ANN Classifying Assignment

Goal:

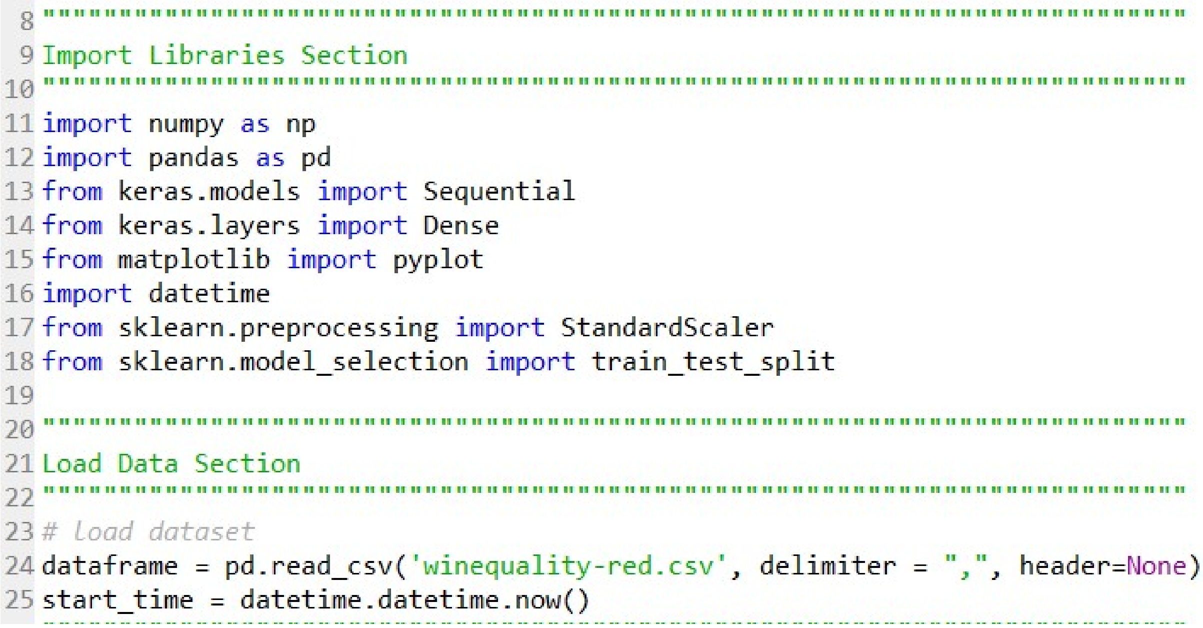
Produce, following the code we have developed in class and using Keras for the neural network functions, a classifier neural network that successfully predicts wine quality.

Dataset website: https://archive.ics.uci.edu/ml/datasets/Wine+Quality

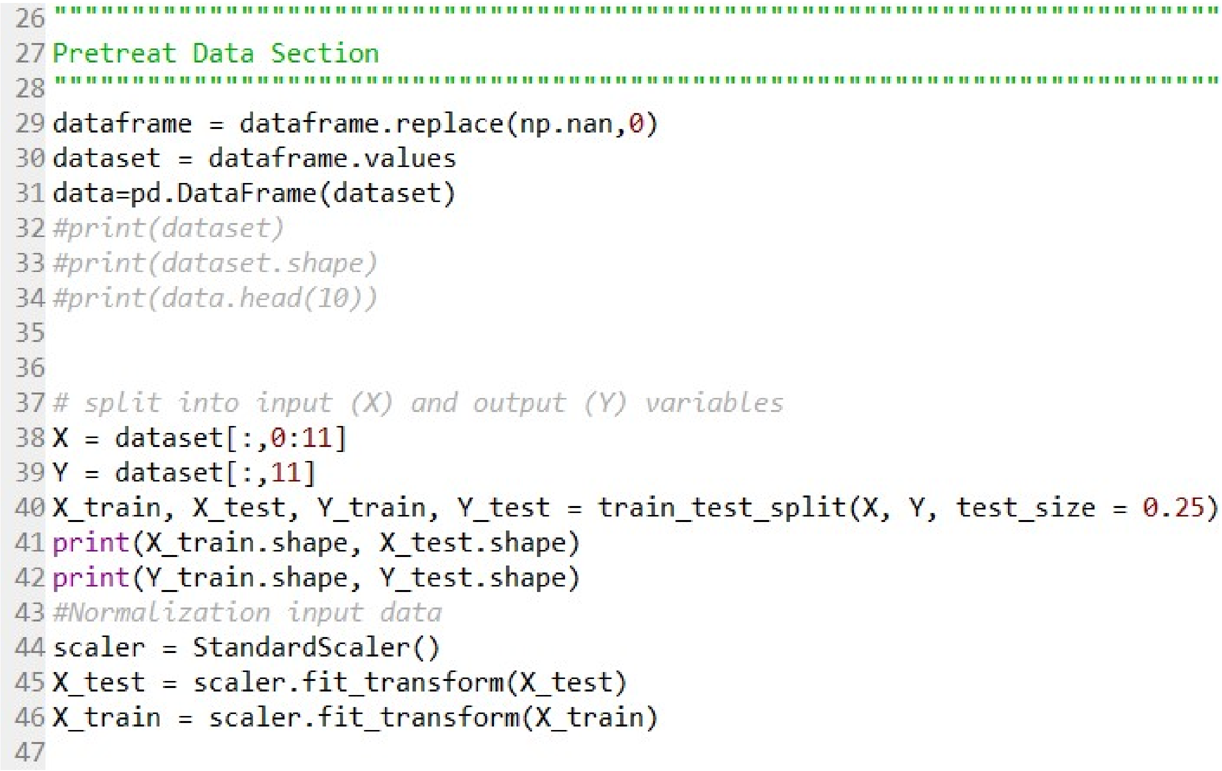
1. Your objective should be stated in written form. What are you trying to accomplish? Predicting a number? Classifying? Your objective must reference the context of the problem, specifically.

My objective is to build a classifying model to predict the wine quality based on the variables given including fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, and sulphates.

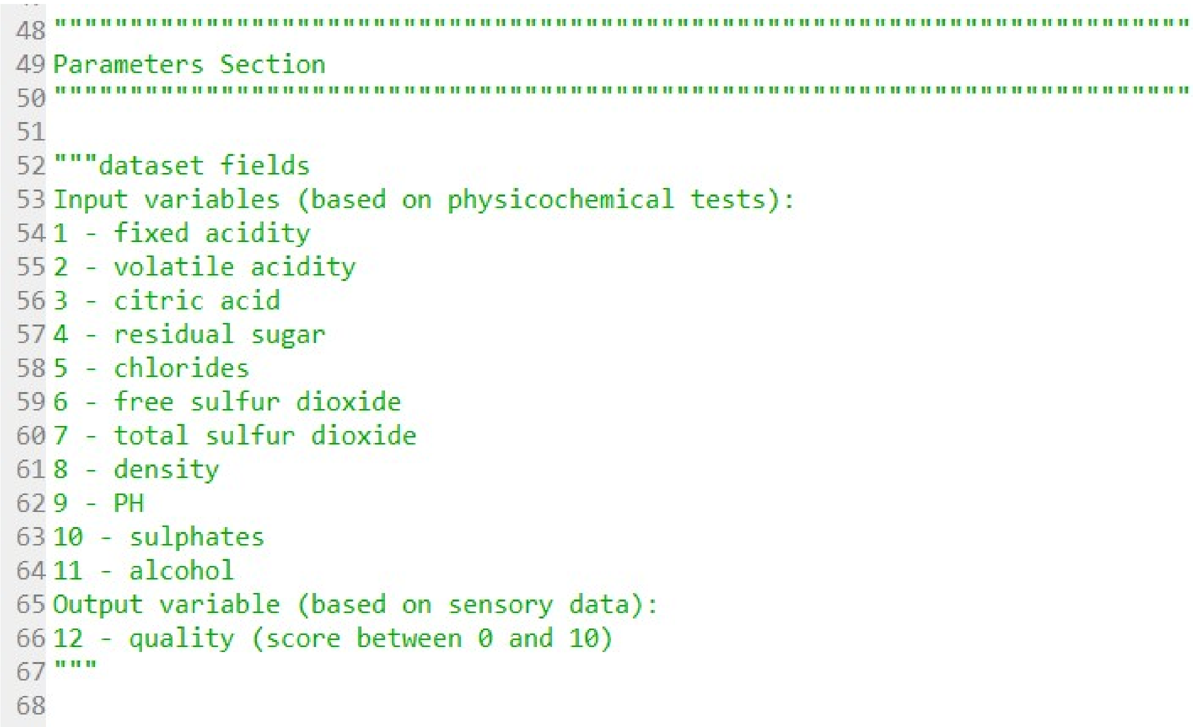
2. Your final ANN model, in code.



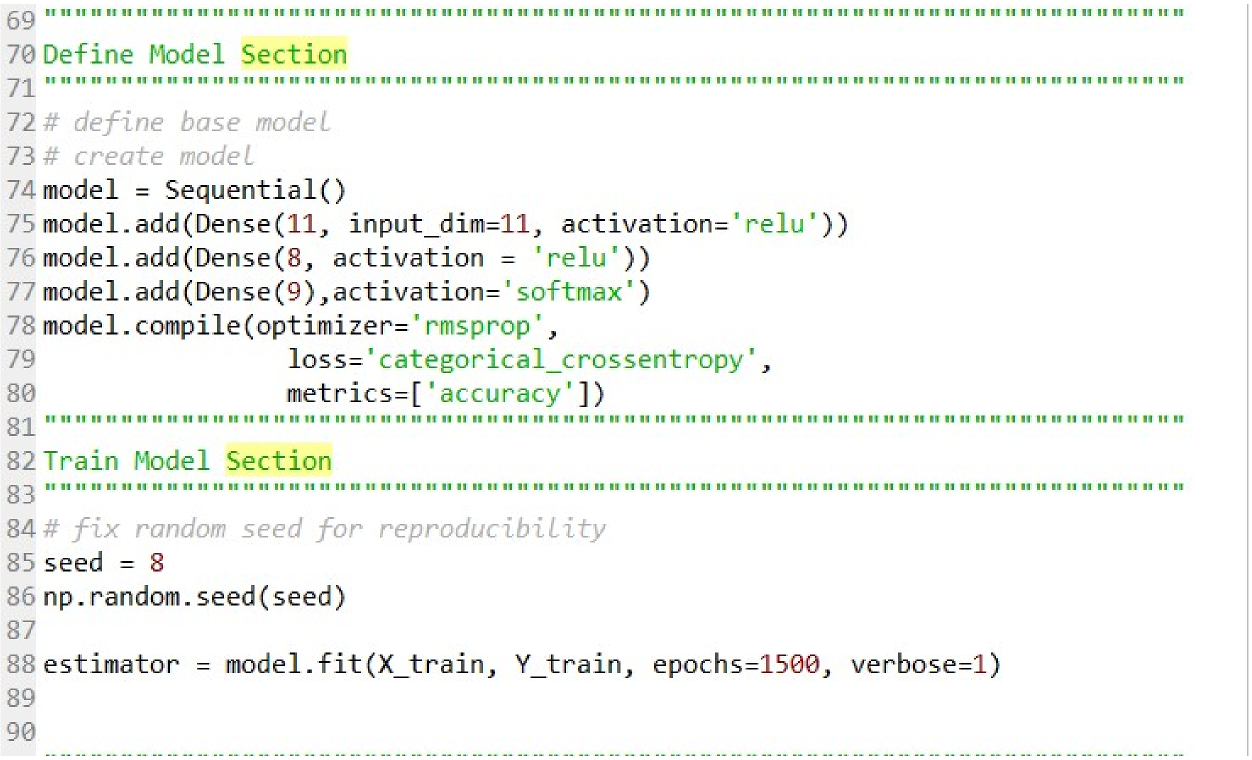
```
""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
Import Libraries Section
""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
import numpy as np
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense
from matplotlib import pyplot
import datetime
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
Load Data Section
""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
# load dataset
dataframe = pd.read_csv('winequality-red.csv', delimiter = ",", header=None)
start_time = datetime.datetime.now()
```



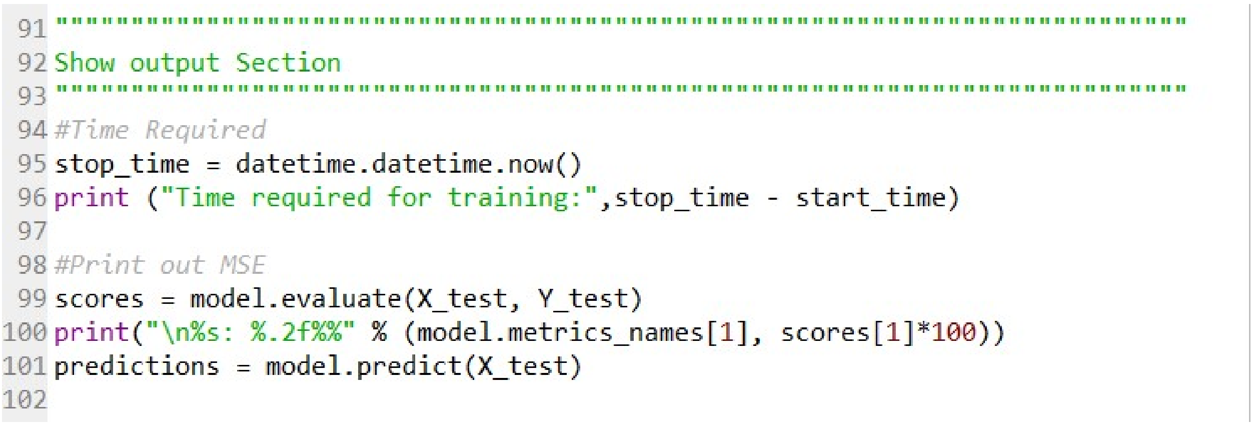
```python
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
Pretreat Data Section
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
dataframe = dataframe.replace(np.nan,0)
dataset = dataframe.values
data=pd.DataFrame(dataset)
#print(dataset)
#print(dataset.shape)
#print(data.head(10))


# split into input (X) and output (Y) variables
X = dataset[:,0:11]
Y = dataset[:,11]
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.25)
print(X_train.shape, X_test.shape)
print(Y_train.shape, Y_test.shape)
#Normalization input data
scaler = StandardScaler()
X_test = scaler.fit_transform(X_test)
X_train = scaler.fit_transform(X_train)

```



```
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
Parameters Section
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""

"""dataset fields
Input variables (based on physicochemical tests):
1 - fixed acidity
2 - volatile acidity
3 - citric acid
4 - residual sugar
5 - chlorides
6 - free sulfur dioxide
7 - total sulfur dioxide
8 - density
9 - PH
10 - sulphates
11 - alcohol
Output variable (based on sensory data):
12 - quality (score between 0 and 10)
"""

```



```
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
Define Model Section
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
# define base model
# create model
model = Sequential()
model.add(Dense(11, input_dim=11, activation='relu'))
model.add(Dense(8, activation = 'relu'))
model.add(Dense(9),activation='softmax')
model.compile(optimizer='rmsprop',
                loss='categorical_crossentropy',
                metrics=['accuracy'])
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
Train Model Section
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
# fix random seed for reproducibility
seed = 8
np.random.seed(seed)

estimator = model.fit(X_train, Y_train, epochs=1500, verbose=1)


```



```
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
Show output Section
"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
#Time Required
stop_time = datetime.datetime.now()
print ("Time required for training:",stop_time - start_time)

#Print out MSE
scores = model.evaluate(X_test, Y_test)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
predictions = model.predict(X_test)

```

3. Your final model and training algorithm, in words.

This is a multi-class classification problem, meaning that there are more than two classes to be predicted. My final model is multi-class classification model and my training algorithm is classification algorithm. The classification model has 11 units for the first layer, 8 hidden units for the intermediate layer, and 9 unit in the final layer. Relu(rectified linear unit) activation function is applied to both first and second layer. The last layer use activation as soft_max. The optimizer is rmsprop, the loss function is categorical_crossentropy, and the matrix is accuracy.

4. Your experimental plan for arriving at the final model

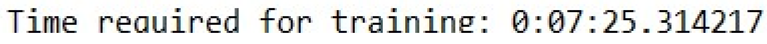
Answer: The experimental plan is to change epochs, change batch size, add intermediate layer, change unit in layers. After rerunning the code, the ACC will be changed.

I change the epochs to 50, 100, 200, 300, 500, 1000. The final epoch is 1000.

I change the batch size to 20, 32, 64, 128, 256, 512. The final batch size is 32(default).

I add a intermediate layer and set the unit to 8.

5. How long it took to run all the models in your experimental plan



```
Time required for training: 0:07:25.314217
```

6. An explanation of the input variables and any preprocessing steps you took

"""dataset fields

Input variables (based on physicochemical tests):

1 - fixed acidity

2 - volatile acidity

3 - citric acid

4 - residual sugar

5 - chlorides

6 - free sulfur dioxide

7 - total sulfur dioxide

8 - density

9 - PH

10 - sulphates

11 - alcohol

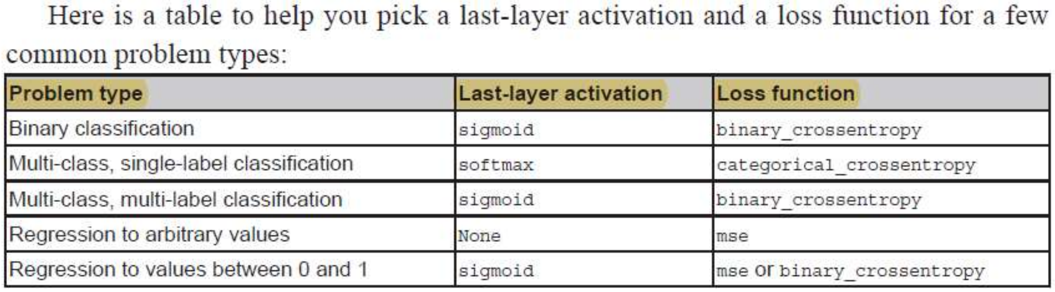
Output variable (based on sensory data):

12 - quality (score between 0 and 10)

"""

7. An explanation of your metrics and justification for your choice.

The best loss function to use in this case is categorical_crossentropy. The matrix used should be accuracy.



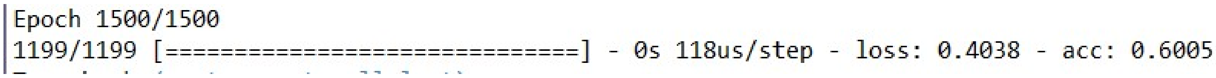
Here is a table to help you pick a last-layer activation and a loss function for a few common problem types:

Problem type	Last-layer activation	Loss function
Binary classification	`sigmoid`	`binary_crossentropy`
Multi-class, single-label classification	`softmax`	`categorical_crossentropy`
Multi-class, multi-label classification	`sigmoid`	`binary_crossentropy`
Regression to arbitrary values	`None`	`mse`
Regression to values between 0 and 1	`sigmoid`	`mse` or `binary_crossentropy`

8. An explanation of your method to validate the model

To use test dataset to validate the train dataset. Use acc metrics to evaluate the result.

9. Your results in terms of appropriate metrics for the objective and problem



```
Epoch 1500/1500
1199/1199 [==============================] - 0s 118us/step - loss: 0.4038 - acc: 0.6005
```

The ACC is 0.6005.

The most appropriate metrics for this progression problem is ACC.