import base64

import io

import pandas as pd

import plotly.express as px

import dash

from dash import dcc, html

from dash.dependencies import Input, Output

import matplotlib.pyplot as plt

import seaborn as sns

# Load data

df = pd.read\_csv('C:/music\_data\_dash/spotify\_top\_songs\_audio\_features.csv')

# Preprocess data to create the 'popularity\_category' column

def preprocess\_data(data):

"""

Preprocess the dataset to add the 'popularity\_category' column.

"""

# Add 'popularity\_category' based on quantiles of the 'streams' column

data['popularity\_category'] = pd.qcut(data['streams'], q=3, labels=['Low', 'Medium', 'High'])

return data

df = preprocess\_data(df)

# Heatmap generation using matplotlib

def generate\_heatmap\_image(data):

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

heatmap\_data = pd.crosstab(data['key'], data['mode'])

sns.heatmap(heatmap\_data, annot=True, fmt='d', cmap='Blues')

plt.title("Key and Mode Combinations")

plt.xlabel("Mode")

plt.ylabel("Key")

buffer = io.BytesIO()

plt.savefig(buffer, format='png')

buffer.seek(0)

heatmap\_image = base64.b64encode(buffer.read()).decode('utf-8')

plt.close()

return heatmap\_image

heatmap\_image = generate\_heatmap\_image(df)

# App setup

app = dash.Dash(\_\_name\_\_)

app.layout = html.Div(

style={'backgroundColor': '#f4f4f4', 'font-family': 'Arial'},

children=[

html.H1("Spotify Music Dashboard 🎵", style={'textAlign': 'center', 'color': 'green'}),

html.H4("Explore Spotify's top songs with interactive visuals.", style={'textAlign': 'center'}),

# Top Songs by Weeks on Chart

html.Div([

dcc.Graph(

id='bar-chart',

figure=px.bar(

df.nlargest(10, 'weeks\_on\_chart'),

x='weeks\_on\_chart',

y='track\_name',

orientation='h',

color='track\_name',

title='Top 10 Songs by Weeks on Chart'

).update\_layout(showlegend=False)

)

]),

# Streams vs. Weeks on Chart

html.Div([

dcc.Graph(

id='scatter-plot',

figure=px.scatter(

df, x='streams', y='weeks\_on\_chart',

color='popularity\_category',

log\_x=True,

title='Streams vs. Weeks on Chart (Log Scale)'

).update\_traces(marker=dict(size=8)).update\_layout(showlegend=True)

)

]),

# Danceability by Popularity Category

html.Div([

dcc.Graph(

id='violin-plot',

figure=px.violin(

df, x='popularity\_category', y='danceability',

color='popularity\_category', box=True, points=False

).update\_traces(meanline\_visible=True).update\_layout(

title="Danceability by Popularity Category",

annotations=[

dict(

x=cat, y=median, text=f"{median:.2f}", showarrow=False, font=dict(color="red", size=14)

)

for cat, median in zip(df['popularity\_category'].unique(),

df.groupby('popularity\_category')['danceability'].median())

]

)

)

]),

# Key and Mode Combinations (Static Heatmap)

html.Div([

html.H3("Key and Mode Combinations", style={'textAlign': 'center'}),

html.Img(

src=f"data:image/png;base64,{heatmap\_image}",

style={'display': 'block', 'margin': '0 auto', 'max-width': '80%'}

)

])

]

)

if \_\_name\_\_ == '\_\_main\_\_':

app.run\_server(debug=True)