Advanced Machine Learning Algorithms in Future Sales Prediction

Mouratidis Konstantinos
School of Science & Technology
International Hellenic University
Thessaloniki, Greece
k.mouratidis@ihu.edu.gr

Tzimtzimis Manolis
School of Science & Technology
International Hellenic University
Thessaloniki, Greece
m.tzimtzimis@ihu.edu.gr

***Abstract* – Over the last few years many companies are focusing on the Data Science field and more specific on the Advanced Machine Learning area, in order to predict their future sales or even to extract knowledge of their historical data. The following paper is one of the examples for future sales predictions. Several techniques were applied and tested, as well as some attributes were removed or added afterwards. Finally, some charts and plots explain the differences that each alteration originated.**

***Keywords*— Regression, Keras, Machine Learning, Neural Networks, Voting , RNN, LSTM**

I. Introduction

The need for predictions of the future sales have been a mandatory subject as businesses try to become more competitive in market. This necessity resulted a high demand on data-driven decision-making, with the use of Neural Networks and other Deep Learning methods. The sales are one of the most important parameters for a company's longevity. For this reason, it was important to proceed with a Machine Learning approach to a given dataset, so to predict the future sales for a specific company that provided the data. The targets of this paper were the following:

- Understand the given dataset and explain the meaning of it
- Subtract any useless features, and generate new ones
- Apply Machine Learning Methods to predict Future Sales
- Use some charts and plots to evaluate the most important factors of the process.

The structure of the paper was organized as comprehensible as possible to highlight all the important steps of each process. The first Section defines the problem of this implementation and Section II introduces the given dataset and all of the available features within it. Section III describes the algorithms that were used through the entire process as well as the different approaches on the preprocessing phase before each algorithm was applied. The Section IV presents the optimal model and discusses the results found. The last section (V) summarizes the conclusions of the advanced machine learning methodology.

II. Data and problem description

A. Dataset Description

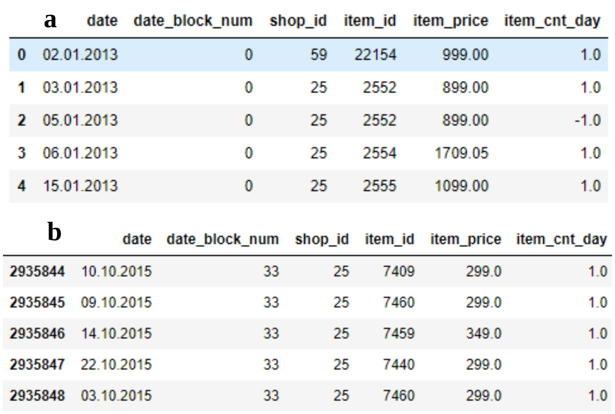
The given dataset was published in one of the Kaggle's competitions and was provided by one of the largest Russian software firms – 1C Company [1]. The dataset consists of different data frames that are linked with each other by using unique ID codes. Two of the data-frames ("Shop.csv", "Items.csv") include details, namely the ID code and the name, of each shop and each item respectively. One other dataframe ("Item_Categories.csv") comprises the same information (name and ID) for the category that every item belongs to. The "Sales_train_v2.csv" contains all the important information of the sales in the duration from September 2013 until October 2015. These are:

- A unique number for every month (0 corresponds to September 2013, and 33 to October 2015)
- The number of items that sold in a specific day
- The date of every registration

Finally, it links all the previous dataframes by using their ID numbers.

In Machine Learning processes, there are two different datasets being used in almost every application. The one is known as "train set" and the other as "test set". The train set is the total of all the independent variables, whereas the test set is the total of the dependent variables, also referred to as "target set". This model is a Regression Problem case, so the target value is a real-value variable.

The first step before any data preprocessing was to detect whether or not there are any missing values in the dataset. It was distinguished that none of the aforementioned dataframes had missing values, which is not always a real-case scenario, yet it is really useful to handle. Figure 1 demonstrates the first five and the last five tuples of the initial train set.



a

	date	date_block_num	shop_id	item_id	item_price	item_cnt_day
0	02.01.2013	0	59	22154	999.00	1.0
1	03.01.2013	0	25	2552	899.00	1.0
2	05.01.2013	0	25	2552	899.00	-1.0
3	06.01.2013	0	25	2554	1709.05	1.0
4	15.01.2013	0	25	2555	1099.00	1.0

b

	date	date_block_num	shop_id	item_id	item_price	item_cnt_day
2935844	10.10.2015	33	25	7409	299.0	1.0
2935845	09.10.2015	33	25	7460	299.0	1.0
2935846	14.10.2015	33	25	7459	349.0	1.0
2935847	22.10.2015	33	25	7440	299.0	1.0
2935848	03.10.2015	33	25	7460	299.0	1.0

Figure 1 a) First Five Tuples and b) Last Five Tuples of the Initial Train Set

It is essential to mention that the items' names were all in Russian language. Thought it was one of the features that were not considered important, so no translation was made. Moreover, the price of each item is on a daily basis, while the prediction must be made on a month basis criterion. The importance of that configuration is described later on the Data preprocessing stage.

B. Data preprocessing

The data preprocessing is the most significant and critical step of every machine learning application. The manipulation of the data, and how they are "cleaned" or not, can result to a confused dataset and though, to a difficult to handle process. It can reduce the complexity of the data, offers better conditions for further analysis and the data become more understandable [2].

Although different techniques were applied to the model, it was later found out that the accuracy of the models could not exceed the accuracy of the already used Kernels in the Kaggle competition. For this reason, some Kernels was used in the preprocessing stage, but several changes were made in order to detect whether a better accuracy could be achieved or not. Nevertheless, some small attempts were initially generated without the Kernels. Figure 2 illustrates that in the first step all the different dataframes joined into a single one.

After joining all the dataframes, some attributes were created in order to make the algorithms more efficient. The average price per category, the average and median values of the prices and the sales, the mean and the median values of sales as well as the categorization of the items were all extra features that added to the dataset.

To test those attributes three different algorithms where implemented.

a) Random Forest Regressor

The Random Forest Regressor is a meta estimator that uses several sub-samples of the dataset in order to fit its classified decision trees and by using the average values improves the accuracy of the model. [3]

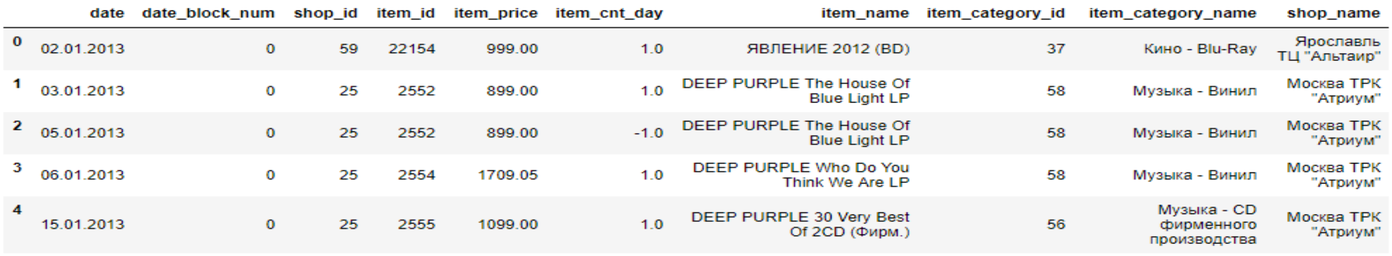
b) Linear Regression Model

The Linear Regression Model tries to create a model that represents the relationship between an explanatory variable and a dependent variable. Therefore, it must always exist a relationship (not necessary causation) between the two variables. [4]

c) Ridge Regression Model

The Ridge Regression is used when a model has multicollinearity problems. Due to multicollinearity the variances of the estimated least squares are large, although the least squares by themselves are unbiased. With the Ridge Regression the standard error can be reduced, by implementing a degree of bias in the model [5].

The results of this first attempt of the model without using a Kernel are presented in Figure 3. It is observable that the best model was fitted with the Linear Regressor. Additionally, the extra features decreased the performance of the model, instead of increasing it.



	date	date_block_num	shop_id	item_id	item_price	item_cnt_day	item_name	item_category_id	item_category_name	shop_name
0	02.01.2013	0	59	22154	999.00	1.0	ЯВЛЕНИЕ 2012 (BD)	37	Кино - Blu-Ray	Ярославль ТЦ "Альтаир"
1	03.01.2013	0	25	2552	899.00	1.0	DEEP PURPLE The House Of Blue Light LP	58	Музыка - Винил	Москва ТРК "Атриум"
2	05.01.2013	0	25	2552	899.00	-1.0	DEEP PURPLE The House Of Blue Light LP	58	Музыка - Винил	Москва ТРК "Атриум"
3	06.01.2013	0	25	2554	1709.05	1.0	DEEP PURPLE Who Do You Think We Are LP	58	Музыка - Винил	Москва ТРК "Атриум"
4	15.01.2013	0	25	2555	1099.00	1.0	DEEP PURPLE 30 Very Best Of 2CD (Фирм.)	56	Музыка - CD фирменного производства	Москва ТРК "Атриум"

Figure 2 First Five Tuples of the Joined Dataframes

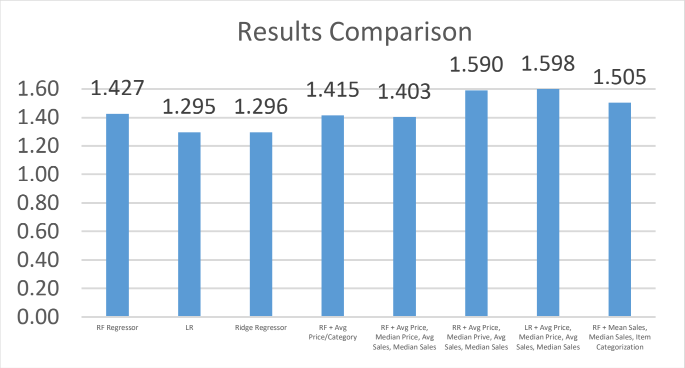


Figure 3 Comparison of First Attempt

III. DESCRIPTION OF MODELS USED IN EACH SUBMISSION

The results of the previous process were not really adequate. Hence, it was decided that some ready Kernels must be used in the preprocessing step, before proceeding with any Machine Learning Training.

A. Ready Kernel with XGBoost and Clipping

The first Kernel [6] that was used, had a lot of different features, and a lot of preprocessing stages. The owner of the Kernel, removed all the outliers, with respect to the price (more than 100.000) and the sales (more than 1001) of the products. Furthermore, with the use of shops, items, categories and many combinations between them, several extra features were generated. The information of all the attributes in the dataset were transposed from daily data to monthly data. This transpose was necessary, since the target value is a prediction of a month-based feature.

Next, the XGBoost algorithm was used as a machine learning tool for the prediction of next month's sales. The XGBoost is a "state-of-the-art" machine learning algorithm that is very popular these days [7]. The XGBoost can be used in both classification or regression problems, and it belongs to a family of boosting algorithms. The main reason that XGBoost is globally one of the most important algorithms is that it combines different "weak learners" to improve the general performance of the model and thus, the prediction accuracy. In this Kernel the XGBoost gave a score of 0.906.

The alteration that was conducted in the ready Kernel, was a new model, that used all the data for one shop at a time and the XGBoost algorithm was fitted in every shop. The XGBoost predicted the next month sales and from these predictions, only those needed in the test set were used. The score increased dramatically in 1.439, really close to the initial process of the previous chapter. (Section II.B).

However, another attempt of modifying the Kernel was considered important. It was detected that the owner of the Kernel split that data in the test set in cliques of 0-20. Clipping is a very useful technique in clustering problems [8]. Yet, it proved to be useful in this process as well. The clipping technique reduces the memory requirements and gives better accuracy results in less computational time. The initial clipping of 0-20 gave the 0.906 score as already mentioned before.

The goal was to alternate the limits of the clips in the test set, to improve the overall accuracy. The limits were changed several times from 0.01 to 22 but the accuracy became worth or at least remain the same. The results of this experimentation are presented in Figure 4.

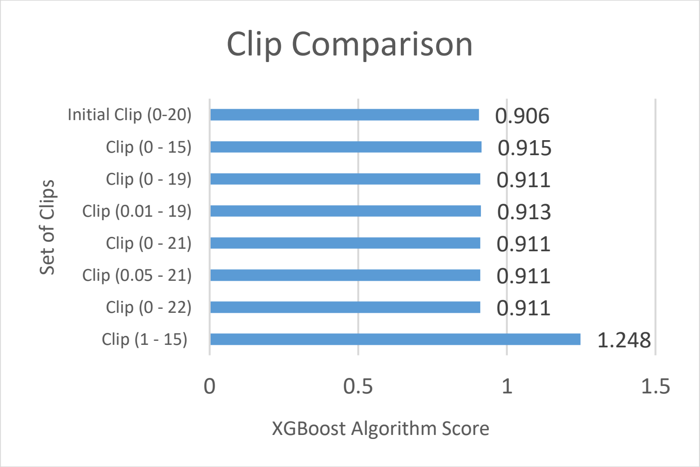


Figure 4 Clip Comparison with XGBoost Algorithm

B. Sine and Cosine of Months

Another thought to preprocess the data and add extra features in the ready Kernel, was the computation of the sine and cosine for every month. The sine and cosine gave a more spherical explanation of the circularity of the year's seasons. This means, that many data were better described on a season basis period rather than on a monthly basis approach.

These new features added an extra hidden information on the dataset and with the use of the already prepared (from the Kernel) features, the score reached the 0.935. The algorithm for this score was a combination of a RandomForest Regressor and a new introduced algorithm, the Bagging Regressor.

The Bootstrap Aggregative Regressor, also called as Bagging Regressor, is another ensemble meta-estimator, as RandomForest and LinearRegressor. The difference of Bagging Regressor is that it splits the random dataset in small subsets and fits the regressors in each one of them. It aggregates these predictions and forms the final prediction of the model, in less computational time [9].

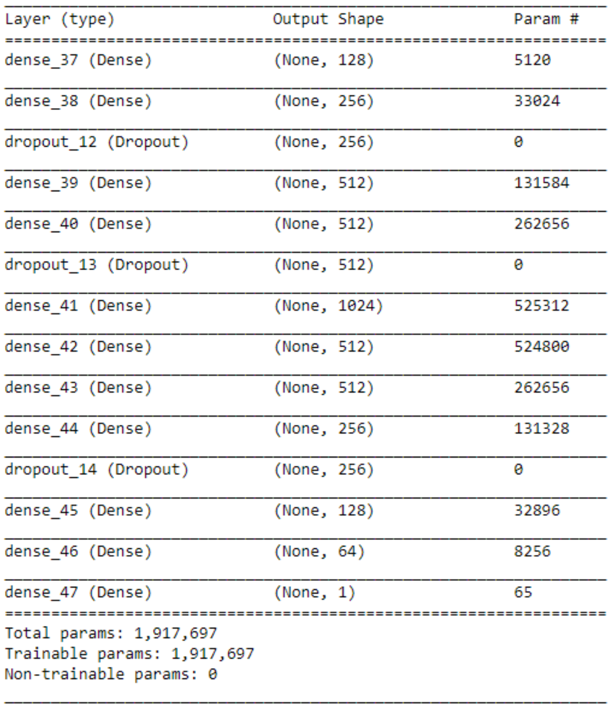
C. Keras Library with Simple MultiLayer Perceptron

In the second attempt of the same Kernel as before, the Keras Library was applied. The Keras Library is a high level Neural Networks API, that used in Python and it is running on top of TensorFlow [10]. It allows fast and easy prototyping and it can support both convolutional and recurrent networks. Therefore, is considered to be a useful tool for this particular dataset.

In the beginning, the preprocess step of the dataset remained exactly the same as in the XGBoost Algorithm procedure. This means that all the features were re-used, and the only difference is the addition of a Multilayer Perceptron (MLP) Network.

Initially, a simple MLP with only Dropouts and with no time delay or Convolutional 2D Networks was generated. The Dropout is a technique capable to minimize the risk of overfitting. In the Dropout technique in every training process a neuron is ignored and so the weights of each neuron are updated in every iteration [11]. Figure 5, displays all the simple MLP structure, consisted only of Dense and Dropout steps.

The result of this simple MLP network gave only a 1.372 score, far away from the XGBoost classifier. As a result, the addition of Convolutional Neural Networks was considered important.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_37 (Dense)             (None, 128)               5120
_________________________________________________________________
dense_38 (Dense)             (None, 256)               33024
_________________________________________________________________
dropout_12 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_39 (Dense)             (None, 512)               131584
_________________________________________________________________
dense_40 (Dense)             (None, 512)               262656
_________________________________________________________________
dropout_13 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_41 (Dense)             (None, 1024)              525312
_________________________________________________________________
dense_42 (Dense)             (None, 512)               524800
_________________________________________________________________
dense_43 (Dense)             (None, 512)               262656
_________________________________________________________________
dense_44 (Dense)             (None, 256)               131328
_________________________________________________________________
dropout_14 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_45 (Dense)             (None, 128)               32896
_________________________________________________________________
dense_46 (Dense)             (None, 64)                8256
_________________________________________________________________
dense_47 (Dense)             (None, 1)                 65
=================================================================
Total params: 1,917,697
Trainable params: 1,917,697
Non-trainable params: 0
_________________________________________________________________
```

Figure 5 Simple MLP Structure

D. Keras Library with Convolutional Neural Networks

The Convolutional Neural Networks (CNN) are mostly used in Image classification processes, or even in hand-written recognition [12], but nowadays are used in many applications as they cannot be challenged in terms of performance by shallow nets [11].

Apart from the addition of the CNN the Kernel was totally dropped out, and the preprocess step changed entirely. The data were reshaped into a 3D Dataframe with respect to month, items and shops. Nevertheless, the dataframe was very sparse (meaning a lot of missing values), which were filled with zeros afterwards. Moreover, a generator that returns 24-month windows was implemented.

The model finally fitted in a Time-Delay MLP with both Dropouts and Convolutional Neural Networks, and with the use of Adam optimizer the Mean Squared Error was minimized as Figure 6 represents. The score was decreased to 1.252 (compared to the simple MLP), but once more did not overcame the 0.906 from the ready Kernel.

The generator was reformed in 12-month and 5-month windows but the score stayed worse than the 24-month windows.

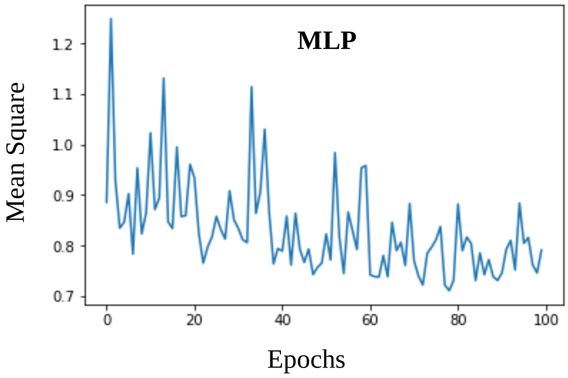


Figure 6 MLP Progress with CNN

E. Keras Library with CNN and Kernel

The last process with the Keras Library was to regenerate all the preprocess from the Kernel described previously, with the MLP Network from the last submission. It can be defined as a combination of the two aforementioned phases (C and D). The 1.259 score in Kaggle verified that there was no need for any further experimentation with the Keras Library.

F. RandomForest and LinearRegressor to new Kernels

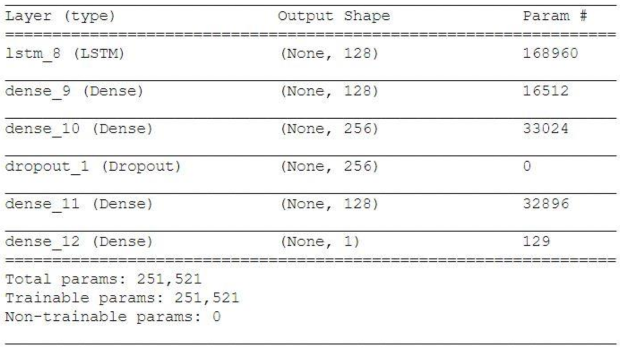
The next challenge was based on a combination of different Kernels. Both the pre-processing and the features extracted from these Kernels.

The data of one shop at a time were selected and fitted in both a RandomForest and a LinearRegressor models to predict the sales of the next month. The predictions of the sales were separated and only the ones need in the data set were retained. The problem at this point was treated as a simple (linear) time series decomposition problem. Thus, the trend was calculated and then subtracted from the data. The monthly index, was also calculated and the trend extrapolated to the next month, so the index could be added again. The final score was 1.281.

G. Long Sort Term Memory Model

The Long Sort Term Memory (LSTM) Model is part of the Recurrent Neural Networks (RNN). The RNNs are artificial neural networks, mainly for pattern recognition. As a Neural Network the RNN produces errors in the backpropagation phase. LSTMs models helps preserve those errors in any layer of the neural network [13].

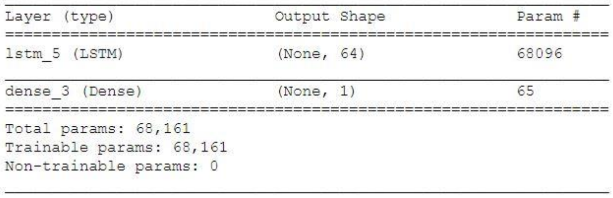
The Neural Network of this phase had only one LSTM without time dimension (window = 1). The features was taken from the first Kernel [6], and the final structure of the LSTM is explained in Figure 7.



```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_8 (LSTM)                (None, 128)               168960
_________________________________________________________________
dense_9 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_10 (Dense)             (None, 256)               33024
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_11 (Dense)             (None, 128)               32896
_________________________________________________________________
dense_12 (Dense)             (None, 1)                 129
=================================================================
Total params: 251,521
Trainable params: 251,521
Non-trainable params: 0
_________________________________________________________________
```

Figure 7 LSTM Structure

This LSTM gave a score of 0.927. It was later decided to simplify the neural network but the result of the new structure (Figure 8) differentiated only on the fourth (4th) decimal number. The computational time was not considerable larger between the two models, and since the first LSTM can work with more complex patterns, it was decided to maintain the model to its initial structure.



```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_5 (LSTM)                (None, 64)                68096
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 65
=================================================================
Total params: 68,161
Trainable params: 68,161
Non-trainable params: 0
_________________________________________________________________
```

Figure 8 Simplified LSTM

H. Majority Voting Model

The majority voting process uses a set of different algorithms, each one of them with its own prediction in the corresponding test instance. The model evaluates the individual predictions and extract as a final output the one that receives the half of the votes [14]. This results to a better accuracy in the final model, since the best combination of algorithms is applied to it.

For this paper, three different models were imported to a voting model. Two of them consisted of a Bagging Regressor with different features (one with sin/cos of the months and the ready Kernel, and the other with only the Kernel itself), and the third one was the LSTM of the previous process (chapter G). Every alteration on these three algorithms resulted a score between 0.92 and 0.95.

The voting model gave a score up to 1.036 which is really abnormal, since the process of Majority Voting must give a more accurate result than the algorithms it was applied to [15].

IV. Comparative Experiments and Results

Having tested several algorithms and different preprocess techniques, the selected model was the LSTM with no time dimension (section III.G). The features of these model were prepared from the owner of the Kernel used it that step.

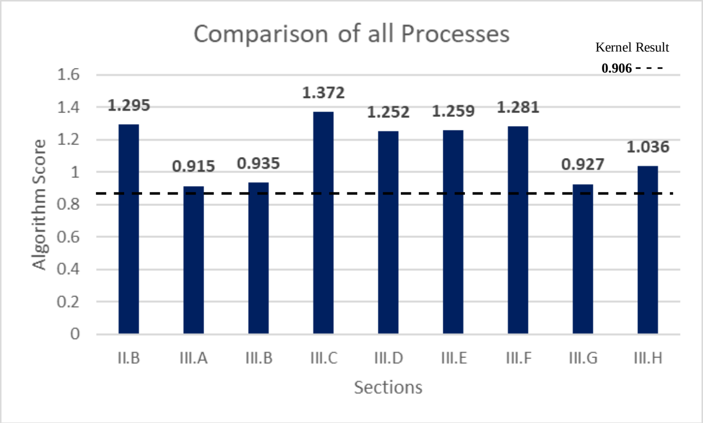


Figure 9 Comparison of Processes with Kernel Result

Figure 9 above represents the score of all the algorithms used in the previous section. It is observable that none of the scores exceeded the ready Kernel. The names of the processes are the names of the chapter that each process is described. This selection of names gives a better visualization of the chart, since the names of the processes could be large and not very visible. Moreover, the selected score is the best one from each occasion and not the mean of them. So every column of the chart has a lot of hidden scores beneath it.

As an example the II.B corresponds to the best score of the initial model with the RandomForest, the Linear Regressor and the Ridge Regressor (chapter II section B).

Although the best result was given by the XGBoost Regressor and the different clipping process of section III.A, it was decided that this process is not representative for this research since a big amount of the code was taken by the Kernel.

As a result, the LSTM Model proved to be the best algorithm since it was the one that had the most changes and alterations in the programming code, regarding the machine learning approach and the preprocess stage as well.

The improvement of the accuracy in each epoch of the LSTM in illustrated in Figure 10

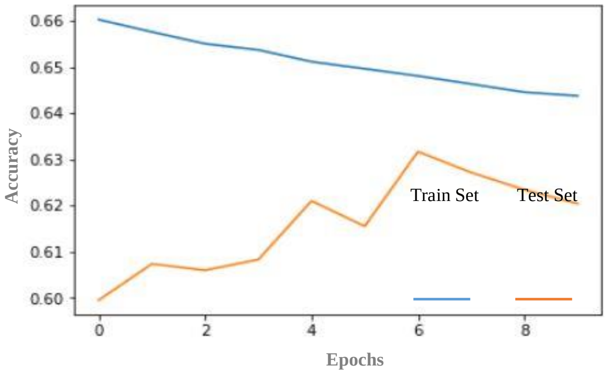


Figure 10 Accuracy Improvement of LSTM

The last Figure 11), explains a small effort that was made, regarding the total epochs of the LSTM. As the epochs increased beyond 15 the precision of the Neural Network was dropped. In particular, by setting the number of epochs to 20, the score became 0.982. Therefore, no further research for training the Neural Network was necessary.

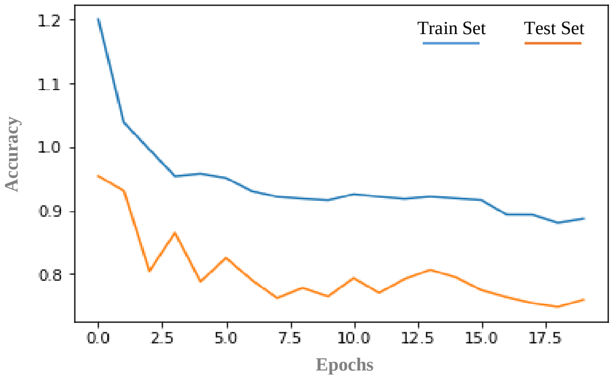


Figure 11 Testing the Number of Epochs

V. Conclusions

The given dataset and the prediction of the future sales were proved to be difficult to handle. The final 0.927 score was very convenient for the purposes of this machine learning challenge. The 0.906 of the ready Kernel had never been beaten from any of the described processes, though the knowledge that was extracted throughout the whole procedure was much more important.

By testing several algorithms and different preprocessing methods, the final outcome of this paper can be described as a research of different machine learning methodologies on regression problems, that are highly correlated with future sales predictions.

As a conclusion this paper was not only used as a tool for understanding an advanced machine learning problem, but it could also be a very comprehensive guide for anyone to utilize for other machine learning applications.

VI. References

[1] Kaggle, "Predict Future Sales," Kaggle, [Online]. Available: https://www.kaggle.com/c/competitive-data-science-predict-future-sales.

[2] S. M. Jasdeep , G. Prachi and S. Mr.Akhilesh , "A Comprehensive Approach Towards Data Preprocessing Techniques & Association Rules," India.

[3] S. Learn, "Ensemble Methods," [Online]. Available: https://scikit-learn.org/stable/modules/ensemble.html.

[4] Y. University, "Linear Regression Course," [Online]. Available: http://www.stat.yale.edu/Courses/1997-98/101/linreg.htm.

[5] N. S. Software, "Ridge Regression," NCSS Statistical Software, [Online]. Available: https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Ridge_Regression.pdf.

[6] D. Larionov, "Predict Future Sales," Kaggle, [Online]. Available: https://www.kaggle.com/dlarionov/feature-engineering-xgboost.

[7] M. Pathak, "Using XGBoost in Python," Data Camp, July 2018. [Online]. Available: https://www.datacamp.com/community/tutorials/xgboost-in-python.

[8] A. Bagnall and G. Janacek, "Clustering Time Series with Clipped Data," [Online]. Available: https://archive.uea.ac.uk/~ajb/Papers/BagnallMachineLearning2004.pdf.

[9] S. Learn, "Bagging Regressor," [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.BaggingRegressor.html.

[10] Keras, "Keras: The Python Deep Learning library," Keras, [Online]. Available: https://keras.io/.

[11] K. Diamantaras, "Deep Learning," in *Advanced Machine Learning*, International Hellenic University.

[12] M. D. Zeiler and R. Fergus, "Visualizing and Understanding," [Online]. Available: https://cs.nyu.edu/~fergus/papers/zeilerECCV2014.pdf.

[13] Skymind, "A Beginner's Guide to LSTMs and Recurrent Neural Networks," Skymind, [Online]. Available: https://skymind.ai/wiki/lstm.

[14] D. Necati, "Ensemble Methods: Elegant Techniques to Produce Improved Machine Learning Results," Developers, [Online]. Available: https://www.toptal.com/machine-learning/ensemble-methods-machine-learning.

[15] K. Diamantaras, "Ensemble Methods," in *Advanced Machine Learning*, International Hellenic University.