

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


Lecture – 19
Data Visualization Part –I

(Refer Slide Time: 00:21)


In this lecture

We will learn how to create basic plots using *matplotlib* library

- Scatter plot
- Histogram
- Bar plot



Python for Data Science



Hello all welcome to the lecture on the module Data Visualization. So, in this lecture, we are going to do data visualization, where we are going to learn how to create basic plots using the matplotlib library. The basic plots include scatter plot, histogram and bar plot.


(Refer Slide Time: 00:33)

Data Visualization


- Data visualization allows us to quickly interpret the data and adjust different variables to see their effect
- Technology is increasingly making it easier for us to do so

Why visualize data?

- Observe the patterns
- Identify extreme values that could be anomalies
- Easy interpretation



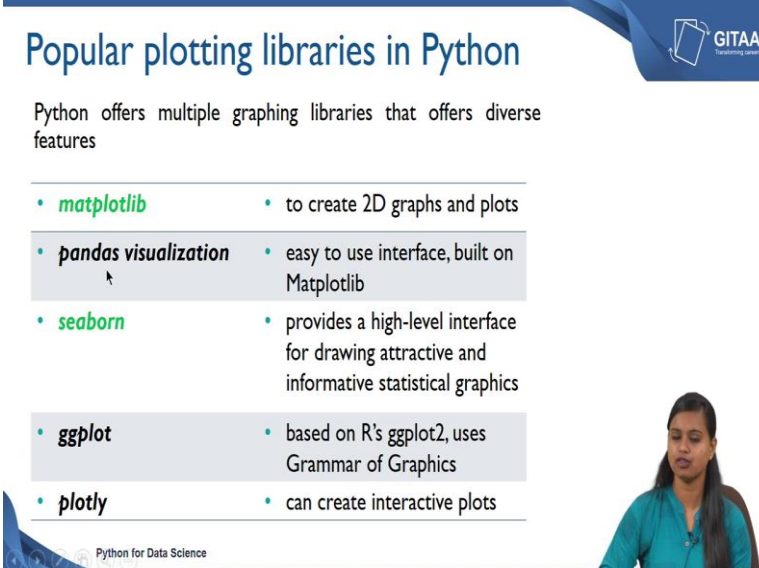
Python for Data Science



So, now let us see what data visualization is about data visualization, allows us to quickly interpret the data and adjust different variables to see their effect. What do we mean by quickly interpreting the data, because if we were to understand the data using numbers, then it would be difficult for us to understand the data and quickly interpret the data. But if we were to understand through graphical representation of data, it would make us; it would make the work easier in terms of interpretation and in terms of understanding.

If you see here the technology is also increasingly making it easier for us to do so because there are so many visualization techniques tools that are available to do visualization to understand the data even more better than interpreting from the numbers. But why do we need to visualize the data? We can observe the patterns. You can also identify the extreme values that could be anomalies. And if you want to interpret the data easily, then you have to go for visualization. So, these are the important points to visualize the data.

(Refer Slide Time: 01:37)



The slide is titled "Popular plotting libraries in Python" and features a GITAA logo in the top right corner. Below the title, it states "Python offers multiple graphing libraries that offers diverse features". A table lists five libraries with their descriptions:

Library	Features
matplotlib	to create 2D graphs and plots
pandas visualization	easy to use interface, built on Matplotlib
seaborn	provides a high-level interface for drawing attractive and informative statistical graphics
ggplot	based on R's ggplot2, uses Grammar of Graphics
plotly	can create interactive plots

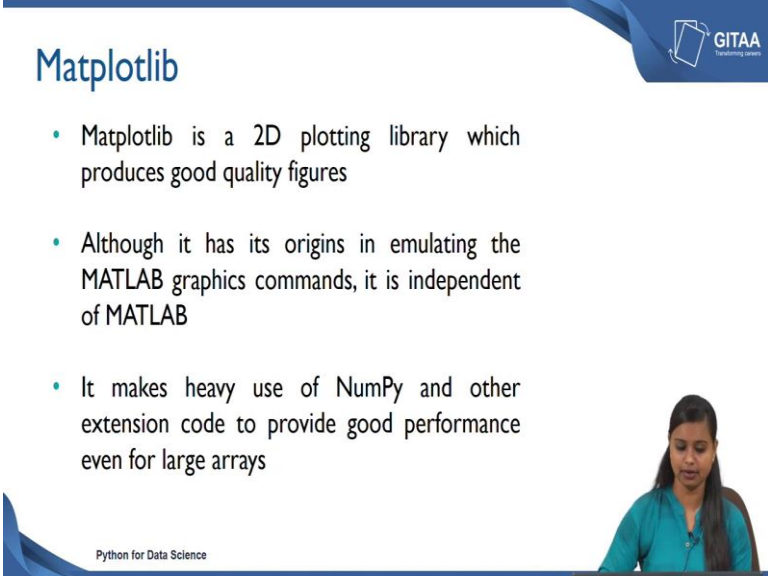
The slide also includes a small video inset of a woman in the bottom right corner and a "Python for Data Science" logo in the bottom left corner.

So, python offers multiple graphing libraries that offers diverse features, we are going to look at few graphing libraries that are available in python. And what is the need, first one is matplotlib which is used to create 2D graphs and plots. And I have mark it as green because we are going to look at matplotlib library in this lecture. The next one is pandas visualization which has easy to use interface and that was built on top of matplotlib.

The next one is seaborn it provides a high level interface for drawing interactive and informative statistical graphics and that was also built on top of matplotlib, whatever graph and plots that you are creating using seaborn library can also be created using matplotlib library. And you can also have a control on what insights you can get of your plot using seaborn. The next one is ggplot; ggplot is used whenever you want to do advanced graphics, and this ggplot is entirely based on R's ggplot, R's another programming language that is used for analytics. And it has a package called ggplot2 which is entirely dedicated for doing visualization.

And in python we have ggplot that is based on R's ggplot2, and it basically uses grammars of graphics the next is plotly. If you want to create interactive plots, then you can go for plotly. But in this course we are going to look at two graphing libraries that are matplotlib and seaborn.

(Refer Slide Time: 03:11)



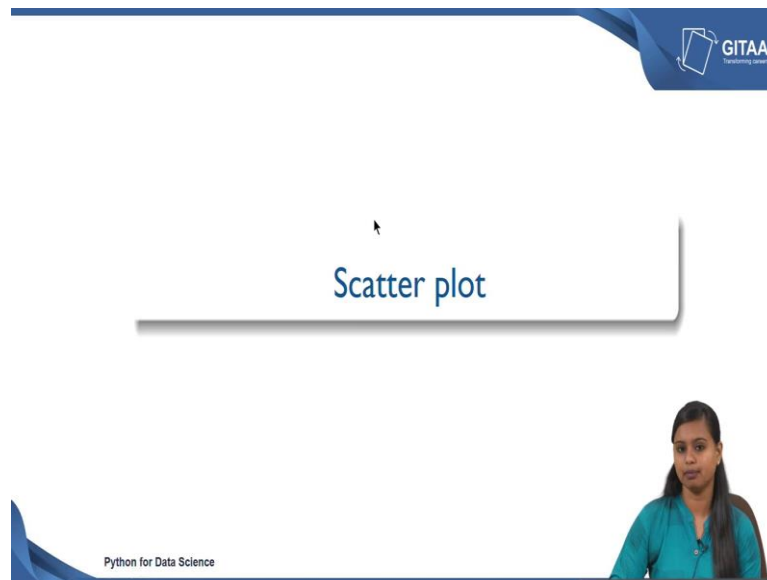
Matplotlib

- Matplotlib is a 2D plotting library which produces good quality figures
- Although it has its origins in emulating the MATLAB graphics commands, it is independent of MATLAB
- It makes heavy use of NumPy and other extension code to provide good performance even for large arrays

Python for Data Science

First we are going to look at a matplotlib library. So, let us see what matplotlib library is about. Matplotlib library is a 2D plotting library which produces good quality figures. Although it has its origin in emulating the MATLAB graphic commands it is independent of MATLAB. It makes heavy use of NumPy and other extension code to provide good performance even for large arrays. So, this is the overview of the matplotlib library.

(Refer Slide Time: 03:39)



(Refer Slide Time: 03:42)

A presentation slide with a blue header and footer. The header contains the GITAA logo (Transforming careers) on the right. The main content area is white and features the title 'Scatter Plot' in a blue font. Below the title, there are two sections: 'What is a scatter plot?' followed by a bullet point defining it as a set of points for two variables on horizontal and vertical axes; and 'When to use scatter plots?' followed by two bullet points explaining their use for conveying relationships and being called correlation plots. In the bottom right corner, there is a video inset of the same woman in the teal shirt. The bottom left of the slide has the text 'Python for Data Science'.

So, under the matplotlib library we are going to look at a plot called scatter plot. So, let us see what is a scatter plot first. A scatter plot is basically a set of points that represents the values that obtained for two different variables plotted on a horizontal and vertical axis. So, if you look at a scatter plot, you will have to access to it x and y. And the points from the data frame will be scattered all over the place of the plot or in a separate pattern. And using that and using the variables that you have given as an input, you will get the relationship out of it.

And there can be a question when to use those scatter plots. So, basically the scatter plots are used to convey the relationship between any two numerical variables, how one variable varies with respect to the other variable. And scatter plots are also sometimes called as correlation plots, because they show how two variables are correlated, whether they are negatively correlated or positively correlated, all this information you can get that by looking at the patterns of your scatter plots.

(Refer Slide Time: 04:44)

The slide is titled "Importing data into Spyder" and features the GITAA logo in the top right corner. It lists the following code snippets with corresponding callouts:

- `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

The slide also includes a small video inset of a woman in the bottom right corner and the text "Python for Data Science" in the bottom left corner.

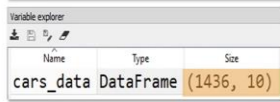
So, now we are going to import the data into Spyder to do a scatter plot. Prior to importing the data we have to import the necessary libraries. The first library is pandas and we have imported it with the as pd and we have to import the pandas library to work with data frames. And then we will also be doing some numerical operations on it. So, we need to import numpy as np. And then since we are going to use the matplotlib library, and we have to import pyplot from the matplotlib as plt. And we use this to do visualization.

(Refer Slide Time: 05:21)

Importing data into Spyder

- Importing data

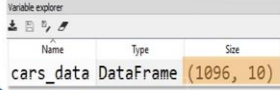
```
cars_data = pd.read_csv('Toyota.csv', index_col=0,  
                        na_values=["??", "???"])
```



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)

Python for Data Science

Now, we can import this data into Spyder using the `read_csv` command. And we have said the first column as the index column, and we have considered all the question marks as default nan values, so that we can do any operations that are related to nan values. So, `cars_data` is your data frame now and the size of it is 1436 with 10 columns. We are going to remove all the missing values from the data frame, because we are going to visualize the data. In that case I do not want to consider all the missing values rather I just want to consider all the rows where there are no missing values.

So, in that case, if you want to drop out all the records where there are missing values, then you can use the command called `dropna` preceded with the data framing. And inside the function if you set `axis` is equal to 0, then you consider all the rows where there are missing values and remove that. And I have also given `inplace` is equal to `true` that is because I have not assigned this to a new data frame as `cars_data` to or are in the existing data frame rather I have just used `inplace` is equal to `true` my setting `inplace` is equal to `true`, the modifications that you are making here will be reflected in the data.


If you do not give this, the records will not be dropped from your data frame. If you look at the size of your data frame after removing the missing values, it turned out to be having 1096 observations with 10 columns. So, we have left out around 400 observations.

(Refer Slide Time: 06:57)

Scatter plot

```
plt.scatter(cars_data['Age'], cars_data['Price'], c='red')
plt.title('Scatter plot of Price vs Age of the cars')
plt.xlabel('Age (months)')
plt.ylabel('Price (Euros)')
plt.show()
```

Python for Data Science

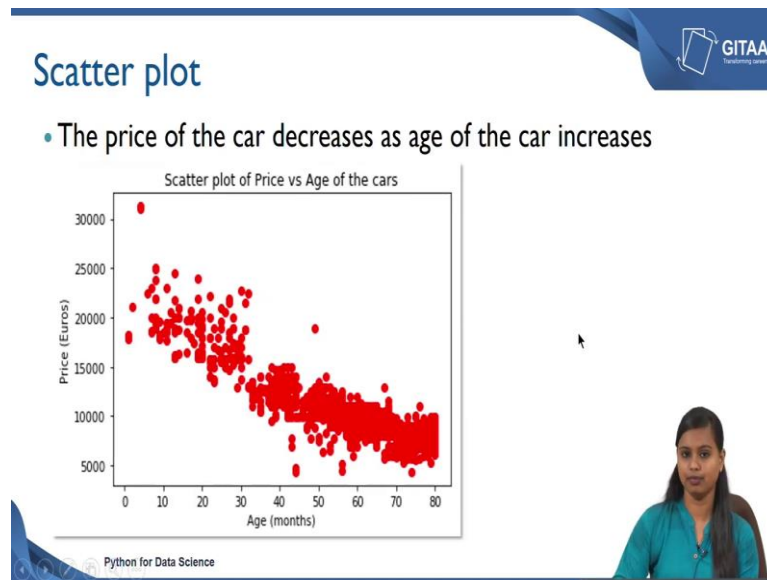


So, now let us see how to create a scatter plot. So, for creating a scatter plot, scatter is the function since we are doing it from the matplotlib library. We have saved pyplot as plt, so that is why I have used the allies called plt to access the function scatter. And inside the function I have given the x coordinate variable that is age and I have also given the y coordinate variable that is price. So, it is very evident that we are going to check the relationship between age and price through scatter plot.

And the next thing is I have given red under the parameter c that means that I am going to color the markers using the red color. So, we will see what markers when we get the output. This line will produce a scatter plot. But if you want to add title to your plot and if you want to add labels to your x-axis and y-axis, you can also give that. See if you want to add title to your plot, you can give it as plt.title. And inside that I have just given it a scatter plot of prices age of the cars.

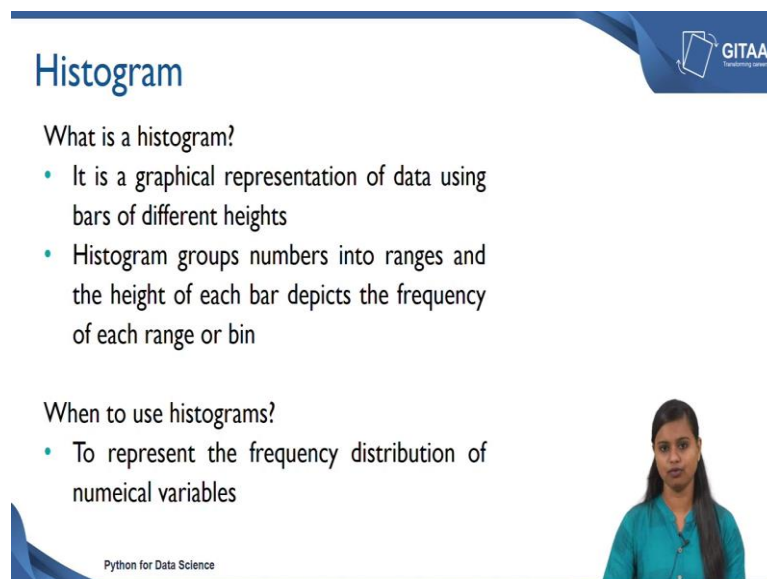
And the x label represents the age which are in months, and the y-axis represents the price which are Euros. Now, we have created a scatter plot to show the scatter plot you have to use plt.show. Even if you select the lines to plt.y label, you will be able to show the plot. Since this is a standardized code, whenever you are building your plots layer by layer by giving the type of plot first and then giving titles after that, then you have to run everything in one shot to plot.show to get all the information in a single plot.

(Refer Slide Time: 08:43)



So, now let us see what is the output by using the scatter function. Here this is the scatter plot to show the relationship between price of the car and the age of the car. So, if you see from the output, your x-axis represents the age and y-axis represents the price; if you see here, as the age increases the price decreases. The points that are shown here are called as markers. And the vertical line that is shown here are ticks, and the 60, 70, 80 are called tick labels. Since these are an x-axis, these are called xticks and x-labels, and these lines are called yticks and these values are called ytick labels.

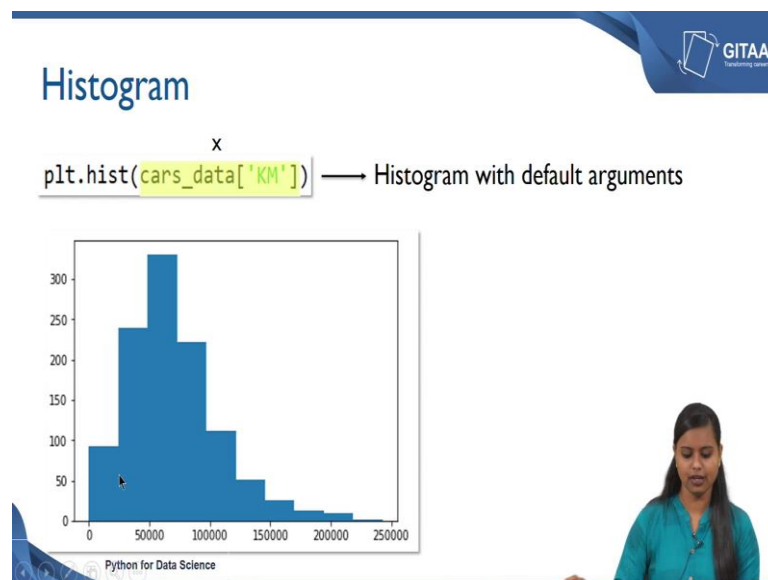
(Refer Slide Time: 09:37)



So, now we are going to see how to create the histogram. Before creating we need to know what histogram is. So, what is a histogram? It is a graphical representation of data using bars of different heights. So, whenever you have a numerical variable, and if you want to check the frequency distribution of the variables, you can go for histogram, so that the each bar will give you different heights that represents the frequencies.

And histograms groups numbers in two ranges, because on the x-axis you will have intervals or ranges that can also be called as bins. And for each of the bins, you will get a corresponding frequency by looking at the height of each bars. So, when to use the histogram? As I mentioned to represent the frequency distribution of any numerical variables you can look for histogram.

(Refer Slide Time: 10:27)

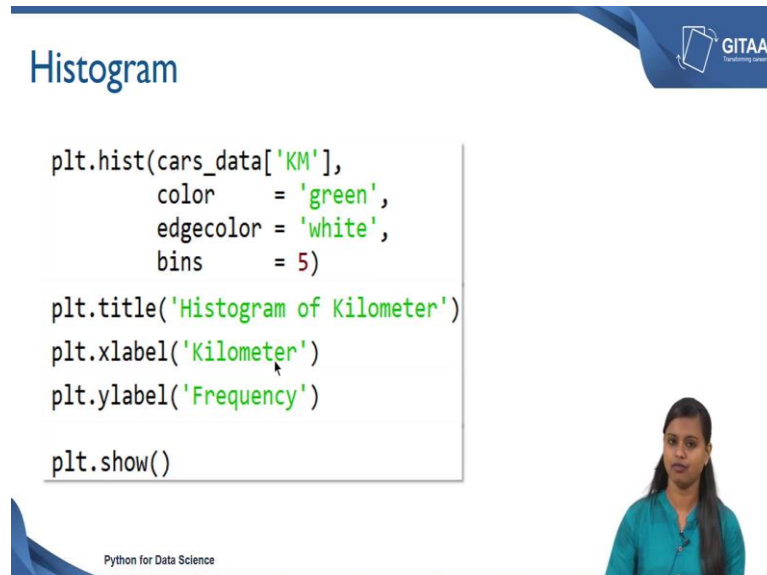


Now, we will see how to create a histogram using matplotlib. So, the command is hist that stands for histogram, and that is from the library called pyplot library called matplotlib, and pyplot is the sub library that is available under mat plot and plt is just analyzed to the pyplot. And inside the function, I was just given kilometer from the cars_data, data frame. So, that becomes the x-axis. So, we are creating a histogram with the default arguments, because we have just given the x variable alone.

And let us look at the output. So, here is the histogram with default arguments on the x-axis you have the bins. The first bin is from 0 to 50,000, and the second bin is from 50001 to 100000 lakh, and the third bin is from 100001 to 150000. For example, using

this histogram I am not able to get the exact bin range or the exact range the corresponding frequency, because it does not give me the separation between each bars.

(Refer Slide Time: 11:36)



Histogram

```
plt.hist(cars_data['KM'],
        color = 'green',
        edgecolor = 'white',
        bins = 5)

plt.title('Histogram of Kilometer')
plt.xlabel('Kilometer')
plt.ylabel('Frequency')

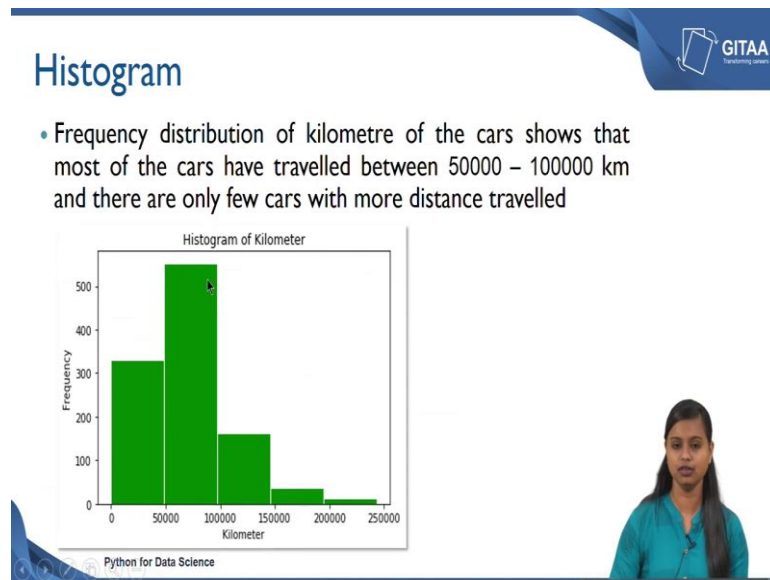
plt.show()
```

Python for Data Science

So, now, what I can do here is I can create a histogram by specifying the edge color and the number of bins I have. If I have too many bins, then it would not be able to easier for us to understand the frequencies for each interval, in that case you can specify or fix the number of bins as well. So, here we are using the same command, where we have fixed the x-coordinate that is kilo meter.

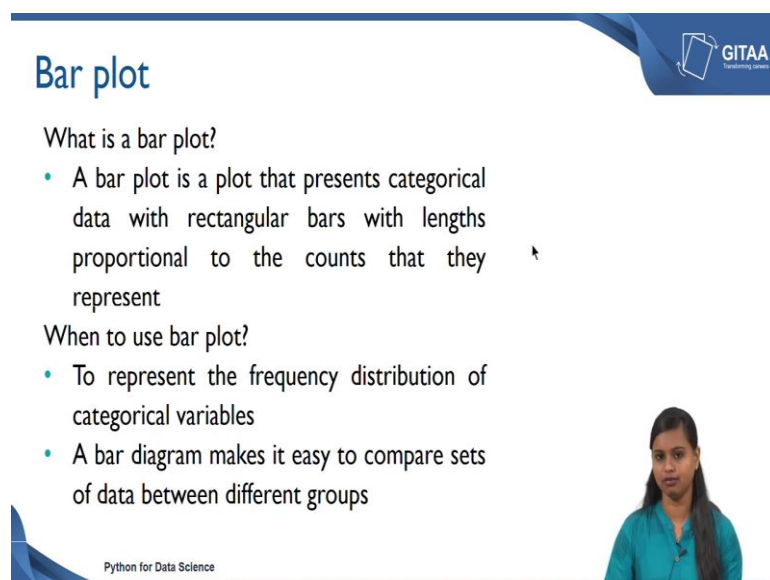
Since we are looking at the frequency distribution of the kilometer of the car and the color for the bars will be green, and the edge color of the bars I am giving it as white, so that there is a clear demarcation between two bars, and I have fixed the number of bins as 5. And you can also add title xlabel and the ylabel to your plot here the title for the plot is histogram of kilo meter and the xlabel is kilometer and the ylabel is frequency. And if you give plt.show and if you run from plt.hist to plt.show, you will get a histogram with those specified number of bins.

(Refer Slide Time: 12:42)



So, here is the output that we get from the previous code. So, here we are looking at the frequency distribution of kilometer of the cars. And it shows that most of the cars have traveled between 50000 to 100000 kilometer and there are only few cars with more distance travelled. Now, if you see that there is a clear demarcation between each of the bus, and you will be able to look at the frequencies of it easily.

(Refer Slide Time: 13:12)



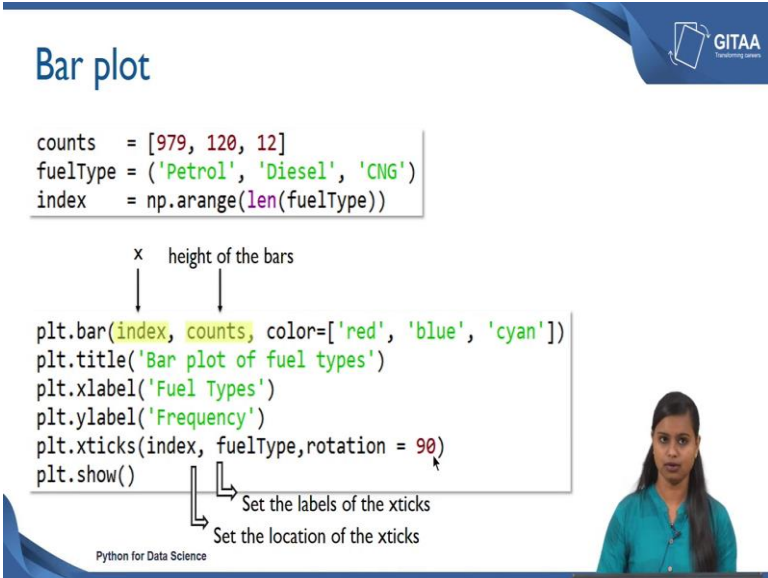
Next we are going to look at a bar plot. Let us see what is a bar plot. Bar plot is a plot that represents categorical data with rectangular bars. Whenever you have a categorical data,

and if you want to look at the frequencies of each categories in a variable, then we will look at in terms of a bar plot. Bar plot is also similar to histogram, there would not be any space in between the bars for this histogram, because you are looking at in terms of continuous range.

But here you will be looking at in terms of frequencies of categories, so you will have space in between the bars that is what the difference between the bar and the histogram. And you use bar plot for a categorical variable and we use histogram for a numerical variable. And those length of the bar is basically depicts the frequencies of each categories.

And when to use a bar plot to represent the frequency distribution of any categorical variables, and the bar diagram makes it easier to compare sets of data between different groups. If you have a free categorical variable and if you have so many categories that under that variable if you want to look at distribution of each variable or how each category is distributed, then bar diagram is easier to interpret more on the categories of a variable.

(Refer Slide Time: 14:26)



Bar plot

```
counts = [979, 120, 12]
fuelType = ('Petrol', 'Diesel', 'CNG')
index = np.arange(len(fuelType))
```

x height of the bars
↓ ↓

```
plt.bar(index, counts, color=['red', 'blue', 'cyan'])
plt.title('Bar plot of fuel types')
plt.xlabel('Fuel Types')
plt.ylabel('Frequency')
plt.xticks(index, fuelType, rotation = 90)
plt.show()
```

Set the labels of the xticks
Set the location of the xticks

Python for Data Science

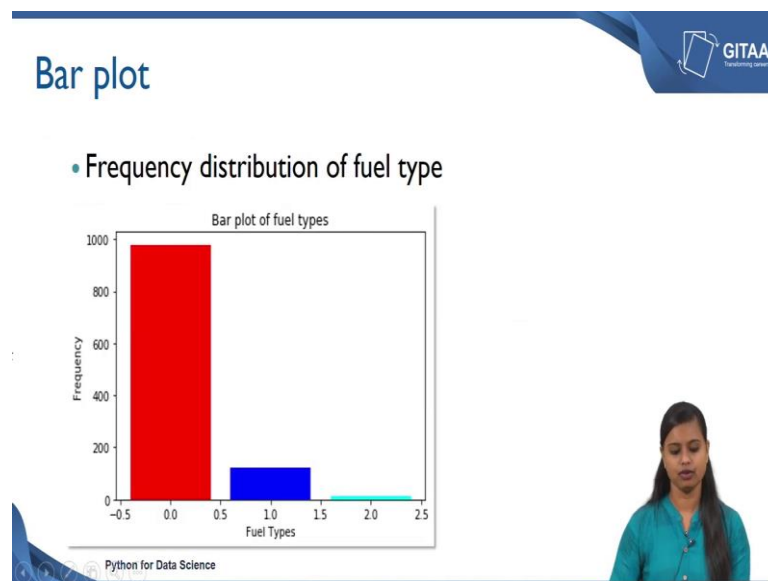
GITAA
Transforming careers

Now, let us see how to create a bar plot, basically you have to have an array which shows the counts of each categories. And the next object or the next variable represents a fuelType here we have petrol diesel and CNG as fuelTypes of the cars. And since we know that apriori have just given otherwise you can also give value_counts as an input,

because that will have the category as well as the frequencies. And I have given index because on the x-axis, it should know the index the index is whatever is the length of fuelType what is the length of the fuelType it would be 3. So, the range would be 0, 1 and 2. And to create a bar plot, bar is the function that is from the pyplot sub library; and inside the function, you just basically need to give what should be there in the x-axis and what should be there in the y-axis.

And x-axis I want the index to be present, and on the y-axis I want the counts to be present. And I am going to color my bars differently red will go for petrol and diesel will be colored based on blue color and CNG will be colored based on cyan color. And I have also given the title and the labels for x and y-axis, so that I can create a bar plot which will show the frequencies of the fuelType categories. And as I have mentioned earlier index is nothing but x coordinate and count will represent the height of the bar, at height of the bars have been represented using counts.

(Refer Slide Time: 16:10)

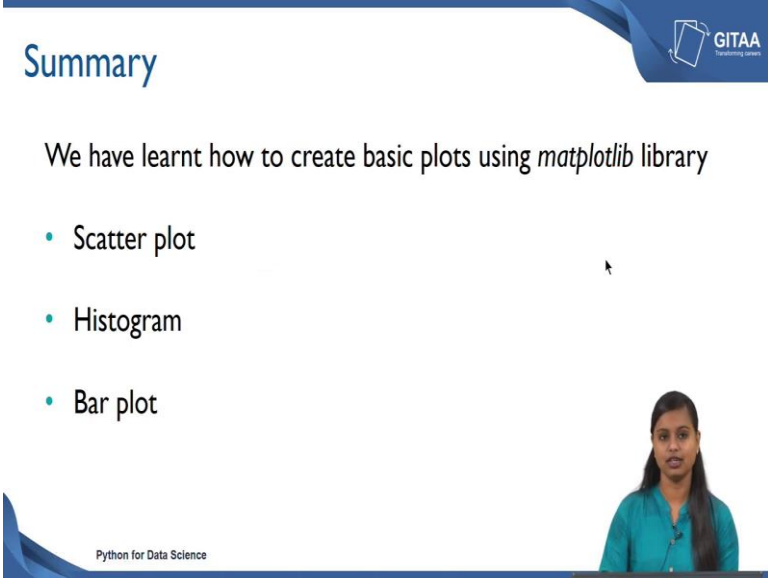


So, this is the output that we got. But if you see on the x-axis, you have only the range. Here we are not able to find out which is petrol which is diesel or which is CNG, in that case you can also specify the labels to your x-axis by giving xticks. So, all the other codes reminds the same, but since I just want to add labels to each of the bars on the x-axis that can be given using xticks you can set the labels of the xsticks using fuelType, because you have already assigned the variable called fuelType with some string values,

so that will be labels for the xticks. You can also set the location for x ticks where petrol should be present and diesel should be present and CNG should be present. And I have given rotation is equal to 90, so that all the labels will be 90 degree rotation.

So, now let us see how the plot will look like. Here if you see this the first bar is corresponding to the petrol fuelType and it is very evident that most of the cars around are 900 and odd cars have the fuelType as petrol, and there are only few cars for which the fuelType is about diesel and CNG.

(Refer Slide Time: 17:32)



Summary

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

- Scatter plot
- Histogram
- Bar plot

Python for Data Science

Now, we have come to the end of the session. In this lecture, we have learned how to create basic plots using matplotlib library. Those basic plots were scatter plot, histogram and bar plot.

Thank you.