DS4200 Assignment 5

March 17, 2023

1 DS4200 Assignment 5

Anjali Tanna ***

1.0.1 Part 1

Dataset of choosing: `nbadata.csv`

1.0.2 Part 2

Dataset explored and cleaned in Jupyter Notebooks as shown below.

[1]:
```python
# Imports
import pandas as pd
import numpy as np
from datetime import datetime
from vega_datasets import data
import altair as alt
from altair import pipe, limit_rows, to_values
```

CITATION - To solve the error of the dataset being too big: source

[2]:
```python
# Fix issue of dataset being too big
t = lambda data: pipe(data, limit_rows(max_rows=100000), to_values)
alt.data_transformers.register('custom', t)
alt.data_transformers.enable('custom')
```

[2]:
```
DataTransformerRegistry.enable('custom')
```

[3]:
```python
# Read nbadata.csv file
nba = pd.read_csv("nbadata.csv")
```

[4]:
```python
# Drop NaN rows
nba = nba[nba['Player'].notna()]
```

[5]:
```
# Change 'Year' column from float to datetime
nba['Season'] = pd.to_datetime(nba['Season'], format='%Y')
```

[6]:
```
# Display nba DataFrame (head)
nba.head()
```

[6]:
```
      Season         Player Position   Age Team     G    GS      MP     FG  \
0 2014-01-01    A.J. Price       SG  27.0  MIN  28.0   0.0    99.0   19.0
1 2014-01-01  Aaron Brooks       PG  29.0  TOT  72.0  12.0  1557.0  233.0
2 2014-01-01  Aaron Brooks       PG  29.0  HOU  43.0   0.0   716.0  104.0
3 2014-01-01  Aaron Brooks       PG  29.0  DEN  29.0  12.0   841.0  129.0
4 2014-01-01    Aaron Gray        C  29.0  TOT  37.0   6.0   355.0   27.0

     FGA  …    FT%   ORB   DRB    TRB    AST   STL   BLK    TOV     PF  \
0   46.0  …  0.000   1.0   9.0   10.0   13.0   1.0   0.0    7.0    5.0
1  581.0  …  0.874  43.0  97.0  140.0  233.0  52.0  13.0  117.0  146.0
2  263.0  …  0.841  23.0  39.0   62.0   83.0  25.0   6.0   51.0   76.0
3  318.0  …  0.902  20.0  58.0   78.0  150.0  27.0   7.0   66.0   70.0
4   61.0  …  0.550  42.0  69.0  111.0   22.0  10.0   8.0   31.0   64.0

     PTS
0   44.0
1  645.0
2  299.0
3  346.0
4   65.0

[5 rows x 29 columns]
```

1.0.3 Part 3

Jupyter Notebook for Altair visualizations created.

1.0.4 Part 4

Dataset Source: NBA Dataset found on Basketball-Reference.com

The reasoning behind my choice of this dataset has to do with my passion for basketball and the NBA. I love watching basketball games as well as reading all of the game statistics right after. Seeing how quickly an NBA player's stats can fluctuate and how the top leaders change daily is something that is super interesting to me. I would like to compare different stats amongst teams and players through the past couple of seasons to see who the top players are in the NBA. These types of visualizations and analysis would provide helpful insight for game managers to see whether players are doing the best in order to increase their contract or figuring out which players are playing the worst and deserve a trade.

This dataset was found on basketball-reference.com (link provided above) and it includes data from the 2014-2022 seasons of the NBA. It was interesting to see the change in trends for points per game depending on the year, leading scorers, and other basketball statistics. I also found it interesting to see the difference in stat trends as time goes on, since the game of basketball is constantly changing. The skill levels of NBA players and athleticism is constantly increasing as time goes on and by exploring this dataset, I am able to visually showcase this. It is also interesting to see how the players are getting traded, as they have statistics for multiple teams. Overall, using NBA data allows me to make complex and interesting visualizations.

1.0.5 Part 5

CITATION - The code for the following 3 visualizations adapted from Vega-Altair

1.0.6 STREAMGRAPH

This streamgraph is displaying the sum of points per age group based on the season. The marks used are lines and the channels used are area and color. The color indicates the age that is being described. The x-axis indicates the season and the y-axis shows the sum of points for that age category. There is also an interactive element as there is hover data that displays the age, sum of points, and the season. The reason I chose to use color in this graph was because it is the only efficient way to show the differences in the ages.

[11]:
```python
# Streamgraph comparing sum of points per year per each age of NBA players

# Create interactive streamgraph
nba_stream = alt.Chart(nba).mark_area().encode(
    alt.X('Season:T',
        axis=alt.Axis(format='%Y', domain=False, tickSize=0)
    ),
    alt.Y('sum(PTS):Q', stack='center'),
    alt.Color('Age:Q',
        scale=alt.Scale(scheme='category20b')
    ), tooltip=['Age', 'sum(PTS)', alt.Tooltip('Season:T', format='%Y')]
).properties(title='Sum of PTS per Age based on Season').interactive()

nba_stream
```

[11]:
```
alt.Chart(...)
```

1.0.7 BOX AND WHISKER PLOT

This box and whisker plot showcases the breakdown of ages across the NBA teams. The box and whisker plot shows the locality, spread, and skew of ages across NBA teams. It is interesting to see which teams have older ages compared to those with younger teams. It is also interesting to see the outliers, as they are all generally older players. The marks used in this visualization are points and lines. The channels used are area, position, and color. I chose to incorporate color into this visualization just for aesthetic purposes.

[8]:
```
# Box and whisker plot for teams and their player's ages

# Select relevant columns for the plot
age_df = nba[['Team', 'Age', 'Player', 'Season']]

# Create an interactive boxplot
nba_box = alt.Chart(age_df).mark_boxplot().encode(
    x='Team:N',
    y='Age:Q',
    tooltip=['Team', 'Age', 'Player', alt.Tooltip('Season:T', format='%Y')],
    color='Team:N'
).properties(title='Ages of NBA Teams').interactive()

nba_box
```

```
/Users/anjalitanna/opt/miniconda3/lib/python3.8/site-
packages/altair/utils/data.py:226: AltairDeprecationWarning: alt.pipe() is
deprecated, and will be removed in a future release. Use toolz.curried.pipe()
instead.
  warnings.warn(
```

[8]:
```
alt.Chart(…)
```

1.0.8 BUBBLE PLOT

This visualization shows the sum of points per player based on the season, team, and position. This visualization utilizes two dropdown menus, one to select the team and one to select the position. The marks used in this visualization are points and the channels used are color and size. I used color to show the players field goal percentage. The darker the blue, the better the player's field goal percentage is. As for size, it represents the amount of total points scored by that player in that season. The bigger the circle, the more points scored.

[9]:
```
# Bubble Plot comparing sum of Points based on Season, Team, and Position

# Create lists for dropdowns
teams = nba['Team'].unique()
positions = nba['Position'].unique()

# A dropdown filter to select the NBA team
team_dropdown = alt.binding_select(options=teams, name='Team')
team_select = alt.selection_single(fields=['Team'], bind=team_dropdown)

# A dropdown filter to select the player position
pos_dropdown = alt.binding_select(options=positions, name='Position')
pos_select = alt.selection_single(fields=['Position'], bind=pos_dropdown)

# Create bubble plot
```

```
nba_bubble = alt.Chart(nba).mark_circle().encode(
    x='Season:T',
    y='Player:N',
    size='sum(PTS):Q',
    tooltip=['Player', 'Position', 'FG%', 'Age', 'sum(PTS)', 'Team'],
    color='FG%'
).add_selection(
    team_select, pos_select
).transform_filter(
    team_select
).transform_filter(
    pos_select
).properties(title='Sum of Points Based on Season, Team, and Position')


nba_bubble
```

```
[9]: alt.Chart(…)
```

CITATION - Code adapted from Interactive data viz using Altair

1.0.9 SCATTER PLOT AND BAR CHART

This brushing and linking scatter plot and bar chart shows the minutes played per player vs. the amount of points scored. When a specific interval is selected on the scatter plot, a corresponding bar chart will show on the bottom. This bar chart displays the sum of points per team based on the minutes played interval that is selected. The marks used on the scatter plot are points and the marks on the bar chart are lines. The channels used for both are position, length, and color. I used color to make it easier to see which player belongs to which team.

```
[10]: # Interactive Scatter Plot and Bar Chart showcasing Points Scored based on␣
       ↪Minutes Played

      # Create teams scale and color scheme
      teams = alt.Scale(domain=['ATL', 'CHI', 'PHI', 'BOS', 'DEN', 'MIL', 'DAL',␣
       ↪'PHX', 'MIN',
              'UTA', 'LAL', 'BKN', 'GSW', 'CHA', 'MEM', 'TOR', 'CLE', 'NYK',
              'SAS', 'OKC', 'NOP', 'SAC', 'MIA', 'DET', 'IND', 'LAC', 'HOU',
              'ORL', 'WAS', 'POR'],
                        range=['#FF4040', '#8B2323', '#0000FF', '#00C957', '#00BFFF',
                               '#006400', '#1874CD', '#9A32CD', '#104E8B', '#FFD700',
                               '#BF3EFF', '#1A1A1A', '#FFC125', '#48D1CC', '#191970',
                               '#EE2C2C', '#8B1A1A', '#FF7D40', '#2B2B2B', '#FF8C00',
                               '#8B6508', '#9400D3', '#DC143C', '#3D59AB', '#DC143D',
                               '#0000EE', '#FF3030', '#IC86EE', '#104E8B', '#B22222'])
      color = alt.Color('Team:N', scale=teams)

      # Brush on top panel
```

```python
# Multiclick on bottom panel
brush = alt.selection_interval(encodings=['x'])
click = alt.selection_multi(encodings=['color'])

# Scatter plot of MP vs. Points Scored
points = alt.Chart().mark_point().encode(
    alt.X('MP:Q', title='Minutes Played'),
    alt.Y('PTS:Q',
        title='Points Scored',
        scale=alt.Scale(domain=[0, 3000])
    ),
    color=alt.condition(brush, color, alt.value('lightgray')),
    size=alt.Size('FG%:Q', scale=alt.Scale(range=[0, 100])),
    tooltip=['Player', 'PTS', 'Team', 'Age']
).properties(
    width=550,
    height=300
).add_selection(
    brush
).transform_filter(
    click
)

# Bar chart at the bottom
bars = alt.Chart().mark_bar().encode(
    x='sum(PTS):Q',
    y='Team:N',
    color=alt.condition(click, color, alt.value('lightgray')),
).transform_filter(
    brush
).properties(
    width=550,
).add_selection(
    click
)

# Build Compound Plot
alt.vconcat(
    points,
    bars,
    data=nba,
    title='Minutes Played vs Points Scored of each NBA Player'
)
```

```
[10]: alt.VConcatChart(…)
```

1.0.10 Part 6 - Color

Streamgraph

While all of my visualization utilize color, the one that uses color in the most effective way to encode data is the streamgraph. The streamgraph uses color to distinguish the age of the NBA players. By using color, it is easy to see which age demographics scored the most amount of points for that season. I chose this type of color map because it makes it clear to see the differences amongst the different ages.

Box and Whisker Plot

The box and whisker plot also uses color to encode data. This was purely a stylistic choice, but the color channel shows what team is being displayed. There is also a label on top of the graph for each team.

Bubble Plot

The bubble plot also utilizes color to encode the data. I use color to show a player's field goal percentage while also showcasing the player's sum of points for that season in a size channel. The darker the blue, the higher a player's field goal percentage is. This helps with showcasing the shooting accuracy of a player. I used this to add another layer of data to the bubble plot. The hover data also encapsulates all features that are encoded through channels.

Scatter Plot and Bar Chart

Similar to the box and whisker plot, this visualization uses color encoding for mainly stylistic purposes. Colors were assigned to each NBA team based on their team colors. Using colors on the scatter plot helps distinguish each player and what team they are on.

1.0.11 Part 7 - Interactivity

Streamgraph

The interactivity on this graph includes zooming in and also hover data. You can scroll across the streamgraph to get a closer view. There is also hover data that provides more context to the data. It provides the age, the total points scored for that age group, as well as the season.

Box and Whisker Plot

The box and whisker plot also uses hover data to provide interactivity. The hover data provides the maximum age of the players on the team, the Q3, the median, the Q1, and the minimum age. This provides more context to the visualization as well as the exact values.

Bubble Plot

The bubble plot also utilizes hover data. It provides the player, position, FG%, age, sum of points, and team. This makes it easier to get more information on the player and their statistics without it being too cluttered. I also used two dropdown menus to select the position and team. This allows users to explore the different player positions within the team they select to see the breakdown of total points scored as well as field goal percentages. It also makes it so that the specific dataset you are looking at isn't too large and cluttered.

Scatter Plot and Bar Chart

The scatter plot and bar chart uses brushing and selection. You can select a portion on the scatter plot, and the bar chart on the bottom will reflect that selection. This interactivity breaks down the data further. The selection of the minutes played interval is then broken down on the bar chart, which shows the sum of points scored per team within that time interval. I used this feature to compare the minutes played per team with the total points scored.

1.0.12 Part 8

---

1. Brushing selections
   - I used brushing selections on the scatter plot to create a dynamic bar chart on the bottom. I used this so users would be able to look deeper into the data and break down the large dataset we are utilizing.
2. Dropdown menus
   - I used dropdown menus on the bubble plots to select the team and position of the player. This was utilized to also break down the large dataset but also compare the sum of points per player based on position. This can show what the highest scoring position is as well as comparing players within the same position on the same team to see who should be given more playing time.
3. Hover data
   - I used hover data on all of my graphs to provide a bit more context to my visualizations. It also provides the specific values that we are looking at on the graphs without the visualization looking too cluttered.
4. Scrolling
   - I used scrolling on the streamgraph. This allows users to zoom into the data as well as scroll through the seasons. This would be a better feature if I included more seasons, which is something I look to do in the future.