Отчет о практическом задании «Метрические алгоритмы классификации».

Практикум 317 группы, ММП ВМК МГУ.

Павленко Иван Александрович.

Октябрь 2024.

Содержание

1 Введение **2**
- 1.1 Пояснения к задаче . . . 2

2 Эксперименты **2**
- 2.1 Исследование скорости алгоритма на разных способах реализации . . . 2
- 2.2 Кросс-валидация для исследования эффективности и точности метода на разных значениях параметра k и разных метриках поиска расстояний . . . 3
- 2.3 Исследование алгоритма с использованием весов и без на разных значениях параметра и метрики . . . 4
- 2.4 Проверка лучших значений и анализ ошибок . . . 5
- 2.5 Аугментация обучающей выборки . . . 7
- 2.6 Аугментация тестовой выборки . . . 10

3 Выводы **10**

1 Введение

Целью данного задания было тестирование одного из метрических алгоритмов классификации - метода k ближайших соседей (KNN) на наборе данных MNIST для классификации рукописных цифр. Рассматривались разные стратегии поиска соседей, параметры (количество соседей), а также разные преобразования выборки для демонстрации их влияния на временную эффективность алгоритма и на его точность.

1.1 Пояснения к задаче

Метод KNN является методом "ленивого обучения" поскольку перед применением (почти всегда) не требует предварительных подсчётов на обучающей выборке. Отсюда следует и его неизбежный недостаток - каждое тестовое применение требует больших вычислений, то есть подсчёта расстояний от всех тестовых объектов до всех объектов обучающей выборки. Чтобы облегчить нагрузки на оперативную память, всюду при применении метода расстояния находились последовательно для блоков тестовой выборки размера 1000. Общие размеры обучающей и тестовой выборок - 60000 и 10000 объектов соответственно.

2 Эксперименты

2.1 Исследование скорости алгоритма на разных способах реализации

В данном эксперименте проводился анализ различных методов ускорения поиска k ближайших соседей при фиксированном значении $k = 5$ и различных количествах признаков, выбранных из изображений (n_feat). Рассматривались наборы из 10, 20 и 100 признаков, выделенных случайно из всех объектов (то есть для всех объектов был выделен один и тот же набор).

Рассмотренные методы:

- K-мерное дерево (kd-Tree) (встроено в sklearn)
- Ball-Tree (встроено в sklearn)
- Brute Force (встроенный в sklearn)
- Собственный метод (own), основан на подсчёте всех расстояний между обучающими и тестовыми объектами в матричном виде.

n_feat \ Метод	kd-Tree	ball-Tree	brute	own
10	0.953566	3.310309	6.435393	52.189703
20	1.52265	12.668767	5.914864	56.454598
100	92.06085	85.09427	6.302815	155.704806

Таблица 1: Время (в секундах) поиска соседей при различных параметрах и методах

Из таблицы 1 можно видеть, что на маленьких количествах признаков kd-tree показывает хорошие результаты благодаря эффективному способу хранения данных. Однако при значительном увеличении их числа стратегия brute сильно выигрывает у всех остальных. Это объясняется тем, что kd-tree и ball-tree проявляют всю свою эффективность на данных больших размерностей, тогда как в исследуемой задаче все объекты представлены одномерно. Собственная реализация сильно проигрывает встроенному в sklearn брут-форсу, поэтому в дальнейшем в экспериментах использовалась только стратегия brute.

2.2 Кросс-валидация для исследования эффективности и точности метода на разных значениях параметра k и разных метриках поиска расстояний

В данном эксперименте исследовалось поведение алгоритма при значениях параметра k от 1 до 10 при подсчёте расстояний с помощью евклидовой и косинусной метрики. Точность и время считались на кросс-валидации с 3-мя фолдами на обучающей выборке. Точность и время усреднялись по всем фолдам. Результаты представлены на графиках 1, 2.

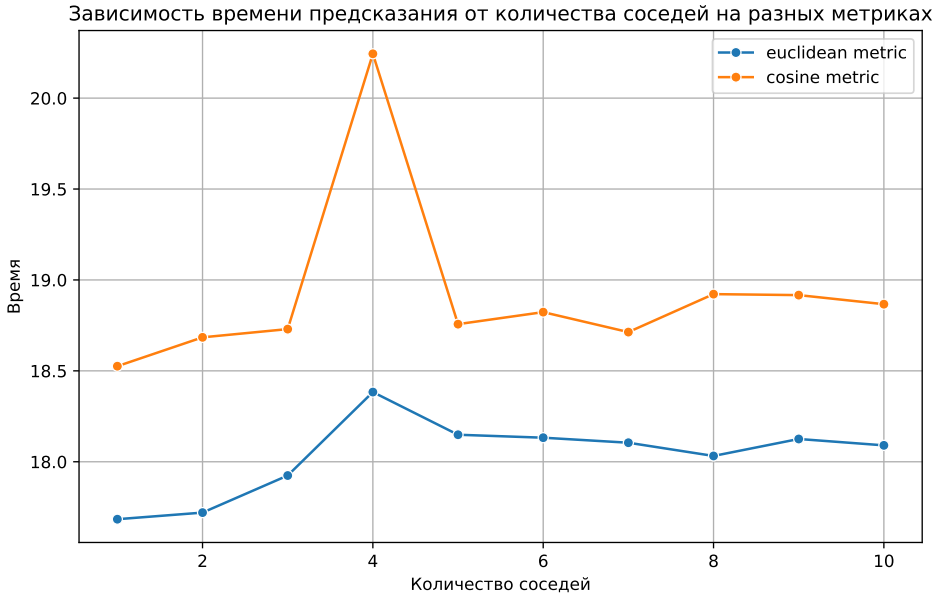

Рис. 1: Зависимость времени выполнения (в секундах) алгоритма от количества соседей на разных метриках

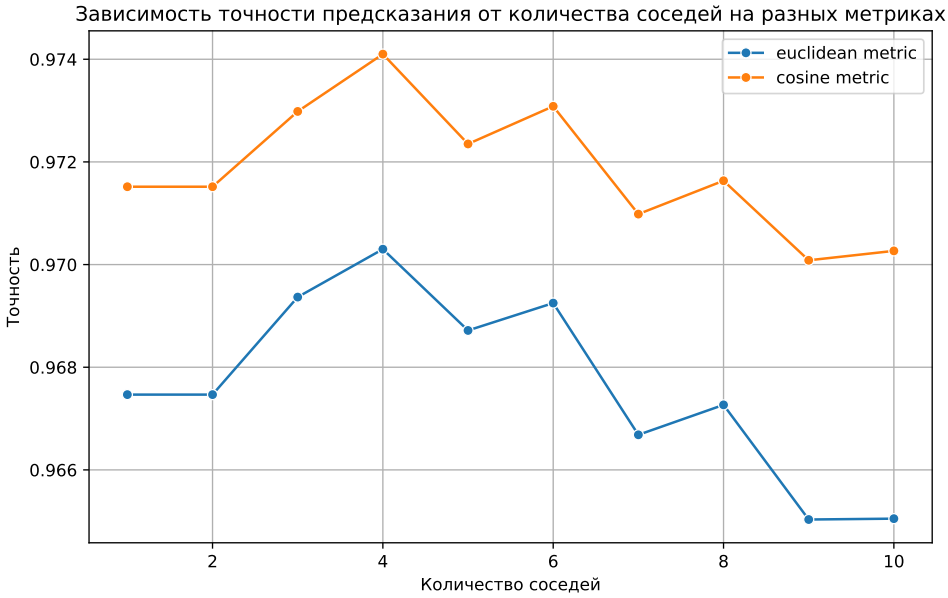

Рис. 2: Зависимость точности алгоритма от количества соседей на разных метриках

Видно, что косинусная метрика проигрывает евклидовой во времени, но серьёзно выигрывает в качестве.

Разница во времени обусловлена особенностями технической реализации (подсчёт угла может быть вычислительно чуть сложнее подсчёта расстояния).

Разница в качестве может быть вызвана тем, что косинусное расстояние лучше уменьшается при схожести форм объектов на изображении, в то время как евклидово может сильно увеличиваться из-за небольшой разницы в положении объекта. Также причиной может быть то, что косинусная метрика не зависит от абсолютного значения признаков (яркости пикселей), то есть будет определять близкими изображения одной формы с разной яркостью, в то время как евклидово расстояние может сильно увеличиваться лишь от разницы абсолютных значений.

На графике точности предсказания нет падения на значении $k = 2$, и других чётных k, которое можно было бы ожидать из-за случайности выбора из двух ближайших соседей, поскольку, из-за особенностей реализации, из ближайших соседей для предсказания выбирается самый близкий к исследуемому объкту. В то же время видно, что при $k \geq 3$ чётные значения показывают себя лучше нечётных. $k = 4$ - оптимальное значение параметра.

2.3 Исследование алгоритма с использованием весов и без на разных значениях параметра и метрики

В этом эксперименте провелись вычисления, аналогичные предыдущему, но с использованием взвешенного алгоритма, где при выборе значения таргета каждый из k соседей имеет вес, равный $\frac{1}{distance+\varepsilon}$, где *distance* - расстояние от соседа до исследуемого объекта, $\varepsilon = 10^{-5}$. Исследовалась тоько точность. Результаты представлены на графиках 3, 4.

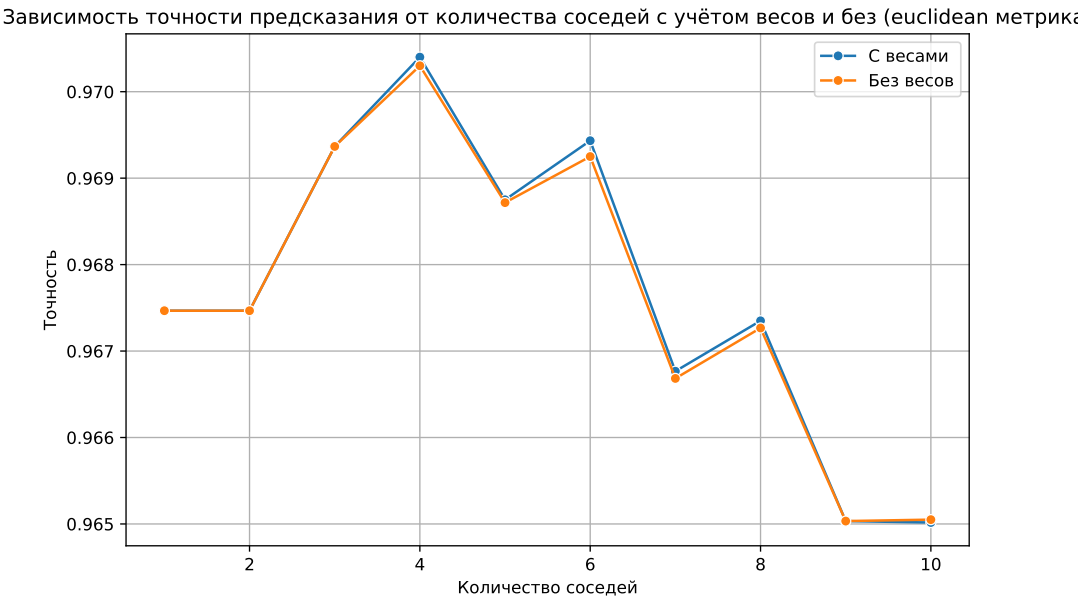

Рис. 3: Зависимость точности от количества соседей с евклидовой метрикой, для взвешенного и невзвешенного алгоритма

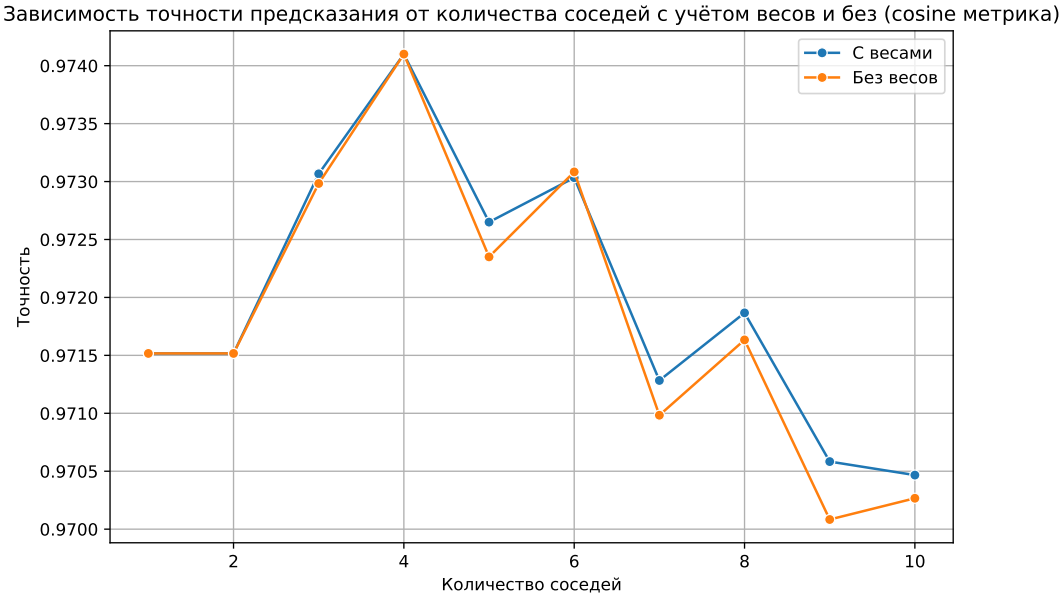

Рис. 4: Зависимость точности от количества соседей с косинусной метрикой, для взвешенного и невзвешенного алгоритма

Видно, что взвешенный метод показал себя лучше на обеих метриках (но не очень значительно), однако лучшее значение точности (косинусная метрика, $k = 4$) одинаково с весами и без них, поэтому для упрощения вычислений в дальнейшем будут использоваться именно эти параметры, без учёта весов.

2.4 Проверка лучших значений и анализ ошибок

В этом эксперименте была подсчитана точность алгоритма при лучших параметрах на тестовой выборке (2)

	Test	Cross-Validation
Точность	0.9759	0.9741

Таблица 2: Точность на тесте и кросс-валидации

Точность лучших алгоритмов на наборе данных MNIST без использования свёрточных сетей превышают 98%, с их использованием - 99%. Был проведён анализ ошибок, на рис. 5 можно видеть матрицу ошибок для сделанного предсказания на тестовой выборке. Значения количества ошибок были прологарифмированы для наглядности и простоты анализа матрицы:

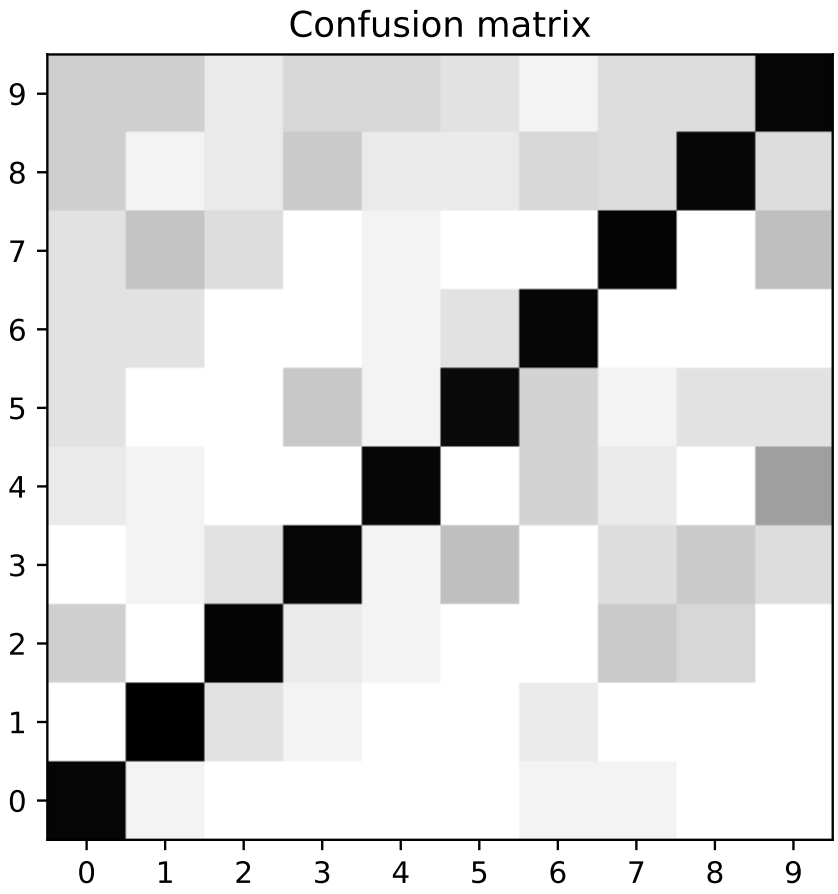

Рис. 5: Матрица ошибок для алгоритма на тестовой выборке

Также на рисунке 6 были представлены примеры объектов, на которых была допущена ошибка.

Рис. 6: Примеры объектов, на которых ошибся алгоритм

По матрице видны числа, которые легко путает алгоритм. Самой распространённой ошибкой был ответ 9 на цифре 4. Это можно легко объяснить тем, что в наборе данных, как видно на 6, много нечётко написанных цифр, и цифра 4 может быть похожа на 9 при определённом написании. По аналогичным причинам на матрице выделяются отдельные пары цифр, в написании похожих.

Объекты, на которых произошла ошибка, выделяются либо нечётким написанием, либо сильным поворотом, либо необычной формой цифры.

2.5 Аугментация обучающей выборки

В этом эксперименте на кросс-валидации с тремя фолдами проводилась оценка точности с аугментированной выборкой. Размер дополнения к выборке во всех экспериментах составил 30000 объектов (случайных), к ним были применены одинаковые преобразования. Были проверены такие преобразования, как:

- Повороты изображений в обе стороны на 5, 10 и 15 градусов
- Сдвиги изображений по обоим осям в обе стороны на 1, 2 и 3 пикселя
- Комбинации сдвигов по осям
- Комбинация сдвига по x и поворота
- Фильтр гаусса с ядром размера 5 и дисперсиями 0.5, 1, 1.5
- Эррозия с ядром 2X2
- Дилатация с ядром 2X2
- Открытие и закрытие с ядром 2X2
- Сочетание эррозии с гауссовым фильтром

Ниже приведены таблицы с точностями на всех экспериментах.

Угол поворота	-15	-10	-5	5	10	15
Точность	0.9655	0.9787	0.9849	0.9845	0.9785	0.9656

Таблица 3: Точности на аугментированной выборке с разными углами

Сдвиг по x	-3	-2	-1	1	2	3
Точность	0.8251	0.9221	0.9770	0.9779	0.9259	0.8320

Таблица 4: Точности на аугментированной выборке с разными сдвигами по оси x

Сдвиг по y	-3	-2	-1	1	2	3
Точность	0.8074	0.9131	0.9753	0.9753	0.9120	0.8434

Таблица 5: Точности на аугментированной выборке с разными сдвигами по оси y

Дисперсия	0.5	1	1.5
Точность	0.9853	0.9845	0.9774

Таблица 6: Точности на аугментированной выборке с фильтром Гаусса с разными дисперсиями

	Эррозия	Эррозия + фильтр Гаусса	Дилатация	Дилатация + фильтр Гаусса
Точность	0.9743	0.9775	0.9783	0.9769

Таблица 7: Точности на аугментированной выборке с морфологическими операциями

Результаты некоторых экспериментов не приведены в виде таблицы из-за того, что использованные преобразования не были включены в итоговую аугментированную выборку. Далее приведены значения этих преобразований на при лучших параметрах:

- Смещения по x и y на 1 пиксель в случайном направлении (сдвиг на более чем один пиксель показал бы ещё менее удовлетворительные результаты, поскольку такие сдвиги даже по одной из осей приводят к потери точности даже): 0.9567
- Смещение по оси x и поворот (лучшие параметры: $angle = 5, x\_shift = 1$): 0.9756
- Открытие: 0.9406
- Закрытие: 0.9452

Было принято решение в соответствии с полученными результатами аугментировать выборку таким образом: количество новых объектов было выбрано 120000. Из них 60000 разных изображений были получены из обучающей выборки для сдвигов и поворотов, и 60000 - для фильтра Гаусса и морфологических операций.

Были выбраны только те преобразования, значение точности по кросс-валидации на которых превышало значение точности на начальной кросс-валидации:

- Повороты: -5, 5, -10, 10
- Сдвиги по осям (отдельно): -1, 1
- Эррозия + фильтр гаусса
- Дилатация

Конечное значение точности - 0.9799.

На рисунке 7 показана матрица ошибок для предсказания на аугментированной выборке и матрица до аугментации. Видно, что в основном аугментация помогла исправить ошибки на цифрах 0 и 6. В этом могли помочь повороты, а так же дилатация или эррозия с гауссовым фильтром, которые помогают более разнообразно выделять границы цифр, делать их более различимыми. Также аугментация в целом немного уменьшила число ошибок на некоторых цифрах. Аугментация не смогла устранить ошибки на цифрах, написанных неразборчиво, как на рисунке 6.

В таблице 8 приведены точности при последовательном применении метода к выборке, аугментированной различными преобразовании. Итоговая точность увеличилась на 0.0040, то есть на аугментированной выборке было правильно предсказано на 40 больше тестовых объектов.

Выборка	Точность
Повороты на 5	0.9752
+ повороты на 10	0.9777
+ сдвиги по x	0.9780
+ сдвиги по y	0.9780
+ эррозия с фильтром Гаусса	0.9783
+ дилатация	0.9799

Таблица 8: Изменение точности при последовательной аугментации

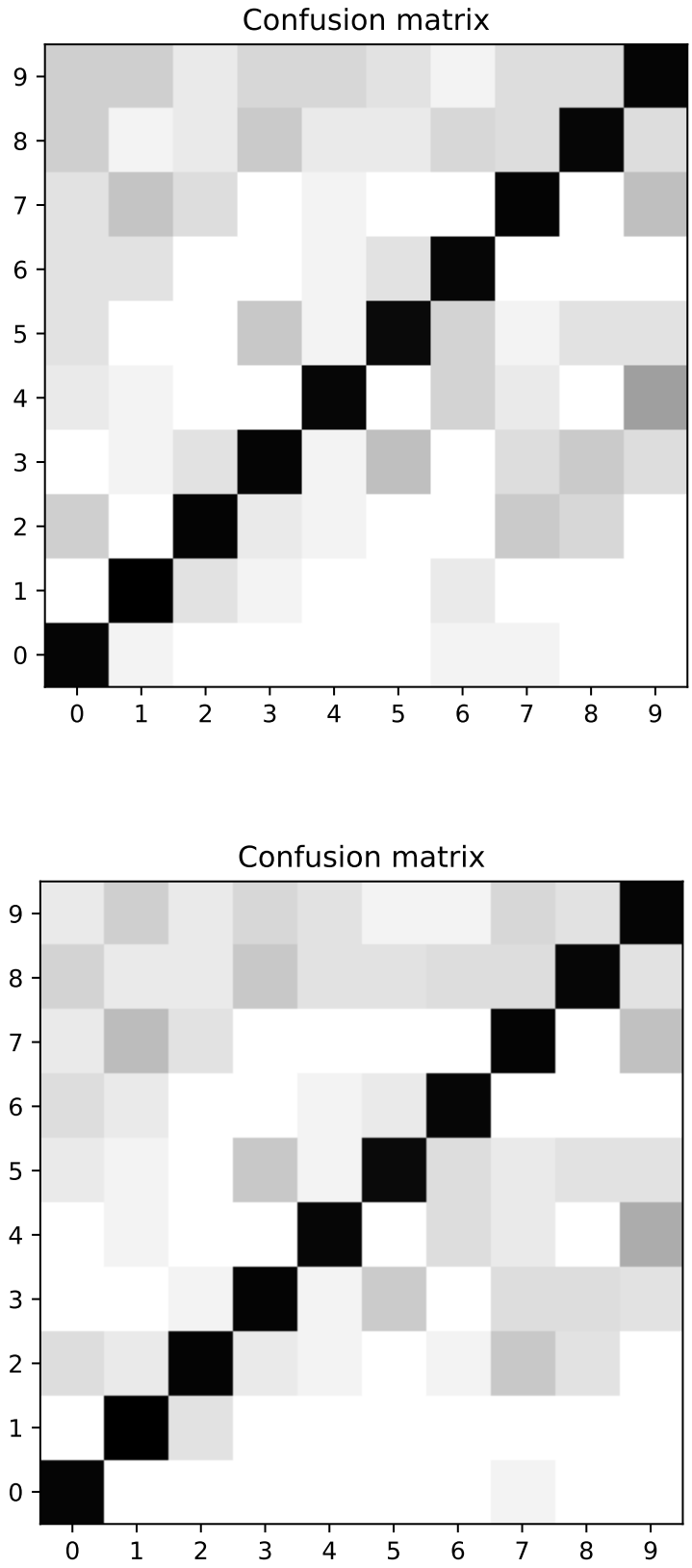

Рис. 7: Матрица ошибок до и после аугментации

2.6 Аугментация тестовой выборки

В данном разделе проводился эксперимент, в котором предсказания делались на начальной обучающей выборке, а соседи искались по нескольким тестовым, аугментированным указанными в предыдущем разделе преобразованиями. Результат на объекте выбирался путём голосования. На рисунке 8 показана матрица ошибок. Значение точности при таком методе получилось 0.9771. Число ошибок уменьшилось по сравнению с начальным благодаря тому, что изменения тестовой выборки могли уменьшить число случайных предсказаний на начальном тесте при повёрнутых или размытых изображениях.

Отличие этого эксперимента от предыдущего в том, что если тогда была сделана попытка расширить выборку, чтобы для каждого тестового объекта с большей вероятностью был найден подходящий сосед, то в данном случае цель заключается в подборе уже тестовых данных под обучающую выборку. Этот эксперимент позволяет увидеть работу метода на более широком наборе данных.

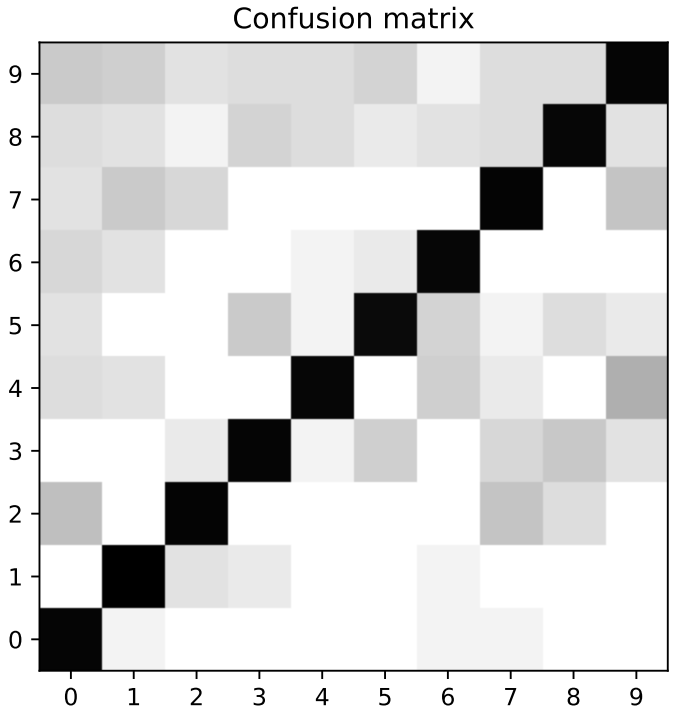

Рис. 8: Матрица ошибок при аугментированной тестовой выборке

3 Выводы

Проведённая работа продемонстрировала различные подходы к улучшению применения метода KNN, такие как:

- Выбор подходящего метода поиска соседей соразмерно виду данных
- Выбор гиперпараметра и метрики с помощью кросс-валидации
- Использование весов в поиске соседей
- Различные способы аугментации выборки и оценка их влияния
- Рассмотрение результатов метода на расширенных тестовых данных

Исследование такого простого набора данных показывает, как разнообразно можно подходить к работе с самыми простыми методами метрической классификации, такими как KNN.