Project 2: Exploratory Data Analysis and Unsupervised Learning

Version: 0.1

Outline

In this project, you will require to perform exploratory data analyses on a well-known dataset Iris, implement an unsupervised clustering algorithm *k*-means, and compare the performance of *k*-means algorithm with agglomerative clustering algorithm (using the package sci-kit learn).

- Introduction
- Question 1: Visualizing the dataset using scatterplots
- Question 2: Visualizing the dataset using boxplots and histograms
- Question 3: Visualizing the dataset using 3D-plots (by selecting any three features)
- Question 4: Implement *k*-means clustering algorithm
- Question 5: Compare the results of both *k*-means and agglomerative clustering algorithms using the package sci-kit learn
- Question 6: Selecting K. Come up with an empirical strategy
- References

Evaluation

The total marks for this project is 50. Questions 1-3 have a total of 20 marks while questions 4-6 have a total of 30 marks. Your score depends upon the correctness of implementation and how you analyze your results.

Dataset

The iris dataset consists of 3 different types of irises' (Setosa, Versicolour, and Virginica) .
The rows being the samples and the columns or features being: Sepal Length, Sepal Width, Petal Length and Petal Width.

Questions

Question 1: Visualizing the dataset using scatterplots

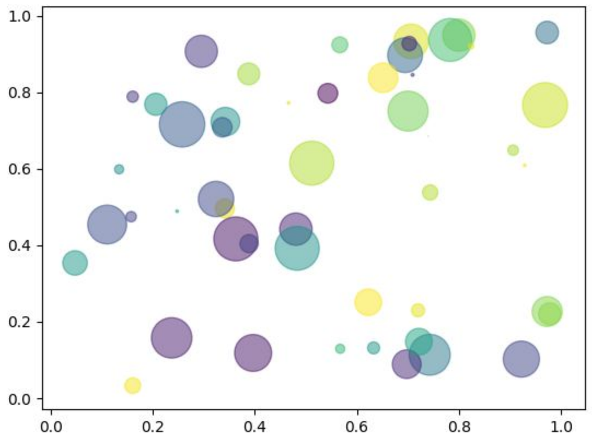
- Perform a visual exploration of the Iris dataset using scatterplots
- Use `pairplot()` for the whole dataset to look at all of our features simultaneously
- Explain what insights you can get from the plots
- What conclusions could be drawn regarding the correlations among the numerical features in our dataset.

Note: `Import sklearn, matplotlib.pyplot,numpy` modules in python

Code sample:
This example showcases a simple scatter plot.

```
import numpy as np
import matplotlib.pyplot as plt
# Fixing random state for reproducibility
np.random.seed(19680801)
N = 50
x = np.random.rand(N)
y = np.random.rand(N)
colors = np.random.rand(N)
area = (30 * np.random.rand(N))**2  # 0 to 15 point radii
plt.scatter(x, y, s=area, c=colors, alpha=0.5)
plt.show()
```

The output of the code above is displayed below.

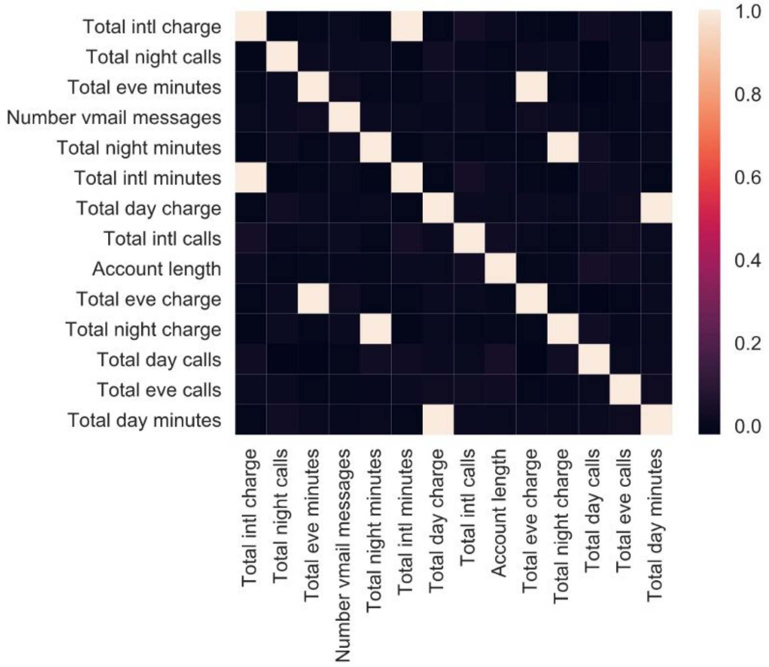


Calculate the *correlation* between each pair of features.
Use the method `corr()` on a DataFrame that calculates the correlation between each pair of features. Then, we pass the resulting *correlation matrix* to `heatmap()` from seaborn, which renders a color-coded matrix for the provided values:

Code Sample:

```
from string import ascii_letters
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

sns.set(style="white")

# Generate a large random dataset
rs = np.random.RandomState(33)
d = pd.DataFrame(data=rs.normal(size=(100, 26)),
                 columns=list(ascii_letters[26:]))

# Compute the correlation matrix
corr = d.corr()

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr)
```

Below is an example of correlation matrix



The scatter plot displays values of two numerical variables as Cartesian coordinates in 2D space. Scatter plots in 3D are also possible. Use the function `scatter()` from the `matplotlib` library:

Question 2: Visualizing the dataset using boxplots and histograms

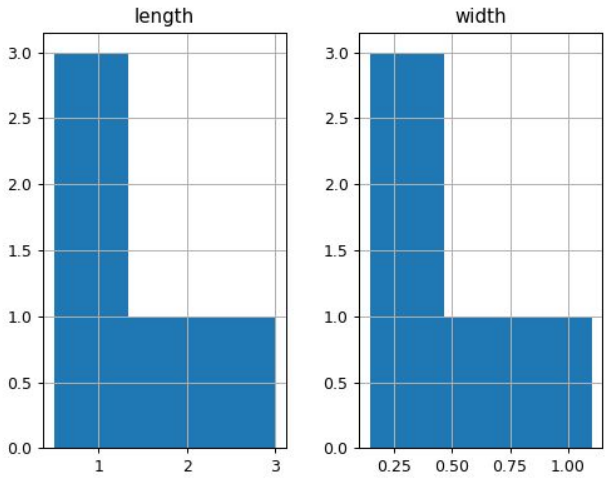
- Visualize the features of Iris images using histograms ,boxplots
- State your inferences about the iris dataset

We analyze the distribution of a numerical variable by plotting its *histogram* using the DataFrame's method `hist()`.

Code sample:

```
>>> df = pd.DataFrame({
...     'length': [1.5, 0.5, 1.2, 0.9, 3],
...     'width': [0.7, 0.2, 0.15, 0.2, 1.1]
...     }, index= ['pig', 'rabbit', 'duck', 'chicken', 'horse'])
>>> hist = df.hist(bins=3)
```

```
Output:
```

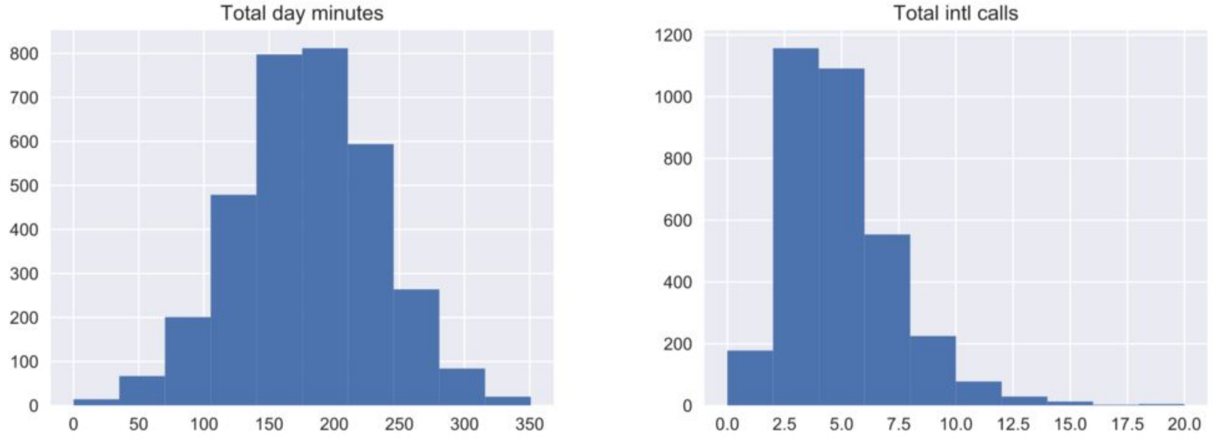


Code sample:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('../../data/filename.csv')
```

```
features = ['Total day minutes', 'Total intl calls']
df[features].hist(figsize=(12, 4) );
```

Output:



Note: There is also another, often clearer, way to grasp the distribution: *density plots* or, more formally, *Kernel Density Plots*. They can be considered a smoothed version of the histogram. Their main advantage over the latter is that they do not depend on the size of the bins

Question 3: Visualizing the dataset using 3D-plots

- Analyse the Iris dataset by plotting a 3D view using any three features
- Explain your observations

Question 4: Implement *k*-means clustering algorithm and test using the Iris dataset

- Perform change of color code for clusters at each iterations.
- Compute the sum of squared error (i.e. function J from class notes) for each iteration
- Visualize the sum of squared error and check for convergence of the k-means algorithm using line plot (error vs. iteration: iteration numbers on *x*- axis and error values on y-axis)
- Suggest different ways to choose the number of iterations to get quality clusters

Notes:

1. We use the Euclidean distance.
2. In case the algorithm never settles on a final solution, it may be a good idea to set a maximum number of iterations.

3. *k*-means is much faster if you write the update functions using operations on numpy arrays, instead of manually looping over the arrays and updating the values yourself.

Question 5: Compare the results of both ***k*-means** and **agglomerative** clustering algorithms

- Compare the performance of ***k-means*** and **agglomerative** clustering methods on the iris dataset.
    1. You can use the implementations of ***k-means*** and **agglomerative** clustering available in the python package *sci-kit learn* for this question.
    2. You can compare the two algorithms based on the voting percentage of the data points for each clusters. Analyse the inference of the results with respect to the true labels.

- Compare the two algorithms with respect to the cluster formation; for example, plot the results of the two algorithms using 3-D scatter plots, and explain.
- Study the effect of initial configuration for the two algorithms..

Question 6: Selecting k. Come up with an empirical strategy.

- How do you choose *k* for the k-mean algorithm?
- How do you choose *k* for the agglomerative clustering algorithm?

References

1. Iris dataset

2. Data Visualization

3. Exploratory data analysis with python, Kaggle