Samuel Bernal

Rao

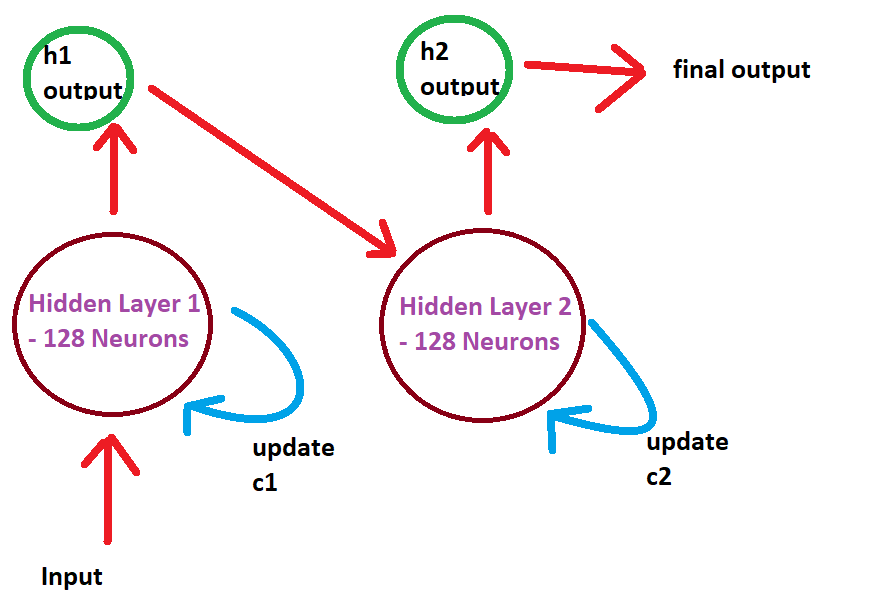
INFO 4150

29 July 2022

MP3 Report & Deliverables

**LSTM Analysis**

For MP3, we used the Long Short Term Memory network to build a predictive model for future Google stock prices. The LTSM network is variant type of RNN which stores memory regarding sequential data. An advantage of this network compared to a traditional RNN is the presence of a of long-term memory gates which can let go of useless information while retaining useful data from earlier in memory.

1. Import
   1. In the first cell, we import all the needed modules to process the data and to build the architecture of our LSTM module.
2. Organizing Data
   1. The Google stock price csv is read into code and converted to a dataframe object. We drop all unneeded information except for the Date and Adj Close value. All the data within “Date” is converted into a datetime object so that the information can be read as date or times instead of being read as strings. The “Date” column is then set as the dataframe index so that adj values can be indexed via date instead of integer location on the dataframe. We then create a “target” column of values which lists the adj close values for the day after the indexed date.
3. Visualize
   1. Within the next cells, the data points are plotted to visualize the trend of adj value. The cells show that the stock price has a steady increase from 2004 to 2020.
4. Create Sequence
   1. What’s going on here?
      1. In this cell, the sequence features and labels are being created. Here, the length of the sequences are of length 1, and the ‘Adj Close’ column is being adjusted as a Series of 1x1 sized arrays. These arrays act as the X data features. There are 3820 sequences each with length 1.
      2. The Y labels are similarly being formed with the ‘target’ column being reformed as a Series with 1x1 sized arrays. Each Y label corresponds with the X feature within the same index.
   2. Normalize data – why?
      1. The X feature data is normalized in order to minimize the scale on which the data is computed. This allows the model to have cleaner and more efficient means of processing the data. Additionally, if there were more features or sequences, there would be a standard scale that the data would be processed on. As a result, training speed would increase and complexity would decrease.
      2. The X features are normalized with Min-Max normalization in order to model the distribution of data from 0 to 1. The minimum value in “adj close” is set to zero while the max is set to 1. All values in between are represented as a proportional decimal point.
   3. Create train and test sets
      1. In this portion of the cell, the features and labels are split into train and tests sets.
      2. The train data contains all the data except for the last 64 sequences.
      3. The test data contains all the data for the last 64 sequences.
5. Convert to torch sensors – why?
   1. Next, we convert the split data into torch sensors. This step is necessary because the tensor object is used as the input for the Neural Network module that we import from PyTorch. By converting data into tensors, we can easily use data to build the model.
6. Build LSTM model class
   1. How many inputs, hidden layers, neurons per layer?
      1. In the LSTM class, there is one input into the model (size one sequence)
      2. There are two hidden cell layers (lstm1 and lstm2)
      3. There are 128 neurons in each hidden layer
   2. Explain architecture
      1. The LTSM architecture works as follows
         1. The input goes into the first layer where the cell outputs both the hidden layer result 1 and cell state 1. Both variables are saved.
         2. The hidden layer result 1 (size 128) is inputted into layer 2 where the layer 2 outputs both the hidden result 2 and cell state 2. Both variables are saved again.
      2. These hidden layers and cell states are saved so that the weights can be updated within the neurons in each hidden layer. This allows the model to build off sequential memory and past data.
   3. Draw Sketch
      1. 
7. Instantiate model, metric and optimizer – what’s going on here?
   1. In this cell, the LTSM model is being initialized as the variable “model”
   2. Additionally, parameters for the minimizing the loss function are being instantiated
      1. The criterion for the magnitude of the loss or cost function will be the mean squared error calculated between the predicted value of the model and the actual value.
      2. Second, the optimizing algorithm (Limited-memory BFGS) for minimizing the loss is being instantiated with a learning parameter/step of 0.25 with a max of 15 steps to minimize the cost. This algorithm works by updating the values or parameters that will approach the lowest possible MSE between the predicted values and actual values
8. Build the training loop
   1. After initializing the optimizer and loss function, we build the training loop function to automate the process of updating model parameters with back propagation. The main structure is as follows:
      1. def closure:
         1. In this function we call the for the loss function to compare the training input to the actual true value. The loss function is returned to the optimizer
      2. The optimizer runs for 15 steps, adjusting the gradient to minimize the loss at each step
   2. The entire function runs for the number of epochs requested. 1 epoch means one run through the entire training dataset.
   3. At the end of the function, the loop will create an updated predicted value from the test inputs and will also print out the RMSE loss from the loss function at each epoch.
9. Training and testing model – what’s going on here?
   1. Here we call for the training loop function which will return our predicted values and the updated RMSE loss. We have selected the function to run for 10 epochs, so the model will train 10 times over the entirety of the training set.
   2. At epoch 9, we have the smallest RMSE loss of 27.8
10. Accuracy – what’s going on here?
    1. To calculate the accuracy of the model, we get the mean of the true test values and the mean squared error of our final epoch. The formula to calculate accuracy is (google\_mean – (rmse)^2) / (google\_mean). The mean squared error shows how far away from the mean the model was. The accuracy of the model is about 98.9% accurate.
11. Plotting actual predicted versus actual values
    1. In the actual plot for the predictions, we can see that the predicted values follow the trend for the test data average. However, the actual stock value is around 100 more than the model prediction.

**Feedback**

What was your biggest takeaway from this course?

* My biggest takeaway from this course is that prediction or classification problems cannot all be solved by the same process, algorithm, or model. Designing a model takes time and consideration as to what data is available and present and what you are trying to predict or classify. I used to think that using Neural Networks or just increasing complexity of models could solve most problems, but given this project and previous class examples, I have realized that models have to be tailored specifically to the type of data you are intending to analyze.

What do you enjoy most about this course?

* I have enjoyed seeing the wide range of applications that predictive models can be applied to. This course was the first time I had experience with Natural Language Processing. I did some research and saw that LSTM can be used to apply to NLP, music, and more sequential data that relies on long term memory or context. By seeing the complexity and structure of NN and LSTM models for the last two projects, I have felt more confident about the NN structure and how to optimize models for specific applications.

What one thing can be done better to improve the delivery of this course?

* I think the delivery of this course is great already. You provide a great amount of information on your lecture slides online, you upload your lecture videos, and you provide some code examples for challenging bits of code. I think the delivery worked well for me.

Do you plan to pursue a career in data science?

* I am currently about to interview for an undergrad research position for cyber-physical systems and security. I am hoping to get some of that experience this fall before I graduate. I am working full time in the power delivery field when I start next spring, but I am curious to see if that field may merge with more cyber-security or IoT technology in the future.