

# Data Exploration, Visualization, and Feature Engineering

Data Science Dojo

# Agenda

- Why data exploration and visualization
- Exploration and visualization using R
  - Core R functionality – iris dataset
  - Lattice package – mtcars dataset
  - ggplot2 package – diamonds data set
- Story-telling with data
  - Titanic data set

# WHY DATA EXPLORATION AND VISUALIZATION

# Data beats algorithm but...

- More data usually yields good generalization performance, even with a simple algorithm
- But there are caveats
  - Amount of data may have diminishing returns
  - Data quality and variety matters
  - A decent performing learning algorithm is still needed
  - Most importantly, extracting useful features out of data is important

# Dispelling common myths

- There is *NO* single ML algorithm that will take raw data and give you the best model



- You do *NOT* need to know a lot of machine learning algorithms to build robust predictive models

# Janitorial work is important

- Not spending time on understanding your data is a source of many problems!
- Remember the 80/20 rule
  - 80% : Data cleaning, data exploration, feature engineering, pre-processing etc...
  - 20% : Model building

# EXPLORATION AND VISUALIZATION USING R

# Objectives

- Develop an understanding of the high-level thinking process of data exploration
- Making sense of data using visualization techniques
- Learning to perform feature engineering
- Becoming a good story teller.



# I am new to R

- Focus on ideas/concepts rather than exact syntax. R help is your friend. 😊

`?mean, ?sd`

`help( )`

`example( )`

- All slides have code samples
- Sample code + slides: 'Data Exploration and Visualization' folder

# Common Graphical Parameters

- Title of graph using the **main** function, main = "title"
- Label x- axis by using the **xlab** function, xlab = "label x axis"
- Label y- axis by using the **ylab** function, ylab = "label y axis"
- Colors controlled by **col**
- Get legends of layered plots with **auto.key=TRUE**

# R to start exploring data commands

Commands	Description
<b>read.csv() , read.table()</b>	Load data/file into a dataframe
<b>data()</b>	Loads or resets a dataset
<b>names()</b>	List names of variables in a dataframe
<b>head()</b>	First 6 rows of data
<b>tail()</b>	Last 6 rows of data
<b>str()</b>	Display internal structure of R object
<b>View()</b>	View dataset in spreadsheet format in RStudio
<b>dim()</b>	Dimensions( rows and columns) of dataframe
<b>summary()</b>	Display 5-number summary and mean
<b>colnames()</b>	Provide column names

# CORE R GRAPHICS

# The “iris” data set

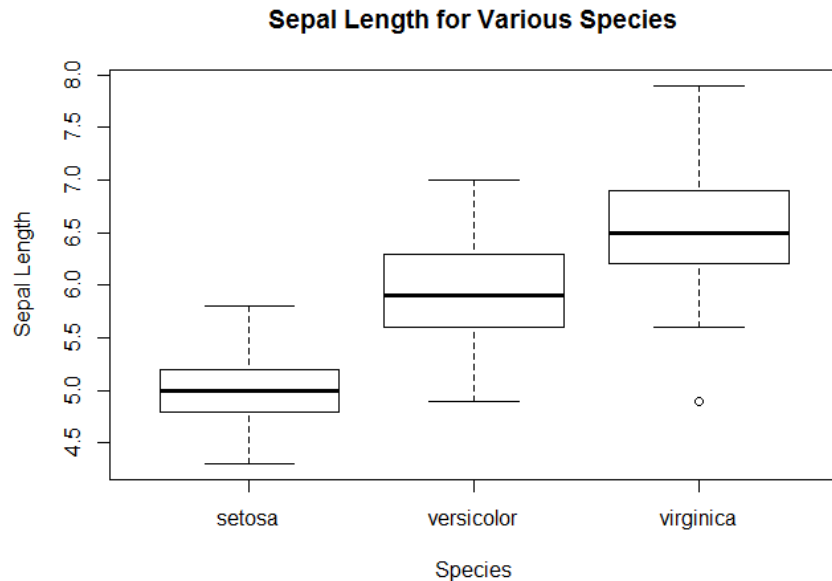
```
data(iris)  
head(iris)
```

```
> head(iris)  
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species  
1          5.1         3.5          1.4          0.2  setosa  
2          4.9         3.0          1.4          0.2  setosa  
3          4.7         3.2          1.3          0.2  setosa  
4          4.6         3.1          1.5          0.2  setosa  
5          5.0         3.6          1.4          0.2  setosa  
6          5.4         3.9          1.7          0.4  setosa
```

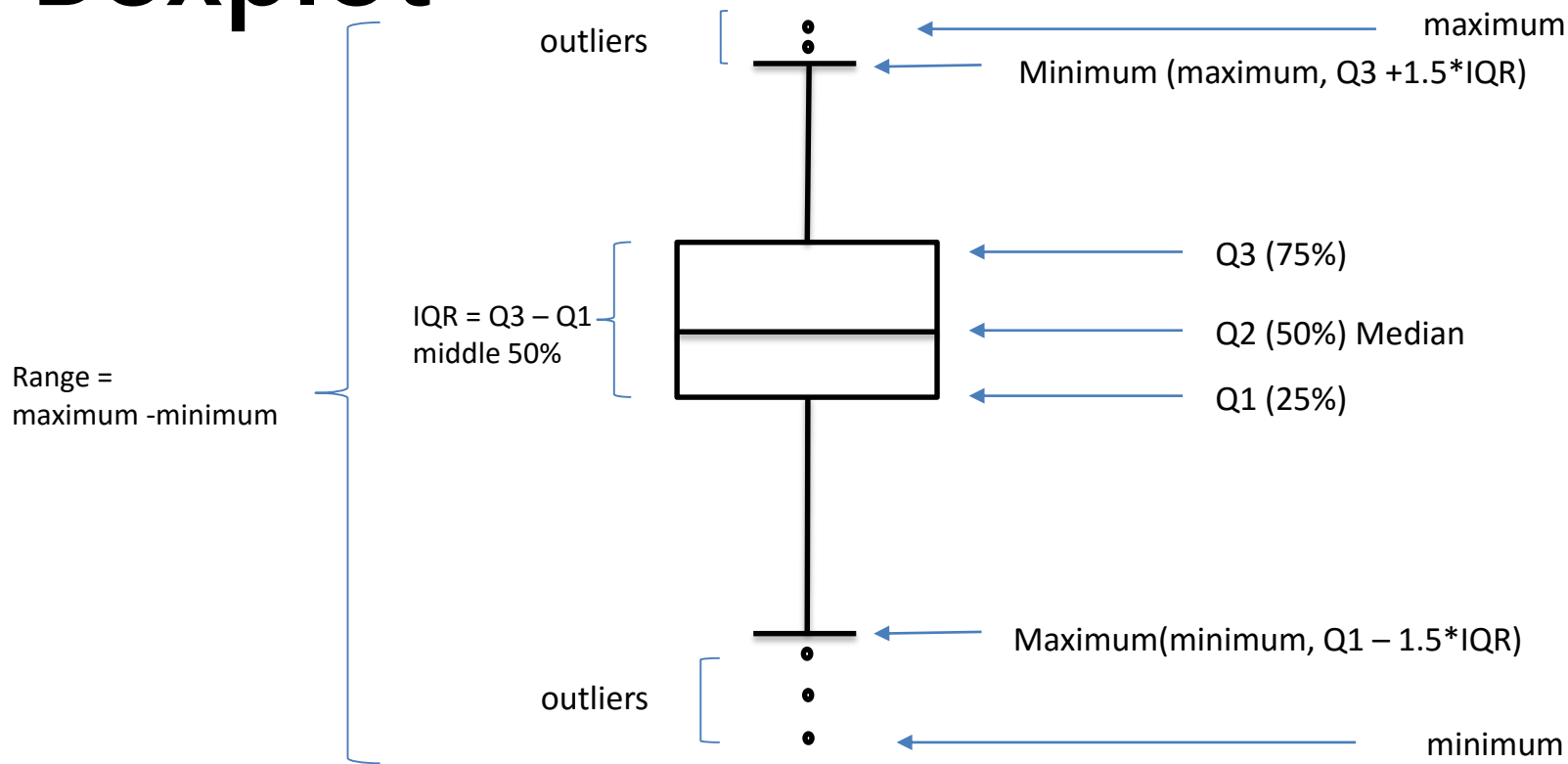
# Boxplots

- Summarizes *quantitative/numeric* data

```
# Core Graphics
boxplot(
  Sepal.Length ~ Species,
  data=iris,
  main="Sepal Length for
  Various Species",
  xlab="Species",
  ylab="Sepal Length"
)
```



# Boxplot



# Saving plots

- Save various formats

```
pdf("myplot.pdf")  
boxplot(Sepal.Length ~ Species,  
data=iris)  
dev.off() # Returns plot to the  
IDE
```

Function	Output to
pdf("mygraph.pdf")	pdf file
win.metafile("mygraph.wmf")	windows metafile
png("mygraph.png")	png file
jpeg("mygraph.jpg")	jpeg file
bmp("mygraph.bmp")	bmp file
postscript("mygraph.ps")	postscript file

Windows Saves to default: Libraries\Documents

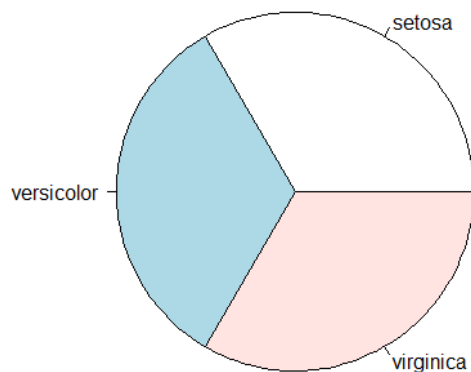
R Studio makes it easier



# Pie Chart

- Summarizes *qualitative/categorical* variables

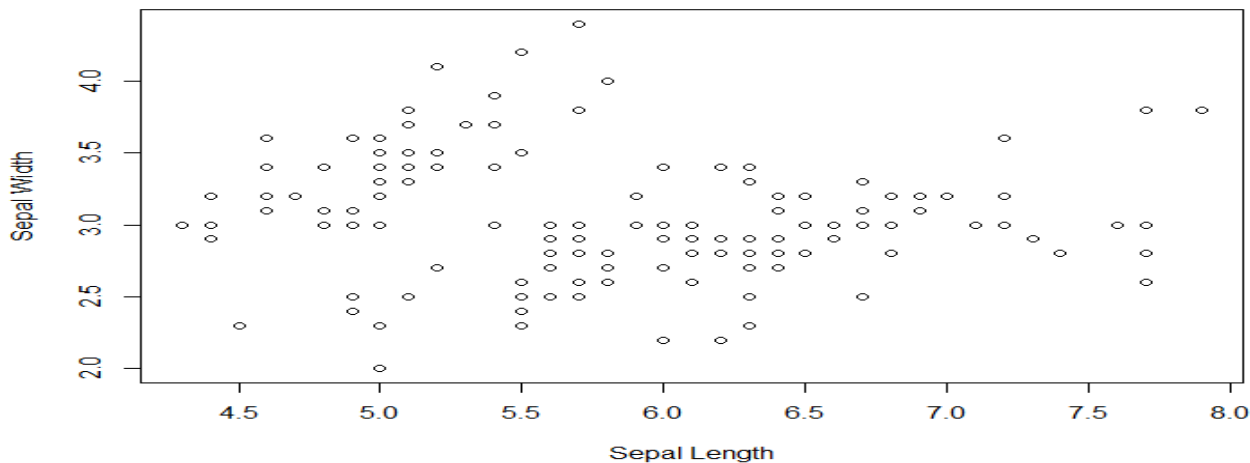
```
# Core Graphics  
pie(table(iris$Species))
```



# Plot

- Visual depiction of correlation between numeric quantities

```
# Core Graphics  
plot(Sepal.Length ~ Sepal.Width,  
data=iris, xlab= "Sepal Length",  
ylab= "Sepal Width")
```

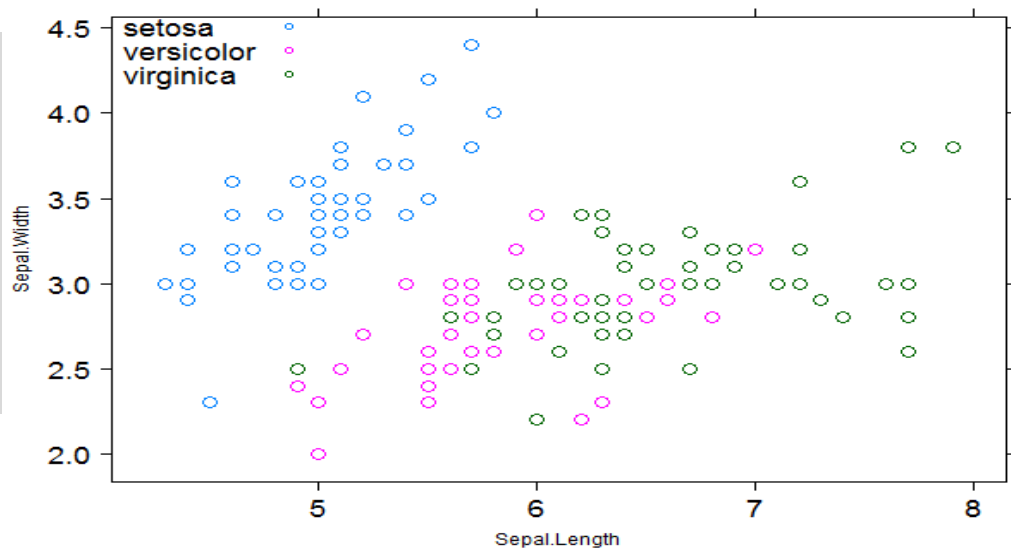


# LATTICE GRAPHICS

# xyplot

- Plot counterpart in Lattice package.

```
# Lattice Graphics
library(lattice)
xyplot(Sepal.Width ~
  Sepal.Length, data=iris,
  groups=Species,
  auto.key=TRUE
)
```

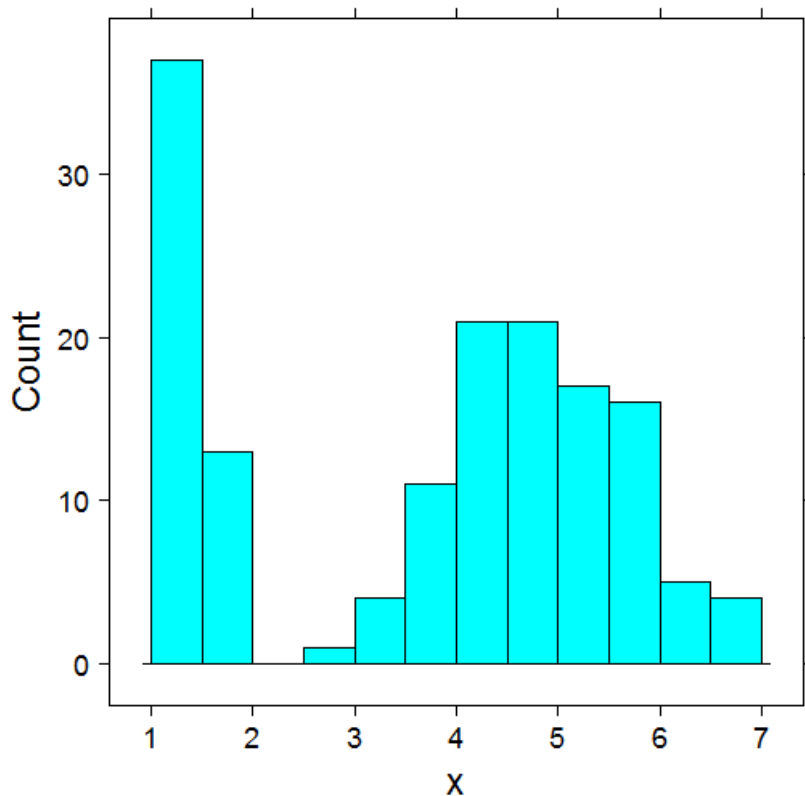


# In-class Exercise 1

Contrast 2-D scatterplots for iris dataset(Petal.Length and Petal.Width) Summarize your findings.

# Histogram

Histogram

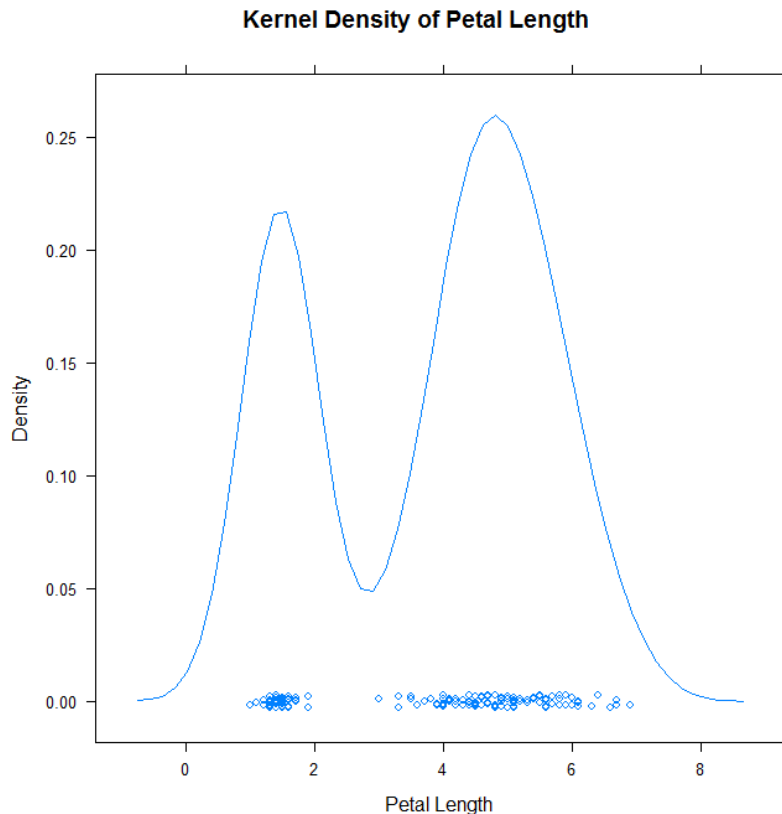


- Spread of a numeric feature
- Places values in "bins"

```
histogram(iris$Petal.Length,  
breaks=10, main="Histogram")
```

Vary 'breaks' parameter

# Density plots



- Variation on histogram
- Estimates density function from counts
- Does not work with missing values

```
densityplot(iris$Petal.Length,  
main="Kernel Density of Petal  
Length", xlab="Petal Length")
```

Try adding `plot.points=F`

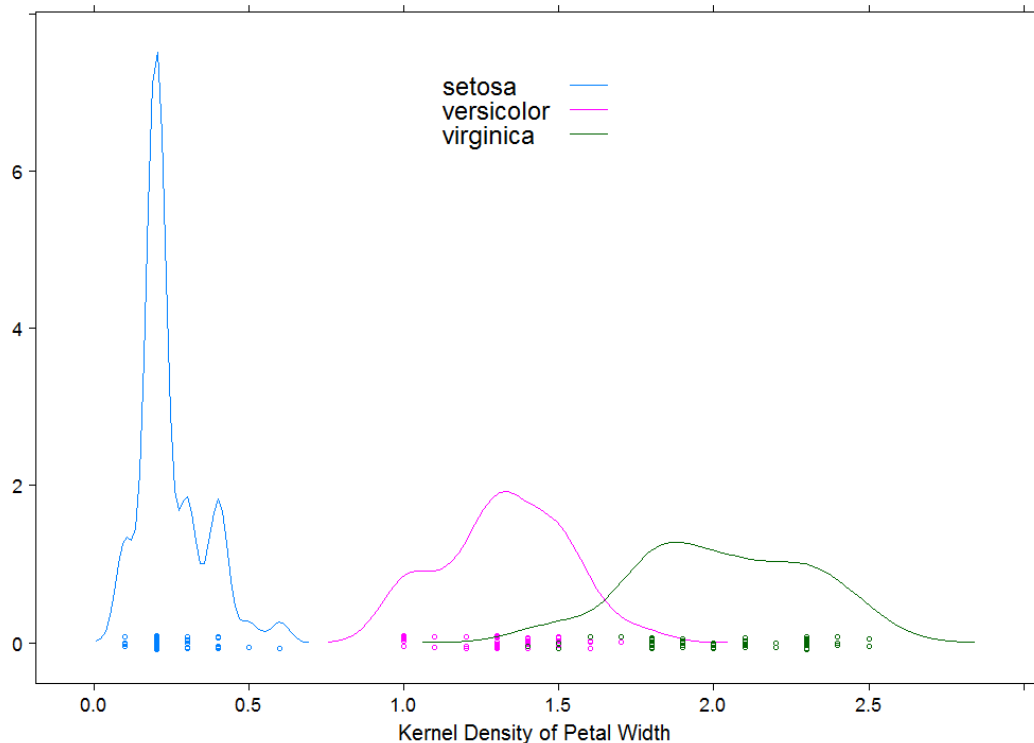
# The devil is in the details

- The details are in segments
- Segmentation reveals hidden patterns
- Create as many segments as possible
  - Your domain understanding will help in creating segments



# Multiple density plots

Density of Petal Width by Species



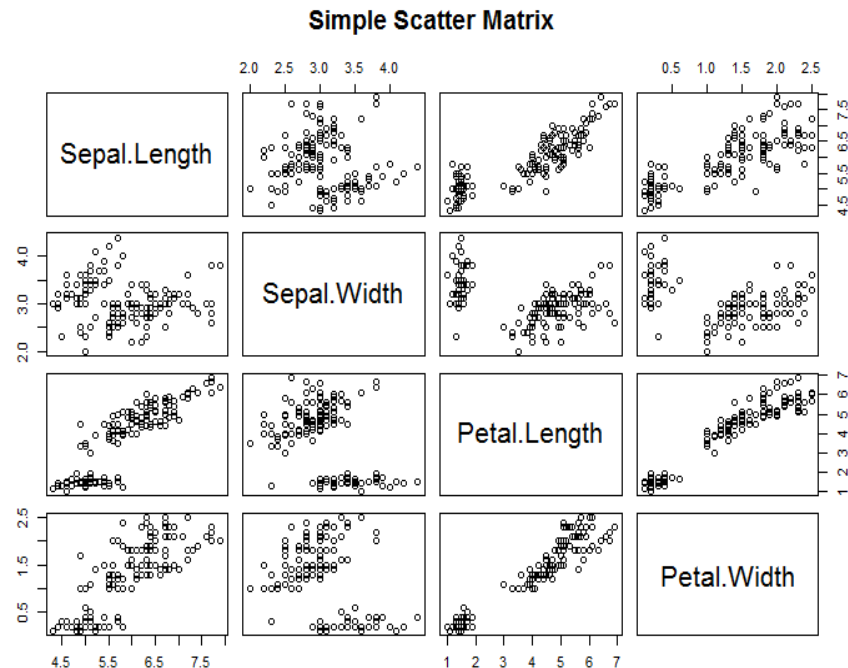
```
densityplot(~Petal.Width,  
data=iris,  
groups=Species,  
auto.key=TRUE,  
xlab="Kernel Density of  
Petal Width",  
ylab="Frequency",  
main=list(label="Density  
of Petal Width by  
Species"))
```

# Scatterplot matrix

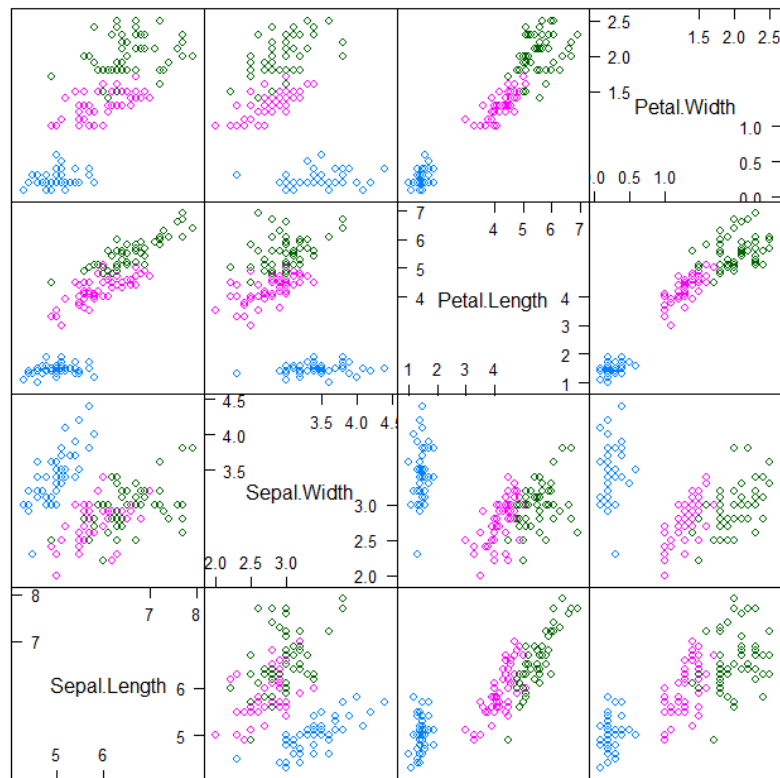
- Multiple relationships on one graph
- Good for initial explorations

```
# Core Graphics
```

```
pairs(  
iris[,1:4],  
main="Scatterplot Matrix"  
)
```



# Scatterplot matrix

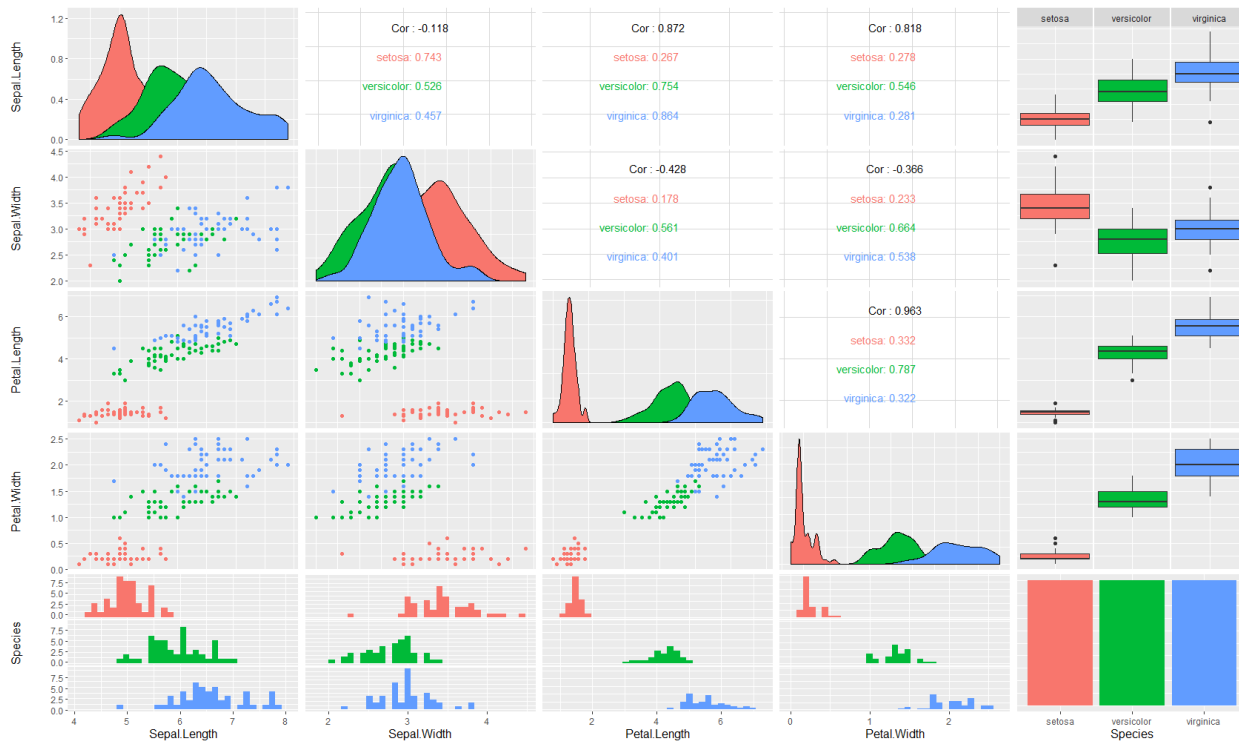


Scatter Plot Matrix

```
# Lattice Graphics
```

```
splom(iris[1:4],  
groups=iris$Species)
```

# Enhanced Scatterplot matrix



- Very slow!
- Use carefully

```
library(GGally)
```

```
ggpairs(iris,  
  ggplot2::aes(color=Species))
```

# In-class Exercise 2

Using the "mtcars" dataset, predict mpg based on other columns.

Create at least 2 different plots illustrating useful relationships in data and summarize your findings.

# The “mtcars” data set

```
data(mtcars)
head(mtcars)
```

```
> head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

# GGPLOT2 GRAPHICS

# The diamonds data set

```
library(ggplot2)
data(diamonds)
head(diamonds)
```

```
> head(diamonds)
```

	carat	cut	color	clarity	depth	table	price	x	y	z
1	0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
2	0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
3	0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
4	0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
5	0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
6	0.24	Very Good	J	VVS2	62.8	57	336	3.94	3.96	2.48



# ggplot Fundamentals

- `ggplot()` is the basic function
- `geom_*()` creates a graph layer
  - `geom_histogram()`
  - `geom_boxplot()`
- `aes()` defines an "aesthetic" either globally or by layer

# Layering

```
ggplot(diamonds, aes()) + geom_point()
```

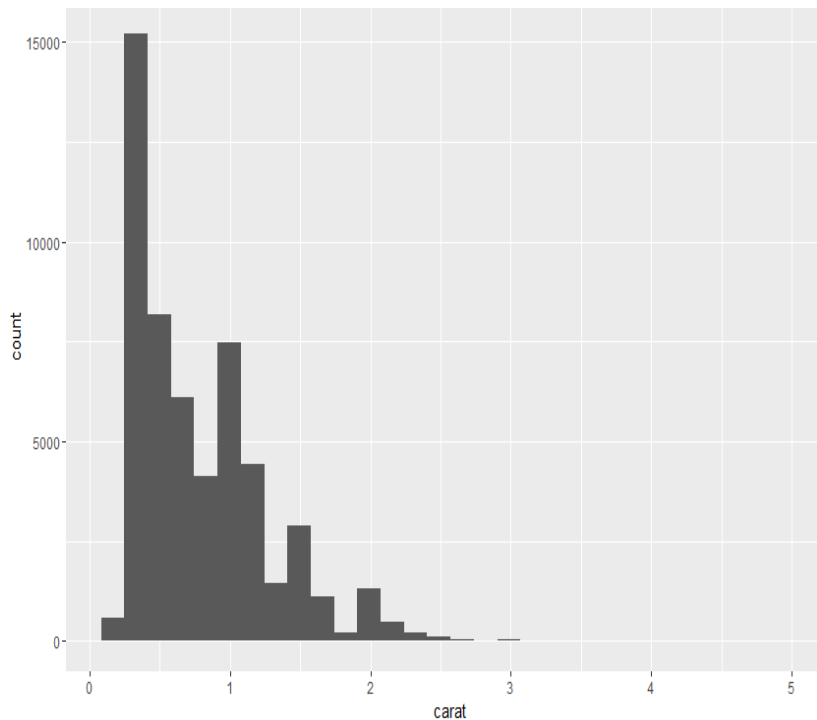


Layer 1



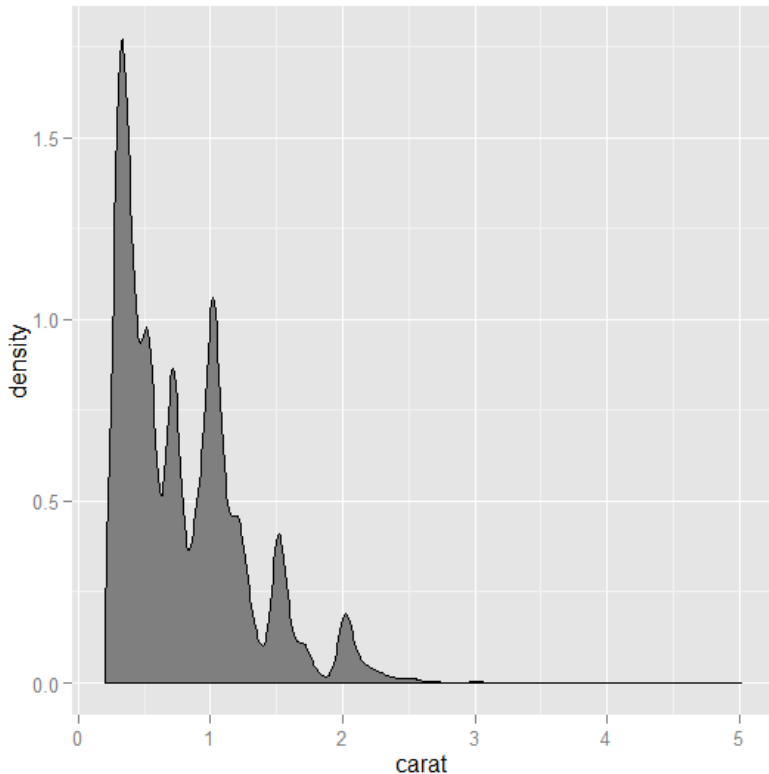
Layer 2

# Histogram



```
ggplot(diamonds, aes(x=carat)) +  
geom_histogram()
```

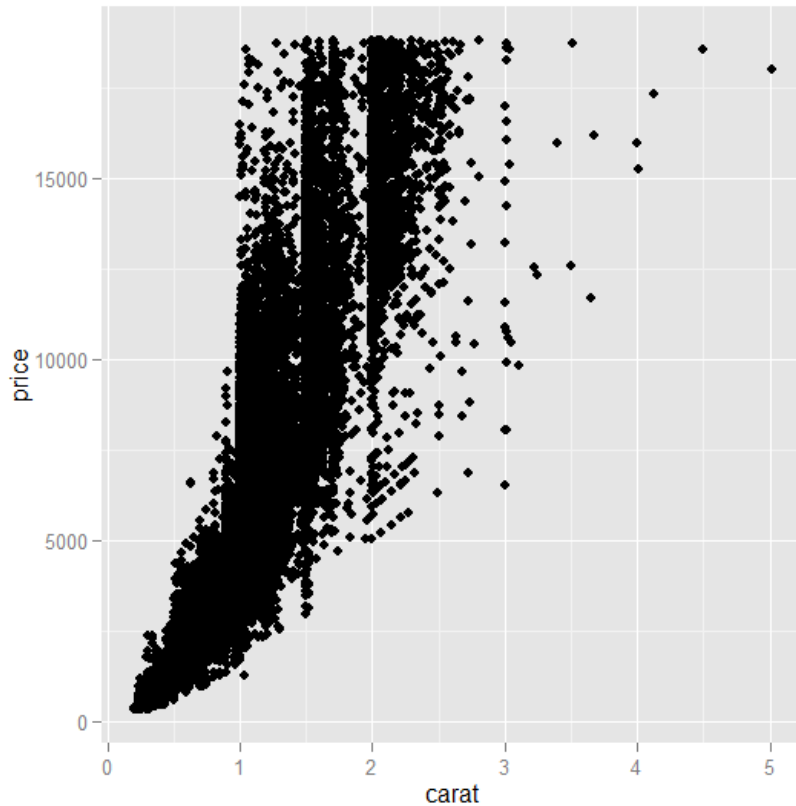
# Density plot



```
ggplot(diamonds) +  
  geom_density(aes(x=carat),  
    fill="gray50")
```

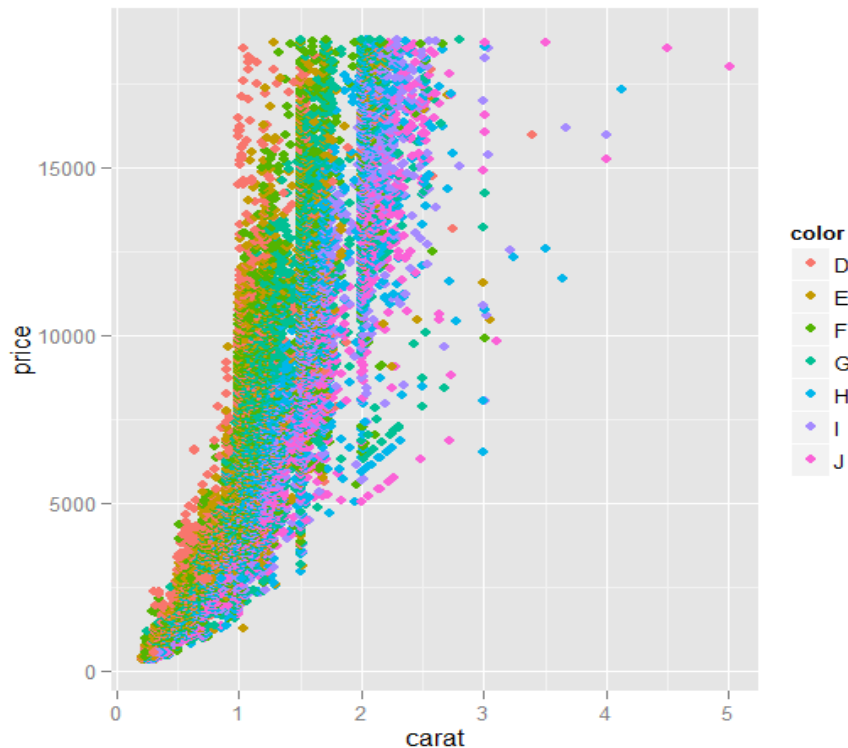
- Note the location of aes()

# Scatter plots



```
ggplot(diamonds,  
  aes(x=carat,y=price)) +  
  geom_point()
```

# ggplot object



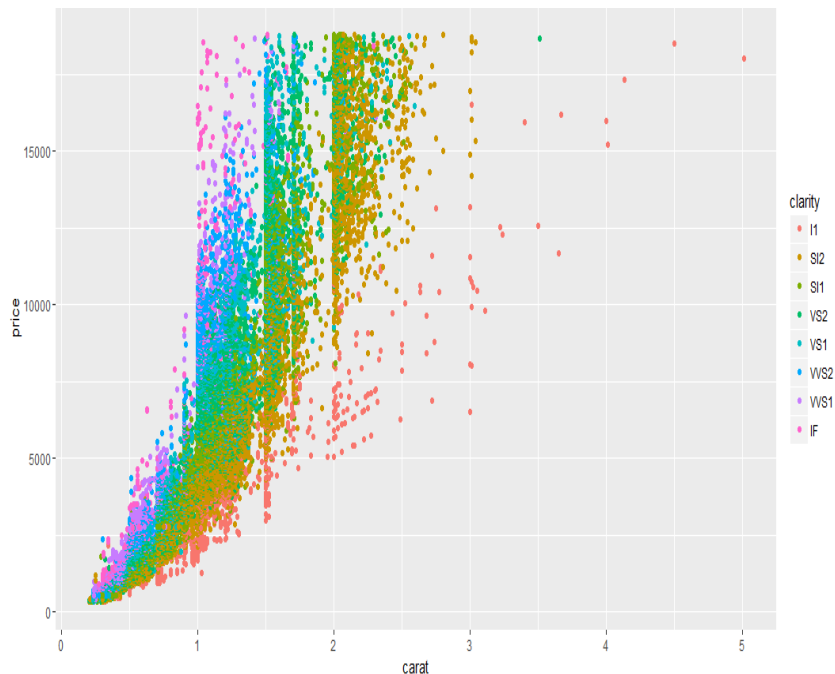
```
# Store the plot for future  
modification
```

```
g <- ggplot(diamonds,  
aes(x=carat, y=price))
```

```
# add settings specific to  
geom_point layer
```

```
g + geom_point(aes(color=color))
```

# ggplot object



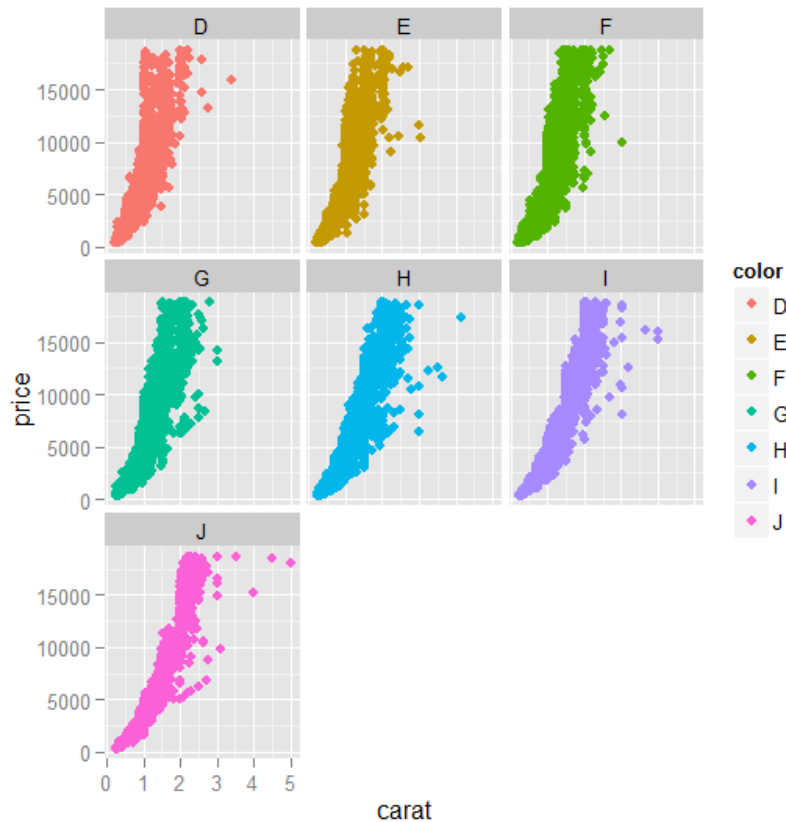
```
# Store the plot for future  
modification
```

```
g <- ggplot(diamonds,  
aes(x=carat, y=price))
```

```
# add settings specific to  
geom_point layer
```

```
g +  
geom_point(aes(color=clarity))
```

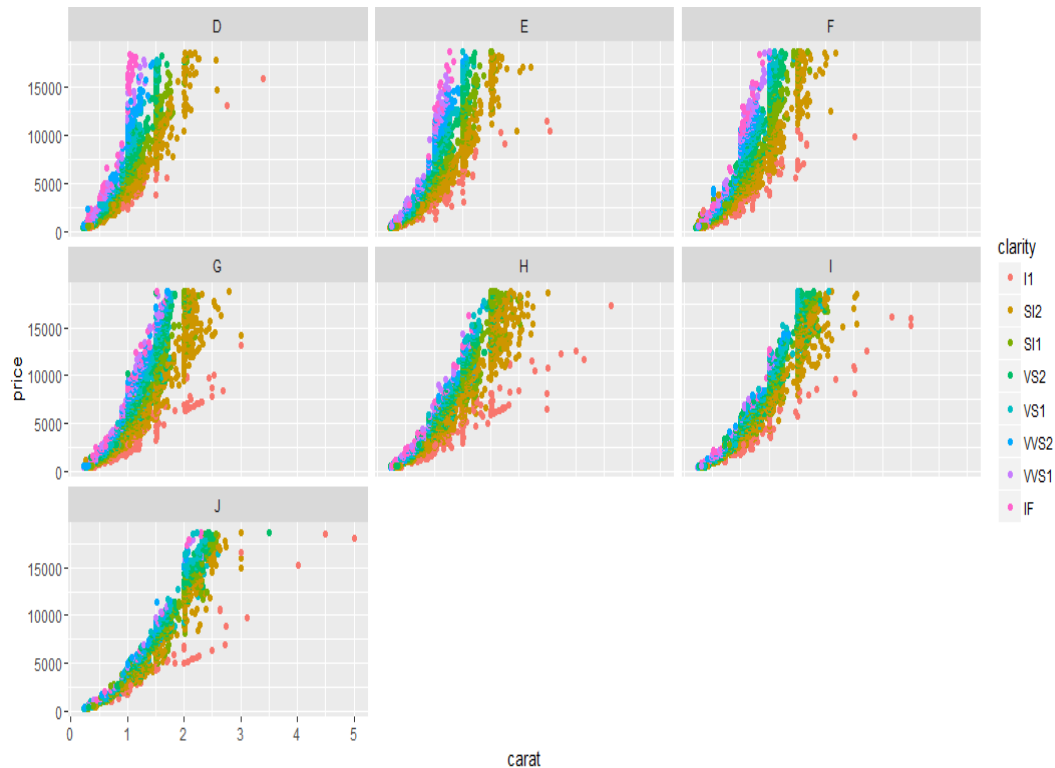
# Separating the segments



```
g +  
  geom_point(aes(color=color)) +  
  facet_wrap(~ color)
```

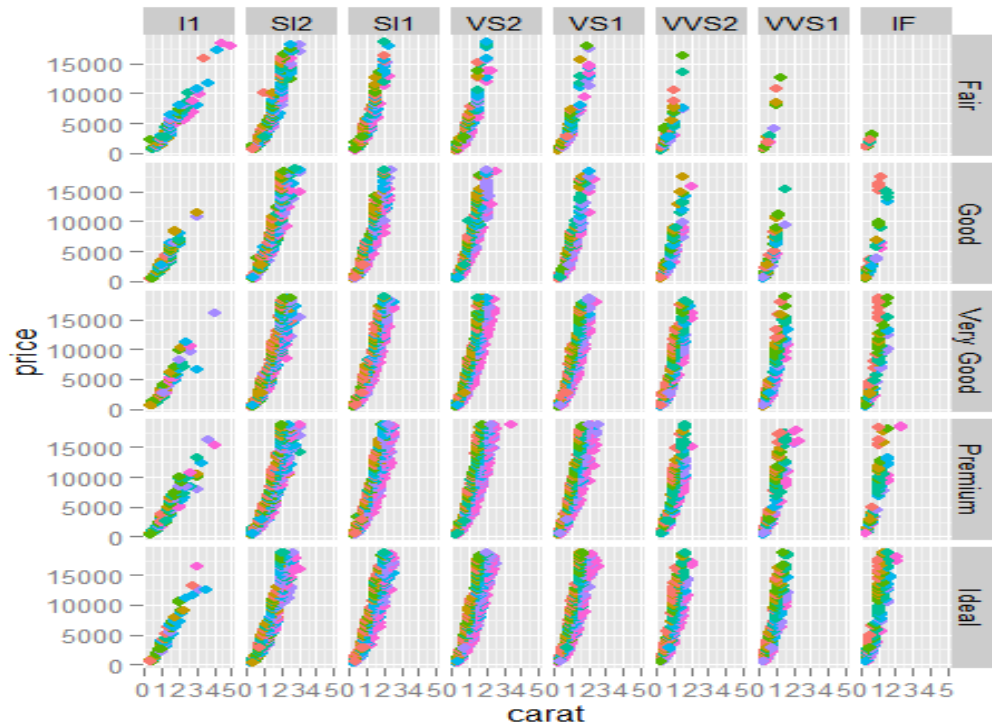


# Separating the segments



```
g +  
  geom_point(aes(color  
=clarity)) +  
  facet_wrap(~ clarity)
```

# More segments!

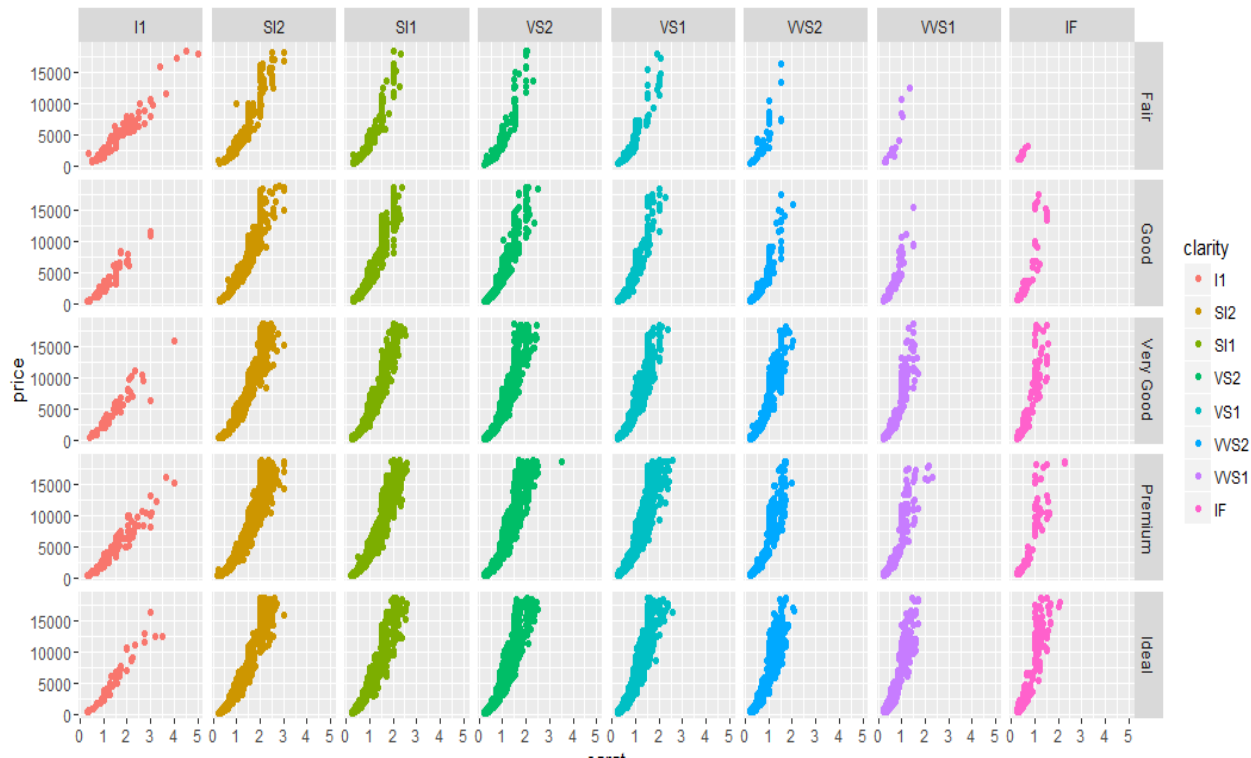


```
g +  
  geom_point(aes(color=color)) +  
  facet_grid(cut ~ clarity)
```

color

D  
E  
F  
G  
H  
I  
J

# More segments!



```
g +  
  geom_point(aes(  
    color=clarity  
  )) +  
  facet_grid(cut  
    ~ clarity)
```

# Summary

- ✓ Basics of R
- ✓ Graphing in R – core, lattice and ggplot2  
Look at multiple types of graphs.
- ✓ Visualize and segment dataset to gain insights about data.
- ✓ Identify key features
- ✓ Summarizing findings

# STORYTELLING WITH TITANIC

# Finding the data set

- Set your working directory to the bootcamp root
- Load data in from "Datasets/titanic.csv"

# Looking at the first few rows

```
titanic <- read.csv("Datasets/titanic.csv")  
head(titanic)
```

```
> head(titanic)
```

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500		S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250		S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500		S
6	0	3	Moran, Mr. James	male	NA	0	0	330877	8.4583		Q

```
> |
```

What features should we consider?

# What is the data type of each column?

```
str(titanic)
```

```
'data.frame':      891 obs. of  12 variables:
```

```
$ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
```

```
$ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
```

```
$ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
```

```
$ Name       : Factor w/ 891 levels "Abbing, Mr. Anthony",...: 109 191 358 277 16 559 520 629 417 581 ...
```

```
$ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
```

```
$ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
```

```
$ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
```

```
$ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
```

```
$ Ticket     : Factor w/ 681 levels "110152","110413",...: 524 597 670 50 473 276 86 396 345 133 ...
```

```
$ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
```

```
$ Cabin      : Factor w/ 148 levels "", "A10", "A14",...: 1 83 1 57 1 1 131 1 1 1 ...
```

```
$ Embarked   : Factor w/ 4 levels "", "C", "Q", "S": 4 2 4 4 4 3 4 4 4 2 ...
```



# Casting

## Set target column as a factor

```
titanic$Survived <- as.factor(titanic$Survived)
```

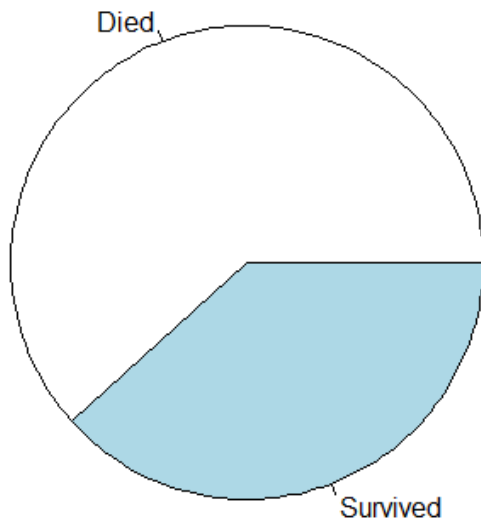
## Rename factors and columns

```
levels(titanic$Survived) <- c("Dead", "Survived")  
levels(titanic$Embarked) <- c("Unknown", "Cherbourg",  
                             "Queenstown", "Southampton")  
str(titanic[,c("Embarked", "Survived")])
```

```
'data.frame':  891 obs. of  2 variables:  
 $ Embarked: Factor w/ 4 levels  
 "Unknown", "Cherbourg", ...: 4 2 4 4 4 3 4 ...  
 $ Survived: Factor w/ 2 levels "0","1": 1 2 2 2  
 1 1 1 1 2 2 ...
```

# Class distribution: Pie Chart

```
survivedTable <- table(titanic$Survived)
par(mar=c(0, 0, 0, 0))
pie(survivedTable, labels=c("Died", "Survived"))
```



# Is **Sex** a Good predictor?

```
#Identify where sex = male for all columns
male <- titanic[titanic$Sex == "male",]

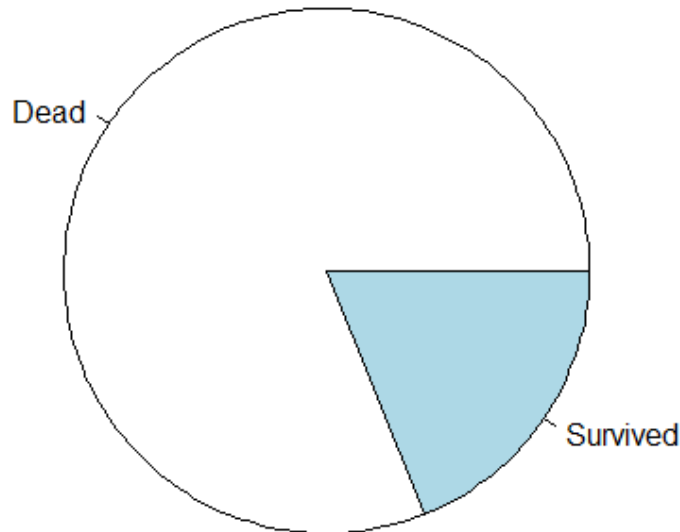
#Identify where sex = female for all columns
female <- titanic[titanic$Sex == "female",]

par(mfrow=c(1,2)) #two figures arranged in 1 row and 2
columns
pie(table(male$Survived), labels=c("Dead","Survived"))

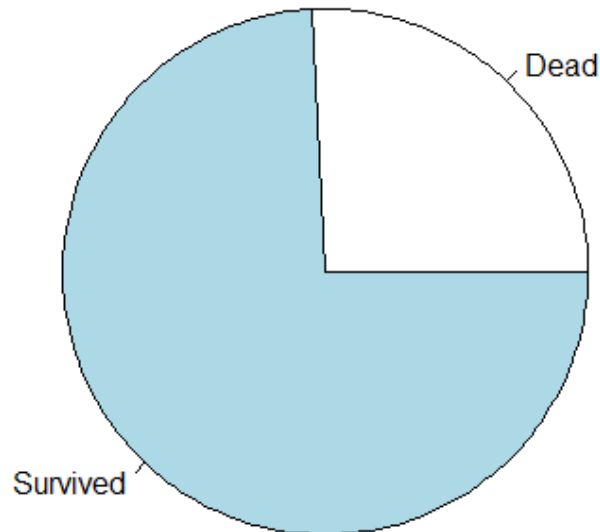
pie(table(female$Survive), labels=c("Dead","Survived"))
```

# Is **Sex** a Good predictor?

Survival Proportion Among Men



Survival Proportion Among Women



# Is Age a Good Predictor?

```
summary(titanic$Age)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0.42	20.12	28.00	29.70	38.00	80.00	177

## ■ How about by survival?

```
summary(titanic[titanic$Survived  
=="Dead",]$Age)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
1.00	21.00	28.00	30.63	39.00	74.00	125

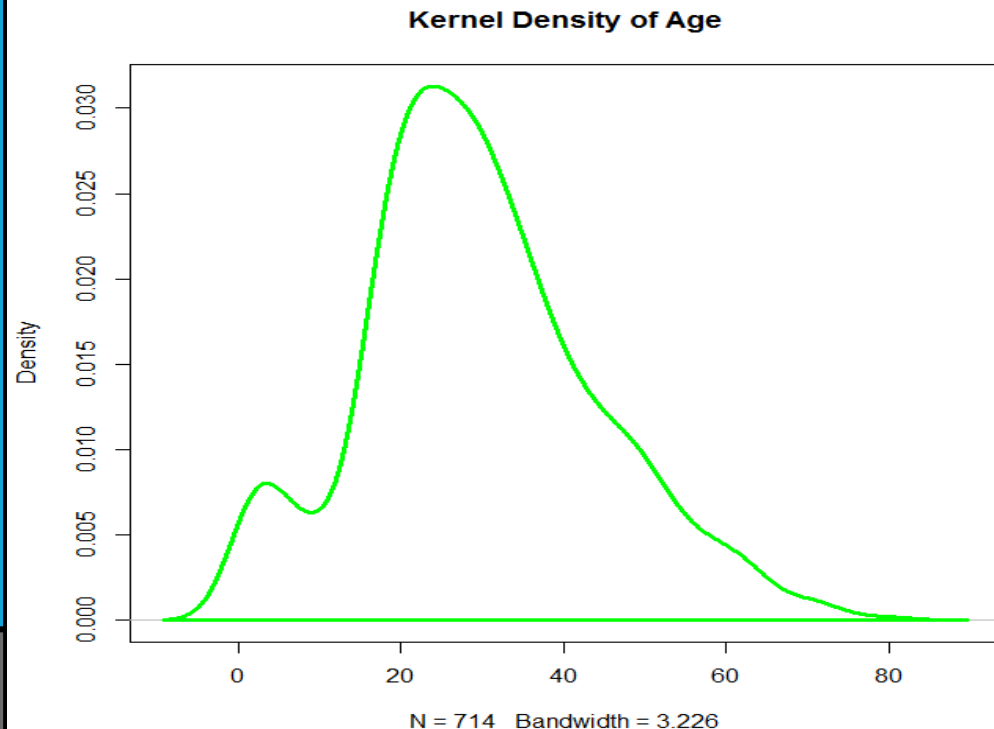
```
summary(titanic[titanic$Survived  
=="Survived",]$Age)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0.42	19.00	28.00	28.34	36.00	80.00	52

# In-class Exercise 3

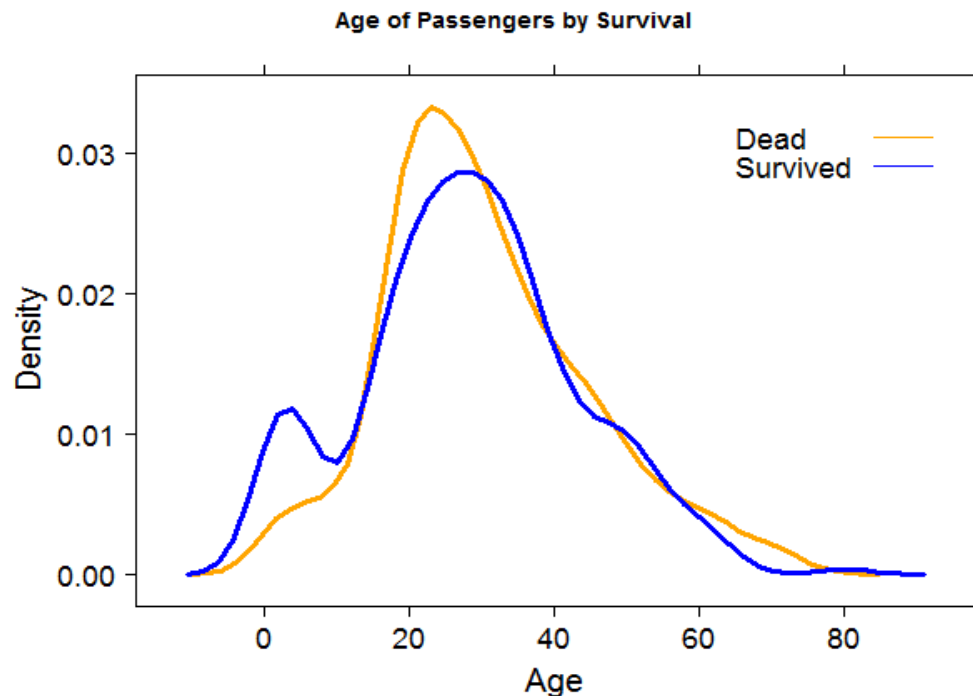
- Create 2 boxplots of Age
  - Segmented by Gender
  - Segmented by Survived
- Create a density plot of Age
  - `na.omit()` may be useful

# Sample solution



```
density(titanic$Age)
#NAs prevent this
> d <-
density(na.omit(titanic$Age))
> plot(d, main="Kernel
Density of Age")
>
polygon(d,border="green",l
wd=3)
```

# Is Age a good predictor for Survival?

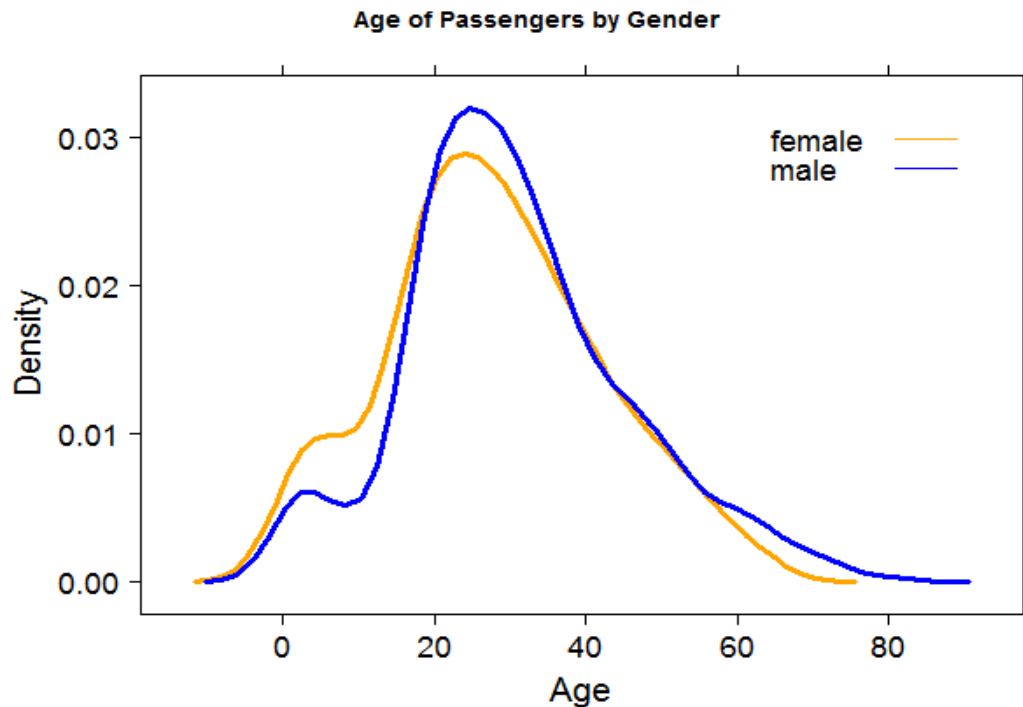


```
densityplot(~ Age,  
data=titanic,  
groups=Survived,  
plot.points=F, lwd=3)
```

Note: will break with missing values



# Is Age a good predictor for Gender?



```
densityplot(~ Age,  
data=titanic, groups=Sex,  
plot.points=F, lwd=3)
```

# Questions?