Enter the information in the table below, which will be entered into a database to produce the front page automatically.

Time allowed:	2 hours 30 minutes
No of qu to be answered	4
Rest of rubric	Answer 4 questions
Any special requirements for resit candidates	No
Is an appendix included?	Yes
Examiners	Ioannis Patras and Chrisantha Fernando

Question 1

(a) Define the conditional probability $p(A|B)$. You may want to use a diagram/sketch.

[3 Marks]

Answer

The conditional probability p(A|B) is the probability that the event A occurs, given that the event B has occurred. It is given by the formula $p(A|B) = P(A,B) / P(B)$

(b) Give the law of total probabilities, that is express $p(A)$ using a set of events $B_1, B_2, \ldots, B_N$ and the corresponding conditional (or joint) probabilities. You may want to use a diagram/sketch. What are the conditions that need to hold?

[3 Marks]

Answer

$p(A) = \sum_{i=1\ldots N} p(A|B_i)p(B_i) = \sum_{i=1\ldots N} p(A,B_i)$ In order for this formula to hold true, the events Bi need to be mutually exclusive and their union needs to be the event space. That is, $B_i \cap B_j = \varnothing$ and $B_1 \cup B_2 \cup \ldots \cup B_N = \Omega$

(c) Show that the expected value of the sum of two independent random variables X and Y is equal to the sum of the expected values of X and Y. That is, show that $E\{X+Y\} = E\{X\} + E\{Y\}$. You may show it either for continuous or discrete variables.

(Hint: You need to work with the joint probability $P(X,Y)$. Recall that the expected value of a Random Variable X is given by $E\{X\} = \int_X X p_X(X) dx$).

[4 Marks]

Answer

$$
\begin{aligned}
&E\{X+Y\} \\
&= \int_{X,Y} (X+Y)\,p(X,Y)\,dxdy = \int_{X,Y} (X+Y)\,p_{XY}(X,Y)\,dxdy \\
&= \int_{X,Y} Xp_{XY}(X,Y)\,dxdy + \int_{X,Y} Yp_{XY}(X,Y)\,dxdy \\
&= \int_{X,Y} Xp_X(X)\,p_Y(Y)\,dxdy + \int_{X,Y} Yp_X(X)\,p_Y(Y)\,dxdy \\
&= \int_{X,Y} Xp_X(X)\,p_Y(Y)\,dxdy + \int_{X,Y} Yp_X(X)\,p_Y(Y)\,dxdy \\
&= \int_X \int_Y Xp_X(X)\,p_Y(Y)\,dxdy + \int_X \int_Y Yp_X(X)\,p_Y(Y)\,dxdy \\
&= \int_X Xp_X(X)\,dx \int_Y p_Y(Y)\,dy + \int_X p_X(X)\,dx \int_Y Yp_Y(Y)\,dy \\
&= \int_X Xp_X(X)\,dx + \int_Y Yp_Y(Y)\,dy = E\{X\} + E\{Y\}
\end{aligned}
$$

(d) If two random variables X and Y are independent then, the expected value of the their product is the product of their expected values that is, $E\{XY\} = E\{X\}E\{Y\}$. Use this to show that the variance of the sum of two independent random variables X and Y is equal to the sum of their variances, that is $Var(X+Y) = Var\{X\} + Var\{Y\}$.

Recall that $Var(X) = E\{(X - E\{X\})^2\} = E\{X^2\} - E\{X\}^2$

[5 Marks]

Answer

$$
\begin{aligned}
&Var\{X+Y\} = E\{(X+Y)^2\} - E\{X+Y\}^2 \\
&= E\{X^2 + 2XY + Y^2\} - (E\{X\} + E\{Y\})^2 \\
&= E\{X^2\} + 2E\{XY\} + E\{Y^2\} - E\{X\}^2 - 2E\{X\}E\{Y\} - E\{Y\}^2 \\
&= E\{X^2\} + 2E\{X\}E\{Y\} + E\{Y^2\} - E\{X\}^2 - 2E\{X\}E\{Y\} - E\{Y\}^2 \\
&= E\{X^2\} - E\{X\}^2 + E\{Y^2\} - E\{Y\}^2 \\
&= Var\{X\} + Var\{Y\}
\end{aligned}
$$

(e) An IT worker works from home 1 day a week. When she works from home she answers 80% of the emails within an hour, 15% of the emails within 2 hours and 5% of the emails within the day. When she is at the office, she answers 50% of the emails within an hour, 40% within 2 hours and 10% within the day. If you send her an email, what is the probability that she will answer within 2 hours? Given that she hasn't replied to your email within the first 2 hours, what is the probability that she is working from home?

[10 Marks]

[Total 25 Marks]

Answer

The joint probability table is given by:

0.8*0.5	0.8*0.4	0.8*0.1	=	0.4	0.32	0.08
0.2*0.8	0.2*0.15	0.2*0.05		0.16	0.03	0.01

The probability that she will answer within two hours is the sum of the values of the first two columns that equals 0.91

Let D denote the event that she will answer after two hours. P(D) = 0.09 (sum of last column)

Let H denote the event that she works at home. P(H) = 0.2 (sum of second row)

P(D,H) = 0.01 is the probability that she works from home AND will answer after two hours.

$P(H|D) = P(D,H) / P(D) = 0.01 / 0.09 = 0.111 < p(H) = 0.2$

-

Question 2

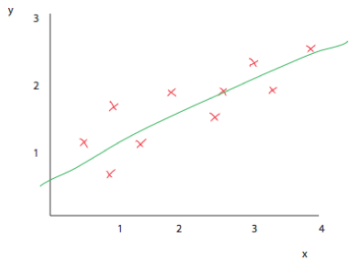
(a) Complete this table describing the form of the model to be learned, the cost function, and the possible methods for minimizing this cost function, for linear regression and logistic regression.

	Form of Model	**Cost Function**	**Gradient method to find minimum of cost function**
Linear Regression for a vector x			
Logistic Regression			

(10 Marks)

Answer

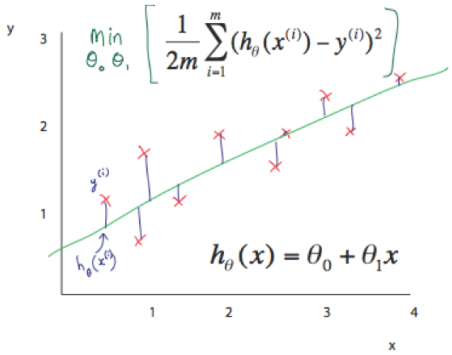
Linear Regression Form of Model =



We want a hypothesis **h** that maps x to y

$$h_\theta(x) = \theta_0 + \theta_1 x$$

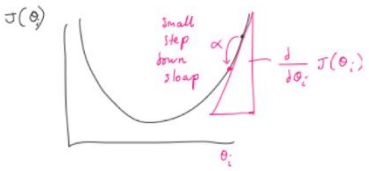
Linear Regression Cost Function =



The cost function $J(\theta_0, \theta_1)$ is the Sum of Squares.

Turn over until you reach the "End of Paper" line

- Use gradient descent algorithm...



repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

}

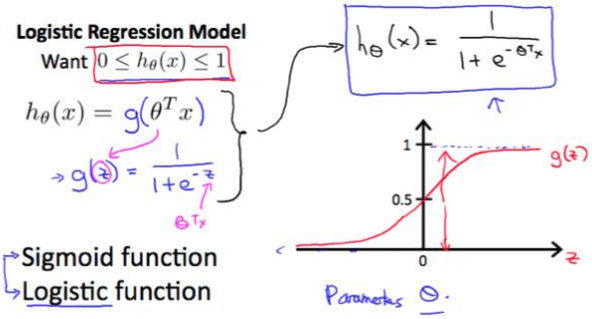
repeat until convergence {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$

}

Logistic Regression Model =



Logistic Regression Cost Function =

It's the cross-entropy function

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \text{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m} [\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

Method of minimizing it is again

We need to minimize this w.r.t. θ

Repeat

$$\theta_j := \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

} (simultaneously update all)

Use gradient descent again to change the parameters along their gradient w.r.t. the cost function.

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

}

(b) Consider a perceptron neuron (i.e. one with a threshold activation function +1 if input > 0 and 0 if input < 0) with 2 inputs and 1 output. If the two weights are $w_1 = 1$ and $w_2 = 1$, and the bias is $b = -1.5$, then what is the output for input (0,0)? What about for inputs (1,0), (0,1) and (1,1)? **(4 Marks)**

Answer

(0,0): 1 * -1.5 + 0 *1 + 0 * 1 = -1.5 → 0

(0,1): 1 * -1.5 + 0 * 1 + 1 * 1 = -0.5 → 0

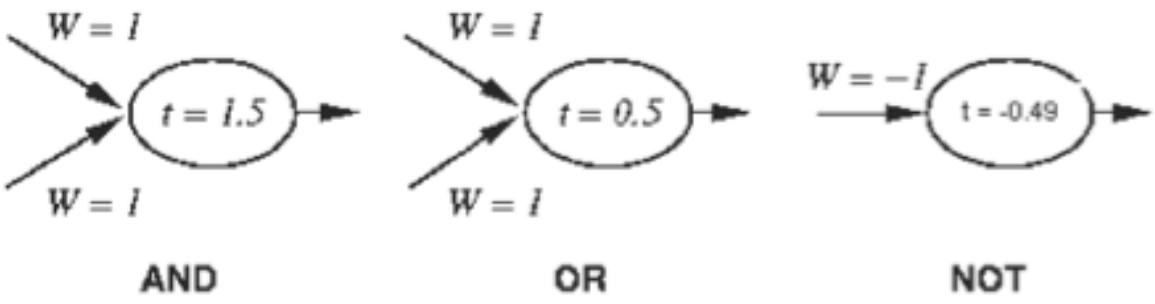
(1,0): 1 * -1.5 + 1 * 1 + 0 * 1 = -0.5 → 0

(1,1): 1 -1.5 + 1* 1 + 1 * 1 = 0.5 → 1

(c) Work out the logistic regression units for calculating NOT, AND and OR of their inputs and demonstrate that they classify the input vectors properly.

(6 Marks)

Answer



Turn over until you reach the "End of Paper" line

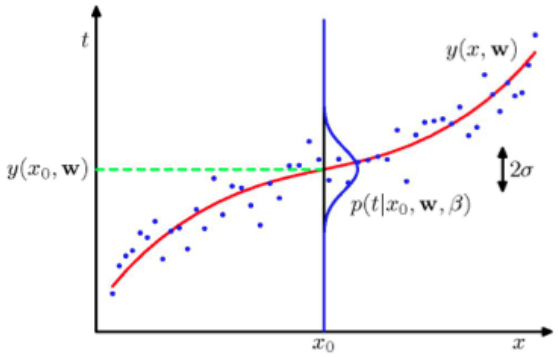
(d) What are the underlying Bayesian assumptions behind the use of the Least Mean Squares cost function, i.e. what assumptions justify the use of this cost function in the first place? What additional assumption is made by the use of regularization? **(5 Marks)**

[Total 25 Marks]

Answer

- What is the Bayesian basis for linear regression with LMS and with regularization?

Basically this means, what are our assumptions about the FORM of the model we are fitting?



LMS assumes a Gaussian noise Distribution.

Regularization assumes a prior over the weights, favoring low weights.

Question 3.

(a) Describe your favourite example of a real-world problem that has been solved by a feedforward network using backpropogation. Give a reference to the example. [**5 Marks]**

Answer Mark Guidelines: Each student was asked to prepare such an example during the coursework and told that this might be asked for in the exam. Each answer is expected to be unique, a result of the students individual and independent research. Originality of examples will be awarded high marks, as will an understanding of the problems in the design of each example.

(b) Write the pseudocode for the backpropogation algorithm describing the initialization, training phase consisting of the forward pass and the backwards pass, i.e. the method of passing error backwards to the hidden neurons and adjusting their weights by gradient descent on this backpropogated error. Make sure you make it clear what the form of the model, the cost function, and the method of minimizing the cost function is. **[9 Marks]**

Answer

The Multi-Layer Perceptron Algorithm

- **Initialisation**
  - initialise all weights to small (positive and negative) random values

- **Training**
  - repeat:
    - for each input vector:

      Forwards phase:

      - compute the activation of each neuron j in the hidden layer(s) using:

$$h_j = \sum_i x_i v_{ij} \tag{3.4}$$

$$a_j = g(h_j) = \frac{1}{1 + \exp(-\beta h_j)} \tag{3.5}$$

      - work through the network until you get to the output layers, which have activations:

$$h_k = \sum_j a_j w_{jk} \tag{3.6}$$

$$y_k = g(h_k) = \frac{1}{1 + \exp(-\beta h_k)} \tag{3.7}$$

      Backwards phase:

      - compute the error at the output using:

$$\delta_{ok} = (t_k - y_k)\, y_k (1 - y_k) \tag{3.8}$$

      - compute the error in the hidden layer(s) using:

$$\delta_{hj} = a_j (1 - a_j) \sum_k w_{jk} \delta_{ok} \tag{3.9}$$

      - update the output layer weights using:

$$w_{jk} \leftarrow w_{jk} + \eta \delta_{ok} a_j^{\text{hidden}} \tag{3.10}$$

      - update the hidden layer weights using:

$$v_{ij} \leftarrow v_{ij} + \eta \delta_{hj} x_i \tag{3.11}$$

    - randomise the order of the input vectors so that you don't train in exactly the same order each iteration
  - until learning stops (see Section 3.3.6)
- **Recall**
  - use the Forwards phase in the training section above

Turn over until you reach the "End of Paper" line

Minimize a similar cost function as for logistic regression…

Cost function

Logistic regression:

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1 - h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

Neural network:

$$h_\Theta(x) \in \mathbb{R}^K \quad (h_\Theta(x))_i = i^{th} \text{ output}$$

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\Theta(x^{(i)}))_k + (1-y_k^{(i)}) \log(1 - (h_\Theta(x^{(i)}))_k)\right]$$

$$+\frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}} (\Theta_{ji}^{(l)})^2$$

Summed over m training examples

(c) **Briefly** define the following terms giving a clear example in each case. i. Local minima, ii. Linear discriminability, iii. The curse of dimensionality, iv. adaptive basis function expansion, v. overfitting, vi. underfitting. **[6 Marks]**

Answers

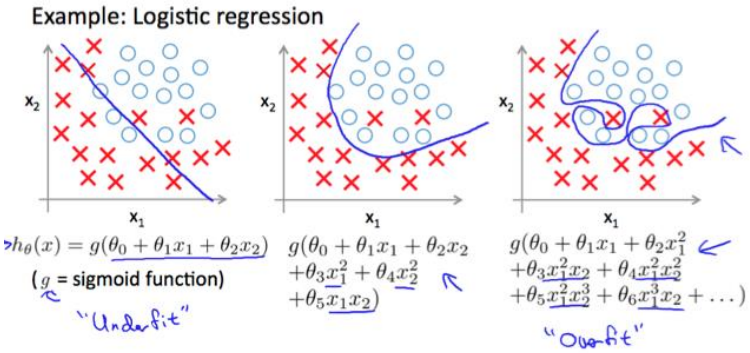
i. Local minima: A point in a function where the first derivative is zero but where the function is not at a global minimum.

ii. Linear discriminability : A decision boundary can be drawn through the space to separate data points into their respective classes without error, e.g. AND/OR but not XOR in a 2D 2 class example.

iii. The curse of dimensionality: As parameter dimension D grows the number of training examples required to train the system grows exponentially with D. This means some kind of function approximation assumptions such as smoothness etc… are required, and indeed these assumptions apply in the FFNN.

iv. Adaptive basis function expansion. What the FFNN does in its hidden layers, obviating the need for hand designed basis function expansion which would suffer from a dimensionality explosion as well.

v. Overfitting and underfitting



(d) You make a neural network to control a robot. It takes the sensory inputs of the robot and outputs the motor commands. You want the robot to learn to walk, when initially it can't do anything. What would you need to use a FFNN and backprop to train such a robot? If you didn't have this, how else might you train the neural network paramaters to allow the robot to walk? **[5 Marks]**

[Total 25 Marks]

Answer

This answer should show a general understanding of the kinds of machine learning problem, and need not show detailed knowledge of reinforcement learning or evolutionary computation, for example**:** To use a supervised method, you would need to give examples of the actual motor commands required to walk, e.g. by getting data from real human walking and using this to train the network. If this was not available then the problem could be dealt with as a reinforcement learning problem or an optimization problem. For example, you might use a black-box optimization algorithm such as a genetic algorithm to learn the parameters of the network. You would have to design a suitable representation of the controller that could be mutated and crossed over. You would also have to design a fitness function for the controller which for example would be the distance walked, or the time before falling over.

Question 4

(a) Describe the difference between supervised and unsupervised learning. Give one example of a supervised learning problem and an example of an unsupervised learning problem.

[4 marks]

Answer:

In supervised learning one is give a set of data and the ground truth targets. For example, in the case of classification, one is given a set of data and for each datum one is given the class to which it belongs.

In the case of unsupervised learning the targets are unknown. The tasks there is to extract some meaningful information out of the data, organise it in a useful manner or extract characteristics from it. An example is clustering where the goal is to organise a set of data into clusters.

(b) Describe in detail the steps of the K-means algorithm. Make sure that you define the dimensionality of all the variables that you use.

[8 marks]

Answer:

Given a dataset with examples $X = (x_1, \ldots, x_N)$ with $x_i \in \mathbb{R}^m$, K-Means will output K centres $\mu_1, \ldots, \mu_K$, $\mu_k \in \mathbb{R}^m$ and assign each data example to the nearest centre. The nearest centre is the one with the smallest Euclidean distance.

Step 0: Initialise the cluster centres, for example sample from a uniform distribution within the extremes of the data.

$$\forall k \in \{1, \ldots, K\} : \mu_k \sim Uniform(\min(X), \max(X))$$

Step 1: Class assignment stage. Assign each data point to the nearest cluster.

$$\forall i \in \{1, \ldots, N\} : u_i \leftarrow \arg\min_k \| x_i - \mu_k \|$$

Step 2: Re-estimate the cluster centres

$$\forall k \in \{1, \ldots, K\} : \mu_k \leftarrow \frac{1}{|\mathsf{C}_k|} \sum_{i \in \mathsf{C}_k} x_i \text{ , where } \mathsf{C}_k = \{i \in \{1, \ldots, N\} \mid u_i = k\}$$

Step 3: Until there is no data that changes cluster go back to step 1.

(c) The probability density function of the Gaussian distribution is given by $p(x) = \frac{1}{(2\pi)^{m/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$. Explain what each symbol represents and give its dimensionality.

[4 marks]

Answer:

m is the dimensionality of the data vectors (scalar)

x is a data vector (dimensionality m)

μ is the mean vector (dimensionality m)

Σ is the covariance matrix

(d) What are the differences between the Expectation Maximisation algorithm that estimates the components of a Mixture of Gaussians (EM-GMM) and the K-Means algorithm? Give the modification of the assignment and the estimation steps of the K-means algorithm so that you arrive at the EM algorithm that estimates the mixture of K Gaussians.

[4 marks]

Answer

The main differences between K Means and the EM-GMM algorithm are:

a. The data samples are assigned softly to each cluster

b. The data in each group/cluster is described by both the mean and a covariance matrix instead of only the mean.

Given a dataset with examples $X = (x_1, \ldots, x_N)$ with $x_i \in \mathbb{R}^m$, EM_GMM will output K mean vectors $\mu_1, \ldots, \mu_K$, $\mu_k \in \mathbb{R}^m$, K covariance matrices $\Sigma_1, \ldots, \Sigma_K$, $\mu_k \in \mathbb{R}^m$ and K mixing coefficient $\pi_1, \ldots, \pi_K$, $\mu_k \in \mathbb{R}^m$

Step 1: Class assignment stage. Assign softly each data point to the nearest cluster.

$$\forall i \in \{1, \ldots, N\}, \forall k \in \{1, \ldots, K\} : q_{ik} = p(k \mid x_i) = p(x_i \mid k)\pi_k \sum_{k'} p(x_i \mid k')\pi_{k'}$$

Step 2: Re-estimate the parameters for each Gaussian

$$\forall k \in \{1, \ldots, K\} : \pi_k \leftarrow \sum_i q_{ik} ,$$

$$\forall k \in \{1,\ldots,K\} : \mu_k \leftarrow \frac{1}{\pi_k} \sum_i q_{ik} x_i$$

$$\forall k \in \{1,\ldots,K\} : \Sigma_k \leftarrow \frac{1}{\pi_k} \sum_i q_{ik} \left(x_i - \mu_k\right)\left(x_i - \mu_k\right)^T$$

(e) Both the K-Means algorithm and the EM-GMM algorithm derive their solutions by attempting to optimise certain quantities. Describe one practical problem that either of these algorithm faces and give a way to deal to with the problem.

[5 marks]

[Total 25 Marks]

Answer:

Both algorithms reach local optima. A way to deal with the problem is to run the algorithm several times with different re-initialisation.

EM-GMM faces the singularity problem where one or more covariance matrices become singular. A way to deal with this is to regularise the covariance matrix, for example by adding a diagonal matrix with small values along the diagonal.

End of Paper