Data Cleansing

As many of you have heard the expression before, “put good in, get good out,” this phrase holds true for data science. Data collection and cleansing is extremely important for having accurate, accountable data that will explain trends or variances in the real world. For instance, if you are collecting data for many companies for many years and are missing data, are you going to keep that data that is missing variables or are you going to delete it?

Data cleansing is one of the most important pieces of data science. According to Forbes, data cleansing and organizing consists of 60% of the time spent by data scientists. Another 19% of the total time spent by data scientists consists of collecting data sets.In addition these two tasks were voted as the least liked tasks by data scientists at 57% and 21% respectively. (https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#690b6416f637).

Data Cleansing Steps

WHO data on obesity rates
http://apps.who.int/gho/data/view.main.CTRY2450A?lang=en
PPP International $
Explain why data is good or not

GDP does not account for some industries
Obesity could be due to other factors(race, region)

Datathon example with pharmaceutical companies

-Show different types of downloads similarity
-Download as csv
-Delete unnecessary rows from the top (2 and 3)
-Find & Select under home tab
    -Find All “male” (this will include female as well)
    -Close
    -Delete sheet columns

-Delete both sexes row (now a constant)
-Open new sheet
    =LEFT function
        =LEFT(‘xmart’!B3,4)
    -Insert new row at top
        -copy and paste in years
    -insert new row at side
        -copy and paste countries
    -Alternatively can do the previous two steps first or add insert a step down
-Show how find and replace does not select ‘No d’

-Save file as excel workbook

-Make sure to move the workbook you want displayed to the front

-Right click on workbook you want to move to end

-Do the same for “No d values but change the look function so it looks for values

GGPLOT and Testing
https://www.kaggle.com/uciml/forest-cover-type-dataset/data

```
graphname <- ggplot(data, aes(x variable, y variable)) +
labs(title=”title”) + geom type()
```

Standard histogram

```
graph1<- ggplot(cover_type_forest, aes(Elevation)) + labs(title = "Elevation") +
geom_histogram()
Graph1

graphtest1 <- ggplot(cover_type_forest, aes((Slope))
graphtest1 <- graphtest1 +geom_histogram()
graphtest1

#Trying log on aes(slope) makes weird value



graph2 <- ggplot(cover_type_forest, aes(Elevation, Slope)) + labs(title="Elevation and Slope") +
geom_point()
graph2
```

(Above sucks)

```
install.packages
library(ggthemes)

#Heat map
graphheat <- ggplot(cover_type_forest, aes( Elevation, Slope)) + geom_bin2d(bins=4.5) +
scale_fill_gradient(low = "white", high = "steelblue") + labs(title = "Heatmap Elevation versus
Slope", subtitle = "1 = low, 5 = high") + theme_tufte() + labs(x = "Elevation")+ labs(y = "Slope" )
graphheat

#alpha alters transparency of the data
scatter1<-ggplot(assignment4, aes(gdp_log , PHYSINT))
scatter1<-scatter1 + geom_point(alpha=0.3) + geom_smooth(method = "loess", )
```

```
scatter1

#binhex command
#install hexbin package
#formula is for trend line
#hexagonal heat map
scatter2<-ggplot(assignment4, aes(gdp_log , democ))
scatter2<-scatter2 + stat_bin_hex() + stat_smooth(method = "lm", formula = y ~ poly(x, 2), size
= 1)
scatter2

#density plot
scatter3<-ggplot(assignment4, aes(PHYSINT , democ))
scatter3<-scatter3 + stat_density_2d() + geom_smooth(method = "glm")
scatter3

#Need scatterplot3d package
# note you can also do 3dbarplotsd
# the arguments are x,y, and z variable.
# The angle changes the way you look at it
# You will need the scatterplot3d package to do this
# angle adjust the angle of the graph. Alter it to look at the graph from different angles
# highlight.3d=TRUE creates the red and black color to tell you where in the graph the dots are

scatterplot3d(assignment4$gdp_log, assignment4$democ, assignment4$PHYSINT, angle=75,
highlight.3d = TRUE, length(1))

linearMod <- lm(dist ~ speed, data=cars)  # build linear regression model on full data
print(linearMod)

Linear regression <- lm(y ~ x1 + x2…., data = my data)

summary
#> Call:
#> lm(formula = dist ~ speed, data = cars)
#>
#> Coefficients:
#> (Intercept)       speed
#>     -17.579       3.932

#Need summary statistics to see if it is a good fit
```

```
summary(linearMod)  # model summary
#> Call:
#> lm(formula = dist ~ speed, data = cars)
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -29.069  -9.525  -2.272   9.215  43.201
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) -17.5791     6.7584  -2.601    0.0123 *
#> speed         3.9324     0.4155   9.464 1.49e-12 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 15.38 on 48 degrees of freedom
#> Multiple R-squared:  0.6511, Adjusted R-squared:  0.6438
#> F-statistic: 89.57 on 1 and 48 DF,  p-value: 1.49e-12
```

Aesthetics	Option	Outcome
Linetype	linetype=1	Solid line (default)
	linetype = 2	Hashed
	linetype = 3	Dotted
	lintetype = 4	Dot and hash
	linetype = 5	Long hash
	linetype = 6	Dot and long hash
Size	size = value	Change size of something
The default size of 0.5 so less than that shrinks something		
Greater than 0.5 makes it bigger		
	size = 0.25	produces line/point/text of 0.25mm
Shape	shape = integer	0 to 25 each value is a different shape
Colour	coulour = "Name"	Replace name with a color like red
Alpha	alpha(colour, value)	changes the transparency of an object

ggplot aesthetics

	Required	options
geom_bar()	X: variable on x axis	colour, size, fill, linetype, weight, alpha
geom_point()	x and y variables	colour, size, fill, linetype, weight, alpha
geom_line()	x and y variable	colour, size, linetype, alpha
geom_smooth()	x and y variable	colour, size, fill, linetype, weight, alpha
geom_histogram()	x variable	colour, size, fill, linetype, weight, alpha
geom_boxplot()	x variable, ymin, ymax lower upper middle	colour, size, fill, weight, alpha
We must specify what all the different parts of the boxplot look like		
geom_text	x: horizontal coordinate y: vertical coordinate label to be printed can be values of variables	colour, size, fill, weight, alpha
geom_density()	x and y variable	colour, size, fill, linetype, weight, alpha
geom$_e$*rrorbar*()	x variable, ymin, ymax lower and upper value	colour, size, linetype, width, alpha
geom$_h$*line*()	y intercept = value	colour size linetype alpha
geom$_v$*line*()	x intercept = value	

ggplot geoms

Geometric objects (geoms) are functions that determine what kind of object we produce. Below are ones that are commonly used though there are hundreds more.

- geom_bar(): creates a layer with bars
- geom_point(): creates a layer showing datapoints (scatterplot)
- geom_line(): creates a layer connecting data points with a straight line
- geom_smooth(): creates a smoother line
- geom_histogram(): creates a histogram layer
- geom_boxplot(): creates a boxplot
- geom_text(): creates a layer with text on it
- geom_density(): creates a layer with a density plot on it
- geom_errorbar(): creates a layer with error bars
- geom_hline() and geom_vline(): create a layer with a horizontal or vertical line