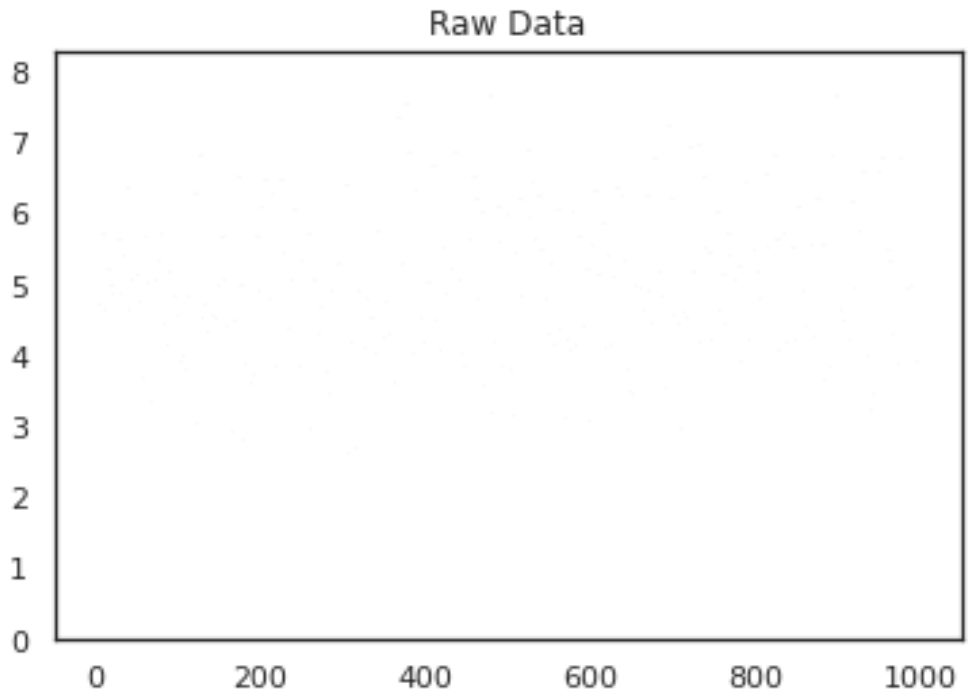
```
import seaborn as sns

import matplotlib.pyplot as plt

sns.set(style="white",color_codes=True)

import pandas as pd

data=pd.read_csv("/games.csv")

data.head()
```

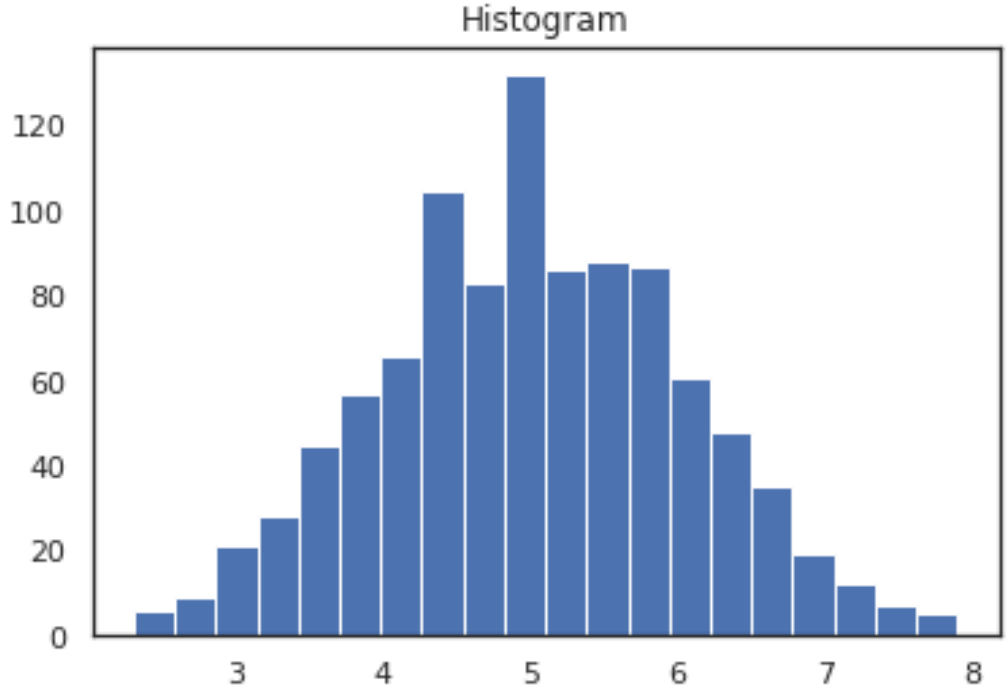
```
         id  rated    created_at  last_move_at  turns victory_status
winner  \
0  TZJHLljE  False  1.504210e+12  1.504210e+12     13      outoftime
white
1  l1NXvwaE   True  1.504130e+12  1.504130e+12     16         resign
black
2  mIICvQHh   True  1.504130e+12  1.504130e+12     61           mate
white
3  kWKvrqYL   True  1.504110e+12  1.504110e+12     61           mate
white
4  9tXo1AUZ   True  1.504030e+12  1.504030e+12     95           mate
white

  increment_code       white_id  white_rating      black_id
black_rating  \
0           15+2       bourgris          1500          a-00
1191
1           5+10           a-00          1322     skinnerua
1261
2           5+10         ischia          1496          a-00
1500
3           20+0  daniamurashov          1439  adivanov2009
1454
4           30+3      nik221107          1523  adivanov2009
1469

                                                 moves opening_eco  \
0  d4 d5 c4 c6 cxd5 e6 dxe6 fxe6 Nf3 Bb4+ Nc3 Ba5...         D10
1  d4 Nc6 e4 e5 f4 f6 dxe5 fxe5 fxe5 Nxe5 Qd4 Nc6...         B00
2  e4 e5 d3 d6 Be3 c6 Be2 b5 Nd2 a5 a4 c5 axb5 Nc...         C20
3  d4 d5 Nf3 Bf5 Nc3 Nf6 Bf4 Ng4 e3 Nc6 Be2 Qd7 O...         D02
4  e4 e5 Nf3 d6 d4 Nc6 d5 Nb4 a3 Na6 Nc3 Be7 b4 N...         C41

                            opening_name  opening_ply
0        Slav Defense: Exchange Variation            5
1  Nimzowitsch Defense: Kennedy Variation            4
2   King's Pawn Game: Leonardis Variation            3
3  Queen's Pawn Game: Zukertort Variation            3
4                      Philidor Defense              5
```

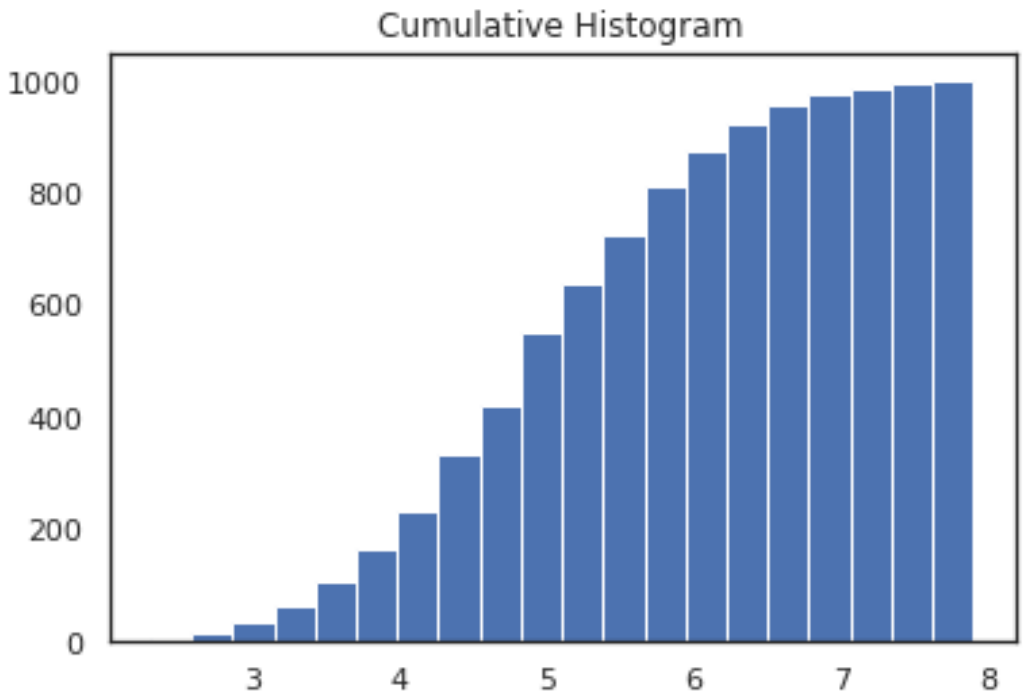
```
data["turns"].value_counts()
```

```
53     303
45     302
51     299
57     297
39     297
      ...
216      1
208      1
176      1
218      1
201      1
Name: turns, Length: 211, dtype: int64
```

```
import matplotlib.pyplot as plt
import numpy as np

# Use numpy to generate a bunch of random data in a bell curve around
5.
n = 5 + np.random.randn(1000)

m = [m for m in range(len(n))]
plt.bar(m, n)
plt.title("Raw Data")
plt.show()

plt.hist(n, bins=20)
plt.title("Histogram")
plt.show()

plt.hist(n, cumulative=True, bins=20)
plt.title("Cumulative Histogram")
plt.show()
```


Raw Data
8
7
6
5
4
3
2
1
0
0
200
400
600
800
1000

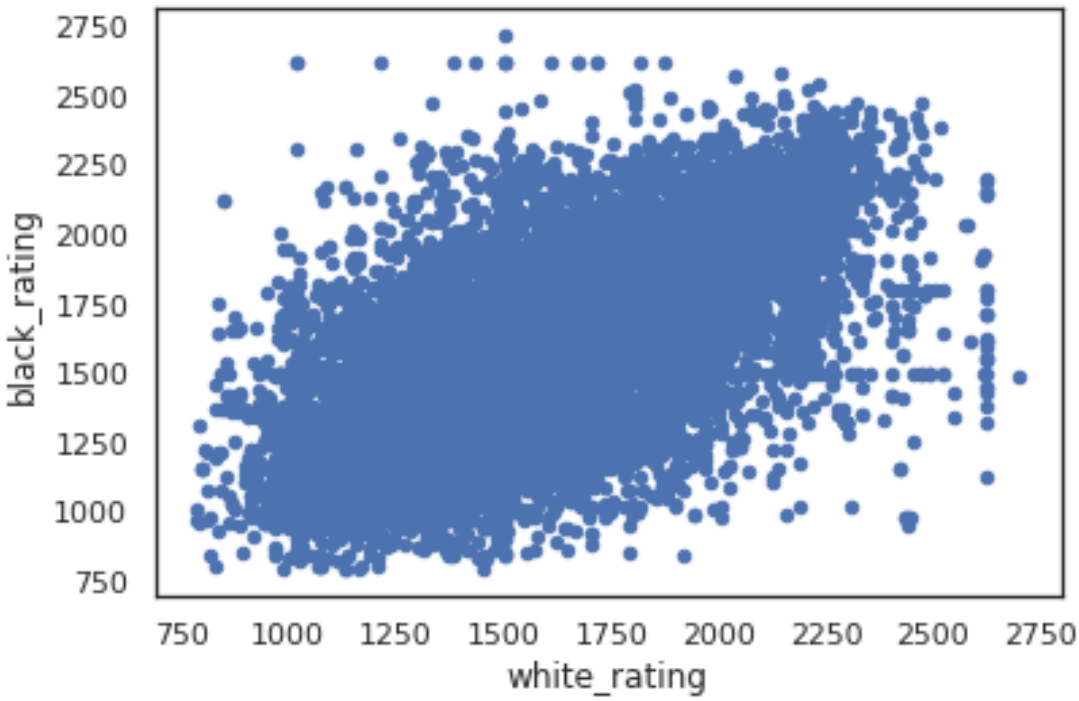

Histogram
120
100
80
60
40
20
0
3
4
5
6
7
8




```
data.plot(kind="scatter",x="white_rating",y="black_rating")
```

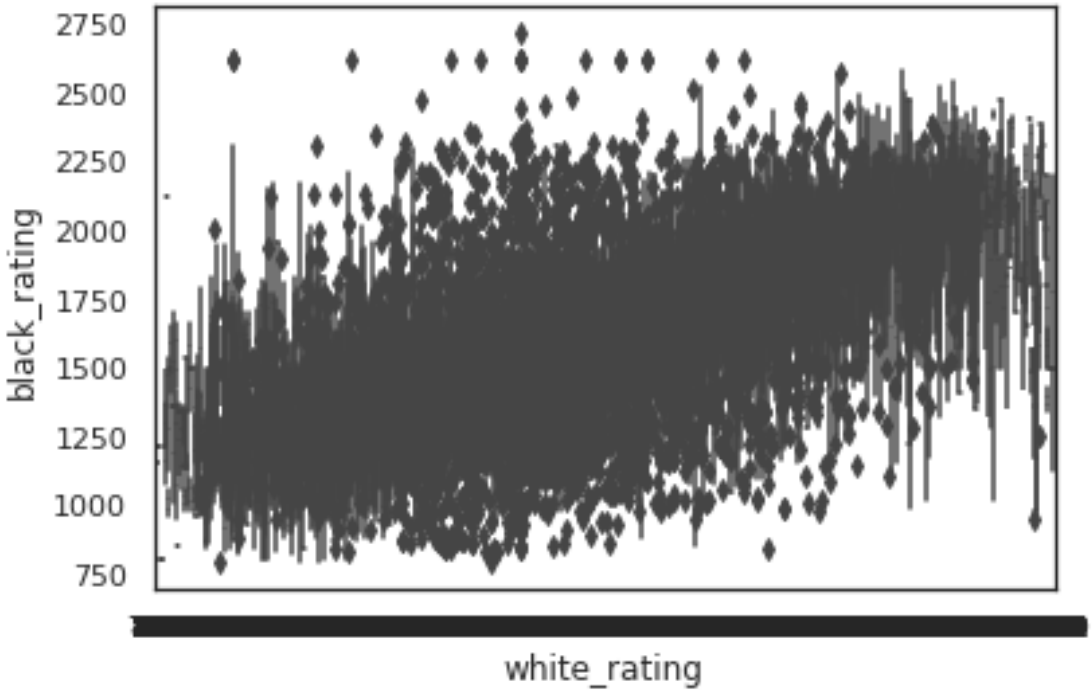
```
*c* argument looks like a single numeric RGB or RGBA sequence, which
should be avoided as value-mapping will have precedence in case its
length matches with *x* & *y*.  Please use the *color* keyword-
argument or provide a 2-D array with a single row if you intend to
specify the same RGB or RGBA value for all points.
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc83913a5d0>
```

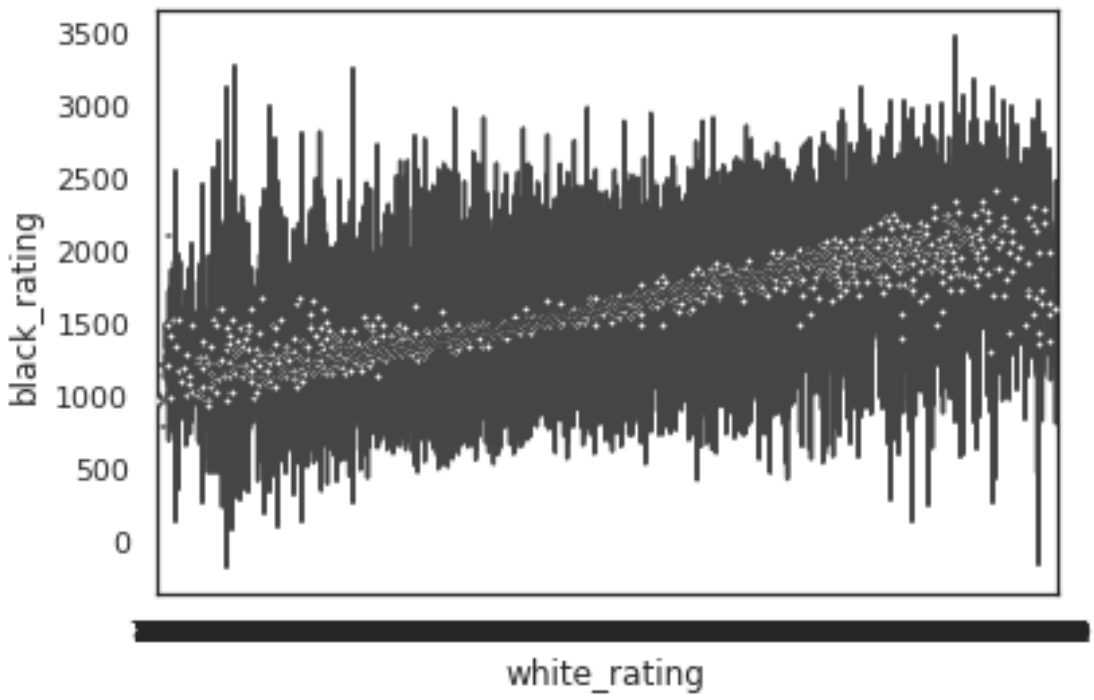



```
sns.boxplot(x="white_rating",y="black_rating",data=data)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc8387f5c50>
```

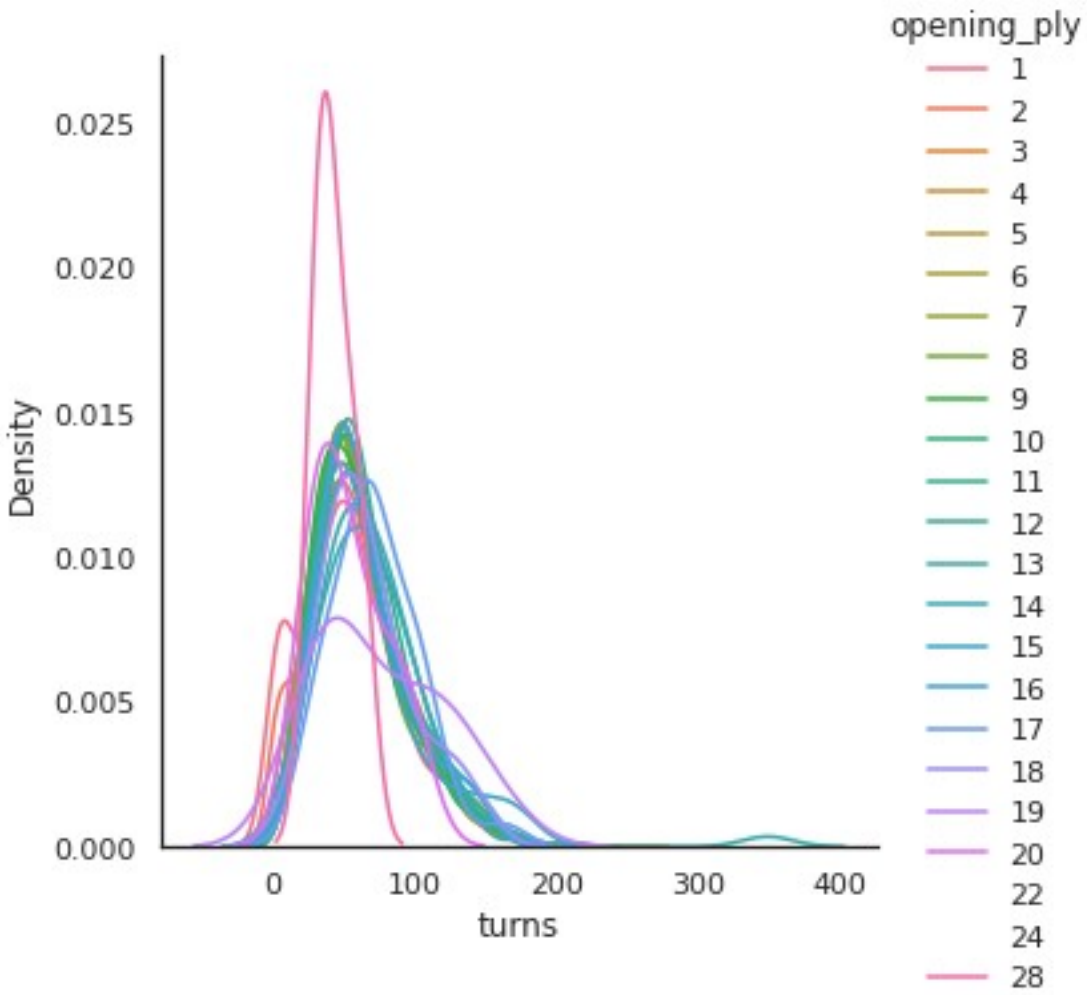



```
sns.violinplot(x="white_rating",y="black_rating",data=data,size=6)
plt.show()
```

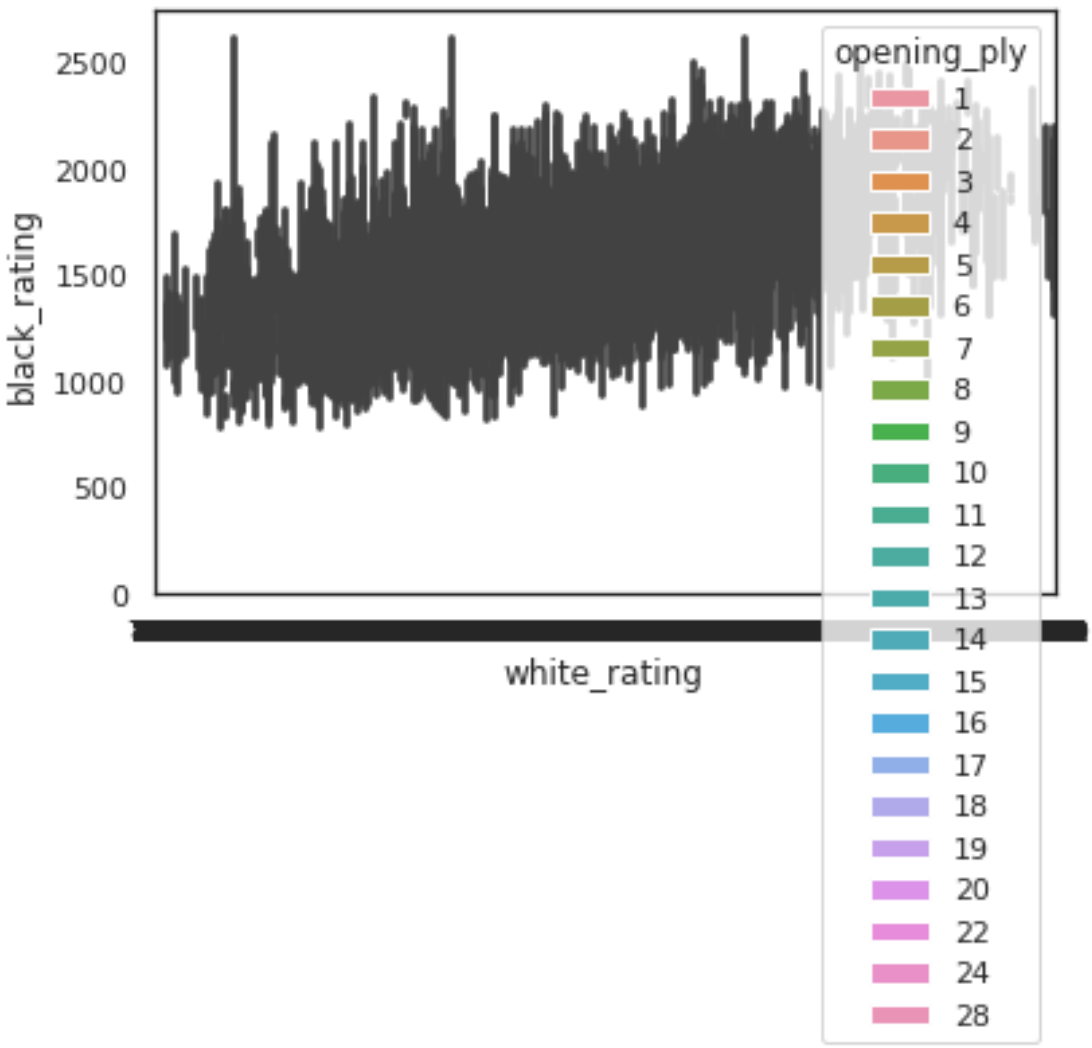



```
sns.FacetGrid(data,hue="opening_ply",size=5).map(sns.kdeplot,"turns").
add_legend()
plt.show()
```

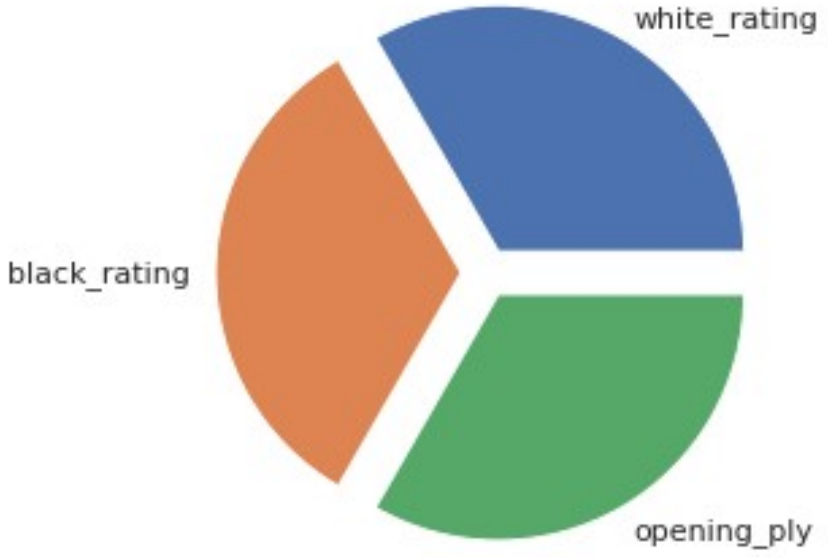
```
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:337:
UserWarning: The `size` parameter has been renamed to `height`; please
update your code.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:316:
UserWarning: Dataset has 0 variance; skipping density estimate. Pass
`warn_singular=False` to disable this warning.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:316:
UserWarning: Dataset has 0 variance; skipping density estimate. Pass
`warn_singular=False` to disable this warning.
  warnings.warn(msg, UserWarning)
```




```
sns.barplot(x="white_rating",y="black_rating",data=data,hue="opening_p
ly")
plt.show()
```




```
labels = ["white_rating","black_rating","opening_ply"]
sizes = [50,50,50]
plt.pie(sizes,labels=labels,explode=(0.1,0.1,0.1))
plt.axis("equal")
plt.show()
```




```
sns.displot(data["white_rating"],bins=25,kde=True)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fc809899990>
```