Introduction to Machine Learning

Evaluation: Simple Metrics for Regression and Classification

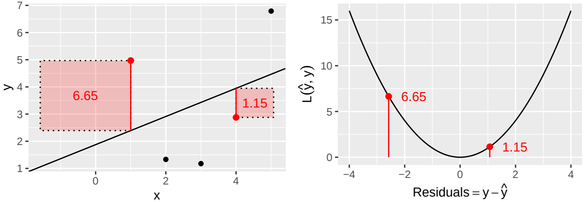
Department of Statistics – LMU Munich

REGRESSION: MSE

The **Mean Squared Error** compares the mean of the squared distances between the target variable y and the predicted target $\hat{y}$.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - \hat{y}^{(i)})^2 \in [0; \infty]$$

Single observations with a large prediction error heavily influence the **MSE**, as they enter quadratically.



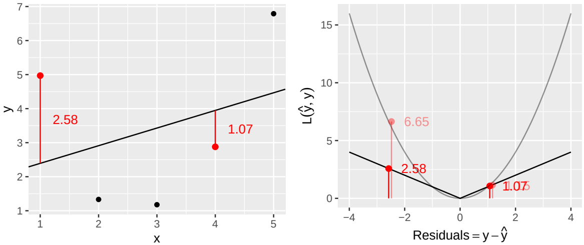
We could also sum the errors up (SSE), or take the root (RMSE) to bring the measurement back to the original scale of the outcome.

REGRESSION: MAE

A more robust (but not neccessarily better) way to compute a performance measure is the **Mean Absolute Error**:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y^{(i)} - \hat{y}^{(i)}| \in [0; \infty]$$

Less influenced by large errors and maybe more intuitive than the MSE.



Instead of averaging we might also consider the median for even more robustness.

REGRESSION: R^2

Another well known measure from statistics is R^2.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y^{(i)} - \hat{y}^{(i)})^2}{\sum_{i=1}^{n}(y^{(i)} - \bar{y})^2} = 1 - \frac{SSE_{LinMod}}{SSE_{Intercept}}$$

- Usually introduced as *fraction of variance explained* by the model
- Much simpler explanation: It compares the SSE of a constant model (baseline) with a more complex model (LM), on some data, usually the same as used for model fitting
- $R^2 = 1$ implies: all residuals are 0, we predict perfectly, $R^2 = 0$ implies we predict as badly as a naked constant
- If measured on the training data, $R^2 \in [0; 1]$, as the LM must be at least as good as the constant, and both SSEs are non-negative
- On other data it could even be negative, as there is no guarantee that the LM generalizes better than a constant (overfitting possible)

GENERALIZED R^2 FOR ML

A simple generalization of R^2 for ML seems to be:

$$1 - \frac{Loss_{ComplexModel}}{Loss_{SimplerModel}}$$

- This introduces a general measure of comparison between a simpler baseline, and a more complex model considered as an alternative
- This works for arbitrary measures (not only SSE), for arbitrary models, on any data set of interest
- E.g. model vs constant, LM vs. non-linear model, tree vs. forest, model without some features vs. model with them included
- In ML we would rather use that metric on a holdout-test set, there is no reason not to do that
- I do not see this being used or known very often, and my terminology (generalized R^2) is non-standard

LABELS: ACCURACY / MCE

The misclassification error rate (MCE) simply counts the number of incorrect predictions and presents them as a rate, accuracy is defined in a similar fashion for correct classifications

$$MCE = \frac{1}{n}\sum_{i=1}^{n}[y^{(i)} \neq \hat{y}^{(i)}] \in [0;1]$$

$$ACC = \frac{1}{n}\sum_{i=1}^{n}[y^{(i)} = \hat{y}^{(i)}] \in [0;1]$$

- If the data set is small this can be quite a brittle measure
- The MCE says nothing about how good or skewed predicted probabilities are
- Errors on all classes are weighed equally, that is often inappropriate

LABELS: CONFUSION MATRIX

Much better than simply reducing prediction errors to a simple number we can tabulate them in a confusion matrix, tabulating true classes in rows and predicted classes in columns. We can nicely see class sizes (predicted and true) and where errors occur.

```
##            setosa versicolor virginica -err.- -n-
## setosa         50          0         0      0  50
## versicolor      0         46         4      4  50
## virginica       0          4        46      4  50
## -err.-          0          4         4      8  NA
## -n-            50         50        50     NA 150
```

LABELS: COSTS

We can also assign different costs to different errors via a cost matrix.

$$Costs = \frac{1}{n}\sum_{i=1}^{n} C[y^{(i)}, \hat{y}^{(i)}]$$

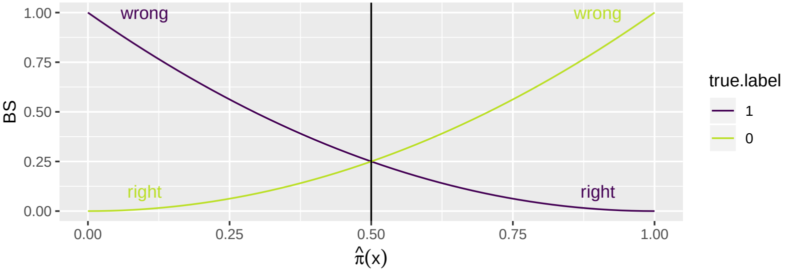
```
## Confusion matrix
##             predicted
## true         setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         46         4
##   virginica       0          4        46
## Cost matrix C
##             predicted
## true         setosa versicolor virginica
##   setosa          0          1         1
##   versicolor      2          0         5
##   virginica       1          1         0
```

- Here, we penalize errors on class *versicolor* more heavily
- $Costs = (3 \cdot 5 + 4 \cdot 1)/150$

PROBABILITIES: BRIER SCORE

Measures squared distances of probabilities from the true class labels:

$$BS1 = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{\pi}(\mathbf{x}^{(i)}) - y^{(i)}\right)^2$$



- Usual definition for binary case, $y^{(i)}$ must be coded as 0 and 1.
- Fancy name for MSE on probabilities

PROBABILITIES: BRIER SCORE

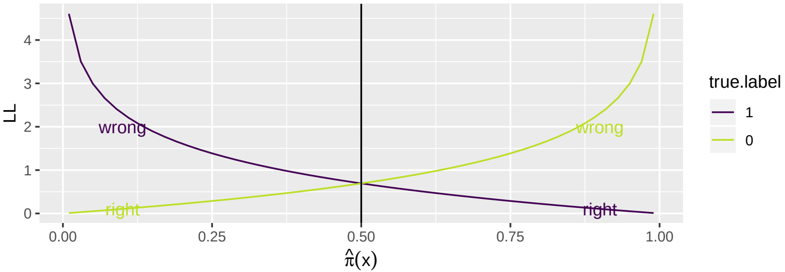
$$BS2 = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{g} \left(\hat{\pi}_k(\mathbf{x}^{(i)}) - o_k^{(i)} \right)^2$$

- Original one by Brier, works also for multiple classes
- $o_k^{(i)} = [\hat{y}^{(i)} = k]$ is a 0-1-one-hot coding for labels
- For the binary case, BS2 is twice as large as BS1, because in BS2 we sum the squared difference for each observation regarding class 0 AND class 1, not only the true class.

PROBABILITIES: LOG-LOSS

Logistic regression loss function, a.k.a. Bernoulli or binomial loss, $y^{(i)}$ coded as 0 and 1.

$$LL = \frac{1}{n}\sum_{i=1}^{n}\left(-y^{(i)}\log(\hat{\pi}(\mathbf{x}^{(i)})) - (1-y^{(i)})\log(1-\hat{\pi}(\mathbf{x}^{(i)}))\right)$$



- Optimal value is 0, "confidently wrong" is penalized heavily
- Multiclass version: $LL = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{g} o_k^{(i)}\log(\hat{\pi}_k(\mathbf{x}^{(i)}))$