Statistical Data Mining and Machine Learning
Hilary Term 2016

Dino Sejdinovic
Department of Statistics
Oxford

Slides and other materials available at:
http://www.stats.ox.ac.uk/~sejdinov/sdmml

Naïve Bayes

- Naïve Bayes: another plug-in classifier with a simple generative model - it assumes all measured variables/features are independent given the label.
- Often used in text document classification, e.g. of scientific articles or emails.
- A basic standard model for text classification consists of considering a pre-specified dictionary of p words and summarizing each document i by a binary vector x_i where

$$x_i^{(j)} = \begin{cases} 1 & \text{if word } j \text{ is present in document} \\ 0 & \text{otherwise.} \end{cases}$$

- Presence of the word j is the j-the feature/dimension.
- To implement a plug-in classifier, we need a model for the conditional probability mass function $g_k(x) = \mathbb{P}(X = x|Y = k)$ for each class $k = 1, ..., K$.

Naïve Bayes

- Naïve Bayes is a plug-in classifier which **ignores feature correlations**[1] and assumes:

$$\begin{aligned} g_k(x_i) = \mathbb{P}(X = x_i | Y = k) &= \prod_{j=1}^{p} \mathbb{P}(X^{(j)} = x_i^{(j)} | Y = k) \\ &= \prod_{j=1}^{p} (\phi_{kj})^{x_i^{(j)}} (1 - \phi_{kj})^{1 - x_i^{(j)}}, \end{aligned}$$

  where we denoted parametrized conditional PMF with $\phi_{kj} = \mathbb{P}(X^{(j)} = 1 | Y = k)$ (probability that j-th word appears in class k document).

- Given dataset, the MLE of the parameters is:

$$\hat{\pi}_k = \frac{n_k}{n}, \qquad \hat{\phi}_{kj} = \frac{\sum_{i: y_i = k} x_i^{(j)}}{n_k}.$$

[1] given the class, it assumes each word appears in a document independently of all others

Naïve Bayes

- MLE:

$$\hat{\pi}_k = \frac{n_k}{n}, \qquad \hat{\phi}_{kj} = \frac{\sum_{i:y_i=k} x_i^{(j)}}{n_k}.$$

- A problem with MLE: if the ℓ-th word did not appear in documents labelled as class k then $\hat{\phi}_{k\ell} = 0$ and

$$\mathbb{P}(Y = k | X = x \text{ with } \ell\text{-th entry equal to 1})$$
$$\propto \hat{\pi}_k \prod_{j=1}^{p} \left(\hat{\phi}_{kj}\right)^{x^{(j)}} \left(1 - \hat{\phi}_{kj}\right)^{1-x^{(j)}} = 0$$

  i.e. we will never attribute a new document containing word ℓ to class k (regardless of other words in it).

- This is an example of **overfitting**.

Generative Learning

- Classifiers we have seen so far are **generative**: we work with a joint distribution $p_{X,Y}(x,y)$ over data vectors and labels.
- A learning algorithm: construct $f:\mathcal{X}\to\mathcal{Y}$ which predicts the label of X.
- Given a loss function L, the risk R of $f(X)$ is

$$R(f)=\mathbb{E}_{p_{X,Y}}[L(Y,f(X))]$$

- For 0/1 loss in classification, Bayes classifier

$$f_{\text{Bayes}}(x)=\underset{k=1,\dots,K}{\operatorname{argmax}}\, p(Y=k|x)=\underset{k=1,\dots,K}{\operatorname{argmax}}\, p_{X,Y}(x,k)$$

  has the minimum risk (Bayes risk), but is unknown since $p_{X,Y}$ is unknown.
- Assume a parameteric model for the joint: $p_{X,Y}(x,y)=p_{X,Y}(x,y|\theta)$
- Fit $\hat{\theta}=\operatorname{argmax}_\theta\sum_{i=1}^n\log p(x_i,y_i|\theta)$ and plug in back to Bayes classifier:

$$\hat{f}(x)=\underset{k=1,\dots,K}{\operatorname{argmax}}\, p(Y=k|x,\theta)=\underset{k=1,\dots,K}{\operatorname{argmax}}\, p_{X,Y}(x,k|\hat{\theta}).$$

Generative vs Discriminative Learning

- **Generative learning**: find parameters which **explain all the data available**.

$$\hat{\theta} = \operatorname*{argmax}_{\theta} \sum_{i=1}^{n} \log p(x_i, y_i | \theta)$$

  Examples: LDA, QDA, naïve Bayes.
  - Makes use of all the data available.
  - Flexible modelling framework, so can incorporate missing features or unlabeled examples.
  - Stronger modelling assumptions, which may not be realistic (Gaussianity, independence of features).
- **Discriminative learning**: find parameters that aid in **prediction**.

$$\hat{\theta} = \operatorname*{argmin}_{\theta} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) \quad \text{or} \quad \hat{\theta} = \operatorname*{argmax}_{\theta} \sum_{i=1}^{n} \log p(y_i | x_i, \theta)$$

  Examples: logistic regression, neural nets, support vector machines.
  - Typically performs better on a given task.
  - Weaker modelling assumptions: essentially no model on X, only on $Y|X$.
  - Can overfit more easily.

Logistic regression

- A **discriminative classifier**. Consider binary classification with $\mathcal{Y} = \{-1, +1\}$. Logistic regression uses a parametric model on the conditional $Y|X$, not the joint distribution of (X, Y):

$$p(Y = y|X = x; a, b) = \frac{1}{1 + \exp(-y(a + b^\top x))}.$$

- a, b fitted by minimizing the empirical risk with respect to **log loss**.

Hard vs Soft classification rules

- Consider using LDA for binary classification with $\mathcal{Y} = \{-1, +1\}$. Predictions are based on linear decision boundary:

$$\begin{aligned}\hat{y}_{\text{LDA}}(x) &= \text{sign}\left\{\log \hat{\pi}_{+1} g_{+1}(x|\hat{\mu}_{+1}, \hat{\Sigma}) - \log \hat{\pi}_{-1} g_{-1}(x|\hat{\mu}_{-1}, \hat{\Sigma})\right\} \\ &= \text{sign}\left\{a + b^\top x\right\}\end{aligned}$$

  for a and b depending on fitted parameters $\hat{\theta} = (\hat{\pi}_{-1}, \hat{\pi}_{+1}, \hat{\mu}_{-1}, \hat{\mu}_{+1}, \Sigma)$.
- Quantity $a + b^\top x$ can be viewed as a soft classification rule. Indeed, it is modelling the difference between the log-discriminant functions, or equivalently, the **log-odds ratio**:

$$a + b^\top x = \log \frac{p(Y = +1|X = x; \hat{\theta})}{p(Y = -1|X = x; \hat{\theta})}.$$

- $f(x) = a + b^\top x$ corresponds to the “confidence of predictions” and loss can be measured as a function of this confidence:
  - exponential loss: $L(y, f(x)) = e^{-yf(x)}$,
  - log-loss: $L(y, f(x)) = \log(1 + e^{-yf(x)})$,
  - hinge loss: $L(y, f(x)) = \max\{1 - yf(x), 0\}$.

Linearity of log-odds and logistic function

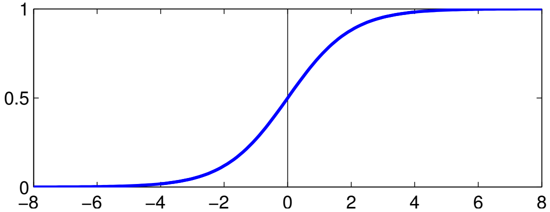
- We can treat a and b as parameters in their own right in the model of the conditional $Y|X$.

$$\log \frac{p(Y=+1|X=x;a,b)}{p(Y=-1|X=x;a,b)} = a + b^\top x.$$

- Solve explicitly for conditional class probabilities:

$$p(Y=+1|X=x;a,b) = \frac{1}{1+\exp(-(a+b^\top x))} =: s(a+b^\top x)$$
$$p(Y=-1|X=x;a,b) = \frac{1}{1+\exp(+(a+b^\top x))} = s(-a-b^\top x)$$

where $s(z) = 1/(1+\exp(-z))$ is the **logistic function**.



Fitting the parameters of the hyperplane

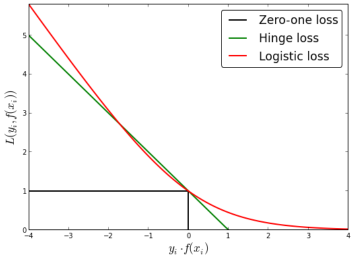
- Consider maximizing the **conditional log likelihood**:

$$\ell(a,b) = \sum_{i=1}^{n} \log p(Y = y_i | X = x_i) = \sum_{i=1}^{n} \log s(y_i(a + b^\top x_i)).$$

- Equivalent to minimizing the empirical risk associated with the **log loss**:

$$\hat{R}_{\mathrm{log}}(f_{a,b}) = \frac{1}{n}\sum_{i=1}^{n} -\log s(y_i(a + b^\top x_i)) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i(a + b^\top x_i)))$$

over all linear soft classification rules $f_{a,b}(x) = a + b^\top x$.



Logistic Regression

- Not possible to find optimal a, b analytically.
- For simplicity, absorb a as an entry in b by appending '1' into x vector.
- Objective function:

$$\hat{R}_{\text{log}} = \frac{1}{n}\sum_{i=1}^{n} -\log s(y_i x_i^\top b)$$

Logistic Function

$$\begin{aligned} s(-z) &= 1 - s(z) \\ \nabla_z s(z) &= s(z)s(-z) \\ \nabla_z \log s(z) &= s(-z) \\ \nabla_z^2 \log s(z) &= -s(z)s(-z) \end{aligned}$$

- Differentiate wrt b:

$$\begin{aligned} \nabla_b \hat{R}_{\text{log}} =& \frac{1}{n}\sum_{i=1}^{n} -s(-y_i x_i^\top b) y_i x_i \\ \nabla_b^2 \hat{R}_{\text{log}} =& \frac{1}{n}\sum_{i=1}^{n} s(y_i x_i^\top b) s(-y_i x_i^\top b) x_i x_i^\top \succeq 0. \end{aligned}$$

Logistic Regression

- Second derivative is positive-definite: objective function is **convex** and there is **a single unique global minimum**.
- Many different algorithms can find optimal b, e.g.:
  - Gradient descent:
$$b^{\text{new}} = b + \epsilon \frac{1}{n} \sum_{i=1}^{n} s(-y_i x_i^\top b) y_i x_i$$
  - Stochastic gradient descent:
$$b^{\text{new}} = b + \epsilon_t \frac{1}{|I(t)|} \sum_{i \in I(t)} s(-y_i x_i^\top b) y_i x_i$$
    where $I(t)$ is a subset of the data at iteration t, and $\epsilon_t \to 0$ slowly ($\sum_t \epsilon_t = \infty, \sum_t \epsilon_t^2 < \infty$).
  - Newton-Raphson:
$$b^{\text{new}} = b - (\nabla_b^2 \hat{R}_{\text{log}})^{-1} \nabla_b \hat{R}_{\text{log}}$$
    This is also called **iterative reweighted least squares**.
  - Conjugate gradient, LBFGS and other methods from numerical analysis.

Logistic Regression vs. LDA

- Both have linear decision boundaries and model log-posterior odds as

$$\log \frac{p(Y=+1|X=x)}{p(Y=-1|X=x)} = a + b^\top x$$

- LDA models the marginal density of x as a Gaussian mixture with shared covariance

  $$g(x) = \pi_{-1}\mathcal{N}(x;\mu_{-1},\Sigma) + \pi_{+1}\mathcal{N}(x;\mu_{+1},\Sigma)$$

  and fits the parameters $\theta = (\mu_{-1},\mu_{+1},\pi_{-1},\pi_{+1},\Sigma)$ by maximizing joint likelihood $\sum_{i=1}^{n} p(x_i,y_i|\theta)$. a and b are then determined from θ.
- Logistic regression leaves the marginal density $g(x)$ as an **arbitrary density function**, and fits the parameters a,b by maximizing the conditional likelihood $\sum_{i=1}^{n} p(y_i|x_i;a,b)$.

Linearly separable data

Assume that the data is linearly separable, i.e. there is a scalar α and a vector β such that $y_i(\alpha + \beta^\top x_i) > 0$, $i = 1, \ldots, n$. Let $c > 0$. The empirical risk for $a = c\alpha$, $b = c\beta$ is

$$\hat{R}_{\mathsf{log}}(f_{a,b}) = \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-cy_i(\alpha + \beta^\top x_i)))$$

which can be made arbitrarily close to zero as $c \to \infty$, i.e. soft classification rule becomes $\pm\infty$ (overconfidence).

Multi-class logistic regression

The **multi-class/multinomial** logistic regression uses the **softmax** function to model the conditional class probabilities $p\left(Y=k|X=x;\theta\right)$, for K classes $k=1,\ldots,K$, i.e.,

$$p\left(Y=k|X=x;\theta\right)=\frac{\exp\left(w_k^\top x+b_k\right)}{\sum_{\ell=1}^K\exp\left(w_\ell^\top x+b_\ell\right)}.$$

Parameters are $\theta=(b,W)$ where $W=(w_{kj})$ is a $K\times p$ matrix of weights and $b\in\mathbb{R}^K$ is a vector of bias terms.

Logistic Regression: Summary

- Makes less modelling assumptions than generative classifiers: often resulting in better prediction accuracy.
- Diverging optimal parameters for linearly separable data: need to **regularise** / pull them towards zero.
- A simple example of a generalised linear model (GLM), for which there is a well established statistical theory:
  - Assessment of fit via deviance and plots,
  - Well founded approaches to removing insignificant features (drop-in deviance test, Wald test).

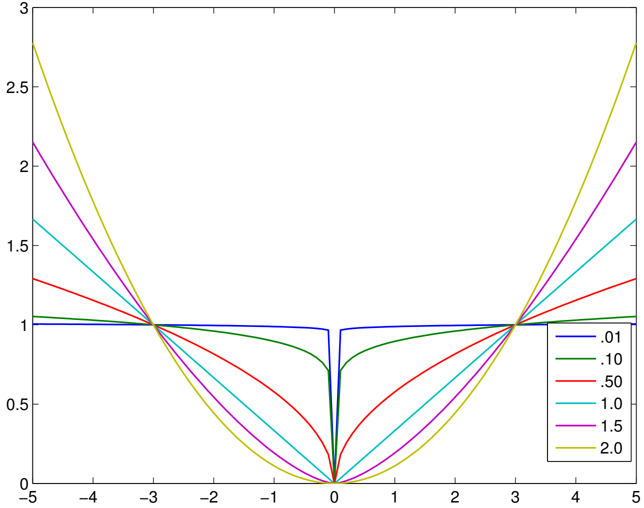
Regularization

- Flexible models for high-dimensional problems require many parameters.
- With many parameters, learners can easily overfit.
- **Regularization**: Limit flexibility of model to prevent overfitting.
- Add term **penalizing large values of parameters** θ.

$$\min_{\theta} \hat{R}(f_\theta) + \lambda\|\theta\|_\rho^\rho = \min_{\theta} \frac{1}{n}\sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda\|\theta\|_\rho^\rho$$

  where $\rho \geq 1$, and $\|\theta\|_\rho = (\sum_{j=1}^{p} |\theta_j|^\rho)^{1/\rho}$ is the L_ρ norm of θ (also of interest when $\rho \in [0, 1)$, but is no longer a norm).
- Also known as **shrinkage** methods—parameters are shrunk towards 0.
- λ is a **tuning parameter** (or **hyperparameter**) and controls the amount of regularization, and resulting complexity of the model.

Regularization



L_ρ regularization profile for different values of ρ.

Types of Regularization

- **Ridge regression / Tikhonov regularization**: $\rho = 2$ (Euclidean norm)
- **LASSO**: $\rho = 1$ (Manhattan norm)
- **Sparsity-inducing** regularization: $\rho \leq 1$ (nonconvex for $\rho < 1$)
- **Elastic net** regularization: mixed L_1/L_2 penalty:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f_\theta(x_i)) + \lambda \left[(1 - \alpha) \|\theta\|_2^2 + \alpha \|\theta\|_1 \right]$$

L_1 promotes sparsity

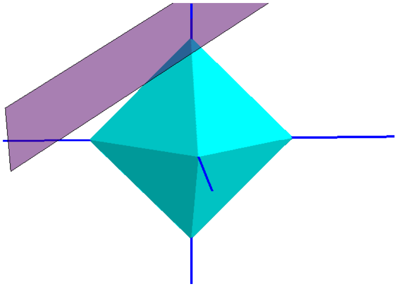
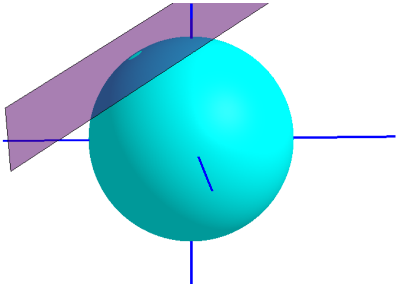



Figure : The intersection between the L_1 (left) and the L_2 (right) ball with a hyperplane.

L_1 regularization often leads to optimal solutions with many zeros, i.e., the regression function depends only on the (small) number of features with non-zero parameters.

figure from M. Elad, Sparse and Redundant Representations, 2010.