8-дәріс.

Интеллектуалды жүйелердегі машиналық оқытудың негіздері

Пәні: Интеллектуалды ақпараттық жүйелер Ұзақтығы: 50 минут Форматы: теориялық түсіндіру, мысалдарды талдау, алгоритмдерді салыстыру және шағын практикалық тапсырма Алдыңғы тақырып: Логикалық шығару механизмдері

Дәрістің мақсаты

Студенттерге машиналық оқытудың теориялық негіздерін, оның интеллектуалды ақпараттық жүйелердегі рөлін, негізгі түрлерін, модельді оқыту кезеңдерін және нәтижелерді бағалау әдістерін түсіндіру.

Күтілетін оқу нәтижелері: дәріс соңында студенттер машиналық оқытудың анықтамасын түсіндіре алады, оны ережелерге негізделген интеллектуалды жүйелермен салыстырады, бақыланатын және бақыланбайтын оқытуды ажыратады, классификация мен регрессия есептерін анықтайды, оқыту және тестілеу деректерінің рөлін түсінеді, модель сапасын бағалаудың негізгі метрикаларын қолданады.

1. Кіріспе. Логикалық қорытындыдан машиналық оқытуға көшу

Алдыңғы дәрістерде біз интеллектуалды ақпараттық жүйелердің білімді қалай сақтайтынын және оны қалай пайдаланатынын қарастырдық.

Логикалық модельдерде білім фактілер мен формальды тұжырымдар арқылы сипатталды. Өнімдік модельдерде «ЕГЕР – ОНДА» ережелерін қолдандық. Сараптамалық жүйелерде маманның білімін компьютерге енгізіп, логикалық шығару механизмінің көмегімен қорытынды алдық.

Бірақ осы жерде жаңа мәселе пайда болады.

Егер шешім қабылдау ережелері алдын ала белгісіз болса немесе оларды нақты жазу өте қиын болса, интеллектуалды жүйе қалай жұмыс істеуі керек?

Мысалы, университетте студенттердің оқу үлгерімін талдайтын интеллектуалды жүйе бар делік.

Бізде соңғы бірнеше жылдағы студенттер туралы ақпарат сақталған:

- GPA көрсеткіштері;
- сабаққа қатысу пайызы;
- тапсырмаларды орындау мерзімдері;
- академиялық қарыздар саны;
- аралық бақылау нәтижелері;
- семестр соңындағы академиялық нәтижелер.

Алдыңғы дәрістерде қарастырған тәсіл бойынша сарапшы белгілі ережелерді құрастыра алады.

Мысалы:

ЕГЕР GPA 2.0-ден төмен ЖӘНЕ академиялық қарыз саны екіден көп болса, ОНДА студент академиялық тәуекел тобына жатады.

Бұл ереже түсінікті және оны компьютер оңай орындайды.

Алайда нақты өмірде жағдай күрделірек болуы мүмкін.

GPA көрсеткіші **3.2** болатын студенттің сабаққа қатысуы төмендеуі, тапсырмаларды кеш тапсыруы және соңғы бақылау нәтижелері нашарлауы мүмкін. Ал **GPA** көрсеткіші **2.5** болатын басқа студенттің оқу нәтижелері тұрақты түрде жақсарып отыруы ықтимал.

Осы екі студенттің қайсысына қосымша академиялық қолдау қажет?

Мұндай шешімді бір немесе екі қарапайым ережемен қабылдау әрдайым тиімді емес.

Біз бұрынғы студенттердің деректерін және олардың нақты академиялық нәтижелерін талдап, белгілі заңдылықтарды анықтай аламыз.

Мысалы, жүйе өткен семестрлердегі мәліметтерден тапсырмаларды кеш тапсыру жиілігі, үлгерім динамикасы және сабаққа қатысу өзгерісі академиялық тәуекелмен байланысты екенін анықтауы мүмкін.

Бұл жағдайда компьютерге барлық ережелерді алдын ала қолмен жазудың орнына, деректерден заңдылықтарды үйренуге мүмкіндік береміз.

Осы тәсіл машиналық оқыту деп аталады.

2. Машиналық оқыту дегеніміз не?

Машиналық оқыту (Machine Learning, ML) — компьютерлік жүйелердің деректерден заңдылықтар мен байланыстарды анықтап, солардың негізінде болжау, жіктеу немесе шешім қабылдауға арналған модельдерді құру әдістерін зерттейтін жасанды интеллект саласы.

Машиналық оқытудың негізгі ерекшелігі — жүйе барлық шешім ережелерін программистен дайын күйінде алмайды. Ол деректердегі статистикалық байланыстарды оқыту алгоритмі арқылы анықтайды.

Қарапайым түрде:

Деректер-Бұрынғы бақылаулар мен нәтижелер

Оқыту алгоритмі-Заңдылықтарды іздеу және параметрлерді өзгерту

Оқытылған модель-Жаңа деректерге болжам жасау

Машиналық оқытудың маңызды ерекшеліктері

Мысалы, университет жүйесіне бұрынғы студенттердің көрсеткіштері және семестр соңындағы нәтижелері беріледі.

Оқыту алгоритмі осы мәліметтер арасындағы байланысты зерттейді. Нәтижесінде жаңа студенттің академиялық қиындыққа тап болу ықтималдығын бағалайтын модель жасалуы мүмкін.

Бірақ маңызды нәрсені түсінуіміз керек: машиналық оқыту деректердегі байланысты табады, алайда анықталған байланыс міндетті түрде себеп-салдарлық тәуелділік болып саналмайды.

Мысалы, сабаққа қатысу мен жоғары үлгерім арасында байланыс болуы мүмкін. Бірақ тек осы байланысқа сүйеніп, барлық басқа факторларды елемеуге болмайды.

Машиналық оқытудың нәтижесі бастапқы деректердің сапасына, таңдалған алгоритмге және модельді дұрыс бағалауға тәуелді.

3. Дәстүрлі программалау мен машиналық оқытудың айырмашылығы

Дәстүрлі программалауда программист есепті шешудің алгоритмін өзі анықтайды.

Мысалы, студенттің бағасы 50 балдан жоғары немесе тең болса, оны «өтті» деп белгілейтін программа құрастырайық.

```
grade = 75
    if grade >= 50:
                    print("Өтті")
else:
     print("Өтпеді")
```

Бұл жерде программа ештеңені үйренбейді.

Ол алдын ала берілген шартты орындайды.

Ал машиналық оқытуда біз көптеген мысалдар мен олардың нәтижелерін пайдаланып, модельдің параметрлерін анықтаймыз.

Дәстүрлі программалау	**Машиналық оқыту**
Ережелерді программист жазады	Заңдылықтар деректерден үйретіледі
Деректер мен ережелер беріледі	Оқыту деректері мен алгоритм беріледі
Нәтиже анықталған алгоритм бойынша есептеледі	Нәтиже оқытылған модель бойынша болжанады
Ережені өзгерту үшін код өзгеруі мүмкін	Модельді жаңа деректермен қайта оқытуға болады
Мысал: балл бойынша өту шартын тексеру	Мысал: студенттің келесі семестрдегі тәуекелін болжау

Мысалы, университеттің ресми ережесі бойынша емтиханға жіберілу үшін белгілі талаптарды орындау міндетті болса, мұндай нормативтік шартты машиналық модельге болжатып қоюдың қажеті жоқ.

Ал көптеген көрсеткішке сүйеніп, болашақ академиялық тәуекелді болжау қажет болса, машиналық оқыту пайдалы болуы мүмкін.

Осылайша интеллектуалды жүйелерде логикалық ережелер мен машиналық оқыту бір-бірін толықтыра алады.

4. Машиналық оқытудың даму тарихы

Машиналық оқыту бір күнде пайда болған технология емес. Оның қалыптасуына математикалық статистика, ықтималдықтар теориясы, алгоритмдер теориясы, нейробиология және есептеу техникасының дамуы әсер етті.

1950 жылы Алан Тьюринг «Компьютер ойлай ала ма?» деген ғылыми мәселені көтерді. Ол машинаның интеллектуалды әрекетін зерттеуге маңызды негіз қалады.

1957 жылы Фрэнк Розенблатт перцептрон моделін ұсынды. Перцептрон — кіріс белгілерін пайдаланып, қарапайым жіктеу тапсырмасын орындай алатын жасанды нейрон моделі.

1959 жылы Артур Сэмюэл Machine Learning терминін қолданды. Ол дойбы ойнайтын компьютерлік программаларды зерттеп, машинаның тәжірибе арқылы нәтижесін жақсарту мүмкіндігін көрсетті.

Кейін шешім ағаштары, статистикалық жіктеу әдістері, нейрондық желілер және басқа алгоритмдер дамыды.

XXI ғасырда үлкен көлемдегі деректердің қолжетімді болуы, графикалық процессорлардың есептеу қуаты және жаңа алгоритмдер машиналық оқытудың кең таралуына ықпал етті.

Бүгінде машиналық оқыту интеллектуалды ақпараттық жүйелерде, компьютерлік көруде, мәтінді өңдеуде, қаржылық талдауда, өндірістік диагностикада және ұсыным жүйелерінде қолданылады.

Алайда машиналық оқытудың барлық әдісі нейрондық желілерге негізделмейді. Көптеген практикалық есептерді қарапайым статистикалық модельдер немесе шешім ағаштары арқылы да шешуге болады.

5. Машиналық оқытудың негізгі ұғымдары

Машиналық оқыту алгоритмдерін түсіну үшін бірнеше негізгі терминді ажырату қажет.

Деректер жиыны (Dataset) — модельді оқытуға, тексеруге немесе қолдануға арналған бақылаулар жиынтығы.

Объект (Sample, Instance) — деректер жиынындағы жеке бақылау. Мысалы, бір студент туралы жазба.

Белгі (Feature) — объектіні сипаттайтын өлшенетін немесе кодталатын қасиет. Мысалы, GPA, сабаққа қатысу пайызы немесе тапсырманы кеш тапсыру саны.

Мақсатты айнымалы (Target) — модель болжауы қажет нәтиже. Мысалы, студенттің академиялық тәуекел тобына жатуы.

Модель (Model) — кіріс белгілерін шығыс нәтижемен байланыстыратын, оқыту барысында параметрлері анықталатын математикалық немесе алгоритмдік құрылым.

Оқыту (Training) — модель параметрлерін деректерге сүйене отырып анықтау процесі.

Мысал ретінде шартты университеттік деректер жиынын қарастырайық.

Студент	GPA	Қатысуы	Кеш тапсырған жұмыстар	Академиялық тәуекел
A	3.45	92%	0	Төмен
B	2.35	71%	4	Жоғары
C	3.10	78%	3	Төмен
D	2.80	63%	6	Жоғары
E	2.95	88%	1	Төмен
F	3.25	68%	5	Жоғары

Бұл — оқыту принципін түсіндіруге арналған жасанды мысал. Нақты модельді сенімді бағалау үшін әдетте әлдеқайда көп дерек қажет.

Кестеде **GPA**, қатысу пайызы және кеш тапсырылған жұмыстар саны — кіріс белгілері.

Ал академиялық тәуекел — мақсатты айнымалы.

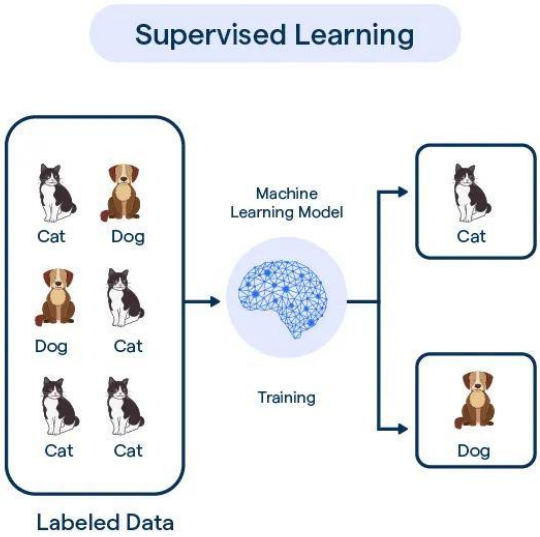
Модельдің міндеті — осы белгілер мен мақсатты айнымалы арасындағы байланысты үйрену.

Егер жаңа студенттің GPA көрсеткіші 3.05, қатысуы 66%, кеш тапсырған жұмыстары 5 болса, модель оның тәуекел деңгейін бағалай алады.

Бірақ модель шешімі міндетті түрде дұрыс болады деген кепілдік жоқ. Сондықтан оқыту нәтижесін бөлек деректермен тексеру қажет.

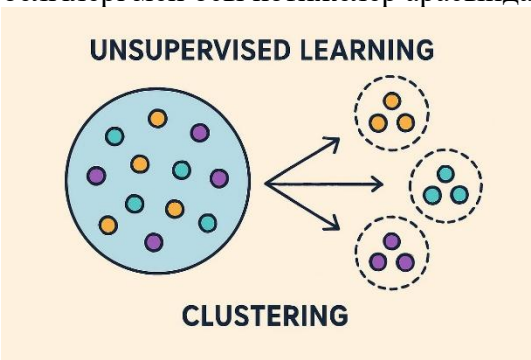
6. Машиналық оқытудың негізгі түрлері

Машиналық оқытуды ұйымдастырудың бірнеше тәсілі бар. Олардың айырмашылығы — оқыту кезінде жүйеге қандай ақпарат берілетіні және қандай мақсат қойылатынында.



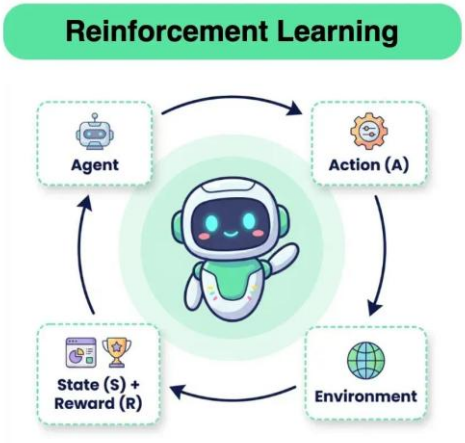
1. Бақыланатын оқыту (Supervised Learning)

Әрбір оқыту объектісі үшін дұрыс нәтиже немесе мақсатты мән белгілі. Модель кіріс белгілері мен осы нәтижелер арасындағы байланысты үйренеді.



2. Бақыланбайтын оқыту (Unsupervised Learning)

Алдын ала берілген мақсатты белгілер жоқ. Алгоритм деректердің ішкі құрылымын, ұқсастықтарын және топтарын анықтауға тырысады.



3. Нығайтумен оқыту (Reinforcement Learning)

Агент ортамен әрекеттесіп, әрекеттерінің нәтижесінде марапат немесе кері байланыс алады. Оның мақсаты — уақыт өте жинақталған марапатты арттыратын стратегияны үйрену.

Бұлар машиналық оқытудың негізгі бағыттары. Сонымен бірге жартылай бақыланатын оқыту, өздігінен бақыланатын оқыту және басқа тәсілдер де бар.

7. Бақыланатын оқыту — Supervised Learning

Бақыланатын оқытуда алгоритмге кіріс деректерімен бірге дұрыс жауаптар беріледі.

Мысалы, университетте өткен жылдардағы студенттердің оқу көрсеткіштері және семестр соңындағы нәтижелері бар делік.

Жүйеге студенттің GPA көрсеткіші, сабаққа қатысуы, тапсырма орындау тәртібі және академиялық тәуекел деңгейі беріледі.

Оқыту барысында алгоритм қандай белгілер қандай нәтижелермен байланысты екенін анықтайды.

Жаңа студент туралы мәлімет түскен кезде модель оның нәтижесін болжайды.

Бақыланатын оқытудың екі маңызды тапсырмасы бар:

Классификация (Classification) — объектіні алдын ала анықталған санаттардың біріне жатқызу.

Мысалы, электрондық хатты «спам» немесе «спам емес» деп жіктеу.

Регрессия (Regression) — үздіксіз сандық мәнді болжау.

Мысалы, ғимараттың энергия тұтынуын немесе студенттің келесі бақылауда алуы мүмкін балын болжау.

Критерий	**Классификация**	**Регрессия**
Нәтиже түрі	Санат	Сандық мән
Мысал	Жоғары/төмен тәуекел	Болашақ балл
Шығыс	Класс немесе класс ықтималдығы	Сан
Алгоритм мысалы	Логистикалық регрессия	Сызықтық регрессия

Мұнда атауларға назар аудару керек: логистикалық регрессия атауында «регрессия» сөзі болғанымен, оның кең таралған қолданылуы — классификация.

8. Регрессия және сызықтық регрессия

Регрессия есептерінде модель белгілі белгілер негізінде сандық мәнді болжайды.

Мысалы, студенттің дайындыққа жұмсаған уақыты мен тест нәтижесі арасындағы байланысты зерттейік.

Дайындық уақыты (сағат)	**Тест балы**
1	48
2	54

3	61
4	66
5	73

Бұл мысалда дайындық уақыты көбейген сайын тест нәтижесінің жоғарылау үрдісі байқалады.
Осындай байланысты сипаттаудың қарапайым тәсілі — сызықтық регрессия.
Оның формуласы:

$$\hat{y} = wx + b$$

Мұндағы:

x — кіріс белгісі;

$\hat{y}$ — модель болжаған нәтиже;

w — белгінің салмақ коэффициенті;

b — бос мүше.

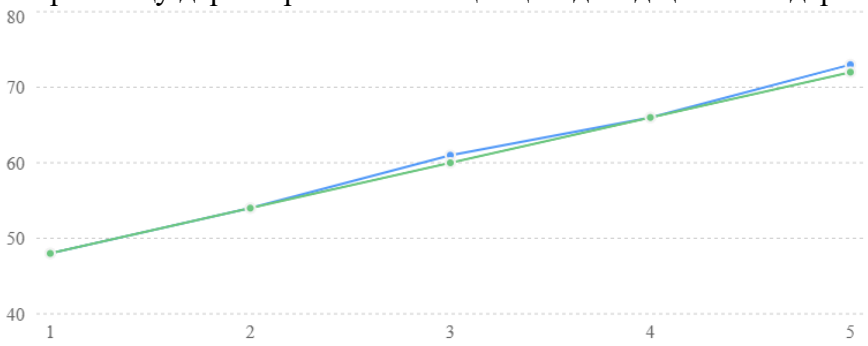
Мысалы, оқыту нәтижесінде шартты түрде:

$$\hat{y} = 6x + 42$$

моделі алынды делік.
Егер студент 4 сағат дайындалса:

$$\hat{y} = 6 \cdot 4 + 42 = 66$$

Модель 66 балл болжайды.
Дайындық уақыты мен тест нәтижесінің байланысы
Шартты оқу деректері және сызықтық модельдің болжамдары.



■ Нақты балл
■ Модель болжамы
Графикте нақты бақылаулар мен модельдің болжамдары бір-біріне жақын орналасқан.
Бірақ сызықтық регрессия барлық тәуелділікті дәл сипаттай бермейді.
Мысалы, дайындық уақыты 20 сағатқа жетсе, тест нәтижесі міндетті түрде сызықтық түрде өсе бермейді. Сонымен қатар ұйқы, бұрынғы білім және тапсырманың күрделілігі сияқты басқа факторлар да әсер етеді.
Сондықтан модельді тек бірнеше бақылауға сүйеніп бағалауға болмайды.

9. Классификация және логистикалық регрессия

Классификация кезінде модель объектінің белгілі бір класқа жататынын болжайды.
Мысалы, электрондық оқыту платформасында студенттің тапсырманы уақытында аяқтай алмау тәуекелін анықтау қажет делік.
Модель келесі белгілерді пайдалана алады:

- соңғы тапсырмалардағы белсенділік;

- кешіктірілген жұмыстар саны;
- оқу материалдарын қарау жиілігі;
- алдыңғы бақылау нәтижелері.

Егер екі класс болса, оны бинарлық классификация деп атаймыз.
Мысалы:

$$y \in \{0, 1\}$$

Мұнда 0 — тәуекел төмен, 1 — тәуекел жоғары.
Логистикалық регрессияда жиі қолданылатын функция — сигмоида:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Бұл функция кез келген нақты санды 0 мен 1 аралығындағы мәнге түрлендіреді.
Мысалы, модель:

$$P(y = 1 \mid x) = 0.78$$

нәтижесін берсін.
Бұл модельдің берілген белгілер үшін оң класқа қатысты бағалауы 0.78 екенін көрсетеді. Оны сенімді ықтималдық ретінде түсіндіру үшін модельдің калибрленуін де тексеру қажет.
Егер классификация шегі 0.5 деп таңдалса, жүйе объектіні 1-класқа жатқызады.
Бірақ шек міндетті түрде 0.5 болуы шарт емес. Қате шешімдердің салдарына қарай оны өзгертуге болады.
Мысалы, академиялық қолдау қажет студентті байқамай қалудың салдары жалған ескерту жіберуден маңыздырақ болуы мүмкін.
Сондықтан классификация шегін таңдағанда жүйенің нақты мақсаты ескеріледі.
Мысалы, біз студенттің оқу үлгеріміне қарап, оның тәуекел тобында бар-жоғын анықтағымыз келеді:

- **0 — тәуекел төмен**;
- **1 — тәуекел жоғары**.

Жүйе шешім қабылдау үшін студенттің:

- тапсырмаларды қаншалықты белсенді орындайтынын;
- сабаққа қаншалықты жиі кешігетінін;
- оқу материалдарын қаншалықты жиі қарайтынын зерттейді.

Сигмоида функциясы осы мәліметтерді 0 мен 1 арасындағы ықтималдыққа айналдырады.
Мысалы, жүйе:
P(y = 1 | x) = 0,78
деген нәтиже берсе, бұл студенттің тәуекел тобына түсу ықтималдығы **78%** дегенді білдіреді.
Содан кейін шекті мән қолданылады. Әдетте ол **0,5** болады:

- ықтималдық **0,5-тен жоғары болса** — студент **1-классқа**, яғни тәуекелі жоғары топқа жатады;
- ықтималдық **0,5-тен төмен болса** — студент **0-классқа**, яғни тәуекелі төмен топқа жатады.

Қарапайым айтқанда, **сигмоида студенттің тәуекел ықтималдығын есептейді, ал бинарлық классификация оны екі топтың біріне бөледі.**

10. Бақыланбайтын оқыту — Unsupervised Learning
Бақыланбайтын оқытуда деректерге алдын ала дұрыс жауаптар берілмейді.
Алгоритмнің мақсаты — объектілер арасындағы ұқсастықтарды, айырмашылықтарды немесе жасырын құрылымды анықтау.

Мысалы, университеттің электрондық оқыту платформасында студенттердің белсенділігі туралы деректер бар:

- жүйеге кіру жиілігі;
- оқу материалдарын қарау саны;
- тапсырма орындауға жұмсалған уақыт;
- форумдағы белсенділік.

Бірақ студенттердің қандай топтарға бөлінетіні алдын ала белгісіз.
Бақыланбайтын оқыту алгоритмі осы көрсеткіштерге сүйеніп, ұқсас студенттерді топтастыра алады.
Мысалы, талдау нәтижесінде шартты түрде үш топ анықталуы мүмкін:

1. Жүйеге жиі кіріп, материалдарды тұрақты қарайтын студенттер.
2. Тапсырмаларды орындайтын, бірақ оқу материалдарын сирек қарайтын студенттер.
3. Белсенділігі тұрақсыз студенттер.

Бұл топтар алгоритмнің өзі анықтаған ұқсастықтарға негізделеді. Оларды бірден «үздік», «орташа» немесе «нашар» студенттер деп атау дұрыс емес.
Алынған топтардың мағынасын зерттеуші деректерді талдау арқылы түсіндіруі қажет.

11. Кластерлеу және K-means алгоритмі

Кластерлеу (Clustering) — объектілерді ұқсастығына қарай топтарға бөлу әдісі.
Кең таралған алгоритмдердің бірі — K-means.
Бұл слайдта **K-means алгоритмі мәліметтерді ұқсастығына қарай бірнеше топқа қалай бөлетіні** түсіндіріледі.
Мысалы, студенттерді оқу белсенділігіне қарай **3 топқа** бөлгіміз келеді:

- белсенділігі жоғары;
- белсенділігі орташа;
- белсенділігі төмен.

Алгоритмнің жұмысы:

1. Алдымен неше топ қажет екенін анықтаймыз. Мысалы, **K = 3**.
2. Әр топқа уақытша бір орталық нүкте таңдалады. Бұл нүкте **центроид** деп аталады.
3. Әр студенттің көрсеткіштері қай орталыққа жақын болса, сол топқа қосылады.
4. Студенттер бөлінгеннен кейін әр топтың жаңа орташа орталығы есептеледі.
5. Топтар өзгермей қалғанша бұл әрекет бірнеше рет қайталанады.

Қашықтық дегеніміз — екі объектінің бір-біріне қаншалықты ұқсас немесе жақын екенін көрсететін өлшем. Қашықтық аз болса, объектілер бір-біріне ұқсас.

Масштабтау не үшін керек?

Мысалы, бір көрсеткіш 0–100 аралығында, ал екіншісі 0–10 000 аралығында болса, үлкен сандар алгоритмге көбірек әсер етеді. Сондықтан барлық көрсеткіштерді шамалас өлшемге келтіреміз.
Шектеуі: K-means бастапқы орталықтарды қалай таңдағанымызға, көрсеткіштердің масштабына және өте ерекше мәндерге сезімтал.
Қарапайым айтқанда: **K-means — ұқсас объектілерді бір топқа жинайтын алгоритм.**
ең қарапайым мысалмен түсіндірейін.
Бізде 6 студент бар делік. Олардың аптасына платформаға неше рет кіретіні белгілі:

- Аружан — 2 рет
- Данияр — 3 рет
- Айбек — 4 рет
- Мадина — 15 рет
- Әлия — 17 рет
- Ержан — 18 рет

Біз студенттерді **екі топқа** бөлгіміз келеді. Сондықтан **K = 2** деп аламыз.

K-means қалай жұмыс істейді?

Алгоритм студенттерді көрсеткіштері бір-біріне жақын болуына қарай бөледі:

1-топ: белсенділігі төмен студенттер

- Аружан — 2 рет
- Данияр — 3 рет
- Айбек — 4 рет

Олардың орташа мәні:
\[(2+3+4)/3=3 \]
Демек, бірінші топтың орталығы — **3**.
2-топ: белсенділігі жоғары студенттер
- Мадина — 15 рет
- Әлия — 17 рет
- Ержан — 18 рет

Олардың орташа мәні:
\[(15+17+18)/3=16,7 \]
Екінші топтың орталығы — шамамен **17**.
Енді жаңа студент платформаға аптасына **5 рет** кірсе, ол 3 санына жақын болғандықтан, **бірінші топқа** қосылады.
Ал аптасына **16 рет** кірсе, 17 санына жақын болғандықтан, **екінші топқа** қосылады.
«Қашықтық» деген не?
Қашықтық — көрсеткіштердің айырмашылығы.
Мысалы, студент платформаға 5 рет кірді:
- бірінші топтың орталығына дейін: \(5-3=2\);
- екінші топтың орталығына дейін: \(17-5=12\).

5 саны 3-ке жақын. Сондықтан студент бірінші топқа қосылады.
Масштабтау не үшін керек?
Студенттерді екі көрсеткішпен салыстырайық:
- сабаққа қатысу — **0-ден 100%-ға дейін**;
- платформадағы әрекеттер саны — **0-ден 10 000-ға дейін**.

Мысалы:
- қатысуы — 90%;
- платформадағы әрекеттері — 8 000.

8 000 саны 90 санынан өте үлкен. Сондықтан алгоритм негізінен платформадағы әрекеттерге қарап топтастыруы мүмкін. Қатысу көрсеткішінің әсері азайып қалады.
Осыны болдырмау үшін екі көрсеткішті де бірдей аралыққа, мысалы **0-ден 1-ге дейін**, келтіреміз. Бұл — **масштабтау**.
Өте қысқа қорытынды
K-means — ұқсас студенттерді бір топқа жинайды. Мысалы:
- белсенділігі төмен студенттер;
- белсенділігі орташа студенттер;
- белсенділігі жоғары студенттер.

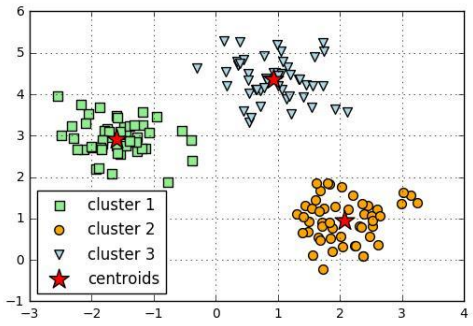
Алгоритмге бұл топтардың атауын алдын ала бермейміз. Ол студенттердің көрсеткіштерін салыстырып, топтарды **өзі қалыптастырады**.
K-means алгоритмінде кластерлер саны K алдын ала беріледі.
Алгоритмнің негізгі қадамдары:
1. Кластерлер саны K таңдалады.
2. Бастапқы кластер орталықтары анықталады.
3. Әр объект ең жақын орталыққа тағайындалады.
4. Әр кластердің орталығы қайта есептеледі.
5. Тағайындаулар немесе орталықтар тұрақталғанша процесс қайталанады.

Екі белгі бойынша қашықтықты есептеу үшін Евклид қашықтығын қолдануға болады:

$$d(A,B)=\sqrt{(x_A-x_B)^2+(y_A-y_B)^2}$$

Мысалы, екі студенттің жүйеге кіру жиілігі және тапсырма орындау белсенділігі бойынша ұқсастығын есептеуге болады.

Бірақ бір белгі 0–100 аралығында, екіншісі 0–10 000 аралығында болса, үлкен масштабтағы белгі қашықтыққа көбірек әсер етеді.
Сондықтан K-means алдында белгілерді масштабтау жиі қажет болады.



rasbt.github.io
Кластерлеу идеясы: бір-біріне ұқсас объектілер топтарға біріктіріледі, ал әр топтың орталығы есептеледі.
K-means алгоритмінің шектеулері де бар. Ол кластерлер санына, бастапқы орталықтарға, масштабтауға және шеткі мәндерге сезімтал болуы мүмкін.
Сондықтан барлық деректер жиынына бірдей қолайлы емес.

12. Нығайтумен оқыту — Reinforcement Learning

Марапат арқылы оқыту дегеніміз — компьютердің әрекет жасап, қателесіп, тәжірибе арқылы дұрыс шешім қабылдауды үйренуі.
Оны баланы үйретумен салыстыруға болады:

- бала дұрыс әрекет жасаса — мақтаймыз немесе сыйлық береміз;
- қате әрекет жасаса — ескерту жасаймыз;
- бала бірнеше рет қайталап, қай әрекеттің дұрыс екенін түсінеді.

Мысал: роботтың лабиринттен шығуы
Робот лабиринттің ішінде тұр делік. Оның мақсаты — шығатын есікті табу.
1. Агент (Agent) — шешім қабылдайтын объект.
Біздің мысалда агент — **робот**.
2. Орта (Environment) — агент әрекет ететін жер.
Біздің мысалда орта — **лабиринт**.
3. Күй (State) — роботтың қазіргі жағдайы.
Мысалы: робот лабиринттің қай бөлігінде тұр, алдында қабырға бар ма, шығатын есік қай жақта?
4. Әрекет (Action) — роботтың жасайтын қадамы.
Мысалы:

- оңға бұрылу;
- солға бұрылу;
- алға жүру;
- артқа қайту.

5. Марапат (Reward) — әрекетті бағалау.
Мысалы:

- робот шығатын есікке жақындаса — **+10 ұпай**;
- есікті тапса — **+100 ұпай**;
- қабырғаға соғылса — **–10 ұпай**;
- қажетсіз бағытта жүрсе — **–1 ұпай**.

Алғашында робот қай бағыттың дұрыс екенін білмейді. Ол әртүрлі әрекет жасап көреді, қателеседі және ұпай жинайды. Бірнеше рет қайталағаннан кейін робот қай әрекеттер көбірек марапат әкелетінін түсінеді.
6. Саясат (Policy) — роботтың үйренген әрекет ету ережесі.
Мысалы, робот мынадай ережені үйренуі мүмкін:
«Алдымда қабырға болса — оңға бұрыламын. Жол ашық болса — алға жүремін».

Университеттен мысал

Жүйе студентке оқу материалдарын ұсынады:

- студент материалды оқып, тестті жақсы тапсырса — жүйе **оң марапат** алады;
- студент материалды ашпаса немесе тестті нашар тапсырса — жүйе **теріс баға** алады;
- уақыт өте келе жүйе әр студентке қандай материал ұсыну тиімді екенін үйренеді.

Қысқаша айтқанда:

Әрекет → нәтиже → марапат немесе жаза → тәжірибе → дұрыс шешімді үйрену.

Нығайтумен оқытуда жүйе дайын дұрыс жауаптардың толық жиынын пайдаланбайды.

Оның орнына агент белгілі ортада әрекет жасап, сол әрекеттердің нәтижесінде марапат немесе кері байланыс алады.

Негізгі ұғымдар:

Ұғым	**Мағынасы**
Агент	Шешім қабылдайтын жүйе
Орта	Агент әрекет ететін жағдай
Күй (State)	Ортаның ағымдағы сипаттамасы
Әрекет (Action)	Агент таңдайтын қадам
Марапат (Reward)	Әрекет нәтижесін бағалайтын сигнал
Саясат (Policy)	Агенттің әрекет таңдау стратегиясы

Мысалы, қоймада қозғалатын роботты қарастырайық.

Роботтың мақсаты — жүкті бастапқы нүктеден қажетті орынға жеткізу.

Ол әр қадамда қозғалу бағытын таңдайды.

Егер робот кедергіге соқтығысса, теріс марапат алады. Ал мақсатқа жетсе, оң марапат алады.

Көптеген әрекет нәтижесін талдау арқылы робот тиімді қозғалыс стратегиясын үйренуі мүмкін.

Нығайтумен оқытудың ерекшелігі — бір әрекеттің әсері бірден емес, бірнеше қадамнан кейін байқалуы мүмкін.

Сондықтан агент тек ағымдағы марапатты емес, болашақтағы жинақталған нәтижені де ескеруі керек.

Бұл тәсіл робототехникада, басқару жүйелерінде, ойын стратегияларында және ресурстарды динамикалық бөлу есептерінде қолданылады.

13. Машиналық оқыту процесінің негізгі кезеңдері

Машиналық оқыту тек алгоритмді іске қосумен шектелмейді.

Толық интеллектуалды жүйені жасау бірнеше өзара байланысты кезеңнен тұрады.

1. Мәселені және мақсатты анықтау
2. Деректерді жинау
3. Деректерді тазалау және дайындау
4. Белгілерді таңдау
5. Модельді таңдау және оқыту
6. Модельді бағалау
7. Жүйеге енгізу және бақылау

Бірінші кезең — мәселені анықтау.

Мысалы, университет жүйесі студенттің келесі семестрде академиялық қиындыққа тап болу тәуекелін болжауы керек.

Бұл жерде нақты не болжанатынын анықтау қажет.

Екінші кезең — деректерді жинау.

Бұрынғы семестрлердегі студенттердің оқу көрсеткіштері жиналады. Деректерді пайдалануда құпиялық пен заңды негіз ескеріледі.

Үшінші кезең — деректерді дайындау.

Қате, қайталанған немесе жетіспейтін мәліметтер өңделеді.

Төртінші кезең — белгілерді таңдау.

Модельге пайдалы болуы мүмкін көрсеткіштер анықталады.

Бесінші кезең — модельді оқыту.
Алгоритм таңдалып, модель параметрлері анықталады.
Алтыншы кезең — бағалау.
Модель бұрын көрмеген деректер арқылы тексеріледі.
Жетінші кезең — енгізу және бақылау.
Модель интеллектуалды жүйеге қосылады. Кейін оның сапасы мен деректердегі өзгерістер бақыланады.
Бұл кезеңдер әрдайым бір рет қана орындалмайды. Нәтижелер қанағаттандырмаса, алдыңғы кезеңдерге қайта оралу қажет болуы мүмкін.
14. Деректерді алдын ала өңдеу
Машиналық оқыту нәтижесінің сапасына деректердің сапасы тікелей әсер етеді.
Егер бастапқы деректерде қателер көп болса, модель қате заңдылықтарды үйренуі мүмкін.
Мысалы, студенттердің сабаққа қатысу деректерінде мынадай мәндер кездесуі мүмкін:

Студент	Қатысу пайызы
A	85
B	92
C	—
D	105
E	68

Мұнда екі мәселе бар.
Біріншіден, C студентінің қатысу пайызы белгісіз.
Екіншіден, D студентінің көрсеткіші 105%, ал қалыпты пайыздық көрсеткіш 0 мен 100 аралығында болуы керек.
Мұндай деректерді өңдемей модельге беру дұрыс емес.
Алдын ала өңдеудің негізгі әрекеттері:

- Жетіспейтін мәндерді өңдеу: мәнді қалпына келтіру, есептеу немесе негізді жағдайда жазбаны алып тастау.
- Қате мәндерді тексеру: мүмкін емес көрсеткіштерді анықтау.
- Қайталанған жазбаларды өңдеу: бір объектінің бірнеше рет есепке алынуын болдырмау.
- Категориялық белгілерді кодтау: мысалы, оқу формасын сандық форматқа түрлендіру.
- Масштабтау: белгілерді салыстырмалы масштабқа келтіру.

Бұл слайдта **стандарттау (Z-score)** түсіндіріледі. Стандарттау әртүрлі өлшемдегі көрсеткіштерді өзара дұрыс салыстыру үшін қажет.
Қарапайым мысал
Топтағы студенттердің тест бойынша:

- орташа бағасы — **70 балл**;
- стандартты ауытқуы — **10 балл**;
- Айдостың бағасы — **85 балл**.

Формула:
$$ z=\frac{x-\mu}{\sigma} $$
Есептейміз:
$$ z=\frac{85-70}{10}=1,5 $$
Бұл нәтиже Айдостың бағасы топтың орташа көрсеткішінен **1,5 стандартты ауытқуға жоғары** екенін білдіреді.
Қарапайым айтқанда: **Айдос орташа студенттен жақсы нәтиже көрсетті.**
Z-score нәтижесін қалай түсінеміз?

- **Z = 0** — нәтиже орташа мәнге тең;
- **Z > 0** — нәтиже орташадан жоғары;
- **Z < 0** — нәтиже орташадан төмен.

Мысалы:

- студент 70 балл алды: \(z=0\) — орташа нәтиже;
- студент 85 балл алды: \(z=1,5\) — орташадан жоғары;
- студент 50 балл алды: \[z=(50-70)/10=-2 \]— орташадан төмен.

Стандарттау не үшін қажет?

Студентті екі көрсеткіш бойынша бағалайық:

- емтихан бағасы — **0–100 балл**;
- платформадағы әрекеттері — **0–10 000 рет**.

10 000 саны 100-ден өте үлкен болғандықтан, алгоритм платформадағы әрекеттерді маңыздырақ деп қабылдауы мүмкін. Стандарттау екі көрсеткішті бірдей деңгейге келтіріп, әділ салыстыруға мүмкіндік береді.

Data Leakage деген не?

Бұл — модельге алдын ала білмеуі тиіс ақпараттың беріліп кетуі.

Мысалы, 100 студенттің:

- 80-і модельді оқыту үшін;
- 20-сы модельді тексеру үшін алынды.

Орташа мән мен стандартты ауытқуды **тек 80 студенттің мәліметтері бойынша** есептеу керек. Егер 100 студенттің барлығы қолданылса, модель тексерілетін 20 студенттің ақпаратын алдын ала біліп қояды.

Қысқаша айтқанда: **стандарттау көрсеткіштерді бірдей өлшемге келтіреді, ал тексеру деректерін есептеуге араластырмау нәтижені шынайы сақтайды.**

15. Оқыту, валидация және тестілеу деректері

Модельдің сапасын дұрыс бағалау үшін деректерді әртүрлі мақсатқа бөлу қажет.

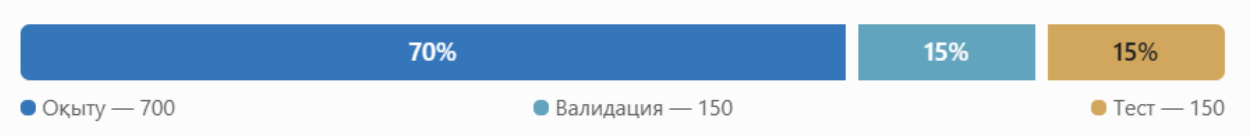
Әдетте **үш** жиын қолданылады:

Training Set — оқыту жиыны. Модель параметрлерін үйренуге арналған деректер.

Validation Set — валидация жиыны. Модель нұсқаларын, гиперпараметрлерді және басқа баптауларды таңдауға арналған деректер.

Test Set — тестілеу жиыны. Соңғы таңдалған модельдің бұрын қолданылмаған деректердегі сапасын бағалауға арналған жиын.

Мысалы, 1000 жазба шартты түрде былай бөлінуі мүмкін:



Бұл 70/15/15 қатынасы — мысал ғана. Нақты бөлу тәсілі деректер көлеміне және есептің ерекшелігіне байланысты таңдалады.

Мысалы, бір студенттің бірнеше семестрдегі жазбалары болса, деректерді кездейсоқ бөлу кезінде бір студент туралы ақпарат оқыту мен тест жиындарына қатар түсуі мүмкін.

Егер мақсат жаңа студенттерге жалпылау болса, студенттер бойынша бөлу тиімдірек болуы мүмкін.

Ал болашақтағы нәтижелерді болжағанда уақыт ретін сақтау маңызды.

Сондықтан деректерді дұрыс бөлу модельді бағалаудың негізгі шарттарының бірі болып саналады.

16. Модельді оқыту және қателік функциясы

Бұл слайдта модельдің **қалай болжам жасайтыны және оның қатесі қалай есептелетіні** көрсетілген.

Қарапайым мысал

Біз студенттердің емтихан бағасын болжайтын модель жасадық делік.

Нақты бағалар:

- Аружан — **60 балл**
- Данияр — **70 балл**
- Айбек — **80 балл**

Модельдің болжамы:

- Аружан — **62 балл**

- Данияр — **65 балл**
- Айбек — **83 балл**

Енді болжамның қаншалықты дұрыс екенін тексереміз.

1. Әр студент бойынша қатені табамыз

Қате — нақты баға мен болжамның айырмасы:

- Аружан: \(60-62=-2\)
- Данияр: \(70-65=5\)
- Айбек: \(80-83=-3\)

«Минус» модельдің артық болжағанын, ал «плюс» төмен болжағанын білдіреді.

2. Неліктен қатені квадраттаймыз?

Егер қателерді жай ғана қоссақ:

\[-2+5-3=0 \]

Нәтиже нөл болды. Бірақ модельде қателер бар. Оң және теріс сандар бірін-бірі жойып жіберді.

Сондықтан әр қатені квадраттаймыз:

- \((-2)^2=4\)
- \(5^2=25\)
- \((-3)^2=9\)

Енді барлық нәтиже оң санға айналды.

3. Орташа қатені есептейміз

\[MSE=\frac{4+25+9}{3}=12,67 \]

Демек, модельдің орташа квадраттық қатесі — **12,67**.

Бұл көрсеткішті қалай түсінеміз?

- MSE **аз болған сайын**, модельдің болжамы жақсы;
- MSE **үлкен болған сайын**, модель көп қателеседі;
- MSE **0 болса**, модель барлық мәнді дәл болжаған.

Модельді оқыту кезінде жүйе өзінің параметрлерін өзгертіп, осы MSE мәнін біртіндеп азайтуға тырысады.

Қарапайым айтқанда:

Модель жауап береді → нақты жауаппен салыстырамыз → қатені есептейміз → модель қатесін азайтып үйренеді.

Мысалы, студент 80 балл алды, ал модель 79 балл болжаса — қате аз. Егер модель 40 балл болжаса — қате үлкен. Сондықтан модель дұрыс болжамға жақындауға тырысады.

Машиналық оқыту кезінде модель бастапқы параметрлермен жұмысын бастайды.

Ол болжам жасайды, болжамды нақты нәтижемен салыстырады және қателікті азайтуға тырысады.

Модель болжамын:

$$\hat{y} = f(x;\theta)$$

деп белгілейік.

Мұнда:

- x — кіріс белгілері;
- θ — модель параметрлері;
- f — модельдің математикалық функциясы;
- $\hat{y}$ — болжам.

Нақты нәтиже болсын.

Модельдің қаншалықты қателескенін өлшеу үшін ***қателік немесе шығын функциясы (Loss Function)*** қолданылады.

Регрессияда кең таралған функциялардың бірі — орташа квадраттық қате:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Мысалы, нақты нәтижелер:

$$y = [60, 70, 80]$$

ал модель болжамдары:

$$\hat{y} = [62, 65, 83]$$

болсын.

Онда:

$$MSE = \frac{(60-62)^2 + (70-65)^2 + (80-83)^2}{3}$$

$$MSE = \frac{4 + 25 + 9}{3} \approx 12.67$$

MSE мәні неғұрлым төмен болса, берілген бағалау деректеріндегі квадраттық қателік соғұрлым аз.
Алайда тек оқыту деректеріндегі қатені азайту жеткіліксіз. Біз модельдің жаңа деректерде де жақсы жұмыс істеуін қалаймыз.

17. Градиенттік түсу әдісі — Gradient Descent

Көптеген машиналық оқыту модельдерінің параметрлерін оңтайландыру үшін градиенттік түсу әдісі қолданылады.
Градиенттік түсу (Gradient Descent) — қателік функциясының мәнін азайту мақсатында модель параметрлерін градиент бағытына қарсы өзгерту әдісі.
Бұл слайдта **градиенттік түсу әдісі** түсіндіріледі. Бұл — модельдің қатесін азайтып, дұрыс болжам жасауға үйрену тәсілі.

Қарапайым мысал: таудан түсу

Сіз таудың басында тұрсыз, бірақ айнала тұман болғандықтан төмендегі ең ойыс жерді көрмейсіз делік.
Сіз:

1. Аяғыңыз тұрған жердің еңісін тексересіз.
2. Төмен қарай бағытты анықтайсыз.
3. Сол бағытқа бір қадам жасайсыз.
4. Қайтадан еңісті тексересіз.
5. Ең төменгі жерге жеткенше осы әрекетті қайталайсыз.

Модель де дәл осылай жұмыс істейді:

- **таудың биіктігі** — модельдің қатесі;
- **төмен түсу бағыты** — градиент;
- **жасалатын қадамның көлемі** — оқыту жылдамдығы;
- **таудың ең төменгі нүктесі** — қатенің ең аз мәні.

Студент бағасын болжау мысалы

Студенттің нақты бағасы — **80 балл**, ал модель алғашқыда **50 балл** болжады.
Қате:
\[80-50=30 \]
Модель өзінің параметрлерін сәл өзгертеді:

- бірінші болжам — 50 балл;

- келесі болжам — 60 балл;
- одан кейін — 70 балл;
- кейін — 78 балл;
- соңында — 80 баллға жақындайды.

Әр қадамда қате азая береді. Осы процесс **градиенттік түсу** деп аталады.

Learning Rate деген не?

Learning Rate (\(\alpha\)) — модель әр қадамда параметрін қаншалықты өзгертетінін көрсетеді.

Оны таудан түсудегі қадамның ұзындығымен салыстыруға болады.

Егер қадам өте кішкентай болса:

- модель дұрыс бағытта қозғалады;
- бірақ өте баяу үйренеді;
- қажетті нәтижеге жету үшін көп уақыт кетеді.

Мысалы: **50 → 51 → 52 → 53...**

Егер қадам өте үлкен болса:

- модель қажетті нүктеден асып кетеді;
- біресе бір жаққа, біресе екінші жаққа секіреді;
- қате азаймай, тіпті өсуі мүмкін.

Мысалы: **50 → 95 → 40 → 110...**

Егер қадам дұрыс таңдалса:

- модель қатені біртіндеп азайтады;
- дұрыс нәтижеге тұрақты түрде жақындайды.

Мысалы: **50 → 65 → 74 → 78 → 80.**

Қысқаша айтқанда:

Градиенттік түсу — модельдің қатесін азайту үшін дұрыс бағытта қадам жасау әдісі.
Ал Learning Rate — сол қадамның үлкендігін анықтайды.

~~Жалпы формула:~~

$$\theta_{\text{new}} = \theta_{\text{old}} - \alpha \nabla J(\theta)$$

Мұнда:

- θ — модель параметрлері;
- $J(\theta)$ — оңтайландырылатын функция;
- $\nabla J(\theta)$ — градиент;
- α — оқыту жылдамдығы (Learning Rate).

~~Градиент функцияның қай бағытта өсетінін көрсетеді.~~
~~Ал біз қатені азайтқымыз келгендіктен, градиентке қарсы бағытта қозғаламыз.~~
~~Мысалы, төбеден төмен қарай еңіс бағытты іздеп түсіп келе жатқан адамды елестетейік.~~
~~Әр қадамда адам жердің еңісін анықтап, төмен бағытта қозғалады.~~
~~Градиенттік түсу осы идеяға ұқсас.~~
~~Бірақ оқыту жылдамдығын дұрыс таңдау маңызды.~~
~~Егер α өте кіші болса, модель баяу үйренуі мүмкін.~~
~~Егер α өте үлкен болса, алгоритм минимум маңынан өтіп кетуі немесе тұрақсыз жұмыс істеуі мүмкін.~~
~~Сондықтан оқыту процесінде қателік мәнінің өзгерісін бақылау қажет.~~

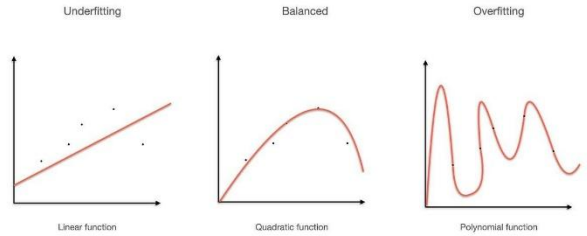
18. Артық үйрену және жеткіліксіз үйрену

Машиналық оқытудың маңызды мәселелерінің бірі — модельдің жаңа деректерге қаншалықты жақсы бейімделетіні.

Екі негізгі жағдайды қарастырамыз.

Жеткіліксіз үйрену (Underfitting) — модель деректердегі маңызды заңдылықтарды жеткілікті меңгере алмаған жағдай.

Мысалы, күрделі тәуелділікті өте қарапайым сызықтық модельмен сипаттауға тырыссақ, модель оқыту деректерінде де көп қателесуі мүмкін.
Артық үйрену (Overfitting) — модель оқыту деректерінің кездейсоқ ерекшеліктері мен шуын шамадан тыс меңгеріп, жаңа деректерде нашар нәтиже көрсететін жағдай.



Bias	High	Low	Low
Variance	Low	Low	High

community.deeplearning.ai
Мысалы, модель 100 студенттің деректеріне үйретілді делік.
Оқыту жиынында ол 99% дәлдік көрсетеді.
Бірақ бұрын көрмеген студенттердің деректерінде дәлдігі 68%-ға дейін төмендейді.
Бұл нәтиже модельдің жалпылау қабілеті жеткіліксіз екенін көрсетуі мүмкін.
Бұл слайд модельдің материалды **қалай меңгергенін** көрсетеді. Оны студенттің емтиханға дайындалуымен салыстыруға болады.
1. Жеткіліксіз үйрену — Underfitting
Студент тақырыпты толық түсінбей, тек үстірт оқыды делік. Сондықтан:
- бұрын көрген сұрақтарға да;
- жаңа сұрақтарға да дұрыс жауап бере алмайды.

Мысалы:
- жаттығу тестінде — **55%**;
- емтиханда — **50%**.

Модельде де осындай жағдай болады. Ол тым қарапайым болғандықтан, деректердегі негізгі заңдылықты үйрене алмайды.
Қарапайым айтқанда: модель жеткілікті түрде оқымаған.
2. Оңтайлы модель — Balanced Fit
Студент тақырыпты түсініп оқыды. Ол сұрақтардың жауабын жаттап қана қоймай, шешу тәсілін меңгерді.
Сондықтан:
- бұрын көрген сұрақтарды да;
- жаңа сұрақтарды да дұрыс орындайды.

Мысалы:
- жаттығу тестінде — **90%**;
- емтиханда — **87%**.

Нәтижелер бір-біріне жақын және екеуі де жоғары. Бұл — ең жақсы жағдай.
Қарапайым айтқанда: модель ережені дұрыс түсінді және оны жаңа деректерге қолдана алады.
3. Артық үйрену — Overfitting
Студент жаттығу тестіндегі сұрақтар мен жауаптарды толық жаттап алды, бірақ тақырыптың өзін түсінбеді делік.
Сондықтан:
- дәл сол сұрақтар қайталанса — бәріне жауап береді;
- жаңа сұрақтар келсе — қиналып қалады.

Мысалы:
- жаттығу тестінде — **99%**;
- жаңа емтихан сұрақтарында — **68%**.

Модель де оқыту деректерін жаттап алады, бірақ жаңа деректермен дұрыс жұмыс істей алмайды.
Қарапайым айтқанда: модель заңдылықты үйренудің орнына жауаптарды жаттап алған.
Үш жағдайдың айырмашылығы

Жағдай	Оқыту нәтижесі	Тест нәтижесі	Түсіндірмесі
Underfitting	55%	50%	Тақырыпты үйренбеген
Balanced Fit	90%	87%	Тақырыпты түсінген
Overfitting	99%	68%	Жауаптарды жаттап алған

Негізгі мақсат — оқыту кезінде де, жаңа тест деректерінде де жақсы нәтиже көрсететін **оңтайлы модель құру**.
Артық үйренуді азайту үшін бірнеше тәсіл қолданылады: модель күрделілігін шектеу, регуляризация, деректер көлемін арттыру, кросс-валидация және оқыту процесін бақылау.
Негізгі мақсат — оқыту деректерін жаттап алу емес, жаңа жағдайларға қолданылатын заңдылықтарды үйрену.
19. Модель сапасын бағалау
Машиналық оқыту моделін жасағаннан кейін оның нәтижесін міндетті түрде бағалау қажет. Бұл слайдта модельдің болжамын қалай дұрыс бағалау керектігі түсіндіріледі.
Қарапайым мысал
Бізде **100 студент** бар делік:

- 95 студенттің тәуекелі жоқ;
- 5 студенттің оқудан шығып қалу тәуекелі бар.

Модель барлық студентке:
«Тәуекел жоқ»
деп жауап берді.
Сонда модель 95 студент бойынша дұрыс жауап берді:
\[Accuracy=\frac{95}{100}=95\% \]
Сырттай қарағанда **95% — өте жақсы нәтиже** сияқты. Бірақ модель тәуекел тобындағы 5 студенттің **біреуін де анықтамады**.
Сондықтан тек Accuracy көрсеткішіне сенуге болмайды.
Confusion Matrix деген не?
Бұл — модельдің қандай жағдайда дұрыс, қандай жағдайда қате жауап бергенін көрсететін кесте.
Студенттің тәуекелін анықтау мысалымен түсіндірейік.
1. TP — дұрыс анықталған тәуекел
Студентте шынымен тәуекел бар. Модель де:
«Тәуекел бар»
деп дұрыс анықтады.
Мысалы, студент сабаққа келмейді, тапсырма тапсырмайды. Жүйе оны тәуекел тобына дұрыс енгізді.
2. FP — жалған дабыл
Студентте тәуекел жоқ. Бірақ модель:
«Тәуекел бар»
деп қате айтты.
Мысалы, студент жақсы оқиды, бірақ бір апта сабаққа келмеді. Жүйе оны қате түрде тәуекел тобына енгізді.
3. FN — байқамай қалу
Студентте шынымен тәуекел бар. Бірақ модель:
«Тәуекел жоқ»
деп қате айтты.
Бұл ең қауіпті жағдай. Себебі студентке уақытында көмек көрсетілмей қалады.

4. TN — дұрыс анықталған қауіпсіз жағдай
Студентте тәуекел жоқ. Модель де:
«Тәуекел жоқ»
деп дұрыс анықтады.
Қарапайым кесте

Нақты жағдай	Модель болжамы	Нәтиже
Тәуекел бар	Тәуекел бар	TP — дұрыс
Тәуекел жоқ	Тәуекел бар	FP — жалған дабыл
Тәуекел бар	Тәуекел жоқ	FN — байқамай қалды
Тәуекел жоқ	Тәуекел жоқ	TN — дұрыс

Қысқаша айтқанда:
Accuracy модельдің жалпы қанша дұрыс жауап бергенін көрсетеді. Ал Confusion Matrix оның нақты қандай қателер жібергенін көрсетеді.
Студенттердің тәуекелін анықтауда әсіресе **FN көрсеткіші маңызды**, себебі бұл — жүйе тәуекелі бар студентті байқамай қалды деген сөз.
Классификация есептерінде ең қарапайым көрсеткіштердің бірі — Accuracy.

$$Accuracy = \frac{\text{Дұрыс болжамдар саны}}{\text{Барлық болжамдар саны}}$$

Мысалы, модель 100 студенттің 85-ін дұрыс жіктесе:

$$Accuracy = \frac{85}{100} = 0.85 = 85\%$$

Бірақ Accuracy барлық жағдайда жеткілікті емес.
Мысалы, 100 студенттің 95-інде академиялық тәуекел жоқ, ал 5 студентте бар делік.
Егер модель барлық студентті «тәуекел жоқ» деп белгілесе, Accuracy 95% болады.
Алайда жүйе тәуекелі бар бес студенттің біреуін де анықтай алмайды.
Сондықтан қосымша метрикалар қажет.
Confusion Matrix — қателер матрицасы

	Нақты: тәуекел бар	**Нақты: тәуекел жоқ**
Болжам: тәуекел бар	TP	FP
Болжам: тәуекел жоқ	FN	TN

Мұндағы:
- TP (True Positive) — тәуекелі бар студент дұрыс анықталды.
- FP (False Positive) — тәуекелі жоқ студент қате түрде тәуекел тобына жатқызылды.
- FN (False Negative) — тәуекелі бар студент анықталмай қалды.
- TN (True Negative) — тәуекелі жоқ студент дұрыс анықталды.

Precision және Recall
Бұл слайдта модельдің сапасын бағалайтын үш көрсеткіш берілген: **Precision, Recall және F1-score**.
Мысал ретінде студенттердің тәуекелін анықтайтын жүйені алайық.
Жүйе **24 студентте тәуекел бар** деп белгіледі:
- оның 18-інде шынымен тәуекел бар — **TP**;
- 6 студентте тәуекел жоқ, жүйе қате белгіледі — **FP**.

Сонымен қатар тәуекелі бар тағы 2 студентті жүйе байқамай қалды — **FN**.
1. Precision — дәлдік
Precision мына сұраққа жауап береді:
«Жүйе тәуекел бар деп белгілеген студенттердің қаншасында шынымен тәуекел бар?»
Жүйе 24 студентті белгіледі, оның 18-і дұрыс:
\[Precision=\frac{18}{18+6}=0,75=75\% \]

Яғни жүйенің әр 100 ескертуінің шамамен **75-і дұрыс**, ал 25-і жалған дабыл.

2. Recall — толықтық

Recall мына сұраққа жауап береді:
«Нақты тәуекелі бар студенттердің қаншасын жүйе тауып алды?»
Нақты тәуекелі бар студенттер саны:
\[18+2=20 \]
Жүйе олардың 18-ін тапты:
\[Recall=\frac{18}{18+2}=0,90=90\% \]
Яғни жүйе тәуекелі бар студенттердің **90%-ын анықтады**, ал 10%-ын байқамай қалды.

3. F1-score — ортақ баға

F1-score Precision мен Recall көрсеткіштерін біріктіреді:
\[F1=\frac{2 \times 0,75 \times 0,90}{0,75+0,90}=0,818 \]
Демек:
\[F1=81,8\% \]
Бұл модельдің жалған дабылдары мен байқамай қалған жағдайларын бірге есептегендегі жалпы сапасы.

Айырмашылығы

- **Precision** — жүйе белгі бергенде, оның қаншалықты дұрыс екенін көрсетеді.
- **Recall** — жүйе барлық қауіпті жағдайлардың қаншасын тапқанын көрсетеді.
- **F1-score** — осы екеуінің арасындағы тепе-теңдікті көрсетеді.

Студенттердің тәуекелін ерте анықтау жүйесінде **Recall маңыздырақ**, өйткені тәуекелі бар студентті байқамай қалу оған уақытында көмек көрсетілмеуіне әкелуі мүмкін. Бірақ жалған ескертулер тым көп болмауы үшін Precision көрсеткішін де бақылау қажет.

Precision — модель тәуекелі бар деп белгілеген студенттердің қаншасы шынымен сол топқа жататынын көрсетеді.

$$Precision = \frac{TP}{TP + FP}$$

Recall — нақты тәуекелі бар студенттердің қаншасын модель анықтағанын көрсетеді.

$$Recall = \frac{TP}{TP + FN}$$

Мысалы:

$$TP = 18, \quad FP = 6, \quad FN = 2, \quad TN = 74$$

Онда:

$$Precision = \frac{18}{18 + 6} = 0.75$$

$$Recall = \frac{18}{18 + 2} = 0.90$$

Демек, модель анықтаған тәуекел жағдайларының 75%-ы дұрыс, ал нақты тәуекел жағдайларының 90%-ы табылған.
Бұл екі көрсеткіш әртүрлі сұраққа жауап береді.
Precision — «Ескерту жасаған кезде жүйе қаншалықты жиі дұрыс?»
Recall — «Қауіпті жағдайлардың қаншасын жүйе байқайды?»

F1-score

Precision мен Recall арасындағы тепе-теңдікті бағалау үшін F1-score қолданылады.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

~~Берілген мысалда:~~

$$F1 = 2 \cdot \frac{0.75 \cdot 0.90}{0.75 + 0.90} \approx 0.818$$

Яғни F1-score шамамен 81.8%.

Бұл көрсеткіштерді есептеу үшін тиісті бөлім нөлге тең болмауы керек; практикалық кітапханаларда мұндай жағдайларды өңдеу параметрлері бар.

24. Практикалық талдау

Енді шағын тапсырманы бірге қарастырайық.

Университеттің интеллектуалды жүйесі студенттердің академиялық тәуекелін болжауы керек.

Үш студент туралы келесі мәлімет берілген:

Көрсеткіш	**Айдана**	**Нұрсұлтан**	**Данияр**
GPA	3.35	2.45	3.05
Сабаққа қатысу	64%	87%	76%
Кеш тапсырған жұмыс саны	6	1	3
Соңғы бақылау динамикасы	Төмендеген	Жақсарған	Тұрақты

Осы деректерді пайдаланып, келесі сұрақтарға жауап берейік.

1. Қай көрсеткіштер машиналық оқыту моделінің кіріс белгілері бола алады?
2. Егер мақсат студентті «жоғары тәуекел» немесе «төмен тәуекел» деп жіктеу болса, бұл классификация ма, әлде регрессия ма?
3. Неліктен GPA көрсеткіші жоғары студенттің де академиялық тәуекелі болуы мүмкін?
4. Модельді оқыту үшін осы кестедегі үш студент жеткілікті ме?
5. Егер жүйе тәуекел деңгейін болжаған болса, нақты нәтижені қалай тексеруге болады?

Бұл тапсырманың мақсаты — студенттерге машиналық оқытуда бір ғана көрсеткішке сүйену жеткіліксіз болуы мүмкін екенін көрсету.

Сонымен қатар үш студенттің мәліметі бойынша сенімді модель жасауға болмайтынын түсіну қажет.

Модельге тарихи деректер, нақты анықталған мақсатты айнымалы және тәуелсіз бағалау қажет.

Бұл тапсырмада үш студенттің мәліметіне қарап, олардың оқу тәуекелін бағалау керек.

Тәуекелді анықтау үшін тек GPA-ға емес, барлық көрсеткіштерге бірге қараймыз:

- GPA;
- сабаққа қатысуы;
- жұмысты кеш тапсыру саны;
- оқу нәтижесінің өзгеруі.

1. Айдана

- GPA — **3,35**, яғни жақсы;
- сабаққа қатысуы — **64%**, төмен;
- 6 жұмысты кеш тапсырған;
- көрсеткіштері төмендеп келе жатыр.

GPA жоғары болғанымен, сабаққа аз қатысады, тапсырмаларды жиі кеш тапсырады және оқу динамикасы төмендеген. Сондықтан Айданада тәуекел бар.

Жауабы: Айдана — жоғары тәуекел тобы.

2. Нұрсұлтан

- GPA — **2,45**, төмен;

- сабаққа қатысуы — **87%**, жақсы;
- тек 1 жұмысты кеш тапсырған;
- оқу нәтижесі жақсарып келеді.

Қазіргі GPA көрсеткіші төмен, бірақ студент сабаққа тұрақты қатысады, тапсырмаларын уақытында орындайды және нәтижесі жақсарып келеді. Сондықтан оның жағдайы бақылауды қажет етеді, бірақ тәуекелі Айданадан төмен.

Жауабы: Нұрсұлтан — орташа тәуекел тобы.

3. Данияр

- GPA — **3,05**, орташа;
- сабаққа қатысуы — **76%**;
- 3 жұмысты кеш тапсырған;
- нәтижесі тұрақты.

Даниярдың көрсеткіштері өте жақсы да, өте нашар да емес. Сабаққа қатысуы мен кеш тапсырған жұмыстары бойынша аздаған қауіп бар.

Жауабы: Данияр — орташа немесе төмен тәуекел тобы.

Слайдтағы сұрақтардың жауаптары

1. GPA жоғары Айданада неге тәуекел болуы мүмкін?

Себебі GPA — жалғыз көрсеткіш емес. Айдананың сабақты босатуы көп, 6 жұмысты кеш тапсырған және нәтижесі төмендеп келеді. Бұл болашақта үлгерімінің нашарлау ықтималдығын көрсетеді.

2. Үш студенттің дерегі модельді оқытуға жеткілікті ме?

Жоқ, жеткіліксіз. Үш адамның дерегі бойынша модель жалпы заңдылықты үйрене алмайды. Әртүрлі курс, топ және үлгерім деңгейіндегі жүздеген немесе мыңдаған студенттің мәліметі қажет.

3. Бұл классификация ма, әлде регрессия ма?

Егер нәтиже:

- жоғары тәуекел;
- орташа тәуекел;
- төмен тәуекел

түрінде топтарға бөлінсе, бұл — **классификация**.

Егер модель тәуекелді нақты санмен, мысалы **78%** деп болжаса, бұл сандық нәтиже. Бірақ кейін оны «жоғары/төмен тәуекел» класына айналдырсақ, соңғы міндет классификация болады.

Қорытынды жауап

Студент	Тәуекел деңгейі	Негізгі себеп
Айдана	Жоғары	Қатысуы төмен, кешіктіруі көп, динамикасы төмендеген
Нұрсұлтан	Орташа	GPA төмен, бірақ көрсеткіштері жақсарып келеді
Данияр	Төмен–орташа	Нәтижесі тұрақты, бірақ қатысуы мен кешіктіруі мінсіз емес

Негізгі ой: студенттің тәуекелін бір ғана GPA бойынша емес, барлық көрсеткіштерді бірге талдау арқылы бағалау қажет.

25. Дәрісті қорытындылау

Бүгін біз интеллектуалды ақпараттық жүйелердегі машиналық оқытудың негіздерін қарастырдық.

Алдыңғы дәрістерде интеллектуалды жүйелердің белгілі фактілер мен ережелерді пайдаланып қорытынды шығаратынын зерттедік.

Ал машиналық оқыту бізге жаңа мүмкіндікті көрсетеді: жүйе деректерден заңдылықтарды үйреніп, бұрын кездеспеген объектілер туралы болжам жасай алады.

Машиналық оқытудың негізгі үш бағытымен таныстық: бақыланатын оқыту, бақыланбайтын оқыту және нығайтумен оқыту.

Классификация мен регрессияның айырмашылығын анықтадық, кластерлеу принципін түсіндірдік, модельді оқыту кезеңдерін және нәтижелерді бағалау метрикаларын қарастырдық.

Ең маңызды қорытынды — машиналық оқыту моделі дайын ақиқат көзі емес. Оның нәтижесі деректердің сапасына, алгоритмнің мүмкіндігіне және бағалау тәсіліне байланысты.

Сондықтан интеллектуалды ақпараттық жүйелерді жобалағанда логикалық ережелерді, машиналық оқытуды және адамның сараптамалық бақылауын орынды біріктіру маңызды.

20. Python тілінде машиналық оқыту моделін құру

Енді машиналық оқытудың қарапайым программалық мысалын қарастырайық.

Python тілінде машиналық оқыту үшін кең таралған кітапханалардың бірі — scikit-learn.

Бұл кітапхана классификация, регрессия, кластерлеу, деректерді өңдеу және модельдерді бағалау құралдарын ұсынады.

Мысал ретінде гүлдердің өлшемдері бойынша түрін анықтайтын классификация моделін құрамыз.

Ол үшін Iris деректер жиынын қолданамыз.

```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# 1. Деректерді жүктеу
iris = load_iris()

X = iris.data
y = iris.target

# 2. Оқыту және тест жиындарына бөлу
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

# 3. Модельді құру
model = DecisionTreeClassifier(
```

```
    max_depth=3,
    random_state=42
)

# 4. Модельді оқыту
model.fit(X_train, y_train)

# 5. Болжам жасау
y_pred = model.predict(X_test)

# 6. Нәтижені бағалау
accuracy = accuracy_score(y_test, y_pred)

print("Accuracy:", accuracy)

# 7. Жаңа объектіні жіктеу
new_flower = [[5.1, 3.5, 1.4, 0.2]]

prediction = model.predict(new_flower)

print("Гүл түрі:", iris.target_names[prediction[0]])
```

Бұл программаның жұмысын кезең-кезеңімен түсіндірейік.
Бірінші кезеңде load_iris() функциясы дайын деректер жиынын жүктейді.
X айнымалысында гүлдердің өлшемдері сақталады, ал y айнымалысында олардың түрлері көрсетіледі.
Екінші кезеңде деректер оқыту және тестілеу жиындарына бөлінеді.
Үшінші кезеңде шешім ағашы моделі құрылады.
Шешім ағашы (Decision Tree) — белгілерге қатысты шарттарды біртіндеп тексеру арқылы болжам жасайтын модель.
Төртінші кезеңде:

```
Python

model.fit(X_train, y_train)
```

командасы модельді оқытады.
Бесінші кезеңде:

```
Python

model.predict(X_test)
```

командасы тест деректеріне болжам жасайды.
Алтыншы кезеңде болжамдар нақты нәтижелермен салыстырылады.
Соңғы кезеңде жаңа гүлдің төрт өлшемі енгізіліп, оның түрі анықталады.
Бұл мысалда модельдің оқытылуы мен болжам жасауы бір-бірінен бөлек орындалады.
Осы ерекшелік машиналық оқытудың негізгі принциптерінің бірі болып саналады.

21. Машиналық оқыту және ережелерге негізделген жүйелер

Енді бүгінгі тақырыпты алдыңғы дәрістермен байланыстырайық.
Бұрын біз білім базасы, өнімдік ережелер, фреймдер және логикалық шығару механизмдерін қарастырдық.

Машиналық оқыту осы тәсілдерден несімен ерекшеленеді?

Критерий	**Ережелерге негізделген жүйе**	**Машиналық оқыту**

Білім көзі	Сарапшылар, ережелер, құжаттар	Деректер және оқыту алгоритмі
Шешім қабылдау	Формальды ережелерді қолдану	Үйренілген заңдылықтарды қолдану
Жаңарту	Ережелерді өзгерту	Қайта оқыту немесе параметрлерді жаңарту
Түсіндіру	Ережелер тізбегін көрсетуге болады	Түсіндіру мүмкіндігі модельге байланысты
Артықшылығы	Нақты шарттарды басқару ыңғайлы	Күрделі статистикалық байланыстарды анықтай алады
Шектеуі	Көп ережені басқару қиын	Деректерге тәуелді, қате болжам жасауы мүмкін

Мысалы,
университетте интеллектуалды академиялық кеңесші жүйе құрайық.
Машиналық оқыту моделі студенттің келесі семестрдегі академиялық тәуекелін бағалайды.
Ал ережелерге негізделген модуль университеттің ресми талаптарын тексереді.
Модель жоғары тәуекел анықтаса, жүйе эдвайзерге қосымша тексеру туралы ұсыныс жібере алады.
Бірақ студентке қатысты ресми шешімдер тек модельдің болжамына сүйеніп автоматты түрде қабылданбауы тиіс.
Осылайша екі тәсілді біріктіруге болады:
Деректер → Машиналық оқыту моделі → Тәуекелді бағалау → Логикалық ережелер → Ұсыныс және түсіндіру.
Мұндай тәсіл интеллектуалды ақпараттық жүйенің икемділігін арттыруы мүмкін.

22. Машиналық оқытудың артықшылықтары мен шектеулері

Машиналық оқытудың негізгі артықшылықтары — үлкен деректер жиындарын талдау, күрделі байланыстарды анықтау, жаңа объектілерге болжам жасау және белгілі процестерді автоматтандыру мүмкіндігі.
Мысалы, өндірістік интеллектуалды жүйе сенсорлардан алынған мәліметтер арқылы жабдықтың істен шығу тәуекелін бағалай алады.
Бірақ машиналық оқытудың шектеулері де бар.
Бірінші шектеу — деректер сапасына тәуелділік. Қате немесе жеткіліксіз деректер модель сапасын төмендетуі мүмкін.
Екінші шектеу — артық үйрену. Модель оқыту деректерінде жақсы жұмыс істеп, жаңа жағдайларда қателесуі мүмкін.
Үшінші шектеу — түсіндіру қиындығы. Кейбір күрделі модельдердің нақты шешімін түсіндіру оңай емес.
Төртінші шектеу — әділеттілік мәселесі. Егер оқыту деректерінде белгілі бір топтарға қатысты жүйелі бұрмалану болса, модель сол бұрмалануды қайталауы мүмкін.
Бесінші шектеу — деректердің өзгеруі. Уақыт өте пайдаланушылардың мінез-құлқы, оқу бағдарламалары немесе жұмыс жағдайлары өзгеруі мүмкін. Мұндайда бұрын оқытылған модельдің сапасы төмендеуі ықтимал.
Сондықтан интеллектуалды жүйеде машиналық оқыту моделін енгізу оның жұмысын үнемі бақылауды да талап етеді.

23. Машиналық оқытудың интеллектуалды ақпараттық жүйелерде қолданылуы

Машиналық оқыту әртүрлі пәндік салаларда қолданылады.

Сала	**Қолданылу мысалы**
Білім беру	Оқу үлгерімінің өзгерісін болжау, оқу материалдарын ұсыну
Денсаулық сақтау	Медициналық кескіндерді талдауға көмектесу
Қаржы	Күмәнді операцияларды анықтау
Өндіріс	Жабдық ақауларын алдын ала болжау
Ауыл шаруашылығы	Өсімдік ауруларының белгілерін кескіндерден анықтау

Электрондық коммерция	Тауарлар мен қызметтерді ұсыну
Киберқауіпсіздік	Желідегі аномалияларды анықтау

Мысалы,

университеттің интеллектуалды оқыту платформасы студенттің орындаған тапсырмаларын талдап, келесі оқу материалын ұсына алады.

Алайда мұндай жүйенің ұсыныстары педагогикалық мақсатқа сәйкес келуі тиіс.

Егер модель студенттің белгілі тақырып бойынша қиналып жүргенін анықтаса, жүйе оған қосымша түсіндірме немесе жаттығу ұсына алады.

Бұл жерде машиналық оқыту оқытушының жұмысын алмастырмайды, керісінше оқу процесін қолдауға арналған қосымша құрал ретінде пайдаланылады.

Бақылау сұрақтары

1. Машиналық оқыту дегеніміз не және ол дәстүрлі программалаудан несімен ерекшеленеді?
2. Машиналық оқытудың негізгі үш түрін атаңыз.
3. Классификация мен регрессияның айырмашылығы қандай?
4. K-means алгоритмінің жұмыс принципін түсіндіріңіз.
5. Модельді оқытуда белгілер мен мақсатты айнымалының рөлі қандай?
6. Оқыту, валидация және тестілеу жиындары не үшін қажет?
7. Артық үйрену және жеткіліксіз үйрену дегеніміз не?
8. Precision мен Recall көрсеткіштерінің айырмашылығы қандай?
9. Машиналық оқыту моделін ережелерге негізделген интеллектуалды жүйемен қалай біріктіруге болады?
10. Машиналық оқыту нәтижесін нақты шешім қабылдауда қолданғанда қандай шектеулерді ескеру қажет?

Пайдаланылған әдебиеттер

1. Mitchell, T. M. Machine Learning. McGraw-Hill, 1997.
2. Bishop, C. M. Pattern Recognition and Machine Learning. Springer, 2006.
3. Géron, A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 3rd ed., O'Reilly Media, 2022.
4. Russell, S., Norvig, P. Artificial Intelligence: A Modern Approach. 4th ed., Pearson, 2021.
5. James, G., Witten, D., Hastie, T., Tibshirani, R., Taylor, J. An Introduction to Statistical Learning: with Applications in Python. Springer, 2023.