In [1]:

```
# maha ebrahim mohammed
# 4051350
# IA8G
# lab 6: : Dealing with Missing Values & PCA

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(color_codes=True)
```

In [7]:

```
from sklearn import datasets
dataset=datasets.load_iris()

data=pd.DataFrame(dataset['data'],columns=['patel length','petal width','sepal length','sep
data['species'] = dataset['target']
data['species']=data['species'].apply(lambda x: dataset['target_names'][x])
data.head(10)
```

Out[7]:

	patel length	petal width	sepal length	sepal width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa

In [8]:

```
data.describe()
```

Out[8]:

	patel length	petal width	sepal length	sepal width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [9]:

```
data.isnull().sum()
```

Out[9]:

```
patel length    0
petal width     0
sepal length    0
sepal width     0
species         0
dtype: int64
```

In [16]:

```
modData = data.append({'patel length' : np.nan, 'petal width' : 3.6, 'sepal length' : 0,
                       'sepal width' : 0.2, 'species' : 'setosa'} , ignore_index=True)
modData.describe()
```

Out[16]:

	patel length	petal width	sepal length	sepal width
count	150.000000	151.000000	151.000000	151.000000
mean	5.843333	3.060927	3.733113	1.192715
std	0.828066	0.436650	1.785785	0.764033
min	4.300000	2.000000	0.000000	0.100000
25%	5.100000	2.800000	1.550000	0.300000
50%	5.800000	3.000000	4.300000	1.300000
75%	6.400000	3.350000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [20]:

```
print('Columns with missing values')
print(modData.isnull().sum())
print('\n columns with zero values')
print((modData[['patel length','petal width','sepal length','sepal width','species']]==0).s
```

```
Columns with missing values
patel length    1
petal width     0
sepal length    0
sepal width     0
species         0
dtype: int64

 columns with zero values
patel length    0
petal width     0
sepal length    1
sepal width     0
species         0
dtype: int64
```

In [21]:

```
modData[['patel length','petal width','sepal length','sepal width','species']] = modData[['
print('Columns with missing values')
print(modData.isnull().sum())
```

```
Columns with missing values
patel length    1
petal width     0
sepal length    1
sepal width     0
species         0
dtype: int64
```

In [27]:

```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

modData.fillna(modData.mean(), inplace=True)
print(modData.isnull().sum())
```

```
patel length    0
petal width     0
sepal length    0
sepal width     0
species         0
dtype: int64
```

In [28]:

```
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

modData.fillna(modData.median(), inplace=True)
print(modData.isnull().sum())
```

```
patel length    0
petal width     0
sepal length    0
sepal width     0
species         0
dtype: int64
```

In [2]:

```
url ='https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data'
PCA_df= pd.read_csv(url, names = ['patel length','petal width','sepal length','sepal width'
```

In [4]:

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
features = ['sepal length','sepal width','patel length','petal width']
x = PCA_df.loc[:, features].values
y = PCA_df.loc[:,['target']].values
x = StandardScaler().fit_transform(x)
print (x)
```

```
[[-1.34127240e+00 -1.31297673e+00 -9.00681170e-01  1.03205722e+00]
 [-1.34127240e+00 -1.31297673e+00 -1.14301691e+00 -1.24957601e-01]
 [-1.39813811e+00 -1.31297673e+00 -1.38535265e+00  3.37848329e-01]
 [-1.28440670e+00 -1.31297673e+00 -1.50652052e+00  1.06445364e-01]
 [-1.34127240e+00 -1.31297673e+00 -1.02184904e+00  1.26346019e+00]
 [-1.17067529e+00 -1.05003079e+00 -5.37177559e-01  1.95766909e+00]
 [-1.34127240e+00 -1.18150376e+00 -1.50652052e+00  8.00654259e-01]
 [-1.28440670e+00 -1.31297673e+00 -1.02184904e+00  8.00654259e-01]
 [-1.34127240e+00 -1.31297673e+00 -1.74885626e+00 -3.56360566e-01]
 [-1.28440670e+00 -1.44444970e+00 -1.14301691e+00  1.06445364e-01]
 [-1.28440670e+00 -1.31297673e+00 -5.37177559e-01  1.49486315e+00]
 [-1.22754100e+00 -1.31297673e+00 -1.26418478e+00  8.00654259e-01]
 [-1.34127240e+00 -1.44444970e+00 -1.26418478e+00 -1.24957601e-01]
 [-1.51186952e+00 -1.44444970e+00 -1.87002413e+00 -1.24957601e-01]
 [-1.45500381e+00 -1.31297673e+00 -5.25060772e-02  2.18907205e+00]
 [-1.28440670e+00 -1.05003079e+00 -1.73673948e-01  3.11468391e+00]
 [-1.39813811e+00 -1.05003079e+00 -5.37177559e-01  1.95766909e+00]
 [-1.34127240e+00 -1.18150376e+00 -9.00681170e-01  1.03205722e+00]
 [-1.17067529e+00 -1.18150376e+00 -1.73673948e-01  1.72626612e+00]
```

In [5]:

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents
                           , columns = ['principal component 1', 'principal component 2'])
```

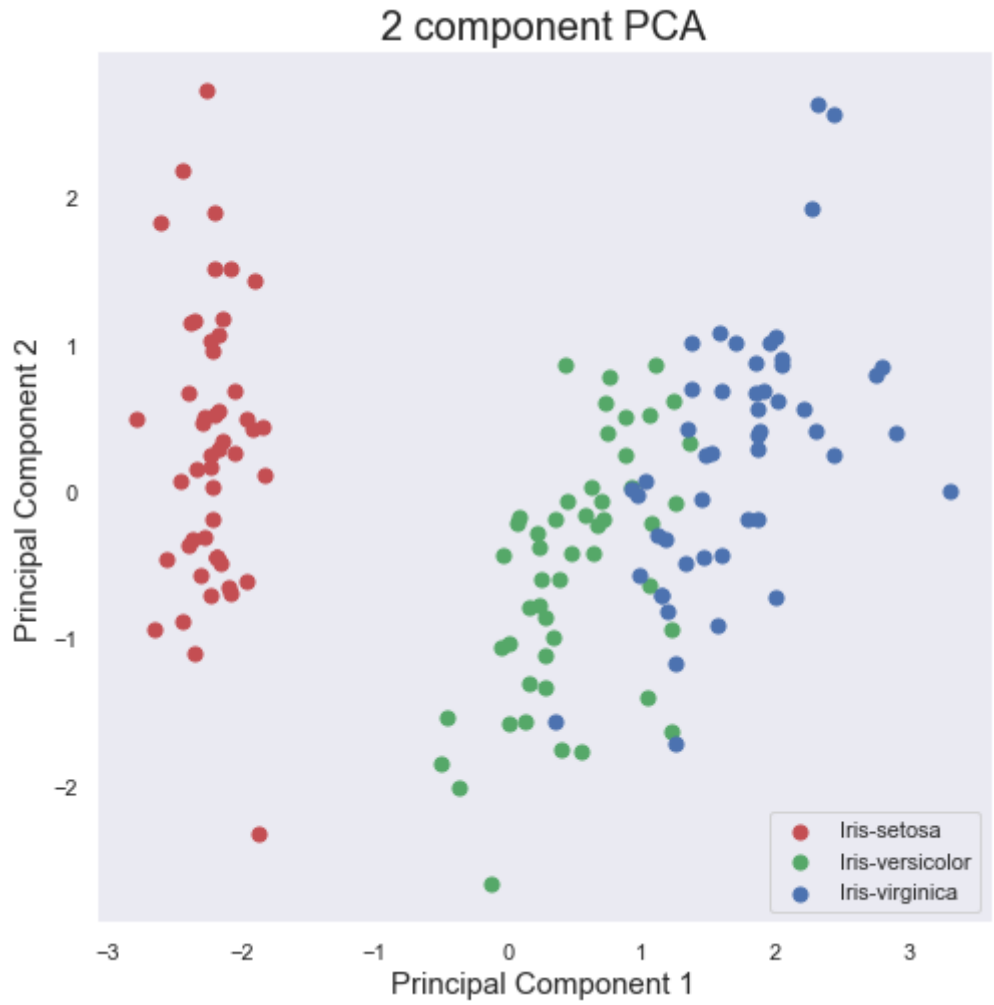
In [6]:

```
finalDf = pd.concat([principalDf, PCA_df[['target']]], axis = 1)
```

In [11]:

```
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(color_codes=True)

fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal Component 1', fontsize = 15)
ax.set_ylabel('Principal Component 2', fontsize = 15)
ax.set_title('2 component PCA', fontsize = 20)
targets = ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
colors = ['r', 'g', 'b']
for target, color in zip(targets, colors):
    indicesToKeep = finalDf['target'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
               , finalDf.loc[indicesToKeep,'principal component 2']
               , c = color
               , s = 50)
ax.legend(targets)
ax.grid()
```



In [12]:

```
print(pca.explained_variance_ratio_)
```

```
[0.72770452 0.23030523]
```

In [ ]:

```

```