Московский государственный университет имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

# Практикум на ЭВМ 2021/2022

Отчёт по практическому заданию №1

**Работу выполнила:**
Козлова Ольга
Группа: 317

Москва
2021

# Содержание

# 1. Введение

Необходимо написать реализации метода ближайших соседей и кросс-валидации на языке программирования Python. Так же необходимо провести эксперименты с датасетом изображений цифр MNIST. Эксперименты анализируют зависимость качества работы от настройки различных параметров (аугментация датасета, выбор алгоритма, метрики, количества соседей и т.д.). Качество работы оценивается как время, затраченное на выполнение задачи, и точность определения правильного ответа.

# 2. Эксперименты

## 2.1. Зависимость времени от алгоритма

**Постановка задачи**

Всего есть 4 алгоритма поиска ближайших соседей: `'my_own'`, `'brute'`, `'kd_tree'`, `'ball_tree'` , надо сравнить время их работы на подмножествах признаков размера 10, 20, 100 и определить самые быстрые из них в различных ситуациях.

Таблица 2.1

**Время работы алгоритмов (сек.)**

| | `'my_own'` | `'brute'` | `'kd_tree'` | `'ball_tree'` |
|---|---|---|---|---|
| 10 признаков | 15.1 | 10.7 | 4.1 | 7.6 |
| 20 признаков | 14.1 | 11.3 | 10.7 | 24.0 |
| 100 признаков | 14.9 | 12.3 | 179.2 | 207.1 |

**Выводы**

Видно, что время работы на алгоритмах `'kd_tree'` и `'ball_tree'` быстро растет с увеличением количества признаков. Поэтому они не подходят для задачи классификации изображений MNIST, так как там 784 признака, алгоритмы будут слишком долго работать. `'my_own'` и `'brute'` показывают сопоставимые результаты по времени, но все же `'brute'` работает лучше.

## 2.2. Зависимость времени и точности от количества соседей и метрики

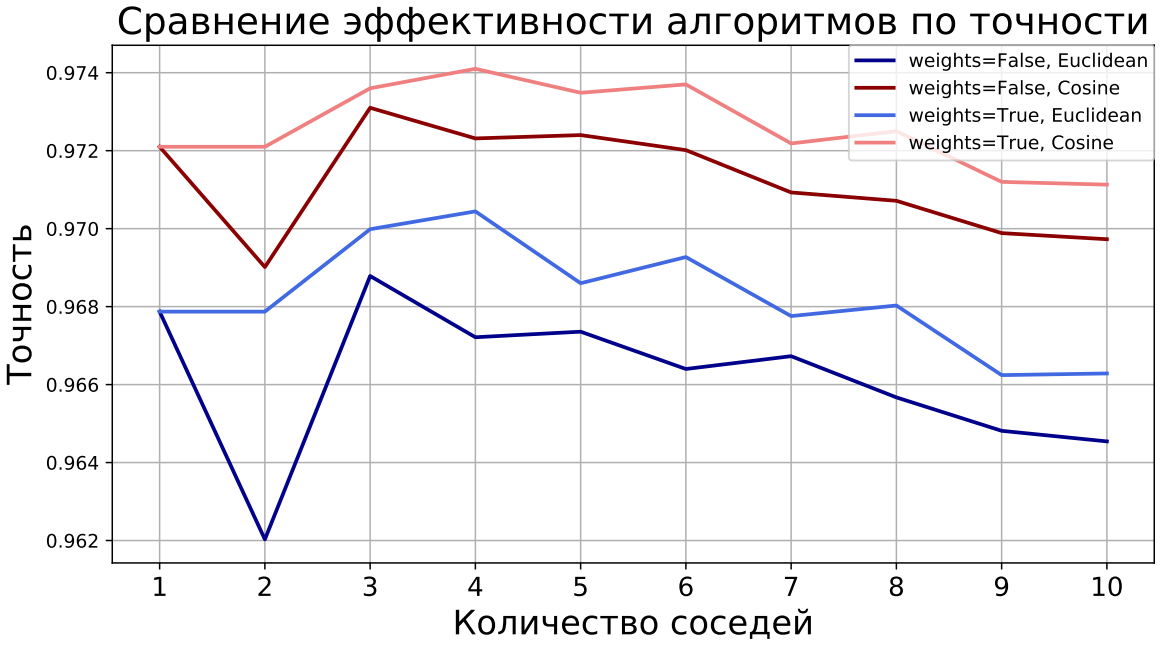
**Постановка задачи**

Рассмотреть все комбинации параметров количества соседей (от 1 до 10), метрик (`'Euclidean'`, `'Cosine'`) и использования весов (считать вклад каждого соседа с весом или без, вес: $1/($ distance $+\varepsilon)$, где $\varepsilon - 10^{-5}$), и определить лучшую комбинацию. Можно считать только для алгоритма `'brute'`, так как на остальных получается слишком долгое время работы.

Таблица 2.2

**Время работы алгоритма для 1-10 соседей (сек.)**

| | `'Euclidean'` | `'Cosine'` |
|---|---|---|
| `'weights=False'` | 273.0 | 272.7 |
| `'weights=True'` | 258.5 | 250.2 |




**Выводы**

Время работы сопоставимо для всех комбинаций, разница в ∼20 секунд в зависимости от наличия весов возникает, скорее всего, из-за особенностей языка Python. И если усреднить результаты по нескольким итерациям, время получится примерно одинаковое. Поэтому по времени нельзя выделить лучшие параметры.

Видно, что самые эффективные по точности параметры - метрика: `'Euclidean'` и `'weights=True'`. Провалы для 2 соседей происходят из-за того, что нужный класс определяется случайным образом из двух. Эта проблема решается, когда учитывается расстояние, так как более близкий сосед берется с большим весом, и с большой вероятностью, именно у ближайшего соседа объекта такой же класс, как и у самого объекта. Косинусная метрика лучше работает, потому что она учитывает только угол между векторами объектов, в то время как Евклидова метрика учитывает и расстояние, и угол. Например, если взять изображение из датасета MNIST и уменьшить величины всех пикселей в 2 раза, то по косинусной метрике изображения окажутся близкими, но по евклидова метрике это уже будет не так.

## 2.3. Исследование точности лучшего алгоритма

**Постановка задачи**

Сравнить точность лучшего алгоритма на тестовой выборке и на кросс-валидации с точностью лучших алгоритмов из Интренета. Проанализировать матрицу ошибок, и посмотреть примеры ошибок.

Таблица 2.3

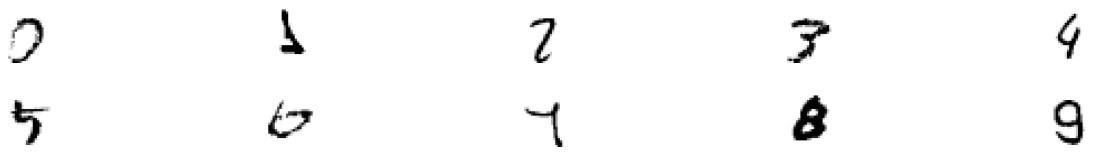
**Точности для разных примеров**

| | тестовая выборка | кросс-валидация | алгоритм из Интернета |
|---|---|---|---|
| Точность | 0.975 | 0.974 | 0.998 |

Точность на кросс-валидации немного ниже, т.к. она опредляет ответ по меньшей выборке. Разница между лучшим на сегодняшний день алгоритмом довольно большая,

но в нем используются принципиально другой подход - CNN (сверточные нейронные сети).

Рисунок 2.1. **Примеры неправильных ответов**



**0**: 7, **1**: 2, **2**: 7, **3**: 7, **4**: 9, **5**: 4, **6**: 0, **7**: 4, **8**: 0, **9**: 8

В основном эти цифры напоминают другие цифры с маленькой разницей в несколько пикселей. Это происходит из-за неудачного изображения (как 0 и 6 в примере) или из-за сходства некоторых цифр в целом. При подсчете расстояния из-за малого отличия алгоритм не может точно определить какой класс верный.

| | 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 |

Рисунок 2.2. **Матрица ошибок**

В ячейке [i, j] этой матрицы находится количество цифр i, которые модель определила как j. Модель ошибается в основном на наиболее похожих цифрах, таких как 1 и 7, 9 и 4. Для человека обычно сложно не различить 2 и 0 или 9 и 7, но из-за похожих очертаний (с точностью до нескольких черточек), алгоритм не различает их. Интересно, что таблица не симметричная, то есть 4 похожа на 9 больше чем 9 на 4. Это происходит из-за особенностей написания цифр человеком - нам сложно изменить какие-то части у определенных цифр. Например, цифру 9 сложно написать без верхней дуги, в то время как в цифре 4 нам легко случайно сблизить верхние 2 палочки.

**Выводы**

Исследуемый алгоритм работает значительно хуже лучшего среди всех существующих на данный момент, его точность на несколько процентов выше. В целом результаты не плохие, и модель почти всегда дает правильный ответ, за исключением не качественных изображений, где распознать цифру сложно даже человеческому глазу, и изображений с очень похожим написанием цифр.

## 2.4. Результаты при различных аугментациях

**Постановка задачи**

По кроссвалидации и аугментации понять, какие варианты аугментаций лучше всего работают (дано 3 варианта: поворот, смещение и размытие фильтром Гаусса изображений). Проанализировать два подхода - расширение с помощью аугментаций обучающей и расширение с помощью аугментаций тестовой выборки.

Таблица 2.4

**Точность при поворотах**

| -15° | -10° | -5° | 5 | 10° | 15° |
|---|---|---|---|---|---|
| 0.969 | 0.981 | 0.994 | 0.994 | 0.981 | 0.969 |

Таблица 2.5

**Точность при смещении вдоль двух осей**

| | 0 pix | 1 pix | 2 pix | 3 pix |
|---|---|---|---|---|
| 0 pix | 1.0 | 0.981 | 0.964 | 0.963 |
| 1 pix | 0.982 | 0.968 | 0.963 | 0.963 |
| 2 pix | 0.964 | 0.963 | 0.963 | 0.963 |
| 3 pix | 0.963 | 0.963 | 0.962 | 0.962 |

Таблица 2.6

**Точность при размытии фильтром Гаусса с различными дисперсиями**

| 0.5 | 1 | 1.5 |
|---|---|---|
| 1.0 | 0.990 | 0.976 |

При аугментации данных, которая мало влияет на исходные происходит переобучение - это видно по очень высоким точностям: поворот на малый угол, маленькое смещение, маленькая дисперсия фильтра Гаусса. Результат почти не зависит от знака угла и оси, вдоль которой происходит смещение, потому что расстояние не зависит от этих параметров.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 974 | 1 | 0 | 0 | 0 | 0 | 3 | 1 | 1 | 0 |
| 1 | 0 | 1132 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 11 | 0 | 1007 | 1 | 0 | 0 | 1 | 8 | 4 | 0 |
| 3 | 0 | 0 | 3 | 986 | 1 | 5 | 0 | 4 | 7 | 4 |
| 4 | 2 | 1 | 0 | 0 | 946 | 0 | 4 | 2 | 0 | 27 |
| 5 | 3 | 0 | 0 | 10 | 1 | 864 | 4 | 2 | 7 | 1 |
| 6 | 3 | 2 | 0 | 0 | 0 | 3 | 948 | 0 | 2 | 0 |
| 7 | 0 | 7 | 8 | 0 | 0 | 1 | 0 | 1007 | 0 | 5 |
| 8 | 5 | 0 | 2 | 4 | 0 | 4 | 0 | 5 | 951 | 3 |
| 9 | 6 | 4 | 3 | 5 | 2 | 3 | 1 | 7 | 3 | 975 |

Рисунок 2.3. **Матрица ошибок при аугментации поворотом**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 978 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1130 | 3 | 0 | 0 | 0 | 2 | 0 | 0 | 0 |
| 2 | 2 | 2 | 1013 | 4 | 1 | 0 | 0 | 8 | 2 | 0 |
| 3 | 0 | 1 | 1 | 987 | 0 | 9 | 0 | 5 | 4 | 3 |
| 4 | 0 | 4 | 0 | 0 | 948 | 0 | 3 | 0 | 1 | 26 |
| 5 | 2 | 0 | 0 | 10 | 2 | 865 | 2 | 1 | 5 | 5 |
| 6 | 4 | 2 | 0 | 0 | 1 | 3 | 948 | 0 | 0 | 0 |
| 7 | 0 | 14 | 7 | 0 | 3 | 0 | 0 | 998 | 0 | 6 |
| 8 | 2 | 1 | 1 | 19 | 2 | 5 | 2 | 3 | 935 | 4 |
| 9 | 1 | 6 | 1 | 7 | 5 | 0 | 1 | 8 | 2 | 978 |

Рисунок 2.4. **Матрица ошибок при аугментации смещением**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 976 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 0 |
| 1 | 0 | 1131 | 3 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 2 | 6 | 1 | 1004 | 1 | 0 | 0 | 2 | 14 | 4 | 0 |
| 3 | 0 | 0 | 1 | 984 | 1 | 10 | 0 | 5 | 4 | 5 |
| 4 | 0 | 2 | 0 | 0 | 958 | 0 | 4 | 2 | 0 | 16 |
| 5 | 1 | 1 | 0 | 8 | 1 | 870 | 5 | 1 | 2 | 3 |
| 6 | 3 | 3 | 0 | 0 | 1 | 3 | 948 | 0 | 0 | 0 |
| 7 | 1 | 8 | 5 | 2 | 2 | 1 | 0 | 998 | 0 | 11 |
| 8 | 6 | 0 | 2 | 6 | 5 | 7 | 4 | 5 | 934 | 5 |
| 9 | 2 | 6 | 0 | 3 | 7 | 1 | 0 | 8 | 3 | 979 |

Рисунок 2.5. **Матрица ошибок при аугментации размытием**

Смещение и поворот в целом улучшают результат, так как увеличивается обучающая выборка. При размытии лучше лучше лучше видно общий контур, и лучше видно, есть ли разрыв между двумя частями цифры или нет. Так, например, цифра 4 реже стала распознаваться как 9.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 977 | 0 | 6 | 0 | 0 | 2 | 3 | 0 | 7 | 2 |
| 1 | 1 | 1132 | 1 | 0 | 2 | 0 | 2 | 12 | 0 | 6 |
| 2 | 0 | 3 | 1011 | 2 | 0 | 0 | 0 | 7 | 3 | 1 |
| 3 | 0 | 0 | 1 | 990 | 0 | 9 | 0 | 0 | 9 | 5 |
| 4 | 0 | 0 | 0 | 1 | 956 | 1 | 0 | 1 | 2 | 5 |
| 5 | 0 | 0 | 0 | 5 | 0 | 869 | 3 | 1 | 3 | 1 |
| 6 | 1 | 0 | 1 | 0 | 4 | 2 | 950 | 0 | 1 | 1 |
| 7 | 1 | 0 | 9 | 4 | 2 | 1 | 0 | 1000 | 4 | 6 |
| 8 | 0 | 0 | 3 | 4 | 0 | 5 | 0 | 0 | 942 | 3 |
| 9 | 0 | 0 | 0 | 4 | 18 | 3 | 0 | 7 | 3 | 979 |

Рисунок 2.6. **Матрица ошибок при аугментации обучающей выборки**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 976 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1129 | 2 | 1 | 0 | 0 | 2 | 0 | 0 | 0 |
| 2 | 14 | 0 | 998 | 1 | 0 | 0 | 0 | 10 | 7 | 0 |
| 3 | 2 | 1 | 3 | 971 | 0 | 8 | 0 | 5 | 14 | 5 |
| 4 | 7 | 2 | 0 | 0 | 930 | 0 | 8 | 1 | 0 | 32 |
| 5 | 7 | 1 | 0 | 15 | 0 | 840 | 13 | 0 | 9 | 4 |
| 6 | 6 | 3 | 0 | 0 | 1 | 1 | 946 | 0 | 0 | 0 |
| 7 | 3 | 10 | 5 | 0 | 2 | 0 | 0 | 984 | 0 | 23 |
| 8 | 6 | 4 | 1 | 5 | 4 | 4 | 4 | 5 | 935 | 4 |
| 9 | 13 | 7 | 1 | 5 | 7 | 3 | 2 | 4 | 4 | 961 |

Рисунок 2.7. **Матрица ошибок при аугментации тестовой выборки, со значениями, деленными на 2**

Для этих двух матриц проводилась аугментация поворотом, смещением и размытием обучающей и тестовой выборок соответственно. Ответ получался путем голосования из 3ех полученных результатов. Точность на аугментации обучающей выборки - **0.981**, тестовой - **0.967**. Без аугментаций точность была **0.975**. Эффективнее расширять обучающую выборку, потому что так увеличивается количество различных объектов, по

которым лучше восстанавливаются объекты тестовой выборки. Если же расширить тестовую выборку, качество обучения упадет, потому что могут появиться какие-то не учтенные новые варианты объектов, класс которых будет плохо определен.

# 3. Заключение

Алгоритм KNN довольно хорошо работает для распознавания цифр из датасета MNIST. Самая быстрая стратегия для KNN - `'brute'`, а метрика - `'Cosine'`. Улучшать результаты работы можно засчет аугментации обучающей выборки, но надо подбирать ее параметры так, чтобы изображения продолжали принадлежать своим классам и, с другой стороны, не повторяли исходные. Алгоритм не достаточно хорошо классифицирует похожие цифры, которые человек может отличать, и аугментация лишь частично решает эту проблему.

# 4. Литература


- https://stackoverflow.com/
- Документации библиотек numpy, matplotlib, sklearn и др.
- Шаблон отчета в LaTeX