Data Science Practical Report

Elim Yi Lam Kwan (ylk25)

1 Introductions

Being able to predict how the stock market will perform is of great demand. However, it is often challenging due to the dynamic and noisy nature of the market. Furthermore, in the process of developing a quantitative prediction model, methodologies have to be developed to extract useful information from the high dimensional financial data and the quality of these features often have a direct impact on models' performances. Extracting useful features from financial data has also been a non-trivial and long-standing research problem. Our work has been inspired by E. Hoseinzade and S. Haratizadeh's paper [1] on utilising Convolutional Neural Network (CNN) with custom filters for financial data feature extractions. The dataset used is from the UCI ML Repository [2]. In this paper, we have adopted the more realistic assumptions of their 3D model, while utilising the lower complexity 2D model framework, as well as experimenting the developed model as an encoder. The proposed CNN model has improved the F1 scores significantly from 0.4837 to 0.9854 and achieved a 98.40% accuracy. Moreover, it also brings 40% improvements in F1 scores compared to the baseline Artificial Neural Network (ANN) model and outperformed most linear models.

This practical report aims to provide a general overview of the stock market predictions pipeline developed and documented the process of formulating machine learning models for analytical problems. In the following sections, we will first give an overview of the machine learning challenge, then data exploration techniques used will be introduced. After that, the models used for dimension reductions and predictions will be discussed. Lastly, a comparison among various models will be drawn. Implementation details can also be found on the Jupyter Notebook [1].

The outline of the report is:

- Section 2: Problem Formulation
- Section 3: Data Exploration and Data Handling
- Section 4: Machine Learning Algorithms for Dimension Reduction, Prediction and Ensemble Learning
- Section 5: Performance Matrices and Evaluations
- Section 6: Conclusions

2 Problem Formulation

The financial data from UCI ML Repository [2] covers several daily features of 5 well known financial indices (where these are being referred to as 'markets' in this report) from 2010 to 2017: Dow Jones Industrial Average, NASDAQ Composite, NYSE Composite, RUSSELL 2000 and S&P 500. The project objective is to develop a model that predicts the future movement of the close price for a specific market, using data across all markets. And the project has focused on predicting the S&P

[1] https://github.com/elimkwan/Stock-Market-Predictions/

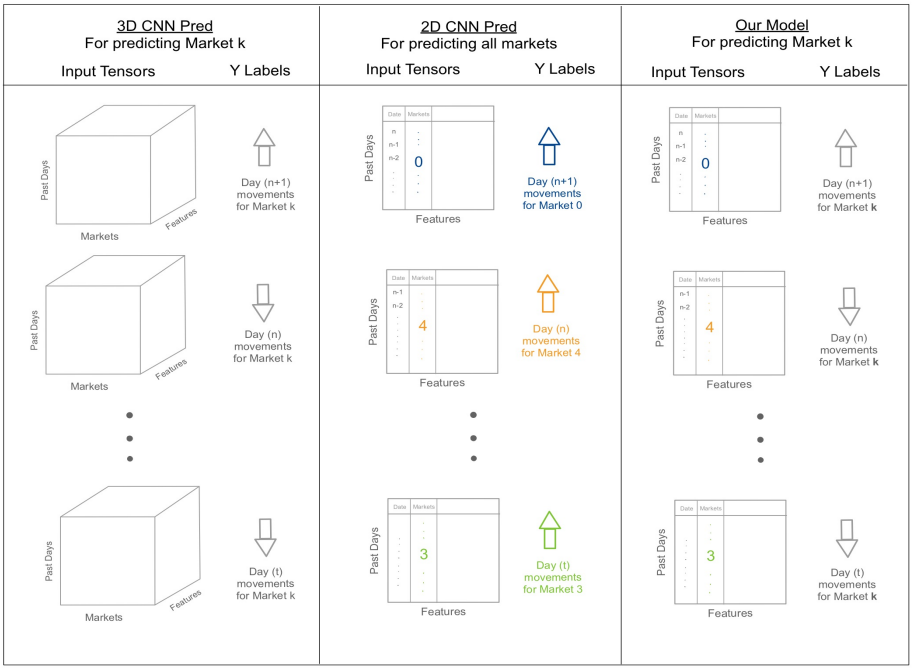

Figure 1: Illustrating the input and output data for our CNN models with respective to CNN Pred [1]

market. Moreover, in formulating the stock market prediction as a machine learning problem, there are a few high-level decisions to be made in terms of the nature of the prediction problem, data format and data selection.

Firstly, although it is common to associate price predication problem with linear regression, it has been treated as a binary classification problem in this project due to the complexity of the models involved. More specifically, the proposed model predicts the movement of the close price for the next day, with binary labels indicating increase and decrease/remains the same. Hence, label y_n for day n equals to:

$$y_n = \begin{cases} 1 & \text{if } Close_{n+1} > Close_n \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

In addition, the number of historical data we considered is 60 days, which is identical to the CNN Pred paper [1].

Secondly, in terms of the data format, in the research of E. Hoseinzade and S. Haratizadeh [1], they have proposed two models based on different assumptions. The 2D CNN model assumes the true mapping from history to future is correct for many markets, hence, training one single predictor is sufficient. Besides, the model is trained with samples from various markets for generalisation purposes. Meanwhile, the 3D model assumes different predictors are required for individual markets and each prediction model can utilise information from other markets. Although the assumptions of the 2D model may hold some sense, to maximise the system's accuracy, we would like to assume specific predictor is required for individual markets. Furthermore, to lower the complexity of the model, the 2D framework is adopted. Hence, as shown in Figure 1, the proposed model has the same input tensors as 2D CNN Pred with "Markets" being one of the features, moreover, the output labels

have been changed to predict for a specific market k.

Finally, in terms of the data selection, the features can be categorised as technical and non-technical indicators, in which, technical indicators are market-specific and non-technical indicators are mostly identical among markets for a given day. Non-technical indicators mainly cover various economic data, well known U.S. companies returns, commodities, exchange rate, future contracts, worlds stock indices etc. Although previous work [3] suspected that the close prices are more correlated with the technical indices, hence, their models only used tech indices to predict the close price. We believe that given the complexity of the market, utilising all the available data for prediction is a better approach.

3 Data Explorations and Data Handling

Understanding the data is important because it helps us gain intuition on preparing higher quality data for model training. In this section, data exploration and data handling techniques used in the project will be discussed.

In the phase of data explorations, the correlations of features with respect to the close price have been generated. We noticed that Exponential Moving Average (EMA) and Rate of Change (ROC) values are highly correlated with the close price, and commonly ranked in the top 10 most correlated features. Besides, offsets of them have been included as other features in the dataset (e.g. EMA-20, EMA-10, ROC-20, ROC-5). To reduce the dimensions of the training data, these redundant features have been removed.

Secondly, there are a number of missing values in the dataset. Features with missing values have been categorised as parameters related to stocks, histories and relative changes. Missing stocks and indices data are best approached with interpolation methods. Parameters related to historical data, such as Return of 2 days before and 5 days Rate of Change, have undefined values mainly in the initialisation phase, thus, it is most appropriate to use next valid observation to fill the gaps. For parameters that consider relative changes, e.g. Volume, they are only missing the first row, hence, it can be filled with zeros.

Another aspect of the dataset that should be handled is the categorical features because most machine learning algorithms prefer to work with numerical data. Date and Name are the only two columns with textual values in the dataset. Moreover, machine learning algorithms will automatically assume that items with similar numerical values represent similar concepts. This assumption applies to the Date feature, hence it can be simply translated into integers with LabelEncoder. However, for the Name features, one hot encoding is required, which translated the 5 market names into individual binary attributes.

In addition, the performance of machine learning algorithms typically declines when features values are of very different ranges. And this applied to our case as well with EMA in the range of 10^4 and other indices in the range of 10^{-2}. Therefore, Standardisation have been applied to all the numerical features, which implies subtracting the mean value and dividing by the variance. Since some features may have gaps that are manually filled in, to avoid significant skew in the data after scaling, Standardisation was chosen over Min-Max because it is more robust to outliers.

Finally, the dataset can be split into training, validation and testing set. Although ideally it should be stratified according to the y-labels, in this project, it was stratified according to the Name attributes, because the y-labels had not been prepared yet. Note that, the training labels up till this stage is only

data of a single day. Hence, the final step is to iterate through each row of the training labels, look up and aggregated the previous 59 days of data with the training labels into one tensor. Then, each input tensors will be of shape $[\text{past days} \times \text{features}](60 \times 82)$ as shown in Figure 1 , and we will obtain $[(\text{total number of data per market} - 60) \times (\text{number of markets})]$ numbers of them.

4 Machine Learning Algorithms for Dimension Reduction, Prediction and Ensemble Learning

In this section, the proposed CNN model will be introduced. Furthermore, various dimension reduction schemes and classification models that have been used in the project will be discussed.

4.1 Dimension Reductions with Convolutional Neural Network

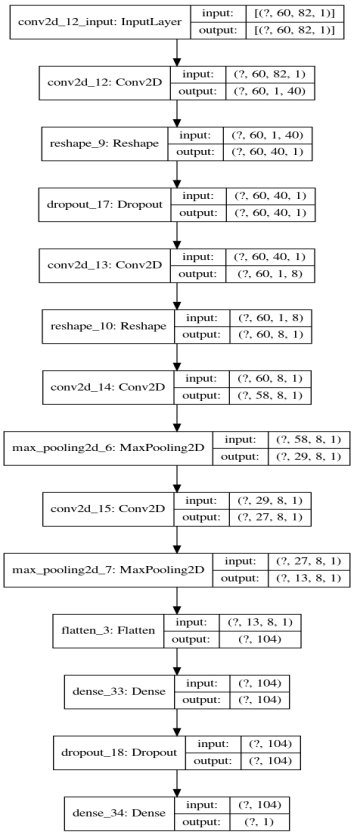

Figure 2: architecture

CNN is often used for solving computer vision tasks. With the convolution operations, it can be utilised for abstracting features into latent spaces. However, the common 3×3 filters in convolutional layers may not be appropriate to apply on financial data, since it was designed for pixels data. Hence, in this project, the CNN architecture was inspired by CNN Pred [1], in which the convolutional filters are designed based on financial facts.

The first layer of the CNN is responsible for daily feature extractions, and filters are designed with the shape [1 × number of initial features], i.e. 1×82. This helps combine the daily features into higher-

level features for representing individual days in history. In the original paper, they have reduced the dimension of each row of daily features from 82 to 8 in one step, while in this paper, the aggregation was divided into two steps, 82 to 40, then to 8.

The following few layers are responsible for durational feature extraction. Useful information from the historical data can be extracted from a time window of three consecutive days (a 3×1 filter). This setting was inspired by the candlestick patterns of Three Line Strike and Three Black Cows [4]. Each of these layers has been followed by a Max Pooling Layers to construct even more complex features. Two of such layer pairs have been applied.

All the convolution layers up till the flatten layers are the encoder part. The CNN encoder helps reduced the dimension of the data from 60×82 to 104. For the predictor, a shallow ANN has been adopted, with Relu and Sigmoid as activation functions. Note that Dropout layers have also been applied in between layers to prevent overfitting.

4.2 Dimension Reductions with Principal Component Analysis and t-Distributed Stochastic Neighbour Embedding

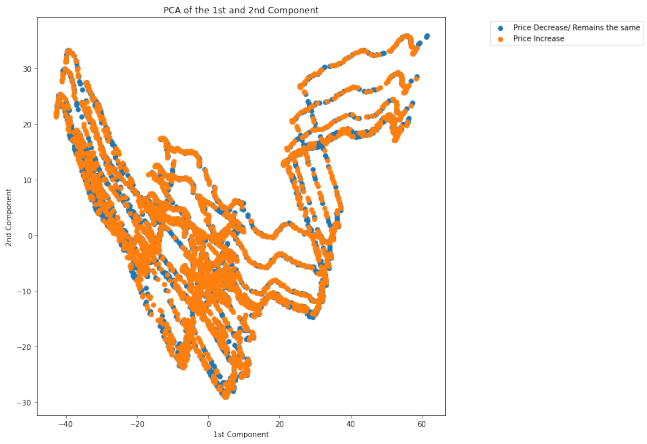

Figure 3: Data projected on 1st and 2nd Components from PCA

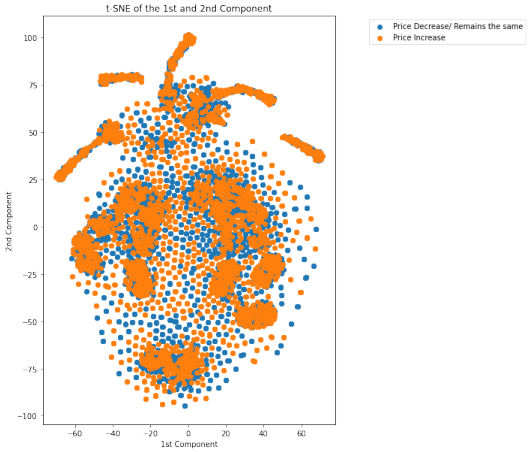

Figure 4: Data projected on 1st and 2nd Components from t-SNE

Besides CNN, conventional approaches to dimension reductions are Principle Component Analysis (PCA) and t-Distributed Stochastic Neighbour Embedding (t-SNE). In this project, both of these techniques have been implemented: a) PCA; b) PCA with t-SNE

On one hand, the principal component for a single feature refers to the best-fit line that minimises the average squared distance from data points to the line. With PCA, the principal components are first computed and each data point can be projected onto the first n^{th} principal components. For drawing a fair comparison with the CNN model, n was chosen to be 104, which is equal to the number of features obtained from the CNN encoder.

On the other hand, t-SNE is a nonlinear dimensionality reduction technique which maps high dimensional data to two/three dimensions suitable for human observation. In this project, PCA is first performed to reduce the number of features to 50 for noise reduction, then t-SNE is applied to further reduced it to two-dimensional data.

Figure 3 and 4 show the results of projecting all the data points on the first two major components computed by the PCA and t-SNE respectively. It can be shown that the major components from t-SNE are slightly better at separating the data. However, with PCA, we have the option of retaining more than two major components, indeed, it is possible to retain up to L major components, with L being the size of the initial features. The impact of these properties on the system performance will be shown in the next section.

4.3 Linear Classifiers and Ensemble Learning

As mentioned previously, a shallow ANN can act as the predictor. Moreover, classic machine learning algorithms can also be trained with the extracted features. In this project, focuses have been put on Gaussian Naive Bayes, Logistic Regression and Perceptron.

Additionally, ensemble-based learning techniques have also been considered. Ensemble learning is based on the theory that the wisdom of the crowd, i.e. the collective opinion of individuals outperform that of a single expert. In particularly, Random Forest and AdaBoost algorithms have been applied.

Random Forest is an ensemble of decision trees trained through the bagging method with extra randomness. Since one of the requirements of ensemble-based learning is that the "individuals" (classifiers) have to be sufficiently diverse in their predictions and one way to achieve it is to train the classifiers with random subsets of data. And bagging methods simply described that we are sampling with replacement.

For the Boosting algorithms, we are also trying to combine several weak learners into a strong learner. The key idea of it is to train predictors sequentially and that the next predictor tries to correct its predecessor.

5 Performance Matrices and Evaluations

Our proposed CNN encoder with a shallow ANN predictor has achieved 0.9854 F1 Scores, a 40% increase compared to the baseline model with no dimension reduction and a shallow ANN as a predictor. It is also a significant improvement compared to the CNN Pred models, which has 0.4799 (2D Model) and 0.4837 (3D Model) F1 Scores only. Moreover, the proposed model has outperformed almost all predictors and encoders tested except for the case with (AdaBoost + No Encoding). This section will entail of a discussion on the experiment set-up, performance matrices used and the final results.

5.1 Experiment Settings and Performance Matrices

Accuracy has been the most straightforward evaluation matrix for the prediction model. However, with imbalanced data, accuracy score may be biased toward the models that tend to predict the more frequent class.

Hence, F1 Score has been used as the main performance measures in the experiments, which summarises a classifier general performance in terms of Precision (P) and Recall (R). Precision measures the reliability of the classifier while Recall measures the coverage of it, and they are calculated by the True Positive (TP), True Negative (TN) and False Positive (FP) Rate. Moreover, the F1 Score

combines both of them, with $F1 = 1$ indicating the best classifiers, which is suitable for our use case.

$$P = \frac{TP}{TP + FP} \tag{2}$$

$$R = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = 1 \times \frac{P \times R}{P + R} \tag{4}$$

In terms of the CNN model training, the data has been divided into $0.7 : 0.15 : 0.15$ for training, validation and testing respectively. The CNN model was also trained with Adam optimiser, with batch size of 64 and 100 epochs.

Finally, for comparing the effectiveness of various encoder and learning algorithms, the baseline is a shallow ANN with no encoder. Moreover, since our experimental set-up and measurements are similar to the CNN Pred paper [1], comparisons will also be drawn between models.

5.2 Evaluations

5.3 Overview of the Performance of various Encoder and Classifiers

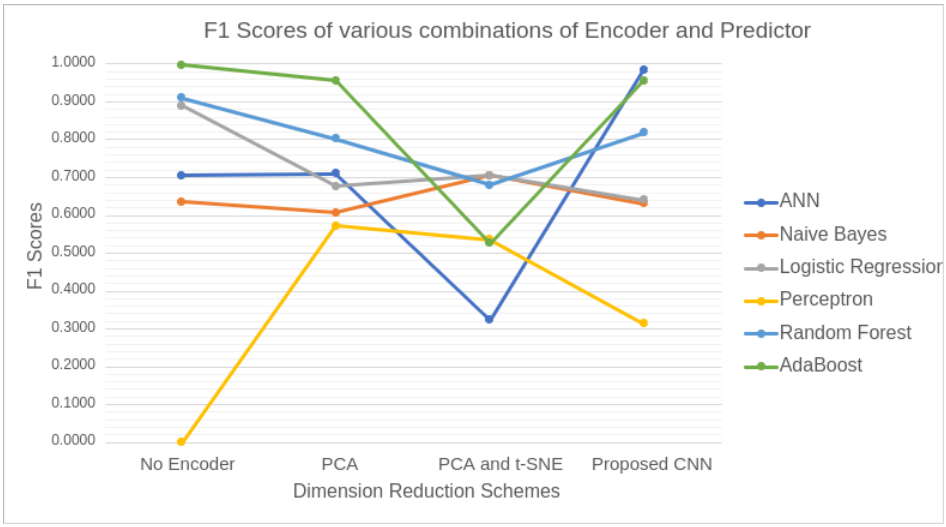

Figure 5: Resultant F1 Scores when using various combinations of dimension reduction schemes and classifiers

	ANN		Naive Bayes		Logistic Regression		Perceptron		Random Forest		Ada Boost	
	Acc (%)	F1 Score	Acc (%)	F1 Score	Acc (%)	F1 Score	Acc (%)	F1 Score	Acc (%)	F1 Score	Acc (%)	F1 Score
No Encoder	54.50	0.7056 (Base)	63.94	0.6353	87.93	0.8900	45.49	0.0000	89.18	0.9097	99.65	0.9968
PCA	67.82	0.7100	58.04	0.6069	60.75	0.6773	53.05	0.5728	73.37	0.8028	95.63	0.9602
PCA + t-SNE	46.95	0.3224	54.51	0.7056	54.51	0.7056	49.17	0.5352	52.49	0.6807	52.57	0.5933
CNN	98.40	0.9854	57.07	0.6313	57.42	0.6405	48.75	0.3138	76.35	0.8181	95.56	0.9597

Avg. F1 Scores	0.5447	0.5158	0.5827	0.2844	0.6423	0.7020

Table 1: Table showing the Accuracy and F1 Scores for adopting various combinations of Encoder and Classifiers

Our results from Table 1 indicated that the proposed CNN Encoder with shallow ANN has achieved a high F1 Scores of 0.9854, which brings a 40% improvements compared to the base model. As shown in Figure 5, it outperformed almost all predictors except AdaBoost. The performance gained provided by CNN Encoder will be further elaborated in the next section. Although accuracy matrix has not been used in the analysis, it established similar trends as the F1 scores and has been provided in Table 1 as references. Moreover, a few general trends can be observed from the results shown in Figure 5 and Table 1.

One of the most observable trends is applying classic encoder such as PCA and PCA with t-SNE often results in a decline in general model performance. This is especially true in the case of t-SNE. As shown in Figure 5, there is a significant dip in F1 scores for the PCA with t-SNE schemes. This can be explained as the dimension reduction process discard a lot of information. Especially for t-SNE, where the dimensions were reduced from 4920 to 2. This aggressive dimension reduction will inevitably impact model performance.

Furthermore, in terms of the performance of classifiers, ensemble-based learning techniques generally perform much better than linear models and shallow ANN. The average F1 Scores of AdaBoost and Random Forest is 0.7020 and 0.6423, which is higher than other models with F1 scores ranging from 0.2844 and 0.5827. The results confirmed the wisdom of a crowd theory that combining knowledge is more superior than that of a single expert.

5.3.1 Performance Gain with CNN Encoding

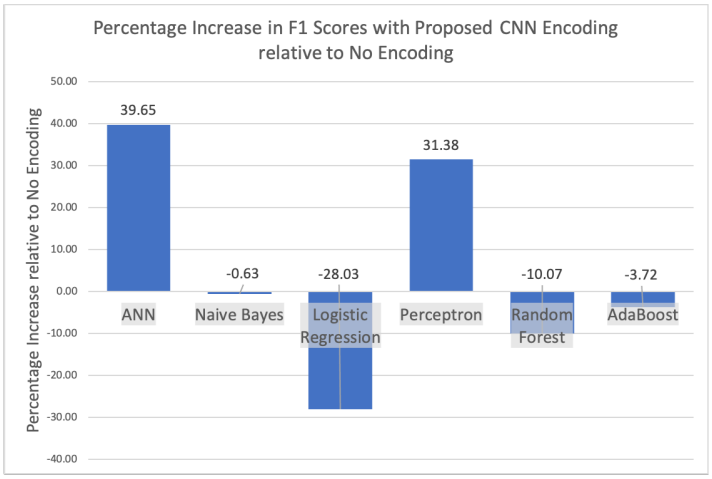

Figure 6: Percentage Increase in F1 for using the CNN Encoding relative to No Encoding

Upon discussing the general performance of the machine learning models, the impact of the proposed CNN Encoder will also be highlighted. Figure 6 summarised the effect of the the CNN Encoder. It shows that it is more effective in ANN and Perceptron models, but not for the linear models. This can be explained as the CNN Encoder and the ANN were trained jointly, hence the feature extraction has been optimised for that specific predictor. It is a supervised approach towards feature extraction, hence, it often outperforms non-supervised approaches, such as PCA and t-SNE. For the Perceptron model, the percentage increase is justified since its F1 measure started off quite low. Although the trained CNN Encoder did not generalise well with other linear classifiers, it is believed that the developed framework is valuable and joint training of the CNN Encoder with target classifiers will generate

better results.

	Proposed CNN+ANN	**2D CNN Pred [1]**	**3D CNN Pred [1]**
Model Nature	for S&P market	for all market	for S&P market
Input Shape	2D (60×82)	2D (60×82)	3D $(60 \times 82 \times 5)$
No. of Input Tensors	$5 \times n$	n	n
F1 Scores&P market	0.9854	0.4799	0.4837

* n represent the number of time series data for each markets in the original UCI ML Repository

Table 2: Table showing the F1 Scores of the proposed CNN model compared to Previous Work

Finally, comparing the developed predictor with the original CNN Pred models [1], the proposed model has significantly outperformed its predecessors as shown in Table 1. Important insights and conclusions can be drawn for neural network designs from Table 2. Readers can also refer to Figure 1 for details of the input format.

The resultant differences between the Proposed CNN and 2D CNN Pred suggested that a specific model performs much better than a general model. This can be expected as the market is quite complicated, it may require a lot more data to generate the true mapping between the markets performance and given indicators.

Moreover, the resultant differences between the Proposed CNN and 3D CNN Pred gave us an important insight that smaller input but more data points are more beneficial than large inputs with fewer data points. Although the two models process the same amount of data, the proposed CNN model has much better performance because simpler inputs were used, and other markets information were aggregated as new data points instead of new features in the third dimension. This can be explained because in our approach we have provided more domain knowledge into the pipeline, while for the 3D CNN Pred, it requires the network to find out the correlation among the additional features on its own.

6 Conclusion

In conclusion, the proposed CNN model has achieved a high F1 score of 0.9854, which doubled that of 3D CNN Pred and is 40% improvements compared to the baseline. The results also suggested the importance of input data preparation, in which simpler input but more data points is the key to performance improvements.

On top of proposing a CNN model for stock market predictions, this report also demonstrated how machine learning pipelines can be designed to solve specific analytical problems.

Future work includes exploring the joint training of the proposed CNN encoder with various linear models, developing an LSTM model for the stock market predictions and adapting the framework for predicting close price as a linear regression task.

References

[1] Ehsan Hoseinzade and Saman Haratizadeh. *CNNPred: CNN-based stock market prediction using several data sources*. 2018. arXiv: 1810.08923 [cs.LG].

[2] Ehsan Hoseinzade and Saman Haratizadeh. "CNNpred: CNN-based stock market prediction using a diverse set of variables". In: *Expert Systems with Applications* 129 (Mar. 2019), pp. 273–285. DOI: 10.1016/j.eswa.2019.03.029.

[3] Yakup Kara, Melek Boyacioglu, and Omer Baykan. “Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange”. In: *Expert Systems with Applications* 38 (May 2011), pp. 5311–5319. DOI: 10.1016/j.eswa.2010.10.027.

[4] Thomas N. Bulkowski. *Encyclopedia of candlestick charts*. 2012.