

# Data Exploration and Visualization

# Agenda

- High Level Process
- Graphing in R: Core and Lattice
- ggplot2 introduction
- Extended Titanic Exploration
- Visualization in Azure

# Agenda

- **High Level Process**
- Graphing in R: Core and Lattice
- ggplot2 introduction
- Extended Titanic Exploration
- Visualization in Azure

# Learning Guidance

- Focus on the ideas rather than exact syntax
- R help is your friend
- I will share output samples on the slides often
- Sample code + slides: Data Exploration and Visualization folder

# Data > Algorithm

- More data == better performance
  - Even with a simple algorithm
- But there are caveats
  - Diminishing returns
  - High quality and high variety needed
  - Algorithm must be appropriate
  - Feature selection/engineering

# Understand Your Data!

- Not spending time on data is a common source of all sorts of problems!
- Are any features correlated with each other?
- What is the scale of your numeric features?
- Are there clear groups of data objects?
- Do you have missing values and where?

And More!

# Agenda

- High Level Process
- **Graphing in R: Core and Lattice**
- ggplot2 introduction
- Extended Titanic Exploration
- Visualization in Azure

# Sidebar: Formula data type

- Built in R type
- Defines relationships between named columns

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

```
f.1 <- Species ~ .
f.2 <- ~ Species + Petal.Width
f.3 <- ~ Petal.Length | Species
```

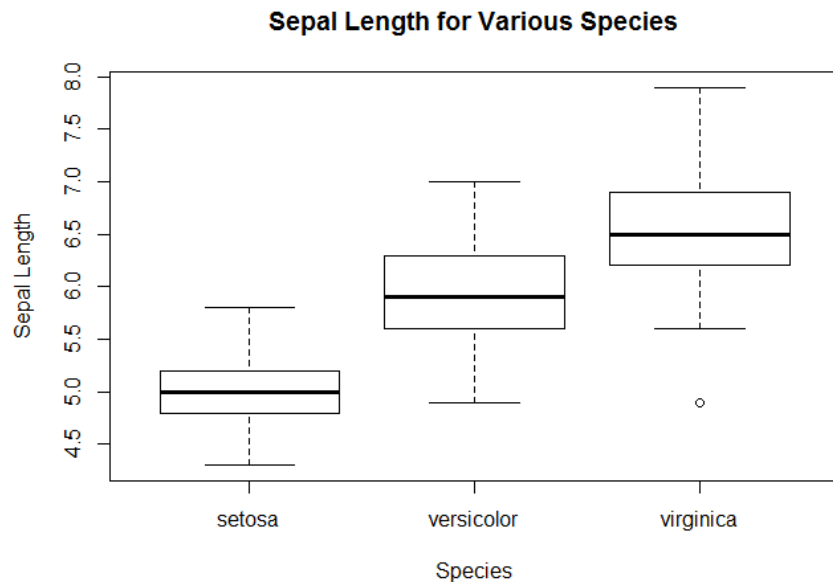
```
class(f.1)
```



# Core: Box

- Distribution of single feature
- Can partition by target class

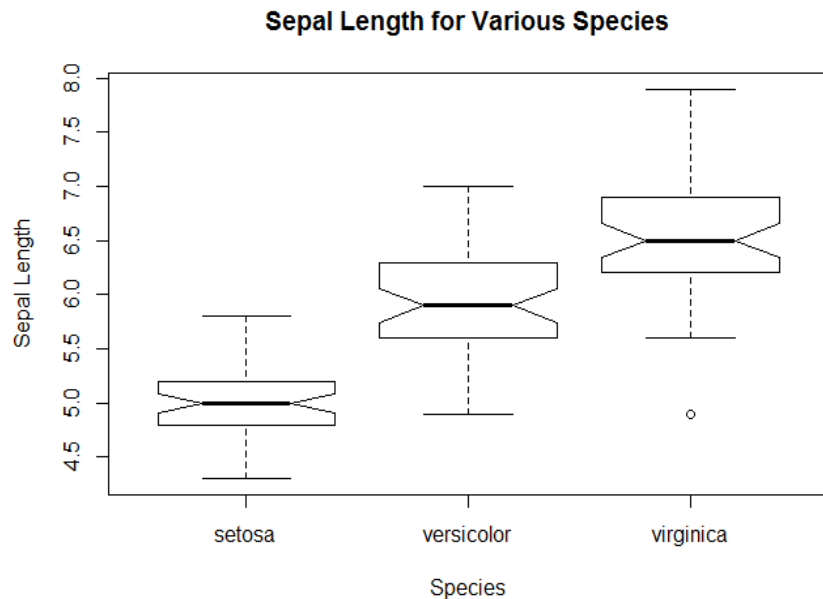
```
boxplot(Sepal.Length ~ Species,  
data=iris,  
main="Sepal Length for Various  
Species",  
xlab="Species",  
ylab="Sepal Length"  
)
```



# Core: Box with Notches

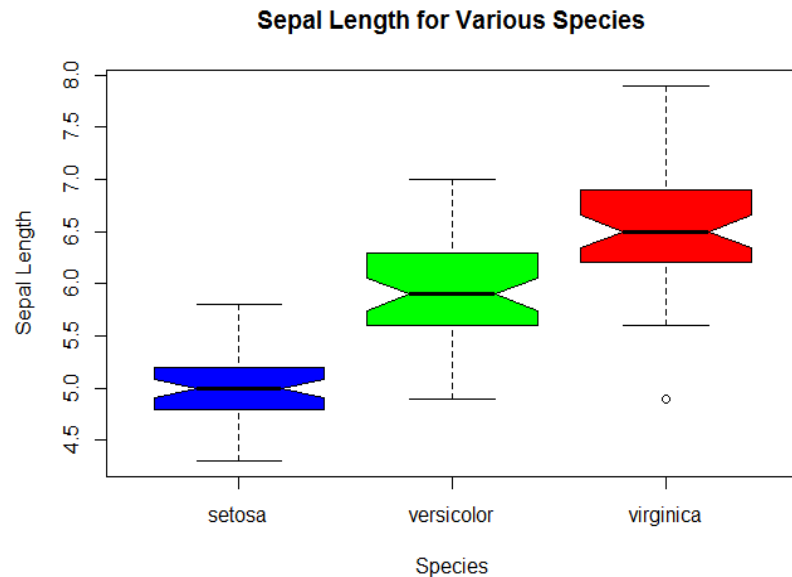
- Estimate of confidence interval of the median
- Notch overlap indicates confidence that median is different

```
boxplot(Sepal.Length ~ Species,  
data=iris,  
main="Sepal Length for Various  
Species",  
xlab="Species",  
ylab="Sepal Length",  
notch=TRUE  
)
```



# Sidebar: Coloring Core Plots

```
boxplot(Sepal.Length ~ Species,  
data=iris,  
main="Sepal Length for Various  
Species",  
xlab="Species",  
ylab="Sepal Length",  
notch=TRUE,  
col=c("blue", "green", "red")  
)
```



# Sidebar: Saving Plots

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

```
pdf("myplot.pdf")
```

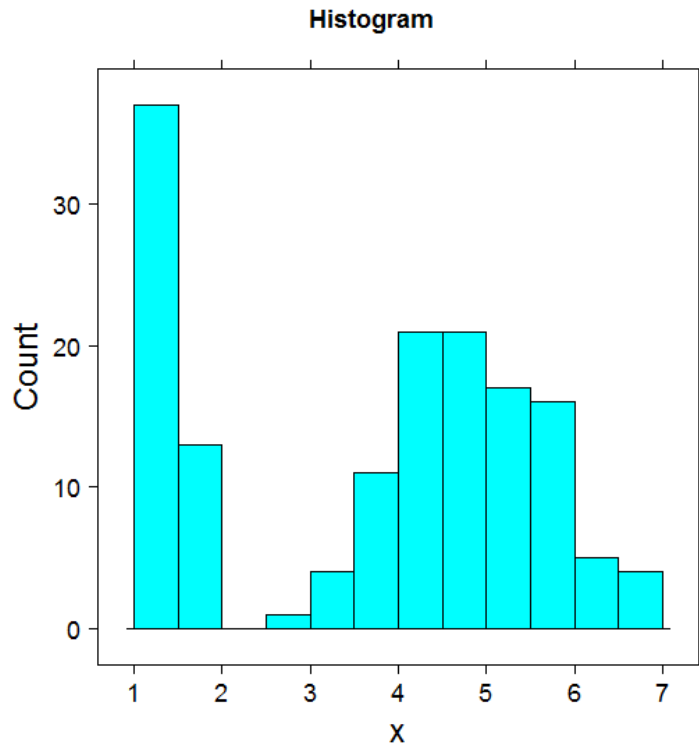
```
boxplot(Sepal.Length ~ Species,  
data=iris,  
main="Sepal Length for Various Species",  
xlab= "Species", ylab="Sepal Length",  
notch=TRUE, col=c("blue", "green", "red")  
)
```

```
dev.off() # Returns plot to the IDE
```

# Lattice: Histogram

- Spread of single feature
- Places values in "bins"
- "breaks" - # of bins

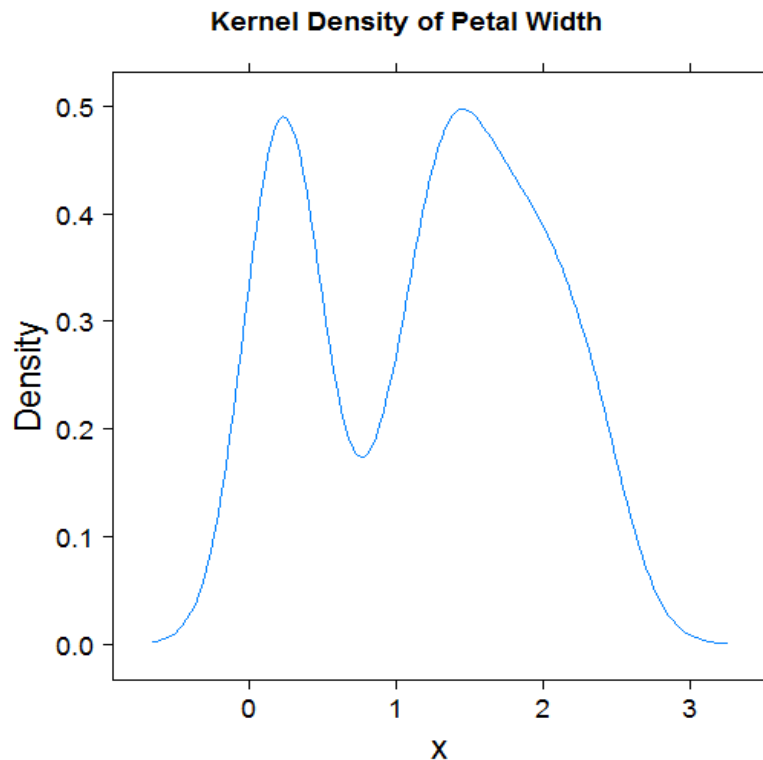
```
histogram(iris$Petal.Length,  
          breaks=10,  
          type="count",  
          main="Histogram")
```



# Lattice: Density plots

- Variant of Histogram
- Scaled for easy comparisons

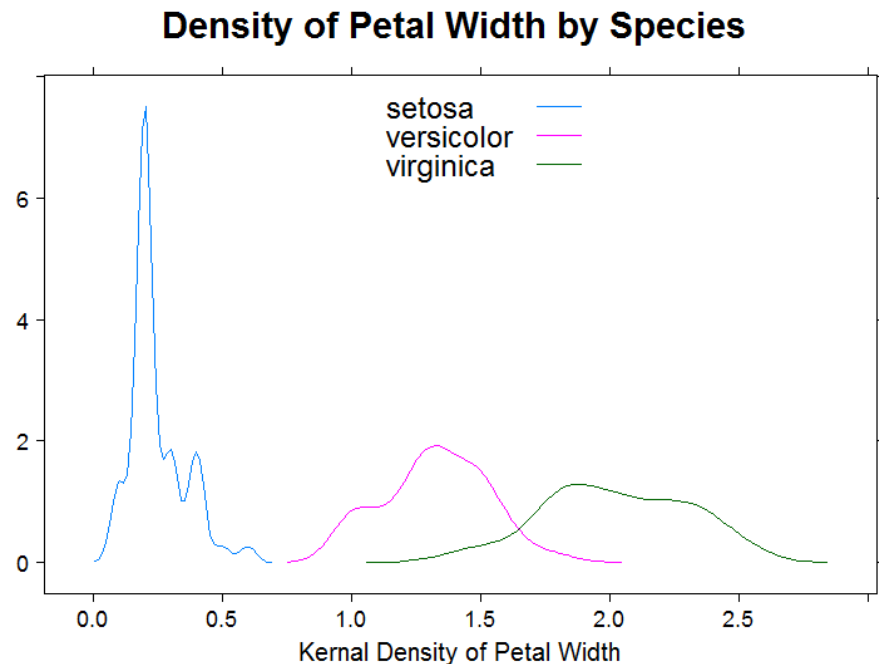
```
densityplot(iris$Sepal.Width,  
main="Kernel Density of Petal  
Width", type="percent",  
n=150)
```



# Lattice: Multiple Density Plots

- Grouping simple
- Styling is not!

```
densityplot(~ Petal.Width,  
data=iris, groups=Species,  
plot.points=F,  
xlab=list(label="Kernel Density  
of Petal Width", fontsize=20),  
ylab="",  
main=list(label="Density of  
Petal Width by Species",  
fontsize=24),  
auto.key=list(corner=c(0,0),  
x=0.4, y=0.8, cex=2),  
scales=list(cex=1.5))
```



# Exercise 1

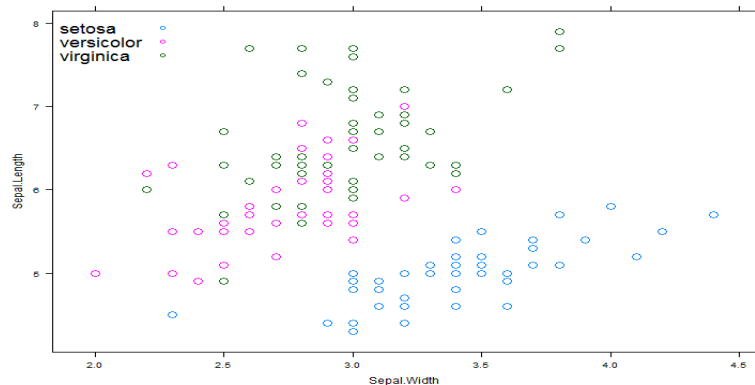
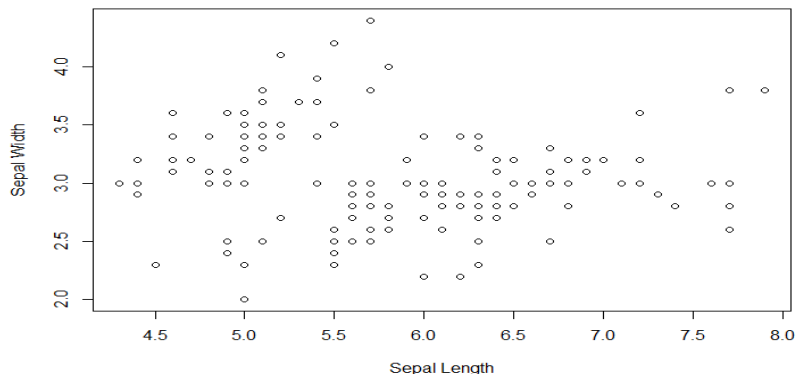
- 2-D Scatter plots – `plot()` and `xyplot()`
  - Sepal Length vs Sepal Width
  - Petal Length vs Petal Width
  - Color based on Species (`lattice`)



# Sample Solution

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

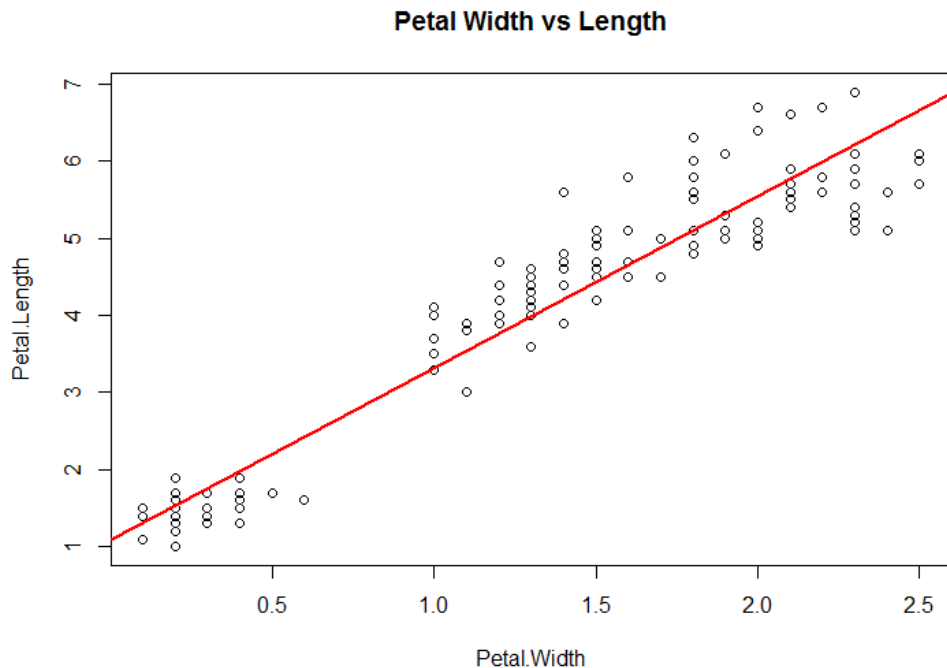
```
# Lattice Graphics
xyplot(Sepal.Length ~
       Sepal.Width, data=iris,
       groups=Species,
       auto.key=list(corner=c(0,0),
                     x=0, y=0.85, cex=1.5), cex=1.5,
       scales=list(cex=1.5))
```



# Core: Extended Scatter Plots

- Add a regression line

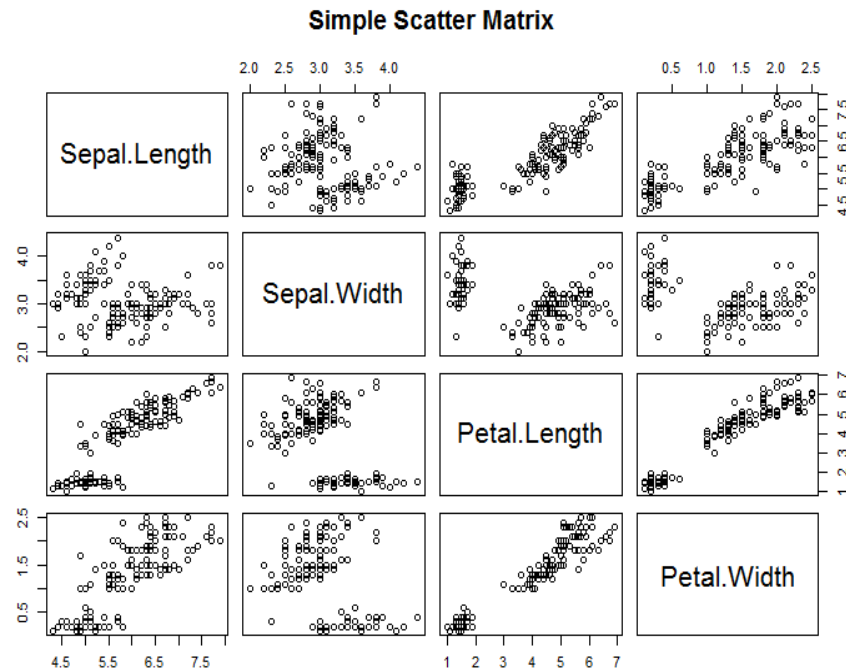
```
plot(Petal.Length ~  
     Petal.Width, data=iris,  
     main="Petal Width vs  
     Length")  
  
abline(lm(Petal.Length ~  
          Petal.Width, data=iris),  
       col="red", lwd=2)
```



# Core: Scatter Plot Matrix

- Multiple relationships on one graph
- Good for initial explorations

```
pairs(~ Sepal.Length +  
      Sepal.Width + Petal.Length +  
      Petal.Width, data=iris,  
      main="Simple Scatter Matrix")
```

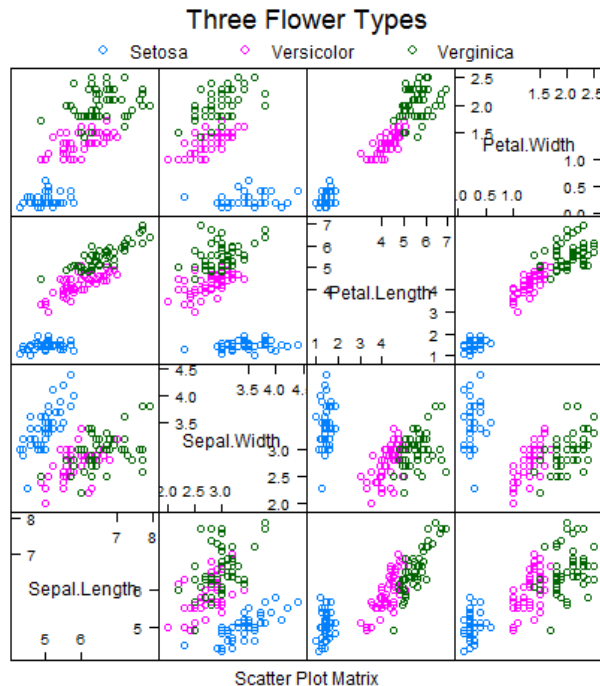


# Lattice: Scatter Plot Matrix

## ■ Simple Grouping

```
# Getting settings for legend
super.sym <-
trellis.par.get("superpose.symbol")

splom(iris[1:4],
      groups=iris$Species,
      panel=panel.superpose,
      key=list(title="Three Flower Types",
              columns=3,
              points=list(pch=super.sym$pch[1:3],
                          col=super.sym$col[1:3])),
      text=list(c("Setosa", "Versicolor", "Verginica"))))
```

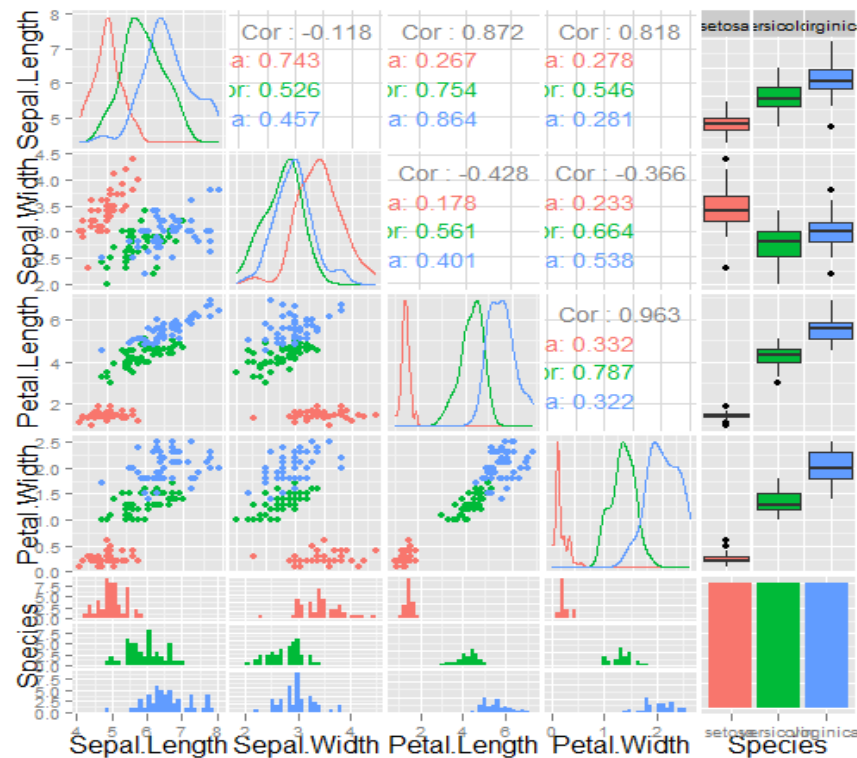


# Enhanced Scatter Plot Matrices

```
library(GGally)

ggpairs(iris, color="Species")
```

- Many packages have custom pairs implementations

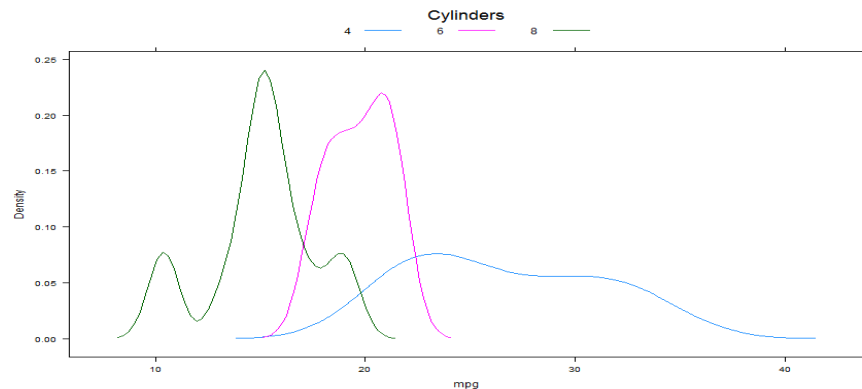


# Exercise 2

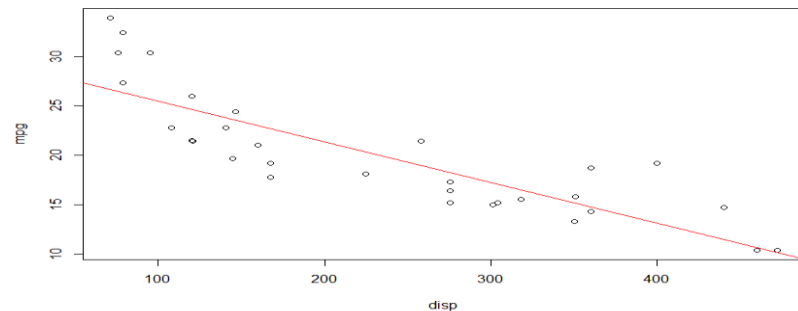
- Load “mtcars” dataset: `data(mtcars)`
  - `?mtcars` for details
- Goal: Predict MPG based on other columns
- Create at least 2 different plots illustrating useful relationships in the data

# Sample Solution 2

```
densityplot( ~ mpg, data=mtcars,  
groups=cyl, plot.points=F,  
auto.key=list(columns=3, title="Cylinders"))
```



```
plot(mpg ~ disp, data=mtcars)  
abline(lm(mpg ~ disp, data=mtcars),  
col="red")
```



# Agenda

- High Level Process
- Graphing in R: Core and Lattice
- **ggplot2 introduction**
- Extended Titanic Exploration
- Visualization in Azure



# Lattice/Core styling is a pain!

- ggplot2 syntax easier to read/write
  - Less intuitive/steeper initial learning curve
  - More power than often needed

# ggplot Basics

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

# Loading

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

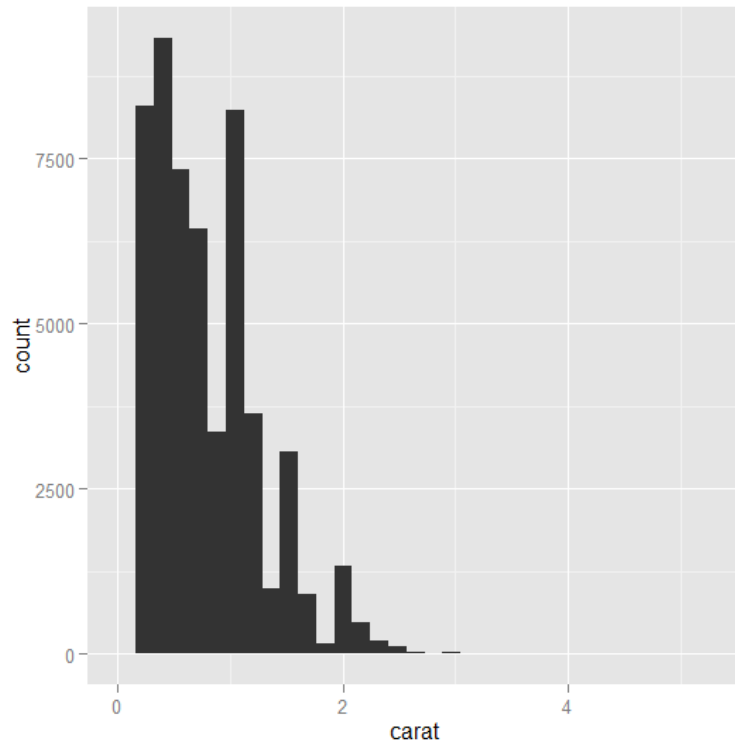
```
> head(diamonds)
```

	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

```
> |
```

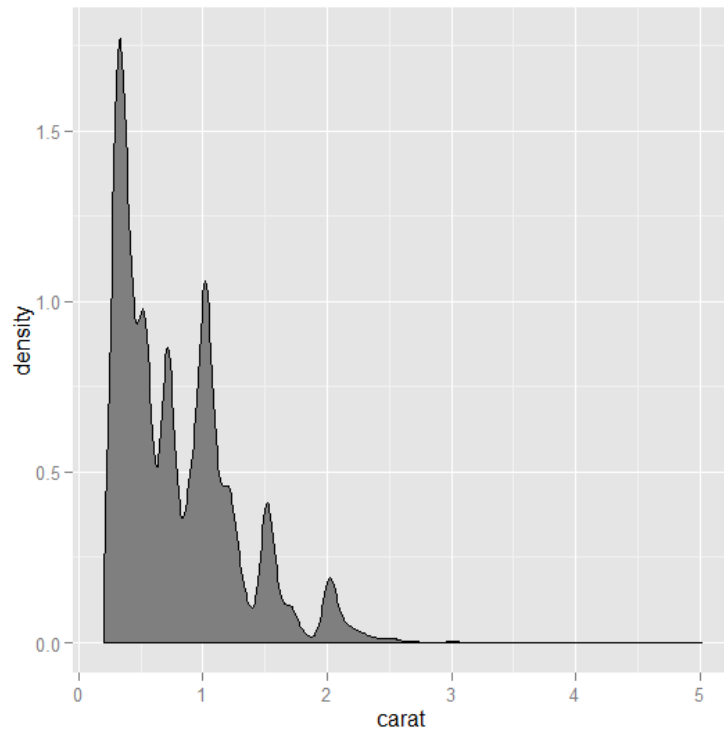
# Histogram

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



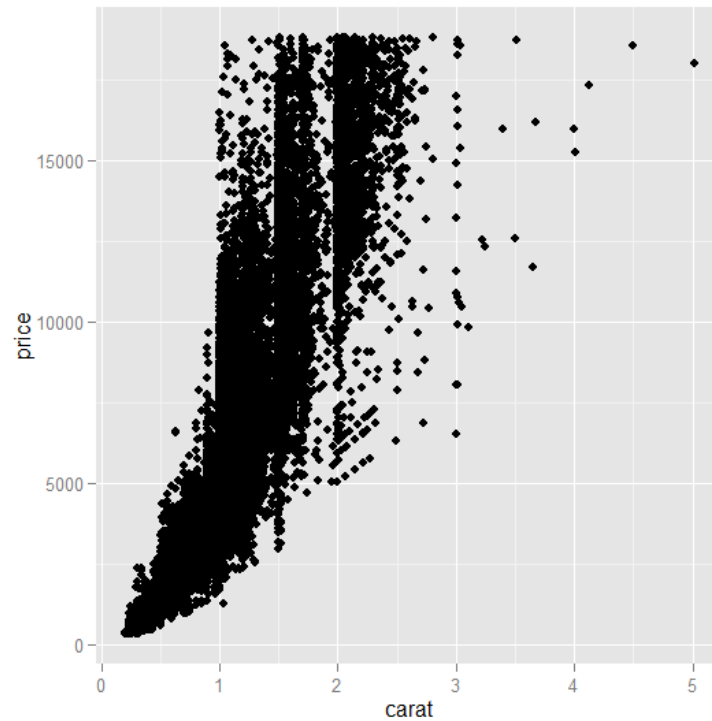
# Density plot

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



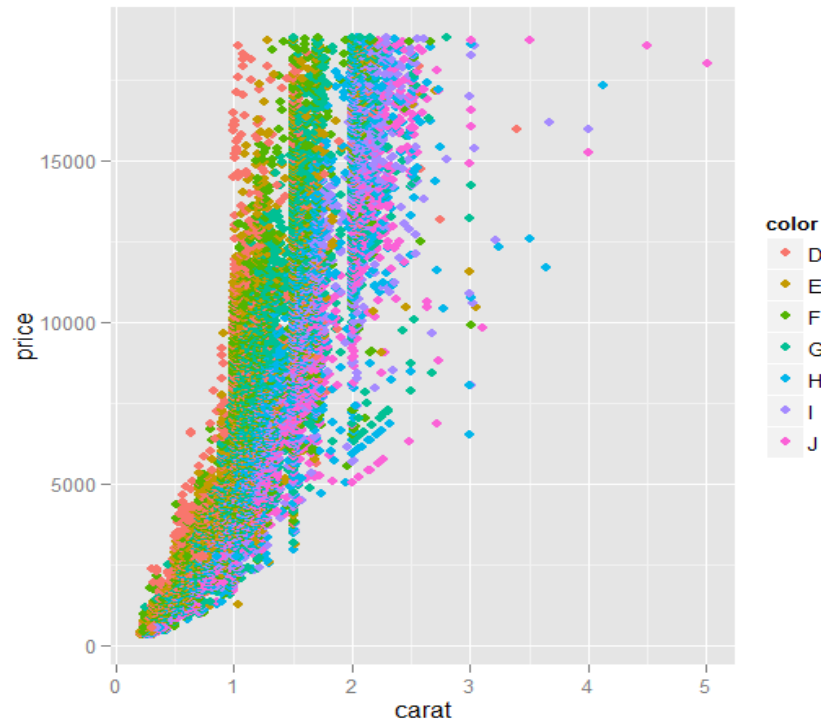
# Scatter plots

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



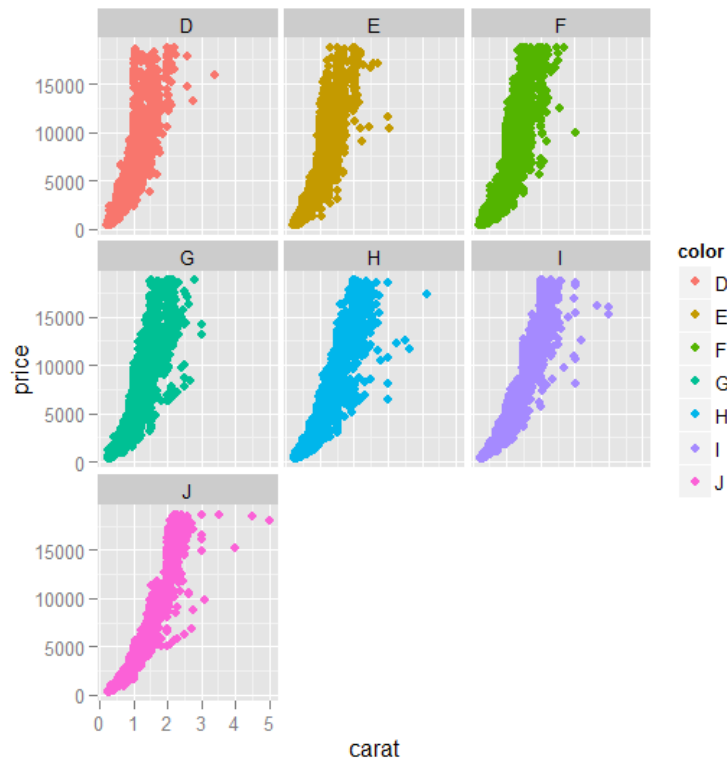
# ggplot Object

```
# 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))
```



# Segment by factor attribute

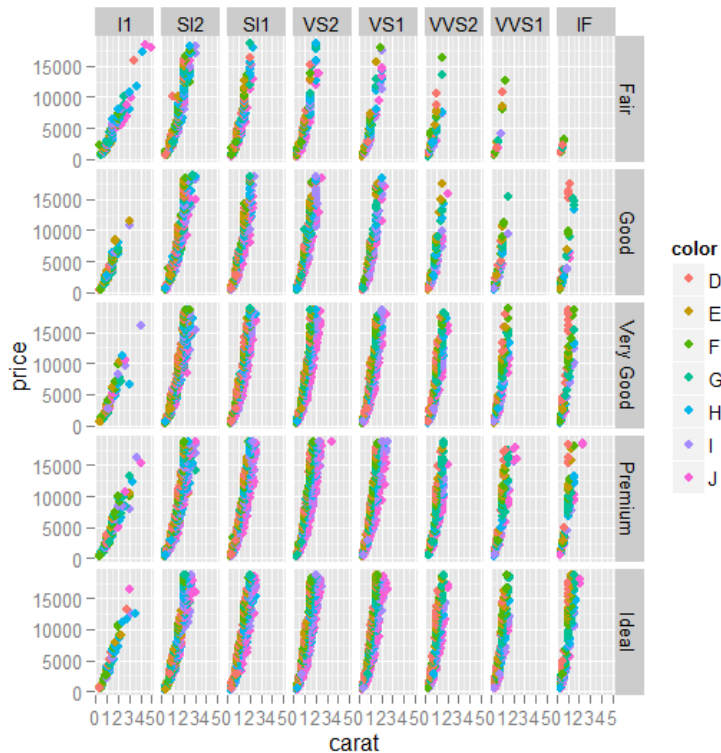
```
g + geom_point(aes(color=color)) +  
facet_wrap(~ color)
```





# More Segments!

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



# Practicing ggplot

- Documentation at <http://docs.ggplot2.org/current/>
- Lists all the different geom\_\* and other functions, with what aes() settings they use

# Agenda

- High Level Process
- Graphing in R: Core and Lattice
- ggplot2 introduction
- **Extended Titanic Exploration**
- Visualization in Azure

# Finding the Code

- 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 & Human Readability

Set target column as a factor

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

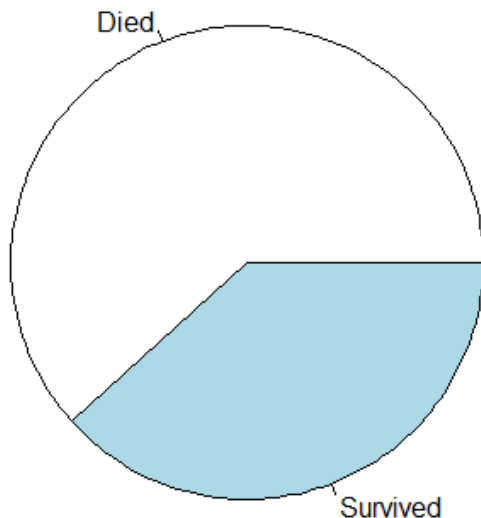
Rename factors and columns

```
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), oma=c(0, 0, 0, 0),
    cex=1.5)
pie(survivedTable, labels=c("Died", "Survived"))
```



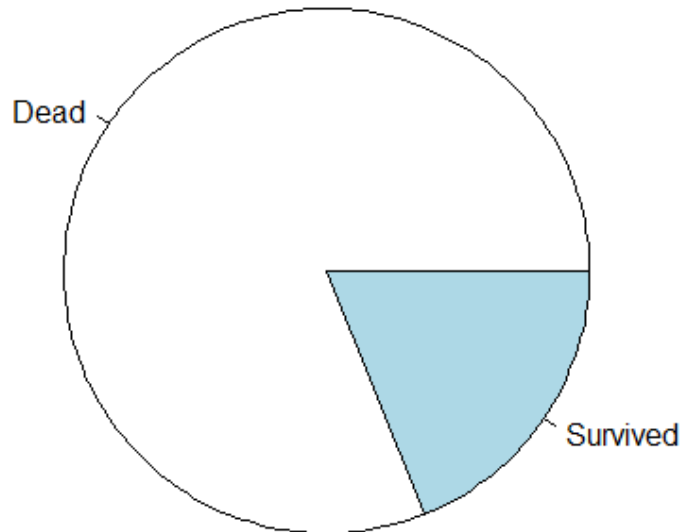


# Is Sex a Good predictor?

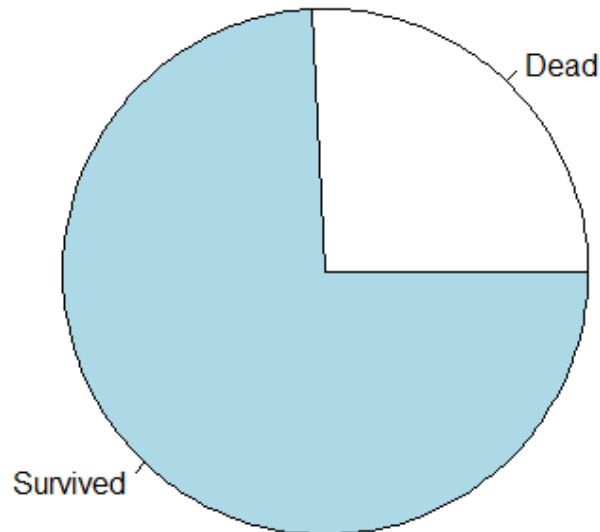
```
male <- titanic[titanic$Sex=="male",]  
female <- titanic[titanic$Sex=="female",]  
par(mfrow=c(1,2))  
pie(table(male$Survived), labels=c("Dead","Survived"),  
main="Survival Portion of Men")  
pie(table(female$Survive), labels=c("Dead","Survived"),  
main="Survival Portion of Women")
```

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

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

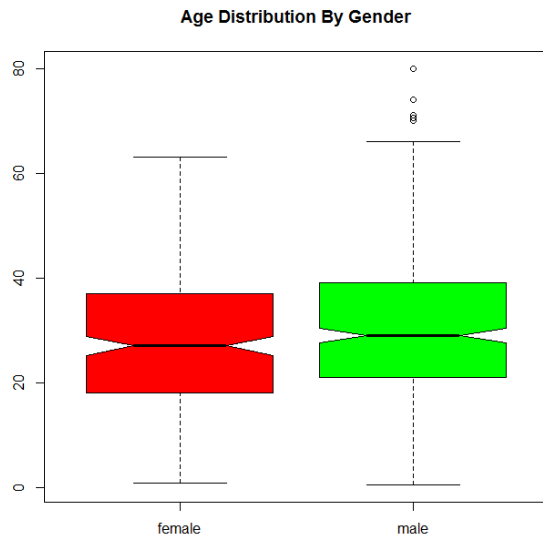
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
0.42	19.00	28.00	28.34	36.00	80.00	52

# Exercise 3

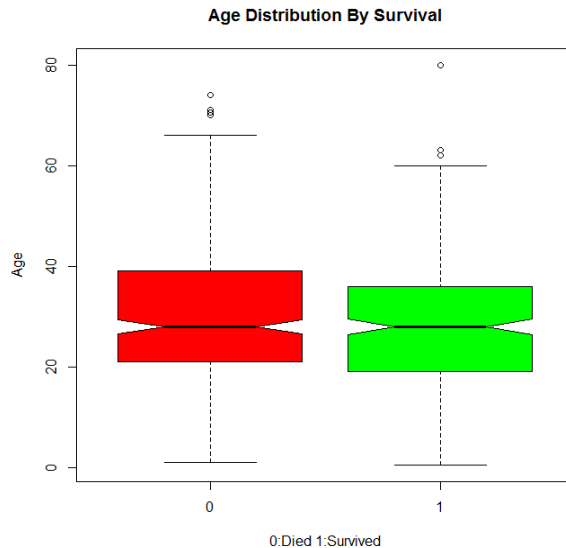
- Create 2 box plots of Age
  - Segmented by Gender
  - Segmented by Survived
- Create a histogram of Age
- Create a density plot of Age
  - `na.omit()` may be useful

# Sample Solution 3

```
boxplot(Age ~ Sex, data=titanic,  
main="Age Distribution By Gender",  
col=c("red","green"), notch=T)
```

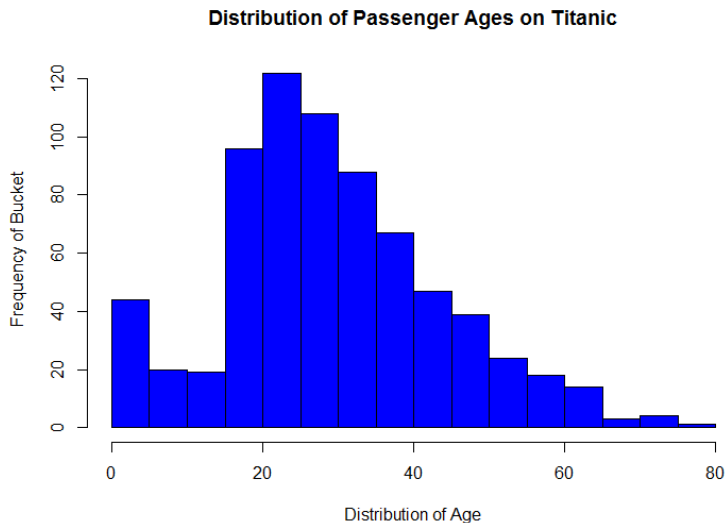


```
boxplot(Age ~ Survived, data=titanic,  
main="Age Distribution By Survival",  
col=c("red","green"), notch=T, ylab="Age")
```

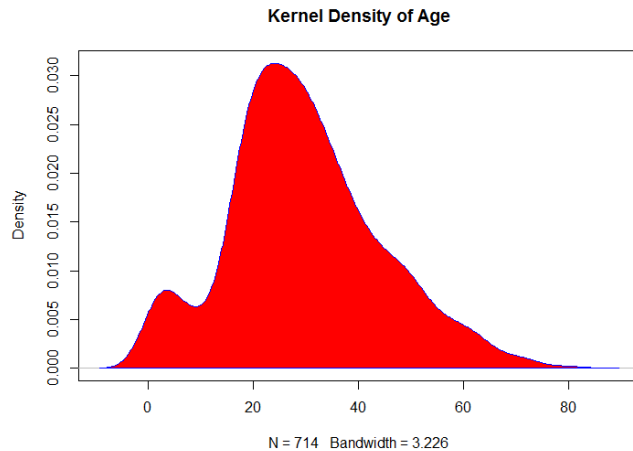


# Sample Solution 3

```
hist(titanic$Age, col="blue", breaks=12,  
xlab="Distribution of Age", ylab="Frequency of  
Bucket", main="Distribution of Passenger Ages  
on Titanic")
```

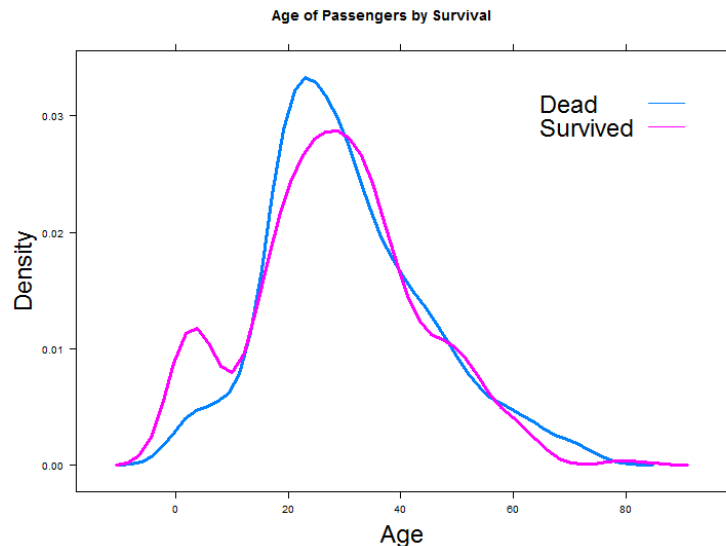


```
density(titanic$Age) #NAs prevent this  
d <- density(na.omit(titanic$Age))  
plot(d, main="Kernel Density of Age")  
polygon(d, col="red", border="blue")
```

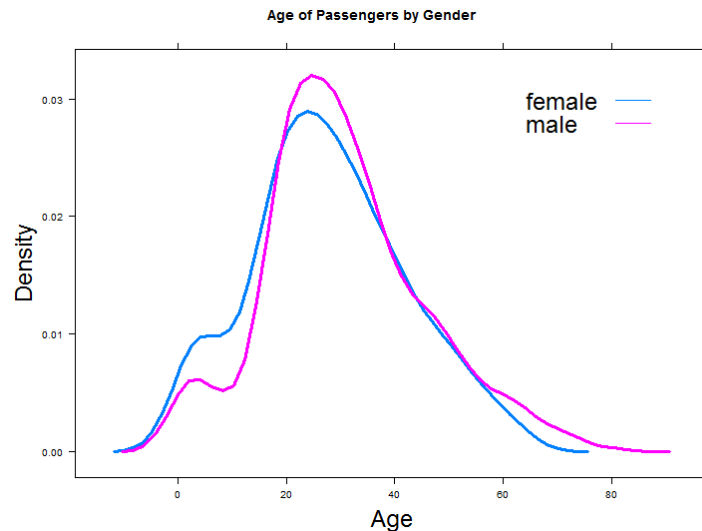


# Is Age a Good Predictor?

```
densityplot(~ Age, data=titanic,  
groups=Sex, plot.points=F, lwd=3,  
auto.key=list(corner=c(0,0), x=0.7, y=0.8))
```



```
densityplot(~ Age, data=titanic,  
groups=Survived, plot.points=F, lwd=3,  
auto.key=list(corner=c(0,0), x=0.7, y=0.8))
```



# Exercise 4

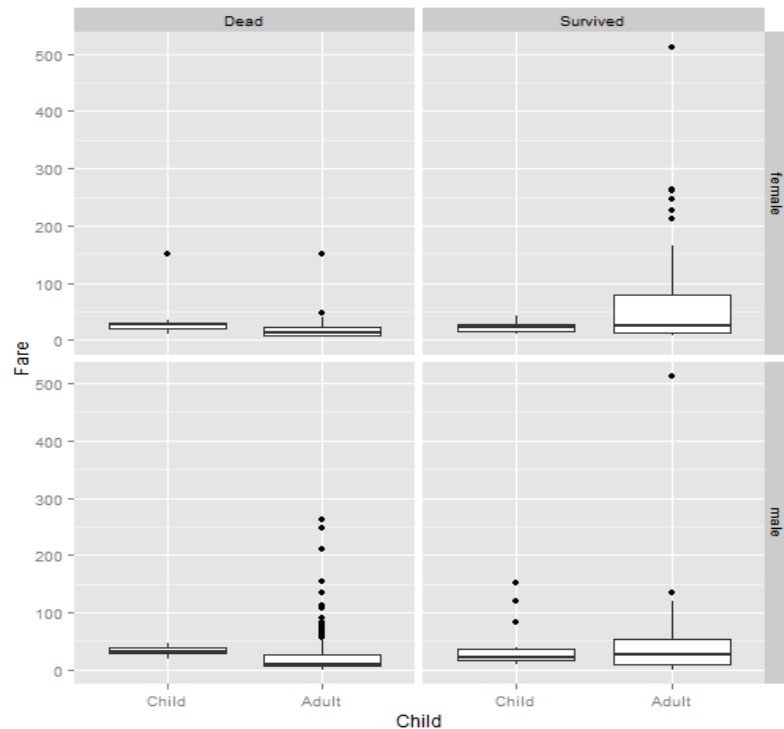
- Create a new column "Child"
  - Assign each row "Adult" or "Child" based on a consistent metric
- Use ggplot to create a series of box plots relating Fare, Child, Sex, and Survived



# Sample Solution 4

```
child <- titanic$Age
child[child < 13] <- 0
child[child >= 13] <- 1
titanic$Child <- as.factor(child)
levels(titanic$Child)
levels(titanic$Child) <- c("Child", "Adult")

g <- ggplot(data=titanic[!is.na(titanic$Child),],
            aes(x=Child, y=Fare))
g.b <- g + geom_boxplot()
g.b + facet_grid(Sex ~ Survived)
```



# Agenda

- High Level Process
- Graphing in R: Core and Lattice
- ggplot2 introduction
- Extended Titanic Exploration
- **Visualization in Azure**