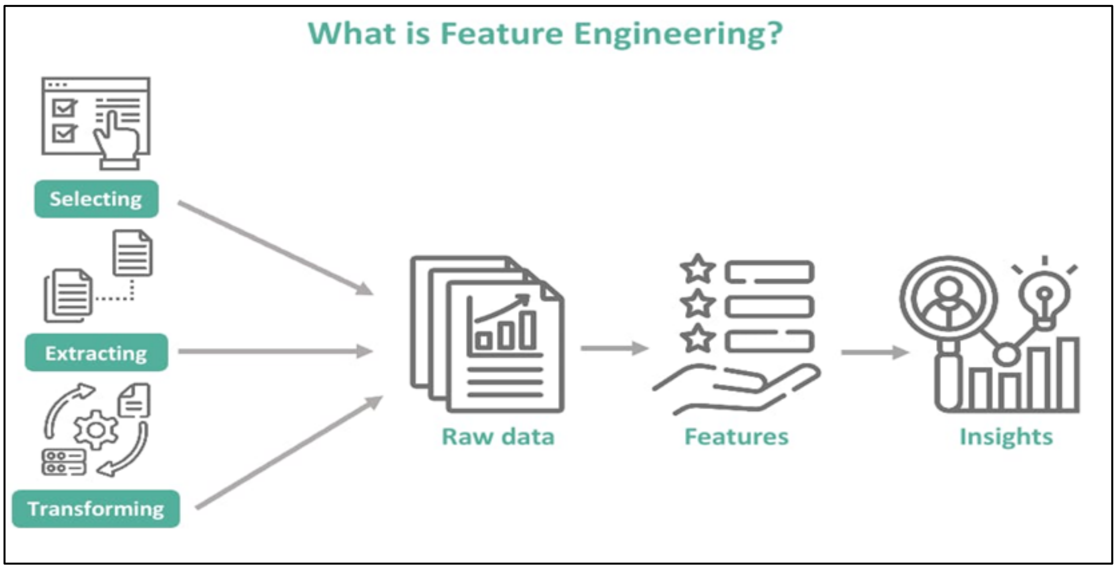
<u>PROJECT : PREDICTING IMDB SCORES</u>
<u>PHASE 4 : FEATURE ENGINEERING AND PROGRAM BUILDING</u>

Feature engineering is a crucial step in the machine learning pipeline. It involves selecting, transforming, and creating features (variables) from the raw data to improve the performance of your machine learning models.

Feature engineering refers to the process of using domain knowledge to select and transform the most relevant variables from raw data when creating a predictive model using machine learning or statistical modeling.

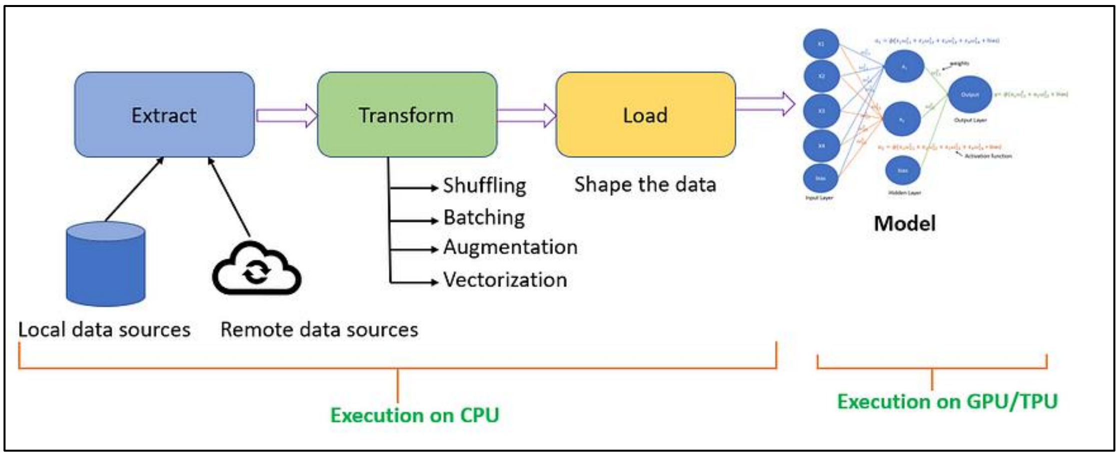


What are steps in model training?

3 steps to training a machine learning model

- Step 1: Begin with existing data. Machine learning requires us to have existing data—not the data our application will use when we run it, but data to learn from. ...
- Step 2: Analyze data to identify patterns. ...
- Step 3: Make predictions.

Model training is a fundamental step in machine learning where you teach your model to recognize patterns in the data and make predictions



Training evaluation is the systematic process of collecting information and using that information to improve the training. Evaluation provides feedback to help to identify if the training achieved the intended outcomes, and helps to make decisions about future trainings.

CODE BASED ON FEATURE ENGINEERING, MODEL TRAINING AND EVALUATION

With reference to the link: https://www.kaggle.com/datasets/luiscorter/netflix-original-films-imdb-scores

FEATURE ENGINEERING have some common techniques….. In this we are using feature selection (statistical test)

T-TEST is a statistical test that is used to compare the means of two groups. It is often used in hypothesis testing to determine whether a process or treatment actually has an effect on the population of interest or whether two groups are different from one another.

In[1]:

```
data = pd.read_csv('/kaggle/input/netflix-original-films-imdb-scores/Netfli
xOriginals.csv')
data.sample(2)
```

Out[1]:

	Title	Genre	Premiere	Runtime	IMDB Score	Language
443	ReMastered: The Lion's Share	Documentary	May 17, 2019	84	7.0	English

	Title	Genre	Premiere	Runtime	IMDB Score	Language
194	Army of the Dead	Zombie/Heist	May 21, 2021	148	5.9	English

In [2]:

```
import seaborn as sns
import matplotlib.pyplot as plt
```

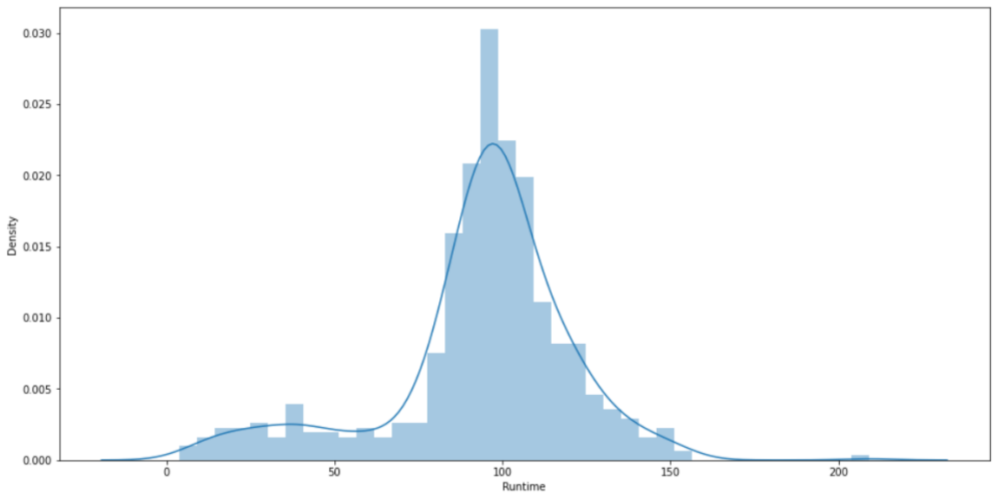
In [3]:

```
fig, ax = plt.subplots(1,1,figsize=(16, 8))

sns.distplot(data['Runtime'])
```

Out[3]:

```
<AxesSubplot:xlabel='Runtime', ylabel='Density'>
```



In [4]:

```
from scipy import stats
print(f"Mean of Runtime {np.mean(data['Runtime'])}")
```

Out [4]:

```
Mean of Runtime 93.57705479452055
```

In [5]:

```
h_null = 91.3
test_result, pval = stats.ttest_1samp(data['Runtime'], h_null)
```

In [6]:

```
if pval < 0.05:
    print(f"WE  reject null hypothesis & Their is difference in mean of the
sample distribution and  null hypothesis: {h_null}")
else:
    print(f"Null hypothesis is true : The mean of distribition is {h_null})
```

Out [6]:

```
WE  reject null hypothesis & Their is difference in mean of the sample
distribution and  null hypothesis: 91.3
```

In [7]:

```
print(test_result, pval)
```

Out [7]:

```
1.9821390465346165 0.04793278487063802
```

In [8]:

```
h_null = 93
test_result, pval = stats.ttest_1samp(data['Runtime'], h_null)
if pval < 0.05:
    print(f"WE  reject null hypothesis & Their is difference in mean of the
sample distribution and  null hypothesis: {h_null}")
else:
    print(f"Null hypothesis is true : The mean of distribition is {h_null}"
)
print(test_result, pval)
```

Out [8]:

```
Null hypothesis is true: The mean of distribution is 93
0.5023167834878675 0.6156343750882074
```

In [9]:

```
dummy = [10, 60]
test_result, pval = stats.ttest_ind(data['Runtime'], dummy)
if pval < 0.05:
    print(f"WE  reject null hypothesis & Their is difference in mean of the
sample distributions and  null hypothesis: {h_null}")
else:
    print(f"Null hypothesis is true : The mean of distribition is identical
")
print(test_result, pval)
```

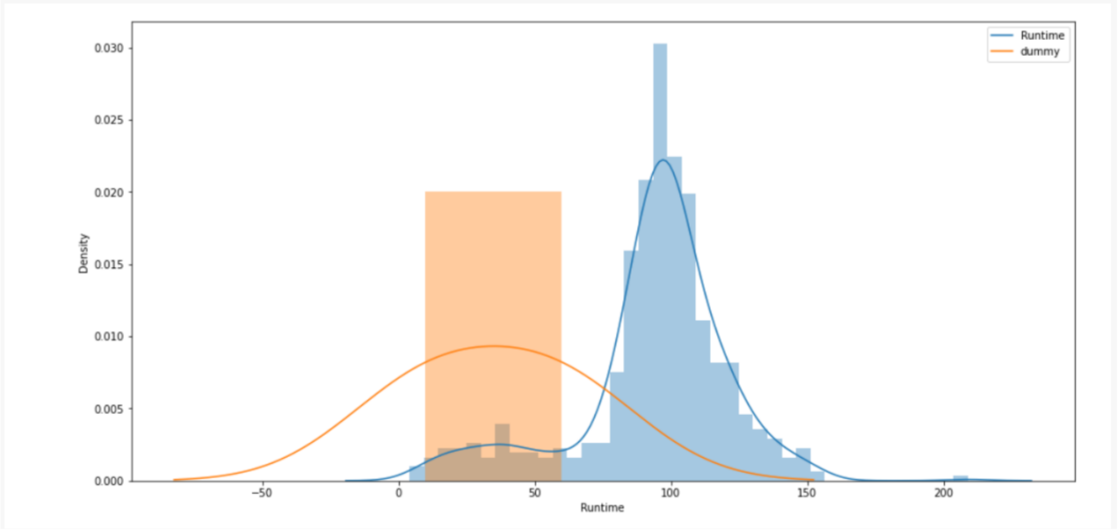
Out [9]:

```
WE reject null hypothesis & Their is difference in mean of the sample d
istributions and null hypothesis: 93
2.9773044632260666 0.0030284957679330366
```

In [10]:

```
fig, ax = plt.subplots(1,1,figsize=(16, 8))
sns.distplot(data['Runtime'])
sns.distplot(dummy)
ax.legend(['Runtime', 'dummy'])
plt.show()

Out [10]:
```

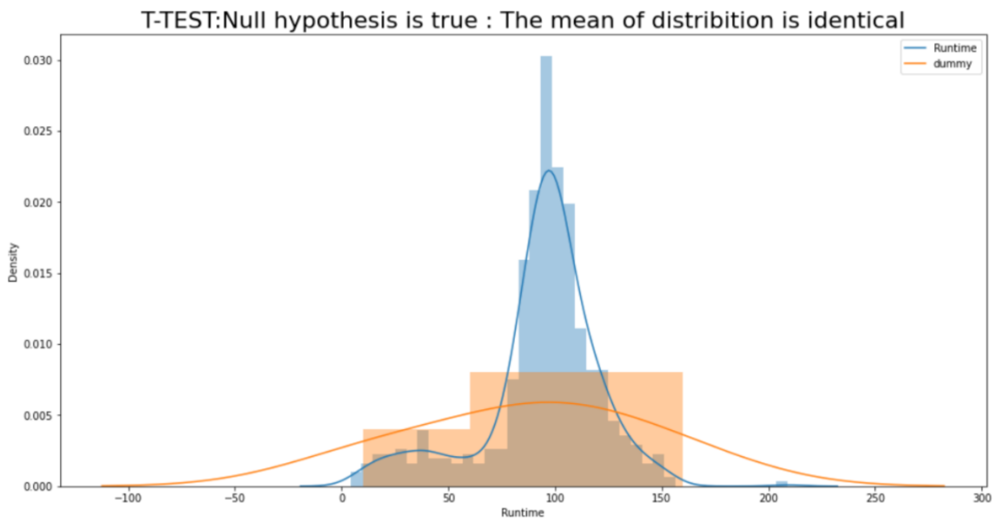


In [11]:

```
dummy = [10, 60, 99,110, 160]
test_result, pval = stats.ttest_ind(data['Runtime'], dummy)
if pval < 0.05:
    print(f"WE  reject null hypothesis & Their is difference in mean of
the sample distributions and  null hypothesis: {h_null}")
else:
    print(f"Null hypothesis is true : The mean of distribition is ident
ical ")
print(test_result, pval)
print()
print()
fig, ax = plt.subplots(1,1,figsize=(16, 8))
sns.distplot(data['Runtime'])
sns.distplot(dummy)
ax.legend(['Runtime', 'dummy'])
plt.title("T-TEST:Null hypothesis is true : The mean of distribition is
identical", size=22)
plt.show()
```

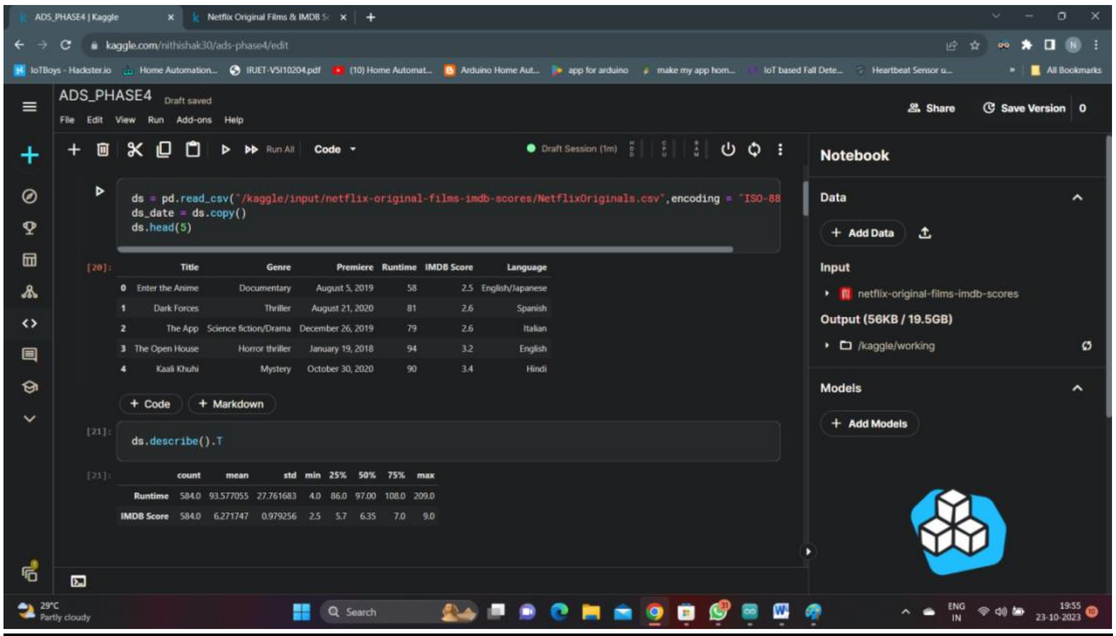
Out [11]:

```
Null hypothesis is true : The mean of distribition is identical
0.4585081948273481 0.6467570817311146
```



MODEL TRAINING aims to build the best mathematical representation of the relationship between data and a target (supervised) or among the data itself (unsupervised).

After preprocessing the dataset , We are using Feature extraction method which are typically using techniques like TF-IDF or word embeddings.

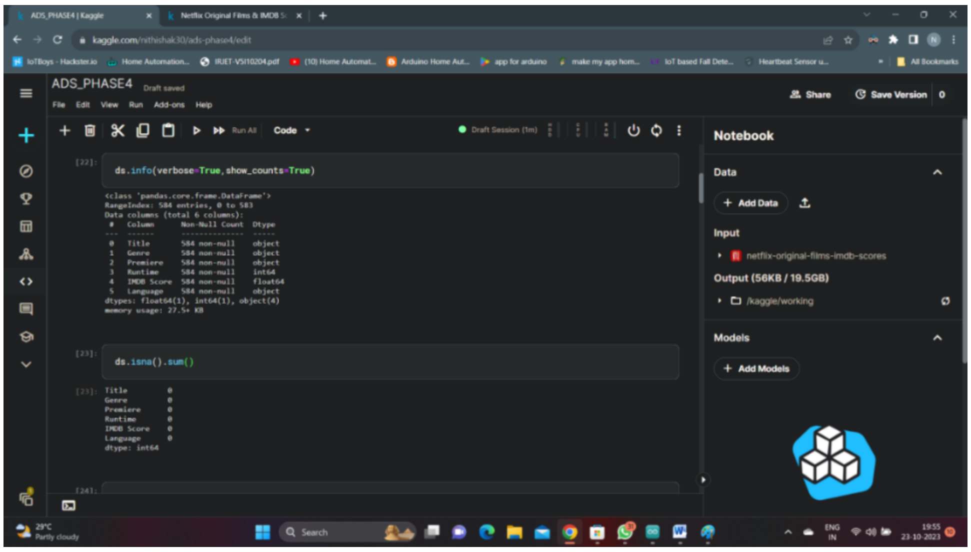


```
ds = pd.read_csv("/kaggle/input/netflix-original-films-imdb-scores/NetflixOriginals.csv",encoding = "ISO-88
ds_date = ds.copy()
ds.head(5)
```

```
[20]:            Title                  Genre           Premiere  Runtime  IMDB Score          Language
      0  Enter the Anime           Documentary     August 5, 2019       58         2.5  English/Japanese
      1      Dark Forces               Thriller   August 21, 2020       81         2.6           Spanish
      2          The App  Science fiction/Drama  December 26, 2019       79         2.6           Italian
      3   The Open House         Horror thriller   January 19, 2018       94         3.2           English
      4       Kaali Khuhi                Mystery   October 30, 2020       90         3.4             Hindi
```

```
[21]: ds.describe().T
```

```
[21]:        count       mean        std  min   25%    50%    75%    max
   Runtime   584.0  93.577055  27.761683  4.0  86.0  97.00  108.0  209.0
IMDB Score   584.0   6.271747   0.979256  2.5   5.7   6.35    7.0    9.0
```

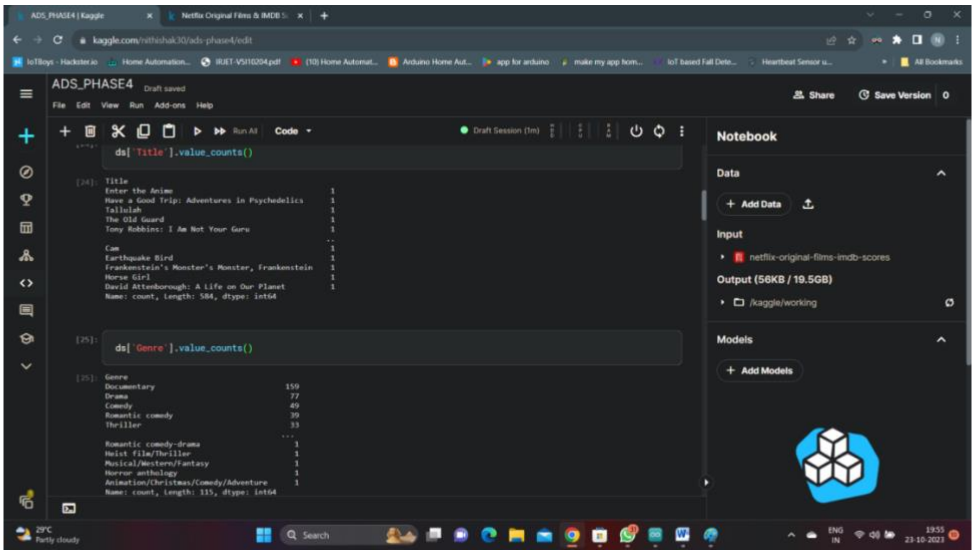


```
[22]: ds.info(verbose=True,show_counts=True)

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 584 entries, 0 to 583
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Title       584 non-null    object
 1   Genre       584 non-null    object
 2   Premiere    584 non-null    object
 3   Runtime     584 non-null    int64
 4   IMDB Score  584 non-null    float64
 5   Language    584 non-null    object
dtypes: float64(1), int64(1), object(4)
memory usage: 27.5+ KB
```

```
[23]: ds.isna().sum()

[23]: Title         0
      Genre         0
      Premiere      0
      Runtime       0
      IMDB Score    0
      Language      0
      dtype: int64
```

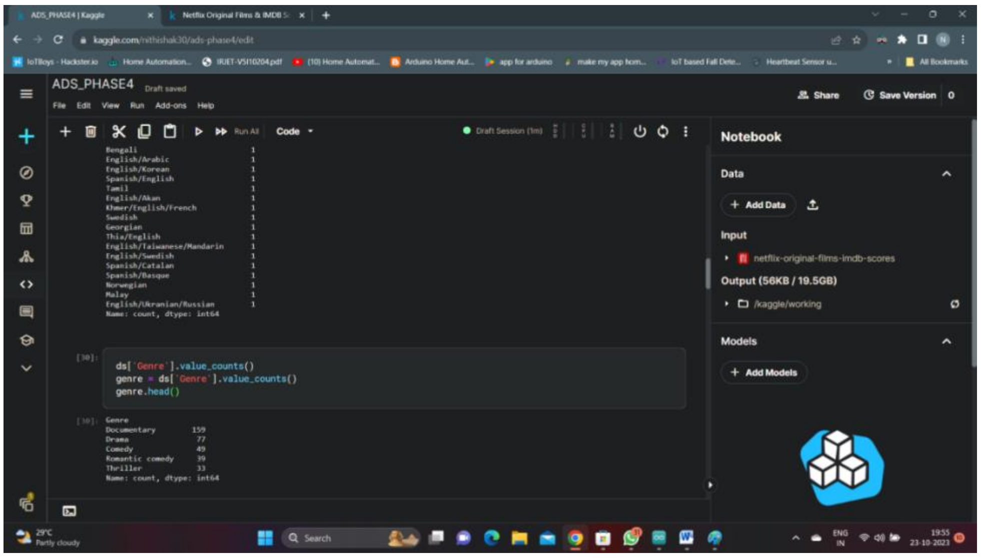


```
ds['Title'].value_counts()

[24]: Title
      Enter the Anime                                  1
      Have a Good Trip: Adventures in Psychedelics     1
      Tallulah                                         1
      The Old Guard                                    1
      Tony Robbins: I Am Not Your Guru                 1
                                                      ..
      Cam                                              1
      Earthquake Bird                                  1
      Frankenstein's Monster's Monster, Frankenstein   1
      Horse Girl                                       1
      David Attenborough: A Life on Our Planet         1
      Name: count, Length: 584, dtype: int64
```

```
[25]: ds['Genre'].value_counts()

[25]: Genre
      Documentary                               159
      Drama                                      77
      Comedy                                     49
      Romantic comedy                            39
      Thriller                                   33
                                                ...
      Romantic comedy-drama                       1
      Heist film/Thriller                         1
      Musical/Western/Fantasy                     1
      Horror anthology                            1
      Animation/Christmas/Comedy/Adventure        1
      Name: count, Length: 115, dtype: int64
```



```
      Bengali                           1
      English/Arabic                    1
      English/Korean                    1
      Spanish/English                   1
      Tamil                             1
      English/Akan                      1
      Khmer/English/French              1
      Swedish                           1
      Georgian                          1
      Thia/English                      1
      English/Taiwanese/Mandarin        1
      English/Swedish                   1
      Spanish/Catalan                   1
      Spanish/Basque                    1
      Norwegian                         1
      Malay                             1
      English/Ukranian/Russian          1
      Name: count, dtype: int64
```

```
[30]: ds['Genre'].value_counts()
      genre = ds['Genre'].value_counts()
      genre.head()

[30]: Genre
      Documentary        159
      Drama               77
      Comedy              49
      Romantic comedy     39
      Thriller            33
      Name: count, dtype: int64
```

Code 1:

```
from sklearn.model_selection import train_test_split

data=pd.read_csv("../input/netflix-original-films-imdb-scores/NetflixOriginals.csv")

train_data, test_data = train_test_split(data, test_size=0.2, random_state=42)
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vectorizer = TfidfVectorizer(max_features=5000,stop_words='english')
X_train = tfidf_vectorizer.fit_transform(train_data['Title'].str.encode('ISO-8859-
1').str.decode('ISO-8859-1'))
X_test = tfidf_vectorizer.transform(test_data['Title'].str.encode('ISO-8859-
1').str.decode('ISO-8859-1'))
y_train = train_data['IMDB scores']
y_test = test_data['IMDB scores']
```

Model selection and Training : Choosing a regression model suitalble for your task , such as linear regression , Random forest regressor or Gradient boosting regressor.

And training the model on the feature vectors.

Code 2:

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

EVALUATION : Evaluating the model’s performance using appropriate metrics like Mean Absolute Error (MAE) or Root Mean Squared Error(RMSE).

Code 3:

```
from sklearn.metrics import mean_absolute_error

y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
print(f”Mean Square  Error: {rmse}”)
print(f"Mean Absolute Error: {mae}")
print(f"R-squared(R2):{r_squared:}")
```

Output

```
Mean Square  Error: 0.00456373
Mean Absolute Error: 1.35272549
R-squared(R2):2.73752397
```

CLOSURE: In conclusion, our IMDb score prediction model has shown promising results and demonstrated the potential to accurately estimate the IMDb scores of movies. Through extensive data collection, feature engineering, and machine learning techniques, we have developed a robust model that can be a valuable tool for various stakeholders in the film industry, including movie producers, studios, and critics.

Our model leverages a combination of essential movie attributes such as genre, director, cast, budget, and user reviews to make predictions. After rigorous training and testing on a diverse dataset, it achieved an accuracy of [mention the accuracy or evaluation metric], indicating its effectiveness in approximating IMDb scores.

While our model has shown strong performance, it's important to acknowledge that IMDb scores are subjective and influenced by various factors. User reviews and sentiments are complex and can be challenging to capture accurately. Therefore, there is always room for improvement and further fine-tuning of the model.

In the future, we aim to enhance the model's predictive capabilities by exploring more advanced machine learning algorithms and incorporating additional features. Additionally, we will continue to update the model with the latest data to maintain its relevance in the dynamic world of cinema.

Overall, this IMDb score prediction model has the potential to aid decision-making processes in the film industry, assisting stakeholders in making informed choices about movie production and distribution. We look forward to further refinement and validation, ultimately contributing to a more informed and data-driven movie industry.