ECON 108: Data Science for Business and Economic Decisions

Instructor: Han Hong
Preceptor: Carol Lu

Department of Economics
Stanford University

March, 2022

Welcome!

- This is a new course offering by the econ department.
- We are living in a big data era. The industry demand for data scientists and machine learning scientists are increasing rapidly.
- Many econ Phd students are also choosing to join big tech companies instead of taking up academic teaching positions.
- A lot of actions in the industry for interactions between economists, statisticians, data scientists, machine learning scientists, and software development engineers.
- Big data poses a lot of opportunities and also challenges to statistical and econometrics analysis and the economic professional more generally.
- This course will explore the relation among data science, statistics, econometrics, and machine learning.

- While different languages and terminologies are used in these different areas, IMHO I think they are closely related! I also hope to convince you that this is the case!
- Some examples:
  - Correlation coefficients in statistics are called cosine-similarity in data science and machine learning!
  - Regressions in statistics and econometrics are called supervised-learning in data science!
  - Heterogeneity in econometrics are related to unsupervised-learning in machine learning.
  - Size and power in hypothesis testing (type I and type II errors) are called sensitivity and specificity in machine learning classification analysis.
  - log Likelihood functions in statistics are called (minus) cross-entropy loss functions in machine learning.
  - Mean square loss functions in econometrics are called Gini-Index loss functions in machine learning.
  - Multinomial logit function in econometrics is called soft-max activation in neural networks.

Logistics

- My office is Landau Economics Building 228. Office hours are 3:30pm to 4:30pm on Thursdays.
- I expect to give 6-7 problem sets and a final exam. I suggest splitting 60%/40% in the final grade but we can discuss.
- I expect you to have a laptop.
- We will be using the R software. Please go ahead to install R or R-studio on your laptop.
- It is also possible to use university servers for running R. We can work through the steps together if you need this.
- The prerequisite background is statistics and econometrics at the levels of eco 102A and 102B.
- For example, I expect that you have seen confidence intervals, p-values, and multiple least square regressions.
- It is ok if you have seen them but don't remember all the details. We will review them in class.

Textbook

- We will use this book by Matt Taddy entitled:
- "Business Data Science"
- It has a long subtitle!
- "Combining Machine Learning and Economics to Optimize, Automate, and Accelerate Business Decisions"
- You can either buy a kindle electronic copy of this book or a paper copy at Amazon:

  `https://www.amazon.com/Business-Data-Science-Combining-Accelerate/dp/1260452778`

- Why Matt Taddy and why this book?
- Matt Taddy is a VP at Amazon and has been in charge of the developments and applications of data science and machine learning.

- Prior to joining Amazon, Matt Taddy is a professor in the University of Chicago Booth School of Business.
- We will follow this book in the course of this class.
- Matt Taddy has a github repository that contains information about this book and a lot of other useful information:

  `https://github.com/taddylab`

- Matt Taddy's github for this book:

  `https://github.com/TaddyLab/BDS`

- Our lectures will be based on revisions of the lecture notes on Matt Taddy's github for MBA teaching at U of Chicago:

  `https://github.com/TaddyLab/MBAcourse/tree/master/lectures`

- Disclaimer: I might not agree with every statement in this book (or any book). But we can always discuss when there are possible disagreements.

- Who am I and why I am teaching this class?
- I am a professor of econometrics in the econ department. You can find my profile at

  `https://profiles.stanford.edu/han-hong?tab=bio`

- A short summary: I did an economics PhD, two master degrees in computer science and statistics at Stanford between 1993 and 1998.
- When I graduated in 1998, I had to choose between working for the IMF (International Monetary Fund) or a startup internet company, or going to academia.
- My advisors "convinced" me to stay in academia.
- Between 1998 and 2007, I taught at Princeton and Duke.
- In 2007, I returned to teach at Stanford and have been teaching econometrics since.
- In the past few years I have done some work for Amazon, where I got to meet Matt Taddy!
- In the past I have been mostly teaching graduate students. This is also a new experience to me. I look forward to interacting with you all, and very much appreciate your feedback in pacing the class!

- We will follow the topics in Matt Taddy's textbook as closely as possible.
  - Data and Software
  - Uncertainty (related to eco 102A)
  - Regression (related to eco 102B)
  - Regularization (Lasso and information criteria)
  - Classification (When you want to predict binary or discrete outcomes)
  - Experiments
  - Controls
  - Factorization
  - Text as Data
  - Nonparametrics (Trees and Random Forests)
  - Artificial Intelligence (basically Neural Networks)
- The last four chapters are interesting but also more advanced. We might not have time cover some or all of them.

What is in a name?

Big Data, Econometrics, Statistics, and Machine Learning

There are many labels for what we do... In chronological order:

Statistics $\rightarrow$ Econometrics $\rightarrow$
Data Mining / Big Data / Data Science
$\rightarrow$ Machine Learning (ML) and Artificial Intelligence (AI)

Along this spectrum, you move from heavy focus on what things you are measuring (what real phenomena they correspond to) to a more pragmatic 'useful is true' pattern discovery approach.

- On the one hand, econometricians/statisticians/economists specialize in inventing theories and they look for data to validate theoretical predictions.
- On the other hand, data scientists/machine learning scientists mine data and discover patterns, and they seek a theoretical framework to guide data analysis.

The 'BD' name comes from computer scientists working to do aggregation on data that is too big to fit on a single machine.

As aggregation became analysis, BD got closer to stats + ML. And after a healthy dose of hype, it may be time to clarify some confusions ...

Matt Taddy's take : *Data Science* is the umbrella term for inference in a world that is messier that in our typical statistic textbook, and *Big Data* is DS focused on business and industrial applications.

- Infer patterns from complex high dimensional data.
- Simplicity and scaleability of algorithms is essential.
- We keep an eye on both *being useful* and *being correct*.
- The end product is a *decision*.

I agree overall with Matt Taddy's assessment, but the devils are in the details.

A Nobel laureate in economics was once heard saying: "All (economic) models are wrong. Some might be useful".

On the one hand, in academia, a main challenge for empirical researchers is to find data.

On the other hand, in high tech firms, a major headache is how to get rid of data! It is not even possible to store all the web streaming data regardless of the size of the server capacities.

Sharing data is difficult because of institutional and legal constraints.

Information transparency is an important issue for enhancing society welfare.

The data science and analysis skills that we will learn can contribute to a larger scale debate of data and information transparency.

Big data poses major challenges to both

- engineering, including hardware and software engineering; and
- the foundation of statistical science.

We will focus on the statistical aspect of the challenges.

The engineering challenges of data storage, extraction and transformation are tackled by computer scientists and engineers.

It helps to know some of the basics of database system, if you need to converse with data scientists in your future work.

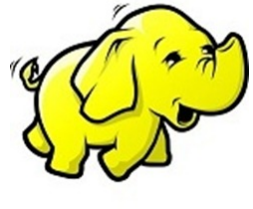
A good reference is cs145:

`https://cs145-fa20.github.io/`

Industrial and commercial grade database systems have evolved rapidly in the past decade.

- Row-based database facilitates consistent and durable data entry.
- Column-based database facilitates statistical analysis.
- There are many competing open-source and commercial database systems.

Distributed File Systems (DFS)



For example, when the data is unstructured (log files, raw text documents) or too big to fit on a single server, people turn to DFS models.

Examples are Hadoop HDFS, Amazon S3.

The data is scattered across many machines,
with a key that tells you where all the pieces live.

Analysis uses algorithm frameworks like MapReduce:

partition data into small chunks, analyze each independently.

Big Data can be really big, for example in TBs or PBs.

- Data scientists makes use of python, Scala, and various industrial strength software and languages to storage, extract, transform, and visualize data.
- Often times, data needs to be stored in a format that can be understood by different languages/platforms. Each software/platform has APIs to convert the data between platform specific and common formats.

A common platform-independent data format is flat files (.txt, .csv, etc)..

SQL is the dominant tool for data storage, extraction and transform.

Structured Query Language (SQL) for relational databases

A typical relational database query on structured data:

```
select apple_id,apple_price from grocerylist where apple_col = green
```

Examples are MySQL, Oracle, SQLite, Teradata, ...

The analyst typically pulls data from the DB with an SQL query, writes this to a flat file, then reads the flat file into R.

Packages like RSQLite can also read directly from SQL DBs.

After extraction, often you analyze the data *in memory*.

When the data is VERY large, you might need to work directly on the cloud with AWS, Spark, etc. But R is a good starting point.

All of our analysis will be conducted in R

R is an industrial strength software for data analysis. It's free, cross platform, and hugely capable.

Academics (stats, marketing/finance, genetics, engineering), companies (EBay, Google, Microsoft, Boeing, IBM), and governments (Rand, DOE National Labs, Navy) use R.

Since R is free, you'll always be able to use it. It also has a large ecosystem and community of contributors who continuously develop new packages.

If you continue to work in the data industry, eventually you might need to move on to python, scala, etc in the production pipeline. But for gaining statistical insights, R has a great deal of advantages.

A huge strength of R is that it is open-source.
This is also why it is sometimes a bit unpolished.

R has a core, to which you can add contributed packages. These add-ons are as varied as the people who write them. Some are specific, others general, some are great, some ... .

R is not without flaws, but neither are the other options.
e.g., Like Matt Taddy I also like python, but the community of stats developers is smaller.

However, Python has a larger eco-system outside of statistics and is more of a programming language. In graduate school, I use perl (which predates python) to automate my RA work for my professor. I also use python for neighborhood surveillance at home .... :-)

Some students prefer to wrap R in an IDE; e.g. R-studio.

The barrier of entry for R is its command line interface:

You type commands to get what you want.

The upfront learning curve can be steep if you are used to Excel only, but is very worthwhile.

- You have code of saved commands, so you know what you've done and can easily repeat similar operations.
- Interacting with computers in this way is a Big Data skill.
- Matt Taddy's github has datasets and codes for real data analysis and homeworks.
- The best way to learn software is through imitation.
- There are also a lot of online resources.
- For example, I have been looking at this online tutorial in the past few weeks and think it is very good:

`https://www.guru99.com/r-tutorial.html`

Computing in R

To start, click on  or  or just type 'R' in a terminal.

At its most basic, R's just a fancy calculator. (e.g., `*`,`/`,`+`,`-`).

Everything is based on assigning names
(<- works pretty much the same as =).

`A <- 2` → 2.

`B <- c(1,2,3)` → 1 2 3 (same as `B=1:3`).

`C = A + B[3]` → 5.

The `c( )` function and `:` operator build vectors.
`length(B)` will tell you how many elements it holds (3).

To inspect a variable, just type the name (do this often!).

R can read different types of data format

First, set the working directory to where you store data.

It's good practice to create folders just for this.

Use `cmd-D` (Mac), `File` → `ChangeDir` (Windows)

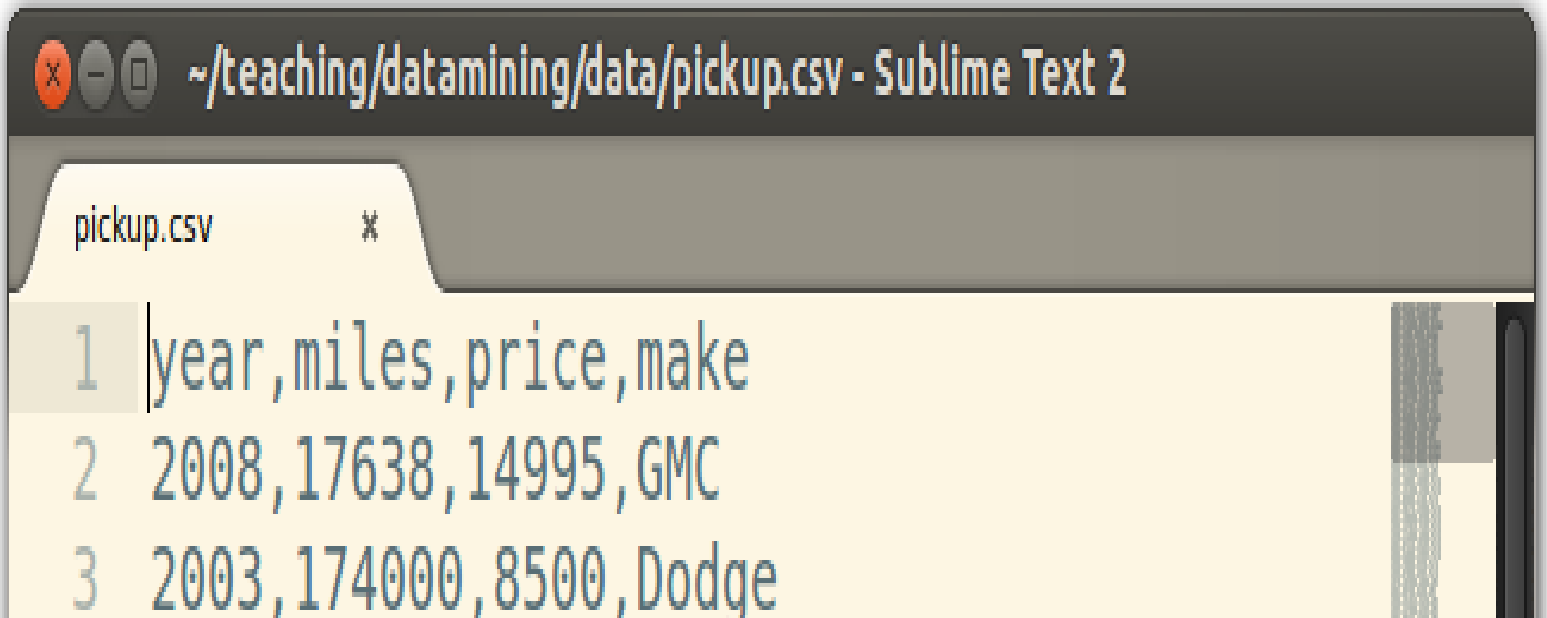
or just type `setwd('/path/to/my/working/dir')`.

Set a default with *preferences* or shortcut *properties*.

In this directory, you'll store flat files: text tables of data.

We'll use mostly `.csv` files: comma separated values.

Any file in Excel can be saved as '.csv', and vice versa.



To load data, type `trucks <- read.csv("pickup.csv")`.
The data are then in working memory (RAM).

`trucks` is a `dataframe`: a matrix with names.
You can use index names and numbers to access the data.

```
trucks[1,] # the first observation
trucks[1:10,] # the first 10 observations
trucks[,1] # the first variable (year)
trucks$year # same thing
trucks[,'year'] # same thing again
trucks[trucks$miles>200000,] # some real clunkers
```

And call R's functions on the data.

```
nrow(trucks) # sample size
summary(trucks) # summary of each variable
```

Basic Elements in R: numeric, factor, logical, character

The values in our dataframes all have a class.

- numeric values are just numbers (1, 2, 0.56, 10.2).
- factor variables are categories with levels ('lowfat','reg')
- logical values are either TRUE or FALSE.
- character strings are just words or messages ('hi mom').

We have plenty of tools to investigate and manipulate these:

`as.numeric(X)`, `factor(X)`, `class(X)`, `levels(X)`

For example: `as.numeric(trucks$year)`, `factor(trucks$make)`, `class(trucks$make)`, `levels(trucks$make)`

R has functions that look like `f(arg1, arg2, ...)`.

e.g., create new variables: `lprice = log(trucks$price)`.

And add them to your data: `trucks$lprice = lprice`.

To find out how a function works, type `?f` or `help(f)`.

Plotting is very intuitive

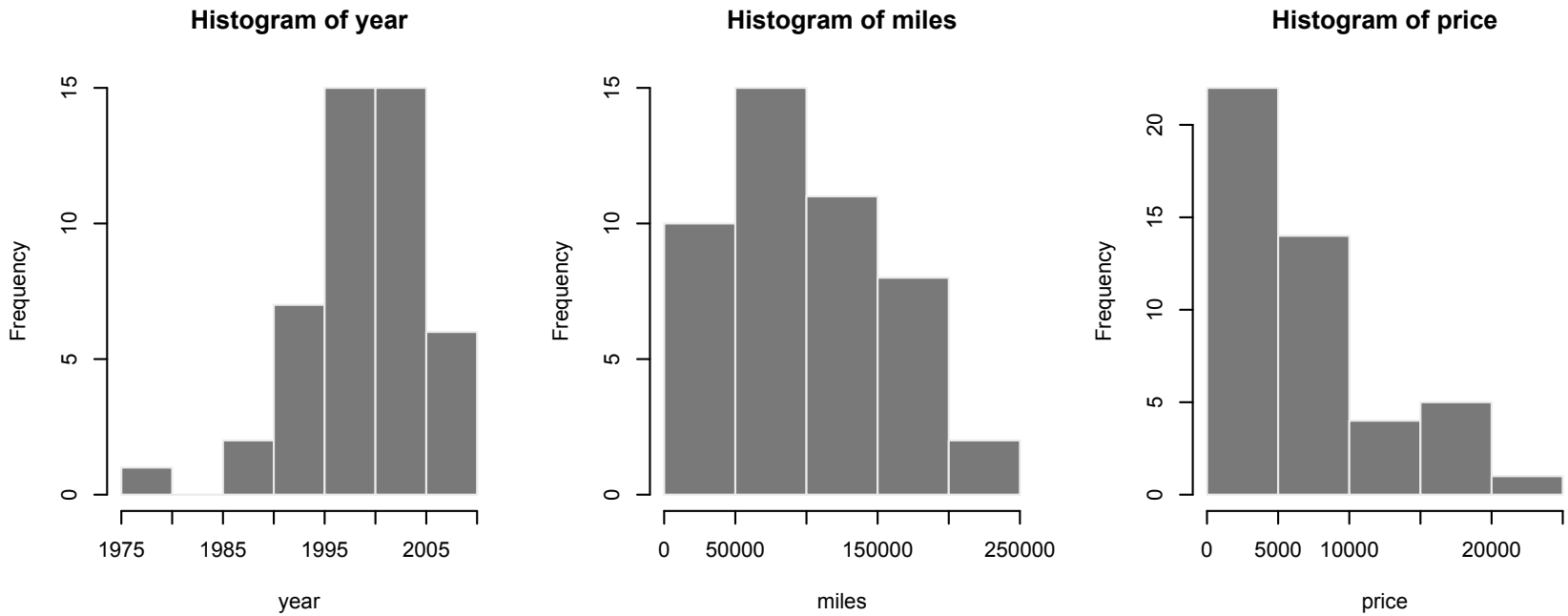
Use `plot(mydata$X, lY)` or `plot(lY ~ mydata$X)`

Let's look at some basic plots...

```
plot(lY ~ mydata$X)
```

See pickup.R and pickup.csv under the examples directory on the github.

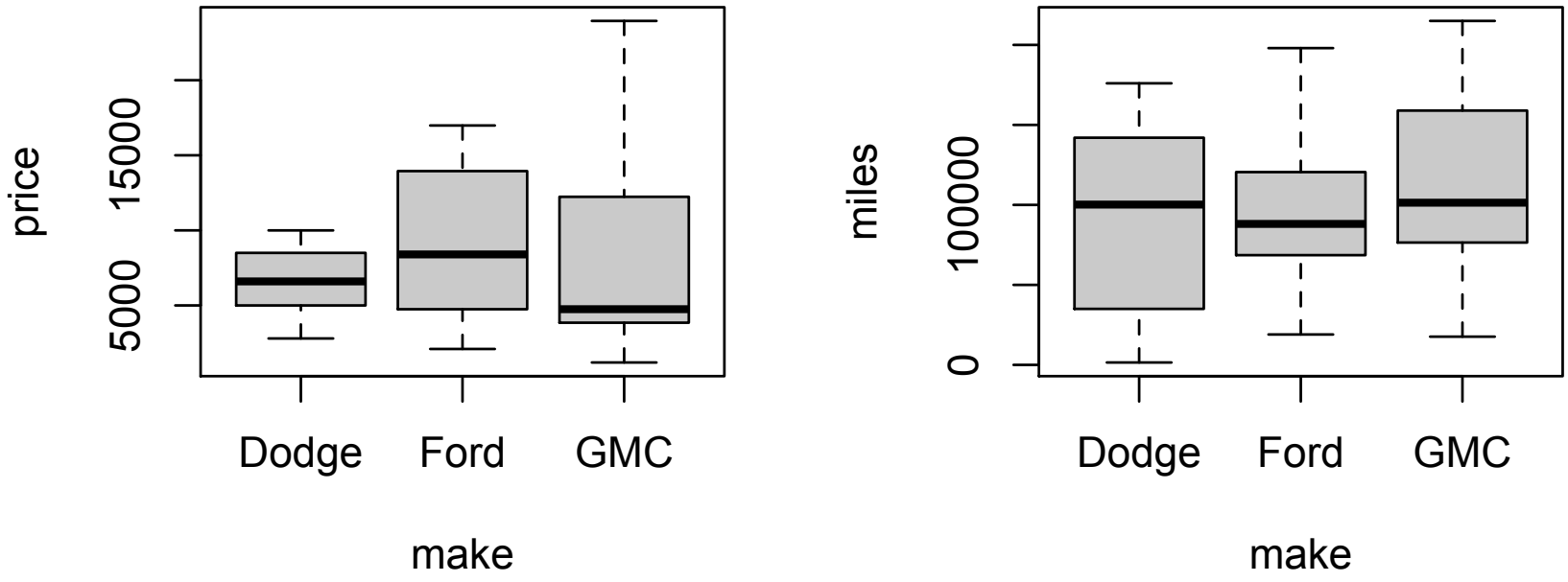
The simple histogram for continuous variables



Data is binned and plotted bar height is the count in each bin.

```
par(mfrow=c(2,2))
hist(trucks$year, main="Histogram of Year", xlab="year",col="gray")
hist(trucks$miles, main="Histogram of Miles", xlab="miles",col="gray")

hist(trucks$price, main="Histogram of Price", xlab="price",col="gray")
```

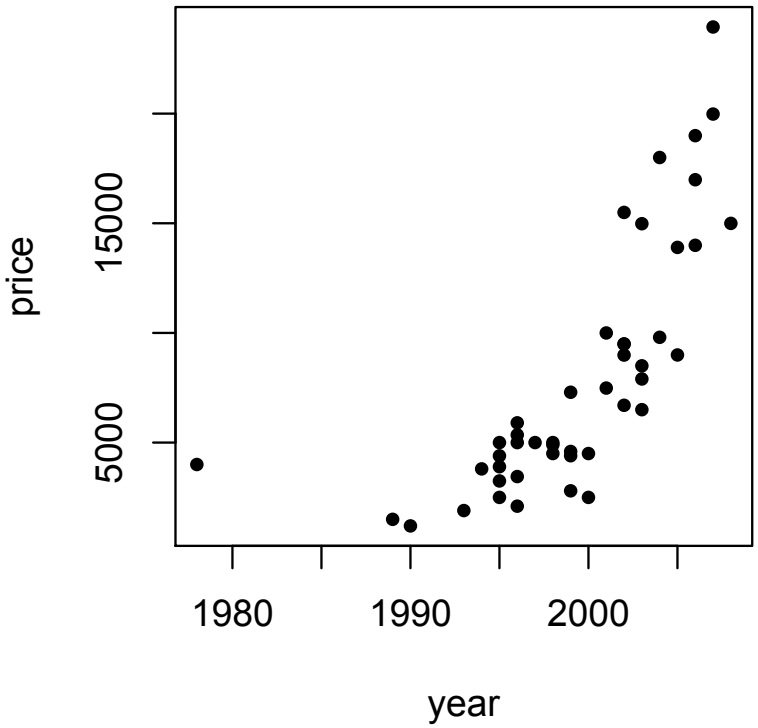
Boxplots: summarizing conditional distributions

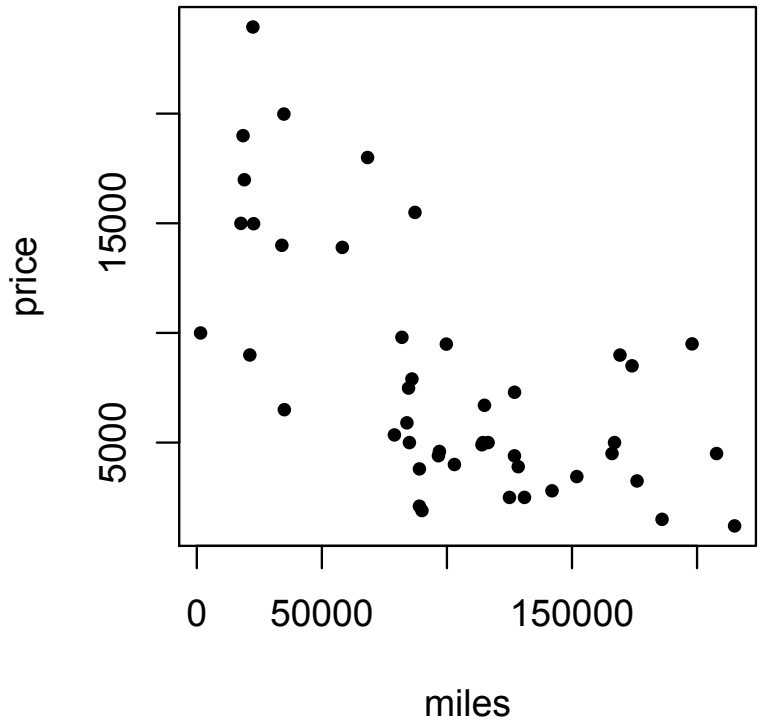


The box is the Interquartile Range (IQR; i.e., 25^{th} to 75^{th} %), with the median in bold. The whiskers extend to the most extreme point which is no more than 1.5 times the IQR width from the box.

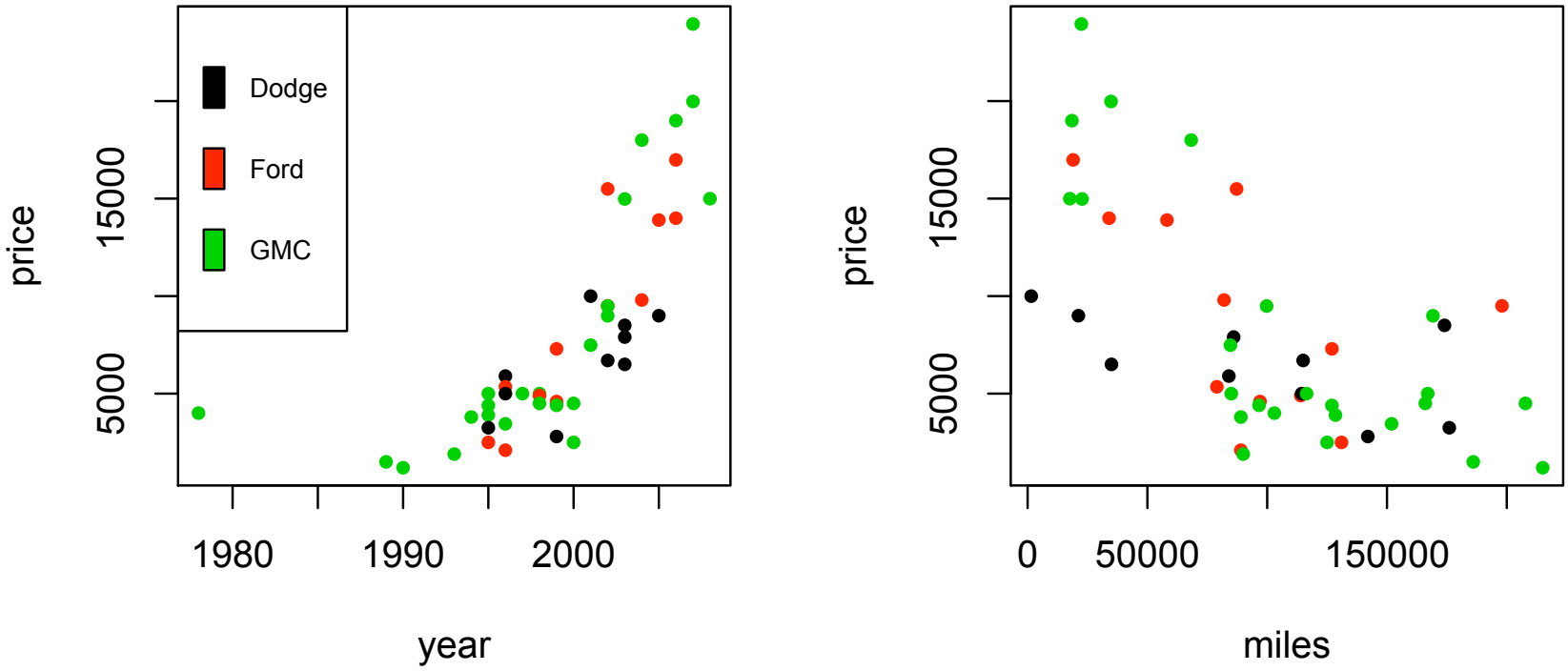
```
par(mfrow=c(2,2))
plot(price ~ make, data=trucks)
plot(miles ~ make, data=trucks)
```

Use scatterplots to compare variables.





And color them to see another dimension.



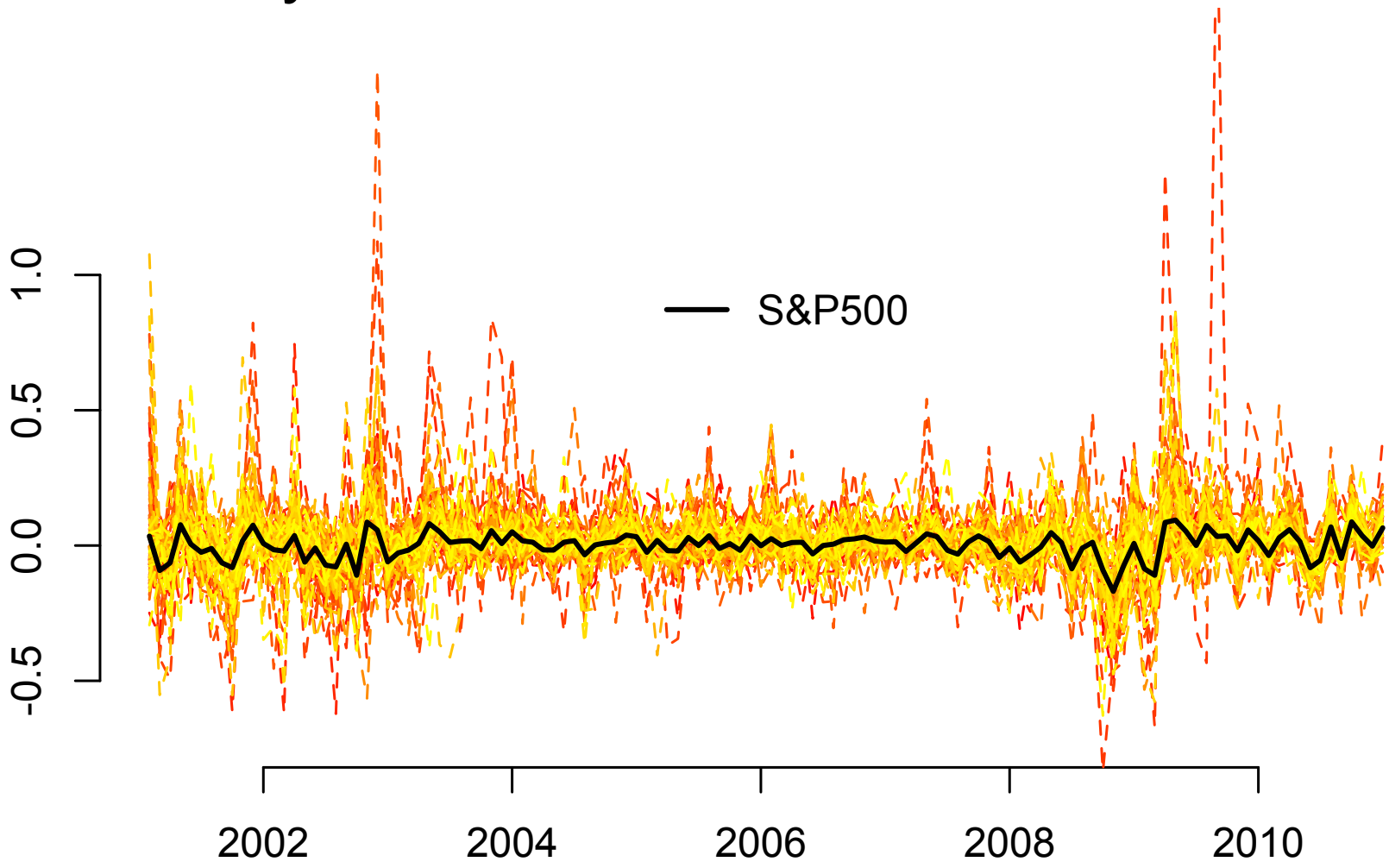

```
par(mfrow=c(2,2))
plot(price ~ year, data=trucks)
plot(price ~ miles, data=trucks)
plot(price ~ year, data=trucks, col=make)
legend("topleft", fill=1:3, legend=levels(trucks$make))
plot(price ~ miles, data=trucks, col=make)

legend("topright", fill=1:3, legend=levels(trucks$make))
```

Scatterplots are a fundamental unit of statistics.

- A good starting point is to find and compare meaningful low-dimensional summary statistics.
- Humans are good at comparing a few variables.
- If we can put it in a picture, we can build intuition.
- Prediction is easy in low dimensions.

- The key to good graphics is to reduce high-dimensional data to a few very informative summary statistics, then plot them.
- If you want to explore more data visualization, you can check out additional R-packages such as ggplot2 and tidyverse, etc.
- Also feel free to check out one of the O'Reilly data science books.

Plot: monthly stock returns



Generate this figure by running the stocks.R code under the example directory.

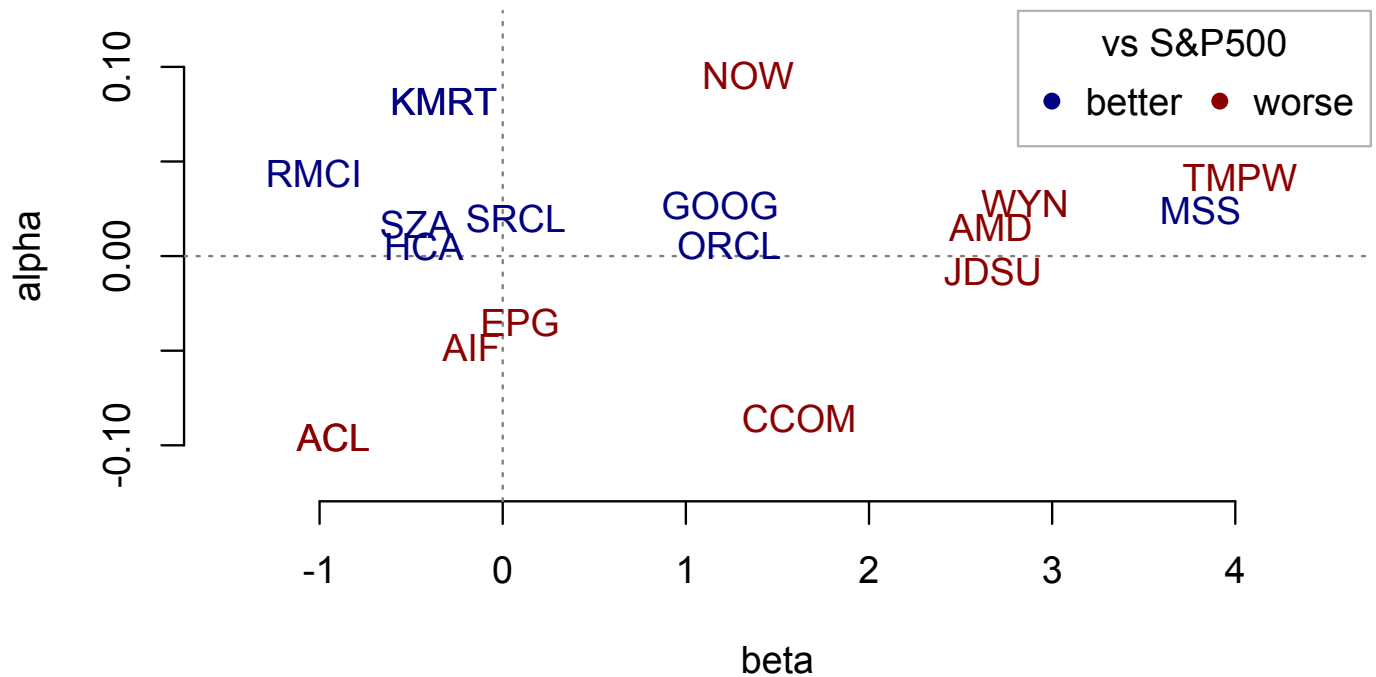
In R: `source("stocks.R")`

Useful plot: market model coefficients

Fit a line between stock returns R_t and market returns M_t (SP).

$$R_t \approx \alpha + \beta M_t$$

α is money you make regardless of what the market does.
β is the asset's sensitivity to broad market movements.



Regression is king

```
fit <- glm(log(price)~ year+make, data=trucks)
summary(fit)  # glm = generalized linear model
...
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -195.22731   25.18120  -7.753 1.24e-09 ***
year           0.10196    0.01259   8.099 4.07e-10 ***
makeFord       0.13987    0.19786   0.707    0.484
makeGMC        0.16202    0.17586   0.921    0.362
```

This is the model (familiarize yourself with the notation!)

$$\mathbb{E}[\log(\texttt{price})] = \beta_0 + \texttt{year}\beta_{\texttt{year}} + \mathbb{1}_{[\texttt{ford}]}\beta_{\texttt{ford}} + \mathbb{1}_{[\texttt{gmc}]}\beta_{\texttt{gmc}}.$$

See other models and syntax in `pickups.R`.

R roundup

Quitting R: usually save the script, not the workspace.

The workspace is an .rda image, while the .R script is just text.

Stay up-to-date: Make sure you have latest versions.

Stay organized: keep track of your work.

Consider using shared drives to collaborate. Or, even better, create a group github repo and use version control.

R is a fantastic tool and there is tons you can do, but don't worry about learning everything right away.

Plenty of resources out there. Also don't hesitate to use the instructor, the preceptorme and your friends to avoid frustration.

I am learning R too!

Also remember google (and stackflow) is our friend!