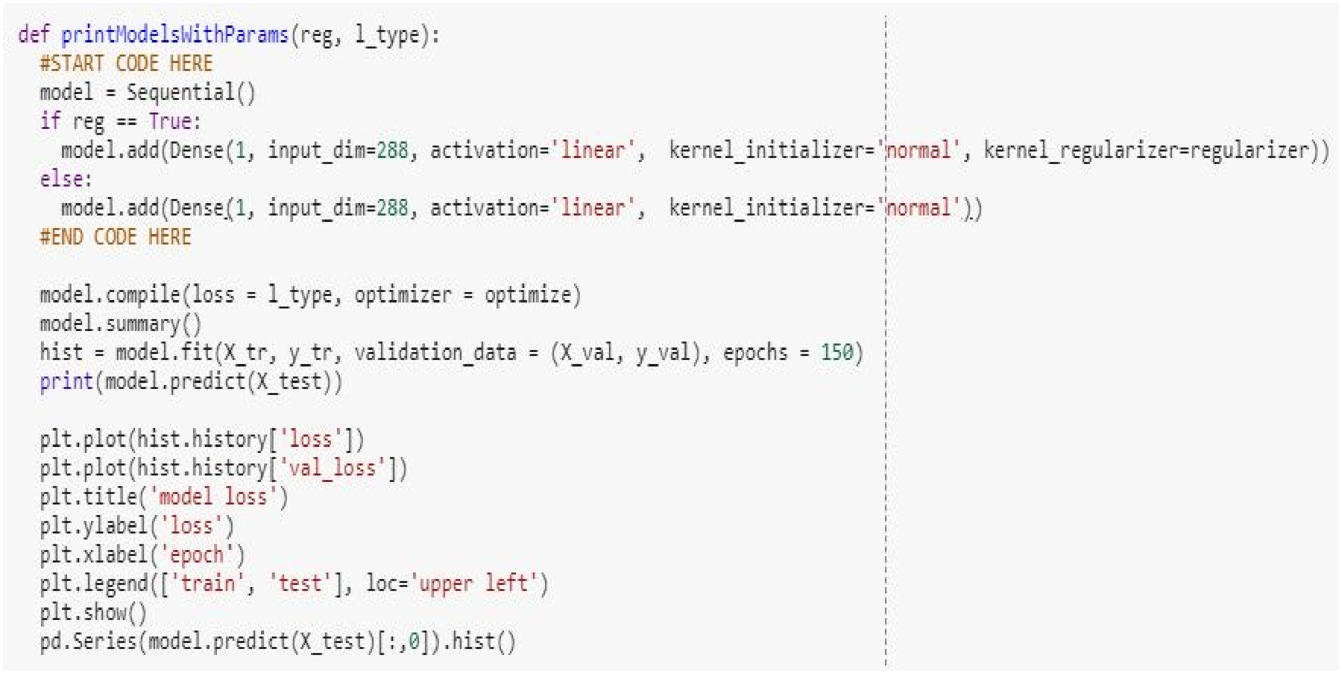
Lab 1 Deep Learning

Name: Michael Raafat Mikhail Zaki
ID: 57

Linear Regression

- **Building the model:**

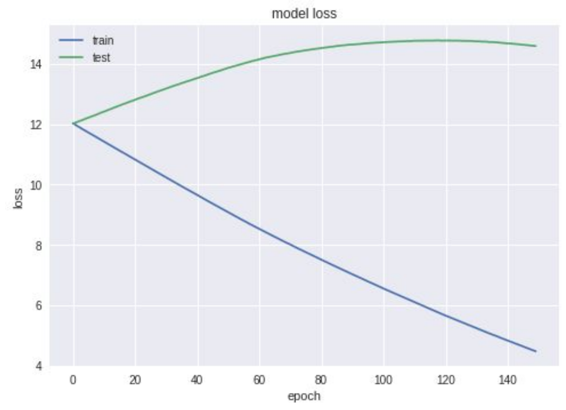


```
def printModelsWithParams(reg, l_type):
  #START CODE HERE
  model = Sequential()
  if reg == True:
    model.add(Dense(1, input_dim=288, activation='linear',  kernel_initializer='normal', kernel_regularizer=regularizer))
  else:
    model.add(Dense(1, input_dim=288, activation='linear',  kernel_initializer='normal'))
  #END CODE HERE

  model.compile(loss = l_type, optimizer = optimize)
  model.summary()
  hist = model.fit(X_tr, y_tr, validation_data = (X_val, y_val), epochs = 150)
  print(model.predict(X_test))

  plt.plot(hist.history['loss'])
  plt.plot(hist.history['val_loss'])
  plt.title('model loss')
  plt.ylabel('loss')
  plt.xlabel('epoch')
  plt.legend(['train', 'test'], loc='upper left')
  plt.show()
  pd.Series(model.predict(X_test)[:,0]).hist()
```

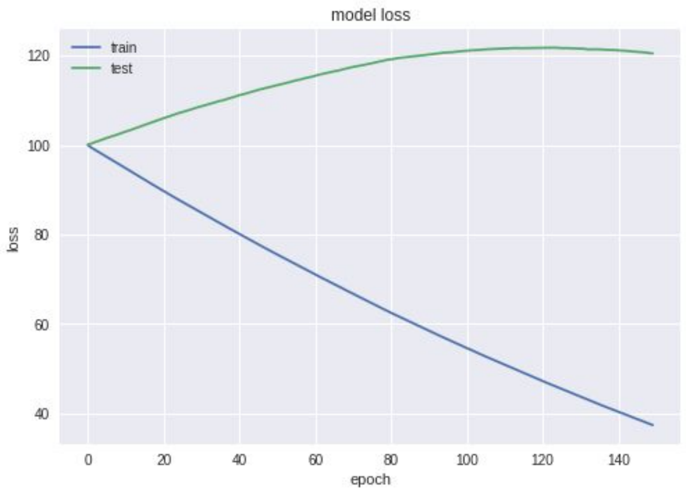
- **Hyper-Parameters:**
  - Regularization
  - Optimization

Note: we decided to fix epochs and random state and change regularization and optimization to reach a nice result.

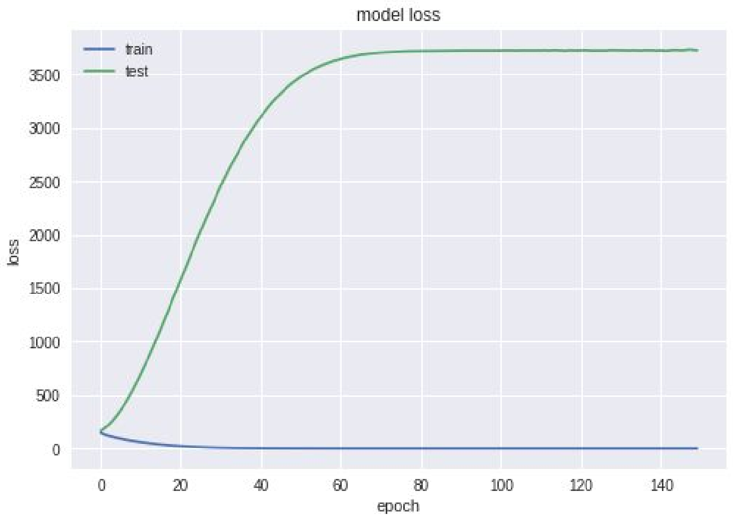
- We Used different loss functions without regularization and with Adam optimizer (default learning rate) :
  - **MAE:**



- **MAPE:**

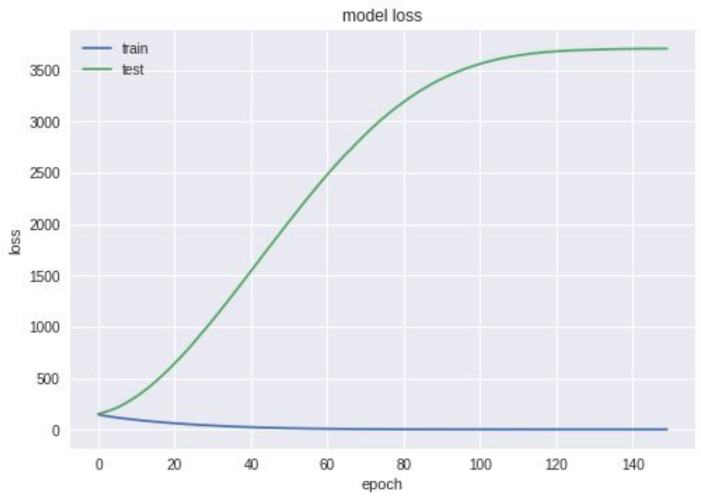


- **MSE:**

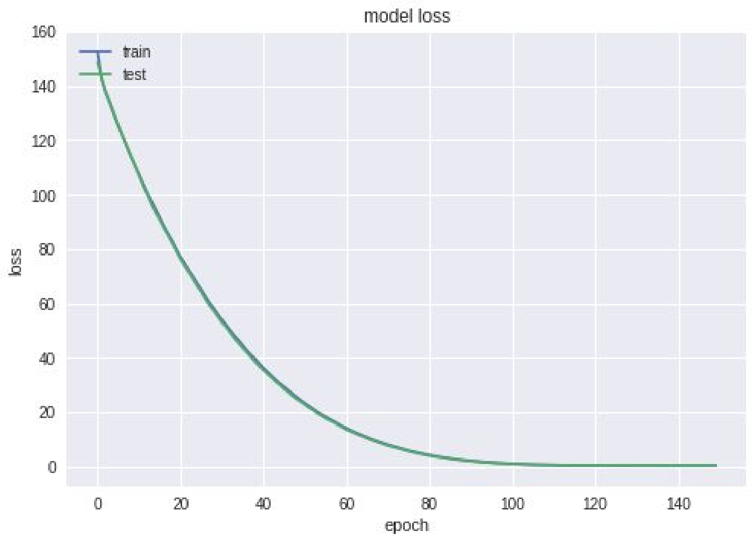


We can see that our model is overfitting without applying regularization using different loss functions, solving this problem by regularization and adjusting regularization term with tuning the learning rate of our optimizer.

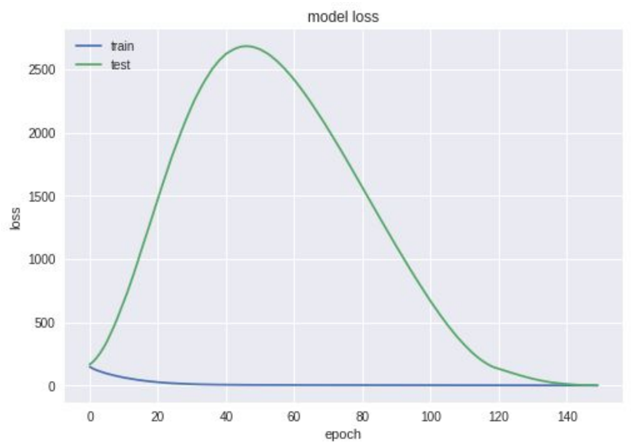
- **<u>This part we will show the results we got by changing our hyper-parameters :</u>**
    - Using Adam (lr= 0.005) , no Regularization
      loss: 0.0116 - val_loss: 3707.5138



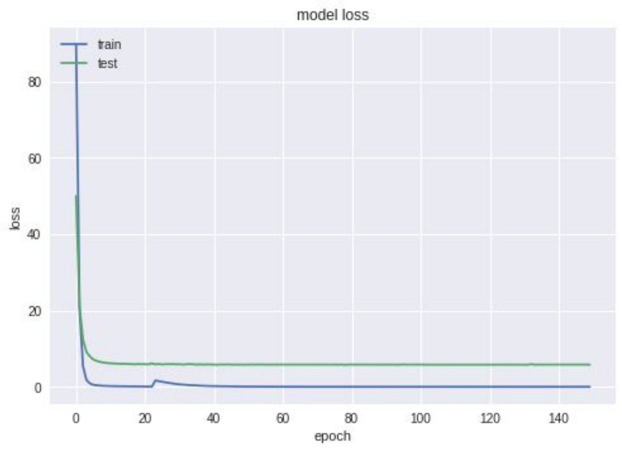
    - Using Adam (lr=0.005), Regularization L1(1)
      loss: 0.3397 - val_loss: 0.3201



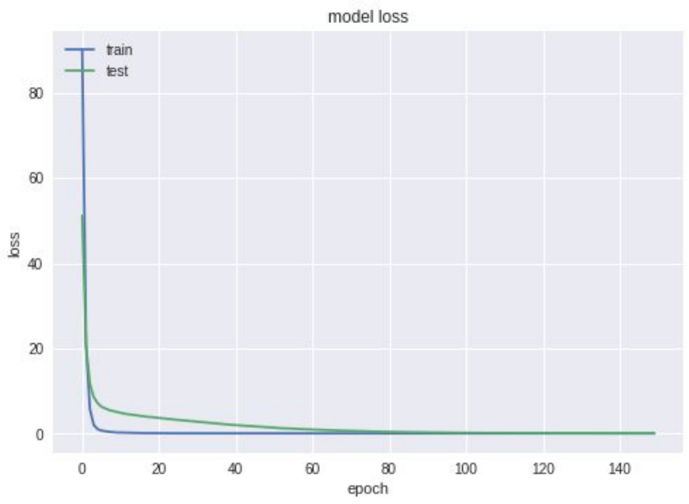
- Using Adam(lr=0.02), Regularization L1(0.02)
  loss: 0.0622 - val_loss: 0.0655



- Using SGD(lr = 0.01), no Regularization
  loss: 0.0108 - val_loss: 5.8230



- Using SGD(lr= 0.01), Regularization L1(0.01)
  loss: 0.0229 - val_loss: 0.0295



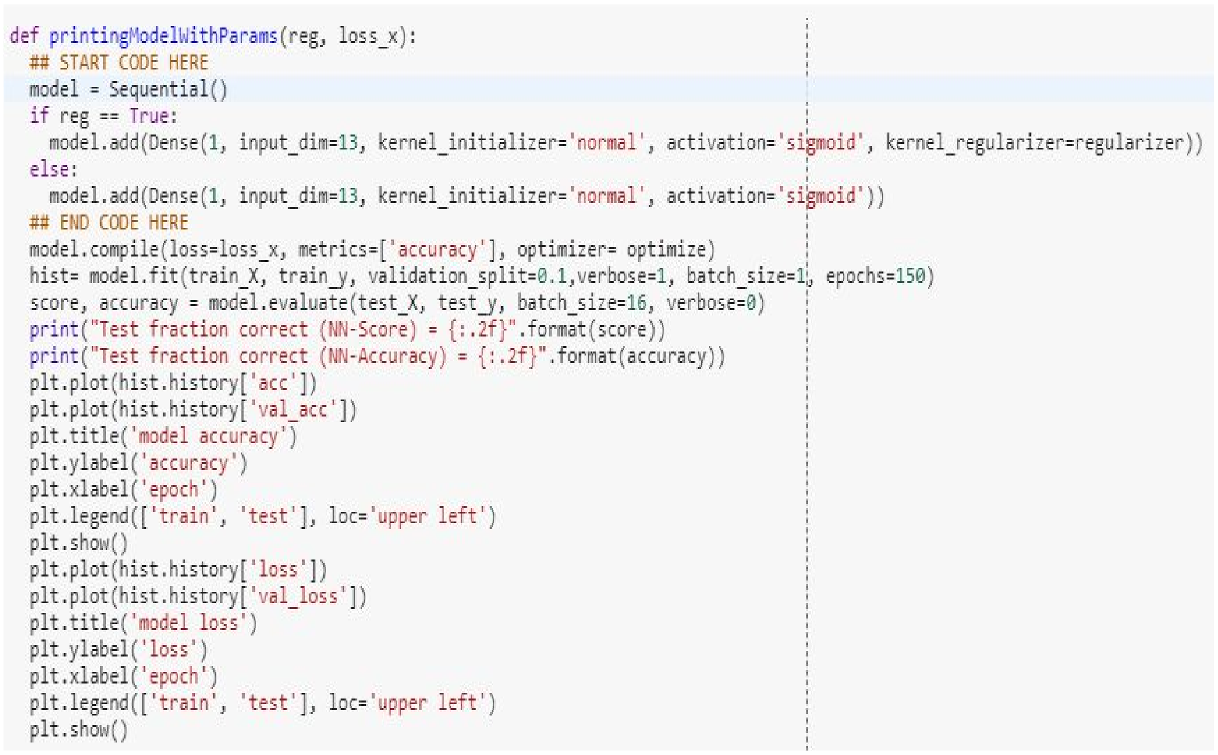
Note:

- SGD is better than Adam in performance with regularization L1.
- Both optimizers with no regularization will lead to an overfitting but SGD is better than Adam.

Logistic Regression

Applying the same criterion as in Linear regression

- **Building Model :**

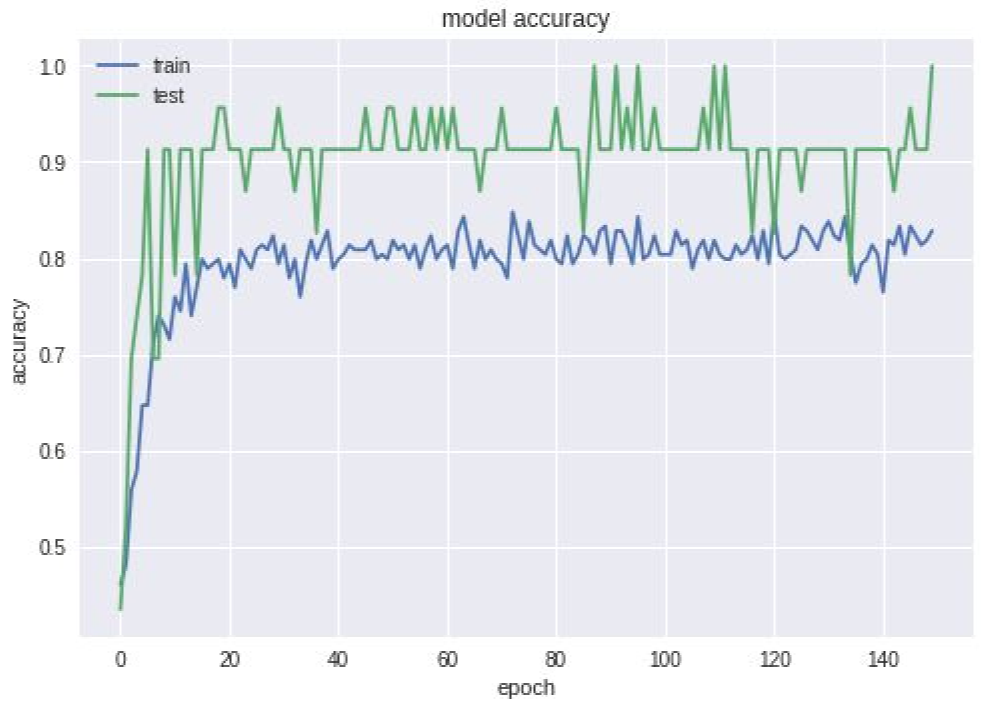


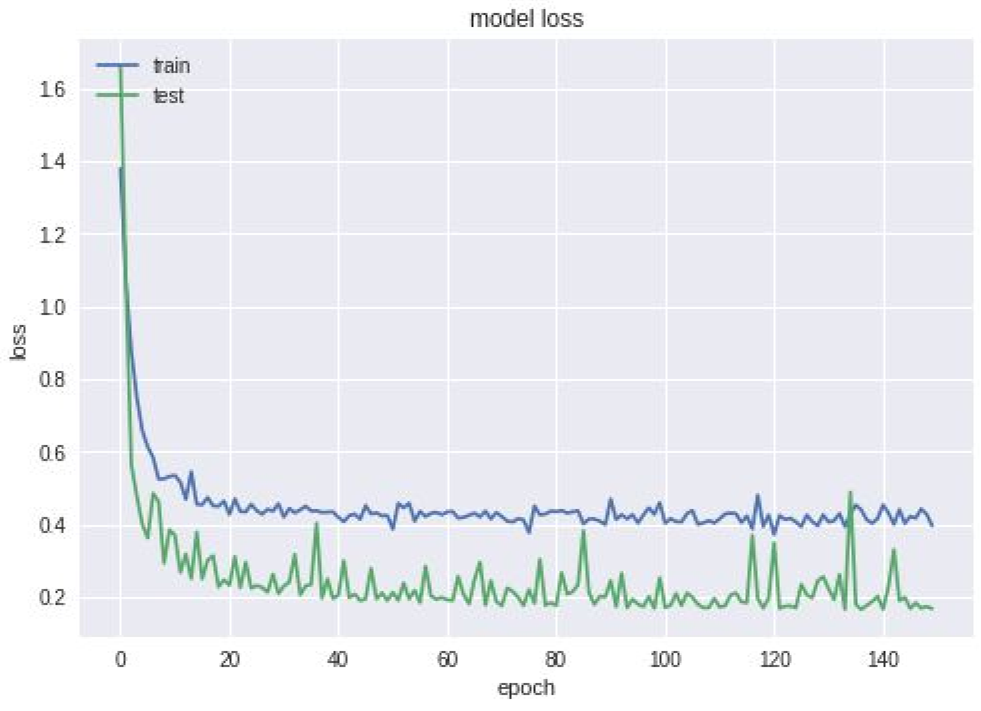
```
def printingModelWithParams(reg, loss_x):
  ## START CODE HERE
  model = Sequential()
  if reg == True:
    model.add(Dense(1, input_dim=13, kernel_initializer='normal', activation='sigmoid', kernel_regularizer=regularizer))
  else:
    model.add(Dense(1, input_dim=13, kernel_initializer='normal', activation='sigmoid'))
  ## END CODE HERE
  model.compile(loss=loss_x, metrics=['accuracy'], optimizer= optimize)
  hist= model.fit(train_X, train_y, validation_split=0.1,verbose=1, batch_size=1, epochs=150)
  score, accuracy = model.evaluate(test_X, test_y, batch_size=16, verbose=0)
  print("Test fraction correct (NN-Score) = {:.2f}".format(score))
  print("Test fraction correct (NN-Accuracy) = {:.2f}".format(accuracy))
  plt.plot(hist.history['acc'])
  plt.plot(hist.history['val_acc'])
  plt.title('model accuracy')
  plt.ylabel('accuracy')
  plt.xlabel('epoch')
  plt.legend(['train', 'test'], loc='upper left')
  plt.show()
  plt.plot(hist.history['loss'])
  plt.plot(hist.history['val_loss'])
  plt.title('model loss')
  plt.ylabel('loss')
  plt.xlabel('epoch')
  plt.legend(['train', 'test'], loc='upper left')
  plt.show()
```

Note: we take 0.1 for validation set to know our accuracy.

- Using Adam, no Regularization

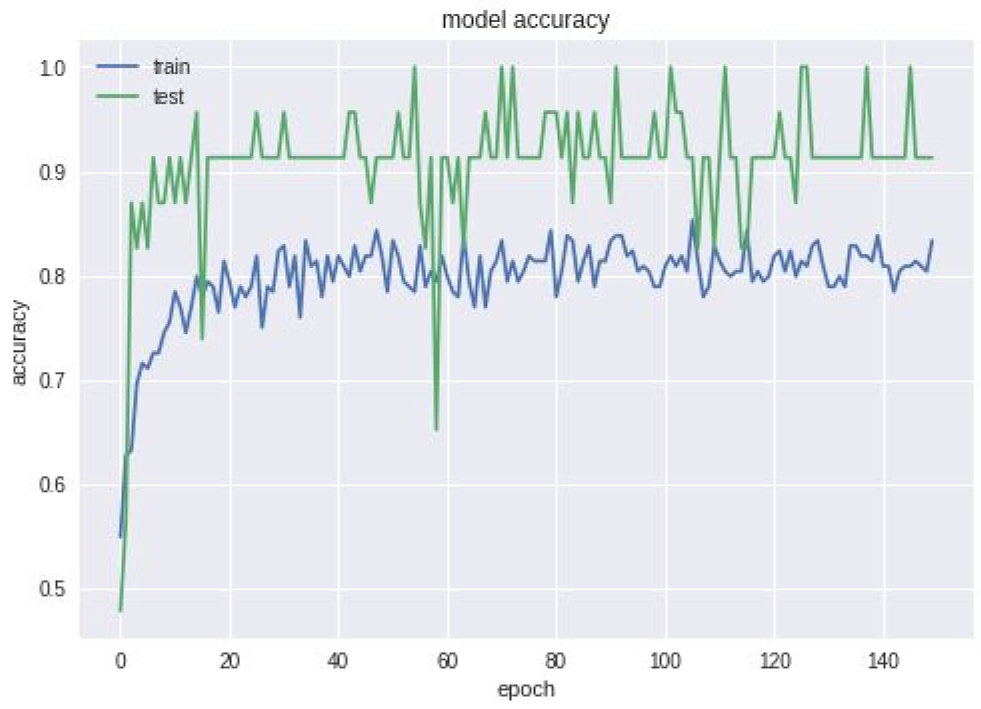
```
loss: 0.3961 - acc: 0.8284 - val_loss: 0.1673 - val_acc: 1.0000
```

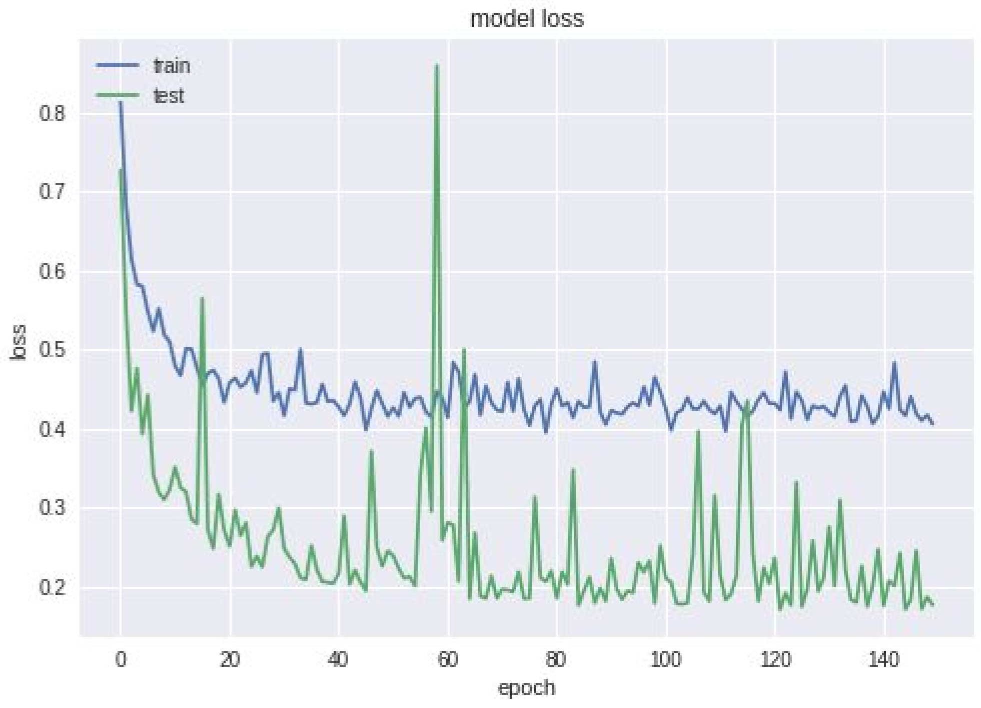




- Using Adam, Regularization L1(0.001)

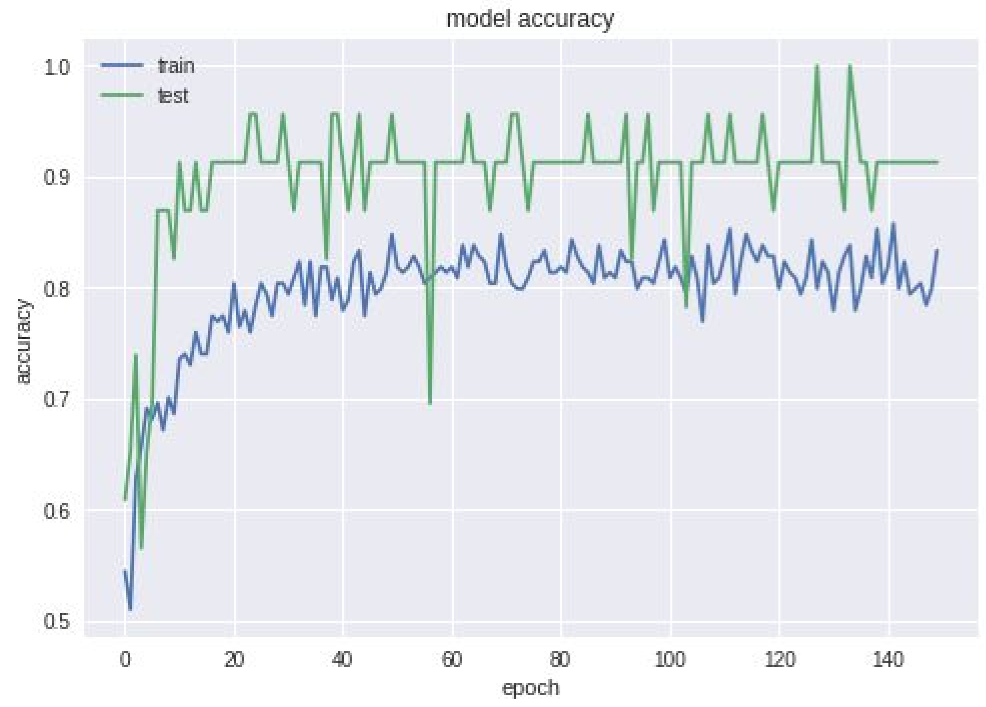
```
loss: 0.4052 - acc: 0.8333 - val_loss: 0.1761 - val_acc: 0.9130
```

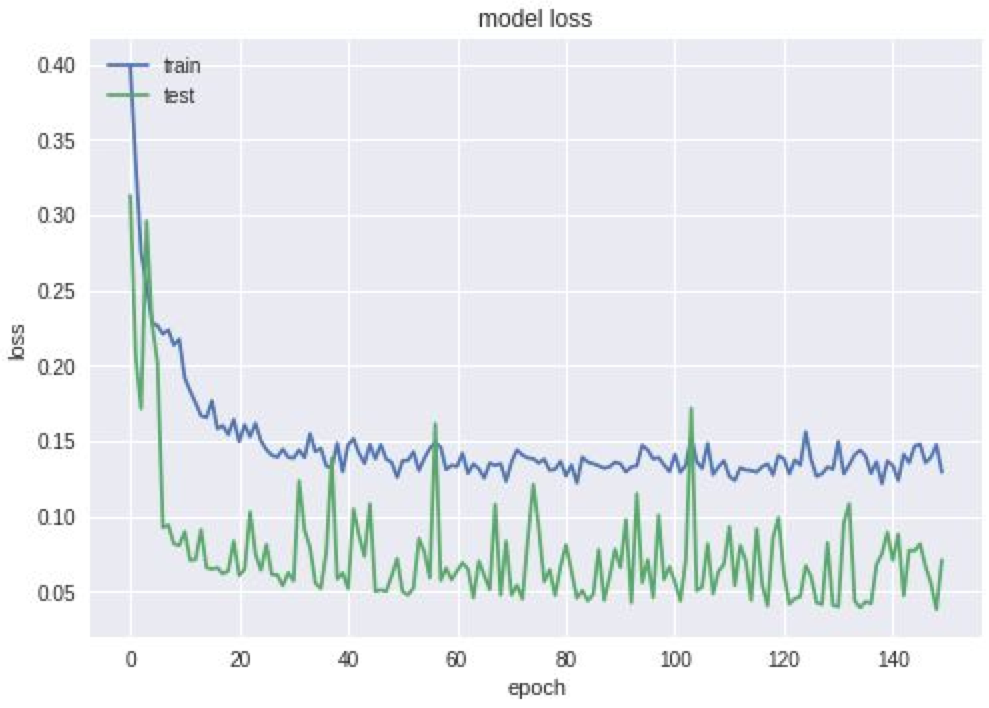




- Using Adam, no Regularization, loss function = “mse”

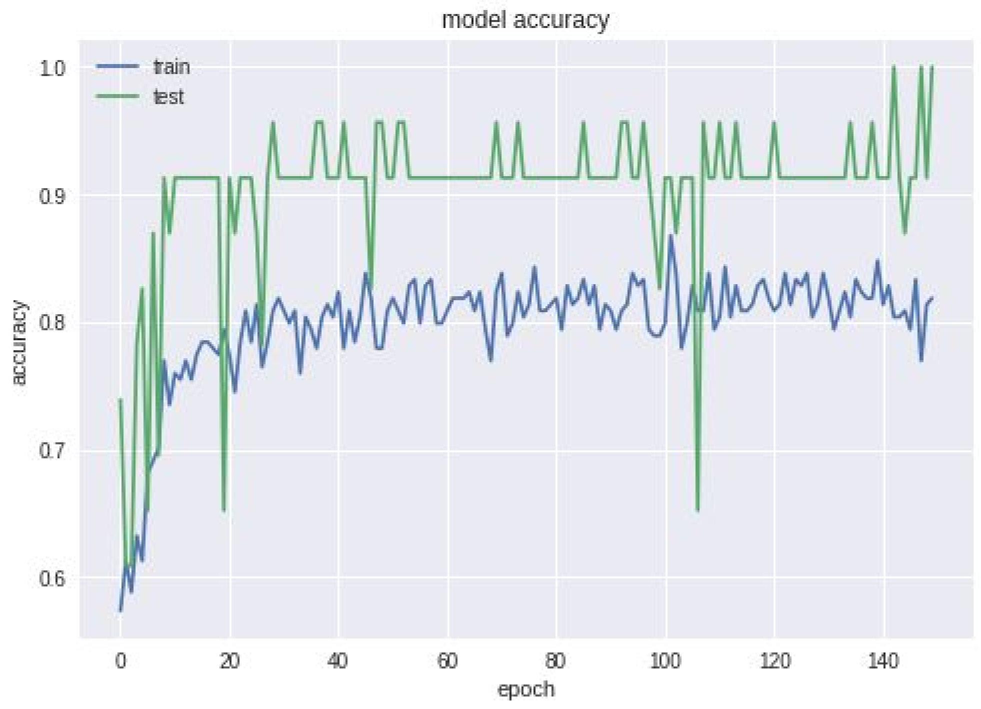
```
loss: 0.1291 - acc: 0.8333 - val_loss: 0.0708 - val_acc: 0.9130
```

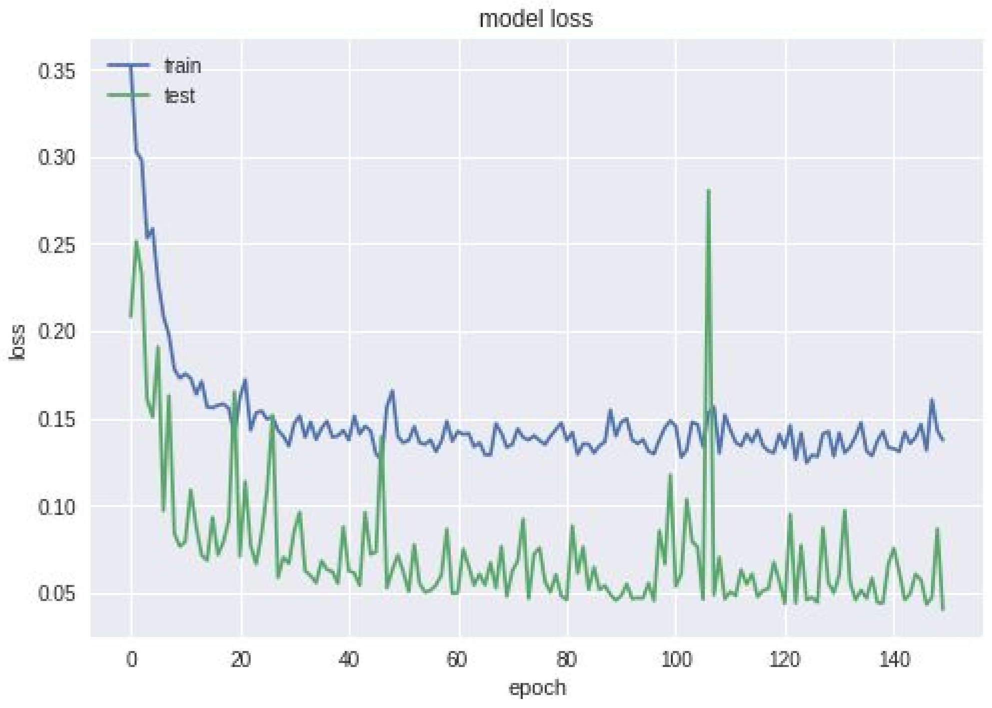




- Using Adam, Regularization L1(0.0004), loss function = “mse”

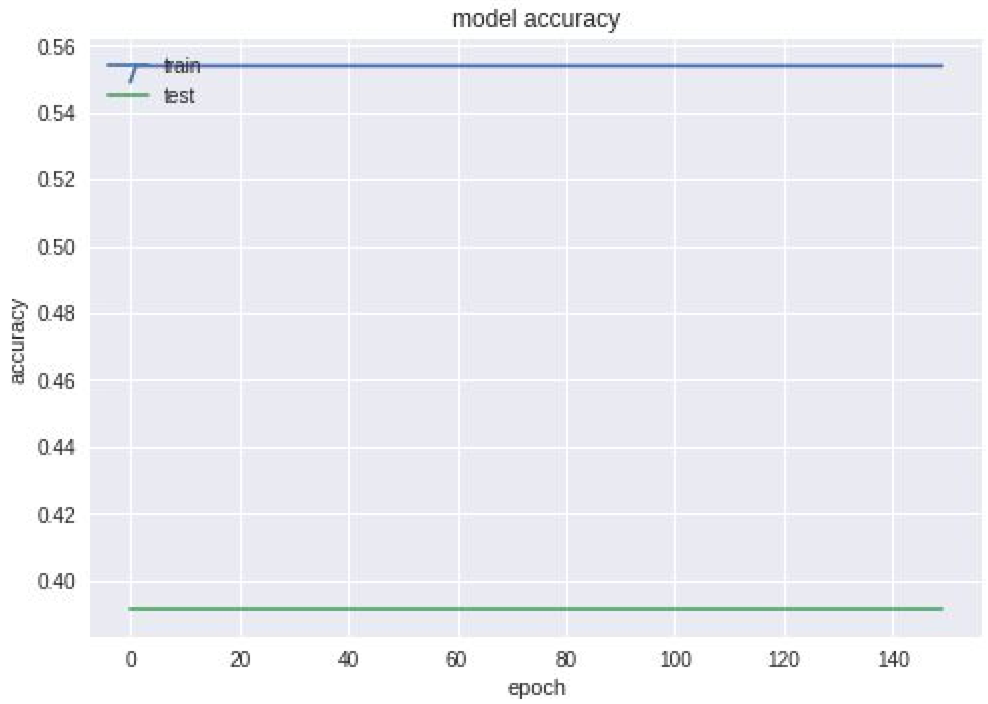
```
loss: 0.1375 - acc: 0.8186 - val_loss: 0.0405 - val_acc: 1.0000
```

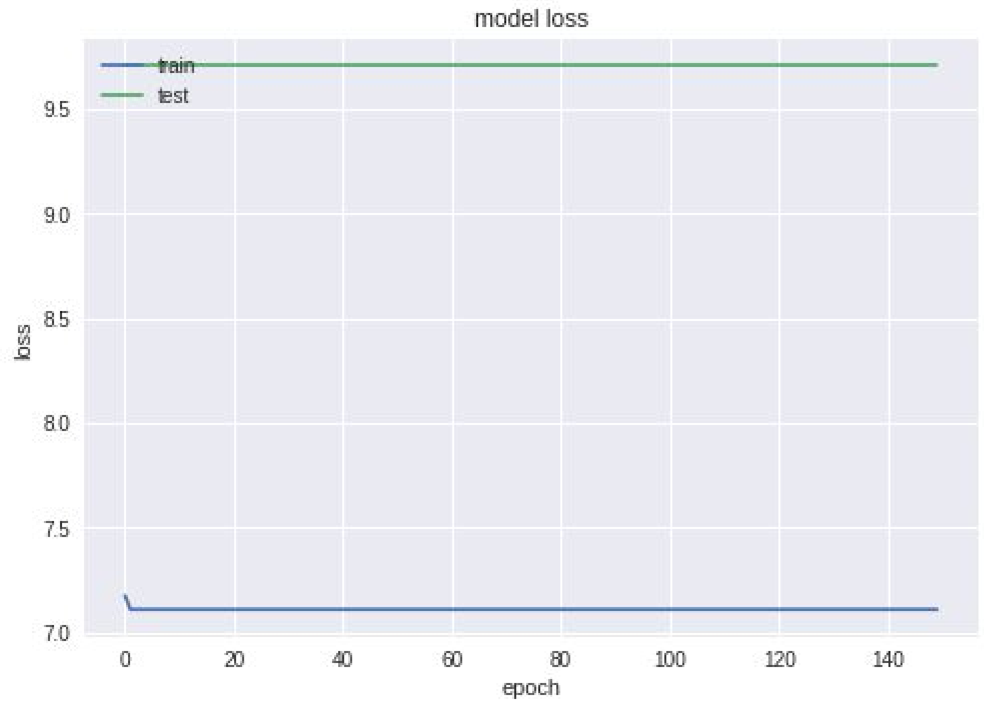




- Using SGD, no Regularization

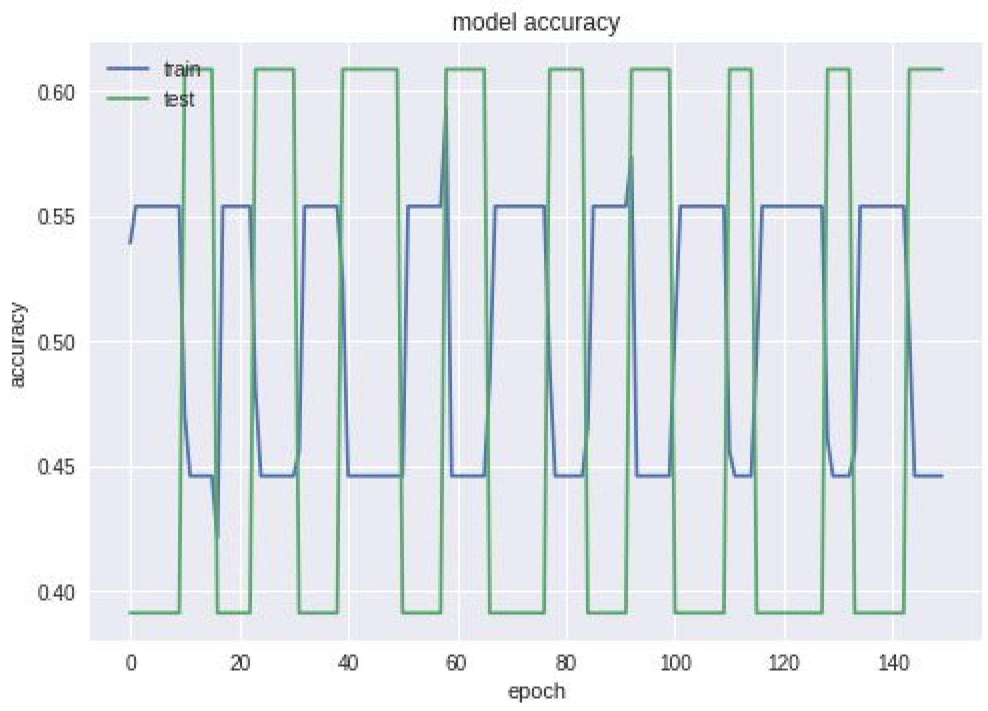
loss: 7.1116 - acc: 0.5539 - val_loss: 9.7041 - val_acc: 0.3913

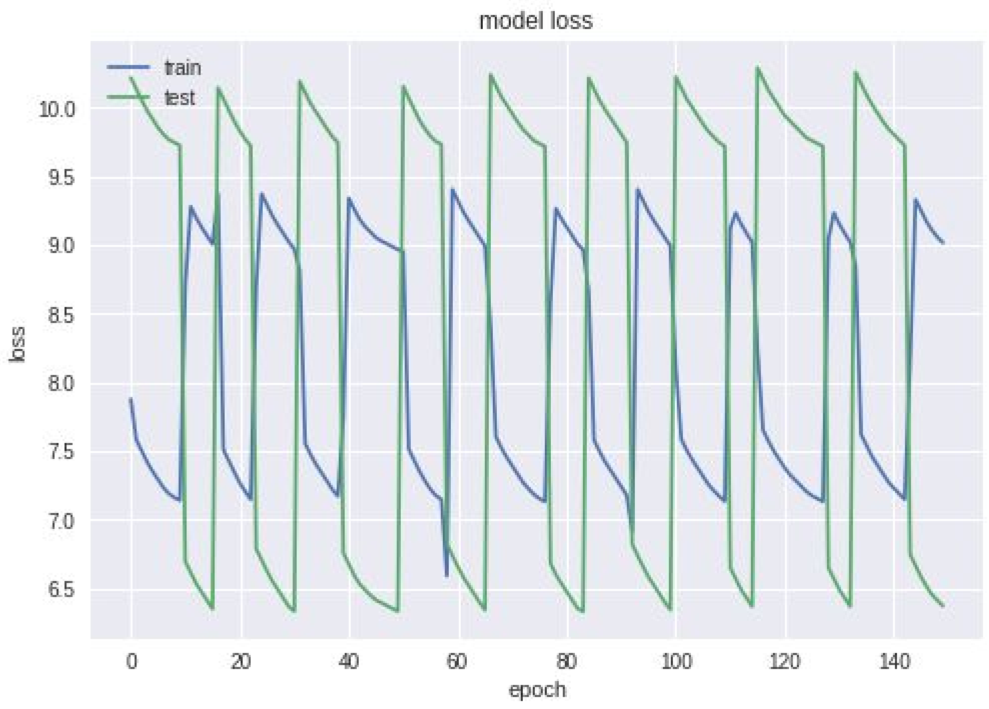




- Using SGD, Regularization L1(0.1)

```
loss: 9.0166 - acc: 0.4461 - val_loss: 6.3751 - val_acc: 0.6087
```





we can see here that Adam is better in performance.