Lecture 14: Discriminant Analysis

CS109A Introduction to Data Science

Pavlos Protopapas and Kevin Rader

Lecture Outline

- Discriminant Analysis
  - LDA for one predictor
  - LDA for $p > 1$
  - QDA
- Comparison of Classification Methods (so far)

Recall the Heart Data (for classification)

response variable ***Y*** is Yes/No

Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD
63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversable	Yes
37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No

Discriminant Analysis for Classification

Linear Discriminant Analysis (LDA)

Linear discriminant analaysis (LDA) takes a different approach to classification than logistic regression. Rather than attempting to model the conditional distribution of Y given X, $P(Y = k|X = x)$, LDA models the distribution of the predictors X given the different categories that Y takes on, $P(X = x|Y = k)$.

In order to flip these distributions around to model $P(X = x|Y = k)$ an analyst uses Bayes' theorem.

In this setting with on … n then be written as:

$$P(Y = k|X = x) = \frac{f_k(x)\pi_k}{\sum_{j=1}^{K} f_j(x)\pi_j}$$

What does this mean?

LDA (cont.)

$$P(Y = k|X = x) = \frac{f_k(x)\pi_k}{\sum_{j=1}^{K} f_j(x)\pi_j}$$

The left hand side, $P(Y = k|X = x)$, is called the *posterior* probability and gives the probability that the observation is in the k^{th} category given the feature, X, takes on a specific value, x. The numerator on the right is conditional distribution of the feature within category k, $f_k(x)$, times the *prior* probability that observation is in the k^{th} category.

The *Bayes' classifier* is then selected. That is the observation assigned to the group for which the posterior probability is the largest.

Inventor of LDA: R.A. Fisher

The 'Father' of Statistics. More famous for work in genetics (statist concluded that Mendel's genetic experiments were 'massaged').

Novel statistical work includes:

- Experimental Design
- ANOVA
- *F*-test (why do you think it's called the *F*-test?)
- Exact test for 2 x 2 tables
- Maximum Likelihood Theory
- Use of $\alpha = 0.05$ significance level: "The value for which P = .05, or 1 in 20, is 1.96 or nearly 2; it is convenient to take this point as a limit in judging whether a deviation is to be considered significant or not".
- And so much more...

LDA for one predictor

LDA has the simplest form when there is just one predictor/feature ($p = 1$). In order to estimate $f_k(x)$, we have to assume it comes from a specific distribution. If X is quantitative, what distribution do you think we should use?

One common assumption is that $f_k(x)$ comes from a Normal distribution:

$$f_k(x) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)$$

In shorthand notation, this is often written as $X \mid Y=k \sim N(\mu_k, \sigma_k^2)$,
meaning, the distribution of the feature X within category k is Normally distributed with mean μ_k and variance σ_k^2.

LDA for one predictor (cont.)

An extra assumption that the variances are equal,

$\sigma_1^2 = \sigma_2^2 = ... = \sigma_K^2$ will simplify are lives.

Plugging this assumed likelihood into the Bayes' formula (to get the posterior) results in:

$$P(Y = k|X = x) = \frac{\pi_k \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu_k)^2}{2\sigma^2}\right)}{\sum_{j=1}^{K} \pi_j \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu_j)^2}{2\sigma^2}\right)}$$

The Bayes classifier will be the one that maximizes this over all values chosen for *x*. How should we maximize?

So we take the log of this expression and rearrange to simplify our maximization...

LDA for one predictor (cont.)

So we maximize the following simplified expression:

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

How does this simplify if we have just two classes (*K* = 2) and if we set our prior probabilities to be equal?

This is equivalent to choosing a decision boundary for *x* for which

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

Intuitively, why does this expression make sense? What do we use in practice?

LDA for one predictor (cont.)

In practice we don't know the true mean, variance, and prior. So we estimate them with the classical estimates, and plug-them into the expression:

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

and

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

where n is the total sample size and n_k is the sample size within class k (thus, $n = \sum nk$).

LDA for one predictor (cont.)

This classifier works great if the classes are about equal in proportion, but can easily be extended to unequal class sizes.

Instead of assuming all priors are equal, we instead set the priors to match the 'prevalence' in the data set:

$$\hat{\pi}_k = \hat{n}_k / n$$

Note: we can use a prior probability from knowledge of the subject as well; for example, if we expect the test set to have a different prevalence than the training set.

How could we do this in the Dem. vs. Rep. data set?

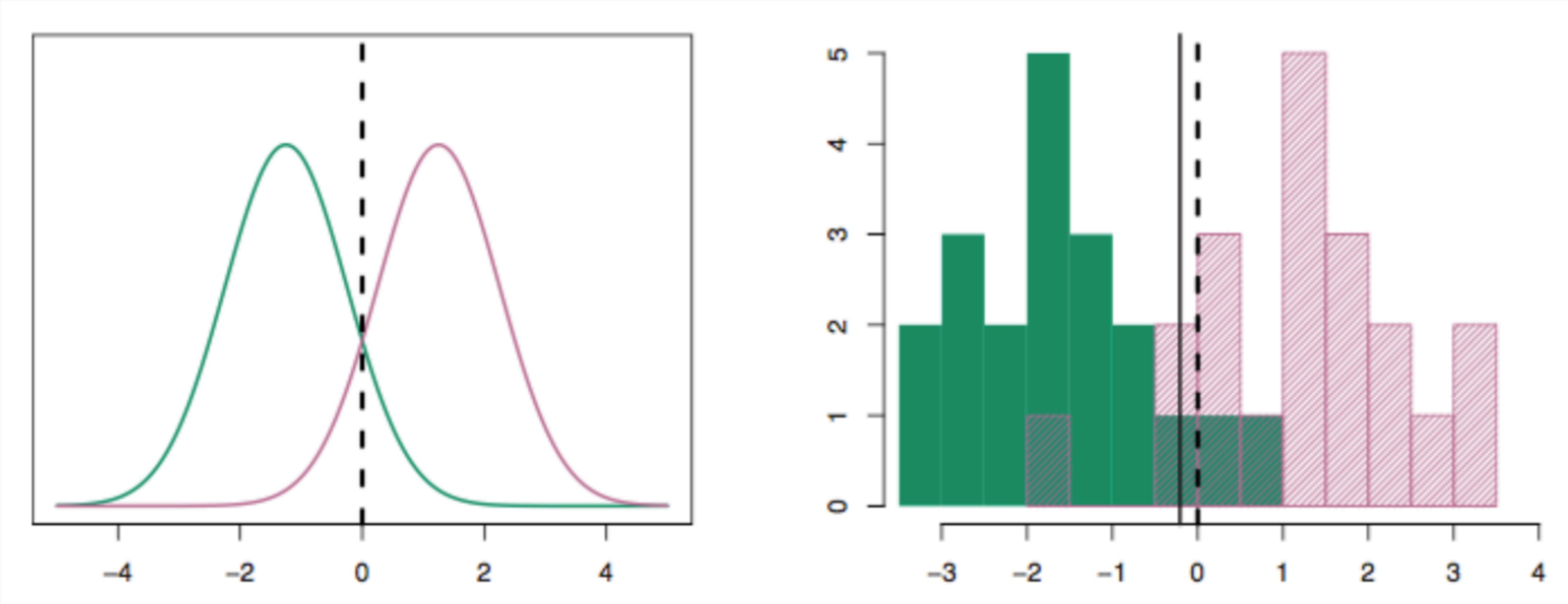
LDA for one predictor (cont.)

Plugging all of these estimates back into the original logged maximization formula we get:

$$\hat{\delta}_k(x) = x\frac{\hat{\mu}_k}{\hat{\sigma}^2} - \frac{\hat{\mu}_k^2}{2\hat{\sigma}^2} + \log \hat{\pi}_k$$

Thus this classifier is called the linear discriminant classifier: this discriminant function is a linear function of *x*.

Illustration of LDA when $p = 1$



LDA when $p > 1$

LDA when $p > 1$

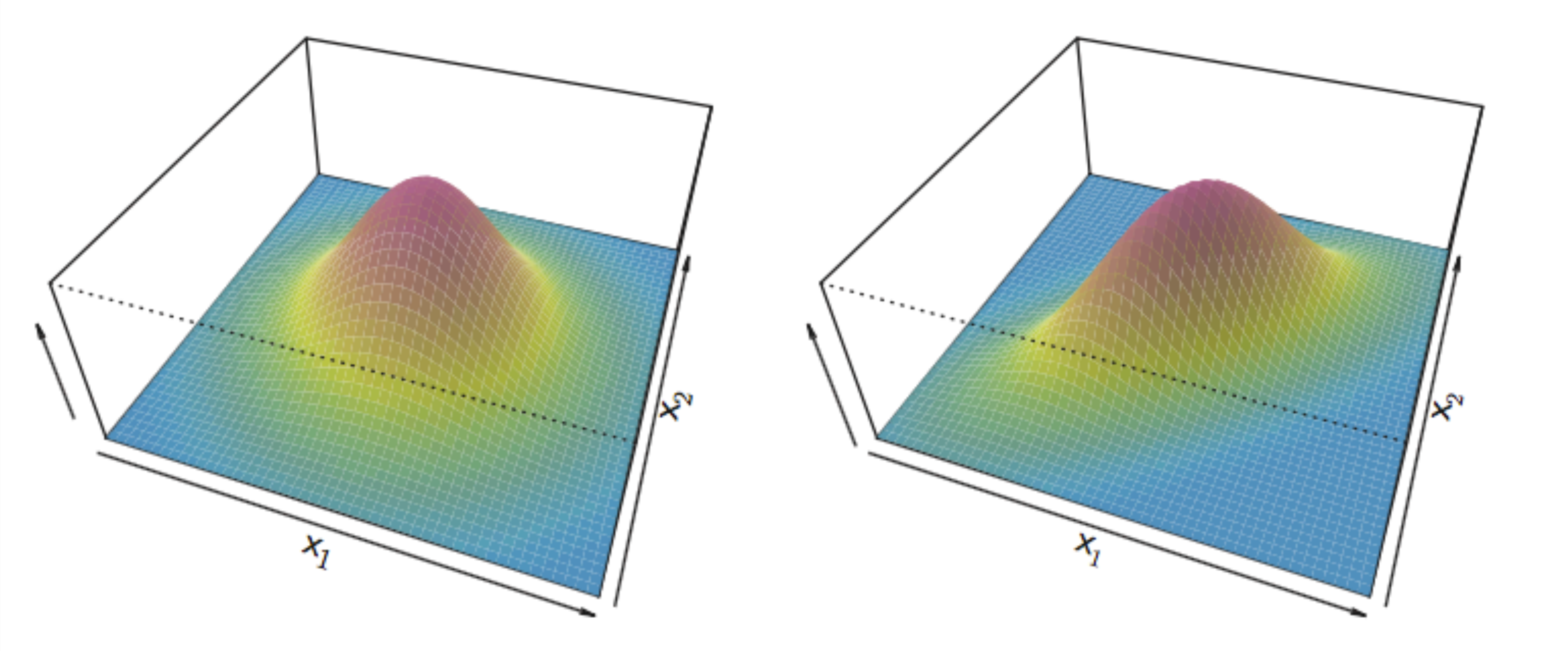
LDA generalizes 'nicely' to the case when there is more than one predictor.

Instead of assuming the one predictor is Normally distributed, it assumes that the set of predictors for each class is 'multivariate normal distributed' (shorthand: MVN). What does that mean?

This means that the vector of X for an observation has a multidimensional normal distribution with a mean vector, μ, and a covariance matrix, $\mathbf{\Sigma}$.

Multivariate Normal Distribution

Here is a visualization of the Multivariate Normal distribution with 2 variables:



MVN Distribution

The joint PDF of the Multivariate Normal distribution,
$\vec{X} \sim MVN(\vec{\mu}, \Sigma)$, is:

$$f(\vec{x}) = \frac{1}{2\pi^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\vec{x}-\vec{\mu})^T\Sigma^{-1}(\vec{x}-\vec{\mu})\right)$$

where $\vec{X}$ is a *p* dimensional vector and $|\Sigma|$ is the determinant of the *p* x *p* covariance matrix.

Let's do a quick dimension analysis sanity check...

What do $\vec{X}$ and Σ look like?

LDA when $p > 1$

Discriminant analysis in the multiple predictor case assumes the set of predictors for each class is then multiva $\vec{X} \sim MVN(\vec{\mu}, \Sigma)$

Just like with LDA for one predictor, we make an extra assumption that the covariances are equal in each group, $\Sigma_1 = \Sigma_2 = ... = \Sigma_K$. in order to simplify our lives.

Now plugging this assumed likelihood into the Bayes' formula (to get the posterior) results in:

$$P(Y = k|\vec{X} = \vec{x}) = \frac{\pi_k \frac{1}{2\pi^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\vec{x} - \vec{\mu}_k)^T \Sigma^{-1} (\vec{x} - \vec{\mu}_k)\right)}{\sum_{j=1}^{K} \frac{1}{2\pi^{p/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\vec{x} - \vec{\mu}_j)^T \Sigma^{-1} (\vec{x} - \vec{\mu}_j)\right)}$$

LDA when p > 1 (cont.)

Then doing the same steps as before (taking log and maximizing), we see that the classification will for an observation based on its predictors, x, will be the one that maximizes (maximum of K of these $\delta_k(x)$):

$$\delta_k(\vec{x}) = \vec{x}^T \Sigma^{-1} \vec{\mu}_k - \frac{1}{2} \vec{\mu}_k^T \Sigma^{-1} \vec{\mu}_k + \log \pi_k$$

Note: this is just the vector-matrix version of the formula we saw earlier in lecture:

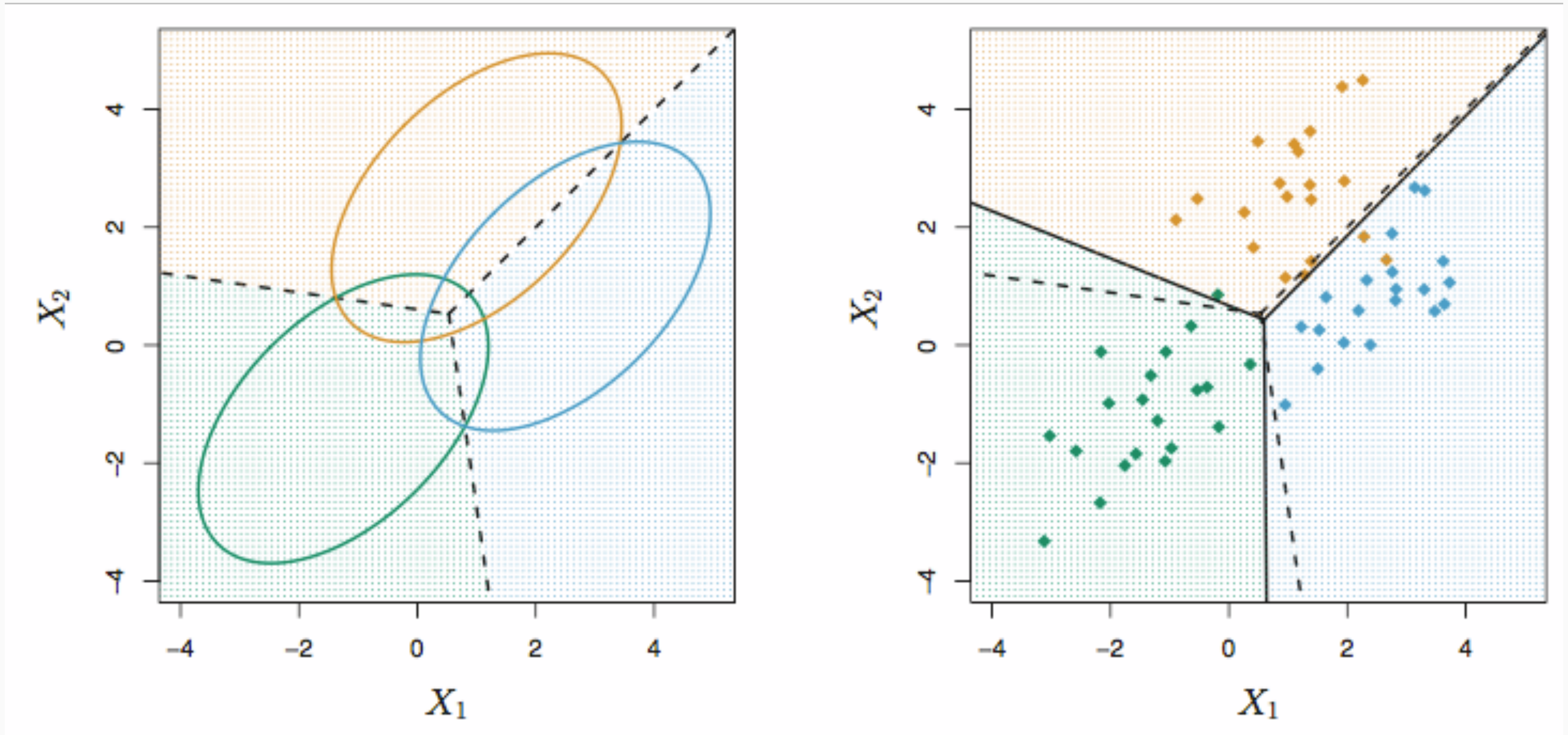
$$\delta_k(x) = x \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log \pi_k$$

What do we have to estimate now with the vector-matrix version? How many parameters are there?

There are *pK* means, *pK* variances, *K* prior proportions, and $\binom{p}{2} = p(p-1)/2$ covariances to estimate.

LDA when $K > 2$

The linear discriminant nature of LDA still holds not only when $p > 1$, but also when $K > 2$ for that matter as well. A picture can be very illustrative:



Quadratic Discriminant Analysis (QDA)

Quadratic Discriminant Analysis (QDA)

A generalization to linear discriminant analysis is quadratic discriminant analysis (QDA).

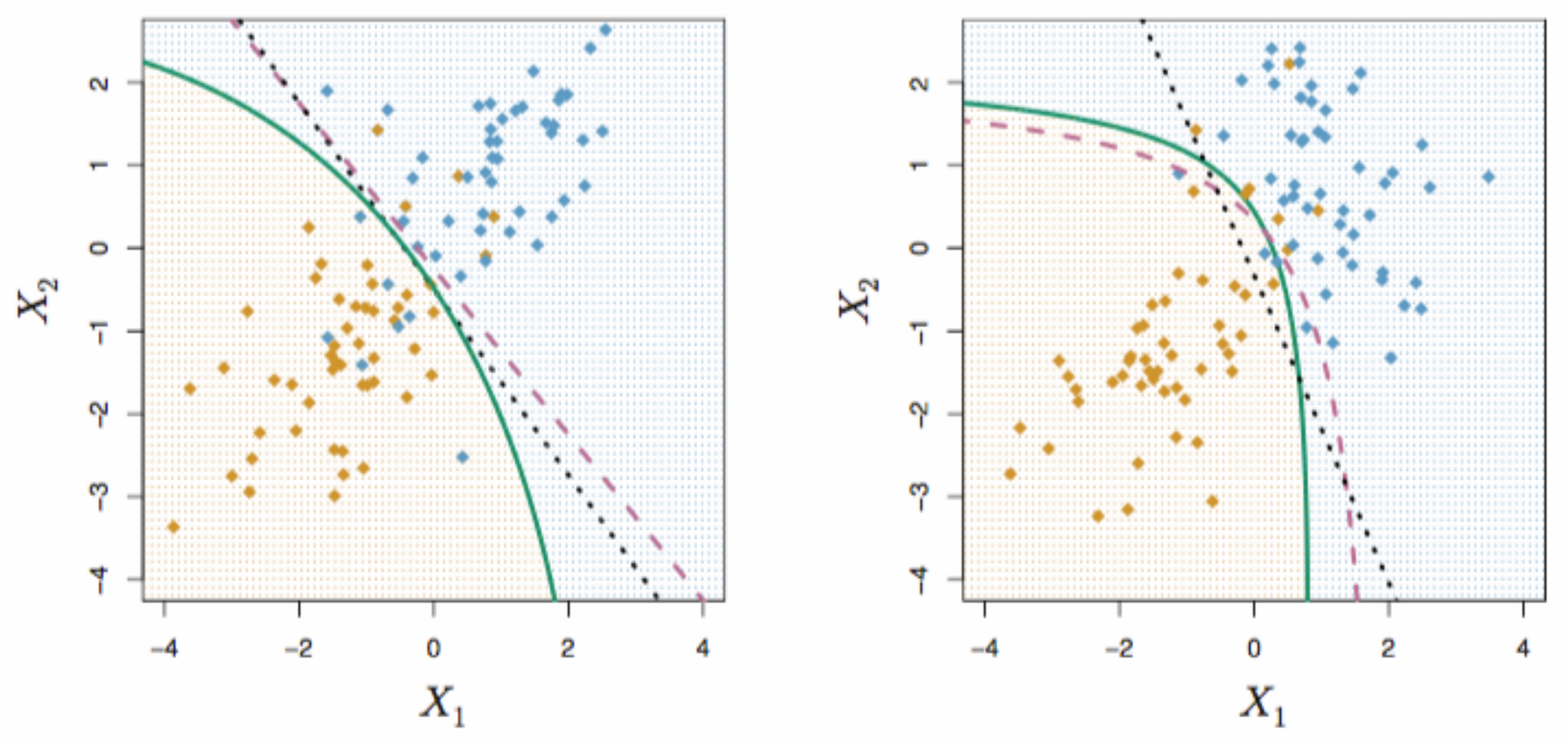
Why do you suppose the choice in name?

The implementation is just a slight variation on LDA. Instead of assuming the covariances of the MVN distributions within classes are equal, we instead allow them to be different.

This relaxation of an assumption completely changes the picture...

QDA in a picture

A picture can be very illustrative:



QDA (cont.)

When performing QDA, performing classification for an observation based on its predictors x is equivalent to maximizing the following over the *K* classes:

$$\delta_k(\vec{x}) = -\frac{1}{2}\vec{x}^T\Sigma_k^{-1}\vec{x} + \vec{x}^T\Sigma_k^{-1}\vec{\mu}_k - \frac{1}{2}\vec{\mu}_k^T\Sigma_k^{-1}\vec{\mu}_k - \frac{1}{2}\log|\Sigma_k| + \log\pi_k$$

Notice the `quadratic form' of this expression. Hence the name QDA.

Now how many parameters are there to be estimated?

There are *pK* means, *pK* variances, *K* prior proportions, and $\binom{p}{2}K=(p(p-1)/2)K$ covariances to estimate. This could slow us down very much if *K* is large...

Discriminant Analysis in Python

LDA is already implemented in Python via the `sklearn.discriminant_analysis` package through the `LinearDiscriminantAnalysis` function.

QDA is in the same package and is the `QuadraticDiscriminantAnalysis` function.

It's very easy to use. Let's see how this works

Discriminant Analysis in Python (cont.)

```
#read in the GSS data
gssdata = pd.read_csv("gsspartyid.csv")
print(gssdata[1:5])
```

```
  politicalparty  income  educ  abortion  republican
1     Republican     906     6         0           1
2       Democrat    1373     6         0           0
3       Democrat    1941     4         0           0
4       Democrat     355     7         0           0
```

```
LDA = da.LinearDiscriminantAnalysis()
X = gssdata[["income","educ","abortion"]]
model_LDA = LDA.fit(X,gssdata['republican'])
print("Specificity is",np.mean(model_LDA.predict(X[gssdata['republican']==0])))
print("Sensitivity is",1-np.mean(model_LDA.predict(X[gssdata['republican']==1])))
print("False positive rate is",np.mean(gssdata['republican'][model_LDA.predict(X)==1]))
print("False negative rate is",1-np.mean(gssdata['republican'][model_LDA.predict(X)==0]))
```

```
Specificity is 0.388387096774
Sensitivity is 0.374060150376
False positive rate is 0.525236593059369
False negative rate is 0.7043090638930163
```

QDA vs. LDA

So both QDA and LDA take a similar approach to solving this classification problem: they use Bayes' rule to flip the conditional probability statement and assume observations within each class are multivariate Normal (MVN) distributed.

QDA differs in that it does not assume a common covariance across classes for these MVNs. What advantage does this have? What disadvantage does this have?

QDA vs. LDA (cont.)

So generally speaking, when should QDA be used over LDA? LDA over QDA?

The extra covariance parameters that need to be estimated in QDA not only slow us down, but also allow for another opportunity for overfitting. Thus if your training set is small, LDA should perform better for ***out-of-sample prediction***, aka, predicting future observations (how do we mimic this process?)

Comparison of Classification Methods (so far)

Quadratic Discriminant Analysis (QDA)

We have seen 3 major methods for doing classification:

- Logistic Regression
- *k*-NN
- Discriminant Analysis (LDA and QDA)

For a specific problem, which approach should be used?

Well of course, it depends on the nature of the data. So how should we decide?

Visualize the data!

Six Classification Models We'll Compare

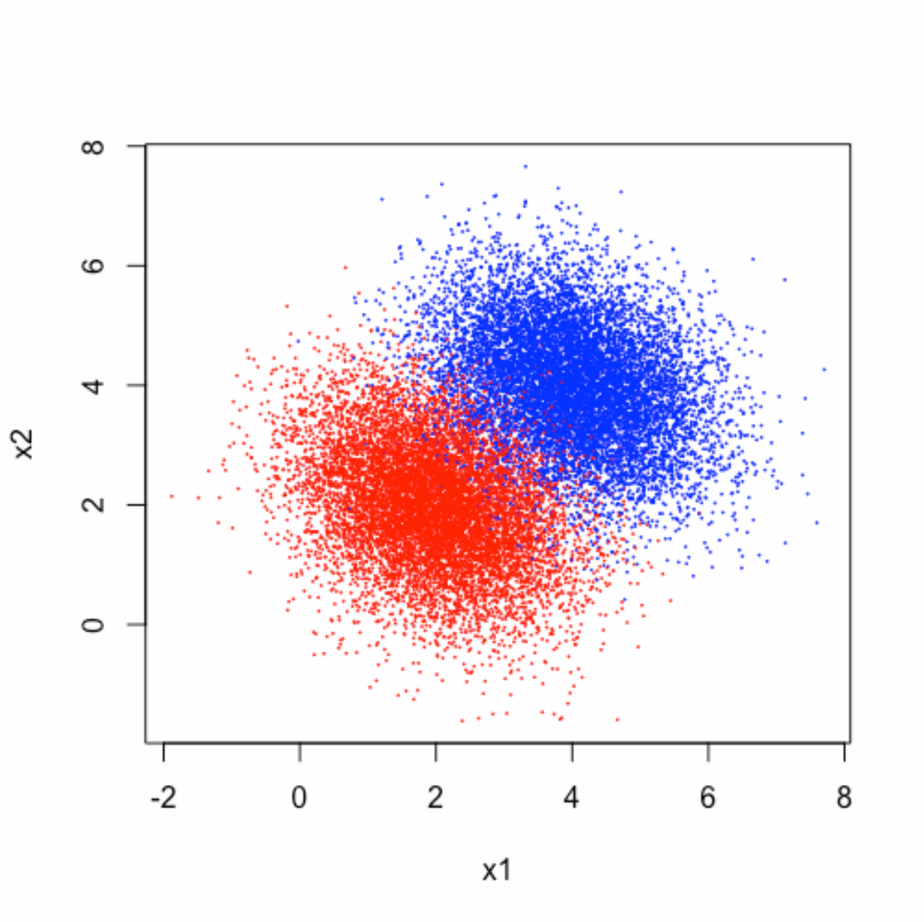
Let's investigate which method will work the best (as measured by lowest overall classification error rate), by considering 6 different models for 4 different data sets (each data set as a pair of predictors...you can think of them as the first 2 PCA components…to come later in the lecture). The 6 models to consider are:

- A logistic regression with only 'linear' main effects}
- A logistic regression with only 'linear' and 'quadratic' effects}
- LDA
- QDA
- *k*-NN where $k = 3$
- *k*-NN where $k = 25$

What else will also be important to measure (besides error rate)?

Which method should perform better? #1

n = 20,000, p = 2, K = 2, $\pi_1 = \pi_2 =$



method	misclass rate	run time (ms)
logit1	0.04410	417.95
logit2	0.04405	229.71
lda	0.04425	50.63
qda	0.04410	49.08
knn3	0.05225	1856.11
knn25	0.04500	2166.57

Notice anything fishy about our answers? What did Kevin do? What should he have done?

Easy to implement in Python

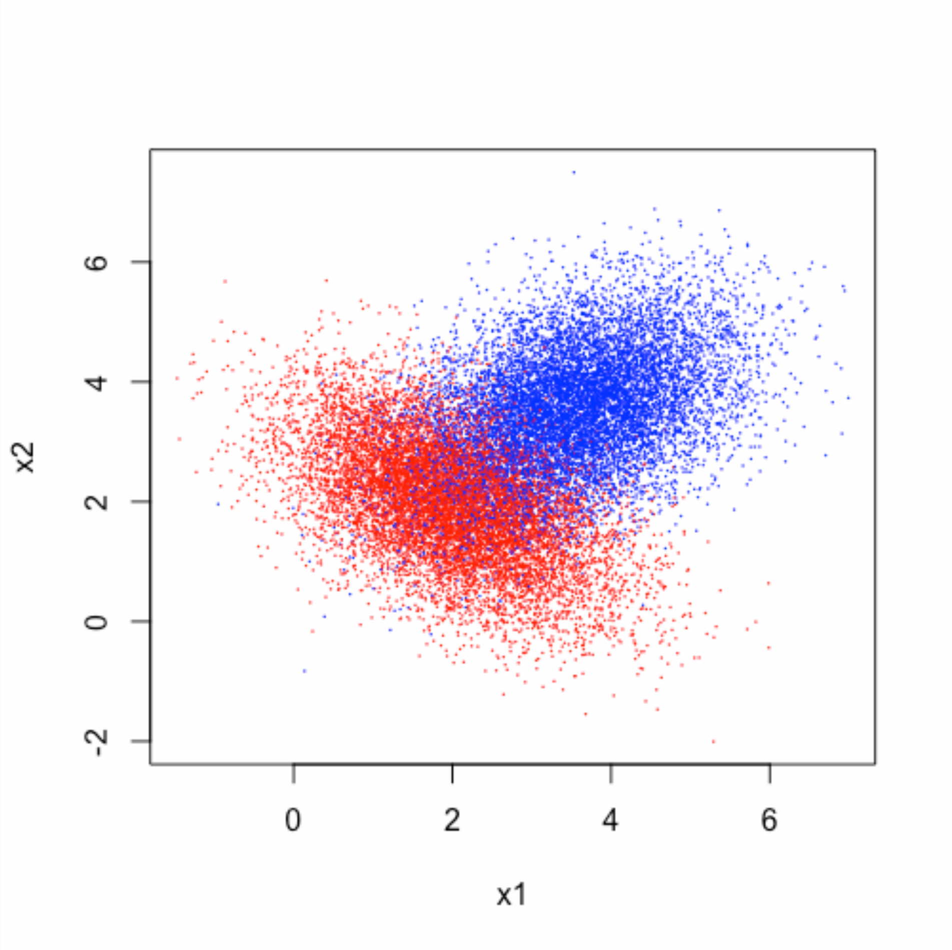
```
lda = da.LinearDiscriminantAnalysis()
qda = da.QuadraticDiscriminantAnalysis()
lda.fit(X,y)
qda.fit(X,y)
logit2 = sk.linear_model.LogisticRegression(C = 1000000)
logit1 = sk.linear_model.LogisticRegression(C = 1000000)
logit1.fit(X,y)
logit2.fit(X2,y)

print("Overall misclassification rate of Logit1 is",(1-logit1.score(X,y)))
print("Overall misclassification rate of Logit2 is",(1-logit2.score(X2,y)))
print("Overall misclassification rate of LDA is",(1-lda.score(X,y)))
print("Overall misclassification rate of QDA is",(1-qda.score(X,y)))
```

```
Overall misclassification rate of Logit1 is 0.0441
Overall misclassification rate of Logit2 is 0.0441
Overall misclassification rate of LDA is 0.04425
Overall misclassification rate of QDA is 0.0441
```

Which method should perform better? #2

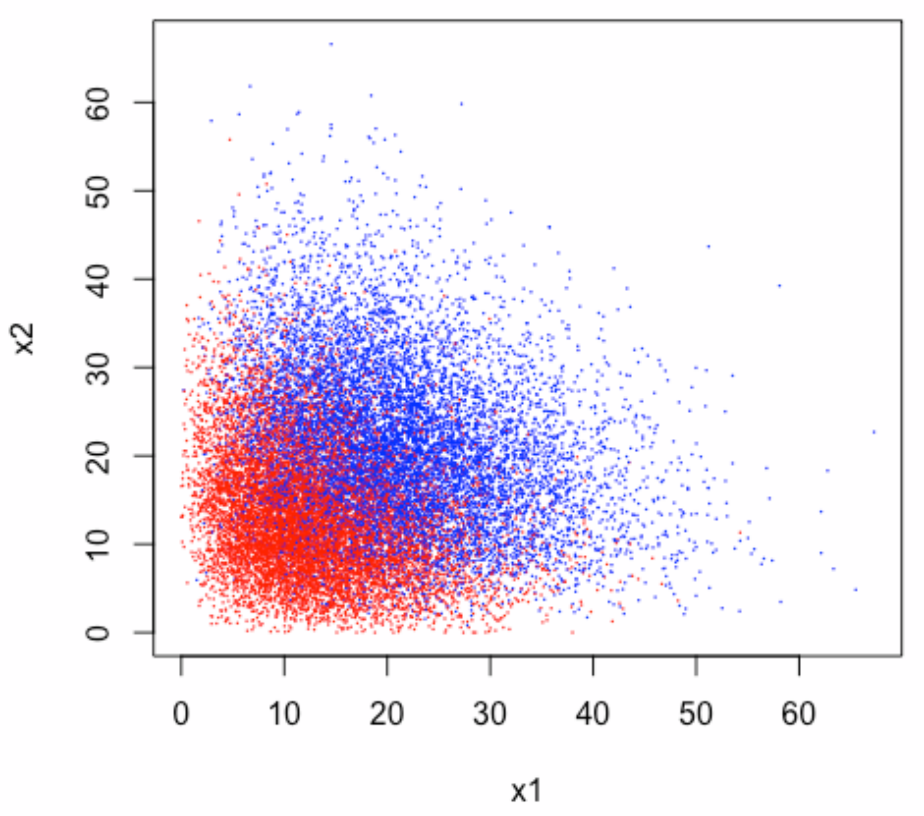
n = 20,000, p = 2, K = 2, $\pi_1 = \pi_2 = 0.5$



method	misclass rate	run time (ms)
logit1	0.12230	169.53
logit2	0.11860	196.42
lda	0.12215	47.93
qda	0.11445	47.03
knn3	0.14380	1861.90
knn25	0.12015	2223.13

Which method should perform better? #3

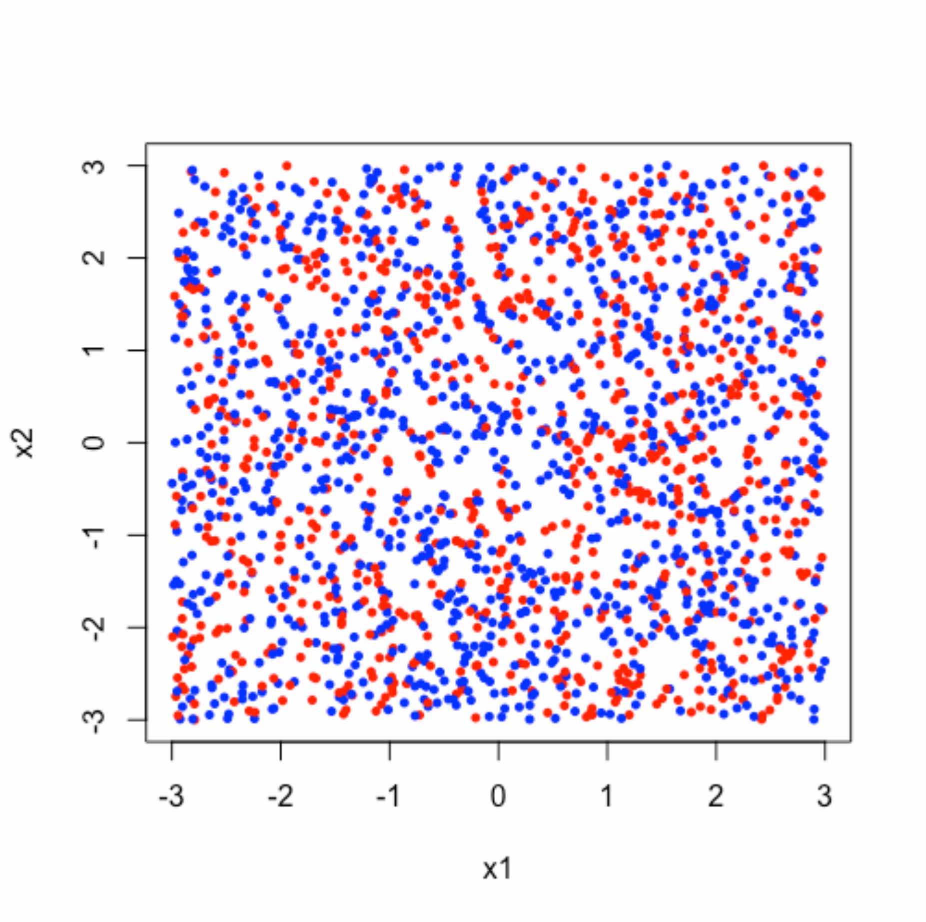
$n = 20{,}000$, $p = 2$, $K = 2$, $\pi_1 = \pi_2 = 0.5$



method	misclass rate	run time (ms)
logit1	0.20260	1234.35
logit2	0.19535	192.99
lda	0.21450	49.08
qda	0.20320	60.61
knn3	0.23300	1869.44
knn25	0.20270	2166.77

Which method should perform better? #4

n = 20,000, p = 2, K = 2, $\pi_1 = \pi_2 = 0.5$



method	misclass rate	run time (ms)
logit1	0.45690	1181.44
logit2	0.37880	147.95
lda	0.45770	51.06
qda	0.40705	44.04
knn3	0.34820	1835.42
knn25	0.30655	2126.38

Summary of Results

Generally speaking:

- LDA outperforms Logistic Regression if the distribution of predictors is reasonably MVN (with constant covariance).
- QDA outperforms LDA if the covariances are not the same in the groups.
- k-NN outperforms the others if the decision boundary is extremely non-linear.
- Of course, we can always adapt our models (logistic and LDA/QDA) to include polynomial terms, interaction terms, etc... to improve classification (watch out for overfitting!)
- In order of computational speed (generally speaking, it depends on *K*, *p*, and *n* of course):

LDA > QDA > Logistic > *k*-NN