Predicting IMDb scores

Name: Kiran Ram N

NM ID:au723721205024

Phase 3 :

IMDB dataset from various sources, like the official IMDB website or data repositories and kaagle. We might also use libraries like pandas or numpy to load the data from a CSV or other structured file.

Data import :

```
Data = pd.read_csv(“/kaggle/input/netflix-original-films-imdb-scores/NetflixOriginals.csv”,encoding = “ISO-8859-1”)

dataDate = data.copy()

data.head()
```

S. No	Title	Genre	Premire	Runtime	IMDb score	Language
1	Enter the anime	Documentary	August 5, 2019	58	**2.5**	English/japanese

2	Dark forces	Thriller	August 21,2020	81	**2.6**	Spanish
3	The app	Science/drama	December 26/2019	79	**2.6**	Italian
4	The open house	Horror Thriller	January19 ,2018	94	**3.8**	English
5	Kaali khuuhi	Mystery	October 30,2020	90	**3.4**	Hindi

Data Preprocessing:

Data Cleaning:

Remove any duplicate entries or irrelevant columns.

Handle Missing Data:

Check for and handle any missing values in the dataset.

Text Processing:

If our dataset contains textual data like reviews, you may need to preprocess and tokenize the text.

Label Encoding:

categorical labels (if any) into numerical format.

Train-Test Split:

Split the dataset into training and testing subsets. This helps assess the model's performance.

Normalization/Scaling:

If the datas are numerical data, it's often a good idea to normalize or scale the features for better model performance.

Model-Specific Preprocessing:

Some machine learning models require specific preprocessing steps. For example, recurrent neural networks (RNNs) for text data might require padding or truncating sequences

Program :

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

from sklearn.feature_extraction.text import CountVectorizer


# Load the dataset
```

```
data = pd.read_csv("imdb_dataset.csv")
# Data preprocessing
# - Drop duplicates and handle missing values
data.drop_duplicates(inplace=True)
data.dropna(inplace=True)
# For text data (e.g., movie reviews), you can use CountVectorizer or other text preprocessing techniques.

# Encode labels to numerical values
label_encoder = LabelEncoder()
data['sentiment'] = label_encoder.fit_transform(data['sentiment'])
# Train-test split
X = data['review']
y = data['sentiment']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Different analysis:

Regression Analysis:

Datas can treat the IMDB scores as continuous values and perform regression analysis to predict scores. Linear regression, decision trees, random forests, or gradient boosting algorithms are commonly used.

Classification Analysis:

Convert IMDB scores into categories (e.g., low, medium, high) and use classification algorithms like logistic regression, SVM, or deep learning models to predict the class.

Deep Learning Models:

Utilize deep neural networks, such as recurrent neural networks (RNNs) for text data or convolutional neural networks (CNNs) for image data, to predict IMDB scores.

DAC:

In this step we can perform the different analysis and Visualisation the datas using IMB cagnos.

Data Preparation:

Make sure the dataset with IMDb scores and other relevant data. This can be a CSV, Excel, or database file.

IBM Cognos Installation:

We have ensure that IBM Cognos installed and set up on our system.

Create a New Report:

Open IBM Cognos and create a new report or dashboard.

Data Connection:

Connect to the dataset within IBM Cognos. Import the data want to use for IMDb score predictions.

Data Modeling:

Create a data model if necessary, which may involve defining relationships between different data tables.

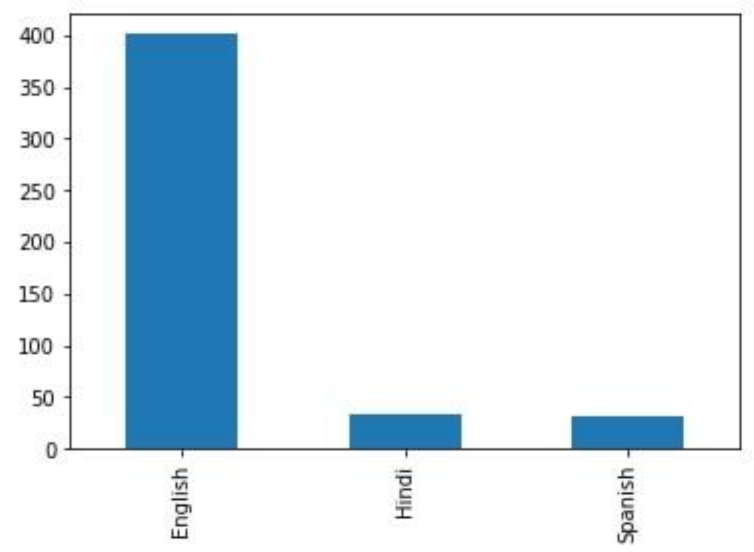
Visualization Creation:

In Cognas we can create various types of visualizations, like bar charts, line charts, or scatter plots. Choose the type of visualization that best represents IMDb score prediction.

Find the 3 most used languages in the movies in the data set.

Bar graph

```
df_lang = df['language'].value_counts()
df_lang.head(3).plot(kind='bar')
plt.show(block=True)
```

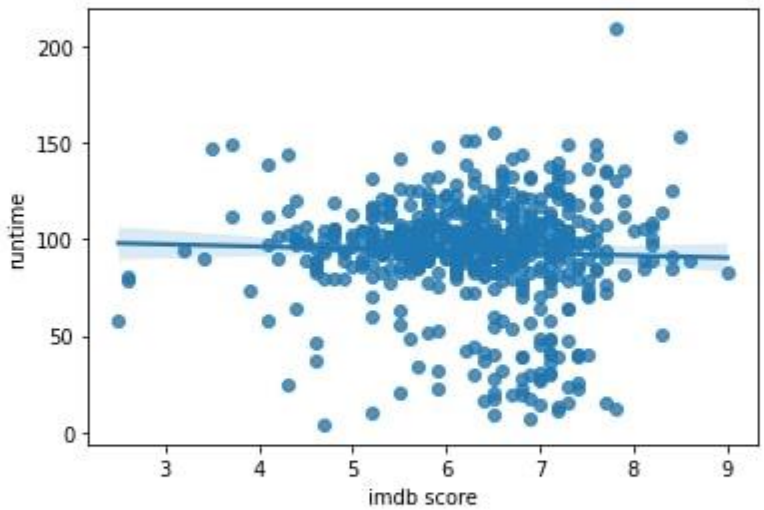


What is the correlation between IMDB score and ‘Runtime’? Examine and visualize.

```
 sns.regplot(x=’imdb score’, y=’runtime’, data=df)
 pot-shot(block=True)
x = round(df[‘imdb score’].corr(df[‘runtime’]), 3)
print(f’The correlation between runtime and imdb score is {x}.’)
```

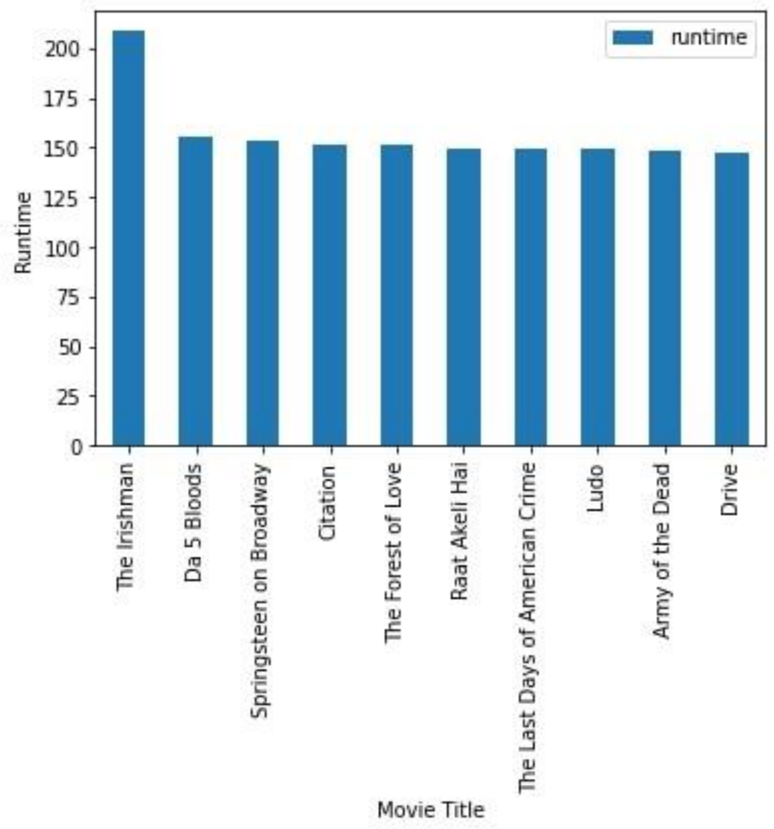
Scattered plot

```
df[['title', 'runtime']].sort_values('runtime', ascending=False).head(10).
plot(x='title', y='runtime', kind='bar')
plt.xlabel('Movie Title')
plt.ylabel('Runtime')
plt.show(block=True) df[['title', 'runtime']].sort_values('runtime', ascen
ding=False).head(10).plot(x='title', y='runtime', kind='bar')
plt.xlabel('Movie Title')
plt.ylabel('Runtime')
plt.show(block=True)
```



What are the top 10 movies with the highest ‘runtime’? Visualize it.

```
Df[[‘title’, ‘runtime’]].sort_values(‘runtime’, ascending=False).head(10).
plot(x=’title’, y=’runtime’, kind=’bar’)
Plt.xlabel(‘Movie Title’)
Plt.ylabel(‘Runtime’)
Plt.show(block=True)
```



Data Analysis:

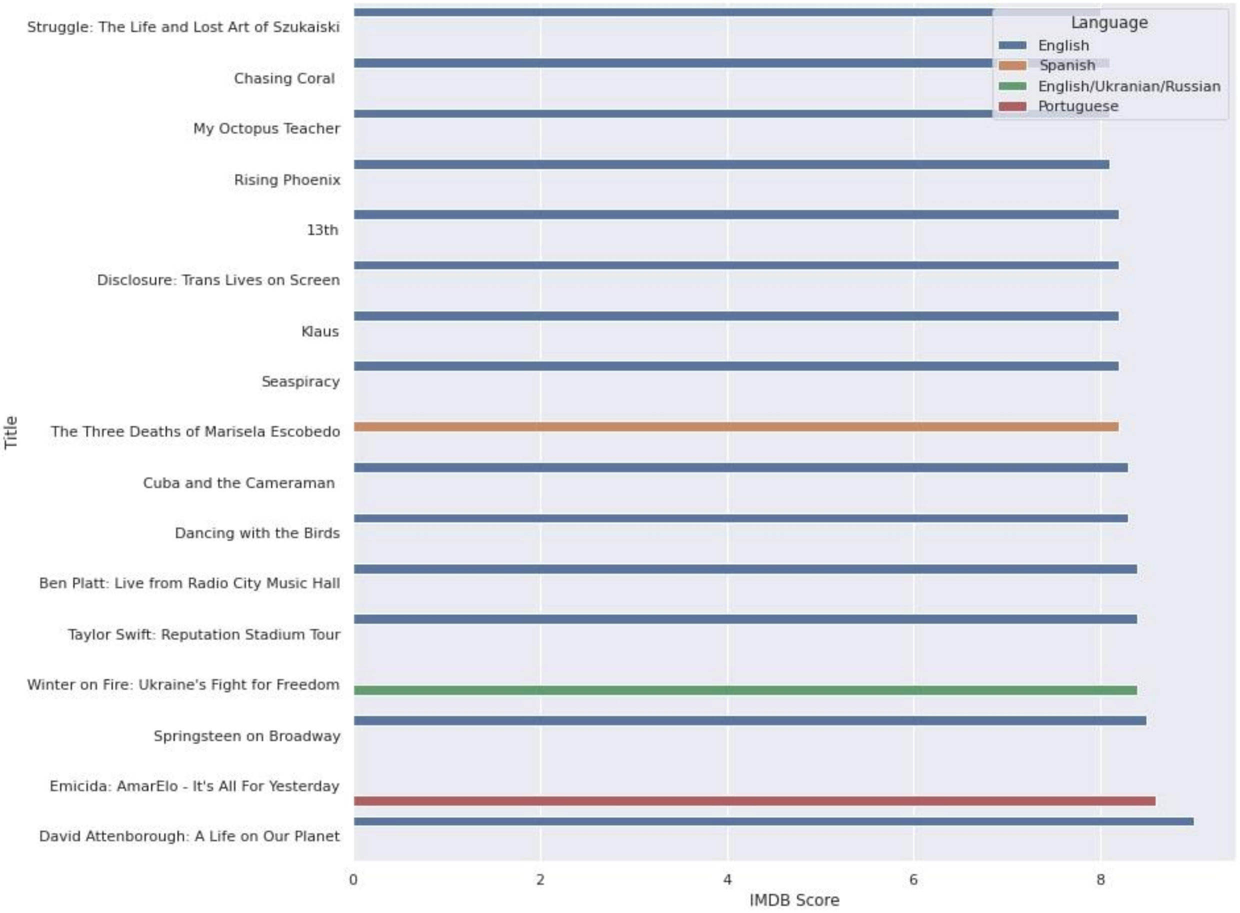
Add your IMDb score prediction data to the visualization, along with any other relevant data we want to display.

Customization & Interactivity :

Customize the visualization by adding labels , titles, and adjusting the colors to make it interactive.

Add interactive features like filters and drill through analysis for deeper analysis

```
Above are the genres with languages and IMDB Score with rating higher than
 7
  Plt.figure(figsize = (12,12))
Sns.barplot(x = ‘IMDB Score’, y = ‘Title’,hue =’Language’, data = score_8)
```



Preview and Publish:

Peview the visualization within IBM Cognos to ensure it looks as expected. Then, we can publish it for others to access.

IOT : In this step we will use the IOT devices and deploying python script for IOT devices

Prepare the Python Script:

- Make sure the python script is optimized for performance, as IoT devices typically have limited resources.
- Use libraries and frameworks that are lightweight and compatible with IoT platforms.
- Test the IMDb score prediction model on the development machine before deployment.

Install Python on IoT Device:

- ❖ Some IoT devices come with Python pre-installed, while others may require manual installation.We have to Ensure that the Python version is compatible with the script.

Transfer The Script:

- ❖ Transfer the Python script to the IoT device using methods like SSH, FTP, or through a development environment provided by the IoT platform.

Manage Dependencies:

- ❖ Ensure that any required libraries and dependencies are installed on the IoT device. Use lightweight libraries when possible.

Run the Script:

- ❖ Execute the Python script on the IoT device. May use terminal commands or scripts for this purpose.

Data Input and Output:

- Define how data will be input to the script and how the predictions will be output. IoT devices might use sensors, external data sources, or APIs to gather data.

Real-time Predictions:

- Consider how often the IMDb score prediction script should run. Set up a schedule or event-triggering mechanism as per your application's requirements.

Monitoring and Maintenance:

- Implement monitoring and error-handling mechanisms to ensure the script runs smoothly on the IoT device.
- Regularly update and maintain the script to accommodate changes or improvements.

Testing and Validation:

- Test the IMDb score prediction on the IoT device thoroughly to ensure accuracy and reliability.

Security :

- Finally we also ensure that the security of IOT devices which we are used for this process to ensure the safety of the process and python script.

CAD:

Data Collection:

Gather IMDb movie data, including movie details and historical IMDb scores. We can use web scraping tools, public datasets, or APIs to obtain this data.

Data Storage:

Store the collected data in a database. IBM Cloud offers various database services, such as IBM Db2, PostgreSQL, or cloud-native databases like IBM Cloudant.

IBM Cloud Foundry:

Create an application on IBM cloud foundry to use popular framework like flask django to developing application in python

Data Ingestion:

Build data ingestion mechanisms to import the movie data into the application's database.

Machine Learning Model:

Develop a machine learning model in Python to predict IMDb scores. Use libraries like scikit-learn or TensorFlow for this. Train the model using historical IMDb scores as your target variable.

API Endpoint:

Endpoin our machine learning model as an API endpoint using your IBM Cloud Foundry application. Then we can use web frameworks to create a REST API.

Different type of Functions:

Prediction Function:

Implement an API route that accepts movie information as input and returns a predicted IMDb score.

Data Update Function:

Create a function to update the model with new data periodically to improve prediction accuracy.

Authentication and Authorization: Implement security mechanisms to the API Functions or API endpoints.

User Interface:

Develop a user interface where users can input movie details and get IMDb score predictions. This interface can be a web application or a mobile app.

Integration with Cloud Services:

Utilise other IBM Cloud services like IBM Watson for natural language processing (NLP) to analyze user reviews, which can be used as additional features for prediction.

Monitoring and Logging:

Implement monitoring tools and logging to track the performance and usage of the application.

Testing and Validation:

Thoroughly test the application and model to ensure accurate predictions. Use techniques like cross-validation and A/B testing to evaluate your model's performance.

Deployment: Deploy the application to IBM Cloud Foundry and make it accessible to users.