Quality Prediction in Mining Process

Guide : Dr. Sathyan Subbiah

Submitted by : Hari Ranjan Meena (ME18B108)

Contents

- Problem statement
- Dataset
- Linear regression
- kNN regression
- Artificial neural networks
- Support vector Regression
- Regression trees
- Random forest
- AdaBoost decision tree
- Comparison of results

Problem Statement

Froth flotation is a process for selectively separating hydrophobic materials from hydrophilic ones. This is used in mineral processing, paper recycling and waste-water treatment industries. Flotation process is used to concentrate the iron ore in mineral processing stage in mining industry.

We want to predict the concentration of silica (impurity) in ore concentrate which helps in improving the process parameters and take prior corrective actions.

Data Set

Variables:

- Iron feed %
- Silica feed %
- Starch Flow
- Amina Flow
- Ore Pulp Flow
- Ore Pulp pH

- Ore Pulp Density
- Flotation Column Air Flow (01 - 07)
- Flotation Column Level (01 - 07)
- Silica concentrate
- Iron concentrate

There are 737453 data-points, 23 variables in total and all of it was used. Around 70% was used to train the AI and 30% was to test and obtain the results.

Data set sample

% Iron Feed	% Silica Feed	Starch Flow	Amina Flow	Ore Pulp Flow	Ore Pulp pH	Ore Pulp Density	Flotation Column 01 Air Flow
55,2	16,98	3019,53	5,57,434	3,95,713	1,00,664	1,74	2,49,214
55,2	16,98	3024,41	5,63,965	3,97,383	1,00,672	1,74	2,49,719
55,2	16,98	3043,46	5,68,054	3,99,668	10,068	1,74	2,49,741
55,2	16,98	3047,36	5,68,665	3,97,939	1,00,689	1,74	2,49,917
55,2	16,98	3033,69	5,58,167	4,00,254	1,00,697	1,74	2,50,203

Linear Regression:

Introduction:
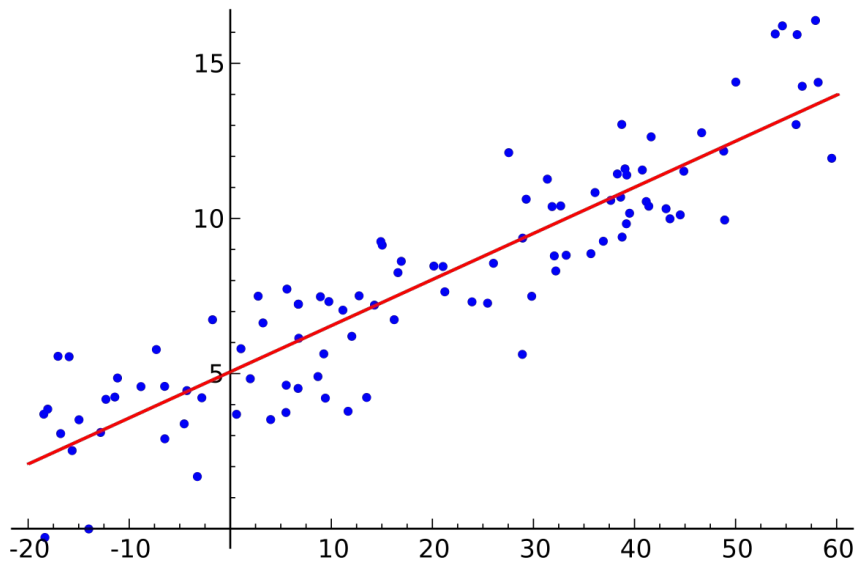
We used simple linear regression

- Sum of squared deviations from the resultant line is minimized
- Faster to compute (faster than all other methods)
- However it can yield only linear equations of the form:

$$Y = Ax + B$$

A is the slope; B is the Y-intercept

Linear Regression:

Results:

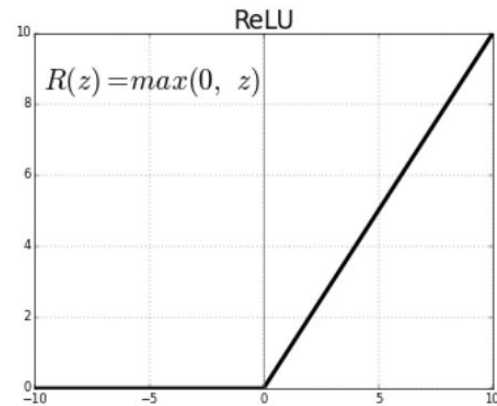| MSE	Error <1%	Error <2%	Error <5%	Error <10%	Error <20%	Error <50%
0.40484924	02.9679	05.8883	14.3083	27.9724	54.1006	89.2685

- This method takes much lesser time to train and test

- It is often used to predict the nature of relationships not to obtain the accurate relationship itself

- It produces significantly inferior results compared to some other algorithms. Linear regression couldn't yield more than 90% accuracy even for <50% error
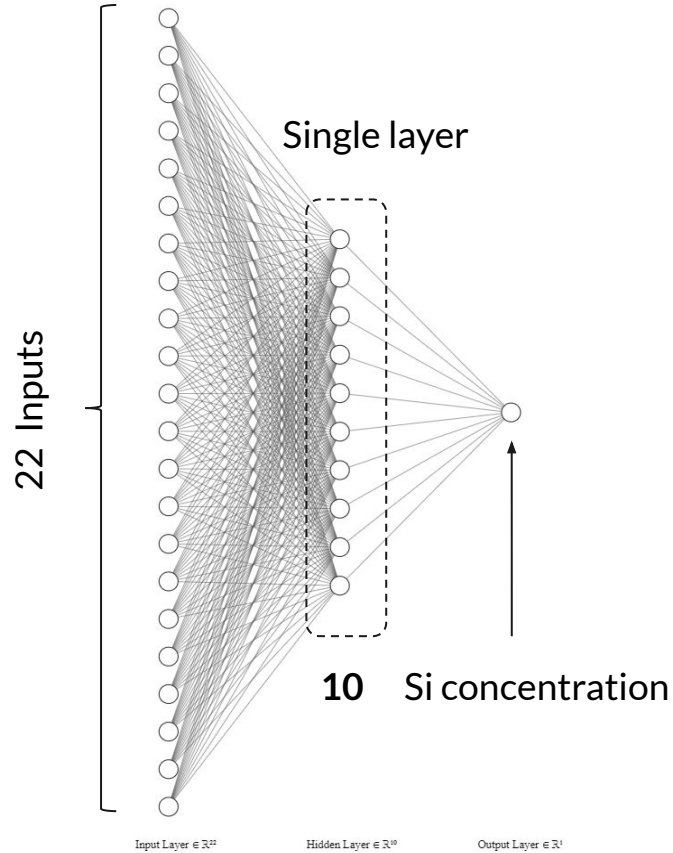
Artificial Neural networks

Input Layer ∈ R⁵   Hidden Layer ∈ R³   Output Layer ∈ R¹
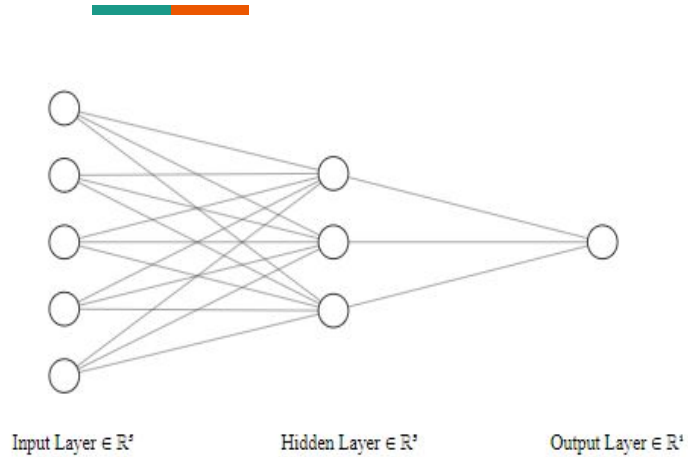
$$\text{ReLU}\left[\begin{pmatrix} W_{11} \dots \dots W_{1n} \\ W_{21} \dots \dots W_{2n} \\ \ . \\ \ . \\ W_{L1} \dots \dots W_{Ln} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ . \\ . \\ . \\ X_n \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ . \\ . \\ . \\ b_L \end{pmatrix}\right] = \begin{pmatrix} Y_1 \\ Y_2 \\ . \\ . \\ . \\ Y_L \end{pmatrix}$$
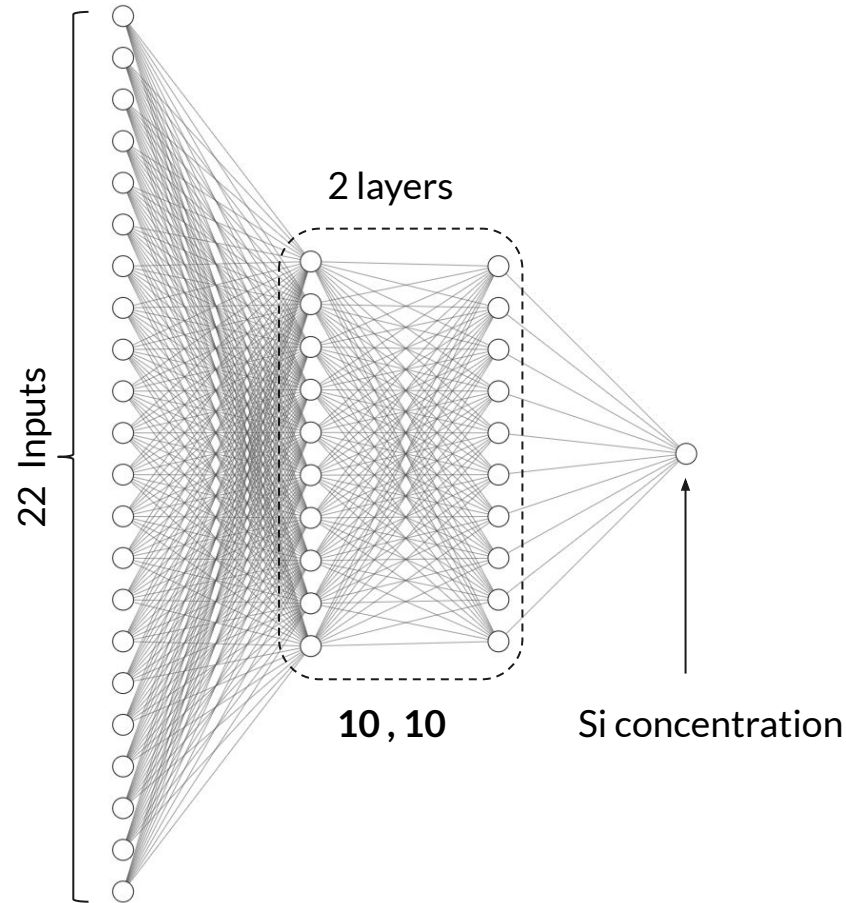
Learning ⟶ Finding the right weights and biases

ReLU

$R(z) = max(0, \ z)$

Artificial Neural networks

Single layer

22 Inputs

10 Si concentration

Input Layer ∈ ℜ²²   Hidden Layer ∈ ℜ¹⁰   Output Layer ∈ ℜ¹

Accuracy	No . of nodes in the hidden layer		
	5	**10**	**20**
MSE	0.459	0.532	1.05
1%	2.62 %	2.32 %	1.59 %
2%	5.25 %	4.63 %	3.19 %
5%	13.32 %	11.91 %	7.80 %
10%	26.44 %	24.43 %	15.35 %

Artificial Neural networks

No of Nodes	5,5	10,10	20,20
MSE	0.468	0.447	0.417
1%	2.59 %	2.94 %	2.63 %
2%	5.19 %	5.92 %	5.32 %
5%	12.99 %	14.44 %	13.32 %
10%	26.23 %	27.5 %	26.53 %

K nearest neighbours(*k*NN)

kNN regression:
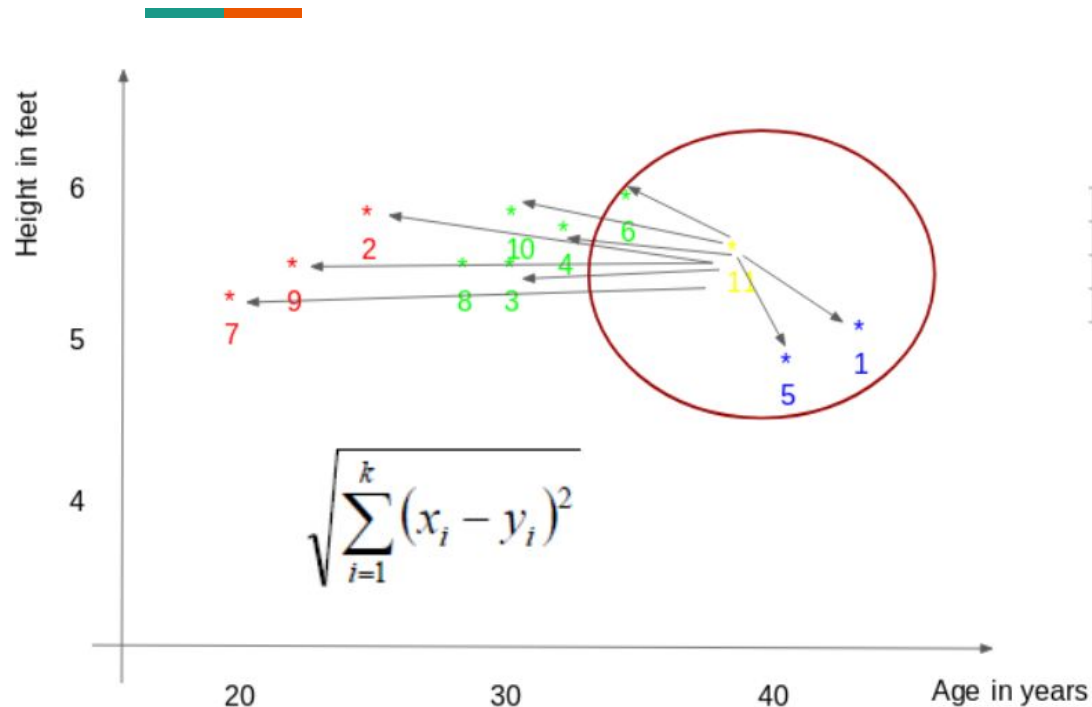
ID	Height	Age	Weight
1	5	45	77
5	4.8	40	72
6	5.8	36	60

ID 11 = (77+72+60) / 3

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

- Value of **k** determines the number of nearest points to be considered.

K nearest neighbours(*k*NN)

Results :

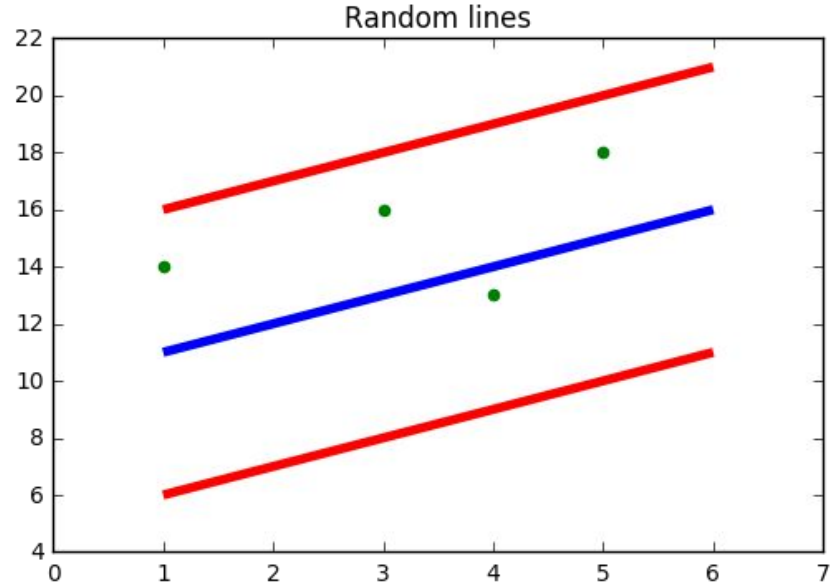| K value	K = 3	K = 5	K = 10
MSE	0.470	0.481	0.522
1%	34.67 %	24.26 %	15.74 %
2%	37.61 %	27.90 %	19.76 %
5%	45.08 %	37.23 %	29.97 %
10%	55.35 %	49.63 %	43.0 %

Support Vector Regression

Here, we use the concept of Support Vector Machines to solve the current regression problem

Terminology:

1. Kernel
2. Hyperplane
3. Boundary Line
4. Support Vectors

Random lines

Support Vector Regression (Contd.)

In simple regression we try to minimise the error rate. While in SVR we try to fit the error within a certain threshold.

We can say that the Equation of the hyper plane is wx+b=0

So we can state that the two the equation of the boundary lines as

$$Wx+b=+e$$
$$Wx+b=-e$$

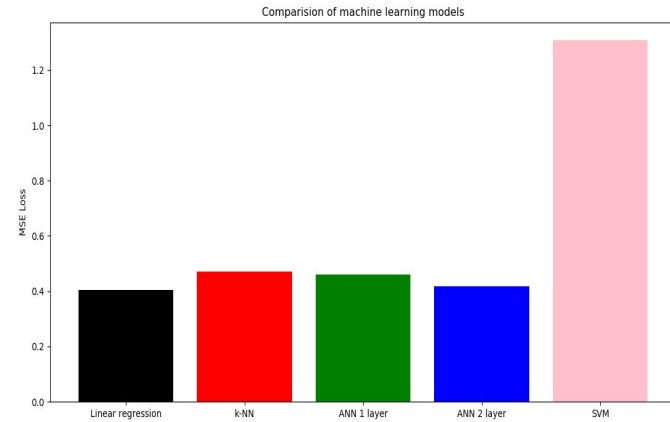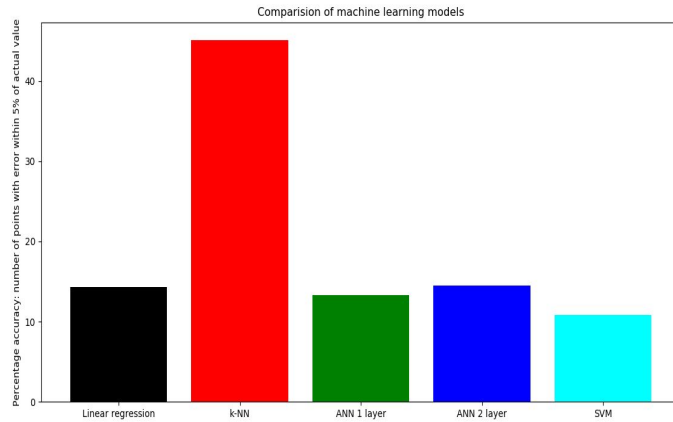So, the equation that satisfies our SVR is:

$$e \leq y-Wx-b \leq +e$$

Support Vector Regression (Contd.)

Results:

MSE	1.3029177550514643	
Accuracy	Error <1%	2.7%
	Error <2%	5.7%
	Error <5%	21.7%
	Error <10%	34.6%

Results

Comparision of machine learning models

Comparision of machine learning models

Regression tree - Introduction

- Decision tree models are nested if-else conditions
- Each node in a decision tree represents a feature and a threshold
- The leaf nodes represent the output value - Continuous in case of regression, discrete in case of classification
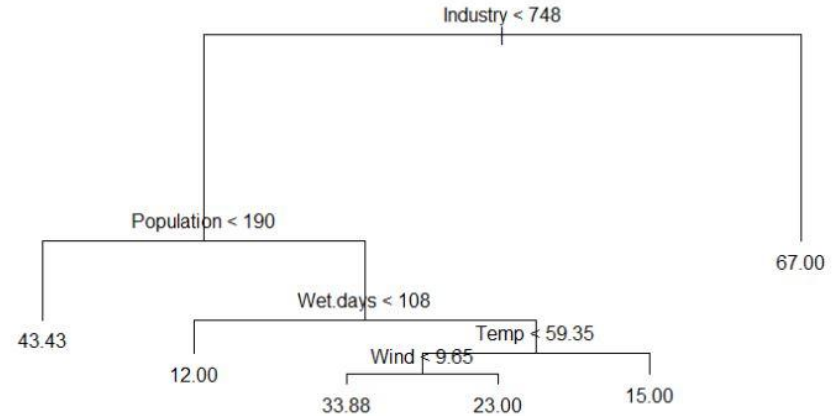- Easy to interpret

Figure-1: Decision Tree of Pollution Dataset

Regression tree - Algorithm

- There are majorly two steps involved:
  - We divide the predictor space set of possible feature variables into distinct and non-overlapping regions.
  - For every observation that falls into a region, we make the prediction which is the mean of response in the training set in that particular region.

$$R_1(j, s) = \{X | X_j \le s\} \text{ and } R_2(j, s) = \{X | X_j > s\}.$$

Figure-3: For any variable j and splitting point s

$$\sum_{i:\ x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:\ x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

Figure-4: RSS of Recursive Splitting

Regression tree - Results

MSE	Error < 1%	Error < 2%	Error < 5%	Error < 10%
0.009402	98.2290	98.4139	98.6589	98.9328

Random forest - Introduction

- Decision trees are very specific to the model on which they are trained
- Decision trees usually result in an overfitted model
- Generalization done using "bootstrap aggregation" (Bagging)
- Bagging is a technique used to reduce variance

Random forest - Algorithm

- We generate multiple decision trees.
- Sample a subset of training data. Create a decision tree from the sampled data
- Create multiple decision trees. Each decision tree is trained using a different subset of training data.
- The output for a test is the mean of outputs across all decision trees

Random forest - Results

n_trees	MSE	Error < 1%	Error < 2%	Error < 5%	Error < 10%
1	0.013792	97.0913	97.3946	97.8051	98.2186
2	0.007684	95.5591	96.0481	96.9277	98.0012
3	0.006035	94.0308	94.8404	96.3844	98.0315
4	0.004958	92.8692	93.9594	96.1466	98.1730
5	0.004316	91.8946	93.3279	96.1688	98.3737

AdaBoost decision tree - Introduction

- Decision trees are very specific to the model on which they are trained
- Decision trees usually result in an overfitted model
- We use boosting to create a strong classifier from a number of weak classifiers
- Boosting is a method in which we fit a first build a model from training data and fit a second model to train the errors of first model and third model to train the errors of second model and so on.
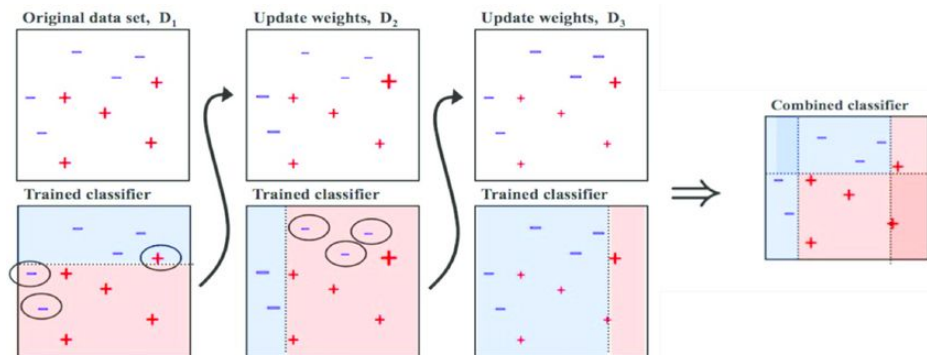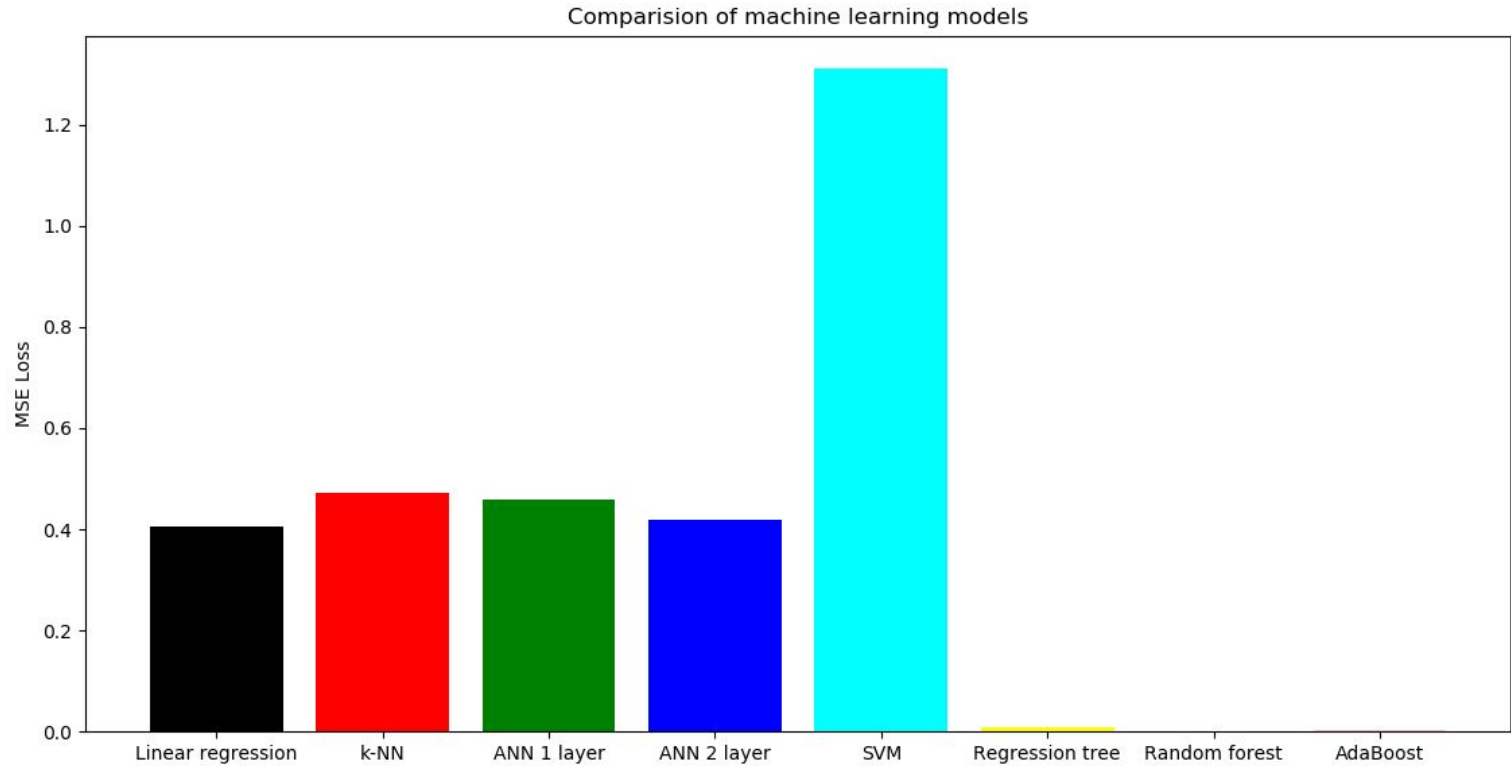
AdaBoost decision tree - Algorithm

- Every observation x_i is assigned a weight $w_i = 1/N$ where N is the total number of observations
- We first fit a decision tree and calculate error.
- Error $= \Sigma \, w_i \, error_i$
- If observation is
  - Incorrectly predicted: Increase weight w_i of x_i
  - Correctly predicted: Decrease weight w_i of x_i
- Repeat the same process till certain number of tree estimators
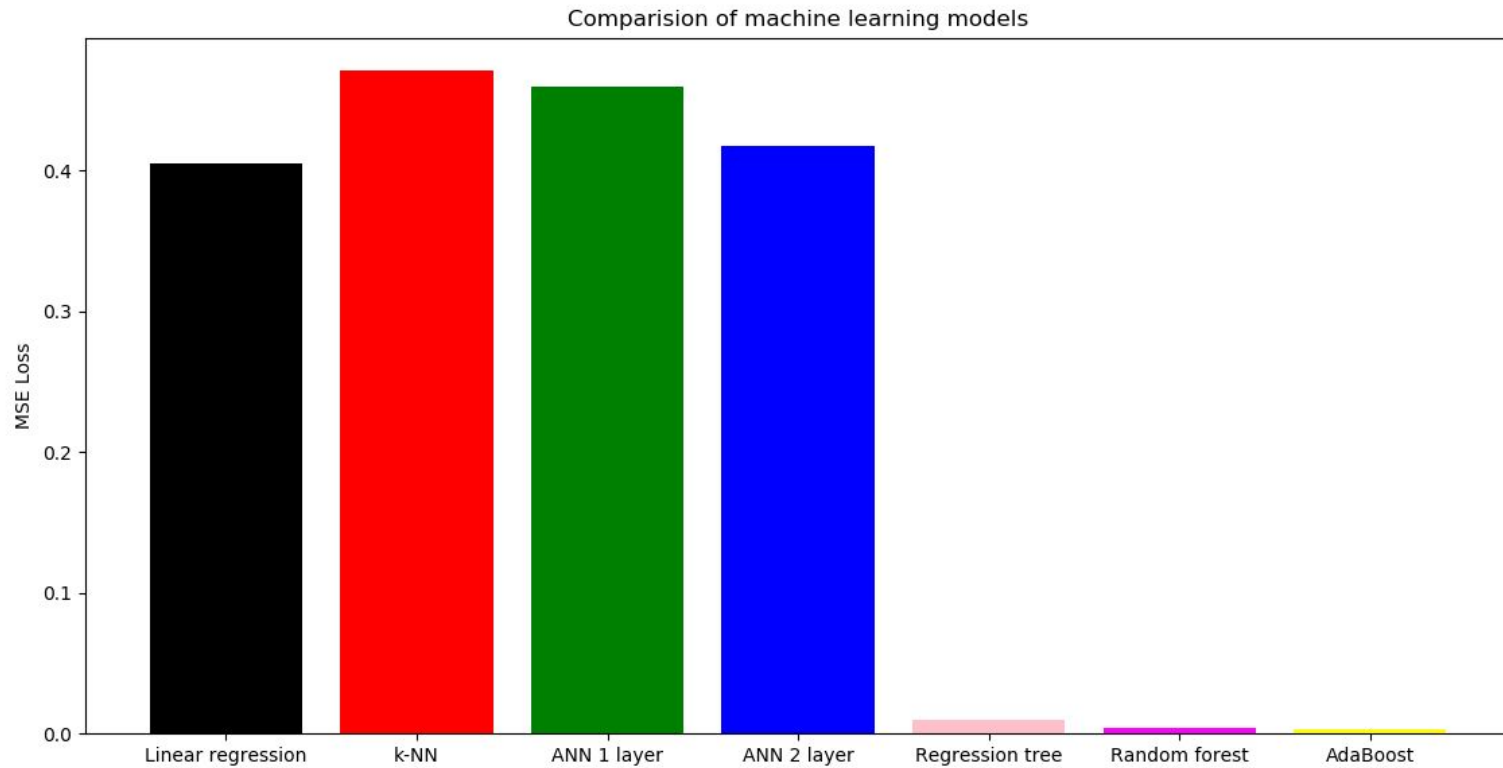
Adaboost decision tree - Results (4 estimators)

MSE	Error < 1%	Error < 2%	Error < 5%	Error < 10%
0.002810	99.1452	99.2908	99.4508	99.5661

Results

Comparision of machine learning models

Results

Comparision of machine learning models

Results

Comparision of machine learning models