In [29]:

```python
import pandas as pd
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn import metrics
from sklearn.metrics import classification_report

cancer=load_breast_cancer()
cancerdf=pd.DataFrame(cancer.data,columns=cancer.feature_names)
print(cancerdf.head()) # head()默认显示前5行数据
```

```
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0        17.99         10.38          122.80     1001.0          0.11840   
1        20.57         17.77          132.90     1326.0          0.08474   
2        19.69         21.25          130.00     1203.0          0.10960   
3        11.42         20.38           77.58      386.1          0.14250   
4        20.29         14.34          135.10     1297.0          0.10030   

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419   
1           0.07864          0.0869              0.07017         0.1812   
2           0.15990          0.1974              0.12790         0.2069   
3           0.28390          0.2414              0.10520         0.2597   
4           0.13280          0.1980              0.10430         0.1809   

   mean fractal dimension  ...  worst radius  worst texture  worst perimeter  \
0                 0.07871  ...         25.38          17.33           184.60   
1                 0.05667  ...         24.99          23.41           158.80   
2                 0.05999  ...         23.57          25.53           152.50   
3                 0.09744  ...         14.91          26.50            98.87   
4                 0.05883  ...         22.54          16.67           152.20   

   worst area  worst smoothness  worst compactness  worst concavity  \
0      2019.0            0.1622             0.6656           0.7119   
1      1956.0            0.1238             0.1866           0.2416   
2      1709.0            0.1444             0.4245           0.4504   
3       567.7            0.2098             0.8663           0.6869   
4      1575.0            0.1374             0.2050           0.4000   

   worst concave points  worst symmetry  worst fractal dimension  
0                0.2654          0.4601                  0.11890  
1                0.1860          0.2750                  0.08902  
2                0.2430          0.3613                  0.08758  
3                0.2575          0.6638                  0.17300  
4                0.1625          0.2364                  0.07678  

[5 rows x 30 columns]
```

In [30]:

```python
print("肿瘤的分类：",cancer['target_names'])
print("肿瘤的分类：",cancer['feature_names'])
```

```
肿瘤的分类: ['malignant' 'benign']
肿瘤的分类: ['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']
```

In [31]:

```
x,y=cancer.data,cancer.target
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state = 100)
print(x_train.shape)# 查看训练集数据形态
print(x_test.shape)# 查看测试集数据形态
```

```
(455, 30)
(114, 30)
```

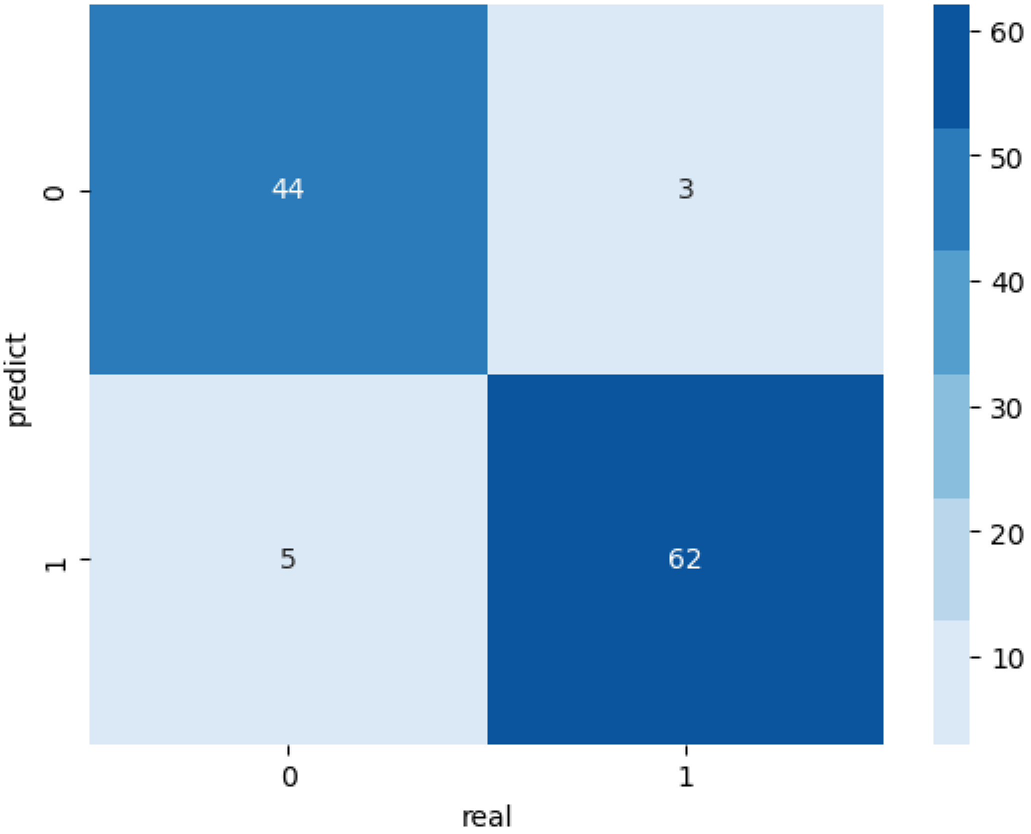
In [32]:

```
clf=GaussianNB()
clf.fit(x_train,y_train)#对训练集进行拟合
Y_pred= clf.predict(x_test)
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: 0.9298245614035088
Precision: 0.9253731343283582
Recall: 0.9538461538461539
```

In [33]:

```
pred=clf.predict(x_test)
cm=confusion_matrix(pred,y_test)
sns.heatmap(cm,cmap=sns.color_palette("Blues"),annot=True,fmt='d')
plt.xlabel('real')
plt.ylabel('predict')
plt.show()
```



In [108…

```
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.datasets import load_breast_cancer
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix

breast = load_breast_cancer()
breast_data = breast.data
breast_input = pd.DataFrame(breast_data)
breast_labels = breast.target
labels = np.reshape(breast_labels,(569,1))
final_breast_data = np.concatenate([breast_data,labels],axis=1)
breast_dataset = pd.DataFrame(final_breast_data)
features = breast.feature_names
features
```

Out[108]:

```
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 [109…

```
features_labels = np.append(features,'label')
breast_dataset.columns = features_labels
breast_dataset['label'].replace('Benign',0,inplace=True)
breast_dataset['label'].replace('Malignant',1,inplace=True)
breast_dataset.tail()
```

Out[109]:

| | mean radius | mean texture | mean perimeter | mean area | mean smoothness | mean compactness | mean concavity | mean concave points | mea symmetr |
|---|---|---|---|---|---|---|---|---|---|
| **564** | 21.56 | 22.39 | 142.00 | 1479.0 | 0.11100 | 0.11590 | 0.24390 | 0.13890 | 0.172 |
| **565** | 20.13 | 28.25 | 131.20 | 1261.0 | 0.09780 | 0.10340 | 0.14400 | 0.09791 | 0.175 |
| **566** | 16.60 | 28.08 | 108.30 | 858.1 | 0.08455 | 0.10230 | 0.09251 | 0.05302 | 0.159 |
| **567** | 20.60 | 29.33 | 140.10 | 1265.0 | 0.11780 | 0.27700 | 0.35140 | 0.15200 | 0.239 |
| **568** | 7.76 | 24.54 | 47.92 | 181.0 | 0.05263 | 0.04362 | 0.00000 | 0.00000 | 0.158 |

5 rows × 31 columns

In [110…

```
from sklearn.preprocessing import StandardScaler


x = breast_dataset.loc[:, features].values
x = StandardScaler().fit_transform(x)
```

In [111…
```
k=9      #设置降维的占比
pca= PCA(n_components=k)#调用PCA函数，先实例化
pcaCom = pca.fit_transform(x)
pcaCom = pd.DataFrame(pcaCom)
print("主成分的数量： ",pca.n_components_)
X = pcaCom.iloc[:, [0, 1, 2, 3, 4, 5, 6, 7, 8]].values
#X = pcaCom.iloc[:, [0, 1, 2, 3, 4, 5, 6, 7, 8]].values
Y = breast_dataset.iloc[:, 30].values
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size=0.8, test_size=
```

```
主成分的数量：  9
```

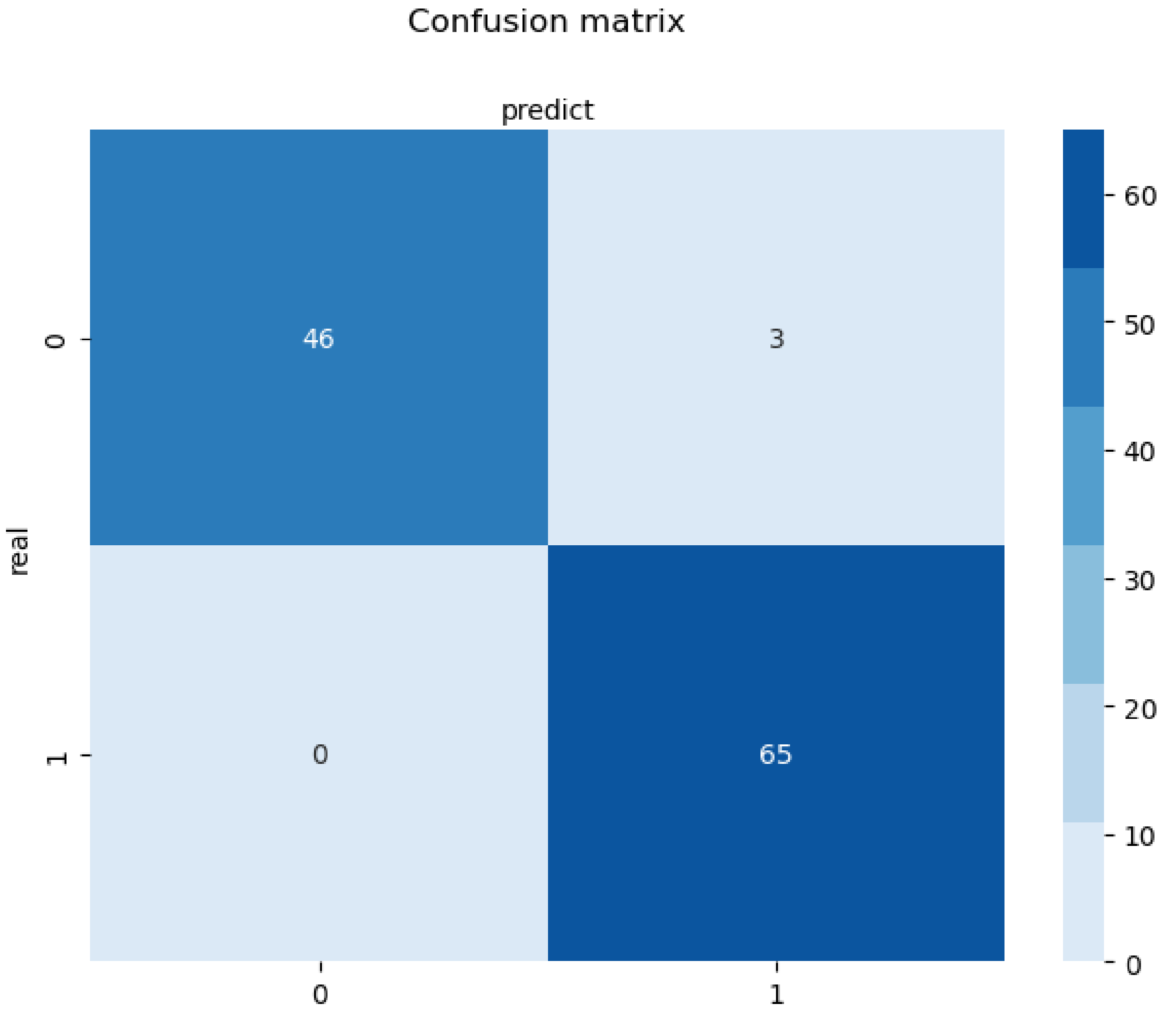
In [112…
```
model=LogisticRegression()
model.fit(X_train,Y_train)
Y_pred= model.predict(X_test)
Y_pred[0:9]
```

Out[112]:
```
array([0, 1, 0, 1, 1, 1, 0, 0, 1])
```

In [113…
```
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: 0.9736842105263158
Precision: 0.9558823529411765
Recall: 1.0
```

In [114…
```
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)
# create heatmap
cm = confusion_matrix(Y_test, Y_pred)
sns.heatmap(pd.DataFrame(cm),cmap=sns.color_palette("Blues"),annot=True,fmt='d')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('real')
plt.xlabel('predict')
plt.show()
```


Confusion matrix
predict
real
0
1
46
3
0
65
0
1
60
50
40
30
20
10
0

In [45]:

```
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.datasets import load_breast_cancer
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix
from sklearn.naive_bayes import GaussianNB

breast = load_breast_cancer()
breast_data = breast.data
breast_input = pd.DataFrame(breast_data)
breast_labels = breast.target
labels = np.reshape(breast_labels,(569,1))
final_breast_data = np.concatenate([breast_data,labels],axis=1)
breast_dataset = pd.DataFrame(final_breast_data)
features = breast.feature_names
features
```

Out[45]:

```
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 [46]:

```
features_labels = np.append(features,’label’)
breast_dataset.columns = features_labels
breast_dataset[’label’].replace(’Benign’,0,inplace=True)
breast_dataset[’label’].replace(’Malignant’,1,inplace=True)
breast_dataset.tail()
```

Out[46]:

| | mean radius | mean texture | mean perimeter | mean area | mean smoothness | mean compactness | mean concavity | mean concave points | mea symmetr |
|---|---|---|---|---|---|---|---|---|---|
| **564** | 21.56 | 22.39 | 142.00 | 1479.0 | 0.11100 | 0.11590 | 0.24390 | 0.13890 | 0.172 |
| **565** | 20.13 | 28.25 | 131.20 | 1261.0 | 0.09780 | 0.10340 | 0.14400 | 0.09791 | 0.175 |
| **566** | 16.60 | 28.08 | 108.30 | 858.1 | 0.08455 | 0.10230 | 0.09251 | 0.05302 | 0.159 |
| **567** | 20.60 | 29.33 | 140.10 | 1265.0 | 0.11780 | 0.27700 | 0.35140 | 0.15200 | 0.239 |
| **568** | 7.76 | 24.54 | 47.92 | 181.0 | 0.05263 | 0.04362 | 0.00000 | 0.00000 | 0.158 |

5 rows × 31 columns

In [47]:

```
from sklearn.preprocessing import StandardScaler


x = breast_dataset.loc[:, features].values
x = StandardScaler().fit_transform(x)
```

In [48]:

```
k=5    #设置降维的占比
pca= PCA(n_components=k)#调用PCA函数，先实例化
pcaCom = pca.fit_transform(x)
pcaCom = pd.DataFrame(pcaCom)
print("主成分的数量： ",pca.n_components_)
X = pcaCom.iloc[:, [0, 1, 2, 3, 4]].values
#X = pcaCom.iloc[:, [0, 1, 2, 3, 4, 5, 6, 7, 8]].values
Y = breast_dataset.iloc[:, 30].values
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size=0.8, test_size=
```

```
主成分的数量：  5
```

In [49]:

```
model = GaussianNB()
model.fit(X_train, Y_train)
Y_pred= model.predict(X_test)
Y_pred[0:9]
```

Out[49]:

```
array([0, 1, 0, 1, 1, 1, 0, 0, 1])
```

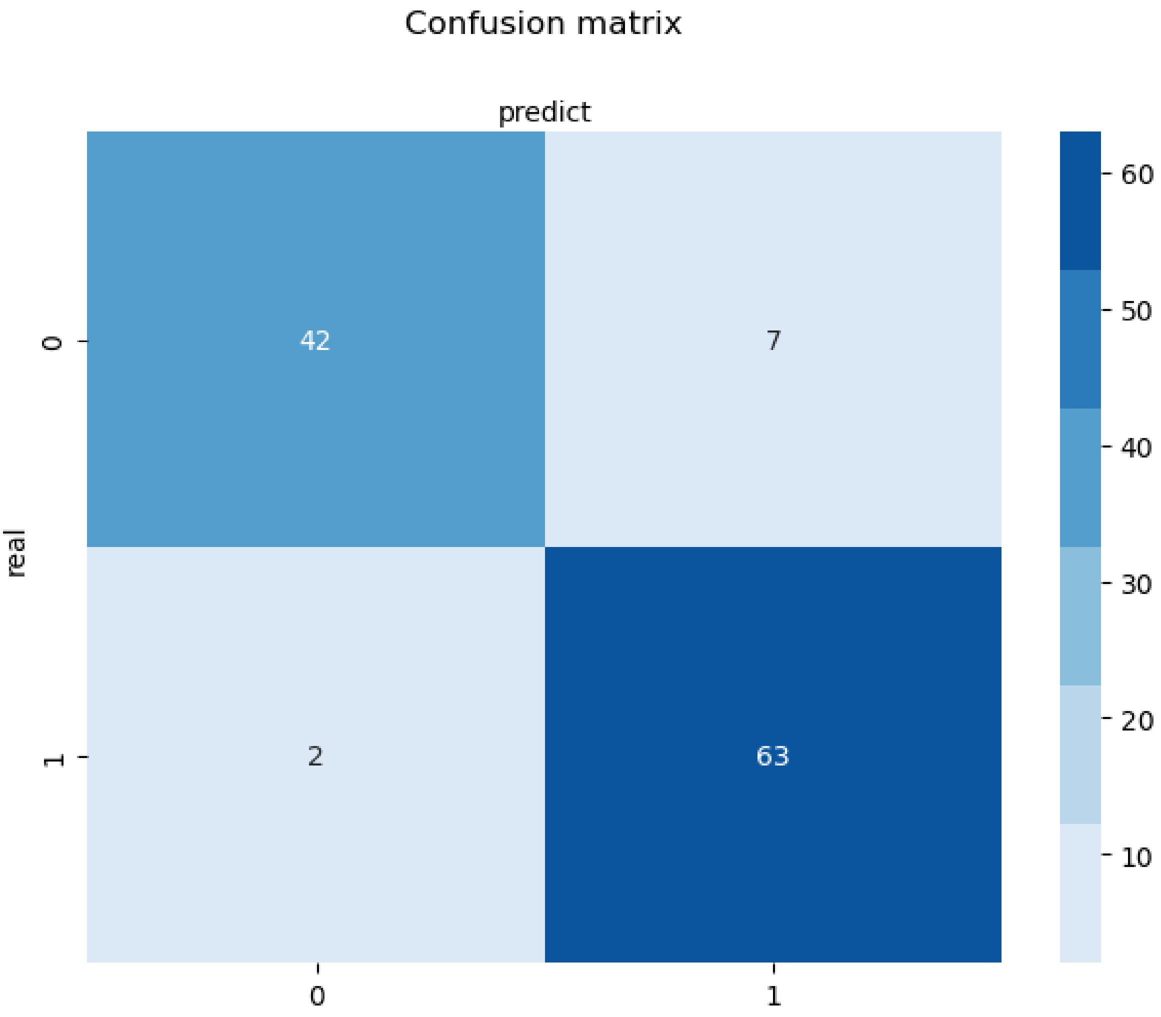
In [50]:

```
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: 0.9210526315789473
Precision: 0.9
Recall: 0.9692307692307692
```

In [51]:

```
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)
# create heatmap
cm = confusion_matrix(Y_test, Y_pred)
sns.heatmap(pd.DataFrame(cm),cmap=sns.color_palette("Blues"),annot=True,fmt='d')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('real')
plt.xlabel('predict')
plt.show()
```


Confusion matrix
predict
real
0
1
42
7
2
63
0
1
60
50
40
30
20
10