a

The regression implementation is detailed in the file Regression.py

In [1]:

```
from Regression import success_for_normal_regression,success_for_ridge_regression
from scipy.io import loadmat
```

b

We gather the error rates for the training sizes {10,50,100,200}

In [2]:

```
data = loadmat('regdata.mat')['R']
sizes = [10,50,100,200]
results = [ (size,success_for_normal_regression(size,data)) if success_for_normal_regression(size,data) is n
ot None else (size,"Singular matrix") for size in sizes]
print(results)
```

```
[(10, 'Singular matrix'), (50, (0.039488592413051526, 1.0013941248633502)), (100, (0.1061509361
7767357, 0.7400597942733053)), (200, (0.27883497996424933, 0.3925386004868606))]
```

We see that the matrix is singular for training size 10 so we use ridge regression instead

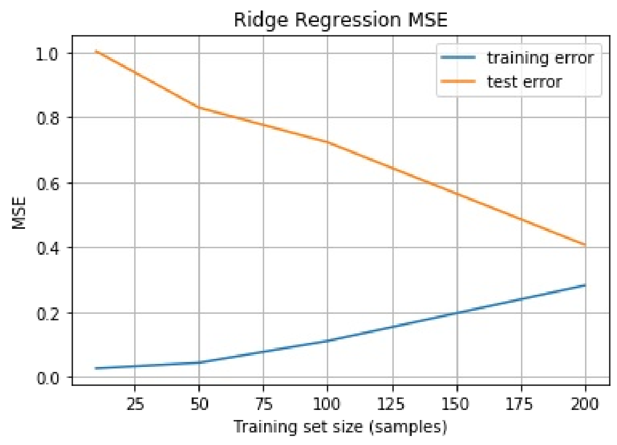
In [3]:

```
results = [(size, success_for_ridge_regression(size, data,gamma=4)) for size in sizes]
print(results)
```

```
[(10, (0.02604874804856051, 1.0027536221066604)), (50, (0.04344212459191873, 0.8292887094113892
)), (100, (0.11015702154100268, 0.7226210586776511)), (200, (0.2811953660750655, 0.407247265311
8184))]
```

In [6]:

```
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
tr_err = [result[1][0] for result in results]
test_err = [result[1][1] for result in results]
ax.plot(sizes, tr_err, label="training error")
ax.plot(sizes, test_err, label="test error")
ax.set(xlabel="Training set size (samples)", ylabel="MSE",title="Ridge Regression MSE")
ax.grid()
ax.legend()
plt.show()
```



c

We can see that the test error is decreasing with the increased training set size meaning we are still generalizeing. Though this might also be because the test set is becoming too small.
Overall we think that there isn't enough data to generalize and test using the costly test/train evaluation