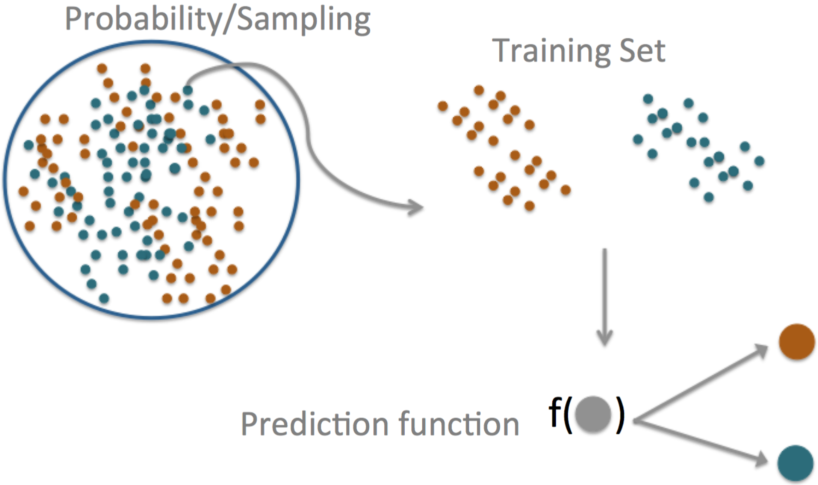
What is prediction?

Jeffrey Leek

May 18, 2016

The central dogma of prediction



What can go wrong

BIG DATA

The Parable of Google Flu: Traps in Big Data Analysis

Large errors in flu prediction were largely avoidable, which offers lessons for the use of big data.

David Lazer,[1,2*] Ryan Kennedy,[1,3,4] Gary King,[3] Alessandro Vespignani[5,6,3]

In February 2013, Google Flu Trends (GFT) made headlines but not for a reason that Google executives or the creators of the flu tracking system would have hoped. *Nature* reported that GFT was predicting more than double the proportion of doctor visits for influenza-like illness (ILI) than the Centers for Disease Control and Prevention (CDC), which bases its estimates on surveillance reports from laboratories across the United States (*1*, *2*). This happened despite the fact that GFT was built to predict CDC reports. Given that GFT is often held up as an exemplary use of big data (*3*, *4*), what lessons can we draw from this error?

The problems we identify are not limited to GFT. Research on whether search or social media can

run ever since, with a few changes announced in October 2013 (*10*, *15*).

Although not widely reported until 2013, the new GFT has been persistently overestimating flu prevalence for a much longer time. GFT also missed by a very large margin in the 2011–2012 flu season and has missed high for 100 out of 108 weeks starting with August 2011 (see the graph). These errors are not randomly distributed. For example, last week's errors predict this week's errors (temporal autocorrelation), and the direction and magnitude of error varies with the time of year (seasonality). These patterns mean that GFT overlooks considerable information that could be extracted by traditional statistical methods.

http://www.sciencemag.org/content/343/6176/1203.full.pdf

Components of a predictor

question -> input data -> features -> algorithm -> parameters -> evaluation

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

Start with a general question

Can I automatically detect emails that are SPAM that are not?

Make it concrete

Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

R: Spam E-mail Database

rss.acs.unt.edu/Rdoc/library/kernlab/html/spam.html

spam {kernlab} R Documentation

Spam E-mail Database

Description

A data set collected at Hewlett-Packard Labs, that classifies 4601 e-mails as spam or non-spam. In addition to this class label there are 57 variables indicating the frequency of certain words and characters in the e-mail.

Usage

```
data(spam)
```

Format

A data frame with 4601 observations and 58 variables.

The first 48 variables contain the frequency of the variable name (e.g., business) in the e-mail. If the variable name starts with num (e.g., num650) the it indicates the frequency of the corresponding number (e.g., 650). The variables 49-54 indicate the frequency of the characters `;', `(', `[', `!', `$', and `#'. The variables 55-57 contain the average, longest and total run-length of captial letters. Variable 58 indicates the type of the mail and is either `"nonspam"` or `"spam"`, i.e. unsolicited commercial e-mail.

Details

The data set contains 2788 e-mails classified as `"nonspam"` and 1813 classified as `"spam"`.

The ``spam'' concept is diverse: advertisements for products/web sites, make money fast schemes, chain letters, pornography... This collection of spam e-mails came from the collectors' postmaster and individuals who had filed spam. The collection of non-spam e-mails came from filed work and personal e-mails, and hence the word 'george' and the area code '650' are indicators of non-spam. These are useful when constructing a personalized spam filter. One would either have to blind such non-spam indicators or get a very wide collection of non-spam to generate a general purpose spam filter.

Source

- Creators: Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt at Hewlett-Packard Labs, 1501 Page Mill Rd., Palo Alto, CA 94304
- Donor: George Forman (gforman at nospam hpl.hp.com) 650-857-7835

These data have been taken from the UCI Repository Of Machine Learning Databases at http://www.ics.uci.edu/~mlearn/MLRepository.html

References

T. Hastie, R. Tibshirani, J.H. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

Dear Jeff,

Can you send me your address so I can send you the invitation?

Thanks,

Ben

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

Dear Jeff,

Can you send me your address so I can send you the invitation?

Thanks,

Ben

Frequency of you $= 2/17 = 0.118$

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

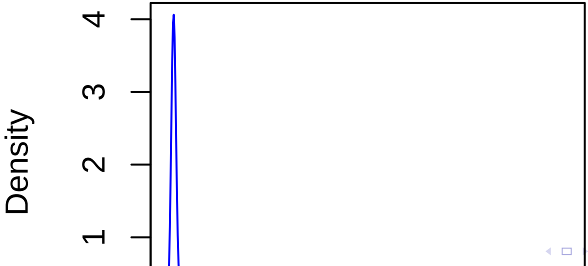
```r
library(kernlab)
data(spam)
head(spam)
```

```
##   make address  all num3d  our over remove internet orde
## 1 0.00    0.64 0.64     0 0.32 0.00   0.00     0.00  0.0
## 2 0.21    0.28 0.50     0 0.14 0.28   0.21     0.07  0.0
## 3 0.06    0.00 0.71     0 1.23 0.19   0.19     0.12  0.6
## 4 0.00    0.00 0.00     0 0.63 0.00   0.31     0.63  0.3
## 5 0.00    0.00 0.00     0 0.63 0.00   0.31     0.63  0.3
## 6 0.00    0.00 0.00     0 1.85 0.00   0.00     1.85  0.0
##   will people report addresses free business email  you
## 1 0.64   0.00   0.00      0.00 0.32     0.00  1.29 1.93
## 2 0.79   0.65   0.21      0.14 0.14     0.07  0.28 3.47
## 3 0.45   0.12   0.00      1.75 0.06     0.06  1.03 1.36
```

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

```
plot(density(spam$your[spam$type=="nonspam"]),
     col="blue",main="",xlab="Frequency of 'your'")
lines(density(spam$your[spam$type=="spam"]),col="red")
```



SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

Our algorithm

- Find a value C.
- **frequency of 'your'** $>$ **C** predict "spam"

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

```
plot(density(spam$your[spam$type=="nonspam"]),
     col="blue",main="",xlab="Frequency of 'your'")
lines(density(spam$your[spam$type=="spam"]),col="red")
abline(v=0.5,col="black")
```

SPAM Example

question -> input data -> features -> algorithm -> parameters -> evaluation

```r
prediction <- ifelse(spam$your > 0.5,"spam","nonspam")
table(prediction,spam$type)/length(spam$type)
```

```
##
## prediction   nonspam      spam
##    nonspam 0.4590306 0.1017170
##    spam    0.1469246 0.2923278
```

Accuracy $\approx 0.459 + 0.292 = 0.751$