In [89]:
```
# Import Data
from sklearn.datasets import load_breast_cancer
from sklearn.metrics import classification_report,accuracy_score, confusion_matrix
from sklearn.model_selection import GridSearchCV
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn import svm
import seaborn as sns
import matplotlib.pyplot as plt
```

In [107…
```
def create_confusion_matrix(x_para,y_para):
    cm = confusion_matrix(x_para, y_para)

    fig, ax = plt.subplots(figsize=(4, 4))
    ax.imshow(cm)
    ax.grid(False)
    ax.xaxis.set(ticks=(0, 1), ticklabels=('Predicted 0s', 'Predicted 1s'))
    ax.yaxis.set(ticks=(0, 1), ticklabels=('Actual 0s', 'Actual 1s'))
    ax.set_ylim(1.5, -0.5)
    for i in range(2):
        for j in range(2):
            ax.text(j, i, cm[i, j], ha='center', va='center', color='black')
    plt.show()
```

In [106…
```
def visualisasiData(X_1,y_1):
    X_1=X_1[:,:2]
    x_min, x_max = X_1[:,0].min()-0.5,X_1[:,0].max()+0.5
    y_min, y_max = X_1[:,1].min()-0.5,X_1[:,1].max()+0.5

    plt.scatter(X_1[:,0],X_1[:,1],c=y_1)
    plt.xlabel("mean radius")
    plt.ylabel("mean texture")

    plt.xlim(x_min,x_max)
    plt.ylim(y_min,y_max)
    plt.grid(True)
    plt.show()
```

Load Data

In [92]:
```
## Fetch Data from SKLearn Breact Cancer Collection
data = load_breast_cancer();
```

In [93]:
```
data.data
```

Out[93]:
```
array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
        1.189e-01],
```

```
       [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
        8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
        8.758e-02],
       ...,
       [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
        7.820e-02],
       [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
        1.240e-01],
       [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
        7.039e-02]])
```

In [94]:
```
list(data.target_names)
```

Out[94]:
```
['malignant', 'benign']
```

In [95]:
```
df = pd.DataFrame(data.data, columns=data.feature_names)

df['target'] = data.target

df.head()
```

Out[95]:

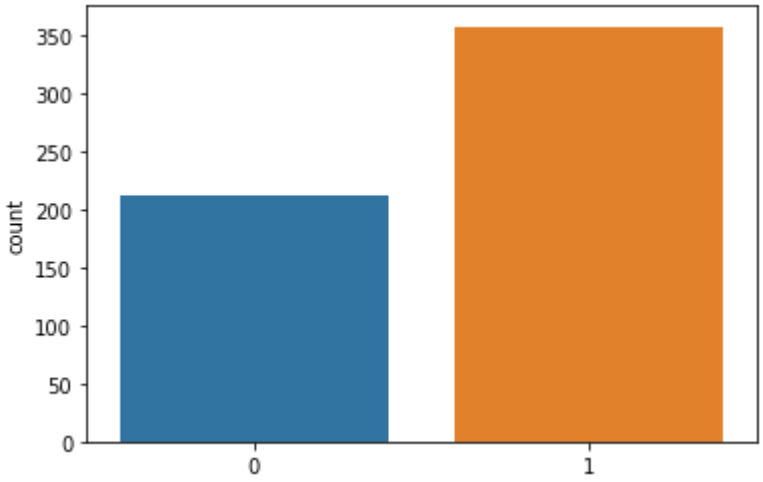
	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	m fra dimen
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05
2	19.69	21.25	130.00	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 [96]:
```
sns.countplot(data.target)
```

```
C:\Users\ronaldo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: P
ass the following variable as a keyword arg: x. From version 0.12, the only valid positi
onal argument will be `data`, and passing other arguments without an explicit keyword wi
ll result in an error or misinterpretation.
  warnings.warn(
```

Out[96]:
```
<AxesSubplot:ylabel='count'>
```



Split Data

In [108…

```
X = data.data
X=X[:,:2]
y = data.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,random_state=1)
```

Klasisikasi dengan KNeighbors

In [109…

```
Kn = KNeighborsClassifier()
Kn.fit(X_train, y_train)
y_pred_kn = Kn.predict(X_test)
```

In [110…

```
parameters = {'leaf_size': [10,20,30,40,50],
                   'n_neighbors':[1,2,3,4,5,6,7,8,9,10,20,30],
                  'p':[1,2]}
Kn_1 = KNeighborsClassifier()
Kn_hyp = GridSearchCV(Kn_1, parameters, cv=2)
Kn_hyp.fit(X_train,y_train)
y_pred_kn_hyp  = Kn_hyp.predict(X_test)
```

Klasifikasi dengan Random Forest

In [100…

```
rf = RandomForestClassifier(max_depth=2, random_state=0)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
```

In [101…

```
parameters = {'bootstrap': [True, False],
 'max_depth': [10, 20, 30],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [100, 200, 400]}
# parameters = {'bootstrap': [True, False],
```

```
#  'max_depth': [10, 20, 30, 40, 50, None],
#  'max_features': ['auto', 'sqrt'],
#  'min_samples_leaf': [1, 2, 4],
#  'min_samples_split': [2, 5, 10],
#  'n_estimators': [200, 400, 600, 800, 1000]}
rf_1 = RandomForestClassifier()

rf_hyp = GridSearchCV(rf_1, parameters, cv=2)
rf_hyp.fit(X_train,y_train)
y_pred_rf_hyp  = rf_hyp.predict(X_test)
```

Klasifikasi dengan Naive Bayes

In [102…

```
gnb = GaussianNB()
gnb.fit(X_train,y_train)
y_pred_nb  = gnb.predict(X_test)
```

In [56]:

```
parameters = {
    'var_smoothing': [1e-2, 1e-3, 1e-4, 1e-5, 1e-6, 1e-7, 1e-8, 1e-9, 1e-10, 1e-11, 1e-
}
gnb_1 = GaussianNB()

gnb_hyp = GridSearchCV(gnb_1, parameters, cv=5)
gnb_hyp.fit(X_train,y_train)
y_pred_nb_hyp  = gnb_hyp.predict(X_test)
```

Klasifikasi dengan SVM

In [58]:

```
svc = svm.SVC()
svc.fit(X_train, y_train)
y_pred_svc  = svc.predict(X_test)
```

In [59]:

```
parameters = {'kernel':('linear', 'rbf'),
              'decision_function_shape':('ovo', 'ovr') ,
              'C':[1, 10]}
svc_1 = svm.SVC()
svc_hyp = GridSearchCV(svc_1, parameters)
svc_hyp.fit(X_train, y_train)
y_pred_svc_hyp = svc_hyp.predict(X_test)
```

Result dengan Kneighbors

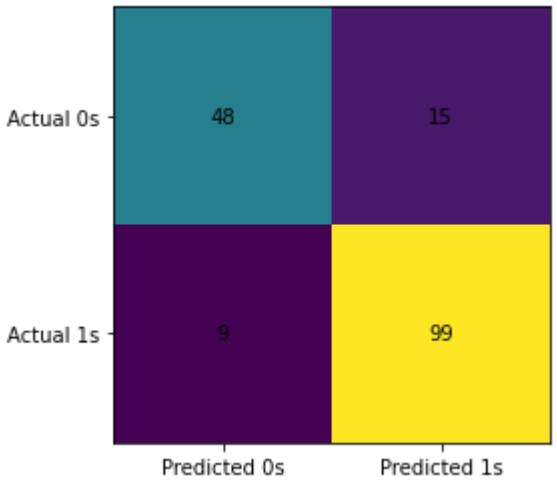
In [61]:

```
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_kn))
print("Classification Report :")
print(classification_report(y_test, y_pred_kn))
```

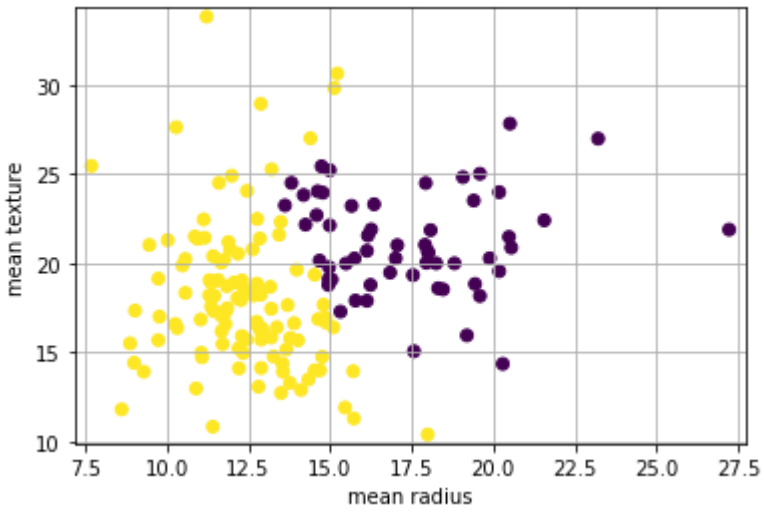
```
Hasil Accuracy : 0.8596491228070176
Classification Report :
              precision    recall  f1-score   support
```

```
              0       0.84      0.76      0.80        63
              1       0.87      0.92      0.89       108

       accuracy                           0.86       171
      macro avg       0.86      0.84      0.85       171
   weighted avg       0.86      0.86      0.86       171
```

In [62]:
```
create_confusion_matrix(y_test,y_pred_kn)
```



In [63]:
```
visualisasiData(X_test,y_pred_kn)
```

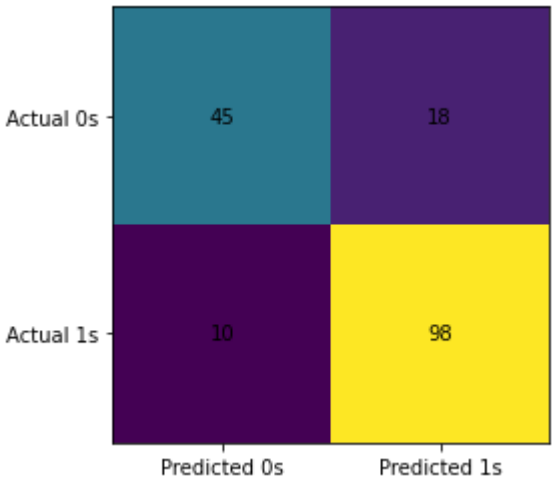


In [64]:
```
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_kn_hyp))
print("Classification Report :")
print(classification_report(y_test, y_pred_kn_hyp))
```

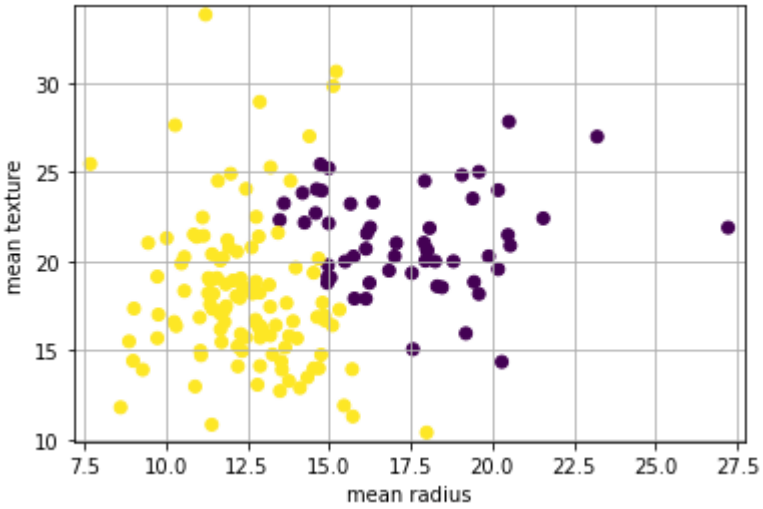
```
Hasil Accuracy : 0.8362573099415205
Classification Report :
              precision    recall  f1-score   support

           0       0.82      0.71      0.76        63
           1       0.84      0.91      0.88       108

    accuracy                           0.84       171
```

```
   macro avg       0.83      0.81      0.82       171
weighted avg       0.84      0.84      0.83       171
```

In [65]:
```
create_confusion_matrix(y_test,y_pred_kn_hyp)
```



In [66]:
```
visualisasiData(X_test,y_pred_kn_hyp)
```



Result dengan Random Forest

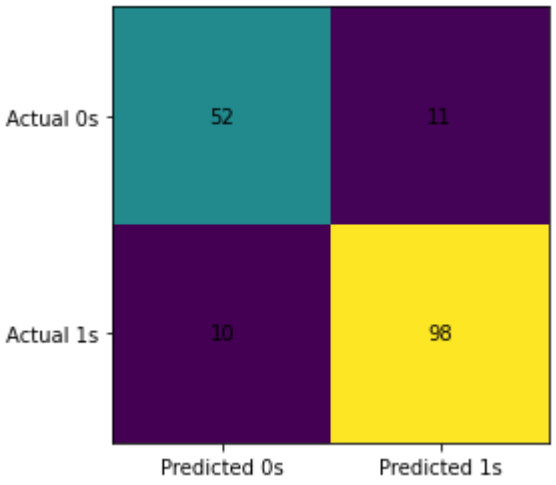
In [113…
```
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_rf))
print("Classification Report :")
print(classification_report(y_test, y_pred_rf))
```

```
Hasil Accuracy : 0.8771929824561403
Classification Report :
              precision    recall  f1-score   support

           0       0.84      0.83      0.83        63
           1       0.90      0.91      0.90       108

    accuracy                           0.88       171
   macro avg       0.87      0.87      0.87       171
```

```
weighted avg       0.88      0.88      0.88       171
```

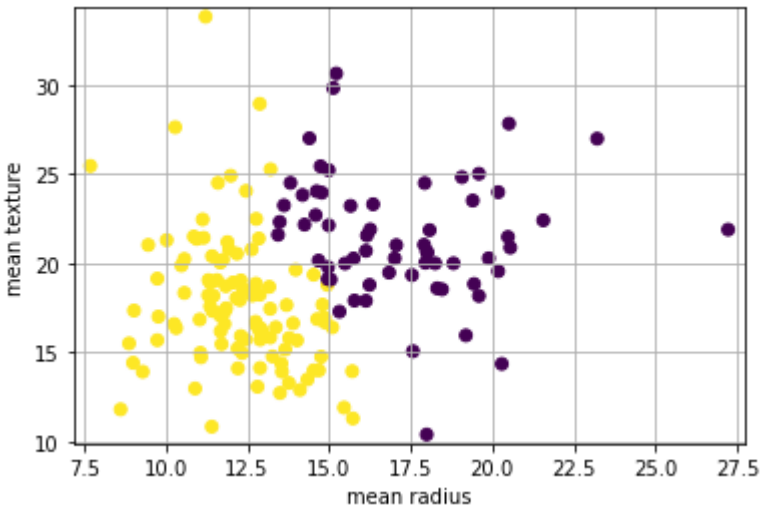
In [114…

```
create_confusion_matrix(y_test,y_pred_rf)
```



In [115…

```
visualisasiData(X_test,y_pred_rf)
```



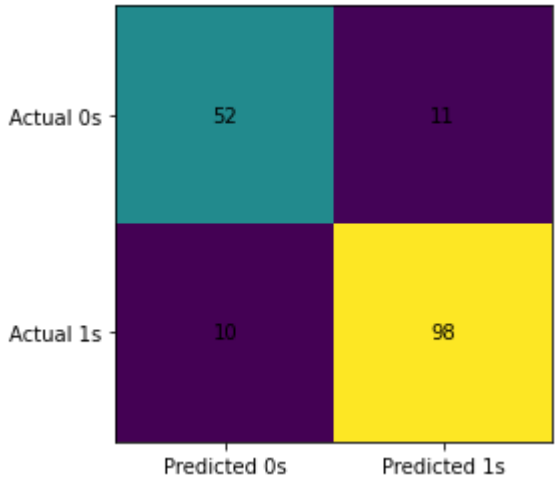
In [128…

```
# Random Forest with GridSearch
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_rf_hyp))
print("Classification Report :")
print(classification_report(y_test, y_pred_rf_hyp))
```

```
Hasil Accuracy : 0.8771929824561403
Classification Report :
              precision    recall  f1-score   support

           0       0.84      0.83      0.83        63
           1       0.90      0.91      0.90       108

    accuracy                           0.88       171
   macro avg       0.87      0.87      0.87       171
weighted avg       0.88      0.88      0.88       171
```

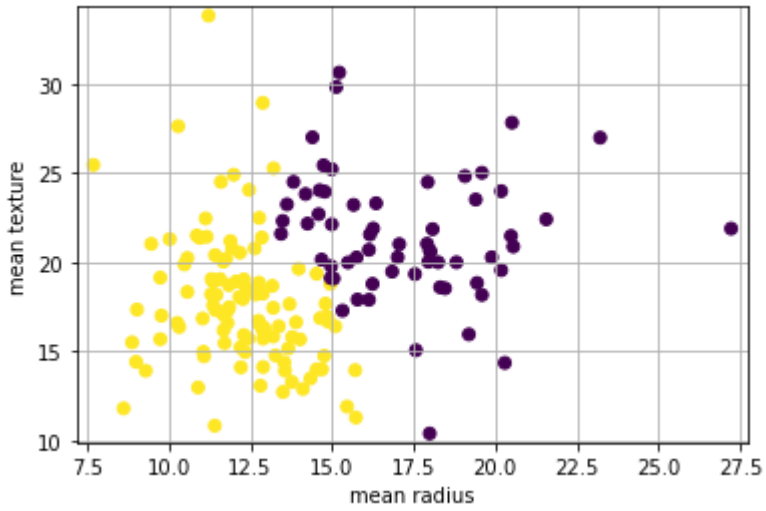
In [127…

```
create_confusion_matrix(y_test,y_pred_rf_hyp)
```



In [129…

```
visualisasiData(X_test,y_pred_rf_hyp)
```



Result dengan Naive Bayes

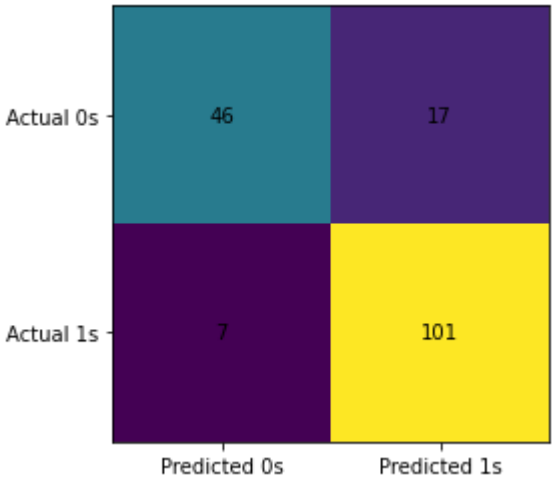
In [116…

```
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_nb))
print("Classification Report :")
print(classification_report(y_test, y_pred_nb))
```

```
Hasil Accuracy : 0.8596491228070176
Classification Report :
              precision    recall  f1-score   support

           0       0.87      0.73      0.79        63
           1       0.86      0.94      0.89       108

    accuracy                           0.86       171
   macro avg       0.86      0.83      0.84       171
weighted avg       0.86      0.86      0.86       171
```

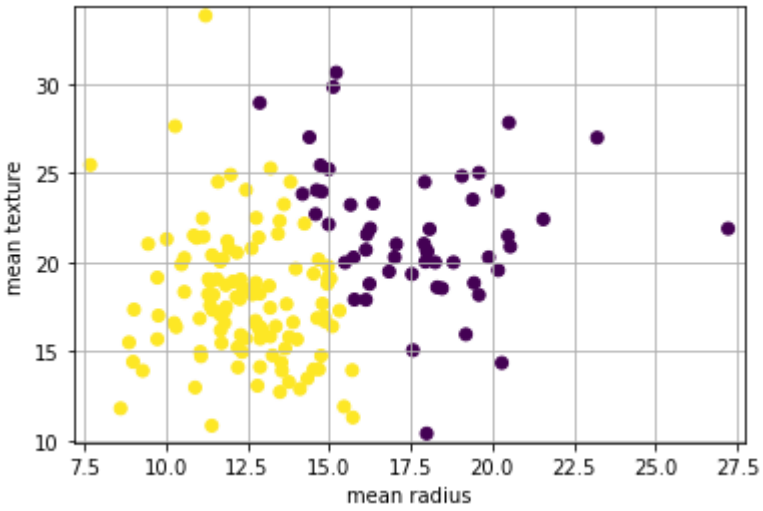
In [117…

```
create_confusion_matrix(y_test,y_pred_nb)
```



In [118…

```
# Visualisasi data
visualisasiData(X_test,y_pred_nb)
```



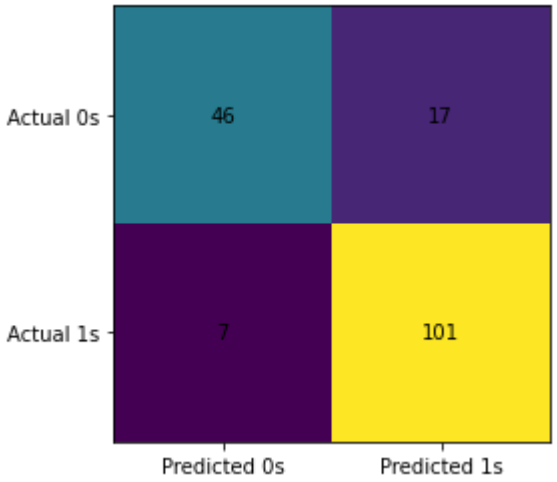
In [75]:

```
# Naive Bayes dengan Grid Search
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_nb_hyp))
print("Classification Report :")
print(classification_report(y_test, y_pred_nb_hyp))
```

```
Hasil Accuracy : 0.8596491228070176
Classification Report :
              precision    recall  f1-score   support

           0       0.87      0.73      0.79        63
           1       0.86      0.94      0.89       108

    accuracy                           0.86       171
   macro avg       0.86      0.83      0.84       171
weighted avg       0.86      0.86      0.86       171
```

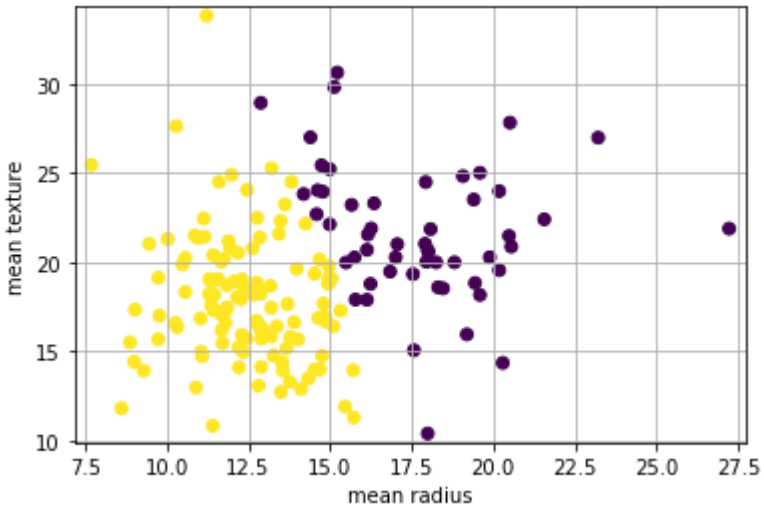
In [119…

```
create_confusion_matrix(y_test,y_pred_nb_hyp)
```



In [120…

```
visualisasiData(X_test,y_pred_nb_hyp)
```



Result SVM

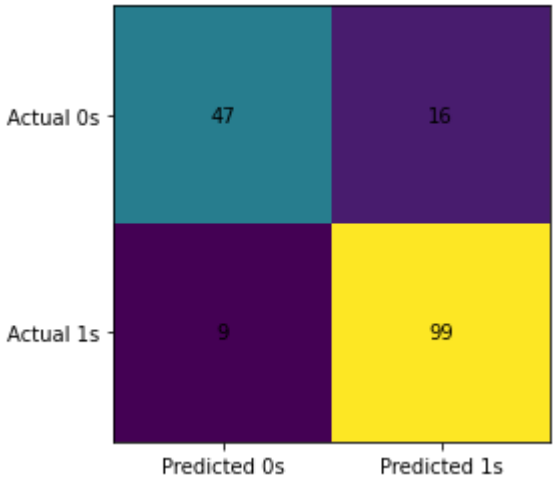
In [121…

```
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_svc))
print("Classification Report :")
print(classification_report(y_test, y_pred_svc))
```

```
Hasil Accuracy : 0.8538011695906432
Classification Report :
              precision    recall  f1-score   support

           0       0.84      0.75      0.79        63
           1       0.86      0.92      0.89       108

    accuracy                           0.85       171
   macro avg       0.85      0.83      0.84       171
weighted avg       0.85      0.85      0.85       171
```

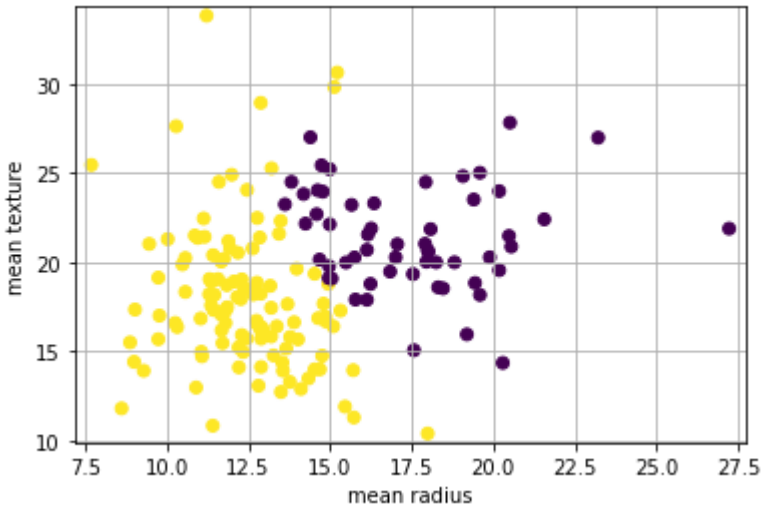
In [122…

```
create_confusion_matrix(y_test,y_pred_svc)
```



In [123…

```
visualisasiData(X_test,y_pred_svc)
```



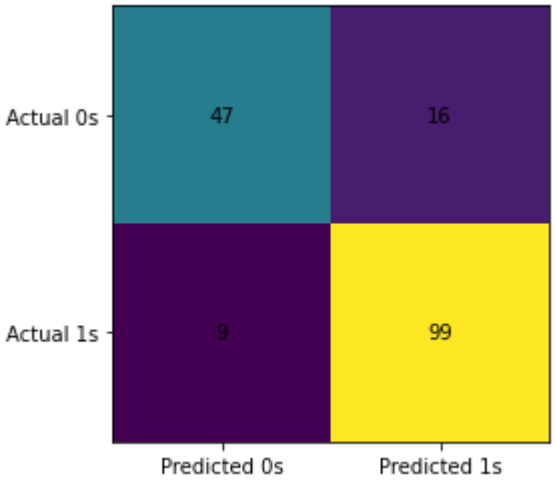
In [124…

```
# SVM dengan Grid Search
print("Hasil Accuracy :", accuracy_score(y_test, y_pred_svc_hyp))
print("Classification Report :")
print(classification_report(y_test, y_pred_svc_hyp))
```

```
Hasil Accuracy : 0.8538011695906432
Classification Report :
              precision    recall  f1-score   support

           0       0.84      0.75      0.79        63
           1       0.86      0.92      0.89       108

    accuracy                           0.85       171
   macro avg       0.85      0.83      0.84       171
weighted avg       0.85      0.85      0.85       171
```

In [125…

```
create_confusion_matrix(y_test,y_pred_svc_hyp)
```



In [126…

```
visualisasiData(X_test,y_pred_svc_hyp)
```