Instagram Reach Forecasting using Python

Problem Statement

Instagram reach forecasting is the process of predicting the number of people that an Instagram post, story, or other content will be reached, based on historical data and various other factors.

For content creators and anyone using Instagram professionally, predicting the reach can be valuable for planning and optimizing their social media strategy. By understanding how their content is performing, creators can make informed decisions about when to publish, what types of content to create, and how to engage their audience. It can lead to increased engagement, better performance metrics, and ultimately, greater success on the platform.

Import Libraries

In [1]:
```
import pandas as pd
import plotly.graph_objs as go
import plotly.express as px
import plotly.io as pio
pio.templates.default = "plotly_white"

df = pd.read_csv("/content/Instagram forecast analysis.csv", encoding = 'latir
print(df.head())
```

```
                  Date  Instagram reach
0  2022-04-01T00:00:00             7620
1  2022-04-02T00:00:00            12859
2  2022-04-03T00:00:00            16008
3  2022-04-04T00:00:00            24349
4  2022-04-05T00:00:00            20532
```

Convert the date column to datetime data type

In [2]:
```
df["Date"] = pd.to_datetime(df["Date"])
print(df.head())
```

```
        Date  Instagram reach
0 2022-04-01             7620
1 2022-04-02            12859
2 2022-04-03            16008
3 2022-04-04            24349
4 2022-04-05            20532
```

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 365 entries, 0 to 364
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Date             365 non-null    datetime64[ns]
 1   Instagram reach  365 non-null    int64
dtypes: datetime64[ns](1), int64(1)
memory usage: 5.8 KB
```

Analyse the trend of instagram using line chart

In [4]:
```
fig = go.Figure()
fig.add_trace(go.Scatter(x = df["Date"], y = df["Instagram reach"], mode = 'li
fig.update_layout(title = "Instgram reach over time", xaxis_title = "Date", ya
fig.show()
```

Analyse instagram reach for each day using a bar chart

In [5]:

```
fig = go.Figure()
fig.add_trace(go.Bar(x= df["Date"], y = df["Instagram reach"], name = "Instagr
fig.update_layout(title = "Instagram Reach Trend", xaxis_title = "Date", yaxis
fig.show()
```

Analyse the distribution of reach using box plot

In [6]:

```
fig = go.Figure()
fig.add_trace(go.Box(y= df["Instagram reach"], name = "Instagram reach" ))
fig.update_layout(title = "Distribution of Instagram Reach", yaxis_title = "Ir
fig.show()
```

Create a day column to analyse the instagram reach based on the days of the week.

In [7]:
```
df["Day"] = df["Date"].dt.day_name()
print(df.head())
```

```
        Date  Instagram reach       Day
0 2022-04-01             7620    Friday
1 2022-04-02            12859  Saturday
2 2022-04-03            16008    Sunday
3 2022-04-04            24349    Monday
4 2022-04-05            20532   Tuesday
```

Analyse the reach based on days of the week. For this we can group the DataFrame of the day column and calculate the mean, median, standard deviation of the Instagram reach column for each day.

In [8]:
```
import numpy as np
day_stats = df.groupby("Day")["Instagram reach"].agg(["mean", "median", "std"]
print(day_stats)
```

```
         Day          mean   median            std
0     Friday  46666.849057  35574.0  29856.943036
1     Monday  52621.692308  46853.0  32296.071347
2   Saturday  47374.750000  40012.0  27667.043634
3     Sunday  53114.173077  47797.0  30906.162384
4   Thursday  48570.923077  39150.0  28623.220625
5    Tuesday  54030.557692  48786.0  32503.726482
6  Wednesday  51017.269231  42320.5  29047.869685
```

Create a bar chart for the reach for each day of the week

In [9]:

```
fig = go.Figure()
fig.add_trace(go.Bar(x = day_stats["Day"], y= day_stats["mean"], name ="Mean"))
fig.add_trace(go.Bar(x = day_stats["Day"], y = day_stats["median"], name = "Me
fig.add_trace(go.Bar(x = day_stats["Day"], y = day_stats["std"], name = "Stand
fig.update_layout(title = "Instagram reach by day of the week", xaxis_title =
fig.show()
```

Instagram Reach Forecasting Using Time Series Forecasting

In [10]:

```
from plotly.tools import mpl_to_plotly
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

data = df[["Date", "Instagram reach"]]
result = seasonal_decompose(df["Instagram reach"], model = 'multiplicative', p

fig = plt.figure()
fig = result.plot()

fig = mpl_to_plotly(fig)
fig.show()
```

```
<Figure size 640x480 with 0 Axes>
```

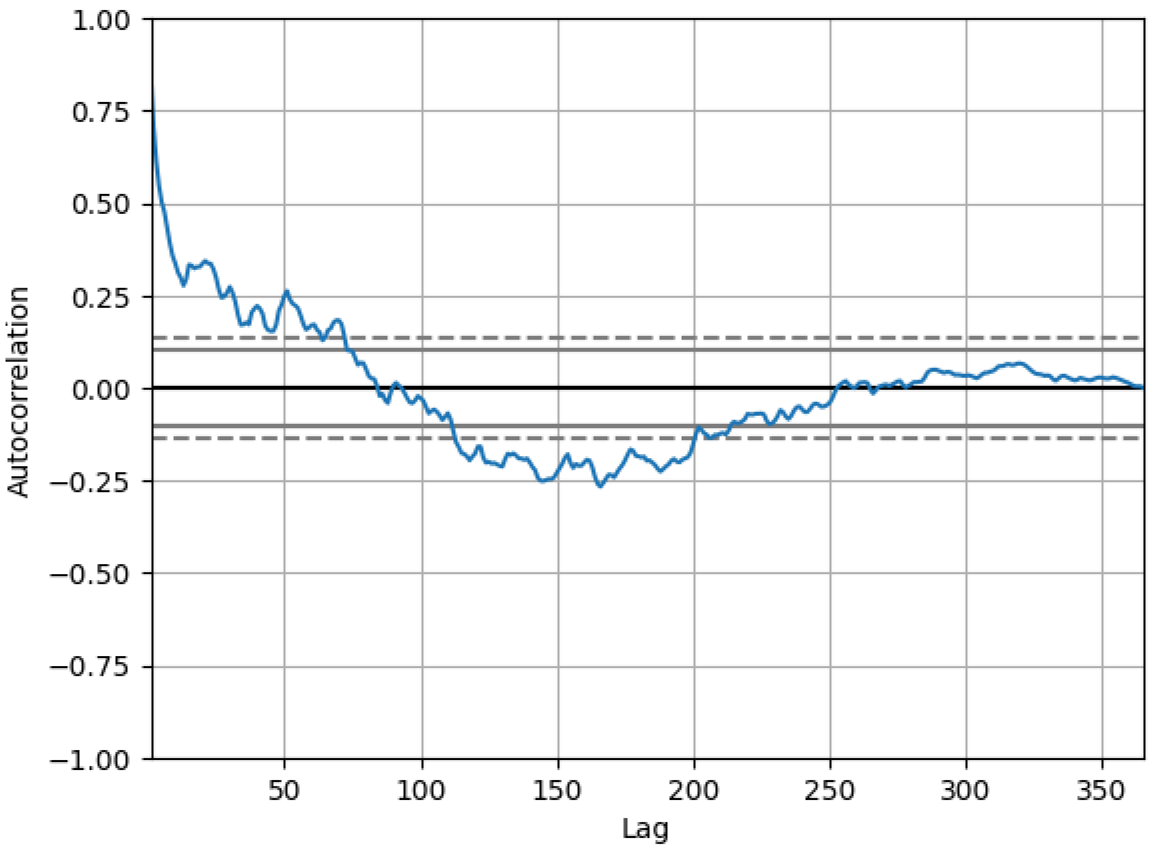
The reach is affected by seasonality, so we can use the SARIMA model to forecast the reach of the instagram account.We need to find the p, d and q values to forecast the reach of the instagram. To find the value of d, we can use autocorrelation plot, to find the value of q, we can use partial autocorrelation plot.

The value of d will be 1.

To visualise autocorrelation plot to find the value of p

In [11]:
```
pd.plotting.autocorrelation_plot(df["Instagram reach"])
```

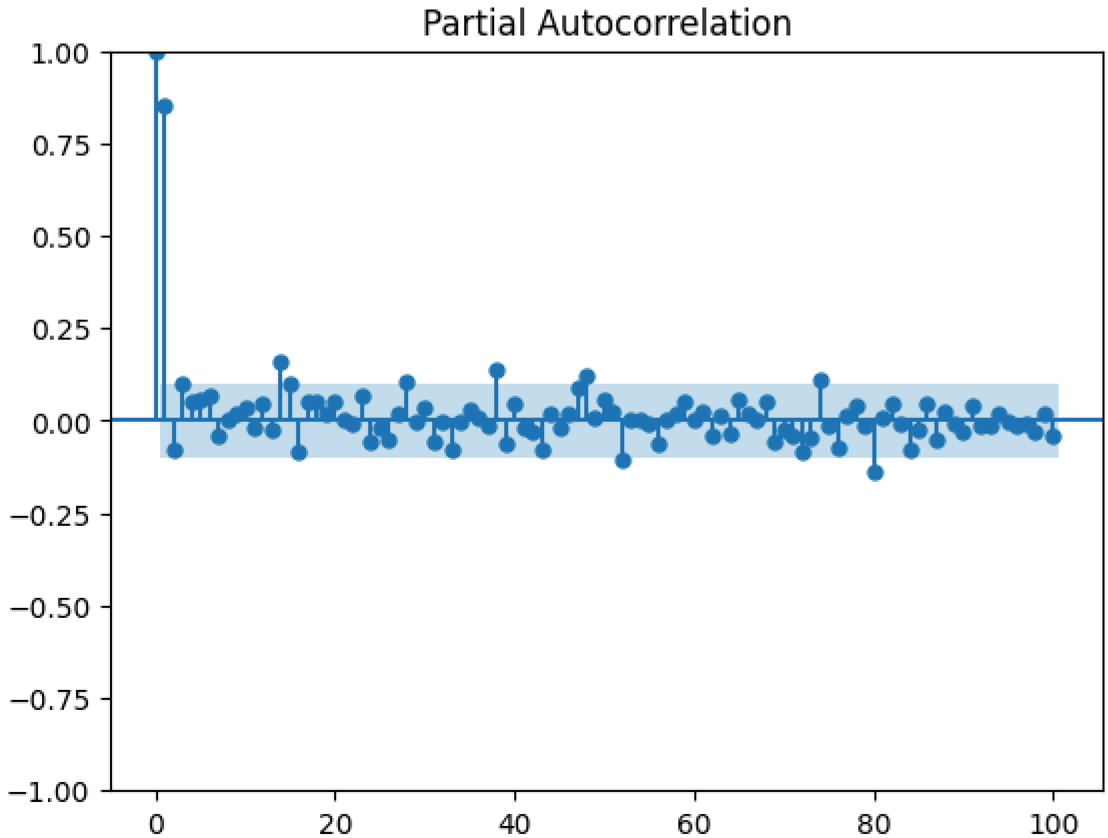
Out[11]: <Axes: xlabel='Lag', ylabel='Autocorrelation'>

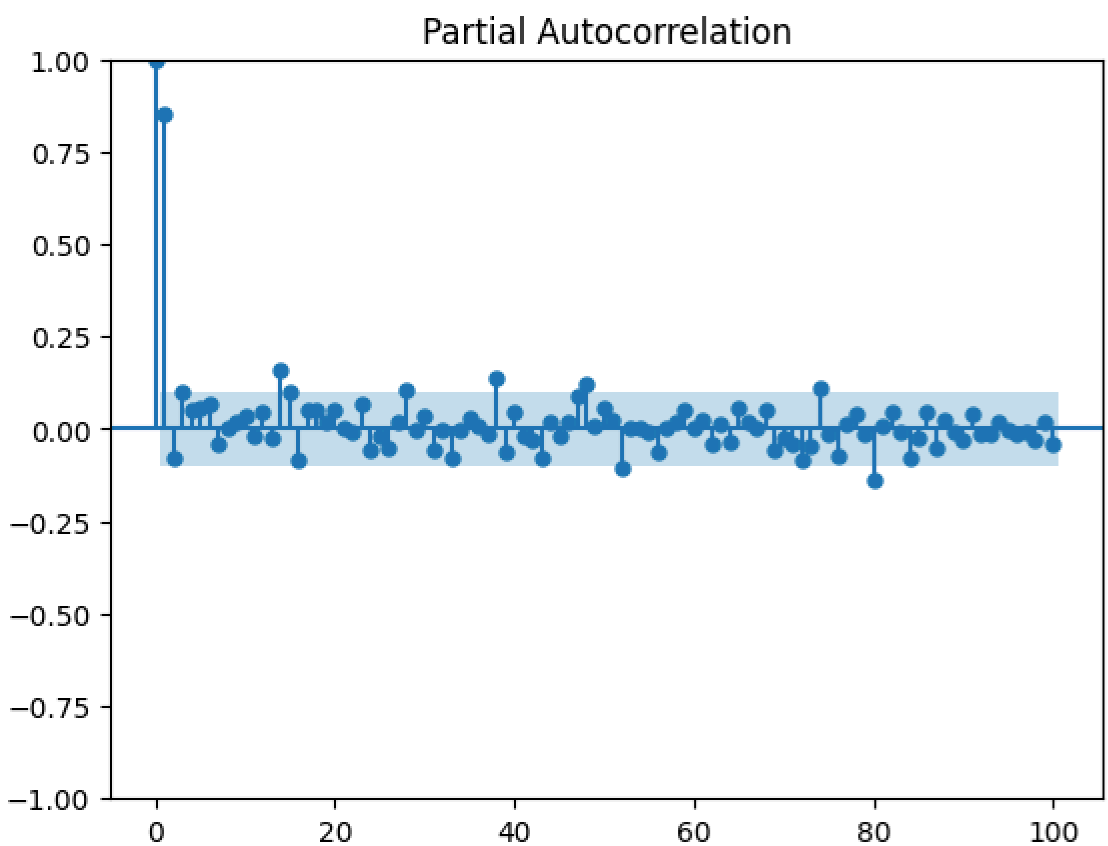


To find the value of q visualise apartial autocorrelation plot.

In [12]:
```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
plot_pacf(df["Instagram reach"], lags = 100)
```

Out[12]:





In [13]:

```
# Train a model using SARIMA
```

```python
p, d, q = 8, 1, 2

import statsmodels.api as sm
import warnings
model = sm.tsa.statespace.SARIMAX(df["Instagram reach"], order = (p, d, q), se
model = model.fit()
print(model.summary())
```

```
/usr/local/lib/python3.12/dist-packages/statsmodels/base/model.py:607: Converge
nceWarning:

Maximum Likelihood optimization failed to converge. Check mle_retvals
```

```
                                      SARIMAX Results
==============================================================================
===========
Dep. Variable:                    Instagram reach   No. Observations:
365
Model:             SARIMAX(8, 1, 2)x(8, 1, 2, 12)   Log Likelihood
-3938.513
Date:                            Sat, 01 Nov 2025   AIC
7919.027
Time:                                    10:45:57   BIC
8000.163
Sample:                                         0   HQIC
7951.315
                                            - 365
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.1898      6.613      0.029      0.977     -12.772      13.152
ar.L2          0.4778      6.164      0.078      0.938     -11.604      12.559
ar.L3         -0.1183      1.407     -0.084      0.933      -2.876       2.640
ar.L4          0.0424      0.266      0.159      0.873      -0.479       0.564
ar.L5         -0.0211      0.186     -0.113      0.910      -0.386       0.344
ar.L6          0.0313      0.268      0.117      0.907      -0.494       0.557
ar.L7          0.0100      0.422      0.024      0.981      -0.818       0.838
ar.L8         -0.0128      0.232     -0.055      0.956      -0.468       0.442
ma.L1         -0.2225      6.610     -0.034      0.973     -13.177      12.732
ma.L2         -0.7134      6.358     -0.112      0.911     -13.175      11.748
ar.S.L12      -1.0866      1.506     -0.721      0.471      -4.039       1.866
ar.S.L24      -1.7432      2.213     -0.788      0.431      -6.080       2.594
ar.S.L36      -1.4276      1.900     -0.752      0.452      -5.151       2.296
ar.S.L48      -1.0809      1.548     -0.698      0.485      -4.115       1.953
ar.S.L60      -0.7795      1.104     -0.706      0.480      -2.943       1.384
ar.S.L72      -0.4468      0.782     -0.571      0.568      -1.980       1.086
ar.S.L84      -0.2204      0.501     -0.440      0.660      -1.202       0.761
ar.S.L96      -0.0526      0.245     -0.215      0.830      -0.533       0.428
ma.S.L12       0.2250      1.507      0.149      0.881      -2.728       3.178
ma.S.L24       0.8222      1.272      0.647      0.518      -1.670       3.314
sigma2      4.863e+08   1.45e-07   3.35e+15      0.000    4.86e+08    4.86e+08
==============================================================================
====
Ljung-Box (L1) (Q):                   0.01   Jarque-Bera (JB):              21
6.02
Prob(Q):                              0.91   Prob(JB):
0.00
Heteroskedasticity (H):               0.71   Skew:
0.29
Prob(H) (two-sided):                  0.07   Kurtosis:
6.79
==============================================================================
====

Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-
```

```
step).
[2] Covariance matrix is singular or near-singular, with condition number 1.29
e+32. Standard errors may be unstable.
```

Make predictions using the model

In [14]:
```
predictions = model.predict(len(df), len(df) + 100)
trace_train = go.Scatter(x=data.index,
                         y=data["Instagram reach"],
                         mode="lines",
                         name="Training Data")
trace_pred = go.Scatter(x=predictions.index,
                        y=predictions,
                        mode="lines",
                        name="Predictions")

layout = go.Layout(title="Instagram Reach Time Series and Predictions",
                   xaxis_title="Date",
                   yaxis_title="Instagram Reach")

fig = go.Figure(data=[trace_train, trace_pred], layout=layout)
fig.show()
```

Summary

Instagram reach prediction is the process of predicting the number of people that an Instagram post, story, or other content will be reached, based on historical data and various other factors.