In [3]:
```
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import load_wine
```

In [5]:
```
wine=load_wine()
X=wine.data
y=wine.target
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=1)
```

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

Out[6]:
```
(178, 13)
```

In [7]:
```
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier()
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
Train Score: 0.9849624060150376
Test Score: 0.9777777777777777
```

In [8]:
```
from sklearn.feature_selection import f_classif
```

In [10]:
```
fvalue,pvalue=f_classif(X,y)
```

In [11]:
```
fvalue
```

Out[11]:
```
array([135.07762424,  36.94342496,  13.3129012 ,  35.77163741,
        12.42958434,  93.73300962, 233.92587268,  27.57541715,
        30.27138317, 120.66401844, 101.31679539, 189.97232058,
       207.9203739 ])
```

In [12]:
```
print(wine.feature_names)
```

```
['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'total_phenols', 'fla
vanoids', 'nonflavanoid_phenols', 'proanthocyanins', 'color_intensity', 'hue', 'od280/od
315_of_diluted_wines', 'proline']
```

In [22]:
```
X1=X[:,[0,5,6,9,10,11,12]]
```

In [23]:
```
X1.shape
```

Out[23]:
```
(178, 7)
```

In [24]:
```
X_train,X_test,y_train,y_test=train_test_split(X1,y,random_state=1)
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier()
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
```

```
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
Train Score: 0.9849624060150376
Test Score: 0.9777777777777777
```

In [25]:
```
from sklearn.feature_selection import SelectKBest
```

In [38]:
```
skb=SelectKBest(score_func=f_classif,k=12)
X_new=skb.fit_transform(X,y)

print(X_new.shape)
X_train,X_test,y_train,y_test=train_test_split(X_new,y,random_state=1)
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier()
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
(178, 12)
Train Score: 0.9774436090225563
Test Score: 0.9777777777777777
```

In [28]:
```
skb.scores_
```

Out[28]:
```
array([135.07762424,  36.94342496,  13.3129012 ,  35.77163741,
        12.42958434,  93.73300962, 233.92587268,  27.57541715,
        30.27138317, 120.66401844, 101.31679539, 189.97232058,
       207.9203739 ])
```

In [39]:
```
from sklearn.datasets import load_breast_cancer
```

In [40]:
```
cancer=load_breast_cancer()
```

In [41]:
```
print(cancer.DESCR)
```

```
.. _breast_cancer_dataset:

Breast cancer wisconsin (diagnostic) dataset
--------------------------------------------

**Data Set Characteristics:**

    :Number of Instances: 569

    :Number of Attributes: 30 numeric, predictive attributes and the class

    :Attribute Information:
        - radius (mean of distances from center to points on the perimeter)
        - texture (standard deviation of gray-scale values)
        - perimeter
        - area
        - smoothness (local variation in radius lengths)
        - compactness (perimeter^2 / area - 1.0)
        - concavity (severity of concave portions of the contour)
        - concave points (number of concave portions of the contour)
        - symmetry
        - fractal dimension ("coastline approximation" - 1)

        The mean, standard error, and "worst" or largest (mean of the three
        worst/largest values) of these features were computed for each image,
```

resulting in 30 features. For instance, field 0 is Mean Radius, field 10 is Radius SE, field 20 is Worst Radius.

- class:
  - WDBC-Malignant
  - WDBC-Benign

:Summary Statistics:

	Min	Max
radius (mean):	6.981	28.11
texture (mean):	9.71	39.28
perimeter (mean):	43.79	188.5
area (mean):	143.5	2501.0
smoothness (mean):	0.053	0.163
compactness (mean):	0.019	0.345
concavity (mean):	0.0	0.427
concave points (mean):	0.0	0.201
symmetry (mean):	0.106	0.304
fractal dimension (mean):	0.05	0.097
radius (standard error):	0.112	2.873
texture (standard error):	0.36	4.885
perimeter (standard error):	0.757	21.98
area (standard error):	6.802	542.2
smoothness (standard error):	0.002	0.031
compactness (standard error):	0.002	0.135
concavity (standard error):	0.0	0.396
concave points (standard error):	0.0	0.053
symmetry (standard error):	0.008	0.079
fractal dimension (standard error):	0.001	0.03
radius (worst):	7.93	36.04
texture (worst):	12.02	49.54
perimeter (worst):	50.41	251.2
area (worst):	185.2	4254.0
smoothness (worst):	0.071	0.223
compactness (worst):	0.027	1.058
concavity (worst):	0.0	1.252
concave points (worst):	0.0	0.291
symmetry (worst):	0.156	0.664
fractal dimension (worst):	0.055	0.208

:Missing Attribute Values: None

:Class Distribution: 212 - Malignant, 357 - Benign

:Creator: Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian

:Donor: Nick Street

:Date: November, 1995

This is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.
https://goo.gl/U2Uwz2

Features are computed from a digitized image of a fine needle
aspirate (FNA) of a breast mass. They describe
characteristics of the cell nuclei present in the image.

Separating plane described above was obtained using
Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree
Construction Via Linear Programming." Proceedings of the 4th
Midwest Artificial Intelligence and Cognitive Science Society,
pp. 97-101, 1992], a classification method which uses linear

```
programming to construct a decision tree.  Relevant features
were selected using an exhaustive search in the space of 1-4
features and 1-3 separating planes.

The actual linear program used to obtain the separating plane
in the 3-dimensional space is that described in:
[K. P. Bennett and O. L. Mangasarian: "Robust Linear
Programming Discrimination of Two Linearly Inseparable Sets",
Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server:

ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

.. topic:: References

   - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction
     for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on
     Electronic Imaging: Science and Technology, volume 1905, pages 861-870,
     San Jose, CA, 1993.
   - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and
     prognosis via linear programming. Operations Research, 43(4), pages 570-577,
     July-August 1995.
   - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques
     to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (1994)
     163-171.
```

In [48]:
```
cancer=load_breast_cancer()
X=cancer.data
y=cancer.target

X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=1)
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier()
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
Train Score: 0.9812206572769953
Test Score: 0.951048951048951
```

In [93]:
```
cancer=load_breast_cancer()
X=cancer.data
y=cancer.target

skb=SelectKBest(score_func=f_classif,k=27)
X_new=skb.fit_transform(X,y)

print(X_new.shape)
X_train,X_test,y_train,y_test=train_test_split(X_new,y,random_state=1)
sc=StandardScaler()
X_train_new=sc.fit_transform(X_train)
X_test_new=sc.transform(X_test)
model=KNeighborsClassifier(n_neighbors=6)
model.fit(X_train_new,y_train)
pred_train=model.predict(X_train_new)
pred_test=model.predict(X_test_new)
print("Train Score:",accuracy_score(y_train,pred_train))
print("Test Score:",accuracy_score(y_test,pred_test))
```

```
(569, 27)
```

```
Train Score: 0.9788732394366197
Test Score: 0.965034965034965
```

In [94]:
```
from sklearn.model_selection import cross_val_score
```

In [106…
```
cancer=load_breast_cancer()
X=cancer.data
y=cancer.target

skb=SelectKBest(score_func=f_classif,k=25)
X_new=skb.fit_transform(X,y)

sc=StandardScaler()
X_new=sc.fit_transform(X_new)

model=KNeighborsClassifier()
test_scores=cross_val_score(model,X_new,y,cv=5)
test_scores.mean()
```

Out[106]:
```
0.9754075454122031
```

In [99]:
```
cancer=load_breast_cancer()
X=cancer.data
y=cancer.target

sc=StandardScaler()
X_new=sc.fit_transform(X)

model=KNeighborsClassifier()
test_scores=cross_val_score(model,X_new,y,cv=5)
test_scores.mean()
```

Out[99]:
```
0.9648501785437045
```

In [ ]:
```

```