Classification
Logistic Regression

Sujay Sanghavi

Classification

Supervised problems where the responses / targets y take finitely many possible values

Binary Classification: y is either 0 or 1; often called the "label"

training samples: $(x^{(i)}, y^{(i)})$, with each $y^{(i)}$ is either 0 or 1.

task: find **classifier** $f(\cdot)$ such that for any x, $f(x) = 0$ or $f(x) = 1$

Often convenient to think of $f(\cdot)$ as partitioning the space of all features x, into $\{x : f(x) = 0\}$ and $\{x : f(x) = 1\}$

The boundary between these two partitions is called the **decision boundary**

Errors: data samples for which classifier is wrong

Methods for Classification

- **Logistic Regression:** Adapting linear regression to binary responses
- **SVM:** Finding linear separators (and kernel extensions to non-linear)
- **Naive Bayes:** Assuming features conditionally independent given target.
- **Decision Trees:** Partition dataset by thresholding features in a tree fashion.
- **Nearest Neighbors:** label y of x is majority of the labels of the k closest training points to x

Linear Classifiers

$f(x)$ is a **linear classifier** if there exists some $\widehat{\beta}$ such that

$$\begin{aligned} f(x) &= 1 \quad \text{if} \quad x^\top \widehat{\beta} > 0 \\ f(x) &= 0 \quad \text{if} \quad x^\top \widehat{\beta} < 0 \end{aligned}$$

i.e. they are classifiers that depend only on a linear function of x.

Learning a linear classifier == finding the $\widehat{\beta}$ from training data

The **decision boundary** is the hyperplane $\{x : x^\top \widehat{\beta} = 0\}$

Linear Classifiers

$f(x)$ is a **linear classifier** if there exists some $\widehat{\beta}$ such that

$$\begin{aligned} f(x) &= 1 \quad \text{if} \quad x^\top \widehat{\beta} > 0 \\ f(x) &= 0 \quad \text{if} \quad x^\top \widehat{\beta} < 0 \end{aligned}$$

i.e. they are classifiers that depend only on a linear function of x.

Learning a linear classifier == finding the $\widehat{\beta}$ from training data

The **decision boundary** is the hyperplane $\{x : x^\top \widehat{\beta} = 0\}$

Note: as before we usually remove intercept by adding an extra coordinate with a "1" value to the feature vector
i.e. x becomes $(1, x_1, \ldots, x_d)$ instead of just $(x_1, \ldots, x_d)$

Towards Logistic Regression

First attempt: just do linear regression:

$$\widehat{\beta} = \arg \min_{\beta} \sum_{i} \left(y^{(i)} - \beta^{\top} x^{(i)} \right)^2$$

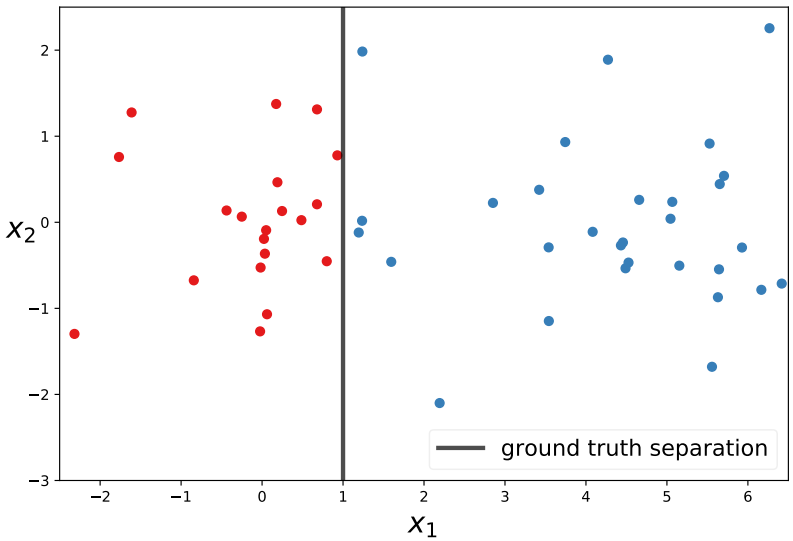
and then threshold : $f(x) = 1$ if $x^{\top} \widehat{\beta} > 0$, else $f(x) = 0$.

To see how this performs, let us look at its performance in an ideal case: linearly separable data

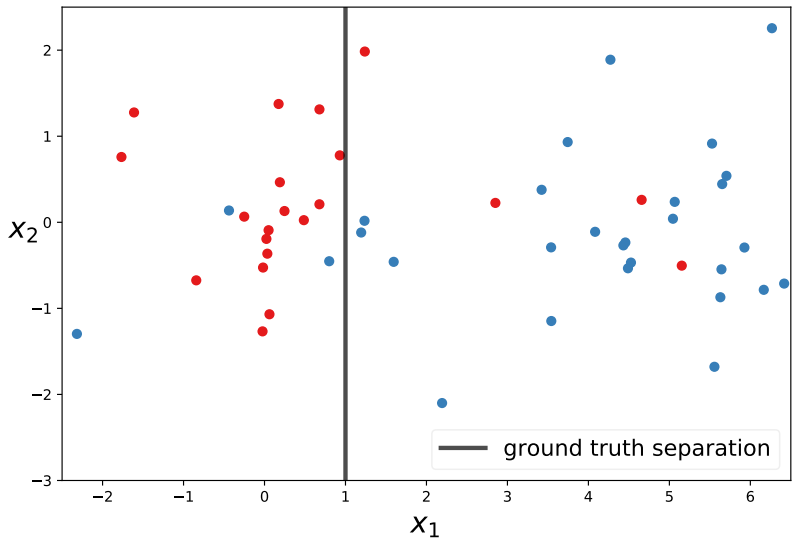
Towards Logistic Regression

Data is called **linearly separable** if there exists a $\beta^\star$ such that

$$y = 1 \text{ if } x^\top \beta^\star > 0 \quad \text{and} \quad y = 0 \text{ if } x^\top \beta^\star < 0$$



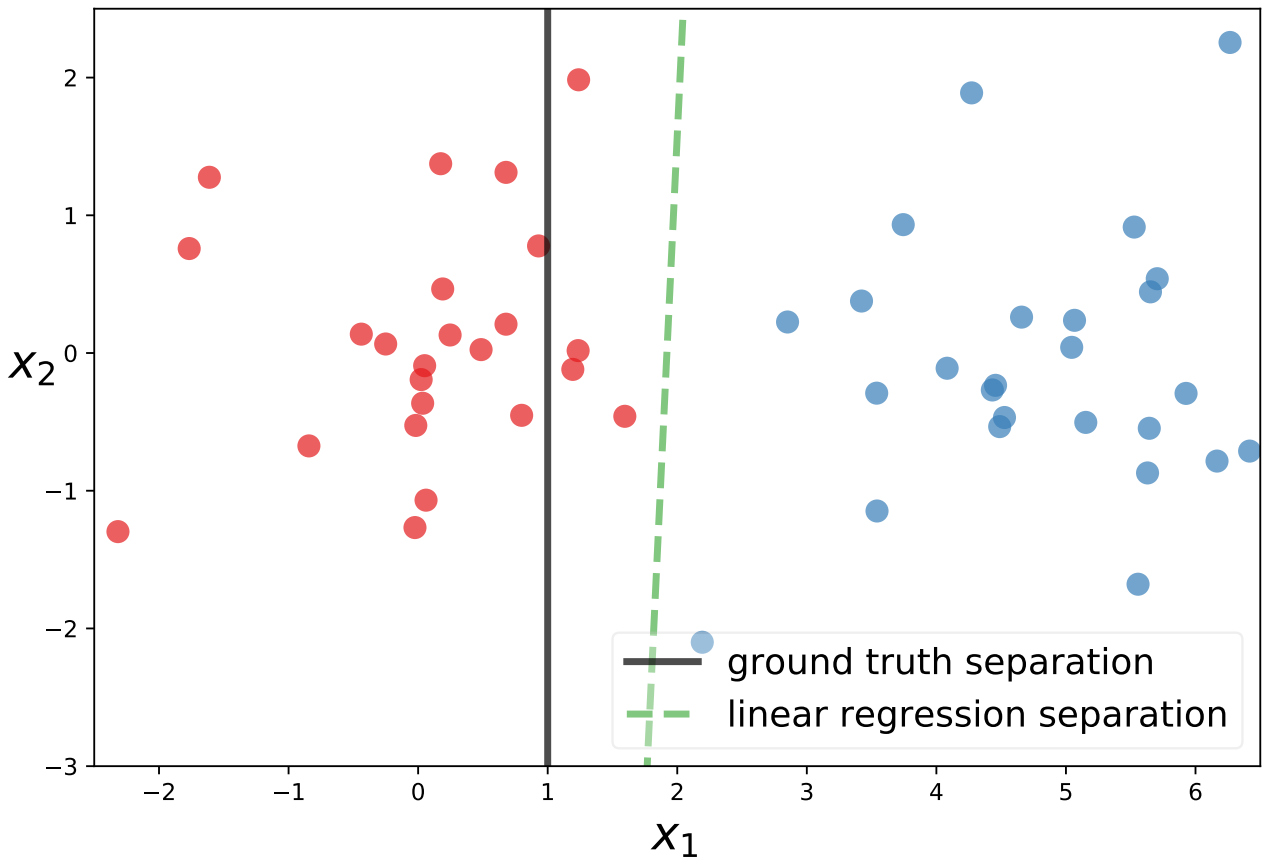
Linearly separable



Not linearly separable

Linearly separable data is the "best case scenario" for linear classifiers

Towards Logistic Regression



Even when data is linearly separable, linear regression does not find this separator

Logistic Regression

Linear regression is a bad fit for classification because its estimate tries to fit a linear function to inherently non-linear target:

- Near boundary: cannot model sharp transition. True y switches from 0 to 1, but $x^\top \widehat{\beta}$ changes slowly
- Far from boundary: will result in large errors, as $x^\top \widehat{\beta}$ will get very large (either very positive or very negative)

Logistic Regression

Linear regression is a bad fit for classification because its estimate tries to fit a linear function to inherently non-linear target:

- Near boundary: cannot model sharp transition. True y switches from 0 to 1, but $x^\top \widehat{\beta}$ changes slowly
- Far from boundary: will result in large errors, as $x^\top \widehat{\beta}$ will get very large (either very positive or very negative)

Idea: use a **non-linear function** to map $x^\top \widehat{\beta}$ to the $y's$

$$y \approx g(x^\top \widehat{\beta}) \quad \text{instead of} \quad y \approx x^\top \widehat{\beta}$$

And then make $f(\cdot)$ by thresholding this function

Logistic Regression

Linear regression is a bad fit for classification because its estimate tries to fit a linear function to inherently non-linear target:

- Near boundary: cannot model sharp transition. True y switches from 0 to 1, but $x^\top \widehat{\beta}$ changes slowly
- Far from boundary: will result in large errors, as $x^\top \widehat{\beta}$ will get very large (either very positive or very negative)

Idea: use a **non-linear function** to map $x^\top \widehat{\beta}$ to the $y's$

$$y \approx g(x^\top \widehat{\beta}) \quad \text{instead of} \quad y \approx x^\top \widehat{\beta}$$

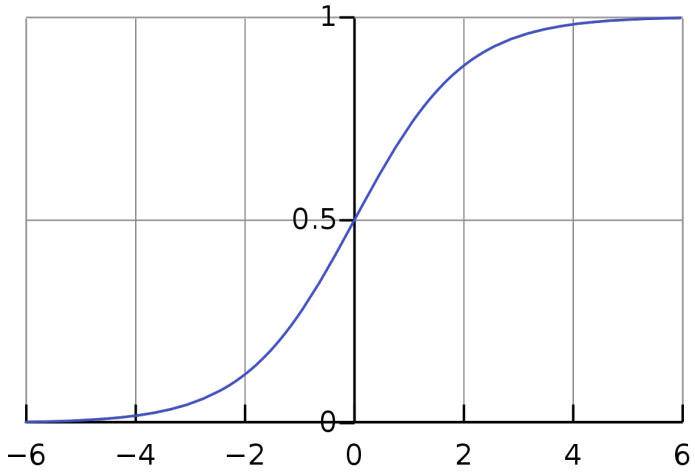
And then make $f(\cdot)$ by thresholding this function

- $g(\cdot)$ is monotone increasing, $g(0) = 1/2$, $g(-\infty) = 0$ and $g(\infty) = 1$
- $g(\cdot)$ is continuous
- $g(\cdot)$ (can) transition from being "close to 0" to being "close to 1" fairly quickly

Logistic Regression

Sigmoid function:

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



We will choose $1/2$ as threshold

Logistic regression:

$$f(x) = 1 \quad \text{if} \quad \frac{\exp(x^\top \widehat{\beta})}{1+\exp(x^\top \widehat{\beta})} > \frac{1}{2}$$

$$f(x) = 0 \quad \text{if} \quad \frac{\exp(x^\top \widehat{\beta})}{1+\exp(x^\top \widehat{\beta})} < \frac{1}{2}$$

Most importantly: change the way we learn $\widehat{\beta}$ from data, by using the fact that we are going to apply the $g(\cdot)$ to it ..

Logistic Regression

To learn the $\widehat{\beta}$ from data, give a probabilistic interpretation to β:

$$\begin{aligned} g(\beta^\top x) &\Leftrightarrow \mathbf{P}_\beta(Y = 1 \,|\, X = x) \\ 1 - g(\beta^\top x) &\Leftrightarrow \mathbf{P}_\beta(Y = 0 \,|\, X = x) \end{aligned}$$

We can then write the **likelihood** of a sample (x, y):

$$L(\beta) = \left[g(\beta^\top x)\right]^y \left[1 - g(\beta^\top x)\right]^{1-y}$$

Logistic Regression

To learn the $\widehat{\beta}$ from data, give a probabilistic interpretation to β:

$$\begin{aligned} g(\beta^\top x) &\Leftrightarrow \mathbf{P}_\beta(Y = 1 \,|\, X = x) \\ 1 - g(\beta^\top x) &\Leftrightarrow \mathbf{P}_\beta(Y = 0 \,|\, X = x) \end{aligned}$$

We can then write the **likelihood** of a sample (x, y):

$$\begin{aligned} L(\beta) &= \left[g(\beta^\top x)\right]^y \left[1 - g(\beta^\top x)\right]^{1-y} \\ &= \left[\frac{1}{1 + \exp(-\beta^\top x)}\right]^y \left[\frac{1}{1 + \exp(\beta^\top x)}\right]^{1-y} \end{aligned}$$

Logistic Regression

To learn the $\widehat{\beta}$ from data, give a probabilistic interpretation to β:

$$\begin{aligned} g(\beta^\top x) &\Leftrightarrow \mathbf{P}_\beta(Y = 1 \,|\, X = x) \\ 1 - g(\beta^\top x) &\Leftrightarrow \mathbf{P}_\beta(Y = 0 \,|\, X = x) \end{aligned}$$

We can then write the **likelihood** of a sample (x, y):

$$\begin{aligned} L(\beta) &= \left[g(\beta^\top x)\right]^y \left[1 - g(\beta^\top x)\right]^{1-y} \\ &= \left[\frac{1}{1 + \exp(-\beta^\top x)}\right]^y \left[\frac{1}{1 + \exp(\beta^\top x)}\right]^{1-y} \\ &= \left[\frac{1}{1 + \exp(-(2y-1)\beta^\top x)}\right] \end{aligned}$$

Logistic Regression

For multiple samples,

$$L(\beta) = \prod_i \left[g(\beta^\top x^{(i)})\right]^{y^{(i)}} \left[1 - g(\beta^\top x^{(i)})\right]^{1-y^{(i)}}$$

Logistic Regression

For multiple samples,

$$L(\beta) = \prod_i \left[g(\beta^\top x^{(i)})\right]^{y^{(i)}} \left[1 - g(\beta^\top x^{(i)})\right]^{1-y^{(i)}}$$

Find $\widehat{\beta}$ from data by maximizing this likelihood

$$\begin{aligned}
\widehat{\beta} &= \arg\max_\beta L(\beta) \\
&= \arg\min_\beta -\log L(\beta) \\
&= \boxed{\arg\min_\beta \sum_i -\log\left(\frac{1}{1+\exp(-(2y^{(i)}-1)\beta^\top x^{(i)})}\right)}
\end{aligned}$$

$-\log L(\beta)$ is a convex function of β. So $\widehat{\beta}$ can be found via **gradient descent**

Summary: Logistic Regression Recipe

Problem: Given binary classification training data of the form $(x^{(i)}, y^{(i)})$, where each feature vector $x^{(i)}$ is a d-dimensional vector and each label $y^{(i)}$ is either 0 or 1, find a d-dimensional vector $\widehat{\beta}$ from which we can make a binary classifier $f(x)$ as follows:

$$\begin{aligned} f(x) &= 1 \quad \text{if} \quad x^\top \widehat{\beta} > 0 \\ f(x) &= 0 \quad \text{if} \quad x^\top \widehat{\beta} < 0 \end{aligned}$$

Solution: Solve the following optimization problem to find $\widehat{\beta}$:

$$\widehat{\beta} = \arg\min_{\beta} \sum_i -\log\left(\frac{1}{1+\exp(-(2y^{(i)}-1)\beta^\top x^{(i)})}\right)$$

Logistic Regression (Extensions)

Just like in linear regression, it is often a good idea to add **regularizers**:

$$\widehat{\beta} \;=\; \arg\min_{\beta} \;\; -\log L(\beta) \;+\; \lambda\|\beta\|_2^2$$

or

$$\widehat{\beta} \;=\; \arg\min_{\beta} \;\; -\log L(\beta) \;+\; \lambda\|\beta\|_1$$

Reasons:

- to guard against overfitting, badly conditioned data, etc.
- to prevent $\beta \to \infty$ when data is separable ...

Probabilistic View of Logistic Regression

Features and targets are random variables: X and Y respectively

Parameter β determines how X is related to Y

$$\mathbf{P}_\beta(Y = 1|X = x) = \frac{1}{1 + \exp(-\beta^\top x)}$$

Probabilistic View of Logistic Regression

Features and targets are random variables: X and Y respectively

Parameter β determines how X is related to Y

$$\mathbf{P}_\beta(Y = 1|X = x) \;=\; \frac{1}{1 + \exp(-\beta^\top x)}$$

Given data (x, y), find $\widehat{\beta}$ by maximizing

$$\widehat{\beta} \;=\; \arg\max_\beta \mathbf{P}_\beta(Y = y \,|\, X = x)$$

Probabilistic View of Logistic Regression

Features and targets are random variables: X and Y respectively

Parameter β determines how X is related to Y

$$\mathbf{P}_\beta(Y = 1|X = x) \;=\; \frac{1}{1 + \exp(-\beta^\top x)}$$

Given data (x, y), find $\widehat{\beta}$ by maximizing

$$\begin{aligned} \widehat{\beta} \;&=\; \arg\max_\beta \mathbf{P}_\beta(Y = y \,|\, X = x) \\ &=\; \arg\max_\beta \mathbf{P}_\beta(Y = y \,|\, X = x)\, \mathbf{P}(X = x) \end{aligned}$$

Probabilistic View of Logistic Regression

Features and targets are random variables: X and Y respectively

Parameter β determines how X is related to Y

$$\mathbf{P}_\beta(Y = 1|X = x) \;=\; \frac{1}{1 + \exp(-\beta^\top x)}$$

Given data (x, y), find $\widehat{\beta}$ by maximizing

$$\begin{aligned}
\widehat{\beta} &= \arg\max_\beta \mathbf{P}_\beta(Y = y \,|\, X = x) \\
&= \arg\max_\beta \mathbf{P}_\beta(Y = y \,|\, X = x)\, \mathbf{P}(X = x) \\
&= \arg\max_\beta \mathbf{P}_\beta(Y = y\,,\, X = x)
\end{aligned}$$

So every β corresponds to a **joint distribution** of X and Y
$\widehat{\beta} \;\Leftrightarrow$ the distribution that makes observed data (x, y) most likely

Multinomial Logistic Regression (aka Softmax)

Extension of Logistic Regression to the case of multiple classes

Recall: logistic regression corresponds to:

$$\begin{aligned} \mathbf{P}_\beta(Y = 1 \,|\, X = x) &= \frac{\exp(\beta^\top x)}{1 + \exp(\beta^\top x)} \\ \mathbf{P}_\beta(Y = 0 \,|\, X = x) &= \frac{1}{1 + \exp(\beta^\top x)} \end{aligned}$$

Multinomial Logistic Regression (aka Softmax)

Extension of Logistic Regression to the case of multiple classes

Recall: logistic regression corresponds to:

$$\begin{aligned} \mathbf{P}_\beta(Y = 1 \,|\, X = x) &= \frac{\exp(\beta^\top x)}{1 + \exp(\beta^\top x)} \\ \mathbf{P}_\beta(Y = 0 \,|\, X = x) &= \frac{1}{1 + \exp(\beta^\top x)} \end{aligned}$$

Writing this another way:

$$\mathbf{P}_\beta(Y = 1 \,|\, X = x) \quad \propto \quad \exp(\beta^\top x)$$

$1 + \exp(\beta^\top x)$ is just the normalization factor. $\mathbf{P}_\beta(Y = 0 \,|\, X = x)$ implicit.

Multinomial Logistic Regression (aka Softmax)

Label y can take values $1, \ldots, K$

Now model β consists of K vectors, i.e. $\beta = (\beta_1, \ldots, \beta_K)$, such that:

$$\mathbf{P}_\beta(Y = k \,|\, X = x) \quad \propto \quad \exp(\beta_k^\top x) \quad \text{for } k = 1, \ldots, K$$

Multinomial Logistic Regression (aka Softmax)

Label y can take values $1, \ldots, K$

Now model β consists of K vectors, i.e. $\beta = (\beta_1, \ldots, \beta_K)$, such that:

$$\mathbf{P}_\beta(Y = k \,|\, X = x) \quad \propto \quad \exp(\beta_k^\top x) \quad \text{for } k = 1, \ldots, K$$

So, more precisely:

$$\mathbf{P}_\beta(Y = k \,|\, X = x) \quad = \quad \frac{\exp(\beta_k^\top x)}{\sum_k \exp(\beta_k^\top x)} \quad \text{for } k = 1, \ldots, K$$

Multinomial Logistic Regression (aka Softmax)

Estimation (aka "training") done by

$$\widehat{\beta} \;=\; \arg\min_{\beta}(-\log L(\beta))$$

which is convex, and solved by gradient descent ...

Prediction (aka "inference") again by choosing the most likely label for that $\widehat{\beta}$:

$$\widehat{y} \;=\; \arg\max_{k} \mathbf{P}_{\widehat{\beta}}(Y = k|X = x)$$

Often appears as the last layer of a neural network classifier ...