UNIVERSITY OF EDINBURGH

FACULTY OF SCIENCE AND ENGINEERING

LFD1

Date: ?? April 2001 DRAFT **Time: ??:??-??:??**

DRAFT

This will describe the degree (MSc/AI4,etc.)

Examiners: Name of external examiner (External)
Name of exam board chair (Chair)

INSTRUCTIONS TO CANDIDATES

Answer TWO questions.

If you attempt three questions, cross out one answer; if you do not, then the examiners will cross out the last one you answered.

Each complete question carries equal weight and is marked out of 100. The parts of a question may not all be worth the same amount; the marks at the side of the questions indicate how these will normally be apportioned.

Write as legibly as possible.

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

LFD1

1. Consider a set of N-dimensional data $\boldsymbol{x}^\mu, \mu = 1, \ldots, P$. Each datapoint $\boldsymbol{x}^\mu$ has a corresponding class label, c^μ.

   (a) Explain how the K nearest neighbour method (KNN) can be used to classify an unlabelled test data point $\boldsymbol{x}^*$.

   For the single nearest neighbour method ($K = 1$) describe, with the aid of a diagram, the geometry of the decision boundary. [30%]

   Discuss a situation in which the nearest neighbour ($K = 1$) method will perform poorly. [15%]

   (b) Describe how to reduce the above training data to M-dimensional data using Principal Components Analysis (PCA). Include a full description of the PCA method. [20%]

   (c) Describe how to combine PCA dimension reduction and the nearest neighbour classification method, and explain why you think this may be an appropriate thing to do. [10%]

   (d) You decide to perform PCA dimension reduction on images, each of which is 5000×5000 pixels. You have 10000 such images. What is the dimension of the covariance matrix of the data? [10%]

   (e) Consider two vectors $\boldsymbol{x}^a$ and $\boldsymbol{x}^b$ and their corresponding PCA approximations $\boldsymbol{m} + \sum_{i=1}^M a_i \boldsymbol{e}^i$ and $\boldsymbol{m} + \sum_{i=1}^M b_i \boldsymbol{e}^i$, where the eigenvectors $\boldsymbol{e}^i, i = 1, \ldots M$ are mutually orthogonal and have unit length and $\boldsymbol{m}$ is the mean of the data. The eigenvector $\boldsymbol{e}^i$ has corresponding eigenvalue λ^i.

   Let $\mathbf{S}$ be the covariance matrix of the data. The Mahalanobis distance between $\boldsymbol{x}^a$ and $\boldsymbol{x}^b$ is defined as

   $$\left(\boldsymbol{x}^a - \boldsymbol{x}^b\right)^T \mathbf{S}^{-1} \left(\boldsymbol{x}^a - \boldsymbol{x}^b\right).$$

   Explain how to approximate this distance using the M-dimensional PCA approximations, as described above. [15%]

Question 2 is on the next page.

2. Consider data $\{(\boldsymbol{x}^i, c^i), i = 1, \ldots, n_1 + n_2\}$. This data consists of inputs $\boldsymbol{x}^i$, each with a corresponding class label, $c^i = 1$ or $c^i = 2$. There are n_1 training examples from class 1, and n_2 training examples from class 2.

   (a) Describe the general procedure of fitting class conditional distributions $p(\boldsymbol{x}|c = j, \theta^j)$ to the data, where θ^j are the parameters of the distribution fitted to the data from class j. How can we use these distributions to form a classifier $p(c = j|\boldsymbol{x})$? [30%]

   (b) For training data with multi-dimensional inputs, $dim(\boldsymbol{x}) = m$, define the Naive Bayes classification method, and discuss the strengths and weaknesses of this approach. Discuss how to train the classifier in the case of discrete inputs, and contrast this with the case of continuous inputs. [25%]

   For the case of m dimensional binary input data, discuss the relationship between the decision boundary of the Naive Bayes method and the decision boundary from logistic regression. [20%]

   (c) For one dimensional input data, show that the value of x that represents the decision boundary of the classifier satisfies the expression

$$\log \frac{p(x|c = 2)}{p(x|c = 1)} = \log \frac{p(c = 1)}{p(c = 2)}.$$

   Write down the Maximum Likelihood (ML) estimates of $p(c = 1)$ and $p(c = 2)$. [10%]

   In one dimension, $dim(\boldsymbol{x}) = 1$, the Gaussian distribution is defined as

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

   You decide to fit a Gaussian to each class and use the ML estimates of the means $\hat{\mu}_1$ and $\hat{\mu}_2$. From the data, you find that the ML estimates of σ_1^2 and σ_2^2 are equal, that is, $\hat{\sigma}_1^2 = \hat{\sigma}_2^2$. Write down the explicit x value that defines the decision boundary. [10%]

   Point out any potential numerical difficulties in directly comparing the values $p(c = 1|x)$ and $p(c = 2|x)$ and explain how you might overcome this. [5%]

Question 3 is on the next page.

3. Consider iid (independently and identically distributed) training data $D = \{(\boldsymbol{x}^\mu, c^\mu), \mu = 1, \ldots, P\}$, $c^\mu \in \{0, 1\}$.

   (a) Explain the following terms : training error, test error, validation error. Explain the concept of overfitting, and how this can be avoided. [15%]

   (b) Explain what is meant by the term "dimension reduction" of input data $\mathbf{x}$. Explain what is meant by linear dimension reduction, and explain how non-linear dimension reduction can be performed. [15%]

   Give an example of a 2 dimensional dataset for which linear dimension reduction is inappropriate. [10%]

   When using PCA for dimension reduction, how can we gain some insight into the appropriate number of dimensions, based on the eigenvalues of the covariance matrix? [10%]

   Briefly describe a general approach to finding the optimal number of dimensions for any dimension reduction method. [10%]

   (c) Define the gradient ascent learning algorithm. With the aid of a diagram, draw an example where a gradient ascent optimisation routine would fail to find the global optimum. [15%]

   (d) Define logistic regression, and comment on the decision boundary. With the aid of a diagram, draw an example dataset that cannot be correctly classified using logistic regression. Would it be appropriate to use PCA to first reduce the dimension of the data, before using logistic regression? [25%]

LFD1

Brief notes on answers:

1. (a) For the nearest neighbour method $K = 1$, we search through the dataset to find the training datapoint $\boldsymbol{x}^n$ that is closest (using the Euclidean distance) to $\boldsymbol{x}^*$. We assign the label of $\boldsymbol{x}^*$ to be c^n. For $K > 1$, we find the K nearest neighbours of $\boldsymbol{x}^*$, and get their associated class labels. The majority class of these neighbours is taken as the class of $\boldsymbol{x}^*$.

   Picture with a few data points. The decision boundary is a Voronoi tesselation. If there is an irrelevant input, this will make things difficult. This could be depicted in two dimensions as a stretched case in which the irrelevant dimension dominates the distances between points. Alternatively, class distributions which are highly non-isotropic can give difficulty. This could be drawn as two very thin pancakes which are separated along the vertical axis by a small amount. Any similar or sensible answers are acceptable.

   (b) Find the sample mean and covariance matrix of the data. Then calculate the M largest eigenvalues of the covariance matrix, and their corresponding eigenvectors, $\boldsymbol{e}^i, i = 1, \ldots, 20$. The sample mean $\boldsymbol{m}$ is given by

$$\boldsymbol{m} = \frac{1}{P} \sum_{\mu} \boldsymbol{x}^{\mu}$$

   The covariance matrix is defined as

$$\mathbf{S} = \frac{1}{P} \sum_{\mu} \boldsymbol{x}^{\mu} (\boldsymbol{x}^{\mu})^T - \boldsymbol{m}\boldsymbol{m}^T$$

   If they define the biased or unbiased version of the covariance, either is fine.

   The lower dimensional data is then given by the projection, $y_i^{\mu} = \left(\boldsymbol{x}^{\mu} - \boldsymbol{m}\right)^T \boldsymbol{e}^i, i = 1, \ldots 20, \mu = 1, \ldots, P$.

   (c) We can use PCA to first reduce the dimension of all the inputs, regardless of the class label. This then defines a linear projection, which we will use to reduce the dimension of any test point $\boldsymbol{x}^*$. We can then use KNN on the reduced vectors, comparing their distances, rather than using the full dimensional vectors. This may be appropriate in retaining only those directions in space in which large variability occurs, and can result in a more robust classifier. In addition, once PCA has been performed, the distance comparisons in the lower dimensional space will be much faster than in the high dimensional space. Storage requirements are also reduced.

   (d) Covariance matrix is 25000000×25000000 dimensional.

   (e) Using the approximations, we have

$$(\boldsymbol{x}^a - \boldsymbol{x}^b)^T \mathbf{S}^{-1} (\boldsymbol{x}^a - \boldsymbol{x}^b) \approx (\sum_i a_i \boldsymbol{e}^i - \sum_i b_i \boldsymbol{e}^i)^T \mathbf{S}^{-1} (\sum_j a_j \boldsymbol{e}^j - \sum_i b_j \boldsymbol{e}^j)$$

   Due to the orthonormality of the eigenvectors, this is $\sum_i a_i^2/\lambda_i - 2a_i b_i/\lambda_i + b_i^2/\lambda_i = (\mathbf{a} - \mathbf{b})^T \mathbf{D}^{-1} (\mathbf{a} - \mathbf{b})$ where $\mathbf{D}$ is a diagonal matrix containing the eigenvalues.

2. (a) We can use the likelihood to fit each distribution. Assuming iid data, this is given by

$$\prod_{\boldsymbol{x} \in class(j)} p(\boldsymbol{x}|c=j, \theta^j)$$

Taking the log is numerically convenient. We can then find the parameters θ^j by numerically maximimising this function. Once trained, we can form a classifier using Bayes' rule:

$$p(c=j|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|c=j)p(c=j)}{p(\boldsymbol{x}|c=1)p(c=1) + p(\boldsymbol{x}|c=2)p(c=2)}$$

The ML estimates of the prior class probabilities are $p(c=1) = n_1/(n_1+n_2)$ and $p(c=1) = n_2/(n_1+n_2)$.

(b) In Naive Bayes, the inputs are assumed to be independent given the class:

$$p(\boldsymbol{x}|c=1) = \prod_{j=1}^{m} p(x_j|c=1).$$

For discrete data, the parameters $p(x_j|c=1)$ are estimated from the data by simply counting the fraction of times that x_j is in a particular state, given that $c=1$. Similarly, the prior $p(c=1)$ is given by the relative number of occurrences of class 1 compared to class 2 in the training data. Training is therefore very fast.

For continuous inputs, we can write down the log likelihood as a sum of log likelihoods for each input (due to the conditional independence assumption). In general, we would need to maximise each term then numerically. For the Gaussian, however, we can fit simply the mean and variance of each term by calculating the mean and variance of each input for a given class.

Classification is then given by comparing

$$\log \frac{p(c=1|\boldsymbol{x})}{p(c=2|\boldsymbol{x})} > 0$$

Or

$$\sum_j \log \frac{p(x_j|c=1)}{p(x_j|c=2)} + \log \frac{p(c=1)}{p(c=2)} > 0$$

A serious weakness of Naive Bayes is the very strong conditional independence assumption.

For binary data, the above rule shows that the decision boundary will be linear. This is the same as for logistic regression, although the boundaries are not necessarily the same.

(c) The decision boundary is given when $p(c=1|x) = 1/2$. Using this gives

$$\frac{1}{2} = \frac{p(x|1)p(c=1)}{p(x|1)p(c=1) + p(x|2)p(c=2)}$$

Rearranging this expression and taking the logarithm gives the desired result.

Using the Gaussians in the expression for the decision boundary, we have

$$-\frac{1}{2\sigma_1^2}\left(\mu_1^2 - \mu_2^2 + 2x\left(\mu_1 - \mu_2\right)\right) = \log\frac{p(c=1)}{p(c=2)}$$

Rearranging this gives

$$x = \frac{\sigma_1^2}{\mu_2 - \mu_1}\left(\log\frac{p(c=1)}{p(c=2)} + \frac{1}{2}\left(\mu_1^2 - \mu_2^2\right)\right)$$

$p(c=1) = n_1/(n_1+n_2)$, $p(c=2) = n_2/(n_1+n_2)$
It is numerically difficult to calculate directly $p(c=1|x)$ since this involves exponentiating a quantity. We will run into over/underflow problems if the variance is small. (This problem is compounded in higher dimensions). Taking the logarithm makes for a numerically feasible approach since to find the decision boundary we can simply compare the log probabilities.

3. (a) Training error : the error that is made by a model on the training data. Test error : the error made on a set of data distinct from the training data. This gives an independent estimate of the prediction error of the model. Validation error : in setting parameters such as regularisation parameters, we need to use another set of data to test how well each model, with a particular regularisation parameter performs in terms of prediction. Overfitting is the problem of training a too complex model such that it fits the data to have very low training error. However, the model will not necessarily perform well on the test set. This situation can be avoided by using a regulariser, and setting this to an appropriate value using a validation set.

   (b) Dimension reduction replaces each $\mathbf{x}$ in the data with a lower dimensional vector $\mathbf{y}$. This mapping is often chosen according to some loss criterion such as to minimise the reconstruction error. Alternatively, class specific dimension reduction, such as Fisher's Linear Discriminant reduce the dimension such that the differences between two data classes are maximised. Linear dimension reduction is performed by $\mathbf{y} = W\mathbf{x}$ for some non-square matrix W. Autoencoders with more than one hidden layer can be used for non-linear dimension reduction.

   Any picture with a dataset that does not lie on a straight line is fine. Eg. data on a circle, or on a wiggly line.

   In PCA, the eigenvalue spectrum tells us how many dimensions are appropriate, since the squared reconstruction error is the sum of the square eigenvalues from the eigenvectors not included in the reduction.

   We can use a validation set in general to find the optimal number of dimensions. That is, split the training data into a training and a validation set. The performance of a model based on a certain reduced dimensionality is evaluated on the validation set. That dimension which produces the best results on the validation set is considered optimal.

(c) Let $L(\boldsymbol{w})$ be a function that we wish to maximise. Gradient ascent performs the update

$$\boldsymbol{w}^{new} = \boldsymbol{w} + \eta \nabla_{\boldsymbol{w}} L$$

where η is the learning rate.

Any picture with more than one maximum, in which we initialise the routine close to the local optimum is fine as an example where gradient ascent will fail.

(d) Logistic regession is such that

$$p(c = 1|\mathbf{x}) = \sigma\left(\theta + \mathbf{w}^T\mathbf{x}\right)$$

The decision boundary is linear. We cannot separate, for example, the XOR problem. Any diagram with non-linearly separable data would be fine.

Arguably, there is little point in first using a linear dimension reduction technique such as PCA, since logistic regression is a function of a linear combination of the inputs. Also, there is more chance that high dimensional data is linearly separable.