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	mean 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

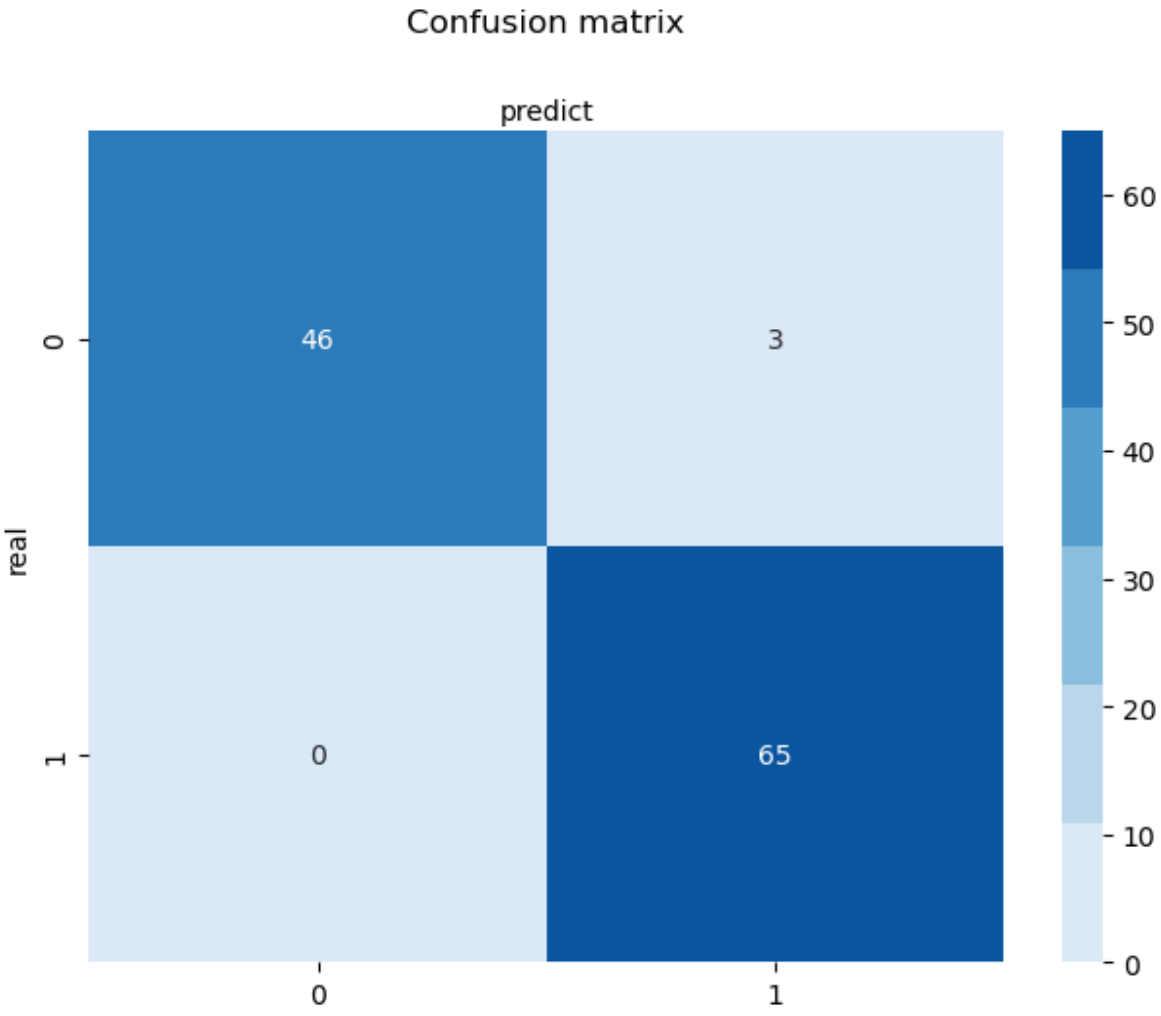
In [112…
```
model=LogisticRegression()
model.fit(X_train,Y_train)
Y_pred= model.predict(X_test)
Y_pred[0:9]
```

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