```
import sklearn
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarnin
  import pandas.util.testing as tm
```

```
from sklearn.datasets import load_iris
iris = load_iris()
y1=iris.target
x1= iris.data
```

```
df = pd.DataFrame(x1,
                  columns = iris.feature_names)
```

```
x1
```

```
df['species']=y1
```

```
df['petal width (cm)'].unique().tolist()
```

```
[0.2,
 0.4,
 0.3,
 0.1,
 0.5,
 0.6,
 1.4,
 1.5,
 1.3,
 1.6,
 1.0,
 1.1,
 1.8,
 1.2,
 1.7,
 2.5,
 1.9,
 2.1,
 2.2,
 2.0,
 2.4,
 2.3]
```

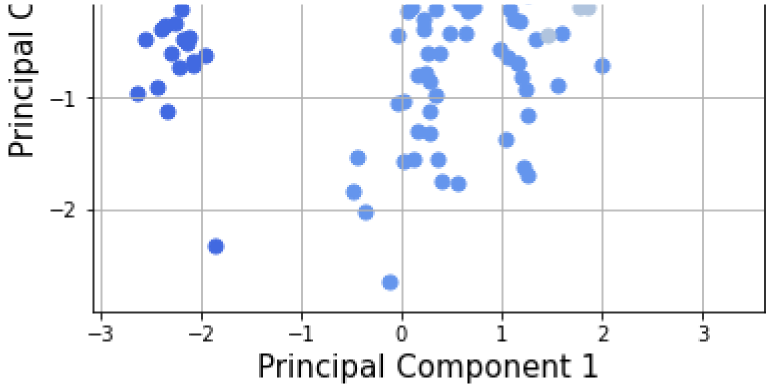
```
def categorize(x):
    if x < 4:
        return 0
```

```
    elif x>=4 and x<=5.5:
        return 1
    else:
        return 2
def categorize2(x):
    if x < 5:
        return 0
    elif x>=5 and x<=6:
        return 1
    else:
        return 2
def categorize3(x):
    if x < 3:
        return 0
    elif x>=3 and x<=4:
        return 1
    else:
        return 2
def categorize4(x):
    if x < 1:
        return 0
    elif x>=1 and x<=2:
        return 1
    else:
        return 2
df['petal length cat']= 0
for i in range(150):
  df['petal length cat'][i]= categorize(df['petal length (cm)'][i])
df['petal width cat']= 0
for i in range(150):
  df['petal width cat'][i]= categorize4(df['petal width (cm)'][i])
df['sepal length cat']= 0
for i in range(150):
  df['sepal length cat'][i]= categorize2(df['sepal length (cm)'][i])
df['sepal width cat']= 0
for i in range(150):
  df['sepal width cat'][i]= categorize3(df['sepal width (cm)'][i])
df
```

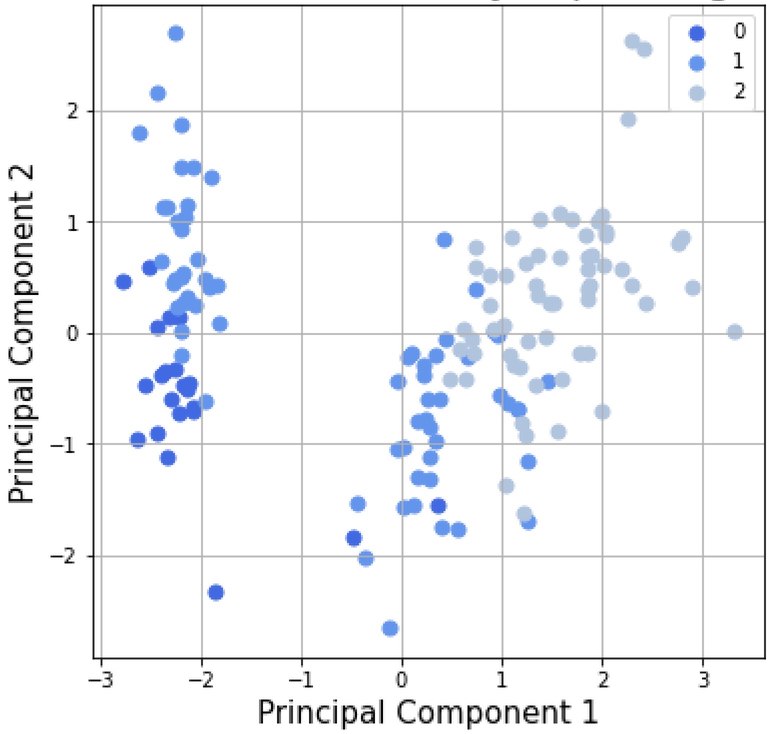
```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:31: SettingWithCopyWarni
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:34: SettingWithCopyWarni
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:37: SettingWithCopyWarni
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:40: SettingWithCopyWarni
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	specie
0	5.1	3.5	1.4	0.2	
1	4.9	3.0	1.4	0.2	

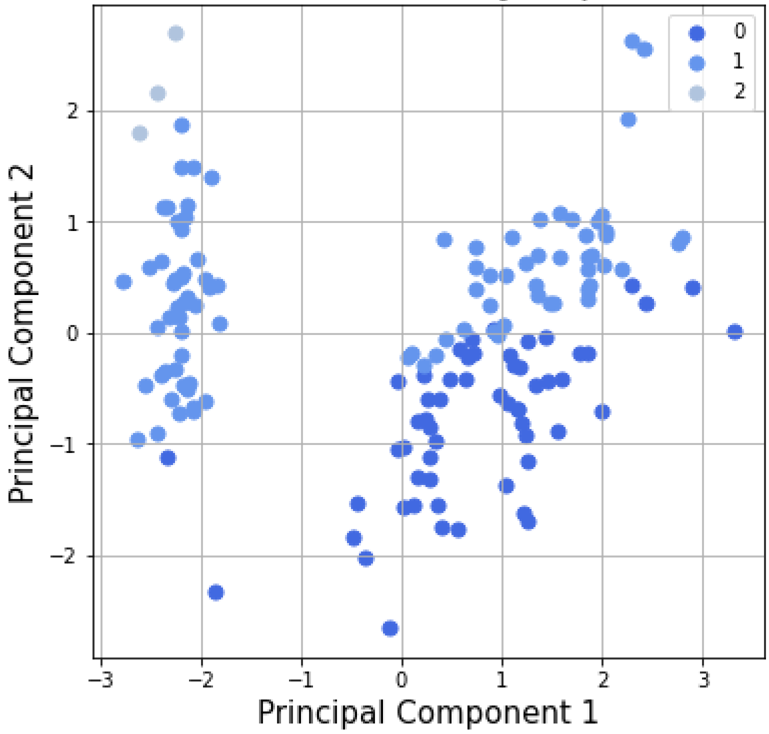
```
from sklearn.preprocessing import StandardScaler
# Standardizing the features
x = StandardScaler().fit_transform(x1)
#principal component analysis
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component 2'])

for i in ['petal length cat','petal width cat','sepal length cat','sepal width cat']:
  finalDf = pd.concat([principalDf, df[[i]]], axis = 1)
  fig = plt.figure(figsize = (6,6))
  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('PCA characterized by {0}'.format(i[:-4]), fontsize = 20)
  targets = [0, 1, 2]
  colors = ['royalblue', 'cornflowerblue', 'lightsteelblue']
  for target, color in zip(targets,colors):
      indicesToKeep = finalDf[i] == target
      ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
               , finalDf.loc[indicesToKeep, 'principal component 2']
               , c = color
               , s = 50)
  ax.legend(targets)
  ax.grid()
```



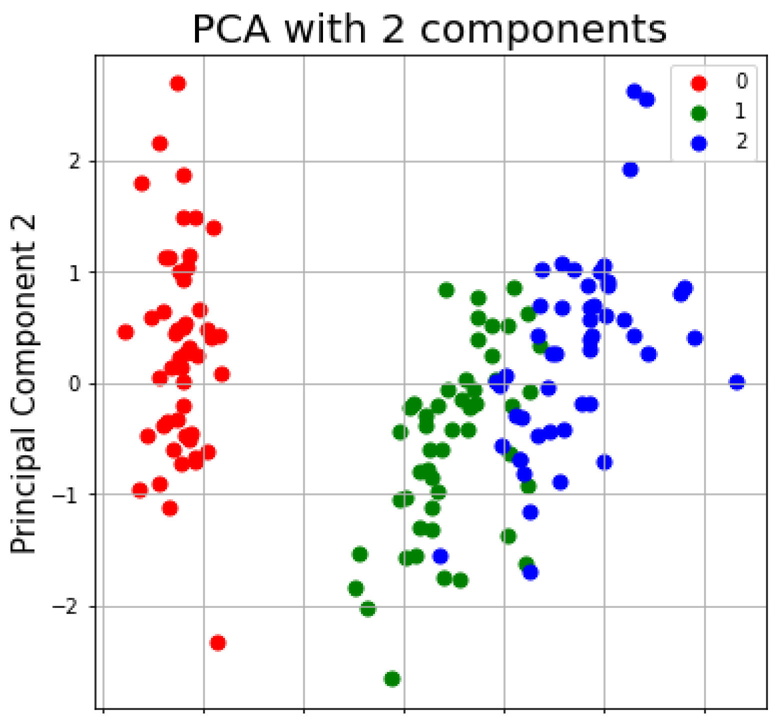
PCA characterized by sepal length



PCA characterized by sepal width

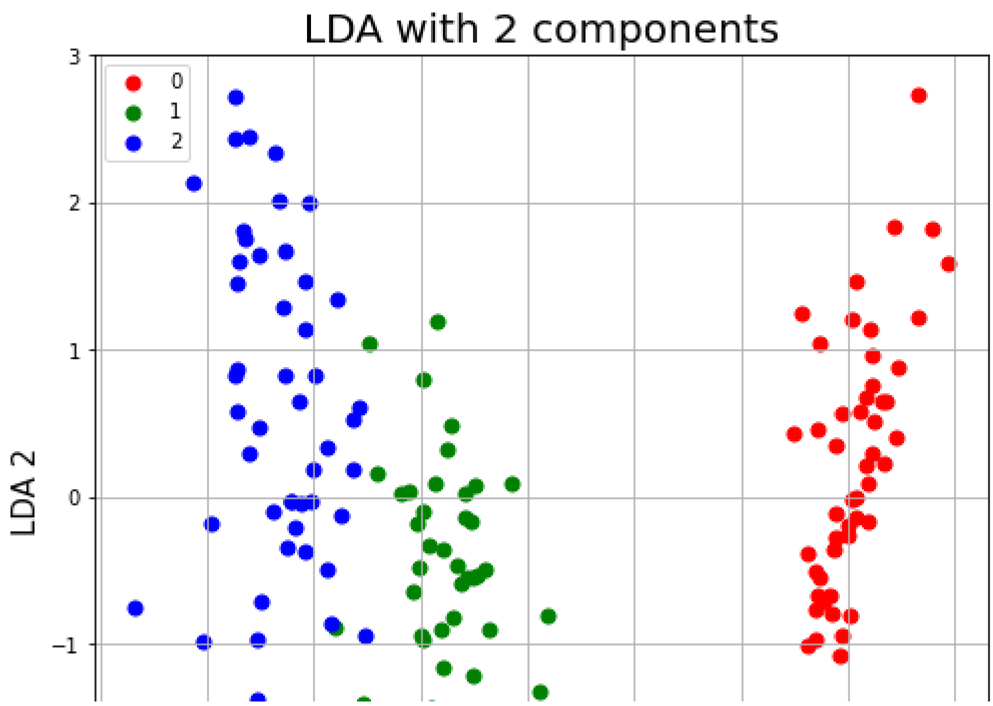


```
from sklearn.preprocessing import StandardScaler
# Standardizing the features
x = StandardScaler().fit_transform(x1)
#principal component analysis
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component 2'])
finalDf = pd.concat([principalDf, df[['species']]], axis = 1)
fig = plt.figure(figsize = (6,6))
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('PCA with 2 components', fontsize = 20)
targets = [0, 1, 2]
colors = ['r', 'g', 'b']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['species'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
               , finalDf.loc[indicesToKeep, 'principal component 2']
               , c = color
               , s = 50)
ax.legend(targets)
ax.grid()
```



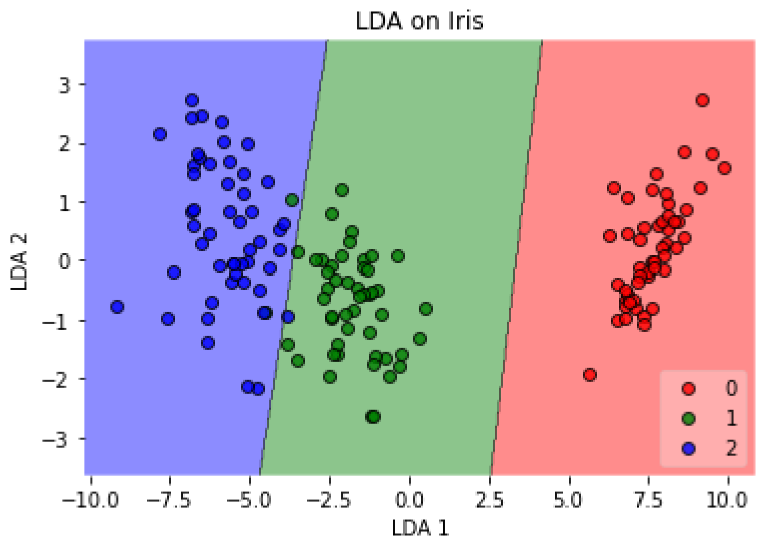
Q2 part2

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
sklearn_lda = LDA(n_components=2)
x = StandardScaler().fit_transform(x1)
X_lda_sklearn = sklearn_lda.fit_transform(x, y1)

principalDf = pd.DataFrame(data = X_lda_sklearn
             , columns = ['LDA component 1', 'LDA component 2'])
finalDf = pd.concat([principalDf, df[['species']]], axis = 1)
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('LDA 1', fontsize = 15)
ax.set_ylabel('LDA 2', fontsize = 15)
ax.set_title('LDA with 2 components', fontsize = 20)
targets = [0, 1, 2]
colors = ['r', 'g', 'b']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['species'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'LDA component 1']
               , finalDf.loc[indicesToKeep, 'LDA component 2']
               , c = color
               , s = 50)
ax.legend(targets)
ax.grid()
```




```
from mlxtend.plotting import plot_decision_regions
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.svm import SVC
svm = SVC(C=0.5, kernel='linear')
svm.fit(X_lda_sklearn, y1)
plot_decision_regions(X_lda_sklearn, y1, clf=svm, legend=4,colors="r,g,b",markers="oooooo"
plt.xlabel('LDA 1')
plt.ylabel('LDA 2')
plt.title('LDA on Iris')
plt.show()
```

```
/usr/local/lib/python3.6/dist-packages/mlxtend/plotting/decision_regions.py:244: Matp
  ax.axis(xmin=xx.min(), xmax=xx.max(), y_min=yy.min(), y_max=yy.max())
```




```
X_lda_sklearn
```

```
[ 7.12868772e+00, -7.86660426e-01],
[ 7.48982797e+00, -2.65384488e-01],
[ 6.81320057e+00, -6.70631068e-01],
[ 8.13230933e+00,  5.14462530e-01],
[ 7.70194674e+00,  1.46172097e+00],
[ 7.21261762e+00,  3.55836209e-01],
[ 7.60529355e+00, -1.16338380e-02],
[ 6.56055159e+00, -1.01516362e+00],
[ 7.34305989e+00, -9.47319209e-01],
[ 8.39738652e+00,  6.47363392e-01],
[ 7.21929685e+00, -1.09646389e-01],
[ 7.32679599e+00, -1.07298943e+00],
[ 7.57247066e+00, -8.05464137e-01],
[ 9.84984300e+00,  1.58593698e+00],
[ 9.15823890e+00,  2.73759647e+00],
[ 8.58243141e+00,  1.83448945e+00],
[ 7.78075375e+00,  5.84339407e-01],
[ 8.07835876e+00,  9.68580703e-01],
[ 8.02097451e+00,  1.14050366e+00],
[ 7.49680227e+00, -1.88377220e-01],
[ 7.58648117e+00,  1.20797032e+00],
[ 8.68104293e+00,  8.77590154e-01],
[ 6.25140358e+00,  4.39696367e-01],
[ 6.55893336e+00, -3.89222752e-01],
[ 6.77138315e+00, -9.70634453e-01],
[ 6.82308032e+00,  4.63011612e-01],
[ 7.92461638e+00,  2.09638715e-01],
[ 7.99129024e+00,  8.63787128e-02],
[ 6.82946447e+00, -5.44960851e-01],
[ 6.75895493e+00, -7.59002759e-01],
[ 7.37495254e+00,  5.65844592e-01],
[ 9.12634625e+00,  1.22443267e+00],
[ 9.46768199e+00,  1.82522635e+00],
[ 7.06201386e+00, -6.63400423e-01],
[ 7.95876243e+00, -1.64961722e-01],
[ 8.61367201e+00,  4.03253602e-01],
[ 8.33041759e+00,  2.28133530e-01],
[ 6.93412007e+00, -7.05519379e-01],
[ 7.68823131e+00, -9.22362309e-03],
[ 7.91793715e+00,  6.75121313e-01],
[ 5.66188065e+00, -1.93435524e+00],
[ 7.24101468e+00, -2.72615132e-01],
[ 6.41443556e+00,  1.24730131e+00],
[ 6.85944381e+00,  1.05165396e+00],
[ 6.76470393e+00, -5.05151855e-01],
[ 8.08189937e+00,  7.63392750e-01],
[ 7.18676904e+00, -3.60986823e-01],
[ 8.31444876e+00,  6.44953177e-01],
[ 7.67196741e+00, -1.34893840e-01],
[-1.45927545e+00,  2.85437643e-02],
[-1.79770574e+00,  4.84385502e-01],
[-2.41694888e+00, -9.27840307e-02],
[-2.26247349e+00, -1.58725251e+00],
[-2.54867836e+00, -4.72204898e-01],
[-2.42996725e+00, -9.66132066e-01],
[-2.44848456e+00,  7.95961954e-01],
[-2.22666513e-01, -1.58467318e+00],
[-1.75020123e+00, -8.21180130e-01],
[-1.95842242e+00, -3.51563753e-01],
[-1.19376031e+00, -2.63445570e+00],
[-1.85892567e+00,  3.19006544e-01],
[-1.15809388e+00, -2.64340991e+00],
```

```
       [-2.66605725e+00, -6.42504540e-01],
       [-3.78367218e-01,  8.66389312e-02],
       [-1.20117255e+00,  8.44373592e-02],
       [-2.76810246e+00,  3.21995363e-02],
       [-7.76854039e-01, -1.65916185e+00],
       [-3.49805433e+00, -1.68495616e+00],
       [-1.09042788e+00, -1.62658350e+00],
       [-3.71589615e+00,  1.04451442e+00],
       [-9.97610366e-01, -4.90530602e-01],
       [-3.83525931e+00, -1.40595806e+00],
       [-2.25741249e+00, -1.42679423e+00],
       [-1.25571326e+00, -5.46424197e-01],
       [-1.43755762e+00, -1.34424979e-01],
       [-2.45906137e+00, -9.35277280e-01],
       [-3.51848495e+00,  1.60588866e-01],
       [-2.58979871e+00, -1.74611728e-01],
       [ 3.07487884e-01, -1.31887146e+00],
       [-1.10669179e+00, -1.75225371e+00],
       [-6.05524589e-01, -1.94298038e+00],
       [-8.98703769e-01, -9.04940034e-01],
       [-4.49846635e+00, -8.82749915e-01],
       [-2.93397799e+00,  2.73791065e-02],
       [-2.10360821e+00,  1.19156767e+00],
       [-2.14258208e+00,  8.87797815e-02],
       [-2.47945603e+00, -1.94073927e+00],
       [-1.32552574e+00, -1.62869550e-01],
       [-1.95557887e+00, -1.15434826e+00],
       [-2.40157020e+00, -1.59458341e+00],
       [-2.29248878e+00, -3.32860296e-01],
       [-1.27227224e+00, -1.21458428e+00],
       [-2.93176055e-01, -1.79871509e+00],
       [-2.00598883e+00, -9.05418042e-01],
       [-1.18166311e+00, -5.37570242e-01],
       [-1.61615645e+00, -4.70103580e-01],
       [-1.42158879e+00, -5.51244626e-01],
       [ 4.75973788e-01, -7.99905482e-01],
       [-1.54948259e+00, -5.93363582e-01],
       [-7.83947399e+00,  2.13973345e+00],
       [-5.50747997e+00, -3.58139892e-02],
       [-6.29200850e+00,  4.67175777e-01],
       [-5.60545633e+00, -3.40738058e-01],
       [-6.85055995e+00,  8.29825394e-01],
       [-7.41816784e+00, -1.73117995e-01],
       [-4.67799541e+00, -4.99095015e-01],
       [-6.31692685e+00, -9.68980756e-01],
       [-6.32773684e+00, -1.38328993e+00],
       [-6.85281335e+00,  2.71758963e+00],
       [-4.44072512e+00,  1.34723692e+00],
       [-5.45009572e+00, -2.07736942e-01],
       [-5.66033713e+00,  8.32713617e-01],
       [-5.95823722e+00, -9.40175447e-02],
       [-6.75926282e+00,  1.60023206e+00],
       [-5.80704331e+00,  2.01019882e+00],
       [-5.06601233e+00, -2.62733839e-02],
       [-6.60881882e+00,  1.75163587e+00],
       [-9.17147486e+00, -7.48255067e-01],
       [-4.76453569e+00, -2.15573720e+00],
       [-6.27283915e+00,  1.64948141e+00],
       [-5.36071189e+00,  6.46120732e-01],
       [-7.58119982e+00, -9.80722934e-01],
       [-4.37150279e+00, -1.21297458e-01],
```

```
       [-5.72317531e+00,  1.29327553e+00],
       [-5.27915920e+00, -4.24582377e-02],
       [-4.08087208e+00,  1.85936572e-01],
       [-4.07703640e+00,  5.23238483e-01],
       [-6.51910397e+00,  2.96976389e-01],
       [-4.58371942e+00, -8.56815813e-01],
       [-6.22824009e+00, -7.12719638e-01],
       [-5.22048773e+00,  1.46819509e+00],
       [-6.80015000e+00,  5.80895175e-01],
       [-3.81515972e+00, -9.42985932e-01],
       [-5.10748966e+00, -2.13059000e+00],
       [-6.79671631e+00,  8.63090395e-01],
       [-6.52449599e+00,  2.44503527e+00],
       [-4.99550279e+00,  1.87768525e-01],
       [-3.93985300e+00,  6.14020389e-01],
       [-5.20383090e+00,  1.14476808e+00],
       [-6.65308685e+00,  1.80531976e+00],
       [-5.10555946e+00,  1.99218201e+00],
       [-5.50747997e+00, -3.58139892e-02],
       [-6.79601924e+00,  1.46068695e+00],
       [-6.84735943e+00,  2.42895067e+00],
       [-5.64500346e+00,  1.67771734e+00],
       [-5.17956460e+00, -3.63475041e-01],
```

```
df_new = pd.DataFrame(x1,
                      columns = iris.feature_names)
```

```
df_new['species']=y1
df_1=df_new.loc[df['species']<2]
df_1.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

```
abc=df_1.to_numpy()
```

```
y1[0:50]
```

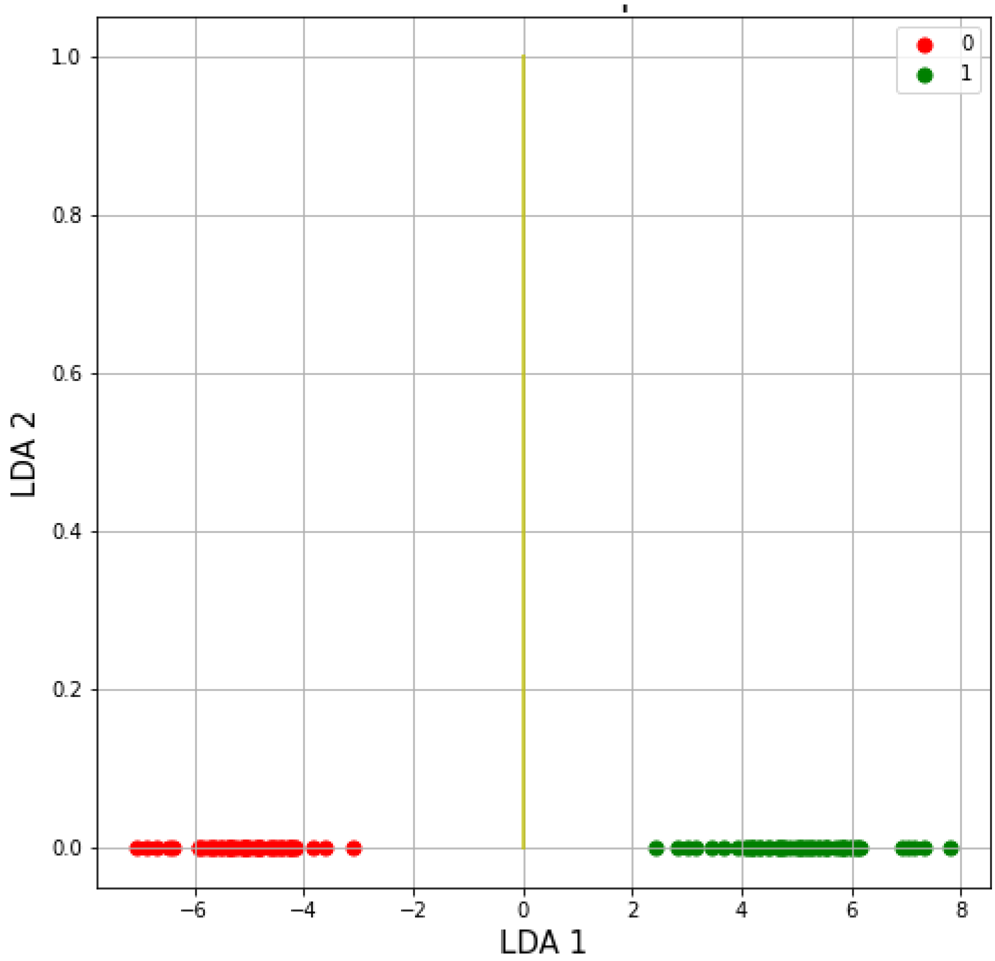
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
x1[0:49,:].shape
```

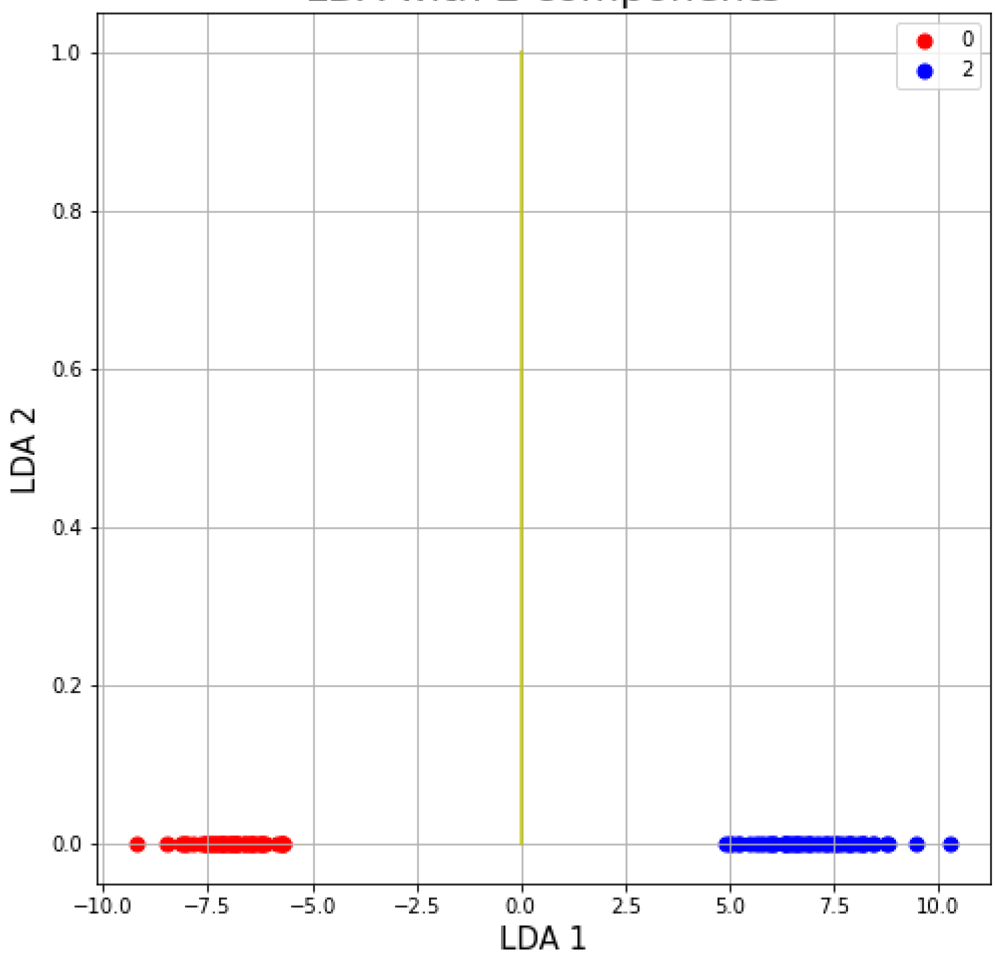
```
(49, 4)
```

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
import matplotlib.pyplot as pp
import matplotlib.lines as lines
def plot_at_y(arr, val, **kwargs):
    pp.plot(arr, np.zeros_like(arr) + val, 'o', **kwargs)
    pp.show()
#1,2
sklearn_lda = LDA(n_components=1)
x = StandardScaler().fit_transform(x1[50:150,:])
X_lda_sklearn = sklearn_lda.fit_transform(x, y1[50:150])
print(y1[50:150])
principalDf = pd.DataFrame(data = X_lda_sklearn
             , columns = ['LDA component 1'])
df_st= pd.DataFrame( y1[50:150] ,columns=['species'] )
finalDf = pd.concat([principalDf, df_st], axis = 1)
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('LDA 1', fontsize = 15)
ax.set_ylabel('LDA 2', fontsize = 15)
ax.set_title('LDA with 2 components', fontsize = 20)
targets = [1, 2]
colors = ['g', 'b']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['species'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'LDA component 1'],[[0] for i in range(50)]
               , c = color
               , s = 50)
ax.legend(targets,)
line = lines.Line2D([0 , 0 ],
                    [0 , 1],
                    lw = 2, color ='y',
                    axes = ax, alpha = 0.7)

ax.add_line(line)

ax.grid()

#0,1
sklearn_lda = LDA(n_components=1)
x = StandardScaler().fit_transform(x1[0:100,:])
X_lda_sklearn = sklearn_lda.fit_transform(x, y1[0:100])
principalDf = pd.DataFrame(data = X_lda_sklearn
             , columns = ['LDA component 1'])
df_st= pd.DataFrame( y1[0:100] ,columns=['species'] )
finalDf = pd.concat([principalDf, df_st], axis = 1)
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('LDA 1', fontsize = 15)
```

```
ax.set_ylabel('LDA 2', fontsize = 15)
ax.set_title('LDA with 2 components', fontsize = 20)
targets = [0, 1]
colors = ['r', 'g']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['species'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'LDA component 1'],[[0] for i in range(50)]
               , c = color
               , s = 50)
ax.legend(targets)
line = lines.Line2D([0 , 0 ],
                    [0 , 1],
                    lw = 2, color ='y',
                    axes = ax, alpha = 0.7)

ax.add_line(line)

ax.grid()

#0,2
a= x1[0:50]
a=np.concatenate((a,x1[100:150]))
b= y1[0:50]
b = np.concatenate((b,y1[100:150]))
df_st= pd.DataFrame( b ,columns=['species'] )
sklearn_lda = LDA(n_components=1)
x = StandardScaler().fit_transform(a)
X_lda_sklearn = sklearn_lda.fit_transform(x, b)
principalDf = pd.DataFrame(data = X_lda_sklearn
             , columns = ['LDA component 1'])
finalDf = pd.concat([principalDf, df_st], axis = 1)
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('LDA 1', fontsize = 15)
ax.set_ylabel('LDA 2', fontsize = 15)
ax.set_title('LDA with 2 components', fontsize = 20)
targets = [0, 2]
colors = ['r', 'b']
for target, color in zip(targets,colors):
    indicesToKeep = finalDf['species'] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'LDA component 1'],[[0] for i in range(50)]
               , c = color
               , s = 50)
ax.legend(targets)
line = lines.Line2D([0 , 0 ],
                    [0 , 1],
                    lw = 2, color ='y',
                    axes = ax, alpha = 0.7)

ax.add_line(line)

ax.grid()
```



LDA with 2 components

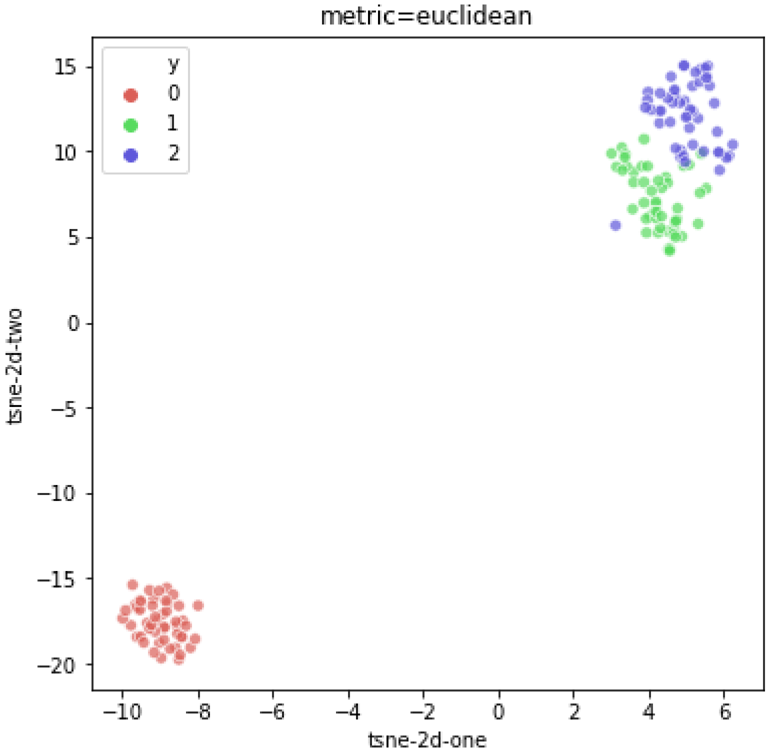


Q2 part3

```
from sklearn.manifold import TSNE
```

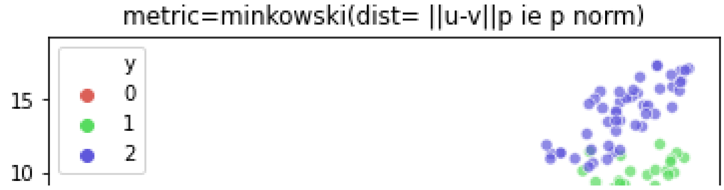
```
tsne=TSNE(n_components=2,perplexity=40,n_iter=1000).fit_transform(x1)
df_subset=pd.DataFrame(x1,columns=iris.feature_names)
df_subset['y']=y1
df_subset['tsne-2d-one'] = tsne[:,0]
df_subset['tsne-2d-two'] = tsne[:,1]
plt.figure(figsize=(6,6))
sns.scatterplot(
    x="tsne-2d-one", y="tsne-2d-two",
    hue="y",
    palette=sns.color_palette("hls", 3),
    data=df_subset,
    legend="full",
    alpha=0.7,
)
plt.title("metric=euclidean")
```

```
Text(0.5, 1.0, 'metric=euclidean')
```



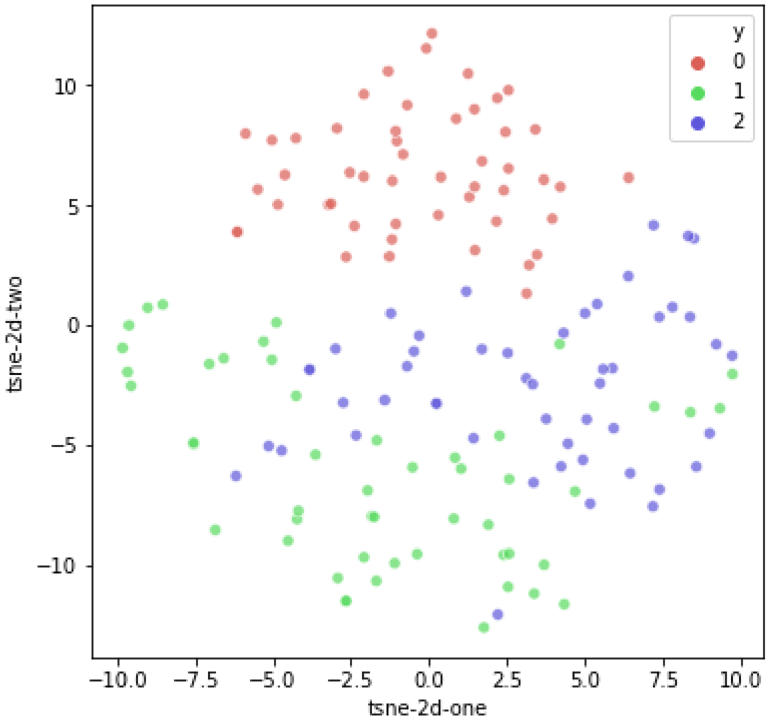

```
tsne=TSNE(n_components=2,perplexity=40,n_iter=1000,metric='minkowski').fit_transform(x1)
df_subset=pd.DataFrame(x1,columns=iris.feature_names)
df_subset['y']=y1
df_subset['tsne-2d-one'] = tsne[:,0]
df_subset['tsne-2d-two'] = tsne[:,1]
plt.figure(figsize=(6,6))
sns.scatterplot(
    x="tsne-2d-one", y="tsne-2d-two",
    hue="y",
    palette=sns.color_palette("hls", 3),
    data=df_subset,
    legend="full",
    alpha=0.7
)
plt.title("metric=minkowski(dist= ||u-v||p ie p norm)")
```

```
Text(0.5, 1.0, 'metric=minkowski(dist= ||u-v||p ie p norm)')
```




```
tsne=TSNE(n_components=2,perplexity=40,n_iter=1000,metric='hamming').fit_transform(x1)
df_subset=pd.DataFrame(x1,columns=iris.feature_names)
df_subset['y']=y1
df_subset['tsne-2d-one'] = tsne[:,0]
df_subset['tsne-2d-two'] = tsne[:,1]
plt.figure(figsize=(6,6))
sns.scatterplot(
    x="tsne-2d-one", y="tsne-2d-two",
    hue="y",
    palette=sns.color_palette("hls", 3),
    data=df_subset,
    legend="full",
    alpha=0.7
)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f40c01c0048>
```




```
tsne=TSNE(n_components=3,perplexity=40,n_iter=1000).fit_transform(x1)
df_subset['tsne-2d-one'] = tsne[:,0]
df_subset['tsne-2d-two'] = tsne[:,1]
df_subset['tsne-2d-three'] = tsne[:,2]
plt.figure(figsize=(8,8))
ax = plt.figure(figsize=(8,8)).gca(projection='3d')
ax.scatter(
    xs=df_subset["tsne-2d-one"],
    ys=df_subset["tsne-2d-two"],
    zs=df_subset["tsne-2d-three"],
```