Методы оптимизации в машинном обучении
Практическое задание #1

Рожин Андрей

НИУ ВШЭ — 20 мая 2022 г.

Вступление

Решение задач оптимизации, является неотъемлемой частью машинного обучения. В этом документе обсуждаются базовые подходы к решению задач этого типа, а также проведение и анализ экспериментов и аналитический вывод формулы логистической регрессии в матричном виде.

$$L(w) = -\frac{1}{m}\sum_{i=1}^{m}\left[y_i log\left(\frac{1}{1+e^{-w^T x_i}}\right) + (1-y_i) log\left(1-\frac{1}{1+e^{-w^T x_i}}\right)\right] + \frac{\lambda}{2}\|w\|^2 \tag{1}$$

Рекомендуется ознакомиться с выкладкой ниже.

Информация:
Документ оформлен согласно этому заданию.
Краткое содержание задания:

I. Двухклассовая логистическая регрессия

Прежде чем начать работу, следует ввести некоторые обозначения, которые будут использоваться в дальнейшем. Все они должны быть привычными, используемыми постоянно в теоретических выкладках.

w - вектор весов объекта
x - матрица значений признаков объектов
y - истинная целевая переменная
m - кол-во объектов
$L(w)$ - функционал ошибки

i. Градиент

Так как мы решаем задачу бинарной классификации, то множество значений, которые принимает целевая переменная y состоит из 2 цифр — {1, 0}. Это очень важный элемент, которые мы будем использовать в дальнейшем, поэтому стоит его запомнить. В формуле (1) заметим сигмоидную функцию, которая дает вероятность класса. Введем функцию.

$$g(z) = \frac{1}{z + e^{-z}} \tag{2}$$

$$h_w(x) = g(w^T x) = \frac{1}{z + e^{-z}}$$

Функция (2) обладает следующими свойствами, которые нетрудно доказать.

$$g\prime(z) = g(z)(1 - g(z))$$
$$g(-z) = 1 - g(z)$$

Перепишем функцию (1), используя новые обозначения

$$L(w) = \frac{1}{m}\sum_{i=1}^{m}\Big[y_i log(g(w^T x)) + (1 - y_i) log(1 - g(w^T x))\Big] + \frac{\lambda}{2}\|w\|^2 \tag{3}$$

Используя эти обозначения и свойства сигмоидной функции, приступим к нахождению производной. Предположим, что у нас есть только один объект с вектором признаков x_i и одно значение целевой переменной y_i. Продифференцируем функцию (3) по j-тому значению вектора весов w. Дифференцировать будем без члена регуляризации, допишем его позднее.

$$\frac{\partial}{\partial w_j} L(w) = -\left[\frac{y_i}{g(w^T x)} - (1 - y_i)\left(\frac{1}{1 - g(w^T x)}\right)\right]\frac{\partial g(w^T x)}{\partial w_j} =$$

$$= -\left[\frac{y_i}{g(w^T x)} - (1 - y_i)\left(\frac{1}{1 - g(w^T x)}\right)\right] g(w^T x)(1 - g(w^T x))\frac{\partial w^T x}{\partial w_j} =$$

$$= -\big[y_i - y_i g(w^T x) - g(w^T x) + y_i g(w^T x)\big] x_j =$$

$$= \big[h_w(x) - y_i\big] x_j$$

Для случая из m объектов получим:

$$\frac{\partial}{\partial w_j} L(w) = \frac{1}{m}\sum_{i=1}^{m}\big[h_w(x_i) - y_i\big] x_{ij} \tag{4}$$

Введем новые обозначения в терминах векторов и матриц.

$X \in \mathbb{R}^{m \times n}$ — матрица объекты-признаки
$y \in \mathbb{R}^{m \times 1}$ — вектор целевых переменных
$w \in \mathbb{R}^{1 \times n}$ — вектор весов
Используя эти обозначения введем матричное представление функции (4)

$$\nabla L(w) = \frac{1}{m} X^T \big[g(Xw) - y\big]$$

Добавим член регуляризации и получим формулу (5).

$$\nabla L(w) = \frac{1}{m} X^T \big[g(Xw) - y\big] + \lambda w \tag{5}$$

Сам функционал логистической регрессии, формула (1), можно представить в такой матричной форме.

$$L(w) = -\frac{1}{m}(1, 1, ..., 1) log\big(-(2y - 1) \circ g(Xw)\big) + \frac{\lambda}{2}\|w\|^2 \tag{6}$$

ii. Гессиан

Выше мы получили функцию (4). Теперь снова продифференцируем ее, в предположении, что у нас один объект в выборке и нет регуляризации.

$$\frac{\partial}{\partial w_j \partial w_j^T} L(w) = \frac{\partial}{\partial w_j^T} \big[g(w^T x_i) - y_i\big] x_i = x_i x_i^T g(w^T x_i)(1 - g(w^T x_i)) \tag{7}$$

Для m объектов и регуляризации формула(7) выглядит так:

$$\nabla^2 L(w) = \frac{1}{m} \sum_{i=1}^{m} \Big[x_i x_i^T g(w^T x_i)(1 - g(w^T x_i)) \Big] - \lambda I \tag{8}$$

Заметим, что матрицу $g(w^T x_i)(1-g(w^T x_i))$ можно заменить диагональной матрицей, на главной диагонали которой, будут располагаться элементы $g(w^T x_i)(1-g(w^T x_i))$. Назовем эту диагональную матрицу буквой Z.

Таким образом, мы получаем матричную форму функции (8).

$$\nabla^2 L(w) = \frac{1}{m} X^T Z X - \lambda I \tag{9}$$

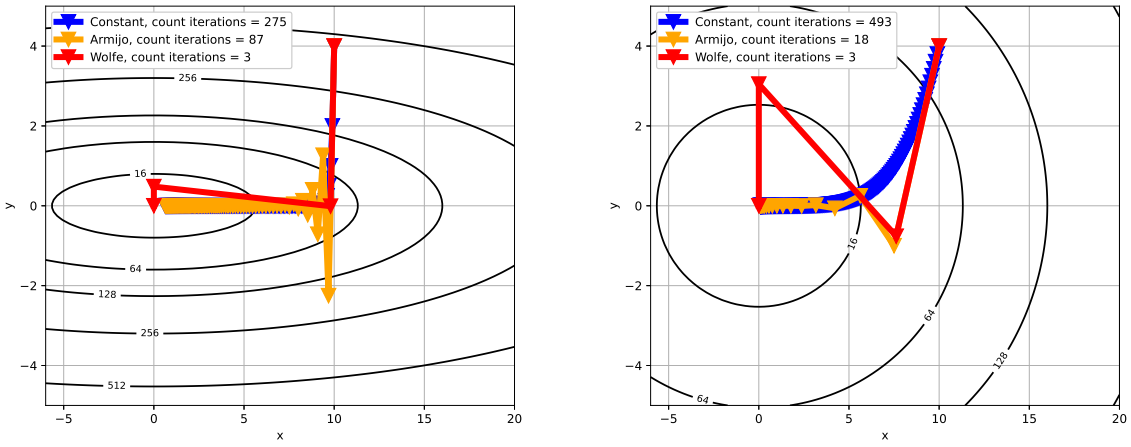
II. Эксперименты

В этой главе мы приступим к оценке реализованных алгоритмов градиентного спуска. Начнем с анализа вариантов линейного поиска шага и закончим логистической регрессией.

i. Траектория градиентного спуска на квадратичной функции

Начнем с *двумерной* квадратичной функции $f(w) = \frac{1}{2}\langle Xw, w\rangle - \langle y, w\rangle$, где $X \in \mathbb{S}^n_{++}$, $b \in \mathbb{R}^n$. Придумаем 4 функции, поведения спуска на которых, будет отличаться.

Обусловленность. Проанализируем графики ниже.

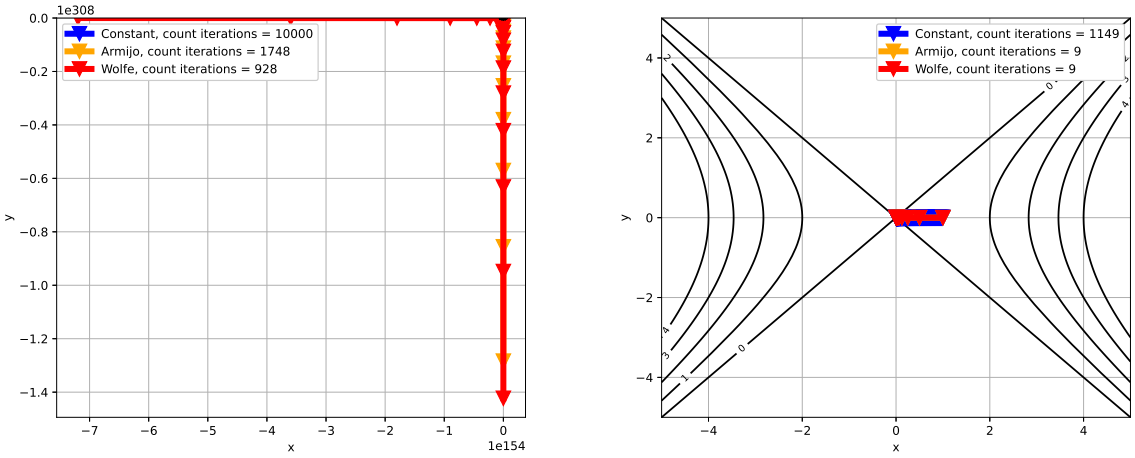


(a) $X = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 50 \end{smallmatrix}\right)$, $w_0 = (10.0, 4.0)$

(b) $X = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 5 \end{smallmatrix}\right)$, $w_0 = (10.0, 4.0)$

Рис. 1: Траектория спуска для различных обусловленностей матрицы

Можно заметить, что, чем ниже обусловленность матрицы, тем больше итераций делает спуск с постоянной длиной шага, а спуску методом Армихо, наоборот, требуется в 3 раза меньше операций. Также траектория спуска с постоянным шагом стала более плавной, на Рис.1 (a), мы видим, как направление спуска резко сменилось на $90°$, а на том же рисунке с литерой (b), траектория меняется более плавно. Траектория Армихо в среднем не изменилась, также видны резкие изменения траектории, но ближе к точке минимума траектория становится более плавной и совпадает со спуском с постоянным шагом. Траектория Вульфа стала более криволинейной, каждый шаг делается практически с разворотом на $90°$, но благодаря подбору шага этому методу требуется меньше итераций, чем двум другим.



(a) $X = \left(\begin{smallmatrix} 0.5 & 0 \\ 0 & -0.5 \end{smallmatrix}\right)$, $w_0 = (3.0, -3.0)$

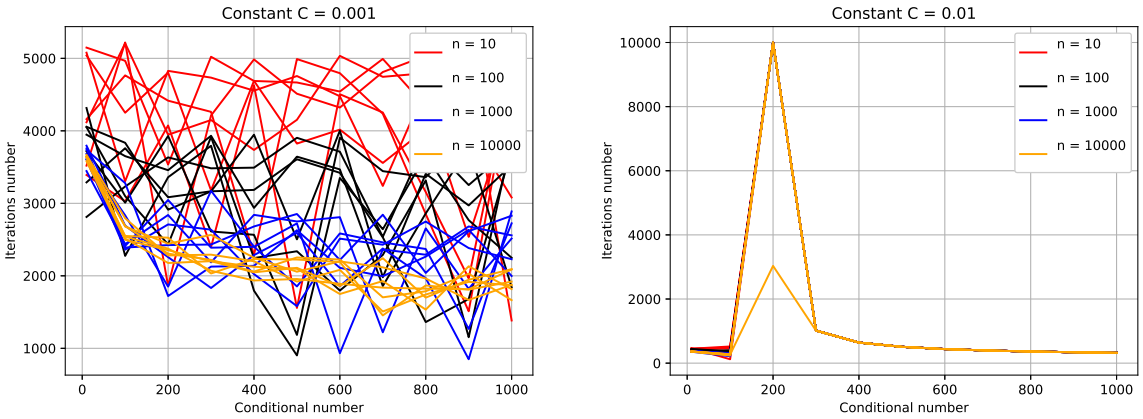
(b) $X = \left(\begin{smallmatrix} 0.5 & 0 \\ 0 & -0.5 \end{smallmatrix}\right)$, $w_0 = (1.0, 0.0)$

Рис. 2: Выбор начальной точки

На Рис.2 ищется локальный минимум функции $f(x) = \frac{1}{2}x^2 - \frac{1}{2}y^2$ — так называемая, седловая функция. Ее особенность заключается в том, что найти ее минимум можно тогда и только тогда, когда начинаем спуск с точки $w_{init} = (1.0, 0.0)$, это можно легко доказать теоретически. Также заметим, что на Рис.2 (a) константный метод дошел до предела итераций, но так и не достиг $-\infty$ по значению функции. Армихо и Вульф достигли этого нижнего предела, сработал критерий останова, причем метод Вульфа снова обогнал другие методы спуска. На Рис. 2 (b) все 3 метода сходятся в одну точку, но спуску с постоянным шагом требуется куда больше итераций для этого.

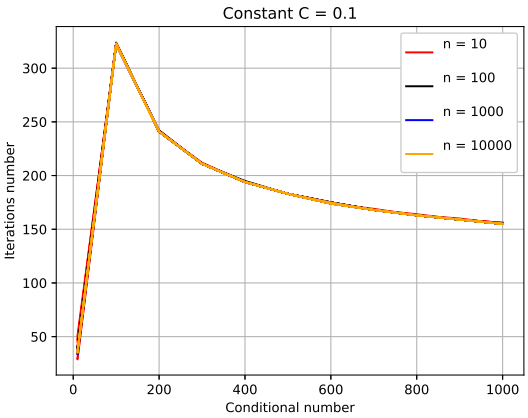
ii. Зависимость числа итераций градиентного спуска от числа обусловленности и размерности пространства

Продолжаем работать с той же самой квадратичной функцией. В предыдущем эксперименте мы сделали некоторые предположения о количестве итераций, необходимых для сходимости спуска с разными методами подбора шага. Сейчас мы построим эту зависимость наглядно. Начнем со спуска с постоянной длиной шага. Изобразим 3 варианта графика, каждый соответсвует уникальной константе для выбора длины шага.

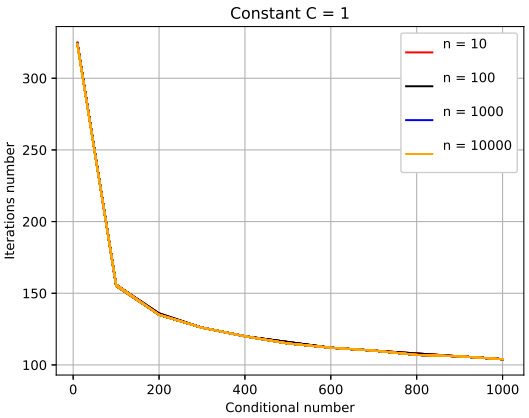


(a)

(b)



(c)

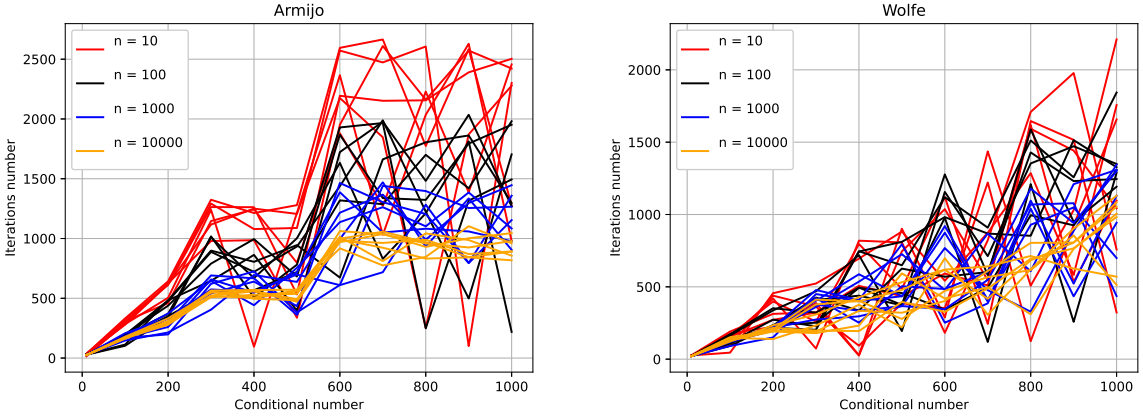


(d)

Рис. 3: Постоянная длина шага

Посмотрим на Рис.3. Под литерой (a) не прослеживается четкой линейной или квадратичной зависимости. Но мы отчетливо видим, что чем ниже размерность матрицы, тем больше итераций для спуска необходимо провести. Кроме того, с увеличением размерности, падает разброс графиков — они становятся более скученными, так, например, для размерности $n = 10000$, мы наблюдаем практически гиперболическую зависимость ($f(x) = \frac{1}{x}$), а также очень малую дисперсию количества итераций, по сравнению с другими размерностями. Если мы увеличим константу на порядок — Рис.3(b) — ,то заметим, как сильно изменилась ситуация. Мы видим довольно скученный график на обусловленности $k <= 100$ для всех размерностей. Далее, в интервале $100 < k <= 300$ для всех размерностей требуется самое большое значение итераций (даже для $n = 10000$, т.к. на графике только 3 линии из 7). Для $k > 300$ значение итераций не столь велико для всех размерностей. Видно, что график выходит на ассимптотику. Продолжаем увеличивать константу — Рис.3(c)(d) — видно, что для таких значений C размерность матрицы не имеет значения, влияет лишь обусловленность, но влияет не сильно, разница между максимальным и минимальным значением итераций на двух графиках равна 200. Также отчетливо прослеживается гиперболическая зависимость, которую мы наблюдали для $C = 0.001$ и $n = 10000$ на Рис.3(a). Также видно, что при $C = 0.1$ и $n \to \infty$ ассимптотически значения количества итераций больше, чем при $C = 1$ и $n \to \infty$.

Теперь переходим к анализу зависимостей для методов Армихо и Вульфа.



(a) (b)

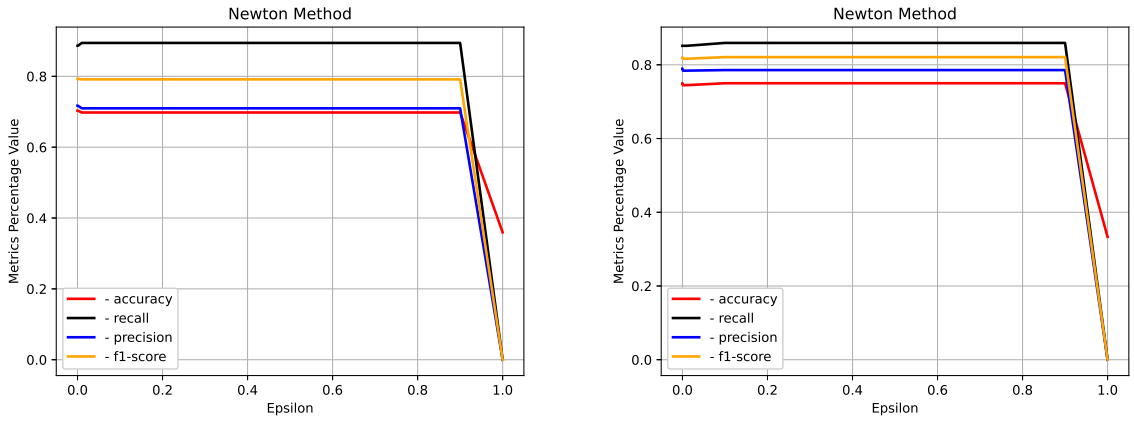
Рис. 4: Армихо и Вульф

На Рис.4(a), который соответствует условию Армихо, мы наблюдаем четкую линейную зависимость в интервале $0 < k <= 300$, далее график становится похож на Рис.3(a), но с меньшим разбросом и меньшей верхней границей количества итераций. При $n = 10000$ график выходит на ассимптотику. На Рис.4(b) — условие Вульфа, видна линейная зависимость на всем интервале обусловленности. Дисперсия количества итераций повышается линейно с увеличением обусловленности функции. На всей области определения разброс всех кривых сильнее, чем у метода Армихо.

iii. Точность оптимизации в реальных задачах для метода Ньютона

Как мы знаем, на практике в задачах классификации, нас редко интересует только минимизация логистической функции потерь. Существует довольно много различных метрик, которые в той или иной степени измеряют качество решения задач. В этом эксперименте, мы попробуем установить зависимость, между точностью ϵ [1] приближения логистической функции и самых распространенных метрик машинной классификации.

Обучать модель будем на реальных данных. В первом сете содержится информация о **наличии диабета**. Во втором о **заболевании сердца**.



(a) Стандартный (b) Масштабированный

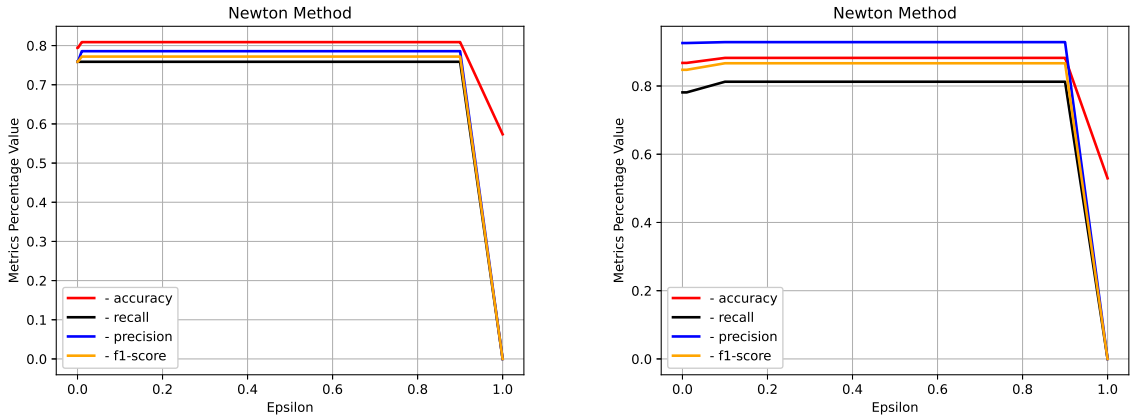
Рис. 5: Диабет

[1] Вспомним критерий останова — $\|\nabla f(x_k)\|^2 \leq \epsilon \|\nabla f(x_0)\|^2$.

Рис.5(a) — обучение модели на ”сырых” данных, тот же рисунок под литерой (б) — данные отмасштабированы на отрезок [-1, 1]. Заметим, что зависимость всех метрик на диапазоне [0.1, 1] линейна от ϵ, это легко понять на уровне интуиции, чем меньше отношение $\frac{\|\nabla f(x_k)\|^2}{\|\nabla f(x_0)\|^2}$, тем ближе мы к точке минимума функции. Очевидно, что ϵ практически не оказывает влияния на метрики, только *accuracy* увеличивается при $\epsilon \neq 1$, ведь $\epsilon = 1$ соответствует случайному угадыванию весов при признаках, соотвественно доля верных ответов вполне может быть в районе 50%.

Куда больший интерес представляет Рис.5(b) — с отмасштабированными признаками. Во-первых, отчетливо видно, что даже при самом плохом приближении — $\epsilon \approx 0.9$ — все метрики, за исключением полноты, оказываются выше примерно на 10 процентных пунктов, относительно метрик на неотмасштабированных данных. Кроме того, при $\epsilon \approx 10^{-8}$ заметны небольшие изменения — повысилась доля верных ответов, упала точность, выросла полнота. У нас появилась гипотеза — метод Ньютона показывает себя лучше на отмасштабированных данных, а от точности приближения практически не зависит.

Проведем этот же эксперимент на других данных.



(a) Стандартный (b) Масштабированный

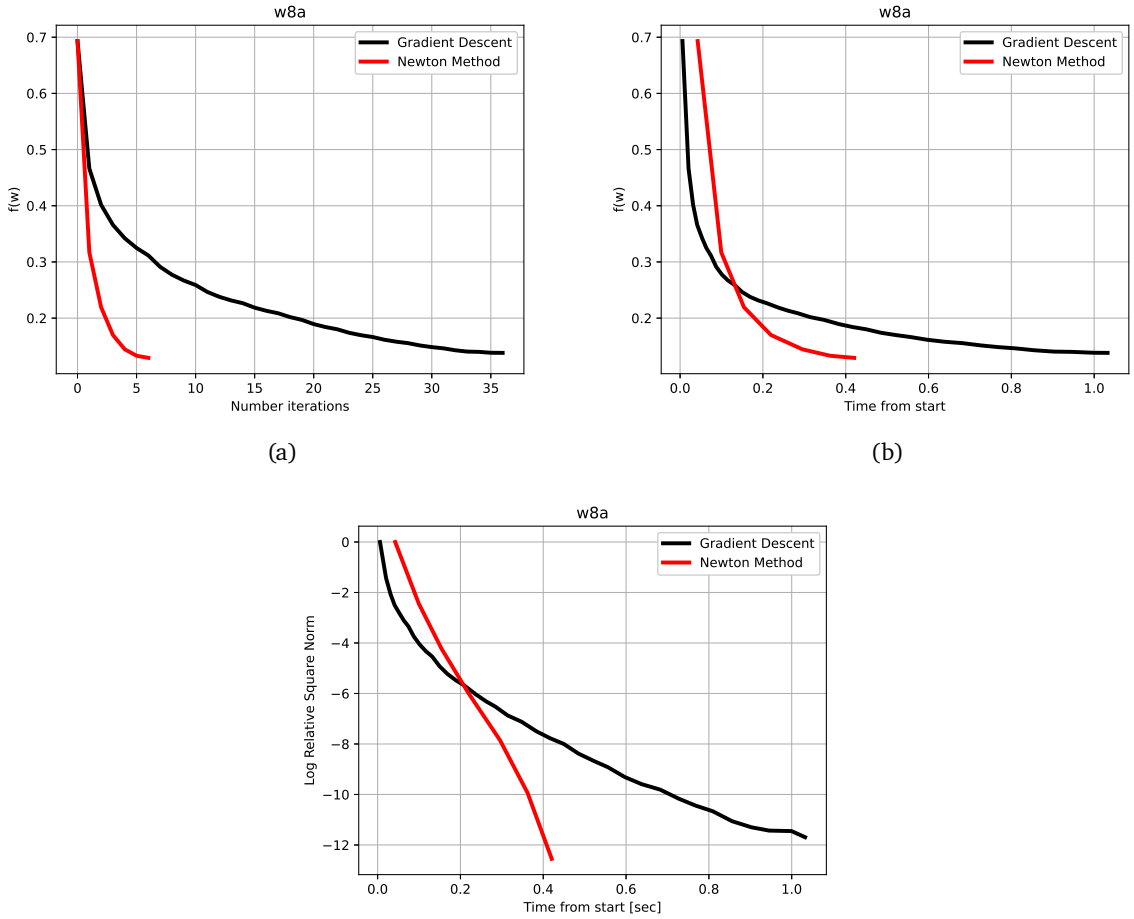
Рис. 6: Заболевания сердца

Посмотрев на Рис.6 заметим, что наша гипотеза подтвердилась на половину — метод Ньютона не всегда ведет себя эффективнее на масштабированных данных. В этом случае, разница между метриками на двух сетах меньше, чем в данных о диабете. Но при этом, у масштабированных данных, точность и f-мера выше, чем у стандартных. В этом случае масштабированные данные более устойчивы к измененим точности приближения.

По итогам эксперимента мы точно установили, что практически все метрики обучения не зависят от точности приближения ϵ, а необходимость масштабирования данных необходимо проверять каждый раз эксперементально.

iv. Сравнение методов на реальной задаче логистической регрессии

В предыдущем эксперименте мы обучались на небольших наборах данных. В каждом было примерно по 800 наблюдений и 15 признаков. Сейчас же мы попробуем увеличить размер выборки как по наблюдениям, так и по признакам. Из-за того, что мой домашний компьютер не может вместить в себя матрицу размером $10^6 \times 10^6$, будем обучаться только на 2 средних сетах. Размер первого будет порядка 50000 наблюдений и 300 признаков, второй имеет размер 72000 на 21000.



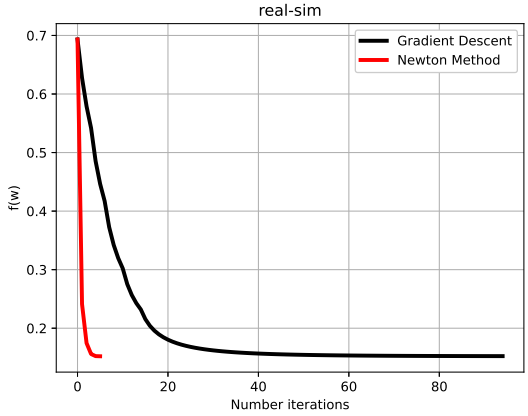
(a) (b)

(c)

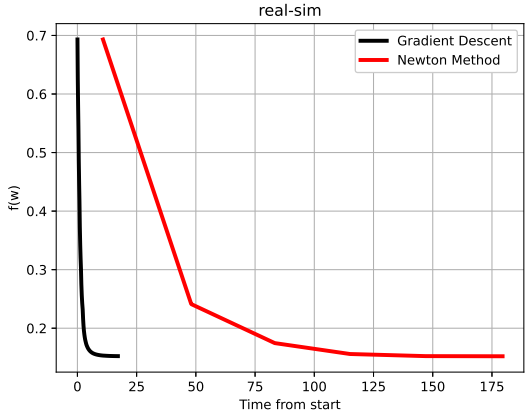
Рис. 7

На Рис.7 использовался набор данных $w8a$. Здесь четко прослеживается превосходство метода Ньютона с точки зрения инженерной задачи — за более меньшее количество итераций и меньшее время мы получаем лучшие результаты, как по функционалу ошибки, так и по логарифму относительной нормы градиента. Метод Ньютона сошелся в 2 раза быстрее и сделал в 7 раз меньше итераций по сравнению с градиентным спуском.

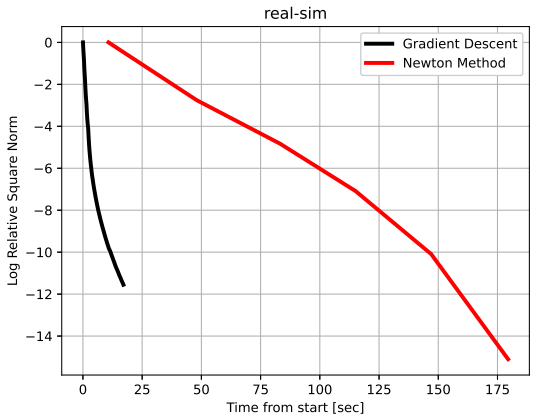
Вполне естественно предположить, что на больших объемах данных у метода Ньютона могут возникнуть вычислительные трудности в операциях разложения Холецкого $A = LL^T$ и решении СЛАУ $Ax = b$.



(a)



(b)



(c)

Рис. 8

Наша гипотеза подтвердилась. На Рис.8 видно, что при размерах матрицы данных порядка 70000×20000 элементов, метод Ньютона сильно проигрывает градиентному спуску. Градиентный спуск находит локальный минимум в 7 раз быстрее, но тратя при этом в 14 раз больше итераций. Также стоит обратить внимание на то, что Рис.8(c) говорит о том, что точность у Ньютоновского метода в 1.5 раза выше, чем у градиентного спуска.

В результате, мы выяснили, что метод Ньютона отлично показывает себя на небольших объемах данных (50000×1000 элементов). При увеличении размерности данных, мы получим более долгое время выполнения алгоритма Ньютона, также нам потребуется очень много памяти для хранения матрицы данных, так как разложение Холецкого и решение СЛАУ занимают $O(n^2)$ памяти.