Prediction of money given by an Insurance Company by number of claims

By Valter Núñez - A01206138

Abstract - The present document shows the usage of linear regression done by hand and done with the framework Sci-kit in order to predict the total claim payments for a Swedish Insurance company.

I - INTRODUCTION

Insurance companies have become one of the most useful and profitable in the last few years. Covering from car accidents, to laptops, to whole apartment complexes, to health related issues, insurance has been part of our lives for a very long time.

In order to obtain one of the benefits that the policy dictates, you need to do a claim. After that, the Insurance Company will look at your case, at the terms of the insurance policy you signed, and decide how to proceed. There is a trend on how much money they give you back depending on the claims done.

II - DATASET

For the present document, the dataset used was obtained from the Department of Mathematics and Statistics of the Faculty of Science from the Masaryk University in the Czech Republic[1]. The data was collected by the Swedish Committee on Analysis of Risk Premium in Motor Insurance.

The dataset is divided into two columns. The first one is the number of claims. The second one is the total payment for all claims in thousands of Swedish Kronor, which is the Swedish currency.

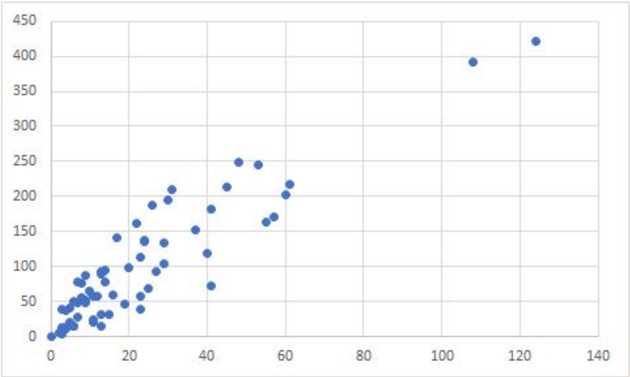

Figure 1 - Scatter plot of the dataset

[1] http://www.math.muni.cz

III - APPROACH

The present project looks for a better prediction of the amount, in thousands, of Kronor, given a number of claims using linear regression. For that reason, there will be a comparison between two ways of calculating said number: doing the algorithm by hand and using a framework. Since the dataset has only 2 columns, and both of them are relevant, there has been no pre-processing of the data, other than having it in a CSV file.

The first approach is done with Python without any external libraries. Because of that, this approach is the most difficult one. The programming of the linear regression algorithm is based on the algebraic formula for the linear regression:

$$y = b0 + b1 * x$$

where

$$b1 = \frac{\sum_{i=1}^{n}((xi - mean(x)) * (yi - mean(y)))}{\sum_{i=1}^{n} (xi - mean(x))^{2}}$$

and

$$b0 = mean(y) - b1 * mean(x)$$

This leads to the calculation of several other important numbers. The first one is the *mean*, that gets calculated with the following formula:

$$mean(x) = \frac{\sum_{i=1} xi}{count(x)}$$

The value of *b1* is calculated in a hard way. For that reason we need to split the calculations to make it easier. The *variance* is the bottom part of the formula of *b1.*

$$variance = \sum_{i=1}^{n} (xi - mean(x))^{2}$$

The *covariance* is the remaining part of the formula, and it is calculated as follows:

$$covariance = \sum_{i=1}^{n} ((xi - mean(x)) * (yi - mean(y)))$$

With the *covariance* and the *variance* already calculated, the first number for the linear regression formula is now possible. We can now calculate b1 like this:

$$b1 = \frac{covariance(x,y)}{variance(x)}$$

This makes the calculation of *b0*, and therefore of *y* direct and fairly easy.

The second approach for the linear regression algorithm is the easiest. Using the Sci-kit framework, you only need to use the linear model to calculate the result. The approach is simple and fast, especially compared to the first one.

On both approaches, the dataset was divided into training and testing sets. The training set is 70% of the original dataset, while the testing set is 30%.

IV - RESULTS

For the present project, we needed to know which of the approaches was the best one for the given dataset. Therefore, we used the RMSE, or root-mean-square error, to decide.

On the case of the first approach (the one done by hand), we got the following results:

Predicted	**Actual**
184.6001553	248.1
102.226753	39.6
49.50777548	48.8
33.03309502	6.6
46.21283939	50.9
46.21283939	14.8
56.09764766	48.7
121.9963695	103.9
39.6229672	11.8
39.6229672	38.1
26.44322283	0
39.6229672	12.6
79.16220031	59.6
224.1393884	202.4
62.68751985	21.3
69.27739203	93
69.27739203	31.9
72.57232813	95.5

RMSE - 31.005

On the other hand, the linear regression done with the framework showed the following results:

Predicted	**Actual**
64.17447	93
102.1312	137.9
84.87813	46.2
50.37202	87.4
98.68057	56.9
29.66836	39.9
40.02019	50.9
98.68057	39.6

33.11897	11.8
67.62508	95.5
105.5818	69.2
36.56958	20.9
46.92141	76.1
43.4708	48.8
95.22996	161.5
57.27324	57.2
157.341	119.4
29.66836	4.4
119.3842	133.3

RMSE - 28.499

V - CONCLUSION

The results show a winner but not by much. Both algorithms worked in a pretty efficient way, but the one done by framework outdid the one by hand for a bit.

The RMSE in the algorithm done with Sci-kit is smaller (28.499) than its competition (31.005), therefore showing that using Sci-Kit for linear regression, in this case, is the best option available.

VI - REFERENCES

[1] Brownlee, J. (2017). *Machine Learning Algorithms from scratch: With Python*. Place of publication not identified: Jason Brownlee.

[2] Robinson, S. (2018). Linear Regression in Python with Scikit-Learn [Web log post]. Retrieved November 7, 2020, from https://stackabuse.com/linear-regression-in-python-with-scikit-learn/

[3] [Swedish Auto Insurance Dataset]. (n.d.). Unpublished raw data. Obtained from the Department of Mathematics and Statistics of the Faculty of Science from the Masaryk University in the Czech Republic. Retrieved from: https://www.math.muni.cz/~kolacek/docs/frvs/M7222/data/AutoInsurSweden.txt