Machine Learning – Project

Title: Wine Quality

Sources Created by:

Paulo Cortez (Univ. Minho), Antonio Cerdeira, Fernando Almeida, Telmo Matos and Jose Reis (CVRVV) @ 2009

Relevant Information:

These datasets can be viewed as classification or regression tasks. The classes are ordered and not balanced (e.g. there are munch more normal wines than excellent or poor ones).

Number of Instances: red wine - 1599; white wine - 4898.

Number of Attributes: 11 + output attribute.

Input variables (based on physicochemical tests):

1 - fixed acidity

2 - volatile acidity

3 - citric acid

4 - residual sugar

5 - chlorides

6 - free sulfur dioxide

7 - total sulfur dioxide

8 - density

9 - pH

10 - sulphates

11 - alcohol Output variable (based on sensory data):

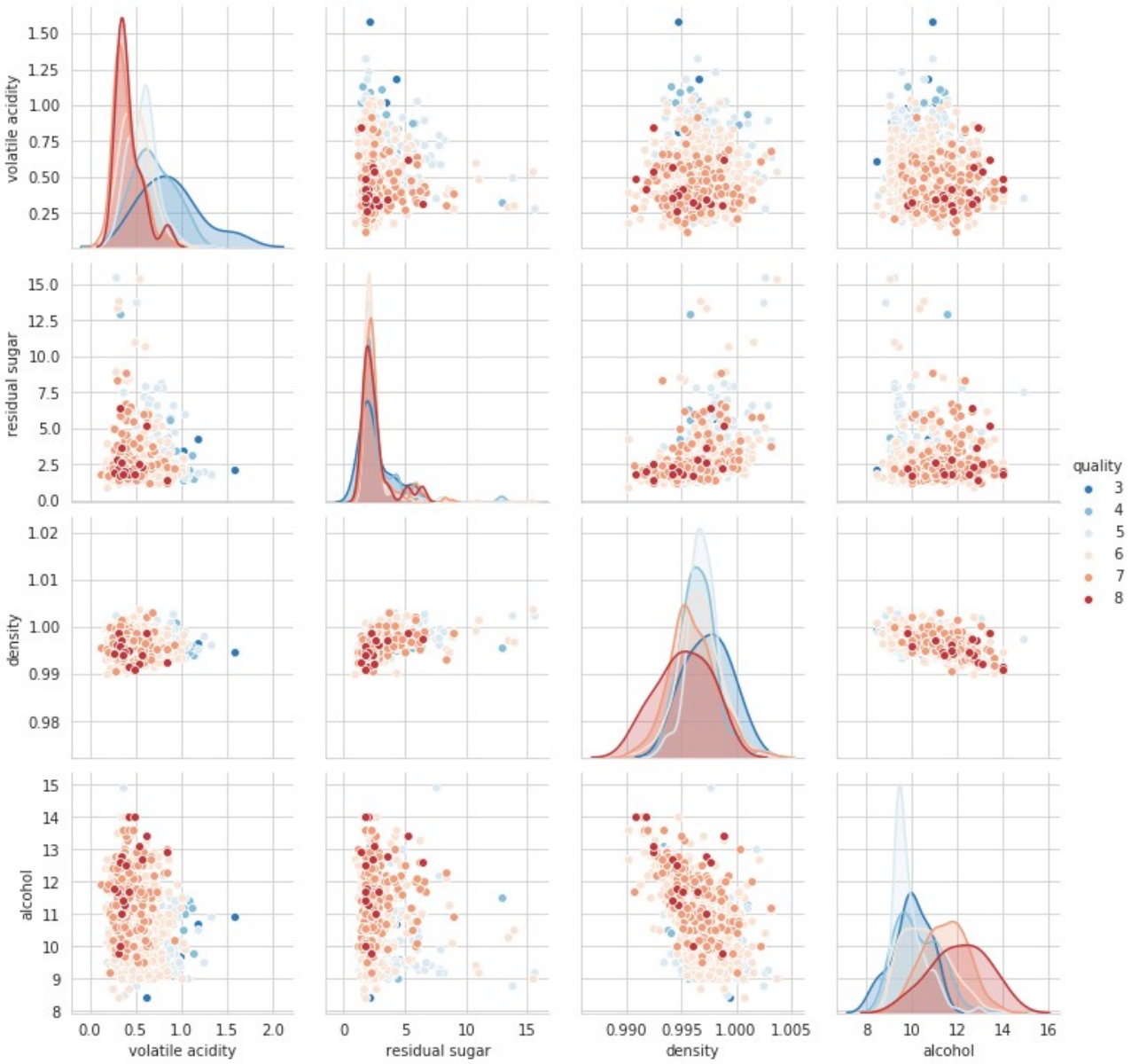
12 - quality (score between 0 and 10)

Missing Attribute Values: None

For more information please visit:

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=b1YFMKCuJYNL

Plot of pairwise relationships for chosen variables from red wines dataset.

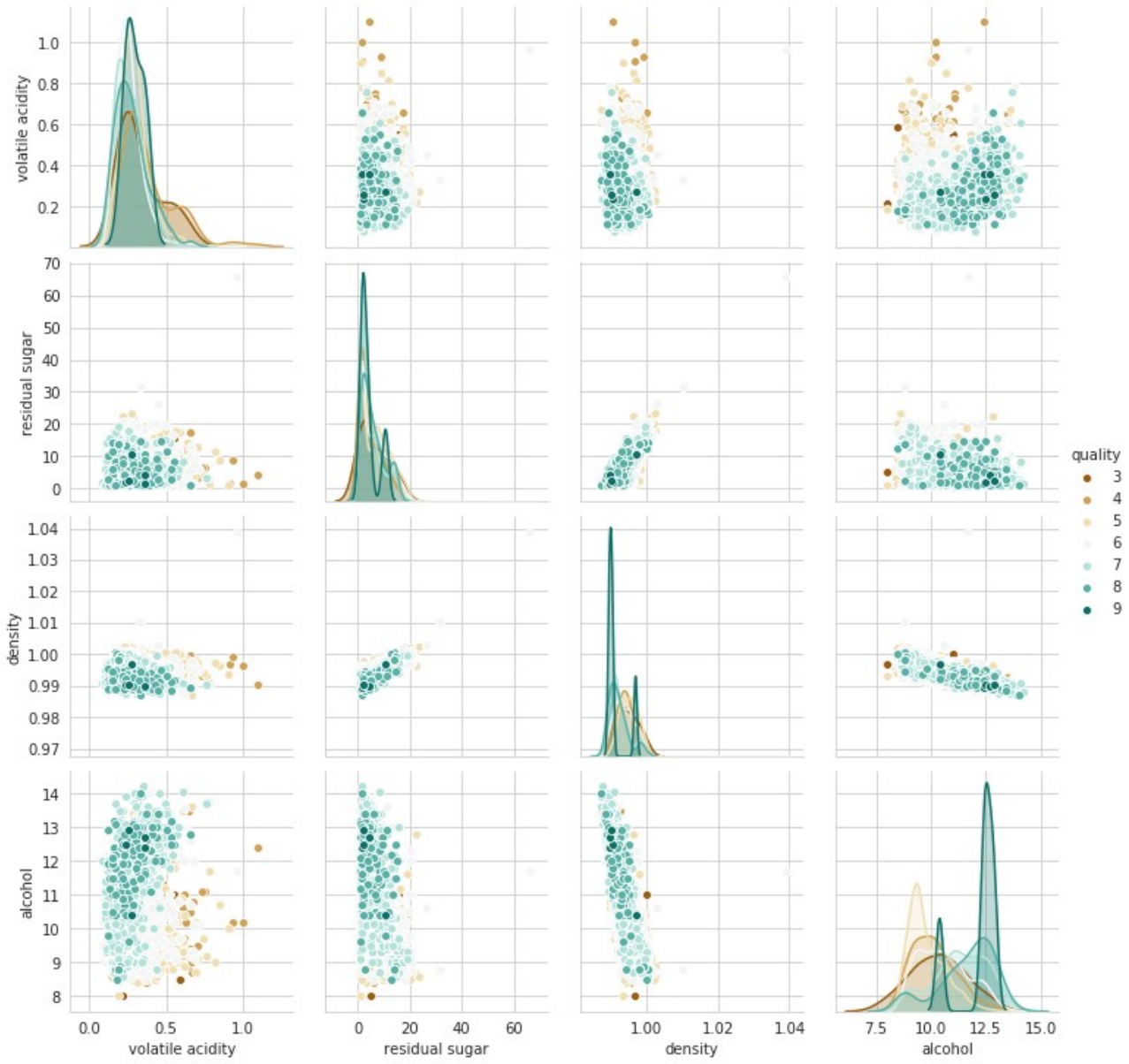


(The diagonal Axes show the univariate distribution of the data for the variable in that column).

For full pairplot please visit:

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=zu0U1ganNa8a

And another plot for white wines dataset.



Now, we will go through a few chosen classification and regression models, and then compare results in predicting quality based on different sets of attributes. Mean squared error was used as a measure tool for evaluating predictions accuracy. Each error is mean result of randomly sampling test and train set 30 times . The size of a train set was 1499 samples and for the test set - 100.

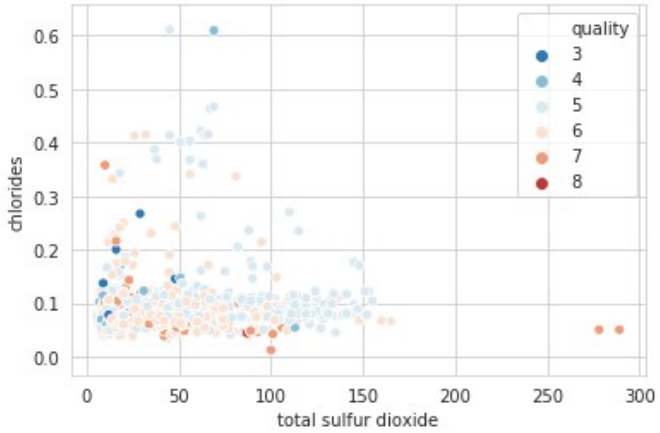
K nearest neighbors classifier

Attributes:

- total sulfur dioxide, chlorides

Best result for k = 20:

Error rate: 0.8022988505747126

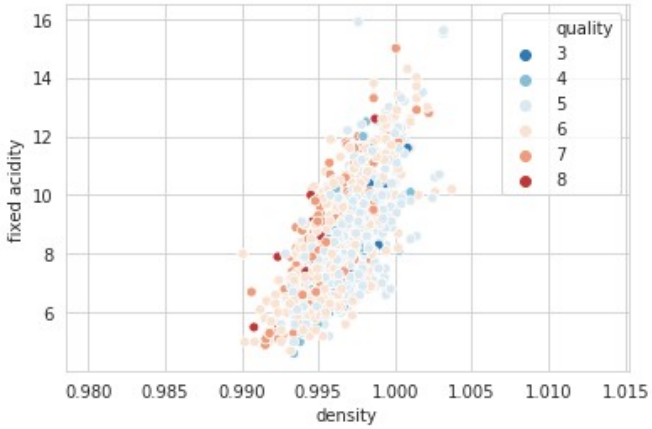


Attributes:

- fixed acidity, density

Best result for k = 50:

Error rate: 0.7866666666666668

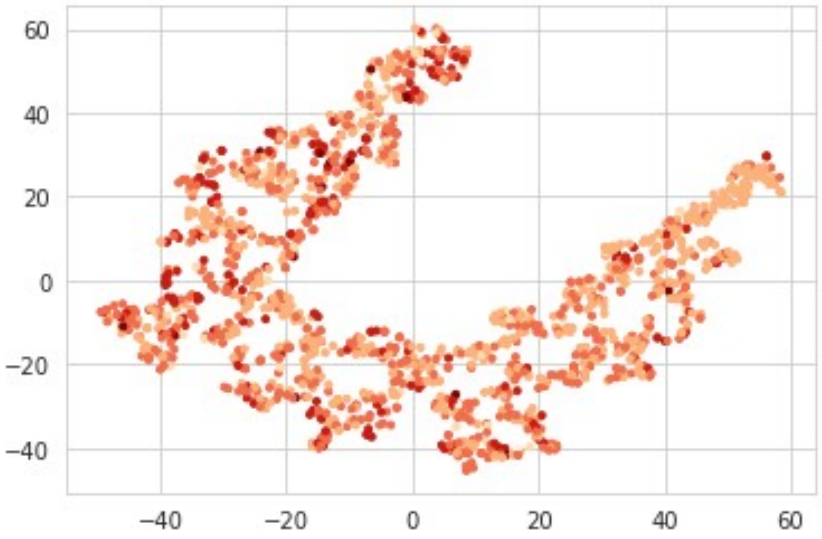


Links:

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=4cxDlkecnia5&line=3&uniqifier=1

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=355avJQYjrpq&line=7&uniqifier=1

Mean squared error rates using all features:



Best result for k = 100

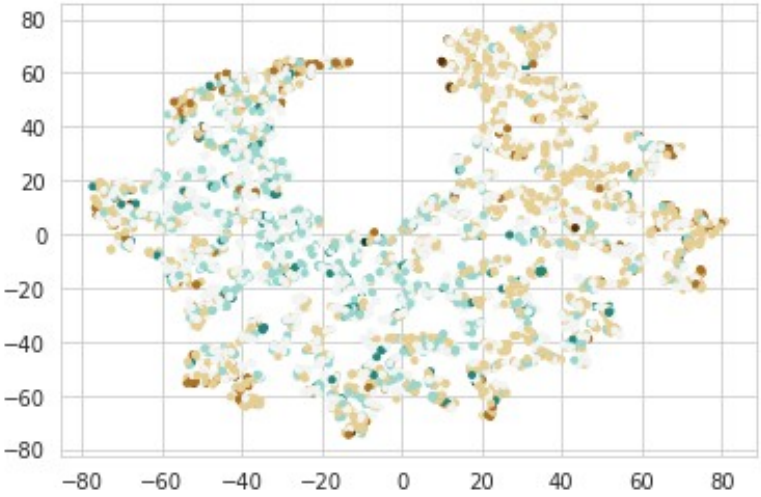
Error rate: 0.6907333333333334

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=_FGDIbu5otDk&line=2&uniqifier=1

Predictions are slightly better, but the difference is not satisfying considering we added 9 more features.

The picture above might explain why. TSNE from sklearn library was used to project all features to two dimensional space.

As we can see, there is no obvious clusters, thus the algorithm did not gain much precision by adding variables. So, let’s test each of them individually.



Mean squared error rates for each variable:

fixed acidity : 0.9503333333333335

volatile acidity : 0.7996666666666666

citric acid : 0.873

residual sugar : 0.9323333333333335

chlorides : 0.8983333333333335

free sulfur dioxide : 0.9570000000000001

total sulfur dioxide : 0.8833333333333331

density : 0.8466666666666666

pH : 0.9393333333333334

sulphates : 0.755

alcohol : 0.6633333333333334

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=rngzRg53pN20&line=8&uniqifier=1

We can see that for some attributes we can accomplish better results that for selected before pairs of variables. Also, predictions for ‘alcohol’ score higher that if we train algorithm on all available features. So I decided to check how (and if) we can improve using mix of the best attributes. Below are the results:

For attributes:

['volatile acidity', 'citric acid', 'chlorides', 'total sulfur dioxide', 'density', 'sulphates', 'alcohol'] 0.6645333333333334

For attributes:

['volatile acidity', 'citric acid', 'chlorides', 'density', 'sulphates', 'alcohol']

0.5780666666666667

For attributes: ['alcohol', 'sulphates', 'volatile acidity'] 0.5940666666666667

For attributes: ['alcohol', 'sulphates'] 0.6600666666666667

For attributes: ['alcohol', 'volatile acidity'] 0.6478666666666667

For attributes: ['sulphates', 'volatile acidity'] 0.7674666666666667

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=yI9kYOZyn9DC&line=13&uniqifier=1

K nearest neighbors Regression

For comparison, here are presented results of regression predictor for each variable:

fixed acidity : 0.6678533333333332
volatile acidity : 0.5939766666666667
citric acid : 0.6455566666666669
residual sugar : 0.7149366666666667
chlorides : 0.6754533333333332
free sulfur dioxide : 0.7119566666666665
total sulfur dioxide : 0.6607
density : 0.6327399999999999
pH : 0.6905033333333336
sulphates : 0.5658666666666667
alcohol : 0.5405

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=rngzRg53pN20&line=8&uniqifier=1

The predictions are significantly better for this model which is kind of expected because now the training algorithm not only tries to just label each wine correctly (whether it’s 5 or 7 for eg.) at all cost, but rather tries to come as close as possible to the most probable number.

Decision Trees

Some of the errors:

For attributes: ['alcohol']
0.6519999999999999

For attributes: ['alcohol', 'sulphates', 'volatile acidity']
0.6229999999999999

For all attributes:
0.6090000000000001

Mean squared error rates for each variable:
https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=Ml7nmiSy9-vs&line=24&uniqifier=1

For decision tree classification we have got very similar results as for the k-nearest neighbors algorithm. But there are some differences. Firstly, using all attributes gives the tree more information (so more information equals better predictions, which wasn't the case in previous algorithm). Secondly, decreasing number of training samples (both in decision trees and random forests) caused a deterioration of results, whereas in k-nearest algorithm it didn't have any significant impact on prediction correctness.

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=kbTJJprkcOLe&line=33&uniqifier=1

Random Forest Classifier

Mean err rate for features ['alcohol', 'sulphates', 'volatile acidity'] and max depth = 10:

0.45

Feature importance:

[0.37080489 0.3065306 0.32266451]

Random forest misclasification rate for each single feature:

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=tRxInIoWbRVZ&line=13&uniqifier=1

Mean err_rate for all attributes (with no restriction on depth of a tree):

0.38133333333333325

Error for depth: 2 is 0.5423333333333334

Error for depth: 3 is 0.5466666666666667

Error for depth: 4 is 0.4763333333333334

Error for depth: 6 is 0.4503333333333333

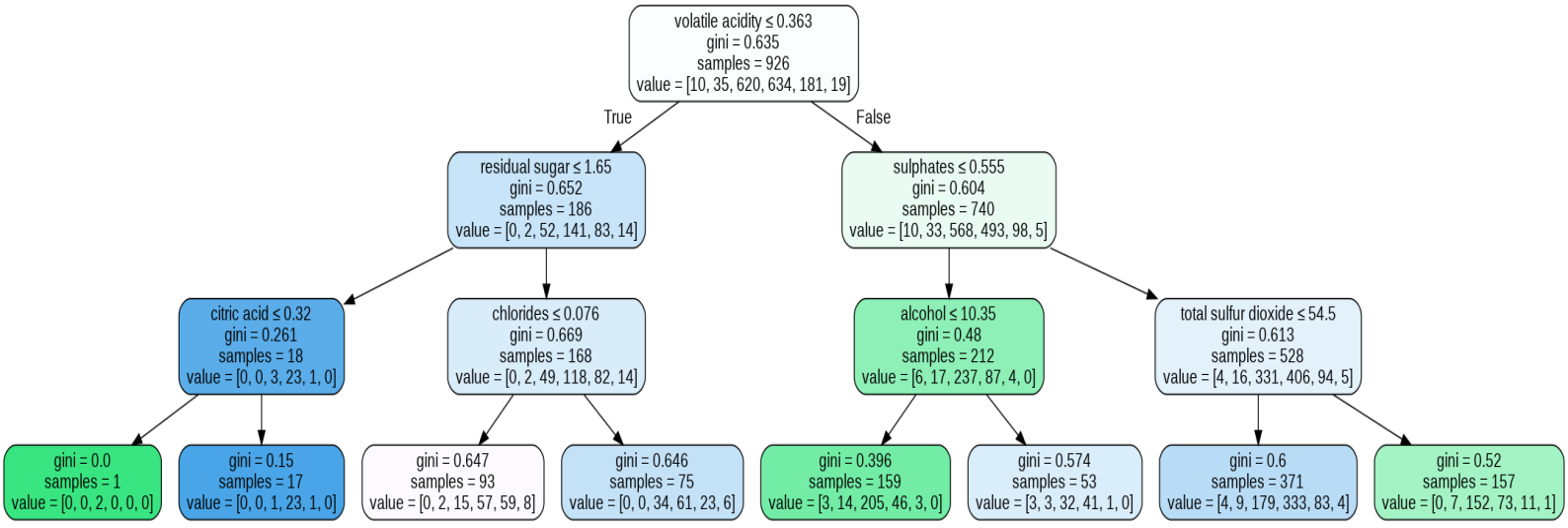
Error for depth: 8 is 0.41966666666666674

Error for depth: 10 is 0.3950000000000001

Error for depth: 100 is 0.38699999999999996

We can see, that to some point, increasing max depth of trees gives relevant gains in predicting the quality of wines. Interestingly, random forest containing only trees of depth = 2 scores better than any single decision tree. And decision tree using only three features is huge:

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=zNQv30GzFq-P&line=6&uniqifier=1



Example tree from random forest for depth = 3

Support Vector Machines

1. Classifier

Mean err_rate for all attributes: 0.6143333333333333

Error rates for each variable:

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=vhIE9FdStcMp&line=7&uniqifier=1

2. Regressor

Mean err_rate for all attributes: 0.4416683384574017

Error rates for each variable:

https://colab.research.google.com/drive/1LCbbKMQGpkAYQU-s4p7QQdV4ZGu3JJQ9#scrollTo=uKSErmlr2kTB&line=9&uniqifier=1

Again, we get better predictions using regression. However, random forest classifier scored highest when using all features, even compared with regression models.

Now, it's time to summarize tested algorithms and compare errors for each feature: