# Exploring WHO data

# Unit 7 - Lecture 2, Predictive Policing

# Unit 7 - Lecture 1

# VIDEO 4 - A BASIC SCATTERPLOT

# Read in data

WHO = read.csv("WHO.csv")

str(WHO)

# Plot from Week 1

plot(WHO$GNI, WHO$FertilityRate)

# Let's redo this using ggplot

# Install and load the ggplot2 library:

install.packages("ggplot2")

library(ggplot2)

# Create the ggplot object with the data and the aesthetic mapping:

scatterplot = ggplot(WHO, aes(x = GNI, y = FertilityRate))

# Add the geom\_point geometry

scatterplot + geom\_point()

# Make a line graph instead:

scatterplot + geom\_line()

# Switch back to our points:

scatterplot + geom\_point()

# Redo the plot with blue triangles instead of circles:

scatterplot + geom\_point(color = "blue", size = 3, shape = 17)

# Another option:

scatterplot + geom\_point(color = "darkred", size = 3, shape = 8)

# Add a title to the plot:

scatterplot + geom\_point(colour = "blue", size = 3, shape = 17) + ggtitle("Fertility Rate vs. Gross National Income")

# Save our plot:

fertilityGNIplot = scatterplot + geom\_point(colour = "blue", size = 3, shape = 17) + ggtitle("Fertility Rate vs. Gross National Income")

pdf("MyPlot.pdf")

print(fertilityGNIplot)

dev.off()

# VIDEO 5 - MORE ADVANCED SCATTERPLOTS

# Color the points by region:

ggplot(WHO, aes(x = GNI, y = FertilityRate, color = Region)) + geom\_point()

# Color the points according to life expectancy:

ggplot(WHO, aes(x = GNI, y = FertilityRate, color = LifeExpectancy)) + geom\_point()

# Is the fertility rate of a country was a good predictor of the percentage of the population under 15?

ggplot(WHO, aes(x = FertilityRate, y = Under15)) + geom\_point()

# Let's try a log transformation:

ggplot(WHO, aes(x = log(FertilityRate), y = Under15)) + geom\_point()

# Simple linear regression model to predict the percentage of the population under 15, using the log of the fertility rate:

mod = lm(Under15 ~ log(FertilityRate), data = WHO)

summary(mod)

# Add this regression line to our plot:

ggplot(WHO, aes(x = log(FertilityRate), y = Under15)) + geom\_point() + stat\_smooth(method = "lm")

# 99% confidence interval

ggplot(WHO, aes(x = log(FertilityRate), y = Under15)) + geom\_point() + stat\_smooth(method = "lm", level = 0.99)

# No confidence interval in the plot

ggplot(WHO, aes(x = log(FertilityRate), y = Under15)) + geom\_point() + stat\_smooth(method = "lm", se = FALSE)

# Change the color of the regression line:

ggplot(WHO, aes(x = log(FertilityRate), y = Under15)) + geom\_point() + stat\_smooth(method = "lm", colour = "orange")

# Unit 7 - Lecture 2, Predictive Policing

# VIDEO 3 - A Basic Line Plot

# Load our data:

mvt = read.csv("mvt.csv", stringsAsFactors=FALSE)

str(mvt)

# Convert the Date variable to a format that R will recognize:

mvt$Date = strptime(mvt$Date, format="%m/%d/%y %H:%M")

# Extract the hour and the day of the week:

mvt$Weekday = weekdays(mvt$Date)

mvt$Hour = mvt$Date$hour

# Let's take a look at the structure of our data again:

str(mvt)

# Create a simple line plot - need the total number of crimes on each day of the week. We can get this information by creating a table:

table(mvt$Weekday)

# Save this table as a data frame:

WeekdayCounts = as.data.frame(table(mvt$Weekday))

str(WeekdayCounts)

# Load the ggplot2 library:

library(ggplot2)

# Create our plot

ggplot(WeekdayCounts, aes(x=Var1, y=Freq)) + geom\_line(aes(group=1))

# Make the "Var1" variable an ORDERED factor variable

WeekdayCounts$Var1 = factor(WeekdayCounts$Var1, ordered=TRUE, levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday","Saturday"))

# Try again:

ggplot(WeekdayCounts, aes(x=Var1, y=Freq)) + geom\_line(aes(group=1))

# Change our x and y labels:

ggplot(WeekdayCounts, aes(x=Var1, y=Freq)) + geom\_line(aes(group=1)) + xlab("Day of the Week") + ylab("Total Motor Vehicle Thefts")

# VIDEO 4 - Adding the Hour of the Day

# Create a counts table for the weekday and hour:

table(mvt$Weekday, mvt$Hour)

# Save this to a data frame:

DayHourCounts = as.data.frame(table(mvt$Weekday, mvt$Hour))

str(DayHourCounts)

# Convert the second variable, Var2, to numbers and call it Hour:

DayHourCounts$Hour = as.numeric(as.character(DayHourCounts$Var2))

# Create out plot:

ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom\_line(aes(group=Var1))

# Change the colors

ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom\_line(aes(group=Var1, color=Var1), size=2)

# Separate the weekends from the weekdays:

DayHourCounts$Type = ifelse((DayHourCounts$Var1 == "Sunday") | (DayHourCounts$Var1 == "Saturday"), "Weekend", "Weekday")

# Redo our plot, this time coloring by Type:

ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom\_line(aes(group=Var1, color=Type), size=2)

# Make the lines a little transparent:

ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom\_line(aes(group=Var1, color=Type), size=2, alpha=0.5)

# Fix the order of the days:

DayHourCounts$Var1 = factor(DayHourCounts$Var1, ordered=TRUE, levels=c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"))

# Make a heatmap:

ggplot(DayHourCounts, aes(x = Hour, y = Var1)) + geom\_tile(aes(fill = Freq))

# Change the label on the legend, and get rid of the y-label:

ggplot(DayHourCounts, aes(x = Hour, y = Var1)) + geom\_tile(aes(fill = Freq)) + scale\_fill\_gradient(name="Total MV Thefts") + theme(axis.title.y = element\_blank())

# Change the color scheme

ggplot(DayHourCounts, aes(x = Hour, y = Var1)) + geom\_tile(aes(fill = Freq)) + scale\_fill\_gradient(name="Total MV Thefts", low="white", high="red") + theme(axis.title.y = element\_blank())

# VIDEO 5 - Maps

# Install and load two new packages:

install.packages("maps")

install.packages("ggmap")

library(maps)

library(ggmap)

# Load a map of Chicago into R:

chicago = get\_map(location = "chicago", zoom = 11)

# Look at the map

ggmap(chicago)

# Plot the first 100 motor vehicle thefts:

ggmap(chicago) + geom\_point(data = mvt[1:100,], aes(x = Longitude, y = Latitude))

# Round our latitude and longitude to 2 digits of accuracy, and create a crime counts data frame for each area:

LatLonCounts = as.data.frame(table(round(mvt$Longitude,2), round(mvt$Latitude,2)))

str(LatLonCounts)

# Convert our Longitude and Latitude variable to numbers:

LatLonCounts$Long = as.numeric(as.character(LatLonCounts$Var1))

LatLonCounts$Lat = as.numeric(as.character(LatLonCounts$Var2))

# Plot these points on our map:

ggmap(chicago) + geom\_point(data = LatLonCounts, aes(x = Long, y = Lat, color = Freq, size=Freq))

# Change the color scheme:

ggmap(chicago) + geom\_point(data = LatLonCounts, aes(x = Long, y = Lat, color = Freq, size=Freq)) + scale\_colour\_gradient(low="yellow", high="red")

# We can also use the geom\_tile geometry

ggmap(chicago) + geom\_tile(data = LatLonCounts, aes(x = Long, y = Lat, alpha = Freq), fill="red")

# VIDEO 6 - Geographical Map on US

# Load our data:

murders = read.csv("murders.csv")

str(murders)

# Load the map of the US

statesMap = map\_data("state")

str(statesMap)

# Plot the map:

ggplot(statesMap, aes(x = long, y = lat, group = group)) + geom\_polygon(fill = "white", color = "black")

# Create a new variable called region with the lowercase names to match the statesMap:

murders$region = tolower(murders$State)

# Join the statesMap data and the murders data into one dataframe:

murderMap = merge(statesMap, murders, by="region")

str(murderMap)

# Plot the number of murder on our map of the United States:

ggplot(murderMap, aes(x = long, y = lat, group = group, fill = Murders)) + geom\_polygon(color = "black") + scale\_fill\_gradient(low = "black", high = "red", guide = "legend")

# Plot a map of the population:

ggplot(murderMap, aes(x = long, y = lat, group = group, fill = Population)) + geom\_polygon(color = "black") + scale\_fill\_gradient(low = "black", high = "red", guide = "legend")

# Create a new variable that is the number of murders per 100,000 population:

murderMap$MurderRate = murderMap$Murders / murderMap$Population \* 100000

# Redo our plot with murder rate:

ggplot(murderMap, aes(x = long, y = lat, group = group, fill = MurderRate)) + geom\_polygon(color = "black") + scale\_fill\_gradient(low = "black", high = "red", guide = "legend")

# Redo the plot, removing any states with murder rates above 10:

ggplot(murderMap, aes(x = long, y = lat, group = group, fill = MurderRate)) + geom\_polygon(color = "black") + scale\_fill\_gradient(low = "black", high = "red", guide = "legend", limits = c(0,10))

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

# Unit 7 - Recitation

# VIDEO 3 - Bar Charts

# Load ggplot library

library(ggplot2)

# Load our data, which lives in intl.csv

intl = read.csv("intl.csv")

str(intl)

# We want to make a bar plot with region on the X axis

# and Percentage on the y-axis.

ggplot(intl, aes(x=Region, y=PercentOfIntl)) + geom\_bar(stat="identity") + geom\_text(aes(label=PercentOfIntl))

# Make Region an ordered factor

# We can do this with the re-order command and transform command.

intl = transform(intl, Region = reorder(Region, -PercentOfIntl))

# Look at the structure

str(intl)

# Make the percentages out of 100 instead of fractions

intl$PercentOfIntl = intl$PercentOfIntl \* 100

# Make the plot

ggplot(intl, aes(x=Region, y=PercentOfIntl)) +geom\_bar(stat="identity", fill="dark blue") +geom\_text(aes(label=PercentOfIntl), vjust=-0.4) +ylab("Percent of International Students") +

theme(axis.title.x = element\_blank(), axis.text.x = element\_text(angle = 45, hjust = 1))

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

#working with world map

#\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#

# VIDEO 5 - World map

# Load the ggmap package

library(ggmap)

# Load in the international student data

intlall = read.csv("intlall.csv",stringsAsFactors=FALSE)

# Lets look at the first few rows

head(intlall)

# Those NAs are really 0s, and we can replace them easily

intlall[is.na(intlall)] = 0

# Now lets look again

head(intlall)

# Load the world map

world\_map = map\_data("world")

str(world\_map)

# Lets merge intlall into world\_map using the merge command

world\_map = merge(world\_map, intlall, by.x ="region", by.y = "Citizenship")

str(world\_map)

# Plot the map

ggplot(world\_map, aes(x=long, y=lat, group=group)) + geom\_polygon(fill="white", color="black") + coord\_map("mercator")

# Reorder the data

world\_map = world\_map[order(world\_map$group, world\_map$order),]

# Redo the plot

ggplot(world\_map, aes(x=long, y=lat, group=group)) + geom\_polygon(fill="white", color="black") + coord\_map("mercator")

# Lets look for China

table(intlall$Citizenship)

# Lets "fix" that in the intlall dataset

intlall$Citizenship[intlall$Citizenship=="China (People's Republic Of)"] = "China"

# We'll repeat our merge and order from before

world\_map = merge(map\_data("world"), intlall,

by.x ="region",

by.y = "Citizenship")

world\_map = world\_map[order(world\_map$group, world\_map$order),]

ggplot(world\_map, aes(x=long, y=lat, group=group)) +

geom\_polygon(aes(fill=Total), color="black") +

coord\_map("mercator")

# We can try other projections - this one is visually interesting

ggplot(world\_map, aes(x=long, y=lat, group=group)) +

geom\_polygon(aes(fill=Total), color="black") +

coord\_map("ortho", orientation=c(20, 30, 0))

ggplot(world\_map, aes(x=long, y=lat, group=group)) +

geom\_polygon(aes(fill=Total), color="black") +

coord\_map("ortho", orientation=c(-37, 175, 0))

# VIDEO 7 - Line Charts

# First, lets make sure we have ggplot2 loaded

library(ggplot2)

# Now lets load our dataframe

households = read.csv("households.csv")

str(households)

# Load reshape2

library(reshape2)

# Lets look at the first two columns of our households dataframe

households[,1:2]

# First few rows of our melted households dataframe

head(melt(households, id="Year"))

households[,1:3]

melt(households, id="Year")[1:10,3]

melt(households, id="Year")[1:10,]

# Plot it

ggplot(melt(households, id="Year"),

aes(x=Year, y=value, color=variable)) +

geom\_line(size=2) + geom\_point(size=5) +

ylab("Percentage of Households")