Neural Network

July 18, 2019

In [1]:
```
from keras.models import Sequential
from keras.layers import Dense
import numpy as np
import time
import csv
import os

from sklearn import preprocessing

from sklearn.preprocessing import  MinMaxScaler
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
```

```
/home/shashwati/anaconda3/envs/py35/lib/python3.5/site-packages/h5py/__init__.py:36: FutureWarn
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
```

In [2]:
```
# load training dataset
dataset = np.genfromtxt("YearPredictionMSD.csv", dtype='float', delimiter=",")
```

In [3]:
```
# split into input (X) and output (Y) variables and preprocessing

train_X = dataset[0:463715,1:91]
# Scale the features so they have 0 mean
train_X = preprocessing.scale(train_X)

sc= MinMaxScaler()
train_Y = dataset[0:463715,0]
train_Y = train_Y.reshape(-1,1)
train_Y = sc.fit_transform(train_Y)

test_X = dataset[463715:,1:91]
test_X = preprocessing.scale(test_X)

test_Y = dataset[463715:,0]
test_Y = test_Y.reshape(-1,1)
test_Y = sc.fit_transform(test_Y)
```

In [4]:
```
# training the neural network model
start = time.time()

model = Sequential()
model.add(Dense(90, input_dim=90, kernel_initializer='normal', activation='relu'))
model.add(Dense(90, kernel_initializer='normal', activation='relu'))
model.add(Dense(90, kernel_initializer='normal', activation='relu'))
model.add(Dense(90, kernel_initializer='normal', activation='relu'))
model.add(Dense(1, kernel_initializer='normal', activation='linear'))
model.compile(loss='mean_squared_error', optimizer='adam', metrics=['mae','accuracy'])
model.fit(train_X, train_Y, epochs=3, batch_size=32)

end = time.time()
```

```
Epoch 1/3
463715/463715 [==============================] - 70s 150us/step - loss: 0.0113 - mean_absolute_
Epoch 2/3
463715/463715 [==============================] - 70s 150us/step - loss: 0.0098 - mean_absolute_
Epoch 3/3
463715/463715 [==============================] - 70s 151us/step - loss: 0.0096 - mean_absolute_
```

In [5]:
```
# training time
print("Training time: ", round(end - start, 5))
```

```
Training time:  209.6338
```

In [6]:
```
# predicting for test data
# results scaled between 0 - 1
pred = model.predict(test_X)
pred = pred.ravel()
pred = pred.reshape(-1,1)

# getting the the original values
label_test = sc.inverse_transform(test_Y)
pred = sc.inverse_transform(pred)

print(mean_absolute_error(label_test,  pred), mean_squared_error(label_test, pred))
```

```
6.080521999420457 78.29599705086922
```

In [7]:
```
print(pred[0:10])
print(label_test[0:10])
```

```
[[2002.8302]
 [2002.4656]
 [2001.7264]
```

```
 [2004.5768]
 [2004.3024]
 [2000.4031]
 [2002.8789]
 [2001.239 ]
 [1997.0228]
 [2001.8604]]
[[2007.]
 [2003.]
 [2005.]
 [2003.]
 [2005.]
 [2007.]
 [2003.]
 [2003.]
 [2003.]
 [2005.]]
```