COMP30030: Introduction to Artificial Intelligence

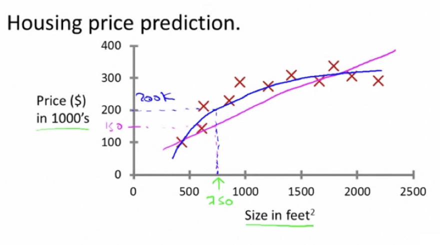
Neil Hurley

School of Computer Science
University College Dublin
neil.hurley@ucd.ie

November 1, 2018

Supervised vs Unsupervised Machine Learning I

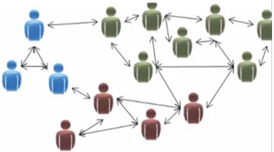
- Supervised Machine Learning
  - Classification
    - Associate a label with problem instances, given examples of problems and their labels.
    - E.g. Label an email as SPAM or NOT SPAM
  - Regression
    - Make a real-valued prediction, based on example data



Supervised vs Unsupervised Machine Learning II

- Unsupervised Machine Learning
  - No labelled answers provided in the training set.
  - Instead data is provided and the task is to find structure or interesting patterns within the data.

Organize computing clusters



Social network analysis

Market segmentation

Astronomical data analysis

1. **Problem Solving by Search**
   - Uninformed Search
   - Informed Search
   - Adversarial Search
   - Game Playing with Reinforcement Learning

2. **Optimisation**
   - Optimisation Overview
   - Combinatorial Optimisation Problems
   - Simulated Annealing
   - Optimisation Problem Examples
   - Convergence of Simulated Annealing
   - Genetic Algorithms
   - Optimisation in Continuous Spaces

3. **Machine Learning**
   - Supervised Machine Learning

Classification I

- To summarise:
  - A supervised classification algorithm is given data in the form of a set of input/output pairs $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i$ represents an instance or example from the problem domain and y_i represents the class to which it belongs.
  - The algorithm then uses these inputs to train a model, so that when presented with a new unseen problem instance $\mathbf{z}$, it can predict which class $\mathbf{z}$ belongs to.
  - The function that the algorithm learns is called the hypothesis.
  - For a classification task, the function should map problem instances to one of a finite number of **labels**.

Representing instances I

Before thinking of an appropriate classification algorithm, an important task is to determine how the problem instances should be represented.

This generally involves forming a feature vector representation i.e. describing each instances as values for a set of features.

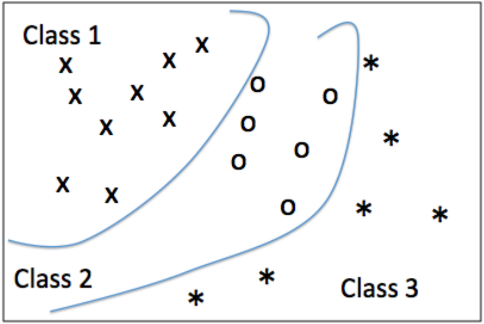
- Imagine a machine learning algorithm that wants to determine if an email is spam or not.
- this is a binary classification problem: each instance can belong to one of two classes – either spam or not spam.
- A feature vector presentation would extract the important information inside each email into a set of feature values e.g.
    1. A count of the number of “spam” words in the email (assuming a list of such “spam” words exist).
    2. A measure of the writing quality of the email e.g. do the sentences in the email represent typical sentences.
- Forming the feature vector representation requires the use of domain knowledge.

Representing instances II

- For example, a feature vector for spam detection in the Machine Learning Repository `https://archive.ics.uci.edu/ml/datasets/Spambase` contains the following 57 features:
  1. 48 continuous real [0,100] attributes = % words in the e-mail that match WORD,
  2. 6 continuous real [0,100] attributes =% characters in the e-mail that match CHAR
  3. 1 continuous real [1,...] attribute = average length of uninterrupted sequences of capital letters
  4. 1 continuous integer [1,...] attribute = length of longest uninterrupted sequence of capital letters
  5. 1 continuous integer [1,...] attribute = total number of capital letters in the e-mail

Separating points in space

- Given that each instance can be represented by, say m values – then each feature vector is a point in m-dimensional space.
- The classification problem then becomes one of identifying regions in space corresponding to each class label.



Linear Separators I

- While the boundaries in the above picture look quite complex, simple classifiers find linear separators, partitioning the feature space in two, in order to solve a binary classification problem.
- Mathematically, we can write a linear separator as an expression of the form

$$\sum_{i=1}^{m} w_i x_i + b$$

  where x_i is the i^{th} component of the feature vector, w_i is a weight and b is a bias.
- If we take x_0 to be an extra input that is always 1 and write $w_0 = b$, then we can write the separator in $m + 1$ dimensional space as

$$\sum_{i=0}^{m} w_i x_i$$

Linear Separators II

- The class can then be determined by a rule $h(\mathbf{x}) = \sum_{i=0}^{m} w_i x_i \gtrless 0$.

- Write $h_j = h(\mathbf{x}_j)$, to represent the value of the hypothesis at the instance.
- Represent one class by the value $y_i = 1$ and the other by the value $y_i = -1$.
- We can write down a loss function, $\mathcal{L}(y_j, h_j)$ that represents the loss or penalty incurred for getting the prediction wrong. For example, a squared loss function would incur a squared penalty:

$$\mathcal{L}_{\mathrm{square}}(y_j, h_j) = (y_j - h_j)^2 \tag{1}$$

Linear Separators III

- Now we can formulate the classification problem, as one which finds the weights w_i, so that the empirical risk is minimised where this is defined as the average loss over all training instances:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{N} \sum_{j=1}^{N} \mathcal{L}(y_j, h_j) = \frac{1}{N} \sum_{j} \mathcal{L}(y_j - \sum_{i=0}^{m} w_i x_i)$$

- We often write Loss functions in terms of a single variable $m_j = y_j h_j$. Multiplying the squared loss by $1 = y_j^2$, we get

$$\begin{aligned}(y_j - h_j)^2 &= y_j^2 (y_j - h_j)^2 = (y_j(y_j - h_j))^2 \\ &= (y_j^2 - y_j h_j)^2 = (1 - m_j)^2\end{aligned}$$

Linear Separators IV

- Two other loss function:

$$\mathcal{L}_{\text{logistic}} = \log(1 + exp(-m_j)) \quad \text{Logistic loss function}$$
$$\mathcal{L}_{\text{hinge}} = \max(0, 1 - m_j) \quad \text{Hinge loss function}$$

- Below is plot of some loss functions against $m = y_j h_j$.
- Note that for all loss functions except Logistic, $L(m) = 0$, when $m = y_j h_j = 1$, which corresponds to $y_j = h_j = 1$ or $y_j = h_j = -1$ i.e. the hypothesis is correct.

Linear Separators V