

Models for classification

Nonlinear methods

A quick recap

Last lectures, we did...

- **Models for classification:** Linear discriminant analysis (LDA)
 - Decision theory for classification tells us that we need to know the posterior $P(Y|X)$ for optimal classification.
 - Let $f_k(x)$ a class-conditional density function of X in $Y = k$.
 - Let π_k be the prior probability of class k , with $\sum_{k=1}^K \pi_k = 1$.
 - The application of the Bayes' theorem gives:

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

A quick recap

Last lectures, we did...

- **Models for classification:** Linear discriminant analysis (LDA)
 - Suppose that we model each class density as multivariate Gaussian:

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

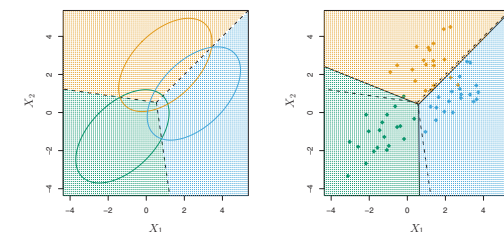
- LDA arises in the special case when we assume that the classes have a common covariance matrix: $\Sigma_k = \Sigma \forall k$.
- Comparing the classes k and l , the log-ratio is an equation linear in x :

$$\begin{aligned} \log \frac{P(Y = k|X = x)}{P(Y = l|X = x)} &= \log \frac{f_k(x)}{f_l(x)} + \log \frac{\pi_k}{\pi_l} \\ &= \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\mu_k + \mu_l)^T \Sigma^{-1} (\mu_k - \mu_l) + x^T \Sigma^{-1} (\mu_k - \mu_l) \end{aligned}$$

A quick recap

Last lectures, we did...

- **Models for classification:** Linear discriminant analysis (LDA)
 - The linear log-odds function implies that the decision boundary between classes k and l is linear in x ; in p dimension hyperplane.
 - If we divide \mathbb{R}^p into regions that are classified as class 1, class 2, ..., these regions will be separated by hyperplanes.
 - In the case $p = 2$



A quick recap

Last lectures, we did...

► Models for classification: Linear discriminant analysis (LDA)

- The linear discriminant functions give the decision rule:

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

- We estimate the parameters of the Gaussians using our training data:

$$\hat{\pi}_k = N_k / N, \quad \text{where } N_k \text{ is the number of the class-}k \text{ observations}$$

$$\hat{\mu}_k = \sum_{i: y_i = k} x_i / N_k$$

$$\hat{\Sigma} = \sum_{k=1}^K \sum_{i: y_i = k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T / (N - K)$$

A quick recap

Last lectures, we did...

► Models for classification: Linear discriminant analysis (LDA)

- For two classes, the LDA rule classifies to class 2 if:

$$x^T \Sigma^{-1} (\hat{\mu}_1 - \hat{\mu}_2) > \frac{1}{2} \hat{\mu}_2^T \Sigma^{-1} \hat{\mu}_2 - \frac{1}{2} \hat{\mu}_1^T \Sigma^{-1} \hat{\mu}_1 + \log(N_1 / N) - \log(N_2 / N)$$

- Otherwise, we classify it as class 1.

A quick recap

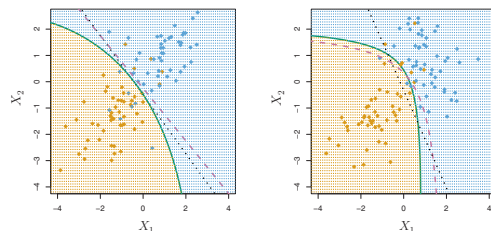
Last lectures, we did...

► Models for classification: Quadratic discriminant analysis (QDA)

- If the Σ_k are not assumed to be equal, we get quadratic discriminant functions:

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

- The decision boundaries between each pair of classes k and l is described by a quadratic equation.



A quick recap

Last lectures, we did...

► Models for classification: Linear and Quadratic discriminant analysis

- The estimates for QDA are similar to those for LDA, except that separate covariance matrices must be estimated for each class.
- When p is large, it means a dramatic increase in the parameters.
- Since the decision boundaries are functions of the parameters of the densities, counting the number of parameters must be done with care.
- For LDA, there are $(K - 1) \times (p + 1)$.
- For QDA, there are $(K - 1) \times \{p(p + 3)/2 + 1\}$.



Today's goal

Today, we going to do...

► Nonlinear models for classification

- ~ K-nearest neighbors
- ~ Examples in R
- Introduce Homework 3

Reading list

-  Max Kuhn and Kjell Johnson. *Applied Predictive Modeling*, Springer (2014)
-  Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. *An Introduction to Statistical Learning with Applications in R*, Springer (2017)

K-Nearest neighbors

Ideally, we would always like to predict qualitative responses using the Bayes classifier.

- For real data, we **do not know the conditional distribution of Y given X** , and so computing the Bayes classifier is impossible.
- The Bayes classifier serves as an unattainable gold standard against which to compare other methods.
- Many approaches attempt to estimate the conditional distribution of Y given X , and then classify a given observation to the class with highest estimated probability.

One such method is the **K-nearest neighbors (KNN) classifier**.

- KNN takes a completely different approach from the classifiers seen so far.
- KNN is a completely non-parametric approach: no assumptions are made about the shape of the decision boundary.
- KNN does not tell us which predictors are important.

K-Nearest neighbors

Given a positive integer K and a test observation x_0 , the KNN classifier first identifies the K points in the training data that are closest to x_0 , represented by N_0 .

It then estimates the conditional probability for class j as the fraction of points in N_0 whose response values equal j :

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

~ $I(y_i = j)$ is an indicator variable that equals 1 if $y_i \neq j$ and zero if $y_i = j$.

~ $I(y_i = j) = 0$ means that i th observation is classified correctly.

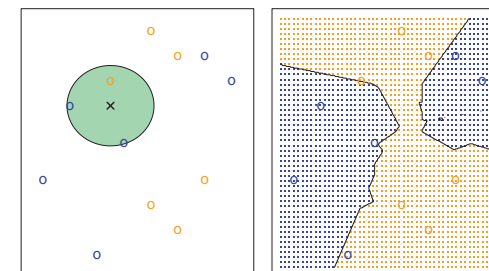
Finally, KNN applies Bayes rule and classifies the test observation x_0 to the class with the largest probability.

K-Nearest neighbors

Consider a small training data set consisting of six blue and six orange observations.

Our **goal** is to make a prediction for the point labeled by the black cross.

- Suppose that we choose $K = 3$.
- Then KNN will first identify the three observations that are closest to the cross.
- It consists of 2 blue and 1 orange point: estimated probabilities of 2/3 for the blue and 1/3 for the orange class.

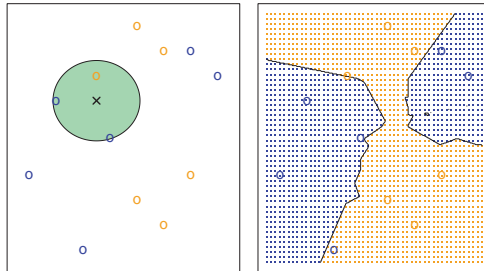


K-Nearest neighbors

Consider a small training data set consisting of six blue and six orange observations.

Our **goal** is to make a prediction for the point labeled by the black cross.

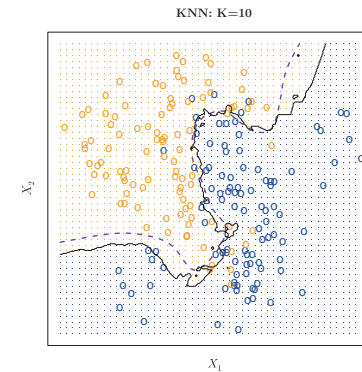
- ▶ KNN will predict that the black cross belongs to the blue class.
- ▶ Applying the KNN approach with $K = 3$ at all of the possible values for X_1 and X_2 , and have drawn in the corresponding KNN decision boundary.



K-Nearest Neighbors

13 / 34

K-Nearest neighbors



KNN is a very simple approach

- ▶ It can often produce classifiers that are surprisingly close to the optimal Bayes classifier.
- ▶ Notice that even though the true distribution is not known by the KNN classifier.
 - ~ The KNN decision boundary is very close to that of the Bayes classifier.

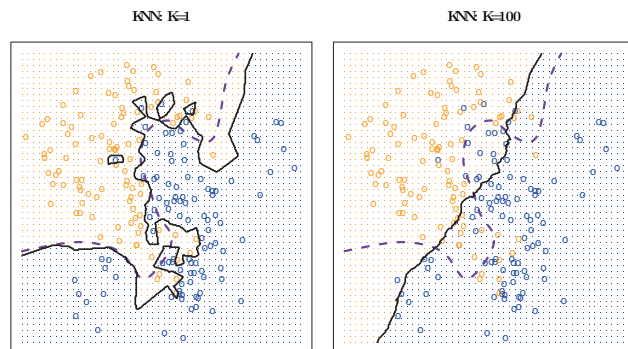
K-Nearest Neighbors

14 / 34

K-Nearest neighbors

The choice of K has a drastic effect on the KNN classifier obtained.

- ▶ $K = 1$: the decision boundary is overly flexible.
- ▶ It finds patterns in the data that do not correspond to the Bayes decision boundary.
- ▶ This corresponds to a **classifier that has low bias but very high variance**.



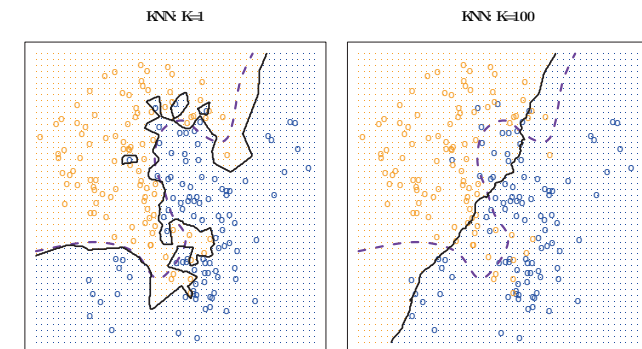
K-Nearest Neighbors

15 / 34

K-Nearest neighbors

The choice of K has a drastic effect on the KNN classifier obtained.

- ▶ As K grows, the method becomes less flexible and produces a decision boundary that is close to linear.
- ▶ This corresponds to a **classifier that has low variance but very high bias**.



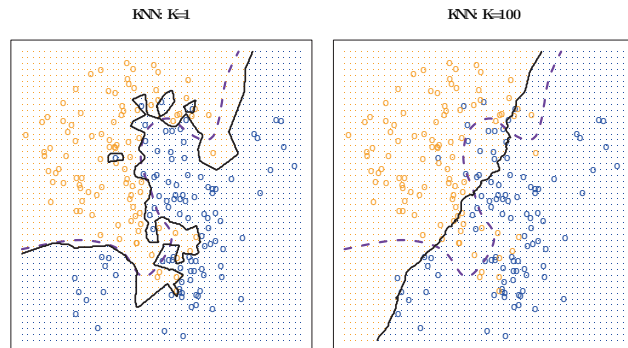
K-Nearest Neighbors

16 / 34

K-Nearest neighbors

The choice of K has a drastic effect on the KNN classifier obtained.

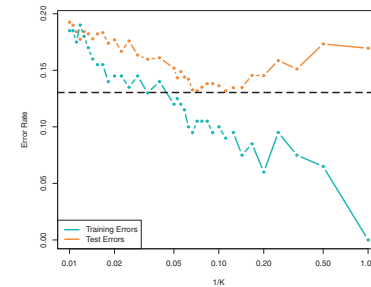
- ▶ With $K = 1$, the KNN training error rate is 0, but the test error rate may be quite high.
- ▶ As we use more flexible classification methods, the training error rate will decline but the test error rate may not.



K-Nearest Neighbors

17/34

K-Nearest neighbors



- ▶ As $1/K$ increases, the method becomes more flexible.
- ▶ The test error exhibits a characteristic U-shape, declining at first (with a minimum at approximately $K = 10$) before increasing again when the method becomes excessively flexible and overfits.

As for the regression, choosing the correct level of flexibility is critical to the success of any statistical learning method.

~ The **bias-variance tradeoff** can make this a difficult task.

K-Nearest Neighbors

18/34

K-Nearest neighbors in R

In R we can perform KNN using the `knn()` function (`class` library).

`knn` forms predictions using a single command and requires four inputs:

- ▶ A matrix containing the predictors associated with the training data.
- ▶ A matrix containing the predictors associated with the data for which we wish to make predictions.
- ▶ A vector containing the class labels for the training observations.
- ▶ A value for K , the number of nearest neighbors to be used by the classifier.

```
knn(train, test, cl, k)
```

K-Nearest Neighbors

19/34

K-Nearest neighbors in R

To illustrate the KNN approach in R, we use the `Caravan` data set in the `ISLR` library.

- ▶ It includes 85 predictors that measure demographic characteristics for 5822 individuals.
- ▶ The response variable is `Purchase`: indicates whether or not a given individual purchases a caravan insurance policy.
- ▶ In this data set, only 6% of people purchased caravan insurance.

```
> # The Caravan Data
> library(ISLR); data(Caravan)
>
> dim(Caravan)
[1] 5822 86
> attach(Caravan)
> summary(Purchase)
No  Yes
5474 348
```

K-Nearest Neighbors

20/34

K-Nearest neighbors in R

KNN classifier predicts the class of a given test observation by identifying the observations that are nearest to it.

↪ The scale of the variables matters.

Any variables that are on a large scale will have a much larger effect on the distance between the observations, and hence on the KNN classifier, than variables that are on a small scale.

↪ We standardize the data so that all variables are given a mean of zero and a standard deviation of one.

```
standardized.X=scale(Caravan [, -86])
var (Caravan [, 1])
var (Caravan [, 2])

var(standardized.X[, 1])
var(standardized.X[, 2])
```

K-Nearest neighbors in R

We split the observations

- ▶ A **test set** containing the first 1000 observations
- ▶ A **training set** containing the remaining observations.

We fit a **KNN model on the training data**.

We evaluate its performance on the test data.

```
> test =1:1000
> train.X=standardized.X[-test ,]
> test.X=standardized.X[test ,]
> train.Y=Purchase [-test]
> test.Y=Purchase [test]
> set.seed (1)
> knn.pred=knn(train.X,test.X,train.Y,k=1)
> mean(test.Y!=knn.pred)
[1] 0.118
> mean(test.Y!="No")
[1] 0.059
```

K-Nearest neighbors in R

With **K = 1** and **K=6**.

```
> knn.table.conf1 <- table(knn.pred,test.Y)
> knn.table.conf1
      test.Y
knn.pred No Yes
      No  873  50
      Yes  68   9
> knn.table.conf1[2,2]/sum(knn.table.conf1[2,])
[1] 0.117
```

K-Nearest neighbors in R

With **K = 3** and **K=6**.

```
> K=3
> knn.pred=knn(train.X,test.X,train.Y,k=3)
> knn.table.k3<- table(knn.pred,test.Y)
> knn.table.k3
      test.Y
knn.pred No Yes
      No  920  54
      Yes  21   5
> knn.table.k3[2,2]/sum(knn.table.k3[2,])
[1] 0.192
> knn.pred=knn(train.X,test.X,train.Y,k=6)
> # K=6
> knn.table.k6<- table(knn.pred,test.Y)
> knn.table.k6
      test.Y
knn.pred No Yes
      No  932  57
      Yes   9   2
> knn.table.k6[2,2]/sum(knn.table.k6[2,])
[1] 0.182
```

K-Nearest neighbors in R

As a comparison, we can also fit a logistic regression model to the data.

```
> glm.probs=predict(glm.fits,Caravan[test,],type="response")
> glm.pred=rep("No",1000)
> cutoff <-0.5; glm.pred[glm.probs > cutoff]="Yes"
>
> table(glm.pred,test.Y)
      test.Y
glm.pred  No  Yes
      No  934  59
      Yes   7   0
> glm.pred=rep("No",1000)
>
> cutoff <-0.25; glm.pred[glm.probs > cutoff]="Yes"
> table(glm.pred,test.Y)
      test.Y1
glm.pred  No  Yes
      No  919  48
      Yes   22  11
```

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Data set:** Kaggle competition sponsored by the University of Melbourne.
 - ▶ **Objective:** Predicting whether or not a grant application would be accepted.
- Classify grant applications based on their likelihood of success could be important for estimating the amount of potential funding to the university.

8708 grants between the years 2005 and 2008 available for **model building** and the **test set** contained applications from 2009 to 2010

- ▶ The data are available at Kaggle (<https://www.kaggle.com/c/unimelb>)
- ▶ **Only the training set data contain the outcomes for the grants**
- ▶ Many pieces of information were collected across grants, including whether or not the grant application was successful.

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Data set:** Kaggle competition sponsored by the University of Melbourne.
- ▶ **Objective:** Predicting whether or not a grant application would be accepted.

Classify grant applications based on their likelihood of success could be important for estimating the amount of potential funding to the university.

The original data contained many predictors such as:

- ▶ The role of each individual listed on the grant.
 - ▶ CI: Chief investigator
 - ▶ DR: Delegated researcher
 - ▶ PS: Principal supervisor
 - ▶ EA: External advisor
 - ▶ ECI: External chief investigator
 - ▶ SCI: Student chief investigator
 - ▶ SR: Student researcher
 - ▶ HV: Honorary visitor
 - ▶ UNK: Unknown

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Data set:** Kaggle competition sponsored by the University of Melbourne.
 - ▶ **Objective:** Predicting whether or not a grant application would be accepted.
- Classify grant applications based on their likelihood of success could be important for estimating the amount of potential funding to the university.

The original data contained many predictors such as:

- ▶ Characteristics of each individual on the grant:
 - ▶ Date of birth
 - ▶ Highest degree
 - ▶ Nationality
 - ▶ Number of prior successful (and unsuccessful) grants
 - ▶ Department
 - ▶ Faculty status
 - ▶ Level of seniority
 - ▶ Length of employment at the university
 - ▶ Number of publications in four different grades of journals

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Data set:** Kaggle competition sponsored by the University of Melbourne.
- ▶ **Objective:** Predicting whether or not a grant application would be accepted.

Classify grant applications based on their likelihood of success could be important for estimating the amount of potential funding to the university.

The original data contained many predictors such as:

- ▶ Codes related to the Australia's research fields, courses and disciplines (RFCD)
 - ▶ Subgroups: Applied Economics, Microbiology, and Librarianship.
 - ▶ 738 possible values of the RFCD codes in the data.
 - ▶ If more than one code was specified for a grant, their relative percentages were recorded.
- ▶ The RFCD codes listed by the Australian Bureau of Statistics range from 210000 to 449999.
- ▶ There were many grants with nonsensical codes (such as 0 or 999999) that were grouped into an unknown category for these analyses.

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Data set:** Kaggle competition sponsored by the University of Melbourne.
- ▶ **Objective:** Predicting whether or not a grant application would be accepted.

Classify grant applications based on their likelihood of success could be important for estimating the amount of potential funding to the university.

The original data contained many predictors such as:

- ▶ Codes related to the socio-economic objective (SEO)
 - ▶ Purpose of the grant: Developing construction activities or health services.
 - ▶ If more than one code was specified for a grant, their relative percentages were recorded.
 - ▶ Some values that did not map to any of the codes listed by the Australian government were grouped into an unknown category.

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Data set:** Kaggle competition sponsored by the University of Melbourne.
- ▶ **Objective:** Predicting whether or not a grant application would be accepted.

Classify grant applications based on their likelihood of success could be important for estimating the amount of potential funding to the university.

The original data contained many predictors such as:

- ▶ The submission date of the grant
- ▶ The monetary value of the grant, binned into 17 groups
- ▶ A grant category code which describes the type sponsor as well as a code for the specific sponsor

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Feature selection:** Transform, or encode, the original data structure into a form that is most informative for the model.
- ▶ The original form of the grant data is not conducive to modeling.
 - ▶ Many fields are broken down for each individual involved in the grant: As such, there are 15 columns in the data for each individual.
 - ▶ There are a large number of columns for a grant, many of which have no data.
- ▶ To solve this, the data are grouped in subgroups and indicators are defined.
- ▶ **1784 possible predictors** are created using this encoding scheme.
- ▶ The vast majority of these predictors are discrete in nature (i.e., either 0/1 dummy variables or counts) with many 0 values.
- ▶ Since many of the predictors are categorical in nature, missing values were encoded as "unknown".
- ▶ For example, 912 grants had missing category codes. A binary predictor for missing grant categories was created to capture this information.

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Unsupervised feature selection:** Removing predictors without measuring their association with the outcome.
- ▶ Two subsets of data are created because a significant number of predictors had pair-wise absolute correlations that were larger than 0.99.
 - ▶ The “full set” of predictors included all the variables regardless of their distribution (1070 predictors).
 - ▶ The “reduced set” is developed for models that are sensitive to sparse and unbalanced predictors and contained 252 predictors.

Introduction to Homework 3

Case Study: Predicting Successful Grant Applications

- ▶ **Task:** You will be asked to develop at least a linear and a nonlinear predictive model to classify the outcome of the grant application problem
 - ▶ Logistic regression
 - ▶ Linear and quadratic discriminant analysis
 - ▶ Neural networks
 - ▶ K-nearest neighbors
 - ▶ Support vector machines
- ▶ **Deadline:** Monday, December 4, 2017