In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
%matplotlib inline
from sklearn import datasets, svm

iris = datasets.load_iris()

X = iris.data
y = iris.target

X = X[y != 0, :2]
y = y[y != 0]

n_sample = len(X)
```

In [2]:
```
iris = sns.load_dataset('iris')
iris_df = pd.DataFrame(iris)
```

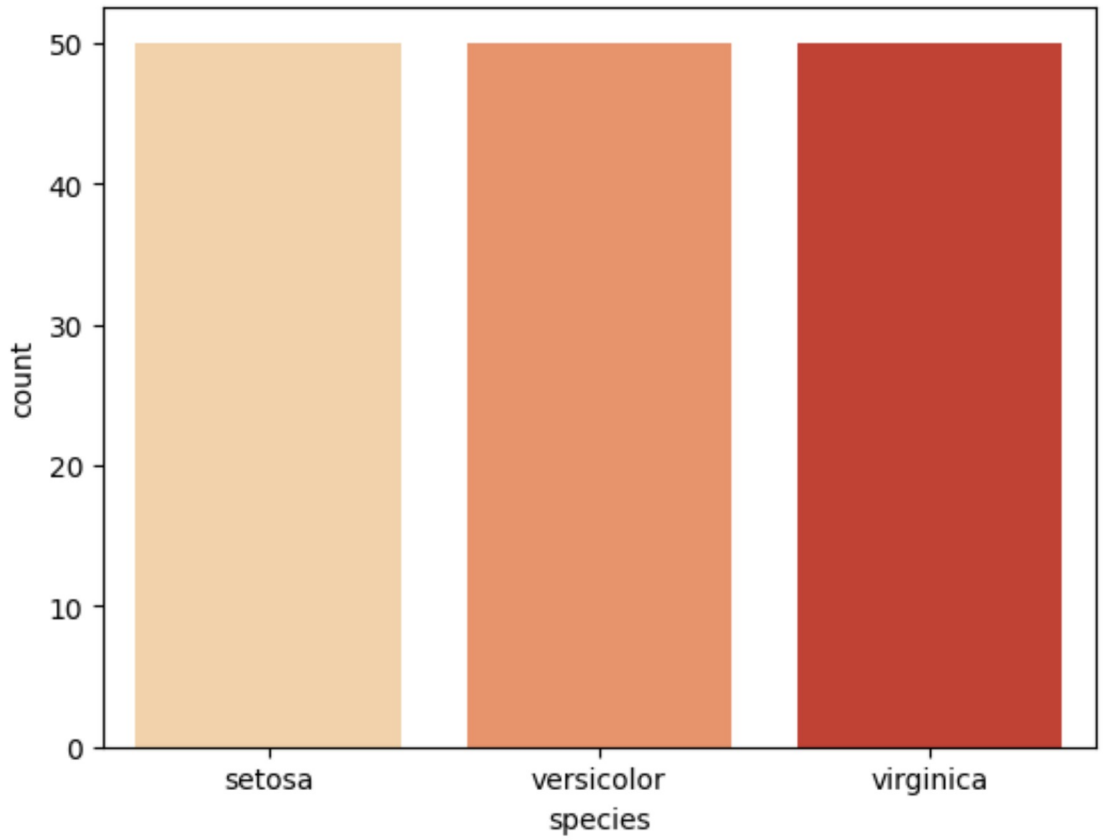
In [3]:
```
#Statistics of this dataset
stats = iris_df.describe()
stats
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [4]:
```
sns.countplot(x='species',data=iris_df, palette="OrRd")
```

Out[4]:
```
<Axes: xlabel='species', ylabel='count'>
```



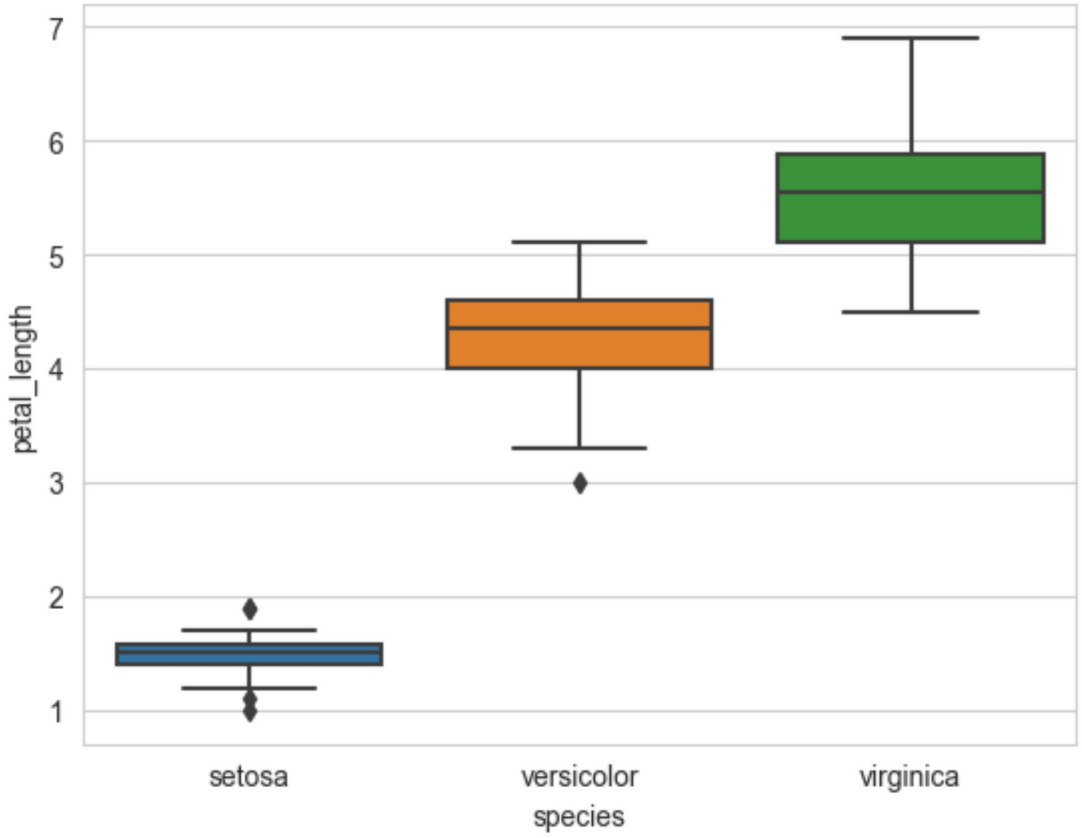
Distribution Plot

In [5]:
```
plt.close();
sns.set_style("whitegrid");
sns.pairplot(iris_df, hue="species",size=3);
plt.show();
```

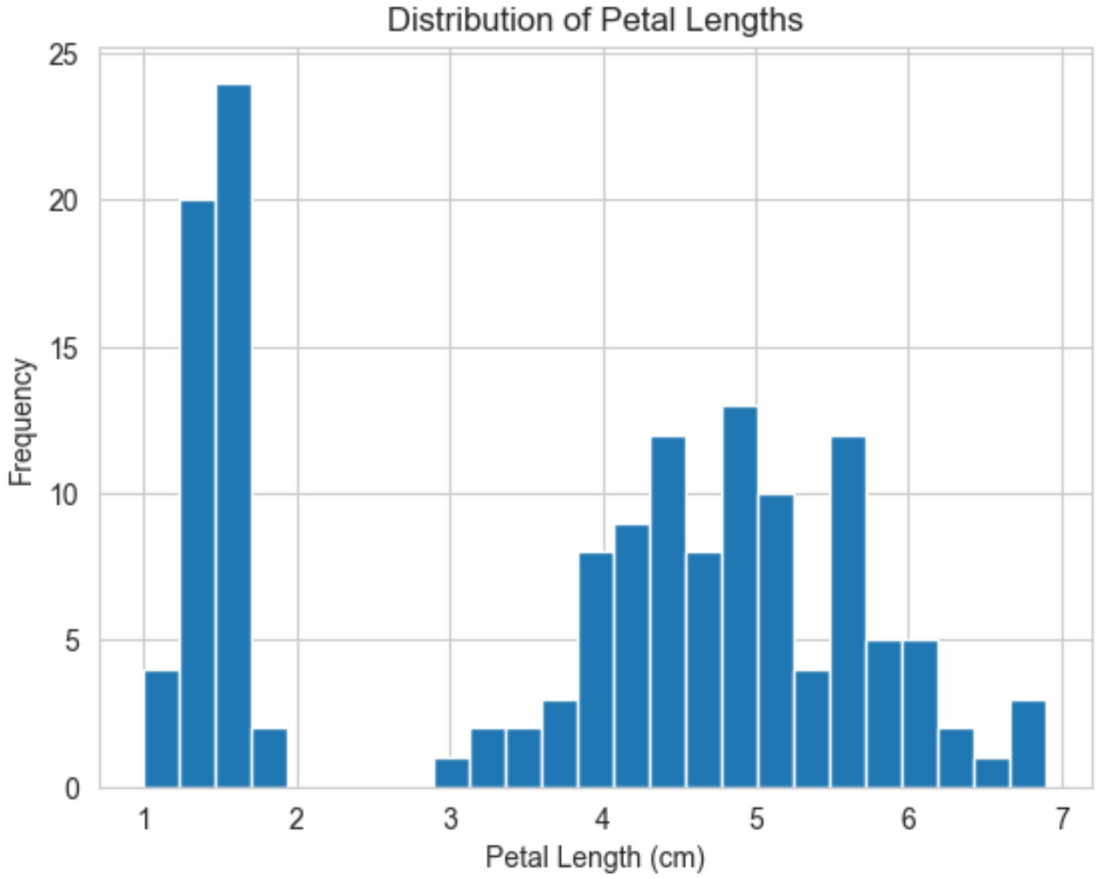
```
C:\Users\ElsaFG\anaconda3\envs\islp\Lib\site-packages\seaborn\axisgrid.py:2095:
UserWarning: The `size` parameter has been renamed to `height`; please update yo
ur code.
  warnings.warn(msg, UserWarning)
C:\Users\ElsaFG\anaconda3\envs\islp\Lib\site-packages\seaborn\axisgrid.py:118: U
serWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Box Plot

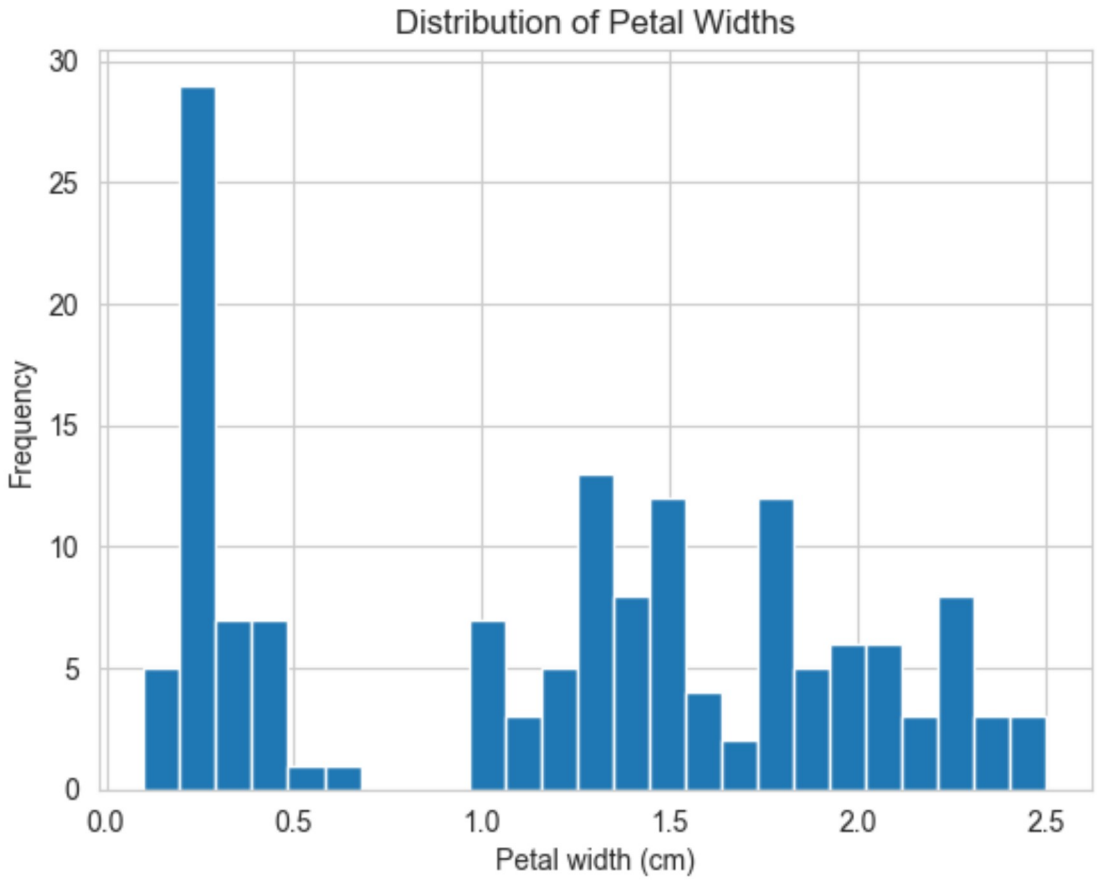
In [6]:
```
sns.boxplot(x='species',y='petal_length', data=iris)
plt.show()
```



In [7]:
```
ax = plt.axes()
ax.hist(iris_df.petal_length, bins=25);

ax.set(xlabel='Petal Length (cm)',
       ylabel='Frequency',
       title='Distribution of Petal Lengths');
```



In [8]:
```
ax = plt.axes()
ax.hist(iris_df.petal_width, bins=25);

ax.set(xlabel='Petal width (cm)',
       ylabel='Frequency',
       title='Distribution of Petal Widths');
```

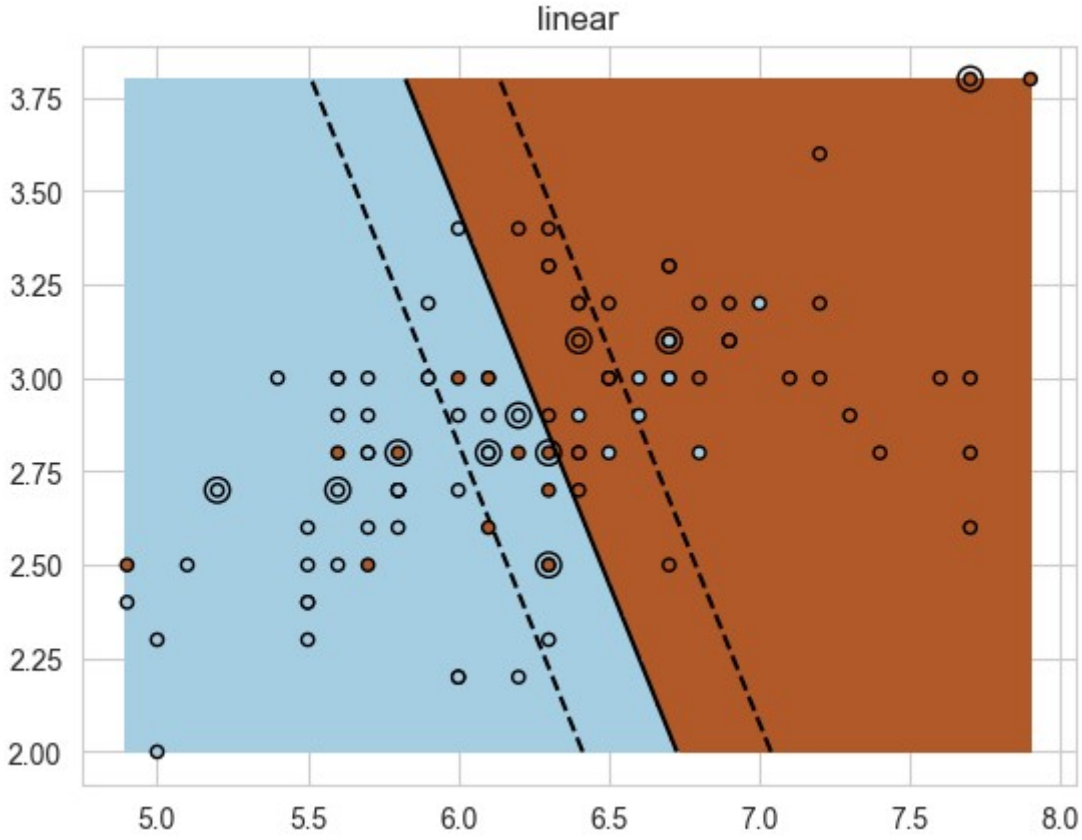


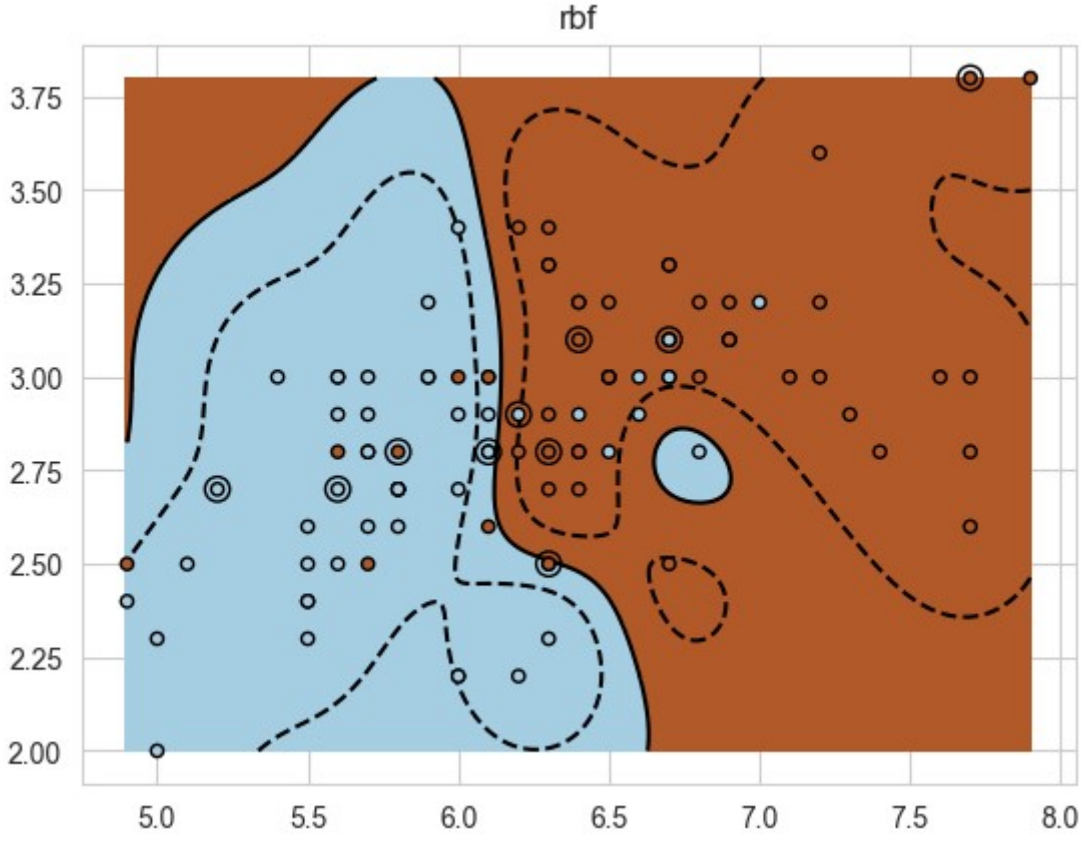
SVM with different kernels

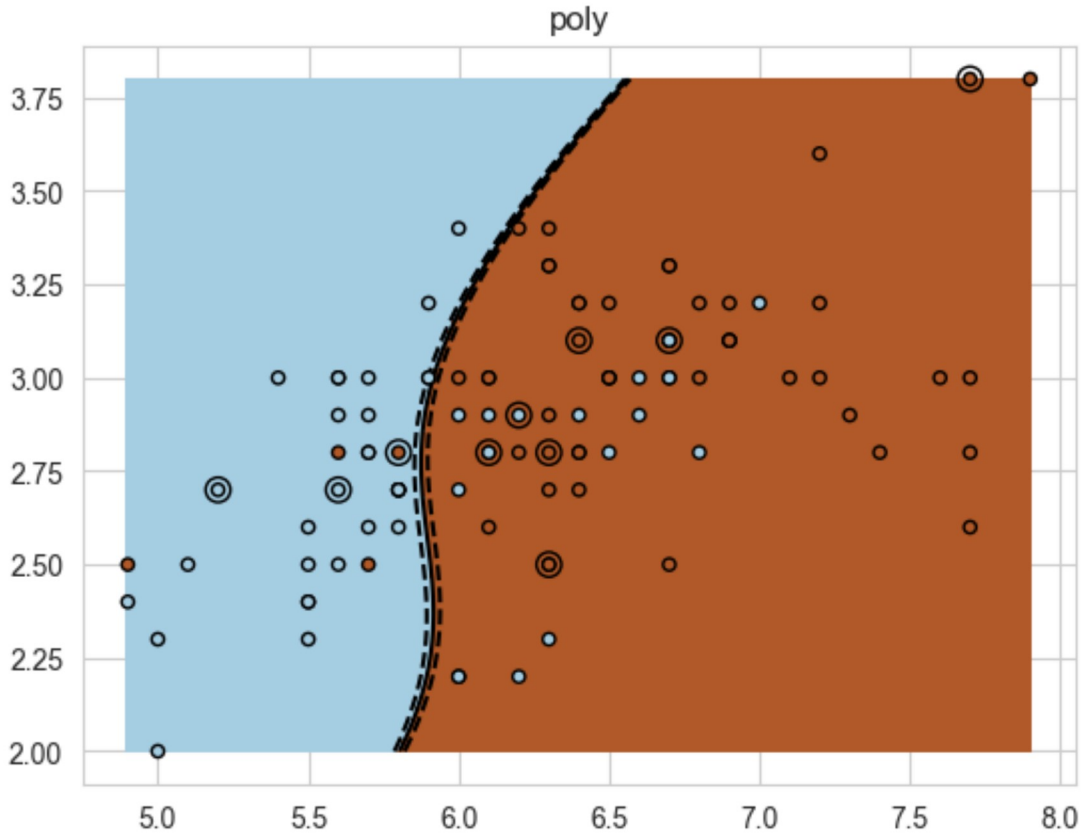
In [9]:
```
np.random.seed(0)

order = np.random.permutation(n_sample)

X = X[order]
y = y[order].astype(float)

X_train = X[: int(0.9 * n_sample)]
y_train = y[: int(0.9 * n_sample)]
X_test = X[int(0.9 * n_sample) :]
y_test = y[int(0.9 * n_sample) :]
```

In [10]:

```
# fit the model
for kernel in ("linear", "rbf", "poly"):
    clf = svm.SVC(kernel=kernel, gamma=10)
    clf.fit(X_train, y_train)

    plt.figure()
    plt.clf()
    plt.scatter(
        X[:, 0], X[:, 1], c=y, zorder=10, cmap=plt.cm.Paired, edgecolor="k", s=2
    )

    # Circle out the test data
    plt.scatter(
        X_test[:, 0], X_test[:, 1], s=80, facecolors="none", zorder=10, edgecolo
    )

    plt.axis("tight")
    x_min = X[:, 0].min()
    x_max = X[:, 0].max()
    y_min = X[:, 1].min()
    y_max = X[:, 1].max()

    XX, YY = np.mgrid[x_min:x_max:200j, y_min:y_max:200j]
    Z = clf.decision_function(np.c_[XX.ravel(), YY.ravel()])

    # Put the result into a color plot
    Z = Z.reshape(XX.shape)
    plt.pcolormesh(XX, YY, Z > 0, cmap=plt.cm.Paired)
    plt.contour(
        XX,
        YY,
        Z,
        colors=["k", "k", "k"],
        linestyles=["--", "-", "--"],
        levels=[-0.5, 0, 0.5],
    )

    plt.title(kernel)
plt.show()
```


linear
3.75
3.50
3.25
3.00
2.75
2.50
2.25
2.00
5.0
5.5
6.0
6.5
7.0
7.5
8.0


rbf
3.75
3.50
3.25
3.00
2.75
2.50
2.25
2.00
5.0
5.5
6.0
6.5
7.0
7.5
8.0



Parameter tunning

In [13]:
```
import sklearn.model_selection as skm

kfold = skm.KFold(5,
                  random_state=0,
                  shuffle=True)
grid = skm.GridSearchCV(svm.SVC(kernel='linear', gamma=10),
                        {'C':[0.001,0.01,0.1,1,5,10,100]},
                        refit=True,
                        cv=kfold,
                        scoring='accuracy')
grid.fit(X, y)
grid.best_params_
```

Out[13]:
```
{'C': 0.1}
```

In [15]:
```
import sklearn.model_selection as skm

kfold = skm.KFold(5,
                  random_state=0,
                  shuffle=True)
grid = skm.GridSearchCV(svm.SVC(kernel='rbf', gamma=10),
                        {'C':[0.1,1,10,100,1000],
                         'gamma':[0.5,1,2,3,4]},
                        refit=True,
                        cv=kfold,
                        scoring='accuracy');
grid.fit(X_train, y_train)
grid.best_params_
```

Out[15]:
```
{'C': 0.1, 'gamma': 0.5}
```

References:

- https://scikit-learn.org/stable/auto_examples/exercises/plot_iris_exercise.html#sphx-glr-auto-examples-exercises-plot-iris-exerc