```
!pip install dash

import pandas as pd
import plotly.express as pl
import dash
import dash_core_components as dcc
import dash_html_components as html
import plotly.graph_objs as go
from dash.dependencies import Input, Output

data = pd.read_csv("/content/netflix_titles.csv")

data.info()

data.isna().sum()

data = data.dropna()

data.isna().sum()

#visualising how the number of tv shows and movies varries across the
years
data['release_year'] = data['date_added'].str[-4:].astype(int)

tv_shows = data[data['type'] == 'TV Show']
movies = data[data['type'] == 'Movie']

# Group by release year and count the number of TV shows and movies
tv_shows_by_year =
tv_shows.groupby('release_year').size().reset_index(name='Count')
movies_by_year =
movies.groupby('release_year').size().reset_index(name='Count')

# Create the interactive line plot using Plotly
fig = pl.line(data_frame=tv_shows_by_year, x='release_year', y='Count',
labels={'Count': 'Number of TV Shows'},
              title='Number of TV Shows on Netflix over the Years')
fig.add_scatter(x=movies_by_year['release_year'],
y=movies_by_year['Count'], mode='lines',
                name='Number of Movies', line=dict(color='orange'))
```

```
# Customize the layout
fig.update_layout(xaxis_title='Release Year', yaxis_title='Number of
Titles', legend_title='Type',
                  hovermode='x unified')

# Show the plot
fig.show()

#Which directors have the most titles available on Netflix?
directors_count = data['director'].value_counts().head(10).reset_index()
directors_count.columns = ['director', 'title']

# Create the horizontal bar chart using Plotly
fig = pl.bar(directors_count, x='title', y='director', orientation='h',
             labels={'Number of Titles': 'title'}, title='Top 10 Directors
with the Most Titles on Netflix',
             color='title', color_continuous_scale='Blues')

# Customize the layout
fig.update_layout(xaxis_title='Number of Titles', yaxis_title='Director',
                  coloraxis_colorbar_title='Number of Titles',
coloraxis_showscale=False)

# Show the plot
fig.show()

#Create a dashboard layout using Plotly Dash, which allows you to combine
multiple visualizations into an interactive dashboard.
#Add interactive components to the dashboard, such as dropdown menus,
sliders, or buttons, to enable users to interact with the data.
#Incorporate tooltips or hover effects to provide additional information
when users interact with the visualizations.
directors_count = data['director'].value_counts().head(10).reset_index()
directors_count.columns = ['director', 'title']

# Preprocess data for TV shows and movies count by release year
tv_shows_by_year = data[data['type'] == 'TV
Show']['release_year'].value_counts().sort_index()
```

```
movies_by_year = data[data['type'] ==
'Movie']['release_year'].value_counts().sort_index()

# Create the Dash app
app = dash.Dash(__name__)

# Define the dashboard layout
app.layout = html.Div([
    html.H1('Netflix Data Dashboard'),

    dcc.Dropdown(
        id='content-type-dropdown',
        options=[
            {'label': 'TV Shows', 'value': 'TV Show'},
            {'label': 'Movies', 'value': 'Movie'}
        ],
        value='TV Show',
        style={'width': '50%'}
    ),

    dcc.Graph(
        id='top-directors-chart',
    ),

    dcc.Graph(
        id='tv-movies-trend-chart',
    ),

    dcc.Slider(
        id='release-year-slider',
        min=data['release_year'].min(),
        max=data['release_year'].max(),
        value=data['release_year'].min(),
        marks={str(year): str(year) for year in
data['release_year'].unique()},
        step=None
    )
])

# Callback to update the figures based on user input
```

```
@app.callback(
    Output('top-directors-chart', 'figure'),
    Output('tv-movies-trend-chart', 'figure'),
    Input('content-type-dropdown', 'value'),
    Input('release-year-slider', 'value')
)
def update_figures(content_type, year):
    # Filter data based on content type
    filtered_data = data[data['type'] == content_type]

    # Filter data based on release year
    filtered_data = filtered_data[filtered_data['release_year'] == year]

    # Update Top Directors chart
    directors_count =
filtered_data['director'].value_counts().head(10).reset_index()
    directors_count.columns = ['director', 'title']

    top_directors_chart = {
        'data': [
            go.Bar(
                x=directors_count['title'],
                y=directors_count['director'],
                orientation='h',
                marker=dict(color='skyblue')
            )
        ],
        'layout': go.Layout(
            title=f'Top 10 Directors with the Most {content_type} on
Netflix ({year})',
            xaxis=dict(title='title'),
            yaxis=dict(title='director'),
            height=400
        )
    }

    # Update TV/Movies Trend chart
    tv_shows_by_year = filtered_data[filtered_data['type'] == 'TV
Show']['release_year'].value_counts().sort_index()
```

```
    movies_by_year = filtered_data[filtered_data['type'] ==
'Movie']['release_year'].value_counts().sort_index()

    tv_movies_trend_chart = {
        'data': [
            go.Scatter(
                x=tv_shows_by_year.index,
                y=tv_shows_by_year.values,
                mode='lines+markers',
                name='TV Shows',
                hoverinfo='x+y'
            ),
            go.Scatter(
                x=movies_by_year.index,
                y=movies_by_year.values,
                mode='lines+markers',
                name='Movies',
                line=dict(color='orange'),
                hoverinfo='x+y'
            )
        ],
        'layout': go.Layout(
            title=f'Number of TV Shows and Movies on Netflix over the
Years ({content_type})',
            xaxis=dict(title='release_year'),
            yaxis=dict(title='title'),
            height=400
        )
    }

    return top_directors_chart, tv_movies_trend_chart

# Run the app
if __name__ == '__main__':
    app.run_server(debug=True)
```