Tomasz Załuska

Khanh Do Van

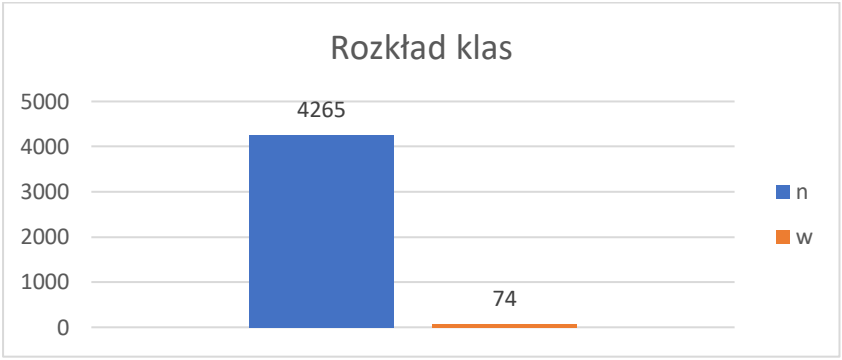
PSZT – projekt 2

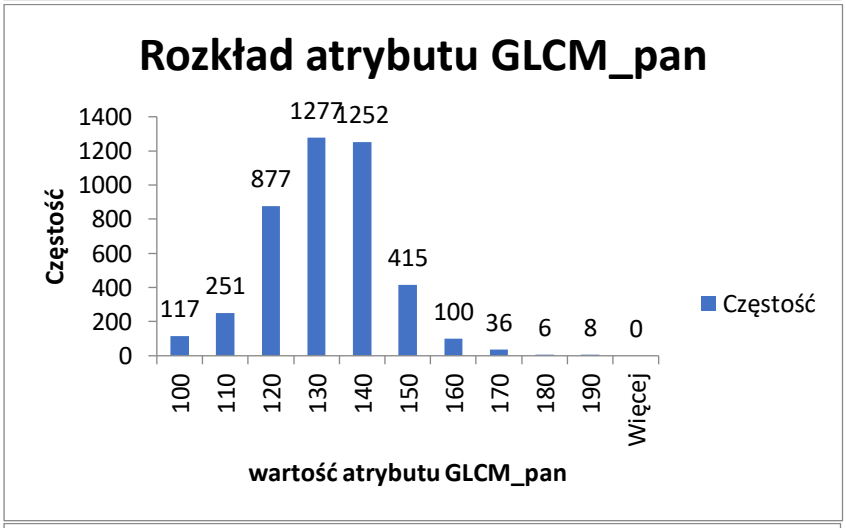
Choroby drzew

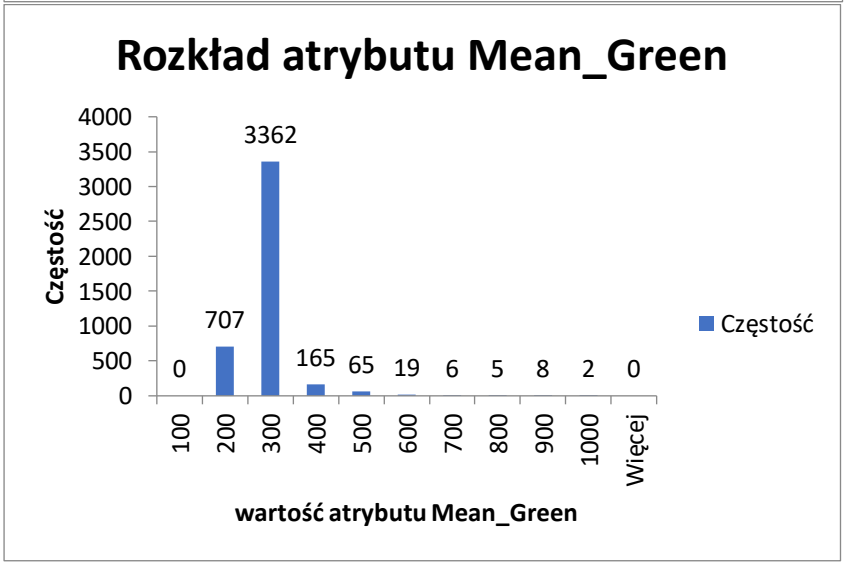
Random Forest

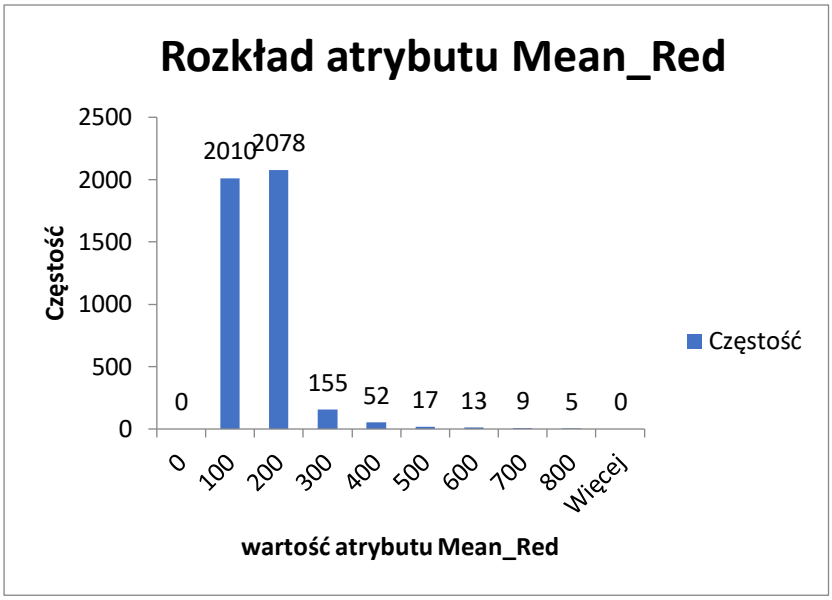
1. Przedstawienie danych treningowy oraz testowych

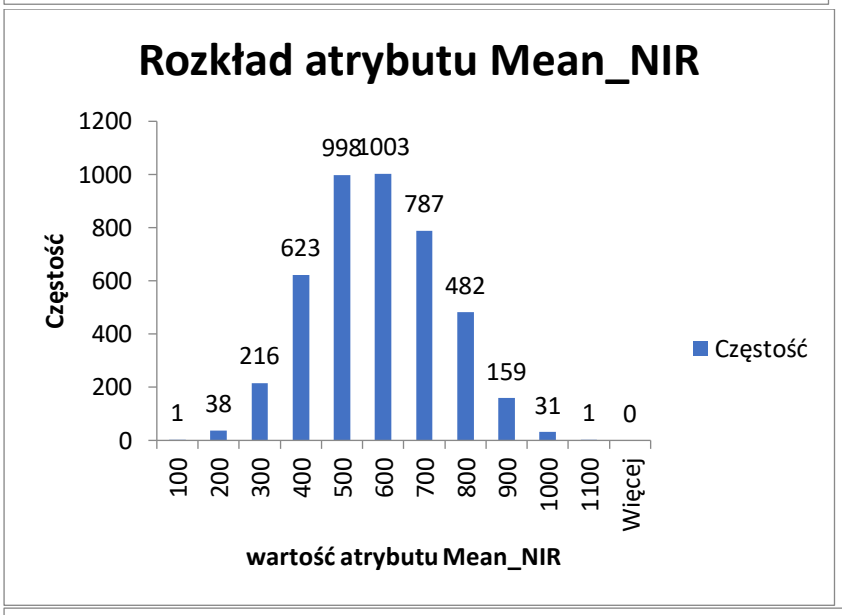
Histogramy dla atrybutów w danych treningowych:

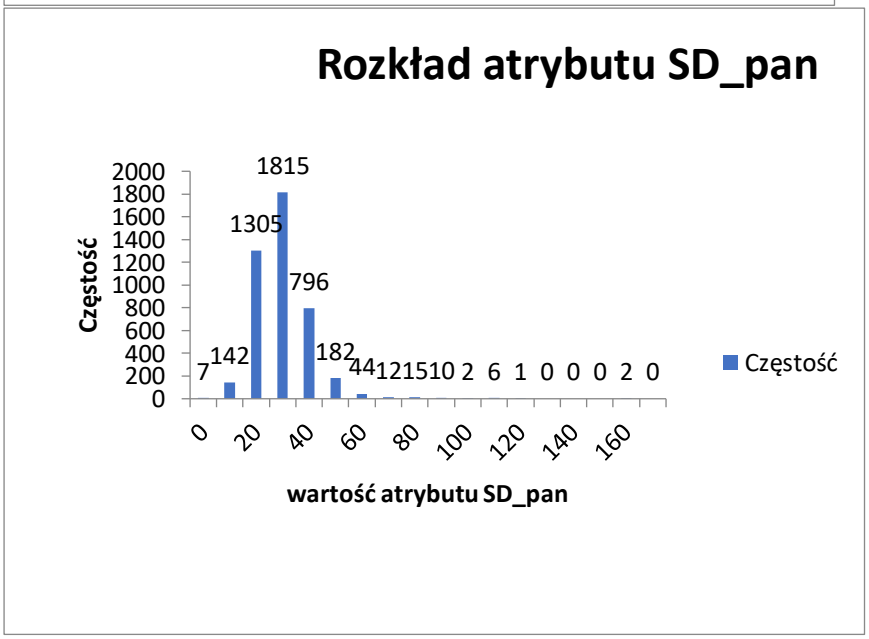




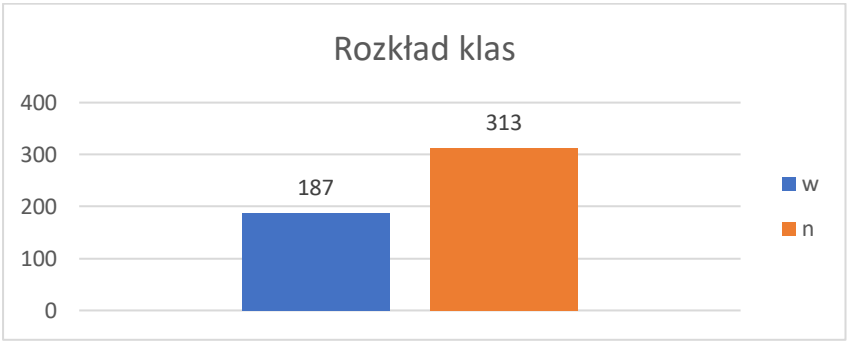


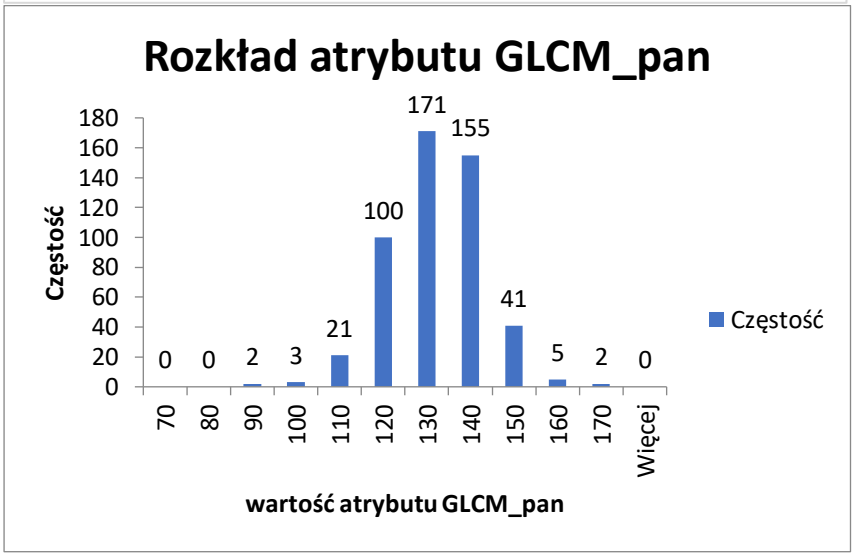

Rozkład atrybutu Mean_Red
Częstość
2500
2000
1500
1000
500
0
0
2010
2078
155
52
17
13
9
5
0
0
100
200
300
400
500
600
700
800
Więcej
wartość atrybutu Mean_Red
Częstość

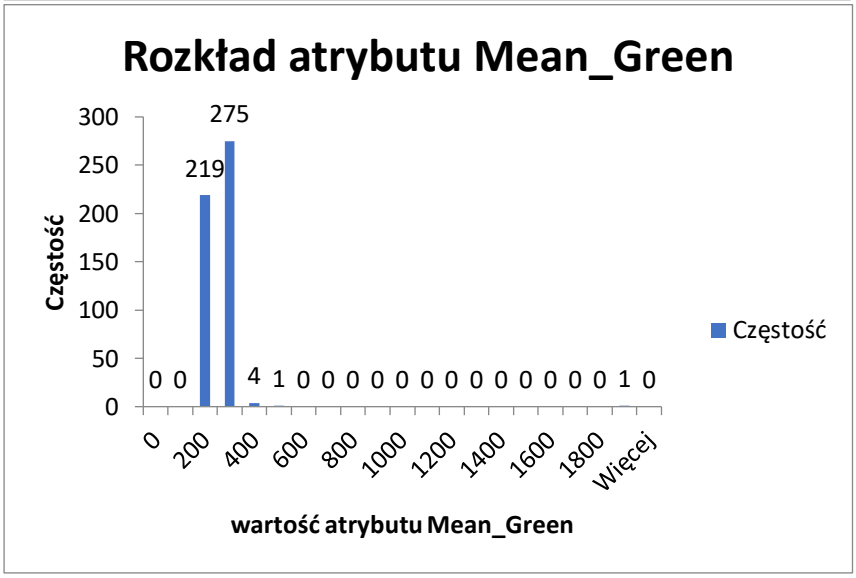

Rozkład atrybutu Mean_NIR
Częstość
1200
1000
800
600
400
200
0
1
38
216
623
998
1003
787
482
159
31
1
0
100
200
300
400
500
600
700
800
900
1000
1100
Więcej
wartość atrybutu Mean_NIR
Częstość

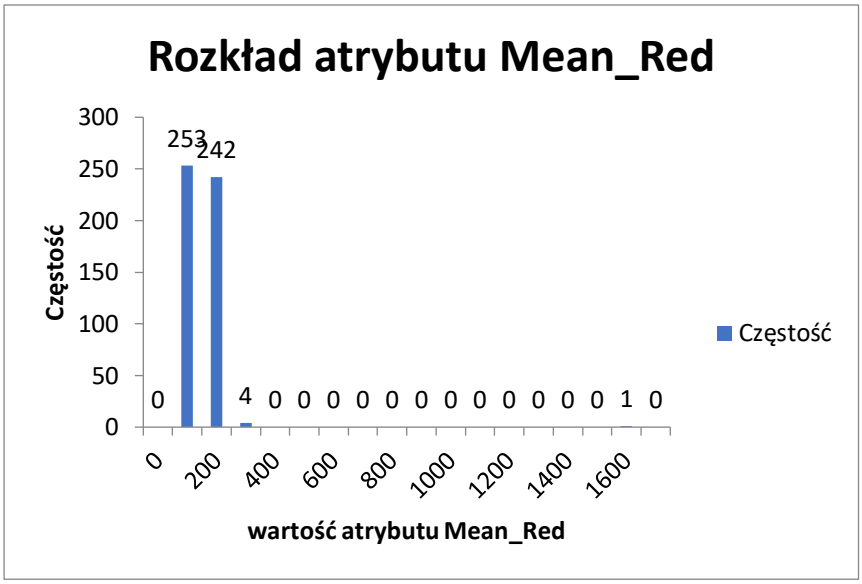

Rozkład atrybutu SD_pan
Częstość
2000
1800
1600
1400
1200
1000
800
600
400
200
0
7
142
1305
1815
796
182
44
12
15
10
2
6
1
0
0
0
2
0
0
20
40
60
80
100
120
140
160
wartość atrybutu SD_pan
Częstość

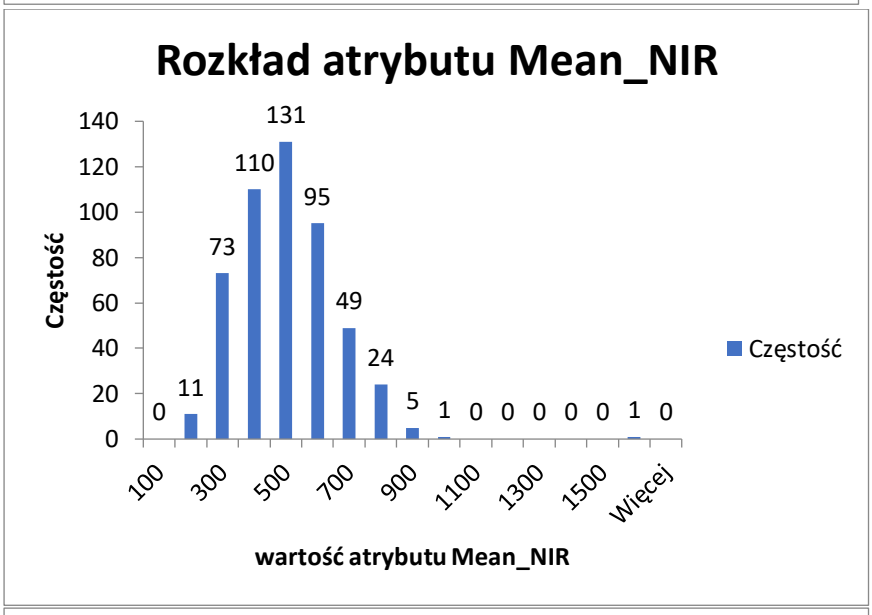
Histogramy dla atrybutów w danych testowych:

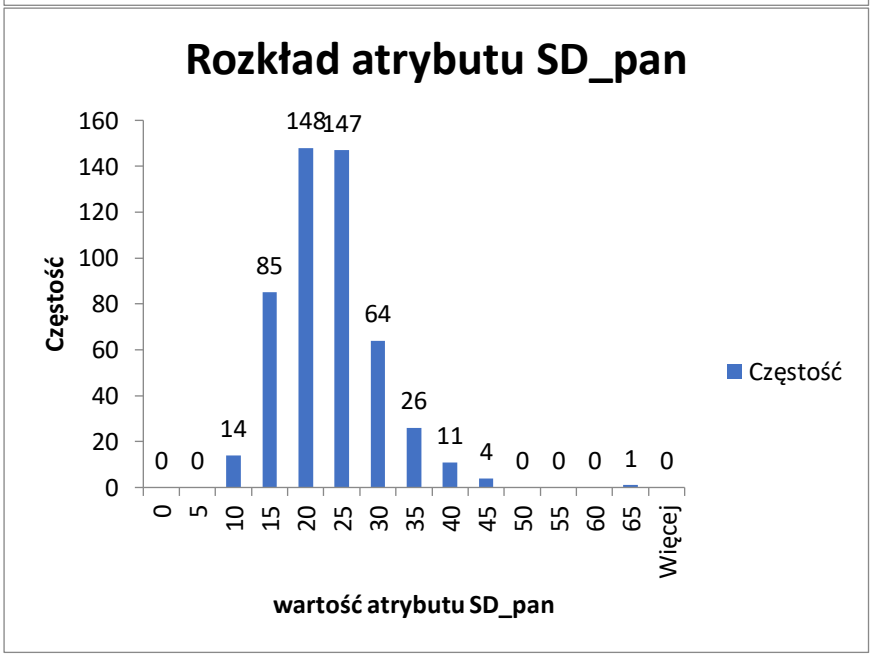







Rozkład atrybutu Mean_Red
Częstość
300
250
200
150
100
50
0
253
242
4
0 0 0 0 0 0 0 0 0 0 0 0 1 0
0
200
400
600
800
1000
1200
1400
1600
wartość atrybutu Mean_Red
Częstość


Rozkład atrybutu Mean_NIR
Częstość
140
120
100
80
60
40
20
0
131
110
95
73
49
24
11
5
0
1 0 0 0 0 0 1 0
100
300
500
700
900
1100
1300
1500
Więcej
wartość atrybutu Mean_NIR
Częstość


Rozkład atrybutu SD_pan
Częstość
160
140
120
100
80
60
40
20
0
148
147
85
64
26
14
11
4
0 0 0 0 0 1 0
0
5
10
15
20
25
30
35
40
45
50
55
60
65
Więcej
wartość atrybutu SD_pan
Częstość

2. Preprocesing danych.

   Jedynym preprocesingiem danych w przypadku użycia do modelu lasu losowego było zmiana wartości klas z znaków(‘w’, ‘n’) na zera i jedynki.

3. Podział danych.

   Podział na zbiór uczący i testowy został taki jak na stronie źródłowej.
   Plik training.csv zawiera dane treningowe, a plik testing.csv dane testowe.

4. Użyte biblioteki:
   -pandas wersja 0.25.3
   -numpy wersja 1.18.1
   -matplotlib wersja 3.1.2
   -scikit-learn wersja 0.22.1

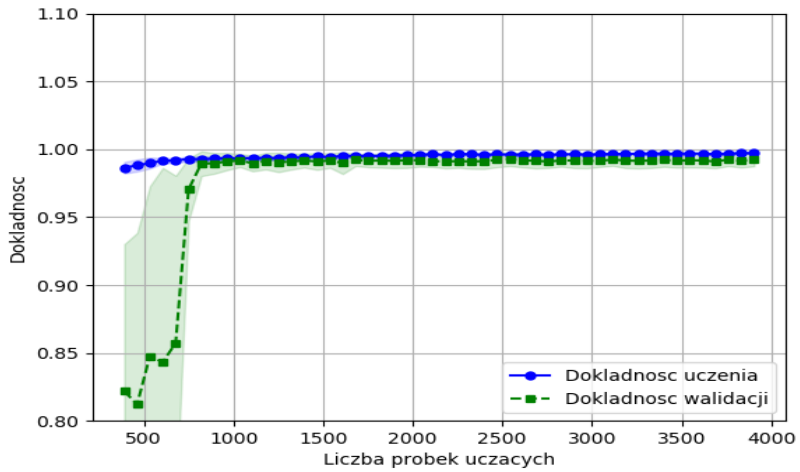
5. Strojenie parametrów.
   Strojone parametry:
   - liczba drzew decyzyjny
   - maksymalna głębokość drzewa decyzyjnego
   - minimalna liczba obiektów potrzebna do stworzenia węzła w drzewie decyzyjnym
   - minimalna liczba obiektów potrzebna do stworzenia liścia

   Do strojenia parametrów został użyty algorytm przeszukiwania losowego. Podczas procesu szukania parametrów zbiór training.csv był dzielony przy użyciu k-krotnej walidacji krzyżowej do określenia dokładności modelu z wygenerowanymi parametrami.

   Przykładowy wynik wywołania algorytmu strojenia:

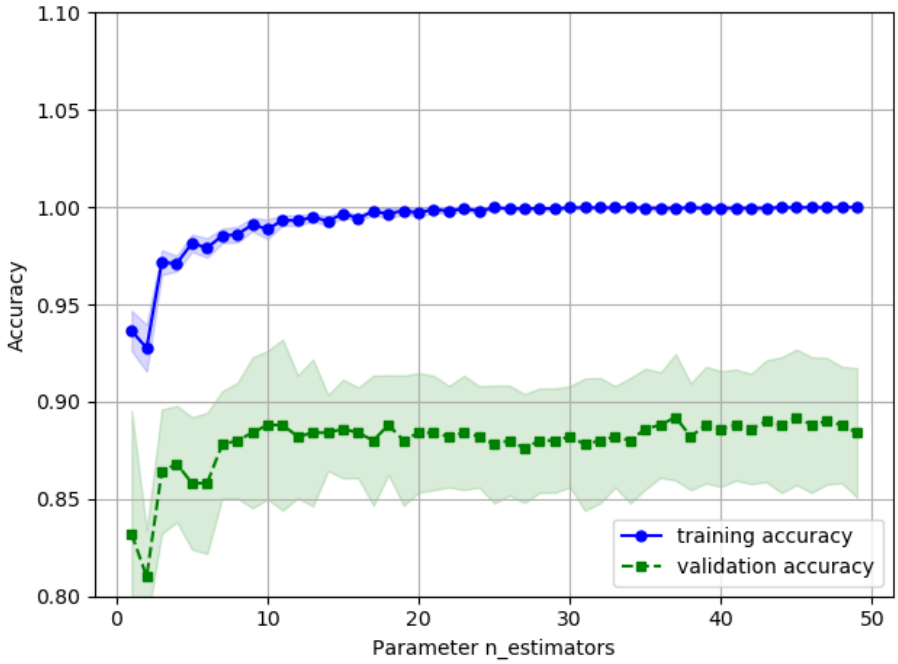
```
Najlepsza dokladnosc znaleziona przeszukiwaniem losowym: 0.9924
Parametry dla najlepszego wyniku:
{'n_estimators': 35, 'min_samples_split': 6, 'min_samples_leaf': 3, 'max_depth': 38}
Dokladnosc na zbiorze testowym: 0.7560
```

   Krzywa uczenia dla podanych wyżej parametrów:

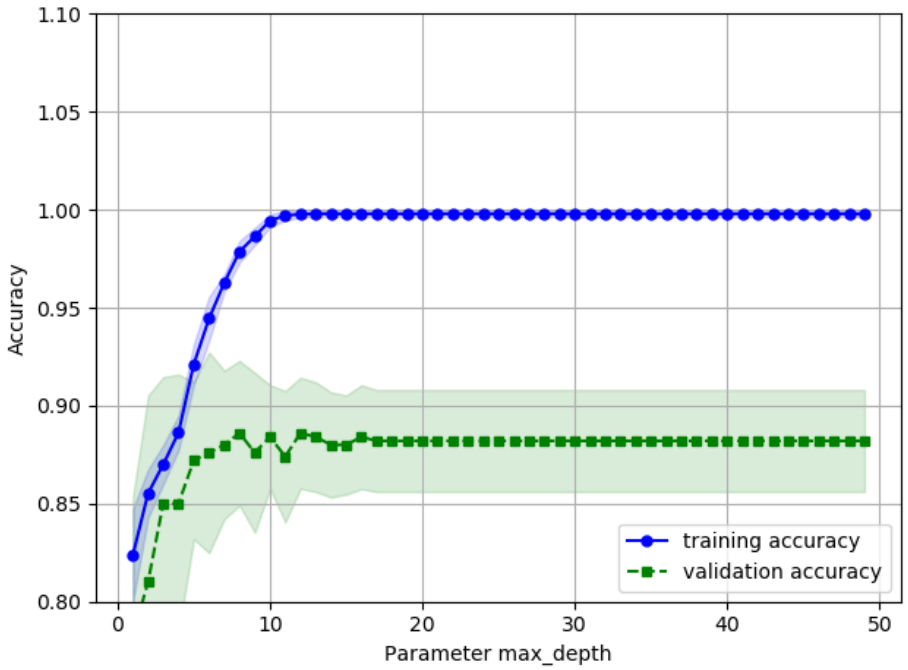


6. Wpływ poszczególnych parametrów na krzywą walidacji.
   Testy są przeprowadzane na zbiorze testing.csv.
   Zmienia się jeden parametr reszta ma wartość domyślną(parametr n_estimatros ma wartość 22, ponieważ jest to przybliżenie pierwiastka z liczby obiektów zbioru treningowego).

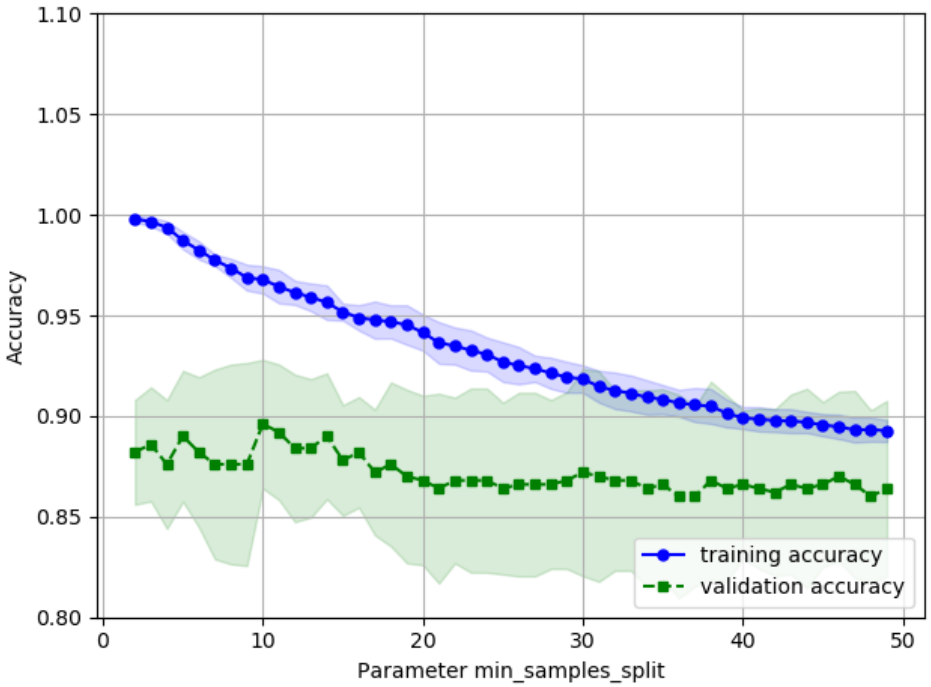
   a) liczba drzew decyzyjny (n_estimators)



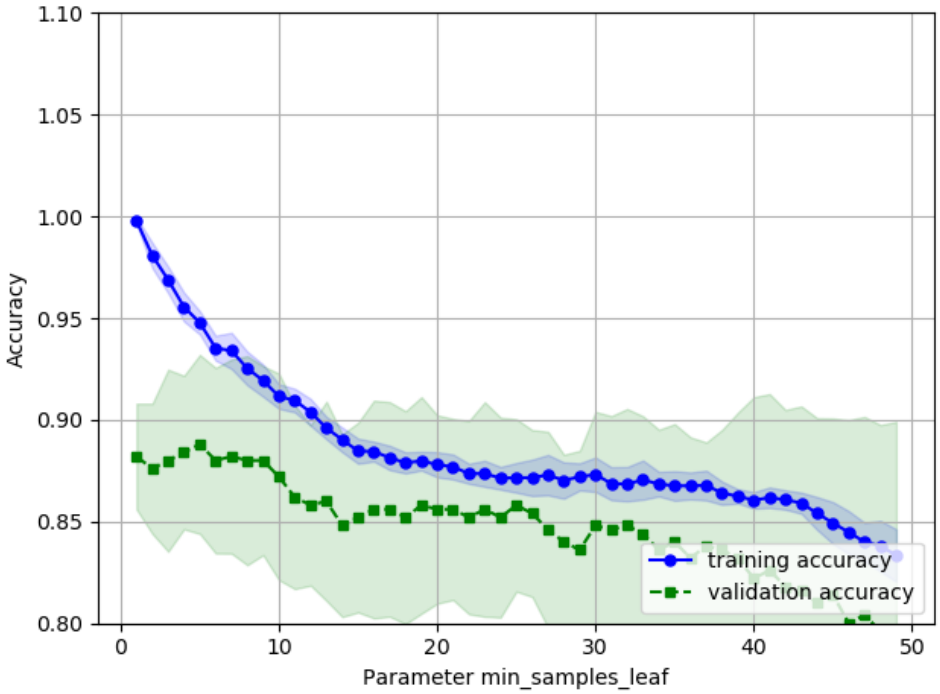
   b) maksymalna głębokość drzewa decyzyjnego (max_depth)



   c) minimalna liczba obiektów potrzebna do stworzenia węzła w drzewie decyzyjnym (min_samples_split)



d) minimalna liczba obiektów potrzebna do stworzenia liścia (min_samples_leaf)



7. Wnioski.

Z przeprowadzonych testów wynika, że największy wpływ na dokładność klasyfikacji maja parametry odnoszące się do liczby drzew decyzyjnych w lesie losowym oraz maksymalna głębokość pojedynczego drzewa decyzyjnego. Parametry min_samples_split oraz min_samples_leaf mogą znacznie pogorszyć dokładność klasyfikacji jeśli przyjmą wartość nieco powyżej 2. Jak widać na pierwszym wykresie funkcja przyjmuje wartość maksymalną nieco wcześniej niż teoretyczna optymalna wartość tego parametru, czyli pierwiastek kwadratowy z liczby obiektów, która w tym przypadku wynosiła 500.

Logistic regression

1. The description of data set

From the online source provided on the project portal, we download 2 files named “training.csv” and “testing.csv”.

Load the data set

	class	GLCM_pan	Mean_Green	Mean_Red	Mean_NIR	SD_pan
0	w	120.362774	205.500000	119.395349	416.581395	20.676318
1	w	124.739583	202.800000	115.333333	354.333333	16.707151
2	w	134.691964	199.285714	116.857143	477.857143	22.496712
3	w	127.946309	178.368421	92.368421	278.473684	14.977453
4	w	135.431548	197.000000	112.690476	532.952381	17.604193
5	w	118.347962	226.150000	138.850000	608.900000	29.072797
6	w	135.436282	184.500000	95.142857	309.190476	13.055264
7	w	121.169643	226.000000	146.214286	595.571429	22.808542
8	w	131.127161	232.784314	144.588235	563.843137	11.948563
9	w	134.498092	210.212121	116.909091	594.848485	27.937685

(4339, 6)

Training set head

	class	GLCM_pan	Mean_Green	Mean_Red	Mean_NIR	SD_pan
0	n	109.828571	183.700000	82.950000	251.750000	16.079412
1	n	130.284483	212.637931	96.896552	482.396552	21.210295
2	n	131.386555	185.466667	85.466667	419.666667	13.339998
3	n	141.345098	180.875000	81.500000	348.062500	18.213577
4	w	121.383408	218.357143	112.017857	426.607143	19.083196
5	n	122.757576	205.960000	86.760000	407.680000	17.823580
6	w	124.010204	215.594595	117.027027	477.297297	24.574628
7	w	125.608407	209.649123	121.228070	443.280702	25.757314
8	n	107.745833	204.133333	83.466667	479.866667	22.956965
9	n	140.203922	188.593750	84.437500	457.250000	30.193759

(500, 6)

Testing set head

Training data set has 6 columns included: **class, GLCM_pan, Mean_Green, Mean_Red, Mean_NIR, SD_pan**. There are in total **4.339** rows of data.

Testing data set has also 6 columns with the same name as training set. And there are in total **500** rows of data.

The distribution of data

The distribution in some **two-feature spaces** of training data and testing data. And the distributions are showed in the form of comparison between these two sets.

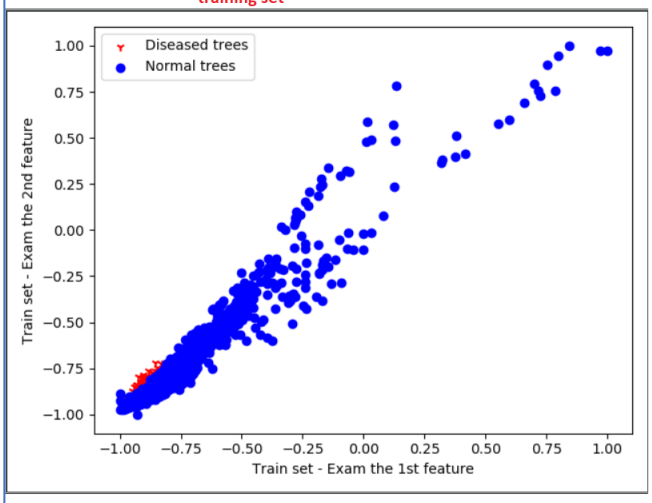
The correlation between training set and testing set with some pairs of dimensions in which setting them apart

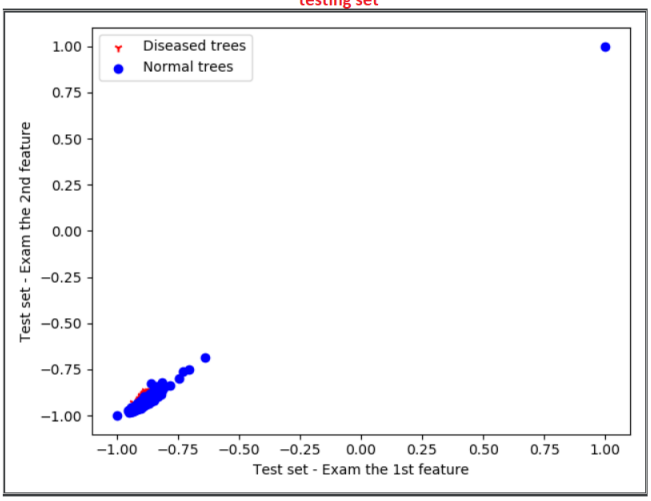
Feature based 0

training set

Distribution of samples in 2-dimension space of feauture 1 and 2

testing set

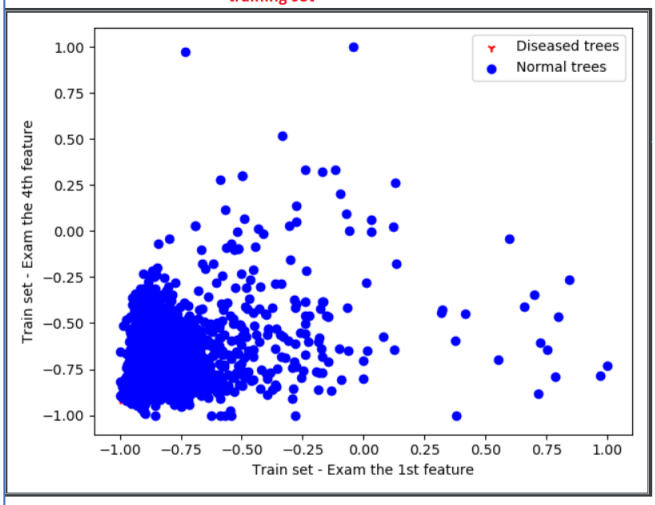


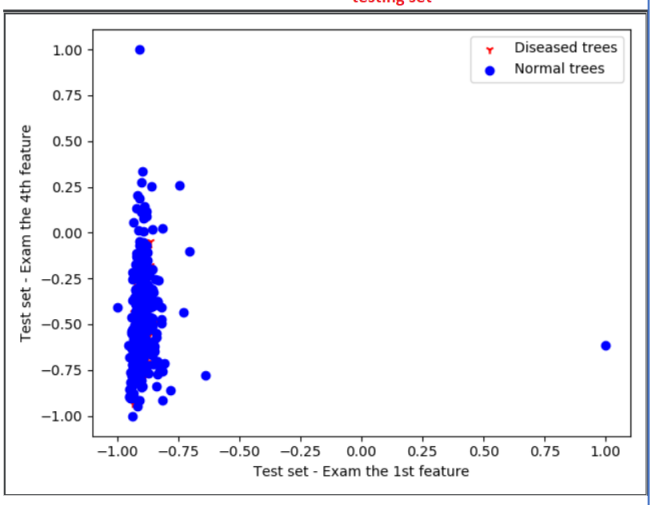


Distribution of samples in 2-dimension space of feauture 1 and 4

training set

testing set

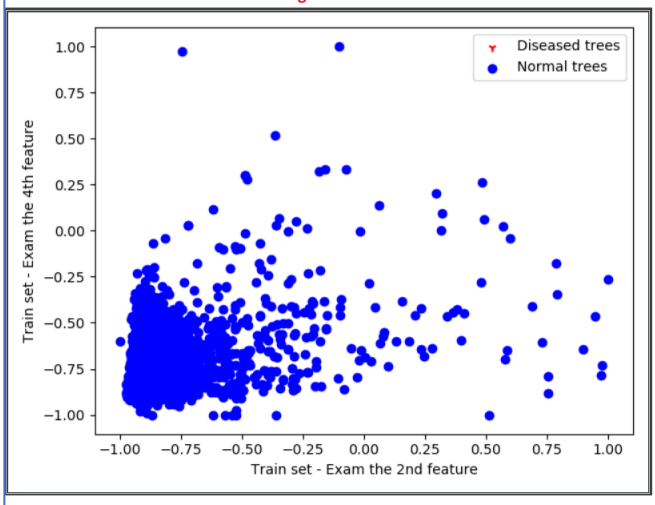


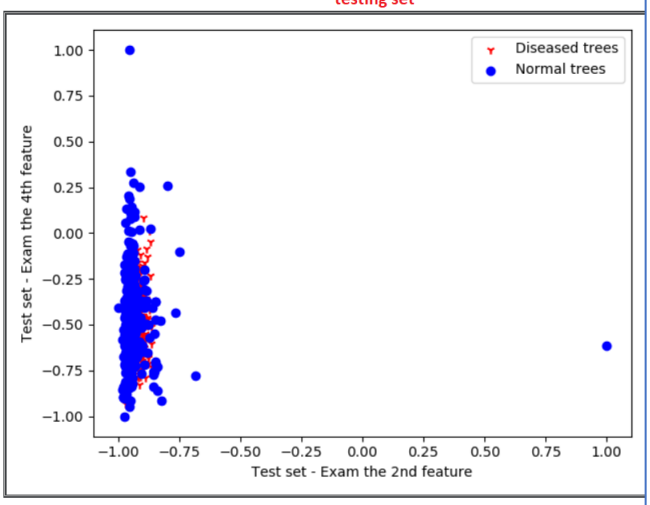


Distribution of samples in 2-dimension space of feauture 2 and 4

training set

testing set





2. Preprocessing data

Encoding data

We can see that the first column named “class” has values with type of character (‘w’ or ‘n’). The data that feeds Logistic Regression algorithm for training must be the form of numeric. So we have to encode values of this column into numberic (Type of Float number with two possible values are 0.0 and 1.0)

	class	GLCM_pan	Mean_Green	Mean_Red	Mean_NIR	SD_pan
0	1.0	120.362774	205.500000	119.395349	416.581395	20.676318
1	1.0	124.739583	202.800000	115.333333	354.333333	16.707151
2	1.0	134.691964	199.285714	116.857143	477.857143	22.496712
3	1.0	127.946309	178.368421	92.368421	278.473684	14.977453
4	1.0	135.431548	197.000000	112.690476	532.952381	17.604193
5	1.0	118.347962	226.150000	138.850000	608.900000	29.072797
6	1.0	135.436282	184.500000	95.142857	309.190476	13.055264
7	1.0	121.169643	226.000000	146.214286	595.571429	22.808542
8	1.0	131.127161	232.784314	144.588235	563.843137	11.948563
9	1.0	134.498092	210.212121	116.909091	594.848485	27.937685

(4339, 6)

Encoded training set

	class	GLCM_pan	Mean_Green	Mean_Red	Mean_NIR	SD_pan
0	0.0	109.828571	183.700000	82.950000	251.750000	16.079412
1	0.0	130.284483	212.637931	96.896552	482.396552	21.210295
2	0.0	131.386555	185.466667	85.466667	419.666667	13.339998
3	0.0	141.345098	180.875000	81.500000	348.062500	18.213577
4	1.0	121.383408	218.357143	112.017857	426.607143	19.083196
5	0.0	122.757576	205.960000	86.760000	407.680000	17.823580
6	1.0	124.010204	215.594595	117.027027	477.297297	24.574628
7	1.0	125.608407	209.649123	121.228070	443.280702	25.757314
8	0.0	107.745833	204.133333	83.466667	479.866667	22.956965
9	0.0	140.203922	188.593750	84.437500	457.250000	30.193759

(500, 6)

Encoded testing set

Features Scaling

Logistic Regression requires the training and testing data to be scaling usually into [-1,1] scale. It helps the algorithm function and speed up the computational process.

	class	GLCM_pan	Mean_Green	Mean_Red	Mean_NIR	SD_pan
0	1.0	0.313422	-0.896661	-0.824641	-0.281664	-0.735780
1	1.0	0.361182	-0.903488	-0.836463	-0.417130	-0.786502
2	1.0	0.469784	-0.912372	-0.832028	-0.148313	-0.712518
3	1.0	0.396175	-0.965254	-0.903300	-0.582219	-0.808605
4	1.0	0.477855	-0.918151	-0.844155	-0.028412	-0.775038
5	1.0	0.291436	-0.844455	-0.768020	0.136868	-0.628483
6	1.0	0.477907	-0.949753	-0.895226	-0.515372	-0.833169
7	1.0	0.322226	-0.844834	-0.746587	0.107862	-0.708533
8	1.0	0.430885	-0.827682	-0.751320	0.038813	-0.847311
9	1.0	0.467669	-0.884748	-0.831877	0.106288	-0.642988

(4339, 6)

Scaled training set

	class	GLCM_pan	Mean_Green	Mean_Red	Mean_NIR	SD_pan
0	0.0	-0.338776	-0.923209	-0.958069	-0.852837	-0.635950
1	0.0	0.132453	-0.889788	-0.940003	-0.535242	-0.454724
2	0.0	0.157841	-0.921169	-0.954809	-0.621619	-0.732708
3	0.0	0.387249	-0.926472	-0.959947	-0.720217	-0.560570
4	1.0	-0.072595	-0.883183	-0.920416	-0.612063	-0.529855
5	0.0	-0.040939	-0.897501	-0.953133	-0.638125	-0.574345
6	1.0	-0.012083	-0.886373	-0.913928	-0.542263	-0.335894
7	1.0	0.024734	-0.893240	-0.908486	-0.589103	-0.294121
8	0.0	-0.386754	-0.899610	-0.957399	-0.538725	-0.393031
9	0.0	0.360961	-0.917557	-0.956142	-0.569868	-0.137423

(500, 6)

Scaled testing set

3. Splitting data into training and testing sets

During project implementation, we used training set to fit the model and then, test the accuracy of the model by using testing set.

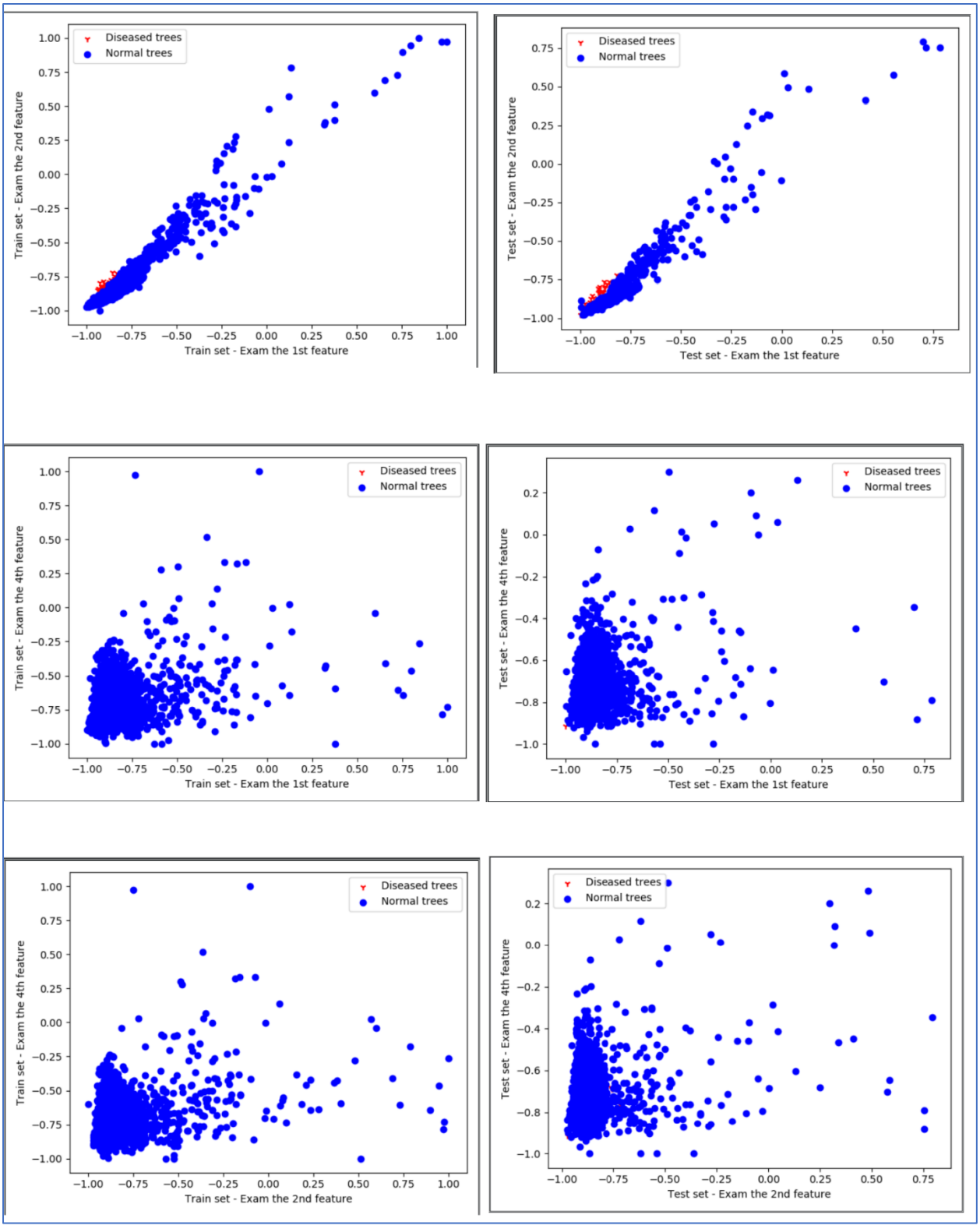
And **the result was at very low of accuracy**, approximately around **61.6%** in which the model classify correctly.

We also used the approach of tuning parameters for optimization (using random seach algorithm) but not improved.

We revised all steps conducted from preprocessing data to training and testing. And we noticed that the distribution of these two sets in the same space is quite different (at **1. The description of data set – The distribution of data**).

This situation leads us to use only training set data (training.csv) and split it into two sets (training and testing set) by randomly picking approach.

The result we got in the form of statistical charts as below:



4. Building Classifier and Test

We use the random search method to tuning parameters of the algorithm to get optimal parameters.

Parameters which we need to tune included **inverse of regularization strength *alpha*** and **number of iterations *iterations.***

```
Best: 0.980393 using {'max_iter': 850, 'C': 0.2}
score Scikit learn tuning:  0.9881284916201117
```

The accuracy we got before tuning is **0.988**

And we can get the optimal classifier with **max_iter=850**, and **C=0.2**. There will be another options because they are local optimization.