WEBSITE TRAFFIC ANALYSIS

INTRODUCTION:

The goal of this project is to leverage machine learning algorithms to gain actionable insights from the data. By harnessing predictive analytics, businesses can anticipate user behavior, detect trends, and optimize website performance. This process aids in making informed decisions to enhance user experiences, boost conversions, and achieve organizational objectives, such as increasing revenue and audience engagement

SOURCE CODE:

```
import numpy as np
import pandas as pd
import pandas_profiling
import warnings
warnings.filterwarnings('ignore')
import datetime
from datetime import date

import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set_style("whitegrid")

# import chart_studio.plotly as py
import cufflinks as cf
import plotly.express as px

from plotly.offline import download_plotlyjs, init_notebook_mode, plot,
iplot
init_notebook_mode(connected=True)


cf.go_offline()
```

```
import pandas_profiling
import plotly.graph_objects as go

from sklearn.model_selection import train_test_split, cross_val_score,
GridSearchCV
from sklearn.metrics import accuracy_score
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
import xgboost as xg
# from prophet import Prophet
```

Importing the required dataset, renaming the columns, removing the commas from the columns and converting their data types

In [2]:

linkcode

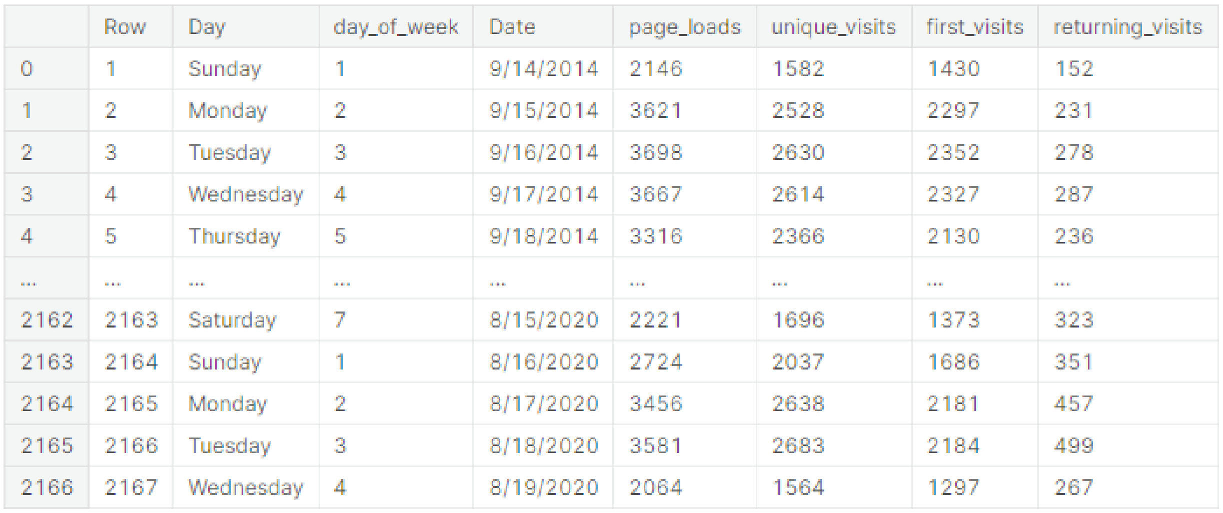
```
df=pd.read_csv('../input/daily-website-visitors/daily-website-visitors.csv
')

df.rename(columns = {'Day.Of.Week':'day_of_week'
                    ,'Page.Loads':'page_loads'
                    ,'Unique.Visits':'unique_visits'
                    ,'First.Time.Visits':'first_visits'
                    ,'Returning.Visits':'returning_visits'}, inplace =
True)

df=df.replace(',','',regex=True)

df['page_loads']=df['page_loads'].astype(int)
df['unique_visits']=df['unique_visits'].astype(int)
df['first_visits']=df['first_visits'].astype(int)
df['returning_visits']=df['returning_visits'].astype(int)

Df
```



	Row	Day	day_of_week	Date	page_loads	unique_visits	first_visits	returning_visits
0	1	Sunday	1	9/14/2014	2146	1582	1430	152
1	2	Monday	2	9/15/2014	3621	2528	2297	231
2	3	Tuesday	3	9/16/2014	3698	2630	2352	278
3	4	Wednesday	4	9/17/2014	3667	2614	2327	287
4	5	Thursday	5	9/18/2014	3316	2366	2130	236
...	...	...	...	...	...	...	...	...
2162	2163	Saturday	7	8/15/2020	2221	1696	1373	323
2163	2164	Sunday	1	8/16/2020	2724	2037	1686	351
2164	2165	Monday	2	8/17/2020	3456	2638	2181	457
2165	2166	Tuesday	3	8/18/2020	3581	2683	2184	499
2166	2167	Wednesday	4	8/19/2020	2064	1564	1297	267

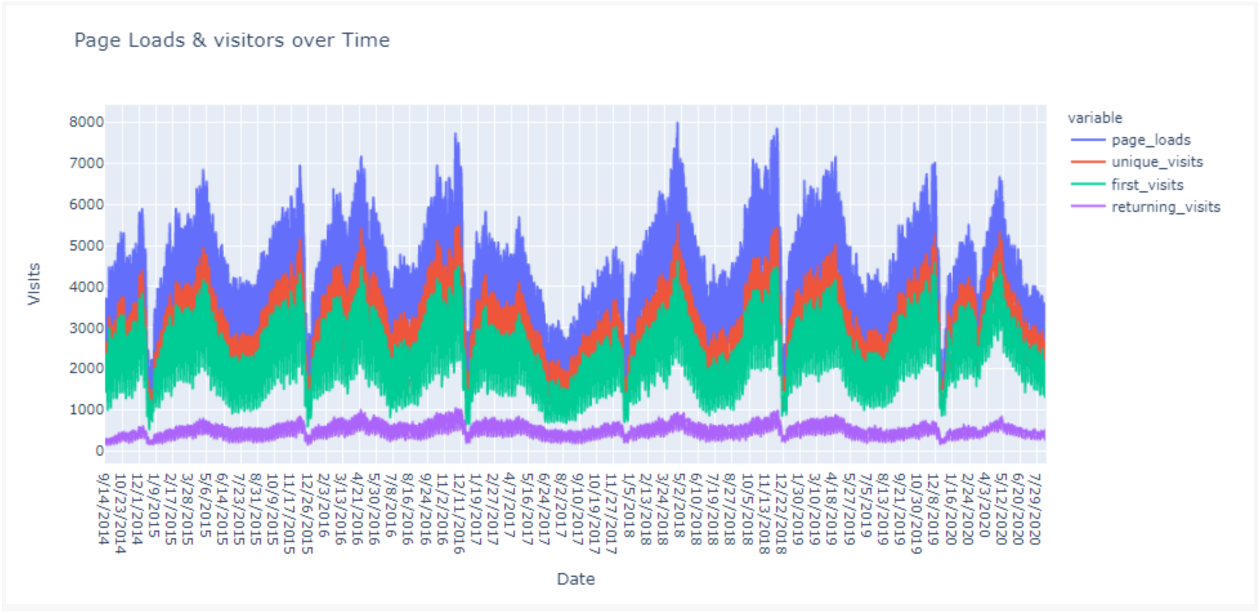
```
df.isna().sum()
```

```
Row                 0
Day                 0
day_of_week         0
Date                0
page_loads          0
unique_visits       0
first_visits        0
returning_visits    0
dtype: int64
```

```
df.duplicated().sum()
```

```
0
```

```
px.line(df,x='Date',y=['page_loads' ,'unique_visits' ,'first_visits'
,'returning_visits'],
       labels={'value':'Visits'}
       ,title='Page Loads & visitors over Time')
```



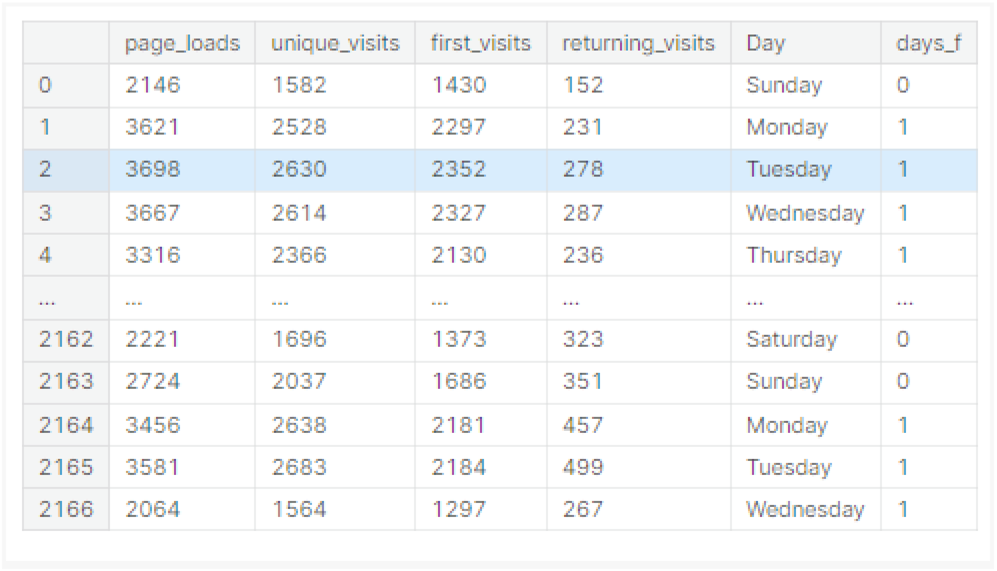
```
pred_df['days_f']=np.where((df['Day']=='Tuesday') |

                      (df['Day']=='Wednesday') |

                      (df['Day']=='Thursday') |

                      (df['Day']=='Monday'),1,0)


pred_df
```



	page_loads	unique_visits	first_visits	returning_visits	Day	days_f
0	2146	1582	1430	152	Sunday	0
1	3621	2528	2297	231	Monday	1
2	3698	2630	2352	278	Tuesday	1
3	3667	2614	2327	287	Wednesday	1
4	3316	2366	2130	236	Thursday	1
...	...	...	...	...	...	...
2162	2221	1696	1373	323	Saturday	0
2163	2724	2037	1686	351	Sunday	0
2164	3456	2638	2181	457	Monday	1
2165	3581	2683	2184	499	Tuesday	1
2166	2064	1564	1297	267	Wednesday	1

```
regressor2.score(X_test,y_test)*100
```

Out[27]:

```
100.0
```

Support Vector Regression

In [28]:

```
svr_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001)
svr_rbf.fit(X_train, y_train)
```

Out[28]:

```
SVR(C=1000.0, gamma=1e-05)
```

In [29]:

```
y_pred3 = svr_rbf.predict(X_test)
```

In [30]:

```
svr = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred3})
svr
```

Out[30]:

	Actual	Predicted
1486	4173	4173.783532
1602	1902	1904.847560
1460	2870	2870.181094
1134	2142	2142.904123
1513	4329	4328.316673
...	...	...

439	2579	2578.897313
271	2494	2493.887467
244	1818	1816.932763
1159	3332	3331.902324
1701	2565	2564.972314