Introduction to Statistical Machine Learning

UNI HEIDELBERG

Erstellt von: Nikolaus Schäfer

Dozent: Junprof. Dr. F. Krüger

12. Juli 2018

Inhaltsverzeichnis

Editor's Note

This script is my personal transcript of the lecture ”Introduction to Statistical Machine Learning” by Junprof. Dr. Fabian Krüger in the summer semester 2018. As it is my personal transcript, I cannot guarantee that it is mistakefree. If you find any mistakes (e.g. typos etc.) please feel free to edit them. Furthermore, I did not maintain the same format throughout the semester, and now I have too little time to edit everything. However, I think the document still has a somewhat clear structure. If you want to edit the format a little, color or bold some words or sentences to make it more readable and highlight the important stuff, again please feel free to do that.
Lastly I hope that it is of some use to the course. Enjoy and study hard!
Niko

1. Lecture April 18th

What is Statistical Learning? (2.1)

Example: Advertising Data

- Sales of a product in 200 different markets
- Budget for advertising in each maket for three channels: TV, radio, newspaper.
- Question: Which advertising channel increaes sales (by how much?)

In this example, $\underbrace{\text{sales}}_{Y}$ is the **output variable** (Y), advertising budgets are **input variables** (X_1: TV Budget, X_2: radio budget, X_3: newspaper budget)
Synonomys: input variable = predicators, regressors, features, independent variables
output variable = response variable, predictand, dependent variable
More generally: Response Y (quantitative, i.e. continuous number), p predicators $X_1, \dots, X_p$.
Aim: Study relationship betwenn Y and $X = (X_1, \dots, X_p)$.

Assuption: $Y = f(X) + \varepsilon$

when

- f is a fixed but unknown function of $X_1, \dots, X_p$
- ε is a random error term, independent of X, with $E(\varepsilon) = 0$

$\Rightarrow f$ represents systematic information that X provides about Y

Why estimate f? (2.1.1)

Two motivations: Prediction and inference

Prediction:

Predict Y using X, i.e. $\hat{Y} = \hat{f}(X)$
Main goal: Accurate prediction, i.e. $\hat{Y}$ as close as possible to $\hat{f}(X)$ (see Section 2.2 below)

Example: Predict next quarters GDP growth rate, Y, based on $X = (X_1, \dots, X_p)$.
X might include other other economic variables like current (this quarter's) GDP growth rate,

inflation rate, interest rates, survey forecasts (e.g. survey of Professional Forecastors conducted by ECB).

Two sources of error: $\underset{\text{(RE)}}{\textbf{Reducible error}}$ and $\underset{\text{(IE)}}{\textbf{irreducible error}}$

RE: Arises because $\hat{f}$ is an imperfect estimate of $f \Rightarrow$ Can be reduced by using the most appropriate statistical learning technique to estimate f

IE: Arises because ε is there and, by definition, cannot be predicted using X. In other words, IE would still be there if we had a perfect estimate of f (such that $\hat{f}(X) = f(X)$).

Main reason for existence of ε: Do not/can not measure all variables X that are relevant for Y. (for legal, technical, or ethical reasons)

Example:

$$\begin{aligned} Y &= \text{Price of a house} \\ X &= \text{information on location, number of rooms, size of the property} \end{aligned}$$

Possible factors that go into ε: Neighbors, urgent sales, psychological factors .

2. Lecture April 19th

$$\begin{aligned} Y &= f(X)+\varepsilon \quad \text{model} \\ \hat{Y} &= \hat{f}(X) \quad \text{forecast} \end{aligned}$$

Reducible error: Results from having poor estimate $\hat{f}$ of f
Irreducible error: Factors in ε are unobservable.

Example: Predict children's income form parents' income.

$$\underbrace{Y}_{\text{child income}} = f(\underbrace{X}_{\text{information on parent income}}) + \overbrace{\varepsilon}^{\text{variables that are unrelated to parents income}}$$

$$\begin{aligned} E\{(Y-\hat{Y})^2\} &= E\{(f(X)+\varepsilon-\hat{f}(X))^2\} = E\{(f(X)-\hat{f}(X)+\varepsilon)^2\} \quad \text{Note: Assume that X is} \\ &= \underbrace{(f(X)-\hat{f}(X))^2}_{\text{reducible error}} + \underbrace{E(\varepsilon^2)}_{V(\varepsilon)} + (f(X)-\hat{f}(X))\underbrace{E(\varepsilon)}_{=0} \quad \text{fixed (no random variable)} \end{aligned}$$

This class: Focus on techniques for estimating f, with the aim to minimize the reducible error.

Inference

Main question: How is Y affected if $X_1,\dots,X_p$ change?
Need to know/estimate precise structure of f to understand the relation between X and Y.

Specific inference questions:

- Which predictors are associated with Y?
- What's the specific form of the relation between Y and X_1 (e.g. linear, quadratic, logarithmic, positive, negative,...)

Example: Clinical trial where a random subgroup of patients gets a drug (treatment group), all others receive a placebo (control group). Based on such data, one can study the relation between a patient's health and whether the patient got the drug (yes/no).

Examples on prediction vs. inference (Einav + Levin, 2014)
Prediction:

- Predict children's wealth from parents' wealth
- Predict whether a user will click on an add

Inference:

- Ask whether taking online classes increases wage
- Evaluate long-run effects of teachers

In some examples, both prediction and inference may be of interest. E.g. Y = value of house, X = features

Prediction: E.g. realtor/salesperson want to predict value of house as it is.
Inference: Find variable that affect house price (in order to maximize value given resources)

How do we estimate f? (2.1.2)

The data set, that is used to estimate f is called **training data**, given by n observations. Let y_i denote the oberserved response for obersvation $i = 1,\ldots,n$ and x_{ij} the value for observation i and predictor $j = 1,\ldots p$.
We denote $x_i = (x_{i1},\ldots,x_{ip})^T$, so that our training data set consists of $\{(x_1,y_1),\ldots,(x_n,y_n)\}$

Based on the training data, we aim to estimate f such that $\hat{f}(X) \approx Y$ for any observation (X,Y).
Two categories of estimation: Parametric and nonparametric.

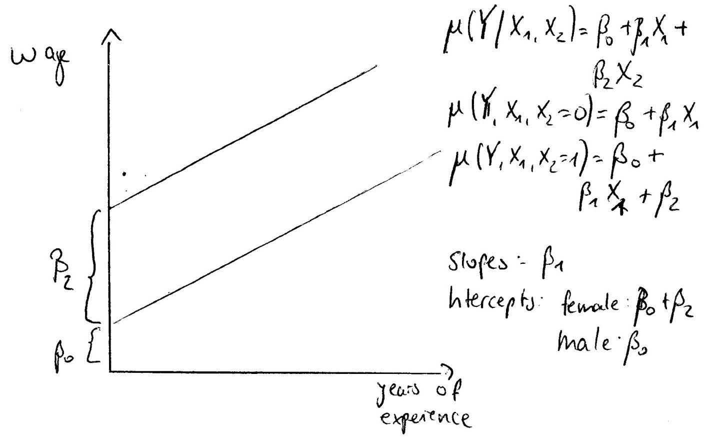
Parametric methods

Two steps:

1. Make an assumption about the functional form of f, e.g. the linear model assumes that Y is a linear function of X: $f(X) = \beta_0 + \beta_1 X_1 + \ldots \beta_p X_p$
2. After the form of f has been selected: Use training data to estimate (or fit) the model. E.g in the linear model we would estimate the coefficients $(\beta_0,\ldots,\beta_p)$ using ordinary least squares. For nonlinear models, maximum likelihood is a popular estimation method.

How to choose f in step 1? Simple form (such as linear model) is easy to estimate, both imposes strong assumptions on the data. E.g.

$$\begin{aligned} Y &= \text{wage} \\ X_1 &= \text{experience}, \quad X_2 = \text{gender(1 if female, 0 if male)} \\ Y &= \beta_0 + \beta_1 X_1 + \beta_2 X_2 \end{aligned}$$


wage
β_2
β_0
years of experience
$\mu(Y/X_1, X_2) = \beta_0 + \beta_1 X_1 + \beta_2 X_2$
$\mu(Y, X_1, X_2 = 0) = \beta_0 + \beta_1 X_1$
$\mu(Y, X_1, X_2 = 1) = \beta_0 + \beta_1 X_1 + \beta_2$
Slopes : β_1
Intercepts: female: $\beta_0 + \beta_2$
male: β_0

Assumption in linear model: Same slope for males and females (experience to wage)
Examples why this might be violated:

- Females/Males might work in different jobs with different returns to experience
- Discrimination against females who work part-time
- Relationship could be much more complex than that (e.g. wage could depend on entire employment history, not just experience)

Bottom line: Linear model implies restrictive assumptions. (Possible fix: interaction terms)
Flexible forms of f that impose less assumptions, but could lead to overfitting (fit training data very closely, but get problems for new (test) data). overfitting: i.e., accidentally modeling noise in the training data.

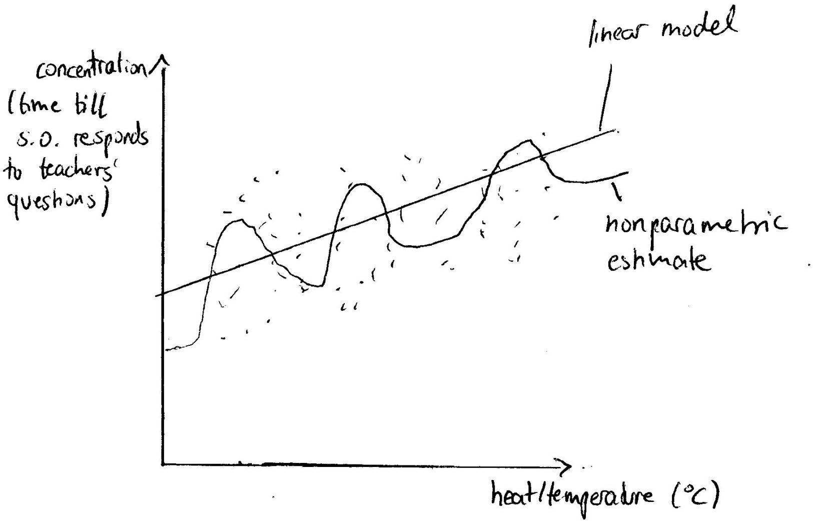
3. Lecture April 25th

$Y = f(X) + \varepsilon$

Parametric Estimate of f

Linear model: $f(x) = \beta_0 + \beta_1 X_1 + \dots \beta_p X_p$

Linear model is linear in the parameters β_{ji} not necessarily linear in the X_j's (e.g. may have quadratic terms or interaction terms)



Non-parametric estimate of f

Try to learn the form of f from the training data.

pro: Avoids risk of misspecifying f

contra: Estimating the entire form of f is ambitious, and typically requires many data points (i.e. large training sample)

Question: Why not always choose the most flexible model that's available? (see section 2.1.3)
Inference: Some of the very flexible models (e.g. Bagging) are hard to interpret, i.e. reveal little about the structure of the data. This leads to a trade-off between a model's flexibility and interpretability (see Figure 2.7)

Subset selection lasso: Take small set of variables from a large linear model
not very flexible but very interpretable
↳ results are easy to interpret, because they choose the relevant variables from the data set.

Least squares: Tradeoff: more flexibility ⇒ less interpretable
Which models to choose depends on the data set and what models should ”show”

Prediction: Choosing a very flexible model may be a bad idea due to overfitting

Supervised vs. unsupervised learning (2.1.4)

Supervised learning:

We have data on repsonse y_i and parameters x_i (e.g. linear regression, logistic regression, regression trees etc.)

Goal: Learn relationship between Y and X

Terminology: Response Y_i ’supervises’ the learning process (clear what the ’correct’ answer should be)

Unsupervised learning:

We have data on X_i but no respone Y_i

Popular tool: Determine dusters of similar observations in terms of X_i (Figure 2.8)

Example: Identifying different groups of customers.
↳ This class: We’ll only look at supervised learning.

Regression vs. classification (2.1.5)

Quantitative variables take on numerical values, e.g.: response time in classroom, price of a house or stock
Qualitative variables belong to one out of the K classes, e.g.: categorical measure of student participation (low/medium/high), information on stock going up or down.

Problems with a quantitative response are called regression problems.
Problems with a qualitative response are called classification models.

Notes:

1. Whether predictors are quantitative or qualitative is usually not important
2. Sometomes we convert a (numerical) probability into a classifier, so distinction between regression and classification may be blurry (e.g. logistic regression)

Assessing model accuracy (2.2)

Question: Many models/data sources around, which one to choose?

For a quantitative response, mean squared error (MSE) is the most common measure:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2, \tag{1}$$

where, y_i = realization $\hat{f}(x_i)$ = its prediction
⇒ smaller MSE corresponds to better model.
Note that MSE in (1) is computed from training data $i = 1, \ldots, n$
But in practice, we want to know, how our method performs for a new, previously unseen test observation (x_0, y_0) that was not used for estimating our method. That is, we are interested in the **test MSE** given by

$$E(y_0 - \hat{f}(x_0))^2 \tag{2}$$

Note: We test y_0 as a random variable but x_0 is fixed
↳ We know characteristics (x_0) of new test observations, but not the outcome (y_0).

How to estimate test MSE in (2)?

- Use part of the data as test data
- What if there is not enough data to have a test sample?
  !!! Never use training MSE as an estimate of test MSE!!!
- Many (all) methods can be tuned to have a small training MSE, but this does not imply a small test MSE (danger of overfitting (see Fig. 2.9)) Green line left image: most flexible matching points too close by: sign of overfitting
  Right image: one of the two curves is training MSE (lower line). Test MSE is the red line.

Why is the training MSE lower than test MSE?

- model can be fitted mechanically to minimize training MSE

- time series data: training MSE (past), test MSE (future).
  easier to depict past than future

4. Lecture May 2nd

Figure 2.9: (In the slides)
(1) Left panel

- Black curve is $f(x)$, the model from which the data was simulated
- Orange line, shows linear regression (rather inflexible, misses a lot of structure in data, i.e. 'underfitting')
- Green curve is a very model called spline (wiggly, 'overfitting')
- Blue curve is a moderately flexible spline variant (just right, closest to the true curve)

(2) Right panel

- Plot shows training and test MSE as a function of the method's flexibility
- Flexibility is measured as degrees of freedom (more degrees of freedom $\Leftrightarrow$ more flexible; Degrees of freedom = # parameters for linear model)
- Training MSE decreases as model becomes more flexible
- Test MSE is U-shaped

Why does overfitting occur?

- Statistical learning methods pick up both signal and noise
- If a method is too flexible, it picks up a lot of noise from the training data
- Test data are different (different noise or ε values than the training data), leading to high test MSE

Ways to find 'best' model in terms of test MSE

- Test data (reserve part of the sample as test data)
- Cross validation (later in the course)
- Information criteria (later in the course)

Parable of Google Flu

GFT = Google Flu Trends
Idea of GFT: Construct early warning indicatior of flu prevalence (Def: how often sth. occurs) based on search queries. ⇝ important information, allows policy-makers to organize health policies, or shift resources from less to more affected states, etc.

GFT specification:

$$\text{logit}(\underbrace{I_t}_{\substack{\text{share of doctor visits related}\\ \text{to the flu, released by}\\ \text{CDC (official health body in the US)}}}) = \alpha \cdot \text{logit}(\underbrace{Q_t}_{\text{share of flu-related search queries}}) + \varepsilon_t \qquad (t \text{ denotes time})$$

$$\text{logit}(p) = \ln(p) - \ln(1-p): \quad (0,1) \Rightarrow \mathbb{R} \text{ (basically for modeling convenience)}$$

Question: How to define Q_t? Based on (1) List/ranking of all search terms, (2) Choose amount of seach terms to include (Ginsberg et. al, Figure 1)
Some efforts to drop seasonal (but not flu-related) terms like college basketball, Oscar nominations.
↳ Critique by Lazer et al (2014): Procedure not transparent, list of search terms not published

Is overfitting a big concern?

- No, in the sense that the model is small (one regressor only)
- Yes, in the sense that there are many possible regressors (> 50 million search terms, choose list of k terms, $k \in \{1,\ldots,50.000.0000\}$)
  ⇝ Have to choose between billions of possible specifications

What went wrong?

Lazer et al, Figure 2: GFT prediction clearly and consistently overpredicts the CDC data.
Most likely explanation: Changes in the search engine algorithm, which led to increase in the Q_t variable, which had no substantive reason.
⇝ Quite complex, result of both search engine technology and user behavior.

Criticism by Lazer et al (2014):

- GFT performance not too great to begin with
  - CDC data fairly easy to predict from past, CDC data and seasonal information
  - GFT predictions should have been compared to such a benchmark method
- Changes in Q_t were not incorporated/were hard to incorporate

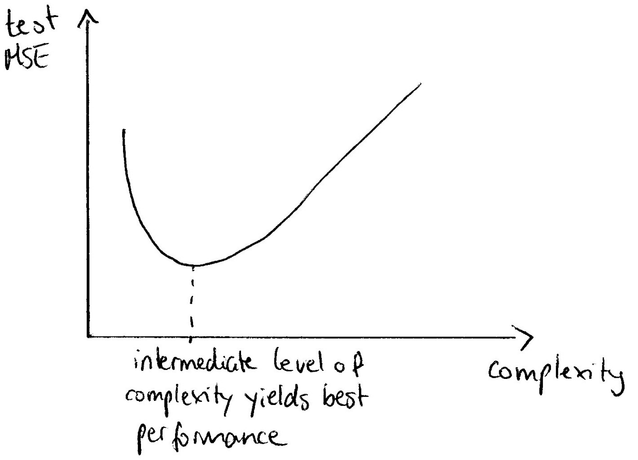
⇝ ’Big data habris’: Data can do magic just because of a large sample

⇝ Need to understand why a model works (substantive or statistical basis for the model’s success)

5. Lecture May 3rd

The bias variance trade-off (2.2.2)

When plotting a method's test sample MSE as a function of the method's complexity, we typically get a U-shaped pattern:(see figure 2.9 right panel)
Sketch:



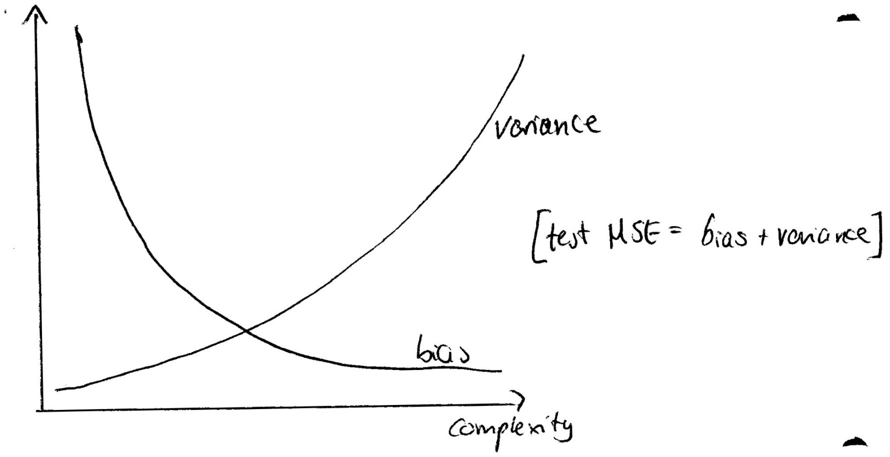
⇝ This pattern can be explained by two competing properties of statistical learning methods: Bias and variance.

$$E\{(y_0 - \hat{f}(x_0))^2\} = E\{(f(x_0) + \varepsilon - \hat{f}(x_0))^2\} = E\{(f(x_0) - E(\hat{f}(x_0)) + E(\hat{f}(x_0)) + \varepsilon - \hat{f}(x_0))^2\}$$
$$= E(\varepsilon^2) + E\{(f(x_0) - E(\hat{f}(x_0)) + E(\hat{f}(x_0)) - \hat{f}(x_0))^2\}$$
$$= \underbrace{E(\varepsilon^2)}_{A} + \underbrace{E\{(f(x_0) - E(\hat{f}(x_0)))^2\}}_{B} + \underbrace{E\{(E(\hat{f}(x_0) - \hat{f}(x_0)))^2\}}_{C} + \underbrace{2E\{(f(x_0) - E(\hat{f}(x_0)))(E(\hat{f}(x_0)) - \hat{f}(x_0))\}}_{D}$$

ε is part of the test sample ⇔ independent of $\hat{f}(x_0)$ (part of the training sample)

$$D = 2E\{(f(x_0) - E(\hat{f}(x_0)))(E(\hat{f}(x_0)) - \hat{f}(x_0))\} = 2E\{(f(x_0) - \hat{f}(x_0))\underbrace{(\hat{f}(x_0) - \hat{f}(x_0))}_{=0}\} = 0$$
$$A = Var(\varepsilon) \geq 0$$
$$B = E\{(\quad \underbrace{f(x_0)}_{\text{optimal prediction}} \quad - \quad \underbrace{E(\hat{f}(x_0))}_{\text{expected value of prediction}} \quad)^2\} = \text{Bias of } \hat{f}(x_0) \geq 0$$
$$C = E\{(E(\hat{f}(x_0) - \hat{f}(x_0))^2\} = Var\{\hat{f}(x_0)\} \geq 0$$

Three components of expected test MSE
(1) **Var**(ε) $\rightsquigarrow$ irreducible error, cannot do anything about this one $\rightsquigarrow$ Measure of difficulty of prediction problem
(2) **Bias of** $\hat{f}(x_0)$: Expresses the error that we make by approximating a (possibly) complex real-life problem with a statistical model. E.g. when the true curve f is nonlinear, then linear model $\hat{f}$ will always lead to a biased estimate (even if we had a huge training data set).
$\rightsquigarrow$ In general, more flexible models will have smaller bias.
Intuition: Flexible methods can produce various curves f (e.g. linear or nonlinear, smooth or rough), depending on the data.
(3) **Variance of** $\hat{f}(x_0)$**:** Expresses how much $\hat{f}$ would change if we used a different training data set to estimate $\hat{f}$. If $Var\{\hat{f}(x_0)\}$ is high, this means that the SL method is fragile (small changes in training data can lead to large changes in $\hat{f}$).
$\rightsquigarrow$ in general, flexible methods tend to have higher variance.



- When using a more flexible method, we increase the variance but reduce the bias

- Challenge is to choose the optimal compromise that minimizes test MSE

$\rightsquigarrow$ This challenge is called the bias-variance trade-off.

The classification setting (2.2.3)

$\rightsquigarrow$ concepts like Bias to Variance Tradeoff transfrer classification setting, but we need some modification since y_i is now qualitative.

Suppose we seek to estimate f based on a training observations $\{(x_1,y_1),\ldots,(x_n,y_n)\}$, where the y_i's are now qualitative.
Training error rate is defined as

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}(y_i \neq \hat{y}_i)$$

where $\hat{y}_i$ is the predicted class label for observation i, based on $\hat{f}(x_i)$. Furthermore,

$$\mathbb{1}(y_i \neq \hat{y}_i) = \begin{cases} 1 \text{ if } y_i \neq \hat{y}_i \text{ (i.e. wrong classification)} \\ 0 \text{ if } y_i = \hat{y}_i \text{ (i.e., correct classification)} \end{cases}$$

Training error rate gives traction of wrong classifications. As in the regression setting, we are most interested in the test sample performance of a SL method. The latter is captured by the test error rate, which for a new observation (x_0,y_0), is given by

$$P(y_0 = \hat{y_0}),$$

i.e the probability that the SL method gives a wrong classification of the new observation.
$\rightsquigarrow$ should be as small as possible.

6. Lecture May 9th

Measures of (test sample) error:

Mean (expected) squared error

$$E(y_0 - \hat{f}(x_0))^2 \qquad \text{numerical/continuous response}$$

$(\underbrace{x_0}_{\text{known}}, \underbrace{y_0}_{\text{unknown}})$ is a new data point that's not used for model fitting.

Probability of false classification:

$$P(y_0 \neq \hat{y}_0), \qquad \text{qualitative response (classification)} \tag{3}$$

where $y_0 \in \{0,1\}$ is the observed class label, and $\hat{y}_0 \in \{0,1\}$ is the predicted label.

↳ Transfers to the case of > 2 classes

Bayes classifier

Can be shown: Test error rate in (3) is minimized by assigning each observation to the most likely class, given its predictor values. I.e., assign test observation with predictor vector x_0 to the class j for which

$$P(Y = j | X = x_0)$$

is largest. ⇝ "Bayes classifier" (BC)
If there are only two classes: Assign to class '1' if

$$P(Y = 1 | X = x_0) > 0.5$$

BC produces the lowest possible error rate ('Bayes error rate'), given by

$$1 - E\left(\max_j P(Y = j | X)\right),$$

this corresponds to the irreducible error in regression.

K-Nearest Neighbors

Problem with the BC: Depends on the unknown true probability. $\rightsquigarrow$ Need to estimate it from training data. One approach for doing this:
K-nearest neighbors (KNN) classifier.

Idea: Find K points in the training data that are closest to x_0.
Denote this set as $\mathcal{N}_0$. Then let

$$\hat{P}(Y = j|X = x_0) = \frac{1}{k} \sum_{i \in \mathcal{N}_0} \mathbb{1}(y_i = j),$$

i.e. the fraction of neighbors that belong to class j.
Example: Figure 2.14

- Training data given by six blue and six orange observations
- x_0 marked by black cross
- Example: $k = 3 \rightsquigarrow$ Three NNs shown (green circle)

$$\left.\begin{aligned} &\hat{P}(Blue|X = x_0) = 2/3 \\ &\hat{P}(Orange|X = x_0) = 1/3 \end{aligned}\right\} \text{Classify as blue (no weighting of neighbors)}$$

Right panel shows KNN decision boundary

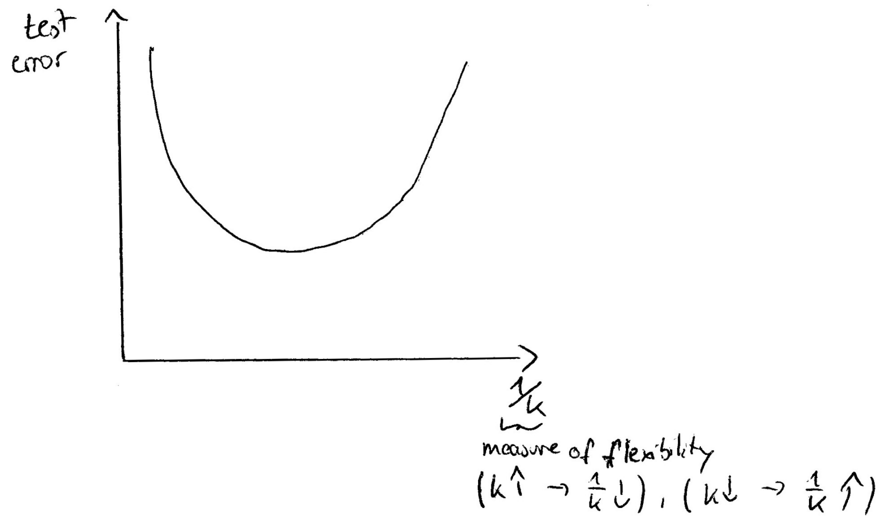
What's the role of K?

- Low value of K: Very flexible method $\rightsquigarrow$ Low bias, high variance
  ↳ Decision boundary becomes wiggly (small change in x_0 can change class)
- High value of K: Inflexible method $\rightsquigarrow$ High bias, low variance.
  ↳ Decision boundary becomes very smooth

Extreme example: $K = n$ (= # traning observations)

- Always predict majority class ($\mathcal{N}_0$ is the complete sample)

Low K means low training error (= zero if $K = 1$)
More importantly: Test error!



K-nearest neighbor example from the book Chapter 2 Problem 7:

(a) Compute euclidean distance:

Obs	*eucl.distance*	*Color*
1	3	R
2	2	R
3	$\sqrt{10}$	R
4	$\sqrt{5}$	G
5	$\sqrt{2}$	G
6	$\sqrt{3}$	R

(b) $\hat{P}(Green|X = x_0) = 1$ for $K = 1$
use only observation 5, i.e. $\mathscr{N}_0 = \{5\}$

(c) $K = 3:$
$\hat{P}(Green|X = x_0)) = 1/3$
use obserservation $2, 5$ and 6, i.e. $\mathscr{N}_0 = \{2, 5, 6\}$

(d) If decision boundary is complex, need a flexible model
↳ Small value for K.

7. Lecture, May 17th

Linear regression

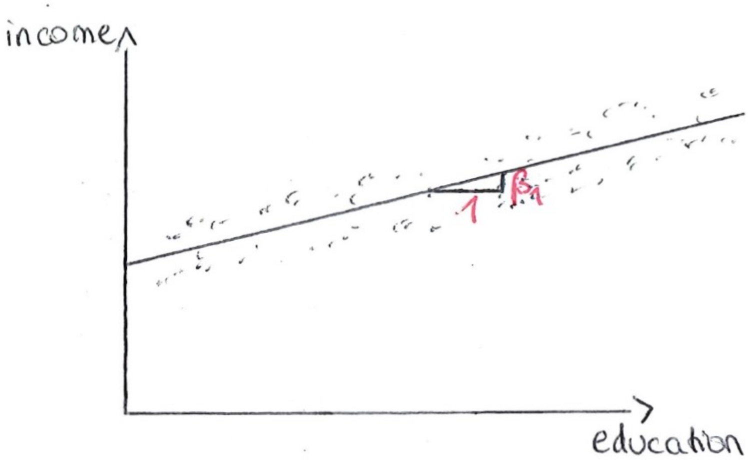
Linear regression is the standard approach for supervised learning with a quantitative response.
$\rightsquigarrow$ useful starting point before discussing more complicated SL methods.

Simplest case: One predictor variable

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

- β_0, β_1 are unknown parameters $\rightsquigarrow$ need to estimate them
- Prediction: $\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$

Example: X = education (years of schooling), Y = Income (Euros)
$\rightsquigarrow$ Prediction $\hat{Y}$ is a linear function of X



Estimating the coeffictions (3.1.1)

Training data $(X_1, Y_1), \ldots, (X_n, Y_n)$.
Goal: Find estimates $\hat{\beta}_0$, $\hat{\beta}_1$ that fit the data, i.e. such that $e_i = (y_i - \hat{\beta}_0 - \hat{\beta}_1 X_i)$ is 'small' for $i = 1, \ldots, n$.
$\rightsquigarrow$ e_i is the residual (difference between observation and prediction) for case i.
Residual sum of squares (RSS) is given by

$$\text{RSS} = e_1^2 + e_2^2 + \ldots + e_n^2$$

Which estimates $(\hat{\beta}_0, \hat{\beta}_1)$ minimize RSS?

Optimization problem:

$$\min_{b_0, b_1} \text{RSS} = \sum_{i=1}^{n} (Y_i - b_0 - b_1 X_i)^2 \equiv \text{RSS } (b_0, b_1)$$

Note, that b_0 and b_1 are estimators for β_0 and β_1 respectively. Solving the problem:

$$\begin{aligned} \frac{\mathrm{dRSS}(b_0, b_1)}{\mathrm{d}b_0} &= -2 \sum_{i=1}^{n} (Y_i - b_0 - b_1 X_i) \overset{!}{=} 0 \\ \Leftrightarrow \qquad \underbrace{\sum_{i=1}^{n} Y_i}_{\equiv n\overline{Y}} &\overset{!}{=} n b_0 + b_1 \underbrace{\sum_{i=1}^{n} X_i}_{=n\overline{X}} \\ \Leftrightarrow \qquad \hat{\beta}_0 &= \overline{Y} - \hat{\beta}_1 \overline{X} \end{aligned}$$

$$\begin{aligned} \frac{\mathrm{d}RSS(b_0, b_1)}{\mathrm{d}b_1} &= -2 \sum_{i=1}^{n} X_i (Y_i - b_0 - b_1 X_i) \overset{!}{=} 0 \\ \Leftrightarrow \quad \sum_{i=1}^{n} X_i Y_i &= b_0 \sum_{i=1}^{n} X_i + b_1 \sum_{i=1}^{n} (X_i)^2 \end{aligned}$$

plug in $\hat{\beta}_0$

$$\begin{aligned}
\sum_{i=1}^{n} X_iY_i &= (\overline{Y} - \hat{\beta}_1\overline{X})n\overline{X} + \hat{\beta}_1\sum_{i=1}^{n}(X_i)^2 = n\overline{XY} + \hat{\beta}_1(\sum_{i=1}^{n}(X_i)^2 - n\overline{X}^2) \\
\Leftrightarrow \quad \underbrace{\frac{1}{n}\sum_{i=1}^{n} X_iY_i - \overline{XY}}_{Cov(X_i,Y_i)} &= \underbrace{\hat{\beta}_1\frac{1}{n}\sum_{i=1}^{n} X_i^2 - \overline{X}^2)}_{Var(X_i)} \\
\Leftrightarrow \quad \hat{\beta}_1 &= \frac{\frac{1}{n}\sum_{i=1}^{n} X_iY_i - \overline{XY}}{\frac{1}{n}\sum_{i=1}^{n}(X_i^2 - \overline{X}^2)}
\end{aligned}$$

- Estimate $\hat{\beta}_1$ is positive whenever there's a positive correlation between X_i and Y_i in the sample.
- Average residual given by

$$\frac{1}{n}\sum_{i=1}^{n} e_i = \frac{1}{n}\sum_{i=1}^{n}(Y_i - (\underbrace{\overline{Y} - \hat{\beta}_1\overline{X}_i}_{=\hat{\beta}_0}) - \hat{\beta}_1X_i) = \frac{1}{n}\sum_{i=1}(Y_i - \overline{Y}) = 0$$

⇝ by construction, average residual is zero

$\hat{\beta}_0$, $\hat{\beta}_1$ shown above are called ordinary least squares (OLS) estimates
⇝ can be shown: $E(\hat{\beta}_0) = \beta_0$, $E(\hat{\beta}_1) = \beta_1$ (i.e. estimates are unbiased)
⇝ if you simulate many data sets of size n, then both $\hat{\beta}_0$ and $\hat{\beta}_1$ will be correct on average (great programming exercise)

Which factors affect accuracy of OLS estimator?
↳ Ideally, want $Var(\hat{\beta}_0)$ and $Var(\hat{\beta}_1)$ to be small.
This is the case if,

- The sample size, n, is large
- The error variance (of ε), σ^2, is small
- The variance of X_i, $Var(X_i)$, is large

⇝ Note: RSS is minimized by construction, hence not a good measure of test sample performance.

Multiple Linear Regression (3.2)

Often want to consider more than one predictor variable

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p + \varepsilon \qquad \text{(p predictor variables)}$$

e.g.

$$\begin{aligned}
Y &= \text{income} && (4)\\
X_1 &= \text{years of schooling} && (5)\\
X_2 &= \text{gender (e.g. 1 if female, 0 if male)} && (6)\\
X_3 &= \text{job experience (in years)} && (7)\\
X_4 &= \text{location1 (e.g. 1 if east, 0 if west)} && (8)\\
X_5 &= \text{location2 (e.g. 1 if urban area, 0 if not)} && (9)\\
X_6 &= \text{online courses (e.g. 1 if taken MOOC)} && (10)\\
&\vdots && (11)
\end{aligned}$$

How to interpret parameters?

⇝ E.g., β_3? ⇝ Change in predicted income of job experience increases by one year but all other variables stay constant

OLS estimation:

$$\text{minimize}_{b_0,\ldots,b_p} \text{ RSS} = \sum_{i=1}^{n} (Y_i - b_0 - b_1 X_{i1} - \ldots - b_p X_{ip})^2$$

⇝ Problem can be solved using matrix Algebra

⇝ Implemented in statistical software, e.g. R function 'lm'

Note: Variable may be important in simple linear regression (1 predictor) but may drop out/be irrelevant in multiple regression.
E.g.: In regression of Y on X_6 only, might find strong positive relation (but might vanish in multiple regression)

Model fit (i.e. how well a model matches the data)

Most popular measure: R^2, given by

$$R^2 = 1 - \frac{\text{RSS}}{\text{TSS}}, \quad \text{TSS} = \sum_{i=1}^{n} (Y_i - \overline{Y})^2$$

Properties:

- $0 \leq R^2 \leq 1$,
    - $RSS = 0 \Rightarrow$ training sample predictions are perfect $\Rightarrow R^2 = 1$
    - RSS = TSS $\Rightarrow \hat{Y}_i = \overline{Y}$, $i = 1, \ldots, n$ (X variables not at all helpful for predicting Y) $\rightsquigarrow R^2 = 0$
- If you include more variables into the model, then R^2 increases by construction
    - TSS stays the same
    - RSS goes down by construction (more degrees of freedom)
- Will look at better measures of model fit (e.g. adjusted R^2, Akaike information criterion) $\rightsquigarrow$ account for the number of predictors p, that are used in the model

8. Lecture May 23rd

Linear regression model

$$Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \varepsilon$$

$R^2 = 1 - \frac{\text{RSS}}{\text{TSS}}$
Note: OLS estimator is designed to minimize RSS $\Leftrightarrow$ maximize R^2. It can be shown that $R^2 = (\underbrace{Cor(Y_i, \hat{Y}_i)}_{\text{Correlation in training sample}})^2$

Main problem with R^2: Increases mechanically when using more predictors.
⇝ We'll look at better ways to measure model fit

<u>Prediction in linear model</u>
...given by $\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \ldots + \hat{\beta}_p X_p$
$\neq$ true expected value (if model is linear)

$$f(X) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$$

Ways to quantify prediction uncertainty

1. <u>Confidence interval</u>: Describe how close $\hat{Y}$ is to $f(X)$. E.g. 95% CI should contain $f(X)$ with probability 95%
   ⇝ Estimation uncertainty about $\beta_0, \beta_1, \ldots$

2. <u>Prediction interval</u>: Describes uncertainty about both $f(X)$ and ε. E.g., 95% PI should contain Y with probability 95%

<u>Note</u>:

- PI generally wider (longer) than CI

- As $n \to \infty$, CI shrinks to true value $f(X)$ (i.e. CI has lenght zero, no estimation uncertainty left).

- As $n \to \infty$, PI only covers uncertainty about ε (does not collapse to a length zero), there can still be a lot of uncertainty left)

Other considerations in the linear regression model (3.3)

Qualitative predictors (3.3.1)

E.g. prediction model for the amount somebody donates to an online charity project.
Possible qualitative predictors in that example:

- gender
- employment status
- income category (often used as categorical variable for data privacy reasons)
- profession (say, K different groups)
- political orientation

$\rightsquigarrow$ Question: How to include these variables into a model?

Case A: Only two levels

Example: Unemployed Yes/No
Create a new variable

$$X_i = \begin{cases} 1 \text{ if person } i \text{ is unemployed} \\ 0 \text{ if not} \end{cases}$$

and use this variable in a regression model.
E.g.

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i = \begin{cases} \beta_0 + \beta_1 \text{ if person } i \text{ is unemployed} \\ \beta_0 \text{ if not} \end{cases}$$

Charity example:

$\beta_0 =$ Mean contribution of employed persons
$\beta_1 =$ Difference in (mean) contribution for unemployed vs. employed persons

Note:

- Coding of X_i is arbitrary: Could also set X_i to 1 for employed persons
  ↳ Interpretation of parameters would change
  ↳ RSS statistic/R^2 would stay the same

Case B: More than two levels
E.g., income category (low/middle/high) $\rightsquigarrow$ 3 levels $\rightsquigarrow$ Need to create two dummy variables

$$X_{i1} = \begin{cases} 1 \text{ if } i \text{ is in the middle income category} \\ 0 \text{ if not} \end{cases}$$

$$X_{i2} = \begin{cases} 1 \text{ if } i \text{ is in the high-income category} \\ 0 \text{ if not} \end{cases}$$

Then

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \varepsilon_i = \begin{cases} \beta_0 + \varepsilon_i & \text{low-income} \\ \beta_0 + \beta_1 + \varepsilon_i & \text{middle-income} \\ \beta_0 + \beta_2 + \varepsilon_i & \text{high-income} \end{cases}$$

- β_0 = mean contribution for low-income
- β_1 = difference in mean contribution for middle vs. low income
- β_2 = difference in mean contribution for high vs. low income

low-income category is the baseline category

Example: Linear models in R

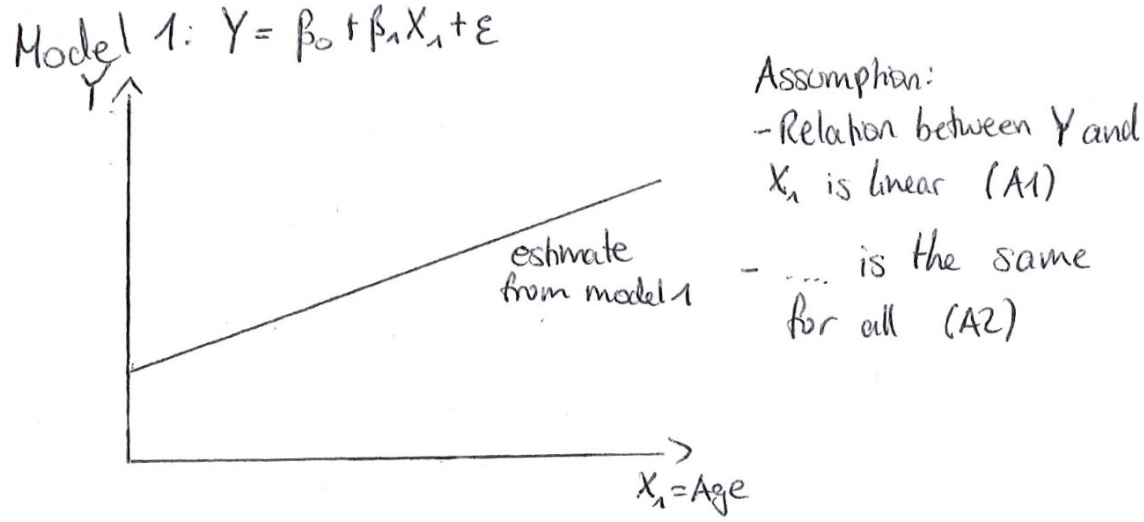
- lm($y \sim x$, data = d) runs linear regression of y on x, whereby both are in data set d.
- If x is a factor variable, then R automatically codes the $(K-1)$ binary variables we discussed, where $K = \#$ of levels for x.

9. Lecture May 24th

Yesterday's example:

$$\underbrace{Y}_{\substack{\text{fundraising} \\ \text{contribution}}} = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \varepsilon$$

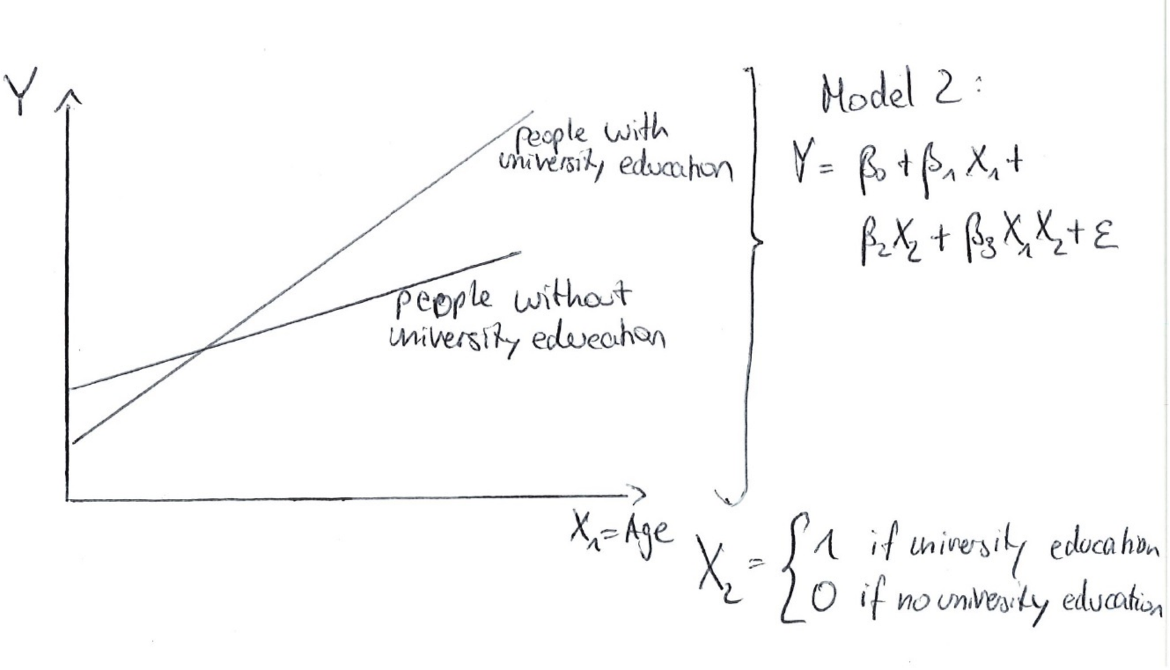
where the X_i are personal charactersitics (age, education, political orientation)



Model 1: $Y = \beta_0 + \beta_1 X_1 + \varepsilon$

Assumption:
- Relation between Y and X_1 is linear (A1)
- ... is the same for all (A2)

Possible violation of A2: Slope of regression line differs across subgroups

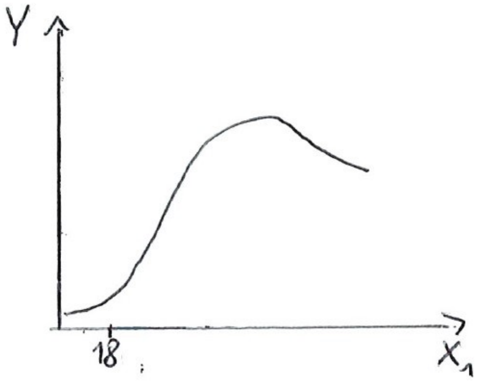

Y
people with university education
people without university education
X_1=Age
Model 2:
$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \varepsilon$
$X_2 = \begin{cases} 1 & \text{if university education} \\ 0 & \text{if no university education} \end{cases}$

Model 2 yields two different regression lines:

- $Y = \beta_0 + \beta_1 X_1 + \varepsilon$ without university education
- $Y = (\beta_0 + \beta_2) + (\beta_1 + \beta_3) X_1 + \varepsilon$ with university education

⇝ Note: Model 2 is more complex than model 1, hence lower bias but higher variance.

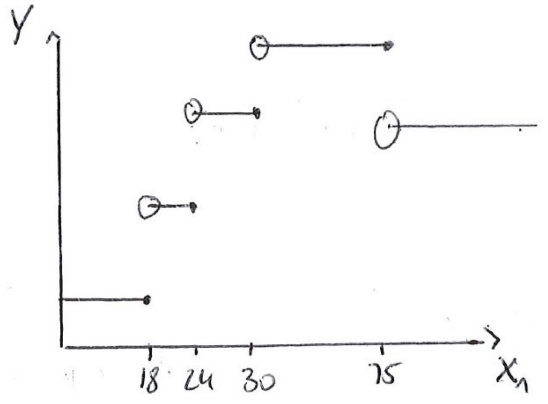
Possible violation of A1: Relation and Y and X_1 is not linear.



$\rightsquigarrow$ Can we cover nonlinear relations in the linear model?

Possible fixes: Define age categories, i.e. qualitative variable

$$X_2 = \begin{cases} 1 \text{ if } X_1 < 18 \\ 2 \text{ if } 18 \leq X_1 < 24 \\ 3 \text{ if } 24 \leq X_1 < 30 \\ \vdots \end{cases}$$



piecewise constant model

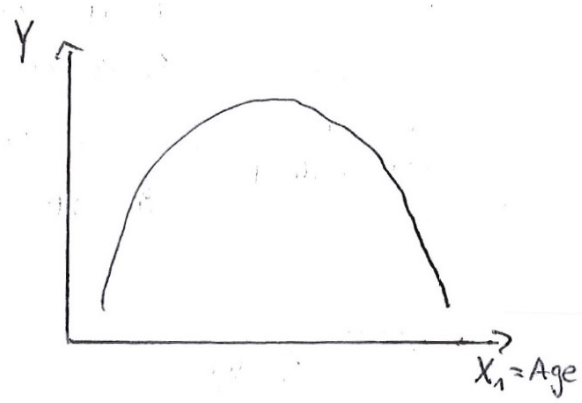
Pro: More flexible

Con:

1. Implicitly adds many variables (K-1 binary variables for K age groups) $\rightsquigarrow$ complex model
2. Piecewise constant (no variation in subgroup) $\rightsquigarrow$ can be generalized to the piecewise linear model.
3. Non-differentiable
4. Uses no information from ’neighboring’ age groups (e.g., coefficient for group 54-60 uses no information from people aged 48-54)

Better alternatives:

1. Use linear model with polynomials of Age



$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2^2 + \varepsilon$$

2. Use a non-linear/non-parametric model (e.g. K-nearest neighbors)

Classification (Chapter 4)

- Predicting qualitative (aka categorical) variable is often called 'classification' ⇝ Need to assign each observation to one category, or class.
- Examples:
  - Predict education of a person (e.g. vocational degree, BSc, MSc, ...)
  - Income categories (if exact numerical income is not available)
  - Acceptance to a university program
- Classification methods often model the probability of each category, and then classify based on the probabilities
- Methods we discuss here:
  - Logistic regression
  - Linear Discriminant Analysis
  - K-nearest Neighbors
- Later in the course (more computer-intensive):

- Trees
- Random Forests
- Boosting

Why not linear regression (4.2)

Example: Predict a person's choice of bus vs. train vs. car for commuting to work/university.

Suppose we have a qualitative variable Y for transportation choice, i.e.

$$Y = \begin{cases} 1 \text{ if train} \\ 2 \text{ if car} \\ 3 \text{ if bus} \end{cases}$$

What if we used least squares/linear model to predict Y?

$$Y = \beta_0 + \beta_1 \underbrace{X_1}_{\text{age}} + \ldots + \beta_p \underbrace{X_p}_{\substack{\text{commuting}\\ \text{distance}}} + \varepsilon$$

Here, the linear model would be based on two unrealistic assumptions

- Ordering in transportation choices (train < car < bus???)
- Numerical interpretation (train = 1/2, car = 1/3 bus???)

⇝ Linear model is not a reasonable choice here

The situation is somewhat better (for the linear model) when there are only two levels e.g.

$$Y = \begin{cases} 1 & \text{if person chooses car} \\ 0 & \text{if not} \end{cases}$$

Here the ordering assumption is less implausible (can be viewed as likelihood of choosing 'car')
⇝ Linear model would predict 'car' if $\hat{Y} > 0.5$.
One can also show: Predictions stay the same if we swap the labels (i.e. 0 if car, 1 if no car).

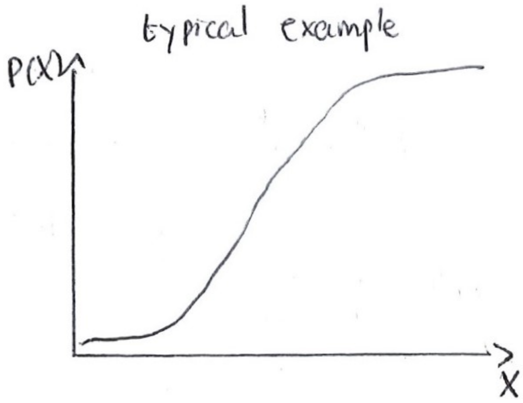
Con: Can have predictions $\hat{Y} < 0$ or $\hat{Y} > 1$, hence interpretation as probability no longer holds.
↳ Better alternative: Logistic regression

Consider a binary response $Y \in \{0,1\}$, e.g. $Y = \begin{cases} 1 \text{ if email is spam} \\ 0 \text{ if not} \end{cases}$

For simplicity, let's consider a single predictor X = Percentage of capital letters in subject line

Logistic regression model sets

$$P(Y=1|X) = P(X) = \frac{\exp(\beta_0 + \beta_1 X)}{1 + \exp(\beta_0 + \beta_1 X)} \in (0,1)$$


typical example

(can also have decreasing probability)

Note that

$$\underbrace{\log\left(\frac{p(x)}{1-p(x)}\right)}_{\text{log-odds ration } \in (-\infty,\infty)} = \beta_0 + \beta_1 X$$

10. Lecture May 30th

Logistic Regression Mode (4.3.1)

Binary response $Y \in \{0,1\}$, e.g.
$Y = 1$ referendum passes (quorum + majority in favor),
$Y = 0$ referendum fails.
Regressor X, e.g.

- share of male voters
- average education level of voters
- voter turnout (percentage of population that participates in referendum)

Aim: Model $Pr(Y = 1|X) = p(x)$
Logistic regression model $p(x) = \frac{\exp(\beta_0+\beta_1 X)}{1+\exp(\beta_0+\beta_1 X)} \in (0,1)$
$\log \frac{p(x)}{1-p(x)} = \beta_0 + \beta_1 X \in (-\infty, \infty)$

Furthermore,

$$\begin{aligned}\frac{\mathrm{d}p(x)}{\mathrm{d}x} &= \frac{\beta_1 \exp(\ldots)\{1+\exp(\ldots)\} - \beta_1 \exp(\ldots)\exp(\ldots)}{(1+\exp(\beta_0+\beta_1 X))^2} \\ &= \frac{\beta_1 \exp(\ldots)}{(1+\exp(\ldots))^2} \begin{cases} > 0 \text{ if } \beta_1 > 0 \\ = 0 \text{ if } \beta_1 = 0 \\ < 0 \text{ if } \beta_1 < 0 \end{cases}\end{aligned}$$

E.g. if we find, that $\beta_1 > 0$, that implies positive association between X and $p(X)$
$\rightsquigarrow$ can interpret sign of β_1!
Note:
$\frac{\mathrm{d}p(x)}{\mathrm{d}x}$ (size of effect of X) depends on X $\rightsquigarrow$ Different for each observation in sample. $\rightsquigarrow$ can not easily interpret size of β_1!

Estimating the regression coefficients (4.3.2)

- For OLS/linear model: There exists an explicit formula for the coefficient estimates. $\rightsquigarrow$ convenient situation, arises because the model is linear in β
- Logistic model no longer linear in β $\rightsquigarrow$ cannot use the OLS estimator
- Instead, use maximum likelihood (ML) estimator
  - Idea: Look for parameter values β_0, β_1 such that the predicted probability corresponds as closely as possible to the oberserved outcome (i.e., probability should be as small as possible if $Y = 0$ and as large as possible if $Y = 1$)
  - Formally: Use likelihood function

$$l(\beta_0, \beta_1) = \underbrace{\prod_{i:Y_i=1} p(X_i)}_{\text{observations for which } Y=1 \rightsquigarrow \text{ want } p \text{ as large as possible}} \underbrace{\prod_{j:Y_i=0} (1 - p(X_j))}_{\text{observations for which } Y=0 \rightsquigarrow \text{ want } p \text{ as small as possible}}$$

where $p(X_i)$ and $p(X_j)$ depend on β_0, β_1

Note:

- In practice, we use the log likelihood function, which has better numerical properties.
- Parameters $\hat{\beta}_0, \hat{\beta}_1$, that maximize $\ln l(\beta_0, \beta_1)$ are called maximum likelihood estimator
  - Good theoretical properties (consistent, low variance)
  - Often easy to use in practice (e.g., *R*'s glm function)

Making predictions (4.3.3)

Example in book: $Y \in \{0, 1\}$, $Y = 1$ means that somebody defaults on a credit.
$\rightsquigarrow$ Want to predict the probability of default, given X ('balance')
In practice: Run MLS to get $\hat{\beta}_0, \hat{\beta}_1$, then we use

$$\hat{p}(X) = \frac{\exp(\hat{\beta}_0 + \hat{\beta}_1 X)}{1 + \exp(\hat{\beta}_0, \hat{\beta}_1 X)})$$

Multiple logistic regression (4.3.4)

Consider using several predictore $X_1, \ldots, X_p$, such that

$$\begin{aligned} \log \frac{p(x)}{1-p(X)} &= \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p \\ \Leftrightarrow \quad p(X) &= \frac{\exp(\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p)}{1 + \exp(\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p)} \end{aligned}$$

Credit example: X_1 = balance, X_2 = income, X_3 = student (yes/no)
$\rightsquigarrow$ Especially helpful if predictors $X_1, \ldots, X_p$ are correlated
Credit example (Figure 4.3):

- Students have higher balance values on average
- For <u>given</u> balance, students are less likely to default.

$\rightsquigarrow$ multiple regression takes care of these effects by allowing for 'all else equal' interpretation.
Yes/no variable as regressor (e.g. X_3 = student yes/no)

$$\hat{p}(X) = \begin{cases} \frac{\exp(\hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2)}{1 + \exp(\hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2)} & \text{non-students} \\ \frac{\exp(\hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2) + \hat{\beta}_3}{1 + \exp(\hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \hat{\beta}_3)} & \text{students} \end{cases}$$

Logistic regression with > 2 response classes (4.3.5)

E.g. three categories: Default vs. renegotiate vs. pay
$\rightsquigarrow$ Model each of <u>three</u> probabilities via logistic regression
$\rightsquigarrow$ Can be done, but not covered in this course
$\rightsquigarrow$ Instead, look at discriminant analysis (classification method that generalizes more easily to > 2 classes)

11. Lecture June 7th

Logistic regression: Model $P(Y = 1|X)$
Alternative method: Linear discriminant analysis (LDA)

- Model $f(X|Y = 1)$
- Model $f(X|Y = 0)$

$\rightsquigarrow$ then flip around to get $P(Y = 1|X)$
Why do we need that?

- Two situations where logistic regression doesn't work well:
  1) If classes ($Y = 0$ and $Y = 1$) are well separated, LR coefficients may be unstable,
  2) if the number of observations in training sample is small
- LDA can be easily generalized to K classes ($Y = 1, 2, \ldots, K$)

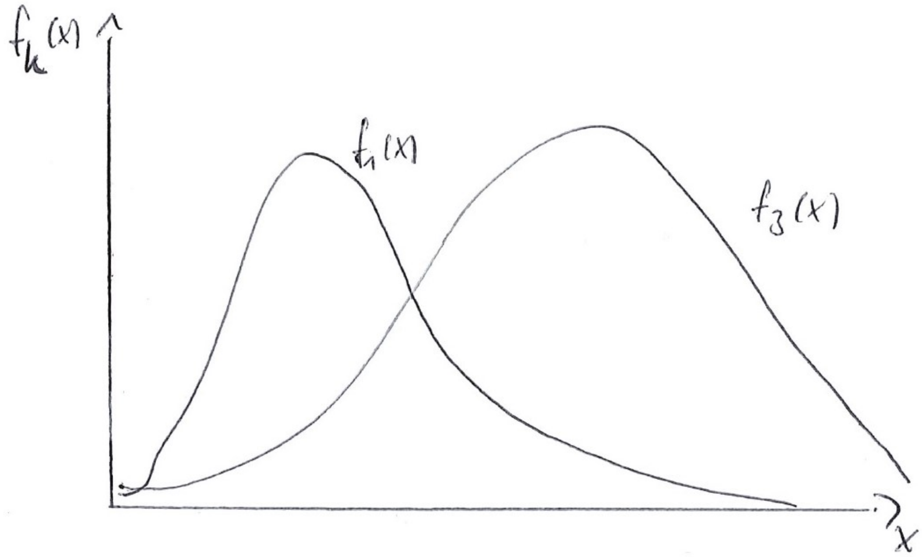
Using Bayes' Theorem for Classification

Setup: Y is a qualitative variable with $K \geq 2$ unordered classes, e.g.

- $Y = 1, 2, \ldots, K$ different types of images
- $Y = 1, 2, \ldots, K$ different types of cars (marketing)

Let π_K be the prior probability (before seeing X) that $Y = \underbrace{K}_{\text{one of the classes}}$
Let $f_K(X)$ denote the $\underline{\text{density function}}$ of X for an observation from the K-th class
$\rightsquigarrow$ $f_K(x)$ large if there's a high probability that an observation from class K has $X \approx x$

E.g. X = income, $Y \in \{\underbrace{\text{cheap car}}_{1}, \underbrace{\text{medium-price car}}_{2}, \underbrace{\text{expensive car}}_{3}\}$



Bayes' theorem states that

$$Pr(Y = K|X = x) = \frac{\pi_K f_k(x)}{\sum\limits_{l=1}^{K} \pi_l f_l(x)} \tag{12}$$

⇝ 'Posterior Probability' after seeing X
⇝ Can estimate $P_K(x) \equiv Pr(Y = K|X = x)$ by inserting estimates for $\{\pi_l\}_{l=1}^{K}$ and $\{f_l\}_{l=1}^{K}$ into (12)

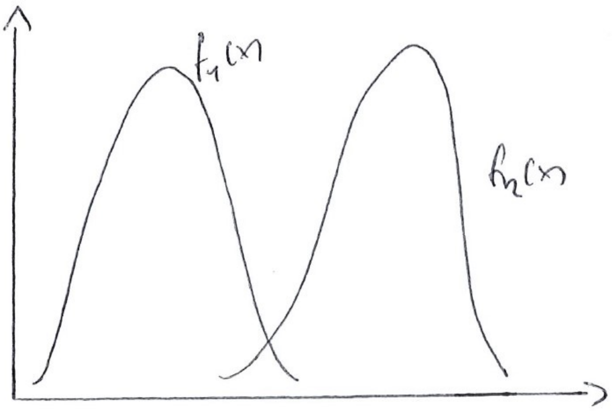
- Estimating the $\{\pi_l\}$ is easy: Can use fraction of training data that fall into each category
- Estimating the densities $\{f_l(x)\}$ is more challenging, but can be simplified with adding assumptions

⇒ Next: Discuss approaches for estimating the terms in (12), with the aim to approximate the Bayes classifier.

Linear discriminant analysis for $p = 1$ (4.4.2)

Suppose that $f_K(x)$ is normal with

$$f_K(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{(x-\mu_k)^2}{2\sigma^2}\}$$



⇒ All densities have the same variance but different μ_k

⇒ All densities have the <u>same variance</u> but different mean μ_K
Plugging that into (12), we get

$$P_K(x) = \frac{\pi_K \exp\{-1/2\sigma^2(x-\mu_K)^2\}}{\sum\limits_{l=1}^{K} \pi_l \exp\{-\frac{1}{2\sigma^2}(x-\mu_K)^2\}} \tag{13}$$

⇝ Choose class K for which $P_K(X)$ is largest!

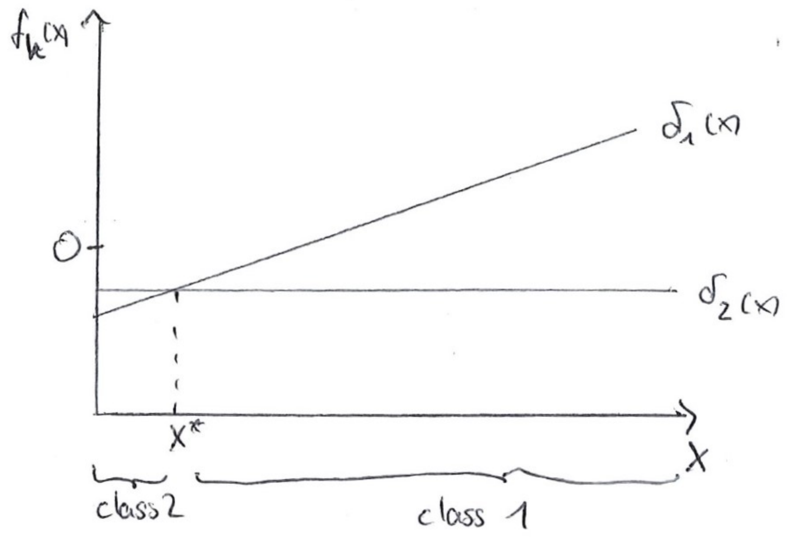
- We can forget about the denominator of (13): He's the same number for each class $K = 1,\ldots,K$
- Hence pick class that maximizes $\pi_K \exp\{-\frac{1}{2\sigma^2}(x-\mu_K)^2\}$
- <u>Equivalently</u>: Pick class that maximizes

$$\log\pi_K - \frac{(x-\mu_K)^2}{2\sigma^2} = \log\pi_K - \underbrace{\frac{x^2}{2\sigma^2}}_{\text{does not depend on } K} + \frac{x\mu_K}{\sigma^2} - \frac{\mu_K^2}{2\sigma^2}$$

⇒ Pick K that maximizes:

$$\log \pi_K + \frac{x\mu_K}{\sigma^2} - \frac{\mu_K^2}{2\sigma^2} \equiv \delta_K(x)$$

Example: k=2



LDA decision rule: Classify as 1 if X exceeds some threshold value X*

In practice we don't know μ_K and σ^2 ⇒ Use estimates (average of X for observations in class K):

$$\begin{aligned} \hat{\mu}_K &= \frac{1}{n_K} \sum_{i:Y_i=K} X_i \\ \hat{\sigma}^2 &= \frac{1}{(n-K)} \sum_{k=1}^{K} \sum_{i:Y_i=K} (X_i - \hat{\mu}_K)^2, \end{aligned}$$

where n = # of training data and n_K = # of training data in class K.

12. Lecture June 13th

see slides

Review: Discriminant Analysis

Goal: Classify $Y \in \{1,2,\ldots,K\}$ based on predictor X

Bayes Formula:

$$Pr(Y=K|X=x) = \frac{\pi_K f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}, \tag{14}$$

where π_k are prior probabilities, and $f_k(x)$ are group-specific densities for X.

Needed : Assumptions/estimates for π_k and $f_k(x)$.

Assume that f_k's are normal with class-specific mean μ_k and common variance σ^2, i.e.

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

Decision rule: Choose k for which

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

is largest. In practice: Use esimates $\hat{\mu}_k$, $\hat{\sigma}^2$

Example: Figure 4.4 in the book

LDA for $p > 1$

Equation (14) can also be used when $p > 1$, i.e. many predictors. In that case, $X = [X_1, \ldots, X_p]'$ is a vector.

Density $f_k(x)$ assumed to be multivariate normal, i.e.

$$f_k(x) = \frac{1}{(2p)^{p/2}|\Sigma|^{1/2}} \exp(-0.5(x-\mu_k)^T\Sigma^{-1}(x-\mu_k))$$

Note: Covariance matrix Σ is the same for all k!

Background: Multivariate normal

- Mean vector μ gives mean for each element of $X = (X_1, \dots, X_p)^T$
- Covariance matrix Σ gives (co-)variances of X, with

$$\Sigma_{[I,j]} = Cov(X_I, X_j) \text{ and } \Sigma_{[I,I]} = Var(X_I)$$

- example pictures for density, see book Figure 4.5

Classification rule: Assign to class k for which

$$\delta(x) = x^T \Sigma^{-1} \mu_k - 0.5 \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$$

is largest.
In practice: Use estimates $\hat{\mu}_k$ and $\hat{\Sigma}$

Example: Figure 4.6

Quadratic Discriminant Analysis

- Now assume that $X|Y = k$ is multiv. normal with mean μ_k and covariance matrix Σ_k.
- $\neq$ LDA, where the covariance is the same for all k
- For QDA, discriminant function $\delta_k(x)$ becomes a quadratic function of x (see eq. 4.23 in the book)

Example: Figure 4.9

QDA: Numerical example

Consider QDA with a single predictor ($p = 1$)

- Formula for $\delta_k(x)$?
- Consider the case of only $K = 2$ classes, and let
  - $\pi_1 = \pi_2 = 0.5$
  - $\mu_1 = 0$, $\sigma_1^2 = 4$
  - $\mu_2 = 1$, $\sigma_2^2 = 1$

- Sketch $f_1(x), f_2(x)$ in a single plot
- Which values for x get assigned to class 1?

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

$$P(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum\limits_{l=1}^{K} \pi_l f_l(x)} \tag{15}$$

$\rightsquigarrow$ Choose K which maximizes (15), given QDA assumptions
maximize $\pi_k f_k(x)$

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

$\rightsquigarrow$ Choose K that maximizes

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

Now let $K = 2$, $\pi_1 = \pi_2 = 0.5$, $\mu_1 = 0, \mu_2 = 1$, $\sigma_1^2 = 4, \sigma_2^2 = 1$

$\delta_1(x) = \log(1/2) - \log(2) - \frac{(x-0)^2}{8} = -2\log(2) - \frac{x^2}{8}$
$\delta_2(x) = -\log(2) - \log(1) - \frac{(x-1)^2}{8} = -\log(2) - \frac{(x-1)^2}{8}$

Classify to $K = 1$ iff

$$\begin{aligned}
\delta_1(x) &> \delta_2(x) \\
\Leftrightarrow \delta_1(x) - \delta_2(x) &> 0 \\
\Leftrightarrow -\log(2) - \frac{x^2}{8} + \frac{(x-1)^2}{2} &> 0 \\
-\log(2) - \frac{1}{8}x^2 + \frac{x^2}{2} - \frac{2x}{2} + \frac{1}{2} &> 0 \\
(-\frac{1}{8} + \frac{4}{8})x^2 - x + \frac{1}{2} - \log(2) &> 0
\end{aligned}$$

$\Rightarrow$ Find x^* such that $\delta_1(x^*) - \delta_2(x^*) = 0$

$$x^* = \frac{1 \pm \sqrt{1 - \frac{3}{2} \cdot c}}{\frac{3}{4}} = \{-0.181; 2.847\}$$

QDA when p is large

- One covariance matrix contains

$$0.5p(p+1)$$

  parameters

- Need one for each class...
- Hence, large training data set needed

Comparison of classification methods

- LDA and logistic regression are quite similar
  - Both have linear decision boundaries, see p. 151 in book
  - Slightly different assumptions, reflected in estimation
- Flexibility: QDA beats LDA
- Stability: LDA beats QDA
- KNN can be good, but K must be chosen with care
  - Bias-variance trade-off
  - Cross-validation methods

Simulation example in the book (p.152)

Six scenarios; $p = 2$ predictors in each.

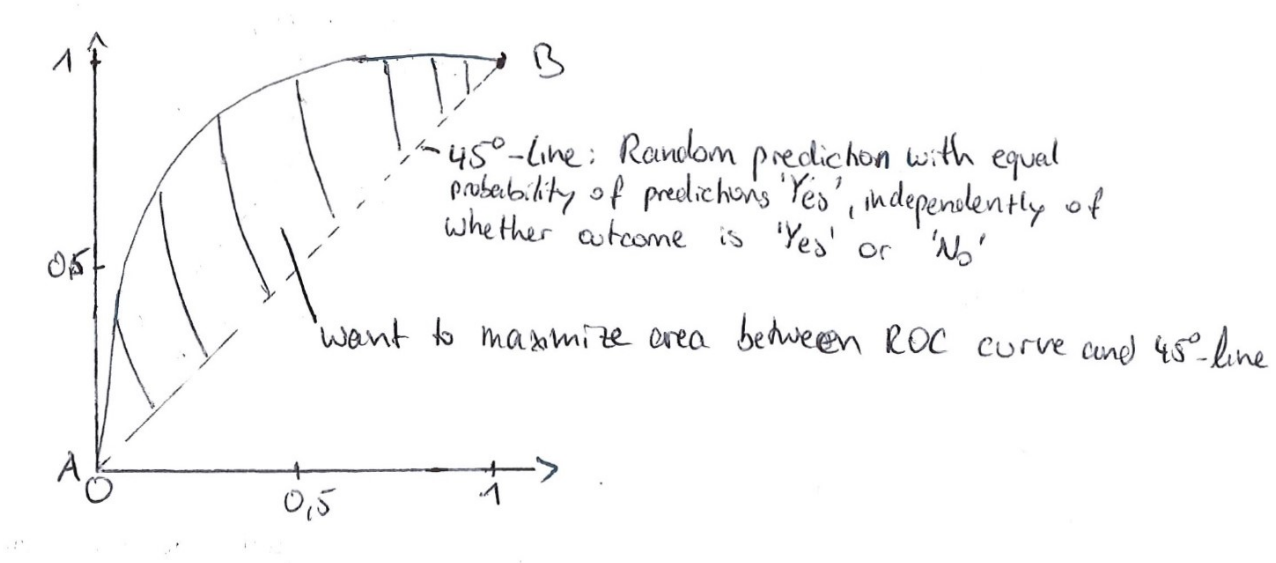
- Scenarios 1/2: X follows a multivariate normal (as in LDA assumptions)

13. Lecture June 14th

False positive rate = $\frac{\#\{\text{Prediction = Yes, Outcome = No}\}}{\#\{\text{Outcome = No}\}}$ = Probability of predicting 'Yes', given that outcome is 'No'

True positive rate = $\frac{\#\{\text{Prediction = Yes, Outcome = Yes}\}}{\#\{\text{Outcome = Yes}\}}$ = Probability of predicting 'Yes', given that outcome is 'Yes'.

ROC curve:


1
0,5
A
0
0,5
1
B
45°-line: Random prediction with equal probability of predictions 'Yes', independently of whether outcome is 'Yes' or 'No'
want to maximize area between ROC curve and 45°-line

A: Always predict 'No' ⇝ no false positives, no true positives

B: Always predict 'Yes' ⇝ FP rate = TP rate = 1

⇝ Good predictor attains good tradeoff ⇝ Area between ROC curve and 45°-line is large

Note:

- Statistical model (like LDA or logistic regression) gives a set of probabilities $\{p_i\}_{i \in I}$, $I = \{1, \dots, n\}$, where

$$\hat{p}_i = (\widehat{Prob} \text{ obs. } i = \text{Yes})$$

- Classification/prediction rule: Classify observation i as 'Yes' if $\hat{p}_i > \tau$
- For each choice of τ, we get a certain point on the ROC curve (i.e. FP rate, TP rate)
- If the model is very good, each choice of τ leads to some point 'far away' from the 45°-line
  (except that $\tau = 0$ always leads to point B, and $\tau = 1$ always leads to point A)

Resampling Methods (Chapter 5)

Broad idea:

- Draw repeated samples from training data
- Fit a given statistical model in each iteration
- Examine model fit

Main purpose: Estimate a model's test sample performance (cross-validation)

Cross validation (5.1)

Question: How to estimate the test-sample performance of a given model (say, $\hat{f}$)?

- Simply using the training sample performance of $\hat{f}$ as a proxy for the test sample performance is a bad idea (the two may differ a lot, e.g. due to overfitting)
- Some methods (e.g. adjusted R^2) make a mathematical adjustment to training sample performance in order to estimate test sample performance $\rightsquigarrow$ see chapter 6 in book
- Here: Discuss cross-validation as an estimator for test sample performance

Validation set approach (5.1.1)

Randomly split sample in two parts (training and validation parts). Fit the model to the training part, and see how it performs on the validation sample.

$\rightsquigarrow$ see figure 5.1 in the book

Pro: Easy to understand and to implement

Contra: You obtain a pessimistic estimate of the model performance (you use only n/2 rather than n observations for model fitting). The Estimate of the test error is quite variable, depends on random split into training vs. validation sample.

Leave-one-out Cross-validation (5.1.2)

Idea: Fit model to observations $i = 2, \ldots, n$, then compute $MSE_1 = (Y_1 - \hat{Y}_1)^2$. Then use observations $i = 1, 3, 4, \ldots, n$, and compute $MSE_2 = (Y_2 - \hat{Y}_2)^2$

Repeating this n times gives n squared errors, $MSE_1, \ldots, MSE_n$.
The LOOCV estimate of the test error is then given by

$$CV_{(n)} = \frac{1}{n} \sum_{i=1}^{n} MSE_i$$

Pro: (Compared to validation set approach)

- Get realistic estimate of test error (use $n - 1 \approx n$ observations for model fitting)
- No randomness in training/validation splits

Con:

- Need to fit model n times $\rightsquigarrow$ may take very long time

14. Lecture June 20th

Cross-Validation: Tool to estimate a model's performance for new (AKA test) data

K-fold Cross-Validation (5.1.3)

Idea: Divide data into K groups (or folds) of approximately equal size. Then use the first group as a validation set, and groups 2 to K for model fitting. Compute MSE as the average squared error in group 1.
Repeat this process K times, using a different group as validation set each time. The K-fold Cross Validation estimate is then given by

$$CV_{(K)} = \frac{1}{K}\sum_{i=1}^{K} MSE_i$$

Note that LOOCV (Leave-One-Out-Cross-Validation) is a special case of K-fold Cross Validation, with $K = n$. In practice, one often uses $K = 5$ or $K = 10$.
Pro:

- Much faster than LOOCV (need to refit model 5 or 10 times instead of n times)
- Estimate of the test error has a lower variance than for LOOCV (see below).

Example: Want to compare linear regression to a more fancy method.

- Run K-fold CV for linear regression, get estimate $CV_{(K)}^{LR}$ (LR = linear regression)
- Do the same for the fancy method, get $CV_{(K)}^{FM}$
  $\rightsquigarrow$ Choose method with better (=lower) $CV_{(K)}$ estimate.

Bias-Variance Trade-Off for K-fold Cross-Validation (5.1.4)

- LOOCV gives nearly unbiased estimate of the test MSE (use $n-1 \approx n$ observations for the model fitting)
- With K-fold CV, we use $\frac{K-1}{K} \times n$ observations for the model fitting
$\rightsquigarrow$ introduces some bias since we are using fewer than n observations for fitting.
- In LOOCV, we average over n training samples that are almost identical, hence the MSE estimates are highly correlated. In K-fold Cross Validation, we average over K training samples that have less overlap, and are thus less correlated.

$\rightsquigarrow$ To sum up, K-fold CV gives an estimator of test MSE that has (i) more bias and (ii) less variance than the estimator from LOOCV.
In practice, $K=5$ or $K=10$ often work well.

Note: For logistic regression, one can replace the MSE criterion by the log-likelihood function. One could also use the MSE for probabilities, though: If $\hat{p}_j$ is the predicted probability for observation j, and observation $Y_j \in \{0,1\}$, then

$$(Y_j - \hat{p}_j)^2 = \begin{cases} (1-\hat{p}_j)^2 & \text{if } Y_j = 1 \\ \hat{p}_j^2 & \text{if } Y_j = 0 \end{cases}$$

15. Lecture June 21st

Chapter 6: Linear model selection and regularization

Setup: Regression model

$$Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + \varepsilon \tag{16}$$

(e.g. Y = Goal difference between Team 1 and Team 2, $X_1, \ldots, X_p$ encode information on both teams, such as performance, betting odds, info on players, etc.)
$\rightsquigarrow$ p can be quite large

- often unclear ex ante which variables are informative
- considering interactions and polynomials further increases p

Until now: Fitted Equation (16) with OLS. This chapter: Look at fitting procedures that are better in terms of prediction accuracy and/or interpretability.

- Prediction accuracy: If there are many predictors (p is large), OLS prediction has large variance.
  Solution: Decrease variance by $\underbrace{\text{constraining}}_{\text{e.g. set to zero}}$ or $\underbrace{\text{shrinking}}_{\text{e.g. shrink toward zero}}$ some of the coefficient estimates $\hat{\beta}_j$
- Interpretability: In many data sets, coefficients $\beta_1, \ldots, \beta_p$ contain irrelevant variables that are not associated with Y. The presence of many variables j with β_j 'close to zero' (but not exactly zero) makes OLS hard to interpret.
  Solution: Variable methods for choosing a relevant subset of predictors.

Subset Selection (6.1)

Idea: Find the best subgroup of all p possible vectors.

Algorithm 6.1(from the book): Best Subset Selection

1. Let M_0 = null model (no predictors, i.e. $\hat{Y} = \overline{Y}$ (training sample average))
2. For $K = 1, 2, \ldots, P$
   (a) Fit all $\binom{P}{K}$ models that contain exactly K predictors
   (b) Let M_K be the best of theses models (in terms of R^2)
3. Select a simple best model from among $M_0, \ldots, M_P$ using CV (or adjusted R^2)

Note: Adjusted $R^2 = 1 - \frac{RSS/(n-d-1)}{TSS/(n-1)}$,
where $RSS = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$, $TSS = \sum_{i=1}^{n} (Y_i - \overline{Y})^2$, $d = \#$ of predictors
$\rightsquigarrow$ Like R^2, but imposes a penalty on the # of predictors, d
$\rightsquigarrow$ Fixes the tendancy of R^2 to select overfitted models
$\rightsquigarrow$ Like CV, adjusted R^2 is an estimate of a model's test error (adjusted R^2 simpler, CV tends to be more accurate)

16. Lecture June 28th

Stepwise selection (6.1.2)

Best Subset Selection runs into problems when p gets large:
Need to evaluate many (!) combinations of variables $\rightsquigarrow$ very expensive in terms of computer time
$\rightsquigarrow$ One fix: Forward step-wise selection

Algorithm 6.2 (from the book): Forward stepwise selection

1. Let M_0 denote the null model without any predictors
2. For $K = 0, \ldots, p-1$
   a) Consider all models that add one additional predictor to M_K
   b) Choose the best among these $p-K$ models, say M_{K+1} (use R^2 as criterion)
3. Select a single best model from $M_0, \ldots, M_p$ using CV or adjusted R^2 (or BIC, Akaike,...)

Note: This is much simpler than Best Subset Selection: At step K, we need to consider only $p-K \ll \binom{p}{K}$ models.

Alternative Strategy to deal with many predictors (i.e., large p):
Shrinking the coefficients towards zero $\rightsquigarrow$ We'll look at two shrinkage methods, Ridge Regression and Lasso.

Ridge Regression (6.2.1)

OLS minimizes

$$RSS = \sum_{i=1}^{n}(Y_i - \beta_0 - \sum_{j=1}^{p}\beta_j X_{ij})^2$$

Ridge Regression minimizes

$$RSS + \lambda \sum_{j=1}^{p}\beta_j^2,$$

where $\lambda \geq 0$ is a tuning parameter.
The term $\lambda \sum_{j=1}^{p} \beta_j^2$ is called a shrinkage penalty (SP)

- Small when β_j's are close to zero
  ↳ Hence, sets the incentive to choose 'smaller' β_j's
- $\lambda = 0$, No SP, gets OLS estimates
- λ large: SP grows, coefficients get smaller (as $\lambda \to \infty$, all β_j's get to zero)
- Get different set of coefficients for each possible value of λ
- By construction, β_j's are not exactly equal to zero (cannot easily tell which variables are 'in' or 'out')
  $\rightsquigarrow$ makes sense for prediction, but not for interpretation
- Good choice of λ is crucial, can be done using CV

Implementation:

- SP does not affect the intercept, β_0

- Before running Ridge Regression, one should standardize the predictors using the following formula

$$\widetilde{X}_{ij} = \frac{X_{ij} - \overline{X}_j}{\sqrt{\frac{1}{n}\sum\limits_{i=1}^{n}(X_{ij} - \overline{X}_j)^2}}$$

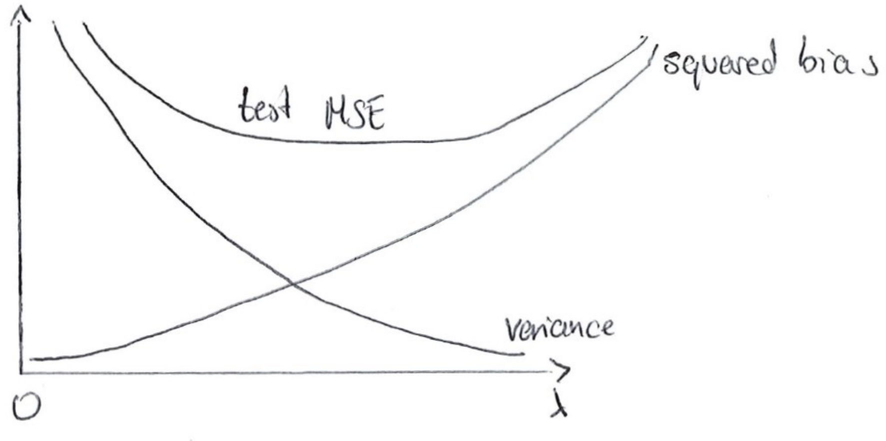
  with $\overline{X}_j = \frac{1}{n}\sum\limits_{i=1}^{n} X_{ij}$
  $\rightsquigarrow \widetilde{X}_{ij}$ has mean 0 and variance 1.

Why does Ridge improve upon OLS? E.g., suppose the true model is linear, but there are many predictors.

OLS ($\lambda = 0$)	Ridge ($\lambda > 0$)
no bias	bias
higher variance	lower variance

In general, increase in λ leads to an increase in (squared) bias, a decrease in variance and an increase or decrease in MSE

↳ Use CV to find good choice of λ



Possible factors determining optimal choice of λ

- If predictors are noisy, and have little explanatory power: Expect large λ^*, i.e. don't use predictors a lot, set coefficients to 'almost zero'
- If training sample size (n) is large: Expect small value λ^*, since estimation noise plays a smaller role

Drawback of Ridge: Includes all p predictors in final model

- Penalty $\lambda \sum_{j=1}^{p} \beta_j^2$ shrinks coefficients toward zero, but not exactly equal to zero
- Not a problem for prediction, but for interpretation (makes it harder to explain model choice)

↳ Seek a method that keeps benefits of Ridge (shrinkage/prediction) but is easier to interpret ↳ Fix: Lasso (least absolute shrinkage and selection operator)

Lasso coefficients minimize

$$\underbrace{\sum_{i=1}^{n}(Y_i - \beta_0 - \sum_{j=1}^{p} \beta_j X_{ij})^2}_{RSS} + \lambda \sum_{j=1}^{p} |\beta_j|$$

Note: Lasso penalty $\neq$ Ridge penalty = $\lambda \sum_{j=1}^{p} \beta_j^2$

Lasso penalty is called an L_1 penalty, since L_1 norm of $\beta = (\beta_1, \ldots, \beta_p)$ is given by $\|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$

Effect of L_1 penalty: When λ is sufficently large, some of the coefficients get kicked out completely (i.e., $\beta_j = 0$ for these coefficients). Hence Lasso performs both variable selection and shrinkage, and is thus easier to interpret than Ridge.

17. Lecture July 3rd

Another formulation for Ridge and Lasso

One can show, that Ridge Regression solves the following problem:

$$\min_{\beta}\underbrace{\{\sum_{i=1}^{n}(Y_i-\beta_0-\sum_{j=1}^{p}\beta_j X_{ij})^2\}}_{=RSS(\beta)} \quad \text{subject to} \quad \sum_{j=1}^{p}\beta_j^2 \leq s$$

Similarily, Lasso solves

$$\min_{\beta} RSS(\beta) \quad \text{subject to} \quad \sum_{j=1}^{p}|\beta_j| \leq s$$

<u>Notes</u>:

- s is a 'budget' that we can 'spend' on the coefficients $\beta_1,\ldots,\beta_p$
- $s\to\infty$: Constraint is non-binding, and we end up at the OLS solution for β
- $s=0$: Get $\beta_0\neq 0$, $\beta_1=\ldots=\beta_p=0$
- More generally: There is a one-to-one correspondence between λ (weight on the coefficient penalty, we used before) and s.
  That means for each s we can find a value of λ that results in the same Ridge (or Lasso) coefficients. Larger values of s (large budget) correspond to smaller values for λ (small penalty).

Best Subset Selection (BSS) can also be written as a 'budget' problem: It solves

$$\min_{\beta} RSS(\beta) \quad \text{subject to} \quad \sum_{j=1}^{p}\mathbb{1}(\beta_j\neq 0)\leq s$$

In words: BSS chooses best model with at most s nonzero coefficients.

$\rightsquigarrow$ As noted, earlier, BSS is tedious since it entails $\binom{p}{s}$ estimation problems.

$\rightsquigarrow$ Lasso/Ridge are feasible alternatives to BSS.

Comparing Ridge and Lasso

- Lasso is easier to interpret than Ridge
  ↳ can look at set of 'selected' variables

- Which one is better at prediction?
  Hard to tell, ultimately an empirical question. Ridge 'assumes' that there are many variables with small effects/coefficients, Lasso assumes, that there are a few key variables with big effects/coefficients (sparsity).

Figure 6.7 (book): Graphical illustration of budget formulation for Lasso (left) and Ridge (right).
Lasso: Budget represented as a diamond type object
Ridge: Budget represented as a circle
Coefficient solution = tangential point between budget and RSS contours (= lowest RSS value that's within the budget).
$\rightsquigarrow$ Figure illustrates that the Lasso budget is more likely to yield zero solutions (for β_1 in this case)

High-Dimensional Data (6.4.1)

- Traditional Statistics: Large n, small p
  - Medicine: Predict blood pressure of n patients based on age, gender, BMI
  - Econ: Predict wage based on a few features (experience, education, industry dummies)
- 21st century: p may be large
  - Often: Much cheaper/easier to increase p than increase n
  - Electronic tracking (e.g. browser, fitness gadgets): No limit on p, what about n?
  - Similarily, text mining
  - Statistical ways to increase p (interactions, polynomials)

What goes wrong in high dimensions? (6.4.2)

Drastic example: $p = n$

- Linear model (OLS) gives percet fit
- I.e., $\hat{y}_i = y_i$ for all i in training sample
- Overfitting, i.e. test sample performance likely very poor

Similar (but less drastic) phenomena if p is just a little smaller than n

Regression in high dimensions (6.4.3)

Linear model $Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_P + \varepsilon$, with large p

- Model fitting: BSS/FSS, Ridge, Lasso much better than OLS
- Complexity reduction via selection (BSS, Lasso) and shrinkage (Ridge, Lasso)

Prediction when p is large: Key Points

- Shrinkage helps!
- Tuning parameter selection (degree of shrinkage, λ) important
- Adding new predictors: Signal or noise
  - Signal helps, noise hurts
  - Ideally: Algorithm separates signal from noise
  - In practice: Common sense helps to design/select features

Interpreting Results in High Dimensions (6.4.4)

Don't overstate results

- Identify subset of useful features, even if they don't form the true model
- Correlation vs. causation

Communication: Report test-sample or cross-validation metrics, not training-sample stuff (p-values, R^2)

Example: Report subset of features that gets selected when $\lambda = \lambda^*$ is selected by CV.

18. Lecture July 4th

Chapter 8: Tree Based-Methods

- Regression or classification trees (Ch. 8.1) are intuitive and have natural interpretation
  - Useful for interpretation/communication
  - See examples in slides
- Prediction accuracy inferoir to other methods
- Ensemble methods (Ch. 8.2) greatly improve trees' prediction accuracy
  - Tree ⇒ Random Forests & co.
  - Prediction accuracy highly competitive, interpretation not straightforward

Figure 8.1 and 8.2

Trees: Terminology

- Tree is defined by binary splits of predictor space
  - Recursively put observations in two bins
- R_1, R_2, R_3 from example are terminal nodes or leaves
- Other (non-terminal) nodes are called internal nodes
  - Branches connect several nodes (e.g. left vs right branch)

Prediction via trees

- Basis: Predictor space, i.e. set of possible values for $X_1, \dots, X_p$
- Tree prediction:
  - Divide predictor space into J distinct and non overlapping regions, $R_1, \dots, R_J$
  - Prediction is the same for all observations that fall into a region R_j: Mean response for all observations in R_j

RSS of tree:

$$\underbrace{\sum_{j=1}^{J} \sum_{i:x_i \in R_j} (y_i - \hat{y}_{R_j})^2}_{= \text{RSS within leaf } R_j},$$

$\hat{y}_{R_j}$ = prediction for observation in leaf R_j (same for all $i : x_i \in R_j$)

How to find the partitions $R_1, \ldots, R_J$

Goal: Find partitions which minimize RSS,

$$\sum_{j=1}^{J} \sum_{i:x_i \in R_j} (y_i - \hat{y}_{R_j})^2$$

Note: prediction is the same for all $i \in R_j$!

Given choice of leaves $R_1, \ldots, R_J$: optimal to set

$$\hat{y}_{R_j} = \frac{1}{|R_j|} \sum_{i:x_i \in R_j} y_i, = \text{ mean of } y_i \text{ within leaf } R_j$$

$|R_j|$ = number of observations within leaf R_j

Recursive binary splits

- Find, best split given all previous splits
- ’Greedy’, i.e. don’t incorporate possible future splits
- Procedure: Find variable j and split point s which minimize

$$RSS(j,s) = \sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

  where $R_1(j,s)$ and $R_2(j,s)$ are the two ’buckets’ that result from using j and s for splittiing:

$$R_1 = \{X | X_j < s\}, \quad R_2 = \{X | X_j \geq s\}$$

Splitting always reduces RSS

(very) simple example: Only one predictor ($p = 1$), two observations ($n = 2$)

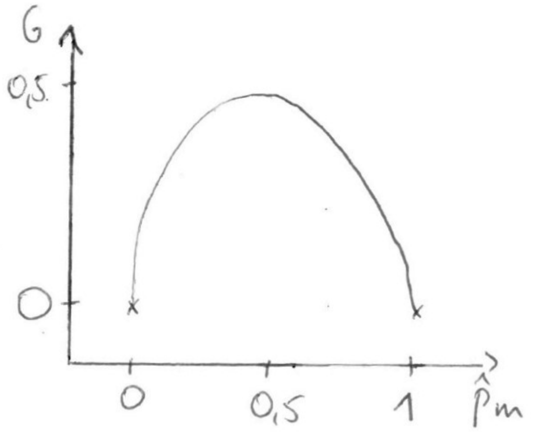
i	x_i	y_i
1	0	2
2	4	5

Before split: RSS = $(2-3.5)^2 + (5-3.5)^2 = 2 \cdot 2.25$
Split: Only have one prediction (X) that can be used for splitting. all 'reasonable split points are $\in (0,4)$ e.g. split at $\{X < 2\}$ vs. $\{X \geq 2\}$
$\rightsquigarrow$ each bucket contains only one observation, RSS = 0

Example shows that splitting always reduces RSS, up to a (crazy) limiting case where each bucket contains exactly one observations

Gini index

$$G = \hat{p}_m(1-\hat{p}_m)$$



E.g. $\hat{p}_m = 0.2 \rightsquigarrow G = 0.16$
$\hat{p}_m = 0.5 \rightsquigarrow G = 0.25$

Idea: Want to have buckets that separate 'zeros' from 'ones' (i.e. sharp prediction)

Finding the optimal tree size

- Size $|T|$ of a tree T can be measured by the number of terminal nodes (three in simple example above)
- Cost-complexity pruning: Grow very large tree T_0, then find optimal subtree $T \subset T_0$ that minimizes

  $$\sum_{m=1}^{|T|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T|$$

  where α is a tuning parameter that penalizes complexity
- α can be determined via cross-validation

Algorithm: (8.1 in the book)

1. Use recusive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations
2. Apply cost complexity pruning to the large tree in order to obtain a sequence of the best subtrees, as a function of α
3. Use K-fold cross-validation to choose α. That is, divide the training observations into K folds. For each $k = 1, \ldots, K$:
   a) Repeat Steps 1 and 2 on a llbut the kth fold of the training data
   b) Evaluate the mean squared prediction error on the data in the left-out kth fold, as a function of α.

   Average the results for each value of α, and pick α to minimize the average error
4. Return the subtree from Step 2 that corresponds to the chosen value of α

Large tree T_0 for the Baseball data

Figure 8.4

MSE plotted against tree size

Figure 8.5

19. Lecture July 5th

Classification Trees

- Like regression tree, but for qualitative varibale
- Main idea is the same
- Focus on binary variable (two classes) here
- Different criteria for making binary splits:
  - Gini index

    $$G = 2\hat{p}_m(1-\hat{p}_m)$$

    where m is the region (or bucket) of interest, and $\hat{p}_m$ is the fraction of ones
  - Idea: Good split (small G) contains only zeroes or ones
  - Alternatively: Cross-entropy, similar idea

Classification Tree: Heart Disease Data

Figure 8.6

Criteria for making binary splits

- Theoretically: Aim to minimize missclassification rate

  $$E = 1 - \max(\hat{p}_m, 1-\hat{p}-m)$$

  However, does not work well in practice
- Smoother alternatives:
  - Gini index,

    $$G = 2\hat{p}_,(1-\hat{p}-m),$$

    where m is the region (or bucket) of interest, and $\hat{p}_m$ is the fraction of ones
  - Cross-entropy

    $$D = -(\hat{p}_m \log(\hat{p}_m) + (1-\hat{p}_m)\log(1-\hat{p}_m))$$
- Idea: Ideal split (small E, G or D) contains only zeros or ones

Trees: Pros

- Easy to explain
- Mirror human decision making (?)
- Easy to draw
- Can easily handle qualitative predictors w/o dummies

Trees: Cons

- Prediction performance not great
- Can be very non-robust (aka high variance)

Bagging, Random Forests, Boosting (8.2)

- Until bow: Trees are nice, but too shaky to forecast well
- Here: Ways to fix this problem
- Will imporve prediction performance, at the cost of interpretation

Bagging regression trees

Want: Prediction at a new (test sample) point x, say $\hat{f}(x)$
If $\hat{f}(x)$ is produced by a single tree, then prediction tends to be shaky
↳ Idea: Compute B predictions from B different trees, and then take the average.
⇝ Question: How to get 'different' trees?
⇝ Answer: Take B random samples from training data, and fit new tree each time
Prediction then given by

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{b}(x),$$

where $\hat{f}^b$ is the tree obtained from the b-th sample.

Bagging (from the slides)

- Stats 1: If random variables $Z_1, \ldots, Z_n$ are indepedent and $V(Z_i) = \sigma^2$, then $\frac{1}{n} \sum\limits_{i=1}^{n} Z_i$ has variance σ^2/n
- Hence, averaging reduces variance
- This intuition suggests to ’somehow’ combine many trees to reduce variance
- Ideally: Compute tree predictions $\hat{f}_{avg}(x), \ldots, \hat{f}^B(x)$ from B different training data sets,then compute low-variance prediction

$$\hat{f}_{avg}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^b(x)$$

- In practice: Only a single training data set
- Pragmatic alternative: Bagging (Bootstrap Aggregating)
- Bootstrap - What’s that?

Bootstrap: Recipe

For data set of size n:

- Draw n numbers (integers) between 1 and n, with replacement
- Get Bootstrap data set by selecting the rows (observations) drawn in the previous step
- Produces synthetic data set, with some of the original observations appearing more than once, others not at all

Bootstrap: Illustration

Figure 5.11

Bagging: Recipe

- Create B bootstrap resamples of the data
- $\hat{f}^{*b}(x)$ = prediction for the bth sample
- Bagging prediction given by

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{*b}(x)$$

- Simple and effective way to reduce variance of prediction

Bagging: Notes

- Can be applied to any procedure $\hat{f}$, but particularly useful for trees
- Why? Trees are very shaky on their own
- Recipe for bagging trees:
  - Each tree $b = 1, \ldots, B$ is grown very large
    - Hence low bias, high variance
    - No pruning, hence no choice for tuning parameter needed
  - Then average over B trees to reduce variance
  - Prediction
    - Regression: See $\hat{f}_{bag}(x)$ above
    - Classification: Do majority vote among B trees

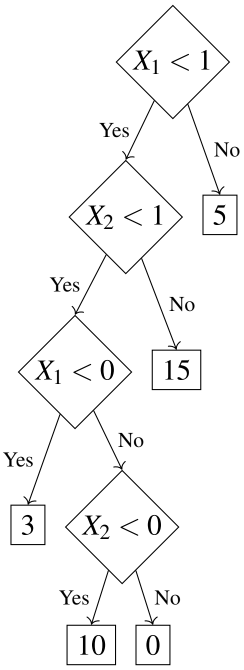
Excercise 5.4/2

(a) Prob(1st bootstrap draw $\neq$ jth obs.) = 1 - Prob(1st bootstrap draw = jth obs.) = $\frac{n-1}{n}$
(b) $\frac{n-1}{n}$
(c) Prob(jth obs. not in bootstrap sample)
= Prob(1st bootstrap draw $\neq j$, 2nd bootstrap draw $\neq j, \ldots,$ nth bootstrap draw $\neq j$)
= $(\frac{n-1}{n})^n = (1-\frac{1}{n})^n$ $\Rightarrow$ Prob(obs. j is in bootstrap sample) = $1-(1-\frac{1}{n})^n$
For $n \to \infty$, $1-(1-\frac{1}{n})^n \to 0.632\ldots$

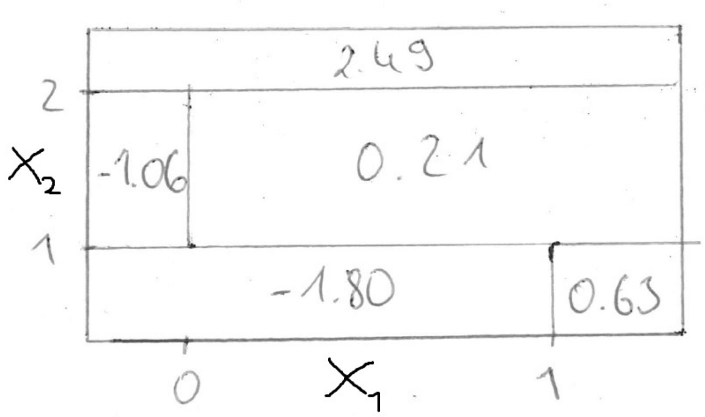
Note: Convergence is quite fast, so that approximations works well even for a ’small’ $n \approx 50$
$\rightsquigarrow$ Each observation has a chance of $\approx \frac{2}{3}$ of ending up in the bootstrap sample.

Exercise 8.4/4

(a)



(b)



20. Lecture July 11th

Figure 8.8(book): Shows the impact of B (# of bagging draws) on test sample performance.

- Bagging average $\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{f}^{*b}(x)$ can be seen as expected value of $\hat{f}^{*b}(x)$, where the expectation is taken with respect to 'bootstrap randomness'
- Typically find, that $\hat{f}_{bag}(x)$ (and thus its test sample performance) stabilize at some large value for $B \rightsquigarrow$ Estimate of the expected value (see last point) does not improve much, if B is further increased.
- B cannot be chosen too large (if computer time was irrelevant, we would always pick a very large value of B)
- In practice: Choose a value of B that's feasible (regarding computer time) and leads to robust estimates.
  $\rightsquigarrow$ e.g. plot $\hat{f}_{bag}(x)$ against B, and check if the estimate stabilizes.
  $\rightsquigarrow$ If computations are fast enough, may be possible to simply choose a very large value of B, e.g. $B = 1000$

Out-of-Bag error estimation

How to estimate test error for bagging-based forecast?

- Each bootstrap sample contains $\approx 2/3$ of the obs, $\approx 1/3$ not present
- Consider predicting observation $i = 1$
  - Select bagging iterations b for which $i = 1$ is not used for model fitting (about $B/3$)
  - Average their prediction for $i = 1$, compare to true outcome
- Repeat the above for $i = 1, \ldots, n$
  - Valid estimate of test error (based on out-of-sample-predictions!)
  - Very similar (but much simpler than) CV for large B

Variable Importance Measures

- Bagging is harder to interpret than a single tree
- Rough interpretation device: variable importance
- Suppose we want to measure the importance of the variable ’years’ in the Baseball data
  - For all B trees, add the RSS reduction that are due to splits in ’years’
- Similarily, add reductions in Gini measures in case of classification trees

Figure 8.9

Random forests

- Small tweak of bagged trees, aiming to decorrelate the B trees
- Each time a split is considered, select subset of m predictors as split candidates
  - Fresh sample of m predictors taken at each split
  - Popular choice is $m = \sqrt{p}$
- Idea: Make B trees less similar
  - E.g., consider data set with one dominant predictor
  - Bagging: Predictor present in most splits
  - RFs: $(p-m)/p$ of the splits do no even consider the dominant predictor

Tuning parameter m (# of split candidates)

- $m = p$ leads to bagged trees
- $m = \sqrt{p}$ often used in practice
- Small m helpful with many correlated predictors
- Example: Cancer prediction
  - $n = 349$ patients, qualitative outcome (health condition, 15 categories)
  - $p = 500$ predictors
  - Results on the next slide (slides Lecture 23 (slide 12), Figure 8.10)

Boosting

Algorithm 8.2

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set
2. For $b = 1, 2, \dots, B$, repeat:
   a) Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r)

   b) Update $\hat{f}$ by adding in a shrunken version of the new tree:

   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$

   c) Update the residuals,

   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)$$

Boosting: Tuning Parameters

- Number of trees, B
  - Unlike Bagging: Can overfit if B too large
  - Unlike Bagging: Use CV to select B
- Shrinkage parameter, λ
  - Controls size of updates
  - Typical values are $\lambda = 0.01$ or 0.001
  - If λ ist too small, large B is needed for good performance
- Size of each individual tree, d
  - Often $d = 1$ (single tree, i.e. 'stump') works well

Bagging vs. Boosting

Bagging/RF	Boosting
Ensemble method (combine B trees)	Ensemble method (combine B trees)
Individual trees are large	Individual trees are small
Trees independent (Bootstrap)	Trees dependent (grown sequentially)
Tuning parameter m (RF only)	Tuning parameters B, λ, p
No overfitting if B is large	Overfitting if B is large

Cancer Data: Figure 8.11