Support Vector Machine

In [17]:
```
# imports
import scipy.io as sio
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import math
import random
from sklearn import svm, preprocessing
from sklearn.utils import column_or_1d
from sklearn.model_selection import GridSearchCV
%config InlineBackend.figure_format = 'retina'
```

In [ ]:
```
def draw_heatmap_linear(acc, acc_desc, C_list):
    plt.figure(figsize = (2,4))
    ax = sns.heatmap(acc, annot=True, fmt='.3f', yticklabels=C_list, xti
cklabels=[])
    ax.collections[0].colorbar.set_label("accuracy")
    ax.set(ylabel='$C$')
    plt.title(acc_desc + ' w.r.t $C$')
    sns.set_style("whitegrid", {'axes.grid' : False})
    plt.show()
```

Adult

In [5]:
```
arr = np.load('adult.npy') # load the data
np.random.shuffle(arr)    # Shuffle the data.
x = arr[:,:-1]    # First column to second last column: Features (numeri
cal values)
y = arr[:,-1:]    # Last column: Labels (0 or 1)
print(arr.shape, x.shape, y.shape)  # Check the shapes.
```

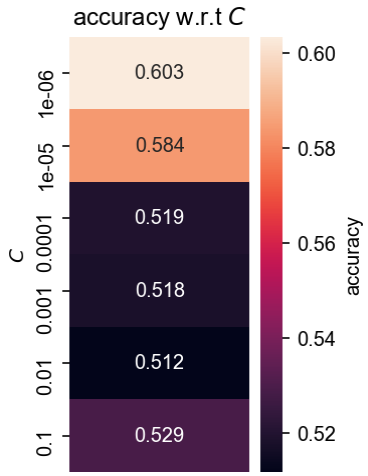
```
(32561, 203) (32561, 202) (32561, 1)
```

In [6]:
```
# Take the first 5000 entries as training data (this will be random as t
he data has been shuffled)
x_train = x[:5000]
y_train = y[:5000]
# the rest is the test set
x_test = x[5000:]
y_test = y[5000:]
print(x_train.shape, y_train.shape, x_test.shape, y_test.shape) # check
 the shapes
```

```
(5000, 202) (5000, 1) (27561, 202) (27561, 1)
```

In [7]:
```
classifier = svm.LinearSVC()
C_list     = [math.pow(10, -6), math.pow(10, -5),math.pow(10, -4),math.p
ow(10, -3),math.pow(10, -2),math.pow(10, -1)]
clf = GridSearchCV(classifier, {'C':C_list})
```

In [18]:
```
clf.fit(x_train, column_or_1d(y_train))
acc = []
for result in clf.cv_results_['mean_test_score']:
    acc.append([result])
draw_heatmap_linear(acc, 'accuracy', C_list)
```



In [39]:
```
# Use the best C to calculate the test accuracy.
c_star_classifier = svm.LinearSVC(C=0.000001)
c_star_classifier.fit(x_train, column_or_1d(y_train))
predictions = c_star_classifier.predict(x_test)
total = 0
for i in range(0, len(predictions)):
    total += 1 if predictions[i] == y_test[i] else 0
test_acc = total / len(predictions)
print('Testing accuracy: ' + str(test_acc))
```

```
Testing accuracy: 0.6132702096484101
```

Cover Type

In [30]:
```
arr = np.load('covtype.npy') # load the data
np.random.shuffle(arr)    # Shuffle the data.
x = arr[:,:-1]    # First column to second last column: Features (numeri
cal values)
y = arr[:,-1:]    # Last column: Labels (0 or 1)
print(arr.shape, x.shape, y.shape)  # Check the shapes.
```

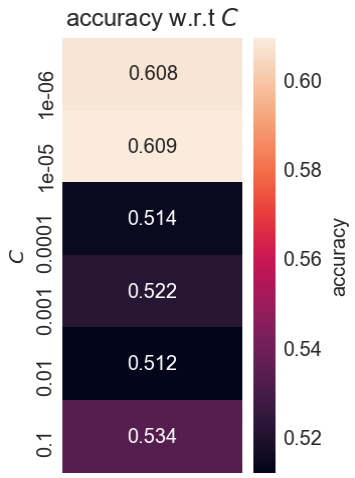
```
(581012, 55) (581012, 54) (581012, 1)
```

In [31]:
```
# Take the first 5000 entries as training data (this will be random as t
he data has been shuffled)
x_train = x[:5000]
y_train = y[:5000]
# the rest is the test set
x_test = x[5000:]
y_test = y[5000:]
print(x_train.shape, y_train.shape, x_test.shape, y_test.shape) # check
 the shapes
```

```
(5000, 54) (5000, 1) (576012, 54) (576012, 1)
```

In [32]:
```
classifier = svm.LinearSVC()
C_list     = [math.pow(10, -6), math.pow(10, -5),math.pow(10, -4),math.p
ow(10, -3),math.pow(10, -2),math.pow(10, -1)]
clf = GridSearchCV(classifier, {'C':C_list})
```

In [33]:
```
clf.fit(x_train, column_or_1d(y_train))
acc = []
for result in clf.cv_results_['mean_test_score']:
    acc.append([result])
draw_heatmap_linear(acc, 'accuracy', C_list)
```



In [36]:
```
# Use the best C to calculate the test accuracy.
c_star_classifier = svm.LinearSVC(C=0.00001)
c_star_classifier.fit(x_train, column_or_1d(y_train))
predictions = c_star_classifier.predict(x_test)
total = 0
for i in range(0, len(predictions)):
    total += 1 if predictions[i] == y_test[i] else 0
test_acc = total / len(predictions)
print('Testing accuracy: ' + str(test_acc))
```

```
Testing accuracy: 0.6113883044103248
```

Letter P1

In [59]:
```
arr = np.load('letter_p1.npy') # load the data
np.random.shuffle(arr)    # Shuffle the data.
x = arr[:,:-1]    # First column to second last column: Features (numeri
cal values)
y = arr[:,-1:]    # Last column: Labels (0 or 1)
print(arr.shape, x.shape, y.shape)  # Check the shapes.
```

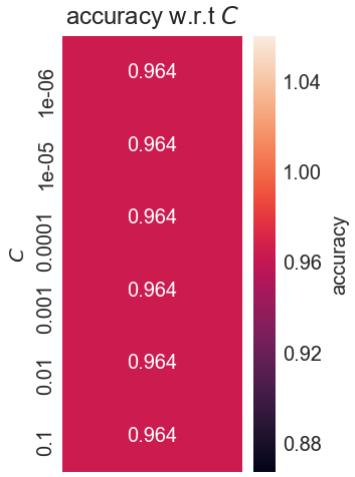
```
(20000, 17) (20000, 16) (20000, 1)
```

In [60]:
```
# Take the first 5000 entries as training data (this will be random as t
he data has been shuffled)
x_train = x[:5000]
y_train = y[:5000]
# the rest is the test set
x_test = x[5000:]
y_test = y[5000:]
print(x_train.shape, y_train.shape, x_test.shape, y_test.shape) # check
 the shapes
```

```
(5000, 16) (5000, 1) (15000, 16) (15000, 1)
```

In [61]:
```
classifier = svm.LinearSVC()
C_list     = [math.pow(10, -6), math.pow(10, -5),math.pow(10, -4),math.p
ow(10, -3),math.pow(10, -2),math.pow(10, -1)]
clf = GridSearchCV(classifier, {'C':C_list})
```

In [62]:
```
clf.fit(x_train, column_or_1d(y_train))
acc = []
for result in clf.cv_results_['mean_test_score']:
    acc.append([result])
draw_heatmap_linear(acc, 'accuracy', C_list)
```



In [63]:
```
# Use the best C to calculate the test accuracy.
c_star_classifier = svm.LinearSVC(C=0.00001)
c_star_classifier.fit(x_train, column_or_1d(y_train))
predictions = c_star_classifier.predict(x_test)
total = 0
for i in range(0, len(predictions)):
    total += 1 if predictions[i] == y_test[i] else 0
test_acc = total / len(predictions)
print('Testing accuracy: ' + str(test_acc))
```

```
Testing accuracy: 0.9619333333333333
```

Letter P2

In [64]:
```
arr = np.load('letter_p2.npy') # load the data
np.random.shuffle(arr)    # Shuffle the data.
x = arr[:,:-1]    # First column to second last column: Features (numeri
cal values)
y = arr[:,-1:]    # Last column: Labels (0 or 1)
print(arr.shape, x.shape, y.shape)  # Check the shapes.
```

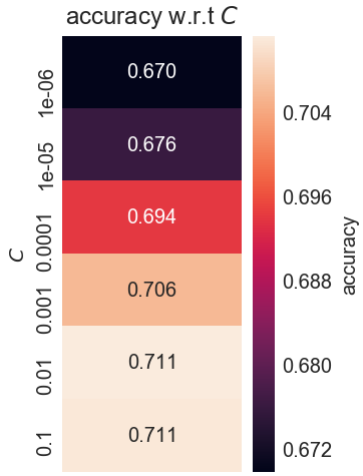
```
(20000, 17) (20000, 16) (20000, 1)
```

In [65]:
```
# Take the first 5000 entries as training data (this will be random as t
he data has been shuffled)
x_train = x[:5000]
y_train = y[:5000]
# the rest is the test set
x_test = x[5000:]
y_test = y[5000:]
print(x_train.shape, y_train.shape, x_test.shape, y_test.shape) # check
 the shapes
```

```
(5000, 16) (5000, 1) (15000, 16) (15000, 1)
```

In [66]:
```
classifier = svm.LinearSVC()
C_list     = [math.pow(10, -6), math.pow(10, -5),math.pow(10, -4),math.p
ow(10, -3),math.pow(10, -2),math.pow(10, -1)]
clf = GridSearchCV(classifier, {'C':C_list})
```

In [67]:
```
clf.fit(x_train, column_or_1d(y_train))
acc = []
for result in clf.cv_results_['mean_test_score']:
    acc.append([result])
draw_heatmap_linear(acc, 'accuracy', C_list)
```



In [12]:
```
# Use the best C to calculate the test accuracy.
c_star_classifier = svm.LinearSVC(C=0.1)
c_star_classifier.fit(x_train, column_or_1d(y_train))
predictions = c_star_classifier.predict(x_test)
total = 0
for i in range(0, len(predictions)):
    total += 1 if predictions[i] == y_test[i] else 0
test_acc = total / len(predictions)
print('Testing accuracy: ' + str(test_acc))
```

```
Testing accuracy: 0.7919161133485723
```