Linear Classifiers 3

Christian Hardmeier

2018-04-18

Machine Learning and probability

- Is machine learning based on probability?
  - Yes – all machine learning is based on inductive inference
  - No – we do not need an explicit probability model
- Two roles for probability theory:
  - Theoretical analysis of learning methods
  - Practical use in learning methods

What do we get from probability theory?

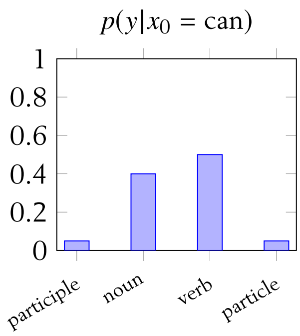
- A principled framework for reasoning under uncertainty.
- Methods for
  - specifying assumptions we're making
  - assessing the uncertainty of our predictions

Classification in a probabilistic framework

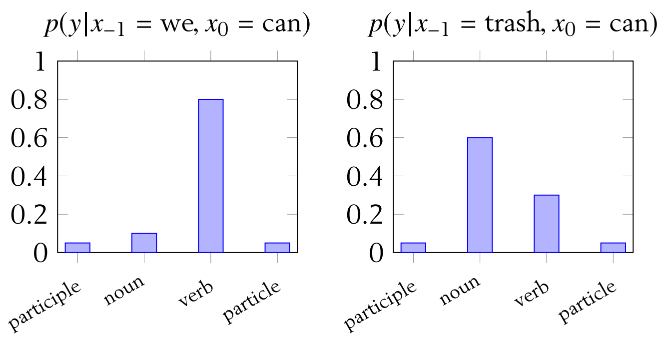
- Estimate a probability distribution over all possible outcomes given your features.
- Predict the class with the highest probability:

$$\hat{y} = \arg\max_y p(y|\mathbf{x})$$

Conditioning on features



Conditioning on features



Bayes optimal classifier

- If you know the distribution generating your data $p(\mathbf{x}, y)$, classification is easy.
- The Bayes optimal classifier has optimal 0-1 loss of all possible classifiers.

$$f_{\mathrm{BO}}(x) = \arg\max_y p(\mathbf{x}, y)$$

- The Bayes optimal classifier gives a lower bound on the error rate.
- But usually we don't know $p(\mathbf{x}, y)$.
- Just counting training data will generalise very poorly!

Decomposing the joint probability

$$\begin{aligned} p(\mathbf{x}, y) &= p(y, x_1, \ldots, x_k) \\ &= p(y) \prod_k p(x_k | y, x_1, \ldots, x_{k-1}) \end{aligned}$$

- This decomposition is valid for any distribution.
- But $p(x_k | y, x_1, \ldots, x_{k-1})$ is difficult to estimate reliably.

Naive Bayes assumption

- We can simplify the problem by making independence assumptions.
- Naive Bayes assumption:
  Features are conditionally independent given the labels.
- Example: If a review is positive,
  the occurrence of the word "amazing"
  is independent of the occurrence of "excellent".
- This reduces the number of parameters and makes the model easier to estimate.
- We can then parametrise the component distributions.

Naive Bayes: Generative story

(for binary classification with binary labels)

- Select a label by drawing from a Bernoulli distribution.

$$
\begin{aligned}
p_\theta(y) &= \begin{cases} \theta_0 & \text{if } y = +1 \\ 1 - \theta_0 & \text{if } y = -1 \end{cases} \\
&= \theta_0^{[y=+1]}(1-\theta_0)^{[y=-1]}
\end{aligned}
$$

- For each feature x_k: Select a feature value from a Bernoulli distribution conditioned on the label y.

$$p_\theta(x_k|y) = \theta_{y,k}^{[x_k=1]}(1-\theta_{y,k})^{[x_k=0]}$$

Complete model

$$
\begin{aligned}
p_\theta(\mathbf{x}, y) &= p_\theta(y, x_1, \ldots, x_k) \\
&= p_\theta(y) \prod_k p_\theta(x_k|y, x_1, \ldots, x_{k-1}) \\
&\approx p_\theta(y) \prod_k p_\theta(x_k|y) \\
&= \theta_0^{[y=+1]}(1-\theta_0)^{[y=-1]} \prod_k \theta_{y,k}^{[x_k=1]}(1-\theta_{y,k})^{[x_k=0]}
\end{aligned}
$$

To obtain the probability of the entire corpus, we multiply the probabilities of all examples:

$$p_\theta(\mathcal{T}) = \prod_{(\mathbf{x},y)\in\mathcal{T}} p_\theta(\mathbf{x}, y)$$

How to train this?

- At training time, we want to find the best parameters for the training set.
- A standard way to do this is to maximise the likelihood of the training set.
- The likelihood is the probability *as a function of the parameters*.
- In the likelihood function, the parameters are regarded as variable and the data as fixed!

$$\hat{\theta} = \arg\max_\theta \mathcal{L}(\theta; \mathcal{T}) = \arg\max_\theta p_\theta(\mathcal{T})$$

Notes on the optimisation

- We usually optimise the logarithm of the likelihood (log-likelihood) to turn products into sums.
- The parameters θ of our model are probabilities and must be between 0 and 1.
  This is a constrained optimisation problem.
- In this case, the resulting estimates are simple relative frequencies:

$$\hat{\theta}_0 = \frac{1}{|\mathcal{T}|} \sum_{(\mathbf{x},y)\in\mathcal{T}} [y = +1], \quad \hat{\theta}_{y,k} = \frac{\sum_{(\mathbf{x},y^*)\in\mathcal{T}}[y^* = y \wedge x_k = 1]}{\sum_{(\mathbf{x},y^*)\in\mathcal{T}}[y^* = y]}$$

- The model has a linear decision boundary.

Generative and discriminative models

- Naive Bayes is a generative probabilistic model because it models the joint distribution of inputs and outputs, $p(\mathbf{x}, y)$.
- An alternative approach is to model the conditional distribution, $p(y|\mathbf{x})$.
- This is sufficient for classification since the input is given whenever the classifier is queried.

Modelling the conditional distribution

- To create a *discriminative* classifier, we parametrise the distribution $p(y|\mathbf{x})$ directly.
- Define any suitable function that gives us a probability, then adjust the parameters by training.

A discriminative linear classifier

- Let's start with a linear classifier:

$$\hat{y} = \mathbf{w} \cdot \mathbf{x} + b$$

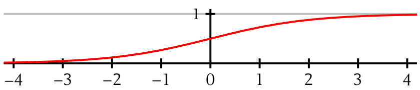
- The margin $\hat{y}$ is an indicator of the classification confidence.
- But it can't be used as a probability – its range is all of $\mathbb{R}$.
- What if we use some mapping $g : \mathbb{R} \to (0, 1)$?

Sigmoid transform

- The logistic sigmoid function maps $\mathbb{R}$ to $(0, 1)$:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- It has many useful properties:
  - As x becomes large, it quickly approaches 1.
  - As x becomes small, it quickly approaches 0.
  - It is symmetric: $\sigma(-x) = 1 - \sigma(x)$.
  - It has a nice derivative: $\frac{d\sigma}{dx} = \sigma(x)(1 - \sigma(x))$.



A discriminative linear classifier

$$p(y|\mathbf{x}) = \sigma(\mathbf{w} \cdot \mathbf{x} + b) = \frac{1}{1 + \exp(-\mathbf{w} \cdot \mathbf{x} - b)}$$

$$\begin{aligned}
\sum_{(\mathbf{x},y)\in\mathcal{T}} \log p(y|\mathbf{x}) &= \sum_{(\mathbf{x},y)\in\mathcal{T}} \begin{cases} \log \sigma(\mathbf{w} \cdot \mathbf{x} + b) & \text{if } y = 1 \\ \log(1 - \sigma(\mathbf{w} \cdot \mathbf{x} + b)) & \text{if } y = -1 \end{cases} \\
&= \sum_{(\mathbf{x},y)\in\mathcal{T}} \log \sigma(y(\mathbf{w} \cdot \mathbf{x} + b)) \\
&= \sum_{(\mathbf{x},y)\in\mathcal{T}} \log \frac{1}{1 + \exp(-y(\mathbf{w} \cdot \mathbf{x} + b))} \\
&= -\sum_{(\mathbf{x},y)\in\mathcal{T}} \log\left(1 + \exp(-y(\mathbf{w} \cdot \mathbf{x} + b))\right)
\end{aligned}$$

Logistic regression

- Maximising the log-likelihood of the sigmoid-transformed linear model is exactly equivalent to minimising the logistic loss function!
- This is called logistic regression.
- There is also a link to information theory: This classifier maximises entropy given a set of constraints.
- Logistic regression is the discriminative counterpart of Naive Bayes.

Maximum a posteriori (MAP) estimation

- Often, we have a prior notion of what our probability distributions should look like.
- Then we can use a maximum a posteriori (MAP) estimate:

$$\hat{\theta} = \arg\max_{\theta} p(\mathcal{T}|\theta)p(\theta)$$

- Using a prior is often referred to as smoothing (or regularisation) because it smoothes the sharp MLE.
- The special case of a Gaussian prior on the weights results in ℓ_2 regularisation:

$$p(\theta_k; \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

Generative vs. discriminative models

Generative models:

- Informative – other distributions can be derived:

$$p(\mathbf{x}) = \sum_y p(\mathbf{x}, y) \qquad p(y|\mathbf{x}) = \frac{p(\mathbf{x}, y)}{\sum_y p(\mathbf{x}, y)}$$

- Can be used to generate data.
- May work with fewer data points.

Discriminative models:

- Only models the distribution relevant for classification.
- Less rigid independence assumptions.
- Often performs better given enough data.

Methods covered so far

- Decision trees
- k-nearest neighbours
- Perceptron
- Linear classifiers
  - Hinge loss
  - Logistic loss
  - Exponential loss
  - Squared loss
- ℓ_1 and ℓ_2 regularisation
- Naive Bayes, generative models

Choosing the right method

6 scenarios

Tools for classification

- General libraries:
  - Weka (GUI): `https://www.cs.waikato.ac.nz/ml/weka/`
  - scikit-learn (Python): `http://scikit-learn.org/`
- Nearest neighbours:
  TiMBL: `https://languagemachines.github.io/timbl/`
- Linear classifiers:
  - libsvm: `https://www.csie.ntu.edu.tw/~cjlin/libsvm/`
  - liblinear:
    `https://www.csie.ntu.edu.tw/~cjlin/liblinear/`
  - MegaM:
    `https://www.umiacs.umd.edu/~hal/megam/version0_3/`

Next up

- Next lecture (with Joakim): 26th April, 14–16 in 2-0024
- Assignment 2 due: 27th April