Get unlimited access to the best of Medium for less than $1/week. Become a member

Dealing with “rare” events in Machine Learning

Deepa Mahidhara · Follow
3 min read · May 2, 2019

142 1

If you are in Data Science, sooner or later, you will have to deal with a common problem — “rare” events! If the rate of occurrence of the predicted event is less than 5%, it is generally considered a rare event.

Why is this a problem? Because most machine learning algorithms assume that the classes are balanced, so we need to apply special techniques if the classes are not balanced, otherwise our predictions will not be accurate. I will discuss three techniques for dealing with rare events or imbalanced classes.

Recently I was working with historical Twitter data for a Disaster Management System. The goal of the system was to identify road closures during natural disasters based on live social media data.

In order to train my model, I gathered 8000 tweets during Hurricane Florence in September 2018. Using NLP techniques, I classified these tweets as a “road closure” tweet if it had specific information on road closures.

After cleaning and data wrangling, I had ~350 tweets classified as “road closure” and ~7600 tweets classified as non-road closure, i.e. **minority class was only 4.6% of the observations**.

After lemmatizing and tokenizing the words, I decided to fit a Logistic Regression model since it was a classification problem.

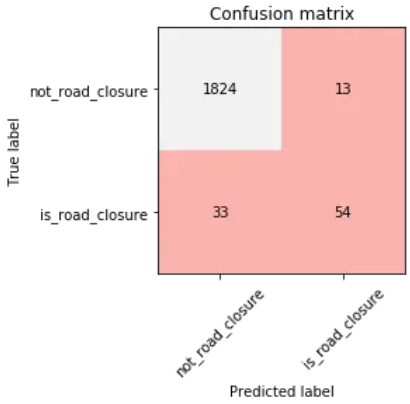
Metric

The first technique I used for predicting this rare event was to change the scoring metric.

Generally, we evaluate the performance of machine learning algorithms using predictive Accuracy. However, this is not appropriate when the classes are heavily imbalanced, because the costs of different types of errors vary

significantly. In the case of a Disaster Management system, it is very important to predict true road closures with a high level of accuracy, so that emergency responders don't get stuck due to a False Positive prediction, i.e. model predicting an open road when the road is actually closed.

So, instead of measuring Accuracy, I measured the ROC AUC score, as that would minimize False Positives.

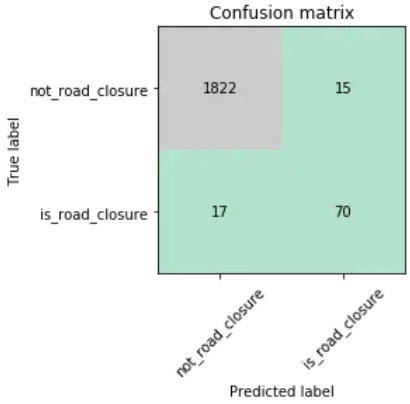


The ROC AUC score for the Logistic Regression model was 0.80. But more importantly, out of 87 Total Positives, the model predicted 33 False Positives,

Model

To improve upon the results of Logistic Regression, I tried boosting — another technique for handling imbalanced classes.

There are many boosting techniques available in Machine Learning. I used Gradient Boosting, which in very simplistic terms, over-weights the weak learners (observations that are difficult to classify), and under-weights the strong learners, while optimizing a user-specified loss function.

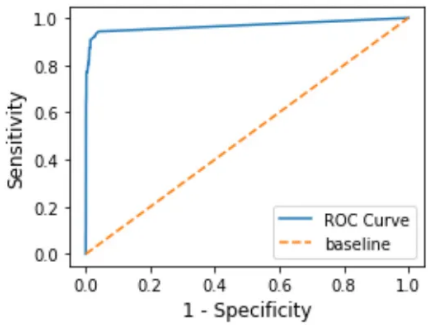


By doing this, my ROC AUC score jumped from 0.80 to 0.90, and I was able to reduce my False Positives by 45% to 17, which was a significant improvement over Logistic Regression.

Over-sampling

Another technique for handling imbalanced classes is SMOTE (Synthetic Minority Over-Sampling Technique), which is an over-sampling technique that generates synthetic samples for the minority class.

By applying SMOTE to my imbalanced data, and then using Gradient Boosting, the new ROC AUC score increased to 0.96, and I was able to reduce the False Positives to from the original 33 down to 7, and had an almost perfect ROC curve.



These are by no means the only techniques available to deal with rare events. However, I stopped here as another modeling tip that I have picked up over time is to not over-tweak the model - it becomes a case of diminishing returns!

Machine Learning | Data Science | Imbalanced Class | Confusion Matrix