In [ ]:

```
from sklearn import datasets
data = datasets.load_iris(return_X_y=False,as_frame=True)
print(data.data.head())
features_name=data.feature_names
```

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

In [ ]:

```
features=data.data
classes=data.target_names
target=data.target
Iris=features.copy()
Iris['target']=target
Iris.describe()
```

Out[ ]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333	1.000000
std	0.828066	0.435866	1.765298	0.762238	0.819232
min	4.300000	2.000000	1.000000	0.100000	0.000000
25%	5.100000	2.800000	1.600000	0.300000	0.000000
50%	5.800000	3.000000	4.350000	1.300000	1.000000
75%	6.400000	3.300000	5.100000	1.800000	2.000000
max	7.900000	4.400000	6.900000	2.500000	2.000000

In [ ]:

```
classes
```

Out[ ]:

```
array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

In [ ]:

```
features_name
```

Out[ ]:

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

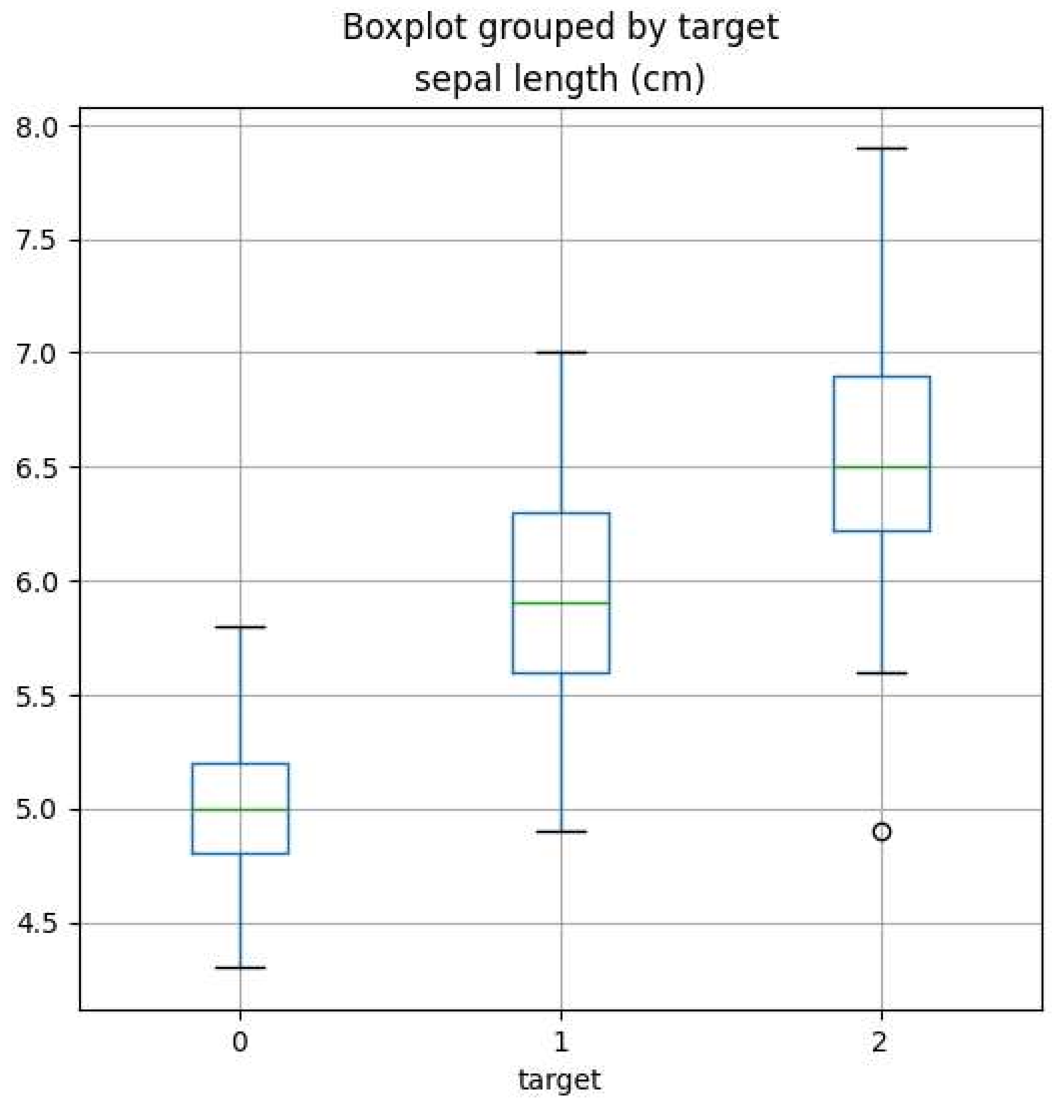
In [ ]:

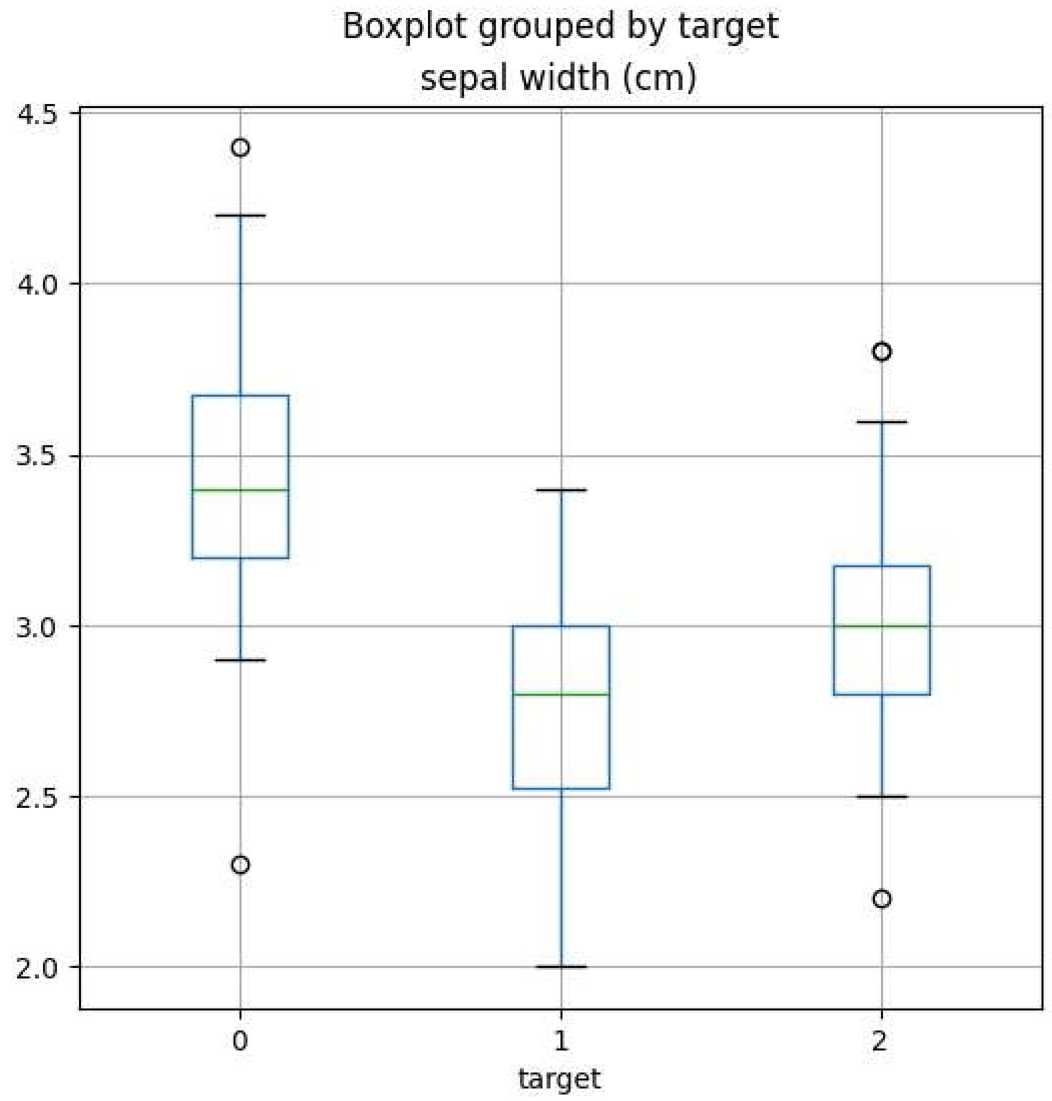
```
Iris.info()
```

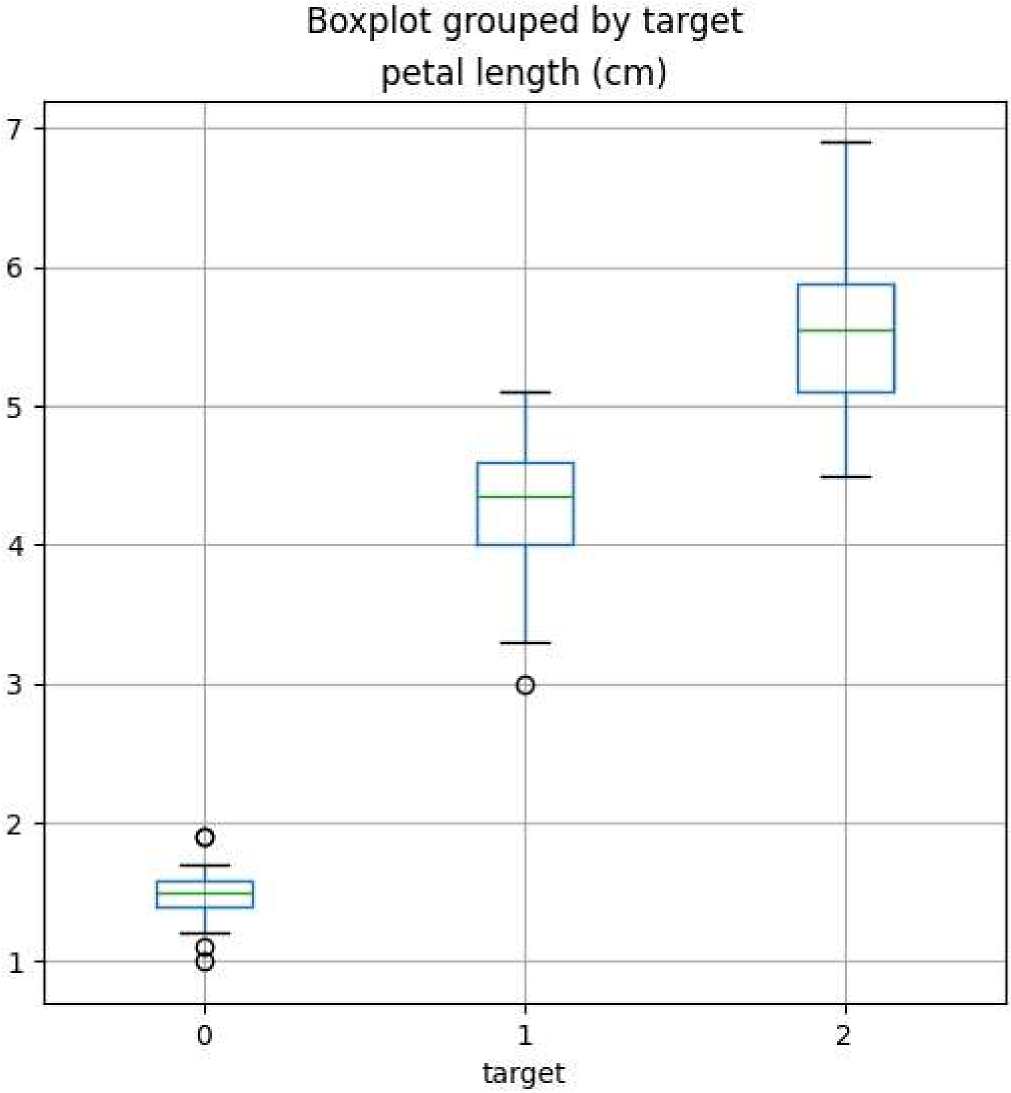
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   target             150 non-null    int32  
dtypes: float64(4), int32(1)
memory usage: 5.4 KB
```

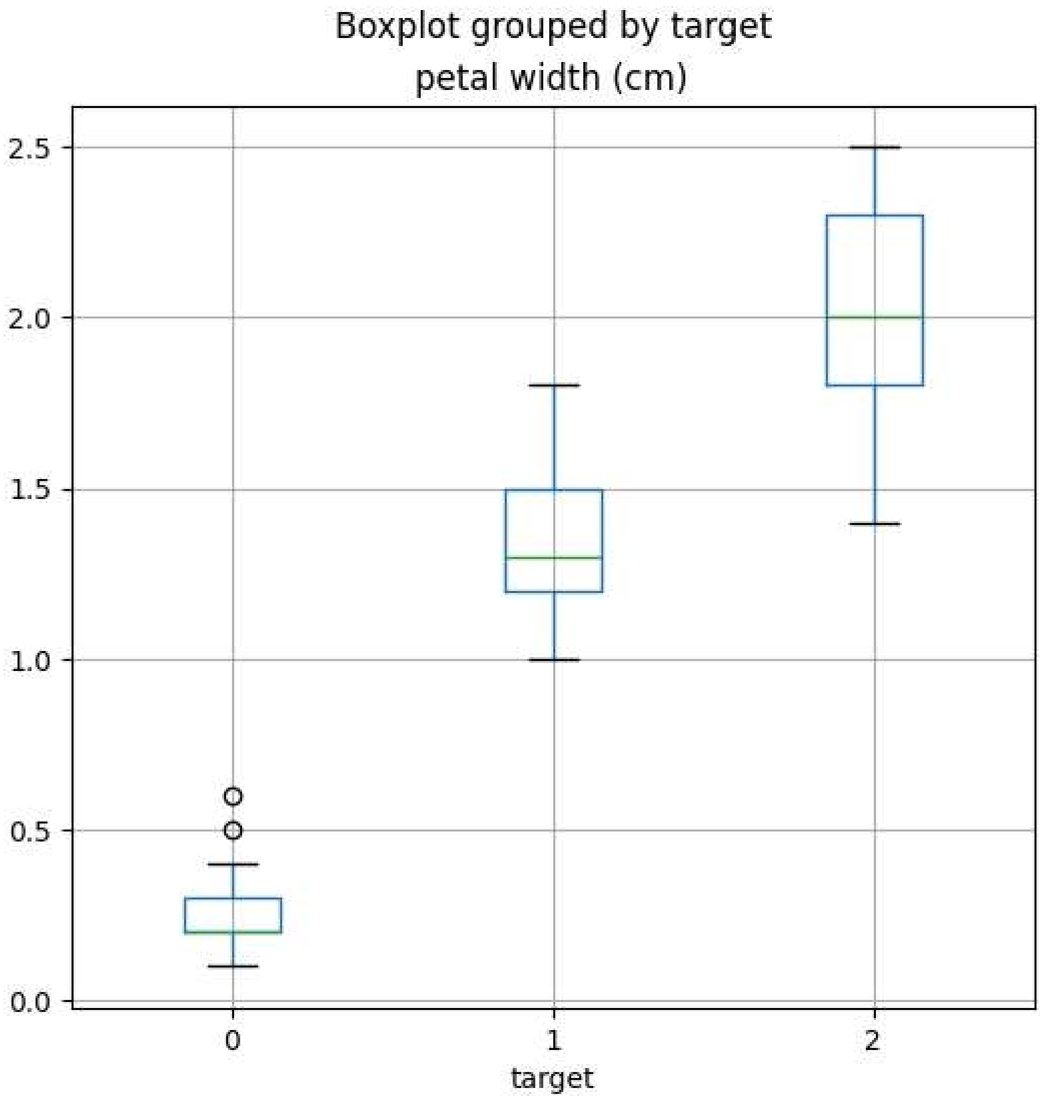
In [ ]:

```
from matplotlib import pyplot as plt
#print(Iris.head())
for col in features_name:
    Iris.boxplot(column=col,by='target', figsize=(6,6))
    plt.title(col)
    plt.show()
```




Boxplot grouped by target
sepal width (cm)
4.5
4.0
3.5
3.0
2.5
2.0
0
1
2
target

Boxplot grouped by target
petal length (cm)
7
6
5
4
3
2
1
0
1
2
target



In [ ]:
```
#check data imbalanced:
Iris.groupby('target').count()
```

Out[ ]:

target	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	50	50	50	50
1	50	50	50	50
2	50	50	50	50

In [ ]:
```
Iris['target'].value_counts()
```

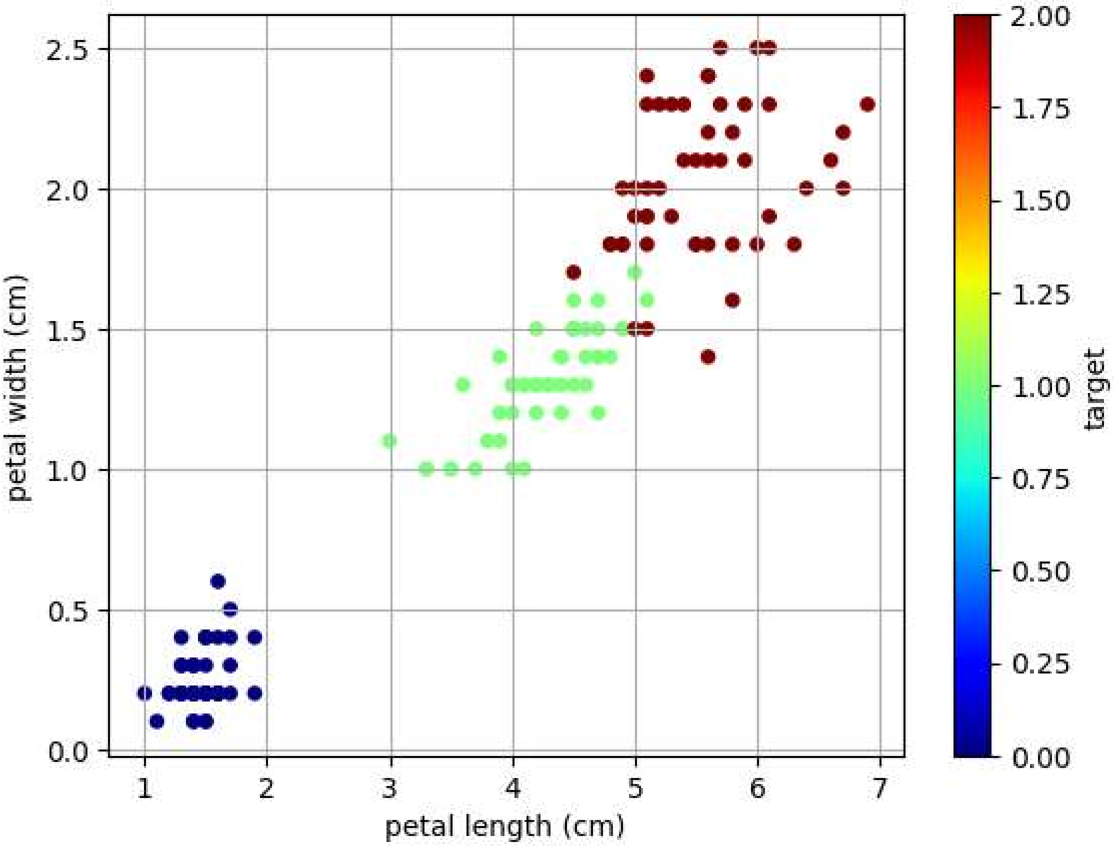
Out[ ]:
```
0    50
1    50
2    50
Name: target, dtype: int64
```

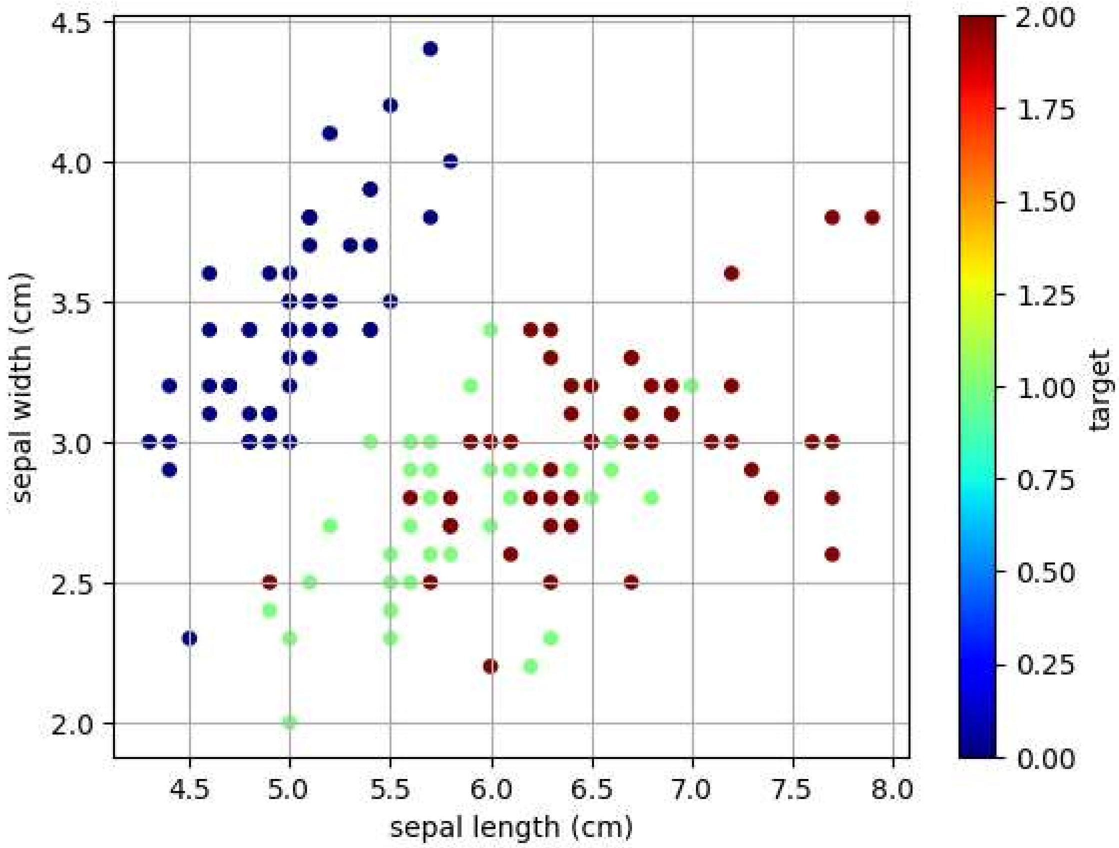
In [ ]:
```
Iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
 4   target             150 non-null    int32  
dtypes: float64(4), int32(1)
memory usage: 5.4 KB
```

In [ ]:

```
Iris.plot(kind='scatter',x='petal length (cm)',y='petal width (cm)',grid=True,c='targe
plt.show()
Iris.plot(kind='scatter',x='sepal length (cm)',y='sepal width (cm)',grid=True,c='targe
plt.show()
```





In [ ]:

```
from sklearn.model_selection import train_test_split
X=features
y=target
Xtrain,Xtest,ytrain,ytest=train_test_split( X,y, test_size=0.2,random_state=42)
Xtrain.shape
```

Out[ ]:

```
(120, 4)
```

In [ ]:

```
Xtest.shape
```

Out[ ]:

```
(30, 4)
```

In [ ]:

```
from sklearn.linear_model import SGDClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC
from sklearn.svm import SVC
from sklearn.ensemble import VotingClassifier
clf={
    'SGDClassifier':SGDClassifier(random_state=42,alpha=.01,max_iter=2000,tol=.0001,lo
    'KNeighborsClassifier':KNeighborsClassifier(n_neighbors=3),
    'DecisionTreeClassifier':DecisionTreeClassifier(random_state=42,max_features=3),
    'RandomForestClassifier':RandomForestClassifier(random_state=42,max_features=3),
    'LinearSVC':LinearSVC(random_state=42,max_iter=2000,multi_class="crammer_singer",C
    'SVC':  SVC(C=3,max_iter=2000),
    #'VotingClassifier':VotingClassifier()
}
```

In [ ]:

```
from sklearn.model_selection import cross_val_score
```

```
import numpy as np
results=[]
for key in clf.keys():
    score=cross_val_score(clf[key], Xtrain, ytrain, scoring="accuracy", cv=3)
    results.append((key,score.mean()*100))
print('models scores:',results)
best_model_idx=np.array(results)[:,1].argmax()
print('best model:',results[best_model_idx][0],results[best_model_idx][1].round(1))
```

```
models scores: [('SGDClassifier', 94.16666666666667), ('KNeighborsClassifier', 95.0),
('DecisionTreeClassifier', 93.33333333333333), ('RandomForestClassifier', 95.0), ('Li
nearSVC', 97.5), ('SVC', 95.0)]
best model: LinearSVC 97.5
```

```
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
```

In [ ]:

```
score=cross_val_score(clf['LinearSVC'], Xtest, ytest, scoring="accuracy", cv=3)
score.mean()
```

```
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
```

Out[ ]:

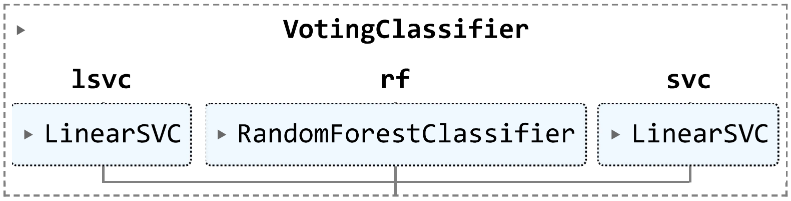
```
1.0
```

In [ ]:

```
eclf = VotingClassifier([("lsvc", clf['LinearSVC']), ("rf", clf['RandomForestClassifie
     ("svc", clf['LinearSVC'])])
eclf.fit(Xtrain, ytrain)
```

```
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
```

Out[ ]:



In [ ]:

```
eclf.get_params()
```

```
Out[ ]: {'estimators': [('lsvc',
           LinearSVC(C=3, max_iter=2000, multi_class='crammer_singer', random_state=42)),
          ('rf', RandomForestClassifier(max_features=3, random_state=42)),
          ('svc',
           LinearSVC(C=3, max_iter=2000, multi_class='crammer_singer', random_state=42))],
         'flatten_transform': True,
         'n_jobs': None,
         'verbose': False,
         'voting': 'hard',
         'weights': None,
         'lsvc': LinearSVC(C=3, max_iter=2000, multi_class='crammer_singer', random_state=4
        2),
         'rf': RandomForestClassifier(max_features=3, random_state=42),
         'svc': LinearSVC(C=3, max_iter=2000, multi_class='crammer_singer', random_state=42),
         'lsvc__C': 3,
         'lsvc__class_weight': None,
         'lsvc__dual': True,
         'lsvc__fit_intercept': True,
         'lsvc__intercept_scaling': 1,
         'lsvc__loss': 'squared_hinge',
         'lsvc__max_iter': 2000,
         'lsvc__multi_class': 'crammer_singer',
         'lsvc__penalty': 'l2',
         'lsvc__random_state': 42,
         'lsvc__tol': 0.0001,
         'lsvc__verbose': 0,
         'rf__bootstrap': True,
         'rf__ccp_alpha': 0.0,
         'rf__class_weight': None,
         'rf__criterion': 'gini',
         'rf__max_depth': None,
         'rf__max_features': 3,
         'rf__max_leaf_nodes': None,
         'rf__max_samples': None,
         'rf__min_impurity_decrease': 0.0,
         'rf__min_samples_leaf': 1,
         'rf__min_samples_split': 2,
         'rf__min_weight_fraction_leaf': 0.0,
         'rf__n_estimators': 100,
         'rf__n_jobs': None,
         'rf__oob_score': False,
         'rf__random_state': 42,
         'rf__verbose': 0,
         'rf__warm_start': False,
         'svc__C': 3,
         'svc__class_weight': None,
         'svc__dual': True,
         'svc__fit_intercept': True,
         'svc__intercept_scaling': 1,
         'svc__loss': 'squared_hinge',
         'svc__max_iter': 2000,
         'svc__multi_class': 'crammer_singer',
         'svc__penalty': 'l2',
         'svc__random_state': 42,
         'svc__tol': 0.0001,
         'svc__verbose': 0}
```

```
In [ ]: score=cross_val_score(eclf,Xtrain,ytrain,scoring='accuracy',cv=3)
        accuracy=score.mean()*100
        print('accuracy percentage:',accuracy)
```

```
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
accuracy percentage: 97.5
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
```

In [ ]:

```python
score=cross_val_score(eclf,Xtest,ytest,scoring='accuracy',cv=3)
accuracy=score.mean()*100
print('accuracy percentage:',accuracy)
```

```
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
accuracy percentage: 100.0
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
```

In [ ]:

```python
#train the model in thewhole data set and Save the final Model:
import joblib
```

```
final_model=eclf.fit(features,target)

joblib.dump(final_model,'clf_final_model.pkl')
```

```
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
c:\Users\user\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\svm\_
base.py:1225: ConvergenceWarning: Liblinear failed to converge, increase the number o
f iterations.
  warnings.warn(
```

Out[ ]:

```
['clf_final_model.pkl']
```

In [ ]:

```
final_model=joblib.load('clf_final_model.pkl')

new_data=features.iloc[:5]
predictions=final_model.predict(new_data)
predictions
```

Out[ ]:

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

In [ ]:

```
target.iloc[:5]
```

Out[ ]:

```
0    0
1    0
2    0
3    0
4    0
Name: target, dtype: int32
```

In [ ]:

```
#perfect!! :)
```