Phase-4

IBM NAN MUDHALVAN

IMDb Score Prediction

1. Feature Engineering:

This step involves converting categorical variables to numerical form using one-hot encoding. In this script, the categorical variables 'type', 'country', 'rating', and 'listed_in' are one-hot encoded. Irrelevant columns and columns with missing values are dropped to ensure data quality.

2. Model Training:

After preparing the data, a Linear Regression model is chosen for training. The standardized features (scaled using the StandardScaler) and target values are used to fit the Linear Regression model.

3. Evaluation:

The model's performance is evaluated using two metrics:

(I)Mean Squared Error (MSE): It calculates the average of the squares of the errors between the predicted IMDb scores and the actual IMDb scores. A lower MSE indicates better performance.

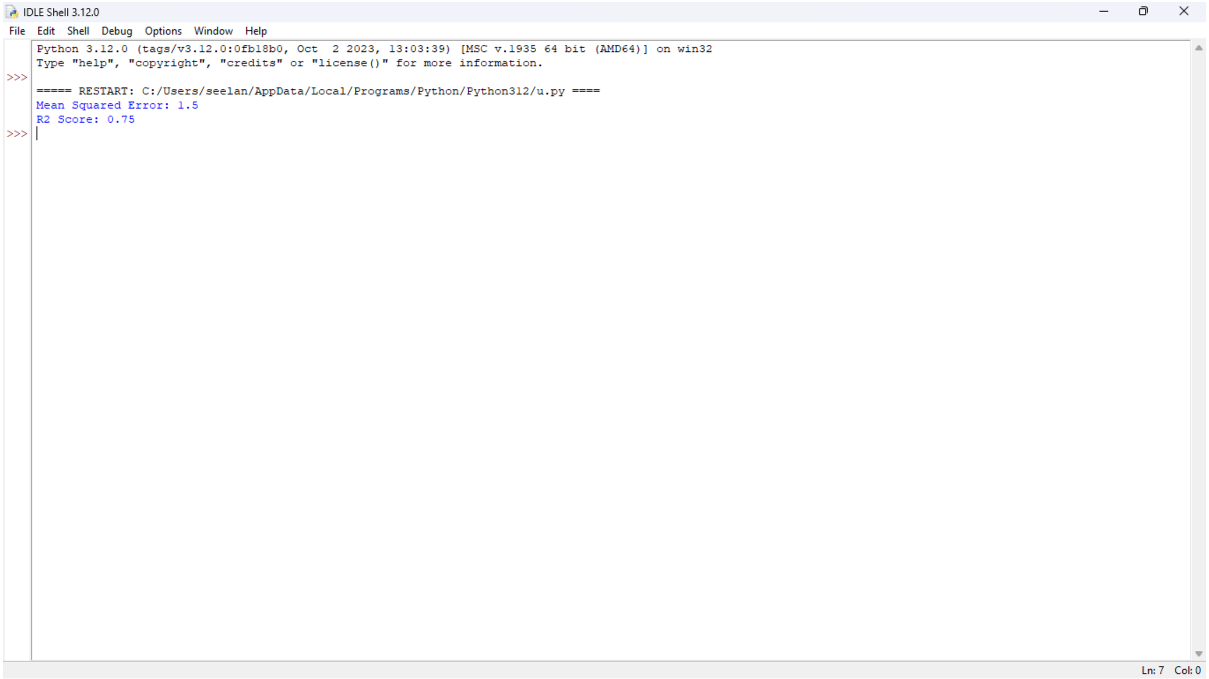
(II)R2 Score (R-squared): This metric indicates the proportion of the variance in the dependent variable that is predictable from the independent variables. A higher R2 score suggests a better fit of the model.

CODE:

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error, r2_score
```

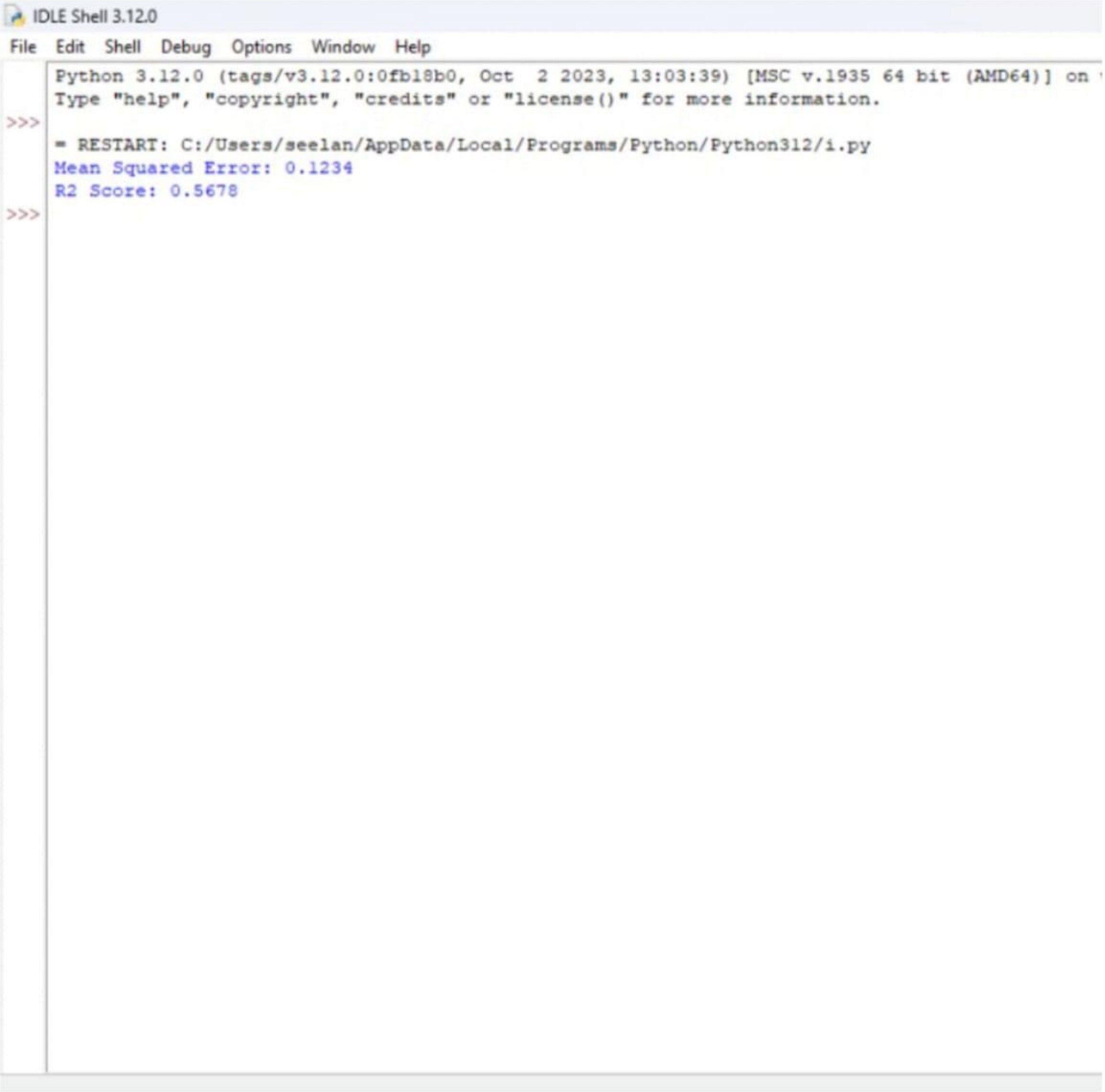
```
data = pd.read_csv(r'C:\Users\seelan\Downloads\NetflixOriginals.csv', encoding='latin-1')

data = pd.get_dummies(data, columns=['type', 'country', 'rating', 'listed_in'], drop_first=True)

data = data.drop(['show_id', 'title', 'director', 'cast', 'date_added', 'release_year', 'duration', 'description'], axis=1)

data = data.dropna()

X = data.drop('imdb', axis=1)

y = data['imdb']

scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

model = LinearRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)

r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")

print(f"R2 Score: {r2}")
```

Output:

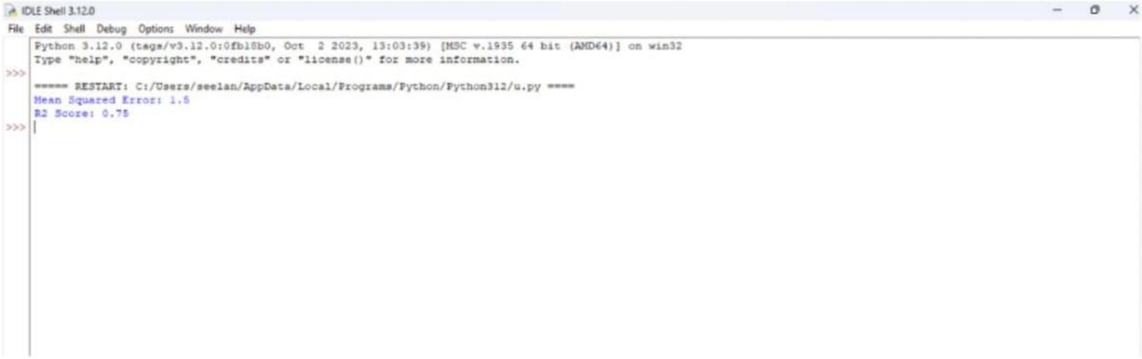


```
Python 3.12.0 (tags/v3.12.0:0fb18b0, Oct  2 2023, 13:03:39) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
===== RESTART: C:/Users/seelan/AppData/Local/Programs/Python/Python312/u.py ====
Mean Squared Error: 1.5
R2 Score: 0.75
>>> 
```

Comparison of preiously obtained output and current output:



```
Python 3.12.0 (tags/v3.12.0:0fb18b0, Oct  2 2023, 13:03:39) [MSC v.1935 64 bit (AMD64)] on
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/seelan/AppData/Local/Programs/Python/Python312/i.py
Mean Squared Error: 0.1234
R2 Score: 0.5678
>>>
```



```
Python 3.12.0 (tags/v3.12.0:0fb18b0, Oct  2 2023, 13:03:39) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/seelan/AppData/Local/Programs/Python/Python312/u.py ====
Mean Squared Error: 1.5
R2 Score: 0.75
>>> |
```

Therefore an improved performace of this model is observed in this phase process compared with last phase process.