Fundamentals of Machine Learning:

Linear Models for Classification: Probabilistic Models

Prof. Andrew D. Bagdanov (`andrew.bagdanov AT unifi.it`)

UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Outline

Introduction

Probabilistic Generative Models

Probabilistic Discriminative Models

Bayesian Logistic Regression

Concluding Remarks

Introduction

Probabilistic approaches to classification

- In the last lecture we looked at linear models for classification from a purely geometric perspective.
- Like least squares regression, these lack the ability to quantify belief in their predictions.
- In this lecture we will look at linear classification from three probabilistic perspectives:
    - Generative: in which a class-conditional data likelihood and class priors will be used to derive a classification rule.
    - Discriminative: in which the posterior class distribution is directly estimated.
    - Bayesian: in which we approximate the parameter distribution from the data likelihood and prior and then integrate to make predictions.

Lecture objectives

At the end of this lecture you will:

- Understand the generative approach to classification and how the linear and quadratic discriminants derive from assumptions about class-conditional likelihoods.
- Understand the discriminative logistic regression approach to classification and how the negative log-likelihood loss can be used to train it.
- Understand the basics (and limits) of the Bayesian approach to classification and why approximate inference is needed.

Probabilistic Generative Models

Again, where are the probabilities?!

- Again, we find ourselves with a nice model, but one entirely unable to provide any measure of belief.
- We will now consider a specific type of generative view that will naturally lead (via Bayes rule) to just such a measure.
- As we saw for linear regression models, we will also find connections between the probabilistic and geometric views.

A generative model

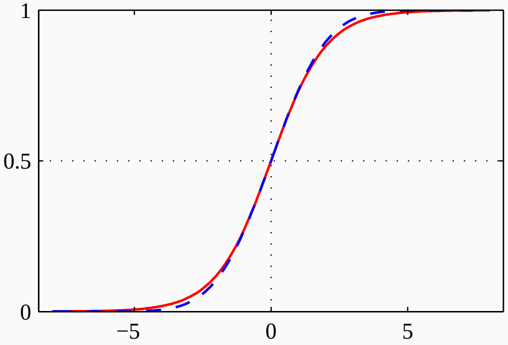
- For $K = 2$ class problems, we can write the posterior for class $\mathcal{C}_1$ as:

$$\begin{aligned} p(\mathcal{C}_1 \mid \mathbf{x}) &= \frac{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x} \mid \mathcal{C}_2)p(\mathcal{C}_2)} \\ &= \frac{1}{1 + \exp(-a)} \equiv \sigma(a(\mathbf{x})) \\ \text{for } a &= \ln \frac{p(\mathbf{x} \mid \mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x} \mid \mathcal{C}_2)p(\mathcal{C}_2)} \end{aligned}$$

- Writing the posterior in this way might seem like a waste of time.
- However, we will see that this helps generalize our results, especially when $a(\mathbf{x})$ has a simple form.

σ, a familiar friend

- The $\sigma(\cdot)$ function is known as the logistic sigmoid function.
- It plays a important role in many classification models.
- It is very important for Artificial Neural Networks.



K-class problems

- For the case of $K > 2$:

$$
\begin{aligned}
p(\mathcal{C}_k \mid \mathbf{x}) &= \frac{p(\mathbf{x} \mid \mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x} \mid \mathcal{C}_j)p(\mathcal{C}_j)} \\
&= \frac{\exp(a_k)}{\sum_j \exp(a_j)}
\end{aligned}
$$

- This is known as the normalized exponential or softmax function.
- Let's see what happens for a specific choice for a_k...

Generative models with continuous inputs

- We can assume that the class-conditional densities (another name for the likelihood) are Gaussian with equal covariance matrices:
- Thus, the density for class $\mathbf{C}_k$ is:

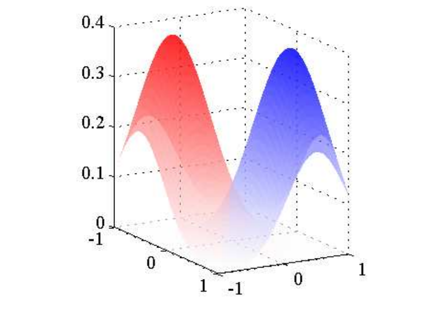
$$p(\mathbf{x} \mid \mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{-1}} \exp\left\{ -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right\}$$

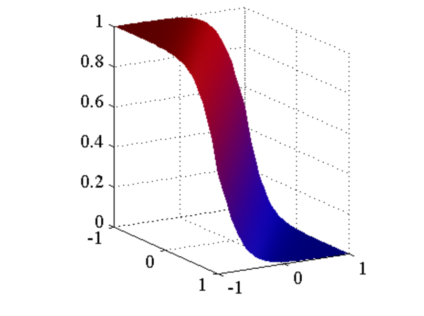
- If we consider only the first class, and recalling the analysis we made about the form of the posterior, we have:

$$\begin{aligned}
p(\mathcal{C}_1 \mid \mathbf{x}) &= \sigma(\mathbf{w}^T \mathbf{x} + w_0) \\
\text{where } \mathbf{w} &= \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \\
\text{and } w_0 &= -\frac{1}{2}\boldsymbol{\mu}_1^T \Sigma^{-1} \boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T \Sigma^{-1} \boldsymbol{\mu}_2 + \ln \frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)}
\end{aligned}$$

Generative models with continuous inputs

- The quadratic terms in $\mathbf{x}$ have canceled (due to the common Σ).
- The decision boundaries are linear in input space.





Generative models with continuous inputs

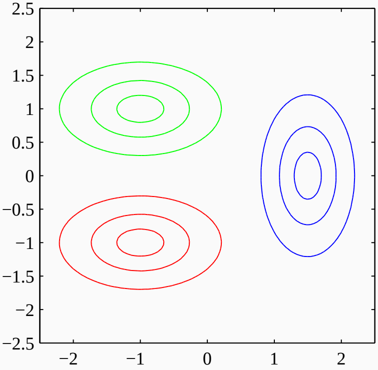
- For the general case of K classes, we use the softmax instead of sigmoid.
- We have:

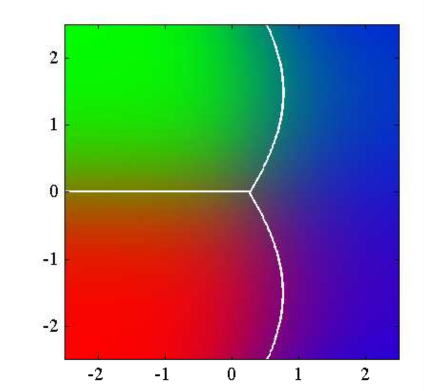
$$\begin{aligned} a_k(\mathbf{x}) &= \mathbf{w}_k^T\mathbf{x} + w_{k0} \\ \text{where } \mathbf{w}_k &= \Sigma^{-1}\boldsymbol{\mu}_k \\ \text{and } w_{k0} &= \frac{1}{2}\boldsymbol{\mu}_k^T\Sigma^{-1}\boldsymbol{\mu}_k + \ln p(\mathcal{C}_k) \end{aligned}$$

- The resulting decision boundaries are where two of the posteriors are equal.
- This corresponds to the minimum misclassification rate (again a linear function of $\mathbf{x}$).

Generative models with continuous inputs

- If we relax the requirement that all covariance matrices are equal, the quadratic terms no longer cancel.
- The result is a quadratic Bayes classifier.





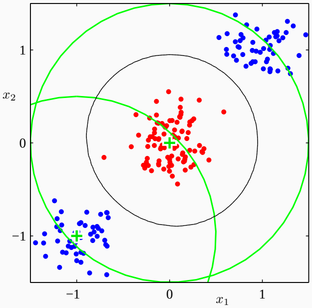
Probabilistic Discriminative Models

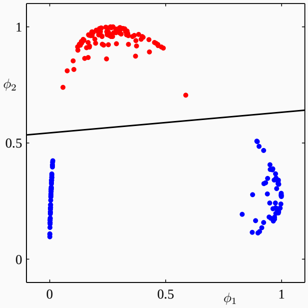
Generative versus discriminative

- We have seen that the posterior probability in a 2-class problem can be written as a logistic sigmoid of a linear function of $\mathbf{x}$.
- Similarly, for the multi-class case we have a softmax function on a linear function of $\mathbf{x}$.
- These are instances of what are known as generalized linear models.
- For specific choices of the class-conditional distributions we can use maximum likelihood to estimate their parameters (and those of the priors).
- These are sometimes called generative models, because we could generate samples $\mathbf{x}$ by sampling from the marginal $p(\mathbf{x})$.
- What if, instead, we use the functional form of the generalized linear model directly to discriminate?

Fixed basis functions

- We have developed all of classifiers to work directly on the original input $\mathbf{x}$.
- However, everything we have derived works equally well if we use a vector of basis functions $\phi(\mathbf{x})$.
- The resulting boundaries are linear in feature space ϕ, but nonlinear in the original space.





Logistic regression

- We have just seen that we can write the posterior for a 2-class problem as:

$$p(\mathcal{C}_1 \mid \phi) = \sigma(\mathbf{w}^T \phi)$$

- This is called (confusingly) a logistic regression model.
- For an M-dimensional feature space, this model has M parameters.
- The generative model would require $2M$ parameters for the means, plus $M(M+1)/2$ parameters for the covariance matrix.
- We can use Maximum Likelihood to fit the parameters of this model, the convenient form of the derivative of the logistic sigmoid:

$$\frac{d}{da}\sigma(a) = \sigma(a)(1 - \sigma(a))$$

Logistic regression

- For dataset $\mathcal{D} = \{\, \phi_n, t_n \,\}$, where $t_n \in \{\, 0, 1 \,\}$ and $\phi_n = \phi(\mathbf{x})$, the likelihood is:

$$\begin{aligned} p(\mathbf{t} \mid \mathbf{w}) &= \prod_{n=0}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n} \\ \text{where } \mathbf{t} &= (t_1, t_2, \ldots t_N)^T \\ \text{and } y_n &= p(\mathcal{C}_1 \mid \phi_n) \\ &= \sigma(\mathbf{w}^T \phi_n) \end{aligned}$$

- For our error function we will use the Negative Log-likelihood:

$$E(\mathbf{w}) = -\ln p(\mathbf{t} \mid \mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

Logistic regression

- Writing the error function in this way:

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

- And recalling that $y_n = \sigma(\mathbf{w}^T \phi_n)$.
- And using our observation about the derivative of the logistic sigmoid:

$$\frac{d}{da}\sigma(a) = \sigma(a)(1 - \sigma(a))$$

- Lets us see the connection more explicitly between likelihood and weights $\mathbf{w}$:

$$\nabla_{\mathbf{w}}(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\phi_n$$

Logistic regression

- Here we see why the model is called logistic regression:

$$\nabla_{\mathbf{w}}(\mathbf{w}) = \sum_{n=1}^{N}(y_n - t_n)\phi_n$$

- This is the same sequential learning update for linear regression with fixed basis functions ϕ.
- And we see that the objective is to regress the target $t_n \in \{0, 1\}$ from $\phi(\mathbf{x})$.
- Note: this Maximum Likelihood solution is highly prone to overfitting when $\mathcal{C}_1$ and $\mathcal{C}_2$ are linearly separable.
- In this case, $||\mathbf{w}||_2$ will go to infinity, converging to a posterior estimate in which for all $\mathbf{x}$, and for some k, $p(\mathcal{C}_k \mid \mathbf{x}) = 1$.

Bayesian Logistic Regression

The recipe for Bayesian ML

- We developed a step towards a recipe for full Bayesian learning in our discussion about regression.
- Let's try to apply it to our classification problem:
  1. Decide on a prior $p(\mathbf{w})$.
  2. Maximize the resulting posterior to arrive at a parameter distribution $p(\mathbf{w} \mid \mathbf{t})$.
  3. Derive the predictive distribution $p(\mathcal{C}_k \mid \boldsymbol{\Phi}, \mathbf{t})$ that we can use on new data $\phi(\mathbf{x})$.
- Step 1 is "easy" – although it is one of the primary criticisms of Bayesian learning.
- Steps 2 and 3 are, unfortunately, intractable: the posterior distribution over the parameters is no longer Gaussian.
- Let's see what we can do.

The Laplace approximation

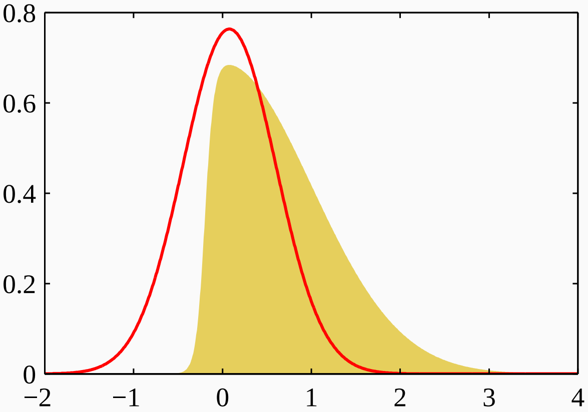
- First, assume a Gaussian prior: $p(\mathbf{w}) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}_0, \mathbf{S}_0)$.
- First step, approximate the parameter distribution.
- A simple method is known as the Laplace Approximation that uses the best Gaussian approximation.

$$q(\mathbf{w}) = \mathcal{N}(\mathbf{w} \mid \mathbf{w}_{\text{MAP}}, \mathbf{S}_N)$$

- Where the parameters are derived from the Maximum a Posteriori (MAP) estimate of the mean and covariance given my minimizing a second-order approximation of the true posterior.

The Laplace approximation

- Here is an example of approximating $p(z) \propto \exp(-z^2/2)\sigma(20z+4)$:



The predictive distribution

- Armed with this approximation we can now write the (approximate) predictive distribution:

$$p(\mathcal{C}_1 \mid \boldsymbol{\phi}, \mathbf{t}) = \int p(\mathcal{C}_1 \mid \boldsymbol{\phi}, \mathbf{w}) p(\mathbf{w} \mid \mathbf{t}) d\mathbf{w} \approx \int \sigma(\mathbf{w}^T \boldsymbol{\phi}) q(\mathbf{w}) d\mathbf{w}$$

- This is a convolution of a logistic sigmoid and a Gaussian, which can be approximated (after very lengthy derivations) as:

$$\begin{aligned} p(\mathcal{C}_1 \mid \boldsymbol{\phi}, \mathbf{t}) &\approx \sigma(\kappa(\sigma_a^2)\mu_a) \\ \text{where } \kappa(\sigma^2) &= (1 + \pi\sigma_a^2/8)^{-1/2} \\ \text{for } \sigma_a^2 &= \mathrm{var}[a] = \boldsymbol{\phi}^T \mathbf{S}_N \boldsymbol{\phi} \\ \text{and } \mu_a &= \mathbf{w}_{\mathrm{MAP}}^T \boldsymbol{\phi} \end{aligned}$$

Concluding Remarks

Probabilistic generative models

- The generative view of $p(\mathcal{C}_k \mid \mathbf{x})$ is appealing for a number of reasons.
- Under Maximum Likelihood estimation of parameters, we just estimate a distribution for class-conditional likelihoods $p(\mathbf{x}|\mathcal{C}_k)$ and posteriors $p(\mathcal{C}_k)$.
- For the Gaussian case, these estimates turn out to be the “usual” ones.
- If we assume equal covariance for all classes, the result is a linear classifier.
- Instead, if we estimate a Σ_k for each class the resulting classifier is quadratic in the input.

Logistic Regression

- Logistic regression is an extremely important model because nearly all Deep Neural Networks for classification are performing multi-class logistic regression.
- A Deep Network estimates the feature embedding $\phi \mathbf{x}$, then a linear function and softmax are applied.
- Then a Negative Log Likelihood – also known as a Cross Entropy – loss is applied.
- Despite its problems, it is a model quite suited to incremental, gradient-based optimization.
- Important: the fact that the outputs sum to one, means little in terms of probabilistic interpretation of the result.

Bayesian Logistic Regression

- Exact Bayesian inference is intractable due to the complexity of the data likelihood.
- And the need to normalize the posterior – we can't just ignore the evidence factor in Bayes rule any longer.
- There are very sophisticated techniques to approximate normalized posteriors:
    - Laplace's Method: approximate with a Gaussian.
    - Variational Inference: match a proxy distribution to posterior.
    - Monte Carlo Methods: use Markov chain sampling for integration.

Reading and Homework Assignments

Reading Assignment:

- Bishop: Chapter 4 (4.2, 4.3, 4.4*, 4.5*)