Задание № Б-3.2. Поиск аномалий

1. Постановка задачи

В этом упражнении вы реализуете алгоритм поиска аномалий и примените его для обнаружения сбоев в работе серверов в компьютерной сети. Предположим, мы делаем замеры следующих характеристик работы сети: пропускную способность (в Мб/с) и задержку (в мс) ответа от сервера. Пока сеть работала, вы собрали $m = 307$ векторов данных, характеризующих работу серверов, и в результате имеете неразмеченную выборку: $\{x^{(1)}, \dots, x^{(m)}\}$. Вы подозреваете, что подавляющая часть замеров соответствует нормальному режиму работы, но в то же время в выборке могут присутствовать и так называемые аномалии – значения характеристик, свидетельствующих о нештатном режиме работы того или иного сервера.

Вы будете использовать нормальное распределение для обнаружения аномалий. Сначала вы поэкспериментируете с двухмерной выборкой, которая позволит вам визуализировать, что делает алгоритм. Для этой выборки вы подберете параметры гауссианы таким образом, чтобы она объясняла все «нормальные» данные. После этого вы сможете использовать эту модель для поиска аномалий – таковыми будут вектора, для которых гауссиана возвращает очень маленькое значение. После этого вы примените данный алгоритм на более серьезной выборке, содержащей много признаков.

Вы будете использовать скрипт `ex8.m` для выполнения данного упражнения. Данный скрипт начнет с того, что, как обычно, визуализирует учебную выборку.

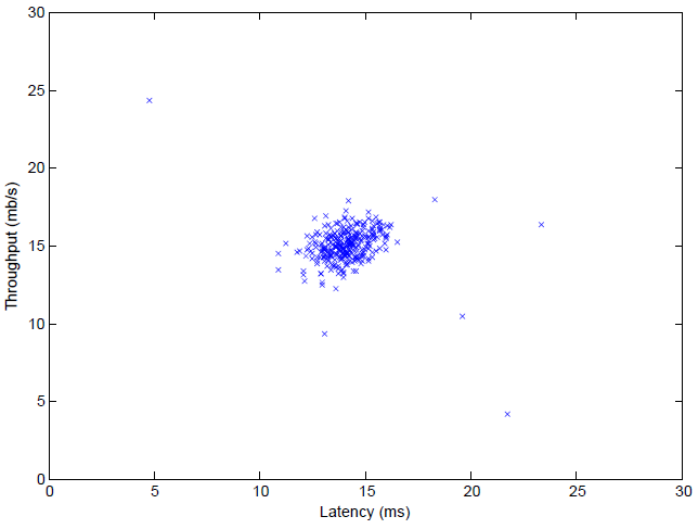

Рисунок 1: Первая выборка данных

1.1 Нормальное распределение

Прежде чем искать аномалии, вы должны подогнать модель к распределению выборки.

Формально задача заключается в следующем: нам дана выборка $\{x^{(1)}, \dots, x^{(m)}\}$, где $x^{(i)} \in \mathbb{R}^n$. Требуется для каждого признака $i = 1 \dots n$ найти параметры распределения μ_i и σ_i^2, которое максимально правдоподобно объясняет данные в i-м измерении: $\left\{x_i^{(1)}, \dots, x_i^{(m)}\right\}$. Гауссовское распределение имеет следующую функцию плотности:

$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

где μ – это среднее, а σ^2 – дисперсия.

1.2 Оценка параметров гауссианы

Вы можете оценить параметры гауссианы (μ_i, σ^2) для i-го признака, используя следующие формулы. Среднее вычисляется как

$$\mu_i = \frac{1}{m}\sum_{j=1}^{m} x_i^{(j)},$$

а дисперсия по формуле:

$$\sigma_i^2 = \frac{1}{m}\sum_{j=1}^{m}\left(x_i^{(j)} - \mu_i\right)^2.$$

Ваша задача – завершить код в файле `estimateGaussian.m`. Функция принимает на вход матрицу данных X, а возвращает n-мерный вектор mu, содержащий средние для каждого измерения, и другой n-мерный вектор sigma2, который, соответственно, хранит дисперсии. Вы можете реализовать код с помощью цикла for по каждому признаку и каждому примеру из учебной выборки (хотя если вам удастся векторизовать код, то он будет работать гораздо эффективней). Заметьте, что в Octave/MATLAB, функция var будет (по умолчанию) умножать сумму квадратов отклонений на $\frac{1}{m-1}$ а не $\frac{1}{m}$ при вычислении σ^2.

После того как вы закончите, скрипт `ex8.m` визуализирует контуры подогнанного распределения. Вы должны получить график как на Рисунке 2. Из графика видно, что основная часть всех примеров учебной выборки находится в области с более высокой плотностью вероятности, в то время как аномалии находятся в областях с наименьшей вероятностью.

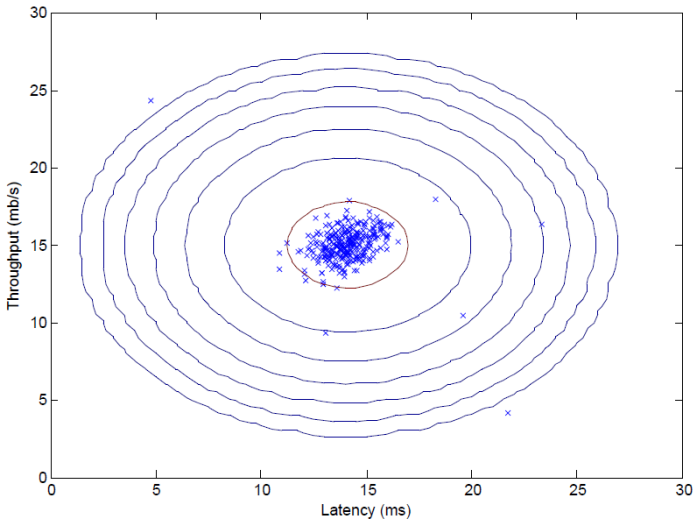

Рисунок 2: Контуры нормального распределения, подогнанного под учебную выборку

1.3 Выбор порогового значения ε

Теперь, когда распределение данных построено, вы можете перейти к анализу того, какие примеры имеют высокую вероятность, а какие низкую. Маловероятные точки имеют высокий шанс оказаться аномалиями данных. Один из способов определения, какие примеры считать аномалиями – это установить пороговое значение на функцию плотности, основываясь на валидационной выборке. В этой части упражнения вы реализуете алгоритм подбора порога ε с помощью метрики качества F_1.

Вам необходимо завершить код в файле `selectThreshold.m`. Вы будете использовать валидационную выборку $\left\{\left(x_{cv}^{(1)}, y_{cv}^{(1)}\right), \ldots, \left(x_{cv}^{(m)}, y_{cv}^{(m)}\right)\right\}$, в которой метка $y = 1$ соответствует

аномальному примеру, а $y = 0$ соответствует нормальному примеру. Для каждого образца из валидационной выборки мы вычислим значение $p\left(x_{cv}^{(i)}\right)$. Вектор всех этих вычисленных значений плотности передается функции selectThreshold в параметре pval. Соответствующие им метки $y_{cv}^{(i)}$ передаются в параметре yval.

Функция selectThreshold должна вернуть две вещи; во-первых, найденное пороговое значение ε. Если пример x имеет низкую плотность вероятности $p(x) < \varepsilon$, то он будет считаться аномалией. Во-вторых, функция должна возвращать значение метрики F_1, означающее, как хорошо вы справляетесь с детектированием известных аномалий с помощью порога ε.

Функция будет вычислять значение F_1 для множества различных значений ε, находя, какое количество примеров из выборки были корректно классифицированы с учетом данного порога.

Метрика F_1 вычисляется с помощью оценок точности (prec) и полноты (recall):

$$F_1 = \frac{2 \cdot prec \cdot recall}{prec + recall}.$$

Точность вычисляется по формуле:

$$prec = \frac{tp}{tp + fp},$$

а полнота:

$$recall = \frac{tp}{tp + fn},$$

где

- tp – количество корректно найденных положительный примеров (true positives): истинная метка свидетельствует, что пример – аномалия, наш алгоритм также распознает аномалию;
- fp – количество некорректно найденных положительных примеров (false positives): истинная метка говорит, что пример – не аномалия, однако наш алгоритм считает его аномалией;
- fn – количество некорректно найденных отрицательных примеров (false negatives): истинная метка говорит, что пример – аномалия, однако наш алгоритм так не считает.

В файле `selectThreshold.m` уже приведен код, который в цикле переберет множество различных значений ε и выберет наилучшее из них с точки зрения F_1-метрики.

Завершите код в `selectThreshold.m`. Вы можете реализовать вычисление F_1-метрики с помощью цикла for, перебирающего все примеры из валидационной выборки (для вычисления tp, fp, fn). Скрипт должен выдать результат тестирования около 8.99e-05.

Замечание. Для более эффективного вычисления tp, fp и fn вы можете векторизовать свой код. Это может быть сделано с помощью выполнения операции сравнения между вектором и числом, в результате чего получится вектор нулей и единиц, в котором на i-м месте будет стоять 1, если i-тый элемент исходного вектора равен числу, и 0 – в противном случае. В частности, вы можете легко посчитать количество нулей в векторе v с помощью операции sum(v == 0). Вы можете также использовать булеву связку & (И), чтобы строить более сложные векторные логические условия. Например, метрику fp можно вычислить так: fp = sum((cvPredictions == 1) & (yval == 0)).

После того, как вы завершите код функции `selectThreshold.m`, скрипт `ex8.m` найдет порог и выделит на графике кружком аномалии, найденные в учебной выборке (см. Рисунок 3).

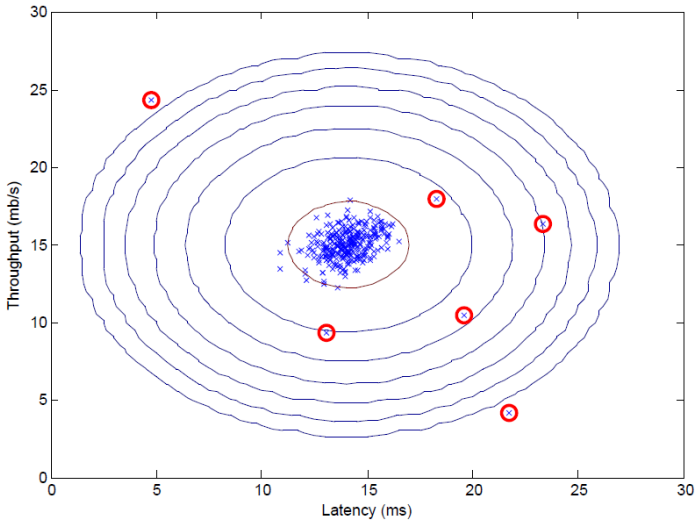

Рисунок 3: Найденные аномалии

1.4 Учебная выборка большой размерности

Последняя часть скрипта `ex8.m` выполнит ваш код для поиска аномалий на данных более высокой размерности. В этой выборке каждый пример описывается одиннадцатью признаками.

Скрипт использует ваш код для оценки параметров гауссианы (μ_i и σ_i^2) и выберет порог ε с помощью валидационной выборки. Вы должны получить значение ε около 1.38e-18 и 117 найденных аномалий.