Московский государственный университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчет по заданию №1

«Метрические алгоритмы классификации»

Выполнила:
студентка 317 группы
Смирнова В. С.

Москва
2023

Содержание

Введение **2**

Результаты экспериментов **2**

Эксперимент 1: Сравнение времени работы алгоритмов 2

Эксперимент 2: Сравнение параметров по кросс-валидации 2

Сравнение точности в зависимости от k 2

Сравнение точности и времени в зависимости от евклидовой или косинусной метрики 3

Эксперимент 3: Сравнение взвешенного knn и без весов 3

Эксперимент 4: Анализ лучшего алгоритма 5

Эксперимент 5: Аугментация обучающей выборки 6

Эксперимент 6: Аугментация тестовой выборки 7

Вывод **8**

Источники **9**

Введение

Данное практическое задание посвящено исследованию алгоритма k ближайших соседей на примере реализации метода и кросс-валидации, исследования его качества и скорости выполнения на датасете MNIST в зависимости от разных параметров.

Результаты экспериментов

В данном разделе приведены результаты экспериментов, теоретические оценки и их сравнения. Эксперименты проводились на датасете MNIST(образцы рукописного написания цифр), данные были разбиты на обучающую выборку(60 000 объектов) и тестовую(10 000 объектов).

Эксперимент 1: Сравнение времени работы алгоритмов

На гистограмме (рис.1) приведены данные о времени нахождения 5 ближайших соседей для каждого объекта тестовой выборки по евклидовой метрике для 4-х алгоритмов - 'kd_tree', 'ball_tree', 'brute' и 'my_own'. Сравнение времени работы алгоритмов происходило в зависимости от размера признакового пространства выборки - 10, 20, 100.

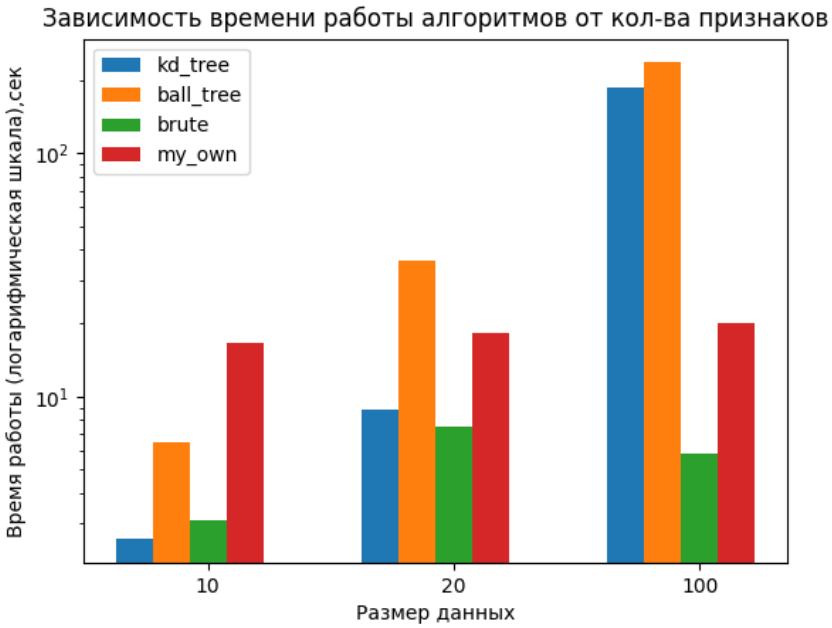

Рис. 1: Зависимость времени работы алгоритмов от кол-ва признаков

Из результатов эксперимента видно, что при малом признаковом пространстве наиболее быстрыми являются алгоритмы kd_tree и brute, однако время работы первого значительно увеличивается при возрастании числа признаков, в то время как brute продолжает оставаться самым быстрым алгоритмом среди рассмотренных.

Эксперимент 2: Сравнение параметров по кросс-валидации

Сравнение точности в зависимости от k

В этой части эксперимента проводилось сравнение точности невзвешенного алгоритма k ближайших соседей brute для евклидовой и косинусной метрики по кросс-валидации с 3 фолдами. На рис.2 представлены результаты.

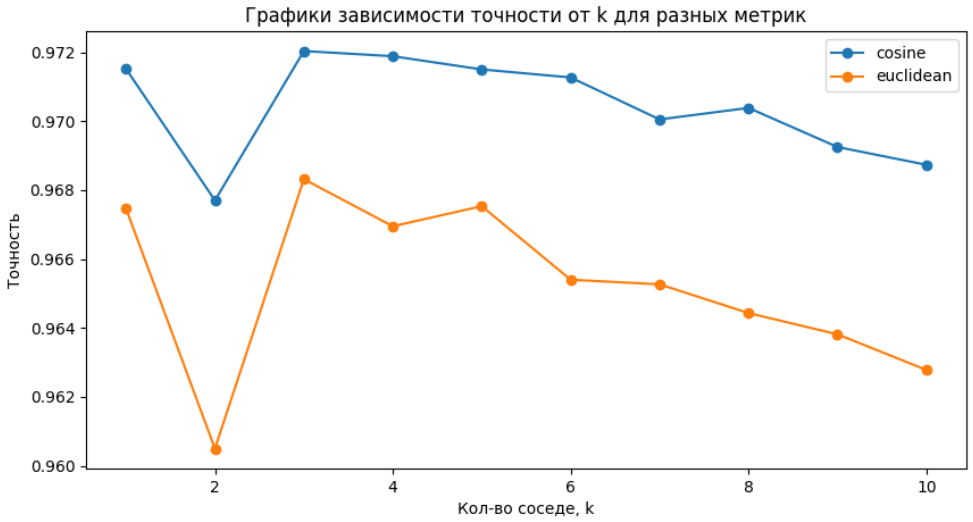

Рис. 2: Зависимости точности от k для разных метрик

Как можно заметить из гистограммы, наилучшая точность достигается при косинусной метрики и k=4 соседях. При k=2 наблюдается резкий скачок вниз, что возможно при переобучении модели, так как 2 ближайших соседа могут иметь мало различий в пикселях, но являются изображениями разных цифр; также такое может происходить из-за специфики датасета, так как он содержит разные стили написания цифр, и такой резкий скачок вниз может свидетельствовать о наличии выбросов.

Сравнение точности и времени в зависимости от евклидовой или косинусной метрики

Ниже(табл.1) представлены данные о работе двух алгоритмов brute и my_own для для k=5 каждой из метрик, вычисление времени и точности производилось по кросс-валидации, как и в предыдущей части эксперимента.

	brute	my_own
Время	208.8803	204.3835
Точность	0.9715	0.9715

(a) Косинусная метрика

	brute	my_own
Время	145.9315	197.3741
Точность	0.9675	0.9675

(b) Евклидовая метрика

Таблица 1: Сравнение работы алгоритмов my_own и brute в зависимости от метрики

Исходя из полученных данных, можно сделать выводы, что алгоритмы работают быстрее на евклидовой метрики, так как косинусная требует более сложных вычислений(из реализации), однако точность на косинусной получается выше, поскольку она устойчива к масштабированию векторов(например, вектор интенсивности), чувствительна к ориентации и направлению.

Эксперимент 3: Сравнение взвешенного knn и без весов

В этой части задания сравнивались взвешенный метод KNN, где голос объекта равен $1/(distance + \epsilon)$, с методом без весов, аналогично эксперименту 2.

Из графика зависимости точности от k для разных метрик для двух типов методов(рис.3) следует, что взвешенный KNN более точен, и наибольшая точность достигается при k=4 соседях и косинусной метрике.

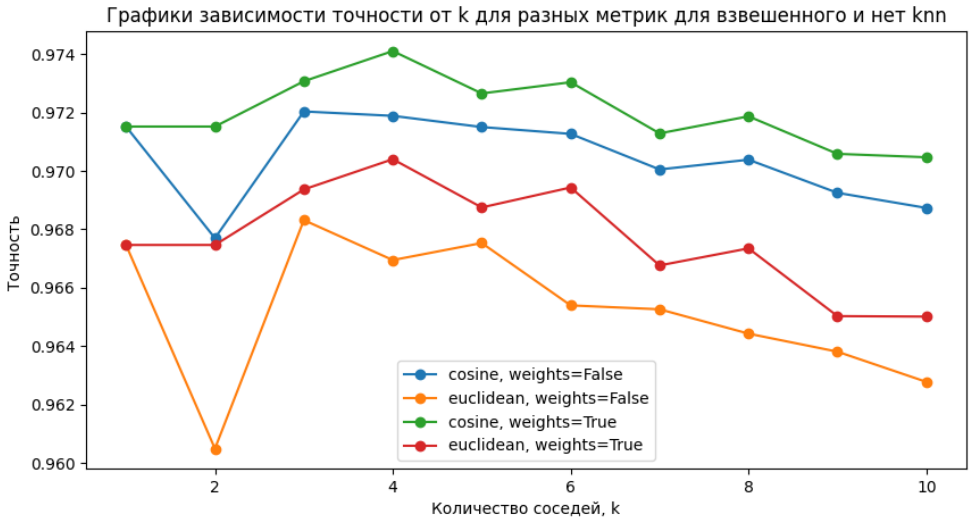

Рис. 3: Зависимость точности от k для разных метрик и взвешенного и нет метода

Также выполнено сравнение времени работы взвешенного KNN с результатами эксперимента 2 при тех же параметрах(рис.4). Из гистограммы следует, что, несмотря на дополнительное вычисление расстояния, взвешенный метод работает не на много дольше метода без весов.

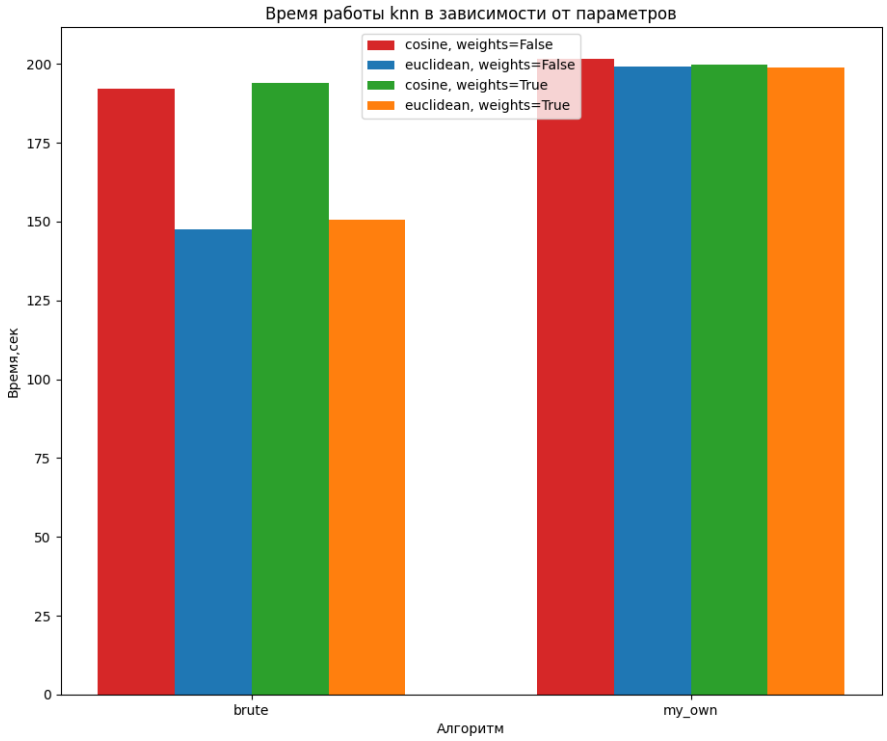

Рис. 4: Время работы KNN в зависимости от параметрок

Исходя из точности и времени работы, в следующих экспериментах в качестве лучшего метода выбран - взвешенный KNN с алгоритмом brute, k=4 соседями и косинусной метрикой.

Эксперимент 4: Анализ лучшего алгоритма

В данной части выполнен анализ работы лучшего алгоритма, сравнение точности и диагностика ошибок. Исходя из того, что точность на тестовой выборке практически совпадает с

	Кросс-валидация	Тестовая выборка	Интернет
Точность	0.9754	0.9752	0.9979

Таблица 2: Точность лучшего алгоритма

точностью по кросс-валидации следует, что наша модель устойчива и хорошо обобщает данные, тем не менее ее точность меньше точности, найденной в интернете. Рассмотрим матрицу ошибок(табл.3), из нее следует, что хуже всего метод отличает 3 от 5, 4 от 9, 7 от 1,7 от 9, 8 от 3.

Цифра	0	1	2	3	4	5	6	7	8	9
0	977	1	0	0	0	0	1	1	0	0
1	0	1129	3	1	0	0	2	0	0	0
2	8	0	1009	1	1	0	0	8	5	0
3	0	1	3	976	1	12	0	4	9	4
4	2	1	0	0	946	0	6	2	0	25
5	4	0	0	9	1	863	7	1	4	3
6	3	3	0	0	1	3	948	0	0	0
7	2	10	4	0	1	0	0	998	0	13
8	7	1	2	9	3	3	5	4	936	4
9	7	7	2	5	7	3	1	4	3	970

Таблица 3: Матрица ошибок

Исходя из вида объектов, на которых были сделаны ошибки, можно сделать вывод, что

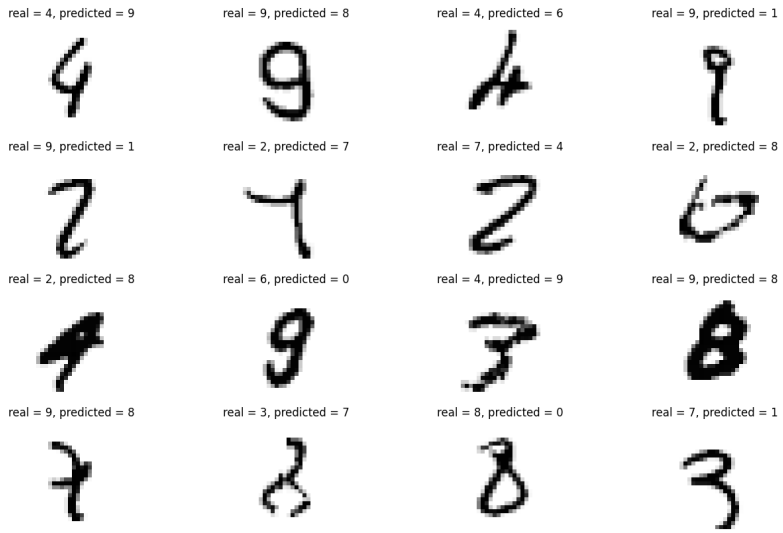

Рис. 5: Объекты, на которых алгоритм ошибся

это связано с особенностями подчерка людей: например, близкое расположение элементов написанных цифр, их сильные или слабые изгибы.

Эксперимент 5: Аугментация обучающей выборки

В этом эксперименте выполнена аугментация обучающей выборки разными преобразованиями(табл.4), оценка качества производилась по кросс-валидации с 3 фолдами. В целях сокращения времени подбора параметров, аугментация проводилась на 3-х случайно выбранных подмножествах элементов из обучающей выборки размера 21 тыс.

Рассмотрим результаты преобразований(поворот, фильтр Гаусса, морфологические операции):

Преобразование	Номер выборки	Точность	Изнач. точность
Поворот $\alpha = 5°$	I	0.96876	0.96862
Поворот $\alpha = 10°$	II	0.96752	0.96748
Поворот $\alpha = 15°$	III	0.96929	0.96927
Фильтр Гаусса, $\sigma = 0.5$	I	0.96914	0.96862
Фильтр Гаусса, $\sigma = 1$	II	0.96252	0.96748
Фильтр Гаусса, $\sigma = 1.5$	III	0.96547	0.96927
Эрозия, ядро 2	I	0.96871	0.96862
Дилатация, ядро 2	I	0.96338	0.96862
Открытие, ядро 2	I	0.96795	0.96862
Закрытие, ядро 2	I	0.94691	0.96862

Таблица 4: Точность по кросс-валидации при преобразованиях

Помимо приведенных выше преобразований, были выполнены смещения на 1,2,3 пикселя по каждой из двух размерностей по всевозможным комбинациям и получены следующие результаты(рис.6):

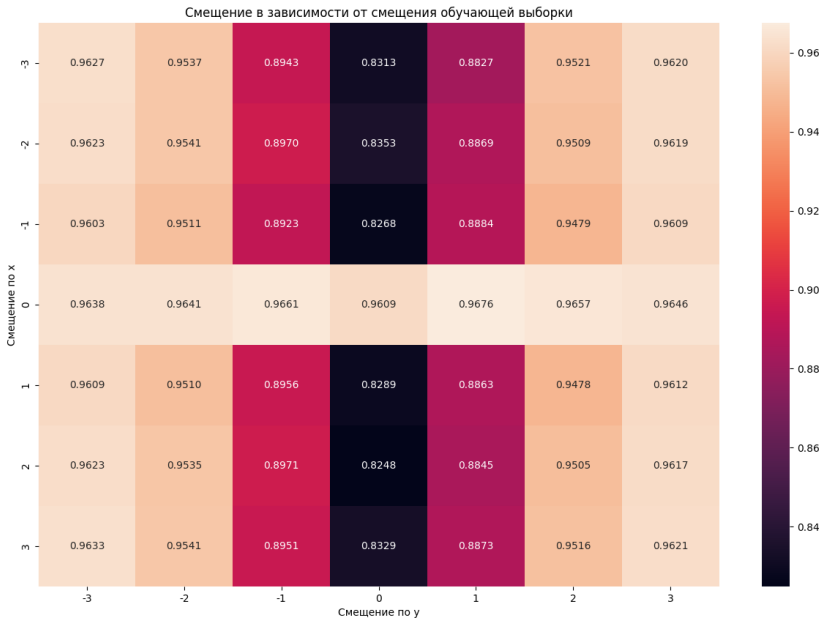

Рис. 6: Точность в зависимости от смещения: вертикально - смещения по x, горизонтально - по y

Таким образом для аугментации обучающей выборки были выбраны 2 преобразования: сдвиг по оси y на 1 пиксел и применение фильтра Гаусса с дисперсией 0.5, так как они показали лучшие результаты в эксперименте.

После преобразований *точность составила 0.9765*. Из разницы матриц ошибок до и после преобразований(табл.5) следует, что модель стала лучше распознавать цифры 0, 1, 3, 4, 6, 8, а также отличать 2 от 8, 3 от 8, 3 от 5, 4 от 9, 7 от 9, 9 от 0, но стала чаще неверно распознавать 7 от 1, 9 от 7.

Цифра	0	1	2	3	4	5	6	7	8	9
0	-1	1	0	0	0	0	0	0	0	0
1	0	-1	0	0	0	0	1	0	0	0
2	1	-1	0	0	0	0	-2	-1	3	0
3	0	1	2	-12	1	4	0	-1	4	1
4	1	-1	0	0	-3	0	1	0	-1	3
5	2	-1	0	-2	0	2	1	-1	-1	0
6	0	1	0	0	-1	1	-1	0	0	0
7	0	-7	0	0	-1	0	0	4	0	4
8	1	1	-1	-2	2	-1	1	0	-1	0
9	3	2	1	2	-1	-1	0	-5	-1	0

Таблица 5: Изменение матрицы ошибок после итоговой аугментации обучающей выборки

Эксперимент 6: Аугментация тестовой выборки

В этой части задания выполнена аугментация тестовой выборки различными преобразованиями:

1. поворот на 5,10,15 в обе стороны
2. смещения на (0, 1) и (1,0), (0, -1) и (-1,0), (0,2) и (2,0), (0,-2) и (-2,0), (0,3) и (3,0), (0,-3) и (-3,0)
3. фильтр Гаусса с дисперсией 0.5, 1, 1.5
4. Морфологические операции: эрозия, дилатация, открытие, закрытие с ядром 2

Результат был получен путем голосования среди соответствующих элементов аугментированной тестовой выборки(табл.6).

Голосование было осуществлено следующим образом(пример для голосования среди элементов тестовой выборки, увеличенной в 3 раза):

```
...
augmentated_pred = model.predict(augmentated_X_test)
pred = np.zeros(X_test.shape[0], dtype=type(augmentated_pred))
fst = augmentated_pred[:X_test.shape[0]]
scnd = augmentated_pred[X_test.shape[0]: 2 * X_test.shape[0]]
thrd = augmentated_pred[2 * X_test.shape[0]:]
mask1 = (fst == scnd) | (fst == thrd)
mask2 = scnd == thrd
y_pred = np.where(mask1, fst, np.where(mask2, scnd, thrd))
...
```

Преобразование	Точность
Поворот $\alpha = 5°$	0.9749
Поворот $\alpha = 10°$	0.976
Поворот $\alpha = 15°$	0.9729
Фильтр $\sigma = [0.5, 1, 1.5]$	0.9644
Смещение, (0,1) и (1,0)	0.9495
Смещение, (0,-1) и (-1,0)	0.9442
Смещение, (0,2) и (2,0)	0.8417
Смещение, (0,-2) и (-2,0)	0.8295
Смещение, (0,3) и (3,0)	0.6084
Смещение, (0,-3) и (-3,0)	0.5478
Морфологические операции	0.7075

Таблица 6: Точность при аугментации тестовой выборки

При аугментации тестовой выборки *точность модели увеличивается (до 0.976)* только при повороте на 10°, в остальных случаях - она уменьшается по сравнению с *изначальным значением 0.9752*. А значит, аугментация обучающей выборки для нашего алгоритма дает большее улучшение.
Рассмотрим изменение матрицы ошибок до и после поворота на 10° тестовой выборки(табл.7):

Цифра	0	1	2	3	4	5	6	7	8	9
0	-1	0	0	0	0	0	1	0	0	0
1	0	1	0	-1	0	0	0	0	0	0
2	-2	0	2	0	0	0	0	-2	2	0
3	-1	1	0	-8	0	6	0	0	2	0
4	0	-2	0	0	5	0	0	0	0	-3
5	-1	-1	0	2	0	-2	0	0	2	0
6	-2	0	0	0	0	3	0	0	-1	0
7	-1	4	-4	0	0	0	0	0	0	1
8	1	0	-1	4	0	-1	1	1	-6	1
9	-1	0	0	1	0	1	0	0	-2	1

Таблица 7: Изменение матрицы ошибок после поворота на 10° тестовой выборки

Из табл.7 следует, что модель стала лучше распознавать цифры 0, 3, 5, 8, а также отличать 2 от 8, 3 от 5, 3 от 8, 5 от 8, 6 от 5, 7 от 1, но стала чаще неверно распознавать 4 от 9, 7 от 2.

Вывод

В результате проведенных экспериментов было установлено, что метод взвешенных 4-х ближайших соседей с алгоритмом brute и косинусной метрикой показал самую высокую точность 0.9765 на датасете MNIST при аугментации обучающей выборки. Однако данное значение оказалось меньше, чем найденное в интернете.

Источники

1. https://www.kaggle.com/code/cdeotte/mnist-perfect-100-using-knn/notebook