Lesson A-4

Getting to Know Your Data

- Data Exploration (Visualization)

Data Visualization

- *"Data visualization is the graphic representation of data"*
- *"The main goal of data visualization is to communicate information clearly and effectively through graphical means"*

  Vitaly Friedman
- Data visualization is nowadays one of the most important tasks in data science
  - It is a part of the data exploratory phase of the data science process or part of modeling as presentation of the output

Data Visualization

- Univariate plots
  - Visualize each feature of dataset independently
  - Bar, histogram, density, box, line, area
- Multivariate plots
  - Visualize the correlations (dependencies, interactions) between multiple variables.
  - Scatter, matrix (heatmap), qq-plot

Types of plots

- line plot
- vertical bar plot
- horizontal bar plot
- histogram
- pie plot
- density plot (Kernel Density Estimation plot)
- area plot
- box (and whisker) plot
- scatter plot
- hexbin plot
- correlation matrix plot (heat map)
- KDE plot

Data Visualization in Python

- Matplotlib library
  - Python's the most popular visualization library
  - is more easily customizable
- Seaborn library
  - Python’s popular visualization library
  - is a library based on Matplotlib and closely integrated with pandas data
    - uses fewer syntax
  - has default themes
    - Provides visualization patterns
- Pandas plotting module
  - built-in plot functions available on its Series and DataFrame objects.

Matplotlib vs. Seaborn

- Functionality
  - Matplotlib is mainly deployed for basic plotting.
  - Seaborn provides a variety of visualization patterns.
- Handling Multiple Figures
  - Matplotlib has multiple figures can be opened, but need to be closed explicitly.
  - Seaborn automates the creation of multiple figures.
- Visualization
  - Matplotlib is a graphics package for data visualization in Python.
  - Seaborn is more integrated for working with Pandas data frames

Matplotlib vs. Seaborn

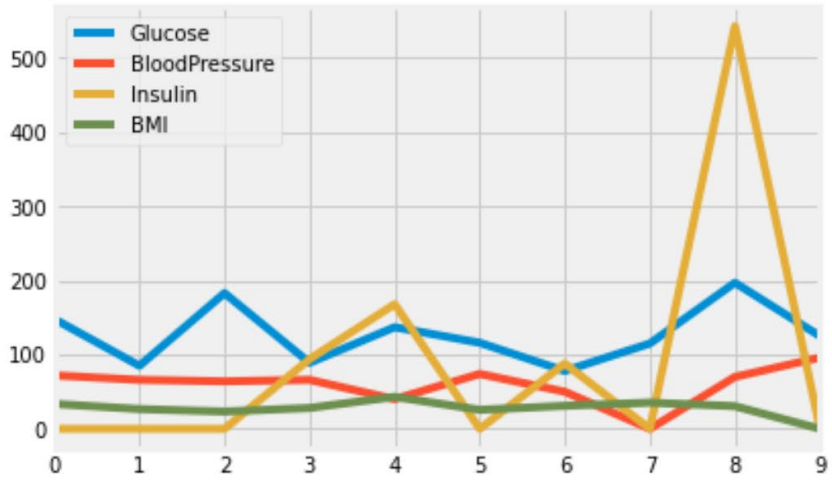
- Data frames and Arrays
  - Matplotlib works with data frames and arrays. It has different stateful APIs for plotting.
  - Seaborn works with the dataset as a whole and is much more intuitive than Matplotlib.
- Flexibility
  - Matplotlib is highly customizable and powerful.
  - Seaborn avoids a lot of boilerplate by providing default themes which are commonly used.

Pandas plotting module

- dataframe `plot()` functions create decent looking plots with the dataframe
- The plot function on Series and DataFrame is a simple wrapper around Matplotlib `plt.plot()`
  - you don’t need to write those long matplotlib codes for plotting.
  - if you want some advanced plots which cannot be done using the plot function then you can switch to matplotlib or seaborn

```
# Pandas Line Chart
# data.plot(data.index.name, ['Glucose','BloodPressure','Insulin','BMI'],kind = 'line')

df1=data[:10]
df1.plot(data.index.name, ['Glucose','BloodPressure','Insulin','BMI'],kind = 'line')
```

0]: <matplotlib.axes._subplots.AxesSubplot at 0x1d379e72508>

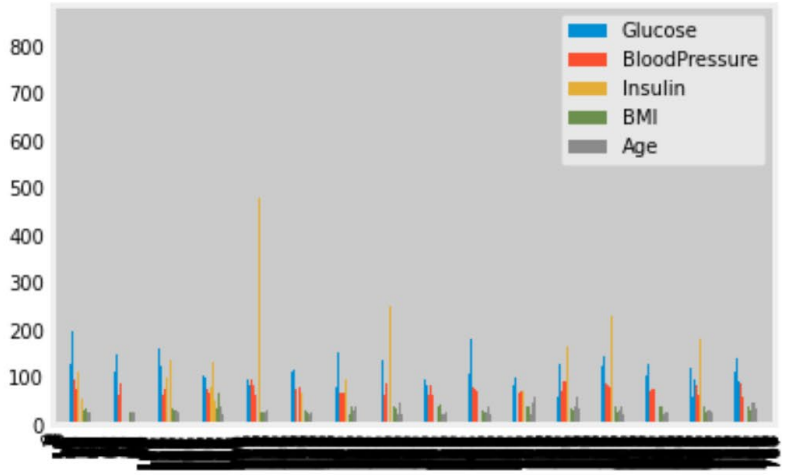


Line Graph

- A line graph, also known as a line chart, is a type of chart used to visualize the value of something over the other one (which is usually time)
  - The line graph is a efficient visual tool for time-series data such as finance, marketing, weather, laboratory data.
  - Data points are plotted and connected by a line

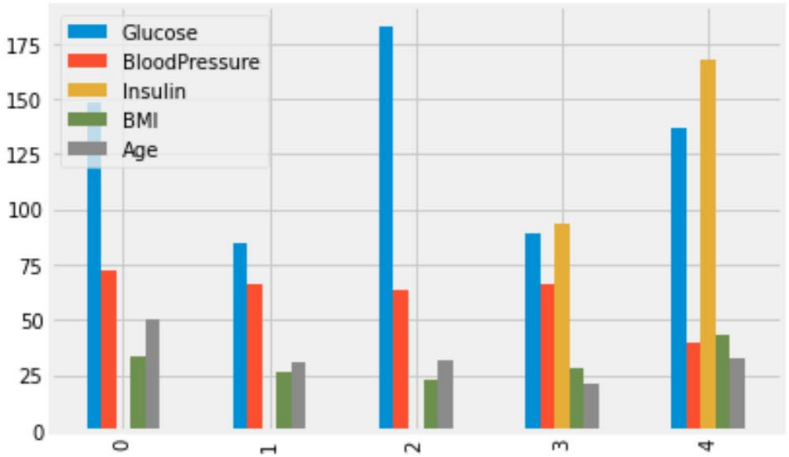
```
# Pandas bar plot
data.plot(data.index.name,['Glucose','BloodPressure','Insulin','BMI', 'Age'],kind = 'bar')
```

6]: <matplotlib.axes._subplots.AxesSubplot at 0x1d30afaf348>




```
# Pandas bar plot
df1=data[:5]
df1.plot(data.index.name,['Glucose','BloodPressure','Insulin','BMI', 'Age'],kind = 'bar')
```

6]: <matplotlib.axes._subplots.AxesSubplot at 0x1d37a2f2588>

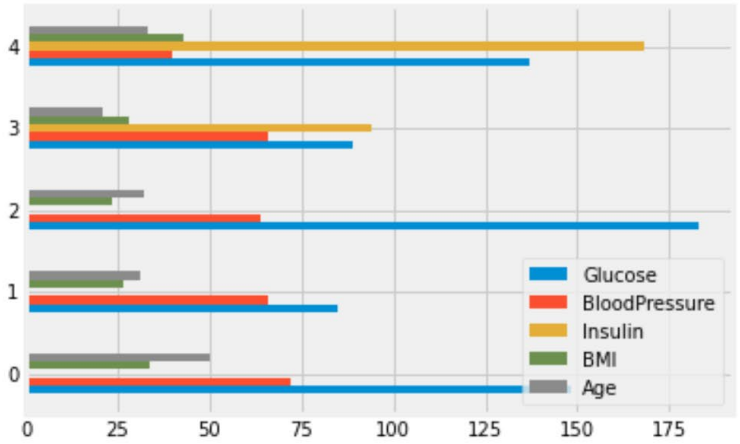


Bar Graph (Chart)

- A bar graph uses bars to compare data among categories.
- Bar graph presents categorical data with rectangular bars with heights or lengths proportional to the values that they represent.
- The bars can be plotted vertically or horizontally.
  - A vertical bar chart is sometimes called a column chart.
  - A bar graph is not good for a large number of categorical data

```
# Horizontal Bar with positions
df1.plot(data.index.name,[1, 2, 4, 5, 7],kind = 'barh')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d37a4606c8>
```

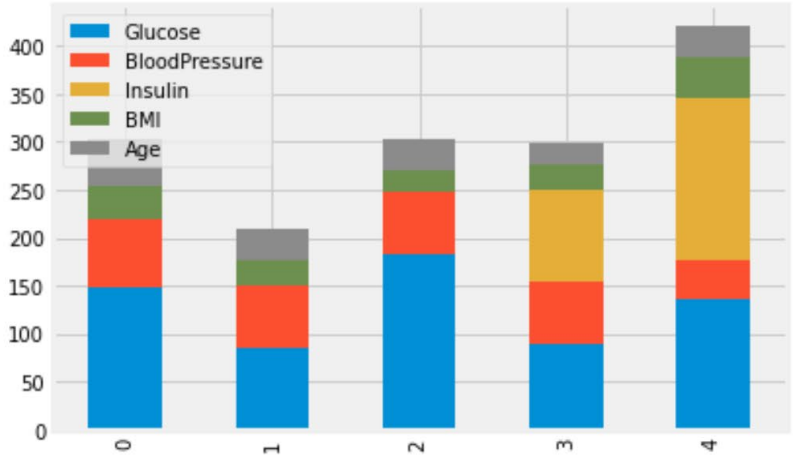


Horizontal Bar Chart

- represent the data horizontally.
  - bars are drawn horizontally.
  - data categories are shown on the vertical axis
  - data values are shown on the horizontal axis.
  - the length of each bar is equal to the value corresponding the data category and bars go across from left to right.

```
# Pandas stacked bar plot
df1.plot(data.index.name,['Glucose','BloodPressure','Insulin','BMI', 'Age'],kind = 'bar', stacked=True)
```

: <matplotlib.axes._subplots.AxesSubplot at 0x1d306c838c8>

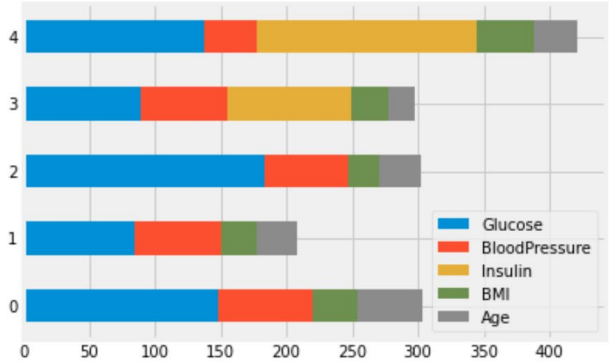


Stacked bar chart

- The stacked bar chart extends the standard bar chart from looking at numeric values across one categorical feature to multiple.
- Each bar in a standard bar chart is divided into a number of sub-bars stacked end to end, each one corresponding to a level of the corresponding categorical feature.

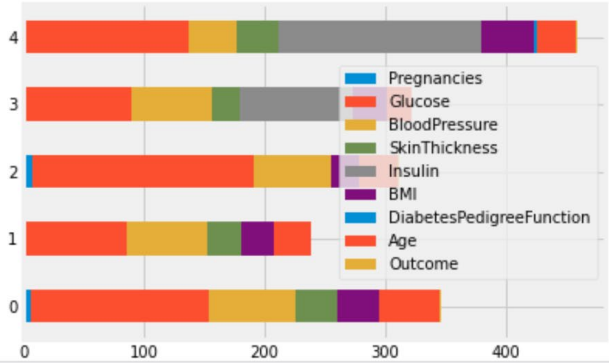
```
# Horizontal Stacked Bar with positions
df1.plot.barh(data.index.name,['Glucose','BloodPressure','Insulin','BMI', 'Age'], stacked=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d306d65f08>
```



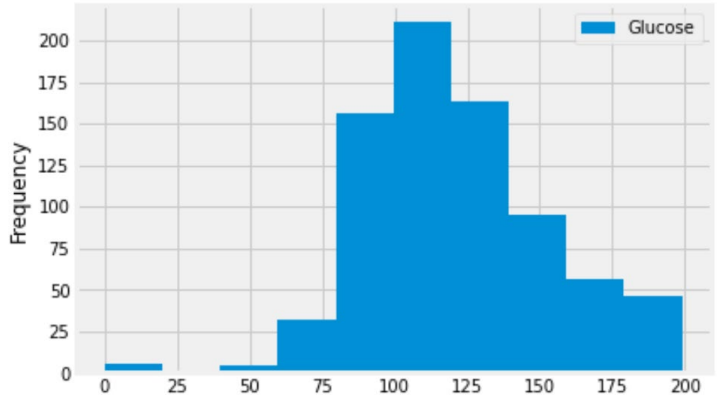

```
# Horizontal Stacked Bar with positions
df1.plot.barh(stacked=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d306dc0c88>
```



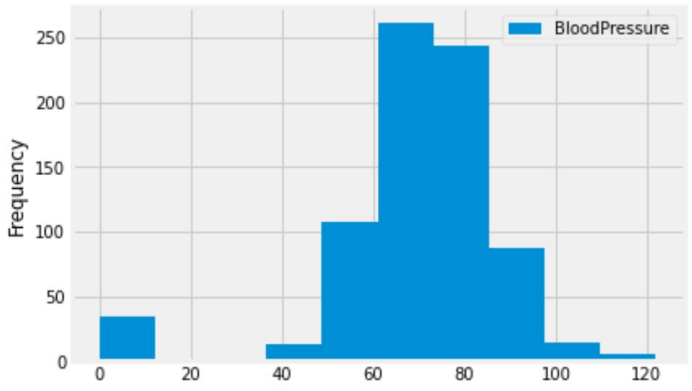
```
# Histogram
data.plot(data.index.name,[1], kind = 'hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d37a711588>
```




```
# Histogram
data.plot(data.index.name,[2],kind = 'hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d37a76ec48>
```

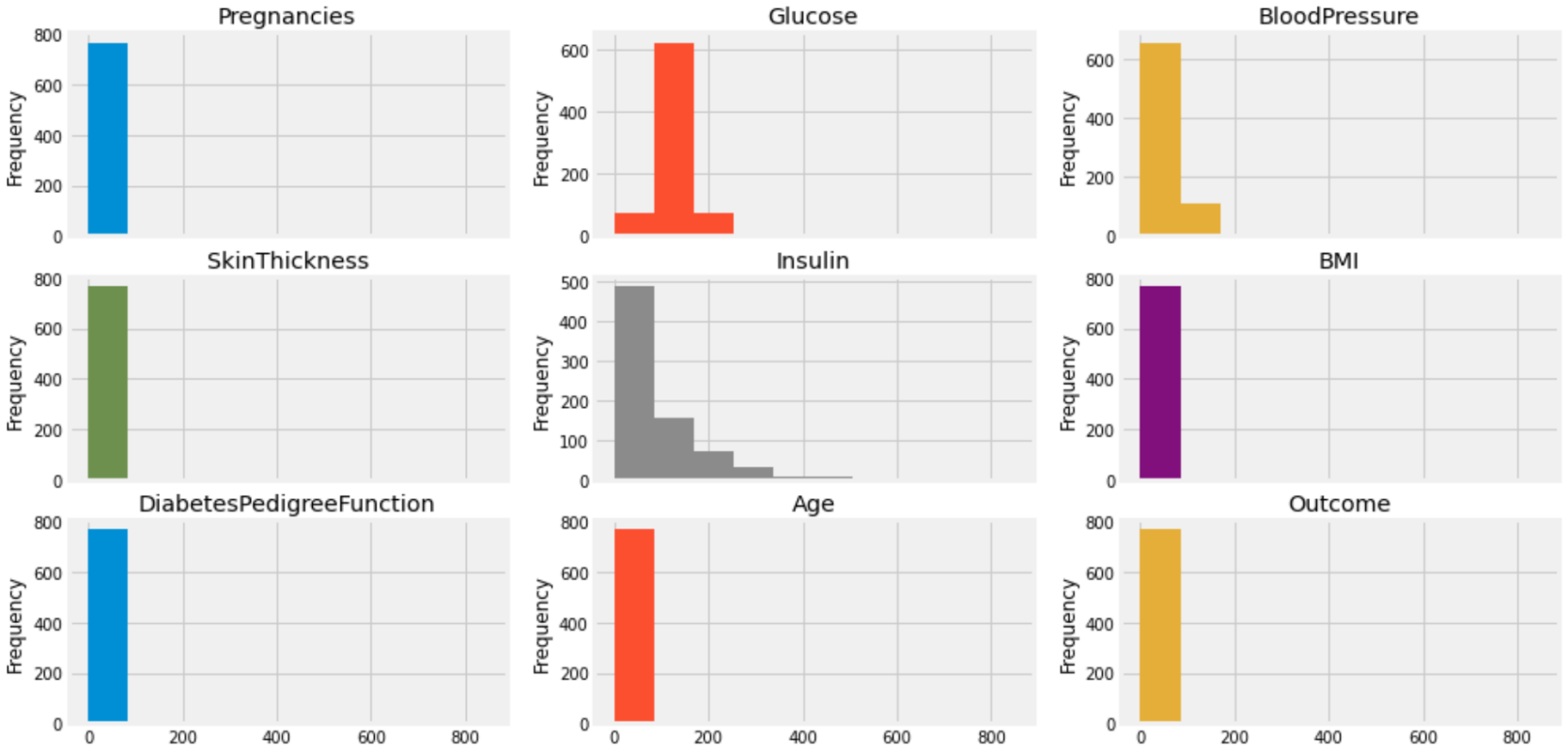


Histogram

- A graphical display of data using bars of different heights.
  - Display the shape and spread of continuous data.
  - Display the frequency of discrete data.
  - Efficient to visualize a large amount of data
- Each bar groups numbers into ranges.
  - Taller bars show that more data falls in that range.
  - visualizes buckets (bins) of quantities and shows frequencies of these buckets.

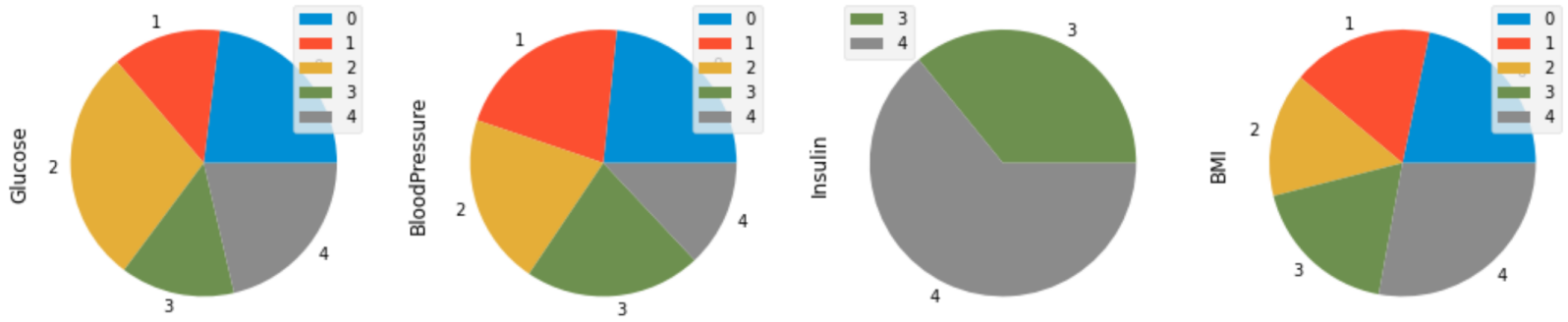
```
# Title above all subplots
data.plot(kind = 'hist',subplots=True, layout = (3,3) ,figsize=(15,8), legend=False,title = ['Pregnancies', 'Glucose','B
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A592F88>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D30AC291C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A8E03C8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A9125C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A9467C8>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A97B9C8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A9AEE88>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A9E8688>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000001D30A9E8788>]],
      dtype=object)
```



```
:
1  # Pandas Pie Chart
2  df1.plot(data.index.name, ['Glucose','BloodPressure','Insulin','BMI'],kind = 'pie', subplots=True, figsize=(15,8))
```

[45]:
```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x000001D302824B48>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x000001D3022739C8>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x000001D3019008C8>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x000001D3023401C8>],
      dtype=object)
```

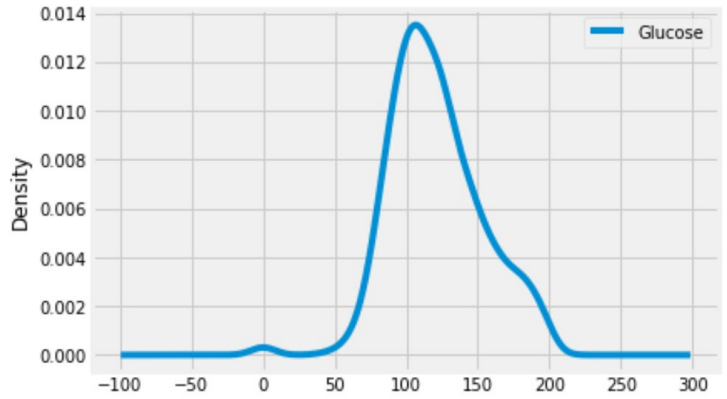


Pie Chart

- A pie chart is typically used with categorical features that have a relatively small number of values.
  - Pie charts are common in articles, but they are used less frequently in the technical publications because the size of relative areas can be hard to judge.

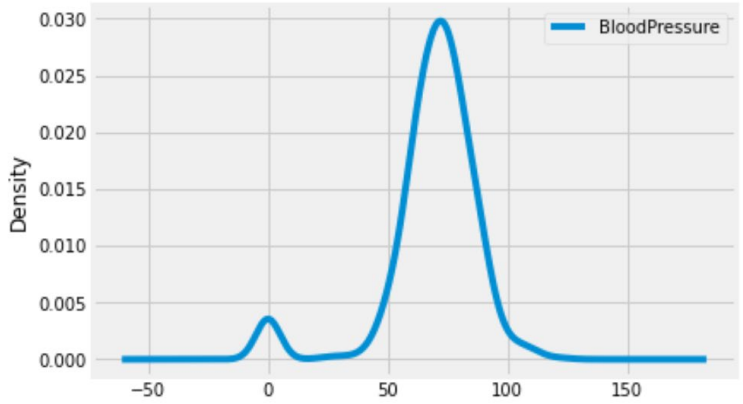
```
: 1 # Pandas density plot
  2 data.plot(data.index.name, ['Glucose'],kind = 'density')
```

```
[47]: <matplotlib.axes._subplots.AxesSubplot at 0x1d30273ba08>
```




```
  1 # Pandas density plot
  2 data.plot(data.index.name, ['BloodPressure'],kind = 'density')
```

```
3]: <matplotlib.axes._subplots.AxesSubplot at 0x1d302710d08>
```

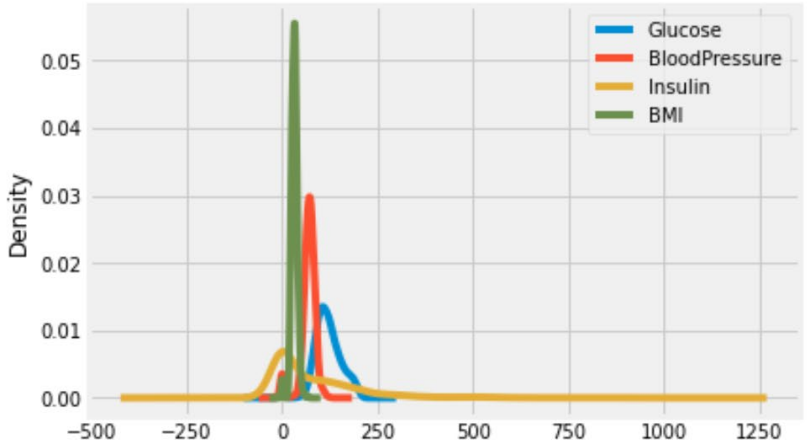


Density Plot

- A density plot is a smoothed, continuous version of a histogram estimated from the data.
- The most common form of estimation is known as kernel density estimation.
  - In this method, a continuous curve (the kernel) is drawn at every individual data point and all of these curves are then added together to make a single smooth probability density estimation.
  - The kernel most often used is a Gaussian
    - produces a Gaussian bell curve at each data point.

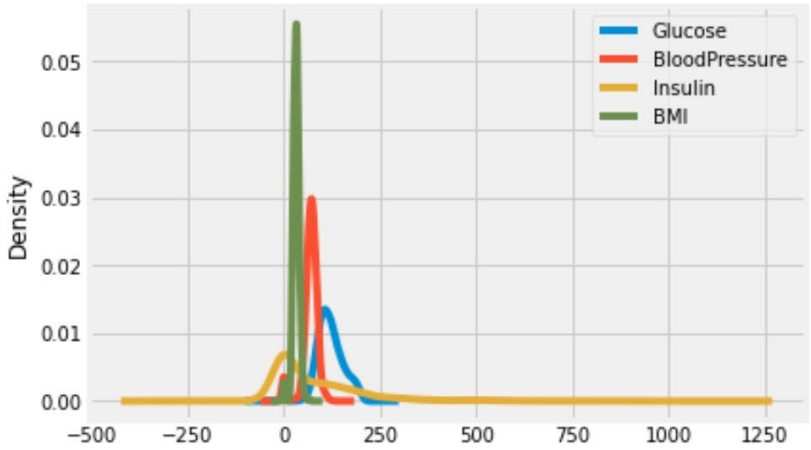
```
# Pandas density plot
data.plot(data.index.name, ['Glucose','BloodPressure','Insulin','BMI'],kind = 'density')
```

6]: <matplotlib.axes._subplots.AxesSubplot at 0x1d3026c5548>



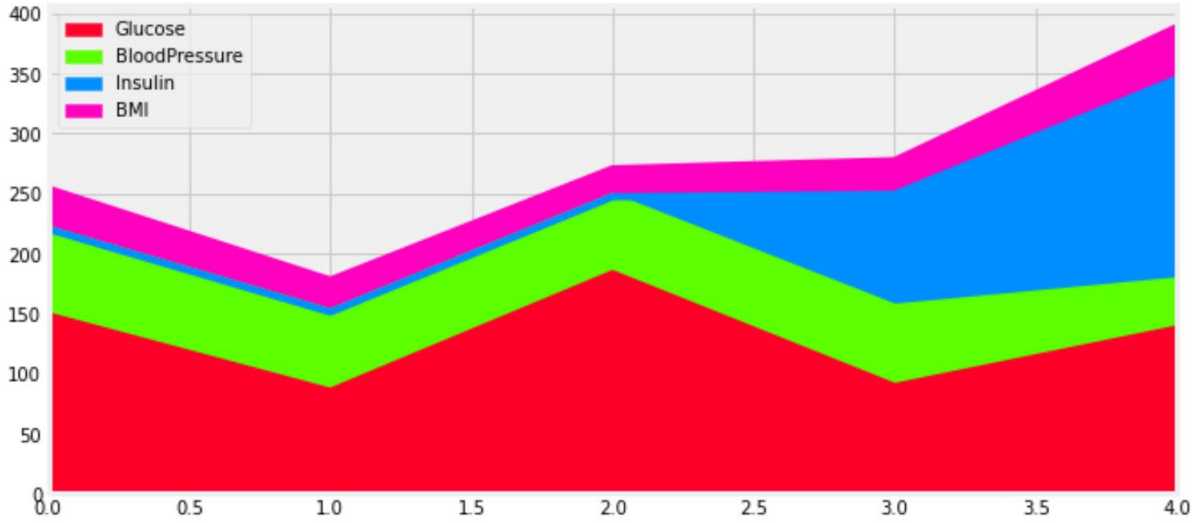

```
#  KDE plot
# Pandas density plot
data.plot(data.index.name, ['Glucose','BloodPressure','Insulin','BMI'],kind = 'kde')
```

0]: <matplotlib.axes._subplots.AxesSubplot at 0x1d302438c48>



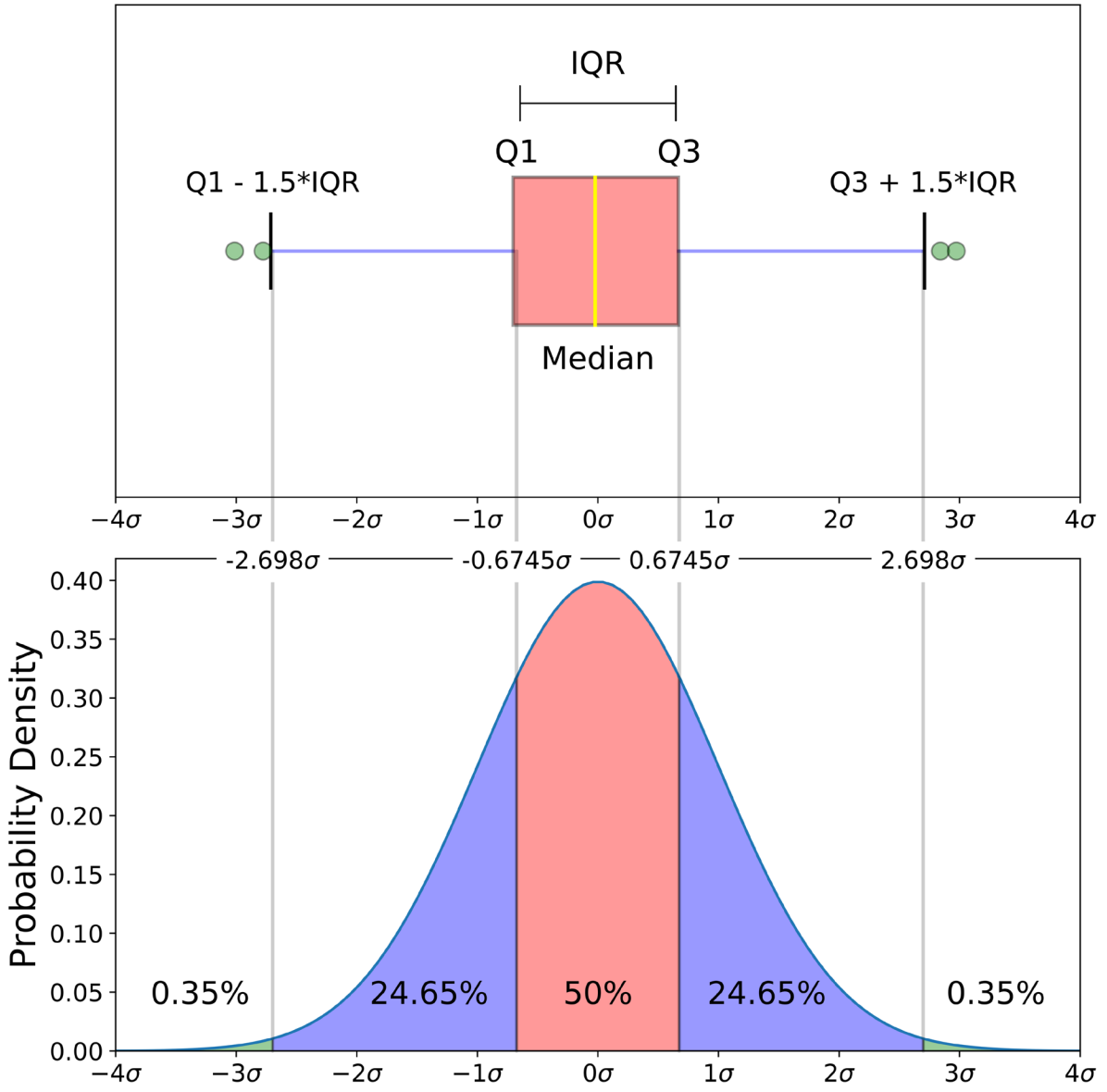
```
# Pandas colormap
df1.plot(data.index.name, ['Glucose','BloodPressure','Insulin','BMI'],kind = 'area', figsize=(10,5),
        colormap='gist_rainbow')
```

3]: <matplotlib.axes._subplots.AxesSubplot at 0x1d30b1da088>



Area Chart

- An area chart or area graph displays graphically quantitative data.
  - It is based on the line chart.
- The area between axis and line are commonly emphasized with colors and textures.
- Commonly one compares two or more quantities with an area chart.



https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

Box Plot

- Box plots are a popular way of visualizing a distribution.
- A boxplot include the five number summary:
  - The ends of the box are at the quartiles so that the box length is the interquartile range.
  - The median is marked by a line within the box.
  - Two lines (called whiskers) outside the box extend to the smallest (Minimum) and largest (Maximum) values.
  - IQR = Q3 – Q1

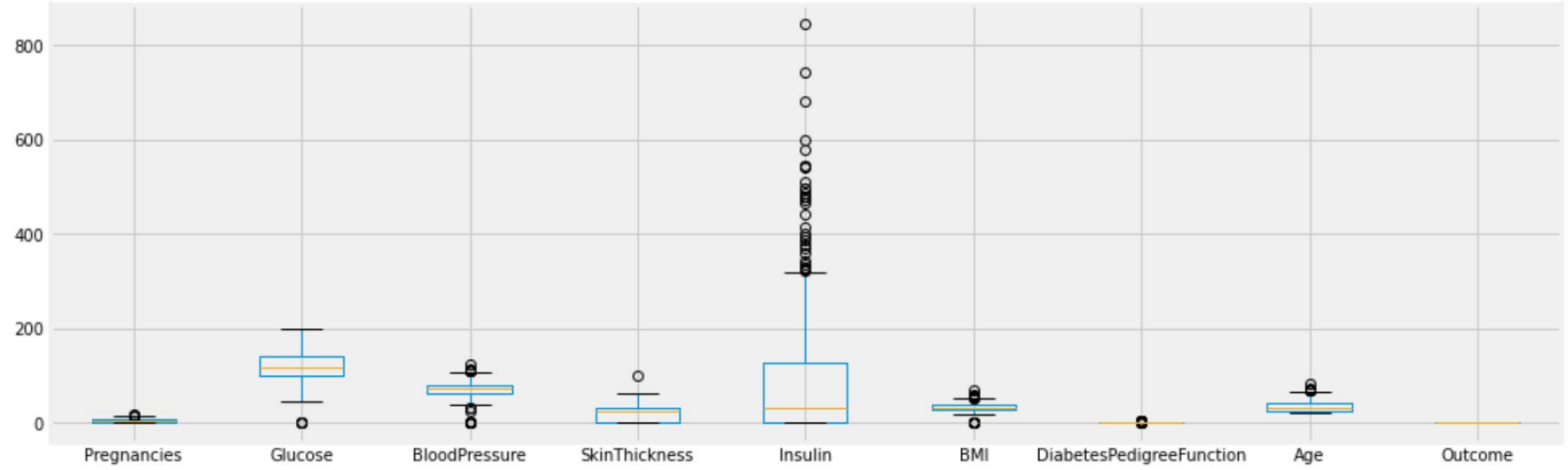
```
1 data.describe()
```

7]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

```
1 data.plot(x=data.index.name,kind='box',figsize=(15,5))
```

]: <matplotlib.axes._subplots.AxesSubplot at 0x1d3017c4588>



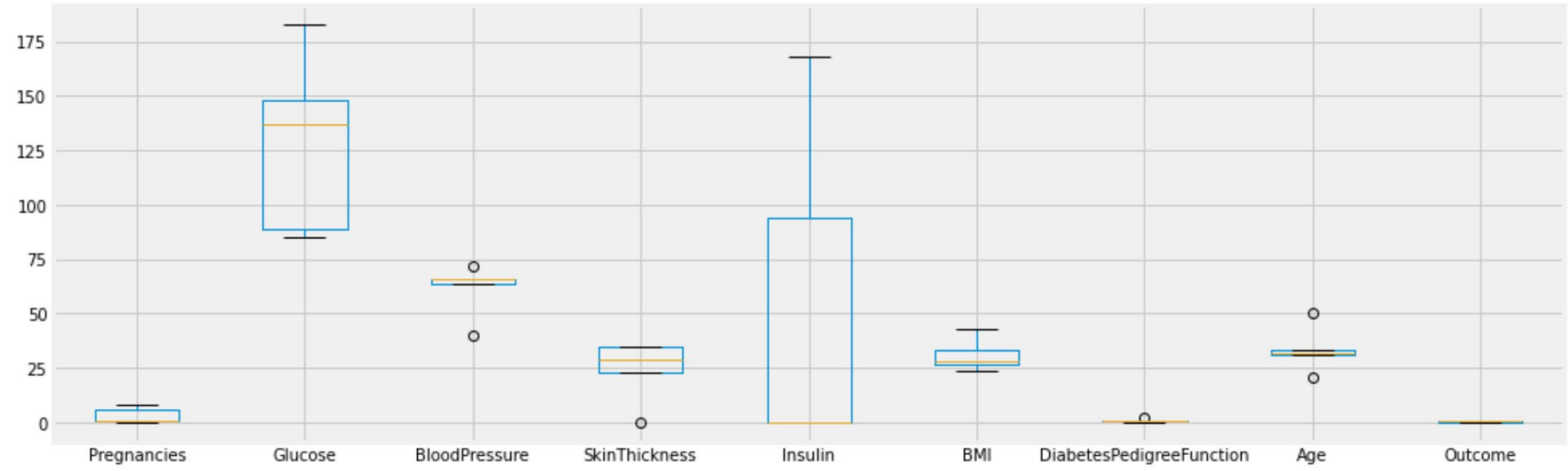
```
1 data.head()
```

8]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
1 data[:5].plot(x=data.index.name,kind='box',figsize=(15,5))
```

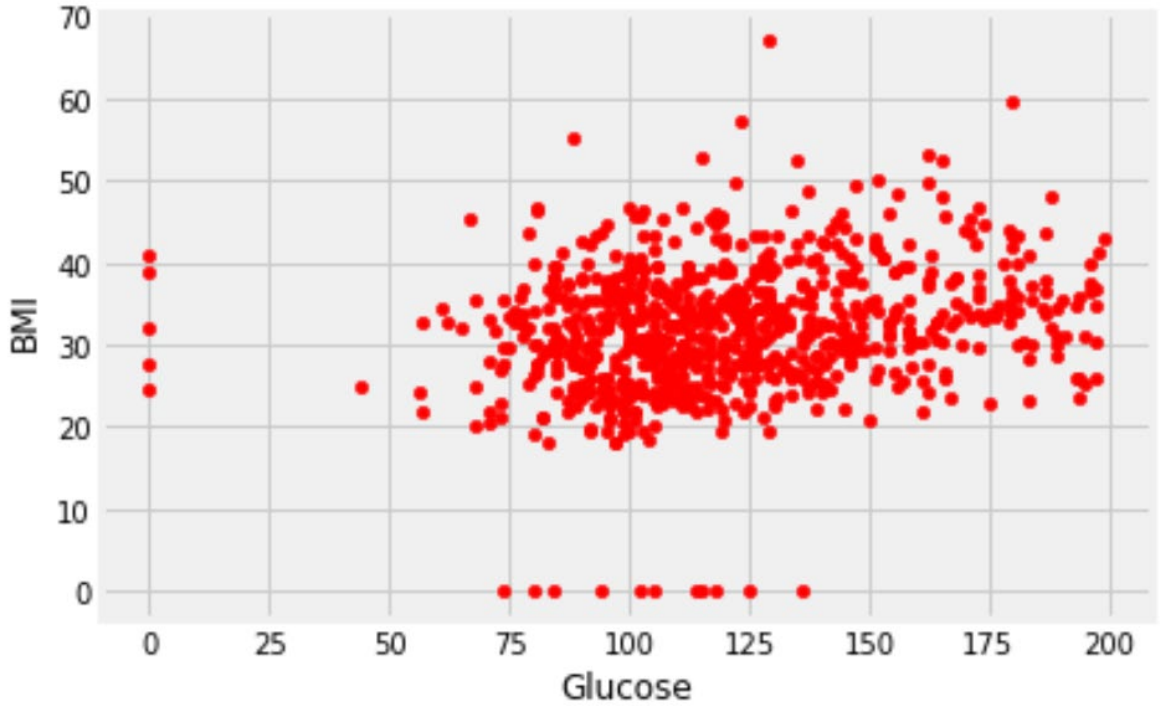
: <matplotlib.axes._subplots.AxesSubplot at 0x1d3017dd5c8>



```
1  # Pandas Scatter Plot
2  data.plot(x='Glucose',y='BMI',kind='scatter',color='R')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplo
r uppercase single-letter colors is deprecated since Matplotlib 3.
d.
  **self.kwds
```

```
: <matplotlib.axes._subplots.AxesSubplot at 0x1d306692b48>
```



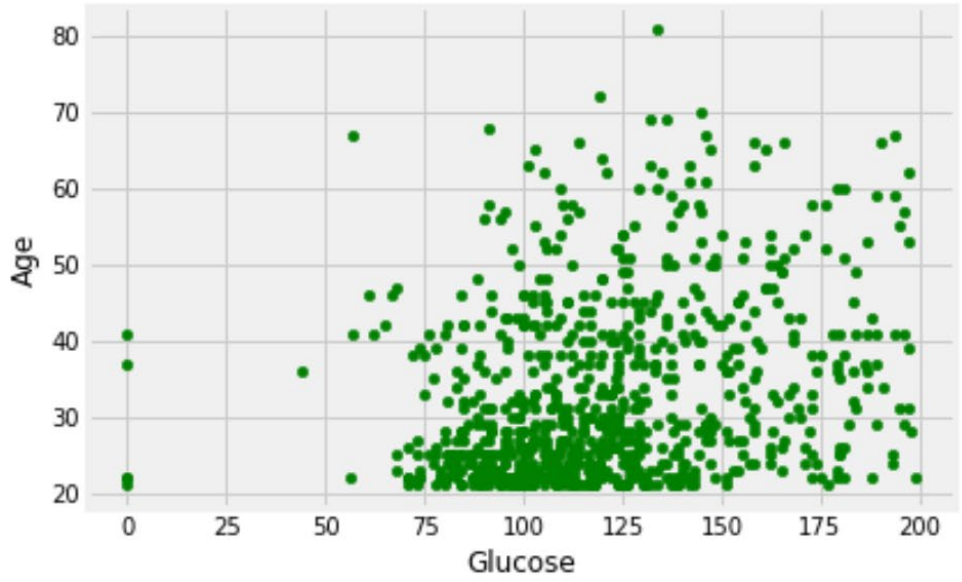
Scatter Plot

- Multivariate Plot
- An effective graphical method for determining if there is a relationship, pattern, or trend between two numeric features.
  - A scatter plot can suggest various kinds of correlations between features
    - Positive Correlation
    - Negative Correlation
    - No Correlation
- To construct a scatter plot, each pair of values is treated as a pair of coordinates in an algebraic sense and plotted as points in the plane.

In [74]:

```
1  # Pandas Scatter Plot
2  data.plot(x='Glucose',y='Age',kind='scatter',color='G')
```

Out[74]: <matplotlib.axes._subplots.AxesSubplot at 0x1d306a94948>

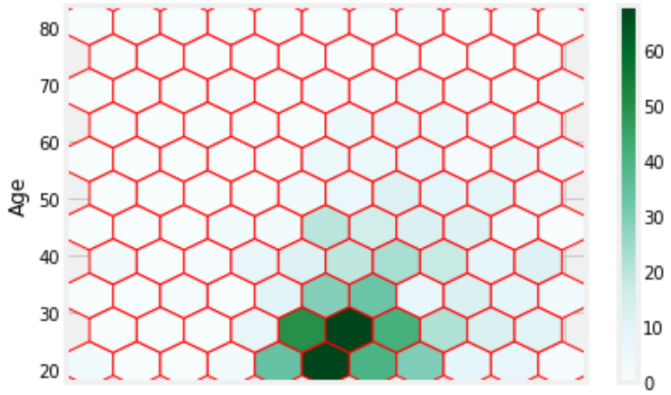


	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
egnancies	1.000	0.129	0.141	-0.082	-0.074	0.018	-0.034	0.544	0.222
Glucose	0.129	1.000	0.153	0.057	0.331	0.221	0.137	0.264	0.467
dPressure	0.141	0.153	1.000	0.207	0.089	0.282	0.041	0.240	0.065
Thickness	-0.082	0.057	0.207	1.000	0.437	0.393	0.184	-0.114	0.075
Insulin	-0.074	0.331	0.089	0.437	1.000	0.198	0.185	-0.042	0.131
BMI	0.018	0.221	0.282	0.393	0.198	1.000	0.141	0.036	0.293
eeFunction	-0.034	0.137	0.041	0.184	0.185	0.141	1.000	0.034	0.174
Age	0.544	0.264	0.240	-0.114	-0.042	0.036	0.034	1.000	0.238
Outcome	0.222	0.467	0.065	0.075	0.131	0.293	0.174	0.238	1.000

```
# hexbin plot
data.plot(x='Glucose',y='Age',kind='hexbin',color='R', gridsize =10)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\tool
The rowNum attribute was deprecated in Matplotlib 3.2 and will be removed t
().rowspan.start instead.
  layout[ax.rowNum, ax.colNum] = ax.get_visible()
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\tool
The colNum attribute was deprecated in Matplotlib 3.2 and will be removed t
().colspan.start instead.
  layout[ax.rowNum, ax.colNum] = ax.get_visible()
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\tool
The rowNum attribute was deprecated in Matplotlib 3.2 and will be removed t
().rowspan.start instead.
  if not layout[ax.rowNum + 1, ax.colNum]:
C:\ProgramData\Anaconda3\lib\site-packages\pandas\plotting\_matplotlib\tool
The colNum attribute was deprecated in Matplotlib 3.2 and will be removed t
().colspan.start instead.
  if not layout[ax.rowNum + 1, ax.colNum]:
```

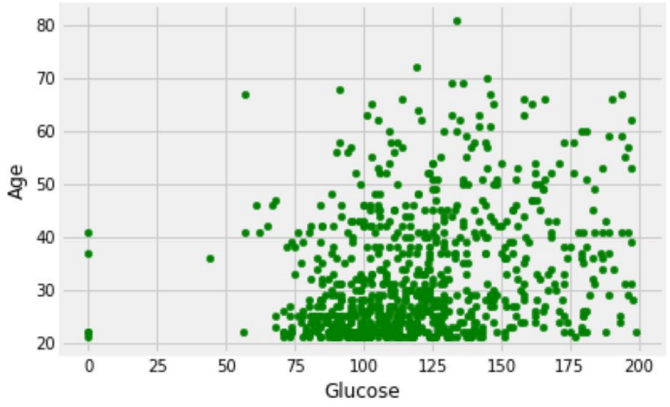
'6]: <matplotlib.axes._subplots.AxesSubplot at 0x1d306b1c608>



In [74]:

```
# Pandas Scatter Plot
data.plot(x='Glucose',y='Age',kind='scatter',color='G')
```

Out[74]: <matplotlib.axes._subplots.AxesSubplot at 0x1d306a94948>

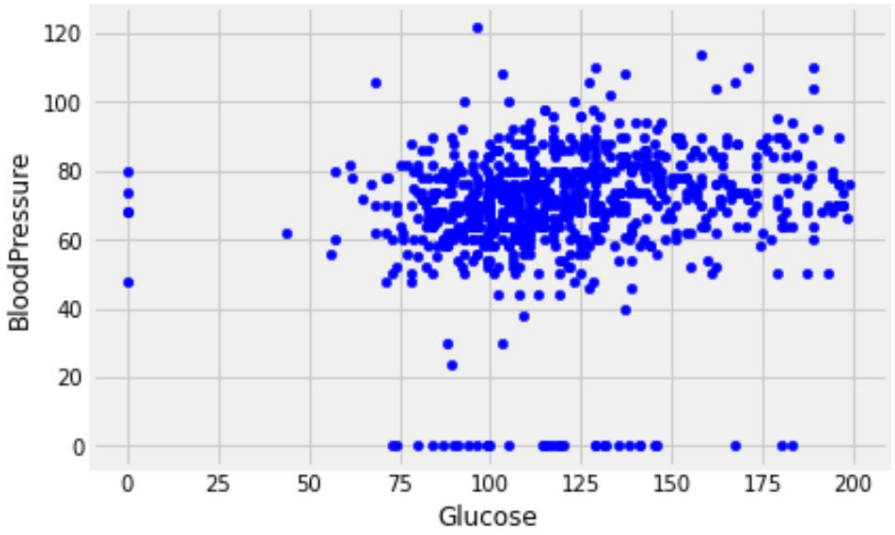


Hexbin Plot

- A Hexbin plot is useful to represent the relationship of 2 numerical variables when you have a lot of data point.
  - Instead of overlapping, the plotting window is split in several hexbins, and the number of points per hexbin is counted.
  - The color denotes this number of points.
  - you can change the size of the bins using the `gridsize` argument.

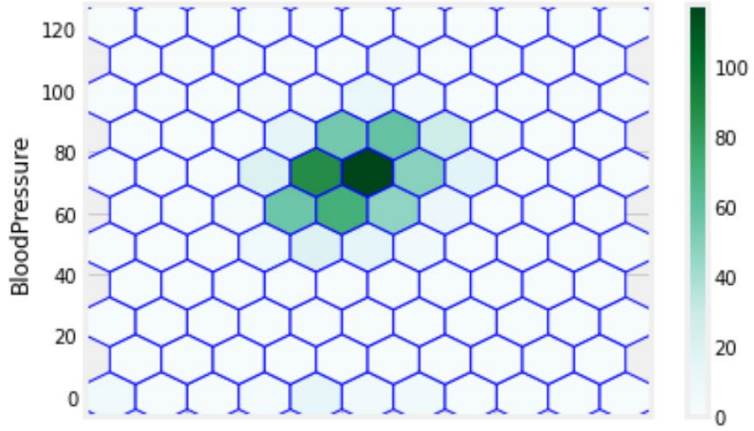
```
# Pandas Scatter Plot
data.plot(x='Glucose',y='BloodPressure',kind='scatter',color='B')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d306a18b48>
```




```
# hexbin plot
data.plot(x='BMI',y='BloodPressure',kind='hexbin',color='B', gridsize =10)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d306eb6cc8>
```



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
egnancies	1.000	0.129	0.141	-0.082	-0.074	0.018	-0.034	0.544	0.222
Glucose	0.129	1.000	0.153	0.057	0.331	0.221	0.137	0.264	0.467
dPressure	0.141	0.153	1.000	0.207	0.089	0.282	0.041	0.240	0.065
Thickness	-0.082	0.057	0.207	1.000	0.437	0.393	0.184	-0.114	0.075
Insulin	-0.074	0.331	0.089	0.437	1.000	0.198	0.185	-0.042	0.131
BMI	0.018	0.221	0.282	0.393	0.198	1.000	0.141	0.036	0.293
eeFunction	-0.034	0.137	0.041	0.184	0.185	0.141	1.000	0.034	0.174
Age	0.544	0.264	0.240	-0.114	-0.042	0.036	0.034	1.000	0.238
Outcome	0.222	0.467	0.065	0.075	0.131	0.293	0.174	0.238	1.000

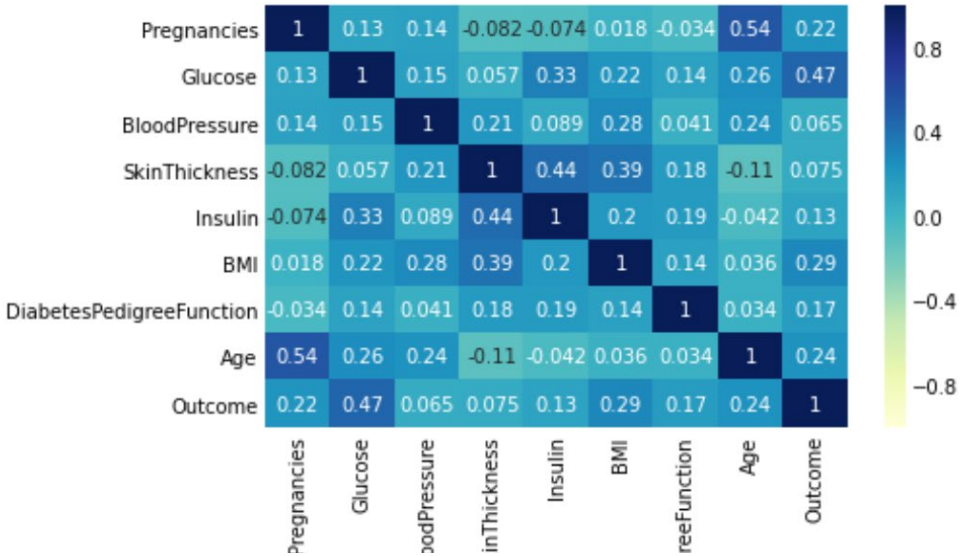
```
# pairwise Pearson correlations

correlations = data.corr(method='pearson')
correlations
```

46]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.221898
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.466581
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.065068
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.074752
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.130548
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.292695
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.173844
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.238356
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.000000

```
sns.heatmap(data.corr(), vmin=-1, vmax=1, cmap="YlGnBu", annot = True)
```

]: <matplotlib.axes._subplots.AxesSubplot at 0x1acaa3e9a48>

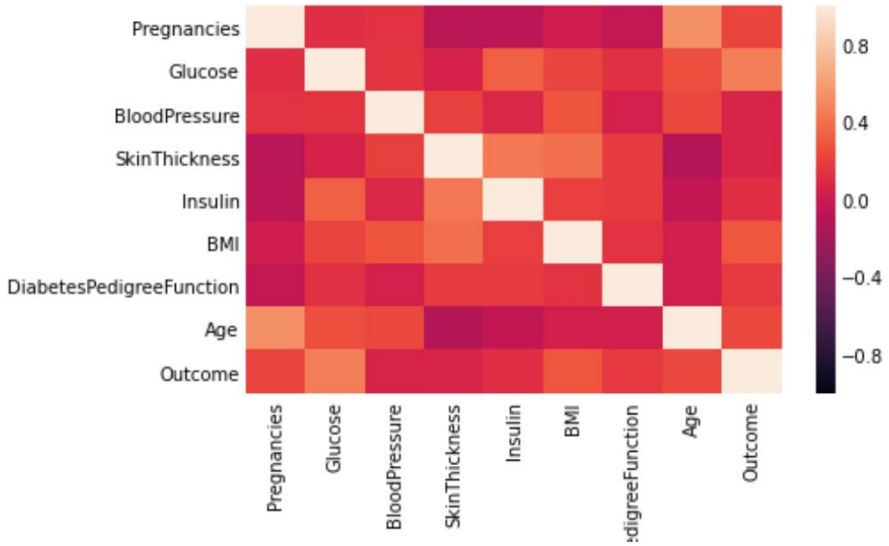


Correlation Matrix Plot (Heatmap)

- Calculate the correlation between each pair of features
  - Correlation gives an indication of how related the changes are between two variables.
    - Positive correlation
    - Negative correlation
    - None
- Plot the correlation matrix and get an idea of which variables have a high correlation with each other
  - This is useful to know, because some machine learning algorithms can have poor performance if there are highly correlated input variables in data

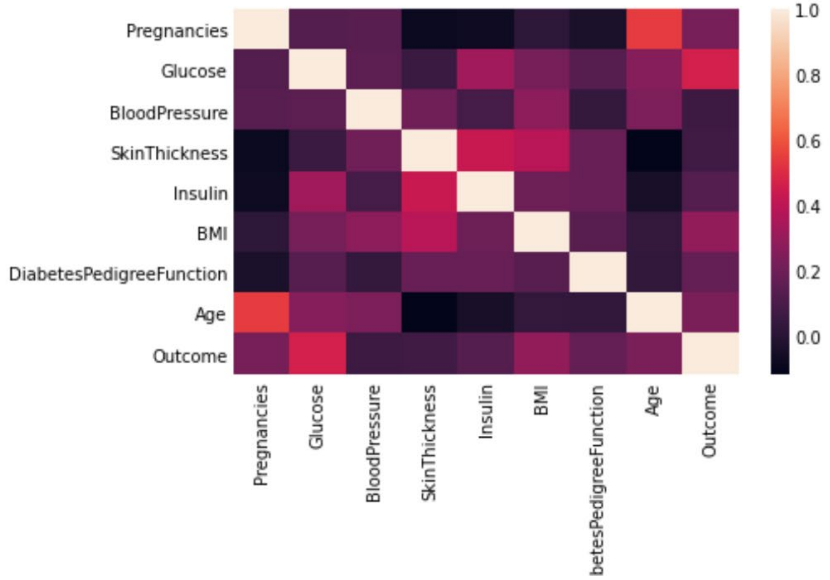
```
1 sns.heatmap(data.corr(), vmin=-1, vmax=1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1aca87e7548>
```




```
1 import seaborn as sns # another popular data visualization tool
2 sns.heatmap(data.corr())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1aca8f0df08>
```

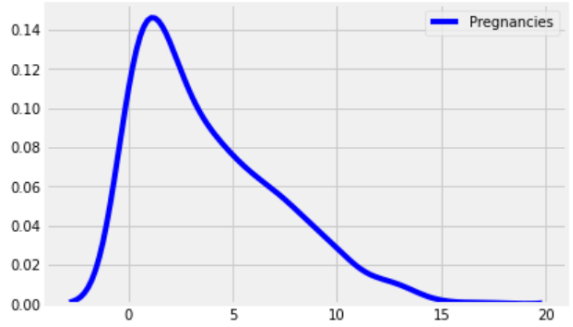


KDE Plot

- KDE (Kernel Density Estimate) Plot is used for visualizing the Probability Density of a continuous variable.
- It depicts the probability density at different values in a continuous variable.

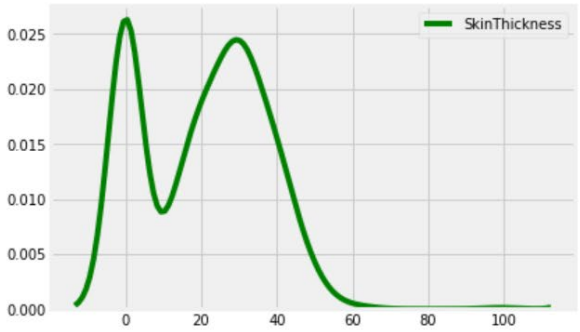
```
sns.kdeplot(data['Pregnancies'], color='b')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18becc82f08>
```



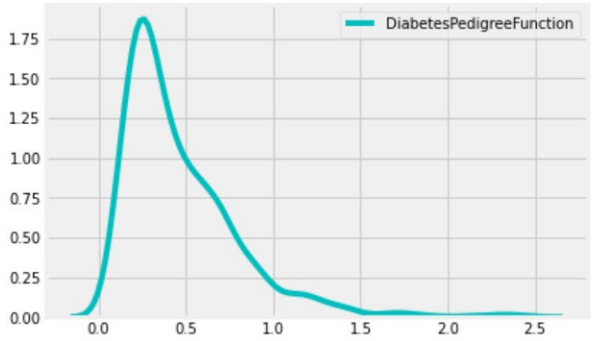

```
sns.kdeplot(data['SkinThickness'], color="g")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18bedd12ac8>
```



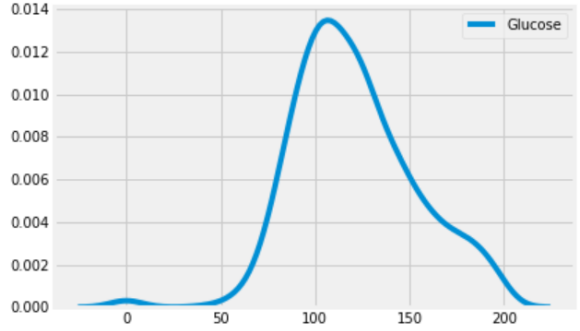

```
sns.kdeplot(data['DiabetesPedigreeFunction'], color="c")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18bedeebec8>
```



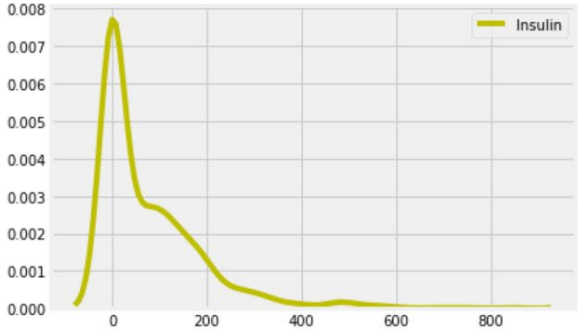

```
sns.kdeplot(data['Glucose'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18bec872f48>
```



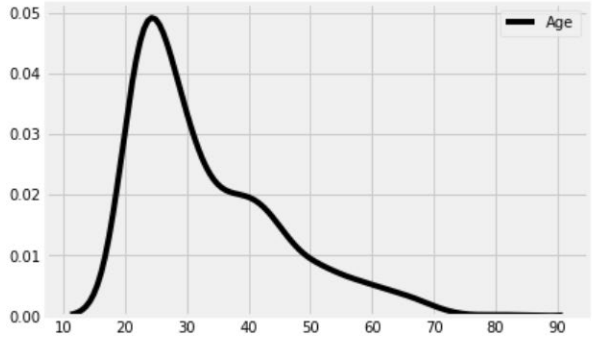

```
sns.kdeplot(data['Insulin'], color="y")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18bedd97f88>
```



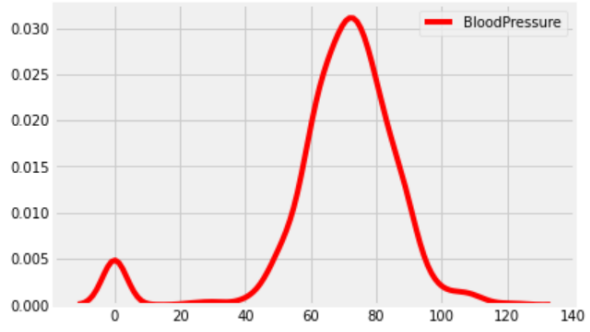

```
sns.kdeplot(data['Age'], color="k")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18bee21b688>
```




```
sns.kdeplot(data['BloodPressure'], color="r")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18bedcc5488>
```




```
sns.kdeplot(data['BMI'], color="m")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x18bec59b888>
```