PROJECT-CUSTOMER SEGMENTATION USING DATA SCIENCE

PHASE 3

DATASET AND ITS EXPLANATION:

https://www.kaggle.com/datasets/kandij/mall-customers

This dataset we used for this project in phase 3. Below is explanation for the dataset

Customer segmentation is a crucial marketing and business strategy that involves dividing a customer base into distinct groups based on specific characteristics. Here's an explanation of some common variables often used in a customer segmentation dataset:

DEMOGRAPHIC VARIABLES:

Age: Segmenting customers by age can help target products and marketing messages to specific age groups. For example, products for children, teenagers, adults, or seniors may differ significantly.

Gender: Gender-based segmentation can be useful for businesses offering gender-specific products.

Income: Income-based segmentation can help target products and services to customers with different spending power.

Customer id: In customer segmentation, it's common to include a unique customer identifier, often referred to as "Customer ID" or "Customer Number," in the dataset.

Spending score: In customer segmentation, the "spending score" is often an essential feature used for clustering customers based on their spending behaviour

STEPS WE USED IN THIS PROJECT PHASE

IMPORT NECESSARY LIBRARIES

```
import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import IsolationForest
```

LOAD THE DATASET

Loading a dataset for customer segmentation involves reading the data from an external source, such as a CSV file, a database, or an API, into your program or environment for further analysis and segmentation.

CSV File (Pandas in Python): If your dataset is in a CSV file, you can use the pd.read_csv() function from the Pandas library in Python to load the data into a DataFrame.

import pandas as pd data = pd.read_csv('your_dataset.csv')

PREPROCESS THE DATASET

Here are some common preprocessing steps:

Data Exploration: Examine the dataset by checking its structure, the first few rows, data types, and summary statistics.

Handling missing values: You can use methods like data.fillna() to fill in missing data or data.dropna() to remove rows with missing values.

Encoding categorical features: If your dataset contains categorical variables, you can use one-hot encoding or label encoding to convert them into numeric values.

Standardizing numerical features: Standardization ensures that all numerical features have a mean of 0 and a standard deviation of 1.

Outlier Detection: Identify and handle outliers in your data.

CODE:

```
import pandas as pd

from sklearn.preprocessing import StandardScaler

data = pd.read_csv("D:\IBM project\Mall_Customers.csv")

print(data.head())

print(data.info())

data.fillna(data.mean(), inplace=True)

data = pd.get_dummies(data, columns=['Gender'], drop_first=True)

numerical_features = ['Age', 'Annual_Income_(k$)','’Spending_Score’]

scaler = StandardScaler()

data[numerical_features] = scaler.fit_transform(data[numerical_features])

X = data[['Age', 'Annual_Income_(k$)','’Spending_Score’]]

from sklearn.ensemble import IsolationForest

outlier_detector = IsolationForest(contamination=0.05)

data['IsOutlier'] = outlier_detector.fit_predict(X)

data = data[data['IsOutlier'] != -1]
```

OUTPUT:

Head data

	CustomerID	Genre	Age	Annual_Income_(k$)	Spending_Score
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

Info data

None