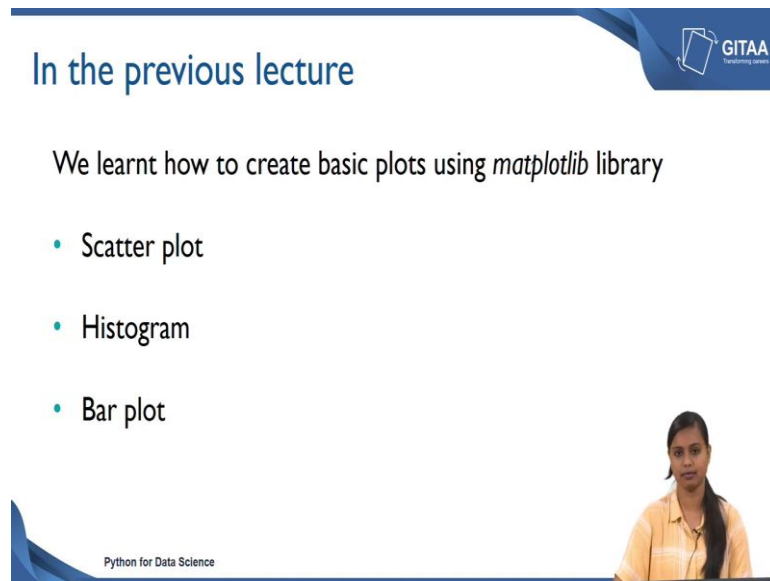


**Python for Data Science**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Madras**

**Lecture - 24**  
**Data visualization Part – II**

Welcome to the lecture on Data Visualization.

(Refer Slide Time: 00:17)



**In the previous lecture**

We learnt how to create basic plots using *matplotlib* library

- Scatter plot
- Histogram
- Bar plot

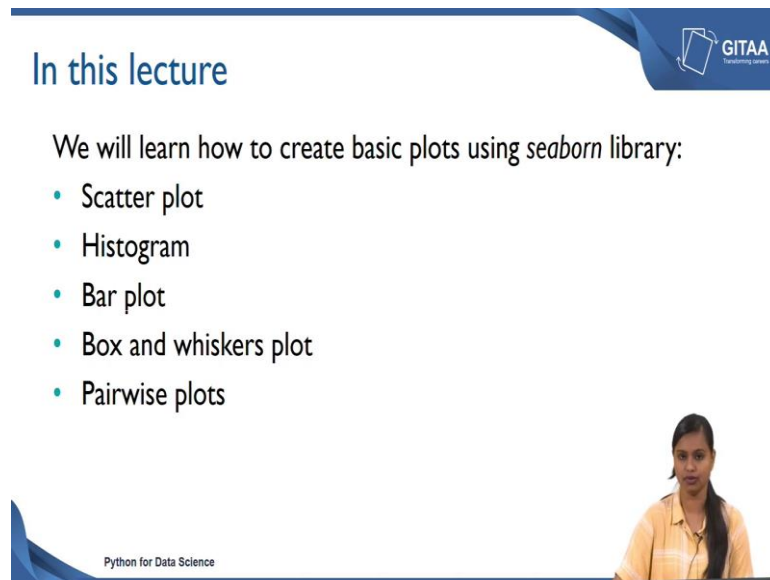
Python for Data Science

GITAA  
Transforming careers

A video inset in the bottom right corner shows a woman with dark hair, wearing a yellow and white checkered shirt, speaking.

In the previous lecture we have been looking about how to create basic plots using *matplotlib* library. The basic plots include scatter plot, histogram and bar plot. In that lecture we have seen how to create these plots and we have also seen what *matplotlib* library is about and how to interpret each of these plots.

(Refer Slide Time: 00:39)



**In this lecture**

We will learn how to create basic plots using *seaborn* library:

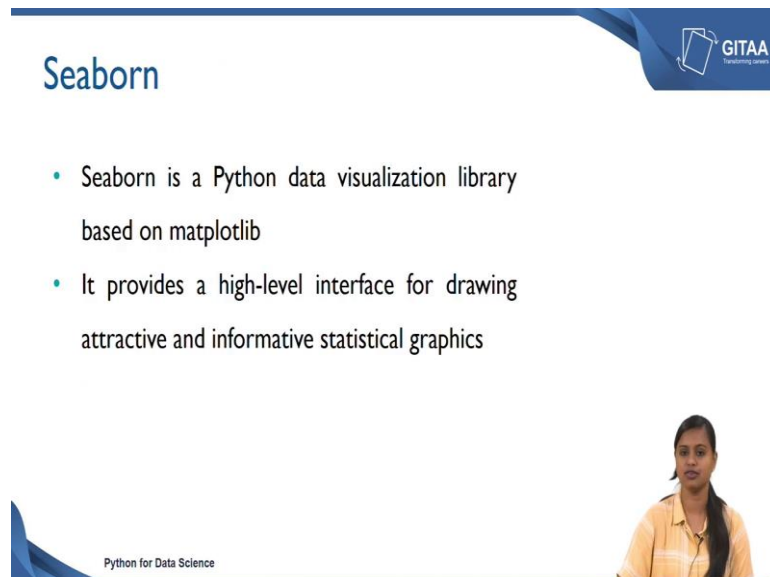
- Scatter plot
- Histogram
- Bar plot
- Box and whiskers plot
- Pairwise plots

Python for Data Science

GITAA  
Teaching Assistants

And in this lecture we are going to see how to create basic plots using seaborn library and the basic plots includes scatter plot, histogram, bar plot, box and whiskers plot and pair wise plots. We will see in detail about the creation of these plots and how to interpret them.

(Refer Slide Time: 00:57)



**Seaborn**

- Seaborn is a Python data visualization library based on matplotlib
- It provides a high-level interface for drawing attractive and informative statistical graphics

Python for Data Science

GITAA  
Teaching Assistants

So, first let us see what seaborn library is about; seaborn library is a Python data visualization library which was built on top of matplotlib library that provides a high

level interface for drawing attractive and informative statistical graphs. So, using seaborn library we are going to first see how to create this scatter plot.

(Refer Slide Time: 01:17)

The slide is titled "Importing libraries" and features the GITAA logo in the top right corner. It lists four libraries with their respective import statements and purposes:

- `import pandas as pd` — 'pandas' library to work with dataframes
- `import numpy as np` — 'numpy' library to do numerical operations
- `import matplotlib.pyplot as plt` — 'matplotlib' library to do visualization
- `import seaborn as sns` — 'seaborn' library to do visualization

A small video inset in the bottom right corner shows a woman in a yellow shirt speaking. The bottom left corner of the slide contains the text "Python for Data Science".

So, before creating the scatter plots we need to import the basic libraries that are needed for visualization. So, let us import the necessary libraries. We are importing the pandas with alias pd because to do the visualization we need to have a data frame, here we are going to create plots using a data frame. Since we are going to deal with data frames we are importing the pandas library, if you if we want to if we wanted to do some numerical operations on it then we need a numpy library.

So, let us import the numpy as np as well, then we can import the libraries that are needed for visualization. Here even though we are going to create plots using the seaborn library we are also importing the pyplot from mat plot library with the alias plt because as we know that seaborn library has been built on top of the matplotlib library. We will be using some of the functionalities that are from the matplotlib library in that case we would be needing that library.

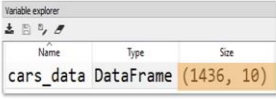
So, let us import matplotlib as well followed by that we can import the seaborn library as sns. Here sns will be realized for the seaborn library and matplotlib and seaborn will be used for data visualization.

(Refer Slide Time: 02:33)

## Importing data into Spyder

- Importing data

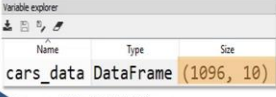
```
cars_data = pd.read_csv('Toyota.csv', index_col=0,  
                        na_values=["??", "???"])
```




Name	Type	Size
cars_data	DataFrame	(1436, 10)

- Removing missing values from the dataframe

```
cars_data.dropna(axis = 0, inplace=True)
```



Name	Type	Size
cars_data	DataFrame	(1096, 10)



So, now we have imported the necessary libraries into spyder. Now, its time to import the data that will be used for the visualization because, as we know we do visualization to understand the data even more better. So, on that note we are going to import a data into spyder to understand the data visually.

So, we are going to import the Toyota data which we have already worked on since it is of csv format I have used `pd.read_csv` and while reading itself I have considered all the missing values as the default nan values and I have ceded to an object calls `cars_data`, now `cars_data` is a `DataFrame`. When we looked at the size of the `DataFrame` through the variable explorer you can see that it has 1436 observations with 10 columns and now we are very familiar with what Toyota data is about.


They are just the details of the cars. So, since our objective is to understand the data pictorially or visually, we need to consider only the rows wherever the records are filled because we would not be able to interpret anything from the question marks or from the nan values. In this case our objective is to basically understand the data and understand the relationship that exists between each of the variables. In that case we wanted to remove missing values, so that we will focus on understanding the data which have complete informations from the `DataFrame`.

So, on that note we can just remove all the missing values from the `DataFrame` and then proceed with the visualization. So, let us remove the missing values from the `DataFrame`

by using the command `dropna` and inside the function you can just say `axis` is equal to 0. So, that all the rows will be removed wherever there are nan values and if you use `inplace` is equal to `true` you are making all the modifications in the data frame that is being used here. Otherwise you have to reassign it to a new data frame called `cars_data2` or you can update it in the existing data frame itself.

So, here I have updated while I am using the function itself by using `inplace` is equal to `True` and after dropping all the missing values if you look at the size of the data frame it is turned out to be 1096 observations with 10 columns on the whole we have removed around 340 observations from the DataFrame and we are going to proceed with the analysis only with the 1096 observations with 10 columns.

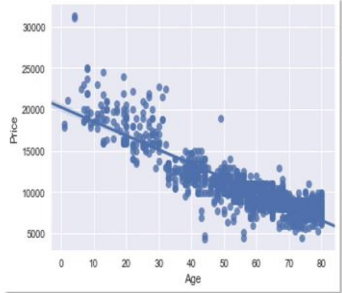
(Refer Slide Time: 05:09)



## Scatter plot


- Scatter plot of *Price vs Age* with default arguments

```
sns.set(style="darkgrid") |  
sns.regplot(x=cars_data['Age'], y=cars_data['Price'])
```



- By default, `fit_reg = True`
- It estimates and plots a regression model relating the x and y variables

Python for Data Science



So, now let us see how to create the scatter plot. So, here we are going to create a scatter plot of Price verses Age and that will be built using the default arguments. So, if you see the first line I have given as `sns.set`; that means, that we are giving a theme to the background of the plot, the theme was being dark grid; that means, that you will have a dark shade as a background of your plot and you will as well have a grid.

So, you can basically set themes to your background and dark grid is a preset seaborn team you can as well set themes that are available from the seaborn package and you can set that under the `style` argument and even by default if you use `sns.set` it will give you a dark grid theme. So, in order to do a scatter plot the function to be used from the `c bond`

library is `regplot`; `regplot` stands for regression plot and inside the function you have to specify the variables for x axis and the y axis.

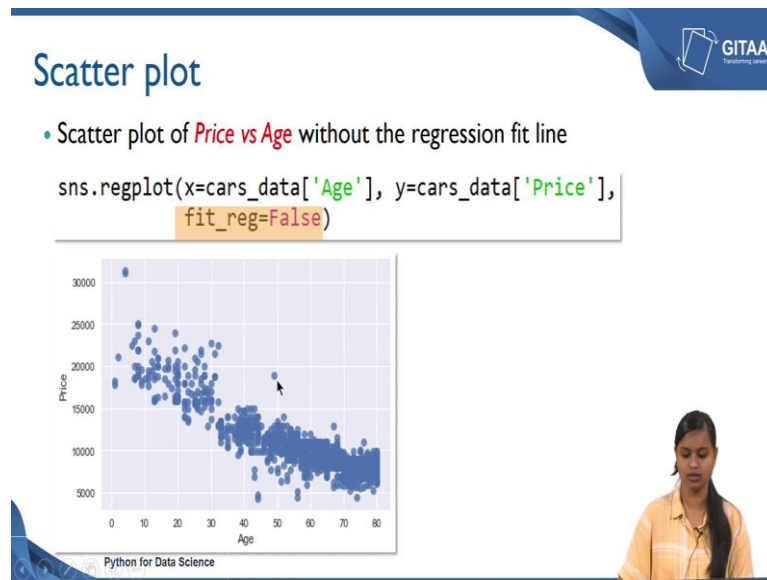
Since it is going to be the scatter plot of Price versus Age we need to give Age under the x axis and Price under the y axis we are going to check how Price varies with respect to the Age of the cars. So, here if you see I have accessed the Age variable from the data frame `cars_data` similarly I have accessed the price variable from the `cars_data`.

So, this would be the output that you will be getting by setting the style as dark grid and by giving x as Age and y as Price. And if you see here as the Age increases the Price decreases and Age that we see here is not in terms of years, but it is in terms of months. So, 70 represents the cars Ages around 5.8 years and 80 represents the cars Ages around 6.6 years.

So, in that case most of the cars or the Price is really low whenever the cars Age is beyond 5 years and the Price is really high for the new Aged car because the car has travelled a lot there could be more damage to it and since it is not new the Price is also coming down. So, it is very obvious from the logic perspective as well as from the data perspective as Age increases the Price automatically decreases. And if you also see there is a regressions line which is plotted over the scatter points that is because by default `fit_reg` is equal to `drew` in `regplot` function.

What it does mean it basically estimates the coefficient of x and then plots a regression line over the points that is why the function is named as regression plot and if you do not want to consider this regression fit line into your scatter plot if you just want to scatter plot of any two variables you can also do that.

(Refer Slide Time: 08:05)



Like here we are going to do a scatter plot of price versus age without the regression fit line that is very easy you can retain the same command as the previous slide, but you just need to add another parameter called `fit_reg` and you can just set them to false.

Because as I mentioned earlier by default it is true that is where you got a regression line over those scattered points, but if you said they are as false you would not be getting the regression line rather you will just get a scatter plot of Price versus Age. So, and here if you see the plot Age is called as the label to the x axis and Price is called as the label to the y axis.

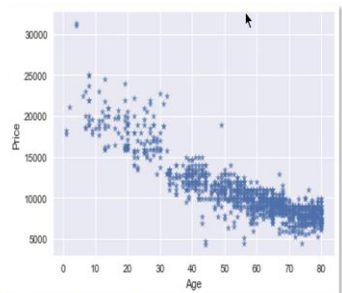
Here we have not specified any labels to x and y axis it is already being shown here that is how because it takes the label from the variable name that we have already given here. And the values 0, 10, 20, 30 are the values from the variable Age and from the plotting context these are called x tick labels and 5,000, 10,000 and so on are called as y tick labels that is on the y axis the boxes are called as grids and the points the scatter points are called as markers. Here we have a solid round points.

(Refer Slide Time: 09:21)


## Scatter plot

- Scatter plot of *Price vs Age* by customizing the appearance of markers

```
sns.regplot(x=cars_data['Age'], y=cars_data['Price'],  
            marker="*", fit_reg=False)
```



Python for Data Science

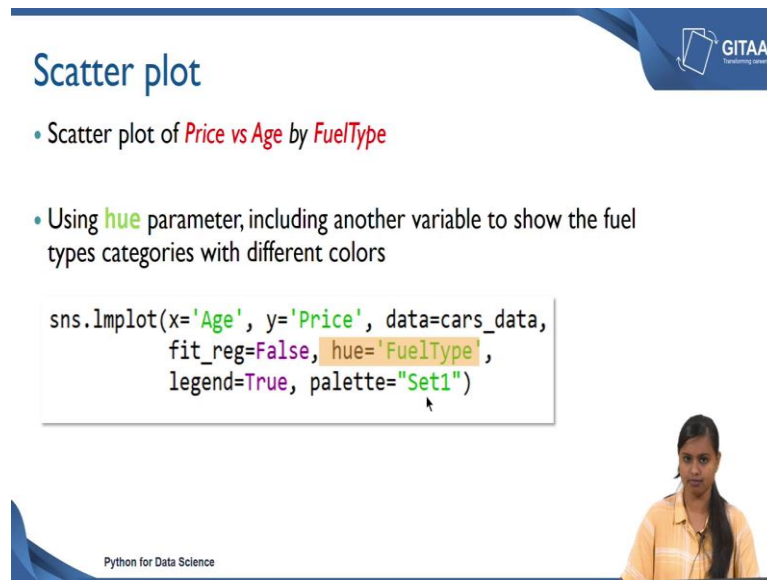


So, now we are going to see how to create a scatter plot of Price versus Age by customizing the appearance of the markers. Appearance of the markers in the sense you can change the shape of your markers to differentiate between the categories, but here I have just taken only the scatter plot of Price versus Age. So, I am going to change all the markers into a different shape.

So, in that case I can do that using the argument called marker and inside the marker argument you can specify the type of marker you wanted to have it in your scatter plot. Here have since I have given the asterisks all the scatter points will be marked as stars this will be very useful whenever you want to differentiate the points with respect to different categories that are available under a variable. Other than that all the codes remains the same I have just added marker is equal to star that should be given under the single or double code.



(Refer Slide Time: 10:21)



The slide features a blue header with the title "Scatter plot" and the GITAA logo. Below the title, there are two bullet points: "Scatter plot of Price vs Age by FuelType" and "Using hue parameter, including another variable to show the fuel types categories with different colors". A code block contains the following Python code: 

```
sns.lmplot(x='Age', y='Price', data=cars_data, fit_reg=False, hue='FuelType', legend=True, palette="Set1")
```

. A presenter is visible in the bottom right corner of the slide.

So, in this example we are going to create a scatter plot of Price versus Age by FuelType; by FuelType in the sense we are going to add one more variable into the scatter plot that is the variable FuelType. Why are we going to add that? Because it is of interest to check the relationship between Price with Age by adding another variable FuelType, like how Price varies when the Age increases with respect to different FuelTypes of the car. So, that can be added by using the parameter called hue and including another variable to show the fuel type categories with different colors.

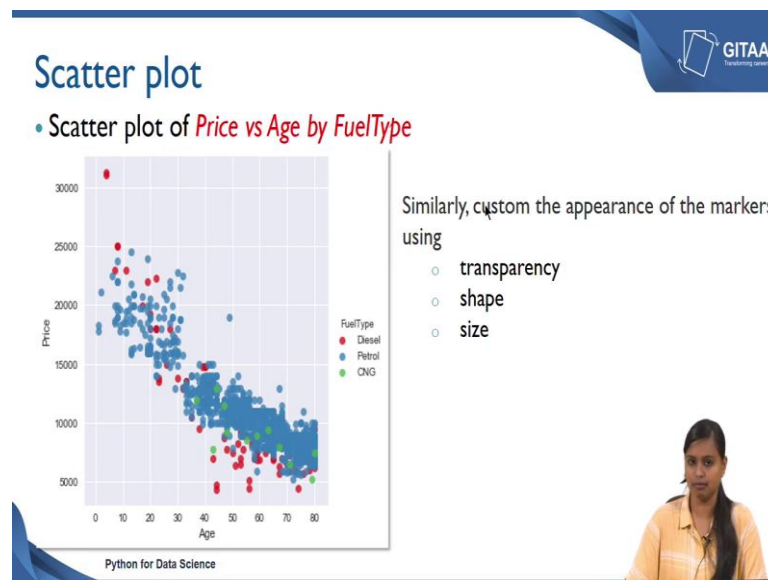
Here we are going to add fuel type under the argument or parameter called hue and we are going to represent the fuel type with respect to different colors. So, here we are going to do that by using the command called lmplot that is from the seaborn library. lmplot is a function which combines regression plot and facet. It is very useful whenever you want to plot a scatter plot by doing conditional subsets of data or by including another variable into the picture in that case a lmplot will be very useful.

And here the x axis is Age variable and the y axis is the Price variable and the data from which we are going to axes the cars\_data and I have also said False different fit\_reg. So, that we do not get a regression fit over the scatter points and here I am adding one more variable into the scatter plot using the hue parameter and since we are going to add one more variable points are going to be differentiated using colors based on the fuel

type of the car in that case we need to know which colour represents what category in that case I need to have legends as well.

So, I have set legends as True and you can also set color palettes using the different color palettes that are available in the seaborn library; one of it is called set1. So, I have just used set1 as the color palette to color the data points based on the FuelType. So, this is what the input is about.

(Refer Slide Time: 12:23)



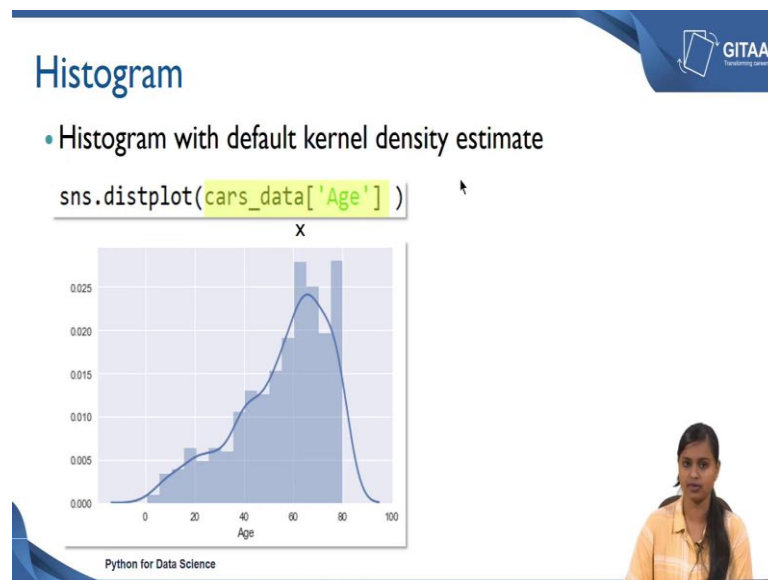
Let us see what is the output that we are getting. So, this is a scatter plot of Price versus Age by FuelType, it is the same scatter plot, but here we have just differentiated the data points using the categories that are available under the FuelType. But we have just differentiated the data points based on the FuelType categories. So, the red represents diesel and the blue represents petrol and the greens are CNG.

So, if you see here there are more data points or there are more cars that are off petrol fuel type and the price is really higher for the cars which has diesel fuel type and the price is comparatively lower for the cars which have CNG fuel type. And you can also check the relationship between price and age as we know that whenever the cars age is high the price is very low whenever the cars age is low the price will be high.

Similarly you can also custom the appearance of the markers using transparency, shape and size. Like how we differentiated the points using colors here different levels of

transparency to the points to represent the categories you can also do the same thing by giving different shapes and by giving different sizes for data points. So, this will help you include more variables into a single plot or you will be able to understand the data quickly rather by looking at the numbers.

(Refer Slide Time: 13:51)



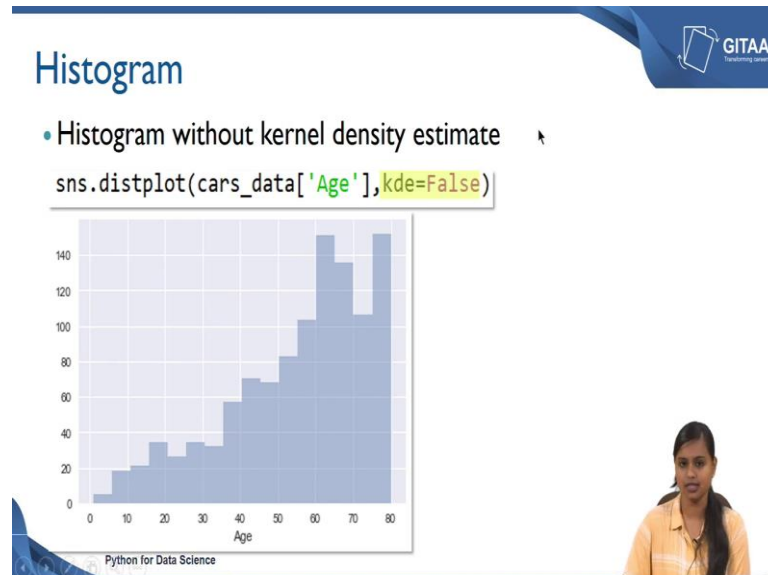
So, next we are going to look at a plot called histogram. So, we are going to plot a histogram with default kernel density estimate because using the seaborn library by default it gives you a histogram with the kernel density estimate. So, the command that needs to be used this dist plot that is the representative of distribution plot.

Since that is from the seaborn library of used `sns.distplot` an input should be any numerical variable for any continuous variable because histogram can be plotted to check the frequency distribution of any continuous variable. In that case Age is one of the continuous variables from the data frame, here we are interested in looking at the frequency distribution of Age how the values are distributed under the Age variable in that case you can use or you can axis the Age variable from the data frame `cars_data`.

So, this is being the representative of the x axis and as you know by default it gives you the kernel density estimate. It gives you work curve over the bars which is the representation of the distribution of the variable Age and on the y axis you have the kernel density estimate, but if your interest is to get the frequency distribution in terms of

counts or numbers, then you can also get rid of the kernel density estimate and then have the counts on your y axis.

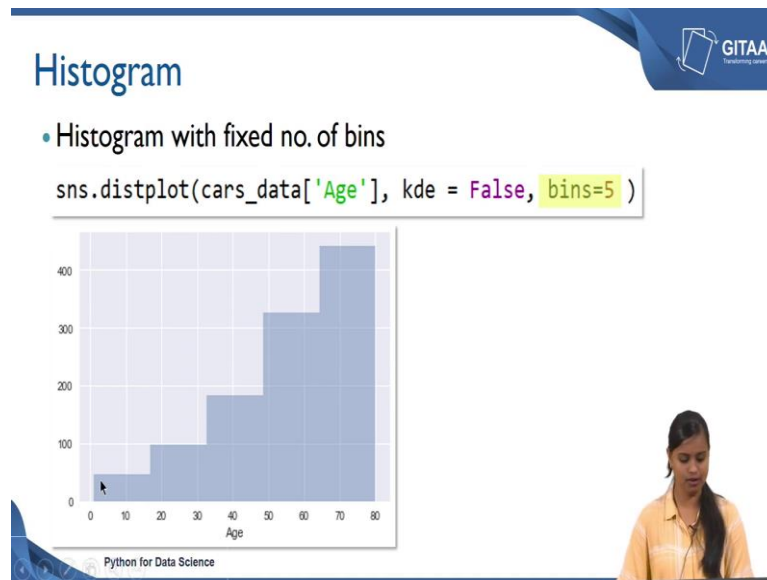
(Refer Slide Time: 15:17)



Let us see how to get that. So, here we are going to plot histogram without the kernel density estimate you can do that by setting kde is equal to False. So, if you give that the output would be similar to this on the y axis it is just a representation of frequencies or the counts on the x axis you have the range of values from the Age variable. And you have too many intervals here it is very difficult to interpret from this plot the range is about 75 to 80, where the cars age is more than 6 years the count is really high. And there are very few cars which are new that is what we can interpret because there are two extremes here; one is really high and one is really low.

So, in that case you can also have a control on how many number of bins you want to have on the plot. So, that by having the control on the bins you will be able to quickly interpret from the distribution of your age variable, let us see how to fix the number of bins.

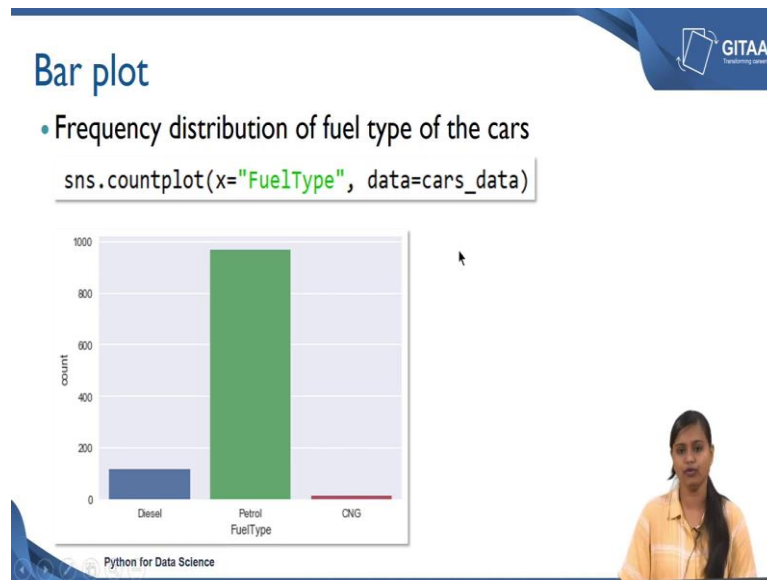
(Refer Slide Time: 16:13)



So, here we are going to plot the histogram with fixed number of bins. So, retain the same codes as in a previous slide you can just add bins is equal to 5. So, that you will have only 5 bins in your histogram.

Bins are nothing but the range or the intervals. So, from this plot by having the number of bins as 5. It is very clear that the frequency is really high for the bin value 70 to 80; that means, that wherever the cars age is above 5 years, the frequency is high. Because most of the cars detail is about the cars whose age is about 70 to 80 months, but you have only very few cars which are very new in our data frame that is why the frequency is very low in the range between 0 to 50 months.

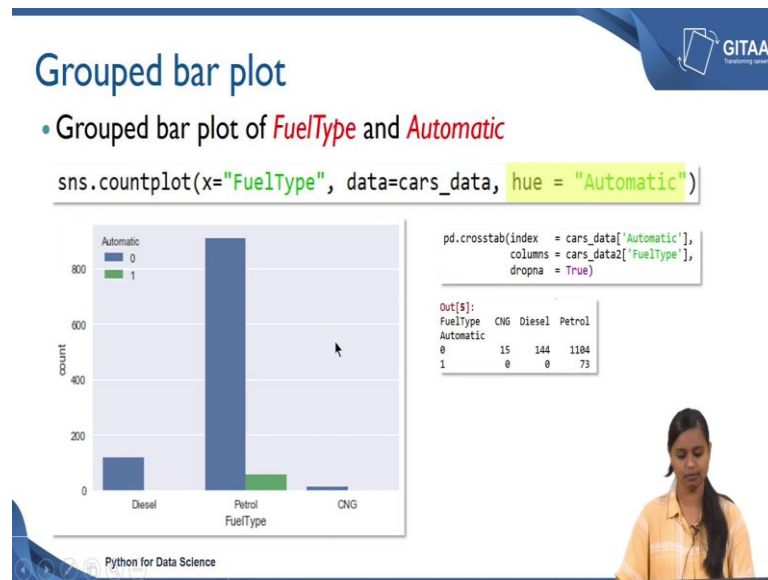
(Refer Slide Time: 17:05)



So, next we are going to look at the bar plot. So, bar plot is basically to check the frequency distribution of any categorical variable, here FuelType is one of the categorical variable in our data frame. So, let us look at the frequency distribution of the fuel type of the cars. So, to create a bar plot countplot is the function that needs to be used under the function you just need to give the variable of interest that is FuelType and you can also specify from which data frame that you are accessing it from, it is cars\_data.

So, it is straightforward you will get the output, you are getting the fuel type on your x axis at the count on your y axis and the bars are the representative for the counts with respect to each FuelType. So, it is very evident from the plot that most of the cars have the FuelType as petrol and there are very few cars whose FuelType are diesel or CNG. So, now, we have seen how to create the bar plot.

(Refer Slide Time: 18:01)



So, here we are going to see how to create group bar plot. We are going to create the group bar plot of FuelType and Automatic, we mean by group bar plot. So, here we are going to have the bar plot of FuelType by adding another variable as Automatic. So, we are going to look at the frequency distribution of the FuelType of the car along with the interpretation whether the cars gearbox is of Automatic or manual.

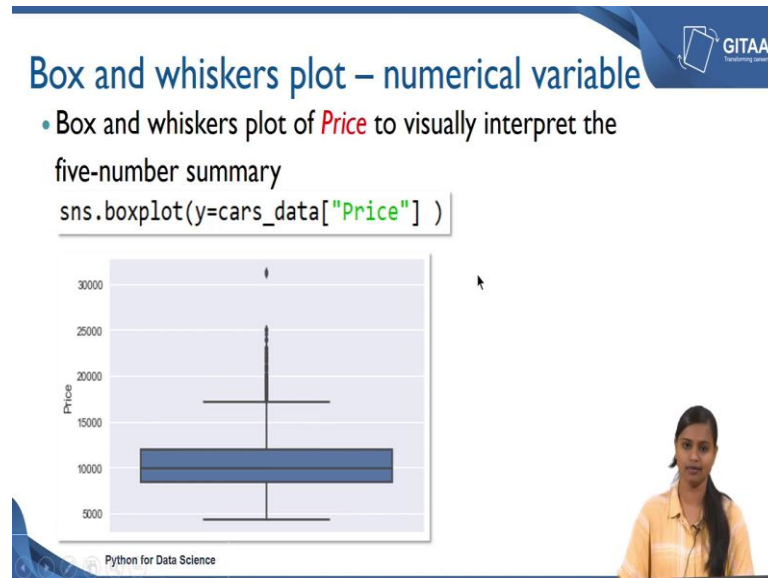
So, let us see how to do that we can use the same function as bar plot that is count plot and you can specify the FuelType under the argument x and you can also specify the data frame. So, that it knows from which it should axis the FuelType from and by adding hue is equal to Automatic we are including one more variable into the bar plot. So, that the we are checking the relationship between the FuelType and Automatic.

So, let us see what would be the output. So, here is the output on the x axis you have the FuelType you have different fuel types like diesel, petrol and CNG and on the y axis you have the count. If you see the legend here it has two value 0s and 1s and the 0s are the representative of manual gearbox and the 1s are the representative of automatic gearbox.

So, if you see here there are only automatic gearbox types whose fuel type is of petrol and there are no cars available with automatic gearbox when the fuel type is about when the fuel types are diesel or CNG. If you recall this is just the visual representation of the crosstab output that we have seen earlier in the lectures. So, we have used the crosstab function to arrive at the relationship between the Automatic on the FuelType where we

have seen that in terms of numbers and we have tried to interpret that, but here using visualization we are going to interpret the same thing, but visually.

(Refer Slide Time: 19:53)



So, next we are going to look at box and whiskers plot we are going to look at is box and whisker plot for numerical variable for if you have a numerical variable in your data frame and if you want to interpret more from the variable, then you can go for box plot. Because the box plot will give you the visual representation of the. The five number summary includes mean, median; the five number summary includes minimum maximum and the three quantiles the those are first quantiles, second quantile and third quantile.

So, let us see what the quantiles are and what the minimum and maximum values are. So, how do we do that? We can create a box plot using the command box plot that is from the seaborn library and since we are interested in getting five number summary of the Price variable I have accessed the Price variable from cars\_data and I have given that under the argument called y. So, that the y will be the representation of Price, but if you give it under x the x will be the representation of Price.

So, now, let us see how the output will look like. So, as I mentioned since we have given Price under the argument y we are getting the Price value on the x axis. So, now, let us try to interpret the box plot it is called as box and whiskers plot because it has some box as well as the whiskers to it the horizontal lines are called as the whiskers, now let us try



it interpret this. This horizontal line is called as the lower whisker or also the representation of the minimum value which is excluding the outliers.

If you have outliers in your data which are nothing, but the extreme values then it excludes that values and then gives you the picture of what is the minimum value of the Price. The minimum value of the Price of the car is about 5000 Euros and the upper whiskers which represents the maximum value that is excluding the outliers.

So, the maximum value of the cars is around 16 to 17 thousand by excluding the outlier or the extreme values and these are called low whisker and the upper whisker. The lowest horizontal line of the box represents the first quartile that is 25 percentage of the cars Price is less than 8000 and the middle line represents the second quartile which is nothing, but the median. That means, that 50 percentage of the data is less than 10,000 Euros. And the upper horizontal line is nothing, but third quartile that is  $q_3$  which represents that 75 percentage of the Price of the car is less than around 12,000 Euros this is called  $q_1$ ,  $q_2$  and  $q_3$ .

And whatever the points that are lying above the whiskers are called as outliers and you can also have the values below the whiskers as well. If you have points about the upper whisker then you can say that these are outliers or the extreme values where the values are really more than the 1.5 times of your  $q_3$ . This is what the  $q_3$  values about.

The  $q_3$  is around 12000, so these values are 1.5 times of  $q_3$  that is why it is called as extreme values, it is not in between the range of the price that is why it is called as the extreme values. Similarly you can ask what happens when you have some values above the lower whiskers in that case those are also called as outliers or extreme values because those values will be less than the 1.5 times of  $q_1$ .

So, using box plot you will be able to visually look at the distribution of the Price variable and what would be the median value and what are the quartile values of the Price. And using the box plot the main key point to the box plot is you will be easily able to identify the outliers of any numerical variable. So, this is how we create a box plot and this is how we interpret the box plot as well.

(Refer Slide Time: 23:57)

## Box and whiskers plot

- Box and whiskers plot for numerical vs categorical variable
- Price of the cars for various fuel types

```
sns.boxplot(x = cars_data['FuelType'], y = cars_data["Price"])
```

Python for Data Science

So, now we are going to look at the box and whisker plot for numerical with this categorical variable because box and whisker plot is very useful when you want to check the relationship between one numerical and one categorical variable, like how the price varies with respect to the another variable. Because in the previous example we have just looked at the distribution and how do we detect the outliers using the box plot.

Here we are going to check the relationship between two variables in that case you can use box plot; you can use box plot when you have one numerical and one categorical variable here we are going to look at the price of the cars for varies fuel types of the cars. So, you can use the same command as in the previous slide that is box plot and under the x argument I have given FuelType and another y argument I have given the variable as Price.

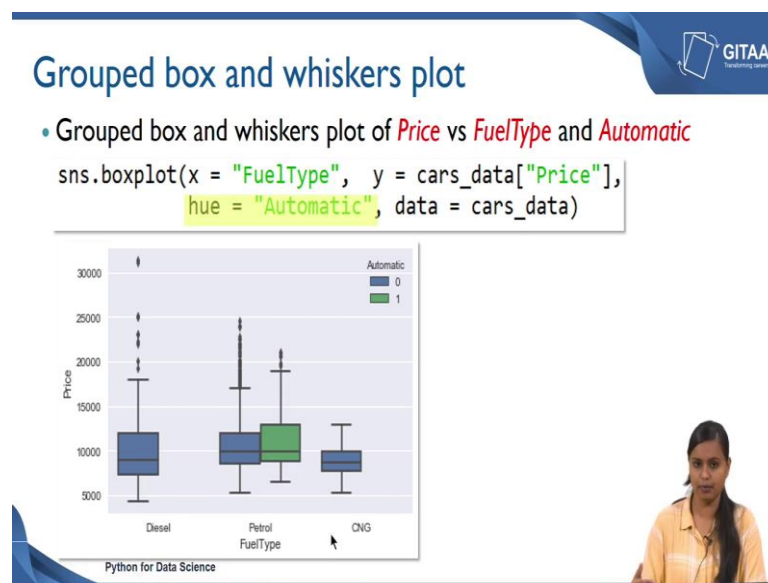
So, you have FuelType on your x axis and you have Price on your y axis and by looking and the plot it is very evident that the Price varies with respect to different FuelTypes of the cars. And as we know that the middle line is the representation of the median if you look at the middle lines of the different FuelTypes of the cars, the median Price of the car is really high when the FuelType of the car is petrol.

And the median value is really low when the FuelType is of either diesel or CNG. It is very evident that the on an average the petrol FuelType has the highest Price among the cars from the data set that we have and you can also see the maximum value of the

maximum price of the car is for the diesel FuelType and the minimum value of the car is also for the diesel FuelType.

So, these are the interpretation that you can make here and you can see there are some extreme values that are above the upper whiskers of the diesel and petrol FuelType and those are called as the extreme values. Because very few observations are there and those are about the 1.5 times of your third quartile that is 12,500 Euros. So, now, we have checked the relationship between FuelType and the Price.

(Refer Slide Time: 26:05)



So, now we are going to look at the group box and whiskers plot. Here we are going to look at the group box and whiskers plot of Prices versus FuelType by including one more variable called Automatic. So, let us see how to do that. The codes remains the same whenever you want to have a grouped box plot you just need to add that variable including the argument called hue and I have added Automatic under hue. So, we will have a group box and whiskers plot, let us see what the output is about.

So, here you have the same box plot as the previous slide, but you have another boxes which is corresponding to the different FuelTypes and you have another boxes which are the representation of the Automatic variable. Here if you see the Automatic variable is being included here and the 0s are the representative of manual gearbox and the 1s are the representative of Automatic gearbox and those are represented by blue and green in color respectively.

So, here whenever the cars fuel type is petrol and the gearbox type is also Automatic there are no cars that are available for the Automatic gearbox when the FuelType is of either diesel or CNG, that is very evident from any of the plots that we have seen in the previous slides as well.

(Refer Slide Time: 27:25)

**Box-whiskers plot and Histogram**

- Let's plot box-whiskers plot and histogram on the same window
- Split the plotting window into 2 parts

```
f,(ax_box, ax_hist)=plt.subplots(2, gridspec_kw={"height_ratios": (.15, .85)})
```

Python for Data Science

Till now we have been looking at having only one plot in a window, we can also have multiple plots in a window, here we are going to look at the box and whisker plot and the histogram on the single window. So, let us plot box whisker plot and histogram on the same window. So, in order to do that we have to first split the windows into two parts. What do we mean by splitting the window is into two parts? This is what we mean.

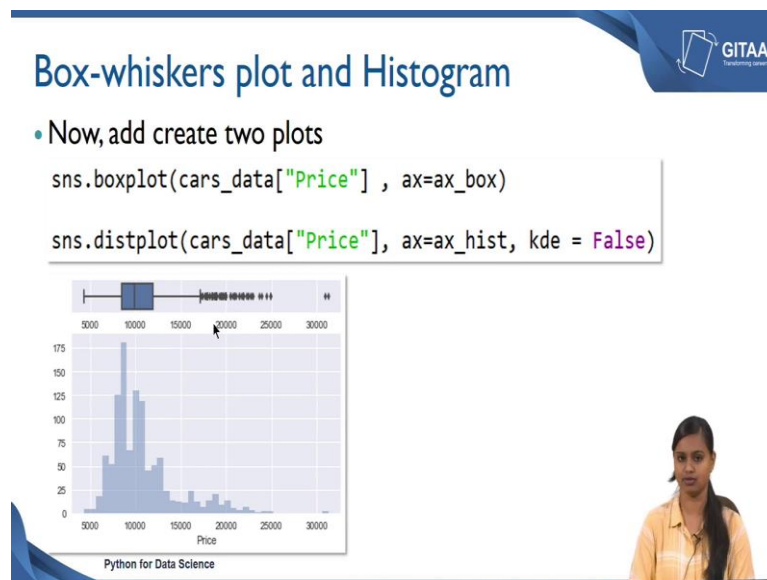
So, this is a single window I have just splitted the window into 2. So, that I can plot two plots in the same window and it will be easier for me to interpret from the two plots instead of creating different plots for the same variable I can as well do that in the same window itself. So, here if you see there is only small space for the upper part and there is a large space for the lower part that is how because I have said the aspect ratio with just because I have said the aspect ratio accordingly. Let us see how to split the window with giving; let us see how to split the window plotting window and we will also see how to customize the aspect ratio.

So, subplots is a function that is used to create the subplots and by giving 2 you mean that you need to have 2 rows. So, you want to split your window into 2 parts in row wise

and by giving specification of grids using grid spec\_kw you can set the ratios of your height I have given 0.15 and 0.85. So, that there is a larger space for the second window and there is a less space for the first window.

I have just splitted my window into 2 and given the aspect ratio and I have saved that output to a and I have saved that output to two variables called f and axis box plot and axis for histogram because these are going to be same for both the box plot and the histogram. Combinely have saved it to two objects and I have also saved it to a figure. So, ax\_box represents the axis for box plot and ax\_hist represents the axis for the histogram.

(Refer Slide Time: 29:33)



The slide features a blue header with the title "Box-whiskers plot and Histogram" and the GITAA logo. Below the title, a bullet point states "Now, add create two plots". Two lines of Python code are shown in a white box: `sns.boxplot(cars_data["Price"], ax=ax_box)` and `sns.distplot(cars_data["Price"], ax=ax_hist, kde = False)`. Below the code is a plot showing a box plot and a histogram of car prices. The x-axis is labeled "Price" and ranges from 5000 to 30000. The y-axis ranges from 0 to 175. The box plot shows a median around 10,000, with whiskers extending from approximately 5,000 to 20,000. The histogram shows a distribution of prices with a peak around 10,000. A woman is visible in the bottom right corner of the slide.

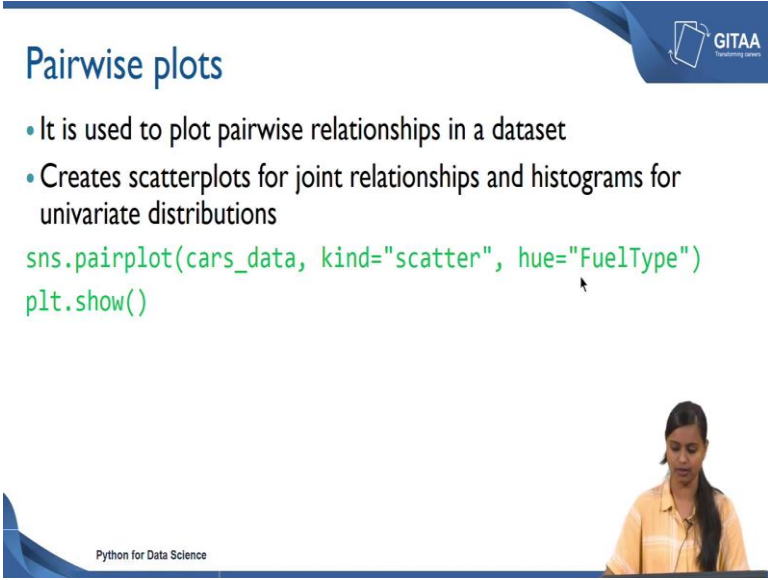
So, let us now we can add or create two plots. So, the codes remains the same because for creating the box plot you have to use a box plot and the variable of interest is Price and using the axis ax\_box. And we have also created the histogram using the dist plot function, where we have specified the variable as price and were we have also specified what is the axis for that that is using the same axis that is ax\_hist. And we wanted just the histogram with not the kernel density estimate; so that is why we have set kb is equal to false.

So, let us see how the output will look like. So, this is the output for the first command using the boxplot you are getting the boxplot for the variable price as well as you are getting the histogram for the variable Price. So, here this is the histogram of Price as well

as the boxplot. So, you will be able to look at the frequency distribution of any continuous variable as well as the five number as well as you will be able to look at the visual representation of five number summary of it.

If you want to look how the outliers are present in a variable and what is the minimum and maximum value of a variable. So, all these information can be get using a single plot when you create plots using subplots.

(Refer Slide Time: 30:57)



The slide features a blue header with the text 'Pairwise plots' and a logo for 'GITAA Transforming careers'. Below the header, there are two bullet points: '• It is used to plot pairwise relationships in a dataset' and '• Creates scatterplots for joint relationships and histograms for univariate distributions'. A code block shows the following Python code: 

```
sns.pairplot(cars_data, kind="scatter", hue="FuelType")  
plt.show()
```

 At the bottom of the slide, there is a small inset image of a woman in a yellow shirt, and the text 'Python for Data Science' is visible in the bottom left corner.

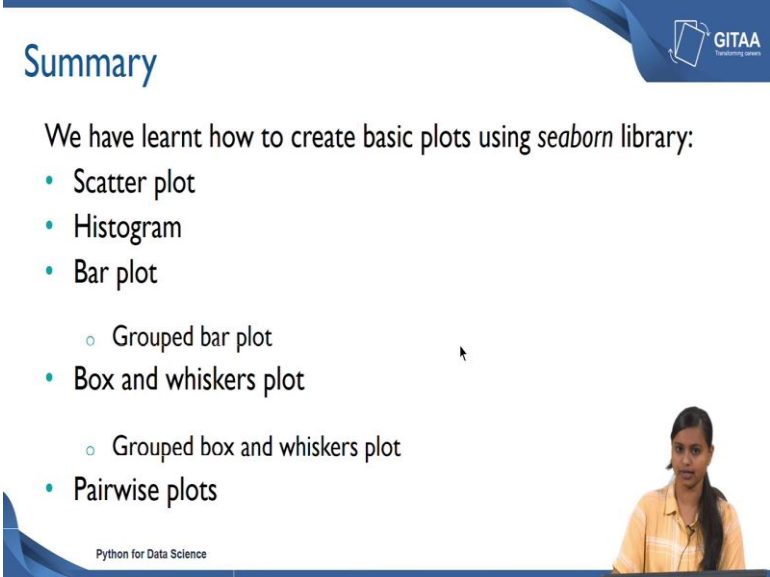
So, now we will see how to create pairwise plots pairwise plots are used to plot pairwise relationships in the data and basically creates the scatter plots for joint relationships and histograms for univariate relationships you will understand what do we mean by joint relationships and univariate relationship when we look at the output of it.

So, to create the pairwise plot the command that is used this pair plot that is also from the seaborn library and you can also specify the data frame because on the whole we are going to look at the relationship between all the variables considering all the variables. And the kind of plot that we are going to plot for any numerical variable is scatter and we are going to colour the data points using another variable called FuelType.



So, the pairwise plot is used to get the plots for all possible combination of variables and you will be able to look at in terms of scatter plot and histogram. This would give you an easier way to understand the relationship that exists between different pairs of variables. You can also see there is a legend which is shown here for the fuel type like how the markers have been colored.

(Refer Slide Time: 33:47)



**Summary**

We have learnt how to create basic plots using *seaborn* library:

- Scatter plot
- Histogram
- Bar plot
  - Grouped bar plot
- Box and whiskers plot
  - Grouped box and whiskers plot
- Pairwise plots

Python for Data Science

So, now we have come to the end of the session, let us summarize whatever we have learned till here. So, we have learned how to create the basic plots using the seaborn library; the first thing that we have seen is how to create scatter plot and then we have seen how to create histogram using the seaborn library. And we have also looked at how to create simple bar plot and then we have seen how to create grouped bar plot.

Followed by that we have seen how to create box and whiskers plot and also, we have seen how to create group box and whiskers plot. At last we have seen how to create the pairwise plot from the seaborn library which was very useful to look at the relationship of all the variables in a single window.

Thank you.