**Faculty of Computers &
Artificial Intelligence**

Benha University

House Price Prediction Using Machine and Deep Learning Models

Final Project

In

Introduction to Big Data

Abdallah Tamer Mohamed El Ghamry

Supervised by

Dr. Mustafa Abdul Salam

February 2021

Table of Contents

1. Abstract

Predicting the price of housing has an enormous importance for near-term economic forecasting of any nation. In an uncertain economic climate, construction companies are confronted with a daunting question: to build or not to build. Little research is reported in the construction literature on the price of housing. In this paper, we have discussed the prediction of the price of housing using some machine and deep learning models such as: SVM, KNN, MLP, ANNs, CNNs, LSTMs, GRUs, LightGBM, and CatBoost. The models incorporate time-dependent and seasonal variations of the variables. The output of the model is an approximate price of a house given the features of the house. The models showed promising results on predicting housing prices.

2. Introduction

To find research on the price of housing, one needs to search mostly the business, economic, finance, and real estate journals. A number of researchers have attempted to describe variables affecting the real state price dynamics or movements. Accurate cost estimation in early stages of construction projects leads to cost savings, thus contributing to a more sustainable project. The estimated cost is commonly computed based on the cost of project determinants such as construction materials, labor, equipment, and method. The construction cost depends on many other factors such as the project locality, type, construction duration, scheduling, and extent of use of recycled materials.

3. Data Preprocessing

Data preparation is the process of cleaning and transforming raw data prior to processing and analysis. It is an important step prior to processing. Estimation of the price of the house is based on building characteristics factors, structural and architectural factors, financial properties factors, social factors, governmental factors, physical/environmental factors, and economic factors. The following table shows 27 different factors control the price of the house. There are 1107 records in the dataset.

Table 1: Features of each house in the dataset.

input number	Descriptions
1	Total land size.
2	Gross area of unit.
3	House age.
4	Land value.
5	Location type.
6	Building type.
7	Exist elevator.
8	Distance from malls.
9	Distance to road.
10	Number of bedrooms.
11	Number of bathrooms.
12	Garage.
13	Garden share.
14	Level of finishing.
15	Preliminary estimated construction.
16	Duration of construction.
17	Price of the unit at the beginning.
18	Population trends in the city.
19	Standard level in the region.
20	Quality of schools
21	Quality of services.
22	Transportation.
23	Interest rate.
24	Inflation rate.
25	Economic climate.
26	Consumer price index (CPI).
27	Months in market.

The following table summarizes important statistics about the dataset.

Table 2: Summary statistics of the dataset.

Number of features	27
Number of records	1107
Maximum price of house	2127500 L.E.
Q1	627500 L.E.
Median price of house	762000 L.E.
Q3	908250 L.E.
Minimum price of house	199000 L.E.
Average price of house	772228 L.E.

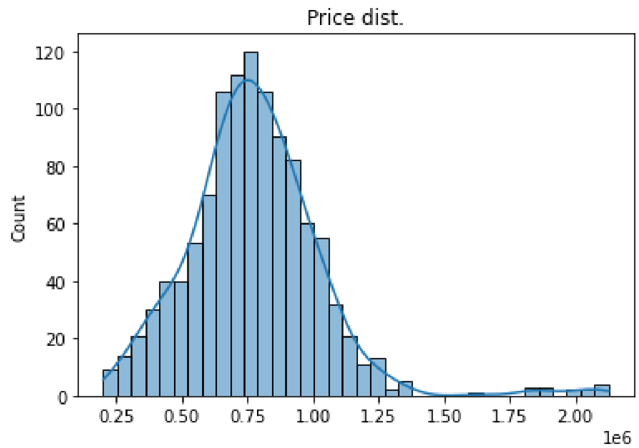

Figure 1 distribution of the prices.

3.1 Removing outliers

The following boxplot shows that there are some outliers in the dataset.

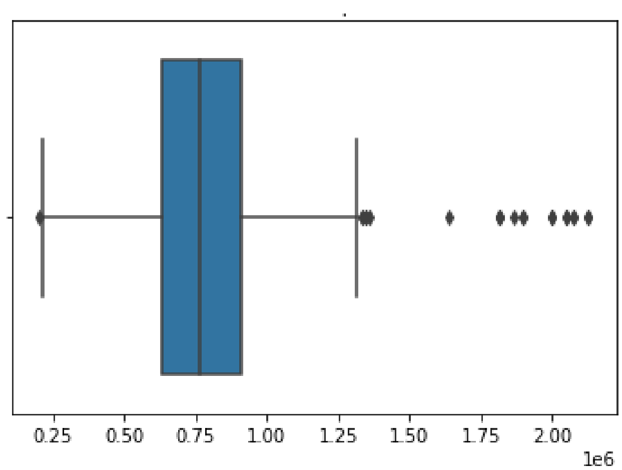

Figure 2 boxplot of the prices.

After removing the outliers the distribution of the prices is approximately normal distribution.

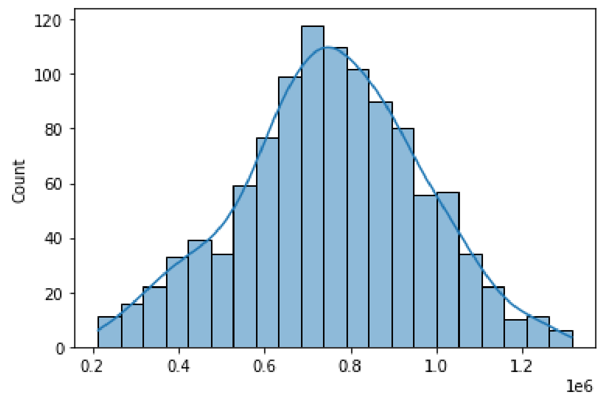

Figure 3 distribution of the prices after removing outliers.

3.2 Splitting the dataset into the training set and test set

The dataset has been split as 80% for training and cross validation and 20% for testing. The cross validation is applied to handle overfitting problem. Overfitting a model result in good accuracy for training data but poor results on the yet-to-be-seen data (test data).

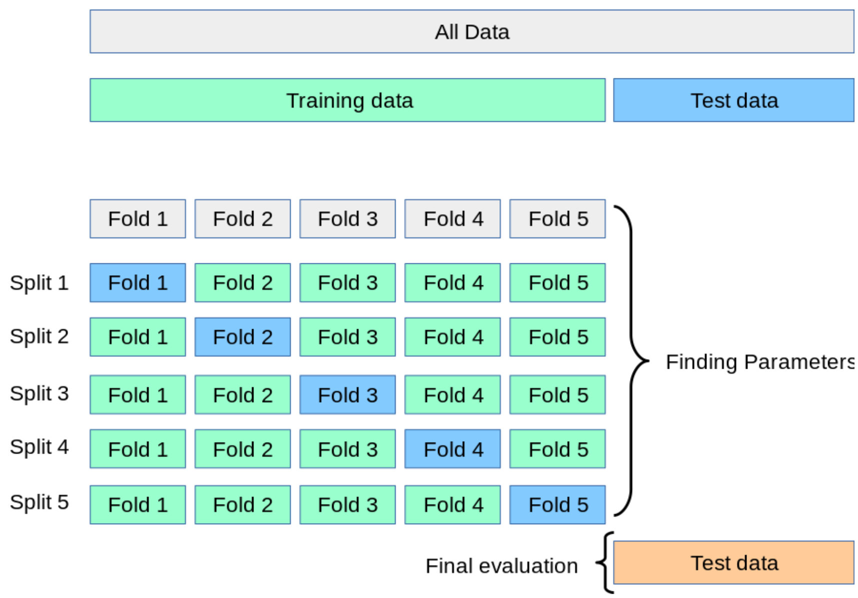

Figure 4 data splitting.

3.3 Feature scaling

Since the dataset has highly varying values, we need to scale it. Standardization is a very effective technique which re-scales the values so that it has distribution with 0 mean value and variance equals to 1. The standardization is applied using the equation:

$$z = \frac{x - \mu}{\sigma}$$

Where μ is the mean and σ is the standard deviation.

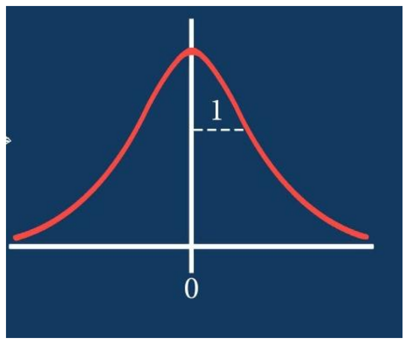

Figure 5 standardization.

4. Model Planning

Since the problem is regression, there are many solutions for it. After searching, we have selected 9 of the most accurate machine and deep learning algorithms for solving the problem.

Table 3: Candidate machine and deep learning models

Support Vector Machine (SVM)
K-Nearest Neighbors (KNN)
Multi-Layer Perceptron (MLP)
Artificial Neural Networks (ANNs)
Convolutional Neural Networks (CNNs)
Long Short-Term Memory (LSTM) Networks
Gated Recurrent Units (GRUs) Networks
LightGBM
CatBoost

5. Model Building

5.1 Support Vector Machine (SVM)

Support vector regression (SVR) is an effective tool in real-value function estimation. As a supervised-learning approach, SVR trains using a loss function, which penalizes high and low misestimates. SVR uses kernels such as: linear, polynomial, RBF and sigmoid.

The best parameters are obtained by using the grid search.

Table 4: SVM cross validation scores with RBF kernel regularization parameter C=10.

Split	Score
Split 1	0.91826
Split 2	0.93880
Split 3	0.85614
Split 4	0.93214
Split 5	0.85069
Split 6	0.93452
Split 7	0.92541
Split 8	0.94486
Split 9	0.91649
Split 10	0.95006
CV Score	0.91674

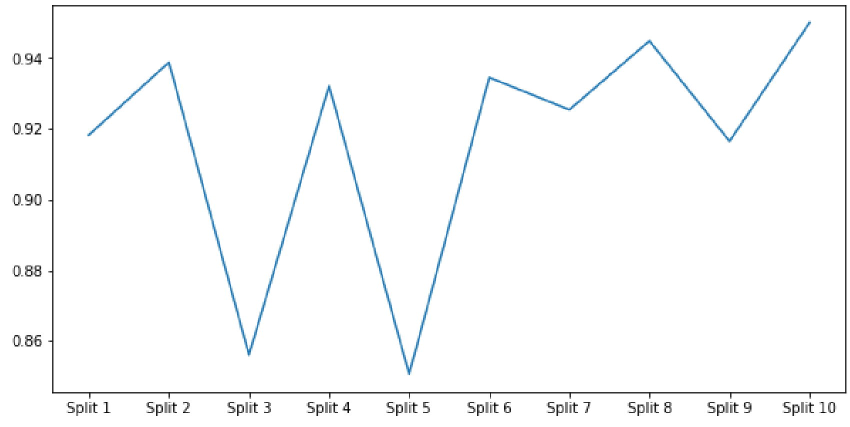

Figure 6 SVM cross validation scores.

Table 5: SVM performance on the testing data.

Metric	Score
R2 Score	0.926817
MAE	39120.636632
RMSE	56632.300182

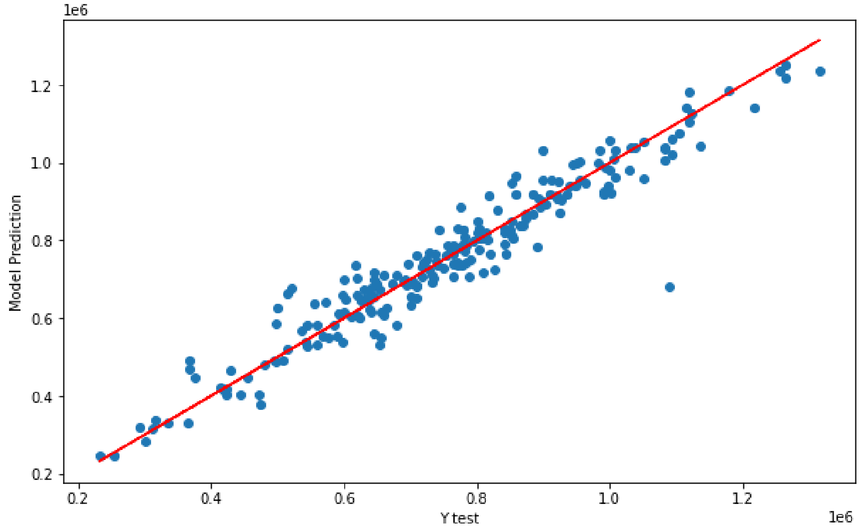

Figure 7 SVM performance on the testing data.

5.2 K-nearest neighbors (KNN)

KNN algorithm stores all available cases and classifies new cases based on distance function. The best parameters are obtained by using the grid search.

Table 6: KNN cross validation scores with Manhattan distance and 2 neighbors.

Split	Score
Split 1	0.92872
Split 2	0.90458
Split 3	0.85733
Split 4	0.89543
Split 5	0.82855
Split 6	0.89616
Split 7	0.86924
Split 8	0.92260
Split 9	0.89145
Split 10	0.91793
CV Score	0.89120

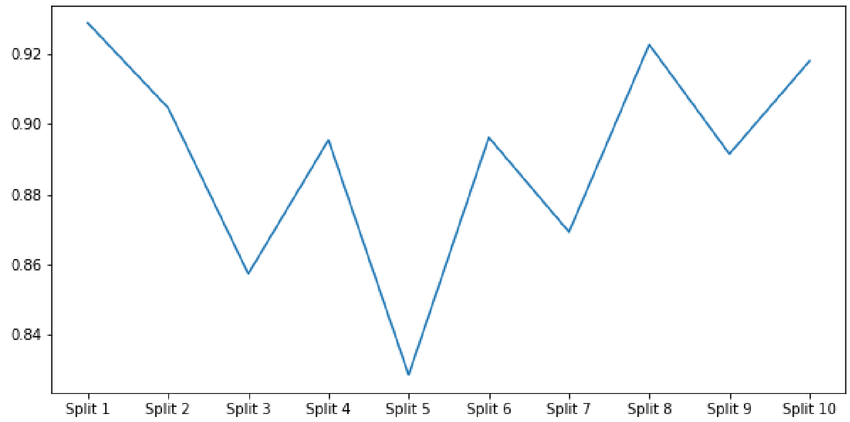

Figure 8 KNN cross validation scores.

Table 7: KNN performance on the testing data.

Metric	Score
R2 Score	0.921054
MAE	47992.505187
RMSE	58819.751581

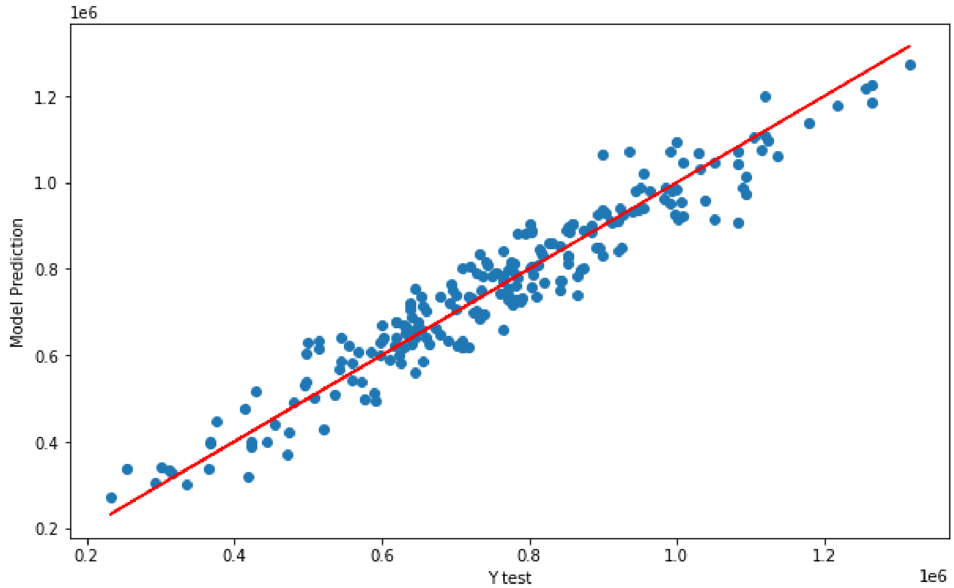

Figure 9 KNN performance on the testing data.

5.3 Multi-layer perceptron (MLP)

A multilayer perceptron (MLP) is a deep learning model which composed of more than one perceptron. It is composed of an input layer, an output layer that makes a prediction about the input, and in between there is a number of hidden layers.

The best parameters are obtained by using the grid search.

Table 8: MLP cross validation scores with tanh activation function, Adam optimizer, and 100 units for each hidden layer.

Split	Score
Split 1	0.92006
Split 2	0.93396
Split 3	0.86432
Split 4	0.93197
Split 5	0.86271
Split 6	0.94622
Split 7	0.89546
Split 8	0.93025
Split 9	0.89680
Split 10	0.92724
CV Score	0.91090

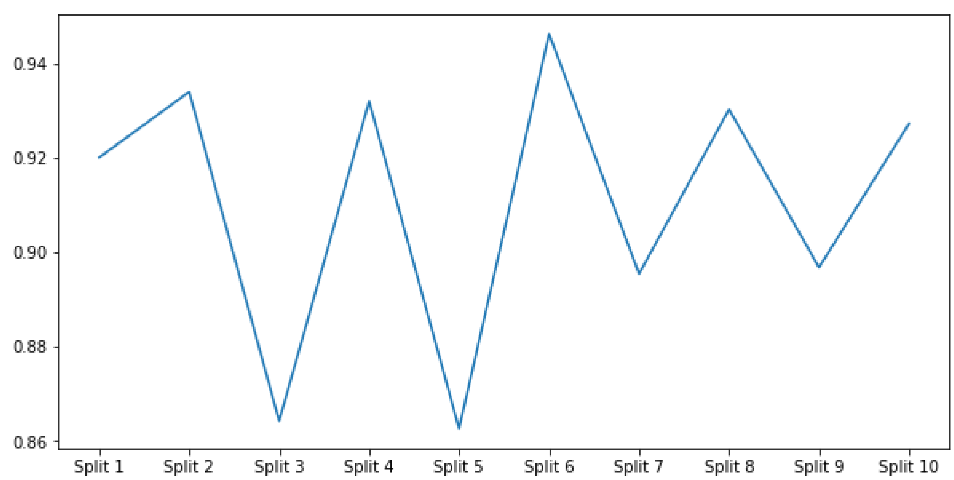

Figure 10 MLP cross validation scores.

Table 9: MLP performance on the testing data.

Metric	Score
R2 Score	0.922150
MAE	42354.136897
RMSE	58409.951576

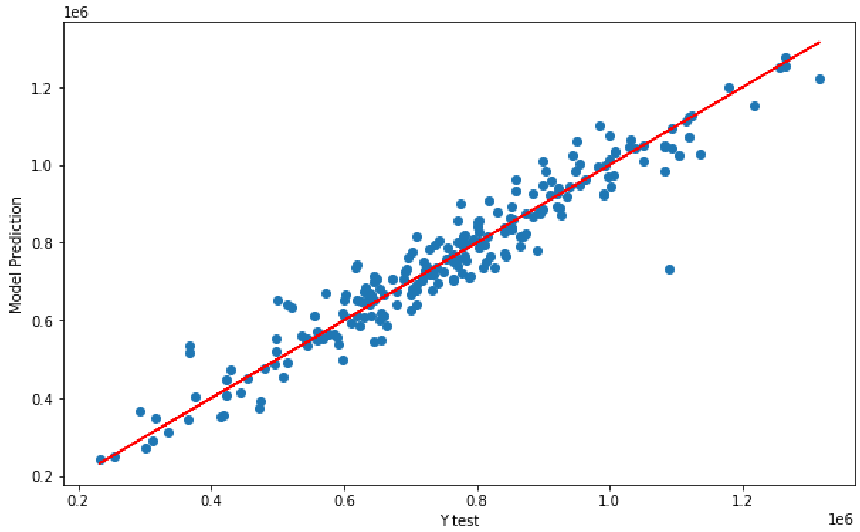

Figure 11 MLP performance on the testing data.

5.4 Artificial Neural Network (ANN)

ANN is a biologically inspired sub-field of AI modeled after the brain. An Artificial neural network is a computational network based on biological neural networks that construct the structure of the human brain.

Table 10: ANN grid search and cross validation results.

Split	Score
Optimizer	RMS prop
Epochs	100
CV Score	0.91381

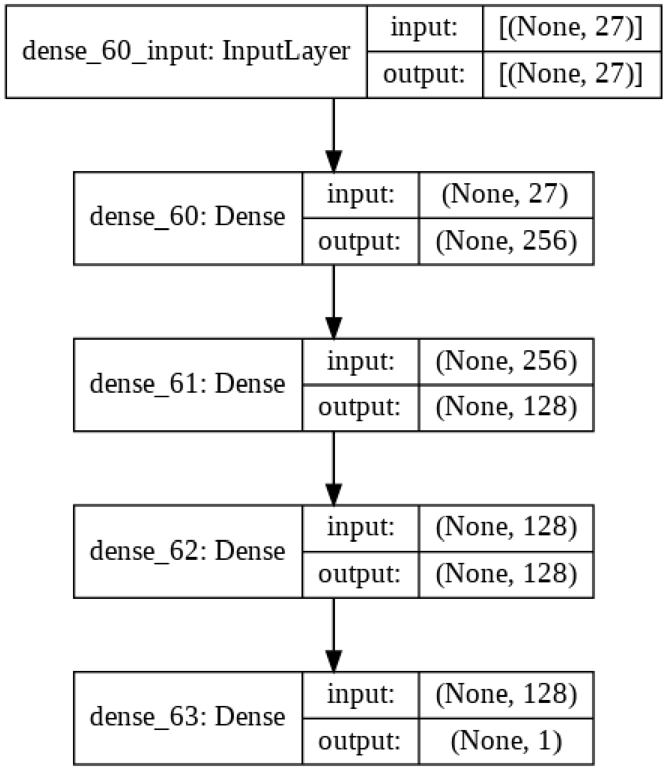

Figure 12 ANN architecture.

Table 11: ANN performance on the testing data.

Metric	Score
R2 Score	0.922918
MAE	38966.350201
RMSE	58121.083906

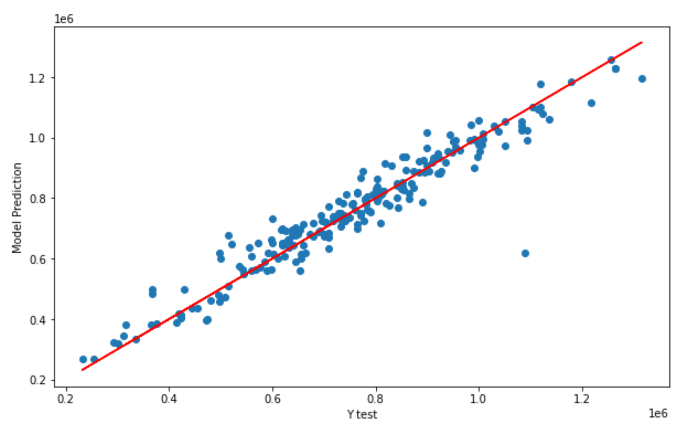

Figure 13 ANN performance on the testing data.

5.5 Convolutional Neural Networks (CNN)

Convolutional neural network (CNN) is a class of ANNs. CNN is designed to automatically and adaptively learn spatial hierarchies of features through backpropagation by using multiple building blocks, such as convolution layers, pooling layers, and fully connected layers.

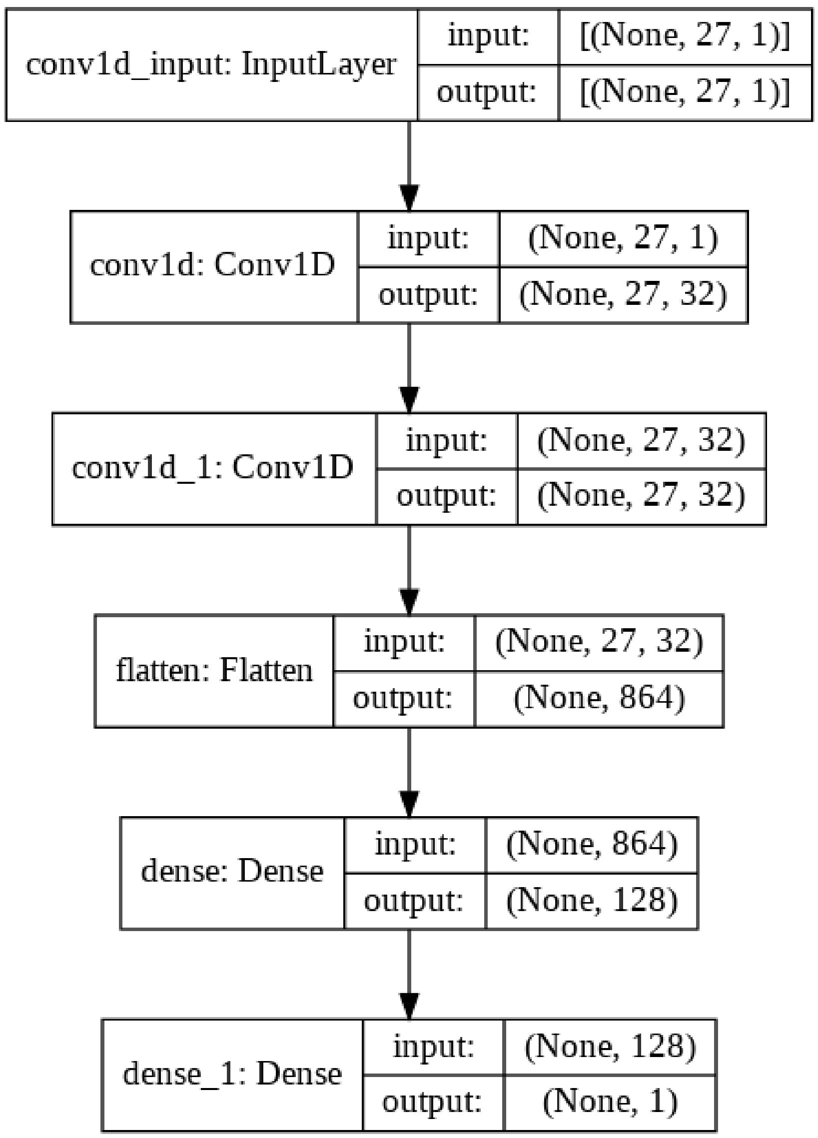

Figure 14 CNN architecture.

Table 12: CNN cross validation results.

Split	Score
Optimizer	Adam
Epochs	195
CV Score	0.91045

Table 13: CNN performance on the testing data.

Metric	Score
R2 Score	0.944156
MAE	35784.833142
RMSE	49470.28617

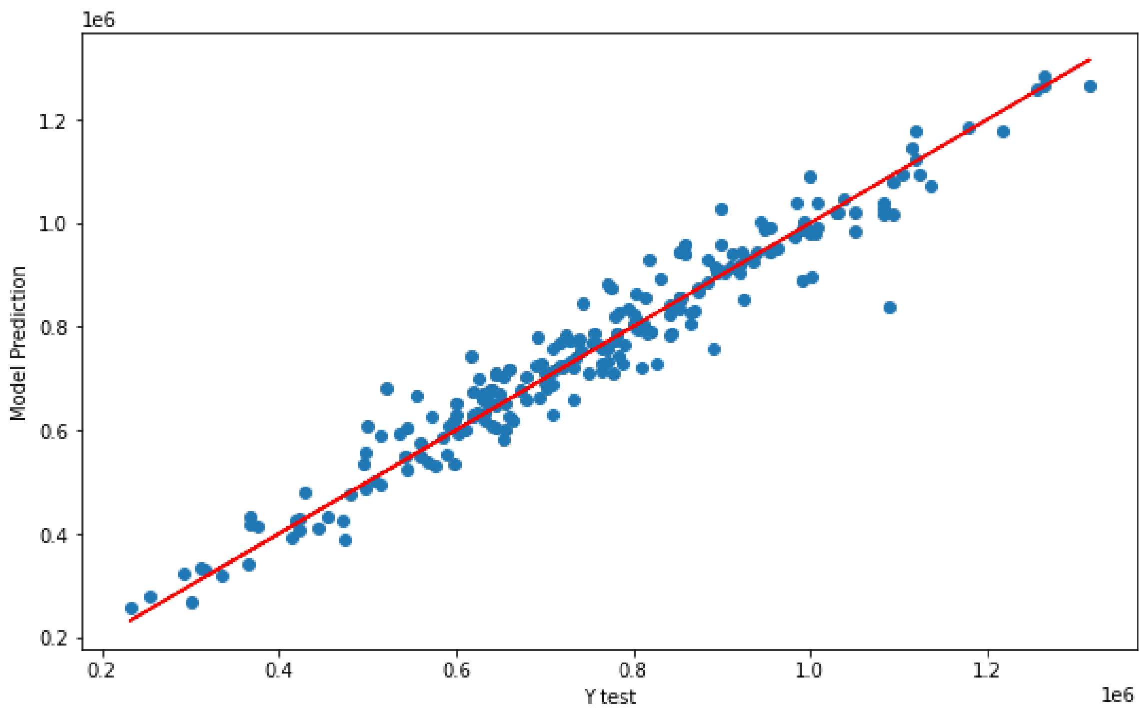

Figure 15 CNN performance on the testing data.

5.6 Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM) is a special kind of Recurrent Neural Network (RNN), capable of learning long-term dependencies. They work tremendously well on a large variety of problems. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior.

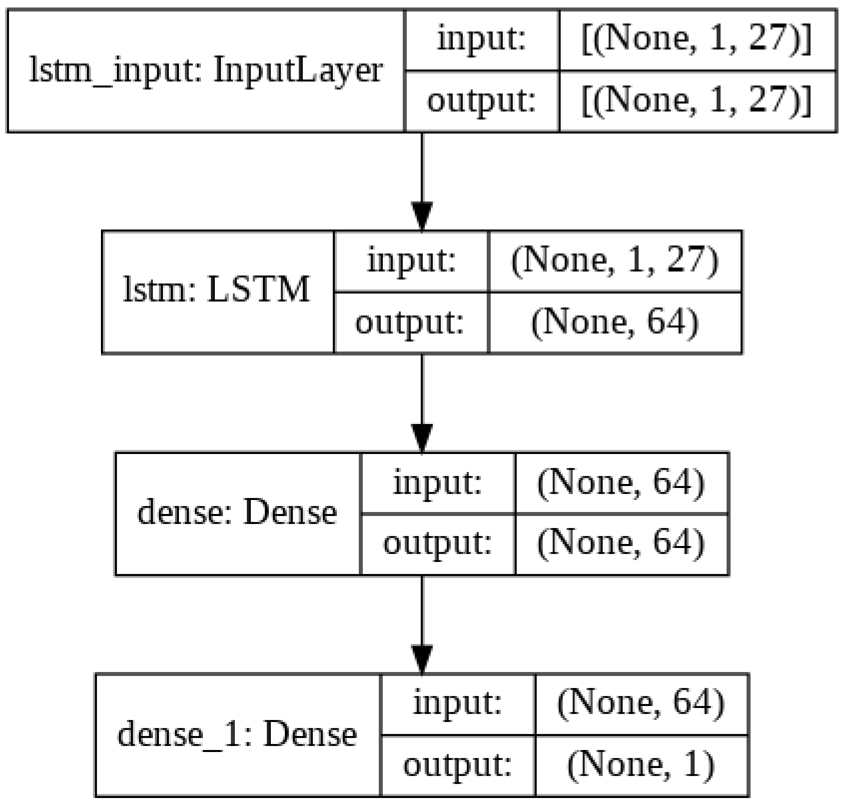

Figure 16 LSTM architecture.

Table 14: LSTM cross validation results.

Split	Score
Optimizer	Adam
Epochs	200
CV Score	0.90011

Table 15: LSTM performance on the testing data.

Metric	Score
R2 Score	0.930847
MAE	38415.125000
RMSE	55050.883954

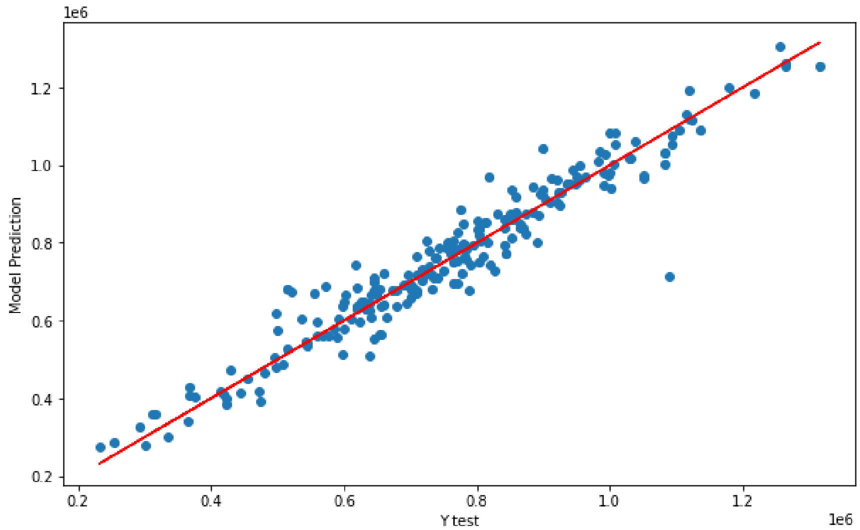

Figure 17 LSTM performance on the testing data.

5.7 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) is a type of Recurrent Neural Network (RNN) that addresses the issue of long term dependencies which can lead to vanishing gradients. GRUs address this issue by storing "memory" from the previous time point to help inform the network for future predictions.

Table 16: GRU cross validation results.

Split	Score
Optimizer	Adam
Epochs	200
CV Score	0.90576

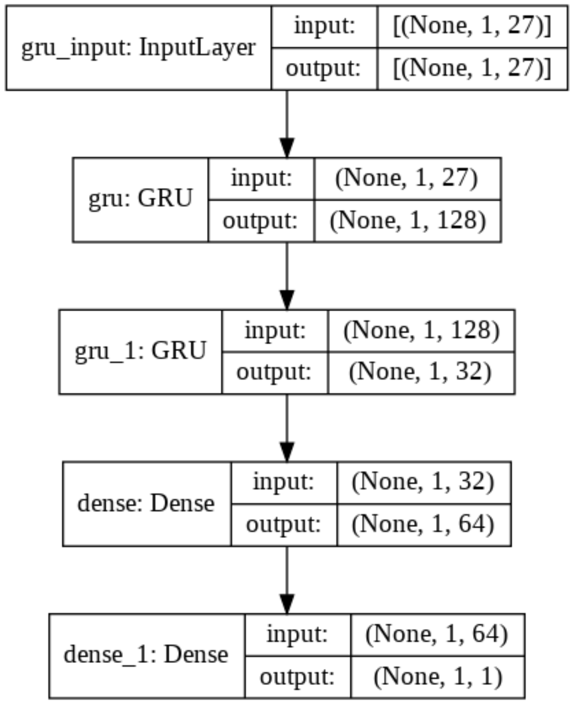

Figure 18 GRU architecture.

Table 17: GRU performance on the testing data.

Metric	Score
R2 Score	0.942044
MAE	36951.294868
RMSE	50397.353193

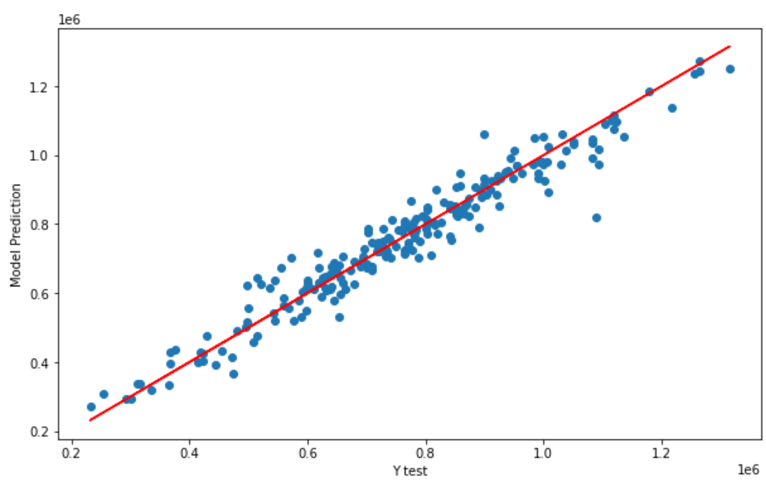

Figure 19 GRU performance on the testing data.

5.8 LightGBM

LightGBM is a gradient boosting framework that uses tree-based learning algorithms. It is efficient, uses less memory, achieves higher accuracy, and fast to train.

Table 18: LightGBM cross validation scores

Split	Score
Split 1	0.90564
Split 2	0.92836
Split 3	0.87427
Split 4	0.93926
Split 5	0.88372
Split 6	0.91254
Split 7	0.91142
Split 8	0.93828
Split 9	0.93286
Split 10	0.94686
CV Score	0.91732

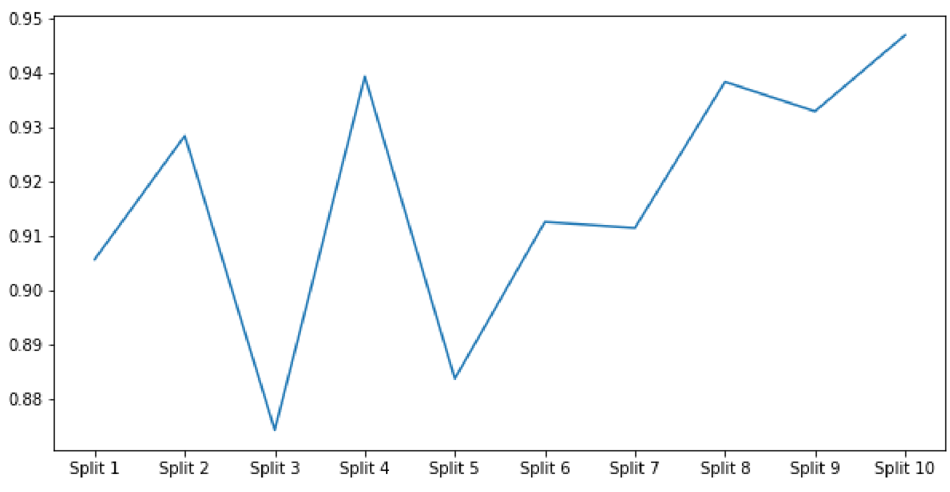

Figure 20 5.8 LightGBM cross validation scores.

Table 19: LightGBM performance on the testing data.

Metric	Score
R2 Score	0.927664
MAE	43026.177840
RMSE	56303.542295

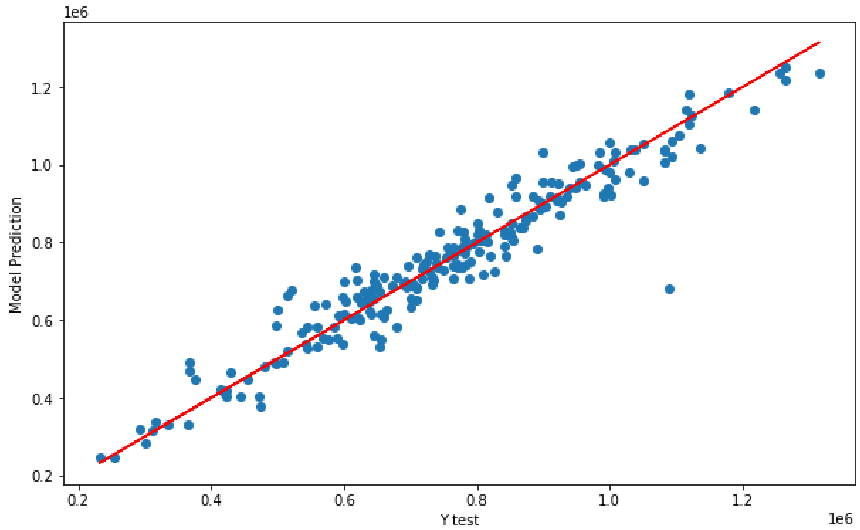

Figure 21 LightGBM performance on the testing data.

The best parameters are obtained by using the grid search.

Table 20: LightGBM grid search results.

Parameter	Value
Booster	gbtree
Feature fraction	0.5
Learning rate	0.05
Estimators	1000
Num leaves	31
Reg alpha	0.1
Bagging fraction	0.75

5.9 CatBoost

CatBoost is an algorithm for gradient boosting on decision trees.

Table 21: CatBoost cross validation scores

Split	Score
Split 1	0.90957
Split 2	0.93407
Split 3	0.87507
Split 4	0.94595
Split 5	0.88275
Split 6	0.92512
Split 7	0.91238
Split 8	0.93417
Split 9	0.91867
Split 10	0.93347
CV Score	0.91712

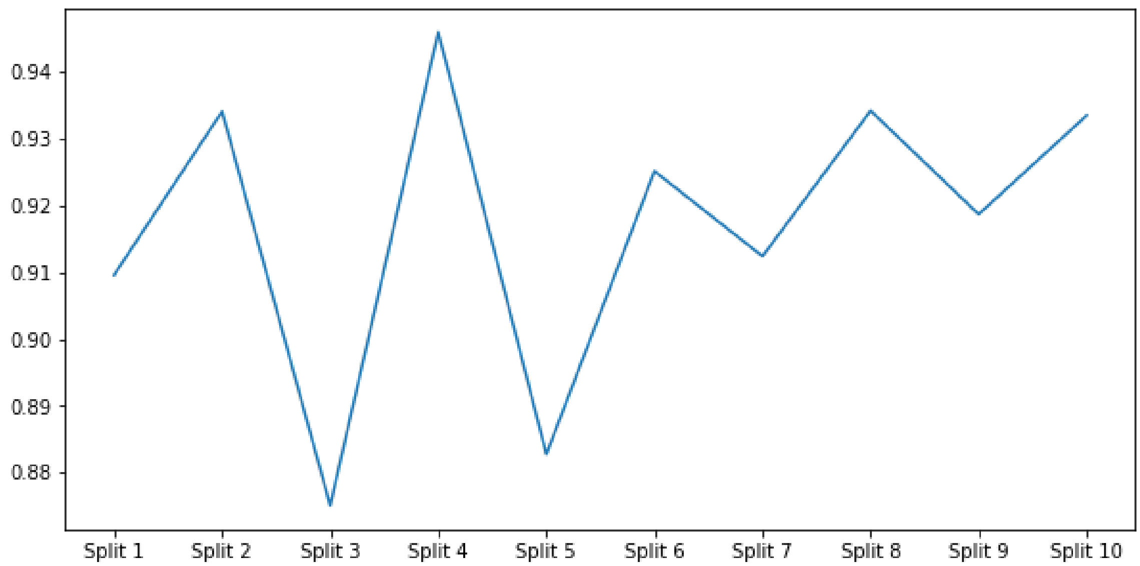

Figure 22 CatBoost cross validation scores.

Table 22: CatBoost performance on the testing data.

Metric	Score
R2 Score	0.929133
MAE	42669.890923
RMSE	55728.831378

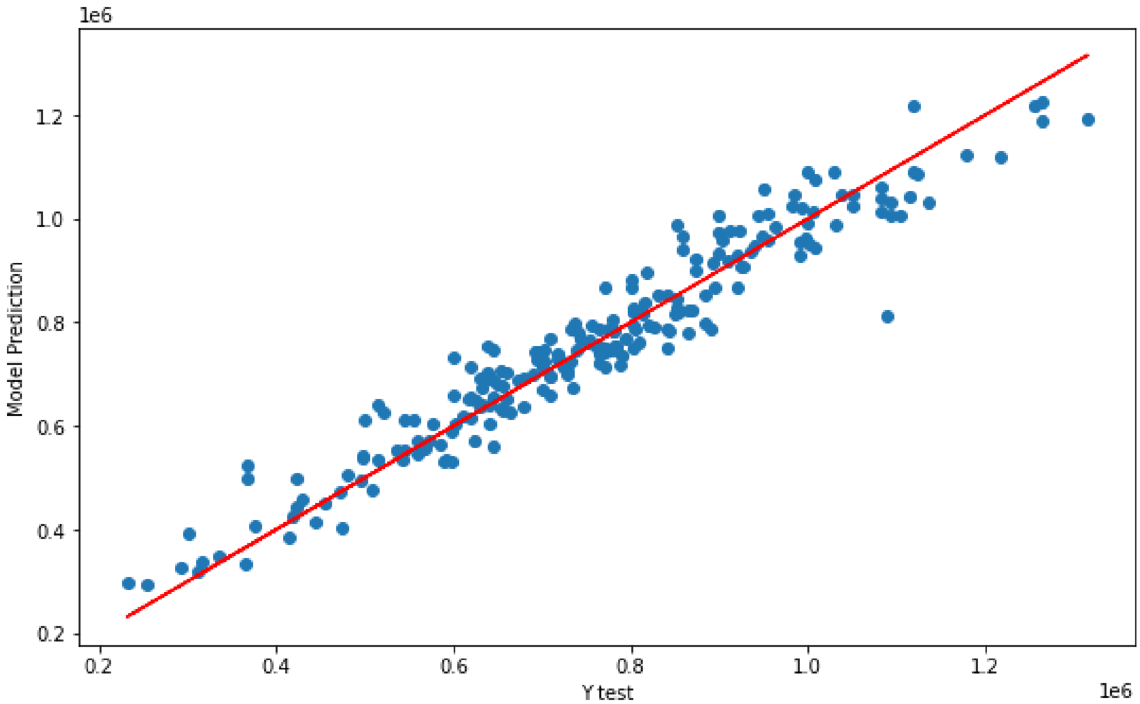

Figure 23 CatBoost performance on the testing data.

The best parameters are obtained by using the grid search.

Table 23: CatBoost grid search results.

Parameter	Value
Iterations	2000
Loss function	RMSE
Depth	6
Border count	64
L2 leaf reg	3

6. Communicate Results

6.1 Metrics

We used three metrics to evaluate the performance of each model.

6.1.1 Mean absolute error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right|$$

6.1.2 Root mean square error (RMSE)

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{(\hat{y}_i - y_i)^2}{n}}$$

6.1.3 R-Squared (R^2)

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

6.2 Models Results

The CNN model achieves the best R-squared, MAE, and RMSE, while the LGBM model achieves the best cross validation score score.

Table 24: Models performance summary.

Model	CV Score	R-Squared	MAE	RMSE
SVM	0.91674	0.926817	39120.64	56632.300
KNN	0.89120	0.921054	47992.51	58819.751
MLP	0.91090	0.922150	42354.14	58409.952
ANN	0.91381	0.922918	38966.35	58121.084
CNN	0.91045	0.944156	35784.83	49470.286
LSTM	0.90011	0.930847	38415.125	55050.884
GRU	0.90576	0.942044	36951.295	50397.353
LGBM	0.91732	0.927664	43026.178	56303.542
CatBoost	0.91712	0.929133	42669.891	55728.831

7. Conclusion

A purpose of the paper was to introduce a comparison among different machine learning and deep learning models to help in the field of construction estimation. Concern about such data is the proportion of the number of data sets to the number features. The higher dimensionality, the larger the number of training samples required for accurate cost estimation. The models were presented for predicting the price of a building. This will minimize the cost and the effort to find a suitable house.

References

[1] How to Grid Search Hyperparameters for Deep Learning Models https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/

[2] Regression with the Deep Learning https://machinelearningmastery.com/regression-tutorial-keras-deep-learning-library-python/

[3] How to Fit Regression Data with CNN Model https://www.datatechnotes.com/2019/12/how-to-fit-regression-data-with-cnn.html

[4] Time Series Prediction with LSTM Recurrent Neural Networks https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/

[5] Recurrent Neural Networks in Python https://towardsdatascience.com/recurrent-neural-networks-by-example-in-python-ffd204f99470

[6] Support Vector Regression https://link.springer.com/chapter/10.1007/978-1-4302-5990-9_4

[7] Gradient Boosting with Scikit-Learn, XGBoost, LightGBM, and CatBoost https://machinelearningmastery.com/gradient-boosting-with-scikit-learn-xgboost-lightgbm-and-catboost/