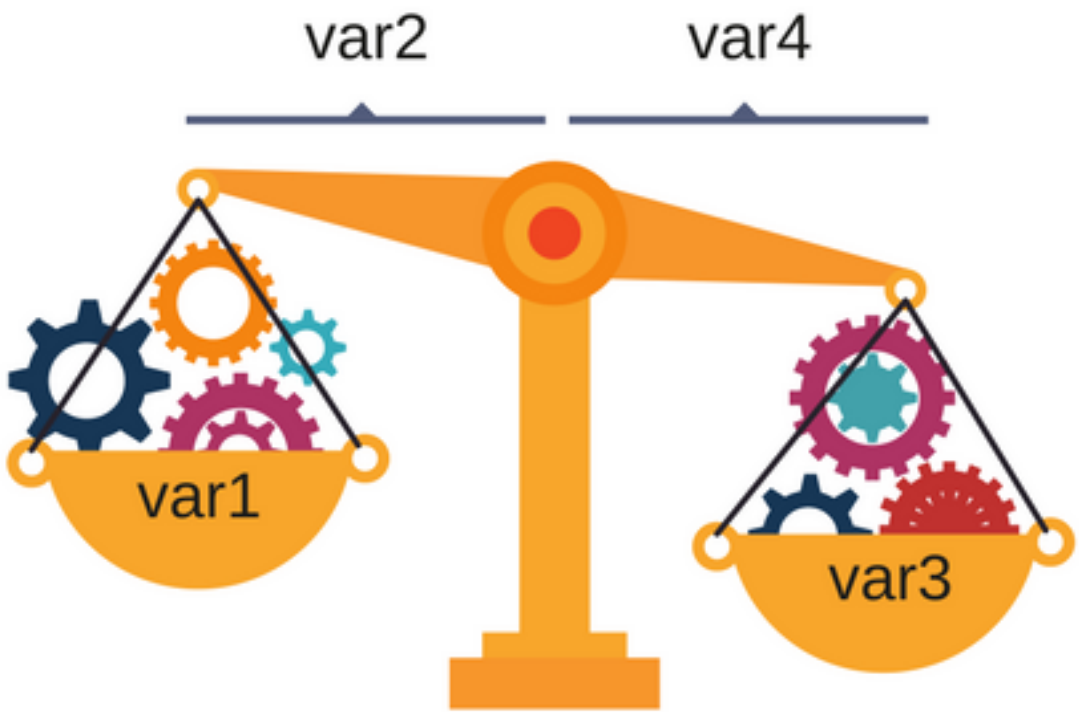
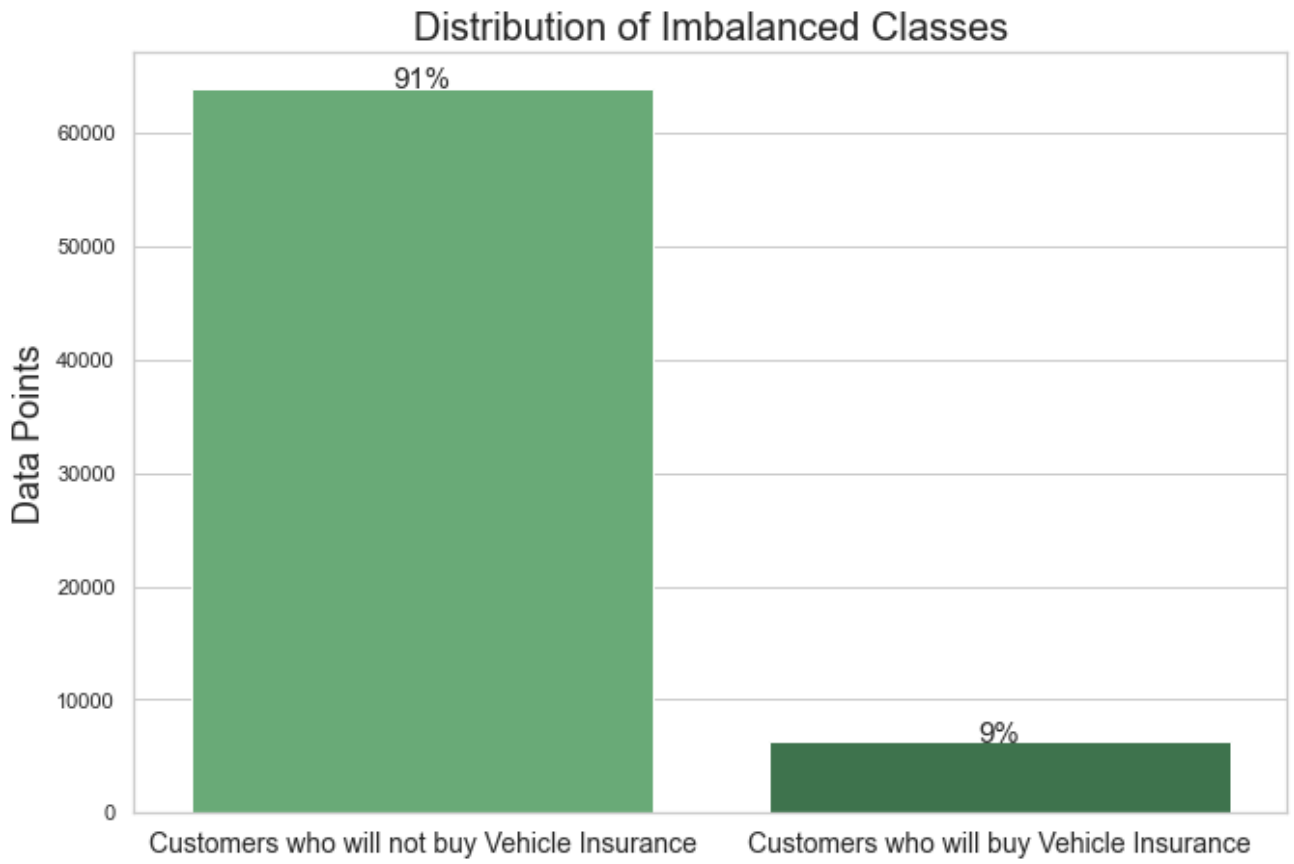
Imbalanced Dataset



Imbalanced Dataset



Is there any rule?

There isn't a strict rule or universally agreed-upon threshold for what constitutes an imbalanced dataset. It's often context-dependent and can vary based on the specific problem, the nature of the data, and the goals of the analysis or model.

However, a commonly used heuristic is the 80-20 rule, where a dataset is considered imbalanced if the class distribution is roughly 80% to 20% or worse. In this case, the majority class would have around 80% of the samples and the minority class around 20%.

That being said, what's considered imbalanced can vary widely. In some cases, a class distribution of 60-40 might be considered imbalanced, especially if the minority class is critical or costly to misclassify. In other cases, a distribution of 90-10 might be considered balanced, especially if the classes are naturally imbalanced in the real-world scenario to which the model will be applied.

Ultimately, it's important to consider the specific domain, the implications of misclassifications, and the goals of the analysis when determining whether a dataset is imbalanced. Additionally, the choice of threshold might be influenced by practical considerations and domain expertise.

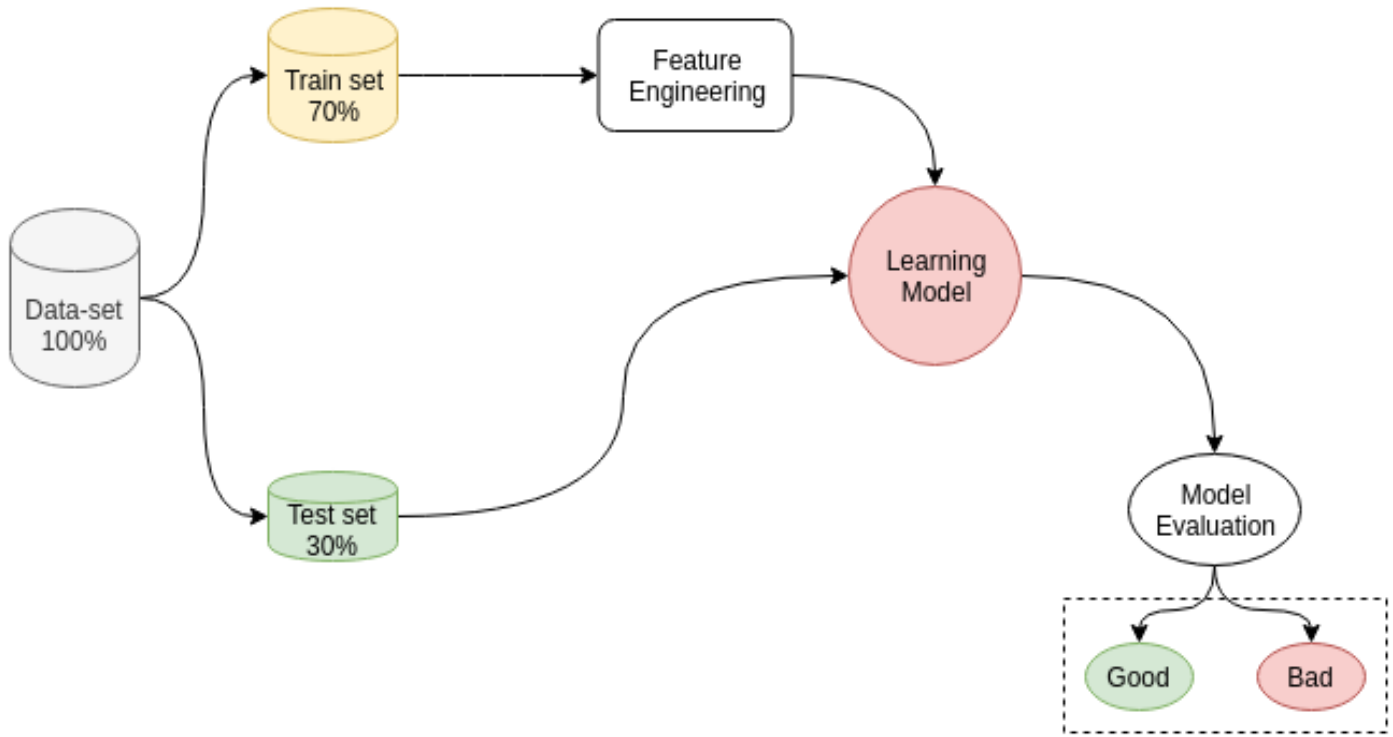
Cross Validation

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. It is a technique used to protect against overfitting in a predictive model, particularly in a case where the amount of data may be limited. In cross-validation, you make a fixed number of folds (or partitions) of the data, run the analysis on each fold, and then average the overall error estimate.

Cross Validation

- Hold Out Cross Validation
- K-Fold Cross Validation
- Leave One-Out Cross Validation (LOOCV)
- Stratified K Fold Cross Validation

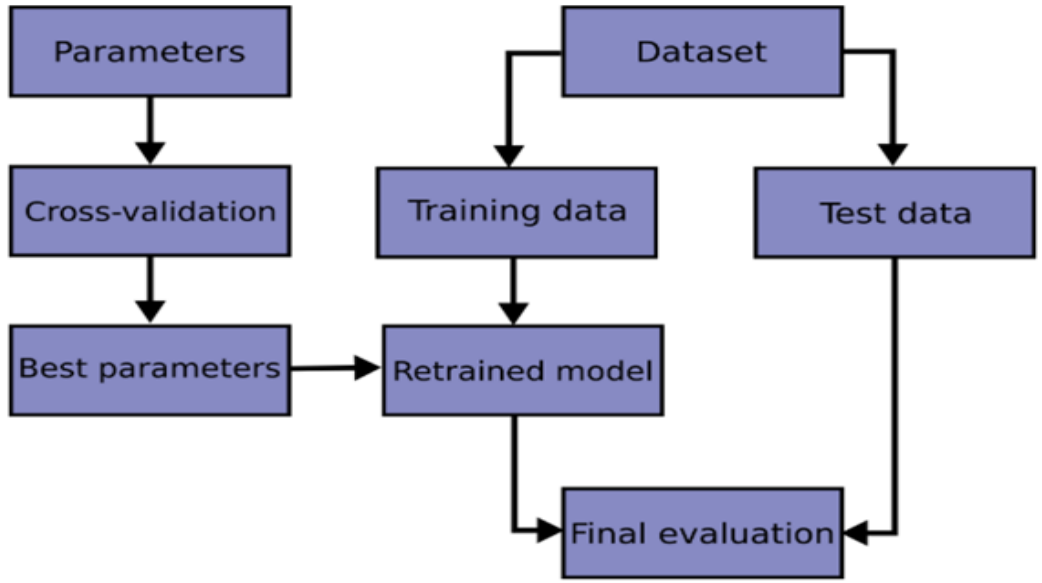
Machine Learning Model



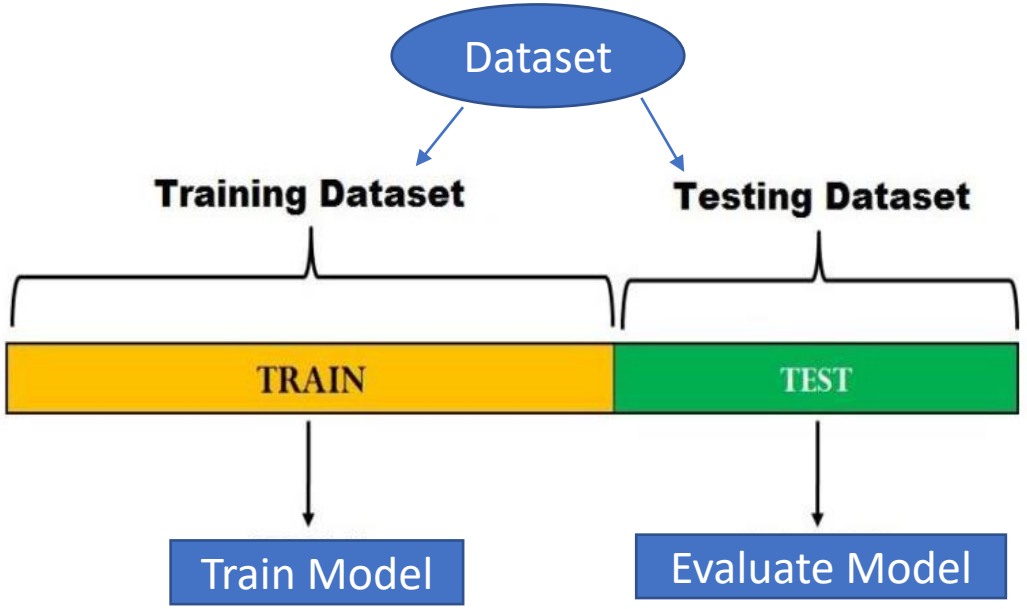
Cross Validation

Cross-validation *is a resampling procedure used to evaluate machine learning models on a limited data sample.*

K = 5



Hold Out Cross Validation



Hold Out Cross Validation

```
from sklearn.model_selection import train_test_split

xtrain, xtest, ytrain, ytest = train_test_split(x,y, train_size=0.7, random_state=1)
```

K-Folds Cross Validation

The general procedure is as follows:

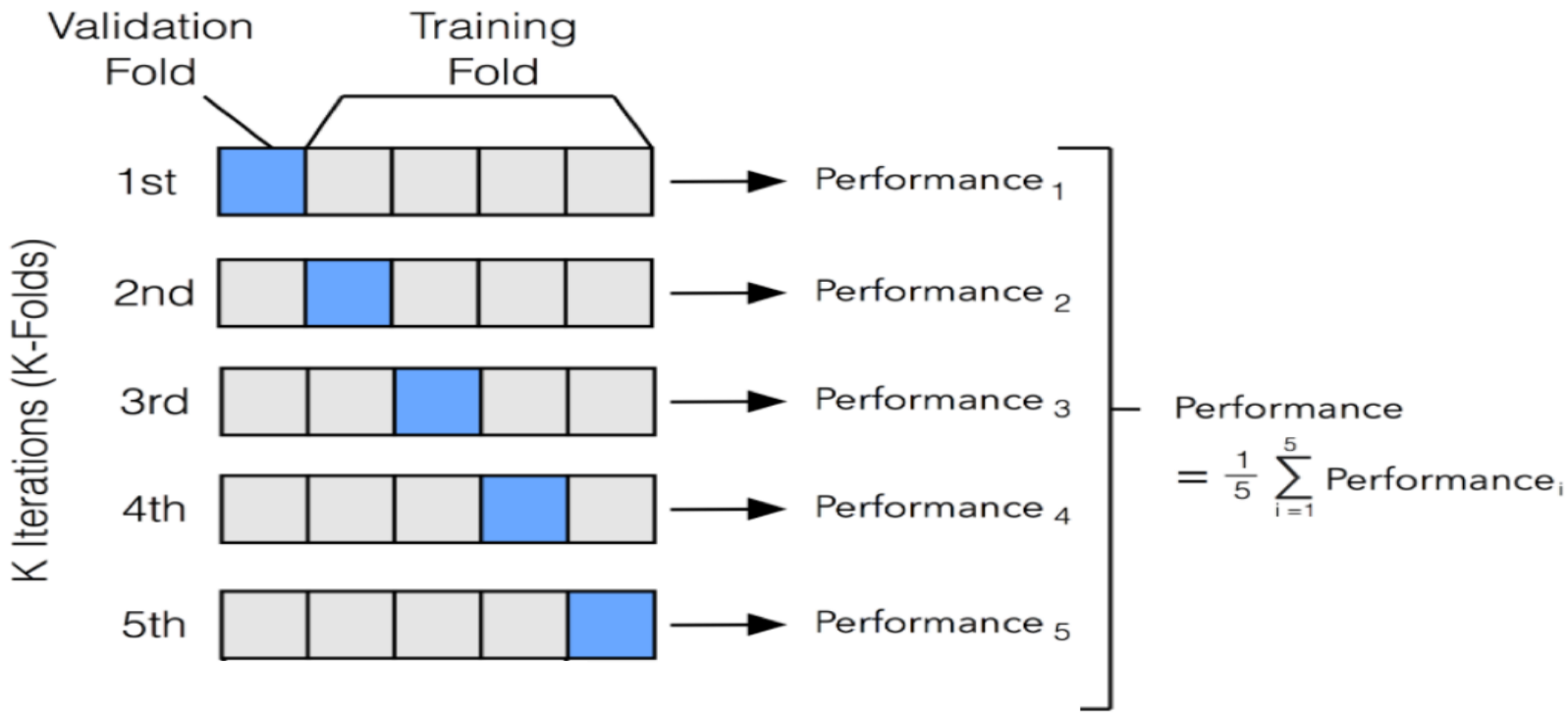
1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:
    1. Take the group as a hold out or test data set
    2. Take the remaining groups as a training data set
    3. Fit a model on the training set and evaluate it on the test set
    4. Retain the evaluation score and discard the model
4. Summarize the skill of the model using the sample of model evaluation scores

K-Folds Cross Validation

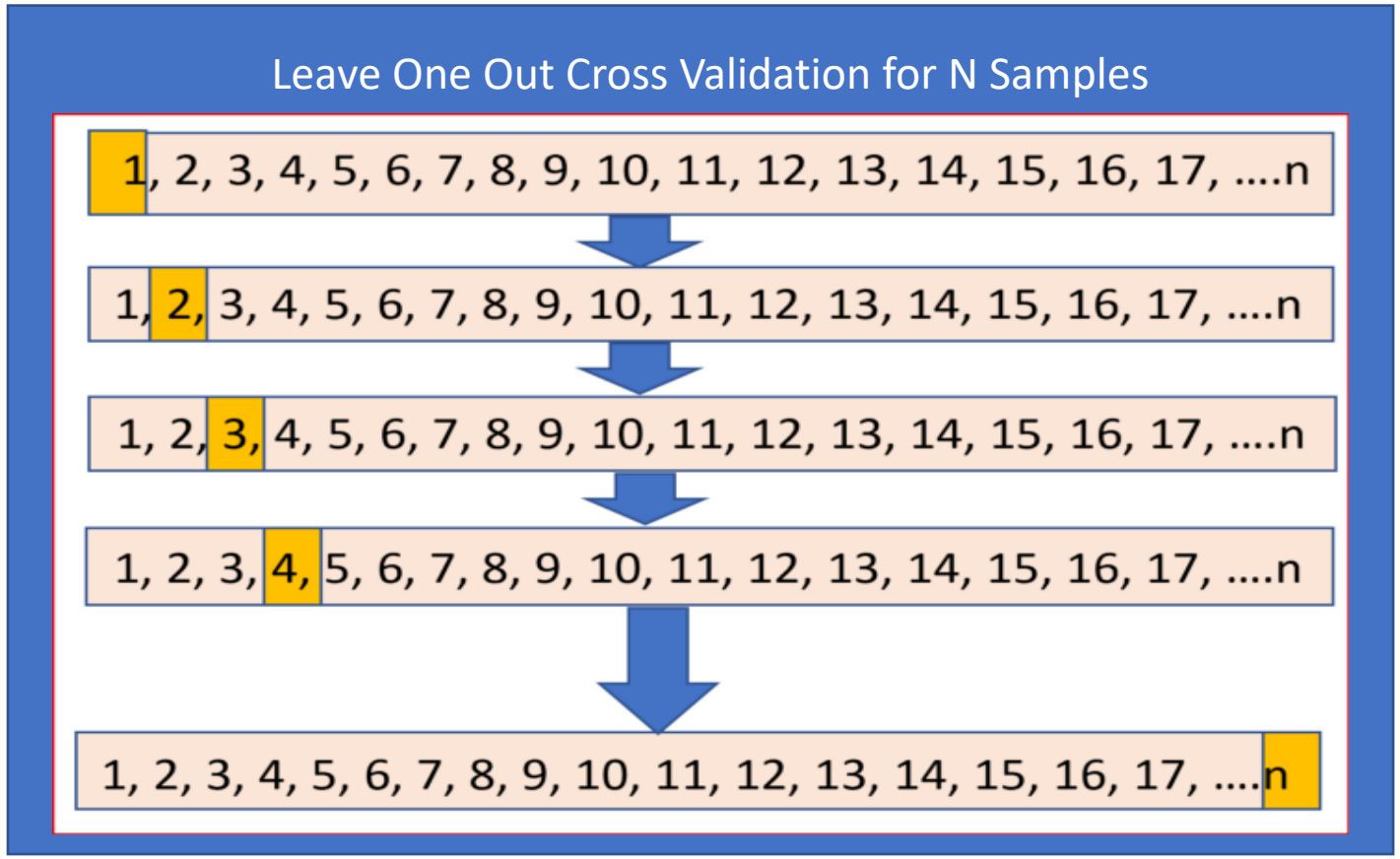
The general procedure is as follows:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:
    1. Take the group as a hold out or test data set
    2. Take the remaining groups as a training data set
    3. Fit a model on the training set and evaluate it on the test set
    4. Retain the evaluation score and discard the model
4. Summarize the skill of the model using the sample of model evaluation scores

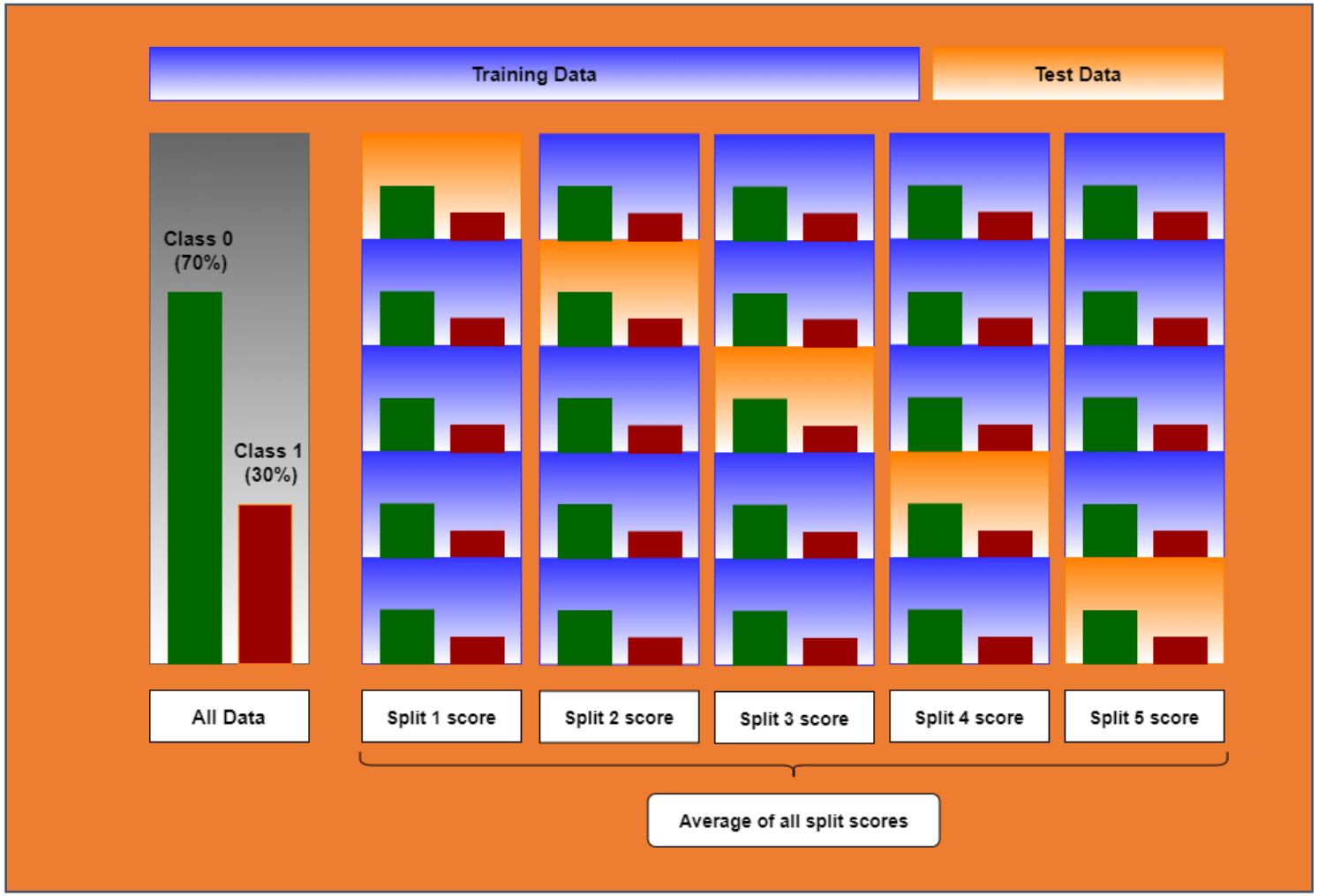
K-Folds Cross Validation



Leave One-Out Cross Validation (LOOCV)



Stratified K Fold Cross Validation



Let’s Do it with PYTHON