Задание № Б-2.3. Машина опорных векторов

Введение

В этом упражнении вы построите спам-фильтр с помощью машины опорных векторов.

1. Машины опорных векторов

В первой части упражнения вы поэкспериментируете с применением машин опорных векторов (SVM) на примере двухмерных задач, которые можно легко визуализировать. Экспериментируя, вы получите более глубокое понимание того, как работает данный вид классификатора, а также как использовать гауссовское ядро с SVM. В следующей части упражнения мы перейдем в более сложные пространства признаков, а именно – мы построим спам-фильтр, классифицирующий письма на два класса: спам и не спам.

Предоставленный скрипт `ex6.m` поможет вам с выполнением первой части упражнения.

1.1 Учебная выборка №1

Мы начнем с двухмерной задачи, которую можно решить с помощью линейной границы, т.е. нам не придется использовать ядра и погружать нашу выборку в более сложные пространства признаков. Скрипт `ex6.m` начинает с визуализации учебной выборки (см. Рисунок 1). Мы видим, что расположение положительных (обозначенных плюсиком) и отрицательных (обозначенных кружком) примеров, между которым явно виден «коридор», подсказывает нам «естественное» решение задачи. Однако мы можем заметить явную аномалию данных – плюсик с координатами (0.1, 0.4), который не позволяет построить «хорошую» решающую границу. В этом упражнении вы посмотрите, как подобные выбросы (outliers) влияют на решающую границу, находимую с помощью SVM.

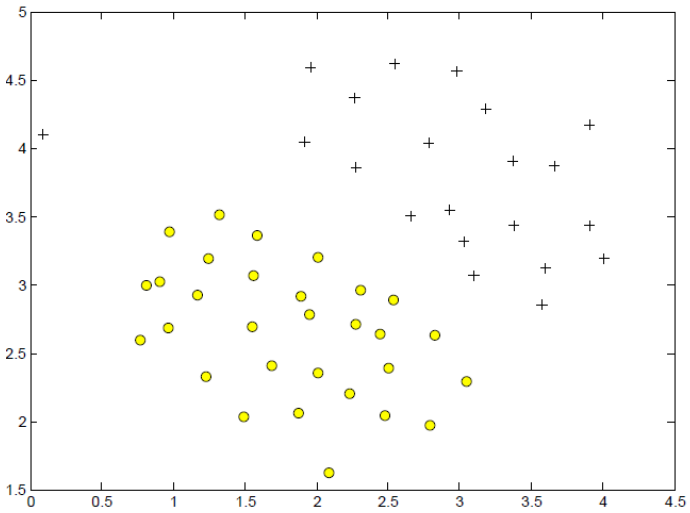

Рисунок 1: Учебная выборка №1

Вы поэкспериментируете с мета-параметром C классификатора SVM. Неформально говоря, параметр C – это положительное число, которое контролирует размер штрафа за неправильно классифицированные примеры из учебной выборки. Чем больше значение C, тем сильнее SVM старается корректно классифицировать все примеры. C играет ту же роль, что и $\frac{1}{\lambda}$, где λ – это параметр регуляризации, который мы использовали в логистической регрессии.

Следующая часть скрипта `ex6.m` запустит обучение SVM со значением $C = 1$, используя уже готовый модуль `svmTrain.m`[1]. Вы можете заметить, что при $C = 1$ классификатор строит границу почти посреди промежутка между примерами из разных классов, при этом указанная нами ранее аномалия оказывается на некорректной стороне от решающей прямой (см. Рисунок 2).

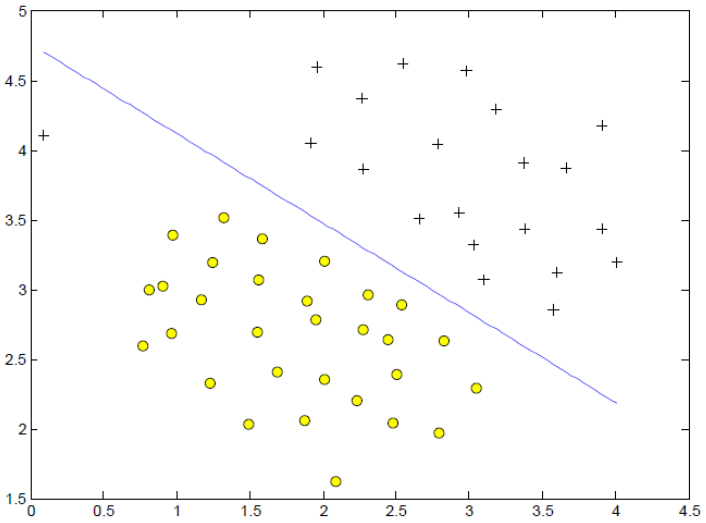

Рисунок 2: Решающая граница машины опорных векторов для $C = 1$

Замечание. Большая часть библиотечных реализаций SVM (включая svmTrain.m) автоматически добавляет к данным дополнительный искусственный признак $x_0 = 1$ и обеспечивает корректное обучение веса θ_0. Поэтому нет необходимости предварительно добавлять столбец из единиц к обучающей выборке, как мы это делали для других классификаторов.

Попробуйте выполнить обучение с разными значениями параметра C. В частности, попробуйте значение $C = 100$, а также меньшие и большие. При $C = 100$ вы заметите, что SVM начнет классифицировать корректно абсолютно все примеры, но это приведет к «переобучению» классификатора и, как следствие, к решающей границе, которая будет внешне выглядеть не совсем естественно для указанной задачи (см Рисунок 3).

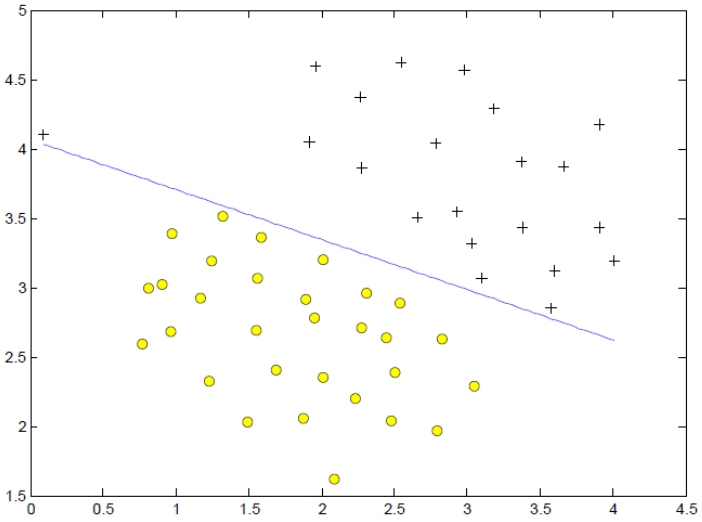

Рисунок 3: Решающая граница машины опорных векторов для $C = 100$

[1] Данный модуль был разработан специально для совместимости кода с платформами Octave/Matlab. При решении реальных задач настоятельно рекомендуем использовать хорошо оптимизированную библиотеку LIBSVM

1.2 SVM с гауссовским ядром

В этой части упражнения мы применим SVM для нелинейной классификации[2]. В частности, мы будем использовать гауссовское ядро для классификации учебной выборки, которая не является линейно сепарабельной.

1.2.1 Гауссовской ядро

Прежде чем применять гауссовской ядро в SVM, нам для начала необходимо разобраться, что это такое и как оно работает. Мы можем представлять гауссовское ядро как некоторую меру близости, которая определяет, насколько «схожи» межу собой пара примеров $(x^{(i)}, x^{(j)})$. Данное ядро имеет параметр σ, которое определяет, насколько быстро значение этой метрики убывает (стремится к нулю) при расхождении примеров. Если мы посмотрим на формулу гауссовского ядра:

$$K_{gaussian}\left(x^{(i)}, x^{(j)}\right) = \exp\left(-\frac{\left\|x^{(i)} - x^{(j)}\right\|^2}{2\sigma^2}\right) = \exp\left(-\frac{\sum_{k=1}^{m}\left(x_k^{(i)} - x_k^{(j)}\right)^2}{2\sigma^2}\right),$$

то мы увидим, что для пары одинаковых чисел $(x^{(i)}, x^{(i)})$ числитель дроби равен нулю, а сама функция будет равна единице, то есть максимальному значению. С другой стороны, чем больше евклидово расстояние[3] между двумя точками, тем меньше будет значение функции.

Вам необходимо завершить код в файле `gaussianKernel.m`, вычисляющий значение гауссовского ядра для двух векторов $(x^{(i)}, x^{(j)})$. После этого скрипт `ex6.m` протестирует вашу функцию для двух тестовых векторов, и вы должны будете увидеть результат, примерно равный 0.324652.

1.2.2 Учебная выборка №2

Следующая часть скрипта `ex6.m` загрузит и отобразит на рисунке вектора из второй учебной выборки (см. Рисунок 4).

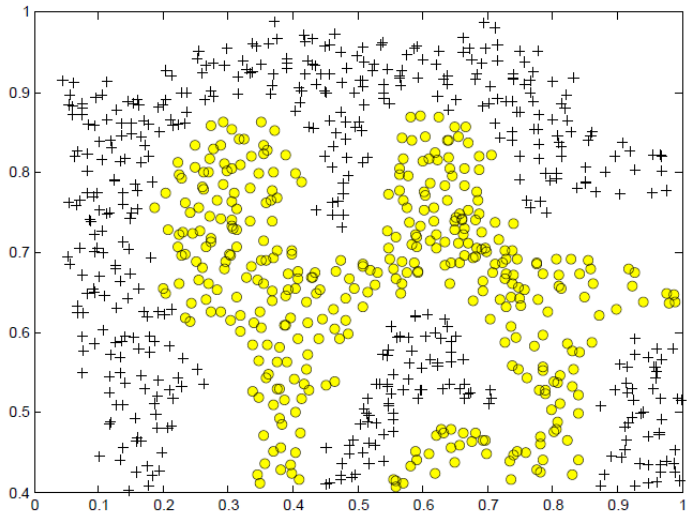

Рисунок 4: Обучающая выборка №2

---

[2] Еще раз обращаем ваше внимание, что речь идет о нелинейности в исходном пространстве признаков. SVM – это линейный классификатор и он находит линейную решающую границу, только эта граница может искаться в более сложном пространстве признаков, чем исходное, и, как следствие, в исходном пространстве признаков будет выглядеть нелинейно.

[3] По умолчанию мы считаем, что $\|a\| = \|a\|_2$ – это L_2 норма, то есть длина вектора a, или для разности: $\|a - b\|$ – евклидово расстояние между точками a и b.

На рисунке наглядно видно, что не существует никакой линейной границы, которая могла быть хоть как-то более-менее адекватно разделить примеры на два класса. Тем не менее, используя гауссовское ядро, вы сможете построить нелинейную решающую границу, которая сможет достаточно хорошо разбить все множество примеров.

Если вы корректно реализовали гауссовское ядро на предыдущем шаге упражнения, то скрипт `ex6.m` приступит к обучению SVM с использованием данного ядра и затем визуализирует полученную границу (см. Рисунок 5). Как видим, найденная решающая граница позволяет правильно классифицировать большинство примеров обучающей выборки.

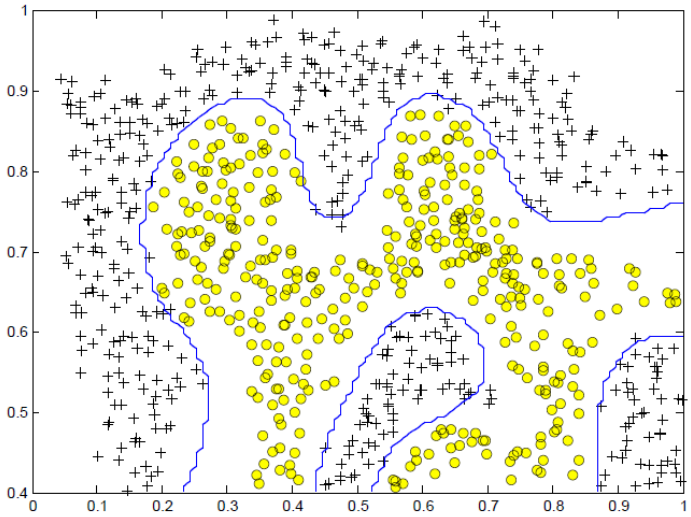

Рисунок 5: Решающая граница для SVM с гауссовским ядром (учебная выборка №2)

1.2.3 Учебная выборка №3

В этой части упражнения вы потренируетесь использовать гауссовское ядро с SVM классификатором. Следующая часть скрипта `ex6.m` загрузит третью учебную выборку и визуализирует ее (см. Рисунок 6).

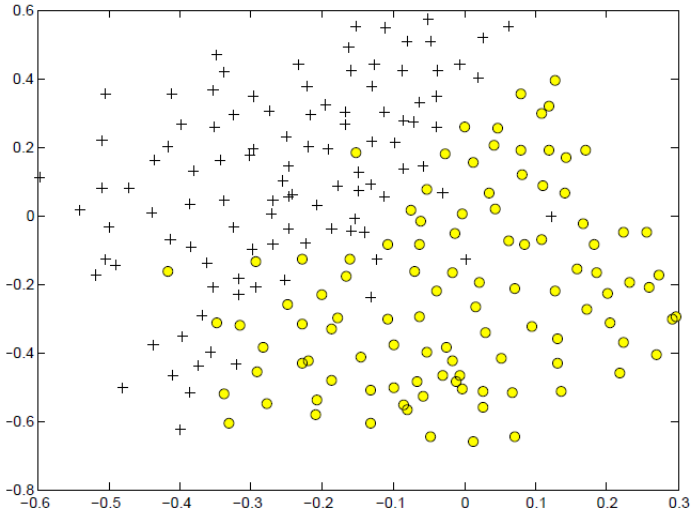

Рисунок 6: Учебная выборка №3

Данная выборка загружается из файла ex6data3.mat в четыре переменные: `X, y, Xval, yval`. Код скрипта `ex6.m` обучает SVM классификатор на этой выборке, используя параметры C и σ, загружаемые из файла `dataset3Params.m`. Изначально значения этих параметров выбраны случайно.

Ваша задача – используя валидационную выборку Xval и yval, определить оптимальные значения для C и σ. Вам необходимо написать произвольный код, который поможет перебрать все пары значений для этих параметров, и найдет ту пару, на которой SVM-классификатор будет показывать наилучший результат. Мы рекомендуем для обоих параметров перебирать значения из какой-нибудь геометрической прогрессии, например, 0.01, 0.03, 0.1, 0.3, 1, 3, 10, 30. Обращаем внимание, что вам нужно перебрать все пары значений. Например, из указанного выше примерного диапазона вариантов у вас получится $8 \times 8 = 64$ пары, с помощью которых вы обучите и протестируете 64 различных модели SVM-классификатора.

После определения оптимальной пары значений параметров, запишите их в файл `dataset3Params.m`. Для ориентира на Рисунке 7 вы видите, как выглядит решающая граница для той пары значений, которую нашли мы.

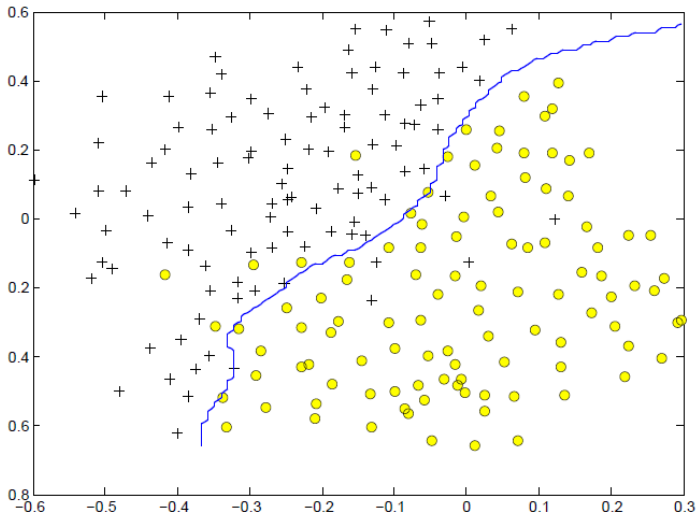

Рисунок 7: Решающая граница для SVM с гауссовским ядром (учебная выборка №3)

Замечание. Для выбора наилучшей пары параметров вам необходимо будет оценивать качество прогноза классификатора на валидационной выборке. Напомним, что ошибка классификации вычисляется как доля примеров валидационной выборки, распознанных классификатором некорректно. В среде Octave/Matlab эту ошибку можно вычислить с помощью выражения `mean(double(predictions ~= yval))`, где predictions – вектор предсказаний, выданных SVM, а yval – вектор истинных меток классов. Вы можете использовать уже готовую функцию svmPredict для получения вектора predictions.

2. Классификация спама

В этой части упражнения мы с вами построим настоящий спам-фильтр, который сможет относить произвольные письма к одному из двух классов: спам и не спам – с достаточно высокой точностью. Для этого мы тоже будем использовать машину опорных векторов.

Вы построите классификатор, который для заданного письма, x, вернет один из двух ответов: $y = 1$ означает, что это спам, а $y = 0$ – что это нормальное письмо. Первое (и главное), что нам предстоит сделать – это научиться преобразовывать произвольное письмо в вектор признаков $x \in \mathbb{R}^n$.

Для этого упражнения вы будете использовать скрипт `ex6_spam.m`. Учебная выборка для данного упражнения является подмножеством датасета SpamAssasin Public Corpus. Мы будем использовать непосредственно тексты писем без заголовочной информации.

2.1 Предварительная обработка писем

Как обычно, прежде чем приступать к решению задачи машинного обучения, полезно посмотреть на данные. На Рисунке 8 приведен образец письма, содержащего e-mail адреса, url ссылки, числа, знаки доллара и т.д.

```
> Anyone knows how much it costs to host a web portal ?
>
Well, it depends on how many visitors youre expecting. This can be
anywhere from less than 10 bucks a month to a couple of $100. You
should checkout http://www.rackspace.com/ or perhaps Amazon EC2 if
youre running something big..
To unsubscribe yourself from this mailing list, send an email to:
groupname-unsubscribe@egroups.com
```

Рисунок 8: Образец письма

И хотя большое количество других писем тоже скорее всего будут содержать сущности подобного рода, но их конкретные значения скорее всего будут отличаться (т.е. это будут какие-то другие адреса, ссылки, числа и т.д.). Поэтому один из методов предварительной обработки заключается в «нормализации» этих сущностей таким образом, чтобы они во всех письмах воспринимались одинаково. Например, мы можем заменить все конкретные URL адреса в письмах на слово “httpaddr”. Эффект от подобной замены будет заключаться в том, что классификатор будет принимать решение, отталкиваясь от того, что в письме содержалась некоторая ссылка, а не от того, какая именно конкретная ссылка это была.

В файле `preprocessEmail.m` мы уже реализовали некоторые шаги по нормализации писем:

- преобразование всех символов в нижний регистр,
- удаление всех HTML тегов,
- вместо всех URL адресов вставляется слово “httpaddr”,
- вместо всех e-mail адресов вставляется слово “emailaddr”,
- вместо всех чисел вставляется слово “number”,
- вместо всех знаков $ вставляется слово “dollar”,
- все слова преобразуются к своим нормальным формам (например, слова “discount”, “discounts”, “discounting” и “discounted” заменяются на “discount”),
- оставляются только слова – убираются все знаки пунктуации и прочие символы; все пробельные символы (пробелы, табуляция, переносы строк) заменяются одним пробелом.

Результат предобработки показан на Рисунке 9. Данная фаза снижает уровень «шума» и оставляет только те элементы письма, которые действительно важны для работы классификатора.

```
anyon know how much it cost to host a web portal well it depend on how mani
visitor your expect thi can be anywher from less than number buck a month
to a coupl of dollarnumb you should checkout httpaddr or perhap amazon ecnumb
if your run someth big to unsubscrib yourself from thi mail list send an
email to emailaddr
```

Рисунок 9: Образец предобработанного письма

2.1.1 Словарь

После предобработки каждого письма из учебной выборки у нас получится список слов для каждого письма (см. Рисунок 9). Следующий шаг – мы должны решить, как слова мы оставим и перенесем в вектор признаков, а какие мы можем выкинуть.

Для этого упражнения мы оставили только самые часто встречающиеся слова. Так как редкие слова встречаются в незначительном количестве писем из учебной выборки, то их наличие может привести к переобучению модели. Полный словарь оставленных слов доступен в файле `vocab.txt` и частично показан на Рисунке 10. Наш словарь был построен следующим образом: отбирались только те слова, который встречаются как минимум 100 раз во всей обучающей выборке. В результате мы отобрали 1899 слов. На практике обычно используются словари с размером от 10000 до 50000 слов.

```
1 aa
2 ab
3 abil
...
86 anyon
...
916 know
...
1898 zero
1899 zip
```

Рисунок 10: Словарь

```
86 916 794 1077 883
370 1699 790 1822
1831 883 431 1171
794 1002 1893 1364
592 1676 238 162 89
688 945 1663 1120
1062 1699 375 1162
479 1893 1510 799
1182 1237 810 1895
1440 1547 181 1699
1758 1896 688 1676
992 961 1477 71 530
1699 531
```

Рисунок 11: Вектор признаков для примерного письма

Имея словарь, мы теперь может поставить в соответствие каждому слову письма его порядковый номер в словаре. На Рисунке 11 изображен вектор признаков, построенный подобный образом по образцу письма, приведенному на Рисунке 9.

Ваша задача – завершить код функции `processEmail.m`, который будет осуществлять построение подобного вектора признаков. В коде вам будет даваться строка str, содержащая одно единственное слово из письма. Вам необходимо найти его в словаре. Если оно там есть, то необходимо добавить его индекс к списку, хранящемуся в word_indices. Если нет – то это слово нужно пропустить.

После того, как вы закончите код в `processEmail.m`, скрипт `ex6_spam.m` запустит его на тестовом примере письма, и вы должны увидеть такой же результат, как на Рисунках 9 и 11.

Подсказка по среде Octave/Matlab. В представленном вам коде переменная vocabList – это так называемый массив ячеек (cell array), содержащий слова из словаря. В среде Octave/Matlab массив ячеек – это массив, в котором можно хранить не только числа, но и строки, и вы индексируете их с помощью фигурных скобок вместо квадратных. В частности, чтобы взять i-тое значение из vocabList необходимо написать vocabList{i}.

2.2 Извлечение признаков из писем

В предыдущем разделе мы познакомились с одним из вариантов построения вектора признаков – в виде списка индексов по словарю. В этом разделе мы рассмотрим альтернативный способ, который преобразует любое письмо в вектор $x \in \mathbb{R}^n$, где каждое $x_i \in \{0,1\}$, а n равно размеру словаря. В данном представлении x_i означает, входит ли i-тое слово в письмо или нет. Иными словами, $x_i = 1$, если слово под номером i из словаря содержится в письме, и $x_i = 0$, если такого слова в письме нет. В результате, для некоторого письма вектор признаков может выглядеть так:

$$x = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{R}^n.$$

Вам необходимо завершить код из файла `emailFeatures.m`, который генерирует вектор признаков описанного формата для произвольного письма, используя массив word_indices.

После того как вы завершите, скрипт `ex6_spam.m` запустит ваш код на тестовом письме. Вы должны увидеть вектор, содержащий 1899 значений, из которых 45 ненулевые.

2.3 Обучение SVM классификатора спам-фильтрации

После того как вы завершите код, преобразующий письма в вектора признаков, скрипт `ex6_spam.m` загрузит предобработанную выборку из `spamTrain.mat` и использует ее для обучения классификатора. Выборка содержит 4000 примеров спама и нормальных писем, а `spamTest.mat` – 1000 примеров. Каждое письмо в этой выборке уже прошло предварительную обработку с помощью функций processEmail и emailFeatures.

После обучения SVM скрипт протестирует качество работы классификатора на обучающей и тестовой выборках. Вы должны получить результаты: 99.8% на обучающей выборке и 98.5% на тестовой.

2.4 Лучшие слова-индикаторы спама

Для лучшего понимания работы спам-фильтра мы можем проанализировать, какие слова он считает наиболее показательными для спама. Следующая часть кода в скрипте `ex6_spam.m` находит веса обученного SVM с наибольшими положительными значениями, и выводит соответствующие им слова (см. Рисунок 12). Таким образом, если письмо содержит такие слова как “guarantee”, “remove”, “dollar” и “price”, то скорее всего оно будет классифицировано как спам.

```
our click remov guarante visit basenumb dollar will price pleas nbsp most
lo ga dollarnumb
```

Рисунок 12: Топ-15 слов-предикторов спама