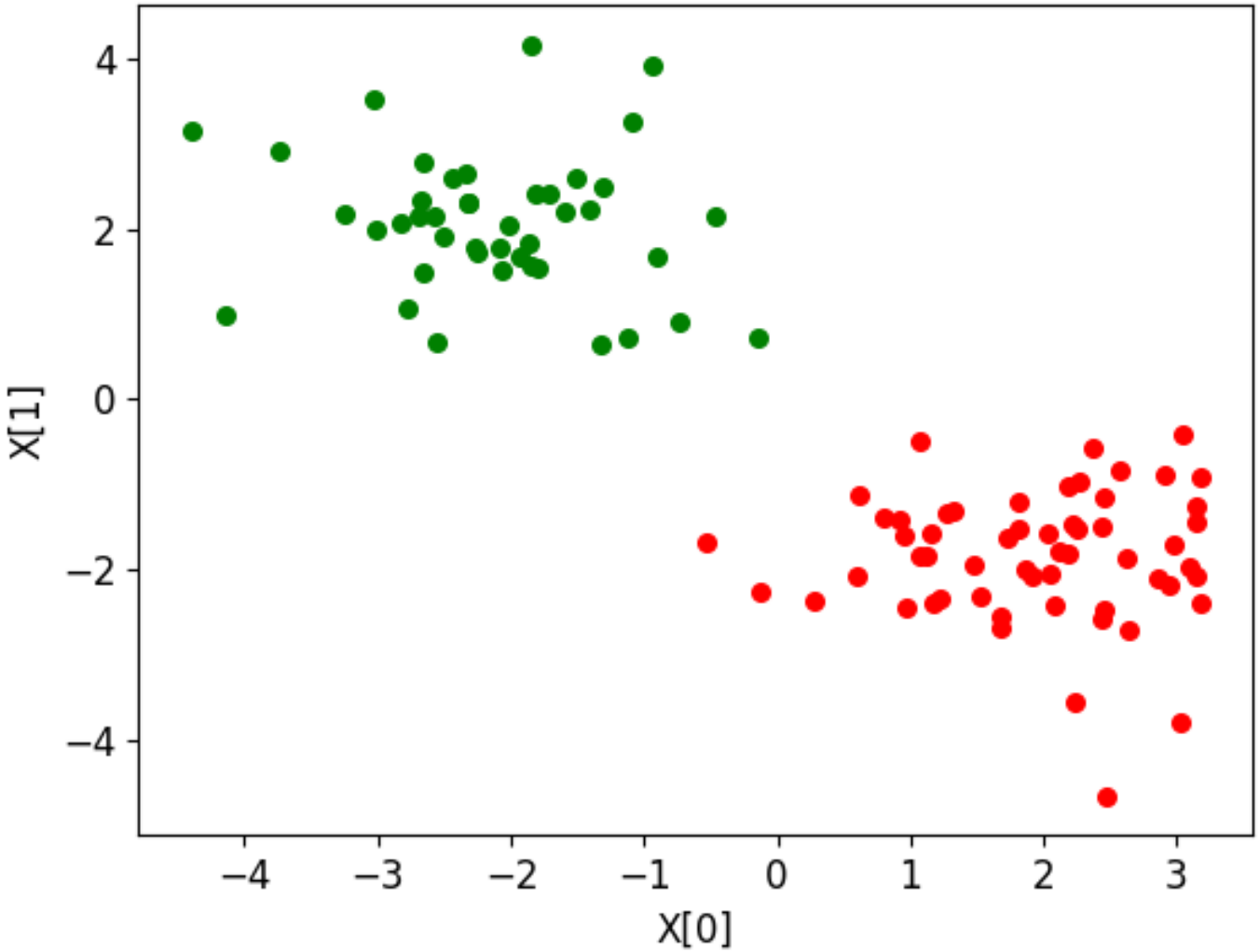
POLITECNICO DI MILANO

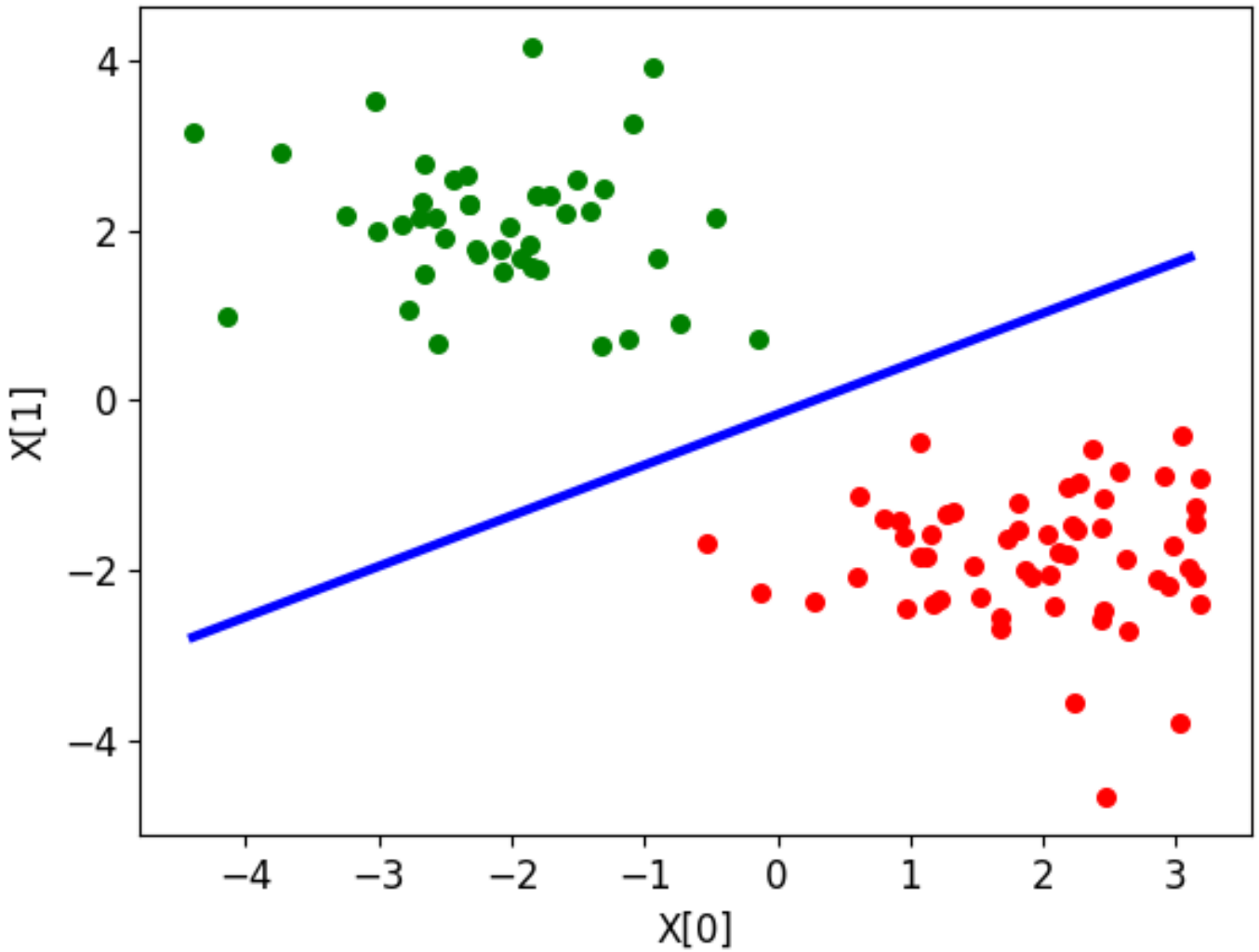
Classification

Data Mining and Text Mining (UIC 583 @ Politecnico di Milano)

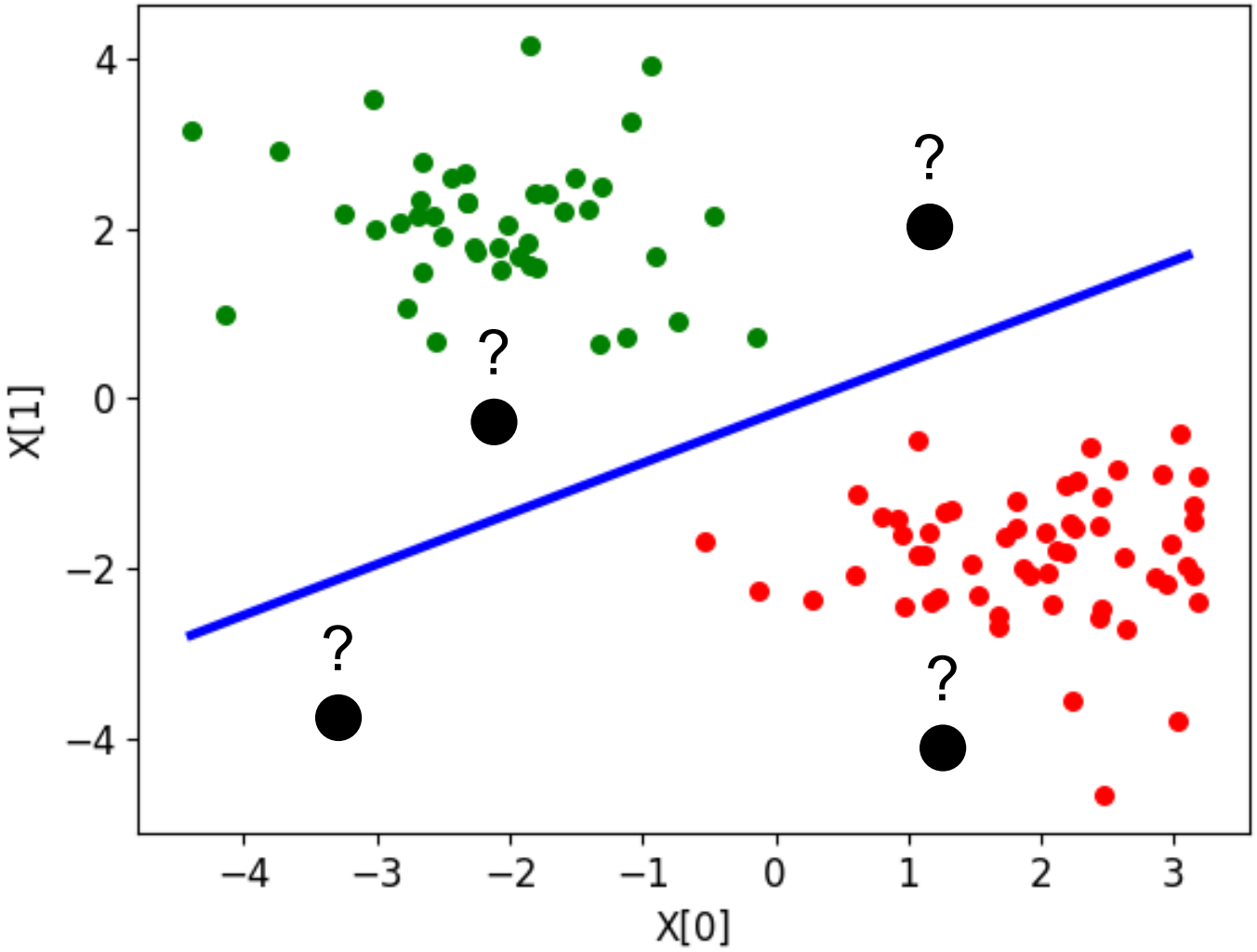
Prof. Pier Luca Lanzi



In classification, we have data that have been labeled according to a certain concept (good/bad, positive/negative, etc.) identified by a supervisor who actually labeled the data



The goal of classification is to compute a model that can discriminate between examples that have been labeled differently, e.g., the points above the planes are green and the one below are red



The model is then used to label previously unseen points.

As for regression, we have to compute a model using known data that will perform well on unknown data.

Contact Lenses Data

Age	Spectacle prescription	Astigmatism	Tear production rate	Recommended lenses
Young	Myope	No	Reduced	None
Young	Myope	No	Normal	Soft
Young	Myope	Yes	Reduced	None
Young	Myope	Yes	Normal	Hard
Young	Hypermetrope	No	Reduced	None
Young	Hypermetrope	No	Normal	Soft
Young	Hypermetrope	Yes	Reduced	None
Young	Hypermetrope	Yes	Normal	hard
Pre-presbyopic	Myope	No	Reduced	None
Pre-presbyopic	Myope	No	Normal	Soft
Pre-presbyopic	Myope	Yes	Reduced	None
Pre-presbyopic	Myope	Yes	Normal	Hard
Pre-presbyopic	Hypermetrope	No	Reduced	None
Pre-presbyopic	Hypermetrope	No	Normal	Soft
Pre-presbyopic	Hypermetrope	Yes	Reduced	None
Pre-presbyopic	Hypermetrope	Yes	Normal	None
Presbyopic	Myope	No	Reduced	None
Presbyopic	Myope	No	Normal	None
Presbyopic	Myope	Yes	Reduced	None
Presbyopic	Myope	Yes	Normal	Hard
Presbyopic	Hypermetrope	No	Reduced	None
Presbyopic	Hypermetrope	No	Normal	Soft
Presbyopic	Hypermetrope	Yes	Reduced	None
Presbyopic	Hypermetrope	Yes	Normal	None

Rows are typically called
examples or data points

Columns are typically called
attributes, variables or features

The target variable to be
predicted is usually called “class”

A Model for the Contact Lenses Data


```
If tear production rate = reduced then recommendation = none
If age = young and astigmatic = no
   and tear production rate = normal then recommendation = soft
If age = pre-presbyopic and astigmatic = no
   and tear production rate = normal then recommendation = soft
If age = presbyopic and spectacle prescription = myope
   and astigmatic = no  then recommendation = none
If spectacle prescription = hypermetrope and astigmatic = no
   and tear production rate = normal then recommendation = soft
If spectacle prescription = myope and astigmatic = yes
   and tear production rate = normal then recommendation = hard
If age young and astigmatic = yes
   and tear production rate = normal then recommendation = hard
If age = pre-presbyopic
   and spectacle prescription = hypermetrope
   and astigmatic = yes then recommendation = none
If age = presbyopic and spectacle prescription = hypermetrope
   and astigmatic = yes then recommendation = none
```

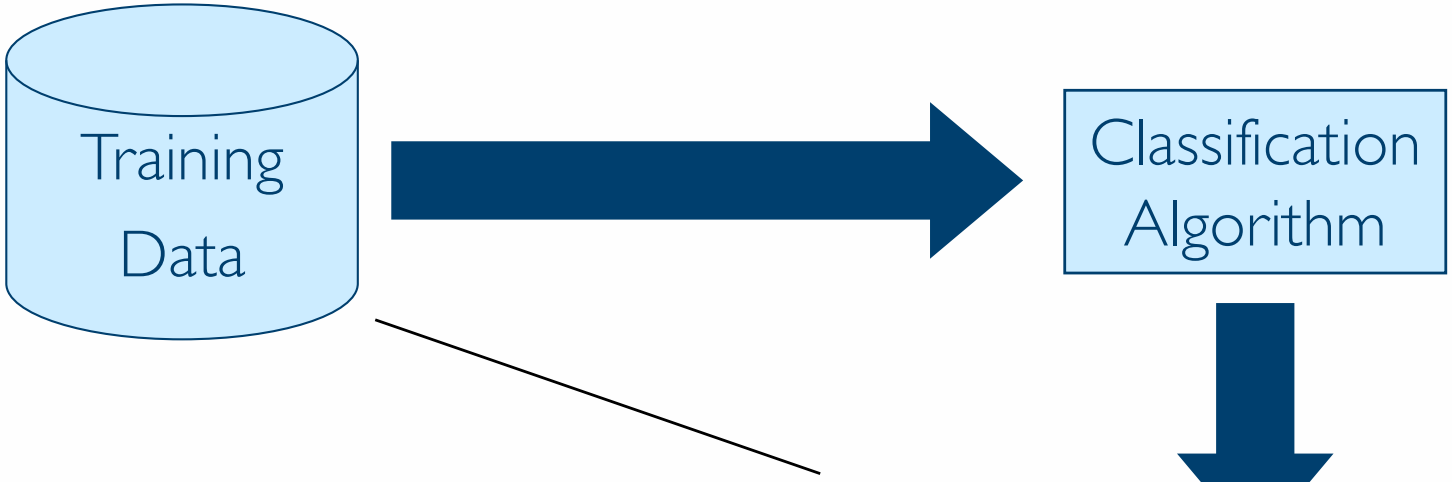
Classification

The target to predict is a label (the class variable)
(good/bad, none/soft/hard, etc.)

Prediction/Regression

The target to predict is numerical

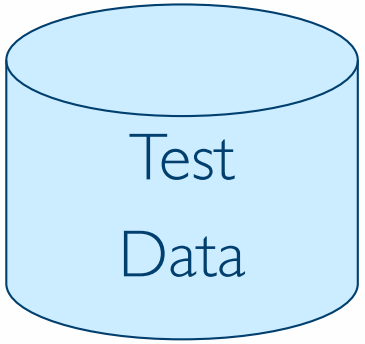
Classification: Model Construction

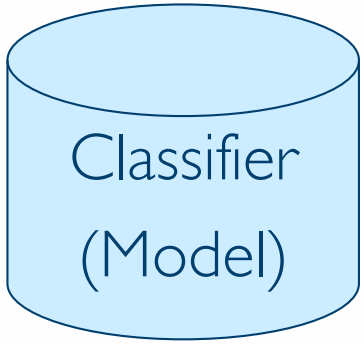


name	rank	years	tenured
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

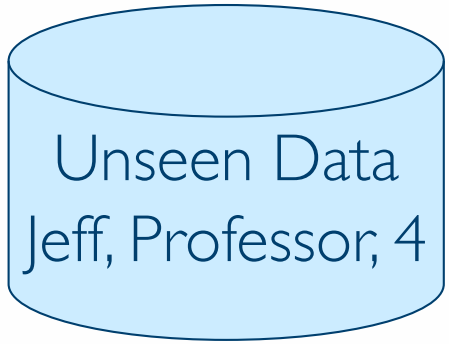
IF rank = 'professor'
OR years > 6
THEN tenured = 'yes'

Classification: Model Usage





name	rank	years	tenured
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

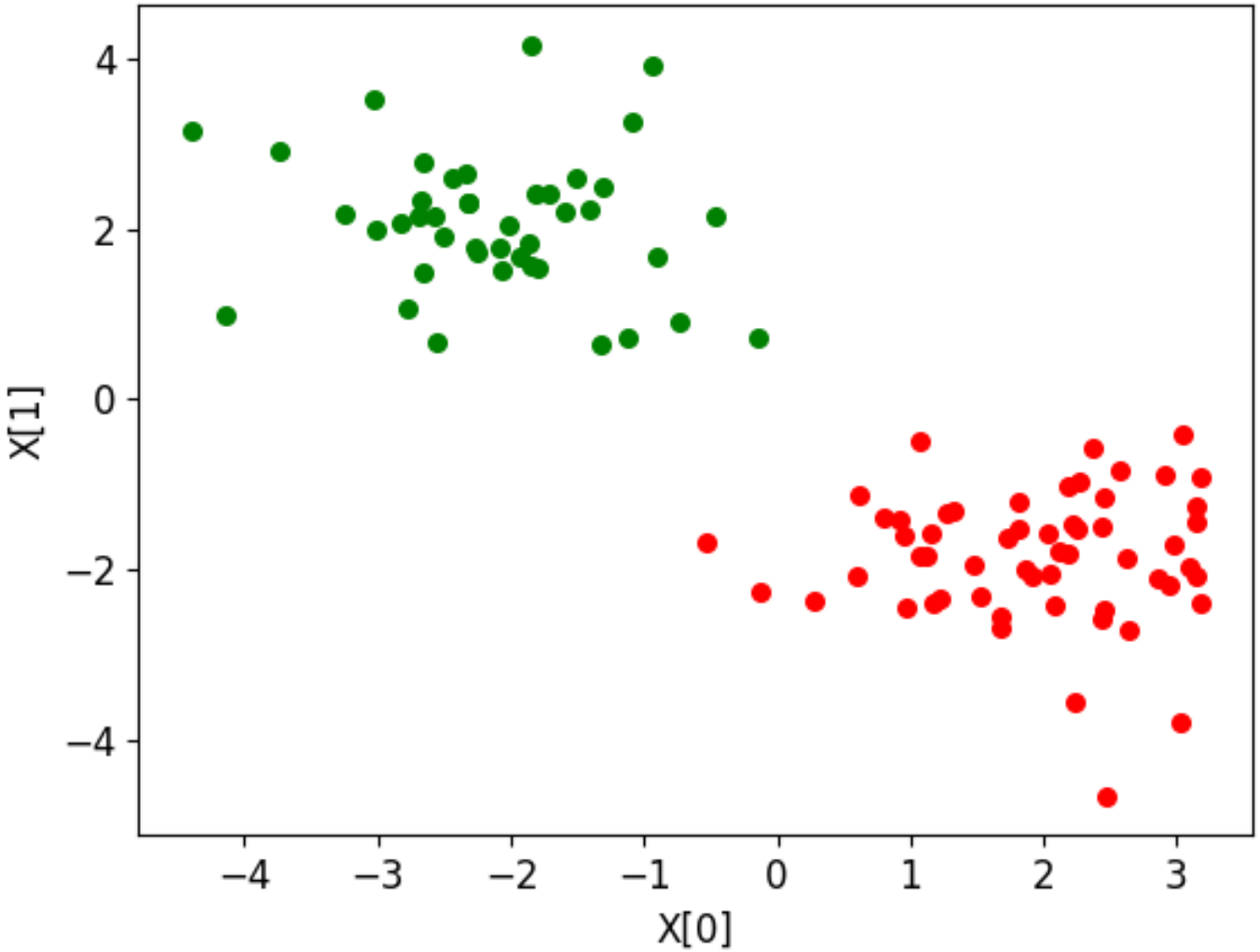


tenured = yes

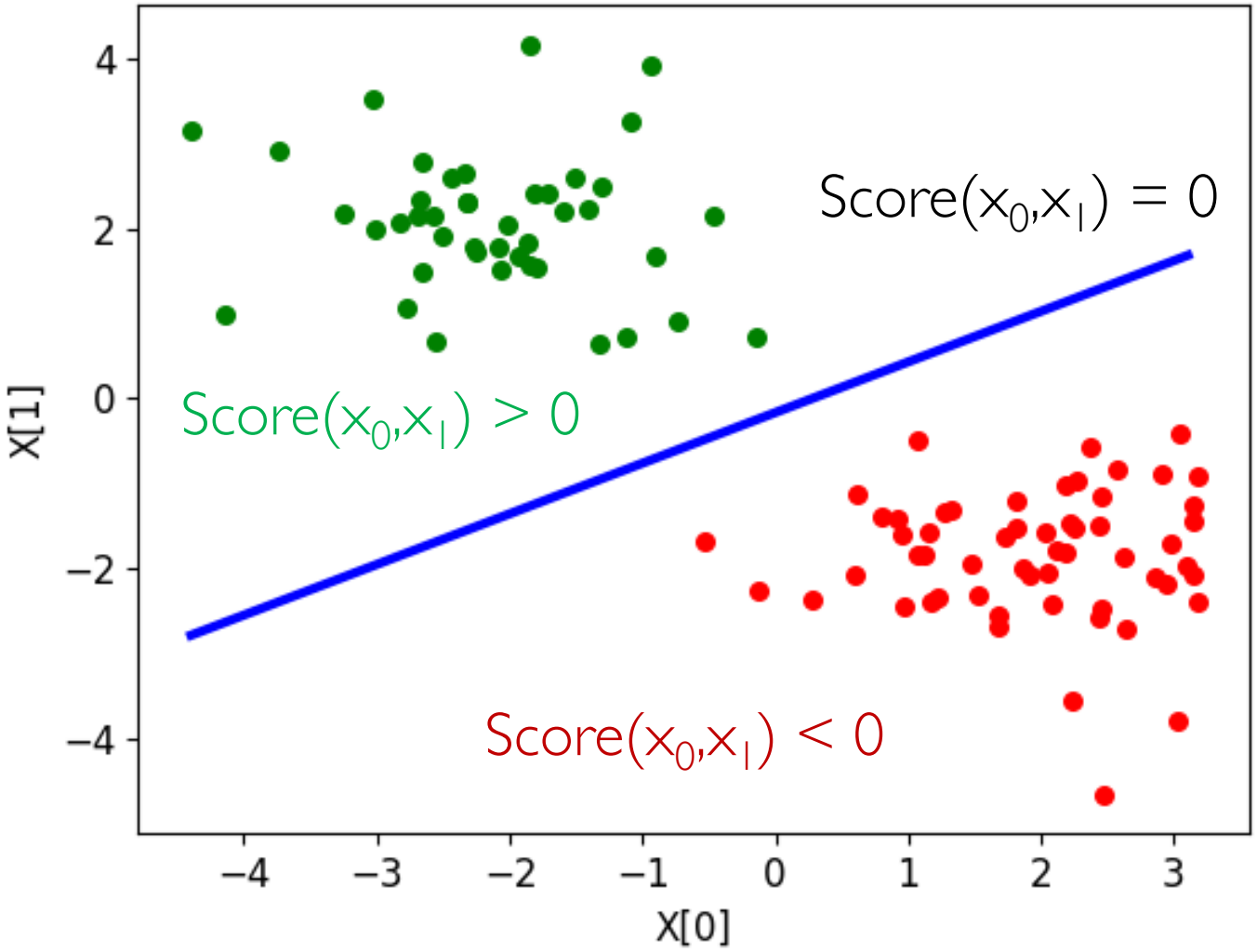
Evaluating Classification Methods

- Accuracy
  - Classifier accuracy in predicting the correct the class labels
- Speed
  - Time to construct the model (training time)
  - Time to use the model to label unseen data
- Other Criteria
  - Robustness in handling noise
  - Scalability
  - Interpretability
  - …

Logistic Regression



To build a model to label the green and red data points we can compute an hyperplane Score(**x**) that separates the examples labeled green from the one labeled red



To predict the labels we check whether the value of Score(x_0, x_1) is positive or negative and we label the examples accordingly

Intuition Behind Linear Classifiers (Two Classes)

- We define a score function similar to the one used in linear regression,

$$Score(\vec{x_i}) = \sum_{j=0}^{D} w_j h_j(\vec{x_i})$$

- The label is determined by the sign of the score value,

$$\hat{y}_i = sign(Score(\vec{x_i}))$$

$$= \begin{cases} +1 & \texttt{if } Score(\vec{x_i}) \geq 0 \\ -1 & \texttt{if } Score(\vec{x_i}) < 0 \end{cases}$$

Logistic Regression

- Well-known and widely used statistical classification method
- Instead of computing the label, it computes the probability of assigning a class to an example, that is,

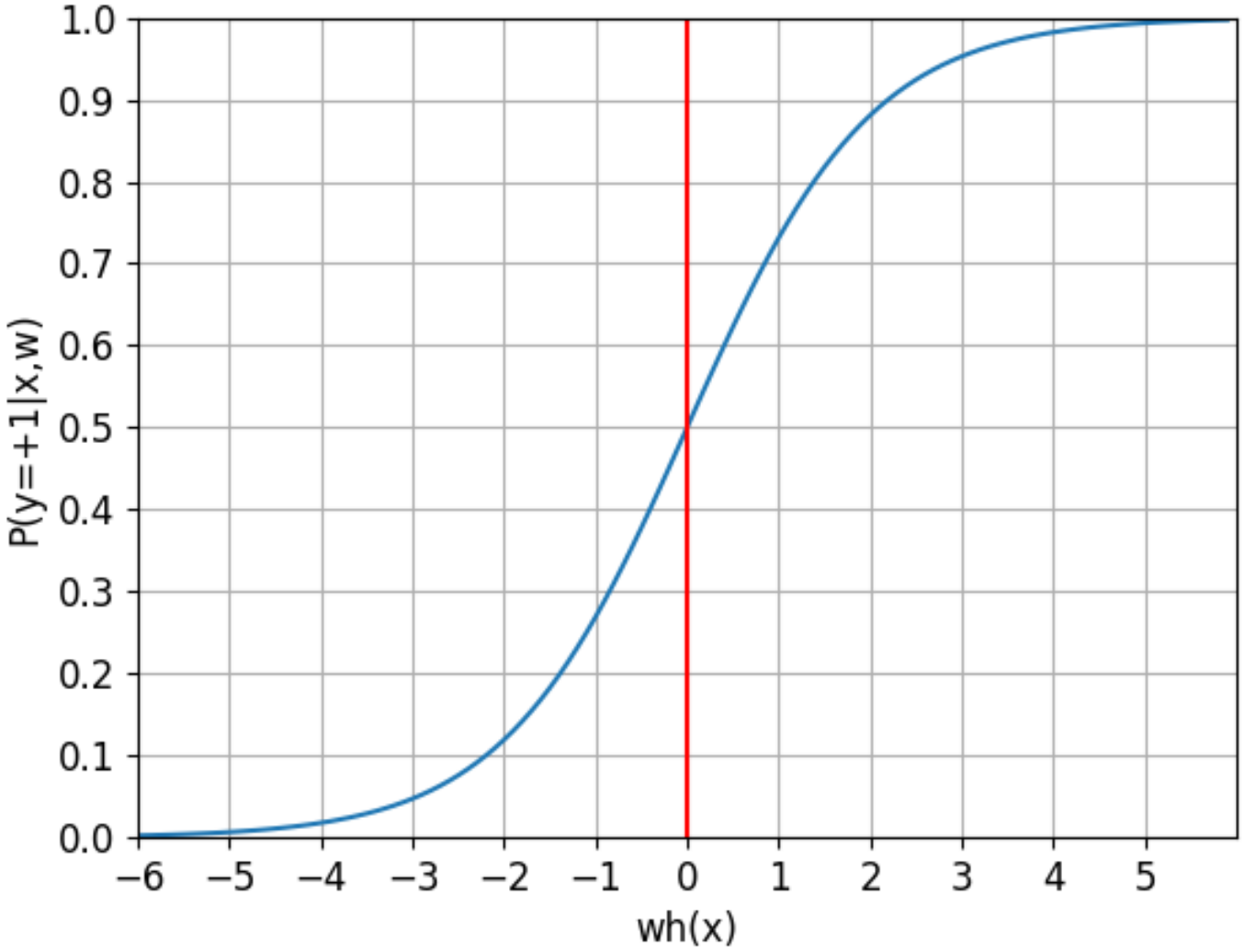
$$P(y_i|\vec{x_i})$$

- For this purpose, logistic regression assumes that,

$$P(\hat{y_i} = +1|\vec{x_i}) = \frac{1}{1 + e^{-Score(\vec{x_i})}}$$

- By making the score computation explicit and using h to identify all the feature transformation h_j we get,

$$P(\hat{y_i} = +1|\vec{x_i}, \vec{w}) = \frac{1}{1 + e^{-\vec{w}h(\vec{x_i})}}$$


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
P(y=+1|x,w)
−6
−5
−4
−3
−2
−1
0
1
2
3
4
5
wh(x)

Logistic Regression

- Logistic Regression search for the weight vector that corresponds to the highest likelihood

$$\ell(\vec{w}) = \prod_{I=0}^{N} P(y_i|\vec{x_i}, \vec{w})$$

- For this purpose, it performs a gradient ascent on the log likelihood function

$$\ell\ell(\vec{w}) = \ln \ell(\vec{w})$$

- Which updates weight j using,

$$\frac{\partial \ell\ell}{\partial w_j} = \sum_{I=1}^{N} h_j(\vec{x_i})(1[y_i = +1] - P(y = +1|\vec{x_i}, \vec{w}))$$

Classification using Logistic Regression

- To classify an example X, we select the class Y that maximizes the probability P(Y=y | X) or check

$$\frac{P(y=-1)|x)}{P(y=+1)|x)} > 1 \qquad \text{or} \qquad e^{-\vec{w}h(\vec{x})} > 1$$

- By taking the natural logarithm of both sides, we obtain a linear classification rule that assign the label Y=-1 if

$$\sum_{i=0}^{D} w_i h_i(x) < 0$$

- Y=+1 otherwise

Overfitting & Regularization

L_1 & L_2 Regularization

- Logistic regression can use L_1 and L_2 regularization to limit overfitting and produce sparse solution
- Like it happened with regression, overfitting is often associated to large weights so to limit overfitting we penalize large weights

Total cost = Measure of Fit - Magnitude of Coefficients

- Regularization
  - L_1 uses the sum of absolute values, which penalizes large weights and at the same time promotes sparse solutions
  - L_2 uses the sum of squares, which penalizes large weights

L_1 Regularization

$$\ell(\vec{w}) - \alpha||\vec{w}||_1$$

L_2 Regularization

$$\ell(\vec{w}) - \alpha||\vec{w}||_2^2$$

If α is zero, then we have no regularization

If α tends to infinity,
the solution is a zero weight vector

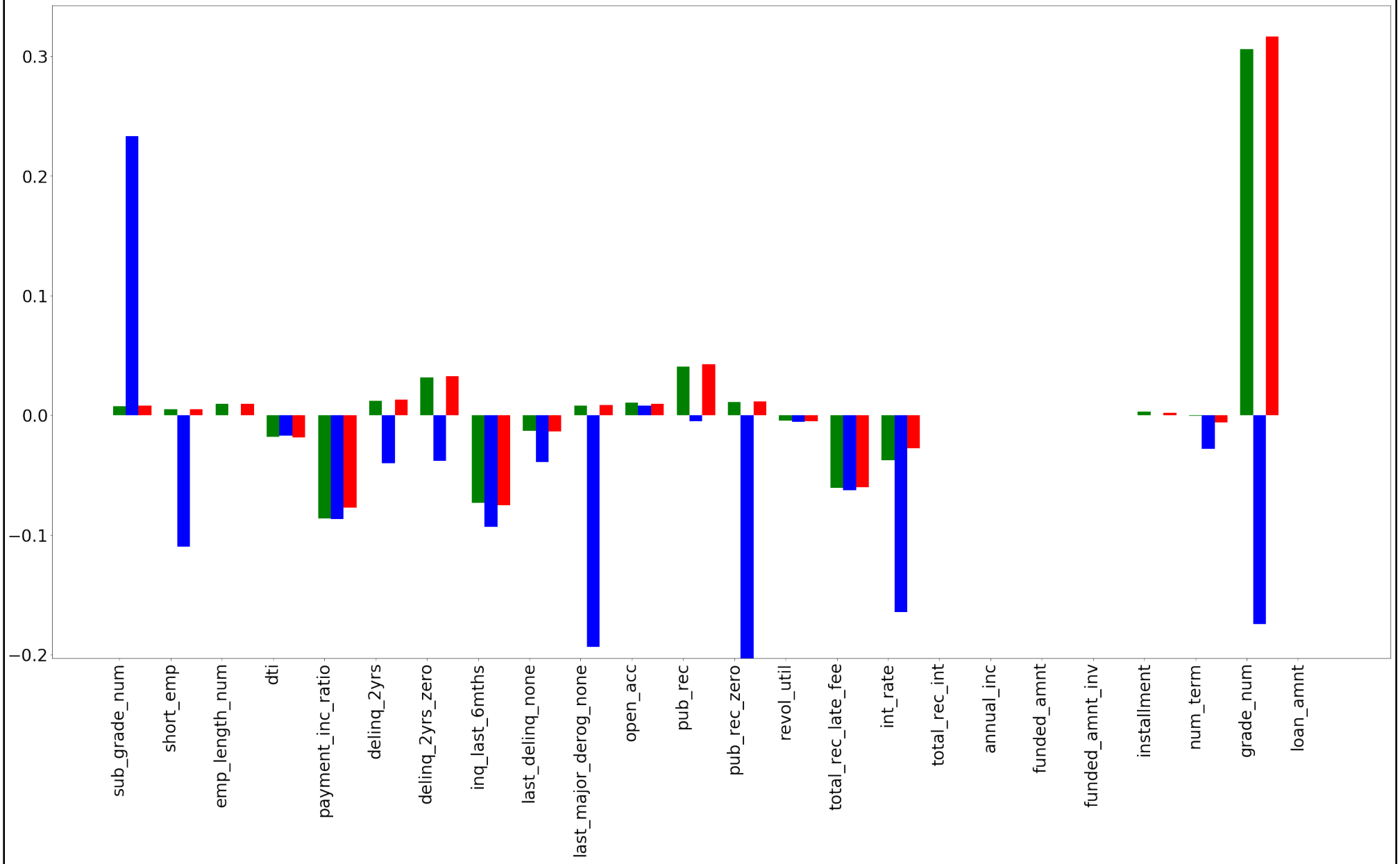
α must balance fit and the weight magnitude

Example

Safe Loans Prediction

- Given data downloaded from the Lending Club Corporation website regarding safe and risky loans, build a model to predict whether a loan should be given

- Model evaluation using 10-fold crossvalidation
  - Simple Logistic Regression μ=0.63 σ=0.05
  - Logistic Regression (L1) μ=0.63 σ=0.07
  - Logistic Regression (L2) μ=0.64 σ=0.05



Weights of the simple logistic model, the L1 regularized model, and the L2 regularized model.

From now on we are going to use k-fold crossvalidation a lot to score models

k-fold crossvalidation generates k separate models

Which one should be deployed?

K-fold crossvalidation provides an evaluation of model performance on unknown data

Its output it's the evaluation, not the model!

The model should be obtained by running the algorithm on the entire dataset!

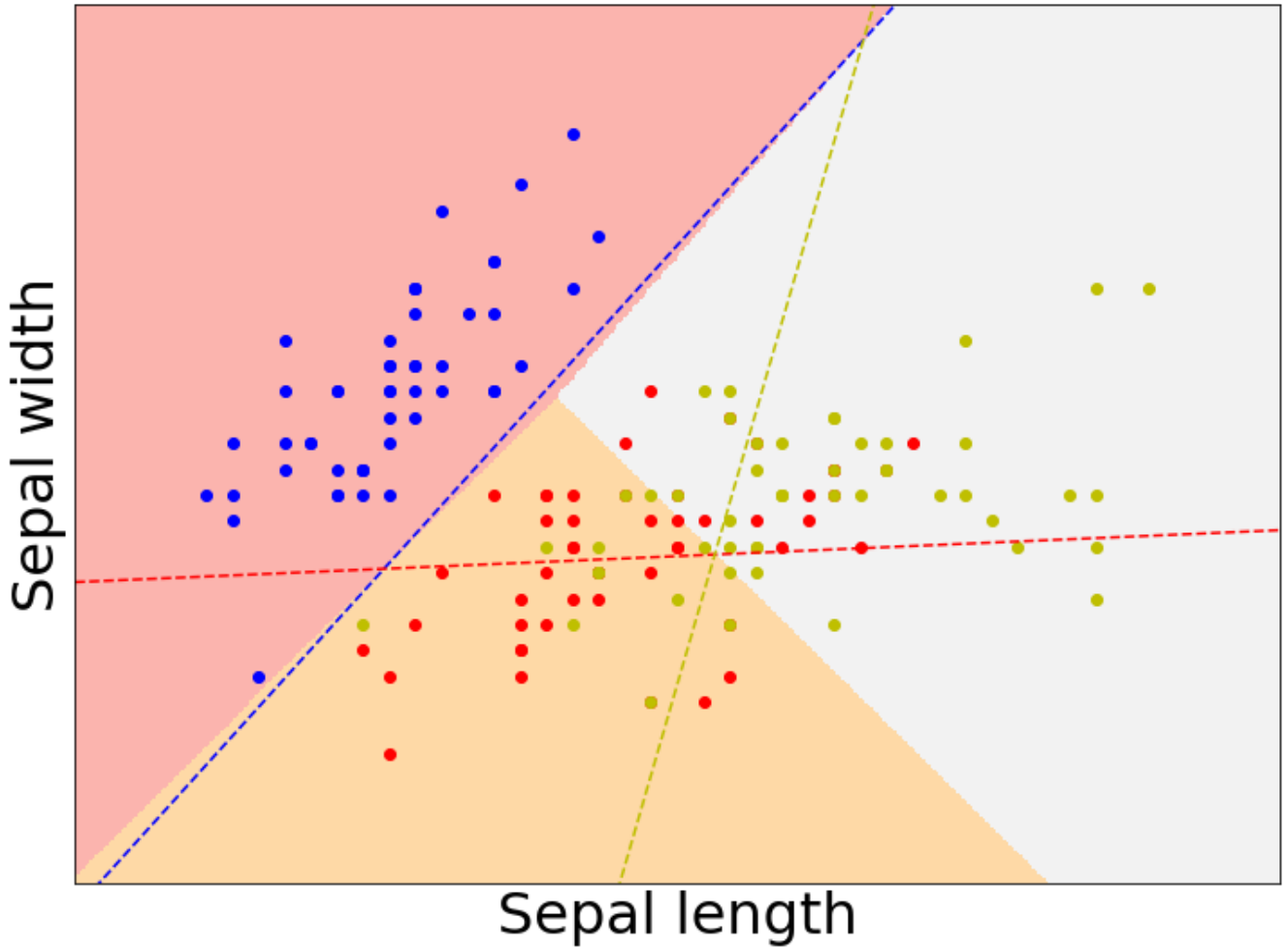
Multiclass Classification

Logistic regression assumes that
there are only two class values (e.g., +1/-1)

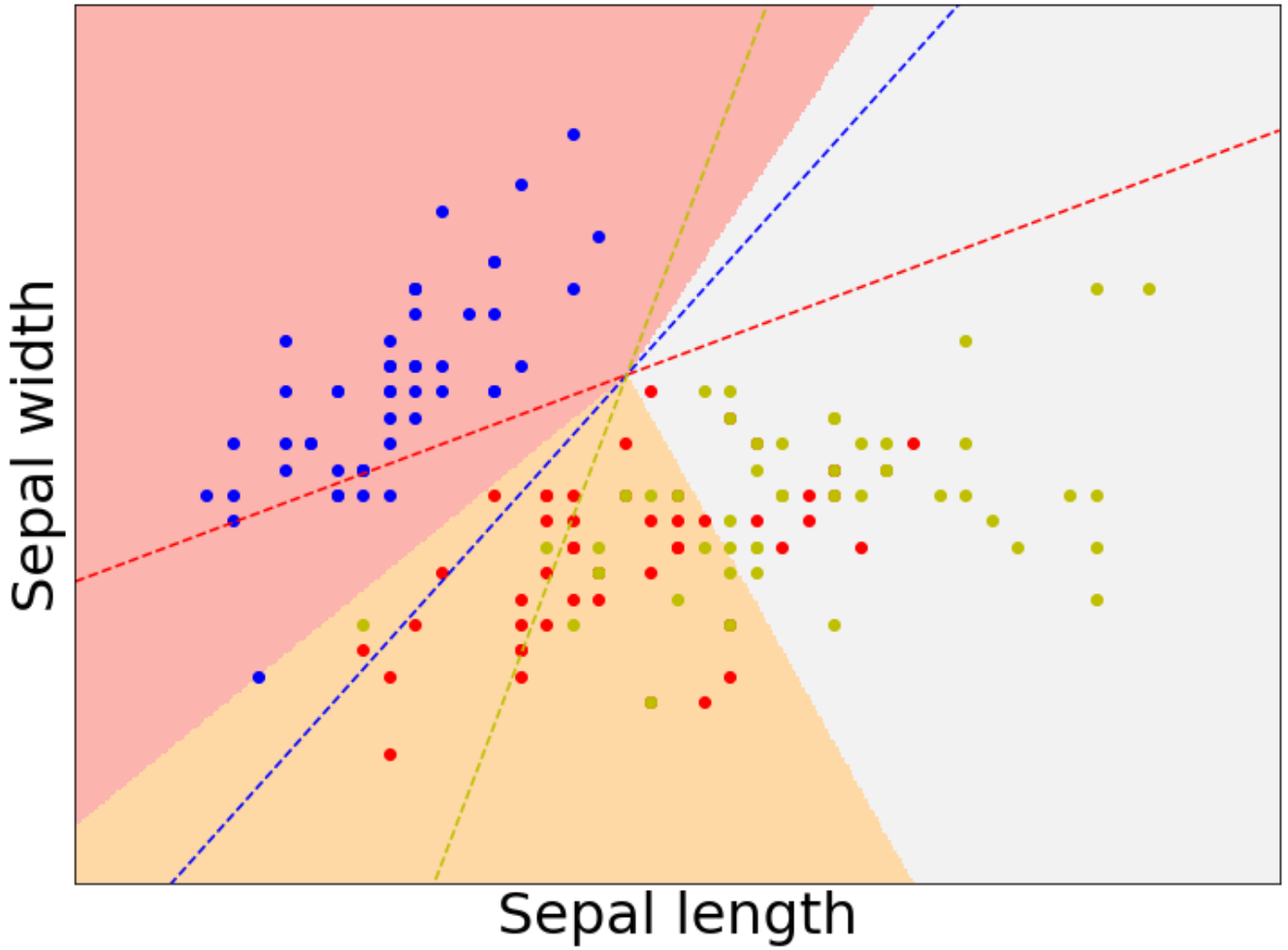
What if we have more?
(e.g., none, soft, hard or Setosa/Versicolour/Virginica)

One Versus the Rest Evaluation

- For each class, it creates one classifier that predicts the target class against all the others
- For instance, given three classes A, B, C, it computes three models: one that predicts A against B and C, one that predicts B against A and C, and one that predicts C against A and B
- Then, given an example, all the three classifiers are applied and the label with the highest probability is returned
- Alternative approaches include the minimization of loss based on the multinomial loss fit across the entire probability distribution



One Versus Rest multiclass model using the Iris dataset



Multinomial multiclass model using the Iris dataset

Categorical Attributes

So far we applied Logistic Regression (and regression) to numerical variables

What if the variables are categorical?

The Weather Dataset

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

One Hot Encoding

Map each categorical attribute with
n values into n binary 0/1 variables

Each one describing one
specific attribute values

For example, attribute Outlook is replaced by three
binary variables Sunny, Overcast, and Rainy

overcast	rainy	sunny
0	0	1
0	0	1
1	0	0
0	1	0
0	1	0
0	1	0
1	0	0
0	0	1
0	0	1
0	1	0
0	0	1
1	0	0
1	0	0
0	1	0

One Hot Encoding for the Outlook attribute.

Evaluation & Credidibility Issues

- What measure should we use? (accuracy might not be enough!)
- How reliable are the predicted results?
- How much should we believe in what was learned?
  - Error on the training data is not a good indicator of performance on future data
  - The classifier was computed from the very same training data, any estimate based on that data will be optimistic.
  - In addition, new data will probably not be exactly the same as the training data!

How to evaluate the performance of a model?

How to obtain reliable estimates?

How to compare the relative performance among competing models?

Given two equally performing models, which one should we prefer?

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the performance of competing models?
- Model Selection
  - Which model should we prefer?

How to evaluate the performance of a model? (Metrics for Performance Evaluation)

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
- Confusion Matrix:

		PREDICTED CLASS	
		Yes	No
TRUE CLASS	Yes	a (TP)	b (FN)
	No	c (FP)	d (TN)

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation

		PREDICTED CLASS	
		Yes	No
ACTUAL CLASS	Yes	a (TP)	b (FN)
	No	c (FP)	d (TN)

- Most widely-used metric:

$$Accuracy = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Sometimes Accuracy is not Enough

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10
- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
- Accuracy is misleading because model does not detect any class 1 example

Cost Matrix

		PREDICTED CLASS	
		Yes	No
ACTUAL CLASS	Yes	C(TP)	C(FN)
	No	C(FP)	C(TN)

C(x): Cost of misclassifying examples of type x

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
	C(.)	+	-
ACTUAL CLASS	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	150	40
	-	60	250

Model M_2	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	250	45
	-	5	200

Accuracy = 80%

Cost = 3910

Accuracy = 90%

Cost = 4255

Cost-Based Model Evaluation

- Predictive accuracy in this case is defined as

$$accuracy = \frac{number\ of\ correct\ predictions}{number\ of\ applications}$$

- True Positives, true negatives
  - Safe and risky loans predicted as safe and risky respectively
- False Positive Errors
  - We accept a risky loan which we predicted was safe
  - We are likely not to get the money back
    (let's say on average 30000 euros)
- False Negative Errors
  - We don't give a safe loan since we predicted it was risky
  - We will loose the interest money
    (let's say on average 10000 euros

Cost-Based Model Evaluation

- Two models, similar performance but different types of errors
- First Model
  - 1576 false positives and 1224 false negatives
  - Total cost is 59525443
- Second Model
  - 1407 false positives and 1093 false negatives
  - Total cost is 53147717 (more than 6 millions saved)
- What if we can change the way our model makes mistakes?
  - 1093 false positives and 1407 false negatives
  - Total cost becomes 46852283 (more than 12 millions saved)

Costs can be used to evaluate
an existing classification models

Or can be used by the algorithm
to guide the search for the model

Some algorithms can use the cost matrix to
build the model (e.g., decision trees)

Precision & Recall

- Alternatives to accuracy, introduced in the area of information retrieval and search engine
- Precision
  - Focuses on the percentage of examples that have been classified positive examples and are actually positive
  - In the information retrieval context represents the percentage of actually good documents that have been shown as a result.
- Recall
  - Focuses on the percentage of positively classified examples with respect to the number of existing good documents
  - In the information retrieval context, recall represents the percentage of good documents shown with respect to the existing ones.

Cost-Sensitive Measures

$$Precision(p) = \frac{TP}{TP+FP} = \frac{a}{a+c}$$

The higher the precision, the lower the FPs

$$Recall(r) = \frac{TP}{TP+FN} = \frac{a}{a+b}$$

Recall

The higher the precision, the lower the FNs

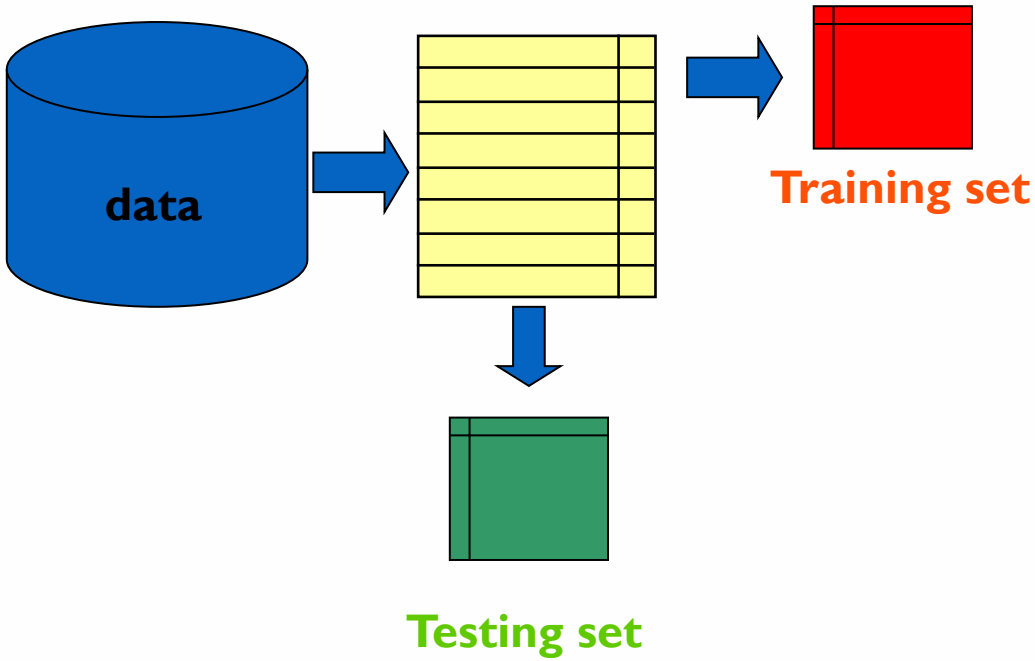
$$F1-measure = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

The higher the F1, the lower the FPs & FNs

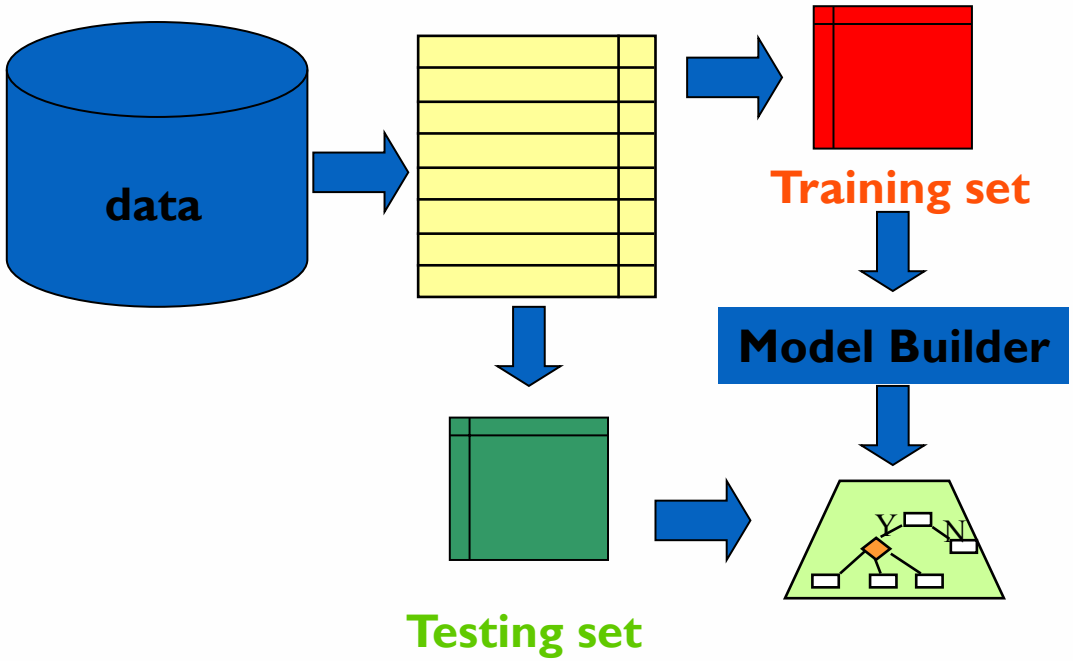
- Precision is biased towards C(TP) & C(FP),
- Recall is biased towards C(TP) & C(FN)
- F1-measure is biased towards all except C(TN),
  it is high when both precision and recall are reasonably high

How to obtain reliable estimates?
(Methods for Performance Evaluation)

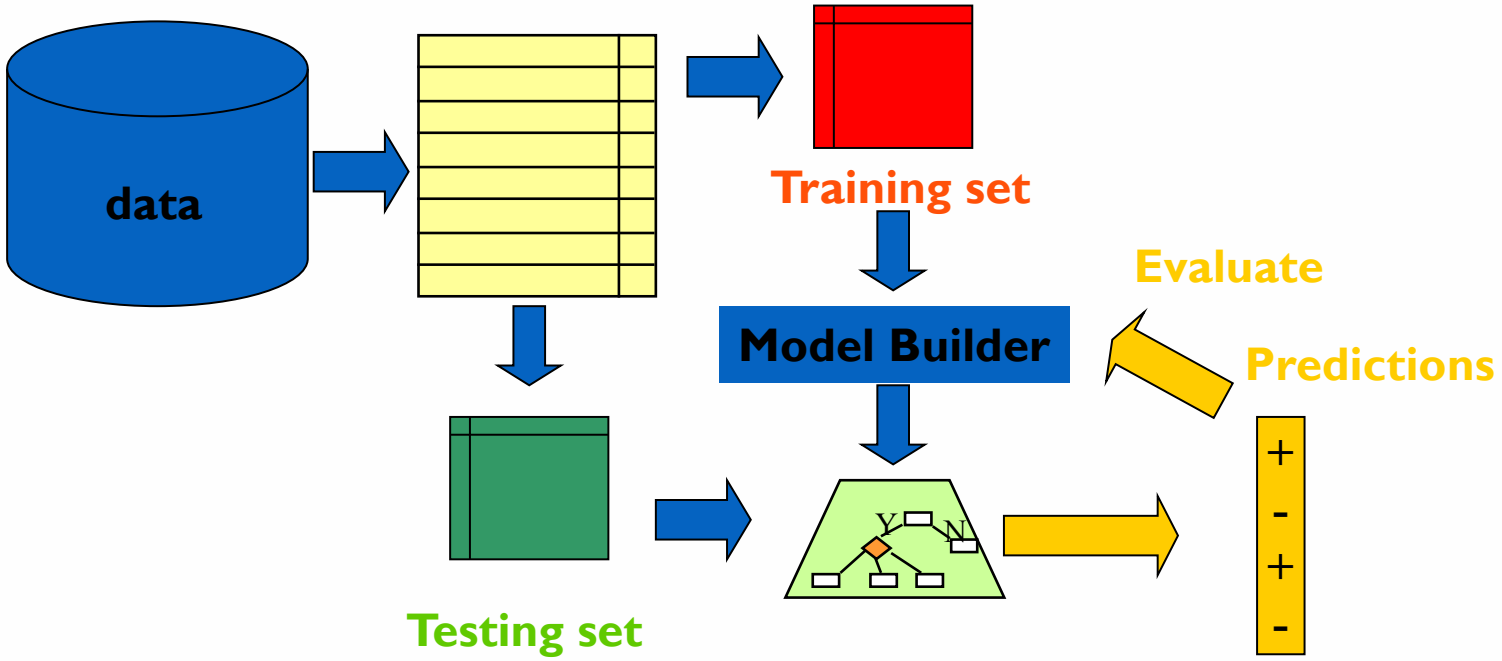
Classification Step 1: Split the Data Between Train and Test Sets



Classification Step 2: Compute a Model using the Data in the Training Set



Classification Step 3: Evaluate the Model using the Test Set



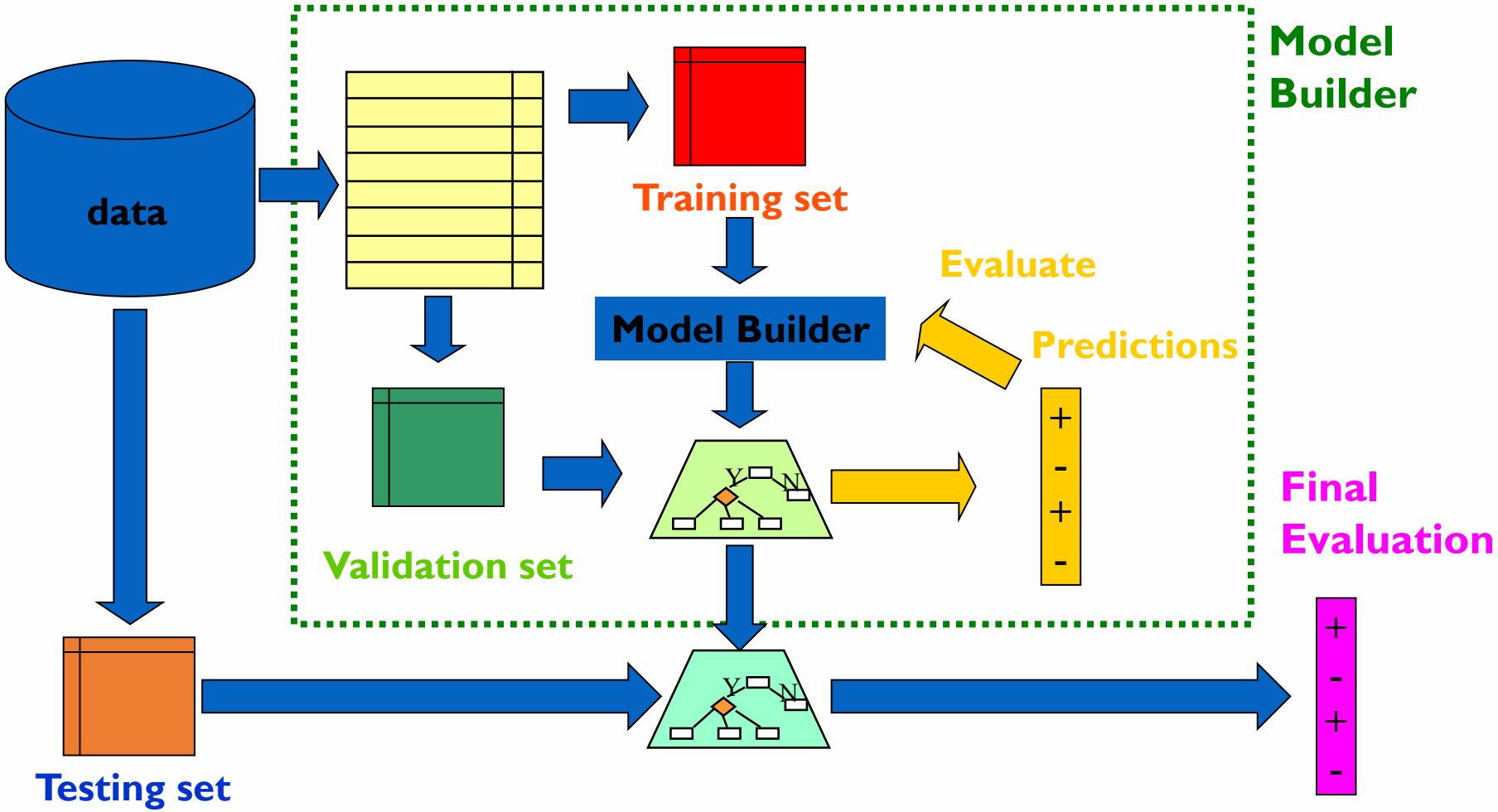
Note on Parameter Tuning

- It is important that the test data is not used in any way to create the classifier
- Some learning schemes operate in two stages:
  - Stage 1: builds the basic structure
  - Stage 2: optimizes parameter settings
- The test data can't be used for parameter tuning!
- Proper procedure uses three sets: training data, validation data, and test data. Validation data is used to optimize parameters

Making the most of the data

- Once evaluation is complete, all the data can be used to build the final classifier
- Generally, the larger the training data the better the classifier (but returns diminish)
- The larger the test data the more accurate the error estimate

The Full Process: Training, Validation, and Testing



Methods for Performance Evaluation

- How to obtain a reliable estimate of performance?

- Performance of a model may depend on other factors besides the learning algorithm:
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets

Methods of Estimation

- Holdout
  - Reserve ½ for training and ½ for testing
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
  - Repeated holdout
- Cross validation
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n
- Stratified sampling
  - oversampling vs undersampling
- Bootstrap
  - Sampling with replacement

Holdout Evaluation & "Small" Datasets

- Reserves a certain amount for testing and uses the remainder for training, typically,
  - Reserve ½ for training and ½ for testing
  - Reserve 2/3 for training and 1/3 for testing
- For small or "unbalanced" datasets, samples might not be representative
- For instance, it might generate training or testing datasets with few or none instances of some classes
- Stratified sampling
  - Makes sure that each class is represented with approximately equal proportions in both subsets

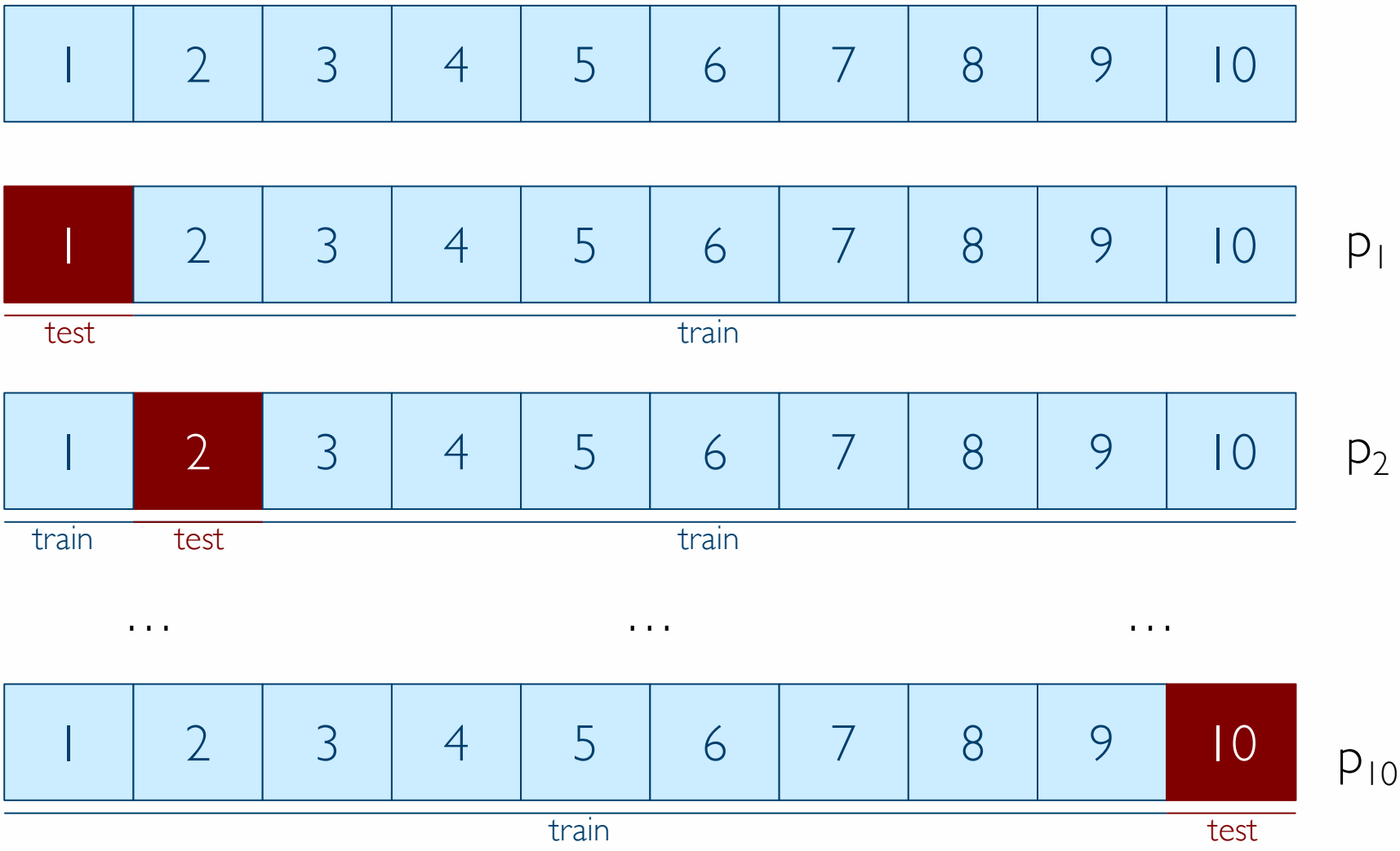
Repeated Holdout

- Holdout estimate can be made more reliable by repeating the process with different subsamples
- In each iteration, a certain proportion is randomly selected for training (possibly with stratification)
- The error rates on the different iterations are averaged to yield an overall error rate
- Still not optimum since the different test sets overlap

Cross-Validation

- First step
  - Data is split into k subsets of equal size
- Second step
  - Each subset in turn is used for testing and the remainder for training
- This is called k-fold cross-validation and avoids overlapping test sets
- Often the subsets are stratified before cross-validation is performed
- The error estimates are averaged to yield an overall error estimate

Ten-fold Crossvalidation



The final performance is computed as the average p_i

Cross-Validation

- Standard method for evaluation stratified ten-fold cross-validation
- Why ten? Extensive experiments have shown that this is the best choice to get an accurate estimate
- Stratification reduces the estimate's variance
- Even better: repeated stratified cross-validation
- E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)
- Other approaches appear to be robust, e.g., 5x2 crossvalidation

Leave-One-Out Cross-Validation

- It is a particular form of cross-validation
  - Set number of folds to number of training instances
  - I.e., for n training instances, build classifier n times
- Makes best use of the data
- Involves no random subsampling
- Computationally expensive

Leave-One-Out and Stratification

- Disadvantage of Leave-One-Out is that: stratification is not possible
- It guarantees a non-stratified sample because there is only one instance in the test set!
- Extreme example: random dataset split equally into two classes
  - Best inducer predicts majority class
  - 50% accuracy on fresh data
  - Leave-One-Out-CV estimate is 100% error!

Bootstraping

- Cross-validation uses sampling without replacement
  - The same instance, once selected, can not be selected again for a particular training/test set
- Bootstrap uses sampling with replacement
  - Sample a dataset of n instances n times with replacement to form a new dataset of n instances
  - Use this data as the training set
  - Use the instances from the original dataset that don't occur in the new training set for testing

The 0.632 Bootstrap

- An instance has a probability of 1–1/n of not being picked
- Thus its probability of ending up in the test data is:

$$\left(1-\frac{1}{N}\right)^{N} \approx e^{-1} = 0.368$$

- This means the training data will contain approximately 63.2% of the instances

Estimating Error using Bootstrap

- The error estimate on the test data will be very pessimistic, since training was on just around 63% of the instances
- Therefore, we compute the overall error **ε** by combining error on the train and test as,

$$\varepsilon = 0.632 \times \varepsilon_{test} + 0.368 \times \varepsilon_{train}$$

- The training error gets less weight than the error on the test data
- Repeat process several times with different replacement samples; average the results

How to compare the relative performance among competing models? (Model Comparison)

How to Compare the Performance of Two Models?

- Two models,
  - Model M1: accuracy = 85%, tested on 30 instances
  - Model M2: accuracy = 75%, tested on 5000 instances
- Can we say M1 is better than M2?
- How much confidence can we place on accuracy of M1 and M2?
- Can the difference in performance measure be explained as a result of random fluctuations in the test set?

Comparing Data Mining Schemes

- Which of two learning schemes performs better?
- The answer is obviously domain-dependent
- The typical approach is to compare 10-fold cross-validation (CV) estimates using the Student's t-test
- Student's t-test tells whether the means of two samples are significantly different
- Use a paired t-test when the individual samples are paired, that is, when the same CV is applied twice. Use unpaired t-test otherwise

Confidence Interval for Accuracy

- Prediction can be regarded as a Bernoulli trial with two possible outcomes, correct or wrong
- A collection of Bernoulli trials has a Binomial distribution
- Given,
  - The number of correct test predictions S
  - The number of test instances N,
  - The accuracy f= S/N
- Can we predict the true accuracy of the model from accuracy f?

Mean and Variance

- Assume that the accuracy f comes from a Bernoulli distribution with mean p and variance p(1-p)
- When N is large enough, f follows a normal distribution with mean p and variance p(1-p)/N
- c% confidence level interval [-z ≤ X ≤z] for a random variable of mean 0 is given by

$$P(-z \leq X \leq z) = c$$

- With a symmetric distribution, we have

$$P(-z \leq X \leq z) = 1 - 2 \times P(X \geq z)$$

Confidence Interval for Accuracy

- For instance, if X has mean 0 and standard deviation 1, the confidence interval for c% is defined by the z values,

Pr[X ≥z]	z
0.1%	3.09
0.5%	2.58
1%	2.33
5%	1.65
10%	1.28
20%	0.84
25%	0.69
40%	0.25

- Thus, for the confidence interval for 90% we have,

$$P(-1.65 \leq X \leq 1.65) = 0.90$$

- To compute the confidence interval for the expected accuracy f we need to,
  - Normalize it to a distribution with mean 0 and standard deviation 1
  - Compute the confidence interval
  - Remap it to the actual distribution

Confidence Interval for Accuracy

- First, transform f using,

$$\frac{f-p}{\sqrt{p(1-p)/N}}$$

- Compute the confidence interval,

$$P(-z \leq \frac{f-p}{\sqrt{p(1-p)/N}} \leq z) = c$$

- Solve it for p,

$$p = \frac{f + \frac{z^2}{2N} \pm \sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}$$

Examples

- When f = 75%, N=1000, c=80% (z=1.28), the confidence interval for the actual performance is [0.732, 0.767]
- When f = 75%, N=100, c=80% (z=1.28), the confidence interval for the actual performance is [0.691, 0.801]
- Note that the assumption about f being normally distributed only applies for large values of N (at least 30)
- When f = 75%, N=10, c=80% (z=1.28), the confidence interval for the actual performance is [0.549, 0.881]

William Gosset (1876-1937)

- Obtained a post as a chemist in the Guinness brewery in Dublin in 1899.
- Invented the t-test to handle small samples for quality control in brewing
- Wrote under the name "Student"
- Unpaired and paired two-sample *t*-test
- Unequal/equal sample size
- Unequal/equal sample variance

Comparing the Performance of Two Models (Paired)

- Fix a significance level
- If a difference is significant at the c% level, there is a (100-c)% chance that the true means differ
- Divide the significance level by two because the test is two-tailed
- Look up the value for z that corresponds to c/2
- Compute the value of t based on the observed performance estimates for the schemes being compared
- If t is less than –z or greater than z then the difference is statistically significant

"paired" when the estimates are from the same datasets

"unpaired" when the estimates are from different datasets

How do we know that the difference in performance is not just due to change?

We computes the odds of it!

Apply the t-test and compute the p-value

The p-value represents the probability that the reported difference is due by chance

In repeated experiments at this sample size, how often would you see such a difference assuming that there is no difference between the two groups?

Multiple Testing

- Say that you perform a statistical test with a 0.05 threshold, but you repeat the test on twenty different observations.
- For example, you want to compare the performance of several classification algorithms
- Assume that all of the observations are explainable by the null hypothesis
- What is the chance that at least one of the observations will receive a p-value less than 0.05?

Multiple Testing - Example

- Say that you perform a statistical test with a 0.05 threshold, but you repeat the test on 20 different observations. Assuming that all of the observations are explainable by the null hypothesis, what is the chance that at least one of the observations will receive a p-value less than 0.05?

- P(making a mistake) = 0.05
- P(not making a mistake) = 0.95
- P(not making any mistake) = 0.95^{20} = 0.358
- P(making at least one mistake) = 1 - 0.358 = 0.642
- There is a 64.2% chance of making at least one mistake.

Bonferroni Correction

- Assume that individual tests are independent.
- Divide the desired p-value threshold by the number of tests performed.
- Example
  - We now have, the threshold set to 0.05/20 = 0.0025.
  - P(making a mistake) = 0.0025
  - P(not making a mistake) = 0.9975
  - P(not making any mistake) = 0.9975^{20} = 0.9512
  - P(making at least one mistake) = 1 - 0.9512 = 0.0488

Non Parametric Tests

- Do not make any assumption about the distribution of the variable in the population
- Mann-Whitney U Test
  - Nonparametric equivalent of the independent t-test
- Wilcoxon matched-pairs signed rank test
  - Used to compare two related groups

Mann Whitney U Test

- To compute the Mann Whitney U:
  - Rank the scores in both groups (together) from highest to lowest.
  - Sum the ranks of the scores for each group.
  - The sum of ranks for each group are used to make the statistical comparison.

Income	Rank	No Income	Rank
25	12	27	10
32	5	19	17
36	3	16	20
40	1	33	4
22	14	30	7
37	2	17	19
20	16	21	15
18	18	23	13
31	6	26	11
29	8	28	9
	85		125

Wilcoxon Matched-pairs Signed Rank test

- Step 1
  - Rank the data of both the groups in ascending order
  - If any values are equal average their ranks
- Step 2
  - Add up the ranks in group with smaller sample size
  - If the two groups are of the same size either one may be picked
  - T= sum of ranks in group with smaller sample size
- Step 3
  - Compare this sum with the critical ranges given in table
  - Look up the rows corresponding to the sample sizes of the two groups
  - A range will be shown for the 5% significance level

Non-smokers (n=15)		Heavy smokers (n=14)	
Birth wt (Kg)	Rank	Birth wt (Kg)	Rank
3.99	27	3.18	7
3.79	24	2.84	5
3.60*	18	2.90	6
3.73	22	3.27	11
3.21	8	3.85	26
3.60*	18	3.52	14
4.08	28	3.23	9
3.61	20	2.76	4
3.83	25	3.60*	18
3.31	12	3.75	23
4.13	29	3.59	16
3.26	10	3.63	21
3.54	15	2.38	2
3.51	13	2.34	1
2.71	3		
	Sum=272		Sum=163

* 17, 18 & 19 are tied hence the ranks are averaged

	Nonparametric tests		Parametric tests
	Nominal data	Ordinal data	Ordinal, interval, ratio data
One group	Chi square goodness of fit	Wilcoxon signed rank test	One group t-test
Two unrelated groups	Chi square	Wilcoxon rank sum test, Mann-Whitney test	Student's t-test
Two related groups	McNemar's test	Wilcoxon signed rank test	Paired Student's t-test
K-unrelated groups	Chi square test	Kruskal -Wallis one-way analysis of variance	ANOVA
K-related groups		Friedman matched samples	ANOVA with repeated measurements

Probabilistic Classifiers

Probabilistic Classifiers & Classification Thresholds

- Up to now we used logistic regression to predict classifier labels, however, logistic regression returns a probability

$$P(y_i|\vec{x_i})$$

- Given an example x_i its predicted class is the label with the largest probability so it is equivalent to using a threshold of 0.5 to decide which class to assign to an example

- However, we can use a different threshold and for instance label as positive only examples we return a 1 only when $P(+1|x)>0.75$

- This would label as positive only cases for which we are more confident that should be labeled as positive.

How the Classification Threshold Influence Precision and Recall?

- Suppose we use a near one threshold to classify positive examples
- Then, we will classify as positives only examples for which we are very confident (this is a pessimistic classifier)
- Precision will be high
  - In fact, we are not likely produce false positives
- Recall will be low
  - In fact, we are likely to produce more false negatives

How the Classification Threshold Influence Precision and Recall?

- Suppose we use a near zero threshold to classify positive examples
- Then, we will classify everything as positives
  (this is an optimistic classifier)
- Precision will be low as we are going to generate the maximum number of false positives (everything is positive!)
- Recall will be high since by classifying everything as positive we are going to generate the maximum number of false negatives

We can use the threshold to optimize our precision and recall

a higher the threshold, increases precision and lower recall

a lower threshold, decreases precision and increase recall

Run the notebook dedicated
to the precision-recall tradeoff

Precision-Recall Tradeoff Example

- In the notebook, a simple logistic regression model applied to the loans data returns the confusion matrix below, corresponding to a precision of 0.82 and recall of 0.99

	Classified -1	Classified +1
Labeled -1	1212	21910
Labeled +1	1057	98283

- By increasing the classification threshold for positive (+1) examples to 0.75 we obtained a new confusion matrix with precision of 0.86 and a recall of 0.99

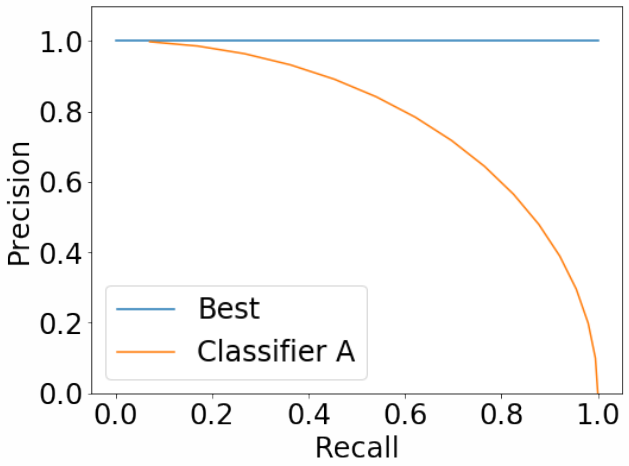
	Classified -1	Classified +1
Labeled -1	10632	12490
Labeled +1	20106	79234

- Overall, we reduced the number of false positives (we did not accept risky loans and we were better at identifying risky loans)

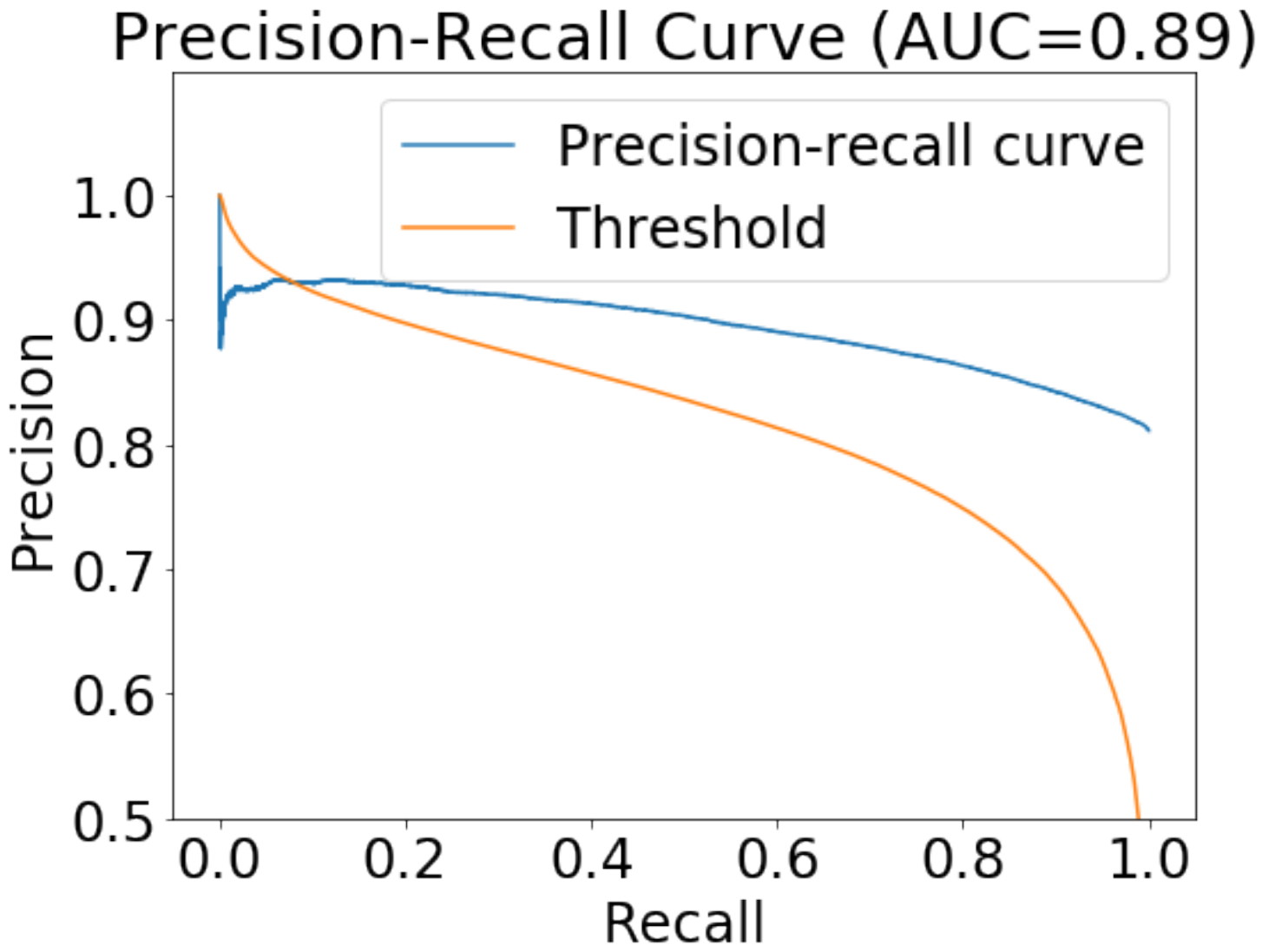
we can explore this tradeoffs
using precision-recall curves

Precision-Recall Curves

- Plot precision as a function of recall for varying threshold values



- The best classifier would be the one that has always a precision equal to one (but never happens)
- More in general classifiers will show of different shapes
- How to decide among more classifiers?
  - Use the area under the curve (the nearer to one, the better)
  - Use F1 measure



Precision-recall curve for the loan dataset (run the python notebook for details)

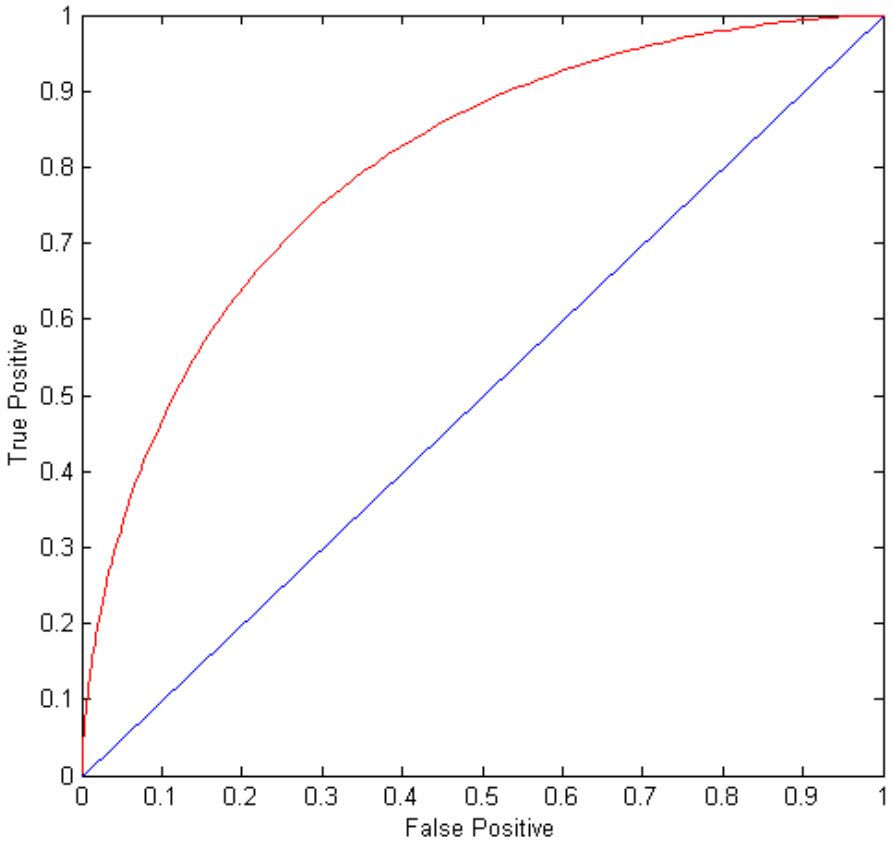
Receiver Operating Characteristic (ROC) Curves

ROC Curve (Receiver Operating Characteristic)

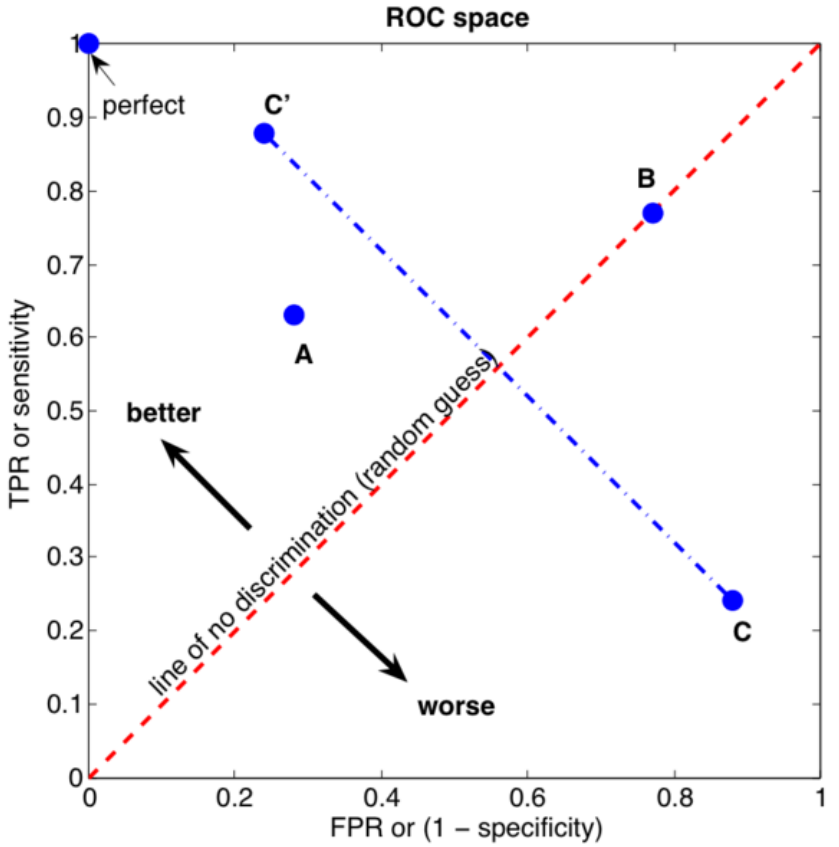
- Developed in 1950s for signal detection theory to analyze signals
- Plot the True Positive Rate (TPR=TP/(TP+FN)) against the False Positive Rate (FPR=FP/(TN+FP))
- Performance of each classifier represented as a point on the ROC curve
- Changing the classification threshold, sample distribution or cost matrix changes the location of the point

ROC Curve

- (FPR, TPR)
  - (0,0): declare everything to be negative class
  - (1,1): declare everything to be positive class
  - (0,1): ideal
- Diagonal line:
  - Random guessing
  - Below diagonal line, prediction is opposite of the true class



ROC Curve



A

TP=63	FP=28	91
FN=37	TN=72	109
100	100	200

TPR = 0.63

FPR = 0.28

ACC = 0.68

B

TP=77	FP=77	154
FN=23	TN=23	46
100	100	200

TPR = 0.77

FPR = 0.77

ACC = 0.50

C

TP=24	FP=88	112
FN=76	TN=12	88
100	100	200

TPR = 0.24

FPR = 0.88

ACC = 0.18

C'

TP=88	FP=24	112
FN=12	TN=76	88
100	100	200

TPR = 0.88

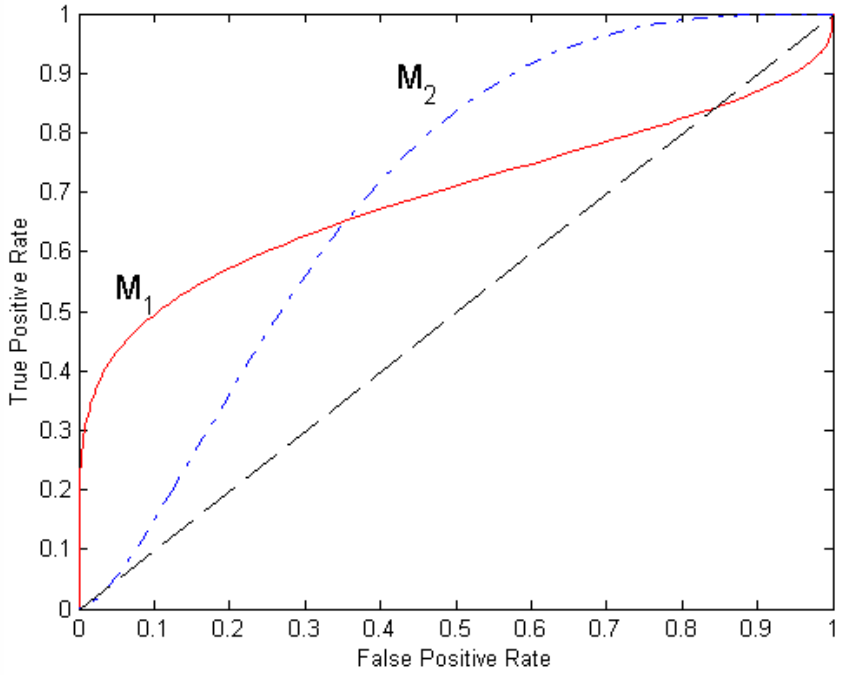
FPR = 0.24

ACC = 0.82

Using ROC for Model Comparison

- No model consistently outperform the other
- M_1 is better for small FPR
- M_2 is better for large FPR

- Area Under the ROC curve
- Ideal, area = 1
- Random guess, area = 0.5



Which model should we prefer?
(Model selection)

Model Selection Criteria

- Model selection criteria attempt to find a good compromise between:
  - The complexity of a model
  - Its prediction accuracy on the training data
- Reasoning: a good model is a simple model that achieves high accuracy on the given data
- Also known as Occam's Razor : the best theory is the smallest one that describes all the facts

William of Ockham, born in the village of Ockham in Surrey (England) about 1285, was the most influential philosopher of the 14th century and a controversial theologian.

The MDL Principle

- MDL stands for minimum description length
- The description length is defined as:

space required to describe a theory

+

space required to describe the theory's mistakes

- In our case the theory is the classifier and the mistakes are the errors on the training data

- We seek a classifier with Minimum Description Language

MDL and compression

- MDL principle relates to data compression
- The best theory is the one that compresses the data the most
- I.e. to compress a dataset we generate a model and then store the model and its mistakes
- We need to compute
  (a) size of the model, and
  (b) space needed to encode the errors
- (b) easy: use the informational loss function
- (a) need a method to encode the model

Which Classification Algorithm?

- Among the several algorithms, which one is the "best"?
  - Some algorithms have a lower computational complexity
  - Different algorithms provide different representations
  - Some algorithms allow the specification of prior knowledge
- If we are interested in the generalization performance, are there any reasons to prefer one classifier over another?
- Can we expect any classification method to be superior or inferior overall?
- According to the No Free Lunch Theorem, the answer to all these questions is known

No Free Lunch Theorem

- If the goal is to obtain good generalization performance, there are no context-independent or usage-independent reasons to favor one classification method over another
- If one algorithm seems to outperform another in a certain situation, it is a consequence of its fit to the particular problem, not the general superiority of the algorithm
- When confronting a new problem, this theorem suggests that we should focus on the aspects that matter most
  - Prior information
  - Data distribution
  - Amount of training data
  - Cost or reward
- The theorem also justifies skepticism regarding studies that "demonstrate" the overall superiority of a certain algorithm

No Free Lunch Theorem

- "[A]ll algorithms that search for an extremum of a cost [objective] function perform exactly the same, when averaged over all possible cost functions." [1]
- "[T]he average performance of any pair of algorithms across all possible problems is identical." [2]
- Wolpert, D.H., Macready, W.G. (1995), No Free Lunch Theorems for Search, Technical Report SFI-TR-95-02-010 (Santa Fe Institute).
- Wolpert, D.H., Macready, W.G. (1997), No Free Lunch Theorems for Optimization, IEEE Transactions on Evolutionary Computation 1, 67.