HW3 Part 1

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.datasets import load_breast_cancer
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

In [2]:
```
breast = load_breast_cancer()
```

In [3]:
```
breast_data = breast.data
breast_data.shape
```

Out[3]:
```
(569, 30)
```

In [4]:
```
breast_input = pd.DataFrame(breast_data)
breast_input.head()
```

Out[4]:

	0	1	2	3	4	5	6	7	8	9	...	20	21	
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23.57	25.53	152
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14.91	26.50	98
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22.54	16.67	152

5 rows × 30 columns

In [5]:
```
breast_labels = breast.target
```

In [6]:
```
breast_labels.shape
```

Out[6]:
```
(569,)
```

In [7]:
```
labels = np.reshape(breast_labels,(569,1))
```

In [8]:
```
final_breast_data = np.concatenate([breast_data,labels],axis=1)
```

In [9]:
```
final_breast_data.shape
```

Out[9]:
```
(569, 31)
```

In [10]:
```
breast_dataset = pd.DataFrame(final_breast_data)
```

In [11]:
```
features = breast.feature_names
features
```

Out[11]:
```
array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter error', 'area error',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error',
       'fractal dimension error', 'worst radius', 'worst texture',
       'worst perimeter', 'worst area', 'worst smoothness',
       'worst compactness', 'worst concavity', 'worst concave points',
       'worst symmetry', 'worst fractal dimension'], dtype='<U23')
```

In [12]:
```
features_labels = np.append(features,'label')
```

In [13]:
```
breast_dataset.columns = features_labels
```

In [14]:
```
breast_dataset.head()
```

Out[14]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	m fra dimen
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05

5 rows × 31 columns

In [15]:
```
breast_dataset.tail()
```

Out[15]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	dim
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	dim
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	C
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	C

5 rows × 31 columns

In [16]:
```
from sklearn.preprocessing import StandardScaler
x = breast_dataset.loc[:, features].values
y = breast_dataset['label'].values
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size = 0.8, test_size =
sc_X = StandardScaler()
sc_X.fit(x_train)
x_train = sc_X.transform(x_train)
x_test = sc_X.transform(x_test)
```

In [17]:
```
breast_dataset['label'].replace(0, 'Benign',inplace=True)
breast_dataset['label'].replace(1, 'Malignant',inplace=True)
```

In [18]:
```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state=0)
classifier.fit(x_train, y_train)
```

Out[18]:
```
LogisticRegression(random_state=0)
```

In [19]:
```
y_pred = classifier.predict(x_test)
```

In [20]:
```
y_pred[0:30]
```

Out[20]:
```
array([1., 1., 1., 0., 1., 1., 1., 1., 0., 0., 1., 1., 1., 1., 0., 0., 0.,
       0., 0., 1., 1., 1., 1., 1., 0., 1., 0., 0., 0., 1.])
```

In [21]:
```
from sklearn.metrics import confusion_matrix
cnf_matrix = confusion_matrix(y_test, y_pred)
cnf_matrix
```

Out[21]:
```
array([[44,  3],
       [ 1, 66]], dtype=int64)
```

In [22]:
```
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))
```

```
Accuracy: 0.9649122807017544
Precision: 0.9565217391304348
```

```
Recall: 0.9850746268656716
```

In [ ]:

HW3 Part 2

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.datasets import load_breast_cancer
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

In [2]:
```
breast = load_breast_cancer()
```

In [3]:
```
breast_data = breast.data
breast_data.shape
```

Out[3]:
```
(569, 30)
```

In [4]:
```
breast_input = pd.DataFrame(breast_data)
breast_input.head()
```

Out[4]:

	0	1	2	3	4	5	6	7	8	9	...	20	21	
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23.57	25.53	152
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14.91	26.50	98
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22.54	16.67	152

5 rows × 30 columns

In [5]:
```
breast_labels = breast.target
```

In [6]:
```
breast_labels.shape
```

Out[6]:
```
(569,)
```

In [7]:
```
labels = np.reshape(breast_labels,(569,1))
```

In [8]:
```
final_breast_data = np.concatenate([breast_data,labels],axis=1)
```

In [9]:
```
final_breast_data.shape
```

```
Out[9]: (569, 31)
```

```
In [10]: breast_dataset = pd.DataFrame(final_breast_data)
```

```
In [11]: features = breast.feature_names
         features
```

```
Out[11]: array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
                'mean smoothness', 'mean compactness', 'mean concavity',
                'mean concave points', 'mean symmetry', 'mean fractal dimension',
                'radius error', 'texture error', 'perimeter error', 'area error',
                'smoothness error', 'compactness error', 'concavity error',
                'concave points error', 'symmetry error',
                'fractal dimension error', 'worst radius', 'worst texture',
                'worst perimeter', 'worst area', 'worst smoothness',
                'worst compactness', 'worst concavity', 'worst concave points',
                'worst symmetry', 'worst fractal dimension'], dtype='<U23')
```

```
In [12]: features_labels = np.append(features,'label')
```

```
In [13]: breast_dataset.columns = features_labels
```

```
In [14]: breast_dataset.head()
```

Out[14]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	m fra dimen
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05

5 rows × 31 columns

```
In [15]: breast_dataset.tail()
```

Out[15]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	dim
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	dim
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	C
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	C

5 rows × 31 columns

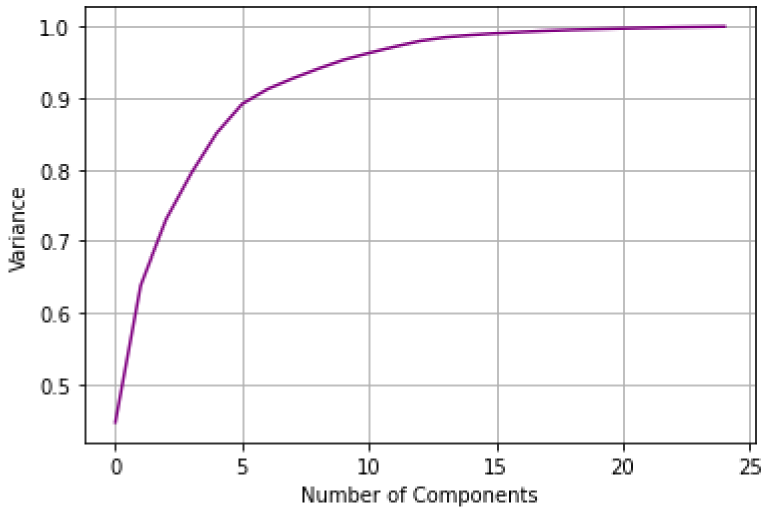
In [16]:
```
x = breast_dataset.loc[:, features].values
y = breast_dataset['label'].values
```

In [17]:
```
from sklearn.preprocessing import StandardScaler
x = breast_dataset.loc[:, features].values
y = breast_dataset['label'].values
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size = 0.8, test_size =
sc_X = StandardScaler()
sc_X.fit(x_train)
x_train = sc_X.transform(x_train)
x_test = sc_X.transform(x_test)
```

In [18]:
```
breast_dataset['label'].replace(0, 'Benign',inplace=True)
breast_dataset['label'].replace(1, 'Malignant',inplace=True)
```

In [19]:
```
from sklearn.decomposition import PCA
PCA_25 = PCA(n_components=25)
PCA_25.fit(x_train)
xPCA_25 = PCA_25.transform(x_train)
```

In [20]:
```
plt.plot(np.cumsum((PCA_25.explained_variance_ratio_)), color = 'purple')
plt.xlabel('Number of Components')
plt.ylabel('Variance')
plt.grid()
```



In [21]:
```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state=0)
classifier.fit(x_train, y_train)
```

Out[21]:
```
LogisticRegression(random_state=0)
```

In [22]:
```
y_pred = classifier.predict(x_test)
```

In [23]:
```
y_pred[0:30]
```

Out[23]:
```
array([0., 0., 0., 1., 1., 0., 1., 0., 1., 0., 0., 1., 1., 1., 1., 0., 1.,
       0., 1., 1., 1., 0., 1., 1., 1., 0., 1., 1., 1., 1.])
```

In [24]:
```
from sklearn.metrics import confusion_matrix
cnf_matrix = confusion_matrix(y_test, y_pred)
cnf_matrix
```

Out[24]:
```
array([[42,  2],
       [ 0, 70]], dtype=int64)
```

In [25]:
```
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))
```

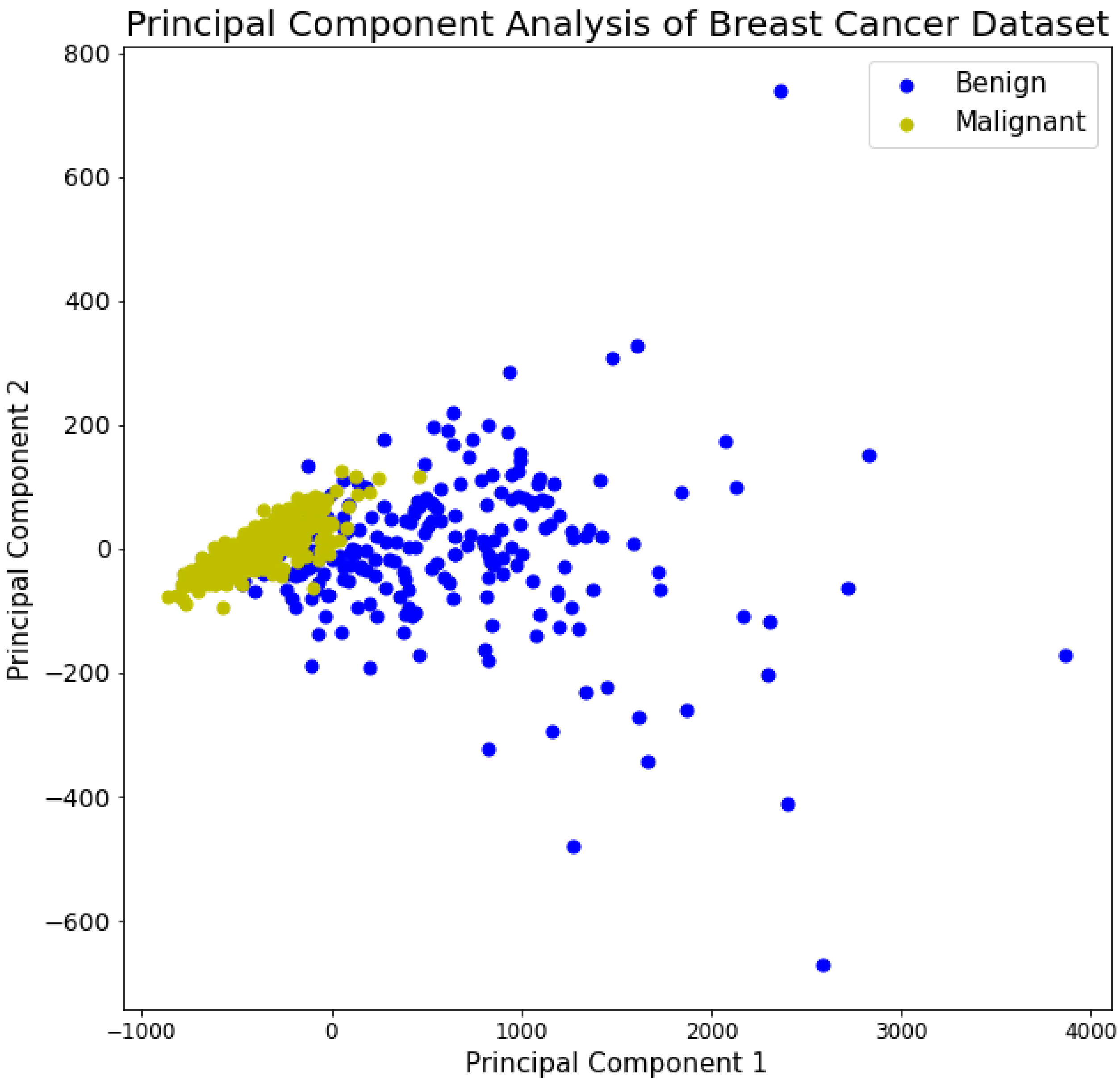
```
Accuracy: 0.9824561403508771
Precision: 0.9722222222222222
Recall: 1.0
```

In [26]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents, columns = ['principal component
```

In [27]:
```
finalDf = pd.concat([principalDf, breast_dataset[['label']]], axis = 1)
```

In [28]:
```
plt.figure()
plt.figure(figsize=(10,10))
plt.xticks(fontsize=12)
plt.yticks(fontsize=14)
plt.xlabel('Principal Component 1', fontsize = 15)
plt.ylabel('Principal Component 2', fontsize = 15)
plt.title('Principal Component Analysis of Breast Cancer Dataset', fontsize = 20)
targets = ['Benign', 'Malignant']
colors = ['b', 'y']
for target, color in zip(targets,colors):
    indicesToKeep = breast_dataset['label'] == target
    plt.scatter(finalDf.loc[indicesToKeep, 'principal component 1'], finalDf.loc[indice
plt.legend(targets,prop={'size': 15})
```

Out[28]:
```
<matplotlib.legend.Legend at 0x27cb0f4ad60>
```
```
<Figure size 432x288 with 0 Axes>
```



In [ ]:

HW3 Part 3

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.datasets import load_breast_cancer
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

In [2]:
```
breast = load_breast_cancer()
```

In [3]:
```
breast_data = breast.data
breast_data.shape
```

Out[3]:
```
(569, 30)
```

In [4]:
```
breast_input = pd.DataFrame(breast_data)
breast_input.head()
```

Out[4]:

	0	1	2	3	4	5	6	7	8	9	...	20	21	
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23.57	25.53	152
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14.91	26.50	98
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22.54	16.67	152

5 rows × 30 columns

In [5]:
```
breast_labels = breast.target
```

In [6]:
```
breast_labels.shape
```

Out[6]:
```
(569,)
```

In [7]:
```
labels = np.reshape(breast_labels,(569,1))
```

In [8]:
```
final_breast_data = np.concatenate([breast_data,labels],axis=1)
```

In [9]:
```
final_breast_data.shape
```

Out[9]:
```
(569, 31)
```

In [10]:
```
breast_dataset = pd.DataFrame(final_breast_data)
```

In [11]:
```
features = breast.feature_names
features
```

Out[11]:
```
array(['mean radius', 'mean texture', 'mean perimeter', 'mean area',
       'mean smoothness', 'mean compactness', 'mean concavity',
       'mean concave points', 'mean symmetry', 'mean fractal dimension',
       'radius error', 'texture error', 'perimeter error', 'area error',
       'smoothness error', 'compactness error', 'concavity error',
       'concave points error', 'symmetry error',
       'fractal dimension error', 'worst radius', 'worst texture',
       'worst perimeter', 'worst area', 'worst smoothness',
       'worst compactness', 'worst concavity', 'worst concave points',
       'worst symmetry', 'worst fractal dimension'], dtype='<U23')
```

In [12]:
```
features_labels = np.append(features,'label')
```

In [13]:
```
breast_dataset.columns = features_labels
```

In [14]:
```
breast_dataset.head()
```

Out[14]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	m fra dimen
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05

5 rows × 31 columns

In [15]:
```
breast_dataset.tail()
```

Out[15]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	dim
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	dim
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0

5 rows × 31 columns

In [16]:
```
x = breast_dataset.loc[:, features].values
y = breast_dataset['label'].values
```

In [17]:
```
from sklearn.preprocessing import StandardScaler
x = breast_dataset.loc[:, features].values
y = breast_dataset['label'].values
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size = 0.8, test_size =
sc_X = StandardScaler()
sc_X.fit(x_train)
x_train = sc_X.transform(x_train)
x_test = sc_X.transform(x_test)
```

In [18]:
```
breast_dataset['label'].replace(0, 'Benign',inplace=True)
breast_dataset['label'].replace(1, 'Malignant',inplace=True)
```

In [19]:
```
LDA = LinearDiscriminantAnalysis(n_components=1)
LDA_t = LDA.fit_transform(x_train,y_train)
```

In [20]:
```
from sklearn.naive_bayes import GaussianNB
gb = GaussianNB()
gb.fit(x_train, y_train)
y_pred = gb.predict(x_test)
```

In [21]:
```
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))
```

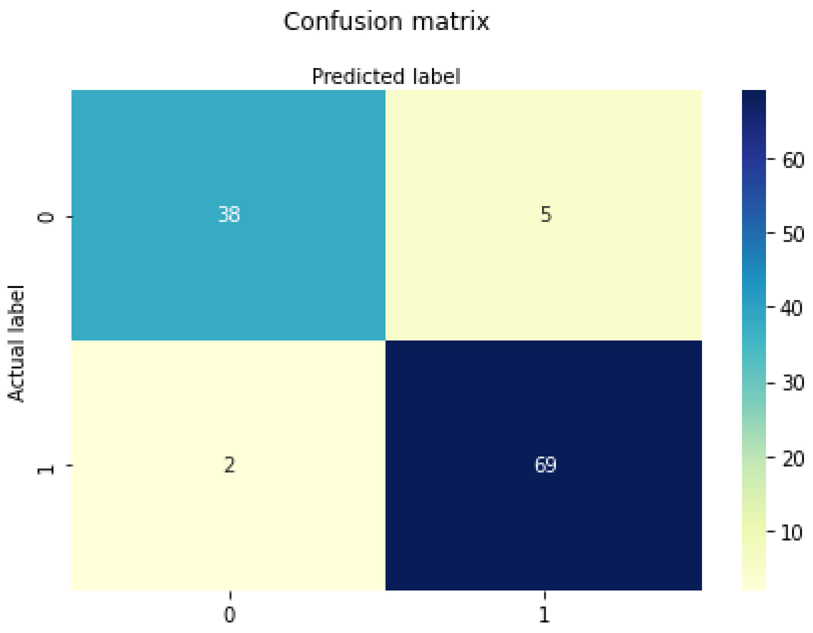
```
Accuracy: 0.9385964912280702
Precision: 0.9324324324324325
Recall: 0.971830985915493
```

In [22]:
```
cnf_matrix = confusion_matrix(y_test, y_pred)
```

In [23]:
```
class_names=[0,1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)

sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

Out[23]:
```
Text(0.5, 257.44, 'Predicted label')
```


Confusion matrix
Predicted label
Actual label