Introduction to Data Mining and Supervised Learning

Siva Balakrishnan

Data Mining: 36-462/36-662

January 15, 2018

ISL Chapters 1 and 2

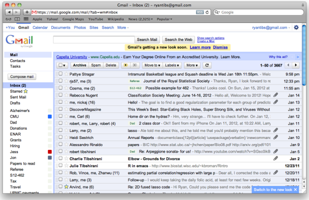
What is data mining?

Data mining (Statistical Learning?) is the science of making predictions using and discovering structure in (large) data sets.

- At intersection of many disciplines – Statistics, Computer Science, Optimization, Information Theory, ...
- Used widely in basic sciences, engineering, economics, public policy, political science, ...

Spam filtering, fraud detection

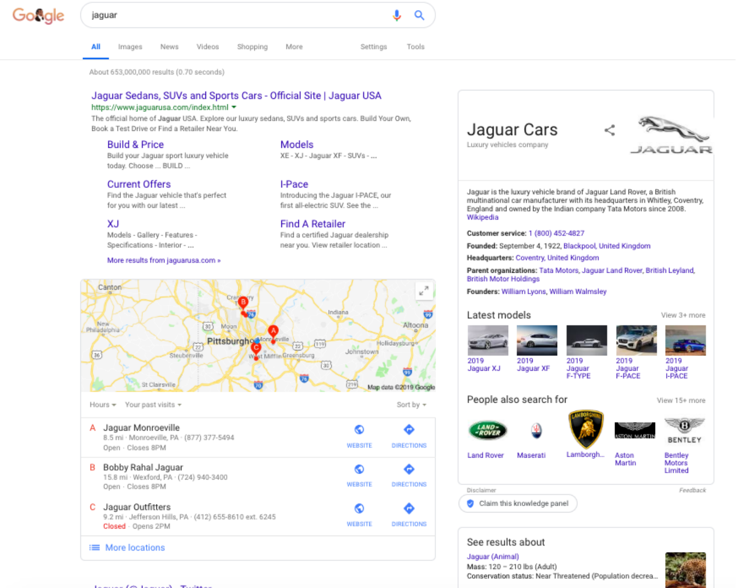
First Generation Successes



- How can we distinguish between spam and real emails?
- How can we identify fradulent transactions?

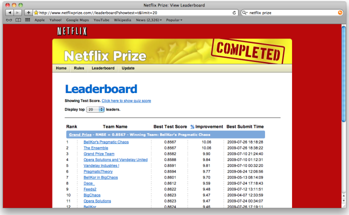
Search

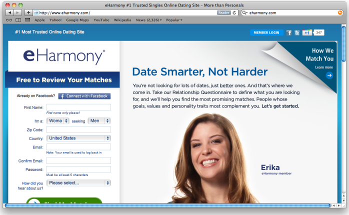
First Generation Successes

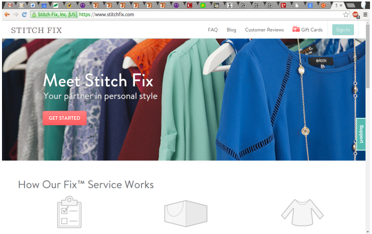


Recommendations

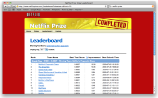
Second Generation Successes

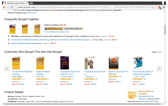


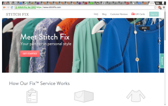




Recommendations



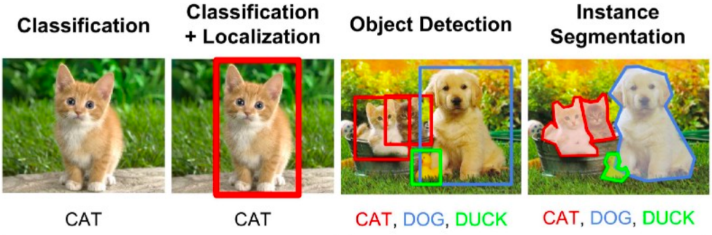




- Which movies should I recommend?
- How should I identify individuals with similar purchasing preferences?
- Which promotional offers should I send out, and to whom?

Computer Vision, Natural Language Processing, Speech

Third Generation Successes



Goals of this course

- Become familiar with common statistical machine learning tools and ideas, from both a theoretical foundation and an applied viewpoint
- Be able to recognize a problem and develop a useful approach to modeling it, and then actually carry it out
- Become comfortable with the fundamentals of statistical machine learning, so that more complicated approaches can be understood and incorporated in the future

Logistics: Prerequisites

- Only formal requirement is 36-401. Strong familiarity with regression, including the linear algebra formulation of it.
- I also assume that you know:
  - Basic probability and statistics
  - Linear algebra
  - R programming

See the syllabus for a detailed list of necessary topics.

Logistics: Lectures, Office hours, Piazza

Lecture slides will be posted shortly before class, so you can write on them if you wish. Scans of what I write will be posted after class (by the next day).

Office hour times are in the Syllabus (on Canvas). Please contact me for outside times.

We will use Piazza for the majority of communication. Please use this instead of email for as many questions about material or homework as possible. Small bonus for answering questions on Piazza.

Logistics: Evaluation

- Homeworks about once a week. Released on Thursdays, due on Wednesdays at Midnight via Gradescope. (30%)
- Two exams. (25% and 30%)
- Final project. (15%)

Logistics: Homeworks

Homeworks about once a week. Released on Thursdays, due on Wednesdays at Midnight via Gradescope. (30%)

Collaboration is encouraged, but write your own homework. (More below).

Your lowest homework score will be dropped. Small bonus for doing well on all HWs.

Because homework is discussed in detail at the start of the next lecture, late homeworks are generally not accepted without prior arrangement.

Logistics: Homework formatting

You are encouraged to type your homeworks, since much of the work will be in R anyway.

You are allowed to scan math, but *you are responsible for making it easily readable for the graders*.

Consider using Rmarkdown in RStudio.

Additionally, minor point deductions are possible for homeworks that are carelessly made particularly challenging to grade (e.g., printing hundreds of pages of zeros...)

Logistics: Homeworks

Homework problems take three main forms:

- Theoretical problems to understand how things work and to become familiar with important concepts and tools.
- Simulations to see how methods perform – and more often, how they break down.
- Data examples to see how to run methods and to build an intuition about what you would see on real data. Also helps to understand problems that arise in practical settings and how to fix them.

Why Theory?

- Real data is messy and always presents new complications
- Understanding why and how things work is a necessary precursor to figuring out what to do.
  - When does a method apply.
  - What might make it fail?
- Provides building blocks to understanding or even designing more complicated approaches.

Collaboration

Talk about the homeworks, help each other learn. However, you need to:

- Write your own code. Making debugging suggestions is ok. Writing one version of the code together is not.
- Do your own writeup of the math. You can talk about and sketch ideas, but you shouldn't be looking at someone else's work when you are writing your own.

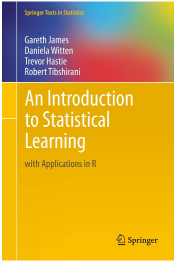
Materials (like solutions) from previous versions of this course are not allowed.

You should be able to explain anything that you submit...

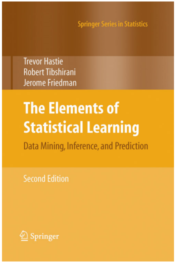
Books

Two useful references for this course are:

- *Introduction to Statistical Learning* by Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. It is easier to read and has more code examples, but covers less material.



- *Elements of Statistical Learning* by Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Harder to read, but much more content.

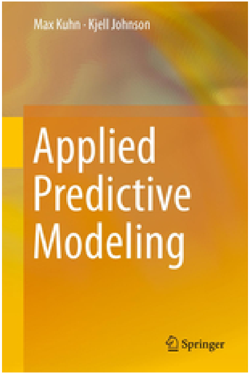


Both textbooks are available (legally) for **free** online from the authors.

Books

The book *Applied Predictive Modeling* by Max Kuhn and Kjell Johnson is also a useful reference for some of the more practical aspects of applying machine learning.

This book is available free online from SpringerLink if you are on the CMU network.

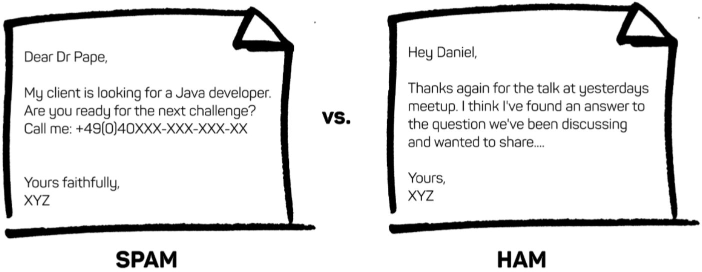


What is *statistical* about statistical learning?

A canonical classification task: given

- text of email
- information about sender and recipient,

determine if email is Spam or Not (Ham).



What is *statistical* about statistical learning?

A rule based classifier:

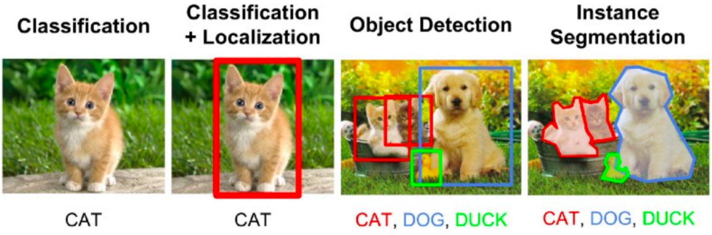
- if text includes my name $\rightarrow$ then (likely) not spam,
- if text includes $\{$`vicodin, prescription, Nigerian Prince, ...` $\}$ then (likely) spam

What is *statistical* about statistical learning?

A rule based classifier:

- if text includes my name $\rightarrow$ then (likely) not spam,
- if text includes {`vicodin`, `prescription`, `Nigerian Prince`, `...` } then (likely) spam

This can get very unwieldy very fast. It is also nearly impossible for even slightly more complex examples.



What is *statistical* about statistical learning?

- Rather than describe a classifier by a collection of rules, maybe we can "learn" these rules from *data* (i.e. learning by example).

What is *statistical* about statistical learning?

- Rather than describe a classifier by a collection of rules, maybe we can "learn" these rules from *data* (i.e. learning by example).
- Collect a large data set of images/emails together with their "labels" (Cat/Dog/Duck, Spam/Ham). Hope to "learn" a mapping from inputs to outputs using this data.

What is *statistical* about statistical learning?

- Rather than describe a classifier by a collection of rules, maybe we can "learn" these rules from *data* (i.e. learning by example).
- Collect a large data set of images/emails together with their "labels" (Cat/Dog/Duck, Spam/Ham). Hope to "learn" a mapping from inputs to outputs using this data.
- Historically, "understanding data" or making predictions using data, was a core statistical task.

What is *statistical* about statistical learning?

- Given a collection of emails and their labels, there is a *perfect* but *useless* classifier!

What is *statistical* about statistical learning?

- Given a collection of emails and their labels, there is a *perfect* but *useless* classifier!
- We do not want a perfect classifier on the emails we have, rather one that is good on *future, unseen* examples.

What is *statistical* about statistical learning?

- Given a collection of emails and their labels, there is a *perfect* but *useless* classifier!
- We do not want a perfect classifier on the emails we have, rather one that is good on *future, unseen* examples.
- This is called *generalization*. It is a core idea that will underlie everything we talk about in this course (whenever you hear terms like bias, variance, overfitting, underfitting connect them to *generalization*).

What is *statistical* about statistical learning?

- Given a collection of emails and their labels, there is a *perfect* but *useless* classifier!
- We do not want a perfect classifier on the emails we have, rather one that is good on *future, unseen* examples.
- This is called *generalization*. It is a core idea that will underlie everything we talk about in this course (whenever you hear terms like bias, variance, overfitting, underfitting connect them to *generalization*).
- Need some *relation* between the training and test data.

What is *statistical* about statistical learning?

- Given a collection of emails and their labels, there is a *perfect* but *useless* classifier!
- We do not want a perfect classifier on the emails we have, rather one that is good on *future, unseen* examples.
- This is called *generalization*. It is a core idea that will underlie everything we talk about in this course (whenever you hear terms like bias, variance, overfitting, underfitting connect them to *generalization*).
- Need some *relation* between the training and test data.
- *Assume that they come from the same distribution.* Another key statistical concept!

Classification of statistical learning problems

Statistical learning problems are often divided as follows

- Supervised learning: Making predictions
  i.e., given measurements $(X_1, Y_1), \dots (X_n, Y_n)$, learn a model to predict Y_i from X_i
  - Regression: Y_i is a continuous value
  - Classification: Y_i is a (unordered) discrete value

- Unsupervised learning: discovering structure
  E.g., given measurements $X_1, \dots X_n$, learn some underlying structure based on similarity

Supervised Learning Notation

We observe a training set of n data points $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ where each y_i is a scalar, and each x_i is a p-dimensional vector, i.e.:

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}.$$

- The x vector goes by many names $\{$`input, independent variables, feature vector`$\}$.
- The y vector is the $\{$`output, dependent variable, prediction`$\}$

Supervised Learning Notation

We can put all the features together into a *design matrix*.

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

Note that the rows $x_i^T \in \mathbb{R}^p$, $x_i^T = (x_{i1}, \ldots, x_{ip})$, are the individual observations, and the columns $\mathbf{x}_j \in \mathbb{R}^n$, $\mathbf{x}_j = (x_{1j}, \ldots, x_{nj})^T$, are all the observations of a particular variable.

Supervised learning: Two Basic Ways of Making Predictions

1. Linear Regression: At any point $x = (x_1, \dots, x_p)$ we predict:

$$\widehat{y} = \widehat{\beta}_0 + \sum_{j=1}^{p} \widehat{\beta}_j x_j.$$

   To construct our predictor we need to "learn" / "estimate" the coefficients using the training data.

Supervised learning: Two Basic Ways of Making Predictions

1. Linear Regression: At any point $x = (x_1, \ldots, x_p)$ we predict:

   $$\widehat{y} = \widehat{\beta}_0 + \sum_{j=1}^{p} \widehat{\beta}_j x_j.$$

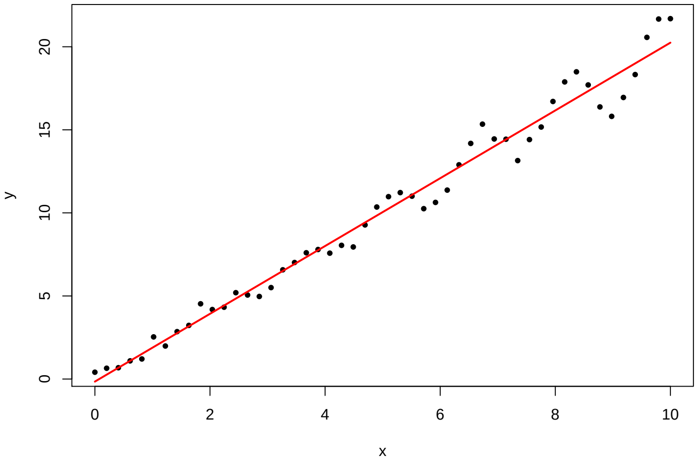
   To construct our predictor we need to "learn"/"estimate" the coefficients using the training data.

2. k-nearest Neighbors Regression: At any point $x = (x_1, \ldots, x_p)$ we predict:

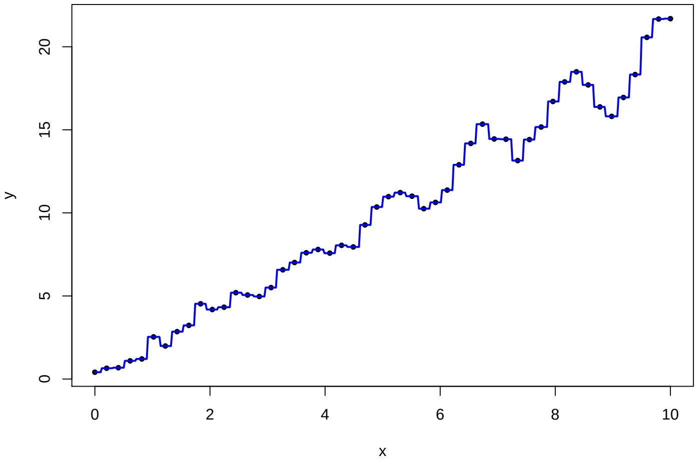
   $$\widehat{y} = \frac{1}{k} \sum_{x_j \in N_k(x)} y_j,$$

   where $N_k(x)$ is the set of the k closest point to x in the training set. Nothing to learn/estimate?

Linear Regression



1-nearest neighbors regression



How might we compare/evaluate our two prediction methods?

- First we define a *loss function*, i.e. decide how much we should penalize an incorrect prediction. We will denote our loss for predicting y by $\widehat{y}$ as $\mathcal{L}(y, \widehat{y})$.

How might we compare/evaluate our two prediction methods?

- First we define a *loss function*, i.e. decide how much we should penalize an incorrect prediction. We will denote our loss for predicting y by $\widehat{y}$ as $\mathcal{L}(y, \widehat{y})$.
- Two canonical loss functions:
  - Squared loss:

$$\mathcal{L}(y, \widehat{y}) = (y - \widehat{y})^2.$$

  - 0/1 loss:

$$\mathcal{L}(y, \widehat{y}) = \mathbb{I}(y \neq \widehat{y}).$$

How might we compare/evaluate our two prediction methods?

- First we define a *loss function*, i.e. decide how much we should penalize an incorrect prediction. We will denote our loss for predicting y by $\widehat{y}$ as $\mathcal{L}(y, \widehat{y})$.
- Two canonical loss functions:
  - Squared loss:

$$\mathcal{L}(y, \widehat{y}) = (y - \widehat{y})^2.$$

  - 0/1 loss:

$$\mathcal{L}(y, \widehat{y}) = \mathbb{I}(y \neq \widehat{y}).$$

- In practice, this is one of the most important choices we need to make. Often requires careful thought, and influences results and the choice of method in important ways.

How might we compare/evaluate our two prediction methods?

What do we care about when making predictions?

How might we compare/evaluate our two prediction methods?

What do we care about when making predictions?

- The average or expected loss, i.e. this is sometimes called the risk of your procedure:

$$R(\widehat{y}(x)) = \mathbb{E}_{(x,y)\sim\mathcal{P}}(y - \widehat{y}(x))^2 = \mathbb{E}_{(x,y)\sim\mathcal{P}}\mathcal{L}(y, \widehat{y}(x)).$$

  i.e. we care about the **average** loss on a new test example.

A good procedure is one that has low risk. How do we calculate the risk?

The complete supervised learning setup

- We imagine our data set is drawn from a distribution, i.e. we observe $\{(x_1, y_1), \ldots, (x_n, y_n)\} \sim \mathcal{P}$.
- We construct our predictor $\widehat{y}(x)$ using the training data.
- We imagine evaluating its performance by measuring its risk, i.e. imagine drawing a hypothetical new example from the same distribution $(x, y) \sim \mathcal{P}$ and evaluating its risk.

Two things we need to specify better:

- How do we construct predictors?
- How do we really calculate the risk of a predictor?

Estimating the risk of a predictor

The second question is somewhat simpler to answer.

- We just keep aside a sufficiently large number of unseen, test examples, and compute the loss on those, i.e.:

$$\widehat{R}(\widehat{y}(x)) = \frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{L}(y_i, \widehat{y}_i(x_i)),$$

  where we now denote our test set by $\{(x_1, y_1), \ldots, (x_{n_t}, y_{n_t})\}$.

Why is this a good idea? What property does this estimate satisfy?

How do we construct good predictors?

This is a much more elaborate question to answer.

- What is a good first idea?

How do we construct good predictors?

This is a much more elaborate question to answer.

- What is a good first idea?

- When is this not ideal?