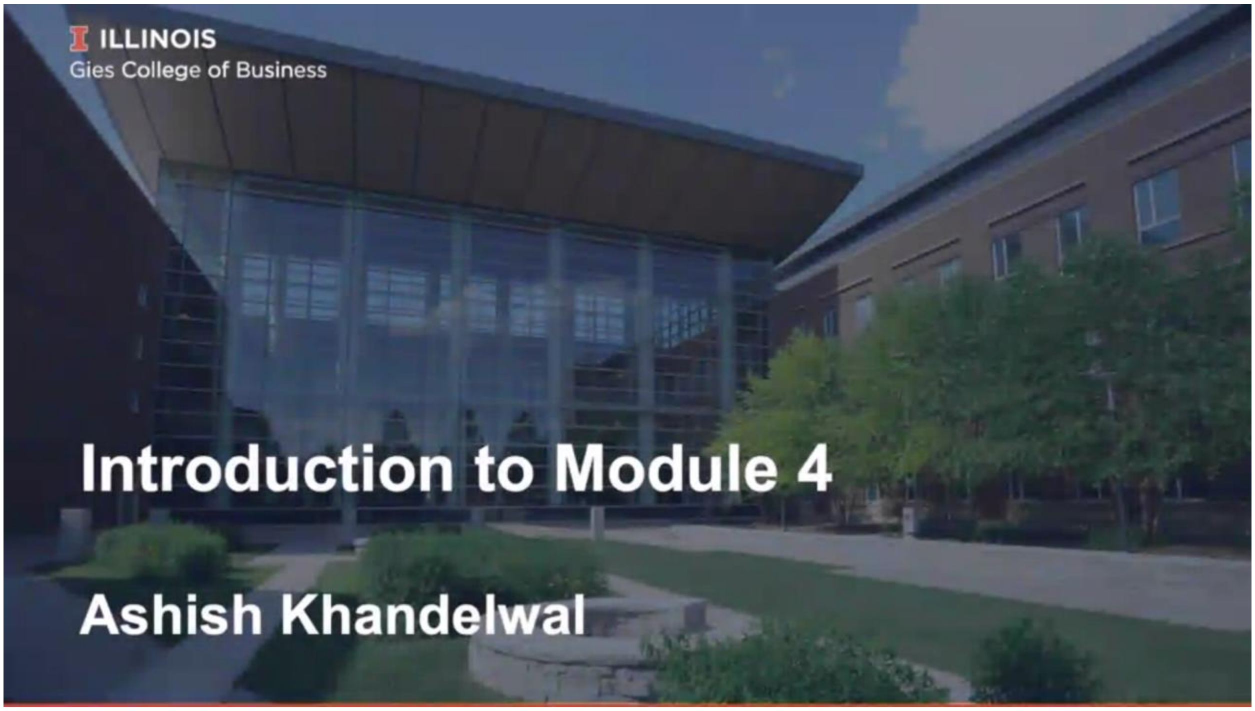
Module 4: Introduction to Business Analytics with R

Table of Contents

Lesson 4 Introduction



ILLINOIS
Gies College of Business

Introduction to Module 4

Ashish Khandelwal

Welcome to Module 4. In this module, we will learn about data visualization techniques and R for exploratory data analysis. Data visualization may serve two purposes, data exploration for patterns in the raw data and communication of the results of analysis. The purpose of this module will be data exploration. The communication aspect will be covered in the next course in this specialization. In this module we will use another popular tidy verse package called ggplot2. In fact, you will see that ggplot2 is much more than a visualization package. It's a data visualization philosophy. I look forward to seeing you in the next video.

Lesson 4-1: What to Explore in the Data?

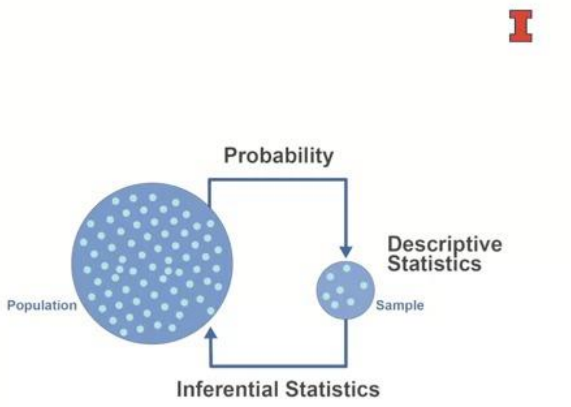
ILLINOIS
Gies College of Business

What to Explore in the Data?

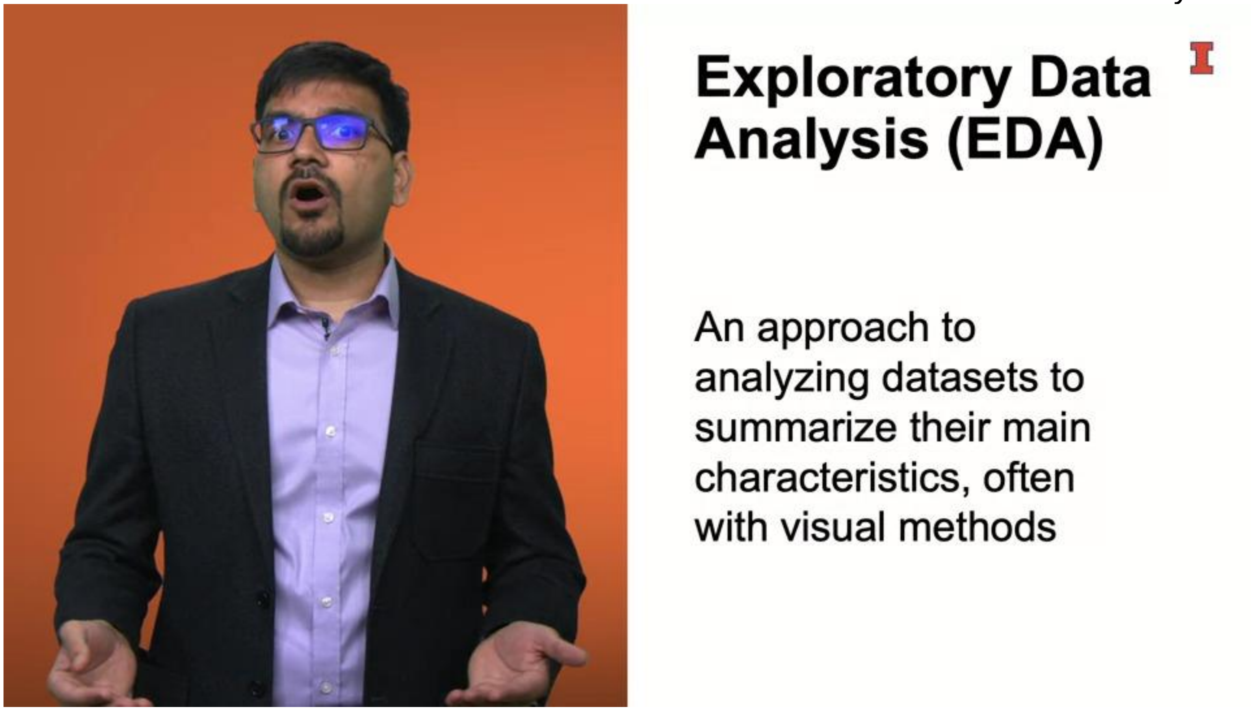
Ashish Khandelwal

In this lesson, we will discuss what EDA is and where does it fit in the overall data analytics?

In statistics, Exploratory Data Analysis is an approach to analyzing data sets to summarize their main characteristics, often with visual methods. Using visual methods like charts, graphs and maps helps us see and understand the trends, outliers and other patents in data.



Let's understand the relation between statistics and data analysis. The core idea is that we are interested in understanding some characteristic of the population, such as mean or proportion. For example, we may be interested in knowing what is the price customers are willing to pay on average. Knowing this information will help us make some prediction for whether a given consumer will be willing to buy our product at a given price. This is where probability comes into the picture. It helps us to use our knowledge about population to make predictions for individuals. However, learning about population is not easy, we seldom have access to all the customers for survey. Therefore, sample is our window to the population, thus we ask a sample of customers about their willingness to pay and make some guess about the population. While generalizing learning from the sample to the population, we need to account for the uncertainty as the sample may not be representative of the population. This is the domain of inferential statistics. You will learn more about this in predictive modeling course. However, when we collect sample data, we are also interested in learning about the sample itself to see some interesting patterns in it. This is the domain of descriptive statistics and exploratory data plays an important role here.

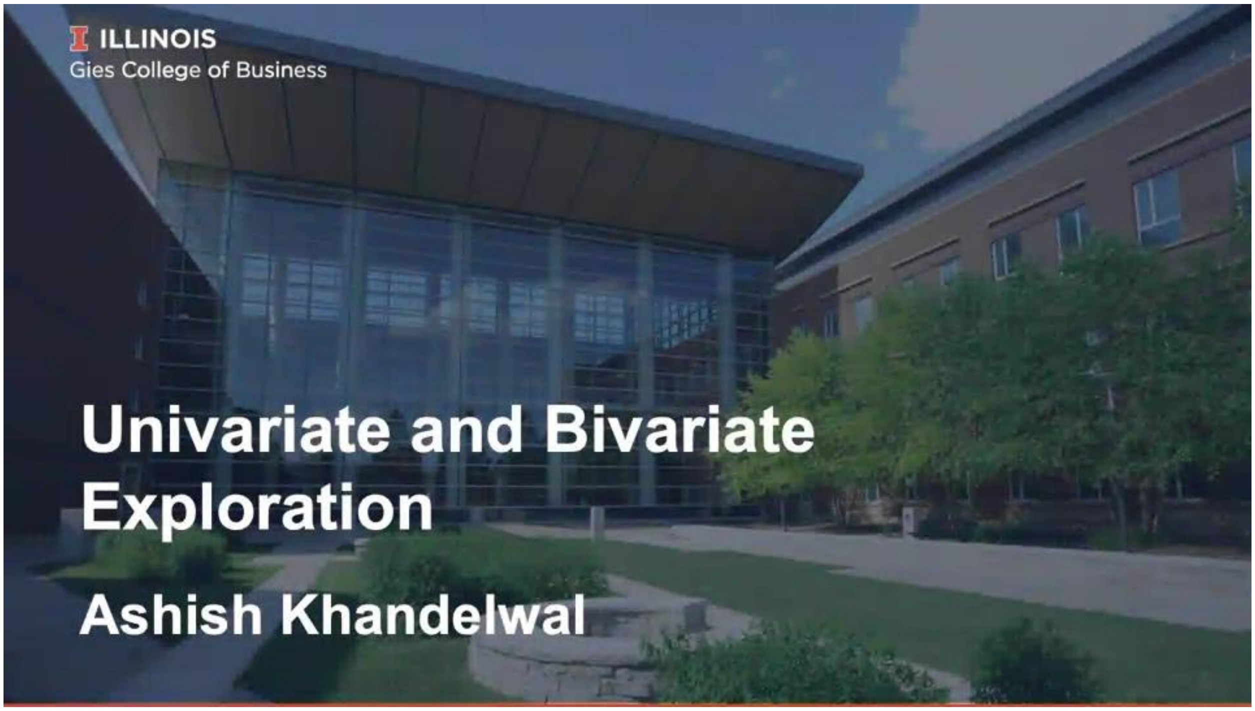


Exploratory Data Analysis (EDA)

An approach to analyzing datasets to summarize their main characteristics, often with visual methods

Let's understand what do we want to explore in our data. We want to explore missing values, outliers, univariate distributions and bivariate and multivariate distributions. Hopefully you see where EDA fits in the overall data analysis.

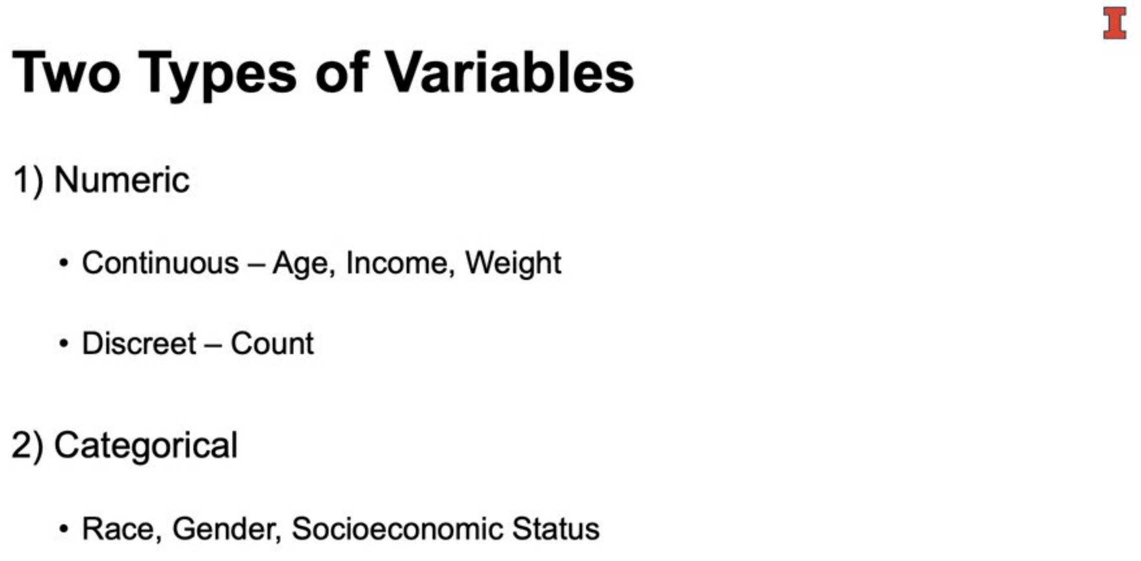
Lesson 4-2: Univariate and Bivariate Exploration



Univariate and Bivariate Exploration

Ashish Khandelwal

In this lesson, we will cover basic visualization types for univariate and bivariate analysis.



Two Types of Variables

1) Numeric

- Continuous – Age, Income, Weight
- Discreet – Count

2) Categorical

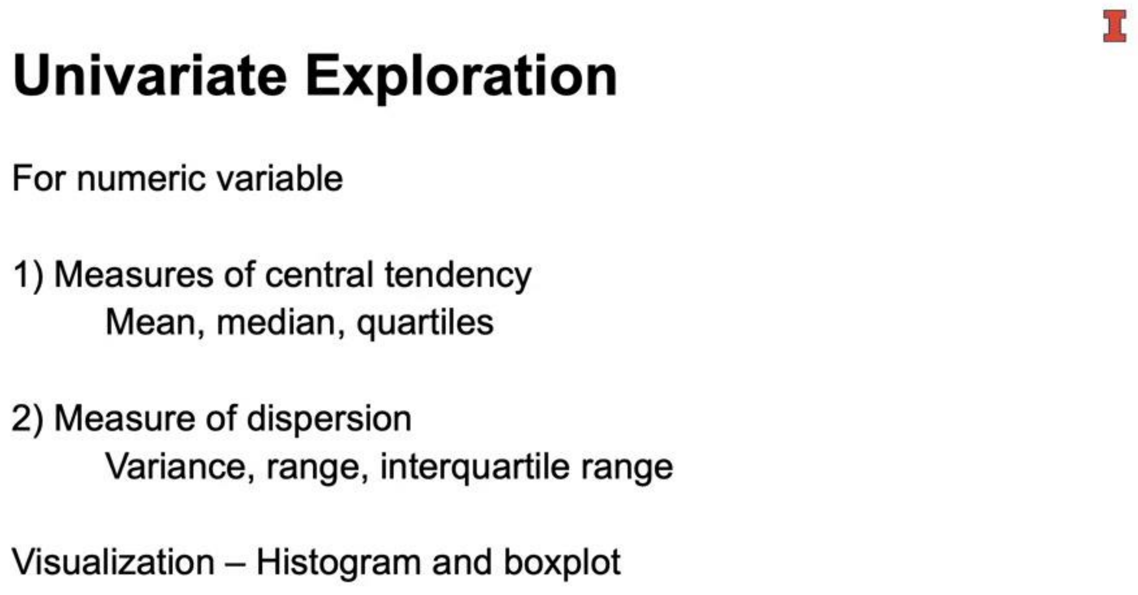
- Race, Gender, Socioeconomic Status

First, let's understand that broadly there are two types of variables: numeric and categorical. Numeric variables are usually continuous variables that can take any values such as age, weight, income. Sometimes numerical variables may also be discrete, such as count variables that can take only integer values, such as the number of people in the family. Categorical variables, as the name suggests, represents categories such as gender, race, socio-economic class, etc.

Exploration

- Univariate
- Bivariate
- Multivariate

We can perform the following explorations in the data: univariate, bivariate, and multivariate explorations. Univariate exploration involves exploring one variable at a time.



Univariate Exploration

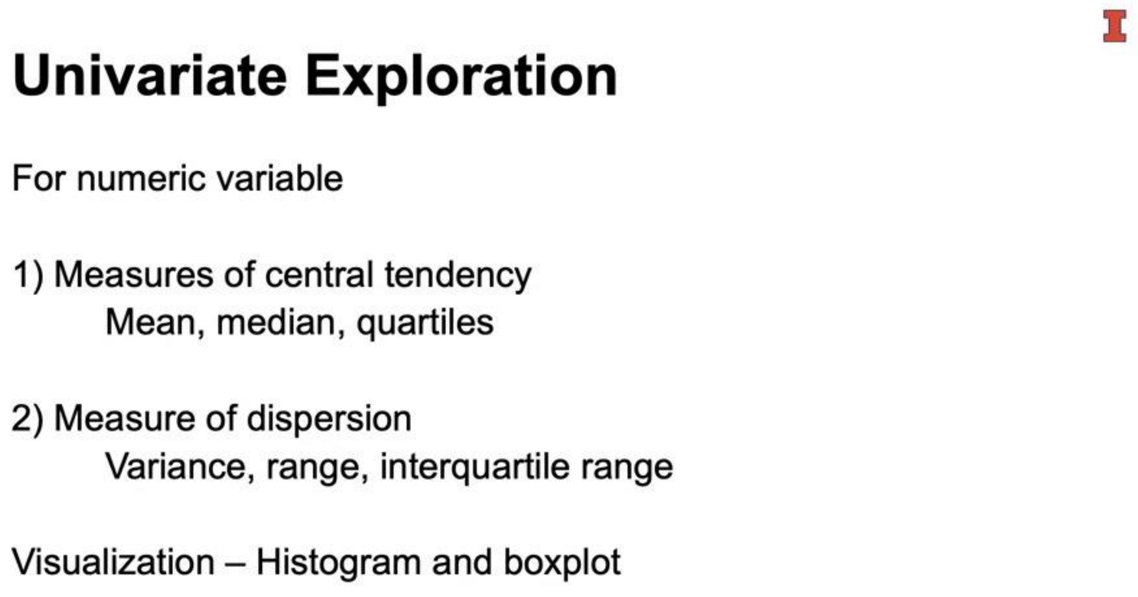
For numeric variable

1) Measures of central tendency
    Mean, median, quartiles

2) Measure of dispersion
    Variance, range, interquartile range

Visualization – Histogram and boxplot

In the univariate case, for numerical variables, we explore two things: measures of central tendency, such as means, medians and quartiles, and measures of dispersion such as variance, range, and interquartile range.



Univariate Exploration

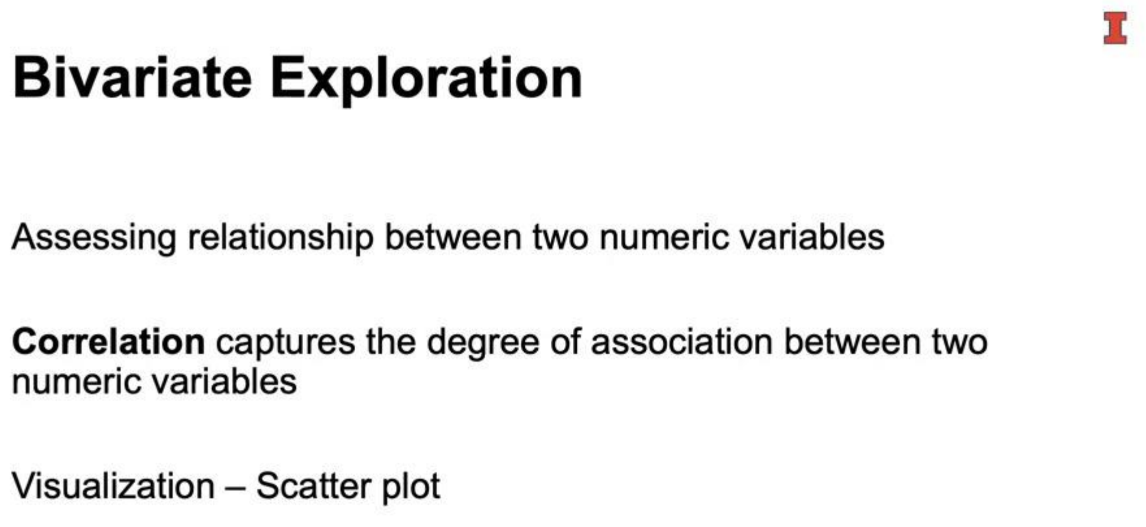
For numeric variable

1) Measures of central tendency
    Mean, median, quartiles

2) Measure of dispersion
    Variance, range, interquartile range

Visualization – Histogram and boxplot

The visualization methods, in this case, are histograms, box plots, and violin plots. Univariate exploration for categorical variables would involve exploring the counts and counts proportions for various categories. You may look at the frequency table to assess these, and you can visualize them using bar charts and column charts.



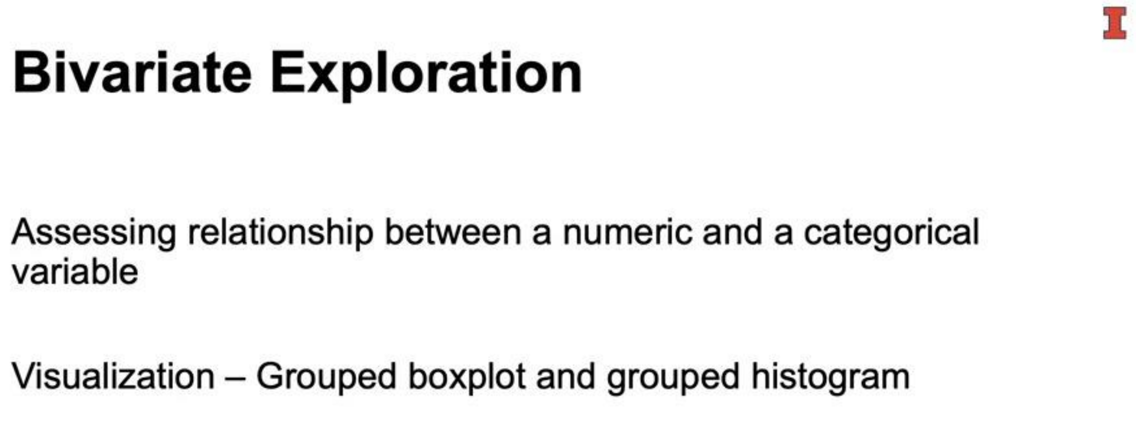
Bivariate Exploration

Assessing relationship between two numeric variables

Correlation captures the degree of association between two numeric variables

Visualization – Scatter plot

Let's understand the bivariate exploration. In the bivariate exploration, we may have both the variables numeric, in which case we will assess the degree of association between the variables. Correlation captures the degree of association between two numeric variables. The visualization method to assess the degree of association between two numeric variables is a scatter plot, where one of the variables goes on x-axis and the other one goes on the y-axis.

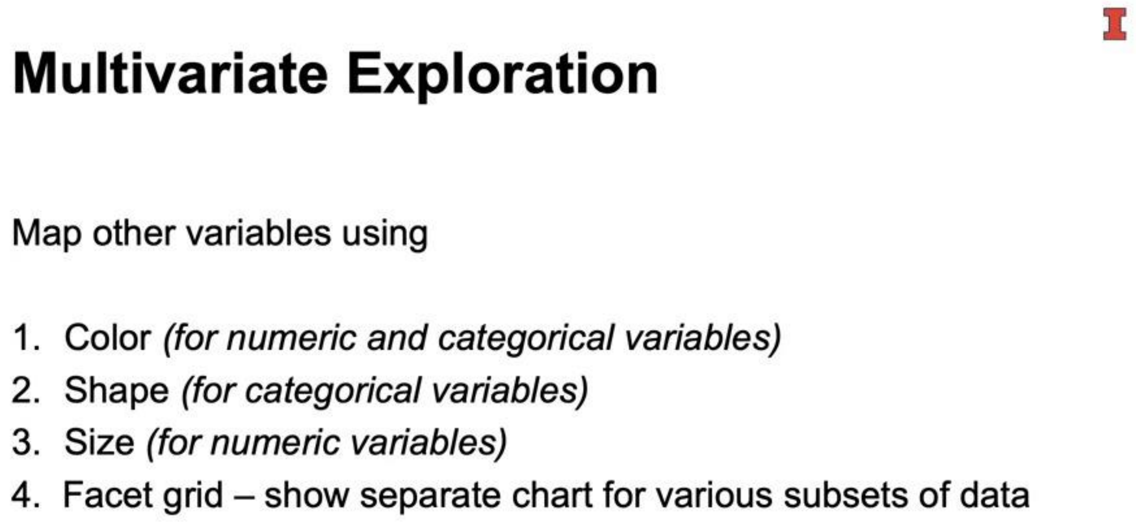


Bivariate Exploration

Assessing relationship between a numeric and a categorical variable

Visualization – Grouped boxplot and grouped histogram

When one of the variables is categorical and another is continuous or numerical, we can have a bivariate exploration using either grouped boxplots or grouped histograms.



Multivariate Exploration

Map other variables using

1. Color *(for numeric and categorical variables)*
2. Shape *(for categorical variables)*
3. Size *(for numeric variables)*
4. Facet grid – show separate chart for various subsets of data

In the case of multivariate exploration, we can bring third, fourth, and even fifth variable using other aesthetics such as color and shapes. Finally, we can also use grade where visual multiple plots for different subsets of data. These things will become clearer when we create these graphics in R.

Lesson 4-3.1: Introduction to the ggplot2 Package

ILLINOIS
Gies College of Business

Introduction to the Ggplot2 Package

Ashish Khandelwal

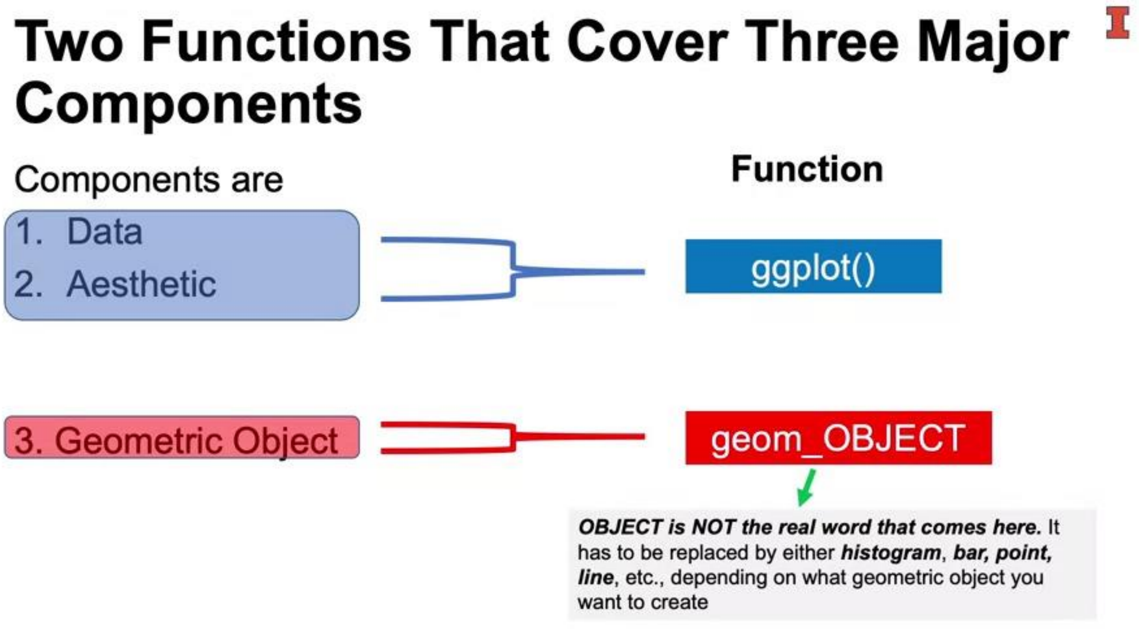
In R, you have mainly two options to create visualizations; base R plotting and ggplot, and it's a topic of big debate which one to use. Base graphics is good because it offers simple and fast way to create plots. However, it offers much less flexibility and there is no consistent coding for various types of plots. On the other hand, ggplot builds more elegant plots with a lot of flexibility, and ggplot also offers qplot for quick plotting similar to base graphics. However, to take the real advantage of ggplot's flexibility, one must use its main plots, and not qplots. The codes for ggplots are not trivial, and may involve learning curve, but don't worry. In this course, I will share a useful tool which would allow us to create ggplot visualization without using any code at all. Feel free to use codes for your visualization task if that's what you prefer. Ggplot is more of a plotting philosophy than just a package. It's built on the concept of grammar of graphics introduced by Wilkinson in 1999, and this grammar of graphics was implemented in R package by Hadley Wickham in 2005. Hadley Wickham is the chief data scientist without Rstudio, and most of our tidy-wise packages are developed by him. Let us understand the basic building blocks of ggplot. In ggplot, each chart is a combination of eight different components. The first component is the data. The second component is called the aesthetic mappings. These are the variables that get mapped to your x-axis and y-axis. The third component is the geometric objects. The objects that we want to create, such as histogram, box-plot, bar charts. Next is statistical transformation. Here you specify whether you want to transform your variables using log square or any other transformation. Then comes the scales. This maps values from data space to aesthetic space. The next is the coordinate system. Most frequently used is Cartesian coordinates. However, you have options to use other coordinates such as polar. The second to last is the faceting. It breaks up the data into subsets for creating separate graphics for each subset. The final one is theme. It allows us to control the font size, background-color, and legend position in our graphics. This could be overwhelming to learn so much. But don't worry, we only need to master the top three layers. The other layers have nice default setup. Thus, you can just focus on the top three layers for most of your basic visualizations. Moreover, in this course, I will show you a nice tool that will allow us to create these charts without coding. Another nice thing about this tool will be that it lets us save the code that we can run from Rstudio if we want to recreate these graphs in the future. Please note, ggplot does not inform us on which graphics to use to answer the questions you are interested in, and it offers no option for interactive plotting. However, there are extension packages such as ggli and gganimate that add interactive plotting features to ggplot. Hopefully, you get an idea that ggplot is a powerful tool that allows us creating very useful charts to explore our data.

Lesson 4-3.2: Introduction to ggplot Syntax

ILLINOIS Gies College of Business

Introduction to Ggplot Syntax

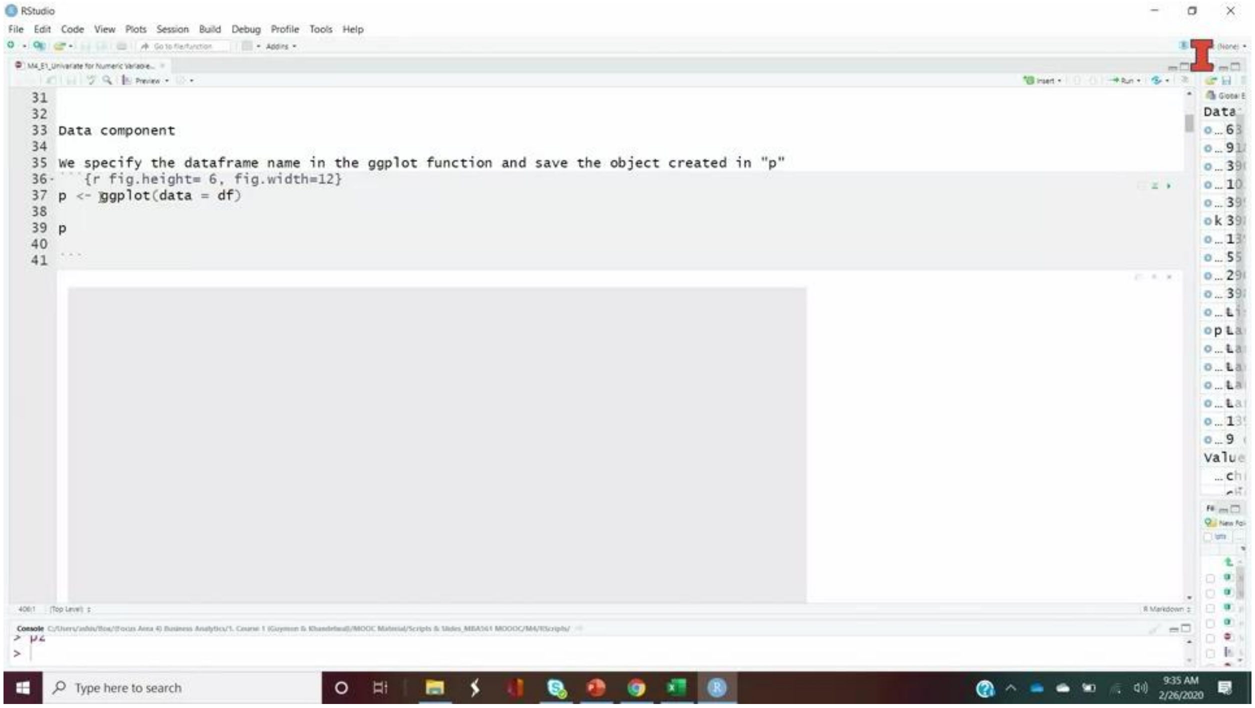
Ashish Khandelwal



Two Functions That Cover Three Major Components

OBJECT is NOT the real word that comes here. It has to be replaced by either ***histogram***, ***bar, point, line***, etc., depending on what geometric object you want to create

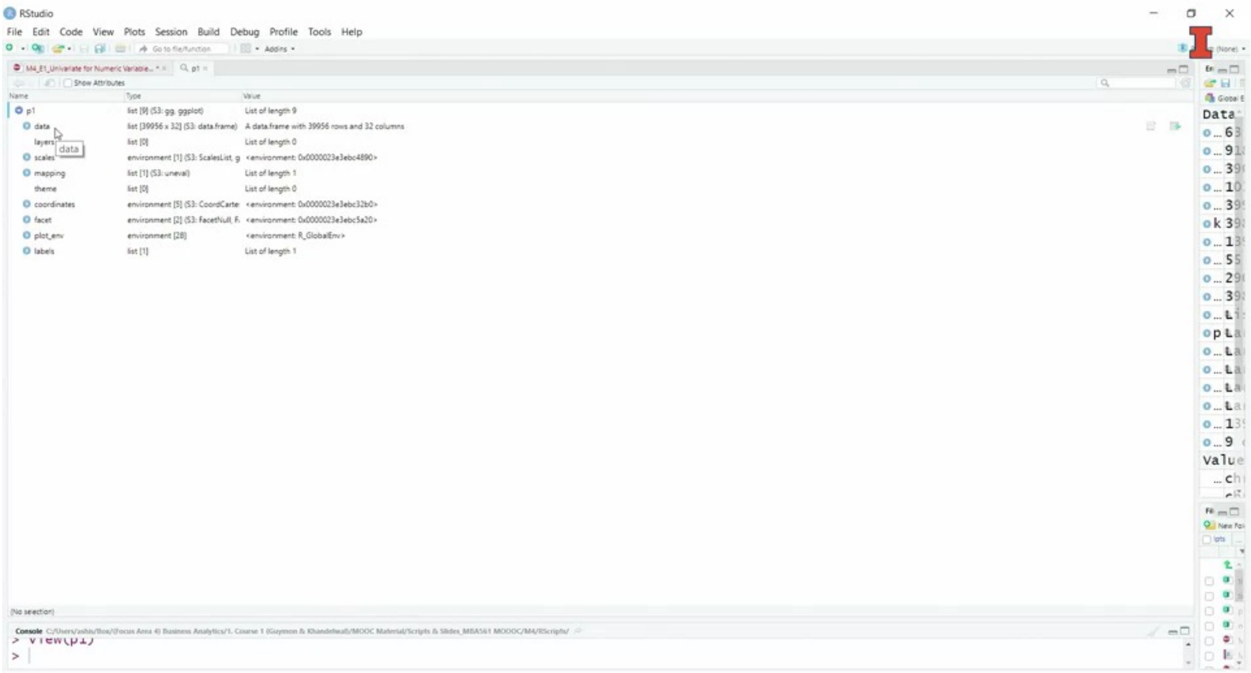
As discussed earlier, we need to use minimum two functions and the specify minimum three components for any ggplot. The other components can be taken care of by nice defaults that the package provides. However, as you gain more familiarity with ggplot, you can start using other functions also. The first component in a ggplot is the data. In the data component, we need to specify what is the data-frame that we are going to use. So you remember that we discussed that minimum two functions have to be specified. The first one is ggplot itself, and the second one is Geom object, which captures the geometric object.



~~~
Data component

We specify the dataframe name in the ggplot function and save the object created in "p"
```{r fig.height= 6, fig.width=12}
p <- ggplot(data = df)

p
```
~~~

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
M4_E1_Univariate for Numeric Variable...
Preview
Console
Type here to search
9:35 AM
2/26/2020

Now let's see what goes in the ggplot function. Data have to be specified in the ggplot function itself, and you would see as soon as I do this, ggplot already lays out the area on which it's going to create the plot. It has got no variables mentioned because we have not specified those. Note, nothing to see because we have not created any object. Let's add the other component to this. Here I am adding the second component. So after the data, we need to specify which variables are we going to use, and that goes into the aesthetic mapping. So here is how I would look at this code, in the P1 object. I am going to put this ggplot object where the data is specified to be df, which is a shorter version of where ities data, and in the mappings argument, I am going to specify the aesthetics. As mentioned, aesthetics are the variables that you want to create charts on. Here, I'm only going to use one variable, that is price, because we are in univariant world right now. As soon as I do this and I run this code, ggplot creates not only the area for the plotting, but also it specifies the variable price. However, we still don't see what we want to visualize. So for univariant, you would remember that we discussed we can have either a histogram or a boxplot, but we don't see anything here yet. Why? Because we have not specified what geometric objects to plot and that's the third component that we need to tell ggplot, and the function that we use is this. So here P1 has already saved our data object along with the aesthetic mapping. It's already available. If we want to see this, we can just view P1 and we can quickly inspect it.


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
M4_E1_Univariate for Numeric Variable...
p1
Show Attributes
(No selection)
Console

Name	Type	Value
p1	list [9] (S3: gg, ggplot)	List of length 9
data	list [39956 x 32] (S3: data.frame)	A data.frame with 39956 rows and 32 columns
layers	list [0]	List of length 0
scales	environment [1] (S3: ScalesList, g	<environment: 0x0000023e3ebc4890>
mapping	list [1] (S3: uneval)	List of length 1
theme	list [0]	List of length 0
coordinates	environment [5] (S3: CoordCarte	<environment: 0x0000023e3ebc32b0>
facet	environment [2] (S3: FacetNull, F.	<environment: 0x0000023e3ebc5a20>
plot_env	environment [28]	<environment: R_GlobalEnv>
labels	list [1]	List of length 1

```
> View(p1)
>
```

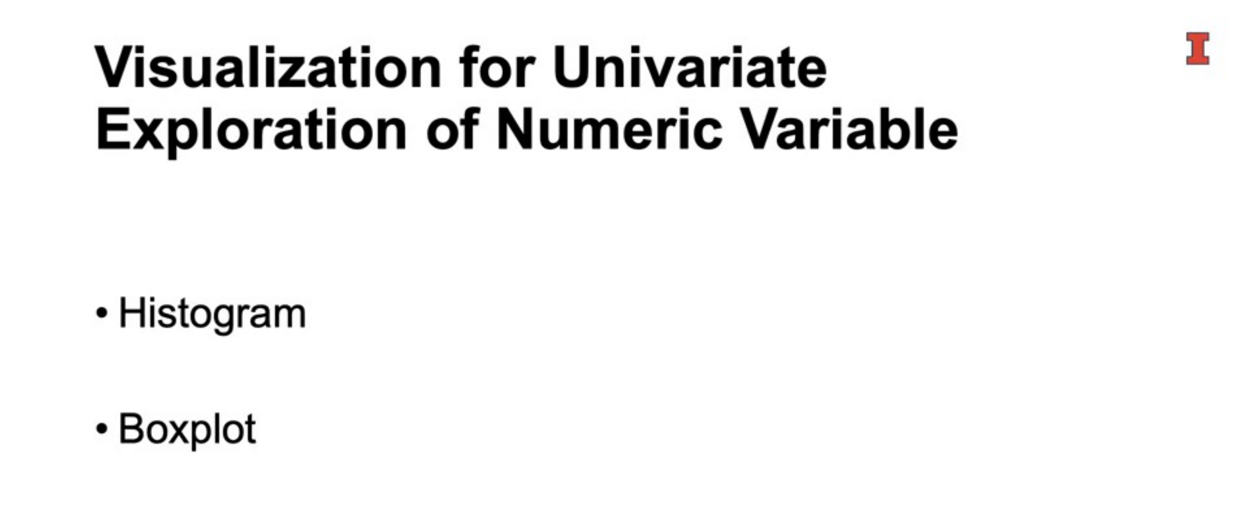
So as you see already in P1, we have data layers, scales, mappings, and other things, but we do not see any object. We have specified data and mapping, so they are there. If you see what we have in data is our entire data-frame, ities data. Now if you see the mappings, it tells us that on the x axis we want price to be specified. That's what it has saved and it has saved the other default options for the other components or layers. Let's go to this chart now where we want to add the histogram. So on top of my P1, I need to add this code geom_histogram, telling R to create histogram for me, and histogram takes one argument, which is bins.

Leson 4-4.1: Univariate Exploration of a Numeric Variable

ILLINOIS
Gies College of Business

Univariate Exploration of a Numeric Variable

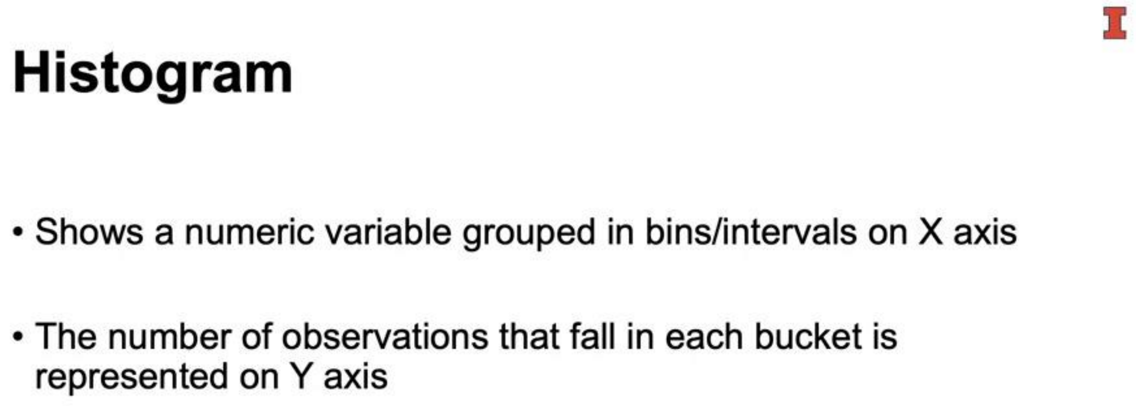
Ashish Khandelwal



Visualization for Univariate Exploration of Numeric Variable

- Histogram
- Boxplot

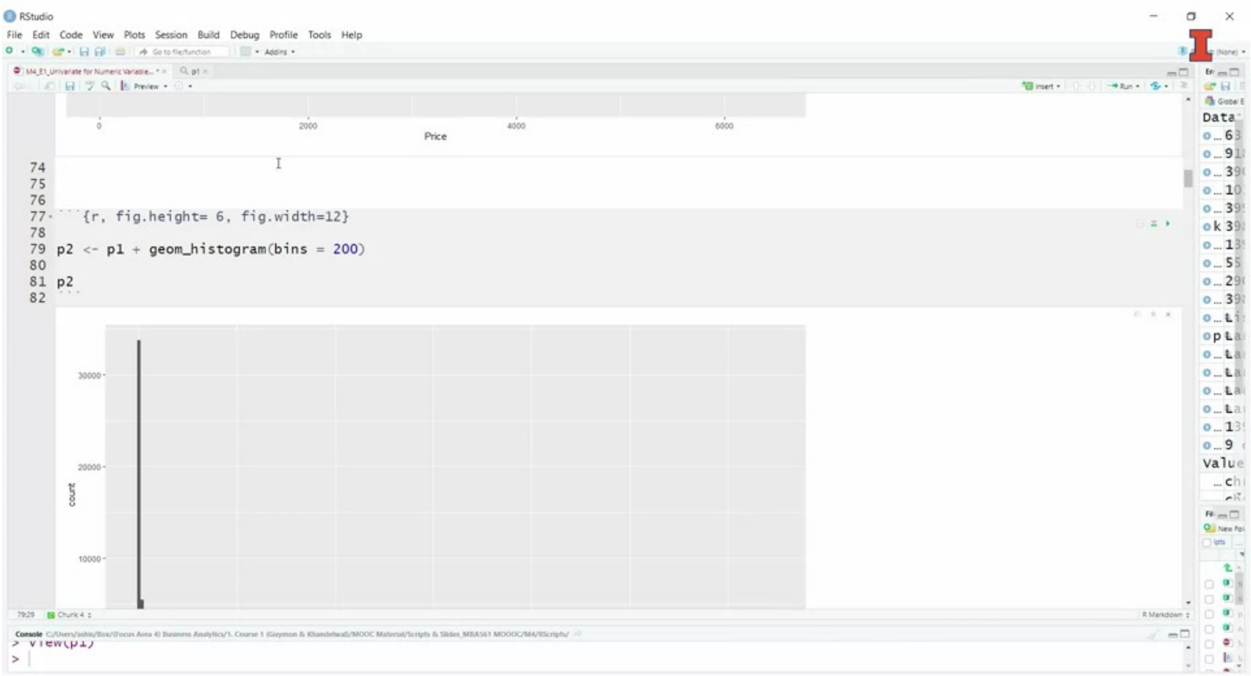
In this lesson, we will be discussing how to plot univariate charts for numerical variables. Examples of numerical variables from itized data could be price or quantity.



Histogram

- Shows a numeric variable grouped in bins/intervals on X axis
- The number of observations that fall in each bucket is represented on Y axis

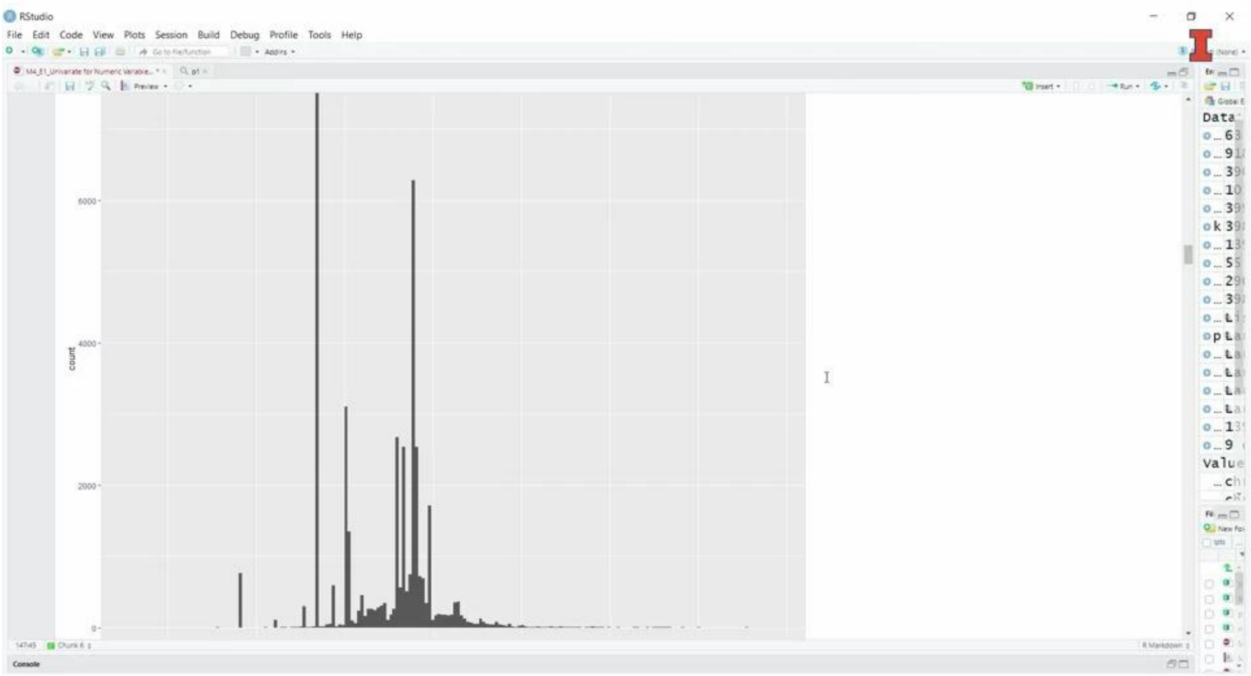
So essentially what is a histogram? Let's first discuss what's histogram. So histogram shows a numerical variable grouped in different bins or buckets on the x-axis. And the number of observations that fall in each of these buckets is represented on Y axis. You usually looking for whether a histogram is symmetric or not that kind of tells how are you values distributed. So if the values are normally distributed you will see histogram, which is sort of forming a bell-shaped figure. If you see a histogram where the mound or the higher values on the Y axis are on the left and sort of smaller values going on the right you call it a right skewed distribution, sort of you are looking for a you are looking at a tail on the right side, so that's a right skewed distribution. Alternatively you can have a left skewed distribution.



~~~
```{r, fig.height= 6, fig.width=12}

p2 <- p1 + geom_histogram(bins = 200)

p2
```
~~~

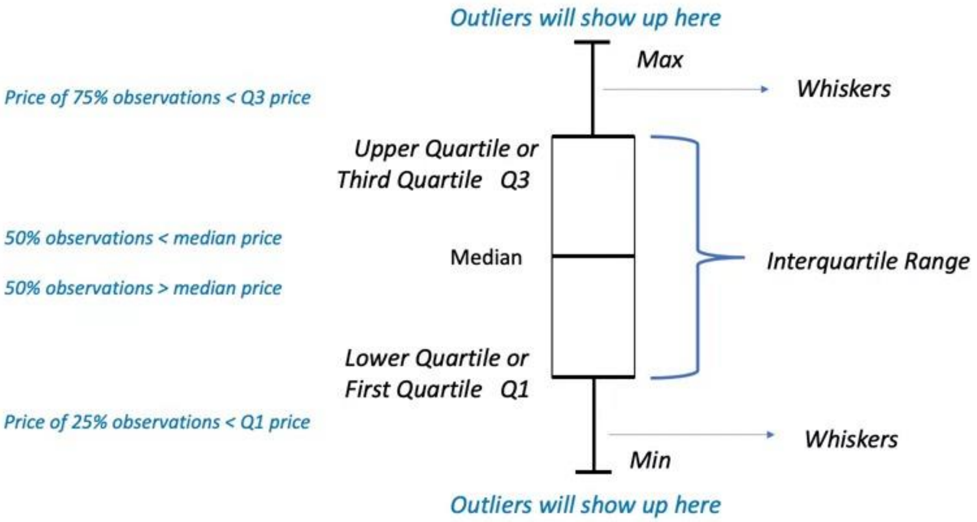
```
> View(p1)
>
```

So one of the arguments that you need to specify in histogram is in how many buckets do you want to divide the entire data? The default is 30 but suggested number is the square root of the total number of observations you have. In our data we have 40,000 observations and that's why approximately 200 would be a good number of bins. So that's why I'm specifying 200 bins here and I plot this histogram. When I plot this histogram I see that the X axis is going all the way up to 7, 8,000, but I really don't see anything happening beyond 10 or 50 on the x-axis and it's not very informative. So our data may be right skewed. Another thing that we can do is we can transform our data may be on a log scale. Because it is just going to bring the larger values close to the smaller values and that might help us see things better. So what I'm doing here in this piece of code is I am going to create instead of price the aesthetic that I am going to use is the log of price, that's it, nothing else. And then I can do bins 230, you have to try various bins to see which really looks better. Let's just run this code to see what we get.


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
count
6000
4000
2000
0
Console

So when we log transformed it, at least we see that the histogram, not exactly symmetric, but has the higher values in the middle.

Boxplot of Price



Outliers will show up here

Max

Whiskers

Price of 75% observations < Q3 price

Upper Quartile or Third Quartile Q3

50% observations < median price

Median

Interquartile Range

50% observations > median price

Lower Quartile or First Quartile Q1

Price of 25% observations < Q1 price

Whiskers

Min

Outliers will show up here

The next thing that I want to discuss for a univariate exploration of a numerical data is Boxplot. So let's quickly see what a box plot offers. Boxplot shows you a couple of things that you may want to investigate. The first one is the first quartile. What's the first quartile? The value of price below which lie 25% observations. Then is median, the value of price below which live 50% of the observations and around 50% of the observations will have price above this given price, that's the median price. Then you have third quartile, which is 75th percentile. It also reports the minimum and maximum values that you see as the two ends of these whiskers. But remember these minimum and maximum are not the actual minimum and maximum in your data. These are the values that are 1.5 interquartile range away from your 25th percentile in 75th percentile. Any values that are above or below these whiskers are outliers. So let's quickly see that when we create a boxplot we get Lot of values at the top as outliers. And just for our visual inspection, I'm going to zoom in onto a narrow range for Y values. And when I do that, We see that there are a lot of these values on the higher side that are outliers because they are beyond this point. This is the point that we are looking for anything beyond this is an outlier, anything below this is an outlier. This is our 25th percentile or first quartile, this is median, and this is third quartile or 75th percentile. Let's quickly see the boxplot for the log price. Now, I see outiers on both the sides. But without zooming on I can see my boxplot clearly here. So this is because we have used the log transformation, which is helpful.

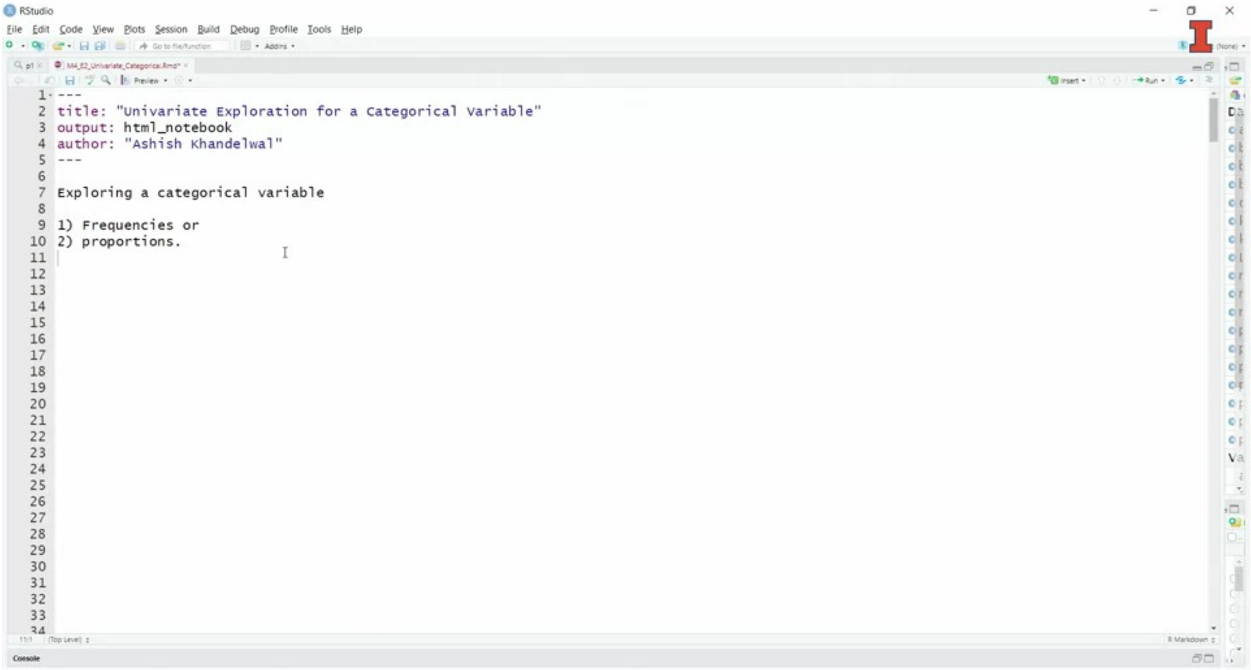
Lesson 4-4.2: Univariate Exploration of a Categorical Variable

ILLINOIS
Gies College of Business

Univariate Exploration of a Categorical Variable

Ashish Khandelwal

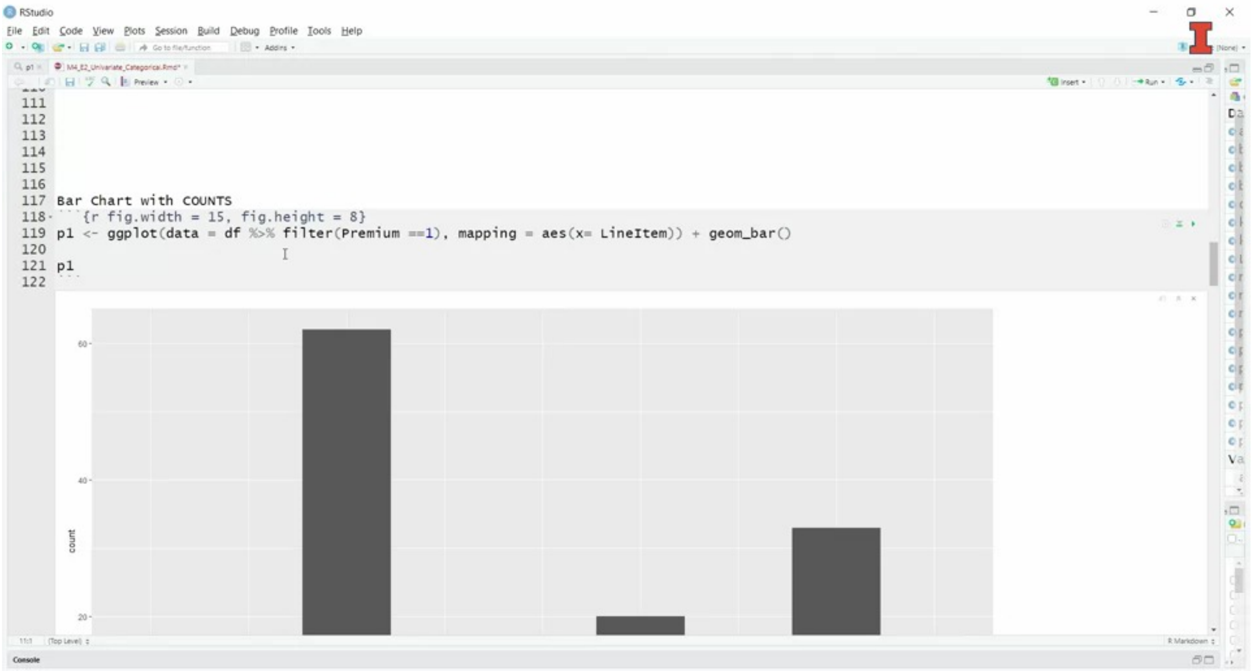
In this lesson, we will see how can we explore a categorical variable that means univariate exploration for the categorical variable.


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
M4_E2_Univariate_Categorical.Rmd*
Preview
Insert
Run
Console

```
---
title: "Univariate Exploration for a Categorical Variable"
output: html_notebook
author: "Ashish Khandelwal"
---

Exploring a categorical variable

1) Frequencies or
2) proportions.
```

So when we explore a categorical variable, we can look at two things, either the frequencies of the categories or the proportions. Visualizing a categorical variable can be done in two ways, either a bar chart or a column chart. They're essentially the same thing. It's just the information is represented in a different format. So you remember we created this variable called Premium

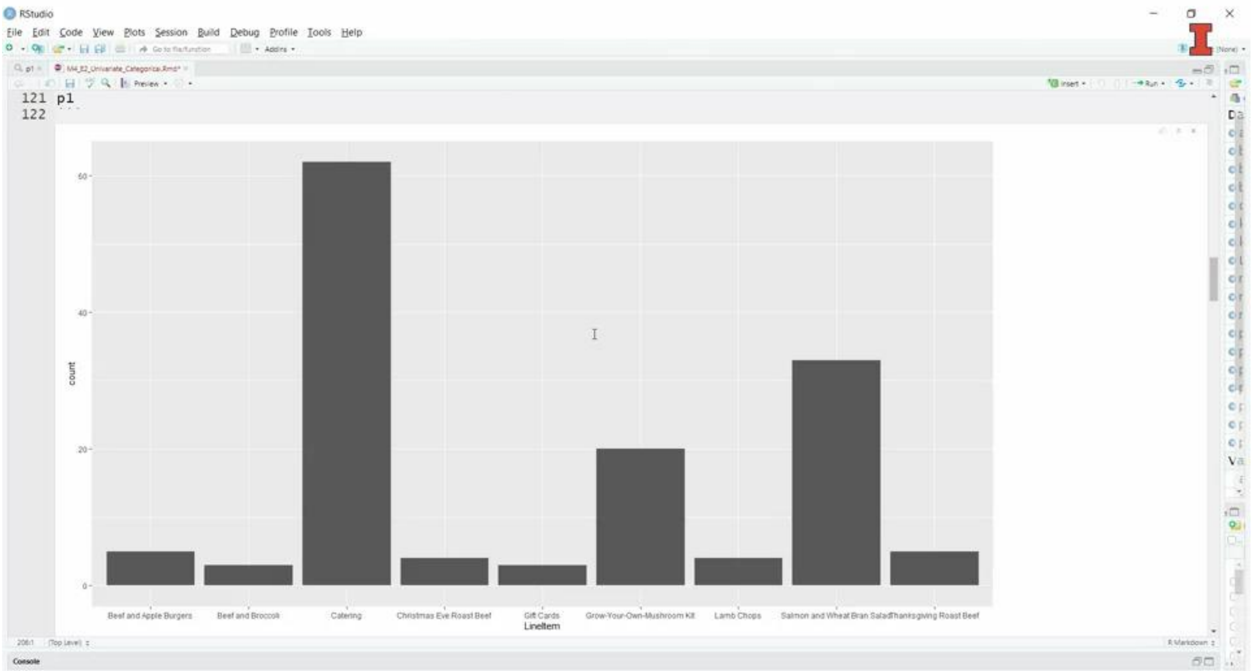
Products. If the price was greater than mean plus two standard deviation, then we coded these products as premium, otherwise non-premium. Now for this lesson's purpose, I am going to focus only on the Premium Products. And we will also learn how ggplot allows us to subset the data on the fly. That means when you are creating these visualizations and if you want to focus only on a given subset of the data, you can just do that within the ggplot code itself.


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
M4_E2_Univariate_Categorical.Rmd*
Preview
Insert
Run
60
40
20
count
Console
R Markdown

~~~
Bar Chart with COUNTS
```{r fig.width = 15, fig.height = 8}
p1 <- ggplot(data = df %>% filter(Premium ==1), mapping = aes(x= LineItem)) + geom_bar()

p1
```
~~~

So here we see that the line items are shown on the x-axis and

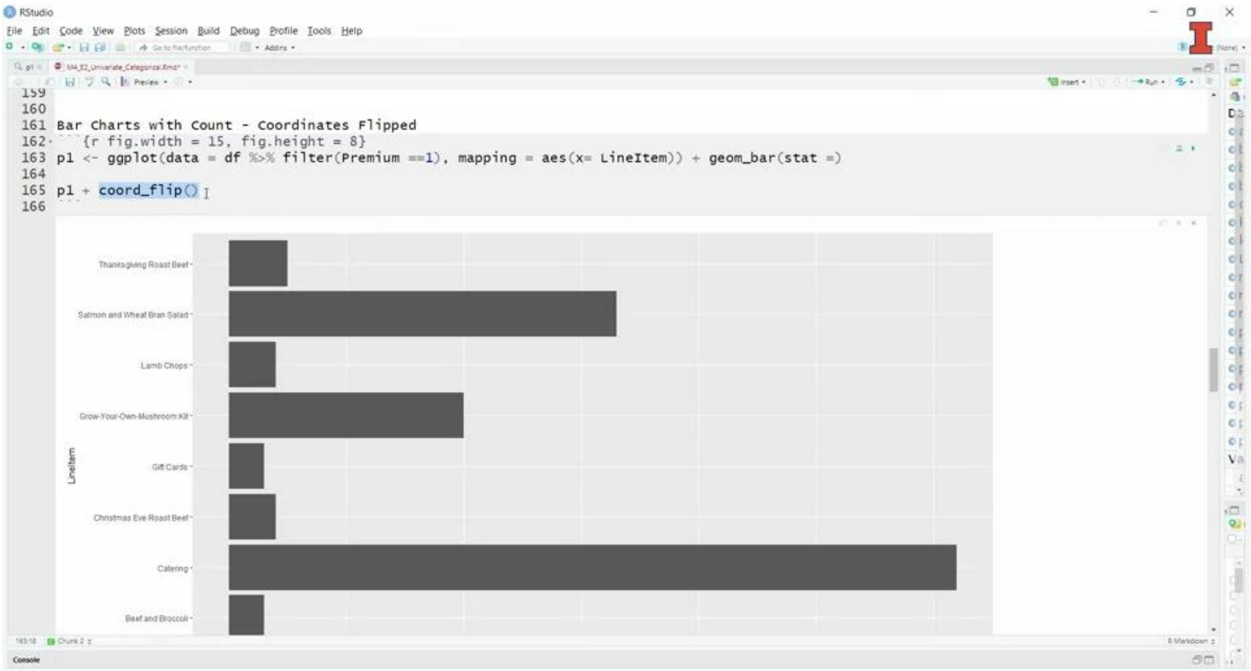
In fact, that's the advantage of working with Tidyverse packages, such as dplyr, tidyr, stringr, ggplot because the function in one of them allows us to use the functions that are available in the other packages. Let's just see, what I am doing here is creating visualization for line items only for Premium Products. And now if you see the code, that is slightly different from what you have been seeing so far. So here you see that in the data, I am not only specifying my data frame df, but also using this little dplyr function called filter. And this is where I am telling r to do this visualization only for the rows where premium is equal to 1. So as I said, you can do this subsetting on the fly within the ggplot function itself. And then in the mapping on the x-axis is going to come the line items. And the geometric object that I'm asking r to create is geom_bar bar chart.



```
p1
```

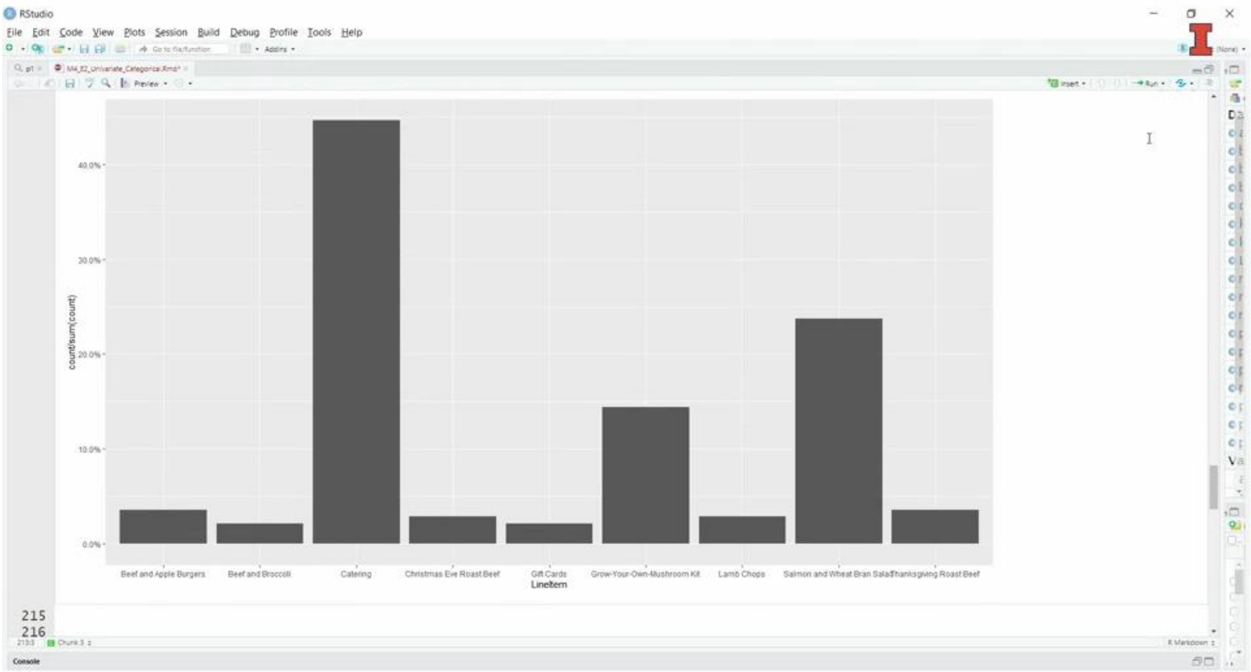
So here we see that the line items are shown on the x-axis and the y-axis captures the count of these variables. We can flip this and show the same information if that looks better.

And the way to do that would be using this small piece of code after you created the visual above.



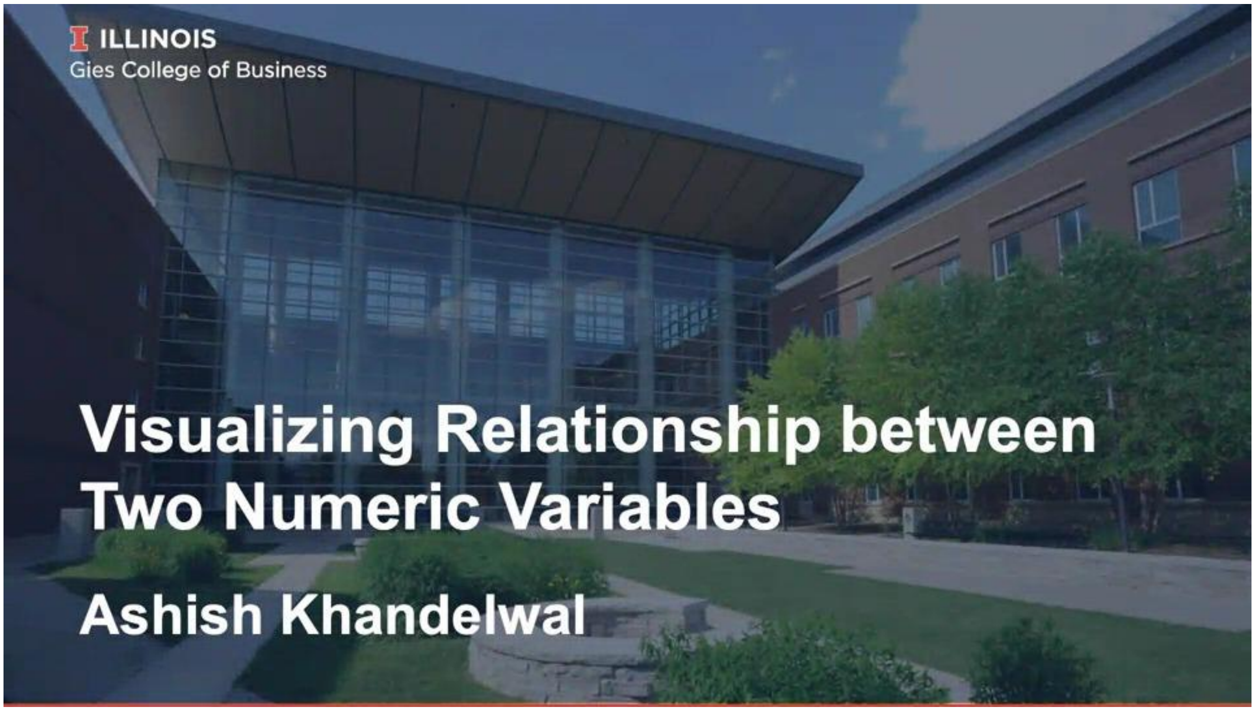
~~~
Bar Charts with Count - Coordinates Flipped
```{r fig.width = 15, fig.height = 8}
p1 <- ggplot(data = df %>% filter(Premium ==1), mapping = aes(x= LineItem)) + geom_bar(stat =)

p1 + coord_flip()
~~~

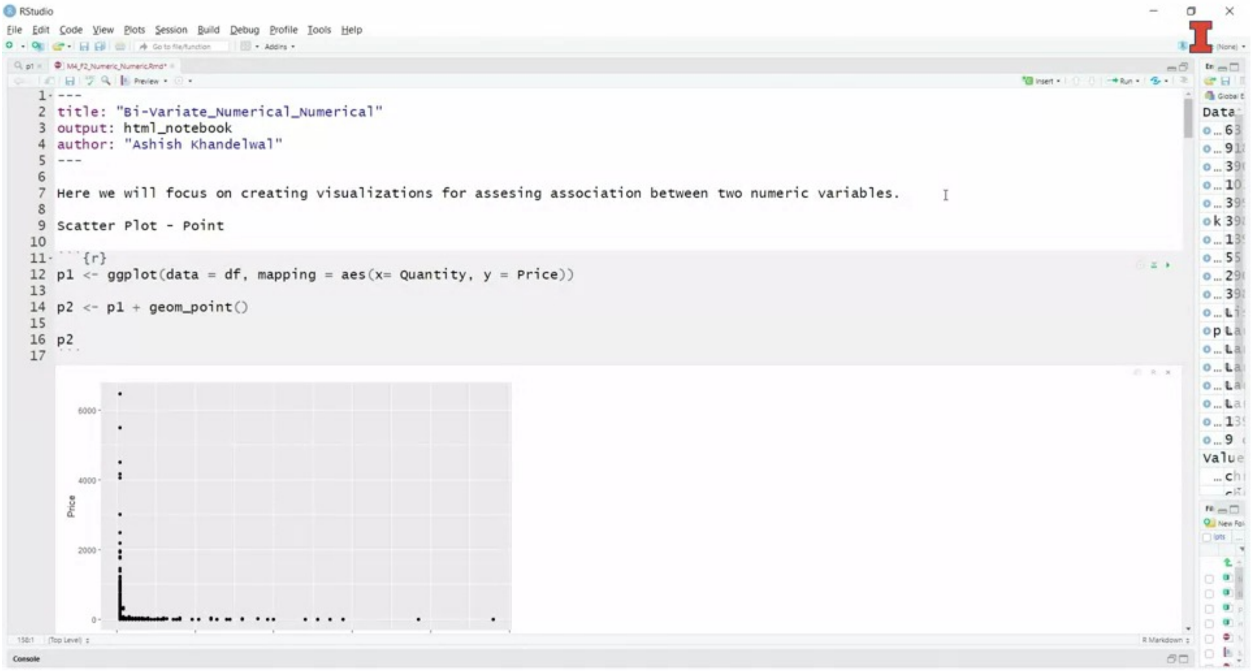
You can do the same thing and add this coord_flip, which is going to tell r to just read these coordinates. So everything same, but it's just now we see the line items on the y-axis and the counts on the x-axis. Now as mentioned earlier for a categorical variable, you can show either the count or the proportions. So here we see that we are creating bar charts with proportion instead of counts. And how do we get there?



So this part remains the same where I am just filtering my data set on the fly and then mapping aes (x= LineItem). But to get the proportions, I have to specify another aesthetic mapping, which is going to get the proportions. So I am mentioning that in the bars what I need, in the bars what I need is the counts divided by sum of counts. This will create the proportions. And then I'm also specifying that on the y-axis, a scale on the y scale, I need labels, which are in the percent format. And when I do this, I get this chart, again the same thing, except that instead of counts, we have got these proportions here.

Lesson 4-5.1: Visualizing Relationship between two Numeric Variables




RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
M4_F2_Numeric_Numeric.Rmd*
Preview
6000
4000
2000
0
Price

~~~
---
title: "Bi-Variate_Numerical_Numerical"
output: html_notebook
author: "Ashish Khandelwal"
---

Here we will focus on creating visualizations for assesing association between two numeric variables.

Scatter Plot - Point

```{r}
p1 <- ggplot(data = df, mapping = aes(x= Quantity, y = Price))

p2 <- p1 + geom_point()

p2
```
~~~

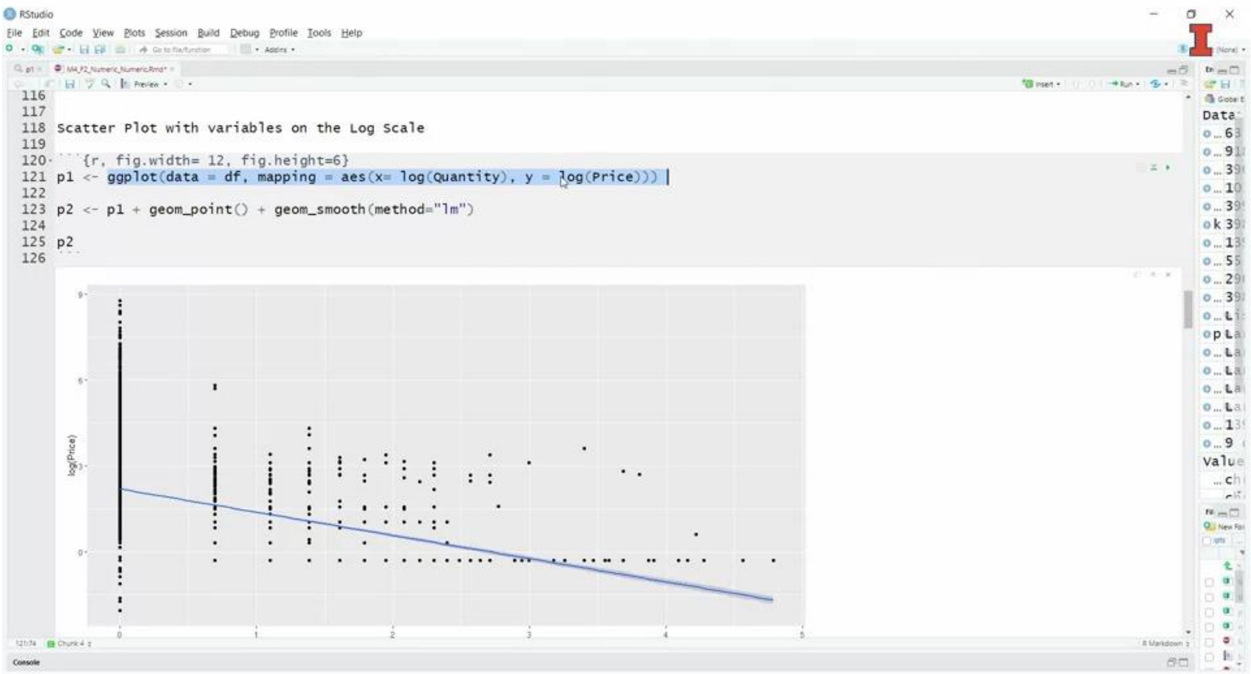
In this lesson, we will discuss how to visualize the relationship between two numerical variables. For example price and quantity.

In fact, first thing that we should do in our data when we have price and quantity to see whether law of demand holds. Law of demand states that as the price goes up the demand comes down for the product, make sense. Let's see, how much does it hold in our data.

So the way you will visualize the relationship between two numerical variables is through a scatter plot, which is also called as a point object because it just represents each observation as a point in two dimensional space. So here we create a scatter plot between price and quantity. The way we go about creating a scatter plot is just changing the geom object. Because everything else is the same that we did in histograms or

for any other geometric object data and the mapping are the same. So we are specifying which data frame do we want to use and which variables to have to go to which axis. And then that saved in P1 and then on top of P1, we are adding that we want to create a geometric object which is a point or a scatter plot. So when we do this we see this scatter plot. Now it's very difficult to really see much pattern because it looks almost like an l.

And you would appreciate that this is likely because we have outliers especially on the higher side for both price and quantity. And they are sort of hiding the pattern there. So can we improve upon it? Certainly, but before that let's understand one more concept. So when you have a scatter plot, you can also add a line to it. So what I do here in this code is, on top of the geom point, I add another geometric object, which is a smooth line, which is just geom_smooth. And when I do this, we see a line, but it's almost like flat on the horizontal axis. So maybe when we log transform our variables, we will see a nicer pattern. Let's just see it in the data. So now what I am going to do is I'm going to create a scatter plot with a line with variables transformed on the log scale.



~~~
Scatter Plot with variables on the Log Scale

```{r, fig.width= 12, fig.height=6}
p1 <- ggplot(data = df, mapping = aes(x= log(Quantity), y = log(Price)))

p2 <- p1 + geom_point() + geom_smooth(method="lm")

p2
```
~~~

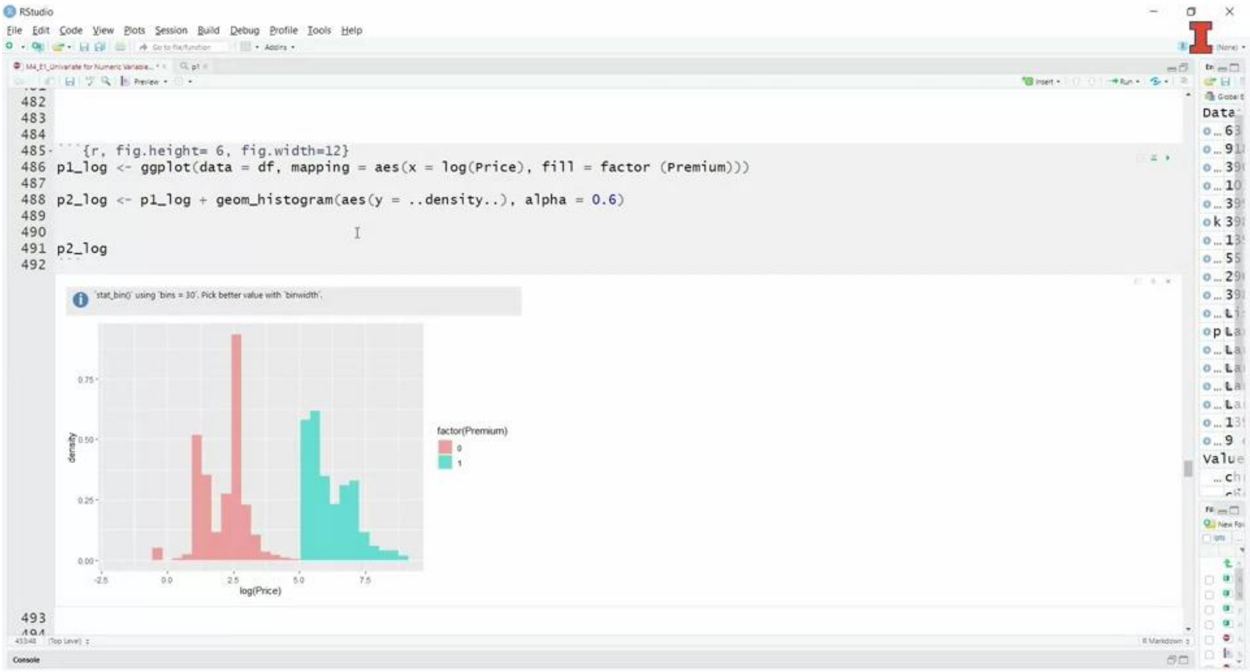
So everything is same here except that I've put the quantity and price with their log values, geom_point and geom_smooth. So point is for getting those points and smooth is for getting the line. And here I am telling r that this line should be a straight line, a linear model has to be used just to make sure that it's a line and not a curve. And when I create this, we now see a much clearer pattern that emerge here. So line going down as we expect. So what did we learn in this lesson? We learned how to visualize the relationship between two numeric variables. We used price and quantity from our data and we had to do some transformation on the data to really catch the pattern there.

Lesson 4-5.2: Visualizing Relationship between a Numeric and a Categorical Variable

ILLINOIS
Gies College of Business

Visualizing Relationship between a Numeric and a Categorical Variable

Ashish Khandelwal



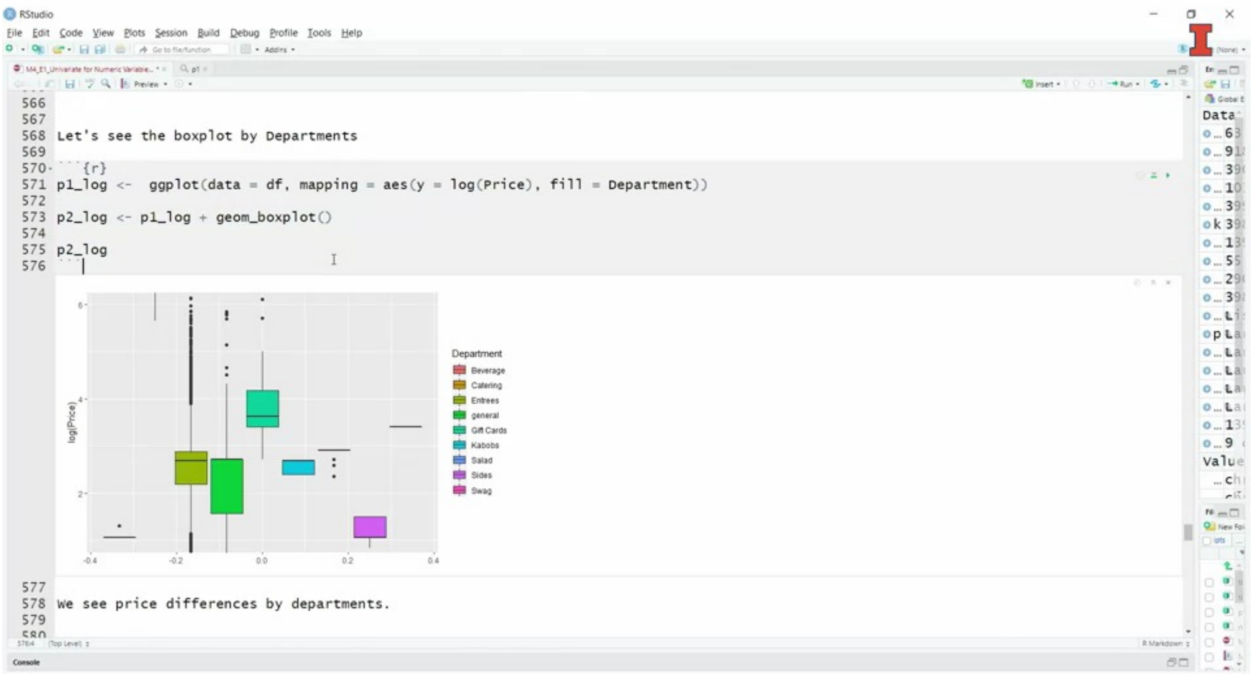
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
M4_E1_Univariate for Numeric Variable...
Preview
Console

~~~
```{r, fig.height= 6, fig.width=12}
p1_log <- ggplot(data = df, mapping = aes(x = log(Price), fill = factor (Premium)))

p2_log <- p1_log + geom_histogram(aes(y = ..density..), alpha = 0.6)


p2_log
```
~~~

'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.

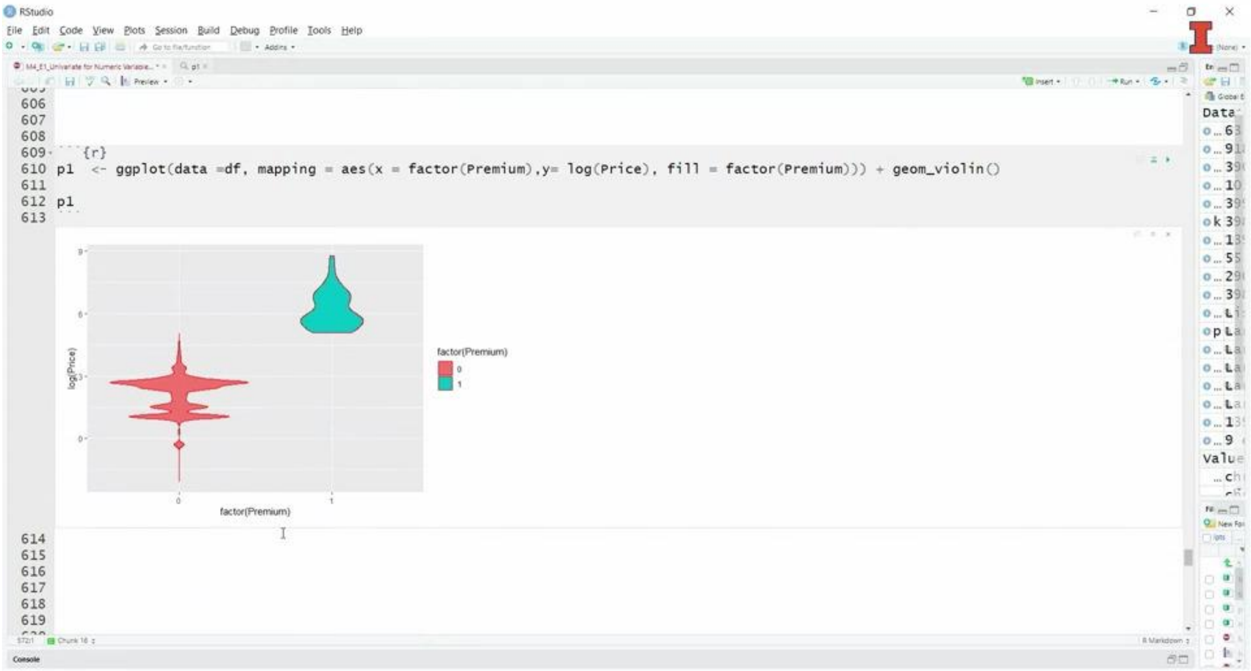
0.75
0.50
0.25
0.00
density
-2.5
0.0
2.5
5.0
7.5
log(Price)
factor(Premium)
0
1

In this lesson, we will discuss how do we relate two variables when one of them is numerical and another one is categorical. How do we visualize the relationship between them? For example, we may be interested in seeing how prices vary by department or by product type. So let's first see how prices vary depending on whether the product is premium or not. This is just to see how the code works. We created this variable called premium earlier, where if the prices are greater than mean plus two standard deviation, we were calling it premium, otherwise non-premium. I want to see how the distribution for two groups of products for prices look like. So essentially what we are seeing is the distribution of prices for these two products. The way I would go about doing this in R is using ggplot is, first, I need to create this first function, ggplot. I am specifying data as df, and then in the mapping I am specifying variable that is going to come on x-axis as the log of price. Here is what you are going to see, the new part in this lesson. I'm going to tell R to create separate histograms and fill them by different colors using this variable called premium, which we created using mutate function. The variable premium is saved as 0 and 1, and this fill function should take a factor variable or a categorical variable. That is why I'm calling this factor before it. So once I do this, this is going to be saved in my object called p1_log, and then I can create a histogram on this using density as the aesthetics and alpha just controls the transparency. You can play with it to see how it works. Then I do this, I see two different histograms for two types of items, premium and non-premium. It makes sense because the way the premium was created, they were of certain higher prices. That's why we see that the entire green distribution is on the right, while the distribution for the non-premium products is on the left.



~~~

Let's see the boxplot by Departments

```{r}
p1_log <-  ggplot(data = df, mapping = aes(y = log(Price), fill = Department))

p2_log <- p1_log + geom_boxplot()

p2_log
~~~

We see price differences by departments.
```

So now we create box plot by departments. Nothing changes in the code above, we did the same thing, except we uses geom_histogram. Here we are doing geom_boxplot. How will you interpret this? You can look at two things. One is all these horizontal bars detail what's the median price for a given department. As you can see, these bars are at different position on the y-axis, indicating that the median price for these departments are different. Secondly, you can also look at how far do they go on the y-axis, like from the bottom to the top. That indicates how much spread or variance is in the price.
```



```{r}
p1  <- ggplot(data =df, mapping = aes(x = factor(Premium),y= log(Price), fill = factor(Premium))) + geom_violin()

p1
```

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
log(Price)
9
6
3
0
factor(Premium)
0
1
factor(Premium)
0
1
Console

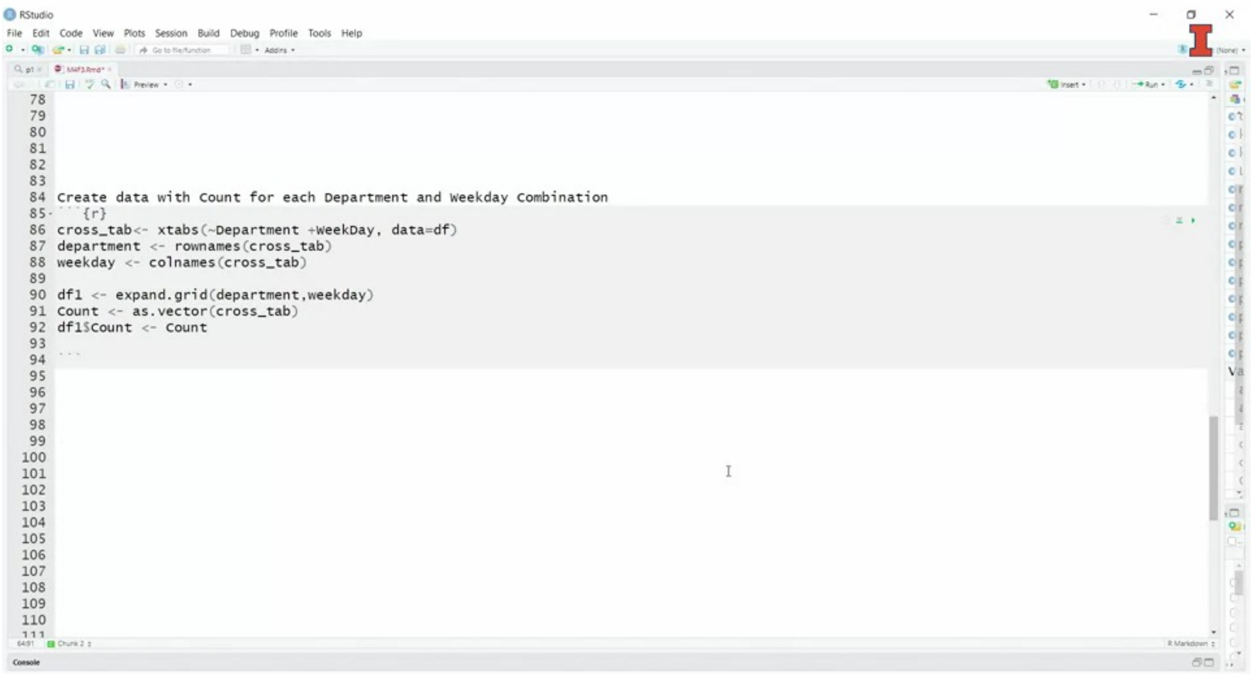
Now finally, we can also do y-line plot. Now, y-line plot is somewhere in-between a histogram and a box plot. So it shows density as a histogram will show, but it shows the values on the y-axis rather than x-axis. In fact, one thing that I want you to remember in this code is in the histogram, the variable that you want to see the distribution of goes on the x-axis, because the price is here on the x-axis. While in case of box plot, the price has to be mapped to the y-axis. In case of violin plots, again, the price has to be mapped on the y-axis. What comes on the x-axis is some grouping variable. As expected, you see that the premium products have higher values on prices. But another thing that you can see in the violin plot is the distribution at different price values, which was not very easily visible in the case of box plots. So let's say for non-premium products, there are many more observations around log price equals to three than at other values. So in this lesson, we learned how to create visualizations for a numeric variable based on another categorical variable.

Lesson 4-5.3: Visualizing Relationship between two Categorical Variables

I ILLINOIS
Gies College of Business

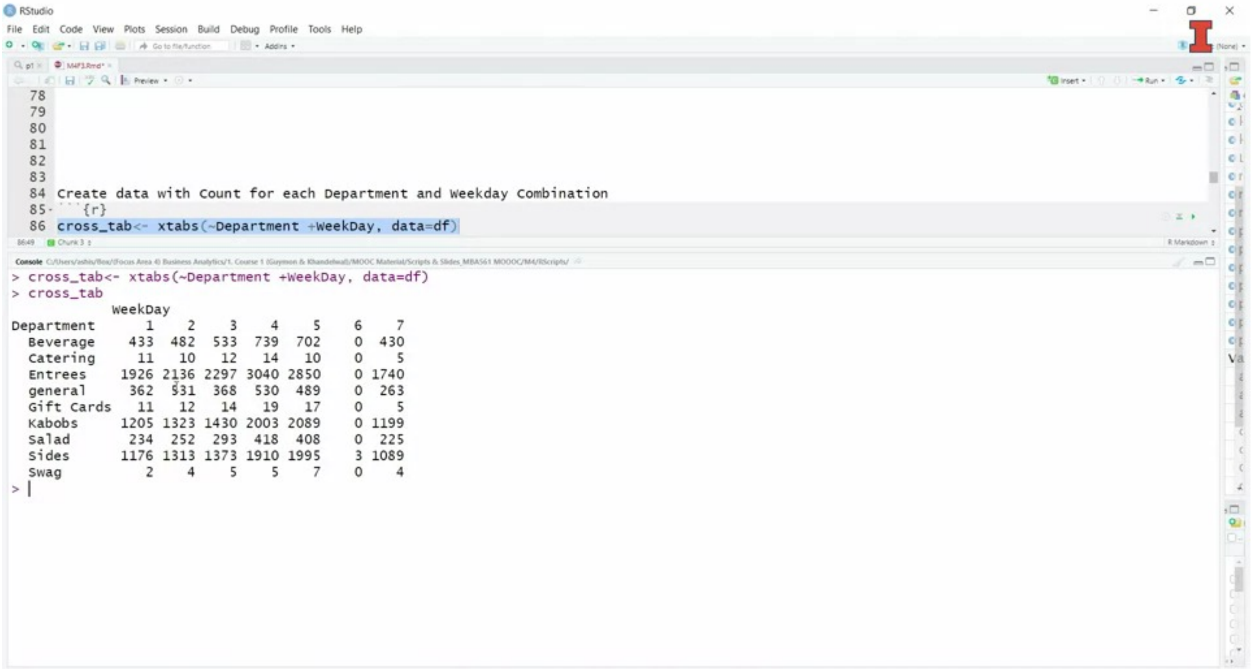
Visualizing Relationship between Two Categorical Variables

Ashish Khandelwal



~~~~
Create data with Count for each Department and Weekday Combination
```{r}
cross_tab<- xtabs(~Department +WeekDay, data=df)
department <- rownames(cross_tab)
weekday <- colnames(cross_tab)

df1 <- expand.grid(department,weekday)
Count <- as.vector(cross_tab)
df1$Count <- Count

```
~~~~

In this lesson, we will discuss how to visualize relationship between two categoric variables. So we have this transaction level data in our itemize.csv. And we want to see how many transactions are there for each department and weekday combination. So what I'm going to do is I am going to create a crosstab. This x tab creates a contingency table. Let's see this here. And I do this I get a crosstab, which shows for each weekday and department combination. So weekdays are in the columns and department is in the row. What are the number of observations that we have in our data?

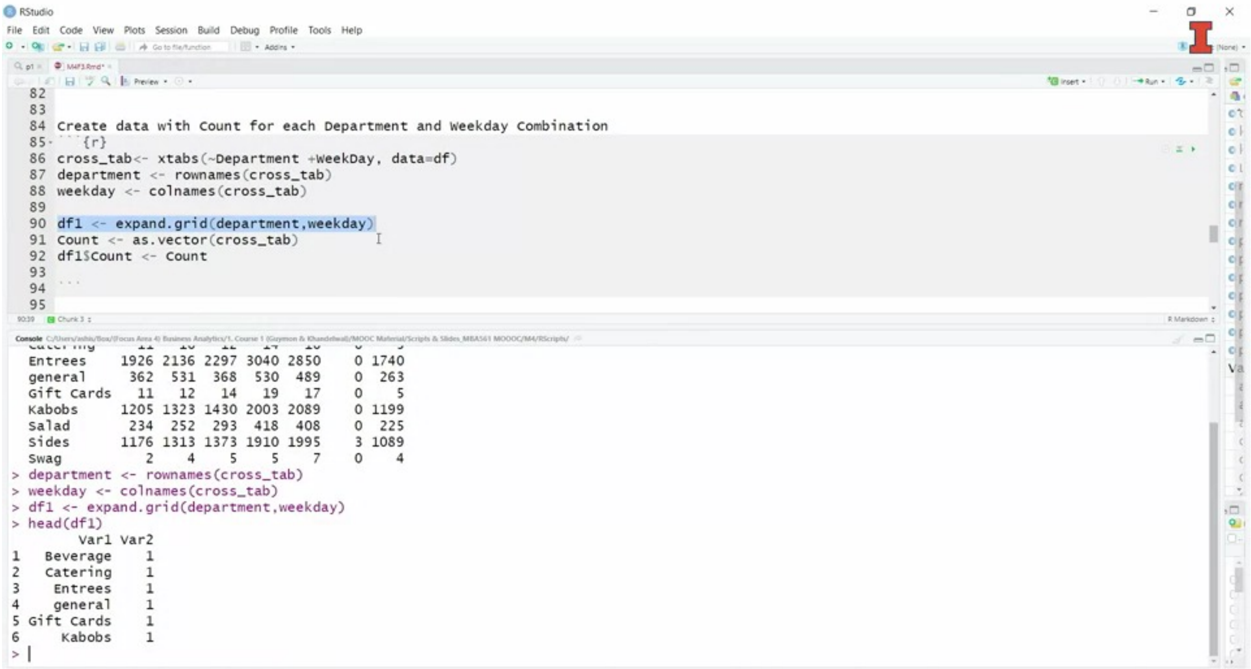


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
M4P3.Rmd*
Preview
Insert
Run
Chunk 3
R Markdown
Console

~~~
Create data with Count for each Department and Weekday Combination
```{r}
cross_tab<- xtabs(~Department +WeekDay, data=df)
~~~

```
> cross_tab<- xtabs(~Department +WeekDay, data=df)
> cross_tab
              WeekDay
Department        1    2    3    4    5    6    7
  Beverage      433  482  533  739  702    0  430
  Catering       11   10   12   14   10    0    5
  Entrees      1926 2136 2297 3040 2850    0 1740
  general       362  531  368  530  489    0  263
  Gift Cards     11   12   14   19   17    0    5
  Kabobs       1205 1323 1430 2003 2089    0 1199
  Salad         234  252  293  418  408    0  225
  Sides        1176 1313 1373 1910 1995    3 1089
  Swag            2    4    5    5    7    0    4
> 
```

So for example, on weekday one, which is set as a Monday, Beverages, there are 433 orders for beverages. While on Tuesday, the number of orders for beverages are 482. So we can interpret it this way, in fact for exploration, you can just use this crosstab itself to see what kind of relationship holds between beverage and weekday when it comes to the number of transactions. But if you want to create a visualization for it, what you can do is you can capture these variables. Let's say the row names are Departments, I'm capturing this here. And the weekdays are column names, so I am keeping them here. And then I'm creating this object called df1, which is nothing but an expanded grid of departments and weekdays. Let's see what you get here. It is just going to create columns, which are combinations of beverage and weekday. So when I do this and when we df1, and I'm going to just look at few of it.

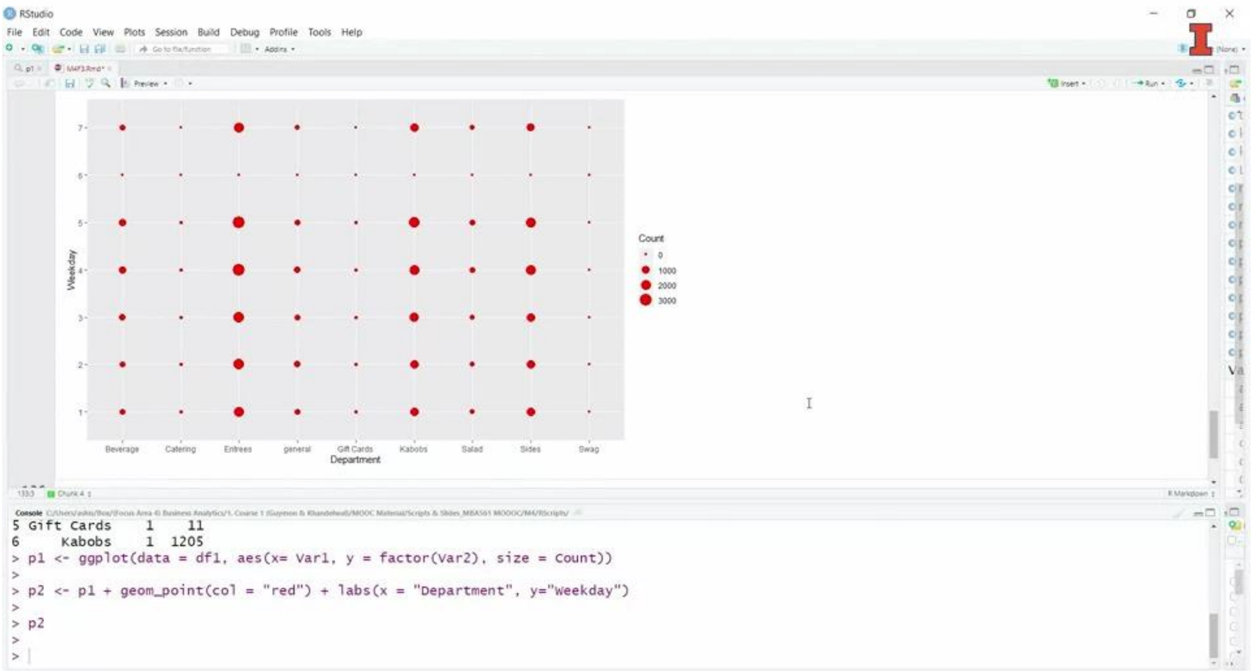

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
p1
M4F3.Rmd*
Preview
Insert
Run
Chunk 3
R Markdown
Console C:/Users/ashis/Box/(Focus Area 4) Business Analytics/1. Course 1 (Guymon & Khandelwal)/MOOC Material/Scripts & Slides_MBA561 MOOOC/M4/RScripts/

~~~
Create data with Count for each Department and Weekday Combination
```{r}
cross_tab<- xtabs(~Department +WeekDay, data=df)
department <- rownames(cross_tab)
weekday <- colnames(cross_tab)

df1 <- expand.grid(department,weekday)
Count <- as.vector(cross_tab)
df1$Count <- Count

```
~~~

```
  Entrees     1926 2136 2297 3040 2850    0 1740
  general      362  531  368  530  489    0  263
  Gift Cards    11   12   14   19   17    0    5
  Kabobs      1205 1323 1430 2003 2089    0 1199
  Salad        234  252  293  418  408    0  225
  Sides       1176 1313 1373 1910 1995    3 1089
  Swag           2    4    5    5    7    0    4
> department <- rownames(cross_tab)
> weekday <- colnames(cross_tab)
> df1 <- expand.grid(department,weekday)
> head(df1)
        Var1 Var2
1   Beverage    1
2   Catering    1
3    Entrees    1
4    general    1
5 Gift Cards    1
6     Kabobs    1
> |
```

It has got two variables, variable one and variable two. Variable one is department variable two is weekday. I can create another variable based on the values that I see here in this crosstab, and I can call it Count. So I create this and I put this Count as the third variable in my df1 dataframe. We have got three variables, var1, var2, and Count. var1 holds department names var2 holds the weekday and Count holds just the number of transactions for each combination. Now our data is ready for visualization. So the visualization that we are going to create here is a simple scatterplot.

But we are going to bring the information of the number of transactions through this aesthetic mapping called Count size. Okay, so what we are going to do here is we are creating a ggplot specifying data equals to df1. In the aesthetic mapping what goes on x axis is variable 1, which is our departments. On y-axis goes the weekdays, and remember that weekdays are saved as 1, 2, 3, 4, that's why we need to do a factor on this. And then the last mapping which is for the third variable, that is the Count, I am using this size. And when we do this the p1 will capture the data and the aesthetic mappings. p2, in p2 we are specifying what is the geometric object we want to create. And I am adding another option here for labeling my plot properly because x and y are labeled as var1 and var2. I have deliberately done that so that you can see that sometimes we need to change the labels and this is how we can change the labels.



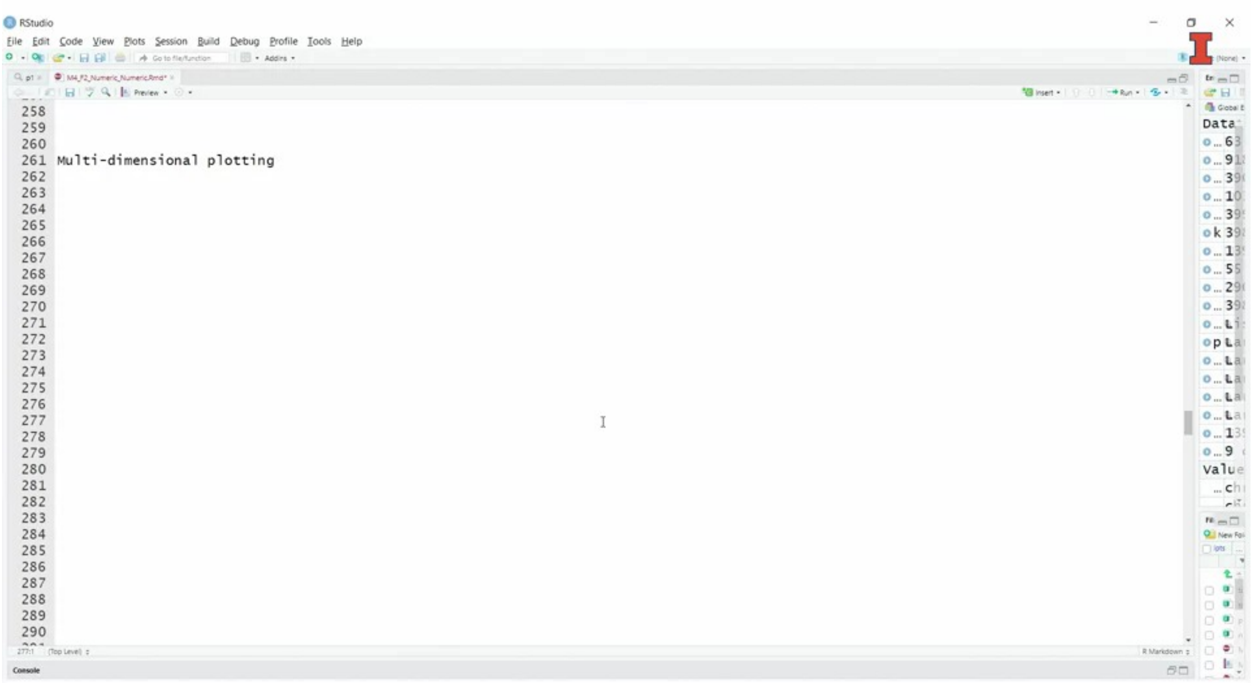
```
5 Gift Cards    1    11
6     Kabobs    1  1205
> p1 <- ggplot(data = df1, aes(x= Var1, y = factor(Var2), size = Count))
> 
> p2 <- p1 + geom_point(col = "red") + labs(x = "Department", y="Weekday")
> 
> p2
> 
> 
```

When I do this plot I get this nice-looking visualization. So what this visualization tells me is the number of transactions for each weekday and Department combination. The size captures the number of transactions. So in this lesson, we learned how to create visualization for two categorical variables. Remember, whenever we want to show two categorical variables, relationship among them, we need to bring some some other variable also.

Lesson 4-6: Multidimensional Charts

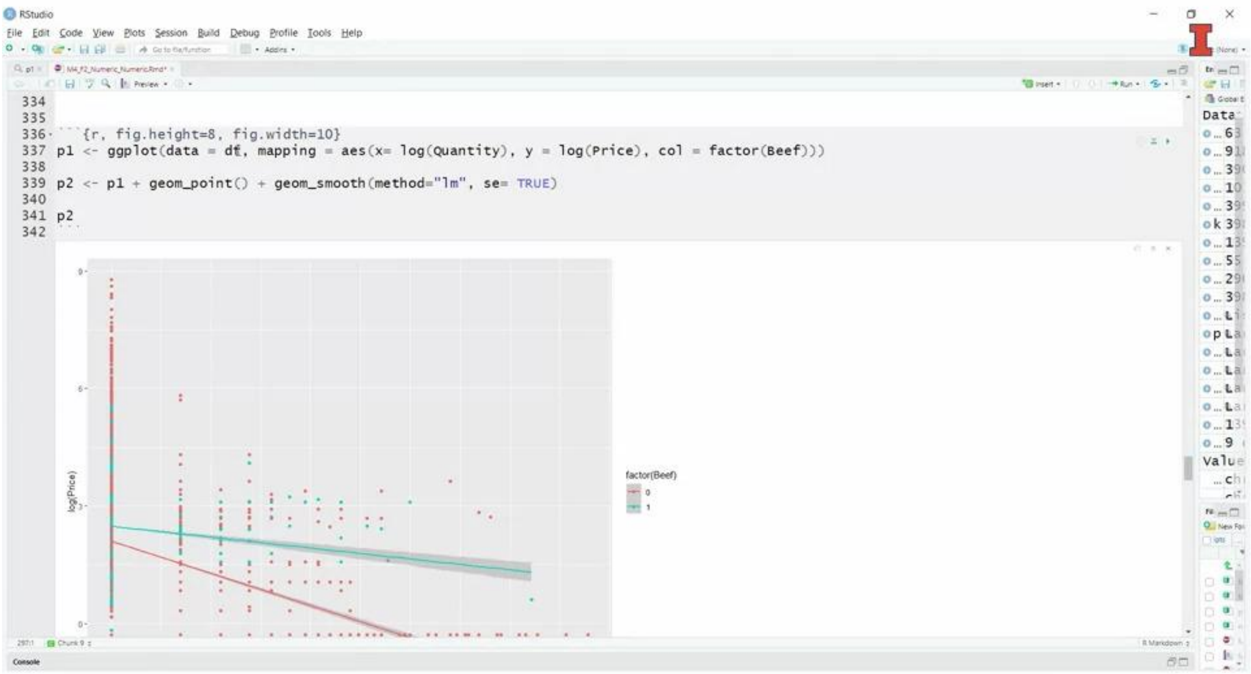
Multidimensional Charts

Ashish Khandelwal


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
M4_F2_Numeric_Numeric.Rmd*
Preview
Insert
Run
Data
Values
Console

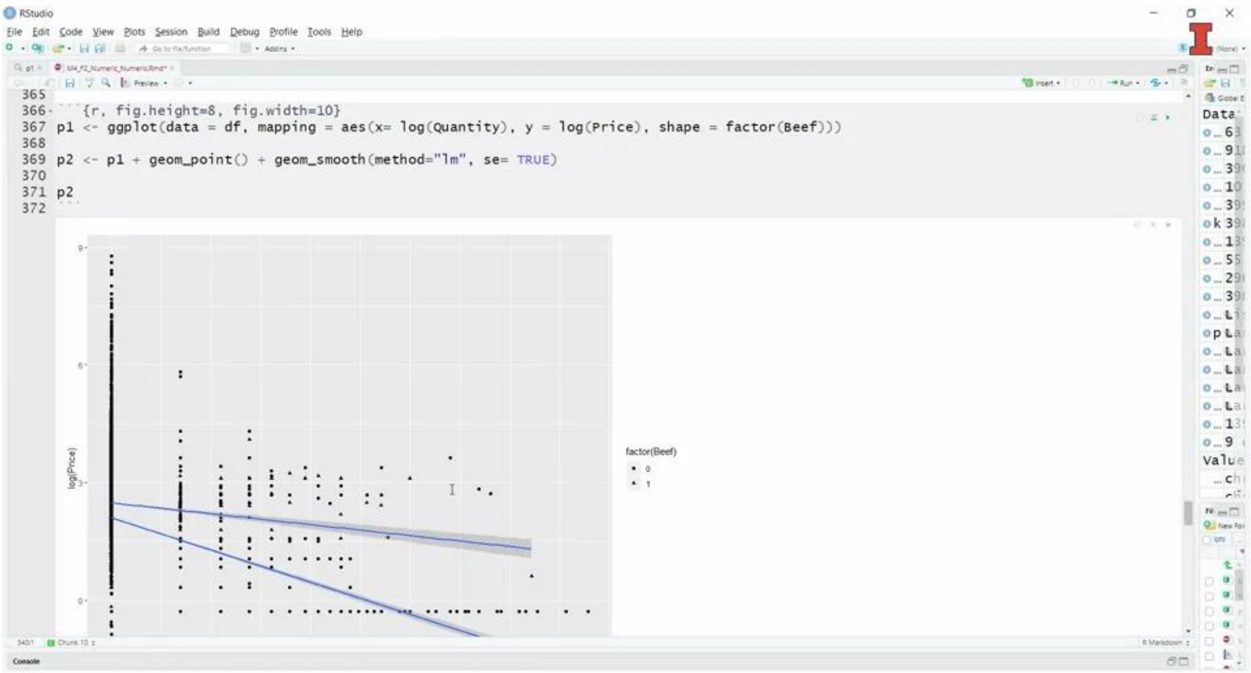
```
Multi-dimensional plotting
```

In this lesson, we will discuss how to create multidimensional plots. So we saw there are univariate exploration where we have just one variable then there are bivariate exploration where we have two variables. When you want to see relationship between more than two variables at a time, we have to rely on multidimensional plotting. One thing that I want you to remember is the variables go into aesthetic mapping. So we are going to get more and more variables maybe third, fourth or even fifth variable in our plot using options within the aesthetic mappings. If you remember this, it's going to be easier to understand the code. So let's play with multidimensional plotting. I have heard that people in Illinois have preference for beef. So I just did a Google search on what do people like to eat and beef actually came up as the first option. So I wanted to see in our data whether that holds, and one of the ways that I can look at whether people like beef or not is just by seeing how much beef is being ordered, but I wanted to also look at, are people more or less sensitive to products that have beef and not have beef in response to their prices. So you remember we discussed that the law of demand says that as price goes up or for the products that have high prices, the demand tends to be lower, and that's known as elasticity. Price elasticity which is sensitivity to changes in price. So I wanted to check whether people have more elasticity for beef versus other products. So what I do is I create another variable using the same mutate function that we saw earlier to see whether the beef is there in the name of the item or not. So let's quickly see the code here. So I go to my data frame, I use this pipe operator, then mutate and the new variable that I want to create is beef and the condition that I'm specifying here is that if you are able to detect beef in a line item, then call it one that means it has beef, as zero it does not have beef. Now we will see how the relationship between price and quantity that we saw earlier changes or differs based on whether the product has beef in it or not. For this, what we are going to do is we are going to have multiple variables in our plot. Now, let's quickly understand.



~~~
```{r, fig.height=8, fig.width=10}
p1 <- ggplot(data = df, mapping = aes(x= log(Quantity), y = log(Price), col = factor(Beef)))

p2 <- p1 + geom_point() + geom_smooth(method="lm", se= TRUE)

p2
```
~~~

So GG plot, I specify the data which is df. In the mappings, I say x is log of quantity and y is log of price. I'm adding another aesthetic mapping here which stands for color, and here I'm specifying that I want this variable beef to be captured with color because x and y-axis are already gone. The only options to show other variables are either using color or shape or size. So I'm using color here and I'll show you why it makes sense to use color here, and remember I need to do this factor because the way I created my variable beef, it's saved as zero and one which is numeric. So I need to use this color. I need to use factor before beef to tell that it has to be captured as a category, and then everything is same, I put this geom point to get the scatter plot. Price and quantity are all numerical variables, so the geometric object that we are going to create is going to be a scatter plot. But we will be able to capture the third variable using this color aesthetic, and then in the geom's smooth I'm telling that I need lines instead of curve and I'm saying that show me the standard error around the line. Standard error just captures the uncertainty, that's it, and by default it is true. So even if I did not specify it, it would not make any change in the code but I want it to be explicit about it. Let's see what we get. So we see two different lines here; one for beef products and another for non beef products. Green for beef and orange for non beef as you can see one and zero on the Legendre here. Please note that the price is the predictor variable and the quantity demanded is the response variable here. Usually in a scatter plot, we met the predictor variable to x-axis and the response variable to y-axis. However, in economics, to show the relationship between the price and quantity demanded, we map price to y-axis and the quantity demanded to x-axis. Thus, we can conclude that the green line has a steeper slope. This shows that the demand for beef products is more sensitive to price, opposite to what I'd anticipated. Now I quickly want to discuss how the choice of aesthetic mapping matters.



~~~
```{r, fig.height=8, fig.width=10}
p1 <- ggplot(data = df, mapping = aes(x= log(Quantity), y = log(Price), shape = factor(Beef)))

p2 <- p1 + geom_point() + geom_smooth(method="lm", se= TRUE)

p2
```
~~~

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Addins
M4_P2_Numeric_Numeric.Rmd*
Preview
Run
log(Price)
9
6
3
0
factor(Beef)
0
1
Data
Value
Console

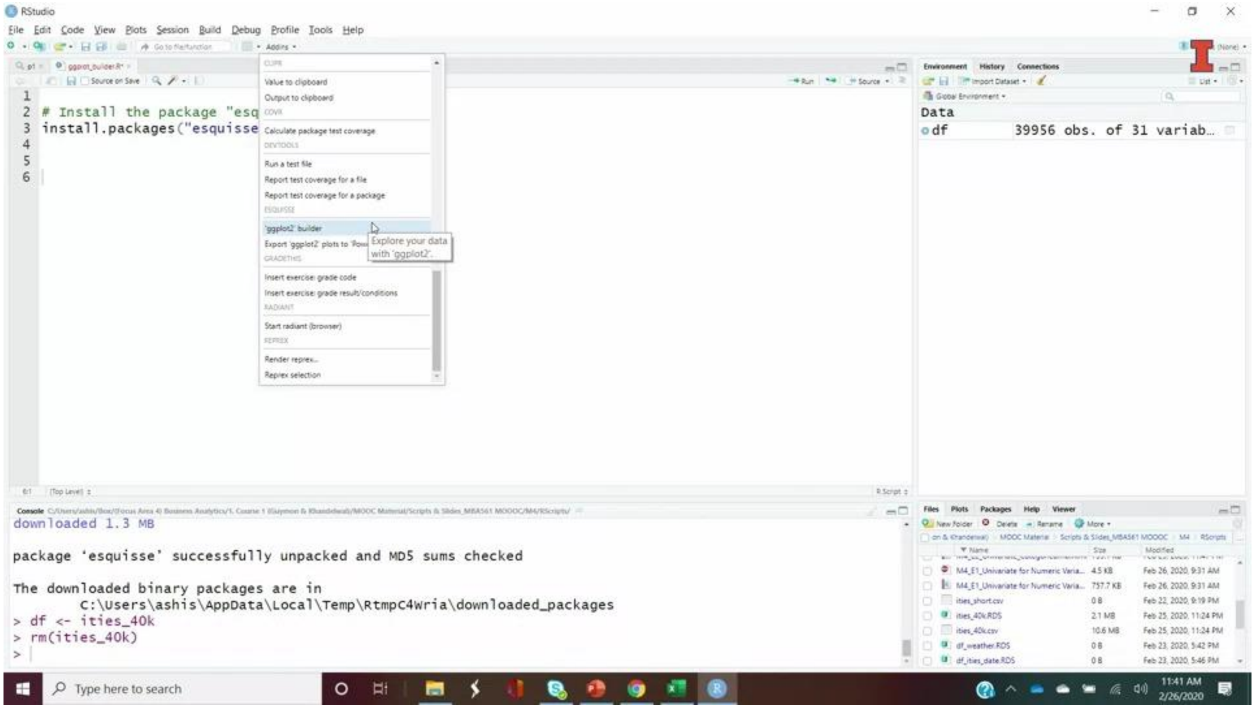
So we used color to show the variable beef, what if we show beef through shapes instead of color, and nothing change from the above code except instead of color equal factor beef, I'm doing shape equal factor beef, and when I run this, what I find is the same thing. Everything is same except that it's difficult for me to now see which line refers to beef and which corresponds to non beef items. So we can of course use either shape or size or color to bring more variables on our plot, but it's always a good idea to see which is more clear in representing what we want to see.

Lesson 4-7: Introduction to ggplot Builder

ILLINOIS Gies College of Business

Introduction to Ggplot Builder

Ashish Khandelwal



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
ggplot_builder.R*
Source on Save
CLIPR
Value to clipboard
Output to clipboard
COVR
Calculate package test coverage
DEVTOOLS
Run a test file
Report test coverage for a file
Report test coverage for a package
ESQUISSE
'ggplot2' builder
Export 'ggplot2' plots to 'Pow
Explore your data with 'ggplot2'.
GRADETHIS
Insert exercise: grade code
Insert exercise: grade result/conditions
RADIANT
Start radiant (browser)
REPREX
Render reprex...
Reprex selection
Environment History Connections
Import Dataset
Global Environment
Files Plots Packages Help Viewer
Console
Type here to search
11:41 AM
2/26/2020

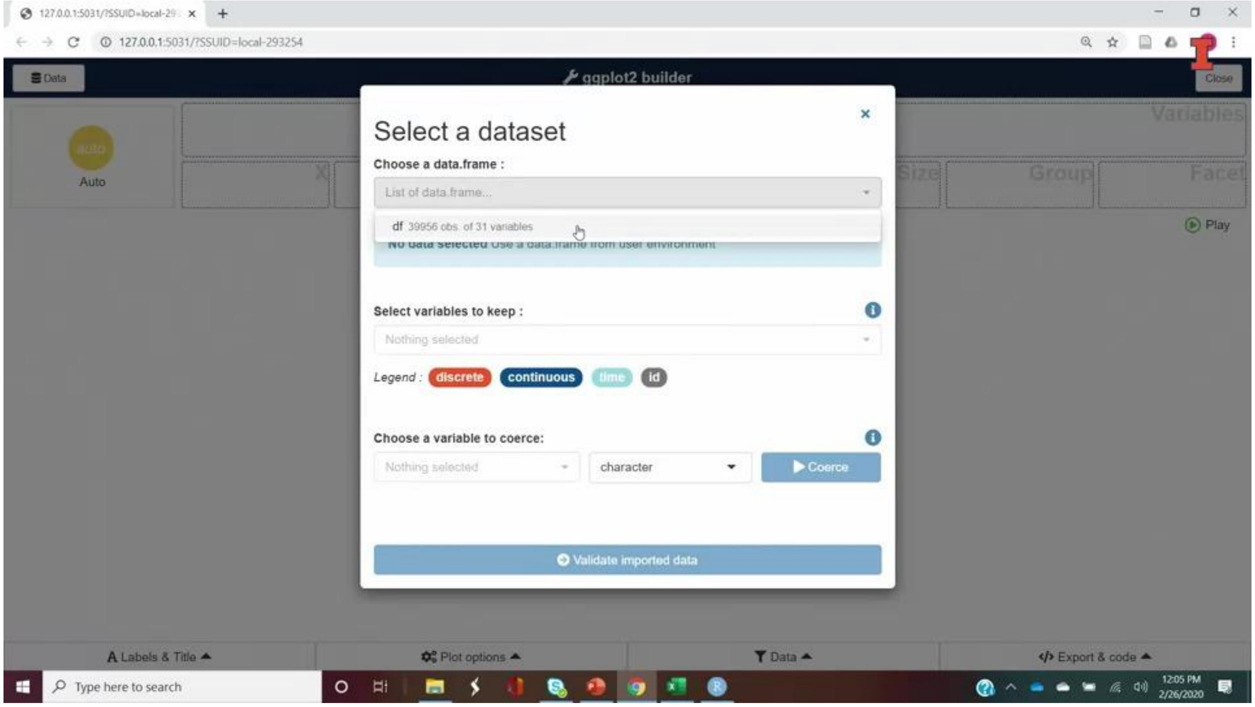
```

# Install the package "esq
install.packages("esquisse



```

Data
df 39956 obs. of 31 variab…

```
downloaded 1.3 MB

package ‘esquisse’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\ashis\AppData\Local\Temp\RtmpC4Wria\downloaded_packages
> df <- ities_40k
> rm(ities_40k)
> 
```

So far, we have seen creating different visualizations through ggplot2 package, and we realize that it's not easy to remember those commands. And that's why as promised we are going to share a tool which is going to allow you to create those graphics without writing codes. But remember, when you move to something which does not require you to write codes the flexibility is going to be less. But for your initial plotting tasks, I think it would just be fine. The package that you need for building ggplots without code is known as esquisse. It's a French word, so you need to install the package. I have already installed it here. So I am directly going to take you to where you start working with this. So you go to Add in and Add in is a very nice feature in R Studio now and it allows us to add lot more functionalities to R without coding. And I use many of those add-ins and one of them is this ggplot2 builder, that actually comes when you install the package esquisse. A few things before we get there. You need to have your data loaded in your environment variable.

So I have already loaded a subset of the, data in the data frame called DF. Once I have it there, I go to Add in, then GGplot Builder.



Select a dataset

Choose a data.frame : List of data.frame...

df 39956 obs. of 31 variables

No data selected Use a data.frame from user environment

Select variables to keep : Nothing selected

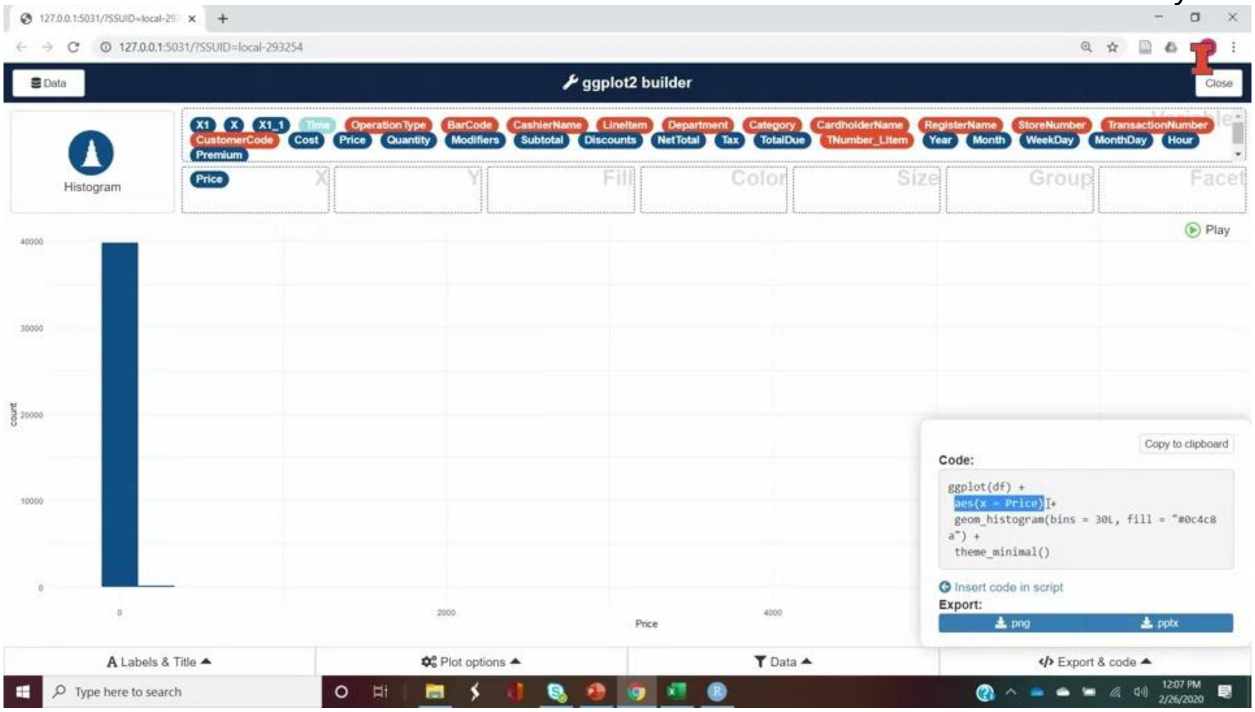
Legend : discrete continuous time id

Choose a variable to coerce: Nothing selected character

So once we are here, first thing that we need to do is choose of a data frame. This is the first object of ggplot2.

Now you can make some selection if you want to make some changes in your variable types. The best way is change them before you bring here because every time you make a change here, it's not recorded in the data. So once you are out of this builder, the changes go away. Right now I'm just going to go as we have the data. So we only have to worry about like top three layers. The first one is the data, second one is the aesthetic mapping, and the third one is the geometric object we want to create. And this is where ggplot builder helps us a lot, because it just forced us to read the data. That means the data part is already taken care of. Now we have all these variables sitting at the top and all these options x-axis, y-axis, fill, color, size group, and facet. They are going to allow us to do the aesthetic mapping part.

And the options here are the geometric objects. So it's so easy, if I have to let's say make a histogram of price.



Copy to clipboard

Code:

```
ggplot(df) +
 aes(x = Price) +
 geom_histogram(bins = 30L, fill = "#0c4c8
a") +
 theme_minimal()
```

Insert code in script

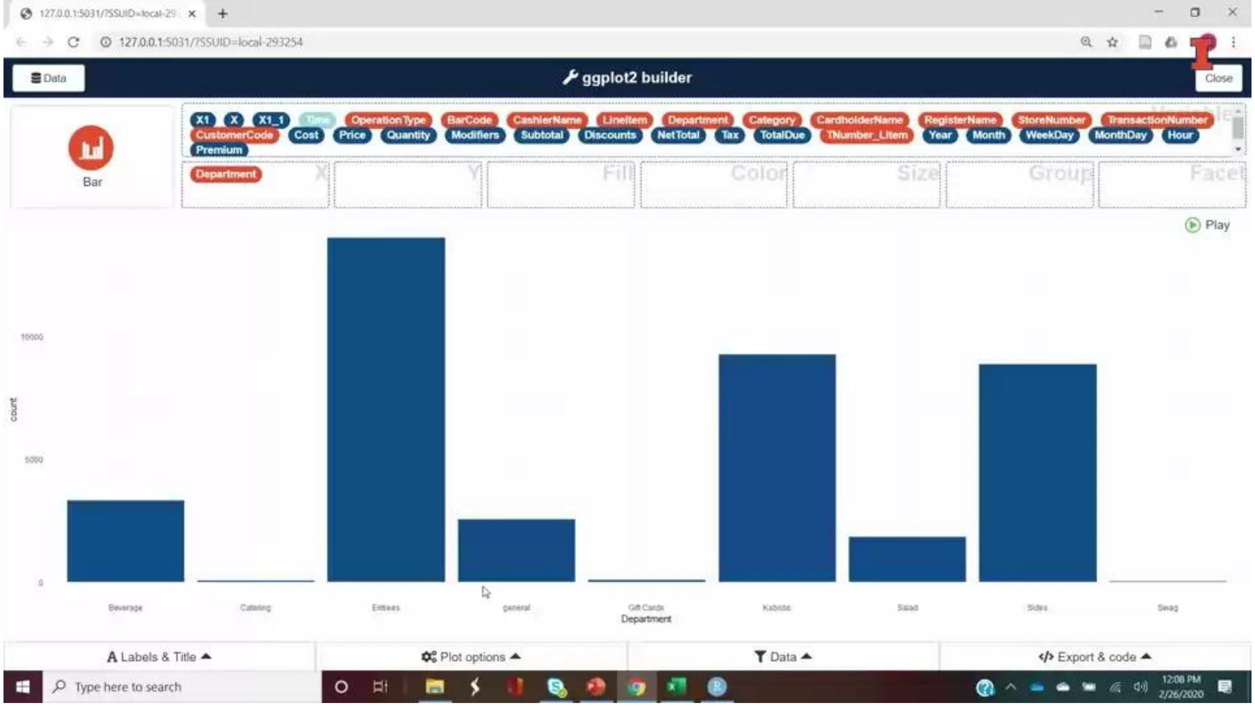
Export:

png | pptx

What I can simply do is data is already there, I need to specify the aesthetic mapping and I just drag price here. When I do this, let's just look at the code. What we have got in the code is ggplot has got the data. In the aesthetic mapping I have got the price and it has automatically selected histogram and certain options in that. You can always copy this code which I would suggest you do, copy this code and take it to your R and run it from there, so that you can recreate these plots again without coming here.

It has automatically selected histogram, if you want to change. So things that are not allowed for this type of variables will be shown like this. We see that we can also create box plot and I create a box plot here and it will looks exactly similar to what we had earlier.

And if you see the code has changed from geom_hist, now he geometric object has changed to geometric underscore box plot. If I bring a categorical variable, let's say department.

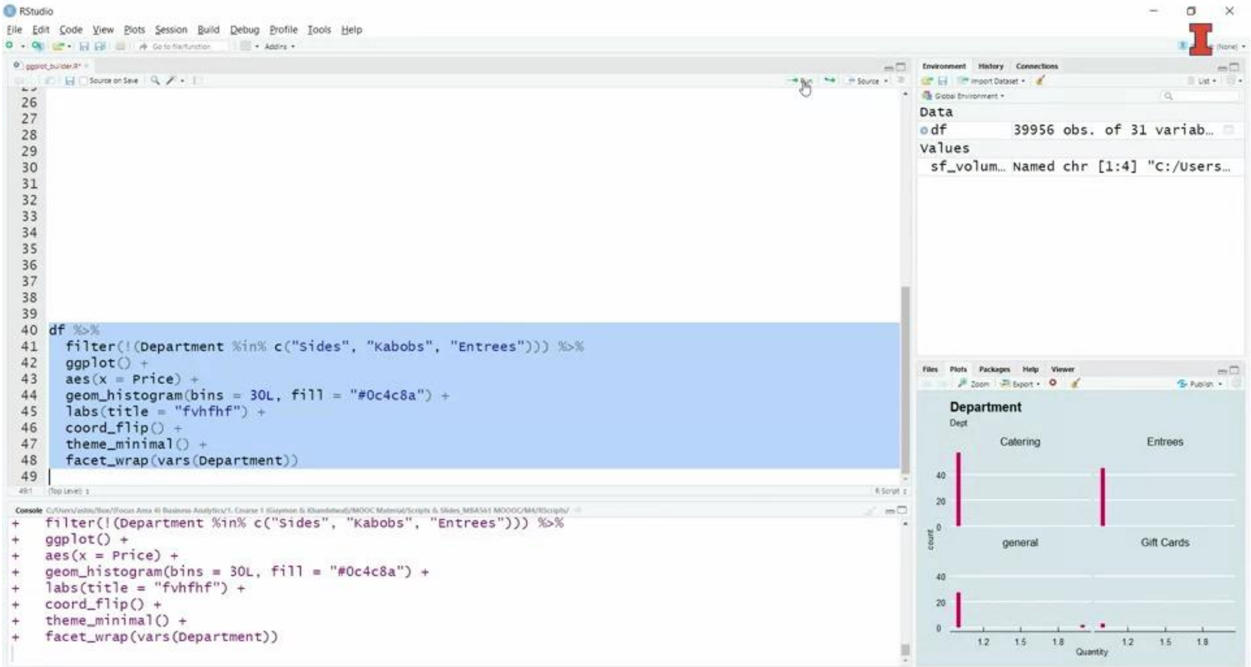


What I see is a bar chart which is the standard graphic for a categorical variable. If you go and see the code, you will see the difference. You know now the geometric object is geom_bar and aes, aesthetic mapping is x equals to department. If you try to do a histogram it will not allow you, make sense. Now, there were other layers also that we really didn't go into much detail such as labeling, such as themes. You can take care of these things here, I would recommend that you try these options like typing something and see where exactly does it come. So the title comes here.

There are these plot options which allow you to either flip your coordinate or do something like choosing themes, try to learn better.

And then the last thing here to see is the data. So you can always subset your data right on the fly here. So if you only want to focus on let's say like there are these nine departments. If you want to focus only on just couple of them just remove the other and it allows you to do that. And when you go to the code, you will see that this filtering is going to happen here.

I can always bring this facet to create charts for separate subsets of data. If I am showing the histogram for price, I can always bring a categorical variable in the facet to create different histograms for each department.



```
df %>%
  filter(!(Department %in% c("Sides", "Kabobs", "Entrees"))) %>%
  ggplot() +
  aes(x = Price) +
  geom_histogram(bins = 30L, fill = "#0c4c8a") +
  labs(title = "fvhfhf") +
  coord_flip() +
  theme_minimal() +
  facet_wrap(vars(Department))
```

```
Data
df          39956 obs. of 31 variab…
Values
 sf_volum… Named chr [1:4] "C:/Users…
```

```
+   filter(!(Department %in% c("Sides", "Kabobs", "Entrees"))) %>%
+   ggplot() +
+   aes(x = Price) +
+   geom_histogram(bins = 30L, fill = "#0c4c8a") +
+   labs(title = "fvhfhf") +
+   coord_flip() +
+   theme_minimal() +
+   facet_wrap(vars(Department))
```

Lastly, don't forget to look at the code every time you run the command and copy the command to your clipboard and take it to R, take it to r to run this code there and you should be able to reproduce the same graphics. Here you see, so hopefully you appreciate that this ggplot builder allows you to create nice ggplot graphics without writing the code. But my suggestion would be to play with ggplot builder to get better at writing ggplot2 codes. Gradually, you should move to writing your codes to really enjoy the flexibility that ggplot2 package offers.

Module 4 Conclusion

Welcome to the end of module 4. In this module, we learn the ggplot2 package to create visualizations to explore our data. Remember there are various other visualization that we didn't cover, but once you get comfortable with the basic plotting, you can always learn to plot more complex visualizations. Hopefully, you'll use your learning in this module to explore your data before you take it for former modeling.

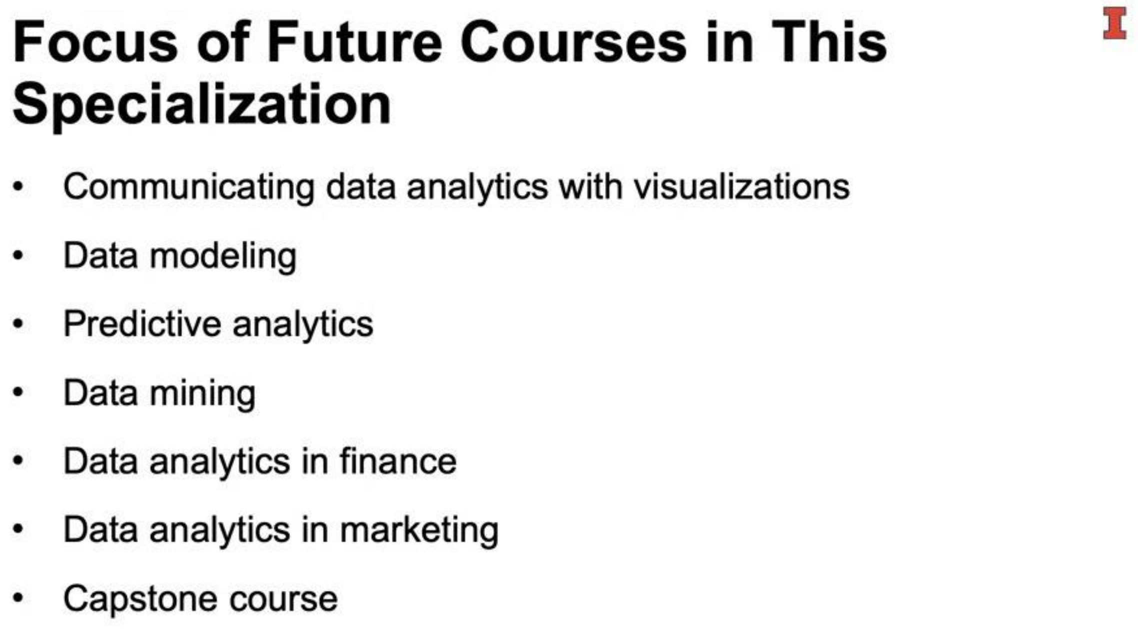
Course Conclusion

ILLINOIS
Gies College of Business

Course Conclusion

Ron Guymon & Ashish Khandelwal

We hope you enjoyed the course on business analytics and you are excited to continue learning more about how you can generate business value through analytics. If you're interested in learning more, then here's what you can expect from future courses in this specialization. The next course, we'll go into more detail on how to use visualizations to communicate data analytic findings.



Focus of Future Courses in This Specialization

- Communicating data analytics with visualizations
- Data modeling
- Predictive analytics
- Data mining
- Data analytics in finance
- Data analytics in marketing
- Capstone course

You will then learn how to create models and use these models to make predictions. You will also learn about data mining. The next courses will focus on applying the data analytics tools that you have learned to finance and marketing settings. The final course in this specialization is a capstone course. All these courses use R. So the skills that you have gained in the course should provide you with a solid foundation for moving forward in future courses. Now, whether you continue on with other courses in this specialization or not, we hope for two things. First, we hope that you are better equipped to guide your company on how to use data to improve business decisions. Second, we're excited about data analytics, and we know that it has useful applications to nearly every aspect of life. This is just the beginning. So we hope that we've at least sparked within you a desire to continue learning about data analytics in one way or another.