



Rensselaer

why not change the world?®

Evaluating Regression & Classification Models

Ahmed Eleish

Data Analytics ITWS/MGMT-4600/6600, CSCI-4600, BCBP-4600

September 29th 2026

Tetherless World Constellation
Rensselaer Polytechnic Institute



Contents

- Model errors, model generalization
- Cross-validation strategies
- Evaluating regression models
- Evaluating classification models



Accurate vs. Precise



**High Accuracy
High Precision**



**Low Accuracy
High Precision**



**High Accuracy
Low Precision**



**Low Accuracy
Low Precision**

<http://climatica.org.uk/climate-science-information/uncertainty>

Evaluation of Model Training

- "Training" refers to the process through which a model "learns" patterns from the dataset.
- To robustly evaluate predictive models the training process is repeated multiple times according to commonly used sampling strategies.
- The goal is for model training to be exposed to as much of the variation in structure in the dataset as is reasonably possible... *remember sample vs. population*
- Each training iteration is evaluated separately, with the average performance of the model over the number of training iterations considered an indicator of training success.

Training, Validation and Test sets

- **Training:** subset of dataset used as input to the model's training algorithm
- **Validation:** subset used to evaluate each round of training
- **Test:** subset used to test the final model

e.g.

- The Iris dataset is initially split into a *training + validation* set (90% - 135 obs) and a *test* set (10% - 15 obs) ~ this depends on the size of the dataset.
- Over 10 iterations, the *training + validation* set is split into *training* (100 obs) and *validation* (35 obs). After training is complete, the average **training error** is calculated.
- The final model is tested on the *test* set (15 obs) and the **test error** is calculated.

Errors

- The error from validation data is called as the “**training error**”
- The error from test data is referred to as the “**test error**”
- The error on the test data is a good indication of how well the classifier will perform on new data (**not used during training**) and this is known as the *generalization*.
- If the model generalizes well, then it will perform well on new data that have *similar structure* to the training data... *sample vs. population*
- The test error is also called the generalization error.

Resource/Reference: Introduction to Statistical Learning with R, 7th Edition

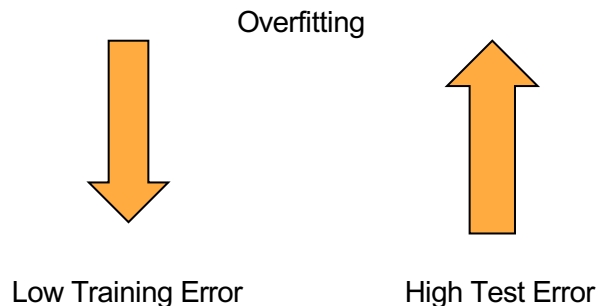
Terminology Confusion!

- 'Test' and 'validation' are used interchangeably in academia and industry!!
- That's fine... just be clear when documenting your analysis strategy.
- Decisions related to model development and evaluation strategies depend on the problem/dataset.

https://en.wikipedia.org/wiki/Training_validation_and_test_data_sets

Overfitting

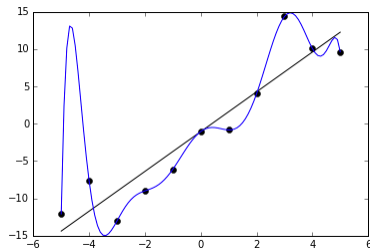
- Another related concept to Generalization is “overfitting”.
- If the model has very **low training error** but it has **high test error**, then it is over fitting.



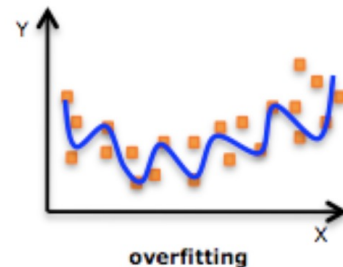
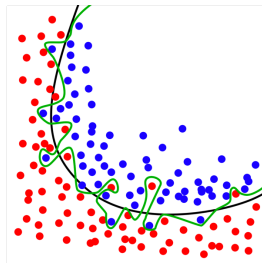
Resource/Reference: Introduction to Statistical Learning with R, 7th Edition

Overfitting

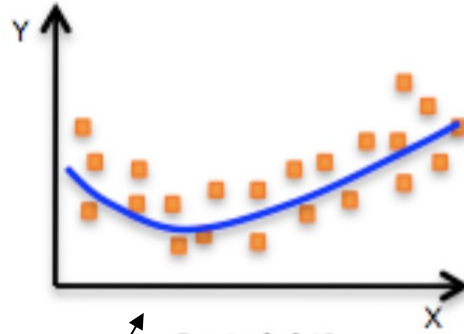
- This is a good indication that the model may have learned to ***model the noise*** in the training data, instead of the learning from the underlying structure of the data.
- Overfitting is an indication of poor generalization.



Image/Photo Credit:
https://en.wikipedia.org/wiki/Overfitting#/media/File:Overfitted_Data.png

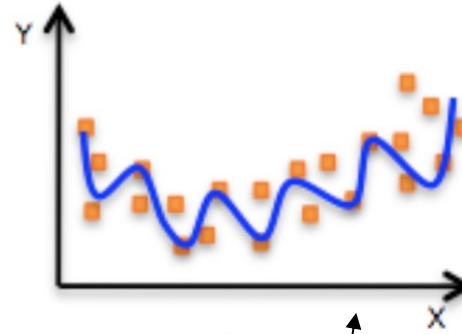


Image/Photo Credit:
<http://pingax.com/regularization-implementation-r/>



Just right!

Model is fitting to
the structure of the data



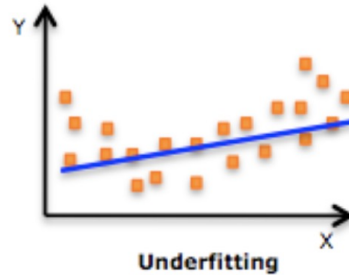
overfitting

Model is fitting to
the noise of the data

Image/Photo Credit: <http://pingax.com/regularization-implementation-r/>

Underfitting

- **Underfitting** occurs when a statistical model cannot adequately capture the underlying structure of the data.
- In other words, **underfitting takes place when the model has not properly learned the structure of the data.**



Image/Photo Credit: <http://pingax.com/regularization-implementation-r/>

Cross-Validation



Robustly Validating Models

- Cross-validation is a method to robustly evaluate/validate models by iteratively splitting the dataset into subsets for training and testing, training the model, and computing and evaluation metric
- There are several cross-validation strategies
 - K-fold cross-validation
 - Monte Carlo cross-validation
 - Leave-One-Out cross-validation

[https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics))

K-fold Cross Validation

- In k-fold cross validation, the data are segmented into k **disjoint partitions**.
- During each iteration, one partition is used as the test set and the remaining $k-1$ (combined) are used for training; The process is repeated k times, so that each partition is used exactly one time for the validation.



By [Gufosowa](#) - Own work, [CC BY-SA 4.0](#),
<https://commons.wikimedia.org/w/index.php?curid=82298768>

Resource/Reference: Introduction to Statistical Learning with R, 7th Edition - Chapter 5

Monte Carlo Cross Validation (Repeated random sub-sampling)

- In Monte Carlo cross validation, the dataset is split into training/test sets over n iterations with the samples in each set selected at random.
- The ratio between partition sizes may be constant or vary over the iterations.
- Commonly used in research, considered robust because of the averaging effect over multiple iterations.
- Downside: since selection is random, some observations may not end up in test sets and some may be oversampled

Resource/Reference: Introduction to Statistical Learning with R, 7th Edition - Chapter 5

Leave One Out Cross Validation (LOOCV)

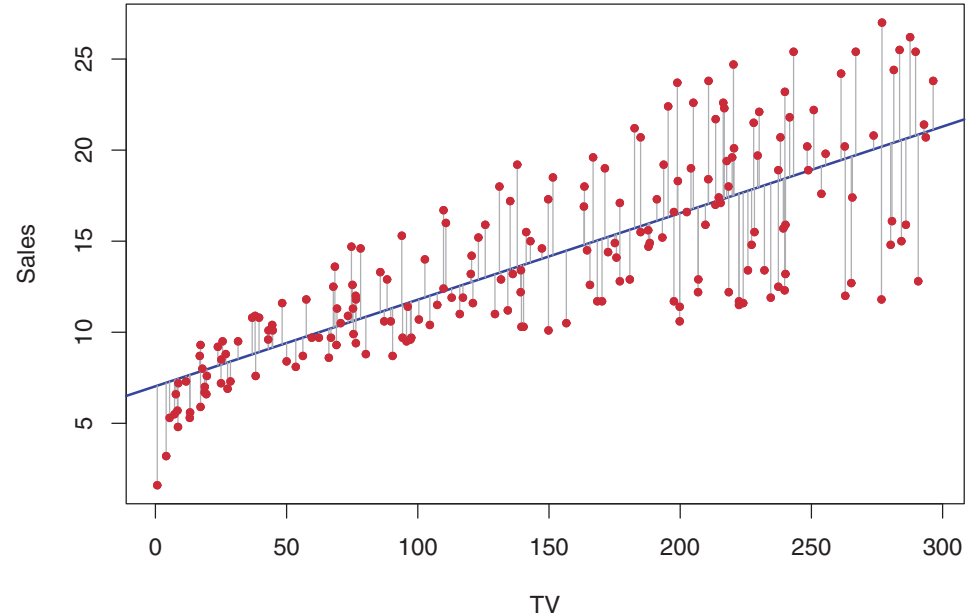
- Given a dataset with n observations, for n iterations drop *one* observation and use all the others for training; test using the 1 observation left out
- Depending on the size of the dataset, may be computationally expensive.

Resource/Reference: Introduction to Statistical Learning with R, 7th Edition - Chapter 5

Evaluating Regression Models

Evaluating Linear Models

- Sales vs. TV ad spending
- Sales in 1000s of units
- TV ad spending in 1000s of \$



Residual Sum of Squares (RSS)

For given data $(x_1, y_1), \dots, (x_n, y_n) \in \mathbb{R} \times \mathbb{R}$,

- Residual Sum of Squares (RSS), the i th residual $e_i = y_i - \hat{y}_i$

$$\text{RSS} = e_1^2 + e_2^2 + \dots + e_n^2$$

Evaluating Linear Models

1. Assessing the coefficient estimates

1.1 Values of coefficients >> their Std. errors

$$SE(\hat{\beta}_1) = \frac{RSE}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2}}$$

1.2. High t-statistic

$$t = \frac{\hat{\beta}_1}{SE(\hat{\beta}_1)}$$

1.3. Low p-value

Hypothesis (more TV ads → more sales)

X ~~H0~~ : There is no relationship between X and Y

✓ ~~Ha~~ : There is some relationship between X and Y

	Coefficient	Std. error	t-statistic	p-value
Intercept	7.0325	0.4578	15.36	< 0.0001
TV	0.0475	0.0027	17.67	< 0.0001

Quantity	Value
Residual standard error	3.26
R^2	0.612
F-statistic	312.1

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

Evaluating Linear Models

2. Assessing the model's fit

Residual Standard Error

- Mean sales $\approx 14,000$ units
- $RSE = 3.26 = 3,260$ units

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

R^2

- Measures the how much of the variability in Y can be explained using X (as a predictor)
- has a value between 0,1

$$R^2 = \frac{TSS - RSS}{TSS} = 1 - \frac{RSS}{TSS}$$

$$RSS = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad TSS = \sum_{i=1}^n (y_i - \bar{y}_i)^2$$

Quantity	Value
Residual standard error	3.26
R^2	0.612
F-statistic	312.1

Measures of Model Error

Mean Absolute Error

- Mean(||Predicted value - Real value||)

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

- low sensitivity to outliers
- measured in the same units as target variable
- good for noisy data

Mean Squared Error

- Mean((Predicted value - Real value)²)

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

- High sensitivity to outliers
- measured in squared units of target variable
- good for model optimization (penalizes larger errors more severely)

Root Mean Squared Error

- $\text{SquareRoot}(\text{Mean}((\text{Predicted value} - \text{Real value})^2))$

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

- High sensitivity to outliers
- measured in the same units as target variable
- good for reporting cross-model performance

R code:

<https://rpi.box.com/s/vyl1b7hspsy0540uw9ceb8dstml837azz>

Evaluating Classification Models

Classification Accuracy

- *Accuracy = (Number of correct predictions) / (Total number of data points)*

$$= \frac{TP+TN}{N}$$

- Simplistic evaluation of model
- Classification error = 1 – **Accuracy**

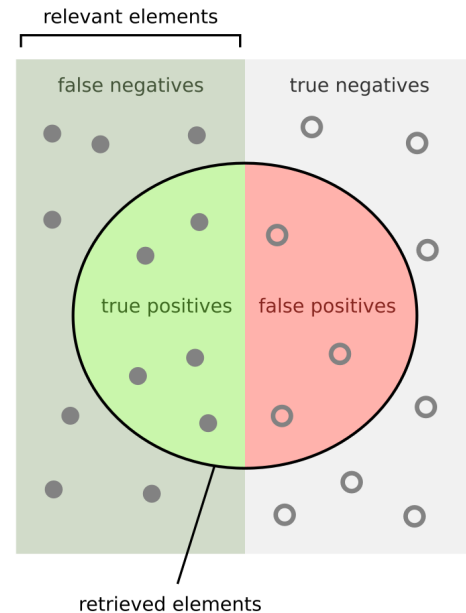
$$= \frac{FP+FN}{N}$$

		<i>Predicted Value</i>	
		Positive	Negative
<i>Real Value</i>	Positive	TP	FN
	Negative	FP	TN

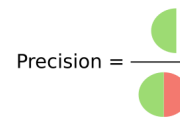
Per Class Evaluation

$$\text{Recall} = \frac{\text{instances **correctly** classified as positive}}{\text{all positive instances}}$$

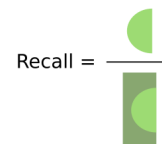
$$\text{Precision} = \frac{\text{instances **correctly** classified as positive}}{\text{all instances classified as positive}}$$



How many retrieved items are relevant?



How many relevant items are retrieved?

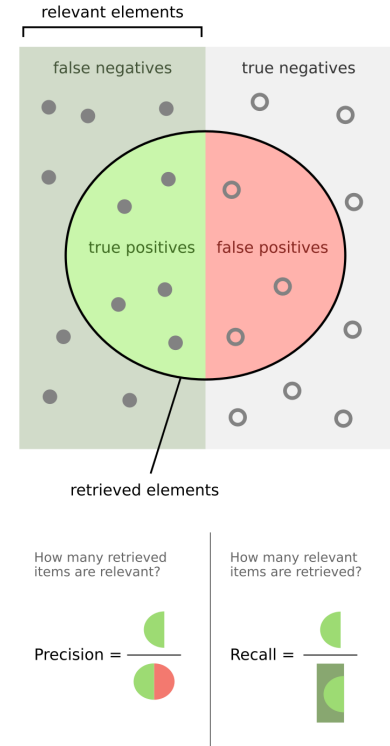


https://en.wikipedia.org/wiki/Precision_and_recall

Credit (unmodified): Walber (own work) - [CC BY-SA 4.0](https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg) - https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg

Evaluation Metrics – Per Class

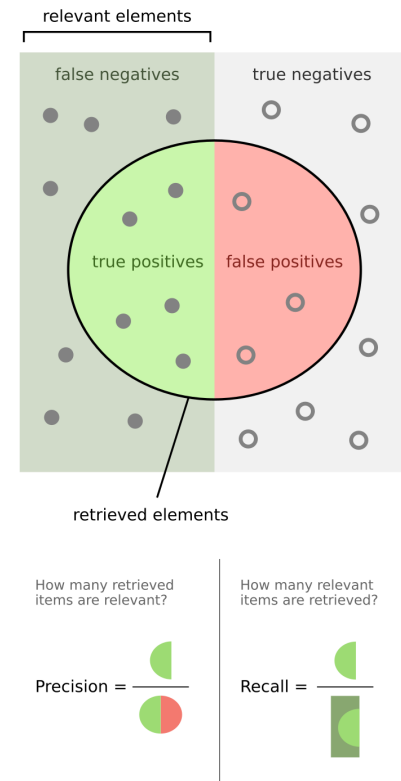
- **Recall** = $(\text{True Positive}) / (\text{True Positive} + \text{False Negative})$
 - Fraction of positive class that is correctly identified
- **Precision** = $(\text{True Positive}) / (\text{True Positive} + \text{False Positive})$
 - Fraction of positive predictions that belong to the positive class
- **F1** = $2 [(\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})]$
 - $F1 = (\text{True Positive}) / [\text{True Positive} + 1/2 * (\text{False Positive} + \text{False Negative})]$
 - Harmonic mean (weighted average) of precision and recall



Credit (unmodified): Walber (own work) - [CC BY-SA 4.0](https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg) - https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg

Additional Evaluation Metrics – Per Class

- **Specificity** = $(\text{True Negative}) / (\text{True Negative} + \text{False Positive})$
 - Fraction of negative predictions belonging to negative class
- **Fall-out** = $(\text{False Positive}) / (\text{True Negative} + \text{False Positive})$
 - Fraction of negative class correctly identified
- **Miss Rate** = $(\text{False negative}) / (\text{True positive} + \text{False negative})$
 - Fraction of positive class misclassified



Credit (unmodified): Walber (own work) - [CC BY-SA 4.0](https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg) - https://en.wikipedia.org/wiki/Precision_and_recall#/media/File:Precisionrecall.svg

Example

- Accuracy = $(31+37+34) / 105 = \sim 97\%$
- Versicolor:
 - TP: 37
 - FP: 1
 - FN: 2
 - TN: 65
 - Precision = $TP / (TP + FP) = 37 / (37 + 1) = \mathbf{0.973}$
 - Recall = $TP / (TP + FN) = 37 / (37 + 2) = \mathbf{0.948}$
 - F1 = $TP / [TP + 0.5 \cdot (FP + FN)] = 37 / (37 + 0.5 \cdot (1 + 2)) = \mathbf{0.961}$

Real labels

Predicted labels

	setosa	versicolor	virginica
setosa	31	0	0
versicolor	0	37	2
virginica	0	1	34

R code:

<https://rpi.box.com/s/vyl1b7hspsy0540uw9ceb8dstml837azz>

Thanks!