import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

df = pd.read\_csv("C:/Users/ragha/OneDrive/Desktop/df\_modcloth.csv")

print("Dataset Loaded Successfully”)

print(df.head())

print(df.info())

df = df.dropna()

target\_column = df.columns[-1] # Assuming the last column is the target

print(f'Target column identified: {target\_column}')

label\_encoders = {}

for column in df.select\_dtypes(include=['object']).columns:

le = LabelEncoder()

df[column] = le.fit\_transform(df[column])

label\_encoders[column] = le

X = df.drop(columns=[target\_column])

y = df[target\_column]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print(f'Accuracy: {accuracy:.2f}')

print(classification\_report(y\_test, y\_pred))

df['category'].value\_counts().plot.pie(autopct='%1.1f%%', startangle=90,cmap='Set3', figsize=(8,8))

plt.title('Category Distribution')

plt.ylabel('')

plt.show()

plt.figure(figsize=(12,6))

sns.countplot(data=df, x='category', palette='viridis')

plt.xticks(rotation=45)

plt.title('count of Items per Category')

plt.xlabel('category')

plt.ylabel('count')

plt.show()

plt.figure(figsize=(10, 6))

corr\_matrix = df.corr()

sns.heatmap(corr\_matrix, annot=True, fmt=".2f", cmap='coolwarm', linewidths=0.5)

plt.title("Correlation Heatmap")

plt.show()

plt.figure(figsize=(8,6))

sns.histplot(df['rating'], bins=20, kde=True, color='blue')

plt.title('Rating Distribution')

plt.xlabel('Rating')

plt.ylabel('Frequency')

plt.show()