Lecture 6: Non-parametric methods and probability theory

EMAT31530/Feb 2018/Raul Santos-Rodriguez

Have a look at ...

... Russell and Norvig (Ch. 13)

... Hastie, Tibshirani, Friedman. The elements of statistical learning, (Ch. 1)

... Python: `http://scikit-learn.org/`

... Java: `http://www.cs.waikato.ac.nz/ml/weka/`

Outline

This lecture presents an example of non-parametric models and a describes a probabilistic framework for Machine Learning. You will study:

- Nearest Neighbours.
- Brief review of probability theory.
- Statistical learning.

Exploiting similarity: Nearest Neighbours

Algorithm

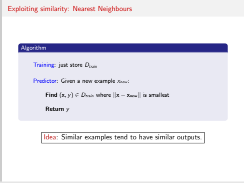
```
Training: just store $D_{train}$

Predictor: Given a new example $x_{new}$:

    Find $(\mathbf{x}, y) \in D_{train}$ where $||\mathbf{x} - \mathbf{x_{new}}||$ is smallest

    Return $y$
```

Idea: Similar examples tend to have similar outputs.



Exploiting similarity: Nearest Neighbours

Algorithm

Training: just store D_{train}

Predictor: Given a new example x_{new}:

```
    Find $(\mathbf{x}, y) \in D_{train}$ where $||\mathbf{x} - \mathbf{x_{new}}||$ is smallest
    Return $y$
```

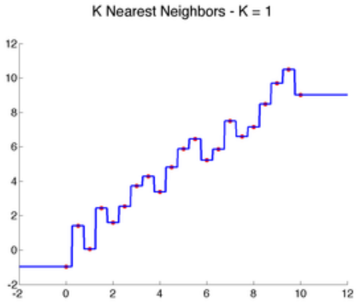
Idea: Similar examples tend to have similar outputs.

We will consider nearest neighbours, which is based on computing similarities between examples.

Nearest neighbours is perhaps conceptually one of the simplest learning algorithms. In a way, there is no learning. At training time, we just store the entire training examples. At prediction time, we get an input x_{new} and we just find the input in our training set that is most similar, and return its output. The intuition being expressed here is that similar (nearby) points tend to have similar outputs. This is a reasonable assumption in most cases.

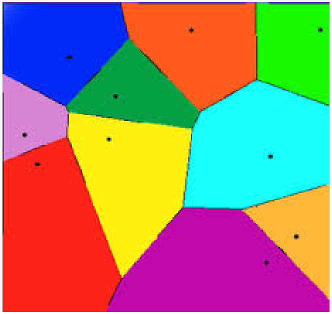
NN for regression

Also called piecewise constant interpolation: locate the nearest data value, and assign the same value.

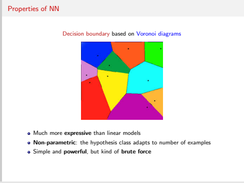


Properties of NN

Decision boundary based on Voronoi diagrams

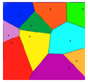


- Much more **expressive** than linear models
- **Non-parametric**: the hypothesis class adapts to number of examples
- Simple and **powerful**, but kind of **brute force**



Properties of NN

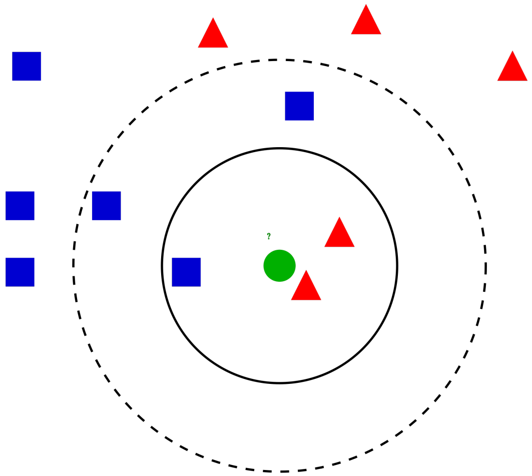
Decision boundary based on Voronoi diagrams



- Much more **expressive** than linear models
- **Non-parametric**: the hypothesis class adapts to number of examples
- Simple and **powerful**, but kind of **brute force**

Let's look at the decision boundary of nearest neighbours. The input space is partitioned into regions, such that each region has the same closest point (this is a Voronoi diagram), and each region could get a different output. Notice that this decision boundary is much more expressive than what you could get with linear models. In particular, one interesting property is that the complexity of the decision boundary adapts to the number of training examples. As we increase the number of training examples, the number of regions will also increase. Such methods are called non-parametric. They are powerful and easy to learn, but are slow to use for prediction because they involve enumerating (or looking up points in) the training data.

KNN



http://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html#sphx-glr-auto-examples-neighbors-plot-classification-py

Probability theory

- Let X be a discrete random variable taking values from the alphabet $\mathcal{X}$.

Probability distribution

The probability distribution of X is denoted by $p_X = \{p_X, x \in \mathcal{X}\}$, where

- $p_X(x)$ represents the probability that $X = x$
- $p_X(x) \geq 0$
- $\sum_x p_X(x) = 1$

- Let S_X be the support of X, i.e. $S_X = \{x \in X : p(x) > 0\}$.

Probability theory: example

Example

Let X be the outcome of a dice

- Let $\mathcal{X} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, \ldots\}$ equal to all positive integers. In this case, $\mathcal{X}$ is a countably infinite alphabet.
- $S_X = \{1, 2, 3, 4, 5, 6\}$ which is a finite alphabet.
- If the dice is fair, then $p_X(1) = p_X(2) = \cdots = p_X(6) = 1/6$.

If $\mathcal{X}$ is a subset of the real numbers, e.g., $\mathcal{X} = [0, 1]$, $\mathcal{X}$ is a continuous alphabet and X is a **continuous random variable**.

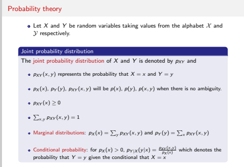
Probability theory

- Let X and Y be random variables taking values from the alphabet $\mathcal{X}$ and $\mathcal{Y}$ respectively.

Joint probability distribution

The joint probability distribution of X and Y is denoted by p_{XY} and

- $p_{XY}(x, y)$ represents the probability that $X = x$ and $Y = y$
- $p_X(x)$, $p_Y(y)$, $p_{XY}(x, y)$ will be $p(x)$, $p(y)$, $p(x, y)$ when there is no ambiguity.
- $p_{XY}(x) \geq 0$
- $\sum_{x,y} p_{XY}(x, y) = 1$
- Marginal distributions: $p_X(x) = \sum_y p_{XY}(x, y)$ and $p_Y(y) = \sum_x p_{XY}(x, y)$
- Conditional probability: for $p_X(x) > 0$, $p_{Y|X}(y|x) = \frac{p_{XY}(x,y)}{p_X(x)}$ which denotes the probability that $Y = y$ given the conditional that $X = x$



Probability theory

- Let X and Y be random variables taking values from the alphabet $\mathcal{X}$ and $\mathcal{Y}$ respectively.

Joint probability distribution

The joint probability distribution of X and Y is denoted by p_{XY} and

- $p_{XY}(x, y)$ represents the probability that $X = x$ and $Y = y$
- $p_X(x)$, $p_Y(y)$, $p_{XY}(x, y)$ will be $p(x)$, $p(y)$, $p(x, y)$ when there is no ambiguity.
- $p_{XY}(x) \geq 0$
- $\sum_{x,y} p_{XY}(x, y) = 1$
- Marginal distributions: $p_X(x) = \sum_y p_{XY}(x, y)$ and $p_Y(y) = \sum_x p_{XY}(x, y)$
- Conditional probability: for $p_X(x) > 0$, $p_{Y|X}(y|x) = \frac{p_{XY}(x,y)}{p_X(x)}$ which denotes the probability that $Y = y$ given the conditional that $X = x$

The marginal distribution of a subset of a collection of random variables is the probability distribution of the variables contained in the subset. It gives the probabilities of various values of the variables in the subset without reference to the values of the other variables. On the other hand, a conditional probability measures the probability of an event given that (by assumption, presumption, assertion or evidence) another event has occurred. For example, the probability that any given person has a cough on any given day may be only 5%. But if we know or assume that the person has a cold, then they are much more likely to be coughing. The conditional probability of coughing given that you have a cold might be a much higher 75%.

Probability theory: expectation and variance

Expectation

The expectation of X is given by $E[X] = \sum_x x \cdot p_X(x)$

Variance

The variance of X is given by $E[(X - E[X])^2] = E[X^2] - (E[X])^2$

- The expected value of $f(X)$ is $E[f(X)] = \sum_x f(x) \cdot p_X(x)$
- The expected value of $g(X, Y)$ is $E[g(X, Y)] = \sum_{x,y} g(x, y) \cdot p_{XY}(x, y)$
- We can also take the expectation on Y only, i.e., $E_Y[g(X, Y)] = \sum_y g(X, y) \cdot p_Y(y)$, which is still a random variable

Probability theory: independence

Independence

Two random variables X and Y are independent if $p(x, y) = p(x)p(y)\ \forall x, y$

Conditional independence

For random variables X, Y, and Z, X and Z are conditionally independent given Y, denoted by $X \perp Z|Y$ if

$$p(x, z|y) = p(x|y)p(z|y)$$

or

$$p(x|y) = p(x|z, y)$$

X and Z are conditionally independent given Y if and only if, given knowledge that Y occurs, knowledge of whether X occurs provides no information on the likelihood of Z occurring, and knowledge of whether Z occurs provides no information on the likelihood of X occurring.

Probability theory: example

$\boxed{\text{Whiteboard}}$

Statistical decision theory

- Let $X \in \mathbb{R}^d$ denote a real valued random input vector, and $Y \in \mathbb{R}$ a real valued random output variable, with joint distribution $p(x, y)$.
- We seek a function $f(X)$ for predicting Y given the values of the input X.
- This theory requires a Loss function $L(Y, f(X))$ for penalising errors in prediction, e.g. *Squared error loss*

$$L(Y, f(X)) = (Y - f(X))^2$$

Expected prediction error

Criterion for choosing f, the expected prediction error:

$$EPE(f) = \int L(y, f(x))p(x, y)dxdy$$

Example: Least squares

$$EPE(f) = E(Y - f(X))^2 = \int (y - f(x))^2 p(x, y) dxdy$$

By conditioning on X,

$$EPE(f) = E_X E_{Y|X}[(Y - f(X))^2 | X]$$

and it suffices to minimise EPE pointwise:

$$f^*(x) = \arg\min_c E_{Y|X}[(Y - c)^2 | X = x]$$

The solution is

$$f^*(x) = E[Y | X = x]$$

The best prediction of Y at any point $X = x$ is the conditional mean, when best is measured by average squared error.

Example: Least squares vs K-nearest neighbours

The nearest-neighbour method attempts to directly implement the Least squares objective using the training data.

K-nearest neighbours

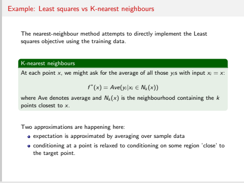
At each point x, we might ask for the average of all those y_is with input $x_i = x$:

$$f^*(x) = Ave(y_i | x_i \in N_k(x))$$

where Ave denotes average and $N_k(x)$ is the neighbourhood containing the k points closest to x.

Two approximations are happening here:

- expectation is approximated by averaging over sample data
- conditioning at a point is relaxed to conditioning on some region 'close' to the target point.



Example: Least squares vs K-nearest neighbours

The nearest-neighbour method attempts to directly implement the Least squares objective using the training data.

K-nearest neighbours

At each point x, we might ask for the average of all those y_is with input $x_i = x$:

$$f^*(x) = Ave(y_i | x_i \in N_k(x))$$

where Ave denotes average and $N_k(x)$ is the neighbourhood containing the k points closest to x.

Two approximations are happening here:

- expectation is approximated by averaging over sample data
- conditioning at a point is relaxed to conditioning on some region 'close' to the target point.

So both K-nearest neighbours and least squares end up approximating conditional expectations by averages. But they differ dramatically in terms of model assumptions.

- Least squares assumes $f(x)$ is well approximated by a globally linear function.
- K-nearest neighbours assumes $f(x)$ is well approximated by a locally constant function.

Classification

$0-1$ loss function: all misclassifications are charged a single unit.

The expected prediction error is

$$EPE = E[L(y, \hat{y}(x))]$$

We condition, and can write EPE as

$$EPE = E_X \left[\sum_{y=1}^{C} L(y, \hat{y}(x)) p(y|x) \right]$$

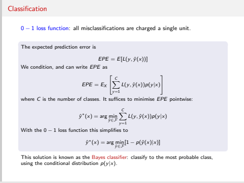
where C is the number of classes. It suffices to minimise EPE pointwise:

$$\hat{y}^*(x) = \arg\min_{\hat{y} \in \mathcal{F}} \sum_{y=1}^{C} L(y, \hat{y}(x)) p(y|x)$$

With the $0-1$ loss function this simplifies to

$$\hat{y}^*(x) = \arg\min_{\hat{y} \in \mathcal{F}} [1 - p(\hat{y}(x)|x)]$$

This solution is known as the Bayes classifier: classify to the most probable class, using the conditional distribution $p(y|x)$.



Classification

$0-1$ loss function: all misclassifications are charged a single unit.

The expected prediction error is

$$EPE = E[L(y, \hat{y}(x))]$$

We condition, and can write EPE as

$$EPE = E_X \left[\sum_{y=1}^{C} L(y, \hat{y}(x)) p(y|x) \right]$$

where C is the number of classes. It suffices to minimise EPE pointwise:

$$\hat{y}^*(x) = \arg\min_{\hat{y} \in \mathcal{F}} \sum_{y=1}^{C} L(y, \hat{y}(x)) p(y|x)$$

With the $0-1$ loss function this simplifies to

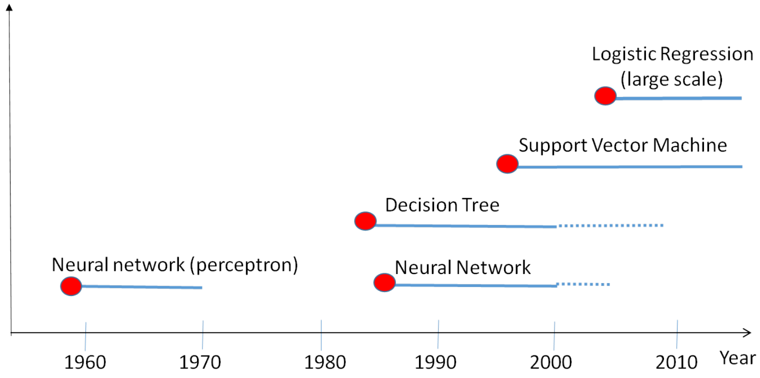
$$\hat{y}^*(x) = \arg\min_{\hat{y} \in \mathcal{F}} [1 - p(\hat{y}(x)|x)]$$

This solution is known as the Bayes classifier: classify to the most probable class, using the conditional distribution $p(y|x)$.

See that the K-nearest neighbours classifier directly approximates this solution - a majority vote in a nearest neighbourhood amounts to exactly this, except that conditional probability at a point is relaxed to conditional probability within a neighbourhood of a point, and probabilities are estimated by training-sample proportions.

A bit of history

1795: Gauss proposed least squares (astronomy)
1940s: Logistic regression (statistics)
1952: Arthur Samuel built program that learned to play checkers (AI)
1957: Rosenblatt invented Perceptron algorithm (like SGD)



`http://www.economist.com/news/obituary/21692845-marvin-minsky-pioneer-artificial-intelligence-died-january-24th-aged-88-mind-and`

Next lecture

We will discuss the main concepts of information theory. In the meantime ...

... something to read: `http://www.theverge.com/2016/2/11/10965142/microsoft-fetch-ios-app-dog-breed-identifier`

... something to think about:
probability of error of 1-NN $\leq 2\times$ probability of error of Bayes classifier