DATA VISUALIZATION

www.linkedin.com/in/irwanwahyu

Data Visualization

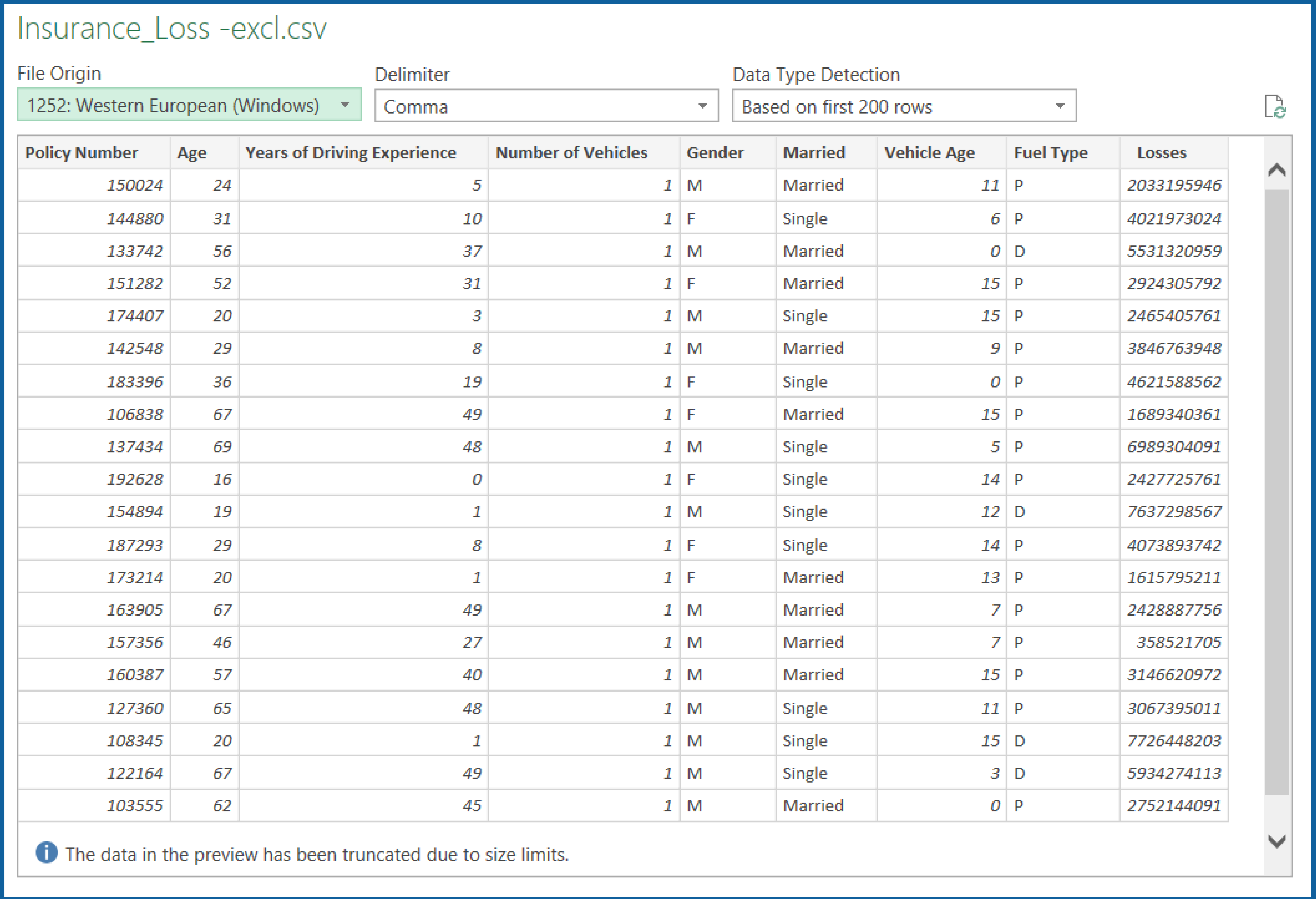
How to Display Data for Easy Understanding

Data visualization is a graphical or visual display of information and data.

In other words, data visualization turns data sets into simpler things to display.

Fast Answer.

Can get insight from this data...???



Insurance_Loss -excl.csv

File Origin: 1252: Western European (Windows)

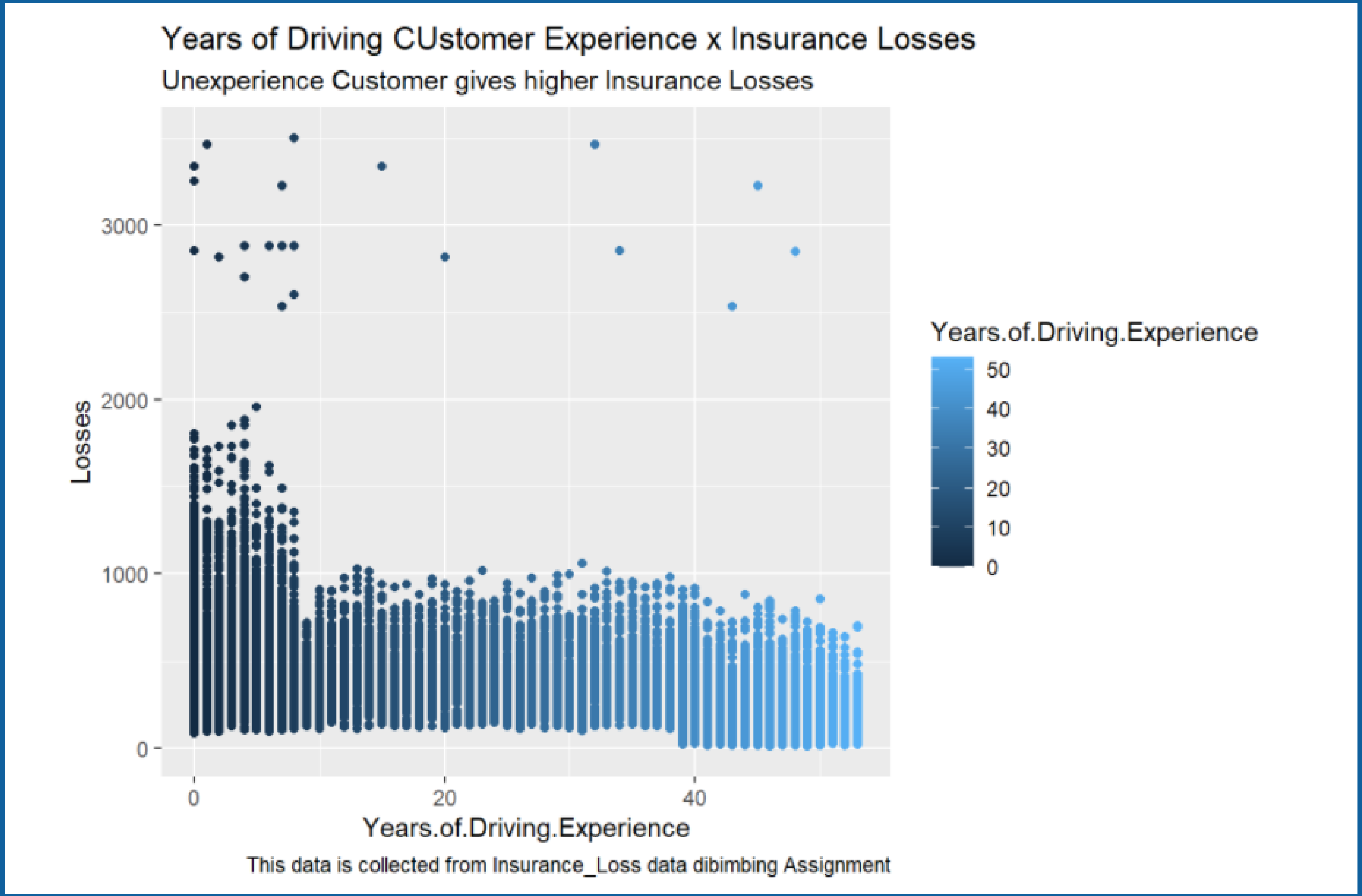
Delimiter: Comma

Data Type Detection: Based on first 200 rows

Policy Number	Age	Years of Driving Experience	Number of Vehicles	Gender	Married	Vehicle Age	Fuel Type	Losses
150024	24	5	1	M	Married	11	P	2033195946
144880	31	10	1	F	Single	6	P	4021973024
133742	56	37	1	M	Married	0	D	5531320959
151282	52	31	1	F	Married	15	P	2924305792
174407	20	3	1	M	Single	15	P	2465405761
142548	29	8	1	M	Married	9	P	3846763948
183396	36	19	1	F	Single	0	P	4621588562
106838	67	49	1	F	Married	15	P	1689340361
137434	69	48	1	M	Single	5	P	6989304091
192628	16	0	1	F	Single	14	P	2427725761
154894	19	1	1	M	Single	12	D	7637298567
187293	29	8	1	F	Single	14	P	4073893742
173214	20	1	1	F	Married	13	P	1615795211
163905	67	49	1	M	Married	7	P	2428887756
157356	46	27	1	M	Married	7	P	358521705
160387	57	40	1	M	Married	15	P	3146620972
127360	65	48	1	M	Single	11	P	3067395011
108345	20	1	1	M	Single	15	D	7726448203
122164	67	49	1	M	Single	3	D	5934274113
103555	62	45	1	M	Married	0	P	2752144091

The data in the preview has been truncated due to size limits.

How About this ?



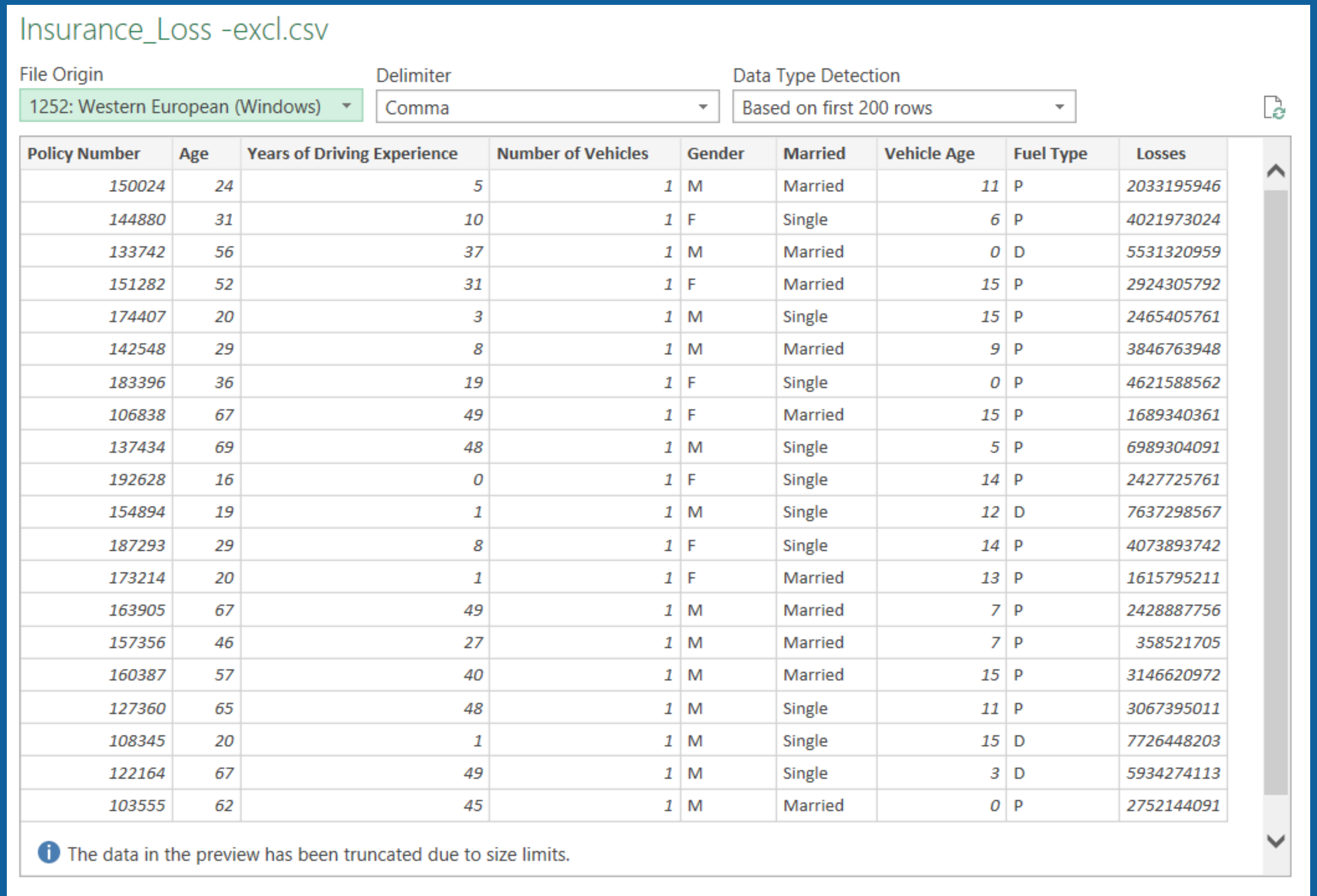
Years of Driving CUstomer Experience x Insurance Losses

Unexperience Customer gives higher Insurance Losses

This data is collected from Insurance_Loss data dibimbing Assignment

www.linkedin.com/in/irwanwahyu

Did you know
it from same data source...



Insurance_Loss -excl.csv

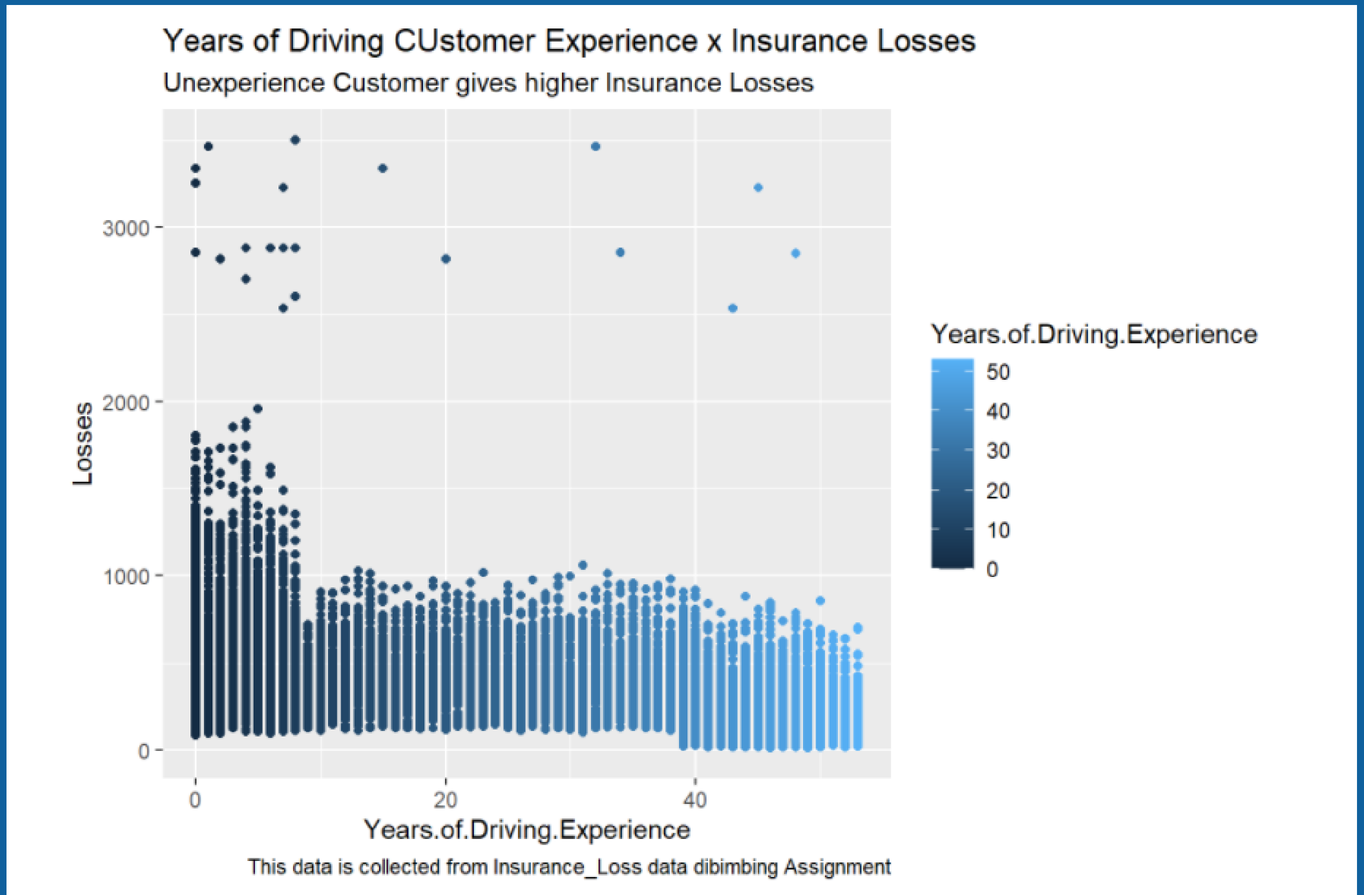
File Origin: 1252: Western European (Windows)

Delimiter: Comma

Data Type Detection: Based on first 200 rows

Policy Number	Age	Years of Driving Experience	Number of Vehicles	Gender	Married	Vehicle Age	Fuel Type	Losses
150024	24	5	1	M	Married	11	P	2033195946
144880	31	10	1	F	Single	6	P	4021973024
133742	56	37	1	M	Married	0	D	5531320959
151282	52	31	1	F	Married	15	P	2924305792
174407	20	3	1	M	Single	15	P	2465405761
142548	29	8	1	M	Married	9	P	3846763948
183396	36	19	1	F	Single	0	P	4621588562
106838	67	49	1	F	Married	15	P	1689340361
137434	69	48	1	M	Single	5	P	6989304091
192628	16	0	1	F	Single	14	P	2427725761
154894	19	1	1	M	Single	12	D	7637298567
187293	29	8	1	F	Single	14	P	4073893742
173214	20	1	1	F	Married	13	P	1615795211
163905	67	49	1	M	Married	7	P	2428887756
157356	46	27	1	M	Married	7	P	358521705
160387	57	40	1	M	Married	15	P	3146620972
127360	65	48	1	M	Single	11	P	3067395011
108345	20	1	1	M	Single	15	D	7726448203
122164	67	49	1	M	Single	3	D	5934274113
103555	62	45	1	M	Married	0	P	2752144091

ⓘ The data in the preview has been truncated due to size limits.



Years of Driving CUstomer Experience x Insurance Losses

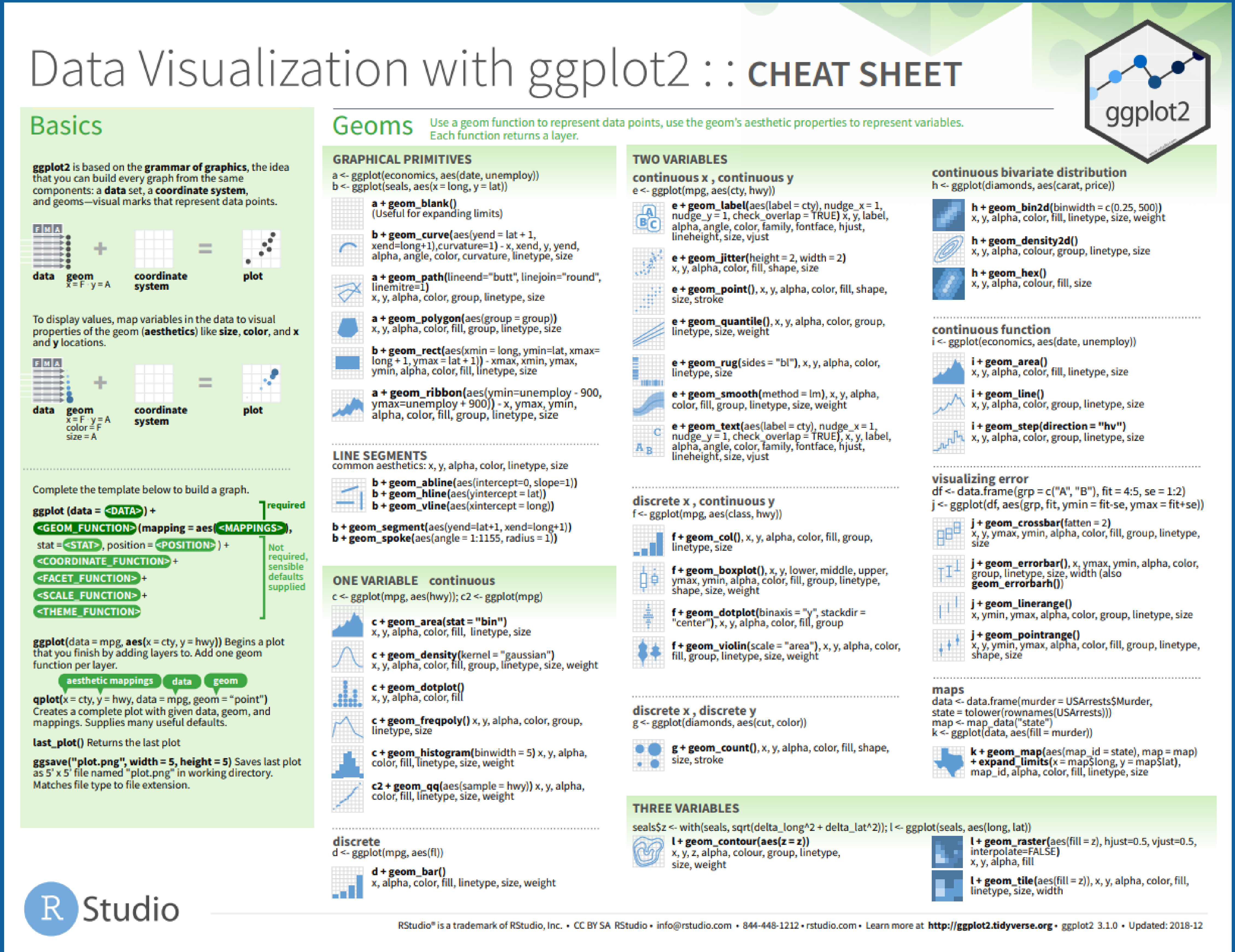
Unexperience Customer gives higher Insurance Losses

This data is collected from Insurance_Loss data dibimbing Assignment

www.linkedin.com/in/irwanwahyu

Why Visualization is so important?

- What People see First is the Graphic (Most of the time)
- It help people understand the data more easy
- It can summarize your findings/insight in one image
- It Fun :D



Data Visualization with ggplot2 : : CHEAT SHEET

ggplot2

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and geoms—visual marks that represent data points.

F M A
data
geom
x = F · y = A
coordinate system
plot

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

Complete the template below to build a graph.

```
ggplot (data = <DATA>) +                                   required
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
 stat = <STAT>, position = <POSITION>) +                    Not
<COORDINATE_FUNCTION> +                                     required,
<FACET_FUNCTION> +                                          sensible
<SCALE_FUNCTION> +                                          defaults
<THEME_FUNCTION>                                            supplied
```

ggplot(data = mpg, **aes**(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

aesthetic mappings · data · geom

qplot(x = cty, y = hwy, data = mpg, geom = “point") Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

last_plot() Returns the last plot

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

a + geom_blank()
(Useful for expanding limits)

b + geom_curve(aes(yend = lat + 1, xend=long+1),curvature=1) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + geom_path(lineend="butt", linejoin="round", linemitre=1**)**
x, y, alpha, color, group, linetype, size

a + geom_polygon(aes(group = group)**)**
x, y, alpha, color, fill, group, linetype, size

b + geom_rect(aes(xmin = long, ymin=lat, xmax= long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900)) - x, ymax, ymin, alpha, color, fill, group, linetype, size

LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

b + geom_abline(aes(intercept=0, slope=1))
b + geom_hline(aes(yintercept = lat))
b + geom_vline(aes(xintercept = long))

b + geom_segment(aes(yend=lat+1, xend=long+1))
b + geom_spoke(aes(angle = 1:1155, radius = 1))

ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian")
x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot()
x, y, alpha, color, fill

c + geom_freqpoly() x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5) x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy)) x, y, alpha, color, fill, linetype, size, weight

discrete
d <- ggplot(mpg, aes(fl))

d + geom_bar()
x, alpha, color, fill, linetype, size, weight

TWO VARIABLES

continuous x , continuous y
e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE**)** x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_jitter(height = 2, width = 2**)**
x, y, alpha, color, fill, shape, size

e + geom_point(), x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile(), x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl"**)**, x, y, alpha, color, linetype, size

e + geom_smooth(method = lm**)**, x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE), x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

discrete x , continuous y
f <- ggplot(mpg, aes(class, hwy))

f + geom_col(), x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot(), x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center"**)**, x, y, alpha, color, fill, group

f + geom_violin(scale = "area"**)**, x, y, alpha, color, fill, group, linetype, size, weight

discrete x , discrete y
g <- ggplot(diamonds, aes(cut, color))

g + geom_count(), x, y, alpha, color, fill, shape, size, stroke

continuous bivariate distribution
h <- ggplot(diamonds, aes(carat, price))

h + geom_bin2d(binwidth = c(0.25, 500)**)**
x, y, alpha, color, fill, linetype, size, weight

h + geom_density2d()
x, y, alpha, colour, group, linetype, size

h + geom_hex()
x, y, alpha, colour, fill, size

continuous function
i <- ggplot(economics, aes(date, unemploy))

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv")
x, y, alpha, color, group, linetype, size

visualizing error
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

j + geom_crossbar(fatten = 2**)**
x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + geom_errorbar(), x, ymax, ymin, alpha, color, group, linetype, size, width (also **geom_errorbarh()**)

j + geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size

j + geom_pointrange()
x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps
data <- data.frame(murder = USArrests$Murder,
state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

k + geom_map(aes(map_id = state), map = map)
+ expand_limits(x = map$long, y = map$lat),
map_id, alpha, color, fill, linetype, size

THREE VARIABLES

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2)); l <- ggplot(seals, aes(long, lat))

l + geom_contour(aes(z = z))
x, y, z, alpha, colour, group, linetype, size, weight

l + geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE**)**
x, y, alpha, fill

l + geom_tile(aes(fill = z)), x, y, alpha, color, fill, linetype, size, width

RStudio

RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at **http://ggplot2.tidyverse.org** • ggplot2 3.1.0 • Updated: 2018-12

in R you can visualize data with **ggplot2 implements the grammar of graphics, a coherent system for describing and building graphs.**

With ggplot2, you can do more faster by learning one system and applying it in many places

www.linkedin.com/in/irwanwahyu