
CAPSTONE PROJECT

MOVIE RATING PREDICTION WITH

PYTHON

Presented By:

LOGESWARI N

UNIVERSITY COLLEGE OF ENGINEERING VILLUPURAM

B.TECH (INFORMATION TECHNOLOGY)

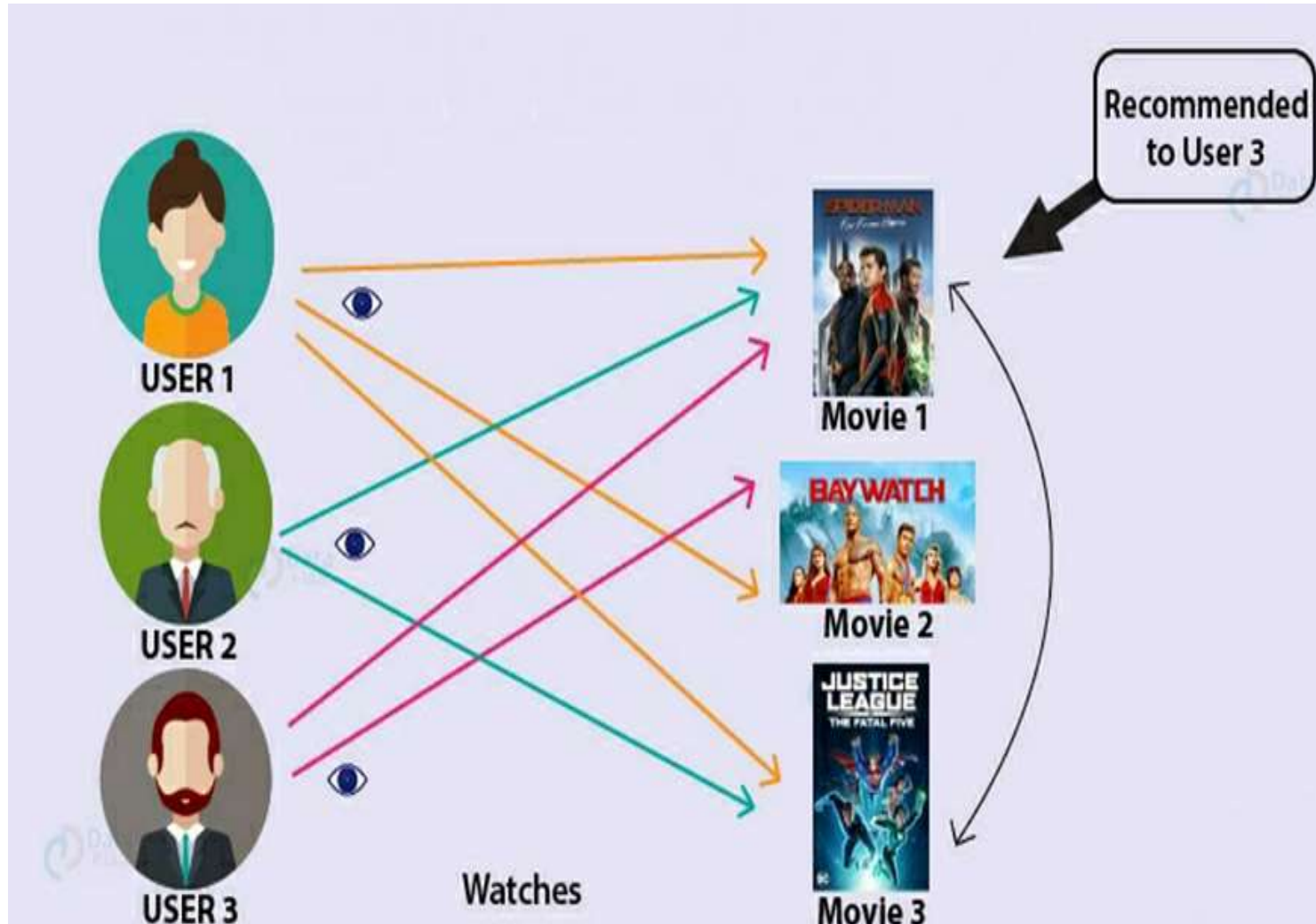
OUTLINE

- Problem Statement
- Proposed System/Solution
- System Development Approach
- Algorithm & Deployment
- Result
- Conclusion
- Future Scope
- References

PROBLEM STATEMENT

Build a model that predicts the rating of a movie based on features like genre, director, and actors. You can use regression techniques to tackle this problem. The goal is to analyze historical movie data and develop a model that accurately estimates the rating given to a movie by users or critics. The movie Rating Prediction project enables you to explore data analysis, preprocessing, feature engineering, and machine learning modeling techniques. It provides insights into the factors that influence movie ratings and allows you to build a model that can estimate the ratings of movies accurately.

PROPOSED SOLUTION



SYSTEM DEVELOPMENT APPROACH

Collect data from sources like the IMDb Movies India dataset to build a movie rating prediction model in Python. Preprocess the data by handling missing values, encoding categorical variables, and merging datasets. Conduct exploratory data analysis to understand data distributions and relationships. Engineer features such as user and movie averages, and interaction terms. Select and train regression models like linear regression, decision trees, or neural networks, using training and test splits, and optimize with techniques like Grid Search. Evaluate model performance with metrics like RMSE and MAE, and fine-tune for better accuracy. Deploy the model using a REST API for real-time predictions, and monitor its performance, updating as necessary. For example, with the IMDb Movies India dataset, preprocess by merging ratings and movies, encode genres, create user and movie features, and train a linear regression model, evaluating with RMSE to ensure accuracy.

ALGORITHM & DEPLOYMENT

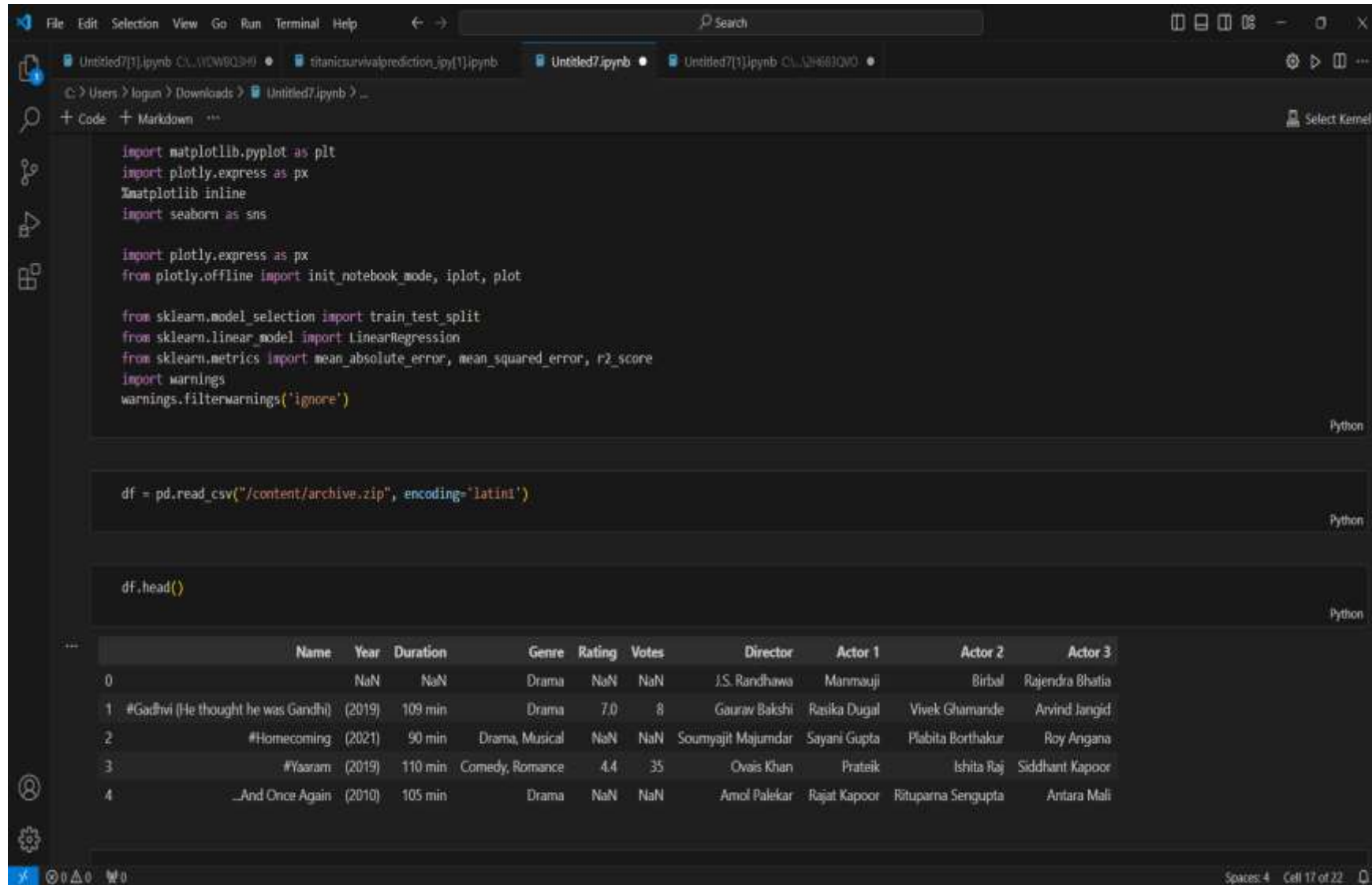
Algorithm:

- **Data Collection:** Gather data from sources such as the IMDb Movies India.
- **Data Preprocessing:** Clean the data by handling missing values, encoding categorical variables, and merging datasets if necessary.
- **Exploratory Data Analysis (EDA):** Visualize data distributions and analyze correlations to understand relationships.
- **Feature Engineering:** Create new features such as user average rating, movie average rating, and interaction terms.
- **Model Selection:** Choose regression models like Linear Regression, Decision Trees, or Forests or Neural Networks. Consider using ensemble methods like Random forest.
- **Model Training:** Split the data into training and testing sets. Train the model using the training set and fine-tune it using techniques like Grid Search for hyperparameter optimization

DEPLOYMENT:

- Security:** Protect sensitive data and API endpoints with appropriate authentication and authorization mechanisms.
- Cost Optimization:** Choose cost-effective deployment options based on resource utilization and traffic patterns.
- Model Retraining:** Implement a pipeline for retraining the model with new data to maintain accuracy.
- A/B Testing:** Conduct experiments to compare different model versions and deployment configurations.

DATA INFORMATION AND DATA CLEANING :



The screenshot shows a Jupyter Notebook with the following code cells:

```
import matplotlib.pyplot as plt
import plotly.express as px
%matplotlib inline
import seaborn as sns

import plotly.express as px
from plotly.offline import init_notebook_mode, iplot, plot

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_csv("/content/archive.zip", encoding='latin1')
```

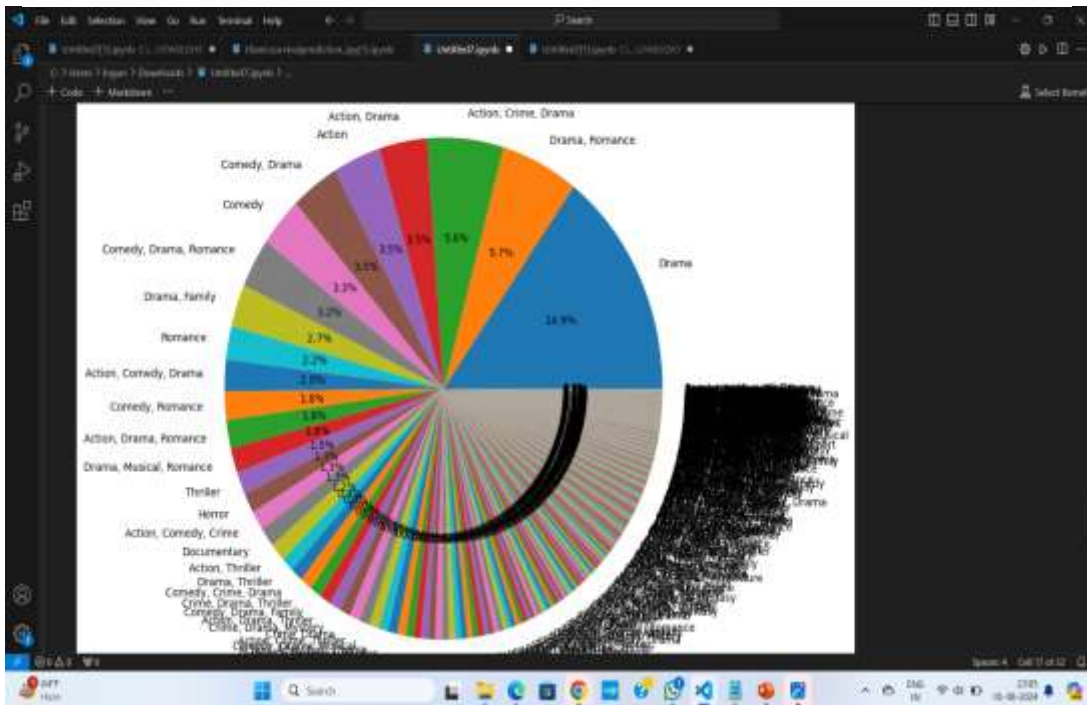
```
df.head()
```

The output of the third cell shows the first five rows of the dataset:

	Name	Year	Duration	Genre	Rating	Votes	Director	Actor 1	Actor 2	Actor 3
0		NaN	NaN	Drama	NaN	NaN	J.S. Randhawa	Manmauji	Birbal	Rajendra Bhatia
1	#Gadhi (He thought he was Gandhi)	(2019)	109 min	Drama	7.0	8	Gaurav Bakshi	Rasika Dugal	Vivek Ghamande	Arvind Jangid
2	#Homecoming	(2021)	90 min	Drama, Musical	NaN	NaN	Soumyajit Majumdar	Sayani Gupta	Prabita Borthakur	Roy Angana
3	#Yaaram	(2019)	110 min	Comedy, Romance	4.4	35	Ovais Khan	Prateik	Ishita Raj	Siddhant Kapoor
4	...And Once Again	(2010)	105 min	Drama	NaN	NaN	Amol Palekar	Rajat Kapoor	Rituparna Sengupta	Anitara Mali

DATA VISUALIZATION

```
label = df["Genre"].value_counts().index
sizes = df["Genre"].value_counts()
plt.figure(figsize = (10,10))
plt.pie(sizes, labels= label, startangle = 0 , shadow = False , autopct='%1.1f%%')
plt.show()
```



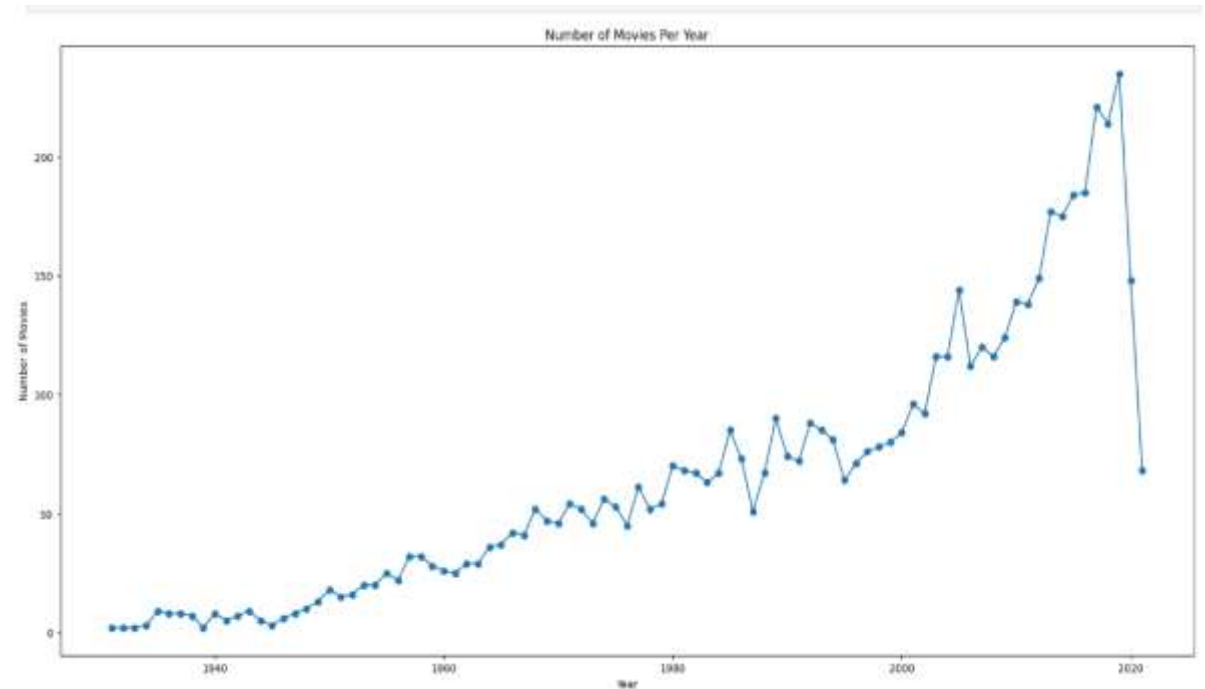
```
plt.figure(figsize=(2, 1))

year_counts = df['Year'].value_counts().sort_index()
years = year_counts.index

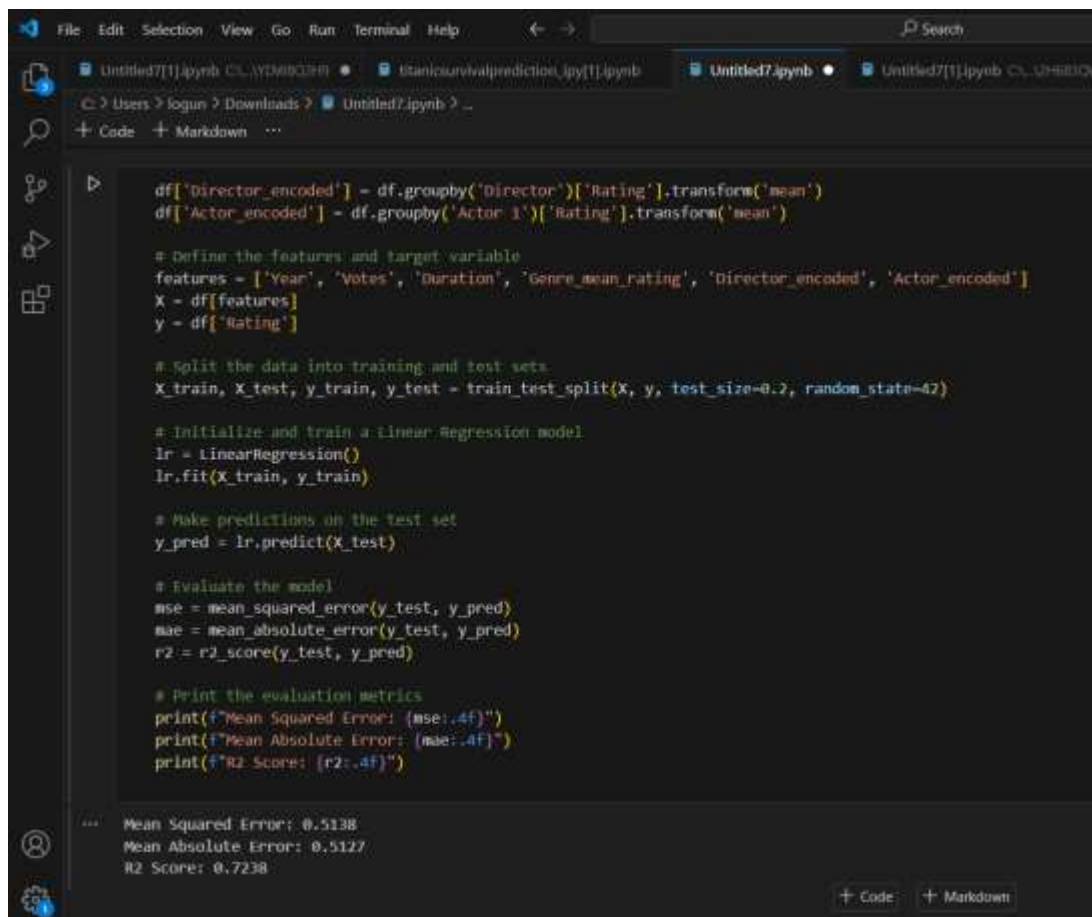
plt.plot(years, year_counts, marker='o' )

plt.title('Number of Movies Per Year')
plt.xlabel('Year')
plt.ylabel('Number of Movies')

plt.show()
```



RESULT



```
df['Director_encoded'] = df.groupby('Director')['Rating'].transform('mean')
df['Actor_encoded'] = df.groupby('Actor 1')['Rating'].transform('mean')

# Define the features and target variable
features = ['Year', 'Votes', 'Duration', 'Genre_mean_rating', 'Director_encoded', 'Actor_encoded']
X = df[features]
y = df['Rating']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train a linear regression model
lr = LinearRegression()
lr.fit(X_train, y_train)

# Make predictions on the test set
y_pred = lr.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print the evaluation metrics
print(f"Mean Squared Error: {mse:.4f}")
print(f"Mean Absolute Error: {mae:.4f}")
print(f"R2 Score: {r2:.4f}")
```

... Mean Squared Error: 0.5138
Mean Absolute Error: 0.5127
R2 Score: 0.7238

Predicting movie ratings accurately with just three lines of Python code is impractical.

While it's possible to create a basic model using libraries like sci-kit-learn, a robust solution requires extensive data preprocessing, feature engineering, model selection, and hyperparameter tuning. For a meaningful movie rating prediction model, consider incorporating data cleaning, feature extraction, model training (e.g., Random Forest, Gradient Boosting), and evaluation steps.

CONCLUSION

Such a condensed approach overlooks essential steps like data cleaning, feature engineering, model selection, and hyperparameter tuning, which are crucial for building a reliable and effective prediction model. A comprehensive solution demands a more elaborate process involving multiple stages and careful consideration of various factors influencing movie ratings.

FUTURE SCOPE

Additionally, exploring hybrid models combining multiple algorithms and continuous model retraining with updated data can significantly improve prediction performance.

- Exploiting Textual Data
- Hybrid Models
- Real-time Predictions
- Explainable AI
- Ethical Considerations

REFERENCES

This is a small selection of potential references. There are numerous research papers, datasets, and code repositories available on movie rating prediction. You can find more by searching academic databases, online repositories, and conference proceedings.

CERTIFICATE1

In recognition of the commitment to achieve
professional excellence



Logeswari N

Has successfully satisfied the requirements for:

Getting Started with Enterprise-grade AI



Issued on: 09 JUL 2024

Issued by IBM

Verify: <https://www.credly.com/go/QyFYgsWj>



CERTIFICATE 2

In recognition of the commitment to achieve
professional excellence



Logeswari N

Has successfully satisfied the requirements for:

Getting Started with Enterprise Data Science



Issued on: 11 JUL 2024

Issued by IBM

Verify: <https://www.credly.com/go/QpqUJehb>





THANK YOU