Python Code for Chapter 2 of Introduction to Data Mining

1.Data Preparation

Load necessary packages.

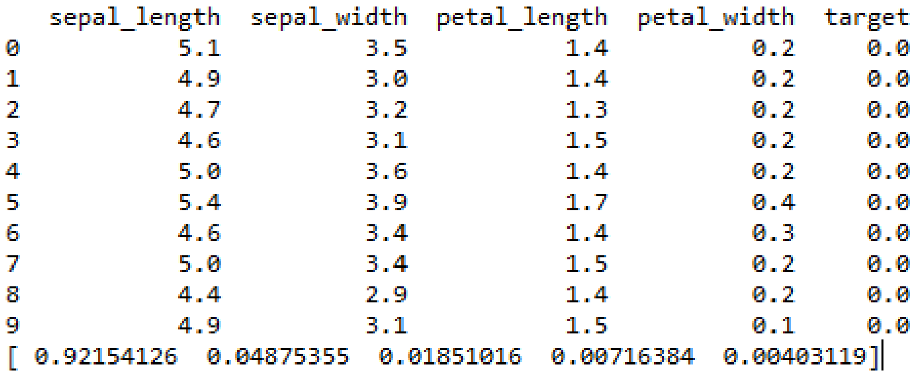
```
@Load packages
import pandas as pd
from pandas import Series,DataFrame
import numpy as np
from sklearn.datasets import load_iris
import matplotlib.pyplot as plt
import seaborn as sns
```

Load Iris dataset and store it in a dateframe.

```
data=load_iris()
data.target.shape=(len(data.data),1)
#concatenate the target column and the feature columns
new_data=np.concatenate((data.data,data.target),axis=1)
iris=pd.DataFrame(new_data,columns=
['sepal_length','sepal_width','petal_length','petal_width','target'])
```

Let us take a look at the iris dataframe.

```
print(iris.head(10))
```



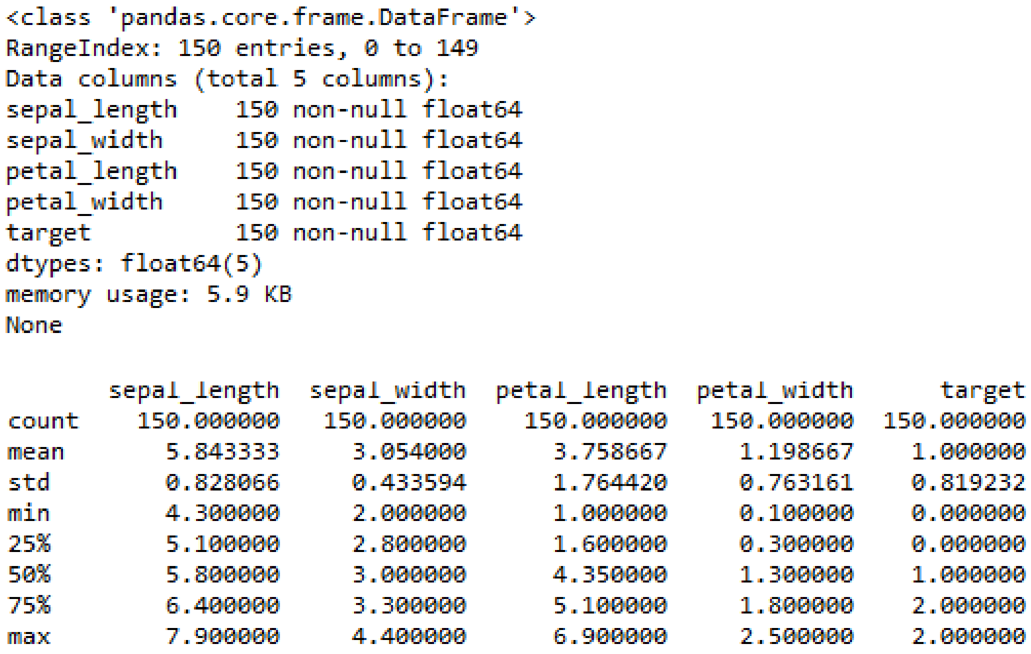
```
   sepal_length  sepal_width  petal_length  petal_width  target
0           5.1          3.5           1.4          0.2     0.0
1           4.9          3.0           1.4          0.2     0.0
2           4.7          3.2           1.3          0.2     0.0
3           4.6          3.1           1.5          0.2     0.0
4           5.0          3.6           1.4          0.2     0.0
5           5.4          3.9           1.7          0.4     0.0
6           4.6          3.4           1.4          0.3     0.0
7           5.0          3.4           1.5          0.2     0.0
8           4.4          2.9           1.4          0.2     0.0
9           4.9          3.1           1.5          0.1     0.0
[ 0.92154126  0.04875355  0.01851016  0.00716384  0.00403119]
```

2. Data Quality

Check the basic statistics of iris dataset.

```
print(iris.info())
print(iris.describe())
```

Note that no missing values in iris dataset.



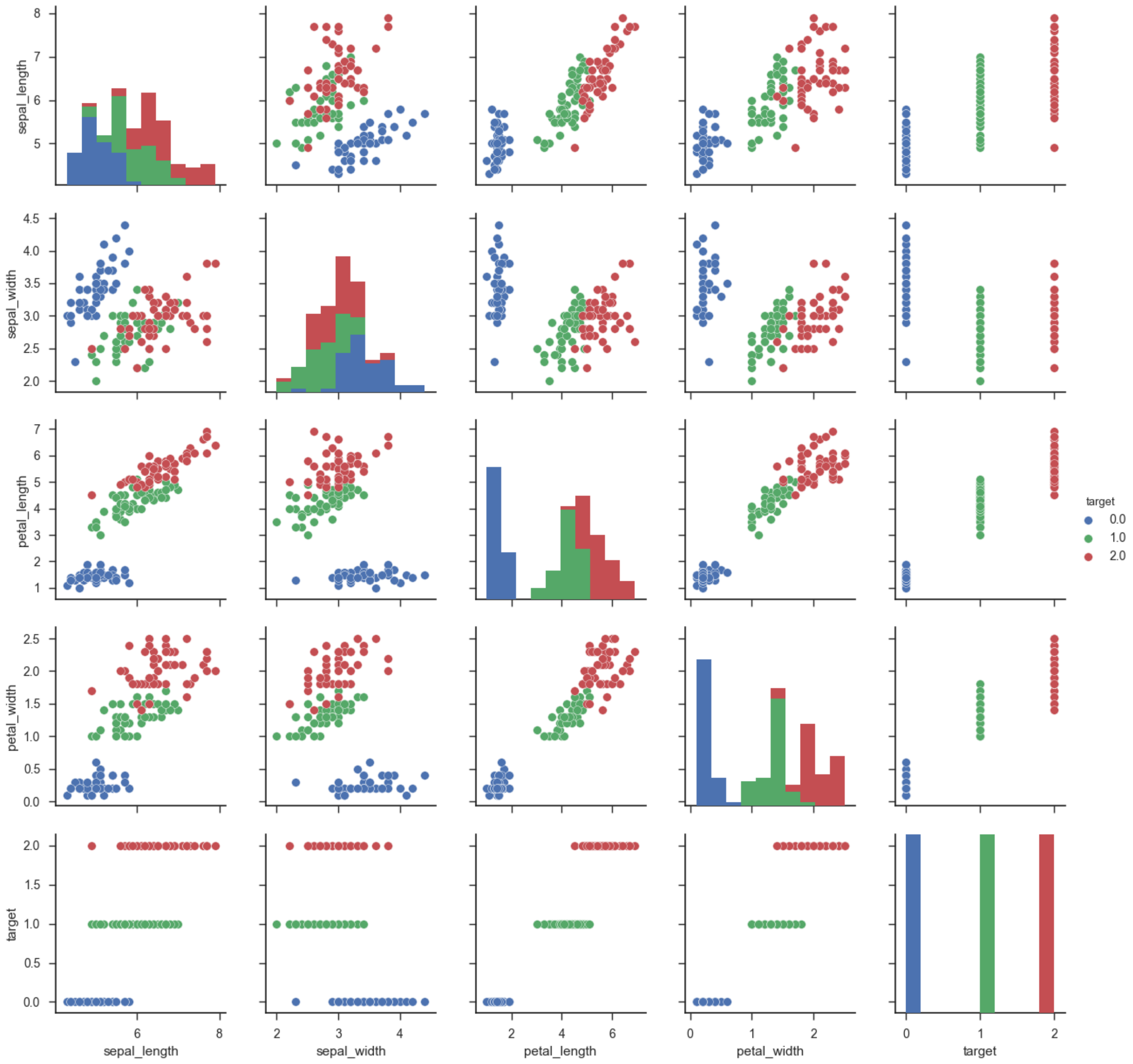
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepal_length    150 non-null float64
sepal_width     150 non-null float64
petal_length    150 non-null float64
petal_width     150 non-null float64
target          150 non-null float64
dtypes: float64(5)
memory usage: 5.9 KB
None

       sepal_length  sepal_width  petal_length  petal_width      target
count    150.000000   150.000000    150.000000   150.000000  150.000000
mean       5.843333     3.054000      3.758667     1.198667    1.000000
std        0.828066     0.433594      1.764420     0.763161    0.819232
min        4.300000     2.000000      1.000000     0.100000    0.000000
25%        5.100000     2.800000      1.600000     0.300000    0.000000
50%        5.800000     3.000000      4.350000     1.300000    1.000000
75%        6.400000     3.300000      5.100000     1.800000    2.000000
max        7.900000     4.400000      6.900000     2.500000    2.000000
```

Inspect the scattter matrix plot of iris dataset.

```
#The scatter-matrix of iris
sns.set(style="ticks")
sns.pairplot(iris,hue="target")
```

```python
plt.savefig("Scatter Matrix.png")
plt.show()
```



Check the duplicated rows and remove them.

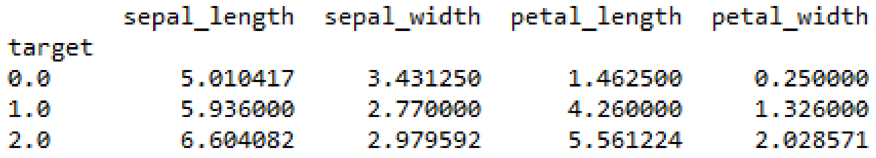
```python
print('number of duplicated:',iris.duplicated().sum())
iris=iris.drop_duplicates()
```

```
number of duplicated: 3
```

3. Aggregation

Aggregate by species using mean.

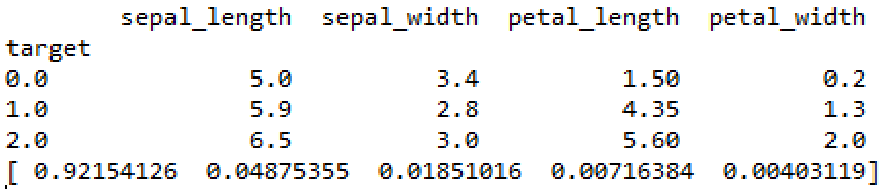
```
grouped_mean=iris.groupby('target').mean()
print(grouped_mean)
```



```
        sepal_length  sepal_width  petal_length  petal_width
target
0.0         5.010417     3.431250      1.462500     0.250000
1.0         5.936000     2.770000      4.260000     1.326000
2.0         6.604082     2.979592      5.561224     2.028571
```

Aggregate by species using mean.

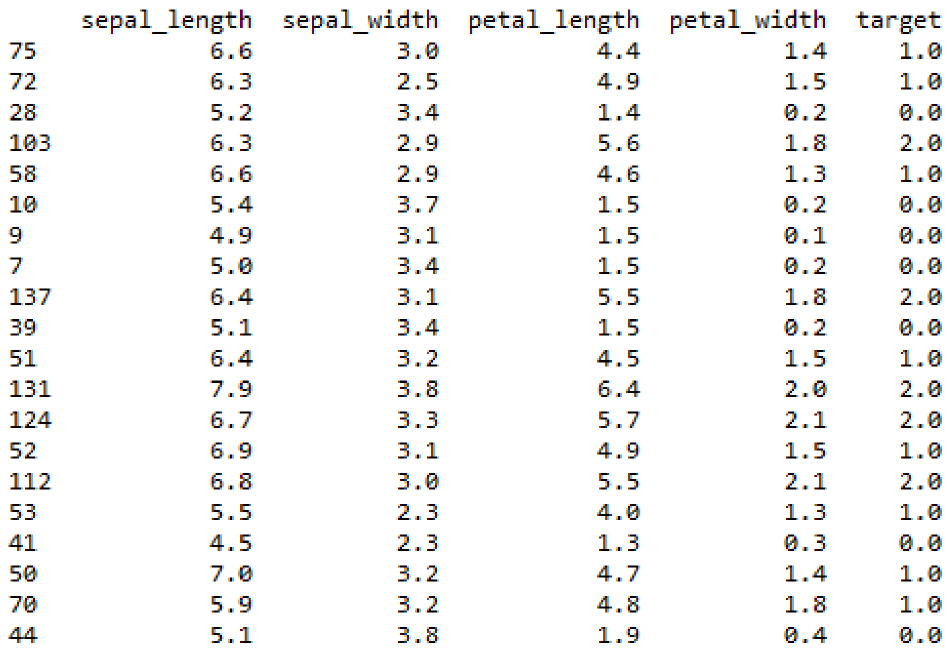
```
grouped_median=iris.groupby('target').median()
print(grouped_median)
```



```
        sepal_length  sepal_width  petal_length  petal_width
target
0.0              5.0          3.4          1.50          0.2
1.0              5.9          2.8          4.35          1.3
2.0              6.5          3.0          5.60          2.0
[ 0.92154126  0.04875355  0.01851016  0.00716384  0.00403119]
```

4. Random Sampling

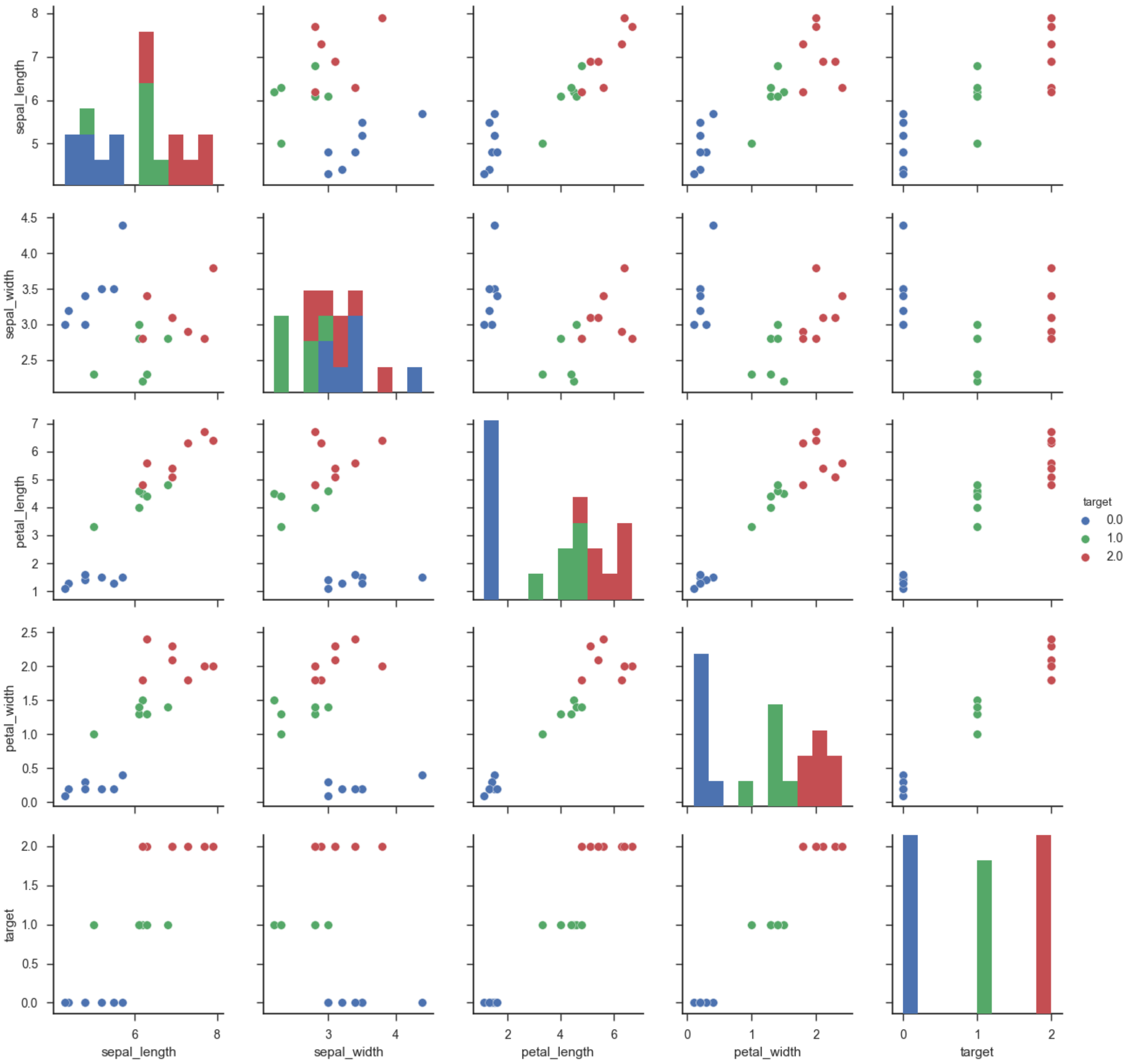
```
random_sampled=iris.take(np.random.permutation(len(iris))[:20])
print(random_sampled)
```



```
     sepal_length  sepal_width  petal_length  petal_width  target
75            6.6          3.0           4.4          1.4     1.0
72            6.3          2.5           4.9          1.5     1.0
28            5.2          3.4           1.4          0.2     0.0
103           6.3          2.9           5.6          1.8     2.0
58            6.6          2.9           4.6          1.3     1.0
10            5.4          3.7           1.5          0.2     0.0
9             4.9          3.1           1.5          0.1     0.0
7             5.0          3.4           1.5          0.2     0.0
137           6.4          3.1           5.5          1.8     2.0
39            5.1          3.4           1.5          0.2     0.0
51            6.4          3.2           4.5          1.5     1.0
131           7.9          3.8           6.4          2.0     2.0
124           6.7          3.3           5.7          2.1     2.0
52            6.9          3.1           4.9          1.5     1.0
112           6.8          3.0           5.5          2.1     2.0
53            5.5          2.3           4.0          1.3     1.0
41            4.5          2.3           1.3          0.3     0.0
50            7.0          3.2           4.7          1.4     1.0
70            5.9          3.2           4.8          1.8     1.0
44            5.1          3.8           1.9          0.4     0.0
```

Scatter plot on sampling dataset.

```
sns.set(style="ticks")
sns.pairplot(random_sampled,hue="target")
plt.savefig('sampling')
plt.show()
```

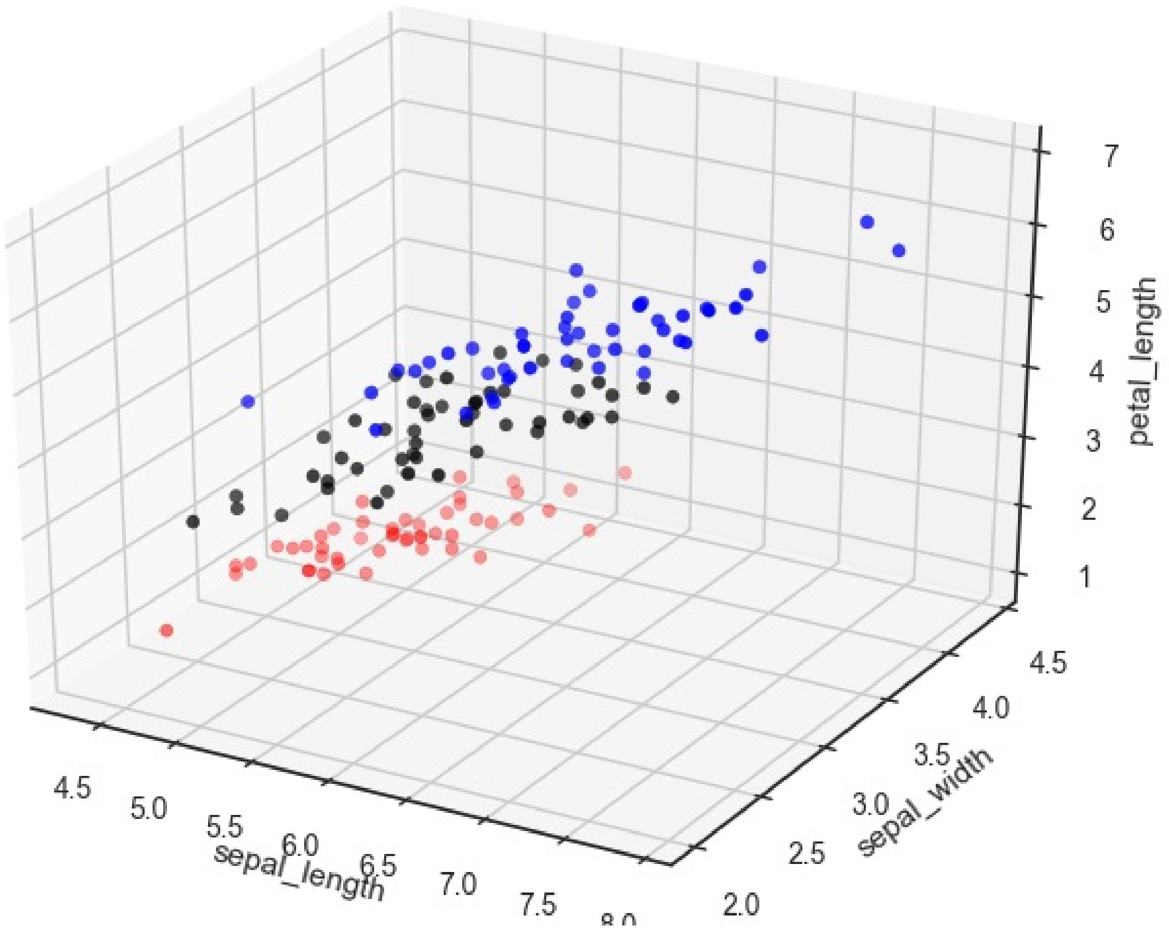


5. Dimensionality reduction (Principal Components Analysis - PCA)

Plot 3D scatter plot for the first three features.

```
from mpl_toolkits.mplot3d import Axes3D
fig=plt.figure()
ax=Axes3D(fig)
colors=['red','k','blue']
x_vals=iris.sepal_length; y_vals=iris.sepal_width;
```

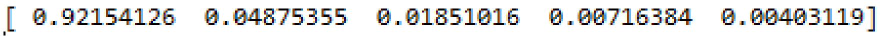
```
z_vals=iris.petal_length
ax.scatter(x_vals,y_vals,z_vals,c=iris.target.apply(lambda x: colors[int(x)]))
ax.set_xlabel('sepal_length')
ax.set_ylabel('sepal_width')
ax.set_zlabel('petal_length')
plt.show()
```



Find the suitable components for PCA.

```
from sklearn.decomposition import PCA
pca=PCA()
pca.fit(iris)
print(pca.explained_variance_ratio_)
```

Note that only the first dimension matters, but in this example we consider the first two dimensions.

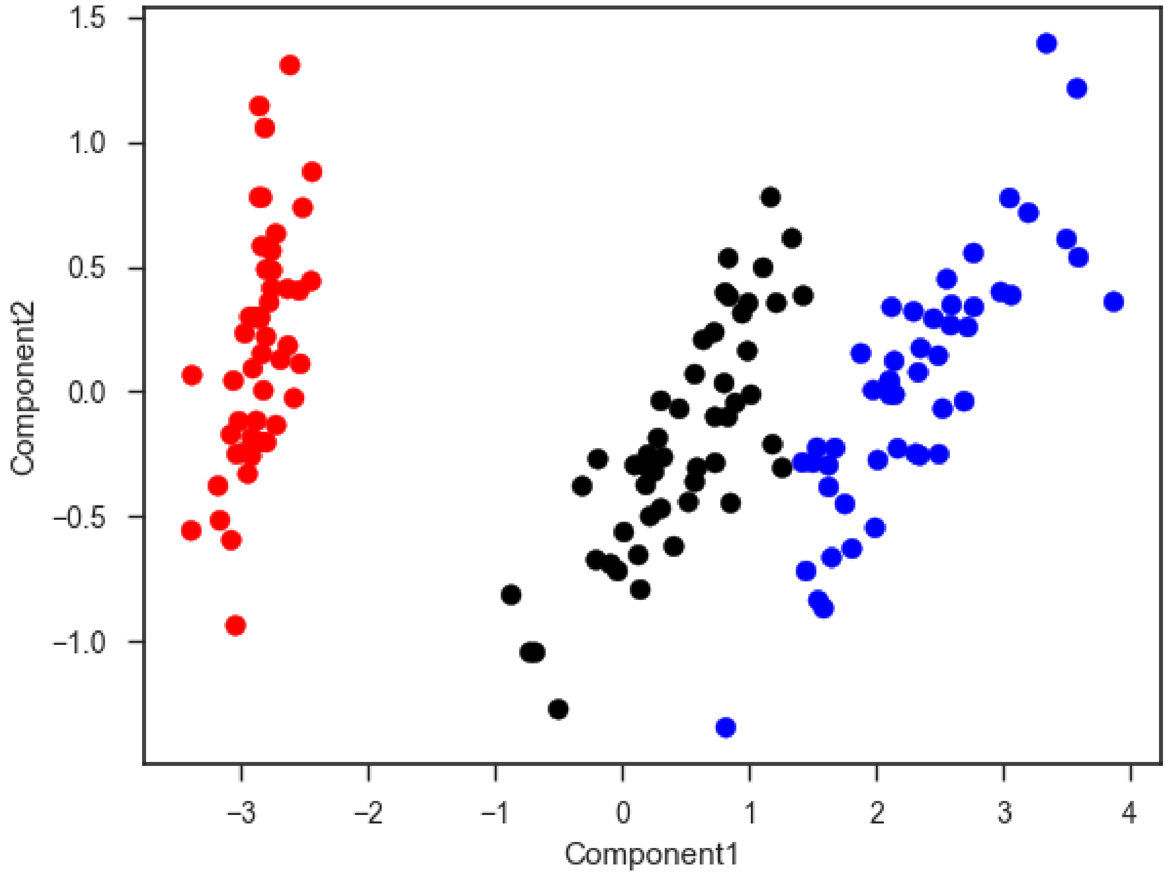


```
[ 0.92154126  0.04875355  0.01851016  0.00716384  0.00403119]
```

```python
pca.n_components=2
iris_reduced=pca.fit_transform(iris)
```

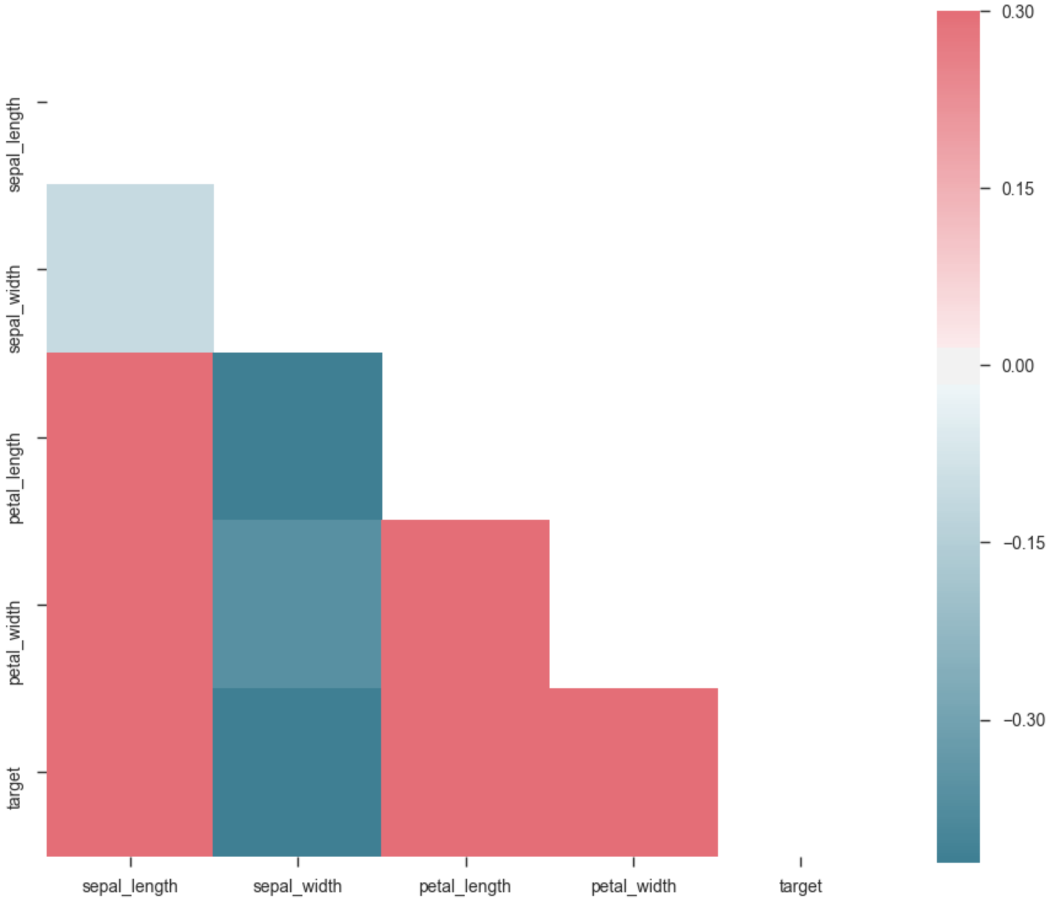
Plot the scatter of new reduced dataset.

```python
fig=plt.figure()
ax1=fig.add_subplot(111)
x_val=iris_reduced[:,0];y_val=iris_reduced[:,1]
colors=['red','k','blue']
ax1.scatter(x_val,y_val,c=iris.target.apply(lambda x:colors[int(x)]))
ax1.set_xlabel('Component1')
ax1.set_ylabel('Component2')
```



6 Correlation Matrix

```
correlation_matrix=iris.corr()
mask = np.zeros_like(correlation_matrix, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True
f, ax = plt.subplots(figsize=(11, 9))
cmap = sns.diverging_palette(220, 10, as_cmap=True)
sns.heatmap(correlation_matrix, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, ax=ax)
plt.show()
```



To be continued.....