# COSC343: Artificial Intelligence

Lecture 15 : Supervised learning – a review

Lech Szymanski

Dept. of Computer Science, University of Otago



## In today's lecture

- Review of the supervised learning techniques we learned about
- A classification example using all these techniques



## Supervised learning: the problem

We have a problem, where a machine needs to make an inference about something based on observations

- Inference is a decision (or prediction) about something we are interested in
  - Is there a face in the image, and if so where is it?
  - Is the patient at a risk of a heart disease?
- Observations are the data that machine uses to make its decision (or prediction)
  - Pixel values in the image.
  - Attributes of the patient + results of various medical tests.
- Formulate the problem as function of the observed data that produces output value(s) which can be interpreted as machine's decision (or prediction)
  - Input is a single image as a vector of pixel values; one output is a 0 or 1 value corresponding to no-face or face, other output correspond to the location of the face specified as square region of interest within the image.
  - Input is patient's age, blood pressure, cholesterol test; output is a 0 or 1 value corresponding to healthy or heart disease.



## Supervised learning: the challenge

Typically we don't know what the relationship between input and output is...

- Though our brains do it with ease, it's hard to engineer an algorithm that detects a face in an image.
- Doctors diagnosis is a combination of experience and intuition...which is hard to explain and not immune to mistakes, when there are many factors to consider.

....but we do have a data sample with examples of inputs and corresponding known outputs.

- Set of images with labels indicating the presence of the face (or not), and the location of the face if there's on in the image (presumably labelled by a person).
- Information and test results for a number of patients were diagnosed by a doctor.

Often the training data is not perfect (there's noise)

- Things in the images that are not faces, things obstructing the face, multiple faces, wrongly labelled data.
- Errors in the tests, errors in record keeping, bad diagnosis.

## Supervised learning: the training

1. **Examine the data** (set a portion aside for testing)
2. **Make a new hypothesis** – pick a computational model/learner
3. **Train the model** – find parameter values that fit the hypothesis to the training data, i.e. the model produces output that is similar to the desired output for the training set.
4. **Test the model** on the test data to check if the hypothesis generalises well. If it doesn't, go back to Step 2.
5. **Use the model** to make predictions on new data.



## Supervised learning: making a hypothesis

It takes some time, and experience of using different models on different datasets, in order to develop an intuition for what hypothesis/learner might work well in what situation. Some principles to keep in mind:

- The ultimate objective is **generalisation** – poor performance on test data despite good performance on training data is a sign of **overtraining**.
- Aim for the **simplest hypothesis** that is **fairly consistent** with the training data – more complex models have more representational power which is prone to overfitting.



## A problem: Heart disease diagnosis

Observations (input of 5-dimensions for a given patient)

Decision that needs to be made

$x_1$ $x_2$ $x_3$ $x_4$ $x_5$ $\tilde{y}$

| | Sample | Age | Sex | Chest pain type<br>1 – typical angina<br>2 – atypical angina<br>3 – non-anginal pain<br>4 – asymptomatic | Resting blood pressure (mm Hg) | Cholesterol Level (mg/dL) | Heart disease |
|---|---|---|---|---|---|---|---|
| $\mathbf{x}_1$ | 1 | 54 | F | 3 | 110 | 214 | N |
| $\mathbf{x}_2$ | 2 | 68 | M | 4 | 144 | 193 | Y |
| $\mathbf{x}_3$ | 3 | 64 | M | 3 | 140 | 335 | Y |
| $\mathbf{x}_4$ | 4 | 58 | F | 1 | 150 | 283 | N |
| | . . . | . . . | . . . | . . . | . . . | . . . | . . . |
| $\mathbf{x}_{297}$ | 297 | 53 | F | 4 | 138 | 234 | N |

http://archive.ics.uci.edu/ml/datasets/Heart+Disease



## A problem: Heart disease diagnosis

Continuous Discrete {F,M} Discrete {1,2,3,4} Continuous Continuous

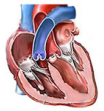
Decision that needs to be made

$\tilde{y}$

| | Sample | Age | Sex | Chest pain type<br>1 – typical angina<br>2 – atypical angina<br>3 – non-anginal pain<br>4 – asymptomatic | Resting blood pressure (mm Hg) | Cholesterol Level (mg/dL) | Heart disease |
|---|---|---|---|---|---|---|---|
| $\mathbf{x}_1$ | 1 | 54 | F | 3 | 110 | 214 | N |
| $\mathbf{x}_2$ | 2 | 68 | M | 4 | 144 | 193 | Y |
| $\mathbf{x}_3$ | 3 | 64 | M | 3 | 140 | 335 | Y |
| $\mathbf{x}_4$ | 4 | 58 | F | 1 | 150 | 283 | N |
| | . . . | . . . | . . . | . . . | . . . | . . . | . . . |
| $\mathbf{x}_{297}$ | 297 | 53 | F | 4 | 138 | 234 | N |

http://archive.ics.uci.edu/ml/datasets/Heart+Disease

# A problem: Heart disease diagnosis



- Use 237 randomly picked patients for training
- Use the remaining 60 for testing

| Sample | Age | Sex | Chest pain type<br>1 – typical angina<br>2 – atypical angina<br>3 – non-anginal pain<br>4 – asymptomatic | Resting blood pressure (mm Hg) | Cholesterol Level (mg/dL) | Heart disease |
|---|---|---|---|---|---|---|
| 1 | 54 | F | 3 | 110 | 214 | N |
| 2 | 68 | M | 4 | 144 | 193 | Y |
| 3 | 64 | M | 3 | 140 | 335 | Y |
| 4 | 58 | F | 1 | 150 | 283 | N |
| . . . | . . . | . . . | . . . | . . . | . . . | . . . |
| 297 | 53 | F | 4 | 138 | 234 | N |

http://archive.ics.uci.edu/ml/datasets/Heart+Disease



# Naive Bayes Classifier

- Assuming independence of attributes, compute the probability of each input (symptom) given each possible output value (diagnosis).

$$\prod_j p(x_j|\tilde{y}=Y)p(\tilde{y}=Y)$$

vs.

$$\prod_j p(x_j|\tilde{y}=N)p(\tilde{y}=N)$$

Decision should be made based on whichever probability is higher

- Training consists of computing the probability distributions required for the above computation.



# Naive Bayes Classifier: heart disease example

| | | |
|---|---|---|
| $x_1$ | $p(x_1\|y=N)=\mathcal{N}(52.7,93.2)$ | |
| | $p(x_1\|y=Y)=\mathcal{N}(56.8,65.4)$ | |
| $x_2$ | $p(x_2=F\|y=N)=0.44$ | $p(x_2=M\|y=N)=0.56$ |
| | $p(x_2=F\|y=Y)=0.18$ | $p(x_2=M\|y=Y)=0.82$ |
| $x_3$ | $p(x_3=1\|y=N)=0.10$ | $p(x_3=2\|y=N)=0.23$ |
| | $p(x_3=1\|y=Y)=0.06$ | $p(x_3=2\|y=Y)=0.06$ |
| $x_3$ | $p(x_3=3\|y=N)=0.42$ | $p(x_3=4\|y=N)=0.24$ |
| | $p(x_3=3\|y=Y)=0.14$ | $p(x_3=3\|y=Y)=0.74$ |
| $x_4$ | $p(x_4\|y=N)=\mathcal{N}(128.0,251.4)$ | |
| | $p(x_4\|y=Y)=\mathcal{N}(135.1,365.5)$ | |
| $x_5$ | $p(x_5\|y=N)=\mathcal{N}(243.0,3111)$ | |
| | $p(x_5\|y=Y)=\mathcal{N}(249.0,2603)$ | |
| $y$ | $p(y=N)=0.58$ | |
| | $p(y=Y)=0.42$ | |

Normal (or Gaussian) distribution:

$$\mathcal{N}(\mu,\sigma^2)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Estimating the mean of a Gaussian distribution from data:

$$\mu_m=\frac{1}{N}\sum_i x_{mi}$$

Estimating the variance of a Gaussian distribution from data:

$$\sigma_m^2=\frac{N}{N-1}\sum_i (x_{mi}-\mu_m)^2$$



# Naive Bayes Classifier: heart disease example

| | | |
|---|---|---|
| $x_1$ | $p(x_1\|y=N)=\mathcal{N}(52.7,93.2)$ | |
| | $p(x_1\|y=Y)=\mathcal{N}(56.8,65.4)$ | |
| $x_2$ | $p(x_2=F\|y=N)=0.44$ | $p(x_2=M\|y=N)=0.56$ |
| | $p(x_2=F\|y=Y)=0.18$ | $p(x_2=M\|y=Y)=0.82$ |
| $x_3$ | $p(x_3=1\|y=N)=0.10$ | $p(x_3=2\|y=N)=0.23$ |
| | $p(x_3=1\|y=Y)=0.06$ | $p(x_3=2\|y=Y)=0.06$ |
| $x_3$ | $p(x_3=3\|y=N)=0.42$ | $p(x_3=4\|y=N)=0.24$ |
| | $p(x_3=3\|y=Y)=0.14$ | $p(x_3=3\|y=Y)=0.74$ |
| $x_4$ | $p(x_4\|y=N)=\mathcal{N}(128.0,251.4)$ | |
| | $p(x_4\|y=Y)=\mathcal{N}(135.1,365.5)$ | |
| $x_5$ | $p(x_5\|y=N)=\mathcal{N}(243.0,3111)$ | |
| | $p(x_5\|y=Y)=\mathcal{N}(249.0,2603)$ | |
| $y$ | $p(y=N)=0.58$ | |
| | $p(y=Y)=0.42$ | |

$$\mathcal{N}(\mu,\sigma^2)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$x_1=52, x_2=M,$
$x_3=3, x_4=172,$
$x_5=199$

Healthy
Heart Disease

## Heart disease problem results

| Hypothesis | Train result (% of 237 samples classified incorrectly) | Test results (% of 60 samples classified incorrectly) |
|---|---|---|
| Naive Bayes Classifier | 24.9 | 23.3 |

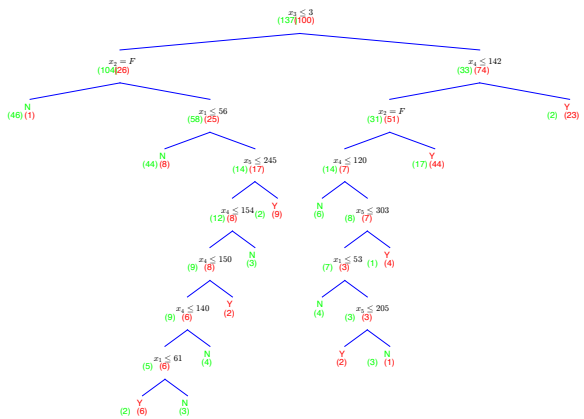


## Decision Tree

1. Divide training data into subsets in such a way that subsets become *more pure* based on their label information
   - Divide based on value of a single attribute
   - Pick the attribute and the split point such that entropy (disorder) of the data labels after the split is minimized
2. For each subset:
   - If the subsets is pure enough (in terms of data labels) make this a leaf node indicating that label
   - If not, then split this subset again – go back to step 1.



## Decision tree: heart disease example





## Heart disease problem results

| Hypothesis | Train result (% of 237 samples classified incorrectly) | Test results (% of 60 samples classified incorrectly) |
|---|---|---|
| Naive Bayes Classifier | 24.9 | 23.3 |
| Decision Tree (with some pruning) | 14.3 | 20.0 |

# A problem: Heart disease diagnosis (normalised)

| Sample | (Age-34)/43 | Sex (0-F, 1-M) | Chest pain type<br>0 – typical angina<br>0.33 – atypical angina<br>0.67 – non-anginal pain<br>1.0 – asymptomatic | Resting blood pressure (mm Hg-94)/98 | Cholesterol Level (mg/dL-126)/438 | Heart disease (0-N, 1-Y) |
|---|---|---|---|---|---|---|
| 1 | 0.4651 | 0 | 0.67 | 0.1633 | 0.2009 | 0 |
| 2 | 0.7907 | 1 | 1.00 | 0.5102 | 0.1530 | 1 |
| 3 | 0.6977 | 1 | 0.67 | 0.4694 | 0.4772 | 1 |
| 4 | 0.5581 | 0 | 0.00 | 0.5714 | 0.3584 | 0 |
| . . . | . . . | . . . | . . . | . . . | . . . | . . . |
| 297 | 0.4419 | 0 | 1.00 | 0.4490 | 0.2466 | 0 |

http://archive.ics.uci.edu/ml/datasets/Heart+Disease



# Linear model

- Decide on the feature space (the set of base functions that transform input into features)

$$y = \sum_j w_j f_j(\mathbf{x}) = \begin{bmatrix} w_1 & \dots & w_U \end{bmatrix} \begin{bmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_U(\mathbf{x}) \end{bmatrix} = \mathbf{w}^T \Phi(\mathbf{x}) \text{ , where } \Phi(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_U(\mathbf{x}) \end{bmatrix}$$

- Use the formula to compute the weighted sum of the components of input in the feature space

$$\mathbf{w} = \left(\mathbf{F}\mathbf{F}^T\right)^{-1} \mathbf{F}\tilde{\mathbf{y}}^T$$

$$\mathbf{F} = \begin{bmatrix} f_1(\mathbf{x}_1) & \dots & f_1(\mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ f_U(\mathbf{x}_1) & \dots & f_U(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} \Phi(\mathbf{x}_1) & \dots & \Phi(\mathbf{x}_N) \end{bmatrix} \qquad \tilde{\mathbf{y}} = \begin{bmatrix} \tilde{y}_1 & \dots & \tilde{y}_N \end{bmatrix}$$



# Linear model: heart disease example

- Linear features

Solving for w gives:

$$\mathbf{w}^T = \begin{bmatrix} 0.3129 & 0.3198 & 0.5827 & 0.4957 & 0.2184 & -0.6002 \end{bmatrix} \qquad \Phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ 1 \end{bmatrix}$$

- Quadratic features

Solving for w gives:

$$\mathbf{w}^T = \begin{bmatrix} 0.3888 & 0.2756 & -0.7567 & -0.1118 & 0.4146 & -0.2300 & -0.1394 & 0 & 1.1601 & 0.6180 & -0.2727 \end{bmatrix} \qquad \Phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ 1 \\ x_1x_1 \\ x_2x_2 \\ x_3x_3 \\ x_4x_4 \\ x_5x_5 \end{bmatrix}$$



# Optimisation

- Closed form solution is defined only for linear models...
- ...and even then it doesn't always work (matrix in the inversion not guaranteed to have an inverse).
- The parameters of a chosen hypothesis can be learned iteratively - starting with an initial guess and followed by a sequence of parameter updates.
- The updates should be such that some chosen evaluation measure of the performance of the model keeps improving (cost is minimised or fitness is maximised)
- Methods for computing the update
  - Random walk – change parameters randomly, keep the change if the resulting cost goes down
  - Steepest gradient descent – compute negative derivative (gradient) of the cost with respect to parameters, and use it as the update
  - Simulated annealing – random change, with initially high (and then gradually lower) probability of accepting a state that leads to higher cost
  - Genetic algorithm – evolution of a set of solutions using crossover and mutations of the state (*chromosome*)

# Heart disease problem results

| Hypothesis | | Train result (% of 237 samples classified incorrectly) | Test results (% of 60 samples classified incorrectly) |
|---|---|---|---|
| Naive Bayes Classifier | | 24.9 | 23.3 |
| Decision Tree (with some pruning) | | 14.3 | 20.0 |
| Linear model (linear in parameters) | Linear features | 21.9 | 23.3 |
| | Quadratic features | 21.5 | 20.0 |



# Linear model with maximum margin (SVM)

- Support Vector Machine optimisation and output computation relies on the relationship of points in the features space: $\Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$
- Choice of kernel function determines what feature space is used

- **Linear features** – use a support vector machine with the linear kernel function

$$\Phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ 1 \end{bmatrix} \quad \text{and so} \quad \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j$$

  - Need to choose SVM's *C* parameter, which penalises for errors in training – large *C* leads to less emphasis on consistency during training

- Non-linear features –use a support vector machine with a non-linear kernel function, such as **Radial Basis Function (RBF)**:

$\Phi(\mathbf{x})$ not computable, but $\Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j) = e^{-\gamma(\mathbf{x}_i-\mathbf{x}_j)^T(\mathbf{x}_i-\mathbf{x}_j)}$

  - Need to chose SVM's *C* parameter as well as the *gamma* parameter for the kernel function

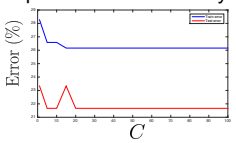


# SVM: heart disease example

- **Linear features** – use a support vector machine with the linear kernel function

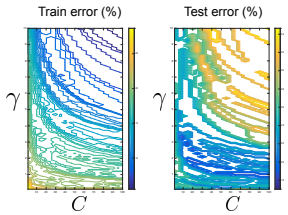
$$\Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j$$

  - Need to choose SVM's *C* parameter, which penalises for errors in training – large *C* leads to less emphasis on consistency during training




- Non-linear features –use a support vector machine with non-linear kernel function, such as **RBF**: $\Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j) = e^{-\gamma(\mathbf{x}_i-\mathbf{x}_j)^T(\mathbf{x}_i-\mathbf{x}_j)}$

  - Need to chose SVM's C parameter as well as the gamma parameter for the kernel function






# Heart disease problem results

| Hypothesis | | Train result (% of 237 samples classified incorrectly) | Test results (% of 60 samples classified incorrectly) |
|---|---|---|---|
| Naive Bayes Classifier | | 24.9 | 23.3 |
| Decision Tree (with some pruning) | | 14.3 | 20.0 |
| Linear model (linear in parameters) | Linear features | 21.9 | 23.3 |
| | Quadratic features | 21.5 | 20.0 |
| | Linear kernel, C=20 | 26.2 | 21.7 |
| | RBF kernel, C=45, gamma=1.1 | 18.1 | 16.7 |

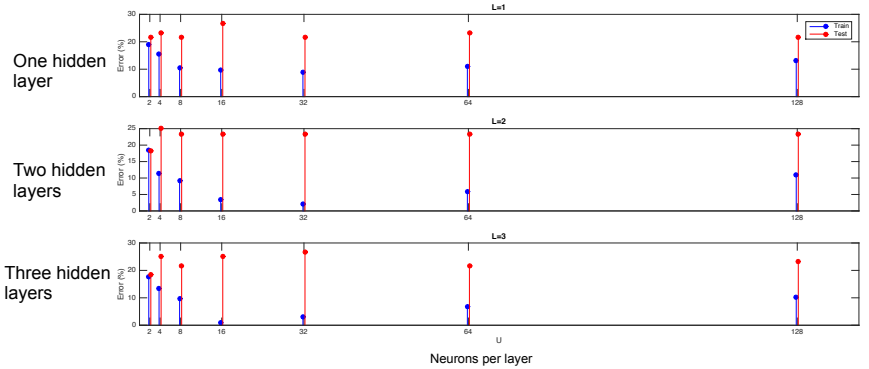
# Multilayer Perceptron (MLP)

- Create an artificial neural network
  - Decide on the number of hidden layers – more layers give a more powerful model, but harder to train
  - Decide on the number of neurons in each layer – more neurons give a more powerful model, but harder to train
  - Decide on the activation function (or combination of activation functions) in the hidden layer
  - Pick initial parameter values (network weights and biases) – usually a random number close to zero
- Train the network for a chosen number of iterations using backpropagation to derive steepest gradient update for the weights and biases that minimise a chosen cost:
  - Sum of squared errors – good for regression, or two-class classification
  - Cross-entropy – good for classification
  - Softmax with cross-entropy – good for classification with more than two classes, where output is a probability distribution of the class given the input



# MLP: heart disease example

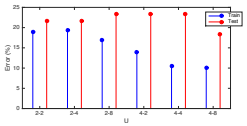
- Tinkering with different network architectures to get a sense of which one might generalise well
  - Same number of neurons per layer, trying different number of layers and different number of neurons per layer



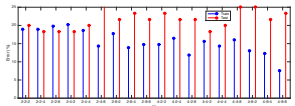



# MLP: heart disease example

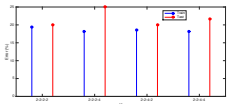
- Tinkering with different network architectures to get a sense of which one might generalise well
  - Same number of neurons per layer, trying different number of layers and different number of neurons per layer
  - 2-hidden-layer architectures with different combinations of neurons



  - 3-hidden-layer architectures with different combinations of neurons



  - 4-hidden-layer architectures, with 2 neurons in the first two layers and 4 or 2 neurons in the other layers





# Heart disease problem results

| Hypothesis | | Train result (% of 237 samples classified incorrectly) | Test results (% of 60 samples classified incorrectly) |
|---|---|---|---|
| Naive Bayes Classifier | | 24.9 | 23.3 |
| Decision Tree (with some pruning) | | 14.3 | 20.0 |
| Linear model (linear in parameters) | Linear features | 21.9 | 23.3 |
| | Quadratic features | 21.5 | 20.0 |
| | Linear kernel, C=20 | 26.2 | 21.7 |
| | RBF kernel, C=45, gamma=1.1 | 18.1 | 16.7 |
| MLP, 2 hidden layers, 2 neurons per hidden layer (best of 100 runs) | | 21.5 | 18.3 |