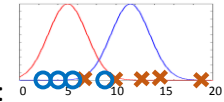


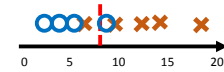
Logistic Classifier

CISC 5800
Professor Daniel Leeds

Classification strategy:
generative vs. discriminative



- Generative, e.g., Bayes/Naïve Bayes:
 - Identify probability distribution for each class
 - Determine class with maximum probability for data example
- Discriminative, e.g., Logistic Regression:
 - Identify boundary between classes
 - Determine which side of boundary new data example exists on



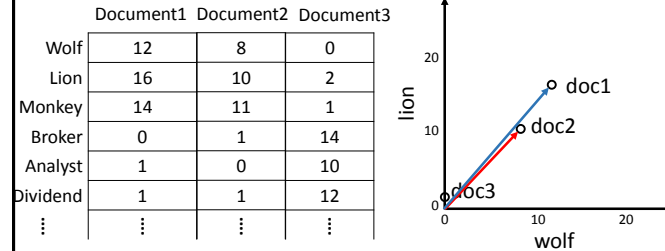
Linear algebra: data features

- Vector – list of numbers:
each number describes a data **feature**
- Matrix – list of lists of numbers:
features for each data point

	Document 1	Document 2	Document 3
Wolf	12	8	0
Lion	16	10	2
Monkey	14	11	1
Broker	0	1	14
Analyst	1	0	10
Dividend	1	1	12
⋮	⋮	⋮	⋮

Feature space

- Each data feature defines a dimension in space

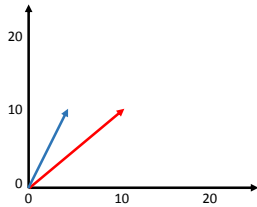


The dot product

The dot product compares two vectors:

$$\mathbf{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

$$\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^n a_i b_i = \mathbf{a}^T \mathbf{b}$$



$$\begin{bmatrix} 5 \\ 10 \end{bmatrix} \cdot \begin{bmatrix} 10 \\ 10 \end{bmatrix} = 5 \times 10 + 10 \times 10 \\ = 50 + 100 = 150$$

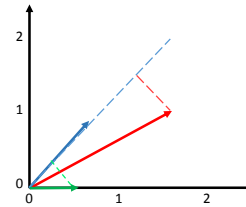
5

The dot product, continued $\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^n a_i b_i$

Magnitude of a vector is sum of squares of the elements

$$|\mathbf{a}| = \sqrt{\sum_i a_i^2}$$

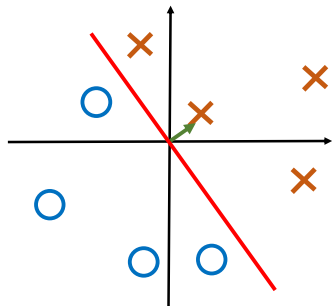
If \mathbf{a} has unit magnitude, $\mathbf{a} \cdot \mathbf{b}$ is “projection” of \mathbf{b} onto \mathbf{a}



$$\begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix} \cdot \begin{bmatrix} 1.5 \\ 1 \end{bmatrix} = 0.6 \times 1.5 + 0.8 \times 1 \\ = 0.9 + 0.8 = 1.7$$

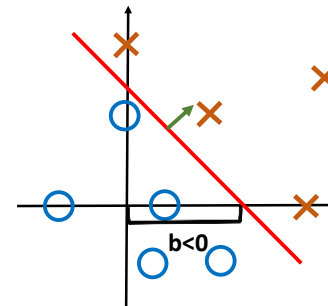
6

Separating boundary, defined by \mathbf{w}



- Separating **hyperplane** splits **class 0** and **class 1**
- Plane is defined by line \mathbf{w} perpendicular to plan
- Is data point \mathbf{x} in class 0 or class 1? $\mathbf{w}^T \mathbf{x} + b > 0$ class **1**
 $\mathbf{w}^T \mathbf{x} + b < 0$ class **0**

Separating boundary, defined by \mathbf{w} and b



Example:

$$\mathbf{w} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad b = -4$$

$$\mathbf{x}^1 = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad \mathbf{w}^T \mathbf{x}^1 + b = 1$$

$$\mathbf{x}^2 = \begin{bmatrix} 0 \\ 2 \end{bmatrix} \quad \mathbf{w}^T \mathbf{x}^2 + b = -2$$

9

Notational simplification

Recall: $\mathbf{w}^T \mathbf{x} = \mathbf{w} \cdot \mathbf{x} = \sum_{i=1}^n w_i x_i$

Define $x'_{1:n} = x_{1:n}$ and $x'_{n+1} = 1$ for all inputs \mathbf{x} and $w'_{1:n} = w_{1:n}$ and $w'_{n+1} = b$

Now $\mathbf{w}'^T \mathbf{x}' = \mathbf{w}^T \mathbf{x} + b$

Let's assume $x_{n+1}=1$ always, and $w_{n+1}=b$ always

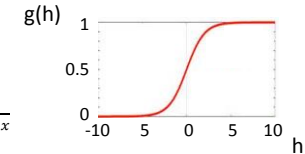
From real-number projection to 0/1 label

- Binary classification: 0 is class A, 1 is class B
- Sigmoid function stands in for $p(x|y)$

• Sigmoid: $g(h) = \frac{1}{1+e^{-h}}$

• $p(y = 0|x; \theta) = 1 - g(\mathbf{w}^T \mathbf{x}) = \frac{e^{-\mathbf{w}^T \mathbf{x}}}{1+e^{-\mathbf{w}^T \mathbf{x}}}$

• $p(y = 1|x; \theta) = g(\mathbf{w}^T \mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}^T \mathbf{x}}}$



$$\mathbf{w}^T \mathbf{x} = \sum_j w_j x_j + b$$

Learning parameters for classification

- Similar to MLE for Bayes classifier
- "Likelihood" for data points y^1, \dots, y^n (different from Bayesian likelihood)
 - If y^i in class A, $y^i=0$, multiply $(1-g(x^i; \mathbf{w}))$
 - If y^i in class B, $y^i=1$, multiply $(g(x^i; \mathbf{w}))$

$$\operatorname{argmax}_{\mathbf{w}} L(y|x; \mathbf{w}) = \prod_i (1 - g(x^i; \mathbf{w}))^{(1-y^i)} g(x^i; \mathbf{w})^{y^i}$$

12

Learning parameters for classification

The long derivation

- Similar to MLE for Bayes classifier
- "Likelihood" for data points y^1, \dots, y^n (different from Bayesian likelihood)
 - If y^i in class A, $y^i=0$, multiply $(1-g(x^i; \mathbf{w}))$
 - If y^i in class B, $y^i=1$, multiply $(g(x^i; \mathbf{w}))$

$$\operatorname{argmax}_{\mathbf{w}} L(y|x; \mathbf{w}) = \prod_i (1 - g(x^i; \mathbf{w}))^{(1-y^i)} g(x^i; \mathbf{w})^{y^i}$$

$$LL(y|x; \mathbf{w}) = \sum_i (1 - y^i) \log(1 - g(x^i; \mathbf{w})) + y^i \log(g(x^i; \mathbf{w}))$$

$$LL(y|x; \mathbf{w}) = \sum_i y^i \log \frac{g(x^i; \mathbf{w})}{1 - g(x^i; \mathbf{w})} + \log(1 - g(x^i; \mathbf{w}))$$

13

Learning parameters for classification

$$g(h) = \frac{1}{1 + e^{-h}}$$

$$LL(y|x; w) = \sum_i y^i \log \frac{g(x^i; w)}{1 - g(x^i; w)} + \log(1 - g(x^i; w))$$

$$LL(y|x; w) = \sum_i y^i \log \frac{1}{1 + e^{-w^T x^i}} + \log \left(\frac{e^{-w^T x^i}}{1 + e^{-w^T x^i}} \right)$$

$$LL(y|x; w) = \sum_i y^i \log \frac{1}{1 + e^{-w^T x^i - 1}} + \log \left(\frac{e^{-w^T x^i}}{1 + e^{-w^T x^i}} \right)$$

$$LL(y|x; w) = \sum_i y^i w^T x^i - w^T x^i - \log(1 + e^{-w^T x^i})$$

The long derivation

Learning parameters for classification

$$LL(y|x; w) = \sum_i y^i w^T x^i - w^T x^i + \log(g(w^T x^i))$$

$$w^T x = \sum_j w_j x_j$$

$$g'(h) = \frac{e^{-h}}{(1 + e^{-h})^2}$$

$$\frac{\partial}{\partial w_j} LL(y|x; w) = \sum_i y^i x_j^i - x_j^i + \frac{x_j^i e^{-w^T x^i}}{1 + e^{-w^T x^i}}$$

$$\frac{\partial}{\partial w_j} LL(y|x; w) = \sum_i x_j^i (y^i - (1 - (1 - g(w^T x^i))))$$

$$\frac{\partial}{\partial w_j} LL(y|x; w) = \sum_i x_j^i (y^i - g(w^T x^i))$$

The long derivation

Iterative gradient ascent

y^i – true data label
 $g(w^T x^i)$ – computed data label

- Begin with initial guessed weights w
- For each data point (y^i, x^i) , update each weight w_j

$$w_j \leftarrow w_j + \varepsilon x_j^i (y^i - g(w^T x^i))$$

- Choose ε so change is not too big or too small – “step size”

Intuition

- $x_j^i (y^i - g(w^T x^i))$
 - If $y^i=1$ and $g(w^T x^i)=0$, and $x_j^i > 0$, make w_j larger and push $w^T x^i$ to be larger
 - If $y^i=0$ and $g(w^T x^i)=1$, and $x_j^i > 0$, make w_j smaller and push $w^T x^i$ to be smaller

Iterative gradient ascent – big picture

Initialize w with random values

- Loop across all training data x^i for each feature x_j^i
- Repeat this loop many times (100x, or 1000x, etc.)

Repeat process 10-100 times, each time with new random initial w Find w with high training set classification performance

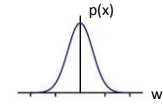
- Goal of repeat initialization: avoid settling on local maximum

MAP for discriminative classifier

- MLE: $P(y=1|x;w) \sim g(w^T x)$
- MAP: $P(y=1, w|x) \propto P(y=1|x;w) P(w) \sim g(w^T x) ???$
(different from Bayesian posterior)
- $P(w)$ priors
 - L2 regularization – minimize all weights
 - L1 regularization – minimize number of non-zero weights

18

MAP – L2 regularization



- $P(y=1, w|x) \propto P(y=1|x;w) P(w)$:

$$L(y, w|x) = \prod_i (1 - g(x^i; w))^{(1-y^i)} g(x^i; w)^{y^i} \prod_j e^{-\frac{w_j^2}{2\lambda}}$$

$$LL(y, w|x) = \sum_i y^i w^T x^i - w^T x^i + \log(g(w^T x^i)) - \sum_j \frac{w_j^2}{2\lambda}$$

$$\frac{\partial}{\partial w_j} LL(y, w|x) = \sum_i x_j^i (y^i - g(w^T x^i)) - \frac{w_j}{\lambda}$$

19

Thinking about your data:
numeric and non-numeric features

Data to be classified can have multiple features $x^i = \begin{bmatrix} x_1^i \\ \vdots \\ x_n^i \end{bmatrix}$

Each feature could be:

- Numeric: Loudness of music, from 0 to 30 decibels
- Non-numeric: Action, including Laugh, Cry, Jump, Dance

Classifier choice

Logistic regression only makes sense for numeric data

Gaussian Bayes only makes sense for numeric data

Multinomial Bayes makes sense for non-numeric data

Non-numeric features -> numeric

You may map non-numeric features to continuous space

Example:

- Mood={Depressed, Disappointed, Neutral, Happy, Excited}
- Switch to: HappinessLevel = {-2, -1, 0, 1, 2}