Generative Models for Classification

In the case of Logistic Regression we directly model $\Pr(Y = k|X = x)$ using the Sigmoid Function (Logistic Function) its a simple conditional probability approach **Predicting Y given X**.
Considering the alternative Probability of the the the predictors X given a class Y which is the logic behind **Bayes Theorem**, When the distribution of X is normal the model turn to be very similar to Logistic Regression.

Why its needed?

- If There is a substantial separation between the two classes the **parameter estimates** for logistic regression model will be unstable
- If the predictors X are normally distributed and the sample size n is small Using **Generative Models** will be more accurate than Logistic Regression

Suppose that we want to classify an observation into on of the K classes where$K \geq 2$, means the Response Y can take K possible classes.

- Let π_k be the **Prior Probability** that the a random Observation comes from the $k\text{th}$ class $P(Y = k)$, Intuitively think of it as how much the class k is represented in the data set
  - For **Example** : We have a data set with two classes and class 1 have 1000 while class 2 have 10 , so we start with a prior knowledge that we expect this observation to fall into class 1
- Let $f_k(X) = \Pr(X|Y = k)$ is the **Density Function** of X
  - $f_k(x)$ will be large if there is **high probability** of that observation being in the $k\text{th}$ class

$$\Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum^K \pi_i f_i(x)}$$

- This is just the Bayes' theorem Formula
- Let $p_k(x) = \Pr(Y = k|X = x)$ and will also be called the **Posterior** probability
- So our goal is to know the probability of the Observation being in the $k\text{th}$ class *Given* the class k aka Response Y

Linear Discriminant Analysis Vs Logistic Regression

First before diving into **LDA** ,Lets establish a clear difference between What Logistic Regression Does and What **LDA Does** , cause at the end they both Classify a new observation into a class.
What Logistic Regression Does

- It tries to directly model the **posterior probability**

$$P(Y = k|X = x) = \frac{e^{\vec{X}\vec{\beta}}}{1 + e^{\vec{X}\vec{\beta}}}$$

- You just want to separate the classes,Its **Discriminative** just making distinctions
- *We don't care how the data X looks like inside each class*(We make no assumption about the distribution of X, We aim class each observation
- We also don't care about how the data was generated

What LDA Does

- Its a **generative**, it models how the data X looks within each class
- With the assumption of the data X taking a normal distribution form

$$P(X|Y = k) = \mathcal{N}\left(\mu_k, \sum\right)$$

- Then use Bayes' theorem to compute

$$P(Y = k|X = x) = \frac{P(X|Y = k)P(Y = k)}{P(X)}$$

- Then classify x by choosing the class with the highest **posterior** $P(Y = k|X = x)$
- Simply its saying *This new point looks most like the kind of data that class k generates*

Aspect	Logistic Regression	LDA
Type	Discriminative ,**Making distinctions**	Generative
Models	$P(Y\|\|X)$	$P(X\|\|Y), P(Y)$
Assumes	No assumptions	$X\|\|Y$ is Normally Distributed, All classes variances are equal
Uses	Decision boundaries	Bayes's Classifying

Linear Discriminant Analysis for $p = 1$

For this section we will assume that $p = 1$ only one predictor, The goal is :

- Obtain estimate for $f_k(x) \rightarrow \text{The update}$
- We plug it into the Bayes' theorem formula to get the estimate for $p_k(x) \rightarrow \text{ The posterior}$
- Then we **Classify** an Observation to the class with the highest $p_k(x)$ The posterior

Estimating $f_k(x)$

Also known as the The update noted $P(X|Y)$ the **easy to measure** probability used to **update** the The prior π_k, It can also be referred to as the likelihood of the class k

- We assume that $f_k(x)$ is **normal or Gaussian**

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-1/2\sigma_k^2(x-\mu_k)^2}$$

- $\mu_k \rightarrow$ The mean for the $k\text{th}$ class
- $\sigma_k^2 \rightarrow$ the variance for the $k\text{th}$ class

Assuming that the variance of all classes K are equal so $\sigma_k^2 = \sigma^2$ for simplicity

$$P(Y = k|X) = p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} e^{-1/2\sigma^2(x-\mu_k)^2}}{\sum^k \pi_i \frac{1}{\sqrt{2\pi}\sigma} e^{-1/2\sigma^2(x-\mu_k)^2}}$$

- This is the **Posterior** with the assumption of all **variances are equal** and that $f_k(x)$ is **normally distributed**

Dropping the denominator in the **Posterior** formula we get the Discriminant Functions which is the log of the **normalized posterior**, The denominator is constant across all classes k

$$\log(p_k(x)) = \log(\pi_k) - \frac{1}{2\sigma^2}(x - \mu_k)^2$$

- Dropping the term $\frac{1}{\sqrt{2\pi}\sigma}$ for being a constant

$$\log(p_k(x)) = \log(\pi_k) - \frac{1}{2\sigma^2}x^2 - \frac{1}{2\sigma^2}\mu_k^2 + \frac{x\mu_k}{\sigma^2}$$

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

- Dropping x^2 will get δ which is the a **Linear Discriminant Function** of x
- δ_k is the score for class k
- We aim to assign $X = x$ to the class that gives the largest δ_k

If $K = 2$ and $\pi_1 = \pi_2$ then the Bayes decision boundary is the point where $\delta_1 = \delta_2$

$$\delta_1 = \delta_2$$

$$x\frac{\mu_1}{\sigma^2} - \frac{\mu_1^2}{2\sigma^2} + \log(\pi_1) = x\frac{\mu_2}{\sigma^2} - \frac{\mu_2^2}{2\sigma^2} + \log(\pi_2)$$

$$\log(\pi_1) = \log(\pi_2)$$

$$x\frac{\mu_1}{\sigma^2} - \frac{\mu_1^2}{2\sigma^2} = x\frac{\mu_2}{\sigma^2} - \frac{\mu_2^2}{2\sigma^2}$$
$$x\frac{\mu_1 - \mu_2}{\sigma^2} = \frac{\mu_1^2 - \mu_2^2}{2\sigma^2}$$
$$x = \frac{\mu_1^2 - \mu_2^2}{2(\mu_1 - \mu_2)} = \frac{\mu_1 - \mu_2}{2}$$

- If $x > \frac{\mu_1 - \mu_2}{2}$ we assign it to class 1 otherwise class 2

In practice we are not quite certain of our assumption that X is **normally distributed** and we will still need to estimate : LDA Mean And Variance Estimates

- $\mu_1 \dots \mu_K$
- $\pi_1 \dots \pi_k$
- σ^2

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

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

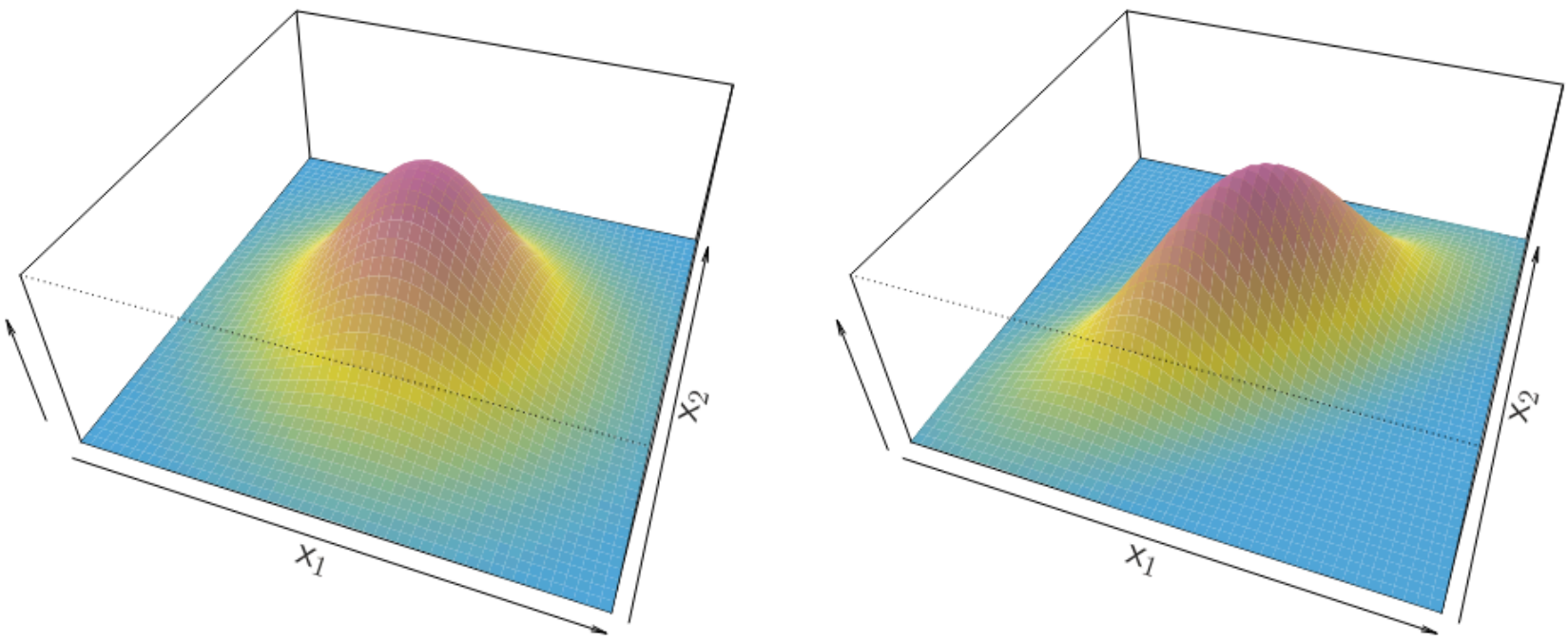
$$\hat{\pi_k} = \frac{n_k}{n}$$

The estimated discriminant function for class k is given then :

$$\hat{\delta_k}(x) = x\frac{\hat{\mu_k}}{\hat{\sigma^2}} - \frac{\hat{\mu_k}^2}{2\hat{\sigma}^2} + \log(\hat{\pi_k})$$

Linear Discriminant Analysis for $p > 1$

Here we assume that the X predictors are drawn from a Multivariate Normal Distribution with a mean for each class k μ_k and a **covariance matrix**



- The **Left Figure** shows a multivariate normal distribution of two predictors X with no **correlation**, each of the predictors is a one dimensional normal distribution
- The **Right Figure** shows the same as the left one but with a correlation value of 0.7
  The formula for Multivariate Normal Distribution is given by:

$$f_k(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{\frac{-1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu)}$$

- μ_k is a mean vector special for each k class
- Σ is a covariance matrix that is common among all classes k

Plugging it into the Bayes classifier format:

$$P(Y=k|X=x) = p_k(x) = \frac{\pi_k \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{\frac{-1}{2}(X-\mu_k)^T\Sigma^{-1}(X-\mu_k)}}{\sum_i^K \pi_i \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{\frac{-1}{2}(X-\mu_i)^T\Sigma^{-1}(X-\mu_i)}}$$

- Σ is the covariance matrix common with all K classes $p \times p$
- X observation vector $p \times 1$
- μ_k Mean vector for class k with $p \times 1$

From that expression we can derive and get this Discriminant Functions :

$$\log(p_k(x) = \log(\frac{\pi_k \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{\frac{-1}{2}(X-\mu_k)^T\Sigma^{-1}(X-\mu_k)}}{\sum_i^K \pi_i \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{\frac{-1}{2}(X-\mu_i)^T\Sigma^{-1}(X-\mu_i)}}))$$

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

- The **denominator** is shared across all classes k so its can be dropped
- $\frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}}$ is a **constant** term so we can drop it

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

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

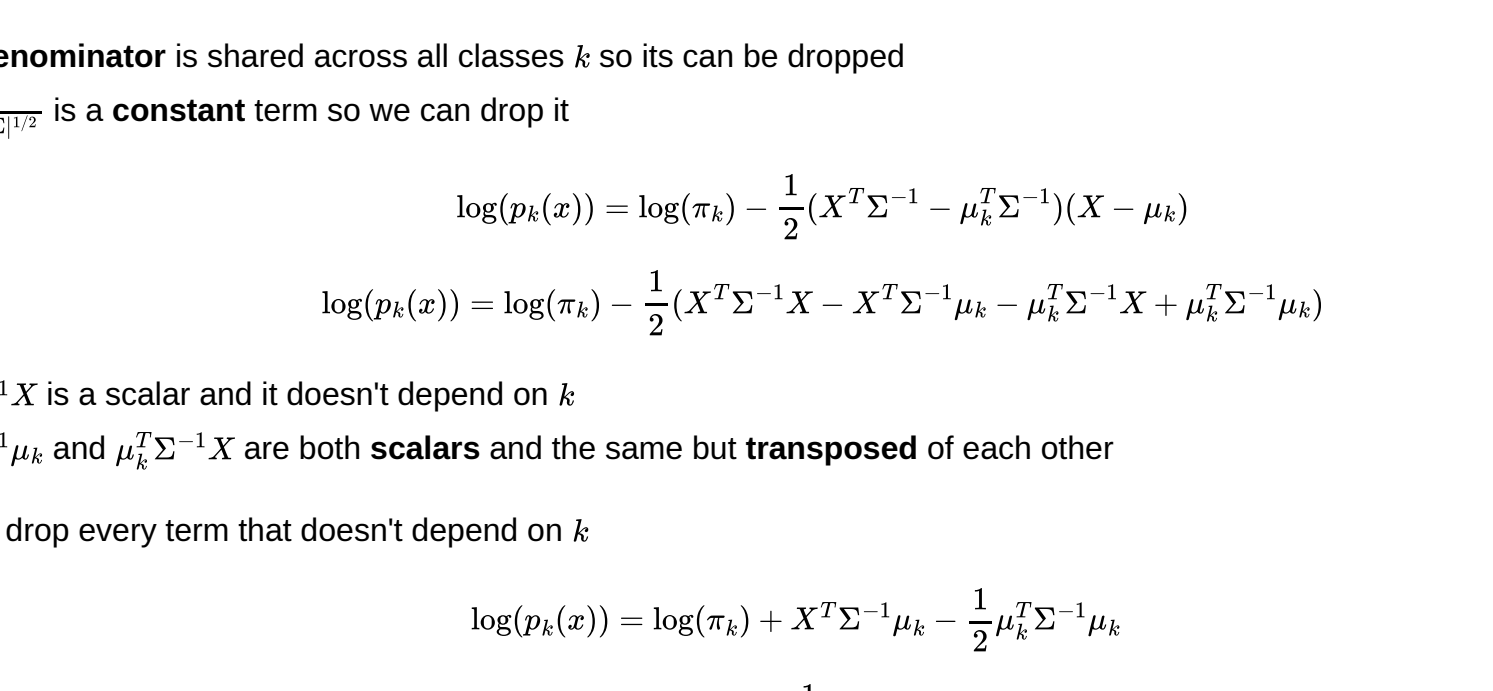
- $X^T\Sigma^{-1}X$ is a scalar and it doesn't depend on k
- $X^T\Sigma^{-1}\mu_k$ and $\mu_k^T\Sigma^{-1}X$ are both **scalars** and the same but **transposed** of each other

Note : We drop every term that doesn't depend on k

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

$$\delta_k(x) = X^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log(\pi_k)$$

- $\delta_k(x)$ is the **Discriminant Function**
- We assign an observation X to the class k with the highest $\delta_k(x)$



- Dashed Lines represent where $\delta_k(x) = \delta_j(x)$, Which is the Bayes decision boundary
- The ellipses represent regions with 95% of the **probability** for each class k

- π_k is equal across all classes k
  Note : $\pi_k = P(Y = k)$ its the **Prior Probability**, The number of Observation does effect π_k, Since we estimate $\hat{\pi_k} = \frac{n_k}{n}$

		True default status		
		No	Yes	Total
Predicted default status	No	9644	252	9896
	Yes	23	81	104
	Total	9667	333	10000

- This is a **Confusion matrix** Where the diagonals shows the correct predictions
- Non-diagonal values shows the incorrect predictions LDA resulted

Out of 9667 only 23 were falsely predicted to **default** their credit card accounts, But our of 333, 252 been missed by LDA

While the overall error rate is low but from the perspective of a company predicting 252 out of 333 wrong is unacceptable. This is a Sensitivity Vs Specificity case where :

- The percentage of true defaulters identified is 24%
- While the non-defaulters that are correctly identified is 99% (Specificity)

The question is *Why LDA have such a low sensitivity*

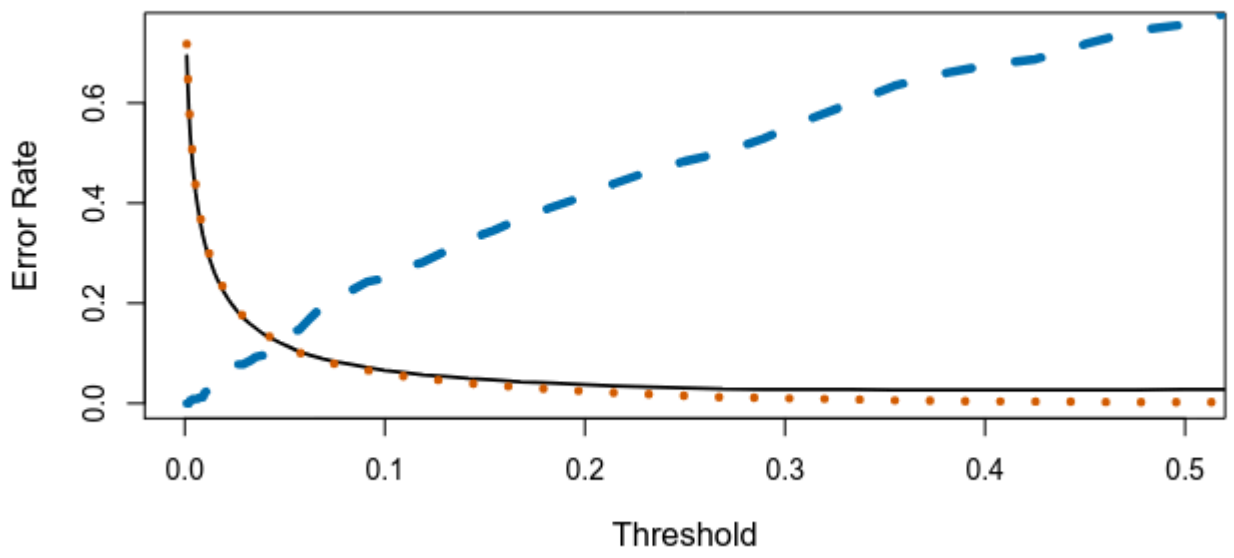
- **LDA** is trying to approximate **Bayes Classifier** which yields the smallest possible total number of misclassified observations
- The overall error rate is low and that's the result of using **Bayes Classifier** which yeah gives the lowest but the cost of the people who defaulted and predicted to not is much more costly than wrongly classifying someone who wont default and predicted to be default

Changing the threshold of 50% that the **Bayes Classifier** uses

$$Pr(\text{default} = \text{Yes}|X = x) > 0.5$$

to a lower threshold of 20%

$$Pr(\text{default} = \text{Yes}|X = x) > 0.2$$



- Will result in a higher overall error rate but will reduce the cost error of misclassified a person who default
- Here it comes the to **Domain Knowledge** and the trade off you willing to take depends on how costly the False positive or False negative on the problem we predicting
- More details about Threshold selection on ROC Curve

Quadratic Discriminant Analysis

As discussed above **LDA** makes to assumptions :

- The observations are **Gaussian** or **Normally** distributed
- All classes k have the same variance σ, **covariance** matrix Σ

Quadratic Discriminant Analysis is a alternative approach where it makes one assumption :

- The observations are **Gaussian**
- While each class k has its own variance σ , **covariance** matrix Σ_k

Since each class k has its own **covariance matrix** the Discriminant Functions for **QDA** becomes :

$$\log(p_k(x)) = \log(\pi_k) - \frac{1}{2}(X-\mu_k)^T\Sigma_k^{-1}(X-\mu_k) - \frac{1}{2}\log|\Sigma_k|$$

$$\delta_k = \log(p_k(x)) = \log(\pi_k) - \frac{1}{2}X^T\Sigma_k^{-1}X + X^T\Sigma_k^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma_k\mu_k - \frac{1}{2}\log|\Sigma_k|$$

The Bayes classifier assigns an observation $X = x$ to the class with the largest δ_k, As you notice that the **discriminant function** is non-linear and quadratic.

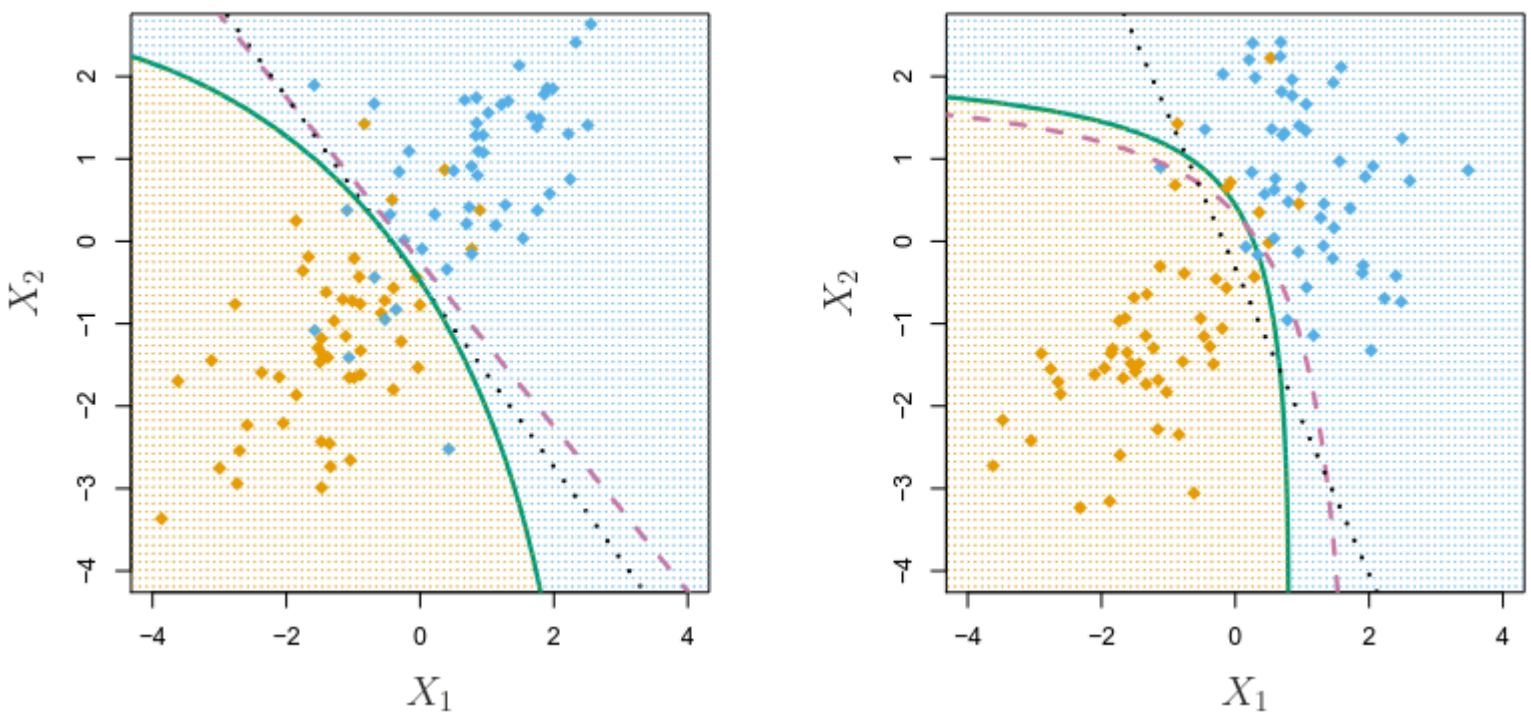
*The question know what makes **LDA** better than **QDA** or vice versa?*

Here its all about bias-variance trade-off, When there is p features, in the case of **QDA** we estimate the covariance matrix for each class k

$$\frac{p(p+1)}{2}$$

- Since its covariance matrix is $p \times p$
- **diagonal elements** are the variance for the class k
- the **upper** and **lower** elements $\frac{p(p-1)}{2}$, since the upper and lower elements repeat
- so the number of unique parameters $= p + \frac{p(p-1)}{2} = \frac{p(p+1)}{2}$

Which means when the number of **features** is large the **QDA** requires large amounts of **observations**, which the **LDA** doesn't struggle with since it assumes all classes k have equal covariance matrices, hence why **LDA** is less flexible classifier than **QDA**



- When the **Bayes classifier** which is the "Golden Standard classifier" is linear left figure
  - Which we can also notice that both classes have the *kinda* same variance
  - Using **LDA** the less flexible approach results in a better classifying black dotted
  - And the **QDA** preforms worse Green curve
- When the **Bayes classifier** is non-linear right figure
  - Looking at the data they are more spread and higher **variance**
  - So using **QDA** which calculate each indivudual covariance matrix for each class, results in better classification(similar to **Bayes classifier**)
  - While **LDA** struggles where the variance is higher cause of its assumption it makes

Naive Bayes

In both **LDA** and **QDA** we used Bayes' theorem to develop the classifiers, Building off the same explained Bayes theorem expression introduced earlier

$$P(Y = k|X = x) = \frac{P(X|Y = k)P(Y = k)}{P(X)}$$

in The **Naive Bayes** the sol assumption it makes is within the class k the features p are **independent**

$$f_k(x) = f_{k1}(x_1) + f_{k2}(x_2) + \cdots + f_{kp}(x_p)$$

Simply it means that each predictor(feature) has its own distribution, which also implies that there is no covariates between features p

Which is a pretty strong assumption to make but it works pretty decent where n isn't large enough relative to p , Also **Naive Byes** introduce some bias while reducing the variance, So we get

$$\Pr(Y = k|X = x) = \frac{\pi_k \times f_{k1}(x_1) \times f_{k2}(x_2) \cdots \times f_{kp}(x_p)}{\sum_{i=1}^{K} \pi_i \times f_{i1}(x_1) \times f_{i2}(x_2) \cdots \times f_{ip}(x_p)}$$

$$f_j(X) = \prod_{k=1}^{p} f_{jk}(X_k)$$

- The goal is to estimate the joint probability of $f_j(X) = f_{k1}(x_1) \times f_{k2}(x_2) \cdots \times f_{kp}(x_p)$
- Its called **Naive** cause of the assumption it makes which is so rare to have **independent features**
- And its main strong point is its only require small amount of Training Data to estimates the parameters needed for classification

Types of Naive Bayes

Multinomial Naive Bayes

Its widely used in NLP (spam detection , sentiment analysis , document classification) with the assumption of :

- **Features** are independent
- **Features** follows a **multinomial distribution**

Following this :

$$\Pr(Y = k|X = x) = \frac{\pi_k \times f_{k1}(x_1) \times f_{k2}(x_2) \cdots \times f_{kp}(x_p)}{\sum_{i=1}^{K} \pi_i \times f_{i1}(x_1) \times f_{i2}(x_2) \cdots \times f_{ip}(x_p)}$$

we can model it for the **Multinomial Naive Bayes** by :

$$P(X|Y = k) = \frac{(\sum_{i=1}^{p} x_i)!}{\prod_{i=1}^{p} x_i!} \prod_{i=1}^{p} \theta_{k,i}^{x_i}$$

- Which is the likelihood or the **update** in Bayes' theorem and since the **denominator** is constant

We can represent the **Multinomial Naive Bayes** as :

$$P(Y = k|X) = \pi_k \times P(X|Y = k)$$

$$\log P(Y = k|X) \propto \log \pi_k + \log P(X|Y = k)$$

With :

$$\log P(X|Y = k) = \log \frac{(\sum x_i)!}{\sum x_i!} + \sum_{i=1}^{p} x_i \log \theta_{k,i}$$

- $\theta_{k,i} = P(\text{word } i|\text{class } k)$ probability of the word i given a class k

To calculate the θ

$$\theta_{k,i} = \frac{N_{k,i}}{N_k}$$

Adding $\alpha = 1$ for *Laplace* smoothing

$$\theta_{k,i} = \frac{N_{k,i} + \alpha}{N_k + \alpha p}$$

- $N_{k,i}$ Number of times the feature x_i appear in the class k
- N_k Total count of features in class k

the estimate for the **prior** π_k is calculated the same as in **LDA** and **QDA** :

$$\pi_k = \frac{n_k}{n}$$

- Ignoring out both of $(\sum x_i)!$ and $\sum x_i!$ cause they do not depend on the class k
- They are constant with respect to X_k

We get :

$$\log P(Y = k|X) \propto \log \pi_k + \sum_{i=1}^{n} x_i \log \theta_{k,i}$$

$$\hat{y} = \operatorname{argmax} P(X_k|Y) = \operatorname{argmax} \left[\log \pi_k + \sum_{i=1}^{p} x_i \theta_{k,i} \right]$$

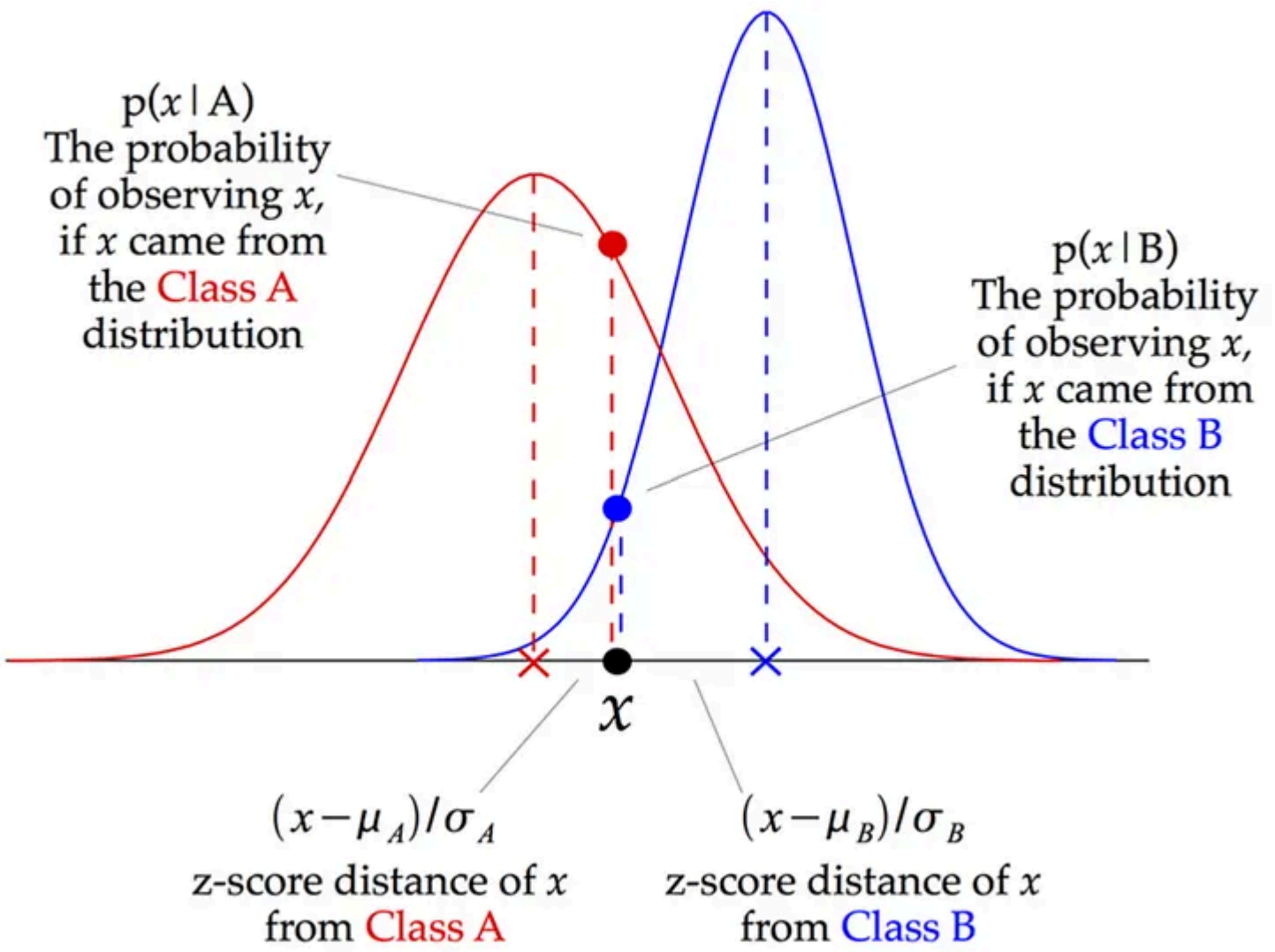
There is a fatal error that the **Naive Bayes** could make in the case of the likelihood of a word i given a class k is zero

- Which results in $\hat{y} = \operatorname{argmax} \pi_k \prod P(X|Y = k)$ cannot be conditioned away
- Its solved with *Laplace* smoothing

Gaussian Naive Bayes

Naive Bayes can also be used in continuous data, where the features follows a **Gaussian** distribution

$$P(X|Y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(X_i - \mu_{k,i})^2 / 2\sigma_{k,i}^2}$$



- To classify each new data point x the algorithm finds the maximum value of posterior probability of each class and assign the data point to that class

Following these assumptions :

- The **Features** are Gaussian (normally) distributed
- So we only need to estimate the mean μ and standard deviation σ^2
- Without estimating the distribution for each of the features

Estimating the mean μ and std σ^2

Since we assume that the predictors follows a normal distribution its the same estimated values for μ and σ^2 as in LDA Mean And Variance Estimates

$$L(\mu_{k,i}, \sigma^2_{k,i}) = \prod_{i=1}^{k} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(X_i-\mu)^2/2\sigma^2}$$

Log Likelihood

$$l(\mu_{k,i}, \sigma^2_{k,i}) = \log L(\mu, \sigma^2_{k,i}) = \sum_{i=1}^{k} \left[-\frac{1}{2}\log(2\pi\sigma^2) - \frac{(X_i - \mu_{k,i})^2}{2\sigma^2_{k,i}} \right]$$

Derivation w.r.t $\mu_{k,i}$:

$$\frac{d\ l(\mu, \sigma^2)}{d\mu} = -\sum_{i=1}^{k} X_i - \mu_{k,i} = 0$$

$$\frac{dl(\mu, \sigma^2)}{d\mu} = n\mu_{k,i} = \sum_{i=1}^{k} X_i$$

$$\hat{\mu}_{k,i} = \frac{1}{n_k} \sum_{i=1}^{k} X_i$$

Derivation w.r.t $\sigma^2_{k,i}$:

$$\frac{d\ l(\mu, \sigma^2)}{d\mu} = \sum_{i=1}^{k} \left[-\frac{1}{2}\log(2\pi\sigma^2_{k,i}) - \frac{(X_i - \mu_{k,i})^2}{2(\sigma^2)^2} \right]$$

Multiply by σ^4

$$\hat{\sigma^2_{k,i}} = \frac{1}{n_k} \sum_{i=1}^{k} (X_i - \mu_{k,i})^2$$

- These are the estimates $\hat{\mu}_{k,i}$ and $\hat{\sigma^2_{k,i}}$

With this we get the likelihood function to plug it into the **posterior** :

$$\log P(Y|X) \propto \log P(Y) + \log P(X|Y)$$

With : $P(X|Y)$

$$P(X|Y) = \frac{1}{\sqrt{2\pi\sigma^2_{k,i}}} e^{-(X_i-\mu_{k,i})^2/2\sigma^2_{k,i}}$$

$$\log P(Y|X) = \text{argmax}\ [\log \pi_k - \sum \frac{(X_i - \mu_{k,i})^2}{2\sigma^2_{k,i}}]$$

Conclusion

- Gaussian Naive Bayes assumes that all the features follows a **Normal** distribution
- Both the mean μ and variance σ are estimated for each feature $X_i$in each class k
- Assumes the Independence of all the features X

Naive Bayes and Text Classification