IMD0033 - Probabilidade

Aula 17 - Média Ponderada e Mediana

Ivanovitch Silva

Maio, 2019

Agenda

- Média ponderada
- A mediana de distribuições abertas
- Cálculo da mediana
- A mediana como uma estatística de resistência
- A mediana para variáveis ordinais
- Sensitividade a mudanças

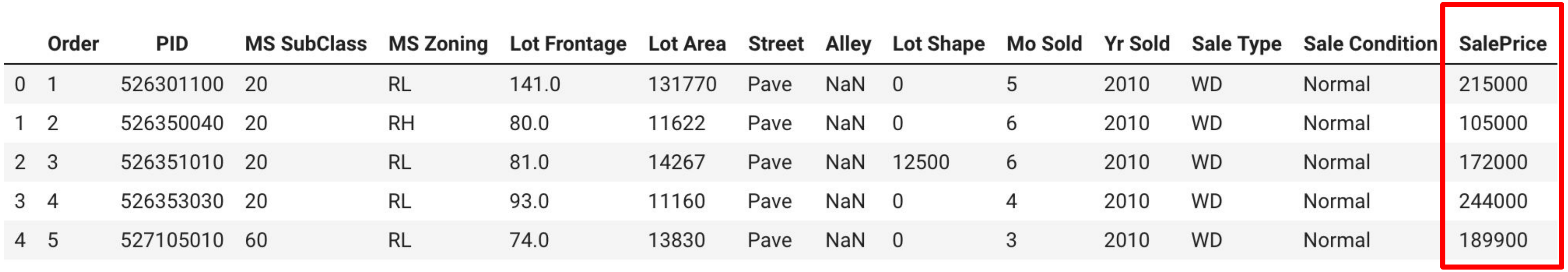
Atualizar o repositório

```
git clone https://github.com/ivanovitchm/imd0033_2019_1.git
```

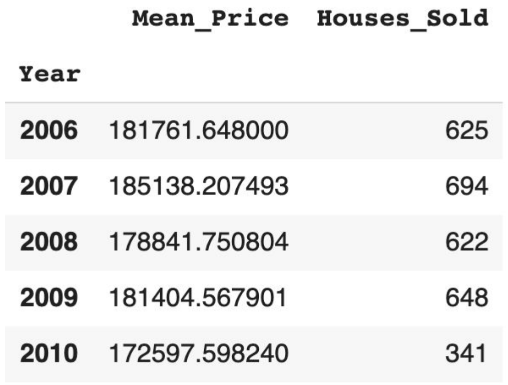
Ou ….

```
git pull
```

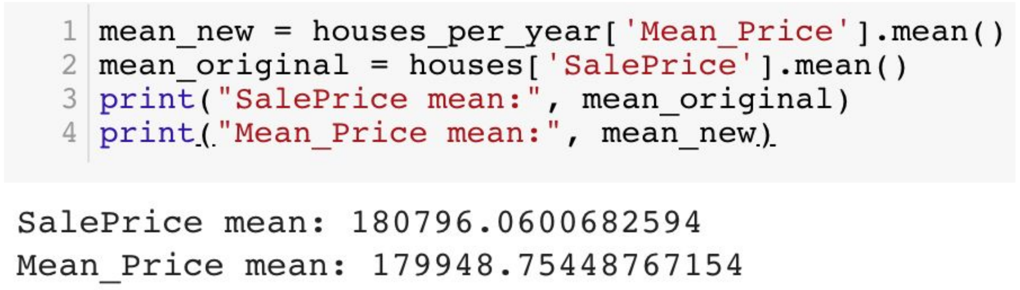
Introduction



	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	Mo Sold	Yr Sold	Sale Type	Sale Condition	SalePrice
0	1	526301100	20	RL	141.0	131770	Pave	NaN	0	5	2010	WD	Normal	215000
1	2	526350040	20	RH	80.0	11622	Pave	NaN	0	6	2010	WD	Normal	105000
2	3	526351010	20	RL	81.0	14267	Pave	NaN	12500	6	2010	WD	Normal	172000
3	4	526353030	20	RL	93.0	11160	Pave	NaN	0	4	2010	WD	Normal	244000
4	5	527105010	60	RL	74.0	13830	Pave	NaN	0	3	2010	WD	Normal	189900



	Mean_Price	Houses_Sold
Year		
2006	181761.648000	625
2007	185138.207493	694
2008	178841.750804	622
2009	181404.567901	648
2010	172597.598240	341



```
mean_new = houses_per_year['Mean_Price'].mean()
mean_original = houses['SalePrice'].mean()
print("SalePrice mean:", mean_original)
print("Mean_Price mean:", mean_new)
```

```
SalePrice mean: 180796.0600682594
Mean_Price mean: 179948.75448767154
```

Different Weights

$$2009 : [28\,700, 142\,500, 440\,000, 336\,860, 207\,500]$$

$$2010 : [135\,000, 139\,000]$$

$$\bar{x} = \frac{\overbrace{(28\,700 + 142\,500 + 440\,000 + 336\,860 + 207\,500)}^{2009} + \overbrace{(135\,000 + 139\,000)}^{2010}}{7}$$

$$\bar{x} = \frac{\overbrace{1\,413\,860}^{2009} + \overbrace{274\,000}^{2010}}{7} = 241122.86$$

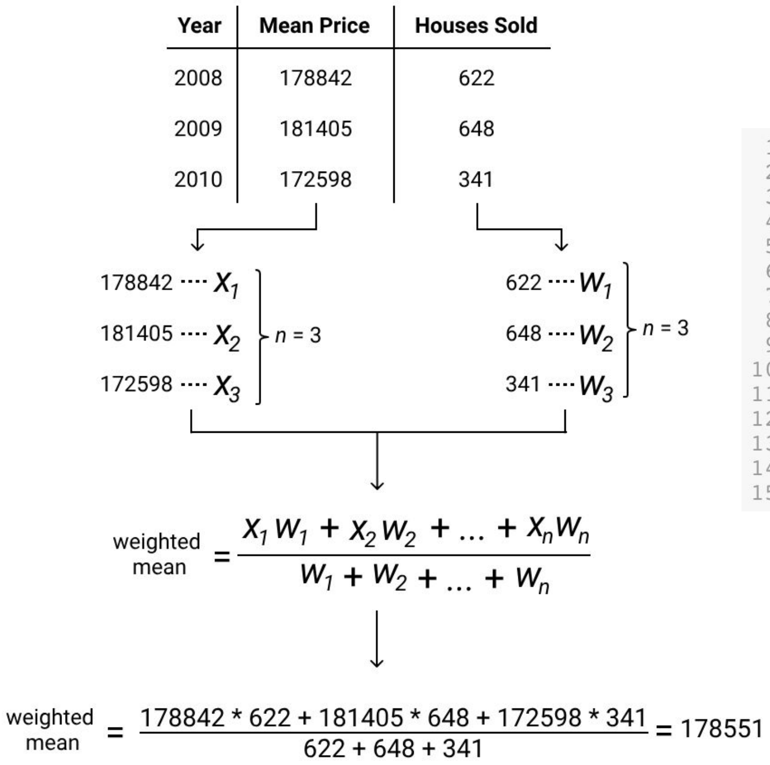
Different Weights

$$2009: \quad \bar{x} = \frac{28\,700 + 142\,500 + 440\,000 + 336\,860 + 207\,500}{5} = 282\,772$$

$$2010: \quad \bar{x} = \frac{135\,000 + 139\,000}{2} = 137\,000$$

$$overall\ mean\ : \quad \bar{x} = \frac{282\,772 + 137\,000}{2} = \boxed{209\,886}$$

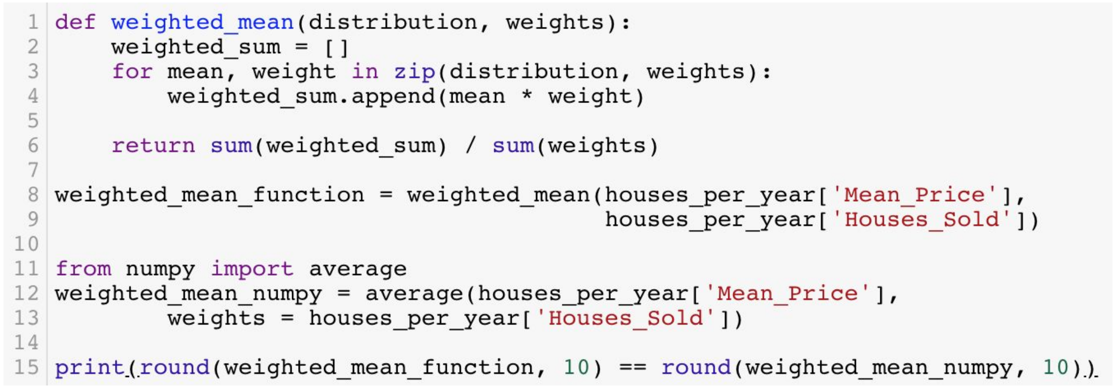
The Weighted Mean



Year	Mean Price	Houses Sold
2008	178842	622
2009	181405	648
2010	172598	341

$$\text{weighted mean} = \frac{X_1 W_1 + X_2 W_2 + \ldots + X_n W_n}{W_1 + W_2 + \ldots + W_n}$$

$$\text{weighted mean} = \frac{178842 * 622 + 181405 * 648 + 172598 * 341}{622 + 648 + 341} = 178551$$



```python
def weighted_mean(distribution, weights):
    weighted_sum = []
    for mean, weight in zip(distribution, weights):
        weighted_sum.append(mean * weight)

    return sum(weighted_sum) / sum(weights)

weighted_mean_function = weighted_mean(houses_per_year['Mean_Price'],
                                       houses_per_year['Houses_Sold'])

from numpy import average
weighted_mean_numpy = average(houses_per_year['Mean_Price'],
        weights = houses_per_year['Houses_Sold'])

print(round(weighted_mean_function, 10) == round(weighted_mean_numpy, 10))
```

When is not possible to compute the mean

```
houses['TotRms AbvGrd'].value_counts()
```

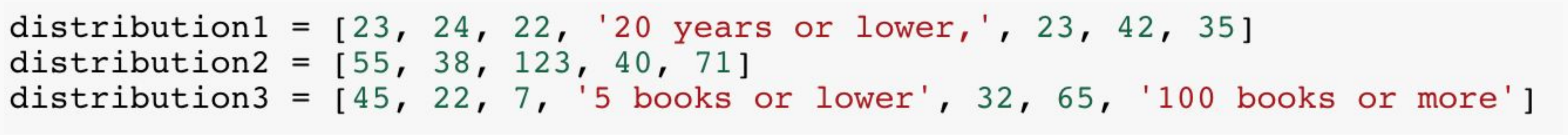
```
6              844
7              649
5              586
8              347
4              203
9              143
10 or more     131
3               26
2                1
```

Median = the middle value

$$[\underbrace{5, 6}_{\text{length = 2 values}}, 7, \overbrace{7, 10 \text{ or more}}^{\text{length = 2 values}}]$$

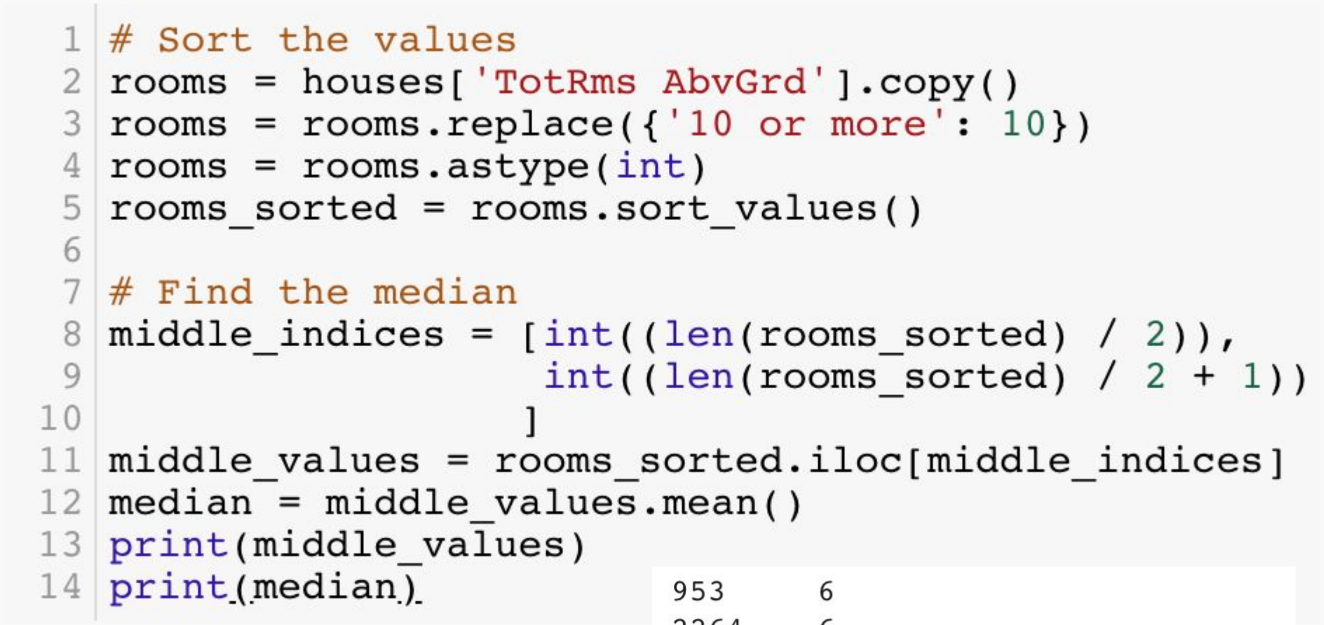
$$[\underbrace{5, 6}_{\text{length = 2 values}}, 7, 7, \overbrace{8, 10 \text{ or more}}^{\text{length = 2 values}}]$$

Computer the median



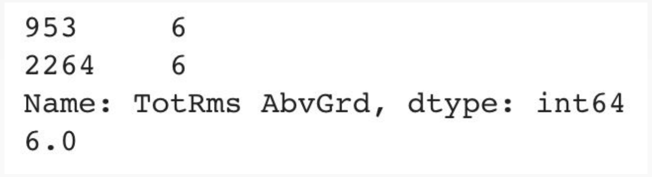
```
distribution1 = [23, 24, 22, '20 years or lower,', 23, 42, 35]
distribution2 = [55, 38, 123, 40, 71]
distribution3 = [45, 22, 7, '5 books or lower', 32, 65, '100 books or more']
```

Computer the median



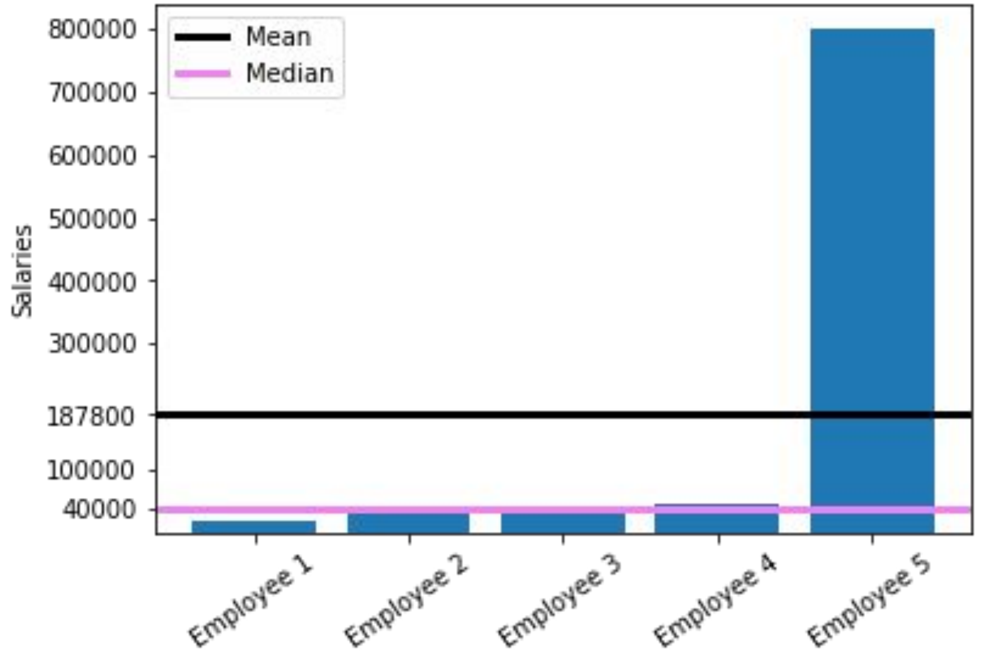
```python
# Sort the values
rooms = houses['TotRms AbvGrd'].copy()
rooms = rooms.replace({'10 or more': 10})
rooms = rooms.astype(int)
rooms_sorted = rooms.sort_values()

# Find the median
middle_indices = [int((len(rooms_sorted) / 2)),
                  int((len(rooms_sorted) / 2 + 1))
                 ]
middle_values = rooms_sorted.iloc[middle_indices]
median = middle_values.mean()
print(middle_values)
print(median)
```



```
953     6
2264    6
Name: TotRms AbvGrd, dtype: int64
6.0
```

The median as a resistant statistic



$[20000, 34000, 40000, 45000, 800000]$

The median ideal for finding reasonable averages for distributions containing outliers.

The Median for Ordinal Scales

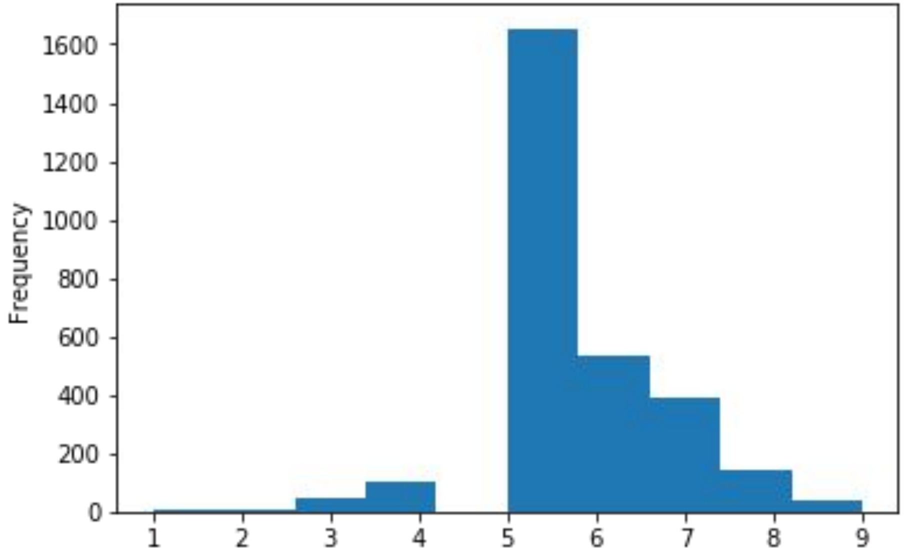
```
houses['Overall Cond'].value_counts().sort_index()
```

```
1        7
2       10
3       50
4      101
5     1654
6      533
7      390
8      144
9       41
```

If the overall condition of a house is rated with an 8 (Very good), and another house gets a 4 (Below average), we can't say that the conditions of the former are twice as better than the latter.

Code	Quality
1	Very poor
2	Poor
3	Fair
4	Below average
5	Average
6	Above average
7	Good
8	Very good
9	Excellent
10	Very excellent

The Median for Ordinal Scales



Mean: 5.56
Median: 5

The mean seems more representative and more informative because it captures the fact that there are more houses rated above 5 than rated under 5. Because of this,
the mean is slightly shifted above 5.

Sensitivity to Changes

Code	Answer
1	Strongly disagree
2	Disagree
3	Neither agree nor disagree
4	Agree
5	Strongly agree

mean, median = 2

$$[1, 1, 1, 2, 2, 2, 2, 3, 3, 3]$$

$$\downarrow$$

$$[1, 2, 2, 2, 2, 2, 4, 5, 5, 5]$$

mean = 3
median = 2

Next Steps

- We continue the discussion about finding averages for ordinal data and also learn new things, like finding the average value for nominal variables.

MacBook Pro