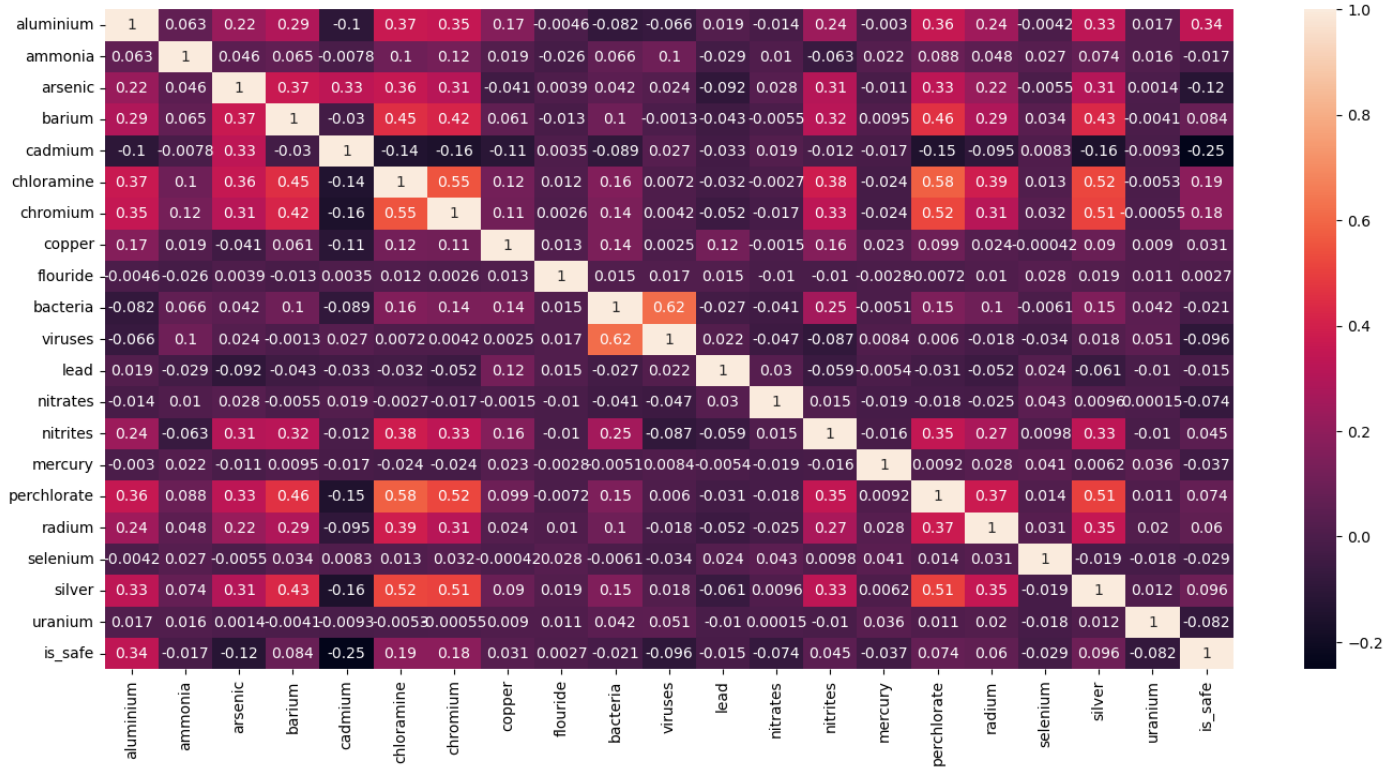
COMP4211 Assignment 1 Report

Q1.



Q2.

Mineral: perchlorate
R2 Score: 0.34727163558690055

Mineral: chromium
R2 Score: 0.31138489178953244

Mineral: silver
R2 Score: 0.29193122809804106

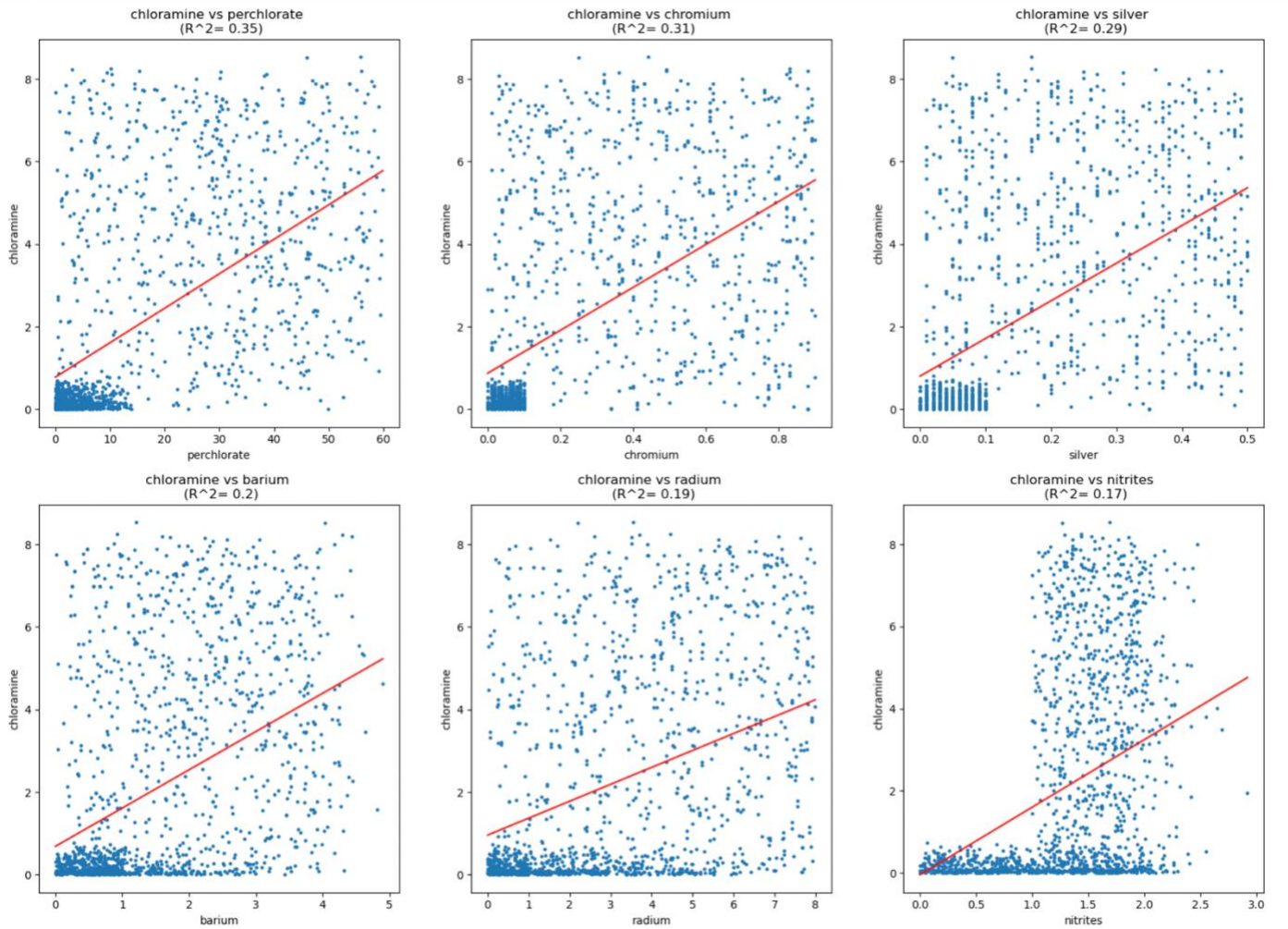
Mineral: barium
R2 Score: 0.2017232988334794

Mineral: radium
R2 Score: 0.18902582154390024

Mineral: nitrites
R2 Score: 0.1675063127334001

Mineral: ['perchlorate', 'chromium', 'silver', 'barium', 'radium', 'nitrites']
R2 Score: 0.5005101127629697

Q3



Q4
Mineral: perchlorate
Mean Square Error: 4.5180732447819425

Mineral: chromium
Mean Square Error: 4.766475100489592

Mineral: silver
Mean Square Error: 4.901130007843806

Mineral: barium
Mean Square Error: 5.525533747435942

Mineral: radium
Mean Square Error: 5.6134234969029455

Mineral: nitrites
Mean Square Error: 5.762377828135342

Mineral: ['perchlorate', 'chromium', 'silver', 'barium', 'radium', 'nitrites']
Mean Square Error: 3.457382915470382

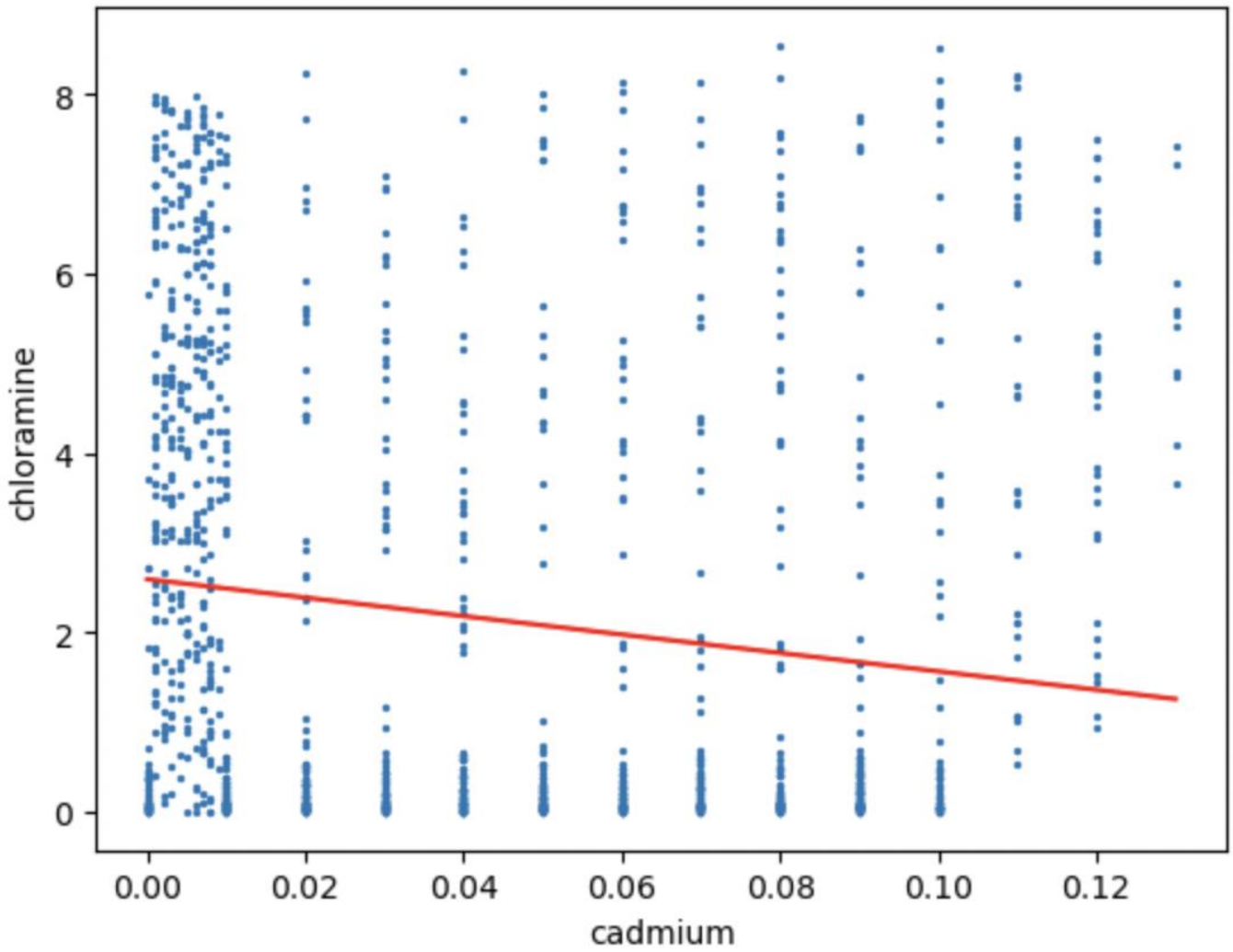
Evaluation:
In our example, the R2 score is a superior metric for measuring the machine learning model's performance. This is because the value of Mean Squared Error does not contain much meaning alone. The mean squared error for perchlorate is 4.518073244781942. It is difficult to gauge whether this is a good or bad value unless there is a comparison with other models.

On the other hand, the R2 score is bounded to 1. Therefore, the closer the R2 score of a model gets to 1, we can assume a better model. For the same case with perchlorate, the R2 value was 0.34727163558690055**,** which shows that the model's performance is questionable.

As the mean squared error increases/R2 value decreases, we can see that the linear regression line does not fit well with the validation data.

Based on the mean squared error and R2 score, the linear model using all 6 highly correlated minerals with chloramine showed the best performance, with the highest R2 score and lowest mean squared error.

Q5.



Evaluation:
The main difference between plots from Q3 and the plot in Q5 would be that the gradient of the plotted line.

The linear lines from the graphs Q3 show a positive gradient. However, the linear line with the lowest correlated mineral, cadmium, shows a negative gradient. The R2 score, 0.012403388531904369, of the linear model from Q5, is lower than any of the linear models in graph Q3.

The linear line from Q5 shows no relationship with the scattered points whereas the linear line from Q3 shows some relationship (to some extent).

Q6.

Training Time Mean of 3 Trials

	1	4	16	64	128
Time	0.632	1.763	1.505	2.521	4.436

Training Time std of 3 Trials

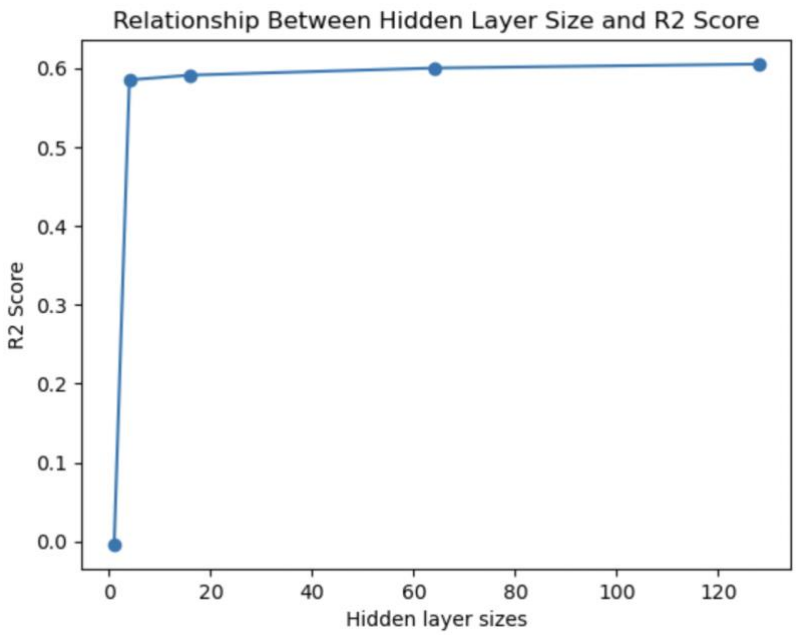
	1	4	16	64	128
Time	0.302	0.079	0.313	0.365	0.163

R2 Score Mean of 3 Trials

	1	4	16	64	128
Score	-0.004	0.585	0.591	0.6	0.605

R2 Score std of 3 Trials

	1	4	16	64	128
Score	0.002	0.017	0.007	0.002	0.003



Q7.

The linear model's best R2 score was **0.501.** On the other hand, the best R2 score for a neural network is 128 neurons in each of the three layers, with a score of **0.605**. Overall multi-layer perceptrons with the neurons of 4,16,64,128 all performed better than the best linear model.

However, the training time to train a neural network took much longer than a linear regression model. The training time for a linear model was almost imperceptibly quick, but for the MLP with 128 neurons, the average time for training was 4.436 seconds.

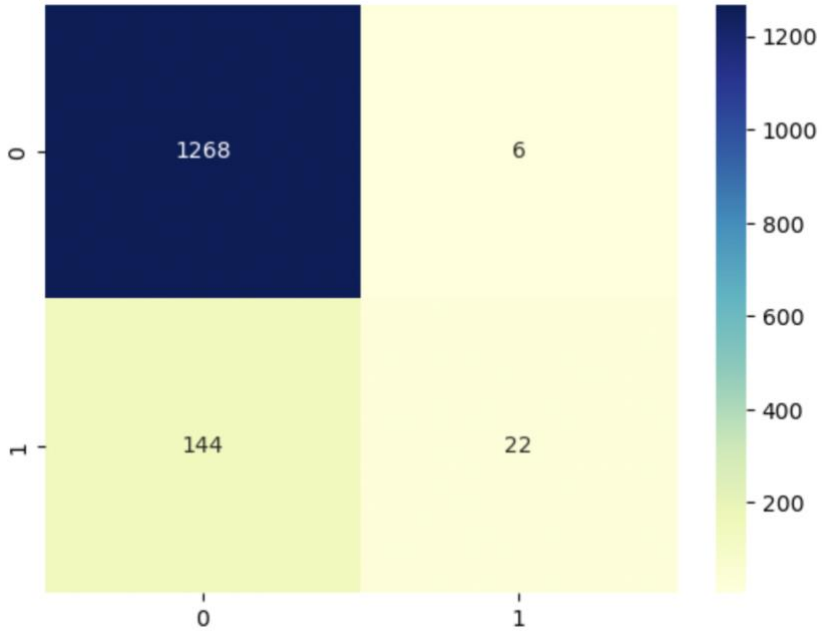
Q8.

aluminium	0.074511
ammonia	0.000000
arsenic	0.038366
barium	0.001245
cadmium	0.075890
chloramine	0.022316
chromium	0.028156
copper	0.007881
flouride	0.000760
bacteria	0.001603
viruses	0.003643
lead	0.000000
nitrates	0.000000
nitrites	0.007882
mercury	0.000000
perchlorate	0.017848
radium	0.010955
selenium	0.000000
silver	0.006992
uranium	0.012337

Q9.
loss="log_loss", learning_rate='optimal'

Training Time Mean: 0.059
Training Time std: 0.008
Accuracy Score Mean: 0.792
Accuracy Score std: 0.143
f1 Score Mean: 0.245
f1 Score std: 0.097

Q10.



We need to examine the confusion matrix well because, in some cases, the number of false negatives and false positives can be crucial factors in determining the performance of a machine learning model. For example, if we examine people with covid virus using a machine learning model, having a false negative might cause severe societal harm.

Q11.

	sag	lbfgs	sgd
Accuracy	0.890972	0.897222	0.895833
F1_score	0.364372	0.408000	0.226804

Q12.

Training Time Mean of 3 Trials

	1	4	16	64	128
Time(s)	0.131	0.305	0.754	0.802	1.431

Training Time std of 3 Trials

	1	4	16	64	128
Time(s)	0.028	0.194	0.064	0.162	0.059

Accuracy Score Mean of 3 Trials

	1	4	16	64	128
Score	0.628	0.887	0.944	0.944	0.948

Accuracy Score std of 3 Trials

	1	4	16	64	128
Score	0.363	0.003	0.007	0.003	0.003

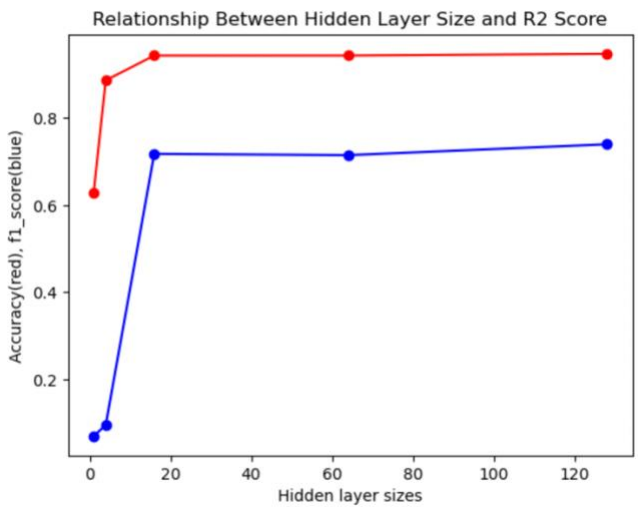
f1_score Mean of 3 Trials

	1	4	16	64	128
Score	0.069	0.095	0.718	0.715	0.74

f1_score std of 3 Trials

	1	4	16	64	128
Score	0.097	0.134	0.035	0.032	0.019

Q13.



The reason for the gap may be due to the imbalanced dataset. The train data has a lot of data points with 'is_safe' == 0, but there aren't as many samples that has data points with 'is_safe' == 1. The data distribution of the test should be similar. Therefore, the model may have low accuracy in classifying the test data as 'safe' and may generate many **false negatives** (the actual data point should be classified as 'safe' but the model classified the data point as 'unsafe')**.** Having a lot of false negatives or false positives cause the f1_score to be low.

However, since there are more test data with 'is_safe' == 0, the model's accuracy would be high, but with a low f1 score.

Q14.

The best logistic regression model used the learning rate type of "lbfgs" with an accuracy of 0.897 and an f1 score of 0.408.

On the other hand, the best MLP classifier model had 128 neurons per layer with an accuracy of 0.948 and an f1 score of 0.74.

Clearly, the MLP classifier performed better than the logistic regression model in both criteria, accuracy and f1 score. However, in terms of training time the logistic regression models trained much quicker. In terms of SGDClassifier from Q9, the mean of the training time of the model with three different trials only took 0.066 seconds, whereas the MLP classifier with 128 neurons took 1.447 seconds.

Q15.

The accuracy and f1 score increase as the number of neurons per layer increases. However, the performance saturates as the number of neurons reaches 16. This may be because of the small dataset. The accuracy won't significantly increase with increasing the number of neurons after 16 because the dataset size is too small and the model complexity is too low. The model has only 3 layers with few neurons per layer.

Training time increases as the number of neurons per layer increases. This is because forward propagation, backward propagation, and gradient updates are done toward more neurons.

Q16.

```
parameters = {

        'activation':['identity', 'logistic', 'tanh', 'relu'],
        'solver':['lbfgs', 'sgd', 'adam']

        }
```

Therefore, there can be 12 (4*3) combinations in total.

6 of them would be

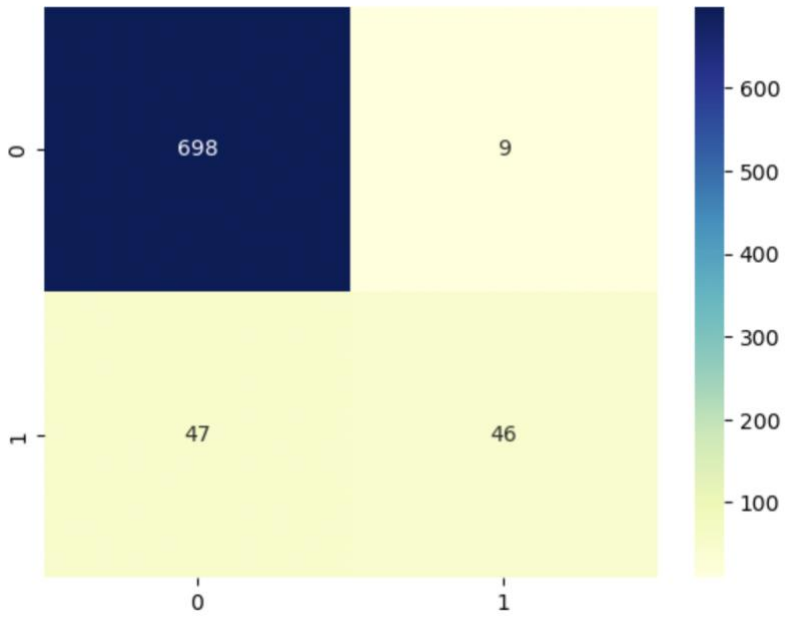
```
3    {'activation': 'logistic', 'solver': 'lbfgs'}
6       {'activation': 'tanh', 'solver': 'lbfgs'}
9       {'activation': 'relu', 'solver': 'lbfgs'}
0    {'activation': 'identity', 'solver': 'lbfgs'}
11       {'activation': 'relu', 'solver': 'adam'}
4     {'activation': 'logistic', 'solver': 'sgd'}
```

Q17.

	rank_test_score	params	mean_test_score	std_test_score
5	1	{'activation': 'logistic', 'solver': 'adam'}	0.936769	0.008346
6	2	{'activation': 'tanh', 'solver': 'lbfgs'}	0.926903	0.003826
11	3	{'activation': 'relu', 'solver': 'adam'}	0.924958	0.008539

Q18.

Accuracy Score: 0.93
f1_score: 0.622

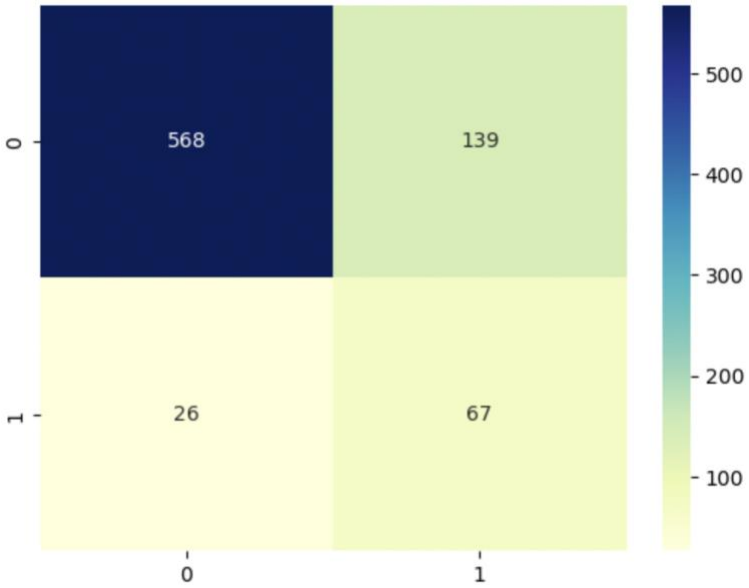


Q19.

Oversampling method

Accuracy Score: 0.794

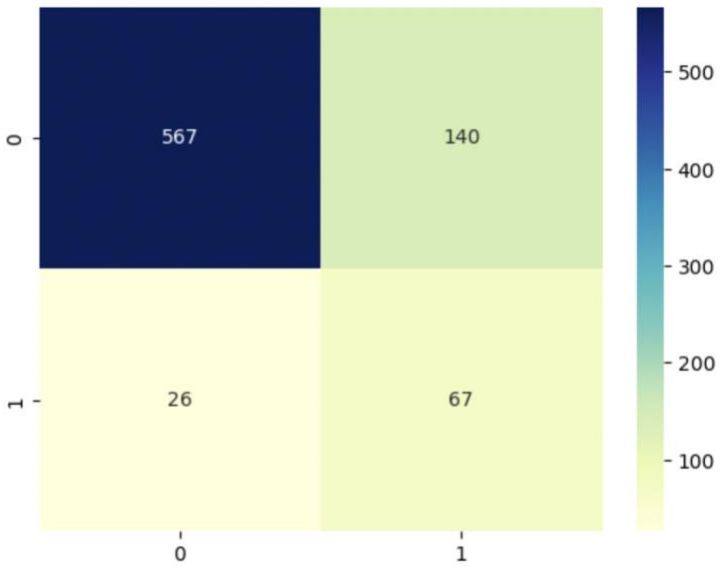
f1_score: 0.448



Different weight method

Accuracy Score: 0.792

f1_score: 0.447



Q20

Both oversampling method and weight method have decreased the accuracy as well as the f1_score compared to the result from 7.1. The reason is unclear, but we can estimate the reason is due to the balanced training set. The balanced training set must have increased the model's tendency to classify a data point to 'is_safe.' Originally the data was very imbalanced and had a lot more data points that were 'unsafe.' Thus, the model has learned to classify data points to the category 'unsafe.' On the other hand, the model couldn't learn as much in classifying a data point into the 'safe' category as the train data was imbalanced. However, through using oversampling method and the class weight method, the train data has been balanced and the model has much more tendency in classifying a data point into the 'safe' category.

Unfortunately, the model has classified too many data points into the 'safe' category generating numerous false positives. This is why from the confusion matrix from Q19, the number of false positives has increased in comparison to the confusion matrix from Q18. But on the flip side, the number of false negatives has decreased which is a very

good sign. It is may be fine to assign drinkable water to the ‘unsafe’ category. However, assigning undrinkable, unsafe water to the ‘safe’ category should not occur. Therefore in terms of the confusion matrix the model with the balanced train data might be a better model for real-life use cases.