
DataCamp
TV, HALFTIME SHOWS, AND THE BIG GAME
JOVAN TRAJCESKI
PROJECT

Summary

DATA SETS

1. Game (Super Bowl) data
2. TV data
3. Halftime Musician data

QUESTIONS TO ANSWER

1. What are the most extreme game outcomes?
2. How does the game affect television viewership?
3. How have viewership, TV ratings, and ad cost evolved over time?
4. Who are the most prolific musicians in terms of halftime show performances?

TOOLS USED

Project Workflow

LOAD DATA

DATA EXPLORATION

CLEAN DATA

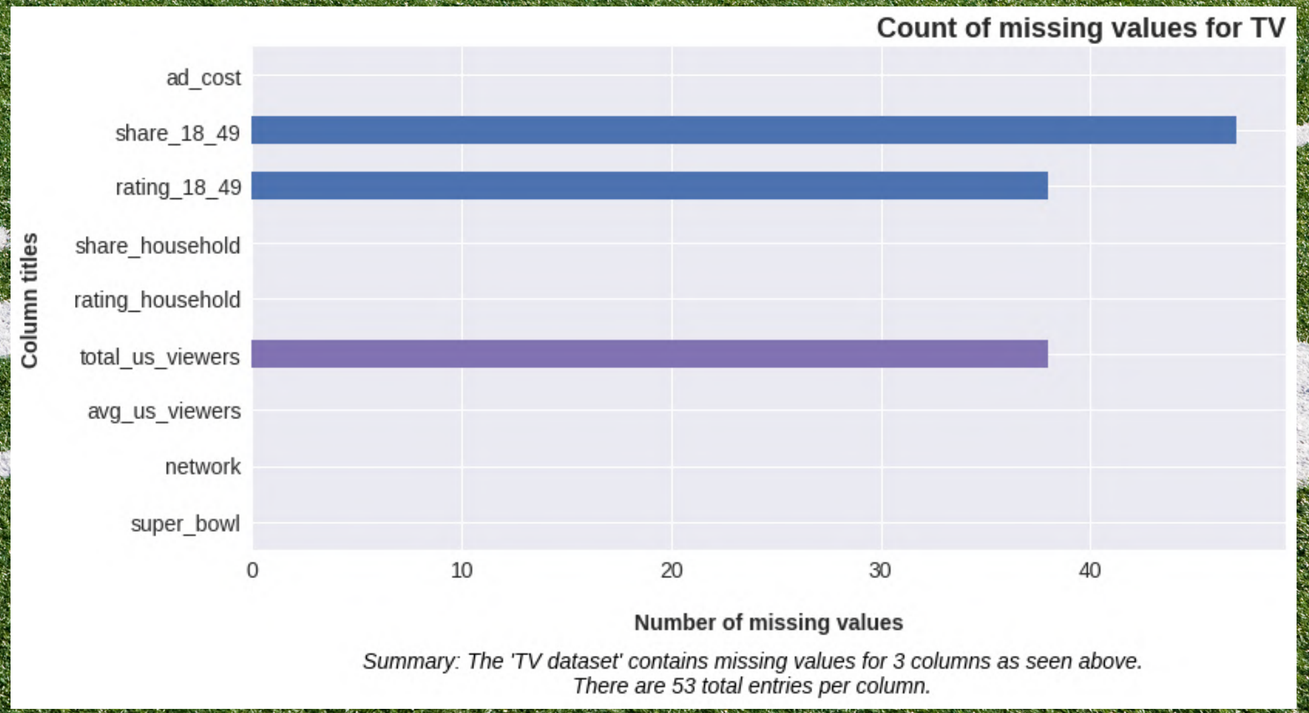
VISUALIZATION AND DISCOVERY

NOTE: All charts on next slides were created directly in Python (no manual intervention after the fact)

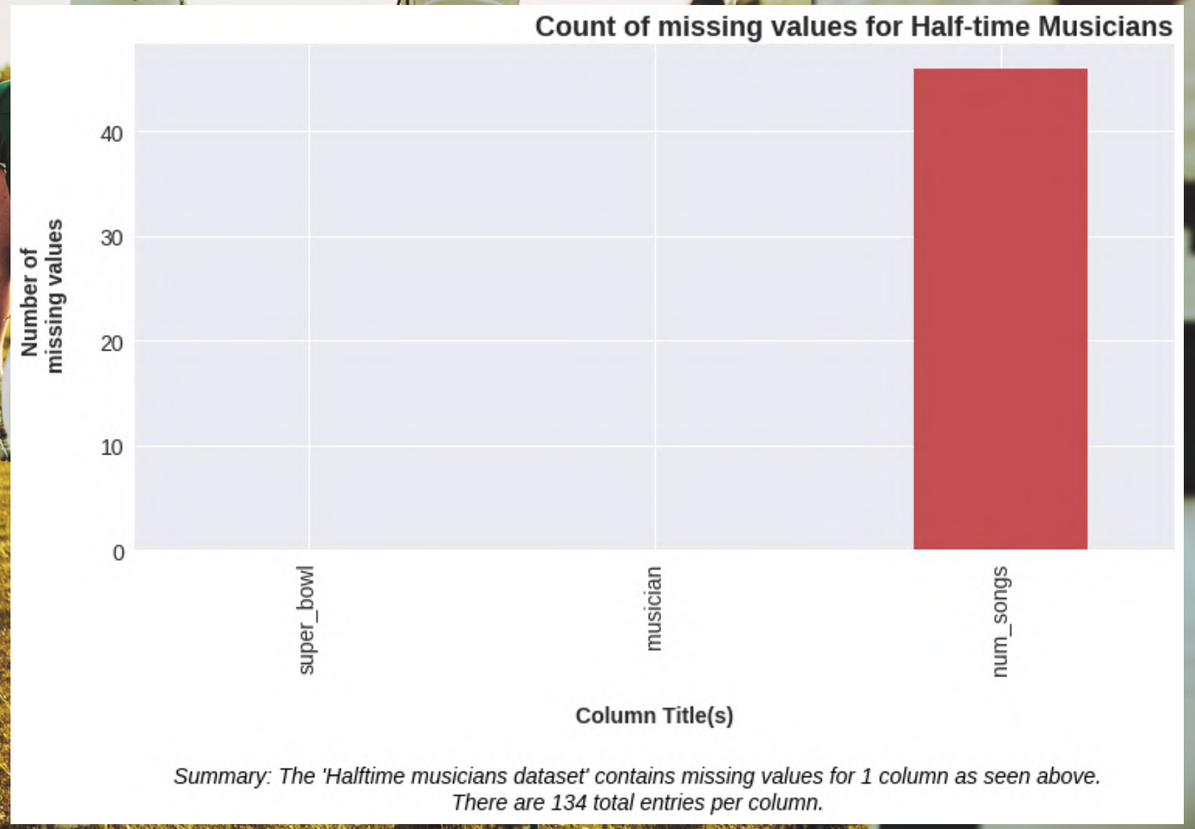
Taking note of data set issues

```
# Lets visualize the missing values
tv.isna().sum().plot(kind='barh', figsize=[12,6])
plt.title("Count of missing values for TV", weight='bold', size=18, loc='right')
plt.xlabel("Number of missing values", labelpad=20, weight='semibold', size=14)
plt.ylabel("Column titles", labelpad=20, weight='semibold', size=14)
plt.tick_params(axis='x', labelsize=14)
plt.tick_params(axis='y', labelsize=14)
plt.savefig('TV_NaN.png')
txt="Summary: The 'TV dataset' contains missing values for 3 columns as seen above.\n There are 53
total entries per column."
plt.figtext(0.5, -0.09, txt, wrap=True, horizontalalignment='center', fontsize=14, color='slategrey',
style='italic')
plt.show()
```

Taking note of data set issues



Taking note of data set issues

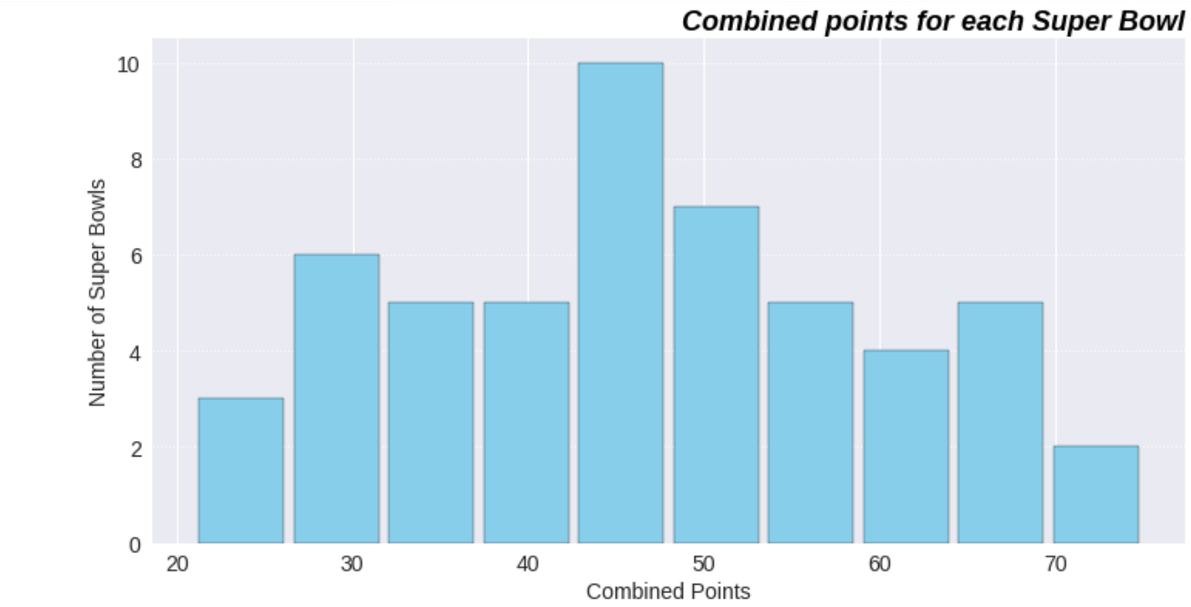


Summary: The 'Halftime musicians dataset' contains missing values for 1 column as seen above. There are 134 total entries per column.

Distribution for Super Bowl data

```
# Import matplotlib and set plotting style
# %matplotlib inline is a magic Jupyter Notebook command that allows you to display your graphs without plt.show()
from matplotlib import pyplot as plt
%matplotlib inline
plt.style.use('seaborn')

# Plot a histogram of combined points
plt.figure(figsize=[12,6])
plt.grid(axis='y', alpha=1, linestyle='dotted')
plt.hist(super_bowls.combined_pts, bins=10,  color = 'skyblue', edgecolor = 'black', rwidth=0.9)
plt.xlabel('Combined Points', fontsize=14)
plt.ylabel('Number of Super Bowls', fontsize=14)
plt.title('Combined points for each Super Bowl', weight='bold', size=18, loc='right', style='italic', fontweight=1000, color='black')
plt.tick_params(axis='x', labelsize=14)
plt.tick_params(axis='y', labelsize=14)
txt="Summary: Most combined scores are around 40-50 points, with the extremes being roughly equal distance away in opposite directions.\n Going up to the highest combined scores at 74 and 75, we find two games featuring dominant quarterback performances.\n Going down to the lowest combined scores, we have Super Bowl III and VII, which featured tough defenses that dominated."
plt.figtext(0.5, -0.10, txt, wrap=True, horizontalalignment='center', fontsize=14,color='black', style='italic')
plt.show()

# Display the Super Bowls in tabular format with the highest and lowest combined scores
display(super_bowls[super_bowls['combined_pts'] > 70])
display(super_bowls[super_bowls['combined_pts'] < 25])
```

Distribution for Super Bowl data

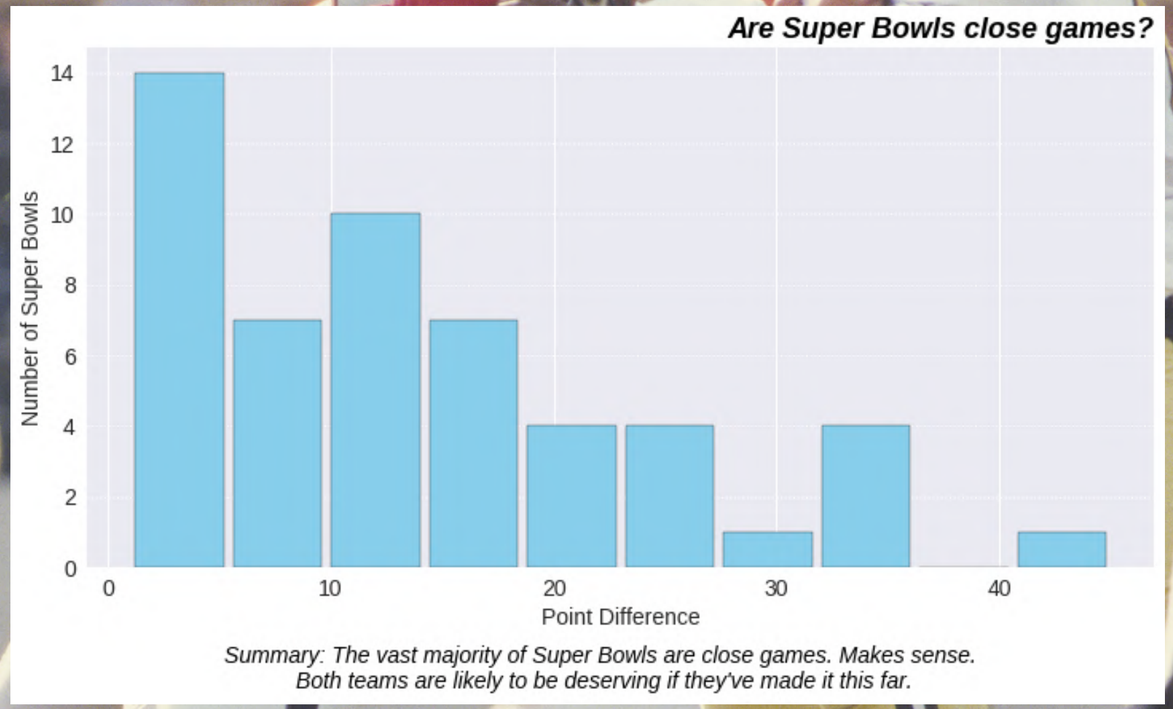


Summary: Most combined scores are around 40-50 points, with the extremes being roughly equal distance away in opposite directions. Going up to the highest combined scores at 74 and 75, we find two games featuring dominant quarterback performances. Going down to the lowest combined scores, we have Super Bowl III and VII, which featured tough defenses that dominated.

Point Difference & Close Games

```
# Histogram of point differences + display the rows with the most extreme point difference outcomes
plt.figure(figsize=[12,6])
plt.grid(axis='y', alpha=1, linestyle='dotted')
plt.hist(super_bowls.difference_pts,bins=10,  color = 'skyblue', edgecolor = 'black', rwidth=0.9)
plt.xlabel('Point Difference', fontsize=14)
plt.ylabel('Number of Super Bowls', fontsize=14)
plt.title('Are Super Bowls close games?', weight='bold', size=18, loc='right', style='italic',
fontweight=1000, color='black')
plt.tick_params(axis='x', labelsize=14)
plt.tick_params(axis='y', labelsize=14)
txt="Summary: The vast majority of Super Bowls are close games. Makes sense. \n Both teams are likely
to be deserving if they've made it this far."
plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center', fontsize=14,color='black',
style='italic')
plt.show()

print('\n')

# Display the closest game(s) and biggest blowouts
display(super_bowls[super_bowls['difference_pts'] == 1])
display(super_bowls[super_bowls['difference_pts'] >= 35])
```

Point Difference & Close Games

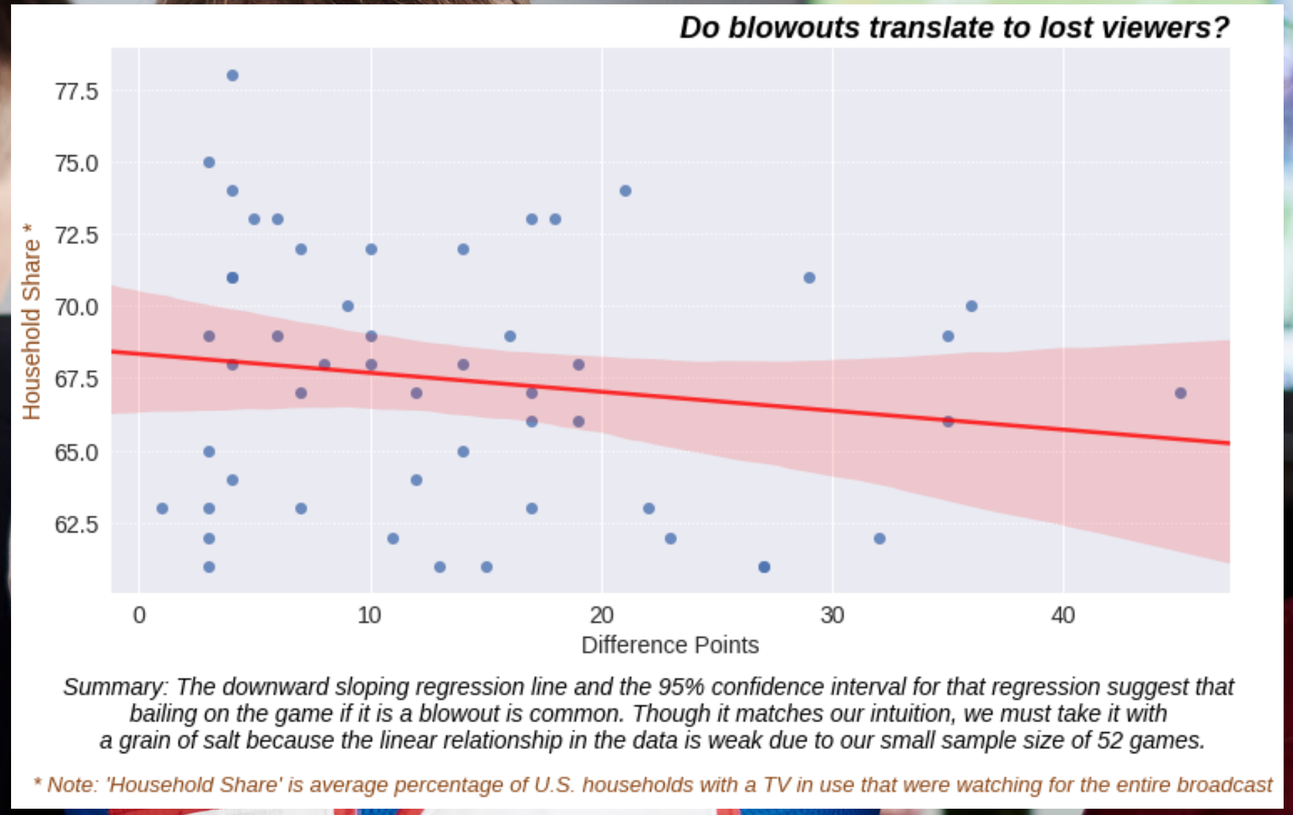


Summary: The vast majority of Super Bowls are close games. Makes sense. Both teams are likely to be deserving if they've made it this far.

Point differences & lost viewership

```
# Do large point differences translate to lost viewers?
# Join game and TV data, filtering out SB I because it was split over two networks

games_tv = pd.merge(tv[tv['super_bowl'] > 1], super_bowls, on='super_bowl')

# Create a scatter plot with a linear regression model fit: plot household share vs. point difference
# Plots the resulting regression line and a 95% confidence interval for that regression.

plt.figure(figsize=[12,6])
plt.grid(axis='y', alpha=1, linestyle='dotted')
sns.regplot(x='difference_pts', y='share_household', data=games_tv, line_kws=
{"color":"red","alpha":.8,"lw":2.5})
plt.xlabel('Difference Points', fontsize=14)
plt.ylabel('Household Share *', fontsize=14, color='saddlebrown')
plt.title('Do blowouts translate to lost viewers?', weight='bold', size=18, loc='right',
style='italic', fontweight=1000, color='black')
plt.tick_params(axis='x', labelsize=14)
plt.tick_params(axis='y', labelsize=14)

txt="Summary: The downward sloping regression line and the 95% confidence interval for that regression
suggest that \n bailing on the game if it is a blowout is common. Though it matches our intuition, we
must take it with \n a grain of salt because the linear relationship in the data is weak due to our
small sample size of 52 games."
plt.figtext(0.5, -0.09, txt, wrap=True, horizontalalignment='center', fontsize=14, color='slategrey',
style='italic')

txt="* Note: 'Household Share' is average percentage of U.S. households with a TV in use that were
watching for the entire broadcast"
plt.figtext(0.5, -0.15, txt, wrap=True, horizontalalignment='center', fontsize=13, color='saddlebrown',
style='italic')

plt.show()
```

Point differences & lost viewership



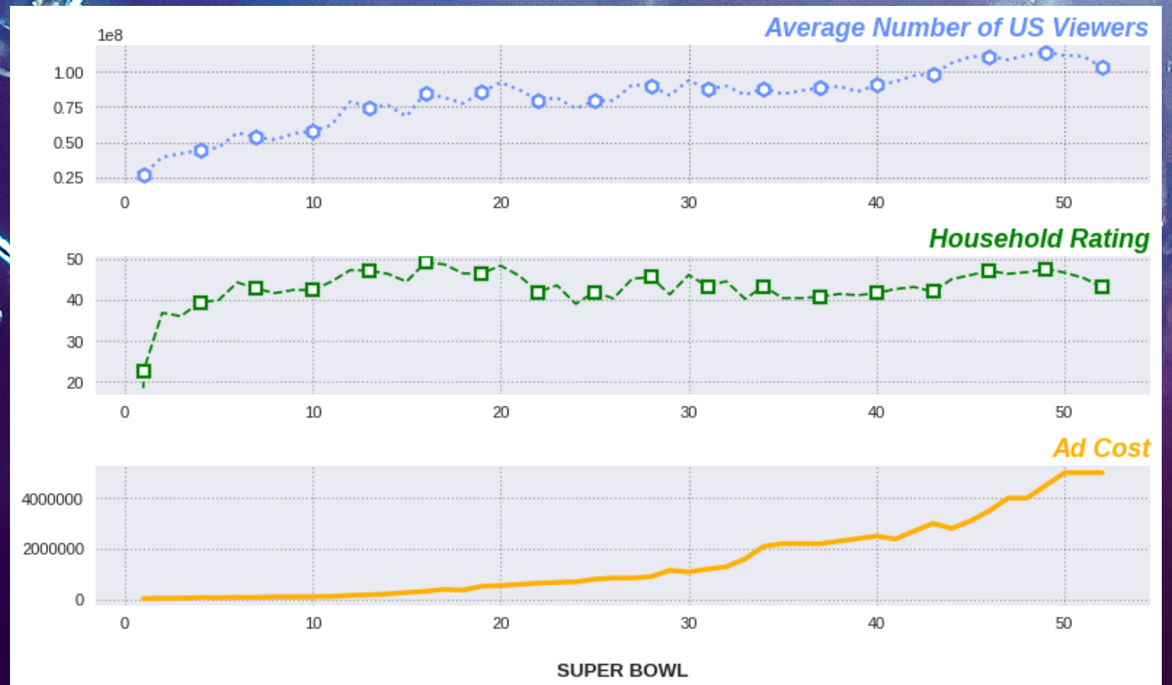
Summary: The downward sloping regression line and the 95% confidence interval for that regression suggest that bailing on the game if it is a blowout is common. Though it matches our intuition, we must take it with a grain of salt because the linear relationship in the data is weak due to our small sample size of 52 games.

** Note: 'Household Share' is average percentage of U.S. households with a TV in use that were watching for the entire broadcast*

Viewership & ad industry over time

```
# Size the plot
from matplotlib.pyplot import figure
figure(num=None, figsize=(10, 6), dpi=80, facecolor='w', edgecolor='k')

# Create a figure with 3x1 subplot and activate the top subplot
plt.subplot(3, 1, 1)
plt.plot(tv.super_bowl, tv.avg_us_viewers, linestyle = ':',color="#648FFF",linewidth=1.9,marker='h',
markerfacecolor='white', markeredgewidth=2, markersize=8, markevery=3)
plt.grid(axis='y', alpha=0.4,linestyle='dotted', color="black")
plt.grid(axis='x', alpha=0.4,linestyle='dotted', color="black")
plt.title('Average Number of US Viewers',weight='bold', size=16, loc='right', style='italic',
color="#648FFF")

# Activate the middle subplot
plt.subplot(3,1,2)
plt.plot(tv.super_bowl, tv.rating_household, linestyle = '--',linewidth=1.5, color="green", marker='s',
markerfacecolor='white', markeredgewidth=2,markersize=8, markevery=3)
plt.grid(axis='y', alpha=0.3,linestyle='dotted', color="black")
plt.grid(axis='x', alpha=0.3,linestyle='dotted', color="black")
plt.title('Household Rating', weight='bold', size=16, loc='right', style='italic',color="green")

# Activate the bottom subplot
plt.subplot(3, 1, 3)
plt.plot(tv.super_bowl, tv.ad_cost, color="#FFB000", linewidth=3)
plt.grid(axis='y', alpha=0.3,linestyle='dotted', color="black")
plt.grid(axis='x', alpha=0.3,linestyle='dotted', color="black")
plt.title('Ad Cost', weight='bold', size=16, loc='right', style='italic', color="#FFB000")
plt.xlabel('SUPER BOWL', labelpad=20, weight='semibold', size=12)

# Improve the spacing between subplots
plt.tight_layout()

txt="Viewpoint: A 30-second spot costs a pretty $5 million now, but has it always been that way? \n And
how have number of viewers and household ratings trended alongside ad cost?"
plt.figtext(0.5, -0.09, txt, wrap=True, horizontalalignment='center', fontsize=14, color='red',
alpha=0.6, style='italic')

txt="Summary: We can see viewers increased before ad costs did. Maybe the networks weren't very data
savvy \n and were slow to react? Perhaps, we can explore another hypothesis in the next chart."
plt.figtext(0.5, -0.18, txt, wrap=True, horizontalalignment='center', fontsize=14, color='black',
style='italic')
```

Viewership & ad industry over time



Viewpoint: A 30-second spot costs a pretty $5 million now, but has it always been that way? And how have number of viewers and household ratings trended alongside ad cost?

Summary: We can see viewers increased before ad costs did. Maybe the networks weren't very data savvy and were slow to react? Perhaps, we can explore another hypothesis in the next chart.

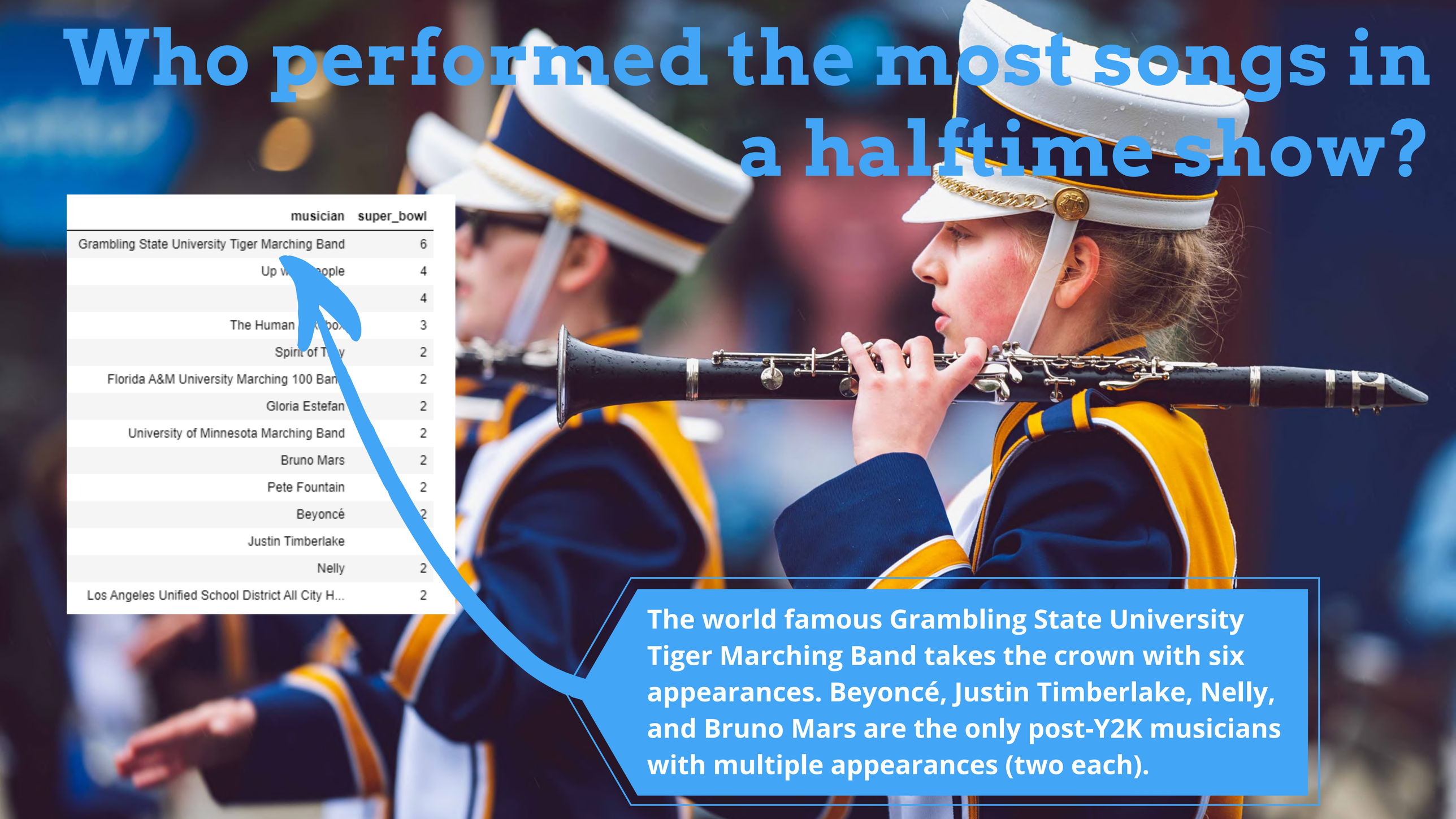
super_bowl	musician	num_songs
27	Michael Jackson	5.0
26	Gl Estefan	2.0
26	✓ University of Minnesota Ma Band	NaN
25	New Ki ock	2.0
24	Pe Fo ntain	1.0
24	Do g Kershaw	1.0
24	na Thomas	1.0
24	✓ Pride of Nicholls M ching Band	NaN
24	The Hu an Jukebox	NaN
24	Pri of Acadiana	NaN
23	Elvis Presto	7.0
22	Ch by Checker	2.0
22	✓ San Diego State University Ma hing Aztecs	NaN
22	pirit of Troy	NaN
✓	Grambling State University Tiger M hing Band	8.0
21	rit of Troy	8.0
20	Up n People	NaN
19	To n Blue	NaN
✓	The University of Florida Fightin' Gator rch...	7.0
✓	The Florida State University Marching	
✓	Los Angeles Unified School District All City H..	
16	Up with People	
15	The Human Jukebo	
15	Helen O'Connell	
14	Up with People	
✓	Grambling State University Tiger Marching Band	

Halftime shows weren't always this great

✓ Lots of marching bands!

The halftime shows before MJ Super Bowl XXVII performance indeed weren't that impressive, which we can see by filtering our halftime musician data

Who performed the most songs in a halftime show?



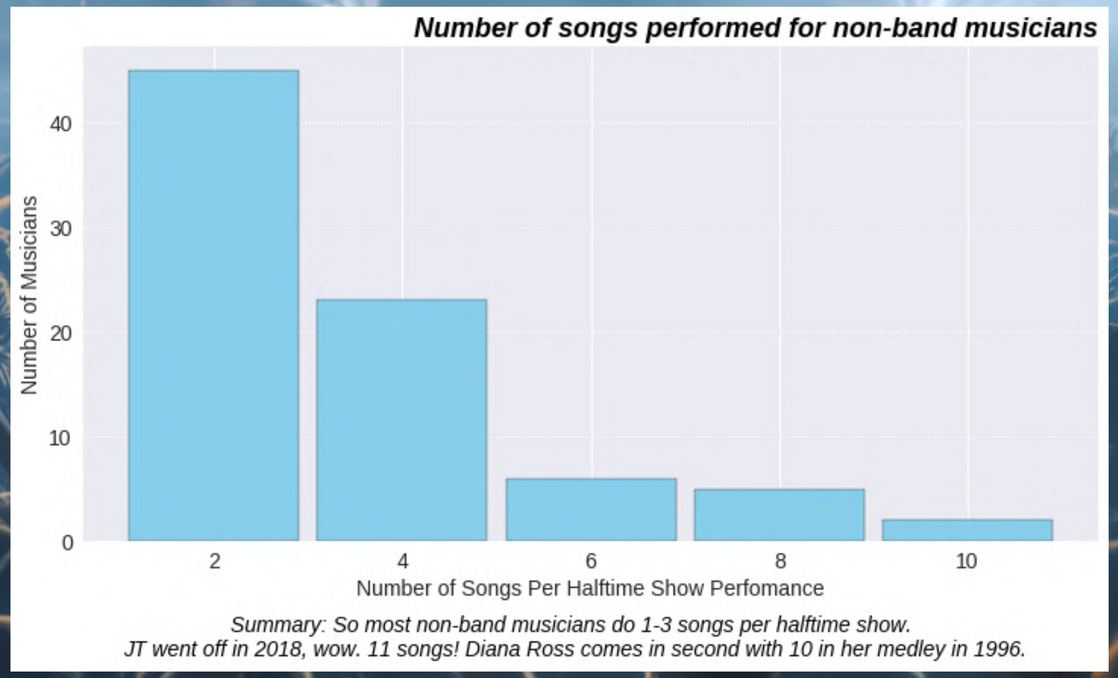
musician	super_bowl
Grambling State University Tiger Marching Band	6
Up w… …ople	4
	4
The Human … box	3
Spirit of T… …y	2
Florida A&M University Marching 100 Ban…	2
Gloria Estefan	2
University of Minnesota Marching Band	2
Bruno Mars	2
Pete Fountain	2
Beyoncé	2
Justin Timberlake	2
Nelly	2
Los Angeles Unified School District All City H...	2

The world famous Grambling State University Tiger Marching Band takes the crown with six appearances. Beyoncé, Justin Timberlake, Nelly, and Bruno Mars are the only post-Y2K musicians with multiple appearances (two each).

Non-band musicians after cleaning data issues

```
# Filter out most marching bands
no_bands = halftime_musicians[~halftime_musicians.musician.str.contains('Marching')]
no_bands = no_bands[~no_bands.musician.str.contains('Spirit')]

# Plot a histogram of number of songs per performance
most_songs = int(max(no_bands['num_songs'].values))
plt.figure(figsize=[12,6])
plt.grid(axis='y', alpha=1, linestyle='dotted')
plt.hist(no_bands.num_songs.dropna(), bins= 5,  color = 'skyblue', edgecolor = 'black', rwidth=0.9)
plt.xlabel('Number of Songs Per Halftime Show Perfomance', fontsize=14)
plt.ylabel('Number of Musicians', fontsize=14)
plt.title('Number of songs performed for non-band musicians', weight='bold', size=18, loc='right',
style='italic', fontweight=1000, color='black')
plt.tick_params(axis='x', labelsize=14)
plt.tick_params(axis='y', labelsize=14)
txt="Summary: So most non-band musicians do 1-3 songs per halftime show. \n JT went off in 2018, wow.
11 songs! Diana Ross comes in second with 10 in her medley in 1996."
plt.figtext(0.5, -0.05, txt, wrap=True, horizontalalignment='center', fontsize=14,color='black',
style='italic')
plt.show()

# Sort the non-band musicians by number of songs per appearance...
no_bands = no_bands.sort_values('num_songs', ascending=False)
# ...and display the top 15
display(no_bands.head(15))
```

Non-band musicians after cleaning data issues



Summary: So most non-band musicians do 1-3 songs per halftime show.
JT went off in 2018, wow. 11 songs! Diana Ross comes in second with 10 in her medley in 1996.

super_bowl	musician	num_songs
52	Justin Timberlake	11.0
30	Diana Ross	10.0
49	Katy Perry	8.0
51	Lady Gaga	7.0
23	Elvis Presto	7.0
41	Prince	7.0
47	Beyoncé	7.0
48	Bruno Mars	6.0
50	Coldplay	6.0
45	The Black Eyed Peas	6.0
46	Madonna	5.0
44	The Who	5.0
27	Michael Jackson	5.0
32	The Temptations	4.0
39	Paul McCartney	4.0

Jovan Trajceski

https://www.linkedin.com/in/trajceski/

#100DaysofCode

#100DaysofLearning


Have a Great Day !