```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

DATA UPLOAD

```
from google.colab import files
uploaded = files.upload()
```

Choose Files tmdb_5000_movies.csv

- **tmdb_5000_movies.csv**(text/csv) - 5698602 bytes, last modified: 9/19/2019 - 100% done

```
Saving tmdb_5000_movies.csv to tmdb_5000_movies.csv
```

DATA EXPLORATION

```
import pandas as pd

# Automatically gets the filename from the uploaded dictionary
filename = next(iter(uploaded))
df = pd.read_csv(filename)

# Show the first 5 rows of the dataset
df.head()
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popu
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112
4	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://movies.disney.com/john-carter	49529	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	John Carter is a war-weary, former military ca...	43

Next steps: Generate code with df | View recommended plots | New interactive sheet

DESCRIBE

```
# Show statistical summary of numerical columns
df.describe()
# Include all columns, even non-numeric
df.describe(include='all')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4803 entries, 0 to 4802
Data columns (total 20 columns):
```

```
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   budget                4803 non-null   int64  
 1   genres                4803 non-null   object 
 2   homepage              1712 non-null   object 
 3   id                    4803 non-null   int64  
 4   keywords              4803 non-null   object 
 5   original_language     4803 non-null   object 
 6   original_title        4803 non-null   object 
 7   overview              4800 non-null   object 
 8   popularity            4803 non-null   float64
 9   production_companies  4803 non-null   object 
 10  production_countries  4803 non-null   object 
 11  release_date          4802 non-null   object 
 12  revenue               4803 non-null   int64  
 13  runtime               4801 non-null   float64
 14  spoken_languages      4803 non-null   object 
 15  status                4803 non-null   object 
 16  tagline               3959 non-null   object 
 17  title                 4803 non-null   object 
 18  vote_average          4803 non-null   float64
 19  vote_count            4803 non-null   int64  
dtypes: float64(3), int64(4), object(13)
memory usage: 750.6+ KB
```

DATA CLEANING

NULL

```
df.isnull().sum()
```

	0
budget	0
genres	0
homepage	3091
id	0
keywords	0
original_language	0
original_title	0
overview	3
popularity	0
production_companies	0
production_countries	0
release_date	1
revenue	0
runtime	2
spoken_languages	0
status	0
tagline	844
title	0
vote_average	0
vote_count	0

```
# Shows number of missing values in each column
missing_values = df.isnull().sum()
print("Missing values per column:")
print(missing_values)
```

```
Missing values per column:
budget                     0
genres                     0
homepage                3091
id                         0
keywords                   0
original_language          0
original_title             0
overview                   3
```

```
popularity                0
production_companies      0
production_countries      0
release_date              1
revenue                   0
runtime                   2
spoken_languages          0
status                    0
tagline                 844
title                     0
vote_average              0
vote_count                0
dtype: int64
```

DUPLICATE

```
# Count duplicate rows
duplicate_rows = df.duplicated().sum()
print(f"Number of duplicate rows: {duplicate_rows}")
```

```
Number of duplicate rows: 0
```

```
df[df.duplicated()]
```

budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	production_

DATA UPLOAD

```
from google.colab import files
uploaded = files.upload()
```

Choose Files tmdb_5000_credits.csv

- **tmdb_5000_credits.csv**(text/csv) - 40044293 bytes, last modified: 9/19/2019 - 100% done

DATA EXPLORATION

```
import pandas as pd

# Automatically gets the filename from the uploaded dictionary
filename = next(iter(uploaded))
df = pd.read_csv(filename)

# Show the first 5 rows of the dataset
df.head()
```

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...

Next steps: Generate code with df | View recommended plots | New interactive sheet

DESCRIBE

```
# Show statistical summary of numerical columns
df.describe()
# Include all columns, even non-numeric
df.describe(include='all')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4803 entries, 0 to 4802
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   movie_id  4803 non-null   int64
 1   title     4803 non-null   object
 2   cast      4803 non-null   object
```

```
 3   crew      4803 non-null   object
dtypes: int64(1), object(3)
memory usage: 150.2+ KB
```

DATA CLEANING

NULL

```
df.isnull().sum()
```

	0
movie_id	0
title	0
cast	0
crew	0

dtype: int64

```
# Shows number of missing values in each column
missing_values = df.isnull().sum()
print("Missing values per column:")
print(missing_values)
```

```
Missing values per column:
movie_id    0
title       0
cast        0
crew        0
dtype: int64
```

DUPLICATE

```
# Count duplicate rows
duplicate_rows = df.duplicated().sum()
print(f"Number of duplicate rows: {duplicate_rows}")
```

```
Number of duplicate rows: 0
```

```
df[df.duplicated()]
```

	movie_id	title	cast	crew

MERGE

```
import pandas as pd

# Read both CSV files
movies_df = pd.read_csv('tmdb_5000_movies.csv')
credits_df = pd.read_csv('tmdb_5000_credits.csv')

# Merge on the 'title' column
merged_df = movies_df.merge(credits_df, on='title')

# Display the merged dataframe
merged_df.head()
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popu
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112
4	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://movies.disney.com/john-carter	49529	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	John Carter is a war-weary, former military ca...	43

5 rows × 23 columns

ONE-HOT ENCODING

```
import pandas as pd
import ast

# Read and merge the CSV files
movies_df = pd.read_csv('tmdb_5000_movies.csv')
credits_df = pd.read_csv('tmdb_5000_credits.csv')
df = movies_df.merge(credits_df, on='title')

# Parse the 'genres' column (list of dicts) and extract genre names
df['genres'] = df['genres'].apply(lambda x: [i['name'] for i in ast.literal_eval(x)])

# Convert list of genres to individual columns using one-hot encoding
genres_encoded = df['genres'].explode().str.get_dummies().groupby(level=0).sum()

# Concatenate with the original dataframe
df = pd.concat([df, genres_encoded], axis=1)

# Display result
df.head()
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popul
0	237000000	[Action, Adventure, Fantasy, Science Fiction]	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.4
1	300000000	[Adventure, Fantasy, Action]	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.0
2	245000000	[Action, Adventure, Crime]	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.3
3	250000000	[Action, Crime, Drama, Thriller]	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.3
4	260000000	[Action, Adventure, Science Fiction]	http://movies.disney.com/john-carter	49529	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	John Carter is a war-weary, former military ca...	43.9

5 rows × 43 columns

TRAIN TEST

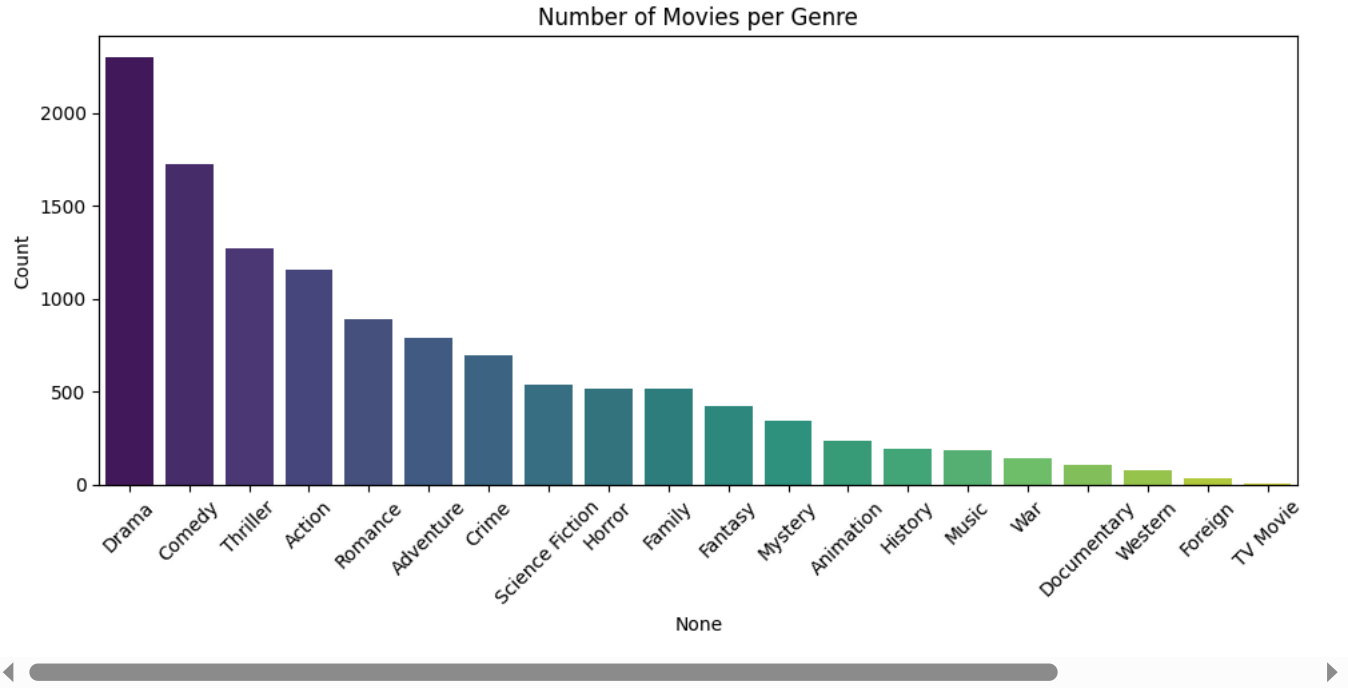
```
import pandas as pd
import ast
from sklearn.model_selection import train_test_split

# Read and merge CSV files
movies_df = pd.read_csv('tmdb_5000_movies.csv')
credits_df = pd.read_csv('tmdb_5000_credits.csv')
df = movies_df.merge(credits_df, on='title')

# One-hot encode the 'genres' column
df['genres'] = df['genres'].apply(lambda x: [i['name'] for i in ast.literal_eval(x)])
genres_encoded = df['genres'].explode().str.get_dummies().groupby(level=0).sum()
df = pd.concat([df, genres_encoded], axis=1)

# Example feature and target selection
X = df[genres_encoded.columns]  # One-hot encoded genres as features
y = df['vote_average']
```

VISUALIZATION

```
import pandas as pd
import ast
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split

# Step 1: Load and merge data
movies_df = pd.read_csv('tmdb_5000_movies.csv')
credits_df = pd.read_csv('tmdb_5000_credits.csv')
df = movies_df.merge(credits_df, on='title')

# Step 2: One-hot encode genres
df['genres'] = df['genres'].apply(lambda x: [i['name'] for i in ast.literal_eval(x)])
genres_encoded = df['genres'].explode().str.get_dummies().groupby(level=0).sum()
df = pd.concat([df, genres_encoded], axis=1)
```

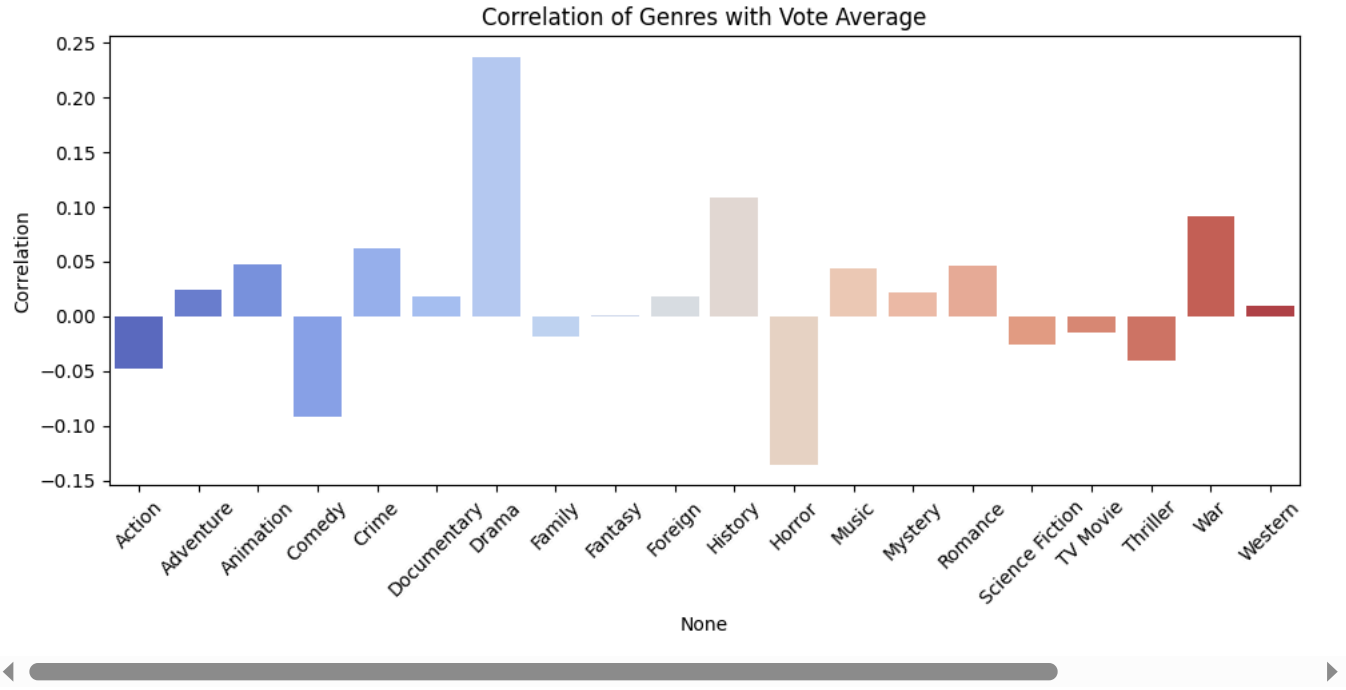
```
# Step 3: Train-test split
X = df[genres_encoded.columns]
y = df['vote_average']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 4: Visualization

# Genre frequency
plt.figure(figsize=(10, 5))
genre_counts = genres_encoded.sum().sort_values(ascending=False)
sns.barplot(x=genre_counts.index, y=genre_counts.values, palette="viridis")
plt.title('Number of Movies per Genre')
plt.xticks(rotation=45)
plt.ylabel('Count')
plt.tight_layout()
plt.show()
```

```
<ipython-input-27-934ea965d3b3>:27: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(x=genre_counts.index, y=genre_counts.values, palette="viridis")
```




```
# Correlation between genres and vote_average
corr = pd.concat([genres_encoded, y], axis=1).corr()['vote_average'].drop('vote_average')
plt.figure(figsize=(10, 5))
sns.barplot(x=corr.index, y=corr.values, palette='coolwarm')
plt.title('Correlation of Genres with Vote Average')
plt.xticks(rotation=45)
plt.ylabel('Correlation')
plt.tight_layout()
plt.show()
```

```
<ipython-input-28-3d68d7da85b4>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(x=corr.index, y=corr.values, palette='coolwarm')
```

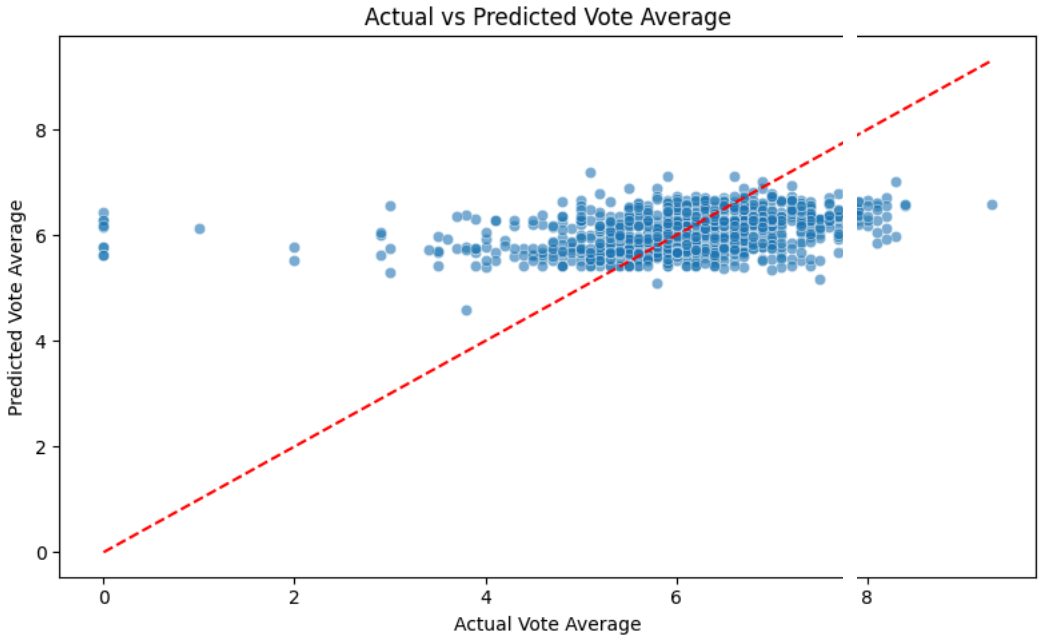


MODEL

```
import pandas as pd
import ast
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np # Import numpy for sqrt

# Step 1: Read and merge data
movies_df = pd.read_csv('tmdb_5000_movies.csv')
credits_df = pd.read_csv('tmdb_5000_credits.csv')
df = movies_df.merge(credits_df, on='title')

# Step 2: One-hot encode genres
df['genres'] = df['genres'].apply(lambda x: [i['name'] for i in ast.literal_eval(x)])
genres_encoded = df['genres'].explode().str.get_dummies().groupby(level=0).sum()
df = pd.concat([df, genres_encoded], axis=1)

# Step 3: Split features and target
X = df[genres_encoded.columns]         # One-hot encoded genres
y = df['vote_average']                 # Target variable
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42

# Step 4: Model building
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# Step 5: Evaluation
# Calculate Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
# Calculate Root Mean Squared Error by taking the square root of MSE
rmse = np.sqrt(mse)

r2 = r2_score(y_test, y_pred)
print(f"RMSE: {rmse:.2f}")
print(f"R² Score: {r2:.2f}")

# Step 6: Visualization
plt.figure(figsize=(8,5))
sns.scatterplot(x=y_test, y=y_pred, alpha=0.6)
plt.xlabel("Actual Vote Average")
plt.ylabel("Predicted Vote Average")
plt.title("Actual vs Predicted Vote Average")
# Make sure the plot line spans the actual data range
```

```
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--r')
plt.tight_layout()
plt.show()
```

```
RMSE: 1.07
R² Score: 0.11
```



DEPLOYMENT

```
import pandas as pd
```