*Fifth essay* Raul Adell

This essay may contain ideas given in class complemented by the book given on the references of the course as well as content from Stanford's online ITSL course given by R. Tibshirani and T. Hastie. Book section: 4. Classification. Along with reading the book chapter, and contrasting it with online videos and class notes, this is what I considered most important.


**Classification** is a subfield of machine learning that deals with the task of predicting categorical outcomes. In this context, the goal is to develop a model that can learn from a set of input-output pairs, known as the training data, and then use this knowledge to classify new, unseen examples. This problem can be expressed in the predictor's space as a hyperplane separating (two) classes.

Thus, the equation of this hyperplane would be determined by the normal vector of the plane, $\boldsymbol{\beta}$, which would have the same dimensions as the predictor vector **x** and would determine the orientation, and the coefficient $\boldsymbol{\beta}_0$, which is in charge of moving the plane up or down, depending on the importance of type 1 errors compared to type 2. This last, concept is in fact related to MAP in Bayesian probability.

One of the most widely used algorithms in classification is **logistic regression**. The basic idea behind logistic regression is to model the probability of a binary outcome, such as the presence or absence of a disease, as a function of one or more input variables. The logistic regression model assumes that the log odds of the outcome being positive is a linear function of the input variables, which can be written as:

log(p(x)/(1-p(x))) = β_0 + β_1 *x_1* + *β_2* x_2 + ... + β_p x_p

where p(x) is the probability of the positive outcome given the input variables x_1, x_2, ..., x_p, and β_0, β_1, β_2, ..., β_p are the model coefficients or parameters that need to be estimated from the training data.

The logistic model maps the linear combination of input variables to the probability of the outcome variable taking a particular value. The logistic function is used to transform the linear equation into a value between 0 and 1, which is the probability of a positive outcome. The logistic function is defined as:


This equation is present in many fields of science, from statistical physics to neuronal responses. Is very convenient because it relates the projection of an observation **x** into the plane previously described with a probability.

$$p(y = 1|x) = \frac{1}{1 + e^{-\beta_0 - \beta_1 x_1 - \cdots - \beta_p x_p}}$$

The further the projection from the plane, the bigger the probability (close to 1 or 0). If the projection is in the plane, the model acknowledges the uncertainty and gives it a value of ½.

The coefficients of a logistic regression model can be estimated using maximum likelihood estimation (MLE) method, which involves minimizing the cross-entropy between the predicted and true labels. This loss function solves the underflow problem. The cross-entropy loss function is defined as:

$$J(\beta) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$$

where N is the number of training instances, y_i is the true label of the i-th instance, y_i hat is the predicted probability of the i-th instance belonging to the positive class (i.e., the class with the label 1).

The gradient of the loss function with respect to the coefficients is:

$$\frac{\partial J(\beta)}{\partial \beta_j} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)x_{ij}$$

By iteratively adjusting the values of the coefficients using the gradient descent algorithm or Newton's method, the logistic regression model finds the set of coefficients that minimize the cross-entropy loss function and best fit the training data. These coefficients can then be used to make predictions on new data and give information about the relevance of each feature in the prediction.

One of the advantages of logistic regression is that it produces probabilistic predictions, rather than just binary predictions, which can be useful in some applications. For example, if the logistic regression model estimates a probability of 0.8 for a certain outcome, it means that the model is 80% confident that the outcome will be positive, given the input variables.

Another advantage of logistic regression is that it can handle both linear and nonlinear relationships between the input variables and the outcome. This is achieved by including higher-order terms, interaction terms, or other transformations of the input variables in the model. However, care should be taken to avoid overfitting, which can occur when the model is too complex relative to the amount of available training data.

What we have seen up to now is the classification between classes, but what happens when there are 3 or more? Softmax regression, also known as multinomial logistic regression, is a type of logistic regression that can handle multi-class classification problems. In softmax regression, the probability of an instance belonging to each class is modeled as a function of the input features.

log(odds of category k) = β_k0 + β_k1*x_1* + *β_k2*x_2 + ... + β_kp*x_p

where β_k0, β_k1, β_k2, ..., β_kp are the model coefficients for category k, and x_1, x_2, ..., x_p are the input variables.

The probability of each category is then calculated using the softmax function, which is a generalization of the logistic function. The softmax function transforms the linear equation into a probability distribution over all possible categories. The softmax function is defined as:

$$p(y = k|x) = \frac{e^{x\beta_k}}{\sum_{j=1}^{K} e^{x\beta_j}}$$

where P(Y=k|X=x) is the probability of category k given input x, and the summation is over all possible categories.

$$L(\beta) = \prod_{i=1}^{N} \prod_{k=1}^{K} p(y_i = k|x_i)^{y_{ik}}$$

The likelihood function for softmax regression is:

As described above, we can take the logarithm of the likelihood and find its derivative with respect to the probability coefficients, which yields:

$$\frac{\partial \ell(\beta)}{\partial \beta_{kj}} = \sum_{i=1}^{N} x_{ij}(y_{ik} - p(y_i = k|x_i))$$

The coefficients **β** can be estimated using the maximum likelihood estimation (MLE) method, which involves finding the values of **β** that maximize the log-likelihood function. In the same way as before, this can be done using numerical optimization algorithms.

**Generative models** take a probabilistic approach to classification, modeling the joint probability distribution of the input features and the class label. They use Bayes' theorem to compute the posterior probability of the class label given the input features. In contrast, discriminative models such as logistic regression model the conditional probability of the class label given the input features directly.

$$\Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}.$$

In generative models, the probability density function $f_i(x)$ is modeled separately for each class using a probability density function such as a Gaussian distribution. The prior

probability $\pi_k$ is estimated as the proportion of instances in the training data that belong to each class.

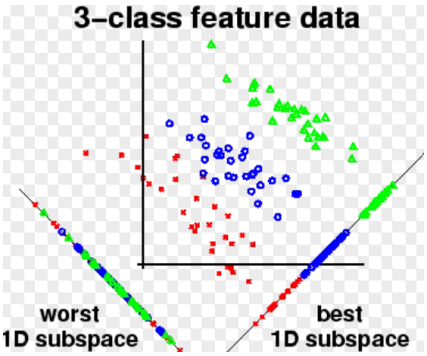
Generative models estimate the prior probability of each class by counting the proportion of instances in the training data that belong to each class. They then use Bayes' theorem to compute the posterior probability of each class given the input features.

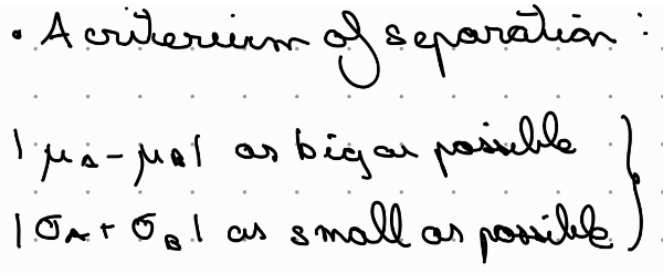
The most simple example of generative model is **Linear Discriminant Analysis**. LDA seeks to find a linear combination of input features that best separates the classes in the training data. To do this, LDA makes the following assumptions:

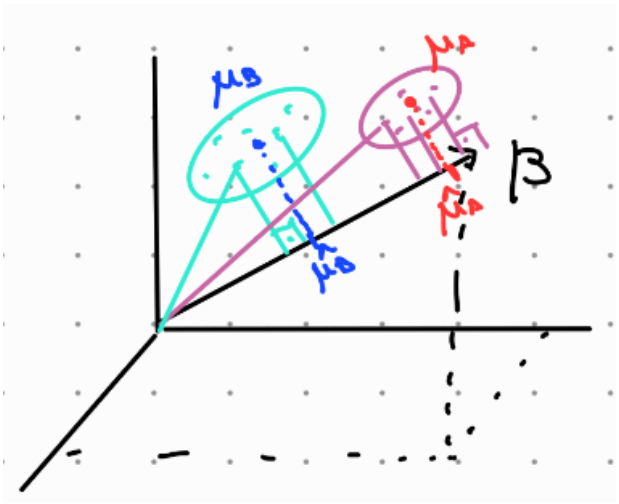
- The input features are normally distributed within each class.
- The classes have equal covariance matrices.

It then uses Bayes' theorem to compute the posterior probability of each class given the input features. The decision boundary is defined as the hyperplane that separates the classes with the maximum margin.

The key idea behind LDA is to project the input features onto a lower-dimensional subspace that maximizes the separation between the classes. This can be seen as maximizing distance between means and minimizing overlapping.




Let's try to mathematically describe the step by step working principle of LDA in order to understand it:

Probability density function of the input features for class k:

$$p(X|y=k) = (2\pi)^{-p/2}|\Sigma|^{-1/2}\exp\left(-\tfrac{1}{2}(X-\mu_k)^T\Sigma^{-1}(X-\mu_k)\right)$$

where μk is the mean of the input features for class k, Σ is the covariance matrix, and |Σ| is the determinant of the covariance matrix.

To classify a new instance x, LDA computes the posterior probability of each class given the input features:

$$p(y=k|x) = \frac{p(x|y=k)p(y=k)}{p(x)}$$

LDA assumes that the covariance matrix Σ is the same for all classes. Under this assumption, the optimal decision boundary is linear and can be defined as the hyperplane that maximizes the separation between the classes. This expression is derived by finding the hyperplane that maximizes the separation between the two classes. The hyperplane is chosen such that the projection of the data points onto it are as far apart as possible in the direction perpendicular to the hyperplane. This is because the decision boundary should be midway between the means of the two classes to ensure that the distance between each class and the decision boundary is equal.

The within-class scatter matrix, SW, is the sum of the scatter matrices of each class, weighted by the proportion of data points in that class. The between-class scatter matrix, SB, is a measure of the spread between the class means, and is calculated as the outer product of the difference between the class means.

$$S_W = \sum_{k=1}^{K}\sum_{i\in C_k}(x_i-\mu_k)(x_i-\mu_k)^T$$

$$S_B = \sum_{k=1}^{K} N_k(\mu_k-\mu)(\mu_k-\mu)^T$$

The Fisher's linear discriminant analysis problem is then formulated as finding the hyperplane that maximizes the ratio of the between-class scatter matrix to the within-class scatter matrix. This can be written, similarly to our initial separation criterium, as:

$$J(\omega) = \frac{\omega^T S_B \omega}{\omega^T S_W \omega}$$

The between-class scatter matrix measures the separation between the means of the different classes, while the within-class scatter matrix measures the spread of the data within each class.

The optimal solution for the discriminant function can be derived by solving the following eigenvalue problem, which arises after rearranging:

$$\frac{\partial J(\omega)}{\partial \omega} = \frac{2S_B\omega \cdot \omega^T S_W\omega - 2\omega^T S_B\omega \cdot S_W\omega}{(\omega^T S_W\omega)^2} = 0 \longrightarrow S_W^{-1} S_B\omega = \lambda\omega$$

where lambda is a scalar eigenvalue and omega is the corresponding eigenvector. This eigenvalue problem can be solved using standard numerical methods such as singular value decomposition (SVD) or eigendecomposition. The resulting eigenvectors correspond to the directions of maximum separation between the different classes.

Thus, taking into account initial assumptions, coefficients have the following expression:

$$\omega = \Sigma^{-1}(\mu_1 - \mu_2)$$

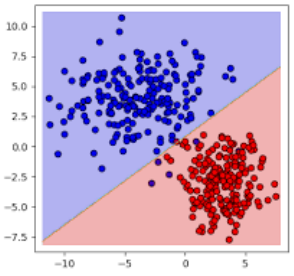
Therefore, the hyperplane that yields better separability following our criterium data is:

$$\omega^T x + b = 0$$

where ω is a vector of coefficients and b is a scalar. The vector ω is normal to the hyperplane.

This expression tells us that the direction of the optimal decision boundary is given by the difference between the two class means, weighted by the inverse covariance matrix of the data. The inverse covariance matrix acts as a scaling factor that gives more weight to the features with lower variance and less weight to the features with higher variance. This ensures that the decision boundary is aligned with the directions of maximum variance in the data.

The eigenvectors corresponding to the largest eigenvalues of the generalized eigenvalue problem in LDA represent the directions in the feature space that maximize the separation between the classes. These directions are also known as the discriminant directions, decision boundaries or the Fisher faces. They are a linear combination of the original features and can be used as a new set of features to project the data onto a lower-dimensional subspace that preserves the class-discriminatory information.



PCA vs LCA

LCA (Linear Discriminant Analysis) and PCA (Principal Component Analysis) are both linear transformations that are commonly used in multivariate data analysis. However, they serve different purposes and have different underlying assumptions.

PCA is a technique used to reduce the dimensionality of a dataset by identifying a smaller number of uncorrelated variables, called principal components, that explain most of the variance in the original data. The goal of PCA is to find a lower-dimensional representation of the data that retains as much information as possible. PCA assumes that the data is centered, and that the variables are linearly related to each other.

LCA, on the other hand, is a technique used for supervised classification, where the goal is to find a linear combination of variables that can best discriminate between two or more groups. LCA finds a set of discriminant functions that maximize the ratio of the between-group variance to the within-group variance. LCA assumes that the data is normally distributed and that the covariance matrix is the same for each group.

In summary, PCA is a technique for reducing the dimensionality of data by finding the most important variables that explain the most variation in the data, while LCA is a technique for finding the linear combination of variables that best discriminate between groups.