. Background and Introduction: Topic: The economic outcomes of college majors and their impact on job prospects and income inequality. Goal: Advocate for informed decision-making in higher education and highlight the importance of supporting students in fields with lower economic returns. Key Question: Which major categories yield the highest median earnings, and how do they compare in terms of unemployment rates?

2. Questions to Address: Which major categories offer the highest and lowest median earnings? How does the unemployment rate vary across different majors, and what are the trade-offs between earnings and job stability? How do gender distributions within these majors correlate with median earnings?
3. Visualizations and Rationale:

Visualization 1: Animated Bar Chart for Median Earnings by Major Category. Purpose: Show disparities in median earnings, emphasizing economic inequality between fields. Design Decision: Use an animated bar chart to guide viewers through each category, allowing for a focused analysis of each group.

Visualization 2: Interactive Scatter Plot for Median Earnings vs. Unemployment Rate. Purpose: Illustrate the trade-offs between potential earnings and job stability, advocating for informed career decisions. Design Decision: Include hover and filter features to let users explore data for specific majors and categories interactively.

4. Implementation Plan: Tools: Use Plotly for creating interactive and animated visualizations in Python. Data Preparation: Load the recent-grads.csv dataset. Clean and preprocess the data as needed. Code for Visualizations:

```
pip install dash
```

```
Requirement already satisfied: dash in /usr/local/lib/python3.10/dist-packages (2.18.2)
Requirement already satisfied: Flask<3.1,>=1.0.4 in /usr/local/lib/python3.10/dist-packages (from dash) (3.0.3)
Requirement already satisfied: Werkzeug<3.1 in /usr/local/lib/python3.10/dist-packages (from dash) (3.0.6)
Requirement already satisfied: plotly>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (5.24.1)
Requirement already satisfied: dash-html-components==2.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (2.0.0)
Requirement already satisfied: dash-core-components==2.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (2.0.0)
Requirement already satisfied: dash-table==5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (5.0.0)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.10/dist-packages (from dash) (8.5.0)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.10/dist-packages (from dash) (4.12.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from dash) (2.32.3)
Requirement already satisfied: retrying in /usr/local/lib/python3.10/dist-packages (from dash) (1.3.4)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.10/dist-packages (from dash) (1.6.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from dash) (75.1.0)
Requirement already satisfied: Jinja2>=3.1.2 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (3.1.4)
Requirement already satisfied: itsdangerous>=2.1.2 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (2.2.0)
Requirement already satisfied: click>=8.1.3 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (8.1.7)
Requirement already satisfied: blinker>=1.6.2 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (1.9.0)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash) (9.0.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash) (24.2)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from Werkzeug<3.1->dash) (3.0.2)
Requirement already satisfied: zipp>=3.20 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata->dash) (3.21.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->dash) (2024.8.30)
Requirement already satisfied: six>=1.7.0 in /usr/local/lib/python3.10/dist-packages (from retrying->dash) (1.16.0)
```

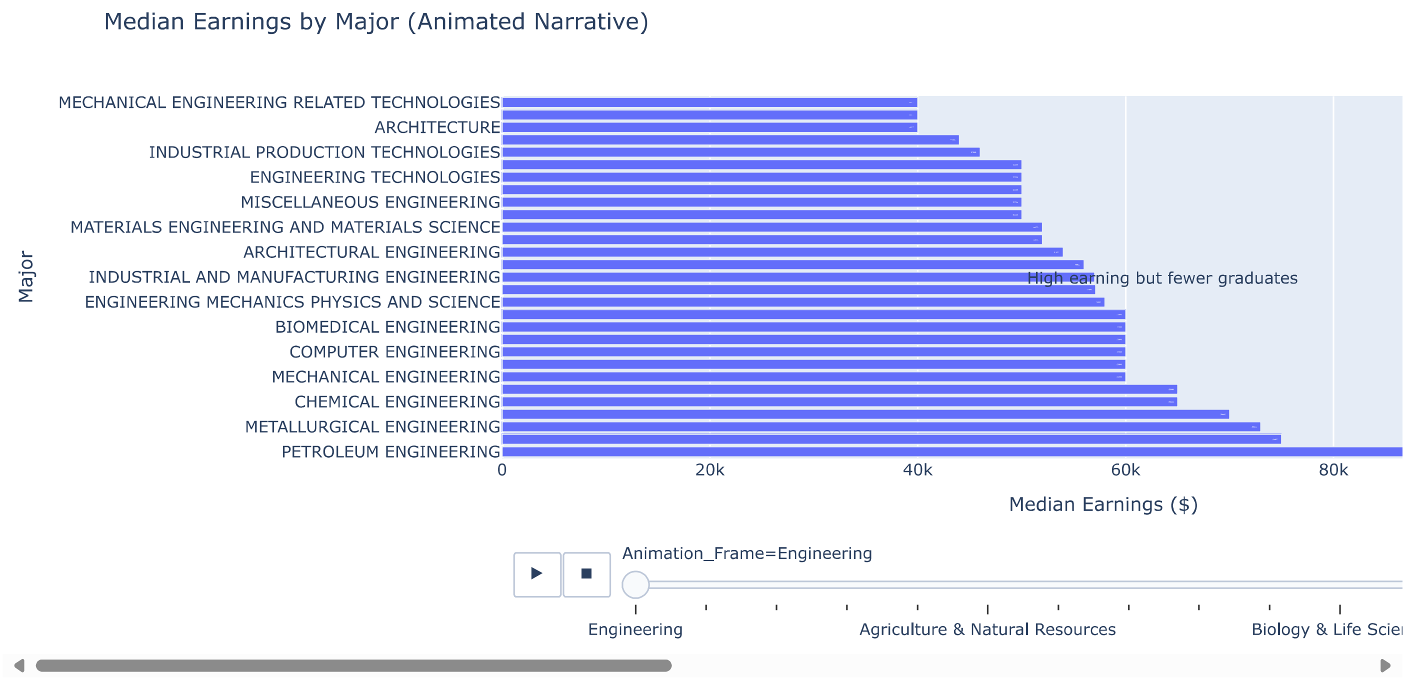
```
!pip install jupyter-dash
```

```
Requirement already satisfied: jupyter-dash in /usr/local/lib/python3.10/dist-packages (0.4.2)
Requirement already satisfied: dash in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (2.18.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (2.32.3)
Requirement already satisfied: flask in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (3.0.3)
Requirement already satisfied: retrying in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (1.3.4)
Requirement already satisfied: ipython in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (7.34.0)
Requirement already satisfied: ipykernel in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (5.5.6)
Requirement already satisfied: ansi2html in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (1.9.2)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.10/dist-packages (from jupyter-dash) (1.6.0)
Requirement already satisfied: Werkzeug<3.1 in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (3.0.6)
Requirement already satisfied: plotly>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (5.24.1)
Requirement already satisfied: dash-html-components==2.0.0 in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (2.0.0)
Requirement already satisfied: dash-core-components==2.0.0 in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (2.0.0)
Requirement already satisfied: dash-table==5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (5.0.0)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (8.5.0)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (4.12.2)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from dash->jupyter-dash) (75.1.0)
Requirement already satisfied: Jinja2>=3.1.2 in /usr/local/lib/python3.10/dist-packages (from flask->jupyter-dash) (3.1.4)
Requirement already satisfied: itsdangerous>=2.1.2 in /usr/local/lib/python3.10/dist-packages (from flask->jupyter-dash) (2.2.0)
Requirement already satisfied: click>=8.1.3 in /usr/local/lib/python3.10/dist-packages (from flask->jupyter-dash) (8.1.7)
Requirement already satisfied: blinker>=1.6.2 in /usr/local/lib/python3.10/dist-packages (from flask->jupyter-dash) (1.9.0)
Requirement already satisfied: ipython-genutils in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter-dash) (0.2.0)
Requirement already satisfied: traitlets>=4.1.0 in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter-dash) (5.7.1)
Requirement already satisfied: jupyter-client in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter-dash) (6.1.12)
Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter-dash) (6.3.3)
Requirement already satisfied: jedi>=0.16 in /usr/local/lib/python3.10/dist-packages (from ipython->jupyter-dash) (0.19.2)
Requirement already satisfied: decorator in /usr/local/lib/python3.10/dist-packages (from ipython->jupyter-dash) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.10/dist-packages (from ipython->jupyter-dash) (0.7.5)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from ipython->ju
Requirement already satisfied: pygments in /usr/local/lib/python3.10/dist-packages (from ipython->jupyter-dash) (2.18.0)
Requirement already satisfied: backcall in /usr/local/lib/python3.10/dist-packages (from ipython->jupyter-dash) (0.2.0)
Requirement already satisfied: matplotlib-inline in /usr/local/lib/python3.10/dist-packages (from ipython->jupyter-dash) (0.1.7)
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.10/dist-packages (from ipython->jupyter-dash) (4.9.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->jupyter-dash) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->jupyter-dash) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->jupyter-dash) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->jupyter-dash) (2024.8.30)
Requirement already satisfied: six>=1.7.0 in /usr/local/lib/python3.10/dist-packages (from retrying->jupyter-dash) (1.16.0)
Requirement already satisfied: parso<0.9.0,>=0.8.4 in /usr/local/lib/python3.10/dist-packages (from jedi>=0.16->ipython->jupyter-dash) (
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2>=3.1.2->flask->jupyter-dash) (3.0
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.10/dist-packages (from pexpect>4.3->ipython->jupyter-dash) (0.7
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash->jupyter-dash) (9.0.
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly>=5.0.0->dash->jupyter-dash) (24.2)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.10/dist-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ip
Requirement already satisfied: zipp>=3.20 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata->dash->jupyter-dash) (3.21
Requirement already satisfied: jupyter-core>=4.6.0 in /usr/local/lib/python3.10/dist-packages (from jupyter-client->ipykernel->jupyter-d
Requirement already satisfied: pyzmq>=13 in /usr/local/lib/python3.10/dist-packages (from jupyter-client->ipykernel->jupyter-dash) (24.0
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.10/dist-packages (from jupyter-client->ipykernel->jupyter-
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.10/dist-packages (from jupyter-core>=4.6.0->jupyter-client->i
```

Step 1: Code for Animated Bar Chart python Copy code

```
import plotly.express as px
import pandas as pd

# Load the dataset
df = pd.read_csv('sample_data/recent-grads.csv')

# Prepare the data for animation using a logical frame sequence (e.g., by rank or earnings)
df['Animation_Frame'] = df['Major_category']  # Replace with a relevant column if needed

# Create an animated bar chart for narrative storytelling
fig_bar = px.bar(
    df,
    x='Median',
    y='Major',
    color='Major_category',
    animation_frame='Animation_Frame',
    orientation='h',
    title='Median Earnings by Major (Animated Narrative)',
    labels={'Median': 'Median Earnings ($)', 'Major': 'Major'},
    text='Median'  # Display median earnings on bars
)

# Adjust animation settings for better pacing
fig_bar.layout.updatemenus[0].buttons[0].args[1]['frame']['duration'] = 2500  # Slow down to 2.5 seconds per frame

# Add annotations for key insights (example for a single frame)
fig_bar.add_annotation(
    x=80000,  # Example coordinate
    y='Engineering',  # Example major
    text="Highest earning major category",
    showarrow=True,
    arrowhead=2
)

# Customize layout for storytelling
fig_bar.update_layout(
    xaxis_title='Median Earnings ($)',
    yaxis_title='Major',
    annotations=[  # Add custom annotations if needed
        dict(
            x=70000, y='Math & Statistics', text="High earning but fewer graduates",
            showarrow=True, arrowhead=2, ax=-50, ay=0
        )
    ]
)
```

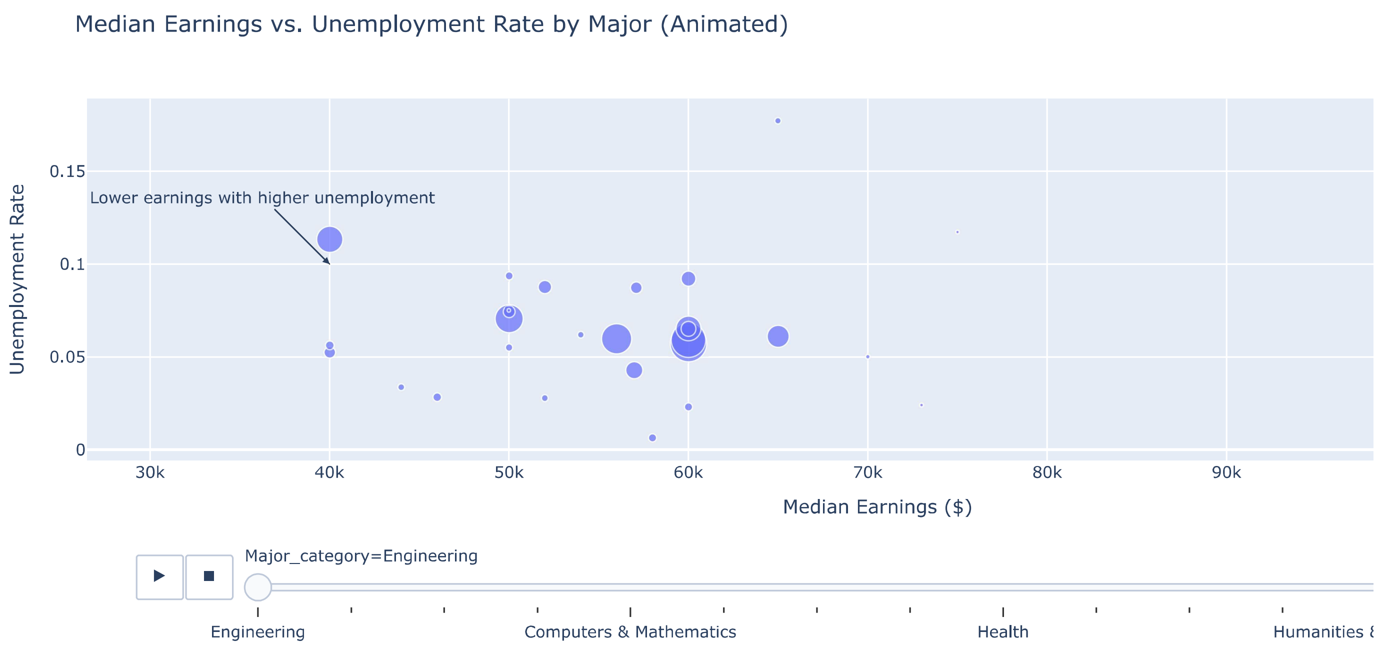
```
# Show the figure with narrative features
fig_bar.show()
```



code for interacctive scatter plt

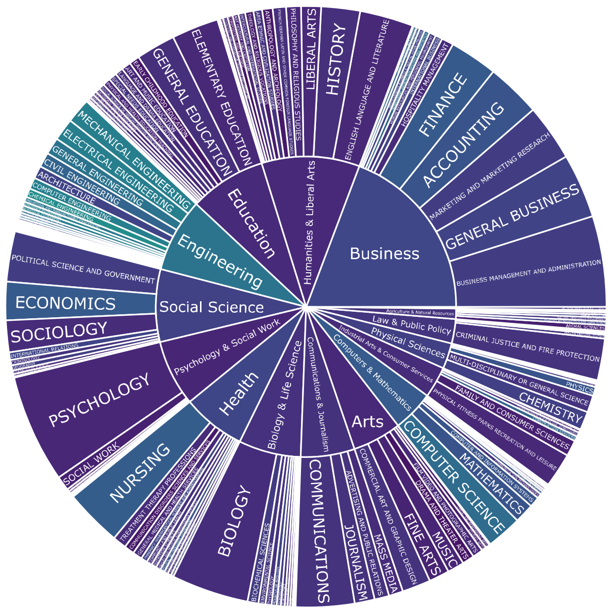
```
import plotly.express as px
import pandas as pd

# Load the dataset
df = pd.read_csv('sample_data/recent-grads.csv')

# Handle missing values in columns used for size or any other relevant field
df['Total'].fillna(df['Total'].median(), inplace=True)

# Create an animated scatter plot for median earnings vs. unemployment rate
fig_scatter = px.scatter(
    df,
    x='Median',
    y='Unemployment_rate',
    size='Total',
    color='Major_category',
    animation_frame='Major_category',  # Adjust or use another column for the sequence
    hover_name='Major',
    title='Median Earnings vs. Unemployment Rate by Major (Animated)',
    labels={'Median': 'Median Earnings ($)', 'Unemployment_rate': 'Unemployment Rate'},
    size_max=40  # Adjust size for better visualization
)

# Adjust animation settings for a slower and clearer pace
fig_scatter.layout.updatemenus[0].buttons[0].args[1]['frame']['duration'] = 2500  # Slow down to 2.5 seconds per frame

# Add annotations to guide storytelling
fig_scatter.add_annotation(
    x=80000,  # Example coordinate for annotation
    y=0.05,   # Example unemployment rate coordinate
    text="High-earning majors with low unemployment",
    showarrow=True,
    arrowhead=2
)

# Update layout for enhanced storytelling and clarity
fig_scatter.update_layout(
    xaxis_title='Median Earnings ($)',
    yaxis_title='Unemployment Rate',
    annotations=[  # Add multiple annotations if needed
        dict(
            x=40000, y=0.10, text="Lower earnings with higher unemployment",
            showarrow=True, arrowhead=2, ax=-50, ay=-50
        )
    ]
)

# Show the animated scatter plot
fig_scatter.show()
```

```
<ipython-input-10-919d78e82a1a>:8: FutureWarning:

A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting valu

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].me
```



```
import plotly.express as px

# Clean the data
sunburst_data=pd.read_csv('sample_data/recent-grads.csv')
cleaned_data = sunburst_data[sunburst_data['Total'] > 0].dropna(subset=['Median', 'Total'])

# Create the Sunburst Chart
sunburst_fig = px.sunburst(
    cleaned_data,
    path=['Major_category', 'Major'],
    values='Total',
    color='Median',
    color_continuous_scale='Viridis',
    title='Economic Outcomes of College Majors',
    hover_data={'Median': True, 'ShareWomen': True}
)

# Update layout for better presentation
sunburst_fig.update_layout(
    margin=dict(t=50, l=25, r=25, b=25),
    coloraxis_colorbar=dict(title="Median Earnings ($)"),
    title_font_size=16
)

# Display the chart
sunburst_fig.show()
```

Economic Outcomes of College Majors

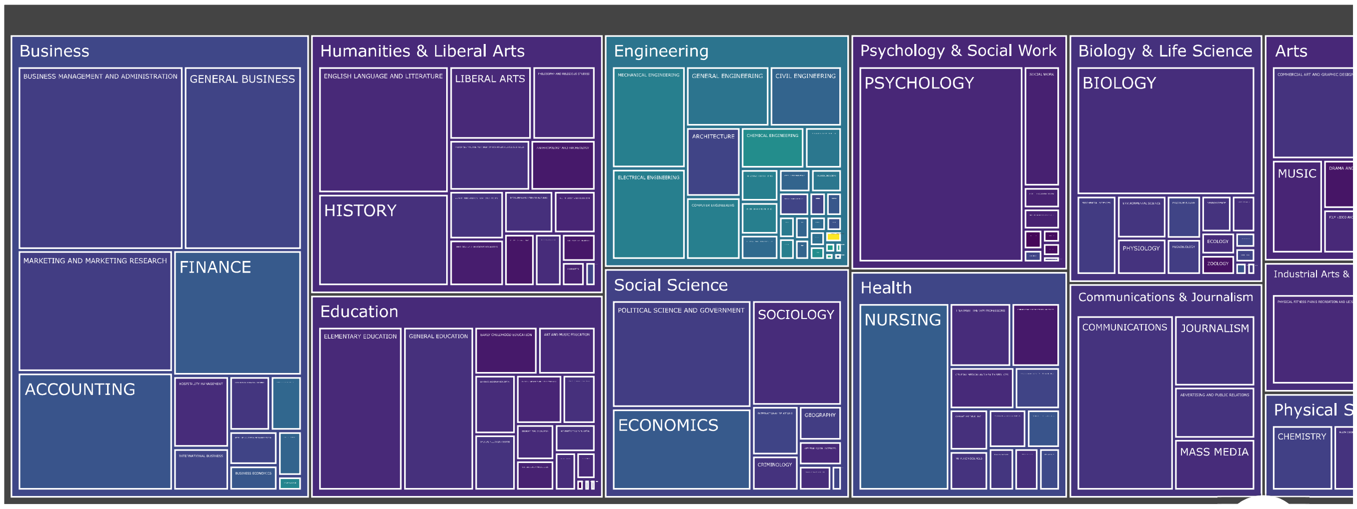


```
import pandas as pd
import dash
from dash import dcc, html, Input, Output
import plotly.express as px

# Load the dataset
file_path = 'recent-grads.csv'
data = pd.read_csv(file_path)

# Filter out rows with zero or missing values in 'Total'
data = data[data['Total'] > 0].dropna(subset=['Total', 'Median'])

# Prepare the dataset for the Treemap
treemap_data = data.groupby(['Major_category', 'Major']).agg({
    'Total': 'sum',
    'Median': 'mean'
}).reset_index()

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

# Layout of the app
app.layout = html.Div([
    html.H1("Treemap of College Majors: Total Graduates vs. Median Earnings"),
    dcc.Dropdown(
        id='major-category-dropdown',
        options=[{'label': category, 'value': category} for category in treemap_data['Major_category'].unique()],
        placeholder="Select a Major Category",
        value=None,  # Default value (None shows all categories)
        multi=False
    ),
    dcc.Graph(id='treemap-chart')
])

# Callback to update the Treemap based on dropdown selection
@app.callback(
    Output('treemap-chart', 'figure'),
    [Input('major-category-dropdown', 'value')]
)
def update_treemap(selected_category):
    # Filter the data based on the selected Major Category
    filtered_data = treemap_data[treemap_data['Major_category'] == selected_category] if selected_category else treemap_data

    # Create the Treemap
    fig = px.treemap(
        filtered_data,
        path=['Major_category', 'Major'],  # Hierarchy levels
        values='Total',                 # Size of each block
        color='Median',                 # Color based on median earnings
        color_continuous_scale='Viridis', # Color scale
        title=f"Treemap: {selected_category if selected_category else 'All Majors'}",
        hover_data={'Median': True}       # Additional hover info
    )

    # Update layout for clarity
    fig.update_layout(
        margin=dict(t=50, l=25, r=25, b=25),
        coloraxis_colorbar=dict(title="Median Earnings ($)")
    )

    return fig

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

Treemap of College Majors: Total Graduates vs. Median Earnings

Select a Major Category

Treemap: All Majors



Double-click (or enter) to edit