MALAD KANDIVALI EDUCATION SOCIETY'S
NAGINDAS KHANDWALA COLLEGE OF COMMERCE,
ARTS & MANAGEMENT STUDIES & SHANTABEN NAGINDAS
KHANDWALA COLLEGE OF SCIENCE
MALAD [W], MUMBAI – 64
(AUTONOMOUS)

(Reaccredited 'A' Grade by NAAC)
(AFFILIATED TO UNIVERSITY OF MUMBAI)
(ISO 9001:2015)

CERTIFICATE

Name: Mr. kuldeep sushil patel

Roll No : 574 **Programme**: BSc CS **Semester**: VI

This is certified to be a bonafide record of practical works done by the above student in the college laboratory for the course **Data Science** (Course Code: **1866UCSPR**) for the partial fulfillment of Sixth Semester of BSc CS during the academic year 2020-2021.

The journal work is the original study work that has been duly approved in the year 2020-2021 by the undersigned.

External Examiner **Mr. Gangashankar Singh**
(Subject-In-Charge)

Date of Examination: **(College Stamp)**

Name: **Kuldeep Sushil Patel** Roll No: **574**

Subject: **Data Science** Class: **T. Y. BSc. CS**

INDEX

Sr. No.	Date	Topic	Sign
1	07-01-21	Practical of Principle Component Analysis	
2	13-01-21	Practical of K Means Clustering	
3	21-01-21	Practical of Time Series Forecasting	
4	28-01-21	Practical of Simple/Multiple Linear Regression	
5	16-02-21	Practical of Logistic Regression	
6	24-02-21	Practical of Hypothesis Testing	
7	09-03-21	Practical of Analysis of Variance	
8	16-03-21	Practical of Decision Tree	

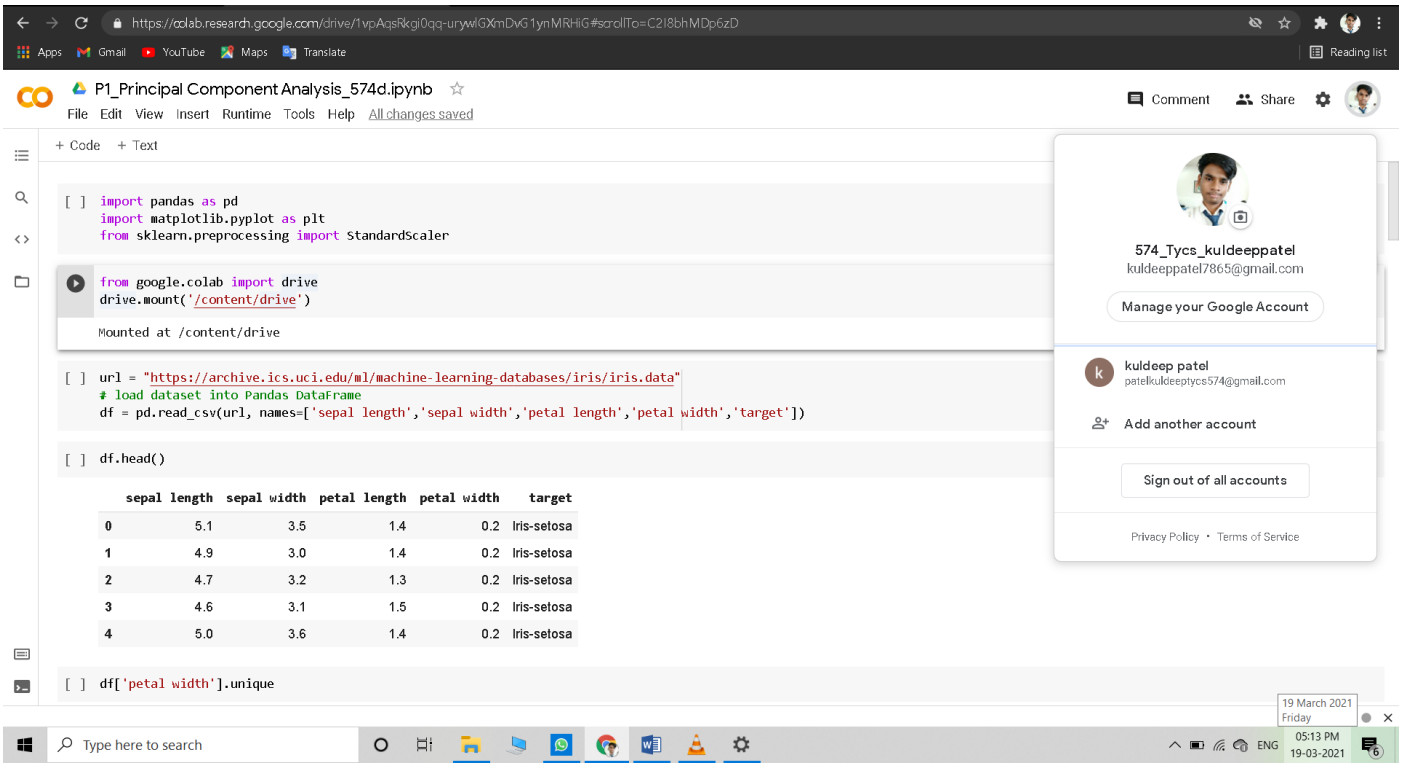
Practical – 1

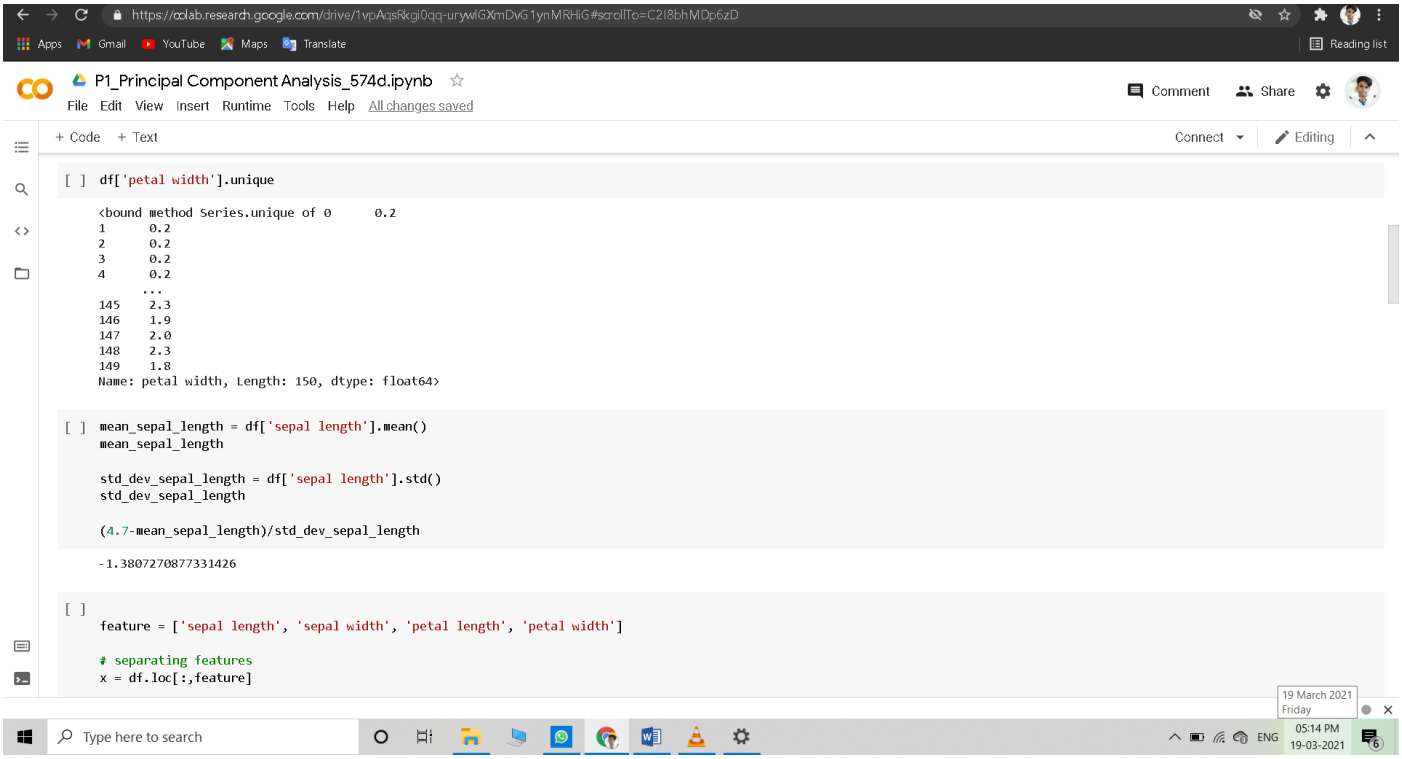
Aim: Practical of Principal component analysis.

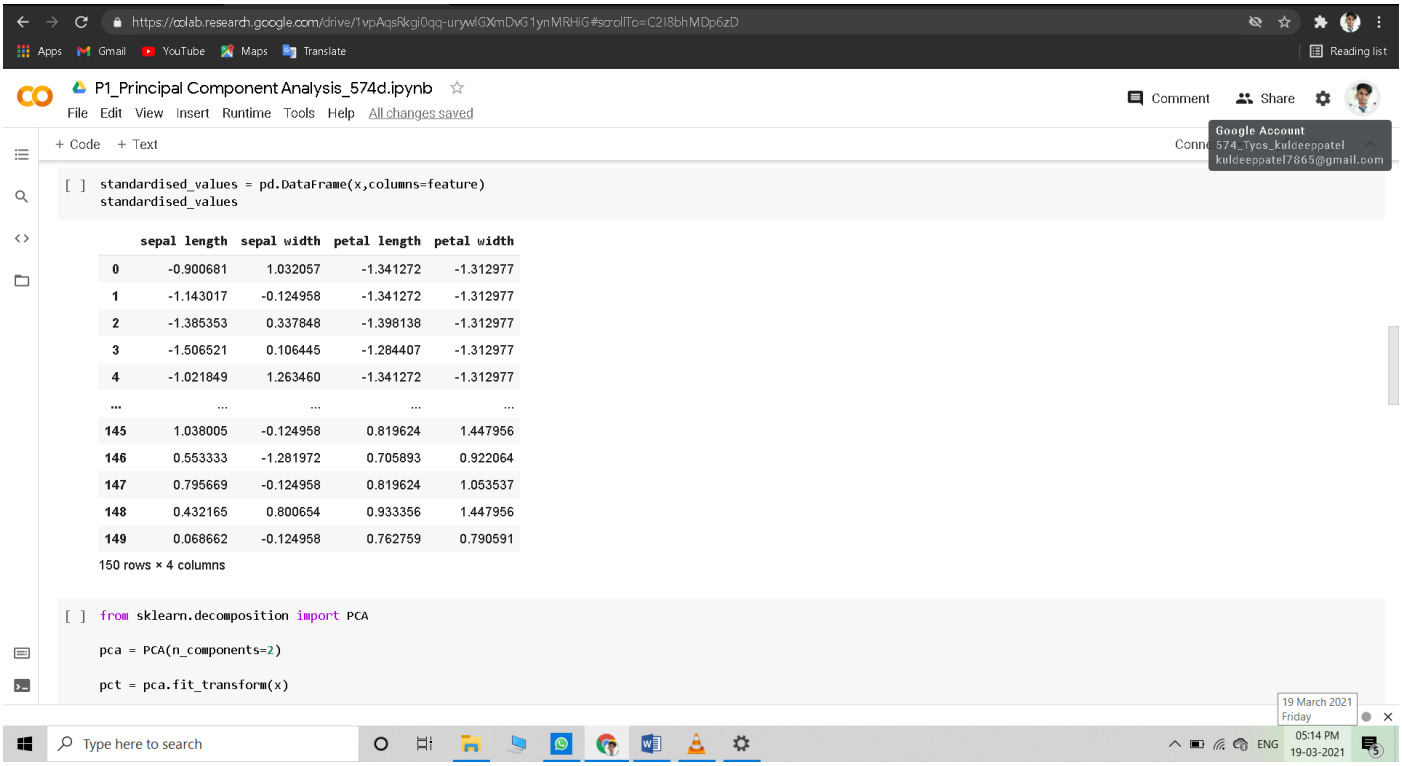
Theory:

- PCA stands for Principal Component Analysis.
- PCA is an unsupervised learning class of statistical techniques used to explain data in high dimensions using a smaller number of variables called the principal components.
- In PCA, we compute the principal component and use it to explain the data
- The principal component analysis is extremely useful for deriving an overall,linearly independent, trend for a given dataset with many variables.
- It allows you to extract important relationships out of variables that may or may not be related.

Code :




https://colab.research.google.com/drive/1vpAqsRkgi0qq-urywlGXmDvG1ynMRHiG#scrollTo=C2I8bhMDp6zD
Apps Gmail YouTube Maps Translate Reading list
P1_Principal Component Analysis_574d.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text Connect Editing
[ ] df['petal width'].unique
<bound method Series.unique of 0 0.2
1 0.2
2 0.2
3 0.2
4 0.2
...
145 2.3
146 1.9
147 2.0
148 2.3
149 1.8
Name: petal width, Length: 150, dtype: float64>
[ ] mean_sepal_length = df['sepal length'].mean()
mean_sepal_length
std_dev_sepal_length = df['sepal length'].std()
std_dev_sepal_length
(4.7-mean_sepal_length)/std_dev_sepal_length
-1.3807270877331426
[ ]
feature = ['sepal length', 'sepal width', 'petal length', 'petal width']
# separating features
x = df.loc[:,feature]
19 March 2021 Friday
Type here to search
ENG 05:14 PM 19-03-2021


https://colab.research.google.com/drive/1vpAqsRkgi0qq-urywlGXmDvG1ynMRHiG#scrollTo=C2I8bhMDp6zD
Apps Gmail YouTube Maps Translate Reading list
P1_Principal Component Analysis_574d.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com
+ Code + Text
[ ] standardised_values = pd.DataFrame(x,columns=feature)
standardised_values
sepal length sepal width petal length petal width
0 -0.900681 1.032057 -1.341272 -1.312977
1 -1.143017 -0.124958 -1.341272 -1.312977
2 -1.385353 0.337848 -1.398138 -1.312977
3 -1.506521 0.106445 -1.284407 -1.312977
4 -1.021849 1.263460 -1.341272 -1.312977
... ...
145 1.038005 -0.124958 0.819624 1.447956
146 0.553333 -1.281972 0.705893 0.922064
147 0.795669 -0.124958 0.819624 1.053537
148 0.432165 0.800654 0.933356 1.447956
149 0.068662 -0.124958 0.762759 0.790591
150 rows × 4 columns
[ ] from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pct = pca.fit_transform(x)
19 March 2021 Friday
Type here to search
ENG 05:14 PM 19-03-2021

https://colab.research.google.com/drive/1vpAqsRkgi0qq-urywlGXmDvG1ynMRHiG#scrollTo=C2I8bhMDp6zD

Apps Gmail YouTube Maps Translate Reading list

P1_Principal Component Analysis_574d.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

+ Code + Text

```
[ ] finaldf.head()
```

	pc1	pc2	target
0	-2.264542	0.505704	Iris-setosa
1	-2.086426	-0.655405	Iris-setosa
2	-2.367950	-0.318477	Iris-setosa
3	-2.304197	-0.575368	Iris-setosa
4	-2.388777	0.674767	Iris-setosa

```
[ ] fig = plt.figure(figsize = (8,8))
    ax = fig.add_subplot(1,1,1)
    ax.set_xlabel('Principal Component 1', fontsize = 15)
    ax.set_ylabel('Principal Component 2', fontsize = 15)
    ax.set_title('2 component PCA', fontsize = 20)
    targets = ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
    colors = ['r', 'g', 'b']
    for target, color in zip(targets,colors):
        indicesToKeep = finaldf['target'] == target
        ax.scatter(finaldf.loc[indicesToKeep, 'pc1']
                   , finaldf.loc[indicesToKeep, 'pc2']
                   , c = color
                   , s = 50)
    ax.legend(targets)
    ax.grid()
```

2 component PCA

19 March 2021 Friday

Type here to search ENG 05:15 PM 19-03-2021

https://colab.research.google.com/drive/1vpAqsRkgi0qq-urywlGXmDvG1ynMRHiG#scrollTo=C2I8bhMDp6zD

Apps Gmail YouTube Maps Translate Reading list

P1_Principal Component Analysis_574d.ipynb

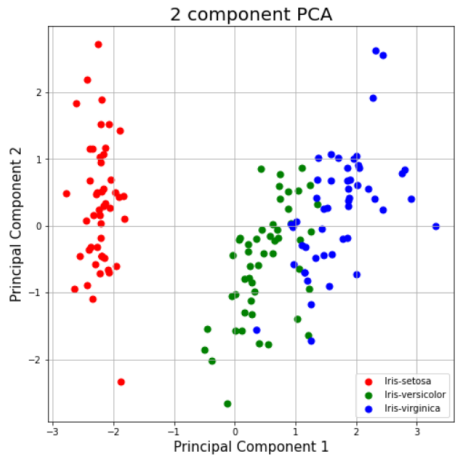
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

+ Code + Text

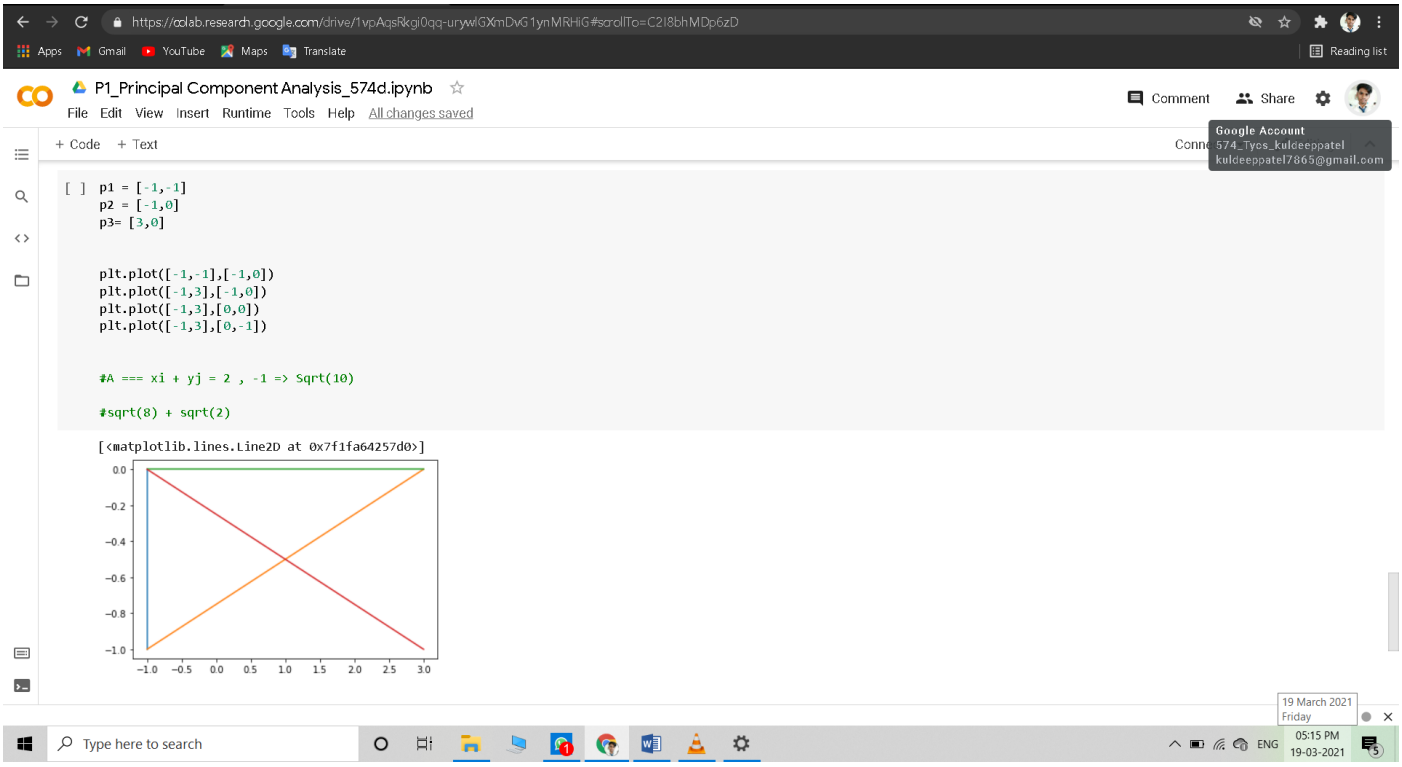
```
[ ]
```




```
[ ] pca.explained_variance_ratio_
    #95, 5% loss
```

19 March 2021 Friday

Type here to search ENG 05:15 PM 19-03-2021



Colab Link :

https://colab.research.google.com/drive/1vpAqsRkgi0qq-urywlGXmDvG1ynMRHiG?usp=sharing

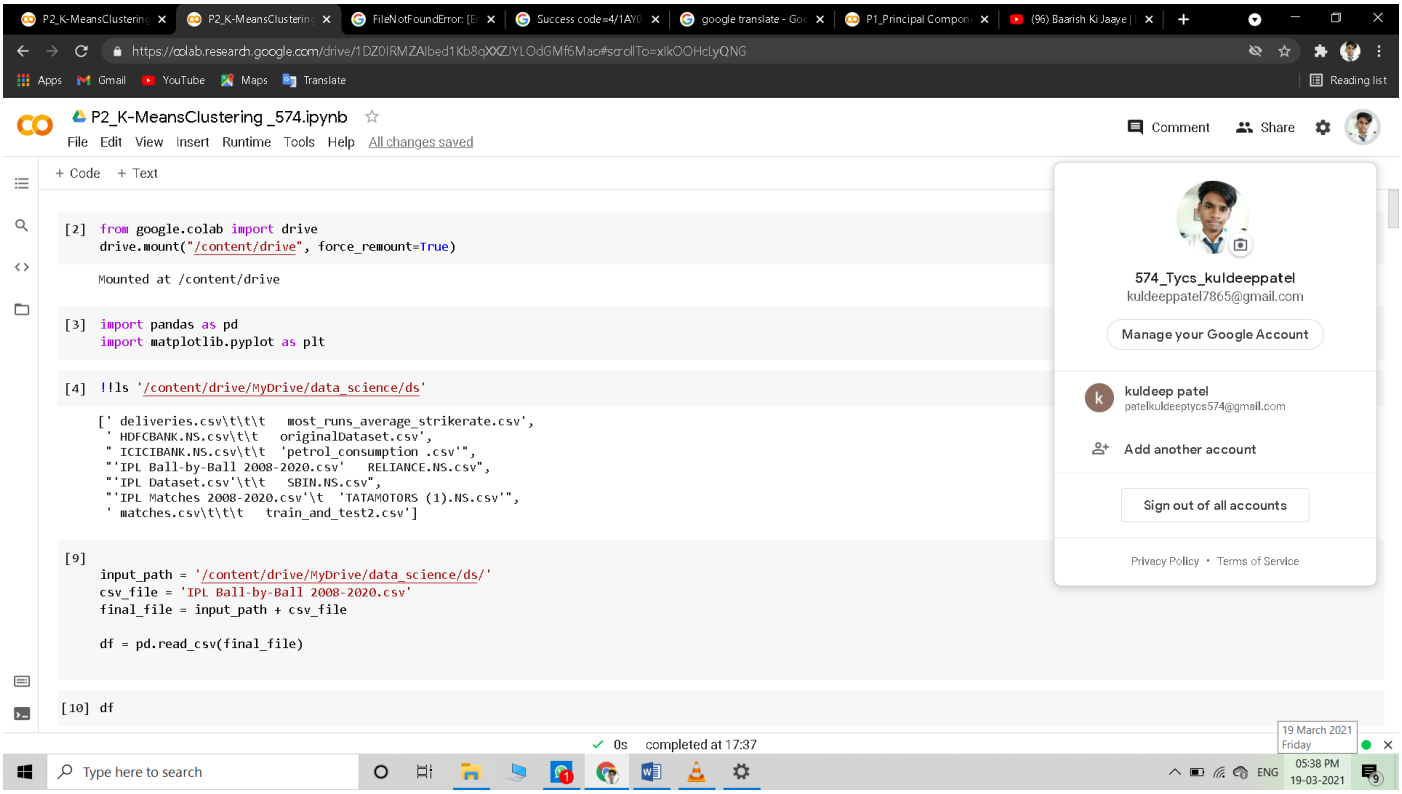
Practical – 2

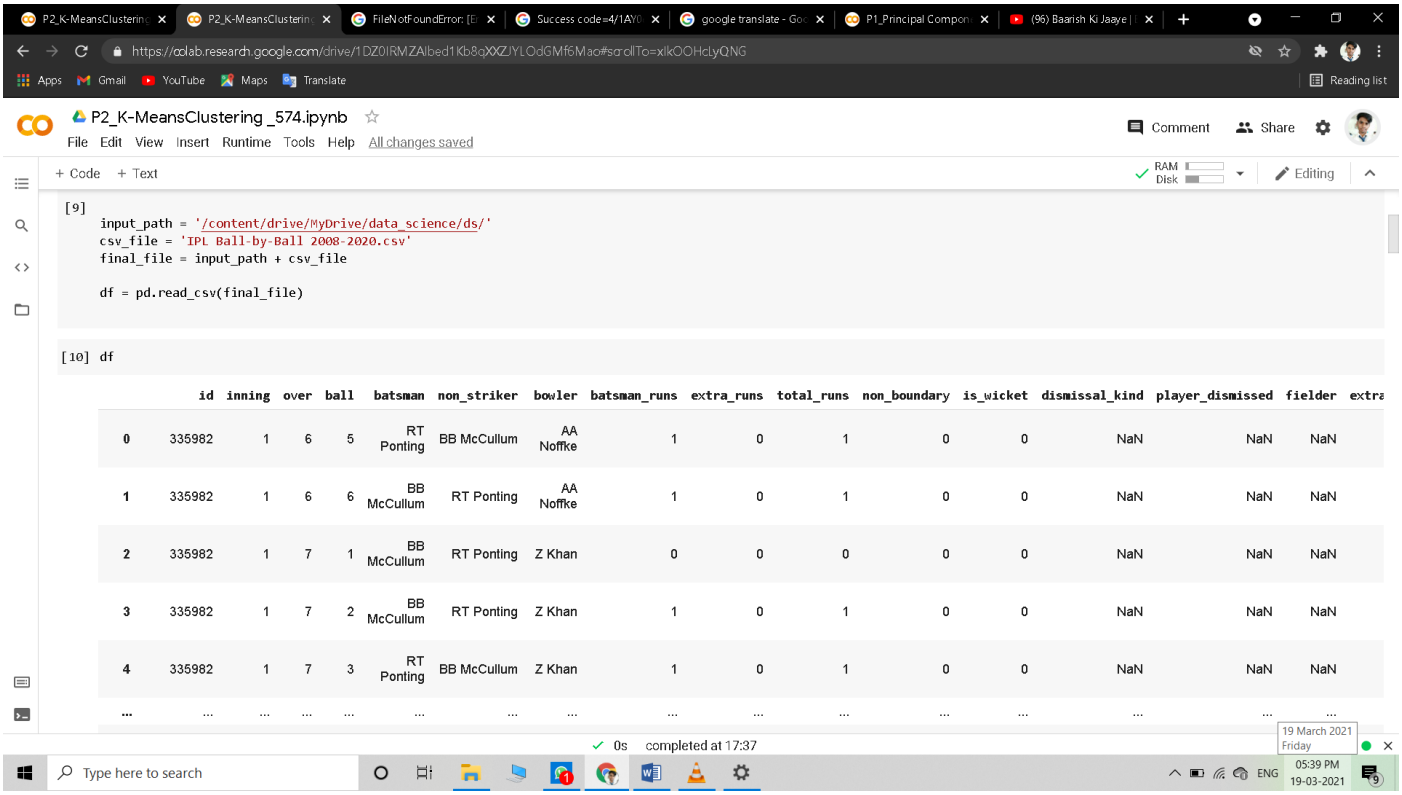
Aim : Practical of K-Means Clustering

Theory :

- K-Means Clustering is an unsupervised learning algorithm that is used to solve clustering problems in machine learning or data science.
- K-Means Clustering groups the unlabelled dataset into different clusters. Here K defines the number of predefined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.
- It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabelled dataset on its own without the need for any training.
- The algorithm takes the unlabelled dataset as input, divides the dataset into knumber of clusters, and repeats the process until it does not find the best clusters.
- The value of k should be predetermined in this algorithm. The k-means clustering algorithm mainly performs two tasks:
  1. Determines the best value for K-center points or centroids by an iterative process.
  2. Assigns each data point to its closest K-center. Those data points which are near to the particular K-center, create a cluster.

Code :





P2_K-MeansClustering _574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

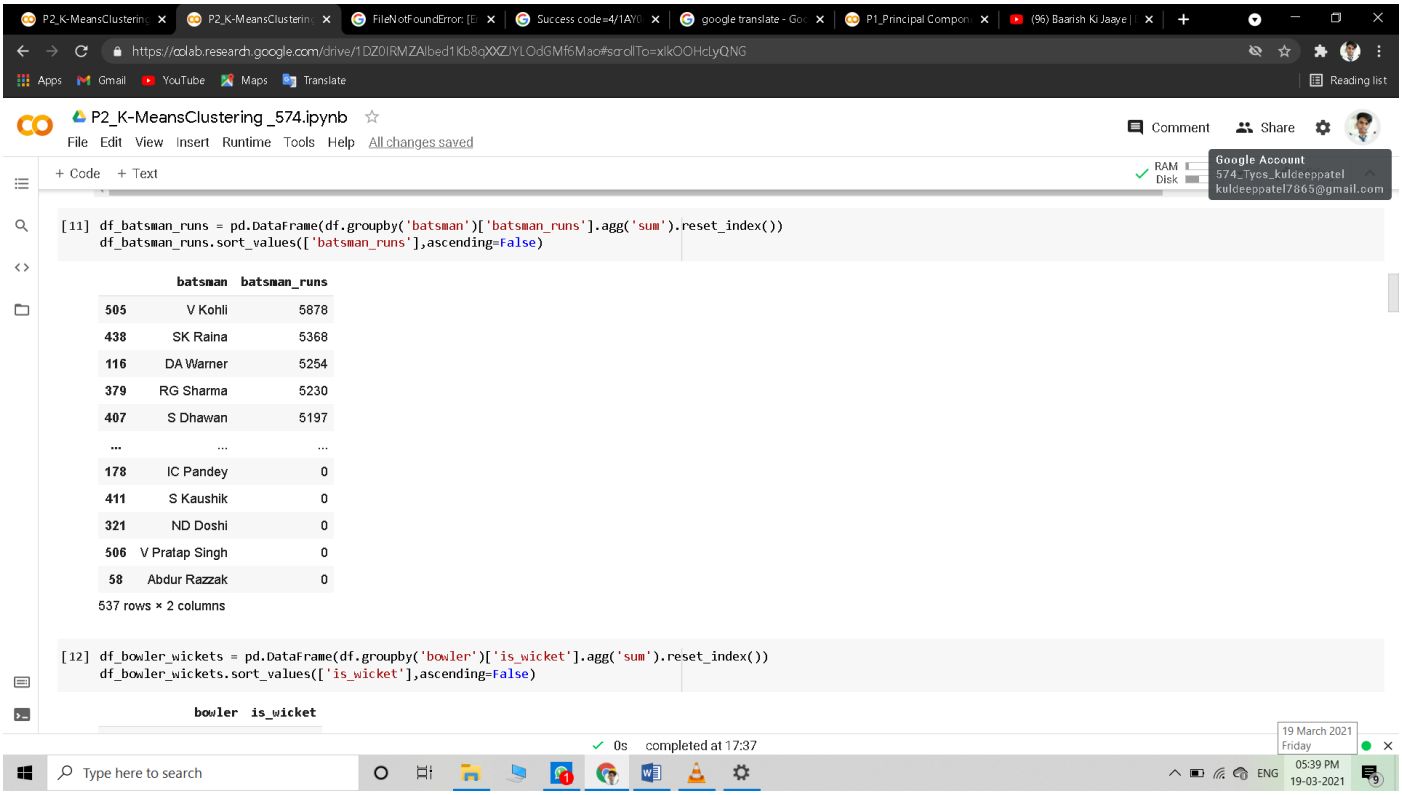
RAM Disk Editing

```
[9]
input_path = '/content/drive/MyDrive/data_science/ds/'
csv_file = 'IPL Ball-by-Ball 2008-2020.csv'
final_file = input_path + csv_file

df = pd.read_csv(final_file)
```

```
[10] df
```

	id	inning	over	ball	batsman	non_striker	bowler	batsman_runs	extra_runs	total_runs	non_boundary	is_wicket	dismissal_kind	player_dismissed	fielder	extra
0	335982	1	6	5	RT Ponting	BB McCullum	AA Noffke	1	0	1	0	0	NaN	NaN	NaN	
1	335982	1	6	6	BB McCullum	RT Ponting	AA Noffke	1	0	1	0	0	NaN	NaN	NaN	
2	335982	1	7	1	BB McCullum	RT Ponting	Z Khan	0	0	0	0	0	NaN	NaN	NaN	
3	335982	1	7	2	BB McCullum	RT Ponting	Z Khan	1	0	1	0	0	NaN	NaN	NaN	
4	335982	1	7	3	RT Ponting	BB McCullum	Z Khan	1	0	1	0	0	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

0s completed at 17:37



P2_K-MeansClustering _574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk

Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

```
[11] df_batsman_runs = pd.DataFrame(df.groupby('batsman')['batsman_runs'].agg('sum').reset_index())
     df_batsman_runs.sort_values(['batsman_runs'],ascending=False)
```

	batsman	batsman_runs
505	V Kohli	5878
438	SK Raina	5368
116	DA Warner	5254
379	RG Sharma	5230
407	S Dhawan	5197
...	...	...
178	IC Pandey	0
411	S Kaushik	0
321	ND Doshi	0
506	V Pratap Singh	0
58	Abdur Razzak	0

537 rows × 2 columns

```
[12] df_bowler_wickets = pd.DataFrame(df.groupby('bowler')['is_wicket'].agg('sum').reset_index())
     df_bowler_wickets.sort_values(['is_wicket'],ascending=False)
```

	bowler	is_wicket

0s completed at 17:37

P2_K-MeansClustering _574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

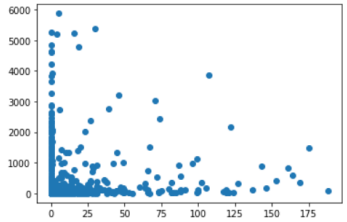
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

+ Code + Text

RAM Disk

```
[15] y = df_final_player_stats['batsman_runs']
     x = df_final_player_stats ['is_wicket']
     plt.scatter(x,y)
```

```
<matplotlib.collections.PathCollection at 0x7faf09ceec50>
```




```
[16] from sklearn.cluster import KMeans

     kmeans_model = KMeans(n_clusters=3,max_iter=1000)
```

```
[17] kmeans_model.fit(df_final_player_stats[['batsman_runs','is_wicket']])
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=1000,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

0s completed at 17:37

19 March 2021 Friday

Type here to search

ENG 05:40 PM 19-03-2021

P2_K-MeansClustering _574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

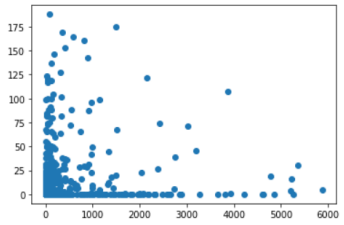
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

+ Code + Text

RAM Disk

```
[21] y = pd.DataFrame(df_final_player_stats['batsman_runs'])
     x = pd.DataFrame(df_final_player_stats ['is_wicket'])
     plt.scatter(y,x)
```

```
<matplotlib.collections.PathCollection at 0x7faef4ae2090>
```



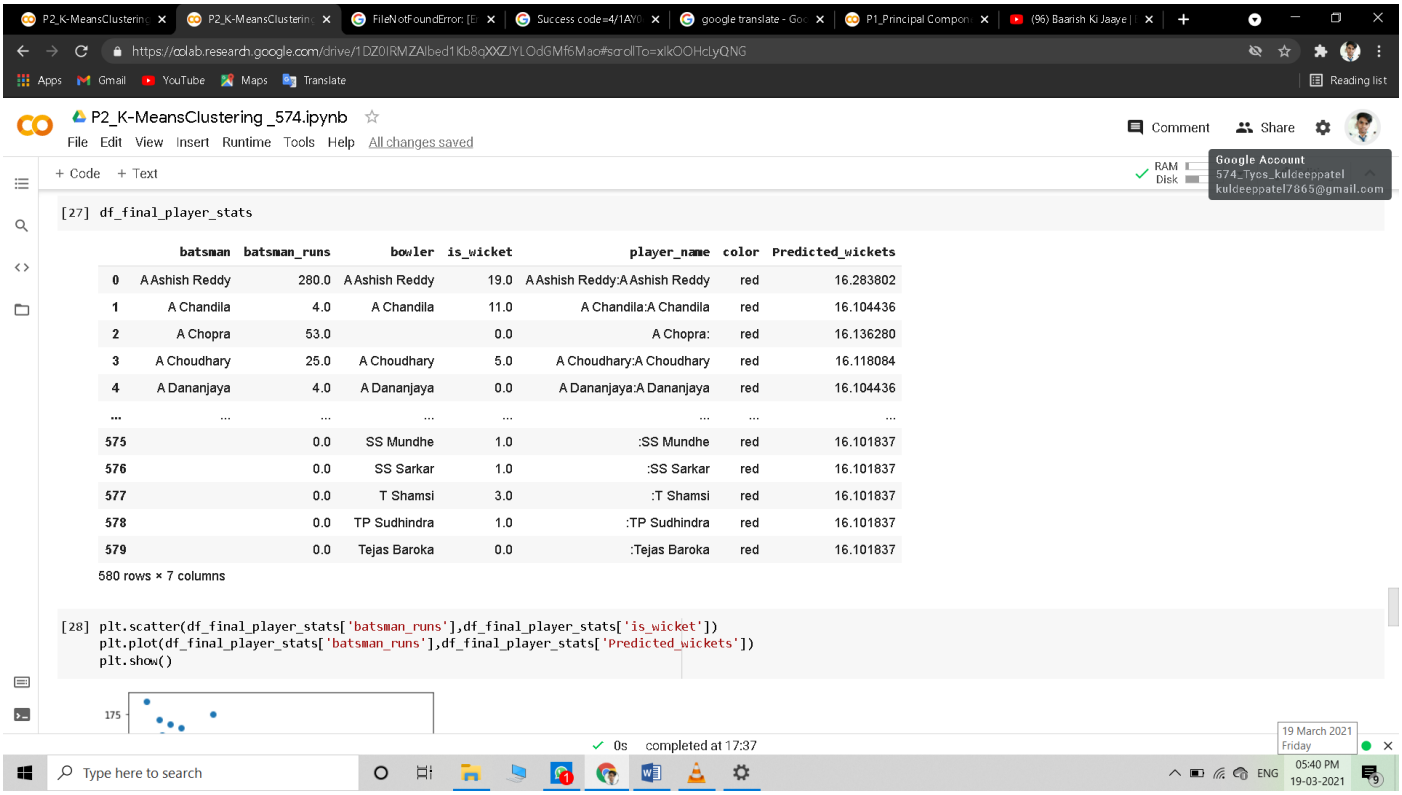

```
x
```

	is_wicket
0	19.0
1	11.0
2	0.0
3	5.0
4	0.0

0s completed at 17:37

Type here to search

ENG 05:40 PM 19-03-2021



P2_K-MeansClustering _574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

+ Code + Text

RAM Disk

[27] df_final_player_stats

	batsman	batsman_runs	bowler	is_wicket	player_name	color	Predicted_wickets
0	A Ashish Reddy	280.0	A Ashish Reddy	19.0	A Ashish Reddy:A Ashish Reddy	red	16.283802
1	A Chandila	4.0	A Chandila	11.0	A Chandila:A Chandila	red	16.104436
2	A Chopra	53.0		0.0	A Chopra:	red	16.136280
3	A Choudhary	25.0	A Choudhary	5.0	A Choudhary:A Choudhary	red	16.118084
4	A Dananjaya	4.0	A Dananjaya	0.0	A Dananjaya:A Dananjaya	red	16.104436
...	...	...	...	...	...	...	...
575		0.0	SS Mundhe	1.0	:SS Mundhe	red	16.101837
576		0.0	SS Sarkar	1.0	:SS Sarkar	red	16.101837
577		0.0	T Shamsi	3.0	:T Shamsi	red	16.101837
578		0.0	TP Sudhindra	1.0	:TP Sudhindra	red	16.101837
579		0.0	Tejas Baroka	0.0	:Tejas Baroka	red	16.101837

580 rows × 7 columns

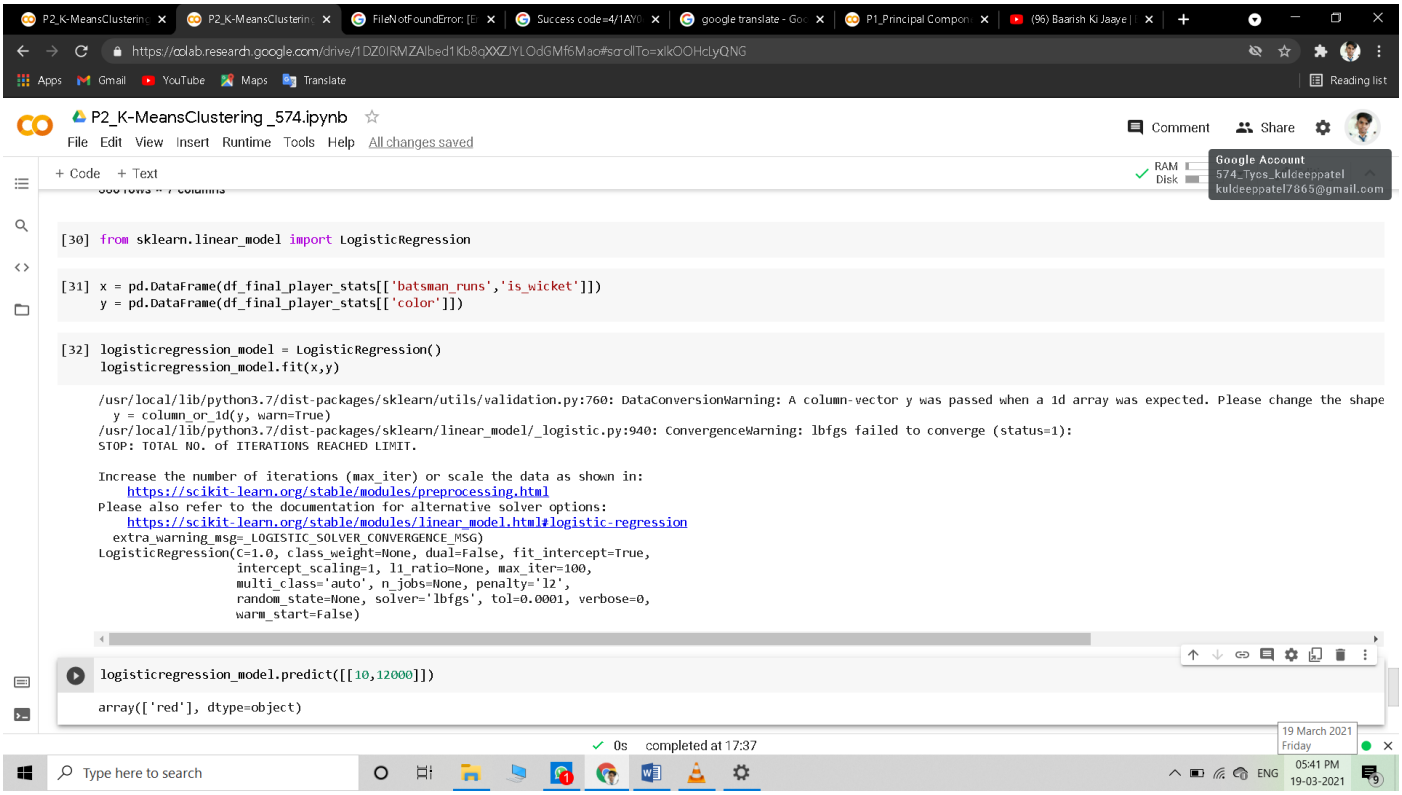
```
[28] plt.scatter(df_final_player_stats['batsman_runs'],df_final_player_stats['is_wicket'])
     plt.plot(df_final_player_stats['batsman_runs'],df_final_player_stats['Predicted_wickets'])
     plt.show()
```


0s completed at 17:37

Type here to search

19 March 2021 Friday

05:40 PM 19-03-2021



P2_K-MeansClustering _574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

+ Code + Text

RAM Disk

```
[30] from sklearn.linear_model import LogisticRegression

[31] x = pd.DataFrame(df_final_player_stats[['batsman_runs','is_wicket']])
     y = pd.DataFrame(df_final_player_stats[['color']])

[32] logisticregression_model = LogisticRegression()
     logisticregression_model.fit(x,y)

/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape
  y = column_or_1d(y, warn=True)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)

logisticregression_model.predict([[10,12000]])

array(['red'], dtype=object)
```

0s completed at 17:37

Type here to search

19 March 2021 Friday

05:41 PM 19-03-2021

Colab Link :

https://colab.research.google.com/drive/1DZ0IRMZAIbed1Kb8qXXZJYLOdGMf6Mao?usp=sharing

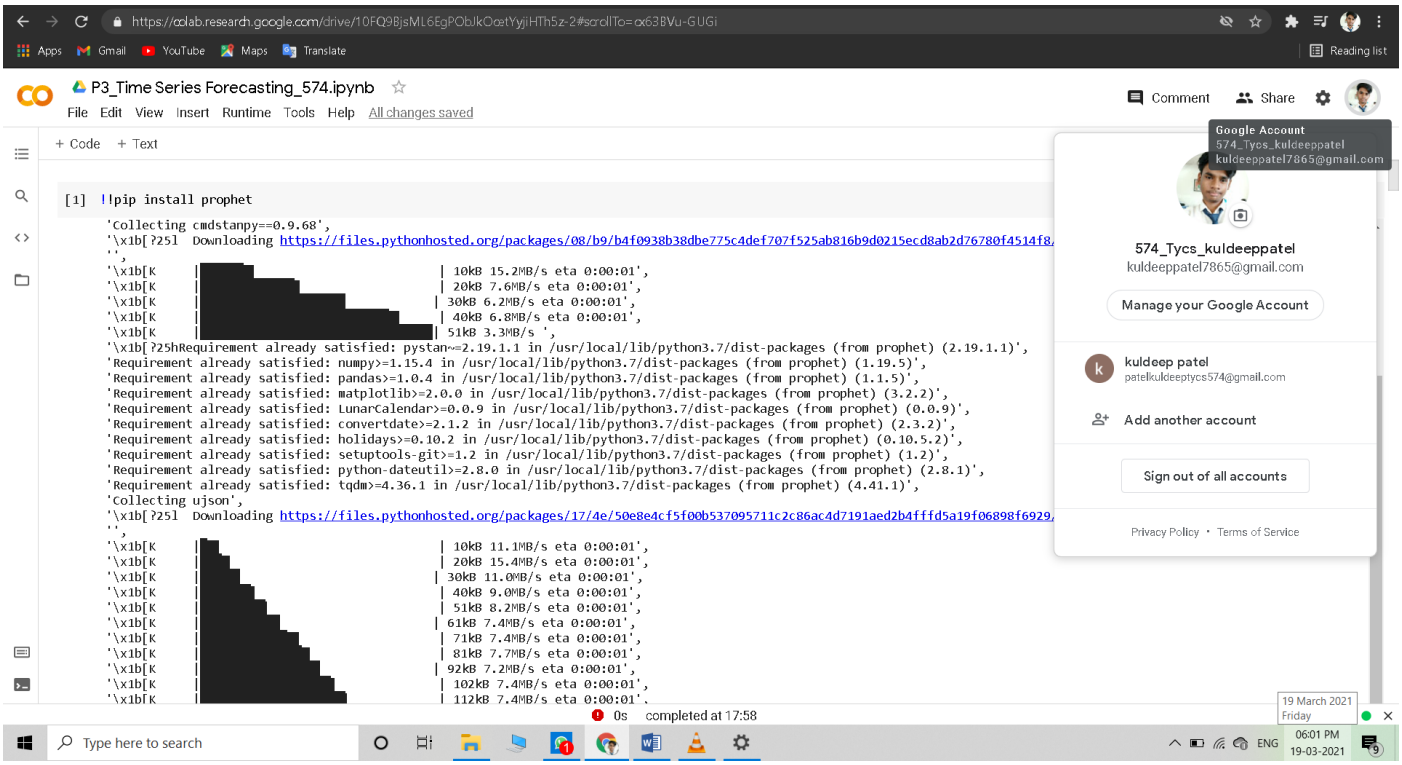
Practical – 3

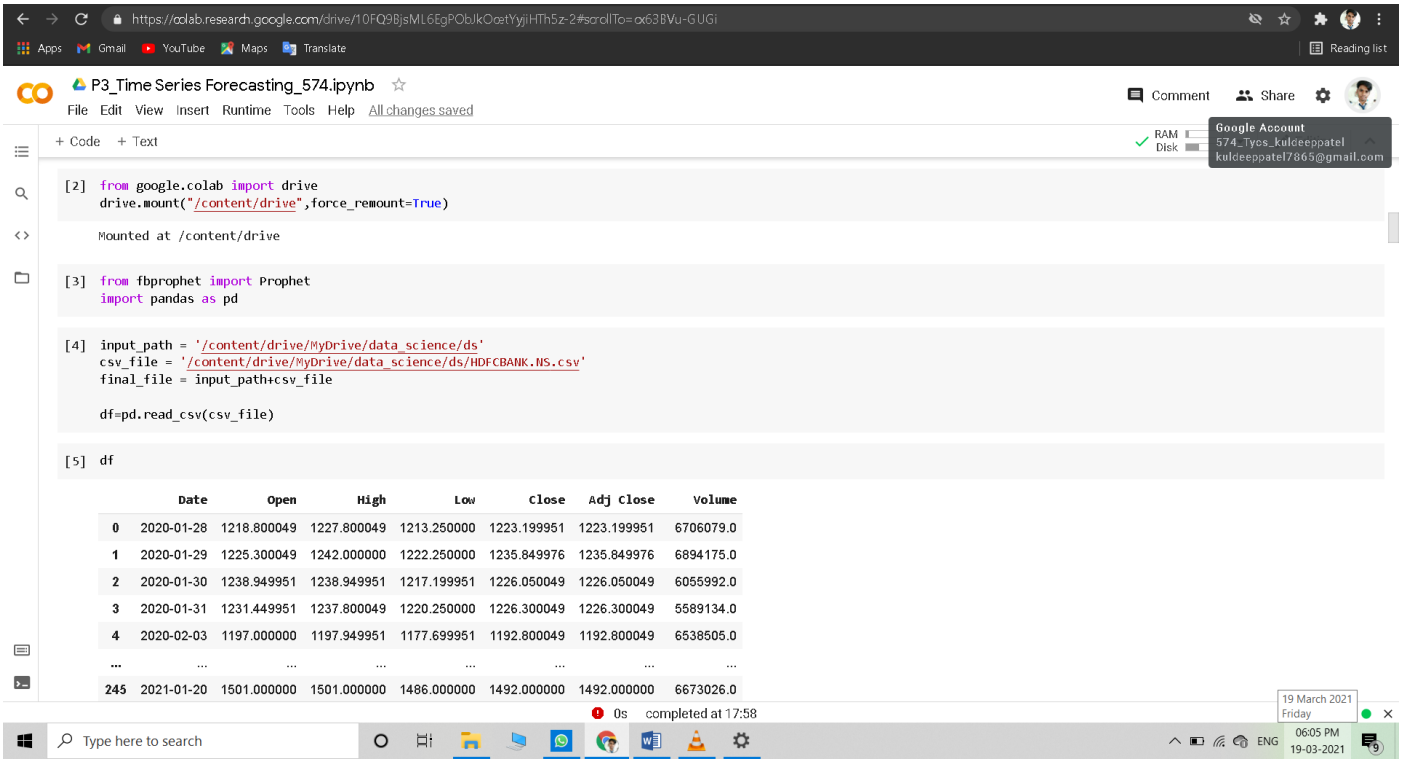
Aim : Practical of Time Series Forecasting

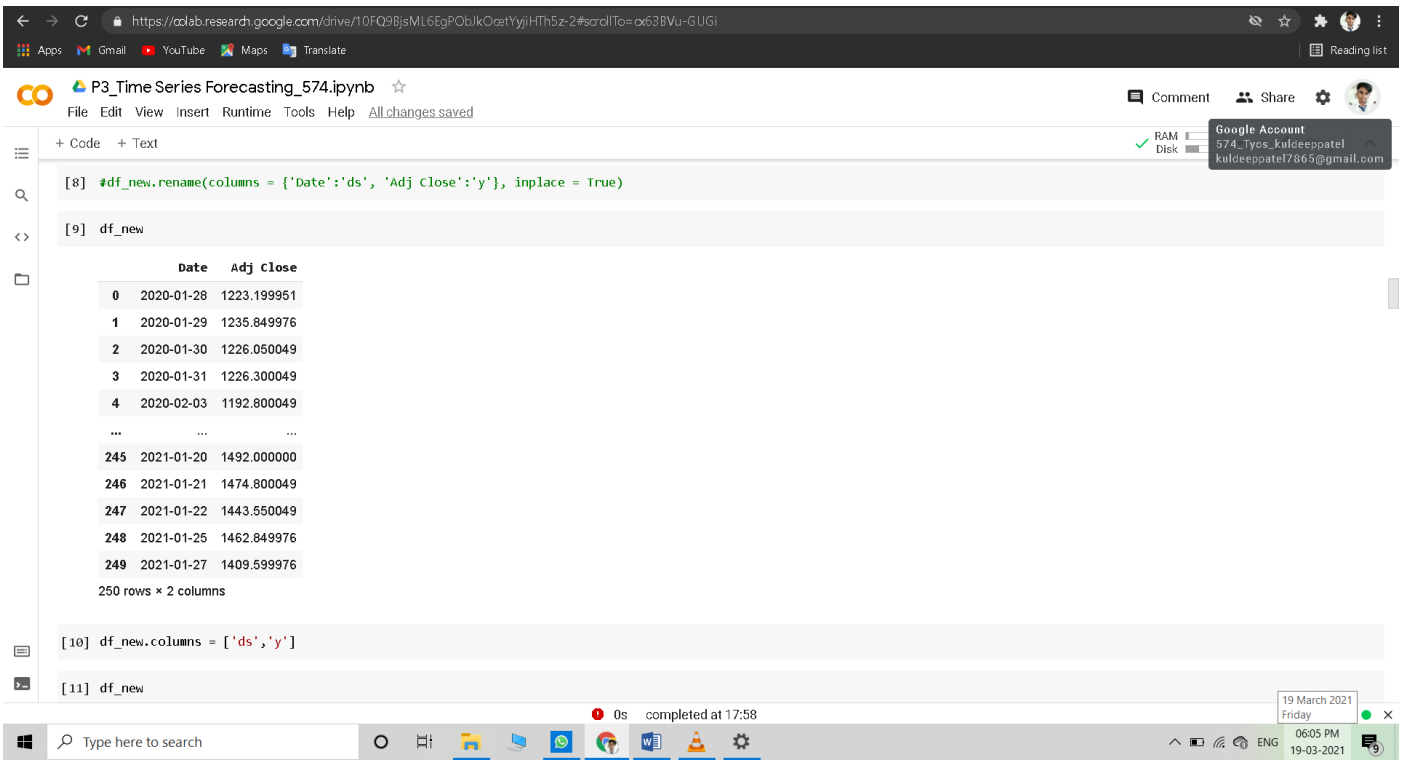
Theory :

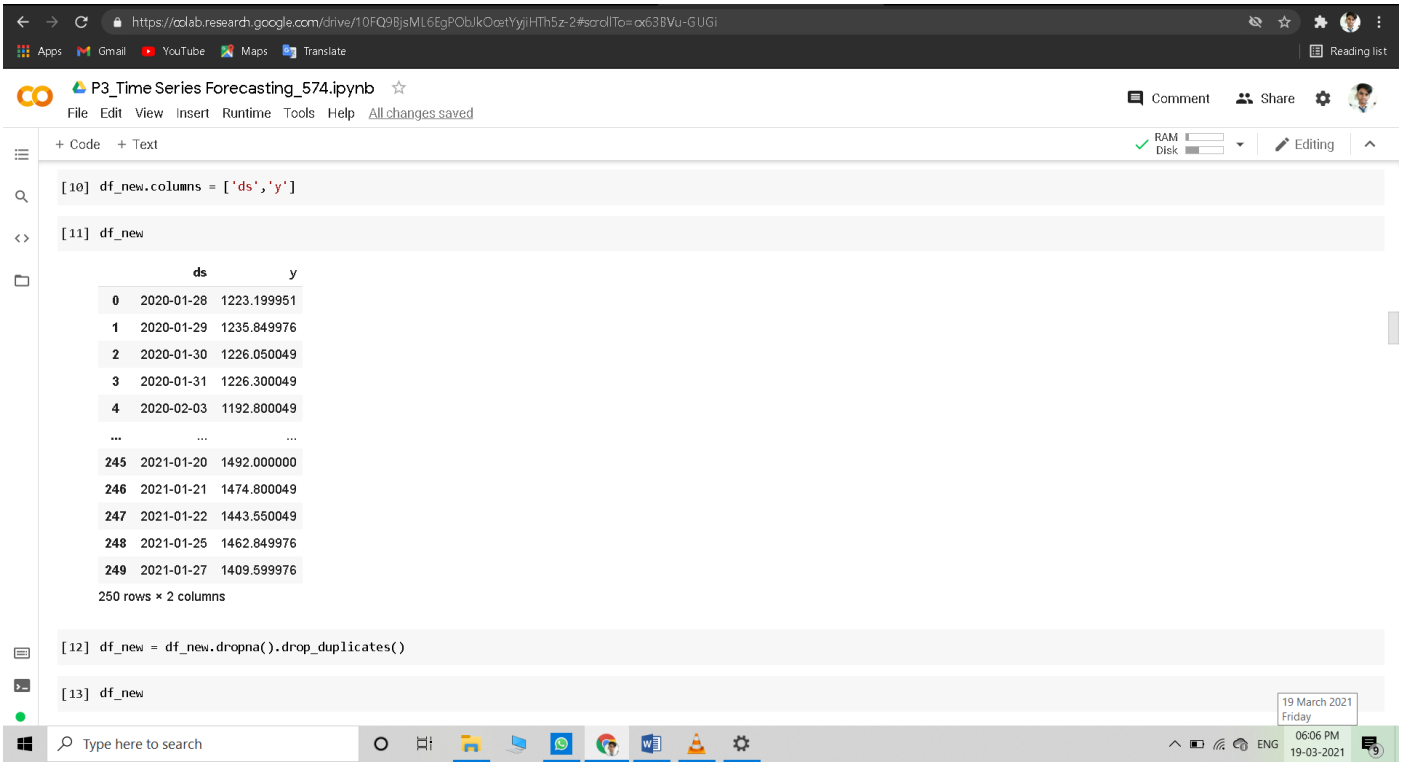
- Time series analysis comprises methods for analyzing time-series data in order to extract meaningful statistics and other characteristics of the data.
- Time series forecasting is the use of a model to predict future values based on previously observed values.
- Time series are widely used for non-stationary data, like economic, weather, stock price, and retail sales in this post.

Code :




https://colab.research.google.com/drive/10FQ9BjsML6EgPObJkOcetYyjiHTh5z-2#scrollTo=ox63BVu-GUGi
Apps Gmail YouTube Maps Translate Reading list
P3_Time Series Forecasting_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com
[2] from google.colab import drive
drive.mount("/content/drive",force_remount=True)
Mounted at /content/drive
[3] from fbprophet import Prophet
import pandas as pd
[4] input_path = '/content/drive/MyDrive/data_science/ds'
csv_file = '/content/drive/MyDrive/data_science/ds/HDFCBANK.NS.csv'
final_file = input_path+csv_file
df=pd.read_csv(csv_file)
[5] df
Date Open High Low Close Adj Close Volume
0 2020-01-28 1218.800049 1227.800049 1213.250000 1223.199951 1223.199951 6706079.0
1 2020-01-29 1225.300049 1242.000000 1222.250000 1235.849976 1235.849976 6894175.0
2 2020-01-30 1238.949951 1238.949951 1217.199951 1226.050049 1226.050049 6055992.0
3 2020-01-31 1231.449951 1237.800049 1220.250000 1226.300049 1226.300049 5589134.0
4 2020-02-03 1197.000000 1197.949951 1177.699951 1192.800049 1192.800049 6538505.0
... ...
245 2021-01-20 1501.000000 1501.000000 1486.000000 1492.000000 1492.000000 6673026.0
0s completed at 17:58
Type here to search
ENG 06:05 PM 19-03-2021
19 March 2021 Friday


https://colab.research.google.com/drive/10FQ9BjsML6EgPObJkOcetYyjiHTh5z-2#scrollTo=ox63BVu-GUGi
Apps Gmail YouTube Maps Translate Reading list
P3_Time Series Forecasting_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com
[8] #df_new.rename(columns = {'Date':'ds', 'Adj Close':'y'}, inplace = True)
[9] df_new
Date Adj Close
0 2020-01-28 1223.199951
1 2020-01-29 1235.849976
2 2020-01-30 1226.050049
3 2020-01-31 1226.300049
4 2020-02-03 1192.800049
... ...
245 2021-01-20 1492.000000
246 2021-01-21 1474.800049
247 2021-01-22 1443.550049
248 2021-01-25 1462.849976
249 2021-01-27 1409.599976
250 rows × 2 columns
[10] df_new.columns = ['ds','y']
[11] df_new
0s completed at 17:58
Type here to search
ENG 06:05 PM 19-03-2021
19 March 2021 Friday



P3_Time Series Forecasting_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[10] df_new.columns = ['ds','y']
```

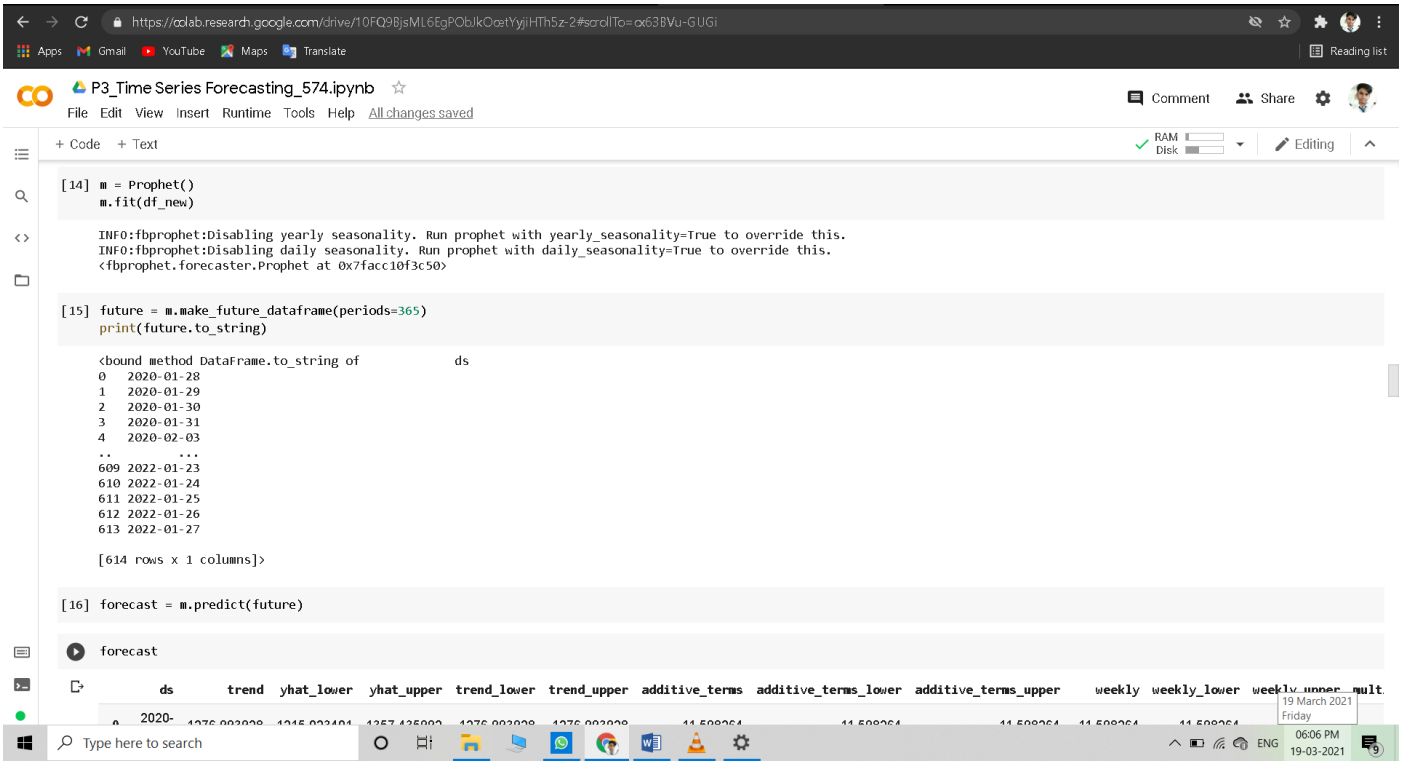
```
[11] df_new
```

	ds	y
0	2020-01-28	1223.199951
1	2020-01-29	1235.849976
2	2020-01-30	1226.050049
3	2020-01-31	1226.300049
4	2020-02-03	1192.800049
...	...	...
245	2021-01-20	1492.000000
246	2021-01-21	1474.800049
247	2021-01-22	1443.550049
248	2021-01-25	1462.849976
249	2021-01-27	1409.599976

250 rows × 2 columns

```
[12] df_new = df_new.dropna().drop_duplicates()
```

```
[13] df_new
```



P3_Time Series Forecasting_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[14] m = Prophet()
     m.fit(df_new)
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
<fbprophet.forecaster.Prophet at 0x7facc10f3c50>
```

```
[15] future = m.make_future_dataframe(periods=365)
     print(future.to_string)
```

```
<bound method DataFrame.to_string of            ds
0   2020-01-28
1   2020-01-29
2   2020-01-30
3   2020-01-31
4   2020-02-03
..         ...
609 2022-01-23
610 2022-01-24
611 2022-01-25
612 2022-01-26
613 2022-01-27

[614 rows x 1 columns]>
```

```
[16] forecast = m.predict(future)
```

```
forecast
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_upper	weekly	weekly_lower	weekly_upper	mult
0	2020-	1276.993928	1216.923401	1257.426902	1276.993928	1276.993928	11.508264	11.508264	11.508264	11.508264	11.508264	11.508264	

https://colab.research.google.com/drive/10FQ9BjsML6EgPObJkOcetYyjiHTh5z-2#scrollTo=cx63BVu-GUGi

Apps Gmail YouTube Maps Translate Reading list

P3_Time Series Forecasting_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```
[18] forecast
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_upper	weekly	weekly_lower	weekly_upper	mult
0	2020-01-28	1276.993928	1215.023401	1357.435992	1276.993928	1276.993928	11.598264	11.598264	11.598264	11.598264	11.598264	11.598264	
1	2020-01-29	1271.086154	1215.210275	1351.461648	1271.086154	1271.086154	13.375342	13.375342	13.375342	13.375342	13.375342	13.375342	
2	2020-01-30	1265.178381	1211.340684	1349.306163	1265.178381	1265.178381	15.687166	15.687166	15.687166	15.687166	15.687166	15.687166	
3	2020-01-31	1259.270607	1201.296831	1342.012398	1259.270607	1259.270607	12.825809	12.825809	12.825809	12.825809	12.825809	12.825809	
4	2020-02-03	1241.547286	1178.660093	1317.474825	1241.547286	1241.547286	4.921369	4.921369	4.921369	4.921369	4.921369	4.921369	
...	...	...	...	...	...	...	...	...	...	...	...	...	
609	2022-01-23	2454.806251	1575.107801	3324.925119	1583.018191	3331.174403	-29.203975	-29.203975	-29.203975	-29.203975	-29.203975	-29.203975	
610	2022-01-24	2457.466127	1606.842020	3371.089852	1581.769724	3338.261883	4.921369	4.921369	4.921369	4.921369	4.921369	4.921369	
611	2022-01-25	2460.126003	1603.404661	3356.197658	1580.521256	3345.537313	11.598264	11.598264	11.598264	11.598264	11.598264	11.598264	
612	2022-01-26	2462.785880	1601.514292	3382.679265	1579.299578	3355.165712	13.375342	13.375342	13.375342	13.375342	13.375342	13.375342	
613	2022-01-27	2465.445756	1615.736551	3384.740749	1578.185968	3363.399354	15.687166	15.687166	15.687166	15.687166	15.687166	15.687166	

614 rows × 16 columns

19 March 2021 Friday

Type here to search ENG 06:06 PM 19-03-2021

https://colab.research.google.com/drive/10FQ9BjsML6EgPObJkOcetYyjiHTh5z-2#scrollTo=cx63BVu-GUGi

Apps Gmail YouTube Maps Translate Reading list

P3_Time Series Forecasting_574.ipynb

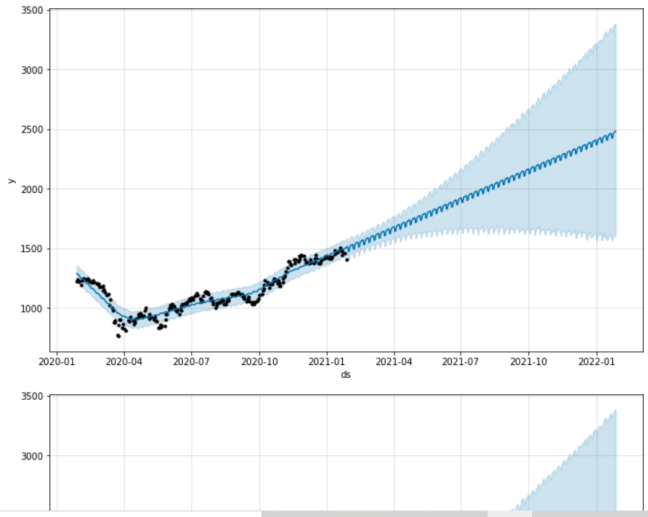
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

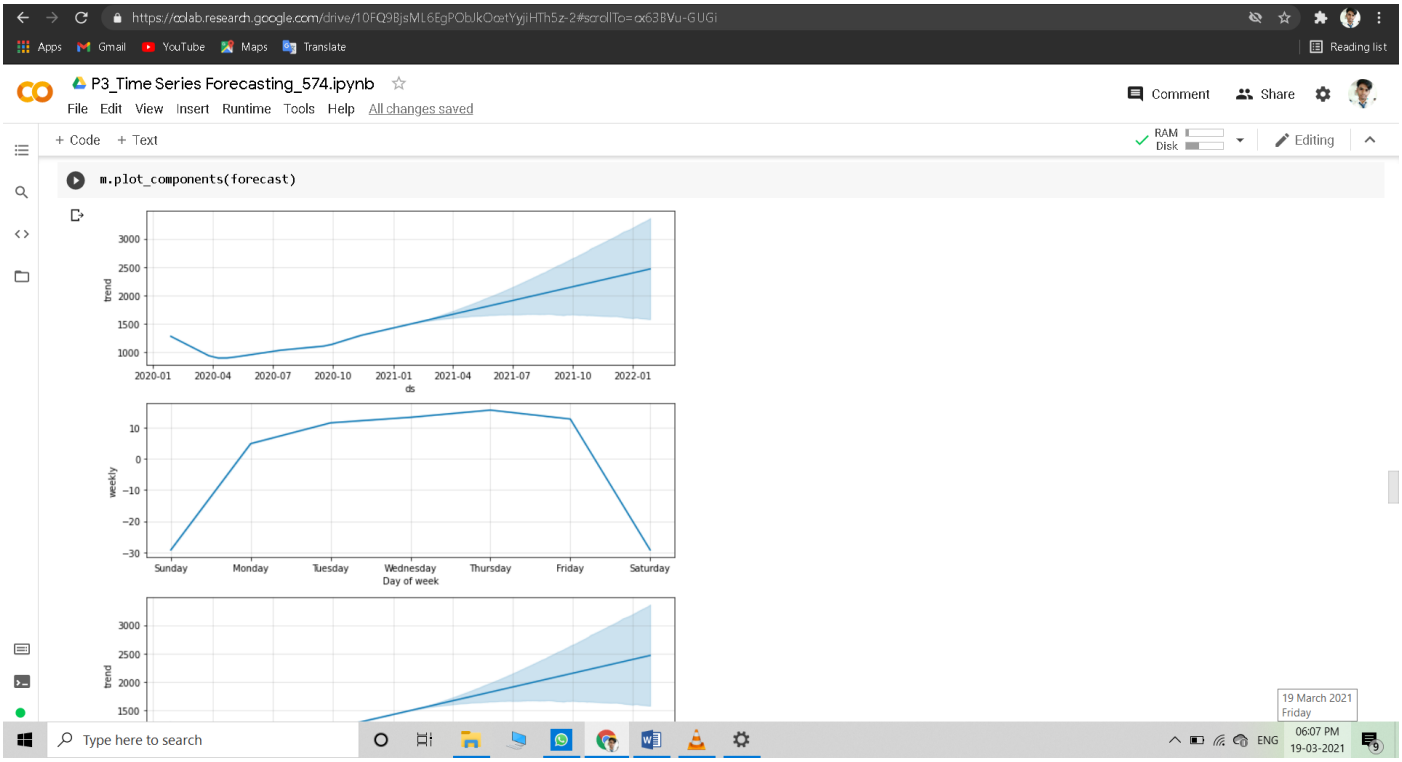
RAM Disk Editing

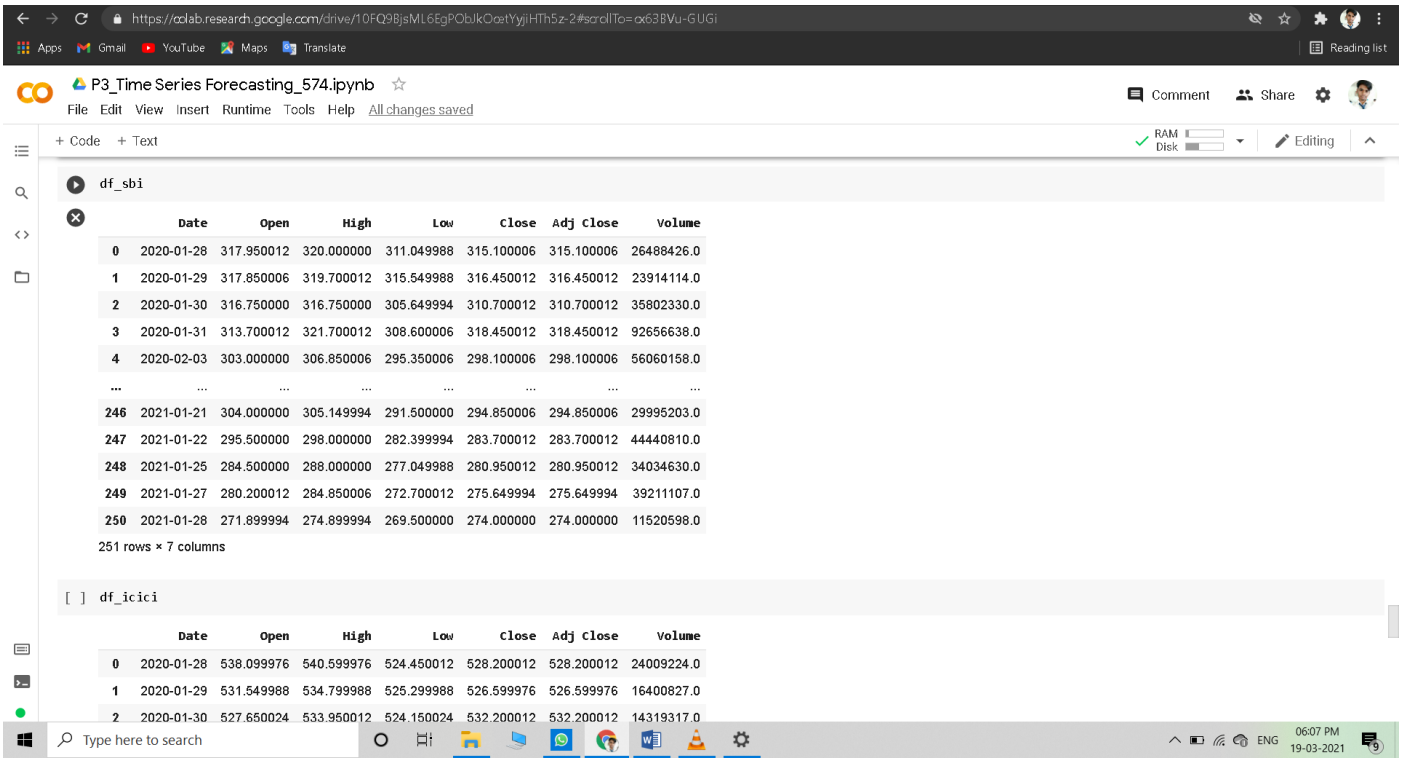
```
[19] m.plot(forecast)
```



19 March 2021 Friday

Type here to search ENG 06:07 PM 19-03-2021

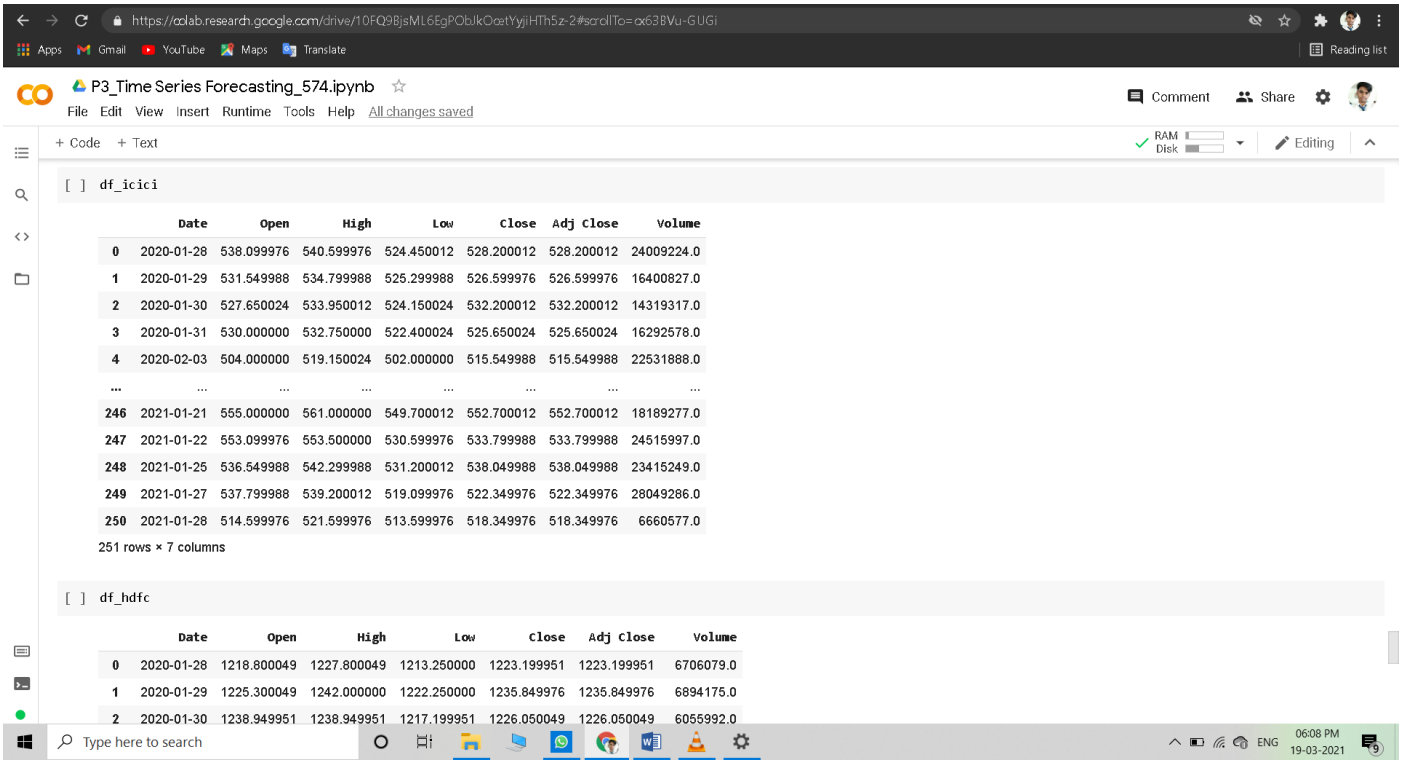




	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-28	317.950012	320.000000	311.049988	315.100006	315.100006	26488426.0
1	2020-01-29	317.850006	319.700012	315.549988	316.450012	316.450012	23914114.0
2	2020-01-30	316.750000	316.750000	305.649994	310.700012	310.700012	35802330.0
3	2020-01-31	313.700012	321.700012	308.600006	318.450012	318.450012	92656638.0
4	2020-02-03	303.000000	306.850006	295.350006	298.100006	298.100006	56060158.0
...	...	...	...	...	...	...	...
246	2021-01-21	304.000000	305.149994	291.500000	294.850006	294.850006	29995203.0
247	2021-01-22	295.500000	298.000000	282.399994	283.700012	283.700012	44440810.0
248	2021-01-25	284.500000	288.000000	277.049988	280.950012	280.950012	34034630.0
249	2021-01-27	280.200012	284.850006	272.700012	275.649994	275.649994	39211107.0
250	2021-01-28	271.899994	274.899994	269.500000	274.000000	274.000000	11520598.0

251 rows × 7 columns

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-28	538.099976	540.599976	524.450012	528.200012	528.200012	24009224.0
1	2020-01-29	531.549988	534.799988	525.299988	526.599976	526.599976	16400827.0
2	2020-01-30	527.650024	533.950012	524.150024	532.200012	532.200012	14319317.0



P3_Time Series Forecasting_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Editing

+ Code + Text

```
[ ] df_icici
```

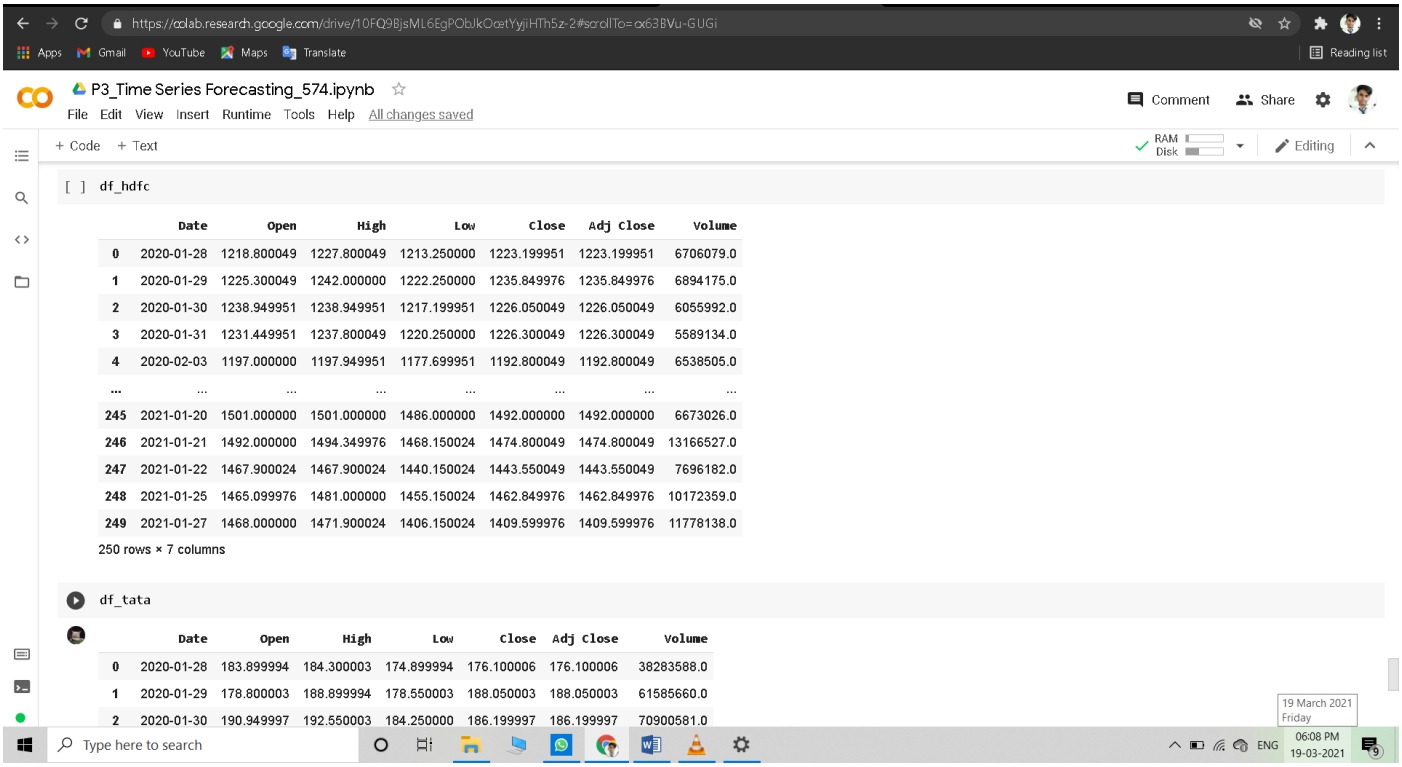
	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-28	538.099976	540.599976	524.450012	528.200012	528.200012	24009224.0
1	2020-01-29	531.549988	534.799988	525.299988	526.599976	526.599976	16400827.0
2	2020-01-30	527.650024	533.950012	524.150024	532.200012	532.200012	14319317.0
3	2020-01-31	530.000000	532.750000	522.400024	525.650024	525.650024	16292578.0
4	2020-02-03	504.000000	519.150024	502.000000	515.549988	515.549988	22531888.0
...	...	...	...	...	...	...	...
246	2021-01-21	555.000000	561.000000	549.700012	552.700012	552.700012	18189277.0
247	2021-01-22	553.099976	553.500000	530.599976	533.799988	533.799988	24515997.0
248	2021-01-25	536.549988	542.299988	531.200012	538.049988	538.049988	23415249.0
249	2021-01-27	537.799988	539.200012	519.099976	522.349976	522.349976	28049286.0
250	2021-01-28	514.599976	521.599976	513.599976	518.349976	518.349976	6660577.0

251 rows × 7 columns

```
[ ] df_hdfc
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-28	1218.800049	1227.800049	1213.250000	1223.199951	1223.199951	6706079.0
1	2020-01-29	1225.300049	1242.000000	1222.250000	1235.849976	1235.849976	6894175.0
2	2020-01-30	1238.949951	1238.949951	1217.199951	1226.050049	1226.050049	6055992.0

Type here to search ENG 06:08 PM 19-03-2021



P3_Time Series Forecasting_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Editing

+ Code + Text

```
[ ] df_hdfc
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-28	1218.800049	1227.800049	1213.250000	1223.199951	1223.199951	6706079.0
1	2020-01-29	1225.300049	1242.000000	1222.250000	1235.849976	1235.849976	6894175.0
2	2020-01-30	1238.949951	1238.949951	1217.199951	1226.050049	1226.050049	6055992.0
3	2020-01-31	1231.449951	1237.800049	1220.250000	1226.300049	1226.300049	5589134.0
4	2020-02-03	1197.000000	1197.949951	1177.699951	1192.800049	1192.800049	6538505.0
...	...	...	...	...	...	...	...
245	2021-01-20	1501.000000	1501.000000	1486.000000	1492.000000	1492.000000	6673026.0
246	2021-01-21	1492.000000	1494.349976	1468.150024	1474.800049	1474.800049	13166527.0
247	2021-01-22	1467.900024	1467.900024	1440.150024	1443.550049	1443.550049	7696182.0
248	2021-01-25	1465.099976	1481.000000	1455.150024	1462.849976	1462.849976	10172359.0
249	2021-01-27	1468.000000	1471.900024	1406.150024	1409.599976	1409.599976	11778138.0

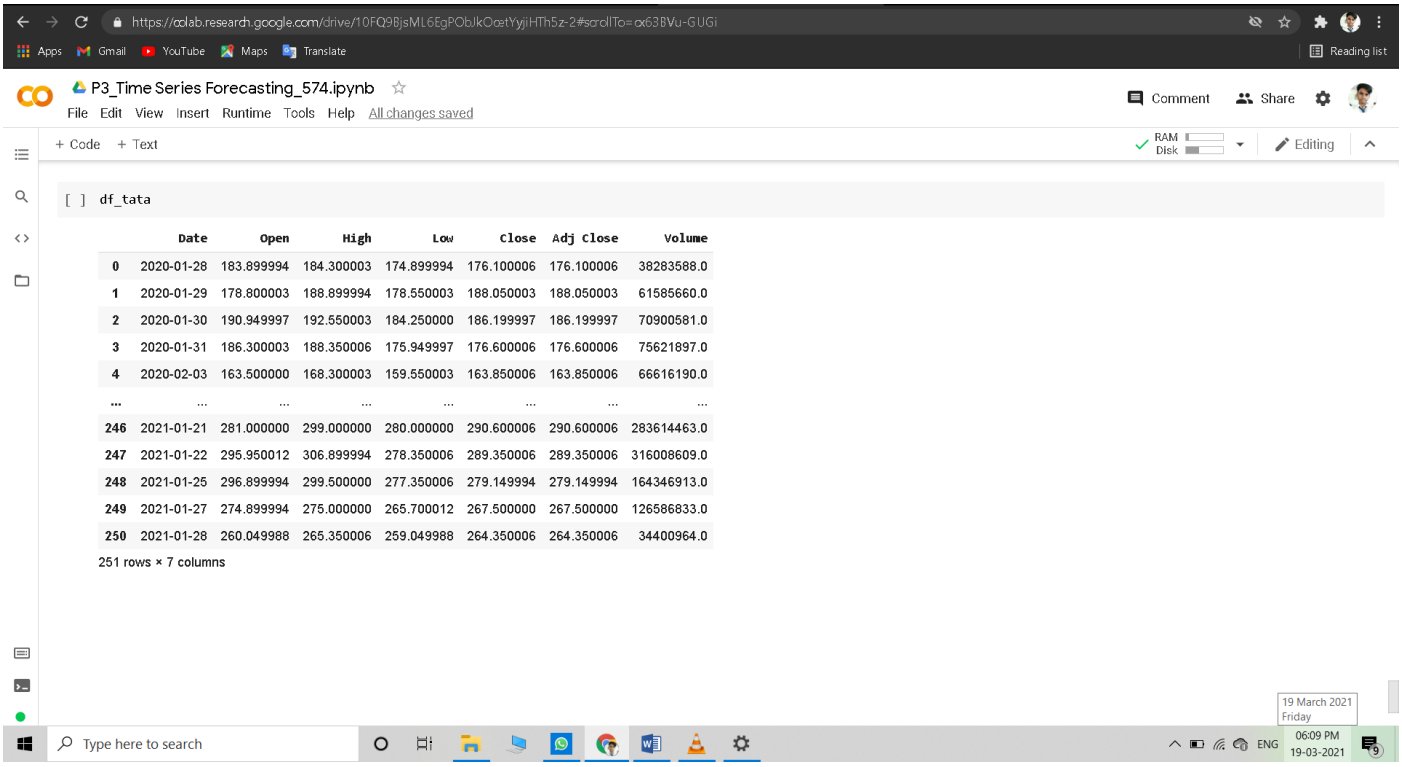
250 rows × 7 columns

```
df_tata
```

	Date	Open	High	Low	Close	Adj Close	Volume
0	2020-01-28	183.899994	184.300003	174.899994	176.100006	176.100006	38283588.0
1	2020-01-29	178.800003	188.899994	178.550003	188.050003	188.050003	61585660.0
2	2020-01-30	190.949997	192.550003	184.250000	186.199997	186.199997	70900581.0

19 March 2021 Friday

Type here to search ENG 06:08 PM 19-03-2021



Colab Link :

https://colab.research.google.com/drive/10FQ9BjsML6EgPObJkOcetYyjiHTh5z-2?usp=sharing

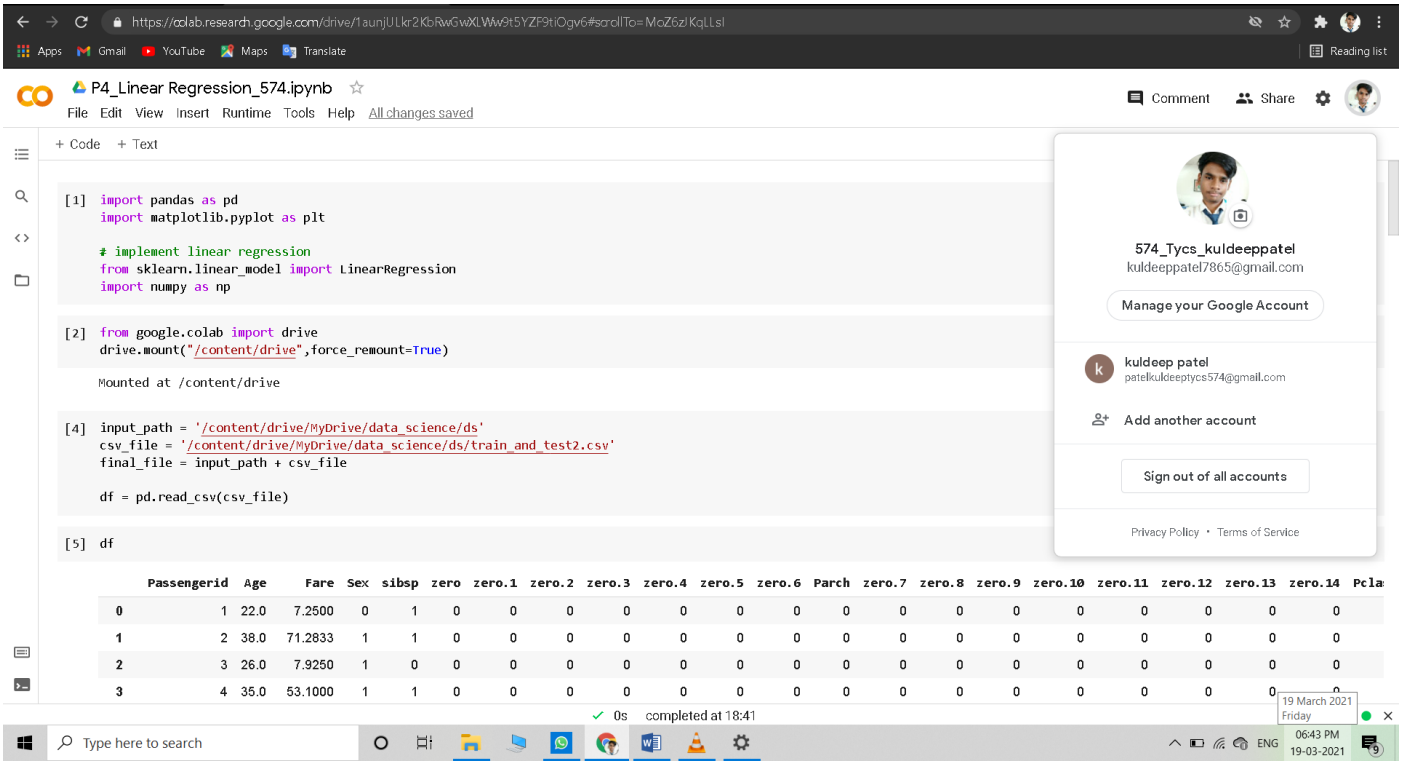
Practical – 4

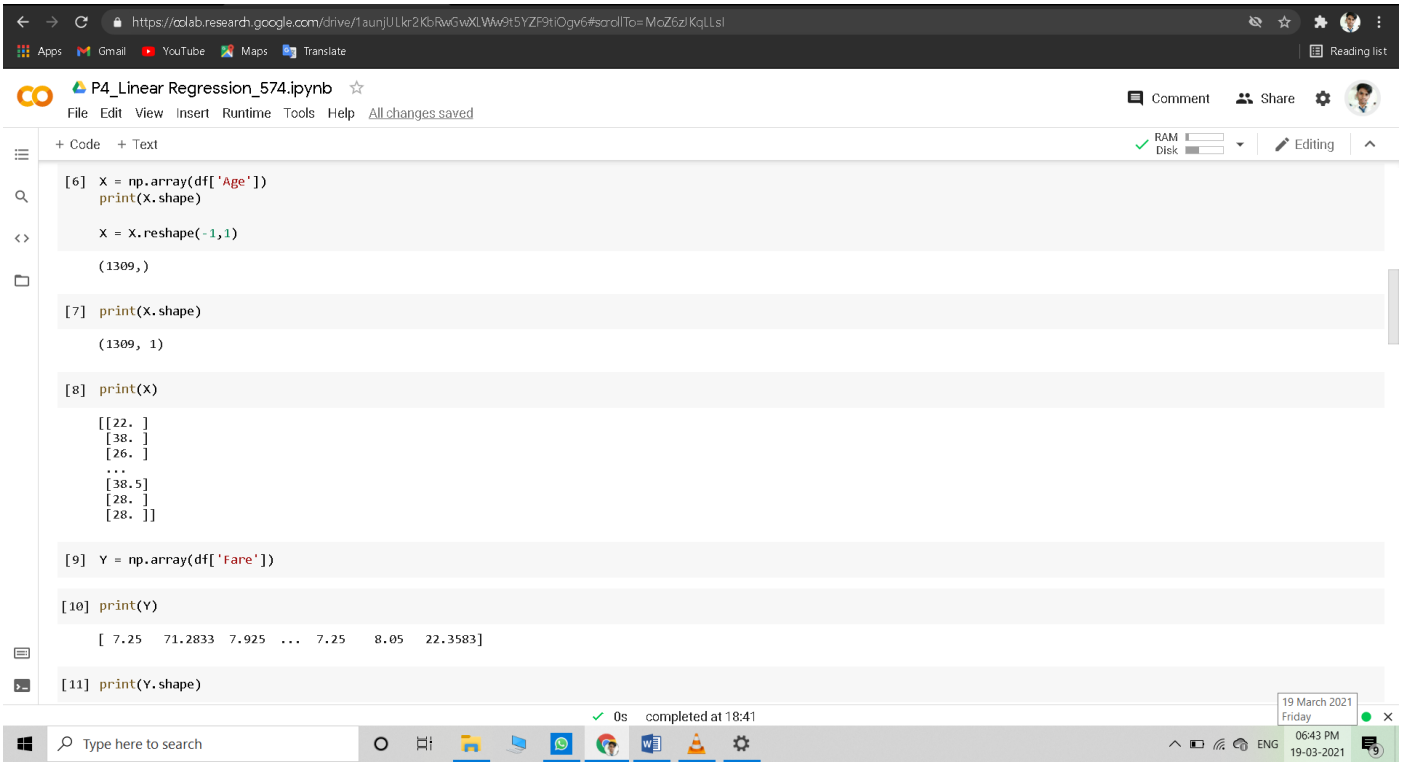
Aim : Practical of Simple/ Multiple Linear Regression

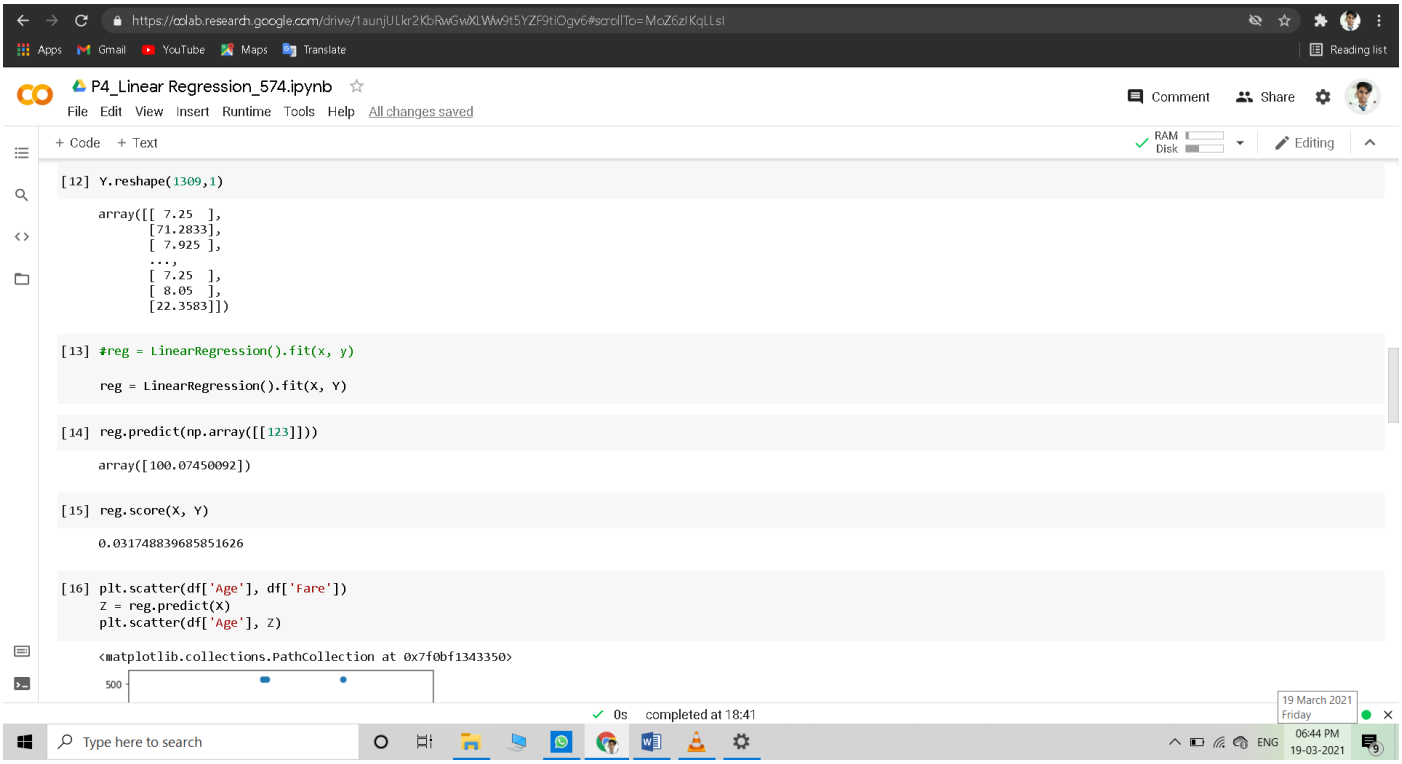
Theory :

- Linear regression is one of the most common techniques of regression analysis. Multiple regression is a broader class of regressions that encompasses linear and nonlinear regressions with multiple explanatory variables.
- Regression as a tool helps pool data together to help people and companies make informed decisions.
- There are different variables at play in regression, including a dependent variable—the main variable that you're trying to understand—and an independent variable—factors that may have an impact on the dependent variable.
- Linear Regression: It is also called simple linear regression. It establishes the relationship between two variables using a straight line. Linear regression attempts to draw a line that comes closest to the data by finding the slope and intercept that define the line and minimize regression errors.
- Multiple Regression: It is rare that a dependent variable is explained by only one variable. In this case, an analyst uses multiple regression, which attempts to explain a dependent variable using more than one independent variable. Multiple regressions can be linear and nonlinear.

Code :




P4_Linear Regression_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[6] X = np.array(df['Age'])
print(X.shape)
X = X.reshape(-1,1)
(1309,)
[7] print(X.shape)
(1309, 1)
[8] print(X)
[[22.]
[38.]
[26.]
...
[38.5]
[28.]
[28.]]
[9] Y = np.array(df['Fare'])
[10] print(Y)
[7.25 71.2833 7.925 ... 7.25 8.05 22.3583]
[11] print(Y.shape)
0s completed at 18:41
Type here to search
19 March 2021
Friday
06:43 PM
19-03-2021


P4_Linear Regression_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[12] Y.reshape(1309,1)
array([[7.25],
[71.2833],
[7.925],
...,
[7.25],
[8.05],
[22.3583]])
[13] #reg = LinearRegression().fit(x, y)
reg = LinearRegression().fit(X, Y)
[14] reg.predict(np.array([[123]]))
array([100.07450092])
[15] reg.score(X, Y)
0.031748839685851626
[16] plt.scatter(df['Age'], df['Fare'])
Z = reg.predict(X)
plt.scatter(df['Age'], Z)
<matplotlib.collections.PathCollection at 0x7f0bf1343350>
500
0s completed at 18:41
Type here to search
19 March 2021
Friday
06:44 PM
19-03-2021

https://colab.research.google.com/drive/1aunjULkr2KbRwGwXLWw9t5YZF9tiOgv6#scrollTo=McZ6zIKqLLsI

Apps Gmail YouTube Maps Translate Reading list

P4_Linear Regression_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

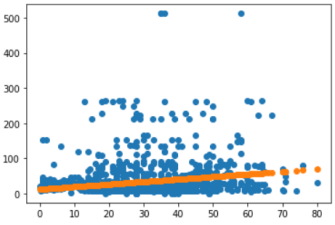
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

RAM Disk

+ Code + Text

```
[16] plt.scatter(df['Age'], df['Fare'])
     Z = reg.predict(X)
     plt.scatter(df['Age'], Z)
```

```
<matplotlib.collections.PathCollection at 0x7f0bf1343350>
```



Linear Regression and GRAPH

- Here, we're performing basic Linear Regression as performed above and
- Also plotting a 3D Graph where more than one variable is used.
- Here, we've taken AGE, SEX and FARE

```
[17] # More than one variables involved here

     X1 = np.array(df[['Age','Sex']])
```

0s completed at 18:41

Type here to search

ENG 06:44 PM 19-03-2021

19 March 2021 Friday

https://colab.research.google.com/drive/1aunjULkr2KbRwGwXLWw9t5YZF9tiOgv6#scrollTo=McZ6zIKqLLsI

Apps Gmail YouTube Maps Translate Reading list

P4_Linear Regression_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com

RAM Disk

+ Code + Text

Linear Regression and GRAPH

- Here, we're performing basic Linear Regression as performed above and
- Also plotting a 3D Graph where more than one variable is used.
- Here, we've taken AGE, SEX and FARE

```
[17] # More than one variables involved here

     X1 = np.array(df[['Age','Sex']])

     Y1 = np.array([df['Fare']])

     Y1= Y1.reshape(1309,1)
     reg = LinearRegression().fit(X1,Y1)
     print(reg.score(X1, Y1))
     Z1 = reg.predict(X1)
     fig = plt.figure()

     ax = fig.add_subplot(111, projection='3d')
     ax.scatter(df['Age'],df['Sex'], Y1)
     ax.scatter(df['Age'],df['Sex'],Z1 )

     df['Predicted_fare'] = Z1

     # plt.scatter(df['Sex'],Z1 )
```

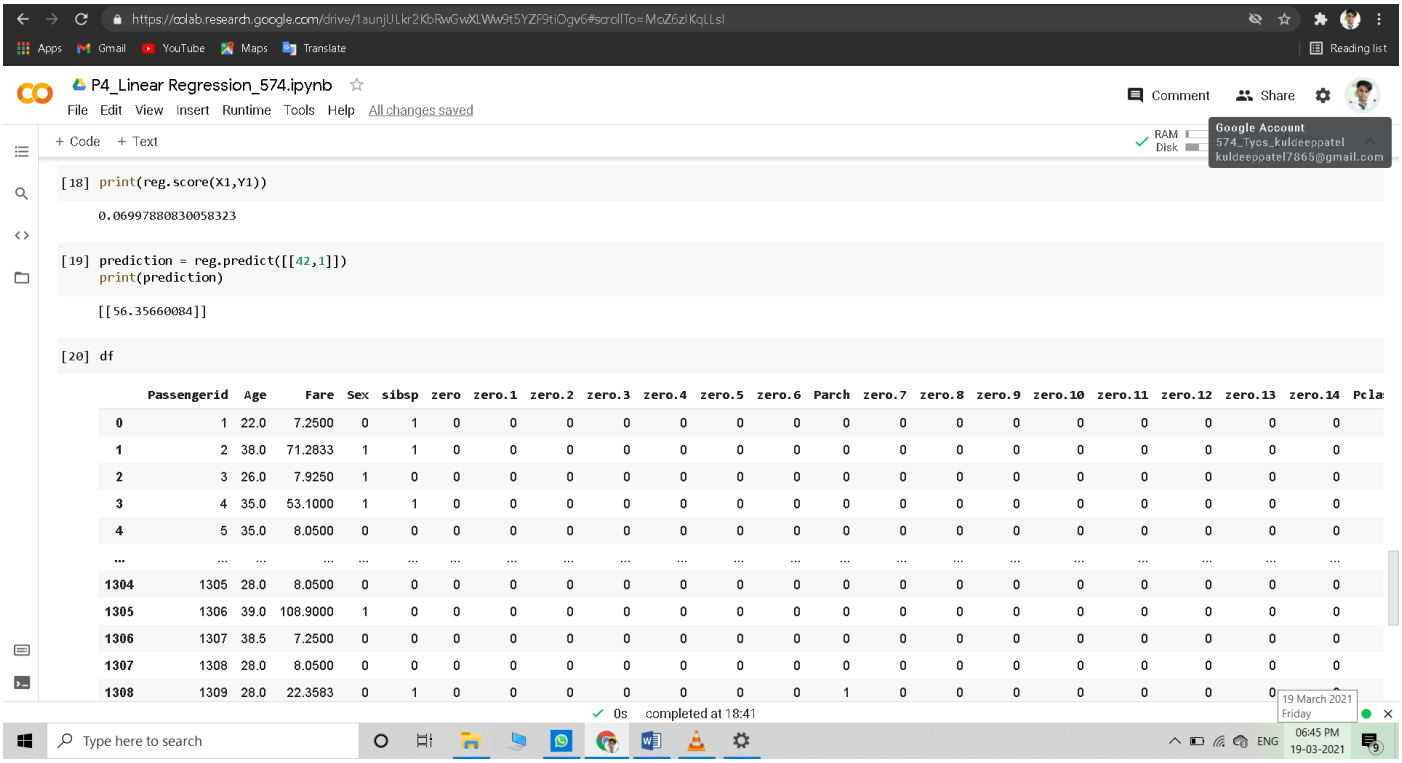
```
0.06997880830058323
```

0s completed at 18:41

Type here to search

ENG 06:44 PM 19-03-2021

19 March 2021 Friday



Colab Link :

https://colab.research.google.com/drive/1aunjULkr2KbRwGwXLWw9t5YZF9tiOgv6?usp=sharing

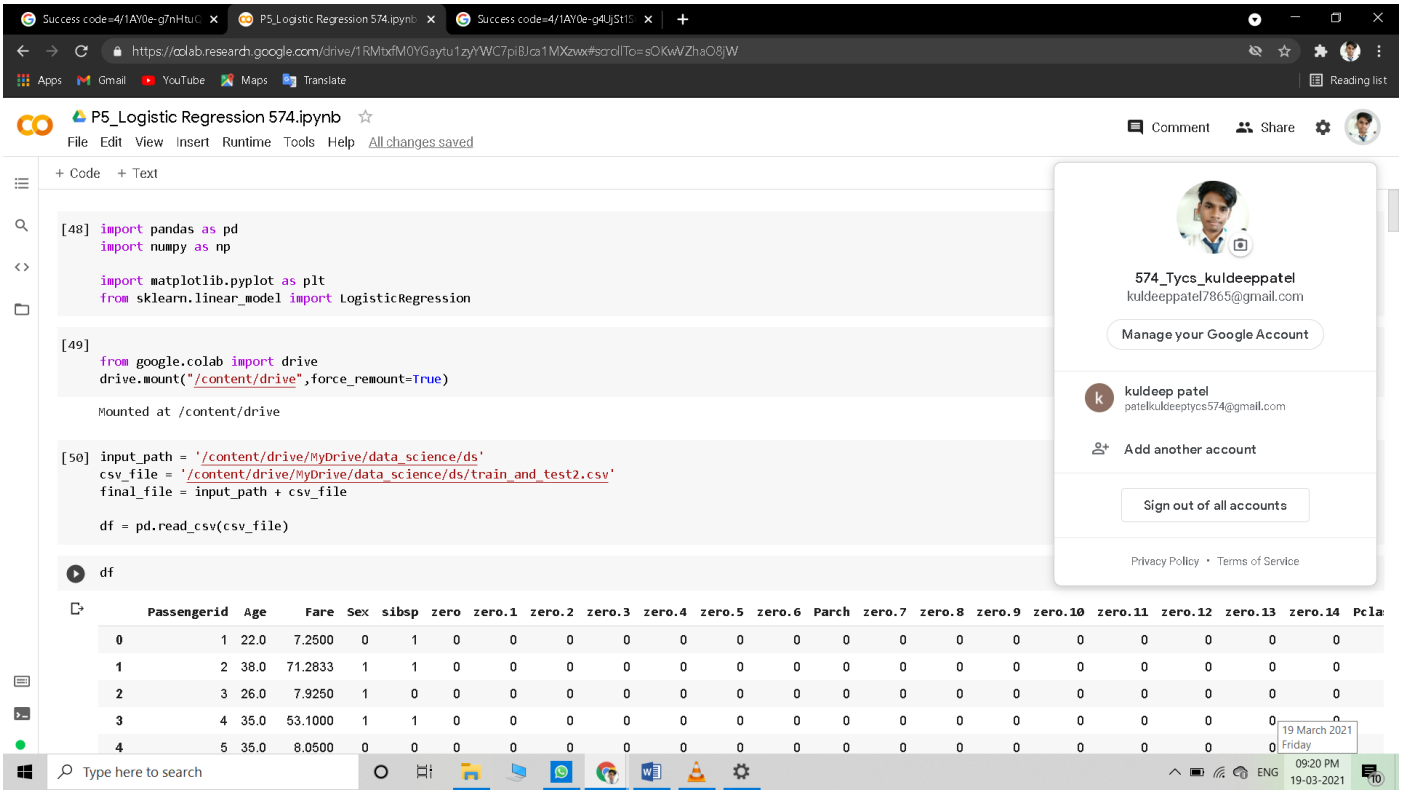
Practical – 5

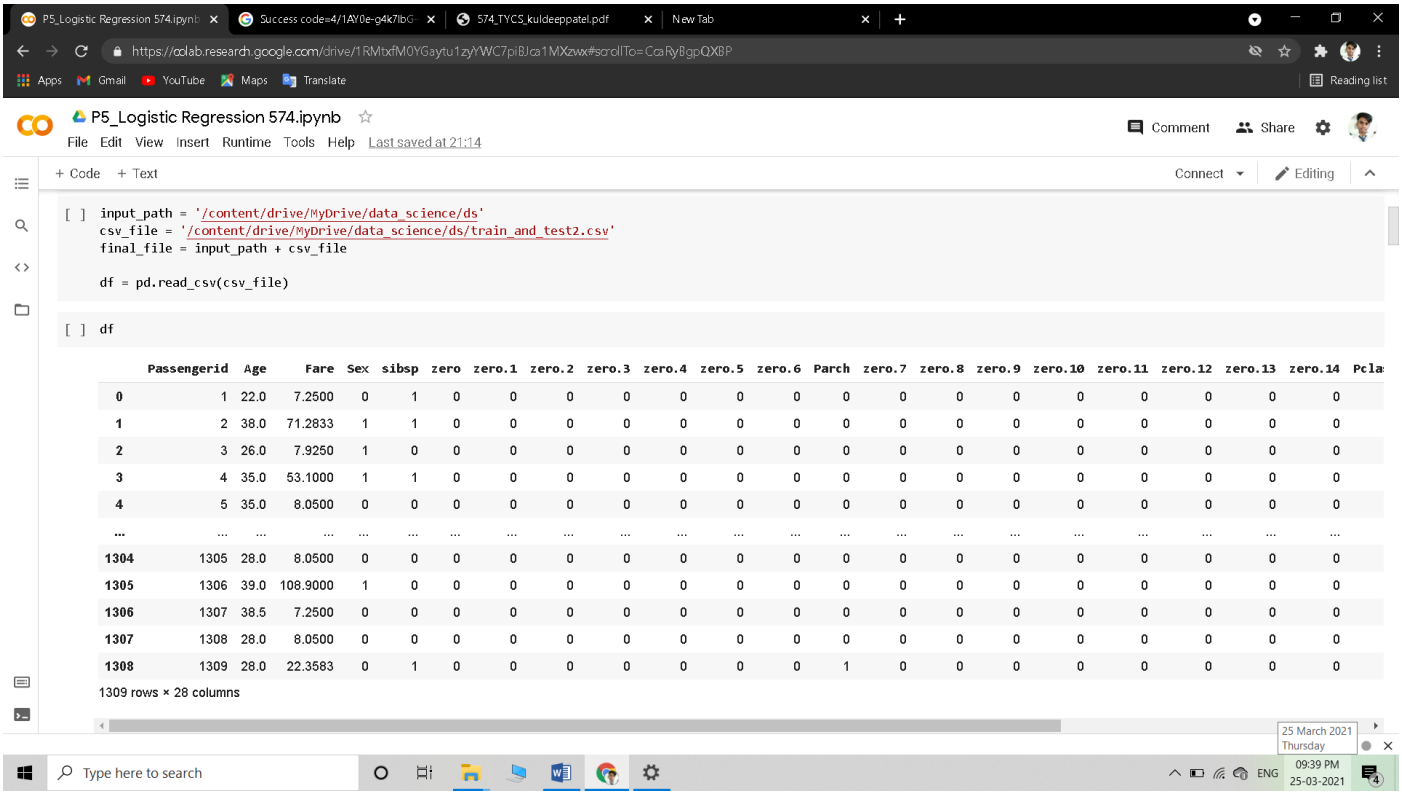
Aim : Practical of Logistic Regression

Theory :

- Logistic regression (LR) is a statistical method similar to linear regression since LR finds an equation that predicts an outcome for a binary variable, Y, from one or more response variables, X.
- However, unlike linear regression, the response variables can be categorical or continuous, as the model does not strictly require continuous data.
- To predict group membership, LR uses the log odds ratio rather than probabilities and an iterative maximum likelihood method rather than least squares to fit the final model.
- Logistic regression assumes independence among variables, which is not always met in morphotropic datasets Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Like all regression analyses, logistic regression is a predictive analysis.
- Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval, or ratio level independent variables.

Code :





P5_Logistic Regression 574.ipynb

File Edit View Insert Runtime Tools Help Last saved at 21:14

Comment Share

+ Code + Text

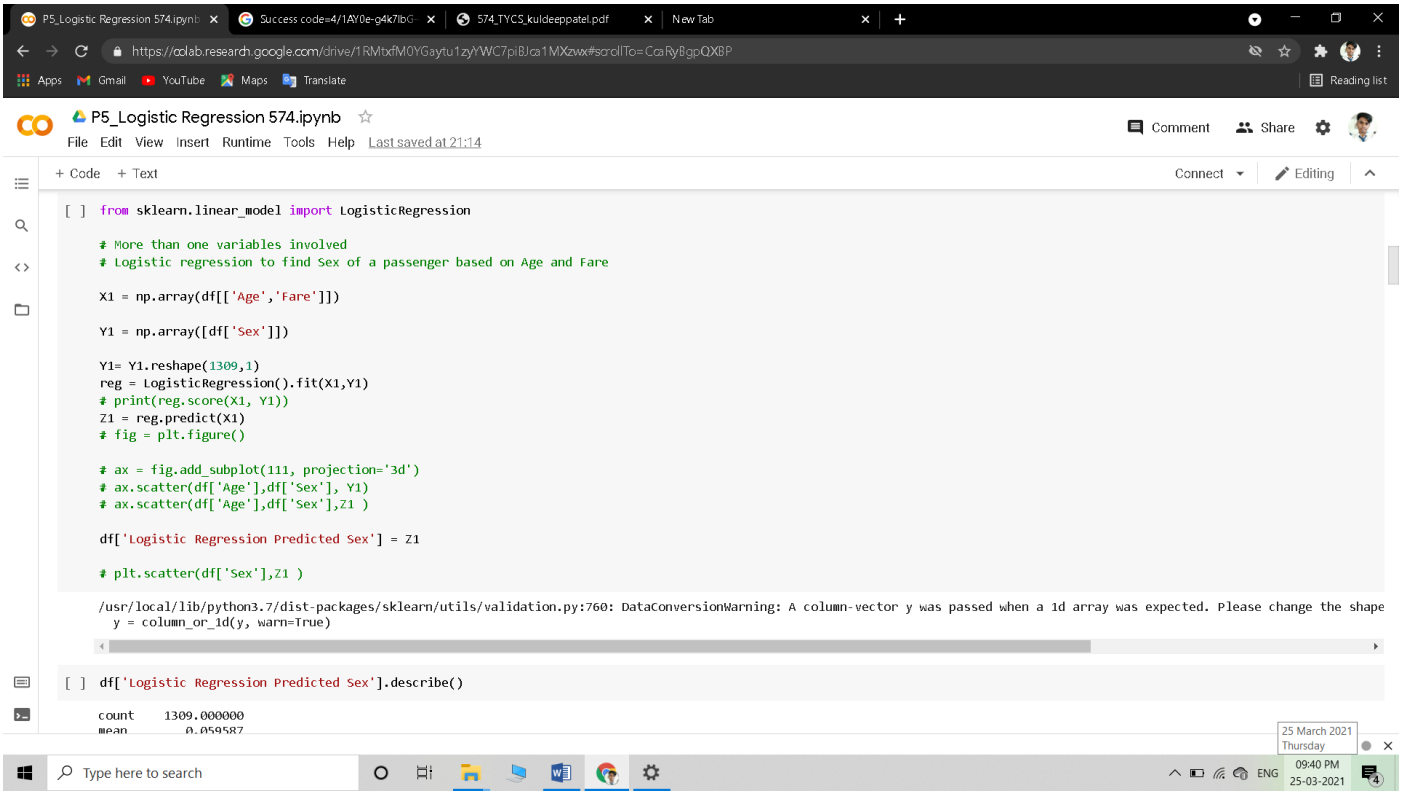
Connect Editing

```
[ ] input_path = '/content/drive/MyDrive/data_science/ds'
    csv_file = '/content/drive/MyDrive/data_science/ds/train_and_test2.csv'
    final_file = input_path + csv_file

    df = pd.read_csv(csv_file)
```

```
[ ] df
```

	Passengerid	Age	Fare	Sex	sibsp	zero	zero.1	zero.2	zero.3	zero.4	zero.5	zero.6	Parch	zero.7	zero.8	zero.9	zero.10	zero.11	zero.12	zero.13	zero.14	Pcla
0	1	22.0	7.2500	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	2	38.0	71.2833	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	3	26.0	7.9250	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	4	35.0	53.1000	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	5	35.0	8.0500	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1304	1305	28.0	8.0500	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1305	1306	39.0	108.9000	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1306	1307	38.5	7.2500	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1307	1308	28.0	8.0500	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1308	1309	28.0	22.3583	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	

1309 rows × 28 columns



P5_Logistic Regression 574.ipynb

File Edit View Insert Runtime Tools Help Last saved at 21:14

Comment Share

+ Code + Text

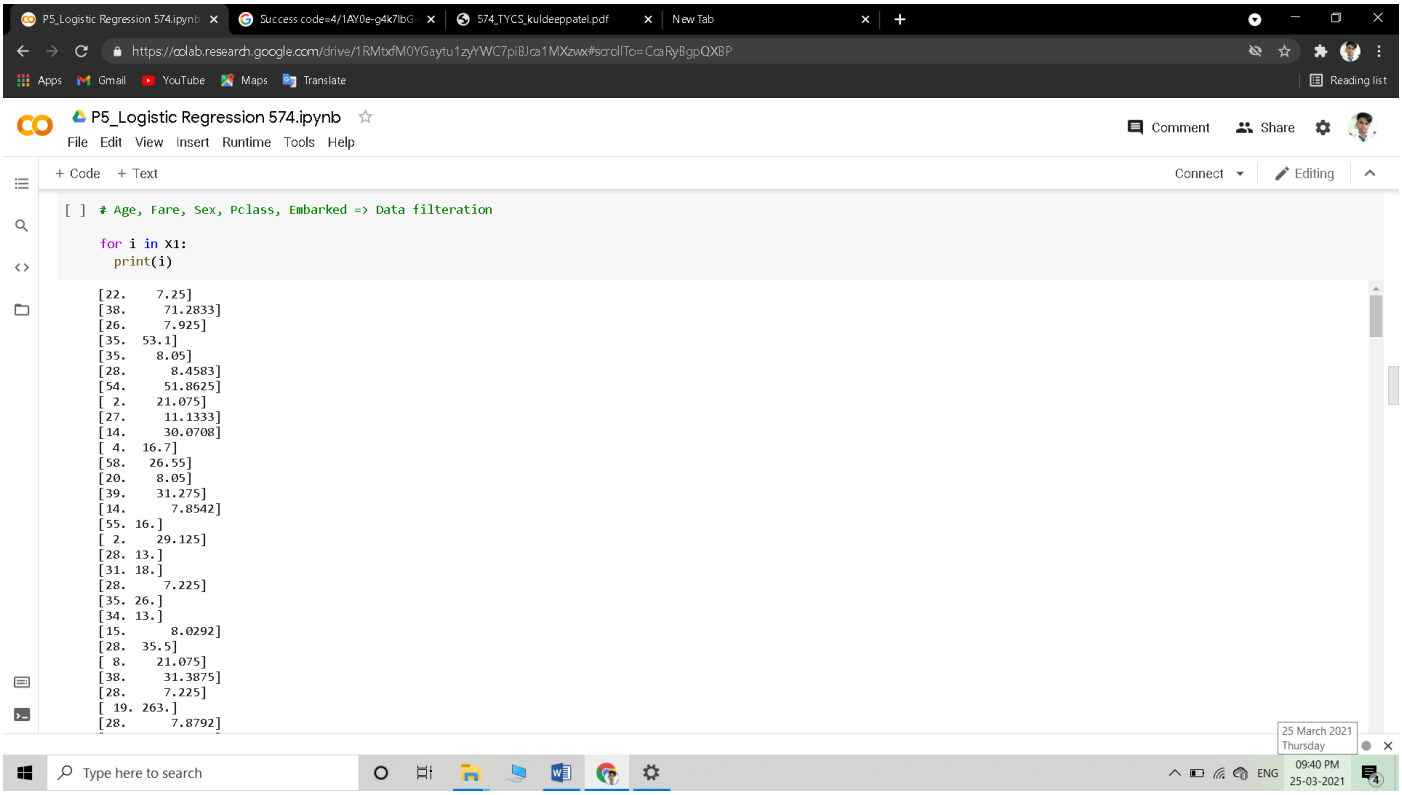
Connect Editing

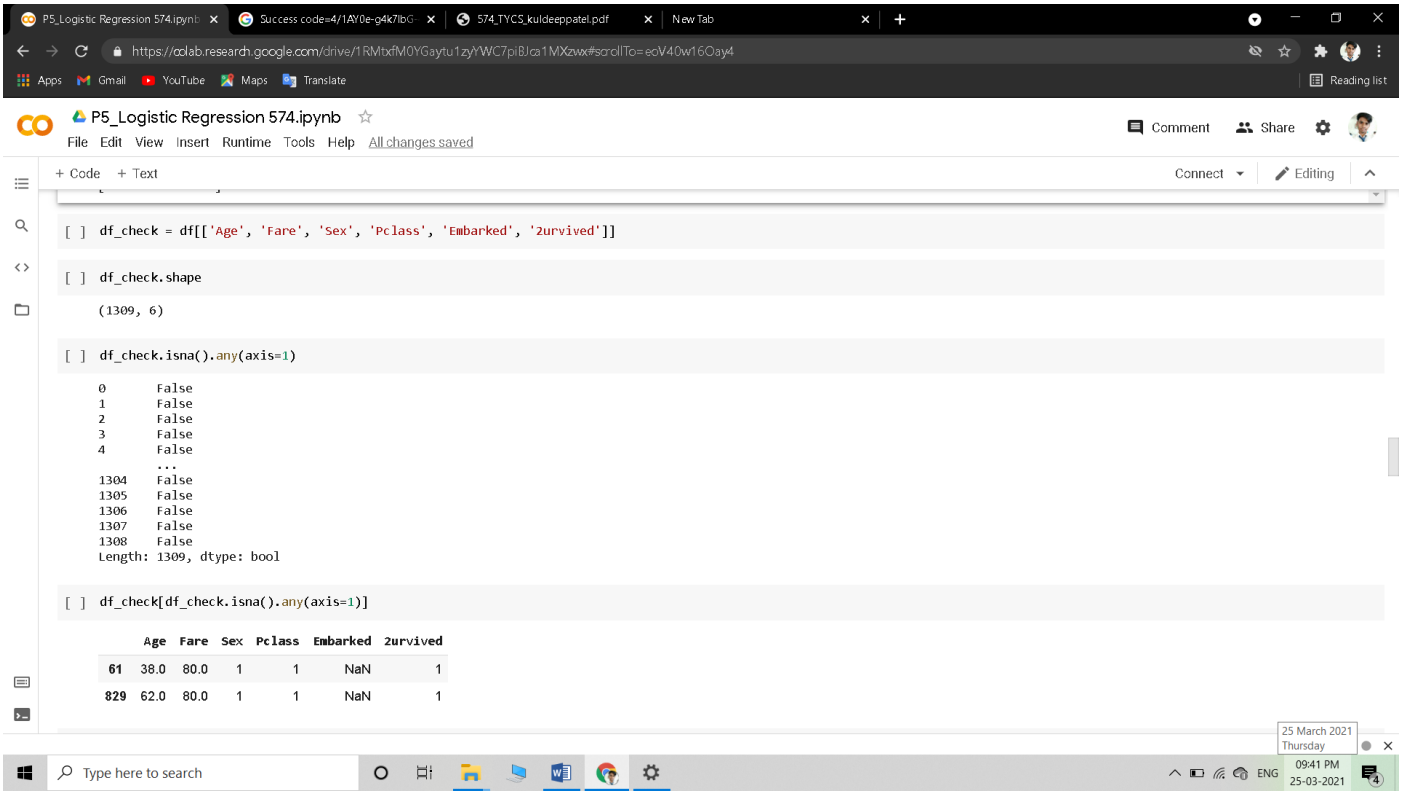
```
[ ] from sklearn.linear_model import LogisticRegression

    # More than one variables involved
    # Logistic regression to find Sex of a passenger based on Age and Fare

    X1 = np.array(df[['Age','Fare']])

    Y1 = np.array([df['Sex']])

    Y1= Y1.reshape(1309,1)
    reg = LogisticRegression().fit(X1,Y1)
    # print(reg.score(X1, Y1))
    Z1 = reg.predict(X1)
    # fig = plt.figure()

    # ax = fig.add_subplot(111, projection='3d')
    # ax.scatter(df['Age'],df['Sex'], Y1)
    # ax.scatter(df['Age'],df['Sex'],Z1 )

    df['Logistic Regression Predicted Sex'] = Z1

    # plt.scatter(df['Sex'],Z1 )
```

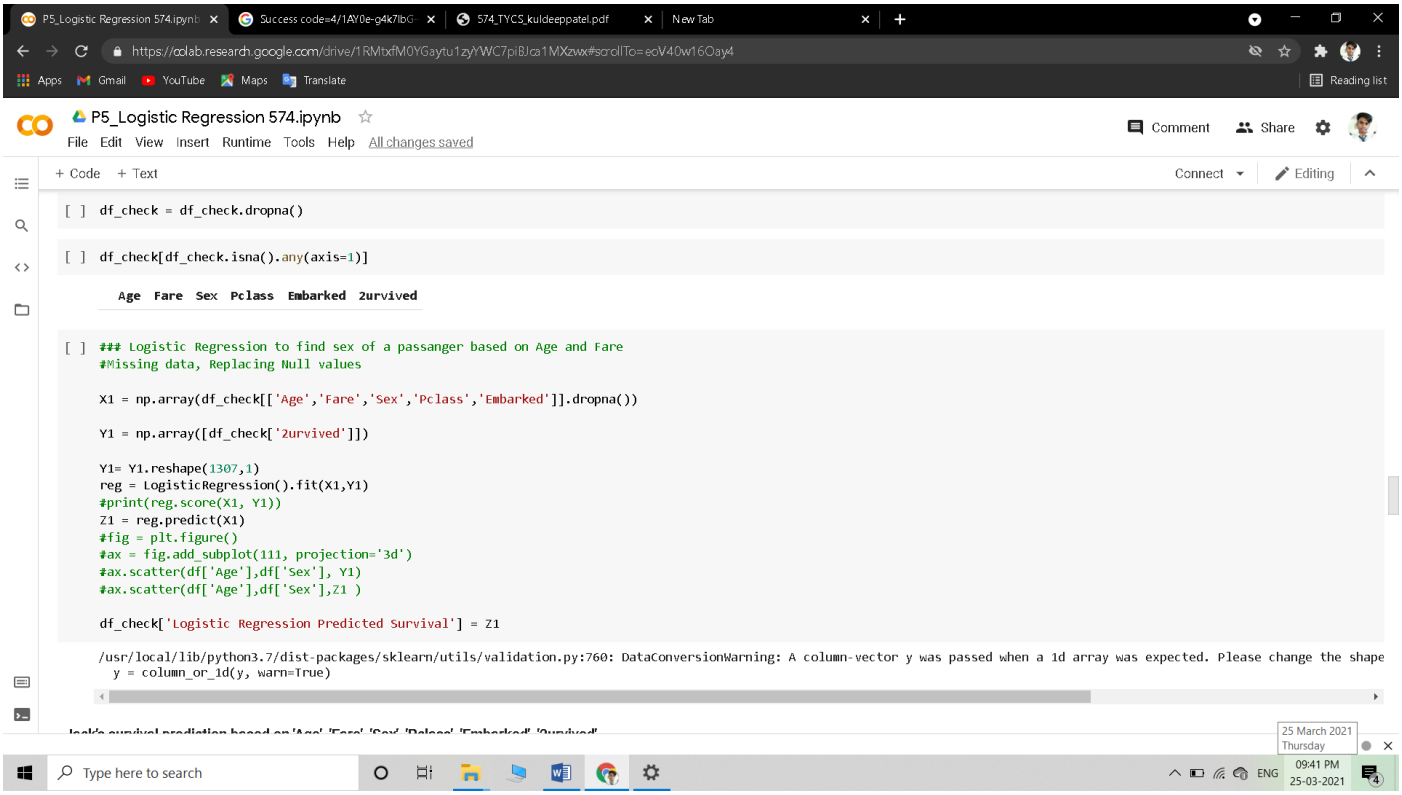
```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape
  y = column_or_1d(y, warn=True)
```

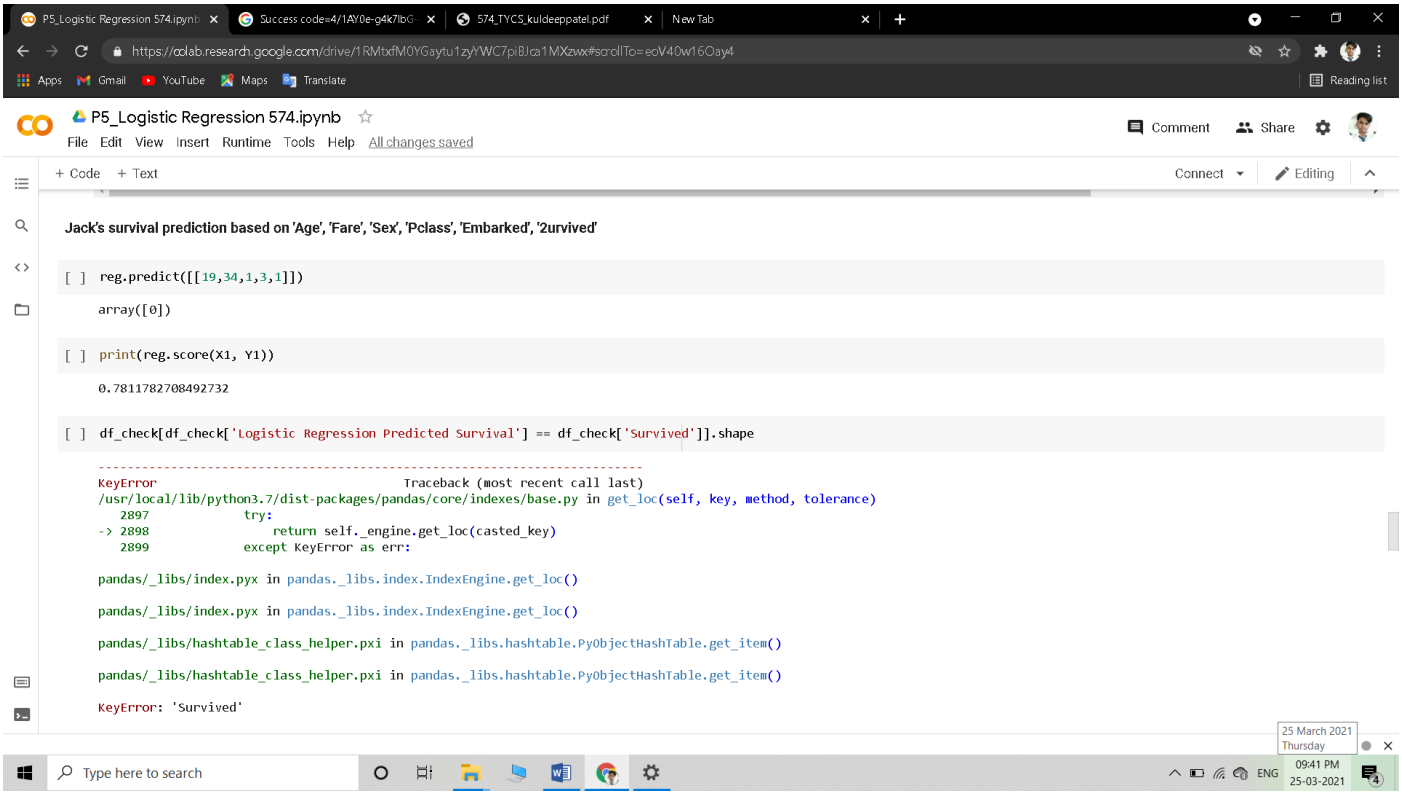
```
[ ] df['Logistic Regression Predicted Sex'].describe()
```

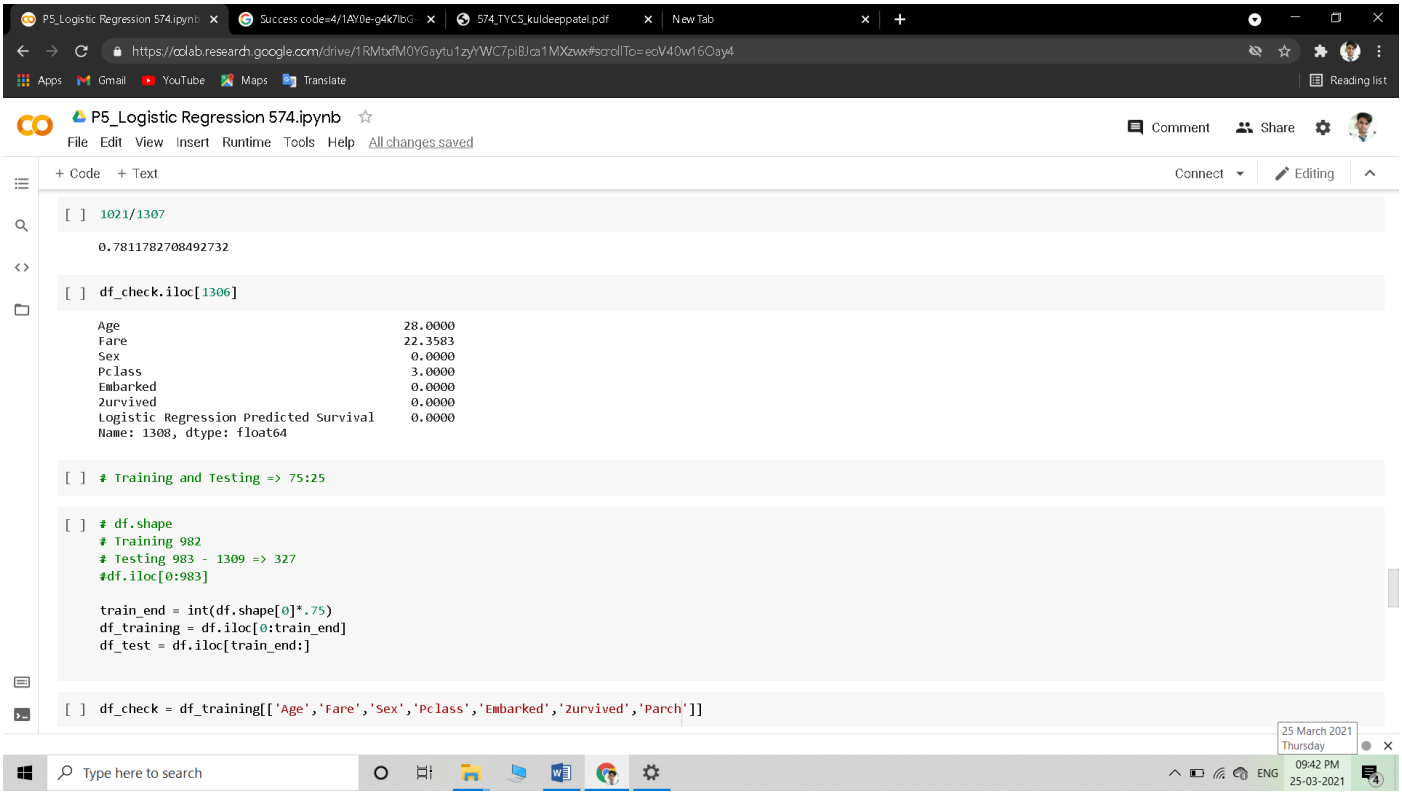
```
count    1309.000000
mean        0.059587
```

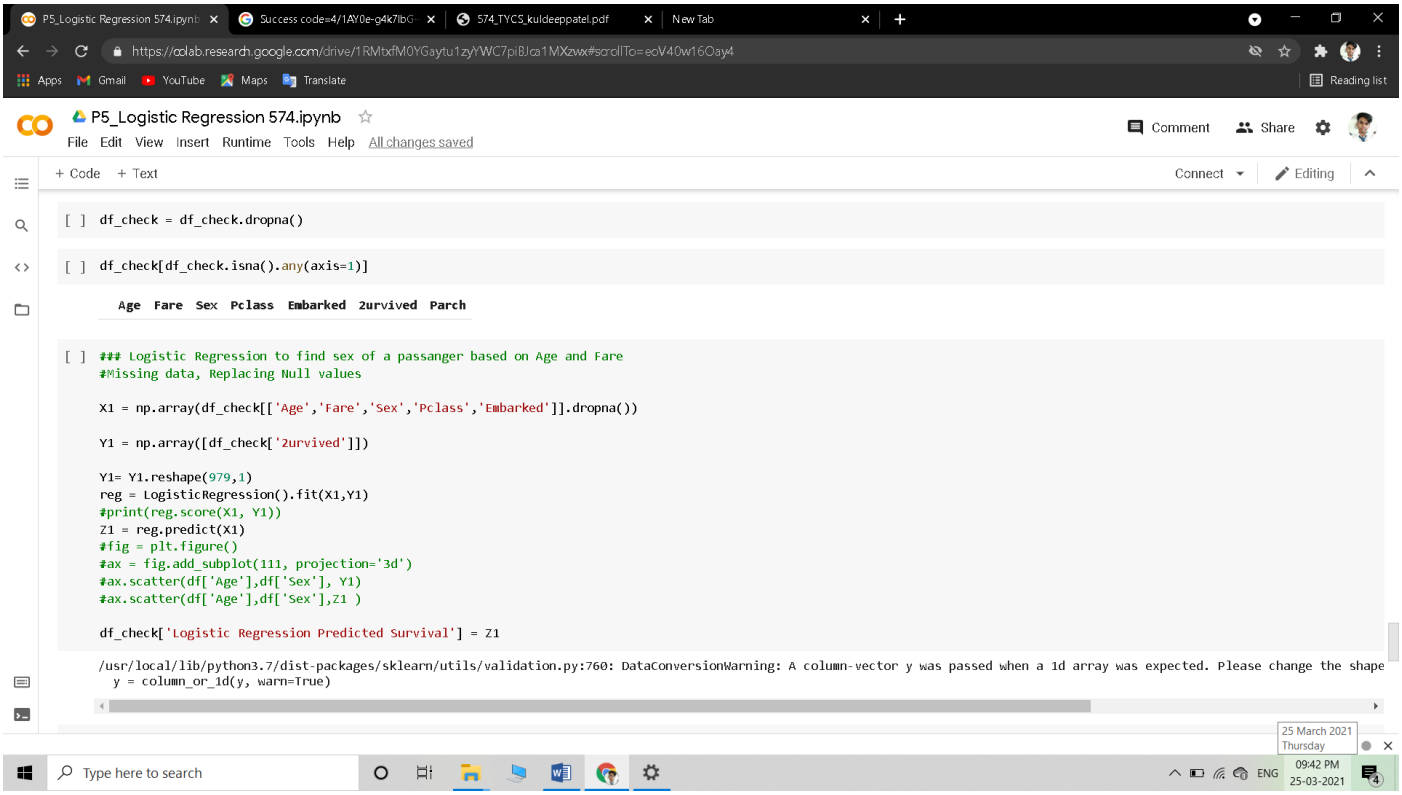

P5_Logistic Regression 574.ipynb
Success code=4/1AY0e-g4k7lbG
574_TYCS_kuldeeppatel.pdf
New Tab
https://colab.research.google.com/drive/1RMtxfM0YGaytu1zyYWC7piBJca1MXzwx#scrollTo=CcaRyBgpQXBP
Apps Gmail YouTube Maps Translate Reading list
P5_Logistic Regression 574.ipynb
File Edit View Insert Runtime Tools Help
Comment Share
+ Code + Text
Connect Editing
[ ] # Age, Fare, Sex, Pclass, Embarked => Data filteration
for i in X1:
print(i)
[22. 7.25]
[38. 71.2833]
[26. 7.925]
[35. 53.1]
[35. 8.05]
[28. 8.4583]
[54. 51.8625]
[2. 21.075]
[27. 11.1333]
[14. 30.0708]
[4. 16.7]
[58. 26.55]
[20. 8.05]
[39. 31.275]
[14. 7.8542]
[55. 16.]
[2. 29.125]
[28. 13.]
[31. 18.]
[28. 7.225]
[35. 26.]
[34. 13.]
[15. 8.0292]
[28. 35.5]
[8. 21.075]
[38. 31.3875]
[28. 7.225]
[19. 263.]
[28. 7.8792]
25 March 2021 Thursday
Type here to search
ENG 09:40 PM 25-03-2021

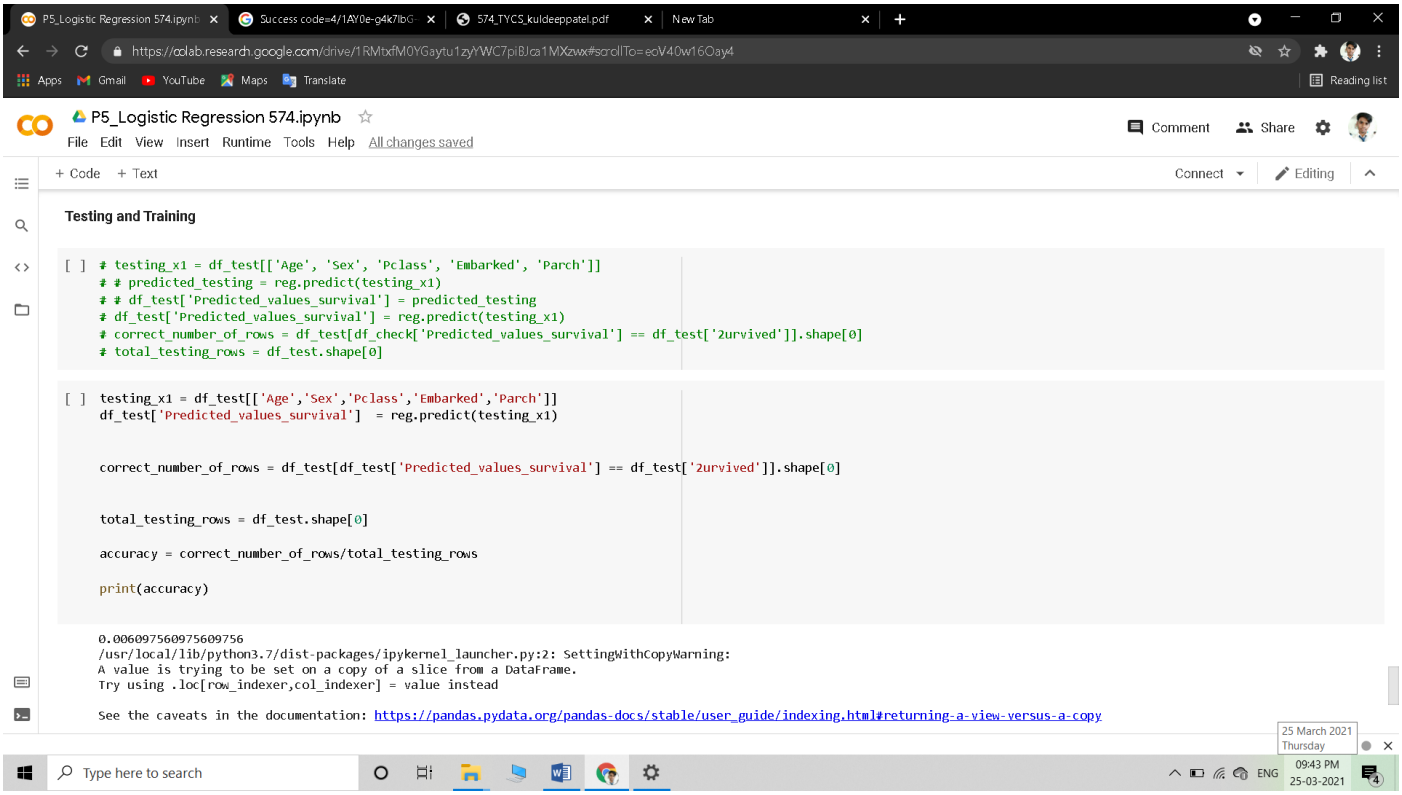

P5_Logistic Regression 574.ipynb
Success code=4/1AY0e-g4k7lbG
574_TYCS_kuldeeppatel.pdf
New Tab
https://colab.research.google.com/drive/1RMtxfM0YGaytu1zyYWC7piBJca1MXzwx#scrollTo=ecV40w16Oay4
Apps Gmail YouTube Maps Translate Reading list
P5_Logistic Regression 574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Connect Editing
[ ] df_check = df[['Age', 'Fare', 'Sex', 'Pclass', 'Embarked', '2urvived']]
[ ] df_check.shape
(1309, 6)
[ ] df_check.isna().any(axis=1)
0 False
1 False
2 False
3 False
4 False
...
1304 False
1305 False
1306 False
1307 False
1308 False
Length: 1309, dtype: bool
[ ] df_check[df_check.isna().any(axis=1)]
Age Fare Sex Pclass Embarked 2urvived
61 38.0 80.0 1 1 NaN 1
829 62.0 80.0 1 1 NaN 1
25 March 2021 Thursday
Type here to search
ENG 09:41 PM 25-03-2021


P5_Logistic Regression 574.ipynb
Success code=4/1AY0e-g4k7lbG
574_TYCS_kuldeeppatel.pdf
New Tab
https://colab.research.google.com/drive/1RMtxfM0YGaytu1zyYWC7piBJca1MXzwx#scrollTo=ecV40w16Oay4
Apps Gmail YouTube Maps Translate Reading list
P5_Logistic Regression 574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Connect Editing
[ ] df_check = df_check.dropna()
[ ] df_check[df_check.isna().any(axis=1)]
Age Fare Sex Pclass Embarked 2urvived
[ ] ### Logistic Regression to find sex of a passanger based on Age and Fare
#Missing data, Replacing Null values
X1 = np.array(df_check[['Age','Fare','Sex','Pclass','Embarked']].dropna())
Y1 = np.array([df_check['2urvived']])
Y1= Y1.reshape(1307,1)
reg = LogisticRegression().fit(X1,Y1)
#print(reg.score(X1, Y1))
Z1 = reg.predict(X1)
#fig = plt.figure()
#ax = fig.add_subplot(111, projection='3d')
#ax.scatter(df['Age'],df['Sex'], Y1)
#ax.scatter(df['Age'],df['Sex'],Z1)
df_check['Logistic Regression Predicted Survival'] = Z1
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape
y = column_or_1d(y, warn=True)
Type here to search
25 March 2021 Thursday
09:41 PM 25-03-2021


P5_Logistic Regression 574.ipynb
Success code=4/1AY0e-g4k7lbG
574_TYCS_kuldeeppatel.pdf
New Tab
https://colab.research.google.com/drive/1RMtxfM0YGaytu1zyYWC7piBJca1MXzwx#scrollTo=ecV40w16Oay4
Apps Gmail YouTube Maps Translate Reading list
P5_Logistic Regression 574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Connect Editing
Jack's survival prediction based on 'Age', 'Fare', 'Sex', 'Pclass', 'Embarked', '2urvived'
[ ] reg.predict([[19,34,1,3,1]])
array([0])
[ ] print(reg.score(X1, Y1))
0.7811782708492732
[ ] df_check[df_check['Logistic Regression Predicted Survival'] == df_check['Survived']].shape
KeyError Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self, key, method, tolerance)
2897 try:
-> 2898 return self._engine.get_loc(casted_key)
2899 except KeyError as err:
pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()
pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()
pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()
pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()
KeyError: 'Survived'
Type here to search
25 March 2021 Thursday
09:41 PM 25-03-2021


P5_Logistic Regression 574.ipynb
Success code=4/1AY0e-g4k7lbG
574_TYCS_kuldeeppatel.pdf
New Tab
https://colab.research.google.com/drive/1RMtxfM0YGaytu1zyYWC7piBJca1MXzwx#scrollTo=ecV40w16Oay4
Apps Gmail YouTube Maps Translate Reading list
P5_Logistic Regression 574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Connect Editing
[ ] 1021/1307
0.7811782708492732
[ ] df_check.iloc[1306]
Age 28.0000
Fare 22.3583
Sex 0.0000
Pclass 3.0000
Embarked 0.0000
2urvived 0.0000
Logistic Regression Predicted Survival 0.0000
Name: 1308, dtype: float64
[ ] # Training and Testing => 75:25
[ ] # df.shape
Training 982
Testing 983 - 1309 => 327
#df.iloc[0:983]
train_end = int(df.shape[0]*.75)
df_training = df.iloc[0:train_end]
df_test = df.iloc[train_end:]
[ ] df_check = df_training[['Age','Fare','Sex','Pclass','Embarked','2urvived','Parch']]
25 March 2021 Thursday
Type here to search
ENG 09:42 PM 25-03-2021


P5_Logistic Regression 574.ipynb
Success code=4/1AY0e-g4k7lbG
574_TYCS_kuldeeppatel.pdf
New Tab
https://colab.research.google.com/drive/1RMtxfM0YGaytu1zyYWC7piBJca1MXzwx#scrollTo=ecV40w16Oay4
Apps Gmail YouTube Maps Translate Reading list
P5_Logistic Regression 574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Connect Editing
[ ] df_check = df_check.dropna()
[ ] df_check[df_check.isna().any(axis=1)]
Age Fare Sex Pclass Embarked 2urvived Parch
[ ] ### Logistic Regression to find sex of a passanger based on Age and Fare
#Missing data, Replacing Null values
X1 = np.array(df_check[['Age','Fare','Sex','Pclass','Embarked']].dropna())
Y1 = np.array([df_check['2urvived']])
Y1= Y1.reshape(979,1)
reg = LogisticRegression().fit(X1,Y1)
#print(reg.score(X1, Y1))
Z1 = reg.predict(X1)
#fig = plt.figure()
#ax = fig.add_subplot(111, projection='3d')
#ax.scatter(df['Age'],df['Sex'], Y1)
#ax.scatter(df['Age'],df['Sex'],Z1)
df_check['Logistic Regression Predicted Survival'] = Z1
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape
y = column_or_1d(y, warn=True)
25 March 2021 Thursday
Type here to search
ENG 09:42 PM 25-03-2021



Colab Link :

https://colab.research.google.com/drive/1RMtxfM0YGaytu1zyYWC7piBJca1MXzwx?usp=sharing

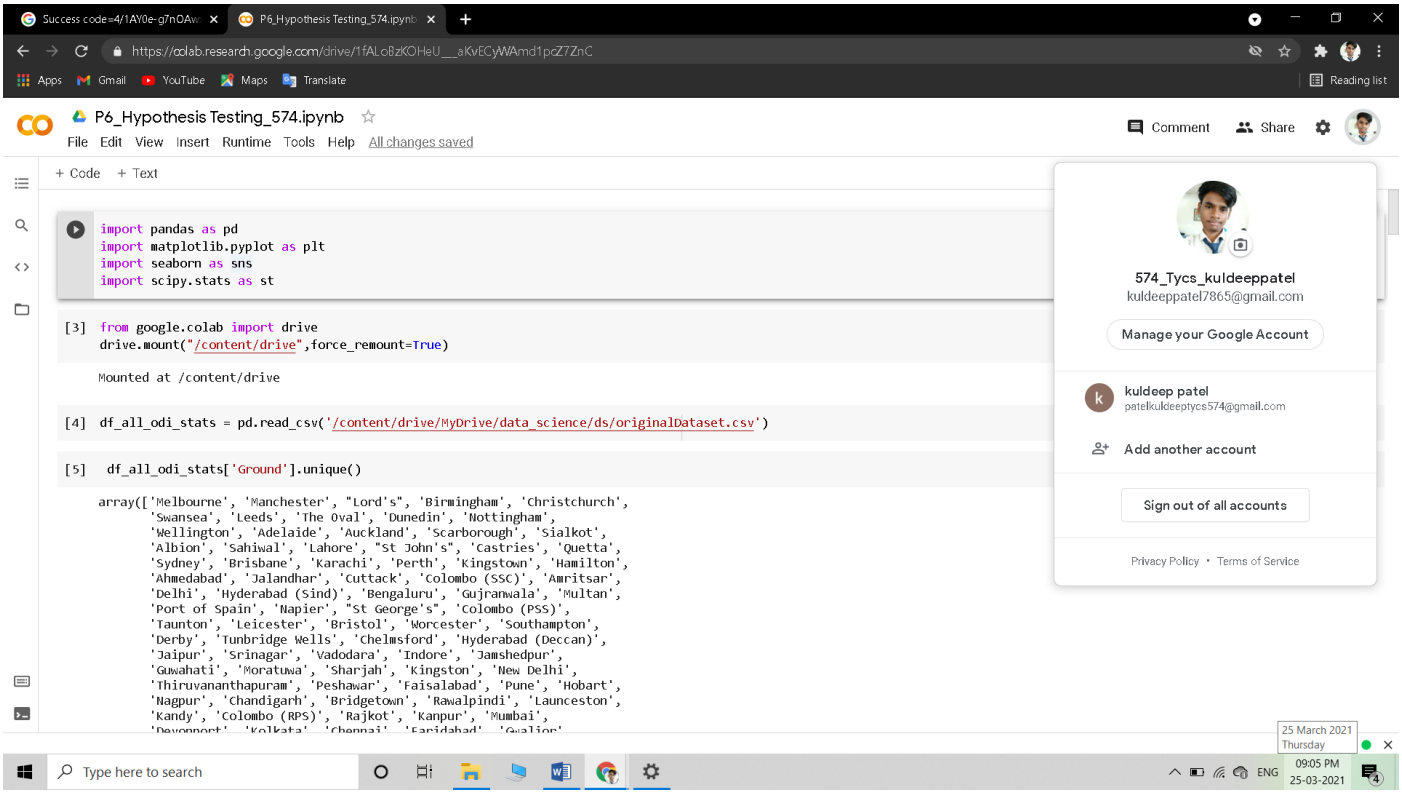
Practical – 6

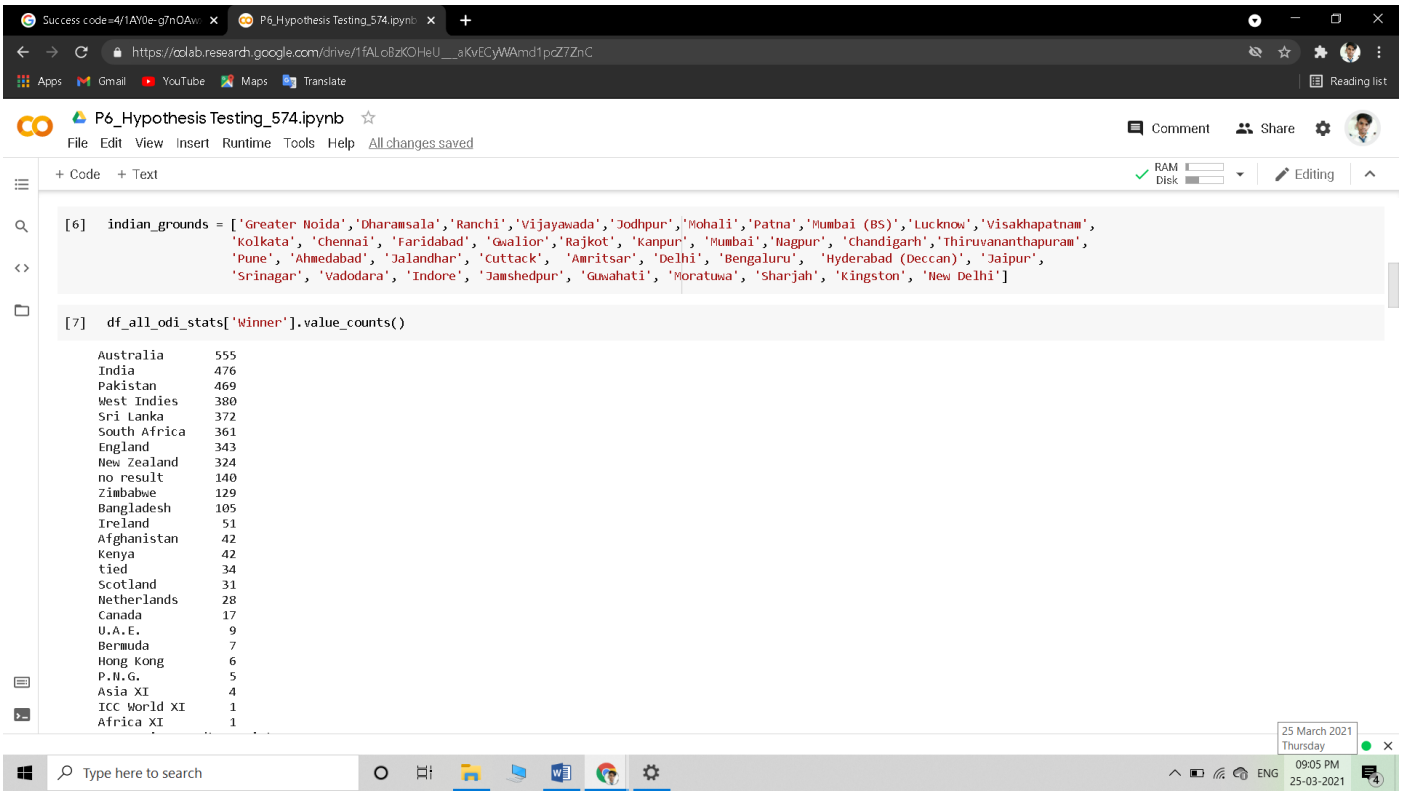
Aim : Practical of Hypothesis Testing

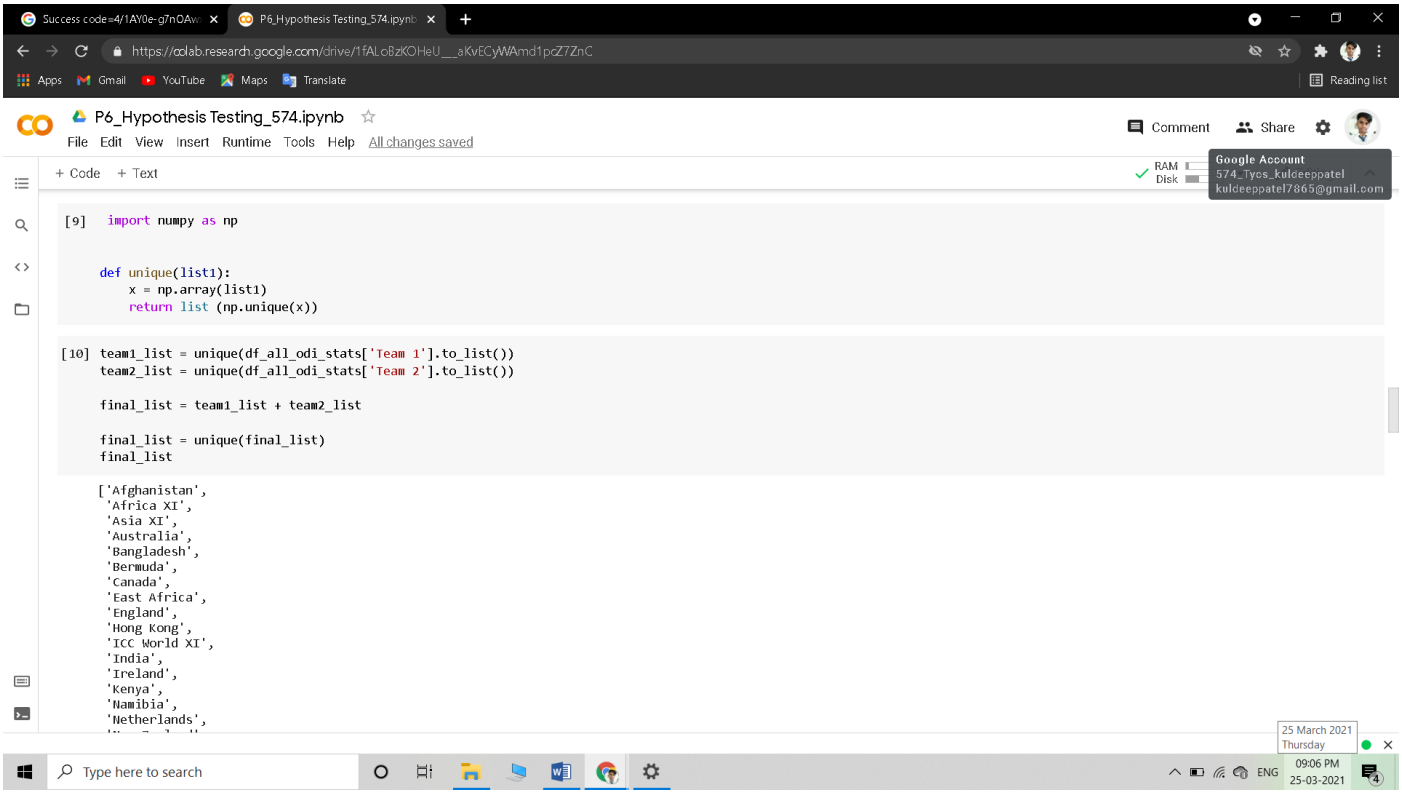
Theory :

- Hypothesis Testing is necessary for almost every sector; it does not limit to Statisticians or Data Scientists.
- For example, if we develop a code we perform testing too. In the same way, for every product or problem that an organization shows, it has to be solved by providing assumptions.
- This can be done using “Hypothesis Testing”. The hypothesis is described as a recommended solution for an undefinable incident that doesn’t into current theory.
- The actual definition of Hypothesis Testing is by which an analyst tests an assumption regarding a population parameter
- The methodology retained by the analyst depends on the nature of the data used and the reason for the analysis.
- Types of hypothesis:
  There are two sorts of hypotheses and both the Null Hypothesis (Ho) and Alternative Hypothesis (Ha) must be totally mutually exclusive events.
- The null hypothesis is usually the hypothesis that the event won't happen.
- An alternative hypothesis is a hypothesis that the event will happen.

Code :




P6_Hypothesis Testing_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
[6] indian_grounds = ['Greater Noida','Dharamsala','Ranchi','Vijayawada','Jodhpur','Mohali','Patna','Mumbai (BS)','Lucknow','Visakhapatnam',
'Kolkata', 'Chennai', 'Faridabad', 'Gwalior','Rajkot', 'Kanpur', 'Mumbai','Nagpur', 'Chandigarh','Thiruvananthapuram',
'Pune', 'Ahmedabad', 'Jalandhar', 'Cuttack', 'Amritsar', 'Delhi', 'Bengaluru', 'Hyderabad (Deccan)', 'Jaipur',
'Srinagar', 'Vadodara', 'Indore', 'Jamshedpur', 'Guwahati', 'Moratuwa', 'Sharjah', 'Kingston', 'New Delhi']
[7] df_all_odi_stats['Winner'].value_counts()
Australia 555
India 476
Pakistan 469
West Indies 380
Sri Lanka 372
South Africa 361
England 343
New Zealand 324
no result 140
Zimbabwe 129
Bangladesh 105
Ireland 51
Afghanistan 42
Kenya 42
tied 34
Scotland 31
Netherlands 28
Canada 17
U.A.E. 9
Bermuda 7
Hong Kong 6
P.N.G. 5
Asia XI 4
ICC World XI 1
Africa XI 1
Type here to search
25 March 2021
Thursday
09:05 PM
25-03-2021


P6_Hypothesis Testing_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com
[9] import numpy as np
def unique(list1):
x = np.array(list1)
return list (np.unique(x))
[10] team1_list = unique(df_all_odi_stats['Team 1'].to_list())
team2_list = unique(df_all_odi_stats['Team 2'].to_list())
final_list = team1_list + team2_list
final_list = unique(final_list)
final_list
['Afghanistan',
'Africa XI',
'Asia XI',
'Australia',
'Bangladesh',
'Bermuda',
'Canada',
'East Africa',
'England',
'Hong Kong',
'ICC World XI',
'India',
'Ireland',
'Kenya',
'Namibia',
'Netherlands',
Type here to search
25 March 2021
Thursday
09:06 PM
25-03-2021

Success code=4/1AY0e-g7nOAw | P6_Hypothesis Testing_574.ipynb

https://colab.research.google.com/drive/1fALoBzKOHeU__aKvECyWAmd1pcZ7ZnC

Apps Gmail YouTube Maps Translate Reading list

P6_Hypothesis Testing_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Google Account
574_Tycs_kulceeppatel
kulceeppatel7865@gmail.com

+ Code + Text

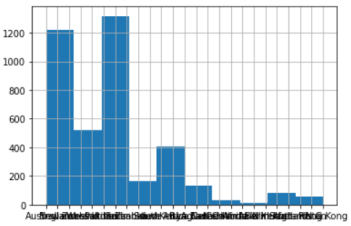
RAM Disk

```
[11]  df_all_odi_stats['Winner'].value_counts().keys()[0]
```

```
'Australia'
```

```
[12] df_all_odi_stats['Winner'].hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc230f78810>
```




```
[13] own_ground = indian_grounds
```

```
[14]  Indian_Grounds=pd.DataFrame(indian_grounds)
      Indian_Grounds.columns= ['GROUNDS']
      Indian_Grounds
```

	GROUNDS

25 March 2021 Thursday

Type here to search

ENG 09:06 PM 25-03-2021

Success code=4/1AY0e-g7nOAw | P6_Hypothesis Testing_574.ipynb

https://colab.research.google.com/drive/1fALoBzKOHeU__aKvECyWAmd1pcZ7ZnC

Apps Gmail YouTube Maps Translate Reading list

P6_Hypothesis Testing_574.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

Google Account
574_Tycs_kulceeppatel
kulceeppatel7865@gmail.com

+ Code + Text

RAM Disk

```
[15]  import scipy.stats as st
      df_own_home_conditions = df_all_odi_stats.groupby(['Winner','Margin'])['Ground'].agg('sum')
      df_own_home_conditions = df_own_home_conditions.reset_index()
```

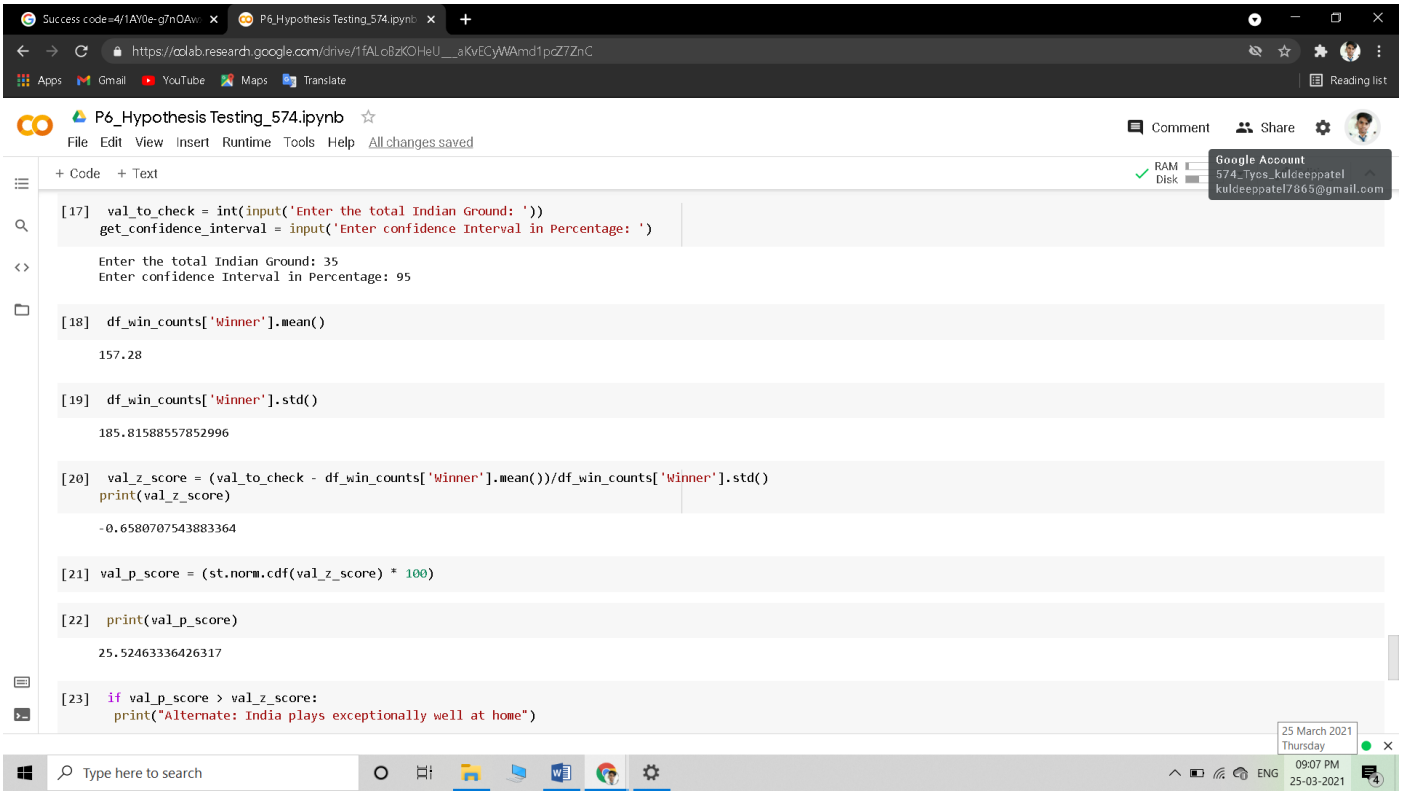
```
[16]  df_win_counts = pd.DataFrame(df_all_odi_stats['Winner'].value_counts())
      df_win_counts
```

	Winner
Australia	555
India	476
Pakistan	469
West Indies	380
Sri Lanka	372
South Africa	361
England	343
New Zealand	324
no result	140
Zimbabwe	129
Bangladesh	105
Ireland	51
Afghanistan	42
Kenya	42

25 March 2021 Thursday

Type here to search

ENG 09:07 PM 25-03-2021



Colab Link :

https://colab.research.google.com/drive/1fALoBzKOHeU___aKvECyWAmd1pcZ7ZnC?usp=sharing

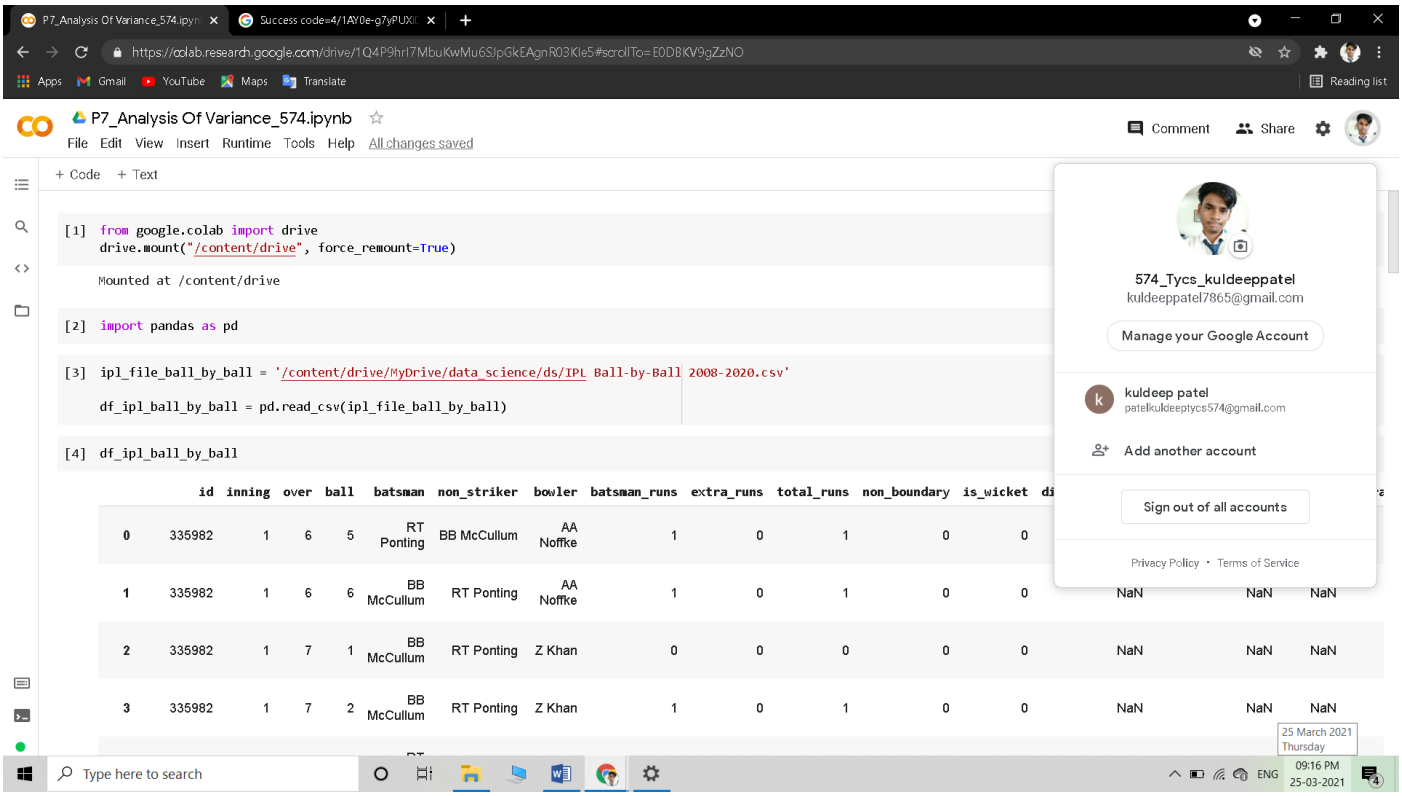
Practical – 7

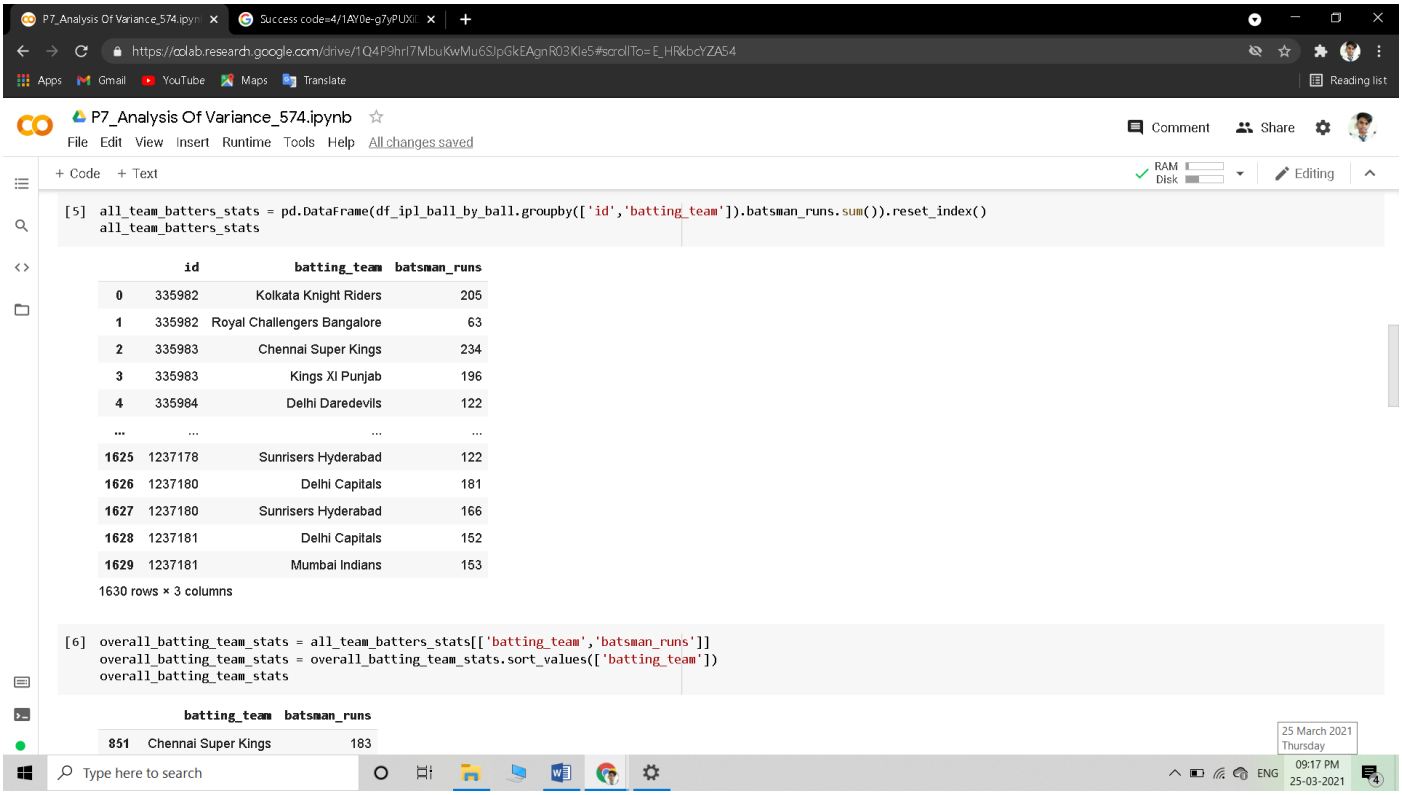
Aim : Practical of Analysis of variance

Theory :

- Analysis of variance (ANOVA) is an analysis tool used in statistics that splits an observed aggregate variability found inside a data set into two parts: systematic factors and random factors.
- The systematic factors have a statistical influence on the given data set, while the random factors do not.
- Analysts use the ANOVA test to determine the influence that independent variables have on the dependent variable in a regression study.
- ANOVA is also called the Fisher analysis of variance, and it is the extension of the t and z-tests.

Code:





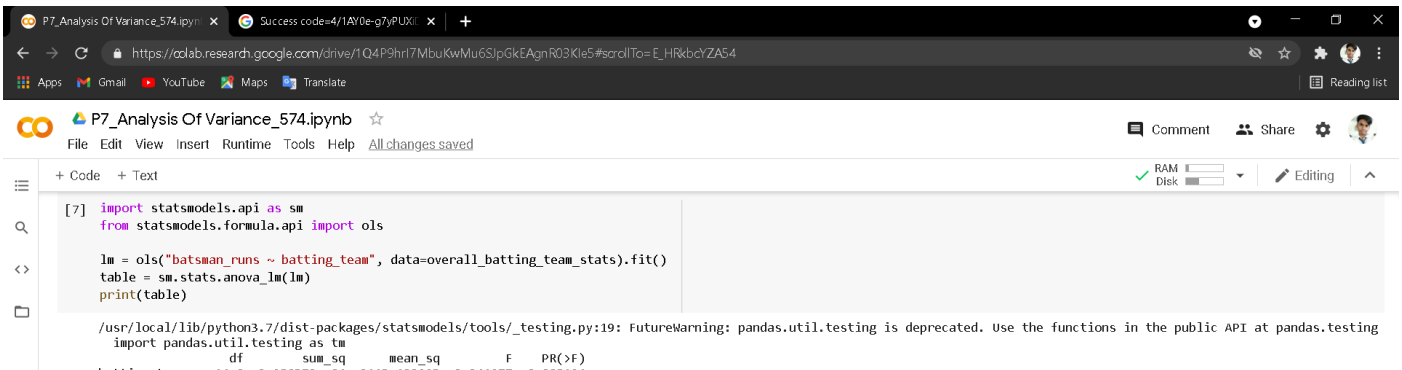
[5]
```
all_team_batters_stats = pd.DataFrame(df_ipl_ball_by_ball.groupby(['id','batting_team']).batsman_runs.sum()).reset_index()
all_team_batters_stats
```

	id	batting_team	batsman_runs
0	335982	Kolkata Knight Riders	205
1	335982	Royal Challengers Bangalore	63
2	335983	Chennai Super Kings	234
3	335983	Kings XI Punjab	196
4	335984	Delhi Daredevils	122
...	...	...	...
1625	1237178	Sunrisers Hyderabad	122
1626	1237180	Delhi Capitals	181
1627	1237180	Sunrisers Hyderabad	166
1628	1237181	Delhi Capitals	152
1629	1237181	Mumbai Indians	153

1630 rows × 3 columns

[6]
```
overall_batting_team_stats = all_team_batters_stats[['batting_team','batsman_runs']]
overall_batting_team_stats = overall_batting_team_stats.sort_values(['batting_team'])
overall_batting_team_stats
```

	batting_team	batsman_runs
851	Chennai Super Kings	183



[7]
```
import statsmodels.api as sm
from statsmodels.formula.api import ols

lm = ols("batsman_runs ~ batting_team", data=overall_batting_team_stats).fit()
table = sm.stats.anova_lm(lm)
print(table)
```

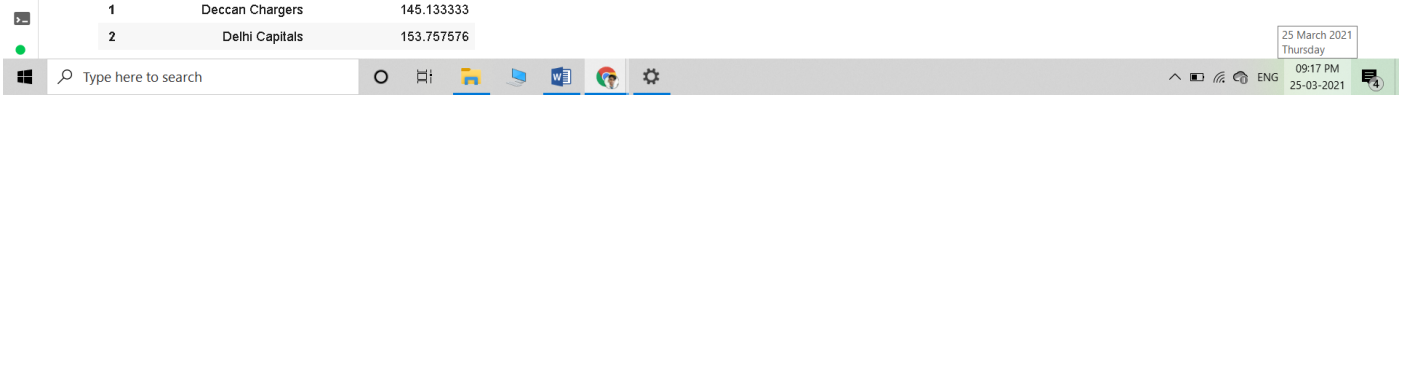
```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing
  import pandas.util.testing as tm
                df        sum_sq      mean_sq         F    PR(>F)
batting_team    14.0  2.958372e+04  2113.122993  2.241977  0.005194
Residual      1615.0  1.522180e+06   942.526581       NaN       NaN
```

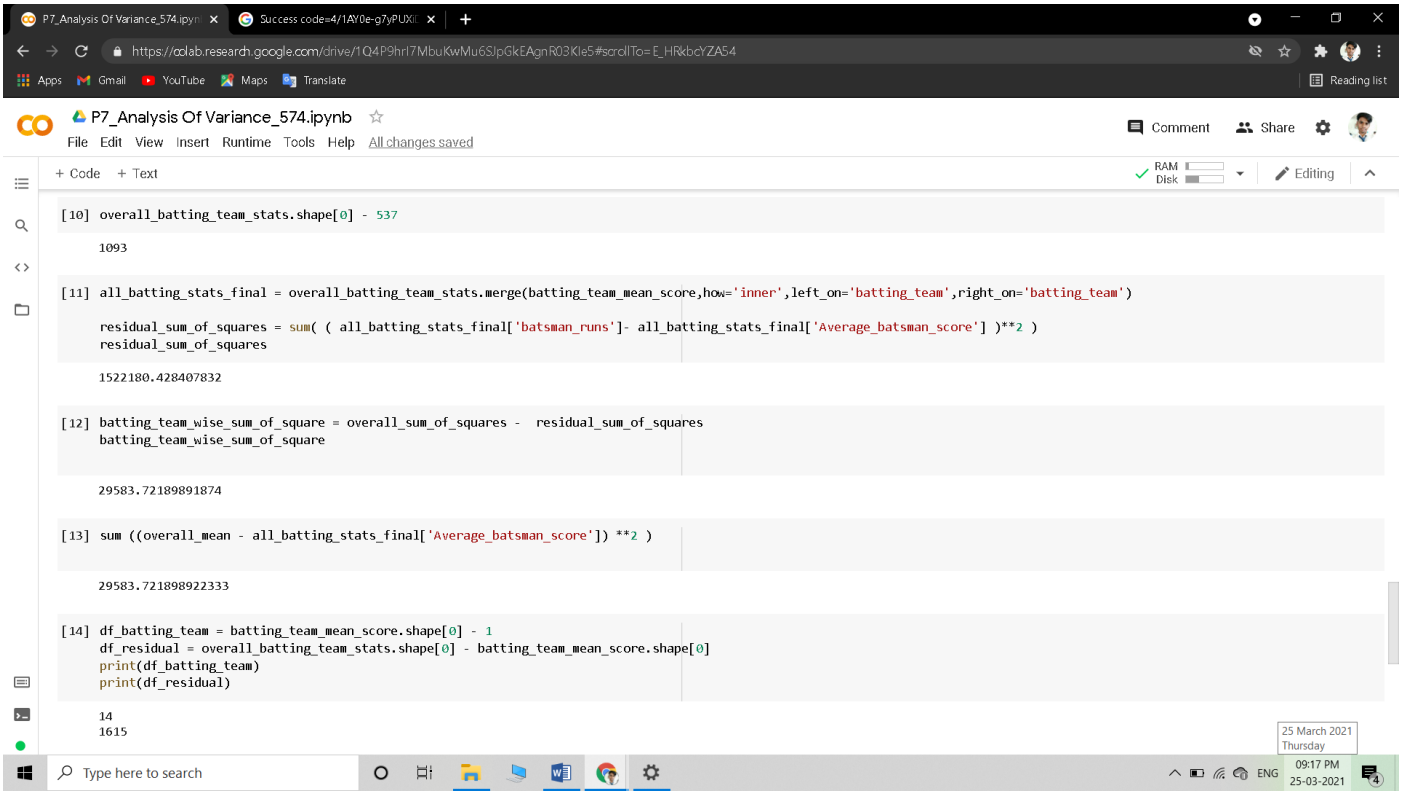
[8]
```
overall_mean = overall_batting_team_stats['batsman_runs'].mean()
overall_sum_of_squares = sum( (overall_batting_team_stats['batsman_runs']-overall_mean)**2)
overall_sum_of_squares
```

```
1551764.1503067508
```

[9]
```
batting_team_mean_score = overall_batting_team_stats.groupby('batting_team').mean().reset_index()
batting_team_mean_score.columns = ['batting_team', 'Average_batsman_score']
batting_team_mean_score
```

	batting_team	Average_batsman_score
0	Chennai Super Kings	151.421348
1	Deccan Chargers	145.133333
2	Delhi Capitals	153.757576





Colab Link :

https://colab.research.google.com/drive/1Q4P9hrI7MbuKwMu6SJpGkEAgnR03KIe5?usp=sharing

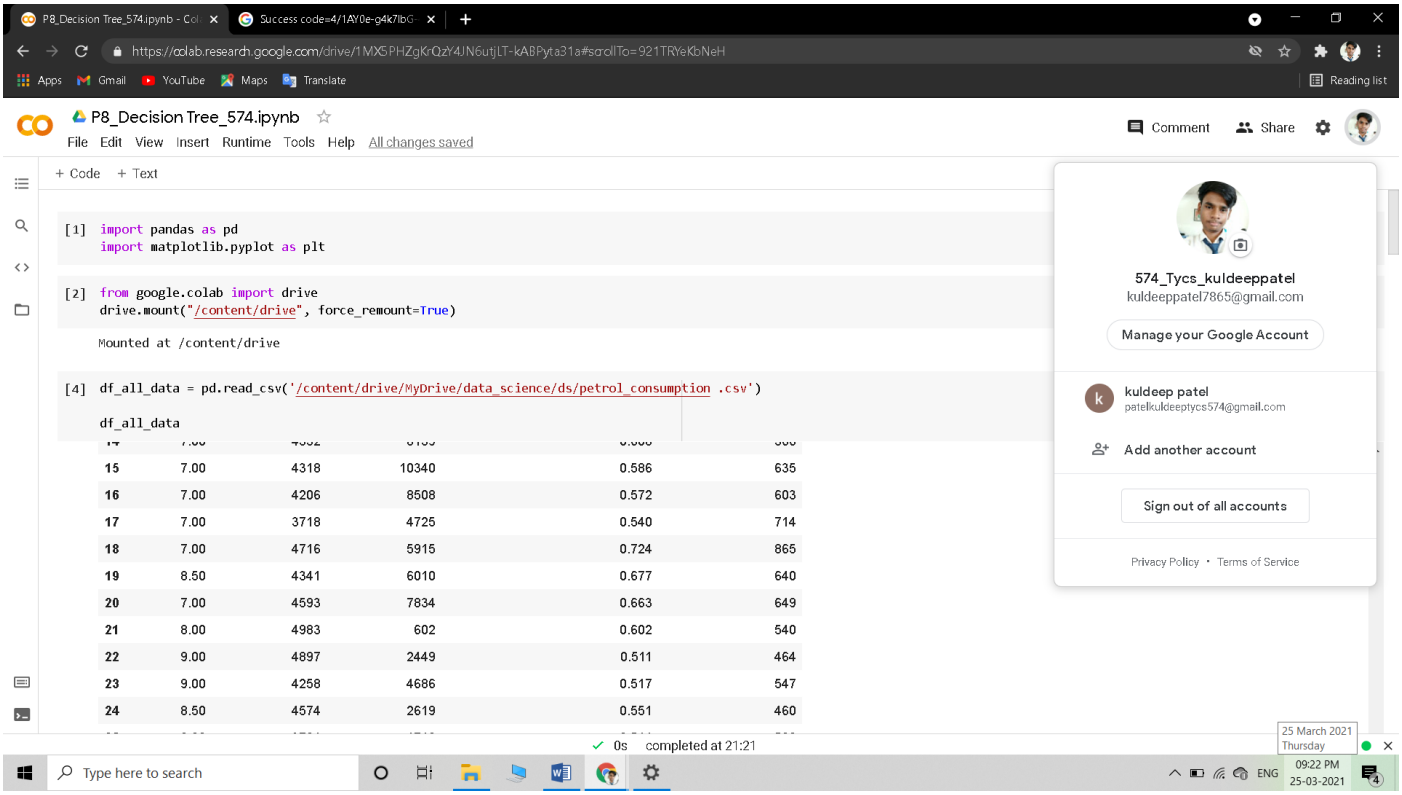
Practical – 8

Aim : Practical of Decision Tree

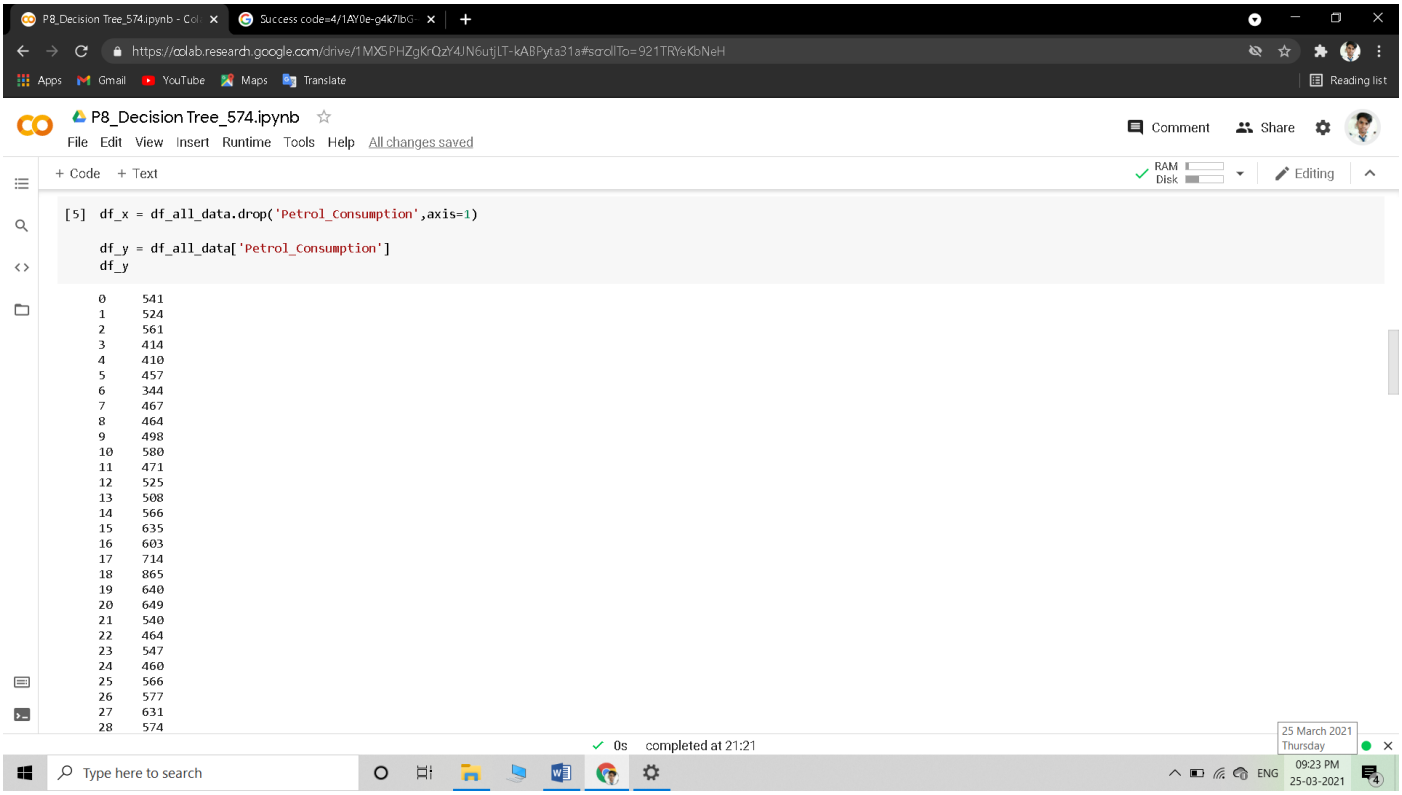
Theory :

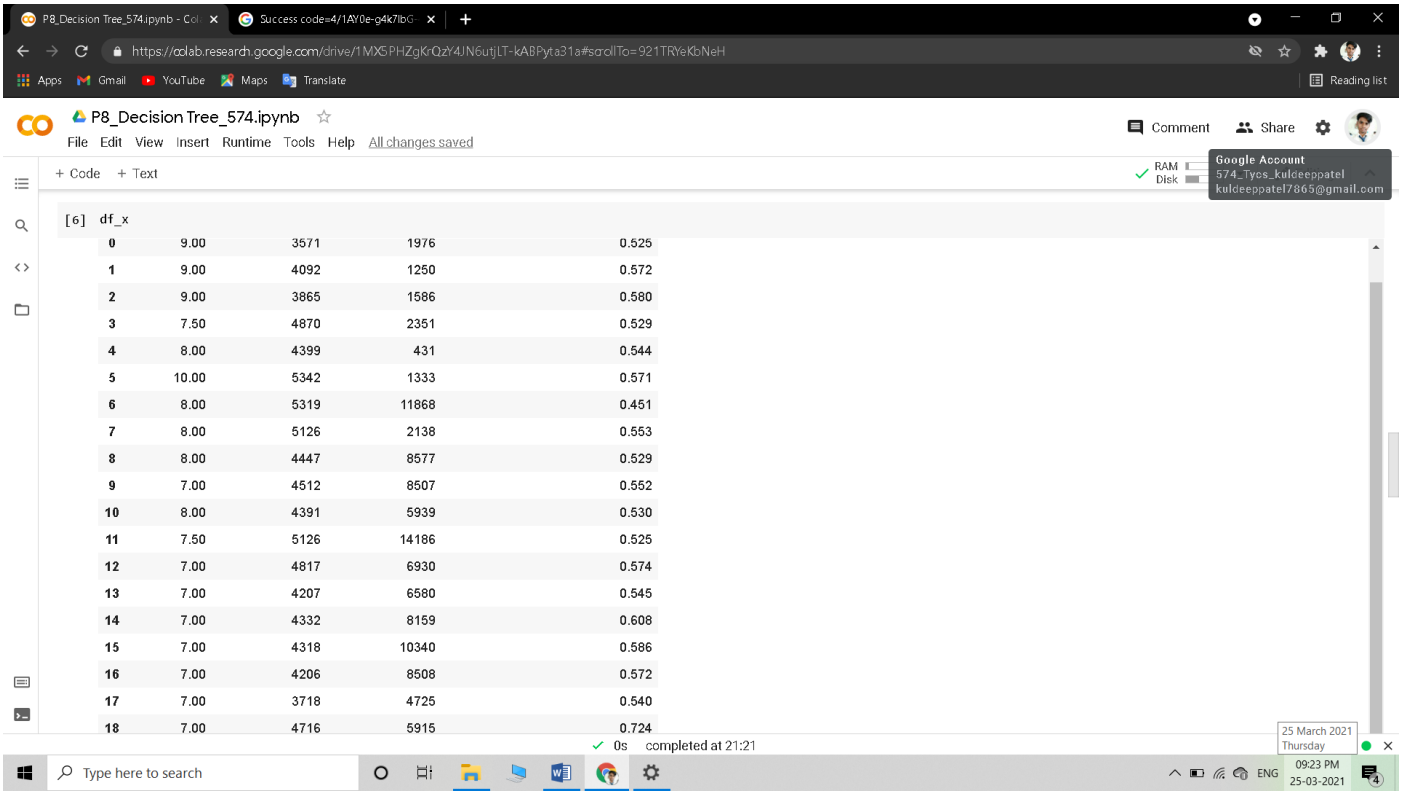
- A decision tree is the most powerful and popular tool for classification and prediction.
- A Decision tree is a flowchart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.
- A decision tree is a structure that includes a root node, branches, and leaf nodes. Each internal node denotes a test on an attribute, each branch denotes the outcome of a test, and each leaf node holds a class label.
- The topmost node in the tree is the root node.
- The benefits of having a decision tree are as follows −
  1. It does not require any domain knowledge.
  2. It is easy to comprehend.
  3. The learning and classification steps of a decision tree are simple and fast.
  4. Decision tree algorithm falls under the category of supervised learning. They can be used to solve both regression and classification problems.
  5. Decision tree uses the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree

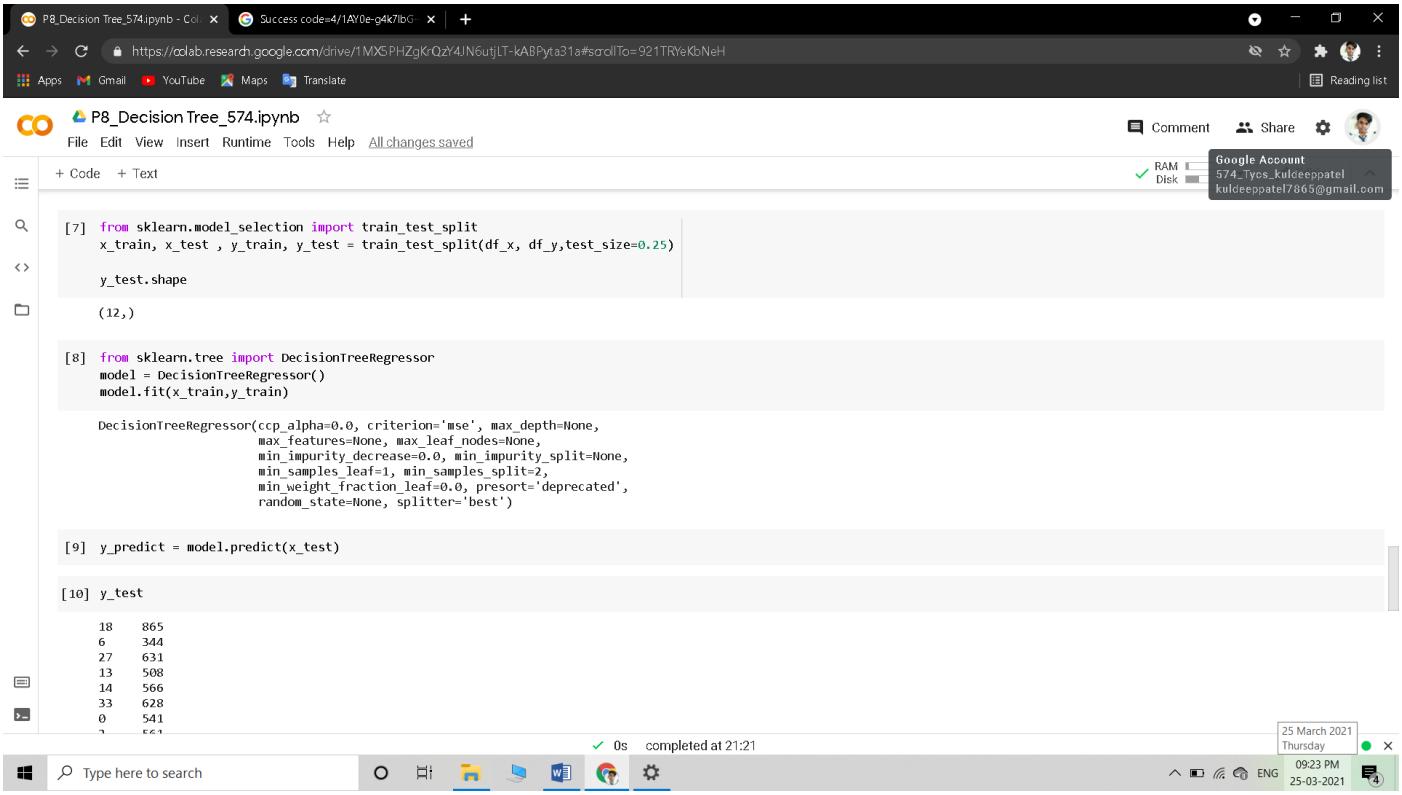
Code :

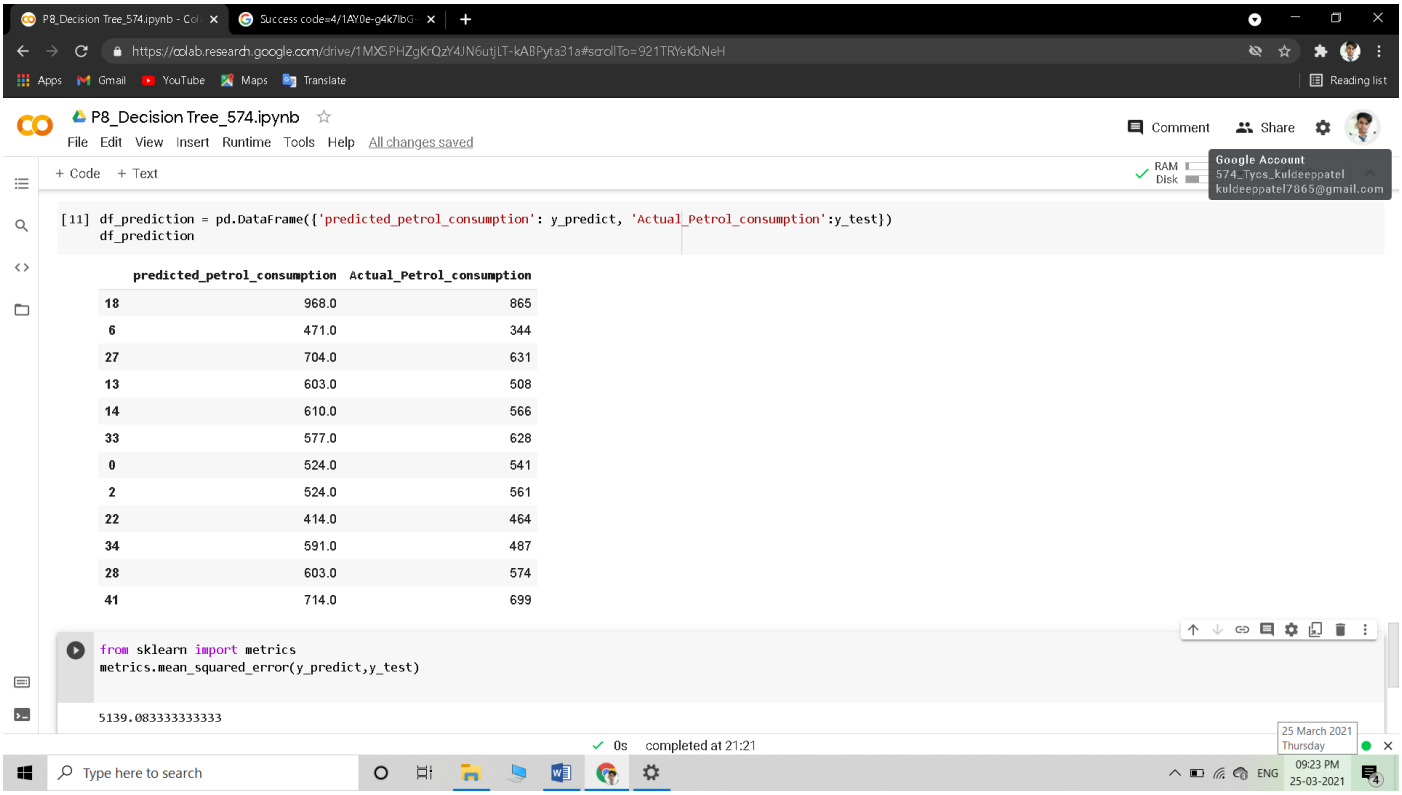


15	7.00	4318	10340	0.586	635
16	7.00	4206	8508	0.572	603
17	7.00	3718	4725	0.540	714
18	7.00	4716	5915	0.724	865
19	8.50	4341	6010	0.677	640
20	7.00	4593	7834	0.663	649
21	8.00	4983	602	0.602	540
22	9.00	4897	2449	0.511	464
23	9.00	4258	4686	0.517	547
24	8.50	4574	2619	0.551	460




P8_Decision Tree_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com
[6] df_x
0 9.00 3571 1976 0.525
1 9.00 4092 1250 0.572
2 9.00 3865 1586 0.580
3 7.50 4870 2351 0.529
4 8.00 4399 431 0.544
5 10.00 5342 1333 0.571
6 8.00 5319 11868 0.451
7 8.00 5126 2138 0.553
8 8.00 4447 8577 0.529
9 7.00 4512 8507 0.552
10 8.00 4391 5939 0.530
11 7.50 5126 14186 0.525
12 7.00 4817 6930 0.574
13 7.00 4207 6580 0.545
14 7.00 4332 8159 0.608
15 7.00 4318 10340 0.586
16 7.00 4206 8508 0.572
17 7.00 3718 4725 0.540
18 7.00 4716 5915 0.724
0s completed at 21:21
Type here to search
09:23 PM
25-03-2021


P8_Decision Tree_574.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
Google Account
574_Tycs_kuldeeppatel
kuldeeppatel7865@gmail.com
[7] from sklearn.model_selection import train_test_split
x_train, x_test , y_train, y_test = train_test_split(df_x, df_y,test_size=0.25)
y_test.shape
(12,)
[8] from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor()
model.fit(x_train,y_train)
DecisionTreeRegressor(ccp_alpha=0.0, criterion='mse', max_depth=None,
max_features=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0, presort='deprecated',
random_state=None, splitter='best')
[9] y_predict = model.predict(x_test)
[10] y_test
18 865
6 344
27 631
13 508
14 566
33 628
0 541
0s completed at 21:21
Type here to search
09:23 PM
25-03-2021



Colab Link :

https://colab.research.google.com/drive/1MX5PHZgKrQzY4JN6utjLT-kABPyta31a?usp=sharing