1 Classification

For more than two classes, it is hard to maintain the ordering between them using linear regression. For two classes however, linear regression can be used to prepare an ordering of the data (although difficult to interpret as probability themselves).
Classification using linear regression to predict binary reponse will be same as Linear Discriminant Analysis (LDA).

1.1 Logistic Regression

Modelling binary response with linear regression might produce values outside the range $[0, 1]$ (and possibly negative as well). Hence we use a logistic function to compress the outputs to $[0, 1]$ range.

$$p(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$
$$odds = \frac{p(Y = 1|X)}{1 - p(Y = 1|X)} = e^{\beta_0 + \beta_1 X}$$

- The solution to this model is obtained via **Maximum Likelihood Estimation**.
- Odds are also used to interpret probability. A low value of odds (close to 0) indicates a low probability while a high value (close to inf) indicates a high probability.
- One unit change in X will cause β_1 change in the *log odds*.

1.1.1 Loss Function

Maximum Likelihood is used to determine the coefficients. Basic intuition is to choose such a pair of $\beta's$ that will make the predicted probability as close to the correct binary response (0 or 1) as possible.

$$\text{likelihood function} = l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'}))$$
$$\text{Taking logarithm, } logloss = \sum_{i:y_i=1} \log p(x_i) + \sum_{i':y_{i'}=0} \log(1 - p(x_{i'}))$$
$$= \sum_i y \log p_i + (1 - y) \log(1 - p_i) \qquad \text{since } y = 0 \text{ or } 1$$

All the formulae listed here and above extend easily for the case of multiple variables, wherein we simply replace the sum $\beta_0 + \beta_1 X$ with $\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$.

1.2 Linear Discriminant Analysis (LDA)

Why use LDA ?

- When the **classes are well separated**, the parameter estimates for the **logistic regression** model are surprisingly **unstable**. **LDA** does not suffer from this problem and is relatively **stable**.
- if n **is small** and the distribution of X **is approximately normal** in each of the classes, **LDA** is again **more stable** than logistic regression.

- LDA is popular when we have **more than two classes**.

LDA first models the distribution of X in each class, and then uses Bayes' rule to flip this and get $p(Y|X)$. When these distributions of X are normal, the model is very similar in form to logistic regression.

1.2.1 Model Derivation

Let the total number of classes be K and the prior probability that a randomly chosen observation comes from the k^{th} class be $\pi_k = P(Y = k)$. Also, let $f_k(x) = P(X = x|Y = k)$ denote the probability distribution function of X for the data points belonging to the class k. By Bayes' Rule

$$\begin{aligned}
\pi_k &= \frac{\text{Observations in class k}}{\text{Total observations}} \\
p(Y = k|X = x) &= \frac{P(Y = k)P(X = x|Y = k)}{P(X = x)} \\
&= \frac{P(Y = k)P(X = x|Y = k)}{\sum_{l=1}^{K} P(Y = l)P(X = x|Y = l)} \\
&= \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}
\end{aligned}$$

1.2.2 Gaussian Model with one Predictor

We assume the predictor to have a Gaussian distribution. For simplicity, also assume that the variances of X for all the K classes are also the same (fundamental assumption for linearity of decision boundary). Then,

$$\begin{aligned}
f_k(x) &= \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu_k)^2}{2\sigma^2}} \\
p_k(x) &= \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu_k)^2}{2\sigma^2}}}{\sum_{l=1}^{K} \pi_l \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu_l)^2}{2\sigma^2}}}
\end{aligned}$$

For any given x, we notice that all $f_k(x)$'s have the same denominator. To assign a class, we just need to find the maximum value. Taking log, removing the denominator and removing the parts corresponding to x from numerator (since they are same across all classses),

$$\log p_k(x) \propto \log \pi_k + \frac{\mu_k^2}{2\sigma^2} - \frac{x\mu_k}{\sigma^2}$$

In the case of two classes, the decision boundary can be found by equating the two log *probabilities* (assume the priors to be same for simplicity)

$$\begin{aligned}
x\frac{\mu_1}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} &= x\frac{\mu_2}{\sigma^2} - \frac{\mu_2^2}{2\sigma^2} \\
\text{or, } x &= \frac{\mu_1 + \mu_2}{2}
\end{aligned}$$

μ_k and σ^2 need to be estimated from the data, which can be done through the following formulae (n is total training examples and n_k is total training examples from class k)

$$\hat{\mu_k} = \frac{1}{n_k} \sum_{i:y_i=k} x_i$$

$$\hat{\sigma^2} = \frac{1}{N-K} \sum_{k=1}^{K} \sum_{i:y_i=k} (x - \hat{\mu_k})^2$$

1.2.3 Multivariate Gaussian

A Multivariate Gaussian is an extension of the 1-D gaussian to multiple dimensions. Here, we assume that each of the individual dimensions is itself a Gaussian, with the different dimensions having correlation with each other, which are all specified in the correlation matrix.

$$X \sim \mathcal{N}(\mu, \Sigma)$$
$$f(x) = \frac{1}{(2\pi)^{p/2} \mid \Sigma \mid^{1/2}} \exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu))$$

Here, μ is the mean vector Σ is the covariance matrix (symmetric).
Assume μ_k represents the mean vector for individual classes and we have a common covariance matrix across all classes. Plugging this into the LDA equation and removing the common part across all classes, the discriminant becomes

$$\log p_k(x) \propto \log \pi_k + x^T \Sigma^{-1} \mu_k - \frac{1}{2}\mu_k^T \Sigma^{-1} \mu_k$$

To calculate the decision boundary, we simply do a pairwise equality between the discriminants of the individual classes and get the pairwise decision boundaries.

1.2.4 Quadratic Discriminant Analysis (QDA)

The assumption of same covariance matrix Σ across all classes is fundamental to LDA in order to create the linear decision boundaries.
However, in QDA, we relax this condition to allow class specific covariance matrix Σ_k. Thus, for the k^{th} class, X comes from $X \sim \mathcal{N}(\mu_k, \Sigma_k$.
Plugging this into the classification rule to get the discriminants (removing denominators as they are common for all classes)

$$\delta_k(x) = \log \pi_k - \frac{1}{2}\log \mid \Sigma \mid - \frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)$$
$$\text{Note that, } x^T \Sigma_k^{-1} \mu_k = \mu_k^T \Sigma_k^{-1} x \text{ since } \Sigma \text{ is symmetric and } x^T \Sigma_k^{-1} \mu_k \text{ is scalar}$$
$$\delta_k(x) = \log \pi_k - \frac{1}{2}\log \mid \Sigma \mid - \frac{1}{2}x^T \Sigma_k^{-1} x + x^T \Sigma_k^{-1} \mu_k - \frac{1}{2}\mu_k^T \Sigma_k^{-1} \mu_k$$

Notice the term $x^T \Sigma_k^{-1} x$ that gives the classifier it's quadratic form.
However, since we are calculating individual covariance matrices for all the classes, we need to calculate more parameters than before which requires more data.
The following points about QDA vs LDA must be noted

- QDA requires evaluation of substantially more parameters than LDA which subsequently means that more training data points must be available.
- QDA will be superior if the decision boundaries are not linear, i.e., LDA's assumption of equal variances for all classes will not hold true which will cause LDA to have a higher bias.
- QDA is more flexible than LDA which can reduce bias. However, bias-variance trade-off implies that variance can be relatively higher for QDA if training examples are not sufficient.

1.3 Comparison of Classifiers

Logistic Regression and LDA are similar in the sense that they produce linear decision boundaries.

- Logistic Regression estimates coefficients using Maximum Likelihood Estimate
- LDA estimates parameters using the sample mean and variance

For both the models, $\log odds$ takes a linear form. LDA adds a strong assumption of normal distribution of the predictor variables.

Comparison of models

- Logistic Regression is the simplest classifier one can build. It assumes linearly separated decision boundaries. It is usually used as a binary classifier. The decision boundary can be made non linear by adding transformed version of the predictors like second powers, interaction terms etc.
- LDA is also a linear classifier, but works under the assumptions that the decision boundaries are linear and all the classes share the same covariance matrix. It works well with multiple classes. The performance can be quite bad if the underlying variables are not normally distributed.
- QDA is a natural extension of LDA that relaxes the assumption of shared covariance matrix and allows each class to have a separate covariance matrix. This causes QDA to work well when decision boundaries have non linearity
- KNN is a non parametric model that is the most flexible. However, we can get no indication of which predictor is important, and the model can suffer from high variance.

1.4 Classfication Metrics

Several classification metrics are available for binary classifiers which are used based on the problem setting.

1.4.1 Confusion Matrix

This matrix tabulates the number of cases we are classifying and misclassifying.

Confusion Matrix	**Actual Positive**	**Actual Negative**
Predicted Positive	True Positive	False Positive
Predicted Negative	False Negative	True Negative

Based on the above table, we define the following terms

- Accuracy = $\frac{TP+TN}{P+N}$
- Sensitivity or True Positive Rate (TPR) or Recall = $\frac{TP}{P} = \frac{TP}{TP+FN}$
- Specificity or True Negative Rate (TNR) = $\frac{TN}{N} = \frac{TN}{FP+TN}$
- Precision or Positive Predicted Value (PPV) = $\frac{TP}{FP+TP}$
- False Positive Rate = $\frac{FP}{N} = \frac{FP}{FP+TN}$
- F_1 Score = $\frac{2*precision*recall}{precision+recall}$

1.4.2 Receiver Operating Characteristics (ROC Curve)

ROC curve is plot between **True Positive Rate** and **False Positive Rate**, or equivalently, between **sensitivity/recall** and $1-$ **specificity**. The area under the plotted curve is know as AUC score.

The curve is plot by repeatedly constructing the confusion matrix at different probability thresholds (i.e. changing the decision boundary to see how the confusion matrix changes).

ROC Curve is agnostic of the class balancing in the data set, and is thus used frequently in case of class imbalance to judge a classifier. A random classifier will have AUC of 0.5 as at any threshold, the number of correctly and incorrectly classified points will roughly be the same. A perfect classifier will be able to segregate the population perfectly and will have the value of AUC as 1.0.