Virginia Tech

Bradley Department of Electrical and Computer Engineering

ECE 5984 Data Engineering Project
Fall 2023

Assignment 3
Machine Learning Pipeline

Please note the following:

- Solutions must be clear and presented in the order assigned. Solutions must show the work needed, as appropriate, to derive your answers.
- Data being processed in both the lab and homework should be in the correct format and location and any other deliverables should also be completed as mentioned in the modules.
- For the Homework, the submission process requires that all code files should be uploaded to Canvas before the deadline.
- Submit your Homework using the respective area of the class website by 11:55 p.m. on the due date.
- When a PDF file must be submitted, include at the top of the first page: your name (as recorded by the university), email address, and the assignment name (e.g., "**ECE 5984, Homework/Project 1**"). Submit a single file unless an additional file is explicitly requested.

Lab 3

3.1 Perform feature extraction on the cleaned data from Lab 2.2

Feature extraction refers to the process of transforming raw data into numerical features that can be processed while preserving the information in the original data set. It yields better results than applying machine learning directly to the raw data. In our example, we will build towards making a prediction model based on the LSTM architecture.

The following steps are then done to perform feature extraction. These steps are executed in code in the featureExtraction.py script. You should read the following steps along with the code inside featureExtraction.py to understand the reasoning behind some of the operations performed and understand the order of operations better. Following is an overview of the code being executed inside featureExtraction.py

1. The cleaned data from lab 2.2 is fetched from the S3 bucket (data warehouse) and loaded as dataframes.
2. The 'adj close' value of each company in question is assigned as a target variable. The target variable is the variable that a user would want to predict using the rest of the dataset in a machine learning context. Next, we pick the features that serve as the independent variables to the target variable (the actual variables you want to train your dataset on). In our case we choose the following features:
   a. 'Close'
   b. 'High'
   c. 'Low'
   d. 'Open'
   e. 'Volume'

3. To decrease the computational cost of the data in the table, we will scale the stock values to values between 0 and 1. As a result, all the data in large numbers is reduced, and therefore memory consumption is decreased. Also, because the data is not spread out in huge values, we can achieve greater precision by scaling down. To perform this, we will be using the MinMaxScaler class of the sci-kit-learn library.
4. Next, the entire dataset is divided into training and test sets before feeding it into any training model. The Machine Learning LSTM model will be trained on the data in the training set and tested for accuracy and backpropagation on the test set. The sci-kit-learn library's TimeSeriesSplit class will be used for this. We set the number of splits to 10, indicating that 10% of the data will be used as the test set and 90% of the data will be used to train the LSTM model.
5. The data from each company is finally divided and stored as variables named X_train, X_test, y_train and y_test. These features are then saved as .pkl files and pushed onto our S3 bucket

Steps to run feature extraction on the pipeline:

1. Open the batch_ingest.py file, the featureExtraction.py file and the transform.py file provided on your local machine
2. Edit and fill in the needed details in the 3 files, key being the S3 bucket locations. Batch_ingest.py should be using an Api to get data and then pushing it to a folder in your S3 bucket. The transform.py file should have location of where to retrieve the .pkl file made at the end of batch_ingest.py and then the new location of where you want your cleaned data to end up at. The featureExtraction.py file should have the S3 bucket location of where you want to retrieve your pickle data file at the end of transform.py file and then the new S3 bucket location of where you want your feature extracted training and testing data to to end up at. These S3 bucket locations can be similar to the ones you have used in the past for other labs as well.
3. Save the featureExtraction.py, batch_ingest.py and transform.py scripts to your local machine
4. Spin up a docker container as shown in Step 2.2: Spin up and exit out of a docker container (lab 0)
5. Install the needed packages for the dataset running the following commands
   a. `pip install pandas-datareader`
   b. `pip install yfinance`
   c. `pip install -U scikit-learn`
6. Navigate to the airflow/dags folder by typing the following command:
   a. `cd airflow/dags`
7. Remove any other .py files already there using the following command (Note: make sure you have a backup of your previous lab code and homework code on your local machine)
   a. `rm file1 file2` (Any number of files can be added with a space in between)
8. Create a new dag.py and copy all the code from your local machine dag.py onto it. To do so enter the command:
   a. `sudo nano dag.py`
9. This will open a command line based text editor. Copy the contents of your local dag.py file to the newly created one inside your container (Tip: use ctrl+shift+V to paste content directly if using the web browser)
10. Save by pressing ctrl+x , then press y to confirm changes and then hit Enter
11. Similarly create the batch_ingest.py , featureExtraction.py and transform.py files and copy code from local machine batch_ingest.py, featureExtraction.py and transform.py onto it using the same steps
12. Access your airflow GUI (Step 2.4: Launching and access the airflow GUI (lab 0))
13. You should see the same **batch_ingest_dag**. Click it and go to graph. On the right side click the play button and press trigger DAG
14. This triggers the dag we just uploaded which intern runs the ingest_data function from batch_ingest.py, the transform_data() function from transform.py and the feature_extract() function from the featureExtraction.py file in sequence.
15. After the DAG finishes running you should be able to navigate to your S3 bucket directory and check to see the data now there in a pickle file format. You should have a data.pkl and then a clean_aapl.pkl file along with X_train_aapl.pkl, X_test_aapl.pkl, y_train_aapl.pkl, y_test_aapl.pkl for apple and then similar .pkl files for google and amazon.
16. After confirming your data arrived at the correct location in your S3 bucket, close airflow by pressing ctrl-C on the command line where airflow is running
17. Exit out of the container you have been working from using the command `exit` and double check if the container actually stopped by using the command `docker ps`. To manually stop the container if it had not done so automatically run the command `docker stop <pid>.`

3.2 Train a machine learning model on the stock market data set being used throughout the previous labs

In this lab we will be building and training an LSTM prediction model that can predict the ‘adj close’ price of the stocks data we feed it. The point of the lab is not to provide a state-of-the-art method to build such a model but rather to give a template on how to implement a pipeline that can train almost any kind of machine learning model.

After performing feature selection and extraction in lab 3.1, the extracted data is then reshaped and fed into the LSTM model according to the input layers of the model. Once the training and test sets are retrieved from our data warehouse, we will input the data into the LSTM model. Before we can do that, we must transform the training and test set data into a format that the LSTM model can interpret. The LSTM needs the data to be in a 3D format, therefore, we first transform the training and test data to NumPy arrays and then restructure them to match the format.

Next, we construct the LSTM Model. In this step, we’ll build a Sequential Keras model with one LSTM layer. The LSTM layer has 32 units and is followed by one Dense Layer of one neuron. We compile the model using Adam Optimizer and the Mean Squared Error as the loss function. For an LSTM model, this is the most preferred combination.

The last step is to use the fit function to train the LSTM model created on the training data for each company for 25 epochs with a batch size of 8. The trained model is then saved and pushed to our s3 bucket in a .h5 file format.

In order to perform the lab, follow these steps:

1. Open the build_train_model.py file on your local machine.
2. Insert the correct S3 bucket locations where needed. They should match the S3 bucket locations used in featureExtraction.py and save the file
3. Open dag.py on your local machine and uncomment all the lines of code that are a comment and then save the file. This means to remove only the ‘#’ symbol at the start of any line of code. Be careful to only remove the ‘#’ symbol and not any other part of the code.
4. Spin up a docker container as shown in Step 2.2: Spin up and exit out of a docker container (lab 0)
5. Install the needed packages for the dataset running the following commands
   a. `pip install pandas-datareader`
   b. `pip install yfinance`
   c. `pip install -U scikit-learn`
6. Navigate to the airflow/dags folder by typing the following command:
   a. `cd airflow/dags`
7. Remove any other .py files already there using the following command (Note: make sure you have a backup of your previous lab code and homework code on your local machine)
   a. `rm file1 file2` (Any number of files can be added with a space in between)
8. Create a new dag.py and copy all the code from your local machine dag.py onto it. To do so enter the command:
   a. `sudo nano dag.py`
9. This will open a command line based text editor. Copy the contents of your local dag.py file to the newly created one inside your container (Tip: use ctrl+shift+V to paste content directly if using the web browser)
10. Save by pressing ctrl+x , then press y to confirm changes and then hit Enter

11. Similarly create the batch_ingest.py , featureExtraction.py, build_train_model.py and transform.py files and copy code from local machine batch_ingest.py, featureExtraction.py and transform.py onto it using the same steps. The batch_ingest.py, transform.py and featureExtraction.py files should be the exact same ones used in lab 3.1
12. Access your airflow GUI (Step 2.4: Launching and access the airflow GUI (lab 0))
13. You should see the same **batch_ingest_dag**. Click it and go to graph. On the right side click the play button and press trigger DAG
14. This triggers the dag we just uploaded which intern runs the ingest_data function from batch_ingest.py, the transform_data() function from transform.py, the feature_extract() function from the featureExtraction.py file and the build_train() function from the build_train_model.py file in sequence. Since we are training 3 models, the time to complete this DAG run should be longer than usual.
15. After the DAG finishes running you should be able to navigate to your S3 bucket directory and check to see the data now there in a pickle file format. You should have a data.pkl and then a clean_aapl.pkl file along with X_train_aapl.pkl, X_test_aapl.pkl, y_train_aapl.pkl, y_test_aapl.pkl for apple and then similar .pkl files for google and amazon similar to lab 3.1. Additionally you should see lstm_aapl.h5, lstm_amzn.h5 and lstm_googl.h5 which are the saved models trained the apple, amazon and google stock data respectively.
16. After confirming your data arrived at the correct location in your S3 bucket and that your model saved, close airflow by pressing ctrl-C on the command line where airflow is running
17. Exit out of the container you have been working from using the command `exit` and double check if the container actually stopped by using the command `docker ps`. To manually stop the container if it had not done so automatically run the command `docker stop <pid>.`

Deliverables (for both 3.1 and 3.2)

1. The dag.py file, batch_ingest.py, featureExtraction.py, build_train_model.py and the transform.py file for only lab 3.2 should be uploaded to Canvas.
2. Data from the lab should be in the appropriate S3 buckets (datalake and data warehouse). Namely the following files:
    a. data.pkl
    b. clean_aapl.pkl, clean_amzn.pkl, clean_googl.pkl
    c. X_train_aapl.pkl, X_test_aapl.pkl, y_train_aapl.pkl, y_test_aapl.pkl
    d. X_train_amzn.pkl, X_test_amzn.pkl, y_train_amzn.pkl, y_test_amzn.pkl
    e. X_train_googl.pkl, X_test_ googl.pkl, y_train_ googl.pkl, y_test_ googl.pkl
3. Saved model should be in .h5 format in the appropriate location within the S3 bucket. Namely:
    a. lstm_aapl.h5, lstm_amzn.h5, lstm_googl.h5