List of Files for this assignment:

ex6.py - Python script that steps you through section 1
ex6data1.mat - Example Dataset 1
ex6data2.mat - Example Dataset 2
ex6data3.mat - Example Dataset 3
ex6utils.py - contains functions for SVM training, SVM prediction, Plot 2D data, Plot linear boundary, and Plot non-linear boundary.
ex6_spam.py - Python script that steps you through section 2
spamTrain.mat - Spam training set
spamTest.mat - Spam test set
emailSample1.txt - Sample email 1
emailSample2.txt - Sample email 2
spamSample1.txt - Sample spam 1
spamSample2.txt - Sample spam 2
vocab.txt - Vocabulary list
[?] ex6modules.py - contains functions for computing Gaussian kernel for SVM and setting parameters to use for Dataset 3
[?] ex6modules_spam.py - contains functions for email preprocessing and feature extraction from emails

Section 1: Support Vector Machines

In this section, you will be using support vector machines (SVMs) with various example 2D datasets. The provided script, ex6.py, will help you step through this exercise.

Part 1: Training SVM w/ Linear Kernel on Dataset 1

We start the exercise by first working with an anonymous dataset that can be separated by a linear boundary. The file ex6data1.mat contains such data. ex6.py loads the data and plots them in the graph. Then, it runs svmTrain() with linear kernel and draws the boundary line that separates the data. The code for doing this has already been written down for you in ex6.py.

Part 2: Training SVM w/ Gaussian Kernel on Dataset 2

In this part of the exercise, we will be working with an anonymous dataset that CANNOT be separated by a linear boundary, but at least can be easily separated with a non-linear boundary. The file ex6data2.mat contains such data. ex6.py loads ex6data2.mat and plots them in the graph. Then, it runs svmTrain() with Gaussian kernel and draws the boundary lines that separate the data. The code for doing this has already been written down for you in ex6.py.

Part 3: Training SVM w/ Gaussian Kernel on Dataset 3

Now, this is the important part. You will be working with an anonymous dataset that CANNOT be separated by a linear boundary, and also CANNOT be easily separated with

a non-linear boundary. The file ex6data3.mat contains such data. ex6.py loads ex6data3.mat and plots them in the graph. It runs dataset3Params() function to optimize the values of C and sigma. Then, it runs svmTrain() with Gaussian kernel and draws the boundary lines that separate the data.
Your task is to properly fill in ex6modules.py. You should complete the module file to return the optimal C and sigma based on a cross-validation set.

Section 2: Spam Classification

In this section, you will be using support vector machines (SVMs) to classify spam emails. The provided script, ex6_spam.py, will help you step through this exercise.

Part 4: Email Preprocessing

To use an SVM to classify emails into Spam vs. Non-Spam, you first need to convert each email into a vector of features. In this part, you will implement the preprocessing steps for each email.
Your task is to complete the code in processEmail.py to produce a word_indices vector for a given email.

Part 5: Feature Extraction

Now, you will convert each email into a vector of features in R^n.
Your task is to complete in ex6modules_spam.py to produce a feature vector for a given email. If your implementation is correct, then the rest of the code in ex6_spam.py will produce outputs as expected. After training the SVM for Spam Classification, you should see that the classifier gets a training accuracy of about 99.8% and a test accuracy of about 98.5%. In addition, note that the “top predictors” for spam are output by ex6_spam.py. “Top predictors” are words that the classifier thinks are the most indicative of spam. For example, if the words “money”, “casino”, and “lottery” are output as “top predictors”, this means that those words makes an email highly likely to be spam. If you have impelmented code correctly, ex6_spam.py will output more than 10 “top predictors” for classifying spam.