# Introduction

Machine learning is an implementation of artificial intelligence (AI) that gives systems the ability to automatically learn and develop from experience without being specifically programmed. Machine learning focuses on the creation of computer systems that can access data and use it to learn about themselves.

In order to search for similarities in data and make informed choices in the future based on the examples we have, the learning process starts with findings or data, such as examples, direct experience, or guidance. The primary objective is to allow computers to learn automatically and adjust actions accordingly without human involvement or assistance (“expert.ai”, 2020).

To forecast future events, supervised machine learning algorithms may apply what has been observed in the past to new data using labelled examples. The learning algorithm generates an inferred function to make assumptions about the output values beginning from the study of an established training dataset. After ample preparation, the device is able to provide targets with any new input. In order to adjust the model accordingly, the learning algorithm can also compare its output with the correct, expected output and identify errors (“expert.ai”, 2020).

In this assignment, I am going to attempt to apply machine learning into predicting the quality or condition of the ball grid array x-ray scan.

|  |  |
| --- | --- |
| Effect of the BGA | example |
| Noisy |  |
| ring |  |
| Sun |  |
| Good |  |
| Popcorn |  |
| Nothing |  |
| distorted |  |

Table 1: Image Classes

For a machine learning model to work, there are a few steps that have to be taken to achieve the best results possible. The steps are as follows:

1. **Gathering Data.**
2. **Data preparation.**
3. **Data Wrangling.**
4. **Analyse Data.**
5. **Train the model.**
6. **Test the model.**
7. **Deployment.**

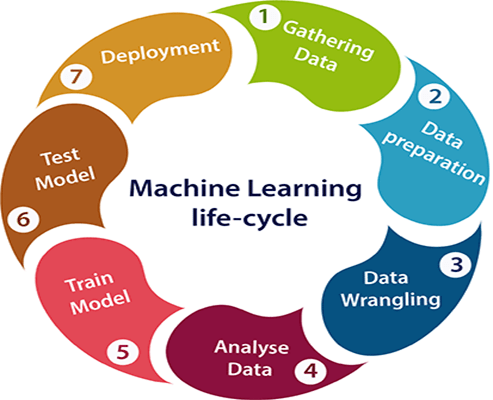


Figure : Machine Learning Process

First, we have to collect the data required to feed into our machine learning model. The rule of thumb is the more data we have, the more accurate the prediction made by the machine learning model will be. Thus, it is important to identify different data sources to maximise the amount of dataset we have. After collecting all the data, data preparation means that we have to put all the data in one place.

Next, we have to undergo data wrangling, which is the process of cleaning and converting raw data into a format understandable to our machine learning model. Every machine learning model requires two types of data, features and labels. Features are like manipulating variables which affect our labels (responding variable). However unlike scientific experiments, these features do not have any scientific or mathematical relationship with our targets because if it did, then we would be able to calculate the target right off the bat with correct targets everytime without needing to predict the target. For example, features of a plant would be it’s characteristics such as petal length, leaf types, root type, flower colour and many others, while the labels would be the different type of plants such as cactus, hibiscus, rose or sunflower. Other than that, some of our raw data may contain information that is not needed in our prediction or invalid, thus we have to also clear out those data so that the predictions can be more accurate.

After processing our raw data collected into features and labels, we now have to determine which kind of machine learning model that we have to use to achieve the best results, and that’s data analysis coming into play. The factors we have to consider for our machine learning model are:

* **Size of training data**
  + It is a good rule of thumb to gather as much data as possible to achieve reliable predictions. However in real life, the availability of data is a constraint and is usually limited. Thus, if we have a smaller amount of training data or high number of features, it is best to choose a machine learning model that has low bias and low variance such as Naive Bayes, Linear Regression or Linear SVM. On the other hand, if the training data is large enough, then it is better to go for low bias and high variance algorithms such as KNN, Decision trees, or kernel SVM.
* **Accuracy or interpretability of the output**
  + Accuracy of a machine learning model is the measure of how close is the prediction made by the function for a given observation (features), to the true response value (target) for that observation (features). High interpretability means one is able to understand easily the function which defines the relationship between prediction and response, while a flexible machine learning model gives higher accuracy while sacrificing interpretability (Kinha, 2020).

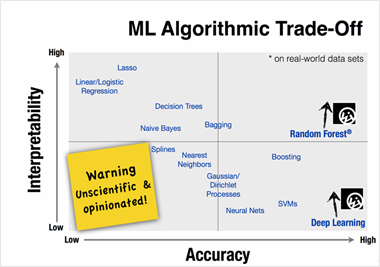


Figure : The general interpretability and accuracy of different machine learning algorithms

* **Speed or training time**
  + Algorithms require more time to train large training data, or even to achieve high accuracy on some. Algorithms themselves can be divided into two categories. The first category includes Naive Bayes, Linear and Logistic regression, which are easy to implement and run. On the other hand, algorithms like SVM, neural networks, and random forests require a lot of time to tune the parameters to achieve the best results, thus it requires a longer time to train data.

#### Linearity

* + Many algorithms work on datasets with the assumption that the classes can be separated by a straight line as shown below. These algorithms include linear, logistic regressions and support vector machines. These algorithms work best when the data is linear. However in most cases the data is not linear, thus we have to rely on other algorithms to handle high dimensional and complex data structures. These include kernel SVM, random forest, and neural networks. A good way to check whether our data is linear or not is by running it through linear or logistic regression or SVM and check for residual errors. A high error means the data is not linear and requires other complex algorithms to run it.

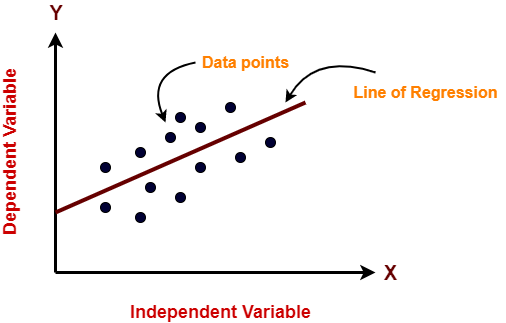


Figure : A linear classification problem

#### Number of features

* + Datasets may have a large number of features, some of which are not relevant and significant to be used to train the machine learning model. For some types of data such as texture or genetics, the number of features greatly exceeds the number of data. This slows down some algorithms, which makes training time very long. Some algorithms work well with large numbers of features and lesser observations such as SVM, but for the other algorithms it is a good way to reduce dimensionality and select important features before training the machine learning model.

# Data Visualization

First, we should be able to visualize our data points in our dataset to get a better understanding of the relationship between the data points and the statistics of data points for each feature. For this purpose, I will show the box and whisker plots, histogram and scatter matrix of the datasets.

## Box Plot

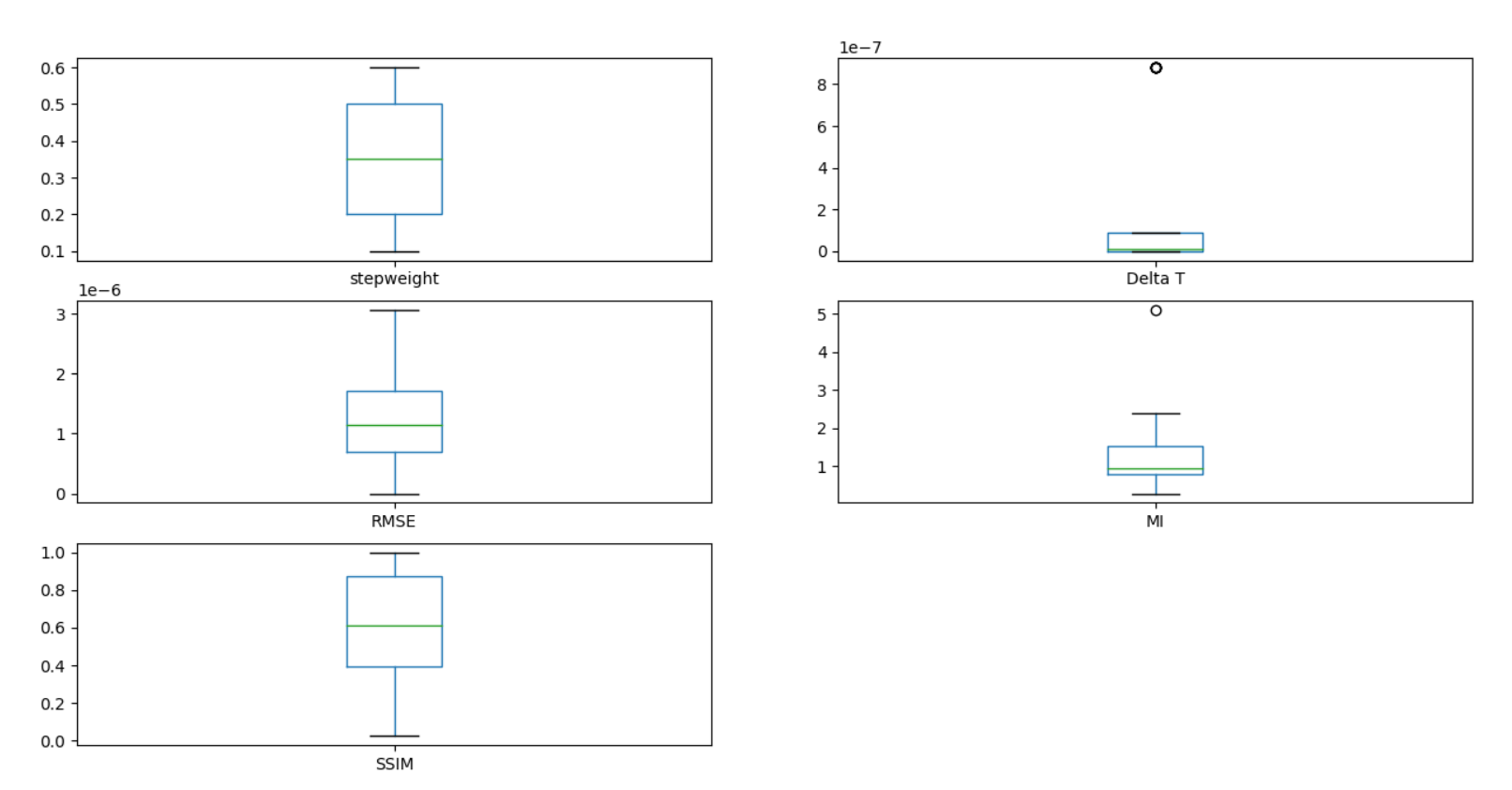


Figure: RRID30 S1 dataset box plot

Takeaways from the box plot:

* Our data is evenly distributed in step weight since there are 6 data points for each step weight.
* For delta T most of the data is distributed evenly around the range 4.84e-10 to 8.80e-08 with outlier data points at 8.80e-07.
* For RMSE the minimum value is 2e-9 while maximum value is 3.06e-6 with the median being 1.135e-06. The distribution is positively skewed.
* For MI, there is an outlier at around 5, which is the MI of a perfectly similar image. The rest are distributed around the range of 0.28 to 2.36 with the median being 0.95.
* For SSIM, the maximum value is 1 which is perfectly similar, and the minimum value is 0.015 with the median being 0.601.

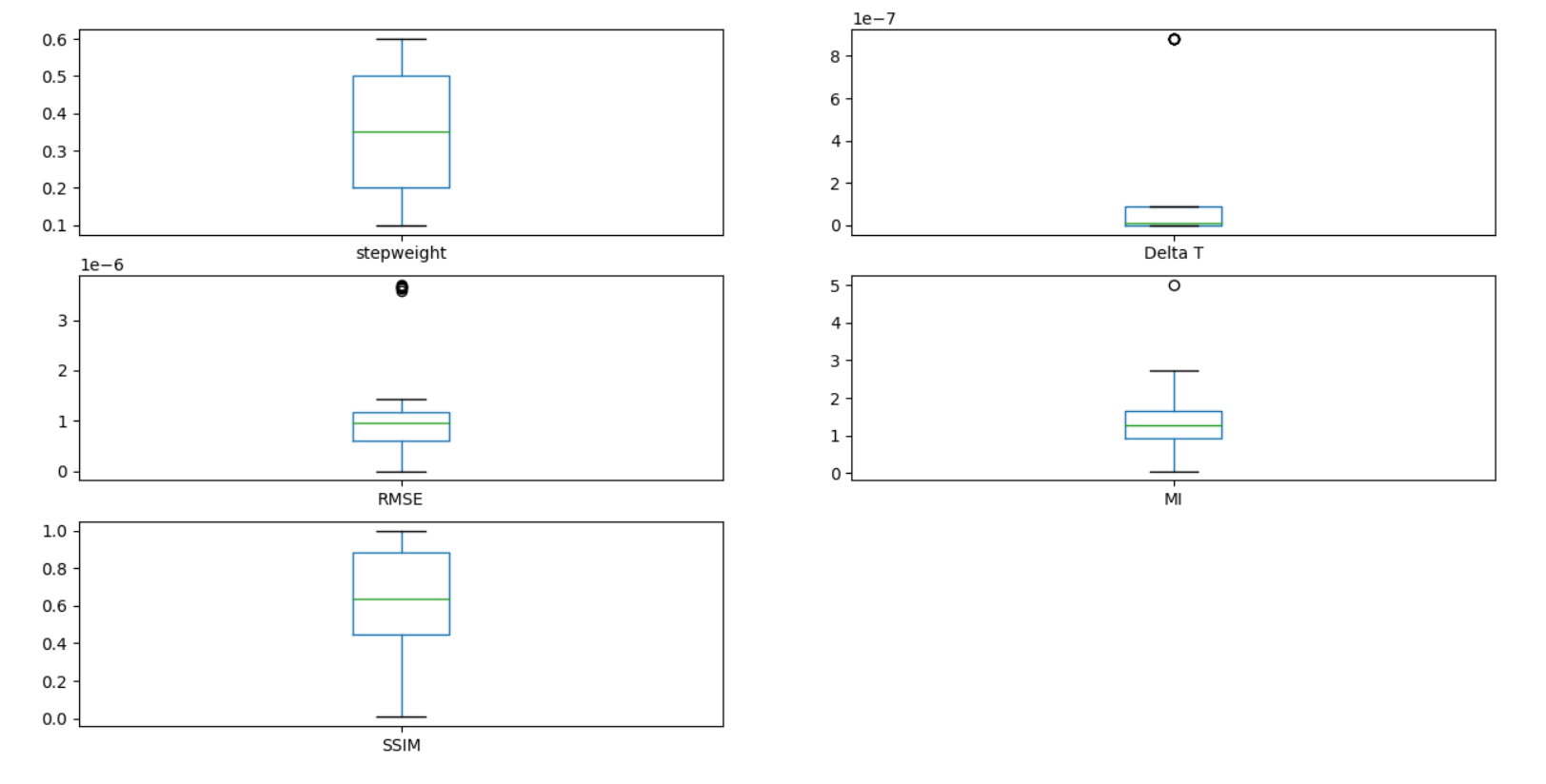


Figure: RRID30 S3 dataset box plot

Takeaways from the box plot:

* Notice RMSE, the range is a lot shorter than S1, -1e-09 to 1.43e-06 with a median of 9.7e -07. There are several outliers around 3. 6e-06
* The rest is same as RRID30 S1

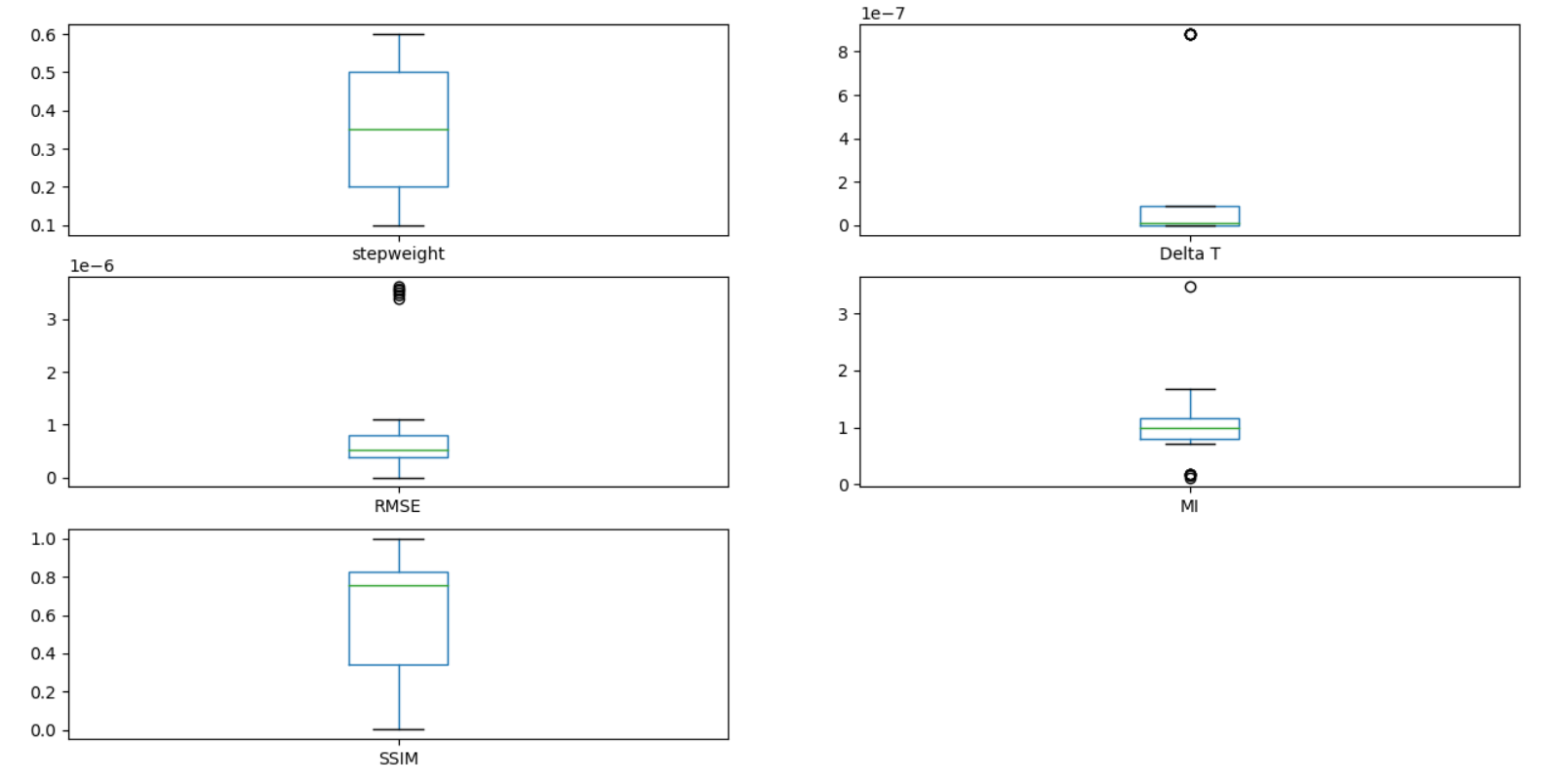


Figure: RRID30 S5 dataset box plot

Takeaways from the box plot:

* Notice RMSE, the range is a lot shorter than S1 and S3, -1e-08 to 1.12e-06 with a median of 5.3e -07. There are several outliers around 3. 6=5e-06
* Notice MI, the range is 0.71 to 1.70 which is much smaller than S1 and S3. The median is 0.97. There are also outliers at around 0.14.
* The rest is same as RRID30 S1

For RRID69 S1, S3, S5, it is almost the same as RRID30, and the RMSE range of data points also decreases from S1 to S5.

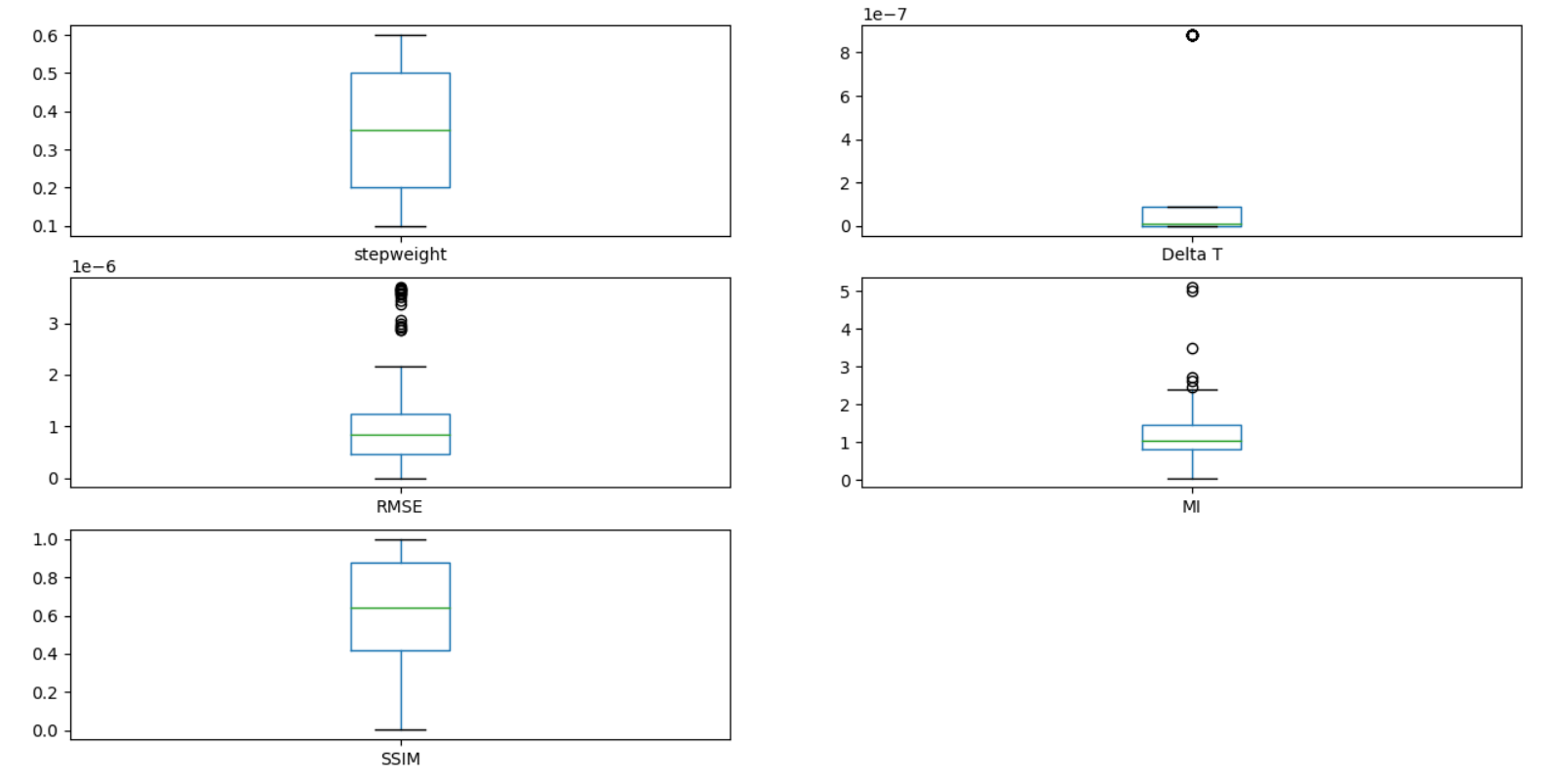


Figure : RRID30 compiled box plot

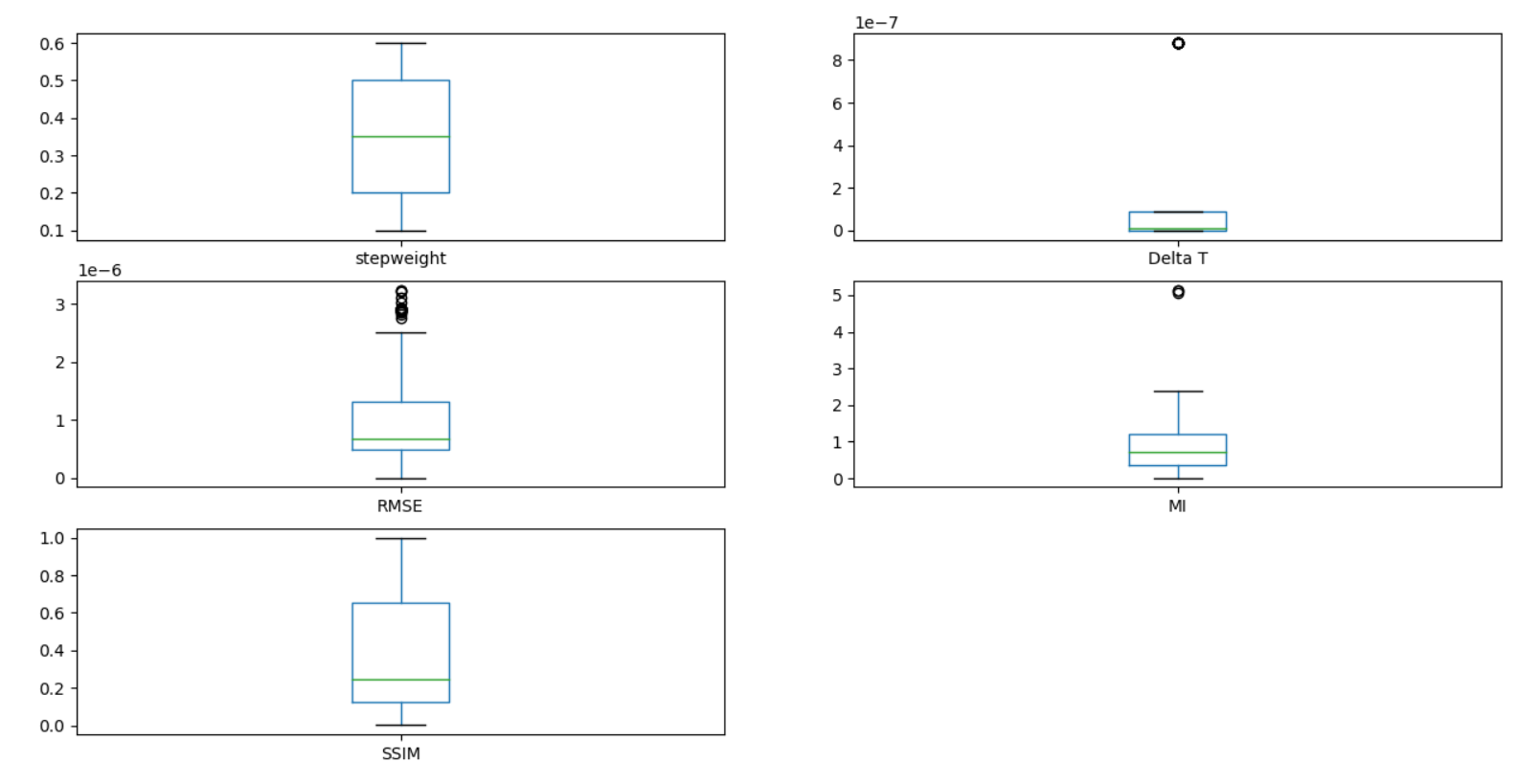


Figure : RRID69 compiled box plot

Take Away from the box plots:

* In RMSE have a lot of outlier above the max
* SSIM is evenly distributed, so there is no outliers in both
* MI for RRID30 is higher median than RRID69

## Histogram

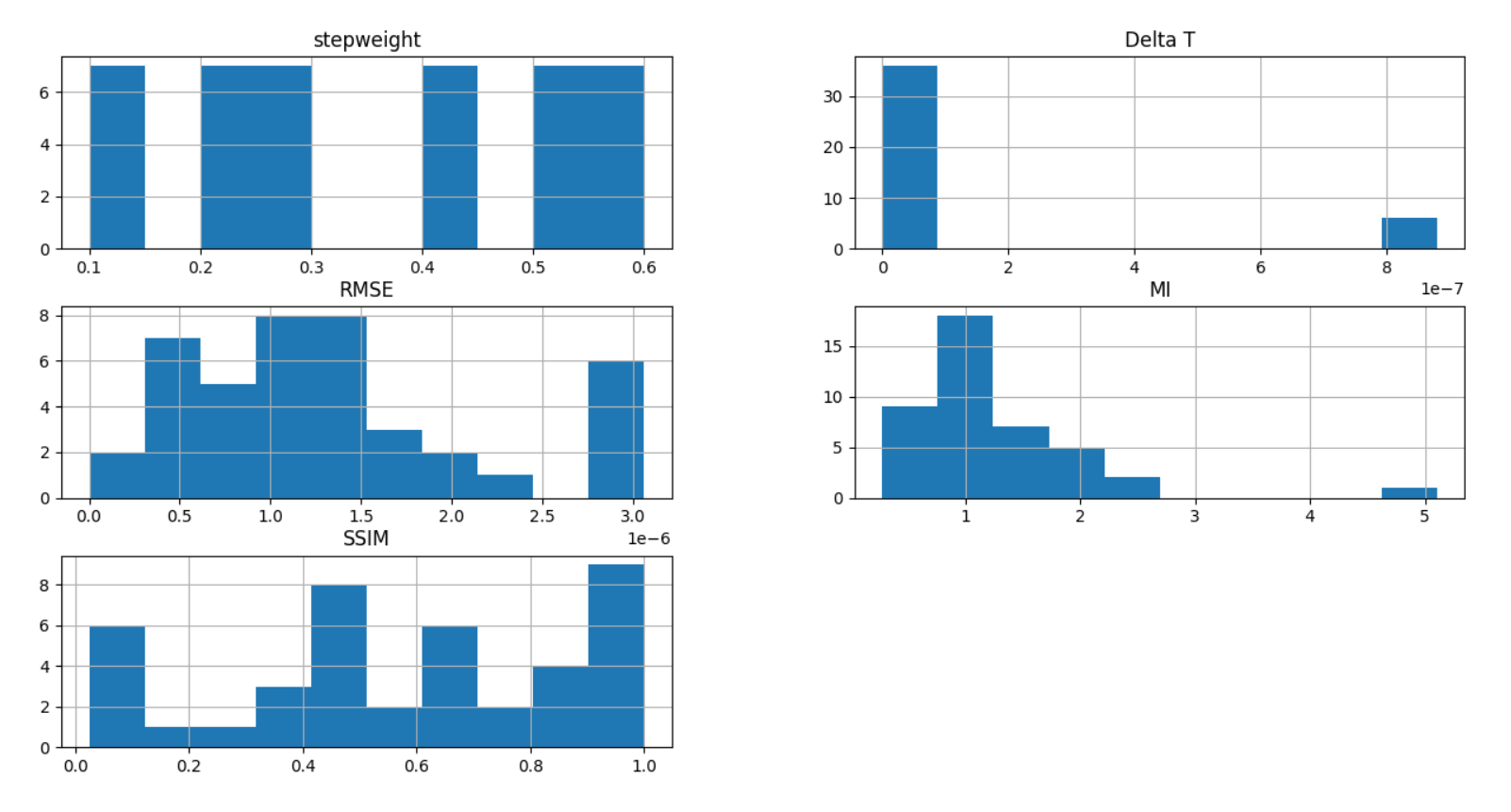


Figure: RRID30 S1 histogram

Takeaway:

* Stepweight evenly distributed
* For delta t most of the data is distributed in the range of 0 to 1e-07, while only some is distributed 8.80e-07
* MI looks like a Gaussian distribution

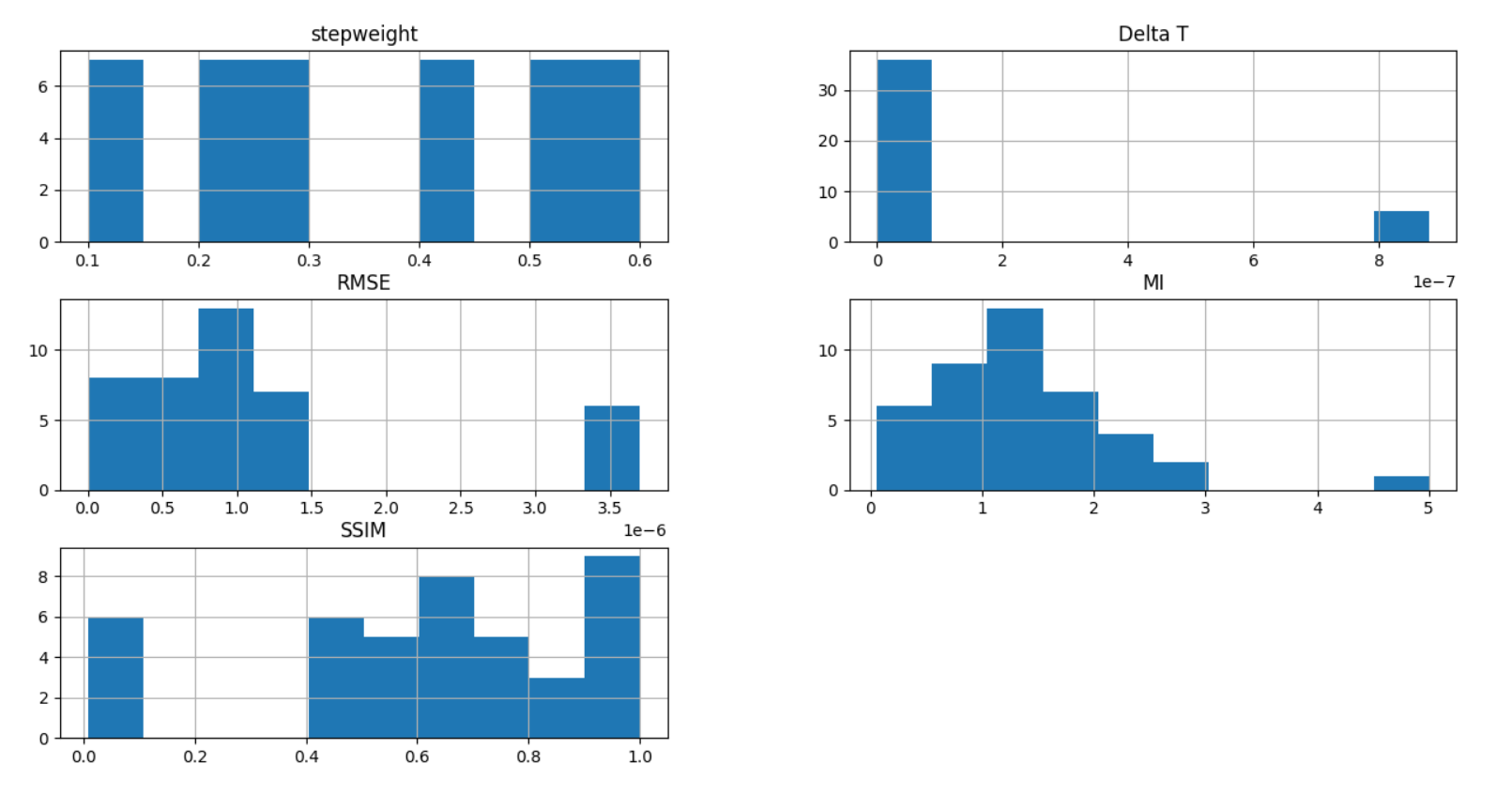


Figure: RRID30 S3 histogram

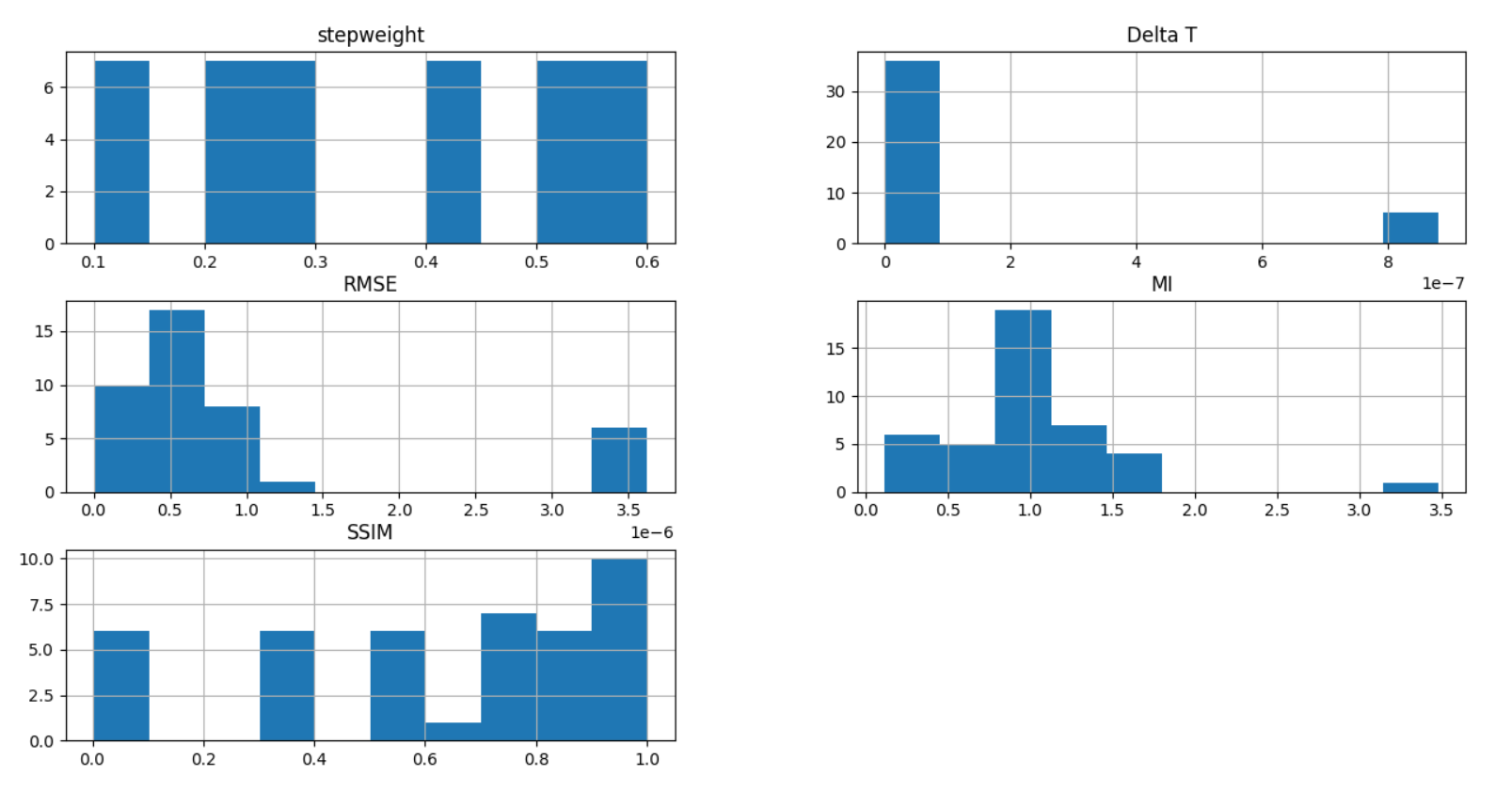


Figure: RRID30 S5 histogram

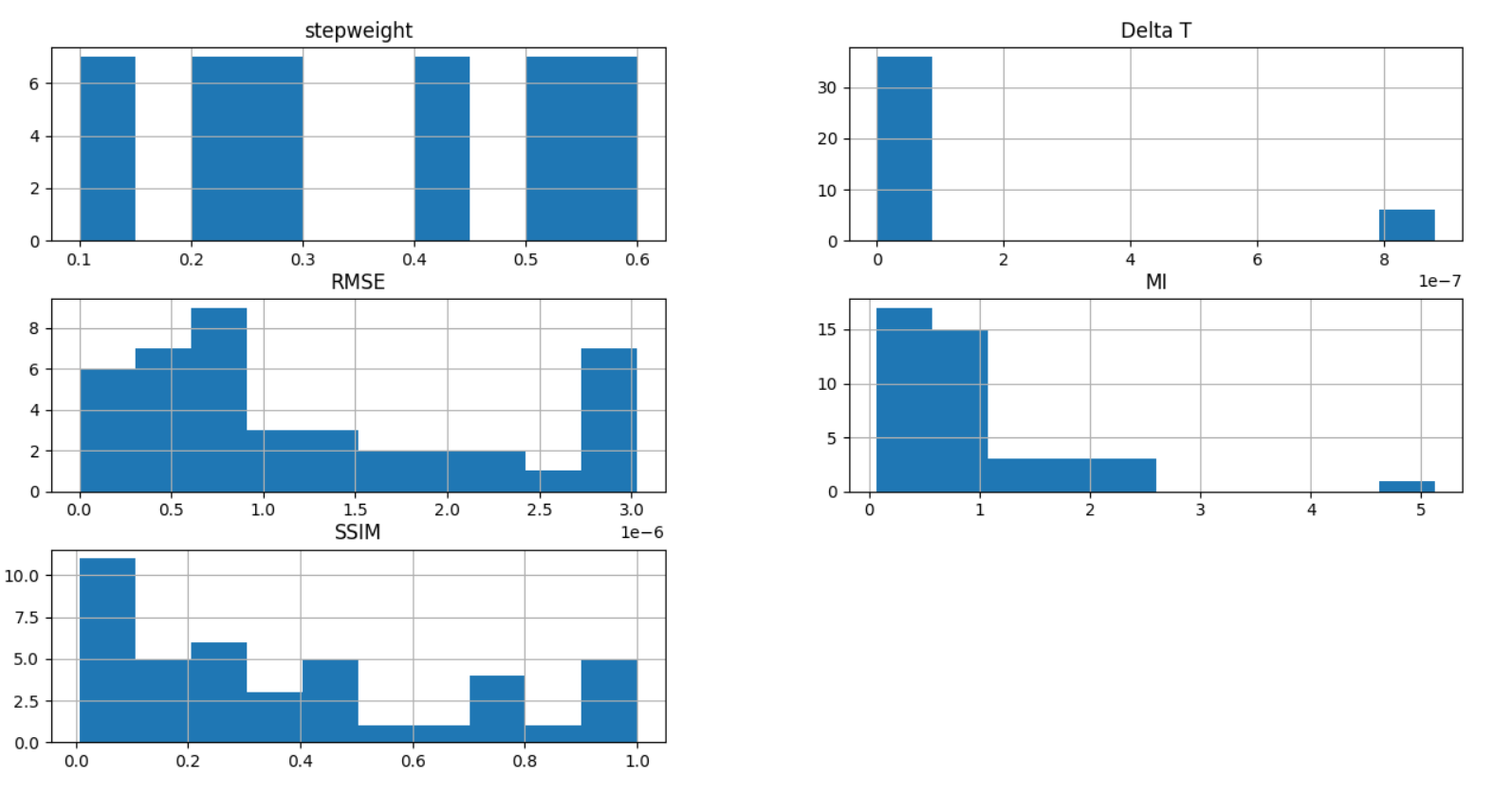


Figure: RRID69 S1 histogram

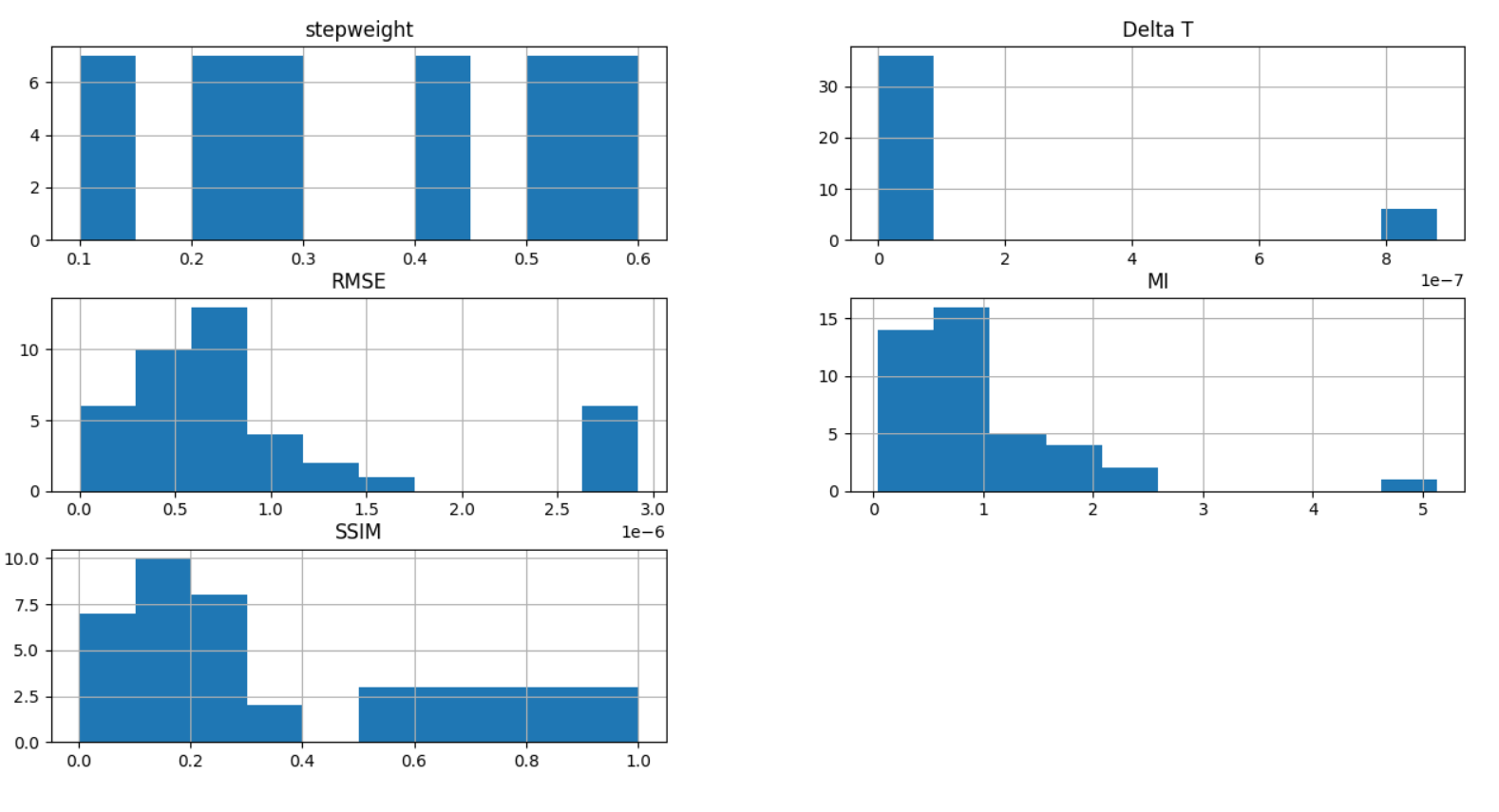


Figure: RRID69 S3 histogram

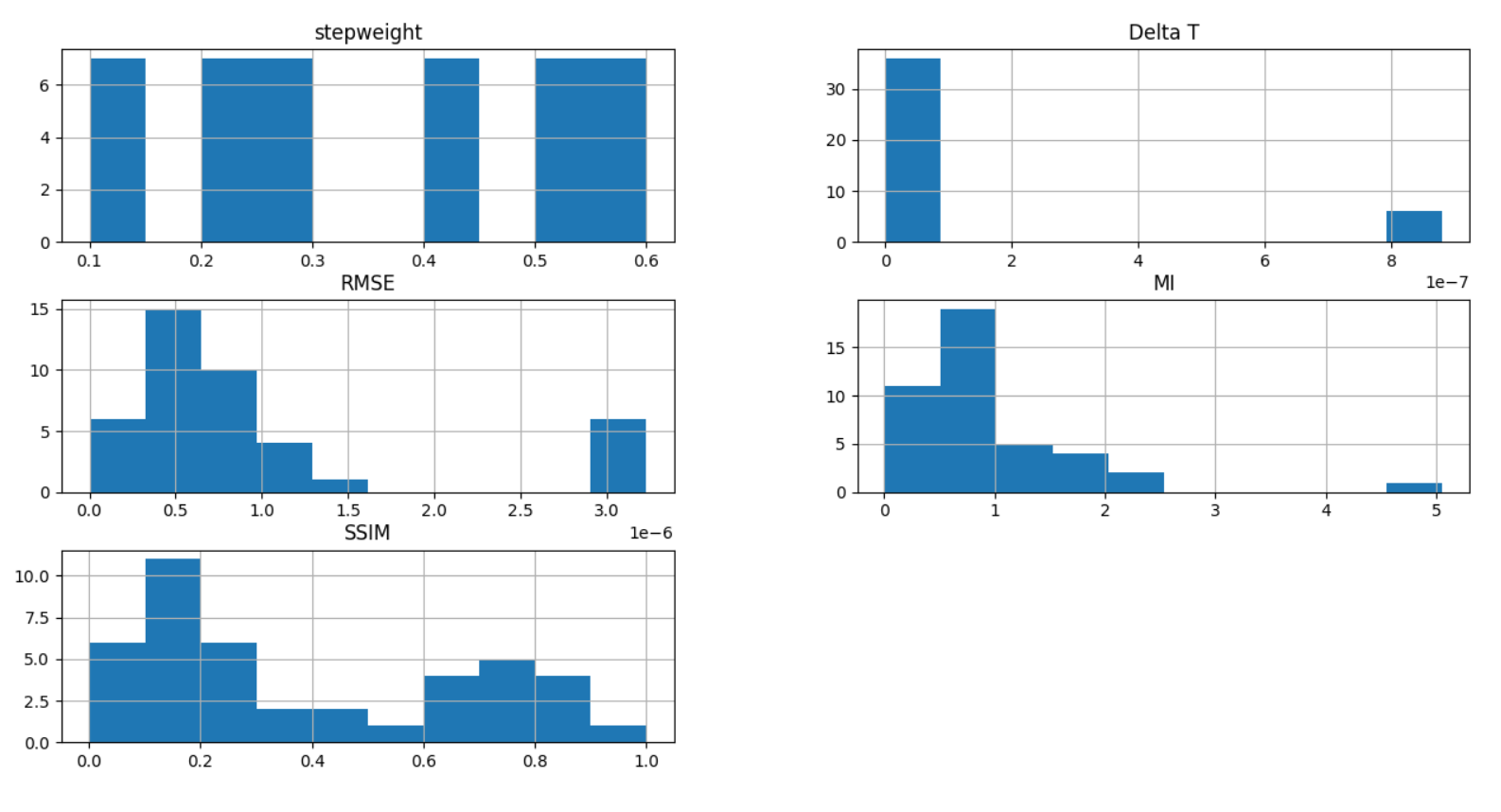


Figure: RRID69 S5 histogram

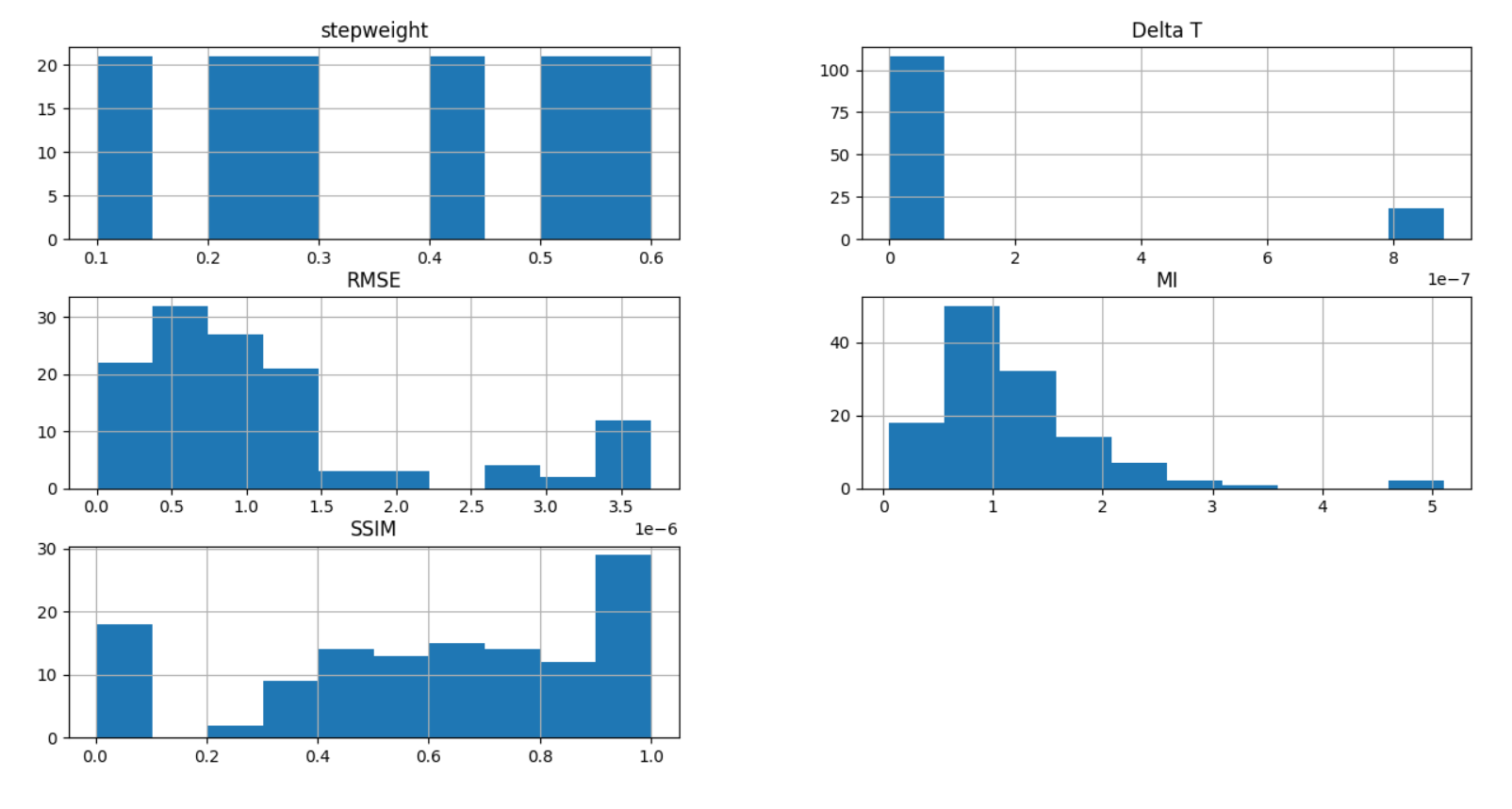


Figure: RRID30 compiled histogram

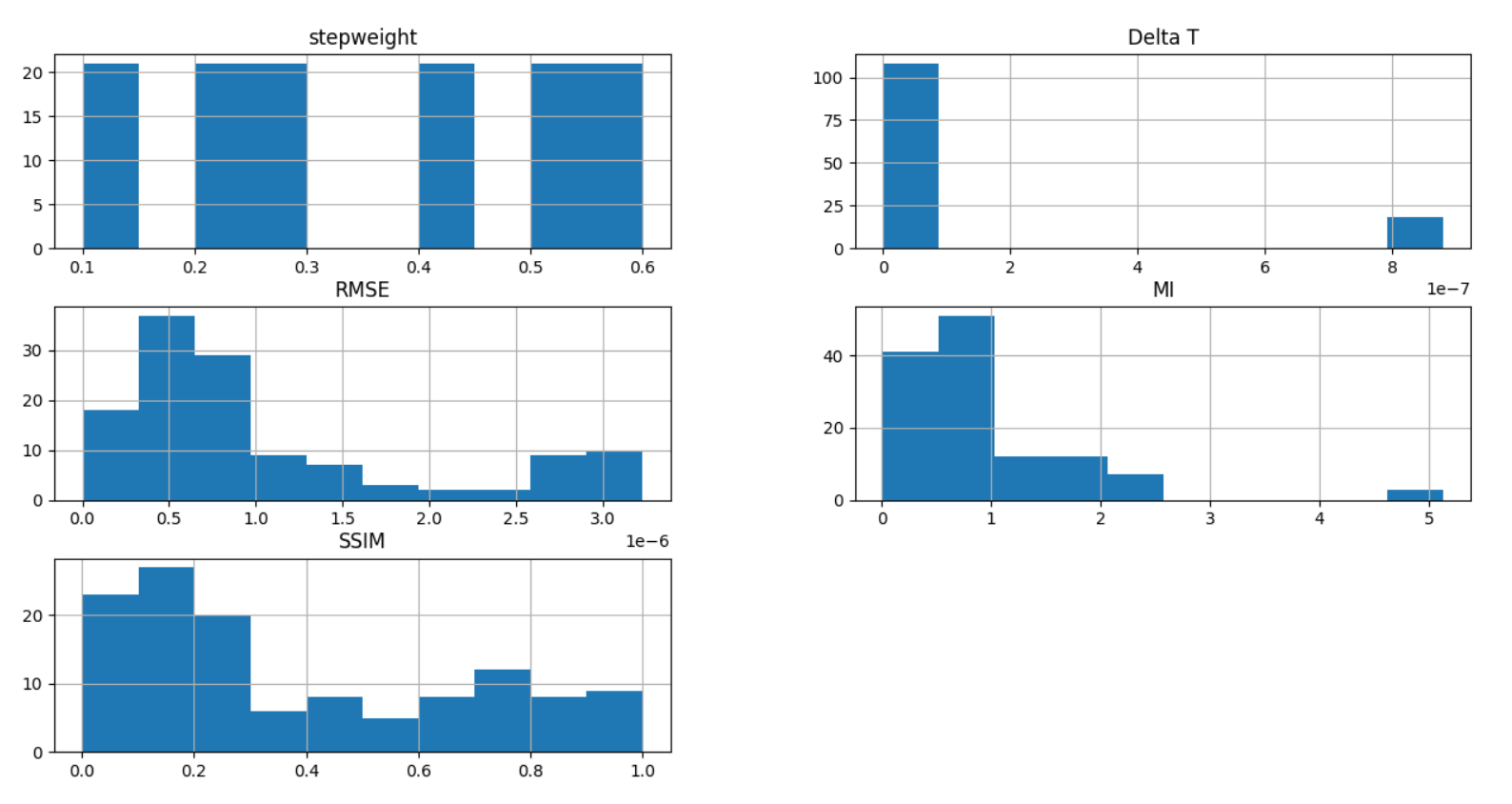


Figure: compile RRID69

## Scatter matrix

With a scatter matrix, we can see the relation between different features of the data points. Note the diagonal grouping of some pairs of attributes. This suggests a high correlation and a predictable relationship. From below we can see that RMSE, MI and SSIM have high correlation between each other for every dataset.

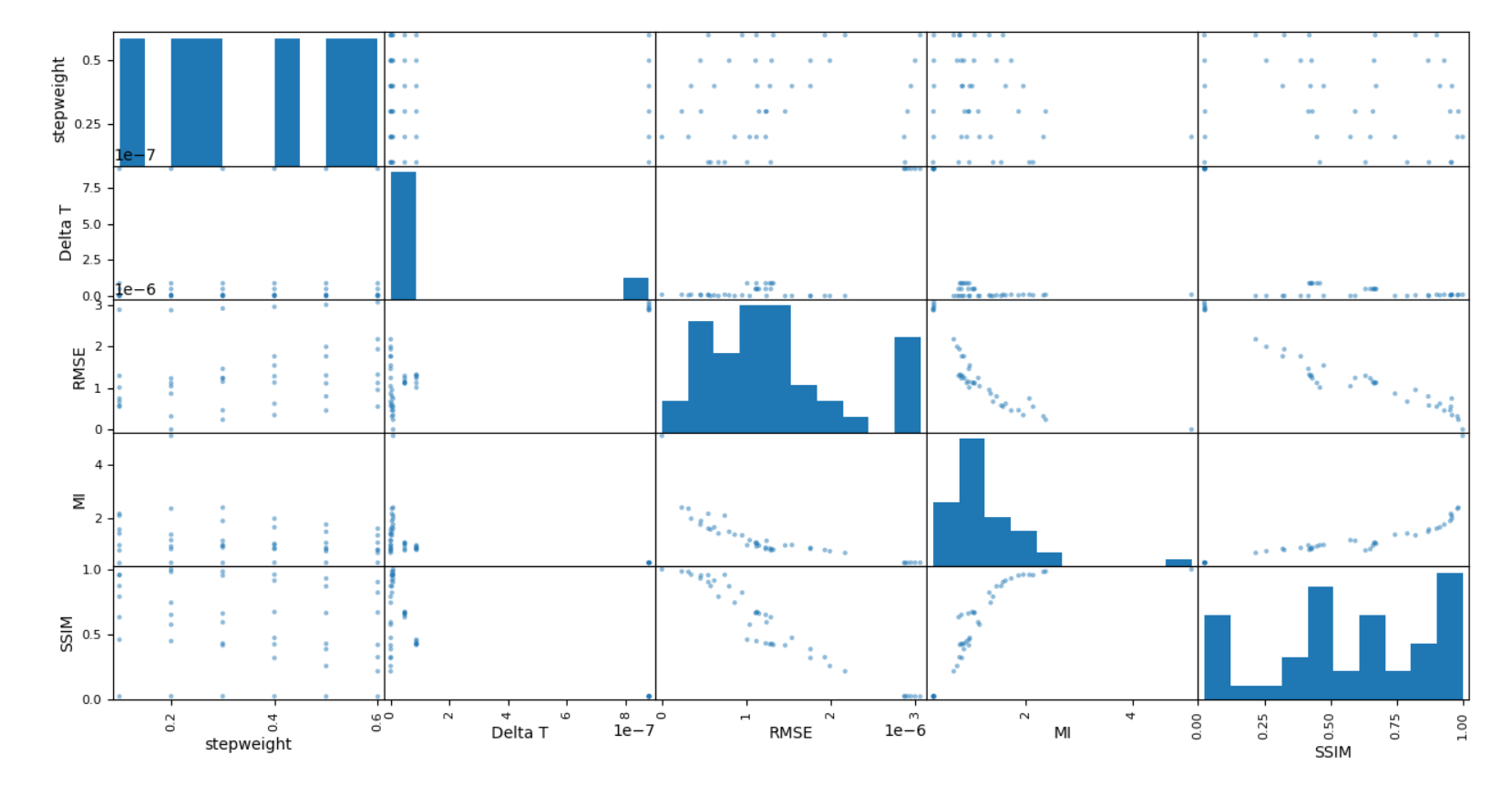


Figure: RRID30 S1 scatter matrix

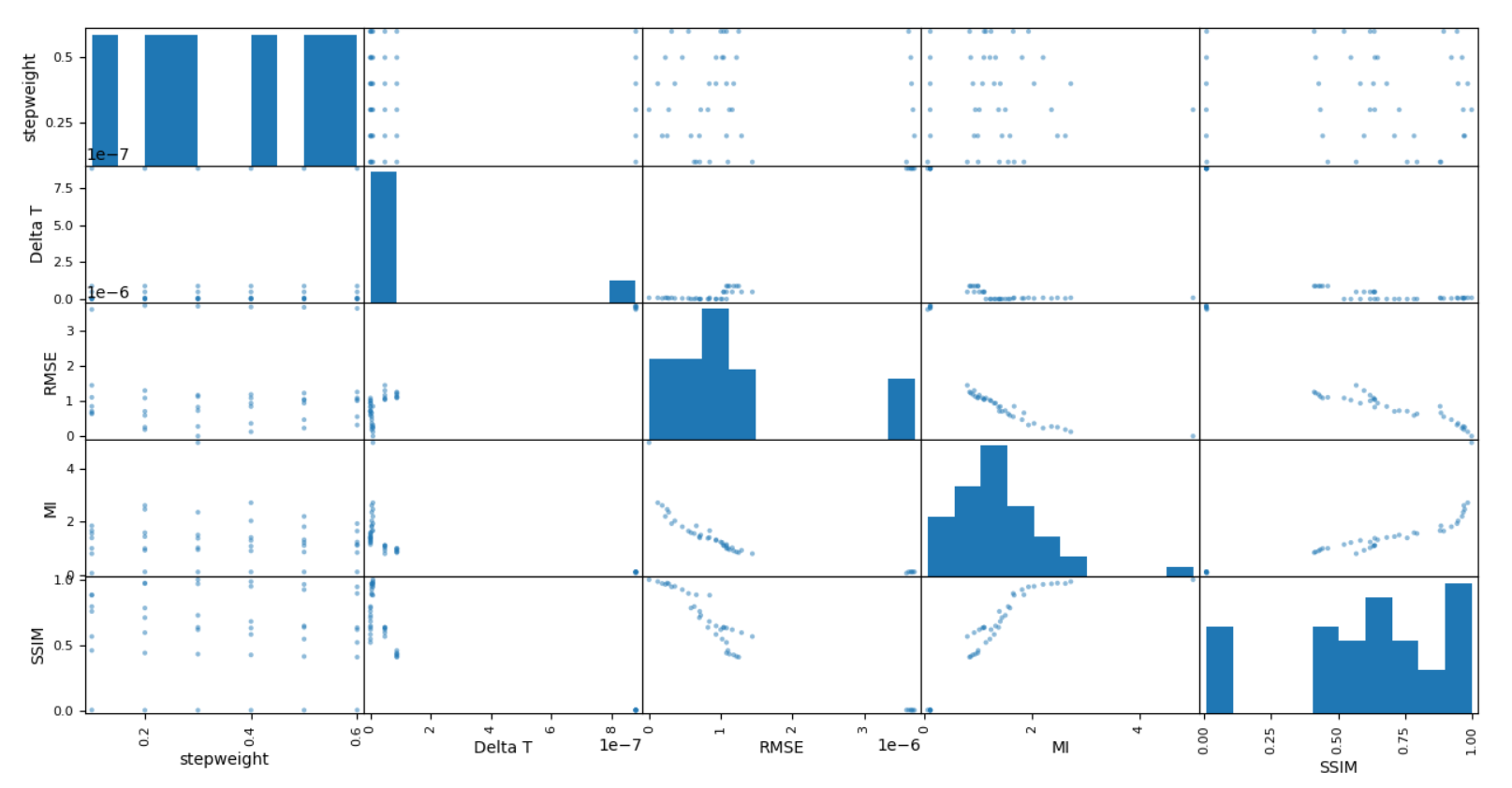


Figure: RRID30 S3 scatter matrix

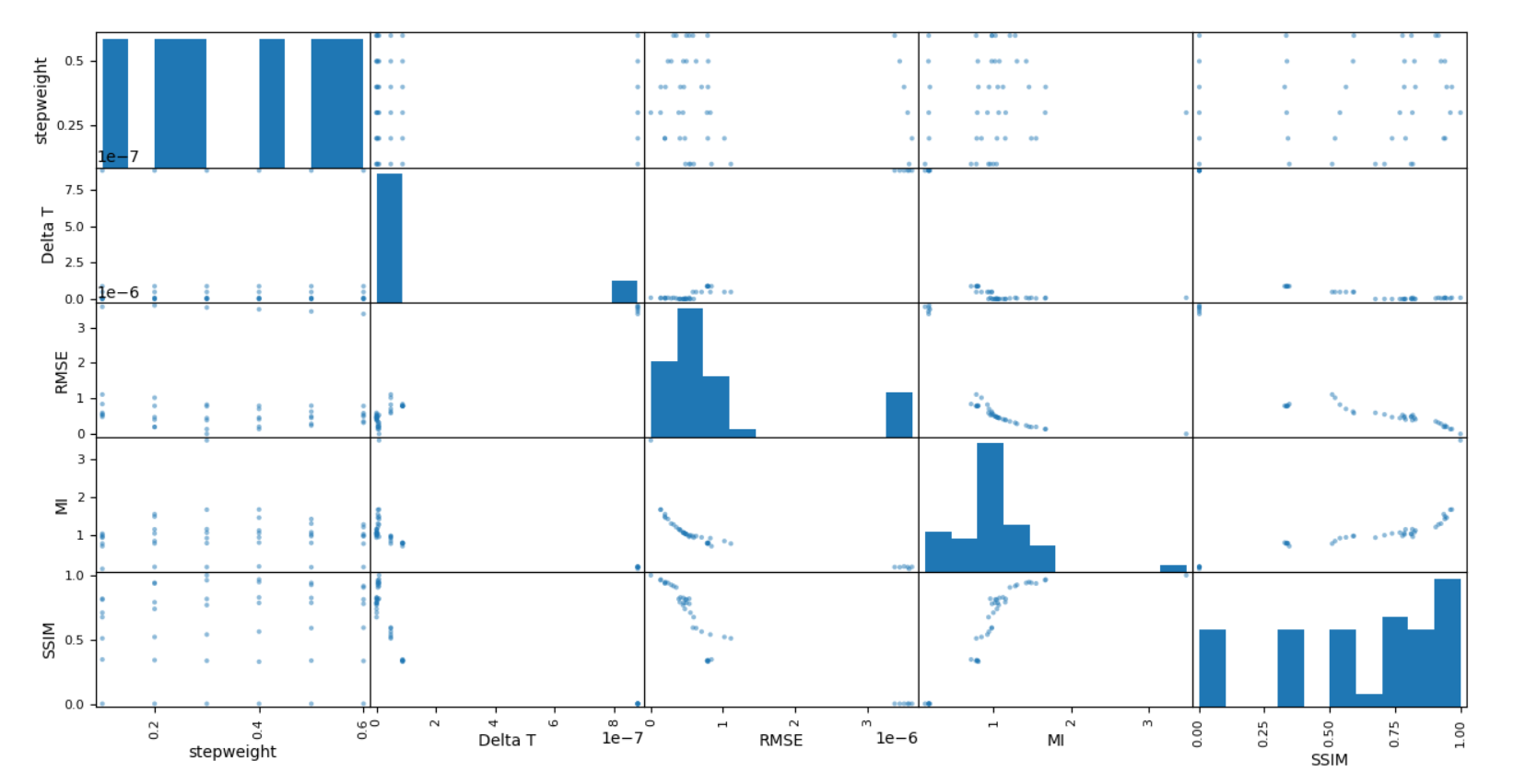


Figure : RRID30 S5 scatter matrix

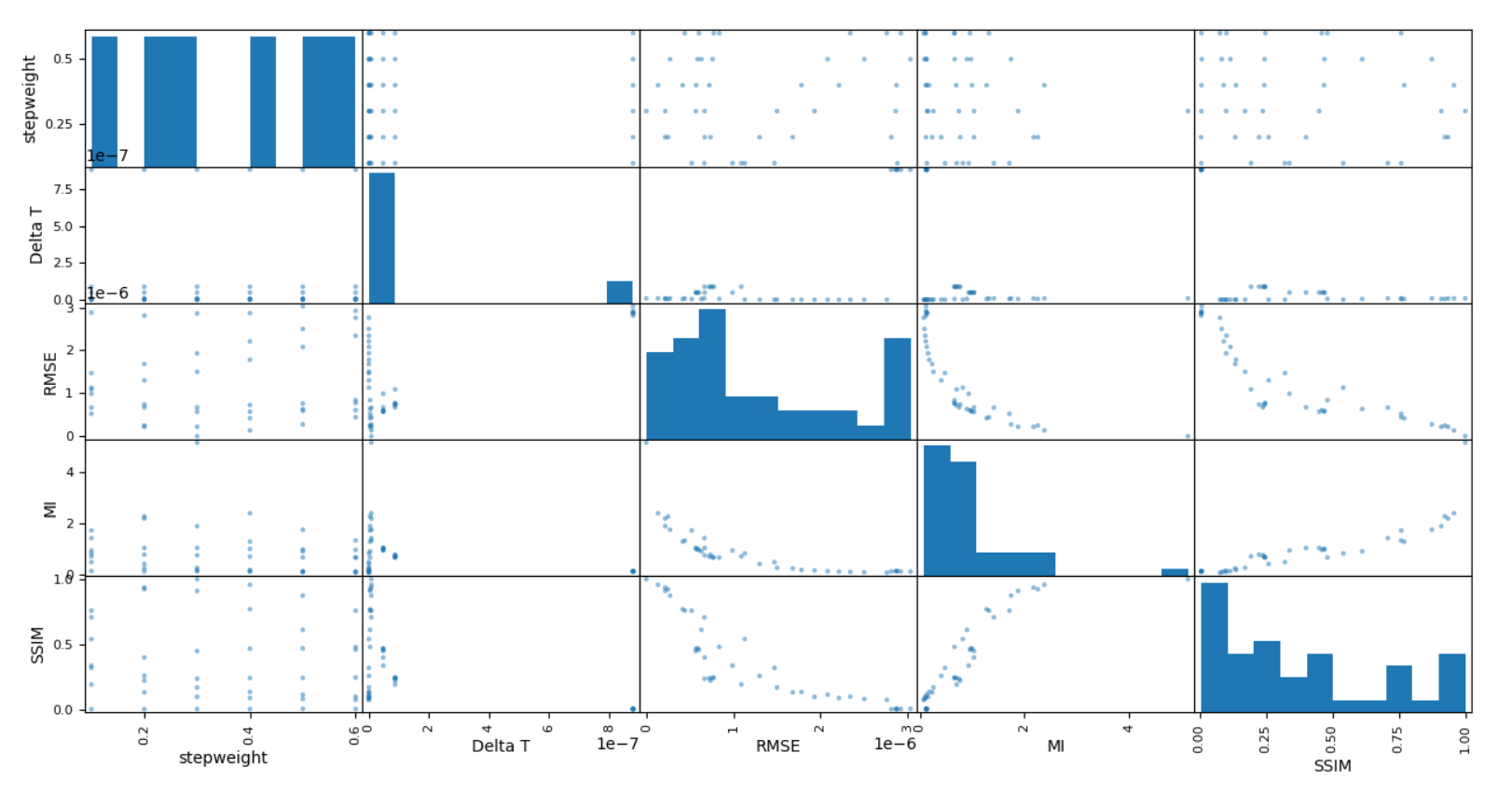


Figure : RRID69 S1 scatter matrix

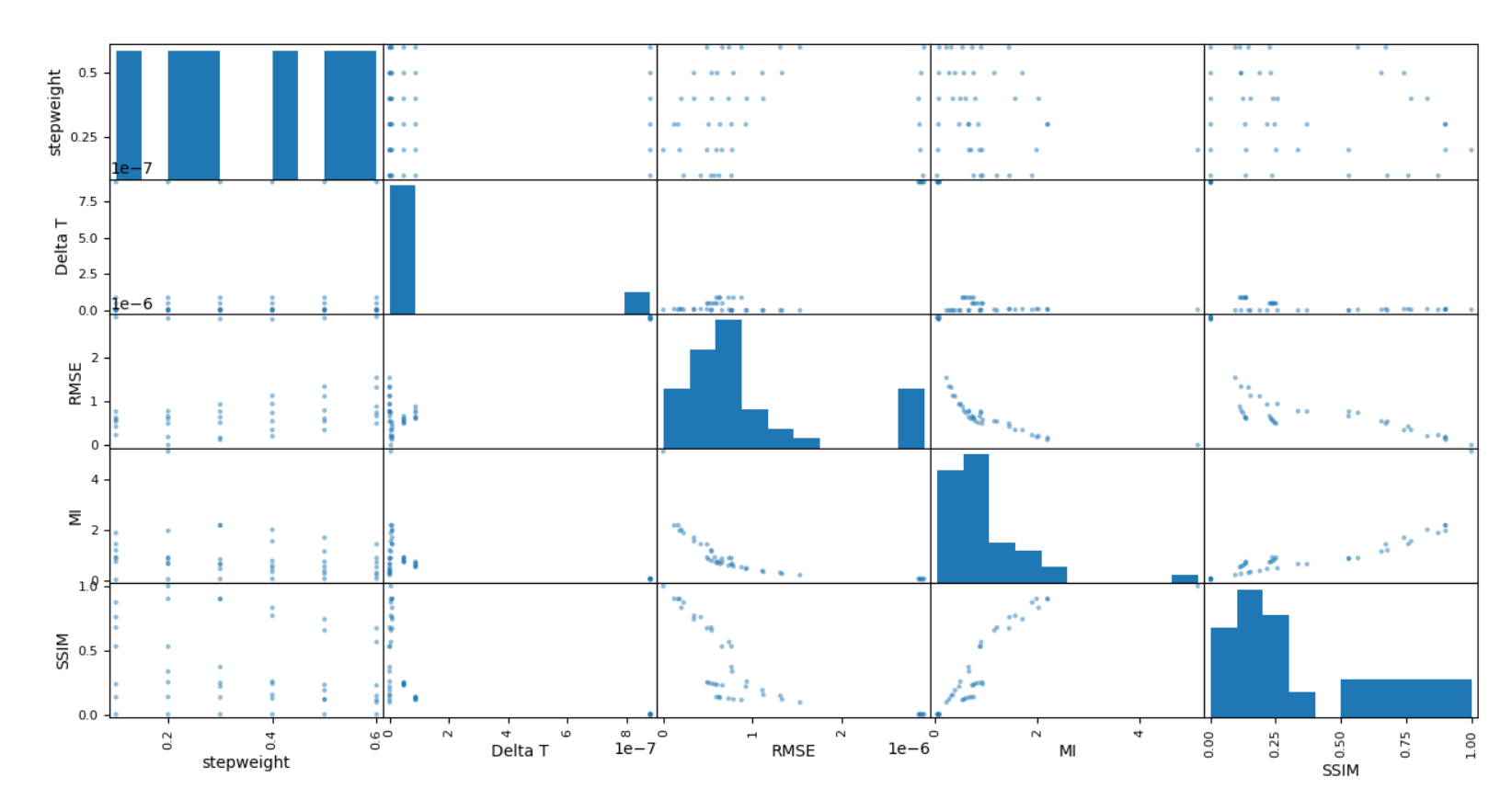


Figure: RRID69 S3 scatter matrix

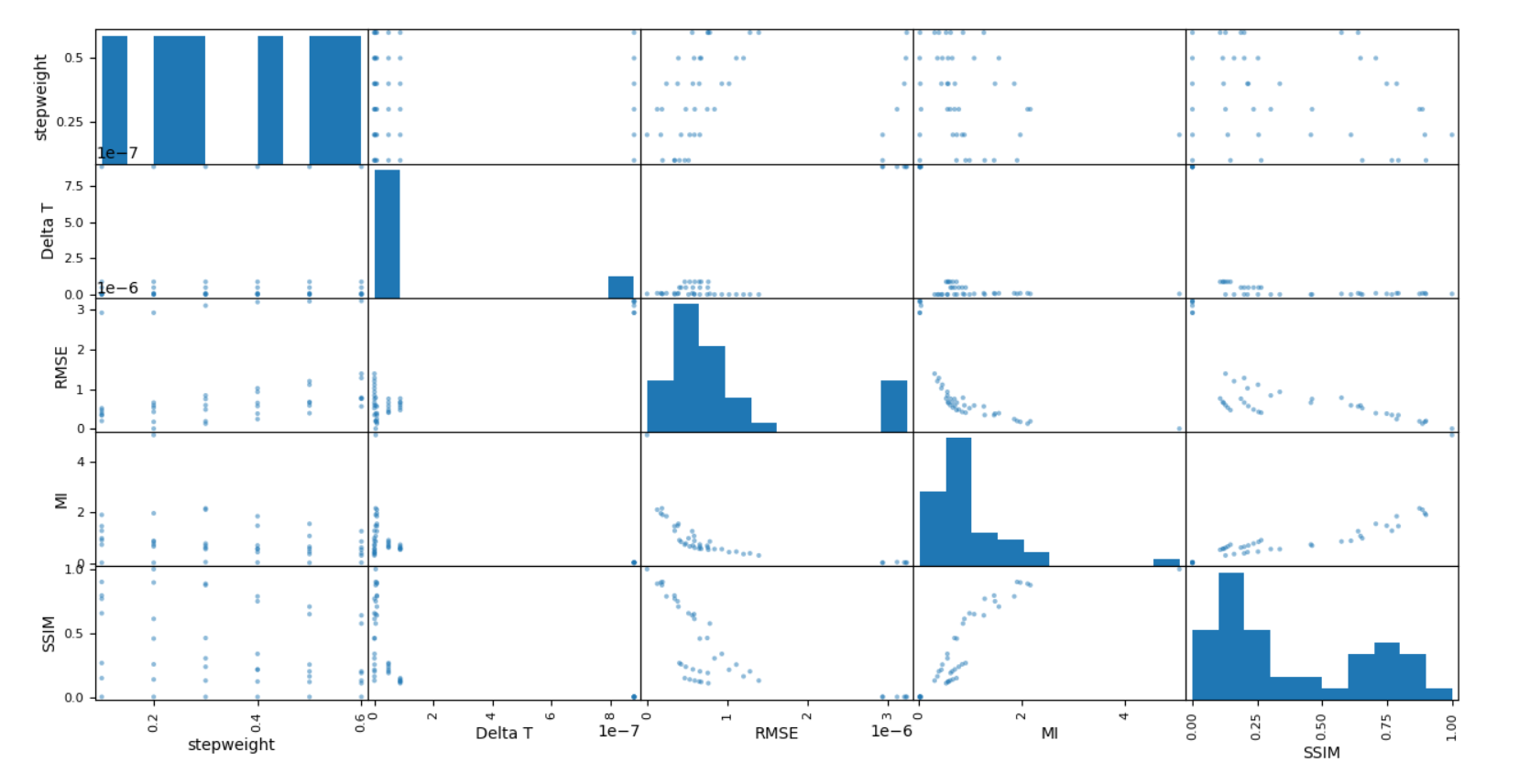


Figure: RRID69 S5 scatter matrix

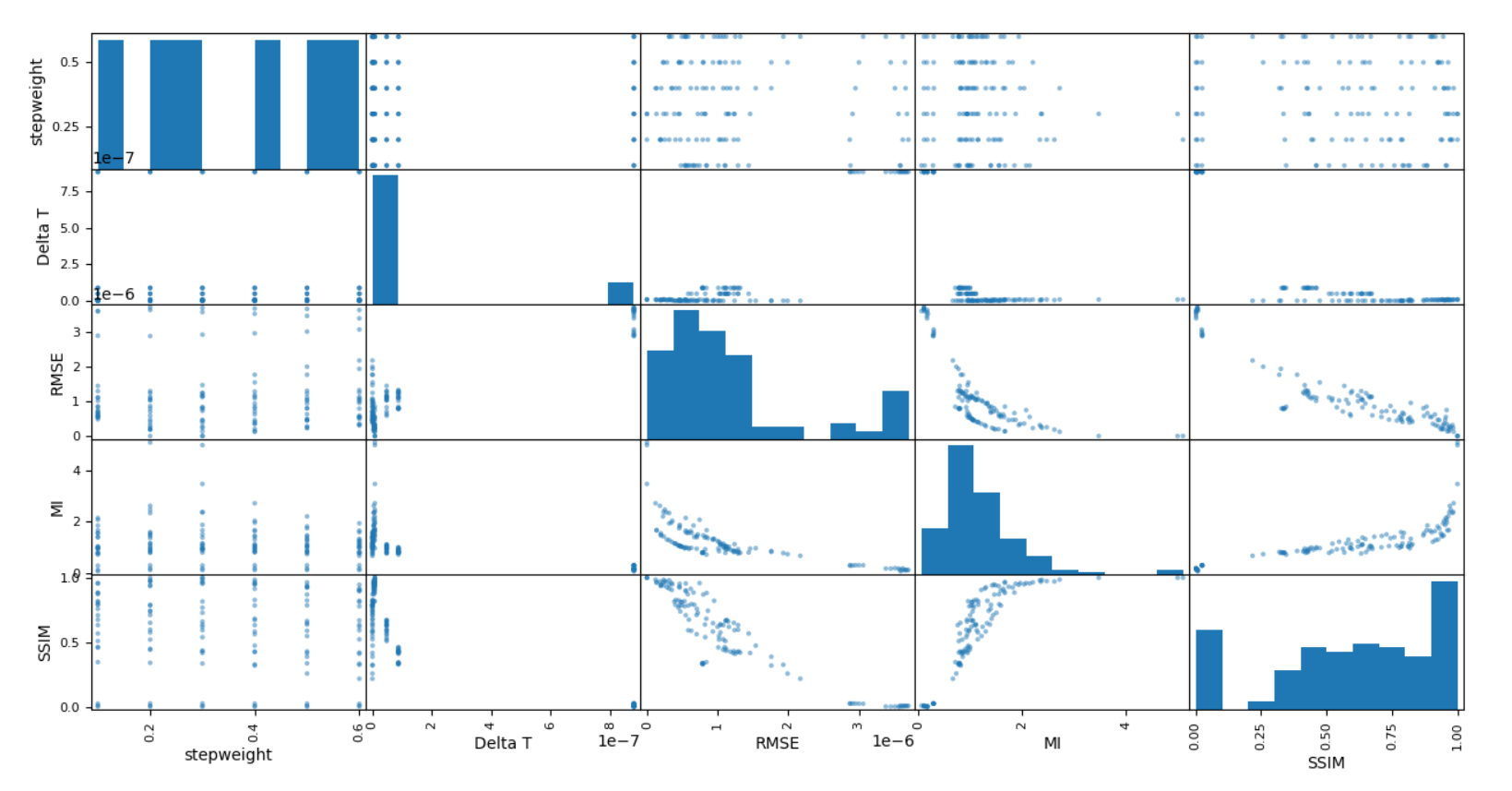


Figure: RRID30 compiled scatter matrix

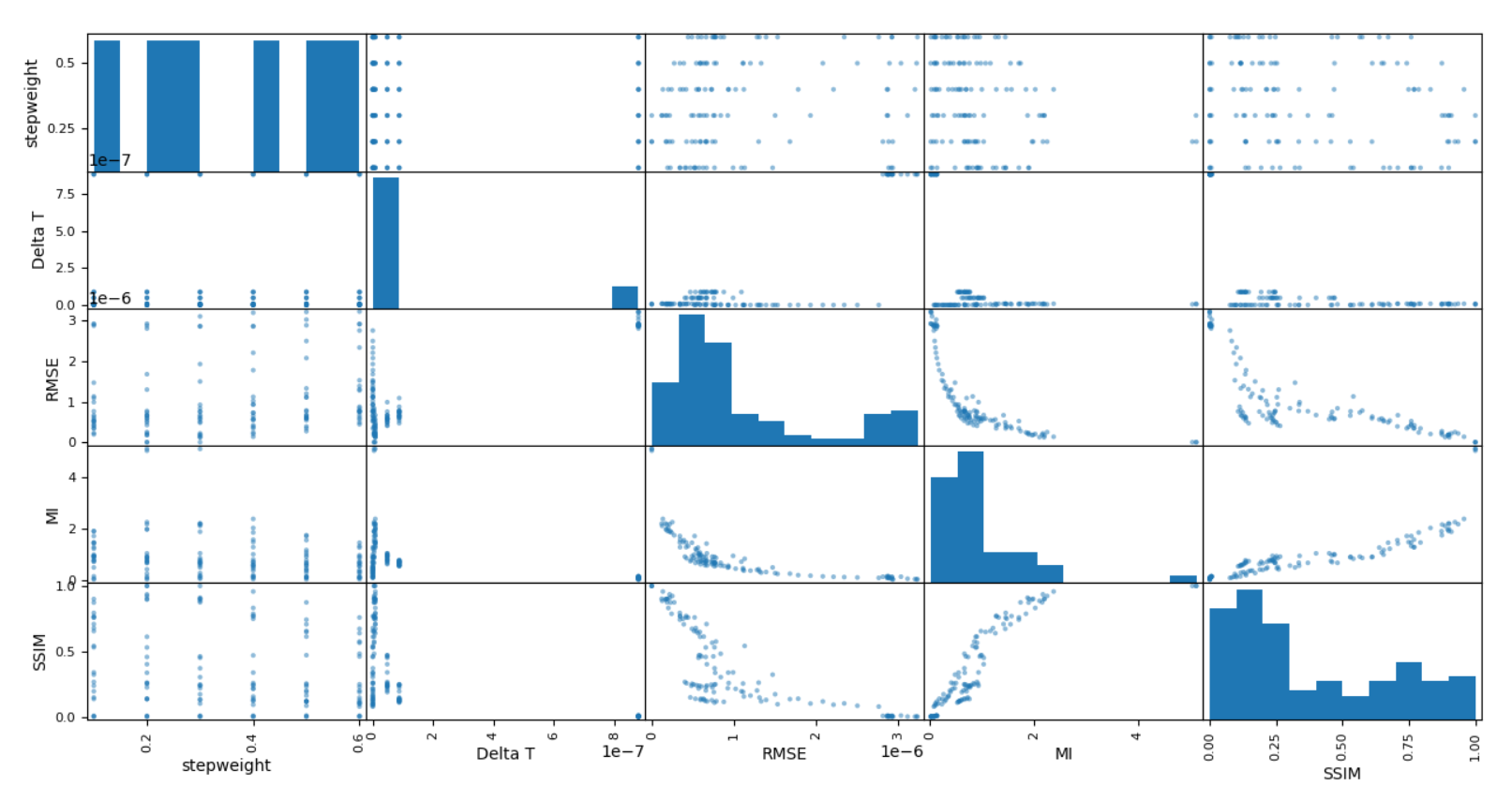


Figure: RRID69 compiled scatter matrix

# Machine learning algorithms

## *M*ultinomial Naive Baye*s*

A machine learning model that is frequently used in multiclass classification is multinomial naive bayes. This method treats all features equal and independent of each other, which means the order or priority of the features is not taken into consideration. In real life, some of the features do have more significance than others, and features usually have some type of relationships between them, that's why this assumption that everything is equal is quite “naive”. This model is able to handle very high dimensionality feature space with thousands of features, which makes it suitable to be used to categorise textual information such as filtering spam emails. Naive Bayes is computationally very efficient and very easy to apply to solve multi classification problems. Since naive bayes are mostly used for text classification, I will use a simple text classification example to ease explanation of the model.

For this example, we are going to separate emails into spam and normal emails. We received 8 normal emails and 4 spam emails. 4 words in each email will be considered as features in the model, which is “dear”, “friend”, “lunch” and “money”. Below shows the frequency of words +that occur in the normal emails.

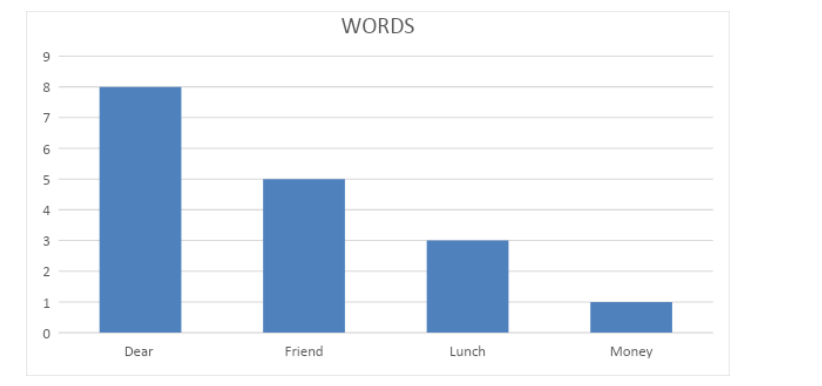


Figure: Histogram of the words that occur in normal emails

The probability of each word to occur in a normal email:

* **Probability (Dear|Normal) = 8 /17 = 0.47**
* **Probability (Friend|Normal) = 5/ 17 =0.29**
* **Probability (Lunch|Normal) = 3/ 17 =0.18**
* **Probability (Money|Normal) = 1/ 17 =0.06**

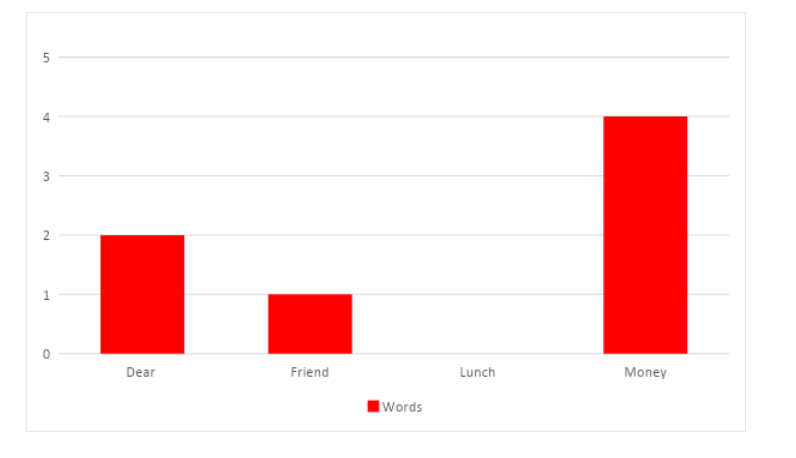
****

Figure : Histogram of words that occur in spam emails

The probability of each word occurring in a spam email:

* **Probability (Dear|Spam) = 2 /7 = 0.29**
* **Probability (Friend|Spam) = 1/ 7 =0.14**
* **Probability (Lunch|Spam) = 0/ 7 =0.00**
* **Probability (Money|Spam) = 4/ 7 =0.57**

We have calculated the probabilities of discrete words occurring in both types of emails, so these probabilities can be called likelihoods.

Now we receive a normal email with “ Dear Friend” in it, so we start our initial guess that the message is a normal email. The probability of the email being normal is:

* **Probability (Normal) = 8 / (8+4) = 0.67**

We then multiply this with the likelihood of occurring “Dear” and “Friend” which is 0.47 and 0.29 respectively.

* **Probability of having “ Dear Friend” in a normal email = 0.67 \* 0.47 \* 0.29 = 0.09**

Now we repeat the same thing if “Dear Friend” happened in a spam email.

* **Probability (Spam) = 4 / (8+4) = 0.33**
* **Probability of having “ Dear Friend” in a normal email = 0.33 \* 0.29 \* 0.14 = 0.01**

Therefore based on the probabilities, we are able to guess that the email belongs to a normal email because the probability is higher.

I chose to use this model because the features I placed are mostly independent of each other. Thus, it may be a good model to predict the condition of the image.

## Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a reduction method for dimensionality. As the name suggests, the number of dimensions (i.e. variables/ features) in a dataset is minimised by dimensionality reduction methods thus preserving as much details as possible. For example, imagine that we have plotted the relationship between two variables in which a particular class is represented by each colour.

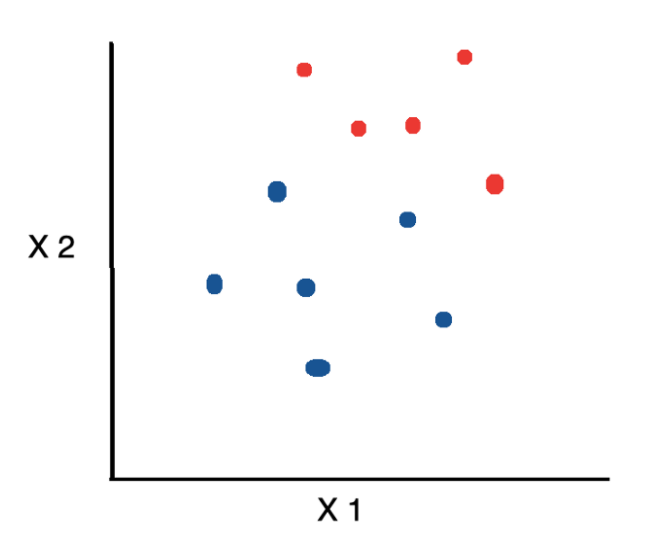


Figure: datasets with 2 manipulating variables

we’d like to reduce the number of dimensions down to 1. We could reduce 2 dimensions into 1 using the method below.

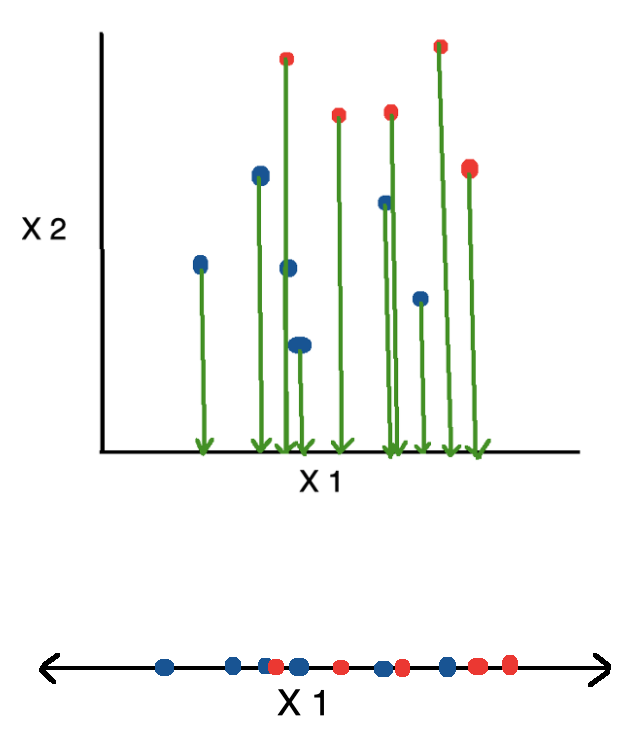


Figure: One method of reducing dimensionality

However if we use this method, the information provided by variable/ feature 2 is all lost. Linear Discriminant Analysis, or LDA, uses the data from both features to construct a new axis and projects the information to the new axis in such a manner that the variation is reduced and the distance between the means of the two groups is maximised as shown in the figures below ( Maklin, 2019).

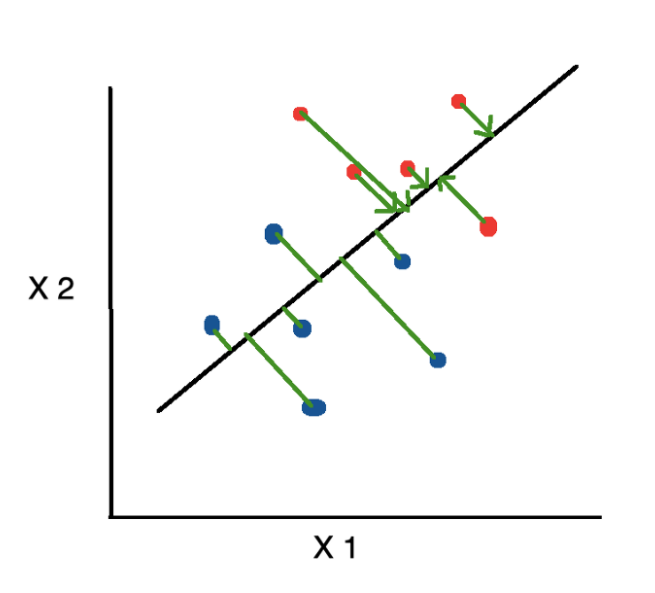


Figure: Concept of Linear Discriminant Analysis Pt.1

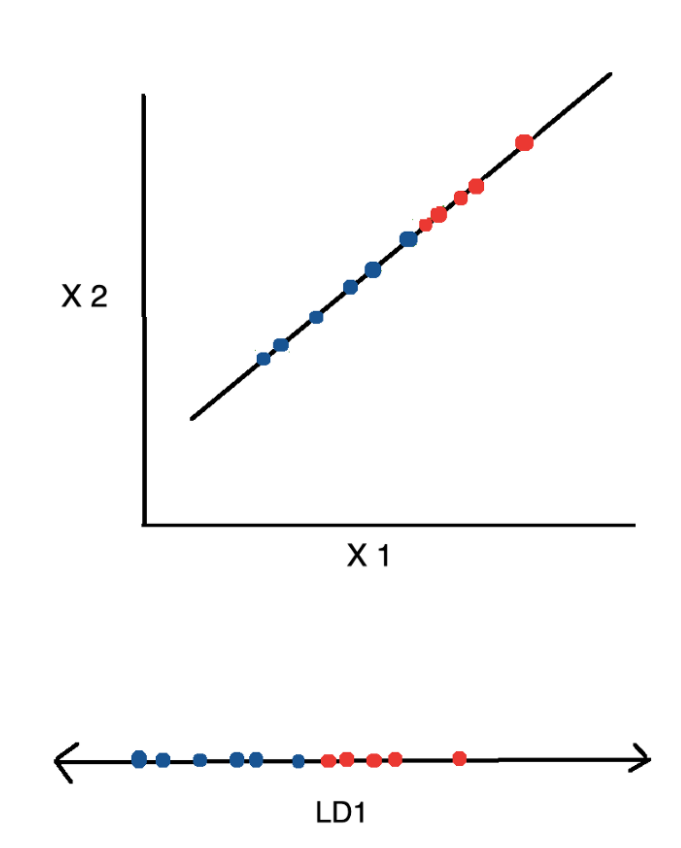


Figure: Concept of Linear Discriminant Analysis Pt.2

## Decision Tree Classifier

A decision tree is a supervised machine learning algorithm where data is continuously split according to a certain parameter, just like how a tree split from the stem, into branches, into leaves (Chakure, 2019). Decision tree consists of:

1. **Nodes:** Test for the value of a certain feature.
2. **Edges/Branch:** Results from the outcome of a test and connect to the the next node or leaf
3. **Leaf nodes:** Terminal nodes that predict the outcome (at the end of the decision tree)

It is very reminiscent of nested if-else statements in programming, and in this case it splits data based on the features of the data.

Classification trees

Yes and no decisions are being made going down the tree. The decision or prediction outcome is discrete or categorical.

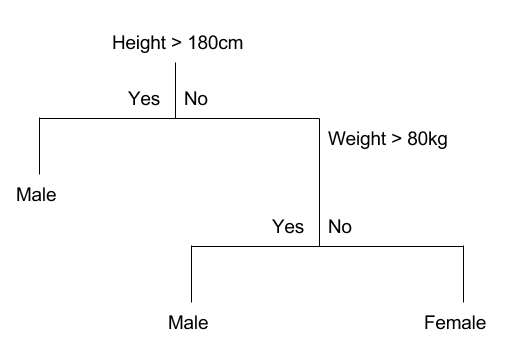


Figure: Example of classification trees

Regression trees

Decision trees where the outcome can take continuous values such as stock prices, height of a child etc.

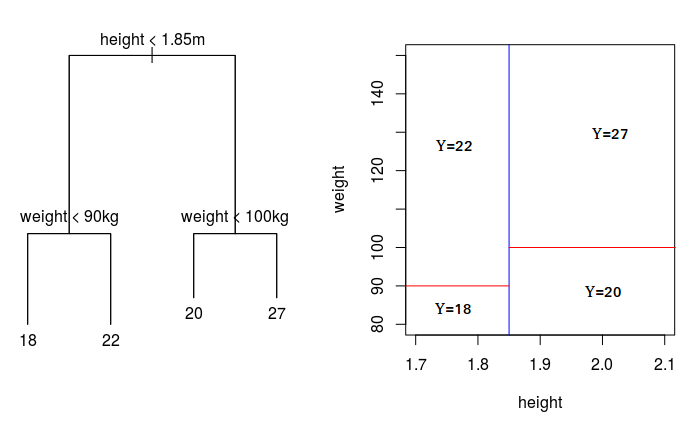


Figure: Example of regression tree

Basically the decision tree predicts following the 3 steps below:

1. We start at the tree root using the decision algorithm and segment the data on the function that results in the largest information gain (IG) (reduction in uncertainty towards the final decision)
2. In an iterative process, we can then repeat this splitting procedure at each child node until the leaves are pure. This means that the samples at each leaf node all belong to the same class.
3. In implementation, to avoid overfitting, we can put a limit on the depth of the tree. Here we compromise slightly on purity, although the final leaves could still have any impurity.

The advantages of decision tree classifiers:

* Inexpensive to construct.
* Extremely fast at classifying unknown records.
* Easy to interpret for small-sized trees
* Accuracy comparable to other classification techniques for many simple data sets.
* Excludes unimportant features

The disadvantages of decision tree classifiers:

* Small changes in the training data can result in large changes to decision logic
* Easy to overfit.
* Large trees can be difficult to interpret and the decisions they make may seem counter intuitive.

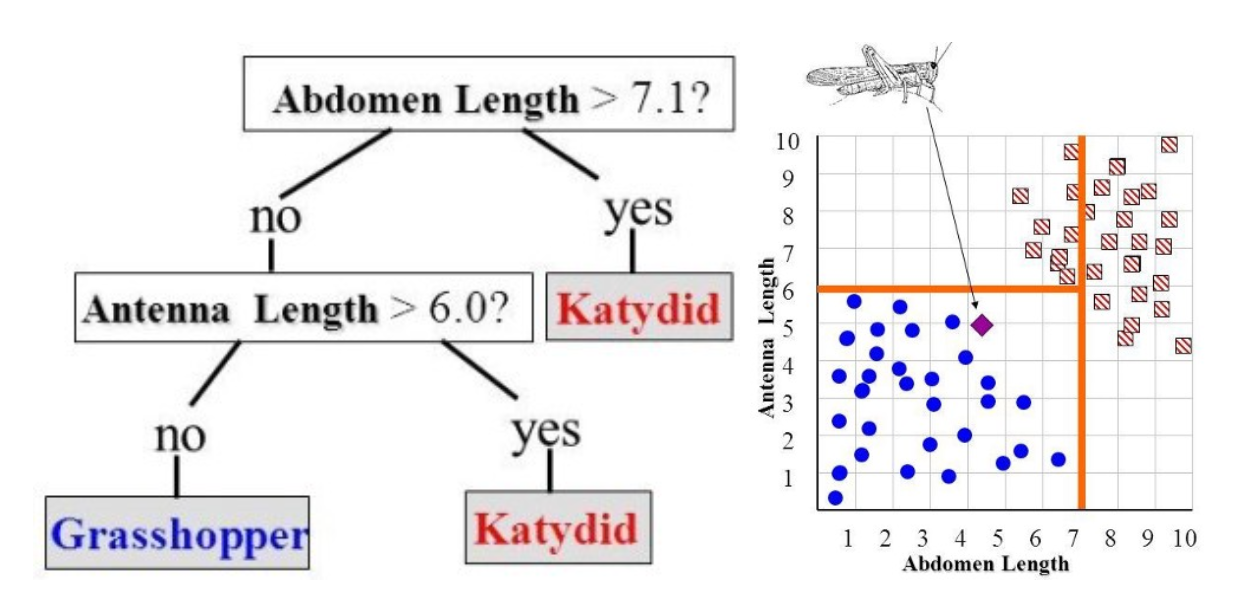


Figure: Classifying Grasshopper/ Katydid based on abdomen and antenna length

## Gaussian Naive Bayes

Gaussian Naive Bayes is a variant of Naive Bayes that follows Gaussian normal distribution and supports continuous data. Same as multinomial naive Bayes, this algorithm is based on the Bayes Theorem, which assumes that there is strong independence or no relationship between features/ variables. This is clearly not the case as we have seen in our data visualization as there is high correlation between RMSE, MI and SSIM. Therefore the “naive” in the name explains this naive assumption. However with this assumption, supervised learning models can be trained very efficiently, and only a small training data is needed to estimate the parameters needed for classification. Unlike multinomial naive bayes, Gaussian Naive Bayes works better for continuous data, like RMSE, MI and SSIM. When working with these kinds of continuous data, an assumption is that the continuous values of each class are distributed according to a Gaussian distribution.

Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution.

An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all what is needed to define such a distribution.

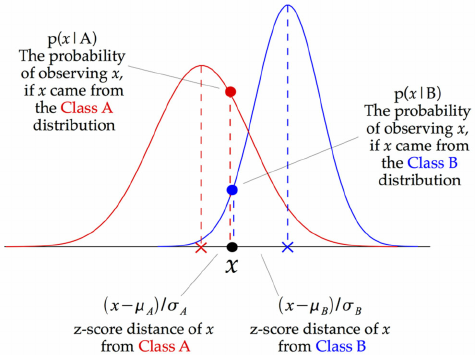


Figure: how a Gaussian Naive Bayes (GNB) classifier works

At every data point, the z-score distance between that point and each class-mean is calculated, namely the distance from the class mean divided by the standard deviation of that class.

# Methodology

* In this assignment I will be using 2 Regions(RRID): **RRID 30** and **RRID 69**. Each region has 3 datasets resulting from different scan speeds, S1, S3 and S5. Therefore, we have a total of 6 datasets.
* Each dataset have 6 Features and 1 Target:
  + **Features:**
    - **Categorical: stepweight, Delta T, Speed**
    - **Numerical: Root Mean Square Error (RMSE), Mutual Information (MI) and Structural Similarity (SSIM)**
  + **Target: Condition of x-ray BGA slice**
* Dataset is split into **80% training set** and **20% test set**
* Tests are done for each speed S1, S3 and S5 individually, then for all of them combined for RRID 30. This step is repeated for RRID 69.
* 1st attempt:
  + Multinomial Naive Bayes is used.
  + The categorial features are replaced by numbers, for example stepweight 0.1, 0.2, 0.3, 0.4, 0.5, 0.6 is converted into 0,1,2,3,4,5 respectively.
  + Each feature is scaled to be between 0 and 1 using SKlearn MinMaxScaler.
* 2nd attempt:
  + Multinomial Naive Bayes is used.
  + The categorial features are replaced by numbers, for example stepweight 0.1, 0.2, 0.3, 0.4, 0.5, 0.6 is converted into 0,1,2,3,4,5 respectively.
  + The features are not scaled.

### Training and Test Data Accuracy for Train/ Test Split ( Multinomial Naive Bayes)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset | Training Accuracy (Feature Scaled) | Test Accuracy  (Feature Scaled) | Training Accuracy | Test Accuracy |
| RRID30 S1 | 0.666667 | 0.444444 | 0.818181 | 0.444444 |
| RRID30  S3 | 0.757576 | 0.444444 | 0.848485 | 0.333333 |
| RRID30  S5 | 0.727273 | 0.444444 | 0.787879 | 0.333333 |
| RRID69  S1 | 0.939394 | 0.444444 | 0.909091 | 0.444444 |
| RRID69  S3 | 0.939394 | 0.333333 | 0.909091 | 0.444444 |
| RRID69  S5 | 0.878788 | 0.222222 | 0.878788 | 0.444444 |
| RRID30 compiled | 0.560000 | 0.576923 | 0.800000 | 0.769231 |
| RRID69  compiled | 0.640000 | 0.769231 | 0.640000 | 0.730769 |
| Average Accuracy | 0.763637 | 0.459936 | 0.823939 | 0.493055 |

Table 2: Accuracy for training and test data set (multinomial naive bayes)

As we can see with feature scaling, it reduces the accuracy for both training and test data. This is because our features are already within the range of 0 and 1, so it is not necessary to scale our features again, which makes it more inaccurate. If our dataset ranges differently, then feature scaling is a must to improve the performance of the prediction of the machine learning model. Another thing we can notice from the table is that multinomial naive bayes achieves an average of 0.823939 for the training data which is decent, but the test data set falls short with only an average accuracy of 0.493055 which is not good because almost half of the data sets given are predicted wrongly. Therefore, we can conclude that multinomial naive Bayes is not a suitable machine learning algorithm for our dataset. Next, we will check other machine learning algorithms to see whether those algorithms are better suited for our dataset..

## Spotcheck different machine learning algorithms

I will run logistic regression, linear discriminant analysis, K- Nearest Neighbor Classifier, Decision Tree Classifier, and Gaussian Naive Bayes, then compare the accuracy in predicting the quality of the image of the ball grid array using the K-fold Cross Validation method. K-fold Cross Validation is a resampling procedure used to evaluate the performance of machine learning algorithms with limited data samples, where k is the number of groups that the data sample will be split into. For example if k=10, it becomes 10-fold cross-validation. This kind of validation is mainly used to estimate the performance of a machine learning model on unseen data. It is quite popular because it is simple to understand and the results are generally less biased than other methods, such as a simple train/test split ( Brownlee, 2018). Each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times. “ Typically, given these considerations, one performs k-fold cross-validation using k = 5 or k = 10, as these values have been shown empirically to yield test error rate estimates that suffer neither from excessively high bias nor from very high variance.” - Page 184, An Introduction to Statistical Learning, 2013. The usual steps are as follows:

1. *The dataset is shuffled randomly*
2. *Split the dataset into k groups*
3. *For each unique group:*
   1. *Take the group as test data set*
   2. *Take the remaining groups as training data set*
   3. *Fit a model on the training set and evaluate the test data set*
   4. *Retain the evaluation score*
4. *Display the skill of the model using average accuracy or other metrics*

The algorithm with the most consistent highest accuracy will be chosen.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Dataset | Logistic Regression | Linear Discriminant Analysis | K Nearest Neighbor | Decision Tree | Gaussian Naive Bayes | SVM |
| RRID30 S1 | 0.75000 | 0.93333 | 0.75000 | 0.96667 | 0.93333 | 0.77500 |
| RRID30  S3 | 0.71667 | 1.00000 | 0.86667 | 1.00000 | 0.97500 | 0.89167 |
| RRID30  S5 | 0.74167 | 0.96667 | 0.83333 | 0.96667 | 0.94167 | 0.85833 |
| RRID69  S1 | 0.80833 | 0.92500 | 0.85833 | 0.92500 | 0.95000 | 0.83333 |
| RRID69  S3 | 0.81667 | 0.94167 | 0.88333 | 0.94167 | 0.97500 | 0.91667 |
| RRID69  S5 | 0.77500 | 0.94167 | 0.85000 | 1.00000 | 0.89167 | 0.86667 |
| RRID30 compiled | 0.76000 | 0.97000 | 0.87000 | 0.94000 | 0.96000 | 0.92000 |
| RRID69  compiled | 0.69000 | 0.91000 | 0.87000 | 0.86000 | 0.89000 | 0.84000 |

Table 3: Accuracy (5 s.d.) of different algorithms after undergo K-fold cross validation

As we can see, decision tree classifier, linear discriminant analysis and Gaussian naive bayes produce the best results among the six algorithms, thus we will use these 3 algorithms to build our machine learning model.

## Decision tree, linear discriminant analysis, Gaussian Naive Bayes

### Training Data Set Accuracy

|  |  |  |  |
| --- | --- | --- | --- |
| Training Dataset | Linear Discriminant Analysis | Decision Tree | Gaussian Naive Bayes |
| RRID30 S1 | 0.939394 | 1.000000 | 0.969697 |
| RRID30  S3 | 1.00000 | 1.00000 | 1.000000 |
| RRID30  S5 | 1.000000 | 1.000000 | 0.969697 |
| RRID69  S1 | 1.000000 | 1.000000 | 1.000000 |
| RRID69  S3 | 1.000000 | 1.000000 | 1.000000 |
| RRID69  S5 | 0.969697 | 1.000000 | 1.000000 |
| RRID30 compiled | 0.970000 | 1.000000 | 0.960000 |
| RRID69  compiled | 0.960000 | 1.000000 | 0.940000 |

Table 2: Training Data Accuracy when fitted to each machine learning algorithm

As we can see, all algorithms produce great results as we have expected from the k-fold cross validation test, but the best among the 3 is decision tree classifier because for every dataset in the training dataset it predicted correctly based on the given features! Now to test the robustness of these 3 algorithms, we are going to run through our test data through these algorithms to see whether the algorithms also work well with our small test datasets.

### 

### Test Data Set Accuracy

|  |  |  |  |
| --- | --- | --- | --- |
| Test Dataset | Linear Discriminant Analysis | Decision Tree | Gaussian Naive Bayes |
| RRID30 S1 | 0.777778 | 1.000000 | 1.000000 |
| RRID30  S3 | 1.00000 | 1.00000 | 1.000000 |
| RRID30  S5 | 0.888889 | 0.888889 | 0.888889 |
| RRID69  S1 | 0.888889 | 0.666667 | 0.555556 |
| RRID69  S3 | 0.888889 | 0.555556 | 0.555556 |
| RRID69  S5 | 0.777778 | 0.777778 | 0.666667 |
| RRID30 compiled | 1.000000 | 0.961538 | 1.000000 |
| RRID69  compiled | 0.961538 | 1.000000 | 1.000000 |
| Average Accuracy | 0.897970 | 0.856304 | 0.833334 |

Table 3: Test Data Accuracy when fitted to each machine learning algorithm

From the test datasets, we see that the decision tree classifier and Gaussian Naive Bayes’ accuracy is not that great for certain datasets, with the lowest accuracy reaching 0.555556 which is not good. This may be due to the test data sets having features and labels that are not trained in the model, thus it is not able to make a good prediction on the dataset. If the training data set is more complete I believe that these 2 algorithms will not have problems with accuracy of predicting. On the other hand, Linear Discriminant Analysis performs better than the two algorithms, with an average accuracy of 0.897970.

# Reference

1. “Expert.ai”.(2020). What is Machine Learning? A Definition. Retrieved from <https://www.expert.ai/blog/machine-learning-definition/#:~:text=Machine%20learning%20is%20an%20application,it%20to%20learn%20for%20themselves>.
2. Y. Kinha. (2020). An easy guide to choose the right Machine Learning algorithm. Retrieved from <https://www.kdnuggets.com/2020/05/guide-choose-right-machine-learning-algorithm.html>
3. J. Brownlee. (2018). A Gentle Introduction to k-fold Cross Validation. Retrieved from <https://machinelearningmastery.com/k-fold-cross-validation/>
4. S. McLeod. (2019). What does a box plot tell you? Retrieved from <https://www.simplypsychology.org/boxplots.html>
5. C. Maklin. (2019). Linear Discriminant Analysis in Python. Retrieved from <https://towardsdatascience.com/linear-discriminant-analysis-in-python-76b8b17817c2#:~:text=Linear%20Discriminant%20Analysis%20(LDA)%20is,as%20much%20information%20as%20possible>.
6. A. Chakure. (2019). Decision Tree Classification. Retrieved from <https://medium.com/swlh/decision-tree-classification-de64fc4d5aac>