Hubert10 😎

Regresja liniowa

Prognozowanie wartości danej zmiennej na podstawie innej.

Będę prognozował wartość 'Mieszkań oddanych do użytkowania' na podstawie 'roku'.

Spis streści

- Pobranie danych
- Wczytanie danych
- Wstępna obróbka danych
- Zmiana typu object na float
- Współczynnik korelacji Pearsona
- Przygotowanie danych do modelu
- Model
- Korzystanie z modelu
- Wykres - regresja liniowa
- Znaczenie graficznej reprezentacji danych

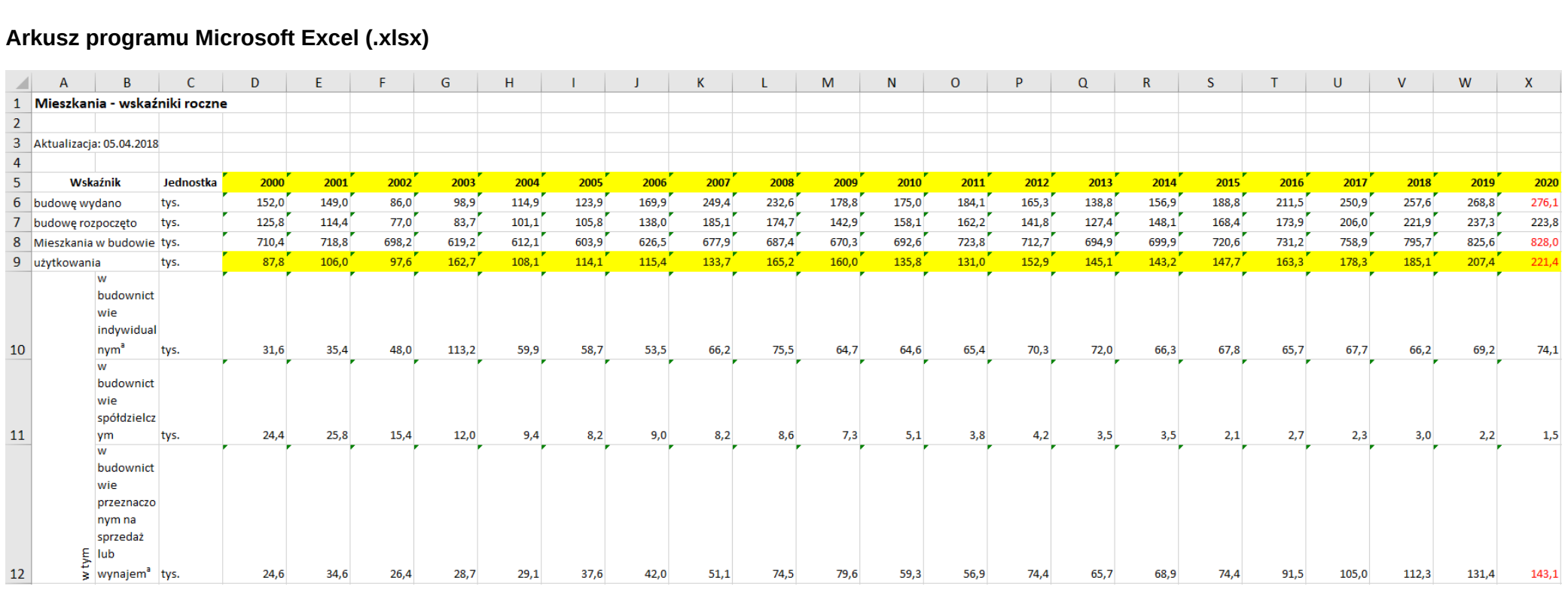
Pobranie danych

https://bdm.stat.gov.pl/ -> MIESZKANIA

Przejdź do ustawień, aby wyświetlić pełny zakres dostępnych danych.

Wskaźnik	Jednostka		2014	2015	2016	2017	2018	2019	2020
Mieszkania, na których budowę wydano pozwolenia lub dokonano zgłoszenia z projektem budowlanym	tys.	3,8	156,9	188,8	211,5	250,9	257,6	268,8	276,1
Mieszkania, których budowę rozpoczęto	tys.	7,4	148,1	168,4	173,9	206,0	221,9	237,3	223,8
Mieszkania w budowie	tys.	4,9	699,9	720,6	731,2	758,9	795,7	825,6	828,0
Mieszkania oddane do użytkowania	**tys.**	**5,1**	**143,2**	**147,7**	**163,3**	**178,3**	**185,1**	**207,4**	**221,4**

Arkusz programu Microsoft Excel (.xlsx)



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	Mieszkania - wskaźniki roczne																							
2																								
3	Aktualizacja: 05.04.2018																							
4																								
5	Wskaźnik		Jednostka	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
6	budowę wydano		tys.	152,0	149,0	86,0	98,9	114,9	123,9	169,9	249,4	232,6	178,8	175,0	184,1	165,3	138,8	156,9	188,8	211,5	250,9	257,6	268,8	276,1
7	budowę rozpoczęto		tys.	125,8	114,4	77,0	83,7	101,1	105,8	138,0	185,1	174,7	142,9	158,1	162,2	141,8	127,4	148,1	168,4	173,9	206,0	221,9	237,3	223,8
8	Mieszkania w budowie		tys.	710,4	718,8	698,2	619,2	612,1	603,9	626,5	677,9	687,4	670,3	692,6	723,8	712,7	694,9	699,9	720,6	731,2	758,9	795,7	825,6	828,0
9	użytkowania		tys.	87,8	106,0	97,6	162,7	108,1	114,1	115,4	133,7	165,2	160,0	135,8	131,0	152,9	145,1	143,2	147,7	163,3	178,3	185,1	207,4	221,4
10		w budownictwie indywidualnym[8]	tys.	31,6	35,4	48,0	113,2	59,9	58,7	53,5	66,2	75,5	64,7	64,6	65,4	70,3	72,0	66,3	67,8	65,7	67,7	66,2	69,2	74,1
11		w budownictwie spółdzielczym	tys.	24,4	25,8	15,4	12,0	9,4	8,2	9,0	8,2	8,6	7,3	5,1	3,8	4,2	3,5	3,5	2,1	2,7	2,3	3,0	2,2	1,5
12	w tym	w budownictwie przeznaczonym na sprzedaż lub wynajem[8]	tys.	24,6	34,6	26,4	28,7	29,1	37,6	42,0	51,1	74,5	79,6	59,3	56,9	74,4	65,7	68,9	74,4	91,5	105,0	112,3	131,4	143,1

Wczytanie danych

In [1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn import linear_model
```

In [2]:
```python
dane = pd.read_excel('C:\\Users\\V6\Desktop\\Mieszkania.xlsx', header=3)
dane
```

Out[2]:

	Unnamed: 0	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	...	Unnamed: 14	Unnamed: 15	Unnamed
0	Wskaźnik	NaN	Jednostka	2000	2001	2002	2003	2004	2005	2006	...	2011	2012	201
1	Mieszkania, na których budowę wydano pozwoleni...	NaN	tys.	152,0	149,0	86,0	98,9	114,9	123,9	169,9	...	184,1	165,3	138,
2	Mieszkania, których budowę rozpoczęto	NaN	tys.	125,8	114,4	77,0	83,7	101,1	105,8	138,0	...	162,2	141,8	127,
3	Mieszkania w budowie	NaN	tys.	710,4	718,8	698,2	619,2	612,1	603,9	626,5	...	723,8	712,7	694,
4	Mieszkania oddane do użytkowania	NaN	tys.	87,8	106,0	97,6	162,7	108,1	114,1	115,4	...	131,0	152,9	145,
5	w tym	w budownictwie indywidualnyma	tys.	31,6	35,4	48,0	113,2	59,9	58,7	53,5	...	65,4	70,3	72,
6	NaN	w budownictwie spółdzielczym	tys.	24,4	25,8	15,4	12,0	9,4	8,2	9,0	...	3,8	4,2	3,
7	NaN	w budownictwie przeznaczonym na sprzedaż lub w...	tys.	24,6	34,6	26,4	28,7	29,1	37,6	42,0	...	56,9	74,4	65,

8 rows × 24 columns

Wstępna obróbka danych

In [3]:
```python
# usuniecie kolumn
dane = dane.drop('Unnamed: 0', axis=1)
dane = dane.drop('Unnamed: 1', axis=1)
dane = dane.drop('Unnamed: 2', axis=1)
```

In [4]:
```python
# usuniecie wierszy
dane = dane.drop([1,2,3,5,6,7])
```

In [5]:
```python
# zmiana nazw indeksów
dane = dane.rename(index={0:"rok", 4:"mieszkania"})
dane.head(3)
```

Out[5]:

	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	Unnamed: 10	Unnamed: 11	Unnamed: 12	...	Unnamed: 14	Unnamed: 15	Unna
rok	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	...	2011	2012	
mieszkania	87,8	106,0	97,6	162,7	108,1	114,1	115,4	133,7	165,2	160,0	...	131,0	152,9	

2 rows × 21 columns

In [6]:
```python
# transponowanie ramki danych
dane = dane.transpose()
dane
```

Out[6]:

	rok	mieszkania
Unnamed: 3	2000	87,8
Unnamed: 4	2001	106,0
Unnamed: 5	2002	97,6
Unnamed: 6	2003	162,7
Unnamed: 7	2004	108,1
Unnamed: 8	2005	114,1
Unnamed: 9	2006	115,4
Unnamed: 10	2007	133,7
Unnamed: 11	2008	165,2
Unnamed: 12	2009	160,0
Unnamed: 13	2010	135,8
Unnamed: 14	2011	131,0
Unnamed: 15	2012	152,9
Unnamed: 16	2013	145,1
Unnamed: 17	2014	143,2
Unnamed: 18	2015	147,7
Unnamed: 19	2016	163,3
Unnamed: 20	2017	178,3
Unnamed: 21	2018	185,1
Unnamed: 22	2019	207,4
Unnamed: 23	2020	221,4

Zmiana typu object na float

In [7]:
```python
dane.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 21 entries, Unnamed: 3 to Unnamed: 23
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   rok         21 non-null     object
 1   mieszkania  21 non-null     object
dtypes: object(2)
memory usage: 1.1+ KB
```

Zmiana przecinka na kropke

In [8]:
```python
# zmienia 'komórki' w których występowałby sam przecinek
# dane = dane.replace(',', '.')
```

In [9]:
```python
# zmienia KAŻDY przecinek w kolumnie 'mieszkania' na kropkę
dane["mieszkania"]=[x.replace(",",".")for x in dane["mieszkania"] ]
dane.head(3)
```

Out[9]:

	rok	mieszkania
Unnamed: 3	2000	87.8
Unnamed: 4	2001	106.0
Unnamed: 5	2002	97.6

po zmianie przecinka na kropke dopiero mogę zmienić typ kolumn z object na int/float

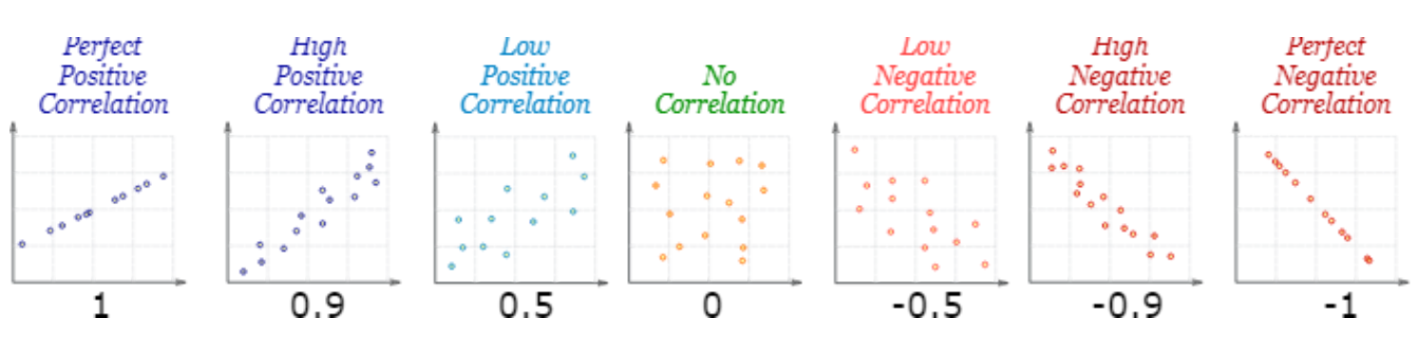
In [10]:
```python
#zmiana object(string) na int

dane["rok"] = pd.to_numeric(dane["rok"])
#dane["rok"] = pd.to_numeric(dane["rok"], downcast='float') - gdybym chciał zamienić na float
dane["mieszkania"] = pd.to_numeric(dane["mieszkania"])
```

In [11]:
```python
dane.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 21 entries, Unnamed: 3 to Unnamed: 23
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   rok         21 non-null     int64  
 1   mieszkania  21 non-null     float64
dtypes: float64(1), int64(1)
memory usage: 1.1+ KB
```

Współczynnik korelacji Pearsona



In [12]:
```python
# przykład
from scipy import stats
a = np.array([3,3,6])
b = np.array([2,3,4])
stats.pearsonr(a, b)
```

Out[12]:
```
(0.8660254037844388, 0.333333333333333315)
```

In [13]:
```python
from scipy import stats
a = dane['rok'].values
b = dane['mieszkania'].values
stats.pearsonr(a, b)

# Odp. r = 0,84
```

Out[13]:
```
(0.8413042761590577, 1.7691785643153179e-06)
```

In [14]:
```python
c = stats.pearsonr(a, b)
print("r czyli współczynnik korelacji Pearsona wynosi: ",round(c[0], 2))
```

```
r czyli współczynnik korelacji Pearsona wynosi:  0.84
```

Przygotowanie danych do modelu

In [15]:
```python
x = dane['rok'].values
y = dane['mieszkania'].values
```

In [16]:
```python
x
```

Out[16]:
```
array([2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010,
       2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020],
      dtype=int64)
```

In [17]:
```python
y
```

Out[17]:
```
array([ 87.8, 106. ,  97.6, 162.7, 108.1, 114.1, 115.4, 133.7, 165.2,
       160. , 135.8, 131. , 152.9, 145.1, 143.2, 147.7, 163.3, 178.3,
       185.1, 207.4, 221.4])
```

In [18]:
```python
x = x.reshape((21, 1))
#y = y.reshape((21, 1))
```

In [19]:
```python
x
```

Out[19]:
```
array([[2000],
       [2001],
       [2002],
       [2003],
       [2004],
       [2005],
       [2006],
       [2007],
       [2008],
       [2009],
       [2010],
       [2011],
       [2012],
       [2013],
       [2014],
       [2015],
       [2016],
       [2017],
       [2018],
       [2019],
       [2020]], dtype=int64)
```

Model

In [20]:
```python
# stworzenie modelu
model = linear_model.LinearRegression()
```

In [21]:
```python
# dopasowanie modelu do danych
model.fit(x, y)
```

Out[21]:
```
LinearRegression()
```

Korzystanie z modelu

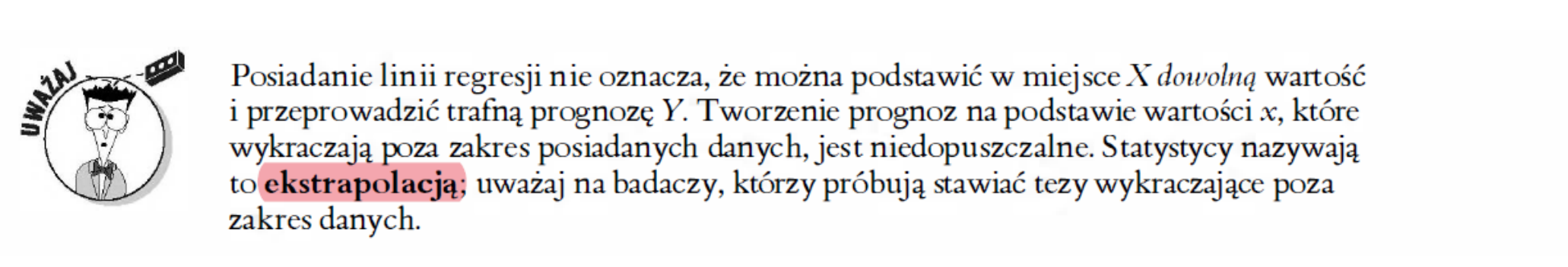
Ile zostanie oddanych mieszkań do użytkowania w 1930 roku?

In [22]:
```python
model.predict([[1930]])
```

Out[22]:
```
array([-233.2961039])
```

Odp. W 1930 roku zostało oddanych -233 tysięcy mieszkań do użytkowania. 😮

WAŻNE



Posiadanie linii regresji nie oznacza, że można podstawić w miejsce *X* *dowolną* wartość i przeprowadzić trafną prognozę *Y*. Tworzenie prognoz na podstawie wartości *x*, które wykraczają poza zakres posiadanych danych, jest niedopuszczalne. Statystycy nazywają to **ekstrapolacją**; uważaj na badaczy, którzy próbują stawiać tezy wykraczające poza zakres danych.

Dlatego w miejsce X będę wstawiał tylko 2000-2020

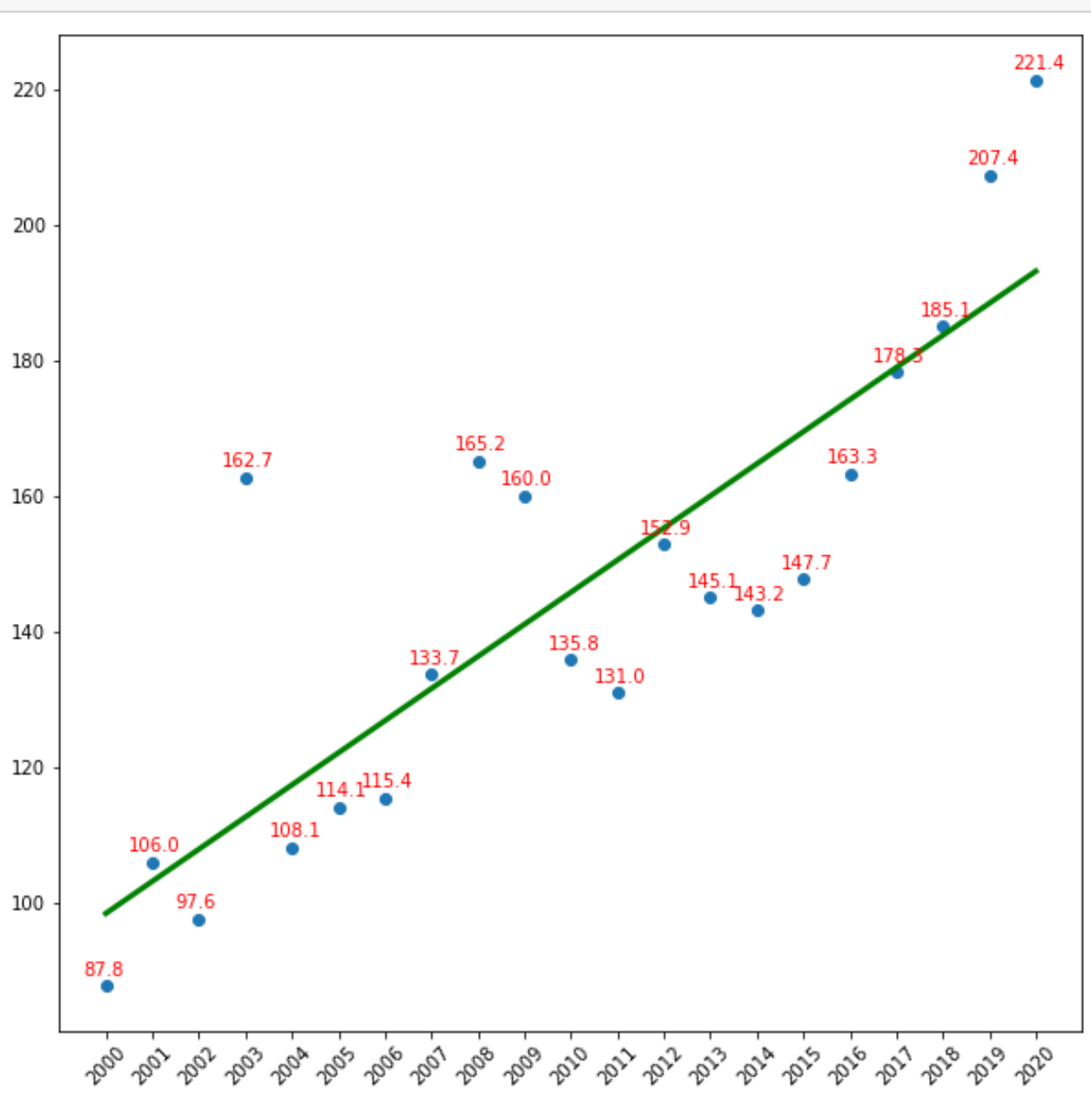
Wykres - regresja liniowa

In [23]:
```python
YPred=model.predict(x)

plt.figure(figsize=(10,10))
plt.scatter(x,y)
plt.plot(x,YPred,color='green',lw=3)

plt.xticks(range(2000,2021), rotation = 45)

for i, txt in enumerate(y):
    plt.annotate(txt, (x[i]-0.5, y[i]+1.5), color='red')
```



In [24]:
```python
import sklearn.metrics as sm

print("średni błąd kwadratowy =", round(sm.mean_absolute_error(y,YPred),2))
print("średni kwadrat błędu =", round(sm.mean_squared_error(y,YPred),2))
print("Medianowy błąd absolutny =", round(sm.median_absolute_error(y,YPred),2))
print("R2 błąd =", round(sm.r2_score(y,YPred),2))
```

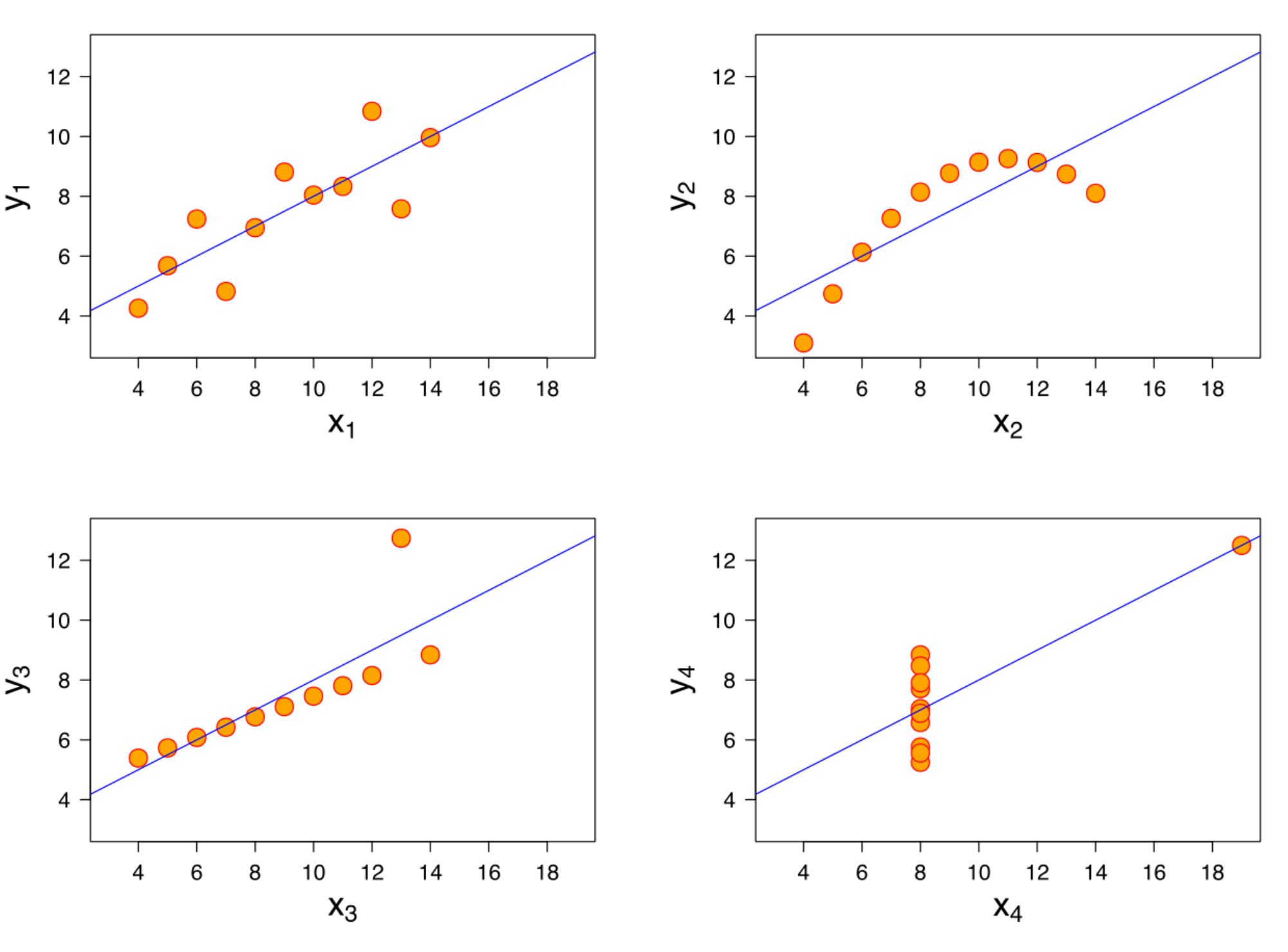
```
średni błąd kwadratowy = 14.42
średni kwadrat błędu = 339.92
Medianowy błąd absolutny = 10.93
R2 błąd = 0.71
```

Znaczenie graficznej reprezentacji danych

https://pl.wikipedia.org/wiki/Kwartet_Anscombe%E2%80%99a

Kwartet Anscombe'a

- zestaw czterech zestawów danych o identycznych cechach statystycznych, takich jak średnia arytmetyczna, wariancja, współczynnik korelacji czy **równanie regresji liniowej**. Układ tych danych został stworzony w 1973 roku aby ukazać znaczenie graficznej reprezentacji danych przy okazji ich analizy statystycznej.



In [ ]: