Отчет по практическому заданию Задание 1. Метрические алгоритмы классификации

Дмитриева Татьяна, 3 курс

Октябрь 2021

Содержание

1 Введение

В данном задании предлагается познакомиться с различными метрическими алгоритмами классификации и методами работы с изображениями. Для этого проводится серия из шести экспериментов с использованием датасета изображений MNIST, который мы разбиваем на обучающую выборку(первые 60 тыс. объектов) и тестовую выборку(10 тыс. последних объектов). Требуется в ходе экспериментов выявить закономерности в полученных данных о работе алгоритмов и сделать выводы о том, каким образом можно их улучшить, чтобы получить более высокий процент правильно предсказанных ответов.

2 Эксперименты

2.1 1 эксперимент

2.1.1 Постановка задачи

В данном эксперименте требуется установить, какой алгоритм поиска ближайших соседей будет работать быстрее. Для этого мы случайным образом выбираем подмножество из 10, 20, 100 признаков, а затем сравниваем время работы, усредненное по трем измерениям, для каждого из алгоритмов во время поиска 5 ближайших соседей на выбранном нами подмножестве признаков обучающей выборки. Для данного эксперимента использовались алгоритмы: my_own, brute, kd_tree, ball_tree.

2.1.2 Результаты

В результате работы данных алгоритмов были получены следующие данные:

Признаки	my_own	brute	kd_tree	ball_tree
10	60.72	11.26	1.68	4.8
20	65.86	11.35	4.13	14.83
100	73.00	14.33	145.76	182.34

2.1.3 Вывод

Таким образом, мы можем заметить, что алгоритм brute оказывается наиболее быстрым, когда мы запускаем его на большом количестве данных. Однако он уступает по времени kd_tree и ball_tree на небольшом количестве данных. Мы можем предположить, что это связано с реализацией данного алгоритма, в которой, вероятно, используются вспомогательные вычисления, увеличивающие его время работы на маленьких данных, но при этом значительно ускоряющие его работу на больших данных.
Если проанализировать время работы алгоритма my_own, он в некотором смысле похож на алгоритм brute: при увеличении в несколько раз объема

данных время увеличивается незначительно по сравнению с алгоритмами kd_tree и ball_tree.

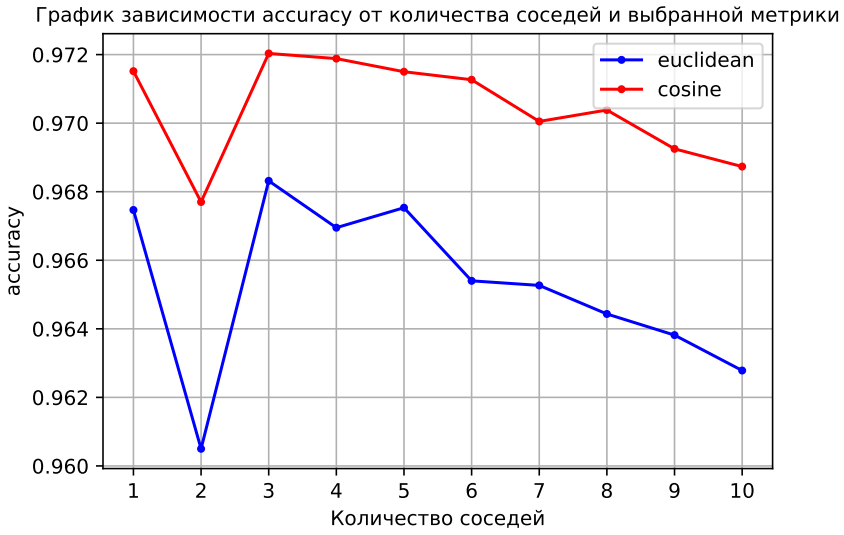
2.2 2 эксперимент

2.2.1 Постановка задачи

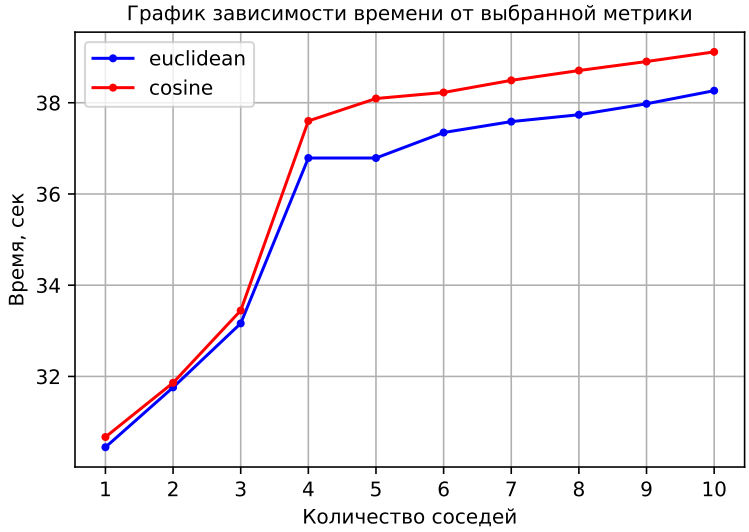
В данном эксперименте требуется оценить по кросс-валидации как точность алгоритма k ближайших соседей зависит от k. Кроме того, мы также должны установить зависимость точности и времени работы алгоритма от того, какую из метрик мы используем при работе алгоритма - евклидову или косинусную. Для того чтобы установить искомые зависимости, мы подсчитаем accuracy для евклидовой и косинусной метрики при k от 1 до 10 с использованием кросс-валидации. Время для исследуемых метрик будем измерять за один проход алгоритма на обучающей и тестовой выборке.

2.2.2 Результаты

После подсчета требуемых данных был построен график зависимости accuracy от количества соседей и выбранной метрики.



Также была исследована зависимость времени работы алгоритма от выбранной метрики.



2.2.3 Вывод

Нетрудно заметить, что точность и время работы k ближайших соседей возрастает при выборе косинусной метрики, а самая большая точность соответствует выбору $k = 3$. Однако следует отметить, что время работы алгоритма возрастает, если мы отказываемся от евклидовой метрики в пользу косинусной.

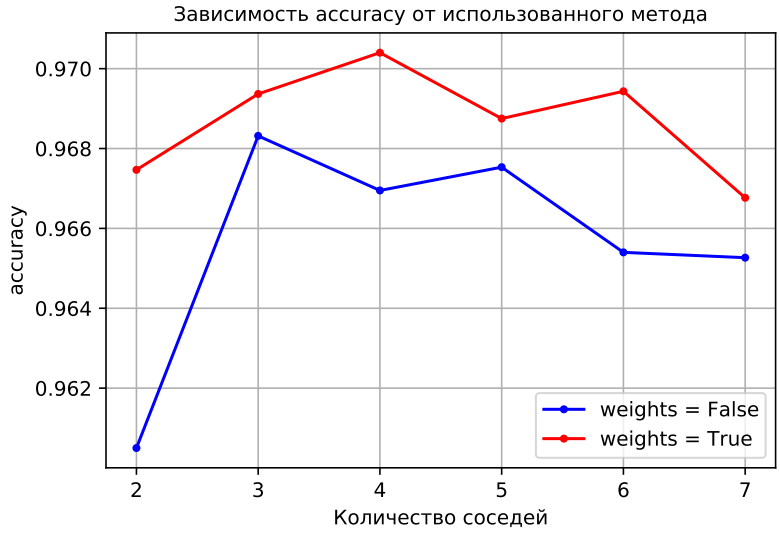
2.3 3 эксперимент

2.3.1 Постановка задачи

В данном эксперименте требуется сравнить метод k ближайших соседей без весов и взвешенный метод, где вес равен $1/(distance + \epsilon)$, где $\epsilon = 10^{-5}$. Для более информативных результатов используется кросс-валидация с теми же параметрами и фолдами для обоих методов.

2.3.2 Результаты

Для исследования различий между этими методами была подсчитана зависимость точности их работы от k, где k от 2 до 7.



2.3.3 Вывод

Внимательно изучив график, можно сделать вывод о том, что взвешенный метод делает более точные прогнозы при классификации объектов. Этот вывод мы могли также получить, предполагая, что близкие к исследуемому объекты чаще принадлежат тому же классу, что и объект, класс которого мы хотим предсказать. Именно поэтому во взвешенном методе предпочтение по прогнозированию класса отдается объектам, наименее удаленным от исследуемого, путем увеличения их веса при голосовании.

2.4 4 эксперимент

2.4.1 Постановка задачи

На основании полученных из предыдущих экспериментов выводов требуется выбрать лучший алгоритм и применить его к исходной обучающей и тестовой выборке, сравнив его точность со значением на кросс-валидации и значением точности лучшего алгоритма, указанного в интернете. Проанализировав матрицу ошибок и несколько визуализированных объектов, класс которых был предсказан неправильно, определить общие черты таких объектов.
Взглянув на предыдущие графики, остановимся на выборе косинусной метрики с 4 соседями, используя взвешенный алгоритм brute.

2.4.2 Результаты

Значения accuracy:
0.9752 - на тестовой выборке
0.9741 - на кросс-валидации
0.9977 - на наилучшем алгоритме

2.4.3 Вывод

Лучший алгоритм из предыдущих экспериментов, как и следовало ожидать, показал себя чуть хуже на кросс-валидации, чем на тестовой выборке, так как кросс-валидация с 3 фолдами усреднила наш результат, приблизив его к реальному значению точности алгоритма. Также использованный в данном эксперименте алгоритм оказался хуже наилучшего алгоритма из интернета, так как структура последнего должна быть во много раз сложнее.
Построенная матрица ошибок явно указывает на то, что алгоритм плохо распознает 7, ошибочно относя их к 1, а также путает 4 с 9 и 3 с 5, 7 с 9. В визуализированных изображениях, на которых были допущены ошибки, можно увидеть небрежное написание цифр, что делает их плохо распознаваемыми даже для человека.

2.5 5 эксперимент

2.5.1 Постановка задачи

В этом эксперименте предлагается размножить обучающую выборку с помощью различных методов работы с изображениями и выбрать из них те, которые могут сделать наш алгоритм более точным. Для этого проверим на кросс-валидации такие параметры, как поворот на 5, 10 или 15 градусов; сдвиг на 1, 2 или 3 пикселя в двумерной размерности; 0,5, 1 или 1,5 - величину дисперсии фильтра Гаусса.
Воспользуемся элементом случайности при выборе направления поворота и направления движения по двумерным осям, для того чтобы сделать выборку более разнообразной и уменьшить вероятность переобучения. Кроме того, вставим преобразованные объекты на четные места, а оригинальные на нечетные места, предполагая, что это увеличит достоверность результатов на кросс-валидации, сделав тестовую выборку как можно более отличающейся от обучающей. После выбора наилучших параметров по accuracy размножим нашу тестовую выборку, увеличивая ее в 4 раза с их помощью, а затем получим новую матрицу ошибок и проанализируем ее.

2.5.2 Результаты

Таблица для поворота

	5°	10°	15°
accuracy	0.97638	0.97575	0.97389

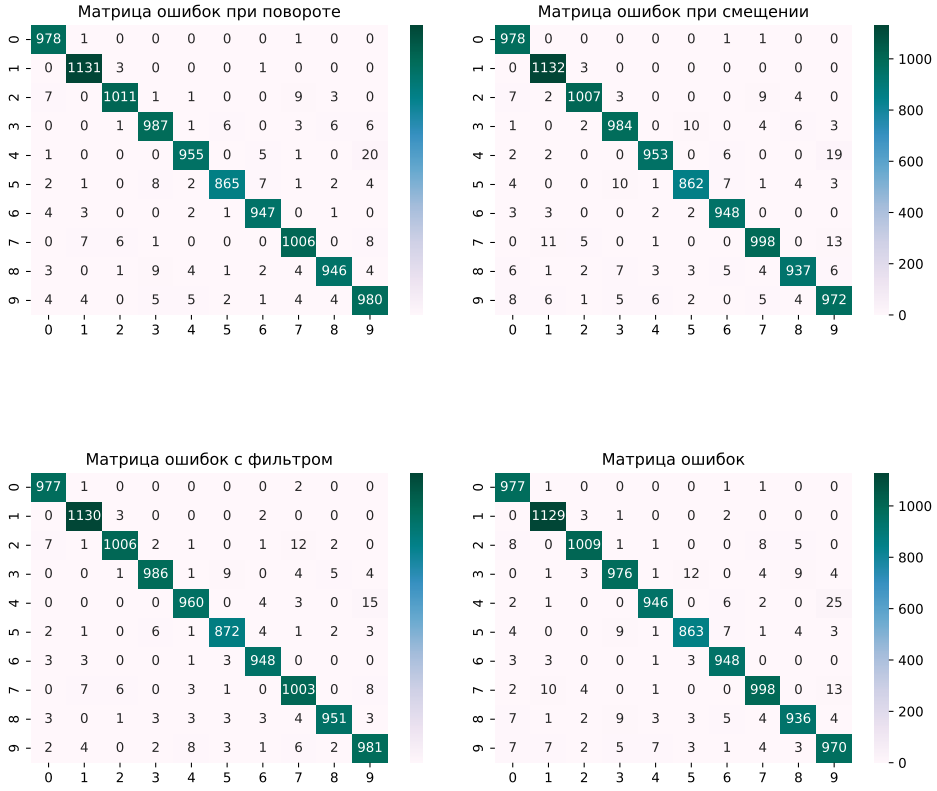
Таблица для смещения

	1 px	2 px	3 px
accuracy	0.96982	0.96723	0.96707

Таблица для фильтра

	0.5	1.0	1.5
accuracy	0.97500	0.97794	0.97537

Подсчитав accuracy для каждого параметра по кросс-валидации, мы можем заметить, что с увеличением угла поворота наша цифра все менее становится похожа на те цифры, которые мы будем подавать в качестве тестовых, поэтому падает и точность алгоритма. Аналогичная ситуация происходит с увеличением смещения цифр. Поэтому выберем в качестве оптимальных параметров угол поворота на 5 градусов и смещение на 1 пиксель.



2.5.3 Вывод

При анализе работы алгоритма на выборке, частично преобразованной с помощью фильтра Гаусса, решено было выбрать параметр truncate=σ*5 для функции ndimage.gaussian_filter, чтобы на нашем изображении все еще угадывалась цифра. Оценивая полученные при тестировании значения, выберем в качестве наилучших параметров поворот в 5 градусов, смещение на 1 пиксель и дисперсию для фильтра Гаусса 1.
Сравнив новую матрицу ошибок и старую матрицу ошибок, можно сделать вывод, что наша новая матрица стала лучше работать с нахождением 4, 3, 8 и 9 и в целом улучшила свои показатели для каждого из чисел. Ее новое значение accuracy оказалось равно 0.9822.

Поворот показал хороший результат для всех цифр кроме 6, с увеличением ее неверного отнесения к 8 и 0. Это может объясняться тем, что 6 близка по написанию к этим цифрам и при повороте становится еще более похожа на них.
Смещение на 1 пиксель в целом улучшило показатели распознавания почти всех цифр, помогая научить алгоритм распознавать смещенные относительно центра изображения, однако немного ухудшило распознавание цифр 2 и 5. Это может быть связано с тем, что цифра 5, вероятно, располагается почти всегда в центре, так как это довольно широкая цифра, поэтому обучение на смещении не приводит к улучшению результата.
Фильтр Гаусса с дисперсией 1 сильно улучшил показатели распознавания всех цифр кроме 1. Это можно связать с тем, что довольно большая часть цифр была написана безотрывно, ухудшая тем самым их распознавание. Однако фильтр Гаусса преобразил часть цифр на обучающей выборке в более смазанные символы, оставляя при этом их форму такой же, в то время как форма 1 пострадала из-за размытия и оказалась менее хорошей для обучения алгоритма.

2.6 6 эксперимент

2.6.1 Постановка задачи

В этом эксперименте требуется размножить тестовую выборку с помощью преобразований из эксперимента 5 и протестировать их на нашем алгоритме. Затем с помощью голосования по предсказаниям на наилучших параметрах предсказать окончательные классы для тестируемых объектов. Для выполнения данной задачи тестовую выборку преобразовали, используя те же вариации параметров, что и в 5 эксперименте. Было получено 9 наборов преобразованных тестовых данных. Далее для нахождения наилучшего сочетания параметров при голосовании были использованы всевозможные комбинации для поворота, смещения и фильтра Гаусса и подсчитано значение accuracy для каждого из них.
В ходе поиска максимума среди этих значений было установлено, что лучшим сочетанием параметров является поворот на 5 градусов и смещение на 1 пиксель со значением accuracy - 0.9787. Для него была построена матрица ошибок.

2.6.2 Результаты

Таблица для отсутствия поворота

	Без фильтра	$\sigma = 0.5$	$\sigma = 1.0$	$\sigma = 1.5$
0 px	0.9752	0.9752	0.9752	0.9752
1 px	0.9752	0.9748	0.9698	0.9637
2 px	0.9752	0.9740	0.9671	0.9485
3 px	0.9752	0.9741	0.9653	0.9431

Таблица при повороте в 5°

	Без фильтра	$\sigma = 0.5$	$\sigma = 1$	$\sigma = 1.5$
0 px	0.9752	0.9759	0.9777	0.9779
1 px	0.9787	0.9782	0.9730	0.9669
2 px	0.9770	0.9762	0.9698	0.9517
3 px	0.9765	0.9755	0.9676	0.9461

Таблица при повороте в 10°

	Без фильтра	$\sigma = 0.5$	$\sigma = 1$	$\sigma = 1.5$
0 px	0.9752	0.9743	0.9718	0.9685
1 px	0.9739	0.9737	0.9674	0.9595
2 px	0.9682	0.9672	0.9619	0.9441
3 px	0.9673	0.9664	0.9580	0.9372

Таблица при повороте в 10°

	Без фильтра	$\sigma = 0.5$	$\sigma = 1$	$\sigma = 1.5$
0 px	0.9752	0.9745	0.9715	0.9662
1 px	0.9705	0.9706	0.9664	0.9555
2 px	0.9625	0.9623	0.9565	0.9372
3 px	0.9576	0.9571	0.9498	0.9283

Сравнивая значение accuracy с использованием нового алгоритма и старого, можно заметить, что качество хоть и улучшилось, но совсем незначительно. Вероятно, это помимо всего прочего связано также с тем, что голосование хорошо работает, когда каждый из комбинируемых методов улучшает распознавание одних и тех же объектов, однако наши параметры улучшают распознавание отдельных объектов, поэтому общая картина оказывается не настолько впечатляющей, как та, что получается в результате использования методов из пятого эксперимента. Кроме того мы не должны исключать вероятность переобучения алгоритма при выполнении пятого эксперимента, что могло также улучшить его точность на кросс-валидации.

3 Вывод

В результате всех проделанных экспериментов было найдено наилучшее соотношение при выборе параметров для метода k ближайших соседей: косинусная метрика с весами при использовании k = 4. Кроме того было найдено оптимальное сочетание для преобразований объектов изучаемого датасета для увеличения точности нашего алгоритма: поворот на 5 градусов, смещение на 1 пиксель и фильтр Гаусса с дисперсией 1. Также при сравнении методов улучшения качества алгоритма было установлено, что алгоритм из 5 эксперимента в сочетании с правильно выбранными параметрами оказывается более качественным, чем алгоритм из эксперимента 4 и алгоритм

из эксперимента 6, однако все еще недостаточно точен, нежели алгоритм, показывающий наилучший результат на анализируемой выборке MNIST.