3.11 Evaluating a Fraud Detection Model

3.11.1 Key Decisions

When evaluating predictive models, **two key decisions** need to be made.

❓ Decision Concerns:

1. Dataset split up.
2. Performance metrics.

3.11.2 Splitting Up the Data Set #Split

The decision how to split up the data set for performance measurement depends on its size.

3.11.2.1 Large Datasets

In large data sets, the data can be split up into:

- (70%) **Training dataset** to *build the model*.
- (30%) **Test dataset** to *calculate its performance*.

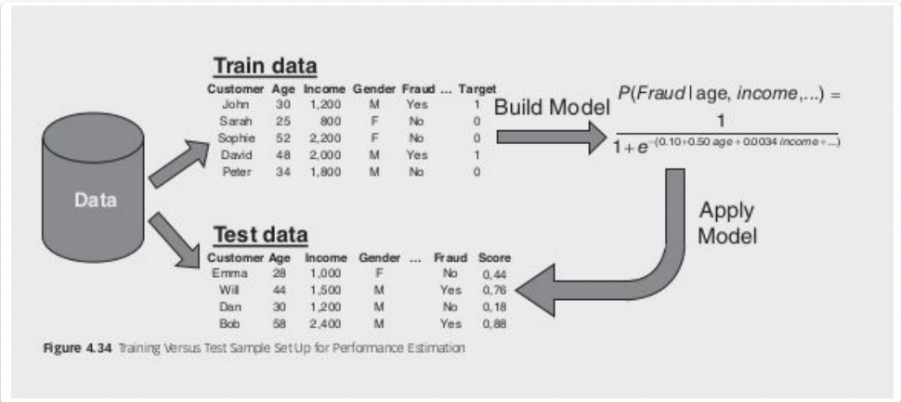


Figure 4.34 Training Versus Test Sample Set Up for Performance Estimation

Strict separation between training and test sample:

- *No observation that was used for training, can be used for testing*.

In case of decision trees or neural networks, the **validation sample is a separate sample** -> it is *used during model development* (i.e., to make the stopping decision).

- 40% training, 30% validation, and 30% test sample.

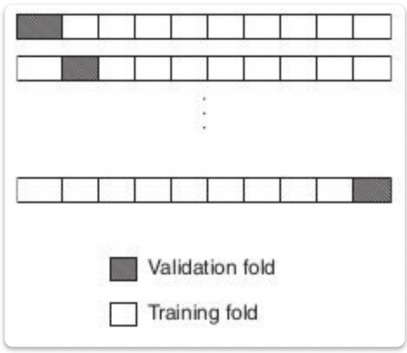
Stratified split-up ensures that *fraudsters / nonfraudsters are equally distributed amongst the various samples*.

3.11.2.2 Small Dataset

In small data sets, special schemes need to be adopted.

- **Cross-validation**:

  - The *data is split into K folds* (e.g., 5, 10). An analytical model is then trained on K – 1 training folds and tested on the remaining validation fold.
  - Repeated for all possible validation folds resulting in *K performance estimates*, which are averaged.



- **Leave-one-out cross-validation**:

  - Every observation is left out in turn and *a model is estimated on the remaining K – 1 observations*.
  - This gives K analytical models in total.

> *In stratified cross validation, make sure the no fraud/fraud odds are the same in each fold.*

3.11.2.2.1 MODEL SELECTION

Cross-validation gives *multiple models*.

🔑 **Key question**: "what should be the final model that is being outputted from the procedure"?

- All models collaborate in an **ensemble** setup by using a (weighted) voting procedure.
- Do **leave one out cross-validation** and *pick one of the models at random*. Since the models differ up to one observation only, they will be quite similar anyway.
- Build **one final model on all observations** but **report the performance coming out of the cross-validation** procedure as the best independent estimate.

3.11.2.2.2 BOOTSTRAPPING PROCEDURES

Bootstrapping procedures: one takes *samples with replacement from a dataset D*.

- The probability that a customer **is** sampled equals:
  - $\frac{1}{N}$, with N $the *number of observations in the data set*.
- The probability that a customer **is not** sampled equals:
  - $1 - \frac{1}{N}$

Assuming a bootstrap with N samples, the *fraction of customers that is not sampled equals*:

- $(1-\frac{1}{N})^N$

We then have: $\lim_{n\to\infty}(1-\frac{1}{N})^N = e^{-1} = 0.368$
where the *approximation already works well for small values of* N.

- *0.368 probability* that **a customer does not appear** in the sample.
- *0.632 probability* that **a customer does appear**.

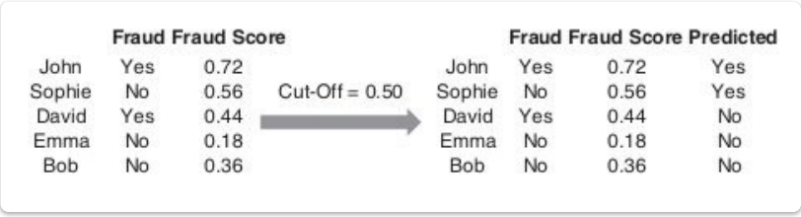
If we then take:

- The training set: the *bootstrap sample*.
- The test set: as *all samples in D but not in bootstrap*.

We can **calculate the performance as follows**:
$Error\,estimate = 0.368\,Error(Training) + 0.632 Error(Test)$

3.11.3 Performance Metrics #Performance

Consider a fraud detection example for a five-customer data set:



#DEF **Confusion matrix**: It is a *special kind of contingency table*, with two dimensions ("actual" and "predicted"), and identical sets of "classes" in both dimensions (each combination of dimension and class is a variable in the contingency table).

		Actual Status	
		Positive (Fraud)	Negative (No Fraud)
Predicted status	Positive (Fraud)	True Positive (John)	False Positive (Sophie)
	Negative (No Fraud)	False Negative (David)	True Negative (Emma, Bob)

#DEF **Classification accuracy**: The percentage of correctly classified observations: $\frac{TP+TN}{TP+FP+FN+TN}$

#DEF **Classification error**: The complement thereof and also referred to as the misclassification rate: $\frac{FP+FN}{TP+FP+FN+TN}$

#DEF **Sensitivity, recall or hit rate**: Measures how many of the fraudsters are correctly labeled as a fraudster: $\frac{TP}{TP+FN}$

#DEF **Specificity**: How many of the nonfraudsters are correctly labeled by the model as nonfraudster: $\frac{TN}{FP+TN}$

#DEF **Precision**: How many of the predicted fraudsters are actually Fraudsters: $\frac{TP}{TP+FP}$

Example:

		Actual Status	
		Positive (Fraud)	Negative (No Fraud)
Predicted status	Positive (Fraud)	True Positive (John)	False Positive (Sophie)
	Negative (No Fraud)	False Negative (David)	True Negative (Emma, Bob)

$$Sensitivity = recall = hit\,rate = \frac{TP}{TP+FN} = \frac{1}{2}$$
$$Specificity = \frac{TN}{FP+TN} = \frac{2}{3}$$
$$Precision = \frac{TP}{TP+FP} = \frac{1}{2}$$
$$FMeasure = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{1}{2}$$

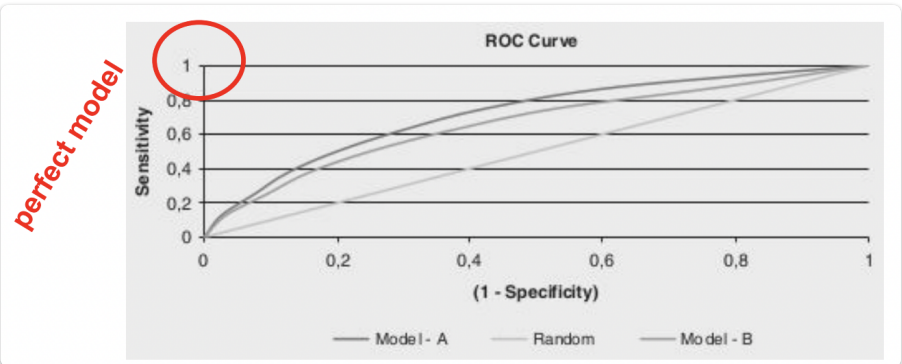
All these classification measures depend on the cut-off.
For a cut-off of 0 (1):

- The classification accuracy becomes 40% (60%).
- The error 60% (40%).
- The sensitivity 100% (0%).
- The specificity 0% (100%).
- The precision 40% (0%).
- The F-measure 0.57% (0%).

> *Given this dependence, it would be nice to have a performance measure that is independent from the cut-off.*

3.1.3.1 Receiver Operating Characteristic #ROC Curve

The receiver operating characteristic (ROC) curve then plots the sensitivity versus 1 - specificity or FPR.



3.1.3.2 Area under the ROC curve #AUC

A problem arises if the curves intersect.

AUC provides a *simple figure-of-merit for the performance*:

- The higher the AUC, the better the performance
- Bounded between 0 and 1.
- Can be interpreted as a probability.
- It represents the probability that a randomly chosen fraudster gets a higher score than a randomly chosen nonfraudster.

> ***A good classifier should have an ROC above the diagonal and AUC bigger than 50%.***

3.11.3.3 Other Performances Metrics

- **Interpretability**:
  - *White-box* : Linear and logistic regression, and decision trees.
  - *Black-box*: Neural networks, SVMs, and ensemble methods.
- **Justifiability**:
  - Verifies to what extent the relationships modeled are in line with expectations:
    - *Verifying the univariate impact of a variable on the model's output*.
- **Operational efficiency**: ease with which one can implement, use, and monitor the final model:
  - To be able to *quickly evaluate the fraud model*.
  - Linear and rule based models are *easy to implement*.

3.11.4 Developing predictive models for skewed data sets

Fraud-detection data sets often have a **very skew target** *class distribution frauds are <= 1%*.

This creates problems for the analytical techniques:

- **Flooded by nonfraudulent observations** and thus *tend toward classifying every observation as nonfraudulent*.

> *Recommended to increase the number of fraudulent observations or their **weight**, such that the analytical techniques can pay better attention to them.*

Increase the number (and variability) of frauds is by:

1. **Increasing the time horizon** for prediction:
   - Instead of predicting fraud with a six-month forward-looking time horizon, a 12-month time horizon.
2. **Sampling every frauds twice** (or more).
   - *We predict fraud with a one-year forward-looking time horizon using information from a one year backward looking time horizon*.

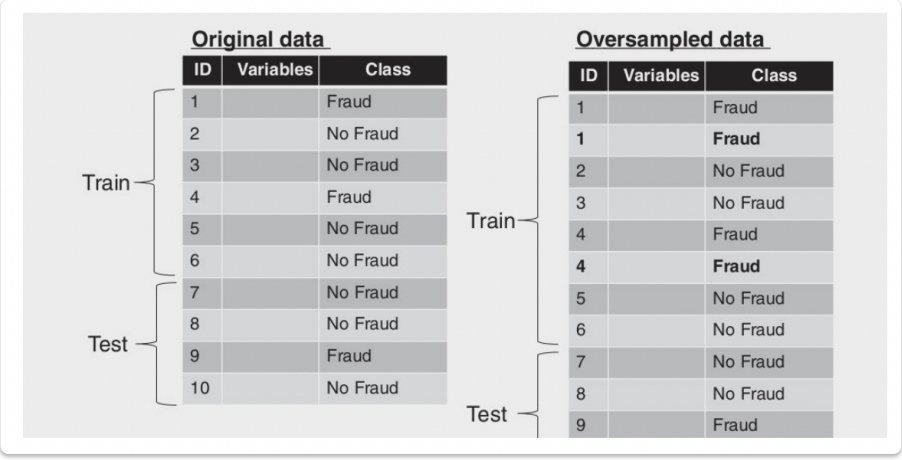
> - By shifting the observation point earlier or later, the same fraudulent observation can be sampled twice.
> - The variables collected will be similar but not perfectly the same, since *they are measured on a different time frame*.

Finding the optimal number is subject to a trial-and-error exercise:

- Depending on the skewness of the target.

3.11.4.1 Oversampling #Oversampling

#DEF ***Replicate frauds*** **two or more times so as to make the distribution less skew.**

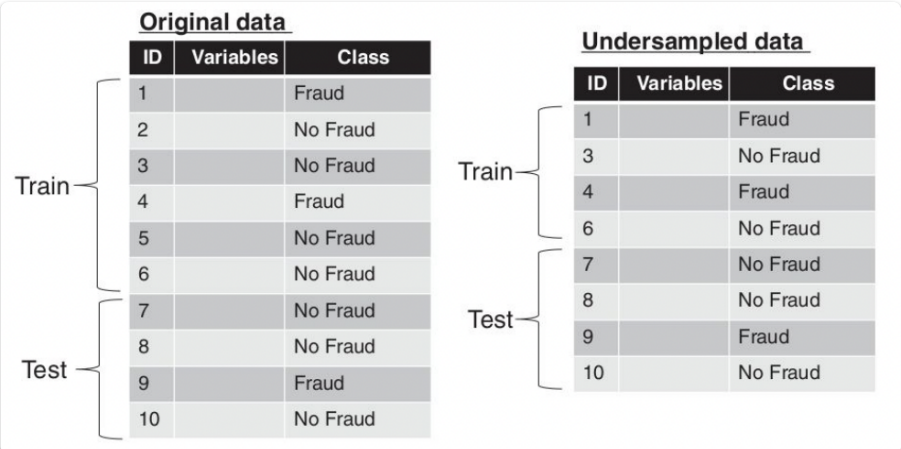


3.11.4.2 Undersampling #Undersampling

#DEF ***Remove nonfrauds*** **two or more times so as to make the distribution less skew.**

Based on business experience whereby obviously legitimate observations are removed:

- Low-value transactions or inactive accounts.



3.11.4.3 Under and oversampling

Under and oversampling can also be combined.
Undersampling usually results in better classifiers than oversampling.

> *They should be conducted on the training data and **not** on the test data to give an unbiased view on model performance.*

What is the optimal nonfraud/fraud odds that should be aimed for by doing under or oversampling?

Working toward a **balanced sample with the same number of fraudsters and nonfraudsters**, it may biases the performances.
It is **recommended to stay as close as possible to the original class distribution** to *avoid unnecessary bias*.

3.11.4.4 Practical approach to determine the optimal class distribution

1. An analytical model is built on the original data set with the skew class distribution (e.g., 95%/5%).
2. The *AUC of this model is recorded* (possibly on an independent validation data set).
3. Over or undersampling is used to change the class distribution by 5 percent (e.g., 90%/10%).
4. The *AUC of the model is recorded*.
5. *Subsequent models are built on samples* of 85%/15%, 80%/20%, 75%/25%, recording their AUC.
6. **Once the AUC starts to stagnate** (or drop), *the procedure stops and the optimal odds ratio has been found.*

3.11.4.5 Synthetic Minority Oversampling Technique #SMOTE

Rather than replicating the minority observations, **SMOTE** works by *creating synthetic observations based on the existing minority observations*.

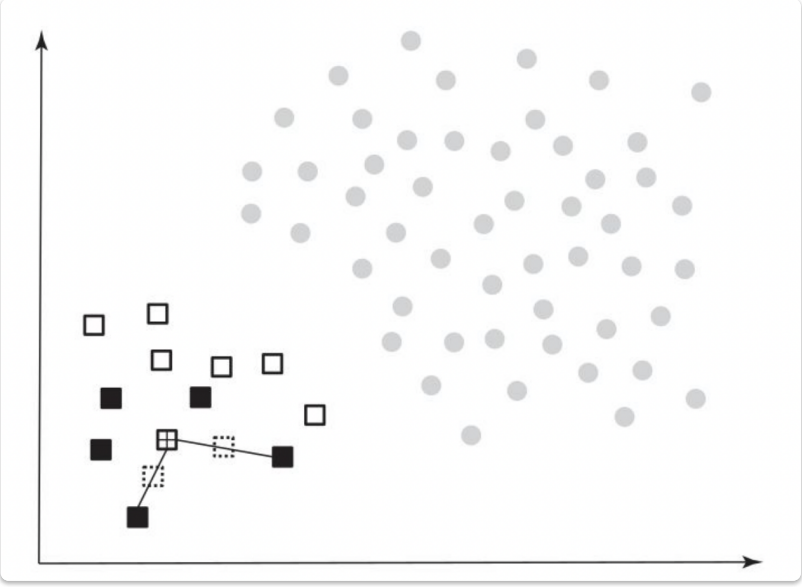
For each minority class observation, it *calculates the k nearest neighbors*:

- Depending on the amount of oversampling needed, one or more of the k-nearest neighbors are selected to create the synthetic examples.

Randomly create two synthetic examples along the line connecting the observation under investigation with the two random nearest neighbors.

SMOTE then combines the synthetic oversampling of the minority class with under-sampling the majority class.

> *SMOTE usually works better than either under- or oversampling.*



3.11.4.6 Cost-sensitive Learning

Assigns higher misclassification costs to the minority class:

		Predicted Class	
		Positive	Negative
Actual class	Positive	C(+,+)	C(–,+)
	Negative	C(+,–)	C(–,–)

Usually $C(+,+) = C(-,-) = 0$, and $C(-,+) > C(+,-)$.
Minimizing the misclassification cost during classifier learning.

$Total\,cost = C(-,+) \times FN + C(+,-) \times FP$