Практическая работа №4: «Подготовка данных для машинного обучения»

Оглавление

Цель работы

Получить практические навыки применения методов очистки данных для машинного обучения.

Задачи работы

1. Научиться находить проблемы в данных.
2. Научиться исправлять найденные проблемы в данных.

Перечень обеспечивающих средств

1. ПК.
2. Учебно-методическая литература.
3. Задания для самостоятельного выполнения.

Общие теоретические сведения

Очистка и исправление данных

Очистка данных – заполнение отсутствующих значений параметров, поиск и удаление некорректных значений параметров.

Очистка данных борется со следующими типами проблем:

- Неполнота данных
- Шум
- Несоответствие

Примеры проблем:

1. Замноженные идентификаторы.
2. Некорректный формат значений.
3. Недопустимые или несуществующие значения.
4. Нелогичные значения.
5. Несоответствующие друг другу значения.
6. Значения другого параметра.
7. Отсутствующие значения.
8. Опечатки.

Уменьшение размерности – упрощение структуры элементов, поиск и удаление несущественных параметров.

Уменьшение размерности борется со следующими типами проблем:

- Необходимость хранить большие объемы информации.
- Долгое время построения и обучения моделей.
- Использование коррелирующих (связанных) параметров.
- Сложность в визуализации данных.
- Сложность интерпретации моделей.

Нормализация данных – приведение значений всех параметров к единому диапазону для исключения шума в данных и улучшения точности работы модели.

Изменение типов данных при нормализации:

- Категориальный параметр в числовой.
- Категориальный параметр в несколько числовых.
- Числовой параметр в категориальный.
- Текст, изображение, звук, виде – в набор числовых параметров.

Типичные проблемы, решаемые при нормализации данных

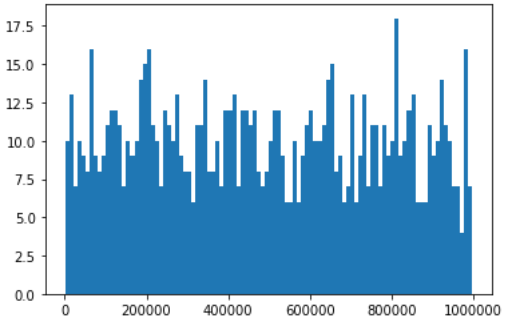
Проблема №1:

параметр принимает очень большие значения, что может привести к ошибкам переполнения, либо очень маленькие значения, что может привести к потерям при округлении.

Решение проблемы №1:

преобразование значений параметра, обычно к интервалу (0, 1) или (-1, 1).

Пример распределения:



В такой ситуации нужно понять, на какой распределение более похоже рассматриваемое — на равномерное (значения распределены примерно одинаково по всей области значений) или на нормальное (распределение симметричное относительно центрального пика). Для визуальной проверки распределения проще всего построить гистограмму значений параметра.

Равномерное распределение приводится к (0, 1) преобразованием X = (X - minX) / (maxX - minX)

Для этого можно использовать библиотеку sklearn:

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
old_x = numpy.array(old_x).reshape(-1,1)
new_x = scaler.fit_transform(old_x)
```

Нормальное распределение приводится в N(0,1) преобразованием X = (X - meanX) / dev,

где mean – среднее значение (пик распределения), а dev – стандартное отклонение.

Аналогично, с помощью sklearn:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
old_x = numpy.array(old_x).reshape(-1,1)
new_x = scaler.fit_transform(old_x)
```

Проблема №2:

несколько сходных по смыслу параметров имеют сильно различающиеся интервалы значений, из-за чего их вклад может различаться сильнее, чем это имеет смысл.

Решение проблемы №2:

преобразование значений таких параметров к общему интервалу.

Аналогично решению проблемы №1, нужно определить тип распределения и выбрать соответствующее преобразование.

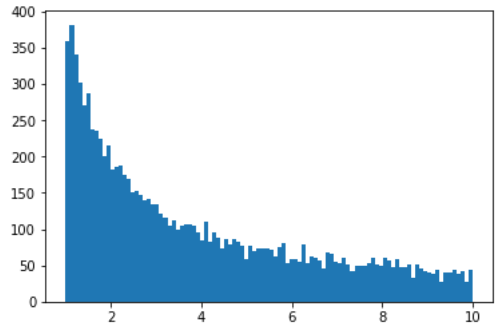
Проблема №3:

неравномерное распределение значений параметра, из-за чего небольшая часть значений повторяется часто, а большая — редко, что приведет к низкому качеству обучения на большей части параметров.

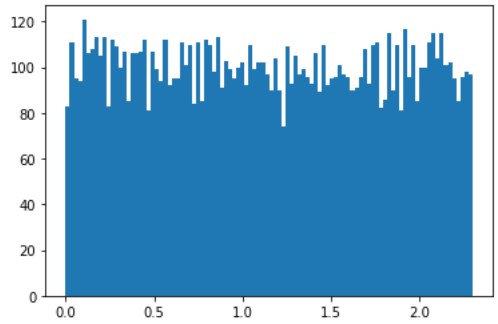
Решение проблемы №3:

Изменение распределения значений параметров путем применения к ним нелинейной функции.

Пример распределения:



В таком случае подойдет функция логарифма: X = log(X), результат будет:



Проблема №4:

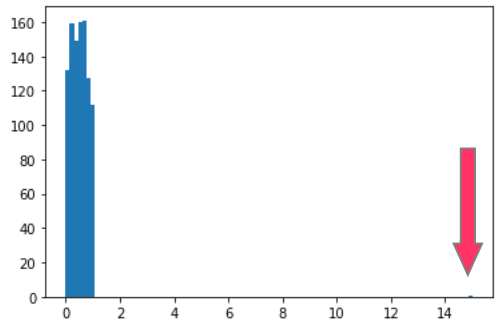
в значениях параметра есть единичные «выбросы», которые сильно увеличивают интервал значений, что препятствует корректному преобразованию интервала значений.

Решение проблемы №4:

отсечение «выбросов», что приводит к уменьшению интервала значений. При этом задается разрешенный интервал значений, а все значения вне интервала заменяются на ближайшую к ним значение разрешенного интервала.

Например, для списка [0, 1, 2, 3, 1000] можно задать разрешенный интервал от 0 до 3, тогда после отсечения список будет иметь вид: [0, 1, 2, 3, 3].

Пример распределения:



Можно использовать метод numpy.clip(<*массив*>, <*минимум*>, <*максимум*>):

```
import numpy
new_x = numpy.clip(old_x, min_x, max_x)
```

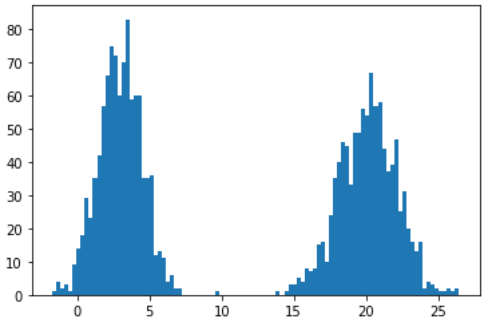
Проблема №5:

распределение значений параметра имеет несколько явно выраженных пиков, что означает, что его корректнее рассматривать как атрибутивный, чем числовой.

Решение проблемы №5:

группирование значений параметра.

Пример распределения:



Достаточно найти число, которое разделяет группы, и, используя сравнение с этим числом, создать два новых параметра.

Проблема №6:

значения параметра нельзя рассматривать как непрерывные числа (строки, даты и т. п.), алгоритмы машинного обучения в принципе не могут оперировать такими данными.

Решение проблемы №6:

прямое унитарное кодирование (one-hot encoding):

1. создается список всех возможных значений параметра,
2. список нумеруется,
3. для каждого номера в списке создается новый параметр, который равен 1, если первоначальный параметр имеет соответствующее значение, и 0 в остальных случаях.

Например, для параметра x = ['один', 'два', 'два', 'три', 'три'] будут созданы три новых параметра со следующими значениями:

параметр №1: [1, 0, 0, 0, 0],

параметр №2: [0, 1, 1, 0, 0],

параметр №3: [0, 0, 0, 1, 1].

Для этого можно использовать метод pandas.get_dummies():

```
new_x = pandas.get_dummies(old_x)
```

Чтобы присоединить полученные новые столбцы к DataFrame используйте метод join(), например: new_dataframe = old_dataframe.join(x).

Проблема №7:

Несколько значений параметра повторяются небольшое количество раз каждое, из-за этого при использовании прямого унитарного кодирования возникает много параметров, на которых сложно проводить обучение.

Решение проблемы №7:

Объединение редких значений параметр в новое, например, «Вне категорий» или «Другое», таким образом не нужно добавлять лишние параметры, а частота нового параметра будет сравнима с частотой других.

Задание

Пояснение

Для сохранения результатов данной работы вам понадобится два файла: doc/docx – для текста и ipynb – для кода. Назовите их одинаково: «*Фамилия* – задание 4».

Часть 1

- Обновите свой репозиторий, созданный в практической работе №1, из оригинального репозитория: https://github.com/mosalov/Notebook_For_AI_Main.

Часть 2

- Откройте свой репозиторий в Binder (https://mybinder.org/).
- Откройте файл «task4.ipynb».
- Используя уже имеющийся в файле код, загрузите данные из файла «top50.csv».
- Обработайте полученные данные, сохраните код в ipynb-файле. Необходимые пояснения опишите в своём docx/doc-файле.

Требования к отчету

Оба файла (doc/docs и ipynb) загрузите в свой репозиторий, созданный в практическом задании №1 по пути: «Notebook_For_AI_Main/2020 Осенний семестр/Практическое задание 4/» и сделайте пул-реквест.

Литература

1. https://ru.wikipedia.org/wiki/Машинное_обучение
2. http://www.machinelearning.ru/
3. https://vas3k.ru/blog/machine_learning/
4. https://habr.com/ru/post/511132/
5. https://docs.microsoft.com/ru-ru/azure/machine-learning/team-data-science-process/prepare-data