Challenge 4 Report

Kiyeob Lee, Amrita Veerabagu, Harinath

I. Introduction

In challenge4, we studied two topics: Linear Regression and Decision Tree. Both Linear Regression and Decision Tree belong to the supervised learning and a setup for the models is the following: Given a training dataset $D_n = \{(x_1, y_1), \cdots, (x_n, y_n) \mid n \in \mathbb{N}\}$ where $x_i \in \mathbb{R}^d$ is an instance of the dataset and y_i is the corresponding true known label, our objective is to predict unknown $\hat{y}_{n+1}$ provided a new data instance x_{n+1} comes in to the system. Linear regression is a model that starts with an assumption that there is a relationship between x_i and y_i. Decision Tree's objective is, similarly, given a training set, to learn which tree minimizes the misclassification error(or maximize the correct-classification). One way to implement decision tree is to use entropy method that you identify a classifier that maximizes an 'information gain' as well as to use GeNie method as learned in a tutorial.

II. Linear Regression

One way to measure the misclassification is to use the squared-loss function, namely,

$$\mathcal{L}(h, (x, y)) = (h(x) - y)^2 \tag{1}$$

where $h(x)$ is a mapping of x into y and $\mathcal{L}$ is a penalty of difference between $h(x)$ and y. From this loss function, we define the Mean Squared Error(MSE), namely,

$$\mathcal{L}(h) = \frac{1}{m}\sum_{i=1}^{m}(h(x_i) - y_i)^2 \tag{2}$$

Since we stared with an assumption that there is a relationship between between x_i and y_i and we have chosen the error is L2 norm difference, we can formulate the problem as a minimization problem:

$$\arg\min_w \mathcal{L}(h_w) = \arg\min_w \frac{1}{m}\sum_{i=1}^{m}(\langle w, x_i\rangle - y_i)^2 \tag{3}$$

where $h(x_i)$ in equation (2) is replaced with an inner product by linearity assumption. To solve the problem, we can get the gradient of the objective function to be zero. That is,

$$\frac{2}{m}\sum_{i=1}^{m}(\langle w, x_i\rangle - y_i)x_i = 0 \tag{4}$$

which we can rewrite as $Aw = b$ where

$$A = \big(\sum_{i=1}^{m} x_i x_i^T\big) \qquad \text{and} \qquad b = \sum_{i=1}^{m} y_i x_i \tag{5}$$

If A is invertible, then the solution to the Empirical Risk Minimizer(ERM) problem is

$$w = A^{-1}b \tag{6}$$

If A is not invertible, it is still solvable, under a linearlity assumption, because b is in the range of A.

III. Decision Tree

The Classification and Regression Tree (CART) builds a multiway tree, finding for each node the categorical feature that will yield the largest information gain for categorical targets. This is an example for a greedy algorithm. The mathematical formulation of the binary tree is given below.

Given a training vector $x_i \in R_n, i = 1, , l$ and its corresponding label value $y \in R_l$, a decision tree recursively partitions the space such that the samples are grouped together.

Let the data at node m be represented by Q. For each candidate split $\theta = (j, t_m)$ consisting of a feature *j* and threshold t_m, partition the data into $Q_{left}(\theta)$ and $Q_{right}(\theta)$ subsets,

$$Q_{left}(\theta) = (x, y)|x_j <= t_m$$
$$Q_{right}(\theta) = Q/Q_{left}(\theta)$$

The impurity at m is computed using an impurity function $H()$, the choice of which depends on the task being solved (classification or regression). Here, we use the H() corresponding to classification, given by

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk})$$

where p_{mk} is the proportion of class k observations in node m and is given by,

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$$

In the above equation, we consider that the target is a classification outcome taking on values $0, 1, , K - 1$, for node m, representing a region R_m with N_m observations.

Therefore the impurity at node m is given by,

$$G(Q, \theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta))$$

Thus we select the parameters which minimises the impurity,

$$\theta^* = argmin_\theta G(Q, \theta)$$

Recurse for subsets $Q_{left}(\theta)$ and $Q_{right}(\theta)$ until the maximum allowable depth is reached, $N_m < min_{samples}$ or $N_m = 1$.

IV. Observations Part 1-3

1) **Part 1** : In part 1, we built and predicted the linear regression model on same type of dataset i.e Trained model on White-Wine training set and predicted it on White-Wine testing with RMSD of 0.71 on test set. Similarly in case of Red-wine dataset, obtained a RMSD of 0.674 on test set. Model fit (RSq) of White wine - 0.984 and Red Wine - 0.987. So we can see that both model fit and test RMSD are good when model is trained and predicted on same type of dataset. Please look at the table for coefficient values of features from individual regression models.
2) **Part 2** : In part 2, decision tree model was built based on 11 input features and output is type of wine. Obtained an accuracy of 0.9953 on test set.
3) **Part 3** : In part 3, we first divided both red and white wine datasets in the ratio 80% and 20% as its training and test set. Then we trained two linear regression models using red-wine and white-wine training sets, Model1 and Model2 respectively. In order to test how well the models are performing on a different test set, the red-wine model was tested on white-wine test set and white-wine model was tested on red-wine dataset.
   The Model 1 gave an MSE of 0.401 and 0.729 in the red-wine and white-wine test set respectively. The Model 2 gave an MSE of 0.941 and 0.616 in the red-wine and white-wine test set respectively.

Feature	Relative Importance - Decision tree	Model Alpha - White Wine	Model Alpha - Red Wine
Total sulfur oxide	0.67	-0.0009	-0.0039
Chlorides	0.21	-1.157	-1.8033
Volatile acidity	0.06	-1.937	-1.0278
Sulphates	0.024	0.4167	0.9907
Density	0.017	1.9986	4.3731
pH	0.008	0.214	-0.5108
Alcohol	0.005	0.358	0.2931
Residual sugar	0.003	0.0236	0.0046
Fixed acidity	0.002	-0.052	0.0057
Free sulfur oxide	0.001	0.0046	0.0066
Citric acid	0.0001	-0.0189	-0.0732

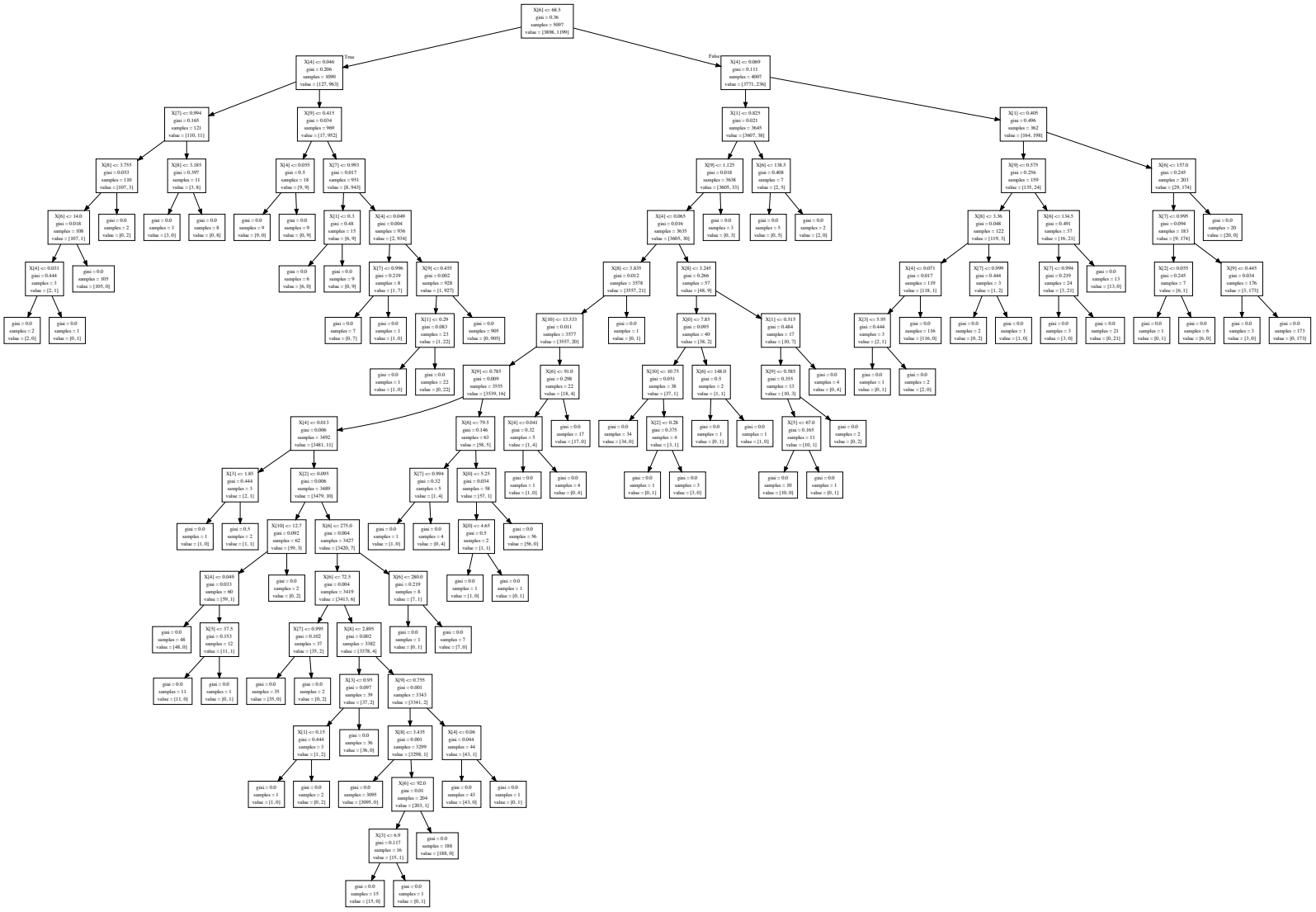
V. Observations Part 4

1) From part 2 results , we can see that `DecisionTreeClassifier` works very well on boht training and test set : which means given set of features can classify the data into wine type with great accuracy (0.9952). This means that data is well separable with respect to set of input features.
2) We can see the features being ranked based on importance values : The importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature. It is also known as the Gini importance. So in the table we see that Total Sulfur Oxide and Chlorides are top two features which help classify the data.
3) For these features we have also plotted the box plots with respect to wine type. (0- White, 1- Red). We can observe that box plots for 'chlorides' and 'total sulfur oxide' looks completely different for Red and White wine types. Now

this indicates why they were picked by Decision tree classifier as top features. Also we see that their ranges are also different among the types.

4) We can observe similar changes in range of features among types in case of 'pH', 'density', 'fixed acidity', 'volatile acidity' etc. This means the combination of features space in case of White Wine is different than feature space of Red Wine! And this is why we have good result in DecisionTreeClassifier.
5) $$\hat{y}_i = \alpha_0 + \sum_j \alpha_i x_{i,j}. \tag{7}$$

   Now in the case of regression ; as seen in the equation above each feature has their coefficient value $\{\alpha_i\}$ found out by Least squares estimate. We can see the alpha values for each features tabulated along with decision tree importance results.
6) Interesting observation is that : when we train model on White-Wine datasst and try to predict red-wine data we see that RMSD is bad compared to RMSD on Red-wine on Red-wine. Similarly we see that RMSD while predicting white dataset using model trained on Red is poor with respect to model trained on White dataset.
7) The reason being, when you train model on white wine dataset , compute alpha values and use them to predict red wine dataset (which has different ranges of X compared to white wine) predictions will be slightly off range since take product of α_i with X_i while predicting it.
8) From decison tree classfier results and box plots we can infer that input space of features in case of Red White is well separated. Hence when alpha values from one type pf regression model is multiplied with completely different range of X values (extrapolation) usually doesnt guarantee good predictions.


X[6] <= 68.5
gini = 0.36
samples = 5097
value = [3898, 1199]
True
False
X[4] <= 0.046
gini = 0.206
samples = 1090
value = [127, 963]
X[14] <= 0.069
gini = 0.111
samples = 4007
value = [3771, 236]
X[7] <= 0.994
gini = 0.165
samples = 121
value = [110, 11]
X[9] <= 0.415
gini = 0.034
samples = 969
value = [17, 952]
X[1] <= 0.825
gini = 0.021
samples = 3645
value = [3607, 38]
X[1] <= 0.405
gini = 0.496
samples = 362
value = [164, 198]
X[8] <= 3.755
gini = 0.053
samples = 110
value = [107, 3]
X[8] <= 3.185
gini = 0.397
samples = 11
value = [3, 8]
X[4] <= 0.055
gini = 0.5
samples = 18
value = [9, 9]
X[7] <= 0.993
gini = 0.017
samples = 951
value = [8, 943]
X[9] <= 1.125
gini = 0.018
samples = 3638
value = [3605, 33]
X[6] <= 138.5
gini = 0.408
samples = 7
value = [2, 5]
X[9] <= 0.575
gini = 0.256
samples = 159
value = [135, 24]
X[6] <= 157.0
gini = 0.245
samples = 203
value = [29, 174]
X[6] <= 14.0
gini = 0.018
samples = 108
value = [107, 1]
gini = 0.0
samples = 2
value = [0, 2]
gini = 0.0
samples = 3
value = [3, 0]
gini = 0.0
samples = 8
value = [0, 8]
gini = 0.0
samples = 9
value = [9, 0]
gini = 0.0
samples = 9
value = [0, 9]
X[1] <= 0.3
gini = 0.48
samples = 13
value = [6, 9]
X[4] <= 0.049
gini = 0.004
samples = 938
value = [2, 936]
X[4] <= 0.065
gini = 0.016
samples = 3635
value = [3605, 30]
gini = 0.0
samples = 3
value = [0, 3]
gini = 0.0
samples = 5
value = [0, 5]
gini = 0.0
samples = 2
value = [2, 0]
X[8] <= 3.36
gini = 0.048
samples = 122
value = [119, 3]
X[6] <= 134.5
gini = 0.491
samples = 37
value = [16, 21]
X[7] <= 0.995
gini = 0.094
samples = 181
value = [9, 174]
gini = 0.0
samples = 20
value = [20, 0]
X[4] <= 0.031
gini = 0.444
samples = 3
value = [2, 1]
gini = 0.0
samples = 105
value = [105, 0]
gini = 0.0
samples = 6
value = [6, 0]
gini = 0.0
samples = 9
value = [0, 9]
X[7] <= 0.996
gini = 0.219
samples = 8
value = [1, 7]
X[9] <= 0.455
gini = 0.002
samples = 928
value = [1, 927]
X[8] <= 3.835
gini = 0.012
samples = 3578
value = [3557, 21]
X[8] <= 3.245
gini = 0.266
samples = 57
value = [48, 9]
X[4] <= 0.071
gini = 0.017
samples = 119
value = [118, 1]
X[7] <= 0.999
gini = 0.444
samples = 3
value = [1, 2]
X[7] <= 0.994
gini = 0.219
samples = 24
value = [3, 21]
gini = 0.0
samples = 13
value = [13, 0]
X[2] <= 0.055
gini = 0.245
samples = 7
value = [6, 1]
X[9] <= 0.445
gini = 0.034
samples = 176
value = [3, 173]
X[4] <= 0.031
gini = 0.444
samples = 3
value = [2, 1]
gini = 0.0
samples = 2
value = [2, 0]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 7
value = [0, 7]
gini = 0.0
samples = 1
value = [1, 0]
X[1] <= 0.29
gini = 0.083
samples = 23
value = [1, 22]
gini = 0.0
samples = 905
value = [0, 905]
X[10] <= 13.533
gini = 0.011
samples = 3577
value = [3557, 20]
gini = 0.0
samples = 1
value = [0, 1]
X[0] <= 7.85
gini = 0.095
samples = 40
value = [38, 2]
X[1] <= 0.515
gini = 0.484
samples = 17
value = [10, 7]
X[3] <= 5.95
gini = 0.444
samples = 3
value = [2, 1]
gini = 0.0
samples = 116
value = [116, 0]
gini = 0.0
samples = 2
value = [0, 2]
gini = 0.0
samples = 3
value = [1, 2]
X[7] <= 0.994
gini = 0.219
samples = 24
value = [3, 21]
gini = 0.0
samples = 21
value = [0, 21]
X[9] <= 0.445
gini = 0.034
samples = 176
value = [3, 173]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 22
value = [0, 22]
X[9] <= 0.785
gini = 0.009
samples = 3555
value = [3539, 16]
X[6] <= 91.0
gini = 0.298
samples = 22
value = [18, 4]
X[10] <= 10.75
gini = 0.051
samples = 38
value = [37, 1]
X[6] <= 148.0
gini = 0.5
samples = 2
value = [1, 1]
X[9] <= 0.585
gini = 0.355
samples = 13
value = [10, 3]
gini = 0.0
samples = 4
value = [0, 4]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 2
value = [2, 0]
X[4] <= 0.013
gini = 0.006
samples = 3492
value = [3481, 11]
X[6] <= 79.5
gini = 0.146
samples = 63
value = [58, 5]
X[4] <= 0.041
gini = 0.32
samples = 5
value = [1, 4]
gini = 0.0
samples = 17
value = [17, 0]
gini = 0.0
samples = 34
value = [34, 0]
X[2] <= 0.28
gini = 0.375
samples = 4
value = [3, 1]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 1
value = [1, 0]
X[5] <= 67.0
gini = 0.165
samples = 11
value = [10, 1]
gini = 0.0
samples = 2
value = [0, 2]
X[13] <= 1.85
gini = 0.444
samples = 3
value = [2, 1]
X[2] <= 0.093
gini = 0.006
samples = 3489
value = [3479, 10]
X[7] <= 0.994
gini = 0.32
samples = 5
value = [1, 4]
X[0] <= 5.25
gini = 0.034
samples = 58
value = [57, 1]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 4
value = [0, 4]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 4
value = [0, 4]
gini = 0.0
samples = 10
value = [10, 0]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 1
value = [1, 0]
X[10] <= 12.7
gini = 0.092
samples = 62
value = [59, 3]
X[6] <= 275.0
gini = 0.004
samples = 3427
value = [3420, 7]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 4
value = [0, 4]
X[0] <= 4.65
gini = 0.5
samples = 2
value = [1, 1]
gini = 0.0
samples = 56
value = [56, 0]
X[4] <= 0.049
gini = 0.033
samples = 60
value = [59, 1]
gini = 0.0
samples = 2
value = [0, 2]
X[6] <= 72.5
gini = 0.004
samples = 3419
value = [3413, 6]
X[6] <= 280.0
gini = 0.219
samples = 8
value = [7, 1]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 48
value = [48, 0]
X[5] <= 37.5
gini = 0.153
samples = 12
value = [11, 1]
X[7] <= 0.995
gini = 0.102
samples = 37
value = [35, 2]
X[8] <= 2.895
gini = 0.002
samples = 3382
value = [3378, 4]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 7
value = [7, 0]
gini = 0.0
samples = 11
value = [11, 0]
gini = 0.0
samples = 1
value = [0, 1]
gini = 0.0
samples = 35
value = [35, 0]
gini = 0.0
samples = 2
value = [0, 2]
X[3] <= 0.05
gini = 0.097
samples = 39
value = [37, 2]
X[9] <= 0.755
gini = 0.001
samples = 3343
value = [3341, 2]
X[1] <= 0.15
gini = 0.444
samples = 3
value = [1, 2]
gini = 0.0
samples = 36
value = [36, 0]
X[8] <= 3.435
gini = 0.001
samples = 3289
value = [3288, 1]
X[4] <= 0.06
gini = 0.044
samples = 44
value = [43, 1]
gini = 0.0
samples = 1
value = [1, 0]
gini = 0.0
samples = 2
value = [0, 2]
gini = 0.0
samples = 3095
value = [3095, 0]
X[6] <= 92.0
gini = 0.01
samples = 204
value = [203, 1]
gini = 0.0
samples = 43
value = [43, 0]
gini = 0.0
samples = 1
value = [0, 1]
X[3] <= 6.9
gini = 0.117
samples = 16
value = [15, 1]
gini = 0.0
samples = 188
value = [188, 0]
gini = 0.0
samples = 15
value = [15, 0]
gini = 0.0
samples = 1
value = [0, 1]

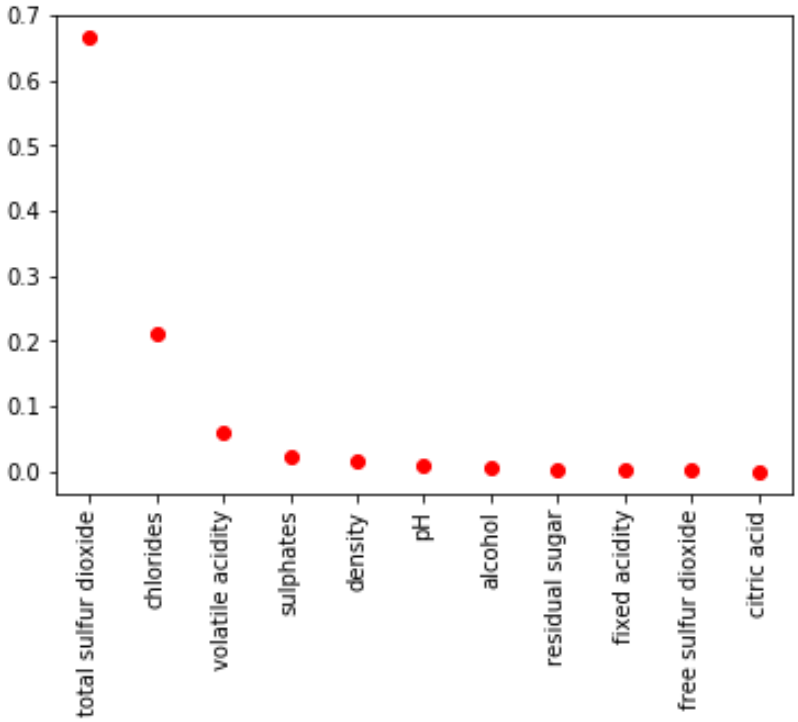


Fig. 1: Feature importance values of decision tree classifier

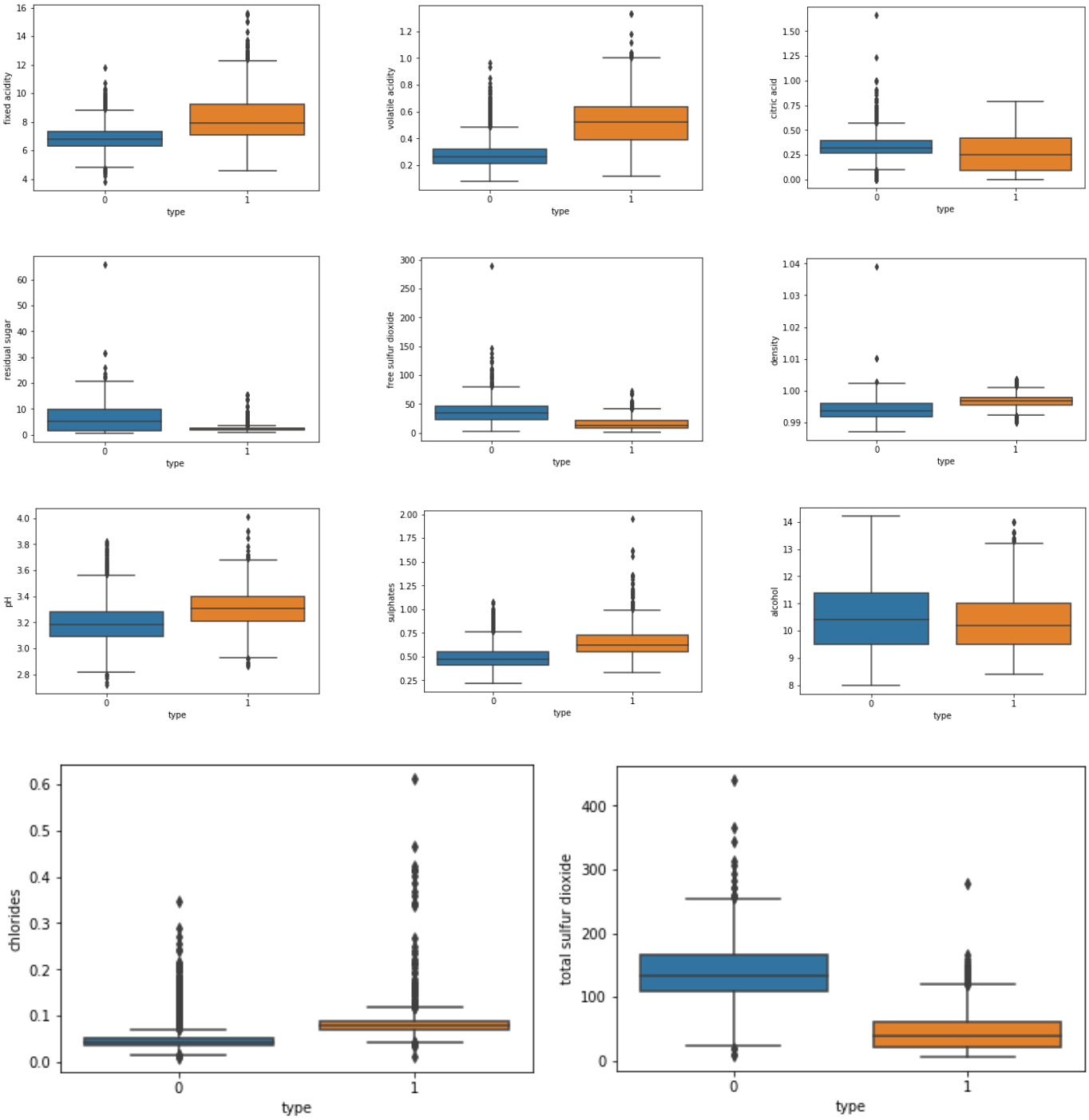


Fig. 2: Box Plots of features for each wine type