**Q: What is this?**

**A: The specifications for the overall program. The recipe.**

**Overall Blueprint:**

*This is a very general overall outline of what needs to be done.*

Python: Grab data from APIs, clean data, store it into SQL. The DB should be hosted but if not, SQLite works too.

Python/Pandas: Rescale the data.

Python or Jupyter Notebook (Keras): Feed data into AI. AI will then poop out results.

Put results into DB.

Python: Run Flask server, create API that allows access to AI results and host website on Heroku or AWS or google cloud. Honestly, if there’s time just learn to port Flask over to Django.

Javascript and HTML: Retrieve AI results and visually show the results.

**More detailed, specific instructions:**

Each of the steps below many require data from the previous step. If that’s the case then just use fake data until the real data works.

The steps below, as you will see, are purposefully designed to take data frames of arbitrary column length. Each of the parts should be able to work with any amount of columns with the exception of the first column being the date and the last column being the training data (to be added in a later step).

**Data Collection:**

Find stock market data since 1980s or 1990s (CSV, API, whatever) for the big Indexes (Dow Jones, and S&P 500). Daily data would be the best. Daily would be the one we use for this project but hourly can be something we use later.

Clean the data so that we can have at least have the following information for each day (Using numbers from MSFT as an example):

Open 141.01

High 141.65

Low 138.25

(IF these three points of data above is not available, then just the closing price or whatever price IS available, is fine)

Mkt cap 1.06T

P/E ratio 27.54

Div yield 1.46%

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Date | Open | High | Low | MktCap | P/E Ratio | Div Yield | Daily Volume |
| 01/01/2001 | 78 | 90 | 20 | 9393 | 32 | 2 | 100000 |

In the pandas, each row should look something like this, above.

Clean the data put it into, make sure they’re all in numbers/float format. Replace all weird data like “NaN” or “N/A” or whatever, with 0.

Store this data into SQLite.

In addition to the raw data, there should also be moving averages and historical data for data from 11 days ago.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Yesterday’s price | 3-Days ago | 5-Days ago | 7-Days ago | 9-days ago | 11-days ago |
| 50 | 32 | 63 | 83 | 29 | 50 |

This would be appended/merged to the first.

**Data Scaling:**

This data must then be compressed into a format that the neural network can process. This means all data should be turned into some number between 0 to 1.

For example, if we are look at data from 9/3/2015 and the price was 100, we will find the maximum and minimum price from the last 365 days to determine the range of the scale.

Our current price: 100

Max price from 365d range: 150

Min price from 365d range: 50

Our **scaled price data**: 0.5

*Optional but highly recommended*: To account for outliers, do not accept data that exceeds 3 standard deviations of the 365d data set.

*What to do:*

Create Jupyter notebook code or python code that can:

1 )Grab the data from SQLite and put it into a dataframe.

2) Create a function in python that can take a data frame of unknown columns and process all the columns in some way. Except the first column which will be the date, all other columns should already be in numerical/float format.

3) Scale the data by taking the min and max of data 365 days before the row being processed. See above for example.

4) Saved the scaled data into the SQLite database.

**Generating training data:**

Random ramblings:

*IF NO RNN: Once the data has been cleaned and scaled, it’s time to generate training data. We will need to decide what is the “range” of data we will want the AI to consider. Should the AI look at what happened the last day, last week or last month?*

*This must be decided on how much time we’re giving the AI to think. The idea is “Would it be a good idea to buy or sell right now if the next opportunity we have to buy or sell will be a day, week, or month from now?”*

*For this project, we will just start with 11 day range. In the future, we may have separate AI’s that use 1 month data, 1 day data, 1 year data, etc.*

**Instructions:**

Python/Jupyter Notebook: Create a function that that will take a dataframe of unknown columns and size, and add a new column to it. The new column will look at the closing price of that day, and compare it to the closing price of a date 11 days in the future.

If the future date’s price is higher, then the row in the new column should be 1.

Caution: 11 days away may not exist due to the market closed on weekends. In that case, just look up the next available day. Or just count current index + 11. When that data no longer exists because you’re at the 10th date from the last day available in the data frame, just stop and splice off/throw away the rest of the data (the last 10 days).

**The Neural Net:**

The AI NN will be a standard feed forward NN that will be fed data from the previous task.

The neural net will be automatically scaling. The width, height, and depth (total number of layers), will be equal to the number of inputs/features available (which is the length of the number of columns available minus two because the first row is the date, the last row is the testing data column)

See example below:

= is an input cell

0 is a feed forward cell

+ is an output cell

=0000

=0000+

=0000+

=0000

=00+

=00+

=000+

=000+

=000

The AI will output its answer. Take this array/series data and add it to the last row of **of the original data**, the one that’s unscaled. Map everything in this last column into “buy” and “sell” as replacements for 1 and 0 respectively.

Append a new column to the column to represent the AI’s ID. And we do this because…

**Optional:** Redo the above but change the parameters of the NN by adding an extra layer or something and in the final data frame, change the ID.

**TODO:** figure out someway to add a description to each AI’s ID. Maybe a new table with two rows: AI ID, and a description of the type of data the AI was looking at, a description of its NN architecture.

Even better, if there’s enough time towards the end of the class, we should create new training data and scaled data where the AI looks at 31 days of data instead of 11 or only yesterday’s data.

**Front end:**

Create the standard file structure of flask. (app.py, static/js/css folders, basic template for routes)

Create the API routes that will be able to return the data from the server as JSON.

Things to show on the website (fee free to get creative here):

What was the average % gains or loss by the AI?

How much money would you end up with if you have given an AI $10,000 since 1980?

Show the performance differences between each AI in some kind of visual comparison.