In [1]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report, mean_squared_error, r2_scor
```

In [14]:
```
file_path = "C:\\Users\\HP\\Desktop\\Copy of Final Lead Data.csv"
```

In [15]:
```
try:
    df = pd.read_csv(file_path, encoding='utf-8')
except UnicodeDecodeError:
    df = pd.read_csv(file_path, encoding='latin-1')

# Now 'df' contains your DataFrame with the data from the CSV file
```

In [16]:
```
df
```

Out[16]:

	ID	First Name	Email	Gender	City	Created	Position	New College Name	Colleges	Academic Year	Branch/ Specialisation	Other Branch
0	68112	ANIKET	aniket@xyz.com	NaN	NaN	04/27/2022 01:41:38 pm	NaN	NaN	NaN	NaN	NaN	NaN
1	68110	Dhanshree	dhanshree@xyz.com	NaN	NaN	04/22/2022 04:08:38 pm	NaN	Lords Universal College	NaN	NaN	NaN	NaN
2	68108	Dhiraj	dhiraj@xyz.com	NaN	NaN	04/16/2022 10:31:59 pm	NaN	NaN	NaN	NaN	NaN	NaN
3	68106	Pooja	pooja@xyz.com	NaN	NaN	04/13/2022 10:05:15 pm	NaN	NaN	NaN	NaN	NaN	NaN
4	68090	Aayush	aayush@xyz.com	NaN	NaN	03/26/2022 07:02:48 pm	NaN	B.k Birla college	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
5298	25834	Pratik	pratik@xyz.com	NaN	NaN	10/16/2019 10:19:42 am	NaN	NaN	NaN	NaN	NaN	NaN
5299	25832	Nikita	nikita@xyz.com	NaN	NaN	10/16/2019 10:19:41 am	NaN	NaN	NaN	NaN	NaN	NaN
5300	25830	Ashwini	ashwini@xyz.com	NaN	NaN	10/16/2019 10:19:10 am	NaN	NaN	NaN	NaN	NaN	NaN
5301	25828	Jheanna Mae	jheannamae@xyz.com	NaN	NaN	10/16/2019 10:19:08 am	NaN	Don Bosco College of Engineering, Fatorda, Margao	Others	3.0	Computer Science	NaN
5302	25826	Krishna	krishna@xyz.com	NaN	NaN	10/16/2019 10:19:06 am	NaN	Don Bosco College Of Engineering	Others	3.0	Computer Science	NaN

5303 rows × 18 columns

In [19]:
```python
df.columns
```

Out[19]:
```
Index(['ID', 'First Name', 'Email', 'Gender', 'City', 'Created', 'Position',
       'New College Name', 'Colleges', 'Academic Year',
       'Branch/ Specialisation', 'Other Branch',
       'What is your current academic year?', 'Company Name/ College Name',
       'Would you like to know more about us and our programs?',
       'Are you interested in knowing more about our events?',
       'Have you recommended Cloud Counselage to anyone?',
       'How did you come to know about this event?'],
      dtype='object')
```

In [21]:
```python
# Data preprocessing
X = df.drop('City', axis=1)
y = df['City']
X.index = y
```

In [22]:
```python
# Label encoding
le = LabelEncoder()
for col in X.columns:
    X[col] = le.fit_transform(X[col])
```

In [23]:
```python
# Dimensionality reduction
X_pca = PCA(n_components=2).fit_transform(X)
```

In [24]:
```python
# Clustering
kmeans = KMeans(n_clusters=2)
kmeans.fit(X_pca)
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

Out[24]:
```
KMeans
KMeans(n_clusters=2)
```

In [25]:
```python
# Regression
X_train, X_test, y_train, y_test = train_test_split(X_pca, y, test_size=0.2, random_state=42)
```

In [44]:
```python
# Calculate the percentage of different interaction types
ID_types = df['ID'].unique()
percentages = {}

for ID_type in ID_types:
    count = df['ID'].sum()
    total = df['ID'].nunique()
    percentage = (count / total) * 100
    percentages[ID_type] = percentage
# Print or use the percentages dictionary as needed
print(percentages)
```

```
4366778.219875542, 66762: 4366778.219875542, 66760: 4366778.219875542, 66758: 4366778.219875542, 66756: 43667
78.219875542, 66754: 4366778.219875542, 66752: 4366778.219875542, 66750: 4366778.219875542, 66748: 4366778.21
9875542, 66746: 4366778.219875542, 66632: 4366778.219875542, 66630: 4366778.219875542, 66628: 4366778.2198755
42, 66626: 4366778.219875542, 66622: 4366778.219875542, 66620: 4366778.219875542, 66618: 4366778.219875542, 6
6616: 4366778.219875542, 66614: 4366778.219875542, 66612: 4366778.219875542, 66610: 4366778.219875542, 66608:
4366778.219875542, 66606: 4366778.219875542, 66604: 4366778.219875542, 66602: 4366778.219875542, 66600: 43667
78.219875542, 66598: 4366778.219875542, 66596: 4366778.219875542, 66594: 4366778.219875542, 66592: 4366778.21
9875542, 66590: 4366778.219875542, 66588: 4366778.219875542, 66586: 4366778.219875542, 66584: 4366778.2198755
42, 66582: 4366778.219875542, 66580: 4366778.219875542, 66578: 4366778.219875542, 66576: 4366778.219875542, 6
6574: 4366778.219875542, 66572: 4366778.219875542, 66570: 4366778.219875542, 66568: 4366778.219875542, 66566:
4366778.219875542, 66564: 4366778.219875542, 66562: 4366778.219875542, 66560: 4366778.219875542, 66558: 43667
78.219875542, 66556: 4366778.219875542, 66554: 4366778.219875542, 66552: 4366778.219875542, 66550: 4366778.21
9875542, 66548: 4366778.219875542, 66546: 4366778.219875542, 66544: 4366778.219875542, 66542: 4366778.2198755
42, 66540: 4366778.219875542, 66538: 4366778.219875542, 66536: 4366778.219875542, 66534: 4366778.219875542, 6
6532: 4366778.219875542, 66530: 4366778.219875542, 66528: 4366778.219875542, 66526: 4366778.219875542, 66524:
4366778.219875542, 66522: 4366778.219875542, 66520: 4366778.219875542, 66518: 4366778.219875542, 66516: 43667
78.219875542, 66514: 4366778.219875542, 66512: 4366778.219875542, 66510: 4366778.219875542, 66508: 4366778.21
9875542, 66506: 4366778.219875542, 66504: 4366778.219875542, 66502: 4366778.219875542, 66500: 4366778.2198755
42, 66498: 4366778.219875542, 66496: 4366778.219875542, 66494: 4366778.219875542, 66492: 4366778.219875542, 6
6488: 4366778.219875542, 66486: 4366778.219875542, 66484: 4366778.219875542, 66482: 4366778.219875542, 66480:
```

In [45]:
```python
# Print the results
for ID_type, percentage in percentages.items():
    print(f"{ID_type}: {percentage}%")
```

```
68112: 4366778.219875542%
68110: 4366778.219875542%
68108: 4366778.219875542%
68106: 4366778.219875542%
68090: 4366778.219875542%
68088: 4366778.219875542%
68086: 4366778.219875542%
68084: 4366778.219875542%
68082: 4366778.219875542%
68080: 4366778.219875542%
68078: 4366778.219875542%
68076: 4366778.219875542%
68074: 4366778.219875542%
68072: 4366778.219875542%
68070: 4366778.219875542%
68068: 4366778.219875542%
68066: 4366778.219875542%
68064: 4366778.219875542%
68030: 4366778.219875542%
```

In [59]:
```python
print(X_test.shape)
print(y_test.shape)
```

```
(1061, 17)
(1061,)
```