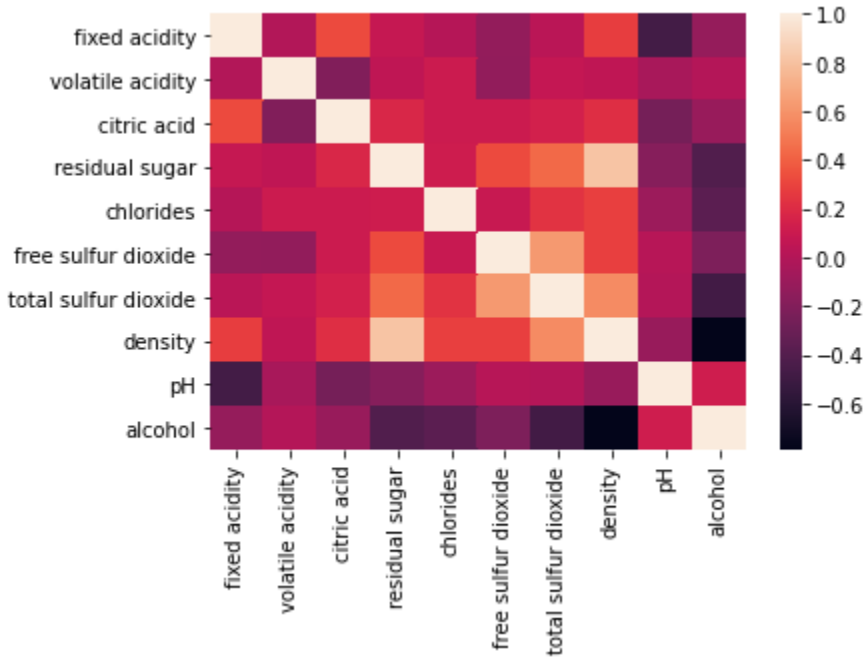
POSWAL, Abhishek
20557578

[Q1] Report the number of remaining records after duplicate removal and paste the screenshot of the heatmap.

```
Number of records in train.csv: (1617, 11)
Number of records in test.csv: (400, 11)
```

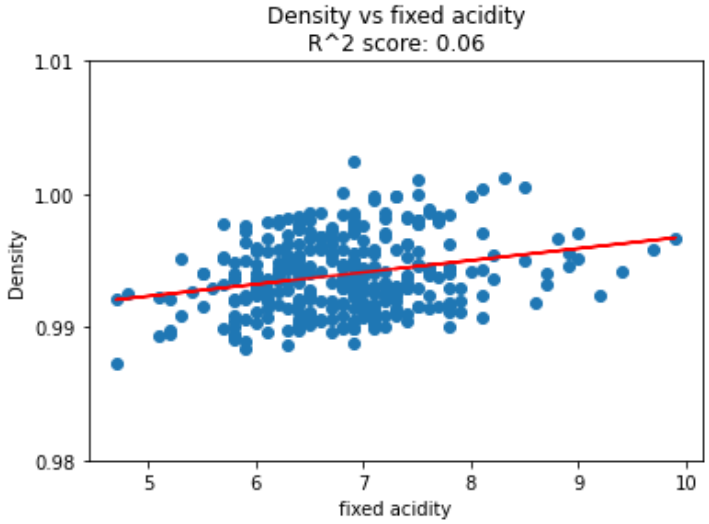
Heatmap



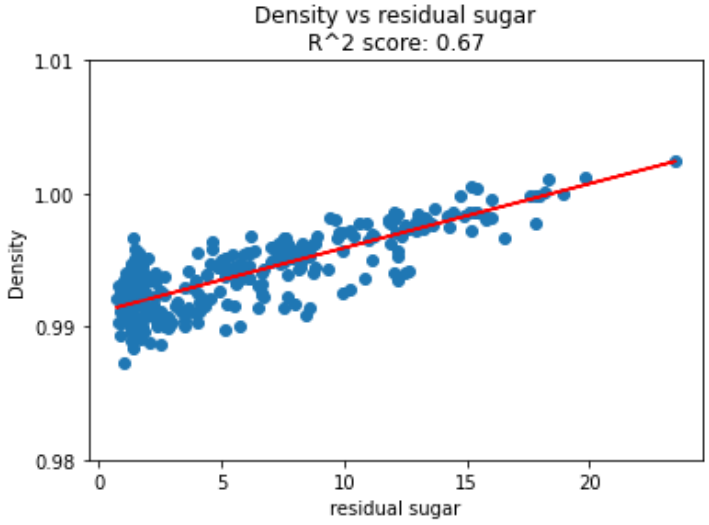
[Q2] Report the validation R2 score of each model to evaluate the relationship between different features and 'density'.

1. Density vs fixed acidity 0.06121630722910132
2. Density vs residual sugar 0.6657111313550805
3. Density vs chlorides 0.08292629933478712
4. Density vs free sulfur dioxide 0.09776154943324744
5. Density vs total sulfur dioxide 0.3058437254204954
6. Density vs alcohol 0.6242802278261513
7. Density vs all 6 models 0.9333038833878456

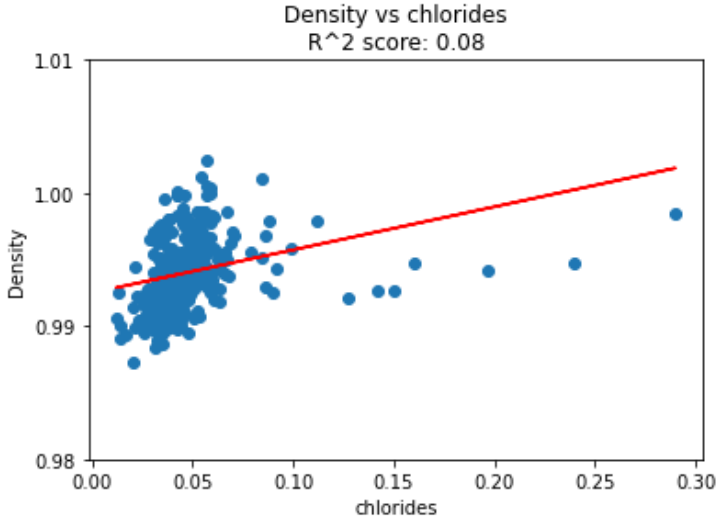
[Q3] After training the models with the training set, use them to make prediction on the validation set. Then, plot the regression line and the data points of the validation set for each of the first six models. For illustration, the figure below shows a plot of the feature 'density' versus the feature 'total sulfur dioxide' and the regression line.



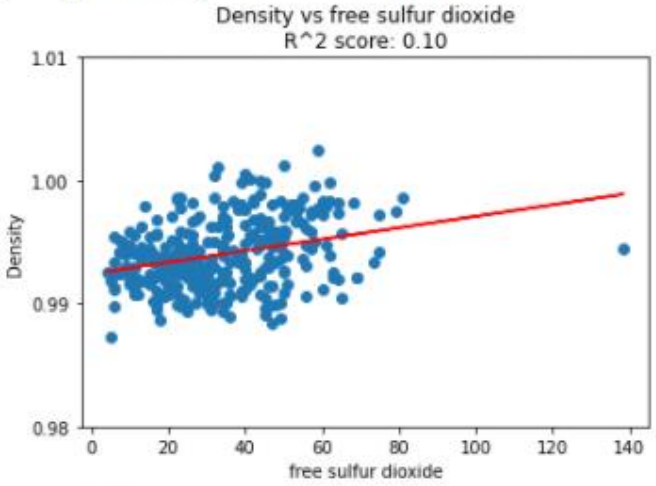

```
y = ax + b
y =  [0.00089539] * x +  0.9878513992648529
```



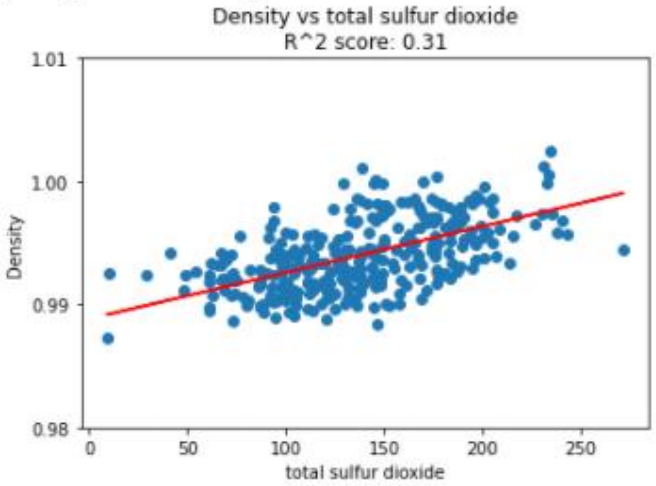

```
y = ax + b
y =  [0.00048185] * x +  0.9911011079618991
```



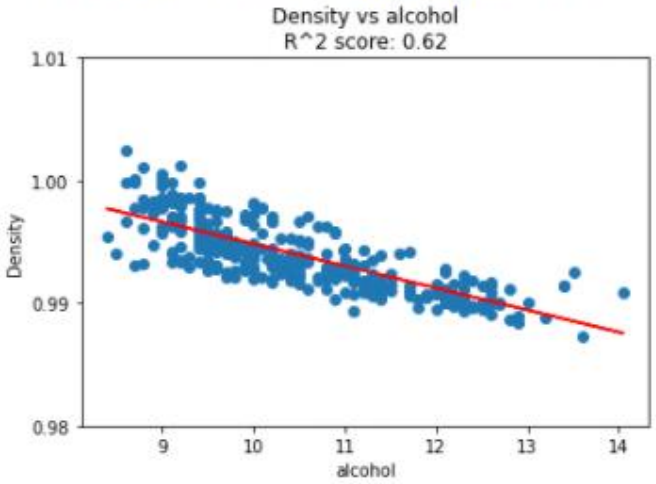
```
y = ax + b
y =  [0.03232686] * x +  0.9924982923778399
```




```
y = ax + b
y =  [4.66458508e-05] * x +  0.9924143005036553
```




```
y = ax + b
y =  [3.75791777e-05] * x +  0.9888192706105808
```




```
y = ax + b
y =  [-0.00179973] * x +  1.0128284132620933
```

[Q4] Report the score for each of the 10 features.

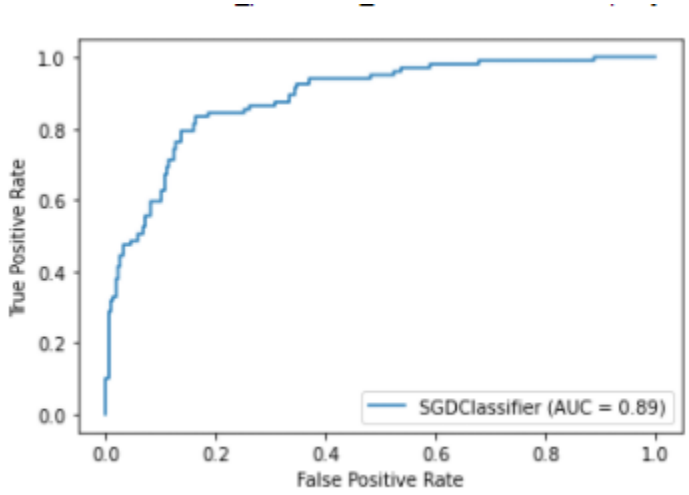
```
chi_squared = [3.23131414e+00, 2.25262908e+00, 2.54079137e-02, 1.9
8105064e+02, 1.16010041e+00, 2.30504295e+00, 1.31569579e+03, 2.407
65966e-03, 2.92731881e-01, 7.05697853e+01]

p-value = [7.22426954e-002, 1.33387609e-001, 8.73354842e-001, 5.41
201533e-045, 2.81444717e-001, 1.28954716e-001, 4.38856455e-288, 9.
60865178e-001, 5.88475172e-001, 4.44269522e-017]
```

[Q5] Report the model setting, training time, and performance of the logistic regression model. Since the solution found may depend on the initial weight values, you are expected to repeat each setting three times and report the corresponding mean and standard deviation of the training time, accuracy, and F1 score for each setting.

```
Model Setting:
Classifier: SGD
Loss: Logistic
eta0: 0.99
power_t: 0.53
Learning Rate: Invscaling
Penalty: L2
random_state: [1010, 3632, 4211]
max_iter: 1000
verbose: 1
    clf_sgd3 = SGDClassifier(loss='log', eta0 = 0.99, power_t = 0.53, learning_rate = 'invscaling', penalty='l2',
                             max_iter=1000, random_state=rand, verbose=1)

Run 3 times.

Accuracy:
        Mean:  0.8405349794238682 Standart Deviation:  0.0029099044493273167
Total Time:
        Mean:  0.0036570231119791665 Standart Deviation:  0.00047047122097760647
F1 Score:
        Mean:  0.7454354643539083 Standart Deviation:  0.006523927056779405
```

[Q6] Plot the ROC curve calculated on the validation set with the last model in [Q5] and report the AUC value.2 Give one advantage of the ROC curve for model evaluation

AUC value: 0.89



[Q7] Report the model setting, training time, and performance of the neural networks for each value of H. You are also expected to repeat each setting three times for the same hyperparameter setting and report the mean and standard deviation of the training time, accuracy, and F1 score for each setting.

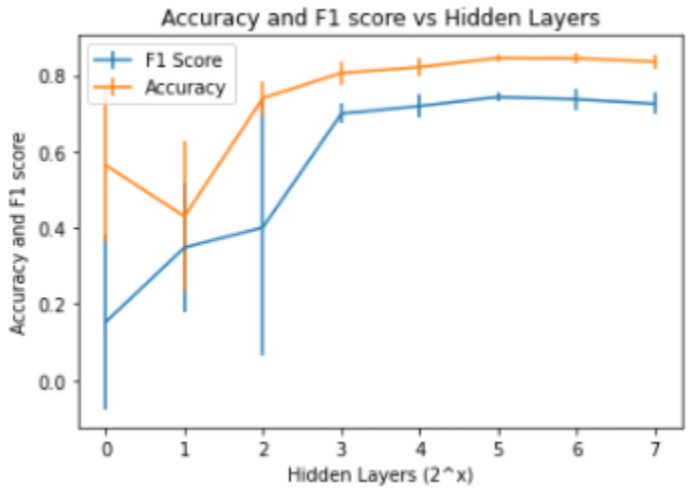
```
Number of hidden layers: 1
Accuracy:
Mean:  0.5637860082304527 Standard Deviation:  0.18699957069050313
Total Time:
Mean:  0.028590281804402668 Standard Deviation:  0.005235625182907372
F1 Score:
Mean:  0.1536025336500396 Standard Deviation:  0.21722678630275566


Number of hidden layers: 2
Accuracy:
Mean:  0.43004115226337447 Standard Deviation:  0.18477893253228708
Total Time:
Mean:  0.08111151059468587 Standard Deviation:  0.0780371966718157
F1 Score:
Mean:  0.348140739814699 Standard Deviation:  0.15933500304771672


Number of hidden layers: 4
Accuracy:
Mean:  0.7407407407407408 Standard Deviation:  0.037122671919071405
Total Time:
Mean:  0.07712904612223308 Standard Deviation:  0.03974276221602449
F1 Score:
Mean:  0.400935550935551 Standard Deviation:  0.28948125035785965


Number of hidden layers: 8
```

```
Accuracy:
Mean:  0.8065843621399177 Standard Deviation:  0.025863796481433214
Total Time:
Mean:  0.0920867125193278 Standard Deviation:  0.04058240596620695
F1 Score:
Mean:  0.699462272831838 Standard Deviation:  0.022038216146095805


Number of hidden layers: 16
Accuracy:
Mean:  0.8220164609053499 Standard Deviation:  0.019081519538056983
Total Time:
Mean:  0.0990585486094157 Standard Deviation:  0.031475464149704604
F1 Score:
Mean:  0.719755671116514 Standard Deviation:  0.02554833021939203


Number of hidden layers: 32
Accuracy:
Mean:  0.8456790123456791 Standard Deviation:  0.0075601535271085625
Total Time:
Mean:  0.10837690035502116 Standard Deviation:  0.020622350268244924
F1 Score:
Mean:  0.7432363973933378 Standard Deviation:  0.010437802720236618


Number of hidden layers: 64
Accuracy:
Mean:  0.8446502057613169 Standard Deviation:  0.011909297225092843
Total Time:
Mean:  0.11469316482543945 Standard Deviation:  0.011015765534411324
F1 Score:
Mean:  0.7371880328572495 Standard Deviation:  0.022382919936773604


Number of hidden layers: 128
Accuracy:
Mean:  0.8364197530864198 Standard Deviation:  0.014031051128586285
Total Time:
Mean:  0.15392136573791504 Standard Deviation:  0.03400851557795805
F1 Score:
Mean:  0.7254197603404376 Standard Deviation:  0.023204553945857555
```

[Q8] Compare the training time, accuracy and F1 score of the logistic regression model and the best neural network model.

```
Best Logistic Model

    Accuracy:

        Mean:  0.8405349794238682 Standart Deviation:  0.002909904
4493273167

    Total Time:

        Mean:  0.0036570231119791665 Standart Deviation:  0.000470
47122097760647

    F1 Score:

        Mean:  0.7454354643539083 Standart Deviation:  0.006523927
056779405



Best Single Layered Neural Network

Number of hidden layers: 64

    Accuracy:

            Mean:  0.8446502057613169 Standard Deviation:  0.01190
9297225092843

    Total Time:

            Mean:  0.11469316482543945 Standard Deviation:  0.0110
15765534411324

    F1 Score:

            Mean:  0.7371880328572495 Standard Deviation:  0.02238
2919936773604



Both are quite similar in terms of accuracy and f1 score, however,
the runtime of the logistic model is comparatively lower. This has
to do with the number of hidden layers in the neural network model
. The more the layers, the higher the number of computations our p
rogram has to do. On the other hand, this logistical model used in
this program does not have any neural network. Therefore, the numb
er of computations in logistic based model are less compared to ne
ural where it has multiple layers with logistical computations.
```

[Q9] Plot the accuracy and the F1 score for different values of H. Suggest a possible reason for the gap between the accuracy and the F1 score.



The graph below has a gap between its accuracy and F1 score. Peopl e often confuse them with each other. Both of them are good metric s to measure how good or bad our program performs as opposed to th e real-world scenarios. However, they are both different.

Accuracy is used when the True Positives and True negatives are mo re important while F1-score is used when the False Negatives and F alse Positives are crucial. Accuracy can be used when the class di stribution is similar while F1-score is a better metric when there are imbalanced classes.

In our scenario, based on our program, it detects that there is le ss imbalance between the classes as opposed to the scatterings in the class distribution.

Also, the values of accuracy and F1_score increases as the hidden layers increase. It is purely because of the more hidden layers do more amount of computation, leading to higher accuracy and f1 scor e. However, it should also be noticed that after a certain amount of layers, both accuracy and f1 score reach a plateau because more computation done in a single layer does not affect the predictabil ity. At that time, it is a waste of time and resources to add more hidden layers.

$$\text{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{(True Positive + False Positive + True Negative + False Negative)}}$$

$$\text{F1-score} = \left(\frac{\text{Recall}^{-1} + \text{Precision}^{-1}}{2}\right)^{-1} = 2 * \frac{\text{(Precision*Recall)}}{\text{(Precision+Recall)}}$$

[Q10] Do you notice any trend when you increase the hidden layer size from 1 to 128? If so, please describe what the trend is and suggest a reason for your observation.

The values of accuracy and F1_score increases as the hidden layers increase. It is purely because of the more hidden layers do more a mount of computation, leading to higher accuracy and f1 score.

Furthermore, it should also be noticed that after a certain number of layers, both accuracy and f1 score reach a plateau because more computation done in a single layer does not affect the predictabil ity and probability. At that time, it is a waste of time and resou rces to add more hidden layers.

Also, it is to be kept in mind that a single hidden layer or when hidden layer in a network is only 1 or 2, the accuracy and f1 scor e are totally arbitrary and should not be trusted because the netw ork has not had enough experience with the predictions and cannot compute accurately.

In general, the trend of accuracy and f1 score is increasing until it reaches plateau or reaches its peak.

[Q11] Report 10 combinations of the hyperparameter setting.

```
Model with rank: 1
Mean validation score: 0.846 (std: 0.032)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'invscaling', 'learning_rate_init': 0.0012318971316554704, 'solver'
: 'adam'}

Model with rank: 2
Mean validation score: 0.846 (std: 0.031)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'invscaling', 'learning_rate_init': 0.003362087797543175, 'solver':
'adam'}

Model with rank: 3
Mean validation score: 0.845 (std: 0.021)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'adaptive', 'learning_rate_init': 0.584322822880992, 'solver': 'sgd
'}

Model with rank: 4
Mean validation score: 0.844 (std: 0.040)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'constant', 'learning_rate_init': 0.028825474602309397, 'solver': '
adam'}

Model with rank: 5
Mean validation score: 0.844 (std: 0.031)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'adaptive', 'learning_rate_init': 0.21443795605263427, 'solver': 's
gd'}

Model with rank: 6
Mean validation score: 0.844 (std: 0.032)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'adaptive', 'learning_rate_init': 0.21210843176425248, 'solver': 's
gd'}

Model with rank: 7
Mean validation score: 0.844 (std: 0.028)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'constant', 'learning_rate_init': 0.34255810844330625, 'solver': 's
gd'}

Model with rank: 8
Mean validation score: 0.843 (std: 0.033)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (128,), 'learning
_rate': 'adaptive', 'learning_rate_init': 0.005796982941839667, 'solver':
'adam'}

Model with rank: 9
Mean validation score: 0.842 (std: 0.030)
```

```
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (128,), 'learning
_rate': 'constant', 'learning_rate_init': 0.04778732263345931, 'solver': '
adam'}

Model with rank: 10
Mean validation score: 0.841 (std: 0.038)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (128,), 'learning
_rate': 'invscaling', 'learning_rate_init': 0.027912042472638933, 'solver'
: 'adam'}
```

[Q12] Report the three best hyperparameter settings in terms of accuracy as well as the mean and standard deviation of the validation accuracy of the five random data splits for each hyperparameter setting.

The three best hyperparameter settings in terms of accuracy as wel l as the mean and standard deviation of the validation accuracy of the five random data splits for each hyperparameter setting are:

1. activation: relu
2. solver: adam
3. learning_rate: invscaling

Additionally, hidden_layer_sizes also plays a big role as aforemen tioned but more layers is not always better. All the below chronol ogically ranked models have a majority of this configuration which leads me to believe that these are the most important hyperparamet ers.

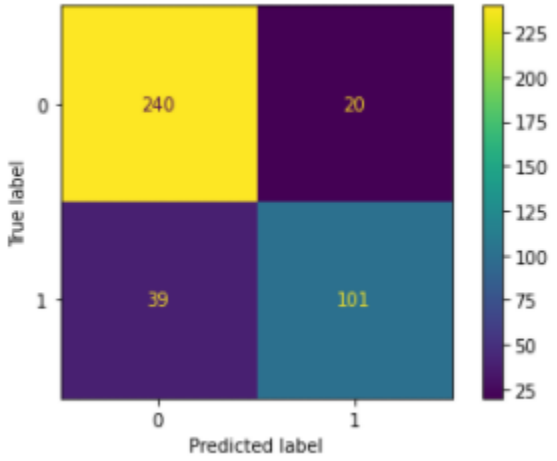
```
Model with rank: 1
Mean validation score: 0.846 (std: 0.032)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'invscaling', 'learning_rate_init': 0.0012318971316554704, 'solver'
: 'adam'}

Model with rank: 2
Mean validation score: 0.846 (std: 0.031)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'invscaling', 'learning_rate_init': 0.003362087797543175, 'solver':
'adam'}

Model with rank: 3
Mean validation score: 0.845 (std: 0.021)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), 'learning_
rate': 'adaptive', 'learning_rate_init': 0.584322822880992, 'solver': 'sgd
'}
```

[Q13] Use the best model in terms of accuracy to predict the instances in the test set (test.csv). Report the accuracy, F1 score and the confusion matrix of the predictions on the test set.

	precision	recall	f1-score	support
0.0	0.86	0.92	0.89	260
1.0	0.83	0.72	0.77	140
accuracy			0.85	400
macro avg	0.85	0.82	0.83	400
weighted avg	0.85	0.85	0.85	400

F1 Score: 0.7739463601532567
Accuracy: 0.8525

Confusion Matrix



[Q14] Name another two methods to deal with the imbalanced dataset problem.

Two methods that deal with imbalanced dataset problem are:

1. Resampling the training set. Under sampling or Oversampling is widely used in such cases. Under sampling for the samples that are in excess and oversampling for the samples of the classes that are in deficit.

2. Cluster the abundant classes. Instead of relying on random samples to cover the variety of the training samples, clustering the abundant class in r groups, with r being the number of cases in r is suggested. For each group, only the medoid (Centre of cluster) is kept. The model is then trained with the rare class and the medoids only.

Apart from these two, there are myriads of other things that can be done example, resampling with different ratios, ensemble different resampled datasets and using the right evaluation metrics.

[Q15] Use the best model in terms of accuracy to predict the instances in the test set (test.csv). Report the accuracy, F1 score and the confusion matrix of the predictions on the test set.

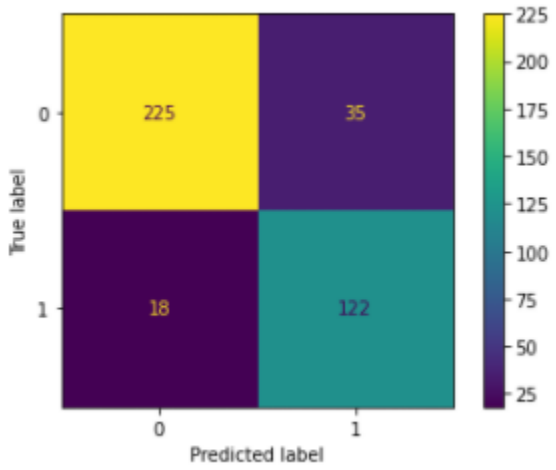
Predictions on test set:

	precision	recall	f1-score	support
0.0	0.93	0.87	0.89	260
1.0	0.78	0.87	0.82	140
accuracy			0.87	400
macro avg	0.85	0.87	0.86	400
weighted avg	0.87	0.87	0.87	400

F1 Score: 0.8215488215488216

Accuracy: 0.8675

Confusion Matrix

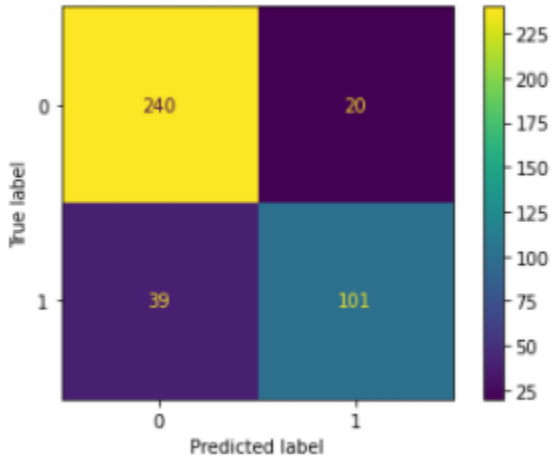


[Q16] Compare the accuracy, F1 score and the confusion matrix with those in Section 7.1.

Before Oversampling: (Data Imbalance)

F1 Score: 0.7739463601532567
Accuracy: 0.8525

Confusion Matrix

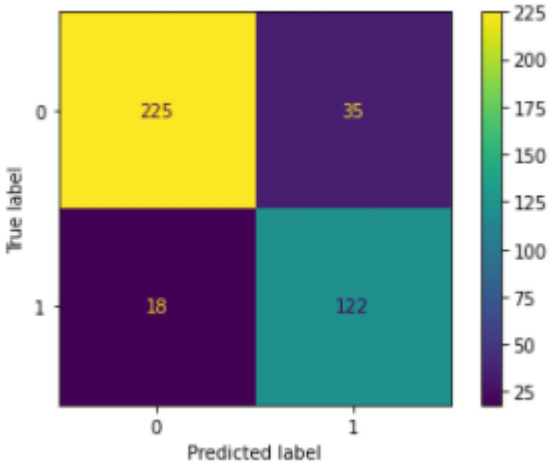


After Oversampling: (Overfitting issue might occur)

f1 score: 0.8215488215488216

Accuracy: 0.8675

Confusion Matrix



There seems to be very minor change in the accuracy and f1 score. If anything, I had predicted for the accuracy to lower down because of the duplicate elements in the data resulting to overfitting. However, the tradeoff between imbalanced dataset and overfitting seems to be about the same in this case. Although if there is one thing that should be noticed it is that the number of errors occurring in section 7.1 were mostly when class/label was equal to 1 which could have very well been due to the unbalanced data. On the contrary, after oversampling, the number of errors mostly occur in class 0, which could be due to overfitting but it cannot truly be said with the current knowledge.

Overall, the total number of errors made by our program are about the same in both the cases leading to a similar accuracy and a slightly higher f1 score. To see why that be, refer to the aforementioned definition of f1 score.