# Class Imbalance

Many machine learning algorithms used for classification contain implicit assumptions about having relatively equal numbers of examples of each class. For example you might train a logistic regression model to predict whether or not an email is spam, or whether or not a customer is likely to churn (i.e. leave a service). Such a model will usually have an easier time classifying these cases if there are similar numbers in each class.

However, there are many important problems where a reasonable balance between classes cannot be expected. A common example from business is in fraud detection, such as when a credit card processor flags a transaction as potentially fraudulent. A large credit card processor will go through many millions of transactions per day and the vast majority of these are likely to be legitimate, so finding suspicious ones requires particular care. Fraud detection in this context presents a prototypical problem of class imbalance.

Dealing with problems of class imbalance often requires finesse, and this unit introduces several potential approaches. These methods employ different tactics to avoid being overwhelmed by the majority class. Typically being overwhelmed during the model's training directly affects the model's accuracy. Accuracy, that is the proportion of predictions that are correct, is an intuitive metric for classification, but can be misleading with imbalanced classes. For example, if one percent of a processor's transactions in a training sample are fraudulent, then a model that *always* predicts "not fraudulent" will be 99% accurate but of no practical use.

**WARNING: Accuracy is not an appropriate metric for imbalanced classes. Use F1 Score or another metric instead.**

The solution to misleading metrics is to break the correct vs. incorrect dichotomy of binary classification into four pieces: true positives, true negatives, false positives, and false negatives. There are many ways to combine these values with an even larger number of naming conventions.

| | | **True condition** | | | | |
|---|---|---|---|---|---|---|
| | Total population | Condition positive | Condition negative | Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$ | Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$ | |
| **Predicted condition** | Predicted condition positive | **True positive** | **False positive**, Type I error | Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$ | False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$ | |
| | Predicted condition negative | **False negative**, Type II error | **True negative** | False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$ | Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$ | |
| | | True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$ | False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$ | Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$ | Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$ | $F_1$ score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$ |
| | | False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$ | Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$ | Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$ | | |

(For more details, and the source of the table, click here: https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers).

Conventionally the rare or minority class (e.g. fraud) is designated as the positive class, so two of the more commonly used alternative metrics to accuracy—precision and recall—have the *true positives* in their numerators. You'll also see F1 score, which is the harmonic mean of precision and recall.

Having better metrics for measuring performance in imbalanced classification is necessary, but not sufficient, to get better results. A simple logistic regression model optimizes a cost function where the cost of a false negative is the same as a false positive, so when negative training examples are much more common the results will tend toward the "always predict negative" end of the spectrum. You can of course adjust your cost function to weigh the errors differently, and certain ML models are more amenable to this than others. But we will start with a different set of solutions where we adjust the training data to make the classes more balanced. This involves either up-sampling or down-sampling your data, making the minority class more common through some type of repetition or making the majority class less common by throwing away some observations.

Different machine learning models vary in how well they deal with making predictions in imbalanced situations, so we end this section with a few examples of models that tend to perform well in such circumstances.