Problem 1: Kernel Regression

In this problem you will work with a well-known example dataset of housing prices in Boston. You will train and test a kernel regression model for this dataset using an RBF kernel.

In [1]:
```
#import necessary libraries
import numpy as np
from sklearn.datasets import load_boston
import warnings
warnings.filterwarnings("ignore") #ignore warnings that may arise. A bad
idea in general, but fine for this exam.

#load dataset
X, y = load_boston(return_X_y=True)
```

1a: Feature Scaling

Apply "standard scaling" to this dataset, resulting in a re-scaled data matrix, `X_rescaled` where the mean and standard devation of each feature is 1.

In [2]:
```
### BEGIN SOLUTION
X_rescaled = (X - X.mean(axis=0))/(X.std(axis=0))
### END SOLUTION
```

In [3]:
```
assert np.isclose(np.linalg.norm(np.mean(X_rescaled, axis=0)), 0)
assert np.isclose(np.std(X_rescaled, axis=0), np.ones(np.size(np.std(X_r
escaled, axis=0)))).all()
```

1b: Construct an RBF kernel

Use every 10th data point from the rescaled data matrix to construct a radial basis function kernel matrix with `gamma = 0.1`. Name your kernel matrix `K_10th`. If you were unable to construct the re-scaled matrix you can use the original data matrix instead.

Recall that the formula for an RBF kernel is:

$$K_{RBF}(i,j) = exp(-\gamma||\vec{x}_i - \vec{x}_j||^2)$$

where $\vec{x}_i$ is the i^{th} data point and $||\vec{a}||$ represents the 2-norm of $\vec{a}$.

In [4]:
```
### BEGIN SOLUTION
X_10th = X_rescaled[::10]

def RBF(X_train, X_predict, gamma=0.1):
    K = np.zeros((X_train.shape[0], X_predict.shape[0]))
    for i in range(K.shape[0]):
        for j in range(K.shape[1]):
            K[i,j] = np.exp(-gamma*(np.linalg.norm(X_predict[j] - X_trai
n[i]))**2)
    return K

K_10th = RBF(X_10th, X_10th)
### END SOLUTION
```

In [5]:
```
eigvals, eigvecs = np.linalg.eig(K_10th)
assert np.isclose(sum(eigvals[:10]), 36.4834)
```

1c: Train and test a kernel regression model.

Train a kernel linear regression model using the RBF kernel for every 10th data point. You do not need to add an intercept term. If you were unable to construct the RBF kernel matrix, you may use every 10th point from the original dataset instead. Compute and report the mean absolute error (MAE) on the full dataset after training.

For full credit, you should only use the Python standard library and `numpy` in this problem, but partial credit will be awarded if `scikit-learn` functions are used.

In [6]:
```
### BEGIN SOLUTION
y_10th = y[::10]
A = np.dot(K_10th.T, K_10th)
b = np.dot(K_10th.T, y_10th)
w = np.linalg.solve(A,b)

K_all = RBF(X_10th, X_rescaled)
yhat = np.dot(w, K_all)
MAE = np.mean(np.abs(y - yhat))
print('MAE = {}'.format(MAE))
### END SOLUTION
```

```
MAE = 3.7956691374521787
```

1d: Hyperparameter optimization for kernel ridge regression

Use the `GridSearchCV` function to determine the optimum hyperparameters for KRR with the Boston housing dataset. You should use the radial basis function kernel, and search over the following range of parameters:

$\alpha \in [1e-4, 1e-3, 1e-2, 1e-1, 1]$

$\gamma \in [1e-6, 1e-5, 1e-4, 1e-3, 1e-2]$

Report the r^2 score of the optimum hyperparamters on a validation set consisting of 30% of the data that is randomly selected from the original dataset, and that is **not used at any point in the hyperparameter training**.

In [7]:

```
### BEGIN SOLUTION
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.kernel_ridge import KernelRidge

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

params = {'alpha':[1e-4,1e-3, 1e-2, 1e-1, 1],
          'gamma':[1e-6, 1e-5, 1e-4, 1e-3, 1e-2]}
KRR = KernelRidge(kernel='rbf')
GSCV = GridSearchCV(estimator=KRR, param_grid=params)
GSCV.fit(X_train, y_train)
best_KRR = GSCV.best_estimator_
print("Optimal hyperparameters: alpha = {}, gamma={}".format(best_KRR.al
pha, best_KRR.gamma))
r2_val = best_KRR.score(X_test, y_test)
print("Validation R^2={:.3f}".format(r2_val))

### END SOLUTION
```

```
Optimal hyperparameters: alpha = 0.0001, gamma=1e-05
Validation R^2=0.841
```