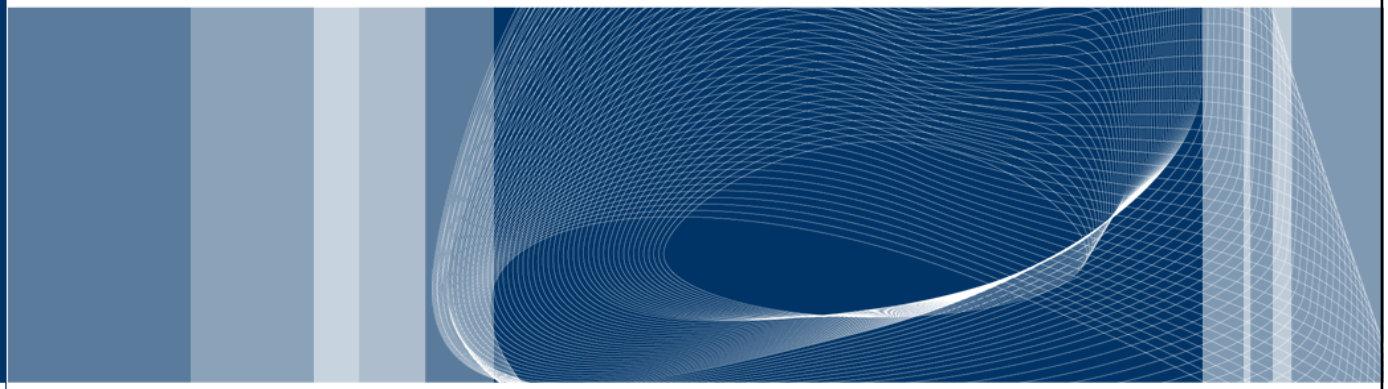
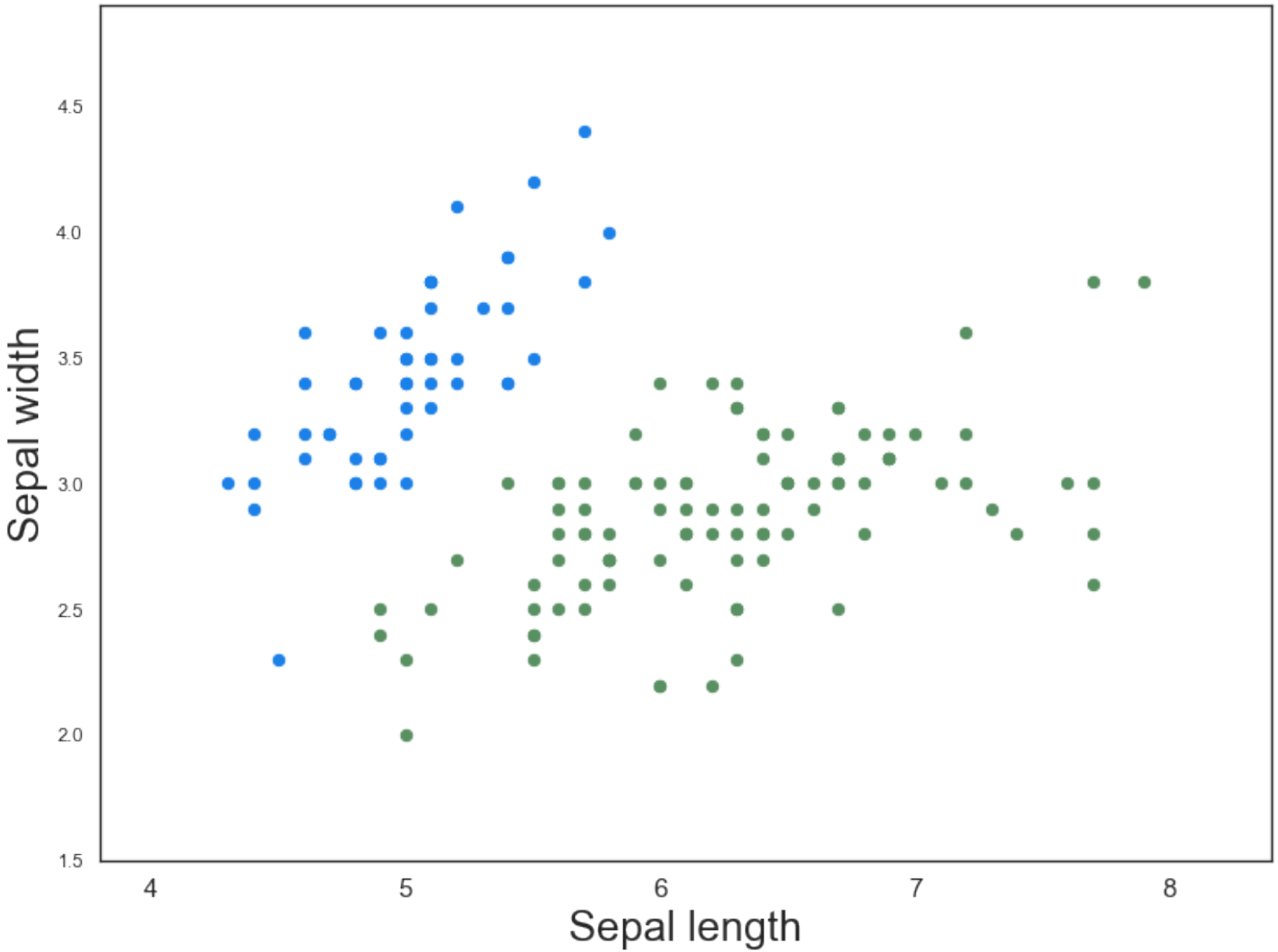
POLITECNICO DI MILANO



Classification

Data Mining and Text Mining

Prof. Pierluca Lanzi


Sepal width
4.5
4.0
3.5
3.0
2.5
2.0
1.5
4
5
6
7
8
Sepal length

Suppose you receive an unlabeled data point

How would you decide, its class?

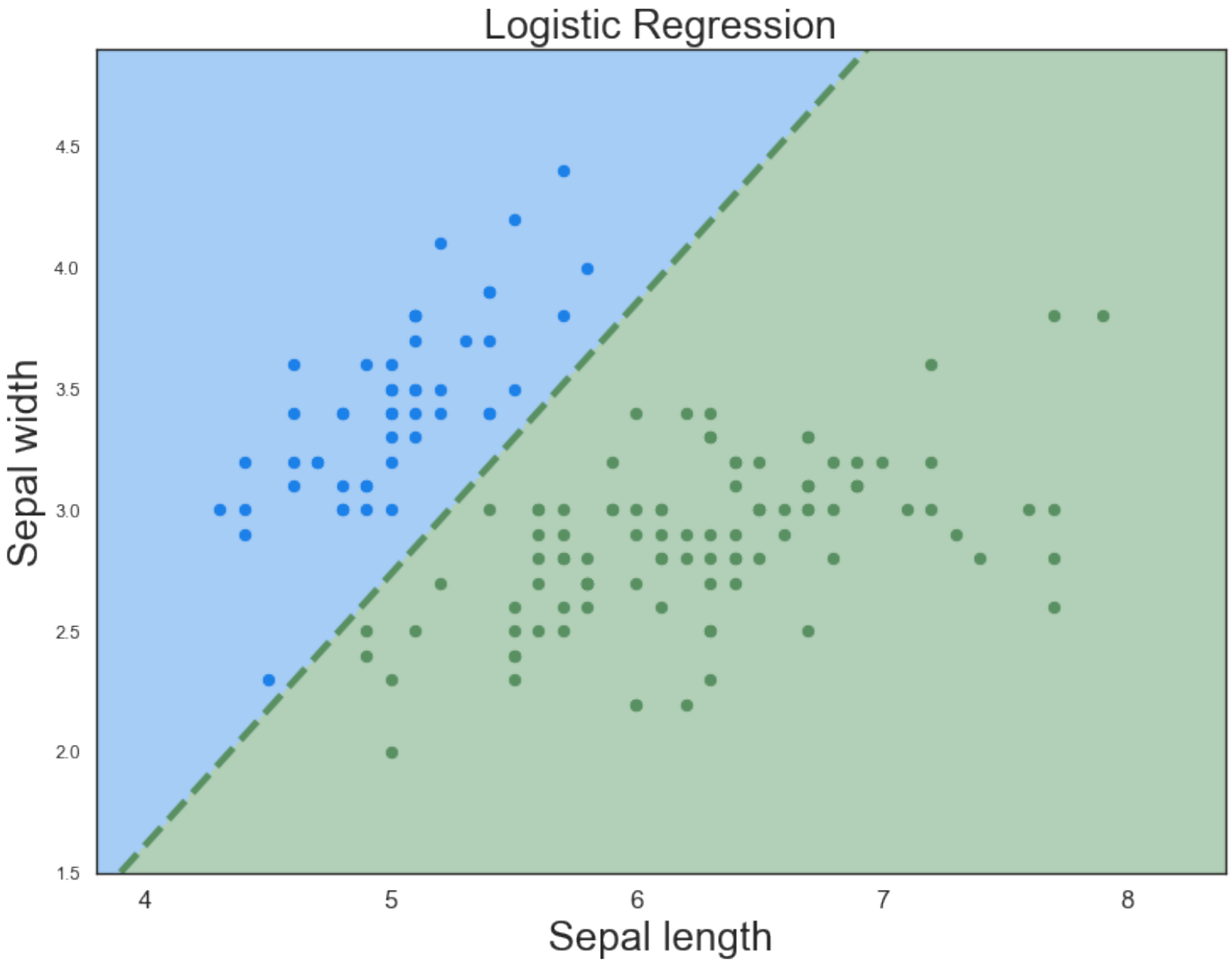
Simplest way would be the majority class

In the example, the number of green data points is larger than the blue ones …

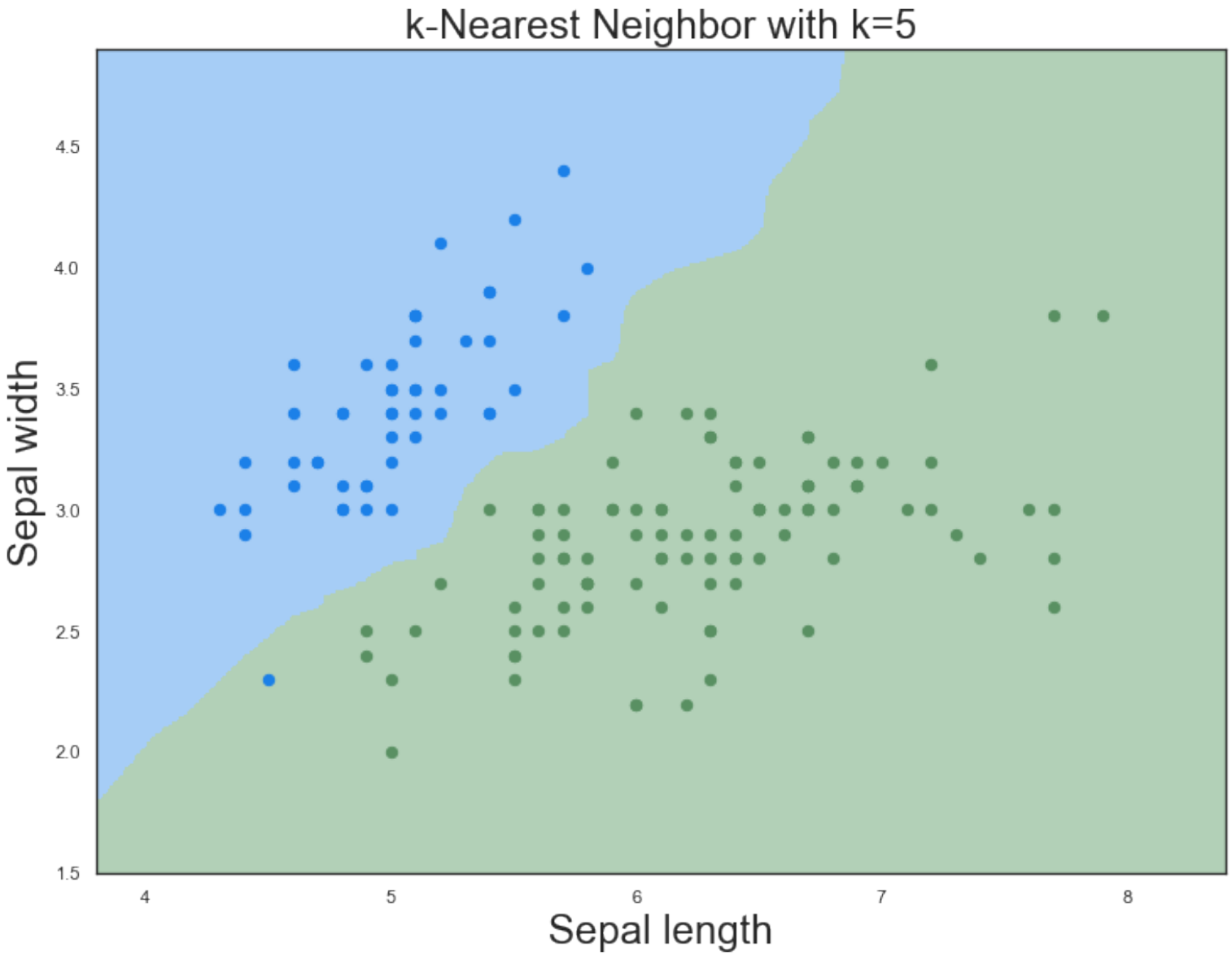
So I might decide to assign a green label to unlabel data points

This gives us a baseline performance

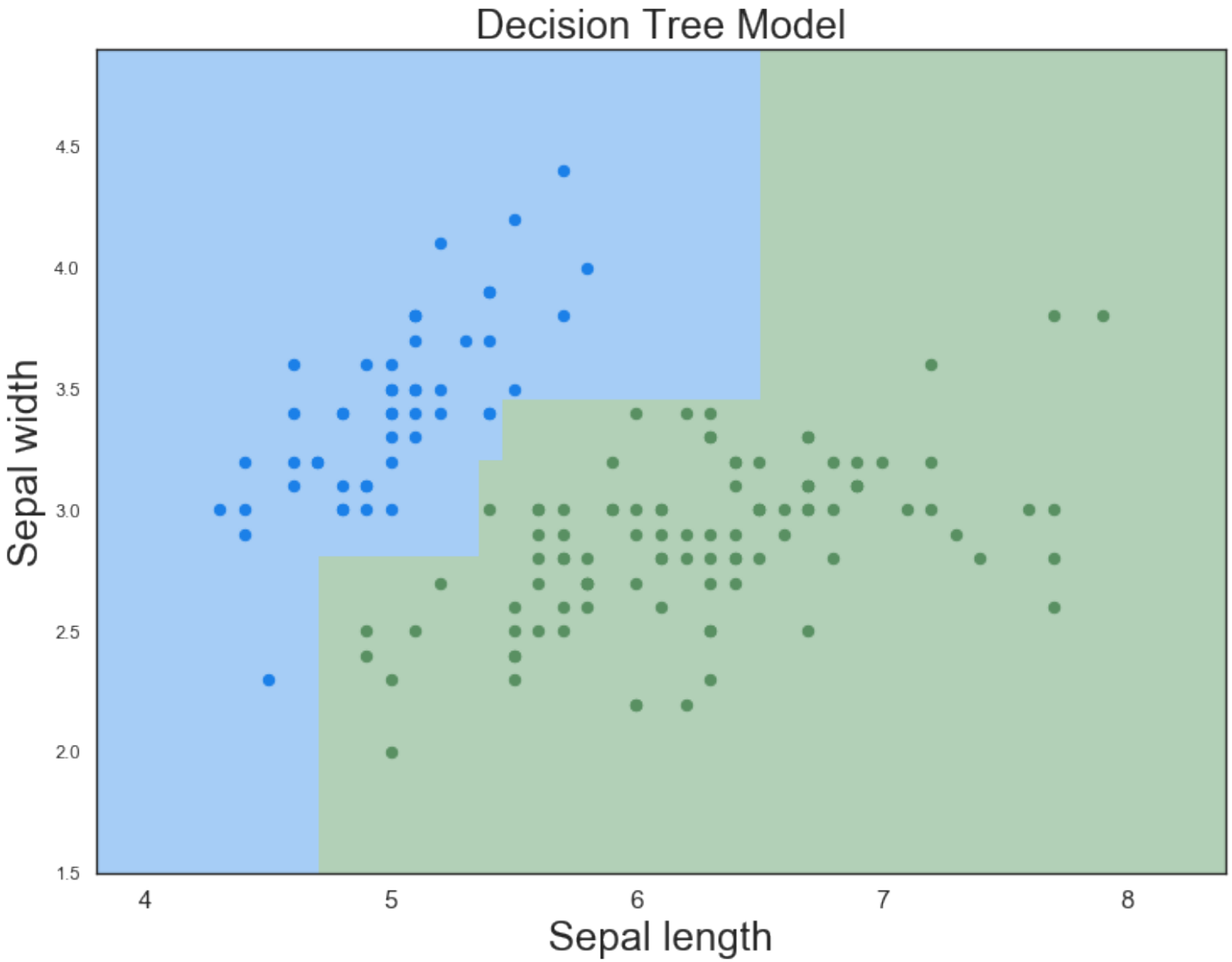
But we can do better than this …



Logistic Regression Model



K-Nearest Neighbor model with a k of 5



Decision Tree Model

As for regression, we must compute
a model using known data

And the model should 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

In classification problems …

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

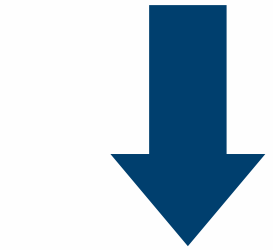
The target to predict is a label (the class variable)
(good/bad, none/soft/hard, etc.)

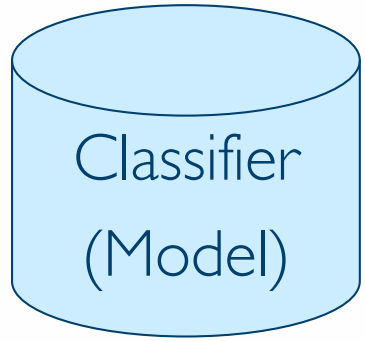
Prediction/Regression

The target to predict is numerical

classification = model building + model usage

Classification: Model Construction



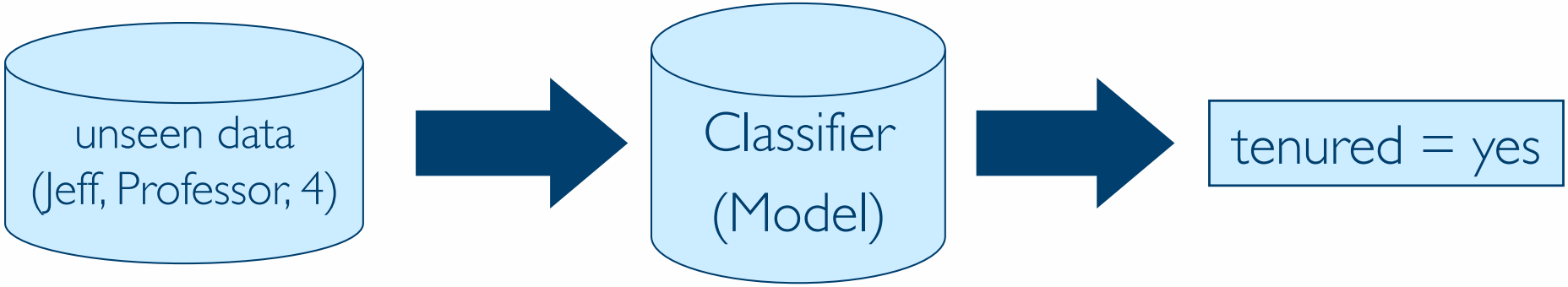


Classification Algorithm

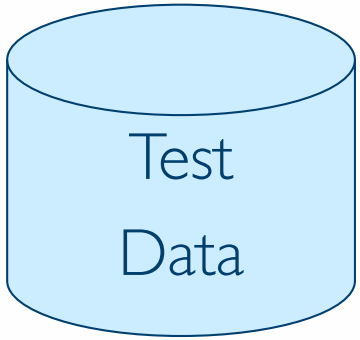
name	rank	years	tenured
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

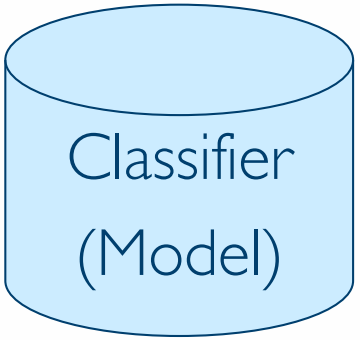
IF rank = 'professor'
OR years > 6
THEN tenured = 'yes'

Classification: Model Usage



Classification: Model Testing





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

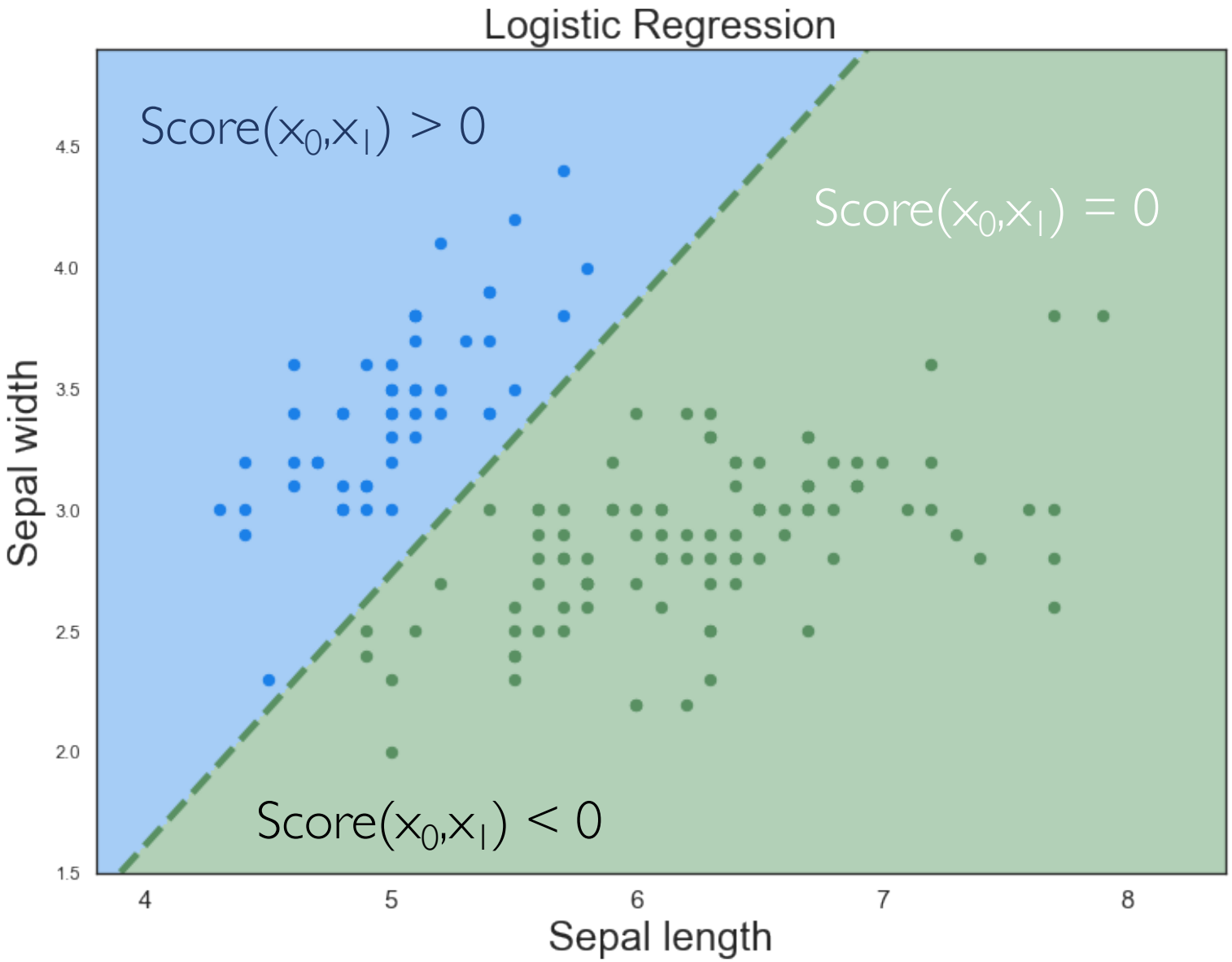
compare

predicted labels
no
yes
yes
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



Logistic Regression Model

To predict the labels we check whether Score(x_0, x_1) is positive or negative

Then, 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,

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

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

- 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,

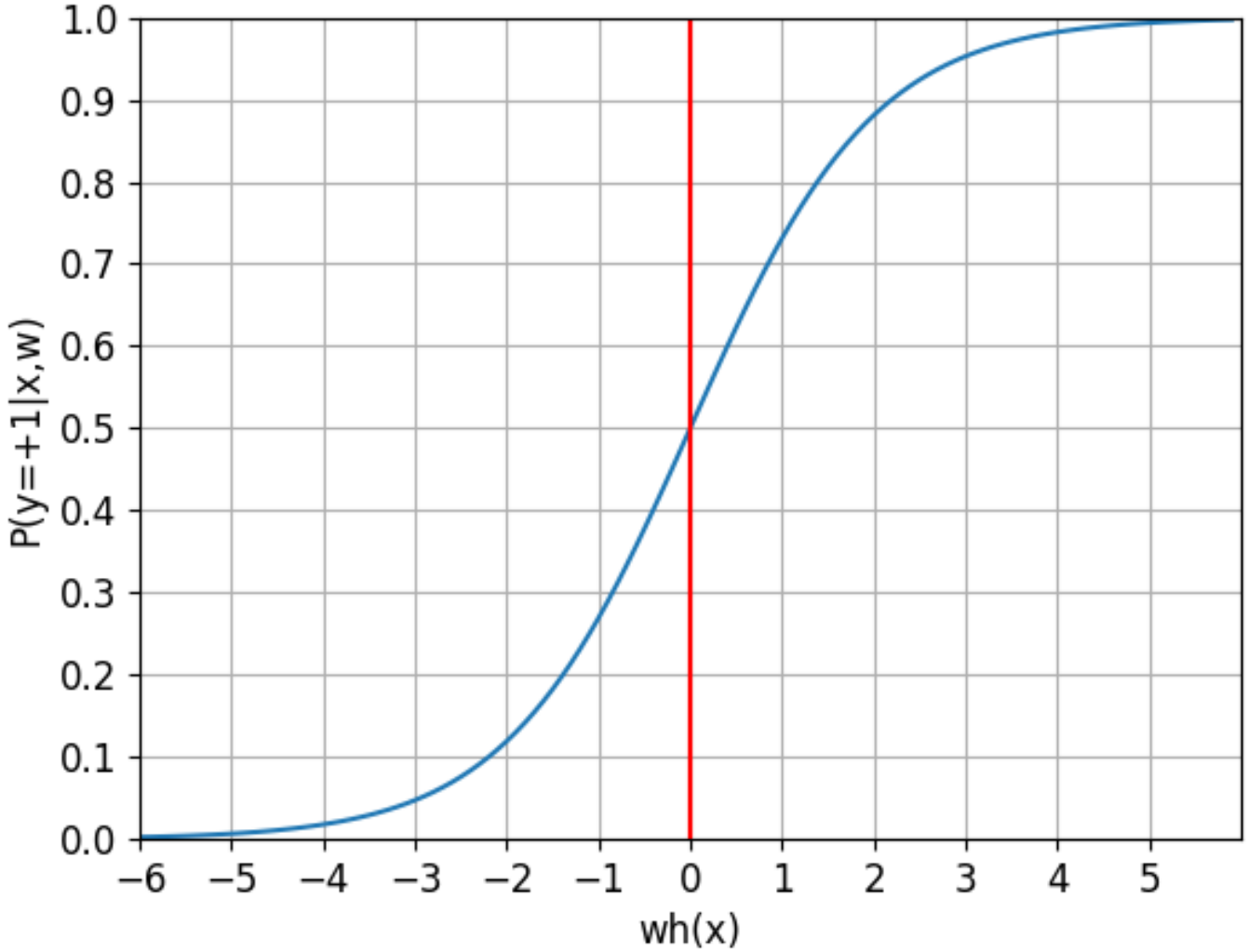
$$P(y_i|\vec{x_i})$$

- 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 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:
  - select the class with the highest probability P(Y=y|x)
  - or check if ratio of probabilities is greater than one:

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

  - choosing the positive class if it is, otherwise, the negative class
- Or equivalently check if natural log of ratio is greater than zero

$$ln(\frac{P(y=+1)|\vec{x})}{P(y=-1)|\vec{x})}) = \vec{w}h(\vec{x}) = \sum_{j=0}^{D} w_j h_j(\vec{x}) > 0$$

  - Note that we're back to a linear classification rule meaning that logistic regression is a linear classifier

Classification using Logistic Regression

- 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

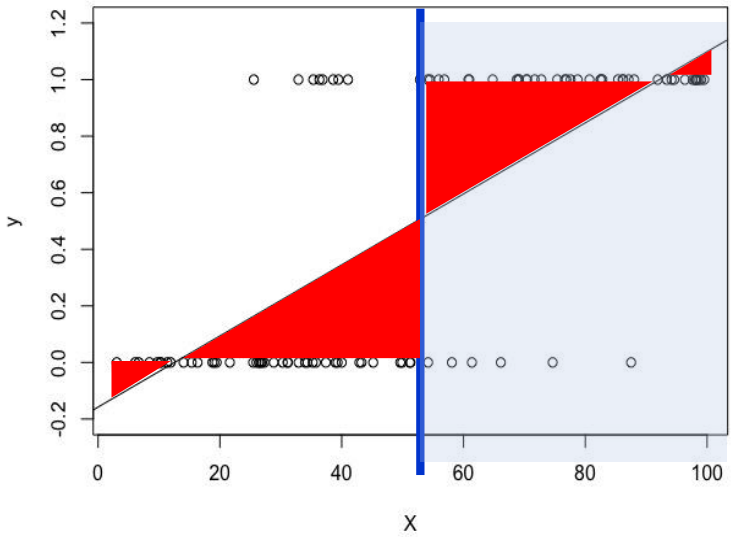
Linear vs Logistic Regression

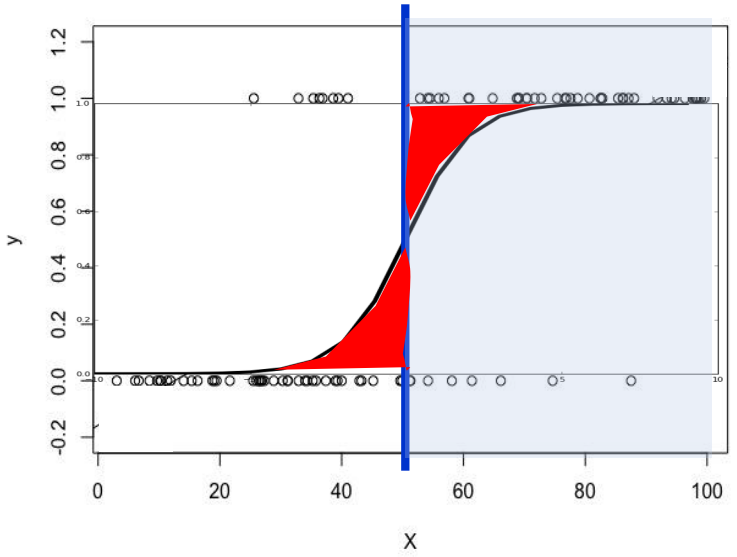
- Logistic curve fits better 0/1 data and results in better decision boundary

- Error is monotonic in distance from boundary!
  - Red areas indicative of size of error for correctly classified examples
  - Incorrectly classified instances have larger error values

- Note that decision boundary for linear regression is highly sensitive to outliers lying far from boundary

- Logistic regression not sensitive to these value





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

Overall Objective= 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

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

k-fold cross-validation generates k separate models

Which one should be deployed?

K-fold cross-validation 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!

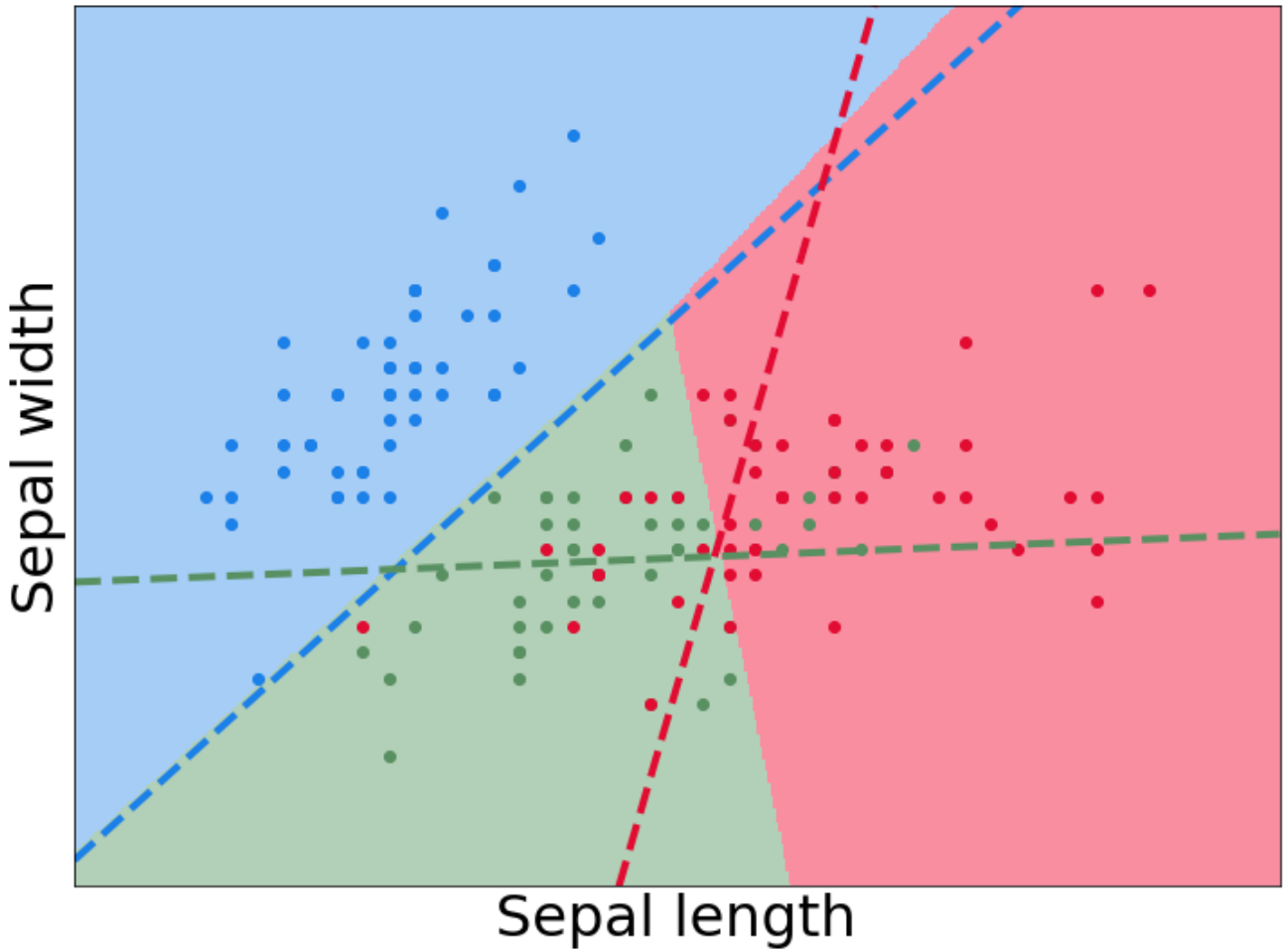
Multiclass Classification

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
- 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

Multiclass Model and loss function

- Alternative approach is to change model itself to be multi-class
- For example, use model that directly predicts 3 (or more) probabilities and predicts multinomial distribution instead of binomial distribution
- Model will have more parameters, 2 vectors for 3 classes, 3 vectors for 4, etc.
- Optimisation routine minimises loss or maximises likelihood over the multiple classes at once

Multiclass Logistic Regression

- Multinomial Logistic Regression model uses the softmax function:

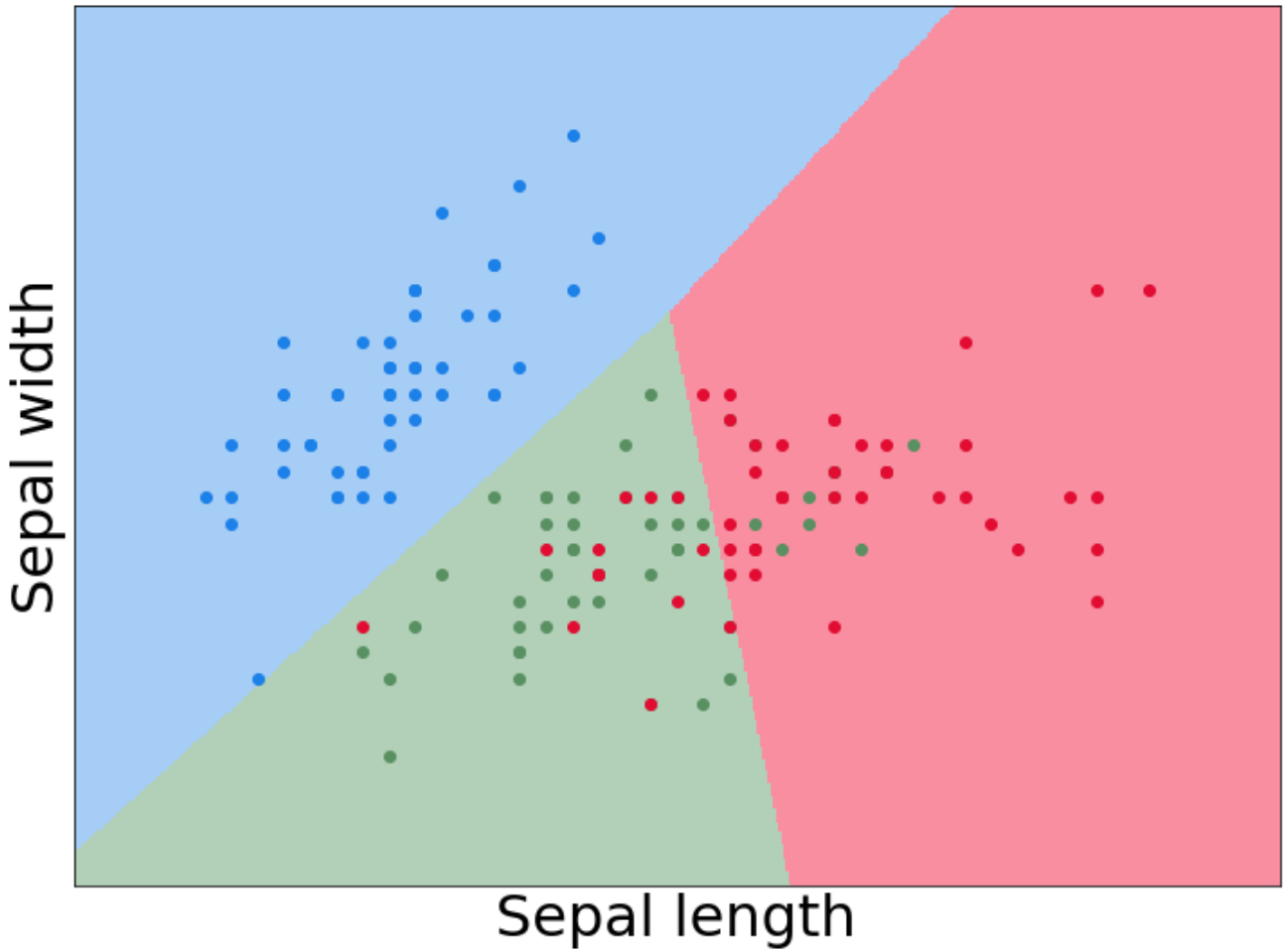
$$P(Y_i = j) = \frac{e^{\vec{w}_j h(\vec{x}_i)}}{\sum_j e^{\vec{w}_j h(\vec{x}_i)}}$$

  - using a weight vector now for each class j

- Don't actually need that many parameters
  - (model is overparameterized)
  - so can set the vector for one class to zero: $\vec{w}_C = \vec{0}$

$$P(Y_i = j) = \frac{e^{\vec{w}_j h(\vec{x}_i)}}{1 + \sum_j e^{\vec{w}_j h(\vec{x}_i)}} \quad if\ j \neq C$$

$$P(Y_i = C) = \frac{1}{1 + \sum_j e^{\vec{w}_j h(\vec{x}_i)}}$$



Multinomial multiclass model using the Iris dataset

Classification Metrics

Metrics for Performance Evaluation: Confusion Matrix

- Focus on the predictive capability of a model

- Confusion Matrix:

		PREDICTED CLASS	
		Yes	No
TRUE CLASS	Yes	TP true positives	FN false negatives
	No	FP false positives	TN true negatives

Metrics for Performance Evaluation: Accuracy

		PREDICTED CLASS	
		Yes	No
ACTUAL CLASS	Yes	#TP	#FN
	No	#FP	#TN

- Most widely-used metric:

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

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

Metrics for Performance Evaluation: Cost

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

Cost(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

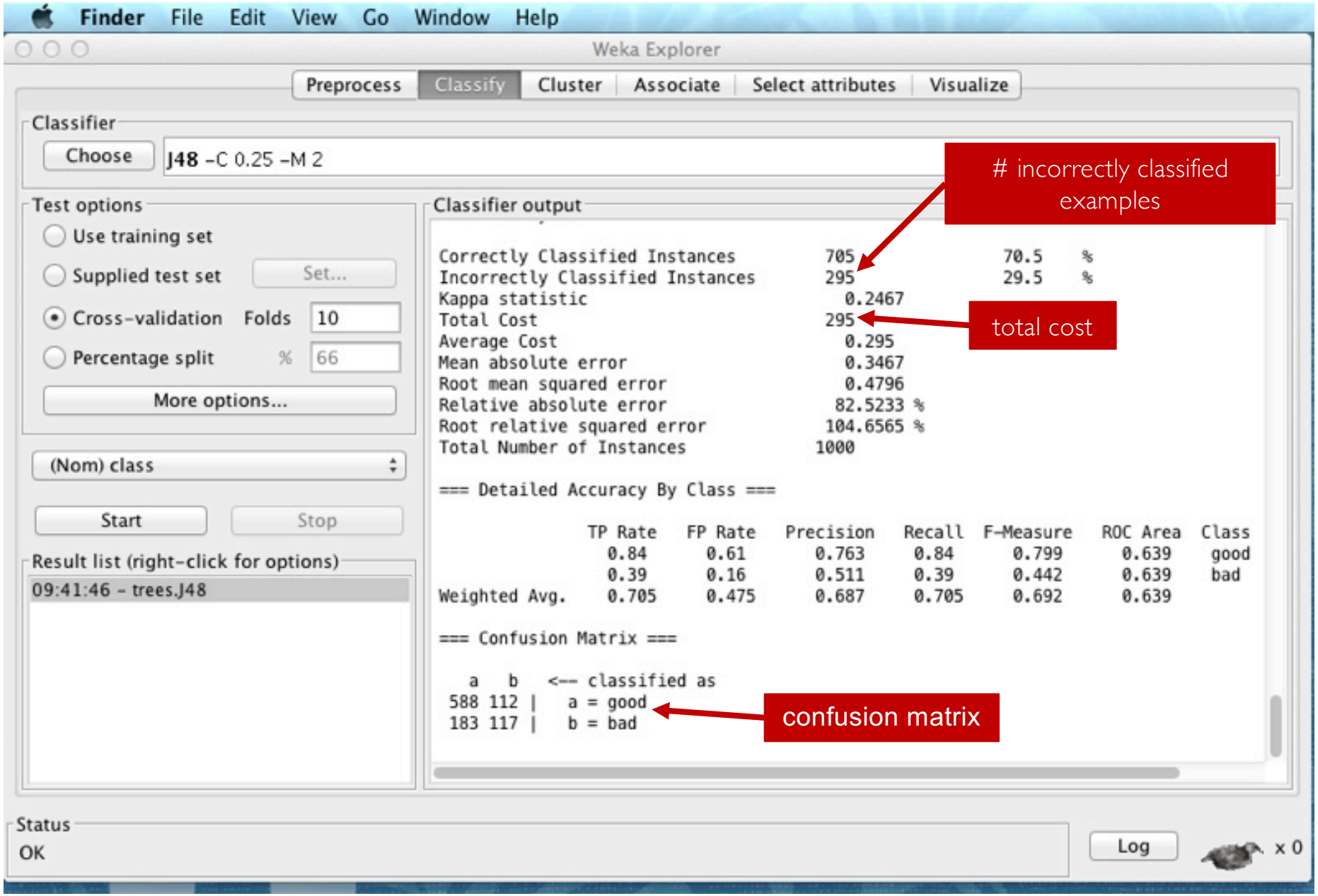
Accuracy = 90%

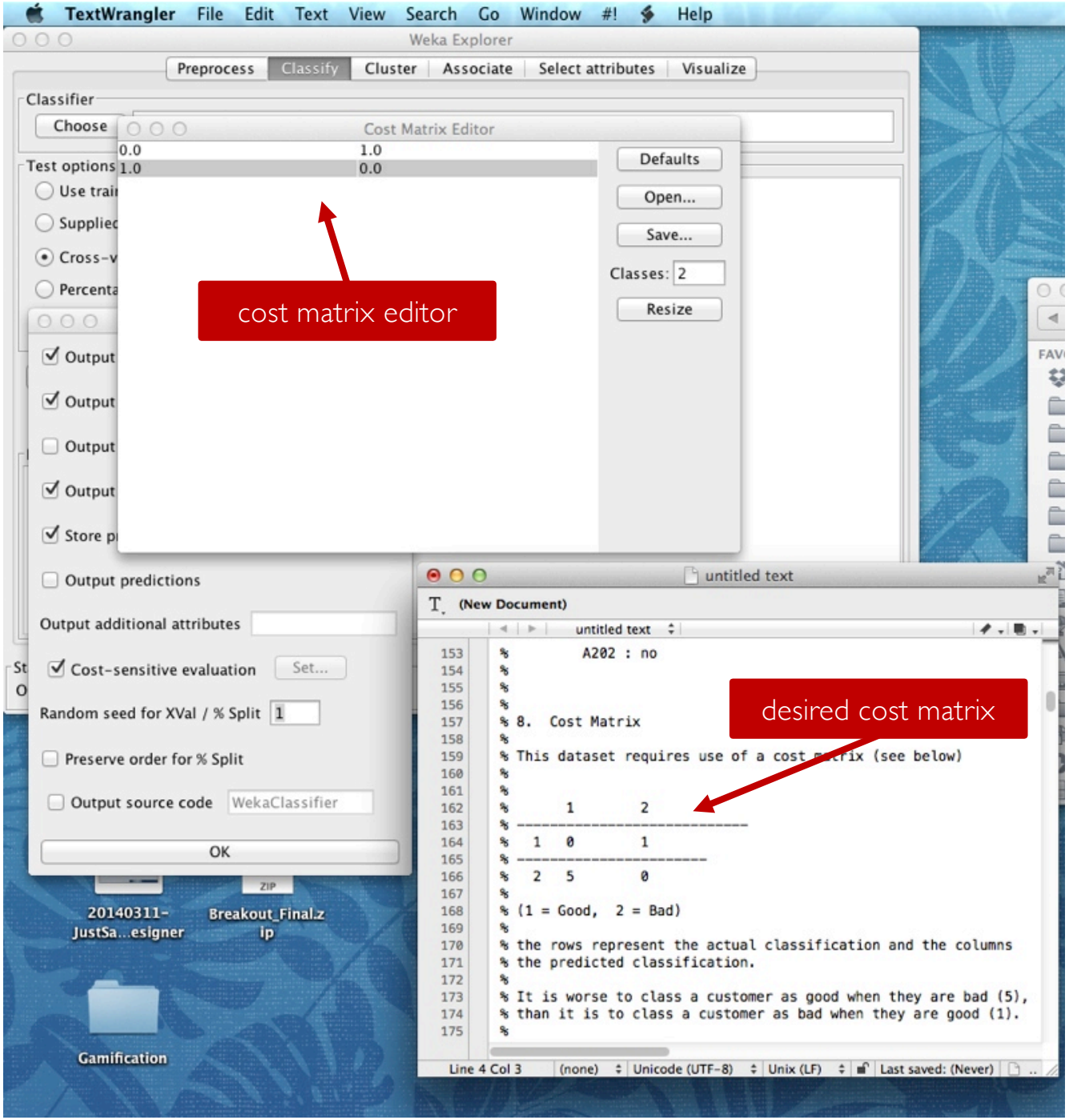
Cost = 4255

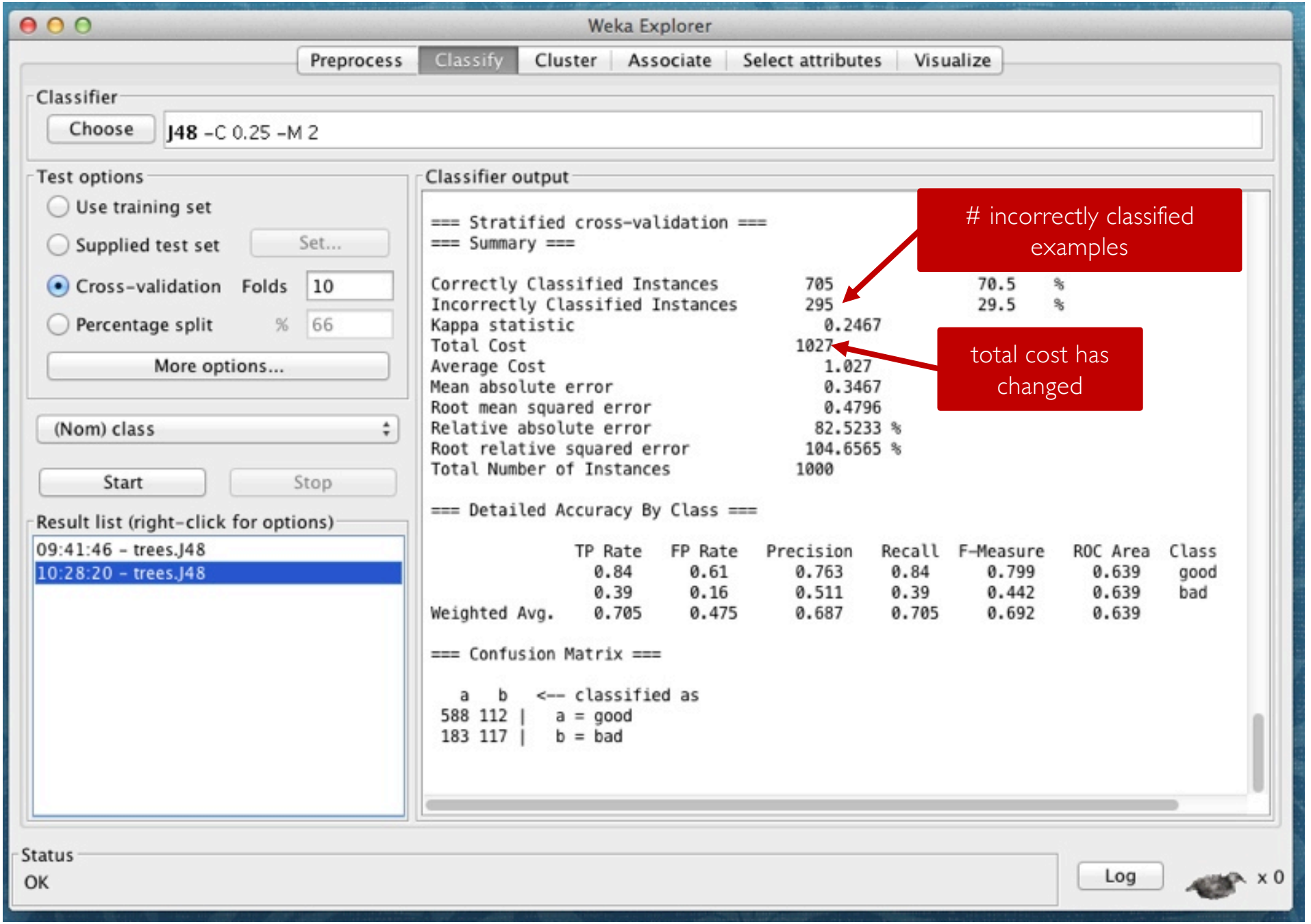
Costs can be used to evaluate
an existing classification models

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)


Finder File Edit View Go Window Help
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation Folds 10
Percentage split % 66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
09:41:46 - trees.J48
Classifier output
Correctly Classified Instances 705 70.5 %
Incorrectly Classified Instances 295 29.5 %
Kappa statistic 0.2467
Total Cost 295
Average Cost 0.295
Mean absolute error 0.3467
Root mean squared error 0.4796
Relative absolute error 82.5233 %
Root relative squared error 104.6565 %
Total Number of Instances 1000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
0.84 0.61 0.763 0.84 0.799 0.639 good
0.39 0.16 0.511 0.39 0.442 0.639 bad
Weighted Avg. 0.705 0.475 0.687 0.705 0.692 0.639
=== Confusion Matrix ===
a b <-- classified as
588 112 | a = good
183 117 | b = bad
incorrectly classified examples
total cost
confusion matrix
Status
OK
Log
x 0


TextWrangler File Edit Text View Search Go Window #! Help
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier
Choose
Test options
Cost Matrix Editor
0.0 1.0
1.0 0.0
Defaults
Open...
Save...
Classes: 2
Resize
cost matrix editor
Output predictions
Output additional attributes
Cost-sensitive evaluation
Set...
Random seed for XVal / % Split 1
Preserve order for % Split
Output source code WekaClassifier
OK
untitled text
(New Document)
% A202 : no
% 8. Cost Matrix
% This dataset requires use of a cost matrix (see below)
% 1 2
% 1 0 1
% 2 5 0
% (1 = Good, 2 = Bad)
% the rows represent the actual classification and the columns
% the predicted classification.
% It is worse to class a customer as good when they are bad (5),
% than it is to class a customer as bad when they are good (1).
desired cost matrix
Line 4 Col 3 (none) Unicode (UTF-8) Unix (LF) Last saved: (Never)
20140311-JustSa...esigner
Breakout_Final.zip
Gamification


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
09:41:46 - trees.J48
10:28:20 - trees.J48
Classifier output
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 705 70.5 %
Incorrectly Classified Instances 295 29.5 %
Kappa statistic 0.2467
Total Cost 1027
Average Cost 1.027
Mean absolute error 0.3467
Root mean squared error 0.4796
Relative absolute error 82.5233 %
Root relative squared error 104.6565 %
Total Number of Instances 1000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
0.84 0.61 0.763 0.84 0.799 0.639 good
0.39 0.16 0.511 0.39 0.442 0.639 bad
Weighted Avg. 0.705 0.475 0.687 0.705 0.692 0.639
=== Confusion Matrix ===
a b <-- classified as
588 112 | a = good
183 117 | b = bad
incorrectly classified examples
total cost has changed
Status
OK
Log
x 0

Precision & Recall

- Alternatives to accuracy, introduced in the area of information retrieval and search engine
- Precision
  - Percentage of items classified as positive that are actually positive
  - In the information retrieval context represents the percentage of actually good documents that have been shown as a result.
- Recall
  - Percentage of positive examples that are classified as positive
  - 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

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

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

Sensitivity & Specificity

- Sensitivity evaluates the ability to correctly identify the elements of the positive class and it is computed as the true positive rate (TPR)

  TPR = TP/(TP+FN)

- Specificity estimates the probability to correctly identify the elements of the negative class and it is computed as the true negative rate

  TNR = TN/(TN+FP)

How to compare the relative performance among competing models?

How to Compare the Performance of Two Models?

- Suppose we have two models
  - Model M_A with an accuracy = 82% computed using 10-fold crossvalidation
  - Model M_B with an accuracy = 80% computed using 10-fold crossvalidation
- How much confidence can we place on accuracy of M_A and M_B?
- Can we say M_A is better than M_B?
- Can the difference in performance measure be explained as a result of random fluctuations in the test set?

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

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 to chance

Paired t-Test Evaluation using k-fold Crossvalidation

- Generate the k folds and for each configuration compute the performance of model A and B,

$$\theta_1^A \ldots \theta_k^A \qquad \theta_1^B \ldots \theta_k^B$$

- compute mean and standard deviation of the differences:

$$\delta_i = \theta_i^A - \theta_i^B \quad \mu_\delta = \frac{1}{k}\sum_i \delta_i \quad \sigma_\delta = \sqrt{\frac{1}{k}\sum_i (\delta_i - \mu_\delta)^2}$$

- And two hypotheses, the null hypothesis and the alternative hypothesis

$$H_0 : \mu_\delta = 0 \qquad H_a : \mu_\delta \neq 0$$

- We apply the t-test to check whether we can reject the null hypothesis H_0 with a target confidence

ALGORITHM 22.4. Paired *t*-Test via Cross-Validation

PAIRED *t*-TEST(α, K, $\mathbf{D}$):

D $\leftarrow$ randomly shuffle **D**
$\{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_K\} \leftarrow$ partition **D** in K equal parts
foreach $i \in [1, K]$ **do**
    $M_i^A, M_i^B \leftarrow$ train the two different classifiers on $\mathbf{D} \setminus \mathbf{D}_i$
    $\theta_i^A, \theta_i^B \leftarrow$ assess M_i^A and M_i^B on $\mathbf{D}_i$
    $\delta_i = \theta_i^A - \theta_i^B$
$\hat{\mu}_\delta = \frac{1}{K} \sum_{i=1}^{K} \delta_i$
$\hat{\sigma}_\delta^2 = \frac{1}{K} \sum_{i=1}^{K} (\delta_i - \hat{\mu}_\delta)^2$
$Z_\delta^* = \frac{\sqrt{K}\hat{\mu}_\delta}{\hat{\sigma}_\delta}$
if $Z_\delta^* \in \left(-t_{\alpha/2,K-1}, t_{\alpha/2,K-1}\right)$ **then**
    Accept H_0; both classifiers have similar performance
else
    Reject H_0; classifiers have significantly different performance

Comparing the Performance of 2 Models using k-fold Cross-validation

- First decide on a confidence level, e.g. 95%
  - Corresponds to false discovery (false positive) rate: $\boldsymbol{\alpha} = 5\%$
  - How frequently you are willing to declare difference when there is none
- Apply k-fold cross-validation to each model
  - Obtaining k evaluations for each algorithm over same folds
- Apply Student's t-test and compute p-value to determine whether reported difference is statistically significant
  - If p-value>$\boldsymbol{\alpha}$ then difference is not significant (can claim nothing)
  - If p-value<$\boldsymbol{\alpha}$ then difference is significant (claim one better than the other)
  - Note that the t-test can be paired or unpaired

"paired" when the estimates
are from the same datasets

"unpaired" when the estimates
are from different datasets

Multiple Hypothesis Testing

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 (95% confidence level), but you repeat the test on 20 different observations. 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.

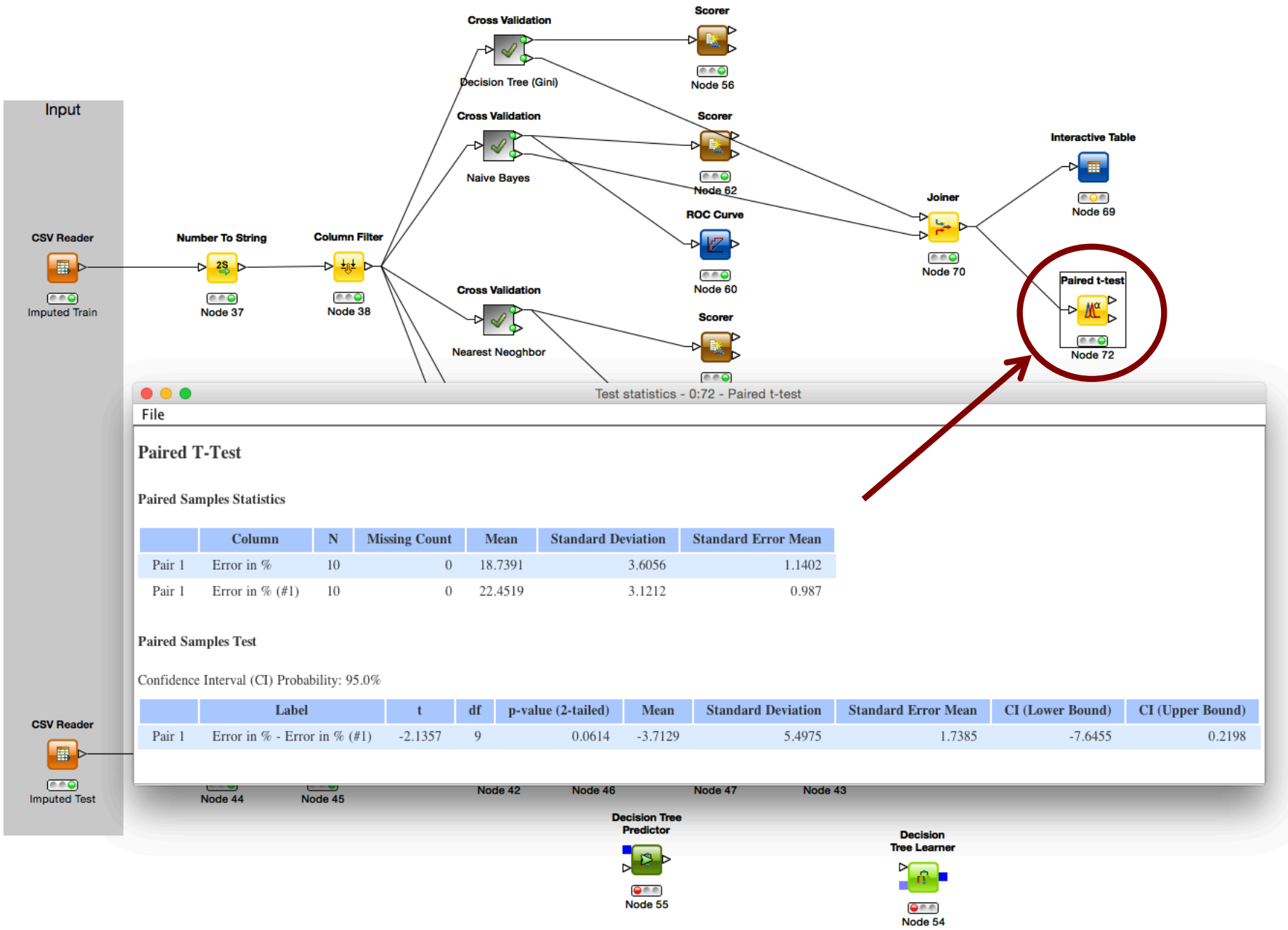
Bonferroni Correction

- 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

- They 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

	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


Input
CSV Reader
Imported Train
Number To String
Node 37
Column Filter
Node 38
Cross Validation
Decision Tree (Gini)
Scorer
Node 56
Cross Validation
Naive Bayes
Scorer
Node 62
ROC Curve
Node 60
Cross Validation
Nearest Neoghbor
Scorer
Joiner
Node 70
Interactive Table
Node 69
Paired t-test
Node 72
Test statistics - 0:72 - Paired t-test
File
Paired T-Test
Paired Samples Statistics
Column | N | Missing Count | Mean | Standard Deviation | Standard Error Mean
Pair 1 | Error in % | 10 | 0 | 18.7391 | 3.6056 | 1.1402
Pair 1 | Error in % (#1) | 10 | 0 | 22.4519 | 3.1212 | 0.987
Paired Samples Test
Confidence Interval (CI) Probability: 95.0%
Label | t | df | p-value (2-tailed) | Mean | Standard Deviation | Standard Error Mean | CI (Lower Bound) | CI (Upper Bound)
Pair 1 | Error in % - Error in % (#1) | -2.1357 | 9 | 0.0614 | -3.7129 | 5.4975 | 1.7385 | -7.6455 | 0.2198
CSV Reader
Imported Test
Node 44
Node 45
Node 42
Node 46
Node 47
Node 43
Decision Tree Predictor
Node 55
Decision Tree Learner
Node 54

Probabilistic Classifiers

Probabilistic Classifiers & Classification Thresholds

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

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

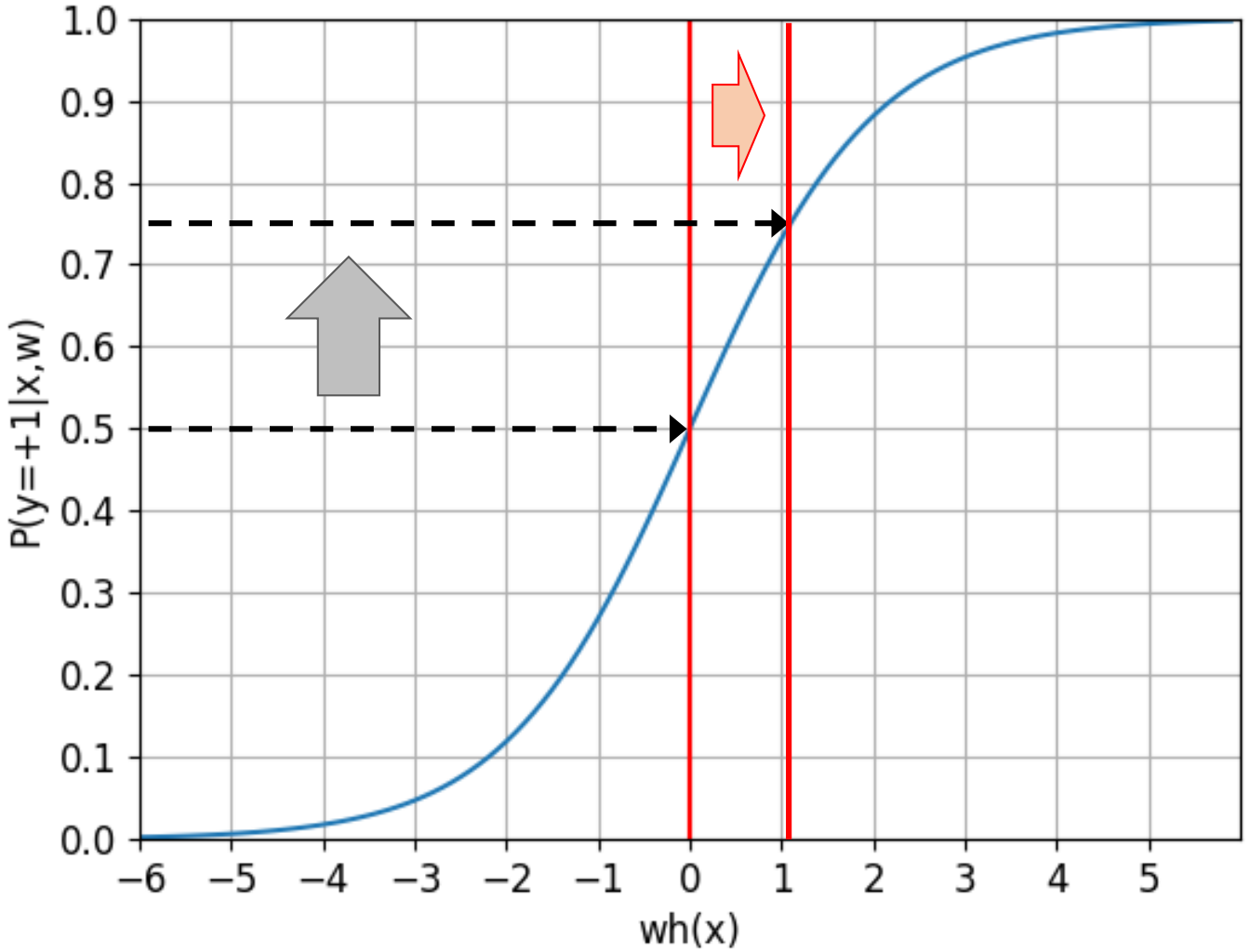
- Given an example x_i its predicted class is the label with the largest probability so it is equivalent to using a threshold of 0.5 to decide which class to assign to an example

- 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.

instances classified as negative

instances classified 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 to produce few false positives
- Recall will be low
  - In fact, we are likely to produce many 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 minimum 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

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

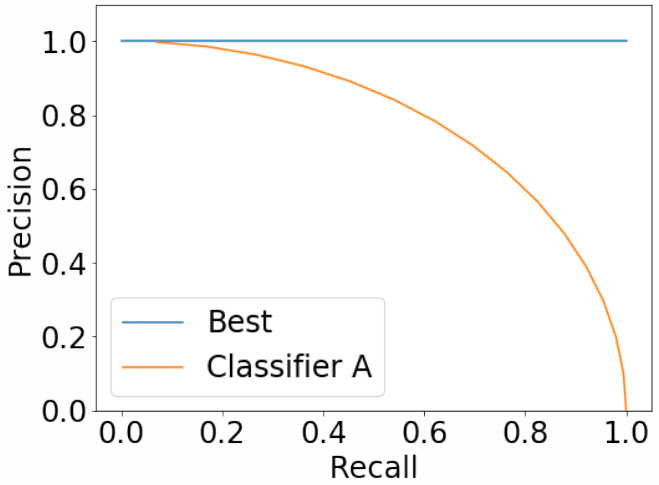
- By increasing the classification threshold for positive (+1) examples to 0.75 we obtained a new confusion matrix with precision of 0.86 and a recall of 0.80

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

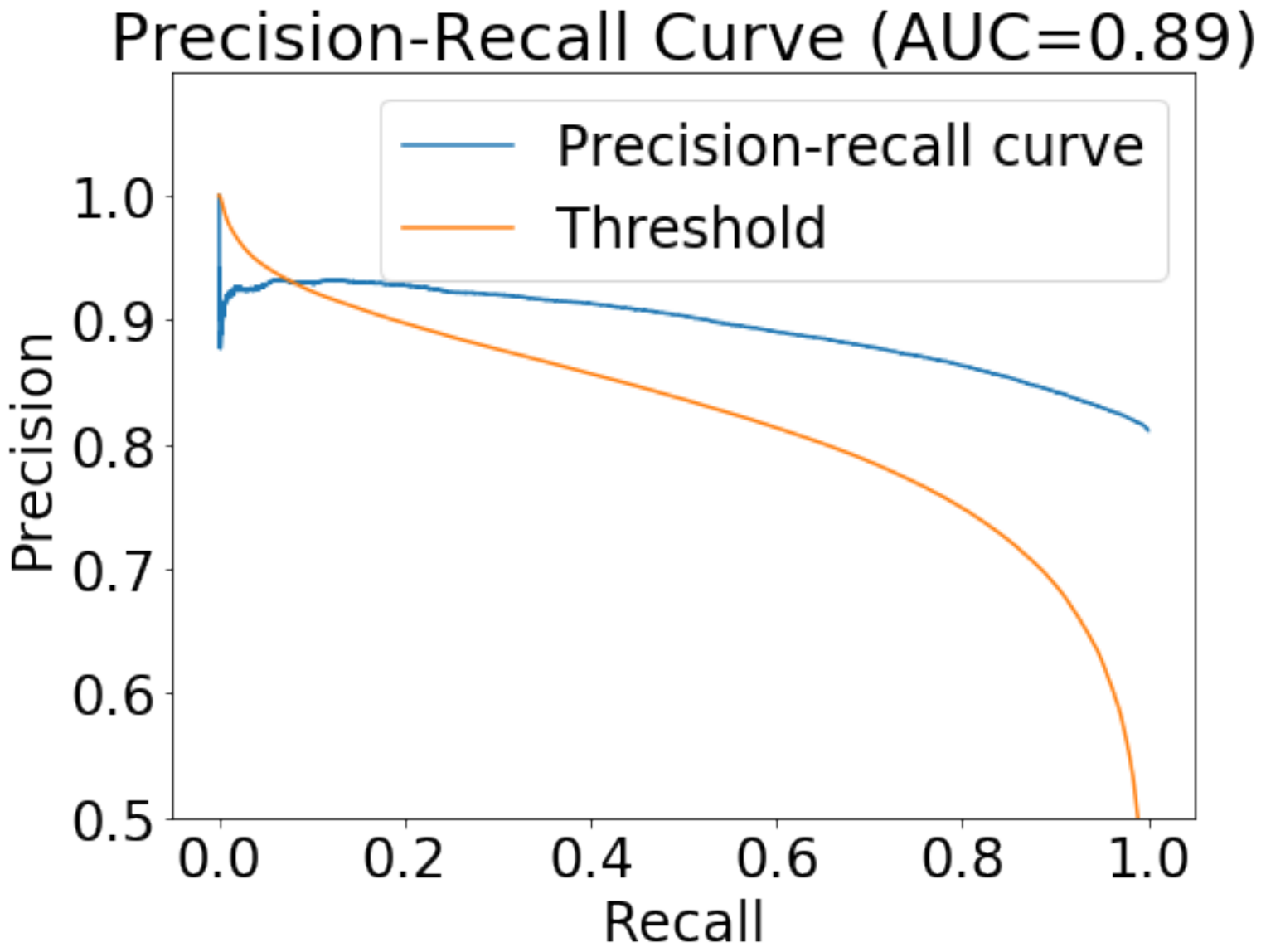
- Overall, we reduced the number of false positives (we did not accept risky loans and we were better at identifying risky loans)

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)

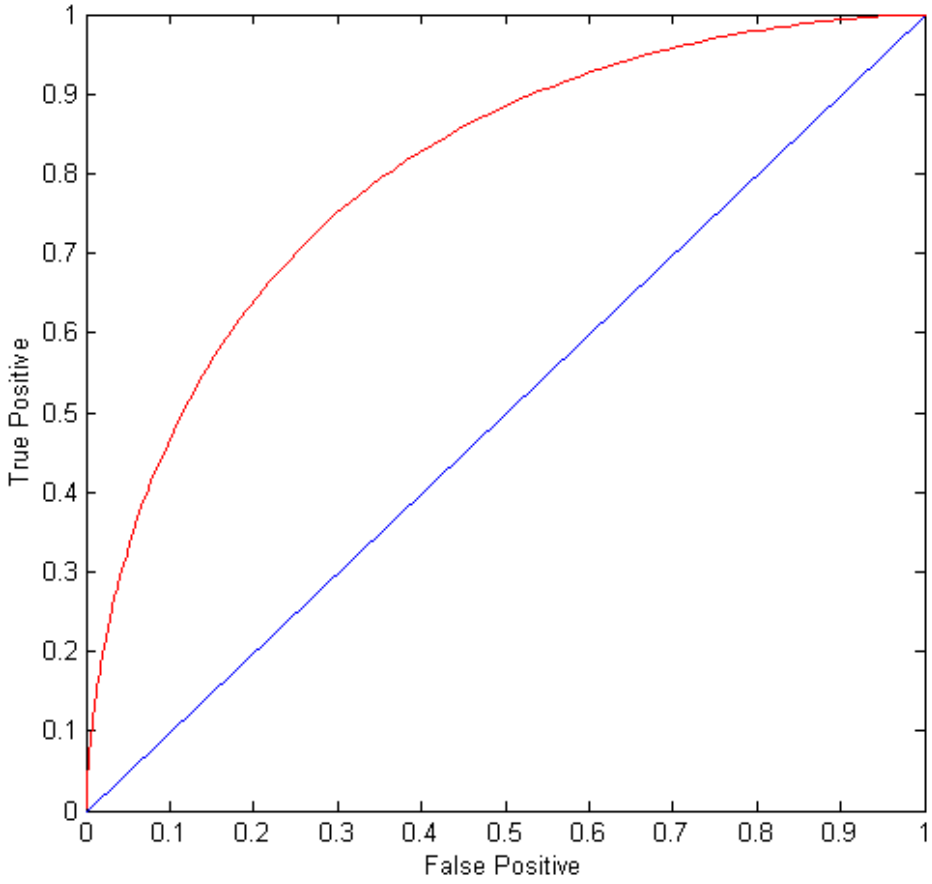
Receiver Operating Characteristic (ROC) Curves

ROC Curve
(Receiver Operating Characteristic)

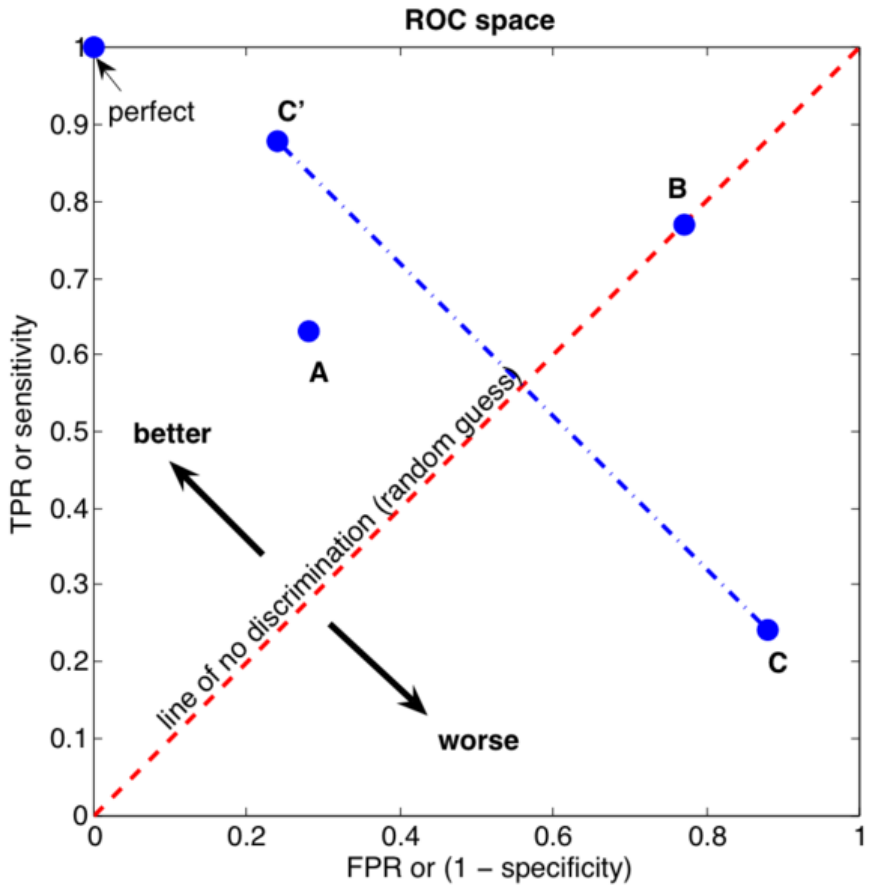
- Developed in 1950s for signal detection theory to analyze signals
- Plot the True Positive Rate (TPR=TP/(TP+FN))
  against the False Positive Rate (FPR=FP/(TN+FP))
- Performance of each classifier represented
  as a point on the ROC curve
- Changing the classification threshold, sample distribution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

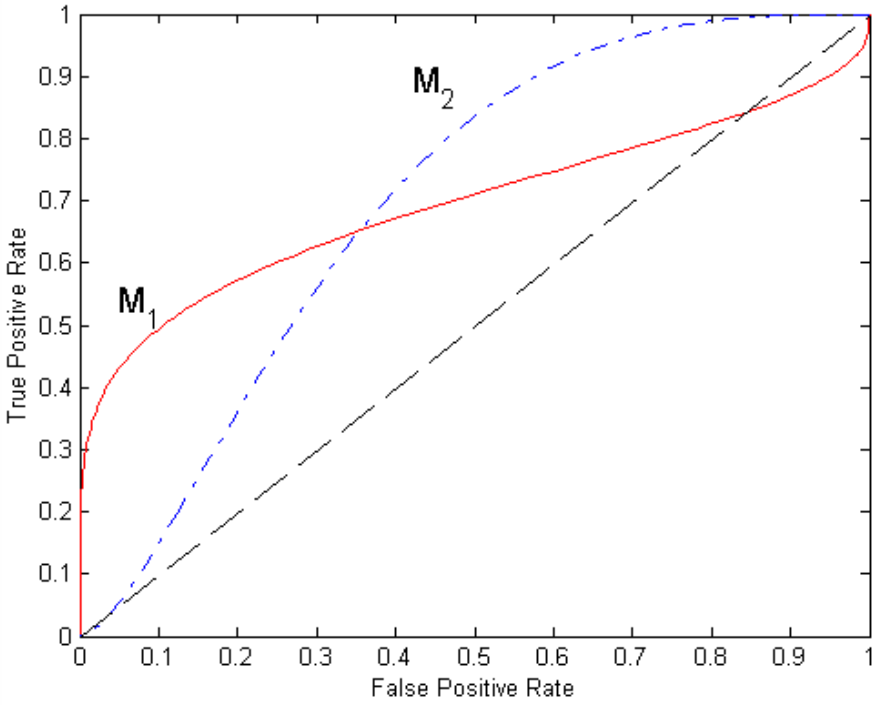
FPR = 0.24

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



There are techniques similar to ROC in other areas

Lift charts are an example

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.

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 no

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

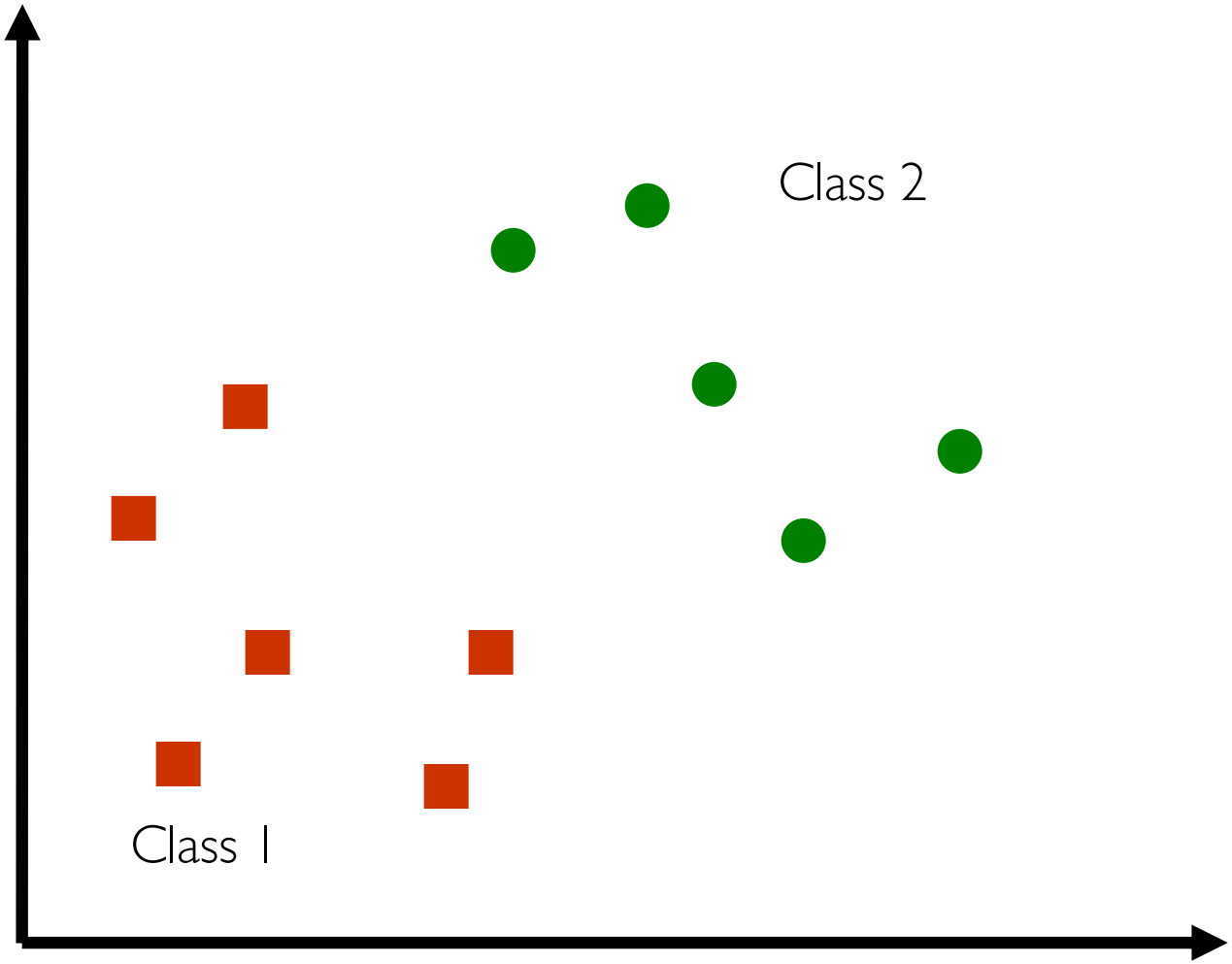
- "[A]ll algorithms that search for an extremum of a cost [objective] function perform exactly the same, when averaged over all possible cost functions." [1]
- "[T]he average performance of any pair of algorithms across all possible problems is identical." [2]
- Wolpert, D.H., Macready, W.G. (1995), 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.

Additional Material

- https://arxiv.org/pdf/1811.12808.pdf
- https://arxiv.org/pdf/1809.09446.pdf

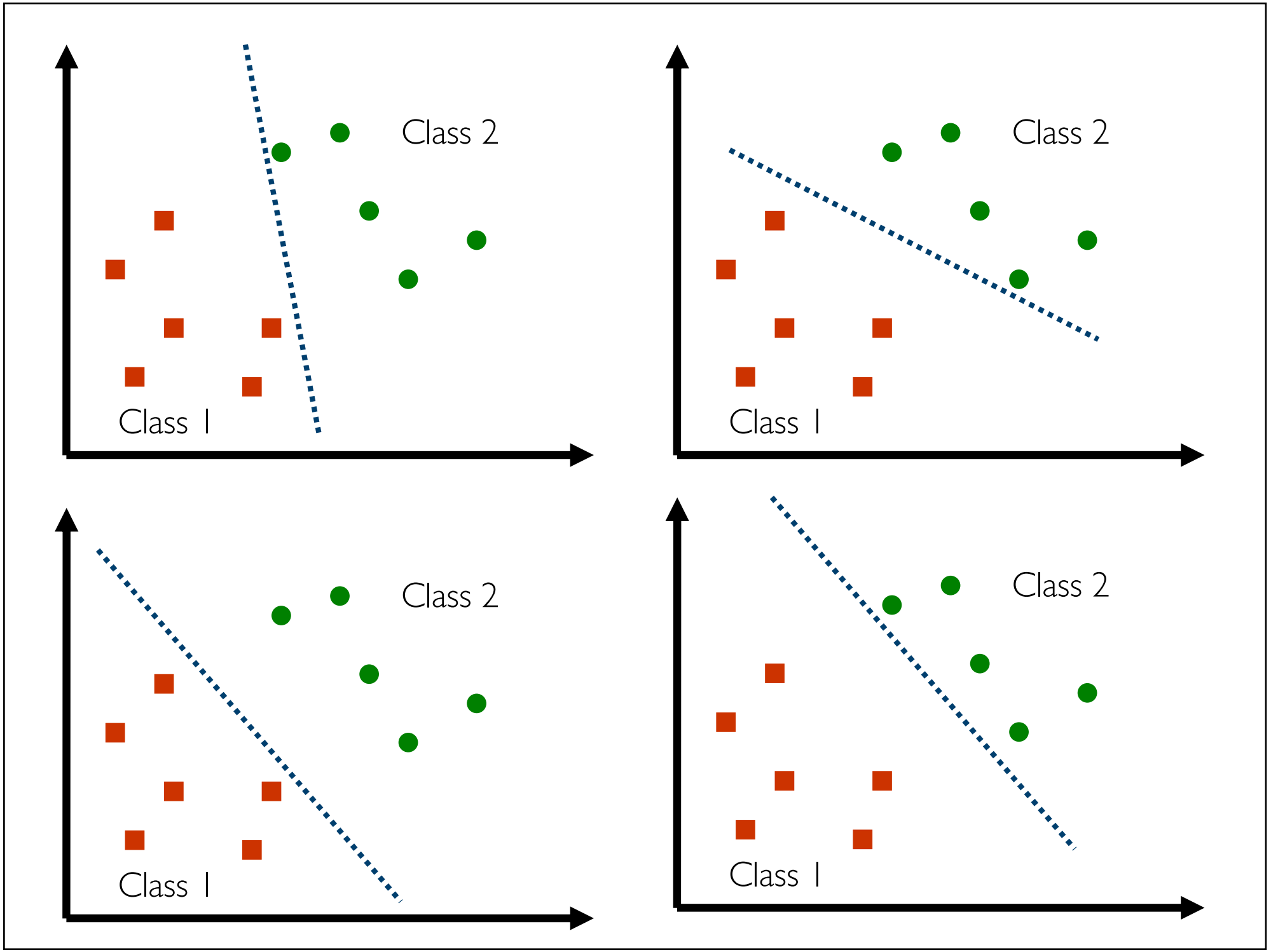
And now some names
you might know …

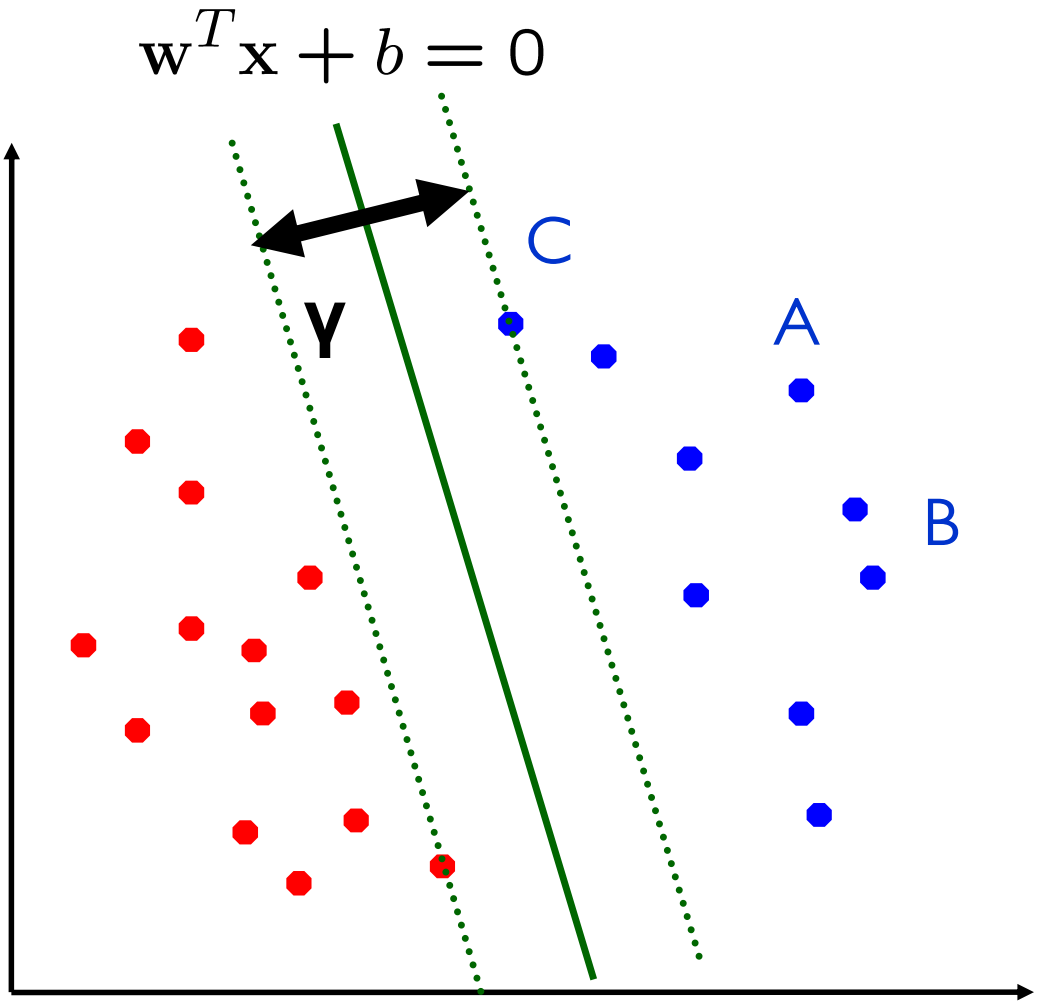
Support Vector Machines



Many decision boundaries can separate these two classes

Which one should we choose?

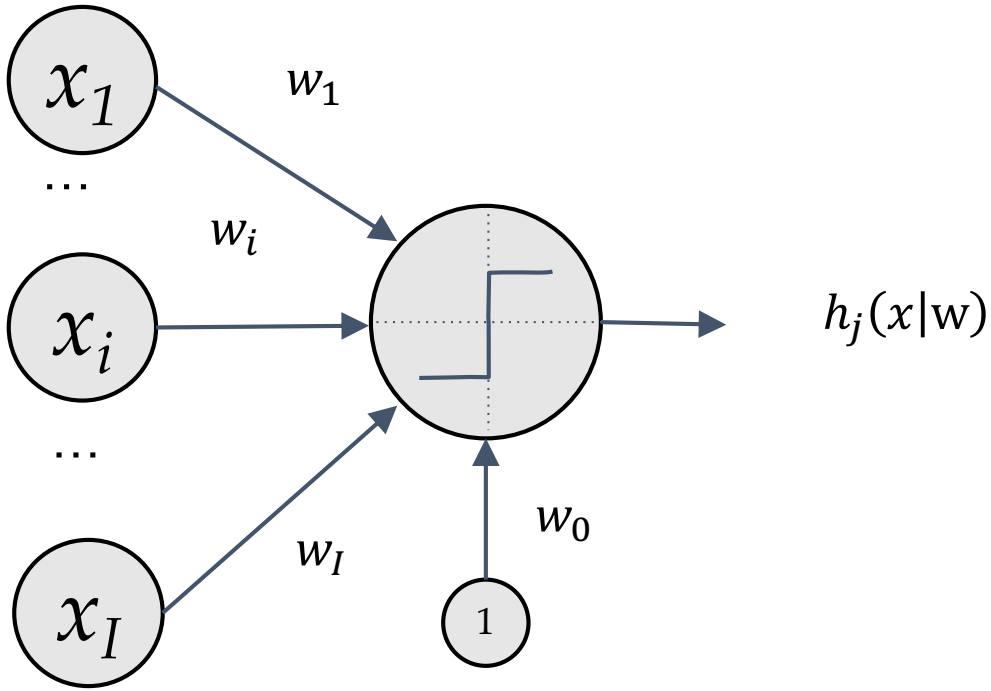

Class 2
Class 1
Class 2
Class 1
Class 2
Class 1
Class 2
Class 1



SVMs work by searching for the hyperplane that maximizes the margin or the largest γ such that

$$\forall i, y_i(wx_i + b) \geq \gamma$$

Neural Networks



$$h_j(x|\mathrm{w},\mathrm{b}) = \mathrm{h_j}\left(\Sigma_{i=1}^{I} w_i \cdot x_i - b\right) = \mathrm{h_j}\left(\Sigma_{i=0}^{I} w_i \cdot x_i\right) = h_j(w^T x)$$

Computation in an artificial neurons

Perceptron Math

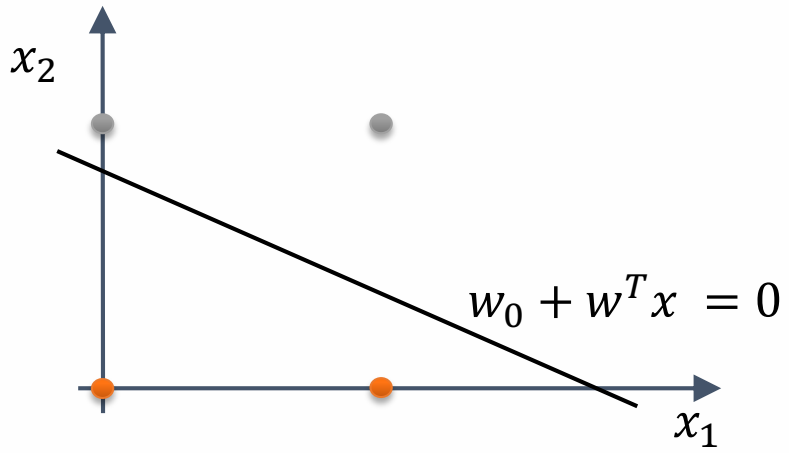
- A perceptron computes the value of a weighted sum and returns its Sign (Thresholding)

$$h_j(x|\mathrm{w}) = \mathrm{h_j}\left(\Sigma_{i=0}^{I} w_i \cdot x_i\right) = Sign(w_0 + w_1 \cdot x_1 + \cdots + w_I \cdot x_I)$$

- It is basically a linear classifier for which the decision boundary is the hyperplane $w_0 + w_1 \cdot x_1 + \cdots + w_I \cdot x_I = 0$

- In 2D, this turns into

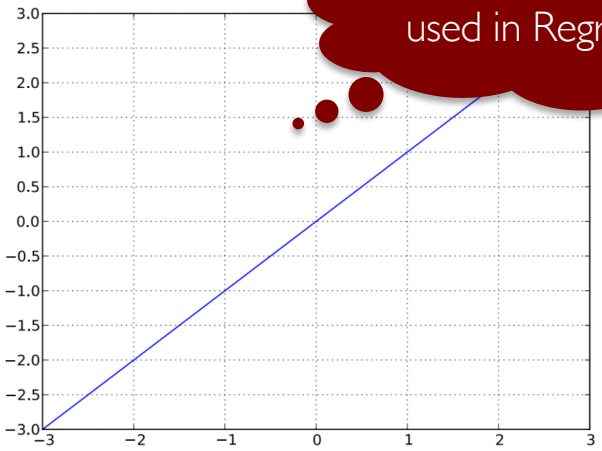
$$w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 = 0$$
$$w_2 \cdot x_2 = -w_0 - w_1 \cdot x_1$$
$$x_2 = -\frac{w_0}{w_2} - \frac{w_1}{w_2} \cdot x_1$$



Which Activation Function?

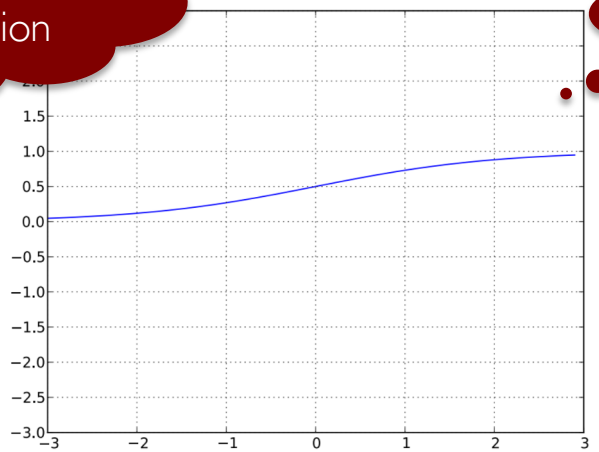
Linear activation used in Regression

Sigmoid and Tanh used in Classification



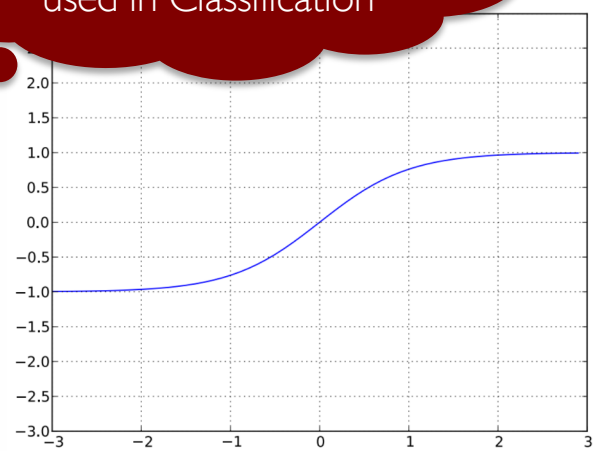
Linear activation function

$$g(a) = a$$
$$g'(a) = 1$$



Sigmoid activation function

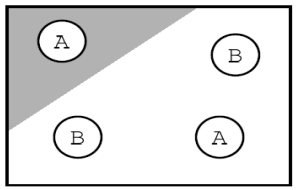
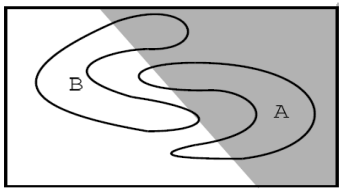
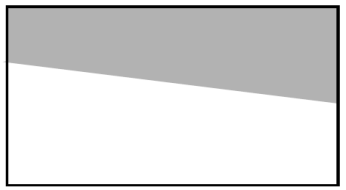
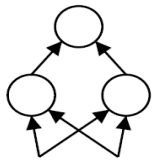
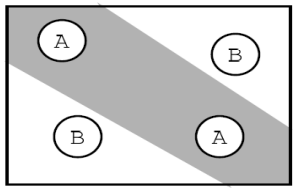
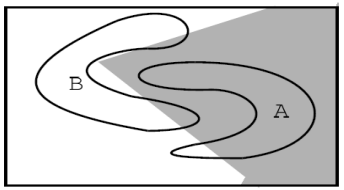
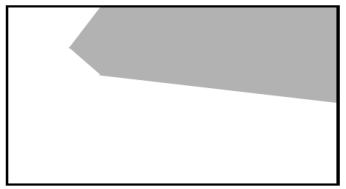
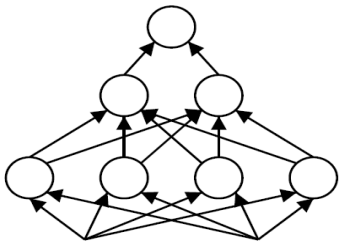
$$g(a) = \frac{1}{1 + \exp(-a)}$$
$$g'(a) = g(a)(1 - g(a))$$



Tanh activation function

$$g(a) = \frac{\exp(a) - \exp(-a)}{\exp(a) + \exp(-a)}$$
$$g'(a) = 1 - g(a)^2$$

What Can We Do With it?

Topology	Type of Decision Region	XOR Problem	Classes with Meshed Regions	Most General Region Shapes
	Half bounded by hyperplanes			
	Convex Open or Clesed Regions			
	Arbitrary Regions (Complexity limited by the number of nodes)