Lecture-1 (Exploratory Data Analysis - EDA)

July 26, 2021

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sb
        import matplotlib.pyplot as plt
        from statsmodels import robust as rb
        from scipy import stats as st

In [2]: iris = pd.read_csv("iris.csv")

In [3]: #get data-points and feature

        print(iris.shape)
```

```
(150, 5)
```

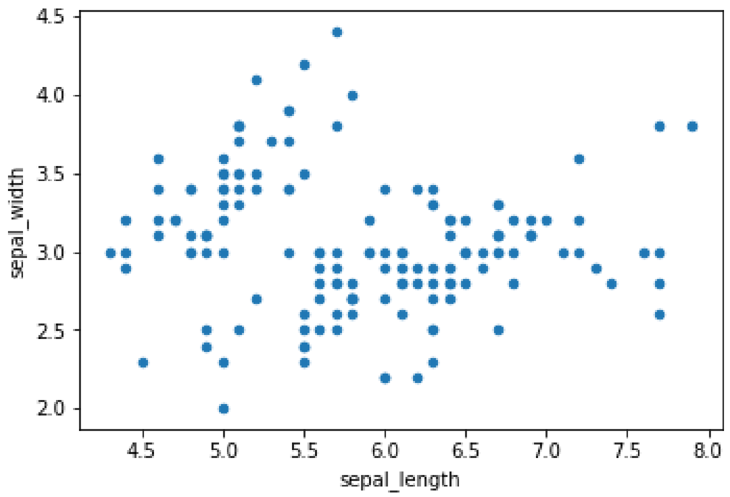
```
In [4]: #get the column name

        print(iris.columns)
```

```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

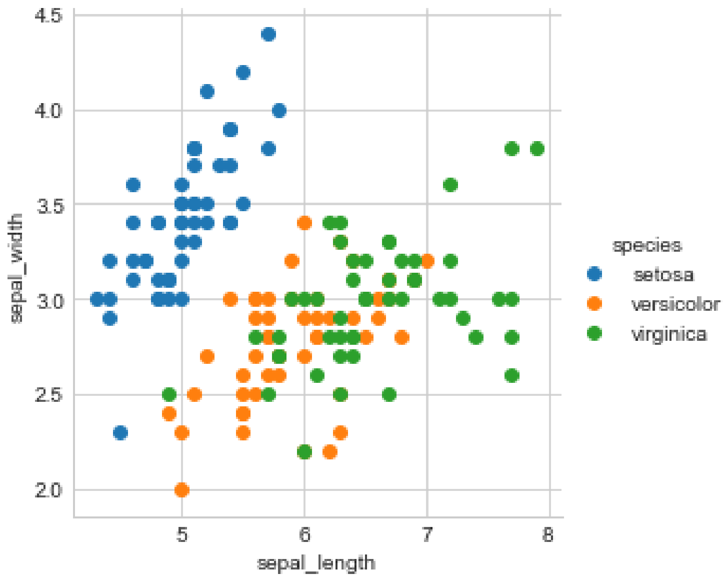
```
In [5]: #get the number of specis with respect to different class

        '''Iris data set is a balance data set. See, balance Vs imbalanace dataset'''

        iris["species"].value_counts()

Out[5]: virginica     50
        versicolor    50
        setosa        50
        Name: species, dtype: int64
```

0.0.1 2-D scatter plot:

```
In [6]: iris.plot(kind="scatter", x="sepal_length", y="sepal_width")
        plt.show()
```



```
In [7]: sb.set_style("whitegrid")
        sb.FacetGrid(iris, hue="species", size=4).map(plt.scatter, "sepal_length",
        "sepal_width").add_legend();

        plt.show()
```

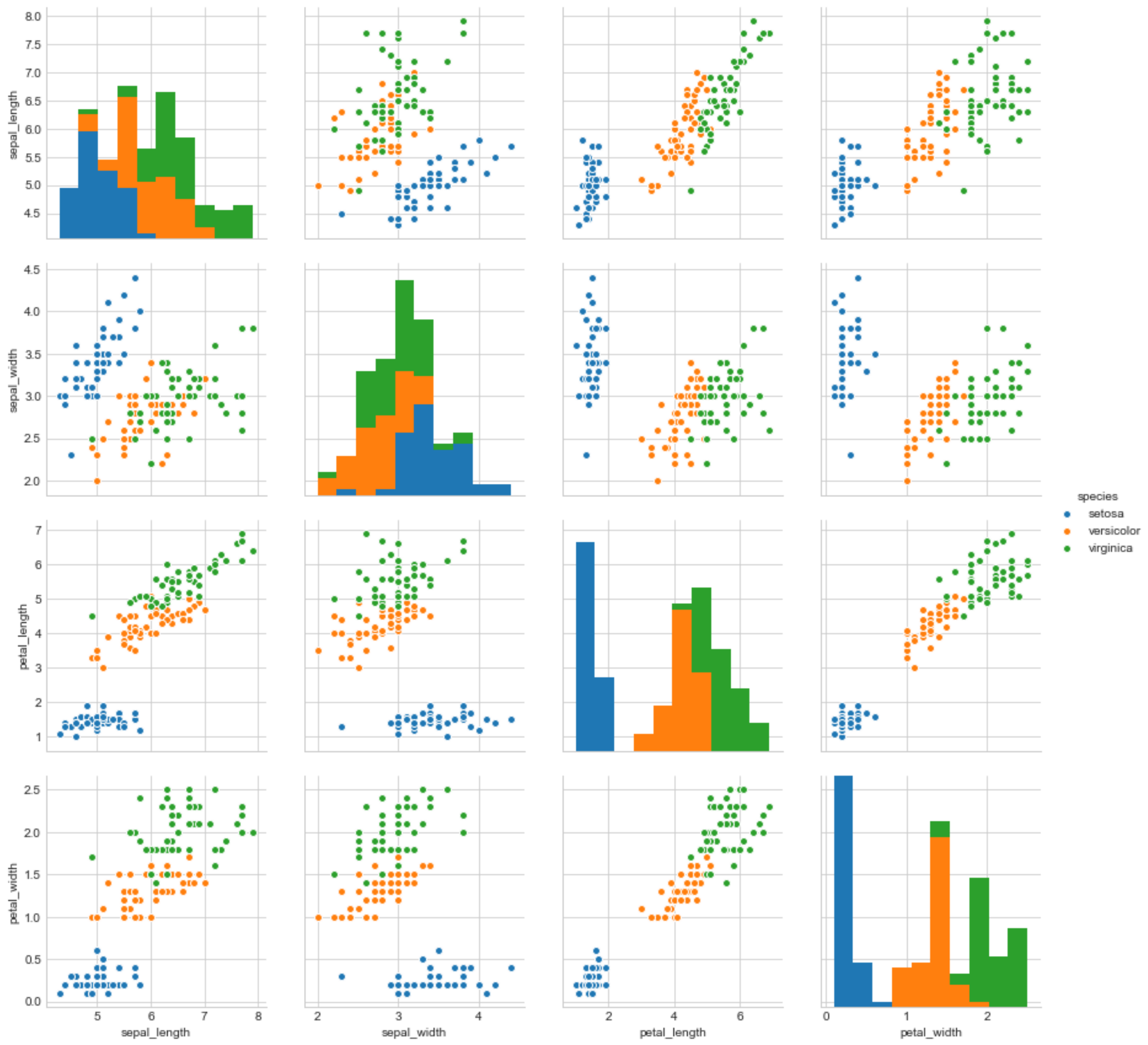


Observation(s):

1. Using sepal_length and sepal_width features, we can distinguish Setosa flowers from others.
2. Seperating Versicolor from Viginica is much harder as they have considerable overlap.

0.0.2 Pair Plots:

```
In [8]: plt.close();

        sb.set_style("whitegrid")
        sb.pairplot(iris, hue="species", size=3);

        plt.show();
```

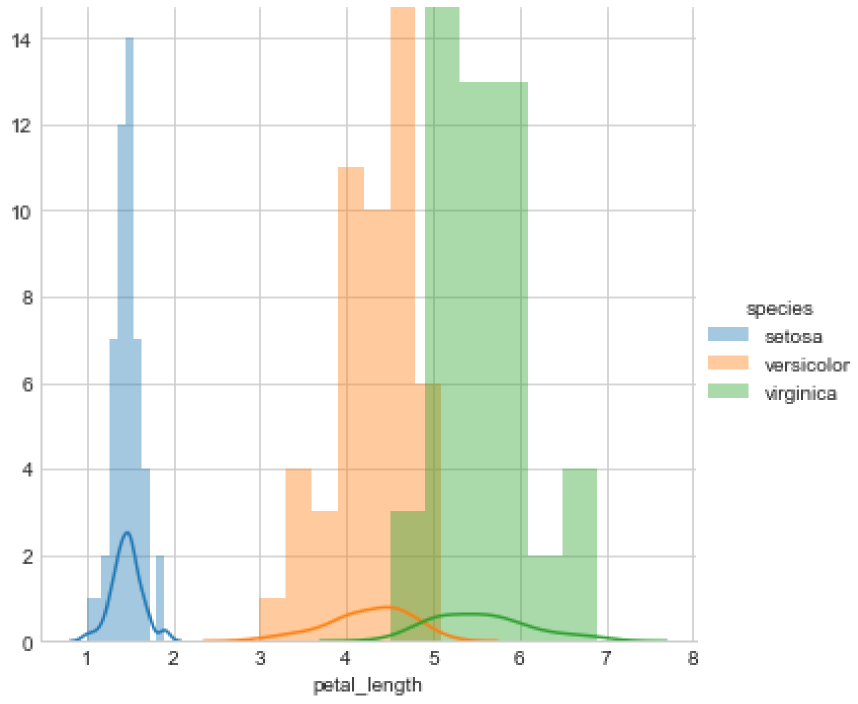


Observations:

1. petal_length and petal_width are the most useful features to identify various flower types.
2. While Setosa can be easily identified (linearly seperable), Virnica and Versicolor have some overlap (almost linearly seperable).
3. We can find "lines" and "if-else" conditions to build a simple model to classify the flower types.

0.0.3 Histogram:

```
In [9]: sb.FacetGrid(iris, hue="species", size=5) \
           .map(sb.distplot, "petal_length") \
           .add_legend();
        plt.show();
```

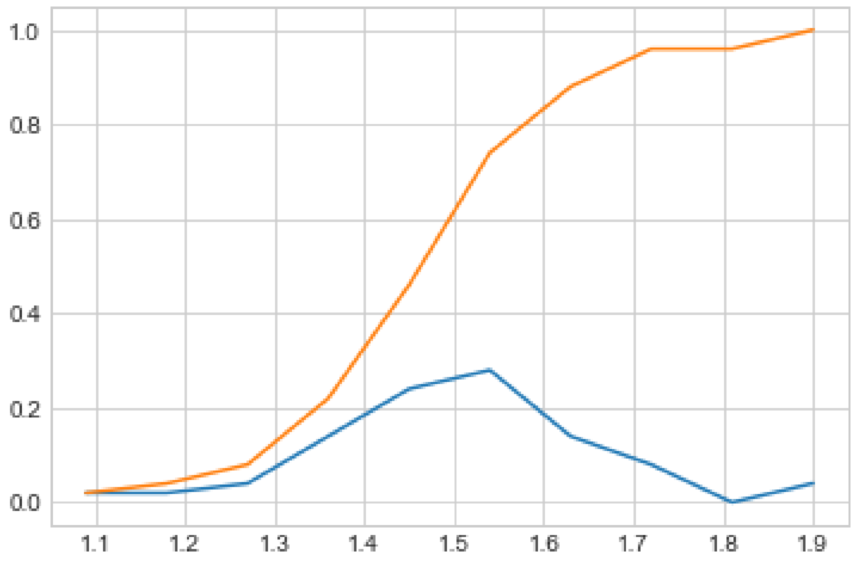


0.0.4 PDF and CDF:

```
In [10]: iris_setosa = iris.loc[iris["species"] == "setosa"];

         counts, bin_edge = np.histogram(iris_setosa['petal_length'], bins=10, density=True)

         pdf = counts / sum(counts)

         np.set_printoptions(formatter={'float': '{: 0.3f}'.format})
         print('COUNT= ', counts)
         np.set_printoptions(None)

         print('EDGE=  ', bin_edge)
         print('\nPDF=    ', pdf)

         #calculate CDF

         cdf = np.cumsum(pdf)

         print('\nCDF=    ', cdf)
```

```
        plt.plot(bin_edge[1:], pdf)
        plt.plot(bin_edge[1:], cdf)

COUNT=  [ 0.222  0.222  0.444  1.556  2.667  3.111  1.556  0.889  0.000  0.444]
EDGE=   [1.   1.09 1.18 1.27 1.36 1.45 1.54 1.63 1.72 1.81 1.9 ]

PDF=    [0.02 0.02 0.04 0.14 0.24 0.28 0.14 0.08 0.   0.04]

CDF=    [0.02 0.04 0.08 0.22 0.46 0.74 0.88 0.96 0.96 1.  ]
```

```
Out[10]: [<matplotlib.lines.Line2D at 0x1eb9241b278>]
```



0.0.5 Mean, Median, Mode:

```
In [11]: #For iris-setosa

        print('Mean= ', np.mean(iris_setosa['petal_length']))
        print('Median= ', np.median(iris_setosa['petal_length']))
        print('Mode= ', st.mode(iris_setosa['petal_length']))

Mean=  1.464
Median=  1.5
Mode=  ModeResult(mode=array([1.5]), count=array([14]))
```

0.0.6 Standard Deviation:

```
In [12]: print('SD= ', np.std(iris_setosa['petal_length']))
```

```
SD=  0.17176728442867115
```

0.0.7 Percentiles and Quintiles:

A percentile (or a centile) is a measure used in statistics indicating the value below which a given percentage of observations in a group of observations fall. For example, the 20th percentile is the value (or score) below which 20% of the observations may be found.

Quartiles are specific types of percentile. The 25th percentile is also known as the first quartile (Q1), the 50th percentile as the median or second quartile (Q2) , and the 75th percentile as the third quartile (Q3).

```
In [13]: print('20 % of Setosa has petal length of ' +
               str(np.percentile(iris_setosa['petal_length'], 20)) + ' or lower')

         print('90 % of Setosa has petal length of ' +
               str(np.percentile(iris_setosa['petal_length'], 90)) + ' or lower')
```

```
20 % of Setosa has petal length of 1.3 or lower
90 % of Setosa has petal length of 1.7 or lower
```

```
In [14]: q1, q2, q3 = np.percentile(iris_setosa['petal_length'], [25, 50, 75])

         print('1st Quentile (Q1) = ', q1)
         print('2nd Quentile (Q2) = ', q2)
         print('3rd Quentile (Q3) = ', q3)
```

```
1st Quentile (Q1) =  1.4
2nd Quentile (Q2) =  1.5
3rd Quentile (Q3) =  1.5750000000000002
```

0.0.8 Median Absolute Deviation (MAD):

The median absolute deviation(MAD) is a robust measure of how spread out a set of data is. The variance and standard deviation are also measures of spread, but they are more affected by extremely high or extremely low values and non normality. If your data is normal, the standard deviation is usually the best choice for assessing spread. However, if your data isn't normal, the MAD is one statistic you can use instead.

if X1, X2, X3,...., Xn dataset MAD is define as: MAD = Median(|Xm - Xi|) [Where, Xm = Median(Xi)and i=1,2,3,...,n]

For example, Consider the data (1, 1, 2, 2, 4, 6, 9). It has a median value of 2. The absolute deviations about 2 are (1, 1, 0, 0, 2, 4, 7) which in turn have a median value of 1 (because the sorted absolute deviations are (0, 0, 1, 1, 2, 4, 7)). So the median absolute deviation for this data is 1.

```
In [15]: print('Using library= ', rb.mad(iris_setosa['petal_length']))

         print('Using Calculation= ', np.median(np.absolute(np.median
                  (iris_setosa['petal_length'])-iris_setosa['petal_length'])))

Using library=  0.14826022185056031
Using Calculation=  0.10000000000000009
```

rb.mad() function is not working properly as it calculate the mad with Gaussian districbution.

0.0.9 Interquartile Range (IQR):

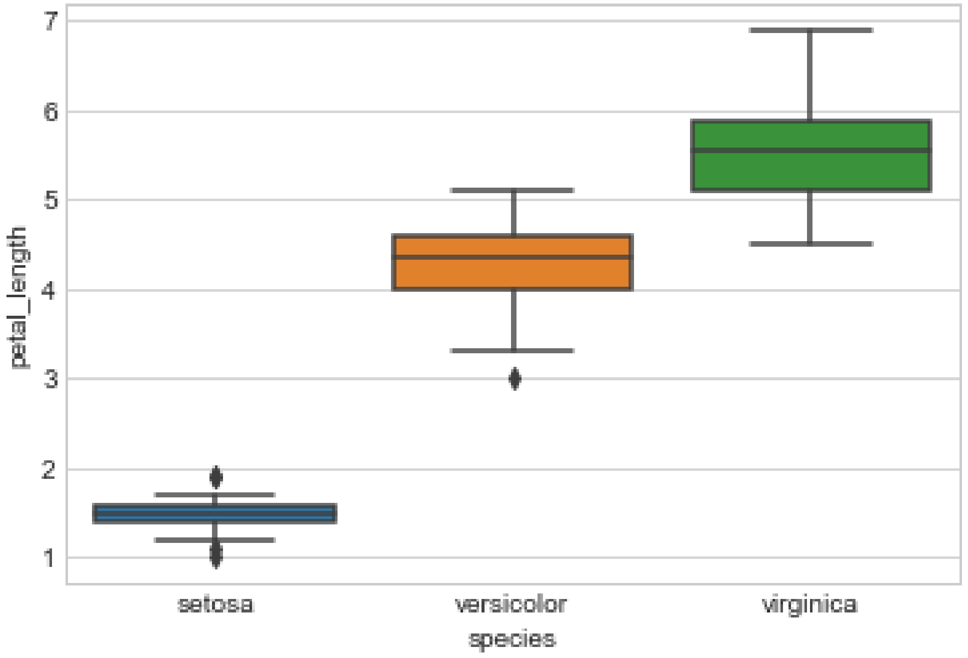
The interquartile range (IQR), also called the midspread, middle 50%, or H-spread, is a measure of statistical dispersion, being equal to the difference between 75th and 25th percentiles, or between upper and lower quartiles. So,IQR = Q3 − Q1.

```
In [16]: print('IQR of setosa petal length= ',st.iqr(iris_setosa['petal_length']))

IQR of setosa petal length=  0.17500000000000027
```

0.0.10 Box Plot:

A box plot is a type of chart often used in explanatory data analysis. Box plots visually show the distribution of numerical data and skewness through displaying the data quartiles (or percentiles) and averages.

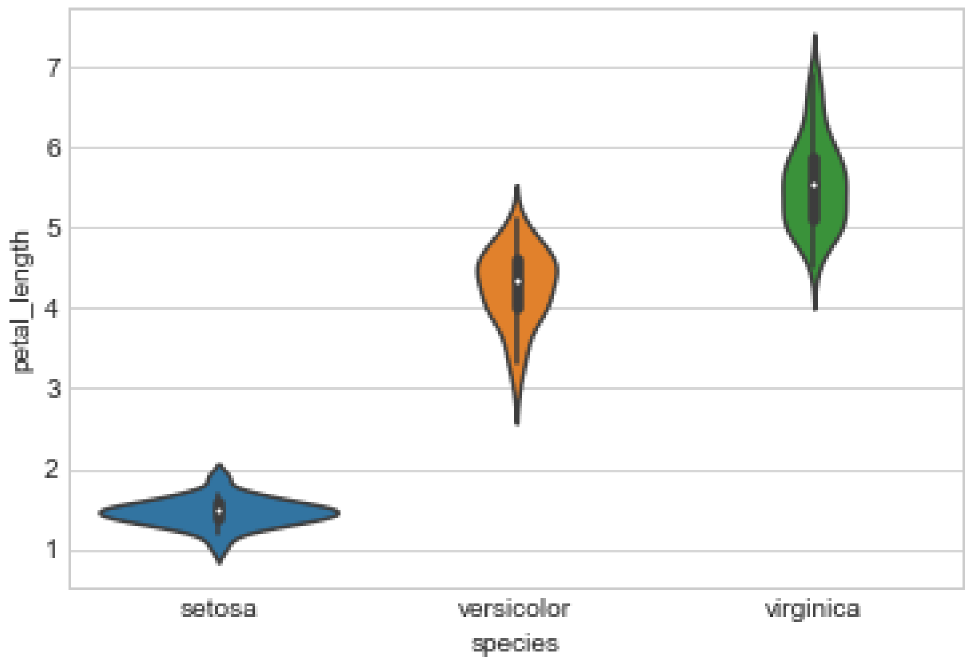
```
In [17]: sb.boxplot(x='species', y='petal_length', data=iris)

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x1eb91fd1dd8>
```



0.0.11 Violin Plot:

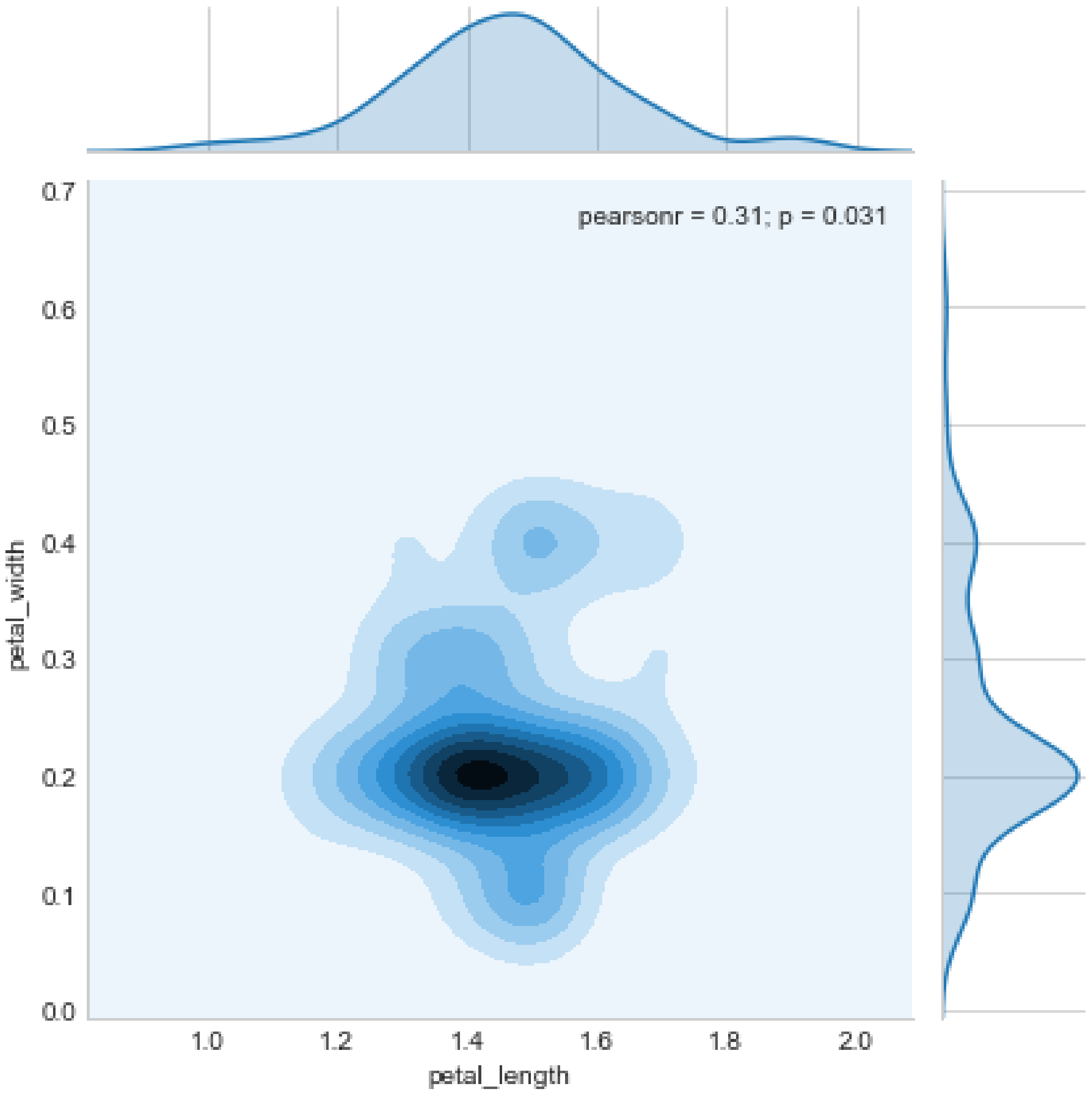
A violin plot is a method of plotting numeric data. It is similar to a box plot, with the addition of a rotated kernel density plot on each side. Violin plots are similar to box plots, except that they also show the probability density of the data at different values, usually smoothed by a kernel density estimator.

```
In [18]: sb.violinplot(x="species", y="petal_length", data=iris, size=8)
         plt.show()
```



0.0.12 Multivariate probability density, contour plot:

```
In [19]: sb.jointplot(x="petal_length", y="petal_width", data=iris_setosa, kind="kde");
         plt.show();
```


pearsonr = 0.31; p = 0.031
petal_width
petal_length
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
1.0
1.2
1.4
1.6
1.8
2.0