# 编程练习6:支持向量机

简介：在练习6中，你将使用支持向量机(SVMs)来构建垃圾邮件分类。在开始本练习之前，使用Matlab中的cd命令切换到该目录。

本练习中包含的文件：

ex6.m - 练习前半部分的Octave脚本

ex6data1.mat – 示例数据集1

ex6data2.mat - 示例数据集2

ex6data3.mat - 示例数据集3

[1] svmTrain.m - svm训练函数

[2] svmPredict.m – svm预测函数

[3] plotData.m – 绘制2D数据

[4] visualizeBoundaryLinear.m –绘制线性边界

[5] visualizeBoundary.m – 绘制非线性边界

[6] linearKernel.m – SVM线性核

[7] gaussianKernel.m – SVM高斯核

[8] dataset3Params.m – 用于数据集3的参数

**1-6：基础部分**

**7-8：核心部分**

ex6 spam.m - 练习后半部分的Octave脚本

spamTrain.mat - 垃圾邮件训练集

spamTest.mat - 垃圾邮件测试集

emailSample1.txt - 示例电子邮件1

emailSample2.txt - 示例电子邮件2

spamSample1.txt - 垃圾邮件示例1

spamSample2.txt - 垃圾邮件示例2

vocab.txt - 词汇列表

[1] getVocabList.m - 加载词汇列表

[2] porterStemmer.m – 词干功能

[3] readFile.m – 将文件读入字符串

[4] processEmail.m – 电子邮件预处理

[5] emailFeatures.m – 从电子邮件中提取特征

在整个练习中，你将使用脚本ex6.m。这些脚本为问题设置了数据集，并调用了你将要编写的函数。你无需修改​​它们中的任何一个。你只需要按照此作业中的说明修改其他文件中的功能。

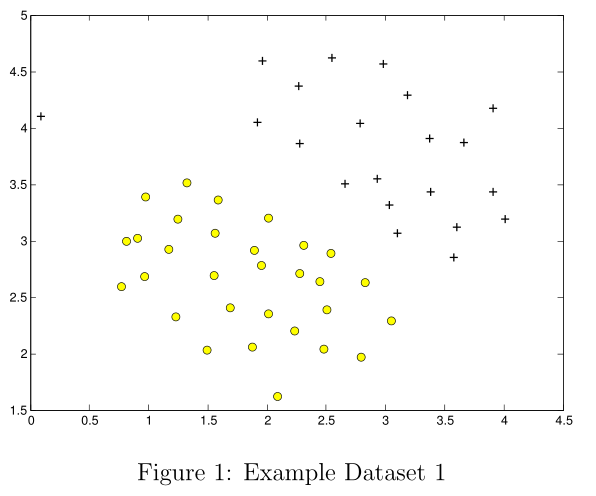
# 1. 支持向量机

在本练习的前半部分，你将使用支持向量机和各种示例2D数据集。对这些数据集进行实验将有助于你直观地了解支持向量机是如何工作的，以及如何在支持向量机中使用高斯核。在练习的下半部分，您将使用支持向量机来构建垃圾邮件分类器。

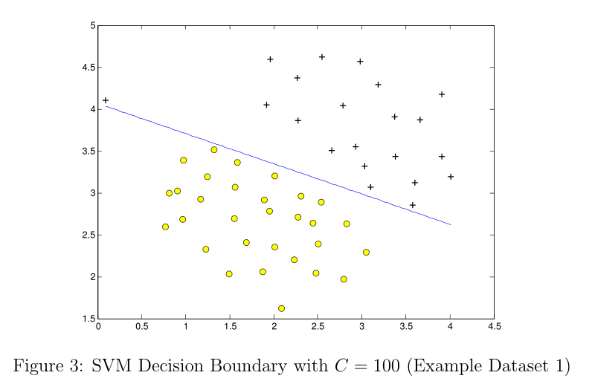
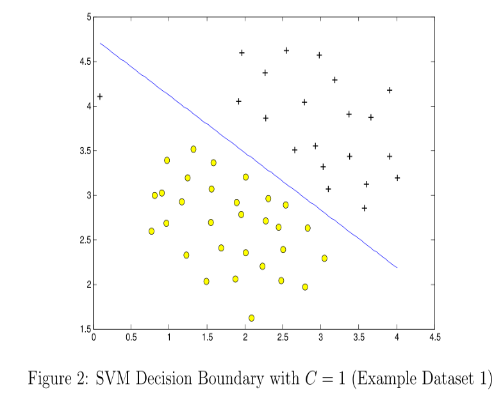
脚本ex6.m将帮助你完成练习的前半部分。

## 1.1示例数据集1

我们将从2D示例数据集开始，该数据集可以由线性边界分隔开。脚本ex6.m将绘制训练数据(图1)。在该数据集中，正例(用+表示)和负例(用o表示)的位置表明由间隙表示的自然分离。然而，请注意，在最左边大约(0.1，4.1)处有一个异常的正例+。作为本练习的一部分，你还将看到这个异常值如何影响SVM决策边界。



在练习的这一部分，你将尝试在SVMs中使用不同的C参数值。一般地说，C参数是一个正值，它控制对错误分类的训练示例的惩罚。一个大的C参数告诉SVM尝试正确分类所有的例子。C起着类似于的作用，其中是我们以前用于逻辑回归的正则化参数。



ex6.m中的下一部分将使用运行SVM训练 (C = 1) 当C = 1时，你会发现SVM将决策边界放在两个数据集之间的间隙中，并将数据点错误地分类在最左边(图2)。

你的任务是在此数据集上尝试不同的C值。具体来说，你应该将脚本中的C值更改为C = 100，并再次运行SVM训练。当C = 100时，你应该会发现SVM现在对每个例子都进行了正确的分类，但是它的决策边界似乎并不自然地适合数据(图3)。

## 1.2高斯核的SVM

在这部分练习中，你将使用支持向量机进行非线性分类。特别注意，你将在不可线性分离的数据集上使用高斯核的支持向量机。

## 1.2.1高斯核

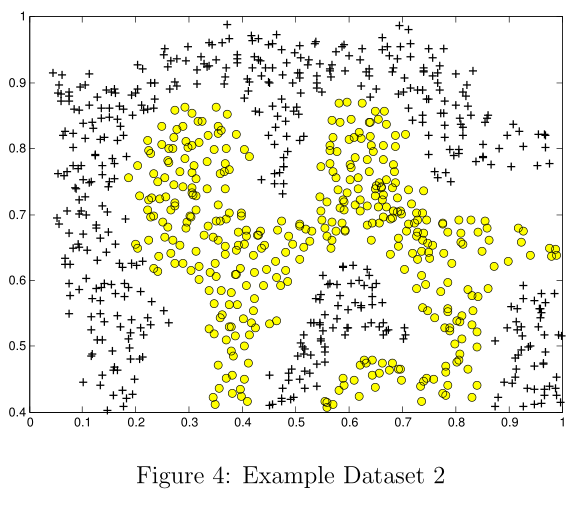
为了找到SVM的非线性决策边界，我们需要首先实现一个高斯核。你可以把高斯核想象成一个相似性函数，用来度量一对例子之间的“距离”(x(i)，x(j))。高斯核也由带宽参数σ参数化，该参数决定了相似性度量随着示例的进一步分离而降低(至0)的速度。

现在，你应该完成gaussianKernel.m中的代码，以计算两个示例x(i)，x(j))之间的高斯内核。高斯核函数被定义为：



完成功能gaussianKernel.m之后，脚本ex6.m将在提供的两个示例中测试你的内核功能，并且你的期望值应该为0.324652。

## 1.2.2示例数据集2



下一部分将加载ex6.m，并绘制数据集2（如图4）。从图中可以看出，没有线性决策边界可以分隔此数据集的正例和负例。但是，通过将高斯内核与SVM结合使用，能够学习到一个非线性决策边界，该边界对于数据集可以表现得相当好。

如果你正确实现了高斯内核函数，则ex6.m将继续在该数据集上训练具有高斯内核的SVM。

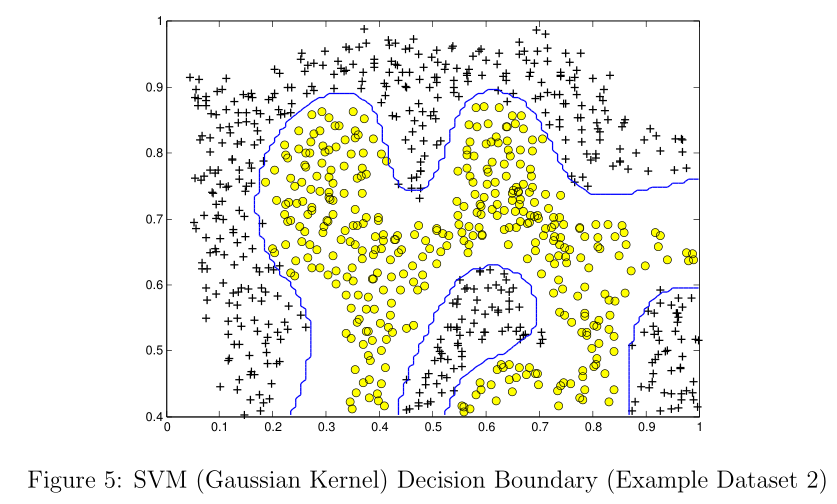
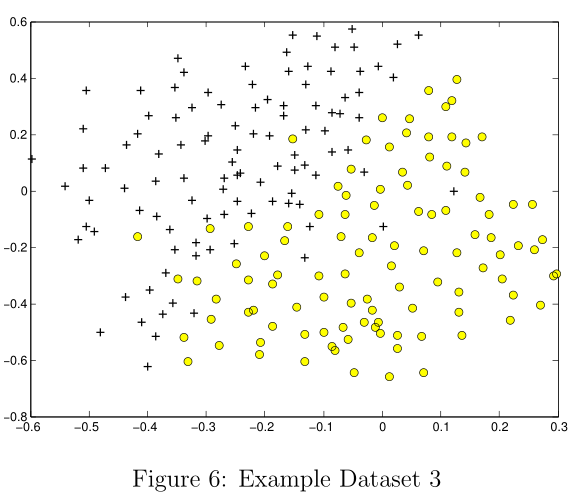


图5显示了具有高斯核的SVM所找到的决策边界。决策边界能够正确地区分大多数正例和负例，并且很好地遵循了数据集的轮廓。

## 1.2.3示例数据集3

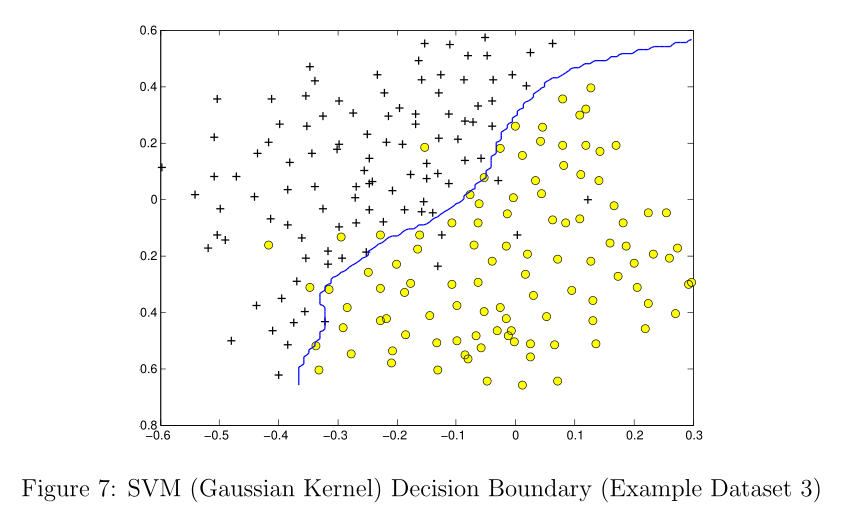
在本部分的练习中，你将获得有关如何将SVM与高斯内核一起使用的更多实用技能。下一部分ex6.m将加载并显示第三个数据集（如图6）。你将在该数据集上使用带有高斯内核的SVM。



在数据集ex6data3.mat中，已经提供了变量X，y，Xval，yval。 ex6.m提供的代码使用从dataset3Params.m中加载的参数，使用训练集（X，y）训练SVM分类器。

你的任务是使用交叉验证集Xval，yval来确定要使用的最佳C和σ参数。你应该编写其他代码来帮助你搜索参数C和σ。对于C和σ，我们建议以乘法步骤尝试值（例如0.01、0.03、0.1、0.3、1、3、10、30）。请注意，你应尝试使用C和σ的所有可能值对（例如C = 0.3和σ= 0.1）。例如，如果尝试为C和列出上面8个值中的每一个，最终将训练和评估（在交叉验证集上）总共82个= 64个不同模型。

确定使用最佳的C和σ参数后，你应该在DataSet3Params.m中修改代码，填充你尝试的最佳参数。如果获得了最佳参数，SVM会返回决策边界，如图7所示。



应用提示：对于分类，错误定义为交叉验证示例中被错误分类的部分。在Octave中，你可以使用mean(double(predictions~= yval))来计算此错误。其中predictions是包含来自SVM的所有预测的向量，而yval是交叉验证集中的真实标签。你可以使svmPredict函数为交叉验证集生成预测。

## 2.垃圾邮件分类

现在，许多电子邮件服务提供了能够将电子邮件分类为垃圾邮件和非垃圾邮件的垃圾邮件过滤器。在本练习中，你将使用SVM来构建您自己的垃圾邮件过滤器。

你可以使用训练分类器来对给定的电子邮件x是垃圾邮件（y = 1）还是非垃圾邮件（y = 0）进行分类。特别的，你需要将每封电子邮件转换为特征向量。本练习的以下部分将带你逐步了解如何从电子邮件中构造这样的特征向量。

在本练习的其余部分中，你将使用脚本ex6\_spam.m。在本练习中，你将仅使用电子邮件的正文（不包括电子邮件标题）。

## 2.1预处理电子邮件

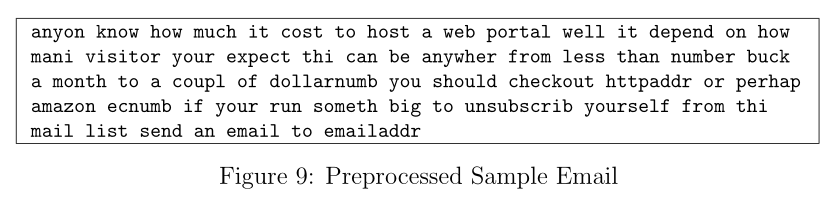


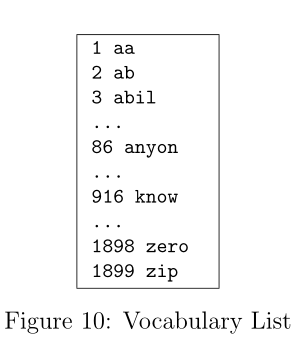
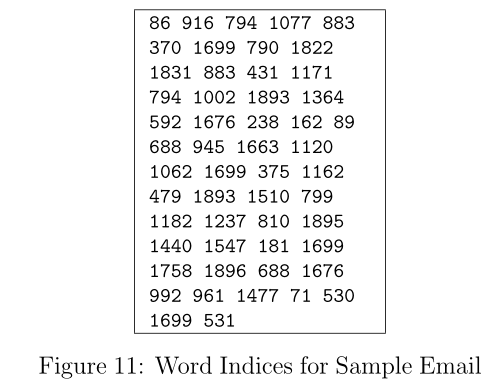
在开始学习任务之前，通常先看一下数据集中的示例是有作用的。图8显示了一个示例电子邮件，其中包含一个URL(统一资源定位系统)，一个电子邮件地址（末尾），数字和美元金额。尽管许多电子邮件将包含相似类型的实体（例如，数字，其他URL或其他电子邮件地址），但几乎每封电子邮件中的特定实体（例如，特定URL或特定美元金额）都会有所不同。因此，处理电子邮件时经常采用的一种方法是“标准化”这些值，以便对所有URL进行相同的处理，对所有数字进行相同的处理，等等。例如，我们可以用唯一的字符串替换电子邮件中的每个URL “ httpaddr”，表示存在URL。（你在现实生活中会遇到的许多其他问题是多维的，无法在二维图上进行绘制。）这能够让垃圾邮件分类程序根据是否存在任何URL而不是是否存在特定URL来做出分类决策的效果。这通常可以提高垃圾邮件分类器的性能，因为垃圾邮件发送者经常将URL随机化，因此在新的垃圾邮件中再次看到任何特定URL的几率很小。

在processEmail.m中，我们将实现以下电子邮件预处理和规范化步骤：：

1. Lower-casing：整个电子邮件被转换成小写，因此大写被忽略(例如，IndIcaTE被视为Indicate)。
2. Stripping HTML: 所有的HTML标签都从电子邮件中删除。许多电子邮件经常带有HTML格式;我们删除所有的HTML标记，这样只剩下内容。
3. Normalizing URLs: 所有的URL被替换成文本“httpaddr”。
4. Normalizing Email Addresses: 所有的电子邮件地址都被替换为文本“emailaddr”。
5. Normalizing Numbers: 所有的数字都被替换为文本“number”。
6. Normalizing Dollars: 所有美元符号($)都被替换为文本“dollar”。
7. Word Stemming: 单词被简化成词根形式。例如，“discount”、“discounts”、“discounted”和“discounting”都被替换为“discount”。有时，词干实际上从末尾去掉了额外的字符，所以" include "， " includes "， " included "和" including"都被替换为" include "
8. Removal of non-words: 非单词和标点符号被删除。所有的空格(制表符、换行符、空格)都被裁剪为一个空格字符。

这些预处理步骤的结果如图9所示。虽然预处理留下了单词和非单词片段，但这种形式在执行特征提取时要容易得多。



## 2.1.1词汇列表

在对电子邮件进行预处理之后，我们为每封电子邮件准备了一个单词列表(如图9)。下一步是选择我们希望在分类器中使用的单词，以及我们希望省略的单词。

在本练习中，我们只选择最常出现的单词作为我们考虑的单词集(词汇表)。由于训练集中很少出现的单词只出现在几封电子邮件中，它们可能会导致模型过度填充我们的训练集。完整的词汇表在文件vocab.txt中，如图10所示。我们的词汇列表是通过选择垃圾邮件语料库中至少出现100次的所有单词来选择的，结果是一个1899个单词的列表。在实践中，通常使用一个大约10，000到50，000个单词的词汇表。

给定词汇表，我们现在可以将预处理电子邮件(如图9)中的每个单词映射到单词索引列表，该列表包含词汇表中单词的索引。图11显示了示例电子邮件的映射。具体来说，在示例电子邮件中，“anyone”一词首先被规范化为“anyon”，然后被映射到词汇表的索引86上。

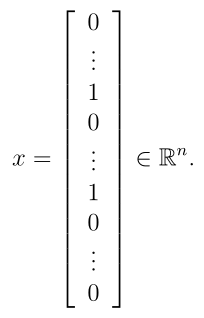
你现在的任务是完成processEmail.m中的代码来执行此映射。在代码中，你会得到一个字符串，它是已处理电子邮件中的一个单词。你应该在词汇表中查找这个词，并找出这个词是否存在于词汇表中。如果该词存在，你应该将该词的索引添加到词索引变量中。如果这个词不存在，因此不在词汇表中，你可以跳过这个词。

一旦您实现了processEmail.m，脚本ex6 spam.m将在电子邮件示例上运行你的代码，你应该会看到类似于图9和11的输出。

注意：在Octave中，可以用strcmp函数比较两个字符串。例如，只有当两个字符串相等时，strcmp(str1，str2)才会返回1。在所提供的起始代码中，不可撤销列表是一个“单元数组”，包含词汇表中的单词。在Octave中，一个单元数组就像一个普通的数组(即一个向量)，除了它的元素也可以是字符串，你用花括号而不是方括号来索引它们。具体来说，要获取索引i处的单词，你可以使用vocablist{i}。你也可以使用length(vocablist)来获得词汇表中的单词数。

## 2.2从电子邮件中提取特征

现在，你将实现特征提取，将每封电子邮件转换为中的向量。在本练习中，你将在词汇表中使用n = #个单词。具体来说，电子邮件的特征对应于字典中的第i个单词是否出现在电子邮件中。也就是说，如果第i个单词在电子邮件中，则= 1，如果第i个单词不在电子邮件中，则= 0。

因此，对于典型的电子邮件，此功能看起来像:

现在，你应该完成emailFeatures.m中的代码，为给定的单词索引(word indices)生成一个电子邮件的特征向量。

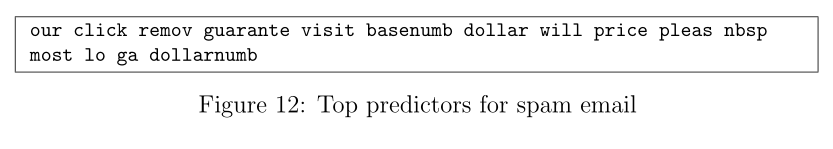
一旦你实现了emailFeatures.m，ex6 spam.m的下一部分将在电子邮件示例上运行你的代码。你应该看到特征向量的长度为1899，有45个非零条目。

## 2.3 SVM垃圾邮件分类训练

完成特征提取功能后，ex6 spam.m的下一步将加载预处理的训练数据集，该数据集将用于训练SVM分类器。spamTrain.mat包含4000个垃圾邮件和非垃圾邮件的训练示例，而spamTest.mat包含1000个测试示例。每封原始电子邮件都使用ProcessemailFeatures和EmailFeatures函数进行处理，并转换为向量。

在加载数据集之后，ex6 spam.m将继续训练一个SVM来对垃圾邮件(y = 1)和非垃圾邮件(y = 0)进行分类。一旦训练完成，你应该会看到分类器的训练准确率约为99.8%，测试准确率约为98.5%。

## 2.4 垃圾邮件的主要预测因素



为了更好地理解垃圾邮件分类器是如何工作的，我们可以检查参数，看看分类器认为哪些单词最能预测垃圾邮件。ex6 spam.m的下一步是在分类器中找到具有最大正值的参数，并显示相应的单词(图12)。因此，如果一封电子邮件包含诸如“保证”、“删除”、“美元”和“价格”(如图12中显示的最重要的预测因素)这样的词，它很可能被归类为垃圾邮件。

## 2.5 可选择练习:尝试你自己的电子邮件

现在你已经训练了一个垃圾邮件分类器，你可以开始在你自己的电子邮件上尝试它。在起始代码中，我们包含了两个电子邮件示例(emailSample1.txt和emailSample2.txt)和两个垃圾邮件示例(spamSample1.txt和spamSample2.txt)。ex6 spam.m的最后一部分在第一个垃圾邮件示例上运行垃圾邮件分类器，并使用所学的SVM进行分类。你现在应该尝试我们提供的其他例子，看看分类器是否正确。你也可以用你自己的电子邮件代替例子(纯文本文件)来尝试你自己的电子邮件。

## 2.6 可选择练习: 构建你自己的数据集

在本练习中，我们提供了预处理的训练集和测试集。这些数据集是使用你现在已经完成的相同函数(processEmail.m和emailFeatures.m)创建的。对于这个练习，你可以使用来自SpamAssassin公共语料库的原始电子邮件构建你自己的数据集。

你的任务是从公共语料库中下载原始文件并提取它们。提取它们之后，你应该在每封电子邮件上运行processEmail3和emailFeatures函数，以从每封电子邮件中提取特征向量。这将允许你建立一个数据集X，y的例子。然后，你应该将数据集随机分为训练集、交叉验证集和测试集。

当你构建自己的数据集时，我们建议你尝试构建自己的词汇表(通过选择高频出现在数据集中的单词) 并添加你认为有用的附加功能。

最后，我们建议使用高度优化的SVM工具箱，如LIBSVM。