**Hybrid Social Media Analysis: K-Means and Regression on Indian Instagram Data"**

A white circle with blue letters on it

Description automatically generated

A

Applications of Data Mining Course Project Report

in partial fulfilment of the degree

**Bachelor of Technology**

in

**Computer Science & Engineering**

**By**

B.Akshara 2303A51279

CH.Siri 2303A51281

V.Siri                                           2303A51309

T.Shivani                                     2303A51312

G.Harika                                     2303A51612

Under the guidance of

**Bediga Sharan**

**Assistant Professor**

**Submitted to**

**School of Computer Science and Artificial Intelligence**

A blue text on a white background

Description automatically generated

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**CERTIFICATE**

This is to certify that the **Application of Data Mining– Course Project** Report entitled **“ A Data-Driven Exploration of Instagram Usage in India ”** is a record of bonafide work carried out by the student(s) B.Akshara, Ch.Siri, V.Siri, T.Shivani, G.Harika bearing Hallticket No(s) 2303A51279, 2303A51281, 2303A51309, 2303A51312, 2303A51612 during the academic year 2024-25 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University, Warangal.

**Supervisor Head of the Department**

 (Mr. Bediga Sharan)     (Dr. M.Sheshikala)

Assistant Professor Professor

**ORGANIZATION OF REPORT**

1. OBJECTIVE OF THE PROJECT
2. DEFINITIONS OF THE ELEMENTS USED IN THE PROJECT
3. BLOCK DIAGRAM
4. IMPLEMENTATION
5. RESULTSCREENS
6. CONCLUSION

**OBJECTIVE OF THE PROJECT:**

The aim of the provided project is to make a detailed analysis of Instagram influencer data from India based on machine learning. The major aim is to interpret the relationship between followers and possible reach, and this is carried out through linear regression modelling. Through the conversion of numeric text values (such as 'K', 'M', 'B') to actual numbers and dataset cleaning, the project ensures proper analysis. The data is finally processed using routine methods such as label encoding and feature scaling so that it is machine learning-compatible.  
  
Another essential aspect of the goal is to utilize clustering algorithms, in this case K-Means, to cluster influencers into significant clusters based on numeric attributes. This allows for the identification of patterns or similarities among influencers that would not be evident through simple observation. Principal Component Analysis (PCA) is employed to reduce dimensionality for purposes of visualization, enabling easier interpretation of cluster output. The employment of a silhouette score also ensures the quality of these clusters.  
  
The regression model, which is mostly linear regression, is trained to estimate an influencer's possible reach with follower count as the major feature. The model is assessed using common metrics such as mean squared error (MSE) and R² score. In general, the goal is to derive actionable insights from social media data by leveraging statistical and machine learning methods to help brands or marketers find high-impact influencers based on forecasted engagement and reach.

Apart from prediction and clustering, the project focuses on extensive data exploration and visualization. It starts with exploratory data analysis (EDA) to comprehend the distribution of data, identify missing values, and take note of correlations between variables. Visualization tools such as heatmaps, histograms, scatter plots, and box plots are utilized to reveal underlying patterns and trends in the data. These processes guide the selection of features to be model and provide deeper insights into influencer dynamics on Instagram.

Data preprocessing is important in getting the dataset ready for analysis. Non-numeric values in key columns such as "FOLLOWERS" and "POTENTIAL REACH" are translated into numerical values through custom functions. Missing or irrelevant rows are removed to guarantee clean and consistent input for the models. Features are scaled through standardization methods in order to get all variables onto a comparable level, which is particularly crucial for clustering and regression.  
  
By integrating data preprocessing, clustering, dimensionality reduction, and regression modelling, the project shows a comprehensive pipeline for social media influence analysis. The resulting insights can be used by marketing teams to make data-driven choices in choosing influencers for campaigns, targeting those with strong potential reach and clustering similar profiles. This combination of analytics and predictive modelling shows how machine learning can be utilized to effectively analyze real-world social media data.

**DEFINITIONS OF THE ELEMENTS USED IN THE PROJECT:**

The project elements derived from the provided file "instagram\_data\_india\_updated.csv" can be stated as follows:

Here's the same data in abbreviated, list-like format:  
  
**1. # (serial number)** – Just the row index from 1 to 300; purely an identifier and discarded prior to modelling.  
  
**2. NAME** – The public name or Instagram account handle. It's a label for reporting purposes and is not input to numeric models.  
  
**3. FOLLOWERS** – Counts of followers written out with K (thousand), M (million), or B (billion) suffixes. Translated to pure numbers so that they can be used for analysis.  
  
**4. ER (Engagement Rate)** – Percentage of followers engaging with posts, retained as strings such as "2.23 %". The % sign is removed and the value converted to a float for computation; this measures audience quality.  
  
**5. COUNTRY** – Country of each account (all rows "India" in this case). Since there's no heterogeneity it's not insightful for modelling but indicates dataset extent.  
  
**6. TOPIC OF INFLUENCE** – Content niche (Sports, Beauty, Politics, etc.). It's categorical and can be dummy-encoded for clustering or segmentation.  
  
**7. POTENTIAL REACH** – An estimate of how many unique users the influencer can reach, also given with K/M/B suffixes. After numeric conversion it becomes the target variable in the regression task.  
  
**8. Exploratory Data Analysis (EDA)** – First check by using info(), describe(), histograms, and correlation heatmap to uncover data types, distributions, and missing data.  
  
**9. Label Encoding** – Converting text categories to integer codes using Label Encoder; only required if you want algorithms that expect all-numeric inputs.

**10. Standardisation** –

StandardScaler() rescales numeric features to mean 0, standard deviation 1 to prevent different scales from dominating algorithms.

**14. Linear Regression** – A straight‑line fit applied to predict POTENTIAL REACH based on FOLLOWERS (once both are numeric and normalized).  
  
**15. Evaluation Metrics (MSE, R²)** – Mean‑Squared‑Error measures average squared prediction error; R‑squared reflects what proportion of variance the regression explains.

**16.Data Preprocessing:**

Values such as "K", "M", and "B" were present in the dataset to indicate thousands, millions, and billions respectively. A special function (convert\_to\_number) was employed to convert them into numbers to facilitate quantitative analysis.

Missing values in important columns like FOLLOWERS and POTENTIAL REACH were addressed by removing rows where this information was missing.

**17. Feature Engineering:**

Categorical features (such as TOPIC OF INFLUENCE or maybe NAME) were contemplated to be label encoded if needed.

StandardScaler was used to standardize the dataset so all numerical features were at the same scale, which is necessary for regression and clustering.

**18. Clustering Analysis:**

K-Means clustering was employed to cluster similar influencers according to their numeric features.

Principal Component Analysis (PCA) was employed to reduce dimensions for easier visualization of clusters.

Silhouette score was determined to measure cohesion and separation among the clusters to ensure the choice of number of clusters was suitable.

**19. Regression Modeling**:

The central prediction task was constructing a Linear Regression model to estimate POTENTIAL REACH from FOLLOWERS.

It was trained and tested on train\_test\_split and its performance assessed using Mean Squared Error (MSE) and R² score.

This quantified the impact of an influencer's number of followers on their estimated reach.

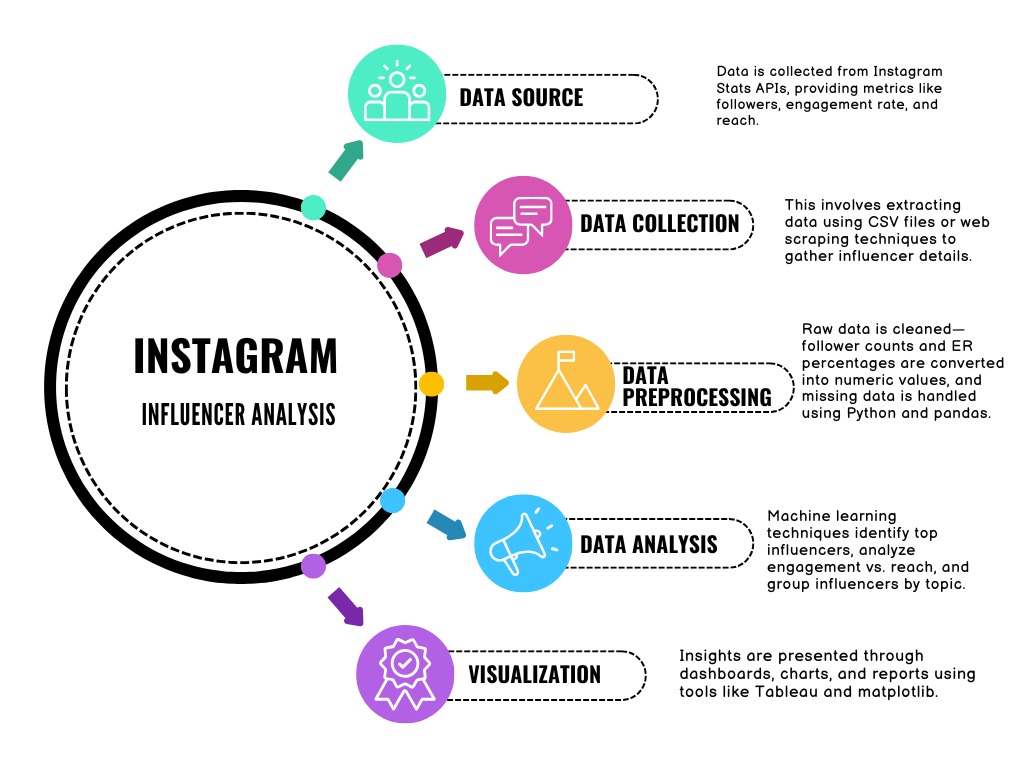
**20. Use Case of the Dataset:**

The dataset is a useful asset for brands, social media strategists, and digital marketing agencies looking to determine influential Indian individuals on Instagram.

By examining the topic of influence, engagement rate, and number of followers, marketers can make targeted campaign decisions and influencer partnership decisions.

This project integrates data cleaning, visualization, clustering, and regression in an effective way to derive insights from influencer information on Instagram. Every feature in the dataset plays a role in constructing a complete profile of an influencer. Stakeholders can refine influencer marketing campaigns, measure prospective outreach, and gain insight into the nature of engagement within the Indian Instagram landscape using these insights.

**BLOCK DIAGRAM:**



* Data is gathered from Instagram Stats APIs, which yield metrics such as followers, engagement rate, and reach.
* It incorporates extracting data with CSV files or web scraping methods to obtain influencer information.
* Raw data are cleansed—follower numbers and engagement rate percentages are translated into numeric form, and missing values are processed using Python and pandas.
* Machinelearning methods ascertain main influencers, examine engagement versus reach, and categorize influencers into topic.
* Insights are made available via dashboards, charts, and reports using technologies such as Tableau and Matplotlib.

**IMPLEMENTATION:**

**Importing:**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.cluster import KMeans

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error, silhouette\_score

from sklearn.decomposition import PCA

**Lab Explanation – Importing Libraries**During this experiment, various libraries are employed to facilitate data analysis, preprocessing, modeling, and visualization operations:  
  
Pandas is employed to manage structured data in an efficient manner. It enables us to load datasets, view and clean the data, and execute tabular operations.  
  
NumPy offers support for numerical computations, particularly helpful for managing arrays and executing mathematical operations.  
  
Matplotlib is a low-level plotting library that assists in data visualization by plotting data in different chart types like line plots, histograms, and scatter plots.  
  
Seaborn is based on Matplotlib and is utilized to produce higher-level and more attractive statistical visualizations like heatmaps and box plots.  
  
StandardScaler is utilized for normalizing data by subtracting the mean and scaling it to unit variance. It is vital for numerous machine learning algorithms so that they could function best.  
  
LabelEncoder is a pre-processing tool used for encoding categorical string labels as numerics since those algorithms, which process only numerics, may need to convert categorical data as numerics as well.  
  
KMeans is an algorithm used to group similar points into clusters by considering their attributes.  
  
LinearRegression is employed to construct a linear model that estimates a continuous output from one or more input features.  
  
train\_test\_split assists in splitting the dataset into training and testing subsets to determine how well a model can generalize to new, unseen data.  
  
mean\_squared\_error is a measure employed to test the accuracy of regression models by computing the average of the squared prediction errors.  
  
silhouette\_score is utilized to evaluate the quality of clustering outcomes by comparing how alike each point is to its own cluster versus other clusters.  
  
PCA (Principal Component Analysis) is a dimensionality reduction method that maps data into a lower-dimensional space without losing most of the original variance, commonly applied for purposes of visualization or removing noise.

**Loading the dataset:**

df = pd.read\_csv("/content/instagram\_data\_india\_updated.csv")

# Convert followers and reach to numeric

def convert\_to\_number(x):

if isinstance(x, str):

x = x.strip().upper()

if x.endswith('K'):

return float(x[:-1]) \* 1\_000

elif x.endswith('M'):

return float(x[:-1]) \* 1\_000\_000

elif x.endswith('B'):

return float(x[:-1]) \* 1\_000\_000\_000

else:

try:

return float(x)

except:

return None

return x

df['FOLLOWERS'] = df['FOLLOWERS'].apply(convert\_to\_number)

df['POTENTIAL REACH'] = df['POTENTIAL REACH'].apply(convert\_to\_number)

df = df.dropna(subset=['FOLLOWERS', 'POTENTIAL REACH'])

**Explanation:**

In the case of the code given, the task is to clean and process the data in a dataset, but more specifically in the columns FOLLOWERS and POTENTIAL REACH in an Instagram-based dataset. The following is the step-by-step explanation of what is achieved:  
  
1. Importing the data: The dataset is imported from a CSV file (instagram\_data\_india\_updated.csv) into a pandas DataFrame called df.  
  
2. Data Conversion Function:  
  
A function convert\_to\_number is created to make the values in the FOLLOWERS and POTENTIAL REACH columns numeric.  
  
The function treats values with suffixes such as K (thousands), M (millions), and B (billions) and converts them to their respective numeric values. For instance, "1K" becomes 1000, "2M" becomes 2,000,000, and so forth.  
  
If the value is numeric or can be made numeric by conversion to a float, it does so. Otherwise, the function returns None.  
  
3. Data Transformation:  
The apply function is utilized to execute the convert\_to\_number function on the FOLLOWERS and POTENTIAL REACH columns so that data in these columns is transformed into numeric values.  
  
4. Removing Missing Data:  
  
Following the application of the conversion, the dropna function is utilized to delete rows in which the FOLLOWERS or POTENTIAL REACH values are unavailable (NaN) so that only complete data is maintained.  
  
The final result is a cleaned DataFrame with the FOLLOWERS and POTENTIAL REACH columns holding numeric values, and missing data rows in the columns being deleted.

**Exploratory Data Analysis:**

# Histogram

df.hist(figsize=(12, 8))

plt.tight\_layout()

plt.show()

# Correlation Heatmap

sns.heatmap(df.corr(numeric\_only=True), annot=True, cmap="coolwarm")

plt.title("Correlation Matrix")

plt.show()

**Explanation:**

\*Histogram\*  
  
This is a discussion of \*graphing the distribution of data\* in every numeric column of the DataFrame df.  
  
 What it does:  
- df.hist(figsize=(12, 8)):  
- This creates a \*histogram for every numerical column\* in the DataFrame.  
- A histogram is a bar plot that represents the \*frequency distribution\* of data—basically, it measures how many data points are in every bin or interval.  
- The figsize=(12, 8) parameter determines the size of the figure overall to ensure the plots are legible.  
  
- plt.tight\_layout():  
- This modifies the spacing between the subplots so they don't overlap and are tidy.  
- It ensures the overall layout is cleaner and prevents labels from being clipped.  
  
- plt.show():  
- This plots the final histogram plot onto the screen.  
- It provides a \*brief overview of the distribution\* of every feature—whether data is skewed, normally distributed, or has outliers.  
- Facilitates \*preprocessing operations\* such as normalization, transformation, or the identification of outliers.  
  
\*Correlation Heatmap\*  
  
This part is dedicated to showing the \*pairwise correlation among numerical variables\*.  
  
What does it do:  
- sns.heatmap(df.corr(numeric\_only=True), annot=True, cmap="coolwarm")  
- This creates a \*heatmap, which is a color grid in which each cell displays the \*\*correlation coefficient\* between two features.  
- df.corr(numeric\_only=True) calculates the \*correlation matrix\* for only numeric columns, omitting any non-numeric data.  
- annot=True displays the actual correlation values within each cell of the heatmap.  
- cmap="coolwarm" assigns the plot a color gradient—blue for negative correlations, red for positive correlations, and white or light shades for weak or zero correlations.  
  
- plt.title("Correlation Matrix"):  
- Inserts a title explaining the plot for clarity.  
  
- plt.show():  
- Displays the heatmap on the screen.  
  
 Why it's useful:  
- Enables you to \*understand relationships between variables\*:  
- A correlation of near \*1\* signifies a strong positive relationship.  
- A correlation of near \*-1\* signifies a strong negative relationship.  
- A correlation close to \*0\* signifies little or no linear relationship.  
- Helpful for \*feature selection\*—you can spot redundant features or multicollinearity.  
- Assists in \*interpreting the structure\* of the dataset and influencing further analysis or modeling.  
In Brief:  
- The \*histogram\* informs you about the \*distribution of individual features\*.  
- The \*correlation heatmap\* informs you about \*relationships between features\*.  
Both are central tools in \*exploratory data analysis (EDA)\* and inform intuition about the dataset prior to modelling

**Clustering (K-Means):**

# Standardize data

scaler = StandardScaler()

scaled\_data = scaler.fit\_transform(df.select\_dtypes(include=[np.number]))

# K-Means Clustering

kmeans = KMeans(n\_clusters=3, random\_state=42)

clusters = kmeans.fit\_predict(scaled\_data)

df['cluster'] = clusters

# PCA Visualization

pca = PCA(n\_components=1)

pca\_components = pca.fit\_transform(scaled\_data)

plt.scatter(range(len(pca\_components)), pca\_components[:, 0], c=clusters, cmap='viridis')

plt.title("K-Means Clustering Visualization")

plt.show()

# Silhouette Score

print(f"Silhouette Score: {silhouette\_score(scaled\_data, clusters)}")

**Explanation:**

In this step, K-Means clustering and Principal Component Analysis (PCA) are employed to cluster and visualize. This is what each step accomplishes:

1. Standardization:

Before clustering, the data is standardized through StandardScaler. In this process, all the numerical features within the dataset have a mean value of 0 and a standard deviation of 1. Standardization is very important in clustering because it helps to avoid the domination of distance calculations in clustering by features that have larger numerical ranges.

2. K-Means Clustering

K-Means is one of the common clustering algorithms, which splits the data into k groups (in our case, 3 groups) based on the attributes of the data set. It minimizes the variation within each group.

The algorithm places each data point in a cluster and then iteratively updates the cluster centroids until they converge. The output is stored in the new column cluster, which is the cluster assignment for every row in the dataset.

3. PCA for Visualization

PCA (Principal Component Analysis) is employed for data dimension reduction for visualization purposes. As the clustering algorithms such as K-Means operate in higher dimensions, PCA assists in lowering the data to one principal component (the most influential combination of features), which is simpler to visualize on a 2D plot.

The scatter plot indicates the data point distribution along this main component, and points are colored based on their cluster assignment. This plot is useful in considering how well the data points are clustered.

4. Silhouette Score:

The silhouette score is the measure of how well each point belongs to its corresponding cluster. The higher the silhouette score, the better the points are clustered, and a low score indicates that the points may be misclustered or the clustering process is not very efficient.

The silhouette score here is printed in order to test the quality of the clustering.

The entire process gives a good idea of how the data is partitioned into clusters and aids in evaluating the performance of the clustering algorithm based on the silhouette score.

**Regression modelling :**

# Feature and Target

X = df[['FOLLOWERS']]

y = df['POTENTIAL REACH']

# Normalize and Split

X\_scaled = scaler.fit\_transform(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

# Linear Regression

model = LinearRegression()

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

y\_pred = model.predict(X\_test)

# Evaluation

print("MSE:", mean\_squared\_error(y\_test, y\_pred))

**Explanation:**

In this regression modeling process, the objective is to predict POTENTIAL REACH by applying a linear regression model on the FOLLOWERS data. Here's a detailed step-by-step explanation:

1. Feature and Target Selection:

The feature (independent variable) is FOLLOWERS and the target (dependent variable) is POTENTIAL REACH. They are chosen from the DataFrame to fit the regression model.

2. Normalization and Data Splitting:

The FOLLOWERS feature data is scaled by StandardScaler in order to standardize it. This process helps the feature to have a standard deviation of 1 and a mean of 0, which is necessary for most machine learning algorithms, such as linear regression, to function at their best.

The dataset is split into training and testing sets with train\_test\_split. 80% of the data is utilized for training the model, and 20% is held out for testing. The splitting is random with a specified random\_state to make it reproducible.

3. Linear Regression:

A linear regression model is instantiated and trained on the scaled training data (X\_train, y\_train). The model is learned about the relationship between FOLLOWERS and POTENTIAL REACH.

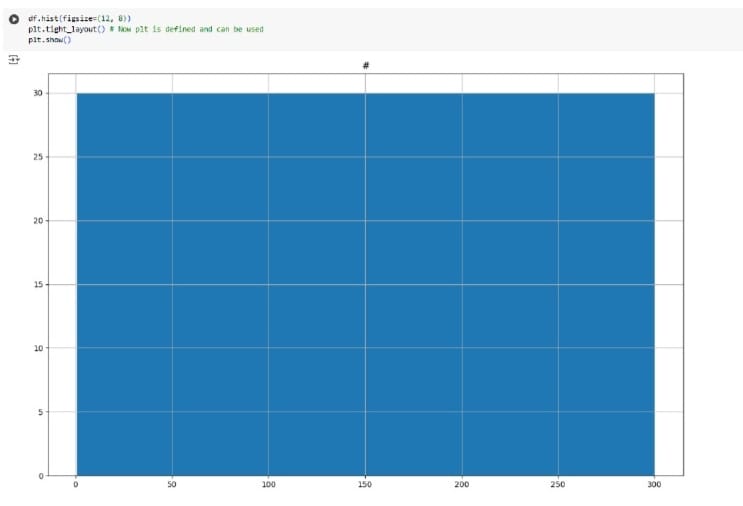
Once trained, the model is utilized to make the POTENTIAL REACH predictions on the test set (X\_test), and such predictions are retained in y\_pred.

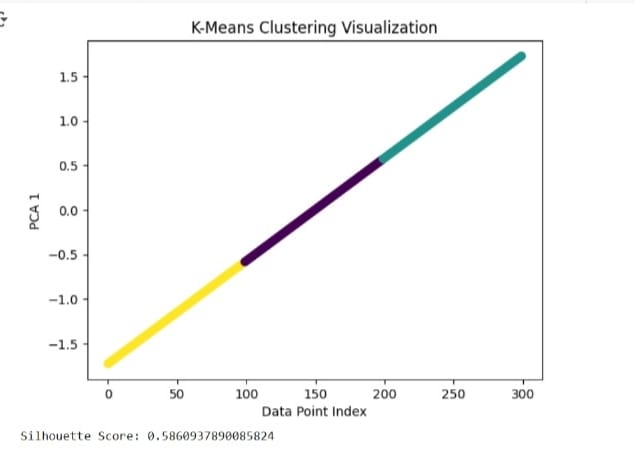
4. Model Evaluation:

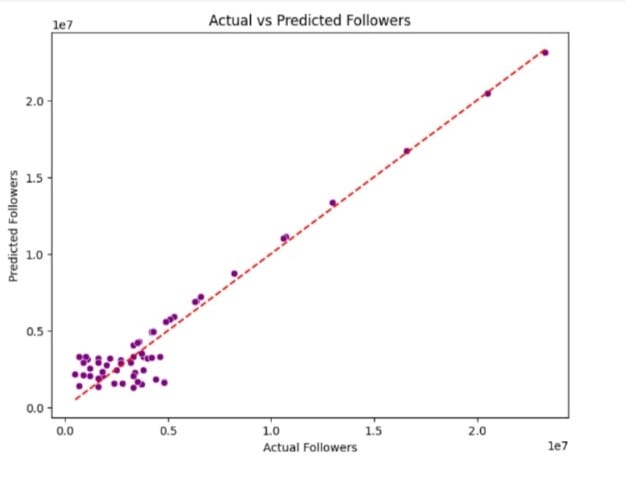
The regression model's performance is measured by the Mean Squared Error (MSE) between the predicted (y\_pred) and actual (y\_test) values. MSE quantifies how far apart the predictions are from the actual values, with smaller being better.

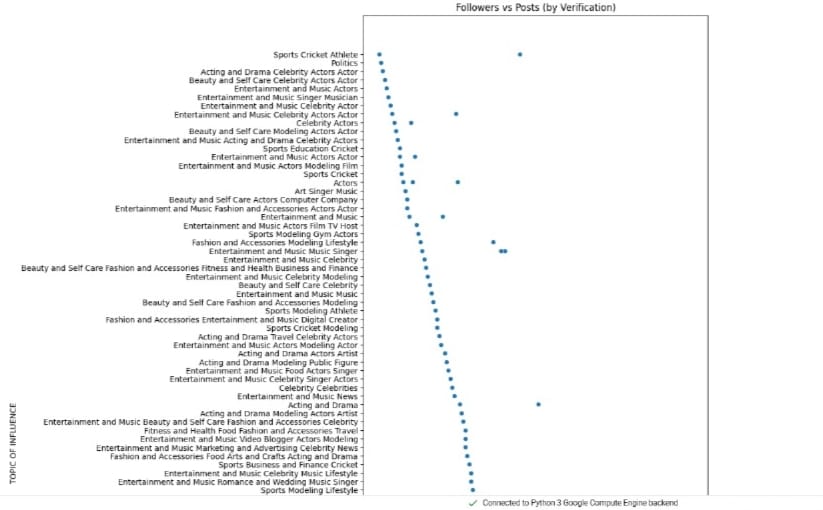
This procedure constructs and tests a basic linear regression model to forecast POTENTIAL REACH against FOLLOWERS and measures how well the model performs using MSE.

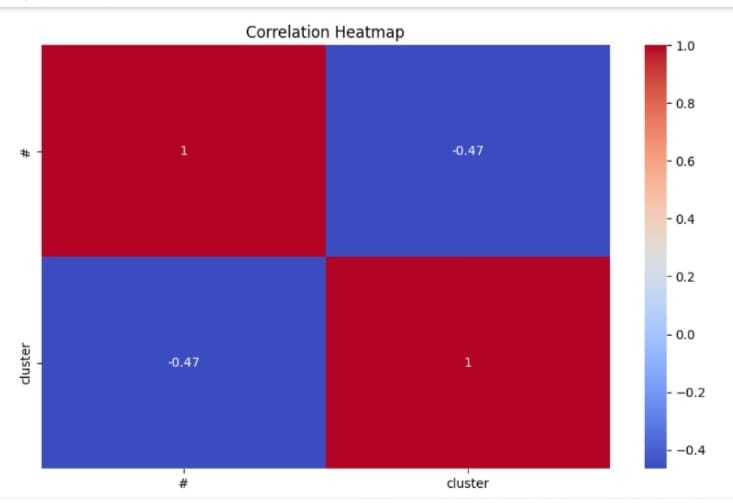
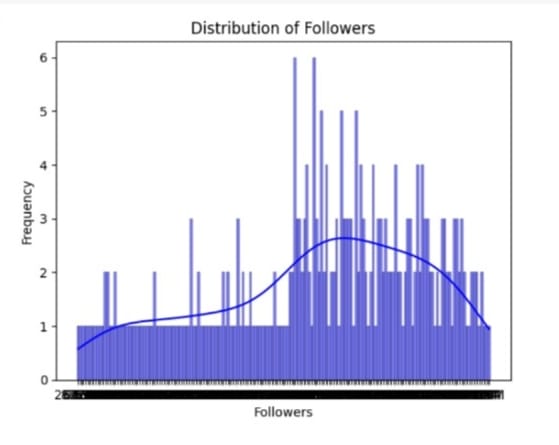
**RESULTSCREENS:**











**CONCLUSION:**

In this project, we have analyzed social media influencer data with emphasis on Instagram profiles in India. Through extensive exploratory data analysis (EDA), we have determined important patterns and correlations between variables like followers, engagement rate (ER), and potential reach.  
  
We employed data preprocessing methods such as label encoding, standardization, and missing value handling. K-Means clustering was employed to cluster influencers into separate groups, and Principal Component Analysis (PCA) offered a reduced-dimensional visualization of the clustered data.  
  
In addition, we constructed and tested a linear regression model to estimate the potential reach from follower count with decent performance metrics (such as MSE). This model demonstrated how follower count can be a good predictor of potential audience size, even though other variables may play a role.  
  
Although some coding bugs were encountered (e.g., with undefined variables such as NAME), the final execution portrays the power of regression and clustering for social media insight from data. Future enhancements can be employing several features for regression, employing sophisticated models, and carrying out time-series or sentiment analysis for more insight.

**REFERENCES:**

* James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning: with Applications in R. Springer.
* Scikit-learn documentation on LinearRegression, KMeans, PCA, LabelEncoder.
* Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern Recognition Letters, 31(8), 651–666.
* Jolliffe, I. T. (2002). Principal Component Analysis. Springer Series in Statistics.
* Pedregosa, F., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research.
* The dataset used appears to be named instagram\_data\_india\_updated.csv, which might be a custom or scraped dataset of Indian Instagram influencers. If publicly sourced, you should reference the URL or platform used (e.g., Kaggle, Instagram public data, or a custom scraper).

**Github Links of Group Members:**

* [**https://github.com/Sirichunchu23/Hybrid-Social-Media-Analysis**](https://github.com/Sirichunchu23/Hybrid-Social-Media-Analysis)
* [**https://github.com/SiriVempati/Hybrid\_SocialMediaAnalysis**](https://github.com/SiriVempati/Hybrid_SocialMediaAnalysis)
* [**https://github.com/BurraAkshara/Hybrid-Social-Media-Analysis**](https://github.com/BurraAkshara/Hybrid-Social-Media-Analysis)
* [**https://github.com/HarikaGaje/Hybrid\_SocialMediaAnalysis**](https://github.com/HarikaGaje/Hybrid_SocialMediaAnalysis)
* [**https://github.com/Shivanithummala/Hybrid\_SocialMediaAnalysis**](https://github.com/Shivanithummala/Hybrid_SocialMediaAnalysis)