Statistical Thinking in Python (Part 1)

November-09-17

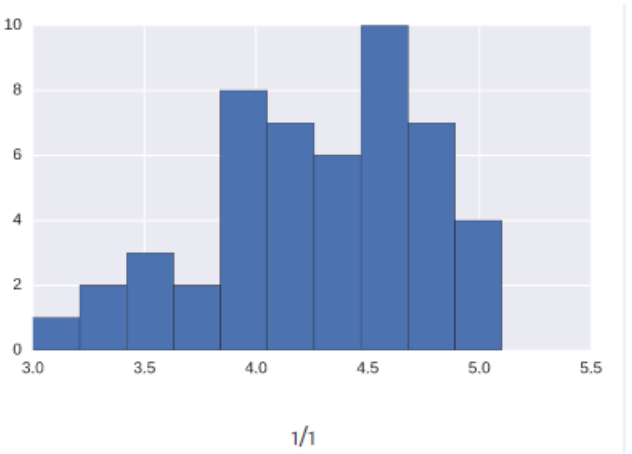
#Ch 1 Graphical exploratory data analysis

#Plotting a histogram of iris data
# Import plotting modules
import matplotlib.pyplot as plt
import seaborn as sns

Set default Seaborn style
sns.set ()

# Plot histogram of versicolor petal lengths
_ = plt.hist (versicolor_petal_length)

Show histogram
plt.show ()

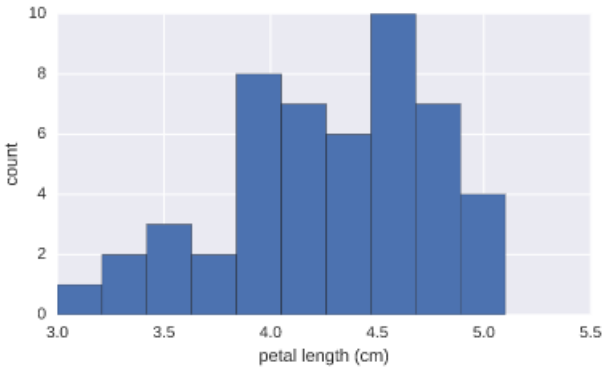


1/1

#Axis labels!
# Plot histogram of versicolor petal lengths
_ = plt.hist(versicolor_petal_length)

# Label axes
_ = plt.xlabel ('petal length (cm)')
_ = plt.ylabel ('count')

Show histogram
plt.show ()



1/1

#Adjusting the number of bins in a histogram
# Import numpy
import numpy as np

# Compute number of data points: n_data
n_data = len (versicolor_petal_length)

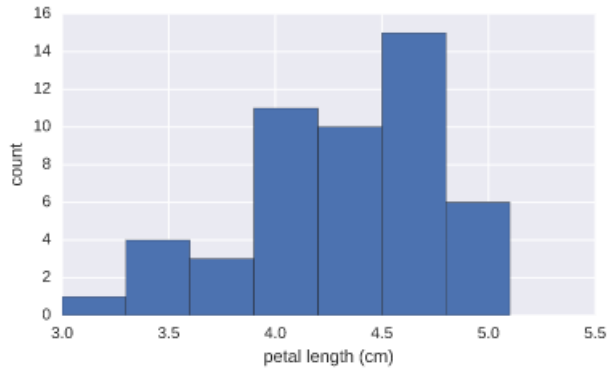
# Number of bins is the square root of number of data points: n_bins
n_bins = np.sqrt (n_data)

# Convert number of bins to integer: n_bins
n_bins = int (n_bins)

# Plot the histogram
_ = plt.hist (versicolor_petal_length, bins = n_bins)

# Label axes
_ = plt.xlabel('petal length (cm)')
_ = plt.ylabel('count')

# Show histogram
plt.show()

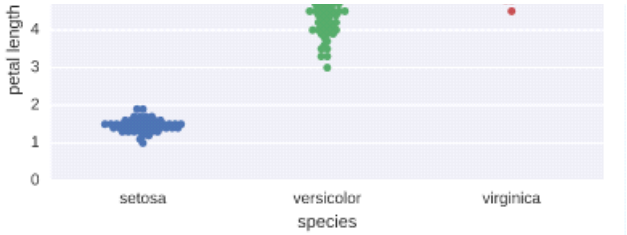


#Bee swarm plot
# Create bee swarm plot with Seaborn's default settings
_ = sns.swarmplot(x='species', y='petal length (cm)', data=df)

# Label the axes
_ = plt.xlabel('species')
_ = plt.ylabel('petal length (cm)')

```
# Label the axes
_ = plt.xlabel('species')
_ = plt.ylabel('petal length (cm)')

# Show the plot
plt.show()
```




```
#Computing the ECDF
def ecdf(data):
    """Compute ECDF for a one-dimensional array of measurements."""

    # Number of data points: n
    n = len(data)

    # x-data for the ECDF: x
    x = np.sort(data)

    # y-data for the ECDF: y
    y = np.arange(1, n+1) / n

    return x, y
```

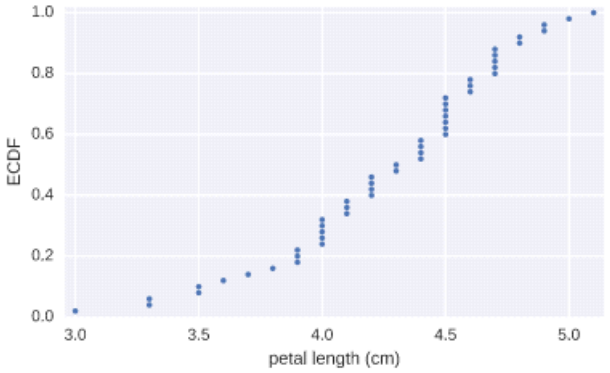
```
#Plotting the ECDF
# Compute ECDF for versicolor data: x_vers, y_vers
x_vers, y_vers = ecdf(versicolor_petal_length)

# Generate plot
_ = plt.plot (x_vers, y_vers, marker = '.', linestyle = 'none')


# Make the margins nice
plt.margins (0.02)

# Label the axes
_= plt.ylabel ('ECDF')
_= plt.xlabel ('petal length (cm)')

# Display the plot
plt.show ()
```




```
#Comparison of ECDFs
# Compute ECDFs
x_set, y_set = ecdf(setosa_petal_length)
x_vers, y_vers = ecdf(versicolor_petal_length)
x_virg, y_virg = ecdf (virginica_petal_length)

# Plot all ECDFs on the same plot
_ = plt.plot (x_set, y_set, marker = ".", linestyle = 'none')
_ = plt.plot (x_vers, y_vers, marker = ".", linestyle = 'none')
_ = plt.plot (x_virg, y_virg, marker = ".", linestyle = 'none')

# Make nice margins
plt.margins (0.02)

# Annotate the plot
plt.legend(('setosa', 'versicolor', 'virginica'), loc='lower right')
_ = plt.xlabel('petal length (cm)')
_ = plt.ylabel('ECDF')

# Display the plot
plt.show()
```