Task2

October 13, 2018

```
In [99]: import numpy as np
         import pandas as pd
```

Считаем данные из файла "Cauchy.csv"

```
In [100]: cauchy = pd.read_csv('Cauchy.csv', header=None, names=["data"])
```

Посмотрим на первые несколько из них

```
In [101]: cauchy.head(10)
```

```
Out[101]:      data
          0 -194.37
          1 -192.37
          2 -195.53
          3 -194.83
          4 -195.28
          5 -196.28
          6 -161.94
          7 -196.10
          8 -198.46
          9 -194.87
```

Выведем также общую статистику по данным

```
In [102]: cauchy.describe()
```

```
Out[102]:               data
          count  1000.000000
          mean   -194.223070
          std      18.549751
          min    -325.010000
          25%    -196.082500
          50%    -195.020000
          75%    -194.047500
          max     294.530000
```

Посчитаем оценку максимального правдоподобия. Для этого напишем функцию логарифма плотности распределения Коши, которая также будем на вход принимать параметр x_0

```
In [103]: def ln_p(x, x0):
              return np.log(1 / (np.pi * (1 + (x - x0) ** 2)))
```

Далее для удобства создадим вектор из наших измерений

```
In [104]: sample = np.array(cauchy.values)
```

Оценим параметр сдвига методом максимального правдоподобия, то есть найдем: $\hat{x_0}(X1,\ldots,X_n) = \underset{x_0}{argmax}\ f_{x_0}(X_1,\ldots,X_n) = \underset{x_0}{argmax}\ L_{x_0}(X_1,\ldots,X_n) = \underset{x_0}{argmax}\ \sum_{i=0}^{n} \ln p_{x_0}(X_i)$. Второе равенство следует из монотонности логарифма.

Известно, что параметр сдвига принадлежит интервалу [-1000, 1000]. Возьмем шаг равный 0.01 и для каждого x_0 будем считать логарифм функции правдоподобия и находить среди них максимум. Для начала возьмем половину выборки (первые 500 наблюдений)

```
In [105]: thetas = [x * 0.01 for x in range(-100000, 100001)]
          max_estimator = thetas[0]
          max_likelihood_function = np.sum(ln_p(sample[:500], thetas[0]))
          for theta in thetas[1:]:
              current_likelihood_function = np.sum(ln_p(sample[:500], theta))
              if current_likelihood_function > max_likelihood_function:
                  max_likelihood_function = current_likelihood_function
                  max_estimator = theta
          print("Оценка максимального правдоподобия -", max_estimator)
```

```
Оценка максимального правдоподобия - -195.11
```

Теперь сделаем то же самое на всей выборке:

```
In [106]: thetas = [x * 0.01 for x in range(-100000, 100001)]
          max_estimator = thetas[0]
          max_likelihood_function = np.sum(p(sample, thetas[0]))
          for theta in thetas[1:]:
              current_likelihood_function = np.sum(p(sample, theta))
              if current_likelihood_function > max_likelihood_function:
                  max_likelihood_function = current_likelihood_function
                  max_estimator = theta
          print("Оценка максимального правдоподобия -", max_estimator)
```

```
Оценка максимального правдоподобия - -195.09
```

Вывод: Как видно из результатов, оценка максимального правдоподобия параметра сдвига почти не отличается для половины и всей выборки. Кроме этого можно увидеть, что оценка почти равна среднему значению по выборке, что и не удивительно, так как наше распределение симметрично относительно параметра x_0. Таким образом мы смогли оценить параметр сдвига в распределении Коши, даже при том, что у этого распределения отсутсвует математическое ожидание и дисперсия, что показывает, что данный метод является довольно универсальным.