quake-data-visualization

November 21, 2023

[ ]:
```
import pandas as pd
import altair as alt
import plotly.express as px
```

[ ]:
```
df = pd.read_csv('mc1-reports-data.csv')
select_year = alt.selection_point(
    name="Location",
    fields=["location"],
    bind=alt.binding_range(min=1, max=19, step=1, name="Select Location")
)

alt.Chart(df).mark_bar(size=35).encode(
    alt.X("power:Q", bin=True).title('Reported Power Score'),
    y= 'count()',
).properties(
    width=500,
    title="Distribution of Power Scores by Location"
).add_params(
    select_year
).transform_filter(
    select_year
).configure_facet(
    spacing=8
)
```

[ ]:
```
alt.Chart(…)
```

[ ]:
```
df2 = df[(df['location'] == 3) | (df['location'] == 10) | (df['location'] ==␣
 ↪11)]
fig = px.histogram(df2, x="time",
                   color = "location",
                   hover_data= 'location',
                   title="Distribution of Reports Recieved for Locations 3, 10,␣
 ↪and 11",
                   width = 600)
fig.show()
```

[ ]:
```
df3 = df[['location','medical', 'roads_and_bridges', 'buildings', 'power',␣
↪'sewer_and_water']]
df3 = df3[(df['location'] == 3) | (df['location'] == 1) | (df['location'] == 9)]
df3 = df3.set_index('location')

fig = px.imshow(df3, width = 400, title = 'Damage scores by location (1, 3, or␣
↪9)')
fig.show()
```

[ ]:
```
options=['sewer_and_water','power','roads_and_bridges','medical','buildings']
labels=[option + ' ' for option in options]

x_dropdown = alt.binding_radio(
    options=options,
    labels=labels,
    name='x Measure: '
)
x_measure_param = alt.param(
    value='sewer_and_water',
    bind=x_dropdown
)

y_dropdown = alt.binding_radio(
    options=options,
    labels=labels,
    name='y Measure: '
)
y_measure_param = alt.param(
    value='power',
    bind=y_dropdown
)



alt.Chart(df).mark_rect().encode(
    alt.X('x:Q', bin=alt.Bin(maxbins=11)),
    alt.Y('y:Q', bin=alt.Bin(maxbins=11)),
    color='count()',
    tooltip=['sewer_and_water','power','count()']
).properties(
    title='Comparison of Measure Scores'
).add_params(
    x_measure_param,
    y_measure_param
).transform_calculate(
    x=f'datum[{x_measure_param.name}]',
    y=f'datum[{y_measure_param.name}]'
)
```

[ ]:
```
alt.Chart(…)
```

[ ]:
```
options=['sewer_and_water','power','roads_and_bridges','medical','buildings']
labels=[option + ' ' for option in options]

input_dropdown = alt.binding_radio(
    options=options,
    labels=labels,
    name='Measure: '
)
measure_param = alt.param(
    value='sewer_and_water',
    bind=input_dropdown
)

alt.Chart(df).mark_bar().encode(
    x='location:O',
    y=alt.Y('mean(y):Q', scale=alt.Scale(domain=[0,10])).title('mean score of␣
 ↪measure'),
    tooltip=['location','mean(y):Q']
).transform_calculate(
    y=f'datum[{measure_param.name}]'
).add_params(
    measure_param
).properties(
    title='Mean Measure Score by Location'
)
```

[ ]:
```
alt.Chart(…)
```

[ ]:
```
report_counts = df.groupby(['location', df['time']]).size().
 ↪reset_index(name='report_count')
report_counts['location'] = report_counts['location'].astype(str)
fig = px.scatter(report_counts, x='time', y='report_count', color='location',
                 labels={
                      "location": "Location",
                      "time": "Date and Time",
                      "report_count": "Number of Reports"},
                 width=1000, height=600)
fig.update_layout(
    title='Total Number of Reports by Neighborhood Over Time',
    xaxis_title='Time',
    yaxis_title='# Reports',
    legend_title='Location')
fig.update_xaxes(tickformat = '%m/%d')

fig.show()
```