BIOST 546: Machine Learning for Biomedical Big Data

Ali Shojaie

Lecture 5: Classification for Biomedical Big Data - Part II
Spring 2017

Recap

- Classification using linear regression
- Fundamentals of classification, Bayes classifier and Bayes error rate
- Logistic regression, and regularization
- Batch effects and pitfalls of classification

Today's Class

- LDA & QDA
- SVM
- KNN

Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA)

- Recall that the Bayes classifier suggests assigning observation i to class h for which
$$p_h(x) = P(Y = h \mid X = x)$$
is the largest.

Linear Discriminant Analysis (LDA)

- Recall that the Bayes classifier suggests assigning observation i to class h for which

$$p_h(x) = P(Y = h \mid X = x)$$

  is the largest.

- However, as we discussed, calculating $p_h(x)$ is in general difficult!

Linear Discriminant Analysis (LDA)

- Recall that the Bayes classifier suggests assigning observation i to class h for which
$$p_h(x) = P(Y = h \mid X = x)$$
is the largest.
- However, as we discussed, calculating $p_h(x)$ is in general difficult!
- One approach for making this problem easier is to use the Bayes theorem to write
$$p_h(x) = P(Y = h \mid X = x) = \frac{\pi_h f_h(x)}{\sum_{l=1}^{H} \pi_l f_l(x)}$$

Linear Discriminant Analysis (LDA)

- Recall that the Bayes classifier suggests assigning observation i to class h for which
$$p_h(x) = P(Y = h \mid X = x)$$
is the largest.
- However, as we discussed, calculating $p_h(x)$ is in general difficult!
- One approach for making this problem easier is to use the Bayes theorem to write
$$p_h(x) = P(Y = h \mid X = x) = \frac{\pi_h f_h(x)}{\sum_{l=1}^{H} \pi_l f_l(x)}$$
Here
  - $\pi_h = P(Y = h)$ is the prior probability
  - $f_h(x) = P(X = x \mid Y = h)$ is the density function of X for an observation coming from class h
  - $p_h(x)$ is the posterior density of y given the data x

Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA)

To use the Bayes theorem

$$p_h(x) = P(Y = h \mid X = x) = \frac{\pi_h f_h(x)}{\sum_{l=1}^{H} \pi_l f_l(x)}$$

we need to estimate π_h and f_h:

Linear Discriminant Analysis (LDA)

To use the Bayes theorem

$$p_h(x) = P(Y = h \mid X = x) = \frac{\pi_h f_h(x)}{\sum_{l=1}^{H} \pi_l f_l(x)}$$

we need to estimate π_h and f_h:

- π_h is often easy to estimate: if we have a random sample, $\hat{\pi}_h = 1/n \sum_i I(Y_i = h)$
- However, estimating f_h can be very challenging, especially in high dimensions
- One solution is to assume a parametric form for f_h

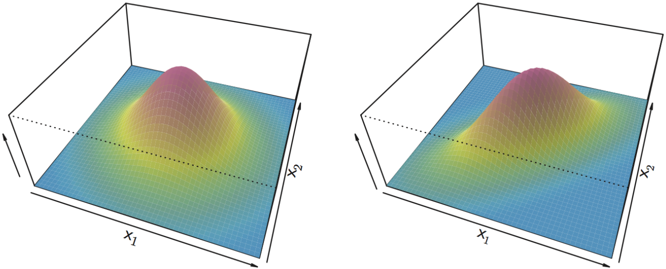
Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA)

- In LDA, we assume $f_h(x)$ is the Gaussian density, $N(\mu_h, \Sigma_h)$

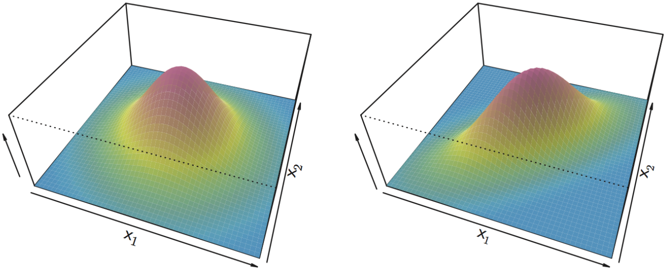
Linear Discriminant Analysis (LDA)

- In LDA, we assume $f_h(x)$ is the Gaussian density, $N(\mu_h, \Sigma_h)$



Linear Discriminant Analysis (LDA)

- In LDA, we assume $f_h(x)$ is the Gaussian density, $N(\mu_h, \Sigma_h)$

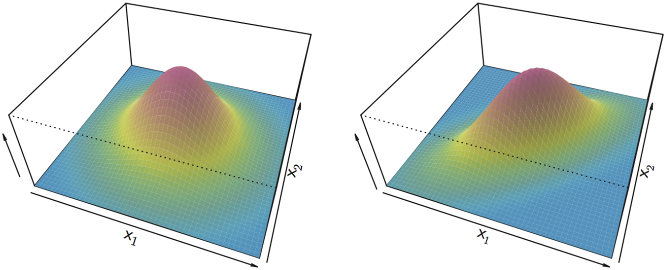


- This is equivalent to assuming that our data is generated from a mixture of H Gaussian distributions

$$X \sim \sum_{h=1}^{H} \pi_h \, \phi(\mu_h, \Sigma_h)$$

Linear Discriminant Analysis (LDA)

- In LDA, we assume $f_h(x)$ is the Gaussian density, $N(\mu_h, \Sigma_h)$



- This is equivalent to assuming that our data is generated from a mixture of H Gaussian distributions

$$X \sim \sum_{h=1}^{H} \pi_h \phi(\mu_h, \Sigma_h)$$

- LDA was developed by R. A. Fisher, and has some interesting connections to analysis of variance

LDA Assumptions

LDA Assumptions

- LDA further assumes that $\Sigma_h = \Sigma \; \forall h$: all classes share a common variance

LDA Assumptions

- LDA further assumes that $\Sigma_h = \Sigma \, \forall h$: all classes share a common variance
- With this assumption, the decision boundary only depends on means μ_h and is always linear (hence the name)

LDA Assumptions

- LDA further assumes that $\Sigma_h = \Sigma \; \forall h$: all classes share a common variance
- With this assumption, the decision boundary only depends on means μ_h and is always linear (hence the name)
- In most cases, it is also assumed that Σ is diagonal: i.e. no correlation among covariates!!

LDA Assumptions

- LDA further assumes that $\Sigma_h = \Sigma \, \forall h$: all classes share a common variance
- With this assumption, the decision boundary only depends on means μ_h and is always linear (hence the name)
- In most cases, it is also assumed that Σ is diagonal: i.e. no correlation among covariates!!
- This latter assumption can be relaxed. However, in high dimensional settings, estimation of Σ is difficult ($\binom{p}{2}$ parameters to estimate!), and there is often not much improvements from this in HD.

LDA for $p = 1$

LDA for $p = 1$

- To start, suppose that $p = 1$

LDA for $p = 1$

- To start, suppose that $p = 1$
- Following the main assumption of LDA, suppose $\sigma_h = \sigma$

LDA for $p = 1$

- To start, suppose that $p = 1$
- Following the main assumption of LDA, suppose $\sigma_h = \sigma$
- We can then show that an observation with covariate x is classified to the class h with the largest value of

$$x\frac{\mu_h}{\sigma^2} - \frac{\mu_h^2}{2\sigma^2} + \log(\pi_h)$$

LDA for $p = 1$

- To start, suppose that $p = 1$
- Following the main assumption of LDA, suppose $\sigma_h = \sigma$
- We can then show that an observation with covariate x is classified to the class h with the largest value of

$$x\frac{\mu_h}{\sigma^2} - \frac{\mu_h^2}{2\sigma^2} + \log(\pi_h)$$

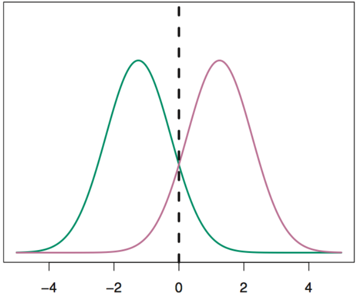
- If we further assume that $H = 2$ and $\pi_1 = \pi_2 = 0.5$, then the Bayes decision boundary is

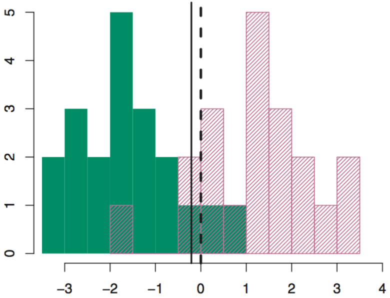
$$\frac{\mu_1 + \mu_2}{2}$$

which is clearly linear!

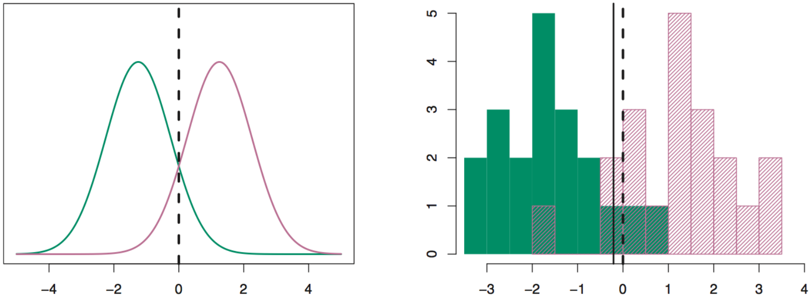
LDA for $p = 1$

LDA for $p = 1$





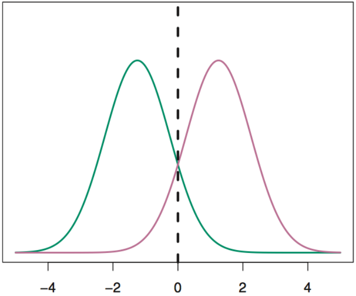
LDA for $p = 1$

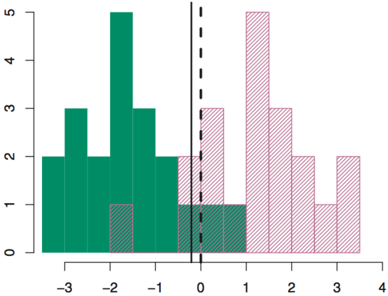


- To make this work, we need to estimate the parameters. The ML estimates are given by $\hat{\pi}_h = n_h/n$ and

$$\hat{\mu}_h = \frac{1}{n_h} \sum_{i:y_i=h} x_i \qquad \hat{\sigma}^2{}_h = \frac{1}{n-H} \sum_{h=1}^{H} \sum_{i:y_i=h} (x_i - \hat{\mu}_h)^2$$

LDA for $p = 1$





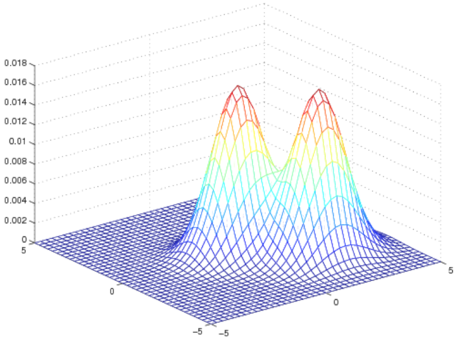
- To make this work, we need to estimate the parameters. The ML estimates are given by $\hat{\pi}_h = n_h/n$ and

$$\hat{\mu}_h = \frac{1}{n_h} \sum_{i:y_i=h} x_i \qquad \hat{\sigma}^2_h = \frac{1}{n-H} \sum_{h=1}^{H} \sum_{i:y_i=h} (x_i - \hat{\mu}_h)^2$$

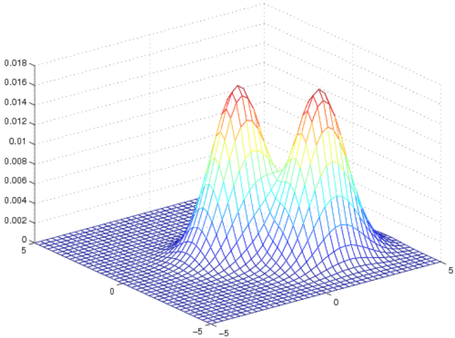
- The picture is very similar if $H > 2$...or if $p > 1$

LDA for $p > 1$

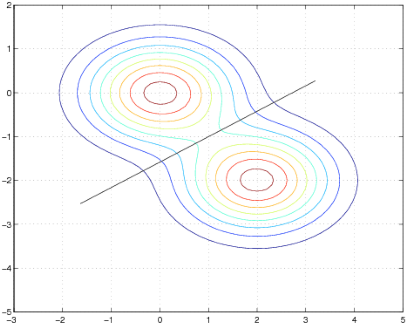
LDA for $p > 1$



LDA for $p > 1$



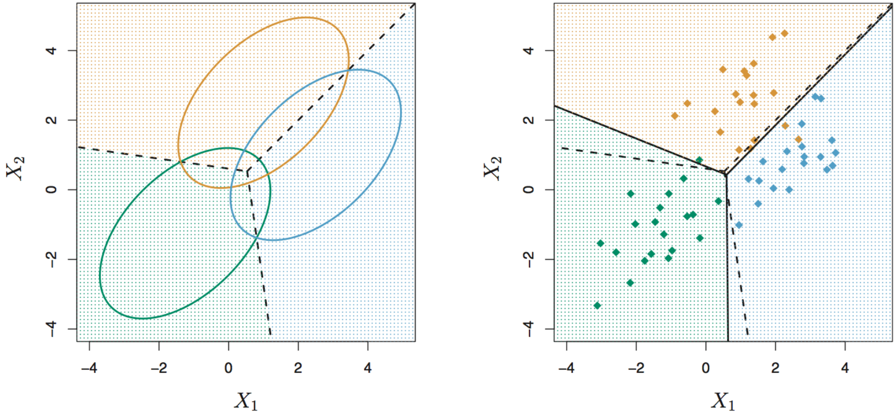
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
5
0
−5
−5
0
5



2
1
0
−1
−2
−3
−4
−5
−3
−2
−1
0
1
2
3
4
5

LDA for $p > 1$

LDA for $p > 1$



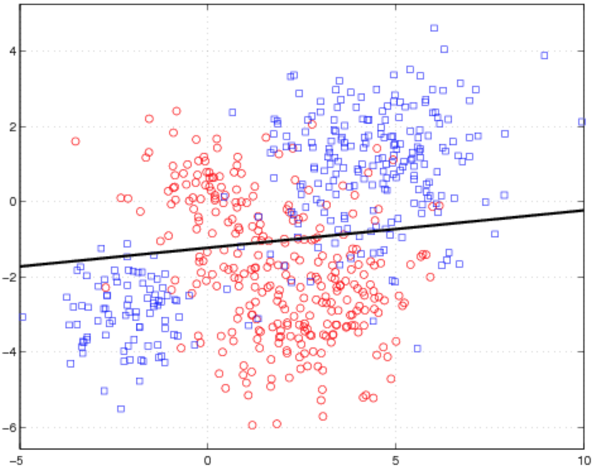
When would LDA fail?

When would LDA fail?

Of course, like any other model-based approach, LDA only works if its underlying assumptions are (almost) valid

When would LDA fail?

Of course, like any other model-based approach, LDA only works if its underlying assumptions are (almost) valid



Quadratic Discriminant Analysis (QDA)

Quadratic Discriminant Analysis (QDA)

- As we discussed, one of the main limitations of LDA is the assumption that $\Sigma_h = \Sigma \, \forall h$, which results in linear decision boundaries

Quadratic Discriminant Analysis (QDA)

- As we discussed, one of the main limitations of LDA is the assumption that $\Sigma_h = \Sigma\ \forall h$, which results in linear decision boundaries
- This assumption is relaxed in QDA, allowing for non-linear (quadratic) decision boundaries. In this case, an observation with covariates x is classified to group c with the largest value of

$$-0.5x^{\intercal}\Sigma_h^{-1}x + x^{\intercal}\Sigma_h^{-1}\mu_h - 0.5{\mu_h}^{\intercal}\Sigma_h^{-1}\mu_h + \log\pi_h$$

Quadratic Discriminant Analysis (QDA)

- As we discussed, one of the main limitations of LDA is the assumption that $\Sigma_h = \Sigma \, \forall h$, which results in linear decision boundaries
- This assumption is relaxed in QDA, allowing for non-linear (quadratic) decision boundaries. In this case, an observation with covariates x is classified to group c with the largest value of

$$-0.5x^{\mathsf{T}}\Sigma_h^{-1}x + x^{\mathsf{T}}\Sigma_h^{-1}\mu_h - 0.5\mu_h{}^{\mathsf{T}}\Sigma_h^{-1}\mu_h + \log \pi_h$$

- Note that we are still assuming that observations are from a mixture of Gaussian distributions
- QDA is much more flexible than LDA, but we need to estimate $Hp(p+1)$ parameters, which is a lot in high dimensions
- To make this a bit simpler, again we often assume that Σ_h is diagonal

LDA vs QDA

LDA vs QDA

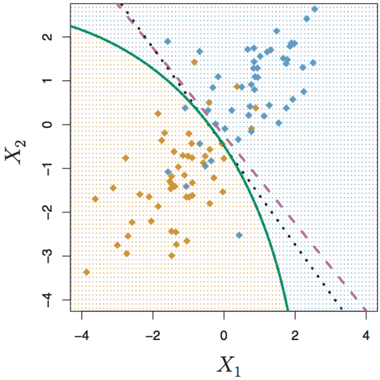
- LDA is rather restrictive; on the other hand, QDA is quite flexible

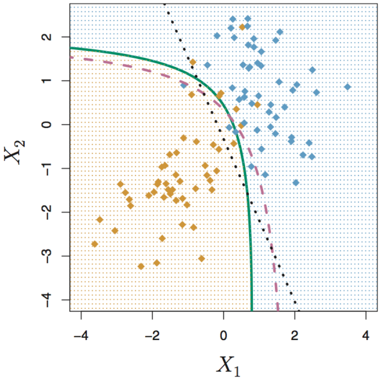
LDA vs QDA

- LDA is rather restrictive; on the other hand, QDA is quite flexible
- However, this flexibility comes at the price of more complex models (as always!), which brings up the bias-variance tradeoff again...

LDA vs QDA

- LDA is rather restrictive; on the other hand, QDA is quite flexible
- However, this flexibility comes at the price of more complex models (as always!), which brings up the bias-variance tradeoff again...





K Nearest Neighbor Classifier

K Nearest Neighbor Classifier

- All of the methods discussed so far (logistic regression, LDA and QDA) rely on specific parametric assumptions about the distribution of the data

K Nearest Neighbor Classifier

- All of the methods discussed so far (logistic regression, LDA and QDA) rely on specific parametric assumptions about the distribution of the data
- These models work, if their underlying assumptions are (roughly) valid, but may break down miserably otherwise

K Nearest Neighbor Classifier

- All of the methods discussed so far (logistic regression, LDA and QDA) rely on specific parametric assumptions about the distribution of the data
- These models work, if their underlying assumptions are (roughly) valid, but may break down miserably otherwise
- The alternative is to avoid making parametric assumptions; and use nonparametric models

K Nearest Neighbor Classifier

- All of the methods discussed so far (logistic regression, LDA and QDA) rely on specific parametric assumptions about the distribution of the data
- These models work, if their underlying assumptions are (roughly) valid, but may break down miserably otherwise
- The alternative is to avoid making parametric assumptions; and use nonparametric models
- The simplest nonparametric approach for classification (also works for regression) is the K nearest neighbor classification method (KNN)

KNN Basics

KNN Basics

- As the name suggests, the simple idea behind KNN is to use the K nearest neighbor of each observation, based on the values of x, to predict its y value

$$p_j(X) = P(Y = j \mid X = x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j)$$

where N_0 is the neighborhood of x_0

KNN Basics

- As the name suggests, the simple idea behind KNN is to use the K nearest neighbor of each observation, based on the values of x, to predict its y value

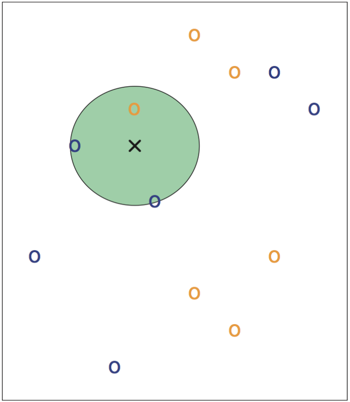
$$p_j(X) = P(Y = j \mid X = x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j)$$

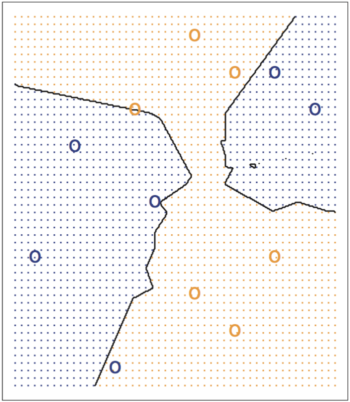
  where N_0 is the neighborhood of x_0

- We then apply the Bayes rule to classify x_0 to the class with largest probability

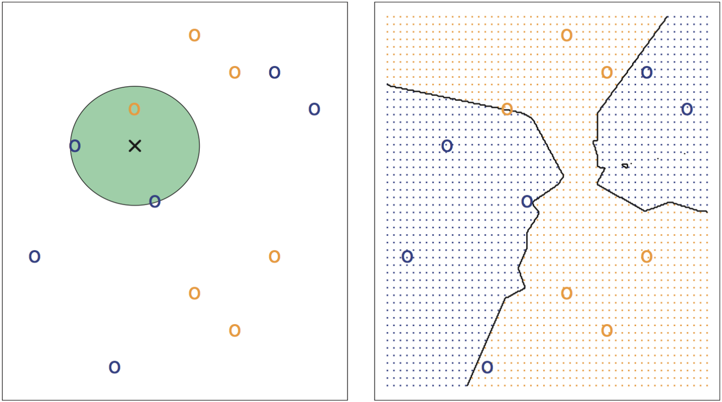
KNN Basics

KNN Basics



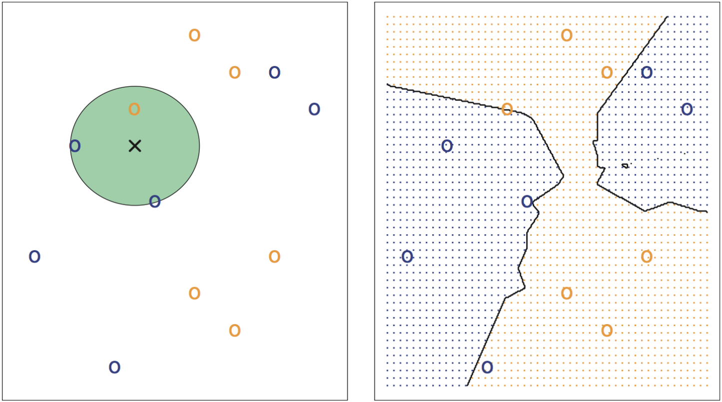


KNN Basics



- Small K: very flexible (read complex!) classifier $\Rightarrow$ low bias and high variance

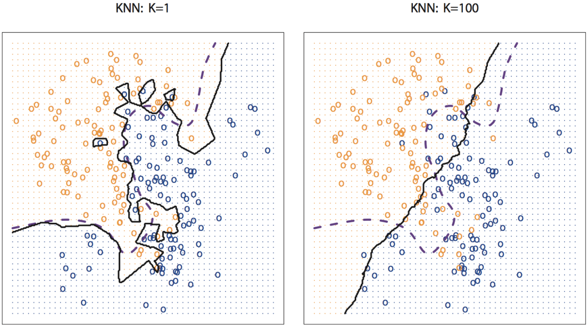
KNN Basics



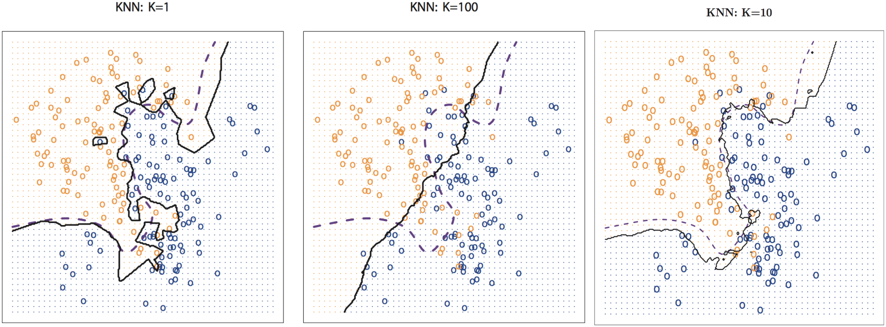
- Small K: very flexible (read complex!) classifier $\Rightarrow$ low bias and high variance
- Large K: less flexible classifier, and a smoother decision boundary $\Rightarrow$ high bias and low variance

Bias-Variance Tradeoff for KNN

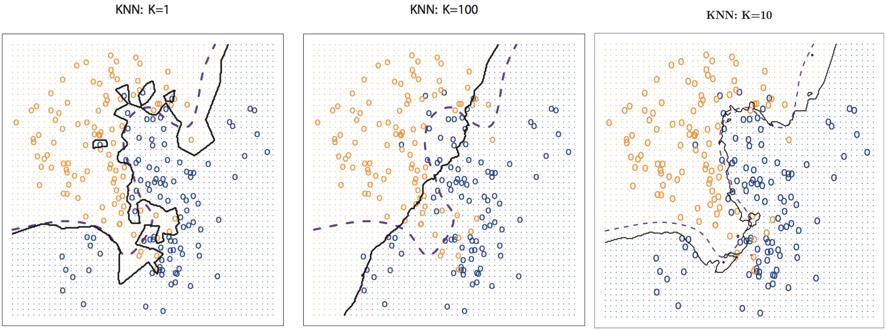
Bias-Variance Tradeoff for KNN



Bias-Variance Tradeoff for KNN



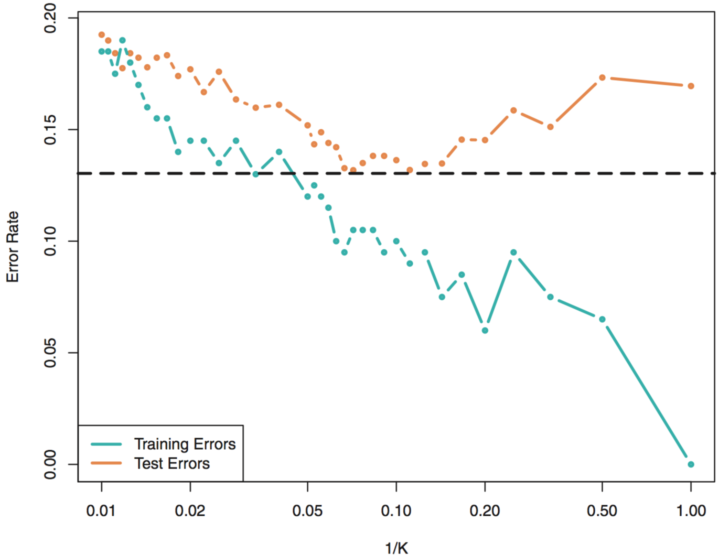
Bias-Variance Tradeoff for KNN



- We again have to face our beloved bias-variance tradeoff, and need to choose K that gives good test error

Bias-Variance Tradeoff for KNN

Bias-Variance Tradeoff for KNN



Support Vector Machines

- Developed in around 1995.

Support Vector Machines

- Developed in around 1995.
- Touted as “overcoming the curse of dimensionality", thought that is not automatic!

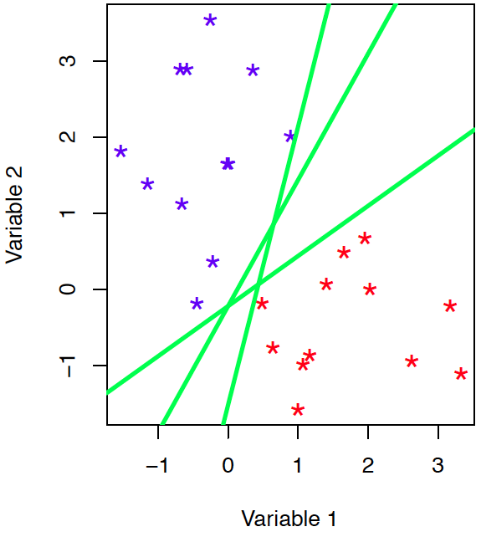
Support Vector Machines

- Developed in around 1995.
- Touted as “overcoming the curse of dimensionality", thought that is not automatic!
- Fundamentally and numerically very similar to logistic regression (more later).

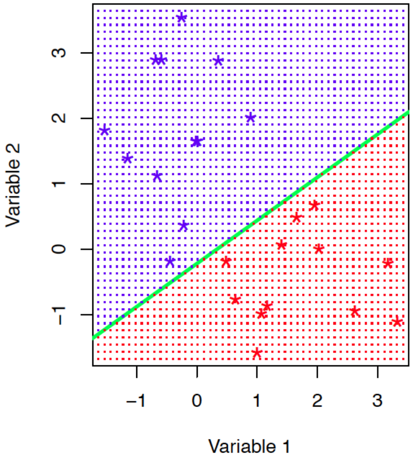
Support Vector Machines

- Developed in around 1995.
- Touted as “overcoming the curse of dimensionality", thought that is not automatic!
- Fundamentally and numerically very similar to logistic regression (more later).
- However, has a very different motivation and is a very nice idea.

Separating Hyperplane



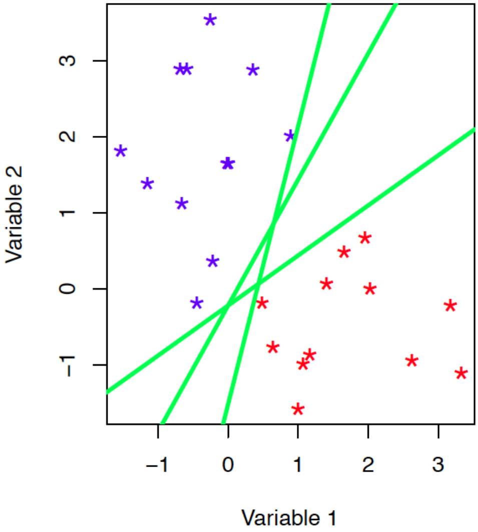
Classification via Separating Hyperplanes



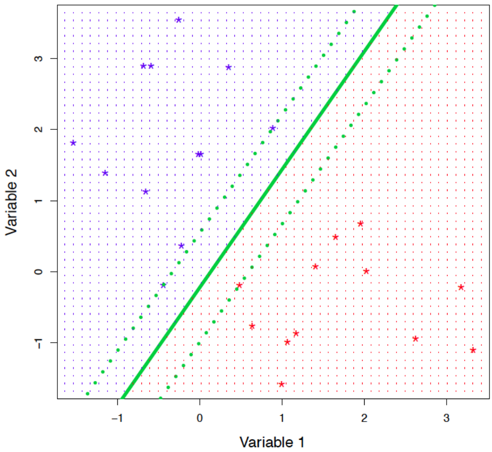
Blue class if $\beta_0 + \beta_1 X_1 + \beta_2 X_2 > c$; red class otherwise.

Which hyperplane?

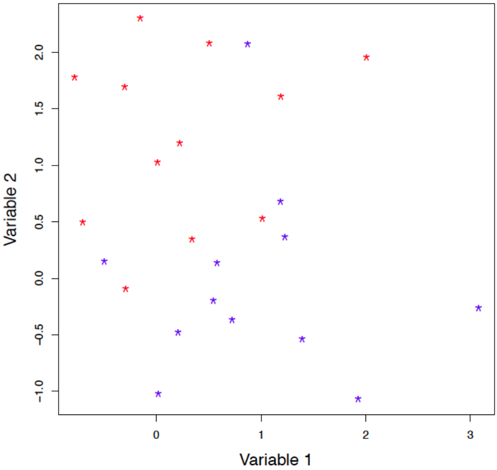
There are potentially many hyperplanes...



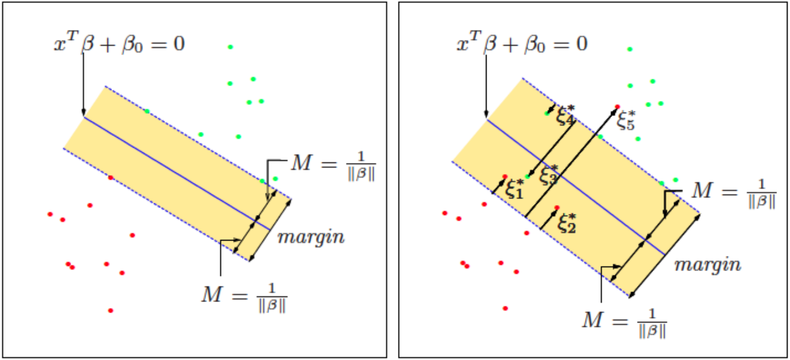
Maximally Separating Hyperplane



What if There is No Separating Hyperplane?



Support Vector Classifier: Allow for Violations



Support Vector Machine

- The support vector machine is just like the support vector classifier, but it elegantly allows for non-linear expansions of the variables: “non-linear kernels".

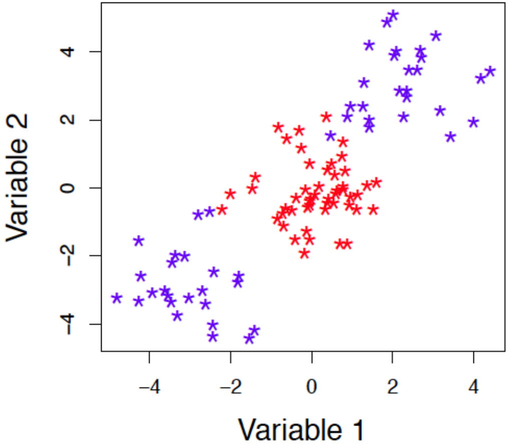
Support Vector Machine

- The support vector machine is just like the support vector classifier, but it elegantly allows for non-linear expansions of the variables: “non-linear kernels".
- However, linear regression, logistic regression, and other classical statistical approaches can also be applied to non-linear functions of the variables.

Support Vector Machine

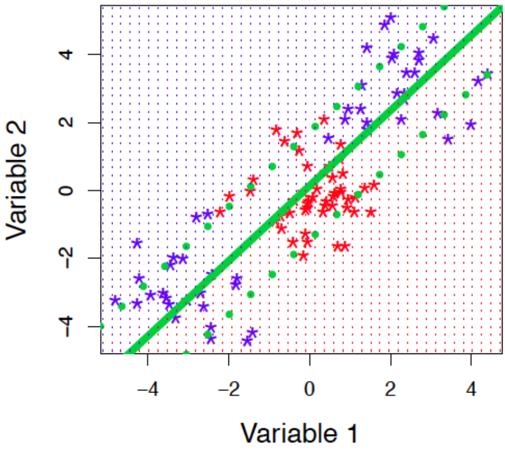
- The support vector machine is just like the support vector classifier, but it elegantly allows for non-linear expansions of the variables: “non-linear kernels".
- However, linear regression, logistic regression, and other classical statistical approaches can also be applied to non-linear functions of the variables.
- For historical reasons, SVMs are more frequently used with non-linear expansions as compared to other statistical approaches.

Non-Linear Class Structure



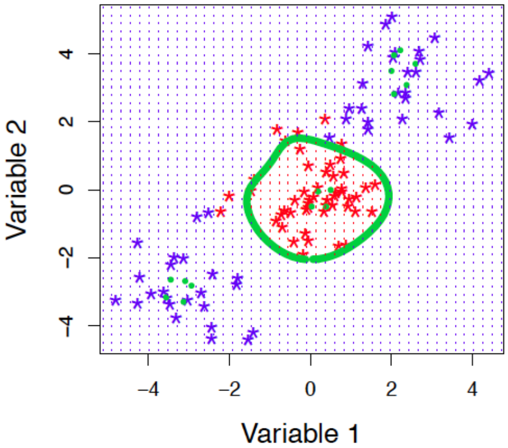
This will be hard for a linear classifier!

Try a Support Vector Classifier



Uh-oh!!

Support Vector Machine



Much Better.

Is A Non-Linear Kernel Better?

Is A Non-Linear Kernel Better?

- Yes, if the true decision boundary between the classes is non-linear, and you have enough observations (relative to the number of features) to accurately estimate the decision boundary.

Is A Non-Linear Kernel Better?

- Yes, if the true decision boundary between the classes is non-linear, and you have enough observations (relative to the number of features) to accurately estimate the decision boundary.
- No, if you are in a very high-dimensional setting such that estimating a non-linear decision boundary is hopeless.

SVM vs Other Classification Methods

- The main difference between SVM and other classification methods (e.g. logistic regression) is the loss function used to assess the “fit”:

$$\sum_{i=1}^{n} L(f(x_i), y_i)$$

SVM vs Other Classification Methods

- The main difference between SVM and other classification methods (e.g. logistic regression) is the loss function used to assess the "fit":

$$\sum_{i=1}^{n} L(f(x_i), y_i)$$

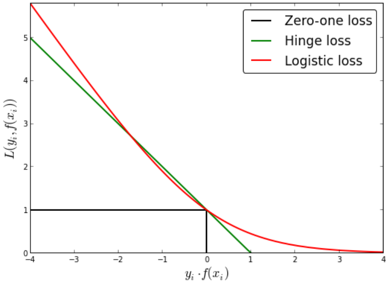
- Zero-one loss: $I(f(x_i) = y_i)$, where $I()$ is the indicator function. Not continuous, so hard to work with!!
- Hinge loss: $\max(0, 1 - f(x_i)y_i)$
- Logistic loss: $\log(1 + \exp f(x_i)y_i$

SVM vs Other Classification Methods

- The main difference between SVM and other classification methods (e.g. logistic regression) is the loss function used to assess the “fit”:

$$\sum_{i=1}^{n} L(f(x_i), y_i)$$

- Zero-one loss: $I(f(x_i) = y_i)$, where $I()$ is the indicator function. Not continuous, so hard to work with!!
- Hinge loss: $\max(0, 1 - f(x_i)y_i)$
- Logistic loss: $\log(1 + \exp f(x_i)y_i$



SVM vs Logistic Regression

SVM vs Logistic Regression

- Bottom Line: Support vector classifier and logistic regression aren't that different!

SVM vs Logistic Regression

- Bottom Line: Support vector classifier and logistic regression aren't that different!
- Neither they nor any other approach can automatically overcome the "curse of dimensionality" – unless carefully tuned.

SVM vs Logistic Regression

- Bottom Line: Support vector classifier and logistic regression aren't that different!
- Neither they nor any other approach can automatically overcome the "curse of dimensionality" – unless carefully tuned.
- The "kernel trick" makes things computationally easier, but it does not remove the danger of overfitting.

SVM vs Logistic Regression

- Bottom Line: Support vector classifier and logistic regression aren't that different!
- Neither they nor any other approach can automatically overcome the "curse of dimensionality" – unless carefully tuned.
- The "kernel trick" makes things computationally easier, but it does not remove the danger of overfitting.
- SVM uses a non-linear kernel... but could do that with logistic or linear regression too!

SVM vs Logistic Regression

- Bottom Line: Support vector classifier and logistic regression aren't that different!
- Neither they nor any other approach can automatically overcome the "curse of dimensionality" – unless carefully tuned.
- The "kernel trick" makes things computationally easier, but it does not remove the danger of overfitting.
- SVM uses a non-linear kernel... but could do that with logistic or linear regression too!
- One of the disadvantages of SVM (compared to some of the other methods) is that it does not provide a measure of uncertainty: cases are "classified" to belong to one of the two classes.

SVM vs Logistic Regression

- Bottom Line: Support vector classifier and logistic regression aren't that different!
- Neither they nor any other approach can automatically overcome the "curse of dimensionality" – unless carefully tuned.
- The "kernel trick" makes things computationally easier, but it does not remove the danger of overfitting.
- SVM uses a non-linear kernel... but could do that with logistic or linear regression too!
- One of the disadvantages of SVM (compared to some of the other methods) is that it does not provide a measure of uncertainty: cases are "classified" to belong to one of the two classes.
- Both SVM and logistic regression are not well-suited for problems with $H > 2$ categories; LDA/QDA may be a better choice in that setting.

In High Dimensions...

In High Dimensions...

- In SVMs, a tuning parameter controls the amount of flexibility of the classifier.

In High Dimensions...

- In SVMs, a tuning parameter controls the amount of flexibility of the classifier.
- This tuning parameter is like a ridge penalty, both mathematically and conceptually. The SVM decision rule involves all of the variables (the SVM problem can be written as a ridge problem but with the Hinge loss).

In High Dimensions...

- In SVMs, a tuning parameter controls the amount of flexibility of the classifier.
- This tuning parameter is like a ridge penalty, both mathematically and conceptually. The SVM decision rule involves all of the variables (the SVM problem can be written as a ridge problem but with the Hinge loss).
- Can get a sparse SVM using a lasso penalty; this yields a decision rule involving only a subset of the features.

In High Dimensions...

- In SVMs, a tuning parameter controls the amount of flexibility of the classifier.
- This tuning parameter is like a ridge penalty, both mathematically and conceptually. The SVM decision rule involves all of the variables (the SVM problem can be written as a ridge problem but with the Hinge loss).
- Can get a sparse SVM using a lasso penalty; this yields a decision rule involving only a subset of the features.
- Logistic regression and other classical statistical approaches could be used with non-linear expansions of features. But this makes high-dimensionality issues worse.

Assessing the Performance of Classifiers

- A binary classifier can have two types of errors: false positives and false negatives

Assessing the Performance of Classifiers

- A binary classifier can have two types of errors: false positives and false negatives

		Predicted Class $-$	Predicted Class $+$
True Class	$-$	TN	FP
True Class	$+$	FN	TP

Assessing the Performance of Classifiers

- A binary classifier can have two types of errors: false positives and false negatives

		Predicted Class $-$	Predicted Class $+$
True Class	$-$	TN	FP
True Class	$+$	FN	TP

- The above table is sometimes refried to as the confusion matrix

Assessing the Performance of Classifiers

- A binary classifier can have two types of errors: false positives and false negatives

		Predicted Class $-$	Predicted Class $+$
True Class	$-$	TN	FP
True Class	$+$	FN	TP

- The above table is sometimes refried to as the confusion matrix
- The misclassification error, which is what is used in Bayes classifier, is one way to combine FP and FN

Assessing the Performance of Classifiers

- There are a number of alternative (and related) measures to assess the performance of classifiers

Assessing the Performance of Classifiers

- There are a number of alternative (and related) measures to assess the performance of classifiers

Name	Definition	Other Names
FPR	FP/N	Type I error, $1-$ Specificity
TPR	TP/P	1 - Type II error, Sensetivity

Assessing the Performance of Classifiers

- There are a number of alternative (and related) measures to assess the performance of classifiers

Name	Definition	Other Names
FPR	FP/N	Type I error, $1-$ Specificity
TPR	TP/P	1 - Type II error, Sensetivity

- By default, all classification methods assume that all errors have the same “cost”; in particular, that FP and FN are equally costly

Assessing the Performance of Classifiers

- There are a number of alternative (and related) measures to assess the performance of classifiers

Name	Definition	Other Names
FPR	FP/N	Type I error, $1-$ Specificity
TPR	TP/P	1 - Type II error, Sensetivity

- By default, all classification methods assume that all errors have the same “cost”; in particular, that FP and FN are equally costly
- However, in many applications (e.g. in cancer diagnostics) it may be more costly to have FN than FP

Assessing the Performance of Classifiers

- There are a number of alternative (and related) measures to assess the performance of classifiers

Name	Definition	Other Names
FPR	FP/N	Type I error, $1-$ Specificity
TPR	TP/P	1 - Type II error, Sensetivity

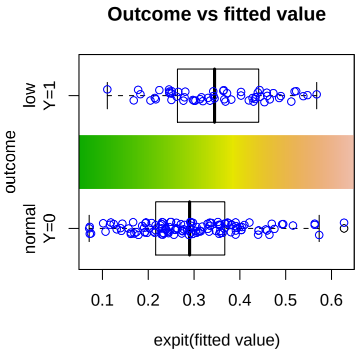
- By default, all classification methods assume that all errors have the same "cost"; in particular, that FP and FN are equally costly
- However, in many applications (e.g. in cancer diagnostics) it may be more costly to have FN than FP
- We can obtain different classifiers by changing the "cutoff"...the ROC plot measures the performance of classifiers

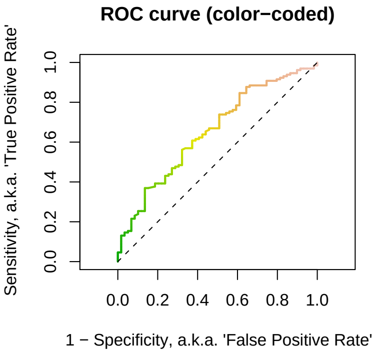
ROC

- The Receiver Operating Characteristic (ROC) curve summarizes the performance of a binary classifier over a range of decisions

ROC

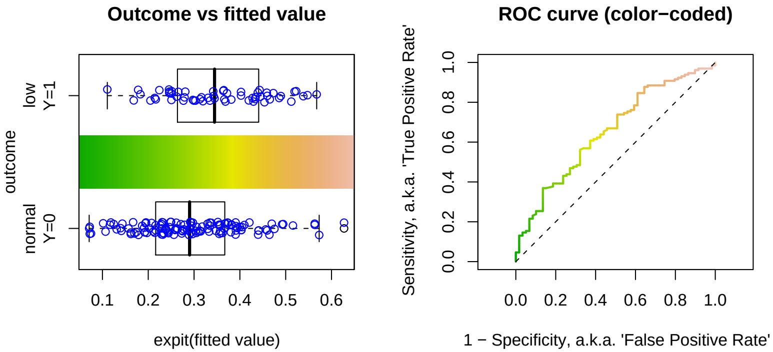
- The Receiver Operating Characteristic (ROC) curve summarizes the performance of a binary classifier over a range of decisions





ROC

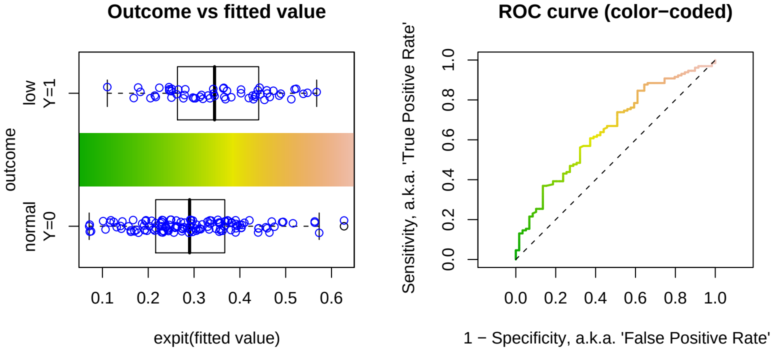
- The Receiver Operating Characteristic (ROC) curve summarizes the performance of a binary classifier over a range of decisions



- Each point on the ROC curve corresponds to a single decision (i.e. cutoff)

ROC

- The Receiver Operating Characteristic (ROC) curve summarizes the performance of a binary classifier over a range of decisions



- Each point on the ROC curve corresponds to a single decision (i.e. cutoff)
- The area under the ROC curve (AUC) measures the overall performance of a classifier (over the range of cutoffs)

Next Lecture

- Tree-based methods