**Covid-19 Exploratory Data Analysis**

**Author:**  Muhammad Asad.

**Importing The Libararies:**

**import** numpy **as** np **import** matplotlib.pyplot **as** plt **import** pandas **as** pd **import** seaborn **as** sns

**Reading the Dataset csv file by using Pandas:**

**df = pd.read\_csv('covid.csv') df.head(10)**

AGE WBC Platelets Neutrophils Lymphocytes Monocytes Eosinophils \

1. 56 2.9 128.0 1.9 0.8 0.2 0.0
2. 56 3.5 151.0 2.1 0.9 0.4 0.0
3. 72 4.6 206.0 NaN NaN NaN NaN
4. 72 16.5 316.0 14.0 1.2 0.3 0.0
5. 77 4.9 198.0 NaN NaN NaN NaN
6. 77 3.0 162.0 2.0 0.4 0.4 0.1
7. 74 5.3 189.0 3.3 1.3 0.7 0.0
8. 74 5.2 144.0 4.2 0.6 0.4 0.0
9. 75 11.6 123.0 10.0 0.8 0.7 0.0 9 75 7.2 26.0 6.2 0.6 0.3 0.1

Basophils CRP AST ALT ALP GGT LDH SWAB

1. 0.0 29.0 36.0 18.0 43.0 21.0 257.0 1
2. 0.0 16.5 25.0 14.0 50.0 17.0 207.0 1
3. NaN 193.7 31.0 22.0 NaN NaN NaN 1
4. 0.0 318.7 96.0 33.0 80.0 42.0 651.0 1
5. NaN NaN NaN NaN NaN NaN NaN 1
6. NaN NaN 21.0 10.0 NaN NaN 220.0 1
7. 0.0 4.6 16.0 13.0 NaN 27.0 117.0 1
8. 0.0 104.0 91.0 131.0 NaN NaN 391.0 1
9. 0.0 244.6 53.0 37.0 43.0 27.0 439.0 1
10. 0.0 248.2 85.0 NaN 130.0 NaN 453.0 1

**df.info()**

<class 'pandas.core.frame.DataFrame'> RangeIndex: 279 entries, 0 to 278 Data columns (total 15 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

1. AGE 279 non-null int64
2. WBC 277 non-null float64
3. Platelets 277 non-null float64
4. Neutrophils 209 non-null float64
5. Lymphocytes 209 non-null float64
6. Monocytes 209 non-null float64
7. Eosinophils 209 non-null float64
8. Basophils 208 non-null float64
9. CRP 273 non-null float64
10. AST 277 non-null float64
11. ALT 266 non-null float64
12. ALP 131 non-null float64
13. GGT 136 non-null float64
14. LDH 194 non-null float64 14 SWAB 279 non-null int64 dtypes: float64(13), int64(2) memory usage: 32.8 KB

We have a total of 279 observations and 16 columns in out dataset. First column represents gender of the patient and second column gives the age of the patients. Columns 3 to 15 are enteries of different blood tests, whereas last column represents the result of covid-19 nasal SWAB test for that particular patient, considering it to be the actual result.

**Data Cleaning:**

We can notice from above that this dataset has a very high number of missing values. Let's first count them.

**df.isnull().sum()**

[AGE 0 WBC 2](#_Toc14710)

[Platelets 2](#_Toc14711)

[Neutrophils 70 Lymphocytes 70 Monocytes 70 Eosinophils 70 Basophils 71 CRP 6](#_Toc14712)

AST 2

ALT 13

ALP 148

GGT 143

LDH 85 SWAB 0 dtype: int64

**Checking the percentage of missing values:**

**df.isnull().sum()/len(df)\*100**

AGE 0.000000

WBC 0.716846

Platelets 0.716846

Neutrophils 25.089606

Lymphocytes 25.089606

Monocytes 25.089606

Eosinophils 25.089606

Basophils 25.448029

CRP 2.150538

AST 0.716846

ALT 4.659498

ALP 53.046595

GGT 51.254480

LDH 30.465950

SWAB 0.000000 dtype: float64

**Dealing with missing values:**

ALP and GGT have a very high percentage of missing values so it would be better to simply remove these columns.

**df.drop(columns=['ALP','GGT'], inplace=True) df.isnull().sum()**

AGE 0

WBC 2

# Platelets 2

# Neutrophils 70

Lymphocytes 70

Monocytes 70

Eosinophils 70

Basophils 71

CRP 6

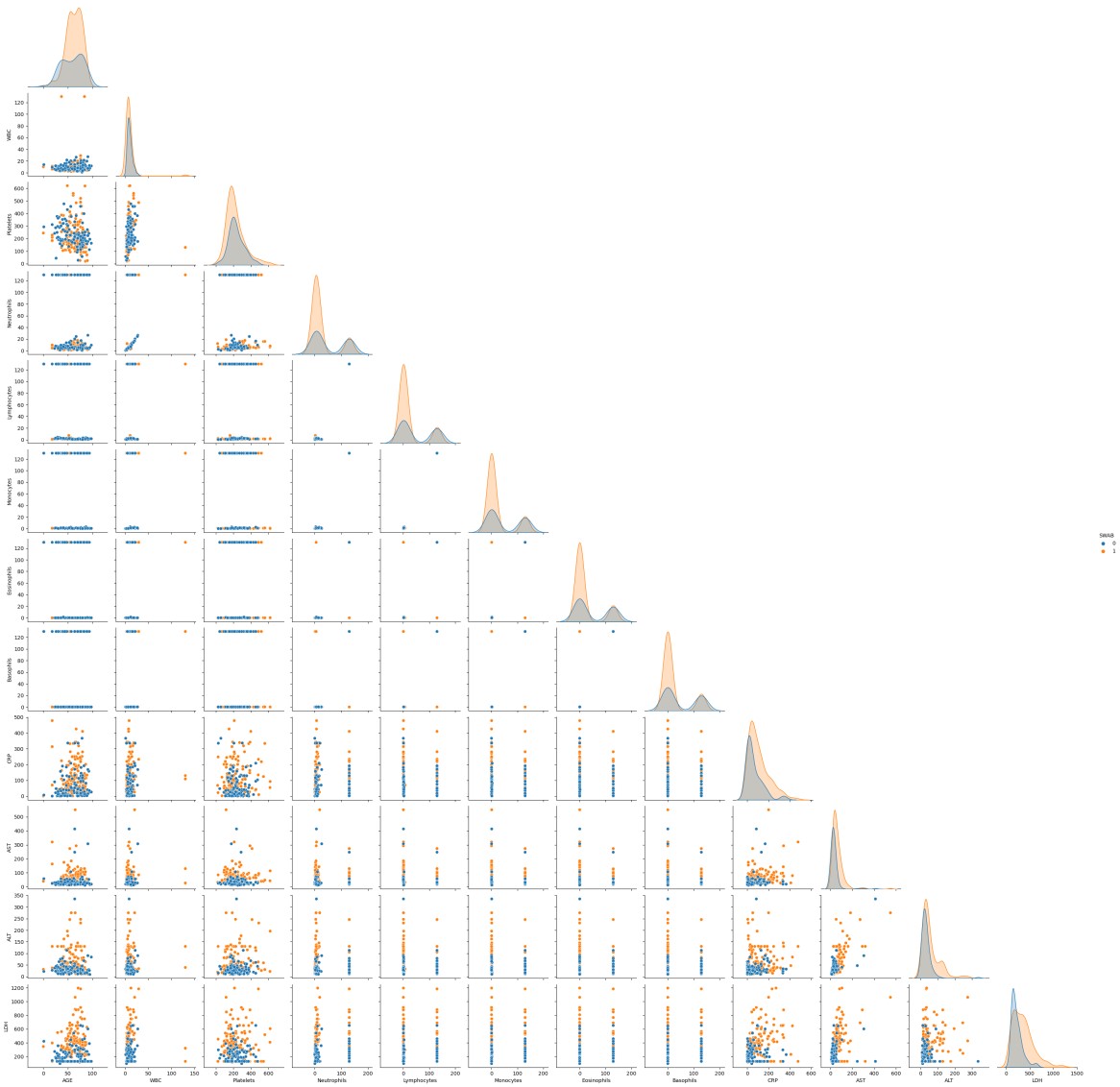
AST 2

ALT 13

LDH 85

SWAB 0 dtype: int64 **sns.pairplot(df, hue='SWAB', corner=True)**

<seaborn.axisgrid.PairGrid at 0x2e7acbe9990>



The increase of white blood cells is a sign that an infection has occurred. We can clearly visualize the direct relation between WBC and Neutrophils. Moreover we can also observe high values of ALT in covid-19 infected patients.

**Replacing missing values with 130:**

**df["Neutrophils"].fillna(130, inplace = True) df["Lymphocytes"].fillna(130, inplace = True) df["Monocytes"].fillna(130, inplace = True) df["Eosinophils"].fillna(130, inplace = True) df["Basophils"].fillna(130, inplace = True) df["LDH"].fillna(130, inplace = True) df["CRP"].fillna(130, inplace = True) df["AST"].fillna(130, inplace = True) df["ALT"].fillna(130, inplace = True) df["WBC"].fillna(130, inplace = True) df["Platelets"].fillna(130, inplace = True) print(df)**

AGE WBC Platelets Neutrophils Lymphocytes Monocytes Eosinophils

\

1. 56 2.9 128.0 1.9 0.8 0.2 0.0
2. 56 3.5 151.0 2.1 0.9 0.4 0.0
3. 72 4.6 206.0 130.0 130.0 130.0 130.0
4. 72 16.5 316.0 14.0 1.2 0.3 0.0
5. 77 4.9 198.0 130.0 130.0 130.0 130.0 .. ... ... ... ... ... ... ...
6. 40 5.6 270.0 130.0 130.0 130.0 130.0
7. 26 4.7 44.0 130.0 130.0 130.0 130.0
8. 44 15.7 356.0 130.0 130.0 130.0 130.0
9. 41 12.0 321.0 6.7 2.3 1.6 1.3 278 53 17.9 272.0 14.1 2.1 1.5 0.2

Basophils CRP AST ALT LDH SWAB

1. 0.0 29.0 36.0 18.0 257.0 1
2. 0.0 16.5 25.0 14.0 207.0 1
3. 130.0 193.7 31.0 22.0 130.0 1
4. 0.0 318.7 96.0 33.0 651.0 1
5. 130.0 130.0 130.0 130.0 130.0 1

.. ... ... ... ... ... ...

1. 130.0 16.7 22.0 29.0 130.0 0
2. 130.0 47.5 30.0 44.0 175.0 0
3. 130.0 2.4 31.0 17.0 272.0 0
4. 0.1 41.2 16.0 11.0 266.0 0
5. 0.1 123.9 13.0 10.0 258.0 0

[279 rows x 13 columns]

**X = df.drop(columns=['SWAB']) y = df['SWAB']**

**Splitting The Dataset into the training set and Test set:**

from **sklearn.model\_selection** import **train\_test\_split**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.30, random\_state = 0)**

**print(X\_train)**

AGE WBC Platelets Neutrophils Lymphocytes Monocytes Eosinophils

\

214 29 11.2 296.0 5.8 4.1 0.8 0.4

187 80 12.8 246.0 10.7 1.6 0.5 0.0

268 83 21.4 398.0 130.0 130.0 130.0 130.0

134 18 11.2 184.0 9.7 0.6 0.8 0.0

33 80 10.9 158.0 10.0 0.3 0.6 0.0 .. ... ... ... ... ... ... ...

251 77 20.4 187.0 17.3 1.0 2.1 0.0

192 87 10.4 180.0 130.0 130.0 130.0 130.0

117 37 6.9 254.0 4.2 2.2 0.4 0.0

47 65 5.7 156.0 130.0 130.0 130.0 130.0 172 46 9.4 191.0 7.3 1.3 0.7 0.0

Basophils CRP AST ALT LDH

214 0.1 2.8 19.0 23.0 182.0

187 0.0 100.3 71.0 30.0 643.0

268 130.0 192.0 29.0 68.0 308.0

134 0.0 313.3 27.0 37.0 130.0

33 0.0 28.4 26.0 36.0 245.0 .. ... ... ... ... ...

251 0.0 127.9 23.0 24.0 130.0

192 130.0 3.2 24.0 15.0 130.0

117 0.0 48.8 54.0 45.0 358.0

47 130.0 111.2 29.0 34.0 400.0

172 0.0 82.1 55.0 64.0 308.0

[195 rows x 12 columns]

**print(y\_train)**

214 0

187 0

268 0

134 1

33 1 ..

251 0

192 0

117 1

47 1

172 1

Name: SWAB, Length: 195, dtype: int64

**print(X\_test)**

AGE WBC Platelets Neutrophils Lymphocytes Monocytes Eosinophils

\

220 79 1.1 57.0 0.5 0.4 0.1 0.1

227 42 12.3 476.0 9.6 1.5 1.0 0.1

252 81 17.0 147.0 130.0 130.0 130.0 130.0

15 18 8.6 210.0 7.0 0.9 0.4 0.2

55 55 2.6 94.0 130.0 130.0 130.0 130.0 .. ... ... ... ... ... ... ...

219 90 7.7 271.0 4.3 2.3 0.6 0.5

217 87 11.0 182.0 4.3 3.3 3.2 0.1

153 60 4.3 188.0 1.8 2.0 0.5 0.0

71 54 10.7 375.0 8.9 1.5 0.2 0.0 206 33 11.3 288.0 6.9 3.6 0.6 0.1

Basophils CRP AST ALT LDH

220 0.0 366.2 19.0 32.0 228.0

227 0.0 103.1 27.0 44.0 130.0

252 130.0 94.8 24.0 30.0 130.0

15 0.1 478.0 320.0 130.0 130.0

55 130.0 27.2 42.0 23.0 274.0 .. ... ... ... ... ...

219 0.0 40.1 25.0 14.0 272.0

217 0.1 81.8 32.0 14.0 274.0

153 0.0 1.3 55.0 51.0 130.0

71 0.0 332.3 98.0 78.0 651.0

206 0.1 1.1 16.0 18.0 184.0

[84 rows x 12 columns]

**print(y\_test)**

220 0

227 0

252 0

15 1

55 1 ..

219 0

217 0

153 1

71 1

206 0

Name: SWAB, Length: 84, dtype: int64

**Feature Scaling:**

from **sklearn.preprocessing** import **StandardScaler**

**sc = StandardScaler()**

**X\_train = sc.fit\_transform(X\_train)**

**X\_test = sc.transform(X\_test)**

**print(X\_train)**

[[-1.67519681e+00 1.13699737e-01 7.30118323e-01 ... -6.10803116e-01

-5.42119232e-01 -6.26555458e-01]

[ 9.97217737e-01 2.35814429e-01 2.26413863e-01 ... 2.69086584e-01

-4.01489670e-01 1.55689170e+00]

[ 1.15441859e+00 8.92180897e-01 1.75767542e+00 ... -4.41593558e-01 3.61927948e-01 -2.97781436e-02] ...

[-1.25599453e+00 -2.14483497e-01 3.07006577e-01 ... -1.85696643e-02 -1.00140611e-01 2.07038251e-01]

[ 2.11213458e-01 -3.06069516e-01 -6.80254165e-01 ... -4.41593558e-01

-3.21129921e-01 4.05964022e-01]

[-7.84391963e-01 -2.36792912e-02 -3.27661043e-01 ... -1.64870851e-03

2.81568198e-01 -2.97781436e-02]] **print(X\_test)**

[[ 9.44817451e-01 -6.57149254e-01 -1.67758900e+00 ... -6.10803116e-01

-3.61309796e-01 -4.08684375e-01]

[-9.93993104e-01 1.97653588e-01 2.54345438e+00 ... -4.75435470e-01

-1.20230548e-01 -8.72844508e-01]

[ 1.04961802e+00 5.56365494e-01 -7.70920968e-01 ... -5.26198337e-01 -4.01489670e-01 -8.72844508e-01] ...

[-5.07879688e-02 -4.12919871e-01 -3.57883310e-01 ... -1.64870851e-03

2.03990133e-02 -8.72844508e-01]

[-3.65189680e-01 7.55388958e-02 1.52597137e+00 ... 7.25952389e-01

5.62827321e-01 1.59478232e+00]

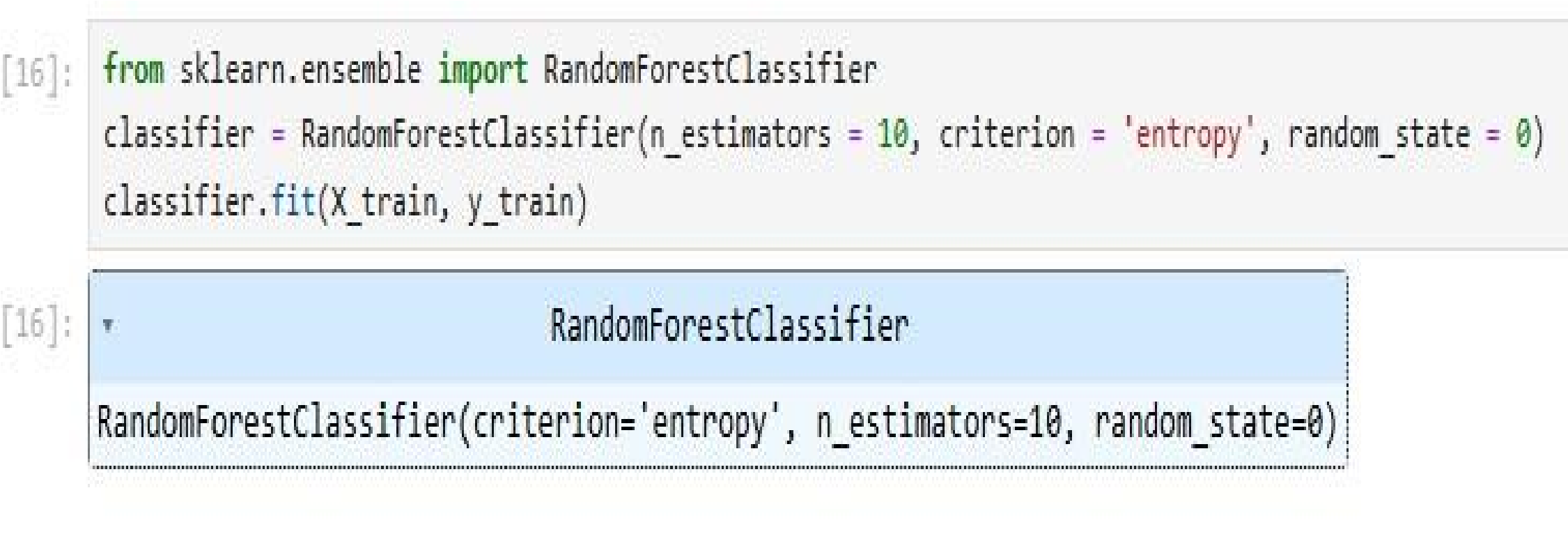
[-1.46559567e+00 1.21331905e-01 6.49525610e-01 ... -6.61565983e-01 -6.42568918e-01 -6.17082802e-01]]

**Training The Random Forest Classification Model on the Training set:**

**Predicting the Test set results:**

**y\_pred = classifier.predict(X\_test)**

**Making The Confusion Matrix:**

from **sklearn.metrics** import **confusion\_matrix, accuracy\_score cm = confusion\_matrix(y\_test, y\_pred) print(cm) accuracy\_score(y\_test, y\_pred)**

[[20 14]

[ 7 43]]

0.75

Survival ratio is 75%.

**Visualizing The Training set results:**

**from** matplotlib.colors **import** ListedColormap X\_set, y\_set = sc.inverse\_transform(X\_train), y\_train

X1, X2 = np.meshgrid(np.arange(start = X\_set[:, 0].min() - 10, stop = X\_set[:, 0].max() + 10, step = 0.25),

np.arange(start = X\_set[:, 1].min() - 1000, stop =

X\_set[:, 1].max() + 1000, step = 0.25))

*#plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(),*

*X2.ravel()]).T)).reshape(X1.shape),*

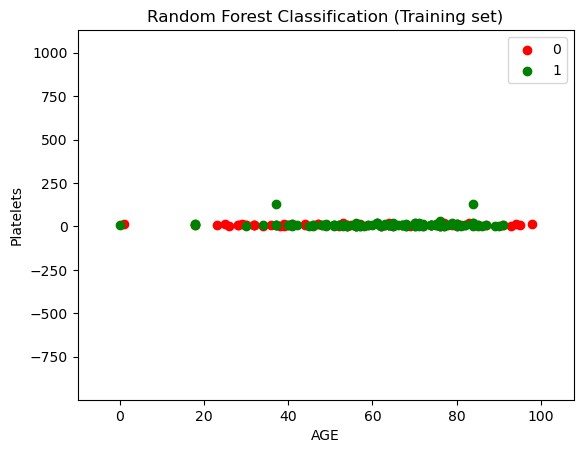
*#alpha = 0.75, cmap = ListedColormap(('red', 'green')))* plt.xlim(X1.min(), X1.max()) plt.ylim(X2.min(), X2.max()) **for** i, j **in** enumerate(np.unique(y\_set)):

plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1], c = ListedColormap(('red', 'green'))(i), label = j) plt.title('Random Forest Classification (Training set)') plt.xlabel('AGE') plt.ylabel('Platelets')

plt.legend() plt.show()

plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1], c =

ListedColormap(('red', 'green'))(i), label = j)



**Visualizing The Test set Results:**

**from** matplotlib.colors **import** ListedColormap X\_set, y\_set = sc.inverse\_transform(X\_test), y\_test

X1, X2 = np.meshgrid(np.arange(start = X\_set[:, 0].min() - 10, stop = X\_set[:, 0].max() + 10, step = 0.25),

np.arange(start = X\_set[:, 1].min() - 1000, stop =

X\_set[:, 1].max() + 1000, step = 0.25))

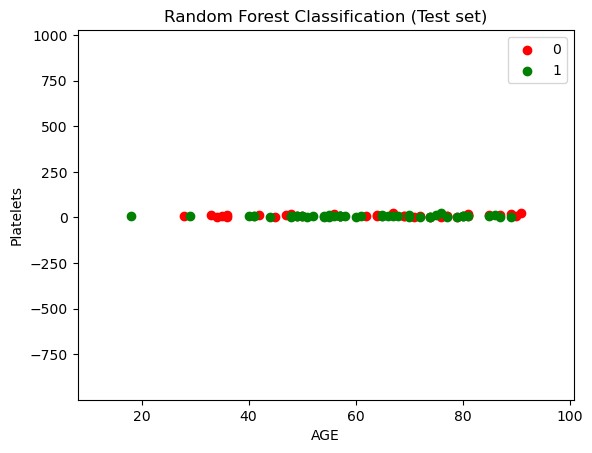
*#plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(),*

*X2.ravel()]).T)).reshape(X1.shape),*

*#alpha = 0.75, cmap = ListedColormap(('red', 'green')))* plt.xlim(X1.min(), X1.max()) plt.ylim(X2.min(), X2.max()) **for** i, j **in** enumerate(np.unique(y\_set)): plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1], c = ListedColormap(('red', 'green'))(i), label = j) plt.title('Random Forest Classification (Test set)') plt.xlabel('AGE') plt.ylabel('Platelets') plt.legend() plt.show()

plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1], c =

ListedColormap(('red', 'green'))(i), label = j)



**Reference:**

This Dataset has been taken from [here](https://zenodo.org/records/3886927)

[Detection of COVID-19 Infection from Routine Blood Exams with Machine Learning: a Feasibility Study (zenodo.org)](https://zenodo.org/records/3886927)