Machine learning

INTERMEDIATE QUESTIONS

Q-1. Imagine you have a dataset where you have different Instagram features like u sername , Caption , Hashtag , Followers , Time_Since_posted , and likes , now your task is to predict the number of likes and Time Since posted and the rest of the features are your input features. Now you have to build a model which can predict the number of likes and Time Since posted.

Dataset This is the Dataset You can use this dataset for this question.

In [3]:

```
!pip install wordcloud
```

```
Collecting wordcloud
  Downloading wordcloud-1.9.2-cp39-cp39-win_amd64.whl (153 kB)
Requirement already satisfied: matplotlib in c:\users\abhik\anaconda3\lib
\site-packages (from wordcloud) (3.4.3)
Requirement already satisfied: numpy>=1.6.1 in c:\users\abhik\anaconda3\l
ib\site-packages (from wordcloud) (1.20.3)
Requirement already satisfied: pillow in c:\users\abhik\anaconda3\lib\sit
e-packages (from wordcloud) (8.4.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\abhik\ana
conda3\lib\site-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\abhik\anacon
da3\lib\site-packages (from matplotlib->wordcloud) (1.3.1)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\abhik\anacond
a3\lib\site-packages (from matplotlib->wordcloud) (3.0.4)
Requirement already satisfied: cycler>=0.10 in c:\users\abhik\anaconda3\l
ib\site-packages (from matplotlib->wordcloud) (0.10.0)
Requirement already satisfied: six in c:\users\abhik\anaconda3\lib\site-p
ackages (from cycler>=0.10->matplotlib->wordcloud) (1.16.0)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.9.2
```

In [4]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import OneHotEncoder
import re
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator

```

In [11]:

```
## Load the dataset using pandas:
data = pd.read_csv("Downloads/archive/instagram_reach.csv")
```

In [12]:

```
## Checking top 5 rows
data.head()
```

Out[12]:

	Unnamed: 0	S.No	USERNAME	Caption	Followers	
0	0	1	mikequindazzi	Who are #DataScientist and what do they do? >>...	1600	#MachineLea
1	1	2	drgorillapaints	We all know where it's going. We just have to ...	880	#deck .#mac #macintosh#
2	2	3	aitrading_official	Alexander Barinov: 4 years as CFO in multinati...	255	#whoiswho #aitrading #a
3	3	4	opensourcedworkplace	sfad	340	#iot #cre#workplace #CDO
4	4	5	crea.vision	Ever missed a call while your phone was chargi...	304	#instamachinelearning #in

In [13]:

```
## Checking Rows & Columns Availabale in Dataset
data.shape
```

Out[13]:

```
(100, 8)
```

In [14]:

```
## Checking Details Information related with Dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Unnamed: 0         100 non-null    int64 
 1   S.No               100 non-null    int64 
 2   USERNAME           100 non-null    object
 3   Caption            94 non-null     object
 4   Followers          100 non-null    int64 
 5   Hashtags           100 non-null    object
 6   Time since posted  100 non-null    object
 7   Likes              100 non-null    int64 
dtypes: int64(4), object(4)
memory usage: 6.4+ KB
```

In [15]:

```
## Checking All Columns name present in dataset
data.columns
```

Out[15]:

```
Index(['Unnamed: 0', 'S.No', 'USERNAME', 'Caption', 'Followers', 'Hashtag
s',
       'Time since posted', 'Likes'],
      dtype='object')
```

In [16]:

```
## checking top 2 rows of dataset
data.head(2)
```

Out[16]:

	Unnamed: 0	S.No	USERNAME	Caption	Followers	
0	0	1	mikequindazzi	Who are #DataScientist and what do they do? >>...	1600	#MachineLearning #AI ;
1	1	2	drgorillapaints	We all know where it's going. We just have to ...	880	#deck .#mac #macintosh#sayhello #

In [17]:

```
# Remove unnecessary columns
data= data.drop(['Unnamed: 0', 'S.No'], axis=1)
```

In [18]:

```
## Checking All Columns name present in Dataset
data.columns
```

Out[18]:

```
Index(['USERNAME', 'Caption', 'Followers', 'Hashtags', 'Time since poste
d',
       'Likes'],
      dtype='object')
```

In [19]:

```
## Checking top 3 rows of dataset after droping unnecessary columns.
data.head(3)
```

Out[19]:

	USERNAME	Caption	Followers	Hashtags	p
0	mikequindazzi	Who are #DataScientist and what do they do? >>...	1600	#MachineLearning #AI #DataAnalytics #DataScien...	
1	drgorillapaints	We all know where it's going. We just have to ...	880	#deck .#mac #macintosh#sayhello #apple #steve...	
2	aitrading_official	Alexander Barinov: 4 years as CFO in multinati...	255	#whoiswho #aitrading #ai #aitradingteam#instat...	

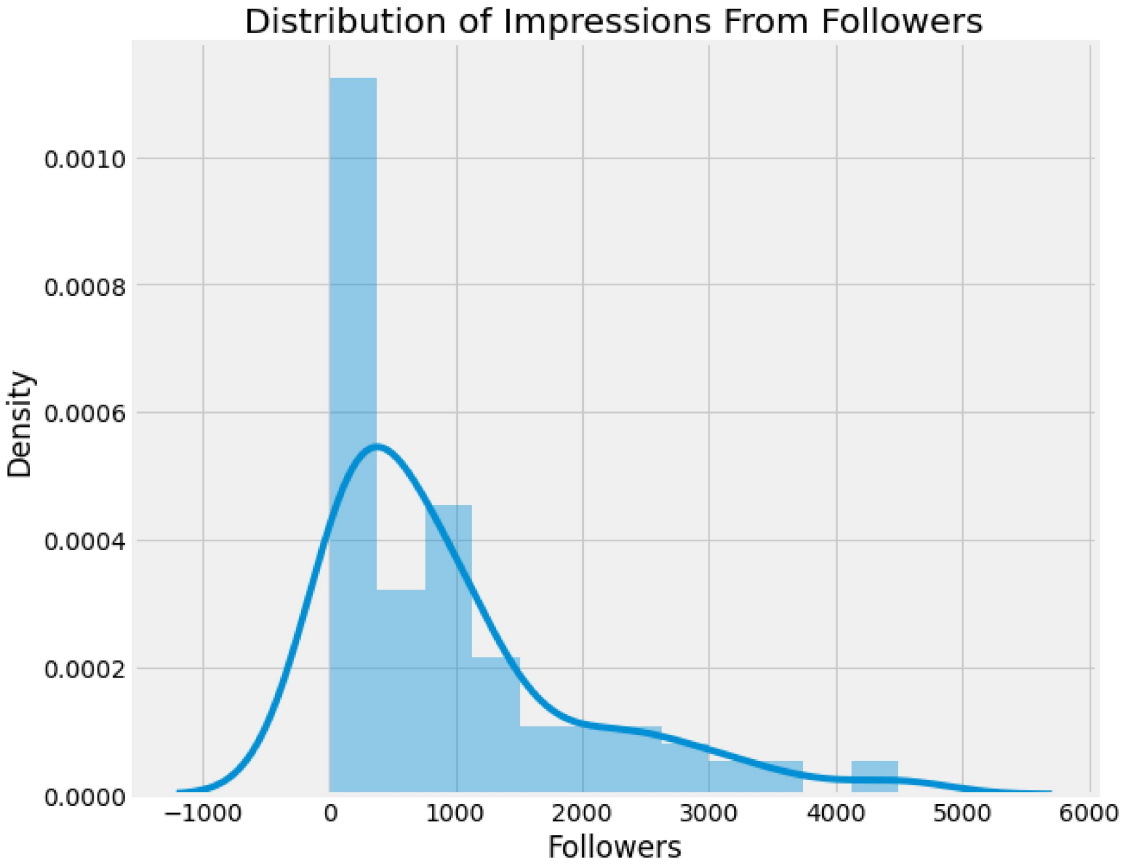
Doing EDA and Analyzing Instagram Reach

In [20]:

```
## Distribution of Impressions From Followers
plt.figure(figsize=(10, 8))
plt.style.use('fivethirtyeight')
plt.title("Distribution of Impressions From Followers")
sns.distplot(data['Followers'])
plt.show()

```

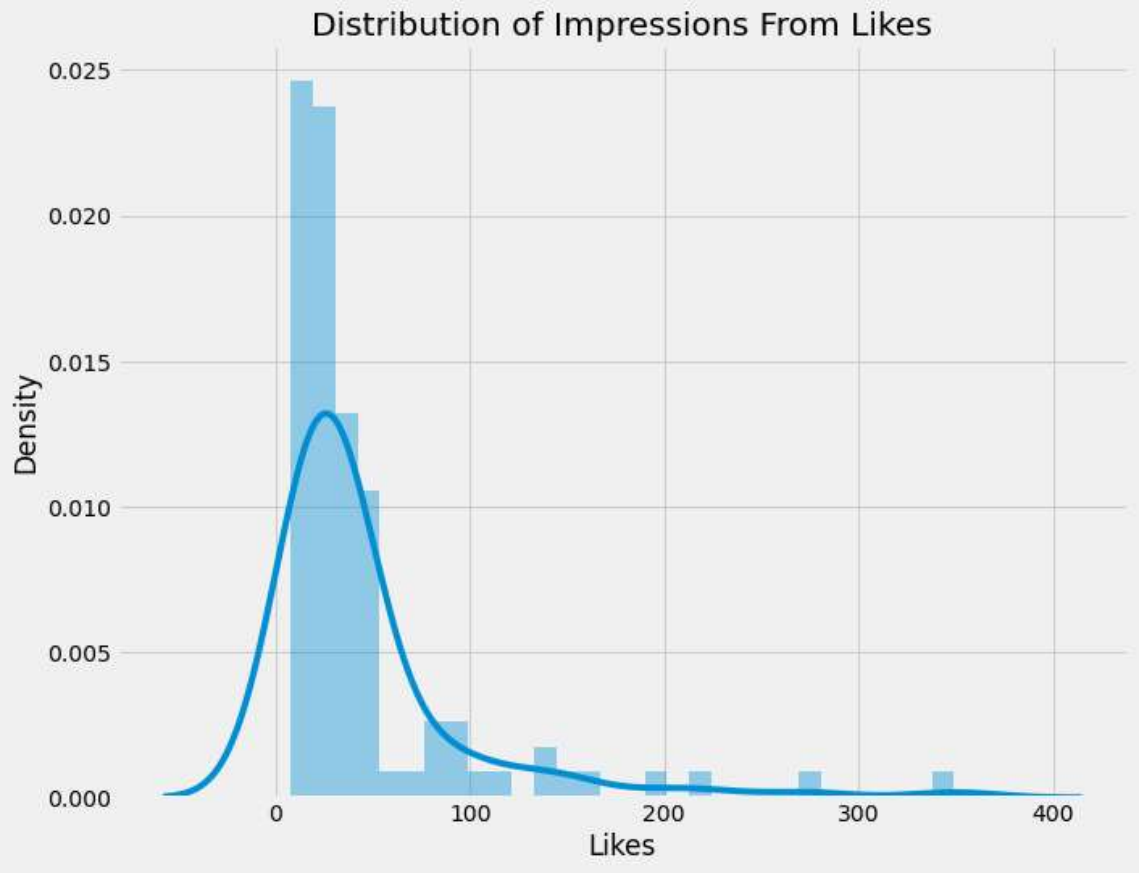
```
C:\Users\abhik\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in
a future version. Please adapt your code to use either `displot` (a figur
e-level function with similar flexibility) or `histplot` (an axes-level f
unction for histograms).
  warnings.warn(msg, FutureWarning)
```



In [21]:

```
## Distribution of Impressions From Likes
plt.figure(figsize=(10, 8))
plt.title("Distribution of Impressions From Likes")
sns.distplot(data['Likes'])
plt.show()
```

```
C:\Users\abhik\anaconda3\lib\site-packages\seaborn\distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in
a future version. Please adapt your code to use either `displot` (a figur
e-level function with similar flexibility) or `histplot` (an axes-level f
unction for histograms).
  warnings.warn(msg, FutureWarning)
```



In [25]:

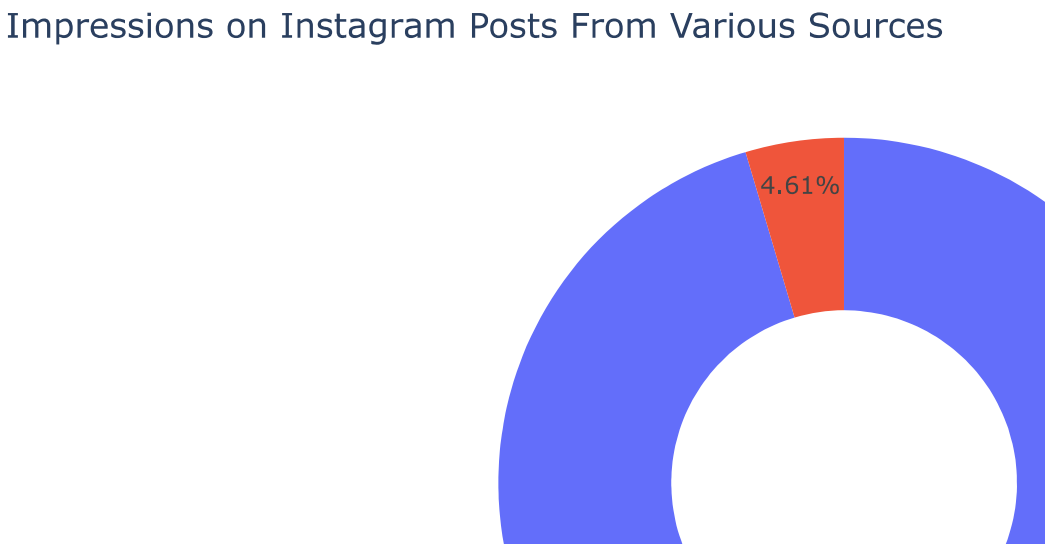
```
## Relation between Likes and Followers

followers = data["Followers"].sum()
likes = data["Likes"].sum()

labels = ['Followers', 'Likes']
values = [followers, likes]

fig = px.pie(data, values=values, names=labels,
             title='Impressions on Instagram Posts From Various Sources', hole=0.5)
fig.show()


```



In [37]:

```
from wordcloud import WordCloud

# Provide the path to the TrueType font file
font_path = "/path/to/your/font.ttf"

# Create a WordCloud object with the specified font
wordcloud = WordCloud(font_path=font_path)
```

In [ ]:

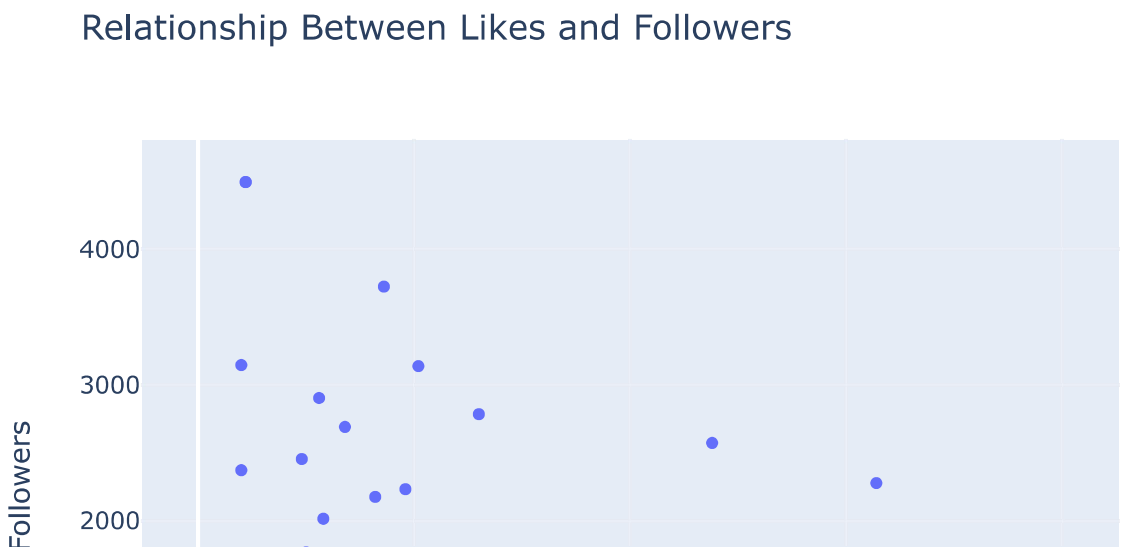
```
## Plotting Word-Cloud for Hashtag Related Data

text = " ".join(i for i in data.Hashtags)
stopwords = set(STOPWORDS)
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(text)
plt.style.use('classic')
plt.figure( figsize=(12,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

In [27]:

```
## Plotting Scatter-plot for showing Relationship Between Likes and Followers

figure = px.scatter(data_frame = data, x="Likes",
                    y="Followers", trendline="ols",
                    title = "Relationship Between Likes and Followers")
figure.show()
```



In [28]:

```
# Select the relevant features and target variables

features = ['USERNAME', 'Caption', 'Hashtags', 'Followers']
target_likes = 'Likes'
target_time_since_posted = 'Time since posted'

```

In [29]:

```
# Split the data into training and testing sets

X = data[features]
y_likes = data[target_likes]
y_time_since_posted = data[target_time_since_posted]
X_train, X_test, y_likes_train, y_likes_test, y_time_since_posted_train, y_time_sin

```

In [30]:

```
# Preprocess the text features using one-hot encoding
encoder = OneHotEncoder(sparse=False, handle_unknown='ignore')
X_train_encoded = encoder.fit_transform(X_train)
X_test_encoded = encoder.transform(X_test)

```

Train a model to predict the number of likes:

In [31]:

```
# Train a model to predict the number of likes
likes_model = LinearRegression()
likes_model.fit(X_train_encoded, y_likes_train)
likes_predictions = likes_model.predict(X_test_encoded)
likes_mse = mean_squared_error(y_likes_test, likes_predictions)
print("Mean Squared Error (Likes):", likes_mse)
```

```
Mean Squared Error (Likes): 1639.6633407976692
```

Train a model to predict the time since posted

In [32]:

```
# Preprocess the time since posted variable
def extract_numerical_value(time_string):
    numerical_value = re.findall(r'\d+', time_string)[0]
    return int(numerical_value)

```

In [33]:

```
y_time_since_posted_train = y_time_since_posted_train.apply(extract_numerical_value
y_time_since_posted_test = y_time_since_posted_test.apply(extract_numerical_value)

```

In [34]:

```
# Train a model to predict the time since posted
time_since_posted_model = LinearRegression()
time_since_posted_model.fit(X_train_encoded, y_time_since_posted_train)
time_since_posted_predictions = time_since_posted_model.predict(X_test_encoded)
time_since_posted_mse = mean_squared_error(y_time_since_posted_test, time_since_pos
print("Mean Squared Error (Time Since Posted):", time_since_posted_mse)
```

```
Mean Squared Error (Time Since Posted): 12.861392010243321
```

In [ ]:

```

```