Data Science with Python

Data Visualization

Agenda

01 Introduction to Seaborn

02 Matplotlib vs Seaborn

03 Seaborn Functions

04 Countplots

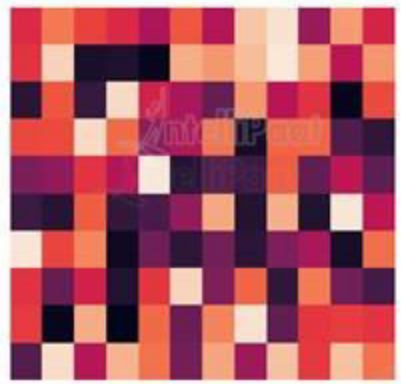
05 Heatmaps

06 Colormaps

Introduction to Seaborn

Introduction to Seaborn

Seaborn is a library that is used for making statistical graphics using Python scripts. It is built on top of Matplotlib and is also compatible with Pandas data structures



Seaborn

Introduction to Seaborn

What are the features of Seaborn?

1. Built-in themes to style up Matplotlib graphs and plots
2. Automatic estimation and plotting of linear regression models
3. Tools to choose color palettes to reveal patterns
4. A dataset-oriented API to compare variables
5. Visualizing univariate or bivariate distributions

Matplotlib vs Seaborn

Matplotlib vs Seaborn

Matplotlib

1. Used for basic plotting and contains bars, lines, and pies
2. A graphics package for data visualization and can mirror MATLAB
3. Multiple figure functions can be opened but have to closed as well
4. Works well with DataFrames and arrays and has some helpful APIs
5. Great customization ability

Seaborn

1. Has more interesting default themes with fewer syntax
2. Better integration with Pandas and also extends Matplotlib for better graphics
3. Automated creation of multiple figures
4. Works with a whole dataset instead of the data structures
5. Provides commonly used templates by default and saves time

Seaborn Functions

Seaborn Functions

If we are good at Matplotlib, then we are already halfway through Seaborn

Importing Seaborn

Import seaborn as sns

```
In [1]: import matplotlib.pyplot as plt
        import seaborn as sns
```

Loading a Dataset

sns.load_dataset("datasetname")

```
In [10]: titanic = sns.load_dataset("titanic")
```

Seaborn Functions

Various Functions with Examples

Function Type	Function	Description
Numerical Data Plotting	**relplot()**	Provides an interface for drawing relational plots
	scatterplot()	Draws a scatter plot
Categorical Data Plotting	**boxplot()**	Shows distribution w.r.t. categories
	catplot()	Provides an interface to draw categorical plots
Linear Regression and Relationship	**regplot()**	Plots data and a linear regression model fit
	lmplot()	Provides an interface to fit regression models across the subsets of a dataset
Visualizing Data Distribution	**jointplot()**	Plots two variables with bivariate or univariate graphs
	distplot()	Plots a univariate distribution of observations

Countplots

Countplots

They show the counts of observations in each column mentioned using a bar chart. It is a histogram for a categorical variable. We can easily create a histogram of the number of ‘Males’ and ‘Females’ by providing a ‘Gender’ column as the input

Countplots

Loading the dataset on which we want to perform operations

```
In [10]: titanic = sns.load_dataset("titanic")
```

Analyzing the dataset to get the column names and the first five records

```
In [11]: titanic.head()
```

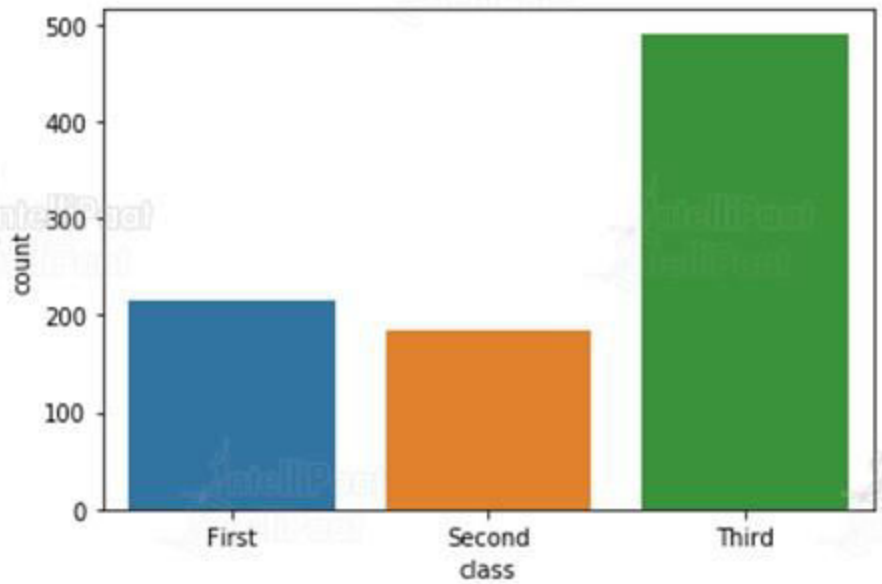
Out[11]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True

Countplots

Creating a countplot on the column **class** to group by unique values

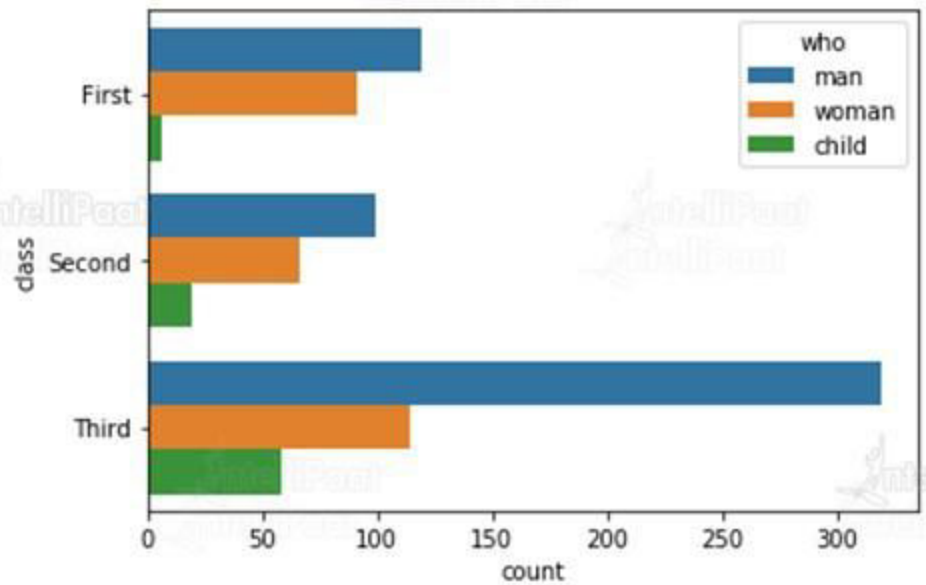
```
In [12]: sns.countplot(x="class", data=titanic)
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x26c2ce5fe48>
```



Countplots

A horizontal countplot with a hue

```
In [13]: sns.countplot(y="class", hue="who", data=titanic)
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x26c2cea4a20>
```

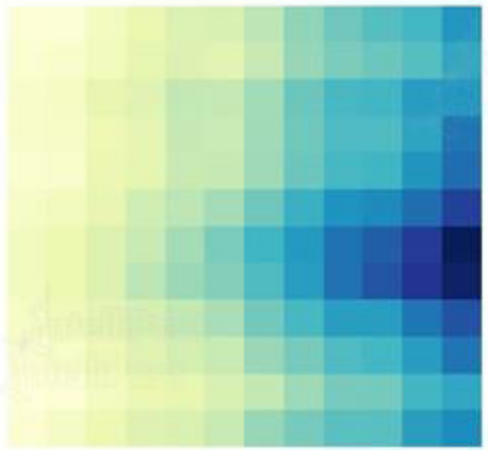


Hands-on: Countplots

Heatmaps

Heatmaps

Heatmaps plot data as a rectangular color-encoded matrix, which is customizable. We can use a heatmap to visualize data from one or multiple columns or through lists and arrays



Heatmaps

Creating a heatmap of a 2D matrix

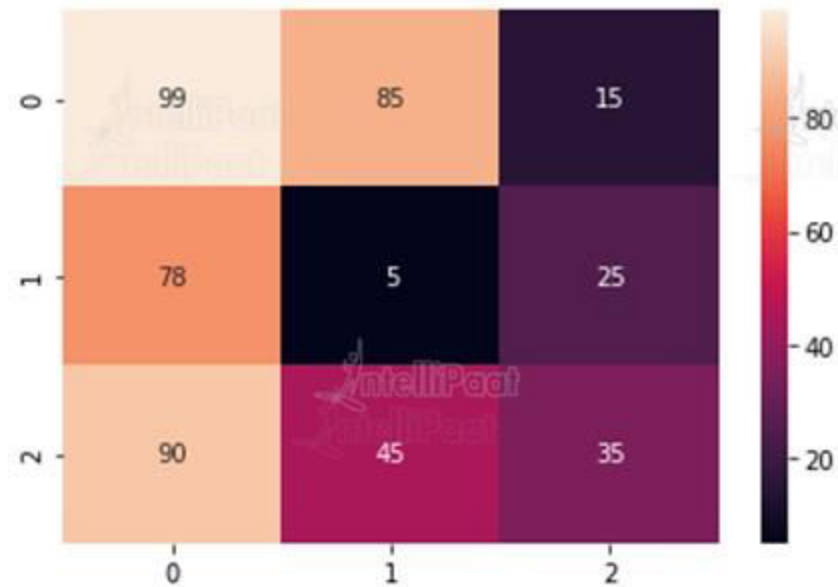
Code for the heatmap

```
In [14]: data = [
             [99, 85, 15],
             [78, 5, 25],
             [90, 45, 35],
         ]

         sns.heatmap(data, annot=True)
```

Heatmap

```
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x26c2cf6ec18>
```



Hands-on: Heatmaps

Colormaps

Colormaps

Color is the most important aspect of visualization. If used properly, it can reveal hidden patterns which cannot be found in plain data. But if used poorly, it may hide the very patterns. We can use colormaps by choosing from the default palettes suitable for the type of data we are visualizing

Colormaps

How to check the current default theme?

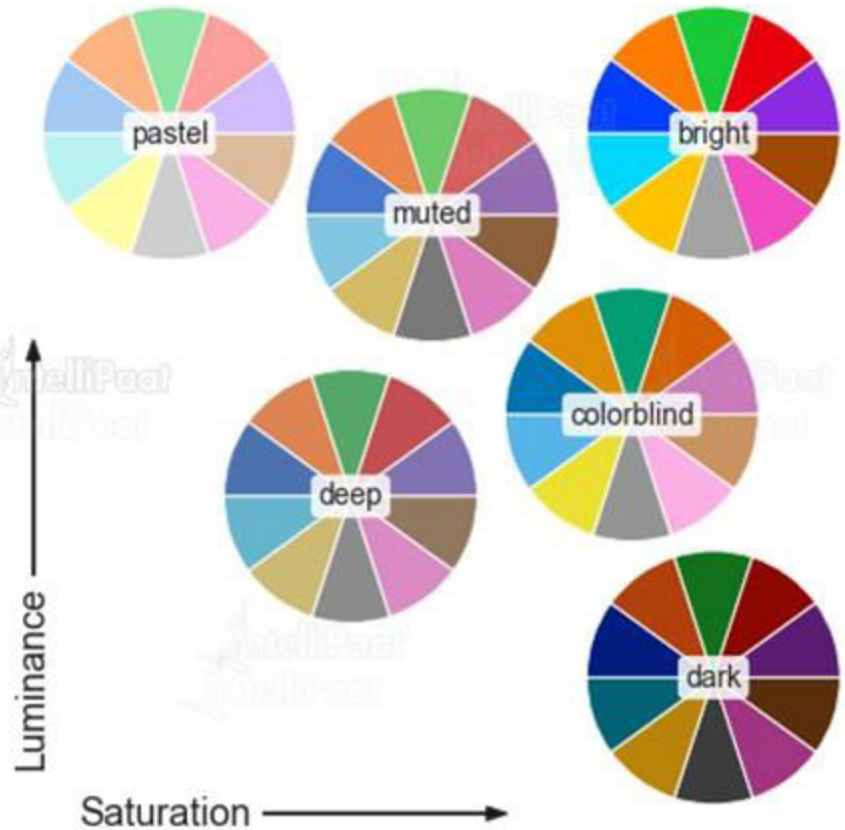
```
sns.palplot(current_palette)
```

How to create a palette of colors we want?

```
mypalette = ["#9b59b6", "#3498db", "#95a5a6"]
sns.palplot(sns.color_palette(mypalette))
```

Colormaps

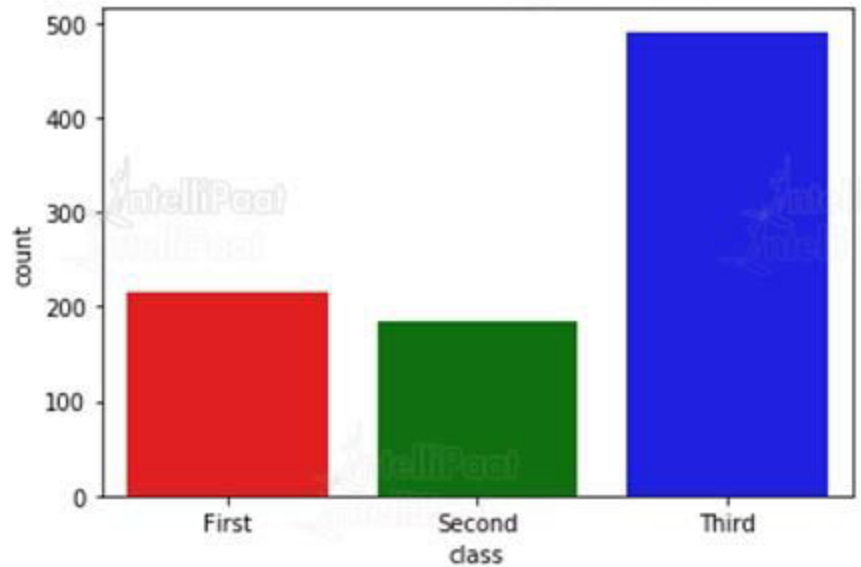
There are six variations of the default theme



Colormaps

Creating a count plot with the custom color palette

```
In [15]: sns.countplot(x="class", data=titanic, palette=['red', 'green', 'blue'])
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x26c2d030898>
```

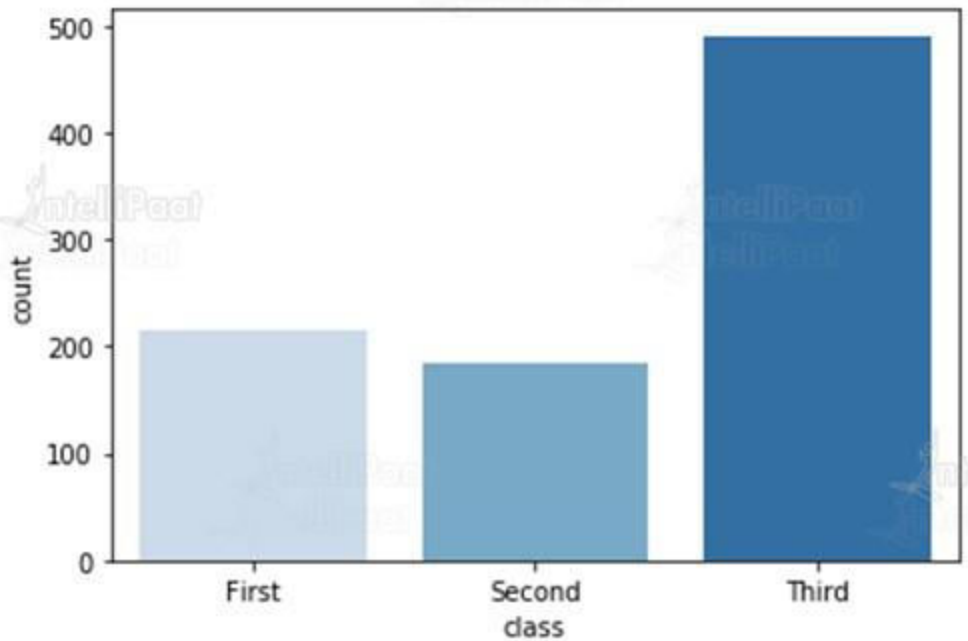


Colormaps

Creating a count plot that uses a built-in color palette

```
In [16]: sns.countplot(x="class", data=titanic, palette="Blues")
```

```
Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x26c2cd5d240>
```



Hands-on: Colormaps

India: +91-7847955955

US: 1-800-216-8930 (TOLL FREE)

support@intellipaat.com

24/7 Chat with Our Course Advisor