Data Visualization

Why is data visualization important?

- Because of the way the human brain processes information, using charts or graphs to visualize large amounts of complex data is easier than poring over spreadsheets or reports. Data visualization is a quick, easy way to convey concepts in a universal manner

Data visualization can also:

- Identify areas that need attention or improvement.
- Clarify which factors influence customer behavior.
- Help you understand which products to place where.
- Predict sales volumes.

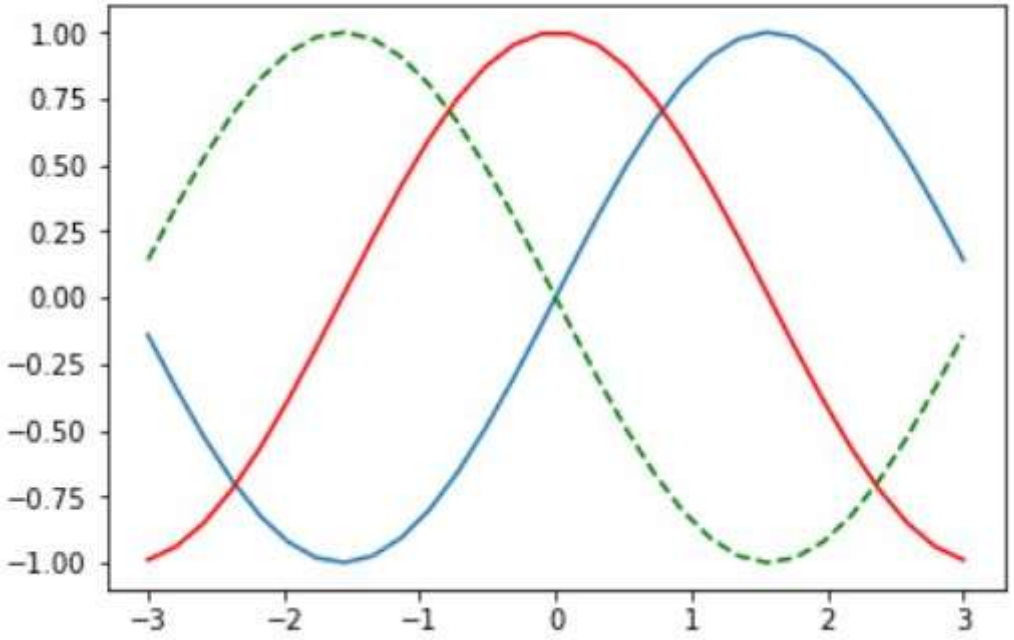
DIFFERENT TYPES OF DATA VISUALIZATION

Basic types of plot

- line plot
- bar plot
- Pie chart
- Scatter plot
- Histogram

line plot

line chart or **line graph** is a type of chart which displays information as a series of data points called 'markers' connected by straight **line** segments.



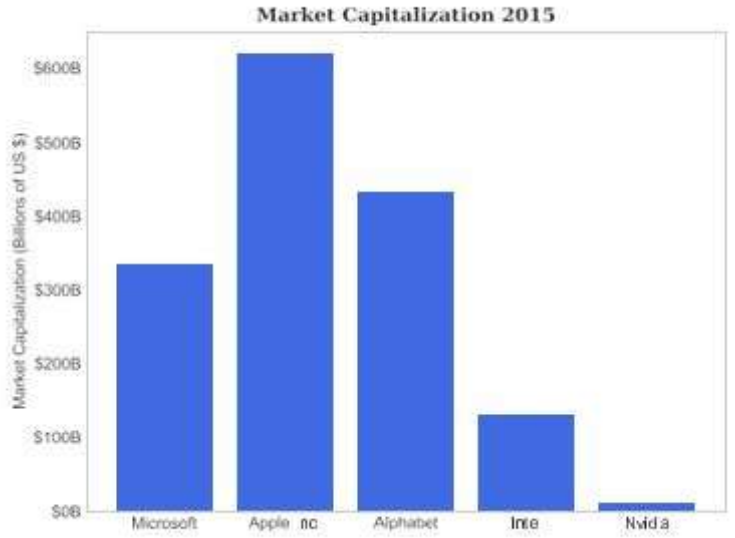
Bar Chart

- **Bar Chart** Shows comparisons between different categories, different parts of a whole.
- Shows relationship between a numerical variable and a discrete variable.

- **Variations of Bar Chart :**
  - **Vertical Bar Chart or Column Chart :** Best used to visualize relationship or comparisons with chronological data.
  - **Horizontal Bar Chart :** Useful when category labels are long.
  - **Stacked Bar Chart :** Used to compare different parts of a whole using discrete or continuous variables.
  - **Grouped Bar Chart :** Used to compare multiple data series in a given category.

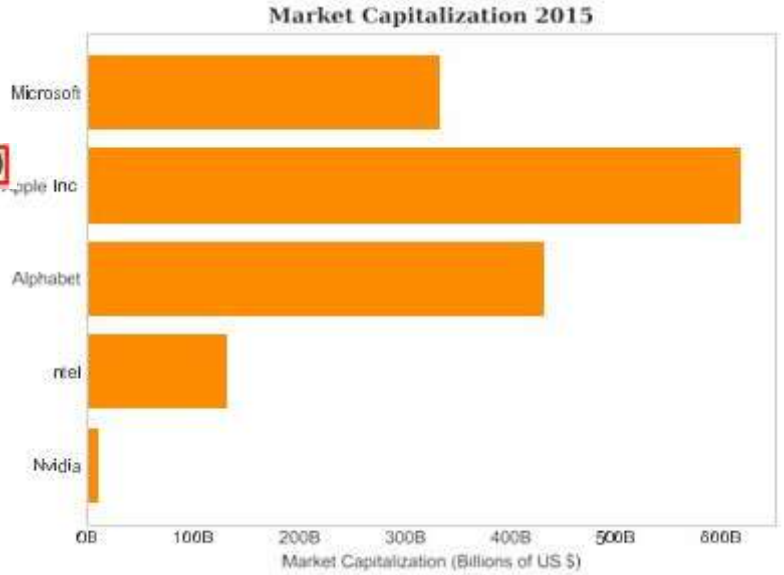
Vertical Bar Chart

```
plt.bar(y, height=market_cap, align='center', color='royalblue')
```



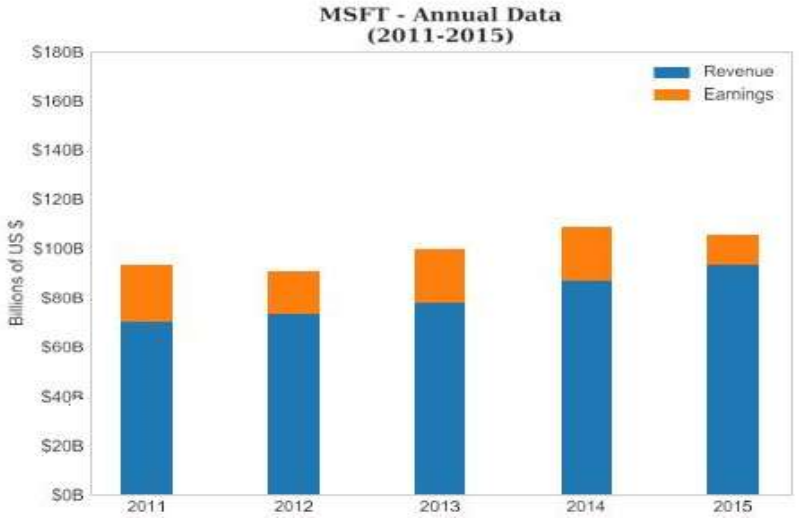
Horizontal Bar Chart

```
ax.barh(y, width=market_cap, align='center', color='darkorange')
```



Stacked Bar Chart

```
p1 = plt.bar(x, revenue, width)
p2 = plt.bar(x, earnings, width, bottom=revenue)
```



Grouped Bar Chart

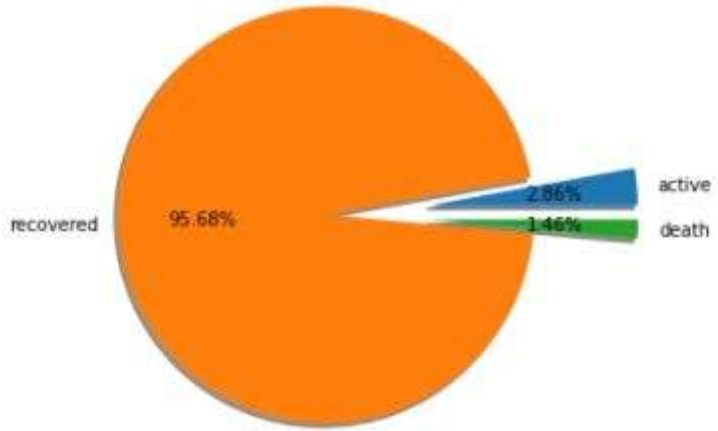
```
fig, ax = plt.subplots()
bar1 = ax.bar(x - width/2, revenue, width, color='royalblue', label='Revenue')
bar2 = ax.bar(x + width/2, earnings, width, color='mediumseagreen', label='Earnings')
```

Pie Chart

- **Pie Chart** is a circular statistical graphic, which is divided into slices to show part-to-whole relationships with continuous or discrete data.
- It is best used with a small data.

•Functions for plotting Pie Chart :

```
axes.pie(df_data['Number of cases'],labels=df_data.index,autopct='%0.2f%%',shadow=True,explode=[0.5,0,0.5])
```



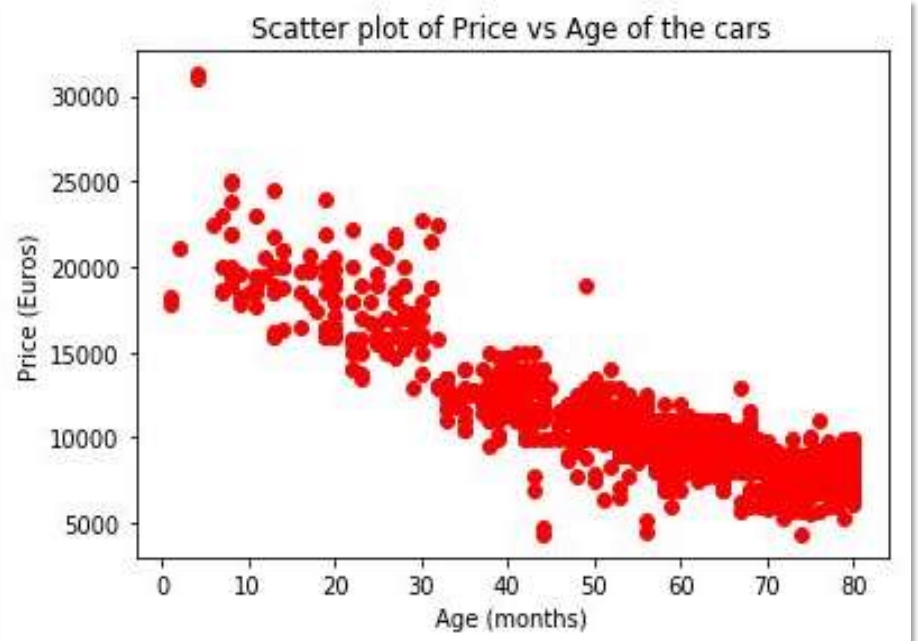
Scatter Plot

- A scatter plot is a set of points that represents the values obtained for two different variables plotted on a horizontal and vertical axes

- When to use scatter plots?
- Scatter plots are used to convey the relationship between two numerical variables
- Scatter plots are sometimes called correlation plots because they show how two variables are correlated
- Pairing regardless of time

```
plt.scatter(cars_data['Age'], cars_data['Price'], c='red')
```

```
sns.regplot(x=cars_data['Age'], y=cars_data['Price'])
```



• Histogram

•What is a histogram?

- It is a graphical representation of data using bars of different heights
- Histogram groups numbers into ranges and the height of each bar depicts the frequency of each range or bin

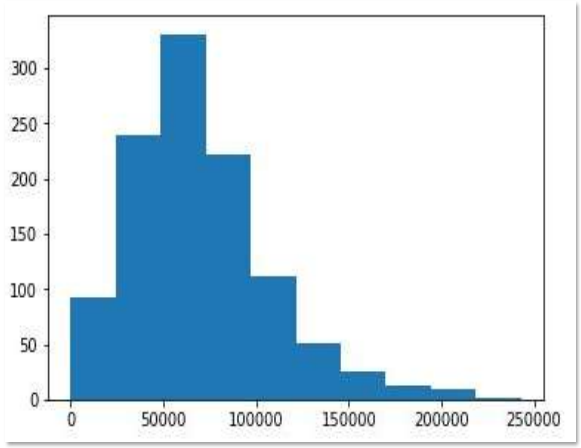
•When to use histograms?

- To represent the frequency distribution of numerical variables

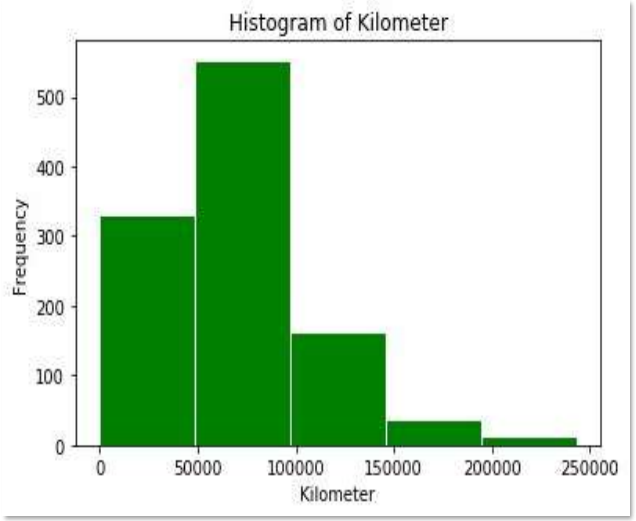
- Histogram

```
plt.hist(cars_data['KM'])
```

Histogram with default arguments



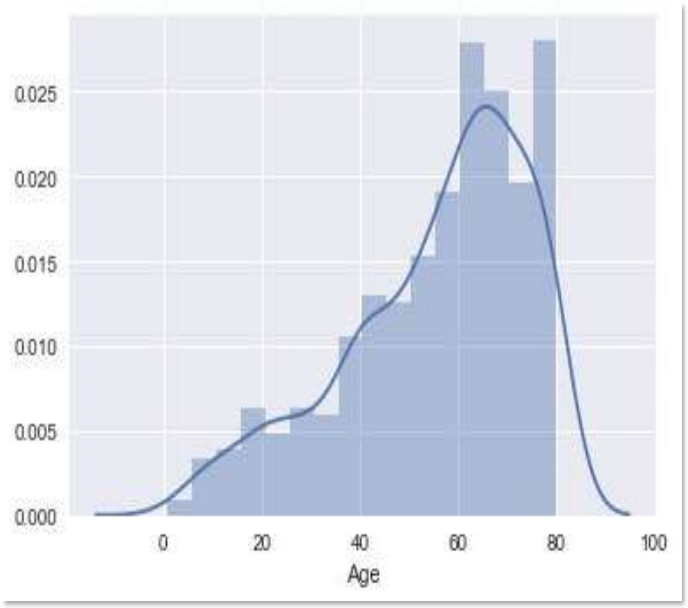

```
plt.hist(cars_data['KM'],
         color     = 'green',
         edgecolor = 'white',
         bins      = 5)
plt.title('Histogram of Kilometer')
plt.xlabel('Kilometer')
plt.ylabel('Frequency')

plt.show()
```



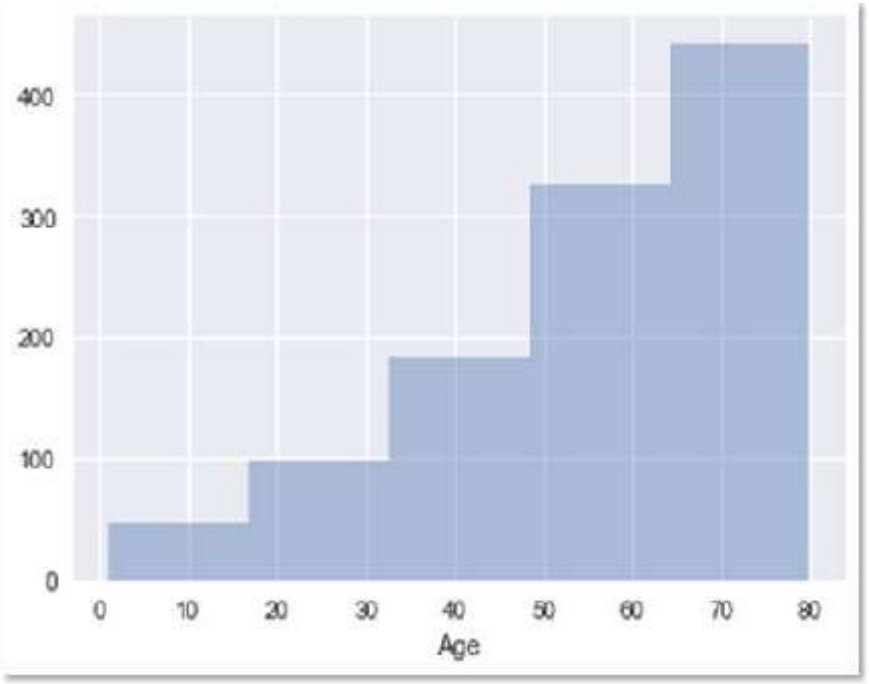
Histogram

- Histogram with and without default kernel density estimate

```
sns.distplot(cars_data['Age'] )
```




```
sns.distplot(cars_data['Age'], kde = False, bins=5 )
```



Categorical Data Plots

Different types of plot for this:

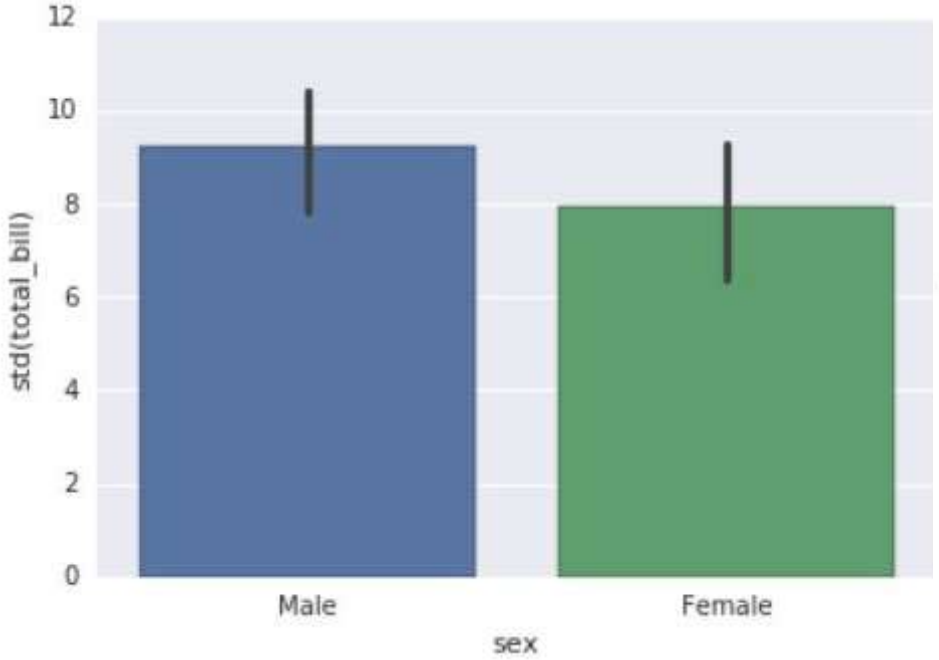
- barplot
- countplot
- boxplot
- violinplot
- stripplot
- Swarmplot
- factorplot

Categorical Data Plots

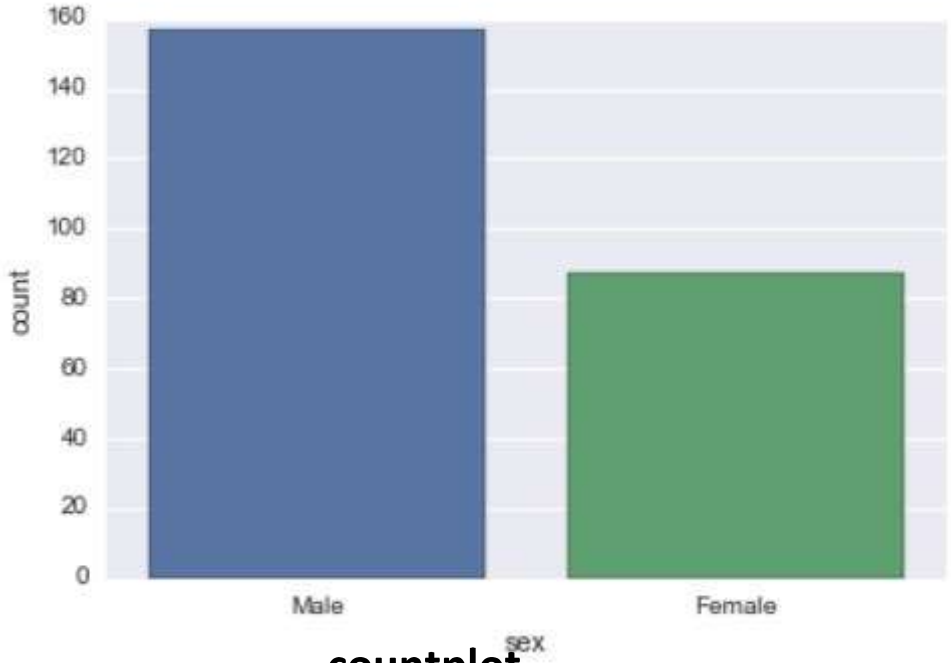
- **barplot and countplot**

barplot is a general plot that allows you to aggregate the categorical data based off some function, by default the mean.

countplot is essentially the same as barplot except the estimator is explicitly counting the number of occurrences.



barplot

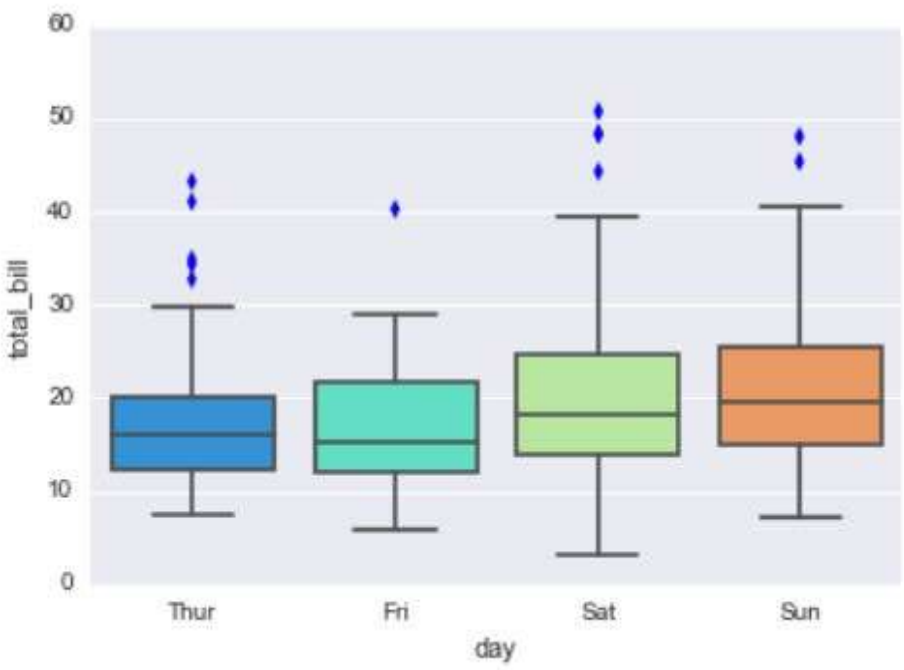


countplot

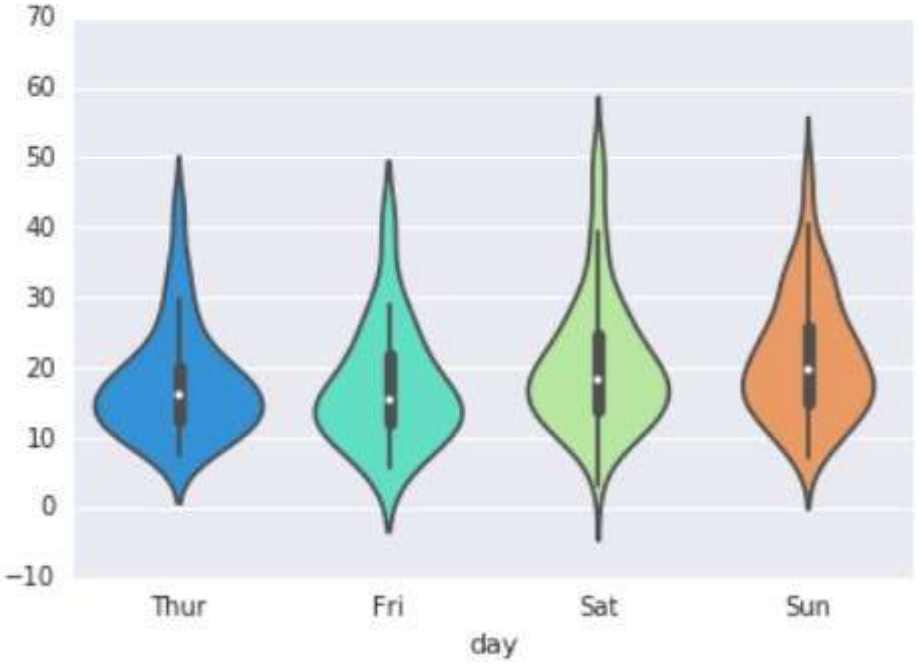
• boxplot and violinplot

The box shows the quartiles of the dataset while the whiskers extend to show the rest of the distribution, except for points that are determined to be “outliers” .

A violin plot plays a similar role as a box and whisker plot. Unlike a box plot, in which all of the plot components correspond to actual datapoints, the violin plot features a kernel density estimation of the underlying distribution.

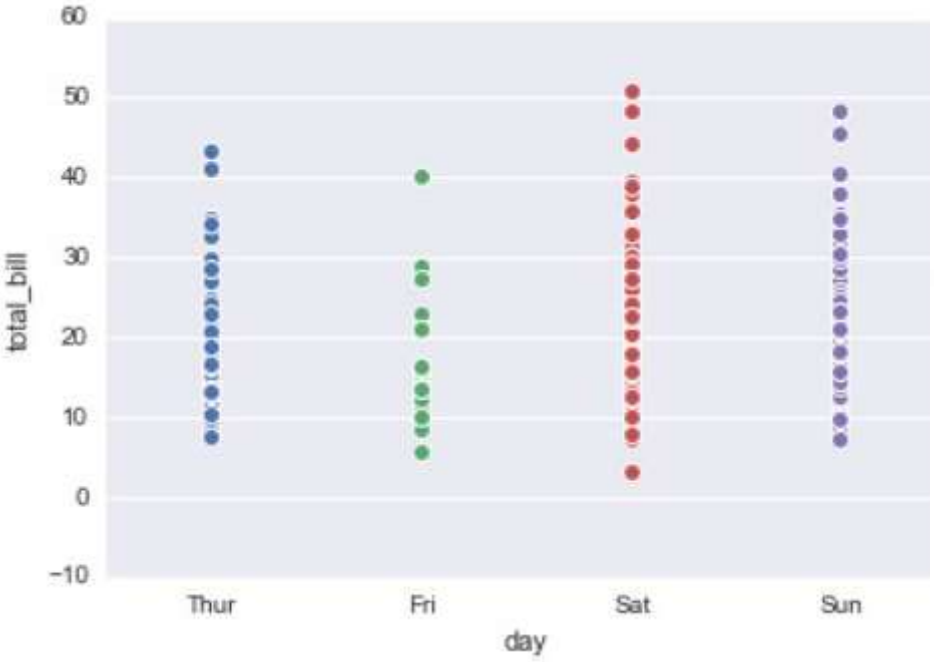


boxplot

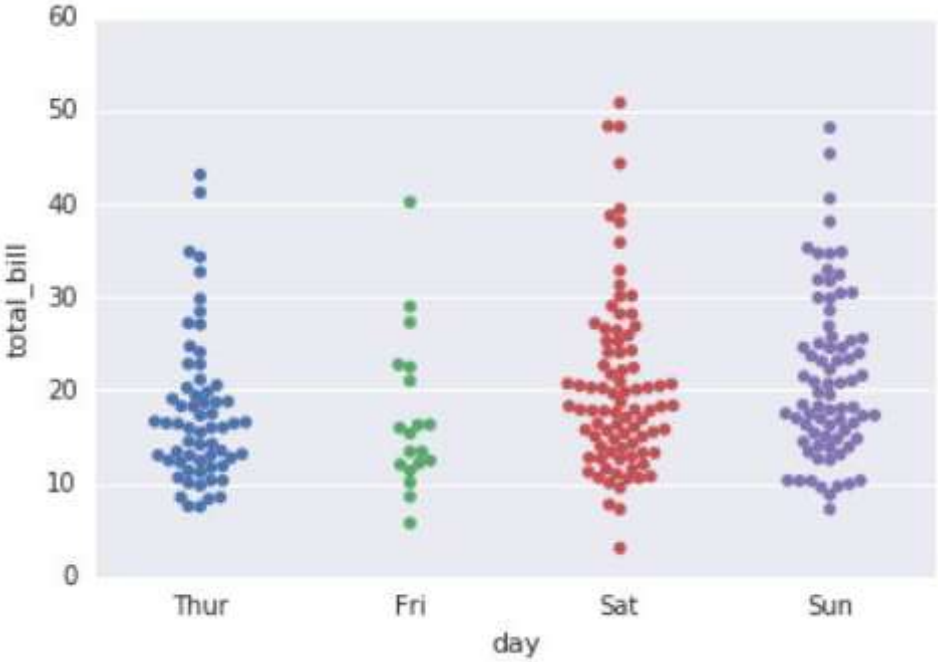


violinplot

- **stripplot and swarmplot**
- A strip plot can be drawn on its own, but it is also a good complement to a box or violin plot in cases where you want to show all observations along with some representation of the underlying distribution.
- The swarmplot is similar to stripplot(), but the points are adjusted (only along the categorical axis) so that they don't overlap. This gives a better representation of the distribution of values.



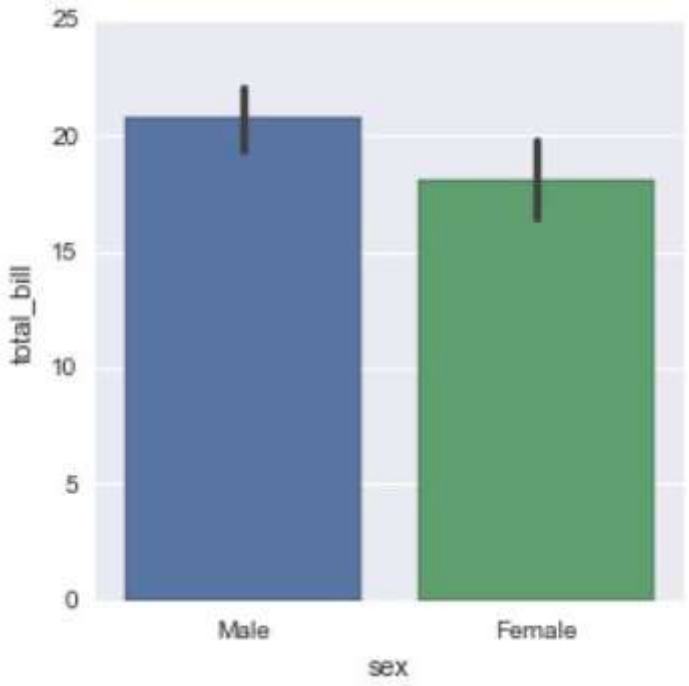
strip plot



swarmplot

- **Factorplot**

factorplot is the most general form of a categorical plot. It can take in a **kind** parameter to adjust the plot ty

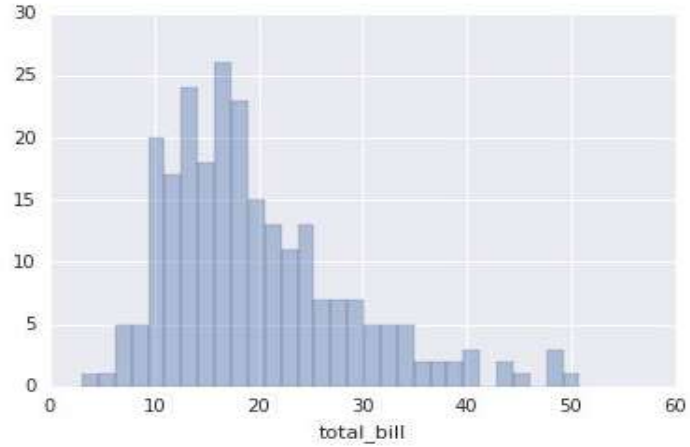


Distribution Plots

These plots allow us to visualize the distribution of a data set.

- distplot
- jointplot
- rugplot
- kdeplot
- pairplot

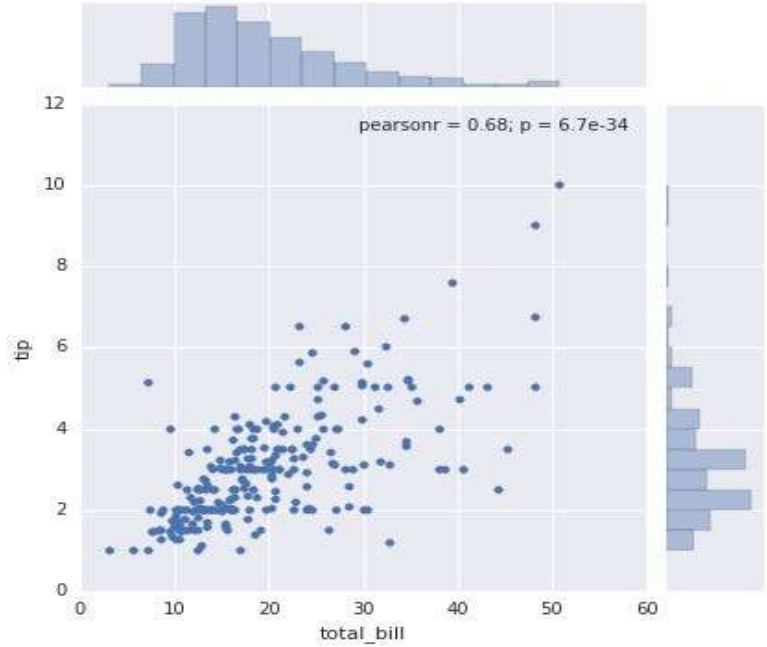
- **Distplot:**The distplot shows the distribution of a univariate set of observations.



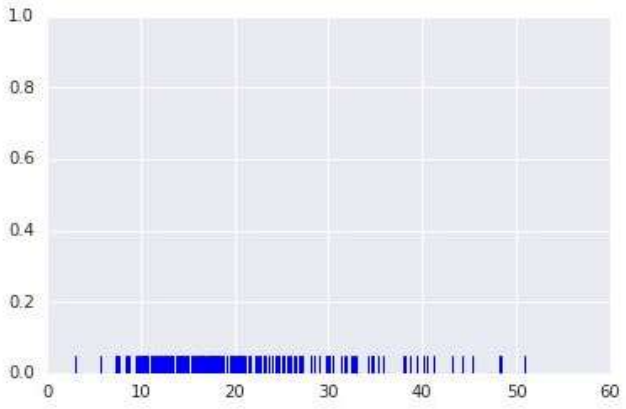
- **jointplot**

jointplot() allows you to basically match up two distplots for bivariate data.

- "scatter"
- "reg"
- "resid"
- "kde"
- "hex"

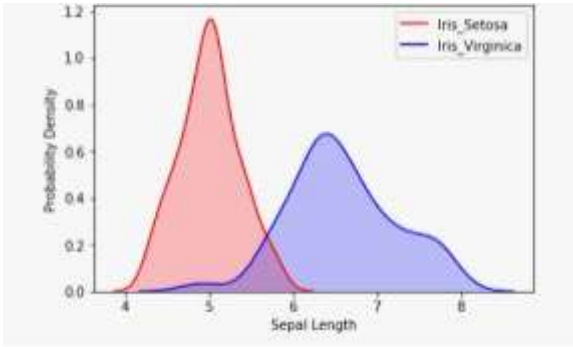


- **rugplot:** rugplots are actually a very simple concept, they just draw a dash mark for every point on a univariate distributior



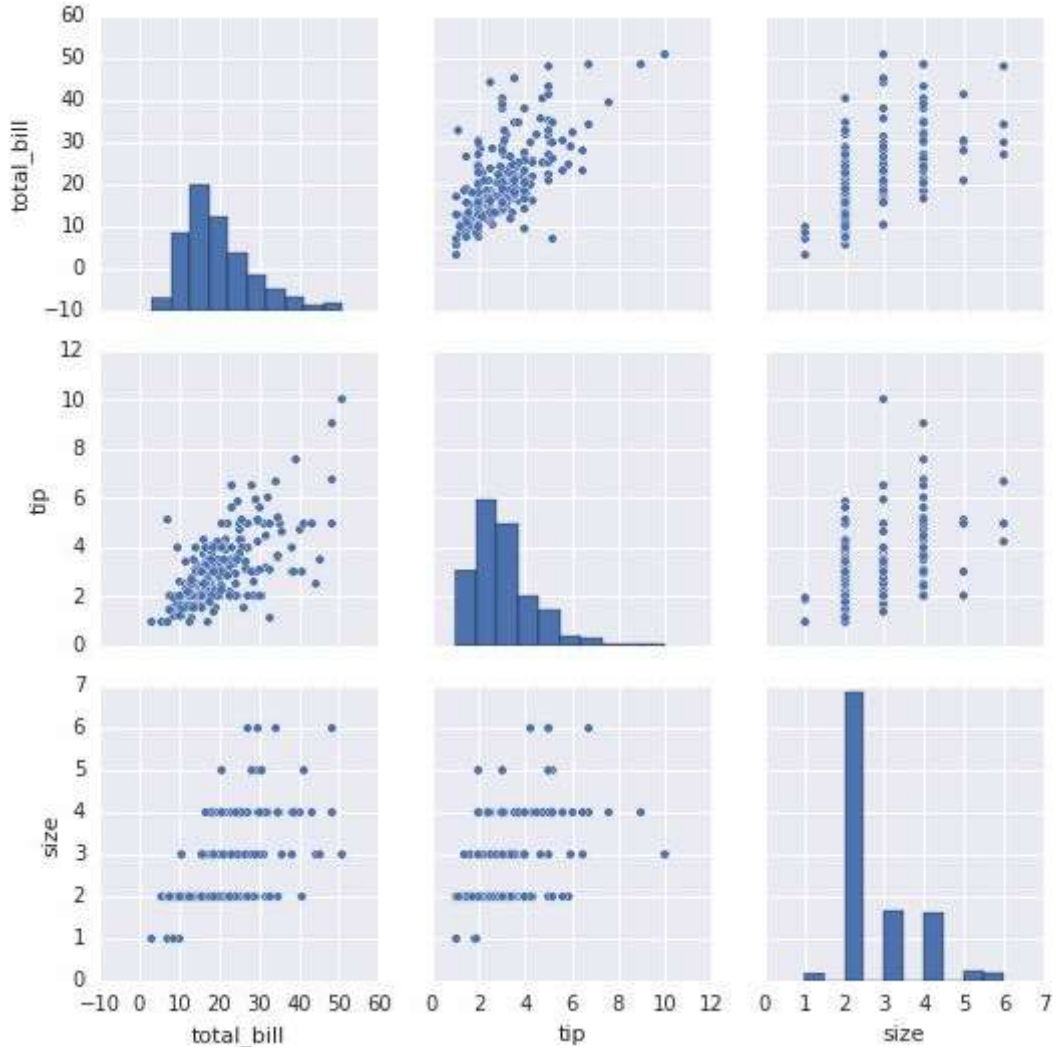
- **kdeplot**

kdeplots are Kernel Density Estimation plots. These KDE plots replace every single observation with a Gaussian (Normal) distribution centered around that value.



- **pairplot**

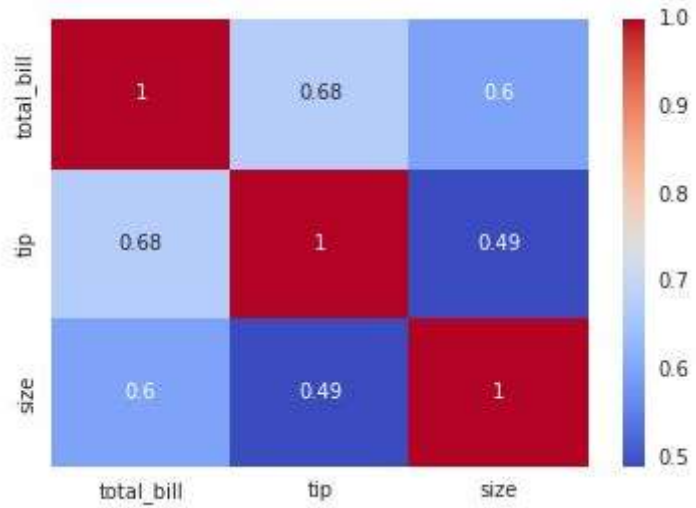
pairplot will plot pairwise relationships across an entire dataframe (for the numerical columns) and supports a color hue argument (for categorical columns).



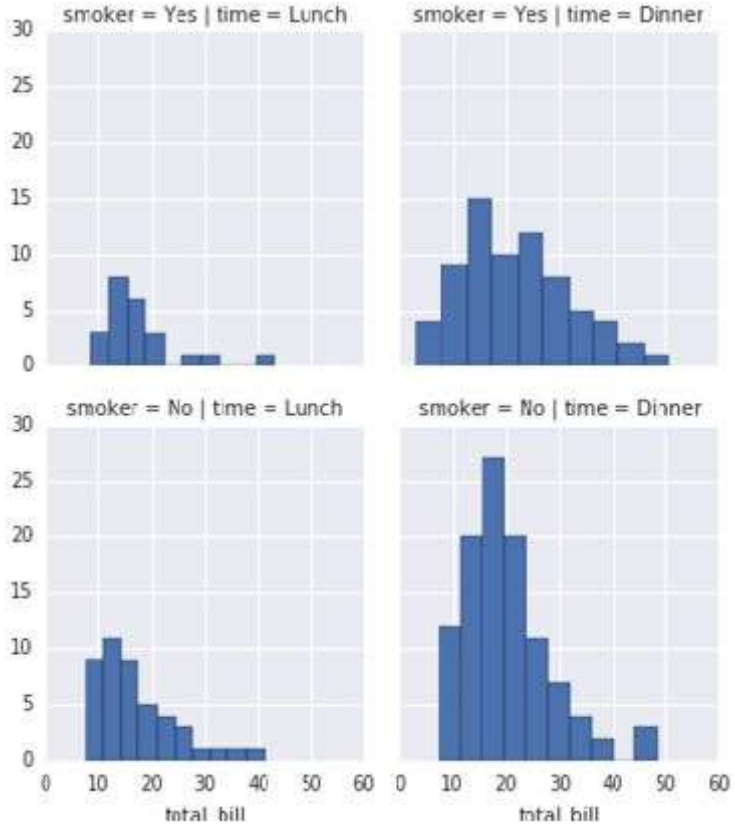
Matrix, Grid and Regression plots

- Matrix plots allow you to plot data as color-encoded matrices and can also be used to indicate clusters within the data **eg HEATmap**
- Grid plots are general types of plots that allow you to map plot types to rows and columns of a grid, this helps you create similar plots separated by features.**eg Pairplot and Facetgrid**
- Regression plots allows you to display linear models, but it also conveniently allows you to split up those plots based off of features, as well as coloring the hue based off of features.**eg IMplot**

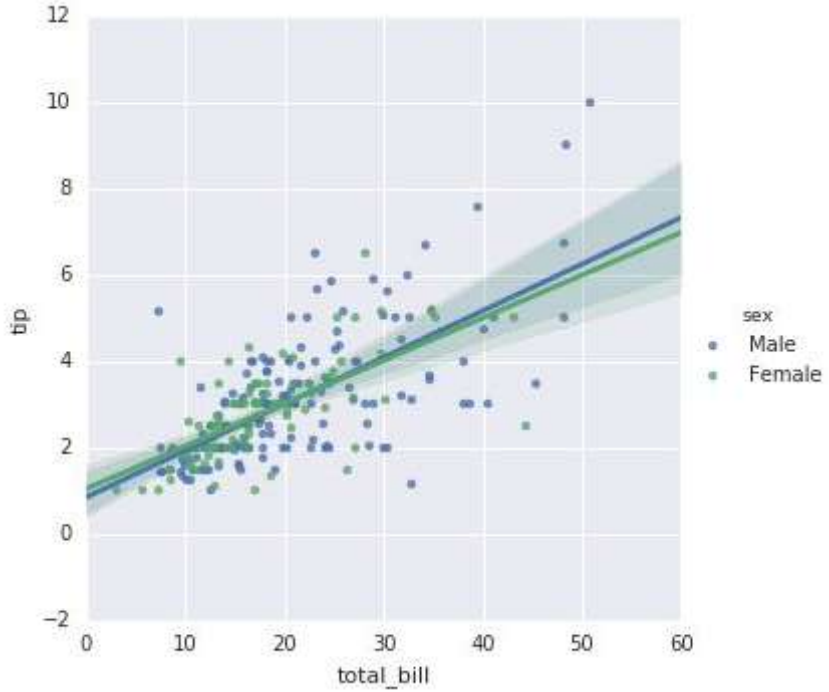
- Heatmap
- A Heatmap is a graphical representation of data where the individual values contained in a matrix are represented as colors. Heatmaps are perfect for exploring the correlation of features in a dataset.



- **Facet Grid:** Faceting is the act of breaking data variables up across multiple subplots and combining those subplots into a single figure.
- **lmplot plot** is used to draw a scatter plot onto a FacetGrid.



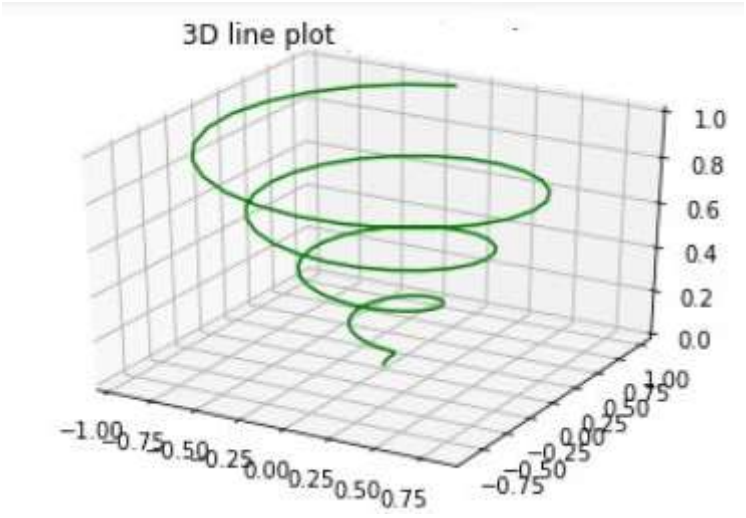
Facet Grid



lmplot plot

Other Plots

- **3-D plots**: 3D utilities were developed upon the 2d and are enabled by importing the mplot3d toolkit.
- Graph with lines and point are the simplest 3 dimensional graph. ax.plot3d and ax.scatter are the function to plot line and point graph respectively.



- Geospatial plot

- Folium makes it easy to visualize data that's been manipulated in Python, on an interactive Leaflet map. This library has a number of built-in tilesets from OpenStreetMap, Mapbox etc.