Artificial Intelligence (DT8012) Laboration 4 Report

Learning

Andreas Häggström

January 3, 2020

Contents

1 Introduction

This lab-report is a part of my studies in the course *Artificial Intelligence* , **DT8012** at Halmstad University. The purpose of this assignment is an increased knowledge and understanding of supervised *Learning algorithms*. The main focus in this report is the algorithm *K-Nearest Neighbors* described in Section 2. This is followed by Section 3 where the task is to use vehicle diagnostic data for both classification and regression learning. Next, in Section 4 the task is to use stored actions of poker games for classification learning.

2 K-Nearest Neighbors

Table lookup is the simplest form of instance based learning, in which one simply stores all training examples in a lookup table. When asked for a item $h(\mathbf{x}$, the corresponding $\mathbf{y}$ is returned if the table contains $\mathbf{x}$. Otherwise a default value is returned which means that this method doesn't generalize well [1].
K-Nearest Neighbors, **KNN** is a improvement of the table lookup method, for a given $\mathbf{x}$ that is not in the lookup table, a corresponding $\mathbf{y}$ is estimated through either *classification* or *regression*. The estimation is performed by comparing the distances of the various input data to the data points within the training set. With k being the amount of neighboring data points to compare to. It is of most importance that the value of k always is an odd number, this to avoid having 50% of each class as neighbors.

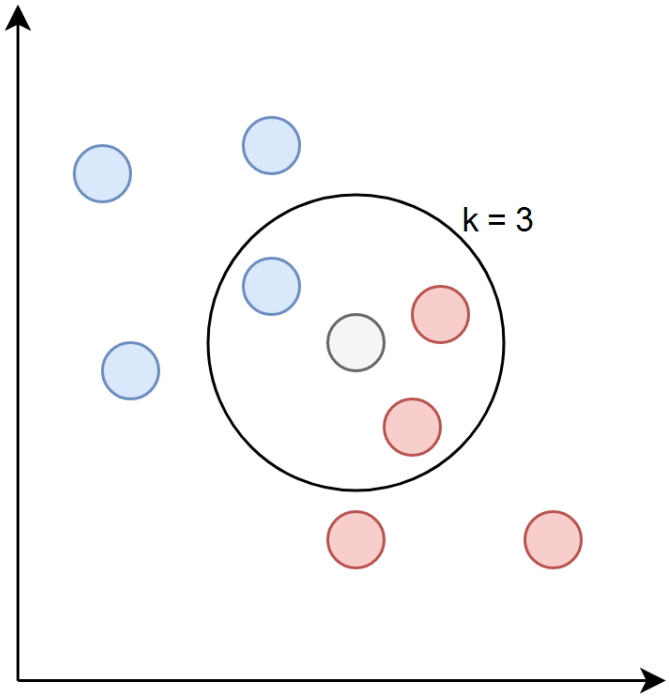

Figure 1: K-Nearest Neighbors classification with $k = 3$

In a data set with two classes, blue and red, a new data point is to be classified as one of them. With $k = 3$ as shown in figure 1, the new data point is to be classified as red according to KNN since the majority of the k neighboring nodes are red.
The distance between data points in KNN is usually measured in *Minkowski distance*, or L^p norm [2] as shown in equation 1.

$$L^p(x_j, x_q) = (\sum_i |x_{j,i} - x_{q,i}|^p)^{1/p} \tag{1}$$

With $p = 1$ this will result in *Manhattan* distance while a value of $p = 2$ will result in the *Euclidean* distance [3]. With the choice of euclidean distance being the most common.

Mainly because of curiosity, this report also contains implementation of both *Cosine* and *Chebyshev* distance for comparison. Cosine distance is measured through comparing the angle between the vectored form of two data points. It is evaluated as shown in equation 2.

$$Cos(\theta) = \frac{\vec{v_1} \cdot \vec{v_2}}{||\vec{v_1}|| \times ||\vec{v_2}||} \tag{2}$$

Cosine distance is popular when it comes to comparing text documents, this since as described in [4], cosine distance still works if the documents are of different lengths.
Chebyshev distance [5] is a more simple distance measurement and is evaluated through equation 3.

$$D(x_j, x_q) = Max|x_{j,i} - x_{q,i}|, i = 1, 2, 3... \tag{3}$$

3 Task-1

The data set of the first task is named *Lab4Data.csv* and contains 2000 samples of various signals and measurements.

- Acceleration
- Gender
- Age
- LateralAcceleration
- VehicleSpeed
- GearChanges
- EngineSpeed
- FuelConsumption
- Weight
- DriverPerformance

For the tasks regarding classification the target value is *DriverPerformance*. Where regression is performed, the *FuelConsumption* is the target value instead.

3.1 Task-1a

To implement a KNN classification algorithm, one first separates the data into *train* and *test* sets. Then one evaluates the train data to generate a lookup table to be used for the test data. To perform the test, the distance between the various data points are measured as described in section 2.

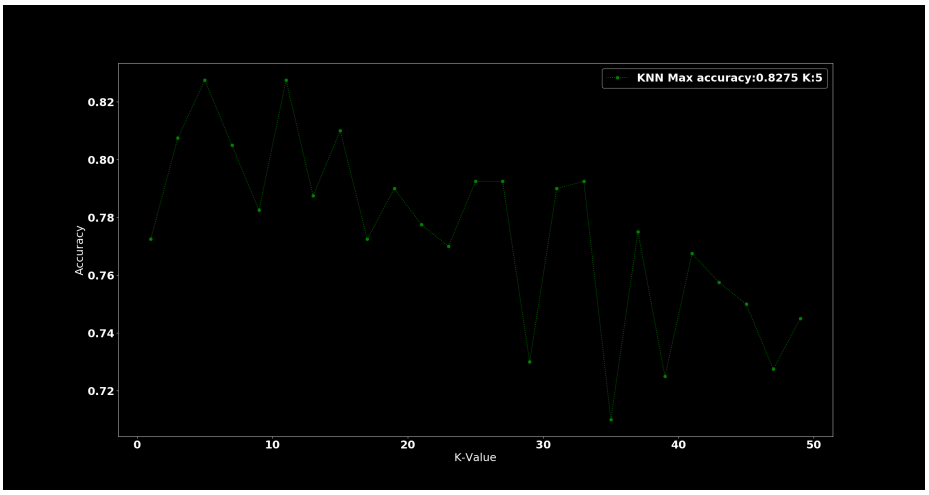

Figure 2: Result of KNN using euclidean distance

The resulting accuracies for various values of k is shown in figure 2. As seen, the optimal value for k is 5 according to this evaluation. It should though be noted that this optimal k varied between evaluations but was for the most part between 5 and 9.

3.2 Task-1b

The different classification algorithms chosen were the following.

- Multi-layer Perceptron, MLP [6]
- Decision Tree [7]
- Random Forest [8]

These were implemented using the python libraries of **Sklearn v.0.22**[9]. As seen in figure 3, the results of the different methods varied. The MLP implementation had the lowest accuracy and also the worst precision of the three. The Decision tree had the best precision and the second best accuracy right after the Random Forest implementation.

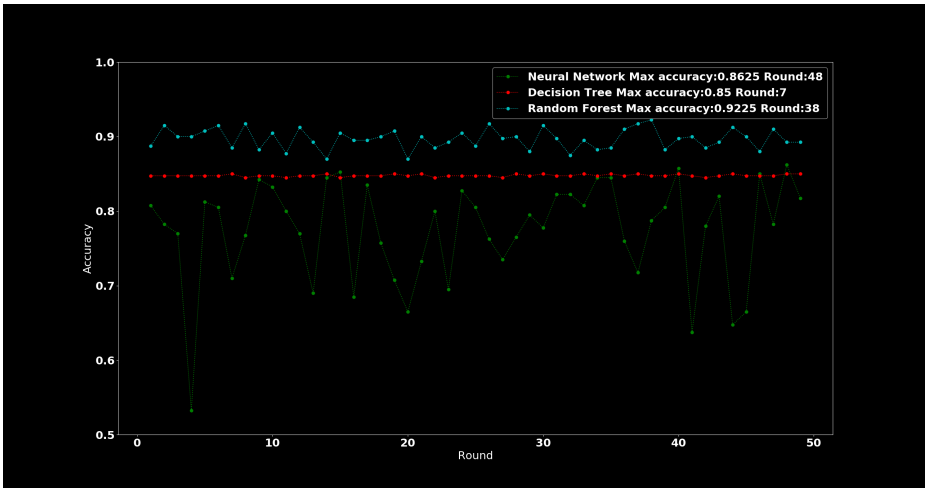

Figure 3: Result of Sklearn classification

3.3 Task-1c

The KNN-classification of Sklearn was implemented for a parameter tuning test, this means that one changes the amount of neighbors (the value of **k**). The different distance measurements are also compared as seen in figure 4. The evaluations were compared using cross validation [10]. As one can see, the methods did not vary drastically but both seem to have a optimal value of **k** below 9.

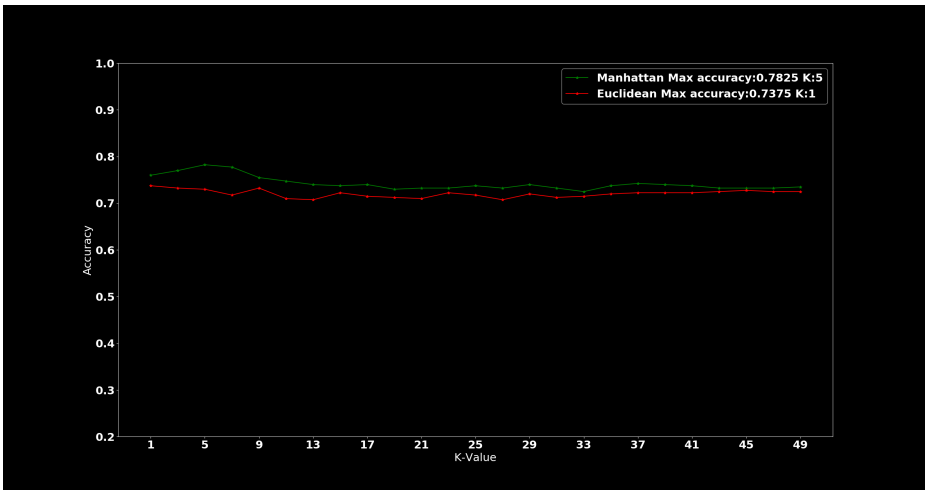

Figure 4: Results of KNN Parameter tuning, cross validation

A second test was also conducted, with the average accuracy of the KNN-classification of SKlearn compared to the custom one. The result of this can be seen in figure 5.

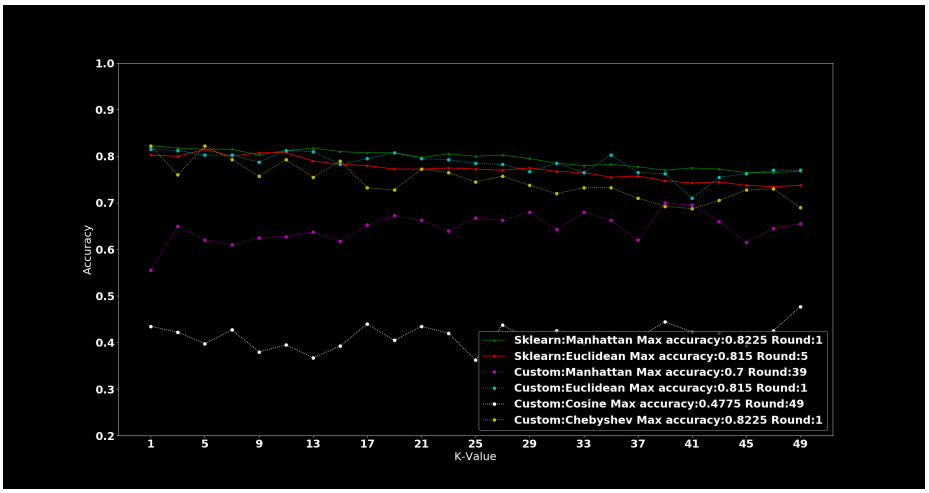

Figure 5: Result of Sklearn classification

3.4 Task-1e

The implementation of KNN-regression is similar to the one of classification. The difference is that one now wishes to estimate a continuous value instead of a discrete class. Instead of choosing the most occurring class, one chooses the mean value of the **k**-neighbors as the expected output value $\hat{y}$ as seen in equation 4. Where d_i is the distance to the ith neighbor and x_i is its value [11]. The result of the evaluations of various **k** values can be seen in figure 6. The accuracy was evaluated through equation 5 with p being the predicted value for $\hat{y}$ and a being the actual value.

$$\hat{y} = \frac{\sum_{i=1}^{k} \frac{1}{d_i} * x_i}{\sum_{i=1}^{k} \frac{1}{d_i}} \tag{4}$$

$$A = 1 - \frac{\sum_{i=1}^{n} \frac{p-a}{a}}{n} \tag{5}$$

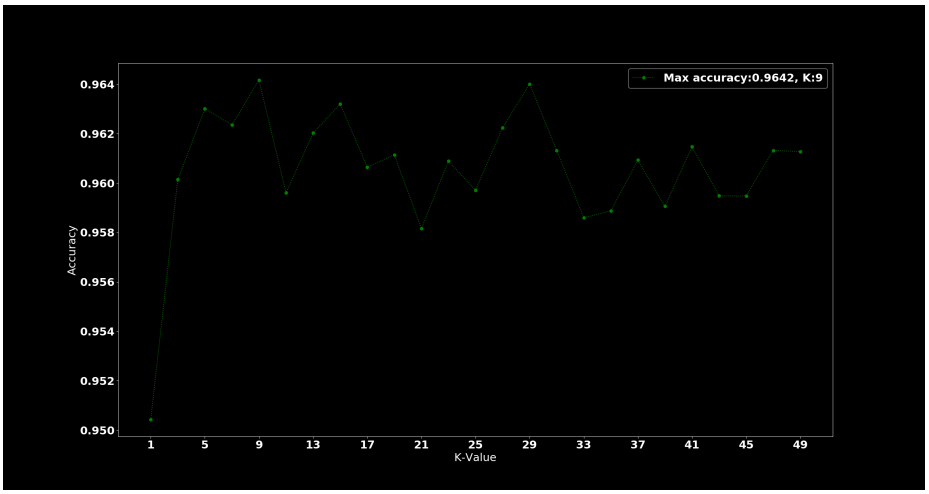

Figure 6: Result of KNN using euclidean distance

3.5 Task-1f

For this task, the same methods were chosen as for the classification in section 3.3. With the difference being that now the regression versions were used. The result of the evaluations can be seen in figure 7. The *Accuracy* of the implementations in this case is instead measured in R^2 score [12]. This measurement explains how well the model fit the data instead of how accurate of a result it achieves [13]. The MLP implementation has a unstable result while the Decision Tree and Random Forest implementations are more stable.

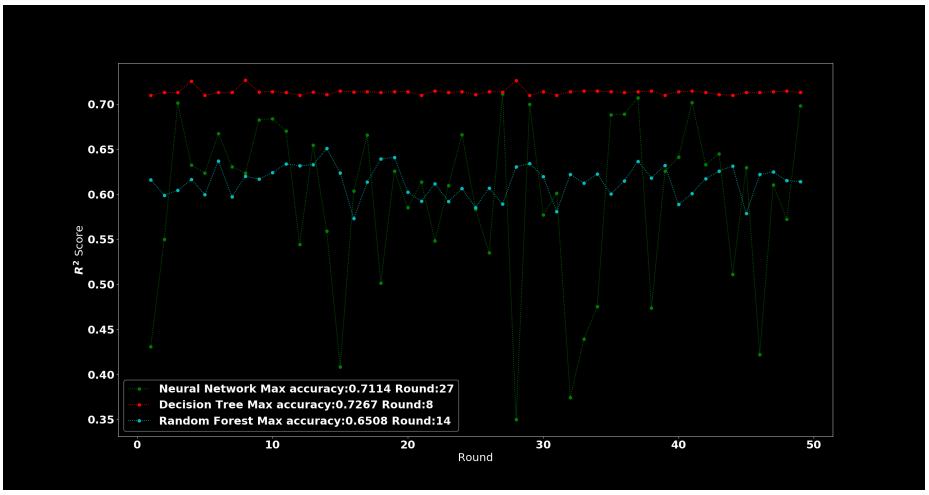

Figure 7: Result of KNN using euclidean distance

3.6 Task-1g

The result of cross validation of the SKlearn KNN-regression can be seen in the 8. Here it can be seen that the model does not achieve a desirable result. The implementations does not fit the data well, this is probably since KNN is a relatively simple regression implementation. For this kind of evaluation, a more complex implementation might be required.

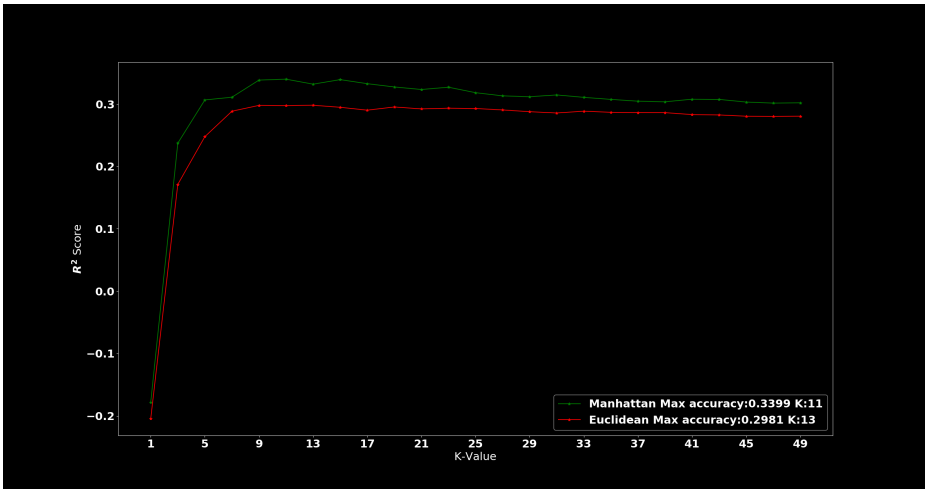

Figure 8: Result of KNN using euclidean distance

4 Task-2

For this task, classification methods are to be implemented in the purpose of predicting an opponents actions in poker. With a dataset *Lab4PokerData.txt* consisting of 200 poker game samples. Each sample contains the player hands, initial money amount as well as their actions.

4.1 Task-2a

For this task, two or more features are to be chosen for the classification to be possible. The ones chosen were the following.

- **Amount of money left**
  This since it would affect e.g if a player has the possibility to raise
- **Players average bet**
  To see how much the player tends to bet
- **Players relative aggressiveness**
  To see if a player consider themselves to be in a spot to be aggressive and bet higher than the opponent

The data was then made numerical for a classification method to be able to understand it. This was done by creating a lookup-table for values of the various hand types, card types etc. A sample of the data can be seen in figure **??** and the numerical version can be seen in table figure **??**.

Player1 Hand	Player2 Hand	P1P2 Action	P1P2 Action
TwoPairs J 35	straightflush 4 406	Raise 5 Raise 10	Raise 20 Call 20

Figure 9: Poker data

P1 Hand	P1 Money	P2 Hand	P2 Money	P1,P2 Money Left	P1,P2 Av.Bet	P1,P2 Agg	P2 Final Action
2 , 9	35	8 , 2	406	10 , 396	12.5 , 10	1.25 , 0.8	1

Figure 10: Numerical version of data

4.2 Task-2b

The results of the classification evaluations can be seen in figure 11 here the KNN-implementation is set to use the distance as weights for the data points. The other methods chosen were Decision Tree and Random Forest, which both generally achieved a higher accuracy than the KNN-implementation. Though the Random Forest had a worse precision than the other two, it performed better than the KNN most of the evaluations.

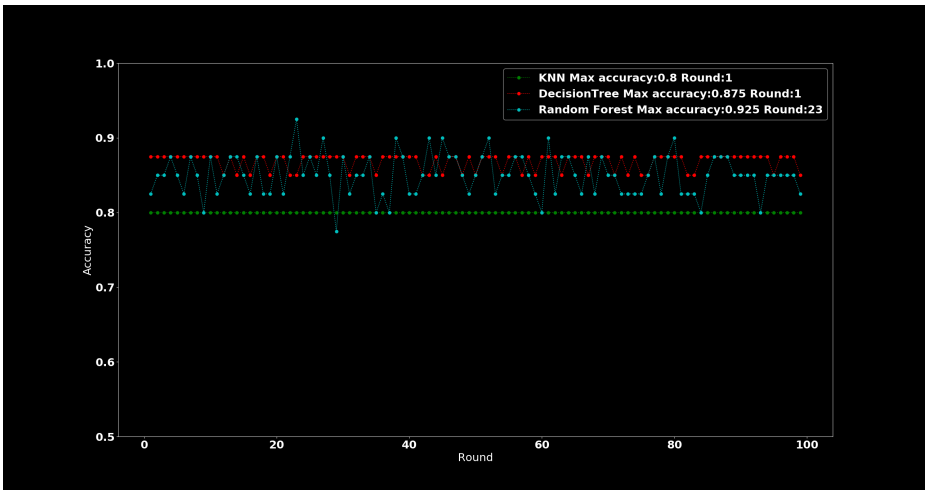

Figure 11: Result of KNN using distance as weight

4.3 Task-2c

In figure 12, the results of the various parameters can be seen. The accuracy was evaluated using cross validation. Similar to section 3.3 the result did not vary much depending on the distance method. What is interesting is that both the accuracy was quite dependant of the value of **k** as expected. Both methods had the same optimal value of **k** as well.

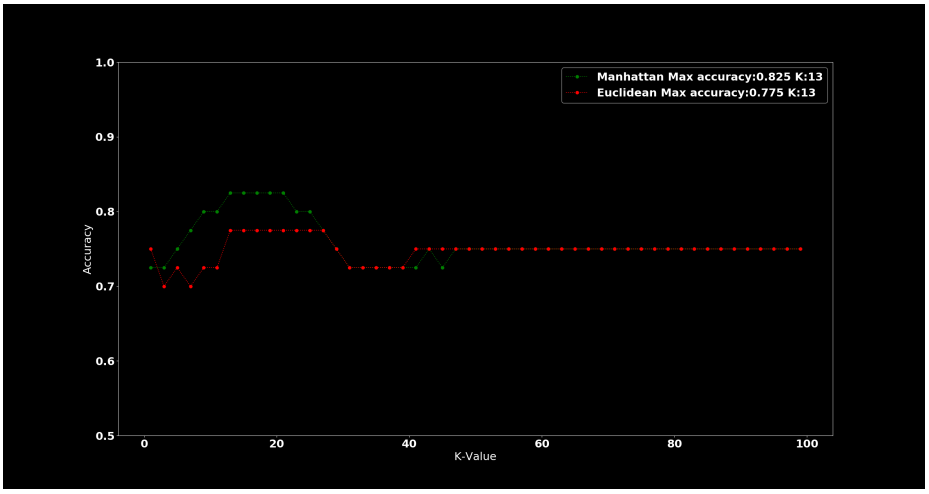

Figure 12: Result of KNN using cross validation

References

[1] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition, 2009. 737.

[2] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition, 2009. 738–739.

[3] Archana Singh, Avantika Yadav, and Ajay Rana. K-means with three different distance metrics. *International Journal of Computer Applications*, 67(10), 2013.

[4] Anna Huang. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, volume 4, pages 51–52, 2008.

[5] Nikolai Krivulin. An algebraic approach to multidimensional minimax location problems with chebyshev distance. *arXiv preprint arXiv:1211.2425*, 2012.

[6] Dennis W Ruck, Steven K Rogers, and Matthew Kabrisky. Feature selection using a multilayer perceptron. *Journal of Neural Network Computing*, 2(2):40–48, 1990.

[7] Yoav Freund and Llew Mason. The alternating decision tree learning algorithm. In *icml*, volume 99, pages 124–133, 1999.

[8] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[9] Scikit-learn. url: https://scikit-learn.org/stable/.

[10] Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in neural information processing systems*, pages 231–238, 1995.

[11] Yunsheng Song, Jiye Liang, Jing Lu, and Xingwang Zhao. An efficient instance selection algorithm for k nearest neighbor regression. *Neurocomputing*, 251:26–34, 2017.

[12] Scikit-learn. sklearn.metrics.r2$_s$*core*. *url* : *https* : *//scikit* − *learn.org/stable/modules/generated/sklearn.metrics.r*2$_s$*core.html*.

[13] Jim Frost. How to interpret r-squared in regression analysis. accessed: 2020-01-03, url: https://statisticsbyjim.com/regression/interpret-r-squared-regression/.