**PHASE 4:Creating interactive dashboards using IBM cognos**

****

**INTRODUCTION:**

Web traffic analytics refers to collecting data about who comes to your website and what they do when they get there. That data is crucial to building effective sales and marketing strategies. While most people assume more traffic is always better, that's not always true.

**STEPS FOR CREATING DASHBOARD:**

1. Prerequisites. None.
2. Task 1: Upload data.
3. Task 2: Assemble a dashboard and work with templates.
4. Task 3: Create a smart visualization.
5. Task 4: Create a user-defined visualization.
6. Task 5: Customize the layout.

**PROGRAM:**

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

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

**out 1:**

**Row Day day\_of\_week Date page\_loads unique\_visit first\_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**

**In 2:**

df.isna().sum()

**out 2:**

Row 0

Day 0

day\_of\_week 0

Date 0

page\_loads 0

unique\_visits 0

first\_visits 0

returning\_visits 0

dtype: int64

**In 3:**

df.duplicated().sum()

**out 3:**

0

**In 4:**

df.info()

**out 4:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 2167 entries, 0 to 2166

Data columns (total 8 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Row 2167 non-null int64

1 Day 2167 non-null object

2 day\_of\_week 2167 non-null int64

3 Date 2167 non-null object

4 page\_loads 2167 non-null int64

5 unique\_visits 2167 non-null int64

6 first\_visits 2167 non-null int64

7 returning\_visits 2167 non-null int64

dtypes: int64(6), object(2)

memory usage: 135.6+ KB

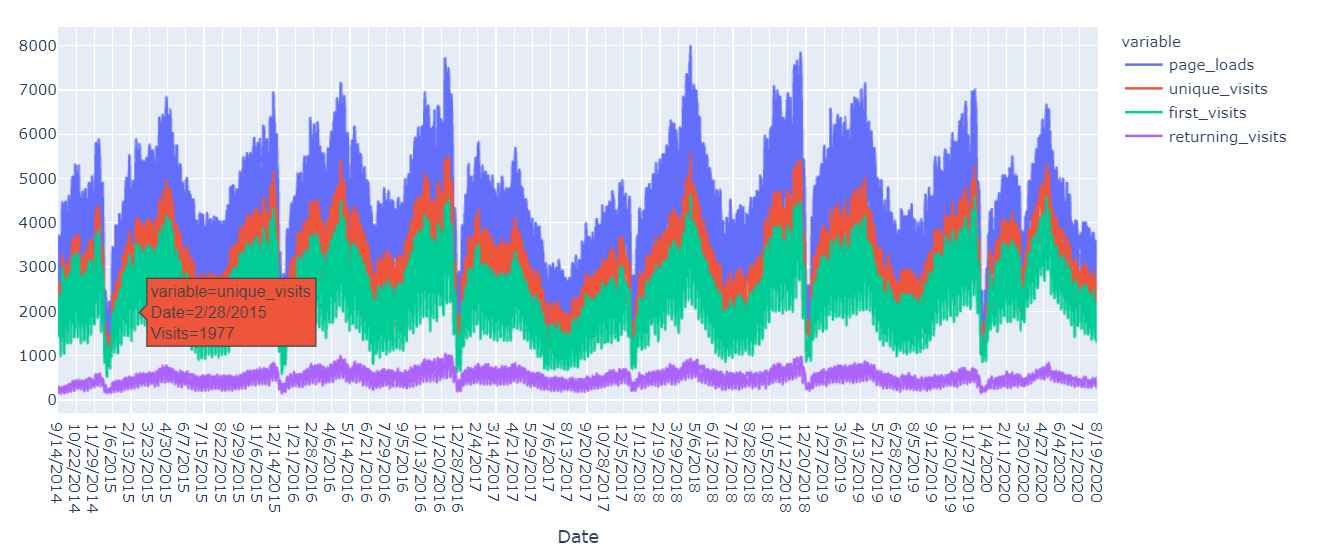
**In 5:**

px.line(df,x='Date',y=['page\_loads' ,'unique\_visits' ,'first\_visits' ,'returning\_visits'],

labels={'value':'Visits'}

,title='Page Loads & visitors over Time')

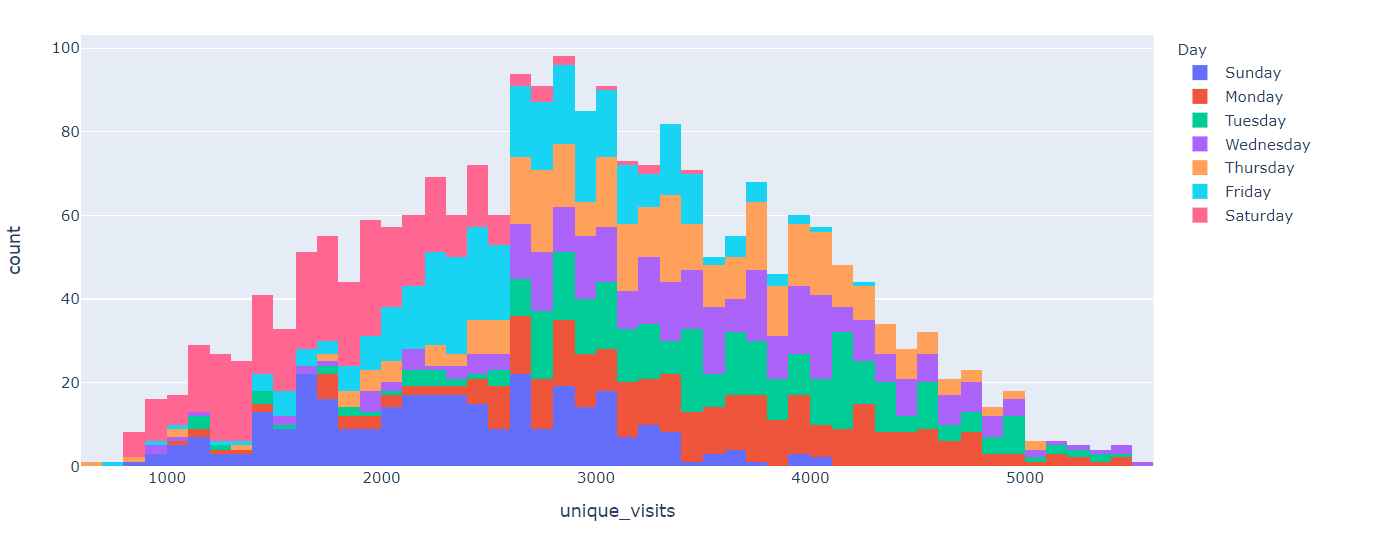
**Out 5:**

****

**In 6:**

px.histogram(df,x='unique\_visits',color='Day',title='unique visits for each day')

**out 6:**

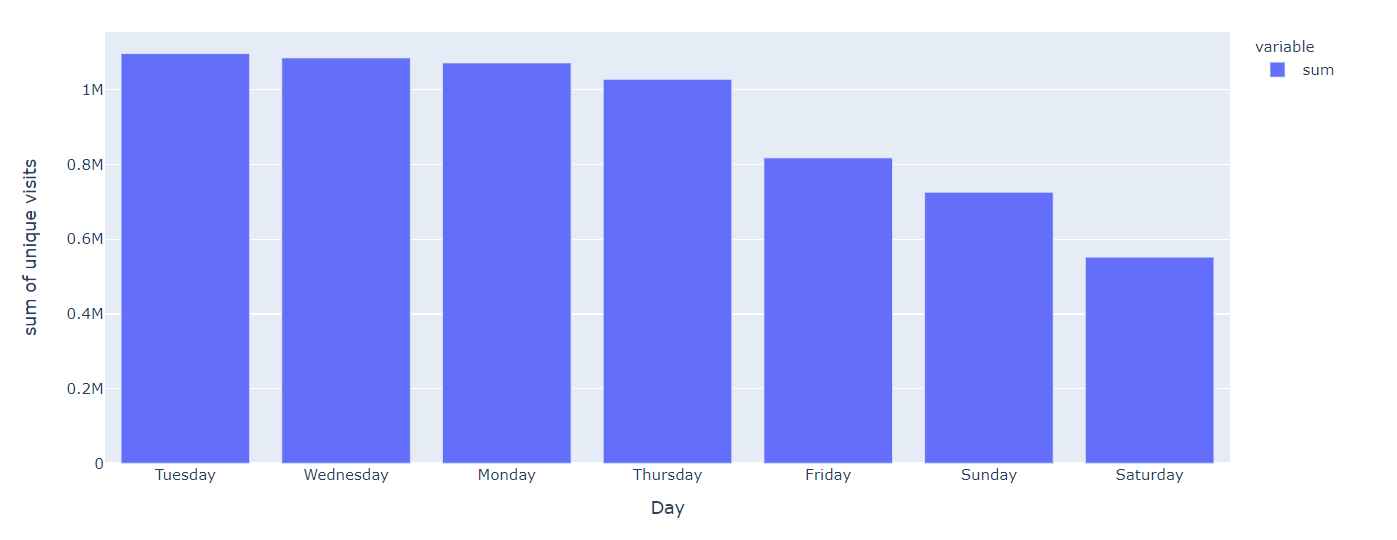
****

**In 7:**

day\_imp=df.groupby(['Day'])['unique\_visits'].agg(['sum']).sort\_values(by='sum',ascending=False)

px.bar(day\_imp,labels={'value':'sum of unique visits'},title='Sum of Unique visits for each day')

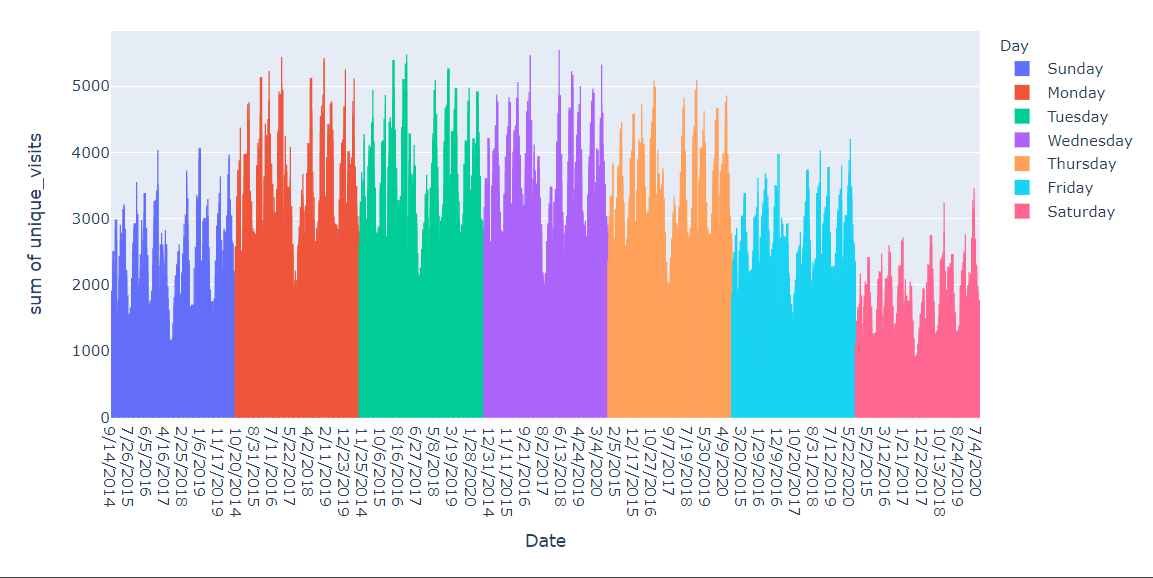
**out 7:**



**In 8:**

px.histogram(df,x='Date',y='unique\_visits',color='Day',title='Sum of unique visits for each day over Time')

**out 8:**

****

**In 9:**

sums=df.groupby(['Day'])[['page\_loads' ,'unique\_visits' ,'first\_visits' ,'returning\_visits']].sum().sort\_values(

by='unique\_visits',ascending=False)

sums

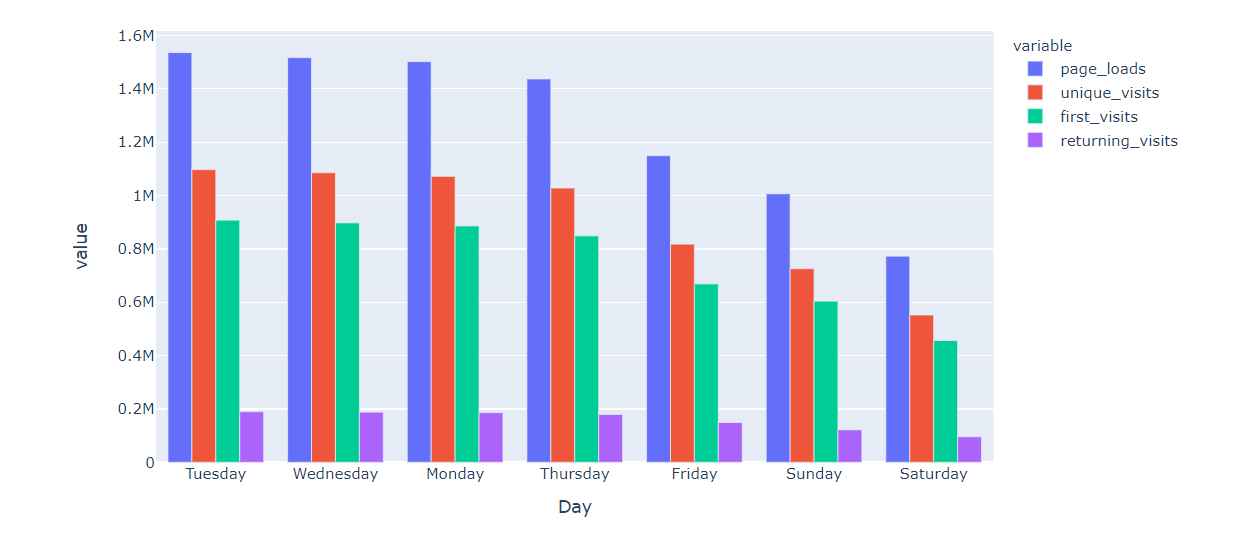
**out 9:**

| page\_loads | unique\_visits | first\_visits | returning\_visits |
| --- | --- | --- | --- |
| Day |  |  |  |
| Tuesday | 1536154 | 1097181 | 907752 |
| Wednesday | 1517114 | 1085624 | 897602 |

**In 10:**

px.bar(sums,barmode='group',title='Sum of page loads and visits for each of their days')

**out 10:**

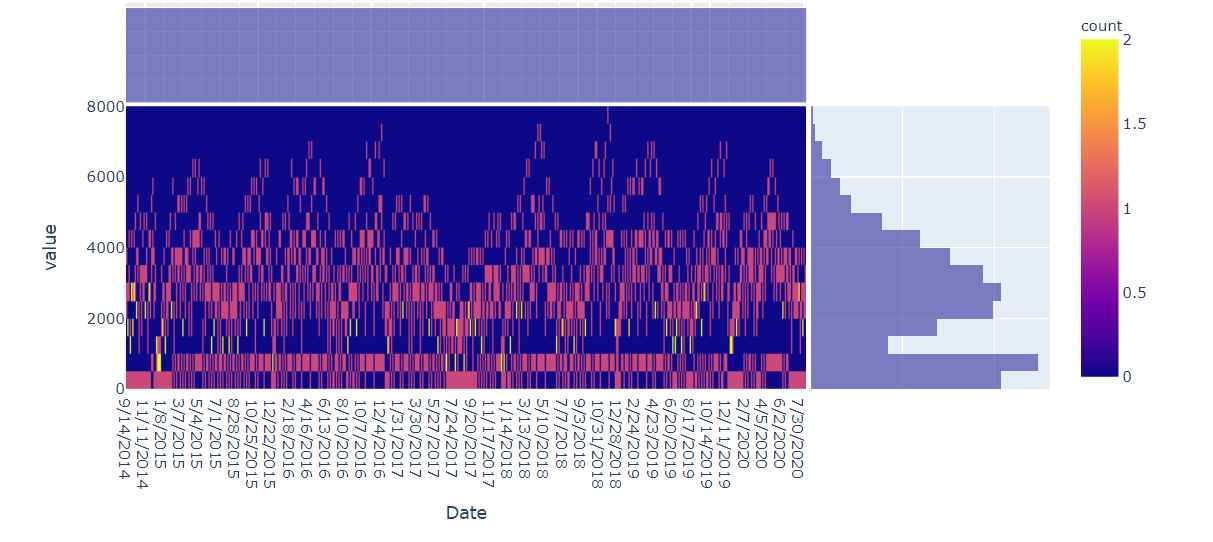
****

**In 11:**

px.density\_heatmap(df, x='Date',y=['page\_loads' ,'unique\_visits' ,'first\_visits' ,'returning\_visits']

*color\_continuous\_scale="Viridis"*

,marginal\_x="histogram", marginal\_y="histogram",title='Correlation for each data point')

**Out 11: **

**In 12:**

fig, ax = plt.subplots()

fig.set\_size\_inches(8, 6)

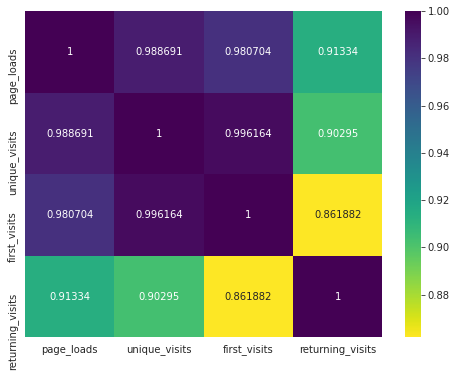
sns.heatmap(df[['page\_loads' ,'unique\_visits' ,'first\_visits' ,'returning\_visits']].corr(),

annot=True,

cmap='viridis\_r',

fmt='g')

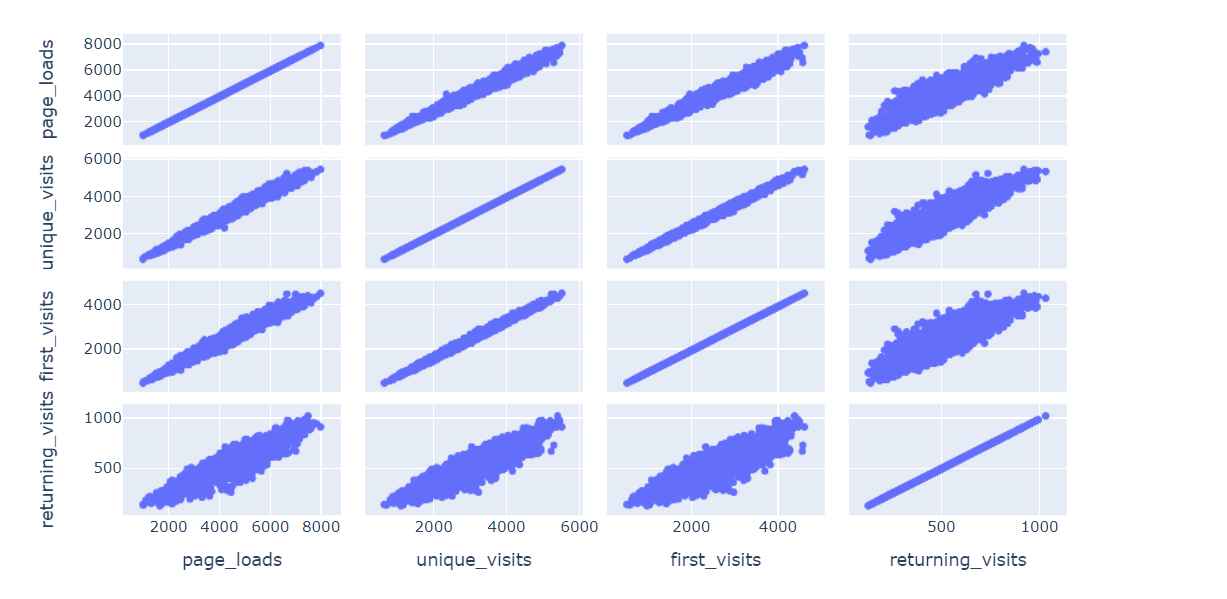
**out 12:**



**In 13:**

px.scatter\_matrix(df[['page\_loads' ,'unique\_visits' ,'first\_visits' ,'returning\_visits']])

**out 13:**

****

**In 14:**

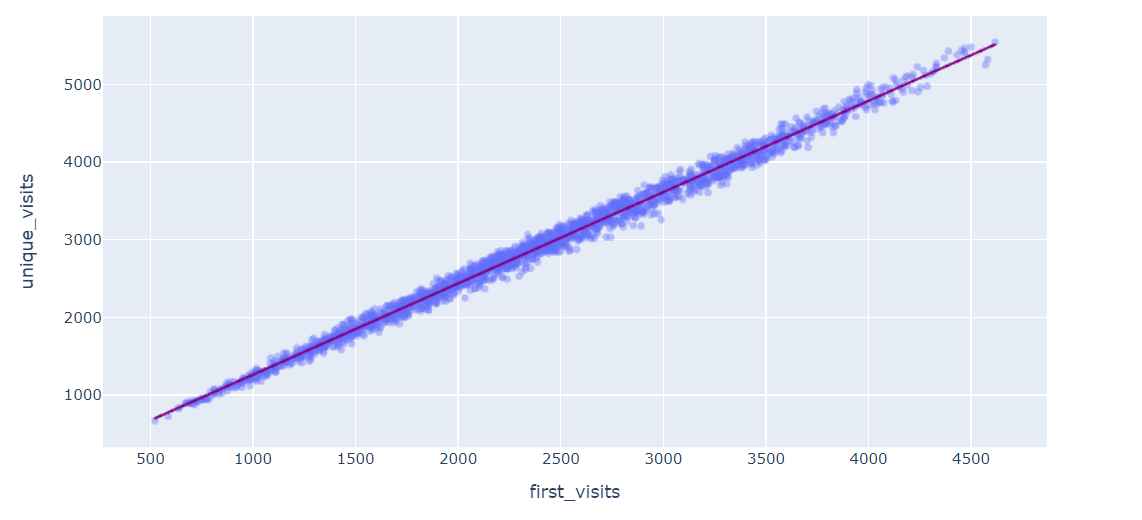
px.scatter(

df, x='first\_visits', y='unique\_visits',opacity=0.4,

trendline='ols', trendline\_color\_override='purple',title="Regression line for unique visits and first visits"

)

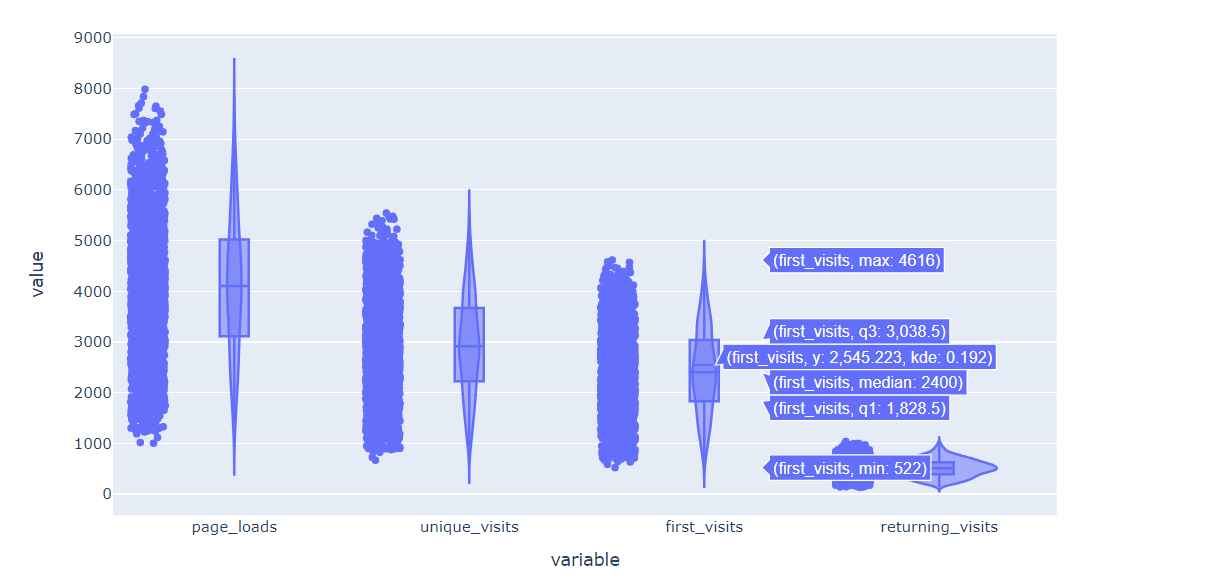
**Out 14:**

****

**In 15:**

px.violin(df,y=['page\_loads' ,'unique\_visits' ,'first\_visits' ,'returning\_visits'],box=True,points='all')

**out 15:**

****

**In 16:**

regressor2=LinearRegression(fit\_intercept=False,normalize=True)

regressor2.fit(X\_train, y\_train)

**out 16:**

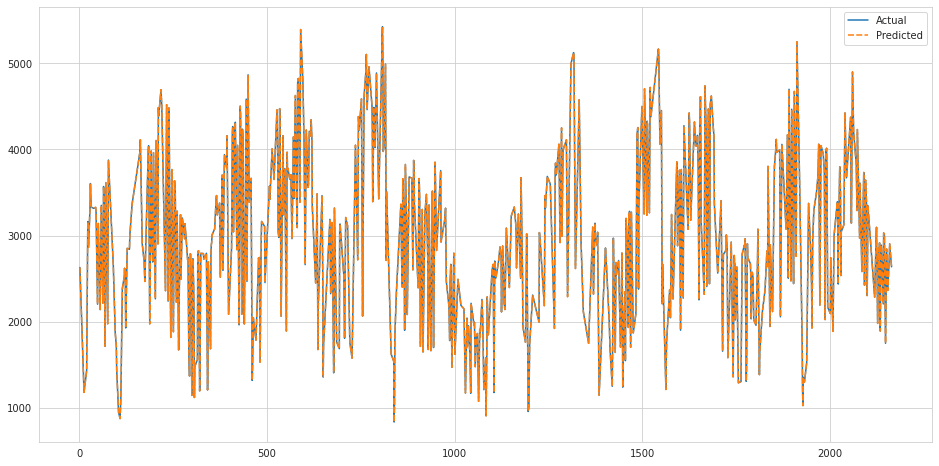
LinearRegression(fit\_intercept=False, normalize=True)

**In 17:**

plt.figure(figsize=(16,8))

sns.lineplot(data=lr2)

**out 17:**



**In 18:**

regressor2.score(X\_test,y\_test)\*100

**out 18:**

100.0

**In 19:**

svr\_rbf = SVR(kernel='rbf', C=1e3, gamma=0.00001)

svr\_rbf.fit(X\_train, y\_train)

**out 19:**

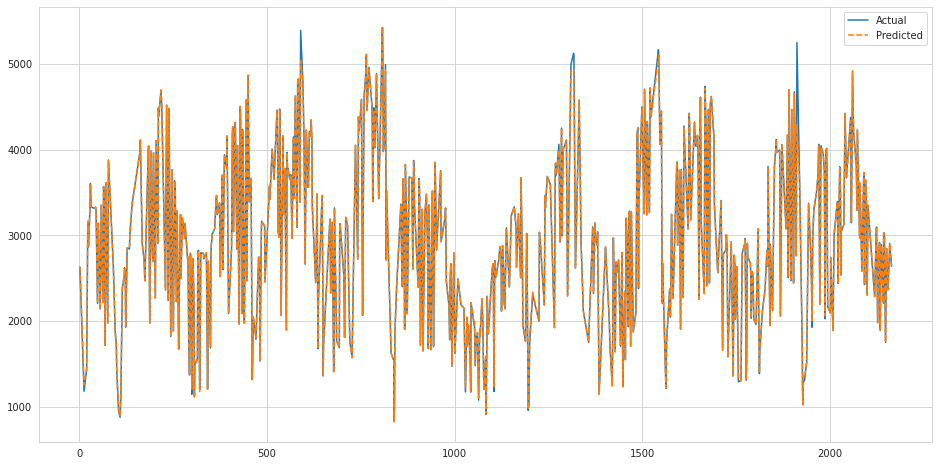
SVR(C=1000.0, gamma=1e-05)

**In 20:**

plt.figure(figsize=(16,8))

sns.lineplot(data=svr)

**out 20:**



**In 21:**

svr\_rbf.score(X\_test,y\_test)\*100

**out 21:**

99.80054455767926

**In 22:**

xgb\_r = xg.XGBRegressor(objective ='reg:squarederror',n\_estimators = 10, seed = 123)

xgb\_r.fit(X\_train, y\_train)

**out 22:**

XGBRegressor(base\_score=0.5, booster='gbtree', colsample\_bylevel=1,

colsample\_bynode=1, colsample\_bytree=1, gamma=0, gpu\_id=-1,

importance\_type='gain', interaction\_constraints='',

learning\_rate=0.300000012, max\_delta\_step=0, max\_depth=6,

min\_child\_weight=1, missing=nan, monotone\_constraints='()',

n\_estimators=10, n\_jobs=4, num\_parallel\_tree=1, random\_state=123,

reg\_alpha=0, reg\_lambda=1, scale\_pos\_weight=1, seed=123,

subsample=1, tree\_method='exact', validate\_parameters=1,

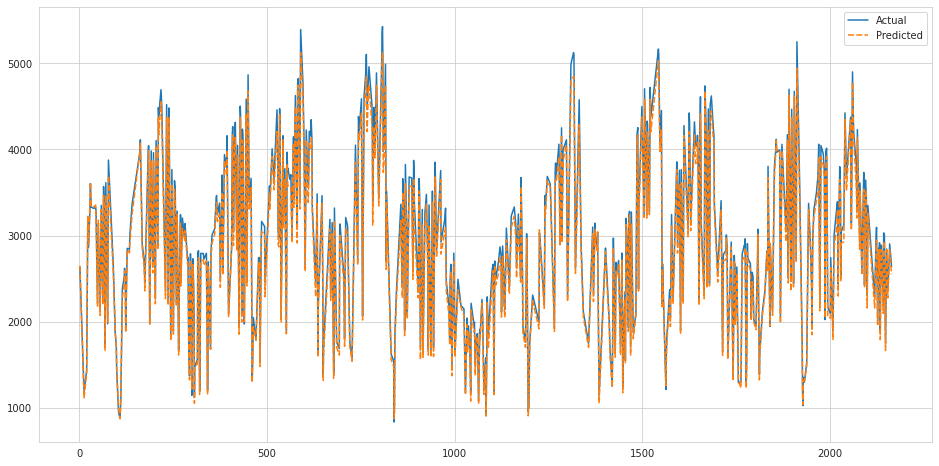
verbosity=None)

**In 23:**

plt.figure(figsize=(16,8))

sns.lineplot(data=xgb\_df)

**out 23:**



**In 24:**

xgb\_r.score(X\_test,y\_test)\*100

**out 24:**

98.7655882096893

**CONCLUSION:**

In conclusion, website traffic is a crucial aspect of any online business. By investing in SEO, social media, content marketing, and paid advertising, businesses can attract more visitors to their website and increase brand awareness, lead generation, and search engine rankings.