



Универзитет у Нишу  
Електронски факултет



## Семинарски рад

*Прикупљање и предобрада података за Машинско учење*

# Трансформација података

**Ментор:** Доц. др Александар Станимировић

**Кандидат:** Петар Петковић 1467

**Ниш, 2023**

## Contents

Увод.....	3
Технике трансформације података.....	4
Скалирање података.....	4
Нормализација података.....	7
Стандардизација података.....	11
Поређење и препоруке.....	13
Дискретизација података.....	14
Ненадгледана дискретизација података.....	15
Бининг једнаке ширине.....	15
Бининг једнаке фреквенције.....	15
Надгледана дискретизација података.....	17
Случаји употребе и препоруке.....	17
Промена расподеле.....	18
Опис проблема.....	18
Препоруке.....	18
Енкодирање категоријских атрибута.....	19
Кодирање ознака.....	19
One-hot кодирање.....	20
Поређење и препоруке.....	21
Уклањање екстремних вредности.....	22
Руковање са екстремним вредностима.....	24
Случаји употребе и препоруке.....	25
Закључак.....	26
References.....	28

## Увод

По дефиницији, трансформација података у контексту машинског учења је процес претварања односно структурирања података у употребљив формат који се може анализирати и на основу кога се може креирати добар модел [1].

У данашњем времену, дневно се генерише и до 118 зетабајта података [2]. Подаци се генеришу са разних сензора, на различите начине и начин записивања података у тренутку генерисања најчешће је у облику који није употребљив. Због тога, трансформација података постаје један од најважнијих корака у припреми података за анализу. Трансформација података обухвата низ техника које се примењују на сирове податке како би се побољшао њихов квалитет, прецизност и поузданост анализе.

Трансформација података мења формат, структуру или вредности података и претвара их у чисте, употребљиве податке. Трансформација података обично подразумева неколико техника, као што су:

- Скалирање података
  - а. Нормализовање података
  - б. Стандардизација података
- Дискретизација података
- Промена расподеле
- Енкодирање категоричких атрибута
- Уклањање екстремних вредности (eng. outliers)

Циљ овог семинарског рада је да објасни правилну употребу трансформација на свакој од могућих аномалија која се може појавити у скупу података. Како би то било могуће, неопходно је детаљно објаснити сваку од трансформација, у којим случајевима се примењује и који проблем решава. Такође, важно је објаснити која се операција извршава над подацима односно који је крајњи исход трансформације. Детаљније о горе наведеним трансформацијама у следећем поглављу.

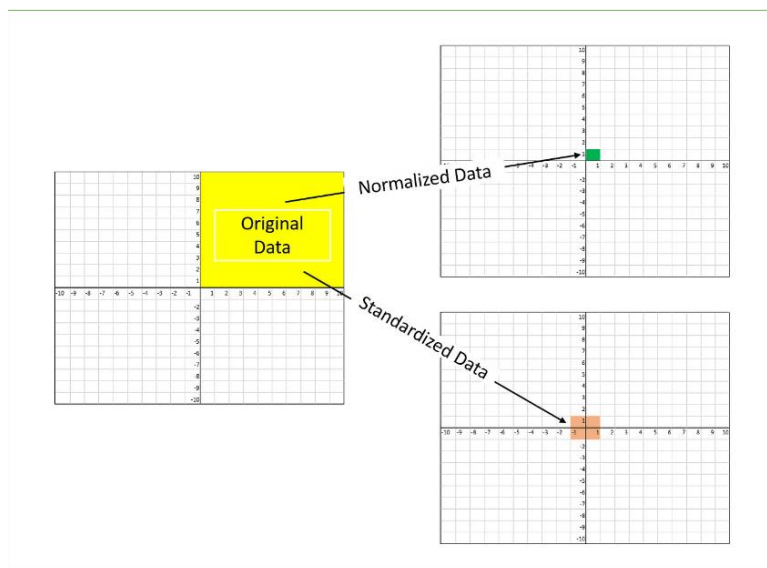
# Технике трансформације података

У овом поглављу, фокус ће бити на детаљној анализи сваке технике трансформације података. За сваку од трансформација следи дефиниција, објашњење, пример како делује на простом скупу података, добре и лоше примене и савети за коришћење у реалним проблемима.

## Скалирање података

Скалирање података у машинском учењу је један од најбитнијих корака током обраде података који се морају обавити пре креирања модела машинског учења. Скалирање може направити разлику између слабог модела и доброг модела машинског учења.

Узмимо за пример два атрибута у почетном скупу података, плату и године. Плата је атрибут који може достићи и до шест цифара, док су године двоцифрени претежно двоцифрени. Модел учења, због велике разлике у вредности, сматра да је атрибут који представља плату доста битнији од атрибута године, што може довести до тога да крајњи модел буде веома лош, јер форсира плату у односу на године. Због тога, неопходно је довести их на сличне вредности, како би се креирао бољи модел. То се постиже помоћу техника скалирања података. Најпознатије технике скалирања података су нормализација и стандардизација, и графички приказ који објашњава разлику ове две технике се може видети на слици 1. [3]

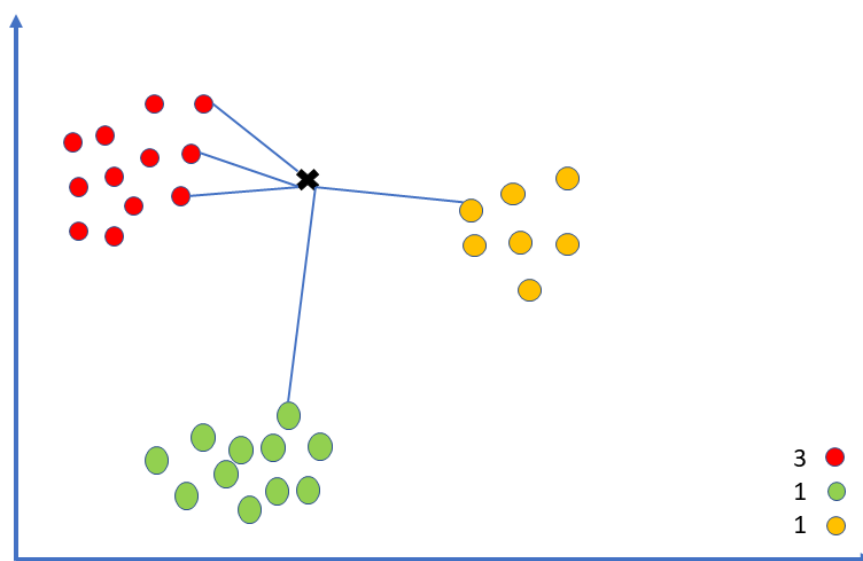


Слика 1 – Разлика између стандардизације и нормализације

Нормализација података се користи када желимо да представимо све вредности одређеног атрибута између два броја. У пракси се те вредности обично крећу у интервалима  $[0,1]$  или  $[-1,1]$ . Стандардизација је техника скалирања где су вредности центриране око средње вредности са јединичном стандардном девијацијом. То значи да средња вредност атрибута постаје нула и резултујућа дистрибуција има јединичну стандардну девијацију. Детаљније о овим два техникама у наредним поглављима.

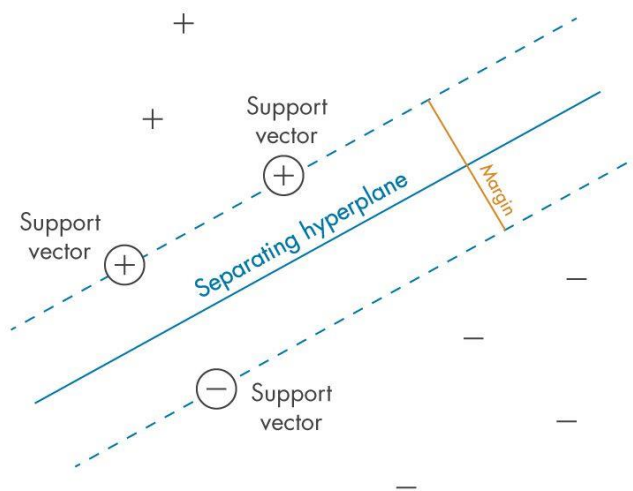
Врло је важно знати када је неопходно да подаци буду скалирани. Скалирање података се користи када се при преирању модела користи алгоритам који рачуна растојање између података. Уколико се пре коришћења таквих алгоритама не скалирају подаци, атрибути са великом вредношћу ће доминирати при израчунавању растојања, као што је малопре напоменуто. Алгоритми који користе растојање и код којих је битно скалирати фичере су:

- **К-најближих суседа** (енг. KNN) који рачуна растојање између података користећи еуклидско или менхетн растојање. Растојање је мера која је осетљива на величине и требало би претходно скалирати податке како би сви тежили једнако, у супротном алгоритам неће имати добре перформансе. Илустрацију овог алгоритма можемо видети на слици 2.



Слика 2 – Илустрација алгоритма К-најближих суседа

- Метод потпорних вектора (SVM)** је алгоритам машинског учења који покушава да пронађе линију (или хипер-раван) која раздваја две различите групе тачака података. То ради покушавајући да пронађе највеће могуће растојање између линије и најближих тачака података из сваке групе. Ово растојање се назива маргина. Скалирање је важно код метода потпорних вектора јер маргина зависи од размере улазних карактеристика. Ако неке карактеристике имају много већу скалу од других, оне могу доминирати у израчунавању маргине и учинити алгоритам мање ефикасним у проналажењу оптималне линије. Скалирањем карактеристика тако да имају сличан опсег, метод потпорних вектора може узети у обзир све карактеристике када покушава да пронађе оптималну линију и ефикасно пронађе највећу могућу вредност маргине. Сликровити приказ алгоритма се налази на слици 3.



Слика 3 – Метода потпорних вектора

- Анализа главних компоненти (Principal Component Analysis)** је метод који тежи смањењу димензионалности скупа података, а да се притом и даље задржи што је могуће више информација. То ради проналажењем нових атрибута које су комбинација оригиналних атрибута. Међутим, PCA се ослања на матрицу коваријансе улазних атрибута. Ова матрица мери колико се атрибути заједно разликују и на њу утиче размера атрибута. Ако неки атрибути имају много веће размере од других, они ће доминирати у израчунавању матрице коваријансе, што може резултовати не баш оптималним представљањем података. Како би се овај проблем избегао, важно је скалирати улазне карактеристике пре примене PCA. Ово осигурава да сви атрибути подједнако доприносе матрици коваријансе и да је резултујућа репрезентација података оптимална.

Са друге стране, постоје алгоритми који не захтевају нормализацију и скалирање с обзиром да се они заснивају на неким другим правилима. Алгоритми који не захтевају претходно скалиране податке су:

- **Алгоритми базирани на стаблу** као што су стабла одлучивања (eng. Decision Tree), насумичне шуме(eng. Random Forests) и машине за повећање градијента (eng. Gradient Boosting Method), сви доносе одлуке на основу серије бинарних подела које деле податке у мање групе на основу вредности неких атрибута. Циљ ових алгоритама је да пронађу најбоље поделе које максимизирају раздвајање између различитих класа у подацима. Пошто су поделе направљене на основу релативног редоследа вредности атрибута, а не на самим стварним вредностима, ови алгоритми нису осетљиви на однос вредности атрибута. Другим речима, исте поделе би биле направљене без обзира да ли су вредности података претходно скалиране. Примера ради, атрибут као што је дужина може бити изражен у инчима, сантиметрима или некој другој јединици мере, то неће утицати на поделу. Метод насумичних шума и машине за повећање градијента су методе ансамбла које користе више стабала за побољшање перформанси. Насумичне шуме користе скуп независних стабала одлучивања које се обучавају на различитим подскуповима података, а затим комбинују резултате да би направили коначно предвиђање. Машине за повећање градијента, с друге стране, користи серију стабала одлучивања која се итеративно обучава, са сваким новим стаблом изграђеним да исправи грешке претходног стабла.
- **Naive Bayes** је пробабилистички алгоритам који се обично користи за задатке класификације. Он израчунава вероватноћу сваке могуће класе дате скупу улазних карактеристика, а затим бира класу са највећом вероватноћом као предвиђену класу. „Наивно“ у наивном Бајесу се односи на претпоставку да су све карактеристике независне једна од друге, с обзиром на ознаку класе. То значи да алгоритам разматра сваку карактеристику изоловано када израчунава вероватноће, уместо да узима у обзир све могуће интеракције или корелације између карактеристика. Наивни Бајес није под утицајем скалирања атрибута јер израчунава вероватноће на основу релативних фреквенција сваке вредности атрибута унутар сваке класе. Апсолутна величина вредности обележја не утиче на ове вероватноће.

#### Нормализација података

Као што је раније речено, нормализација у машинском учењу је процес превођења података у опсег  $[0, 1]$  (или било који други опсег, на пример  $[-1, 1]$ ). Најпознатији алгоритми који се користе за нормализацију података су: [4]

- Min-Max скалер
- Максимално апсолутно скалирање
- Робусно скалирање

У наставку овог поглавља следи детаљан опис и случаји употребе наведених алгоритама нормализације података.

### Min-Max скалер

Min-Max скалер је техника која се користи за нормализацију података. То је метод линеарне трансформације која слика вредности у скупу података на фиксни опсег, обично између 0 и 1, тако да се најмања вредност у скупу података пресликава на 0, док се највећа вредност слика у 1.

Препорука за употребу овог метода је када улазни подаци нису равномерно распоређени или имају широк опсег вредности. Математички се може изразити као:

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Слика 4 – Формула за израчунавање коју користи Min-Max скалер

где је вредност  $x$  оригинална вредност у скупу података,  $x_{min}$  је најмања вредност у скупу података, док је  $x_{max}$  највећа вредност у скупу података. На пример, рецимо да имамо скуп података  $A$  са следећим вредностима:

```
A = [10, 20, 30, 40, 50]
```

Да бисмо применили Min-Max скалер на овај скуп података, прво проналазимо минималне и максималне вредности:

```
minimalna_vrednost = 10  
maximalna_vrednost = 50
```

Затим примењујемо формулу на сваку вредност у скупу података:

```
skalirana_vrednost(10) = (10 - 10) / (50 - 10) = 0  
skalirana_vrednost(20) = (20 - 10) / (50 - 10) = 0,111  
skalirana_vrednost(30) = (30 - 10) / (50 - 10) = 0,333  
skalirana_vrednost(40) = (40 - 10) / (50 - 10) = 0,556  
skalirana_vrednost(50) = (50 - 10) / (50 - 10) = 1
```

Дакле, скалирани скуп података  $B$  би био:

```
B = [0, 0,111, 0,333, 0,556, 1]
```



### Максимално апсолутно скалирање

Максимално апсолутни скалер је техника обраде података која се користи за скалирање и нормализацију података. Он скалира податке тако што сваку вредност у скупу података дели са максималном апсолутном вредношћу у скупу података. Ова трансформација осигурава да су све вредности у скупу података између -1 и 1, а максимална апсолутна вредност је сада 1. Формула за максимално апсолутно скалирање може се математички изразити као:

$$x_{scaled} = \frac{x}{\max(|x|)}$$

Слика 5 – Формула за израчунавање коју користи максимално апсолутни скалер

где је  $x$  оригинална вредност у скупу података, а  $\max(|x|)$  је максимална апсолутна вредност у скупу података. На пример, рецимо да имамо скуп података  $A$  са следећим вредностима:

```
A = [10, -20, 30, -40, 50]
```

Да бисмо применили максимално апсолутно скалирање на овај скуп података, прво налазимо максималну апсолутну вредност  $M$  у скупу података:

```
M = max(abs(10), abs(-20), abs(30), abs(-40), abs(50)) = 50
```

Затим примењујемо горе наведену формулу на сваку вредност у скупу података:

```
skalirana_vrednost(10) = 10 / 50 = 0,2  
skalirana_vrednost(20) = -20 / 50 = -0,4  
skalirana_vrednost(30) = 30 / 50 = 0,6  
skalirana_vrednost(40) = -40 / 50 = -0,8  
skalirana_vrednost(50) = 50 / 50 = 1.0
```

Дакле, скуп података  $B$  који представља скалирани почетни скуп  $A$  је сада:

```
B = [0,2, -0,4, 0,6, -0,8, 1,0]
```

## Робусно скалирање

Робустан скалер је техника за обраду података која се користи за скалирање и нормализацију података. Посебно је користан када се ради са скуповима података који имају одступања, јер скалирају податке одузимањем медијане скупа података и дељењем интерквартилним опсегом (IQR) скупа података. Резултат је скуп података који је усредсређен око нуле и има опсег вредности који је отпорнији на присуство екстремних вредности. Формула за робусни скалер може се математички изразити као:

$$X_{new} = \frac{X - X_{median}}{IQR}$$

Слика 6 – Формула која се примењује при робусном скалирању

где је  $X$  оригинална вредност у скупу података,  $X_{median}$  средња вредност скупа података, а  $IQR$  је интерквартилни опсег скупа података. На пример, рецимо да имамо низ  $A$  са следећим вредностима:

```
A = [10, 20, 30, 40, 50, 100]
```

Да бисмо применили Робустан скалер на овај скуп података, прво налазимо медијану и интерквартилни опсег:

```
median = Q2 = (30+40)/2 = 35  
Q1 = 20  
Q3 = 50  
IQR = Q3 - Q1 = 30
```

Затим примењујемо формулу на сваку вредност у скупу података:

```
skalirana_vrednost(10) = (10 - 35) / 30 = -0,8333  
skalirana_vrednost(20) = (20 - 35) / 30 = -0,5  
skalirana_vrednost(30) = (30 - 35) / 30 = -0,1667  
skalirana_vrednost(40) = (40 - 35) / 30 = 0,1667  
skalirana_vrednost(50) = (50 - 35) / 30 = 0,5  
skalirana_vrednost(100) = (100 - 35) / 30 = 2,1667
```

Дакле, скалирани скуп података  $B$  би био:

```
B = [-0,8333, -0,5, -0,1667, 0,1667, 0,5, 2,1667]
```

Као што видимо, робусни скалер је трансформисао податке тако да су центрирани око нуле и да има опсег вредности који је отпорнији на присуство екстремних вредности (eng. Outliers). Вредност од 100, која је изузетак у оригиналном скупу података, скалирана је на 2,1667, што је и даље значајно више од осталих вредности, али није тако екстремно као што је било у оригиналном скупу података.

#### Препоруке за употребу техника нормализације

Након што смо видели како ради сваки од алгорита нормализације, намеће се питање, у ком случају је најбоље користити сваки од њих. Иако је генерална препорука да се проба сваки од приступа и види какви ће се резултати добити, најбољи приступ би био користити:

- **Min-Max скалер** се користи када је нормална дистрибуција података, али желимо да поредимо вредности које користе различите величине и јединице мере и да се сачува значење нуле у подацима.
- **Максимални апсолутни скалер** је добар избор када желимо да сачувамо знак података, подаци су центрирани око нуле или не желимо да центрирамо податке и када подаци имају велику ширину и желимо ту ширину да сачувамо.
- **Робустан скалер** се користи када међу подацима има екстремних вредности (eng. Outliers). За проблем решавања екстремних вредности ће бити посебно поглавље, али је битно напоменути да је једно од решења ово. Такође, користи се када постоји лоша дистрибуција података.

#### Стандардизација података

Дефиниција стандардизације података гласи :

*„Стандардизација је техника скалирања где су вредности центриране око средње вредности са јединичном стандардном девијацијом. То значи да средња вредност атрибута постаје нула и резултујућа дистрибуција има јединичну стандардну девијацију“.*

На први поглед, дефиниција може деловати јако конфузно, па с тим у вези следи објашњење стручних појмова, а затим и пример рачунања. Подаци се скалирају одузимањем средње вредности од сваке вредности у скупу података, а затим се та вредност подели стандардном девијацијом скупа података. Формула за израчунавање гласи:

$$x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation}(x)}$$

Слика 7 – Формула за израчунавање стандардне девијације

где  $x$  представља тренутну вредност у скупу,  $\text{mean}(x)$  представља средњу

вредност елемента у скупу док standard deviation (x) представља стандардну девијацију скупа података.

Стандардна девијација је мера колико су појединачне тачке података у скупу података распоређене или одступају од средње вредности скупа података.

Стандардна девијација од 1 значи да је просечна удаљеност сваке тачке података од средње вредности 1 стандардна девијација. Ако су тачке података више распоређене, стандардна девијација ће бити већа, а ако су тачке података чвршће груписане око средње вредности, стандардна девијација ће бити мања.

Другим речима, стандардна девијација од 1 указује да већина тачака података у скупу података спада у 1 стандардну девијацију средње вредности. Ово може бити корисно за разумевање варијабилности скупа података и идентификовање било каквих одступања или необичних тачака података. Примера ради, нека постоји скуп података:

```
A = [70, 80, 90, 85, 75, 95, 65, 80, 90, 85]
```

Средња вредност тог скупа би била:

```
Mean(A) =  
(70 + 80 + 90 + 85 + 75 + 95 + 65 + 80 + 90 + 85) / 10 = 81
```

Следећи корак би био одузети од сваког елемента средњу вредност, а затим те вредности квадрирати и за њих израчунати средњу вредност.

```
(70 - 81) = -11 => pow(-11,2) = 121  
(80 - 81) = -1 => pow(-1,2) = 1  
(90 - 81) = 9 => pow(9,2) = 81  
(85 - 81) = 4 => pow(4,2) = 16  
(75 - 81) = -6 => pow(-6,2) = 36  
(95 - 81) = 14 => pow(14,2) = 196  
(65 - 81) = -16 => pow(-16,2) = 256  
(80 - 81) = -1 => pow(-1,2) = 1  
(90 - 81) = 9 => pow(9,2) = 81  
(85 - 81) = 4 => pow(4,2) = 16  
  
(121 + 1 + 81 + 16 + 36 + 196 + 256 + 1 + 81 + 16) / 10 = 68.5
```

Стандардна девијација представља корен ове вредности односно

```
sqrt(68.5) = 8.27
```

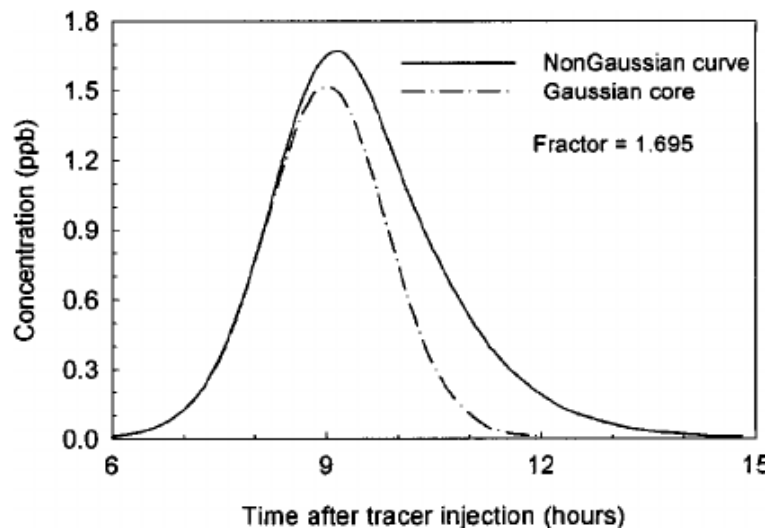
Новодобијене вредности вратимо у почетну формулу чиме се добија 3-скор

```
z = (x - mean(A)) / std_dev(A)
z-score of 70 = (70 - 81) / 8.27 = -1.331
z-score of 80 = (80 - 81) / 8.27 = -0.121
z-score of 90 = (90 - 81) / 8.27 = 1.088
z-score of 85 = (85 - 81) / 8.27 = 0.485
z-score of 75 = (75 - 81) / 8.27 = -0.727
z-score of 95 = (95 - 81) / 8.27 = 1.694
z-score of 65 = (65 - 81) / 8.27 = -1.935
z-score of 80 = (80 - 81) / 8.27 = -0.121
z-score of 90 = (90 - 81) / 8.27 = 1.088
z-score of 85 = (85 - 81) / 8.27 = 0.485
A = [-1.331, -0.121, 1.088, 0.485, -0.727, 1.694, -1.935, -0.121,
1.088, 0.485]
```

### Поређење и препоруке

Што се тиче најбољег случаја примене, иста правила се односе као и код технике нормализације. Стандардизација, као и нормализација, има смисла код алгоритама који узимају у обзир растојање између података. Како је већ детаљно објашњено (страна 5) због чега се скалирање врши пре тих алгоритама, следи само кратак осврт на алгоритме. Дакле, алгоритми пре којих је неопходно извршити скалирање су алгоритми који рачунају растојање између података и то су анализа главних компоненти (PCA), метод потпорних вектора (SVM), K-најближих суседа (KNN), док алгоритми код којих није неопходно претходно скалирање односно стандардизација података су алгоритми базирани на стаблу и наивни бајес. Веома је битно напоменути, када се говори о вештачким неуронским мрежама (ANN), скалирање итекако има велики утицај на побољшање перформанси неуронских мрежа и пожељно га је применити пре тренинга.

Свакако, намеће се питање, када се подаци скалирају, да ли користити технику стандардизације или технику нормализације. Први индикатор треба бити расподела података. Уколико подаци имају Гаусову расподелу/нормалну расподелу (облик звона) и атрибути у скупу података имају различиту скалу, препорука је да се подаци **стандаризују**. Са друге стране, уколико подаци немају Гаусову расподелу (искривљена расподела) и када атрибути имају сличне размере, препорука је да се подаци **нормализују**. Разлика између расподела се најбоље може видети на слици 8.



Слика 8 – Гаусова и не-гаусова расподела података

Уколико постоје недоумице, најбоље је применити обе технике на скуп података. Након примењених техника на почетном скуп података, на основу поређења перформанси модела се може видети која техника више одговара датом скуп података.

#### Дискретизација података

Машинско учење користи технику која се зове дискретизација података да би се континуирани подаци претворили у дискретне вредности или категорије. Континуирани опсег података је подељен на интервале или бинове, који могу послужити за поједностављење података и једноставније испитивање. [5]

Алгоритми машинског учења који захтевају дискретне податке на улазу итекако имају боље перформансе са дискретизованим вредностима, па је дискретизација података кључна у препроцесирању. Штавише, дискретизација може повећати тачност модела класификације и смањити утицај ирелевантних података и побољшати читљивост налаза. Употреба дискретних вредности има пуно предности као што су:

- Дискретне функције захтевају мање меморијског простора.
- Дискретне карактеристике су често ближе репрезентацији на нивоу знања.
- Подаци се могу смањити и поједноставити дискретизацијом, чинећи их лакшим за разумети, употребити и објаснити.
- Учење ће бити прецизније и брже коришћењем дискретних података.

Да би се успоставиле дискретне категорије или интервали, процес дискретизације података често подразумева одабир технике дискретизације, о чему ће касније бити више речи. На пример, приступ дискретизације би се могао користити за раздвајање старости пацијената у старосне групе као што су 18–30, 31–50 и 51–80, што ће касније помоћи при разумевању клиничке слике.

Методe дискретизације могу бити надгледане или ненадгледане у зависности од тога да ли користи информације о скуповима података. Надгледане методе користе називе класе приликом партиционисања континуираних атрибута. С друге стране, методе ненадгледане дискретизације не захтевају назив класе за дискретизацију континуираних атрибута. Уколико говоримо о ненадгледаној дискретизаци неке од техника су:

- Бининг једнаке тежине
- Бининг једнаке фреквенције
- K-Means бининг

Представници надгледане дискретизације су:

- Дискретизација базирана на грешци
- Дискретизација базирана на ентропији

Коначно, може се направити разлика између метода дискретизације одозго (eng. Top-down) на доле и одоздо према горе (eng. Bottom-up). Методе одозго на доле узимају у обзир један велики интервал који садржи све познате вредности атрибута, а затим партиционишу интервале на све мање и мање подинтервале до одређеног критеријума заустављања или оптималног броја постигнутих интервала. Насупрот томе, методе одоздо према горе у почетку узимају у обзир низ интервала, одређен скупом граничних тачака, да комбинује ове интервале током извршавања до одређеног критеријума заустављања.

#### Ненадгледана дискретизација података

Ненадгледана дискретизација података је врста технике дискретизације података која се не ослања на означене податке или претходно знање о дистрибуцији података.

##### *Бининг једнаке ширине*

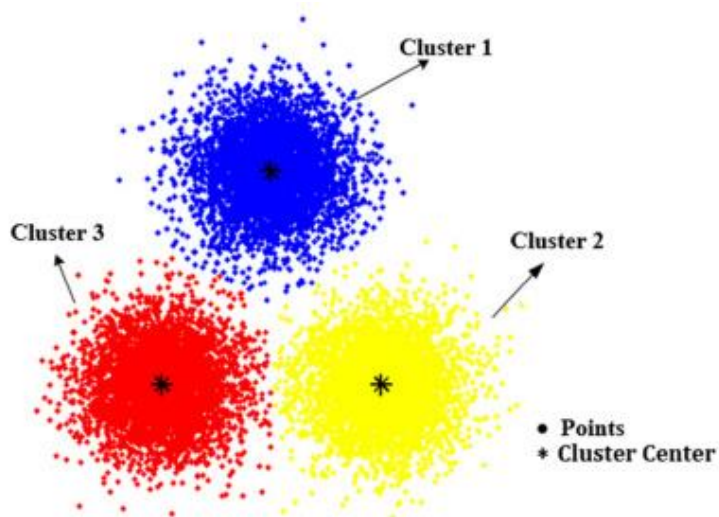
Бининг једнаке ширине дели опсег података на унапред одређен број интервала или бинова једнаке ширине. Дељењем опсега података са бројем потребних бинова, израчунава се ширина сваке корпе. На пример, ако желимо да поделимо опсег вредности од 0 до 100 на 10 бинова, свака би канта била широка 10 јединица (0-10, 10-20, 20-30, итд.).

##### *Бининг једнаке фреквенције*

Често се може наћи и под називом квантизација. Сабирање једнаке фреквенције дели опсег података на унапред одређени број интервала, од којих сваки има исту количину података. Аналитичар бира колико интервала ће користити. Овај приступ је од помоћи када је добијање бинова једнаке величине изазовно због дистрибуције података. Рецимо да постоји скуп података са не-гаусовом дистрибуцијом која се креће од 0 до 100, и рецимо да се интервал дели на пет бинова. Пет бинова са једнаким бројем резултата у свакој канти биће креирано они могу изгледати овако: 0-40, 40-50, 50-70, 70-90 и 90-100.

### *K-means бининг*

K-means бининг је метода инспирисана K-means алгоритмом за груписање тачака. Податаци се групушу у  $k$  кластера на основу њихове сличности. У контексту дискретизације података, K-means се може користити за партиционисање континуиране променљиве на  $k$  дискретних интервала или бинова. Уколико се K-means визуализује у 2D простору, он би изгледао као на слици 9.



*Слика 9 - K-means алгоритам*

K-means алгоритам функционише на следећи начин:

1. Изаберите број кластера  $k$ .
2. Насумично доделити  $k$  центара кластера.
3. Доделите сваку тачку података најближем центру.
4. Поново израчунајте центре на основу средње вредности тачака података додељених сваком кластеру.
5. Поновите кораке 3 и 4 док се додељивање кластера више не мења. Једном када K-Меанс алгоритам конвергира, границе кластера се могу користити као границе бинова за дискретизацију континуираних атрибута.

Претпоставимо да постоји скуп података о трансакцијама клијената, укључујући укупан износ потрошен у свакој трансакцији. Како вредности трансакције јесу континуалне вредности, рецимо да је неопходно дискретизовати укупан потрошен износ у 5 бинова.

Прво се бира  $k=5$  кластера. Затим насумично додељује 5 центара. Затим свака трансакција се додељује најближем центроиду на основу укупног потрошеног износа. Поново се израчунавају центри на основу средње вредности трансакција додељених сваком кластеру. Понавља се додела и кораци ажурирања центра до конвергенције.



## Надгледана дискретизација података

Методе надгледане дискретизације података користе ознаку класе приликом партиционисања континуалних вредности података. Један од представника метода за надгледану дискретизацију података је дискретизација базирана на ентропији.

### *Дискретизација базирана на ентропији*

Једна од метода надгледане дискретизације назива се дискретизација заснована на ентропији. Метода заснована на ентропији користи ентропију информација о класи кандидата партиције за одабир граница за дискретизацију. Ентропија класне информације је мера чистоће и то мери количину информација која би била потребна да би се назначило којој класи припада инстанца припада. Он разматра један велики интервал који садржи све познате вредности неке карактеристике, а затим рекурзивно дели овај интервал на мање подинтервале до неког критеријума заустављања. На тај начин се постиже оптималан број интервала.

### Случаји употребе и препоруке

Неки од сценарија у којима дискретизација података може бити од помоћи јесте када се ради о великим скуповима података. Дискретизација континуираних вредности може смањити количину меморије потребне за складиштење података и може убрзати одређена израчунавања. Када се ради подацима који садрже шуме, дискретизација може помоћи у смањењу шума и учинити препознавање образаца у скупу лакшим за детекцију. Такође, када се користе алгоритми машинског учења који су засновани на стаблима, дискретне вредности се захтевају на улазу.

Са друге стране, постоје случајеви у којима дискретизација података није неопходна. Обрнуто случају у којем се користи, мали скупови података не захтевају дискретизацију јер дискретизација већ смањи информације о скупу података и тако отежава извлачење релевантних информација. Исто, алгоритми као што су регресија и вештачке неуронске мреже се јако добро носе са континуалним подацима тако да не захтевају дискретизацију истих, док алгоритми базирани на стаблу итекако имају боље перформансе са дискретним вредностима. Дискретизација података зна да буде јако сложен процес који одузима доста ресурса и времена а да са друге стране не помогне у подизању перформанси модела, тако да није препоручљиво да се користи у сваком случају. Генерално, одлука да се користи дискретизација података треба да се заснива на специфичним карактеристикама скупа података и анализе која се спроводи. Важно је пажљиво одмерити потенцијалне предности и недостатке дискретизације пре него што се одлука донесе.

## Промена расподеле

Проблем промене расподеле података је када се статистичке карактеристике скупа података мењају током времена или између различитих група. Ово може довести до тога да модели машинског учења обучени на првобитним подацима током времена, како се додају нове информације и нови подаци, изгубе на перформансама. Шта то значи?

### Опис проблема

Узмимо за пример да приликом развоја модела машинског учења за процену трошкова куће постоје информације о њеној локацији, величини и старости. Прављење тачних предвиђања нових домова на истој локацији захтева обуку модела користећи податке из кућа изграђених у последњих десет година. Индустрија некретнина, међутим, еволуира током времена, а грађевинска предузећа почињу да граде различите врсте кућа са новим карактеристикама и стиливима. Као резултат тога, статистичке карактеристике скупа података се мењају, што може учинити модел машинског учења мање тачним.

Једно од решења је прикупити свеже податке који представљају најновије тржишне информације како би се решила ова промена у дистрибуцији података, а затим поново обучити модел користећи ове нове податке како би се повећала тачност модела. У циљу превазилажења ових потешкоћа, кључно је континуирано пратити расподелу података и поново обучавати моделе користећи најновије информације. Промена расподеле података такође подразумева да ће бити неопходно да се некада поново уравнотеже подаци, скалирају и над њима примене друге технике препроцесирања а затим поново обучи модел. [6]

### Препоруке

- **Пазите на дистрибуцију података** Како би уочили све потенцијалне промене, кључно је држити на оку дистрибуцију података током времена и међу различитим групама. Да бисте то урадили, можете проучити податке помоћу алата за визуелизацију података или проверити дистрибуције помоћу статистичких тестова.
- **Поново избалансирајте податке:** Ако је неравнотежа у подацима крива за промену у дистрибуцији података, један приступ је ребаланс података добијањем додатних информација од недовољно заступљених група или превеликим узорковањем података из ових група.
- **Поново обучите модел:** Да би модел машинског учења био исправан и користан, можда ће бити потребно да га поново обучите на најновијим подацима ако је промена дистрибуције података велика. Ово може захтевати прикупљање свежих података који одражавају промењену дистрибуцију података или коришћење техника као што је активно учење за селективно прикупљање података.

## Енкодирање категоричких атрибута

Енкодирање категоричких вредности је техника трансформације података у машинском учењу која претвара категоричке променљиве у нумеричке вредности које се могу уградити у математичке моделе. Категоричке променљиве, попут пола или позиције у одређеном предузећу, имају мали, типично фиксни опсег потенцијалних вредности.

Да би се категоричке променљиве уопште укључиле у ове моделе, променљиве категорије морају бити кодиране јер алгоритми машинског учења обично захтевају нумеричке податке као улаз. Осим што захтевају нумеричке вредности, алгоритми могу различито радити у зависности од тога како су категоричке променљиве кодиране.

Пре свега, неопходно је знати како се категоричке променљиве деле. Категоричке променљиве се могу поделити у две категорије: номиналне (без одређеног реда) и ординалне (са неким редоследом) [7]. Неколико примера номиналних променљивих:

- Боја (Црвена, жута, ружичаста, плава)
- Животиња (Крава, пас, мачка, змија)

Пример ординалних променљивих:

- Висина (ниска, средња, висока)
- Сагласност („У потпуности се слажем“, Слажем се, Неутрално, Не слажем се и „Уопште се не слажем“)

У вези са тим, свака од техника кодирања има своје предности и мане и користи се у специфичним ситуацијама, и када се говори о техникама кодирања постоје две главне технике:

- Кодирање ознака (eng. Label encoding)
- One-hot кодирање

### Кодирање ознака

Кодирање ознака је техника која се користи за трансформацију категоричких променљивих у нумеричке варијабле. Кодер ознака свакој променљивој додељује јединствену нумеричку вредност. Овај приступ је веома једноставан и подразумева претварање сваке вредности која се нађе у колони у број.

Education	Label
Graduate	1
Masters	2
PHD	3

Слика 10 – Кодирање назива

На пример, ако имамо категоричку променљиву која представља ниво образовања са вредностима као што су основне студије, мастер студије и доктор студије, кодер ознака ће свакој категорији доделити нумеричке вредности попут 0, 1 и 2 као што је приказано на слици 10.


Кодирање ознака је најкорисније када категоричка променљива има јасан редослед или хијерархију између својих категорија, односно када спада у ординалну категорију, која је горе поменута. На пример, у случају редних променљивих као што су „ниско“, „средње“ и „високо“ или „мало“, „средње“ и „велико“, где постоји инхерентно рангирање или редослед категорија, кодирање ознака може бити добар избор.

Насупрот томе, за номиналне варијабле, о којима је такође раније било речи, где не постоји инхерентан редослед између категорија, коришћење кодирања ознака може бити неприкладно јер би увело произвољан редослед међу категоријама, чиме модел машинског учења може неке вредности сматрати битнијима од осталих.

### One-hot кодирање

У one-hot кодирању, свака категорија је представљена као бинарни вектор, где сваки елемент у вектору одговара могућој вредности у категорији. Вектор садржи "1" на позицији која одговара вредности категорије и "0" на свим осталим позицијама. Илустрација рада је приказана на слици

id	color			
1	red			
2	blue			
3	green			
4	blue			



id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

Слика 11 - Илустрација кодирања

Дакле, one-hot кодирање претвара оригинални категоријски атрибут у три

бинарна атрибута, где свака атрибут представља присуство или одсуство одређене категорије (слика 11).

One-hot кодирање је корисна техника за руковање категоричким подацима у машинском учењу и анализи података. Међутим, постоје случајеви у којима коришћење оне-хот кодирања можда није најбоља опција. Ево неколико сценарија у којима би one-hot кодирање могло бити добар избор:

- **Када категорички атрибут има мали број уникатних вредности:** Ако категоричка карактеристика има мали број уникатних вредности, one-hot кодирање може бити ефикасан начин за претварање категоричких података у нумеричке.
- **Када је у питању номинална категоричка променљива:** Ако вредности у категоричком атрибуту немају никакав природни редослед, онда је one-hot кодирање добар избор. На пример, ако имамо категорички атрибут који представља боју са вредностима као што су црвена, зелена и плава, нема инхерентног реда ових вредности.

Са друге стране, постоје случајеви у којима one-hot кодирање можда није најбољи избор и када би боље било применити горе описани кодер ознака:

- **Када категорички атрибут има велики број јединствених вредности:** Ако категорички атрибут има велики број могућих вредности, one-hot кодирање може резултирати са веома великим бројем колона, што може бити рачунарски скупо и може изазвати проблеме са преоптерећењем. Наравно, у овом случају је јако важно знати за који алгоритам машинског учења се подаци припремају, јер за разлику од алгоритама који су осетљиви на димензионалност, постоје и они који то нису.
- **Када се ради о ординалном категоричком атрибуту:** ако су вредности у категоричком атрибуту у редоследу који се може поредити, као што су мало, средње и велико, онда one-hot кодирање можда није најбољи избор, јер не сачувати ред вредности.

### Поређење и препоруке

На крају, кодирање категоричких података је неопходно да би модели машинског учења прецизно обрадили и направили предвиђања на основу таквих података. Важно је напоменути да од свих техника трансформације података, ова техника често буде прва која се примењује, из разлога што доста других техника захтева кодиране вредности на улазу јер не могу радити са категоричким.

One-hot кодирање је прикладно за номиналне податке, док је кодирање назива погодно за ординалне податке. One-hot кодирање је посебно корисно када категорије нису повезане, док је кодирање ознака пожељније када постоји јасан редослед категорија.

Међутим, важно је нагласити да избор технике кодирања на крају зависи од специфичних захтева алгоритма машинског учења који се користи, јер неки алгоритми могу боље да раде са одређеним типовима кодираних података. Поред тога, величина категоријске променљиве и број категорија такође могу бити важна разматрања при избору технике кодирања.

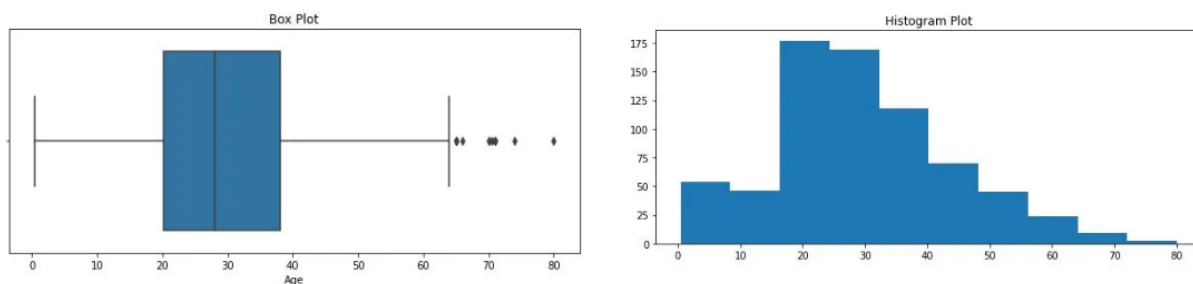
Постоје алгоритми који су осетљиви на димензионалност, док неки то и нису. Алгоритми који нису осетљиви су алгоритми базирани на стаблима, док је велика већина осталих алгоритама у мањој или већој мери осетљива на димензионалност и кодирање скупа података. Исто тако се може рећи за технике препорцесирања, неке боље подносе једну врсту кодирања, док друге другу. Све у свему, треба пажљиво размотрити избор одговарајуће технике кодирања како би се оптимизовале перформансе модела машинског учења на категоријским подацима.

### Уклањање екстремних вредности

Екстремне вредности се дефинишу као вредности које се значајно разликују од осталих вредности атрибута у скупу података. Ове вредности се такође и називају тихим убицама перформанси модела машинског учења из разлога што значајно утичу на основне статистичке параметре [8]. Оне се у скуповима података појављују из више разлога:

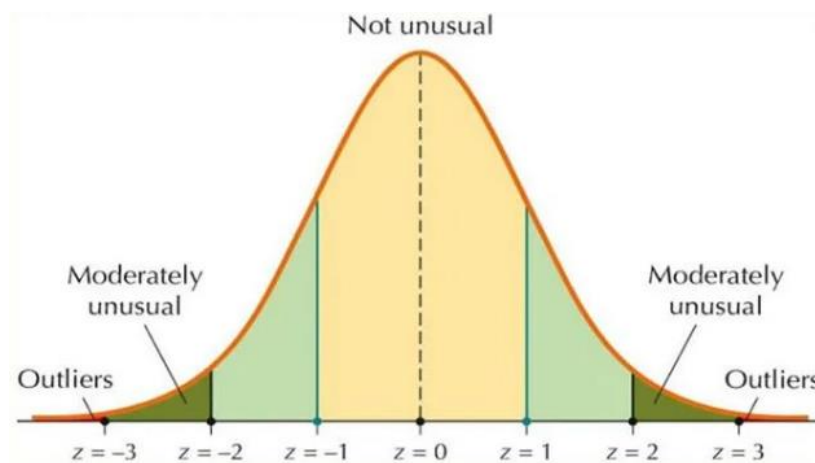
- **Грешке у уносу података:** Људске грешке као што су грешке настале током прикупљања података, снимања или уноса могу да доведу до одступања у подацима.
- **Грешка мерења:** То је најчешћи извор одступања. Ово је узроковано када се испостави да је коришћени мерни инструмент неисправан.
- **Намерно:** Екстремне вредности генерисане са намером за тестирање метода детекције истих
- **Грешке у обради података:** Настале након неуспешне манипулације подацима или ненамерне мутације скупа података
- **Грешке узорковања:** Мешање података из погрешних или различитих извора
- **Природна екстремна вредност** Када екстремна вредност није вештачка (због грешке), то је природна екстремна вредност. Већина података из стварног света припада овој категорији.

Како онда препознати екстремне вредности у скупу података? Постоји доста решења која се односе на детекцију екстремних вредности у скупу података. Неке од техника које су већ детаљно представљене и о којима је било речи су 3-скор и метод интерквартилног опсега. Осим њих, екстремне вредности се могу такође детектовати визуализационим техникама као што је хистограм, бокс плот и сл. и које се могу видети на слици 12.



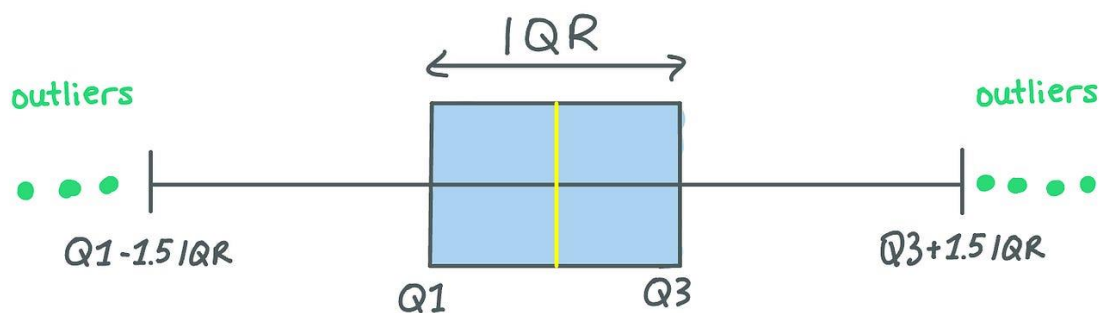
Слика 12 – Визуализација екстремних вредности

Као што је већ напоменуто, о појединим техникама је било речи и сходно томе информације које су већ речене ће се искористити само ради лакшег разумевања примена техника при детекцији екстремних вредности. Када се говори о 3 скору, након што се подаци скалирају, одређује се праг и све вредности које су веће од прага се сматрају екстремним вредностима. Апсолутна вредност прага је увек већа од 2 (најчешће 3, као што је приказано на слици 13).



Слика 13 – Детекција екстремних вредности коришћењем з скорa

Коришћењем интерквartilног опсега (IQR) исто можемо да детектујемо екстремне вредности. IQR нам говори о варијацији у скупу података. Било која вредност која је изван опсега од  $-1.5 \times \text{IQR}$  и преко  $1.5 \times \text{IQR}$  третира се као екстремна вредност. Илустрација ове технике је приказана на слици 14.



Слика 14

Где  $Q1$  представља 1. квартил/25. перцентил података,  $Q2$  представља 2. квартил/медијан/50. перцентил података,  $Q3$  представља 3. квартил/75. перцентил података.  $(Q1-1,5IQR)$  представљаја најмању вредност у скупу података, а  $(Q3+1,5IQR)$  представља највећу вредност у скупу података.

Након што се екстремна вредност детектује, требало би је обрадити на одговарајући начин, јер како је већ речено, екстремне вредности су тихи убица. Екстремне вредности лоше утичу на средњу вредност и стандардну девијацију скупа података. Ово статистички може дати погрешне резултате и самим тим директно утицати на перформансе модела. Већина алгоритама за машинско учење не функционише добро у присуству екстремних вредности у подацима. Дакле, пожељно је открити и што пре поступити на одговарајући начин.

#### Руковање са екстремним вредностима

- **Брисање екстремних вредности** Као што сам назив технике говори, када се екстремна вредност детектује, она се брише.
- **Импутација** У овом случају, екстремне вредности се третирају исто као и вредности које недостају – мењају се средњом вредношћу, медијаном или регресијом.
- **Трансформација вредности** подразумева неку врсту операција над подацима и пре свега се односи на технике скалирања података о којима је било речи. Ове технике смањују утицај екстремних вредности и поправљају лошу дистрибуцију података и то све без губитка информација.

Како се технике значајно разликују, треба узети у обзир неколико разматрања пре доношења одлуке који ће се приступ користити.

- **Квалитет података:** Ако је разлика настала због грешака у уносу података или других проблема са квалитетом података, брисање података би био добар избор.
- **Дистрибуција података:** Ако је разлика настала због дистрибуције података, проблем би требало да реши трансформација података. Ово може помоћи да се смањи утицај екстремних вредности на анализу.
- **Величина узорка:** Ако је скуп података велики, брисање неће утицати на анализу. Међутим, ако је скуп података мали, уклањање екстремних вредности може имати значајан утицај на величину узорка и статистичке параметре. У овим случајевима може бити прикладније трансформисати податке или користити технике импутације.
- **Сврха анализе:** Одговарајући приступ за поступање са екстремним вредностима такође може зависити од сврхе анализе. На пример, уколико су крајњи циљ тачна предвиђања, импутирању вредности које недостају је најприкладнија техника. С друге стране, уколико се врши идентификација образаца или односа у подацима, уклањање или трансформисање може бити прикладније решење.



## Случаји употребе и препоруке

Укратко, одговарајући приступ за поступање са екстремним вредностима зависиће од мноштво фактора. Важно је пажљиво размотрити ове факторе и одабрати одговарајући приступ. Уопштено говорећи, одговарајућа техника за поступање са екстремним вредностима зависиће од природе података и специфичне анализе која се обавља.

У неким случајевима, одступања могу бити важне и информативне тачке података и не треба их уклањати из скупа података. Међутим, у другим случајевима, одступања могу бити последица грешака или аномалног понашања и можда нису репрезентативни за дистрибуцију основних података. Важно је напоменути да, иако уклањање одступања може побољшати перформансе неких алгоритама за машинско учење, то такође може довести до губитка информација и то треба радити са опрезом.

## Закључак

Главна идеја водиља приликом истраживања и писања семинарског рада јесте направити приручник који ће објаснити важност техника трансформације података и њихов утицај на перформансе модела машинског учења. Било је речи о техникама скалирања које се могу поделити на стандардизацију и нормализацију, као и о техникама енкодирања, уклањања екстремних вредности, дискретизације података и промене расподеле података. Свака од ових техника има свој случај употребе и носи са собом мање или више неке добре и лоше ствари.

Неки од главних закључака које се могу донети након овог истраживања и на чему се потенцира кроз цео рад је да је сваки скуп података различит од других и да нема идеалног случаја примене одређених техника. Свакако, на основу структуре скупа података, циља обраде и алгоритма који се користи се може изабрати одређена трансформација, али је и даље најбољи савет пробати више различитих и применити на скупу података и тестирати перформансе.

На крају, као кратак осврт на најбитније ствари које су поменуте у овом семинарском раду, може се рећи да ће **кодираре података** у великом броју случајева бити прва примењена техника трансформације података јер већина алгоритма машинског учења захтева нумеричке вредности на улазу. Такође, најпознатије технике кодираре су **кодираре ознака** и **one-hot кодираре**. Препоруке су да се кодираре ознака користи када су у питању вредности које представљају неки поредак и када се подаци припремају за алгоритам који не трпи велику димензионалност. Са друге стране one-hot кодираре се користи када не постоји поредак. Проблем који one-hot носи са собом јесте повећање димензионалности, па треба бити опрезан са њим. На крају, иако су ово препоруке, у пракси различитим скуповима и алгоритмима одговара различит начин кодираре па је препорука да се ипак проба више начина кодираре и упореди у циљу добијања правог скупа над коме модел има добре перформансе. **Алгоритми** који нису осетљиви на кодираре јесу **алгоритми базирани на стаблу**, док се код других алгоритама примењује разлика у перформансама модела након кодираре, те је пожељно кодирати податке.

**Скалирање података** подразумева две технике **нормализацију** и **стандардизацију**. Расподела података утиче на одабир једне од ове две технике. Уколико је у питању гаусова расподела препорука је да се користи стандардизација док се у другим случајевима подаци нормализују. Скалирање је итекако корисна техника, па алгоритми као што су SVM, KNN и вештачке неуронске мреже итекако дају боље резултате када се подаци скалирају. Са друге стране, алгоритми базирани на стаблу као што је метод насумичних шума и наивни бајес не дају боље резултате након скалирања.

Када се говори о **дискретизацији података**, то је једна врло осетљива техника и са њоме треба бити опрезан. Добре стране дискретизације јесу представљање података на нивоу знања, давање смисла неким подацима, смањење меморијског простора пре свега када су у питању велики скупови података. Ипак, дискретизација смањује скуп података па се њоме и доста информација губи, због чега никако није погодна за мање скупове података. Опис технике указује на то да је веома добра основа за алгоритме базиране на стаблу који захтевају конкретну вредност, па је

препоручено дискретизовати податке за технику као што је метод насумичних шума. Са друге стране, вештачке неуронске мреже се јако добро носе са континуалним подацима, те би дискретизација значила само губитак информација па самим тим и лоше перформансе.

**Екстремне вредности** су тихе убице свих алгоритама машинског учења и генерално је препорука позабавити се њима. До екстремних вредности у скупу података се може доћи на различите начине, а најчешћи случај су природне екстремне вредности. Када су оне у питању треба бити врло опрезан јер екстремне вредности иако су ретке, саставни су део скупа података. Са друге стране, неопходно је уклонити свако аномалично понашање и екстремне вредности сличног типа. Технике које се користе за руковање екстремних вредности су брисање екстремних вредности и импутација која екстремне вредности мења средњом вредношћу. У зависности од величине скупа и битности информација екстремне вредности ће се брисати или мењати средњом вредношћу.

До **промене расподеле**, мало специфичнијег проблема, долази када се скуп података ажурира вредностима које су генерисане под мало другачијим околностима. Те вредности утичу на основне статистичке параметре на основу којих су се модели тренирали што може допринети томе да модел има лоше перформансе над ажурираном скупу података. Овде је нагласак на томе да до тога може доћи у да у складу са тим се скуп података поново треба обрадити.

## References

- [1] "Geeks for geeks," [Online]. Available: <https://www.geeksforgeeks.org/data-transformation-in-data-mining/>.
- [2] "FinancesOnline," [Online]. Available: [https://financesonline.com/how-much-data-is-created-every-day/#:~:text=Every%20day%20Big%20Data%20statistics,\(2021%2C%20February%209\)](https://financesonline.com/how-much-data-is-created-every-day/#:~:text=Every%20day%20Big%20Data%20statistics,(2021%2C%20February%209))  
..
- [3] B. Roy, "Towards Data Science," 2020. [Online]. Available: <https://towardsdatascience.com/all-about-feature-scaling-bcc0ad75cb35>.
- [4] A. DEY, "Kaggle," 2021. [Online]. Available: <https://www.kaggle.com/code/aimack/complete-guide-to-feature-scaling>.
- [5] "JavaTpoint," [Online]. Available: <https://www.javatpoint.com/discretization-in-data-mining>.
- [6] S. Viquez, "nannyML," [Online]. Available: <https://www.nannyml.com/blog/6-ways-to-address-data-distribution-shift>.
- [7] M. S. Rohith, "DevGenius," [Online]. Available: <https://blog.devgenius.io/encoding-methods-to-encode-categorical-data-in-machine-learning-717b5509933c>.
- [8] A. Suresh, "Medium," 2020. [Online]. Available: <https://medium.com/analytics-vidhya/how-to-remove-outliers-for-machine-learning-24620c4657e8>.