image from https://www.enworld.org/resources/d-d-5e-logo-pack.1043/

An Analysis of Dungeons and Dragons 5ed Monsters

Using data found from Patrick Gomes at Kaggle I will look to see if I can answer a few questions regarding the diversity of monsters in Dungeons and Dragons Fifth Edition (to be called 5e for the remainder of this project). I will pose the following questions:

1. What are the most common monster types (shown as race in this data)?

2. Is there any connection between monster type and alignment?

3. What does the spread of alignment look like for an individual monster race?

4. Does monster size impact hit point amounts?

5. Does a monster's armor class have a correlation with its hit points?

Let's start off with some diagnostic analysis and clean up any data we want to work with

In [ ]:

```
# import libraries used for data analysis
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.markers import MarkerStyle

sns.set_theme(style="darkgrid", palette="rocket_r")
```

In [ ]:

```
# read in the CSV file and see a preview of the data
dnd = pd.read_csv('Dd5e_monsters.csv')
dnd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 324 entries, 0 to 323
Data columns (total 7 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   Name                    324 non-null    object
 1   Size                    324 non-null    object
 2   Race + alignment        324 non-null    object
 3   HP                      324 non-null    object
 4   Armor                   324 non-null    object
 5   Speed                   324 non-null    object
 6   Challenge rating  (XP)  324 non-null    object
dtypes: object(7)
memory usage: 17.8+ KB
```

In [ ]:

```
# look over the columns names and first three rows worth of data
dnd.head(3)
```

Out[ ]:

	Name	Size	Race + alignment	HP	Armor	Speed	Challenge rating (XP)
0	Aboleth	Large	aberration, Lawful Evil	135 (18d10+36)	17 (Natural Armor)	10 ft., swim 40 ft.	10 (5,900 XP)
1	Acolyte	Medium	humanoid (any race), Any Alignment	9 (2d8)	10	30 ft.	1/4 (50 XP)
2	Adult Black Dragon	Huge	dragon, Chaotic Evil	195 (17d12+85)	19 (Natural Armor)	40 ft., fly 80 ft., swim 40 ft.	14 (11,500 XP)

In [ ]:

```
# check for any null values even though this data set looks pretty clean at a glance
dnd.isnull().sum()
```

Out[ ]:

```
Name                      0
Size                      0
Race + alignment          0
HP                        0
Armor                     0
Speed                     0
Challenge rating  (XP)    0
dtype: int64
```

In [ ]:

```
# found that some of the data had multiple commas, so starting from the end of the stri
dnd[['Race','Alignment']] = dnd['Race + alignment'].str.rsplit(',', n=1, expand=True)
dnd['Alignment'] = dnd['Alignment'].str.strip()
```

In [ ]:

```
# clean up some of the other columns for future discussion.
dnd['Armor'] = dnd['Armor'].apply(lambda x: int(x.split(' (')[0]))
dnd['HP'] = dnd['HP'].apply(lambda x: int(x.split(' (')[0]))
```

We now have two columns for Race and Alignment so we can remove the original, and we aren't workign with Challenge rating or Speed, so let's drop that.

In [ ]:

```
dnd = dnd.drop(['Race + alignment', 'Challenge rating  (XP)', 'Speed'], axis=1)
```

In [ ]:

```
dnd.head()
```

Out[ ]:

	Name	Size	HP	Armor	Race	Alignment
0	Aboleth	Large	135	17	aberration	Lawful Evil
1	Acolyte	Medium	9	10	humanoid (any race)	Any Alignment
2	Adult Black Dragon	Huge	195	19	dragon	Chaotic Evil
3	Adult Blue Dragon	Huge	225	19	dragon	Lawful Evil
4	Adult Brass Dragon	Huge	172	18	dragon	Chaotic Good

In [ ]:

```
# Use groupby with dropna=False to find any null values in categorical data
dnd.groupby(['Name', 'Size', 'Race', 'Alignment', 'HP', 'Armor'], dropna=False, as_inde
```

Out[ ]:

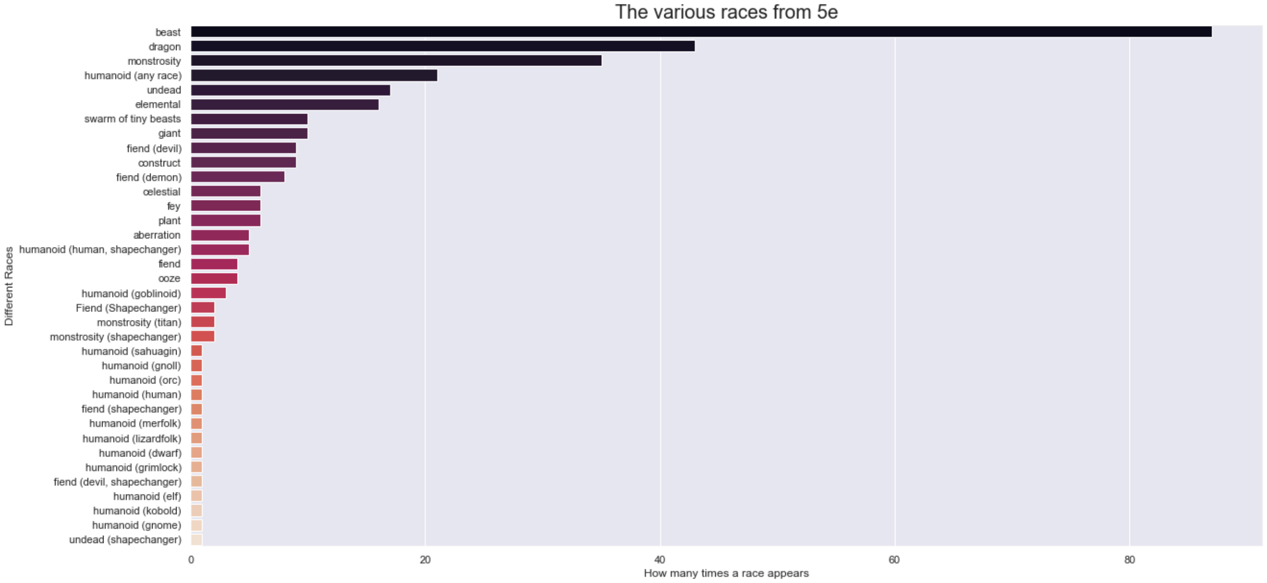
	Name	Size	Race	Alignment	HP	Armor	size
0	Aboleth	Large	aberration	Lawful Evil	135	17	1
1	Acolyte	Medium	humanoid (any race)	Any Alignment	9	10	1
2	Adult Black Dragon	Huge	dragon	Chaotic Evil	195	19	1
3	Adult Blue Dragon	Huge	dragon	Lawful Evil	225	19	1
4	Adult Brass Dragon	Huge	dragon	Chaotic Good	172	18	1
...	...	...	...	...	...	...	...
319	Young Green Dragon	Large	dragon	Lawful Evil	136	18	1
320	Young Red Dragon	Large	dragon	Chaotic Evil	178	18	1
321	Young Silver Dragon	Large	dragon	Lawful Good	168	18	1
322	Young White Dragon	Large	dragon	Chaotic Evil	133	17	1
323	Zombie	Medium	undead	Neutral Evil	22	8	1

324 rows × 7 columns

What are the most common monster types (shown as race in this data)?

In [ ]:

```python
different_races_count = dnd['Race'].value_counts()
different_races = dnd['Race'].value_counts().keys()

fig, ax = plt.subplots(figsize=(20,10))

sns.barplot(ax=ax, x=different_races_count, y=different_races,
            data=dnd,
            palette="rocket")
plt.title('The various races from 5e', fontsize=20)
plt.xlabel('How many times a race appears')
plt.ylabel('Different Races')
plt.show()
```



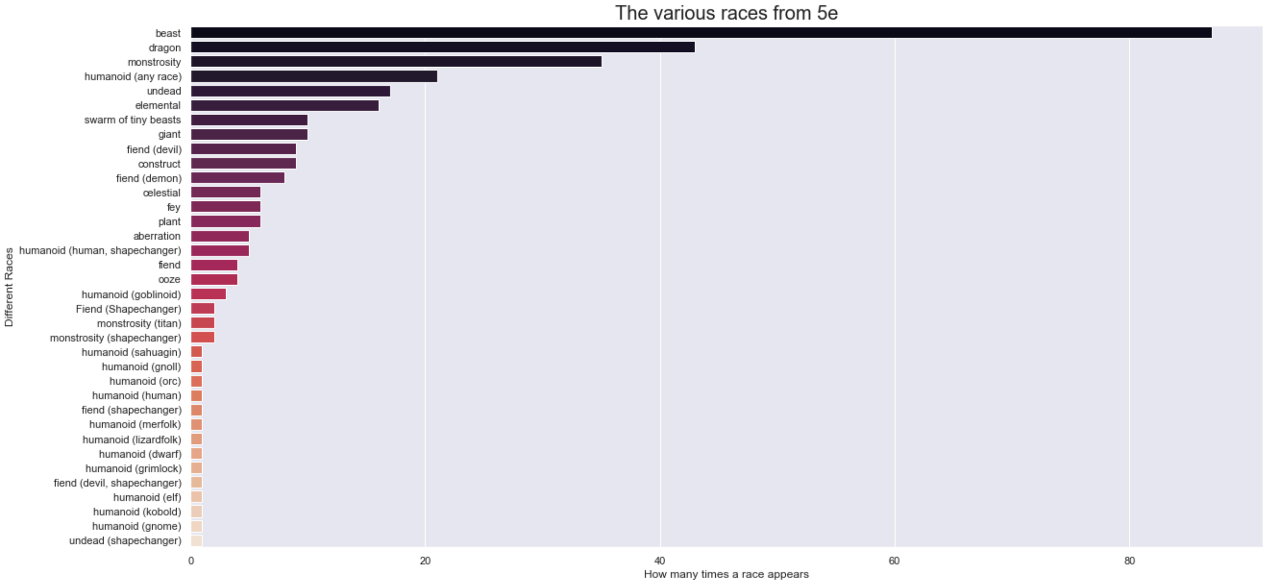
Even if we combined all of the humanoid variants, the number of Beasts surpasses all other individual races by double. Let's try and combine the races into fewer categories.

In [ ]:

```python
# combine all races to their main categories so the data looks more in sync.
dnd['Race'] = dnd['Race'].apply(lambda x: x.split(' ')[0])
```

In [ ]:

```python
fig, ax = plt.subplots(figsize=(20,10))

sns.barplot(ax=ax, x=different_races_count, y=different_races,
            data=dnd,
            palette="rocket")
plt.title('The various races from 5e', fontsize=20)
plt.xlabel('How many times a race appears')
plt.ylabel('Different Races')

# plt.savefig('graphs/races_of_5e.png', bbox_inches = 'tight', edgecolor='w')
plt.show()
```



As we can see from the chart beasts are the most prevalent race in 5e, with Dragons, Humanoids (of varying types), and Monstrosities making up the next bulk of monsters.

Is there any connection between monster type and alignment?

In [ ]:

```
# get an idea for which alignments we are looking at in the data
dnd['Alignment'].value_counts()
```

Out[ ]:

```
Unaligned                                     128
Chaotic Evil                                   44
Lawful Evil                                    37
Neutral Evil                                   28
Lawful Good                                    19
Neutral                                        19
Any Alignment                                  15
Chaotic Good                                   12
Neutral Good                                    6
Any Non-good Alignment                          4
Lawful Neutral                                  3
Chaotic Neutral                                 3
Any Non-lawful Alignment                        2
Neutral Good (50%) Or Neutral Evil (50%)        1
Any Chaotic Alignment                           1
Any Evil Alignment                              1
Any                                             1
Name: Alignment, dtype: int64
```

There are 9 alignments (and unaligned) that we care about, so let's take a look at just the basic alignments

In [ ]:

```
# creating a list of the alignments we care to look over.
alignments = ['Chaotic Evil', 'Neutral Evil','Lawful Evil',
              'Chaotic Neutral','Neutral','Lawful Neutral',
              'Chaotic Good', 'Neutral Good', 'Lawful Good',
              'Unaligned']

# this function will create a new column of 0/1 values so we narrow down our alignments
dnd['True Alignment'] = dnd['Alignment'].apply(lambda alignment: 1 if alignment in alig
dnd.head()
```

Out[ ]:

	Name	Size	HP	Armor	Race	Alignment	True Alignment
0	Aboleth	Large	135	17	aberration	Lawful Evil	1
1	Acolyte	Medium	9	10	humanoid	Any Alignment	0
2	Adult Black Dragon	Huge	195	19	dragon	Chaotic Evil	1
3	Adult Blue Dragon	Huge	225	19	dragon	Lawful Evil	1
4	Adult Brass Dragon	Huge	172	18	dragon	Chaotic Good	1

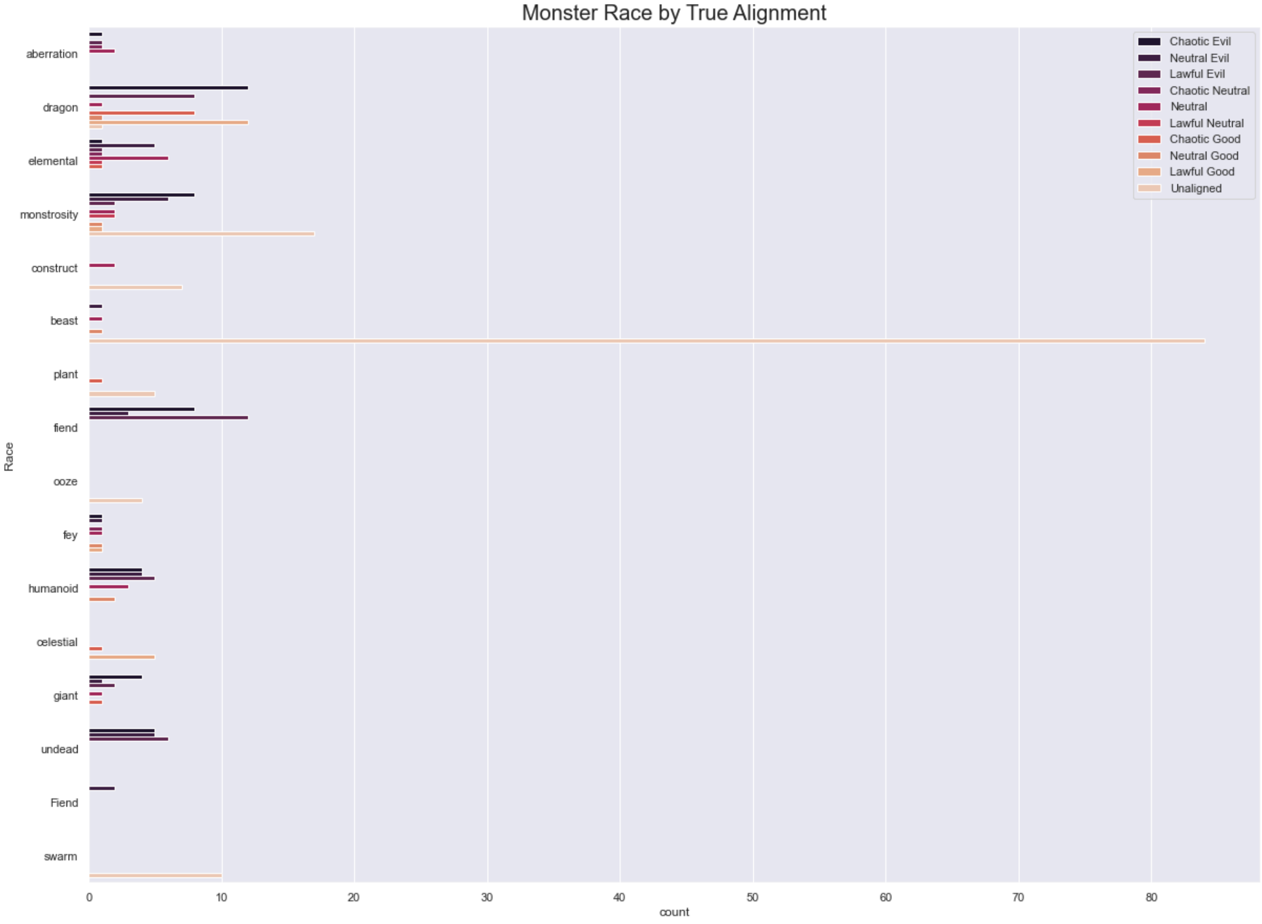
We are using a horizontal bargraph just for visualization because of how this material looks, even though a vertical bar group is best practice.

In [ ]:

```
# Use pd.Categorical to manually order the Size series instead of allowing for an alphal
dnd['Alignment'] = pd.Categorical(dnd['Alignment'], categories=alignments, ordered=True
```

In [ ]:

```
true_alignments = dnd[dnd['True Alignment'] == 1]

fig, ax = plt.subplots(figsize=(20,15))
sns.countplot(y=true_alignments['Race'],
              hue=true_alignments['Alignment'], palette="rocket")
plt.title('Monster Race by True Alignment', fontsize=20)
legend = plt.legend()
# plt.savefig('graphs/monster_Race_by_alignment.png', bbox_inches = 'tight', edgecolor=
plt.show()
```

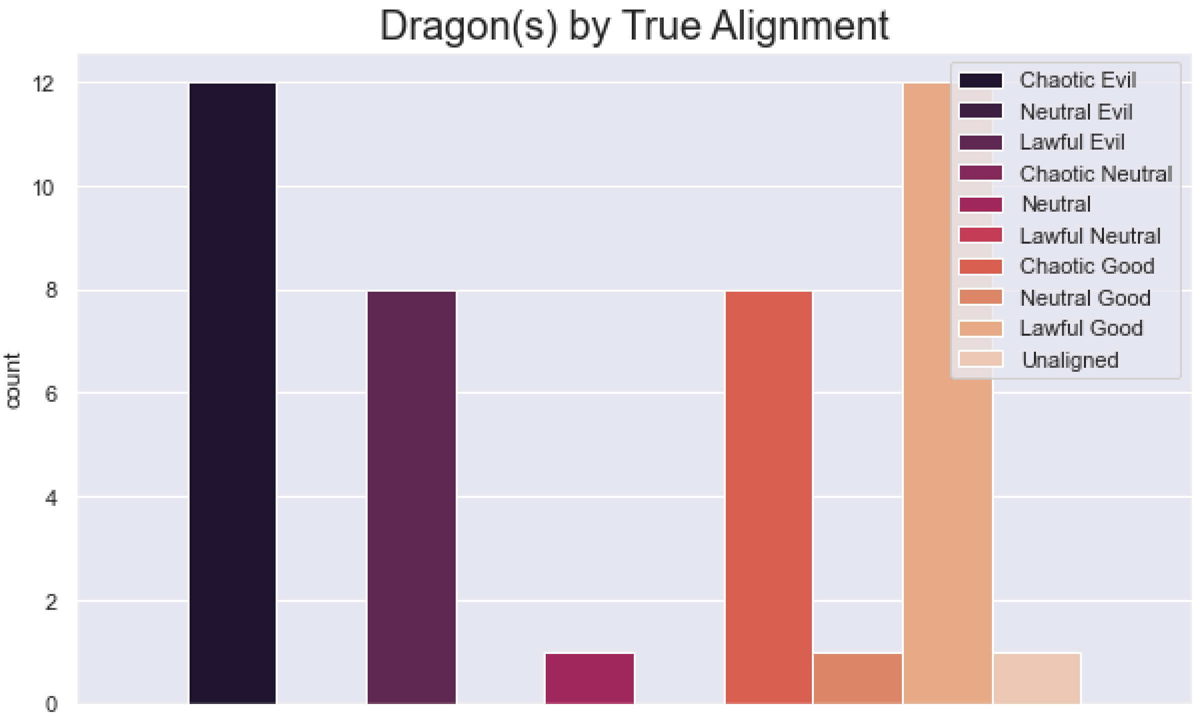


Beasts look to fall mostly in the unaligned category which seems to make sense. We can take a closer look at some of the other races with more variety.

What does the spread of alignment look like for an individual monster race?

In [ ]:

```
# create a function for looking at aligntment in individual races
def check_alignment(race):
    race = race.lower()
    single_race = true_alignments[true_alignments['Race']==race]

    fig, ax = plt.subplots(figsize=(10,6))
    sns.countplot(x=single_race['Race'],
                  hue=single_race['Alignment'], palette='rocket')
    plt.title(f'{race.capitalize()}(s) by True Alignment', fontsize=20)
    legend = plt.legend()
    plt.xticks(visible=False)
    plt.xlabel("")
    plt.savefig(f'graphs/{race}_alignment.png', bbox_inches = 'tight', edgecolor='w')
    plt.show()

check_alignment('dragon')
```



Dragons seem to have a variety, but also a dichotomy between good and evil.

I would like to create a heatmap that follows a 3x3 grid to make the iconic alignment map. This is definitely a future project if possible, but after spending some time researching, this might be out of reach.

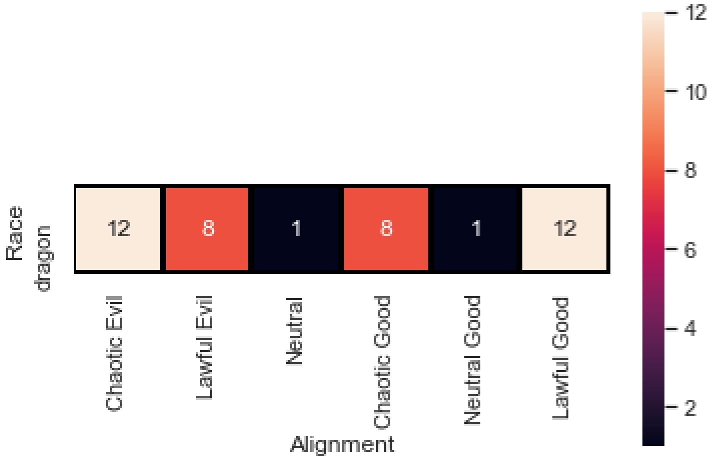
In [ ]:

```
# creating a list of the alignments we care to look over.
alignments_grid = ['Chaotic Evil', 'Neutral Evil','Lawful Evil',
                'Chaotic Neutral','Neutral','Lawful Neutral',
                'Chaotic Good', 'Neutral Good', 'Lawful Good']


# this function will create a new column of 0/1 values so we narrow down our alignments
```

```
dnd['Alignment Grid'] = dnd['Alignment'].apply(lambda alignment: 1 if alignment in alig
dnd

true_alignments_grid = dnd[dnd['Alignment Grid'] == 1.0]
true_alignments_grid
true_alignments_grid[(true_alignments_grid['Race'] == 'dragon') & (true_alignments_grid

substring='Unaligned'
filter=true_alignments_grid['Alignment'].str.contains(substring)

filtered_df = true_alignments_grid[~filter]
df=filtered_df[['Race', 'Alignment']]

counts = df.groupby(['Race', 'Alignment'], observed=True).size().reset_index(name='Coun
counts
heat_dnd = counts[counts['Race'] == 'dragon']
heat_dnd

pivot_dnd = heat_dnd.pivot(index='Race', columns='Alignment', values='Count').fillna(0)
sns.heatmap(pivot_dnd, square=True, annot=True, linewidth=2, cmap='rocket', linecolor='
```

Out[ ]:

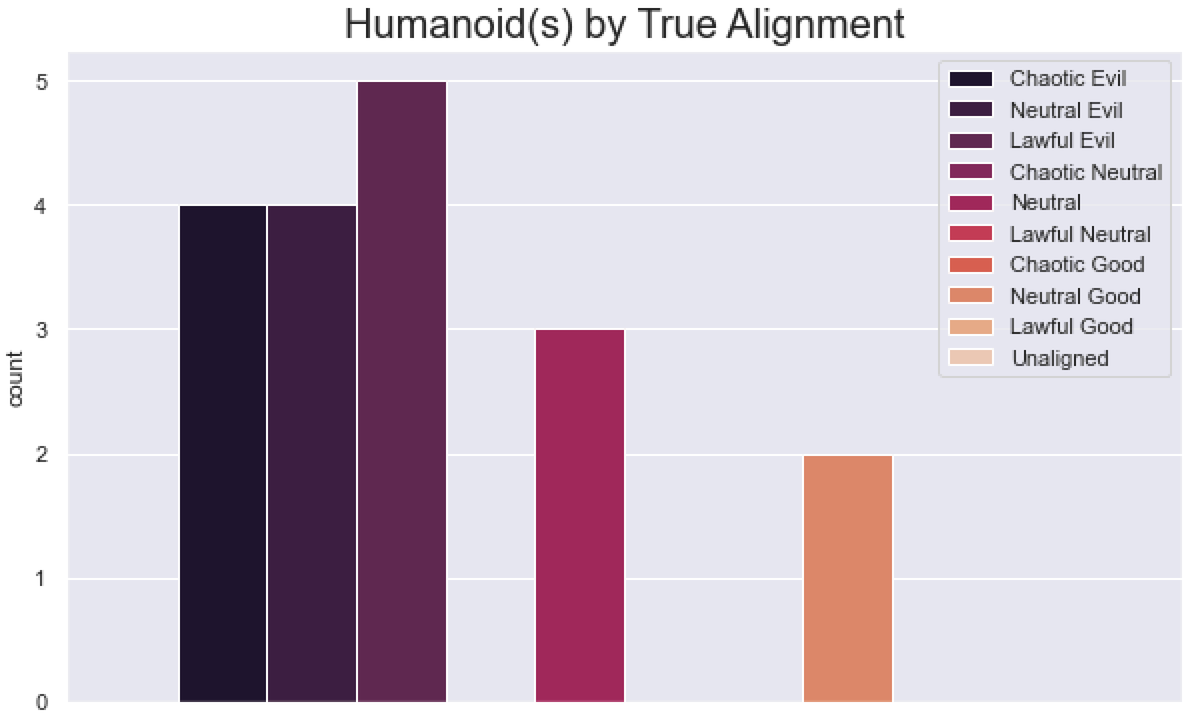
```
<AxesSubplot:xlabel='Alignment', ylabel='Race'>
```



Can't quite get the results I'm looking for here.

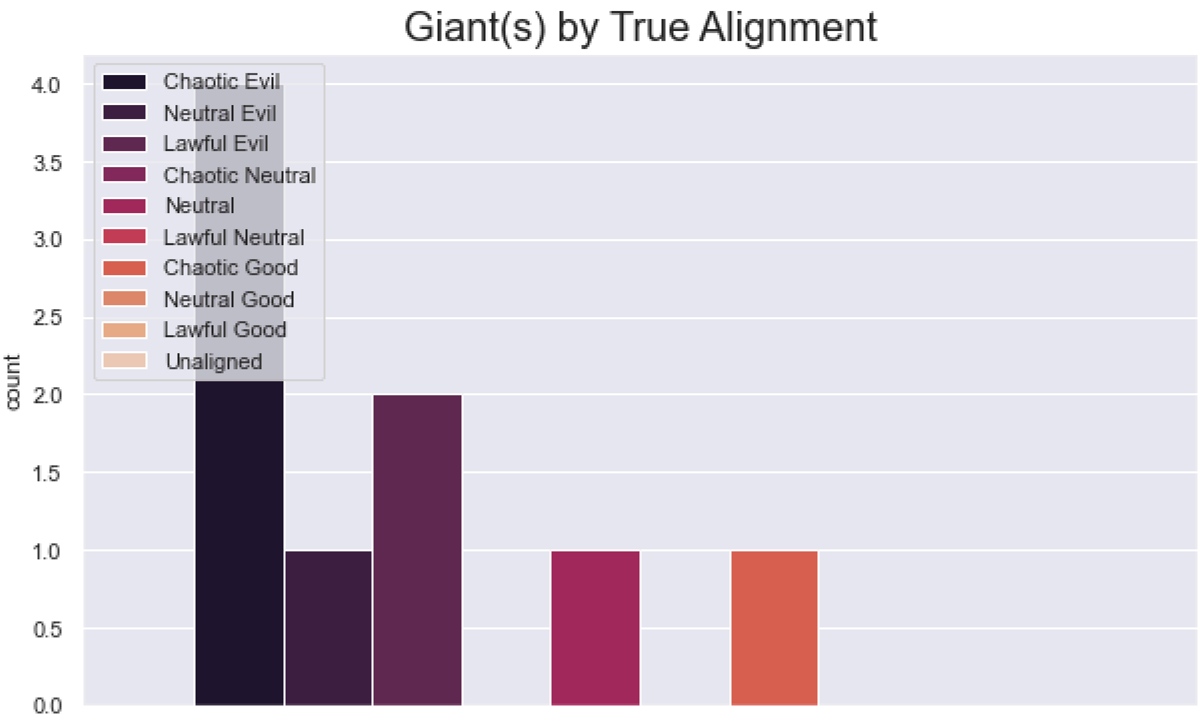
In [ ]:

```
check_alignment('humanoid')
```



In [ ]:

```
check_alignment('giant')
```



It seems like humanoids are mostly neutral or evil, which makes sense since most player characters are going to be the opposite.

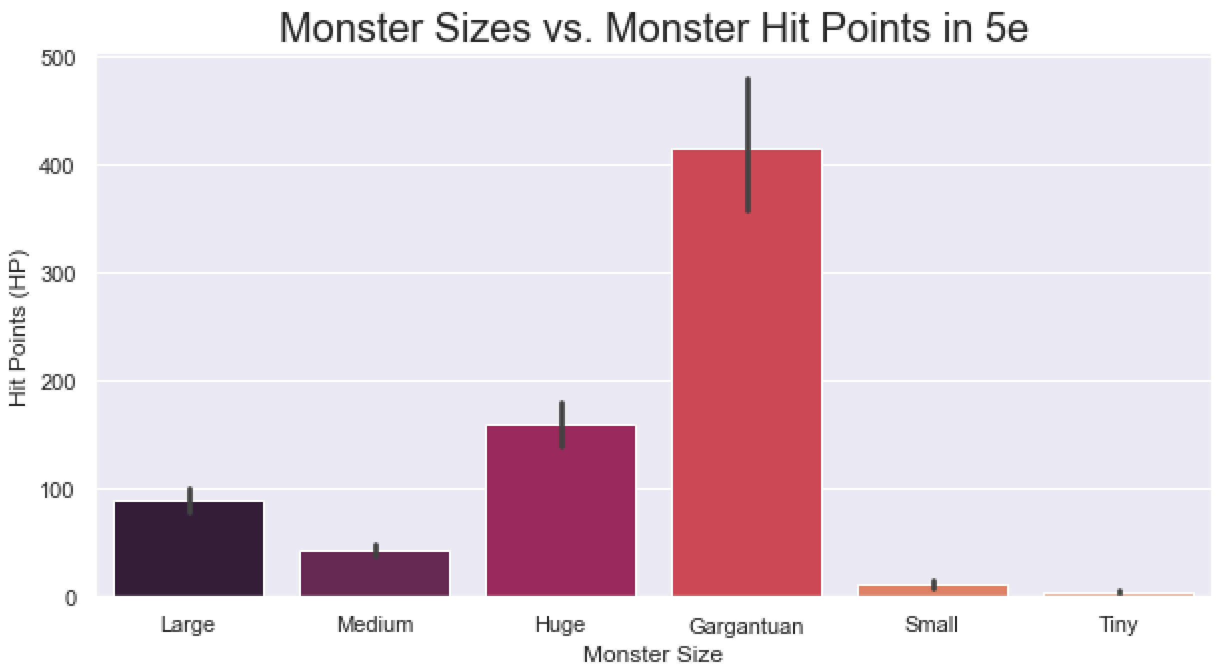
Does monster size impact hit point amounts?

In [ ]:

```
monster_size_and_hp = dnd[['Size', 'HP']]
monster_size = monster_size_and_hp['Size']
monster_hp = monster_size_and_hp['HP']
```

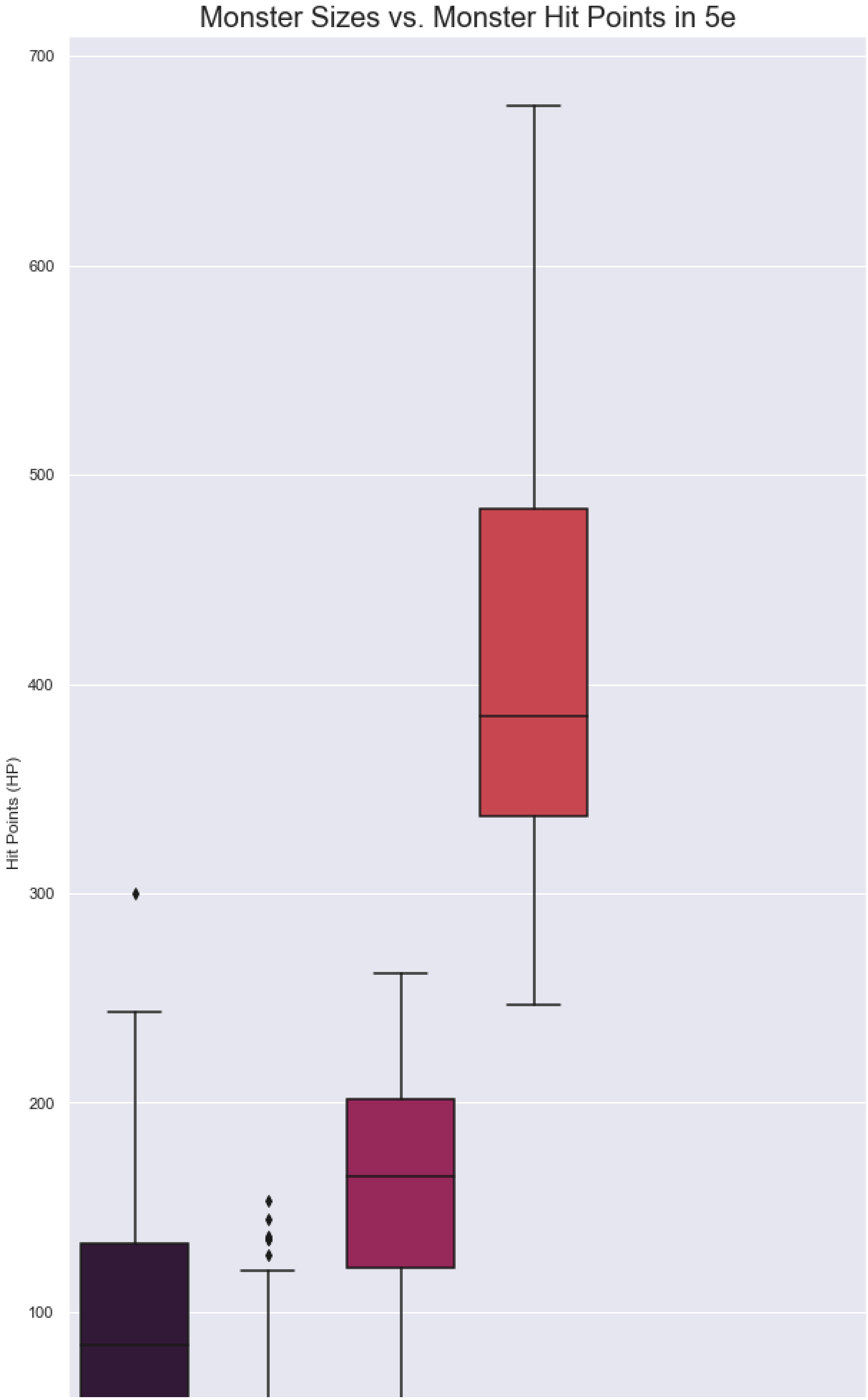
In [ ]:

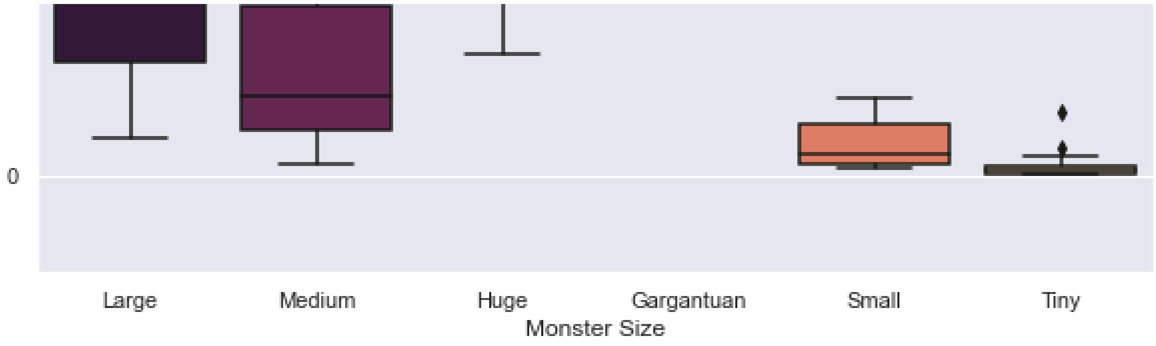
```
fig, ax = plt.subplots(figsize=(10,5))
sns.barplot(y=monster_hp, x=monster_size,
            data=monster_size_and_hp,
            palette='rocket')
plt.title('Monster Sizes vs. Monster Hit Points in 5e', fontsize=20)
plt.ylabel('Hit Points (HP)')
plt.xlabel('Monster Size')
# plt.savefig('graphs/monster_size_v_monster_hp.png', bbox_inches = 'tight', edgecolor=
plt.show()
```



In [ ]:

```
fig, ax = plt.subplots(figsize=(10,20))
sns.boxplot(y=monster_hp, x=monster_size,
            data=monster_size_and_hp,
            palette='rocket')
plt.title('Monster Sizes vs. Monster Hit Points in 5e', fontsize=20)
plt.ylabel('Hit Points (HP)')
plt.xlabel('Monster Size')
plt.savefig('graphs/monster_size_v_monster_hp_box.png', bbox_inches = 'tight', edgecolo
plt.show()
```


Monster Sizes vs. Monster Hit Points in 5e
Hit Points (HP)
700
600
500
400
300
200
100



In [ ]:
```
# finding the largest HP unit in a given monster size category
def find_monster_in_size_with_max_hp(size):
    df = dnd[monster_size==size.capitalize()]
    return df.loc[df['HP'] == df['HP'].max()]['Name'].values[0]
    # dnd[dnd['HP'] == dnd['HP'].max()]['Name'].values[0]

find_monster_in_size_with_max_hp('Gargantuan')
```

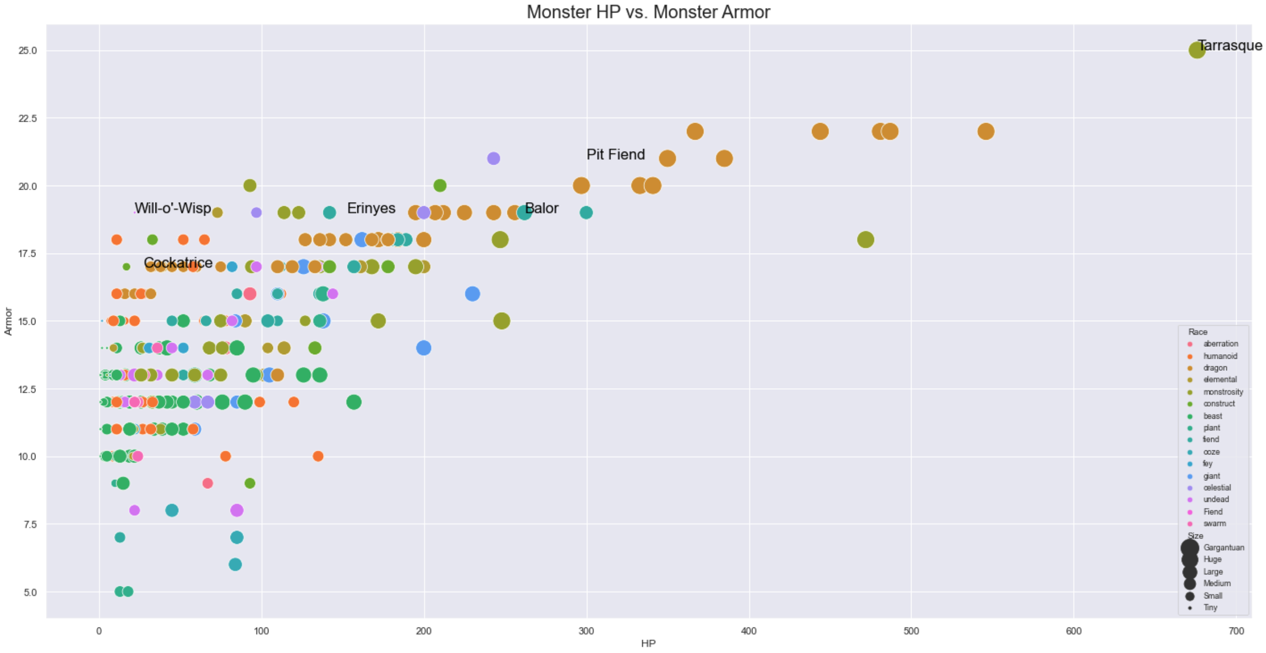
Out[ ]:
```
'Tarrasque'
```

Based off the chart above, the average HP for a monster is definitely associated with its size.

Does a monster's armor class have a correlation with its hit points?

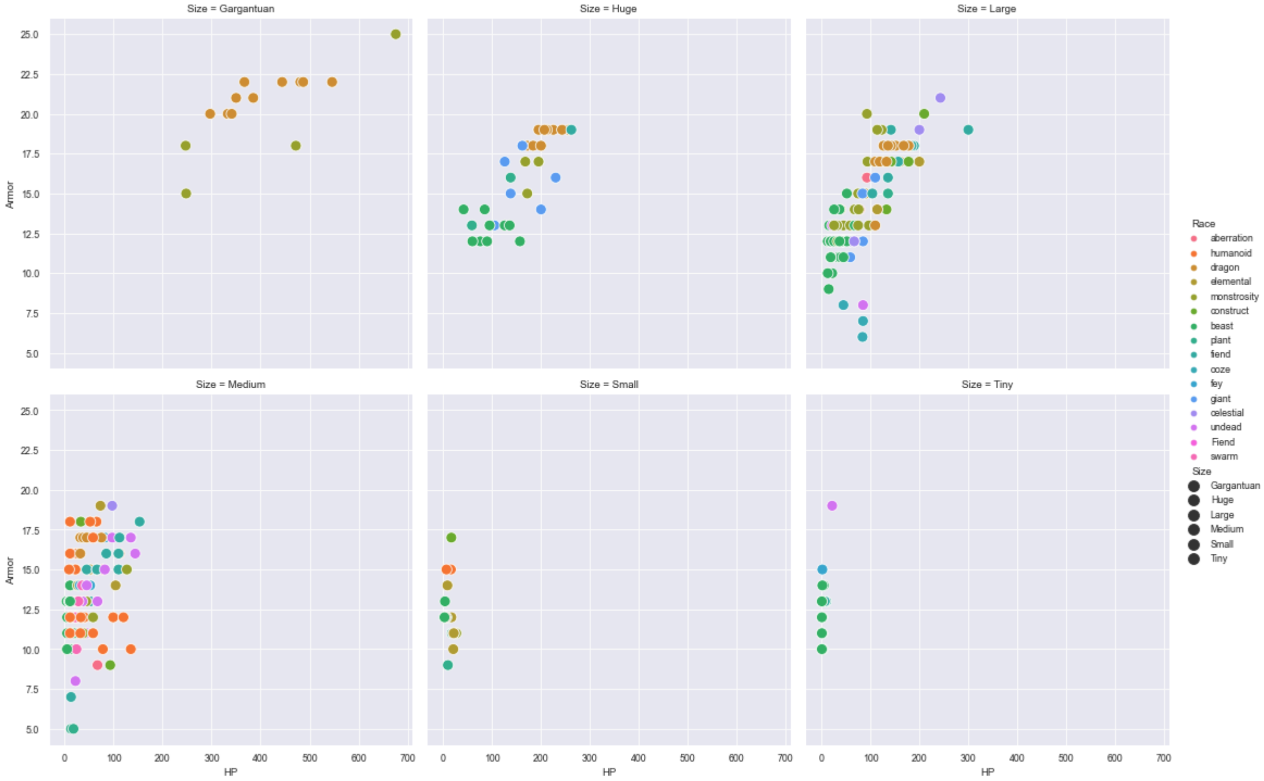
In [ ]:
```
# Use pd.Categorical to manually order the Size series instead of allowing for an alpha

dnd['Size'] = pd.Categorical(dnd['Size'], categories=['Gargantuan', 'Huge','Large','Med
marker_size = dnd.sort_values('Size', ascending=True)
```

In [ ]:
```
monster_hp = dnd['HP']
monster_armor = dnd['Armor']

MONSTER_SYMBOLS = []



fig,ax = plt.subplots(figsize=(20,10))
sns.set_context('paper')

sns.scatterplot(x=monster_hp, y=monster_armor, hue=dnd['Race'], size=marker_size['Size'

# Put the name of the monster that fits the max HP and Max armor for each size next to
plt.text(monster_hp[dnd['Size']=='Tiny'].max(),monster_armor[dnd['Size']=='Tiny'].max()
plt.text(monster_hp[dnd['Size']=='Small'].max(),monster_armor[dnd['Size']=='Small'].max
plt.text(monster_hp[dnd['Size']=='Medium'].max(),monster_armor[dnd['Size']=='Medium'].m
plt.text(monster_hp[dnd['Size']=='Large'].max(),monster_armor[dnd['Size']=='Large'].max
plt.text(monster_hp[dnd['Size']=='Huge'].max(),monster_armor[dnd['Size']=='Huge'].max()
plt.text(monster_hp[dnd['Size']=='Gargantuan'].max(),monster_armor[dnd['Size']=='Gargan
plt.tight_layout()
ax.set_title('Monster HP vs. Monster Armor', fontsize=20)
# plt.savefig('graphs/monster_hp_v_monster_armor.png', bbox_inches = 'tight', edgecolor
plt.show()
```



Let's separate all monster sizes and see how their plots look? What can we glean?

```
In [ ]: sns.relplot(
            x=monster_hp, y=monster_armor, hue=dnd['Race'],
              col=dnd['Size'], col_wrap=3,
              size=marker_size['Size'], sizes=(100, 100))

        # plt.savefig('graphs/monster_hp_v_monster_armor_across_sizes.png', bbox_inches = 'tigh

        plt.show()
```



It seems Medium size and Large size have the widest spread. The HP values for Tiny and Small are so consistent.

Suggestions to the Dungeons and Dragons creators

1. Based off findings, Beasts make up a huge portion of the monster races. I believe it would be best to create more official monsters from other races. The reason being that it promotes more campaign settings, especially for early levels.

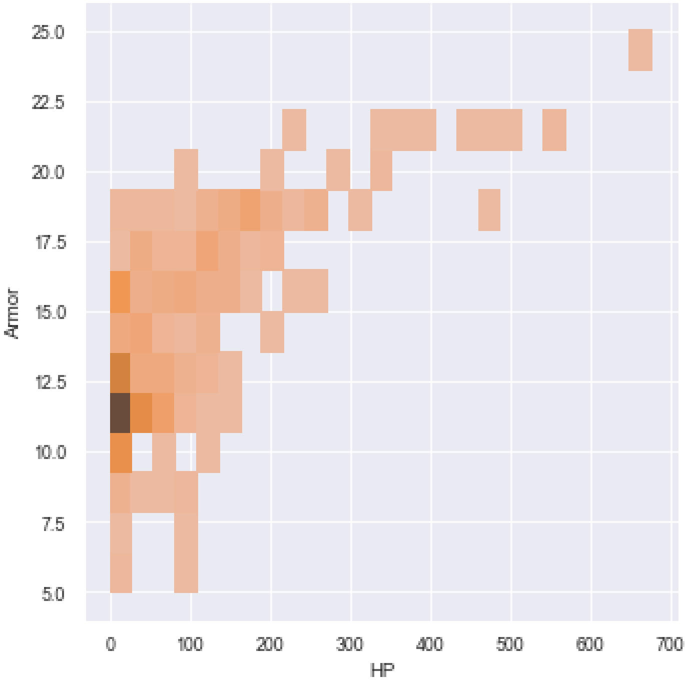
2. Another area of focus would be the bring in more "good" alignments for monster races. The data shows that Humanoids are mostly evil and neutral. Some more guidance for players who come up across lawful good NPCs would be interesting.

3. Last suggestion would be to add a few more variations for HP vs. Monster Size. Right now it's pretty proportional, as size goes up, HP goes up. It would be great to see some Tiny creatures with 20-50 HP just to give players a hard time. Something that is super hard to lock down, but maybe not super powerful.

More Exploration outside of this discovery for a later date.

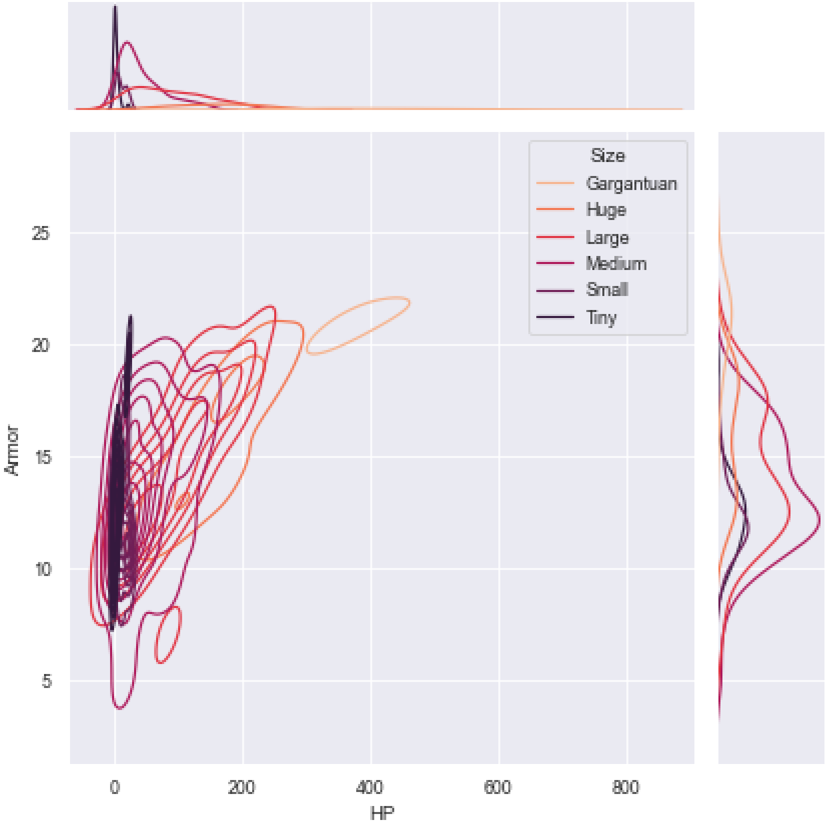
We can take a look at what the most used combination of HP and Armor out of all Monster races.

```
In [ ]: sns.displot(
            dnd, x=monster_hp, y=monster_armor)
        plt.show()
```



```
In [ ]: sns.jointplot(
            data=dnd, x=monster_hp, y=monster_armor, hue=dnd['Size'], kind='kde', observed=True
        plt.show()
```

```
c:\Users\dmm46\anaconda3\envs\learn-env\lib\site-packages\seaborn\distributions.py:1181:
UserWarning: The following kwargs were not used by contour: 'observed'
  cset = contour_func(
```

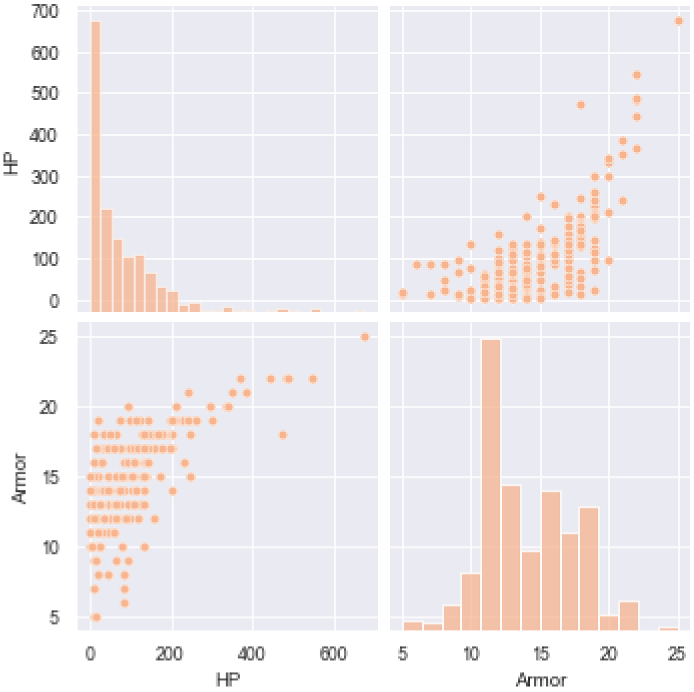


In [ ]:

```
sns.pairplot(dnd[['HP', 'Armor', ]])
```

Out[ ]:

```
<seaborn.axisgrid.PairGrid at 0x1e14f9aa0a0>
```


HP
Armor
HP
Armor