Data Science & ML Course

Lesson #15 Statistics Fundamentals II

Ivanovitch Silva

November, 2018

Agenda

- Frequency Distributions
  - Sorting frequency distribution tables
  - Percentiles and percentiles ranks
  - Information loss
- Visualizing Distributions
  - Bar, Pie, Histograms plots
  - Skewed distributions
  - Symmetrical Distributions
- Comparing Frequency Distribution

Update from repository

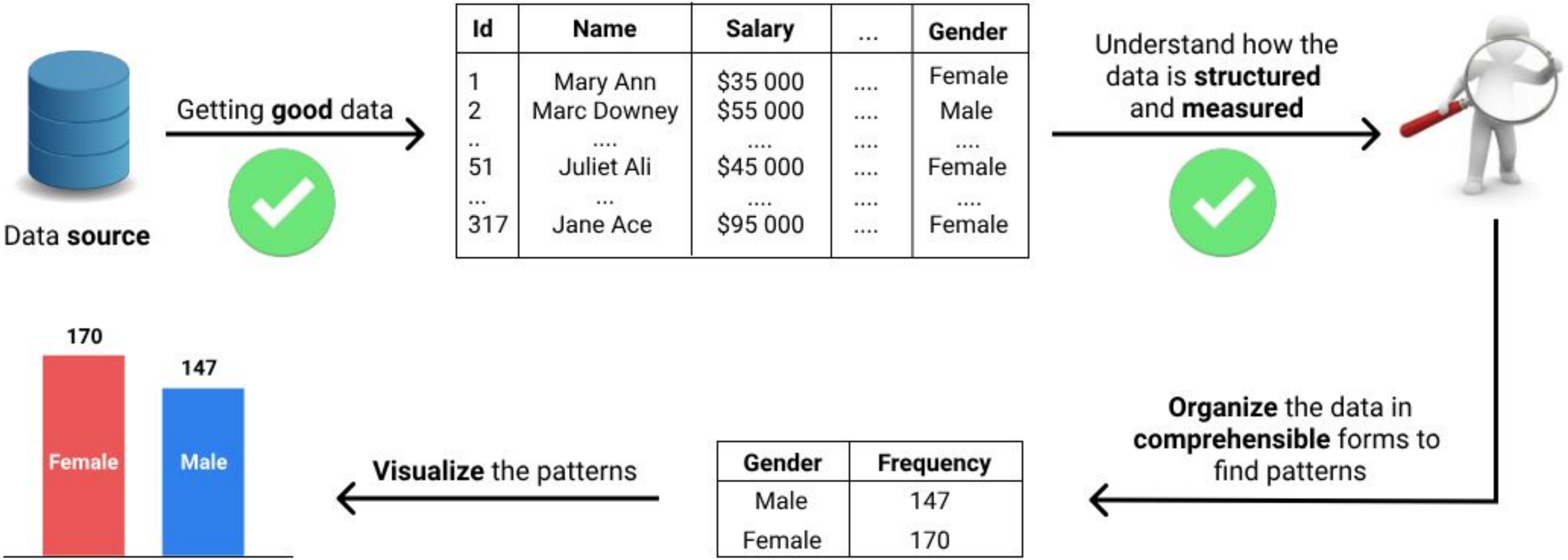
```
git clone https://github.com/ivanovitchm/datascience2machinelearning.git
```

Or ….

```
git pull
```

PREVIOUSLY ON...

(1) collecting data (2) understanding its structure and how it's measured



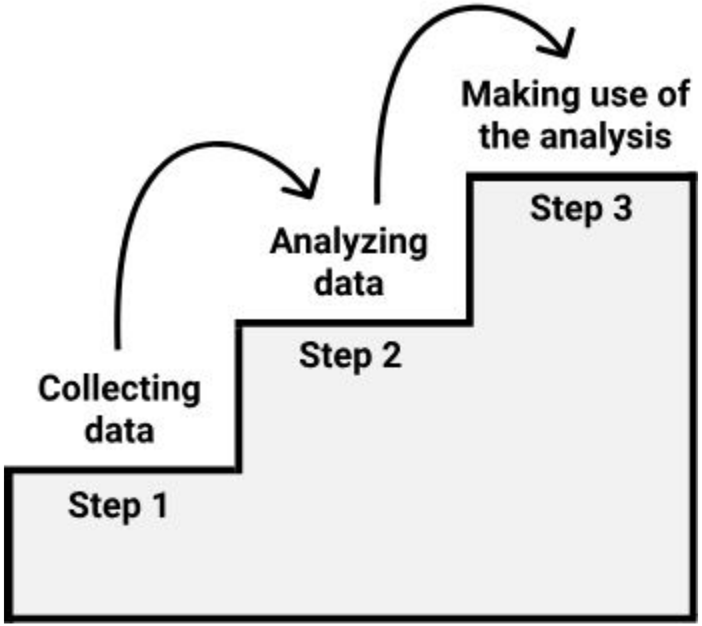
Id	Name	Salary	...	Gender
1	Mary Ann	$35 000	....	Female
2	Marc Downey	$55 000	....	Male
..	....	....	....	....
51	Juliet Ali	$45 000	....	Female
...	...	....	....	....
317	Jane Ace	$95 000	....	Female

Gender	Frequency
Male	147
Female	170

Simplifying Data

We collect data to analyze it, and we analyze it for different purposes:

- To describe phenomena in the world (science).
- To make better decisions (industries).
- To improve systems (engineering).
- To describe different aspects of our society (data journalism); etc.



Frequency Distribution Tables

Our capacity to understand a data set just by looking at it in a table format is limited

Position	Frequency
G	60
F	33
C	25
G/F	13
F/C	12

Unique values

Frequencies

_	Name	Team	Pos	Height	Weight	BMI	Birth_Place	Birthdate	Age	College	Experience	Games Played	MIN
0	Aerial Powers	DAL	F	183	71.0	21.200991	US	January 17, 1994	23	Michigan State	2	8	173
1	Alana Beard	LA	G/F	185	73.0	21.329438	US	May 14, 1982	35	Duke	12	30	947
2	Alex Bentley	CON	G	170	69.0	23.875433	US	October 27, 1990	26	Penn State	4	26	617
3	Alex Montgomery	SAN	G/F	185	84.0	24.543462	US	December 11, 1988	28	Georgia Tech	6	31	721
4	Alexis Jones	MIN	G	175	78.0	25.469388	US	August 5, 1994	23	Baylor	R	24	137

Sorting Frequency Distribution Tables

```
wnba.Pos.value_counts()
```

```
G      60
F      33
C      25
G/F    13
F/C    12
Name: Pos, dtype: int64
```

Nominal Scale

```
wnba.Height.value_counts()
```

```
188    20
193    18
175    16
185    15
191    11
183    11
173    11
196     9
178     8
180     7
170     6
198     5
201     2
168     2
206     1
165     1
Name: Height, dtype: int64
```

How many players are under 170 cm?
How many players are very tall (over 185)?
Are there any players below 160 cm?

Ratio Scale

We can tell the direction of difference

Sorting Tables for Ordinal Variables

The sorting techniques learned in the previous screen can't be used for ordinal scales where the measurement is done using words.

Condition	Label
points <= 20	very few points
20 < points <= 80	few points
80 < points <= 150	many points
points > 150	a lot of points

	Name	PTS	PTS_ordinal_scale
0	Aerial Powers	93	many points
1	Alana Beard	217	a lot of points
2	Alex Bentley	218	a lot of points
3	Alex Montgomery	188	a lot of points
4	Alexis Jones	50	few points

Sorting Tables for Ordinal Variables

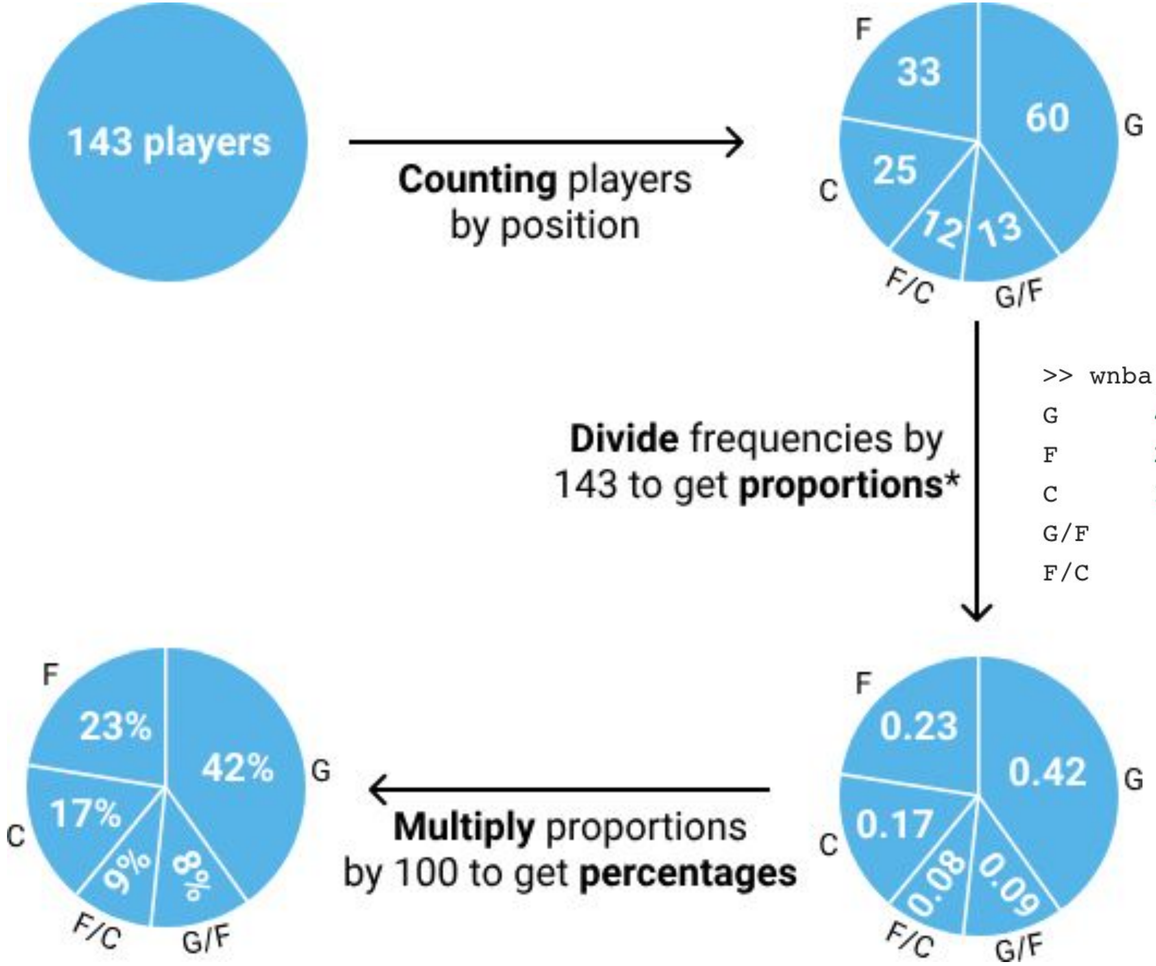
```
>> wnba['PTS_ordinal_scale'].value_counts()
a lot of points     79
few points          27
many points         25
very few points     12
dtype: int64

>> wnba['PTS_ordinal_scale'].value_counts().sort_index()
a lot of points     79
few points          27
many points         25
very few points     12
dtype: int64
```

Sorting Tables for Ordinal Variables

```
>> wnba['PTS_ordinal_scale'].value_counts()
a lot of points     79
few points          27
many points         25
very few points     12
dtype: int64

>> wnba['PTS_ordinal_scale'].value_counts().iloc[[3, 1, 2, 0]]
very few points     12
few points          27
many points         25
a lot of points     79
dtype: int64
```



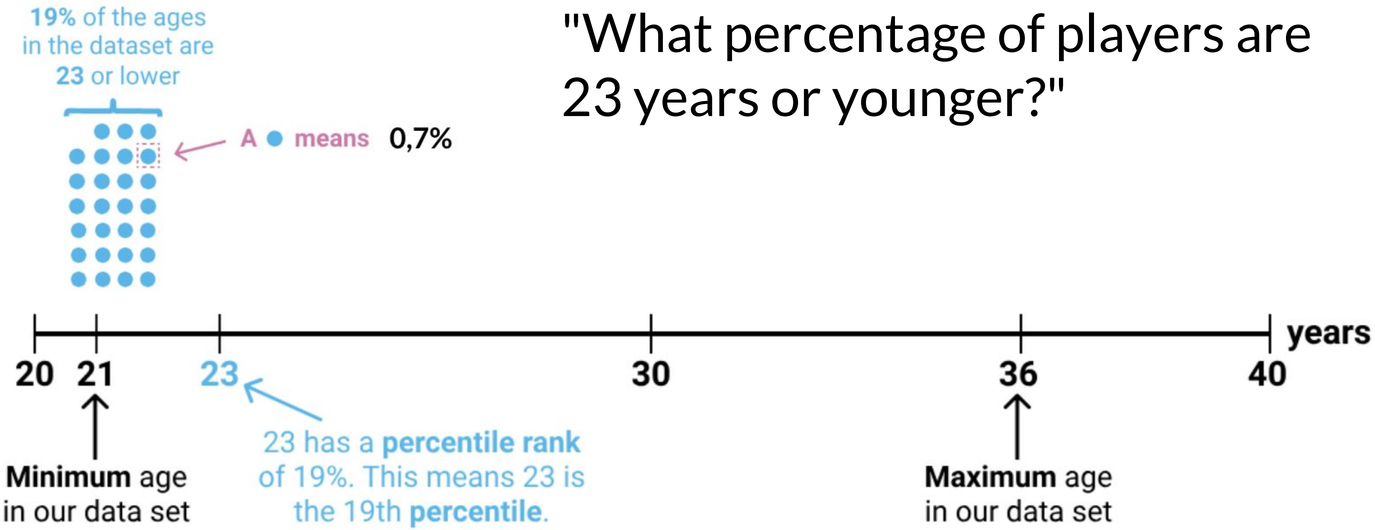

```
>> wnba['Pos'].value_counts(normalize = True) * 100
G       41.958042
F       23.076923
C       17.482517
G/F      9.090909
F/C      8.391608
```

Proportions and Percentages

Percentiles and Percentile Ranks

```
percentages = wnba['Age'].value_counts(normalize = True).sort_index() * 100
percentages
```

21	1.398601
22	6.993007
23	10.489510
24	11.188811
25	10.489510
26	8.391608
27	9.090909
28	9.790210
29	5.594406
30	6.293706
31	5.594406
32	5.594406
33	2.097902
34	3.496503
35	2.797203
36	0.699301



"What percentage of players are 23 years or younger?"

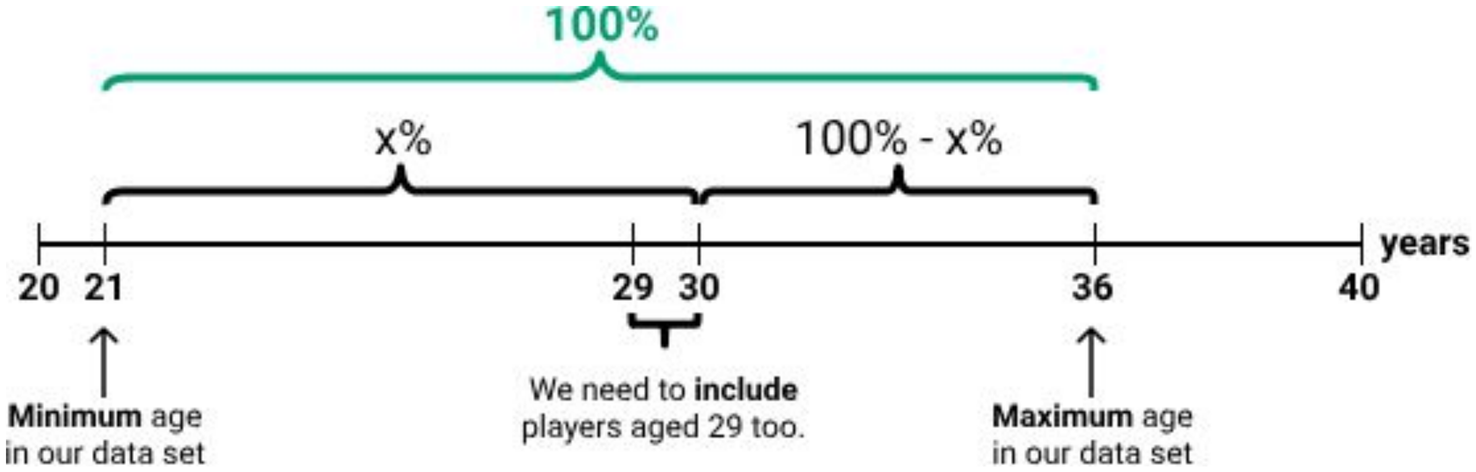
Percentiles and Percentile Ranks

"What percentage of players are 23 years or younger?"

```
>> from scipy.stats import percentileofscore
>> percentileofscore(a = wnba['Age'], score = 23, kind = 'weak')
18.88111888111888
```

Percentiles and Percentile Ranks

"What percentage of players are 30 years or older?"




```
>> 100 - percentileofscore(wnba['Age'], 29, kind = 'weak')
26.573426573426573
```

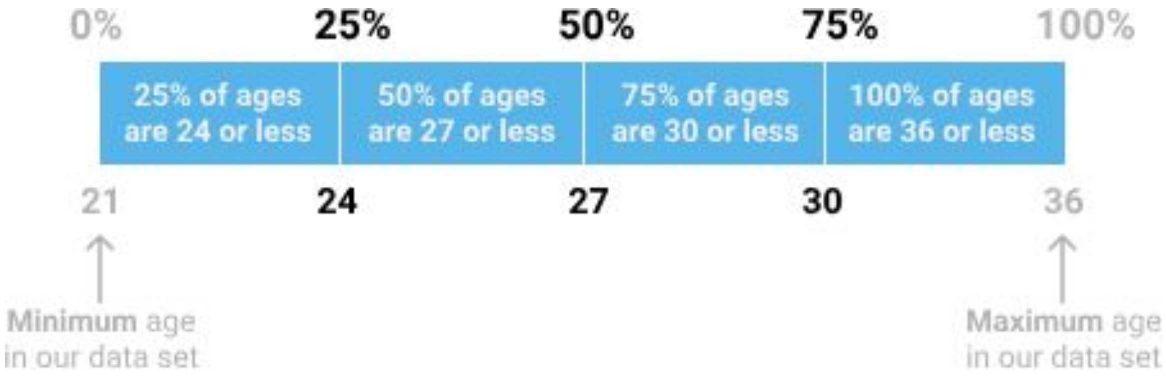
Finding Percentiles with Pandas

```
>> wnba['Age'].describe()
count     143.000000
mean       27.076923
std         3.679170
min        21.000000
25%        24.000000
50%        27.000000
75%        30.000000
max        36.000000
Name: Age, dtype: float64
```

```
>>wnba['Age'].describe().iloc[3:]
min     21.0
25%     24.0
50%     27.0
75%     30.0
max     36.0
Name: Age, dtype: float64
```

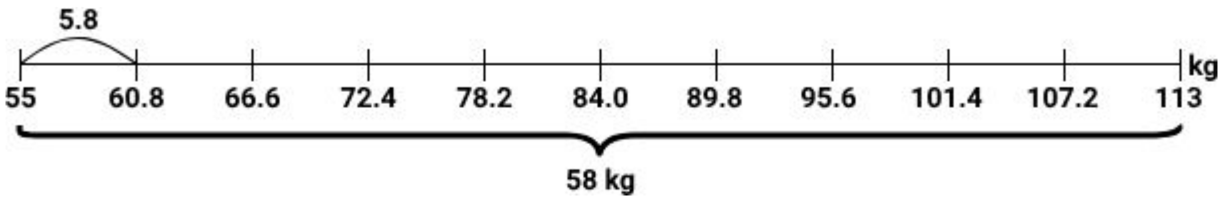
Finding Percentiles with Pandas

```
>> wnba['Age'].describe(percentiles = [.1, .15, .33, .5, .592, .85, .9]).iloc[3:]
min        21.0
10%        23.0
15%        23.0
33%        25.0
50%        27.0
59.2%      28.0
85%        31.0
90%        32.0
max        36.0
Name: Age, dtype: float64
```



Grouped Frequency Distribution Tables

```
>> wnba['Weight'].value_counts(bins = 10).sort_index()
(54.941, 60.8]      5
(60.8, 66.6]       21
(66.6, 72.4]       10
(72.4, 78.2]       33
(78.2, 84.0]       31
(84.0, 89.8]       24
(89.8, 95.6]       10
(95.6, 101.4]       3
(101.4, 107.2]      2
(107.2, 113.0]      3
```

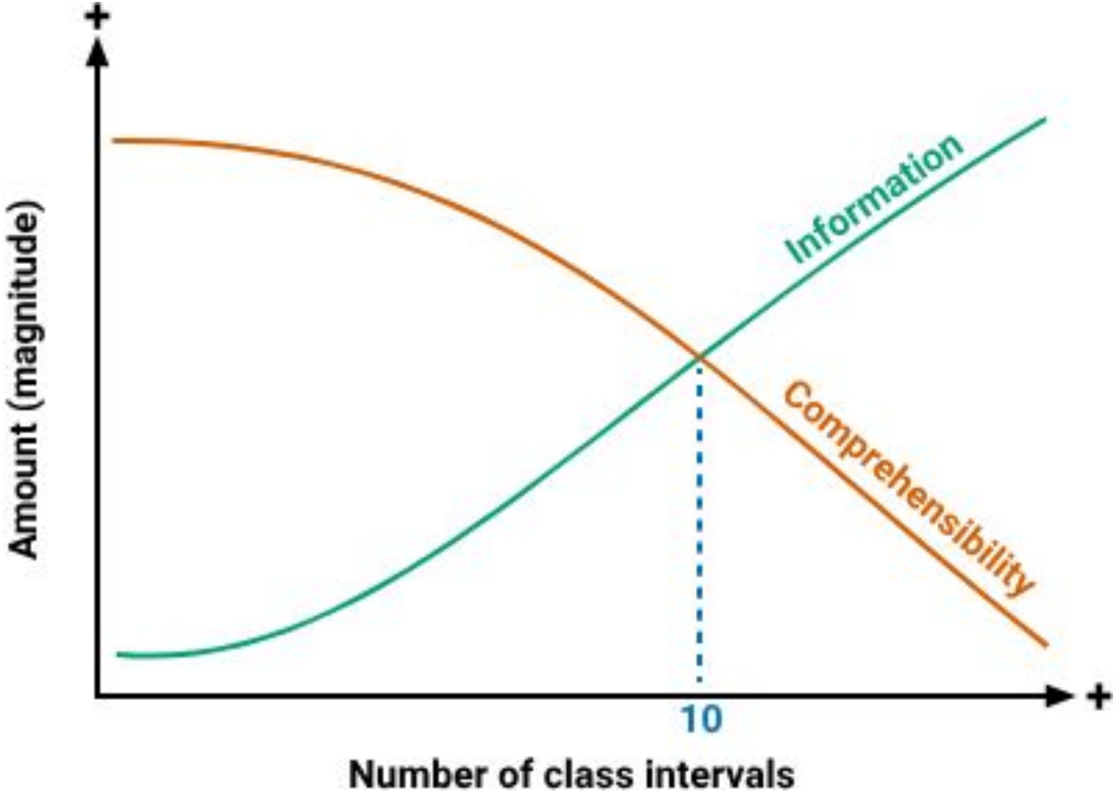


Information Loss

```
>> wnba['PTS'].value_counts(bins = 10)
(1.417, 60.2]     30
(60.2, 118.4]     24
(118.4, 176.6]    17
(176.6, 234.8]    20
(234.8, 293.0]    17
(293.0, 351.2]     8
(351.2, 409.4]    10
(409.4, 467.6]     8
(467.6, 525.8]     4
(525.8, 584.0]     5
Name: PTS, dtype: int64
```

```
wnba['PTS'].value_counts(bins = 5).sort_index()
(1.417, 118.4]    54
(118.4, 234.8]    37
(234.8, 351.2]    25
(351.2, 467.6]    18
(467.6, 584.0]     9
```

Information Loss



Readability for Grouped Frequency Tables

```
wnba['PTS'].value_counts(bins = 5).sort_index()
(1.417, 118.4]     54
(118.4, 234.8]     37
(234.8, 351.2]     25
(351.2, 467.6]     18
(467.6, 584.0]      9
Name: PTS, dtype: int64
```

```
(0,100]       49
(100,200]     28
(200,300]     32
(300,400]     17
(400,500]     10
(500,600]      7
```

Readability for Grouped Frequency Tables

```
>> intervals = pd.interval_range(start = 0, end = 600, freq = 100)
>> intervals
IntervalIndex([(0, 100], (100, 200], (200, 300], (300, 400], (400, 500], (500, 600]]
              closed='right',
              dtype='interval[int64]')
```

```
>> gr_freq_table = pd.Series([0,0,0,0,0,0], index = intervals)
>> gr_freq_table
(0, 100]      0
(100, 200]    0
(200, 300]    0
(300, 400]    0
(400, 500]    0
(500, 600]    0
dtype: int64
```

```
>> for value in wnba['PTS']:
       for interval in intervals:
           if value in interval:
               gr_freq_table.loc[interval] += 1
               break
```

```
>> gr_freq_table
(0, 100]      49
(100, 200]    28
(200, 300]    32
(300, 400]    17
(400, 500]    10
(500, 600]     7
```

Data **source**

Getting **good** data

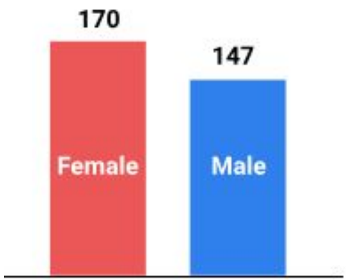
Id	Name	Salary	...	Gender
1	Mary Ann	$35 000	....	Female
2	Marc Downey	$55 000	....	Male
..	....	....	....	....
51	Juliet Ali	$45 000	....	Female
...	...	....	....	....
317	Jane Ace	$95 000	....	Female

Understand how the data is **structured** and **measured**

Organize the data in **comprehensible** forms to find patterns

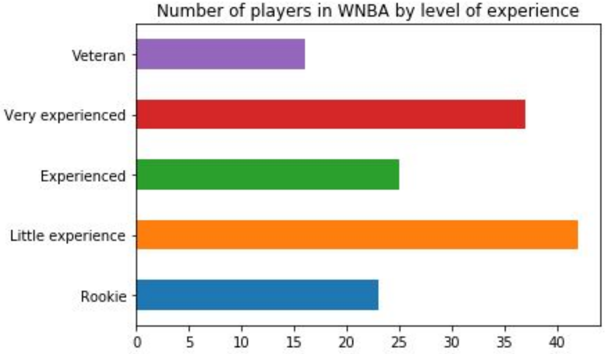
Gender	Frequency
Male	147
Female	170

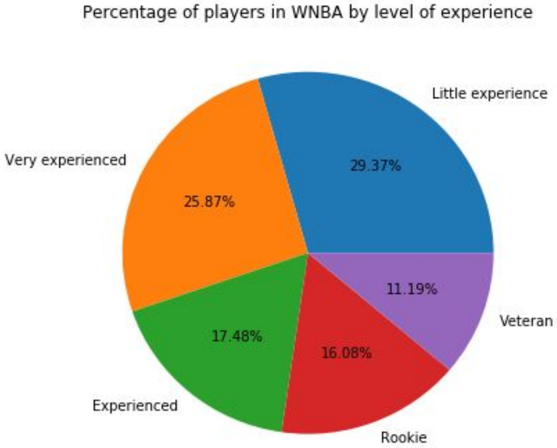
Visualize the patterns

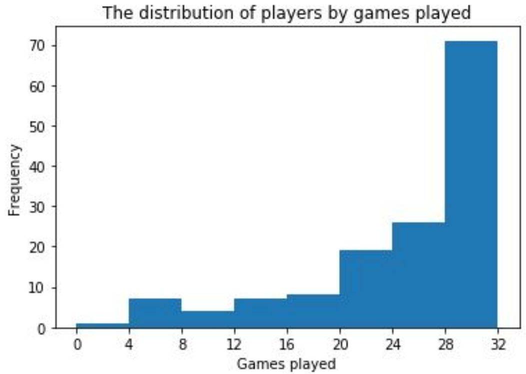


Lesson 15 Statistical Fundamentals II.ipynb
Section 1

Visualizing Distributions



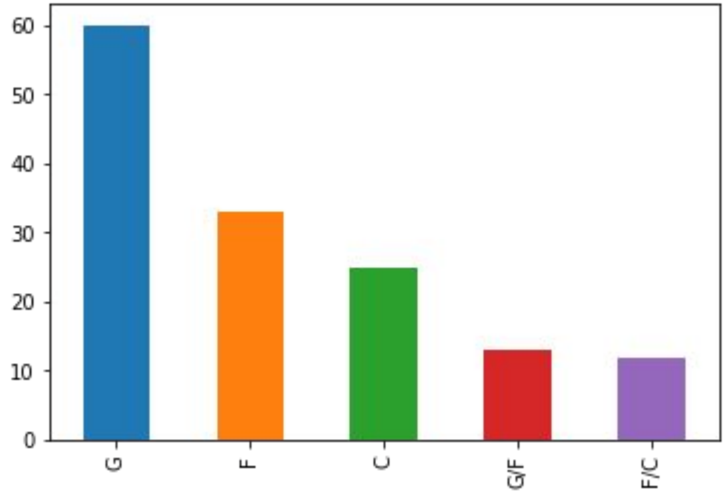




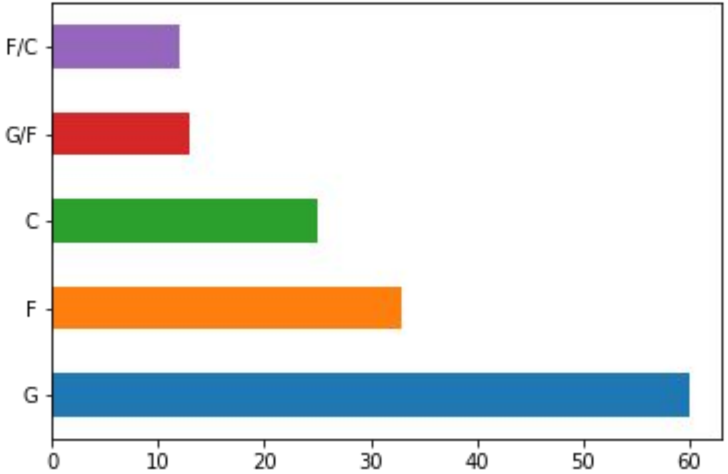
Graphs make easy to scan and compare frequencies, providing us with a single picture of the entire distribution of a variable (**nominal** or **ordinal scale**)

Bar Plots

horizontal bar plots are ideal to use when the labels of the unique values are long



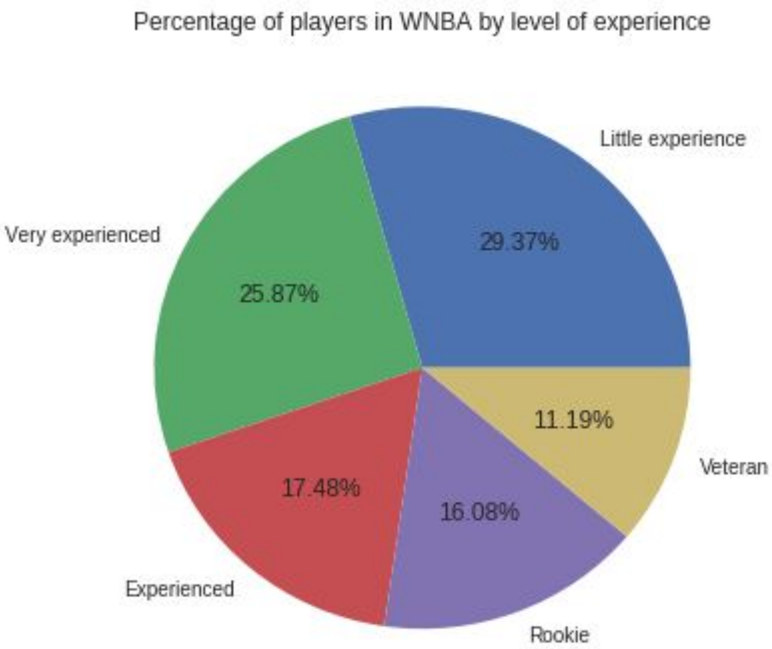

```
wnba['Pos'].value_counts().plot.bar()
```




```
wnba['Pos'].value_counts().plot.barh()
```

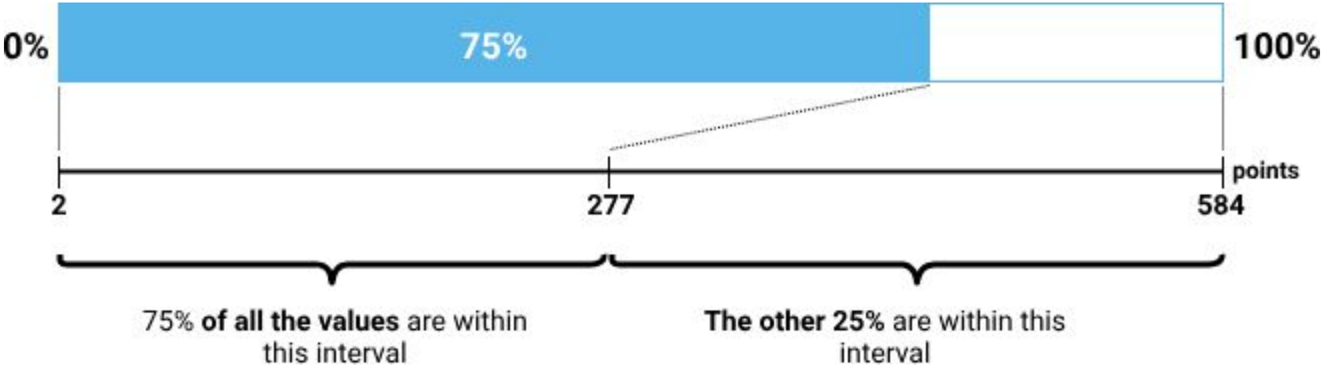
Pie Charts

Pie Charts




```
wnba['Exp_ordinal'].value_counts().\
plot.pie(figsize = (6,6),
         autopct = '%.2f%%',
         title = 'Percentage of players in \
         WNBA by level of experience')
plt.ylabel('')
```

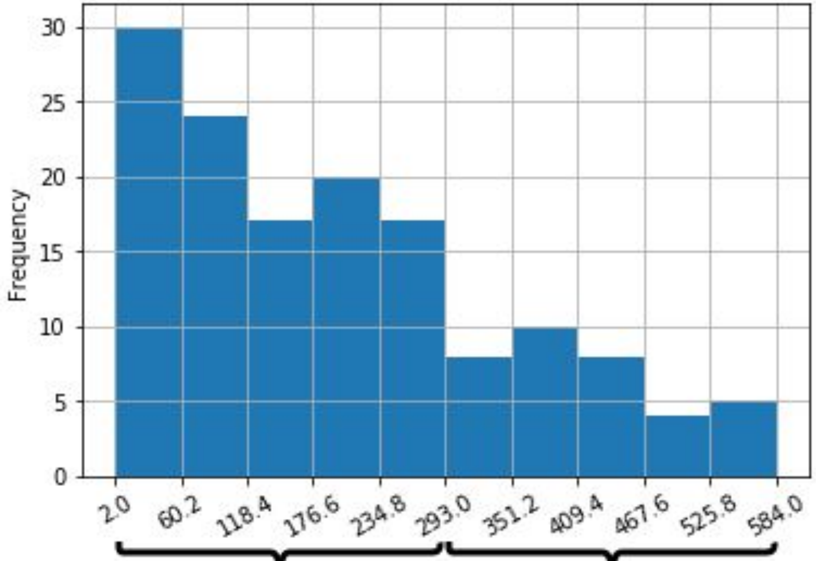
Histograms




```
>> wnba['PTS'].describe()
count    143.000000
mean     201.790210
std      153.381548
min        2.000000
25%       75.000000
50%      177.000000
75%      277.500000
max      584.000000
```

We can see that 75% of the values are distributed within a relatively narrow interval (between 2 and 277), while the remaining 25% are distributed in an interval that's slightly larger.

The Statistics Behind Histograms



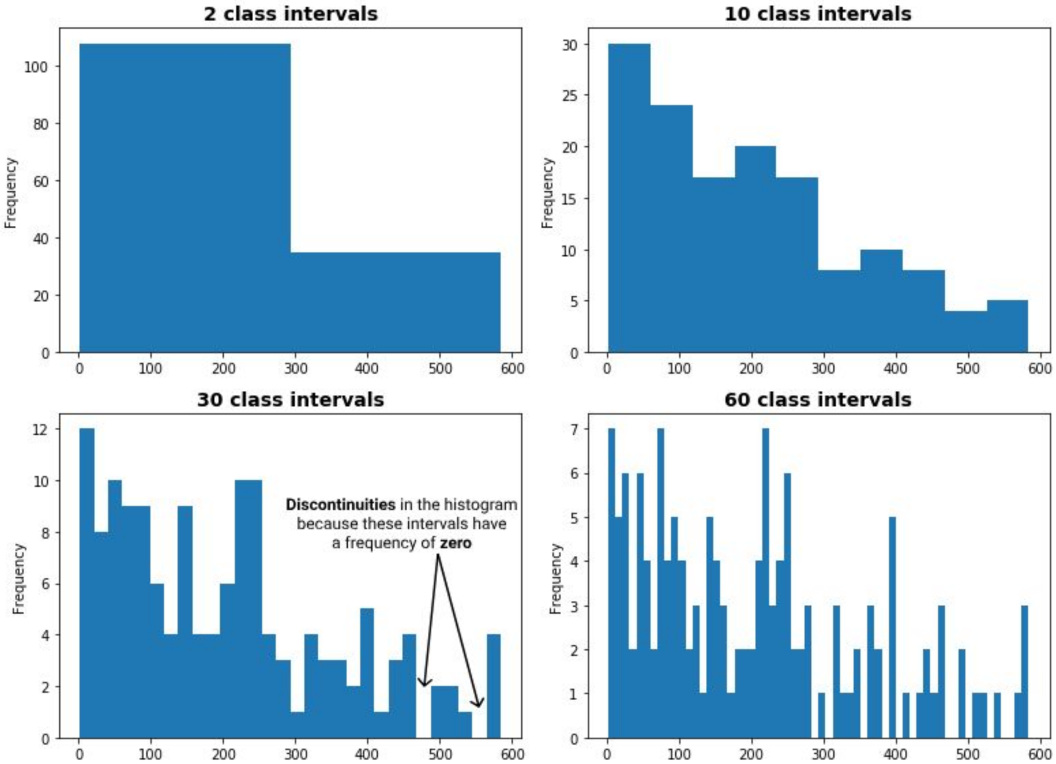
We can **see immediately** that roughly **three quarters (75%)** of the values are within this interval

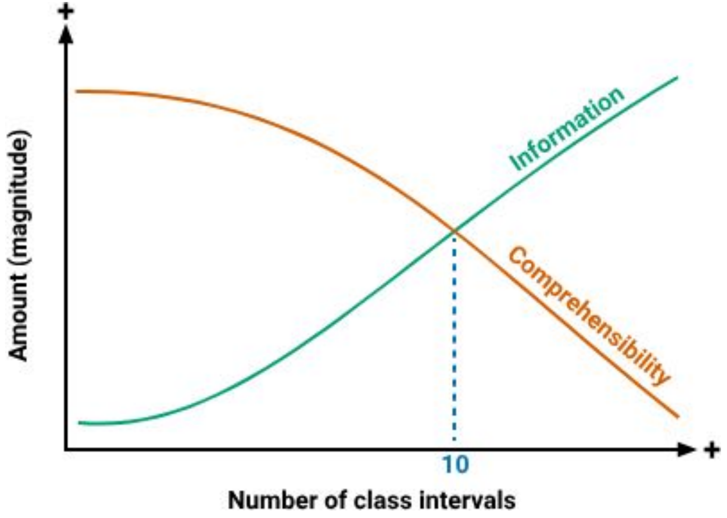
The **remaining quarter** is within this interval

```
>> wnba['PTS'].describe()
count     143.000000
mean      201.790210
std       153.381548
min         2.000000
25%        75.000000
50%       177.000000
75%       277.500000
max       584.000000
Name: PTS, dtype: float64
```

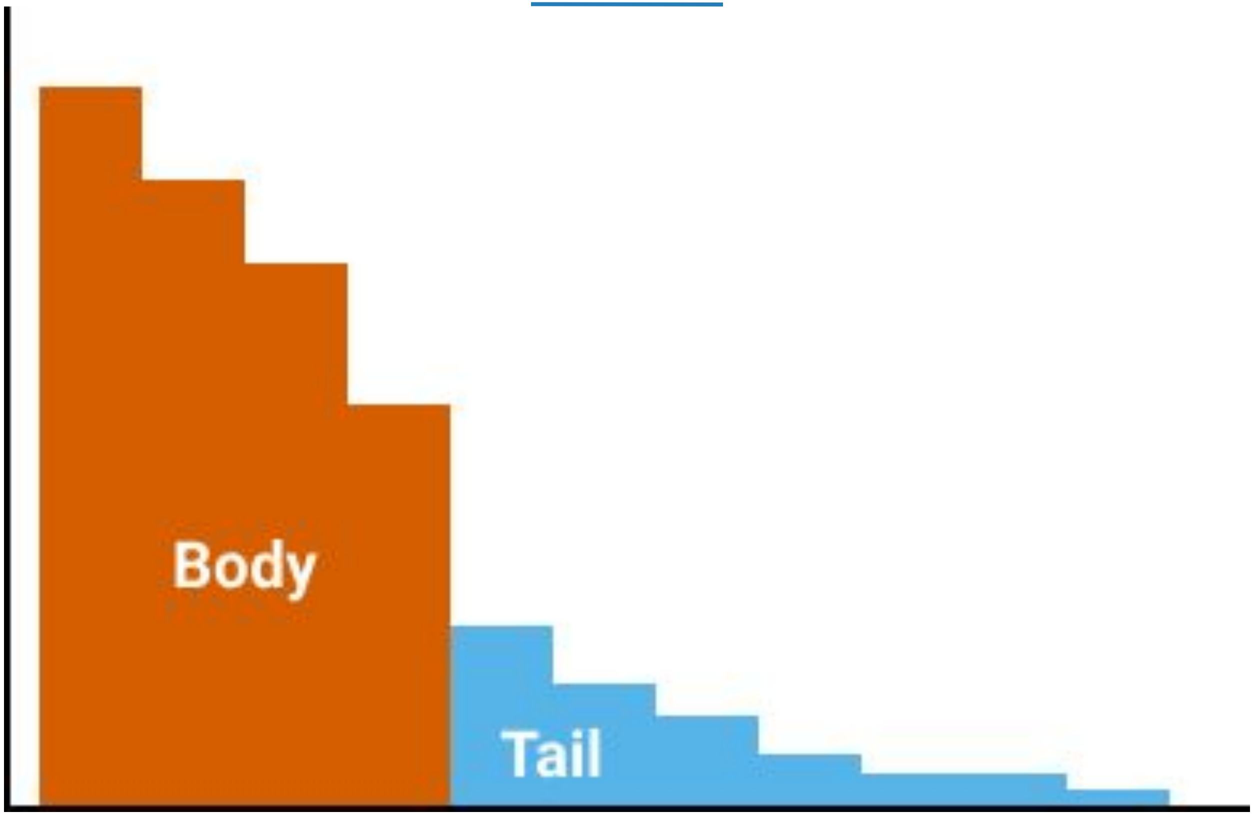
```
>> wnba['PTS'].plot.hist()
```

Binning for Histograms

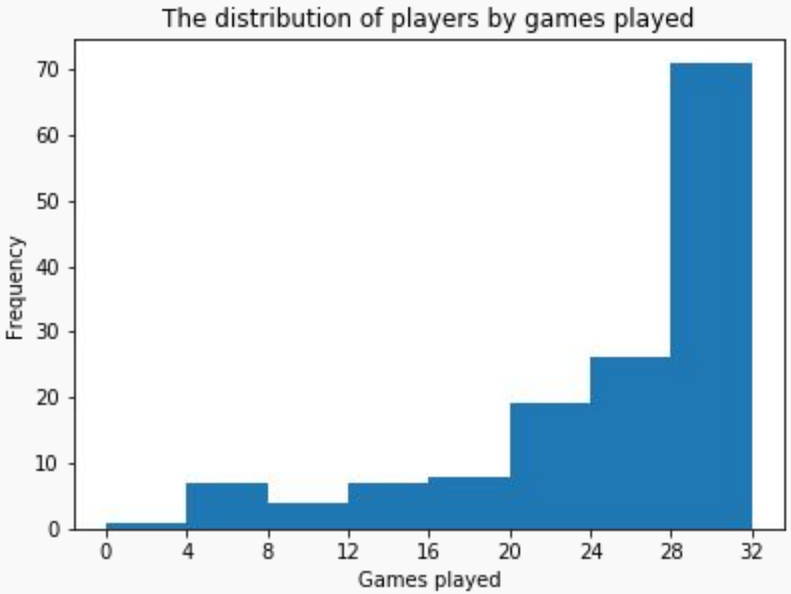


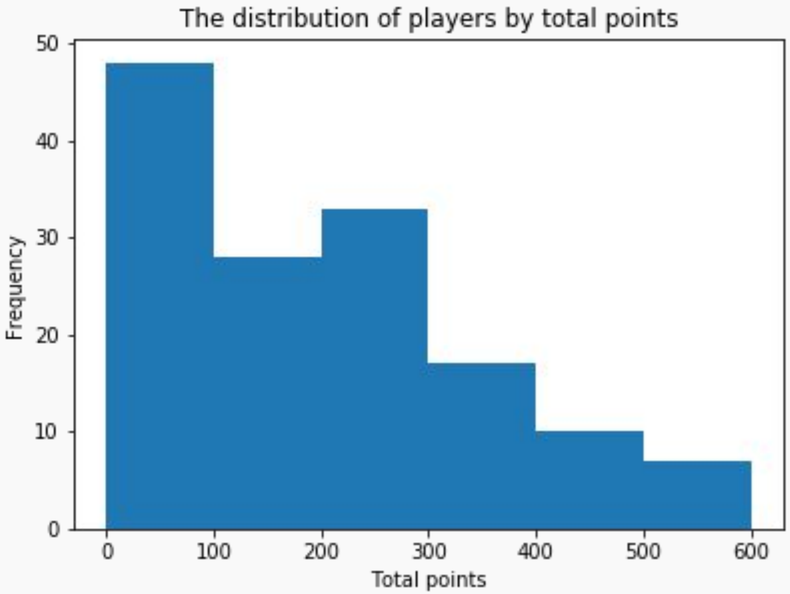


Skewed Distributions

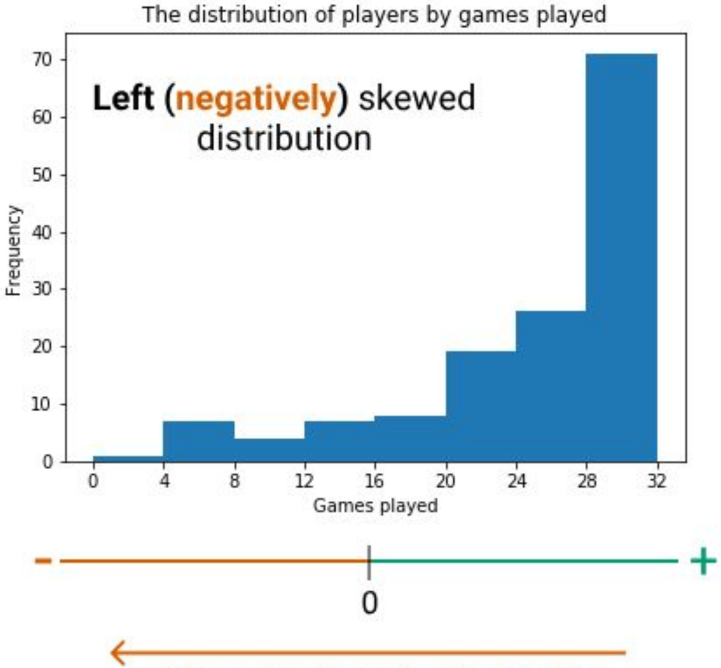


Skewed Distributions

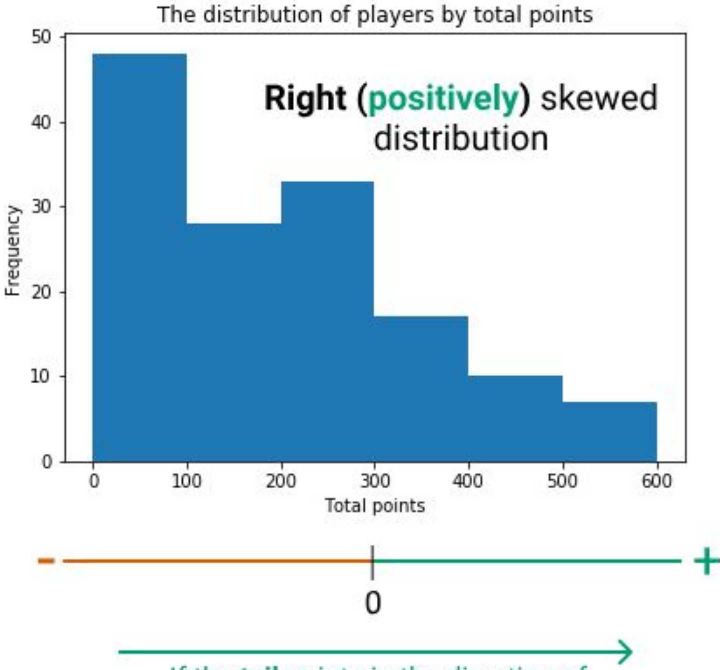




Skewed Distributions

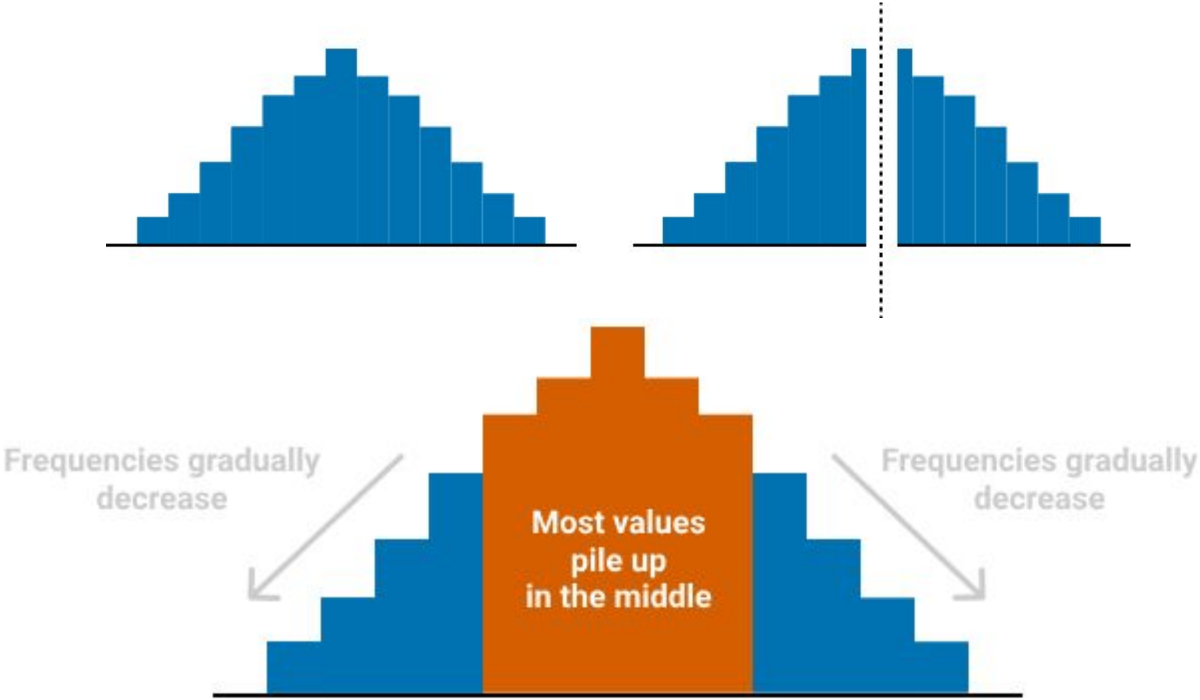


If the **tail** points in the direction of **negative numbers** then the distribution is **negatively skewed**

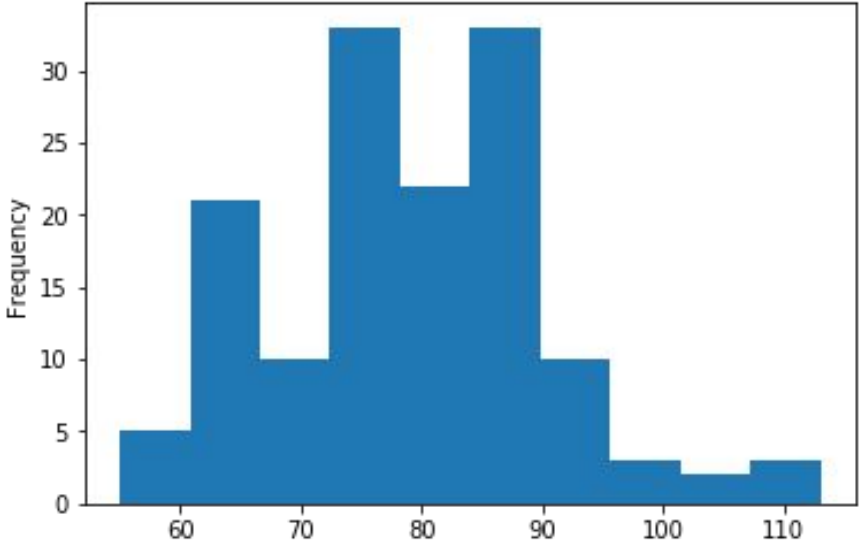


If the **tail** points in the direction of **positive numbers** then the distribution is **positevely skewed**

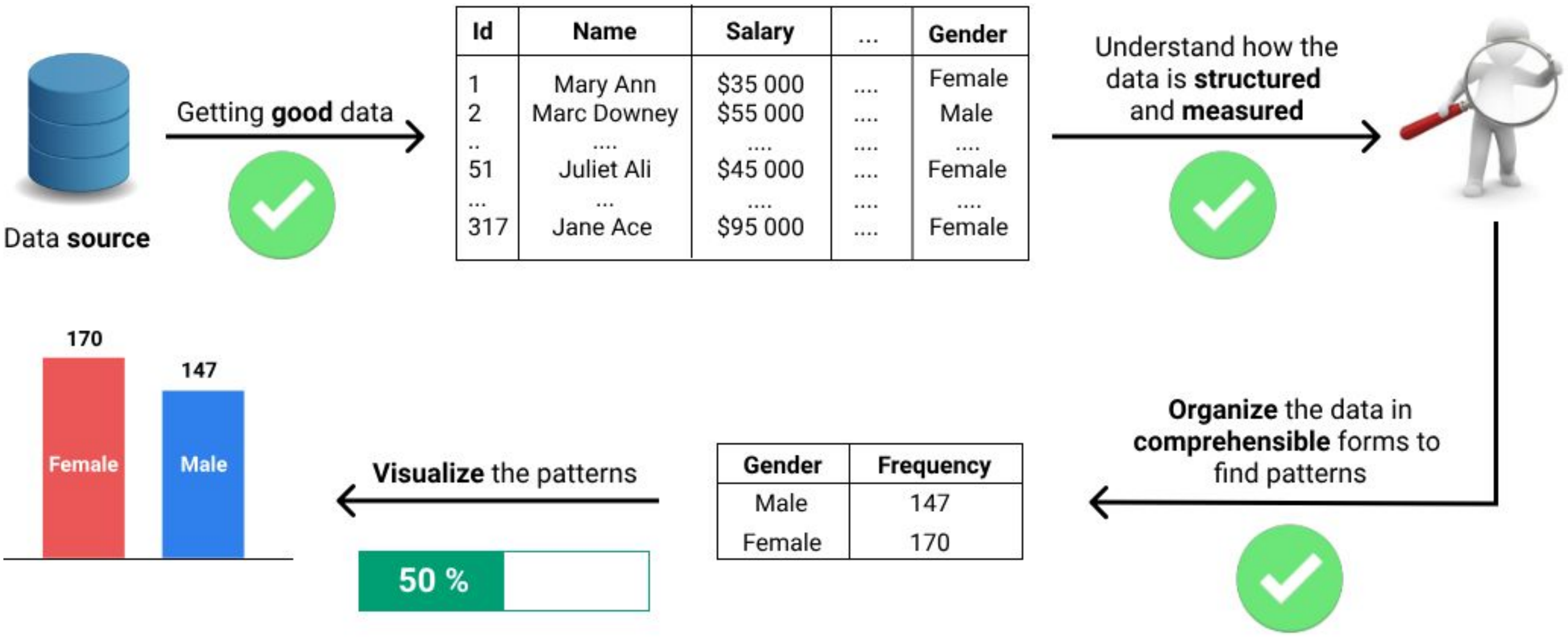
Symmetrical Distributions



Symmetrical Distribution (uniform)

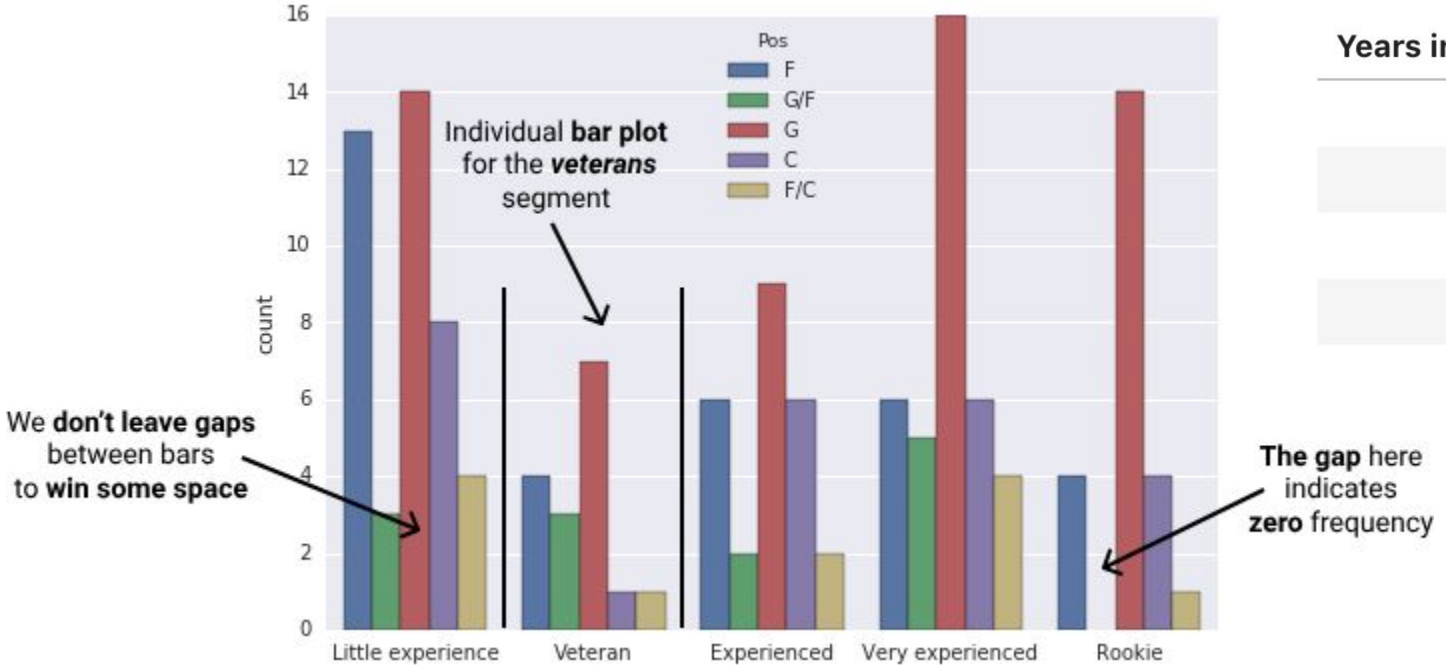


Scale of measurement	Graphs we can use to show the distribution
Nominal	
Ordinal	
Interval	
Ratio	


Data source
Getting good data
Id | Name | Salary | .... | Gender
1 | Mary Ann | $35 000 | .... | Female
2 | Marc Downey | $55 000 | .... | Male
.. | .... | .... | .... | ....
51 | Juliet Ali | $45 000 | .... | Female
... | ... | .... | .... | ....
317 | Jane Ace | $95 000 | .... | Female
Understand how the data is structured and measured
Organize the data in comprehensible forms to find patterns
Gender | Frequency
Male | 147
Female | 170
Visualize the patterns
50 %
170
147
Female
Male

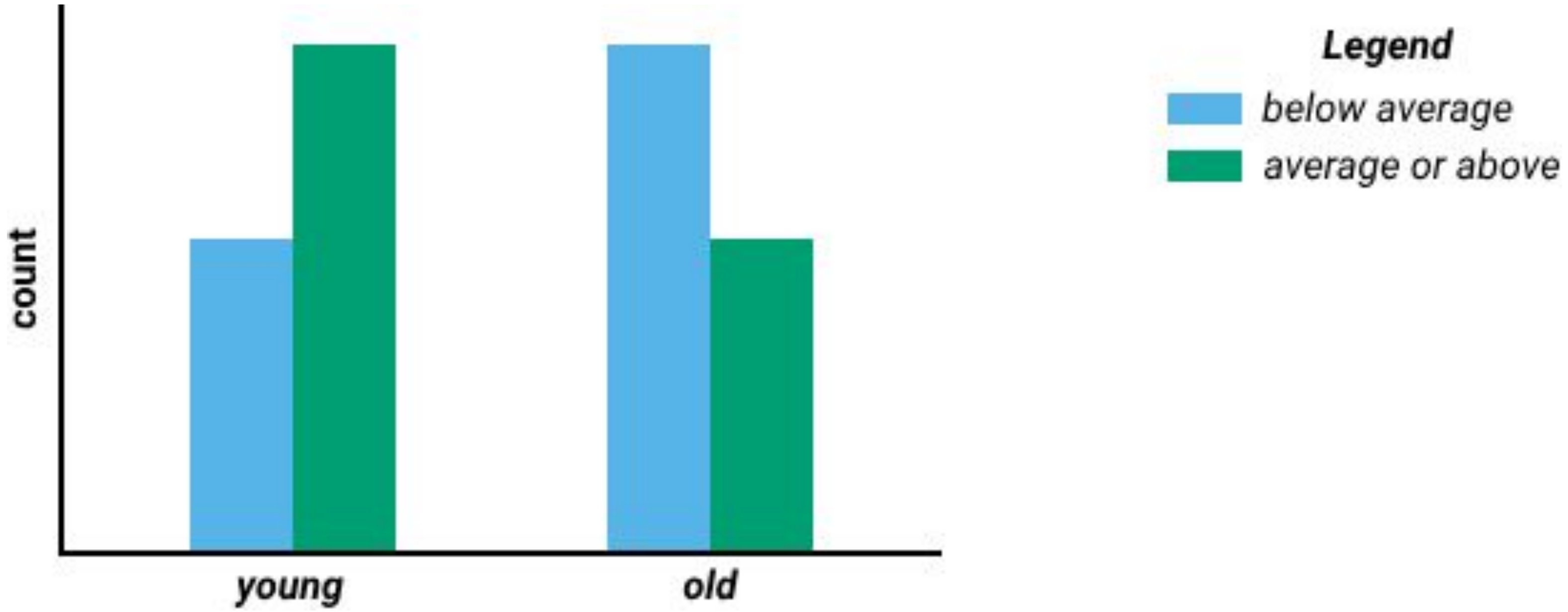
Lesson 15 Statistical Fundamentals II.ipynb
Section 2

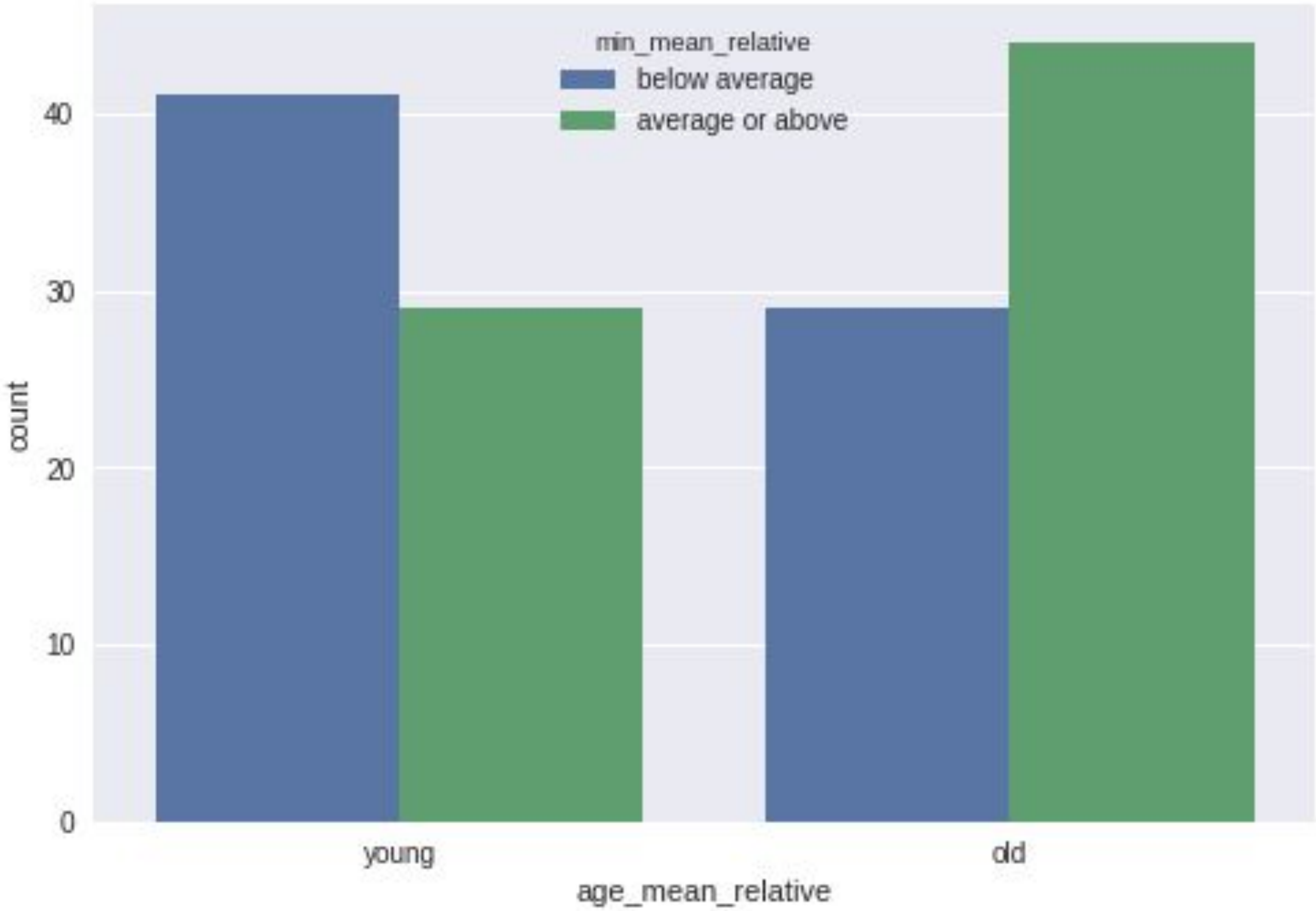
Comparing Frequency Distribution



Years in WNBA	Label
0	Rookie
1-3	Little experience
4-5	Experienced
5-10	Very experienced
>10	Veteran

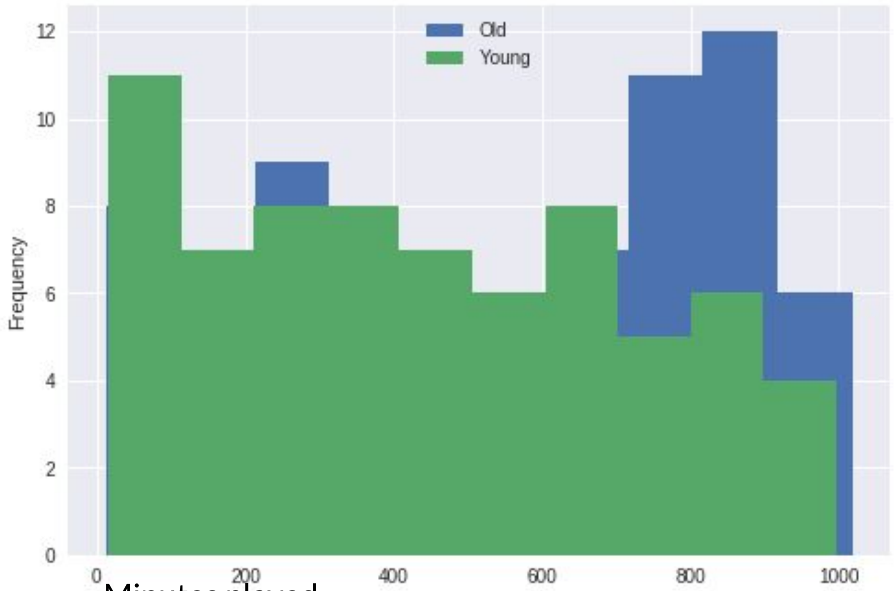
Challenge: Do older players play less?





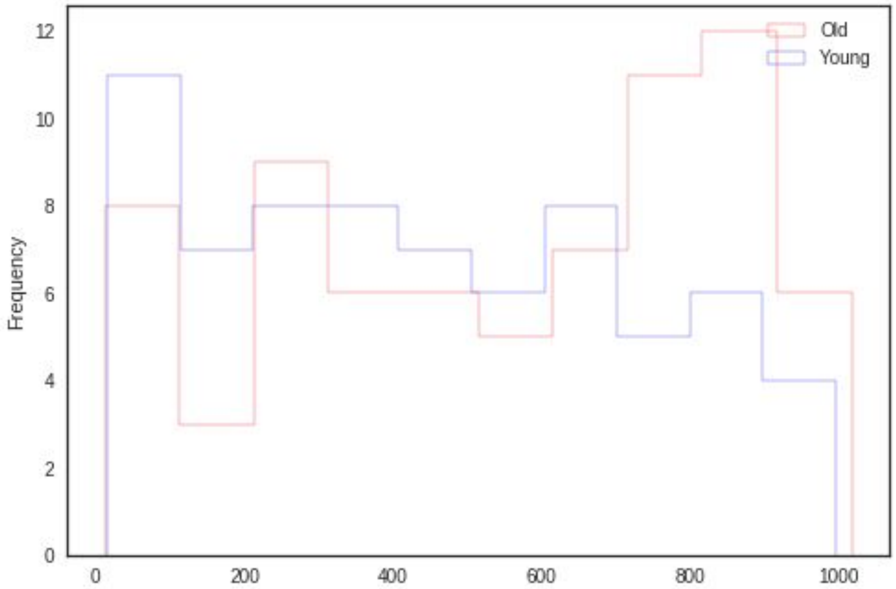

```
sns.countplot(x = 'age_mean_relative', hue = 'min_mean_relative', data = wnba)
```

Comparing Histograms



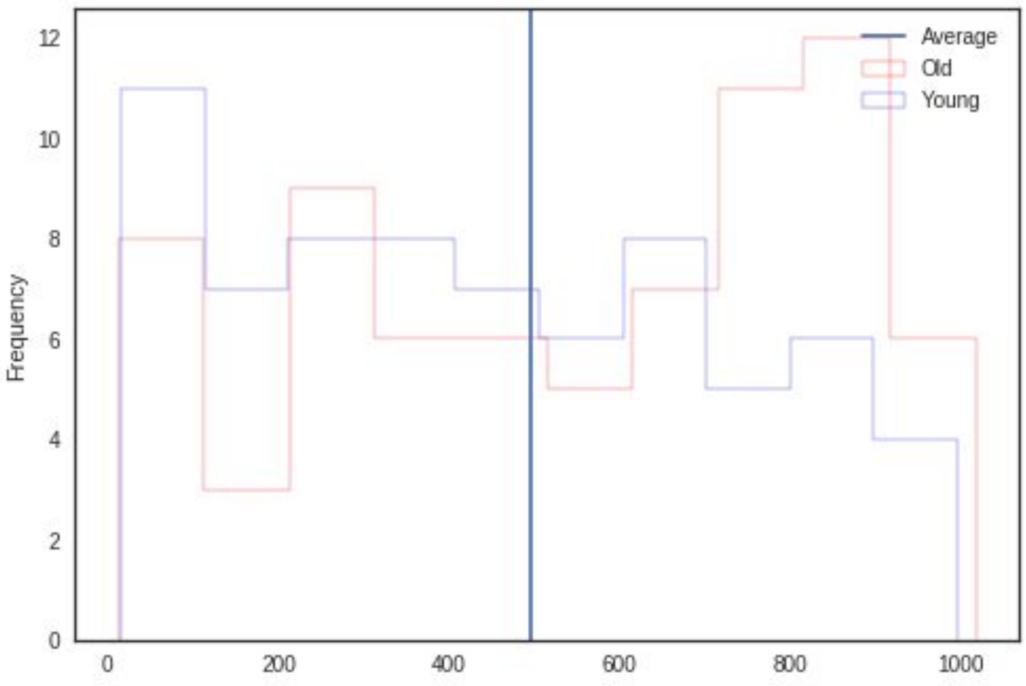
Minutes played

```
wnba[wnba.Age >= 27]['MIN'].plot.hist(label = 'Old', legend = True)
wnba[wnba.Age < 27]['MIN'].plot.hist(label = 'Young', legend = True)
```



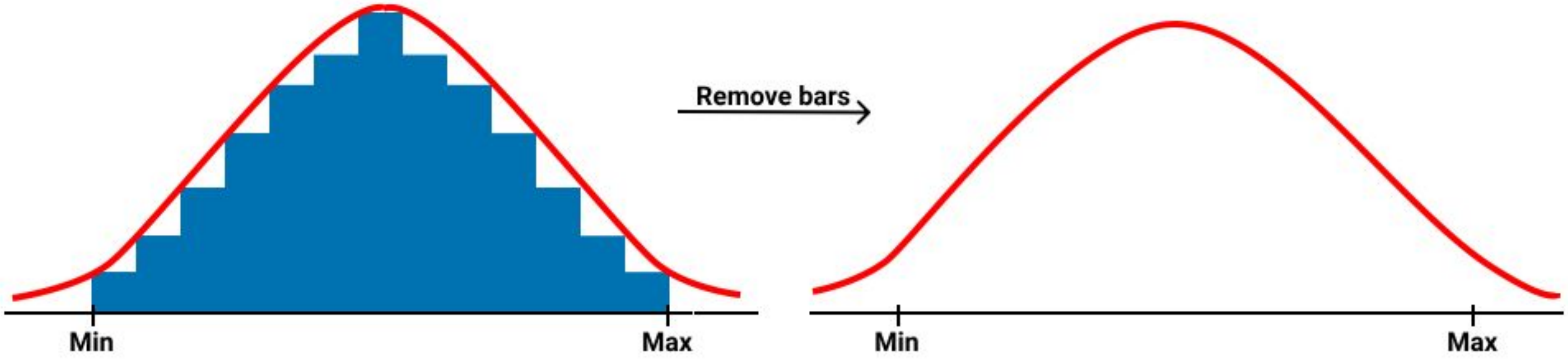

```
sns.set_style("white")
wnba[wnba.Age >= 27]['MIN'].plot.hist(histtype = 'step',
                                      label = 'Old',
                                      legend = True,color="red")
```

Comparing Histograms

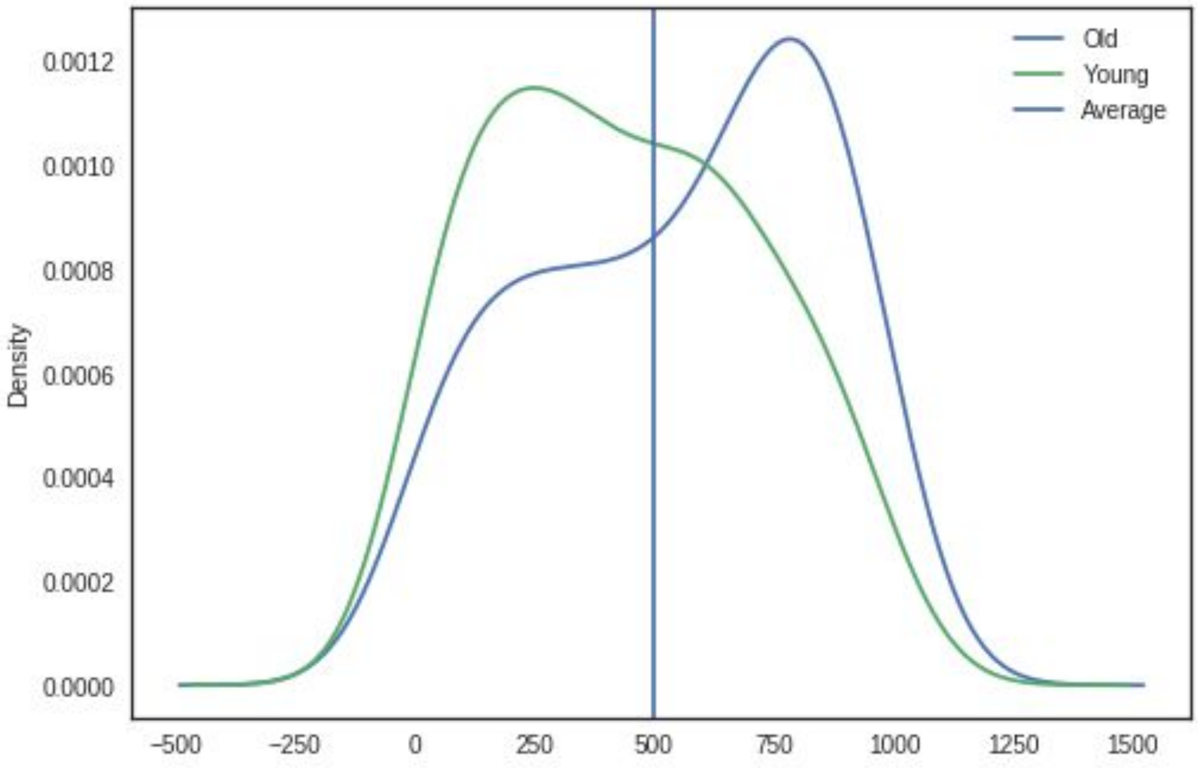



```
plt.axvline(497, label = 'Average')
```

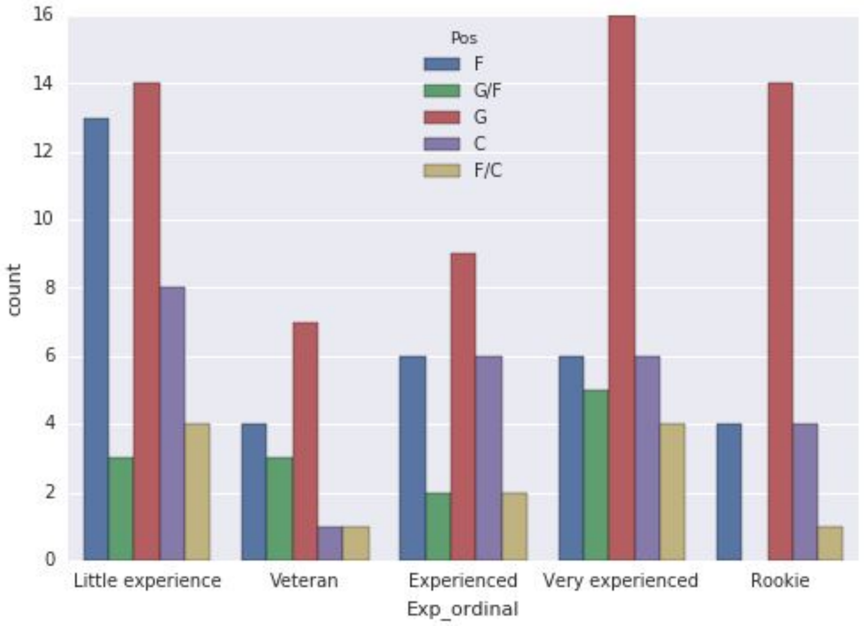
Kernel Density Estimate (KDE) Plots

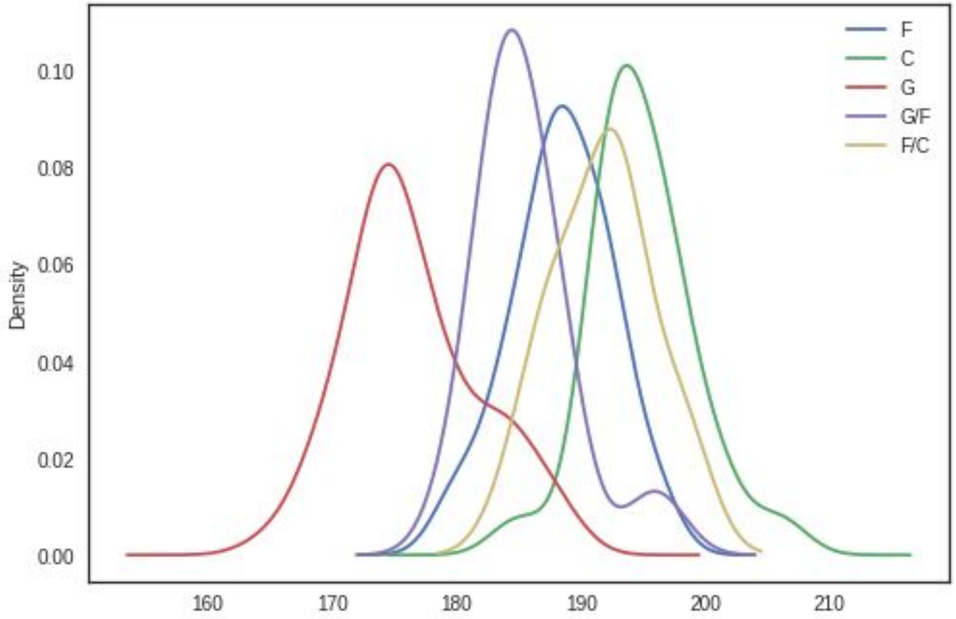


Kernel Density Estimate Plots

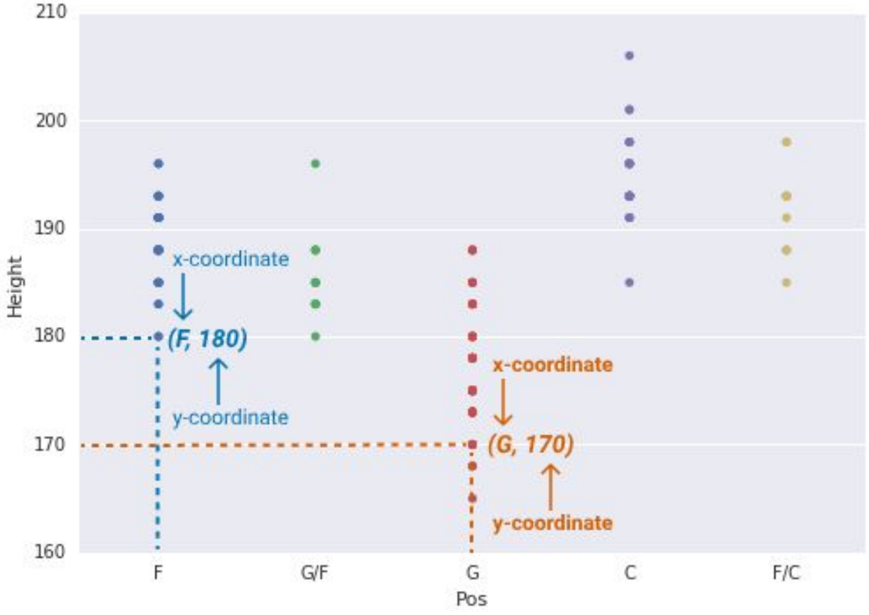


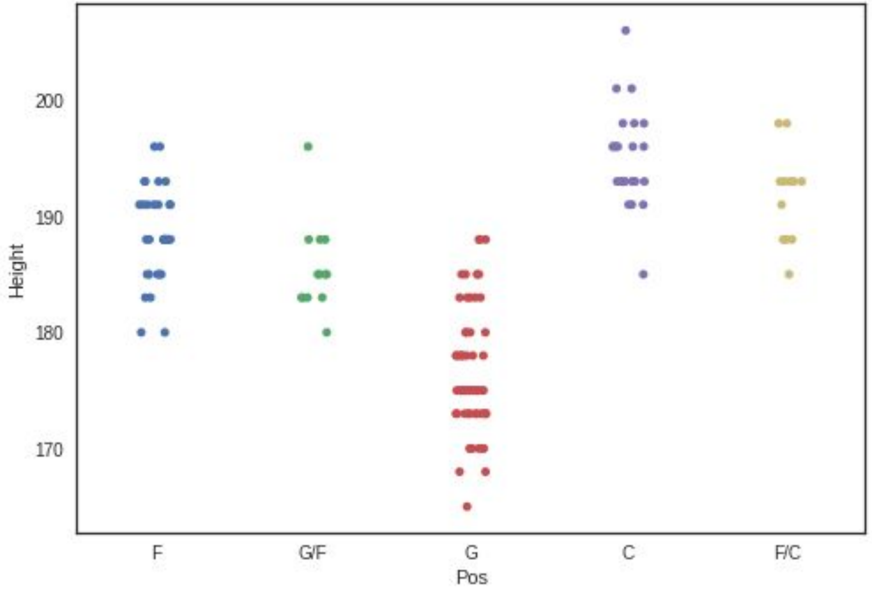
Drawbacks of Kernel Density Plots



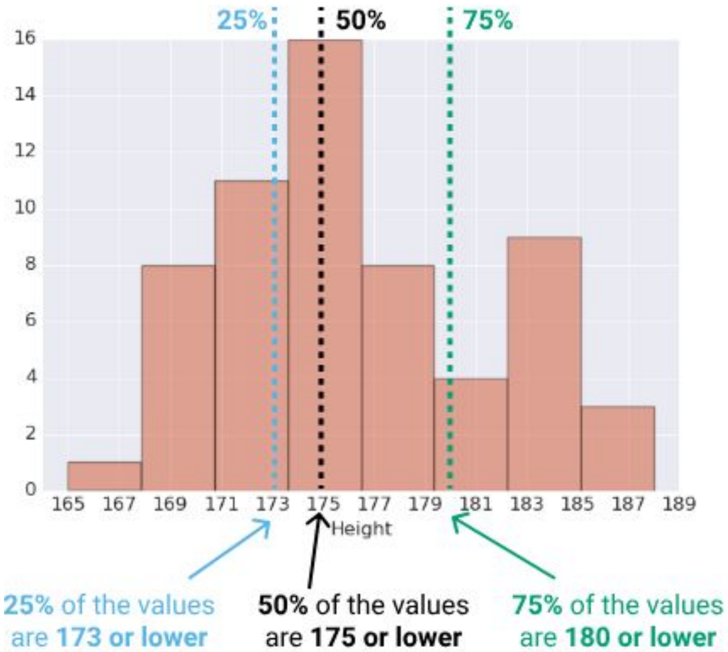


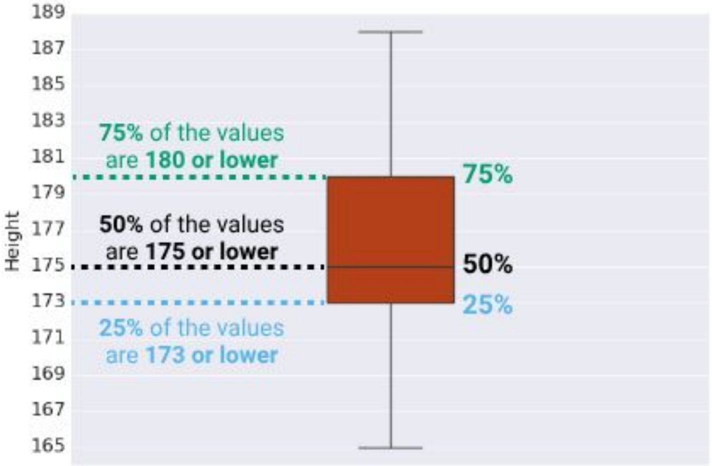
Strip Plots


210
200
190
180
170
160
Height
x-coordinate
(F, 180)
y-coordinate
x-coordinate
(G, 170)
y-coordinate
F
G/F
G
C
F/C
Pos

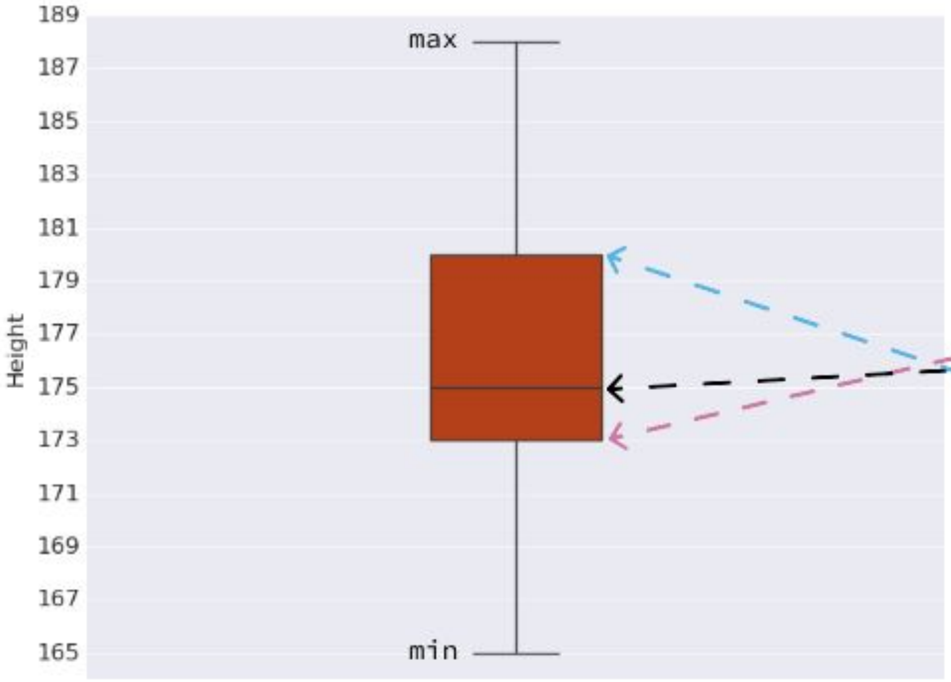

200
190
180
170
Height
F
G/F
G
C
F/C
Pos

Box Plots


25%
50%
75%
16
14
12
10
8
6
4
2
0
165 167 169 171 173 175 177 179 181 183 185 187 189
Height
25% of the values are 173 or lower
50% of the values are 175 or lower
75% of the values are 180 or lower

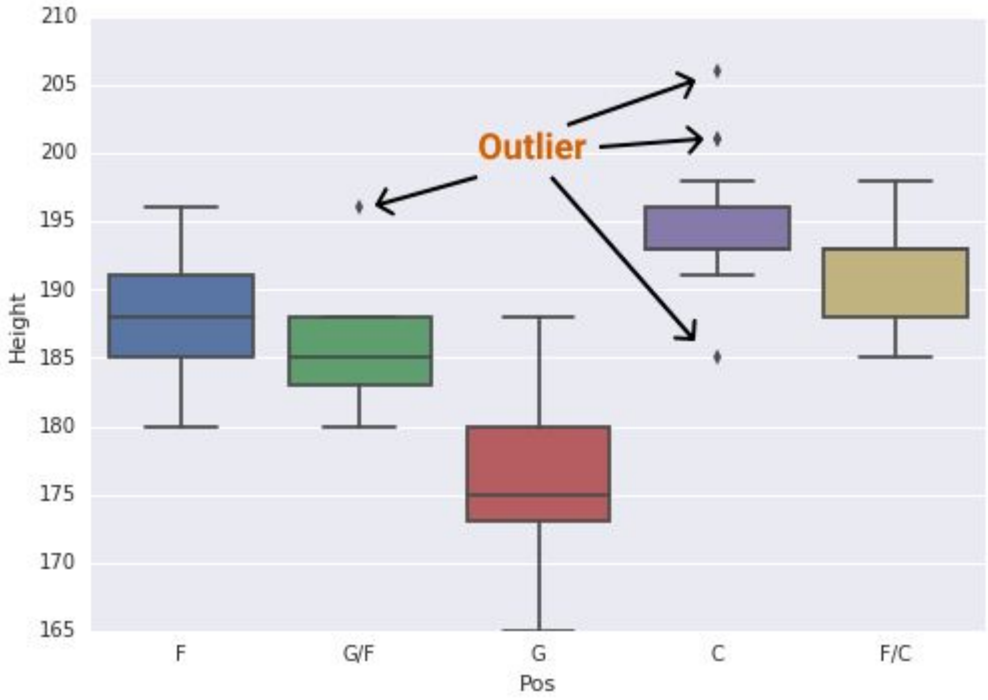

189
187
185
183
181
179
177
175
173
171
169
167
165
Height
75% of the values are 180 or lower
50% of the values are 175 or lower
25% of the values are 173 or lower
75%
50%
25%

Box Plots

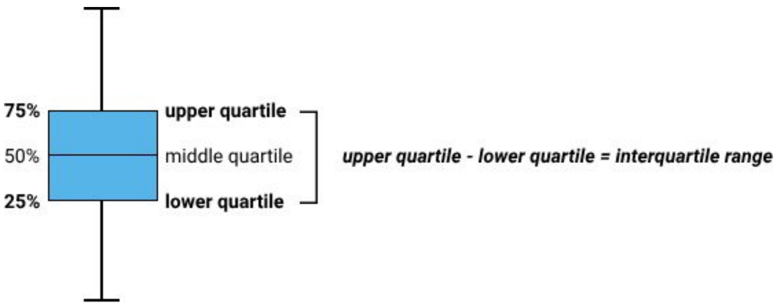



```
>> wnba[wnba['Pos'] == 'G']['Height'].describe()
count     60.000000
mean     176.450000
std        5.312043
min      165.000000
25%      173.000000
50%      175.000000
75%      180.000000
max      188.000000
Name: Height, dtype: float64
```

Outliers

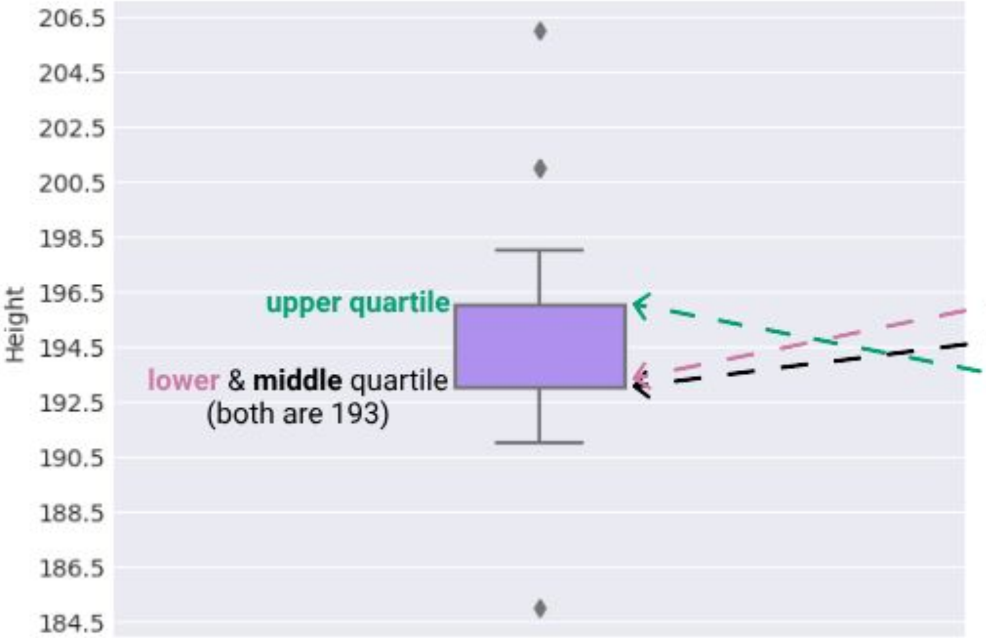


210
205
200
195
190
185
180
175
170
165
Height
Outlier
F
G/F
G
C
F/C
Pos



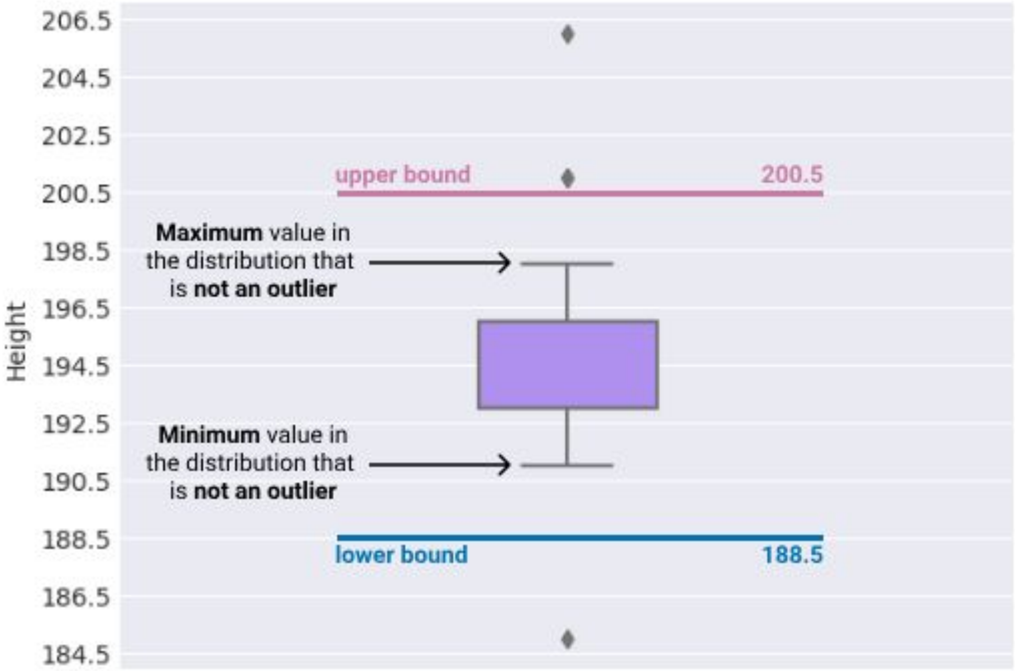
75%
50%
25%
upper quartile
middle quartile
lower quartile
upper quartile - lower quartile = interquartile range

Outliers




```
>> wnba[wnba['Pos'] == 'C']['Height'].describe()
count       25.000000
mean       194.920000
std          4.132392
min        185.000000
25%        193.000000
50%        193.000000
75%        196.000000
max        206.000000
Name: Height, dtype: float64
```

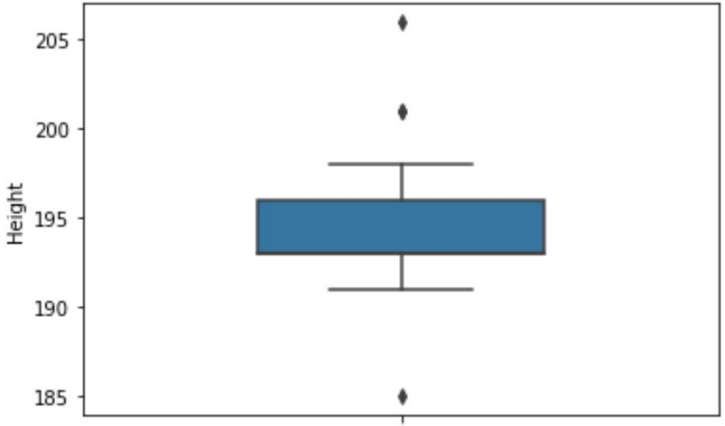
Outliers


Height
206.5
204.5
202.5
200.5
198.5
196.5
194.5
192.5
190.5
188.5
186.5
184.5
upper bound
200.5
Maximum value in the distribution that is **not an outlier**
Minimum value in the distribution that is **not an outlier**
lower bound
188.5

Outliers

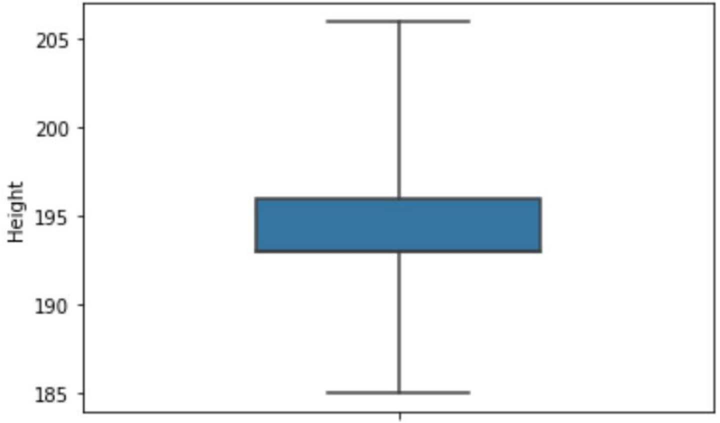
```
sns.boxplot(wnba[wnba['Pos'] == 'C']['Height'], whis = 1.5,
            orient = 'vertical', width = .45)
```

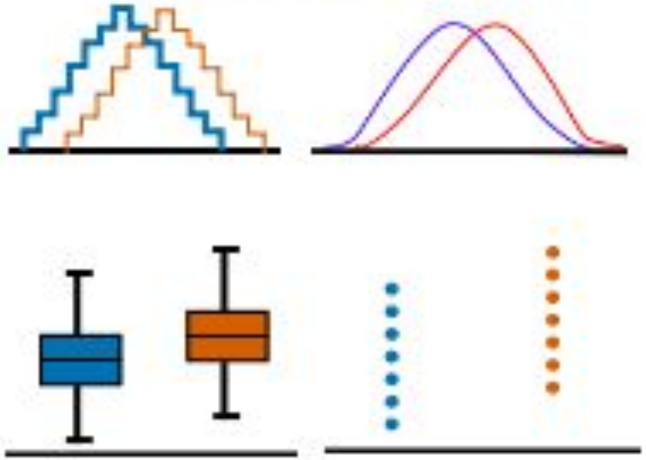
<matplotlib.axes._subplots.AxesSubplot at 0x1a180c4518>

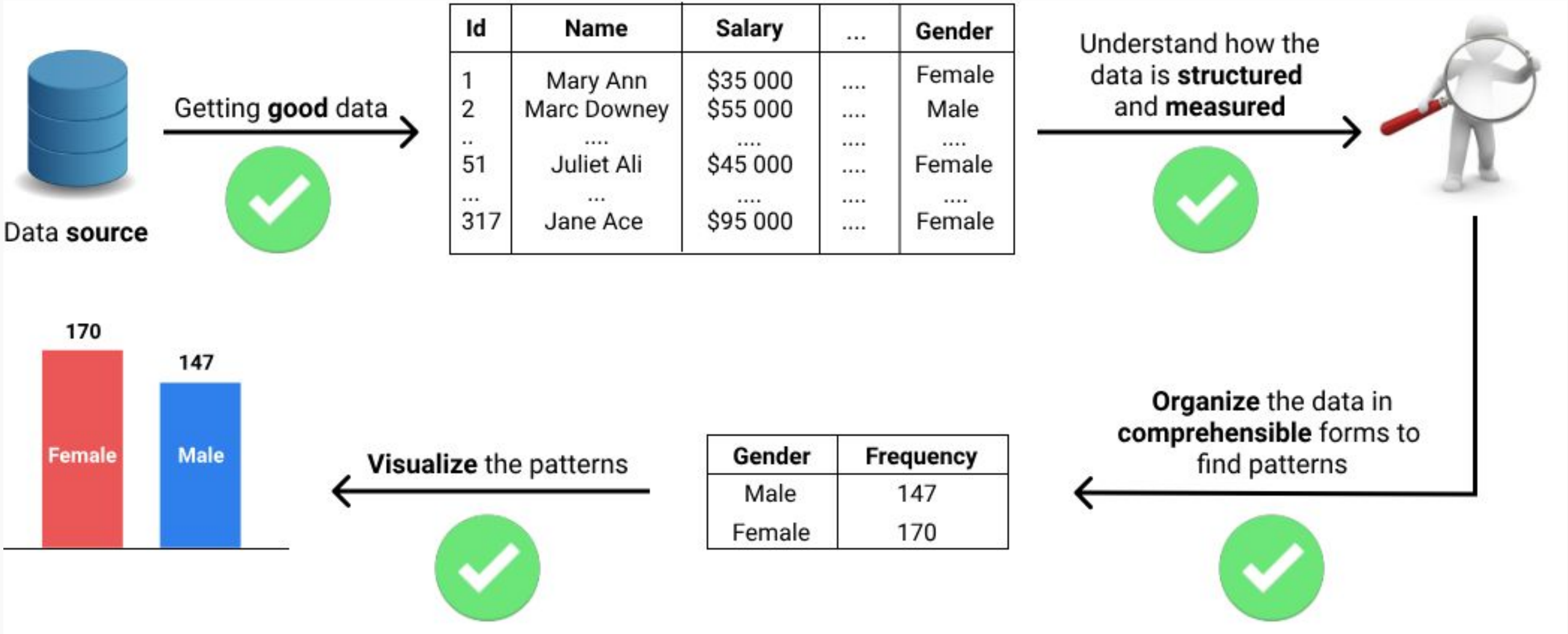



```
sns.boxplot(wnba[wnba['Pos'] == 'C']['Height'], whis = 4,
            orient = 'vertical', width = .45)
```

<matplotlib.axes._subplots.AxesSubplot at 0x1a18180208>



Scale of measurement	Graphs we can use to compare distributions
Nominal	
Ordinal	
Interval & Ratio	


Data source
Getting good data
Id
Name
Salary
...
Gender
1 Mary Ann $35 000 .... Female
2 Marc Downey $55 000 .... Male
51 Juliet Ali $45 000 .... Female
317 Jane Ace $95 000 .... Female
Understand how the data is structured and measured
Organize the data in comprehensible forms to find patterns
Gender Frequency
Male 147
Female 170
Visualize the patterns
170
147
Female
Male