# CRIME RATE AND SCALED SOUND PRESSURE PREDICTION USING FOUR REGRESSION MODELS

By: Adel Samir Saleh ElZemity

## 1. Introduction

Machine learning, especially regression techniques, has been widely used to estimate a variety of variables in a variety of fields by establishing a relationship between the variable to be estimated and a set of other variables known as independent variables. Two different data sets were used in this project to predict two different output variables using supervised machine learning. Based on the first dataset, the number of violent crimes per 100K population needs to be estimated by considering 120 predictive features. The second predicted variable is the Scaled sound pressure level by looking at frequency, Angle of attack, Chord length, Free-stream velocity, and Suction side displacement thickness. For both datasets, four different regression algorithms were used which are:

1- Linear Regression
2- K-Nearest neighbors Regression
3- Decision Trees Regression
4- Random Forest Regression

## 2. Theory

We will provide a brief theoretical description of each of the algorithms listed previously in this section. We will use x to denote input variable(s) or feature(s) and y to denote the output variable, also known as the response or dependent variable, in this section and the rest of the article.

### *2.1. Linear Regression*

One of the oldest and most widely used regression methods is linear regression. It is also regarded as one of the most basic. The relationship between the features and the answer is assumed in multiple linear regression. As a result, the traditional linear regression model can be written as:

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{m} X_j \hat{\beta}_j, \tag{1}$$

Where $\hat{Y}$ is the output variable, $X_j$is the $jth$ feature out of the features, $\hat{\beta}_0$ is the intercept, also known as the bias, and finally $\hat{\beta}_j$ is the $jth$ coefficient, also known as the feature's weight, of the $jth$ feature.

There is only one problem for this model to be properly described, and that is to find the coefficients $\hat{\beta}$, also known as fitting the model. There are several different methods, but the least square method is one of the most common. We choose the coefficients in this method to minimize the residual number of squares, which is:

$$RSS = \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i\right)^2, \tag{3}$$

Where RSS is the previously stated residual number of squares, $Y_i$ is the dataset's true output variable, and $Y_i$ is the model's predicted output variable. The model aims to find the best coefficients for minimizing the RSS.

### *2.2. K-Nearest Neighbors Regression*

K-Nearest Neighbors is a basic machine learning algorithm that predicts the results based on the "nearest" K examples in the training dataset. The Euclidean distance is the most often used metric to calculate the closeness of the features in the training set $X_j$. The projection is computed by calculating the mean of all these points after computing the distance and finding the nearest K-points:

$$\hat{Y} = \frac{1}{k}\sum_{i=1}^{K} Y_i, \tag{4}$$

An example of K-NN classification is shown in Figure 1. The test sample (green dot) should be classified as either too blue squares or too red triangles. If k = 3 (solid line circle) it is assigned to the red triangles because there are 2 triangles and only 1 square inside the inner circle. If k = 5 (dashed line circle) it is assigned to the blue squares (3 squares vs. 2 triangles inside the outer circle).

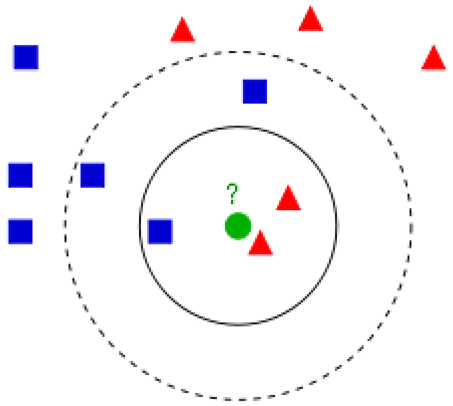

**Fig. 1.** Example of k-NN classification

### *2.3. Decision Trees Regression*

For classification and regression, Decision Trees (DTs) are a non-parametric supervised learning process. The aim is to learn basic judgment rules from data features to build a model that forecasts the value of a target variable. A tree is an approximation to a piecewise constant.

In the example below, decision trees use a set of if-then-else decision rules to approximate a sine curve using data. The judgment laws become more complicated as the tree grows deeper, and the model becomes more accurate.

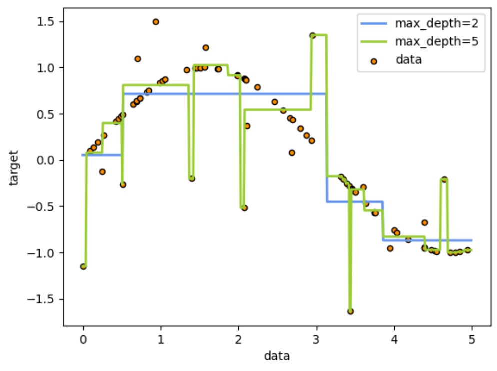

**Fig. 2.** Decision Tree Algorithm

### *2.4. Random Forest Algorithm*

Random Forest Regression is a supervised learning algorithm for regression that employs the ensemble learning process. The ensemble learning approach incorporates predictions from many machine learning models to provide a more reliable prediction than a single model.

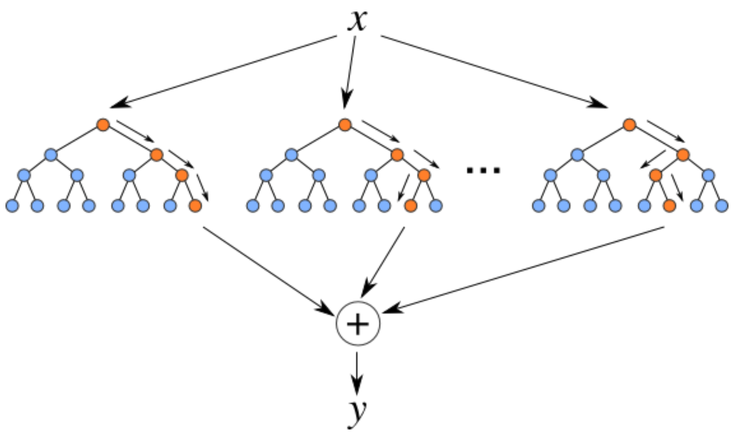

**Fig. 3.** Random Forest Algorithm

$R^2$ score tells us how well our model is fitted to the data by comparing it to the average line of the dependent variable. If the score is closer to 1, then it indicates that our model performs well versus if the score is farther from 1, then it indicates that our model does not perform so well.

## 3. Overview of the used datasets

### *3.1. Communities and Crime Dataset*

Communities within the United States. The data combines socio-economic data from the 1990 US UCR.

Number of Instances: 1994
Number of Attributes: 128 (122 predictive, 5 non-predictive, 1 goal)

The dataset has NULL values in some features, so these features were not included in the algorithm. The non-predictive attributes were also cleared.

The output of this dataset is “ViolentCrimesPerPop”: total number of violent crimes per 100K population.

Examples of the features used:

- medIncome: median household income
- blackPerCap: per capita income for African Americans
- TotalPctDiv: percentage of the population who are divorced.

### *3.2. Airfoil self-noise Dataset*

NASA data set, obtained from a series of aerodynamic and acoustic tests of two and three-dimensional airfoil blade sections conducted in an anechoic wind tunnel. The NASA data set comprises different size NACA 0012 airfoils at various wind tunnel speeds and angles of attack. The span of the airfoil and the observer position was the same in all of the experiments.

Number of Instances: 1503
Number of Attributes: 6

## 4. Experiments

To clean and wrangle the data easily, Jupyter notebooks are used to manage and apply the regression experiments. A couple of libraries were used to help manage and clean the datasets like pandas, NumPy, and other built-in functions. Another library was used to help visualize the results: matplotlib. Finally, along the way to apply regression methods, some libraries like sklearn were utilized.

For these experiments, an $R^2$ score was used. R-squared is a statistical measure that represents the goodness of fit of a regression model. The ideal value for r-square is 1. The closer the value of r-square to 1, the better is the model fitted.

The methodology for the experimenting was as following:

1. Preparing dataset for training by cleaning, optimizing, and feature selection.
2. Applying the following regression algorithms using the cross-fold validation method
    a. Linear Regression
    b. K-Nearest Neighbors
    c. Decision Trees
    d. Random Forest
3. Considering Tuning algorithms by changing the hyperparameters if existed.
4. Visualization of the results by plotting some graphs.

### *4.1. Experiments with Communities and Crime Dataset*

For the following table, the four above mentioned regression model were applied using the default hyperparameters and then comparing the mean of $R^2$ and the mean of fit time as following:

**Table 1:** Comparing $R^2$ score & fit time across all algorithms

| Regression Method | Linear Regression | K-NN Regression | Decision Trees | Random Forest |
|---|---|---|---|---|
| $R^2$ Score | 0.653961155 | 0.0687211055 | 0.2426246529 | 0.6265594 |
| Fit Time | 0.025667881 | 0.0141124010 | 0.1449150800 | 1.6696850 |

According to the data summarized in the previous table, for this dataset, Linear Regression is the best regression model to adopt as it has the highest R-squared score with the relatively lowest fit time compared to other regression methods.

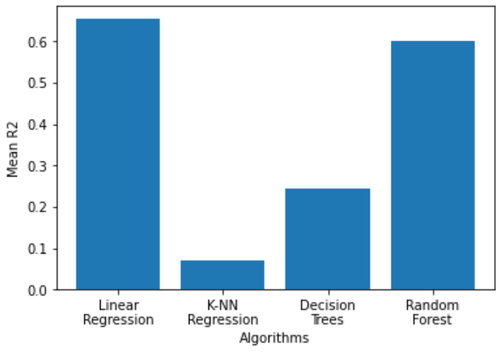

**Fig. 4.** Comparing Mean $R^2$ between algorithms

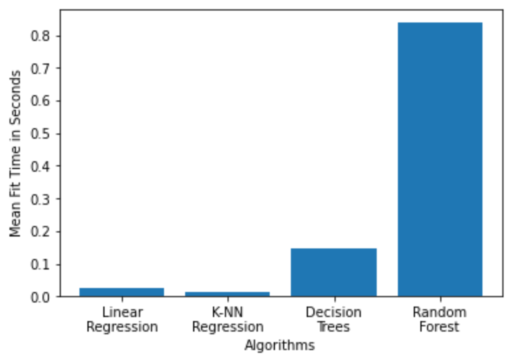

**Fig. 5.** Comparing Mean Fit Time between algorithms

#### *4.1.1. K-NN Algorithm Tuning*

By changing the hyperparameters in any algorithm, both the accuracy and the fitting time change. For K-NN, the number of neighbors was changed, and the R-squared score and fit time were calculated and compared as shown in the following graphs:

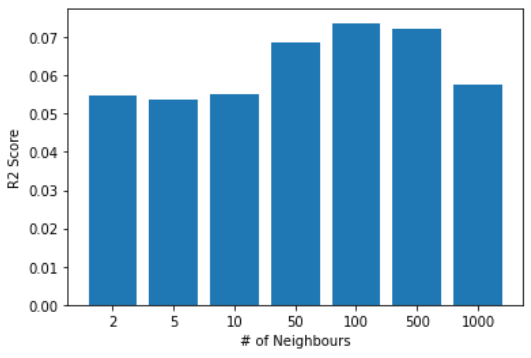

**Fig. 6.** Changing # of neighbors vs change in $R^2$

It is clear that at the beginning there is underfitting and as the number of neighbors starts to approach 100, the accuracy gets to it is optimum. However, when the number of neighbors increases more, overfitting starts to happen, causing the accuracy to go down.

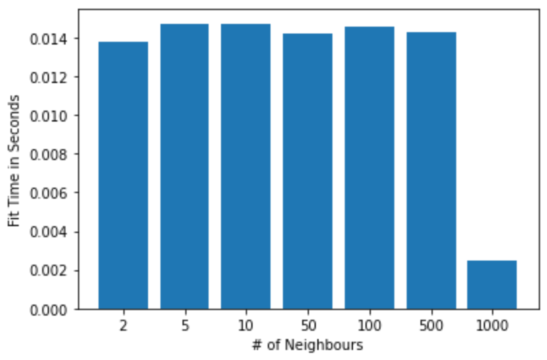

**Fig. 7.** Changing # of neighbors vs change in fit time

As the K-NN algorithm has no fitting methodology as explained in the theory section, it's normal to see that the fit time approaches zero or the lowest one compared to other algorithms.

***4.1.2. Decision Trees Tuning***

By changing the hyperparameters in any algorithm, both the accuracy and the fitting time change. For Decision Trees, the max depth was changed, and the R-squared score and fit time were calculated and compared as shown in the following graphs:

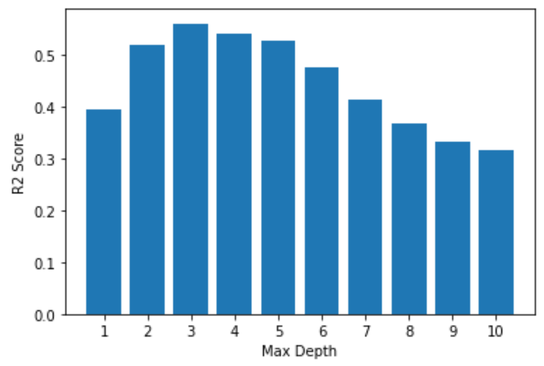

**Fig. 8.** Changing max depth vs change in R$^2$

Increasing the max depth for the decision trees did more harm than good to the R-squared score, so a max depth of 3 is optimum.

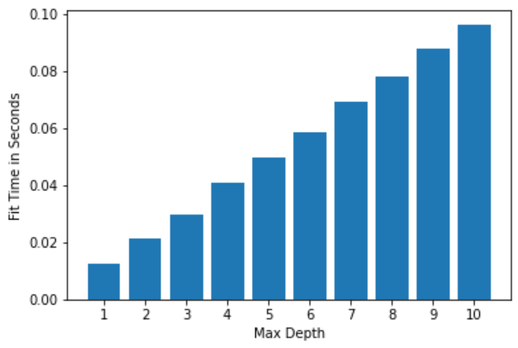

**Fig. 9.** Changing max depth vs change in fit time

As mentioned above, it is good to find the minimum max depth value (3) which will serve as the optimum value because this model adopts a linear increase in the fitting time.

***4.1.3. Random Forest Tuning***

By changing the hyperparameters in any algorithm, both the accuracy and the fitting time change. For Random Forest, the number of estimators was changed, and the R-squared score and fit time were calculated and compared as shown in the following graphs:

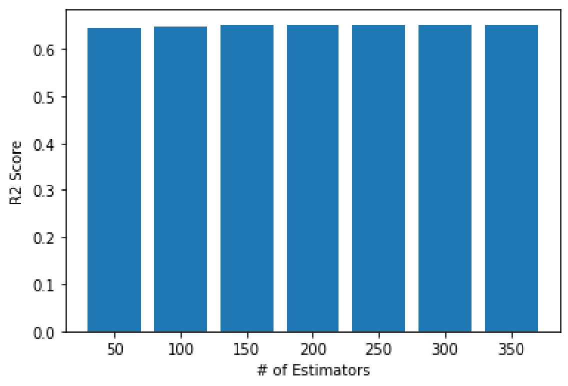

**Fig. 10.** Changing # of estimators vs change in R$^2$

As clearly seen in the previous figure, changing the number of estimators did not make much difference in the long term. However as seen in the following figure, it drastically affected the fitting time. Thus, choosing a minimum number and an optimum number of estimators is essential in this case.

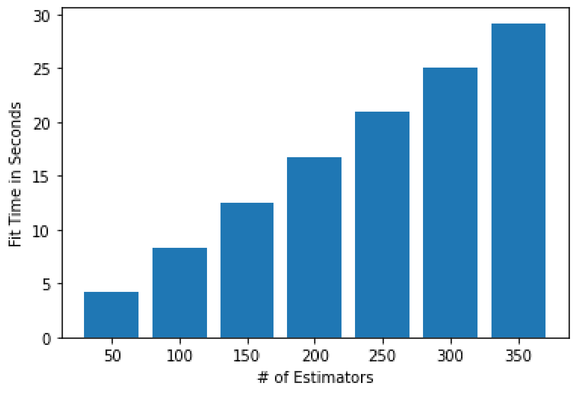

**Fig. 11.** Changing # of estimators vs change in fit time

## *4.2. Experiments with Airfoil self-noise Dataset*

For the following table, the four previously mentioned regression models were applied using the default hyperparameters and then comparing the mean of $R^2$ and the mean of fit time as following:

**Table 1:** Comparing $R^2$ score & fit time across all algorithms.

| Regression Method | Linear Regression | K-NN Regression | Decision Trees | Random Forest |
|---|---|---|---|---|
| $R^2$ Score | 0.500249 | 0.0 | 0.8590372 | 0.9245514 |
| Fit Time | 0.003060 | 0.002562 | 0.0062903 | 0.0389075 |

According to the data summarized in the previous table, for this dataset, Random Forest is the best regression model to adopt as it has the highest R-squared score with relatively highest fit time compared to other regression methods, however, it is still low fit time, so it is the best option.

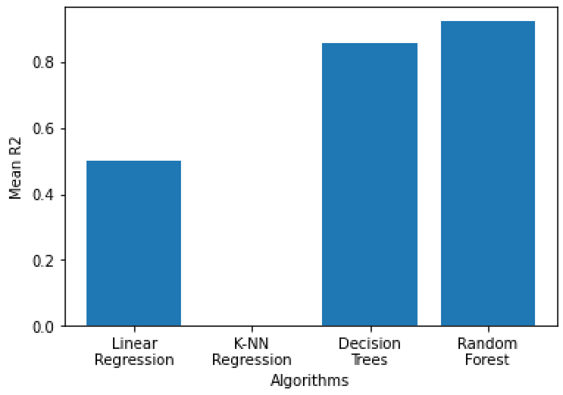

**Fig. 12.** Comparing Mean $R^2$ between algorithms

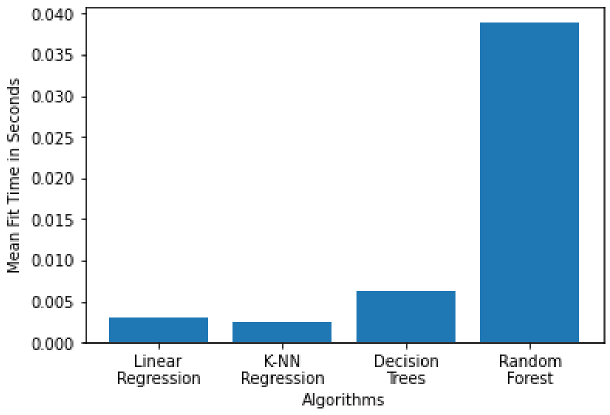

**Fig. 13.** Comparing Mean Fit Time between algorithms

### *4.1.1. K-NN Algorithm Tuning*

By changing the hyperparameters in any algorithm, both the accuracy and the fitting time change. For K-NN, the number of neighbors was changed, and the R-squared score and fit time were calculated and compared as shown in the following graphs:

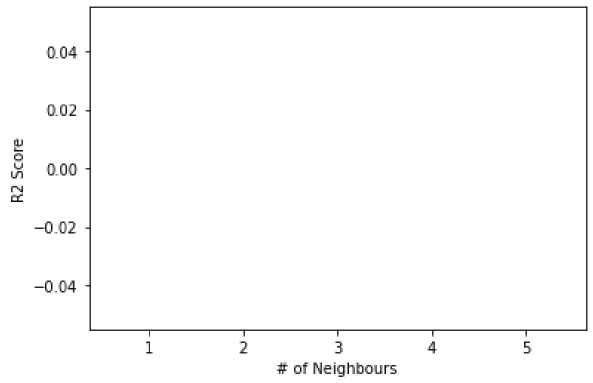

**Fig. 14.** Changing # of neighbors vs change in $R^2$

As seen in the figure that either by changing the number of neighbors, the K-NN algorithm is not the right fit for this dataset.

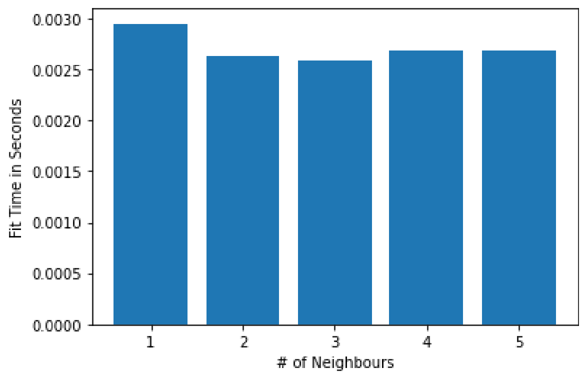

**Fig. 15.** Changing # of neighbors vs change in fit time

As the K-NN algorithm has no fitting methodology as explained in the theory section, it's normal to see that the fit time approaches zero or the lowest one compared to other algorithms.

***4.1.2. Decision Trees Tuning***

By changing the hyperparameters in any algorithm, both the accuracy and the fitting time change. For Decision Trees, the max depth was changed, and the R-squared score and fit time were calculated and compared as shown in the following graphs:

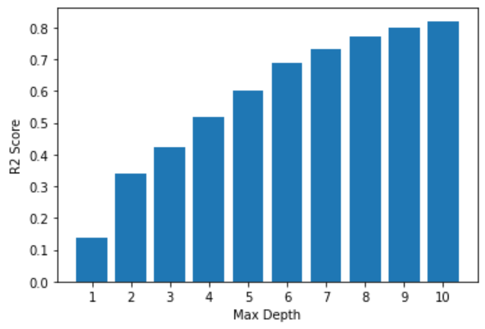

**Fig. 16.** Changing max depth vs change in $R^2$

Increasing the max depth for the decision trees did increase the R-squared score, however, it seems like it started to saturate around 10 as max depth.

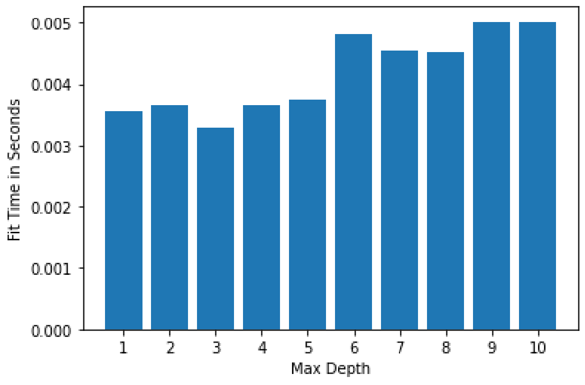

**Fig. 17.** Changing max depth vs change in fit time

As the time variation between 0.004 and 0.005 is so small, it’s expected that the random variation happening is because of the hardware spikes.

***4.1.3. Random Forest Tuning***

By changing the hyperparameters in any algorithm, both the accuracy and the fitting time change. For Random Forest, the number of estimators was changed, and the R-squared score and fit time were calculated and compared as shown in the following graphs:

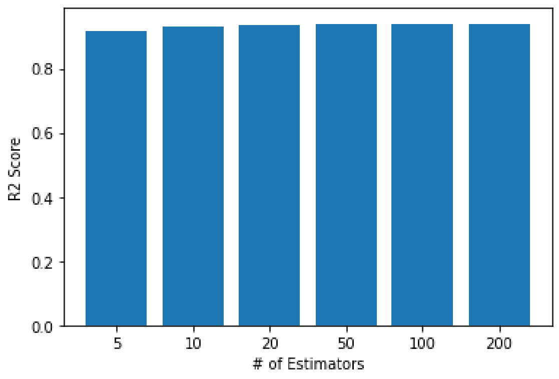

**Fig. 18.** Changing # of estimators vs change in $R^2$

As clearly seen in the previous figure, changing the number of estimators did not make much difference in the long term. However as seen in the following figure, it drastically affected the fitting time. Thus, choosing a minimum number and an optimum number of estimators is essential in this case.

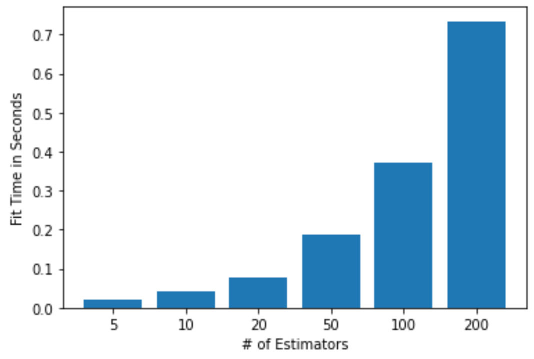

**Fig. 19.** Changing # of estimators vs change in fit time

### *4.3. Comparison of Results*

When comparing both datasets, it is important to list the contents of both datasets as shown in the following table.

**Table 2:** Comparing between datasets

| Dataset | 1st: Communities and Crime | 2nd: Airfoil Self-Noise |
|---|---|---|
| Number of instances | 1994 | 1503 |
| Number of Features | 120 | 6 |

It was expected for the first dataset to have a bigger fitting time considering it has much more features and instances than the second dataset.

In the following table, the accuracy score used "$R^2$" is compared between both datasets for the four algorithms.

**Table 3:** comparing the two datasets by $R^2$ score with the four regression methods.

| Regression Method/Dataset | Linear Regression | K-NN Regression | Decision Trees | Random Forest |
|---|---|---|---|---|
| 1st: Communities and Crime | 0.653961155 | 0.0687211055 | 0.2426246529 | 0.6265594 |
| 2nd: Airfoil Self-Noise | 0.500249 | 0.0 | 0.8590372 | 0.9245514 |

As it is clearly shown in the previous table, the performance of the Random Forest Regression Method is high for both datasets. However, in the second dataset, the Random Forest accuracy is much higher considering the lower feature advantage to the second dataset.

In addition, the Linear Regression seems to be performing the best in the first dataset. What is odd is the behavior of decision trees with both datasets. In the first dataset, it got bad results, however, in the second dataset it got better results. This may be because of the feature selection that directly relates the independent variables to the dependent variable in the second dataset.

## 5. References

Avinash Navlani, “Decision Tree Classification in Python,” *DataCamp Community*. [Online]. Available: https://www.datacamp.com/community/tutorials/decision-tree-classification-python. [Accessed: 03-May-2021].

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research,* 01-Jan-1970. [Online]. Available: https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html. [Accessed: 03-May-2021].

J. Brownlee, “A Gentle Introduction to k-fold Cross-Validation,” *Machine Learning Mastery,* 02-Aug-2020. [Online]. Available: https://machinelearningmastery.com/k-fold-cross-validation/. [Accessed: 03-May-2021].

J. Brownlee, “Linear Regression for Machine Learning,” *Machine Learning Mastery,* 14-Aug-2020. [Online]. Available: https://machinelearningmastery.com/linear-regression-for-machine-learning/. [Accessed: 03-May-2021].

Jnikhilsai, “Cross-Validation with Linear Regression,” *Kaggle*, 20-Aug-2019. [Online]. Available: https://www.kaggle.com/jnikhilsai/cross-validation-with-linear-regression. [Accessed: 03-May-2021].

R. Dwivedi, “How does K-nearest Neighbor Works in Machine Learning: KNN algorithm,” *How does K-nearest Neighbor Works in Machine Learning | KNN algorithm*. [Online]. Available: https://www.analyticssteps.com/blogs/how-does-k-nearest-neighbor-works-machine-learning-classification-problem. [Accessed: 03-May-2021].

T. Hastie, J. Friedman, and R. Tisbshirani, *The Elements of statistical learning: data mining, inference, and prediction*. New York: Springer, 2017. http://statweb.stanford.edu/~tibs/ElemStatLearn/

Ynouri, “Random Forest & K-Fold Cross Validation,” *Kaggle*, 24-Jun-2018. [Online]. Available: https://www.kaggle.com/ynouri/random-forest-k-fold-cross-validation. [Accessed: 03-May-2021].