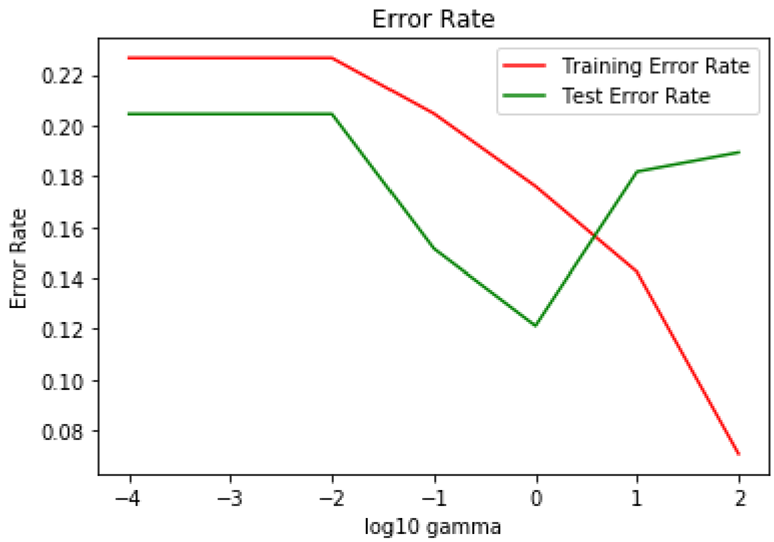
In [39]:
```
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV

import random
%matplotlib inline
```

In [40]:
```
credit_df = pd.read_csv('creditCard.csv')

train, test = train_test_split(credit_df, test_size = 1/10, random_state = 671
)

y_train_df = train.loc[:]['Class']
X_train_df = train.drop(columns=['Class'])

y_test_df = test.loc[:]['Class']
X_test_df = test.drop(columns=['Class'])

min_max_scaler = preprocessing.MinMaxScaler()

# Normalize the dataset
X_train = min_max_scaler.fit_transform(X_train_df)
y_train = y_train_df.to_numpy()
X_test = min_max_scaler.fit_transform(X_test_df)
y_test = y_test_df.to_numpy()
```

In [58]:
```
gamma_range = np.logspace(-4,2,7)

error_test = []
error_train = []
for gamma in gamma_range:
    clf = svm.SVC(gamma = gamma, kernel = 'rbf', C = 1)
    clf.fit(X_train, y_train)
    error_test.append(1-clf.score(X_test, y_test))
    error_train.append(1-clf.score(X_train, y_train))
```

In [88]:
```
def plot_accuracy(acc_train, acc_test, gamma_range):
    gamma_range_log = []
    for i in range(len(gamma_range)):
        gamma_range_log.append(math.log10(gamma_range[i]))
    plt.plot(gamma_range_log,acc_train,'r',label='Training Error Rate')
    plt.plot(gamma_range_log,acc_test,'g',label='Test Error Rate')
    plt.title('Error Rate')
    plt.xlabel('log10 gamma')
    plt.ylabel('Error Rate')
    plt.legend()
    plt.show()
```

In [89]:
```
plot_accuracy(error_train, error_test, gamma_range)
```



The training error decreases as gamma increases.

This is because gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. When gamma is high, the ‘curve’ of the decision boundary is high, which might create islands of decision-boundaries around data points. It will certainly fit the training data better.

Small gamma correspond to large variance.

In [ ]:
```

```