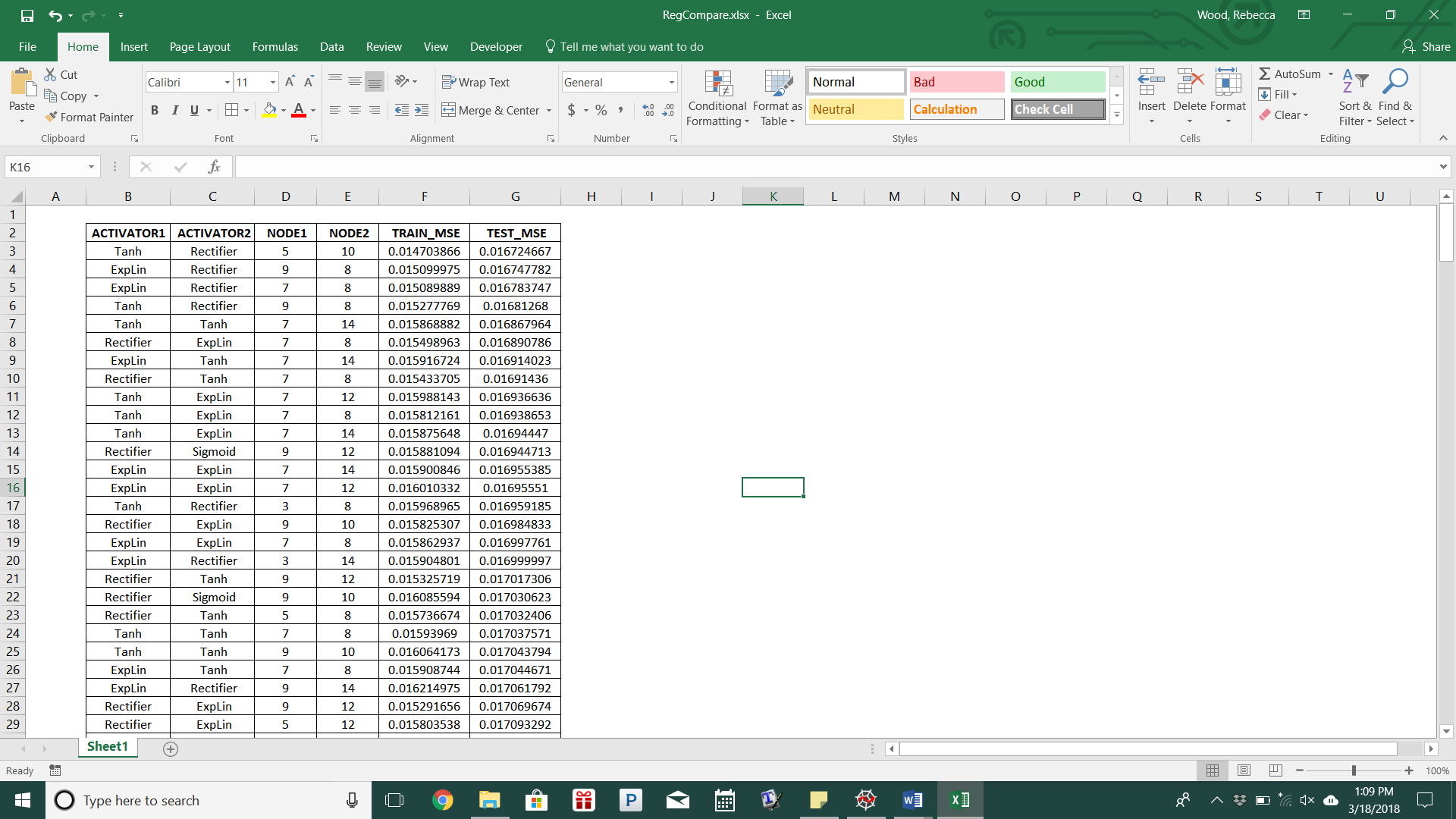
ANN Regressor Assignment – Red

# Experimentation

 After importing the data, the predictors and response columns were scaled using *MinMaxScaler()* from the sklearn package. Using a random state of 2016 and a test size of 33%, the predictors and response sets were split into training and test sets. The model was 2 hidden layers and one result layer. The algorithm filters through the activator list of ‘Rectifier’, ‘Sigmoid’, ‘Tanh’, and ‘ExpLin’ for each hidden layer, and the result layer is ‘Linear’ maintaining a learning rate of 0.2, random state of 2016, and number of iterations as 500. The first hidden layer uses units 3, 5, 7 or 9 and the second layer uses units 8, 10, 12, or 14. For each of these combinations, the training and test mean squared error are calculated and printed for an easy comparison.

At the conclusion, a table is printed with all combinations. After converting this table into an Excel spreadsheet, the combination with the lowest test mean squared error was used for the final model. From the sklearn.metrics package the *mean\_squared\_error()*, *explained\_variance\_score()*, *r2\_score()*, and *mean\_absolute\_error()* functions were used to calculate variance success measures.

Code

**”””PACKAGE IMPORT”””**

from sklearn import preprocessing

from sklearn.model\_selection import train\_test\_split

from sknn.mlp import Regressor, Layer

from sklearn.metrics import mean\_squared\_error, explained\_variance\_score, r2\_score, mean\_absolute\_error

import time

import numpy as np

import pandas as pd

**”””DATA IMPORT”””**

red\_data = pd.read\_csv("winequality-red.csv", encoding = 'utf-8')

x = red\_data.iloc[:, 0:11]

y = red\_data['quality']

**”””PRETREATING”””**

np.random.seed(2016)

x\_MinMax = preprocessing.MinMaxScaler()

y\_MinMax = preprocessing.MinMaxScaler()

x.as\_matrix(x)

x = np.array(x).reshape((len(x), 11))

y.as\_matrix(y)

y = np.array(y).reshape((len(y), 1))

x = x\_MinMax.fit\_transform(x)

y = y\_MinMax.fit\_transform(y)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.33, random\_state=2016)

x\_test.mean(axis=0)

y\_test.mean(axis=0)

x\_train.mean(axis=0)

y\_train.mean(axis=0)

**”””MODEL DEVELOPMENT”””**

start\_time = time.time()

li = ["Rectifier", "Sigmoid", "Tanh", "ExpLin"]

print("ACTIVATOR1", "ACTIVATOR2", "NODE1", "NODE2", "TRAIN\_MSE", "TEST\_MSE")

for activator in li:

for activator2 in li:

node\_li1 = [3,5,7,9]

node\_li2 = [8, 10, 12, 14]

for node1 in node\_li1:

for node2 in node\_li2:

fit\_trial = Regressor(layers = [

Layer(activator, units = node1),

Layer(activator2, units = node2),

Layer("Linear")],

learning\_rate = 0.02,

random\_state = 2016,

n\_iter = 100)

fit\_trial.fit(x\_train, y\_train)

train\_pred = fit\_trial.predict(x\_train)

train\_mse = mean\_squared\_error(train\_pred, y\_train)

test\_pred = fit\_trial.predict(x\_test)

test\_mse = mean\_squared\_error(test\_pred, y\_test)

print(activator, activator2, node1, node2, train\_mse,

test\_mse)

stop\_time = time.time()

print("Time Required for Optimization:", stop\_time - start\_time)

**”””FINAL MODEL”””**

final\_fit = Regressor(layers = [

Layer("Tanh", units = 5),

Layer("Rectifier", units = 10),

Layer("Linear")],

learning\_rate = 0.02,

random\_state = 2016,

n\_iter = 500)

final\_fit.fit(x\_train, y\_train)

final\_pred = final\_fit.predict(x\_test)

final\_mse = mean\_squared\_error(y\_test, final\_pred)

final\_var = explained\_variance\_score(y\_test, final\_pred)

final\_error = mean\_absolute\_error(y\_test, final\_pred)

final\_r2 = r2\_score(y\_test, final\_pred)

# Final Results

After the experimentation phase, the final model was completed using ‘Tanh’ as the first hidden layer with units set at 5 and ‘Rectifier’ as the second layer with units set at 10. The final mean squared error was calculated as 0.0167, the variance score was 0.3692, the mean absolute error was 0.09671, and the R-2 score was 0.36866. From these calculations we can claim that the predictions are significantly accurate, with such a low mean squared error.