Name : Shejal Tiwari Div/Roll No:D15C/57

Experiment No: 8

Aim: To implement recommendation system on your dataset using the following machine learning techniques.
o Regression
o Classification
o Clustering
o Decision tree
o Anomaly detection
o Dimensionality Reduction
o Ensemble Methods

Theory:

1. Regression

Regression is a supervised learning technique used to predict continuous values based on input features. It estimates the relationship between dependent and independent variables.

- Linear Regression: Models a straight-line relationship between variables.
- Polynomial Regression: Captures non-linear relationships by introducing polynomial terms.
- Ridge/Lasso Regression: Regularized versions that prevent overfitting.
- Logistic Regression: Used for classification despite the name "regression."

2. Classification

Classification is a supervised learning technique that categorizes input data into predefined classes or labels.

- Binary Classification: Two possible outcomes (e.g., spam vs. not spam).
- Multiclass Classification: More than two categories (e.g., classifying animals as cat, dog, or bird).

- Popular Algorithms: Logistic Regression, Decision Trees, SVM, Random Forest, Neural Networks.

3. Clustering

Clustering is an unsupervised learning technique used to group similar data points together based on patterns. Unlike classification, clusters are not predefined.

- K-Means: Partitions data into K clusters using centroids.
- Hierarchical Clustering: Forms a tree-like structure of nested clusters.
- DBSCAN: Groups based on density, identifying outliers as noise.

4. Decision Tree

A Decision Tree is a tree-like structure where data is split into branches based on feature values. It is used for both classification and regression.

5. Anomaly Detection

Anomaly detection identifies unusual patterns that deviate significantly from normal data. It is widely used in fraud detection, cybersecurity, and medical diagnosis.

- Statistical Methods: Z-score, Gaussian distribution analysis.
- Machine Learning Methods: Isolation Forest, One-Class SVM, Autoencoders.
- Distance-Based Methods: k-Nearest Neighbors (k-NN) for detecting outliers.

6. Dimensionality Reduction

Dimensionality reduction is used to reduce the number of input features while preserving essential information. This helps improve model efficiency and visualization.

- Principal Component Analysis (PCA): Converts correlated features into uncorrelated principal components.

- t-SNE (t-Distributed Stochastic Neighbor Embedding): Useful for visualizing high-dimensional data.
- Autoencoders: Neural networks that learn compressed representations.

7. Ensemble Methods

Ensemble methods combine multiple models to improve accuracy and robustness. They work by aggregating predictions from multiple weak learners.

- Bagging (Bootstrap Aggregating): Example: Random Forest (uses multiple decision trees).
- Boosting: Example: AdaBoost, XGBoost (sequentially improves weak models).
- Stacking: Combines multiple models using another model (meta-learner) to make final predictions.

1.Clustering

Importing Libraries

```
[1] import pandas as pd
    import numpy as np
    from sklearn.preprocessing import StandardScaler
    from sklearn.cluster import KMeans
    import matplotlib.pyplot as plt
    import seaborn as sns
```

Loading dataset

```
df = pd.read_csv("/content/IMDB-Movie-Data.csv")

# Drop rows with missing important values
df.dropna(subset=['Genre', 'Revenue (Millions)', 'Metascore'], inplace=True)
```

Prepare Features

Name : Shejal Tiwari Div/Roll No:D15C/57

```
[3] # One-hot encode the 'Genre' column
    genre_dummies = df['Genre'].str.get_dummies(sep=',')

    # Select numerical features to use
    numerical_features = df[['Runtime (Minutes)', 'Rating', 'Votes', 'Revenue (Millions)', 'Metascore']]

    # Combine genre and numerical features
    features = pd.concat([genre_dummies, numerical_features], axis=1)
```

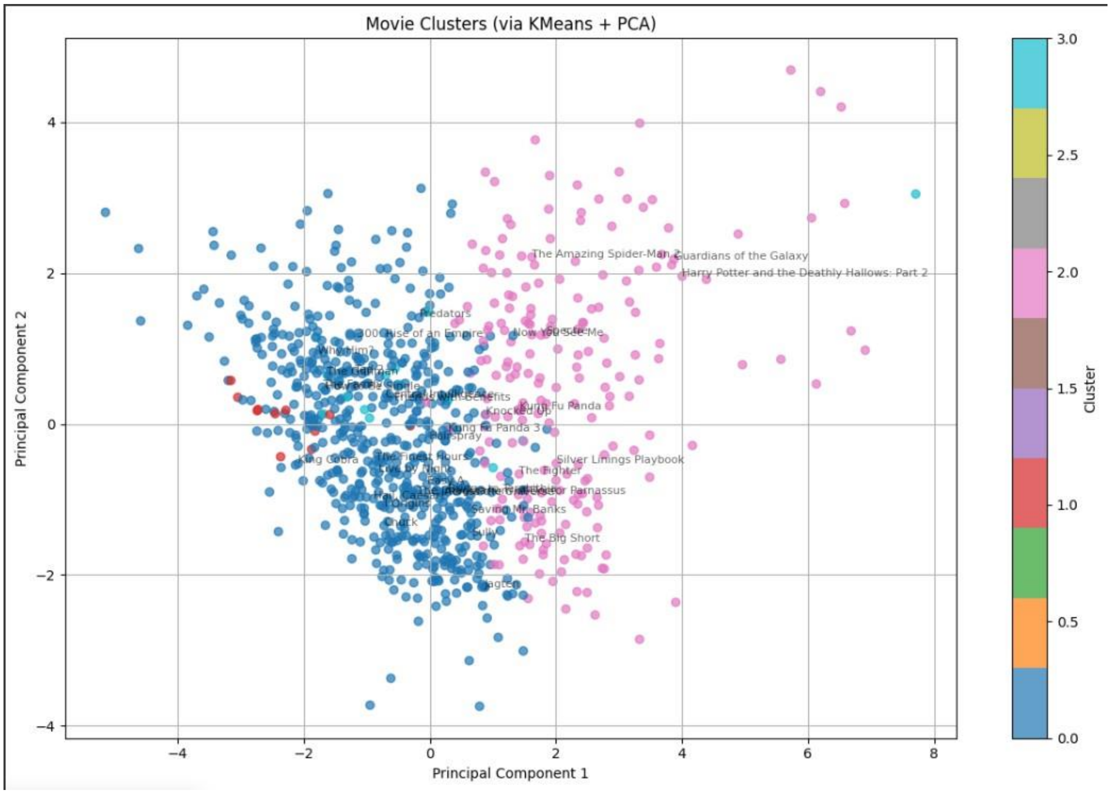
Scale Features

```
[4] scaler = StandardScaler()
    scaled_features = scaler.fit_transform(features)
```

```
kmeans = KMeans(n_clusters=4, random_state=42)
df['Cluster'] = kmeans.fit_predict(scaled_features)
```

Visualize

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

pca = PCA(n_components=2)
pca_result = pca.fit_transform(scaled_features)

plt.figure(figsize=(12, 8))
scatter = plt.scatter(pca_result[:, 0], pca_result[:, 1],
                      c=df_cleaned['Cluster'], cmap='tab10', alpha=0.7)

# Label a few titles
for i in range(0, len(df_cleaned), 25):
    plt.text(pca_result[i, 0], pca_result[i, 1], df_cleaned['Title'].iloc[i],
             fontsize=8, alpha=0.6)

plt.title("Movie Clusters (via KMeans + PCA)")
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.colorbar(scatter, label='Cluster')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Recommendation

```
[7]  def recommend_from_cluster(title, top_n=5):
         matches = df[df['Title'].str.contains(title, case=False, na=False)]
         if matches.empty:
             return f"No match found for '{title}'"

         movie = matches.iloc[0]
         cluster_label = movie['Cluster']

         # Get other movies in same cluster (excluding the one searched)
         same_cluster = df[(df['Cluster'] == cluster_label) & (df['Title'] != movie['Title'])]

         # Recommend top N random movies from that cluster
         recommendations = same_cluster.sample(n=min(top_n, len(same_cluster)), random_state=42)

         print(f"\nRecommendations from Cluster {cluster_label} (same as '{movie['Title']}'):")
         return recommendations[['Title', 'Genre', 'Rating', 'Revenue (Millions)']]
```

```
[8]  recommend_from_cluster("Inception")
```

```
Recommendations from Cluster 2 (same as 'Inception'):
```

	Title	Genre	Rating	Revenue (Millions)
919	The Golden Compass	Adventure,Family,Fantasy	6.1	70.08
203	Iron Man	Action,Adventure,Sci-Fi	7.9	318.30
29	Assassin's Creed	Action,Adventure,Drama	5.9	54.65
735	Hugo	Adventure,Drama,Family	7.5	73.82
37	Doctor Strange	Action,Adventure,Fantasy	7.6	232.60

2.Dimension Reduction

Importing Libraries

```
[15] from sklearn.preprocessing import StandardScaler

     scaler = StandardScaler()
     scaled_features = scaler.fit_transform(features)  # 'features' = genre + numerical columns
```

```
[16] from sklearn.decomposition import PCA

     # Reduce to 5 dimensions for similarity space
     pca = PCA(n_components=5)
     pca_features = pca.fit_transform(scaled_features)
```

Cosine Similarity

```
from sklearn.metrics.pairwise import cosine_similarity

cos_sim_matrix = cosine_similarity(pca_features)
```

Recommendation

Name : Shejal Tiwari Div/Roll No:D15C/57

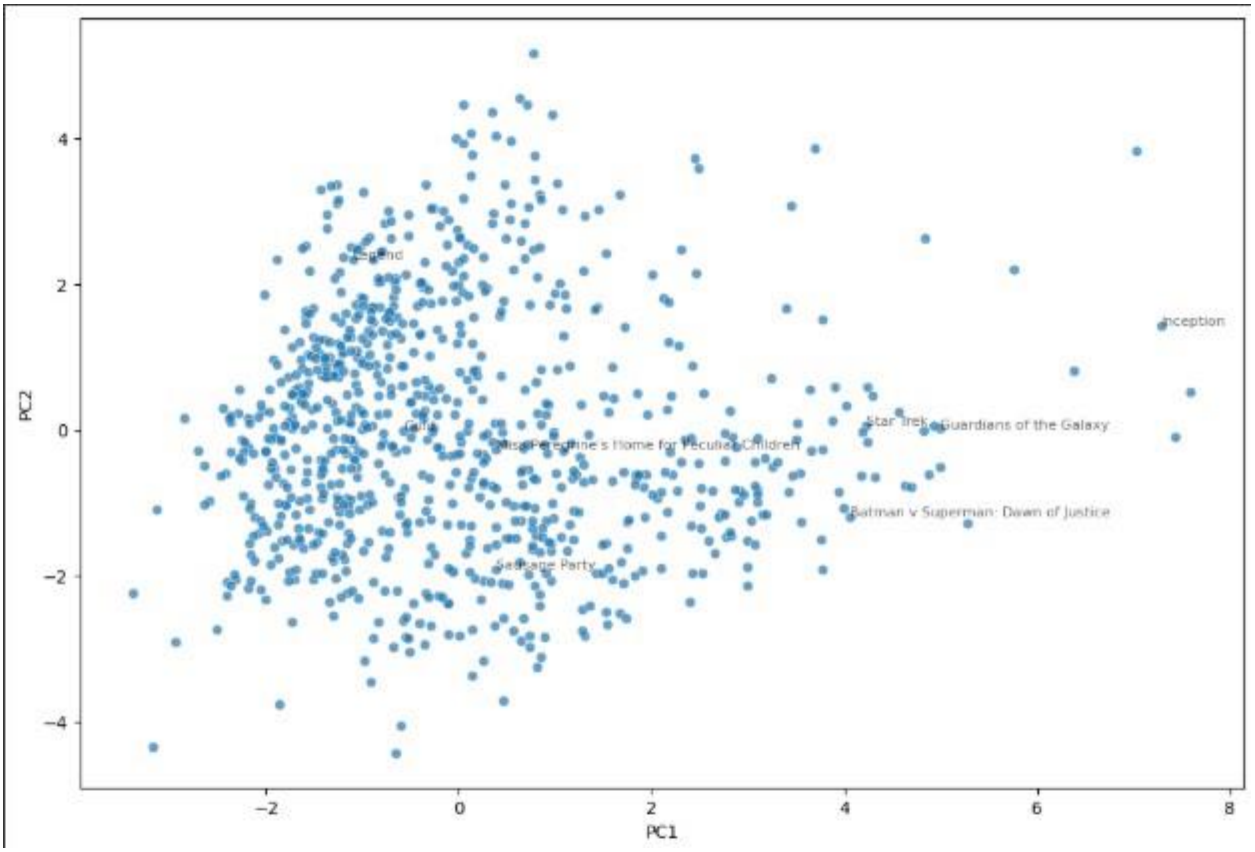
```
def recommend_pca(title, top_n=5):
    matches = df[df['Title'].str.contains(title, case=False, na=False)]
    if matches.empty:
        return f"No movie found for '{title}'"

    idx = matches.index[0]
    sim_scores = list(enumerate(cos_sim_matrix[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)[1:top_n+1]
    top_indices = [i[0] for i in sim_scores]

    print(f"\nTop {top_n} similar movies to '{df.loc[idx, 'Title']}':")
    return df[['Title', 'Genre', 'Rating', 'Revenue (Millions)']].iloc[top_indices]
```

```
[20] recommend_pca("The Avengers")
```

```
Top 5 similar movies to 'The Avengers':
```

	Title	Genre	Rating	Revenue (Millions)
50	Star Wars: Episode VII - The Force Awakens	Action,Adventure,Fantasy	8.1	936.63
271	The Hobbit: An Unexpected Journey	Adventure,Fantasy	7.9	303.00
518	The Hobbit: The Desolation of Smaug	Adventure,Fantasy	7.9	258.36
368	The Amazing Spider-Man	Action,Adventure	7.0	262.03
78	Pirates of the Caribbean: Dead Man's Chest	Action,Adventure,Fantasy	7.3	423.03

Visualization



Conclusion-

In this experiment, we built a recommendation system using various machine learning techniques.Regression and classification helped in predicting and categorizing user preferences.Clustering grouped similar users/items for better suggestions. Dimensionality reduction improved efficiency, while ensemble methods boosted accuracy. Overall, the combined approach enhanced recommendation quality and system performance.