Machine Learning 2018 – Performance Metrics

Huy Ngoc Pham

December 29, 2018

Outline

1. Performance Metrics
   - Regression
   - Classification
2. Evaluation procedures
   - Cross-validation
   - Bootstrap

Roadmap Review

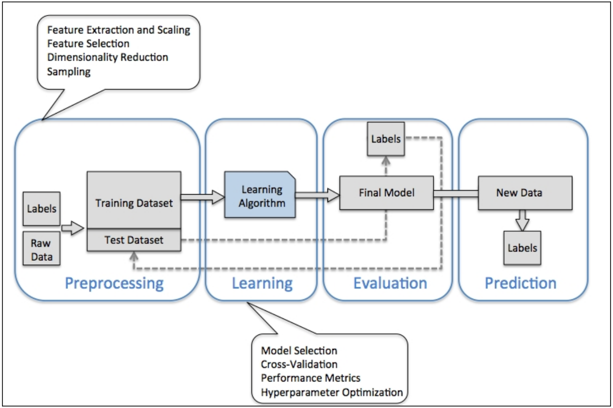


Figure: A roadmap for building machine learning systems (O'Reilly)

Type of Performance Metrics

1. Regression
   - Correlation coefficients
   - Mean Square Error / Root Mean Square Error
   - Mean Absolute Error
   - Residuals
2. Classification
   - Precision
   - Recall/Sensitivity
   - Specificity
   - Accuracy
   - F1-Score

Review of Linear Regression I

Linear model for regression is a linear combination of the input variables.

Formula

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D = w_0 + \sum_{j=1}^{D} w_j x_j$$

Loss function

$$L(w) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

Ordinary Least Squares

Our goal is to find $\hat{\mathbf{w}}$:

$$\hat{\mathbf{w}} = argmin_{\mathbf{w}} L(\mathbf{w}) = argmin_{\mathbf{w}} \left(\frac{1}{2} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 \right)$$

Review of Linear Regression II

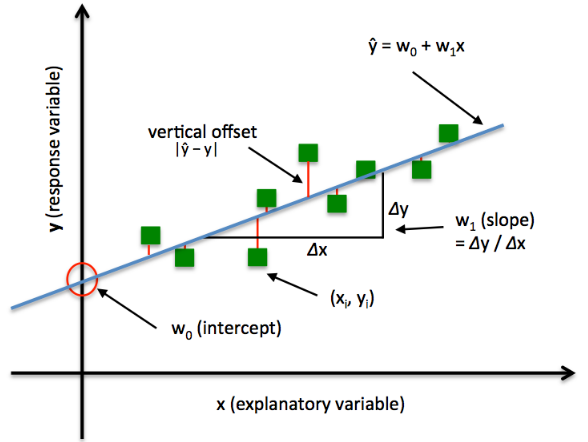


Figure: Linear Regression (Credit: mlxtend)

Performance Metrics for Regression

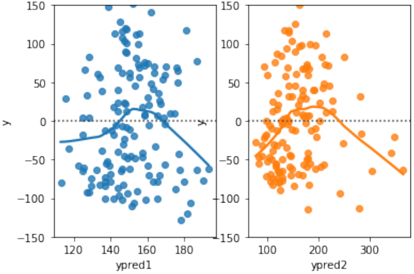
- Mean Square Error:
- Correlation coefficients
- Mean Absolute Error
- Residuals

Residuals

Residual is the difference between true and predicted values:

$$r_i = y_i - \hat{y}_i$$

Residual plot is the scatter-plot of fitted values ($\hat{y}_i$) against residuals (r_i).



Residuals

- Assess visually regression models.
- Diagnostics of regression models:
  - The regression function is nonlinear?
  - Unbalanced data?
  - Outliers?
  - ...

More info: Interpreting residual plots to improve your regression

Mean Square Error and the Brothers

Mean squared error (MSE) is the average of the squares of the errors:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- MSE is a **loss function**,
- MSE is a **measure of the quality** of an estimator—it is always non-negative, and values closer to zero are better,
- MSE will be small if the predicted responses are very close to the true responses, and will be large if for some of the observations, the predicted and true responses differ substantially.
- To compare among models, MSE on the testing dataset should be considered.

Mean Square Error and the Brothers

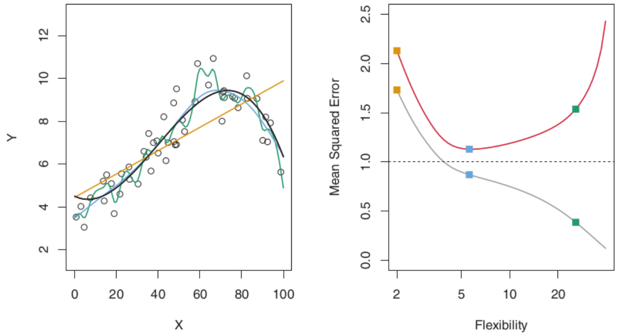


Figure: **Left**: Data simulated from f-black line. Three models: the LR line (orange curve), and two smoothing spline fits (blue and green curves). **Right**: **Training MSE** (grey curve), **test MSE** (red curve), and minimum possible test MSE over all methods (dashed line). Squares represent the MSEs. (Credit: ISL book)

Mean Square Error and the Brothers

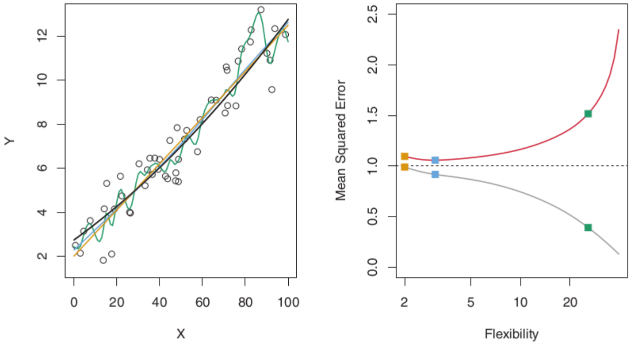


Figure: Using a different true f that is much closer to linear. (Credit: ISL book)

Mean Square Error and the Brothers

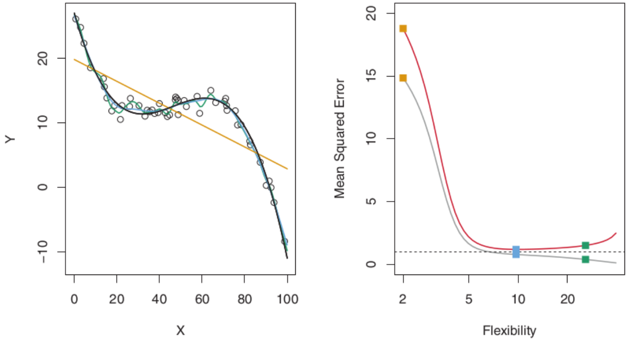


Figure: using a different f that is far from linear.
(Credit: ISL book)

Mean Square Error and the **Brothers**

- Root Mean Square Error (RMSE):

$$MSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

The Bias-Variance Trade-off

By Mathematics, it is easy to prove that for a given point x_0:

$$E\left[(y_0 - \hat{y}_0)^2\right] = \underbrace{Var[\hat{y}_0]}_{Variance} + \underbrace{(Bias[\hat{y}_0])^2}_{(Bias)^2} + \underbrace{Var(\epsilon)}_{Irreducible\ \ error}$$

- **Variance** refers to the amount by which predicted value would change if we estimated it using a different training data set.
- **Bias** refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model.
- As a general rule, as we use **more flexible** $\sim$ methods, **the variance will increase** $\uparrow$ and **the bias will decrease** $\downarrow$.

The Bias-Variance Trade-off

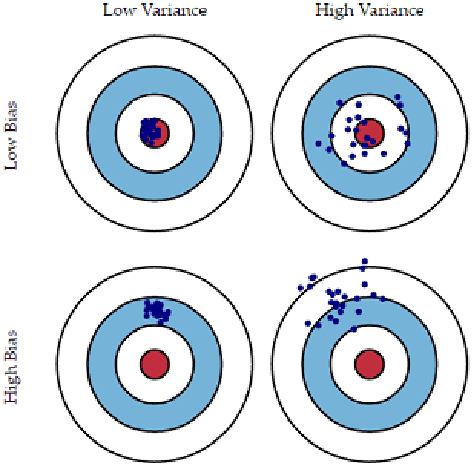


Figure: Illustation of Bias and Variance

The Bias-Variance Trade-off

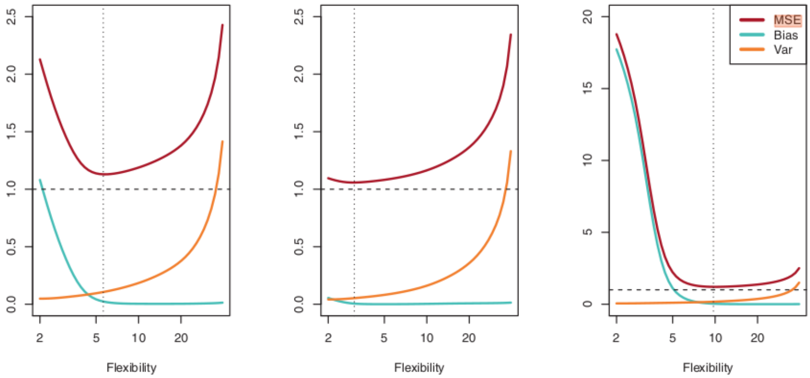


Figure: Squared bias (blue curve), variance (orange curve), Var(ϵ) (dashed line), and test MSE (red curve). The vertical dotted line indicates the flexibility level corresponding to the smallest test MSE.

Correlation coefficient

1. The Pearson product-moment correlation coefficient
2. Rank correlation
    - Spearman's rank correlation coefficient
    - Kendall tau rank correlation coefficient

The Pearson product-moment correlation coefficient

The Pearson product-moment correlation coefficient measures linear association:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} = \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}}{\sqrt{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}\sqrt{\sum_{i=1}^{n} y_i^2 - n\bar{y}^2}}$$

How **Person's r** is interpreted:

- Ranges from -1 to $+1$
- -1 means Perfectly negative linear correlation
- $+1$ means Perfectly positive linear correlation
- 0 means No linear correlation

The Pearson product-moment correlation coefficient

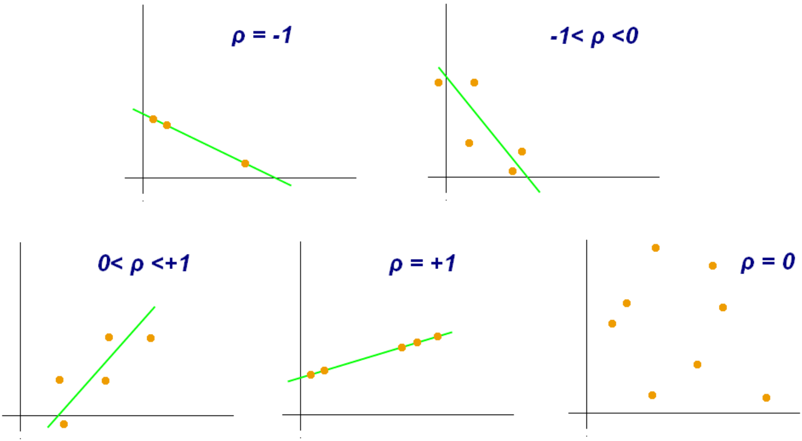


Figure: Illustration of some cases of the Pearson's r. (Credit: Wikipedia)

The Pearson product-moment correlation coefficient

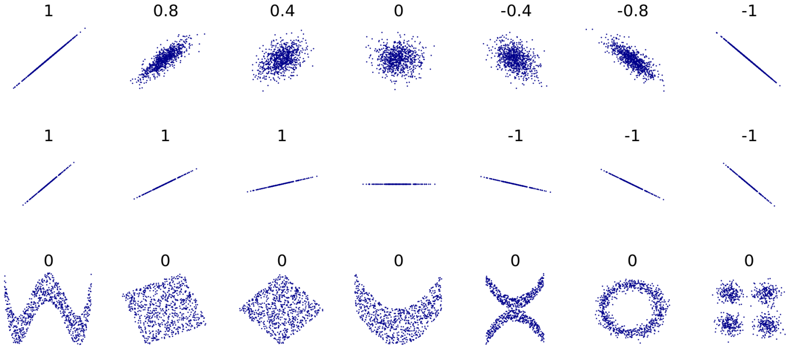


Figure: The Pearson's r is a measure of correlation, not accuracy. (Credit: Wikipedia)

The Pearson product-moment correlation coefficient

The measure r^2 or R^2 is called **coefficient of determination**.

$$R^2 = \frac{TSS - RSS}{TTS}$$

where

- $TTS = \sum (y_i - \bar{y})^2$ is *total sum of squares*, measures the total variance in the response Y.
- $RSS = \sum (y_i - \hat{y}_i)^2$ is *residual sum of squares*, measures the amount of variability that is left unexplained after performing the regression.
- $TSS - RSS$ measures the amount of variability in the response that is explained by regression model.

R^2 measures **the proportion of variability** in response that can be *explained using predictors*.

Algorithms Classification

- Logistics Regression
- Support Vector Machine
- Decision Tress / Random Forest
- Neural Networks
- Nearest Neighbor

Type of Classification Problems

- Binary Classification: Distinguish between 2 classes
- Multiclass Classification: More than 2 classes
- Multilabel Classification: Output muliple classes for each instance
- Multioutput-multiclass Classification (aka Multi-task classification): Generalization of multilabel classification where each label can be multiclass.

Confusion matrix

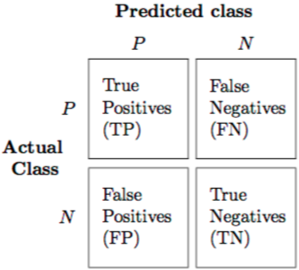


Figure: Confusion matrix.(Credit: mlxtend)

TP = True Positive = Hit.
TN = True Negative = Correct Rejection.
FP = False Positive = False alarm = Type I error.
FN = False Negative = Miss = Type II error

Derivations from a confusion matrix

- **Precision** or Positive Predictive Value (PPV):

$$Precision = \frac{TP}{TP + FP}$$

- **Recall**, Sensitivity, Hit rate, or True Positive rate (TPR) :

$$Recall = \frac{TP}{P} = \frac{TP}{TP + FN}$$

- **Specificity**, Selectivity or True Negative rate (TNR):

$$Specificity = \frac{TN}{N} = \frac{TN}{TN + FP}$$

- **Accuracy**:

$$Accuracy = \frac{TP + TN}{P + N} = \frac{TP + TN}{TN + TN + FP + FN}$$

Why not Accuracy? Accuracy paradox

The **accuracy paradox** *is the paradoxical finding that accuracy is not a good metric for predictive models when classifying in predictive analytics. This is because a simple model may have a high level of accuracy but be too crude to be useful.*

–Wikipedia

Precision & Recall

- **Recall** tells us how confident we can be that all the instances with the positive target level have been found by the model.
- **Precision** captures how often, when a model makes a positive prediction, this prediction turns out to be correct. Precision tells us how confident we can be that an instance predicted to have the positive target level actually has the positive target level.
- Both precision and recall can assume values in the range [0,1], and **higher values** in both cases indicate **better model performance**.

F1 Score

Definition

The **F1 Score** is the harmonic mean of precision and recall and is defined as

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

- Why F1 Score? Less sensitive to large outliers $\rightarrow$ highlight shortcomings rather than hide them
- Ranges (0, 1] and higher values indicate better performance.

Precision-Recall Tradeoff

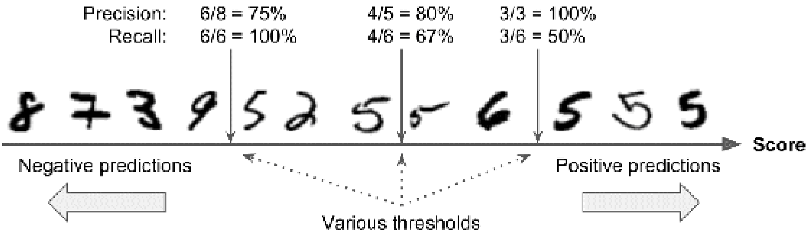
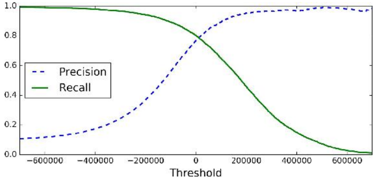
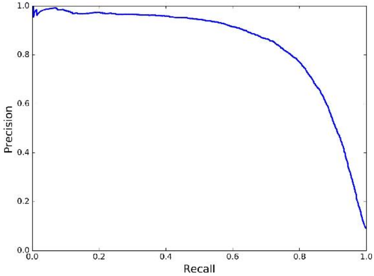


Figure: Threshold and Precision-Recall Tradeoff

Precision-Recall Tradeoff



(a) Threshold and Precision-Recall Tradeoff



(b) Recall vs Precision

Figure: Precision-Recall Tradeoff

Precision-Recall Tradeoff

If someone says "let's reach 99% precision,"
You should ask, "at what recall?"

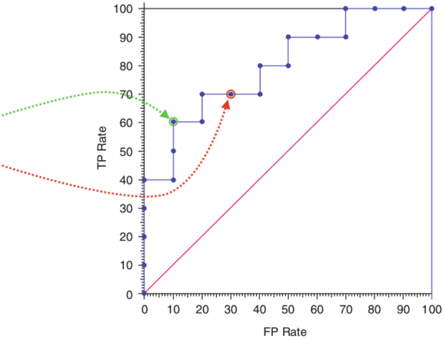
–Aurélien Géron

ROC curve = Receiver operating characteristic curve

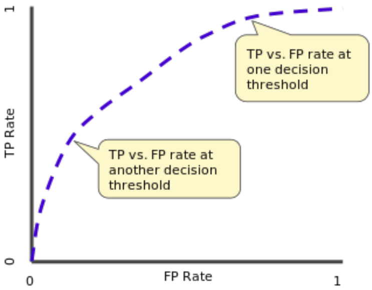
Definition

By joining all possible operating points of a scoring classifier on the ROC plane with line segments, we receive a visual representation of its performance independent of the cutoff value. This is called **the ROC curve**.

Class	Score
+	0.98
+	0.93
+	0.87
+	0.84
–	0.79
+	0.73
+	0.67
–	0.62
+	0.57
–	0.54
–	0.48
+	0.43
–	0.37
+	0.34
–	0.28
–	0.24
+	0.18
–	0.12
–	0.09
–	0.03



ROC curve = Receiver operating characteristic curve



TP rate = Sensitivity
FP rate = 1 - Specificity
(0,1): The perfect model with all **correct** classification.
(1,0): The worst model with all **incorrect** classification.
(0,0): Always predicts class 0.
(1,1): Always predicts class 1.

AUC = Area Under the ROC Curve

Definition

Area Under the ROC Curve (AUC) measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1).

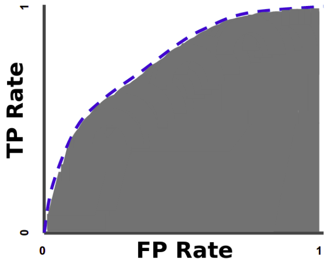


Figure: Area Under the ROC Curve. Credit: Google Developer

AUC = Area Under the ROC Curve

Characteristics of AUC:

- AUC is **scale-invariant**. It measures how well predictions are ranked, rather than their absolute values.
- AUC is **classification-threshold-invariant**. It measures the quality of the model's predictions irrespective of what classification threshold is chosen.

How AUC is interpreted:

- AUC is as the probability that the model ranks a random positive example more highly than a random negative example.
- AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0.

ROC & AUC

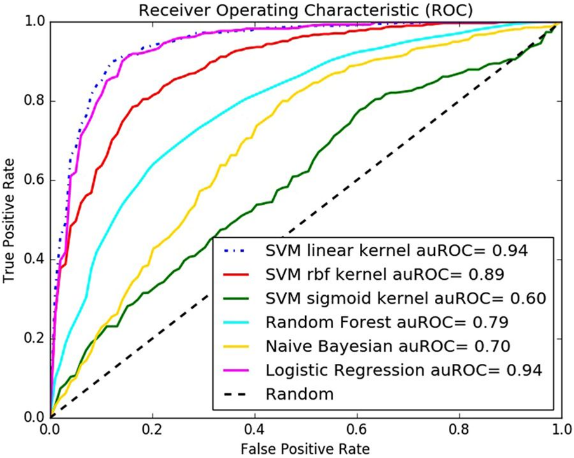
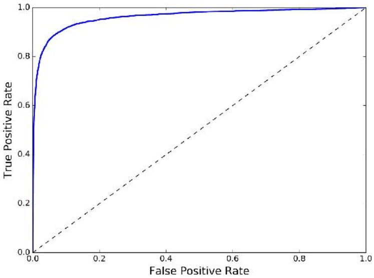
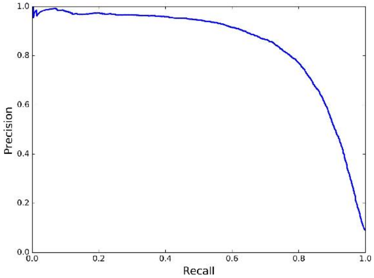


Figure: Model Benchmarks. Credit: Nature

ROC & Precision-Recall Curve



(a) ROC Curve (AUC ≈ 0.97)



(b) PR Curve

Figure: (a) or (b) or both?

Gini Index

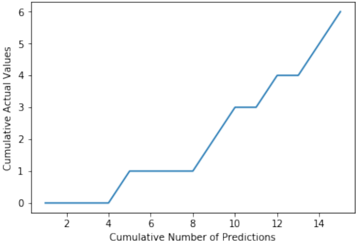
Definition

The Gini coefficient is an empirical measure of classification performance based on the area under an ROC curve (AUC).

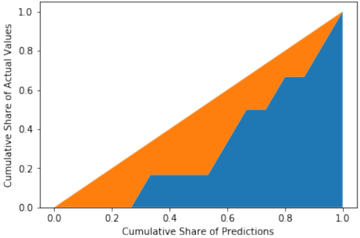
$$\text{Gini Index} = 2 \times \text{AUC} - 1$$

- Ranges [-1, 1], and higher values indicate better model.
- Used in financial modeling scenarios such as credit scoring

Gini Index



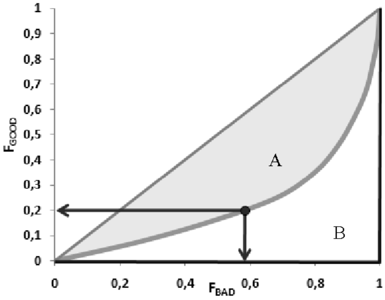
(a) Lorenz Curve



(b) Random guessing and Model

Figure: Gini Index $= \frac{\text{Orange Area}}{\text{Blue Area}}$. Credit: batzner

Gini Index in Credit Scoring



$GI = \frac{A}{B}$

$GI = 1 \rightarrow$ Scoring function perfectly separates

$GI = 0 \rightarrow$ Scoring function assigns ramdomly

Kolmogorov-Smirnov statistic

Definition

The **Kolmogorov-Smirnov statistic** (K-S statistic) is the performance measure that captures the separation between the distribution of prediction scores for the different target levels in a classification problem.

How to calculate K-S statistic:

- Determine the Cumulative probability distributions of the prediction scores for each classes.
- Plot the CP on the Kolmogorov-Smirnov chart (K-S chart)
- The K-S statistic is calculated by determining the maximum difference between CP.

Kolmogorov-Smirnov statistic

- Cumulative probability distribution:

$$CP(positive, ps) = \frac{\text{no. positive test instances with score } \leq ps}{\text{no. positive test instances}}$$

$$CP(negative, ps) = \frac{\text{no. negative test instances with score } \leq ps}{\text{no. negative test instances}}$$

- K-S statistic:

$$\text{K-S} = \max_{ps}(CP(positive, ps) - CP(negative, ps))$$

Kolmogorov-Smirnov statistic

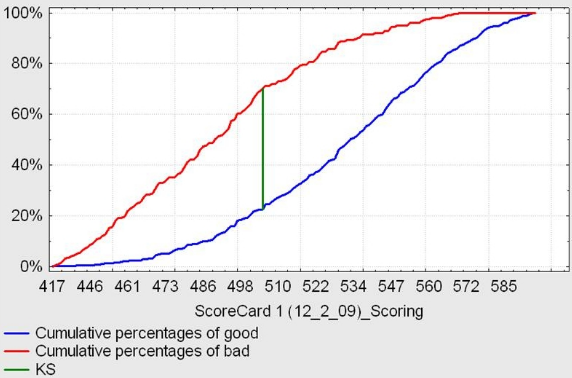


Figure: Kolmogorow-Smirnow chart. Credit: Ong Xuan Hong

Multiclass Classification

- Cohen's kappa
- Confusion matrix
- Hinge loss
- Matthews correlation coefficient (MCC)

Cohen's kappa

Cohen's kappa

Cohen's kappa is a statistic that measures inter-annotator agreement, expresses the level of agreement between two annotators on a classification problem.

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

where:

- p_o: the observed accuracy
- p_e: the expected accuracy based on the marginal totals of the confusion matrix.

Cohen's kappa

- Ranges between -1 and 1
- $\kappa = 0$ means means there is no agreement between the observed and predicted classes
- $\kappa = 0$ means perfect concordance of the model prediction and the observed classes.
- Negative values indicate that the prediction is in the opposite direction of the truth
- When the class distributions are equivalent, overall accuracy and Kappa are proportional.
- Depending on the context, Kappa values within 0.30 to 0.50 indicate reasonable agreement.

Confusion Matrix

Confusion matrix C is such that $C_{i,j}$ is equal to the number of observations known to be in group but predicted to be in group.

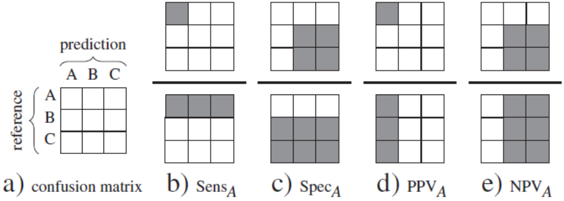


Figure: Confusion Matrix for Multiclass classification. Credit: softclassval

Confusion Matrix

	Predicted			
Actual		A	B	C
	A	$\mathbf{TP_A}$	E_{AB}	E_{AC}
	B	E_{BA}	$\mathbf{TP_B}$	E_{BC}
	C	E_{CA}	E_{CB}	$\mathbf{TP_C}$

		Predicted class	
		P	N
Actual class	P	*TP*	*FN*
	N	*FP*	*TN*

	Predicted		
Actual		A	Not A
	A	$\mathbf{TP_A}$	$E_{AB} + E_{AC}$
	Not A	$E_{BA} + E_{CA}$	$\mathbf{TP_B}$ + E_{BC} E_{CB} + $\mathbf{TP_C}$

	Predicted		
Actual		C	Not C
	C	$\mathbf{TP_C}$	$E_{CA} + E_{CB}$
	Not C	$E_{AC} + E_{BC}$	$\mathbf{TP_A}$ + E_{AB} E_{BA} + $\mathbf{TP_B}$

	Predicted		
Actual		B	Not B
	B	$\mathbf{TP_B}$	$E_{BA} + E_{BC}$
	Not B	$E_{AB} + E_{CB}$	$\mathbf{TP_A}$ + E_{AC} E_{CA} + $\mathbf{TP_C}$

Figure: Confusion Matrix for Multiclass classification. Credit: Tilani Gunawardena

Confusion Matrix

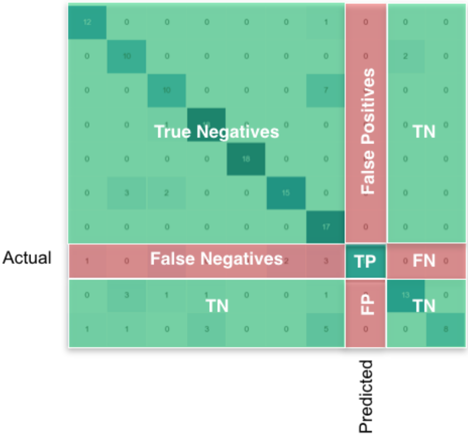


Figure: Confusion Matrix for Multiclass classification.

Confusion Matrix

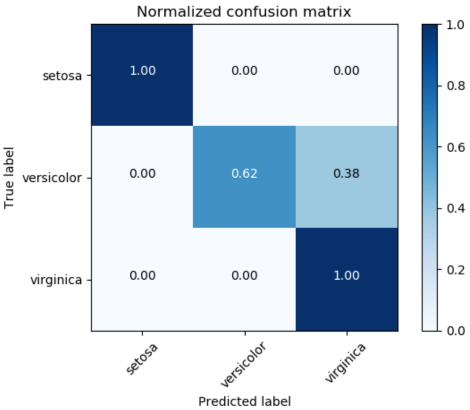


Figure: Confusion Matrix for Multiclass classification.

Multi-label Classification

- Example-based measures
  - Subset Accuracy
  - Hamming Loss
- Label-based measures
  - Macro-B
  - Micro-B

Multi-label Example-based measures

- **Subset Accuracy** evaluates the proportion of test examples whose predicted label set coincides with the ground-truth label set.

$$\frac{1}{p}\sum_{i=1}^{p}[[h(x_i) = Y_i]]$$

  with $[[\pi]]$ returns 1 if predicate π holds and 0 otherwise.

- **Hamming Loss** evaluates the proportion of misclassified instance-label pairs.

$$\frac{1}{p}\sum_{i=1} p\frac{1}{q}|h(x_i)\Delta Y_i|$$

  with Δ is symmetric difference; $|.|$ is the cardinaltily of a set.

Multi-label Example-based measures

```
>>> import numpy as np
>>> from sklearn.metrics import accuracy_score
>>> y_pred = [0,2,1,3]
>>> y_true = [0,1,2,3]
>>> accuracy_score(y_true, y_pred)
0.5
>>> accuracy_score(y_true, y_pred, normalize = False)
2
>>> accuracy_score(np.array([[0,1],[1,1]]),
                   np.array([[1,1],[1,1]]),
                   normalize = False)
1
>>> from sklearn.metrics import hamming_loss
>>> hamming_loss(y_true, y_pred)
0.5
```

Multilabel Example-based measures

- For Hamming loss, the smaller the value, the better the generalization performance.
- For Subset Accuracy, the larger the value, the better the performance.

Multilabel Classification - Label-based measures

On each label y_i, four basic quantitiesregarding the test examples are commonly used: TP_j (True Positive), FP_j (False Positive), TN_j (True Negative), and FN_j (False Negative).
Let $B(TP_j; FP_j; TN_j; FN_j)$ denote a certain binary classification measure:

- $Macro - B = \sum_{j=1}^{q} \frac{1}{q} B(TP_j; FP_j; TN_j; FN_j)$ (assuming equal importance for each label)
- $Micro - B = B(\sum_{j=1}^{q} TP_j; \sum_{j=1}^{q} FP_j; \sum_{j=1}^{q} TN_j; \sum_{j=1}^{q} FN_j)$ (assuming equal importance for each example)

Among popular choices of $B \in \{accuracy, precision, recall, F\}$, the larger the macro/micro-B value, the better the performance.

Workflow

- Training phase: selecting algorithms
- Validation phase: model selection
  - Selecting between multiple methods
  - Fine-tuning parameters (Model complexity and Regularization)
  - Feature selection: No. of input variables and the Correct input variable
- Test phase: model assessment

Selecting algorithms

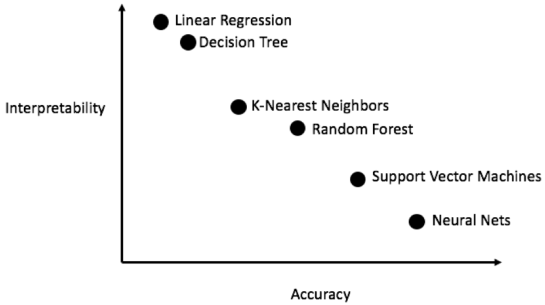


Figure: Accuracy-Interpretability trade-off. Credit: Ansaro

Goals

- **Model selection**: estimating the performance of different models to choose the best model.
- **Model assessment**: having chosen a final model, estimating its prediction error (generalization error) on new data.

Data-rich situation

Randomly divide the dataset into three parts:

- Training set: fit the model
- Validation set: estimate prediction error for *model selection*
- Test set: assessment of the generalization error of the final chosen model

Training/Validation/Test sets splitting

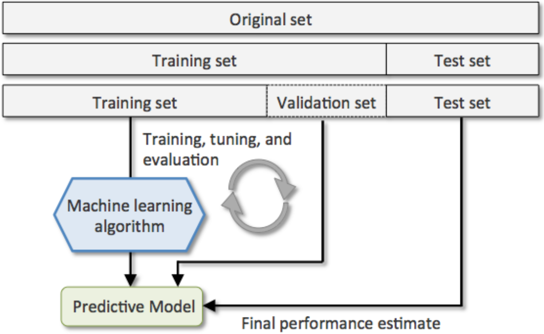


Figure: Training/Validation/Test sets splitting (Credit: Shan-Hung Wu & DataLab)

Training/Validation/Test sets splitting

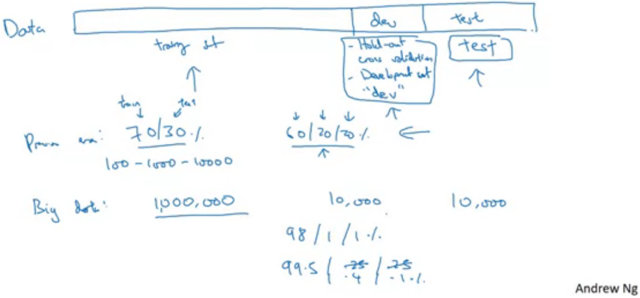


Figure: Training/Dev/Test sets splitting (Credit: Corner)

Stratified sampling

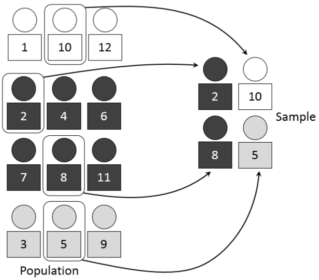


Figure: Stratified sampling. Credit: Wikipedia

Insufficient data

- Too difficult to give a general rule on how much training data
- Depends on the signal-to-noise ratio of the underlying function
- Depends on the complexity of the models

Insufficient data

In order to select the best model with respect to test error, we need to estimate this test error by:

- *Indirectly* estimate test error by making an *adjustment* to the training error to account for the bias due to overfitting
- *Directly* estimate the test error using efficient sample re-use (cross-validation and the bootstrap)

Indirect estimate

C_p statistic

The C_p estimate of the test MSE is computed using the equation

$$C_p = \frac{1}{n}(RSS + 2d\hat{\sigma}^2)$$

where $\hat{\sigma}^2$ is an estimate of $Var[\epsilon]$ [1]

- C_p statistics adds a penalty of $2d\hat{\sigma}^2$ to the training RSS
- C_p tends to take a small value of models with a low test error. The model with the lowest C_p value should be chosen.

[1] $Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon]$

Indirect estimate

For linear regression model with d predictors:

Akaike information criterion

Akaike information criterion or AIC is given by

$$AIC = \frac{1}{n\sigma^2}(RSS + 2d\hat{\sigma}^2)$$

Bayesian information criterion

Bayesian information criterion or BIC is given by

$$BIC = \frac{1}{n\sigma^2}(RSS + \log(n)d\hat{\sigma}^2)$$

Indirect estimate

Adjusted R^2

For a least squares model with d variables, the adjusted R^2 is calculated as

$$Adjusted\ R^2 = 1 - \frac{RSS/(n-d-1)}{TSS(n-1)}$$

Unlike C_p , AIC, and BIC, for which a small value indicates a model with a low test error, a large value of adjusted R^2 indicates a model with a small test error.

Indirect estimate

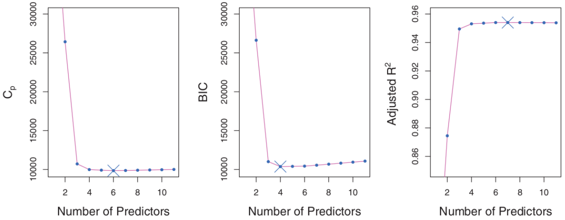


Figure: C_p, BIC and adjusted R^2. Credit: ISL book

K-fold Cross-validation

- The simplest and most popular way to estimate the test error.
- How to do:
  1. Randomly split the data into K roughly equal parts.
  2. For each k:
     - Leave the kth part out, fit the model using the other (k-1) parts, called $\hat{f}^{(-k)}(x)$
     - Calculate the performance or prediction error of the $\hat{f}^{(-k)}(x)$ on the kth part.
  3. Average the errors
- Leave-one-out Cross-validation or LOOCV is a special case of k-fold cross-validation with k = N

K-fold Cross-validation

Figure: 10-fold Cross Validation. Credit: Shan-Hung Wu & DataLab

K-fold Cross-validation

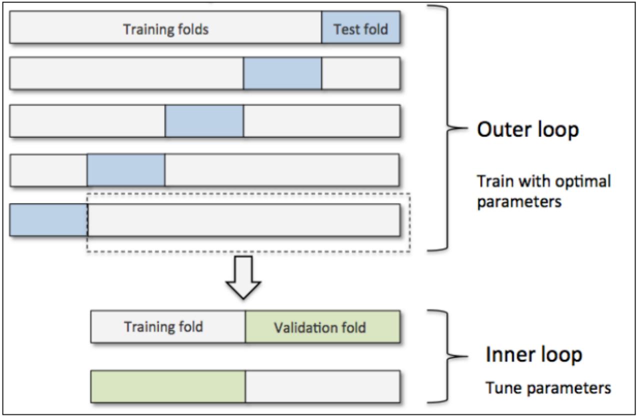


Figure: 5 by 2-fold Cross Validation. Credit: Shan-Hung Wu & DataLab

K-fold Cross-validation

The correct way to carry out cross-validation:

1. Divide the samples into K cross-validation folds (groups) at random.
2. For each fold:
    - Find a subset of “good” predictors that show fairly strong (univariate) correlation with the class labels, using all of the samples except those in fold k.
    - sing just this subset of predictors, build a multivariate classifier, using all of the samples except those in fold k.
    - Use the classifier to predict the class labels for the samples in fold k.

K-fold Cross-validation

- In practice, $k = 5$ or $k = 10$ are recommended.
- "One-standard error" rule could be used with CV.

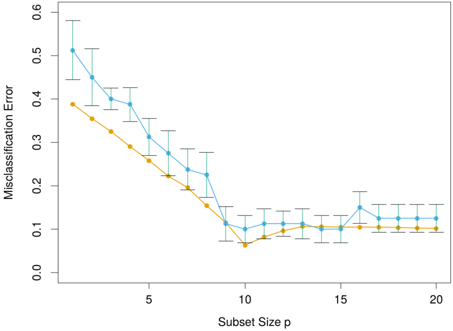


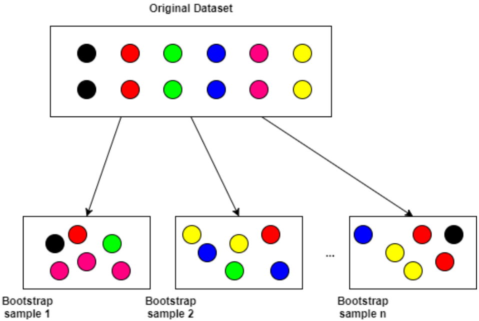
Figure: Prediction error (orange) and tenfold cross-validation curve (blue) estimated from a single training set.A model with $p = 9$ would be chosen. Credit: ESL

Bootstrap

Bootstrap

A **bootstrap sample** is a random sample of the data taken *with replacement*. This means that, after a data point is selected for the subset, it is still available for further selection.

The unselected samples is called Out-of-bag samples.



How Bootstrap estimate Prediction error?

Predict on the original dataset

$$\hat{Err}_{boot} = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{i=1}^{N} L(y_i, \hat{f}_{*b}(x_i))$$

Overlap between bootstrap set and original set can make overfit predictions look unrealistically good

How Bootstrap estimate Prediction error?

Leave-one-out Bootstrap

$$\hat{Err}_{LOOB} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L(y_i, \hat{f}_{*b}(x_i))$$

where:

- C^{-i} is the set of indices of the bootstrap samples b that do not contain observation i.
- Bootstrap sample need to be large enough to ensure $|C^{-i}| > 0 \forall i$

How Bootstrap estimate Prediction error?

".653 estimator"

$$\hat{Err}_{.632} = 0.368 \times \bar{err} + 0.632 \times \hat{Err}_{LOOB}$$

The .632 estimator works well in "light fitting" situations, but can break down in overfit ones.

How Bootstrap estimate Prediction error?

".653+ estimator"

$$\hat{Err}_{.632} = (1 - \hat{w})\bar{err} + \hat{w} \times \hat{Err}_{LOOB}$$

with

$$\hat{w} = \frac{0.632}{1 - 0.368\hat{R}}$$

$$\hat{R} = \frac{\hat{Err}_{LOOB} - \bar{err}}{\hat{\gamma} - \bar{err}}$$

$$\hat{\gamma} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{i'=1}^{N} L(y_i, \hat{f}(x_{i'}))$$

γ: no-information error rate; $\hat{R}$: relative overfitting rate