МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ

Кафедра Прикладної математики

Реферат на тему:
«Машинне навчання. Machine learning»
з дисципліни:
«Основи наукових досліджень, організація науки та авторське право»

Виконав:
ст. гр. СТКСм-18-1
Гой Д. К.

Перевірив:
проф. каф. ПМ
Тевяшев А. Д.

Харків - 2018 рік

ЗМІСТ

# ВСТУП

Всю свою історію людина шукає пізнання себе. Так, в стародавньому світі люди намагались зрозуміти з чого створений людський мозок, як він функціонує, з чого і як створюються думки й сигнали до м'язів, органів. В той момент коли наука змогла відповісти на ці питання настільки, що відповідь задовільнила більшість прогресивних мислителів, постало наступне завдання - створити щось, що буде функціонувати як наш мозок.

Перші електронно-обчислювальні машини зробили великий вклад в розвиток всіх сфер діяльності людини, але найбільший вклад ЕОМ можна вважати в створення «розумних» машин. Перші логічні схеми дали поштовх у створенні перших (примітивних) «розумних» механізмів. Такі механізми виконували цільову задачу використовуючи вхідні дані які могли змінюватись в залежності від потреб.

З розвитком інформаційних технологій людство ще більше наблизилось до розв'язання загадок мозку й спробам повторити діяльність мозку в ЕОМ. На даний момент ці галузі знань, «Штучний інтелект» і суміжні, зараз переживають нову епоху перероджения. Це зумовлено високим рівнем розвитку «заліза» (англ. hardware; син. апаратна конфігурація) яке виконує математичне обчислення алгоритмів.

# 1. ПРО МАШИННЕ НАВЧАННЯ

Машинне навчання (англ. Machine Learning; скор. ML) - це область комп'ютерних наук(англ. Computer Science; скор. CS), яка використовує статистичні методики, щоб дати комп'ютерним програмам можливість навчитися на минулому досвіді і вдосконалювати виконання конкретних завдань.

## 1.1. Загальне поняття про машинне навчання

Наука про дані (англ. Data Science), машинне навчання і штучний інтелект (англ. Artificial Intelligence; скор. AI) є одними з провідних тем у сучасному світі інформаційних технологій. Інтелектуальний аналіз даних та бассівський аналіз - наразі є трендом, і це додає попиту на машинне навчання.

Під поняттям машинного навчання також мається на увазі дисципліна, яка займається програмуванням систем так, щоб вони автоматично вивчали та вдосконалювалися з досвідом. Навчання передбачає визнання та розуміння вхідних даних та прийняття обґрунтованих рішень на основі наданих даних. Дуже важко розглянути всі рішення на основі всіх можливих внесків самостійно людиною, тому для вирішення цієї проблеми розробляються алгоритми, які будують знання з конкретних даних і минулого (пройденого) досвіду, застосовуючи принципи статистичної науки, ймовірності, логіки, математичної оптимізації, підкріплення і теорії управління. Такий підхід до навчання вважається індуктивним, на відміну від дедуктивного -

використання відомих моделей з підстановкою певних даних й виведення зрештою, якщо потрібно, з загального й абстрактного знання часткових та конкретних знань.

Процеси, пов'язані з машинним навчанням:

- аналіз даних (англ. Data Analysis);
- збір даних (англ. Data Mining);
- прогнозуючого моделювання (англ. Predictive Moddeling).

Багато людей, насправді, давно знайомі з машинним навчанням за покупками в Інтернеті та рекламою покупок. Тому, що двигуни з рекомендаціями використовують машинне навчання, щоб персоналізувати об’яви в режимі реального часу. Крім персонального маркетингу, інші поширені випадки зустрічі користувача інтернету з машинним навчанням можуть бути: виявлення шахрайства, фільтрація спаму, виявлення загроз мережевої безпеки, інтелектуальне обслуговування та створення каналів новин.

Цілі машинного навчання можно сформувати так - це не створення автоматизованого дублювання розумної поведінки, а використання потужності комп'ютерів для доповнення та доповнення людського інтелекту. Наприклад, програми машинного навчання можуть сканувати та обробляти величезні бази даних, що виявляють шаблони, які виходять за рамки людського сприйняття.

## 1.2 Принципи машинного навчання

Принципи машинного навчання мають більш-менш стійку класифікацію яка відображає рівень занурення розробника системи навчання в сам процес, й бувають наступними:

- з учителем (англ. suprevised);
- без вчителя(англ. unsupervised);
- навчання під наглядом (англ. semi-supervised);
- за підкріпленням (англ. reinforcement).

Найбільш використовувані вважаються з учителем (далі - керовані) та без вчителя (далі - спонтанні) принципи.

1.2.1 Керовані алгоритми. Ці алгоритми вимагають, щоб вчений або аналітик даних мали навички машинного навчання для забезпечення як вхідного, так і бажаного виходу, на додаток до надання зворотного зв'язку про точність прогнозів під час навчання алгоритму. Дані вчені визначають, які змінні або ознаки модель повинна аналізувати і використовувати для розробки прогнозів.

Контрольоване навчання зазвичай використовується в реальних додатках, таких як розпізнавання обличчя і мови, рекомендації щодо продуктів або фільмів, а також прогнозування продажів. Контрольоване навчання може бути розділене на два типи - регресія і класифікація.

Регресія тренується на те щоб прогнозувати безперервні результати в майбутньому, наприклад, прогнозувати ціни на нерухомість.

Класифікація намагається знайти відповідність, таку як аналіз позитивних або негативних настроїв, чоловіків і жінок, доброякісних і

злоякісних пухлин, безпечних і незахищених кредитів тощо. Може бути будь-яка кількість класів відношення до яких треба передбачити.

У процесі навчання під наглядом дані для навчання надходять з описом, мітками, цілями або бажаними результатами, і метою є знайти загальне правило, яке відображає входи на виходи. Такі навчальні дані називаються маркованими даними (англ. labeled data). Вивчене правило потім використовується для позначення нових даних з невідомими виходами.

Контрольоване навчання передбачає створення моделі машинного навчання, яка базується на маркованих зразках. Наприклад, якщо ми побудуємо систему для оцінки ціни ділянки або будинку на основі різних функцій, таких як розмір, місце розташування тощо, спочатку потрібно створити базу даних і позначити її. Нам потрібно навчити алгоритму, які особливості відповідають яким цінам. Виходячи з цих даних, алгоритм навчиться обчислювати ціну нерухомості за допомогою значень вхідних ознак.

Контрольоване навчання стосується вивчення функції з наявних навчальних даних. Тут алгоритм навчання аналізує навчальні дані і виробляє похідну функцію, яка може бути використана для відображення нових прикладів. Існує багато алгоритмів навчання під наглядом, таких як логістична регресія, нейронні мережі, супровідні векторні машини (SVM) і класичні класифікатори Баєсв (англ. Bayes).

Звичайні приклади навчання під наглядом включають класифікацію електронної пошти в категоріях спаму та не спаму, позначення веб-сторінок на основі їхнього вмісту та розпізнавання голосу. В подібні фільтри

вводяться ключові фрази або певні маркери, які мають певний ваговий коефіцієнт.

1.2.2 Напівавтоматичний (англ. semi-supervised) принцип. Цей принцип полягає в тому, що комп'ютерові дають лише неповний тренувальний сигнал: тренувальний набір, в якому відсутні деякі (часто численні) цільові виходи. Таким чином алгоритм не знає цієї частини правильних відповідей в процесі навчання.

Він використовує велику кількість немічених даних для навчання і невелику кількість позначених даних для тестування. Навчання з напівконтролем застосовується в тих випадках, коли є дорого придбати повністю маркований набір даних, а більш практично - позначити невелику підмножину. Наприклад, це часто вимагає кваліфікованих фахівців для позначення певних зображень з дистанційного зондування, а також великої кількості польових експериментів для виявлення нафти в певному місці, а отримання немечених даних є відносно легким. Це може суттєво вплинути на результат навчання та якість виконання завдання системою.

1.2.3 Навчання з підкріпленням (англ. reinforcement learning). Даний тип алгоритмів полягає в наданні тренувальних даних (у вигляді винагород та покарань) лише як зворотний зв'язок на дії програми в динамічному (частіше - ізольованому) середовищі, як при керуванні автомобілем, або грі в гру з опонентом. Система оцінює свою роботу на основі зворотного зв'язку і реагує відповідно. Найбільш відомі екземпляри включають самостійне водіння автомобілів і алгоритм майстра шахматів AlphaGo.

1.2.4 Неконтрольовані алгоритми. Цей принцип  не потребує додаткової підготовки бажаних даних результатів. Замість цього використовують

ітеративний підхід, який також називається глибоким навчанням (англ. Deep Learning), для перегляду даних і отримання висновків.

Алгоритми навчання без нагляду використовуються для більш складних завдань обробки, ніж контрольовані навчальні системи, включаючи розпізнавання зображень, оцифрування звуку «мова-в-текст» та створення природного мовлення (англ. Natural Language Generation або NLG). Використовується для виявлення аномалій, викидів, таких як шахрайство або несправне обладнання, або для групування клієнтів з подібною поведінкою для рекламної кампанії. Це протилежне навчанню під наглядом. Тут немає позначених даних.

При вивченні даних містяться лише деякі ознаки без будь-якого опису або мітки, це залежить від програміста або алгоритму, щоб знайти структуру базових даних, виявити приховані шаблони або визначити, як описати дані. Такі навчальні дані називаються немеченими даними (англ. unlabeled data).

Припустимо, що у нас є ряд точок даних, і ми хочемо класифікувати їх на кілька груп. Ми не можемо точно знати, якими будуть критерії класифікації. Отже, алгоритм навчання без нагляду намагається оптимально класифікувати даний набір даних у певну кількість груп.

Алгоритми навчання без нагляду є надзвичайно потужним інструментом для аналізу даних і для виявлення закономірностей і тенденцій. Вони найчастіше використовуються для кластеризації подібного введення в логічні групи. Алгоритми навчання без нагляду включають K-середні, «випадкові ліси» (англ. Random Forests), ієрархічну кластеризацію тощо.

Ці системи працюють шляхом комбінації мільйонів прикладів навчальних даних і автоматичного виявлення найтонших кореляцій між

багатьма змінними. Після навчання алгоритм може використовувати свій банк асоціацій для інтерпретації нових даних. Ці алгоритми стали можливими лише у епоху великих даних, оскільки вони потребують великих обсягів навчальних даних.

### 1.3. Застосування машинного навчання

Одне з найбільш об'єктивних та логічних застосування машинного навчання є - використання статистичних прийомів для надання комп'ютерам здатності «навчатися» (тобто, поступово покращувати продуктивність у певній задачі) з даних, без того, щоби бути програмованими явно. Але також існує більш розширені задачі та сфери застосування подібних систем.

- Класифікації (англ. classification). Частіше за все, входи поділяються на два або більше класів, і система-учень мусить породити модель, яка відносить небачені входи до одного з цих класів. Можливо також відношення до кількох класів одночасно в більш складній моделі. Таку задачу найчастіше розв'язують керованим чином. Прикладом класифікації є фільтри спаму, в яких входами є повідомлення електронної пошти (або чогось іншого), а класами є «спам» та «не спам».
- Регресії (англ. regression). Керовані задачі де виходами є безперевні, а не дискретні, значення відповідно до передбачення моделі.
- Кластеруванні (англ. clustering). Входи діляться на групи, на відміну від класифікації, групи не відомі заздалегідь, що зазвичай робить це завданням для спонтанного навчання. Кластеризування має поширену реалізацію у вигляди k-середнього алгоритму, який дозволяє поділити

дані на відповідні підкластери (англ. subclusters) які виражені певною схожістю.

- Оцінка густини. Або знаходження певного разподілу входів у деякому просторі за щільністю знаходження в середовищі.
- Зниження розмірності. Полягає в спрощенні, яке має місце в завданні більш раціонального використання ресурсів. Це застосування може суттєво впливати на кінцевий результат та повинне мати чіткі межі та параметри використання. До цієї області також відноситься тематичне моделювання, де задачею є більш поверхневе занурення в матеріал для класифікації певних даних за схожістю.
- Еволюційне навчання (англ. developmental learning). Використовується для навчання роботів та спрямована на вивчення механізмів розвитку, архітектур і обмежень, які дозволяють безперервному вивченню нових навичок і нових знань у втілених машинах. Як і у людських дітей, навчання, як очікується, буде кумулятивним і поступово зростаючою складністю, а також результатом самодослідження світу у поєднанні з соціальною взаємодією.

## 1.4 Список специфічної лексики

Кожне віяння в науці, інформатиці та суспільстві використовує певні слів в іншому сенсі, ніж люди звикли в звичайному житті. Так, в веб-індустрії слово «павук» може означати пошуковий скріпт, який шукає інформацію на сайтах (такі механізми використовуються в Google, Bing, DuckDuckGo та інших пошукових системах). Так само в машинному навчанні є своя специфічна лексика (Таблиця 1).

Таблиця 1. - Специфічна лексика машинного навчання

| **Переклад українською** | **Слово англійською** | **Значення** |
|---|---|---|
| Точність | Accuracy | Точність - це метрика, за допомогою якої можна вивчити, наскільки гарною є модель машинного навчання. А саме, відношення правильно передбачених класів до загальних прогнозованих класів. |
| Зворотнє поширення | Backpropogation | У нейронних мережах, якщо розрахунковий вихід далеко від фактичного виходу (висока помилка), ми оновлюємо зміщення і ваги на основі помилки. Цей процес оновлення ваги та упередженості відомий як «Пропаганда назад» або «Зворотнє поширення». Алгоритми зворотного поширення (BP) працюють, визначаючи втрати (або помилки) на виході, а потім поширюючи їх назад в мережу.<br>Ваги оновлюються, щоб мінімізувати помилку, що виникає з кожного нейрона. |
| Комп'ютерне бачення | Computer Vision | Комп'ютерне бачення - це сфера комп'ютерних наук, яка займається тим, що дозволяє комп'ютерам візуалізувати, обробляти та ідентифікувати зображення / відео таким же чином, як і людське бачення.<br>Деякі з ключових задач Комп'ютерного бачення:<br>● Пішоходи, автомобілі, виявлення дороги в розумних (самостійних) автомобілях<br>● Розпізнавання об'єктів<br>● Відстеження об'єктів<br>● Аналіз руху<br>● Відновлення зображення |
| Набір даних | Dataset | Набір даних - база даних організована в певну |

| | | |
|---|---|---|
| | | структуру даних. У базі даних, наприклад, набір даних може містити набір бізнес-даних (імена, зарплати, контактну інформацію, показники продажів тощо). Кілька характеристик визначають структуру та властивості набору даних. До них відносяться кількість і типи атрибутів або змінних і різні статистичні заходи, застосовні до них, такі як стандартне відхилення і ексцес. |
| Ітерація | Iteration | Ітерація відноситься до кількості разів, коли параметри алгоритму оновлюються під час навчання моделі набору даних. Наприклад, кожна ітерація навчання нейронної мережі приймає певну кількість тренувальних даних і оновлює ваги за допомогою градієнтного спуску або деякого іншого правила оновлення ваги. |
| Позначені дані | Labeled Data | Позначений набір даних має значущу "мітку", "клас" або "мітку", пов'язану з кожною з її записів або рядків. Наприклад, мітки для набору даних із набору зображень можуть бути, чи містить зображення кішку або собаку.<br>Мічені дані зазвичай є більш дорогими для отримання, ніж необроблені дані, оскільки підготовка мічених даних включає в себе ручне маркування кожного фрагмента немічених даних. Мічені дані необхідні для алгоритмів навчання під наглядом. |
| Логістична регресія | Logistic Regression | Простими словами, це передбачає ймовірність виникнення події шляхом підгонки даних до логістичної функції. Отже, вона також відома як логістична регресія. Оскільки він прогнозує ймовірність, вихідні значення лежать між 0 і 1 (як і очікувалося). |
| Перенавчання | Overfitting | Модель вважається переповненою, коли вона добре працює на наборі даних для тренування, але не працює на тестовому наборі. Це відбувається, коли модель є занадто чутливою і захоплює випадкові шаблони, які присутні тільки в навчальному наборі даних.<br>Є два способи подолання перенавчання:<br>● Зменшити складність моделі |

| | | |
|---|---|---|
| | | ● Регуляризація |
| Параметри | Parameters | Параметри - це набір вимірюваних факторів, що визначають систему. Для моделей машинного навчання параметри моделі є внутрішніми змінними, значення яких можна визначити з даних.<br>Наприклад, вага в лінійній і логістичній регресії підпадає під категорію параметрів. |
| Квартіль | Quartile | Четвертини (квартиль) розділяє ряд на 4 рівні частини. Для будь-якої серії існують 4 квартилі, позначені Q1, Q2, Q3 і Q4. Вони відомі як Перший квартиль, Другий квартиль і так далі. |
| Активаційна функція | Activation function | Функція (наприклад, логістична або сигмоїдна функція), яка приймає зважену суму всіх входів з попереднього шару, а потім генерує і передає вихідне значення (зазвичай нелінійне) до наступного шару. |
| Глибина | Depth | Кількість шарів (включаючи будь-які шари вбудовування) в нейронну мережу, яка вивчає ваги. Наприклад, нейронна мережа з 5 прихованими шарами і 1 вихідним шаром має глибину 6. |
| Епоха | Epoch | Повне навчання проходить по всьому набору даних таким чином, що кожен приклад був помічений один раз. Таким чином, епоха являє собою ітерації з загальна кількість прикладів поділена на кількість в партії. |
| Приклад | Instance | Прийнято вважати синонімом слова «приклад». Найчастіше використовують англіцизм «інстанс». |

# 2. ПРОЦЕС МАШИННОГО НАВЧАННЯ

Процес машинного навчання можна описати як автоматизоване навчання з малим чи без людського втручання. Цей процес включає програмування комп'ютерів таким чином, щоб вони навчалися з доступних входів. Основною метою машинного навчання є вивчення та побудова алгоритмів, які можуть вдосконалюватись на базі попередніх даних і робити прогнози на нові дані.

Для навчання використовуються тренувальні дані (тренувальні дата-сети), що представляють досвід, а вихідні дані є будь-яким досвідом системи, який зазвичай приймає форму іншого алгоритму, який може виконувати певне завдання. Вхідні дані для системи машинного навчання можуть бути числовими, текстовими, аудіо, візуальними або мультимедійними. Відповідні вихідні дані системи можуть бути числом з плаваючою точкою, наприклад, швидкість ракети, ціле число, що представляє категорію або клас, наприклад, голуб або соняшник від розпізнавання зображення.

## 2.1 Етапи машинного навчання

Завдання машинного навчання можна поділити на наступні етапи, які важливо дотримуватись для результативного використання знань та ресурсів.

2.1.1 Пошук та вибірка. Необхідно обрати релевантний (відповідаючий умовам задачі) дата-сет (від англ. data set - вибірка даних), за необхідності,

привести його до умов аналізу, виконати чистку та формалізацію сету для більш чіткого розпізнавання алгоритмом.

2.1.2 Вибір алгоритму. Необхідно впевнитись що методика та алгоритм відповідають вимогам роботи, можуть взаємодіяти з потрібними параметрами даних та є найбільш відповідні для поставленої задачі.

2.1.3 Створення аналітичної моделі. Беручи за основу алгоритм створити алгоритм дії, встановити формули та коефіцієнти відповідно до задачі та підготувати модель для тренування.

2.1.4 Тренування моделі. Використовуються тренувальні дата-сети, завдяки яким модель перевіряє відповідність вихідних даних з тестовими значеннями. Цей етап включає в себе перегляд параметрів, коефіцієнтів або формул за потреби.

2.1.5 Використання моделі. На цьому етапі модель використовує дані для створення показників та рішення задачі в режимі “продакшену” (від англ. production - виробництво).

## 2.2 Алгоритми машинного навчання

Так само, як існує майже безмежна кількість варіантів використання машинного навчання, не існує й браку алгоритмів машинного навчання. Вони варіюються від досить простих (частіше, лінійних) до дуже складних(наприклад, нейронні мережі з не простою схемою взаємодії та розгалуженням).

Далі наведено найпоширеніші моделі машинного навчання.

2.2.1 Алгоритми кореляції. Цей клас алгоритмів машинного навчання передбачає визначення кореляції - як правило, між двома змінними - і використання цієї кореляції для прогнозування майбутніх точок даних.

2.2.2 Дерева рішень. Ці моделі використовують спостереження щодо певних дій і визначають оптимальний шлях для досягнення бажаного результату.

2.2.3 K-середня кластеризація. Ця модель групує певну кількість точок даних у певну кількість угруповань на основі подібних характеристик.

2.2.4 Нейронні мережі. Ці глибокі моделі навчання використовують великі обсяги навчальних даних для визначення співвідношень між багатьма змінними, щоб навчитися обробляти вхідні дані в майбутньому.

2.2.5 Навчання підкріплення. Ця область глибокого навчання включає в себе моделі, які повторюють багато спроб завершити процес та отримують оцінку після виконання відповідно до результатів. Етапи які створюють сприятливі результати - винагороджуються, а кроки що призводять до небажаних результатів - отримають «покарання», поки алгоритм не вивчить оптимальний процес.

# 3. ВИКОРИСТАННЯ МАШИННОГО НАВЧАННЯ

Як вже було зазначено вище - існує схема реалізації (етапи) машинного навчання в дії. Використовуючи ці описи побудуємо свою нейронну мережу, яку навчимо простим принципом класифікації.

На даний момент найбільш використовуваний варіант «Getting Started» або так званий «Hello World» (найбільш використовувані назви для поняття «Швидкого старту») в світі машинного навчання - класифікація рисових квітів (англ.  classification of iris flowers) від Фішера (R.A. Fisher).

## 3.1 Про задачу

Р. А. Фішер опублікував статтю «Використання численних вимірювань у таксономічних задачах» ще 1936 року, які включали набір даних, який ми будемо використовувати. Цей iris-датасет використовується знову і знову по всій літературі, так що це хороший набір даних для ознайомлення. Більш важливим є те, що це дуже чиста проблема класифікації з чотирьох безперервних змінних. Тобто кожен зразок маркується як один з трьох специфічних видів рисових квітів разом з довжиною і шириною листка чаші і пелюстки цього зразка.

Повний набір даних має 150 зразків, з рівним розподілом 50 зразків з кожного класу, де клас є певним типом діафрагми. Назви видів - "setosa", "versicolor", "virginica". Ось уривок із даних:

Таблиця 2 - Частина тестових даних

| **Sepal Length (cm)** | **Sepal Width (cm)** | **Petal Length (cm)** | **Petal Width (cm)** | **Iris Species** |
|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | 0 |
| 5.7 | 4.4 | 1.5 | 0.4 | 0 |
| 5.0 | 2.0 | 3.5 | 1.0 | 1 |
| 6.6 | 3. | 4.4 | 1.4 | 1 |
| 7.6 | 3. | 6.6 | 2.1 | 2 |
| 6.9 | 3.2 | 5.7 | 2.3 | 2 |

3.2 Проектування моделі

Відповідно до інформації що ми маємо, ми можемо припустити, що певний вид має свій діапазон розмірів листків, беручи до уваги такі припущення ми можемо побудувати модель дерева рішень за допомогою якого ми будемо класифікувати квітку за параметрами.

В нашому випадку ми бачимо, наприклад, що параметр “petal lenght” <= “2.5 cm” може бути лише у квітки «setosa», тобто беручи лише цю рівність можна класифікувати об'єкт. У випадку ж, коли цей параметр не задовільний (має булеве значення “False”) то ми можемо припустити, що ця квітка відноситься до двох інших класів. Вибудовуючи таким чином модель порівняння ми побудуємо залежності класу від параметрів (Рисунок 1).

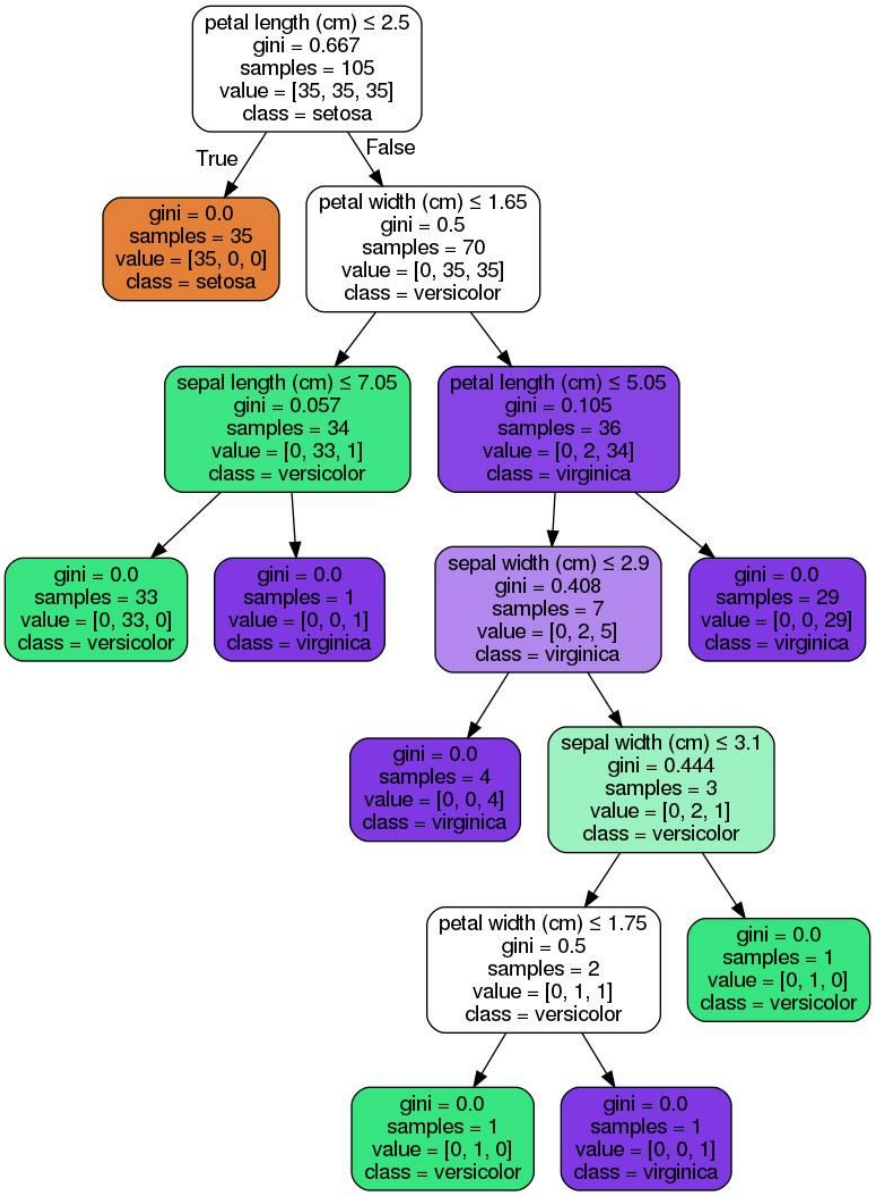


Рисунок 1 - Модель дерева рішень

3.3 Підготовка до процесу моделювання

Окремо варто розповісти про підготовку ЕОМ (в нашому випадку комп’ютеру на базі операційної системи Ubuntu) до процесу моделювання.

Для того, щоб виповнити реалізацію найпростішої задачі варто обрати мову програмування на якій буде реалізовано модель. В нашому випадку це

мова - Python. Окрім цього, можна взяти вже готові бібліотеки за допомогою яких реалізовуються кінцеві системи, наприклад, SciPy, pandas та scikit-learn.

Також треба підготувати дані (датасети). В нашому випадку ми можемо взяти вже готові дані в форматі CSV (англ. від абревіатури Comma Separated Values) з сайту «UCI Machine Learning Repository» (посилання: https://archive.ics.uci.edu/ml/datasets/Iris).

Напишемо код (Додаток 1) який буде виконувати зчитування, поділ сету на тестовий та навчальний, побудову дерева та візуалізацію нашого дерева рішень.

В результаті роботи ми отримаємо файл візуалізації дерева рішень (Рисунок 1), повідомлення про точність класифікації та пакет з даними моделі. Цю модель ми можемо застосувати знов виконавши ті самі дії над даними та взявши в передбачення, наприклад, іншу кількість даних або свої дані.

Так, беручи 50% випадкових даних для навчання ми отримаємо такий результат:

- Кількість даних у відсотках: 30% Точність: 100.0%
- Кількість даних у відсотках: 50% Точність: 98.66666666666667%
- Кількість даних у відсотках: 90% Точність: 99.25925925925925%
- Кількість даних у відсотках: 100% Точність: 99.33333333333333%

Всі матеріали використані для роботи надано в відкритому доступі за посиланням - https://github.com/dmytrohoi/ml_first_step_ua .

# ВИСНОВКИ

Беручи до уваги отримані результати та інформацію яку надано в ході роботи можна зробити висновки, що машинне навчання грає велику роль у розвитку суспільства в усіх сферах за рахунок можливостей які доступні використовуючи різні алгоритми й методи моделювання та передбачення. В результаті використання навіть найпростіших моделей дерев рішень можна спростити роботу по розпізнаванню та класифікації об’єктів.

В сучасних технологічних рішеннях та продуктах багатьох передових компаній використовується безліч подібних результатів машинного навчання. Найбільш поширений на даний момент є - глибоке навчання. За допомогою цього методу будуються різного роду складності нейронні мережі які використовуються для прийняття рішень у складних умовах, розпізнавання облич, керування автомобілями і т.д.

# СПИСОК ЛІТЕРАТУРИ


1. *Anand Rajaraman, Jure Leskovec, Jeffrey D. Ullman*. Mining of Massive Datasets. *Наука та іновації*. 2014. №2
2. pandas: powerful Python data analysis toolkit. *Наука та іновації*. 2018. URL: http://pandas.pydata.org/pandas-docs/stable/ (дата звернення: 30.11.2018 р.)
3. Numpy and Scipy Documentation. *Наука та іновації*. 2018. URL: https://docs.scipy.org/doc/ (дата звернення: 25.11.2018 р.)
4. *Toby Segaran*. Programming Collective Intelligence: Building Smart Web 2.0 Applications. *Комп'ютерні науки.* 2010. 382 с.
5. *Andreas Muller*. Introduction to Machine Learning with Python: A Guide for Data Scientists. *Комп'ютерні науки.* 2016.
6. *Drew Conway, John Myles*. Machine Learning for Hackers: Case Studies and Algorithms to Get You Started. *Комп'ютерні науки.* 2012.
7. *Willi Richert, Luis Pedro Coelho*. Building Machine Learning Systems with Python. *Комп'ютерні науки.* 2013.
8. *Aurelien Geron*. Hands-On Machine Learning with Scikit-Learn and Tensor Flow: Concepts, Tools, and Techniques to Build Intelligent Systems. *Комп'ютерні науки.* 2017.
9. *William McKinney*. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Ipython. *Комп'ютерні науки.* 2017.
10. *Peter Bruce, Andrew Bruce*. Practical Statistics for Data Scientists: 50 Essential Concepts. *Комп'ютерні науки.* 2017.

11. Scikit-learn tutorial: statistical-learning for sientific data processing. *Наука та іновації*. URL: http://gael-varoquaux.info/scikit-learn-tutorial/ (дата звернення: 30.11.2018 р.)
12. *Shai Shalev-Shwartz, Shai Ben-David.* UNDERSTANDING MACHINE LEARNING: From Theory to Algorithms. *Комп'ютерні науки.* 2014.
13. *Nils J. Nilsson*. Introduction to Machine Learning. *Комп'ютерні науки.* 2015.
14. *Ryan J. Urbanowicz, Jason H. Moore*. Learning Classifier Systems: A Complete Introduction, Review, and Roadmap. Journal of Artificial Evolution and Applications. *Комп'ютерні науки.* 2009. 25 с.
15. *David MacKay*. Information Theory, Inference, and Learning Algorithms. *Комп'ютерні науки.* 2003.
16. *Stuart J. Russell, Peter Norvig*. Artificial Intelligence: A Modern Approach, Third Edition. *Комп'ютерні науки.* 2010.
17. *Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar*. Foundations of Machine Learning. *Комп'ютерні науки.* 2012.
18. *Allen B. Downey.* Think Stats: Probability and Statistics for Programmers. *Комп'ютерні науки.* 2011.
19. *Allen B. Downey.* Think Bayes: Bayesian Statistics Made Simple. *Комп'ютерні науки.* 2012.
20. *D. Kriesel.* A Brief Introduction to Neural Networks. *Комп'ютерні науки*. 2005.
21. David Barber. Bayesian Reasoning and Machine Learning. *Комп'ютерні науки*. 2016.

22. *Oliver Theobald*. Machine Learning For Absolute Beginners: A Plain English Introduction (Second Edition). *Комп'ютерні науки*. 2017.
23. *John Paul Mueller, Luca Massaron*. Machine Learning For Dummies. *Комп'ютерні науки*. 2017.
24. *Peter Harrington*. Machine Learning in Action. *Комп'ютерні науки*. 2012.
25. *Sebastian Raschka*. Python Machine Learning, 1st Edition. *Комп'ютерні науки*. 2015.

# ДОДАТОК 1

```
# Імпорт необхідних бібліотек
import pandas
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.externals import joblib
from sklearn.datasets import load_iris

# Завантаження датасету
URL = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
NAMES = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
iris = pandas.read_csv(URL, names=NAMES)

# Перевірка кількості значень в сеті
print("Значень в сеті: {}".format(iris.shape))

# Розмежування target функцій від вхідних даних
y = iris['class']
X = iris.drop('class',axis=1)

# Використовуючи вбудовану функцію поділимо дані на тестові й перевірочні
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                        test_size=0.3, # % кількість тестових значень
                                        random_state=100,
                                        stratify=y) # значення для перевірки правильності

# Побудова дерева рішень на базі наших значень
iris_tree = tree.DecisionTreeClassifier()
iris_tree.fit(X_train,y_train)

# Ця функція scipy має наступні параметри
#DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
#            max_features=None, max_leaf_nodes=None,
#            min_impurity_split=1e-07, min_samples_leaf=1,
#            min_samples_split=2, min_weight_fraction_leaf=0.0,
#            presort=False, random_state=None, splitter='best')

# Для візуалізації дерева використаємо вбудовану функцію
iris=load_iris()
print(tree.export_graphviz(iris_tree, # Наше дерево рішень
                        out_file='iris.dot', # Вихідне зображення
                        feature_names=iris.feature_names,
                        class_names=iris.target_names,
                        filled=True, rounded=True,
                        special_characters=True))
```

```
# Виконаємо перевірку моделі взявши значення з тестового сету
y_pred = (iris_tree.predict(X_test))
print ("Точність передбачення: {}".format(accuracy_score(y_test, y_pred)* 100));

# Збереження моделі дерева
joblib.dump(iris_tree, 'iris.pkl')

# Відновлення моделі з файлу
# iris_tree_backup = joblib.load('iris.pkl')
# iris_tree_backup.predict(X_test)
```