In [1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
#from sklearn.preprocessing import PolynomialFeatures
import sklearn.linear_model as lm
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.read_csv("mushrooms.csv")
df
```

Out[2]:

	class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	...	stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	ring-number	ring-type	spore-print-color	population	habitat
0	p	x	s	n	t	p	f	c	n	k	...	s	w	w	p	w	o	p	k	s	u
1	e	x	s	y	t	a	f	c	b	k	...	s	w	w	p	w	o	p	n	n	g
2	e	b	s	w	t	l	f	c	b	n	...	s	w	w	p	w	o	p	n	n	m
3	p	x	y	w	t	p	f	c	n	n	...	s	w	w	p	w	o	p	k	s	u
4	e	x	s	g	f	n	f	w	b	k	...	s	w	w	p	w	o	e	n	a	g
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8119	e	k	s	n	f	n	a	c	b	y	...	s	o	o	p	o	o	p	b	c	l
8120	e	x	s	n	f	n	a	c	b	y	...	s	o	o	p	n	o	p	b	v	l
8121	e	f	s	n	f	n	a	c	b	n	...	s	o	o	p	o	o	p	b	c	l
8122	p	k	y	n	f	y	f	c	n	b	...	k	w	w	p	w	o	e	w	v	l
8123	e	x	s	n	f	n	a	c	b	y	...	s	o	o	p	o	o	p	o	c	l

8124 rows × 23 columns

In [3]:
```
from sklearn.preprocessing import LabelEncoder
df = df.apply(LabelEncoder().fit_transform)
xx = df.iloc[:,1:]
yy = df.iloc[:,0]
df
```

Out[3]:

	class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	...	stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	ring-number	ring-type	spore-print-color	population	habitat
0	1	5	2	4	1	6	1	0	1	4	...	2	7	7	0	2	1	4	2	3	5
1	0	5	2	9	1	0	1	0	0	4	...	2	7	7	0	2	1	4	3	2	1
2	0	0	2	8	1	3	1	0	0	5	...	2	7	7	0	2	1	4	3	2	3
3	1	5	3	8	1	6	1	0	1	5	...	2	7	7	0	2	1	4	2	3	5
4	0	5	2	3	0	5	1	1	0	4	...	2	7	7	0	2	1	0	3	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8119	0	3	2	4	0	5	0	0	0	11	...	2	5	5	0	1	1	4	0	1	2
8120	0	5	2	4	0	5	0	0	0	11	...	2	5	5	0	0	1	4	0	4	2
8121	0	2	2	4	0	5	0	0	0	5	...	2	5	5	0	1	1	4	0	1	2
8122	1	3	3	4	0	8	1	0	1	0	...	1	7	7	0	2	1	0	7	4	2
8123	0	5	2	4	0	5	0	0	0	11	...	2	5	5	0	1	1	4	4	1	2

8124 rows × 23 columns

In [50]:
```
df = pd.read_csv("iris.csv")
df= df.iloc[:,:4]
xx1 = np.c_[df]
xx1
```

Out[50]:
```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
```

In [5]:
```
a = df.iloc[:,:2]
b = df.iloc[:,4:6]
df1 = pd.concat([a,b],axis=1)
df1
```

Out[5]:

	Color	Eye_color	Moustache	Tail
0	black	black	No	No
1	dark_brown	brown	No	No
2	light_brown	brown	Yes	No
3	light_brown	blue	No	No
4	light_brown	blue	No	No
...	...	...	...	...
195	brown	gray	Yes	Yes
196	white	yellow	Yes	Yes
197	white	black	Yes	Yes
198	brown	green	Yes	Yes
199	brown_white	blue	Yes	Yes

200 rows × 4 columns

In [6]:
```
df2 = pd.concat([df.iloc[:,2:4],df.iloc[:,6]],axis=1)
df2
```

Out[6]:

	Height	Legs	Weight
0	5.14	2	100.000000
1	6.80	2	64.400000
2	5.00	2	64.800000
3	5.90	2	78.800000
4	6.56	2	73.200000
...	...	...	...
195	1.14	4	2.304511
196	1.39	4	5.687970
197	0.53	4	6.364662
198	1.03	4	6.590226
199	0.83	4	0.000000

200 rows × 3 columns

In [7]:
```
x1 = df1.apply(LabelEncoder().fit_transform)
np.c_[x1]
```

Out[7]:
```
array([[0, 0, 0, 0],
       [3, 2, 0, 0],
       [5, 2, 1, 0],
       [5, 1, 0, 0],
       [5, 1, 0, 0],
       [1, 2, 0, 0],
       [0, 1, 1, 0],
       [1, 3, 0, 0],
       [0, 0, 0, 0],
       [3, 2, 0, 0],
       [5, 3, 0, 0],
       [6, 0, 1, 0],
       [3, 0, 1, 0],
       [6, 2, 0, 0],
       [6, 1, 1, 0],
       [3, 3, 1, 0],
       [0, 2, 0, 0],
       [0, 2, 0, 0],
       [3, 2, 1, 0],
```

In [8]:
```
import sklearn.preprocessing as sp
x2 = sp.minmax_scale(df2)
x2
```

Out[8]:
```
array([[0.71604938, 0.        , 1.        ],
       [0.97222222, 0.        , 0.644     ],
       [0.69444444, 0.        , 0.648     ],
       [0.83333333, 0.        , 0.788     ],
       [0.93518519, 0.        , 0.732     ],
       [0.7808642 , 0.        , 0.652     ],
       [0.84259259, 0.        , 0.664     ],
       [0.86111111, 0.        , 0.548     ],
       [0.78395062, 0.        , 0.476     ],
       [0.85802469, 0.        , 0.608     ],
       [0.77160494, 0.        , 0.568     ],
       [0.95679012, 0.        , 0.4       ],
       [0.9691358 , 0.        , 0.62      ],
       [0.7962963 , 0.        , 0.704     ],
       [0.80864198, 0.        , 0.676     ],
       [0.98765432, 0.        , 0.404     ],
       [0.73148148, 0.        , 0.432     ],
       [0.91975309, 0.        , 0.668     ],
       [0.72530864, 0.        , 0.78      ],
```

In [9]:
```
import sklearn.preprocessing as sp
x = np.concatenate([x1,x2],axis = 1)   #Features
x=sp.minmax_scale(x)
y = np.c_[df.iloc[:,7]]
y = np.c_[LabelEncoder().fit_transform(np.c_[y])]   #Labels
x
```

```
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[9]:
```
array([[0.        , 0.        , 0.        , ..., 0.71604938, 0.        ,
        1.        ],
       [0.5       , 0.4       , 0.        , ..., 0.97222222, 0.        ,
        0.644     ],
       [0.83333333, 0.4       , 1.        , ..., 0.69444444, 0.        ,
        0.648     ],
       ...,
       [1.        , 0.        , 1.        , ..., 0.00462963, 1.        ,
        0.06364662],
       [0.16666667, 0.8       , 1.        , ..., 0.08179012, 1.        ,
        0.06590226],
       [0.33333333, 0.2       , 1.        , ..., 0.05092593, 1.        ,
        0.        ]])
```

In [10]:
```
pd.DataFrame(x)
```

Out[10]:

	0	1	2	3	4	5	6
0	0.000000	0.0	0.0	0.0	0.716049	0.0	1.000000
1	0.500000	0.4	0.0	0.0	0.972222	0.0	0.644000
2	0.833333	0.4	1.0	0.0	0.694444	0.0	0.648000
3	0.833333	0.2	0.0	0.0	0.833333	0.0	0.788000
4	0.833333	0.2	0.0	0.0	0.935185	0.0	0.732000
...	...	...	...	...	...	...	...
195	0.166667	0.6	1.0	1.0	0.098765	1.0	0.023045
196	1.000000	1.0	1.0	1.0	0.137346	1.0	0.056880
197	1.000000	0.0	1.0	1.0	0.004630	1.0	0.063647
198	0.166667	0.8	1.0	1.0	0.081790	1.0	0.065902
199	0.333333	0.2	1.0	1.0	0.050926	1.0	0.000000

200 rows × 7 columns

In [11]:
```
from sklearn.model_selection import train_test_split
train_x,test_x,train_y,test_y = train_test_split(x,y,test_size=0.33)
```

In [77]:
```
from sklearn.utils import all_estimators
import sklearn.metrics as mt

estimators = all_estimators(type_filter='classifier')

all_clfs = []
for name, get_model in estimators:
    try:
        model = get_model()
        model.fit(train_x,np.ravel(train_y))
        pred_y=model.predict(test_x)
#         print(sm.classification_report(test_y,pred_y))
        a = mt.accuracy_score(test_y,pred_y)
        b = mt.recall_score(test_y,pred_y)
        c = mt.precision_score(test_y,pred_y)
        d = mt.f1_score(test_y,pred_y)
        print('Working')

    except Exception as e:
        print(name)
        continue
```

```
Working
Working
Working
Working
CategoricalNB
ClassifierChain
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working

C:\Users\pc\anaconda3\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:614: ConvergenceWarning: Stochastic Optimizer: Maximum itera
tions (200) reached and the optimization hasn't converged yet.
  warnings.warn(
C:\Users\pc\anaconda3\lib\site-packages\sklearn\discriminant_analysis.py:808: UserWarning: Variables are collinear
  warnings.warn("Variables are collinear")
C:\Users\pc\anaconda3\lib\site-packages\sklearn\discriminant_analysis.py:833: RuntimeWarning: divide by zero encountered in power
  X2 = np.dot(Xm, R * (S ** (-0.5)))
C:\Users\pc\anaconda3\lib\site-packages\sklearn\discriminant_analysis.py:833: RuntimeWarning: invalid value encountered in multiply
  X2 = np.dot(Xm, R * (S ** (-0.5)))
C:\Users\pc\anaconda3\lib\site-packages\sklearn\discriminant_analysis.py:836: RuntimeWarning: divide by zero encountered in log
  u = np.asarray([np.sum(np.log(s)) for s in self.scalings_])
C:\Users\pc\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Precision is ill-defined and being set to 0.
0 due to no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))

Working
MultiOutputClassifier
Working
Working
Working
OneVsOneClassifier
OneVsRestClassifier
OutputCodeClassifier
Working
Working
Working
Working
Working
Working
Working
Working
Working
Working
StackingClassifier
VotingClassifier
```

In [15]:
```
train_x
```

Out[15]:
```
array([[0.83333333, 0.        , 1.        , 0.        , 0.74074074,
        0.        , 0.488     ],
       [0.16666667, 0.6       , 1.        , 1.        , 0.0308642 ,
        1.        , 0.07041353],
       [0.33333333, 0.8       , 1.        , 1.        , 0.13425926,
        1.        , 0.03657895],
       [0.        , 0.8       , 1.        , 1.        , 0.12037037,
        1.        , 0.06515038],
       [0.5       , 0.4       , 0.        , 0.        , 0.97222222,
        0.        , 0.644     ],
       [0.        , 0.4       , 1.        , 0.        , 0.90740741,
        0.        , 0.604     ],
       [0.5       , 0.2       , 1.        , 0.        , 0.80246914,
        0.        , 0.64      ],
       [0.16666667, 0.        , 1.        , 0.        , 0.81790123,
        0.        , 0.736     ],
       [0.        , 0.4       , 0.        , 0.        , 0.91975309,
        0.        , 0.668     ],
       [0.16666667, 0.6       , 1.        , 0.        , 0.70679012,
```

In [75]:
```
from sklearn.utils import all_estimators
estimators = all_estimators(type_filter = 'classifier')


for name, get_model in estimators:
    try:
        model = get_model()
        model.fit(train_x, train_y)
        pred_y = model.predict(test_x)
        print(name)
        print(sm.precision_score(test_y, pred_y, average = 'weight'))
        print(sm.recall_score(test_y, pred_y, average = 'macro'))
        print(sm.accuracy_score(test_y, pred_y))
    except Exception as e:
        print(name)
        continue
```

```
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
<ipython-input-75-870496dc19ac>:8: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y
```

In [17]:
```
estimators
```

Out[17]:
```
[('AdaBoostClassifier', sklearn.ensemble._weight_boosting.AdaBoostClassifier),
 ('BaggingClassifier', sklearn.ensemble._bagging.BaggingClassifier),
 ('BernoulliNB', sklearn.naive_bayes.BernoulliNB),
 ('CalibratedClassifierCV', sklearn.calibration.CalibratedClassifierCV),
 ('CategoricalNB', sklearn.naive_bayes.CategoricalNB),
 ('ClassifierChain', sklearn.multioutput.ClassifierChain),
 ('ComplementNB', sklearn.naive_bayes.ComplementNB),
 ('DecisionTreeClassifier', sklearn.tree._classes.DecisionTreeClassifier),
 ('DummyClassifier', sklearn.dummy.DummyClassifier),
 ('ExtraTreeClassifier', sklearn.tree._classes.ExtraTreeClassifier),
 ('ExtraTreesClassifier', sklearn.ensemble._forest.ExtraTreesClassifier),
 ('GaussianNB', sklearn.naive_bayes.GaussianNB),
 ('GaussianProcessClassifier',
  sklearn.gaussian_process._gpc.GaussianProcessClassifier),
 ('GradientBoostingClassifier',
  sklearn.ensemble._gb.GradientBoostingClassifier),
 ('HistGradientBoostingClassifier',
  sklearn.ensemble._hist_gradient_boosting.gradient_boosting.HistGradientBoostingClassifier),
 ('KNeighborsClassifier',
  sklearn.neighbors._classification.KNeighborsClassifier),
 ('LabelPropagation',
  sklearn.semi_supervised._label_propagation.LabelPropagation),
 ('LabelSpreading', sklearn.semi_supervised._label_propagation.LabelSpreading),
 ('LinearDiscriminantAnalysis',
  sklearn.discriminant_analysis.LinearDiscriminantAnalysis),
 ('LinearSVC', sklearn.svm._classes.LinearSVC),
 ('LogisticRegression', sklearn.linear_model._logistic.LogisticRegression),
 ('LogisticRegressionCV', sklearn.linear_model._logistic.LogisticRegressionCV),
 ('MLPClassifier',
  sklearn.neural_network._multilayer_perceptron.MLPClassifier),
 ('MultiOutputClassifier', sklearn.multioutput.MultiOutputClassifier),
 ('MultinomialNB', sklearn.naive_bayes.MultinomialNB),
 ('NearestCentroid', sklearn.neighbors._nearest_centroid.NearestCentroid),
 ('NuSVC', sklearn.svm._classes.NuSVC),
 ('OneVsOneClassifier', sklearn.multiclass.OneVsOneClassifier),
 ('OneVsRestClassifier', sklearn.multiclass.OneVsRestClassifier),
 ('OutputCodeClassifier', sklearn.multiclass.OutputCodeClassifier),
 ('PassiveAggressiveClassifier',
  sklearn.linear_model._passive_aggressive.PassiveAggressiveClassifier),
 ('Perceptron', sklearn.linear_model._perceptron.Perceptron),
 ('QuadraticDiscriminantAnalysis',
  sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis),
 ('RadiusNeighborsClassifier',
  sklearn.neighbors._classification.RadiusNeighborsClassifier),
 ('RandomForestClassifier', sklearn.ensemble._forest.RandomForestClassifier),
 ('RidgeClassifier', sklearn.linear_model._ridge.RidgeClassifier),
 ('RidgeClassifierCV', sklearn.linear_model._ridge.RidgeClassifierCV),
 ('SGDClassifier', sklearn.linear_model._stochastic_gradient.SGDClassifier),
 ('SVC', sklearn.svm._classes.SVC),
 ('StackingClassifier', sklearn.ensemble._stacking.StackingClassifier),
 ('VotingClassifier', sklearn.ensemble._voting.VotingClassifier)]
```

In [18]:
```
import sklearn.svm as sv
import sklearn.metrics as sm

model=sv.SVC(kernel='linear')
model.fit(train_x,train_y)
pred_y=model.predict(test_x)
print(sm.classification_report(test_y,pred_y))

import sklearn.model_selection as ms
from sklearn.metrics import confusion_matrix
y_train_pred = ms.cross_val_predict(model, train_x, train_y)
conf_mx = confusion_matrix(train_y, y_train_pred)
conf_mx
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        36
           1       1.00      1.00      1.00        30

    accuracy                           1.00        66
   macro avg       1.00      1.00      1.00        66
weighted avg       1.00      1.00      1.00        66


C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
C:\Users\pc\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expec
ted. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[18]:
```
array([[64,  0],
       [ 0, 70]], dtype=int64)
```

In [78]:
```
from sklearn import decomposition
aa=decomposition.PCA(n_components=2)
data_x=aa.fit_transform(xx)
data_x
```

Out[78]:
```
array([[ 0.22820662, -0.34547209],
       [-1.93656097,  4.79691211],
       [-1.65417317,  2.46436226],
       ...,
       [-1.86050705,  0.47024368],
       [ 7.29306843, -1.25169255],
       [-4.83553281, -3.58375805]])
```

In [ ]:
```
train_x1,test_x1,train_y1,test_y1 = train_test_split(np.c_[data_x],np.c_[yy],test_size=0.33)
```

In [57]:
```
xx1
```

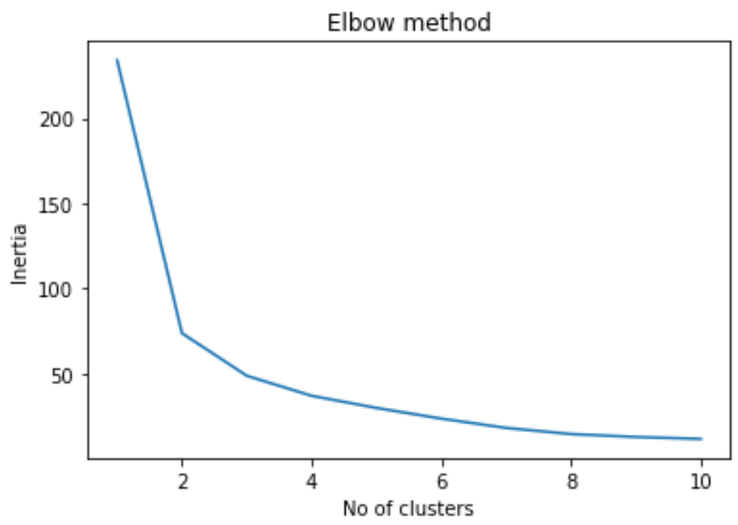
Out[57]:
```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [5.1, 3.8, 1.5, 0.3]
```

In [73]:
```
from sklearn.cluster import KMeans
k = 3
kmeans = KMeans(n_clusters=k)
y_pred = kmeans.fit_predict(xx1)
y_pred
```

Out[73]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2,
       2, 2, 2, 1, 1, 2, 2, 2, 2, 1, 2, 1, 2, 1, 2, 2, 1, 1, 2, 2, 2, 2,
       2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 1, 2, 2, 2, 1, 2, 2, 1])
```

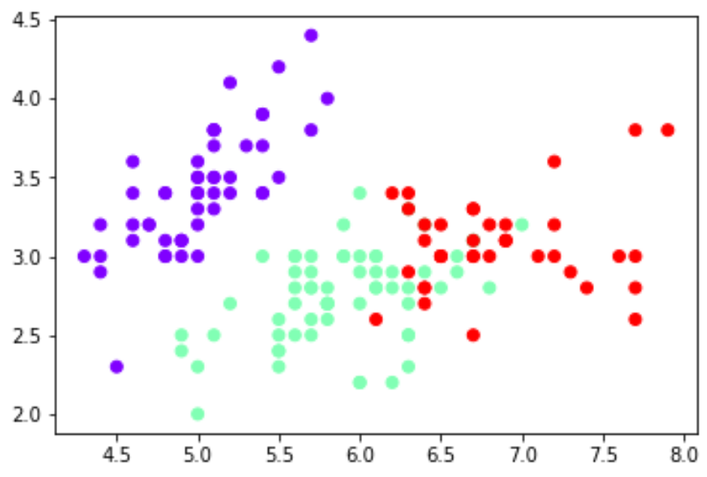
In [71]:
```
Inertia =[]
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i)#.fit(x)
    kmeans.fit(x)
    Inertia.append(kmeans.inertia_)
import matplotlib.pyplot as plt
plt.plot(range(1, 11), Inertia)
plt.title('Elbow method')
plt.xlabel('No of clusters')
plt.ylabel('Inertia')
plt.show()
```

```
C:\Users\pc\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: UserWarning: KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```



In [74]:
```
plt.scatter(xx1[:,0],xx1[:,1],c=y_pred,cmap = 'rainbow')
```

Out[74]:
```
<matplotlib.collections.PathCollection at 0x1c907bc94f0>
```



In [ ]:
```

```