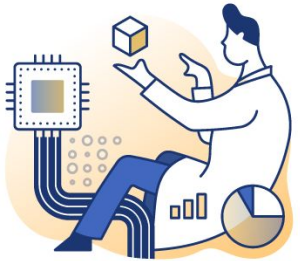
Predicting the stock prices of Ford using AI

Lukas Panos

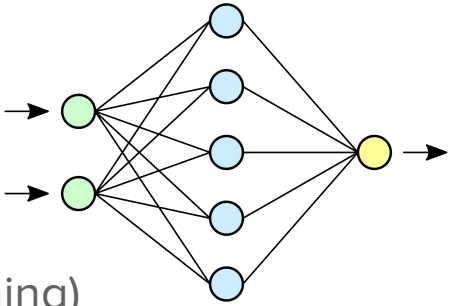
Why?

- Curiosity for the financial market and for computer science
- Future studies/career
- Learned python in 11th grade
- Language used for data manipulation
- Introduction to AI models



Research

- How to import libraries
- Pandas-Numpy-Matplotlib etc...
- Manipulation of arrays and creation of models
- Which company used (Tesla)
- How to manipulate historical data
- How to create visual templates
- Learned a lot about “LSTM” Neural networks (deep learning)



First stages of conception

```
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dropout
from keras.layers import Dense

model = Sequential()
```

- Import the main libraries
- Install on terminal
- Numpy, matplotlib, pandas, sklearn processing
- Difficult to install libraries are keras and Tensorflow
- Consequently, a lot of time spent in research
- Develop my online research/web use skills
- Very useful/essential for a software developer
- Decided to continue with the code without having tf.keras installed
- Only use Keras for models and layers
- Keras alone is slower and less advanced than tf.keras (I based my code on that)
-

Data importation

- Lots of difficulty
- Tiingo, yahoo, pandas_datareader etc...
- API requests
- Used a library with an API already in the system
- ‘Yfinance’
- Used for training and testing data
- How many days training+test

```
import yfinance as yf
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
ticker= yf.Ticker('TSLA')
dataset_train =ticker.history(start='2014-8-10',end='2023-4-30')
training_set = dataset_train.iloc[:, 1:2].values
```

Data treatment

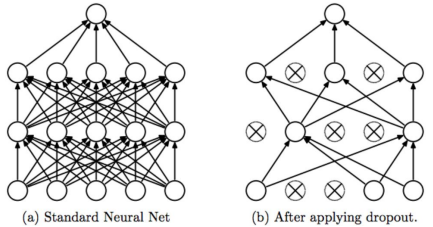
- Normalization (change the scale) of data (MinMax scaler)
- Feature range, fit transform, shape
- Between 0 and 1 to better be able to manipulate data
- Then create 3D arrays with the data
- Define array perimeters
- 16 groups
- For training and testing data
- After LSTM rescale the data to the original scale

```
X_train = []
y_train = []
for i in range(60, 2035):
    X_train.append(training_set_scaled[i-60:i, 0])
    y_train.append(training_set_scaled[i, 0])
X_train, y_train = np.array(X_train), np.array(y_train)

X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
```

LSTM

```
model.add(LSTM(units=50,return_sequences=True,input_shape=(X_train.shape[1], 1)))
model.add(Dropout(0.2))

model.add(LSTM(units=50,return_sequences=True))
model.add(Dropout(0.2))

model.add(LSTM(units=50,return_sequences=True))
model.add(Dropout(0.2))

model.add(LSTM(units=50))
model.add(Dropout(0.2))

model.add(Dense(units=1))
```

- The most important step
- These are the best versions of “recurrent neural networks”
- Recurrent neural networks is an algorithm that imitates the human brain
- LSTMs can remember information over long periods of time
- Allows me to use longer data
- The more layers the more sophisticated but computer faster
- Dropout layers help prevent the program from being overtrained or undertrained
- It will randomly remove neurons
- Over trained = too complex of a model
- Under trained = too simple of a model
-



(a) Standard Neural Net (b) After applying dropout.

The creation of the LSTM

- 90% training 10% test
- 4 Layers
- Each of the dropout layers
- Dense allows the sharing of information between groups of neurons
- Compile gathers pieces of information
- Fit will enter the information in the tables
- Batch size = number of iterations per epoch
- Smallest batch size, the most iterations (32 is a lot of iterations)
- Epochs = number of times training data used to predict test data
- 100 is pretty standard

```
model.compile(optimizer='adam',loss='mean_squared_error')

model.fit(X_train,y_train,epochs=100,batch_size=32)
```

The prediction

- Used .predict
- Used .inverse_transform

```
predicted_stock_price = model.predict(X_test)
predicted_stock_price = sc.inverse_transform(predicted_stock_price)
predicted_stock_price=predicted_stock_price
```

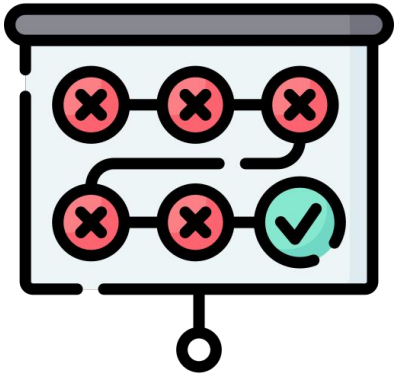
Graphing

- The simplest part of the code
- Learned how to use matplotlib
- Very often used by “data scientists” to model data
- Very useful

```
plt.plot(real_stock_price, color = 'black', label = 'Prix de bourse de Ford ($)')
plt.plot(predicted_stock_price, color = 'green', label = 'Prix prédit de Ford ($)
plt.title('Prix de bourse prédit de Ford ($)')
plt.xlabel('Temps (Jours)')
plt.ylabel('Prix de bourse de Ford ($)')
plt.legend()
plt.show()
```

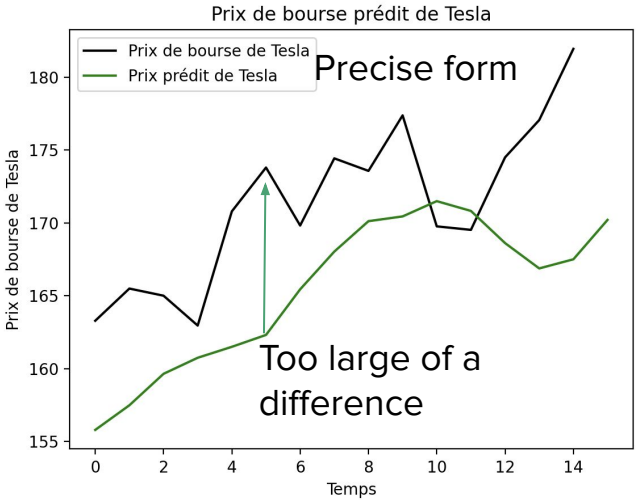
Trials,errors and resolution

- Did a lot of testing
- Test data was out of scope of array dimensions
- Tensorflow (took a lot of time) ➜ Keras
- Yahoo finance/tiingo ➜ y.finance
- Data was not scaled correctly ➜ Search sk.learning
- Not enough dropout layers ➜ found the right amount



Issues with my data set

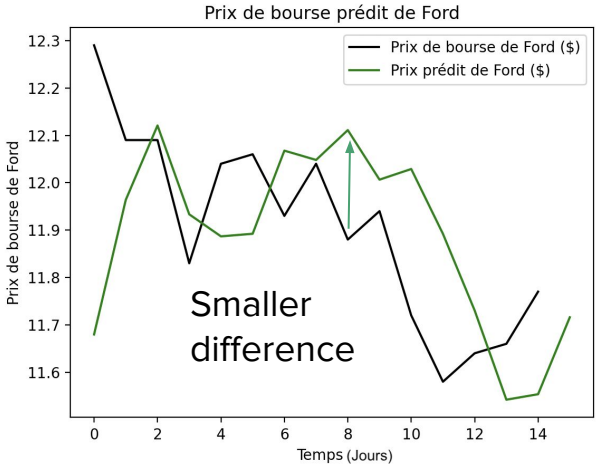
- Works only with specific companies
- Data must be constant
- Min 10 years of constant data - Not Tesla
- Tesla then did not work because the values produced were too low
- Ford constant last 10 years



Results and conclusion

- Take samples of the data I gave
- Each prediction is different because the LSTM takes different samples (shares the information differently)
- Average Ford better than Tesla
- Good final direction but still a significant gap

Encore une forme précise



Limits

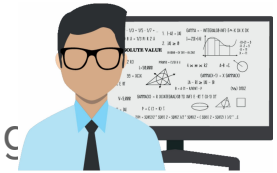
- Use of data already present ➜ Unavoidable bias of the program
- Can't predict the future, only days that have already passed
- Unreliable, financial market is unpredictable
- Do not take into account idiosyncratic events in the market
- Only companies that have been consistent in pricing for over 10 years

Strong points

- Program correctly indicates the direction (growth-decline) of the market
- Pushed me to learn the essential basics of “Machine learning”
- Lots of Data Science Concepts
- Learned tenacity/perseverance
- I was able to run the program
- Hugely improved and worked my problem-solving skills

Link to my future career

- Keras/tensorflow are libraries used by computer scientist working with AI
- Already have this knowledge when I apply for a job
- Most people learn machine learning on the job
- Allows you to create personal projects using “Machine Learning”
- Sk.learning, matplotlib, numpy etc.
- Manipulation of arrays➜ essential to any work related to computer eng
- How to do research effectively on stackoverflow, quora, github etc...
- Perseverance that I learned➜ the heart of coding/computer engineering
- Motivates me a lot to want to pursue this career
-



Work cited

- Smruti, & Smruti. (2023). What is LSTM ? Introduction to Long Short Term Memory. *Intellipaat Blog*. https://intellipaat.com/blog/what-is-lstm/?US#:~:text=LSTM%20Explained,-Now%2C%20let%27s%20understand&text=%27%20LSTM%20stands%20for%20long%20short,especially%20in%20sequence%20prediction%20problems.
- Team, K. (s. d.). Keras : Deep Learning for humans. https://keras.io/
- Rea, L. (2022, 18 novembre). Predicting Future Stock Market Trends with Python & Machine Learning. *Medium*. https://towardsdatascience.com/predicting-future-stock-market-trends-with-python-machine-learning-2bf3f1633b3c
- *Matplotlib — Visualization with Python*. (s. d.). https://matplotlib.org/
- Brownlee, J. (2020). How to Use StandardScaler and MinMaxScaler Transforms in Python. *MachineLearningMastery.com*. https://machinelearningmastery.com/standardscaler-and-minmaxscaler-transforms-in-python/
- *NumPy*. (s. d.). https://numpy.org/