DATA MINING LECTURE 10

Classification

- Nearest Neighbor Classification
- Support Vector Machines
- Logistic Regression
- Naïve Bayes Classifier

Supervised Learning

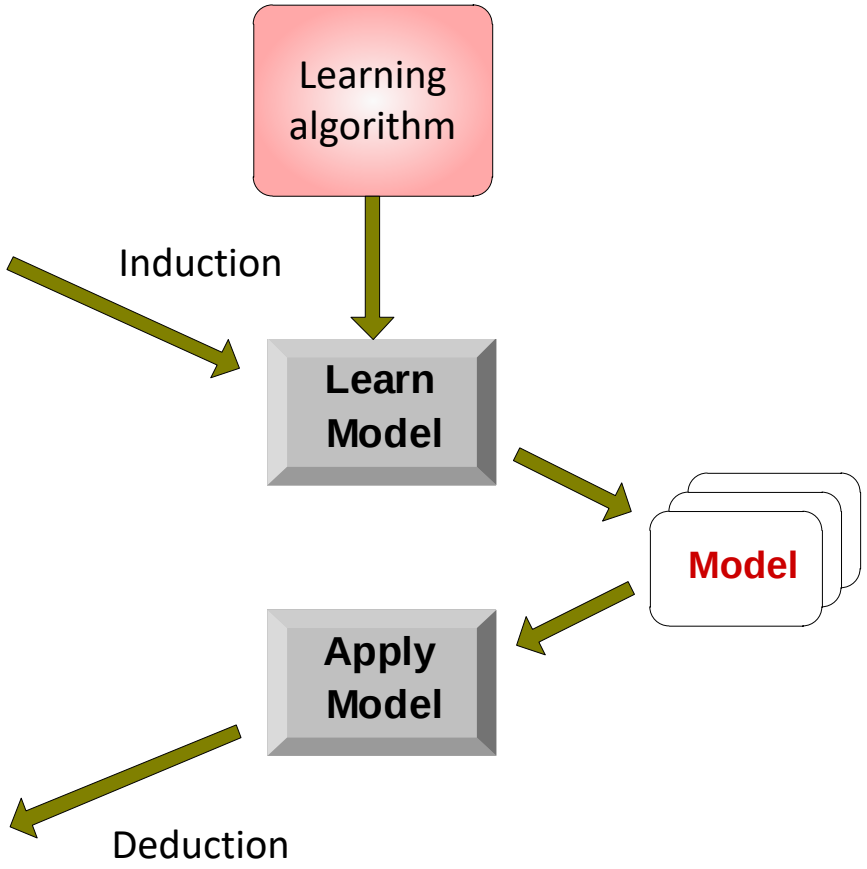
Illustrating Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



NEAREST NEIGHBOR CLASSIFICATION

Instance-Based Classifiers

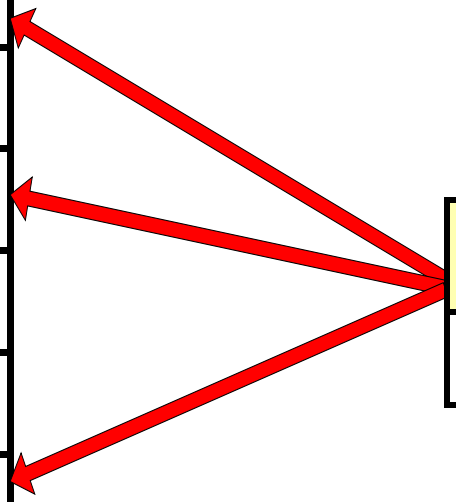
Set of Stored Cases

Atr1	……	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

- Store the training records
- Use training records to predict the class label of unseen cases

Unseen Case

Atr1	……	AtrN

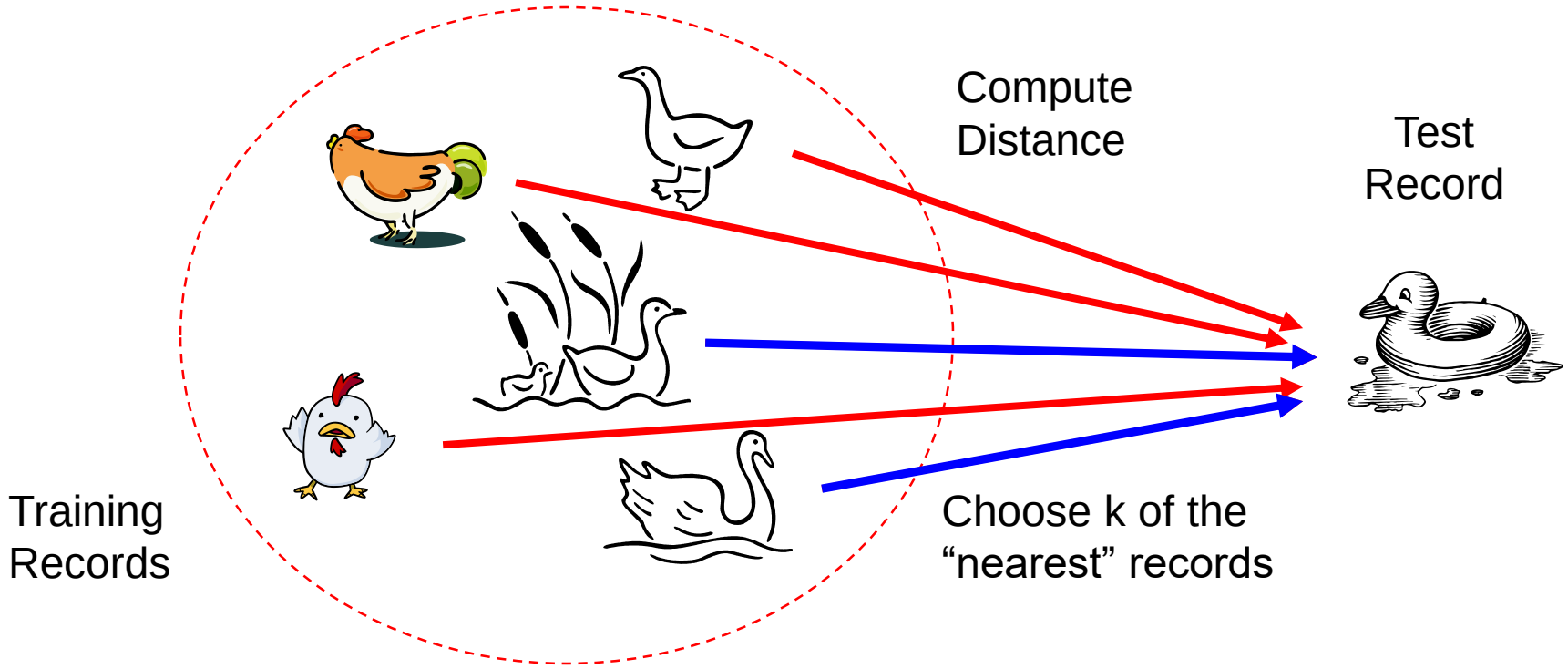


Instance Based Classifiers

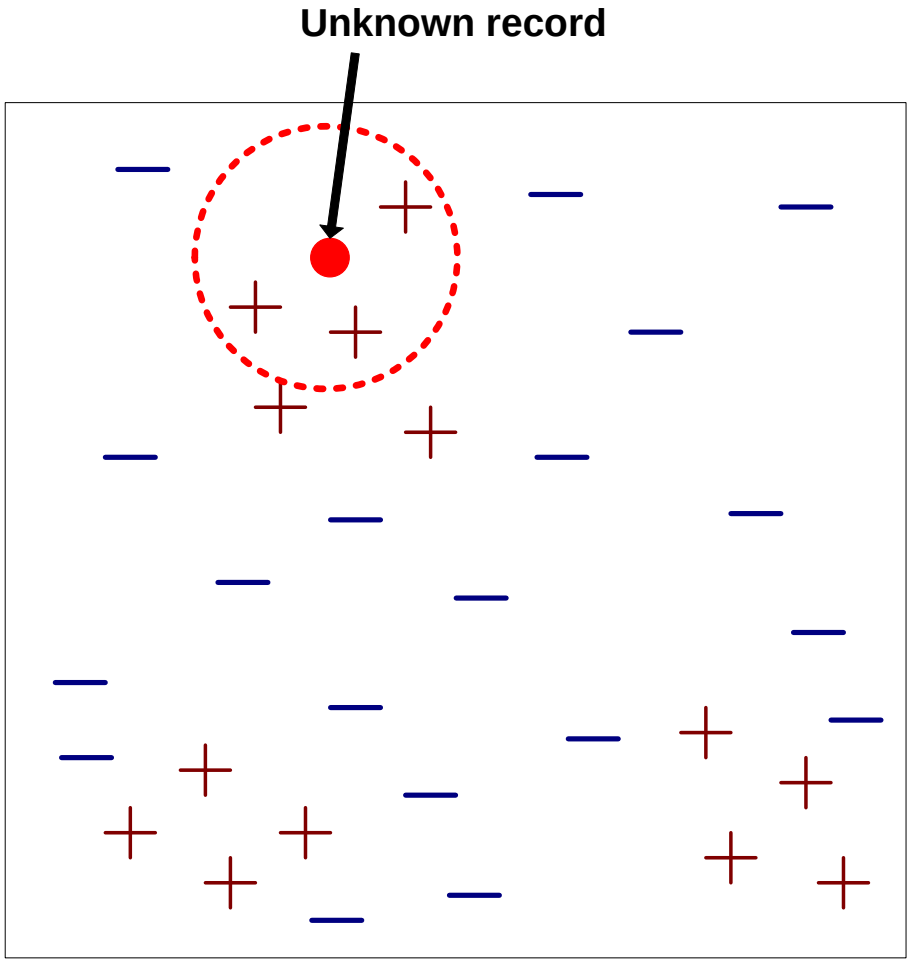
- Examples:
  - Rote-learner
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
  - Nearest neighbor classifier
    - Uses k “closest” points (nearest neighbors) for performing classification

Nearest Neighbor Classifiers

- Basic idea:
  - "*If it walks like a duck, quacks like a duck, then it's probably a duck*"



Nearest-Neighbor Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  1. Compute distance to other training records
  2. Identify *k* nearest neighbors
  3. Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

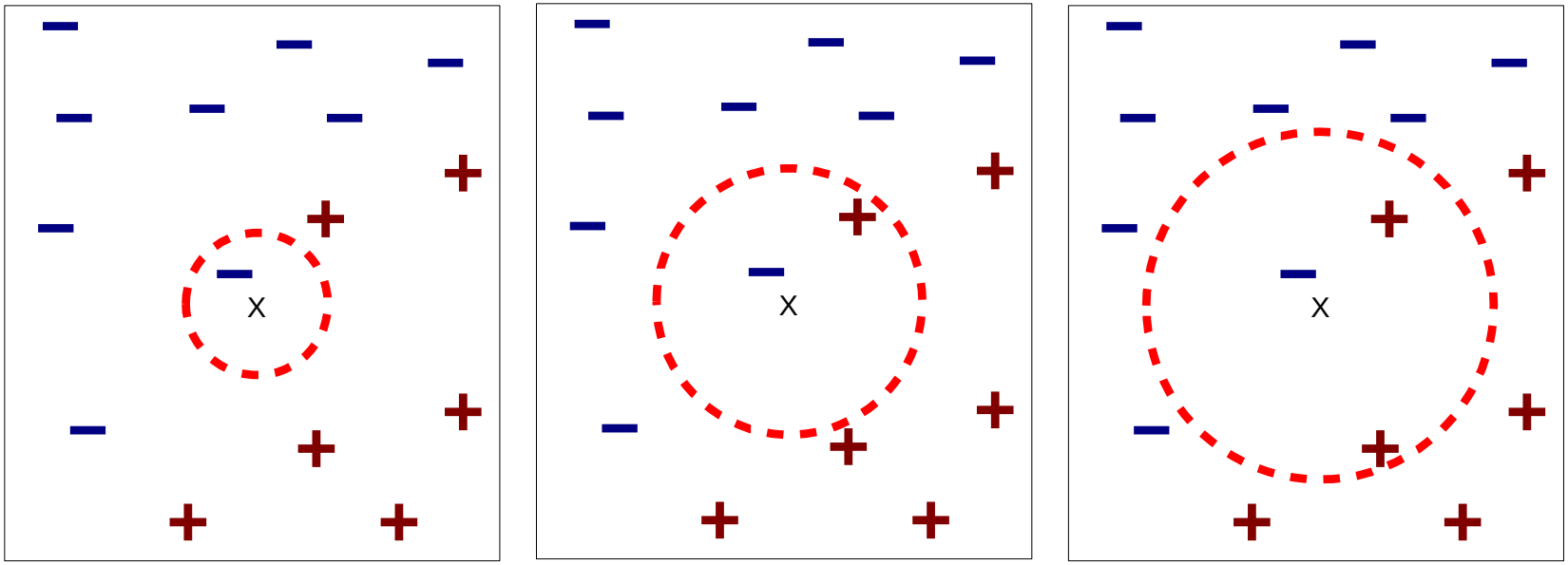
Nearest Neighbor Classification

- Compute distance between two points:
  - Euclidean distance

$$d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

Definition of Nearest Neighbor

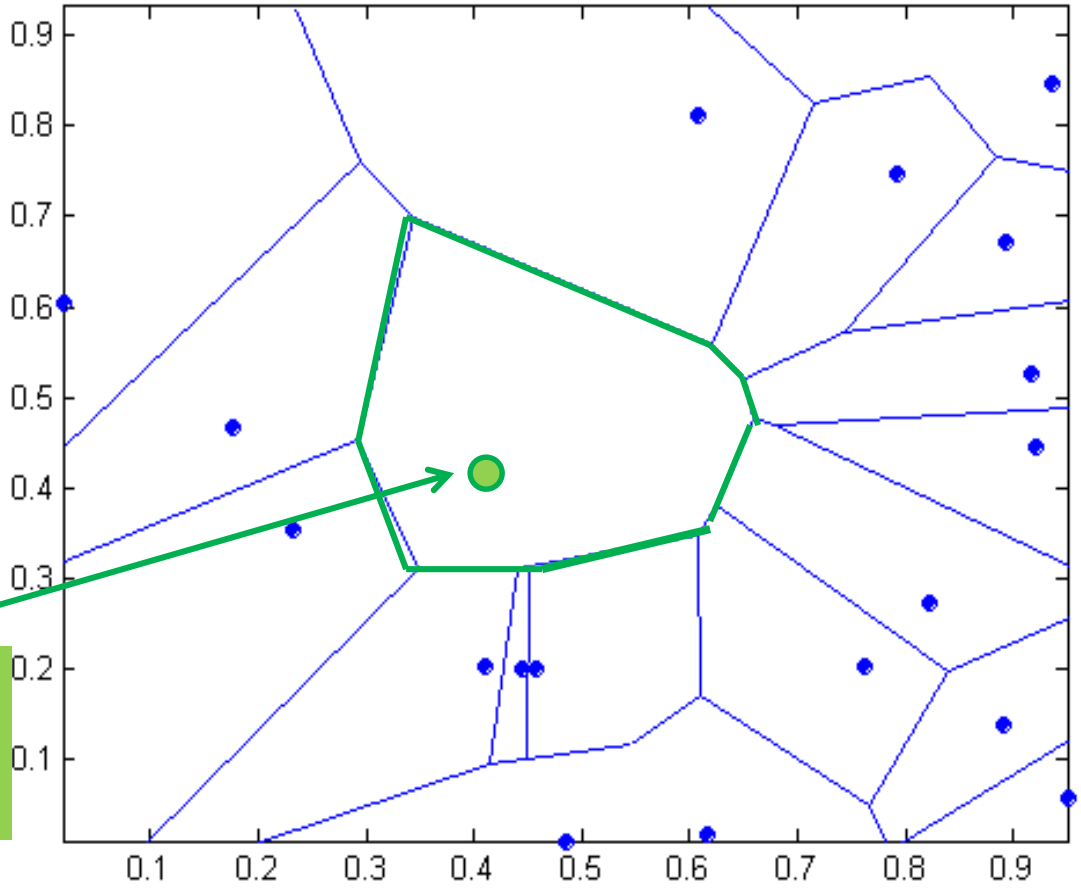


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

1 nearest-neighbor

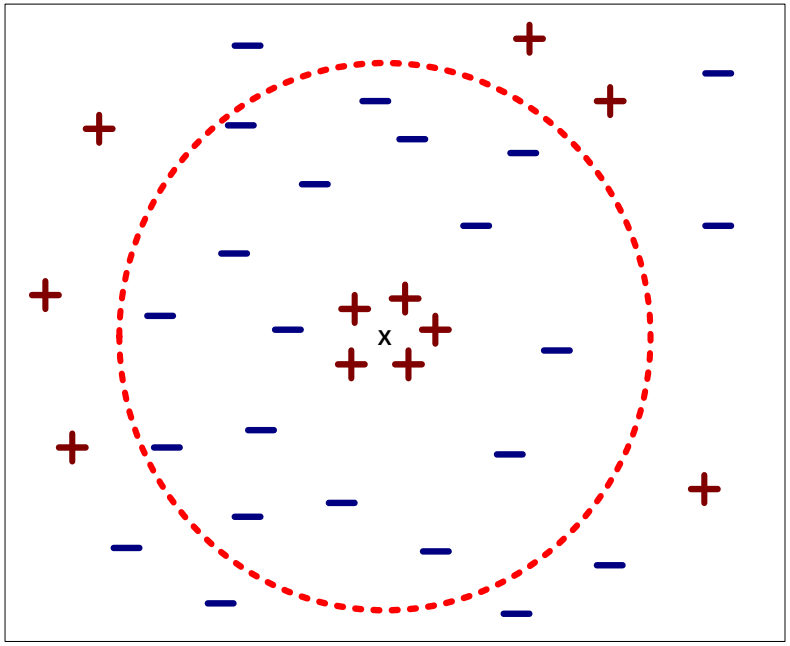
Voronoi Diagram defines the classification boundary



The area takes the class of the green point

Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



The value of k is the complexity of the model

Example

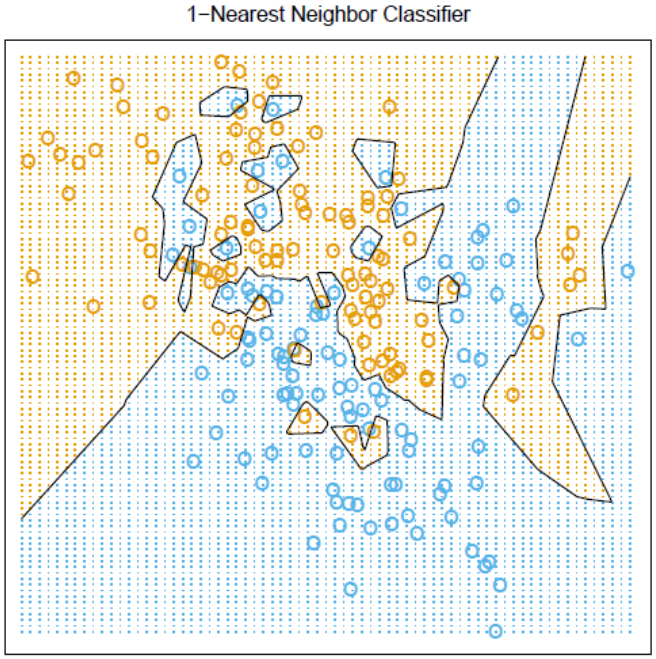


FIGURE 2.3. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$)*, and then predicted by* 1*-nearest-neighbor classification.*

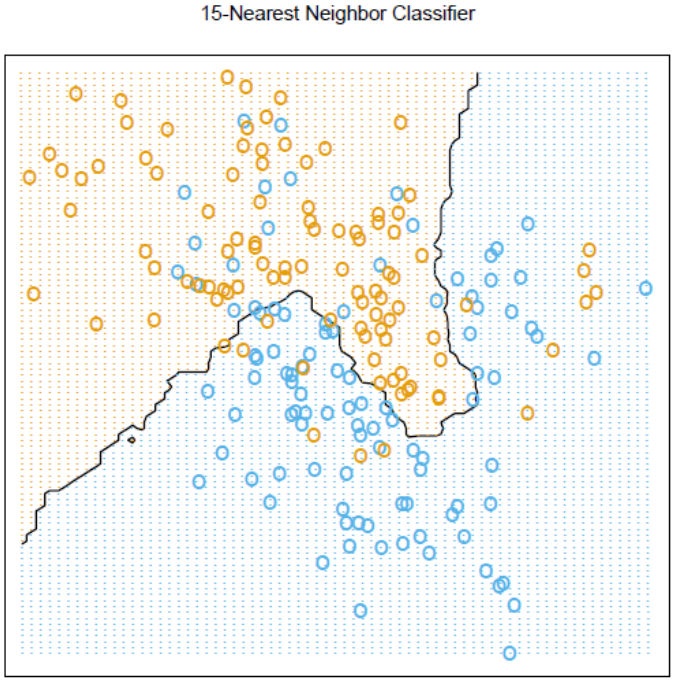


FIGURE 2.2. *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (BLUE $= 0$, ORANGE $= 1$) *and then fit by* 15*-nearest-neighbor averaging as in (2.8). The predicted class is hence chosen by majority vote amongst the* 15*-nearest neighbors.*

Example

FIGURE 2.1. *A classification example in two dimensions. The classes are coded as a binary variable (BLUE = 0, ORANGE = 1), and then fit by linear regression. The line is the decision boundary defined by* $x^T\hat{\beta} = 0.5$. *The orange shaded region denotes that part of input space classified as ORANGE, while the blue region is classified as BLUE.*

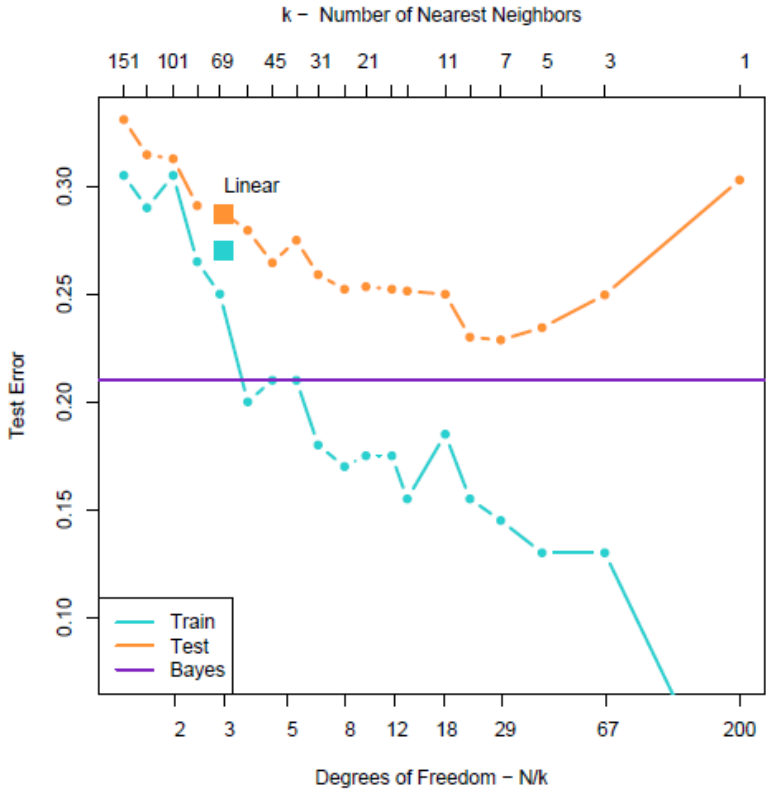


FIGURE 2.4. *Misclassification curves for the simulation example used in Figures 2.1, 2.2 and 2.3. A single training sample of size* 200 *was used, and a test sample of size* 10,000. *The orange curves are test and the blue are training error for k-nearest-neighbor classification. The results for linear regression are the bigger orange and blue squares at three degrees of freedom. The purple line is the optimal Bayes error rate.*

Nearest Neighbor Classification…

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Nearest Neighbor Classification…

- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results

1 1 1 1 1 1 1 1 1 1 1 0		1 0 0 0 0 0 0 0 0 0 0 0
	vs	
0 1 1 1 1 1 1 1 1 1 1 1		0 0 0 0 0 0 0 0 0 0 0 1
d = 1.4142		d = 1.4142

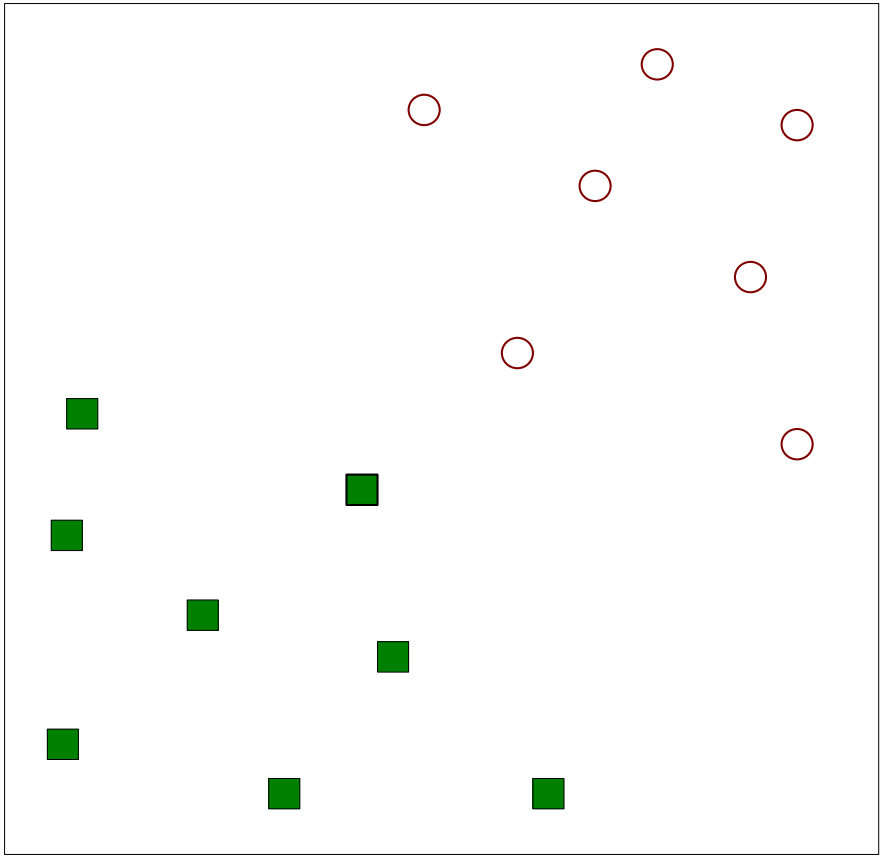
- Solution: Normalize the vectors to unit length

Nearest neighbor Classification…

- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision trees
- Classifying unknown records are relatively expensive
  - Naïve algorithm: O(n)
  - Need for structures to retrieve nearest neighbors fast.
    - The Nearest Neighbor Search problem.
    - Also, Approximate Nearest Neighbor Search

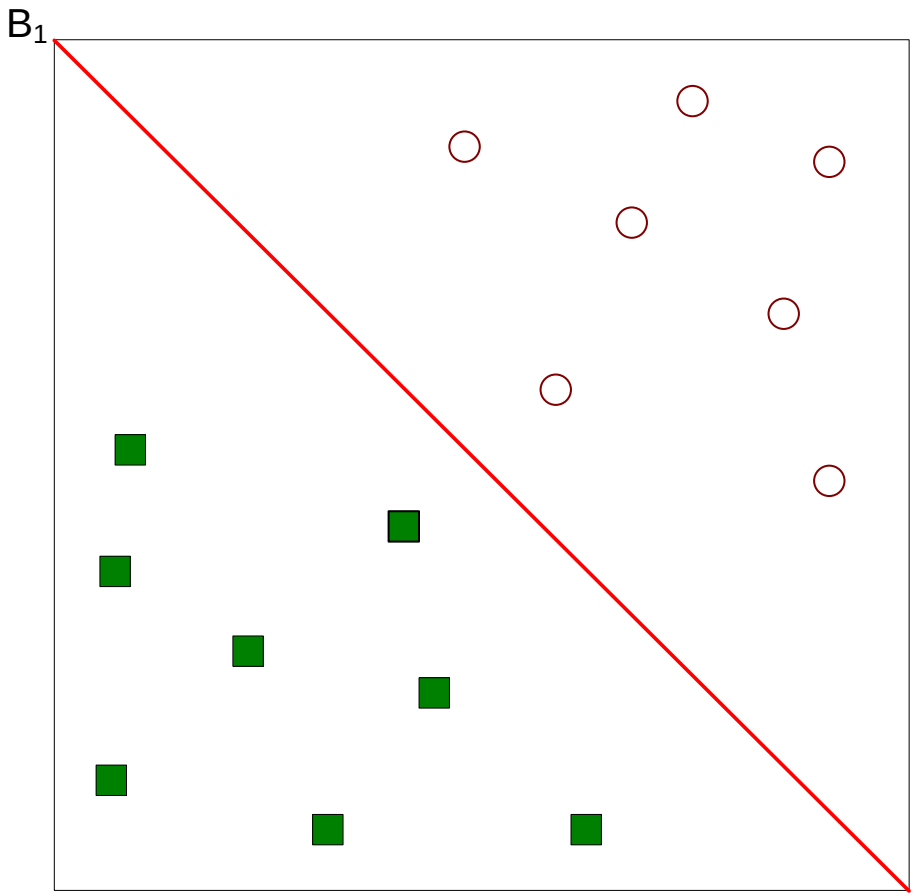
SUPPORT VECTOR MACHINES

Support Vector Machines



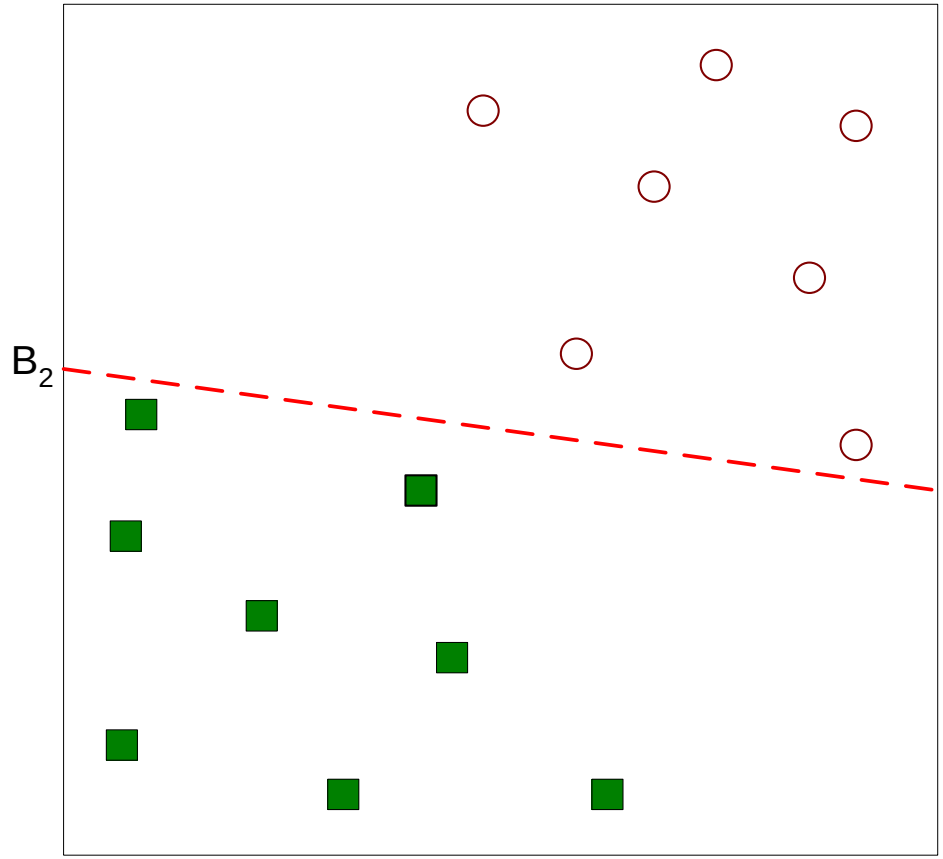
- Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



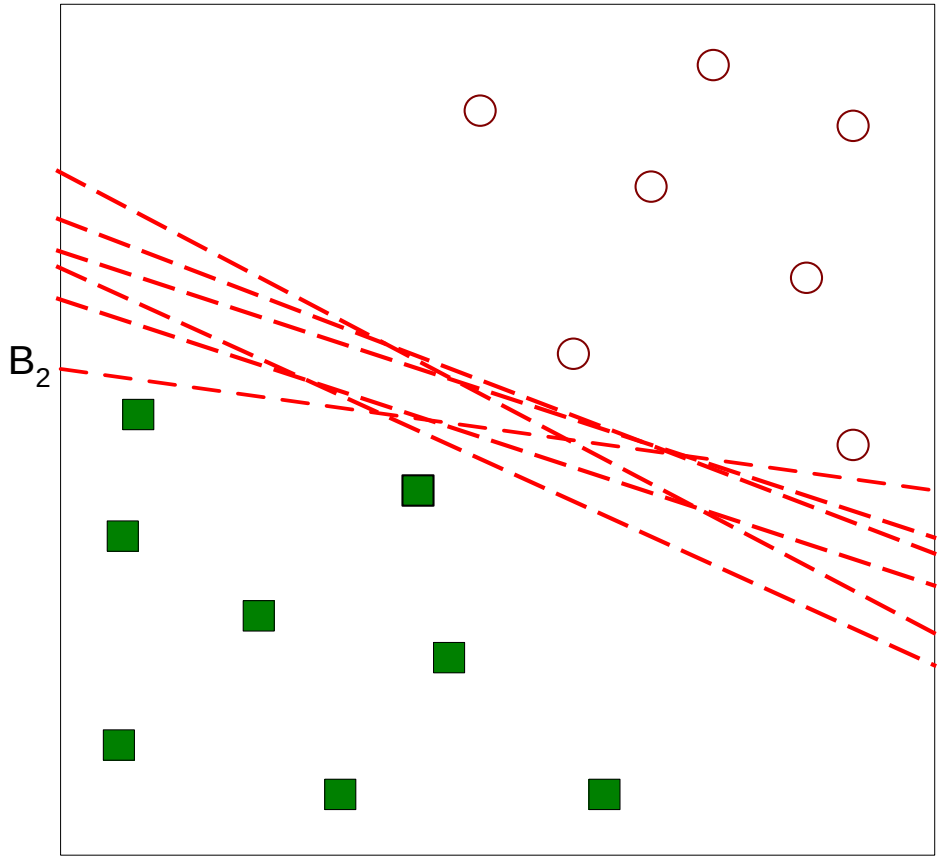
- One Possible Solution

Support Vector Machines



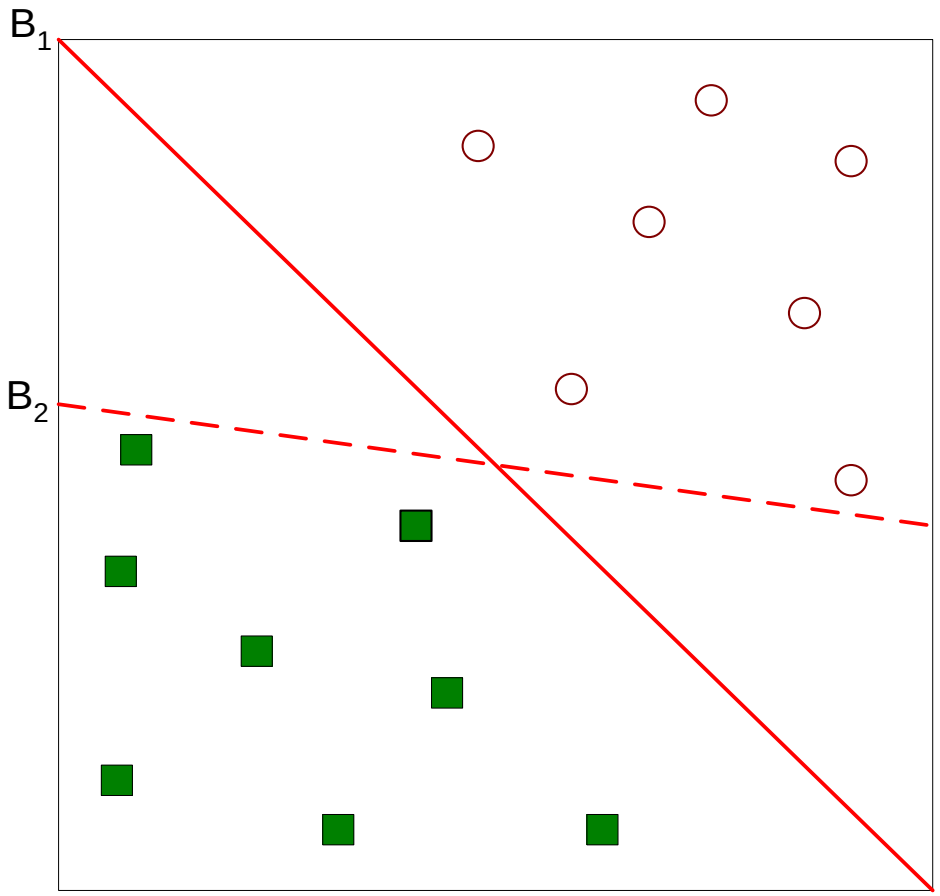
- Another possible solution

Support Vector Machines



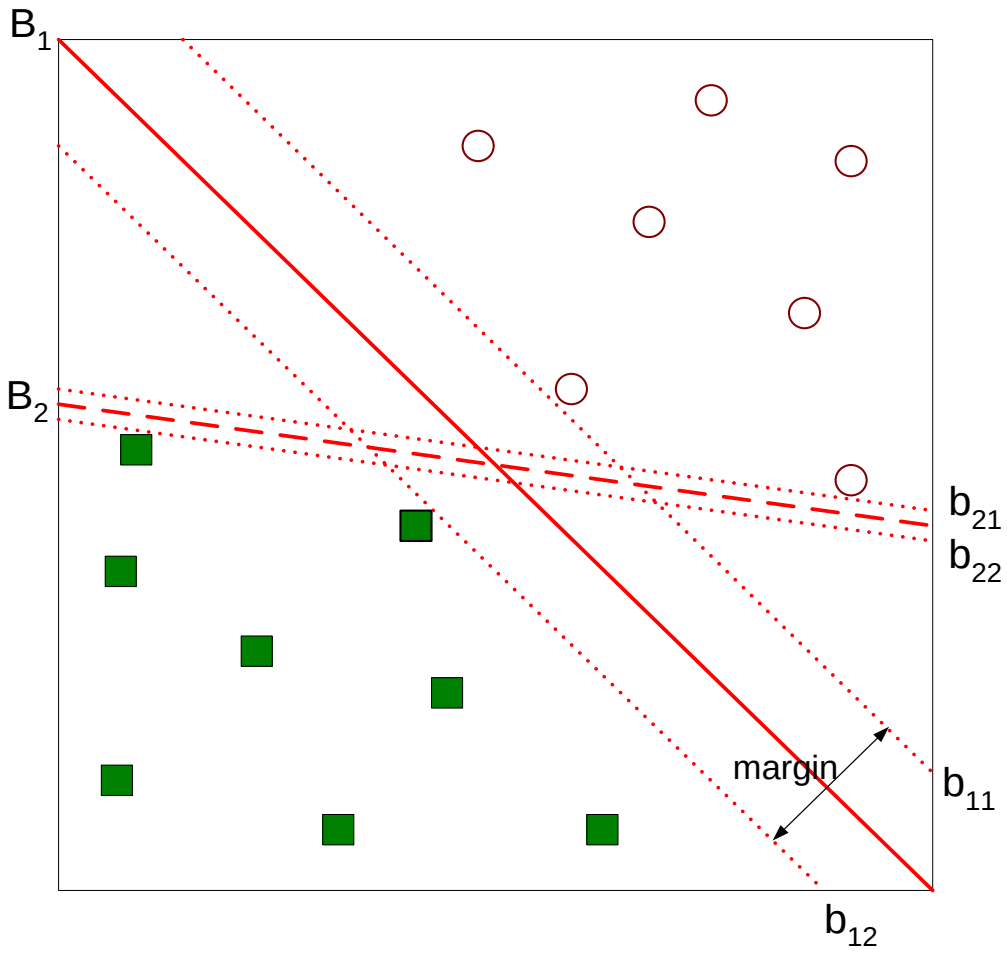
- Other possible solutions

Support Vector Machines



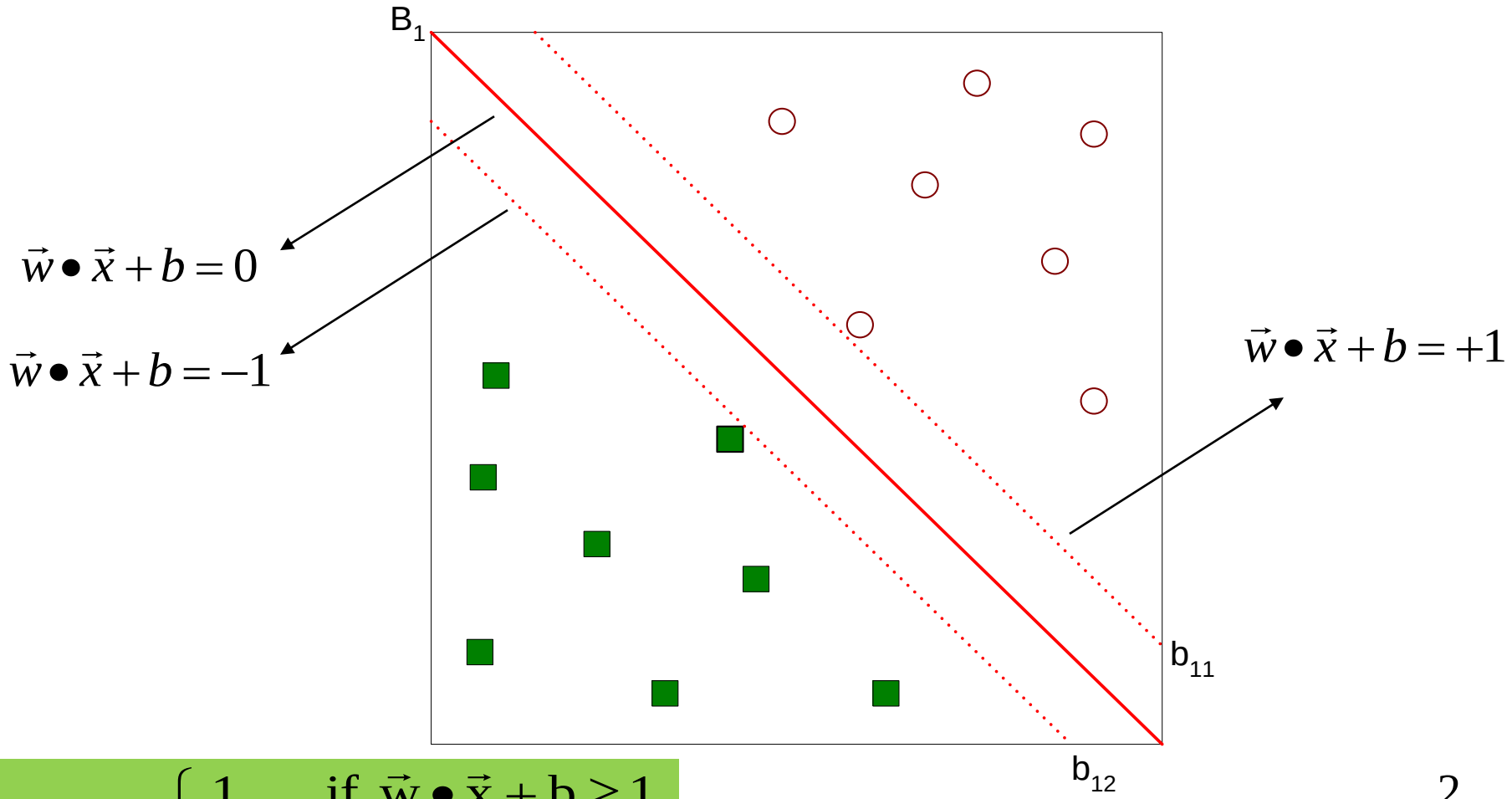
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



- Find hyperplane maximizes the margin => B1 is better than B2

Support Vector Machines



$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\| \vec{w} \|}$$

Support Vector Machines

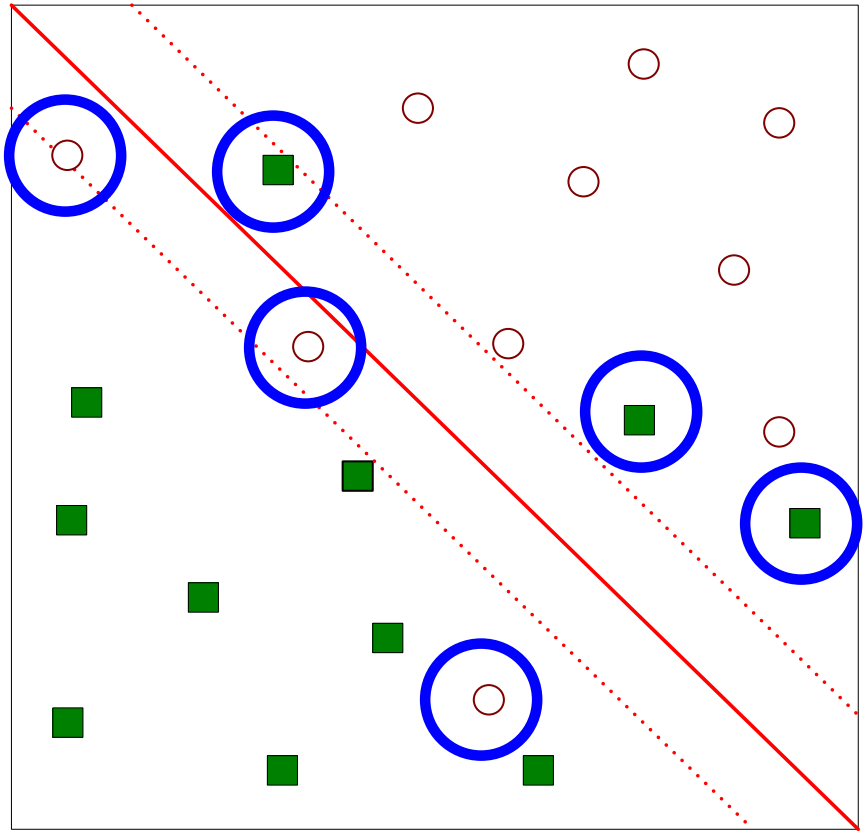
- We want to maximize: $\text{Margin} = \frac{2}{\| \vec{w} \|}$
  - Which is equivalent to minimizing: $L(w) = \frac{\| \vec{w} \|^2}{2}$
  - But subjected to the following constraints:

$$\vec{w} \cdot \vec{x_i} + b \geq 1 \text{ if } y_i = 1$$
$$\vec{w} \cdot \vec{x_i} + b \leq -1 \text{ if } y_i = -1$$

    - This is a constrained optimization problem
      - Numerical approaches to solve it (e.g., quadratic programming)

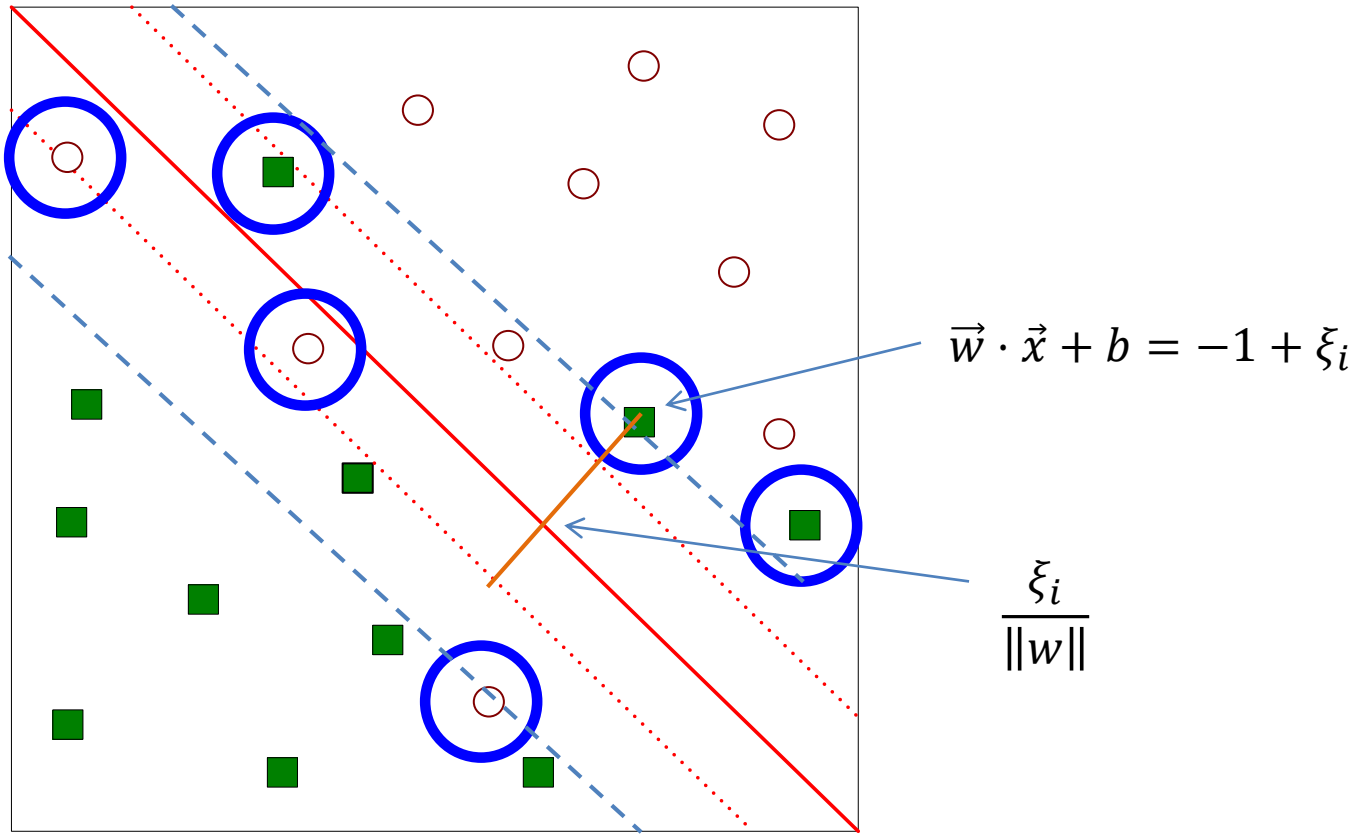
Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

- What if the problem is not linearly separable?



Support Vector Machines

- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize:

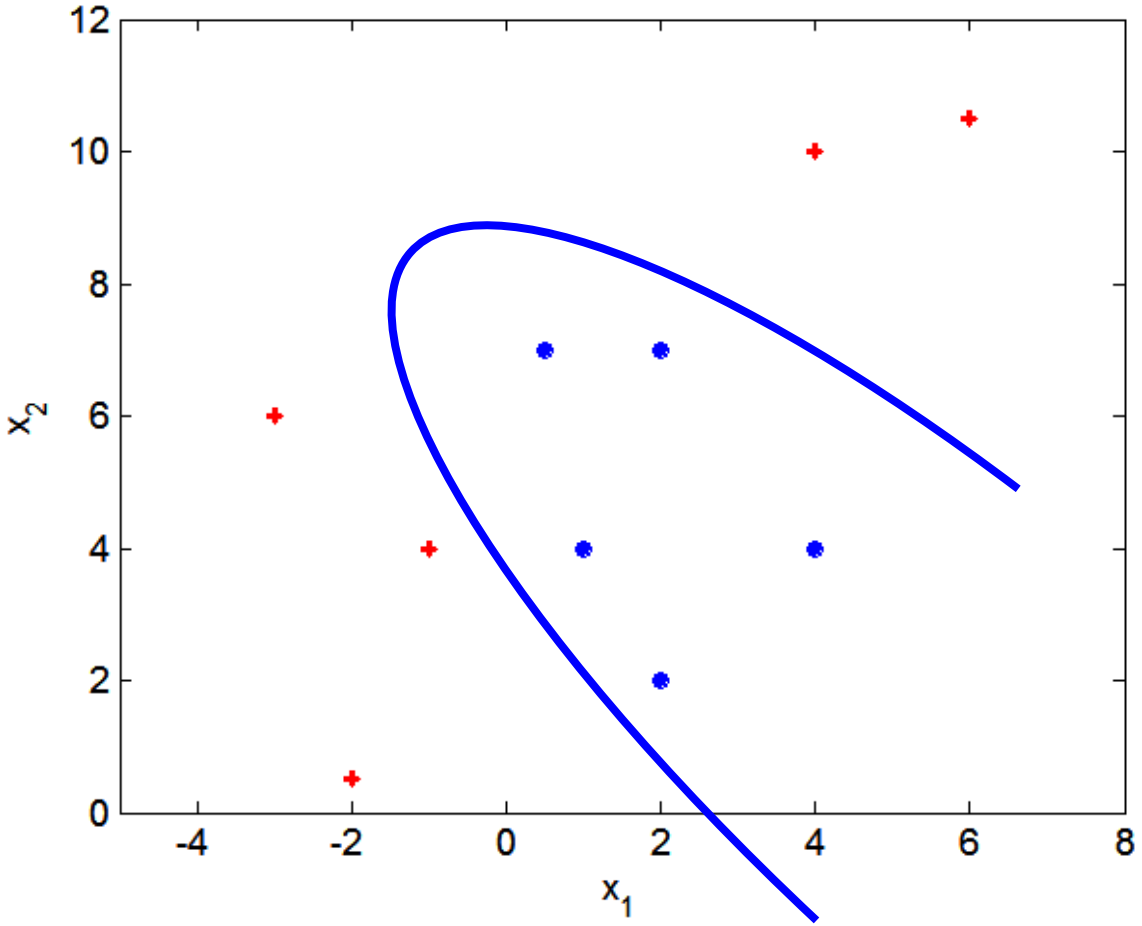
$$L(w) = \frac{\| \vec{w} \|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k \right)$$

    - Subject to:

$$\vec{w} \cdot \vec{x_i} + b \geq 1 - \xi_i \text{ if } y_i = 1$$
$$\vec{w} \cdot \vec{x_i} + b \leq -1 + \xi_i \text{ if } y_i = -1$$

Nonlinear Support Vector Machines

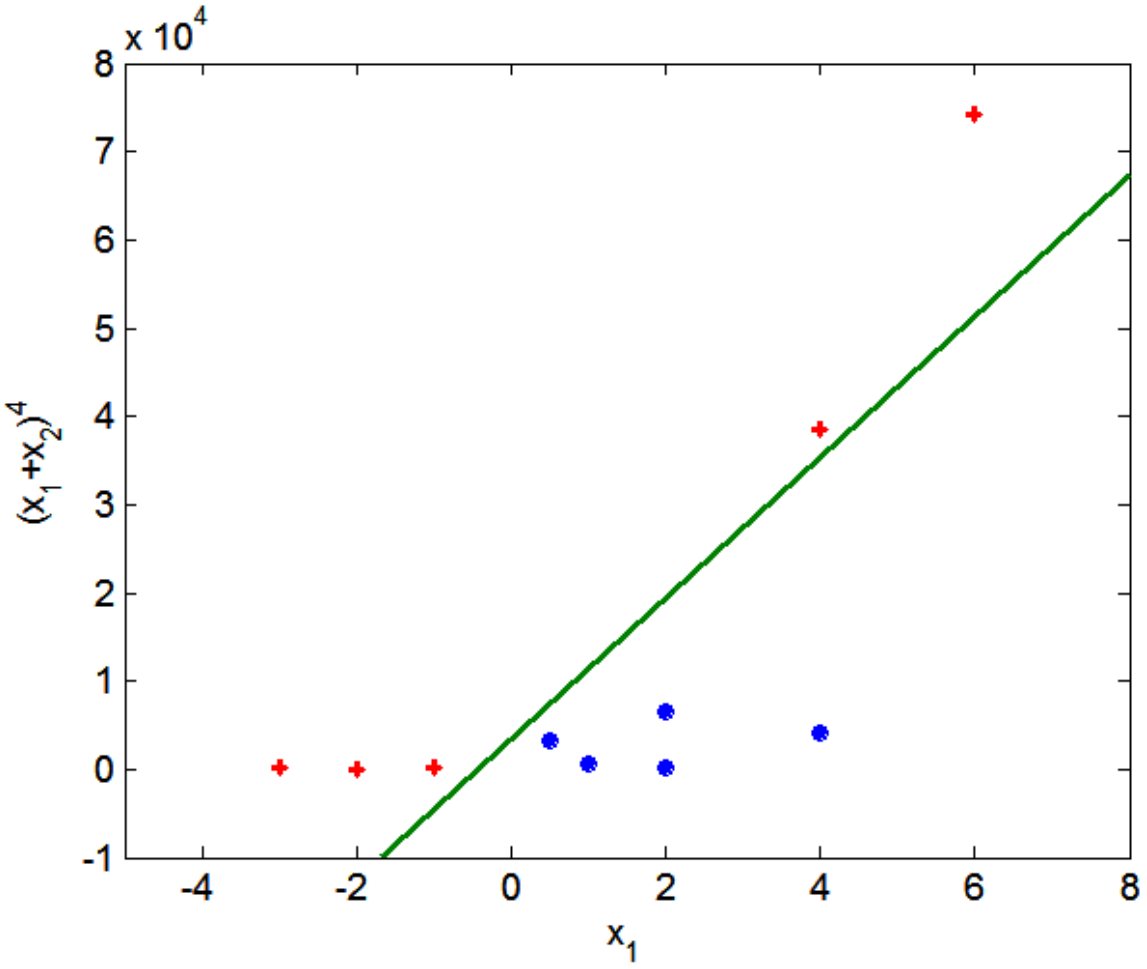
- What if decision boundary is not linear?



Nonlinear Support Vector Machines

- Transform data into higher dimensional space

Use the Kernel Trick



LOGISTIC REGRESSION

Classification via regression

- Instead of predicting the class of an record we want to predict the probability of the class given the record
- The problem of predicting continuous values is called regression problem
- General approach: find a continuous function that models the continuous points.

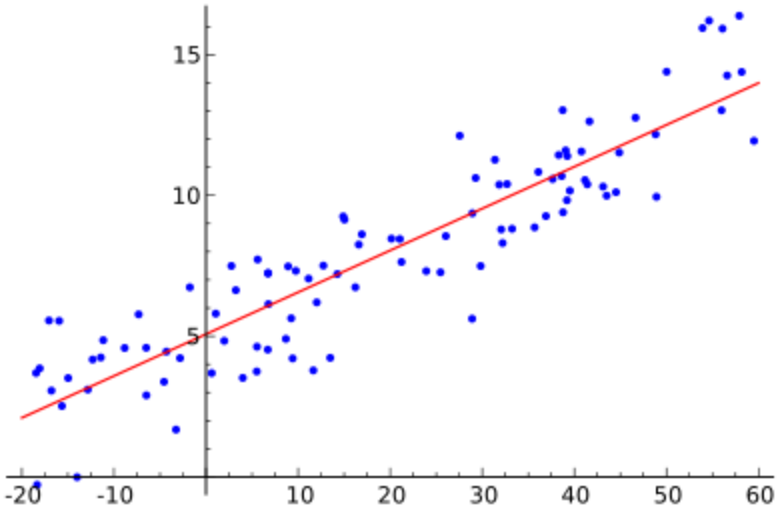
Example: Linear regression

- Given a dataset of the form $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ find a linear function that given the vector x_i predicts the y_i value as $y_i' = w^T x_i$
  - Find a vector of weights w that minimizes the sum of square errors

$$\sum_i (y_i' - y_i)^2$$

  - Closed form solution:

$$w = (X^T X)^{-1} X^T y$$



Classification via regression

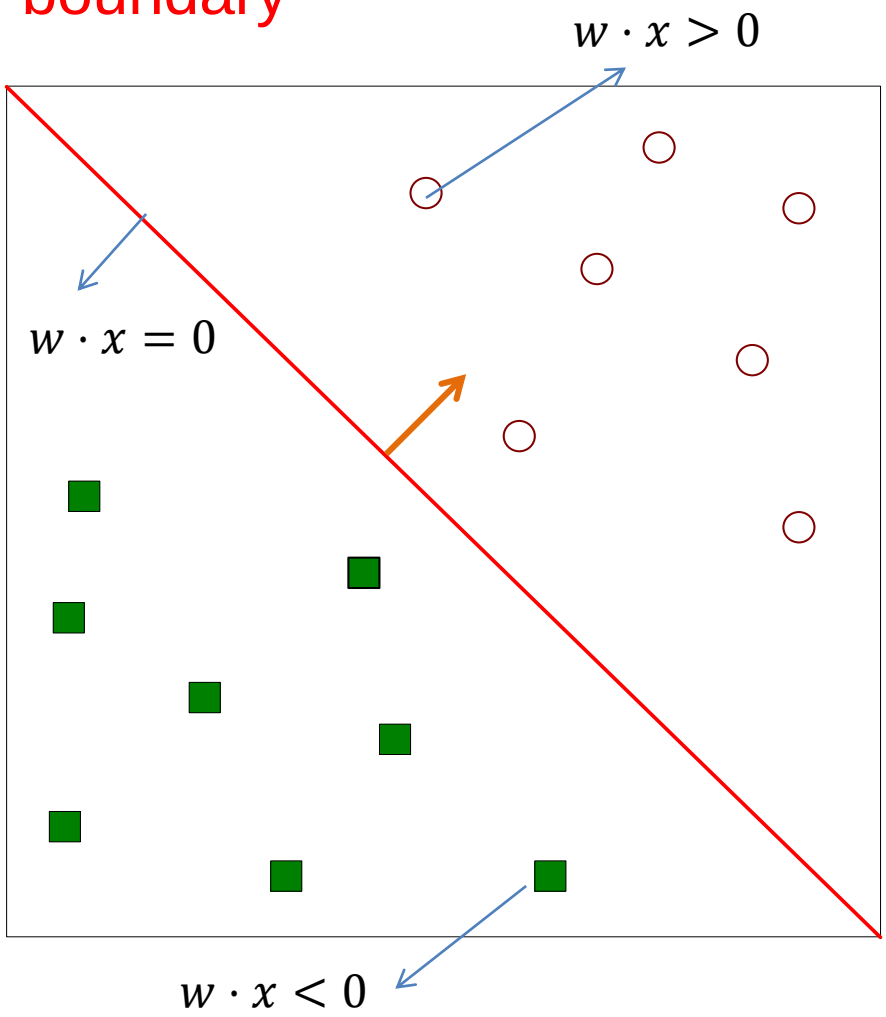
- Assume a linear classification boundary

For the positive class the bigger the value of $w \cdot x$, the further the point is from the classification boundary, the higher our certainty for the membership to the positive class

- Define $P(C_+|x)$ as an increasing function of $w \cdot x$

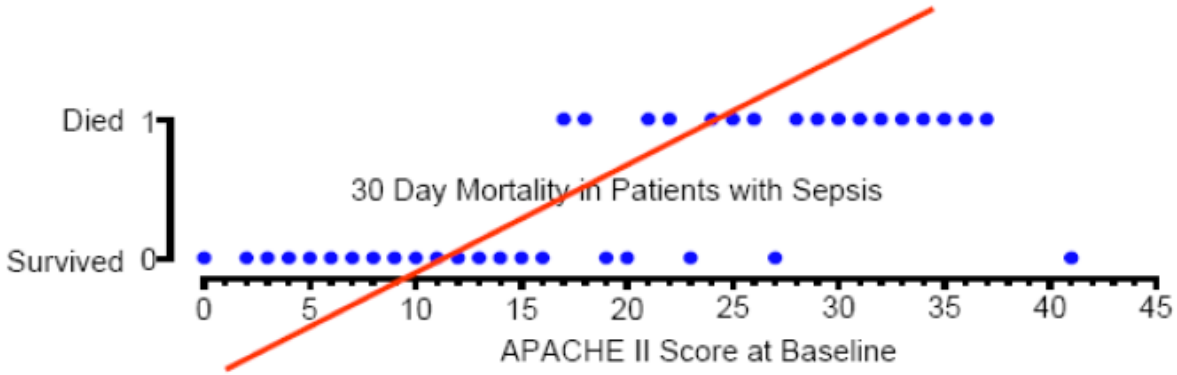
For the negative class the smaller the value of $w \cdot x$, the further the point is from the classification boundary, the higher our certainty for the membership to the negative class

- Define $P(C_-|x)$ as a decreasing function of $w \cdot x$



Linear regression

- A linear function is not good
  - It may produce negative probabilities, or probabilities that are greater than 1.



Logistic Regression

The logistic function

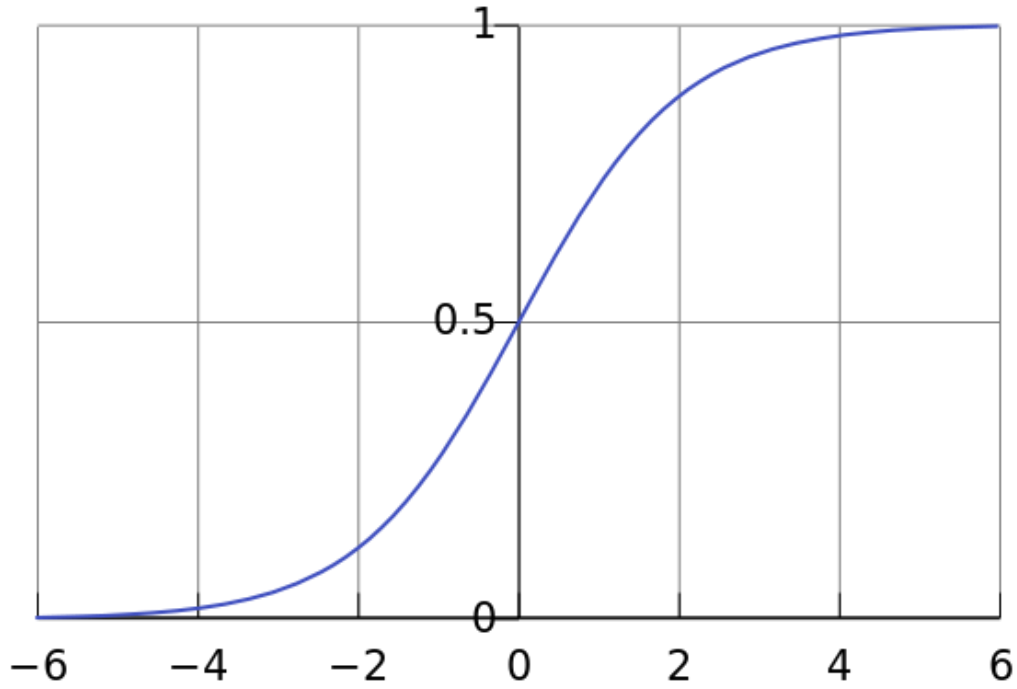
$$f(t) = \frac{1}{1 + e^{-t}}$$

$$P(C_+|x) = \frac{1}{1 + e^{-w \cdot x - a}}$$

$$P(C_-|x) = \frac{e^{-w \cdot x - a}}{1 + e^{-w \cdot x - a}}$$

$$\log \frac{P(C_+|x)}{P(C_-|x)} = w \cdot x + a$$

Linear regression on the log-odds ratio

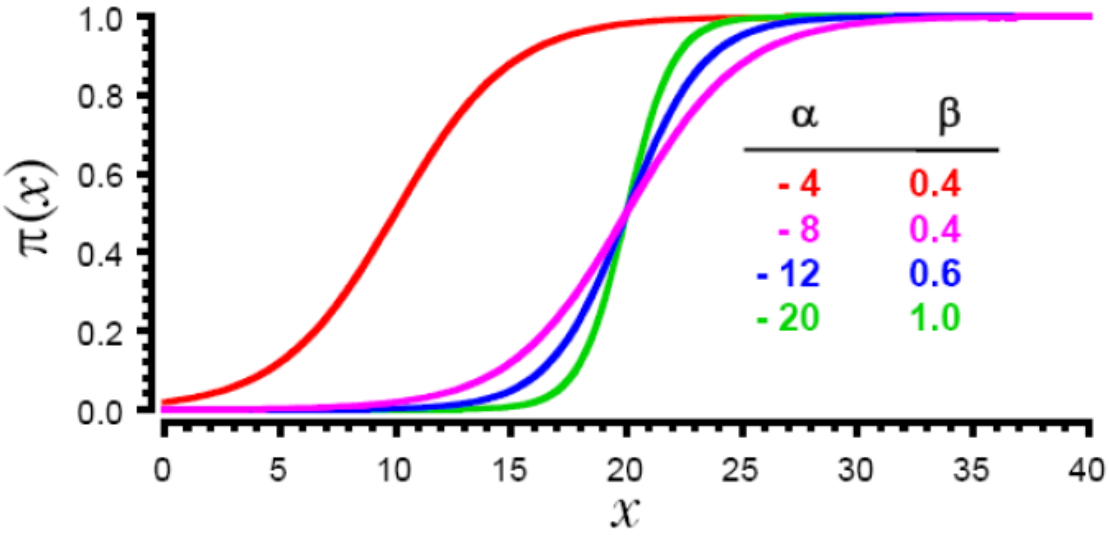


Logistic Regression: Find the vector w that maximizes the probability of the observed data

The logistic function

β controls the slope
α controls the position of the turning point

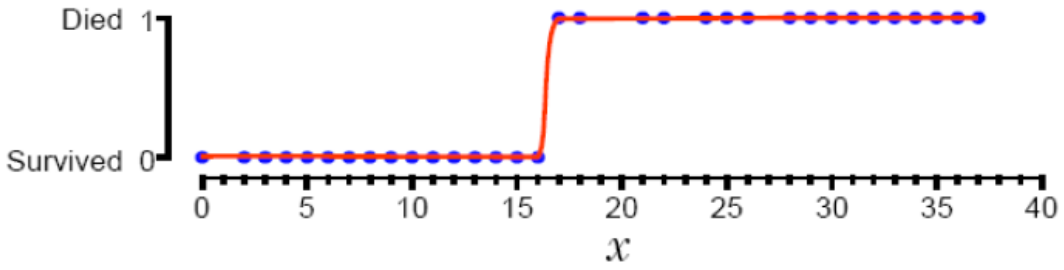
$$\pi(x) = \exp(\alpha + \beta x) / (1 + \exp(\alpha + \beta x))$$



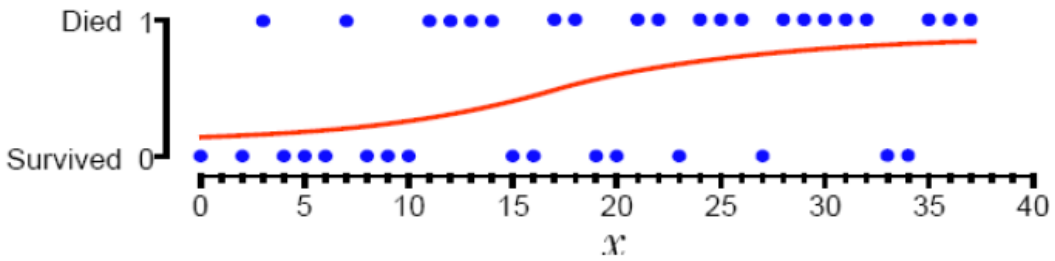
When $x = -\alpha / \beta$, $\alpha + \beta x = 0$ and hence $\pi(x) = 1/(1+1) = 0.5$

Logistic Regression in one dimension

Data that has a sharp survival cut off point between patients who live or die should have a large value of β.



Data with a lengthy transition from survival to death should have a low value of β.



Logistic Regression in one dimension

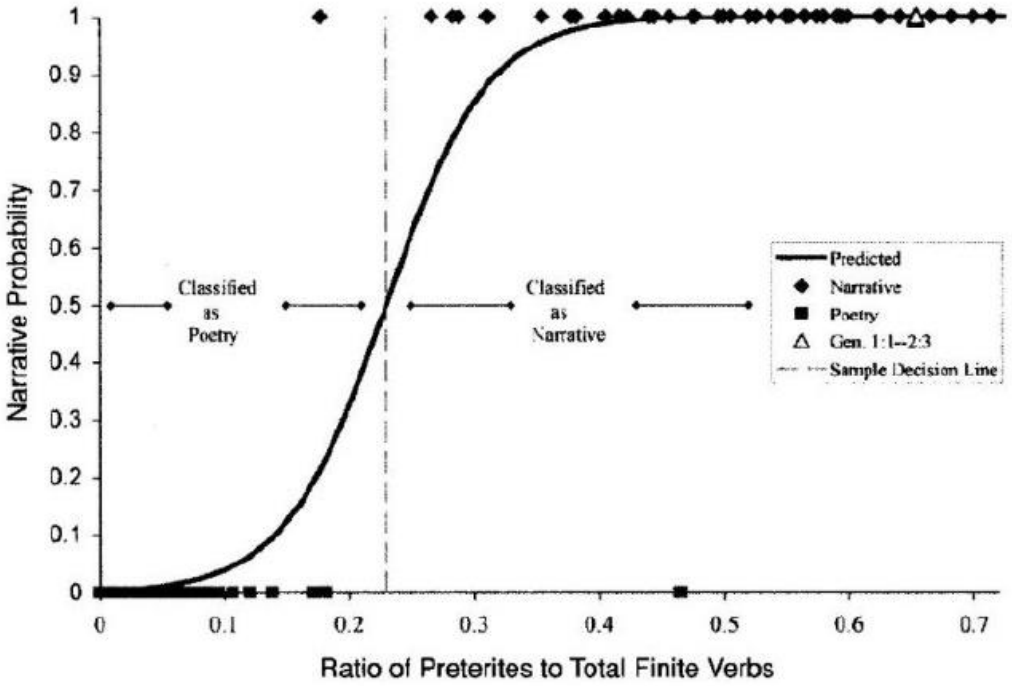
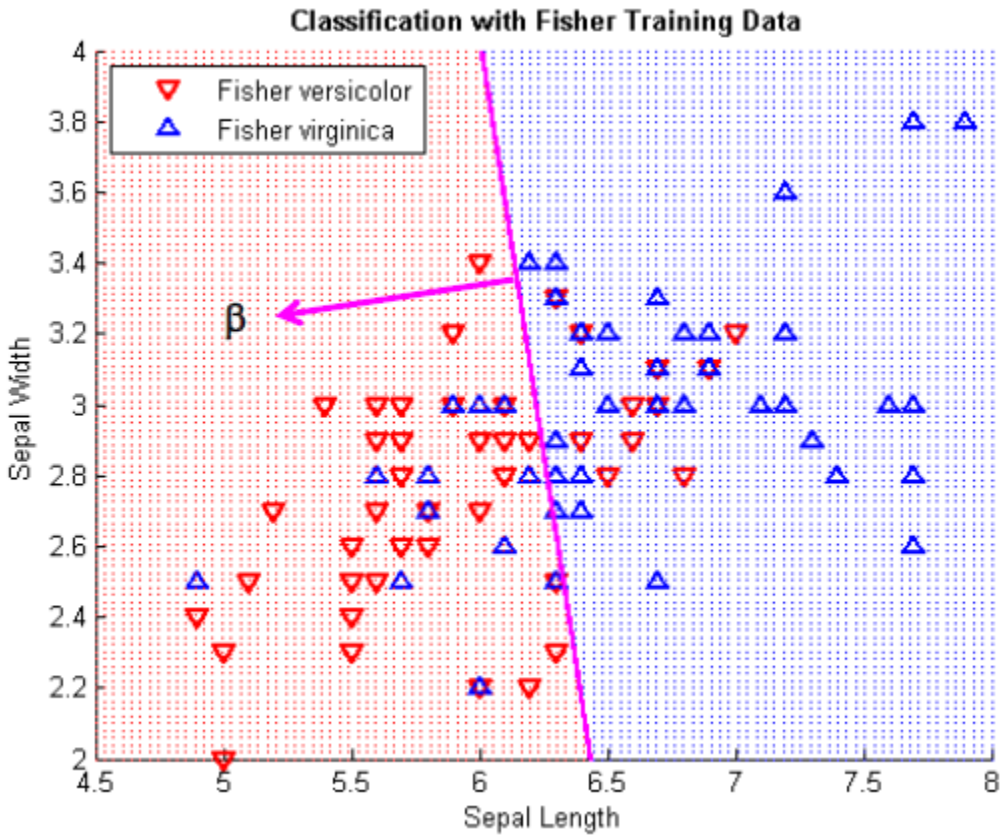


Figure 10-3. The solid curved line is called a logistic regression curve. The vertical axis measures the probability that an Old Testament passage is narrative, based on the use of preterite verbs. The probability is zero for poetry and unity or one for narrative. Passages with high preterite verb counts, falling to the right of the vertical dotted line, are likely narrative. The triangle on the upper right represents Genesis 1:1–2:3, which is clearly literal, narrative history.

Logistic regression in 2-d

Coefficients

$\beta_1 = -1.9$
$\beta_2 = -0.4$
$\alpha = 13.04$



Estimating the coefficients

- Maximum Likelihood Estimation:
  - We have pairs of the form (x_i, y_i)
- Log Likelihood function

$$L(w) = \sum_i [y_i \log P(y_i|x_i, w) + (1 - y_i) \log(1 - P(y_i|x_i, w))]$$

- Unfortunately it does not have a closed form solution
  - Use gradient descend to find local minimum

Logistic Regression

- Produces a probability estimate for the class membership which is often very useful.
- The weights can be useful for understanding the feature importance.
- Works for relatively large datasets
- Fast to apply.

NAÏVE BAYES CLASSIFIER

Bayes Classifier

- A probabilistic framework for solving classification problems
- **A, C** random variables
- Joint probability: **Pr(A=a,C=c)**
- Conditional probability: **Pr(C=c | A=a)**
- Relationship between joint and conditional probability distributions

$$\Pr(C, A) = \Pr(C \mid A) \times \Pr(A) = \Pr(A \mid C) \times \Pr(C)$$

- **Bayes Theorem**:

$$P(C \mid A) = \frac{P(A \mid C)P(C)}{P(A)}$$

Bayesian Classifiers

- How to classify the new record X = ('Yes', 'Single', 80K)

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Find the class with the highest probability given the vector values.

Maximum Aposteriori Probability estimate:

- Find the value c for class C that maximizes P(C=c| X)

How do we estimate P(C|X) for the different values of C?

- We want to estimate P(C=Yes| X)
- and P(C=No| X)

Bayesian Classifiers

- In order for probabilities to be well defined:
  - Consider each attribute and the class label as random variables
  - Probabilities are determined from the data

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Evade C

Event space: {Yes, No}

$P(C) = (0.3, 0.7)$

Refund A_1

Event space: {Yes, No}

$P(A_1) = (0.3,0.7)$

Martial Status A_2

Event space: {Single, Married, Divorced}

$P(A_2) = (0.4,0.4,0.2)$

Taxable Income A_3

Event space: R

$P(A_3) \sim Normal(\mu,\sigma^2)$

$\mu = 104$:sample mean, $\sigma^2=1874$:sample var

Bayesian Classifiers

- Approach:
  - compute the posterior probability $P(C \mid A_1, A_2, \ldots, A_n)$ using the Bayes theorem

$$P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}$$

  - Maximizing

    $P(C \mid A_1, A_2, \ldots, A_n)$

    is equivalent to maximizing

    $P(A_1, A_2, \ldots, A_n \mid C)\ P(C)$

    - The value $P(A_1, \ldots, A_n)$ is the same for all values of C.

- How to estimate $P(A_1, A_2, \ldots, A_n \mid C)$?

Naïve Bayes Classifier

- Assume conditional independence among attributes A_i when class C is given:
  - $P(A_1, A_2, \ldots, A_n|C) = P(A_1|C)\, P(A_2|C) \cdots P(A_n|C)$
  - We can estimate $P(A_i|\, C)$ from the data.
  - New point $X = (A_1 = \alpha_1, \ldots, A_n = \alpha_n)$ is classified to class c if

$$P(C = c|X) = P(C = c) \prod_i P(A_i = \alpha_i|c)$$

  is maximum over all possible values of C.

Example

- Record

  X = (Refund = Yes, Status = Single, Income =80K)

- For the class C = 'Evade', we want to compute:

  P(C = Yes|X) and P(C = No| X)

- We compute:
  - P(C = Yes|X) = P(C = Yes)*P(Refund = Yes |C = Yes)
    *P(Status = Single |C = Yes)
    *P(Income =80K |C= Yes)
  - P(C = No|X) = P(C = No)*P(Refund = Yes |C = No)
    *P(Status = Single |C = No)
    *P(Income =80K |C= No)

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Class Prior Probability:

$$P(C = c) = \frac{N_c}{N}$$

N_c: Number of records with class c

N = Number of records

P(C = No) = 7/10

P(C = Yes) = 3/10

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Refund = Yes|No) = 3/7

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Refund = Yes|Yes) = 0

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Status=Single|No) = 2/7

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Discrete attributes:

$$P(A_i = a|C = c) = \frac{N_{a,c}}{N_c}$$

$N_{a,c}$: number of instances having attribute $A_i = a$ and belong to class c

N_c: number of instances of class c

P(Status=Single|Yes) = 2/3

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Normal distribution:

$$P(A_i = a \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(a-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - One for each (A_i, c_j) pair

- For Class=No
  - sample mean μ = 110
  - sample variance $\sigma^2 = 2975$
- For Income = 80

$$P(Income = 80 \mid No) = \frac{1}{\sqrt{2\pi}(54.54)} e^{-\frac{(80-110)^2}{2(2975)}} = 0.0062$$

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Normal distribution:

$$P(A_i = a \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(a-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

  - One for each (A_i, c_j) pair

- For Class=Yes
  - sample mean μ = 90
  - sample variance $\sigma^2 = 2975$
- For Income = 80

$$P(Income = 80 \mid Yes) = \frac{1}{\sqrt{2\pi}(5)} e^{-\frac{(80-90)^2}{2(25)}} = 0.01$$

Example

- Record

  X = (Refund = Yes, Status = Single, Income =80K)

- We compute:
  - P(C = Yes|X) = P(C = Yes)*P(Refund = Yes |C = Yes)
    *P(Status = Single |C = Yes)
    *P(Income =80K |C= Yes)
    = 3/10* 0 * 2/3 * 0.01 = 0
  - P(C = No|X) = P(C = No)*P(Refund = Yes |C = No)
    *P(Status = Single |C = No)
    *P(Income =80K |C= No)
    = 7/10 * 3/7 * 2/7 * 0.0062 = 0.0005

Example of Naïve Bayes Classifier

- Creating a Naïve Bayes Classifier, essentially means to compute counts:

Total number of records: N = 10

Class No:
Number of records: 7
Attribute Refund:
Yes: 3
No: 4
Attribute Marital Status:
Single: 2
Divorced: 1
Married: 4
Attribute Income:
mean: 110
variance: 2975

Class Yes:
Number of records: 3
Attribute Refund:
Yes: 0
No: 3
Attribute Marital Status:
Single: 2
Divorced: 1
Married: 0
Attribute Income:
mean: 90
variance: 25

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1

P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

Example of Naïve Bayes Classifier

Given a Test Record:

X = (Refund = Yes, Status = Single, Income =80K)

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1

P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=Yes|Class=No)
  × P(Married| Class=No)
  × P(Income=120K| Class=No)
  = 3/7 * 2/7 * 0.0062 = 0.00075

- P(X|Class=Yes) = P(Refund=No| Class=Yes)
  × P(Married| Class=Yes)
  × P(Income=120K| Class=Yes)
  = 0 * 2/3 * 0.01 = 0

- P(No) = 0.3, P(Yes) = 0.7

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)

=> Class = No

Naïve Bayes Classifier

- If one of the conditional probabilities is zero, then the entire expression becomes zero

- Laplace Smoothing:

$$P(A_i = a|C = c) = \frac{N_{ac} + 1}{N_c + N_i}$$

  - N_i: number of attribute values for attribute A_i

Example of Naïve Bayes Classifier

- Creating a Naïve Bayes Classifier, essentially means to compute counts:

With Laplace Smoothing

Total number of records: N = 10

Class No:
Number of records: 7
Attribute Refund:
Yes: 3
No: 4
Attribute Marital Status:
Single: 2
Divorced: 1
Married: 4
Attribute Income:
mean: 110
variance: 2975

Class Yes:
Number of records: 3
Attribute Refund:
Yes: 0
No: 3
Attribute Marital Status:
Single: 2
Divorced: 1
Married: 0
Attribute Income:
mean: 90
variance: 25

naive Bayes Classifier:

P(Refund=Yes|No) = 4/9
P(Refund=No|No) = 5/9
P(Refund=Yes|Yes) = 1/5
P(Refund=No|Yes) = 4/5

P(Marital Status=Single|No) = 3/10
P(Marital Status=Divorced|No)=2/10
P(Marital Status=Married|No) = 5/10
P(Marital Status=Single|Yes) = 3/6
P(Marital Status=Divorced|Yes)=2/6
P(Marital Status=Married|Yes) = 1/6

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

Example of Naïve Bayes Classifier

Given a Test Record:

With Laplace Smoothing

X = (Refund = Yes, Status = Single, Income =80K)

naive Bayes Classifier:

P(Refund=Yes|No) = 4/9
P(Refund=No|No) = 5/9
P(Refund=Yes|Yes) = 1/5
P(Refund=No|Yes) = 4/5

P(Marital Status=Single|No) = 3/10
P(Marital Status=Divorced|No)=2/10
P(Marital Status=Married|No) = 5/10
P(Marital Status=Single|Yes) = 3/6
P(Marital Status=Divorced|Yes)=2/6
P(Marital Status=Married|Yes) = 1/6

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- P(X|Class=No) = P(Refund=No|Class=No)
  × P(Married| Class=No)
  × P(Income=120K| Class=No)
  = 4/9 × 3/10 × 0.0062 = 0.00082

- P(X|Class=Yes) = P(Refund=No| Class=Yes)
  × P(Married| Class=Yes)
  × P(Income=120K| Class=Yes)
  = 1/5 × 3/6 × 0.01 = 0.001

- P(No) = 0.7, P(Yes) = 0.3
- P(X|No)P(No) = 0.0005
- P(X|Yes)P(Yes) = 0.0003

=> Class = No

Implementation details

- Computing the conditional probabilities involves multiplication of many very small numbers
  - Numbers get very close to zero, and there is a danger of numeric instability
- We can deal with this by computing the logarithm of the conditional probability

$$\log P(C|A) \sim \log P(A|C) + \log P(C)$$
$$= \sum_i \log P(A_i|C) + \log P(C)$$

Naïve Bayes for Text Classification

- Naïve Bayes is commonly used for text classification
- For a document with k terms $d = (t_1, \dots, t_k)$

$$P(c|d) = P(c)P(d|c) = P(c)\prod_{t_i \in d} P(t_i|c)$$

Fraction of documents in c

- $P(t_i|c)$= Fraction of terms from all documents in c that are t_i.

$$P(t_i|c) = \frac{N_{ic} + 1}{N_c + T}$$

Number of times t_i appears in all documents in c

Laplace Smoothing

Total number of terms in all documents in c

Number of unique words (vocabulary size)

- Easy to implement and works relatively well
- Limitation: Hard to incorporate additional features (beyond words).
  - E.g., number of adjectives used.

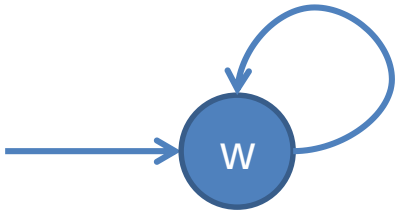
Multinomial document model

- Probability of document $d = (t_1, \ldots, t_k)$ in class c:

$$P(d|c) = P(c) \prod_{t_i \in d} P(t_i|c)$$

- This formula assumes a multinomial distribution for the document generation:
  - If we have probabilities $p_1, \ldots, p_T$ for events $t_1, \ldots, t_T$ the probability of a subset of these is

$$P(d) = \frac{N}{N_{t_1}!\, N_{t_2}! \cdots N_{t_T}!} p_1^{N_{t_1}} p_2^{N_{t_2}} \cdots p_T^{N_{t_T}}$$



- Equivalently: There is an automaton spitting words from the above distribution

TRAINMULTINOMIALNB($\mathbb{C}, \mathbb{D}$)

1 $V \leftarrow$ EXTRACTVOCABULARY($\mathbb{D}$)
2 $N \leftarrow$ COUNTDOCS($\mathbb{D}$)
3 **for each** $c \in \mathbb{C}$
4 **do** $N_c \leftarrow$ COUNTDOCSINCLASS($\mathbb{D}, c$)
5 $prior[c] \leftarrow N_c/N$
6 $text_c \leftarrow$ CONCATENATETEXTOFALLDOCSINCLASS($\mathbb{D}, c$)
7 **for each** $t \in V$
8 **do** $T_{ct} \leftarrow$ COUNTTOKENSOFTERM($text_c, t$)
9 **for each** $t \in V$
10 **do** $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
11 **return** $V, prior, condprob$

APPLYMULTINOMIALNB($\mathbb{C}, V, prior, condprob, d$)

1 $W \leftarrow$ EXTRACTTOKENSFROMDOC(V, d)
2 **for each** $c \in \mathbb{C}$
3 **do** $score[c] \leftarrow \log prior[c]$
4 **for each** $t \in W$
5 **do** $score[c] \mathrel{+}= \log condprob[t][c]$
6 **return** $\arg\max_{c \in \mathbb{C}} score[c]$

► **Figure 13.2** Naive Bayes algorithm (multinomial model): Training and testing.

Example

News titles for Politics and Sports

	Politics	Sports
documents	"Obama meets Merkel" "Obama elected again" "Merkel visits Greece again"	"OSFP European basketball champion" "Miami NBA basketball champion" "Greece basketball coach?"
	P(p) = 0.5	P(s) = 0.5
terms Vocabulary size: 14	obama:2, meets:1, merkel:2, elected:1, again:2, visits:1, greece:1	OSFP:1, european:1, basketball:3, champion:2, miami:1, nba:1, greece:1, coach:1
	Total terms: 10	Total terms: 11

New title: X = "Obama likes basketball"

P(Politics|X) ~ P(p)*P(obama|p)*P(likes|p)*P(basketball|p)
= 0.5 * 3/(10+14) *1/(10+14) * 1/(10+14) = 0.000108

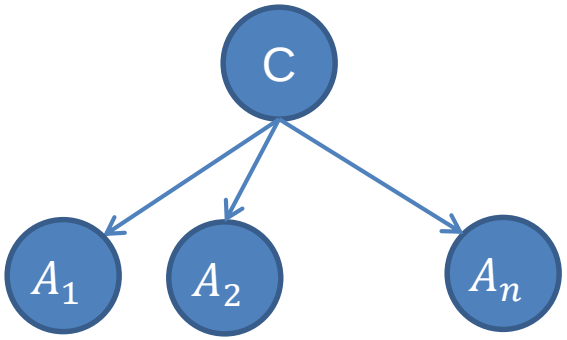
P(Sports|X) ~ P(s)*P(obama|s)*P(likes|s)*P(basketball|s)
= 0.5 * 1/(11+14) *1/(11+14) * 4/(11+14) = 0.000128

Naïve Bayes (Summary)

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Belief Networks (BBN)
- Naïve Bayes can produce a probability estimate, but it is usually a very biased one
  - Logistic Regression is better for obtaining probabilities.

Generative vs Discriminative models

- Naïve Bayes is a type of a generative model
  - Generative process:
    - First pick the category of the record
    - Then given the category, generate the attribute values from the distribution of the category
    - Conditional independence given C



- We use the training data to learn the distribution of the values in a class

Generative vs Discriminative models

- Logistic Regression and SVM are discriminative models
  - The goal is to find the boundary that discriminates between the two classes from the training data

- In order to classify the language of a document, you can
  - Either learn the two languages and find which is more likely to have generated the words you see
  - Or learn what differentiates the two languages.

SUPERVISED LEARNING

Learning

- Supervised Learning: learn a model from the data using labeled data.
  - Classification and Regression are the prototypical examples of supervised learning tasks. Other are possible (e.g., ranking)
- Unsupervised Learning: learn a model – extract structure from unlabeled data.
  - Clustering and Association Rules are prototypical examples of unsupervised learning tasks.
- Semi-supervised Learning: learn a model for the data using both labeled and unlabeled data.

Supervised Learning Steps

- Model the problem
  - What is you are trying to predict? What kind of optimization function do you need? Do you need classes or probabilities?
- Extract Features
  - How do you find the right features that help to discriminate between the classes?
- Obtain training data
  - Obtain a collection of labeled data. Make sure it is large enough, accurate and representative. Ensure that classes are well represented.
- Decide on the technique
  - What is the right technique for your problem?
- Apply in practice
  - Can the model be trained for very large data? How do you test how you do in practice? How do you improve?

Modeling the problem

- Sometimes it is not obvious. Consider the following three problems
  - Detecting if an email is spam
  - Categorizing the queries in a search engine
  - Ranking the results of a web search
  - Predicting the reply to a question.

Feature extraction

- Feature extraction, or feature engineering is the most tedious but also the most important step
  - How do you separate the players of the Greek national team from those of the Swedish national team?
- One line of thought: throw features to the classifier and the classifier will figure out which ones are important
  - More features, means that you need more training data
- Another line of thought: Feature Selection: Select carefully the features using various functions and techniques
  - Computationally intensive
- Deep Neural Networks
  - They extract features from very basic representation.

Training data

- An overlooked problem: How do you get labeled data for training your model?
  - E.g., how do you get training data for ranking?
  - Chicken and egg problem
- Usually requires a lot of manual effort and domain expertise and carefully planned labeling
  - Results are not always of high quality (lack of expertise)
  - And they are not sufficient (low coverage of the space)
- Recent trends:
  - Find a source that generates the labeled data for you.
  - Crowd-sourcing techniques

Dealing with small amount of labeled data

- Semi-supervised learning techniques have been developed for this purpose.
- Self-training: Train a classifier on the data, and then feed back the high-confidence output of the classifier as input
- Co-training: train two “independent” classifiers and feed the output of one classifier as input to the other.
- Regularization: Treat learning as an optimization problem where you define relationships between the objects you want to classify, and you exploit these relationships
  - Example: Image restoration

Technique

- The choice of technique depends on the problem requirements (do we need a probability estimate?) and the problem specifics (does independence assumption hold? do we think classes are linearly separable?)
- For many cases finding the right technique may be trial and error
- For many cases the exact technique does not matter.

Big Data Trumps Better Algorithms

- If you have enough data then the algorithms are not so important

- The web has made this possible.
  - Especially for text-related tasks
  - Search engine uses the collective human intelligence

Google lecture:
Theorizing from the Data

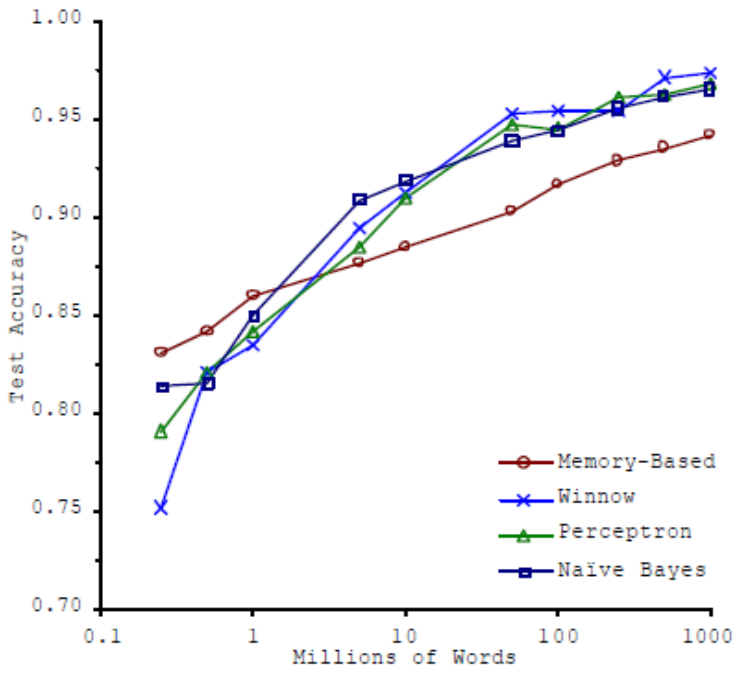


Figure 1. Learning Curves for Confusion Set Disambiguation

Apply-Test

- How do you scale to very large datasets?
  - Distributed computing – map-reduce implementations of machine learning algorithms (Mahaut, over Hadoop)
- How do you test something that is running online?
  - You cannot get labeled data in this case
  - A/B testing
- How do you deal with changes in data?
  - Active learning