Linear Classifiers

CS114B Lab 2

Kenneth Lai

January 27, 2023

Vectors and Vector Spaces

- Wikipedia: A vector space is a set of objects called vectors, which may be added together and multiplied by numbers called scalars

Vectors and Vector Spaces

- Wikipedia: A vector space is a set of objects called vectors, which may be added together and multiplied by numbers called scalars
- Lots of examples of vector spaces; we will focus on a particular type of real vector space/coordinate space, where
  - Vectors are n-tuples of real numbers, for some natural number n
  - Scalars are real numbers

Vectors and Vector Spaces

- Wikipedia: A vector space is a set of objects called vectors, which may be added together and multiplied by numbers called scalars
- Lots of examples of vector spaces; we will focus on a particular type of real vector space/coordinate space, where
  - Vectors are elements of $\mathbb{R}^n$
  - Scalars are elements of $\mathbb{R}$

Vectors and Vector Spaces

- Wikipedia: A vector space is a set of objects called vectors, which may be added together and multiplied by numbers called scalars
- Lots of examples of vector spaces; we will focus on a particular type of real vector space/coordinate space, where
  - Vectors are elements of $\mathbb{R}^n$
  - Scalars are elements of $\mathbb{R}$
- The dimension of such a vector space (not to be confused with a dimension, i.e., axis, of a Numpy array) is n

Vectors in NLP

- Idea: objects of interest (e.g., documents, words, etc.) can be represented as vectors (embeddings) in some vector space
- Coordinates of the vector correspond to features of the object

Vectors in NLP

- Idea: objects of interest (e.g., documents, words, etc.) can be represented as vectors (embeddings) in some vector space
- Coordinates of the vector correspond to features of the object
  - Sometimes, these features are human-interpretable
    - Naïve Bayes features: word counts in a document

Vectors in NLP

- Idea: objects of interest (e.g., documents, words, etc.) can be represented as vectors (embeddings) in some vector space
- Coordinates of the vector correspond to features of the object
  - Sometimes, these features are human-interpretable
    - Naïve Bayes features: word counts in a document
  - Sometimes, they are not
    - Many word vector "features"

Linear Classifiers

- Suppose we have a classification problem (e.g., text classification, sequence labeling, etc.)

Linear Classifiers

- Suppose we have a classification problem (e.g., text classification, sequence labeling, etc.)
- Linear classifiers make their classification decisions based on a linear combination of features
  - Logistic regression
  - Perceptron
  - Naïve Bayes (in a way)
  - ...

Linear Classifiers

- Let $\mathbf{x} = \begin{bmatrix} x_1 & \dots & x_n \end{bmatrix}$ be a feature vector, $\theta = \begin{bmatrix} \theta_1 & \dots & \theta_n \end{bmatrix}$ be a vector of parameters, g be a classification function, $\hat{y}$ be the classification decision, and $\cdot$ denote the dot product

$$\hat{y} = g\left(\sum_{j=1}^{n} \theta_j x_j\right) = g(\theta \cdot \mathbf{x})$$

Linear Classifiers

- Sometimes (especially in logistic regression), within the set of parameters, we distinguish between weights w_j and a bias term b
  - This is equivalent to having a "dummy feature" with value 1

$$\theta = \begin{bmatrix} w_1 & \dots & w_n & b \end{bmatrix} = \left[\begin{array}{c|c} \mathbf{w} & b \end{array} \right]$$

$$\mathbf{x}' = \begin{bmatrix} x_1 & \dots & x_n & 1 \end{bmatrix} = \left[\begin{array}{c|c} \mathbf{x} & 1 \end{array} \right]$$

$$\hat{y} = g\left(\sum_{j=1}^{n} w_j x_j + b \right) = g(\mathbf{w} \cdot \mathbf{x} + b) = g(\theta \cdot \mathbf{x}')$$

Linear Classifiers

- What is g?
  - A dot product of two vectors produces a scalar, but in general, we don't just want an arbitrary real number
    - Sometimes, we want a probability (logistic regression)
    - Sometimes, we just want the decision itself (perceptron)

Linear Classifiers

- Let $z = \theta \cdot \mathbf{x}$ (or $\mathbf{w} \cdot \mathbf{x} + b$)
- Logistic regression: logistic (sigmoid) function
  - $\sigma(z) = \dfrac{1}{1 + e^{-z}}$

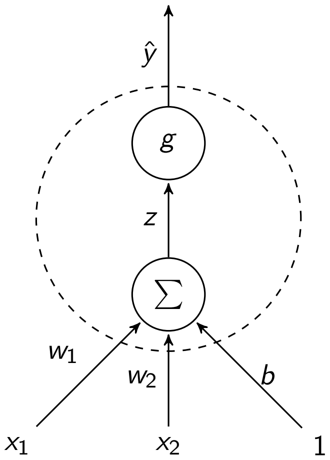
Linear Classifiers

- Let $z = \theta \cdot \mathbf{x}$ (or $\mathbf{w} \cdot \mathbf{x} + b$)
- Logistic regression: logistic (sigmoid) function
  - $\sigma(z) = \dfrac{1}{1 + e^{-z}}$
- Perceptron: (Heaviside) step function
  - $H(z) = \begin{cases} 1, & \text{if } z > 0 \\ 0, & \text{if } z < 0 \end{cases}$
    - Sometimes you may see values 1 and -1, instead of 1 and 0

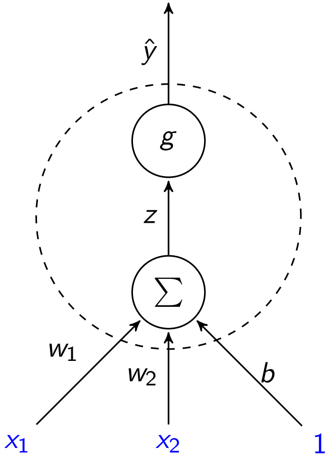
Linear Classifiers

- Let $z = \theta \cdot \mathbf{x}$ (or $\mathbf{w} \cdot \mathbf{x} + b$)
- Logistic regression: logistic (sigmoid) function
  - $\sigma(z) = \dfrac{1}{1 + e^{-z}}$
- Perceptron: (Heaviside) step function
  - $H(z) = \begin{cases} 1, & \text{if } z > 0 \\ 0, & \text{if } z < 0 \end{cases}$
    - Sometimes you may see values 1 and -1, instead of 1 and 0
  - What if $z = 0$?
    - Set by convention (1, 0, or 1/2)

Graphical Representation of a Linear Classifier (1)

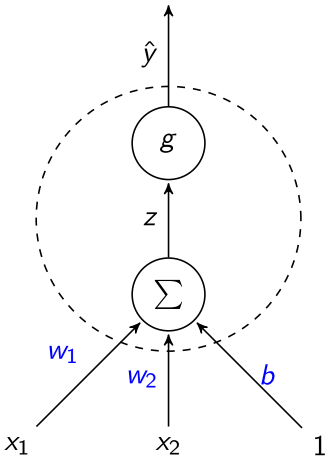


Graphical Representation of a Linear Classifier (1)



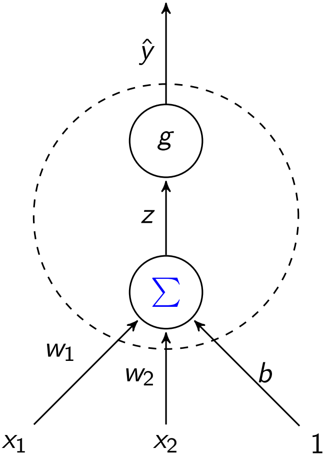
- Input (including dummy feature 1)

Graphical Representation of a Linear Classifier (1)



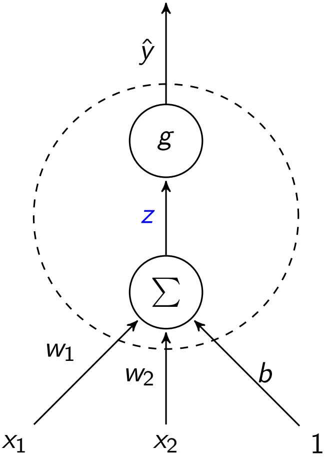
- Parameters (weights and bias term)

Graphical Representation of a Linear Classifier (1)



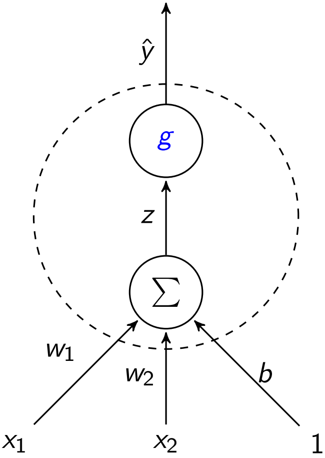
- Sum function $\sum$

Graphical Representation of a Linear Classifier (1)



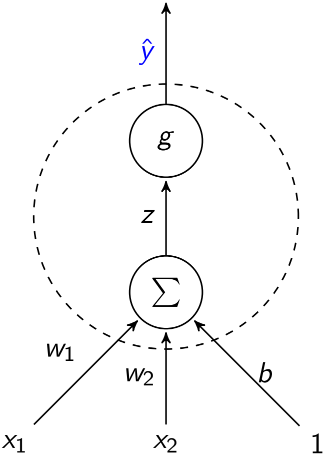
- "Score"
  - Sometimes (if g is the logistic function) called a logit

Graphical Representation of a Linear Classifier (1)



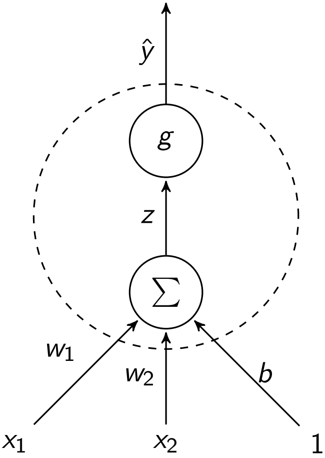
- Classification function g
  - Logistic, step, etc.
  - Later called an activation function

Graphical Representation of a Linear Classifier (1)

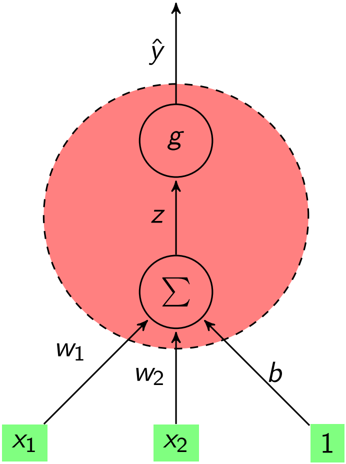


- Output

Graphical Representation of a Linear Classifier (1)



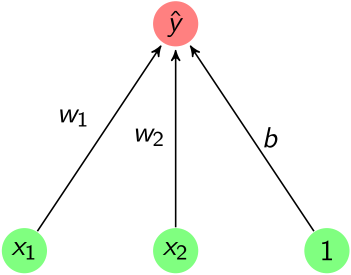
Graphical Representation of a Linear Classifier (1)



- Output
- Input

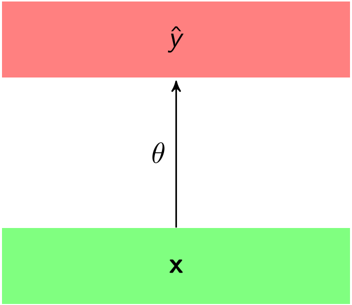
Graphical Representation of a Linear Classifier (1)

- Output
- Input

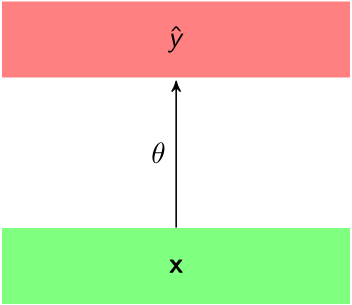


Graphical Representation of a Linear Classifier (2)

- Output
- Input



Graphical Representation of a Linear Classifier (2)



- Output
- Input
- $\hat{y} = g(\theta \cdot \mathbf{x})$
  - We will assume that the dummy feature 1 is part of $\mathbf{x}$

Multi-Class Classification

- Two-class (binary) classification
  - Compute "score" $z = \theta \cdot \mathbf{x}$ (or $\mathbf{w} \cdot \mathbf{x} + b$)
  - Compute decision $\hat{y}$ as a function of z
    - If $\hat{y}$ is interpreted as the probability of (or indicator for) one class, $1 - \hat{y}$ is the probability of (indicator for) the other class

Multi-Class Classification

- Two-class (binary) classification
  - Compute "score" $z = \theta \cdot \mathbf{x}$ (or $\mathbf{w} \cdot \mathbf{x} + b$)
  - Compute decision $\hat{y}$ as a function of z
    - If $\hat{y}$ is interpreted as the probability of (or indicator for) one class, $1 - \hat{y}$ is the probability of (indicator for) the other class
- Multi-class (multinomial) classification
  - Compute a vector of scores $\mathbf{z} = \boldsymbol{\Theta}\mathbf{x}$ (or $\mathbf{W}\mathbf{x} + \mathbf{b}$)
  - Compute decision $\hat{y}$ as a function of $\mathbf{z}$

Matrices and Linear Maps

- A matrix is a rectangular array of scalars
- Two uses of matrices
  1. Matrices represent linear maps
     - Transformations between vector spaces
       - Given a feature vector $\mathbf{x}$, we want a score vector $\mathbf{z}$
       - $\mathbf{z} = \mathbf{\Theta x}$ (or $\mathbf{Wx} + \mathbf{b}$)

Matrices and Linear Maps

- A matrix is a rectangular array of scalars
- Two uses of matrices
  1. Matrices represent linear maps
     - Transformations between vector spaces
       - Given a feature vector $\mathbf{x}$, we want a score vector $\mathbf{z}$
       - $\mathbf{z} = \boldsymbol{\Theta}\mathbf{x}$ (or $\mathbf{W}\mathbf{x} + \mathbf{b}$)
  2. Matrices represent data
     - Stacks of feature vectors
     - Matrix-vector products become matrix-matrix products

Matrices and Linear Maps

- Warning! A note on notation:
  - Math convention: A p-by-n matrix defines a linear map from $\mathbb{R}^n$ to $\mathbb{R}^p$
    - Matrices have shapes (output dimension, input dimension)
    - Let $\mathbf{\Theta} \in \mathbb{R}^{p \times n}$, $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{z} \in \mathbb{R}^p$
    - $\mathbf{z} = \mathbf{\Theta x}$ (or $\mathbf{Wx} + \mathbf{b}$)

Matrices and Linear Maps

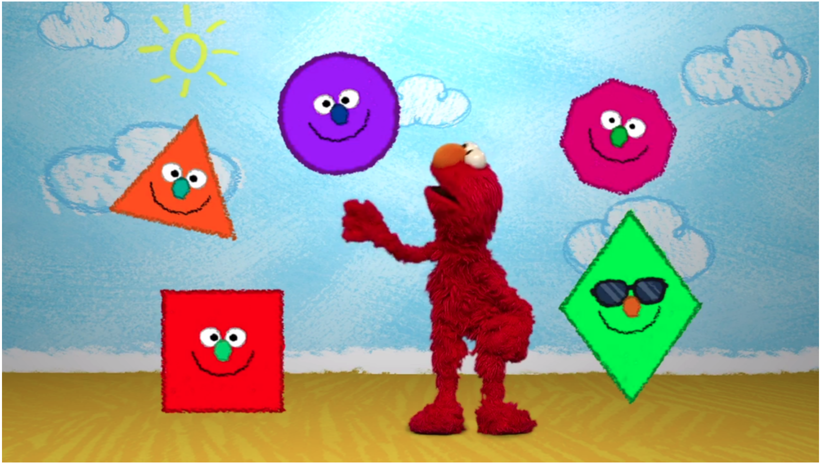
- Warning! A note on notation:
  - Math convention: A p-by-n matrix defines a linear map from $\mathbb{R}^n$ to $\mathbb{R}^p$
    - Matrices have shapes (output dimension, input dimension)
    - Let $\mathbf{\Theta} \in \mathbb{R}^{p \times n}$, $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{z} \in \mathbb{R}^p$
    - $\mathbf{z} = \mathbf{\Theta x}$ (or $\mathbf{Wx} + \mathbf{b}$)
  - Computer science convention (mostly): An n-by-p matrix defines a linear map from $\mathbb{R}^n$ to $\mathbb{R}^p$ (technically $\mathbb{R}^{1 \times n}$ to $\mathbb{R}^{1 \times p}$)
    - Matrices have shapes (input dimension, output dimension)
    - Let $\mathbf{\Theta} \in \mathbb{R}^{n \times p}$, $\mathbf{x} \in \mathbb{R}^{1 \times n}$, and $\mathbf{z} \in \mathbb{R}^{1 \times p}$
    - $\mathbf{z} = \mathbf{x\Theta}$ (or $\mathbf{xW} + \mathbf{b}$)

Matrices and Linear Maps

- Warning! A note on notation:
  - Math convention: A p-by-n matrix defines a linear map from $\mathbb{R}^n$ to $\mathbb{R}^p$
    - Matrices have shapes (output dimension, input dimension)
    - Let $\boldsymbol{\Theta} \in \mathbb{R}^{p \times n}$, $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{z} \in \mathbb{R}^p$
    - $\mathbf{z} = \boldsymbol{\Theta}\mathbf{x}$ (or $\mathbf{W}\mathbf{x} + \mathbf{b}$)
  - Computer science convention (mostly): An n-by-p matrix defines a linear map from $\mathbb{R}^n$ to $\mathbb{R}^p$ (technically $\mathbb{R}^{1 \times n}$ to $\mathbb{R}^{1 \times p}$)
    - Matrices have shapes (input dimension, output dimension)
    - Let $\boldsymbol{\Theta} \in \mathbb{R}^{n \times p}$, $\mathbf{x} \in \mathbb{R}^{1 \times n}$, and $\mathbf{z} \in \mathbb{R}^{1 \times p}$
    - $\mathbf{z} = \mathbf{x}\boldsymbol{\Theta}$ (or $\mathbf{x}\mathbf{W} + \mathbf{b}$)
    - More intuitive (input $\rightarrow$ output)
    - Aligns with the convention in (mini)batch training that the first dimension is the batch size ("feature vectors are stacked row-wise")

General Advice

- Know your shapes!



Source

Multi-Class Classification

- What is the classification function g?
- Let $\mathbf{z} = \begin{bmatrix} z_1 & \dots & z_p \end{bmatrix}$ be a vector of scores for each class

Multi-Class Classification

- What is the classification function g?
- Let $\mathbf{z} = \begin{bmatrix} z_1 & \dots & z_p \end{bmatrix}$ be a vector of scores for each class
- Logistic regression: softmax function
  - $\text{softmax}(z_c) = \frac{e^{z_c}}{\sum_{k=1}^{p} e^{z_k}} = P(y = c|\mathbf{x}) = \hat{y}_c$
  - $\text{softmax}(\mathbf{z}) = \hat{\mathbf{y}}$ is a vector of probabilities for each class

Multi-Class Classification

- What is the classification function g?
- Let $\mathbf{z} = \begin{bmatrix} z_1 & \dots & z_p \end{bmatrix}$ be a vector of scores for each class
- Logistic regression: softmax function
  - $\text{softmax}(z_c) = \dfrac{e^{z_c}}{\sum_{k=1}^{p} e^{z_k}} = P(y = c|\mathbf{x}) = \hat{y}_c$
  - $\text{softmax}(\mathbf{z}) = \hat{\mathbf{y}}$ is a vector of probabilities for each class
- Perceptron: argmax function
  - $\underset{k=1}{\overset{p}{\text{argmax}}}(z_k) = \hat{y}$

Multi-Class Classification

- What is the classification function g?
- Let $\mathbf{z} = \begin{bmatrix} z_1 & \dots & z_p \end{bmatrix}$ be a vector of scores for each class
- Logistic regression: softmax function
  - $\text{softmax}(z_c) = \frac{e^{z_c}}{\sum_{k=1}^{p} e^{z_k}} = P(y = c|\mathbf{x}) = \hat{y}_c$
  - $\text{softmax}(\mathbf{z}) = \hat{\mathbf{y}}$ is a vector of probabilities for each class
- Perceptron: argmax function
  - $\operatorname*{argmax}_{k=1}^{p}(z_k) = \hat{y}$
  - What if there is a tie?
    - Do whatever `numpy.argmax` does

Naïve Bayes as a Linear Classifier

- Suppose we observe a document d. What is the most likely class $\hat{c}$?

Naïve Bayes as a Linear Classifier

- Suppose we observe a document d. What is the most likely class $\hat{c}$?
- $P(c|d) = \dfrac{P(d|c)P(c)}{P(d)}$
  - Bayes' Rule
- $\hat{c} = \operatorname*{argmax}_{c \in C} P(d|c)P(c)$
  - $P(d)$ is the same for each class
- $\hat{c} = \operatorname*{argmax}_{c \in C} P(c) \prod_{i \in \text{positions}} P(w_i|c)$
  - Bag of words assumption, Naïve Bayes assumption
- $\hat{c} = \operatorname*{argmax}_{c \in C} \log P(c) + \sum_{i \in \text{positions}} \log P(w_i|c)$
  - If $xy = z$, then $\log(x) + \log(y) = \log(z)$

Naïve Bayes as a Linear Classifier

- $\hat{c} = \underset{c \in C}{\operatorname{argmax}} \sum_{w \in |V|} \Big[(\text{count}(w, d))(\log P(w|c)) \Big] + \log P(c)$
  - $\sum_{i \in \text{positions}} \log P(w_i|c) = \sum_{w \in |V|} \Big[(\text{count}(w, d))(\log P(w|c)) \Big]$

Naïve Bayes as a Linear Classifier

- $\hat{c} = \operatorname*{argmax}_{c \in C} \sum_{w \in |V|} \left[(\text{count}(w, d))(\log P(w|c)) \right] + \log P(c)$
  - $\sum_{i \in \text{positions}} \log P(w_i|c) = \sum_{w \in |V|} \left[(\text{count}(w, d))(\log P(w|c)) \right]$
- Let $x_w = \text{count}(w, d)$, $\ell_{cw} = \log P(w|c)$, and $p_c = \log P(c)$

$$
\begin{aligned}
\hat{c} &= \operatorname*{argmax}_{c \in C} \sum_{w \in |V|} x_w \ell_{cw} + p_c \\
&= \operatorname*{argmax}_{c \in C} (\mathbf{x} \cdot \ell_{\mathbf{c}} + p_c)
\end{aligned}
$$

Naïve Bayes as a Linear Classifier

- Let $\mathbf{x}$ be a feature vector, $\mathcal{L} = \texttt{self.likelihood}$, and $\mathbf{p} = \texttt{self.prior}$

$$\mathbf{z} = \mathbf{x}\mathcal{L} + \mathbf{p}$$

$$\hat{c} = \operatorname*{argmax}_{c \in C}(z_c)$$