Deep Learning for Time Series

Abstract

With ease of financial data availability, technological, and computational advancements, Artificial intelligence techniques like Machine Learning (ML) and Deep Learning (DL) are extensively used in Finance. The demand for predicting stock price direction and trends is increasing at a very rapid pace. This report and the accompanied solution are the comprehensive work done in order to predict the movement of direction of stock using various financial ratios, technical indicators, and volatility estimators. The work includes fetching raw data, applying feature engineering, building and tuning various models, validating them, and evaluating key metrics to compare different DL model effectiveness. The system achieves overall high level of accuracy for stock market direction prediction. The project intends to find a correct balance of financial domain and technical implementation.

Objective

The objective is to predict up or down movement of a security or an Index using Deep Learning algorithms. The intent is to predict the movement or the direction of the financial instrument instead of predicting a price hence the problem relates to a binary classification. The report corresponds to the analysis of **FTSE Index** using Feature Engineering and Deep Learning based Model construction along with different hyperparameter tuning algorithms. All the models are evaluated on generated key metrics. The report contains visualization for feature selection algorithms and generated metrics. More than 20 years of raw data and Fama French factors are fetched for the analysis and then training & testing Deep Learning models.

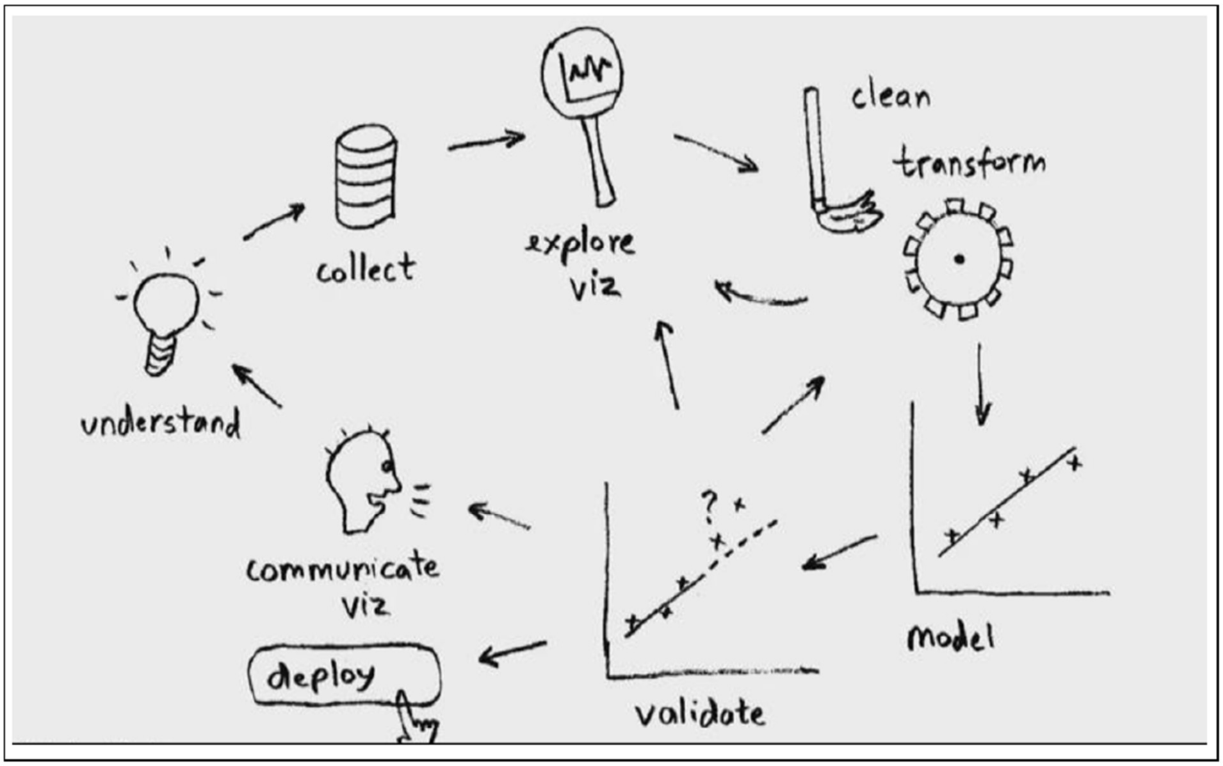
Introduction

The stock market is well known for its dynamic, volatile, and unstable nature. A particular security may be shining on one day and struggling on another. Intelligent traders and big financial institutions make big money by taking corresponding long and short positions based on capturing meaningful insights from data available. Investing money in stock comes with its own risks and rewards. The risk associated with the stock investment can be systematic and idiosyncratic which when attributed and managed properly can result in significant alpha generation for a trader or a portfolio manager. The below work is an effort to capture raw data of a financial instrument (security, commodity, or Index), to perform feature scaling to derive financial, technical, and volatility indicators which are then passed to model construction logic with an intent to predict the movement of that instrument. The financial time series data is then passed to multiple Deep Learning models like Multi-Layer Perceptron, Single Layer LSTM, Dual Layer LSTM and Stacked LSTM to get useful price direction signals. The objective of the report is to build a model which focuses on short term price trend prediction using historical financial time series.

Introduction to Financial Time Series

A Financial Time series is a series of data points ordered by time for a particular stock or Index or any Financial instrument in general. This time series gives an information to the analyst as to how different indicators governing asset prices, risk, and returns evolved over the time-period.

Overview of steps to build a Model



Any ML or DL model building exercise is roughly divided into 7 steps which are:

1. **Understand and articulate the problem.** The Data Science practitioner first analyze the problem, defines an objective, and proposes different models to come up with an efficient solution.

2. **Data Collection.** The raw data is collected from variety of sources. Feature extraction is performed where additional meaningful features are derived from raw data.

3. **Perform Exploratory Data Analysis.** Data is analyzed to check for inconsistencies, missing data, outlier detection etc.

4. **Data cleaning and transformation.** This is an important step after EDA where all the identified inconsistencies are handled. Missing features data is imputed ie. either backfilled or dropped

completely, features are scaled and transformed appropriately. Using feature selection and/or dimensionality reduction techniques only meaningful features are picked up for model construction.

5. **Model hyperparameter tuning and construction.** Once the data is ready, it is passed to a set of pre-identified models where model hyperparameter tuning is performed. Hyperparameter tuning helps in identifying optimal values of model hyperparameters where it would perform best.

6. **Model validation.** In this step all candidate models are evaluated for their effectiveness. All relevant metrics like accuracy, precision, recall, loss, error etc. are analyzed and the best performing model is selected.

7. **Model deployment.** The selected model is then prepared to be deployed in production and is tested with real life data. Model`s performance with the real data is evaluated.

The above steps are iterative ie. once model is deployed to production, it is evaluated for effectiveness and any mismatch in performance expectation is addressed on continuous basis. At some predetermined frequency (monthly/quarterly/annually) models can be retrained with the new incoming data.

Using the above guidance on model building the project work and the report is divided into four broad categories:

1. Fetch the raw data from different public data sources.
2. Perform feature engineering ie. feature extraction, feature scaling & transformation, and feature selection.
3. Model hyperparameter tuning and construction using variety of DL algorithms.
4. Evaluate model effectiveness by looking at key metrics.

The Deep Learning models (MLP, Single Layer LSTM, Dual Layer LSTMs, and Multi-Layer LSTMs) are tested for Index, security, and commodity. The analysis in this report corresponds to an Index **(FTSE)** but the work is also done for security **(Goldman Sachs: GS)** and commodity **(Silver: SI=F)**. All 3 corresponding notebooks are attached containing details of Feature Engineering, Model Construction, Model Validation and Evaluation.

Almost 21 years of data is fetched from Jan 1, 2001 to Dec 21, 2021 with around 5000 data points with daily frequency.

Features Overview – Raw Features and Feature Extraction

Raw prices (Open, High, Low, Close, Volume, Adjusted Close) and Fama French Factors (Mkt-RF, SMB, HML, RMW, CMA, RF) are fetched from yahoo finance and through pandas data reader library. The Adjusted raw price is used for Feature extraction to derive various technical indicators with different lookback periods. The below table contains set of 110 identified features.

Feature	Description
Open | High | Low | Close | Volume	Open, High, Low, Close Price and Volume
Adj Close	Adjusted closing price
RET-5D | RET-10D | RET-21D | RET-50D | RET-200D	5,10,21,50,200 Day Percentage returns
SMB | HML | RMW | CMA | RF | Mkt-RF	Small minus Big, High minus Low, Robust Minus Weak, Conservative Minus Aggressive, Risk Free Return, excess return on Market over Risk Free
ADX_5 | ADX_10 | ADX_21 | ADX_50	5,10,21,50 Day Average Directional Movement Index
MACD_12_26_9	Moving Average Convergence Divergence
RSI_5 | RSI_10 | RSI_21 | RSI_50 | RSI_200	5,10,21,50,200 Day Relative Strength Index
TSI_13_25_13	True strength index
RVGI_5_4 | RVGI_10_4 | RVGI_21_4 | RVGI_50_4	5,10,21,50 Day Relative Vigor Index
APO_12_26	Absolute Price Oscillator
ROC_5 | ROC_10 | ROC_21 | ROC_50	5,10,21,50 Day Rate of Change
CCI_5_0.015 | CCI_10_0.015 | CCI_21_0.015 | CCI_50_0.015 | CCI_200_0.015	5,10,21,50,200 Day Commodity Channel Index
BIAS_SMA_26	Bias
MOM_5 | MOM_10 | MOM_21 | MOM_50	5,10,21,50 Day Momentum
ATRr_5 | ATRr_10 | ATRr_21 | ATRr_50 | ATRr_200	5,10,21,50,200 Day Average True Range
BBP_5_2.0 | BBP_10_2.0 | BBP_21_2.0 | BBP_50_2.0 | BBP_200_2.0	5,10,21,50,200 Day Bollinger Bands Indicator
DCL_20_20 | DCM_20_20 | DCU_20_20	Donchian Channels - Lower, Middle, Upper
KCLe_20_2 | KCBe_20_2 | KCUe_20_2	Keltner Channel - Lower, Basis, Upper
PDIST	Price Distance
RVI_14	Relative Volatility Index
EMA_5 | EMA_10 | EMA_21 | EMA_50 | EMA_200	5,10,21,50,200 Day Exponential Moving Average
SMA_5 | SMA_10 | SMA_21 | SMA_50 | SMA_200	5,10,21,50,200 Day Simple Moving Average
VWAP_D	Volume Weighted Average Price
VWMA_10	Volume Weighted Moving Average
MAD_5 | MAD_10 | MAD_21 | MAD_50	5,10,21,50 Day Mean Absolute Deviation
ZS_5 | ZS_10 | ZS_21 | ZS_50	5,10,21,50 Day Z Score
AD	Accumulation/Distribution Index
ADOSC_3_10	Accumulation/Distribution Oscillator
MFI_5 | MFI_10 | MFI_21 | MFI_50	5,10,21,50 Day Money Flow Index
NVI_5 | NVI_10 | NVI_21 | NVI_50	5,10,21,50 Day Negative Volume Index
PVI_5 | PVI_10 | PVI_21 | PVI_50	5,10,21,50 Day Positive Volume Index
PVT	Price Volume Trend
OBV	On-Balance Volume
LOG-RET-5D | LOG-RET-10D | LOG-RET-21D | LOG-RET-50D | LOG-RET-200D	5,10,21,50,200 Day Log Returns
yz_vol_5D | yz_vol_10D | yz_vol_21D | yz_vol_50D	5,10,21,50 Day Yang & Zhang Drift Independent Volatility

Feature Engineering

Exploratory Data Analysis (EDA) is performed on fetched raw features and Fama French factors.

The extracted features are combined with raw prices and Fama French factors to create a full-fledged initial features list as shown above. Exploratory Data Analysis is done on complete feature list and none of the data was found missing except when features are extracted with different lookback periods ranging from 5 days to 200 days. Moreover, by generating different statistics on data like mean, min, max, standard deviation, and range of different percentiles it was found that data was clean and does not require any imputation.

The data is then transformed using **StandardScalar** in order to organize it using same scale. This is done to Standardize features by removing the mean and scaling to unit variance.

The score of sample 'x' is calculated as:

z = (x -u) / s
where, z is the Standardized score
u is sample`s mean
s is sample`s standard deviation

Feature Selection and Dimensionality Reduction Techniques

Before feeding the data to Deep Learning models, an array of feature selection and dimensionality reduction techniques are applied to original feature set containing 110 features.

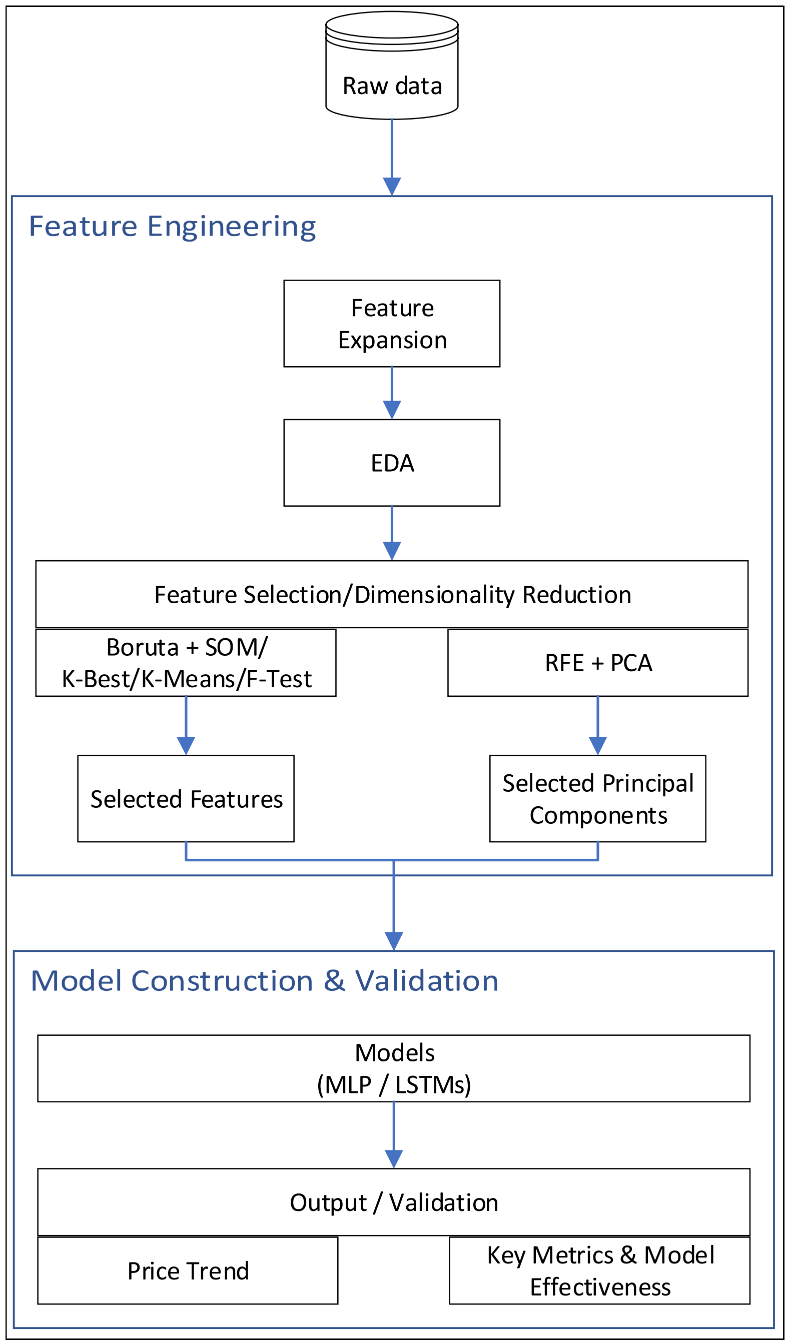
In order to assess the impact this is divided into 2 approaches:

1. Variety of different feature selection algorithms are applied namely:
    a. **Decision Tree Regressor** (DTR)
    b. **K-Means clustering**
    c. A combination of **Boruta** and **Self Organizing Maps** (SOM)
    d. **F-Test** using **SelectKBest**

    Features intersection is done for all identified features using selection algorithms above. The most relevant features are selected and fed to tuned models to evaluate key performance metrics mentioned later.

2. A combination of **Recursive Feature Elimination** (RFE) and then **Principal Component Analysis** (PCA) is applied to get 5, 10, 15 and 20 Principal Components. These Principal Components are then fed into tuned models separately.

High Level Architecture Diagram



Architecture Summary

Once the raw prices and Fama French factors are fetched, they are pass through **Feature Engineering** layer which performs **Feature expansion** to identify key technical indicators and volatility estimators, **Feature Standardization** through scaling and transformation, and finally **Feature Selection** using variety of different algorithms. These are primarily classified into two groups - features selected using different algorithms and using RFE & PCA into key Principal Components. These two groups are passed to different models such as **Multi-layer Perceptron (MLP)** and different layers of **LSTM** models (Single/Dual/Multiple LSTMs). All the key metrics are generated to validate and evaluate model effectiveness. The framework is predicting price trend, but it is equally capable of predicting momentum and volatility movements by changing target accordingly.

Features Selection Algorithms

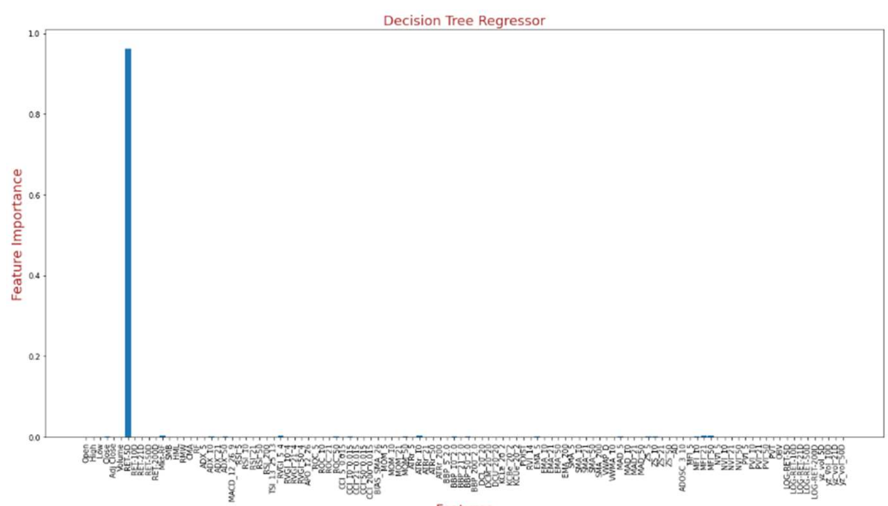
Decision Tree Regressor (DTR)

Decision trees regression normally use mean squared error (MSE) to decide to split a node in two or more sub-nodes. Using a binary tree, the algorithm first will pick a value and split the data into two subsets. For each subset, it will calculate the MSE separately. The tree chooses the value which results in smallest MSE value.

Top 20 features selected by DTR ordered by importance are:

DTR Selected features				
RET-5D	MFI_21	RVGI_5_4	Mkt-RF	MFI_50
ATRr_10	BBP_10_2.0	ADX_10	ROC_5	ADX_50
ZS_10	CCI_10_0.015	BBP_50_2.0	MOM_50	MAD_5
EMA_5	MFI_10	ZS_5	ROC_5	PVI_5

** Features arranged by importance left to right and top to bottom



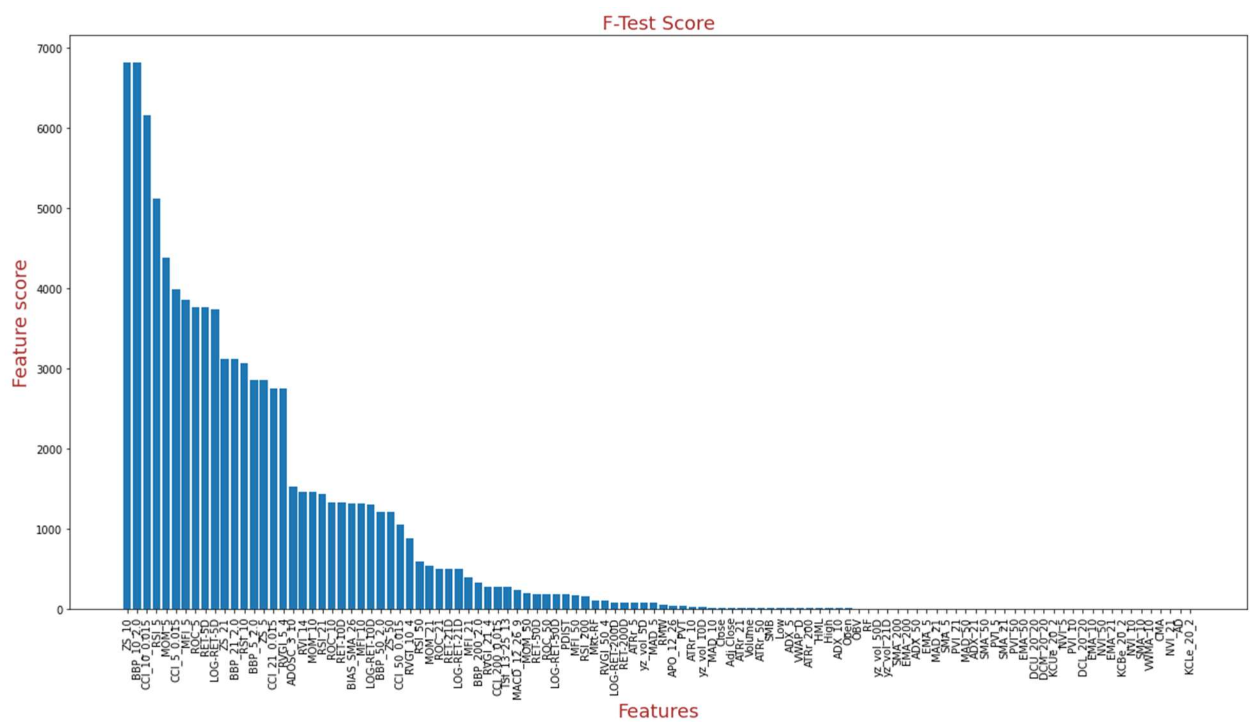
F-Test

The scikit-learn machine library provides an implementation of the correlation statistic in the F-Test (f_regression function). This function can be used in a feature selection strategy, such as selecting the top **k** most relevant features (largest values) using the SelectKBest class.

Top 20 features selected by F-Test using SelectKBest ordered by importance are:

F-Test Selected Features				
ZS_10	BBP_10_2.0	CCI_10_0.015	RSI_5	MOM_5
CCI_5_0.015	MFI_5	ROC_5	RET-5D	LOG-RET-5D
ZS_21	BBP_21_2.0	RSI_10	BBP_5_2.0	ZS_5
CCI_21_0.015	RVGI_5_4	ADOSC_3_10	RVI_14	MOM_10

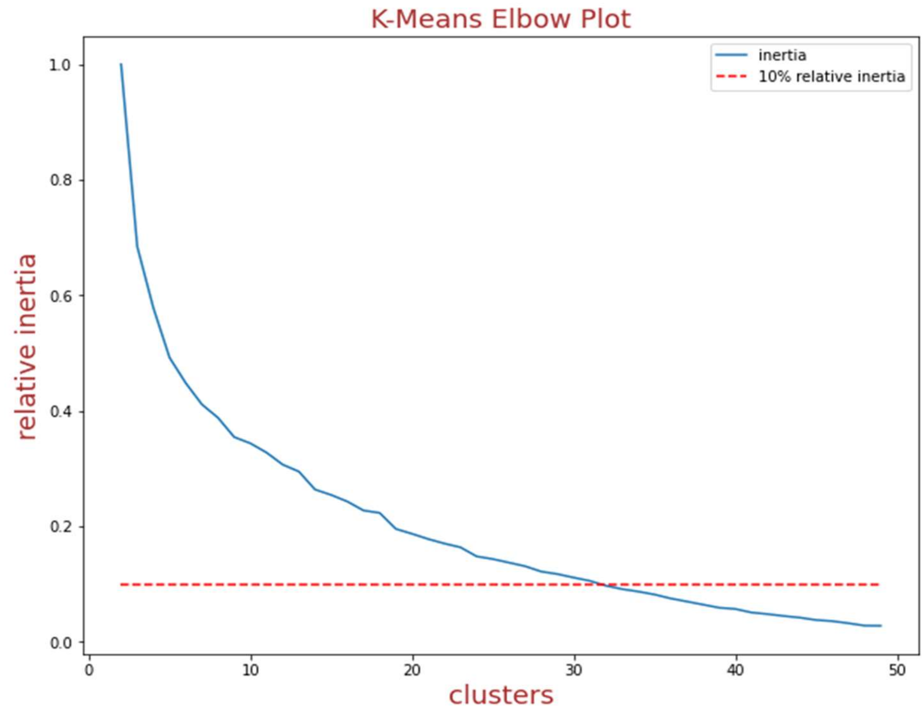
** Features arranged by F-score left to right and top to bottom



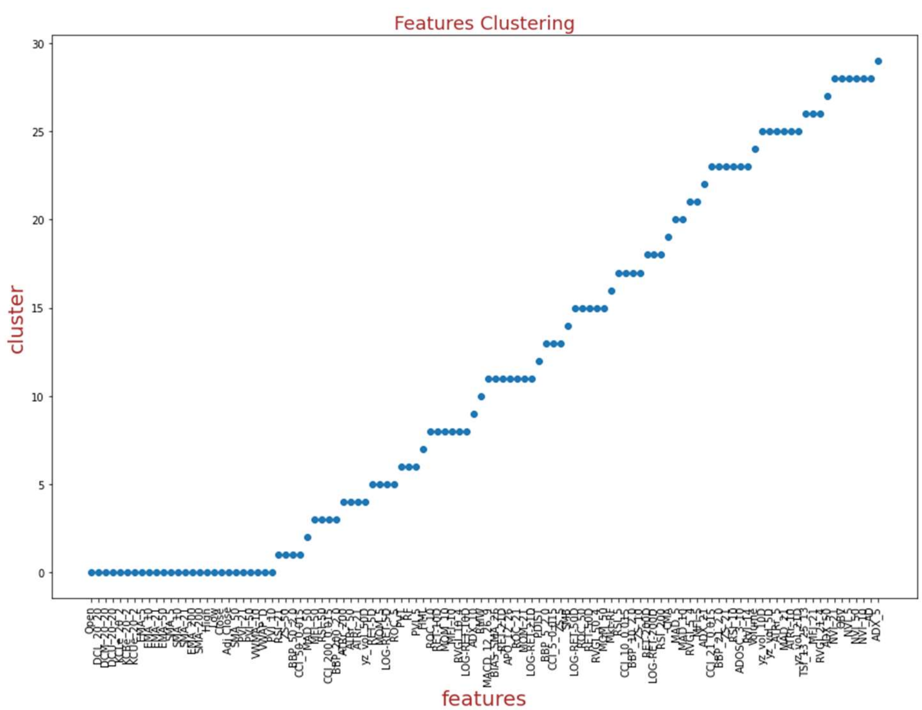
K-Means Clustering

K-means clustering is a method of vector quantization, that aims to partition **n** observations into **k** clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster.

Elbow plot is used to identify the optimum cluster size of 30.



Features are organized within those 30 clusters as shown below.



Below 30 features are collected from every cluster:

K-Means Selected Features				
EMA_5	BBP_50_2.0	MAD_50	MFI_50	yz_vol_50D
RET-5D	PVI_5	HML	MFI_10	ADX_10
RMW	ROC_21	PDIST	ZS_5	SMB
MOM_50	Mkt-RF	ZS_10	RSI_200	CMA
MAD_5	RVGI_5_4	ADX_21	RVI_14	Volume
ATRr_10	MFI_21	ADX_50	AD	ADX_5

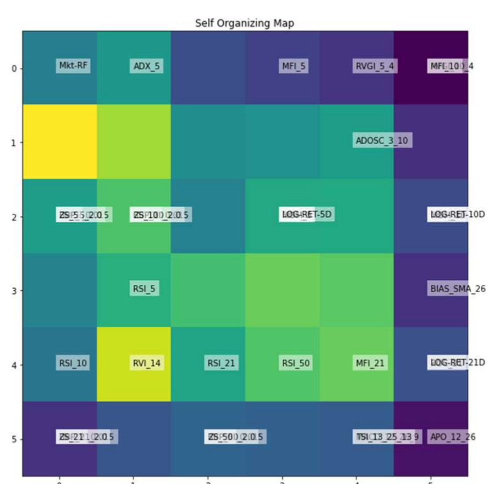
Boruta + Self Organizing Maps (SOM)

A combination of Boruta and SOM is used to identify set of features. Boruta algorithm is designed to take 'all-relevant' approach to feature selection. It a wrapper algorithm built around Random Forest Classifier which runs and selects features without tuning of parameters. The 'all-relevant' approach means that the algorithm would select all the features which are meaningful for target irrespective whether those features are highly correlated hence Boruta is combined with SOM, an algorithm which can identify non-linearity in data which is generally associated with financial time series data.

BorutaPy and MiniSom implementations are used as a combination to achieve this task.
Features selected using Boruta algorithm are:

Boruta Selected Features				
RET-5D	RET-10D	RET-21D	Mkt-RF	ADX_5
MACD_12_26_9	RSI_5	RSI_10	RSI_21	RSI_50
TSI_13_25_13	RVGI_5_4	RVGI_10_4	APO_12_26	ROC_5
ROC_10	ROC_21	CCI_5_0.015	CCI_10_0.015	CCI_21_0.015
CCI_50_0.015	BIAS_SMA_26	MOM_5	MOM_10	BBP_5_2.0
BBP_10_2.0	BBP_21_2.0	BBP_50_2.0	RVI_14	ZS_5
ZS_10	ZS_21	ZS_50	ADOSC_3_10	MFI_5
MFI_10	MFI_21	LOG-RET-5D	LOG-RET-10D	LOG-RET-21D

Boruta selected features are then passed to MinSom which assigns the features to the Best Matching Unit (BMU).



Top 20 features selected from SOM BMU

Boruta + SOM Selected Features				
Mkt-RF	ADX_5	MFI_5	RVGI_5_4	MFI_10
ADOSC_3_10	CCI_5_0.015	ZS_10	ROC_5	RSI_5
BIAS_SMA_26	RSI_10	RVI_14	RSI_21	RSI_50
MFI_21	ROC_21	BBP_21_2.0	BBP_50_2.0	APO_12_26

Features selected from all above algorithms **(DTR, F-Test, K-Means, and Boruta + SOM)** were analyzed and top 15 relevant features are selected to feed into Model construction step. Below is the list of relevant selected features.

Final Selected Features				
RET-5D	MFI_21	RVGI_5_4	Mkt-RF	ADX_5
ROC_5	MFI_5	ADX_10	ZS_10	CCI_10_0.015
BBP_50_2.0	MAD_5	EMA_5	MFI_10	ZS_5

As shown below final set of selected features have representation from all four feature selection algorithms.

DTR Selected features				
RET-5D	MFI_21	RVGI_5_4	Mkt-RF	MFI_50
ATRr_10	BBP_10_2.0	ADX_10	ROC_5	ADX_50
ZS_10	CCI_10_0.015	BBP_50_2.0	MOM_50	MAD_5
EMA_5	MFI_10	ZS_5	ROC_50	PVI_5

F-Test Selected Features				
ZS_10	BBP_10_2.0	CCI_10_0.015	RSI_5	MOM_5
CCI_5_0.015	MFI_5	ROC_5	RET-5D	LOG-RET-5D
ZS_21	BBP_21_2.0	RSI_10	BBP_5_2.0	ZS_5
CCI_21_0.015	RVGI_5_4	ADOSC_3_10	RVI_14	MOM_10

K-Means Selected Features				
EMA_5	BBP_50_2.0	MAD_50	MFI_50	yz_vol_50D
RET-5D	PVI_5	HML	MFI_10	ADX_10
RMW	ROC_21	PDIST	ZS_5	SMB
MOM_50	Mkt-RF	ZS_10	RSI_200	CMA
MAD_5	RVGI_5_4	ADX_21	RVI_14	Volume
ATRr_10	MFI_21	ADX_50	AD	ADX_5

Boruta + SOM Selected Features				
Mkt-RF	ADX_5	MFI_5	RVGI_5_4	MFI_10
ADOSC_3_10	CCI_5_0.015	ZS_10	ROC_5	RSI_5
BIAS_SMA_26	RSI_10	RVI_14	RSI_21	RSI_50
MFI_21	ROC_21	BBP_21_2.0	BBP_50_2.0	APO_12_26

Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA)

A combination of RFE and PCA is adopted as an alternative approach to select features. The selected features from both approaches are fed into identified Deep Learning algorithms to compare key metrics and performance.

First top 30 features are selected by RFE and then PCA algorithm is applied to get 5,10,15,20 Principal Components and their corresponding Explained Variance Ratios.

RFE Selected Features				
RET-5D	RET-50D	Mkt-RF	CMA	ADX_10
RSI_5	RSI_10	RVGI_5_4	RVGI_10_4	ROC_5
ROC_10	CCI_5_0.015	CCI_10_0.015	CCI_200_0.015	MOM_5
BBP_5_2.0	BBP_10_2.0	BBP_21_2.0	BBP_200_2.0	SMA_200
MAD_50	ZS_5	ZS_10	ZS_21	MFI_5
MFI_21	MFI_50	NVI_21	PVI_5	LOG-RET-5D

Principal Components	PC Explained Variance Ratio
5	77%
10	92%
15	98%
20	99.7%

All these 4 PC sets along with feature set selected using DTR, F-Test, K-Means, Boruta & SOM are fed into Deep Learning algorithms. The results of all these 5 feature sets with the combination of all Deep Learning algorithms are analyzed and presented for feature selection and model effectiveness.

5 feature sets and 4 models are considered for analysis

Feature Sets	Description
FS	Combination of Feature Selection techniques: DTR/F-Test/K-Means/Boruta & SOM
PC5	5 Principal Components
PC10	10 Principal Components
PC15	15 Principal Components
PC20	20 Principal Components

Model
Multi-Layer Perceptron (MLP)
Single Layer LSTM
Dual Layer LSTM
Multi-Layer LSTM

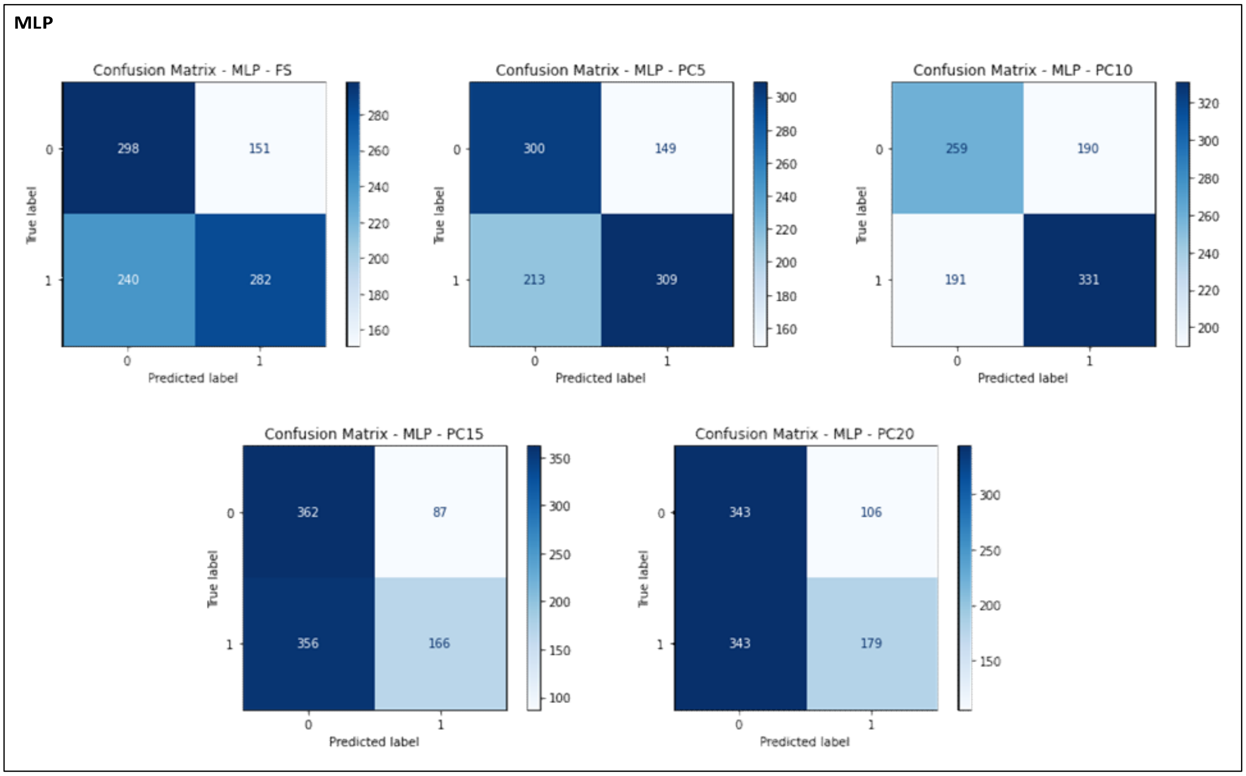
Deep Learning Models

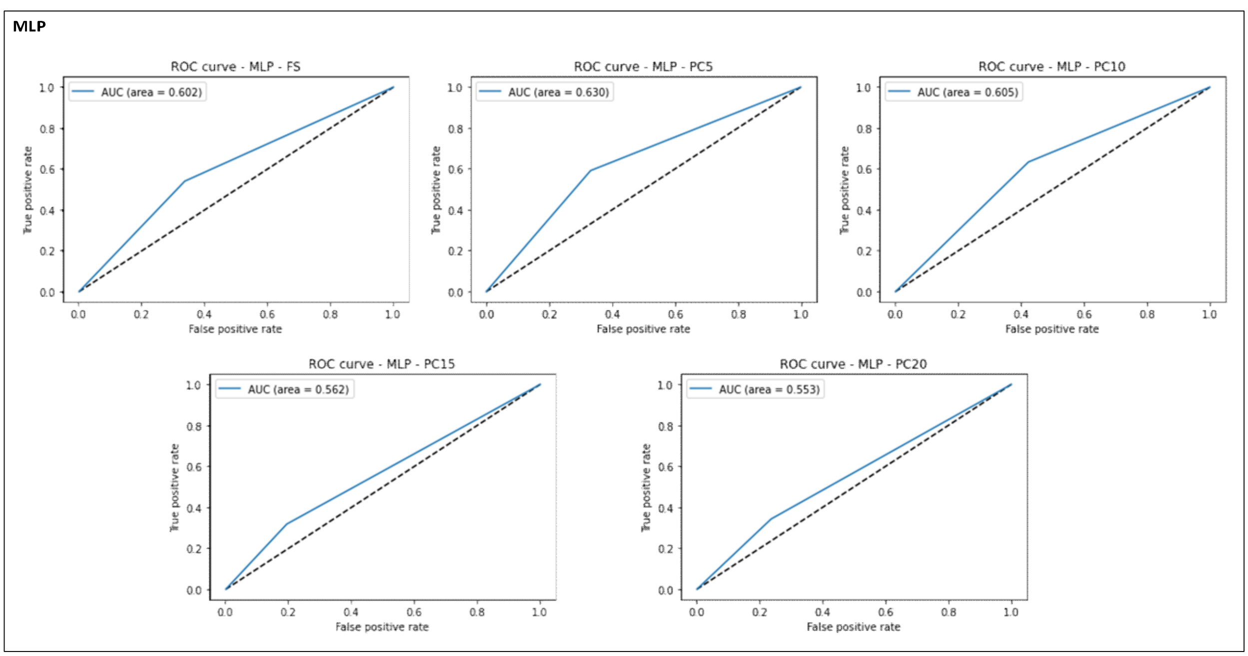
Deep Learning algorithms are efficient enough to pick up only relevant features and discard irrelevant ones but still Feature Selection and Dimensionality reduction techniques are adopted so that only relevant data is passed to models which would improve the runtime performance and reduce noise during model training.

In the project work 4 Deep Learning models (**MLP**, **Single Layer LSTM**, **Dual Layer LSTM**, and **Multi-Layer LSTM**) are tuned and implemented. For each of the model, hyper-parameter tuning is done using FTSE separately. The model specific hyperparameter tuning notebook uses 3 major Keras tuners – **Random Search, Hyperband, and Bayesian Optimization**. The best performant hyperparameters are used in notebook. All the 5 feature sets from previous step are passed to these 4 tuned models thereby having 20 combinations of model runs. As can be observed in notebooks, majority of the **LSTM** layers are using **ReLU** (Rectified Linear Activation Function) which allows a neural network to learn non-linear dependencies. ReLU will return the input directly if the value passed is greater than 0, if the input is less than 0 then 0.0 is returned.

The subsequent pages contain the key metrics generated as part of model execution and their corresponding visualization in form of confusion matrix and ROC Curves. A tabular representation of every metric generated is also provided to compare model effectiveness.

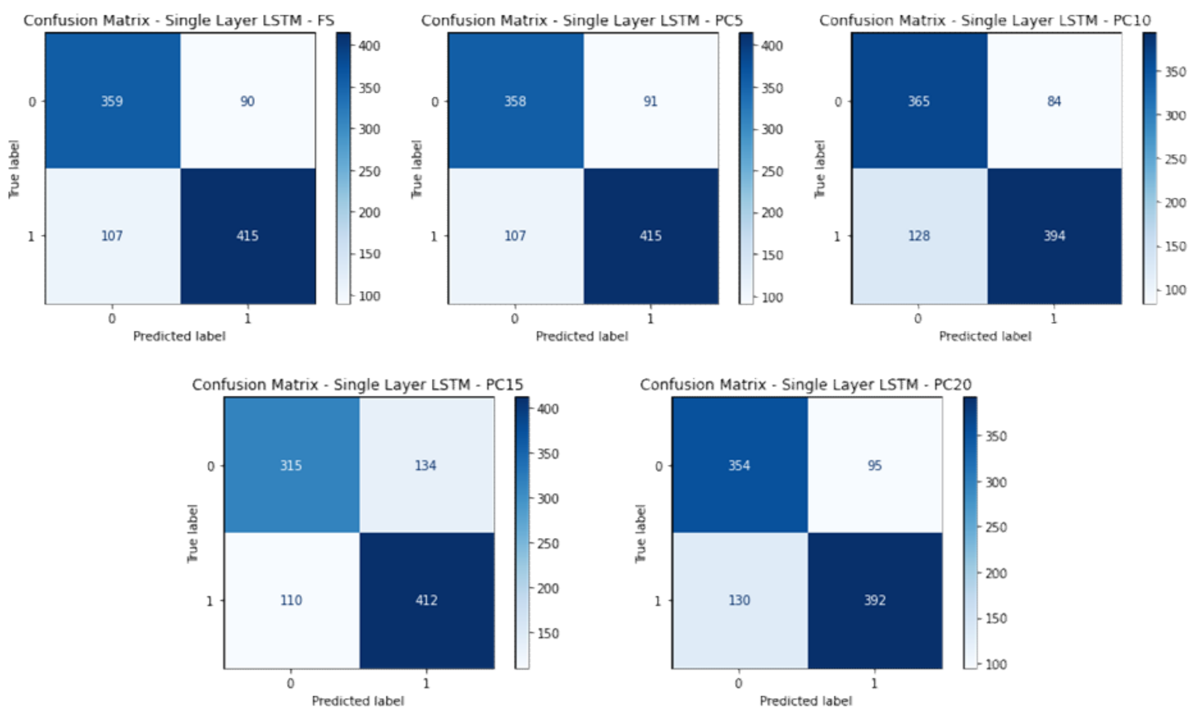
Multi-Layer Perceptron (MLP)



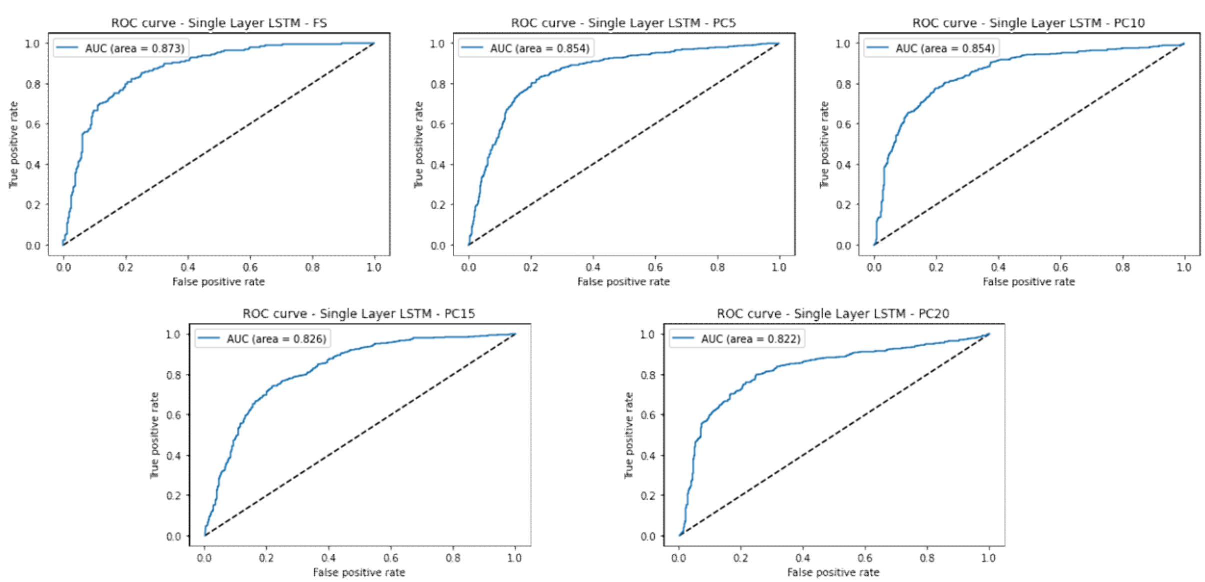


Single Layer LSTM

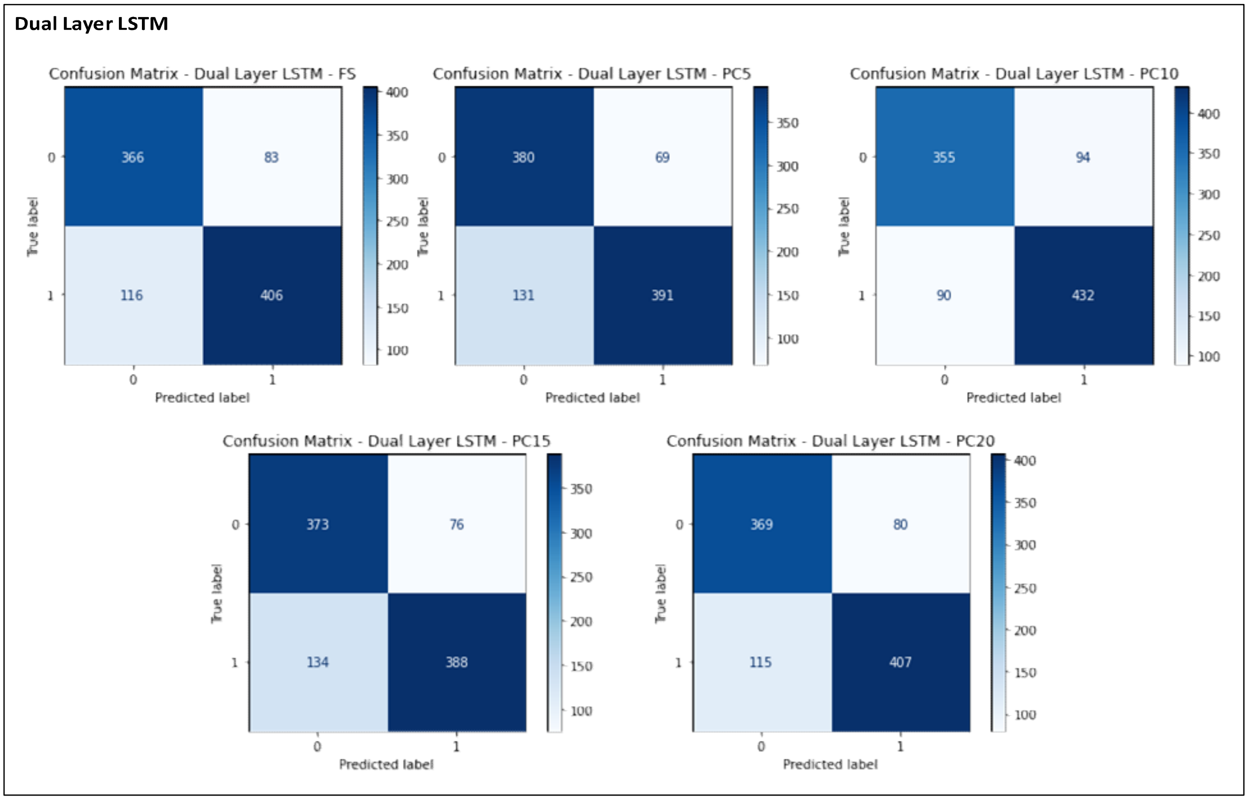
Single Layer LSTM

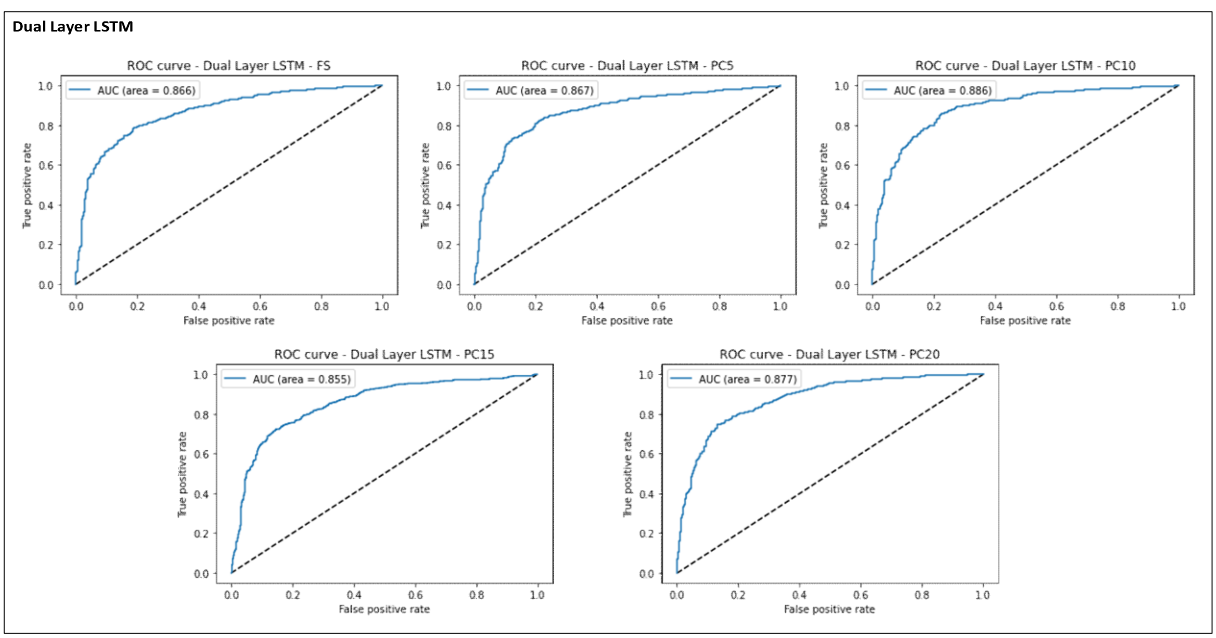


Single Layer LSTM



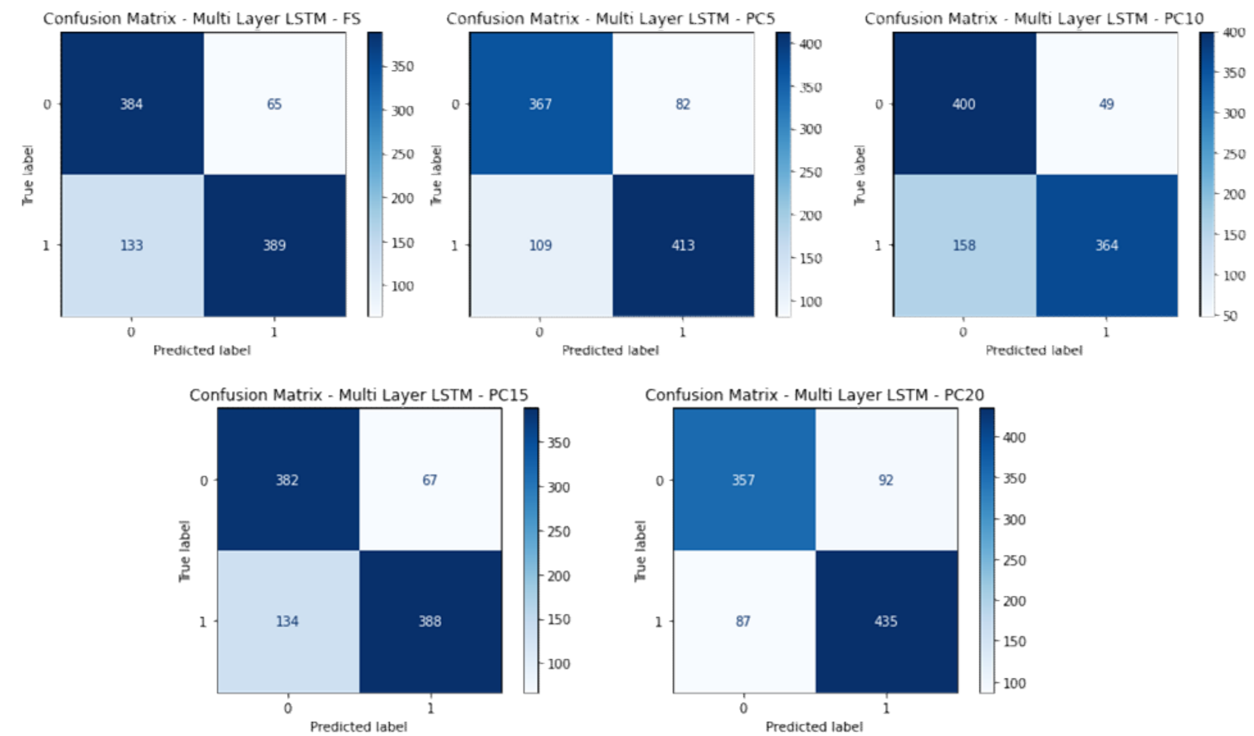
Dual Layer LSTM



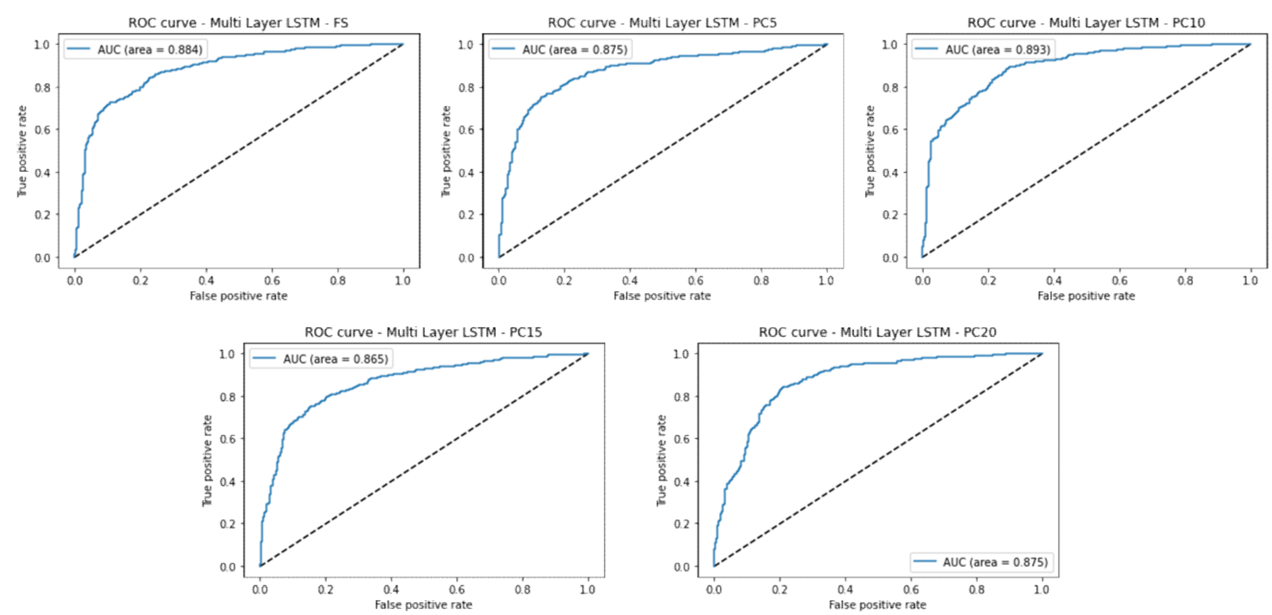


Multi-Layer LSTM

Multi-Layer LSTM



Multi-Layer LSTM



Key metrics

Accuracy					
FTSE	Features Selected	PC5	PC10	PC15	PC20
MLP	59.73%	62.72%	60.76%	54.38%	53.76%
Single Layer LSTM	79.71%	79.61%	78.17%	74.87%	76.83%
Dual Layer LSTM	79.51%	79.40%	81.05%	78.37%	79.92%
Multi-Layer LSTM	79.61%	80.33%	78.68%	79.30%	81.57%

Precision					
FTSE	Features Selected	PC5	PC10	PC15	PC20
MLP	60.62%	63.31%	60.77%	58.59%	56.88%
Single Layer LSTM	79.80%	79.69%	78.55%	74.84%	77.09%
Dual Layer LSTM	79.75%	80.08%	81.04%	78.97%	80.18%
Multi-Layer LSTM	80.41%	80.51%	80.53%	80.08%	81.55%

Recall					
FTSE	Features Selected	PC5	PC10	PC15	PC20
MLP	59.73%	62.72%	60.76%	54.38%	53.76%
Single Layer LSTM	79.71%	79.61%	78.17%	74.87%	76.83%
Dual Layer LSTM	79.51%	79.40%	81.05%	78.37%	79.92%
Multi-Layer LSTM	79.61%	80.33%	78.68%	79.30%	81.57%

F1 Score					
FTSE	Features Selected	PC5	PC10	PC15	PC20
MLP	59.67%	62.74%	60.77%	51.72%	51.80%
Single Layer LSTM	79.73%	79.63%	78.20%	74.81%	76.86%
Dual Layer LSTM	79.53%	79.42%	81.04%	78.39%	79.95%
Multi-Layer LSTM	79.62%	80.36%	78.59%	79.31%	81.56%

Mean Absolute Error					
FTSE	Features Selected	PC5	PC10	PC15	PC20
MLP	49.66%	49.40%	49.49%	49.39%	49.59%
Single Layer LSTM	22.31%	23.68%	23.93%	25.75%	24.60%
Dual Layer LSTM	22.97%	23.00%	21.21%	23.86%	21.83%
Multi-Layer LSTM	22.44%	21.90%	23.05%	23.58%	25.14%

Mean Squared Error					
FTSE	Features Selected	PC5	PC10	PC15	PC20
MLP	24.92%	24.71%	24.74%	25.10%	25.22%
Single Layer LSTM	15.76%	15.92%	17.04%	19.78%	19.17%
Dual Layer LSTM	16.55%	15.94%	14.69%	17.42%	16.21%
Multi-Layer LSTM	15.08%	15.09%	15.67%	16.27%	14.18%

The ones highlighted in green are the most efficient ones ie. maximum percentage for **Accuracy, Precision, Recall, F1 Score** and minimum for **Mean Absolute Error** and **Mean Squared Error.** As highlighted above, Multi-Layer **LSTM** with 20 Principal Components is showing best performance over others.

Technical Notes and Learnings

1. The notebook adopts modular approach of development and is designed in a way where any supported yahoo ticker can be specified and analyzed by cloning the existing one.
2. The notebook gives comparable performance without doing any feature selection analysis and model hyperparameter tuning at a security level which was done initially for FTSE.
3. Jupyter notebook was initially very slow in training DL models. Then it was found that launching notebook with 4GB buffer size increased the runtime performance. Command Line Interface (CLI) to do so:
   jupyter notebook --NotebookApp.max_buffer_size=4294967296

Important Points and Assumptions

1. There are 3 notebooks each for Index **(FTSE)**, Security **(Goldman Sachs: GS)**, and Commodity **(Silver: SI=F)**.
2. Note due to Stochastic nature of the algorithms the results captured in the above report may vary from the subsequent notebook runs. However, proper attention is made to seed the randomness of frameworks/libraries.
3. There is an expectation that with ticker specific feature selection analysis and model hyperparameter tuning, performance can be improved further.

References

- J. Brownlee, 'How to develop MLP for Time Series Forecasting', *https://machinelearningmastery.com/how-to-develop-multilayer-perceptron-models-for-time-series-forecasting/*
- C. Olah, 'Understanding LSTM networks', *https://colah.github.io/posts/2015-08-Understanding-LSTMs/*
- J. Brownlee, 'Time series prediction with LSTM RNN in python with keras', Available: *https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras*
- J. Brownlee, 'A Gentle Introduction to the Rectified Linear Unit', *https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/*
- Jingyi Shen and M. Omair Shafiq, 'Short term stock market price trend prediction using a comprehensive deep learning system', *https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00333-6*
- François Chollet (2017), Deep Learning with Python
- Wes McKinney (2018), Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython