PROJECT 3 REPORT
DATA SCIENCE FOR BUSINESS

FALL 2016

Submitted to Professor Onur Guzey by Muhammad Ali and Taha Moiz

TABLE OF CONTENTS

Contents

Executive Summary

ROSSMAN PREDICTION

The Rossmann Stores data set deals with daily sales at its 1115 stores over a span of 30 months, this, of course, leads to over a million entries of data. The data is supplemented with store-wise information of types of stores, product assortment and applicable promotions, and day-wise information of holidays and whether the store was open or closed on any particular day. In this project we were asked to predict the sales for 6 weeks in advance for each of the Rossman's 1115 stores as accurately as possible using the data analytic tools and skills we developed during our Data science for business class. First and second project the statistical analysis to understand the data for project 3 in which we had to apply complex machine learning algorithm to make prediction.

GETTING STARTED

We obtained the massive datasets from the Kaggle competition website which contained the data for 1115 Rossman's store of 2 and a half years. The data was having missing values too which was taken into account for making the prediction. The large size of the data caused our files to get corrupt several times but after several efforts and trying new algorithms we were able to reach an accurate result. We used the statistical analysis done from the second project to deal with the missing values and process with the process of feature engineering. We created additional features to 33 in number and reduced the redundant features by statistically analyzing them or merging them to make a new single feature.

SOFTWARE AND THE MACHINE LEARNING ALGORITHUM

After having carefully analyzed the data and engineered the features we applied multiple machine learning algorithm to get the best result. The best result we got was from the deep learning algorithm. We performed this machine learning process on "RapidMiner" which is a software with many advance machine learning codes inbuilt in it.

PREDICITION AND RESULTS

We did multiple submission on Kaggle and the best score we reached was **0.12158** and with a rank of **1083**. Which means our model predicted 87.8% accurately the sales for the Rossman stores. After combining the model our new score became **0.11493** with rank of **135**. The procedure and the models developed will be discussed in detail in the following pages.

Statistical analysis

We used data visualization from the past two projects in order to draw simple conclusions from the data, we explored the data statistically to understand trends and influencing factors. At first glance it may seem that similar tools can be used to analyze the data sets, however, the variedly differentiating characteristics of the data sets suggest that differing approaches might be more beneficial.

For the Rossmann data the features provided where expanded to 18 as a part of feature engineering process with 1017209 data entries each. Thus making the final data too big to be handled by Excel or its extensions. Multiple linear regression was done with MATLAB and correlations were identified. A comprehensive statistical analysis was done for all the stores' data using ANOVA, Correlation Coefficients, Standardized Residual Analysis and similar other statistical tools. The features which are positively affecting the sales as well as those having negative impact on the sales were visualized. This analysis helped us understand that sales is not a constant and it varies over the period of time. The correlations of each of the 33 attributes with each other were calculated in a form of size 1089 matrix by using regression that helped to establish the relation among the attributes. In addition to the standardized residual test on the sales and on the multiple regression data revealed that residuals very over the number of observation. This concluded that over some period of time in a year in a periodic manner the residuals were very high or the prediction confidence interval dropped very low which may be due to the factors not taken into account while making adding features and which are somehow influencing the data. These statistical results helped us in finding missing data which will be discussed in later part of this report. Figure 1 shows one such result from the linear regression that helped us in defining the features and thus understanding the problem of over fitting and under fitting in a better way.

After having a through statistical analysis of the big data the trend and relation in the data helped to grab complete overview of the data that will help significantly in choosing the right machine learning algorithm for prediction of the sales.

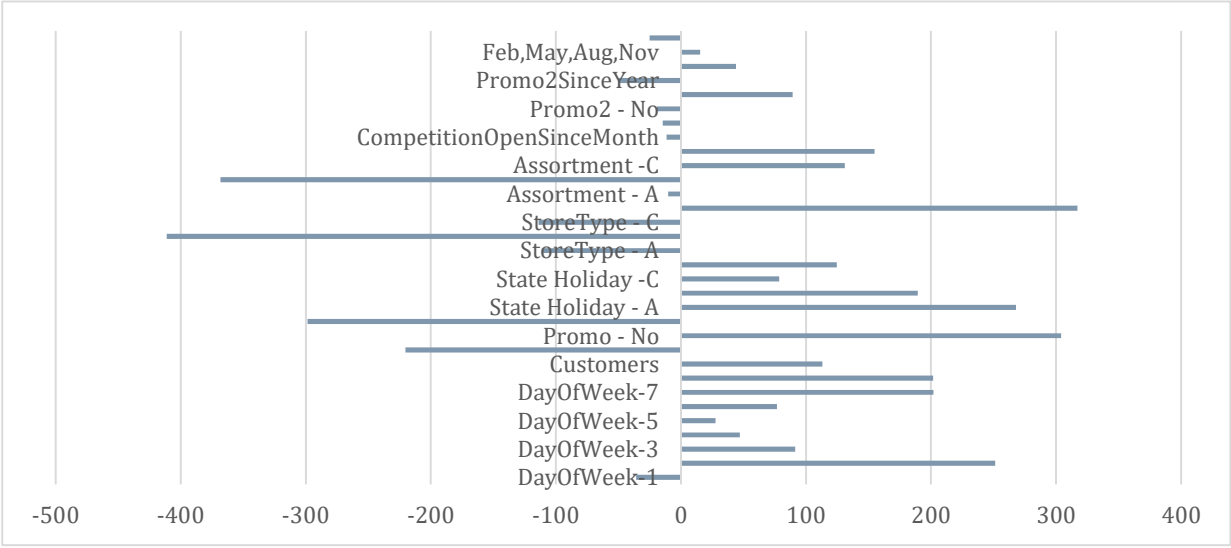


Figure 1 Linear regression- Weights assigned to the attributes

Features Engineering

INITIAL GIVEN FEATURES

There were initially 6 features in the train set and 9 features in the store data. The train set was having 1017209 fields representing the sales values for each of the 1115 stores in a span of two and a half years. The store data was having 1115 field which just showed the details of the particular store. To begin with feature engineering process we consolidated the train set data having 9 features and about a million field with store information data of 9 features. Thus the total features that was predefined from the data was 18 in number.

CREATION OF FEATURES, MERGING AND DATA CLEANING

In addition to the initial features given we elongated the number of features to 33 by breaking the classification features and making them separate. For instance the days of a week was having single column field (feature). We split the single column of 'days of a week' to 7 columns, each representing a day of the week, its row value is 1 if that is the day while 0 if that is not the day. We applied similar operation to the store type, school, holiday, state holiday, assortment type and on other classification type of predefined features.

To address the problem of under fitting we increased the number of features to 33, but it was also important to monitor the over fitting problem of our algorithm that's why after having careful statistical analysis we identified the redundant features by finding their correlations with sales and then removed them. We also merged to features to make them one like competition open since month and competition open since year were merged together to get the total time elapsed of the competition. Likewise, Promo 2 Since Week and Promo 2 Since Year were merged to form a single feature. Also note that the number of customer was only used for calculating the missing values since it was having a very strong correlation with some features however the number of customer's information was not provided in the test set therefore we excluded this feature form the train data. Figure 2 shows the feature generated file in excel.

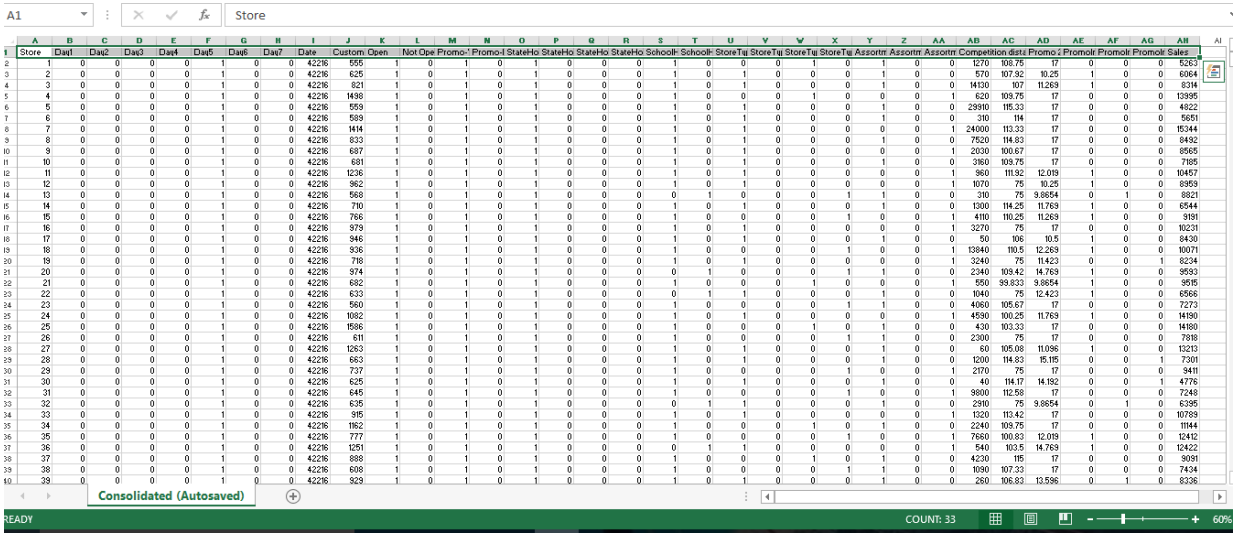


Figure 2Excel features creation

Missing data

UNDERSTANDING THE MISSING DATA

The 'Competition Distance', 'Competition Open Since Month' and 'Competition Open Since Year' for certain stores were missing from the original data set. There are many ways of filling the missing data set but depending on the size of the data and the number of missing value the write and the most efficient tool has to be used for filling the missing values. The Competition Distance missing values were 3234 from the 1017209 fields which means that the missing value was around 3.2% of the values given. Hence such a small percentage of the missing value is likely to have insignificant impact on the overall estimation. Thus we used average which was the most efficient way as the file size became too heavy to perform more advance process. In this case it was reasonable and wise to use average for calculating the competition distance missing values as their number was a small proportion of the overall.

It was also observed from the data that the values of 'Competition Open Since Month' and 'Competition Open Since Year' were missing with the same row indices. These two features were already combined to form a single feature thus reducing the number of missing values to be calculated. The combined feature was 'Competition Open Since' and from the statistical analysis it was found to have a very strong correlation (about 438) with the number of customer's field. The missing value for this field was about 35 percent of the number of rows which is significant to affect the overall accuracy of our prediction. Therefore special focus and in depth analysis was done of which approach to take. We used Linear Regression however it consumed lot of time and resulted in Microsoft Office not responding due to enormous volume of data but still we made it after long time.

Another set of data that was missing was of stores in the train files. About 172 stores data was missing from the file. We then developed the R code for imputation of the missing store values. Figure 3 the graph for imputing store id 100's data. The coding was done in such a way that if more than 50% of the stores on this day was open then make store 100 open. If promo was applied in more than 50% of the stores on that date, then make it too and etc.

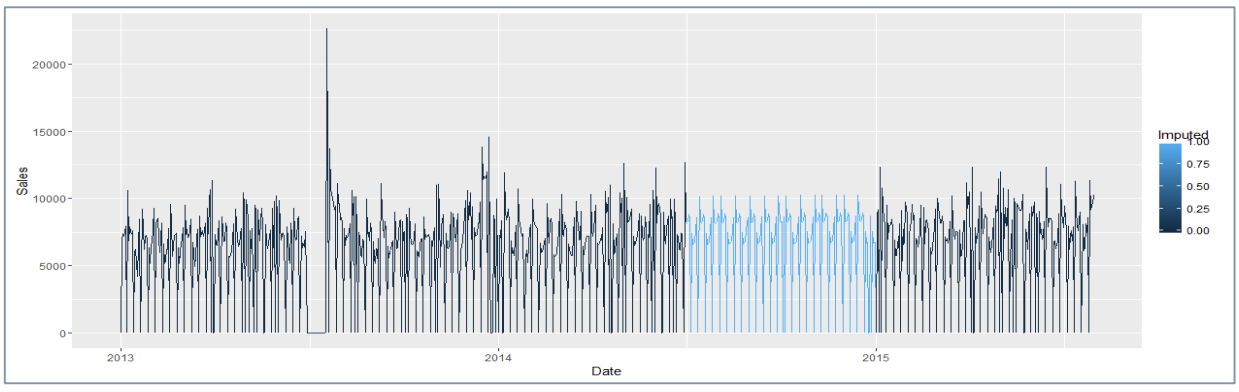


Figure 3 Imputed data for store id 100

RapidMiner

RapidMiner is an all-in-one Data Science and Machine Learning platform for Predictive Analysis. It can be used to prepare and explore data, as well as build models, validate the built models and finally to operationalize the results while running analytics on them. It even works with extensions for R and Python and the output results can be converted to a host of different formats.

However, its single most favorable feature is that it combines all these tasks with a simple to use graphical user interface. This means that the user does not need to deal with long lines to codes and languages. Just by dragging and dropping blocks and connecting them, a multitude of different operational processes can be created.

According to Billsus et (1998) one cannot be 100 percent sure on which machine learning code to use and the best way to select the right algorithm is by trial and error. Therefore we used all the machine learning models available on the RapidMiner software after feature engineering on the data set and preparing the data to be run on those machine learning model. Table 1 shows the results obtained after processing the data on each of the machine learning algorithm. The first iteration from the deep learning algorithm revealed the lowest error and therefore we proceeded with modifying our data after each run to achieve new score with lesser value of error. After several repetitive runs of the deep learning algorithm the most accurate value we were able to obtained was with score 0.12158. We made around 26 submissions on the Kaggle to finally reach this score. Since it was taking more than 6 hours to run the data on SVM model so we aborted the execution in the middle and therefore could not get its error.

We also measured the performance of the prediction on the data by splitting the data into 70/30 ratio and then applying model for performance measurement as can be seen from figure 5. Figure 4, 6 and 7 shows some of the models applied and how their blocking architecture was modelled. Figure 8 shows the results generated for the performance measurement on the deep learning algorithm. Figure 9 shows the root means square error obtained on testing from the cross validation data.

Machine learning model	Run time in minutes	Error
Deep Learning- First iteration	21	0.23985
Linear regression	30	0.24289
KNN	17	0.40777
Gradient Boost	16	0.31028
Neural Network	65	0.42958
SVM	6 Hours and 3 minutes	Aborted

Table 1 Different Machine learning models used

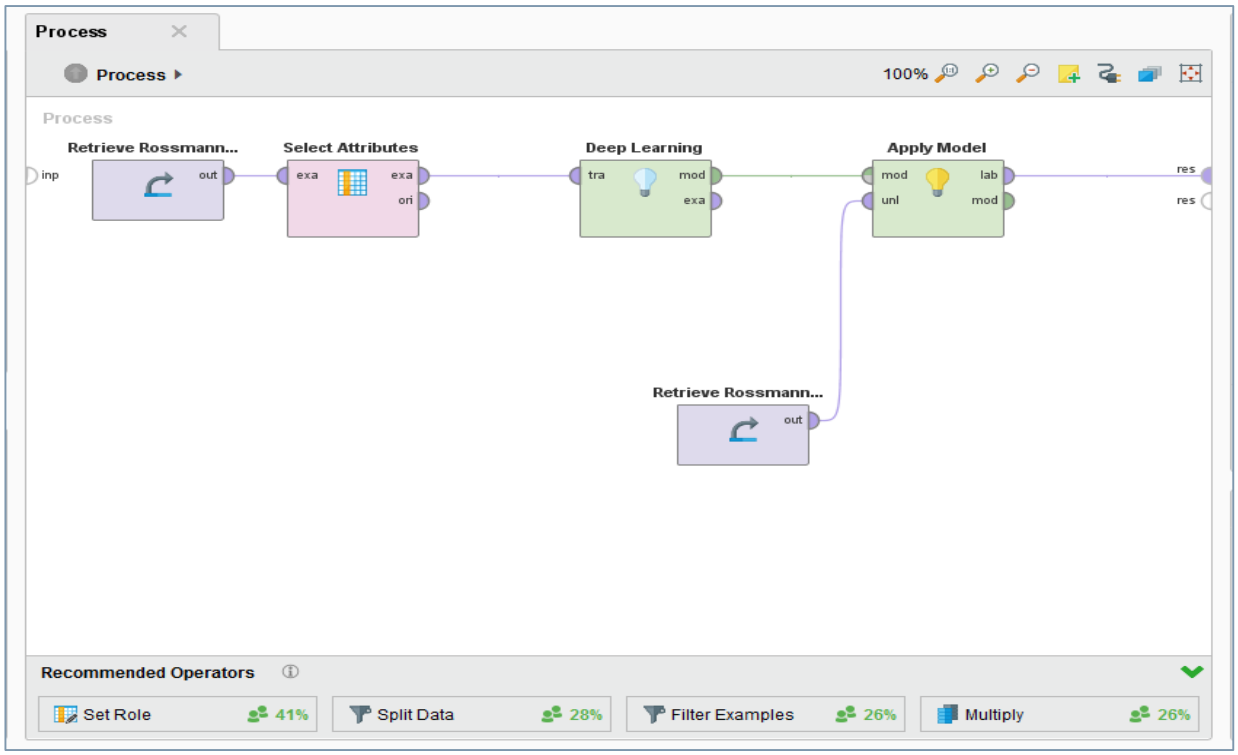


Figure 4 Deep learning algorithm on Rapid Miner

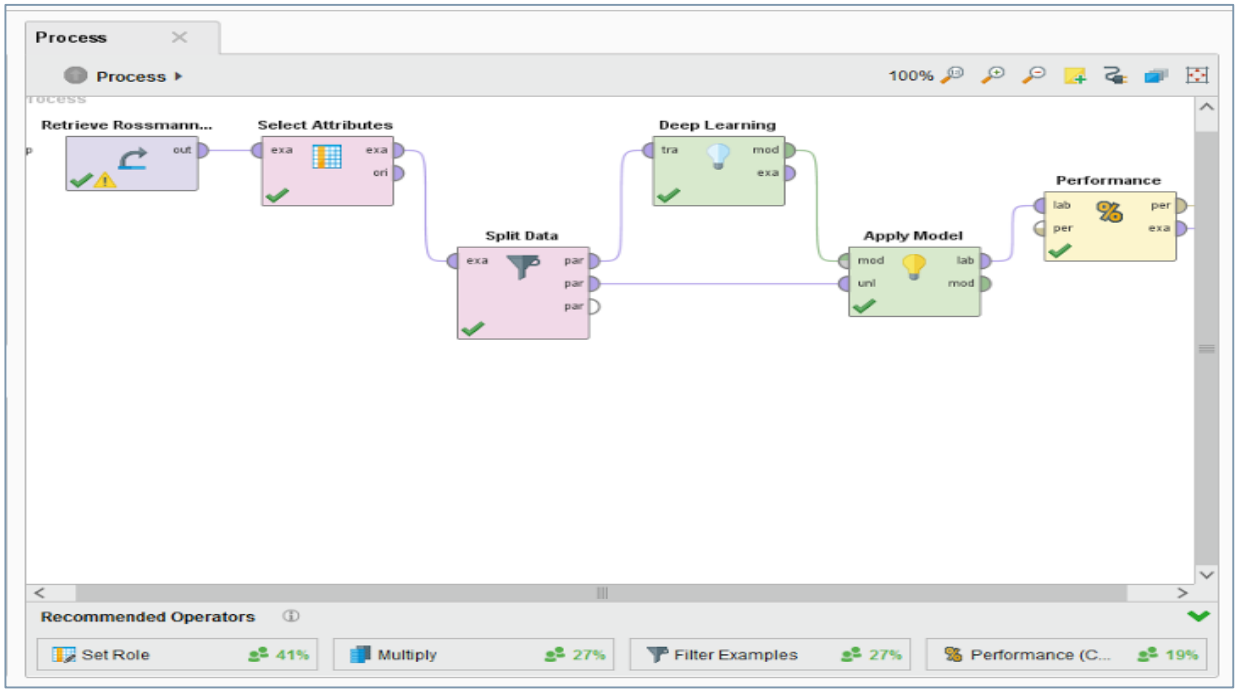


Figure 5 Deep learning algorithm on Rapid Miner with performance measurement

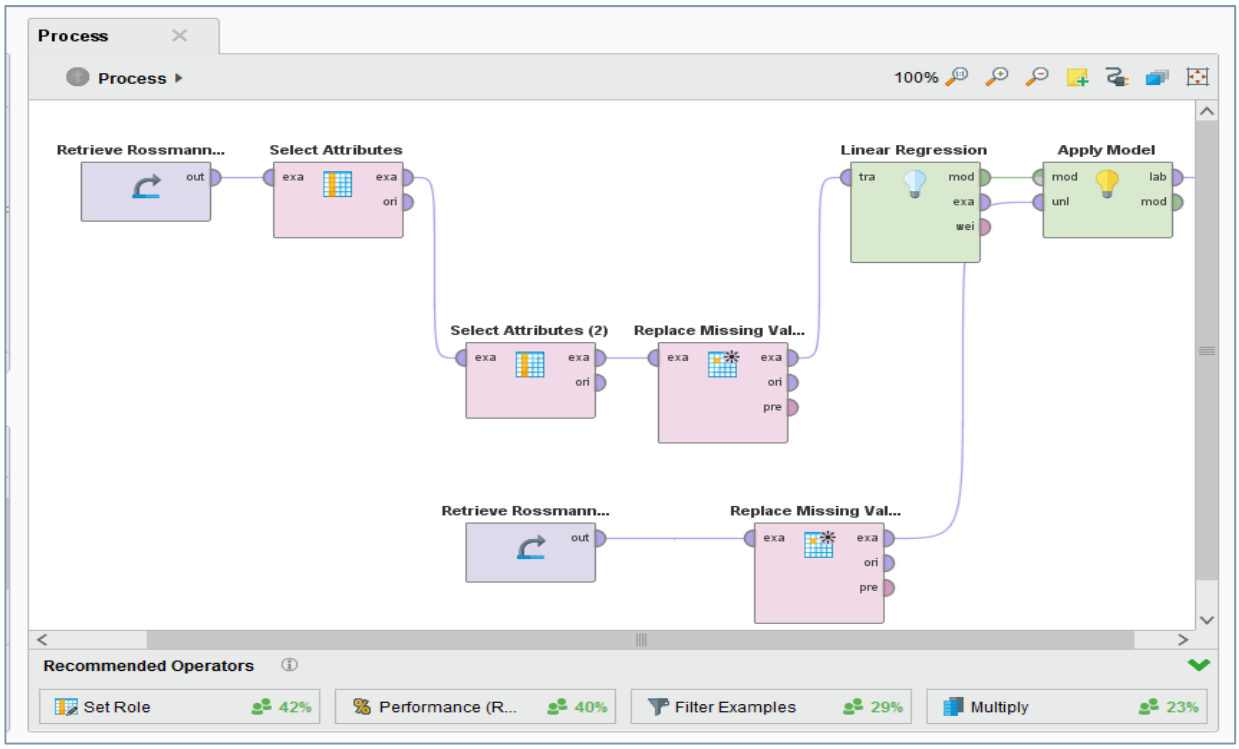


Figure 6 Linear regression algorithms blocks

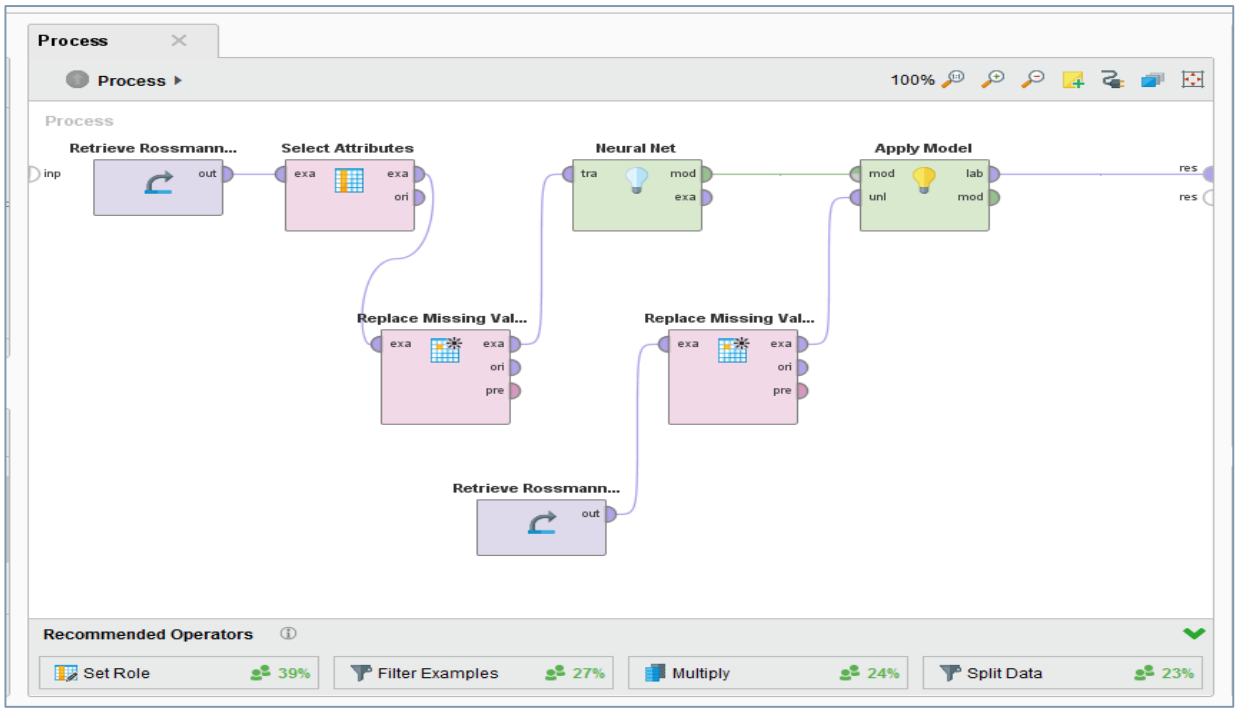


Figure 7 Artificial Neural Network blocks

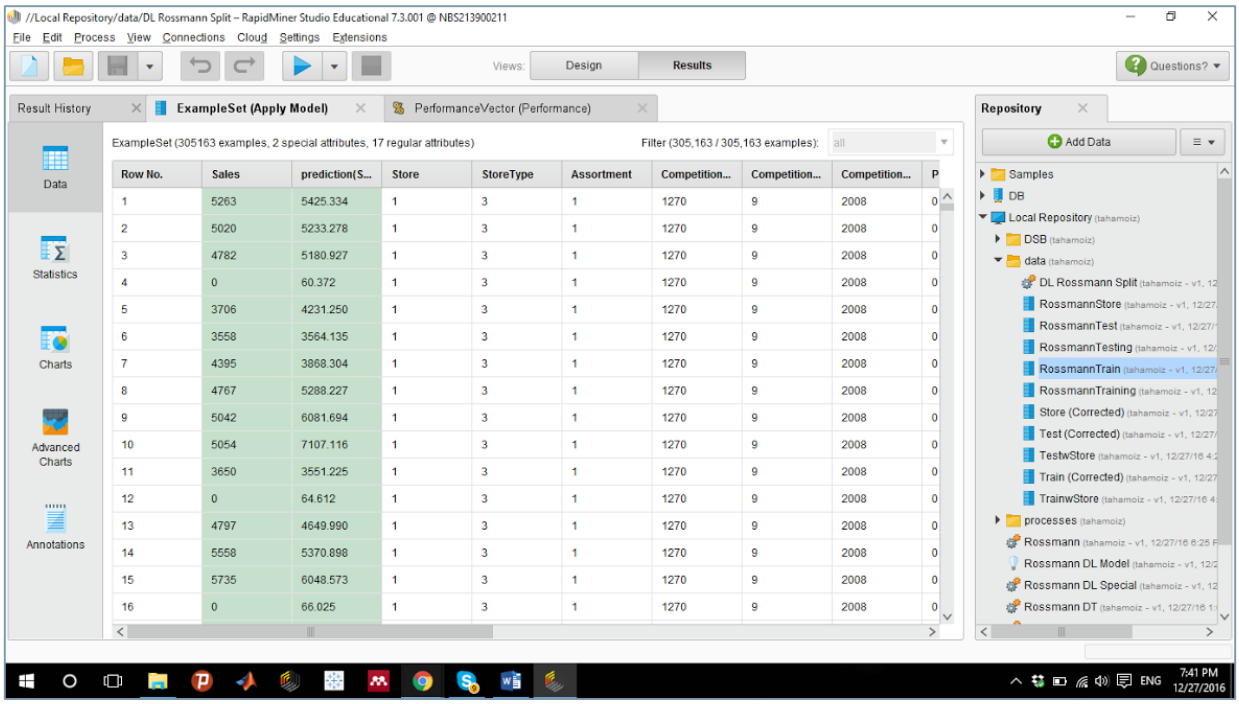


Figure 8 Performance vector generated on deep learning

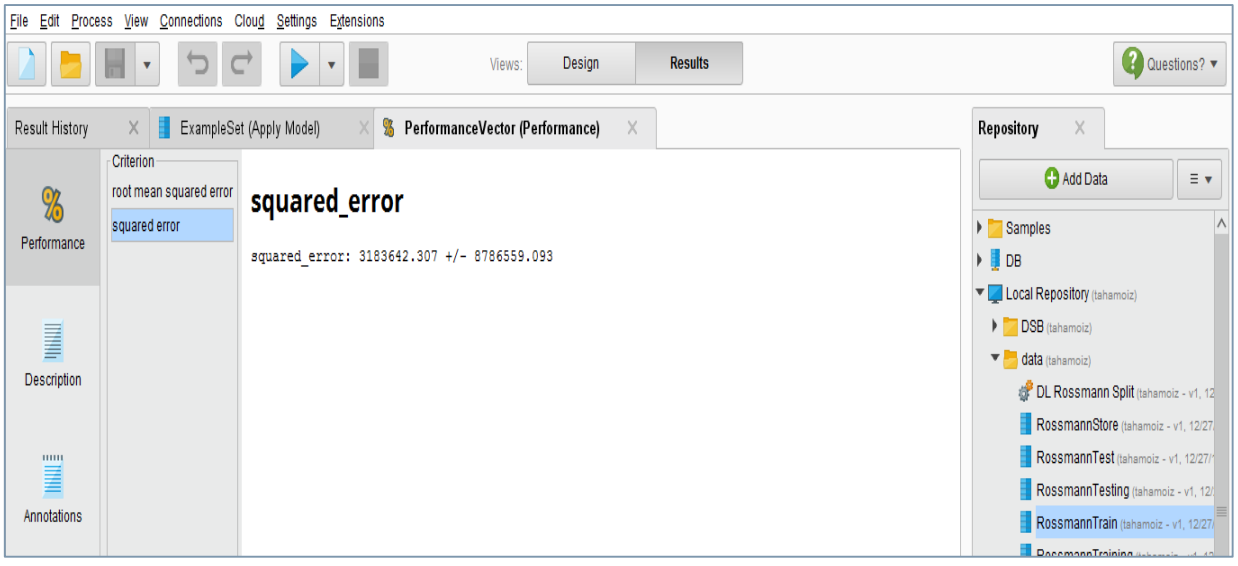


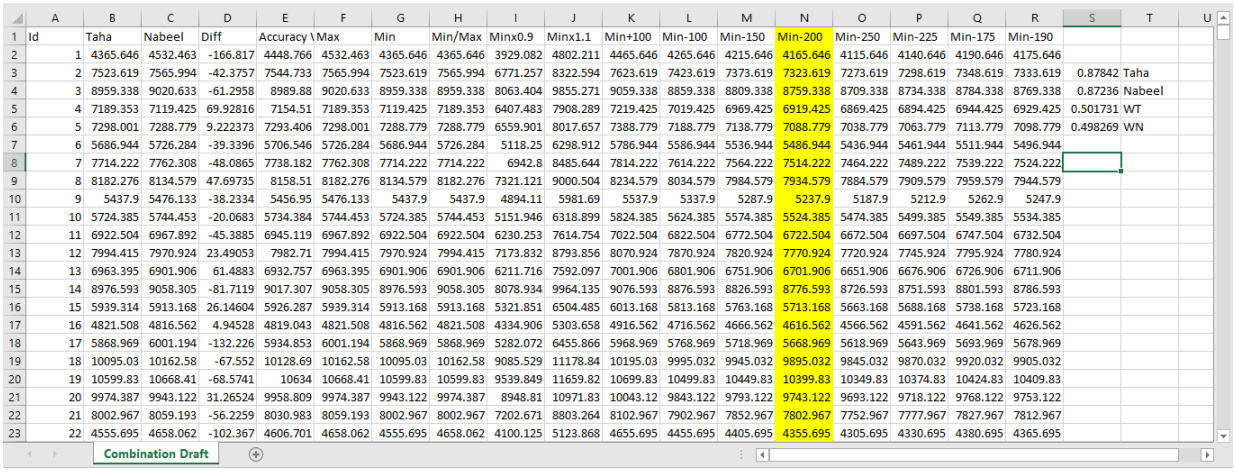
Figure 9 the squared root error for the deep learning model performance on the data

Combining the models

Before combining our deep learning algorithm submission results with any our score was **0.12158** and it ranked us **1083** among all the submissions as shown in figure 10. We combined the results of Nabeel-Sajid-Toqeer (NST) group, whose submission score was **0.12764** with a rank of **1521**, with ours and the combine result was amazing, our ranked went to **135** with score of **0.11493** as shown in figure 11. This means that by combining the model we achieved 1.5% more accuracy. All the scores in the leader board was so close that just 0.5% increase in accuracy will make score from 1083 to 234.

WHAT WE ACTUALLY DID

Considering the size of the data and the differing results each prediction model and tool supply, it was not immediately clear how combining another groups results will improve the accuracy. Another factor to consider was that since we did not know whether the results were underperforming or over performing overall, we could not directly infer which direction to take when combining results. Therefore, we tried different ways to at first gauge the direction to take, and then to refine our approach. We started with a simple average, and as expected it gave us a rank almost in between the two original ranks. Then we tried by taking the maximum and the minimum of the results for each entry. By taking the minimum, the result improved the rank to 1000, hence, we deduced that this is the direction to continue in. We tried other iterations by taking the minimum and maximum together for alternating fields, one by taking 90% of the minimum, one by taking 110% of the minimums, and one by adding 100 to all fields of the minimum, all reduced the accuracy. Finally, as we iterated by subtracting 100 from all fields of the minimum, the rank and score both increased significantly. With further iterations to get till subtracting 200, we reached the final accuracy and even with further iterations, it would not improve further. Figure 10 shows the number of trials made to reach the best value of the score.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Id	Taha	Nabeel	Diff	Accuracy \	Max	Min	Min/Max	Minx0.9	Minx1.1	Min+100	Min-100	Min-150	Min-200	Min-250	Min-225	Min-175	Min-190			
2	1	4365.646	4532.463	-166.817	4448.766	4532.463	4365.646	4365.646	3929.082	4802.211	4465.646	4265.646	4215.646	4165.646	4115.646	4140.646	4190.646	4175.646			
3	2	7523.619	7565.994	-42.3757	7544.733	7565.994	7523.619	7565.994	6771.257	8322.594	7623.619	7423.619	7373.619	7323.619	7273.619	7298.619	7348.619	7333.619	0.87842	Taha	
4	3	8959.338	9020.633	-61.2958	8989.88	9020.633	8959.338	8959.338	8063.404	9855.271	9059.338	8859.338	8809.338	8759.338	8709.338	8734.338	8784.338	8769.338	0.87236	Nabeel	
5	4	7189.353	7119.425	69.92816	7154.51	7189.353	7119.425	7189.353	6407.483	7908.289	7219.425	7019.425	6969.425	6919.425	6869.425	6894.425	6944.425	6929.425	0.501731	WT	
6	5	7298.001	7288.779	9.222373	7293.406	7298.001	7288.779	7288.779	6559.901	8017.657	7388.779	7188.779	7138.779	7088.779	7038.779	7063.779	7113.779	7098.779	0.498269	WN	
7	6	5686.944	5726.284	-39.3396	5706.546	5726.284	5686.944	5726.284	5118.25	6298.912	5786.944	5586.944	5536.944	5486.944	5436.944	5461.944	5511.944	5496.944			
8	7	7714.222	7762.308	-48.0865	7738.182	7762.308	7714.222	7714.222	6942.8	8485.644	7814.222	7614.222	7564.222	7514.222	7464.222	7489.222	7539.222	7524.222			
9	8	8182.276	8134.579	47.69735	8158.51	8182.276	8134.579	8182.276	7321.121	9000.504	8234.579	8034.579	7984.579	7934.579	7884.579	7909.579	7959.579	7944.579			
10	9	5437.9	5476.133	-38.2334	5456.95	5476.133	5437.9	5437.9	4894.11	5981.69	5537.9	5337.9	5287.9	5237.9	5187.9	5212.9	5262.9	5247.9			
11	10	5724.385	5744.453	-20.0683	5734.384	5744.453	5724.385	5744.453	5151.946	6318.899	5824.385	5624.385	5574.385	5524.385	5474.385	5499.385	5549.385	5534.385			
12	11	6922.504	6967.892	-45.3885	6945.119	6967.892	6922.504	6922.504	6230.253	7614.754	7022.504	6822.504	6772.504	6722.504	6672.504	6697.504	6747.504	6732.504			
13	12	7994.415	7970.924	23.49053	7982.71	7994.415	7970.924	7994.415	7173.832	8793.856	8070.924	7870.924	7820.924	7770.924	7720.924	7745.924	7795.924	7780.924			
14	13	6963.395	6901.906	61.4883	6932.757	6963.395	6901.906	6901.906	6211.716	7592.097	7001.906	6801.906	6751.906	6701.906	6651.906	6676.906	6726.906	6711.906			
15	14	8976.593	9058.305	-81.7119	9017.307	9058.305	8976.593	9058.305	8078.934	9964.135	9076.593	8876.593	8826.593	8776.593	8726.593	8751.593	8801.593	8786.593			
16	15	5939.314	5913.168	26.14604	5926.287	5939.314	5913.168	5913.168	5321.851	6504.485	6013.168	5813.168	5763.168	5713.168	5663.168	5688.168	5738.168	5723.168			
17	16	4821.508	4816.562	4.94528	4819.043	4821.508	4816.562	4821.508	4334.906	5303.658	4916.562	4716.562	4666.562	4616.562	4566.562	4591.562	4641.562	4626.562			
18	17	5868.969	6001.194	-132.226	5934.853	6001.194	5868.969	5868.969	5282.072	6455.866	5968.969	5768.969	5718.969	5668.969	5618.969	5643.969	5693.969	5678.969			
19	18	10095.03	10162.58	-67.552	10128.69	10162.58	10095.03	10162.58	9085.529	11178.84	10195.03	9995.032	9945.032	9895.032	9845.032	9870.032	9920.032	9905.032			
20	19	10599.83	10668.41	-68.5741	10634	10668.41	10599.83	10599.83	9539.849	11659.82	10699.83	10499.83	10449.83	10399.83	10349.83	10374.83	10424.83	10409.83			
21	20	9974.387	9943.122	31.26524	9958.809	9974.387	9943.122	9974.387	8948.81	10971.83	10043.12	9843.122	9793.122	9743.122	9693.122	9718.122	9768.122	9753.122			
22	21	8002.967	8059.193	-56.2259	8030.983	8059.193	8002.967	8002.967	7202.671	8803.264	8102.967	7902.967	7852.967	7802.967	7752.967	7777.967	7827.967	7812.967			
23	22	4555.695	4658.062	-102.367	4606.701	4658.062	4555.695	4658.062	4100.125	5123.868	4655.695	4455.695	4405.695	4355.695	4305.695	4330.695	4380.695	4365.695			

Combination Draft

Figure 10 Iterations done by trial and error method for combining the model

Submission on Kaggle

Figure 11 shows that before combining the model with Nabeel-Sajid-Toqeer Group our score was **0.12158** and rank was 1083 and figure 12 shows that after combining with them our model's score increased/decreased to **0.11493** with 135 position in rank.

1080	↓159	Sanjay Vaghela	0.12155	2	Thu, 05 Nov 2015 12:51:12 (-1.3h)
1081	↓107	Charles Roberson	0.12155	23	Mon, 14 Dec 2015 23:33:24 (-21.9h)
1082	↑133	kenvis	0.12155	4	Mon, 02 Nov 2015 13:27:10 (-3.9d)
1083	↓360	NighTurs	0.12157	38	Mon, 14 Dec 2015 23:53:36 (-9.3d)
-		**Taha Moiz**	**0.12158**	-	**Wed, 28 Dec 2016 20:09:07** Post-Deadline
Post-Deadline Entry If you would have submitted this entry during the competition, you would have been around here on the leaderboard.					
1084	↓138	The Smarties	0.12158	5	Sat, 07 Nov 2015 16:25:43 (-7d)
1085	↓237	NetworkMiner	0.12159	18	Mon, 14 Dec 2015 20:46:42 (-34.9d)
1086	↓318	bowen	0.12159	10	Wed, 04 Nov 2015 07:00:10
1087	↓333	udit saini	0.12159	71	Wed, 09 Dec 2015 11:22:06 (-64.3d)

Figure 11 Submission before merging

134	↑712	Avi Blinder	0.11488	69	Sun, 06 Dec 2015 05:09:39 (-24.6d)
135	↑10	U023	0.11488	60	Sat, 12 Dec 2015 06:36:28 (-3.1d)
-		**Taha Moiz**	**0.11493**	-	**Thu, 29 Dec 2016 08:33:25** Post-Deadline
Post-Deadline Entry If you would have submitted this entry during the competition, you would have been around here on the leaderboard.					
136	↑73	Dan Rasay	0.11495	19	Mon, 14 Dec 2015 08:21:47 (-0.2h)
137	↑242	lotus	0.11495	31	Mon, 14 Dec 2015 18:20:33 (-35.4d)

Figure 12 Submission after merging

Review and references

CONCLUSION

This project was very challenging and it enabled us to explore all the aspects of data science, from machine learning algorithms to the statistical evaluation of data in project 2. We developed a great deal of ideas and data analytics skill after this project and it has also sparked in us the interest to venture more into the depth of the vast field of data science.

REFERENCES

1. Billsus, Daniel, and Michael J. Pazzani. "Learning Collaborative Information Filters." Icml. Vol. 98. 1998.
2. https://www.kaggle.com/c/rossmann-store-sales