [ 80.  71.  79.  61.  78.  73.  77.  74.  76.  75. 104.  79.  80.  78.
  75.  78.  86.  80.  75.  82.  69. 100.  72.  74.  75. 104.  72.  71.
 104.]
```

print("New array:",b)

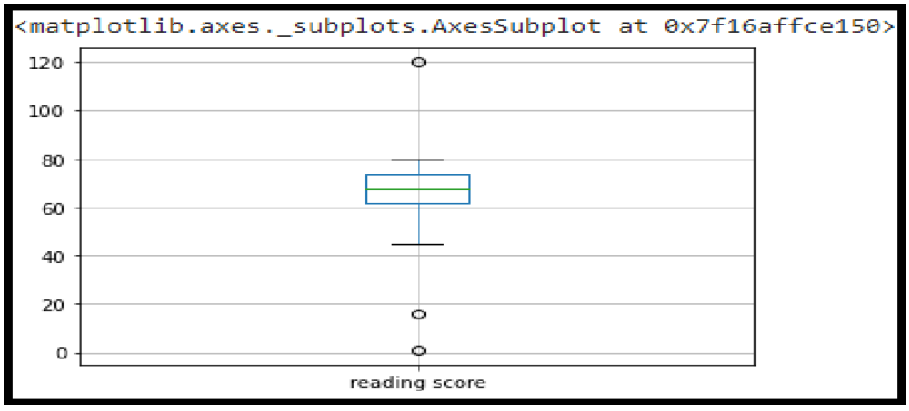
df_stud.insert(1,"m score",b,True)df_stud

	math score	m score	reading score	writing score	placement score	placement offer count
0	80	80.0	68	70	89	3
1	71	71.0	61	85	91	3
2	79	79.0	16	87	77	2
3	61	61.0	77	74	76	2
4	78	78.0	71	67	90	3
5	73	73.0	68	90	80	2
6	77	77.0	62	70	35	2
7	74	74.0	45	80	12	1

Mean/Median imputation:

As the mean value is highly influenced by the outliers, it is advised to replace the outliers with the median value.

1. Plot the box plot for reading score col = ['reading score'] df.boxplot(col)



2. Outliers are seen in box plot.
3. Calculate the median of reading score by using sorted_score

Median = np.median(sorted_rscore)median

1. Replace the upper bound outliers using median value

refined_df=df

refined_df['reading score'] = np.where(refined_df['reading score'] >upr_bound, median,refined_df['reading score'])

2.Display redefined_df

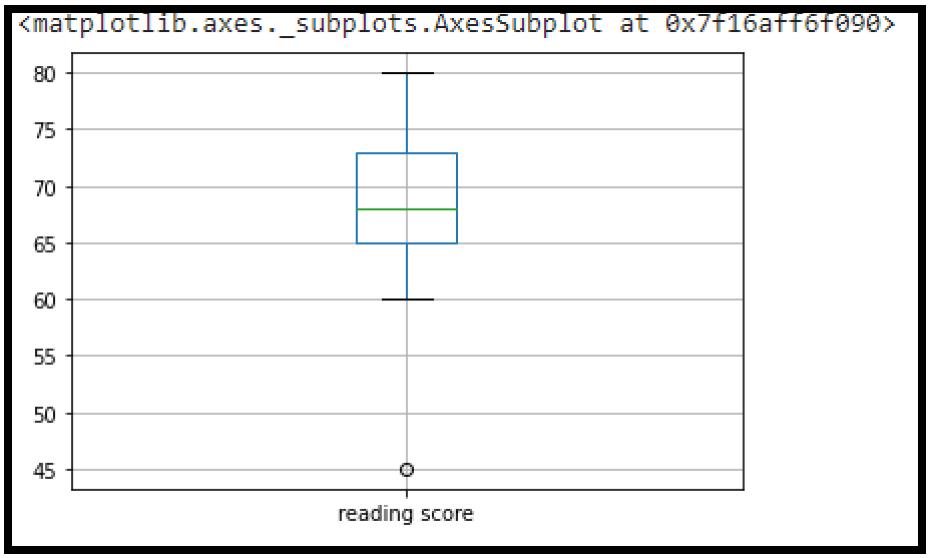
	math score	m score	reading score	writing score	placement score	placement offer count
0	80	80.0	68.0	70	89	3
1	71	71.0	61.0	85	91	3
2	79	79.0	16.0	87	77	2
3	61	61.0	77.0	74	76	2
4	78	78.0	71.0	67	90	3
5	73	73.0	68.0	90	80	2
6	77	77.0	62.0	70	35	2
7	74	74.0	45.0	80	12	1
8	76	76.0	60.0	79	77	2
9	75	75.0	65.0	85	87	3
10	160	104.0	67.0	12	83	2

3.Replace the lower bound outliers using median value refined_df['reading score'] = np.where(refined_df['reading score'] <lwr_bound, median,refined_df['reading score'])

4.Display redefined_df

	math score	m score	reading score	writing score	placement score	placement offer count
0	80	80.0	68.0	70	89	3
1	71	71.0	61.0	85	91	3
2	79	79.0	68.0	87	77	2
3	61	61.0	77.0	74	76	2
4	78	78.0	71.0	67	90	3
5	73	73.0	68.0	90	80	2
6	77	77.0	62.0	70	35	2
7	74	74.0	45.0	80	12	1
8	76	76.0	60.0	79	77	2
9	75	75.0	65.0	85	87	3
10	160	104.0	67.0	12	83	2

Draw the box plot for redefined_dfcol = ['reading score']refined_df.boxplot(col)



Data Transformation :

Data transformation is the process of converting raw data into a format or structure that would be more suitable for model building and also data discovery in general. The process of data transformation can also be referred to as extract/transform/load(ETL). The extraction phase involves identifying and pulling data from the various source systems that create data and then moving the data to a single repository. Next, the raw data is cleansed, if needed. It's then transformed into a target format that can be fed into operational systems or into a data warehouse, a date lake or another repository for use in business intelligence and analytics applications. The data are transformed in ways that are ideal for mining the data. The data transformation involves steps that are.

Smoothing: It is a process that is used to remove noise from the dataset using some algorithms It allows for highlighting important features present in the dataset. It helps in predicting the patterns

Aggregation: Data collection or aggregation is the method of storing and presenting data in a summary format. The data may be obtained from multiple data sources to integrate these data sources into a data analysis description. This is a crucial step since the accuracy of data analysis insights is highly dependent on the quantity and quality of the data used.

Generalization: It convert slow-level data attributes to high-level data attributes

using concept hierarchy. For Example Age initially in Numerical form (22, 25) is converted into

categorical value (young, old).

Normalization: Data normalization involves converting all data variables into a given range. Some of the techniques that are used for accomplishing normalization are:

Min–max normalization: This transforms the original data linearly.

Z-score normalization: In z-score normalization (or zero-mean normalization)the values of an attribute (A), are normalized based on the mean of A and its standard deviation.

Normalization by decimal scaling: It normalizes the values of an attribute by changing the position of their decimal points

Attribute or feature construction.

New attributes constructed from the give nones: Where new attributes are created & applied to assist the mining process from the given set of attributes. This simplifies the original data & makes the mining more efficient.

To change the scale fo rbetter understanding (Attribute or feature construction)

Here the Club_Join_Dateis transferred to Duration.

Algorithm:

Step 1 : Import pandas and numpy libraries

```
import pandas as pd
import numpy as np
```

Step 2: Load the dataset in dataframe object df

```
df=pd.read_csv("/content/demo.csv")
```

Step 3: Display the data from

```
df
```

	math score	reading score	writing score	placement score	placement offer count	club join year
0	80	68	70	89	3	2019
1	71	61	85	91	3	2019
2	79	16	87	77	2	2018
3	61	77	74	76	2	2020
4	78	71	67	90	3	2019

Step 4: Change the scale of Joining year to duration.

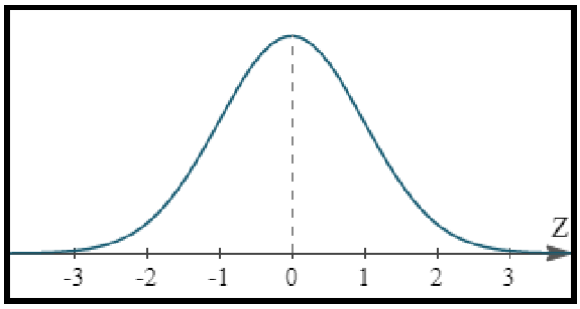
```
[11] df
```

	math score	reading score	writing score	placement score	placement offer count	club join year	Duration
0	80	68	70	89	3	2019	3
1	71	61	85	91	3	2019	3
2	79	16	87	77	2	2018	4
3	61	77	74	76	2	2020	2
4	78	71	67	90	3	2019	3

To decrease the skewness and convert distribution into normal distribution (Normalization by decimal scaling)

Data Skewness: It is asymmetry in a statistical distribution, in which the curve appears distorted or skewed either to the left or to the right. Skewness can be quantified to define the extent to which a distribution differs from a normal distribution.

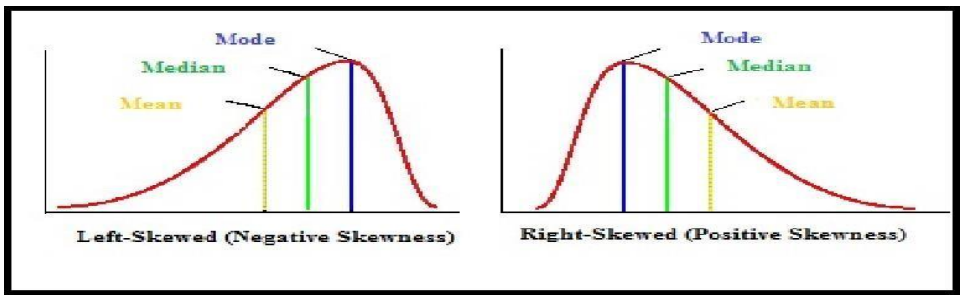
Normal Distribution: In a normal distribution, the graph appears as a classical, symmetrical "bell-shaped curve."The mean, or average, and the mode, or maximum point on the curve, are equal.



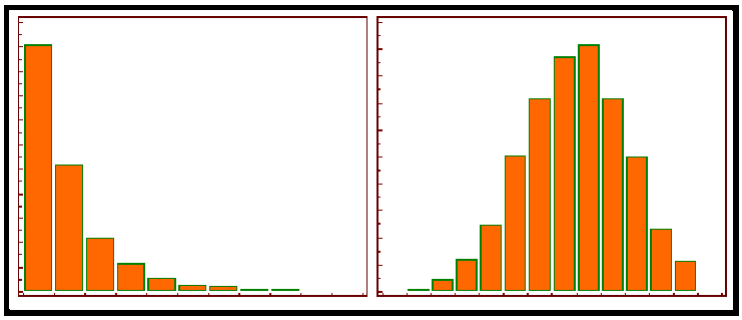
Positively Skewed Distribution

A positively skewed distribution means that the extreme data results are larger. This skews the data in that it brings the mean (average) up. The mean will be larger than the median in a Positively skewed distribution.

A negatively skewed distribution means the opposite: that the extreme data results are smaller. This means that the mean is brought down, and the median is larger than the mean in a negatively skewed distribution



Reducing skewness A data transformation may be used to reduce skewness. A distribution that is symmetric or nearly so is often easier to handle and interpret than a skewed distribution. The logarithm, x to log base 10 of x, or x to log base eofx (lnx), or x to log base 2of x, is a strong transformation with amajor effect on distribution shape. It is commonly used for reducing right skewness and is often appropriate for measured variables. It can not be applied to zero or negative values.



Algorithm:

Step1: Detecting outliers usingZ-ScorefortheMath_score variable and remove the outliers.

Step 2: Observe the histogram for math_score variable.import matplotlib.pyplot as plt new_df['mathscore'].plot(kind='hist')

Step 3: Convert the variables to logarithm at the scale 10.

df['log_math'] = np.log10(df['math score'])

Step 4: Observe the histogram for math_score variable. df['log_math'].plot(kind = 'hist') It is observed that skewness is reduced at some level.

Conclusion: In this way we have explored the functions of the python library for Data Identifying and handling the outliers. Data Transformations Techniques are explored with the Purpose of creating the new variable and reducing the skewness from datasets.

Viva Questions:

1. Explain the methods to detect the outlier.
2. Explain data transformation methods
3. Write the algorithm to display the statistics of Null values present in the dataset.
4. Write an algorithm to replace the outlier value with the mean of the variable.

.

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 3

Title of the Assignment: Descriptive Statistics - Measures of Central Tendency and variability

Perform the following operations using Python on any open source dataset (e.g., data.csv) Import all the required Python Libraries.

1. Provide summary statistics (mean, median, min, max, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorial) variable. For example, if your categorical variable is age group and quantitative variable is income, then provide summary statistics of income grouped by the age group. Create a list that contains a numeric value for each response to the categorical variable.

2. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of 'Iris-setosa', 'Iris-versicolor', and 'iris-versicolor' of iris.csv dataset.

Objective of the Assignment: Able to perform the Descriptive statistic operation using Python on any open source dataset.

Prerequisite:

1. Basic of Python Programming
2. Concept of Data Preprocessing, Data Formatting , Data Normalization and Data Cleaning

Outcome: Implement data representation using statistical methods

CO Relevance: CO2

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

1. **Introduction to Descriptive Statistics**
2. **Python Libraries for Data Science**
3. **Description of Dataset**
4. **Panda Dataframe functions for load the dataset**
5. **Panda functions for Data Preprocessing**
6. **Panda functions for Data Formatting and Normalization**
7. **Panda Functions for handling categorical variables**

Statistics

It means collection, organization, analysis and interpretation of data. Statistics are mainly used to give numerical conclusions. For example, if anyone asks you how many people are watching YouTube, in this case, we can't say more; many people are watching you tube, we have to answer in numerical terms that give more meaning to you. We can say like during weekdays 6 pm-8 pm more people are watching youtube applications and during weekend 8 pm-11 pm. If you want to answer active users, we can say there are two billion+ monthly active users, in the same way; the users spend a daily average of 18 minutes. This is the numerical way to conclude the questions, and statistics is the medium used to make such inference.

Statistics include;

Design of experiments: Used to understand Characteristics of the dataset

Sampling: Used to understand the samples

Descriptive statistics: Summarization of data

Inferential Statistics: Hypothesis way of concluding data Probability Theory: Likelihood estimation

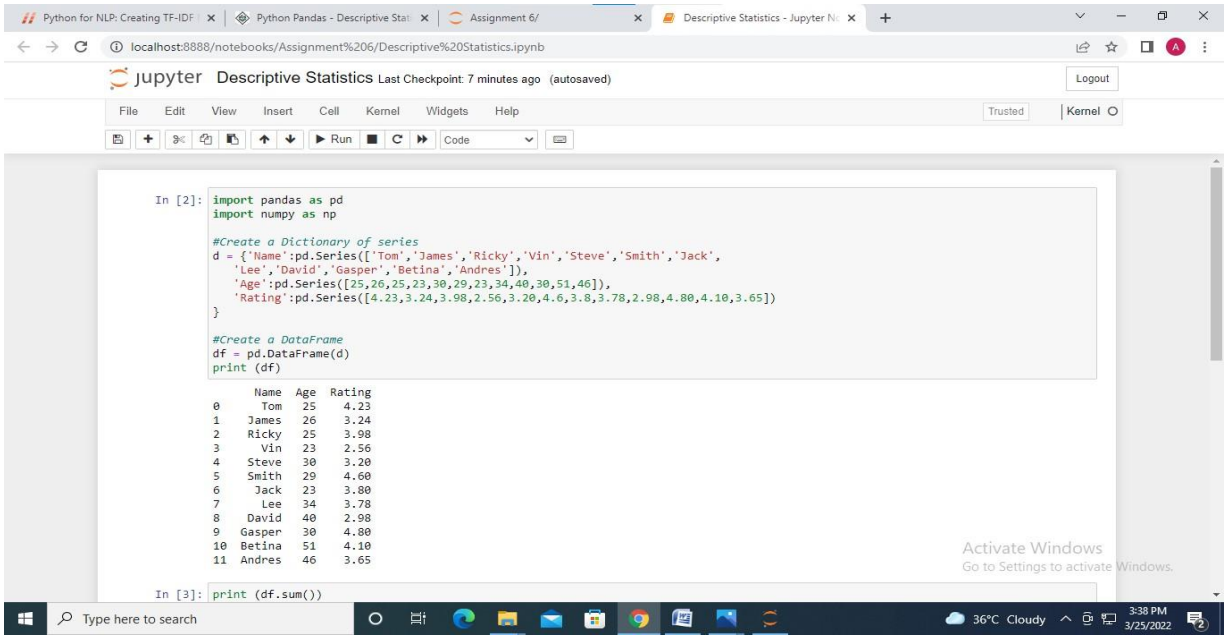
Descriptive statistics:
In statistical analysis, there are three main fundamental concepts associated with describing the data: location or Central tendency, Dissemination or spread, and Shape or distribution. A raw dataset is difficult to describe; descriptive statistics describe the dataset in a way simpler manner through;

- The measure of central tendency (Mean, Median, Mode)
- Measure of spread (Range, Quartile, Percentiles, absolute deviation, variance and standard deviation)
- Measure of symmetry (Skewness)
- Measure of Peakedness (Kurtosis)

Let's see the above one by one by leveraging Python

DataFrame

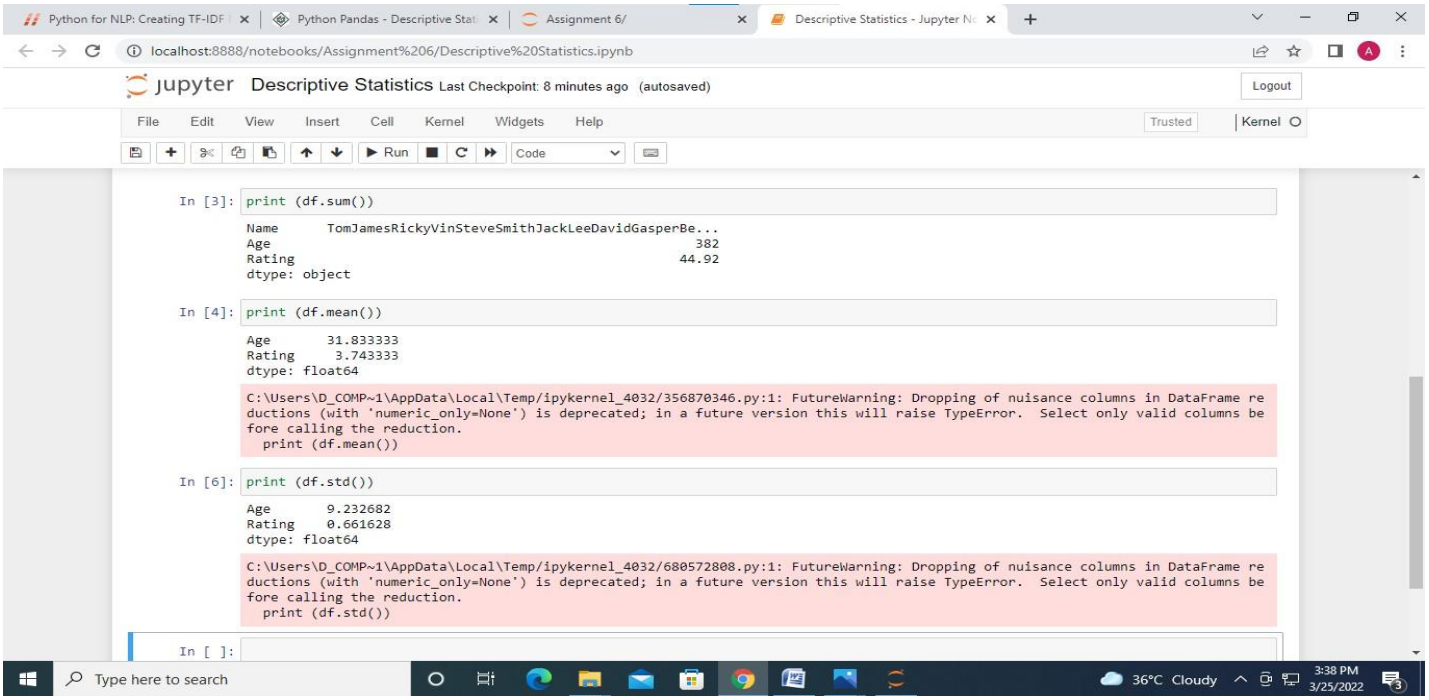
Let us create a DataFrame and use this object throughout this chapter for all the operations.



sum():Returns the sum of the values for the requested axis. By default, axis is index (axis=0).

mean ():Returns the average value

std():Returns the Bessel standard deviation of the numerical columns.



Functions & Description

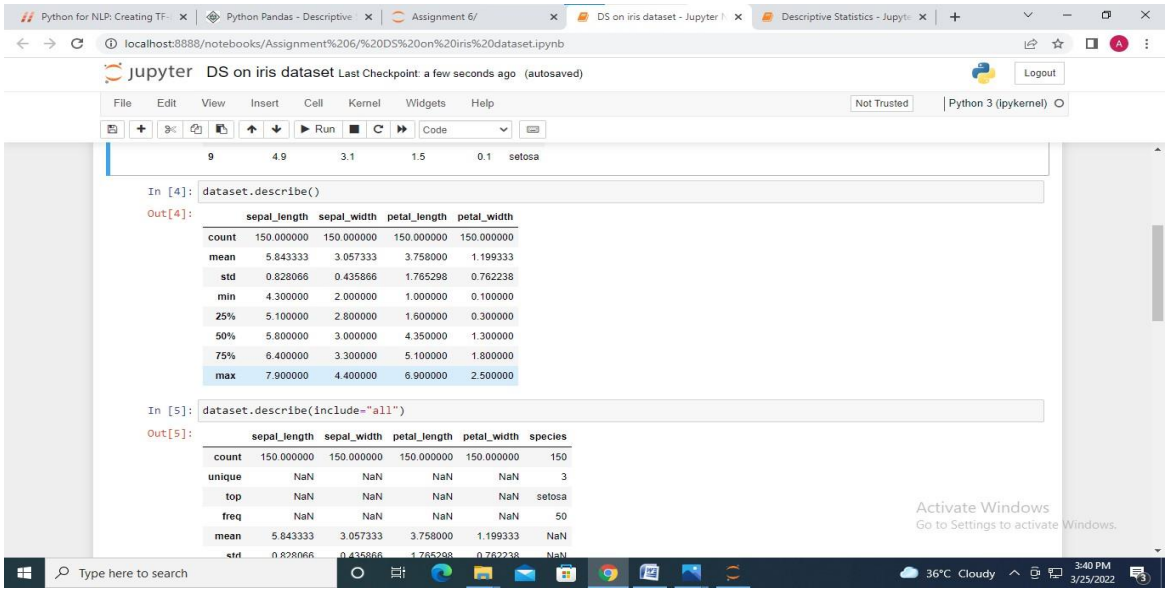
Let us now understand the functions under Descriptive Statistics in Python Pandas. The following table list down the important functions −

Sr.No.	Function	Description
1	count()	Number of non-null observations
2	sum()	Sum of values
3	mean()	Mean of Values
4	median()	Median of Values
5	mode()	Mode of values
6	std()	Standard Deviation of the Values
7	min()	Minimum Value
8	max()	Maximum Value

9	abs()	Absolute Value
10	prod()	Product of Values
11	cumsum()	Cumulative Sum
12	cumprod()	Cumulative Product

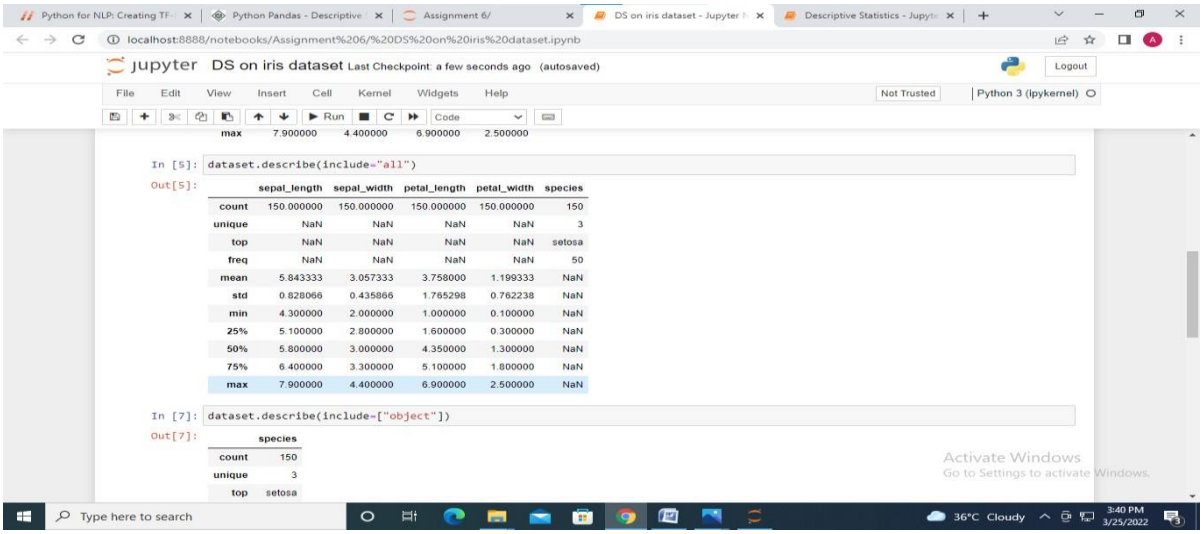
Describe () Method:

This method returns many useful descriptive statistics with a mix of measures of central tendency and measures of variability. This includes the number of non-missing observations; the mean; standard deviation; minimum value; 25th, 50th (a.k.a. the median), and 75th percentile; as well as the maximum value. It's missing some useful information that is typically desired regarding the mean, this is the standard error and the 95% confidence interval.



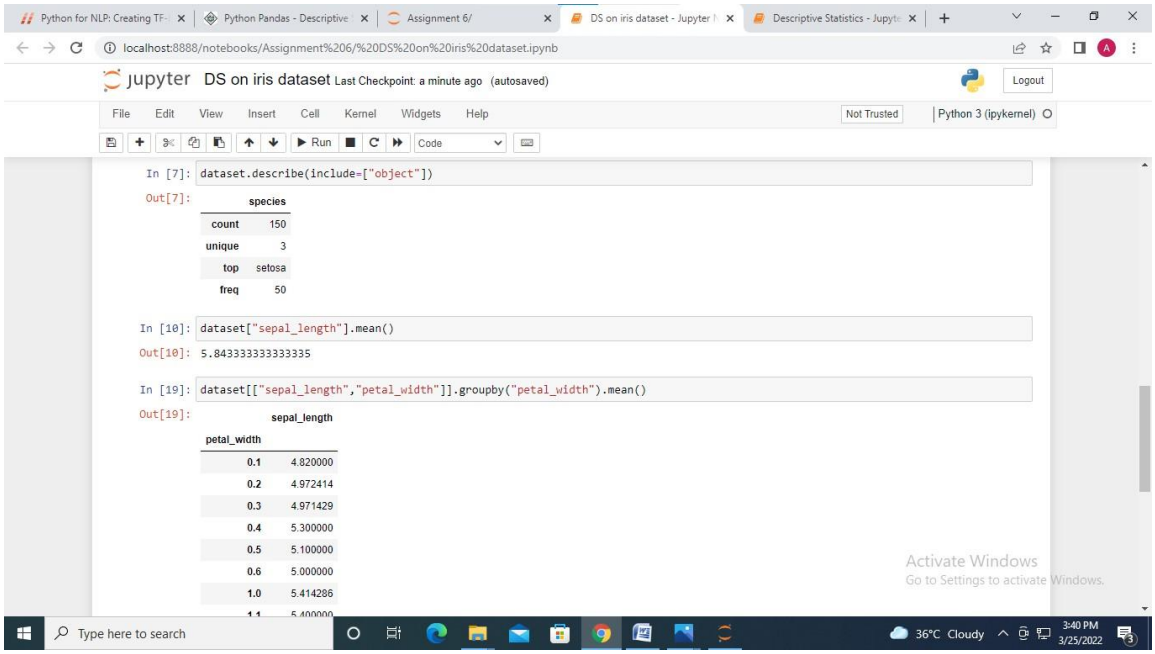
Describe (include="all")

describe() function with include=all gives the summary statistics of all the columns



Groupby() method:

To get the statistics of categorical data this method is used.



Conclusion:

Hence, we studied Python Descriptive Statistics, in which we learned Central Tendency & Dispersion used in Python Statistics Module. In addition, we used the statistics and pandas modules for this.

Viva Question:

1 Why Data Statictics important in Data science?

2 What are different methods include in Data Statistics?

3 What is Mean and Median?

4 What is left skweness?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 4

Title of the Assignment: Data Analytics

Create a linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (http://www.kaggle.com/c/boston-housing). The Boston Housing Dataset contains information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables in this dataset.

Objective of the Assignment: Predict the value of prices of the house using the given features.

Prerequisite:

Basic of Python Programming
Concept of Data Preprocessing, Data Formatting , Data Normalization and Data Cleaning.
Regression algorithms

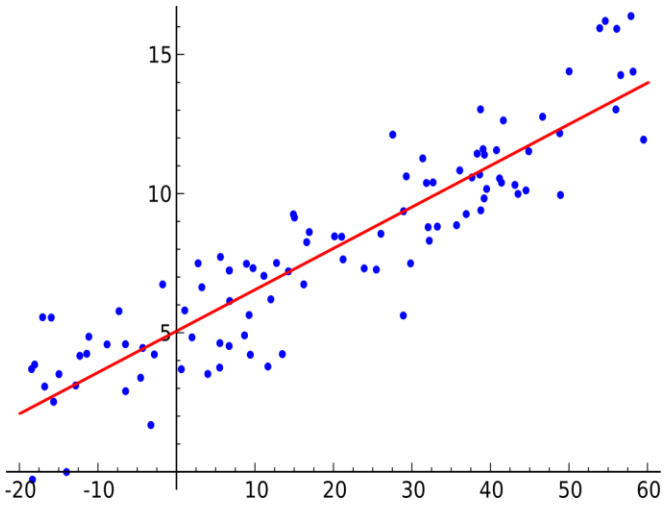
Outcome: To predict home prices for Boston Housing Dataset using Linear Regression Algorithm

Contents for Theory:

1. **Introduction to Regression**
2. **Introduction to Linear Regression**
3. **Python Libraries for Data Science**
4. **Description of Dataset**
5. **Panda Dataframe functions for load the dataset**
6. **Panda functions for Data Preprocessing**
7. **Linear regression implementation.**

Regression

Regression is a method of modeling a target value based on independent predictors. This method is mostly used for forecasting and finding out cause and effect relationship between variables. Regression techniques mostly differ based on the number of independent variables and the type of relationship between the independent and dependent variables



Linear Regression

Simple linear regression is a type of regression analysis where the number of independent variables is one and there is a linear relationship between the independent(x) and dependent(y) variable. The red line in the above graph is referred to as the best fit straight line. Based on the given data points, we try to plot a line that models the points the best. The line can be modelled based on the linear equation shown below.

```
y = a_0 + a_1 * x  ## Linear Equation
```

The motive of the linear regression algorithm is to find the best values for a_0 and a_1. Before moving on to the algorithm, let's have a look at two important concepts you must know to better understand linear regression.

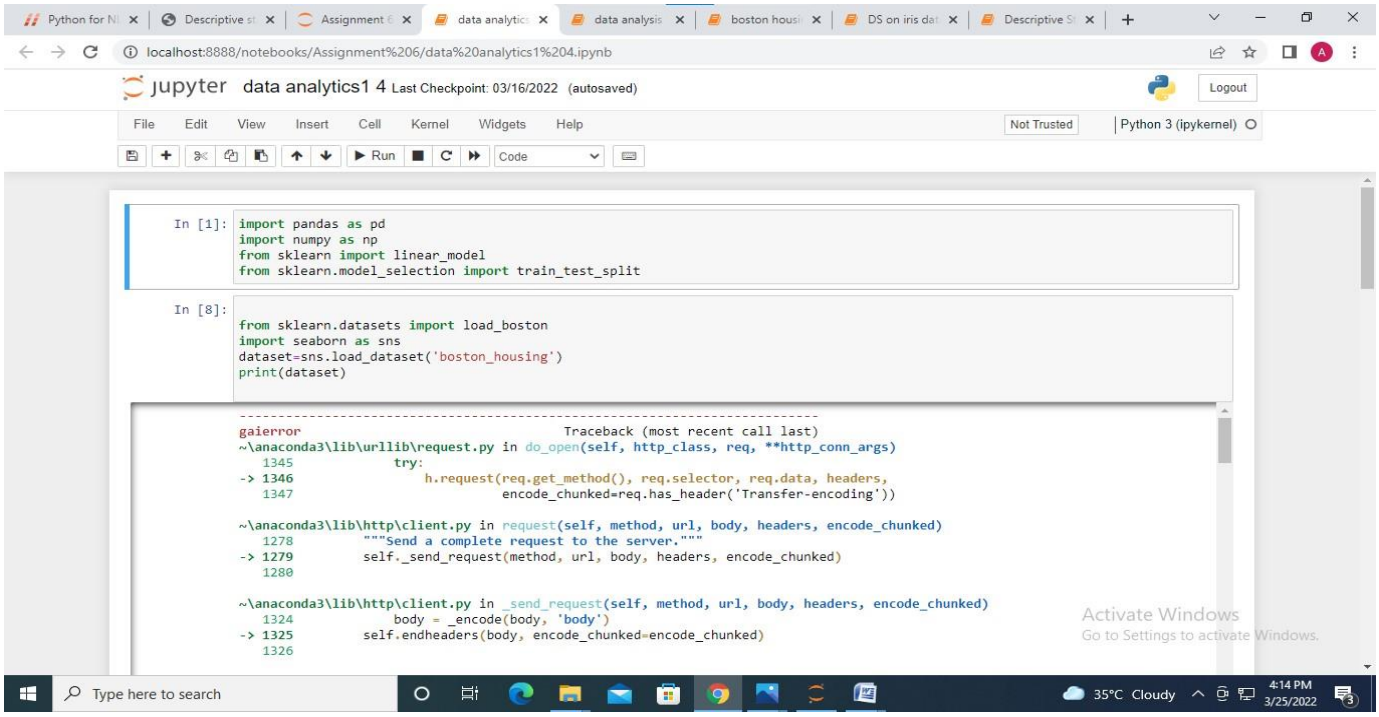
Let's get to the code. We have two choices, we can either use the scikit learn library to import the linear regression model and use it directly or we can write our own regression model based on the equations above. Instead of choosing one among the two, let's do both :)
There are many datasets available online for linear regression. Let's visualize the training and testing data.

Description of Dataset

Boston Housing dataset which contains information about different houses in Boston. This data was originally a part of UCI Machine Learning Repository and has been removed now. We can also access this data from the

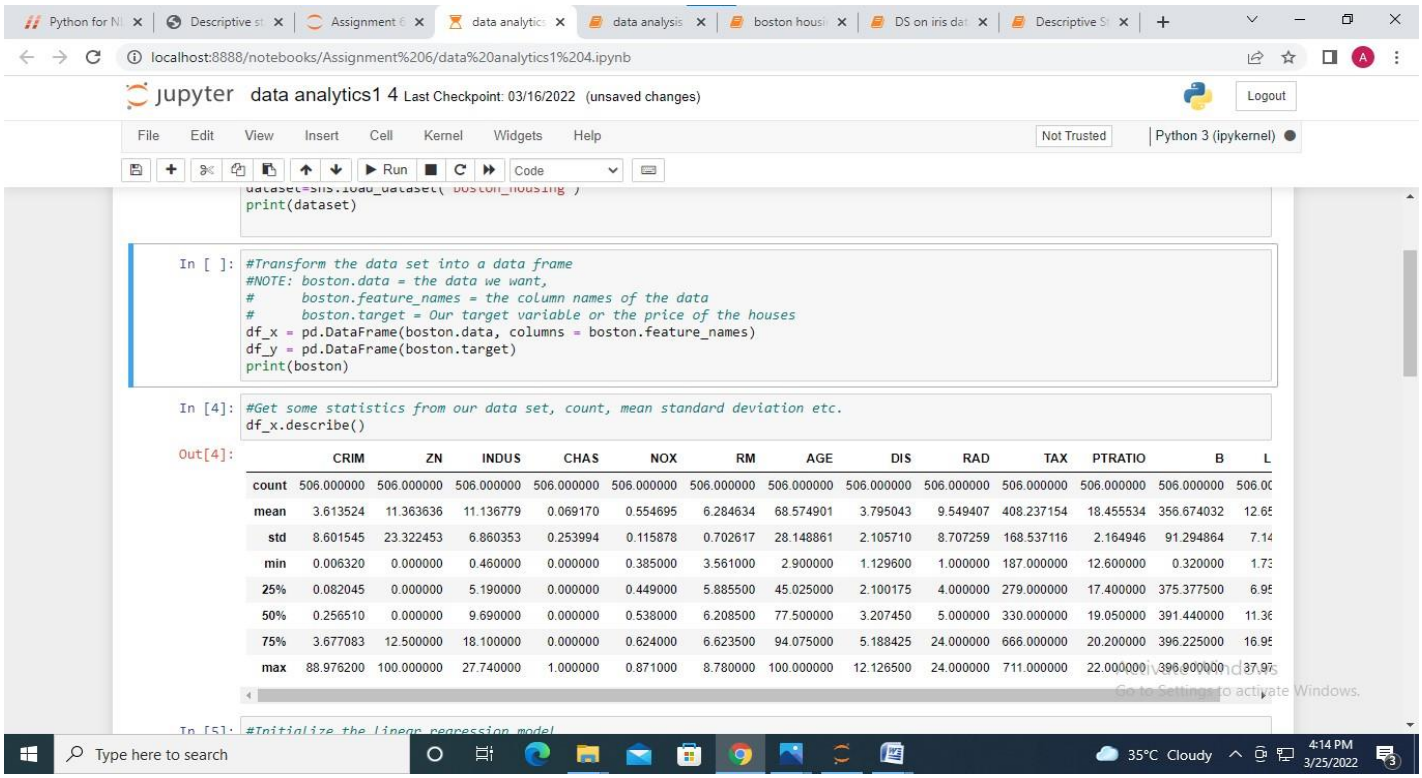
scikit-learn library. There are 506 samples and 14 feature variables in this dataset. The objective is to predict the value of prices of the house using the given features.

First, we will import the required libraries



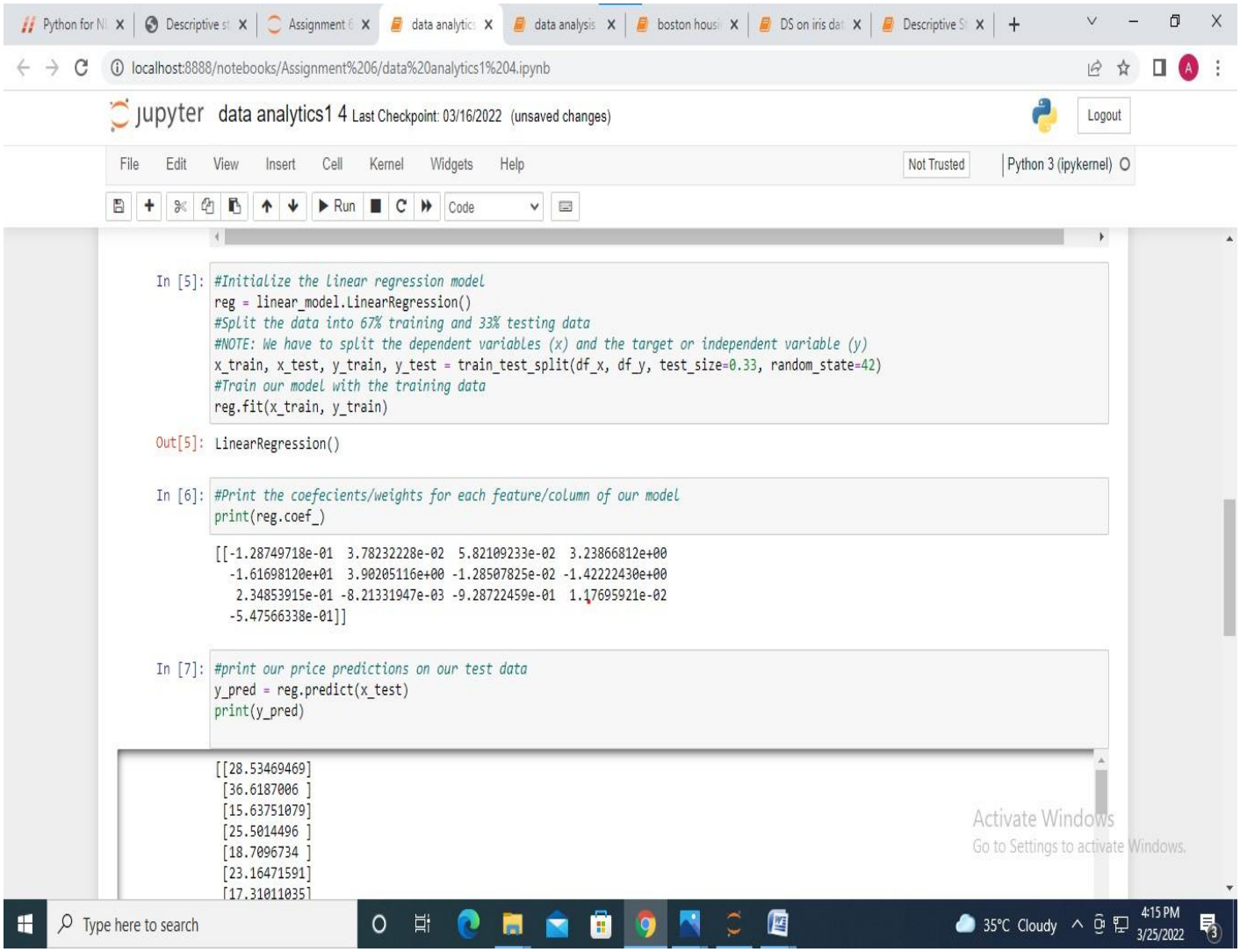
Next, we will load the housing data from the scikit-learn library and understand it.

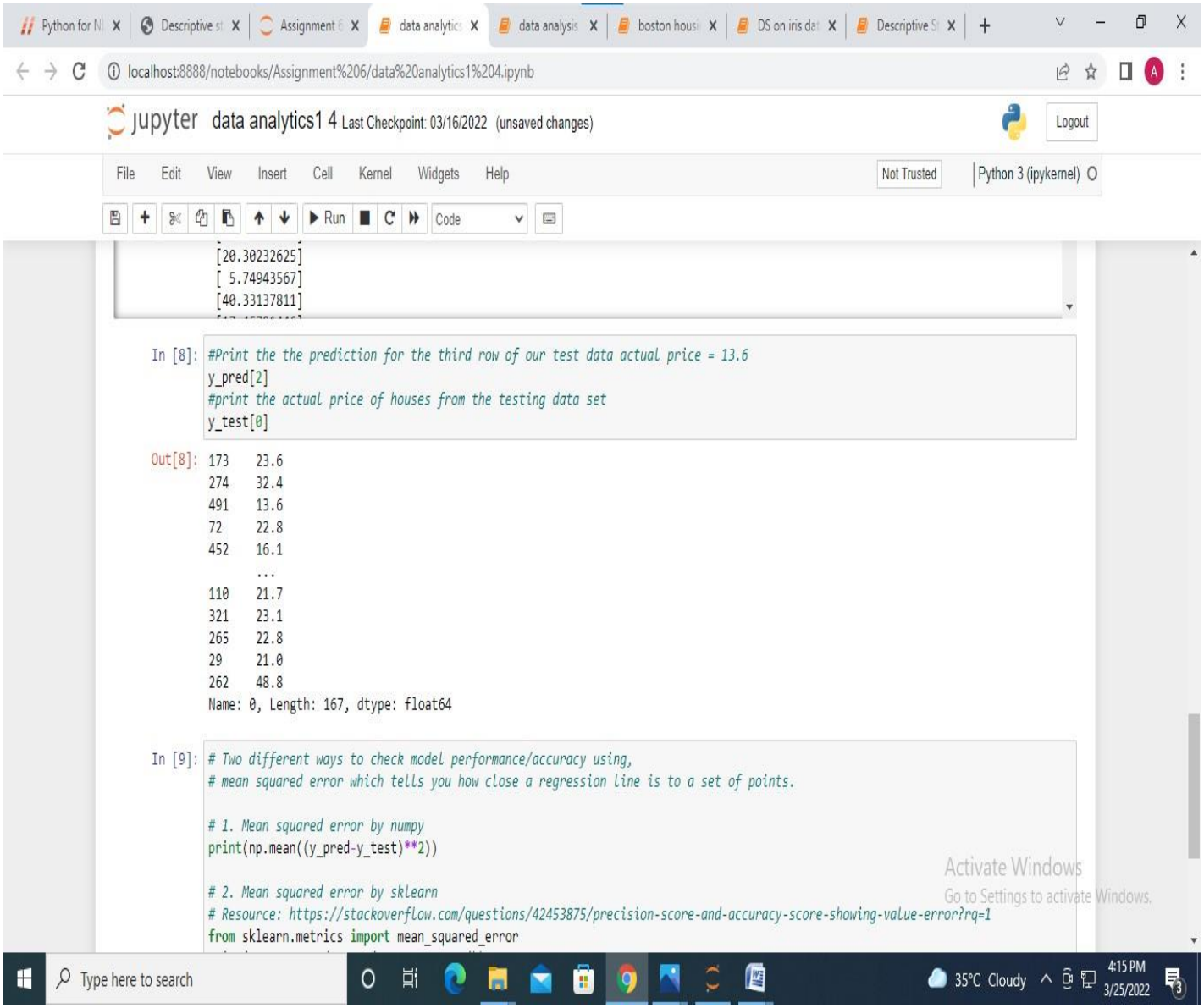
We print the value of the boston_dataset to understand what it contains. print(boston_dataset.keys()) gives dict_keys(['data', 'target', 'feature_names', 'DESCR'])

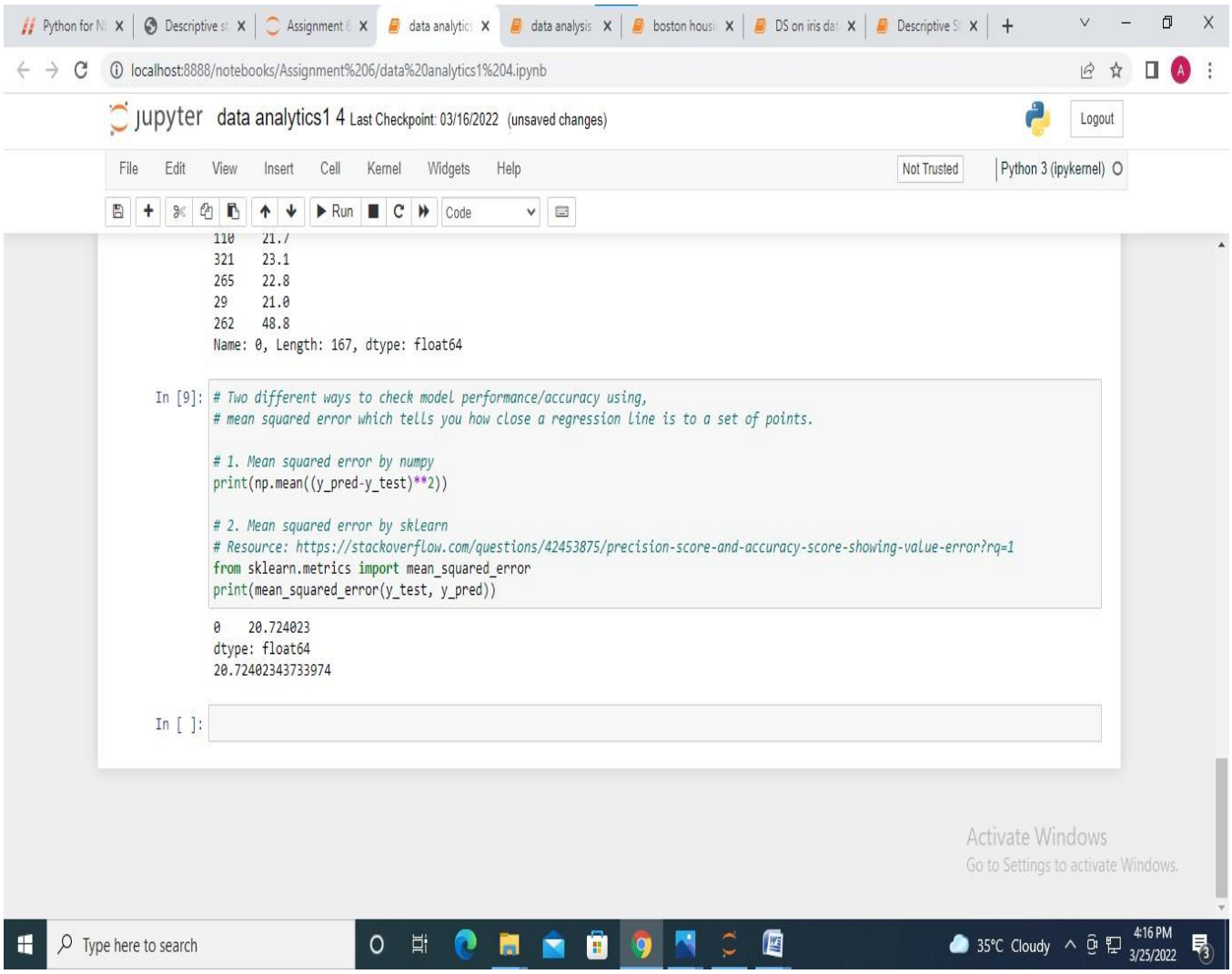


- data: contains the information for various houses
- target: prices of the house
- feature_names: names of the features
- DESCR: describes the dataset

The prices of the house indicated by the variable MEDV is our **target variable** and the remaining are the **feature variables** based on which we will predict the value of a house.


localhost:8888/notebooks/Assignment%206/data%20analytics1%204.ipynb
jupyter data analytics1 4 Last Checkpoint: 03/16/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [5]:
#Initialize the linear regression model
reg = linear_model.LinearRegression()
#Split the data into 67% training and 33% testing data
#NOTE: We have to split the dependent variables (x) and the target or independent variable (y)
x_train, x_test, y_train, y_test = train_test_split(df_x, df_y, test_size=0.33, random_state=42)
#Train our model with the training data
reg.fit(x_train, y_train)
Out[5]:
LinearRegression()
In [6]:
#Print the coefecients/weights for each feature/column of our model
print(reg.coef_)
[[-1.28749718e-01 3.78232228e-02 5.82109233e-02 3.23866812e+00
-1.61698120e+01 3.90205116e+00 -1.28507825e-02 -1.42222430e+00
2.34853915e-01 -8.21331947e-03 -9.28722459e-01 1.17695921e-02
-5.47566338e-01]]
In [7]:
#print our price predictions on our test data
y_pred = reg.predict(x_test)
print(y_pred)
[[28.53469469]
[36.6187006]
[15.63751079]
[25.5014496]
[18.7096734]
[23.16471591]
[17.31011035]
Activate Windows
Go to Settings to activate Windows.
Type here to search
35°C Cloudy
4:15 PM
3/25/2022


Python for N
Descriptive st
Assignment 6
data analytics
data analysis
boston housi
DS on iris dat
Descriptive St
localhost:8888/notebooks/Assignment%206/data%20analytics1%204.ipynb
jupyter data analytics1 4 Last Checkpoint: 03/16/2022 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
[20.30232625]
[5.74943567]
[40.33137811]
In [8]:
#Print the the prediction for the third row of our test data actual price = 13.6
y_pred[2]
#print the actual price of houses from the testing data set
y_test[0]
Out[8]:
173 23.6
274 32.4
491 13.6
72 22.8
452 16.1
...
110 21.7
321 23.1
265 22.8
29 21.0
262 48.8
Name: 0, Length: 167, dtype: float64
In [9]:
# Two different ways to check model performance/accuracy using,
# mean squared error which tells you how close a regression line is to a set of points.
# 1. Mean squared error by numpy
print(np.mean((y_pred-y_test)**2))
# 2. Mean squared error by sklearn
# Resource: https://stackoverflow.com/questions/42453875/precision-score-and-accuracy-score-showing-value-error?rq=1
from sklearn.metrics import mean_squared_error
Activate Windows
Go to Settings to activate Windows.
Type here to search
35°C Cloudy
4:15 PM
3/25/2022



Conclusion:

Studied linear Regression Model using Python/R and applied it Boston Housing Dataset (http://www.kaggle.com/c/boston-housing).

Viva Question:

What are the different types of Logistic Regression?

How can we express the probability of a Logistic Regression model as conditional probability?

Why is Logistic Regression termed as Regression and not classification?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 4

Title of the Assignment: Data Analytics II

1. Implement logistic regression using Python/R to perform classification algorithm on Social_Network_Ads.csv dataset
2. Compute the Confusion matrix to find TP ,FP , TN, FN, ACCURACY , Error Factor, Precision Recall on the given dataset.

Objective of the Assignment: Able to perform the data wrangling operation using Python on any open source dataset

---

Prerequisite:

Basic of Python Programming

Concept of Data Preprocessing, Data Formatting , Data Normalization and Data Cleaning.

Outcome: Implement and evaluate logistic regression algorithm on Social_netwrok_Ads.csv dataset

CO Relevance: CO3

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

1. **Introduction to Logistic Regression**
2. **Description of dataset**
3. **Implementation of logistic Regression**
4. **Introduction to TP, FP ,TN, FN, Accuracy, Error rate, Precision, Recall, Confusion Matrix**

What Is Classification?

Supervised machine learning algorithms define models that capture relationships among data. **Classification** is an area of supervised machine learning that tries to predict which class or category

some entity belongs to, based on its features.

For example, you might analyze the employees of some company and try to establish a dependence on the **features** or **variables**, such as the level of education, number of years in a current position, age, salary, odds for being promoted, and so on. The set of data related to a single employee is one **observation**. The features or variables can take one of two forms:

1. **Independent variables**, also called inputs or predictors, don't depend on other features of interest (or at least you assume so for the purpose of the analysis).
2. **Dependent variables**, also called outputs or responses, depend on the independent variables.

The nature of the dependent variables differentiates regression and classification problems. **Regression** problems have continuous and usually unbounded outputs. An example is when you're estimating the salary as a function of experience and education level. On the other hand, **classification** problems have discrete and finite outputs called **classes** or **categories**. For example, predicting if an employee is going to be promoted or not (true or false) is a classification problem.

There are two main types of classification problems:

1. **Binary** or **binomial classification:** exactly two classes to choose between (usually 0 and 1, true and false, or positive and negative)
2. **Multiclass** or **multinomial classification:** three or more classes of the outputs to choose from

If there's only one input variable, then it's usually denoted with x. For more than one input, you'll commonly see the vector notation $\mathbf{x} = (x_1, \ldots, x_r)$, where r is the number of the predictors (or independent features). The output variable is often denoted with y and takes the values 0 or 1.

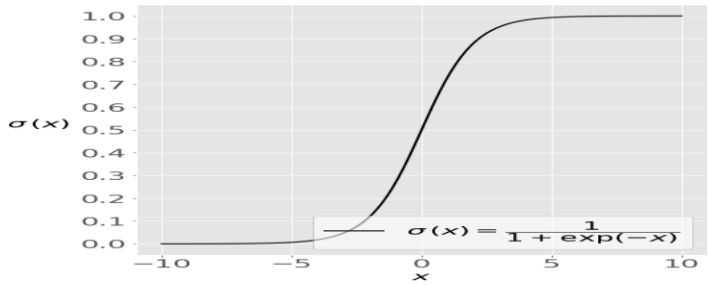
Logistic Regression Overview

Logistic regression is a fundamental classification technique. It belongs to the group of **linear classifiers** and is somewhat similar to polynomial and **linear regression**. Logistic regression is fast and relatively uncomplicated, and it's convenient for you to interpret the results. Although it's essentially a method for binary classification, it can also be applied to multiclass problems.

Math Prerequisites

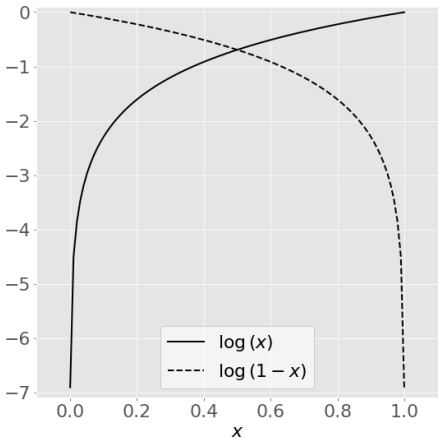
You'll need an understanding of the sigmoid function and the natural logarithm function to understand what logistic regression is and how it works.

This image shows the sigmoid function (or S-shaped curve) of some variable x:



The sigmoid function has values very close to either 0 or 1 across most of its domain. This fact makes it suitable for application in classification methods.

This image depicts the natural logarithm $\log(x)$ of some variable x, for values of x between 0 and 1:



As x approaches zero, the natural logarithm of x drops towards negative infinity. When $x = 1$, $\log(x)$ is 0. The opposite is true for $\log(1 - x)$.

Note that you'll often find the natural logarithm denoted with **ln** instead of **log**. In Python, math.log(x) and numpy.log(x) represent the natural logarithm of x, so you'll follow this notation in this tutorial.

Methodology

Logistic regression is a linear classifier, so you'll use a linear function $f(\mathbf{x}) = b_0 + b_1x_1 + \cdots + b_rx_r$, also called the **logit**. The variables b_0, b_1, …, b_r are the **estimators** of the regression coefficients, which are also called the **predicted weights** or just **coefficients**.

The logistic regression function $p(\mathbf{x})$ is the sigmoid function of $f(\mathbf{x})$: $p(\mathbf{x}) = 1 / (1 + \exp(-f(\mathbf{x}))$. As such, it's often close to either 0 or 1. The function $p(\mathbf{x})$ is often interpreted as the predicted probability that the output for a given $\mathbf{x}$ is equal to 1. Therefore, $1 - p(\mathbf{x})$ is the probability that the output is 0.

Logistic regression determines the best predicted weights b_0, b_1, …, b_r such that the function $p(\mathbf{x})$ is as close as possible to all actual responses y_i, $i = 1, \ldots, n$, where n is the number of observations. The process of calculating the best weights using available observations is called **model training** or **fitting**.

To get the best weights, you usually maximize the **log-likelihood function (LLF)** for all observations $i = 1, \ldots, n$. This method is called the **maximum likelihood estimation** and is represented by the equation $\text{LLF} = \Sigma_i(y_i \log(p(\mathbf{x}_i)) + (1 - y_i) \log(1 - p(\mathbf{x}_i)))$.

When $y_i = 0$, the LLF for the corresponding observation is equal to $\log(1 - p(\mathbf{x}_i))$. If $p(\mathbf{x}_i)$ is close to $y_i = 0$, then $\log(1 - p(\mathbf{x}_i))$ is close to 0. This is the result you want. If $p(\mathbf{x}_i)$ is far from 0, then $\log(1 - p(\mathbf{x}_i))$ drops significantly. You don't want that result because your goal is to obtain the maximum LLF. Similarly, when $y_i = 1$, the LLF for that observation is $y_i \log(p(\mathbf{x}_i))$. If $p(\mathbf{x}_i)$ is close to $y_i = 1$, then $\log(p(\mathbf{x}_i))$ is close to 0. If $p(\mathbf{x}_i)$ is far from 1, then $\log(p(\mathbf{x}_i))$ is a large negative number.

There are several mathematical approaches that will calculate the best weights that correspond to the maximum LLF, but that's beyond the scope of this tutorial. For now, you can leave these details to the logistic regression Python libraries you'll learn to use here!

Once you determine the best weights that define the function $p(\mathbf{x})$, you can get the predicted outputs $p(\mathbf{x}_i)$ for any given input $\mathbf{x}_i$. For each observation $i = 1, \ldots, n$, the predicted output is 1 if $p(\mathbf{x}_i) > 0.5$ and 0 otherwise. The threshold doesn't have to be 0.5, but it usually is. You might define a lower or higher value if that's more convenient for your situation.

There's one more important relationship between $(\mathbf{x})$ and $f(\mathbf{x})$, which is that $\log(p(\mathbf{x}) / (1 - p(\mathbf{x}))) = f(\mathbf{x})$. This equality explains why $(\mathbf{x})$ is the logit. It implies that $(\mathbf{x}) = 0.5$ when $f(\mathbf{x}) = 0$ and that the predicted output is 1 if $f(\mathbf{x}) > 0$ and 0 otherwise.

Classification Performance

Binary classification has four possible types of results:

1. **True negatives:** correctly predicted negatives (zeros)
2. **True positives:** correctly predicted positives (ones)
3. **False negatives:** incorrectly predicted negatives (zeros)
4. **False positives:** incorrectly predicted positives (ones)

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

dataset = pd.read_csv('../input/Social_Network_Ads.csv')
dataset.head()
```

Out[1]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

In [2]:
```
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

print(X[:3, :])
print('-'*15)
print(y[:3])
```

```
[[   19 19000]
 [   35 20000]
 [   26 43000]]
---------------
[0 0 0]
```

In [2]:
```
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

print(X[:3, :])
print('-'*15)
print(y[:3])
```

```
[[   19 19000]
 [   35 20000]
 [   26 43000]]
---------------
[0 0 0]
```

In [3]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

print(X_train[:3])
print('-'*15)
print(y_train[:3])
print('-'*15)
print(X_test[:3])
print('-'*15)
print(y_test[:3])
```

```
[[    44  39000]
 [    32 120000]
 [    38  50000]]
---------------
[0 1 0]
---------------
[[   30 87000]
 [   38 50000]
 [   35 75000]]
---------------
[0 0 0]
```

In [4]:
```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

```
/opt/conda/lib/python3.6/site-packages/sklearn/utils/validation.py:595: DataConversionWarning: Data with input dtype int64 was con
verted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
/opt/conda/lib/python3.6/site-packages/sklearn/utils/validation.py:595: DataConversionWarning: Data with input dtype int64 was con
verted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
/opt/conda/lib/python3.6/site-packages/sklearn/utils/validation.py:595: DataConversionWarning: Data with input dtype int64 was con
verted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
```

In [5]:

In [5]:
```
print(X_train[:3])
print('-'*15)
print(X_test[:3])
```

```
[[ 0.58164944 -0.88670699]
 [-0.60673761  1.46173768]
 [-0.01254409 -0.5677824 ]]
---------------
[[-0.80480212  0.50496393]
 [-0.01254409 -0.5677824 ]
 [-0.30964085  0.1570462 ]]
```

In [6]:
```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0, solver='lbfgs' )
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)

print(X_test[:10])
print('-'*15)
print(y_pred[:10])
```

```
[[-0.80480212  0.50496393]
 [-0.01254409 -0.5677824 ]
 [-0.30964085  0.1570462 ]
 [-0.80480212  0.27301877]
 [-0.30964085 -0.5677824 ]
 [-1.10189888 -1.43757673]
 [-0.70576986 -1.58254245]
 [-0.21060859  2.15757314]
 [-1.99318916 -0.04590581]
 [ 0.8787462  -0.77073441]]
---------------
[0 0 0 0 0 0 0 1 0 1]
```

In [7]:
```
print(y_pred[:20])
print(y_test[:20])
```

```
[0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0]
```

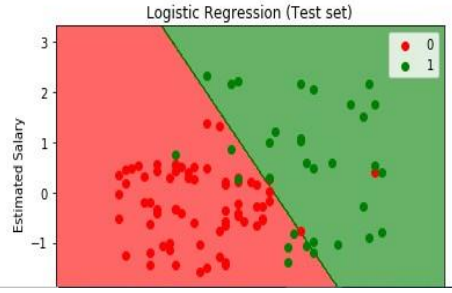
In [8]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[65  3]
 [ 8 24]]
```

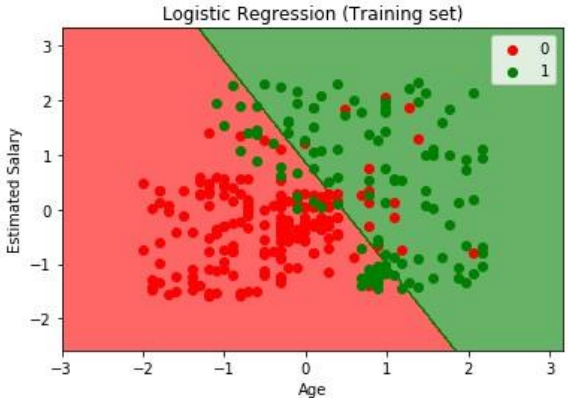
In [9]:

```
# Visualizing the Training set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.6, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```


```
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.6, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('Logistic Regression (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```



```
plt.title('Logistic Regression (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```



Conclusion: Studied Linear Regrasion and applied it on given dataset successfully.

Viva Questions:

- What is a Linear Regression?
- Can you list out the critical assumptions of linear regression?
- What is Heteroscedasticity?
- What is the primary difference between R square and adjusted R square?
- Can you list out the formulas to find RMSE and MSE?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 6

Title of the Assignment: Data Analytics II

Implement simple Naïve Bayes classification algorithm using Python/R to perform classification algorithm on iris.csv dataset.

Compute the Confusion matrix to find TP, FP, TN, FN, ACCURACY, Error Factor, Precision Recall on the given dataset.

Objective of the Assignment: Able to understand the data Classification mechanism using Python on any open source dataset.

Prerequisite:

Basic of Python Programming
Knowledge of Naïve Bayes Algorithm.
Concept of Data Preprocessing, Data Formatting , Data Normalization and Data Cleaning

Outcome: **Implement and evaluate Naïve Bayes algorithm and also compute matrix to find TP,FP,TN,accuracy** , Error Factor, Precision Recall on the given dataset.

CO Relevance: CO3

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

1. **Introduction to Classifier**
2. **Introduction to Naïve Bayes Algorithm.**
3. **Implementation of Naïve Bayes Algorithm.**
4. **Introduction to TP, FP ,TN, FN, Accuracy, Error rate, Precision, Recall, Confusion Matrix**

Classifier:

A classifier is a machine learning model that is used to discriminate different objects based on certain features

Naïve Bayes Algorithm.

Principle of Naive Bayes Classifier:

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification task. The crux of the classifier is based on the Bayes theorem.

Bayes Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.

Example:

Let us take an example to get some better intuition. Consider the problem of playing golf. The dataset is represented as below.

	OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY GOLF
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

We classify whether the day is suitable for playing golf, given the features of the day. The columns represent these features and the rows represent individual entries. If we take the first row of the dataset,

we can observe that is not suitable for playing golf if the outlook is rainy, temperature is hot, humidity is high and it is not windy. We make two assumptions here, one as stated above we consider that these predictors are independent. That is, if the temperature is hot, it does not necessarily mean that the humidity is high. Another assumption made here is that all the predictors have an equal effect on the outcome. That is, the day being windy does not have more importance in deciding to play golf or not.

According to this example, Bayes theorem can be rewritten as:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

The variable **y** is the class variable(play golf), which represents if it is suitable to play golf or not given the conditions. Variable **X** represent the parameters/features.

X is given as,

$$X = (x_1, x_2, x_3, ..., x_n)$$

Here x_1,x_2….x_n represent the features, i.e they can be mapped to outlook, temperature, humidity and windy. By substituting for **X** and expanding using the chain rule we get,

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

Now, we can obtain the values for each by looking at the dataset and substitute them into the equation. For all entries in the dataset, the denominator does not change, it remain static. Therefore, the denominator can be removed and a proportionality can be introduced.

$$P(y|x_1, ..., x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

In our case, the class variable(**y**) has only two outcomes, yes or no. There could be cases where the classification could be multivariate. Therefore, we need to find the class **y** with maximum probability.

$$y = argmax_y P(y) \prod_{i=1}^{n} P(x_i|y)$$

Using the above function, we can obtain the class, given the predictors.

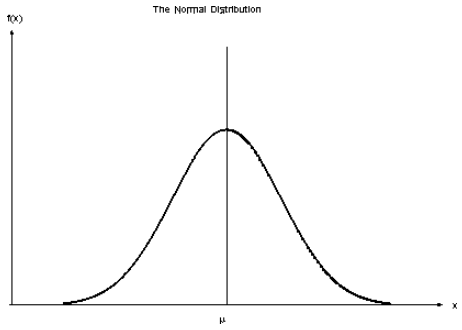
Types of Naive Bayes Classifier:

Bernoulli Naive Bayes:

This is similar to the multinomial naive bayes but the predictors are boolean variables. The parameters that we use to predict the class variable take up only values yes or no, for example if a word occurs in the text or not.

Gaussian Naive Bayes:

When the predictors take up a continuous value and are not discrete, we assume that these values are sampled from a gaussian distribution.



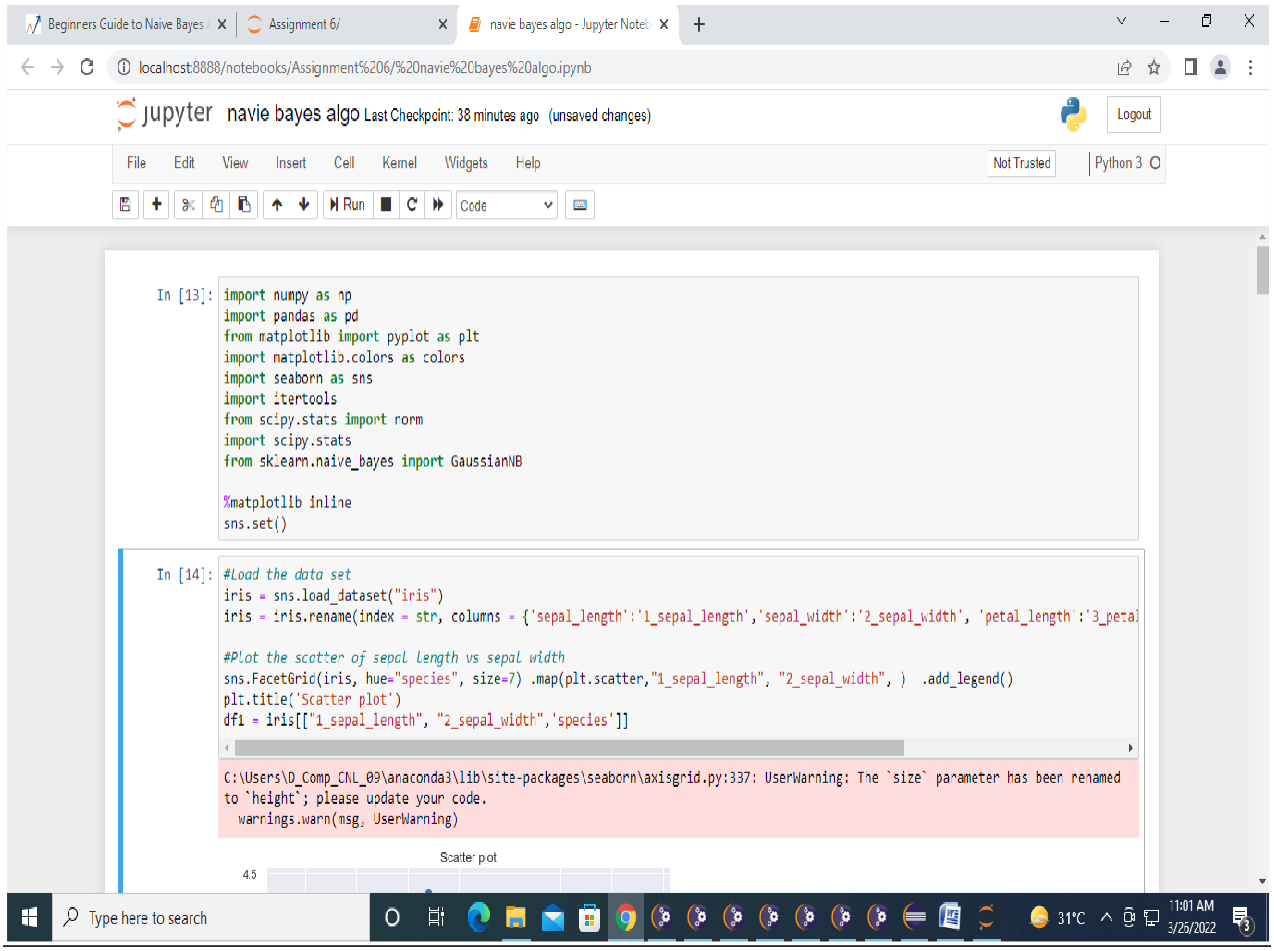
Gaussian Distribution(Normal Distribution)

Since the way the values are present in the dataset changes, the formula for conditional probability changes to,

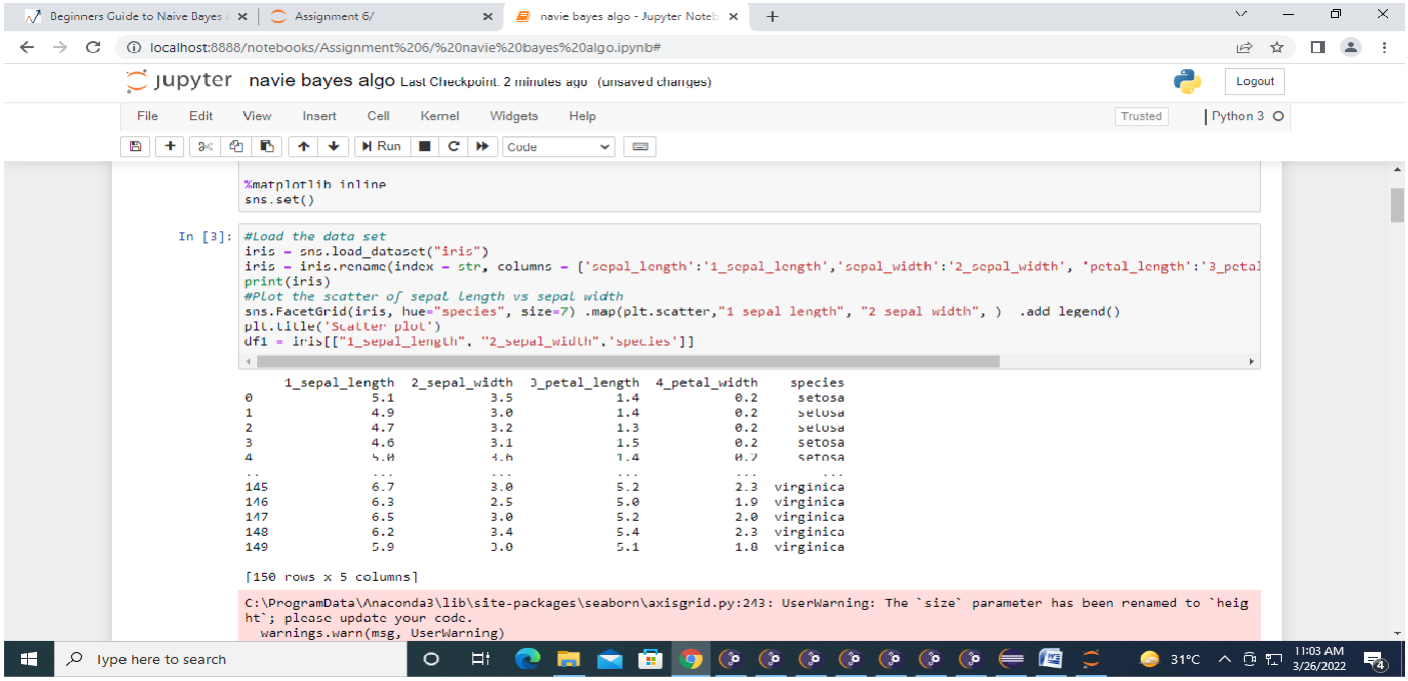
$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)$$

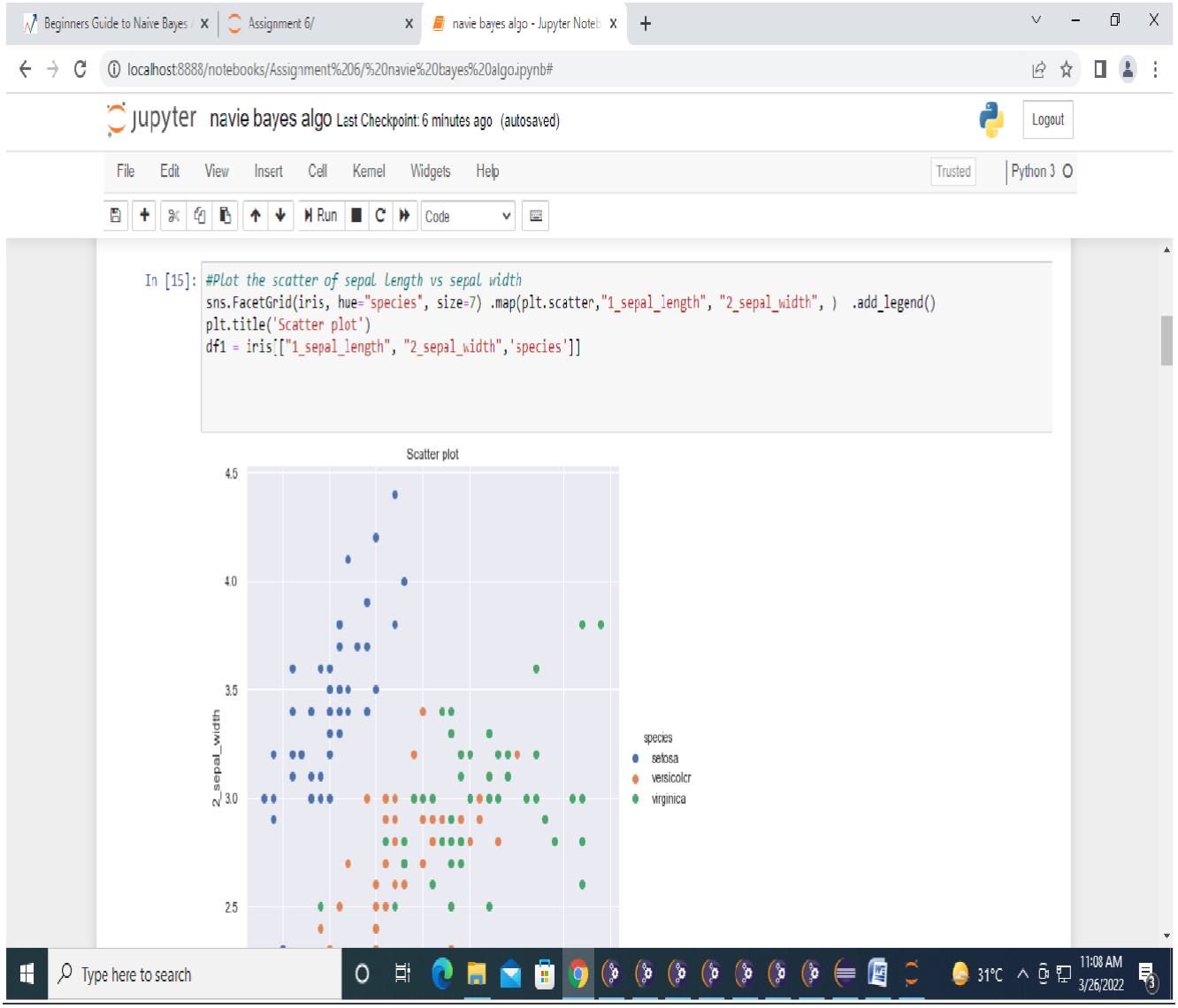
Implementation of Naïve Bayes Algorithm.

Let's start the programming by importing essential libraries required



Importing of dataset





Train test Split

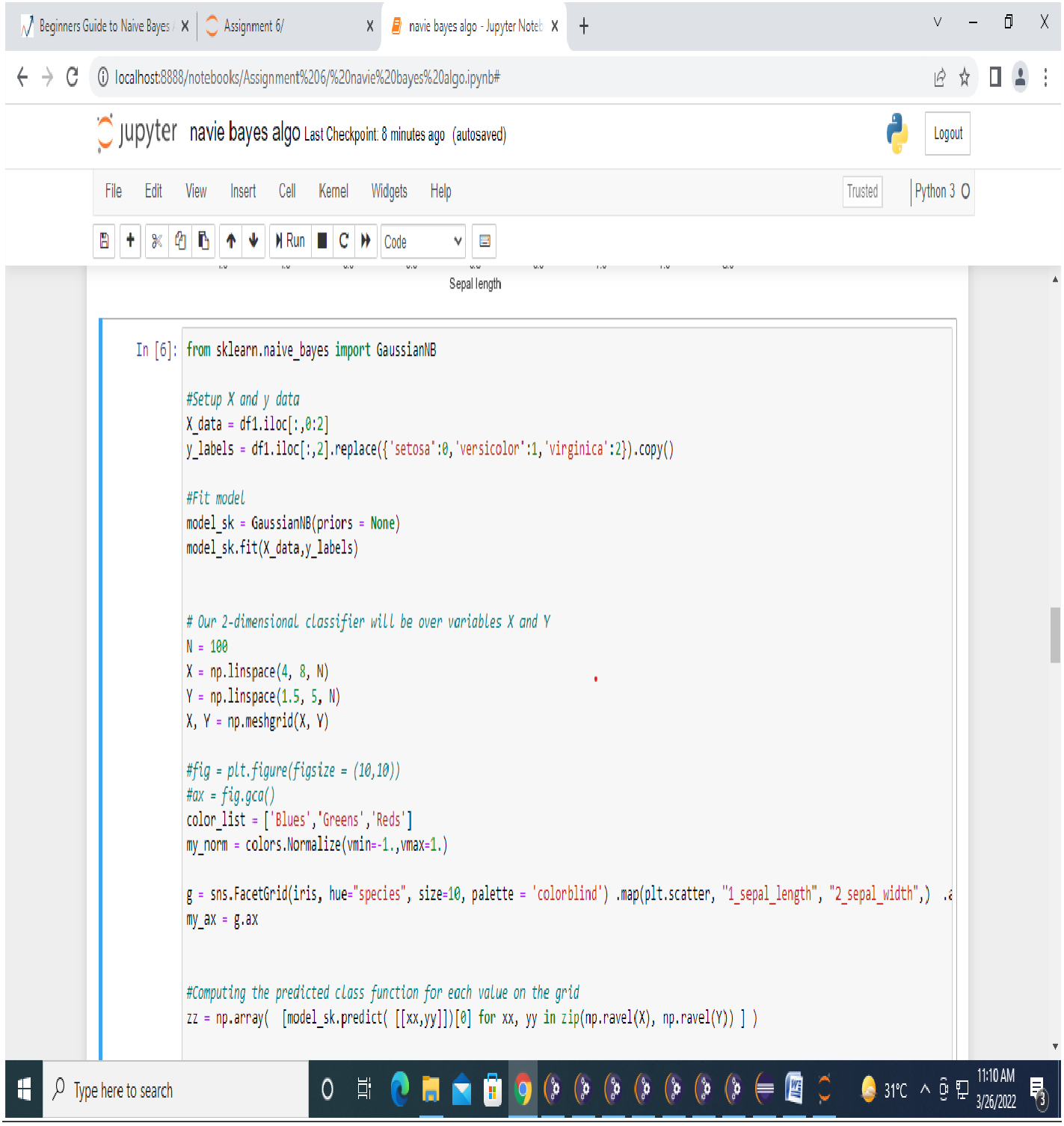
We are performing a train test split on our dataset. We are providing the test size as 0.20, that means our training sample contains 320 training set and test sample contains 80 test set

Feature scaling

Next, we are doing feature scaling to the training and test set of independent variables

Training the Naive Bayes model on the training set

We can evaluate our matrix using the confusion matrix and accuracy score by comparing the predicted and actual test value



Beginners Guide to Naive Bayes | Assignment 6/ | navie bayes algo - Jupyter Noteb

localhost:8888/notebooks/Assignment%206/%20navie%20bayes%20algo.ipynb#

jupyter navie bayes algo Last Checkpoint: 9 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

4.0 4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0

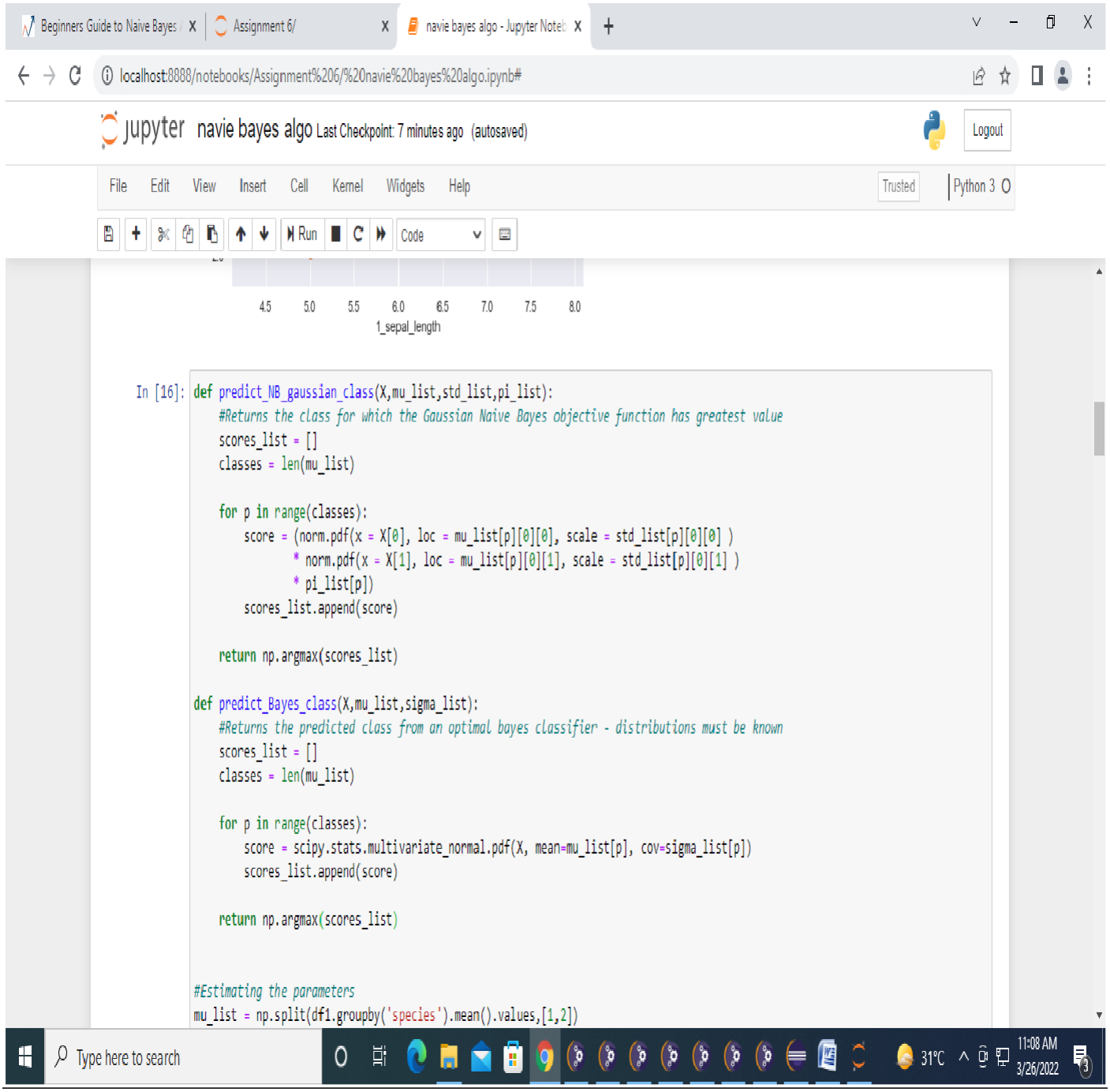
Sepal length

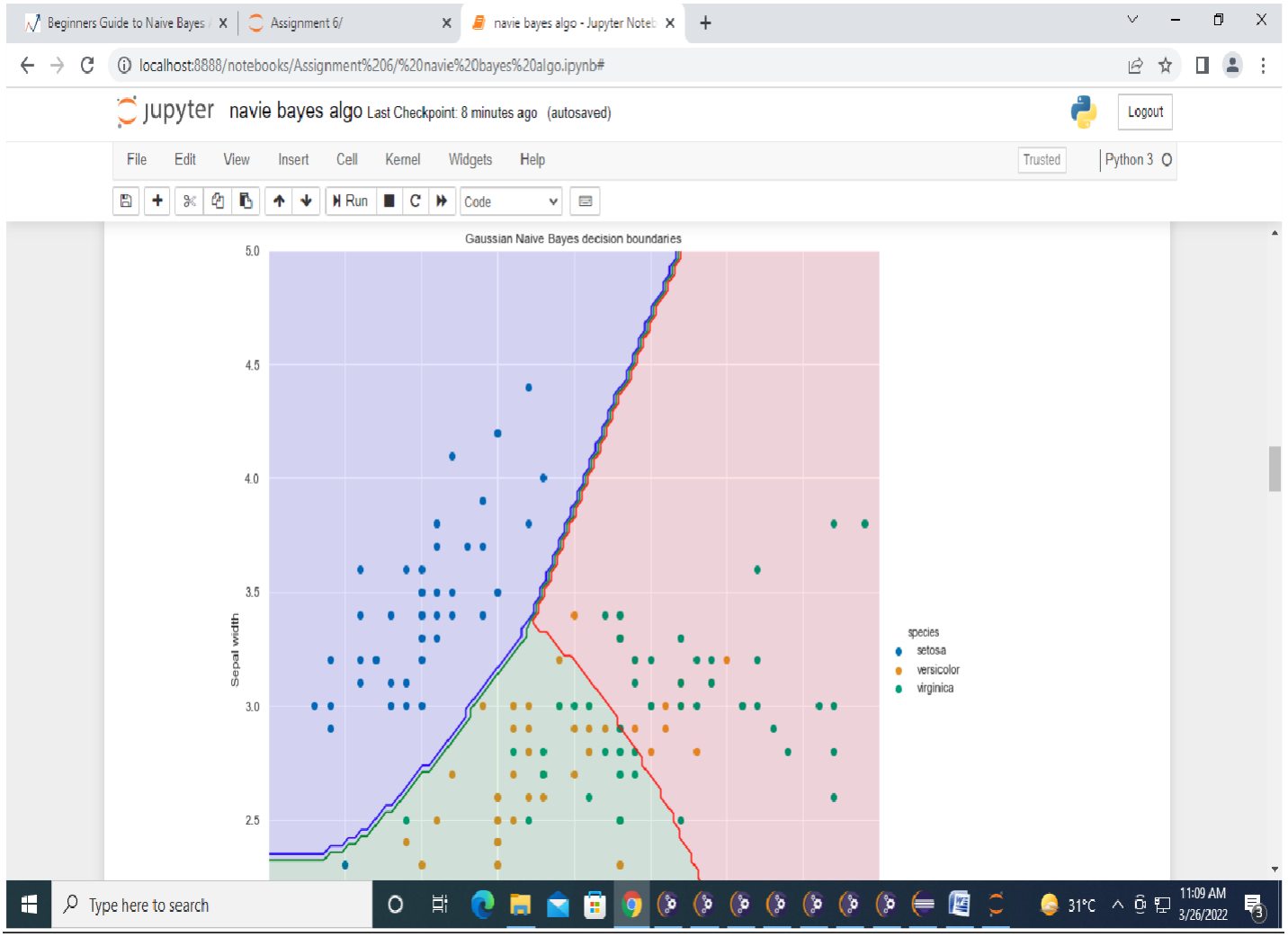
In [7]:
```
#Numpy accuracy
y_pred = np.array(  [predict_NB_gaussian_class( np.array([xx,yy]).reshape(-1,1), mu_list, std_list, pi_list)
                     for xx, yy in zip(np.ravel(X_data.values[:,0]), np.ravel(X_data.values[:,1])) ] )
display(np.mean(y_pred == y_labels))

#Sklearn accuracy
display(model_sk.score(X_data,y_labels))
```

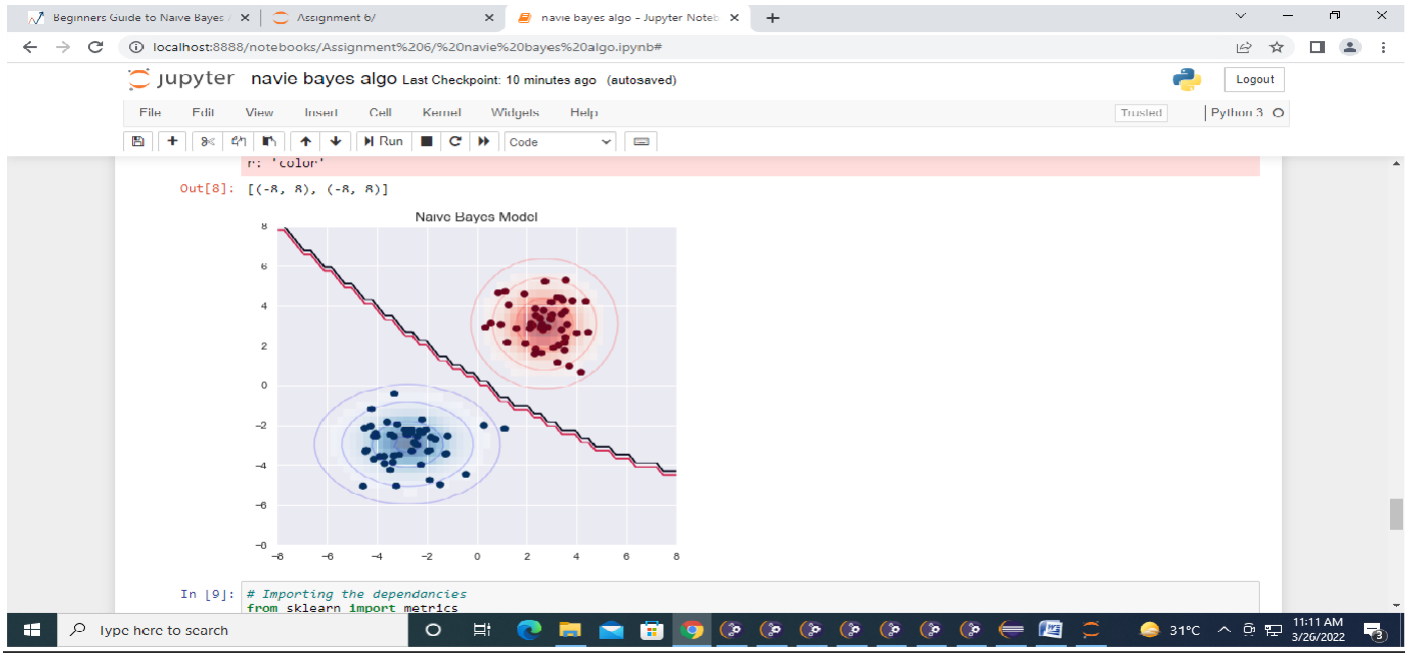
```
0.78

0.78
```

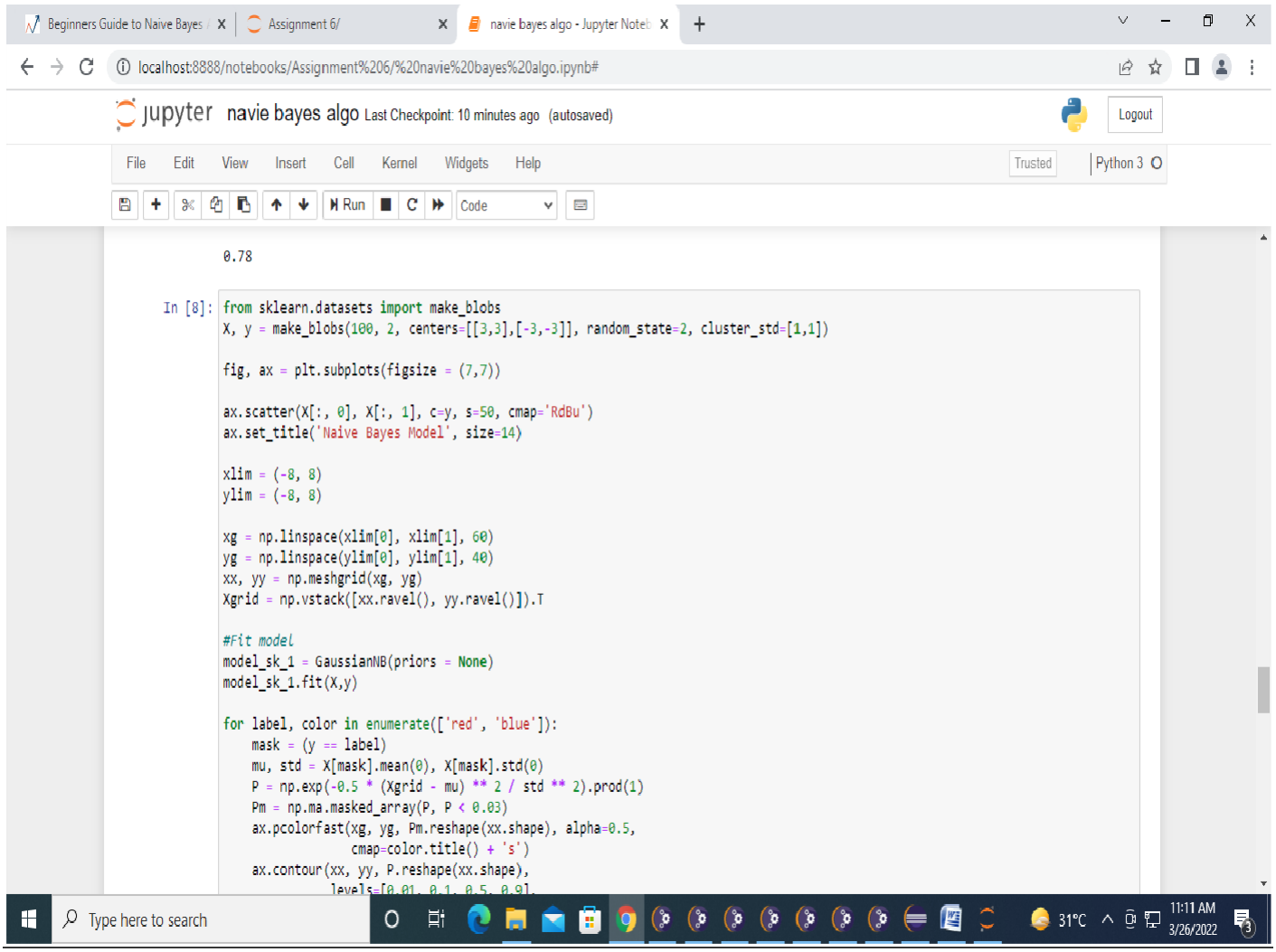
In [8]:
```
from sklearn.datasets import make_blobs
X, y = make_blobs(100, 2, centers=[[3,3],[-3,-3]], random_state=2, cluster_std=[1,1])

fig, ax = plt.subplots(figsize = (7,7))

ax.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='RdBu')
ax.set_title('Naive Bayes Model', size=14)

xlim = (-8, 8)
ylim = (-8, 8)

xg = np.linspace(xlim[0], xlim[1], 60)
yg = np.linspace(ylim[0], ylim[1], 40)
xx, yy = np.meshgrid(xg, yg)
Xgrid = np.vstack([xx.ravel(), yy.ravel()]).T

#Fit model
```

Type here to search 31°C 11:11 AM 3/26/2022

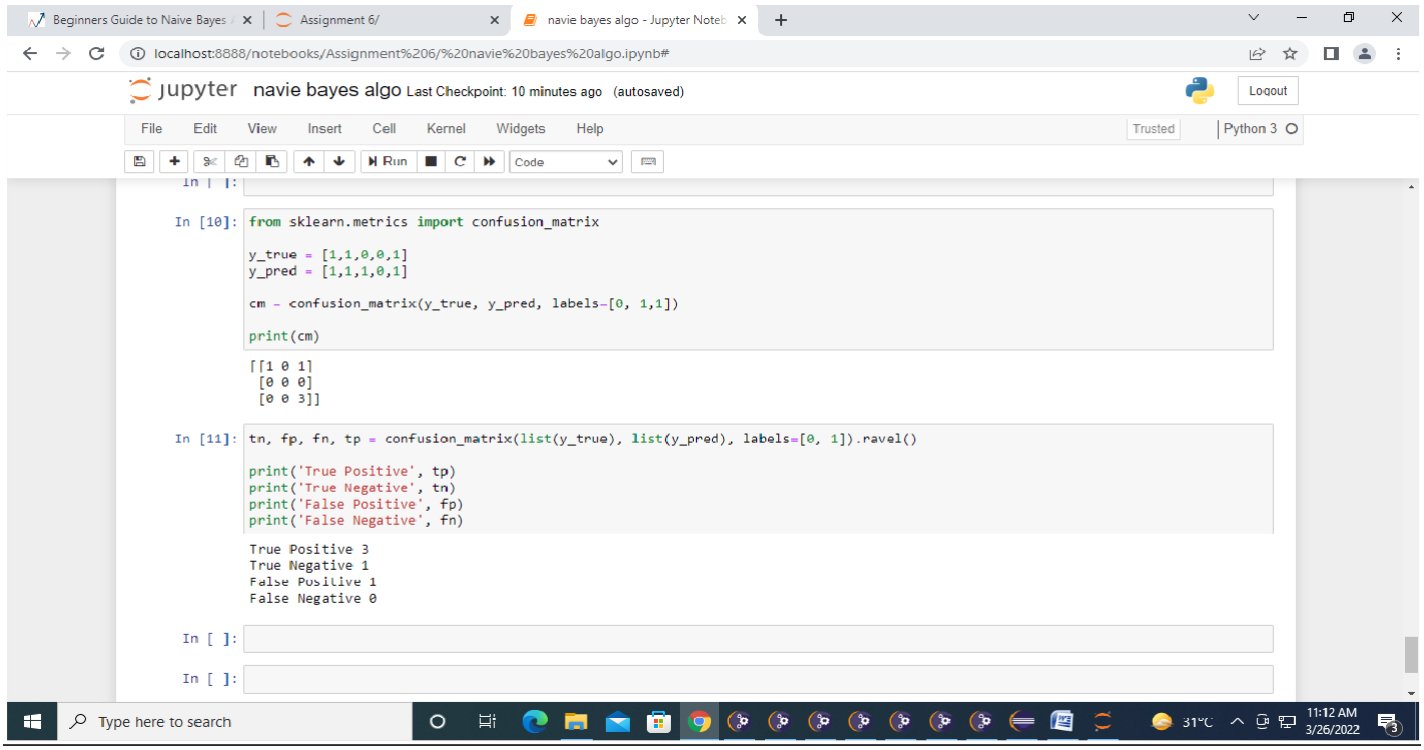

Beginners Guide to Naive Bayes
Assignment 6/
navie bayes algo - Jupyter Noteb
localhost:8888/notebooks/Assignment%206/%20navie%20bayes%20algo.ipynb#
jupyter navie bayes algo Last Checkpoint: 7 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0
1_sepal_length
In [16]:
def predict_NB_gaussian_class(X,mu_list,std_list,pi_list):
#Returns the class for which the Gaussian Naive Bayes objective function has greatest value
scores_list = []
classes = len(mu_list)
for p in range(classes):
score = (norm.pdf(x = X[0], loc = mu_list[p][0][0], scale = std_list[p][0][0])
* norm.pdf(x = X[1], loc = mu_list[p][0][1], scale = std_list[p][0][1])
* pi_list[p])
scores_list.append(score)
return np.argmax(scores_list)
def predict_Bayes_class(X,mu_list,sigma_list):
#Returns the predicted class from an optimal bayes classifier - distributions must be known
scores_list = []
classes = len(mu_list)
for p in range(classes):
score = scipy.stats.multivariate_normal.pdf(X, mean=mu_list[p], cov=sigma_list[p])
scores_list.append(score)
return np.argmax(scores_list)
#Estimating the parameters
mu_list = np.split(df1.groupby('species').mean().values,[1,2])
Type here to search
31°C
11:08 AM
3/26/2022


Beginners Guide to Naive Bayes
Assignment 6/
navie bayes algo - Jupyter Noteb
localhost:8888/notebooks/Assignment%206/%20navie%20bayes%20algo.ipynb#
jupyter navie bayes algo Last Checkpoint: 8 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Gaussian Naive Bayes decision boundaries
Sepal width
5.0
4.5
4.0
3.5
3.0
2.5
species
setosa
versicolor
virginica
Type here to search
31°C
11:09 AM
3/26/2022


Beginners Guide to Naive Bayes
Assignment 6/
navie bayes algo - Jupyter Noteb
localhost:8888/notebooks/Assignment%206/%20navie%20bayes%20algo.ipynb#
jupyter navie bayes algo Last Checkpoint: 10 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Out[8]: [(-8, 8), (-8, 8)]
Naive Bayes Model
In [9]: # Importing the dependancies
from sklearn import metrics
Type here to search
31°C
11:11 AM
3/26/2022



Confusion Matrix TP, FP ,TN, FN, Accuracy, Error rate, Precision, Recall,



Conclusion:

Naive Bayes algorithms are mostly used in sentiment analysis, spam filtering, recommendation systems etc. They are fast and easy to implement but their biggest disadvantage is that the requirement of predictors to be independent. In most of the real life cases, the predictors are dependent, this hinders the performance of the classifier.

Viva Questions:

1.What mathematical concept Naive Bayes is based on?

2.What are the different types of Naive Bayes classifiers?

3.Is Naive Bias a classification algorithm or regression algorithm?

4.What are some benefits of Naive Bayes?

5.What are the cons of Naive Bayes classifier?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 7

Title of the Assignment: Text Analytics

Extract sample document and apply following document preprocessing methods:
Tokenization, POS Tagging, Stop word removal, Stemming & Lemmatization.

Create representation of document by calculating Term frequency and Inverse Document Frequency.

Objective of the Assignment: Able to perform the Document Processing Methods using Python and NLTK.

Prerequisite:

Basic of Python Programming

Concept of Data Preprocessing, Data Formatting, Data Normalization and Data Cleaning.

Installation and basic knowledge about ntlk.

Outcome: Perform text preprocessing

CO Relevance: CO4

PO/PSOs Relevance: PO1, PO2, PO3, PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

1. **Introduction to NPL (Natural Language Processing)**
2. **Setting up the Environment**
3. **Tokenization**
4. **Stop Words**
5. **Counting Words**

6. **Stemming Words**
7. **Word Groups**
   - ✓ **Bigrams**
   - ✓ **Trigrams**
   - ✓ **Ngrams**
8. **Lemmatization**
9. **POS Tagging**

Introduction to NPL (Natural Language Processing)

Computers speak their own language, the binary language. Thus, they are limited in how they can interact with us humans; expanding their language and understanding our own is crucial to set them free from their boundaries.

NLP is an abbreviation for natural language processing, which encompasses a set of tools, routines, and techniques computers can use to process and understand human communications. Not to be confused with speech recognition, NLP deals with understanding the meaning of words other than interpreting audio signals into those words.

If you think NLP is just a futuristic idea, you may be shocked to know that we are likely to interact with NLP every day when we perform queries in Google when we use translators online when we talk with Google Assistant or Siri. NLP is everywhere, and to implement it in your projects is now very reachable thanks to libraries such as NLTK, which provide a huge abstraction of the complexity.

In this article, we'll discuss how to work with NLP using the NLTK library with python.

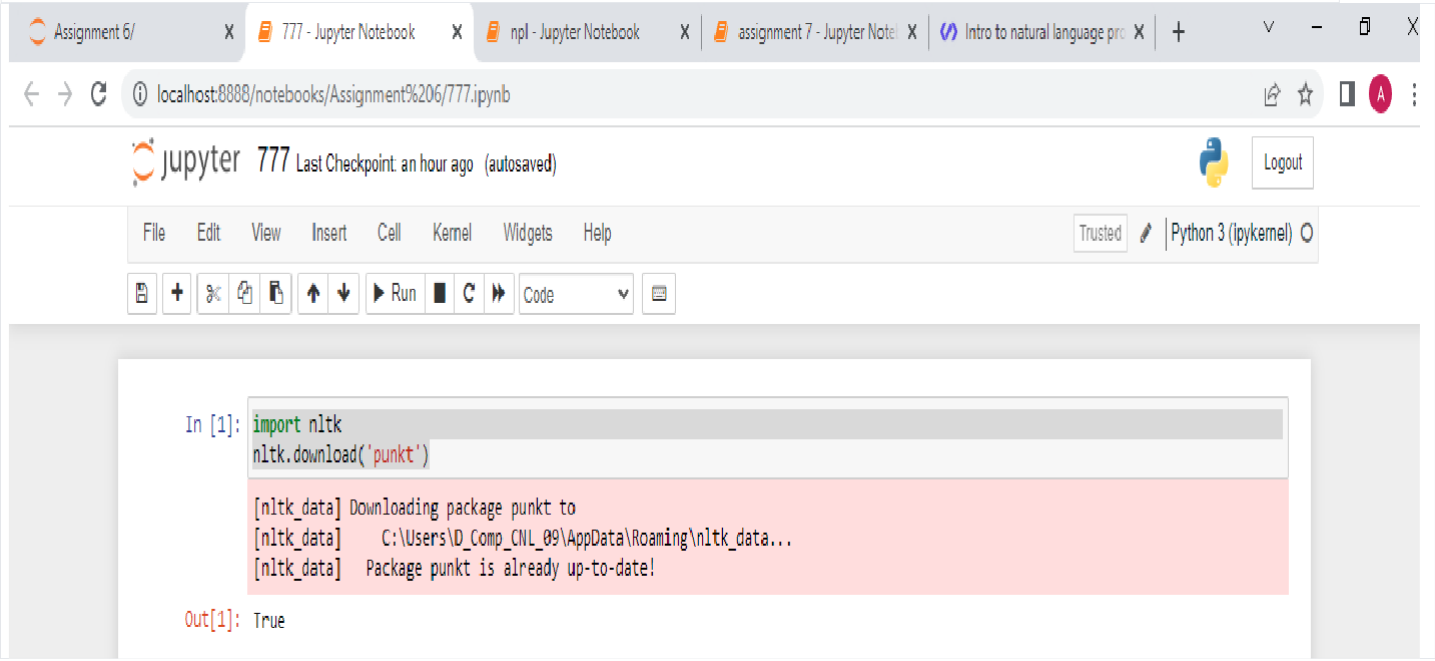
1. Setting up the Environment

For making things easier, we will use jupyter notebooks with Google Colab , and you can follow every step we do by accessing the complementary source code here .

Once you have a jupyter notebook, or any environment of your choice set up, make sure you install the nltk library by using the following command:

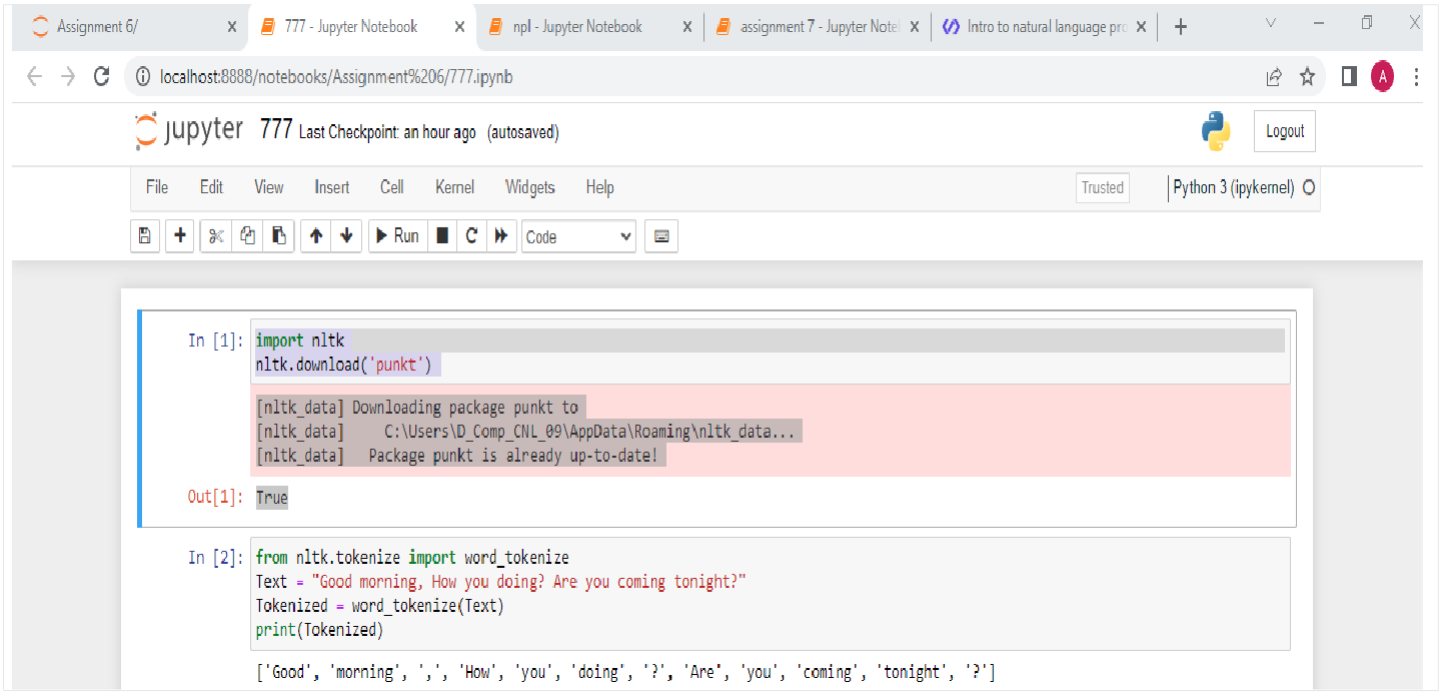
NLTK is a huge library that provides a lot of different tools to work with language. While some functions are available with the library itself, some modules require additional downloads. **punkt** is a

module to work with tokenization, which is the process of separating a paragraph into chunks or words, and it's usually a first step in the process of text analysis. Before starting, make sure you download the module.

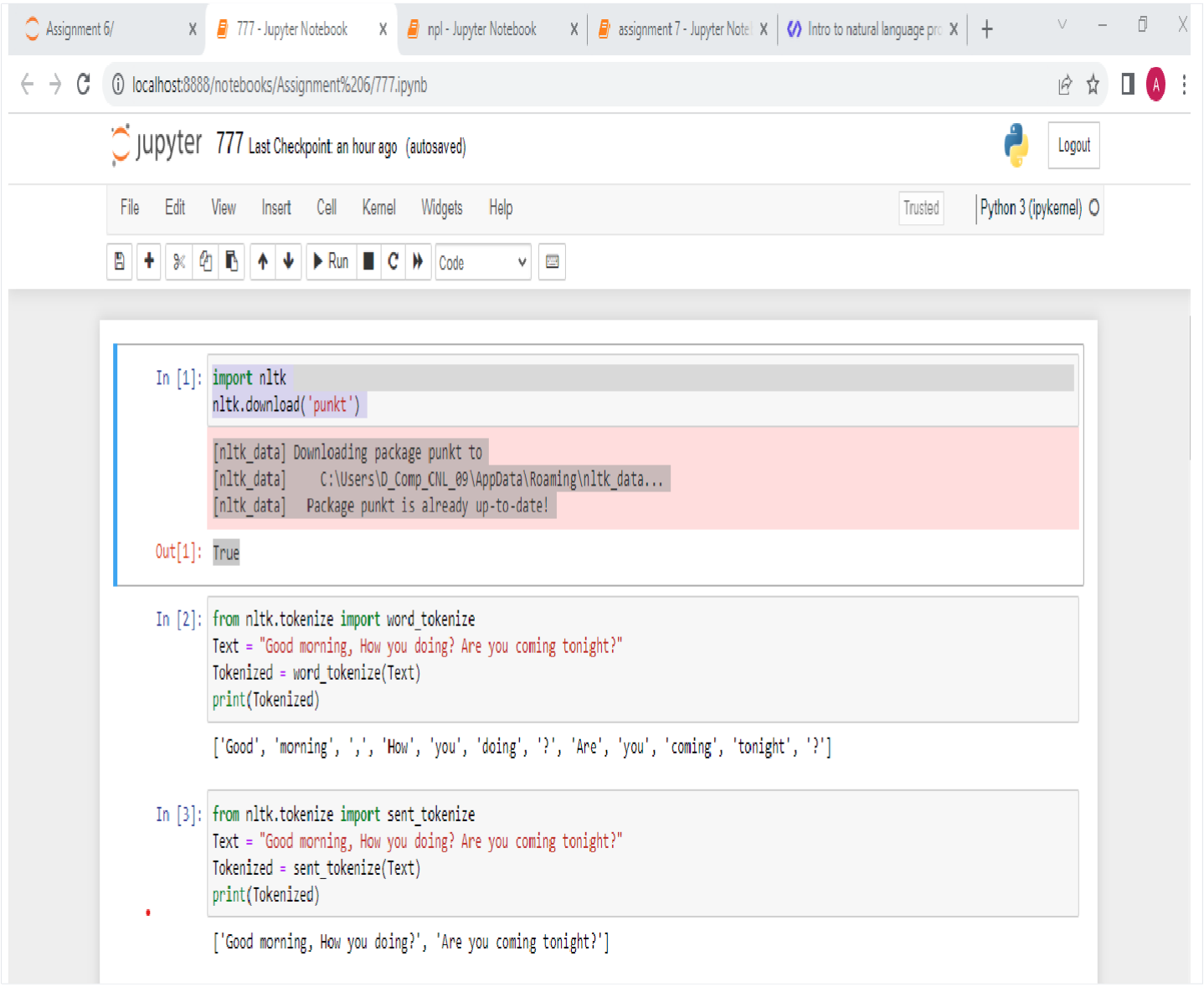


2.Tokenization

NLTK is a huge library that provides a lot of different tools to work with language. While some functions are available with the library itself, some modules require additional downloads. **punkt** is a module to work with tokenization, which is the process of separating a paragraph into chunks or words, and it's usually a first step in the process of text analysis. Before starting, make sure you download the modul

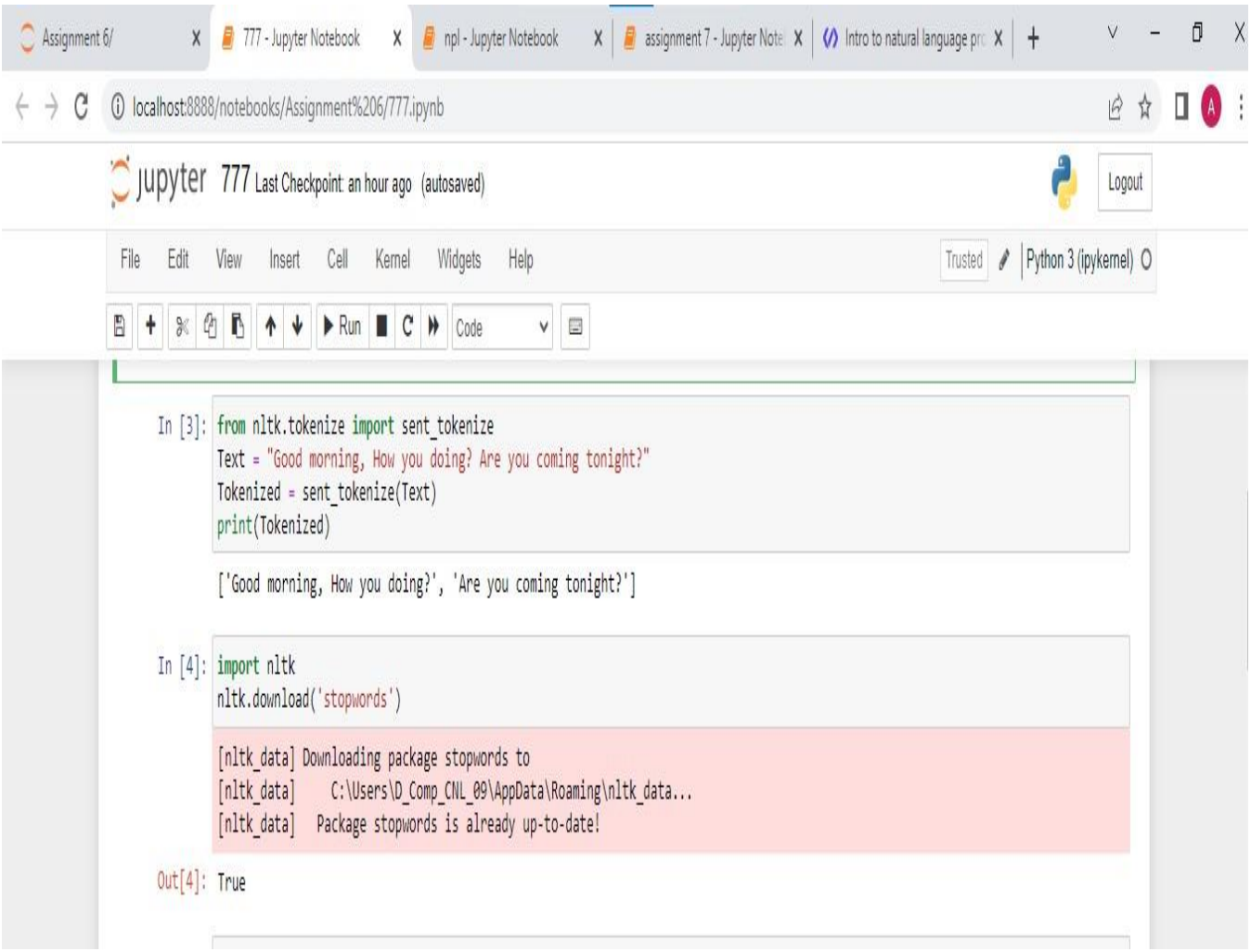


This first function, **word_tokenize** will split a text into words and symbols, however there's more you can do with **punkt**, such as separating a paragraph into sentences.

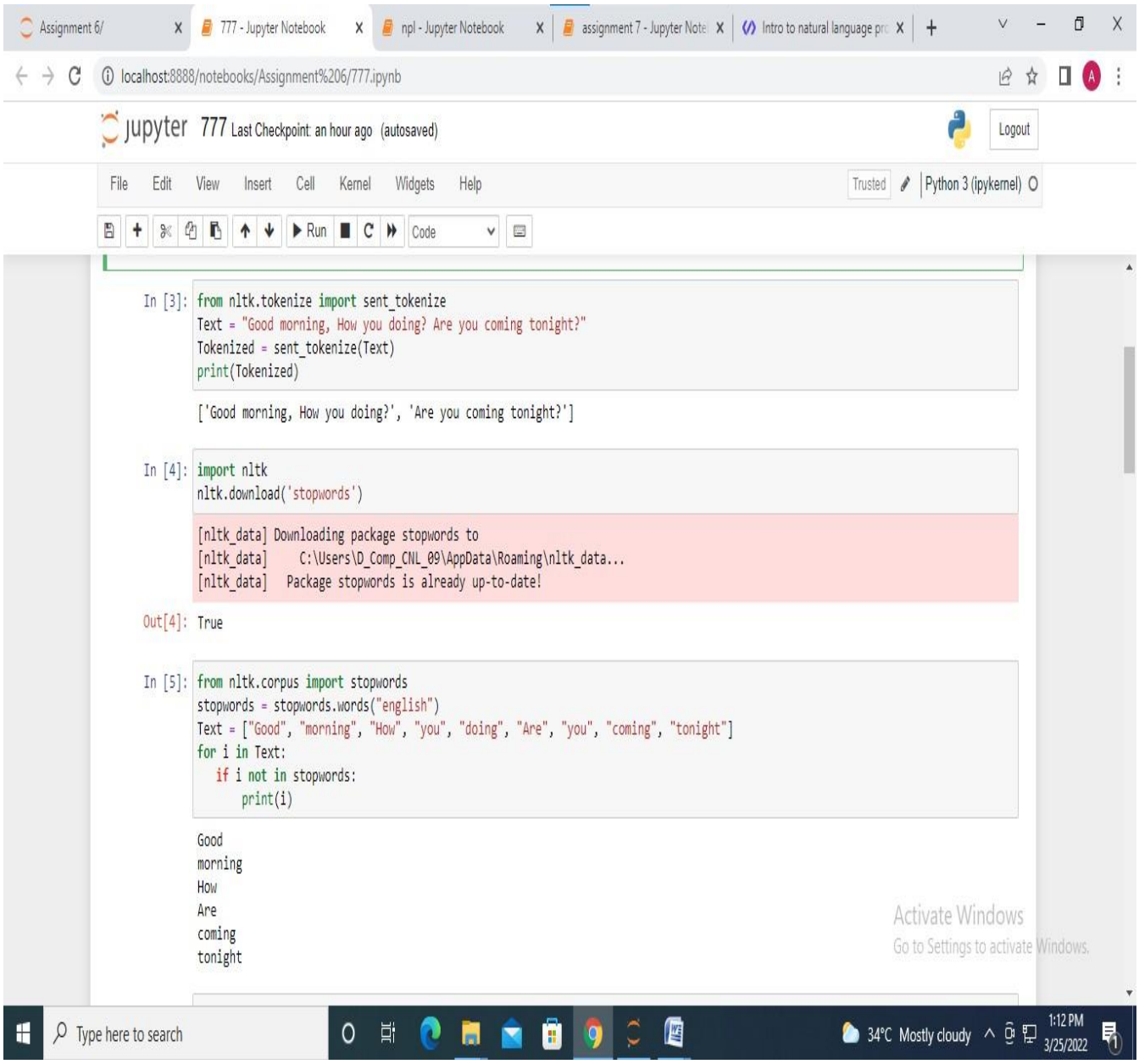


3.Stop Words

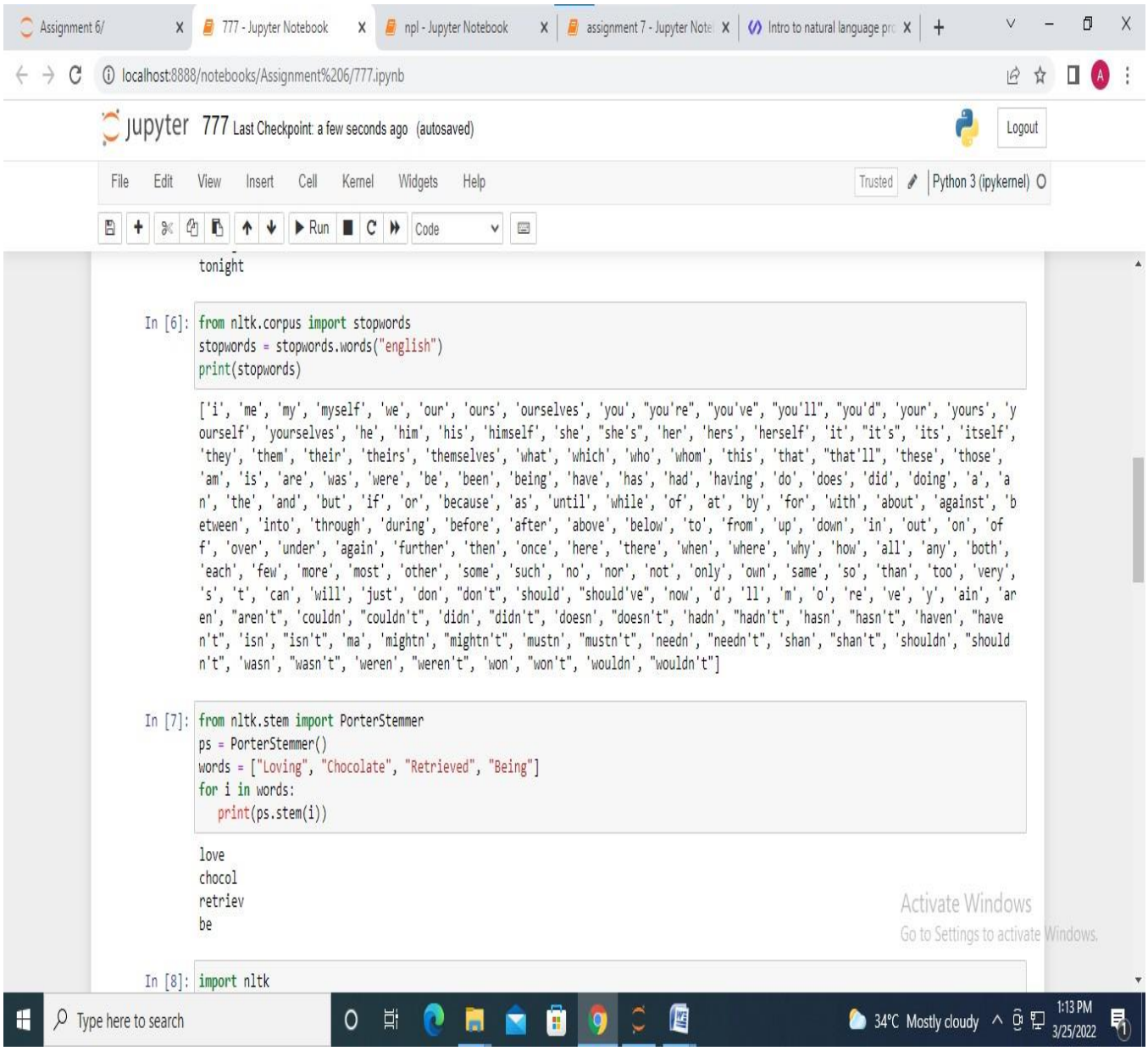
A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query. It’s important in certain situations to ignore such words, and thus having a dictionary of them can become really handy, especially when we need to deal with multiple languages. NLKT provides a module to work with step words, let’s download it next:



Stop words is a simple list of words, so we can operate with it very easily, for example by writing a small routing to get a list of words without stop words in it:

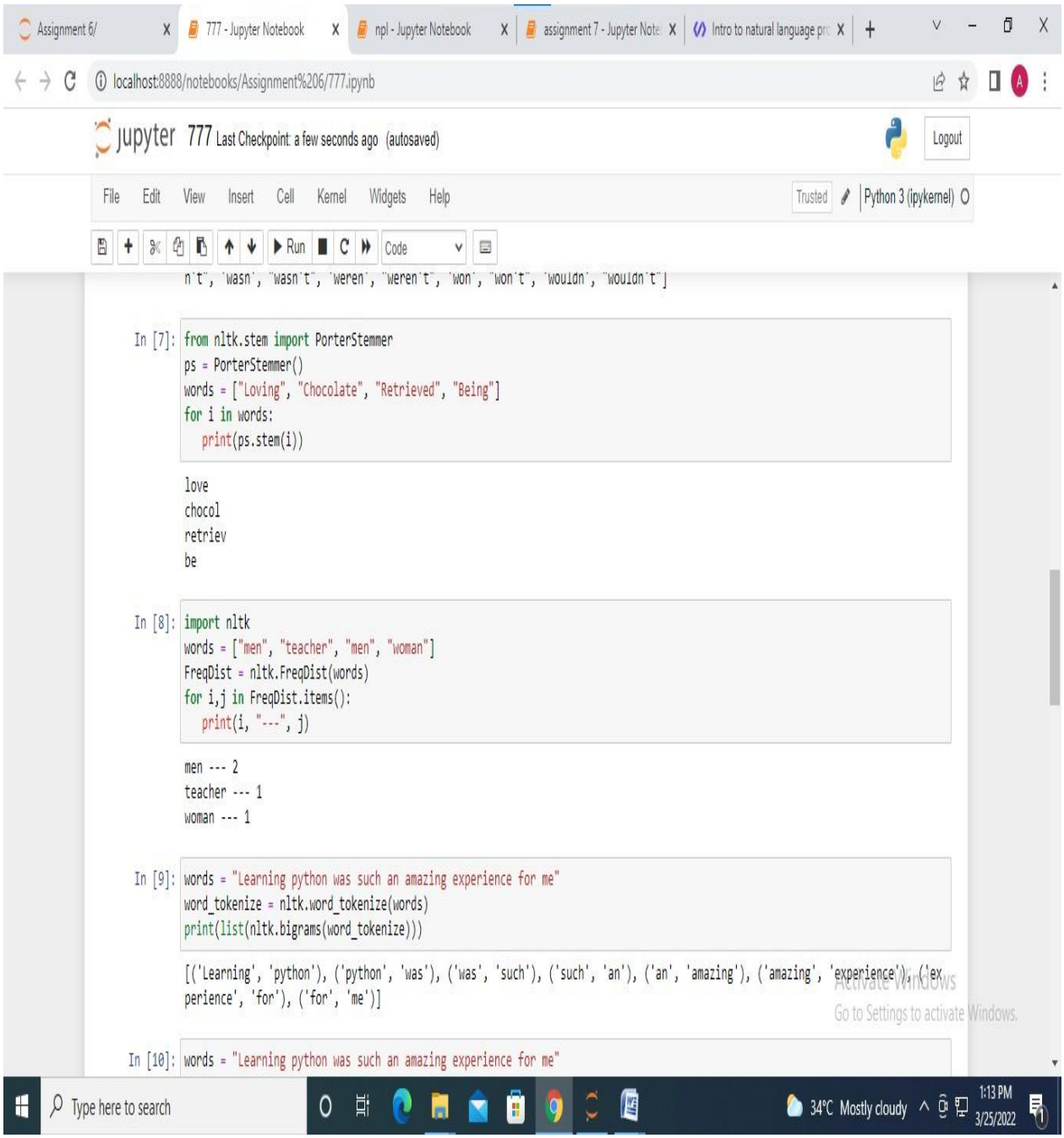


Since we are given a simple list of words, we can simple print it to see all of them for a particular language:



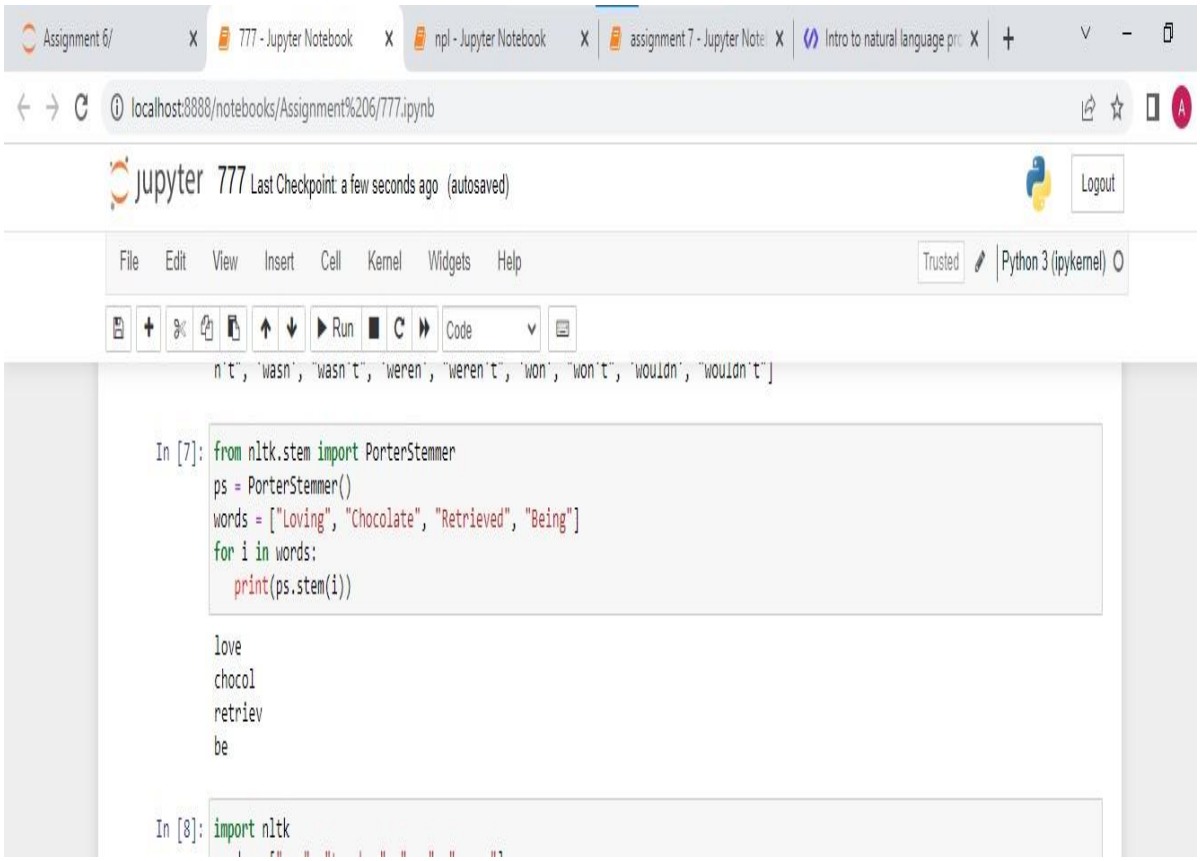
4. Counting Words

Counting how many times each word appears can be very helpful in the context of text analysis. NLTK provides us a neat method to calculate the frequency of words in a text called **FreqDist**.



5.Stemming Words

A word stem is the base or root form of a word, for example the word “loving” has roots in the word “love”, or being” on the word “be”. Stemming is the process to which we transform a given word into its stem word. This is a very complex task to do, words can be written in many forms, and different words have different ways to get its stem. Thankfully, NLTK makes it really easy for us to achieve this, let’s see how:



This simplification of a word can be very helpful in search engines to prevent different ways of writing the same word to be ignored on the search criteria.

6.Word Groups

Oftentimes we see some words being used together to give a specific meaning, for example “let’s go”, “best performance” and others. On text analysis it is important to capture these words as pairs as seeing them together can make a big difference in the comprehension of the text.

Bigrams

NLTK provides a few methods to do exactly that, and we will start with bigrams, which is a method to extract pairs of connected words:

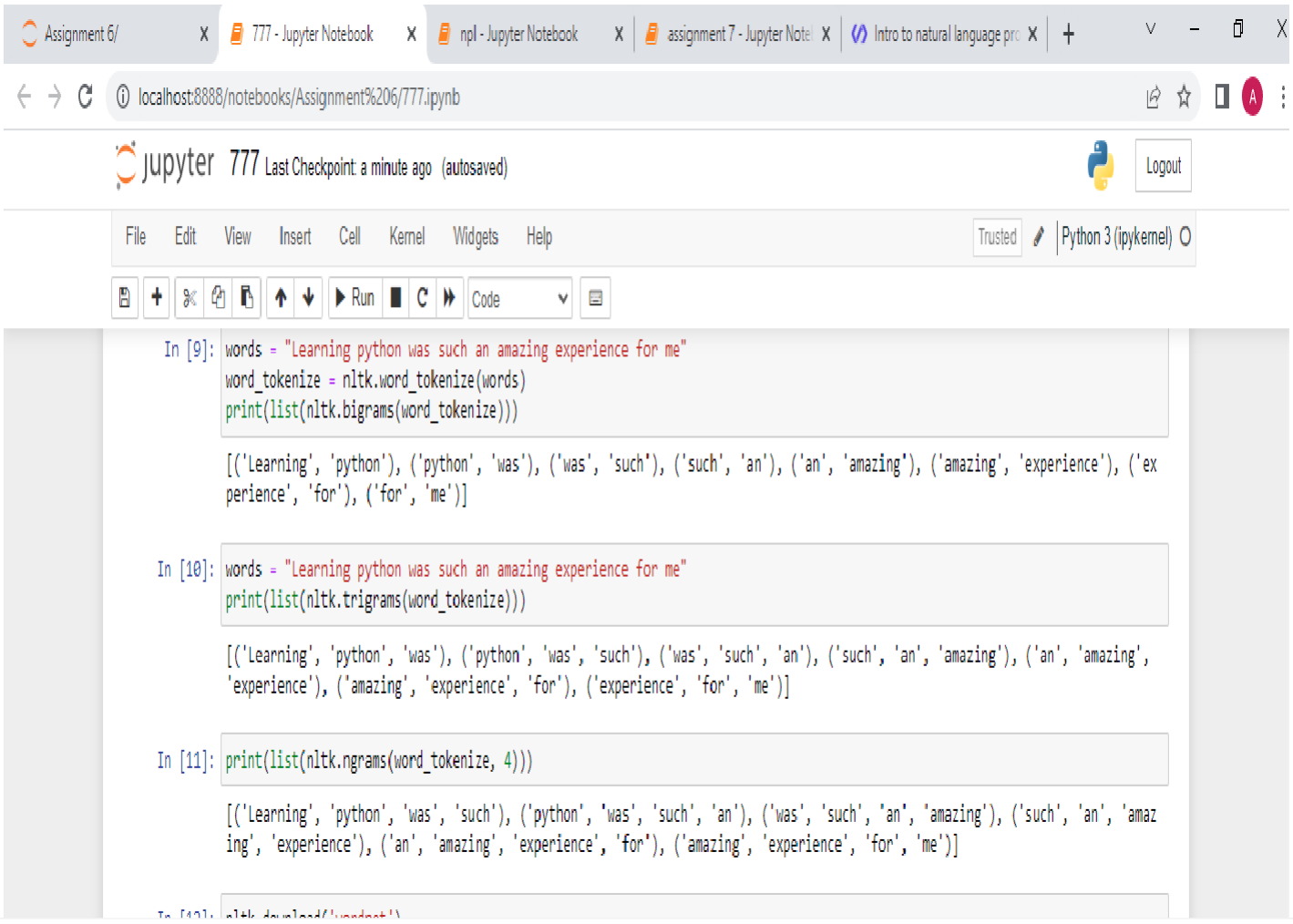
Trigrams

Bigrams is the two words that occur together always but trigrams are the same as bigrams but with three words and there almost no difference in the code:

Ngrams

The Ngrams are also some words or letters or symbols that appear together in a single phrase or document such as the previous two methods bigrams and trigrams but here you can specify the words numbers. Let's see an example:

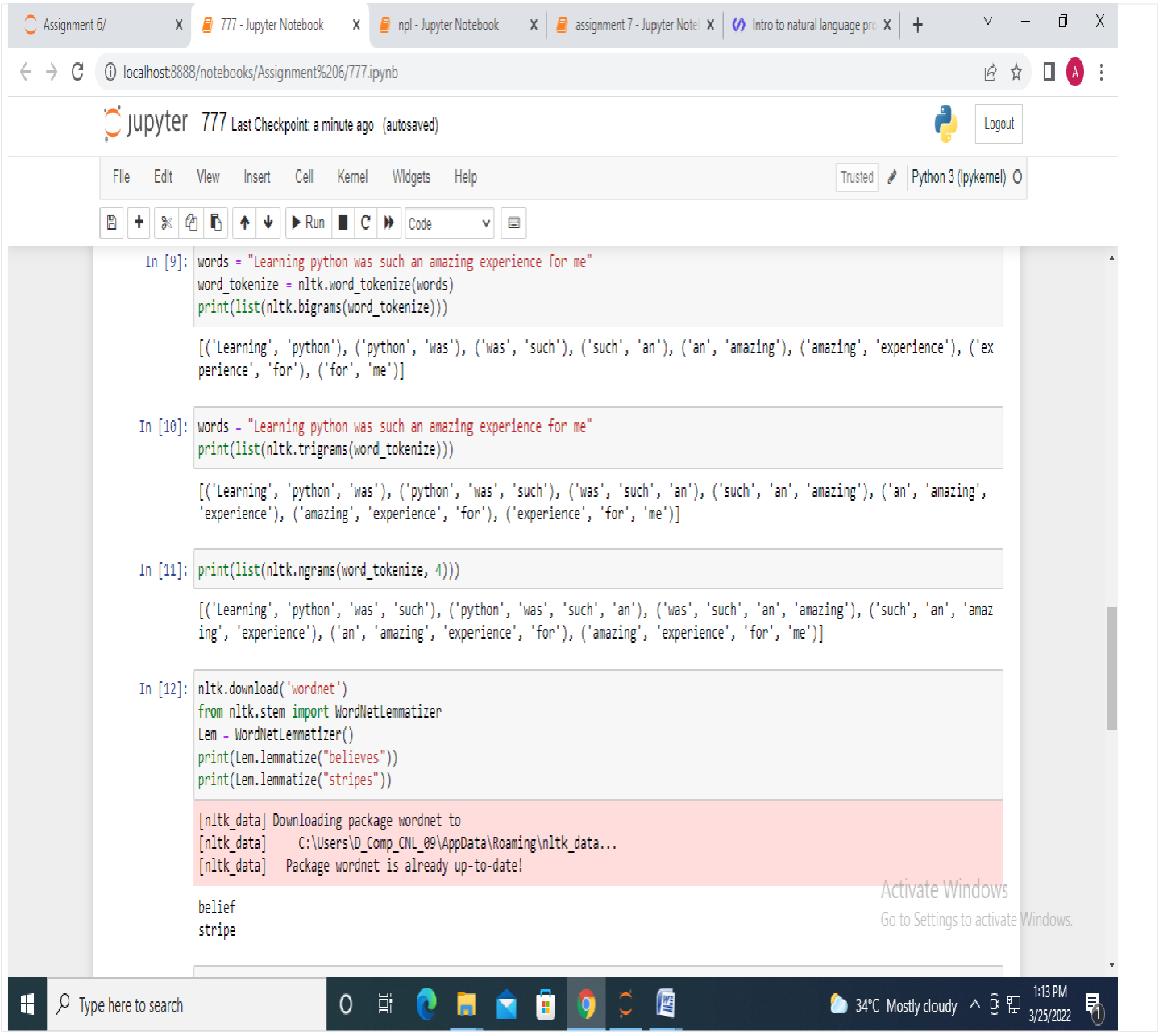
Output:



7.Lemmatization

The concept of lemmatization is very similar to stemming words, just this last one only removes the prefix or suffix of the word and sometimes makes some spelling errors and the lemmatization converts it to its rel base word. Let's see an example but before we do that we need to download the WordNet package using NLTK:

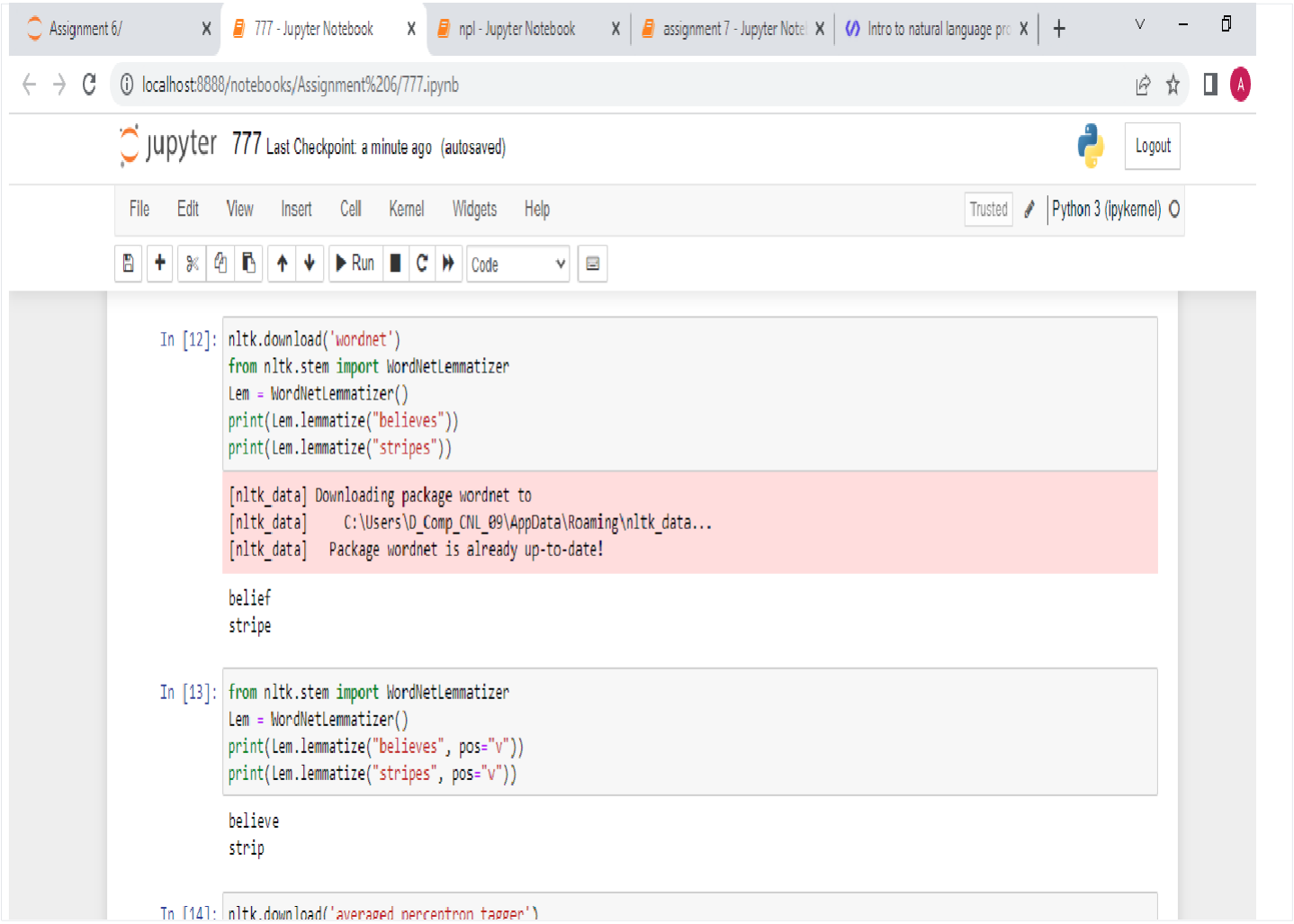
Output:



When you run the code it will convert every word to its base like "believes" to "belief" and "stripes" to "stripe" and so on. The nice thing about this package known as WordNetLemmatizer it has an

argument called pos which stands for “part of speech” and you can specify if you want to get the verb or the adjective of the word. Let’ see an example:

Output:



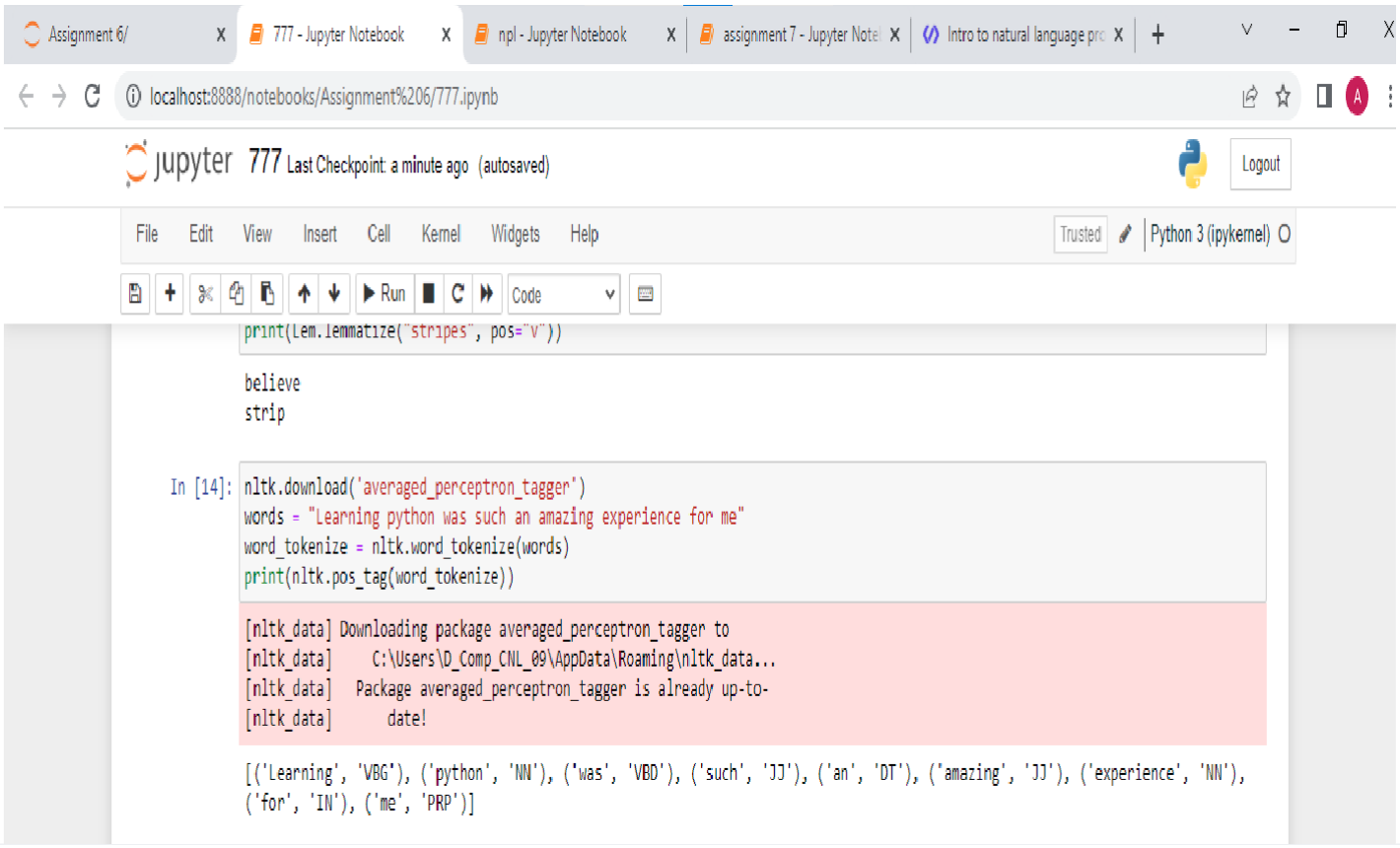
8.POS Taggers

Some time ago when you were at school you probably learned to categorize words into verbs, nouns, adjectives, etc. And today that task is pretty trivial for us humans, but if computers want to understand

human language they need to understand these concepts, they need to differentiate between an action and a target, a verb and a noun. NLTK provides us with POS (Part of Speech) to categorize words.

It's super easy to work with, so let's look at the code:

Output:

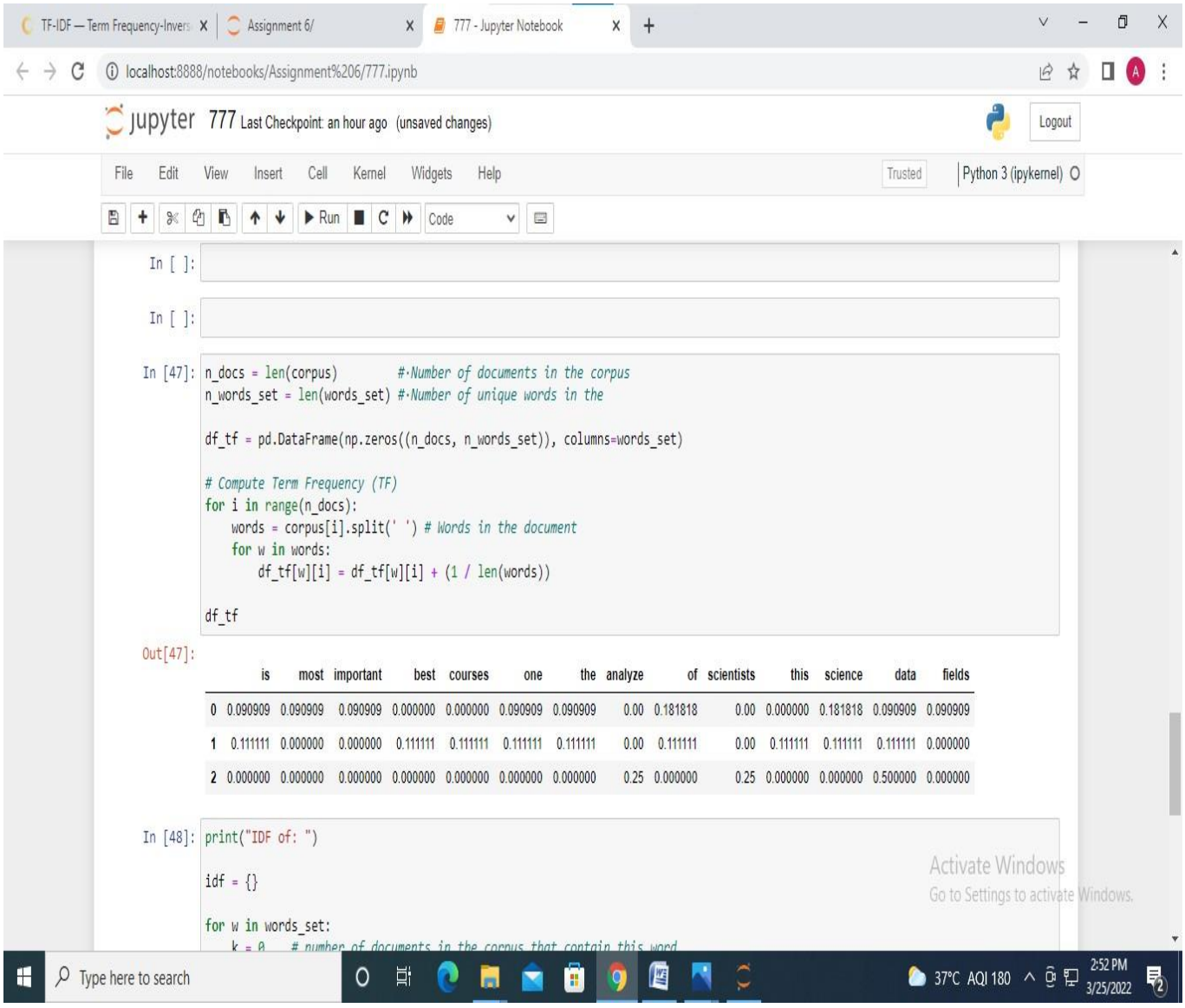


Term Frequency

TF of a term or word is the number of times the term appears in a document compared to the total number of words in the document.

TF=number of times the term appears in the document total number of terms in the document

Some popular python libraries have a function to calculate T F-IDF. The popular machine learning library Sklearn has TfidfVectorizer() function (docs).



Inverse Document Frequency

IDF of a term reflects the proportion of documents in the corpus that contain the term. Words unique to a small percentage of documents (e.g., technical jargon terms) receive higher importance values than words common across all documents (e.g., a, the, and).
IDF=log(number of the documents in the corpusnumber of documents in the corpus contain the term)

The TF-IDF of a term is calculated by multiplying TF and IDF scores.

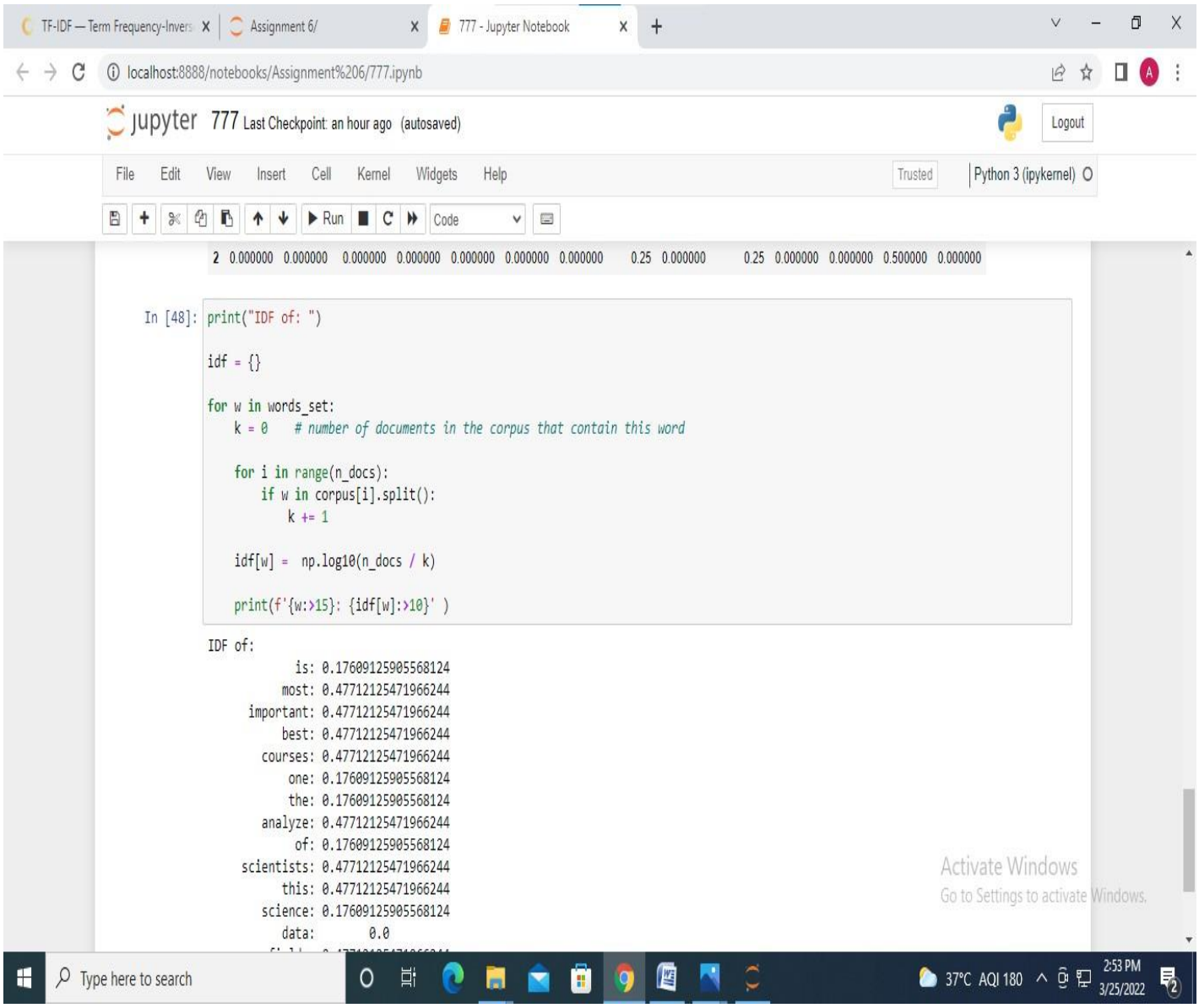
TF-IDF=TF∗IDF

Translated into plain English, importance of a term is high when it occurs a lot in a given document and rarely in others. In short, commonality within a document measured by TF is

balanced by rarity between documents measured by IDF. The resulting TF-IDF score reflects the importance of a term for a document in the corpus.

TF-IDF is useful in many natural language processing applications. For example, Search Engines use TF-IDF to rank the relevance of a document for a query. TF-IDF is also employed in text classification, text summarization, and topic modeling.

Note that there are some different approaches to calculating the IDF score. The base 10 logarithm is often used in the calculation. However, some libraries use a natural logarithm. In addition, one can be added to the denominator as follows in order to avoid division by zero.



Conclusion:

TF-IDF model is one of the most widely used models for text to numeric conversion. In this article, we briefly reviewed the theory behind the TF-IDF model. Finally, we implemented a TF-IDF model from scratch in Python. In the next article, we will see how to implement the N-Gram model from scratch in Python.

Viva Question:

1. What is Data Preprocessing? What preprocessing steps do you know?
2. What is the difference between Data Processing and Data Mining?
3. What is Lemmatizer?
4. What is Postager?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 8

Title of the Assignment: Data Visualization I

Use inbuilt dataset 'titanic'. The dataset contains 891 row and contains information about the passenger who boarded the unfortunate Titanic ship. Use the seaborn library to see if we can find any pattern in the data.

Write a code to check how the price of the ticket(column name: 'fare') for each passenger distributed by plotting a histogram.

Objective of the Assignment: Able to perform the data visualization operation using Python on any open source dataset

---

Prerequisite:
Basic of Python Programming
Concept of Data Preprocessing, Data Formatting , Data Normalization and Data Cleaning.

Outcome: Implement data visualization techniques

CO Relevance: CO5

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

Introduction to Data Visualization

Description of dataset

Implementation of Code.

What is Data Visualization?

Data visualization is a field in data analysis that deals with visual representation of data. It graphically plots data and is an effective way to communicate inferences from data.

Using data visualization, we can get a visual summary of our data. With pictures, maps and graphs, the human mind has an easier time processing and understanding any given data. Data visualization plays a significant role in the representation of both small and large data sets, but it is especially useful when we have large data sets, in which it is impossible to see all of our data, let alone process and understand it manually.

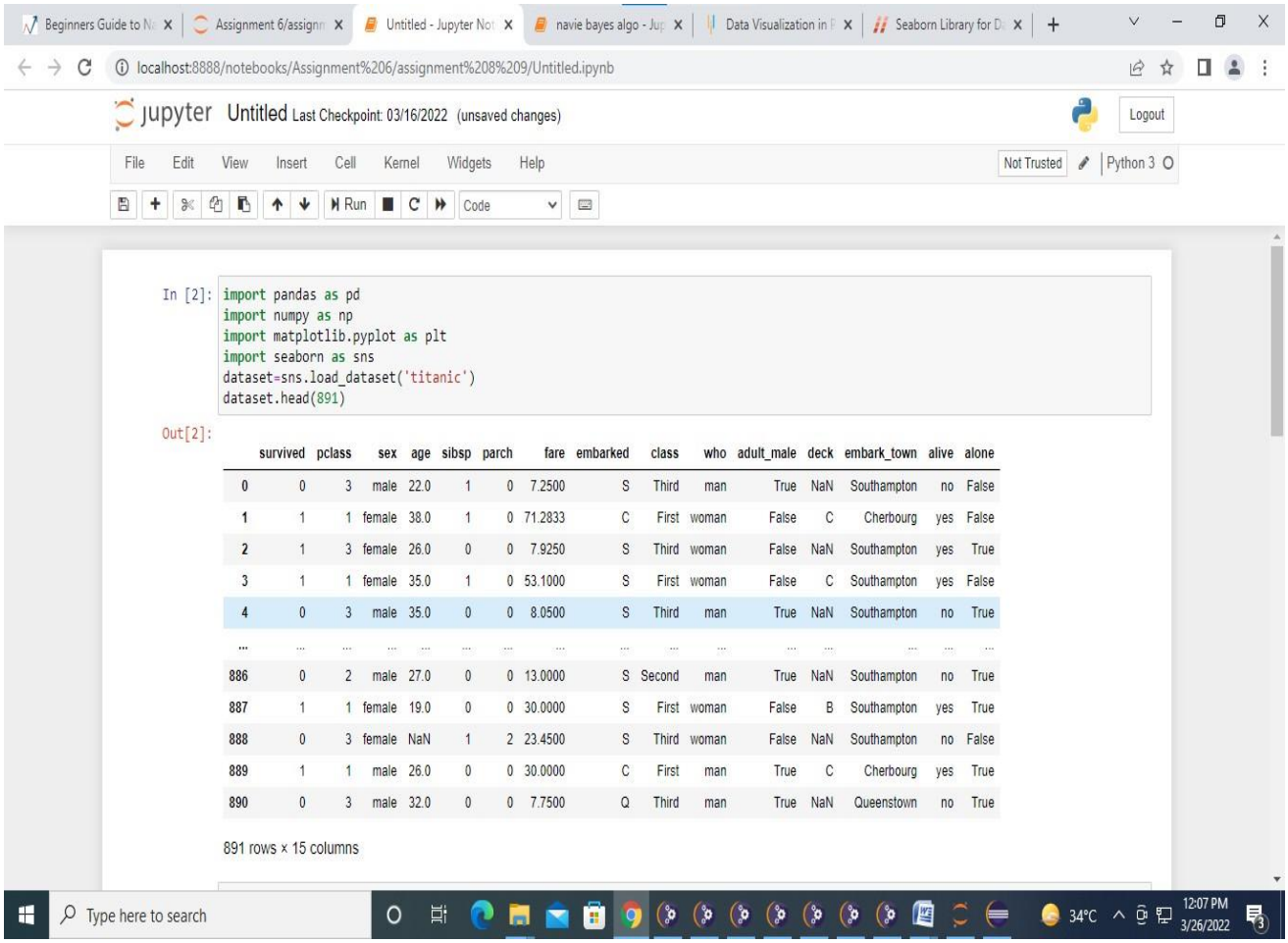
Matplotlib and Seaborn

Matplotlib and Seaborn are python libraries that are used for data visualization. They have inbuilt modules for plotting different graphs. While Matplotlib is used to embed graphs into applications, Seaborn is primarily used for statistical graphs.

But when should we use either of the two? Let's understand this with the help of a comparative analysis. The table below provides comparison between Python's two well-known visualization packages Matplotlib and Seaborn.

Implementation

- Load all important libraries
- Load dataset



	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False	B	Southampton	yes	True
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False	NaN	Southampton	no	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True	NaN	Queenstown	no	True

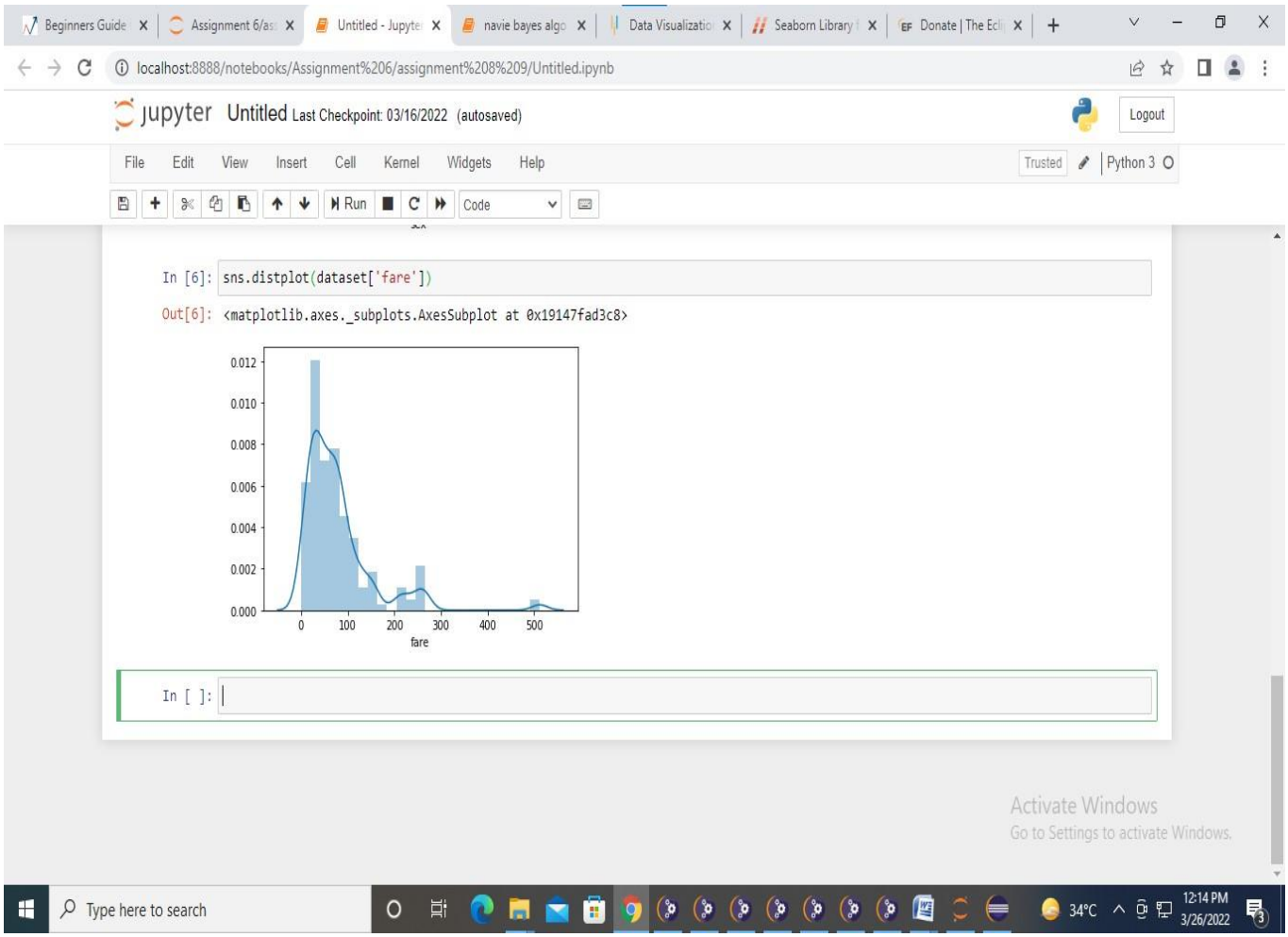
891 rows × 15 columns

Distributional Plots

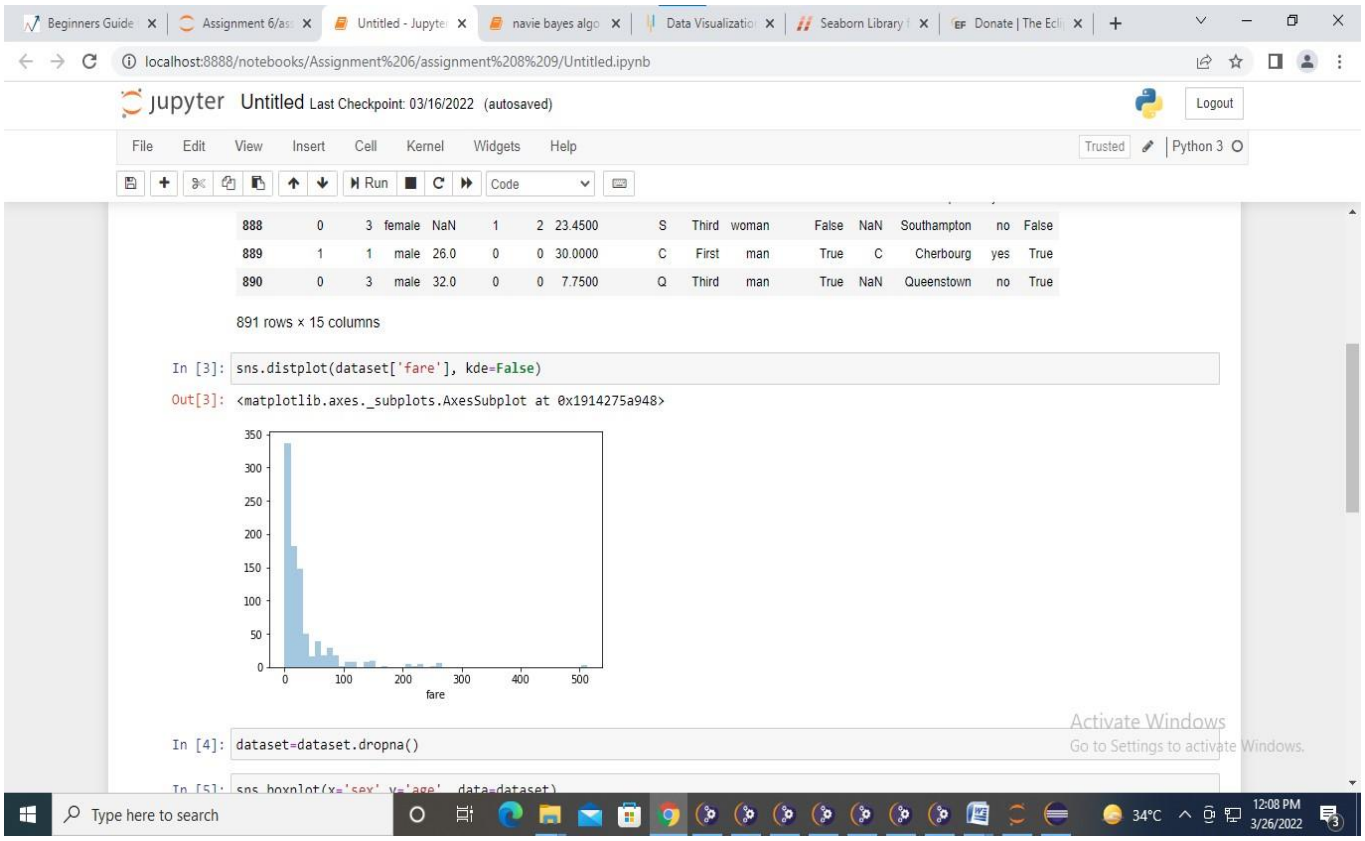
Distributional plots, as the name suggests are type of plots that show the statistical distribution of data. In this section we will see some of the most commonly used distribution plots in Seaborn.

The Dist Plot

The distplot() shows the histogram distribution of data for a single column. The column name is passed as a parameter to the distplot() function. Let's see how the price of the ticket for each passenger is distributed. Execute the following script:



Distribution of fare for each passenger find by : sns.displot(dataset[“fare”], kde=’false’)



Conclusion

Seaborn is an advanced data visualization library built on top of Matplot liblibrary. We looked at how we can draw distribution

Viva Questions:

1. What is Data Visualization?
2. What is an outlier? How would you address outliers?
3. What are different mithods for data visualization?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 9

Title of the Assignment: Data Analytics II

1. Use inbuild dataset titanic . Plot a box plot for distribution of age with respect to each gender with information about whether they survieved or not(Column name: 'sex','age')
2. Write an observation on the inference from above statistics.

Objective of the Assignment: Able to perform the data visualization operation using Python on any open source dataset

---

Prerequisite:

Basic of Python Programming

Concept of Data Preprocessing, Data Formatting , Data Normalization and Data Cleaning.

Outcome: Implement data visualization techniques on inbuilt Titanic dataset.

CO Relevance: CO5

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

Introduction to Data Visualization

Description of dataset

Implementation of Code.

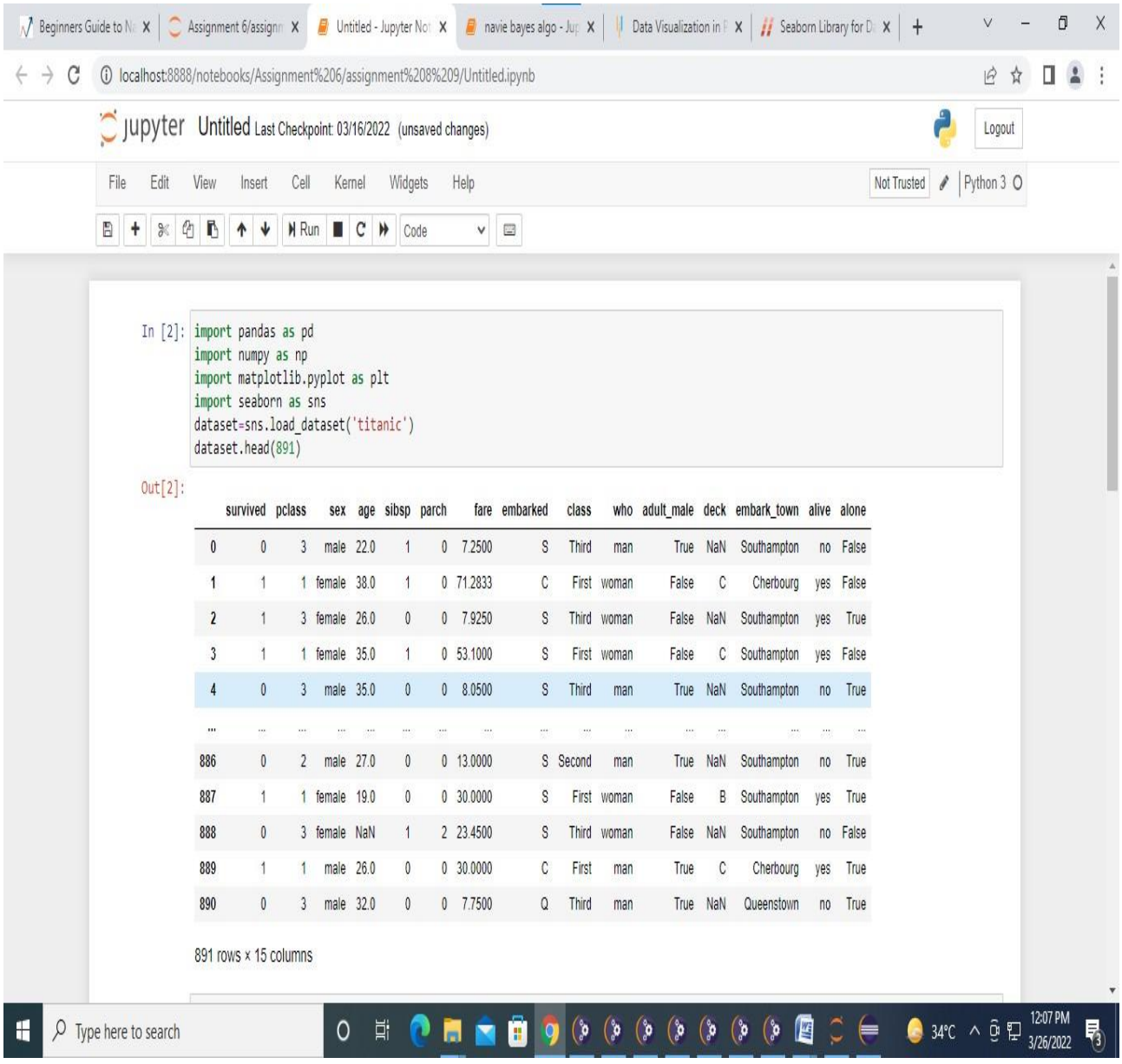
Data Visualization:

Data visualization is a field in data analysis that deals with visual representation of data. It graphically plots data and is an effective way to communicate inferences from data.

Using data visualization, we can get a visual summary of our data. With pictures, maps and graphs, the human mind has an easier time processing and understanding any given data. Data visualization plays a significant role in the representation of both small and large data sets, but it is especially useful when we have large data sets, in which it is impossible to see all of our data, let alone process and understand it manually.

Description of dataset:

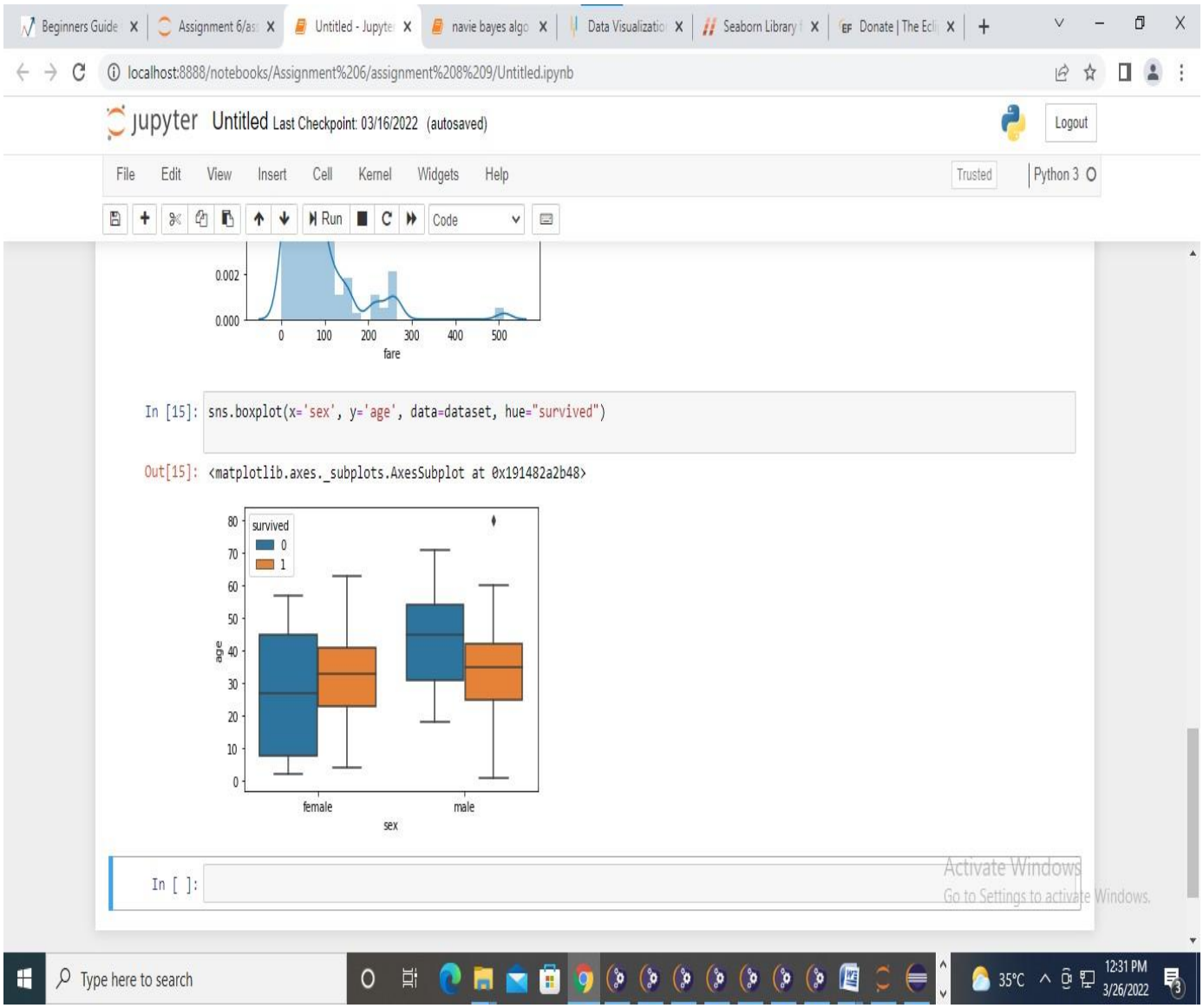
- Load Libraries and dataset



	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False	B	Southampton	yes	True
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False	NaN	Southampton	no	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True	NaN	Queenstown	no	True

Implementation of Code

Box plot for distribution of age with respect to gender with information about survived or not can be get
By: sns.boxplot(x='sex', y='age',hue='survived')



Conclusion

Seaborn is an advanced data visualization library built on top of Matplotlib library. We looked at how we can draw distributional and categorical plots using Seaborn library.

Viva Questions:

1. What is Data Visualization?

2. What is an outlier? How would you address outliers?

3. What are different mithods for data visualization?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group A

Assignment No 10

Title of the Assignment: Data Visualization I

Download the iris flower dataset or any other dataset into a dataframe. Scan the dataset and give the inference as:

1. List down the features and their types available in the dataset.
2. Create a histogram for each feature in the dataset to illustrate the feature distribution
3. Create a box polt for each feature in dataset.
4. Compare distribution and identify outliers.

Objective of the Assignment: Able to perform the data visualization and outlier removal methods using Python on any open source dataset.

Prerequisite:

Basic of Python Programming

Concept of Data Preprocessing, Data Formatting , Data Normalization and Data Cleaning.

Outcome: Implement data visualization techniques on Iris flower dataset.

CO Relevance: CO5

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

Introduction to Data Visualization

Description of dataset

Implementation of Code.

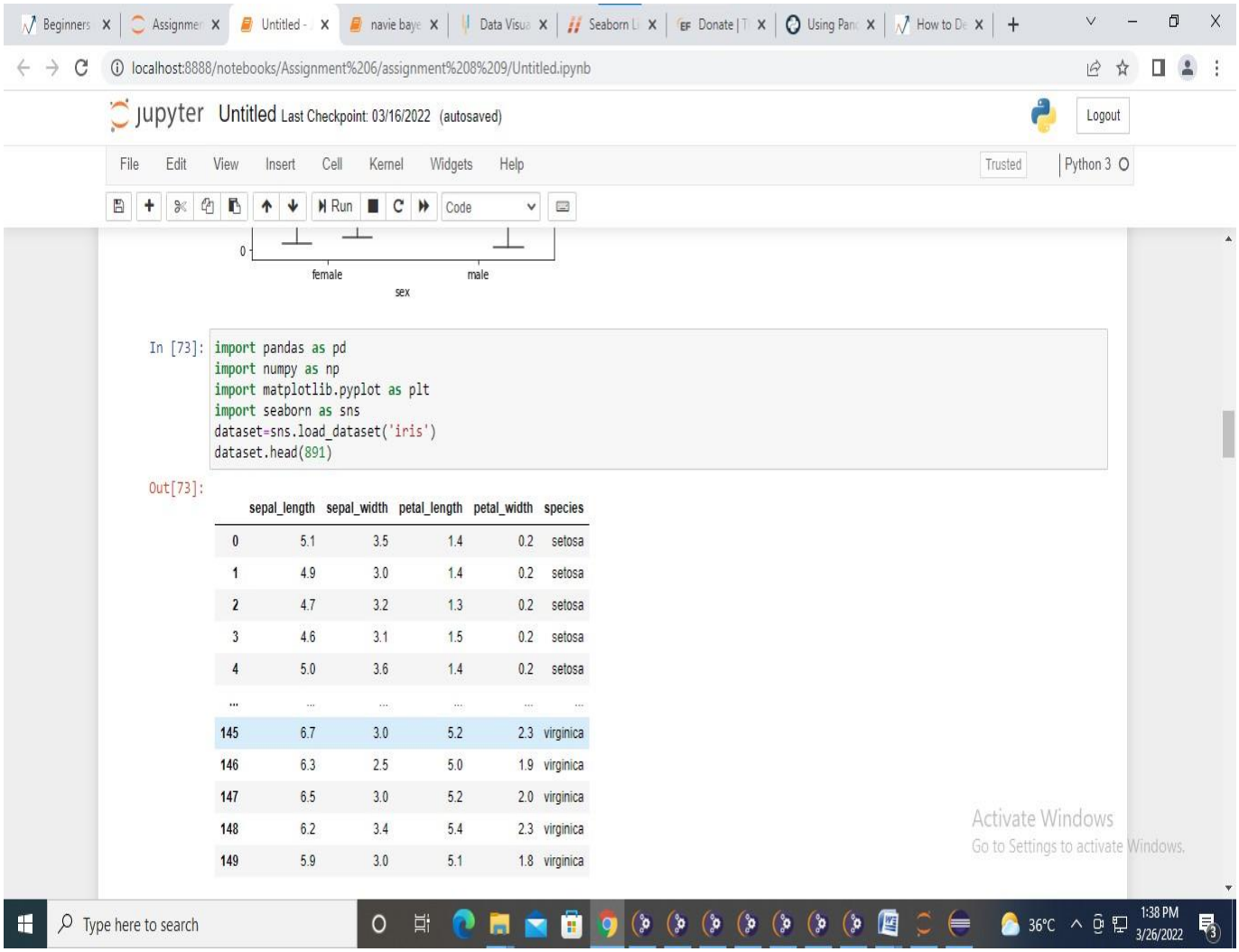
Data Visualization:

Data visualization is a field in data analysis that deals with visual representation of data. It graphically plots data and is an effective way to communicate inferences from data.

Using data visualization, we can get a visual summary of our data. With pictures, maps and graphs, the human mind has an easier time processing and understanding any given data. Data visualization plays a significant role in the representation of both small and large data sets, but it is especially useful when we have large data sets, in which it is impossible to see all of our data, let alone process and understand it manually.

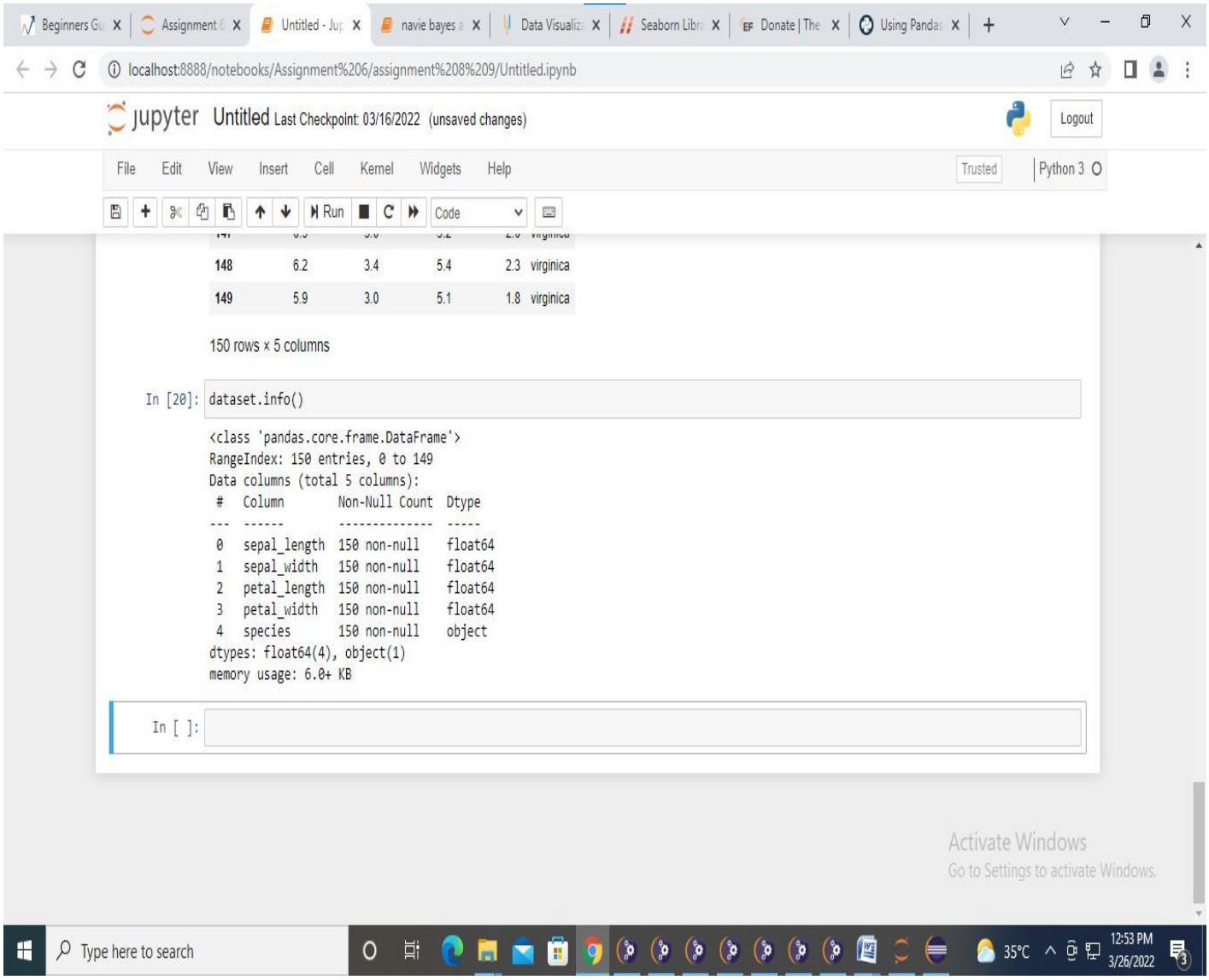
Description of dataset:

- Load Libraries and dataset

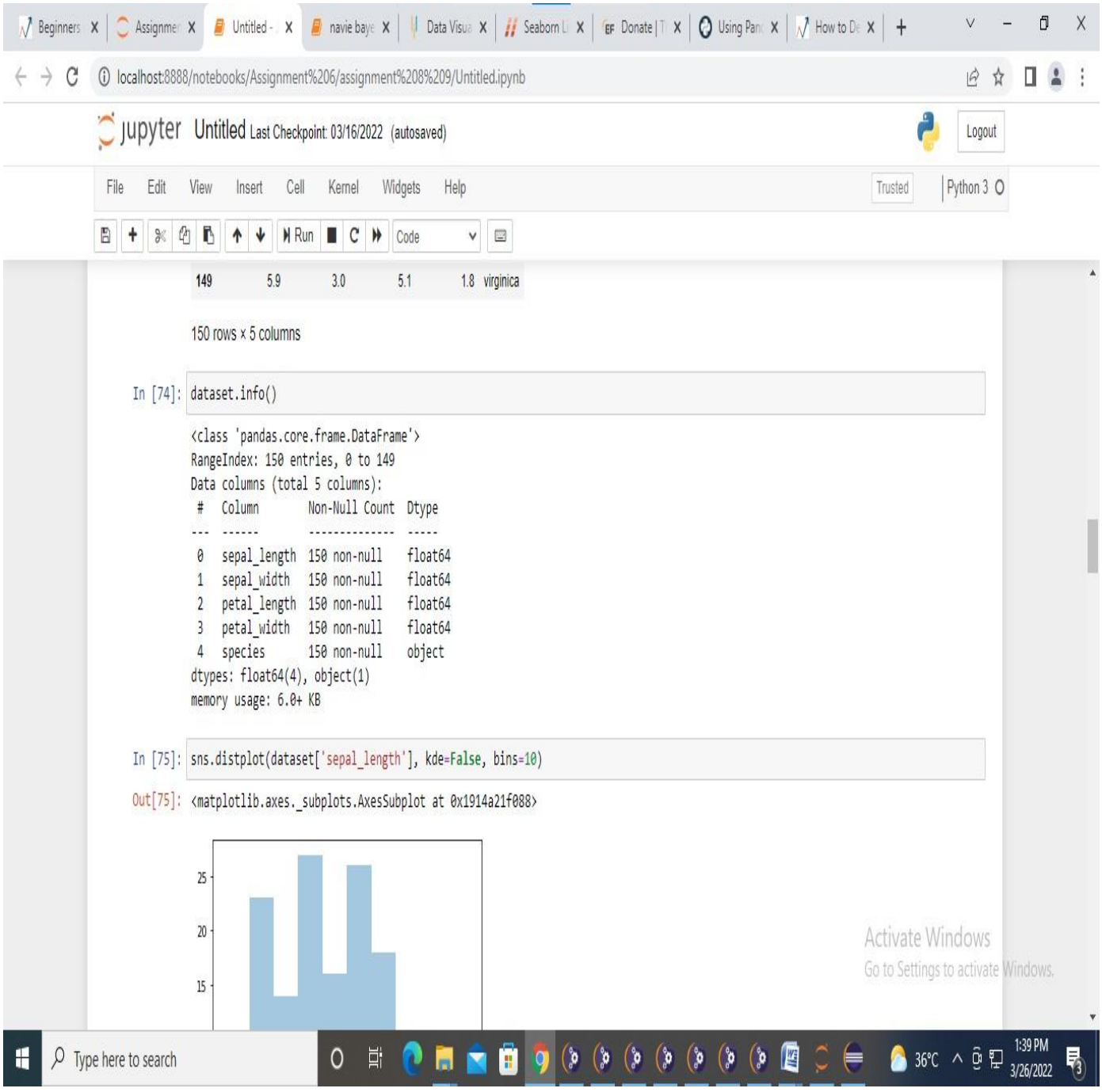


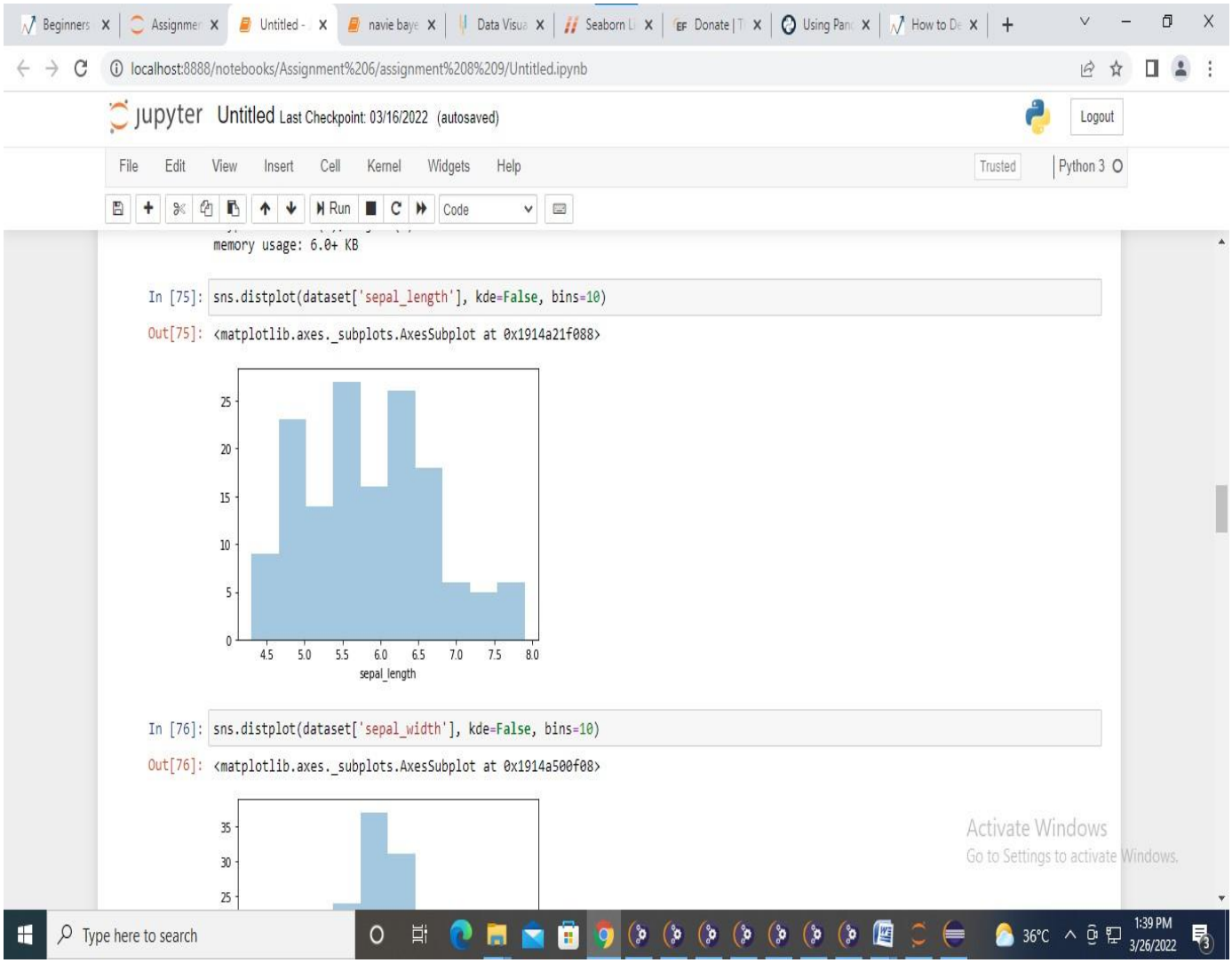
Implementation of Code

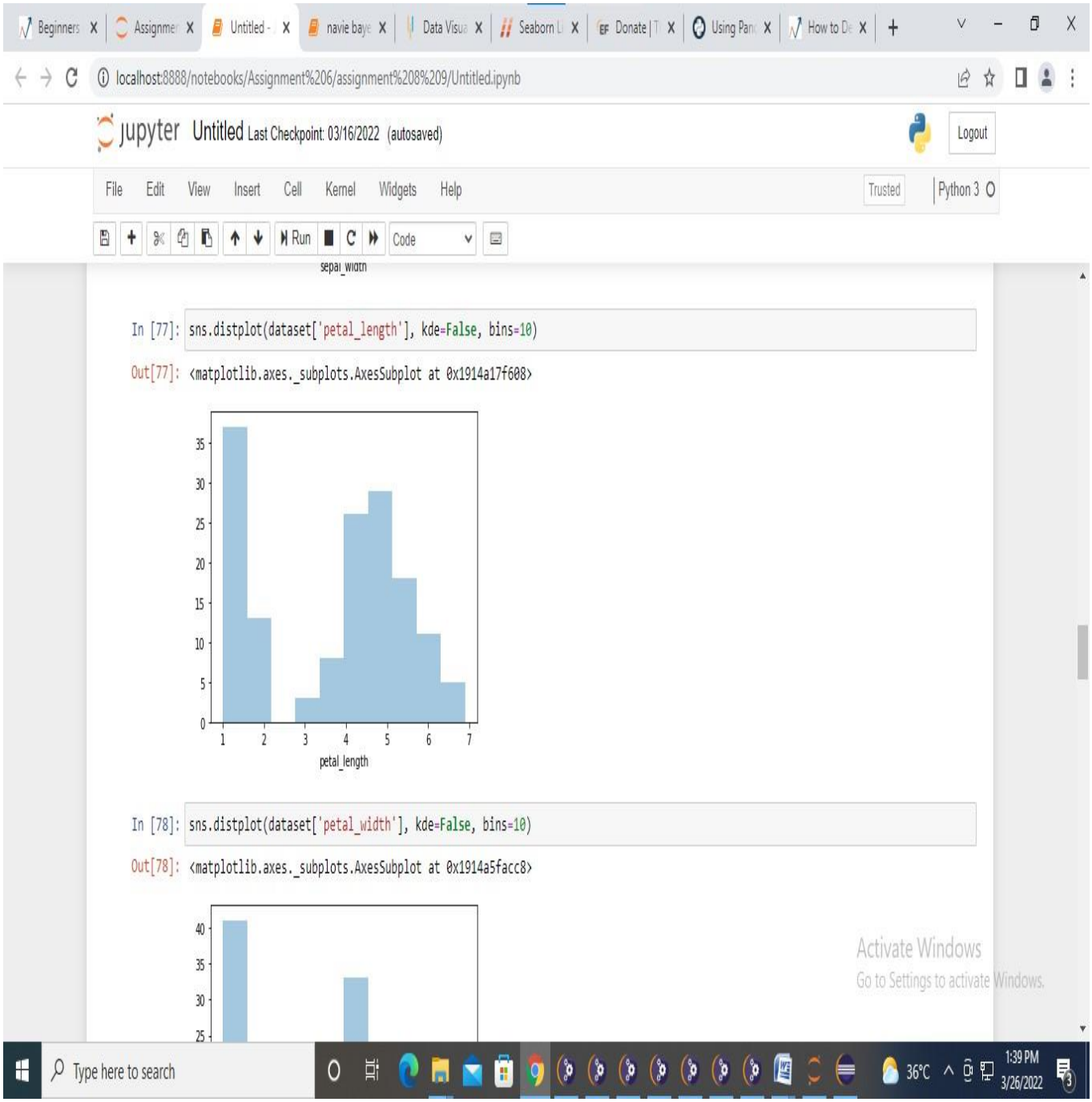
List down the features and their types available in the dataset.



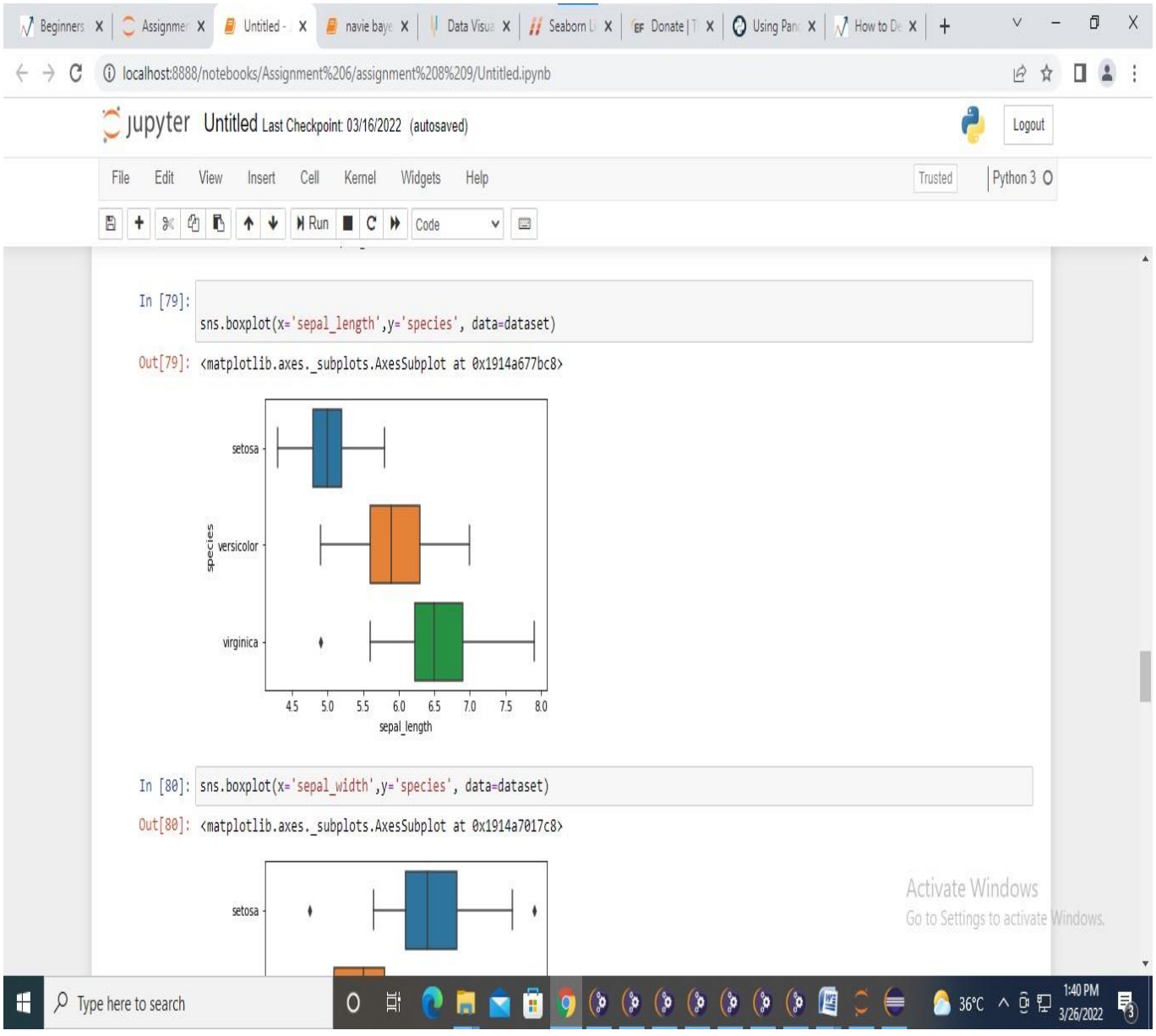
Create a histogram for each feature in the dataset to illustrate the feature distribution

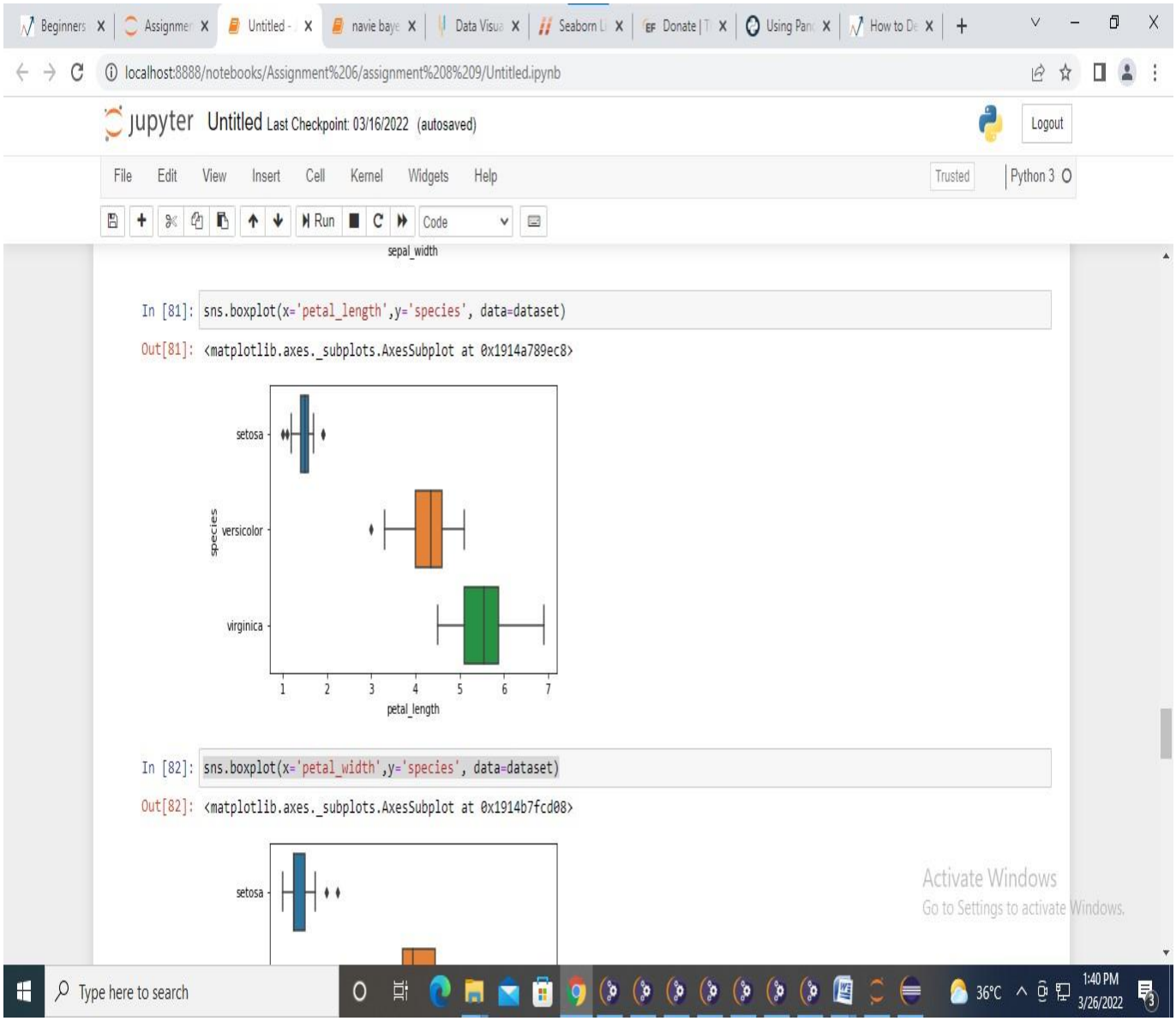



Beginners
Assignmen
Untitled -
navie baye
Data Visua
Seaborn Li
Donate | T
Using Pan
How to De
localhost:8888/notebooks/Assignment%206/assignment%208%209/Untitled.ipynb
jupyter Untitled Last Checkpoint: 03/16/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
memory usage: 6.0+ KB
In [75]: sns.distplot(dataset['sepal_length'], kde=False, bins=10)
Out[75]: <matplotlib.axes._subplots.AxesSubplot at 0x1914a21f088>
25
20
15
10
5
0
4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0
sepal_length
In [76]: sns.distplot(dataset['sepal_width'], kde=False, bins=10)
Out[76]: <matplotlib.axes._subplots.AxesSubplot at 0x1914a500f08>
35
30
25
Activate Windows
Go to Settings to activate Windows.
Type here to search
36°C
1:39 PM
3/26/2022

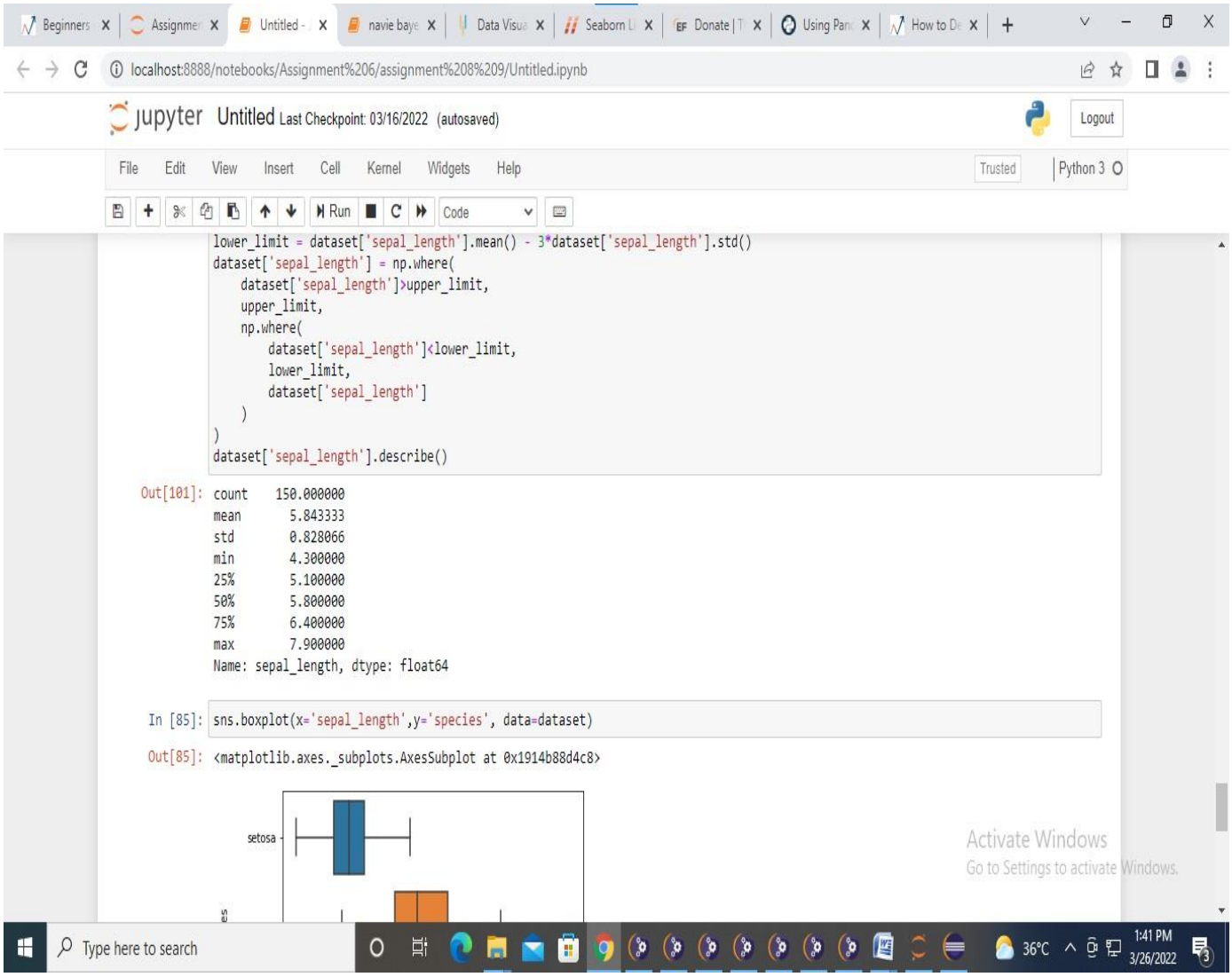

Beginners
Assignment
Untitled -
navie baye
Data Visua
Seaborn Li
Donate | T
Using Pan
How to De
localhost:8888/notebooks/Assignment%206/assignment%208%209/Untitled.ipynb
jupyter Untitled Last Checkpoint: 03/16/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
sepal_width
In [77]: sns.distplot(dataset['petal_length'], kde=False, bins=10)
Out[77]: <matplotlib.axes._subplots.AxesSubplot at 0x1914a17f608>
petal_length
In [78]: sns.distplot(dataset['petal_width'], kde=False, bins=10)
Out[78]: <matplotlib.axes._subplots.AxesSubplot at 0x1914a5facc8>
Activate Windows
Go to Settings to activate Windows.
Type here to search
36°C
1:39 PM
3/26/2022

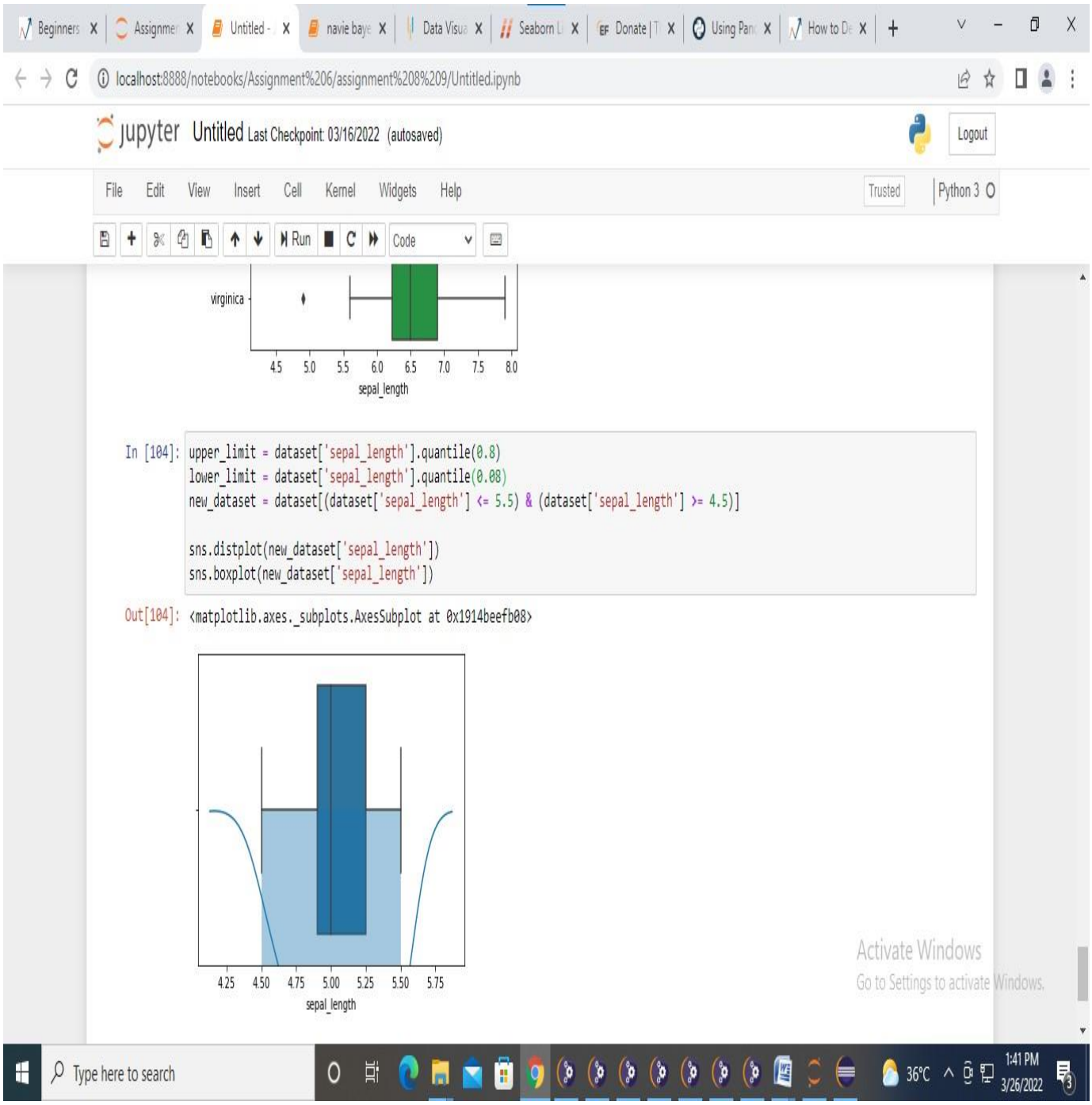
Create a box plot for each feature in dataset.




Beginners
Assignmer
Untitled -
navie baye
Data Visua
Seaborn L
Donate | T
Using Pan
How to De
localhost:8888/notebooks/Assignment%206/assignment%208%209/Untitled.ipynb
jupyter Untitled Last Checkpoint: 03/16/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
sepal_width
In [81]: sns.boxplot(x='petal_length',y='species', data=dataset)
Out[81]: <matplotlib.axes._subplots.AxesSubplot at 0x1914a789ec8>
setosa
versicolor
virginica
species
1 2 3 4 5 6 7
petal_length
In [82]: sns.boxplot(x='petal_width',y='species', data=dataset)
Out[82]: <matplotlib.axes._subplots.AxesSubplot at 0x1914b7fcd08>
setosa
Activate Windows
Go to Settings to activate Windows.
Type here to search
36°C
1:40 PM
3/26/2022

Compare distribution and identify outliers.





Conclusion

Understand the various concepts of data visualization. Performed all the tasks as listed above with extra practice and learn how to find outliers with its elimination method.

Viva Question

1. What is Data Visualization?

2. What is an outlier? How would you address outliers?

3. What are different mithods for data visualization?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group B

Assignment No 11

Title of the Assignment:

Write a code in JAVA for a simple Word Count application that counts the number of occurrences of each word in a given input set using the Hadoop MapReduce framework on local-standalone set-up.

Objective of the Assignment: Able to perform the program on Haddop MapReduce method.

Prerequisite:

Basic of Java Programming
Concept of MapReduce

Outcome: Use cutting edge tools and technologies to analyze Big Data

CO Relevance: CO6

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

Introduction to MapReduce Concept

Implementation of Code.

What is MapReducer?

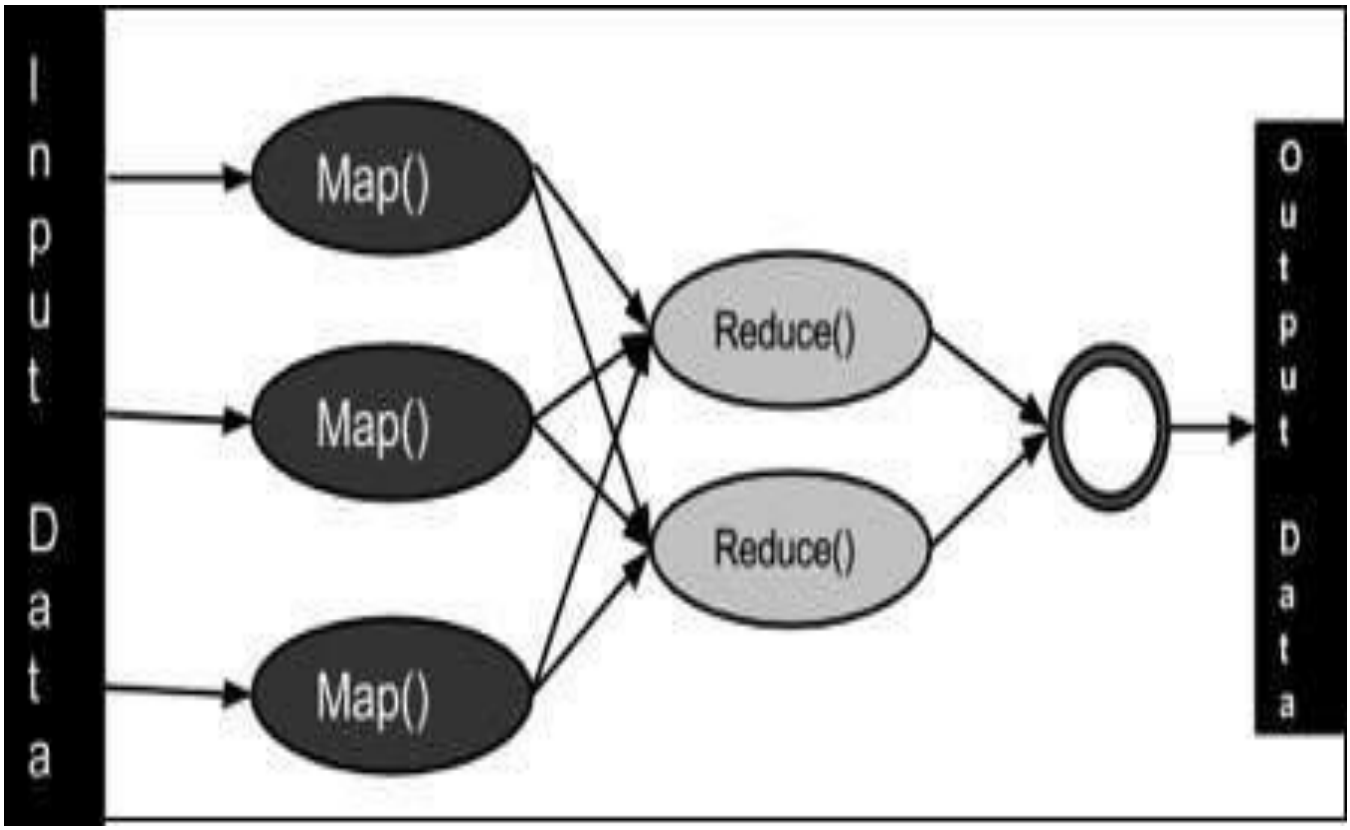
MapReduce is a processing technique and a program model for distributed computing based on java. The MapReduce algorithm contains two important tasks, namely Map and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs).

Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce task is always performed after the map job. The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers. Decomposing a data processing application into *mappers* and *reducers* is sometimes nontrivial. But, once we write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of

machines in a cluster is merely a configuration change. This simple scalability is what has attracted many programmers to use the MapReduce model.

The Algorithm

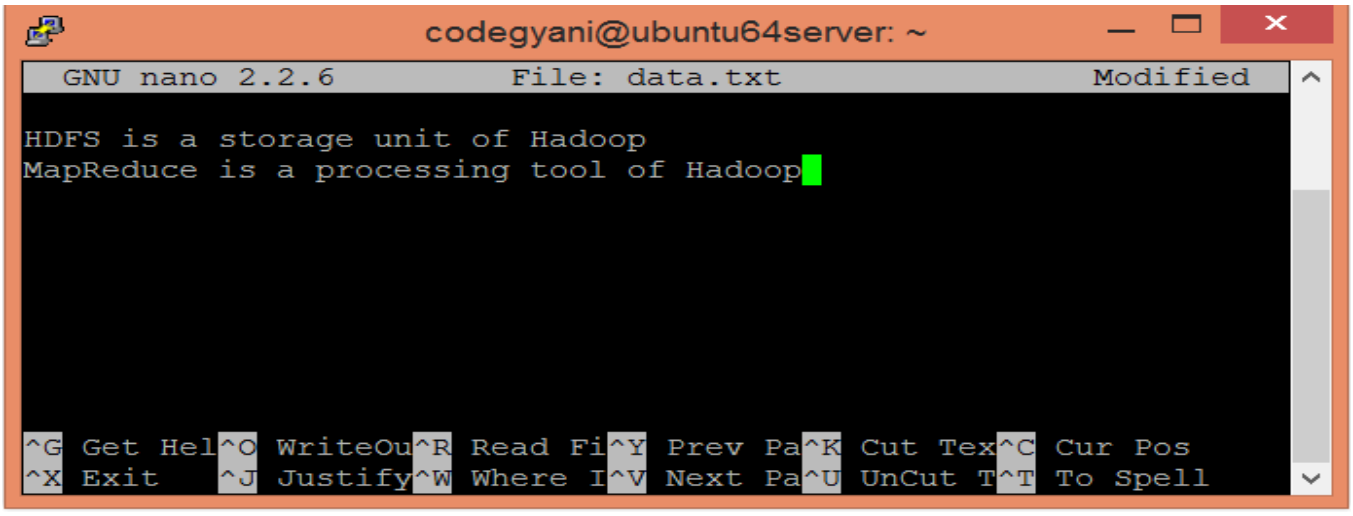
- Generally MapReduce paradigm is based on sending the computer to where the data resides!
- MapReduce program executes in three stages, namely map stage, shuffle stage, and reduce stage.
    - **Map stage** − The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.
    - **Reduce stage** − This stage is the combination of the **Shuffle** stage and the **Reduce** stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.
- During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster.
- The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes.
- Most of the computing takes place on nodes with data on local disks that reduces the network traffic.
- After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.



Implementation of Code

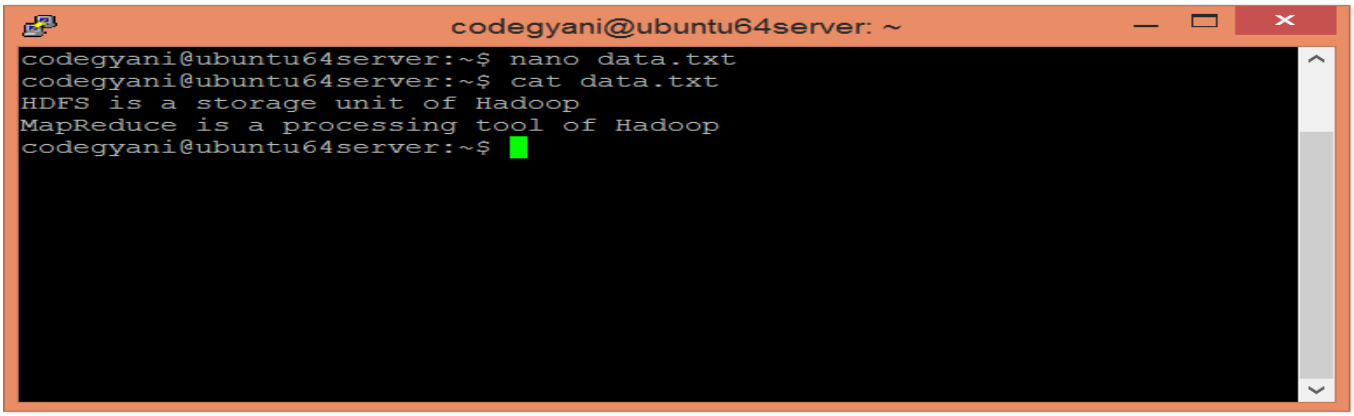
Steps to execute MapReduce word count example

- Create a text file in your local machine and write some text into it.
  $ nano data.txt



- Check the text written in the data.txt file.

  $ cat data.txt

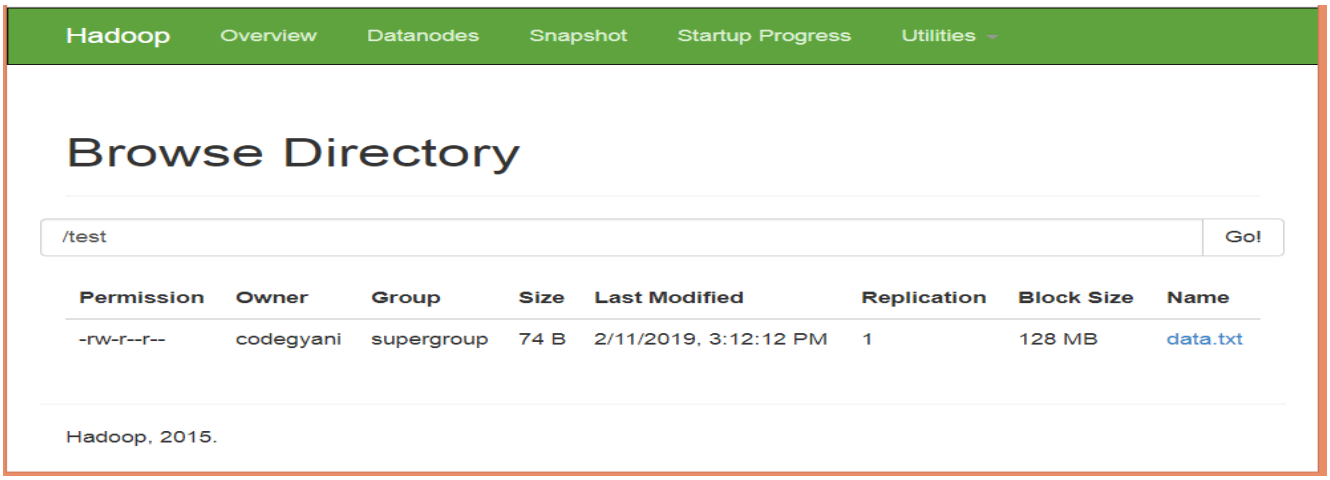


In this example, we find out the frequency of each word exists in this text file.

Create a directory in HDFS, where to kept text file.
$ hdfs dfs -mkdir /test

Upload the data.txt file on HDFS in the specific directory.

$ hdfs dfs -put /home/codegyani/data.txt /test



Write the MapReduce program using eclipse.

File: WC_Mapper.java

```
1.  package com.javatpoint;
2.
3.  import java.io.IOException;
4.  import java.util.StringTokenizer;
5.  import org.apache.hadoop.io.IntWritable;
6.  import org.apache.hadoop.io.LongWritable;
7.  import org.apache.hadoop.io.Text;
8.  import org.apache.hadoop.mapred.MapReduceBase;
9.  import org.apache.hadoop.mapred.Mapper;
10. import org.apache.hadoop.mapred.OutputCollector;
11. import org.apache.hadoop.mapred.Reporter;
12. public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWri
    table>{
13.     private final static IntWritable one = new IntWritable(1);
14.     private Text word = new Text();
15.     public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
16.         Reporter reporter) throws IOException{
17.       String line = value.toString();
18.       StringTokenizer tokenizer = new StringTokenizer(line);
19.       while (tokenizer.hasMoreTokens()){
20.         word.set(tokenizer.nextToken());
21.         output.collect(word, one);
22.       }
23.     }
24.
25. }
```

File: WC_Reducer.java

```
1.  package com.javatpoint;
2.    import java.io.IOException;
3.    import java.util.Iterator;
4.    import org.apache.hadoop.io.IntWritable;
5.    import org.apache.hadoop.io.Text;
```

```
6.  import org.apache.hadoop.mapred.MapReduceBase;
7.  import org.apache.hadoop.mapred.OutputCollector;
8.  import org.apache.hadoop.mapred.Reducer;
9.  import org.apache.hadoop.mapred.Reporter;
10.
11. public class WC_Reducer extends MapReduceBase implements Reducer<Text,IntWritable,Text,In
    tWritable> {
12. public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable> out
    put,
13.  Reporter reporter) throws IOException {
14. int sum=0;
15. while (values.hasNext()) {
16. sum+=values.next().get();
17. }
18. output.collect(key,new IntWritable(sum));
19. }
20. }
```

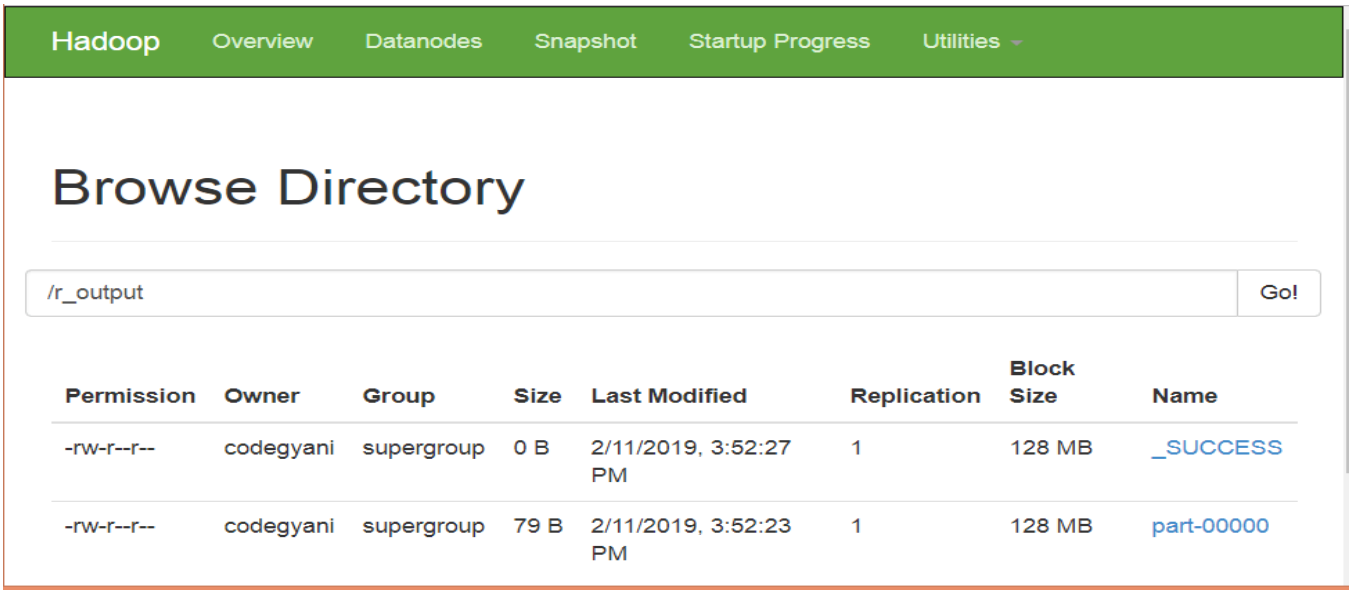
File: WC_Runner.java

```
1.  package com.javatpoint;
2.
3.  import java.io.IOException;
4.  import org.apache.hadoop.fs.Path;
5.  import org.apache.hadoop.io.IntWritable;
6.  import org.apache.hadoop.io.Text;
7.  import org.apache.hadoop.mapred.FileInputFormat;
8.  import org.apache.hadoop.mapred.FileOutputFormat;
9.  import org.apache.hadoop.mapred.JobClient;
10. import org.apache.hadoop.mapred.JobConf;
11. import org.apache.hadoop.mapred.TextInputFormat;
12. import org.apache.hadoop.mapred.TextOutputFormat;
13. public class WC_Runner {
14.  public static void main(String[] args) throws IOException{
15.  JobConf conf = new JobConf(WC_Runner.class);
16.  conf.setJobName("WordCount");
```

```
17.             conf.setOutputKeyClass(Text.class);
18.             conf.setOutputValueClass(IntWritable.class);
19.             conf.setMapperClass(WC_Mapper.class);
20.             conf.setCombinerClass(WC_Reducer.class);
21.             conf.setReducerClass(WC_Reducer.class);
22.             conf.setInputFormat(TextInputFormat.class);
23.             conf.setOutputFormat(TextOutputFormat.class);
24.             FileInputFormat.setInputPaths(conf,new  Path(args[0]));
25.             FileOutputFormat.setOutputPath(conf,new  Path(args[1]));
26.             JobClient.runJob(conf);
27.         }
28.     }
```

Download the source code.

- Create the jar file of this program and name it **countworddemo.jar.**
- Run the jar file
  hadoop jar /home/codegyani/wordcountdemo.jar com.javatpoint.WC_Runner /test/data.txt /r_output
- The output is stored in /r_output/part-00000



- Now execute the command to see the output.
  hdfs dfs -cat /r_output/part-00000

Conclusion: Studied about MapReduce Method and applied it successfully

Viva Questions:

What do you mean by data locality?

What are the advantages of using MapReduce with Hadoop?

What do you mean by shuffling and sorting in MapReduce?

Explain the process of spilling in MapReduce?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group B

Assignment No 12

Title of the Assignment:

Locate dataset (e.g., sample_weather.txt) for working on weather data which reads the text input files and finds average for temperature, dew point and wind speed.

Objective of the Assignment: Able to perform the program on Haddop MapReduce method on any open source dataset

Prerequisite:

Basic of Java Programming

Concepts of MapReduse

Objective: To analyze and demonstrate knowledge of statistical data analysis techniques for decision-making To gain practical, hands-on experience with statistics programming languages and Big Data tools

---

Outcome: Use cutting edge tools and technologies to analyze Big Data

CO Relevance: CO6

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

Introduction to MapReduce Concept

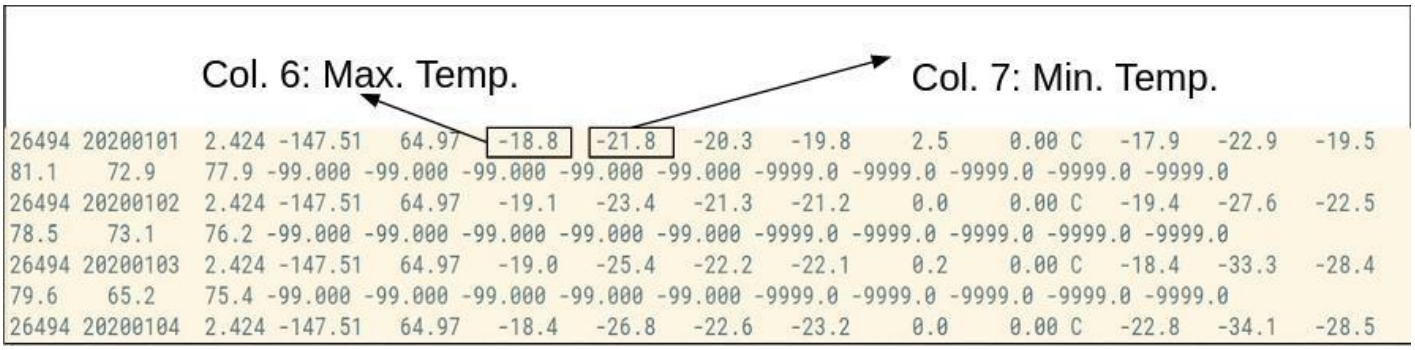
Implementation of Code.

Step 1:

We can download the dataset from Kaggle, For various cities in different years. Choose the particular year and select any one of the data text-file for analyzing.

Step 2:

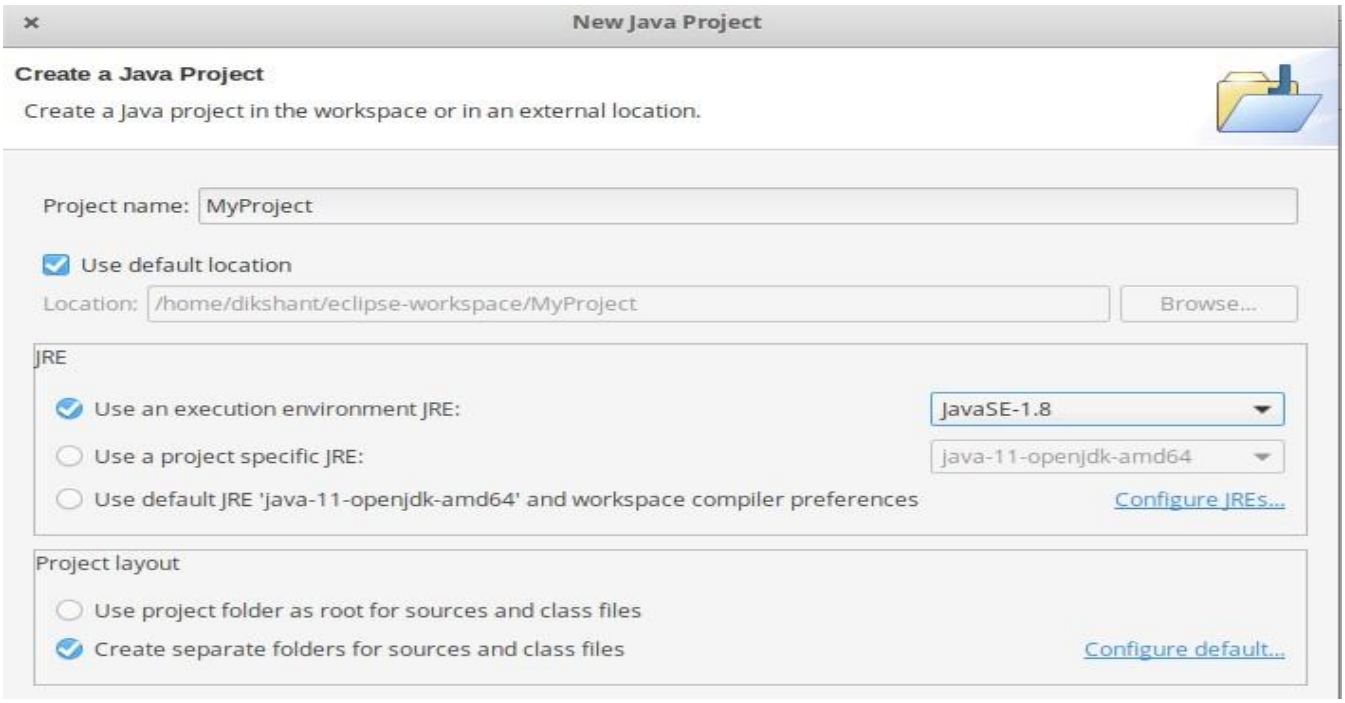
Below is the example of dataset where column 6 and column 7 is showing Maximum and Minimum temperature, respectively.



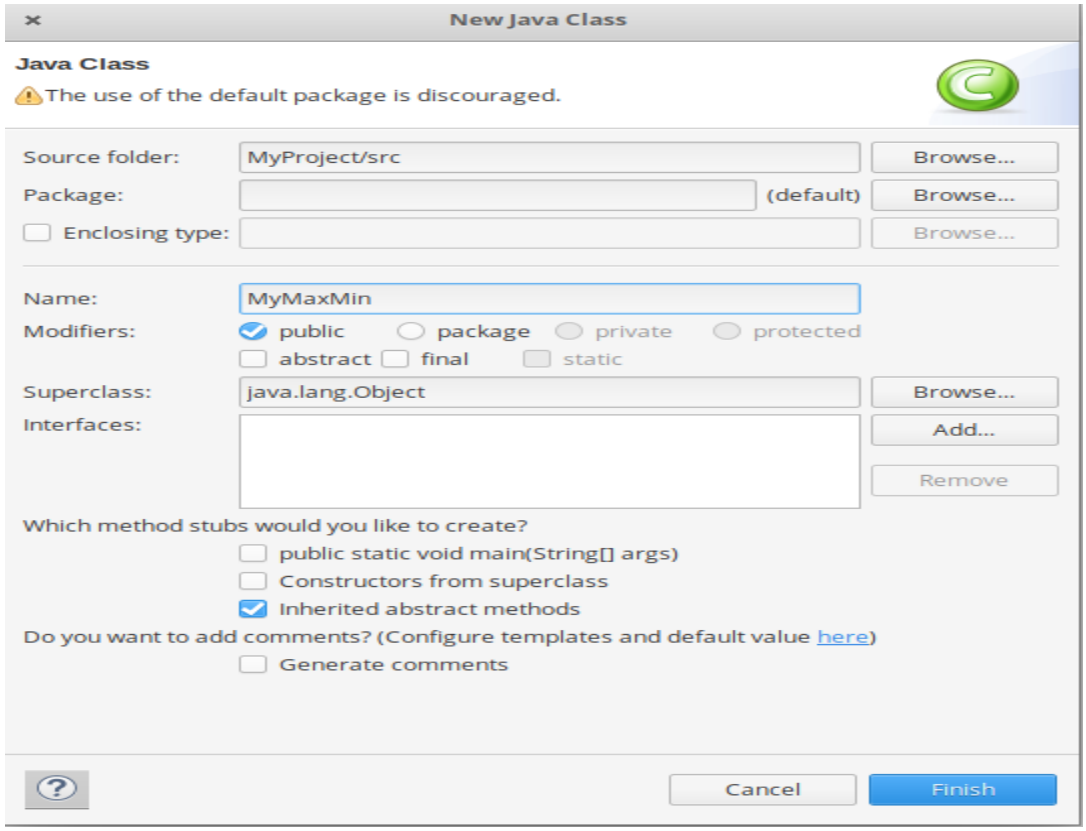
Step 3:

Make a project in Eclipse with below steps:

- First Open **Eclipse** -> then select **File -> New -> Java Project** ->Name it **MyProject** -> then select **use an execution environment** -> choose **JavaSE-1.8** then **next** -> **Finish**.



- In this Project Create Java class with name **MyMaxMin** -> then click **Finish**



Write the source code to this **MyMaxMin** java class

```
// importing Libraries

import java.io.IOException;

import java.util.Iterator;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
```

```
import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.conf.Configuration;


public class MyMaxMin {
 // Mapper

   /*MaxTemperatureMapper class is static

    * and extends Mapper abstract class

    * having four Hadoop generics type

    * LongWritable, Text, Text, Text.

   */

    public static class MaxTemperatureMapper extends

          Mapper<LongWritable, Text, Text, Text> {

          /**

      * @method map

      * This method takes the input as a text data type.

      * Now leaving the first five tokens, it takes

      * 6th token is taken as temp_max and

      * 7th token is taken as temp_min. Now

      * temp_max > 30 and temp_min < 15 are

      * passed to the reducer.

      */

 // the data in our data set with
  // this value is inconsistent data
         public static final int MISSING = 9999;
        @Override
      public void map(LongWritable arg0, Text Value, Context context)
```

```
        throws IOException, InterruptedException {
       // Convert the single row(Record) to
      // String and store it in String
      // variable name line
      String line = Value.toString();
     // Check for the empty line
      if (!(line.length() == 0)) {
    // from character 6 to 14 we have
   // the date in our dataset
    String date = line.substring(6, 14);
 // similarly we have taken the maximum
// temperature from 39 to 45 characters
     float temp_Max = Float.parseFloat(line.substring(39, 45).trim());
   // similarly we have taken the minimum
 // temperature from 47 to 53 characters
float temp_Min = Float.parseFloat(line.substring(47, 53).trim());
  // if maximum temperature is
 // greater than 30, it is a hot day
  if (temp_Max > 30.0) {
  // Hot day
    context.write(new Text("The Day is Hot Day :" + date),
    new Text(String.valueOf(temp_Max)))
 }
  // if the minimum temperature is
// less than 15, it is a cold day
if (temp_Min < 15) {
  // Cold day
```

```
context.write(new Text("The Day is Cold Day :" + date),
 new Text(String.valueOf(temp_Min)));
            }
          }
      }

}

// Reducer

 /*MaxTemperatureReducer class is static and extends Reducer abstract class having four Hadoop generics typeText, Text, Text, Text.

   */ public static class MaxTemperatureReducer extends

  Reducer<Text, Text, Text, Text> {

/**

      * @method reduce

      * This method takes the input as key and

      * list of values pair from the mapper,

      * it does aggregation based on keys and

      * produces the final context.

      */

        public void reduce(Text Key, Iterator<Text> Values, Context context)

            throws IOException, InterruptedException {


           // putting all the values in
         // temperature variable of type String
         String temperature = Values.next().toString();
         context.write(Key, new Text(temperature));

      }
 }

 /**

   * @method main
```

```
    * This method is used for setting

    * all the configuration properties.

    * It acts as a driver for map-reduce

    * code.

    */

      public static void main(String[] args) throws Exception

  // reads the default configuration of the

 // cluster from the configuration XML files

  Configuration conf = new Configuration();
// Initializing the job with the
 // default configuration of the cluster
    Job job = new Job(conf, "weather example");

// Assigning the driver class name

  job.setJarByClass(MyMaxMin.class);

 // Key type coming out of mapper
   job.setMapOutputKeyClass(Text.class);

 // value type coming out of mapper

   job.setMapOutputValueClass(Text.class);

  // Defining the mapper class name

    job.setMapperClass(MaxTemperatureMapper.class);
          // Defining the reducer class name

job.setReducerClass(MaxTemperatureReducer.class);
    // Defining input Format class which is

   // responsible to parse the dataset

 // into a key value pair

  job.setInputFormatClass(TextInputFormat.class);

  // Defining output Format class which is

 // responsible to parse the dataset
   // into a key value pair
```

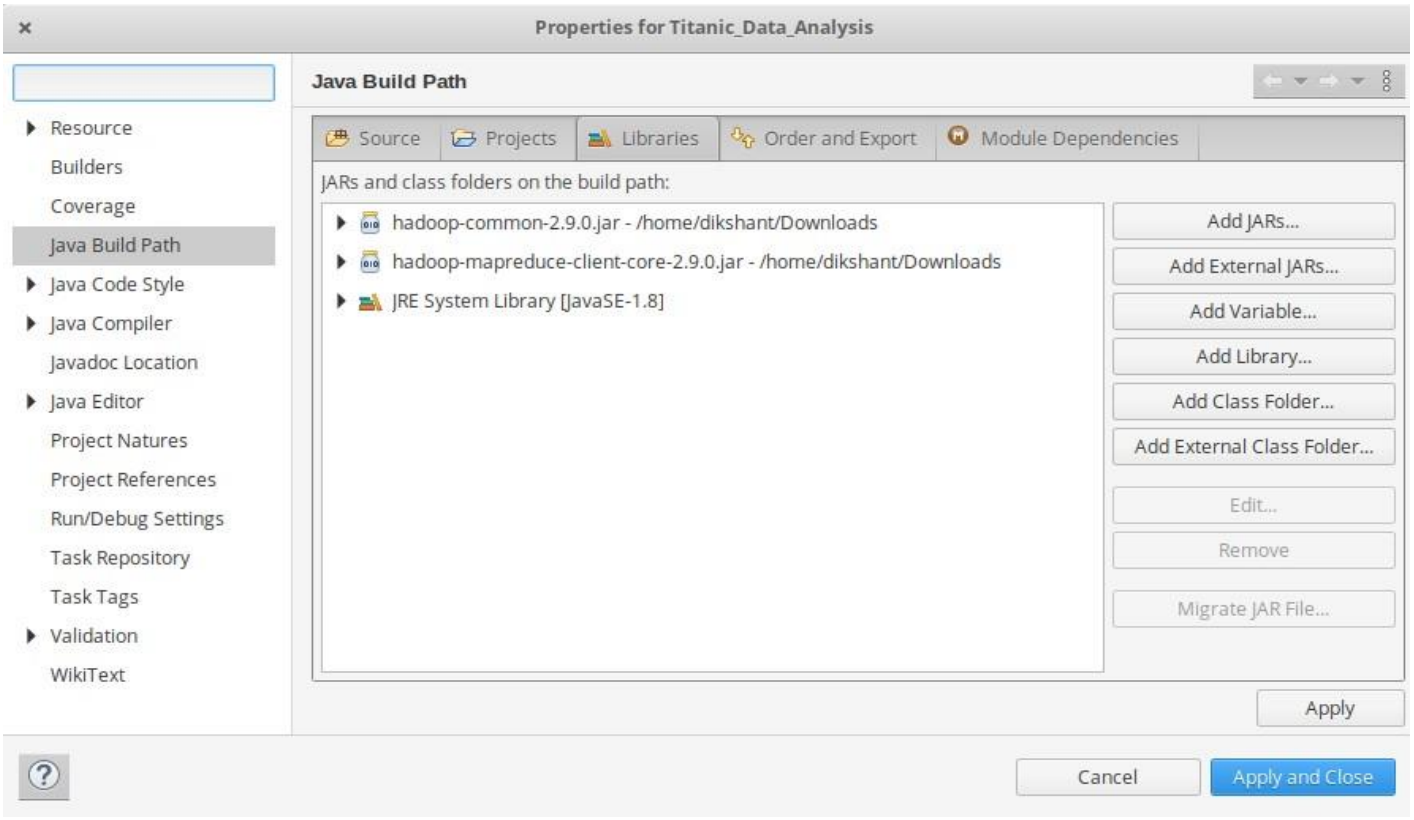
```
job.setOutputFormatClass(TextOutputFormat.class);

// setting the second argument

// as a path in a path variable

 Path OutputPath = new Path(args[1]);


// Configuring the input path
 // from the filesystem into the job
   FileInputFormat.addInputPath(job, new Path(args[0]));

  // Configuring the output path from

 // the filesystem into the job
      FileOutputFormat.setOutputPath(job, new Path(args[1]));
  // deleting the context path automatically
 // from hdfs so that we don't have
// to delete it explicitly

  OutputPath.getFileSystem(conf).delete(OutputPath);

 // exiting the job only if the

// flag value becomes false

 System.exit(job.waitForCompletion(true) ? 0 : 1);
 }
}
```

- Now we need to add external jar for the packages that we have import. Download the jar package Hadoop Common and Hadoop MapReduce Core according to your Hadoop version. You can check Hadoop Version:

Hadoop version

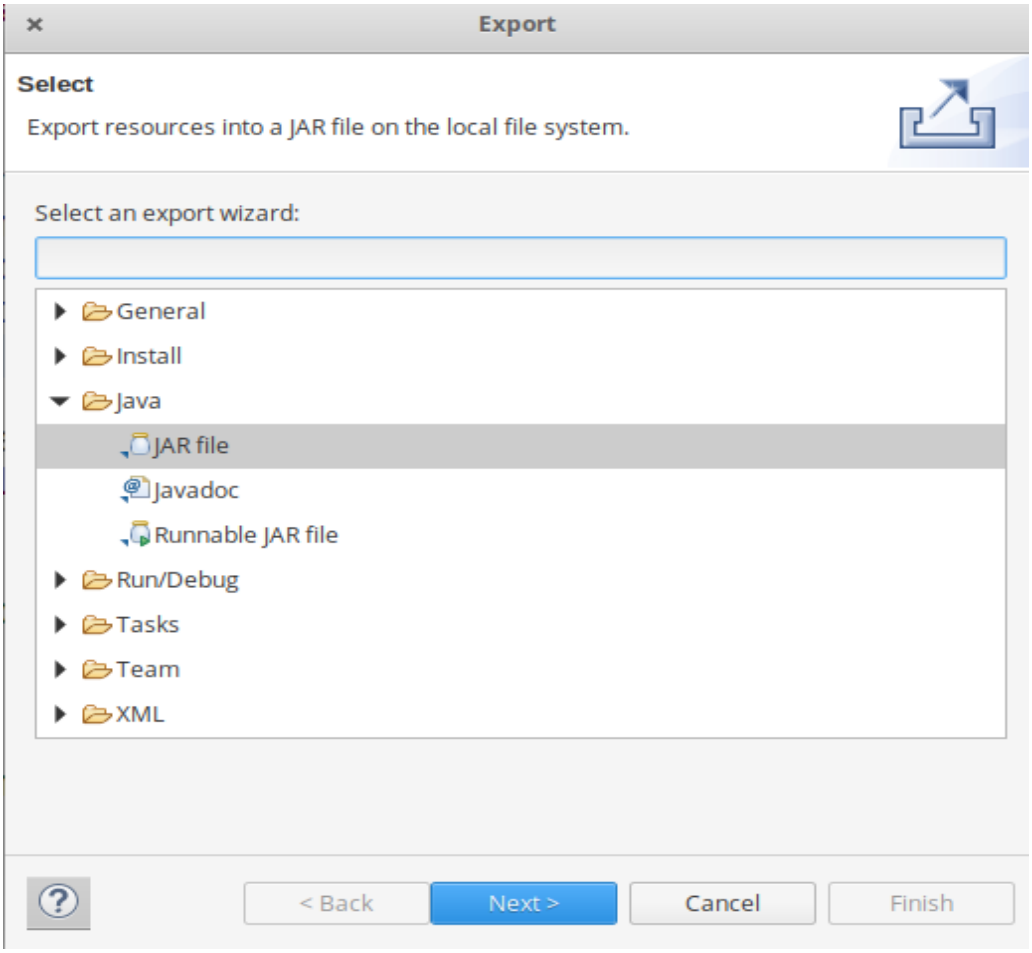
```
dikshant@dikshant-Inspiron-5567:~$ hadoop version
Hadoop 2.9.0
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r 756ebc8394e473ac25feac05fa493f6d612e6c50
Compiled by arsuresh on 2017-11-13T23:15Z
Compiled with protoc 2.5.0
From source with checksum 0a76a9a32a5257331741f8d5932f183
```

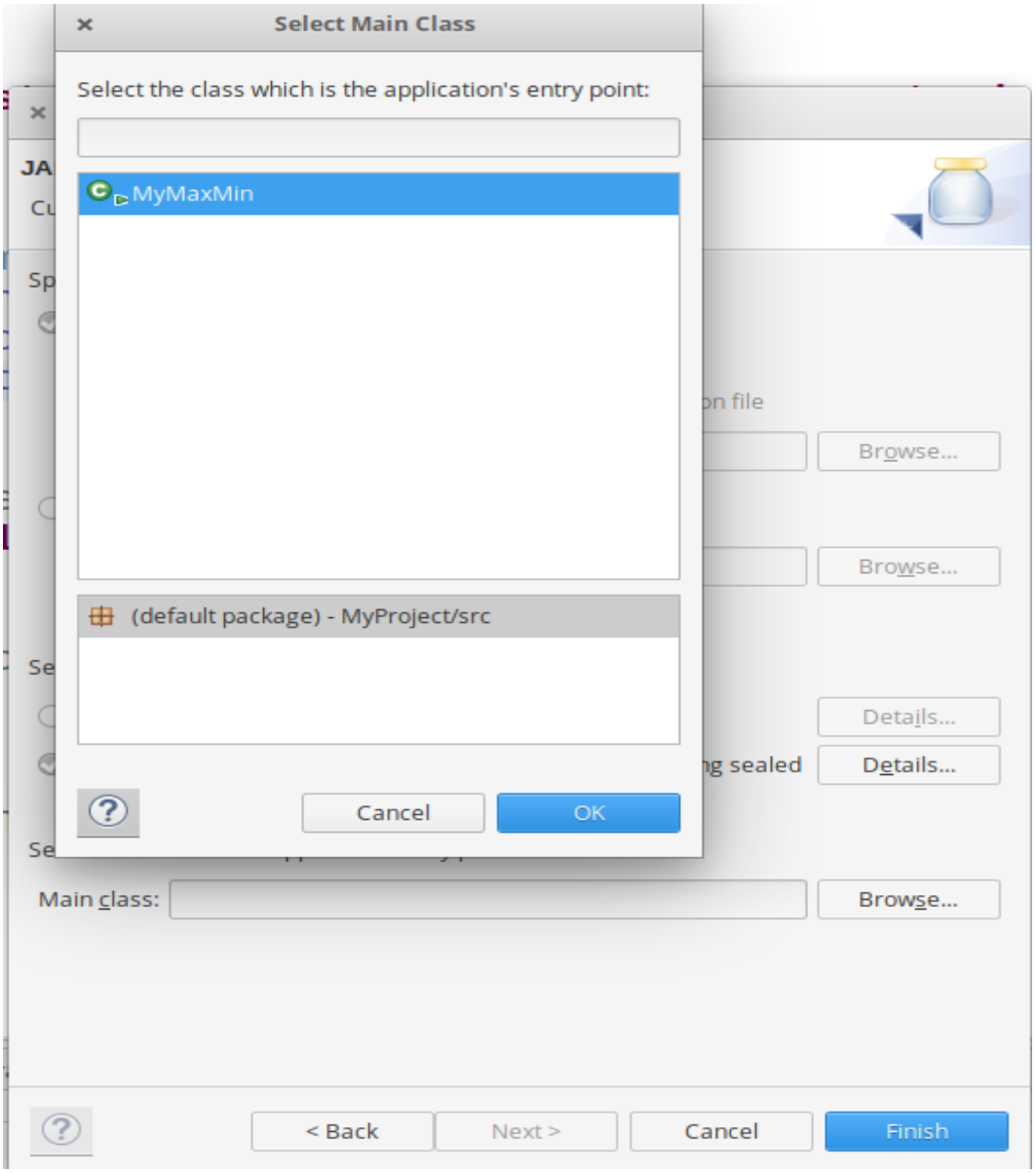
- Now we add these external jars to our **MyProject**. Right Click on **MyProject** -> then select **Build Path**-> Click on **Configure Build Path** and select **Add External jars….** and add jars from it's download location then click -> **Apply and Close**.



- Now export the project as jar file. Right-click on **MyProject** choose **Export..** and go to **Java -> JAR file** click -> **Next** and choose your export destination then click -> **Next**.

  choose Main Class as **MyMaxMin** by clicking -> **Browse** and then click -> **Finish** -> **Ok**.


Export
Select
Export resources into a JAR file on the local file system.
Select an export wizard:
General
Install
Java
JAR file
Javadoc
Runnable JAR file
Run/Debug
Tasks
Team
XML
< Back
Next >
Cancel
Finish



Step 4:

Start our Hadoop Daemons

start-dfs.sh

start-yarn.sh

Step 5:

Move your dataset to the Hadoop HDFS.
Syntax:

hdfs dfs -put /file_path /destination

In below command / shows the root directory of our HDFS.

hdfs dfs -put /home/dikshant/Downloads/CRND0103-2020-AK_Fairbanks_11_NE.txt /
Check the file sent to our HDFS.

hdfs dfs -ls /

```
dikshant@dikshant-Inspiron-5567:~$ hdfs dfs -put /home/dikshant/Downloads/CRND0103-2020-AK_Fairbanks_11_NE
.txt /
dikshant@dikshant-Inspiron-5567:~$ hdfs dfs -ls /
Found 4 items
-rw-r--r--   1 dikshant supergroup      39711 2020-07-04 09:39 /CRND0103-2020-AK_Fairbanks_11_NE.txt
drwxrwxr-x+  - dikshant supergroup          0 2020-06-23 14:23 /Hadoop_File
drwxrwxrwx   - dikshant supergroup          0 2020-06-14 21:43 /tmp
drwxr-xr-x   - dikshant supergroup          0 2020-06-14 21:43 /user
dikshant@dikshant-Inspiron-5567:~$
```

Step 6:
Now Run your Jar File with below command and produce the output in MyOutput File.
Syntax:
hadoop jar /jar_file_location /dataset_location_in_HDFS /output-file_name

Command:
hadoop jar /home/dikshant/Documents/Project.jar /CRND0103-2020-AK_Fairbanks_11_NE.txt

```
dikshant@dikshant-Inspiron-5567:~$ hadoop jar /home/dikshant/Documents/Project.jar /CRND0103-2020-AK_Fairb
anks_11_NE.txt /MyOutput
20/07/04 09:44:40 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.sessi
on-id
20/07/04 09:44:40 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
20/07/04 09:44:41 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Im
```

Step 7:
Now Move to localhost:50070/, under utilities select Browse the file system and download part-r-00000 in /MyOutput directory to see result.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	dikshant	supergroup	38.78 KB	Jul 04 09:39	1	122.07 MB	CRND0103-2020-AK_Fairbanks_11_NE.txt
drwxrwxr-x+	dikshant	supergroup	0 B	Jun 23 14:23	0	0 B	Hadoop_File
drwxr-xr-x	dikshant	supergroup	0 B	Jul 04 09:44	0	0 B	MyOutput
drwxrwxrwx	dikshant	supergroup	0 B	Jun 14 21:43	0	0 B	tmp
drwxr-xr-x	dikshant	supergroup	0 B	Jun 14 21:43	0	0 B	user

/MyOutput Go!

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	dikshant	supergroup	0 B	Jul 04 09:44	1	122.07 MB	_SUCCESS
-rw-r--r--	dikshant	supergroup	3.85 KB	Jul 04 09:44	1	122.07 MB	part-r-00000

Step 8:

See the result in the Downloaded File.

```
1 The Day is Cold Day :20200101   -21.8
2 The Day is Cold Day :20200102   -23.4
3 The Day is Cold Day :20200103   -25.4
4 The Day is Cold Day :20200104   -26.8
5 The Day is Cold Day :20200105   -28.8
6 The Day is Cold Day :20200106   -30.0
7 The Day is Cold Day :20200107   -31.4
8 The Day is Cold Day :20200108   -33.6
9 The Day is Cold Day :20200109   -26.6
10 The Day is Cold Day :20200110   -24.3
```

In the above image, you can see the top 10 results showing the cold days. The second column is a day in yyyy/mm/dd format. For Example, 20200101 means

year = 2020

month = 01

Date = 01

Conclusion: Studied about MapReduce Method and applied it on weather dataset successfully

Viva Questions:

What is a distributed cache in MapReduce Framework?

What is a combiner and where you should use it?

Why the output of map tasks are stored (spilled) into local disc and not in HDFS?

What is the role of a MapReduce Partitioner?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	

Group B

Assignment No 13

Title of the Assignment: Write a simple program in SCALA using Apache Spark framework

Objective of the Assignment:. Able to perform basic operations of Scala

Prerequisite:

Basic of Java Programming

Concepts of MapReduse

Objective: To analyze and demonstrate knowledge of statistical data analysis techniques for decision-making To gain practical, hands-on experience with statistics programming languages and Big Data tools

---

Outcome: Use cutting edge tools and technologies to analyze Big Data

CO Relevance: CO6

PO/PSOs Relevance: PO1, PO2, PO3,PO4, PO5, PO9, PO10, PSO1, PSO2, PSO3

Contents for Theory:

Introduction to Scala And All of Its Basics

Implementation of Code.

What is Scala

code Scala is an acronym for “Scalable Language”. It is a general-purpose programming language designed for the programmers who want to write programs in a concise, elegant, and type-safe way. Scala enables programmers to be more productive. Scala is developed as an object-oriented and functional programming language. If you write a code in Scala, you will see that the style is similar to a scripting language. Even though Scala is a new language, it has gained enough users and has a wide community support. It is one of the most user-friendly languages. Scala is pure Object-Oriented programming languageScala is an object-oriented programming language. Everything in Scala is an

object and any operations you perform is a method call. Scala, allow you to add new operations to existing classes with the help of implicit classes.One of the advantages of Scala is that it makes it very easy to interact with Java

. You can also write a Java code inside Scala class. The Scala supports advanced component architectures through classes and traits.

Scala is a functional language

Scala is a programming language that has implemented major functional programming concepts. In Functional programming, every computation is treated as a mathematical function which avoids states and mutable data. The functional programming exhibits following characteristics:

- Power and flexibility
- Simplicity
- Suitable for parallel processing

Scala is not a pure functional language. Haskell is an example of a pure functional language.

Installing Scala

Scala can be installed in any Unix or windows based system. Below are the steps to install for Ubuntu (14.04) for scala version 2.11.7. I am showing the steps for installing Scala (2.11.7) with Java version 7. It is necessary to install Java before installing Scala. You can also install latest version of Scala(2.12.1) as well.

Step 0: Open the terminal

Step 1: Install Java

```
$ sudo apt-add-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java7-installer
```

If you are asked to accept Java license terms, click on "Yes" and proceed. Once finished, let us check whether Java has installed successfully or not. To check the Java version and installation, you can type:

```
$ java -version
```

Step 2: Once Java is installed, we need to install Scala

```
$ cd ~/Downloads
$  wget  http://www.scala-lang.org/files/archive/scala-2.11.7.deb
$ sudo dpkg -i scala-2.11.7.deb
$ scala –version
```

This will show you the version of Scala installed

Choosing a development environment

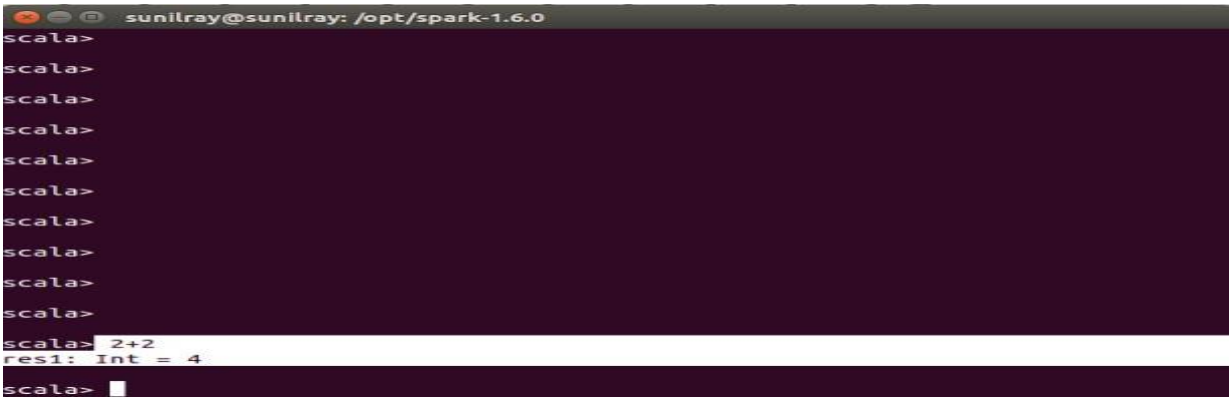
Once you have installed Scala, there are various options for choosing an environment. Here are the 3 most common options:

- Terminal / Shell based
- Notepad / Editor based
- IDE (Integrated development environment)

Choosing right environment depends on your preference and use case. I personally prefer writing a program on shell because it provides a lot of good features like suggestions for method call and you can also run your code while writing line by line.

Warming up: Running your first Scala program in Shell:

Let's write a first program which adds two numbers.



Scala Basics Terms

Object: An entity that has state and behavior is known as an object. For example: table, person, car etc.

Class: A class can be defined as a blueprint or a template for creating different objects which defines its properties and behavior.

Method: It is a behavior of a class. A class can contain one or more than one method. For example: deposit can be considered a method of bank class.

Closure: Closure is any function that closes over the environment in which it's defined. A closure returns value depends on the value of one or more variables which is declared outside this closure.

Traits: Traits are used to define object types by specifying the signature of the supported methods. It is like interface in java.

Things to note about Scala

- It is case sensitive
- If you are writing a program in Scala, you should save this program using ".scala"
- Scala execution starts from main() methods
- Any identifier name cannot begin with numbers. For example, variable name "123salary" is invalid.
- You can not use Scala reserved keywords for variable declarations or constant or any identifiers.

Variable declaration in Scala

In Scala, you can declare a variable using ‘var’ or ‘val’ keyword. The decision is based on whether it is a constant or a variable. If you use ‘var’ keyword, you define a variable as mutable variable. On the other hand, if you use ‘val’, you define it as immutable. Let’s first declare a variable using “var” and then using “val”.

8.1 Declare using var

```
var Var1 : String = "Ankit"
```

In the above Scala statement, you declare a mutable variable called “Var1” which takes a string value. You can also write the above statement without specifying the type of variable. Scala will automatically identify it. For example:

```
var Var1 = "Gupta"
```

8.2 Declare using val

```
val Var2 : String = "Ankit"
```

In the above Scala statement, we have declared an immutable variable “Var2” which takes a string “Ankit”. Try it for without specifying the type of variable. If you want to read about mutable and immutable please refer this link.

Operations on variables

You can perform various operations on variables. There are various kinds of operators defined in Scala. For example: Arithmetic Operators, Relational Operators, Logical Operators, Bitwise Operators, Assignment Operators.

Lets see “+” , “==” operators on two variables ‘Var4’, “Var5”. But, before that, let us first assign values to “Var4” and “Var5”.

```
scala> var Var4 = 2
Output: Var4: Int = 2
scala> var Var5 = 3
Output: Var5: Int = 3
```

Now, let us apply some operations using operators in Scala.

Apply '+' operator
Var4+Var5

Output:

res1: Int = 5

Apply "==" operator

Var4==Var5

Output:

res2: Boolean = false

If you want to know complete list of operators in Scala refer this link:

10. The if-else expression in Scala

In Scala, if-else expression is used for conditional statements. You can write one or more conditions inside "if". Let's declare a variable called "Var3" with a value 1 and then compare "Var3" using if-else expression.

```
var Var3 =1
if (Var3 ==1){
println("True")}else{
println("False")}
```

Output: True

In the above snippet, the condition evaluates to True and hence True will be printed in the output. **Iteration in Scala**

Like most languages, Scala also has a FOR-loop which is the most widely used method for iteration. It has a simple syntax too.

```
for( a <- 1 to 10){
 println( "Value of a: " + a );
```

}

Output:

Value of a: 1

Value of a: 2

Value of a: 3

Value of a: 4

Value of a: 5

Value of a: 6

Value of a: 7

Value of a: 8

Value of a: 9

Value of a: 10

Scala also supports “while” and “do while” loops.

Declare a simple function in Scala and call it by passing value

You can define a function in Scala using “def” keyword. Let’s define a function called “mul2” which will take a number and multiply it by 10. You need to define the return type of function, if a function not returning any value you should use the “Unit” keyword.

In the below example, the function returns an integer value. Let’s define the function “mul2”:

def mul2(m: Int): Int = m * 10

Output: mul2: (m: Int)Int

Now let’s pass a value 2 into mul2

mul2(2)

Output:

res9: Int = 20

Few Data Structures in Scala

- Arrays
- Lists

- Sets
- Tuple
- Maps
- Option

Arrays in Scala

In Scala, an array is a collection of similar elements. It can contain duplicates. Arrays are also immutable in nature. Further, you can access elements of an array using an index:

Declaring Array in Scala

To declare any array in Scala, you can define it either using a new keyword or you can directly assign some values to an array.

Declare an array by assigning it some values

```
var name = Array("Faizan","Swati","Kavya", "Deepak", "Deepak")
```

Output:

```
name: Array[String] = Array(Faizan, Swati, Kavya, Deepak, Deepak)
```

In the above program, we have defined an array called name with 5 string values.

Declaring an array using "new" keywords

The following is the syntax for declaring an array variable using a new keyword.

```
var name:Array[String] = new Array[String](3)
```

or

```
var name = new Array[String](3)
```

Output:

```
name: Array[String] = Array(null, null, null)
```

Here you have declared an array of Strings called "name" that can hold up to three elements. You can also assign values to "name" by using an index.

```
scala> name(0) = "jal"
scala> name(1) = "Faizy"
scala> name(2) = "Expert in deep learning"
```

Let's print contents of "name" array.

scala> name

res3: Array[String] = Array(jal, Faizy, Expert in deep learning)

Accessing an array

You can access the element of an array by index. Lets access the first element of array "name". By giving index 0. Index in Scala starts from 0.

name(0)

Output:

res11: String = jal

List in Scala

Lists are one of the most versatile data structure in Scala. Lists contain items of different types in Python, but in Scala the items all have the same type. Scala lists are immutable.

Here is a quick example to define a list and then access it.

Declaring List in Scala

You can define list simply by comma separated values inside the "List" method.

scala> val numbers = List(1, 2, 3, 4, 5, 1, 2, 3, 4, 5)

numbers: List[Int] = List(1, 2, 3, 4, 5, 1, 2, 3, 4, 5)

You can also define multi dimensional list in Scala. Lets define a two dimensional list:

val number1 = List(List(1, 0, 0), List(0, 1, 0), List(0, 0, 1))

number1: List[List[Int]] = List(List(1, 0, 0), List(0, 1, 0), List(0, 0, 1))

Accessing a list

Let's get the third element of the list "numbers" . The index should 2 because index in Scala start from 0.

scala> numbers(2)

res6: Int = 3

We have discussed two of the most used data Structures.

Writing & Running a program in Scala using an editor

Let us start with a "Hello World!" program. It is a good simple way to understand how to write, compile and run codes in Scala. No prizes for telling the outcome of this code!

```
object HelloWorld {
 def main(args: Array[String]) {
 println("Hello, world!")
 }
 }
```

As mentioned before, if you are familiar with Java, it will be easier for you to understand Scala. If you know Java, you can easily see that the structure of above "HelloWorld" program is very similar to Java program.

This program contains a method "main" (not returning any value) which takes an argument – a string array through command line. Next, it calls a predefined method called "Println" and passes the argument "Hello, world!".

You can define the main method as static in Java but in Scala, the static method is no longer available. Scala programmer can't use static methods because they use singleton objects.

Compile a Scala Program: To run any Scala program, you first need to compile it. "Scalac" is the compiler which takes source program as an argument and generates object files as output. Let's start compiling your "HelloWorld" program using the following steps:

1. For compiling it, you first need to paste this program into a text file then you need to save this program as HelloWorld.scala
2. Now you need change your working directory to the directory where your program is saved
3. After changing the directory you can compile the program by issuing the command.

scalac HelloWorld.scala

4. After compiling, you will get Helloworld.class as an output in the same directory. If you can see the file, you have successfully compiled the above program.

Running Scala Program

After compiling, you can now run the program using following command:

scala HelloWorld

You will get an output if the above command runs successfully. The program will print

"Hello, world!"

Conclusion: Studied all basics of SCAL and implement some of its concepts to write simple program

Viva Questions:

What is Scala?

What are some main features of Scala?

Write some benefits of using Scala.

Name some of the frameworks that Scala supports.

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator :	