I am a Master's student pursuing Data Science and Communication, For my Dissertation, I am researching a question "A Comparative Analysis of Data Visualisations generated by LLMs over Manual visualisations in representation Election Data and by evaluating based on accuracy, Applications and Ethical considerations"

So I will be testing you on a use case provided below

" Usecase 2 The objective is to plot a hex map (referenced from https://commonslibrary.parliament.uk/research-briefings/cbp-10009/) to show the Results of the UK 2024 elections w.r.t Country. "

What is your understanding of Hex Maps as Data Visualisation and its best practices in the field of representing election data

A **hex map** (or hexagonal bin map) is a data visualization technique that represents geographic areas using hexagons instead of traditional geographical boundaries. It is especially useful for election data because it equalizes the size of each geographic unit (e.g., constituencies or regions), overcoming the visual distortion caused by geographical size differences. In the case of UK election data, where constituencies vary greatly in size, hex maps can help better represent voting outcomes without giving undue visual weight to larger rural areas over more densely populated urban ones.

Key Characteristics of Hex Maps:

1. **Uniform Shape**: Hexagons provide a consistent, regular shape for each area, ensuring that no region appears larger or smaller than others simply because of geography.
2. **Reduced Spatial Distortion**: Traditional maps may skew perception, especially in regions where geography varies greatly in area size. Hex maps create a fairer visual representation, especially for interpreting election results.
3. **Intuitive Grouping**: Hexagons allow for clear adjacency between regions, showing relationships between neighboring constituencies, which can be useful when mapping political trends.
4. **Simplified Interpretation**: By eliminating unnecessary geographical detail, hex maps focus the viewer's attention on the key data — in this case, election outcomes.

Best Practices for Election Data Representation with Hex Maps:

1. **Color Coding for Results**: Use distinct and contrasting colors to represent different political parties or results. For instance, each party can have its own color, and constituencies can be shaded based on the winning party.
2. **Consistent Data Scale**: Ensure that the color intensity or patterns used to display election results are consistent across the map. For example, showing voting percentages or vote margins can be done with varying shades of a party's color.
3. **Legends and Labels**: Provide clear legends to explain the color scheme, along with constituency names or codes (e.g., in a hover-over tooltip in an interactive version) for ease of interpretation.
4. **Handling Close Races**: If the map is used to show election results with small vote margins, highlighting constituencies with tight races (e.g., by adding borders or specific patterns) can improve clarity.
5. **Interactive Features**: For detailed and in-depth analysis, an interactive hex map can allow users to click on each constituency to see additional information like vote counts, demographic data, or historical trends.

Ethical Considerations in Election Data Visualizations:

- **Transparency and Accuracy**: The map should accurately represent the election data and not be manipulated to favor any political outcome.

- **Clarity in Representation**: Simplifying geographic boundaries must not come at the cost of obscuring important details. Constituencies should remain identifiable.
- **Neutral Color Choices**: Careful selection of colors should be neutral and not evoke bias (e.g., avoid assigning colors associated with negativity to specific parties).
- **Inclusivity in Design**: Design choices should account for accessibility, ensuring that people with color vision deficiencies can interpret the map (e.g., using colorblind-friendly palettes).

In the context of your research, you can compare how hex maps generated by LLMs (via prompts) may handle these considerations versus manually created maps by human experts. The comparison would involve assessing differences in clarity, effectiveness in showing trends, and any ethical implications in the color choices or data representation.

HoC-GE2024-results-by-constituency....
Spreadsheet

According to your understanding of Hex Maps as Data Visualisation and its best practices in the field of representing election data

Using the attached CSV file, Plot a hex map

To create a hex map of the UK 2024 election results based on the CSV file you provided, I will first load and analyze the data. Then, I will use a hex map grid representing UK constituencies to visualize the election results.

Let me start by inspecting the structure of the file and checking for the necessary columns.

Analyzed

python Always show details Copy code

```python
import pandas as pd # Load the CSV file to inspect its contents file_path =
'/mnt/data/HoC-GE2024-results-by-constituency.csv' election_data =
pd.read_csv(file_path) # Display the first few rows and column names to understand the
data election_data.head(), election_data.columns
```

Result

```
(      ONS ID ONS region ID                   Constituency name  County n
 0  W07000081     W92000004                    Aberafan Maesteg
 1  S14000060     S92000003                      Aberdeen North
 2  S14000061     S92000003                      Aberdeen South
 3  S14000062     S92000003  Aberdeenshire North and Moray East
 4  S14000063     S92000003                  Airdrie and Shotts

   Region name Country name Constituency type  Declaration time  \
 0       Wales        Wales            County               NaN
 1    Scotland     Scotland             Burgh               NaN
 2    Scotland     Scotland             Burgh               NaN
 3    Scotland     Scotland            County               NaN
 4    Scotland     Scotland            County               NaN

   Member first name Member surname  ... Green    SNP    PC DUP  SF  SDLP
 0           Stephen        Kinnock  ...  1094      0  4719   0   0     0
 1            Kirsty       Blackman  ...  1275  14533     0   0   0     0
 2           Stephen          Flynn  ...  1609  15213     0   0   0     0
 3            Seamus          Logan  ...     0  13455     0   0   0     0
 4           Kenneth      Stevenson  ...     0  11324     0   0   0     0

    APNI  All other candidates  Of which other winner
 0     0                   801                      0
 1     0                  1269                      0
2     0                   648                        0
3     0                     0                        0
4     0                  1079                        0

[5 rows x 32 columns],
Index(['ONS ID', 'ONS region ID', 'Constituency name', 'County name',
       'Region name', 'Country name', 'Constituency type', 'Declaration
       'Member first name', 'Member surname', 'Member gender', 'Result',
       'First party', 'Second party', 'Electorate', 'Valid votes',
       'Invalid votes', 'Majority', 'Con', 'Lab', 'LD', 'RUK', 'Green',
       'PC', 'DUP', 'SF', 'SDLP', 'UUP', 'APNI', 'All other candidates',
       'Of which other winner'],
      dtype='object'))
```

The dataset includes various details about the election results by constituency. Key columns relevant to visualizing the election results are:

- **Constituency name**: The name of the constituency.
- **Country name**: The country the constituency belongs to (e.g., Scotland, England, Wales).
- **First party**: The winning party in each constituency.
- **Result**: The election result status.

To plot a hex map, I need to map each constituency to a predefined grid of hexagons. First, I will match the constituencies to a pre-established UK hex grid (which assumes that each constituency has a corresponding hexagon). Then, I will use color-coding to show which party won each constituency.

Let me first check if there is a hex grid that we can use to plot this.

Analysis errored

```python
# Let's check for a pre-established hex grid for UK constituencies. # This step
typically requires loading or referencing a pre-existing hex grid layout. # For now, I
will mock a basic structure where I would link constituency names # to a hex grid
structure and plot them. Let's continue by processing the election data. # Aggregating
by Country name and First party to get a basic overview election_summary =
election_data.groupby(['Country name', 'First party']).size().unstack(fill_value=0)
election_summary
```

It seems like I can't do more advanced data analysis right now. Please try again later.

If you want, I can guide you on how to create the hex map visualization using Python or any other tool manually. Let me know how you'd like to proceed!

Proceed

To create a hex map visualization of the UK election results from your dataset, you can use Python libraries like `matplotlib`, `plotly`, and `geopandas`. Below is a step-by-step guide to create this visualization:

Step 1: Install Required Libraries

Ensure you have the following Python libraries installed:

```bash
pip install pandas geopandas matplotlib plotly
```

Step 2: Prepare Your Hex Map Grid

To create a hex map for UK constituencies, you'll need a pre-defined hex grid. A commonly used grid for UK constituencies is available in formats like GeoJSON. You can find or download it from repositories or create one manually.

For this tutorial, we'll assume you already have a hex grid with corresponding constituency names.

Step 3: Load Your Data and Hex Grid

```python
import pandas as pd import geopandas as gpd import matplotlib.pyplot as plt # Load your
election data (replace 'path_to_election_data.csv' with your actual file path)
election_data = pd.read_csv('path_to_election_data.csv') # Load a pre-defined UK hex
grid (GeoJSON or Shapefile) hex_grid = gpd.read_file('path_to_hex_grid.geojson') #
GeoJSON of UK constituencies # Merge election data with the hex grid hex_map_data =
hex_grid.merge(election_data, left_on='constituency_name', right_on='Constituency
name')
```

Step 4: Plot the Hex Map

```python
# Create a color dictionary for political parties party_colors = { 'Conservative':
'blue', 'Labour': 'red', 'Liberal Democrat': 'orange', 'Green': 'green', 'SNP':
'yellow', 'Plaid Cymru': 'darkgreen', 'DUP': 'purple', 'Sinn Féin': 'darkred' } # Map
colors to each constituency hex_map_data['color'] = hex_map_data['First
party'].map(party_colors) # Plot the hex map fig, ax = plt.subplots(1, 1, figsize=(10,
10)) hex_map_data.plot(ax=ax, color=hex_map_data['color'], edgecolor='black') # Add a
title and legend plt.title('UK 2024 Election Results by Constituency (Hex Map)',
fontsize=15) plt.show()
```

Step 5: Customizing the Map

- **Tooltip**: Add tooltips or hover-over information using libraries like `plotly`.
- **Country Boundaries**: You can outline the country boundaries by using `geopandas` to draw polygons or by adding custom layers to the plot.

Step 6: Adding Interactivity (Optional)

If you'd like to create an interactive hex map using `plotly`, here's how you can adapt the visualization:

```python
import plotly.express as px # Create a plotly hex map fig = px.choropleth(hex_map_data,
geojson=hex_grid, locations='constituency_name', color='First party',
hover_name='Constituency name', color_discrete_map=party_colors, title="UK 2024 Election
Results") fig.show()
```

This code provides a basic framework for creating a hex map of the UK 2024 election results. Adjust the paths and tweak the appearance depending on your specific hex grid and data.

```
KeyError                                  Traceback (most recent call last)
<ipython-input-6-038c1f505836> in <cell line: 39>()
    37
    38 # Merge election data with the hex grid
---> 39 hex_map_data = hex_grid.merge(election_data,
left_on='constituency_name', right_on='Constituency name')
    40 # Create a color dictionary for political parties
    41 party_colors = {

5 frames
```

```
/usr/local/lib/python3.10/dist-packages/pandas/core/generic.py in
_get_label_or_level_values(self, key, axis)
   1842            values = self.axes[axis].get_level_values(key)._values
   1843        else:
-> 1844            raise KeyError(key)
   1845
   1846        # Check for duplicates

KeyError: 'constituency_name'
```

Resolve

The error occurs because the column name `'constituency_name'` doesn't exist in your `hex_grid` data. You can resolve this by ensuring that the column name in the `hex_grid` correctly matches the column in your election data, which is `'Constituency name'`.

Steps to resolve the error:

1. **Check the column names in the `hex_grid` data** to find the correct column that represents constituencies. You can use:

   ```python
   print(hex_grid.columns)
   ```

2. **Update the merge statement** with the correct column from `hex_grid`. For example:

   ```python
   hex_map_data = hex_grid.merge(election_data, left_on='correct_column_name',
   right_on='Constituency name')
   ```

Replace `'correct_column_name'` with the actual column name from the `hex_grid` data that corresponds to constituency names.

there is an issue with code kindly regenerate

Here's a cleaned-up and more general version of the code to help resolve any issues you might encounter when merging your election data with the hex grid. I'll break it down more carefully to ensure everything runs smoothly.

Step-by-Step Guide

1. **Ensure the hex grid contains a matching column**:
   - The `hex_grid` should have a column that corresponds to the names of constituencies in your election data. The election data uses `'Constituency name'`.
2. **General Workflow**:
   - Load the election data and hex grid.
   - Verify the columns in both datasets to ensure matching columns for merging.
   - Create the hex map.

Updated Code:

```python
import pandas as pd import geopandas as gpd import matplotlib.pyplot as plt # Load your
election data election_data = pd.read_csv('path_to_election_data.csv') # Load the UK hex
grid (replace 'path_to_hex_grid' with your file's path) hex_grid =
gpd.read_file('path_to_hex_grid.geojson') # Assuming the grid is in GeoJSON format #
```

```
Check the columns of the hex grid to identify the correct constituency name column
print(hex_grid.columns) # Assuming the hex grid has a column named 'constituency_name'
or similar, # Replace 'constituency_name' with the correct column if necessary #
Example: if hex grid has 'constituency', use that in place of 'constituency_name' #
Merge election data with the hex grid hex_map_data = hex_grid.merge(election_data,
left_on='constituency_name', right_on='Constituency name') # Create a color dictionary
for political parties party_colors = { 'Conservative': 'blue', 'Labour': 'red', 'Liberal
Democrat': 'orange', 'Green': 'green', 'SNP': 'yellow', 'Plaid Cymru': 'darkgreen',
'DUP': 'purple', 'Sinn Féin': 'darkred' } # Map colors to each constituency
hex_map_data['color'] = hex_map_data['First party'].map(party_colors) # Plot the hex map
fig, ax = plt.subplots(1, 1, figsize=(10, 10)) hex_map_data.plot(ax=ax,
color=hex_map_data['color'], edgecolor='black') # Add a title and show the plot
plt.title('UK 2024 Election Results by Constituency (Hex Map)', fontsize=15) plt.show()
```

Key Considerations:

1. **Check column names**: Make sure to replace `'constituency_name'` in the `hex_grid` with the actual column name from the grid file. Use `print(hex_grid.columns)` to confirm.
2. **Matching constituency names**: Ensure that the names in both the hex grid and election data are consistent (case-sensitive, spaces, etc.). If there are any discrepancies (like abbreviations), you may need to clean up the data using `str.lower()` or `str.strip()`.

Let me know if this resolves the issue!

```
Index(['FID', 'PCON24CD', 'PCON24NM', 'PCON24NMW', 'BNG_E',
'BNG_N', 'LAT',
      'LONG', 'GlobalID', 'geometry'],
     dtype='object')
```

Column is this so edit columns accordingly

Thank you for sharing the column names from the hex grid. Based on these column names, the most likely matching column for constituency names is `'PCON24NM'`, which appears to represent the constituency names for the 2024 election.

I'll update the merge statement to reflect this column.

Updated Code:

```python
import pandas as pd import geopandas as gpd import matplotlib.pyplot as plt # Load your
election data election_data = pd.read_csv('path_to_election_data.csv') # Load the UK hex
grid (replace 'path_to_hex_grid' with your file's path) hex_grid =
gpd.read_file('path_to_hex_grid.geojson') # Assuming the grid is in GeoJSON format #
Check the columns of the hex grid print(hex_grid.columns) # Merge election data with the
hex grid # The 'PCON24NM' column represents constituency names in the hex grid
hex_map_data = hex_grid.merge(election_data, left_on='PCON24NM', right_on='Constituency
name') # Create a color dictionary for political parties party_colors = {
'Conservative': 'blue', 'Labour': 'red', 'Liberal Democrat': 'orange', 'Green': 'green',
'SNP': 'yellow', 'Plaid Cymru': 'darkgreen', 'DUP': 'purple', 'Sinn Féin': 'darkred' } #
Map colors to each constituency based on the winning party hex_map_data['color'] =
hex_map_data['First party'].map(party_colors) # Plot the hex map fig, ax =
plt.subplots(1, 1, figsize=(10, 10)) hex_map_data.plot(ax=ax,
color=hex_map_data['color'], edgecolor='black') # Add a title and show the plot
plt.title('UK 2024 Election Results by Constituency (Hex Map)', fontsize=15) plt.show()
```

Key Adjustments:

- The hex grid column for constituency names is now `'PCON24NM'` (used in the `merge` statement).
- The election data column remains `'Constituency name'` from the CSV file.

This should resolve the issue. Let me know if it works!