Performance Metrics

The simplest performance metric is the *model error* defined as the number of mistakes the model makes on a data set divided by the number of observations in the data set,

$$\text{err} = \frac{\text{number of mistakes}}{\text{total number of observations}}.$$

The Model Error

In order to define the model error formally we introduce the *0-1 loss function*. This function compares the output of a model for a particular observation with the label of this observation. If the model commits a prediction error on this observation then the loss function returns a 1, otherwise it returns a 0.

Formally, let $(\overline{x}, y) \in D$ be an observation where $D \subseteq \mathbb{R}^n \times \{+1, -1\}$ and let $\hat{f} : \mathbb{R}^n \to \{+1, -1\}$ be a model, then we define the 0-1 loss function $\mathcal{L} : \{+1, -1\} \times \{+1, -1\} \to \{0, 1\}$ as,

$$\mathcal{L}\left(y, \hat{f}(\overline{x})\right) = \begin{cases} 0 \text{ if } y = \hat{f}(\overline{x}), \\ 1 \text{ if } y \neq \hat{f}(\overline{x}). \end{cases}$$

Let $D = \{(\overline{x}_1, y_1), \ldots, (\overline{x}_l, y_l)\} \subset \mathbb{R}^n \times \{+1, -1\}$, let the model $\hat{f}$ and the loss function $\mathcal{L}$ be as defined above, then we can write the model error as,

$$\text{err}_D[\hat{f}] = \frac{1}{l} \sum_{i=1}^{l} \mathcal{L}\left(y_i, \hat{f}(\overline{x}_i)\right),$$

where $(\overline{x}_i, y_i) \in D$.

The model error is the *average loss* of a model over a data set.

Model Accuracy

We can also characterize the performance of a model in terms of its *accuracy*,

$$\text{acc} = \frac{\text{number of correct predictions}}{\text{total number of observations}}.$$

Again, we can use the 0-1 loss function to define this metric more concisely,

$$\begin{aligned}
\text{acc}_D[\hat{f}] &= \frac{1}{l}\left(l - \sum_{i=1}^{l} \mathcal{L}\left(y_i, \hat{f}(\overline{x}_i)\right)\right) \\
&= 1 - \frac{1}{l}\sum_{i=1}^{l} \mathcal{L}\left(y_i, \hat{f}(\overline{x}_i)\right) \\
&= 1 - \text{err}_D[\hat{f}].
\end{aligned}$$

$$\boxed{\text{acc}_D[\hat{f}] = 1 - \text{err}_D[\hat{f}]}$$

Example

As an example of the above metrics, consider a model $\hat{g}$ which commits 5 prediction errors when applied to a data set Q of length 100. We can compute the error as,

$$\text{err}_Q[\hat{g}] = \frac{1}{100}(5) = 0.05.$$

We can compute the accuracy of the model as,

$$\text{acc}_Q[\hat{g}] = 1 - \text{err}_Q[\hat{g}] = 1 - 0.05 = 0.95.$$

Model Errors

Let $(\overline{x}, y) \in \mathbb{R}^n \times \{+1, -1\}$ be an observation and let $\hat{f} : \mathbb{R}^n \rightarrow \{+1, -1\}$ be a model, then we have the following four possibilities when the model is applied to the observation,

$$\hat{f}(\overline{x}) = \begin{cases} +1 \text{ if } y = +1, \text{ called the } \textit{true positive} \\ -1 \text{ if } y = +1, \text{ called the } \textit{false negative} \\ +1 \text{ if } y = -1, \text{ called the } \textit{false positive} \\ -1 \text{ if } y = -1, \text{ called the } \textit{true negative} \end{cases}$$

This means that models can commit two types of model errors.

Under certain circumstances it is important to distinguish these types of errors when evaluating a model.

We use a *confusion matrix* to report these errors in an effective manner.

The Confusion Matrix

A confusion matrix for a binary classification model is a 2×2 table that displays the observed labels against the predicted labels of a data set.

Observed (y) \ Predicted ($\hat{y}$)	+1	-1
+1	True Positive (TP)	False Negative (FN)
-1	False Positive (FP)	True Negative (TN)

One way to visualize the confusion matrix is to consider that applying a model $\hat{f}$ to an observation $(\overline{x}, y)$ will give us two labels. The first label y is due to the observation and the second label $\hat{y} = \hat{f}(\overline{x})$ is due to the prediction of the model. Therefore, an observation with the label pair $(y, \hat{y})$ will be mapped onto a confusion matrix as follows,

$$
\begin{array}{ccc}
(+1,+1) & \mapsto & \text{TP} \\
(-1,+1) & \mapsto & \text{FP} \\
(+1,-1) & \mapsto & \text{FN} \\
(-1,-1) & \mapsto & \text{TN}
\end{array}
$$

The Confusion Matrix

Example:

Observed $\backslash$ Predicted	+1	-1
+1	95	7
-1	4	94

A confusion matrix of a model applied to a set of 200 observations. On this set of observations the model commits 7 false negative errors and 4 false positive errors in addition to the 95 true positive and 94 true negative predictions.

Example

Wisconsin Breast Cancer Dataset:

```
> library(e1071)
> wdbc.df <- read.csv("wdbc.csv")
> svm.model <- svm(Diagnosis ~ .,
                   data=wdbc.df,
                   type="C-classification",
                   kernel="linear",
                   cost=1)
> predict <- fitted(svm.model)
> cm <- table(wdbc.df$Diagnosis,predict)
> cm
   predict
      B   M
  B 355   2
  M   5 207
> err <- (cm[1,2] + cm[2,1])/length(predict) * 100
> err
[1] 1.230228
```

Model Evaluation

Model evaluation is the process of finding an optimal model for the problem at hand.

You guessed it, we are talking about optimization!

Let,

$$D = \{(\overline{x}_1, y_1), \ldots, (\overline{x}_l, y_l)\} \subset \mathbb{R}^n \times \{+1, -1\}$$

be our training data, then we can write our parameterized model as,

$$\hat{f}_D[k, \lambda, C](\overline{x}) = \text{sign}\left(\sum_{i=1}^{l} \alpha_{C,i} y_i k[\lambda](\overline{x}_i, \overline{x}) - b\right),$$

where $(\overline{x}_i, y_i) \in D$.

With this we can write our model error as,

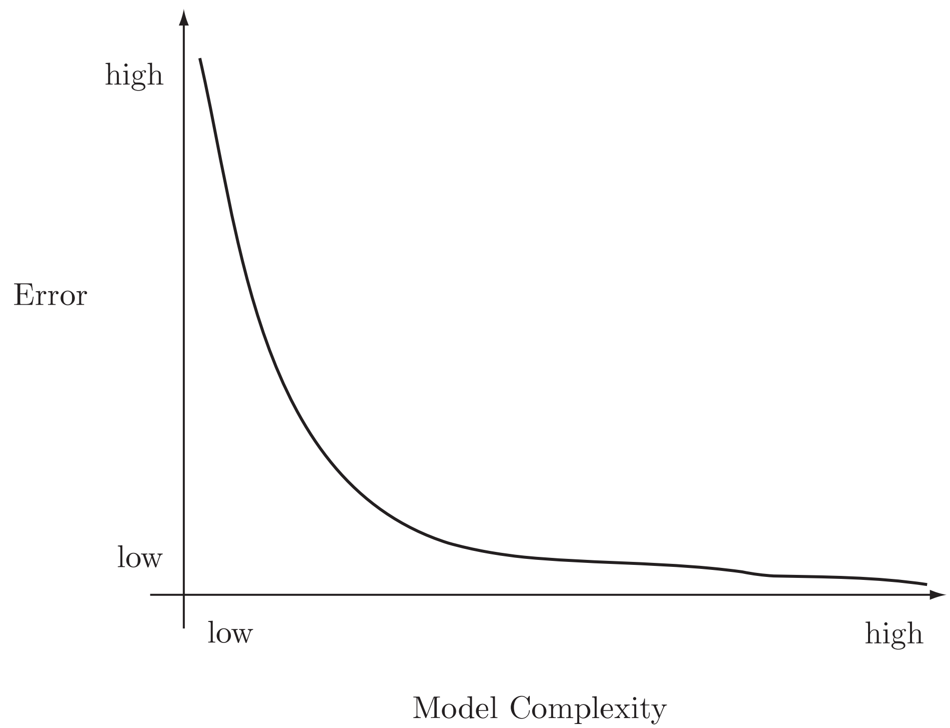
$$\text{err}_D\left[\hat{f}_D[k, \lambda, C]\right] = \frac{1}{l}\sum_{i=1}^{l} \mathcal{L}\left(y_i, \hat{f}_D[k, \lambda, C](\overline{x}_i)\right),$$

where $(\overline{x}_i, y_i) \in D$.

Training Error

The optimal training error is defined as,

$$\min_{k,\lambda,C} \text{err}_{D}\left[\hat{f}_{D}[k,\lambda,C]\right] = \min_{k,\lambda,C} \frac{1}{l}\sum_{i=1}^{l} \mathcal{L}\left(y_i, \hat{f}_D[k,\lambda,C](\overline{x}_i)\right).$$



Training Error

Observation:

The problem here is that we can always find a set of model parameters that make the model complex enough to drive the training error down to zero.

The training error as a model evaluation criterion is overly optimistic.

The Hold-Out Method

Here we split the training data D into a *training set* and a *testing set* P and Q, respectively, such that,

$$D = P \cup Q \text{ and } P \cap Q = \emptyset.$$

The *optimal training error* is then computed as the optimization problem,

$$\min_{k,\lambda,C} \mathrm{err}_{P} \left[\hat{f}_{P}[k,\lambda,C] \right] = \mathrm{err}_P \left[\hat{f}_P[k^\bullet, \lambda^\bullet, C^\bullet] \right].$$

Here, the optimal training error is obtained with model $\hat{f}_P[k^\bullet, \lambda^\bullet, C^\bullet]$.

The *optimal test error* is computed as an optimization using Q as the test set,

$$\min_{k,\lambda,C} \mathrm{err}_{Q} \left[\hat{f}_{P}[k,\lambda,C] \right] = \mathrm{err}_Q \left[\hat{f}_P[k^*, \lambda^*, C^*] \right].$$

The optimal test error is achieved by some model $\hat{f}_P[k^*, \lambda^*, C^*]$.

The Hold-Out Method