Part 1: Data Preprocessing

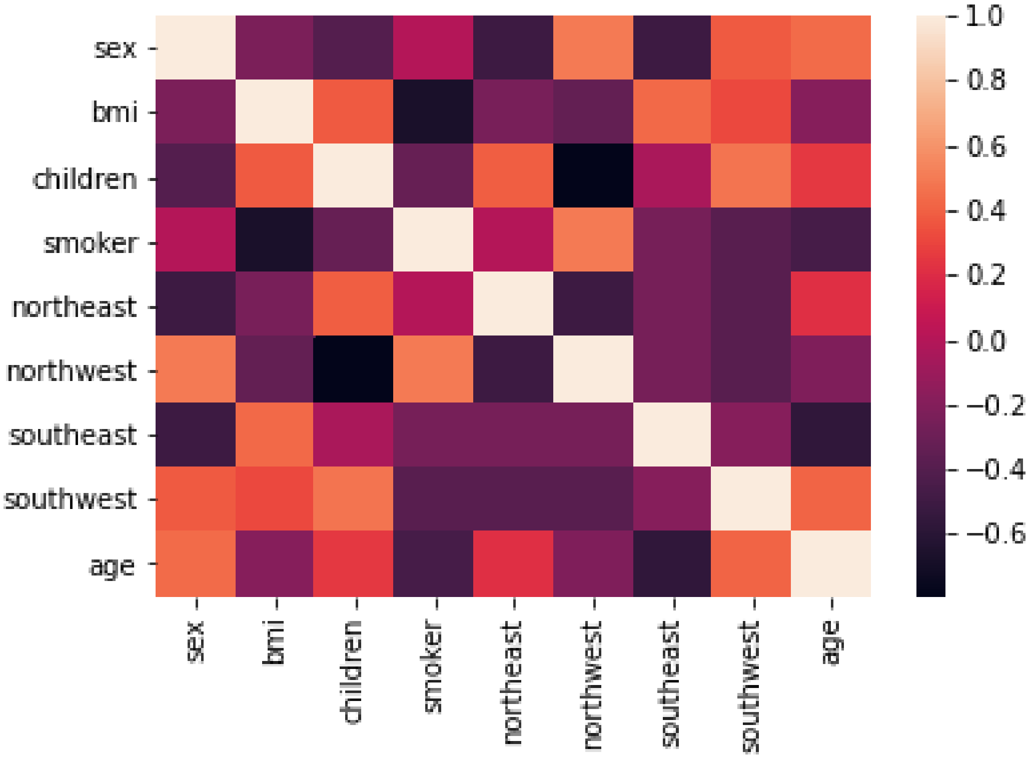
Q1

Train.head() and test.head()

	age	sex	bmi	children	smoker	charge	label	northeast	northwest	southeast	southwest
0	27	1	28.500	0	1	18310.74200	1	0	1	0	0
1	18	0	37.290	1	0	2219.44510	0	0	0	1	0
2	43	0	20.045	2	1	19798.05455	1	1	0	0	0
3	35	0	38.095	2	0	24915.04626	1	1	0	0	0
4	59	1	25.460	1	0	12913.99240	0	1	0	0	0

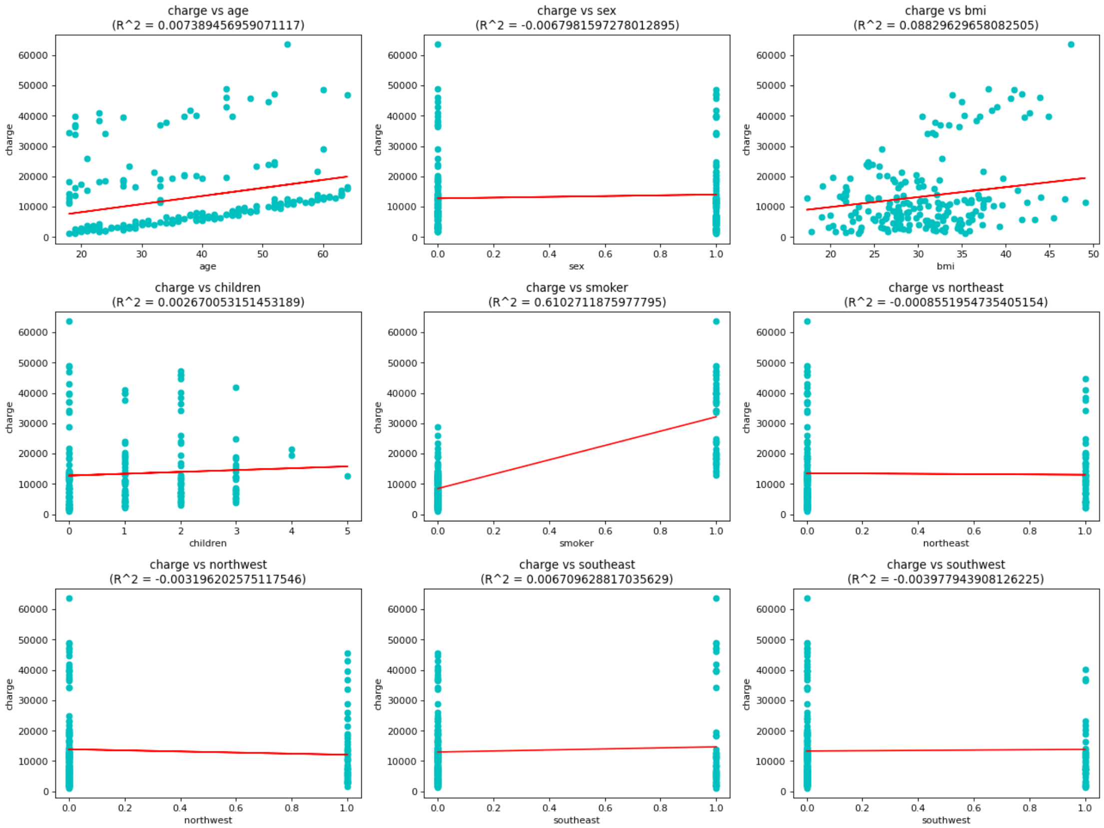
	age	sex	bmi	children	smoker	charge	label	northeast	northwest	southeast	southwest
0	59	0	26.505	0	0	12815.44495	0	1	0	0	0
1	35	0	31.000	1	0	5240.76500	0	0	0	0	1
2	57	0	23.180	0	0	11830.60720	0	0	1	0	0
3	52	0	37.525	2	0	33471.97189	1	0	1	0	0
4	44	0	36.955	1	0	8023.13545	0	0	1	0	0

heatmap



Part 2: Regression

Q2 - Q3




```
R^2 score on the validation set for the 10th regression model(the one
 that uses all the nine features): 0.7632440007914385
```

Part 3: Classification

6.1 Feature Selection

Q4

	age	sex	bmi	children	smoker	northeast	northwest	southeast	southwest
f_classif	16.206831	0.184330	1.349763	1.954379	1363.904410	0.114417	0.233475	1.724736	1.468004
chi2	80.688568	0.092425	1.661031	2.569718	454.075507	0.086750	0.175358	1.264119	1.117395
mutual_info_classif	0.052957	0.000609	0.000000	0.008538	0.320690	0.000000	0.000000	0.010120	0.005580

selected features = `'age', 'bmi', 'children', 'smoker', 'northwest', 'southeast', 'southwest'`

Q5

using eta0 = [0.001, 0.005, 0.01, 0.05, 0.1]

the model setting (for the repeated 3 times) followed by the training time and performance of the logistic regression model

```
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.001,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
```

```
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.001,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.001,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
```

	training time	accuracy	F1 score
0	0.007999	9.050000e-01	0.825688

	training time	accuracy	F1 score
1	0.008000	9.050000e-01	0.825688
2	0.003995	9.050000e-01	0.825688
mean	0.006665	9.050000e-01	0.825688
std	0.001888	1.110223e-16	0.000000

```
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.005,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.005,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
```

```
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.005,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
```

	training time	accuracy	F1 score
0	0.002998	9.050000e-01	0.825688
1	0.006000	9.050000e-01	0.825688
2	0.004995	9.050000e-01	0.825688
mean	0.004665	9.050000e-01	0.825688
std	0.001248	1.110223e-16	0.000000

```
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.01,
```

```
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.01,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.01,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
```

```
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
```

	training time	accuracy	F1 score
0	0.002999	0.905000	0.825688
1	0.004002	0.905000	0.825688
2	0.007000	0.910000	0.836364
mean	0.004667	0.906667	0.829247
std	0.001700	0.002357	0.005033

```
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.05,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
```

```
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.05,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.05,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
```

	training time	accuracy	F1 score
0	0.003000	0.915000	0.849558
1	0.005002	0.905000	0.825688

	training time	accuracy	F1 score
2	0.003999	0.915000	0.849558
mean	0.004001	0.911667	0.841601
std	0.000817	0.004714	0.011252

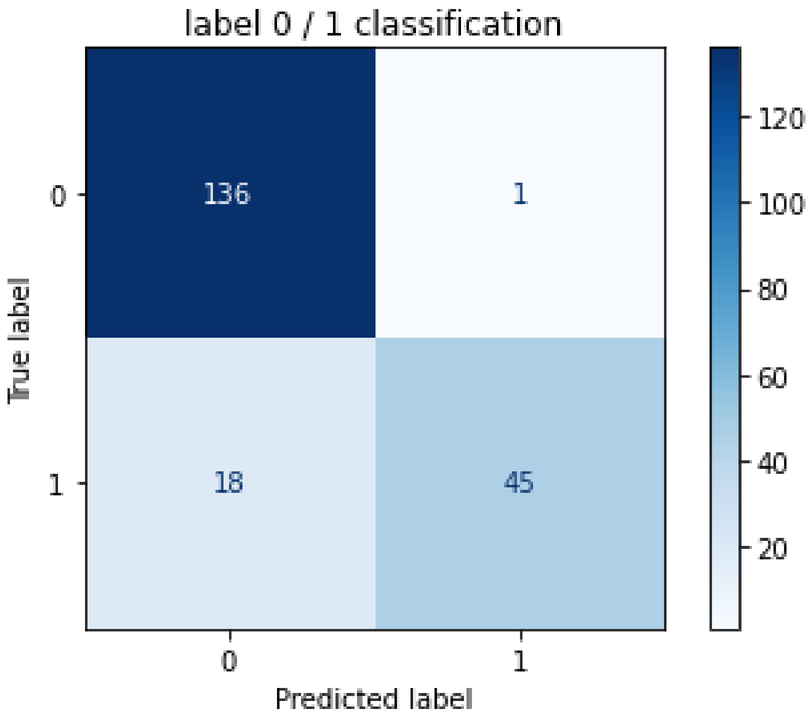
```
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.1,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.1,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
```

```
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
{'alpha': 0.0001,
 'average': False,
 'class_weight': None,
 'early_stopping': False,
 'epsilon': 0.1,
 'eta0': 0.1,
 'fit_intercept': True,
 'l1_ratio': 0.15,
 'learning_rate': 'constant',
 'loss': 'log',
 'max_iter': 1000,
 'n_iter_no_change': 5,
 'n_jobs': None,
 'penalty': 'l2',
 'power_t': 0.5,
 'random_state': None,
 'shuffle': True,
 'tol': 0.001,
 'validation_fraction': 0.1,
 'verbose': 0,
 'warm_start': False}
```

	training time	accuracy	F1 score
0	0.003000	0.915000	0.852174
1	0.004002	0.910000	0.842105
2	0.002999	0.905000	0.825688
mean	0.003334	0.910000	0.839989
std	0.000473	0.004082	0.010916

```
Best eta0 based on mean accuracy: 0.05
```

Q6



true positive = 45, true negative = 136, false positive = 1, and false negative = 18

reason why we need to focus on these numbers: depending on the domain problem, we need to ensure that the frequency of each predicted label is balanced with the true label. In the extreme case, like robbery, the event that alarm is off is expected to be significantly greater than the event that alarm is on. In real life, robbery does not happen very often. Therefore, if the classifier simply predicts all the time that there is no robbery despite the input feature, then it is safe to say that the model does not perform well. However, this observation can not be concluded from the accuracy, but the above four values can do so instead. Additionally, sometimes we also want to balance between false positive and false negative.

6.3 Single-hidden-layer Neural Networks

Q7

`'hidden_layer_sizes = 1'`

	training time	accuracy	F1 score
0	0.040994	0.805000	0.589474
1	0.029000	0.315000	0.479087
2	0.029998	0.315000	0.479087
mean	0.033331	0.478333	0.515883

	training time	accuracy	F1 score
std	0.005434	0.230988	0.052037

```
'hidden_layer_sizes = 2'
```

	training time	accuracy	F1 score
0	0.035000	0.315000	0.479087
1	0.092000	0.810000	0.577778
2	0.033000	0.315000	0.479087
mean	0.053333	0.480000	0.511984
std	0.027353	0.233345	0.046523

```
'hidden_layer_sizes = 8'
```

	training time	accuracy	F1 score
0	0.226997	0.880000	0.769231
1	0.119003	0.805000	0.597938
2	0.106000	0.840000	0.666667
mean	0.150667	0.841667	0.677945
std	0.054234	0.030641	0.070383

```
'hidden_layer_sizes = 16'
```

	training time	accuracy	F1 score
0	0.119999	0.870000	0.779661
1	0.082000	0.885000	0.776699
2	0.149003	0.900000	0.814815
mean	0.117001	0.885000	0.790392

	training time	accuracy	F1 score
std	0.027436	0.012247	0.017312

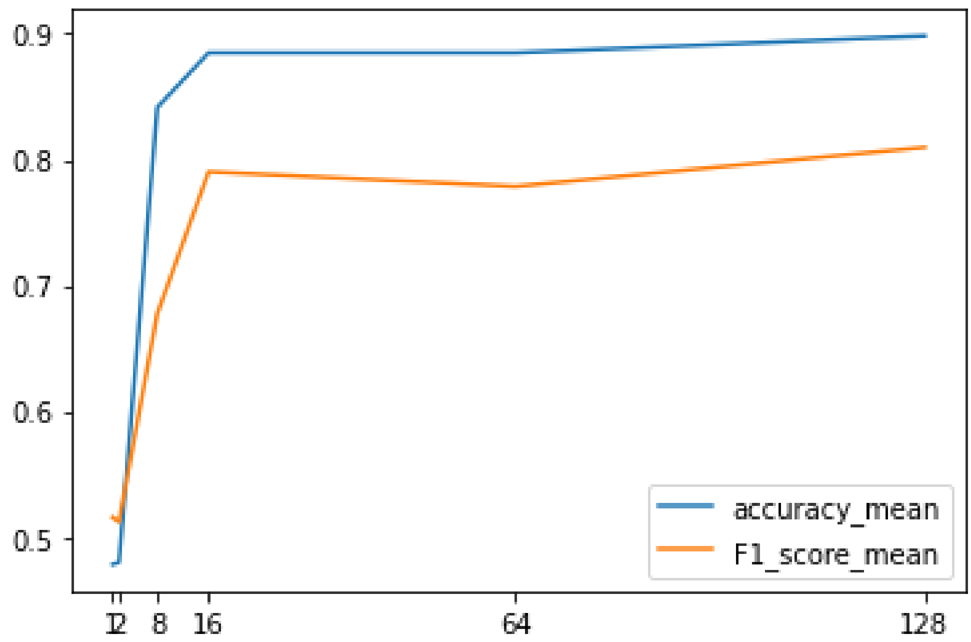
'hidden_layer_sizes = 64'

	training time	accuracy	F1 score
0	0.112000	0.895000	0.803738
1	0.105002	0.890000	0.792453
2	0.083001	0.870000	0.740000
mean	0.100001	0.885000	0.778730
std	0.012356	0.010801	0.027771

'hidden_layer_sizes = 128'

	training time	accuracy	F1 score
0	0.111001	0.895000	0.803738
1	0.096006	0.910000	0.833333
2	0.091002	0.890000	0.792453
mean	0.099336	0.898333	0.809841
std	0.008497	0.008498	0.017238

Q8



A possible reason for the gap between the accuracy and the F1 score: The classes are likely to be imbalanced. f1 score is the harmonic mean of precision and recall, which means that accuracy=(TP+TN)/(TP+TN+FN+FP) and F1=TP/(TP+(FP+FN)/2).

Q9

	logistic regression	best neural network
Training time	0.004334	0.099336
Accuracy	0.911667	0.898333
F1 score	0.841601	0.809841

Q10

trend when increase the hidden layer size from 1 to 128:

1. Larger hidden layer model tends to train for much longer than smaller hidden layer model. The reason is obvious which is because larger model has higher number of parameters or weight to be updated compared to smaller model.
2. The accuracy and f1 score increase as the size of hidden layer is increased up to a point when both scores no longer increase or reach the phase with steady but slow increase. According to the plot above, hidden layer size of 16 seems to fit the data well enough, meaning that 16 hidden layers is likely to be close to ideal to capture the all the features from the input and give a correct prediction. Thus, further increasing the hidden layer size is no longer necessary, i.e. 64 and 128 hidden layers are not helpful for the classification. On the other hand, increasing the layer too much may actually decreases the score, which happen when the model becomes very flexible and overfits to the training data.

Part 4: Performance Enhancement

7.1 Hyperparameter Tuning

Q11

```
Model with rank: 1
Mean validation score: 0.887 (std: 0.016)
Parameters: {'activation': 'tanh', 'hidden_layer_sizes': (256,),
'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 2
Mean validation score: 0.885 (std: 0.019)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (256,),
'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 3
Mean validation score: 0.885 (std: 0.011)
Parameters: {'activation': 'tanh', 'hidden_layer_sizes': (64,), '
learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 4
Mean validation score: 0.883 (std: 0.020)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (128,),
'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 5
Mean validation score: 0.871 (std: 0.007)
Parameters: {'activation': 'tanh', 'hidden_layer_sizes': (128,),
'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 6
Mean validation score: 0.850 (std: 0.025)
Parameters: {'activation': 'tanh', 'hidden_layer_sizes': (256,),
'learning_rate': 'invscaling', 'solver': 'sgd'}

Model with rank: 7
Mean validation score: 0.837 (std: 0.044)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (64,), '
learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 8
Mean validation score: 0.790 (std: 0.014)
Parameters: {'activation': 'logistic', 'hidden_layer_sizes': (256
,), 'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 9
Mean validation score: 0.684 (std: 0.002)
Parameters: {'activation': 'logistic', 'hidden_layer_sizes': (128
,), 'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 9
Mean validation score: 0.684 (std: 0.002)
Parameters: {'activation': 'logistic', 'hidden_layer_sizes': (256
,), 'learning_rate': 'invscaling', 'solver': 'sgd'}
```

Q12

```
Model with rank: 1
Mean validation score: 0.887 (std: 0.016)
```

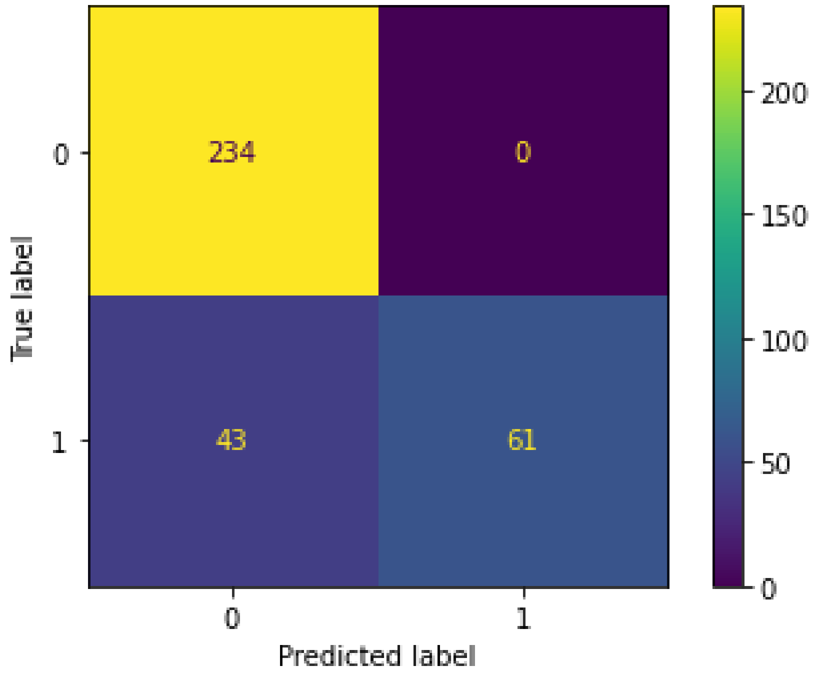
```
Parameters: {'activation': 'tanh', 'hidden_layer_sizes': (256,),
'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 2
Mean validation score: 0.885 (std: 0.019)
Parameters: {'activation': 'relu', 'hidden_layer_sizes': (256,),
'learning_rate': 'constant', 'solver': 'sgd'}

Model with rank: 3
Mean validation score: 0.885 (std: 0.011)
Parameters: {'activation': 'tanh', 'hidden_layer_sizes': (64,), '
learning_rate': 'constant', 'solver': 'sgd'}
```

Q13

Accuracy = `0.8727810650887574`, F1 score = `0.74`



7.2 Comparison of Classification Methods

Q14

In logistic regression, it is generally not applicable to non-linear problems, and it can only predict categorical outcome, as in the above datasets.

Neural network often needs more training time compared to logistic regression when the hidden layer is comparably large.

Q15

	accuracy	F1 score
constant	0.872781	0.739394
optimal	0.896450	0.808743
invscaling	0.872781	0.739394

Q16

	accuracy	F1 score
logistic	0.70	0.189189
tanh	0.91	0.836364
relu	0.90	0.814815

Q17

Logistic: not zero centered, vanishing gradient

Tanh: vanishing gradient when saturated

Relu: not zero centered, never activate at less than zero values or vanishing gradient which lead to not updating weight or parameters