Malaysian Journal of Computing, 4 (2): 304-316, 2019

eISSN: 2600-8238 online

HANDLING HIGHLY IMBALANCED OUTPUT CLASS LABEL: A CASE STUDY ON FANTASY PREMIER LEAGUE (FPL) VIRTUAL PLAYER PRICE CHANGES PREDICTION USING MACHINE LEARNING

Muhammad Muhaimin Khamsan and Ruhaila Maskat

Faculty of Computer and Mathematical Sciences,
Universiti Teknologi MARA (UiTM) Shah Alam, Selangor, Malaysia
muhaiminkhamsan@gmail.com, ruhaila@tmsk.uitm.edu.my

ABSTRACT

In practice, a balanced target class is rare. However, an imbalanced target class can be handled by resampling the original dataset, either by oversampling/upsampling or undersampling/downsampling. A popular upsampling technique is Synthetic Minority Over-sampling Technique (SMOTE). This technique increases the minority class by generating synthetic class labels and assigned the class based on the K-Nearest Neighbour (K-NN). SMOTE upsampling can only upsample at most one minority class at a time, which means for a multiclass dataset, it needs to undergo multilayer SMOTE to balance the class label distribution. This paper aims to find a suitable method in handling imbalanced class using dataset from Fantasy Premier League (FPL) virtual player to predict price changes. The cleaned dataset has a highly imbalanced class distribution, where the frequency of "Price Remain Unchanged (PRU)" is higher than "Price Fall (PF)" and "Price Rise (PR)". This paper compared between the baseline (original) dataset, SMOTE-applied dataset and shuffled, linear and stratified sampling in split train-test subset, based on a deep learning algorithm. This paper also proposed criteria of low values in standard deviation (distribution of true positive on each class label on accuracy) as a measurement for finding the best method in handling imbalanced class labels. As a result, multilayer SMOTE until all the classes distribution is the same, combined with stratified sampling in split training and testing subset, get the lower standard deviation (5.7873), high accuracy (80.06%) and less execution runtime (1 minute 41 seconds) compared to the original highly imbalanced dataset.

Keywords*: Imbalanced class label, SMOTE upsampling, machine learning, price changes prediction.*

Received for review: 14-07-2019; Published: 22-11-2019

1. Introduction

In real life, it is quite difficult to get an equally class proportion, where the dataset usually has an imbalance output class label. This had been encounter by real-world situation such as predicting dropout high school students for early warning system (Chung & Lee, 2019), detection of financial frauds (Benabbou *et al.*, 2019), software defect prediction (Liu *et al.* 2019), predicting diabetes patients (Kaur & Kumari, 2018), bankruptcy prediction (Severin & Veganzones, 2018), text categorization (Ho *et al.*, 2014), customer classification of churn or non-churn classification (He *et al.*, 2011), and several others. When the training sets are in imbalance distribution, it could lead to biased machine learning (Glauner *et al.*, 2017). This kind of machine learning will be considered as bad and biased even though the accuracy value is high during validation by testing subset in the model creation phase. This statement supported by Stapor (2018) where accuracy and classification error may be appropriate only when the dataset is balanced, but in imbalance dataset cases which is a typical situation, other scores such as recall, precision, and F-Measure are more appropriate. Stapor (2018) statement is also supported by Barandela et al, (2002) stated that performance of a classifier in application with class imbalance not be expressed in terms of the average accuracy only, but measured by ROC curve and geometric mean as an indicator. Other than that, Kubat & Matwin (1997) proposed an alternative criterion of measure the percentage of positive examples and percentage of negative examples (for 2 class problem) correctly recognized, other than average classification accuracy on the dataset. For polynomial class (more than 2 output class labels), Rocca. (2018) stated that true positive for each class label means, that the predicted class is the same with the actual class, while false positive class is the predicted class that is mispredicted from the actual class. The combination of all true positive class on each output class label over the overall dataset is known as accuracy.

On the other hand, the accuracy value is being contributed by several class labels. By measure the amount of each positive class over the total positive class, we will get the amount of accuracy contributed by each different class. It is best practice to measure the values in terms of percentage rather than real amount because percentage form is the universal form where we can compare the measurement result even though the total of true positive is different among different sampling methods and classifiers. In this case, the measurement result used is the standard deviation, where the standard deviation is widely used in the statistic field, basically to measure the consistency or dispersion of the data towards its mean (Helmenstine, 2018). Small standard deviation value means that the dataset values are close to their average value and more consistent, while big standard deviation value indicate that the dataset values are spread out, and less consistency. Currently, a standard deviation calculator can be used using the webpage https://www.calculator.net/standard-deviation-calculator.html even without the user need to know the calculation behind it. In this research the standard deviation values will be used as a criterion to measure the distribution of the percentage of true positives in the accuracy, as a measurement comparison between original highly imbalance dataset and dataset that undergo SMOTE upsampling, other than high accuracy and less runtime execution measurement.

Regarding the research domain, which is the Fantasy Premier League (FPL) virtual player price changes prediction using machine learning, it utilized Artificial Intelligent (AI) and Machine Learning in predicting the price changes domain in Fantasy Premier League (FPL). FPL is a mobile apps for performance determinant strategy games domain in soccer matches, where in these games with a limitation of $100 million virtual funding resources, the participant of FPL can choose and construct a 15-man team (known as virtual player) consist of a real footballer in English Premier League (EPL) into their fantasy team. However, with limited virtual funding resources, FPL participant needs to strategizing on the combination of cheap and expensive virtual players. In each gameweek of the EPL season, the virtual player will be given FPL points according to his performance such as goals, assist, cleansheet and many more that are already predefined in this apps. At the end of the season, the participant who gets the highest FPL points will be declared as the winner and might get prizes from the fantasy league and community that he/she joined.

In these games, at the beginning of each season, the virtual player will be given an initial price value assigned by experts. Experts here is a group of sports journalists responsible to evaluate the performance of an athlete and assign it in a form of price values. The better a player's past performance, the more expensive his initial price value. A virtual player's price value can fluctuate from time to time, it can be price rise or price fall due to his performance and transfer activity of virtual player among FPL participants. Currently, there is no specific performance and transaction rule to predict which virtual player will have price changes and participants find it difficult to predict which virtual player will have a price change over time. If these price changes can be predicted beforehand, then FPL participant can do necessary transactions such as acquiring price rise-to-be and selling price fall-to-be virtual player, before the real price changes happen. This action will help participants constructing a fantasy team consisting of many premium expensive players hence increasing the possibilities of gaining huge FPL points in the future. The research domain cleaned dataset has a highly imbalanced class label distribution, where the class label "Price Remain Unchanged (PRU)" have 16,743, "Price Fall (PF)" have 1,730 and "Price Rise (PR)" have 553 rows of the dataset.

There are several papers addressing this imbalance class label issues and how to handle it. Alejo *et al.*, (2007) stated that imbalance output class label issues can be managed by resampling the original dataset, either oversampling the minority class or undersampling the majority class. Both strategies have shown important significance, where oversampling increases the size of the dataset artificially and burden the computational learning algorithm, while undersampling may throw out potentially useful data. Currently oversampling is also known as upsampling while undersampling is also known as downsampling. The simplest method of the oversampling method is random sampling, where it selects minority instances and duplicates it to have the same distribution as the majority class label. However, this duplicate approach will lead to over-fitting.

In order to overcome this over-fitting issue, Bowyer *et al.*, (2002) proposed a technique named "Synthetic Minority Over-sampling Technique (SMOTE)", where this technique basically upsampled the minority class label by creating a synthetic class label and assigned the class based on the K-Nearest Neighbour (K-NN) class label, where the k value specified by user. Currently, in Rapidminer tools, the SMOTE upsampling technique can be utilized by installing the SMOTE oversampling extension. Basically, a single SMOTE upsampling can only upsample one class label only. To upsample several class until all classes have fair distribution, user need to do an iterative SMOTE upsampling and the number of iterative SMOTE upsampling can be known using formula, Iterative SMOTE upsampling = (amount of different output class label) – 1, which means that if the dataset has 3 kinds of output class label, it needs to undergo two times of SMOTE upsampling, or if the dataset has 5 kinds of output class it needs to undergo four times of SMOTE upsampling and so on, in order to have a fair output class label distribution.

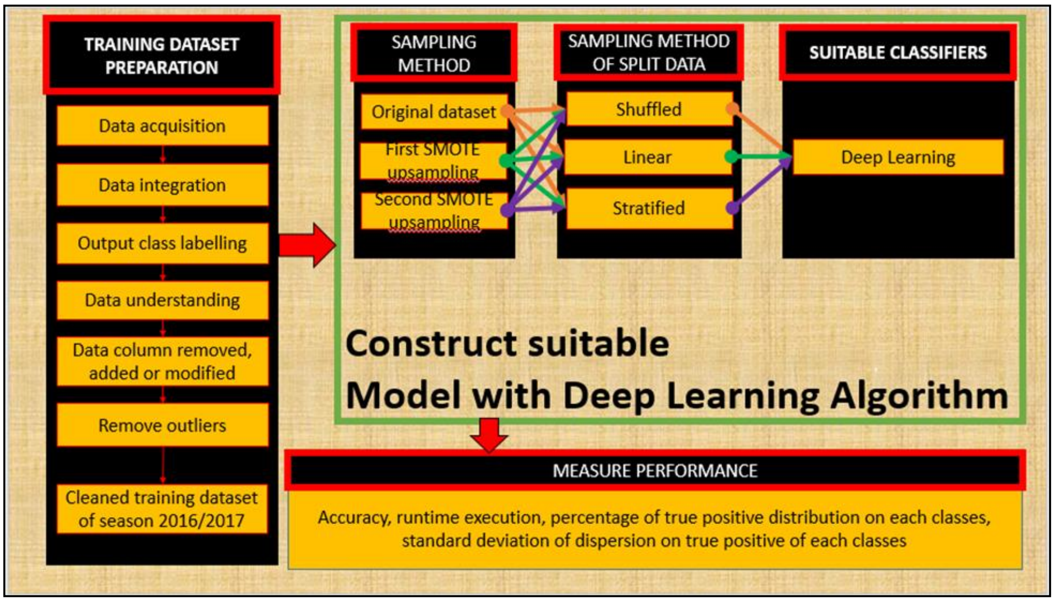

Figure 1. Overall research design workflow

2. Methodology

This section provides a brief methodology or research design of the experiment from the beginning. Figure 1 shows the infographic workflow of the overall research design that consist of two parts which is preparing a cleaned training dataset of Fantasy Premier League

(FPL) virtual player season 2016/2017 followed by constructing a suitable FPL virtual player price changes predictive modelling by comparing the performance of 9 different sampling methods, in order to find the best method in handling highly imbalanced dataset. The detail of each part and subpart will be explained in detail later.

2.1 Preparing a Cleaned Season 2016/2017 Training Dataset

For the part of preparing a cleaned training dataset of season 2016/2017, it started by data acquisition, data integration, output class labelling, data column removed, added and modified, and also remove outliers. For data acquisition, the dataset is collected at Github retrieved from https://github.com/vaastav/Fantasy-Premier-League/tree/master/data that are already scrapped by a person named Vastaav Anand (2018). As the datasets in the Github are segregated by gameweek, hence all the dataset will be integrating into a single comma-separated values (CSV) file using a virtual players' name as id and sort it by name and gameweek file. Next, after the dataset is segregated into a single file, an output class labeling will be conducted by comparing a player's price values from one gameweek to the next gameweek. The output class labeling will be assigned using python based on simple rule statement which is:

```
IF (gameweek price values > next gameweek price values):
      Print ("PRICE FALL")
ELSE IF (gameweek price values = next gameweek price values):
      Print ("PRICE REMAIN UNCHANGED")
ELSE IF (gameweek price values < next gameweek price values):
      Print ("PRICE RISE")
```

After the dataset had been undergoing output class labeling, there will be a data understanding phase, where in this phase, the dataset will be examined especially on the output class labeling distribution. The initial dataset after undergoing an output class labeling process, had about 90% distribution on price remain unchanged while price rise and price fall are very little. In this phase also, we can see some attributes can be combined to make a single attribute especially when it comes to virtual players information attributes such as virtual players' names, team, gameweek, round and the name of the opponent on that particular gameweek. All these information attributes will be combined to form a new attribute classified as an ID, as this information attributes do not really give any informative information during the machine learning model creation, and also to reduce the dimension of the dataset. This process is known as the data column removed, added or modified process.

Other than that, there is some error in the output class labeling process for the last gameweek on each virtual player as there is no next gameweek to be compared to. Hence the last gameweek of each virtual player will be removed as it is considered as an outlier. In this English premier league, the teams are not only playing in the league only, but they also participate in the international and domestic tournament such as UEFA Champions League, UEFA Europa League, FA Cup and also Carabao Cup. Due to this, the gameweek schedule that should play on a certain gameweek had to be postponed and need to be rescheduled. Thus, there will be blank gameweek and double gameweek for the certain teams in the league. In this blank and double gameweek, a certain virtual player dataset could be none or doubled in a certain gameweek and need to be removed from the research scope as it is considered as an outlier. Based on some reading and research on related articles and websites, it is confirming that in FPL virtual player dataset season 2016/2017, gameweek 17,26,28,34,36,37 considered as a blank and double gameweek, thus removed from this research scope.

After the dataset undergoes all the above process, a cleaned FPL virtual player season 2016/2017 dataset had been constructed, with a distribution of 16,743 price remain unchanged, 1,730 price fall and 553 price rises. The cleaned FPL virtual player dataset is ready to be used in the second part of this research, which is to construct a suitable FPL virtual player price changes prediction by comparing the prediction result between original highly imbalance dataset and dataset that undergo SMOTE upsampling, and also comparing sampling type of shuffled sampling, linear sampling and stratified sampling in split the dataset into train-test subsets.

2.2 Constructing a Suitable FPL Virtual Player Price Changes Predictive Modeling by Comparing Original Dataset with SMOTE Upsampling dataset

In this part of constructing a suitable FPL virtual player price changes predictive modeling, it is used to conduct machine learning experiments and compare between original highly imbalance output class label dataset, first SMOTE upsampling and second SMOTE upsampling. Explanation of these three different samplings are:

1. The original dataset is the cleaned season 2016/2017 dataset that is not undergoing any SMOTE upsampling and maintains the original output class label distribution.
2. The first SMOTE upsampling dataset is the original dataset that had been undergoing SMOTE upsampling once. It is to increase the amount of most minority output class label, using k-nearest neighbours (K-NN) concept in creating a new synthetic dataset for that most minority output class label. In this context, the price rise label which is the most minority output class label will be upsampled into having a distribution same as the price remain unchanged, which is the most majority output class label distribution. Distribution of other output class label is maintaining in this process.
3. Second upsampling dataset is quite the same with the first upsampling, but the upsampling process had been done twice in a row, where after the first SMOTE upsampling had been done, the most minority output class label on that time will be upsampled into having the same distribution as the most majority output class label. In this context, the price fall which is the most minority output class label after the first SMOTE upsampling, will be upsampled into having a distribution same as the price remain unchanged and prise rise class, which is the most majority output class label distribution. Distribution of other output class label is maintaining in this process

The distribution of output class labels of the original dataset, first SMOTE upsampling and second SMOTE upsampling will be shown in Figure 2.

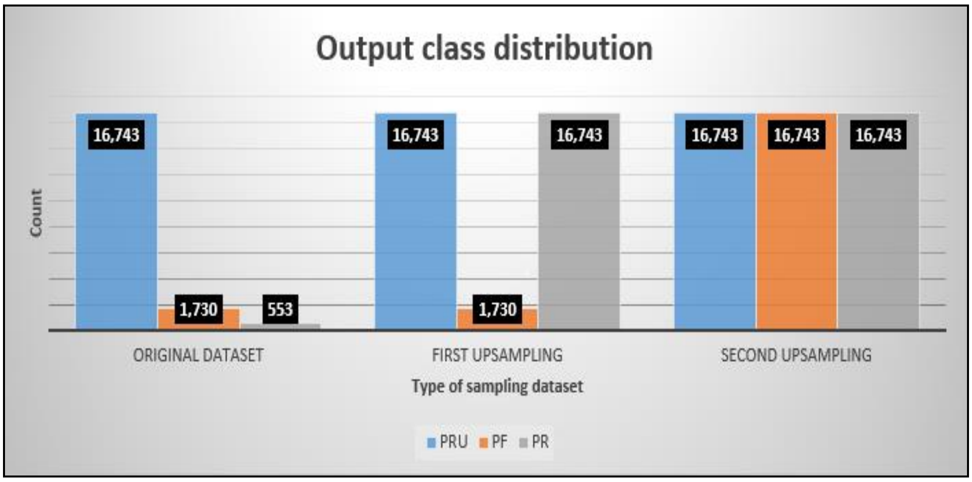

Figure 2. Distribution of original dataset, first SMOTE upsampling and second SMOTE upsampling

2.3 Constructing a Suitable FPL Virtual Player Price Changes Predictive Modeling by Comparing Shuffled Sampling, Linear Sampling and Stratified Sampling in Train-Test Subsets.

The 3 different sampling methods above (original dataset, first SMOTE upsampling and second SMOTE upsampling) will be combined with 3 different sampling methods in split data (into training and testing subset), which is shuffled sampling, linear sampling, and stratified sampling. Explanation on all 3 different sampling methods in the split data process is as below statement.

1. Shuffled sampling is a sampling method that builds random subsets of the dataset. Datasets are chosen randomly for making subsets.
2. Linear sampling is a sampling method that simply divided the dataset into partitions by a consecutive sequence row without changing the order of the dataset.
3. Stratified sampling is a sampling method that builds random subsets and ensures that the class distribution in the subsets is almost the same as in the original dataset.

Figure 3 to 5 show an example of a dataset divided into training and testing subset at ratio 0.75 training and 0.25 testing subsets, on all shuffled, linear and stratified sampling.

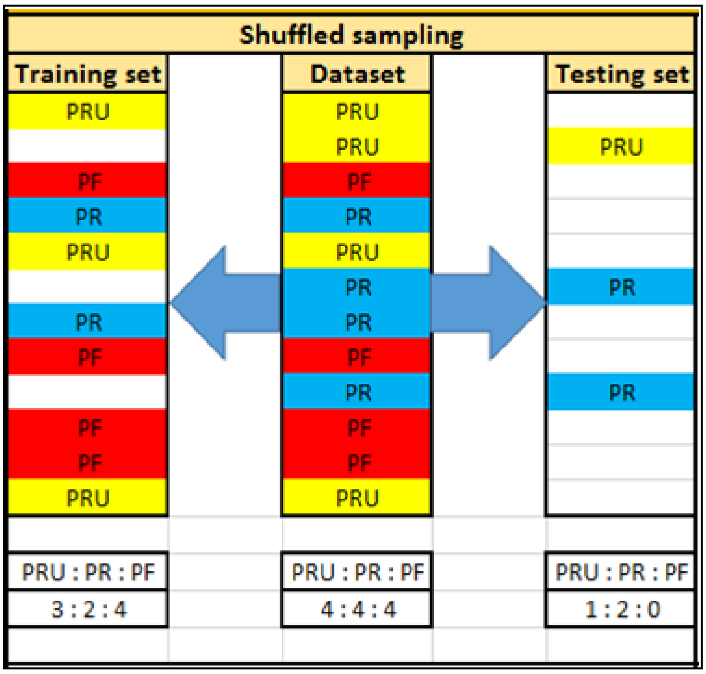

Figure 3. Example of shuffled sampling in split data

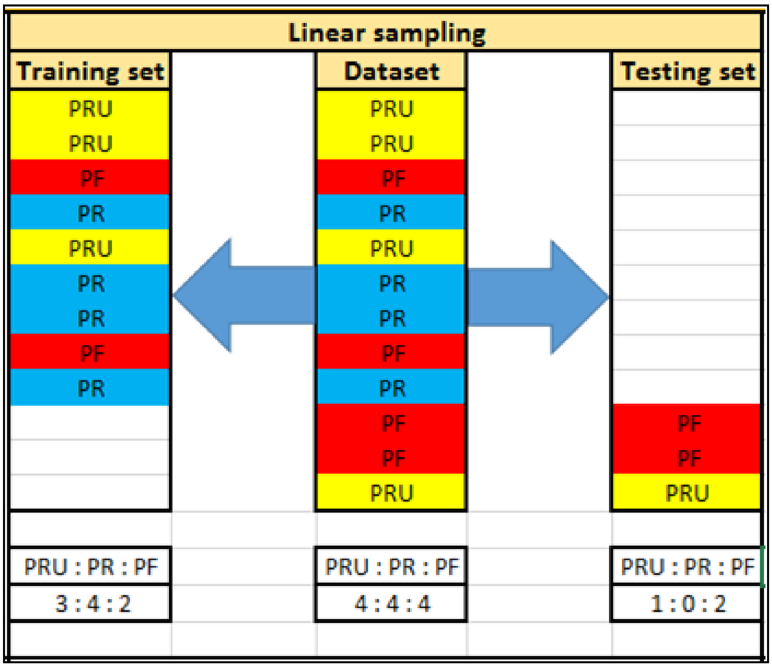

Figure 4. Example of linear sampling in split data

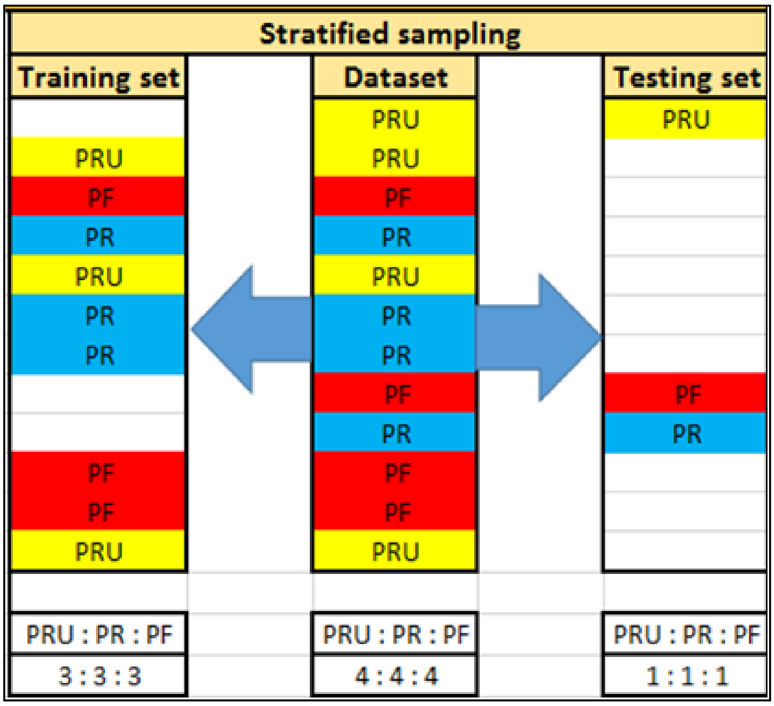

Figure 5. Example of stratified sampling in split data

The combination of 3x3 sampling method above will lead to a total of 9 different sampling method, which is:

1. Original dataset + Shuffled sampling in split data
2. Original dataset + Linear sampling in split data
3. Original dataset + Stratified sampling in split data.
4. First SMOTE upsampling dataset + Shuffled sampling in split data
5. First SMOTE upsampling dataset + Linear sampling in split data
6. First SMOTE upsampling dataset + Stratified sampling in split data.

7. Second SMOTE upsampling dataset + Shuffled sampling in split data
8. Second SMOTE upsampling dataset + Linear sampling in split data
9. Second SMOTE upsampling dataset + Stratified sampling in split data.

Then all the above 9 different sampling methods will undergo Deep Learning algorithm in Rapidminer auto model. Run and compare all 9 different predictive modelling method results and find the best suitable method in handling highly imbalanced output class label, in terms of high accuracy, less runtime execution and less standard deviation values (distribution of true positive on each class label on accuracy). All 9 different methods experiments conducted on the default setting of Rapidminer auto model tools, with the local random seed (LRS) = 1992, the partition of 0.6 training and 0.4 testing subsets.

2.4 Step-by-step to Reproduce the Comparison of 9 Sampling Methods Experiments in Rapidminer

1. Download Rapidminer educational version. It is important to download educational version instead of public version, because this experiment require utilization of rapidminer auto model option, and only Rapidminer educational provides this auto model option.
2. Retrieved the cleaned training 2016/2017 dataset. The cleaned training 2016/2017 dataset can be retrieved at google drive (https://drive.google.com/drive/folders/1BSu6dFuquAbeXG7vIBMNtya8kyoSITN5) in compressed file Muhaimin FYP dataset.raw and extract the excel file named 'TRAINING DATASET 2016/2017 and saved the excel file.
3. Open the Rapidminer educational version, then download the SMOTE upsampling extension at the Rapidminer marketplace in the extension toolbar
4. Run the automodel option, with the cleaned training dataset, and choose only deep learning algorithm only, with all default settings.
5. After the result of the deep learning predictive modelling has shown, click the show process, and it will show the whole process of the deep learning predictive modelling as shown in the Figure 6.

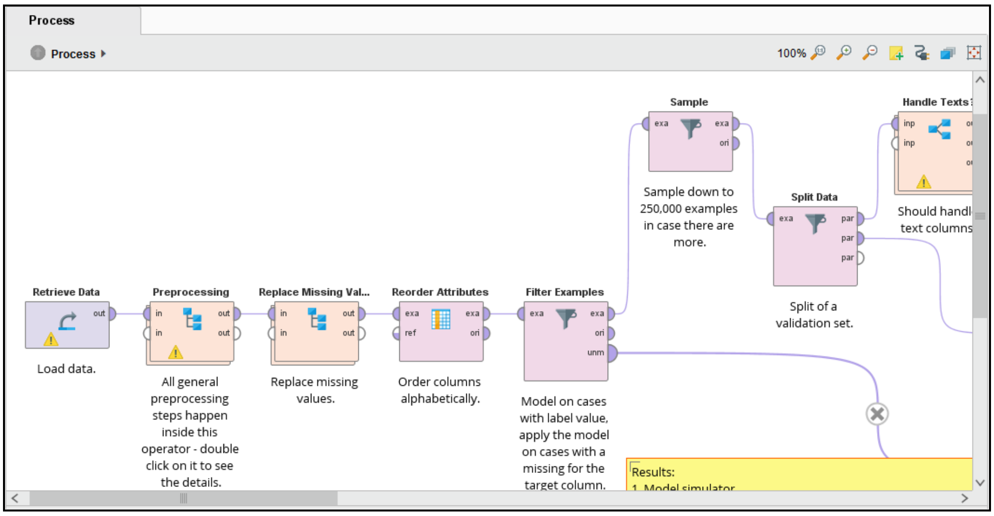

Figure 6. Rapidminer Automodel Process Flow

1. Remove the box named 'Sample' (in Figure 6), as this box function to sample down (also known as downsampling). These experiments do not use any downsampling method at all.

2. In the box named ‘Split Data’ (in Figure 6), choose the linear sampling parameter in the sampling type. By default, the local random seed is set at 1992 and the partition is 0.6 training and 0.4 testing subset.
3. Run the whole process and observe the confusion matrix result
4. Convert the confusion matrix result into true positive percentage form by using formula: percentage of true positive of each class = true positive each class / total accuracy.
   Do the same step for each true positive class in the experiment result, then saved the results in excel file.
5. Rerun the experiment step 7-9, by changing the sampling type in the parameter into shuffled sampling and stratified sampling
6. Implement first SMOTE upsampling. First SMOTE upsampling can be done by search SMOTE upsampling in the operator and grab into the process flow. Link the ‘SMOTE upsampling’ box between the box named ‘Reorder Attributes’ and ‘Filter Example’ as shown in the Figure 7.

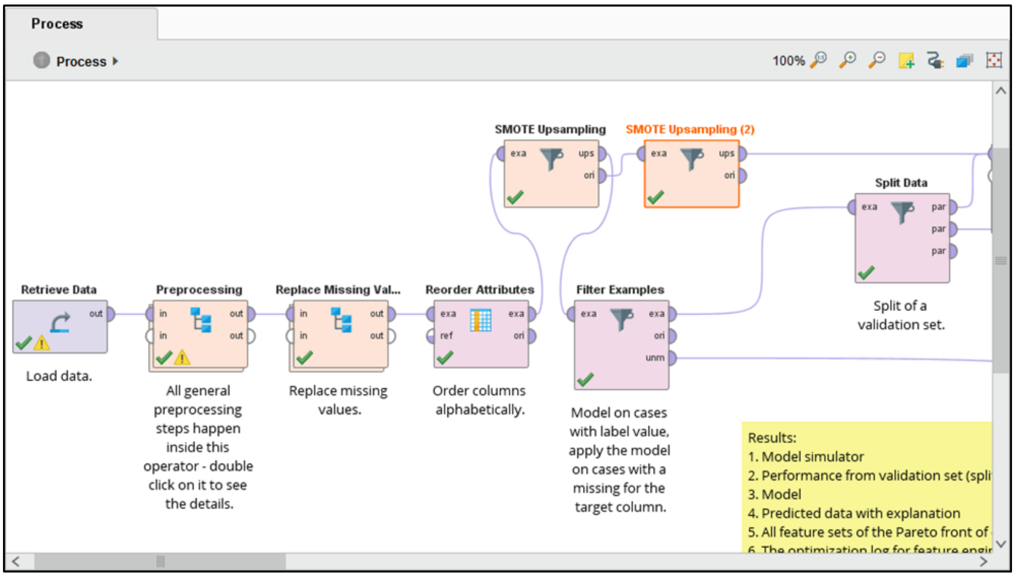

Figure 7. Rapidminer Automodel Modification by Implement First SMOTE upsampling

1. Run the whole process after implementing the first SMOTE upsampling, and observed the confusion matrix result
2. Repeat step 9 and 10
3. Implement second SMOTE upsampling as shown in the Figure 8. Second SMOTE upsampling is same as the first SMOTE upsampling with an additional ‘SMOTE upsampling’ box in the process flow.
4. Repeat step 9 and 10
5. Save the result of percentage of true positive on each class, of all experiments into 1 excel file. The result of overall experiment is shown in the result section.

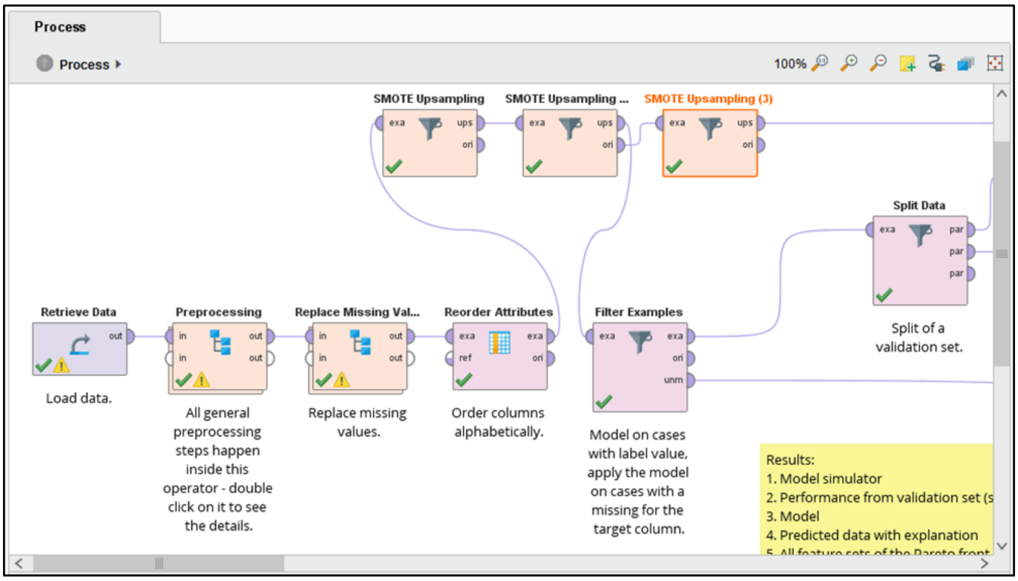

Figure 8. Rapidminer Automodel Modification by Implement Second SMOTE Upsampling

3. Results

This section will display the result of the experiments. Table 1 shows the result of the accuracy, execution runtime, percentage of each true positive class (PRU = Price Remain Unchanged, PF = Price Fall and PR = Price Rise) and standard deviation on the percentage of true positive distribution, conducted on all 9 different methods, based on Deep Learning algorithm.

Table 1. Result performance for Deep Learning classifier

Performance Measurement		**Sampling method**								
		Original Dataset			**First SMOTE Upsampling**			**Second SMOTE Upsampling**		
		SH	LI	ST	SH	LI	ST	SH	LI	ST
Accuracy (%)		89.35	88.69	89.44	91.97	85.47	91.81	80.71	37.03	80.06
True Positive	PRU	96.50	96.27	96.05	49.60	0.00	49.01	25.36	0.00	30.90
	PF	1.90	2.01	2.04	0.54	0.00	0.42	34.25	86.12	29.16
	PR	1.60	1.72	1.91	49.86	100.00	50.57	40.39	13.88	39.94
	SD	54.704	54.505	54.314	28.400	57.735	28.514	7.557	33.122	5.787
Runtime (sec)		9	9	9	38	42	38	98	103	101

SD=Standard Deviation; SH=Shuffled; LI=Linear; ST= Stratified

4. Findings and Discussion

From the result of 9 different sampling methods comparison based on deep learning algorithm, there are 2 main findings to find the suitable sampling method in constructing a data-driven FPL virtual player price changes prediction modelling. The 2 main findings are, findings on sampling method on the dataset and findings on split data into training and testing subsets. Each finding found will be explained in detail in the subchapter below.

4.1 Findings on Sampling Method on the Dataset

The first finding is, to handle a highly imbalanced output class label where the output class distribution between the most majority output class label and most minority output class label has a very huge difference, the best method is to do resampling the original dataset, either oversamling (also known as upsampling) the minority class or undersampling (also known as downsampling) the majority class. In this case study where the class distribution is highly imbalanced (16,743 price remain unchanged, 1,730 price fall and 553 price rise), downsampling the majority class until all classes will have fair distribution will lead to removal of potential useful data in majority class. Hence in this situation, performing SMOTE upsampling will be a better option compared to downsampling, in order to prevent removal of potential useful data, eventhough performing SMOTE upsampling will lead to increases the size of the dataset artificially and burden the computational learning algorithm. Note that a single SMOTE upsampling can only upsample one most minority class label, which means for multiclasses dataset, it needs to undergo multilayer SMOTE upsampling until all class label have the same distribution. Performing multilayer SMOTE upsampling until all the class labels have the same distribution is the best practice, to prevent a biased machine learning construction. In this experiment, although the original dataset and first SMOTE upsampling have quite a high accuracy on each algorithm, the ratio of the percentage of true positive are very biased, where the true positive on price fall is very little due to insufficient training set on these labels. Apart from that, the standard deviation values, which is dispersion on the ratio of the percentage of true positive are very high in this part, which means, there is a huge imbalance of true positive class label distribution contributed to the accuracy values. However, in the second SMOTE upsampling, the standard deviation value is lower compare to original and first SMOTE upsampling. This section shows that upsampling the dataset until all the output class labels distribution have the about the same distribution is the best practice in handling imbalance class label issues, compared to using an original highly imbalanced dataset or dataset that not undergo multilayer SMOTE upsampling until all classes have same distribution.

4.2 Findings on Sampling Method on Split Data into Training and Testing Subsets.

The next finding is, linear sampling in split data into the training and testing subset, is the worse sampling method if the dataset had been undergoing SMOTE upsampling (only for the dataset that has a huge difference amount between the most majority output class label and most minority output class label). It had been proved in Table 1 where the section of the combination of first SMOTE upsampling and linear sampling has 0.00% distribution of true positive on price remain unchanged, 0.00% distribution of true positive on price fall and 100% true positive on price rise. It is because when this dataset is being upsampled on the price rise label and append as the new synthetic row dataset in a huge amount, that particular upsampled data are being used as the 0.4 ratio of the testing subset. Linear sampling concept where it simply divided the dataset into partitions by a consecutive sequence row without changing the order of the dataset, forces the synthetic row dataset from the SMOTE upsampling will distributed more on testing subset, and distribute very little in the training subsets. That explains the reason why on the testing subset validation phase, there is no price remain unchanged and price fall class label in the confusion matrix. The same situation applied to the result of the combination of second SMOTE and linear sampling in split data, where price remain unchanged have 0.00% true positive.

The experiment result had shown that second SMOTE upsampling is the best practice while linear sampling is the worst sampling method in split the data into training and testing subset, if the dataset had been undergoing a very huge amount of row data being upsampled, in terms of having a fair distribution of true positive classes contributed for accuracy. Hence it will be in between the shuffled sampling and stratified sampling combined with second SMOTE upsampling to be selected as the best sampling method.

Then an additional experiment is conducted to measure the differences of these both sampling methods in split data into training and testing subsets. This experiment used an original dataset that had been undergoing SMOTE upsampling twice, used deep learning model, split data into training and testing subset (uses both shuffled and stratified then compare), and apply on the several different local random seed values, to see the amount of testing set class label distribution. Table 2 below indicates the result of this experiment.

Table 2. Experiment on performance comparison of shuffled and stratified sampling in split data

Deep Learning model, after being SMOTE upsampling twice		**Testing set percentage distribution**			
Local Random Seed (LRS)	Sampling Method	PRU	PF	PR	Standard Deviation
1990	Shuffled sampling	32.74%	33.91%	33.35%	0.585178
	Stratified Sampling	33.43%	33.22%	33.35%	0.046862
1991	Shuffled sampling	33.28%	32.52%	34.20%	0.841269
	Stratified Sampling	33.39%	33.36%	33.25%	0.073711
1992	Shuffled sampling	33.01%	33.12%	33.87%	0.468010
	Stratified Sampling	33.35%	33.31%	33.34%	0.020817

Table 2 above shows that stratified sampling in split data is much better compared to shuffled sampling, in providing a fair class label on both training and testing subsets, by having a lower standard deviation value on the dispersion of true positive classes percentage distribution. As a conclusion, having a multilayer SMOTE upsampling (in this case is second SMOTE upsampling) combined with stratified sampling in split data into training and a testing subset is the best method for handling highly imbalanced output class label, thus prevent a biased machine learning algorithm.

5. Conclusion

In this research paper where the frequency of class label "Price Remain Unchanged (PRU)" is higher than "Price Fall (PF)" and "Price Rise (PR)", a SMOTE upsampling method was applied, to ensure the fair distribution of all classes. A single SMOTE upsampling can only upsample one most minority class label, which means for multiclasses dataset, it needs to undergo multilayer SMOTE upsampling. Multilayer SMOTE upsampling can be done using formula, Iterative SMOTE upsampling = (amount of different output class label) – 1, which means that if the dataset has 3 kinds of output class label, it needs to undergo two times of SMOTE upsampling. Next, stratified sampling methodology of ensuring the class label distribution in subset having almost identical to the original dataset, had been proved in this research, as the best sampling method in handling highly imbalanced output class label dataset, compared to shuffled sampling and linear sampling, whose methodology based on random sampling and divides the dataset into partitions in a consecutive row of data respectively. This paper compared between the baseline (original) dataset, SMOTE-applied dataset and shuffled, linear and stratified sampling in split train-test subset, based on a deep learning algorithm. This paper also proposed criteria of low values in standard deviation (distribution of true positive on each class label on accuracy) as a measurement for finding the best method in handling imbalanced class labels. As a result, multilayer SMOTE until all the classes distribution is the same, combined with stratified sampling in split training and testing subset, got the lower standard deviation (5.7873), high accuracy (80.06%) and less execution runtime (1 minute 41 seconds) compared to the original highly imbalanced dataset.

References

Alejo. R., Garcia. V., Mollineda . R. A., Sanchez. J. S., & Sotoca. J. M. (2007). *The class imbalance problem in pattern classification and learning.* Dept de Llenguatjes i Sistemes Informatics, Universitat Jaume I, Spain.

Anand. V. (2018). *Fantasy Premier League dataset from season 2016/2017 [Data file]*. Retrieved from Github: https://github.com/vaastav/Fantasy-Premier-League/tree/master/data/2016-17/gws

Barandela. R., Garcia. V., Rangel. E., & Sanchez. J. S. (2002). Strategies for learning in class imbalance problems. *The journal of the recognition society*, 849-851.

Benabbou. F., Sadgali. A., & Sael. N. (2019). Performance of machine learning techniques in the detection of financial frauds. *Second International Conference on Intelligent Computing in Data Sciences (ICDS 2018)* (pp. 45-54). Morocco: Elsevier.

Bowyer. K. W., Chawla. N. V., Hall. L.O., & Kegelmeyer, W. P. (2002). SMOTE : Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research, vol 16,* 321-357. AI Access Foundation and Morgan Kaufmann Publishers.

Chung. J. Y & Lee. S. (2019). Droupout early warning system for high school students using machine learning. *Children and Youth Services Review*, 346-353.

Glauner. P., State. R., & Valtchev. P. (2017). *Impact of Biases in Big Data.* Luxembourg: National Research Fund.

He. C., Jiang. X., Xiao**.** J., & Xie. L. (2011). Dynamic classifier ensemble model for customer classification with imbalance class distribution. *Expert System with Application 39 (2012)*, Elsevier Ltd.

Helmenstine. A. M. (2018, September 27). *How to calculate population standard deviation*. Retrieved from ThoughtCo.: https://www.thoughtco.com/population-standard-deviation-calculation-609522

Ho. S., Ng. M. K., Wu. Q., Ye. Y., & Zhang. H. (2014). ForesTexter: An efficient random forest algorithm for imbalanced text categorization. *Knowledge Based System 67 (2014)*, Elsevier B. V.

Kaur. H., & Kumari. V. (2018). Predictive modelling and analytics for diabetes using a machine learning approach. *Applied Computing and Informatics*.

Kubat. M., & Matwin. S. (1997). Addressing the Curse of Imbalanced Training Sets: One sided Selection. *Proceedings of the 14th International Conference on Machine Learning* (pp. 179-186). Nashviille USA: University of Ottawa.

Liu. J., Luo. X., Tang. Y., Xu. Z., Yang. Z., Yuan. P., Zhang. T., & Zhang. Y. (2019). Software defect prediction based on kernel PCA and weighted extreme learning machine. *Inforrmation and Software Technology*, 182-200.

Ma. Z., Wang. G., Wang. Z., Xue. J., & Zhu. R. (2018). LRID : A new metric of multi-class imblance degree based on likelihood-ratio test. *Pattern Recognition Letters 116 (2018)*, 36-42.

Rocca. B. (2018, January 28). *Handling imblanced datasets in machine learning*. Retrieved from Towads Data Science: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28

Severin. E., & Veganzones. D. (2018). An investigation of bankruptcy prediction in imbalanced datasets. *Decision Support Systems 112 (2018)*, 111-124.

Stapor. K. (2018). Evaluating and Comparing Classifiers: Review, Some Recommendations and Limitations. *Proceedings of the 10th International Conference on Computer Recognition System CORES 2017. Advanced in Intelligent and Computing, vol 578.* Springer.