Linear and Logistic Regression

Machine Learning

Overview

1. Linear regression assumptions.
2. Linear regression algorithm.
3. Logistic regression assumptions.
4. Logistic regression algorithm.

Supervised Regression

Supervised Least-Squares Regression Problem

- $\mathcal{X} = \mathbb{R}^d$.
- $\mathcal{Y} = \mathbb{R}^k$.
- $\ell_h((x, y)) = \|h(x) - y\|_2^2$.

Linear Regression Assumptions

- **I.I.D. assumption:** Data $z_1, \ldots, z_n$ are independent and identically distributed (i.i.d.).
  - Important! Rules out e.g. linear regression for time series, since data is *not* i.i.d.
  - Still a very “natural” assumption to make for tabular data.
- **Non-redundancy assumption:** entries of x are linearly independent random variables.
  - Means that there is *no* $p \in \mathbb{R}^d$ such that $\langle p, x \rangle = 0$ for every realization of x.
  - If we don’t have this property, we can just throw away redundant columns.
- **Linear + noise assumption**: There is $\theta_\star \in \mathbb{R}^{d \times k}$ such that

$$y = \theta_\star^* x + \varepsilon$$

where ε is independent of everything else, and $\mathbb{E}[\varepsilon] = 0$ and $\mathrm{Var}(\varepsilon) = \sigma^2 I_k$.
  - Important! If errors are correlated, we have to use more sophisticated techniques.

General Linear Models

- Actually, the structural assumption

$$y = \theta_\star^* x + \varepsilon$$

  is pretty powerful.
- If we think y is approximately a linear combination of *some features extracted from* x, then we can arrange for those to be used in the model instead.
- More precisely, let $\phi\colon \mathbb{R}^d \to \mathbb{R}^D$ take in an “raw” input x and output a list of features extracted from x.
- Then by replacing x by $\phi(x)$, the assumption becomes that there exists $\theta_\star \in \mathbb{R}^{D\times k}$ such that

$$y = \theta_\star^* \phi(x) + \varepsilon$$

  with the same conditions on ε.
- Basically, this just treats $\phi(x)$ as the new data instead of x.

General Linear Models

Example (Model Intercept)

Suppose $x \in \mathbb{R}^d$ and we know that y is approximately an affine function of x. Then

$$\phi(x) = \begin{bmatrix} 1 \\ x \end{bmatrix} \quad \text{and} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$

so

$$\theta^* \phi(x) = \theta_0 + \theta_1^* x.$$

This is an arbitrary affine function of x.

This transform is very popular and used by default by many machine learning packages.

General Linear Models

Example (Polynomial Feature)

Suppose $x, y \in \mathbb{R}$ and we know that y is approximately a D-degree polynomial of x. Then

$$\phi(x) = \begin{bmatrix} 1 \\ x \\ \vdots \\ x^D \end{bmatrix} \quad \text{and} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_D \end{bmatrix}$$

so

$$\theta^* \phi(x) = \sum_{i=0}^{D} \theta_i x^i.$$

This is an arbitrary D-degree polynomial and learning θ gives us the coefficients.

Least-Squares Linear Regression

Collect data and labels from s_{train} in a matrix:

$$X_{\text{train}} = \begin{bmatrix} x_1^* \\ \vdots \\ x_n^* \end{bmatrix} \in \mathbb{R}^{n \times d} \quad \text{and} \quad Y_{\text{train}} = \begin{bmatrix} y_1^* \\ \vdots \\ y_n^* \end{bmatrix} \in \mathbb{R}^{n \times k}$$

Want to minimize in θ:

$$L_{s_{\text{train}}}(h_\theta) = \frac{1}{n} \sum_{i=1}^{n} \|h_\theta(x_i) - y_i\|_2^2 = \frac{1}{n} \sum_{i=1}^{n} \left\|\theta^* x_i - y_i\right\|_2^2 = \frac{1}{n} \|X_{\text{train}}\theta - Y_{\text{train}}\|_2^2 .$$

By vector calculus, optimal θ is

$$\widehat{\theta} = \left(X_{\text{train}}^* X_{\text{train}}\right)^{-1} X_{\text{train}}^* Y_{\text{train}}.$$

Prediction is

$$h_{\widehat{\theta}}(x) = \widehat{\theta}^* x = Y_{\text{train}}^* X_{\text{train}} \left(X_{\text{train}}^* X_{\text{train}}\right)^{-1} x.$$

Linear Regression Algorithm

Simple supervised regression algorithm.

$$\mathcal{H} = \left\{ x \mapsto \theta^* x \colon \theta \in \mathbb{R}^{d \times k} \right\}$$

Least-Squares Linear Regression Algorithm

procedure TRAIN(s_{train})

$X_{\text{train}}, Y_{\text{train}} \leftarrow s_{\text{train}}$

$\widehat{\theta} \leftarrow \left(X^*_{\text{train}} X_{\text{train}} \right)^{-1} X^*_{\text{train}} Y_{\text{train}}$

procedure INFERENCE(x)

return $\widehat{\theta}^* x$

Supervised Classification

Logistic Classification Problem

- $\mathcal{X} = \mathbb{R}^d$.
- $\mathcal{Y} = [0, 1]^k$.
- $\ell_h((x, y)) = -\sum_{i=1}^{k} y_i \log(h(x)_i)$.

Logistic Regression Assumptions

- **I.I.D. assumption:** Data $z_1, \ldots, z_n$ are independent and identically distributed (i.i.d.).
  - Important! Rules out e.g. logistic regression for time series, since data is *not* i.i.d.
  - Still a very "natural" assumption to make for tabular data.
- **Non-redundancy assumption:** entries of x are linearly independent random variables.
  - Means that there is *no* $p \in \mathbb{R}^d$ such that $\langle p, x \rangle = 0$ for every realization of x.
  - If we don't have this property, we can just throw away redundant columns.
- **Linear + noise assumption**: There is $\theta_\star \in \mathbb{R}^{d \times k}$ such that

$$\begin{bmatrix} \log\left(p_{Y|X}(e_1 \mid x)\right) \\ \vdots \\ \log\left(p_{Y|X}(e_k \mid x)\right) \end{bmatrix} = \theta_\star^* x + \begin{bmatrix} C(x) \\ \vdots \\ C(x) \end{bmatrix} + \varepsilon.$$

where ε is independent of everything else, and $\mathbb{E}[\varepsilon] = 0$ and $\mathrm{Var}(\varepsilon) = \sigma^2 I_k$.
  - The constant $C(x) = -\log\left(\sum_{y \in \mathcal{Y}} \mathrm{e}^{y^* \theta_\star^* x}\right)$ is there to ensure that $\sum_{y \in \mathcal{Y}} p_{Y|X}(y \mid x) = 1$.
  - Important! If errors are correlated, we have to use more sophisticated techniques.

Logistic Regression Algorithm

Prerequisite: **softmax** function $\mathbb{R}^k \to \mathbb{R}^k$:

$$\operatorname{softmax}(x)_i = \frac{e^{x_i}}{\sum_{j=1}^k e^{x_j}}$$

Run linear regression, then use softmax to “squash” all values into $[0, 1]$.

$$\mathcal{H} = \left\{x \mapsto \operatorname{softmax}\left(\theta^* x\right) : \theta \in \mathbb{R}^d\right\}.$$

Logistic Regression Algorithm

procedure TRAIN(s_{train})

$\quad\widehat{\theta} \leftarrow \texttt{Optimize}\left(\theta \mapsto \frac{1}{n}\sum_{i=1}^n \ell(y, \operatorname{softmax}(\theta^* x))\right)$

procedure INFERENCE(x)

$\quad$**return** $\operatorname{softmax}\left(\widehat{\theta}^* x\right)$