cob-phase-1

February 4, 2024

0.0.1 1. Create a csv dataset using python , pandas and any public api:

[1]:
```
!pip install pandas requests
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(1.5.3)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (2.31.0)
Requirement already satisfied: python-dateutil>=2.8.1 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas) (2023.4)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-
packages (from pandas) (1.23.5)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests) (2023.11.17)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
```

[2]:
```
import requests

def check_api_key(api_key):
    url = f"http://api.openweathermap.org/data/2.5/weather?
 ↪q=London&appid={api_key}"
    response = requests.get(url)
    if response.status_code == 200:
        print("API key is valid.")
    else:
        print(f"API key is invalid. Status Code: {response.status_code}")

api_key = "b589980eb3bde31a19a2d66804981916"
check_api_key(api_key)
```

```
API key is valid.
```

[3]:
```python
import pandas as pd
import requests

def fetch_weather_data(api_key, city):
    url = f"http://api.openweathermap.org/data/2.5/weather?
 ↪q={city}&appid={api_key}"
    response = requests.get(url)
    if response.status_code == 200:
        data = response.json()
        weather_data = {
            "City": data["name"],
            "Country": data["sys"]["country"],
            "Temperature (Celsius)": data["main"]["temp"] - 273.15,  # Convert␣
 ↪from Kelvin to Celsius
            "Humidity (%)": data["main"]["humidity"],
            "Wind Speed (m/s)": data["wind"]["speed"],
            "Weather Description": data["weather"][0]["description"]
        }
        return weather_data
    else:
        print(f"Failed to fetch data for {city}. Status Code: {response.
 ↪status_code}")
        return None

def save_to_csv(data, filename):
    df = pd.DataFrame(data)
    df.to_csv(filename, index=False)
    print(f"Data saved to {filename}")

def main():
    api_key = "b589980eb3bde31a19a2d66804981916"
    city = "New York"  # You can change this to any city
    data = fetch_weather_data(api_key, city)
    if data:
        save_to_csv([data], "weather_data.csv")

if __name__ == "__main__":
    main()
```

```
Data saved to weather_data.csv
```

0.0.2 2. Clean the dataset replace missing values, remove outliers etc.

[4]:
```python
import pandas as pd
```

[5]:
```
# Load the dataset
df = pd.read_csv("/content/dataset - netflix1.csv")
df.head()
```

[5]:

	show_id	type	title	director	country	date_added	release_year	rating	duration	listed_in
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	United States	9/25/2021	2020	PG-13	90 min	Documentaries
1	s3	TV Show	Ganglands	Julien Leclercq	France	9/24/2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act…
2	s6	TV Show	Midnight Mass	Mike Flanagan	United States	9/24/2021	2021	TV-MA	1 Season	TV Dramas, TV Horror, TV Mysteries
3	s14	Movie	Confessions of an Invisible Girl	Bruno Garotti	Brazil	9/22/2021	2021	TV-PG	91 min	Children & Family Movies, Comedies
4	s8	Movie	Sankofa	Haile Gerima	United States	9/24/2021	1993	TV-MA	125 min	Dramas, Independent Movies, International Movies

Replace missing values:

[6]:
```
# Replace missing values
df.fillna({
    'show_id': 'unknown_show_id',
    'type': 'unknown_type',
    'title': 'unknown_title',
    'director': 'unknown_director',
    'country': 'unknown_country',
    'date_added': 'unknown_date_added',
    'release_year': 0,
    'rating': 'unknown_rating',
    'duration': 'unknown_duration',
    'listed_in': 'unknown_listed_in'
}, inplace=True)
```

Removing Outliers:

[7]:
```
# Calculate the first and third quartiles
Q1 = df['release_year'].quantile(0.25)
Q3 = df['release_year'].quantile(0.75)

# Calculate the interquartile range (IQR)
```

```
IQR = Q3 - Q1

# Define the lower and upper bounds to identify outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Remove outliers from the 'release_year' column
df = df[(df['release_year'] >= lower_bound) & (df['release_year'] <=␣
 ↪upper_bound)]
```

Check for duplicates:

[8]:
```
# Check for duplicate entries
df.drop_duplicates(inplace=True)
df.head()
```

[8]:

	show_id	type	title	director	country	date_added	release_year	rating	duration	listed_in
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	United States	9/25/2021	2020	PG-13	90 min	Documentaries
1	s3	TV Show	Ganglands	Julien Leclercq	France	9/24/2021	2021	TV-MA	1 Season	Crime TV Shows, International TV Shows, TV Act…
2	s6	TV Show	Midnight Mass	Mike Flanagan	United States	9/24/2021	2021	TV-MA	1 Season	TV Dramas, TV Horror, TV Mysteries
3	s14	Movie	Confessions of an Invisible Girl	Bruno Garotti	Brazil	9/22/2021	2021	TV-PG	91 min	Children & Family Movies, Comedies
5	s9	TV Show	The Great British Baking Show	Andy Devonshire	United Kingdom	9/24/2021	2021	TV-14	9 Seasons	British TV Shows, Reality TV

Data type conversion:

[9]:
```
# Converting Data Types
df['date_added'] = pd.to_datetime(df['date_added'])
# If it's categorical, convert it to string
df['release_year'] = df['release_year'].astype(str)
```

Feature Engineering:

[10]:
```
# Feature Engineering
df['date_added_year'] = df['date_added'].dt.year
```

```
df['date_added_month'] = df['date_added'].dt.month
df['date_added_dayofweek'] = df['date_added'].dt.dayofweek
# Assuming duration is in minutes
df['duration_minutes'] = df['duration'].str.extract('(\d+)').astype(float)
```

Data cleaning:

[11]:
```
# Text Data Cleaning
text_cols = ['director', 'listed_in']

# Check if the columns exist in the DataFrame before performing operations
for col in text_cols:
    if col in df.columns:
        df[col] = df[col].str.lower()
        df[col] = df[col].str.replace('[^\w\s]', '')  # Remove special␣
 ↪characters
    else:
        print(f"Column '{col}' does not exist in the DataFrame.")
```

```
<ipython-input-11-0b538afed5b7>:8: FutureWarning: The default value of regex
will change from True to False in a future version.
  df[col] = df[col].str.replace('[^\w\s]', '')  # Remove special characters
```

[12]:
```
# Remove columns like 'show_id' or 'title' if not relevant for analysis
df.drop(['show_id', 'title'], axis=1, inplace=True)
df.head()
```

[12]:

	type	director	country	date_added	release_year	rating	duration	listed_in	date_added_year	date_added_month	date_added_dayofweek	duration_minutes
0	Movie	kirsten johnson	United States	2021-09-25	2020	PG-13	90 min	documentaries	2021	9	5	90.0
1	TV Show	julien leclercq	France	2021-09-24	2021	TV-MA	1 Season	crime tv shows international tv shows tv actio…	2021	9	4	1.0
2	TV Show	mike flanagan	United States	2021-09-24	2021	TV-MA	1 Season	tv dramas tv horror tv mysteries	2021	9	4	1.0
3	Movie	bruno garotti	Brazil	2021-09-22	2021	TV-PG	91 min	children family movies comedies	2021	9	2	91.0
5	TV Show	andy devonshire	United Kingdom	2021-09-24	2021	TV-14	9 Seasons	british tv shows reality tv	2021	9	4	9.0

Data Splitting:

[13]:
```
from sklearn.model_selection import train_test_split
```

[14]:
```
# Split dataset into features and target variable
X = df.drop(columns=["release_year"])
y = df["release_year"]

# Split dataset into training set and test set (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Print the shapes of the resulting datasets
print("Training set shape:", X_train.shape, y_train.shape)
print("Testing set shape:", X_test.shape, y_test.shape)
```

```
Training set shape: (6458, 11) (6458,)
Testing set shape: (1615, 11) (1615,)
```

Random forest:

[15]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.metrics import accuracy_score
```

[16]:
```
# Define features (X) and target variable (y)
X = df[['director', 'country', 'release_year', 'rating', 'duration',␣
 ↪'listed_in']]
y = df['type']  # Assuming 'type' is the target variable (TV Show or Movie)

# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Preprocessing for categorical features
categorical_features = ['director', 'country', 'rating', 'listed_in']
categorical_transformer = Pipeline(steps=[
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

# Combine preprocessing steps
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', categorical_transformer, categorical_features)
    ])
```

```
# Define the Random Forest Classifier pipeline
model_rf = Pipeline(steps=[('preprocessor', preprocessor),
                           ('classifier',␣
 ↪RandomForestClassifier(n_estimators=100, random_state=42))])

# Train the model on the training data
model_rf.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model_rf.predict(X_test)

# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.9820433436532507
```

[17]:
```
# Save the cleaned dataset
df.to_csv("cleaned_dataset.csv", index=False)
```