Open in app

Published in MLearning.ai

Angelleoncollado Follow
Jan 23 · 9 min read · Listen

Save

Data sampling methods for imbalanced data

Most machine learning algorithms are designed to work with the same proportion of observations for each class when we are facing a classification problem. Because of that, when there is a class with many fewer observations than the majority class (imbalanced datasets), the algorithm ignores that class, reducing its performance and its application capacity. To improve it, one solution is to use data sampling algorithms. Data sampling methods provide several techniques to balance the volumetrics of both classes, both increasing the minority class (oversampling) and reducing the majority class (undersampling).

The effect of imbalanced data in one model could be seen in Fig. 1. In this case, we have three classes: class 2 which is the majority class and class 1 and 0 which are the minority classes. Since we are interested in learning from these data, and it is a multi-classification problem, if we considered a linear SVC model to classify our data set, we can see the effect of adding more data on the minority classes regarding the model classification. At the first train data set, the imbalanced data is abruptly and the classifier can only predict two classes. Even if one person, like you, see this data and think if we can do a better performance of the model, we would say no since not enough data we have to differentiate the three classes. In the opposite case, in the last example (bottom-right image), all parts have the same volumetric, around 330 examples. In this case, the model is working quite well, and a linear model like Linear SVC is enough to guarantee a good performance to delimit the boundary between these three classes.

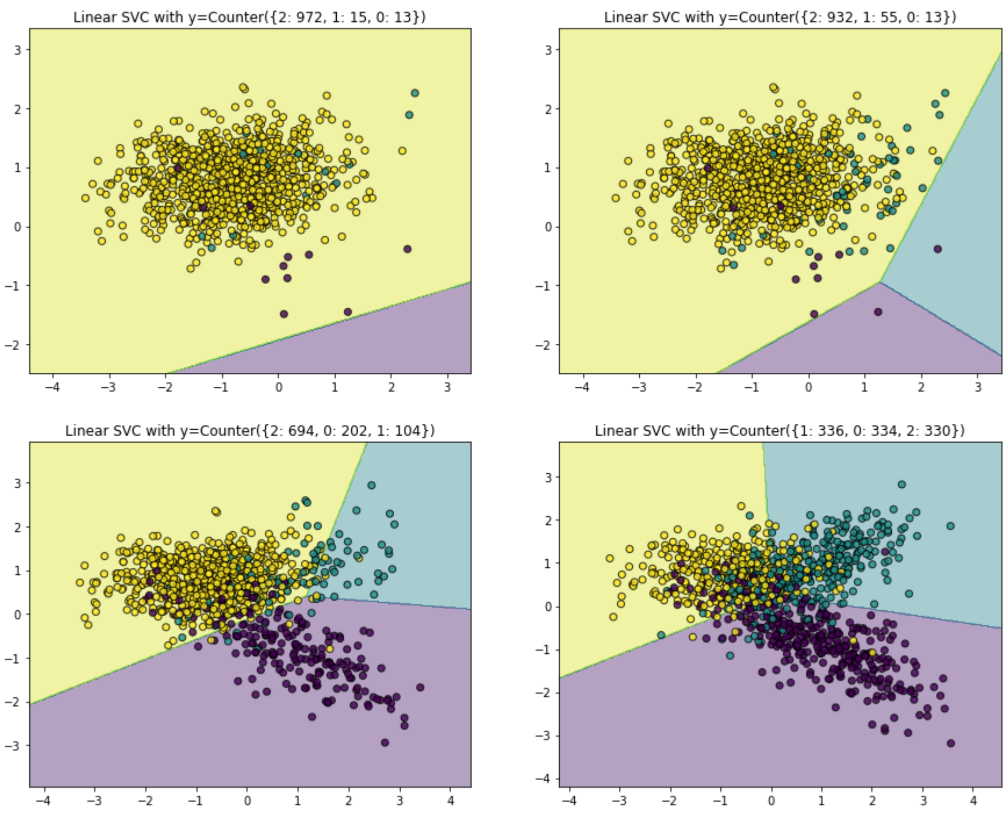

Fig. 1: Effect of imbalanced data on machine learning algorithms

Imbalanced data could not be an issue if the minority classes have enough data to be treated statistically and generate synthetic data, or undersampling the majority class. The big challenge in most of the critical imbalanced datasets lies in the fact that you should apply data sampling techniques carefully to not change the original distribution or the original information of the data.

Data sampling provides a collection of techniques that transform a training dataset for better balancing. There is not the best method for data sampling: as in life, there are many data sampling methods and each one is better depending on the application or model used in the project. Therefore, it is really important to carry out sensitivities studies about the effect of the sampling method on the machine learning model. We can differentiate two data sampling method categories:

Random Oversampling: This technique consists of duplicating examples randomly from the minority dataset. It could be effective for datasets with a skewed distribution where duplicate examples could improve the model performance. However, the reader should pay attention to overfitting the model since this technique duplicates examples. Therefore, it is important to carry out sensitivity studies for both training and test datasets and do metrics evaluation to prevent overfitting. Random oversampling is implemented using the *RandomOverSampler* class in *imblearn* library.

Synthetic Minority Oversampling Technique (SMOTE): This technique selects a minority class instance at random and finds its k nearest minority class neighbours. The synthetic instance is then created by choosing one of the k nearest neighbours b at random and connecting a and b to form a line segment in the feature space. The synthetic instance is generated as a convex combination of the two chosen instances a and b. The reader should pay attention to the data distribution after applying smote since the original data has an important impact on this strategy. In addition, k-neighbours is a hyperparameter to be tested to improve model learning. SMOTE is implemented using the *SMOTE* class in *imblearn* library.

Borderline-SMOTE: In this case, it consists of a modification of the SMOTE technique, focusing on the boundary of both classes, by using a KNN. A classification problem consists mainly in establishing the boundary line between the different classes. It could be, as we saw in the previous image, that the boundary line is not well defined and it is to distinguish between classes in that line, including the classification model. Therefore, this algorithm consists of applying SMOTE technique around the borderline between classes, to improve their limits and improve the model performance. Borderline-SMOTE is implemented using the BorderlineSMOTE class in *imblearn* library.

Borderline-SMOTE SVM: The technique consists of a methodology very similar to the previous technique, borderline-smote, with the difference that the boundary-line is not imposed by the dataset itself, but by the SVM algorithm, prior learning. Therefore, the SVM is used to locate the boundary-line and the SMOTE algorithms works based on this borderline. SVM SMOTE is implemented by using the SVM SMOTE class in *imblearn* library.

Adaptive Synthetic Sampling (ADASYN): This technique involves generating synthetic samples inversely proportional to the density of the examples in the minority class. This technique generates synthetic examples in regions space where the density of minority sample is low, and fewer where the density of the sample is high. ADASYN is implemented by using the ADASYN class in *imblearn* library.

GAN: Generative adversarial networks can be used as an oversampling tool. It is a new topic and there are many papers in the literature explaining the benefits of using the GAN technique for oversampling.

Data augmentation: It consists of generating additional data to increase the model performance. It is used mainly on images and language processing. Some techniques on data augmentations are the following: adding noise, cropping, flipping, scaling, wavelet transformation, etcétera.

Undersampling

Undersampling techniques delete a subset of examples from the majority class. The most usual techniques are:

Random Undersampling: This technique involves randomly selecting examples from the majority class to delete examples from the training dataset. Commonly, the majority class dataset is reduced randomly to make more uniform the volumetric of both classes. In this case, the reader should pay attention to not removing important examples. If the majority class is large enough and uniformly distributed, removing randomly examples does not change the fundamentals of the statistics and apply this technique could be interesting. Random undersampling is implemented using the *RandomUnderSampler* class in *imblearn* library.

New Miss Undersampling: In this case, three different types of these techniques are in the literature: NearMiss-1 chooses examples from the majority class that have the smallest average distance to the three closest examples from the minority class, NearMiss-2 selects examples from the majority class that have the smallest average distance to the three furthest examples from the minority class, NearMiss-3 involves selecting a given number of majority class examples for each example in the minority class that is closest. The reader should care about applying this methodology since it could reduce drastically the number of examples of the majority class and concentrate them closed to the minority class. The effect of this methodology could be reducing drastically a lot of information from the majority class. NewarMiss is implemented using the *NearMiss* class in *imblearn* library.

Condensed Nearest Neighbor Rule (CNN): This technique is much different from the previous one. It consists of using a KNN algorithm for classifying the different classes, and the majority set that cannot be classified correctly are added incrementally to the store. Therefore, It stars with one store of the minority class, and it works by adding examples from the majority class that cannot be classified correctly. It is an undersampling technique that seeks a subset of a collection of samples that results in no loss in the model performance of a KNN algorithm. The reader should pay attention to the following: this technique is focused on the border-line between classes and there are not saving the datasets from the other parts. CondensedNearestNeighbour is implemented using the *CondensedNearestNeighbour* class in *imblearn* library.

Tomek Links Undersampling: This technique is based on the CNN but modified. One of the issues of CNN technique is that examples are selected randomly. The modification of this technique is the selection of pairs of examples, one from each class, with the smallest Euclidean distance of each other. Once the volume of pairs remains constant in the algorithm, they are removed from the majority class. TomekLinks is implemented using the *TomekLinks* class in *imblearn* library.

Edited Nearest Neighbours Rule (ENN): This rule involves using k=3 nearest neighbours to locate those examples in a dataset that are misclassified and that are then removed before a k=1 classification rule is applied. It can be applied to each example in the majority class, allowing those examples that are misclassified as

One-Sided Selection (OSS): This technique combines Tomek Links and CNN. Tomek links are ambiguous points on boundary and are identified and removed in the majority class. The CNN method is then used to remove redundant examples from the majority class that are far from the decision boundary. OneSidedSelection is implemented using the *OneSidedSelection* class in *imblearn* library.

Neighbourhood Cleaning Rule (NCR): It combines both the CNN rule to remove redundant examples and the ENN rule to remove noisy or ambiguous examples. The CNN is applied in a one-step manner, then the examples that are misclassified according to a KNN classifier are removed, as the ENN rule. Unlike OSS, fewer of the redundant examples are removed and more attention is placed on cleaning those examples that are retained. The reason for this is to focus on the quality (unambiguity) of the examples that are retained in the majority class. NeighbourhoodCleaningRule is implemented using the *NeighbourhoodCleaningRule* class in *imblearn* library.

Images compression techniques: There are many compression techniques such as SVD (PCA), Fourier transform, wavelet basis transformation, convolutions techniques, and many techniques for compression information that could you applied to reduce a data set, always keeping in mind the effect of this reduction into the model.

Combine Data Sampling Methods

There are combinations of oversampling and undersampling methods that have been proven effective techniques and together may be considered as sampling techniques.

Random Oversampling and Undersampling

They can be effective when combined. Random oversampling involves randomly duplicating examples in the minority class, whereas random undersampling involves randomly deleting examples from the majority class. It is a middle point between both techniques.

SMOTE and Random Undersampling

SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples and creating a new sample as a point along that line. It is recommended to use SMOTE on the minority class, followed by an undersampling technique on the majority class.

SMOTE and Tomek Links Undersampling

SMOTE is an oversampling method that synthesizes new plausible examples in the minority class. Tomek Links refers to a method for identifying pairs of nearest neighbours in a dataset that have different classes. Removing one or both of the examples in these pairs (such as the examples in the majority class) has the effect of making the decision boundary in the training dataset less noisy or ambiguous. The goal of using Tomek Links is to better define the boundary between classes in a classification problem.

SMOTE and ENN

SMOTE may be the most popular oversampling technique and can be combined with many different undersampling techniques. Another very popular undersampling method is the Edited Nearest Neighbors, or ENN. This rule involves using k = 3 nearest neighbours to locate those examples in a dataset that are misclassified and that are then removed. It can be applied to all classes or just those examples in the majority class. Other combinations of data sampling methods could be the followings:

> *Condensed Nearest Neighbors + Tomek Links*
>
> *SMOTE + Tomek Links*
>
> *SMOTE + Edited NearestNeighbors*

Conclusions and recommendations

As the reader can see, there are different techniques to approach data sampling within a machine learning project. Apriori it is unknown which technique will be the most appropriate for the application, or even for the used model. Therefore, It is necessary to use different techniques, to combine undersampling techniques with oversampling techniques, and to measure the impact of their effect on model training. It is important that these techniques should be used in the training step. It is also necessary to measure the different model performance metrics of the test and validation steps, without using data sampling techniques, in order to see how the model works properly and how the model has learnt. Moreover, it should be emphasized that there is no one methodology that is better than another, since depending on the data set as well as the model and the application, one technique will be more interesting than another. It is always necessary to test different techniques, combine undersampling with oversampling, and perform sensitivity analysis to see how they improve the model. The reader could learn in deep consulting some books like "Imbalanced Classification with Python" and some projects in Kaggle. And finally, the reader should never forget to enjoy and be happy working. Knowing that the most important thing in life is to be happy. So, I hope you have enjoyed this reading and have fun with it!

Mlearning.ai Submission Suggestions

How to become a writer on Mlearning.ai

Open in app

132

Sign up for Machine Learning Art

By MLearning.ai

Be sure to SUBSCRIBE here 🔵 to never miss another article on Machine Learning & AI Art Take a look.

Get this newsletter

Emails will be sent to siqifang47@gmail.com.
Not you?