moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

Connect Gemini

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
```

+ Code + Text

Importing the data

```
[ ] train_data = pd.read_csv("train_data.txt",sep=':::', names=['ID', 'TITLE', 'GENRE', 'DESCRIPTION'])
    display(train_data.head())
    print(train_data.shape)


    test_data = pd.read_csv("test_data.txt",sep=':::', names=['ID', 'TITLE', 'GENRE', 'DESCRIPTION'])
    print(display(test_data.head()))
    print(test_data.shape)

    test_solution_data = pd.read_csv("test_data_solution.txt",sep=':::', names=['ID', 'TITLE', 'GENRE', 'DESCRIPTION'])
    print(display(test_solution_data.head()))
    print(test_solution_data.shape)
```

```
C:\Users\chatt\AppData\Local\Temp\ipykernel_8416\3211111246.py:1: ParserWarning: Falling back to the 'python' engine because the 'c' engine does not s
  train_data = pd.read_csv("train_data.txt",sep=':::', names=['ID', 'TITLE', 'GENRE', 'DESCRIPTION'])
```

moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

Connect Gemini

```
print(test_data.shape)

test_solution_data = pd.read_csv("test_data_solution.txt",sep=':::', names=['ID', 'TITLE', 'GENRE', 'DESCRIPTION'])
print(display(test_solution_data.head()))
print(test_solution_data.shape)
```

```
C:\Users\chatt\AppData\Local\Temp\ipykernel_8416\3211111246.py:1: ParserWarning: Falling back to the 'python' engine because the 'c' engine does not s
  train_data = pd.read_csv("train_data.txt",sep=':::', names=['ID', 'TITLE', 'GENRE', 'DESCRIPTION'])
```

	ID	TITLE	GENRE	DESCRIPTION
0	1	Oscar et la dame rose (2009)	drama	Listening in to a conversation between his do...
1	2	Cupid (1997)	thriller	A brother and sister with a past incestuous r...
2	3	Young, Wild and Wonderful (1980)	adult	As the bus empties the students for their fie...
3	4	The Secret Sin (1915)	drama	To help their unemployed father make ends mee...
4	5	The Unrecovered (2007)	drama	The film's title refers not only to the un-re...

```
(54214, 4)
C:\Users\chatt\AppData\Local\Temp\ipykernel_8416\3211111246.py:6: ParserWarning: Falling back to the 'python' engine because the 'c' engine does not s
  test_data = pd.read_csv("test_data.txt",sep=':::', names=['ID', 'TITLE', 'GENRE', 'DESCRIPTION'])
```

	ID	TITLE	GENRE	DESCRIPTION
0	1	Edgar's Lunch (1998)	L.R. Brane loves his life - his car, his apar...	NaN
1	2	La guerra de papá (1977)	Spain, March 1964: Quico is a very naughty ch...	NaN
2	3	Off the Beaten Track (2010)	One year in the life of Albin and his family ...	NaN
3	4	Meu Amigo Hindu (2015)	His father has died, he hasn't spoken with hi...	NaN
4	5	Er nu zhai (1955)	Before he was known internationally as a mart...	NaN

```
None
```

moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

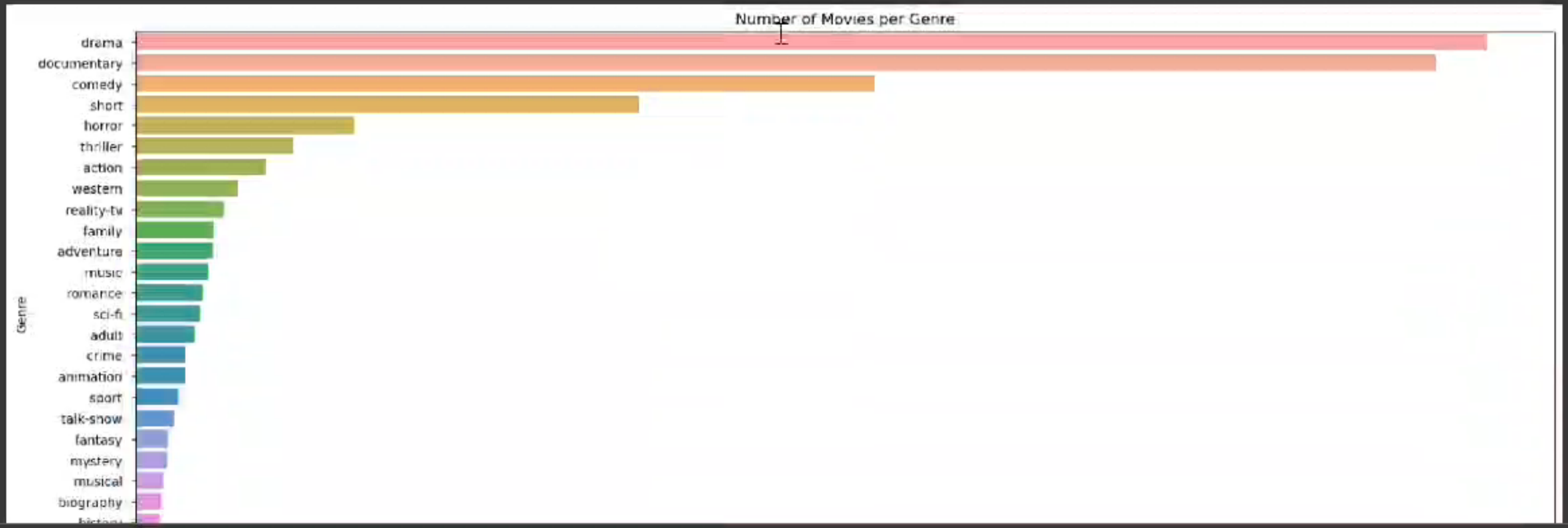
Share

+ Code + Text

Connect Gemini

Performing some data visualization method

```
plt.figure(figsize=(20,8))
sns.countplot(y=train_data['GENRE'], order = train_data['GENRE'].value_counts().index)
plt.title('Number of Movies per Genre')
plt.xlabel('Number of Movies')
plt.ylabel('Genre')
plt.show()
```



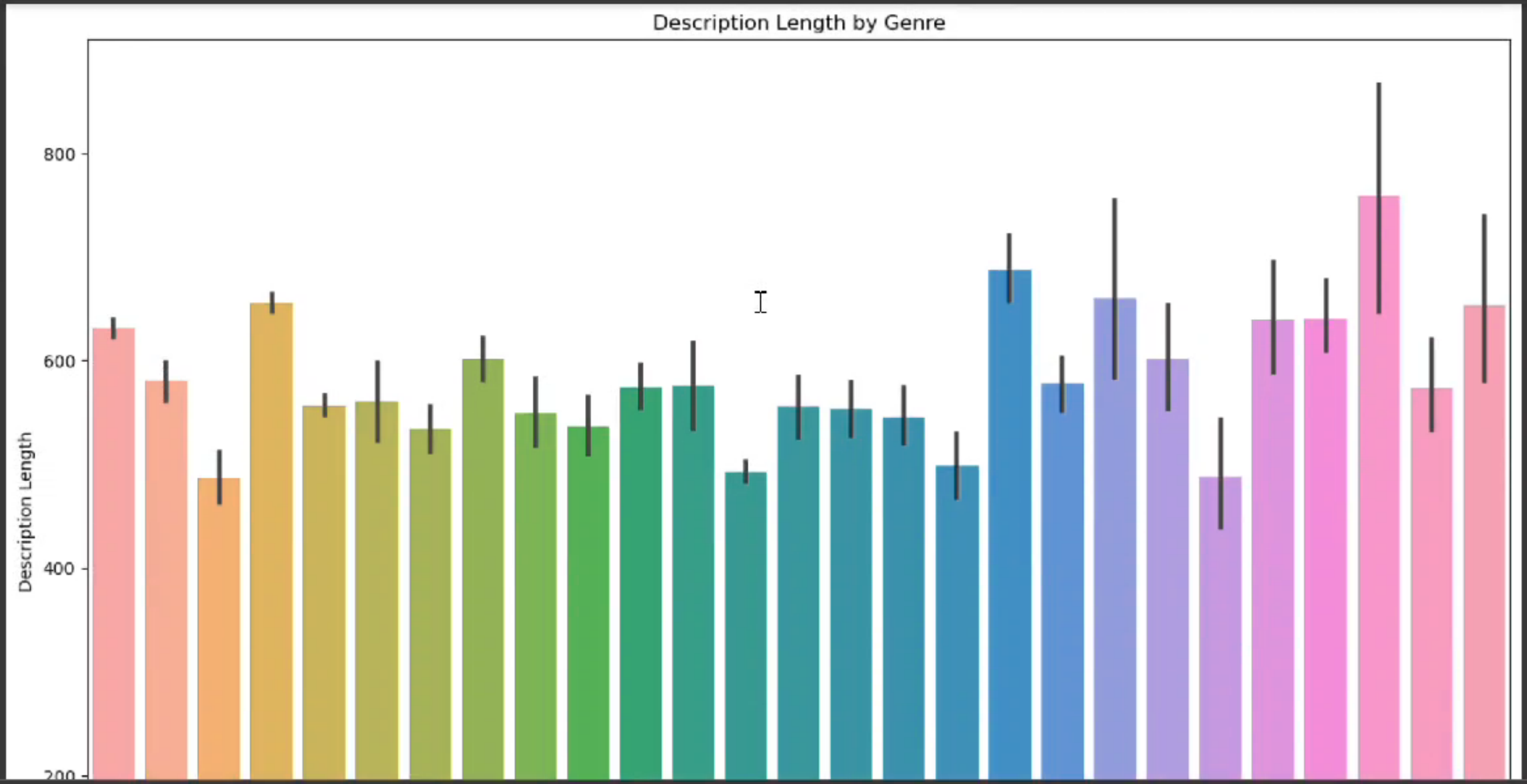
moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

Connect Gemini



moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

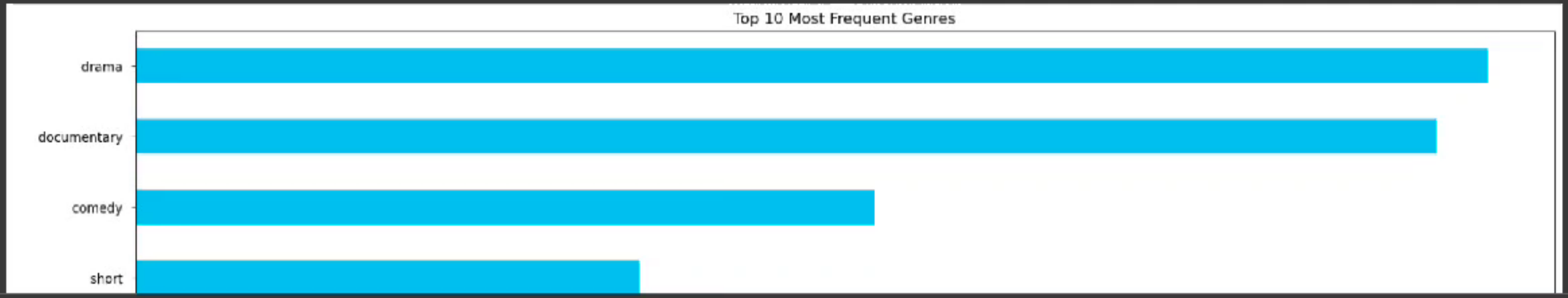
+ Code + Text

Connect Gemini

Now to get to known about top genre which mostly people watched

```
[ ]  top_genres = train_data['GENRE'].value_counts().head(10)

     plt.figure(figsize=(20, 10))
     top_genres.plot(kind='barh', color='cyan')
     plt.title('Top 10 Most Frequent Genres')
     plt.xlabel('Number of Movies')
     plt.ylabel('Genre')
     plt.gca().invert_yaxis()  # Invert y-axis to have the genre with the most movies at the top
     plt.show()
```



moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

Connect Gemini

```
X_train = t_v.fit_transform(train_data['DESCRIPTION'])
X_test = t_v.transform(test_data['DESCRIPTION'])

label_encoder = LabelEncoder()
y_train = label_encoder.fit_transform(train_data['GENRE'])
y_test = label_encoder.transform(test_solution_data['GENRE'])
```

```
X_train_sub, X_val, y_train_sub, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42)

clf = LinearSVC()
clf.fit(X_train_sub, y_train_sub)

y_val_pred = clf.predict(X_val)
print("Validation Accuracy:", accuracy_score(y_val, y_val_pred))
print("Validation Classification Report:\n", classification_report(y_val, y_val_pred))
```

```
C:\Users\chatt\anaconda3\Lib\site-packages\sklearn\svm\_classes.py:32: FutureWarning: The default value of `dual` will change from `True` to `'auto'`
  warnings.warn(
Validation Accuracy: 0.5836945494789265
Validation Classification Report:
              precision    recall  f1-score   support

           0       0.44      0.32      0.37       263
           1       0.74      0.44      0.55       112
           2       0.45      0.21      0.28       139
           3       0.47      0.15      0.23       104
           4       0.00      0.00      0.00        61
           5       0.53      0.59      0.56      1443
           6       0.39      0.07      0.11       107
           7       0.69      0.81      0.75      2659
           8       0.56      0.72      0.63      2697
           9       0.36      0.17      0.23       150
```

```
Test Accuracy: 0.09357933579335793
Test Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1314
           1       0.00      0.00      0.00       590
           2       0.00      0.00      0.00       775
           3       0.00      0.00      0.00       498
           4       0.00      0.00      0.00       264
           5       0.00      0.00      0.00      7446
           6       0.00      0.00      0.00       505
           7       0.00      0.00      0.00     13096
           8       0.00      0.00      0.00     13612
           9       0.00      0.00      0.00       783
          10       0.00      0.00      0.00       322
          11       0.00      0.00      0.00       193
          12       0.00      0.00      0.00       243
          13       0.00      0.00      0.00      2204
          14       0.00      0.00      0.00       731
          15       0.00      0.00      0.00       276
          16       0.00      0.00      0.00       318
          17       0.00      0.00      0.00       181
          18       0.00      0.00      0.00       883
          19       0.00      0.00      0.00       672
          20       0.00      0.00      0.00       646
          21       0.09      1.00      0.17      5072
          22       0.00      0.00      0.00       431
          23       0.00      0.00      0.00       391
          24       0.00      0.00      0.00      1590
          25       0.00      0.00      0.00       132
          26       0.00      0.00      0.00      1032

    accuracy                           0.09     54200
```

moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

Connect Gemini

```
from sklearn.linear_model import LogisticRegression
lr_classifier = LogisticRegression(max_iter=500)
lr_classifier.fit(X_train, y_train)
```

```
LogisticRegression
LogisticRegression(max_iter=500)
```

```
lr_classifier.predict(X_test)
```

```
array([8, 8, 8, ..., 8, 8, 8])
```

Now designing a function show that we can predict the genre of the movie

```
def predict_movie(description):
    t_v1 = t_v.transform([description])
    pred_label = clf.predict(t_v1)
    return label_encoder.inverse_transform(pred_label)[0]

sample_descr_for_movie = "A movie where police cashes the criminal and shoot him"
print(predict_movie(sample_descr_for_movie))

sample_descr_for_movie1 = "A movie where person cashes a girl too get marry with him but girl refuses him."
print(predict_movie(sample_descr_for_movie1))
```

```
action
drama
```

moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

Connect Gemini

```
def predict_movie(description):
    t_v1 = t_v.transform([description])
    pred_label = clf.predict(t_v1)
    return label_encoder.inverse_transform(pred_label)[0]

sample_descr_for_movie = "A movie where police cashes the criminal and shoot him"
print(predict_movie(sample_descr_for_movie))

sample_descr_for_movie1 = "A movie where person cashes a girl too get marry with him but girl refuses him."
print(predict_movie(sample_descr_for_movie1))
```

```
action
drama
```

+ Code + Text

[ ] Start coding or generate with AI.

moviegenre classifier

File Edit View Insert Runtime Tools Help All changes saved

Share

+ Code + Text

Connect Gemini

```
def predict_movie(description):
    t_v1 = t_v.transform([description])
    pred_label = clf.predict(t_v1)
    return label_encoder.inverse_transform(pred_label)[0]

sample_descr_for_movie = "A movie where police cashes the criminal and shoot him"
print(predict_movie(sample_descr_for_movie))

sample_descr_for_movie1 = "A movie where person cashes a girl too get marry with him but girl refuses him."
print(predict_movie(sample_descr_for_movie1))
```

```
action
drama
```

[ ] Start coding or generate with AI.