This is my capstone project for the Udacity Machine Learning Nanodegree.

Import the libraries needed.

In [1]:
```
import pandas as pd
import numpy as np
import keras as kr
import tensorflow as tf
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
from os import listdir
```

```
Using TensorFlow backend.
```

Get the data.

Since we already know the name of the specific stock we are trying to get from the name of the file, we can drop that column in the dataframe.

In [2]:
```
directory = 'sandp500/individual_stocks_5yr'
dir_listing = listdir(directory)
```

In [3]:
```
symbols_list = []

for symbol in dir_listing:
    symb = symbol.split('_')[0]
    symbols_list.append(symb)

csv_file = '{}/{}_data.csv'.format(directory, symbols_list[0])
dataset = pd.read_csv(csv_file)
```

Normalize Stock Prices

In [4]:
```
def getting_preprocessed_data(symbol):
        csv_file = '{}/{}_data.csv'.format(directory, symbol)
        df = pd.read_csv(csv_file)
        df = df.assign(trading_date = pd.to_datetime(df['Date']))
        df = df.drop('Name', 1)
        df = df.drop('Date', 1)
        df.set_index(['trading_date'], inplace=True)
        #df = np.log(df)
        #below was found at https://stackoverflow.com/questions/29314033/pytho
n-pandas-dataframe-remove-empty-cells
        df.replace('', np.nan, inplace=True)
        df.dropna(subset=['Open'], inplace=True)
        return df
```

In [5]:
```
display(getting_preprocessed_data('AAPL').head(5))
```

	Open	High	Low	Close	Volume
trading_date					
2012-08-13	89.06	90.00	89.04	90.00	69707463
2012-08-14	90.27	91.23	90.03	90.24	85041824
2012-08-15	90.19	90.57	89.68	90.12	64377278
2012-08-16	90.17	90.97	90.07	90.91	63694204
2012-08-17	91.43	92.60	91.26	92.59	110689894

In [6]:
```
data = getting_preprocessed_data('AAPL')
```

In [7]:
```
from sklearn.preprocessing import StandardScaler
```

In [8]:
```
scaler = StandardScaler()
print(scaler.fit(data))
```

```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

In [9]:
```
print(scaler.mean_)
```

```
[  1.01053164e+02   1.01910930e+02   1.00148633e+02   1.01041208e+02
   6.47195915e+07]
```

In [10]:
```
type(data)
```

Out[10]:
```
pandas.core.frame.DataFrame
```

In [11]:
```
max_values = data.max()
```

```
In [12]: max_values
```

```
Out[12]: Open      1.599000e+02
         High      1.618300e+02
         Low       1.591100e+02
         Close     1.610600e+02
         Volume    3.652130e+08
         dtype: float64
```

```
In [13]: max_values[1]
```

```
Out[13]: 161.83000000000001
```

```
In [ ]:
```