DATA ANALYTICS AND MACHINE LEARNING WITH R

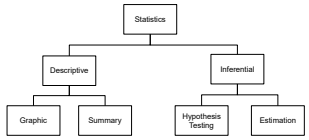
EXPLORATORY DATA ANALYSIS

LUIS GUSTAVO NARDIN
INTERNET TECHNOLOGY
BRANDENBURG UNIVERSITY OF TECHNOLOGY

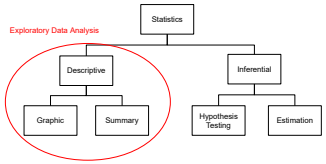
EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis refers to the critical process of performing **initial investigations on data** to discover patterns and to spot anomalies with the help of **graphic representations** and **summary statistics**.

EXPLORATORY DATA ANALYSIS



EXPLORATORY DATA ANALYSIS



GRAPHIC REPRESENTATION

GRAPHIC REPRESENTATION

- Basic Graphs
- Introduction to `ggplot2`
- `qplot()` function
- `ggplot()` function

BASIC GRAPHS

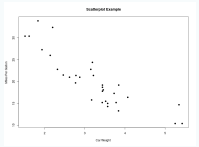
R provides some basic commands to create a graph

- Scatterplot Chart
- Bar Chart
- Line Chart
- Pie Chart
- Boxplot Chart

SCATTERPLOT CHART

The basic function is `plot(x, y)`, where x and y are numeric vectors denoting the `(x,y)` points to plot.

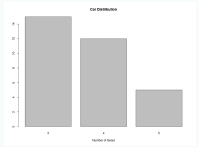
```
> attach(mtcars)
> plot(wt, mpg, main="Scatterplot Example",
+ xlab="Car Weight ", ylab="Miles Per Gallon ", pch=
```



BAR CHART

Create barplots with the `barplot(height)` function where height is a vector or matrix.

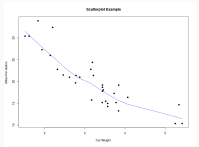
```
> counts <- table(mtcars$gear)
> barplot(counts, main="Car Distribution" , xlab="Nu
```



LINE CHART

Line charts are created with the function `lines(x, y, type)` where x and y are numeric vectors of `(x,y)` points to connect. type is the type of the line (see **Line Types**).

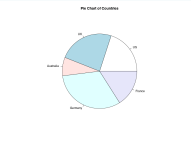
```
> attach(mtcars)
> plot(wt, mpg, main="Scatterplot Example",
+ xlab="Car Weight ", ylab="Miles Per Gallon ", pch=
> lines(lowess(wt,mpg), col="blue")
```



PIE CHART

Pie charts are created with the function `pie(x, labels)` where `x` is a non-negative numeric vector indicating the area of each slice and `labels` notes a character vector of names for the slices.

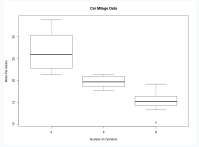
```
> slices <- c(10, 12,4, 16, 8)
> lbls <- c("US", "UK" , "Australia" , "Germany" , "
> pie(slices, labels=lbls, main="Pie Chart of Countri
```



BOXPLOT CHART

Boxplots can be created for individual variables or for variables by group. The format is `boxplot(x, data)`, where `x` is a formula and `data` denotes the data frame providing the data.

```
> boxplot(mpg~cyl,data=mtcars, main="Car Milage Data
+ xlab="Number of Cylinders", ylab="Miles Per Gallon
```



INTRODUCTION TO GGPLOT2

- A powerful R package for producing statistical, or data, graphics
- Based on the Grammar of Graphics (Wikinson, 2005)
- Not installed on R by default
  - `install.packages("ggplot2")`
- Load the package
  - library(ggplot2)

DOCUMENTATION

- Website **http://had.co.nz/ggplot2/**
- Books
  - Wilkison, L. (2005). *The grammar of graphics*. Springer.
  - Wickham, H. (2009). *ggplot2: Elegant graphics for data analysis*. Springer.
    - Code and sample chapters available at **http://ggplot2.org/book/**
  - Chang, W. (2013). *R graphics cookbook*. O'Relly.
    - Code and useful information about R and more specifically about ggplot2 available at **http://www.cookbook-r.com/**

DATA SETS

- `diamonds` data set provides ~ 54000 diamonds entries from **http://www.diamondse.info/**
- Structure of the data frame
  - `help(diamonds)`
  - `str(diamonds)`
- 10 variables: `price`, `carat`, `cut`, `color`, `clarity`, `x`, `y`, `z`, `depth`, and `table`

DATA SETS

- `economics` data set provides 478 US economic time series data from **http://research.stlouisfed.org/fred2**
- Structure of the data frame
  - `help(economics)`
  - `str(economics)`
- 6 variables: `date`, `psavert`, `pce`, `unemploy`, `uempmed` and `pop`

TERMINOLOGY

- **Aesthetics** refers to characteristics of the plots like shape, color, and size
- **Faceting** refers to generate a plot displaying multiple plots of different subsets of the data

GGPLOT2 FUNCTIONS

- Provide two major functions
  - `qplot` - for quick plots
  - `ggplot` - for fine, granular control of everything
- Main difference between `qplot` and `ggplot` is that the former allows the use of vectors, while the latter requires the use of data frame

QPLOT

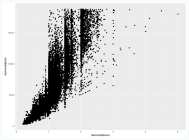
- Wraps up al the details of `ggplot` with a familiar syntax from `plot`
- Automatically scales data
- Can produce any type of plot
- Faceting and margins
- Creates objects that can be saved and modified

QPLOT

```
qplot(x, y = NULL, ..., data, facets = NULL,
 margins = FALSE, geom = "auto", xlim = c(NA,
NA), ylim = c(NA, NA), log = "", main = NULL,
    xlab = deparse(substitute(x)), ylab =
      deparse(substitute(y)), asp = NA)
```

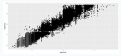
SCATTERPLOT

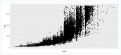
- `qplot(diamonds$carat, diamonds$price)`
- `qplot(carat, price, data=diamonds, geom="point")`
- `qplot(carat, price, data=diamonds)`

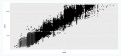


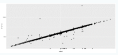
TRANSFORMING VARIABLES

- `qplot` qplot accepts functions of variables as arguments
  - `qplot(log(carat), log(price), data=diamonds)`
  - `qplot(carat, price, data=diamonds,log="x")`
  - `qplot(carat, price, data=diamonds, log="xy")`
  - `qplot(carat, x * y * z, data=diamonds, log="xy")`



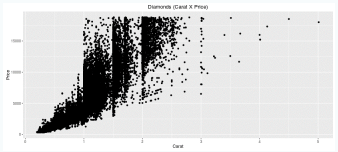






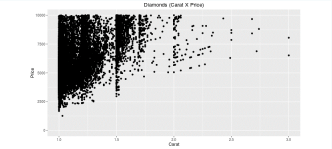
SIMPLE LAYOUT MODIFICATIONS

- Change the labels `main`, `xlab`, and `ylab`
- `qplot(carat, price, data=diamonds, **main="Diamonds (Carat X Price)", xlab="Carat", ylab="Price"**)`



SIMPLE LAYOUT MODIFICATIONS

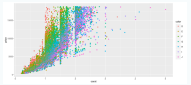
- Change the limits and aspect `xlim`, `ylim`, and `asp`
- `qplot(carat, price, data=diamonds, main="Diamonds (Carat X Price)", xlab="Carat", ylab="Price", xlim=c(1,3), ylim=c(0,10000), asp=0.5)`



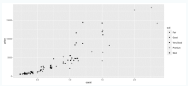
COLOR, SHAPE, AND SIZE

- `qplot` automatically handles color and shape. In addition, it also provide a legend.

```
qplot(carat, price,
data=diamonds, color=color)
```

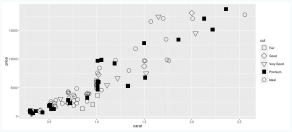


```
qplot(carat, price,
data=diamonds[sample(nrow(diamonds),
100),], shape=cut)
```



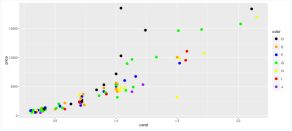
COLOR, SHAPE, AND SIZE

- Manually change the color and shape defined by `qplot` (**Shapes and Line Types**)
- `qplot(carat, price, data=diamonds[sample(nrow(diamonds), 100),], **shape=cut, size=I(5)) + scale_shape_manual(values = c(0, 5, 6, 15, 1))**`



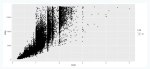
COLOR, SHAPE, AND SIZE

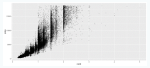
- Manually change the color and shape defined by `qplot` (**Shapes and Line Types**)
- `qplot(carat, price, data=diamonds[sample(nrow(diamonds), 100),], color=color, size=I(3)) + scale_color_manual(values = c("black", "orange", "blue", "green", "yellow", "red", "purple"))`

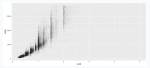


TRANSPARENCY

- The `alpha` parameter allows to manipulate transparency
  - `qplot(carat, price, data=diamonds, **alpha=1/10**)`
- Necessary to use `I()` to inhibit interpretation
  - `qplot(carat, price, data=diamonds, **alpha=I(1/10)**)`
  - `qplot(carat, price, data=diamonds, **alpha=I(1/10)**)`







GEOMETRIC OBJECTS

- Scatterplot is the default geometric object of `qplot`
- ggplot2 provides several other geometric objects to generate graphics.

GEOMETRIC OBJECTS

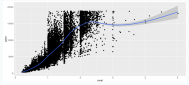
The most common and useful geoms are:

- `geom="point"` draws points to produce scatterplots
- `geom="smooth"` fits a smoother to the data
- `geom="boxplot"` produces a box-and-whisker plot
- `geom="path"` and `geom="line"` draw line between data points
- `geom="histogram"` draws a histogram
- `geom="density"` creates a density plot
- `geom="bar"` makes bar charts for discrete variables

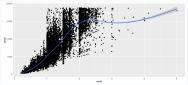
SMOOTH

- It can be hard to see the trend shown by the data. Smooth creates a line representing this trend.

```
qplot(carat, price,
data=diamonds,
geom=c("point", "smooth"))
```



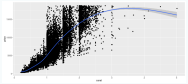
```
qplot(carat, price,
data=diamonds) +
stat_smooth(se=TRUE,
level=0.5))
```



SMOOTH

- There are different smoothers that can be chosen by using the `method` argument in the `stat_smooth`:
  - `loess` (default) uses a smooth local regression
  - `gam` (requires `mgcv` package) fits a generalized additive model
  - `lm` (requires `splines` package) fits a linear model
  - `rlm` (requires `MASS` package) uses a robust fitting algorithm

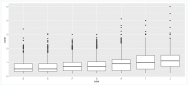
```
library(splines)
qplot(carat, price,
data=diamonds) +
stat_smooth(method="lm",
formula=y ~ ns(x,5))
```



BOXPLOT AND JITTER

- Data includes a categorical and one or more continuous variables
- You can plot how the values of the continuous variables vary with the levels of the categorical variable

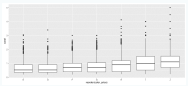
```
qplot(color, carat,
data=diamonds,
geom="boxplot")
```



BOXPLOT AND JITTER

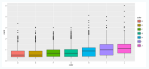
- Reordering the boxplots

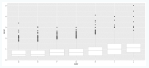
```
qplot(reorder(color, price),
carat, data=diamonds,
geom="boxplot")
```

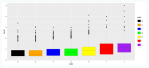


BOXPLOT AND JITTER

- qplot(color, carat, data = diamonds, geom="boxplot", **fill=color**)
- qplot(color, carat, data = diamonds, geom="boxplot", **size=I(0.1)**)
- qplot(color, carat, data = diamonds, geom="boxplot", **fill=color, size=I(0.1)) + scale_fill_manual(values = c("black", "orange", "blue", "green", "yellow", "red", "purple")**)



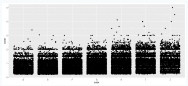




BOXPLOT AND JITTER

- Jitter allows to see the actual distribution of the data as it plots all the points categorized

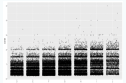
```
qplot(color, carat,
data=diamonds, geom="jitter")
```



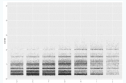
BOXPLOT AND JITTER

```
qplot(color, carat, data=diamonds,
geom="jitter", alpha=[alpha])
```

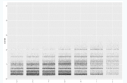
[alpha]=I(1/5)



[alpha]=I(1/10)



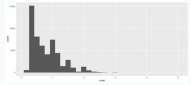
[alpha]=I(1/100)



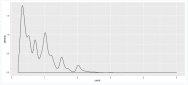
HISTOGRAMS AND DENSITIES

- Show the distribution of a single variable.

```
qplot(carat, price,
data=diamonds,
geom="histogram")
```



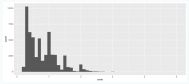
```
qplot(carat, price,
data=diamonds,
geom="density")
```



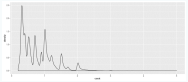
HISTOGRAMS AND DENSITIES

- The control of smoothness in histogram and density plots can be changed. Use `binwidth` for histogram and `adjust` for density plots.

```
qplot(carat, price,
data=diamonds,
geom="histogram",
binwidth=0.1)
```

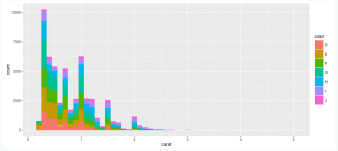


```
qplot(carat, price,
data=diamonds,
geom="density", adjust=0.5)
```



HISTOGRAMS AND DENSITIES

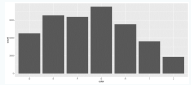
```
qplot(carat, data=diamonds, geom="histogram",
      binwidth=0.1, fill=color)
```



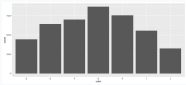
BAR CHART

- Discrete analogue of histogram.

```
qplot(color, data=diamonds,
geom="bar")
```



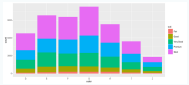
```
qplot(color, data=diamonds,
geom="bar", weight=carat)
```



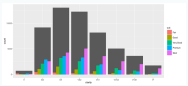
BAR CHART

- Discrete analogue of histogram.

```
qplot(clarity, data=diamonds,
geom="bar", fill=cut)
```



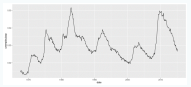
```
qplot(clarity, data=diamonds,
geom="bar") +
geom_bar(aes(fill=cut),
data=diamonds,
position="dodge")
```



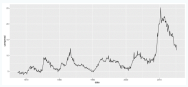
LINE AND PATH

- Line and path plots are typically used for time series data `economics` data set.

```
qplot(date, unemploy / pop,
data=economics,
geom="line")
```



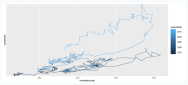
```
qplot(date, uempmed,
data=economics,
geom="line")
```



LINE AND PATH

```
year <- function(x){
as.POSIXlt(x)$year + 1900 }
```

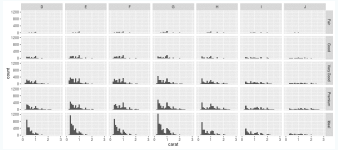
- ```
  qplot(unemploy / pop,
  uempmed, data=economics,
  geom="path",
  color=year(date)) +
  scale_y_discrete()
  ```
- ```
  qplot(unemploy / pop,
  uempmed, data=economics,
  geom="path",
  color=as.factor(year(date)))
  + scale_y_discrete()
  ```



FACETING

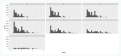
- Creates plots arranged on a grid specified by a faceting formula

```
qplot(carat, data=diamonds, facets=cut ~ color,
geom="histogram", binwidth=0.1, xlim=c(0,3))
```

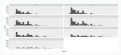


FACETING

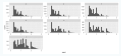
```
g <- qplot(carat,
data=diamonds,
geom="histogram",
binwidth=0.1, xlim=c(0,3))
```



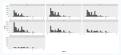
- g + facet_wrap(~color)



- g + facet_wrap(~color, ncol=2)



- g + facet_wrap(~color, scales="free")

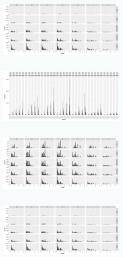


- g + facet_wrap(~color, scales="free_x")

FACETING

```
g <- qplot(carat,
data=diamonds,
geom="histogram",
binwidth=0.1, xlim=c(0,3))
```

- `g + facet_grid(cut~color)`
- `g + facet_grid(~color+cut)`
- `g + facet_grid(cut~color, scales="free")`
- `g + facet_grid(~color, scales="free_x")`



GGPLOT

OTHER PLOT FUNCTIONS

- `qplot` or `ggplot`can be assigned to a variable in R for manipulation
  - `g <- qplot(carat, data=diamonds, geom="histogram", binwidth=0.1, xlim=c(0,3))`
  - `g + facet_grid(cut~color)`
- Save/load raw plot into a file
  - `save(g, file="rawG.data")`
  - `load("rawG.data")`

OTHER PLOT FUNCTIONS

- Get information about the plot
  - `summary(g)`
- Saving a figure
  - `ggsave(file="test.pdf", plot=g)`
  - `ggsave(file="test.jpeg", dpi=72, plot=g)`
  - `ggsave(file="test.png", plot=g, width=10, height=5)`

EXERCISES

EXERCISE 1

Upload the file `bp.txt`

- `HEIGHT` (cm)
- `WEIGHT` (cm)
- `WAIST` (cm)
- `HIP` (cm)
- `BPSYS` (Systolic pressure)
- `BPDIA` (Diastolic pressure)

EXERCISE 1

- Generate a scatterplot of HEIGHT and WEIGHT
- Experiment with color, size, and shape aesthetics
- Add a column TYPE=[A, B, D, B, A, A]
- Generate a boxplot of TYPE with BPSYS and BPDIA

EXERCISE 2

- `diamonds` data set provides ~ 54000 diamonds entries from **http://www.diamondse.info/**
- Available in `ggplot2` package
- Structure of the data frame
  - `help(diamonds)`
  - `str(diamonds)`
- 10 variables: `price`, `carat`, `cut`, `color`, `clarity`, `x`, `y`, `z`, `depth`, and `table`

EXERCISE 2

- Identify the numeric columns and summarize them: Average, Median, Standard Deviation, Minimum and Maximum
- Generate a new data frame containing all diamonds whose price is greater than 10000 and cart greater or equal than 3
- Generate a boxplot by price and color
- Generate a histogram of the price

SUMMARY STATISTICS