CMT316 - Coursework 1, Part 1

Peter Kennedy, 2092220

April 20, 2021

Part 1

1)

The following table summarises the total number of True Positive (TP), False Positive (FP), True Negative (TN) and False Negative results given in question 1, part 1 of the coursework assignment.

	True	False
Positive	TP: 6	FP: 4
Negative	TN: 7	FN: 3

All answers in this question have been quoted to 3 decimal places.

Accuracy

The accuracy of the dataset can be calculated using the following formula

$$\frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$= \frac{6 + 7}{6 + 7 + 4 + 3}$$

$$= 0.650$$

Precision

The precision of the dataset can be calculated using the following formula

$$\frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$= \frac{6}{6 + 4}$$

$$= 0.600$$

Recall

The recall of the dataset can be calculated using the following formula

$$\frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$= \frac{6}{6+3}$$

$$= 0.667$$

F1-Score

The F1-score of the dataset can be calculated using the following formula

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

$$= 2 \times \frac{0.6 \times 2/3}{0.6 + 2/3}$$

$$= 0.632$$

2)

Regression

To predict the value of the houses in the test set, a simple neural network was constructed and trained with the following architecture

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 6)                 42
_________________________________________________________________
dense_2 (Dense)              (None, 6)                 42
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 7
=================================================================
Total params: 91
Trainable params: 91
Non-trainable params: 0
_________________________________________________________________
None
```

The number of input variables simply relates to the number of explanatory variables in the training and test sets. The Relu activation function was used for all neurons. The optimiser used was adam and the loss function was mean squared error. To train this model the following line of code was used

```
model.fit(X_train_np_norm, Y_train_reg_np, epochs=250,
↪  batch_size=10, verbose=0)
```

Upon completion of training, to check the root mean squared error between the true values in the test set and predictions based on the explanatory variables in the test set, the following function was defined and used.

```
def regresseion_rmse(model, X, Y):

    Y_pred = model.predict(X)
    rmse = math.sqrt(mean_squared_error(Y, Y_pred))

    return rmse
```

and so, on running the line

```
regresseion_rmse(model, X_test_np_norm, Y_test_reg_np)
```

a rmse of 7.455 (to 3 decimal places) was returned.

Classification

For the classification task, an additional column was added to the pandas DataFrame (used to import the real-state datasets) indicating whether the house was 'expensive' or not, as defined in the question. A zero was assigned to non-expensive houses. A 1 was assigned to expensive houses.

After this, a simple SVM classifier was able to be constructed and trained using the lines

```
svm_clf = sklearn.svm.SVC(kernel='linear', gamma='auto')
svm_clf.fit(X_train_np_norm, Y_train_class_np)
```

Upon completion of training, predictions were able to be made on the test set using the following code

```
Y_test_class_predictions = svm_clf.predict(X_test_np_norm)
```

To then gather performance matrix, a classification report was able to be written using

```
print(classification_report(Y_test_class_np,
↪  Y_test_class_predictions))
```

which gives

	precision	recall	f1-score	support
0.0	0.81	0.81	0.81	32
1.0	0.93	0.93	0.93	81
accuracy			0.89	113
macro avg	0.87	0.87	0.87	113
weighted avg	0.89	0.89	0.89	113

From this, the accuracy from when comparing the predictions to the true labels (of expensive vs non-expensive) is 0.89.