МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

АНАЛІЗ ТА РОЗПІЗНАВАННЯ БІОМЕДИЧНИХ СИГНАЛІВ МЕТОДАМИ ШТУЧНОГО ІНТЕЛЕКТУ
ЛАБОРАТОРНИЙ ПРАКТИКУМ

Навчальний посібник

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня магістра
за освітньою програмою «Електронні мікро- і наносистеми та технології»
спеціальності 153 Мікро- та наносистемна техніка

Укладачі: К.О.Іванько, А.О.Попов

Електронне мережне навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2022

Рецензент *Обухова Т.Ю.,* к.т.н, доцент кафедри МЕ

Відповідальний редактор *Тимофєєв В.І.*, д.т.н, проф., проф.каф. ЕІ

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № Х від DD.MM.YYYY р.)
за поданням Вченої ради факультету/навчально-наукового інституту
(протокол № Х від DD.MM.YYYY р.)

Навчальний посібник призначений для здобувачів ступеня магістра за освітньою програмою «Електронні мікро- і наносистеми та технології» спеціальності 153 Мікро- та наносистемна техніка. Він допоможе студентам у виконанні лабораторних робіт з дисципліни «Аналіз та розпізнавання біомедичних сигналів методами штучного інтелекту». В посібнику викладено інформацію щодо методів обробки та аналізу біомедичних сигналів різного походження, а також застосування методів штучного інтелекту для потреб діагностики. Приділено увагу публічно доступним базам даних біомедичних сигналів, а також застосуванню методів машинного навчання на практиці для класифікації біомедичних даних, сигналів та зображень.

Реєстр. № НП ХХ/ХХ-ХХХ. Обсяг Х,Х авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Перемоги, 37, м. Київ, 03056
https://kpi.ua
Свідоцтво про внесення до Державного реєстру видавців, виготовлювачів
і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

ЗМІСТ

Вступ

Штучний інтелект - це сукупність технологій автоматизованого вирішення складних задач, які раніше були доступні лише людині. Застосування методів машинного навчання відбувається в області комп'ютерного зору, обробки природної мови, аналізі та прогнозуванні часових рядів, та робототехніці [1-6]. Мета дисципліни «Аналіз та розпізнавання біомедичних сигналів методами штучного інтелекту» – отримання теоретичних знань та навичок практичного застосування обробки біомедичних сигналів та зображень для машинного навчання у біомедичних електронних системах [7]. Дисципліна «Аналіз та розпізнавання біомедичних сигналів методами штучного інтелекту» базується на знаннях, набутих під час вивчення вищої математики, основ техніки вимірювань, схемотехніки, теорії сигналів, імовірнісних основ обробки даних, біофізики.

Лабораторні роботи виконуються за темами: програмування на мові Пайтон для машинного навчання, ресурси біомедичних даних, методи попередньої обробки та інженерії ознак для машинного навчання, машинне навчання без вчителя, машинне навчання з вчителем, нейронні мережі.

Після засвоєння навчальної дисципліни студенти мають отримати навички обробки, перетворення, аналізу та класифікації біомедичних сигналів і зображень, розробки прикладних програм обробки біомедичних сигналів і зображень. Вони зможуть реалізовувати методи обробки та аналізу біомедичних сигналів і зображень та застосовувати методи машинного навчання для інтелектуального аналізу даних в біомедичних електронних системах. Ці знання є необхідними для формування світогляду фахівців з електронної техніки біомедичного призначення.

Лабораторна робота №1.
Основи програмування на мові Пайтон для машинного навчання

Мета роботи: Ознайомлення з основами програмування мовою Python на рівні, достатньому для виконання лабораторних робіт з дисципліни.

Загальні теоретичні відомості

Python – високорівнева мова програмування загального призначення, орієнтована на підвищення продуктивності розробника і читання коду з мінімалістичним синтаксисом ядра та великим набором корисних функцій. Python підтримує структурне, узагальнене, об'єктно-орієнтоване, функціональне і аспектно-орієнтоване програмування. Мова Python є однією з найбільш поширених мов програмування, та застосовується для розробки веб-додатків, у машинному навчанні, аналізі та візуалізації даних, розробці графічних інтерфейсів, роботі з аудіо та відео даними, програмуванні вбудованих систем та ін.

На першій лабораторній роботі треба опанувати основи Python на рівні, достатньому для виконання наступних лабораторних робіт, присвячених реалізації методів машинного навчання.

Для цього корисними будуть наступні бібліотеки:

NumPy: математичні операції

SciPy: науково-технічні обчислення

Matplotlib (Seaborn): візуалізація даних

SymPy: символьна математика

Pandas: обробка і аналіз табличних даних

З метою написання коду в Python, інтегрування модулів і бібліотек, необхідно обрати для себе та підготувати інтегроване середовище розробки (наприклад, PyCharm, Komodo IDE, Spyder (рис.1.1), PyScripter або інш.)

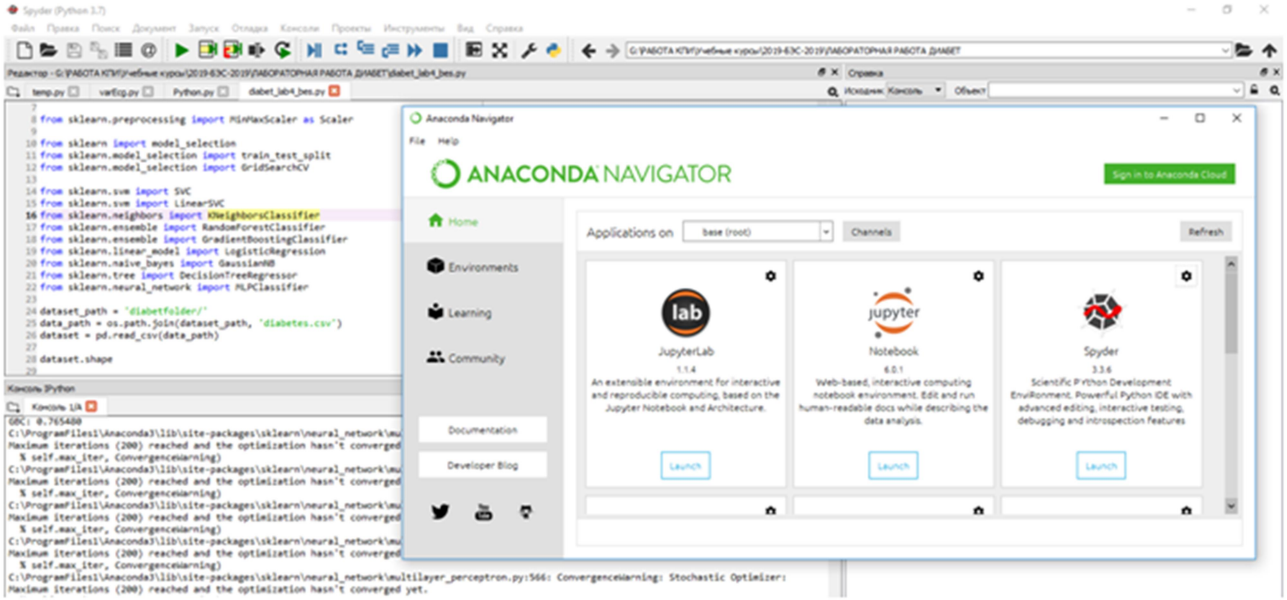


Рис.1.1. Дистрибутив Anaconda, використання Spyder для написання коду на Python

Для цього студенти мають обрати для себе зручний для вивчення онлайн-ресурс, з допомогою якого будуть вивчати мову Python. Пропонуються звернути увагу в першу чергу на такі ресурси та онлайн-курси:

The Python Tutorial
https://docs.python.org/3/tutorial/

Основи програмування
https://courses.prometheus.org.ua/courses/KPI/Programming101/2015_T1/about

Python для начинающих, видеоуроки на русском
https://www.youtube.com/watch?v=cKRRysbQZsM

Python for Data Science and AI (IBM by Coursera)
https://www.coursera.org/learn/python-for-applied-data-science-ai#syllabus

Python for Everybody Specialization (the University of Michigan by Coursera)
1. Programming for Everybody (Getting Started with Python)

https://www.coursera.org/learn/python?specialization=python#syllabus

2. Python Data Structures

https://www.coursera.org/learn/python-data?specialization=python#syllabus

How to Learn Python in 21 Days?

https://www.geeksforgeeks.org/how-to-learn-python-in-21-days/?ref=leftbar-rightbar

Learn Python 2

https://www.codecademy.com/learn/learn-python

Також корисним для створення програм на мові Python використання середовища розробки. Пропонується звернути увагу та використовувати в роботі такі продукти:

- **PyCharm Community Edition https://www.jetbrains.com/pycharm/** – середовище для розробки. **Ознайомлення з роботою в ньому можна поєднувати з вивченням мови Python** в цьому курсі: PyCharm Edu -- Learning Python https://www.jetbrains.com/help/education/learner-start-guide.html?section=Introduction%20to%20Python

- **Jupyter https://jupyter.org/** – інтерактивне веб-середовище для розробки. Для ознайомлення з роботою в Jupyter можна скористатися цими ресурсами:

- Jupyter Notebook Tutorial: The Definitive Guide

https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook

- Jupyter Notebook for Beginners: A Tutorial

https://www.dataquest.io/blog/jupyter-notebook-tutorial/

- Jupyter Tutorial

https://www.tutorialspoint.com/jupyter/index.htm

Порядок роботи.

1. Визначитися та обрати онлайн-ресурс для вивчення мови Python, почати навчання.
2. Надати сертифікат про успішне проходження курсу.

Контрольні питання:

1. Які бібліотеки використовуються для реалізації методів машинного навчання?
2. Які інтерактивні середовища для розробки програм на мові Python ви знаєте?
3. Якими онлайн-ресурами для вивчення мови Python ви користувалися?

Лабораторна робота №2

Знайомство з публічно доступними ресурсами біомедичних даних

Мета роботи: ознайомлення з ресурсами публічно доступних баз біомедичних даних та вибір бази даних для подальшого використання у лабораторних і практичних завдань з навчальної дисципліни “Машинне навчання та обробка сигналів в біомедичних електронних системах”

Загальні теоретичні відомості

Лабораторні і практичні заняття з навчальної дисципліни дисципліни “Машинне навчання та обробка сигналів в біомедичних електронних системах” передбачають індивідуальну навчальну траєкторію для кожного зі студентів. Цей підхід реалізується за рахунок обрання кожним зі студентів публічно доступних баз біомедичних даних, одна з яких після обговорення з викладачами буде використовуватися студентом у подальшій роботі на кожній з лабораторних та практичних занять. Таким чином, студент ознайомиться з певною біомедичною проблемою, отримає діагностичну задачу, яку протягом семестру буде вирішувати за допомогою методів машинного навчання. В кінці семестру, після захисту протоколів всіх лабораторних робіт, передбачається підготовка окремого інтегрального звіту з обґрунтуванням вибору найкращого рішення поставленої задачі та презентація цього рішення на занятті.

Physionet

Ресурс (рис.2.1-2.2) доступний за посиланням https://physionet.org/

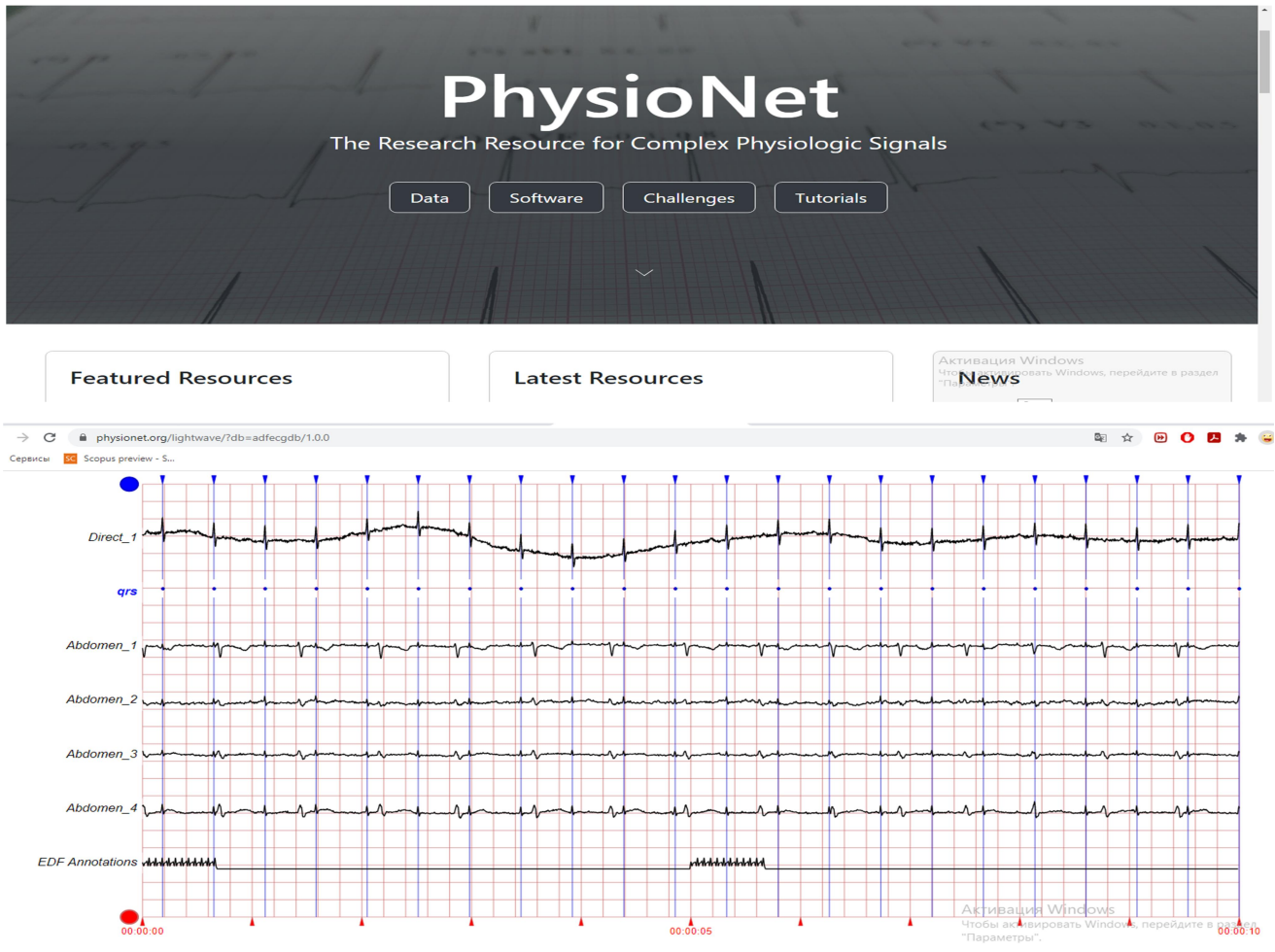


Рис. 2.1. Приклад багатоканального сигналу з бази

Abdominal and Direct Fetal ECG Database

https://physionet.org/content/adfecgdb/1.0.0/

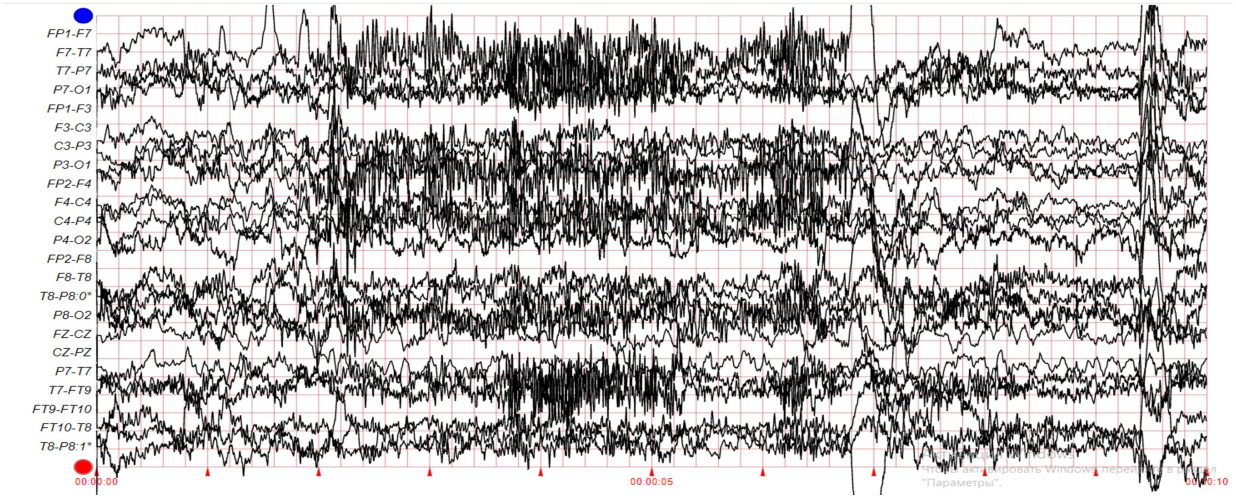


Рис.2.2. Приклад багатоканального сигналу з бази

CHB-MIT Scalp EEG Database

https://physionet.org/content/chbmit/1.0.0/

Kaggle

Ресурс (рис.2.3-2.5) доступний за посиланням https://www.kaggle.com/datasets https://www.kaggle.com/datasets?topic=healthDataset

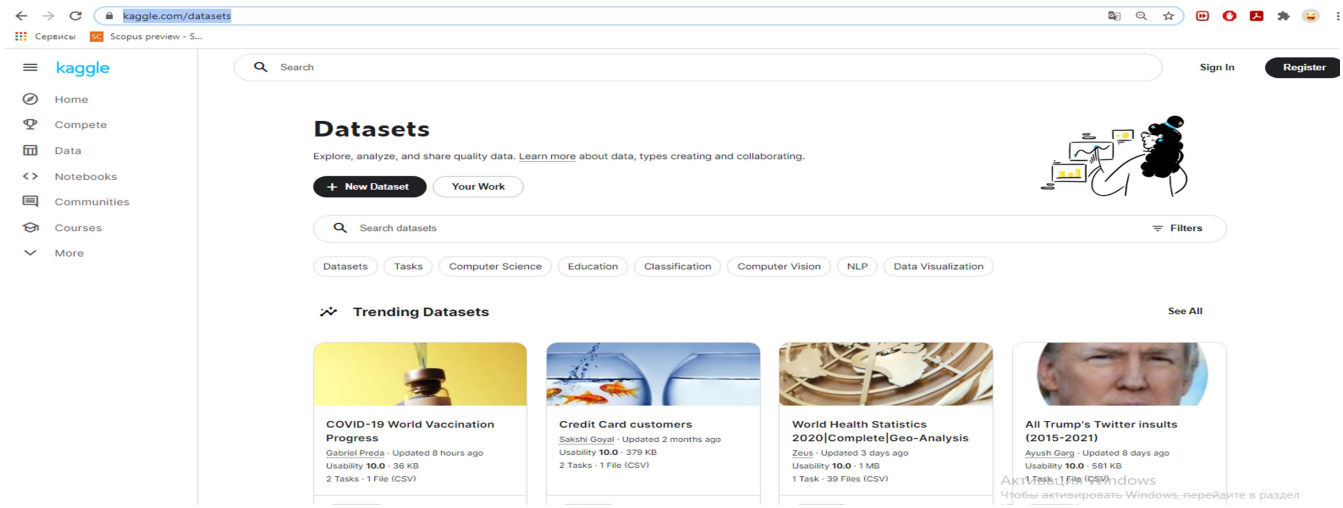


Рис.2.3. Ресурс Kaggle

Pima Indians Diabetes Database

https://www.kaggle.com/uciml/pima-indians-diabetes-database

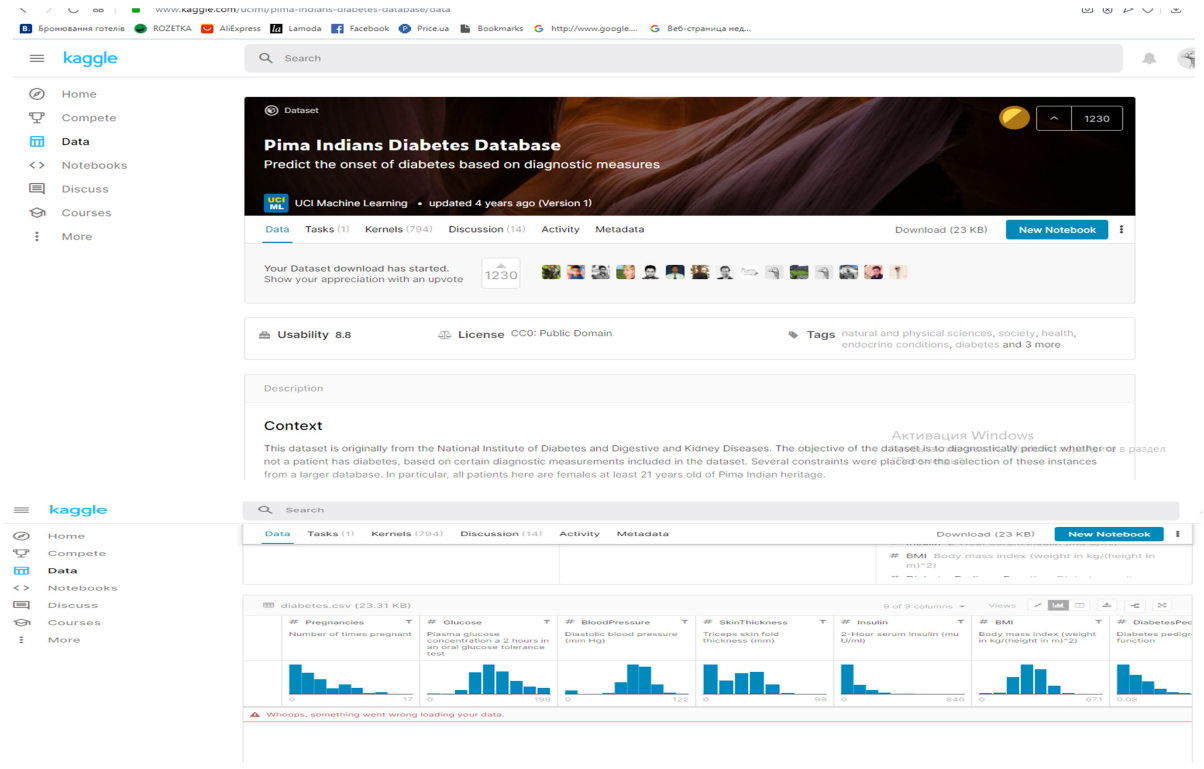


Рис.2.4. Приклад даних з бази, яка створена з метою прогнозування виникнення гестаційного діабету у вагітних на основі аналізу ряду діагностичних показників

Fetal Health Classification

https://www.kaggle.com/andrewmvd/fetal-health-classification

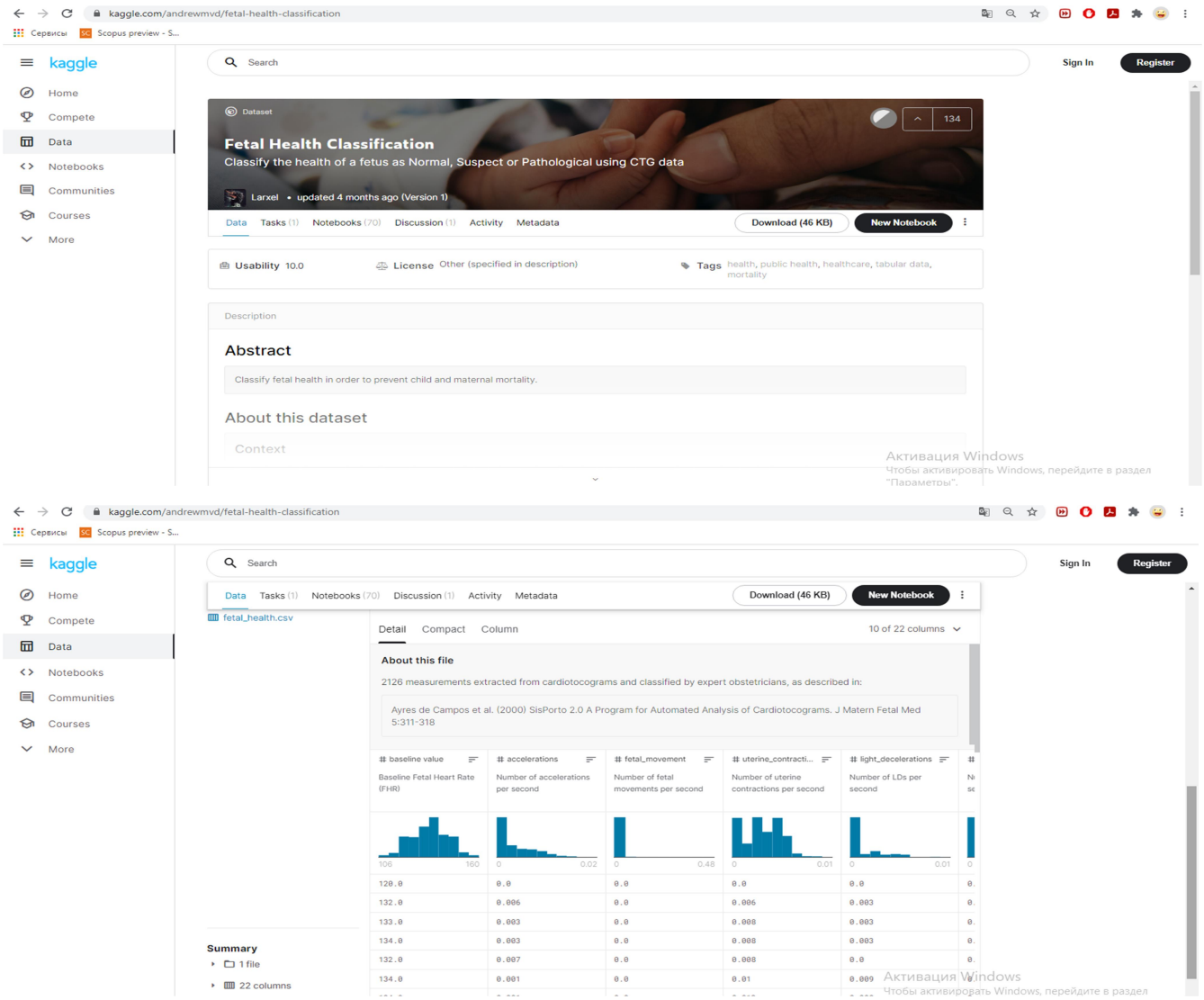


Рис.2.5. Приклад даних з бази, яка створена з метою виявлення патологій розвитку плоду під час вагітності

IEEE Data Port

Ресурс (рис.2.6-2.8) доступний за посиланням https://ieee-dataport.org/

(*для отримання доступу для завантаження даних звернутися до викладача*)

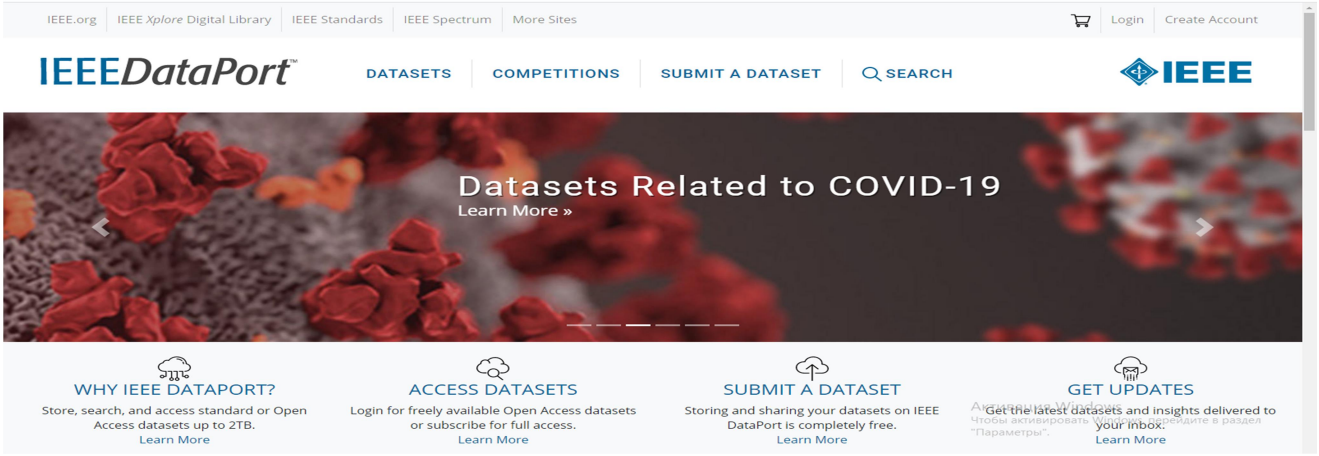


Рис.2.6. Ресурс IEEE Data Port

Розділ біомедичних даних доступний за посиланням

https://ieee-dataport.org/topic-tags/biomedical-and-health-sciences

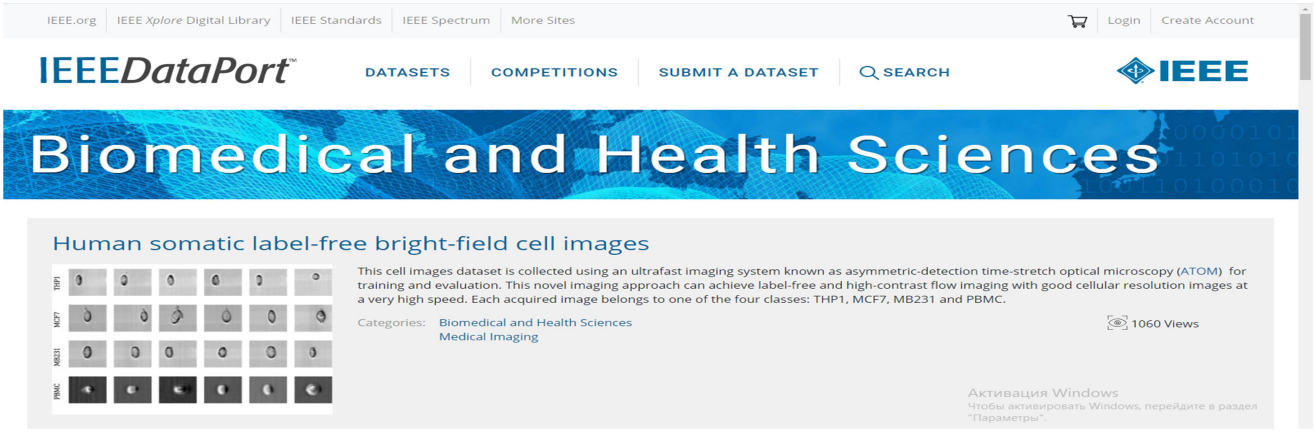


Рис.2.7. Розділ біомедичних даних

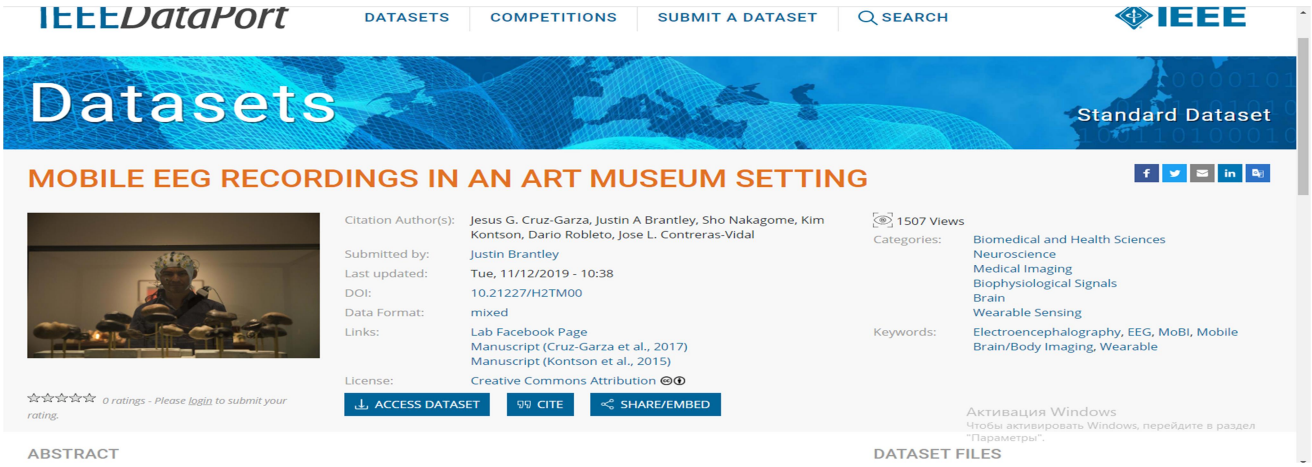


Рис. 2.8. Приклад бази даних для дослідження активності мозку при відвідуванні музею

https://ieee-dataport.org/documents/mobile-eeg-recordings-art-museum-setting

Репозиторій ZENODO

Ресурс (рис.2.9) доступний за посиланням **https://zenodo.org**

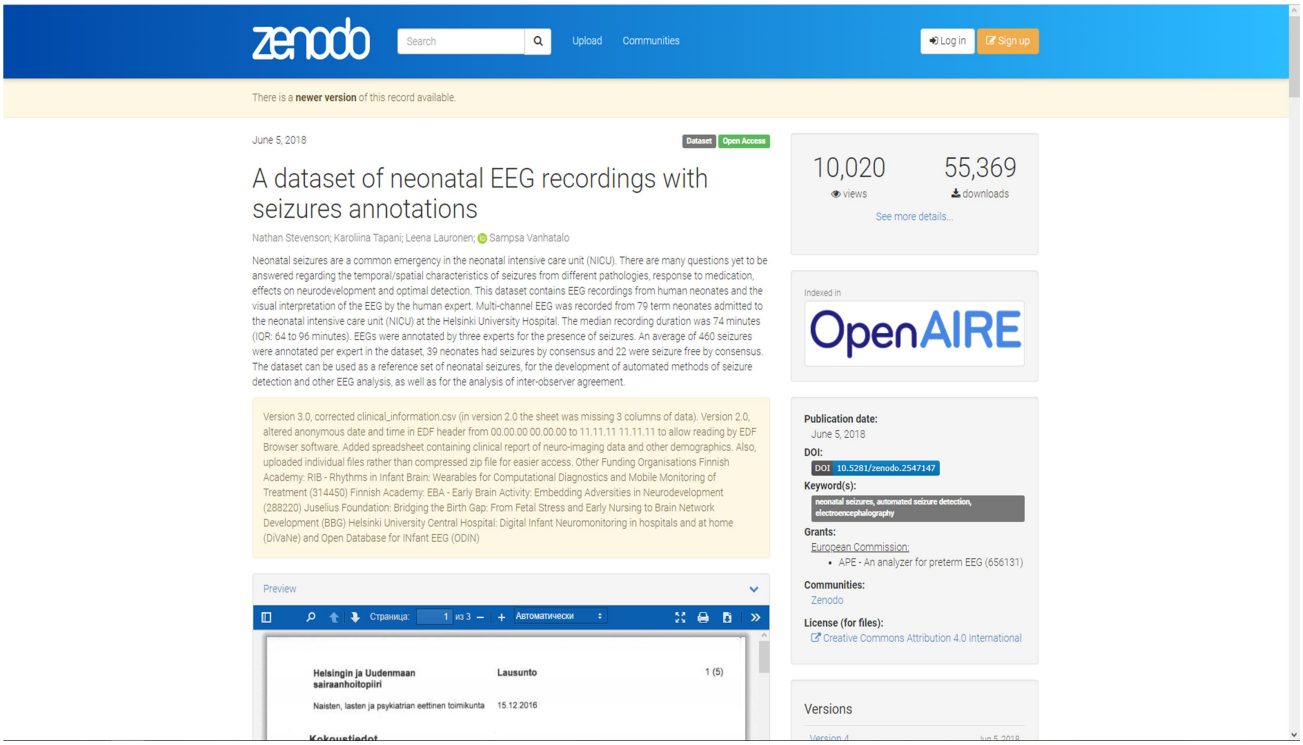


Рис. 2.9. Приклад бази даних для дослідження епілептичних нападів у новонароджених https://zenodo.org/record/2547147#.YFH7cbZR2M8

Робоче завдання

1. Ознайомитися з ресурсами публічно доступних баз біомедичних даних на платформах Physionet, Kaggle, IEEE Data Port, Zenodo або інших.
2. Кожному студенту обрати мінімум 3 бази біомедичних даних, які найбільше зацікавили з точки зору об’єкта та теми дослідження. В пріоритетному порядку в протоколі навести посилання на ці бази даних, принтскрин головної сторінки бази, її опис, задачу дослідження, тип файлів даних.
3. Перевірити наявність даних та їх кількість у кожній базі, приймаючи до уваги, що машинне навчання потребує досить великої вибірки даних для навчання та тестування алгоритмів.
4. Візуалізувати з допомогою Python та навести в протоколі декілька сигналів з кожної бази.

5. Виконати EDA (explorative data analysis). Ознайомитися з засобами візуалізації розподілу даних, наприклад бібліотеками **seaborn** (https://seaborn.pydata.org/tutorial/distributions.html) та **matplotlib** (https://matplotlib.org/stable/plot_types/index.html) та застосовувати їх у подальших пунктах роботи.

6. Навести опис параметрів, які досліджуються, сигналів, які виміряні.

7. Автоматично з допомогою Python отримати з бази даних інформацію про кількість записів, кількість учасників, кількість каналів, тривалість, та т.і.

8. Побудувати гістограми розподілу основних параметрів даних з бази (рис. 2.10-2.11).

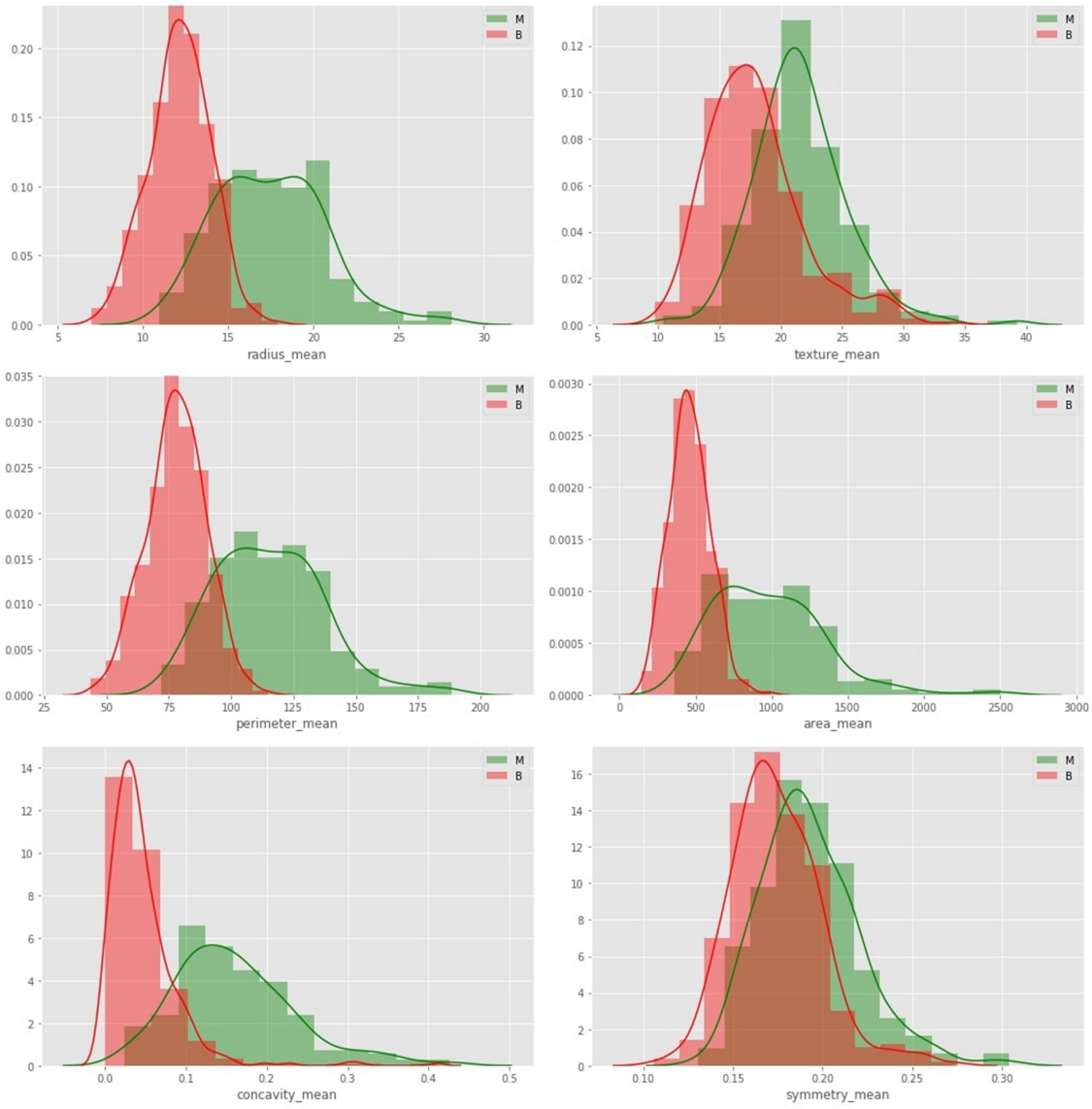


Рис.2.10. Гістограми розподілу основних параметрів даних з бази Breast Cancer Wisconsin Diagnostic Data Set (червоний колір відповідає доброякісним пухлинам, зелений - злоякісним)

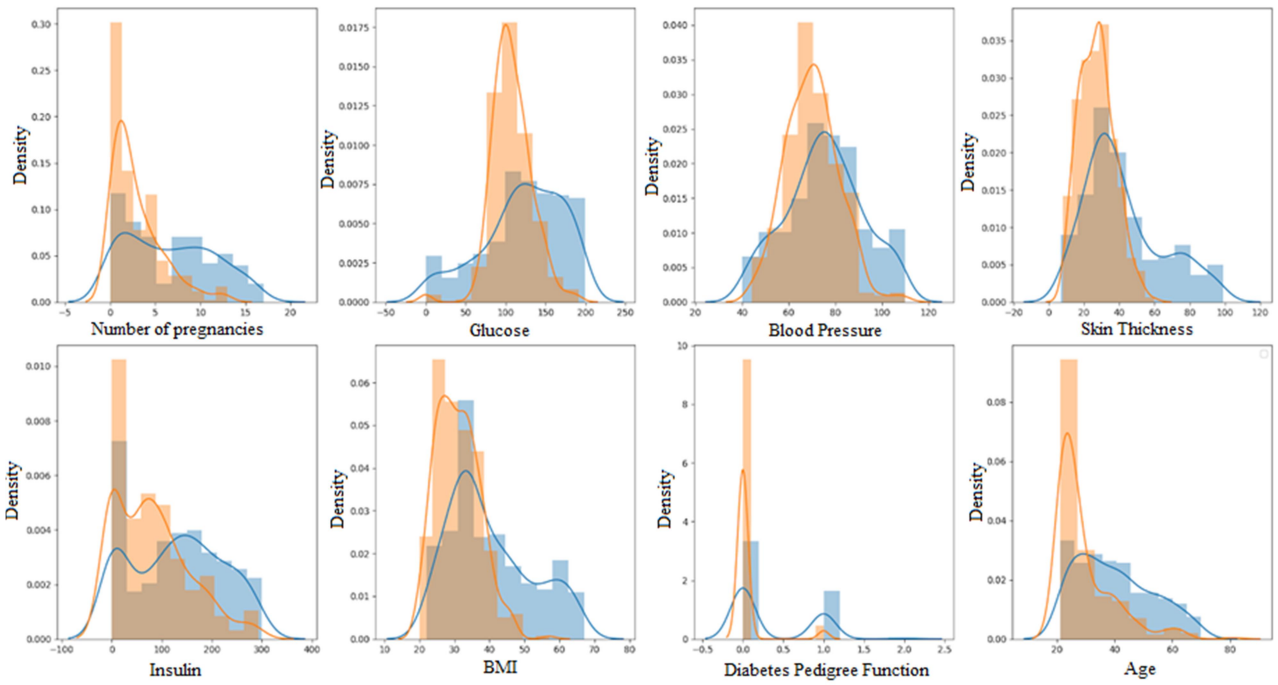


Рис.2.11 Гістограми розподілу основних параметрів даних з бази Pima Indians Diabetes Database (червоний відповідає класу норми, синій — гестаційному цукровому діабету)

9. Проаналізувати матрицю кореляції параметрів (рис.2.12), парні діаграми розсіювання ознак (рис.2.13-2.14) та бокспроти для різних для ознак (рис.2.15).

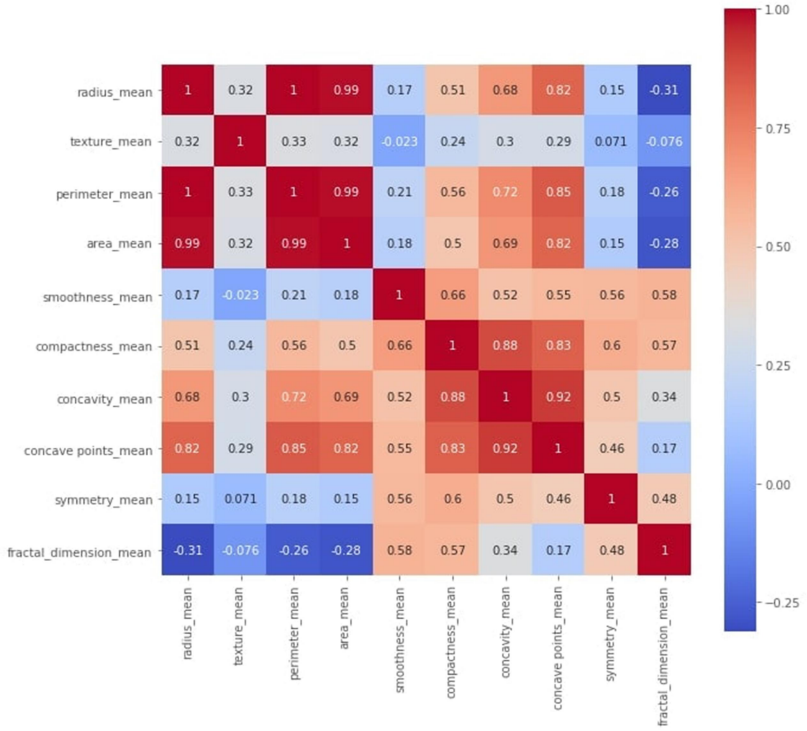


Рис.2.12. Матриця кореляції параметрів, побудовані за допомогою засобів seaborn.heatmap (https://seaborn.pydata.org/generated/seaborn.heatmap.html)

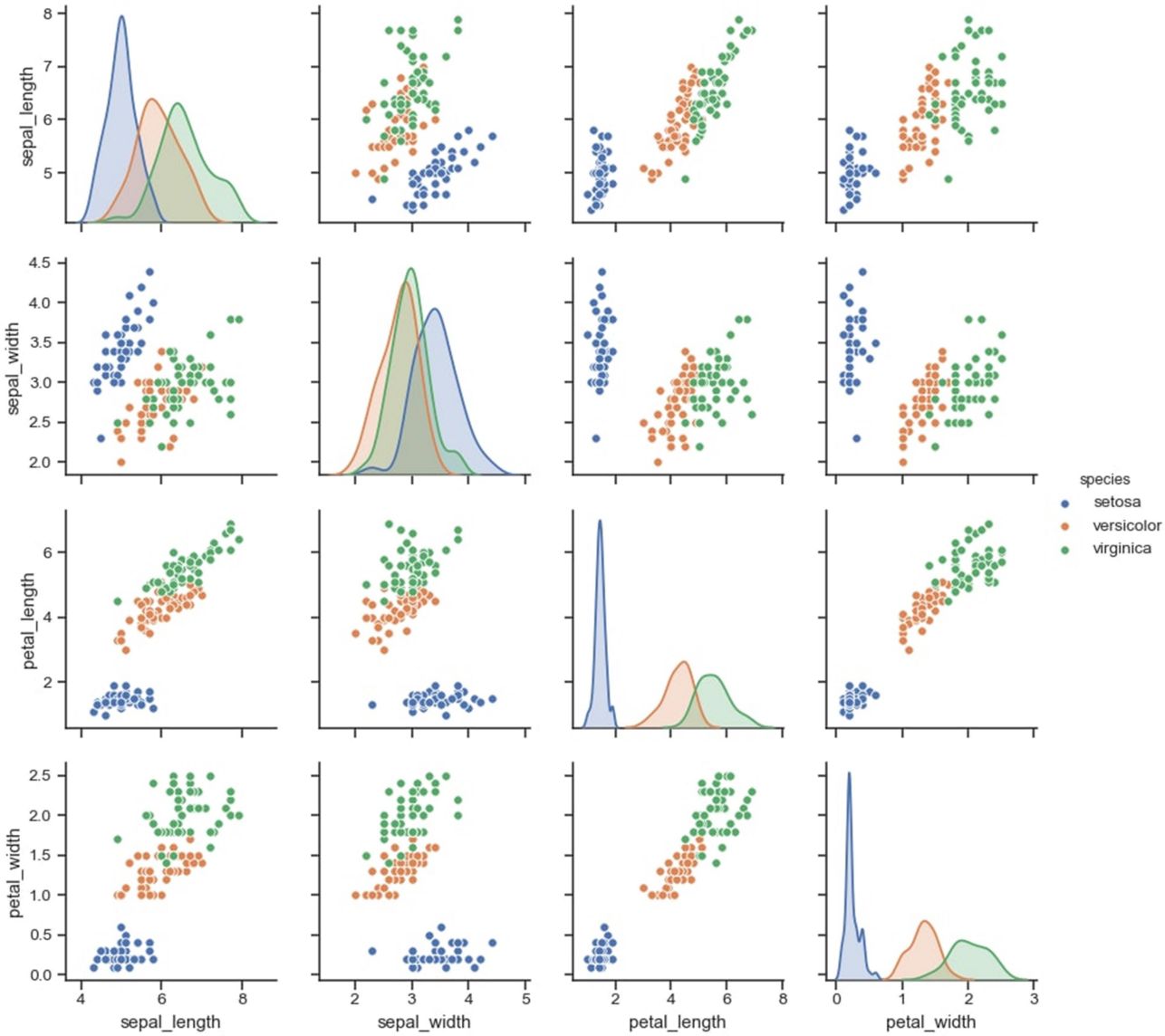


Рис. 2.13. Парна діаграма розсіювання параметрів для задачі з наявністю 3 класів, побудована засобами sns.pairplot (https://seaborn.pydata.org/generated/seaborn.pairplot.html)

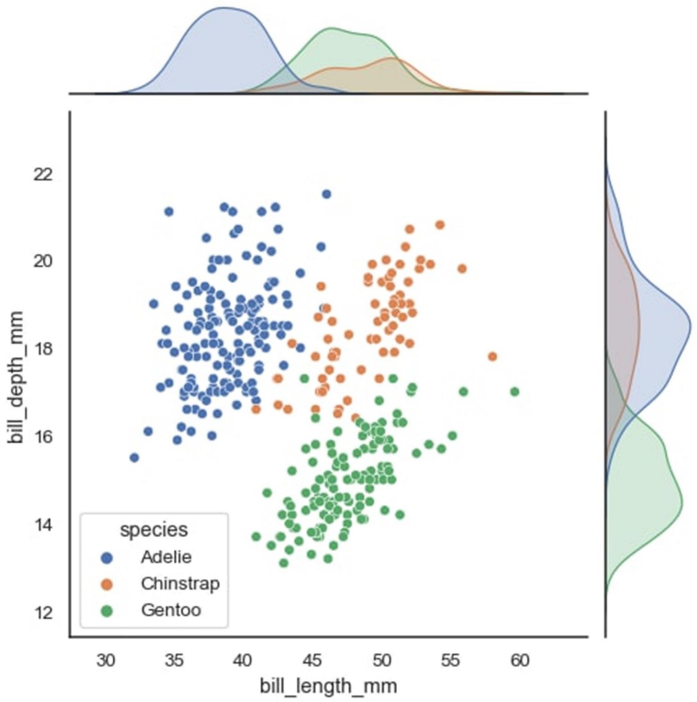


Рис. 2.14. Діаграма розсіювання параметрів для задачі з наявністю 3 класів, побудована за допомогою засобів seaborn.jointplot (https://seaborn.pydata.org/generated/seaborn.jointplot.html)

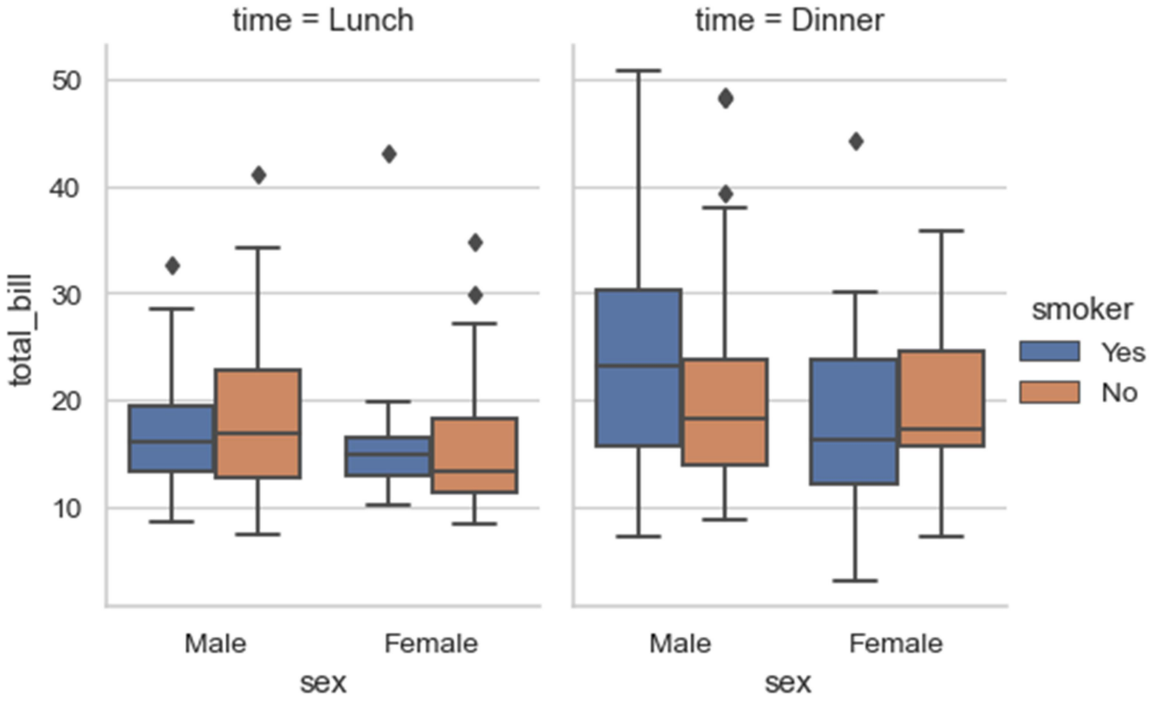


Рис. 2.15. Розподіл даних, побудований за допомогою засобів seaborn.boxplot (https://seaborn.pydata.org/generated/seaborn.boxplot.html)

10. Ознайомитись з науковими статтями з тематики, яка стосується обраних баз сигналів. Для кожної з обраних баз даних сформулювати та навести в протоколі мету власного дослідження та задачі, які будуть розв'язані.

11. Презентувати викладачам свої ідеї щодо задач дослідження, затвердити тему дослідження та внести її у таблицю за посиланням https://docs.google.com/spreadsheets/d/1LyLCG4_G-NX731uD2miVXFv2KwqLPMbhw5TYAVnye80/edit?usp=sharing.

Контрольні питання:

1. З якими відкритими ресурсами даних для машинного навчання ви знайомі?
2. Яким чином вами було проведено дослідницький аналіз даних, які ви будете застосовувати для машинного навчання?
3. Яку інформацію можна отримати з матриці кореляції параметрів?
4. Як побудувати та проаналізувати парні діаграми розсіювання параметрів?

Лабораторна робота №3

Методи попередньої обробки та інженерії ознак для машинного навчання

Мета роботи: ознайомлення з методами та алгоритмами попередньої обробки даних та методами інженерії ознак, що використовуються для машинного навчання.

Загальні теоретичні відомості

Вхідні дані, що в загальному випадку реєструються з пацієнта та підлягають подальшій класифікації, піддаються попередній обробці з метою їх перетворення в необхідний для реалізації алгоритмів машинного навчання вигляд. На етапі аналізу створюється формальний опис біомедичних даних пацієнта, який несе в собі інформацію, що є найбільш істотною з точки зору якості прийняття рішення при класифікації [7]. Процедура подальшої редукції даних ґрунтується на оцінці корисності інформації для правильної класифікації і полягає в виділенні з усієї множини ознак найбільш інформативних ознак, характерних для певного захворювання.

Досить часто у біомедичних задачах мають справу з ознаками, які отримані в результаті дослідження сигналів (рис.3.1-3.2) [7]. Це можуть бути амплітуди та тривалості деяких коливань, форма коливань, ймовірнісні характеристики сигналів (закони розподілу густини ймовірностей, значення моментів тощо), кількість перетинів нульової лінії, розташування локальних максимумів в спектрі сигналів, потужності в певних діапазонах частот та ін. При класифікації зображень ознаками можуть виступати розміри певних областей, середня яскравість пікселів в певній області зображення, гістограми яскравості, спектральні характеристики, морфологічні ознаки (форма областей) та ін.

Попередня обробка даних у машинному навчанні виконується для перетворення даних із необробленої форми у більш придатну для застосування методів машинного навчання форму. Початкові дані, як правило, знаходяться в необробленому форматі, тобто можуть мати шуми, відсутні значення, зайву інформацію, цифри у строковому форматі, або вони можуть бути

неструктурованими. Попередня обробка даних підвищує ефективність та точність моделей машинного навчання.

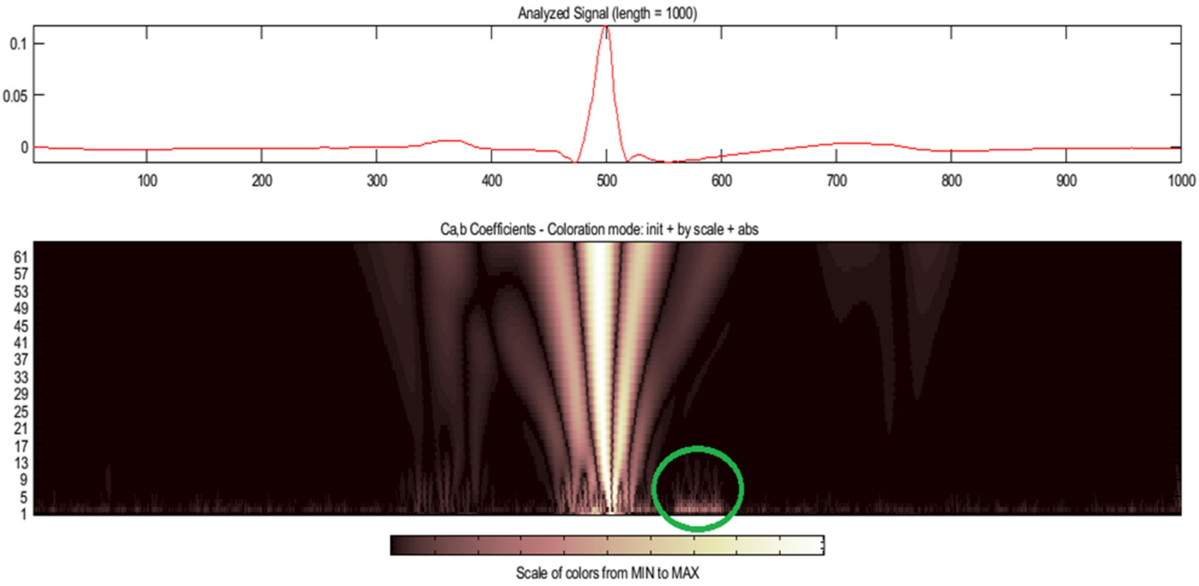


Рис. 3.1. Приклад формування ознак електрокардіосигналу за допомогою вейвлет-аналізу

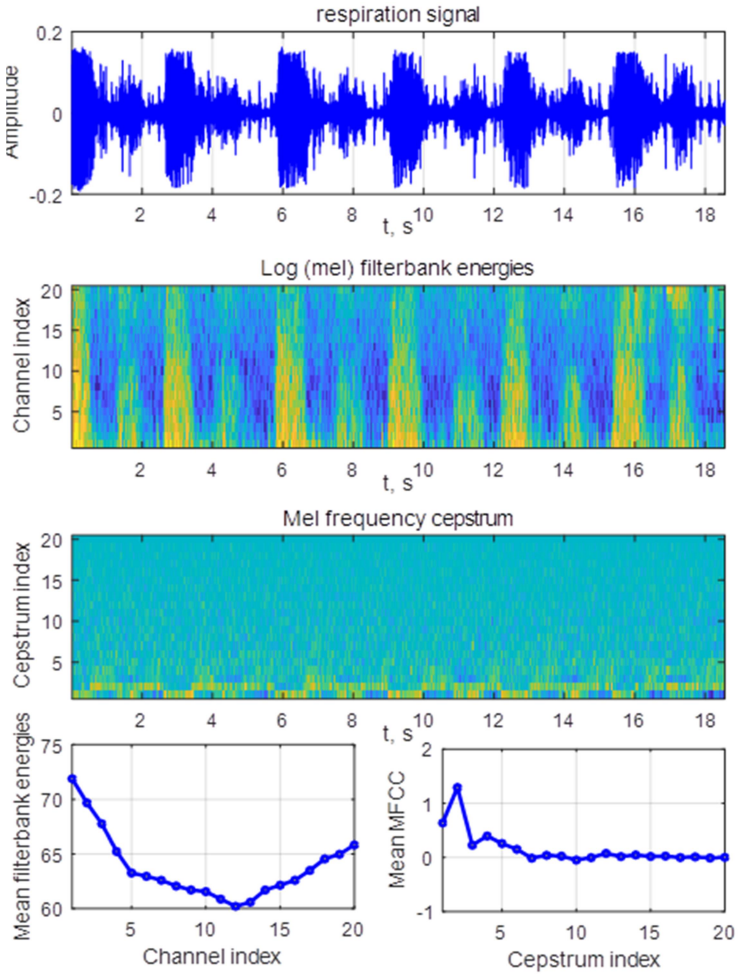


Рис. 3.2. Приклад застосування мел-кепстрального аналізу для формування ознак респіраторного сигналу

Можливі етапи попередньої обробки даних для подальшого застосування методів машинного навчання:

- перевірка і підготовка категоріальних даних (рис.3.3-3.4);
- заміна або вилучення спостережень з відсутніми даними;
- усунення викидів (рис.3.5);
- генерація нових ознак на основі наявних ознак (рис.3.6);
- логарифмічне перетворення даних (рис.3.7);
- масштабування ознак (рис.3.8);
- зниження розмірності даних;
- розбиття даних на навчальний та тестовий набори.

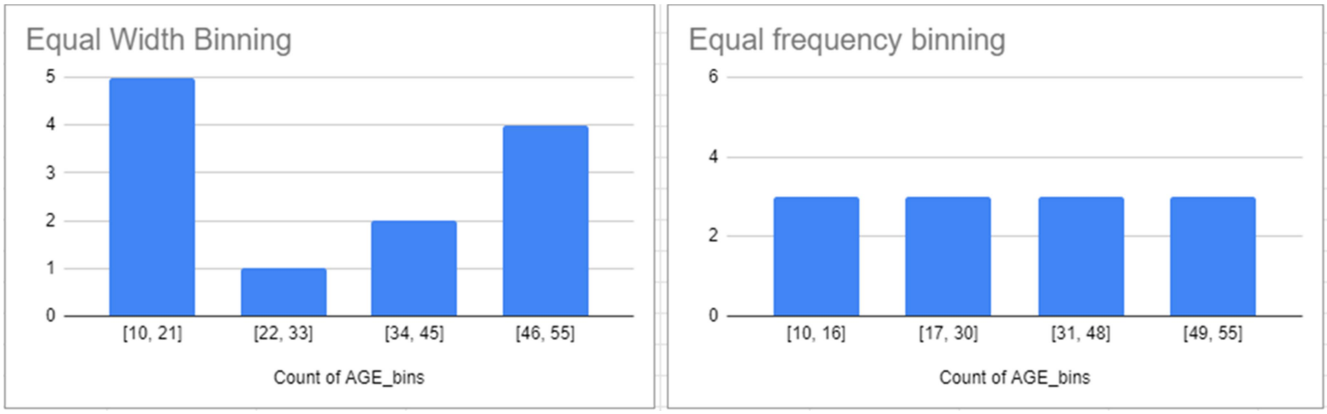


Рис.3.3. Приклад реалізації бінінгу ознак

(https://towardsdatascience.com/feature-engineering-deep-dive-into-en)

Index	Animal
0	Dog
1	Cat
2	Sheep
3	Horse
4	Lion

One-Hot code

Index	Dog	Cat	Sheep	Lion	Horse
0	1	0	0	0	0
1	0	1	0	0	0
2	0	0	1	0	0
3	0	0	0	0	1
4	0	0	0	1	0

Рис. 3.4. Приклад кодування категоріальних даних

(https://cdn.analyticsvidhya.com/wp-content/uploads/2020/08/Screenshot-from-2020-08-12-17-16-03.png)

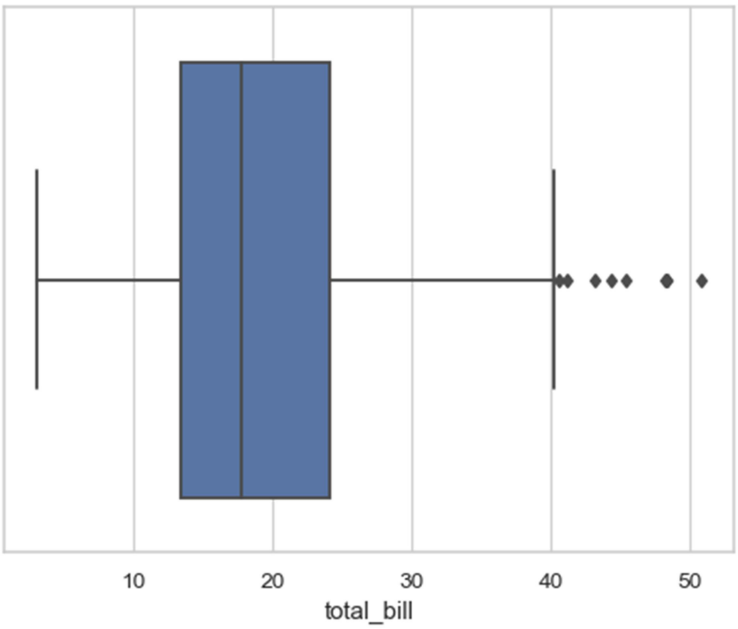


Рис. 3.5. Виявлення викидів у даних
(https://seaborn.pydata.org/generated/seaborn.boxplot.html)

	Attack	Defense	Attack^2	Attack x Defense	Defense^2
0	49.0	49.0	2401.0	2401.0	2401.0
1	62.0	63.0	3844.0	3906.0	3969.0
2	82.0	83.0	6724.0	6806.0	6889.0
3	100.0	123.0	10000.0	12300.0	15129.0
4	52.0	43.0	2704.0	2236.0	1849.0

Рис. 3.6. Приклад поліноміальної взаємодії ознак

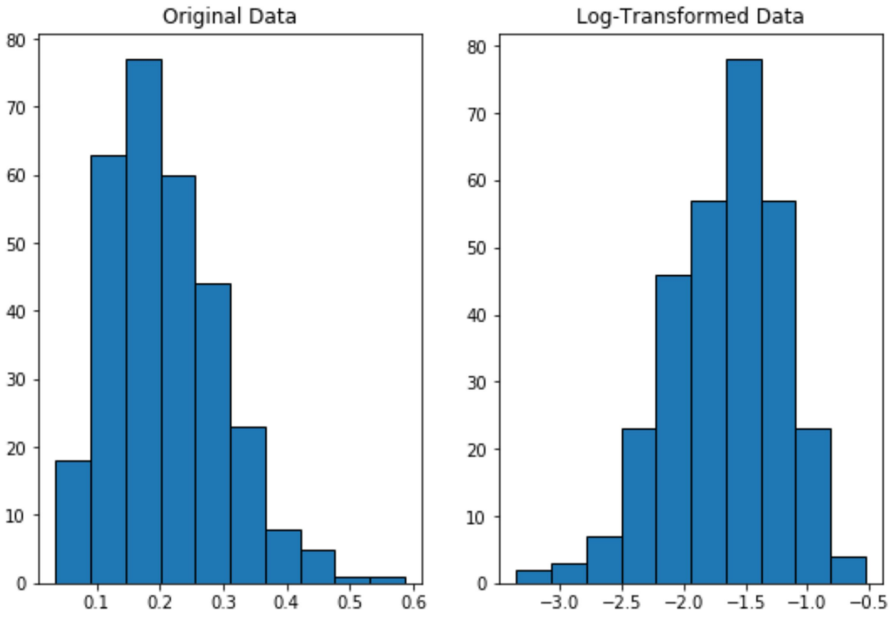


Рис. 3.7. Приклад логарифмічного перетворення даних

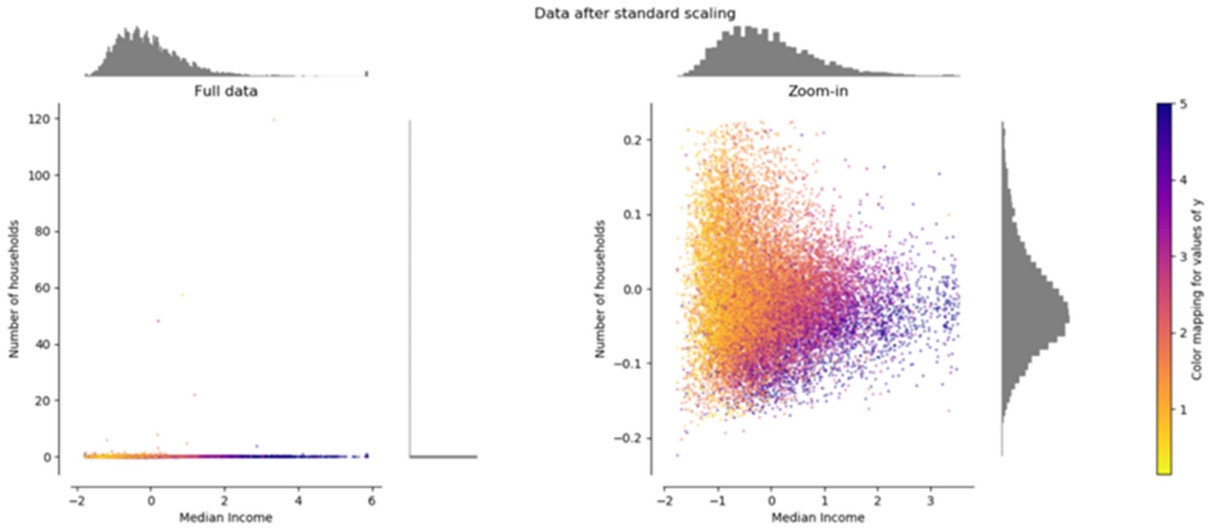


Рис. 3.8. Приклад масштабування даних

(https://miro.medium.com/max/3516/1*UPLv3kNw9JTtNabr70dQDQ.png)

Методи масштабування ознак у бібліотеці scikit-learn [3]:

StandardScaler в scikit-learn гарантує, що для кожної ознаки середнє дорівнюватиме 0, а дисперсія буде дорівнювати 1, в результаті чого всі ознаки матимуть один і той самий масштаб. Однак це масштабування не гарантує отримання якихось конкретних мінімальних і максимальних значень ознак.

RobustScaler аналогічний StandardScaler в тому плані, що в результаті його застосування ознаки матимуть один і той же масштаб. Однак RobustScaler замість середнього та дисперсії використовує медіану і квартилі. Це дозволяє RobustScaler ігнорувати точки даних, які сильно відрізняються від інших (наприклад, помилки вимірювань), які ще називаються викидами (outliers) і можуть стати проблемою для інших методів масштабування.

MinMaxScaler перетворює дані таким чином, щоб всі ознаки знаходилися строго в діапазоні від 0 до 1. Для двовимірного набору даних це означає, що всі дані містяться в прямокутнику, утвореному віссю x з діапазоном значень від 0 і 1 і віссю y з діапазоном значень від 0 і 1.

Normalizer масштабує кожну точку даних таким чином, щоб вектор ознак мав евклидову довжину 1. Іншими словами, він проектує точку даних на коло з радіусом 1 (або сферу в разі великого числа вимірювань). Вектор множиться на

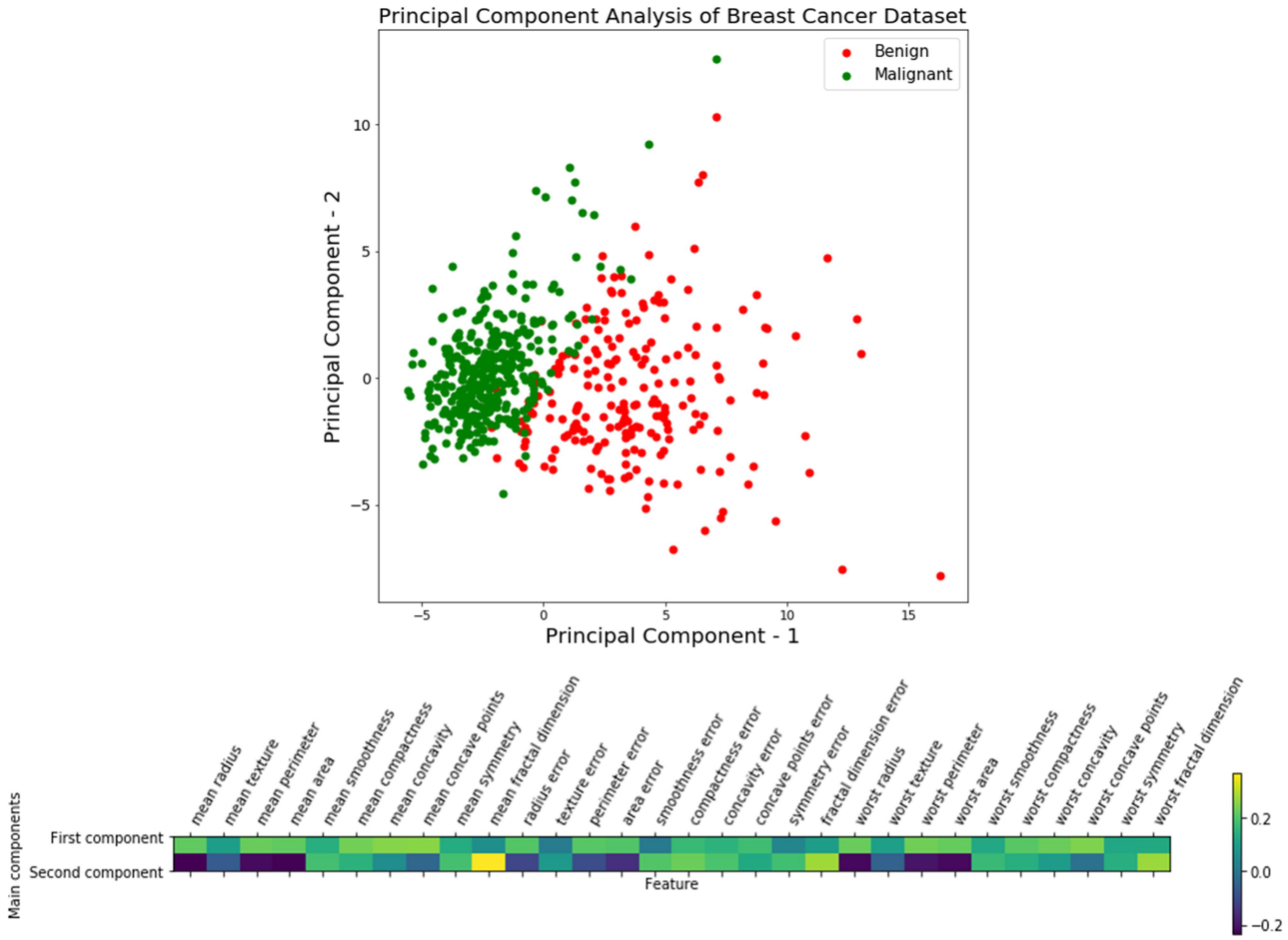
інверсію своєї довжини. Подібна нормалізація використовується тоді, коли важливим є напрямок (але не довжина) вектора ознак.

Викид — це точка даних, віддалена від інших подібних точок. Викиди можуть бути пов'язані з мінливістю вимірювань або можуть вказувати на експериментальні помилки або помилки в зборі даних. Якщо можливо, викиди слід виключити з набору даних. Викиди даних можуть зіпсувати та ввести в оману процес навчання, що призведе до збільшення часу навчання, менш точних моделей і, зрештою, до неточних результатів [1-6].

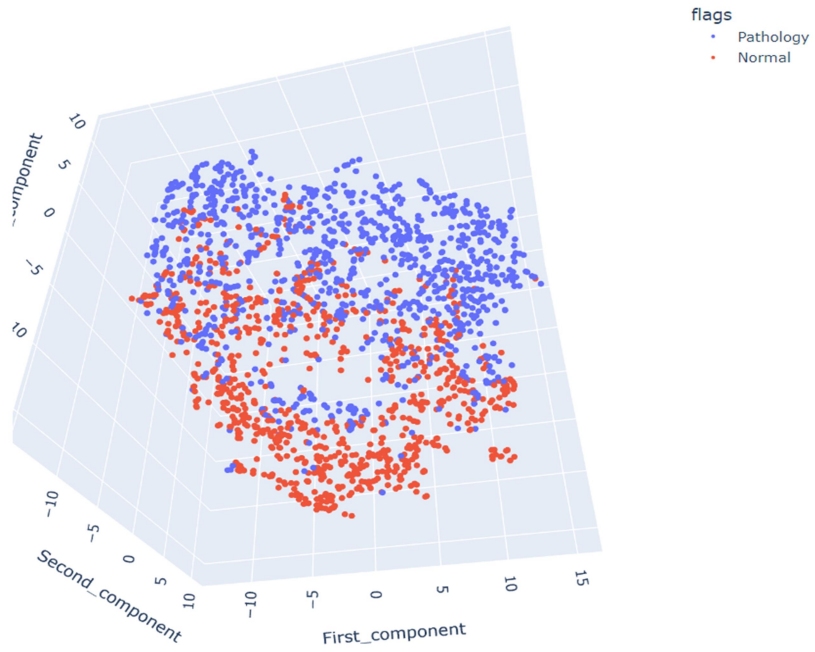
Робоче завдання

1. Ознайомитись з методами попередньої обробки даних, розрахунку ознак, вибору ознак та зменшення розмірності ознак.

2. Для даних з обраної бази визначити, які методи обробки даних треба застосувати (наприклад, прибрати викиди в даних, закодувати категоріальні змінні, замінити або видалити пропущені дані, та ін.) Включити в протокол стисле обґрунтування обраних методів.
3. Реалізувати обрані методи обробки даних. Навести приклади застосування, проілюструвати роботу алгоритмів.
4. Обрати необхідні методи розрахунку ознак (наприклад, спектральний аналіз, спектрально-часовий розклад, імовірнісні характеристики, та їх комбінації), виходячи із особливостей даних та специфіки задачі. Навести в протоколі стисле обґрунтування обраних методів.
5. Реалізувати обрані методи розрахунку ознак. Навести приклади застосування, проілюструвати роботу алгоритмів.
6. Обрати необхідні методи вибору ознак (наприклад, кореляційний метод, рекурсивне видалення, та ін.) та зменшення розмірності ознак (наприклад, PCA, ICA), виходячи із особливостей даних та специфіки задачі (рис.3.9-3.10). Навести в протоколі стисле обґрунтування обраних методів.



а)



б)

Рис.3.9. Приклади застосування аналізу головних компонент (PCA): а) задача класифікації пухлин молочної залози на доброякісні та злоякісні; б) задача виявлення ішемії міокарда

7. Реалізувати обрані методи вибору ознак та зменшення розмірності ознак. Навести приклади застосування, проілюструвати роботу алгоритмів.

8. Зробити висновки щодо застосовності обраних методів та результативності їх застосування для обраної бази даних.

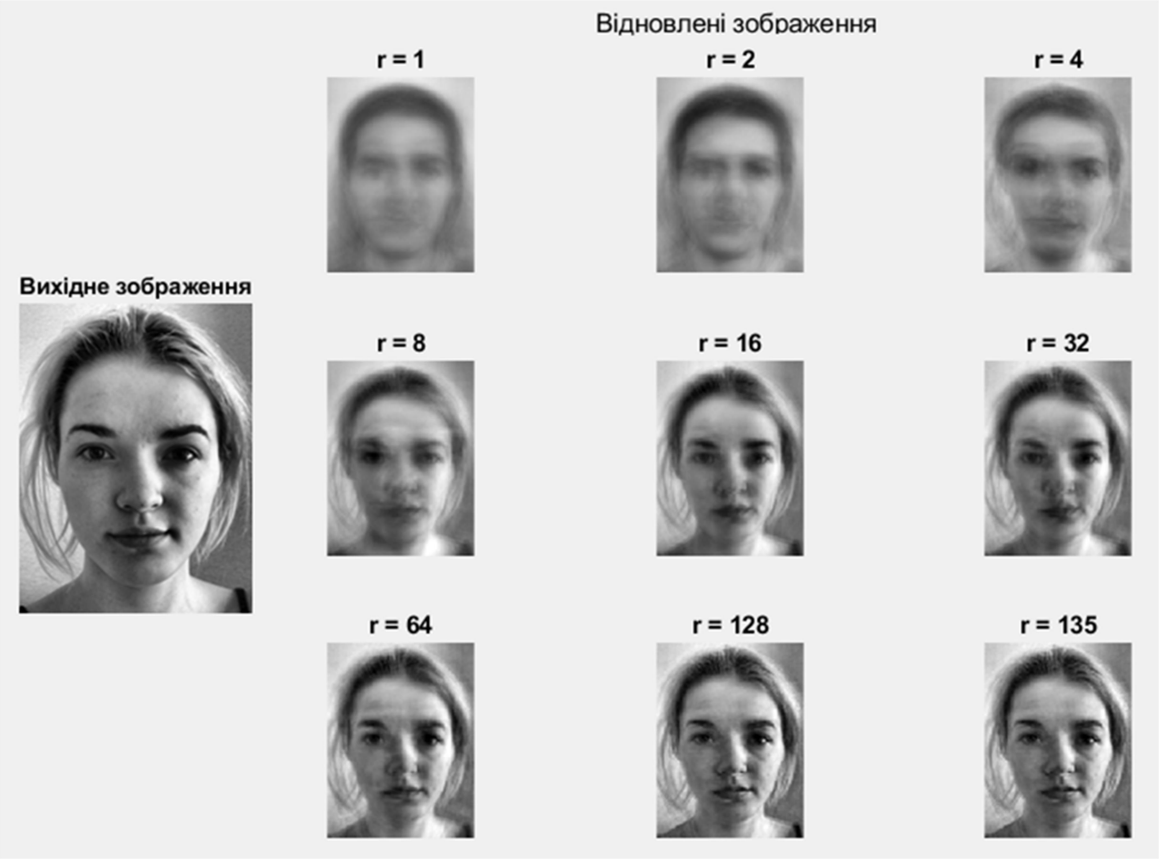


Рис.3.10. Реконструкція зображень особи за допомогою поступового збільшення кількості головних компонент

Контрольні питання:

1. Які методи попередньої обробки ознак ви знаєте?
2. Для чого використовується логарифмічне перетворення даних?
3. Для чого використовується масштабування даних?
4. Як використовувати категоріальні дані у машинному навчанні?
5. Які методи розрахунку ознак було реалізовано для аналізованої вами бази даних?
6. В чому полягає метод PCA? Як його можна застосувати для задач машинного навчання?

Лабораторна робота №4

Реалізація методів машинного навчання без вчителя

Мета роботи: ознайомлення та реалізація методів машинного навчання без вчителя.

Загальні теоретичні відомості

Машинне навчання без вчителя включає в себе всі види машинного навчання, коли відповідь невідома і відсутній учитель, який вказує відповідь алгоритму [1-6]. У машинному навчанні без учителя є лише вхідні дані і алгоритму необхідно витягти знання з цих даних. Головна проблема машинного навчання без учителя - оцінка корисності інформації, отриманої алгоритмом. Алгоритми машинного навчання без вчителя, як правило, застосовуються до даних, які не містять ніяких міток, таким чином, відомо, якою повинна бути правильна відповідь. Тому дуже важко судити про якість роботи моделі. Як наслідок, алгоритми машинного навчання без вчителя часто використовуються в розвідувальних цілях, коли фахівець хоче краще вивчити самі дані.

Алгоритми кластеризації (clustering algorithms) розбивають дані на окремі групи схожих між собою елементів, які називаються кластерами (рис.4.1-4.4). Мета -розділити дані таким чином, щоб точки, що знаходять в одному і тому ж кластері, були дуже схожі один з одним, а точки, що знаходяться в різних кластерах, відрізнялися одна від одної. Як і алгоритми класифікації, алгоритми кластеризації прогнозують кожній точці даних номер кластера, якому вона належить.

Кластеризація k-середніх - один з найпростіших алгоритмів кластеризації. Спочатку вибирається число кластерів k. Після вибору значення k алгоритм k-середніх відбирає точки, які будуть представляти центри кластерів (cluster centers). Потім для кожної точки даних обчислюється його евклідова відстань до кожного центру кластера. Кожна точка призначається найближчому центру кластера. Алгоритм обчислює центроїди (*centroids*) – центри важкості кластерів. Кожен центр ваги - це вектор, елементи якого являють собою середні значення характеристик, обчислені по всіх точках кластера. Центр кластера зміщується в

його центр ваги. Точки заново призначаються найближчому центру кластера. Етапи зміни центрів кластерів і перепризначення точок ітеративно повторюються до тих пір, поки границі кластерів і розташування центроїдів не перестануть змінюватися, тобто на кожній ітерації в кожен кластер будуть потрапляти одні і ті ж точки даних. Навіть якщо є відомою «правильна» кількість кластерів для конкретного набору даних, алгоритм k-середніх не завжди може виділити їх. Кожен кластер визначається виключно його центром, це означає, що кожен кластер має опуклу форму. В результаті цього алгоритм k-середніх може описати відносно прості форми. Крім того, алгоритм k-середніх передбачає, що всі кластери в певному сенсі мають однаковий «діаметр», він завжди проводить границю між кластерами так, щоб вона проходила точно посередині між центрами кластерів.

Агломеративна кластеризація (agglomerative clustering) відноситься до сімейства алгоритмів кластеризації, в основі яких лежать однакові принципи: алгоритм починає свою роботу з того, що кожну точку даних заносить в свій власний кластер і в міру виконання умов об'єднує два найбільш схожих між собою кластера до тих пір, поки не буде задоволено певний критерій зупинки. Критерій зупинки, реалізований в scikit-learn - це кількість кластерів, тому схожі між собою кластери об'єднуються до тих пір, поки не залишиться задана кількість кластерів. Є кілька критеріїв зв'язку (linkage), які задають точний спосіб вимірювання «найбільшої схожості кластерів». В основі цих критеріїв лежить відстань між двома існуючими кластерами.

У scikit-learn реалізовані наступні три критерії:

Ward - метод за замовчуванням (метод Варда) вибирає і об'єднує два кластера так, щоб приріст дисперсії всередині кластерів був мінімальним. Часто цей критерій призводить до отримання кластерів однакового розміру.

Average - метод average (метод середнього зв'язку) об'єднує два кластери, які мають найменше середнє значення всіх відстаней, виміряних між точками двох кластерів.

Complete - метод complete (метод повного зв'язку або метод максимальної зв'язку) об'єднує два кластери, які мають найменшу відстань між двома їх найбільш віддаленими точками.

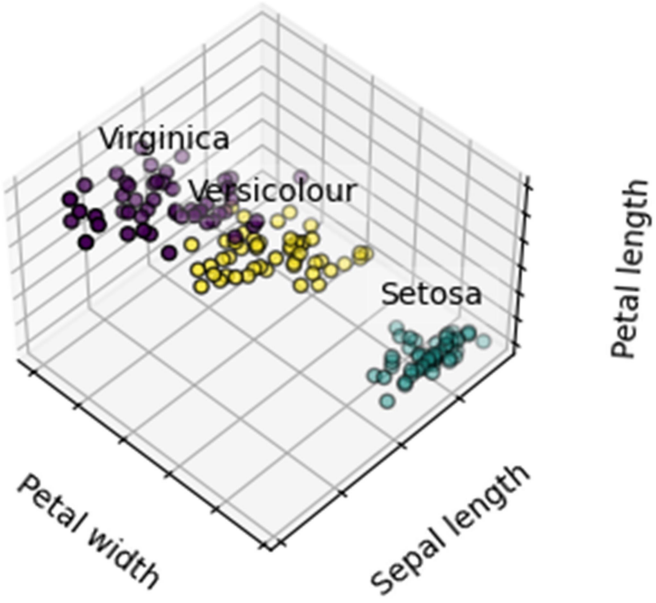


Рис.4.1. Приклад машинного навчання без вчителя за допомогою кластерізації за методом k-means (https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_iris.html#sphx-glr-auto-examples-cluster-plot-cluster-iris-py)

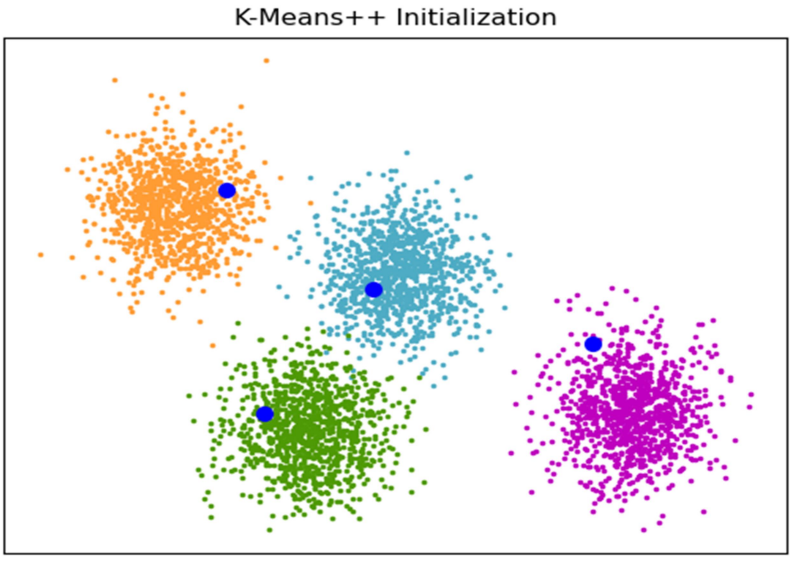


Рис.4.2. Приклад початкової ініціалізації центрів кластерів за методом k-means (https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_plusplus.html#sphx-glr-auto-examples-cluster-plot-kmeans-plusplus-py)

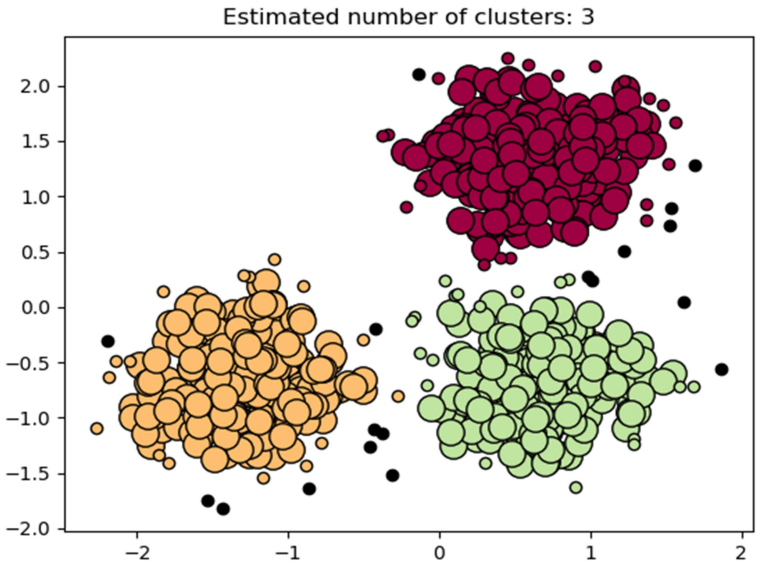


Рис.4.3. Приклад реалізації DBSCAN алгоритму (https://scikit-learn.org/stable/auto_examples/cluster/plot_dbscan.html#sphx-glr-auto-examples-cluster-plot-dbscan-py)

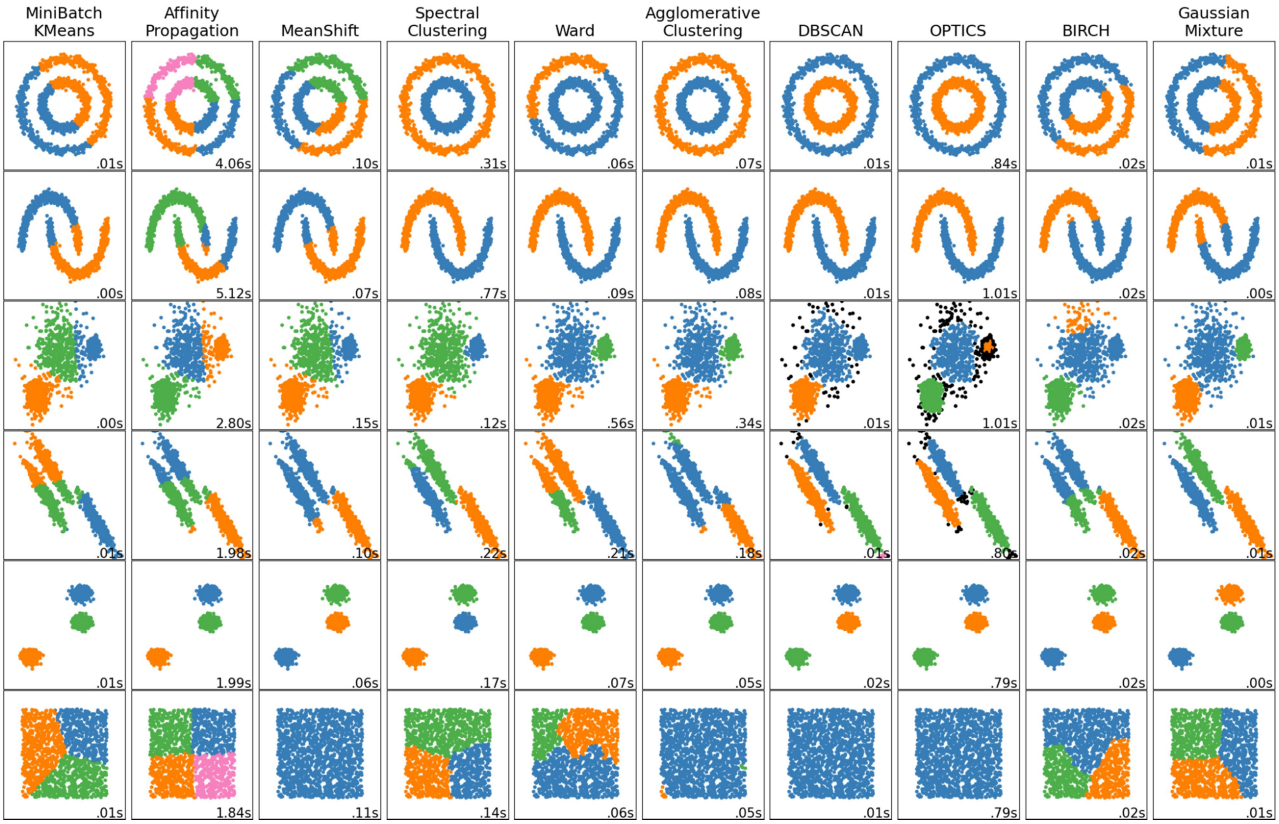


Рис.4.4. Приклад порівняння роботи різних алгоритмів кластеризації (https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html#sphx-glr-auto-examples-cluster-plot-cluster-comparison-py)

Робоче завдання

1. Виконати кластеризація методом К-середніх для випадку трьох та восьми кластерів за прикладом. Зробити висновки щодо впливу кількості кластерів та кількості проходів алгоритму з різними початковими центрами кластерів на результати кластеризації.
2. Дослідити вплив статистичних характеристик розподілу ознак на результати кластеризації за прикладом. Зробити висновки.
3. Вирішити задачу color quantization методом К-середніх за прикладом, обравши різну кількість кольорів (наприклад 2, 8, 16, 64).
4. Порівняти різні стратегії ініціалізації кількості кластерів по точності при кластеризації зображень рукописних цифр за прикладом. Зробити висновки.

Контрольні питання:

1. Як визначається та на що впливає кількість кластерів для методу К-середніх?
2. В чому полягає метод кластеризації К-середніх?
3. Які міри можуть використовуватися у методі кластеризації К-середніх?
4. Який показник якості кластеризації мінімізується у методі К-середніх?
5. Які інші методи кластеризації ви знаєте окрім методу К-середніх?

Лабораторна робота №5

Реалізація методів машинного навчання з вчителем

Мета роботи: ознайомлення та реалізація методів машинного навчання з вчителем, таких як регресійний аналіз, метод k найближчих сусідів, дискримінантний аналіз, метод опорних векторів, наївний байєсівський класифікатор, дерева рішень та ансамблі дерев рішень.

Загальні теоретичні відомості

Бібліотека **Scikit-learn** використовується для вирішення завдань класичного машинного навчання. Scikit-learn працює на основі декількох поширених математичних бібліотек і інтегрує їх одна з одною. Для своєї роботи scikit-learn використовує такі бібліотеки як **NumPy** (математичні операції), **SciPy** (науково-технічні обчислення), **Matplotlib** (візуалізація даних), **SymPy** (символьна математика), **Pandas** (обробка та аналіз даних).

Scikit-learn спеціалізується на алгоритмах машинного навчання для вирішення завдань навчання з учителем [1-6]: **класифікації** (прогноз ознаки, множина допустимих значень якого обмежена) і **регресії** (прогноз ознаки з речовими значеннями), а також для задач навчання без учителя: **кластеризації** (розбиття даних по класах, які модель визначить сама), **зниження розмірності** (подання даних в просторі меншої розмірності з мінімальними втратами корисної інформації) і детектування аномалій.

Бібліотека Scikit-learn реалізує наступні основні методи для машинного навчання з вчителем:

Лінійні методи: моделі, завдання яких побудувати розділяючу (для класифікації) або апроксимуючу (для регресії) гіперплощину. Двома найбільш поширеними алгоритмами лінійної класифікації є ***логістична регресія (logistic regression)***, реалізована в класі **linear_model.LogisticRegression**, і лінійний метод опорних векторів (linear support vector machines) або ***лінійний SVM***, реалізований в класі **svm.LinearSVC** (SVC розшифровується як support vector classifier - класифікатор опорних векторів). Незважаючи на назву, логістична регресія є

алгоритмом класифікації, а не алгоритмом регресії, і його не слід плутати з лінійної регресією.

Метричні: моделі, які обчислюють відстань по одній з метрик між об'єктами вибірки, і приймають рішення в залежності від цієї відстані (наприклад, метод K найближчих сусідів).

Дерева рішень (рис.5.1)**:** навчання моделей, що базуються на безлічі умов, оптимально обраних для вирішення завдання. Дерева рішень є моделями, які широко використовуються для вирішення завдань класифікації і регресії. По суті вони задають питання і вибудовують ієрархію правил «якщо ... то», яка найкраще приводить до рішення. Щоб побудувати дерево, алгоритм перебирає всі можливі тести і знаходить той, який є найбільш інформативним з точки зору прогнозування значень цільової змінної. Цей рекурсивний процес будує в результаті бінарне дерево рішень, в якому кожен вузол відповідає певному критерію. Це дозволяє скласти уявлення про алгоритм як спосіб побудувати ієрархію розбиття. Оскільки кожен критерій розглядає тільки одну ознаку, області, що виходять в результаті розбиття, завжди мають границі, паралельні осям. Рекурсивне розбиття даних повторюється до тих пір, поки всі точки даних в кожній області розбиття (кожному листі дерева рішень) не належатимуть одному і тому ж значенню цільової змінної (класу або кількісному значенню). Лист дерева, який містить точки даних, що відносяться до одного і того ж значення цільової змінної, називається чистим (pure).

Ансамблеві методи: методи, засновані на деревах рішень, які комбінують міць множини дерев, і таким чином підвищують їх якість роботи, а також дозволяють проводити відбір ознак (бустінг, беггінг, випадковий ліс, мажоритарне голосування). Ансамблі (ensembles) - це методи, які поєднують в собі набір моделей машинного навчання, щоб у підсумку отримати більш потужну модель. Існує багато моделей машинного навчання, які належать до цієї категорії, але є дві ансамблеві моделі, які довели свою ефективність на різних наборах даних для задач класифікації і регресії, обидві використовують дерева рішень в якості будівельних блоків: **випадковий ліс дерев рішень** і **градієнтний бустінг дерев рішень**. Основним недоліком дерев рішень є їх схильність до

перенавчання. **Випадковий ліс** є одним із способів вирішення цієї проблеми. По суті випадковий ліс - це набір дерев рішень, де кожне дерево трохи відрізняється від інших. Ідея випадкового лісу полягає в тому, що кожне дерево може досить добре прогнозувати, але швидше за все перенавчати на частині даних. Якщо побудувати багато дерев, які добре працюють і перенавчають з різним ступенем, можна зменшити перенавчання шляхом усереднення їх результатів.

Для реалізації такої стратегії потрібно побудувати велику кількість дерев рішень. Кожне дерево має на прийнятному рівні прогнозувати цільову змінну і має відрізнятися від інших дерев. Випадкові ліс отримав свою назву через те, що в процесі побудови дерев внесена випадковість, покликана забезпечити унікальність кожного дерева. Існує дві техніки, що дозволяють отримати рандомізовані дерева в рамках випадкового лісу: спочатку вибираються точки даних (спостереження), які будуть використовуватися для побудови дерева, а потім відбираються ознаки в кожному розбитті. Для побудови моделі випадкових лісів необхідно визначитися з кількістю дерев (параметр **n_estimators** для RandomForestRegressor або RandomForestClassifier). Ці дерева будуть побудовані абсолютно незалежно один від одного, і алгоритм буде випадковим чином відбирати ознаки для побудови кожного дерева, щоб отримати не схожі один на одного дерева. Щоб дати прогноз для випадкового лісу, алгоритм спочатку дає прогноз для кожного дерева в лісі. Для регресії можна усереднити ці результати, щоб отримати остаточний прогноз. Для класифікації використовується стратегія «м'якого голосування». Це означає, що кожен алгоритм дає «м'який» прогноз, обчислюючи ймовірності для кожного класу. Ці ймовірності усереднюються по всьому деревам і прогнозується клас з найбільшою ймовірністю.

Градієнтний бустинг дерев рішень – ще один ансамблевий метод, який об'єднує в собі безліч дерев для створення більш потужної моделі. Незважаючи на слово «регресія» в назві, ці моделі можна використовувати для регресії і класифікації. На відміну від випадкового лісу, градієнтний бустинг будує послідовність дерев, в якій кожне дерево намагається виправити помилки попереднього. За замовчуванням в градієнтному бустингу дерев регресії відсутня випадковість, замість цього використовується сувора попередня обрізка. У

градієнтному бустингу дерев часто використовуються дерева невеликої глибини, від одного до п'яти рівнів, що робить модель менше з точки зору пам'яті і прискорює обчислення прогнозів. Основна ідея градієнтного бустинга полягає в об'єднанні безлічі простих моделей, дерев невеликої глибини. Кожне дерево може дати хороші прогнози тільки для частини даних і таким чином для ітеративного поліпшення якості додається все більша кількість дерев. На відміну від випадкового лісу він, як правило, трохи більше чутливий до налаштування параметрів, однак при правильно заданих параметрах може дати більш високе значення правильності. Крім попереднього обрізання і числа дерев в ансамблі, ще один важливий параметр градієнтного бустинга - це **learning_rate**, який контролює, наскільки сильно кожне дерево буде намагатися виправити помилки попередніх дерев. Більш висока швидкість навчання означає, що кожне дерево може внести більш сильні коригування і це дозволяє отримати більш складну модель. Додавання більшої кількості дерев в ансамбль, здійснюване за рахунок збільшення значення **n_estimators**, також збільшує складність моделі, оскільки модель має більше шансів виправити помилки на навчальному наборі.

Нейронні мережі: комплексний нелінійний метод для задач регресії і класифікації.

SVM: нелінійний метод, який навчається визначати границі прийняття рішень. Ядерний метод опорних векторів (SVM) - це розширення методу опорних векторів, воно дозволяє отримувати більш складні моделі, які не зводяться до побудови простих гіперплощин в просторі. В ході навчання SVM обчислює важливість кожної точки навчальних даних з точки зору визначення границі прийняття рішення між двома класами. Зазвичай лише частина точок навчального набору важлива для визначення границі прийняття рішень: точки, які лежать на границі між класами. Вони називаються опорними векторамі (support vectors) і дали свою назву машині опорних векторів SVM. Щоб отримати прогноз для нової точки, вимірюється відстань до кожного опорного вектора. Класифікаційне рішення приймається, виходячи з відстаней до опорних векторів, а також важливості опорних векторів, отриманих в процесі навчання (зберігаються в атрибуті **dual_coef_** класу SVC). SVM вимагає ретельної попередньої обробки

даних і налаштування параметрів. Загальнопоширений метод масштабування для ядерного SVM полягає в масштабуванні даних так, щоб всі ознаки приймали значення від 0 до 1. Важливими параметрами ядерного SVM є параметр регуляризації C, тип ядра, а також параметри, які визначаються ядром.

Наївний Байєсівський метод: пряме ймовірнісне моделювання для задач класифікації. Наївні байєсівські класифікатори представляють собою сімейство класифікаторів, які вони оцінюють параметри, розглядаючи кожну ознаку окремо і за кожною ознакою збирають прості статистики класів. У scikit-learn реалізовані три види наївних байєсівських класифікаторів: **GaussianNB**, **BernoulliNB** і **MultinomialNB**. BernoulliNB і MultinomialNB в основному використовуються для класифікації текстових даних.

Крос-валідація: метод, при якому для навчання використовується весь набір даних (на відміну від розбиття на вибірки train/test), проте навчання відбувається багаторазово, і в якості валідаційної вибірки на кожному кроці виступають різні частини датасета. Підсумковий результат являє собою усереднення отриманих результатів.

Grid Search: метод для знаходження оптимальних гіперпараметрів моделі шляхом побудови сітки з значень гіперпараметрів і послідовного навчання моделей з усіма можливими комбінаціями гіперпараметрів з сітки.

Крім цього, Scikit-learn містить функції для розрахунку значень метрик, вибору моделей, препроцесінга даних та інші (рис.5.2-5.5).

Робоче завдання

1. Розділити підготовлений у попередній лабораторній роботі набір даних на дві частини: навчальний набір даних і тестовий набір даних (80% та 20%). Навчальний набір даних використовувати для навчання моделей.
2. Порівняти роботу різних алгоритмів з варіацією налаштувань їх параметрів, серед яких:

- регресія (логістична, лінійна, гребенева, лассо),
- дискримінантний аналіз,
- метод k-найближчих сусідів (1, 5, 10 найближчих сусідів),

- ядерний метод опорних векторів (ядра поліноміальне, гаусівське, сигмоїдальне),
- дерева рішень (глибина дерева 4, 7, 10),
- ансамблі дерев рішень (випадковий ліс - 10, 20, 30 дерев у ансамблі),
- градієнтний бустінг дерев рішень (також додатково XGBoost),
- наївний Байєсівський класифікатор,
- стекінг.

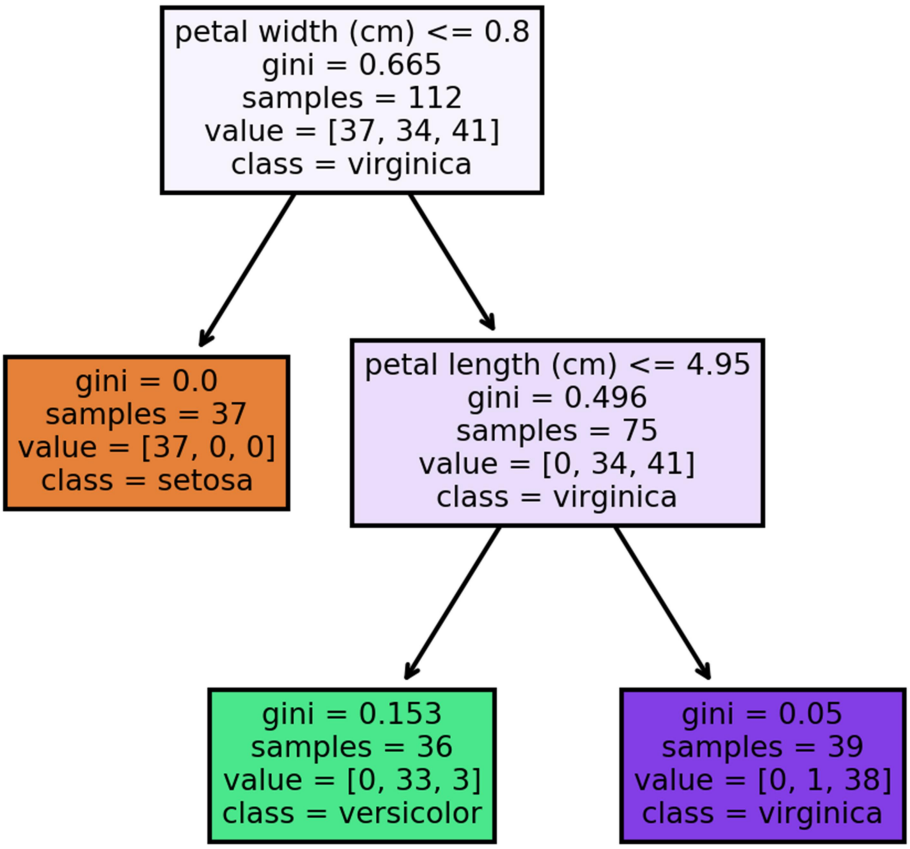


Рис.5.1. Приклад візуалізації дерева рішень

(https://programming.vip/docs/python-visual-decision-tree-matplotlib-graphviz.html)

Наприклад, деякі з методів класифікації та регресії можна імпортувати як

```
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeRegressor
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.svm import LinearSVC
```

```
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.linear_model import RidgeClassifierCV

from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import StackingClassifier
```

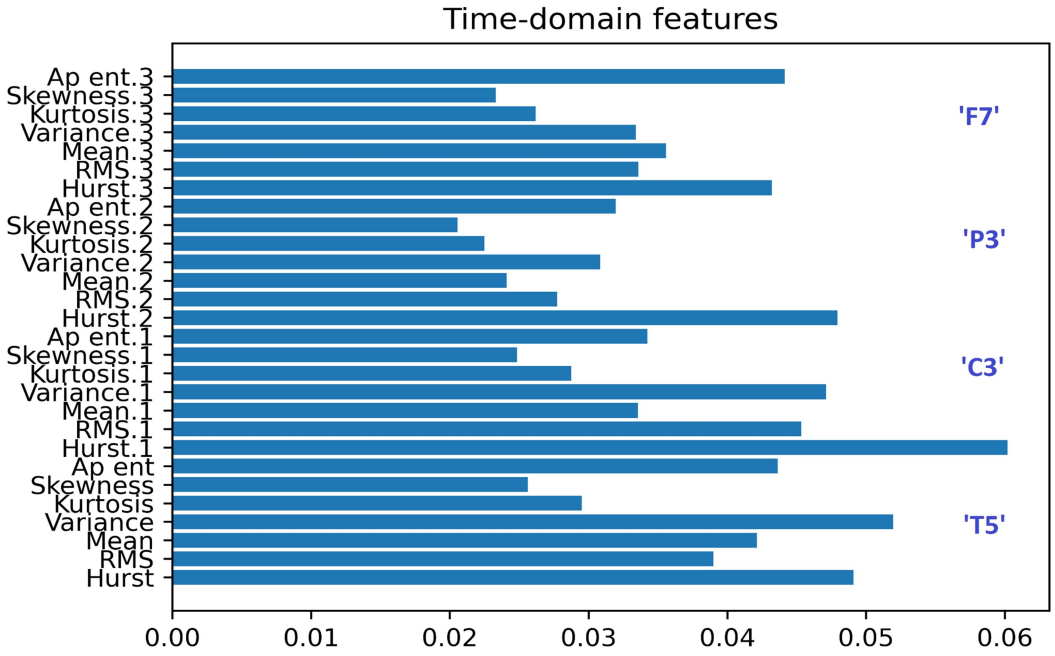


Рис.5.2. Оцінювання важливості ознак для Gradient Boosting Classifier (https://machinelearningmastery.com/calculate-feature-importance-with-python/)

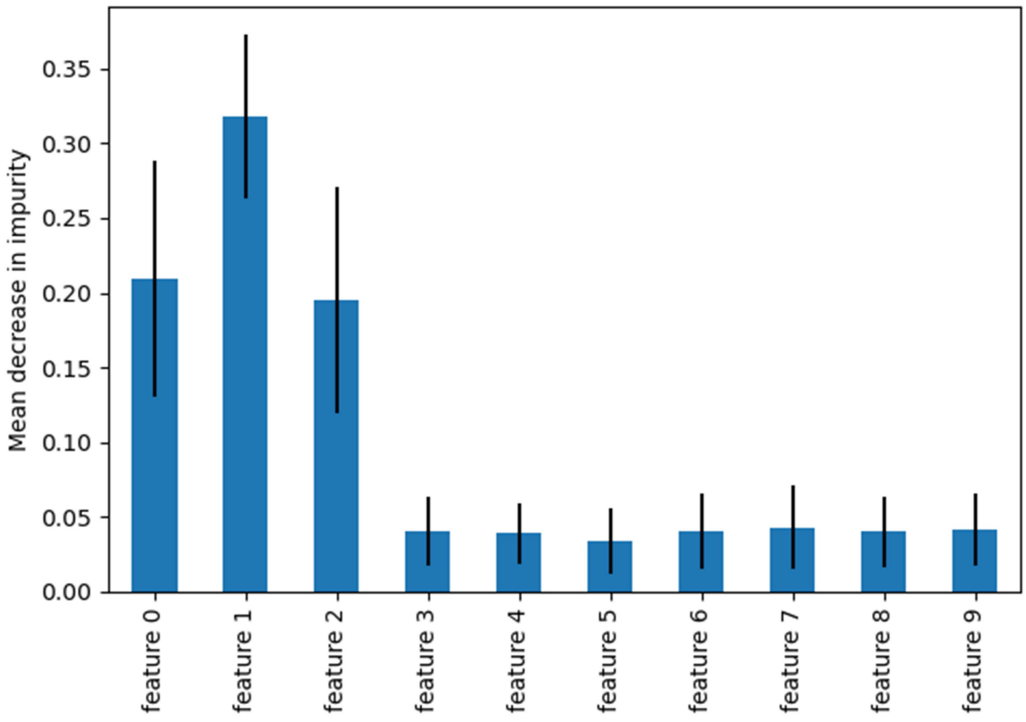


Рис.5.3. Оцінювання важливості ознак для дерев рішень (https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html)

3. Оцінити та навести у протоколі показники якості класифікації для використаних алгоритмів.

Наприклад, деякі з методів оцінювання якості класифікації можна імпортувати як

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
```

4. За попередніми результатами обрати найкращий класифікатор та далі застосувати для нього гратчастий пошук `GridSearchCV` для побудови сітки параметрів і розрахунку всіх можливих комбінацій.

Гратчастий пошук можна імпортувати як

```
from sklearn.model_selection import GridSearchCV
```

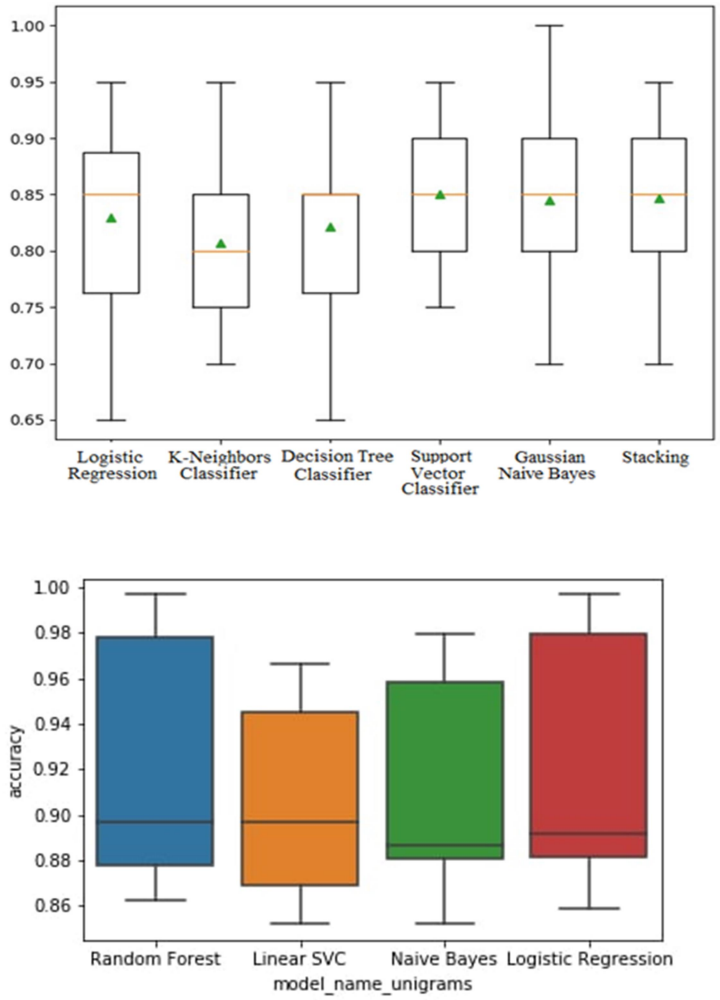


Рис. 5.4. Приклади порівняння точності класифікації для різних моделей, отриманої за допомогою перехресної перевірки та візуалізованої за допомогою boxplot (https://seaborn.pydata.org/generated/seaborn.boxplot.html)

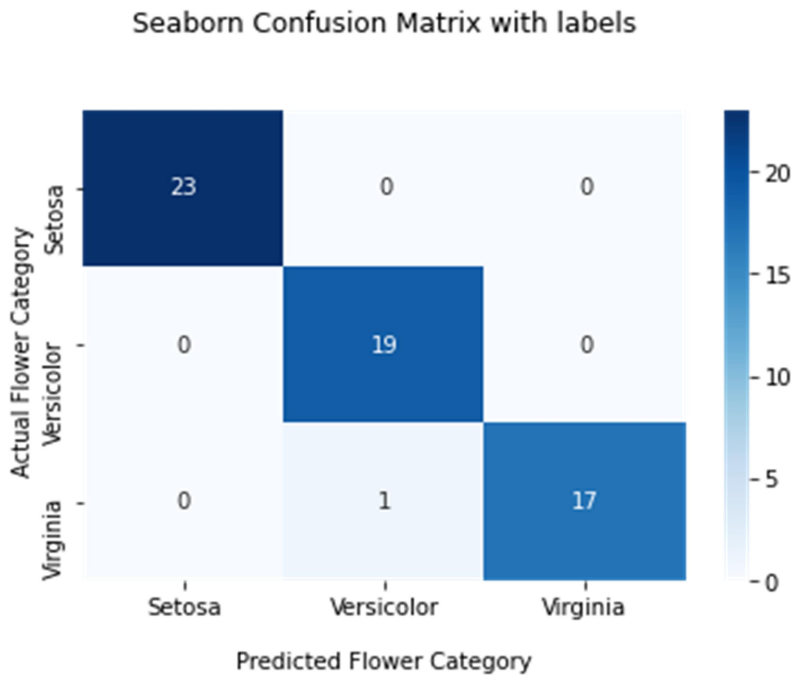


Рис. 5.5. Матриця невідповідності

(https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html)

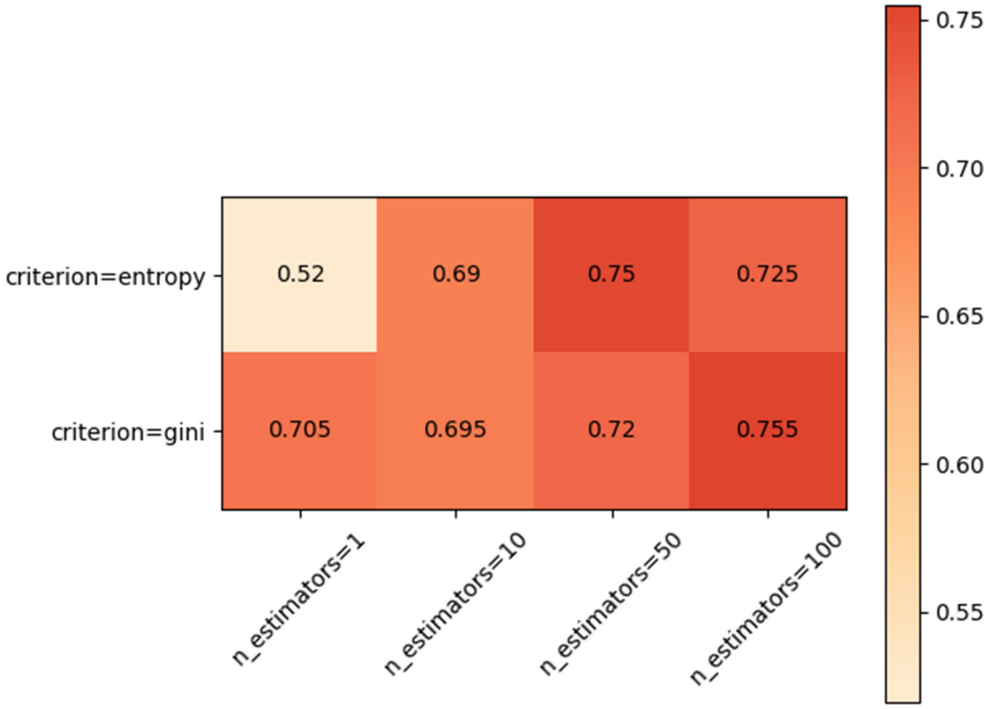


Рис. 5.7. Приклад візуалізації результатів гратчастого пошуку у вигляді heatmap

(https://stackoverflow.com/questions/68138679/gridsearchcv-results-heatmap)

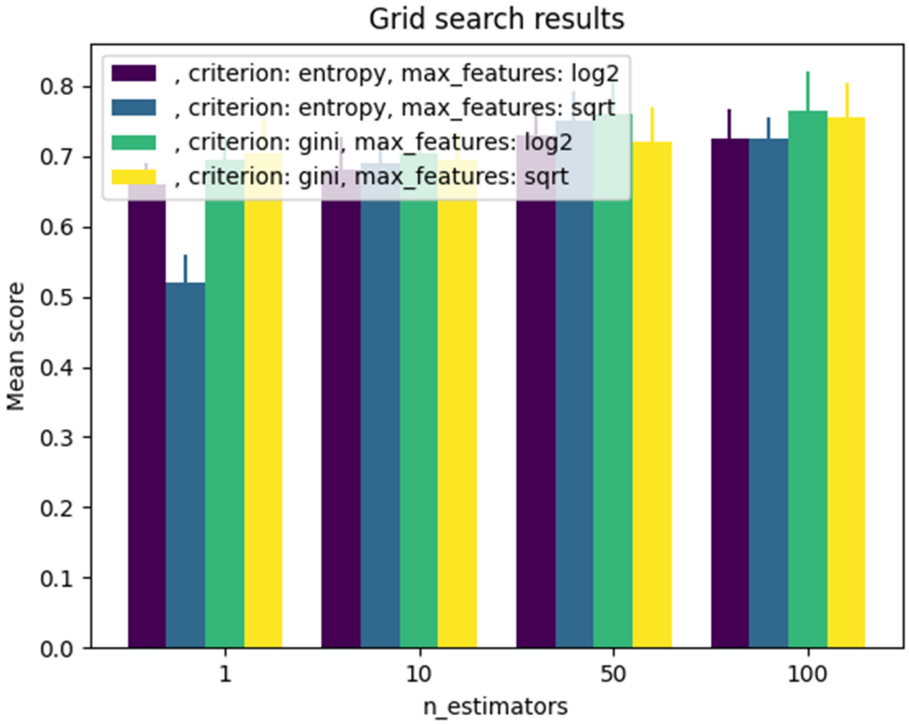


Рис. 5.7. Приклад візуалізації результатів гратчастого пошуку (https://sklearn-evaluation.readthedocs.io/en/stable/user_guide/grid_search.html)

5. Перевірити властивість **best_estimator_** та отримати найкращий набір параметрів.

6. Використати навчену модель для прогнозування нових даних (які не є частиною навчального набору) за допомогою **predict.**

7. Зробити висновки щодо результативності застосування розглянутих методів для досліджуваної бази даних.

Контрольні питання:

1. В чому полягає задача класифікації? Які ви знаєте методи класифікації?
2. Які основні методи машинного навчання з вчителем реалізує бібліотека Scikit-learn?
3. В чому полягає задача регресії?
4. Які основні методи для задачі регресії реалізує бібліотека Scikit-learn?
5. Що таке дерева рішень?

6. Що таке ансамблеві методи машинного навчання. Наведіть приклади.

7. В чому полягає метод опорних векторів? Як він реалізується за допомогою бібліотеки Scikit-learn?

8. Для чого в машинному навчанні застосовується метод граткового пошуку Grid Search з бібліотеки Scikit-learn?

9. Як можна оцінити важливість ознак для різних методів машинного навчання?

10. Як оцінити якість класифікації за допомогою бібліотеки Scikit-learn?

Лабораторна робота №6

Нейронні мережі

Мета роботи: набуття навичок реалізації штучних нейронних мереж за допомогою PyTorch на прикладі вирішення задачі класифікації зображень рукописних цифр.

Загальні відомості

Багатошарові персептрони (MLP) також називають простими нейронними мережами прямого поширення. MLP можна розглядати як узагальнення лінійних моделей, яке перш ніж прийти до прийняття рішення виконує кілька етапів обробки даних [1-6]. Згадаймо, що в лінійної регресії прогноз отримують за допомогою наступної формули:

$$\hat{y} = w[0] * x[0] + w[1] * x[1] + ... + w[p] * x[p] + b,$$

де $\hat{y}$ це зважена сума вхідних ознак $x[0] ... x[p]$. Вхідні ознаки зважені за обчисленими під час навчання коефіцієнтами $w[0] ... w[p]$. Можна представити їх графічно, як показано на рис.6.1.

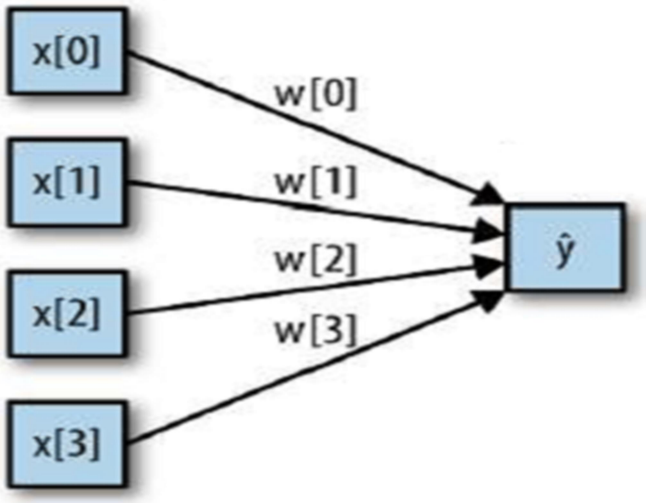


Рис.6.1. Візуалізація логістичної регресії, в якій вхідні ознаки і прогнози показані в вигляді вузлів, а коефіцієнти - у вигляді сполучень між вузлами

Тут кожен вузол, показаний зліва, є вхідною ознакою, з'єднувальні лінії - коефіцієнти, а вузол справа – це вихід, який є зваженою сумою входів. У MLP процес обчислення зважених сум повторюється кілька разів. Спочатку обчислюються приховані елементи (hidden units), які являють собою проміжний етап обробки. Вони знову об'єднуються за допомогою зважених сум для отримання кінцевого результату (рис.6.2).

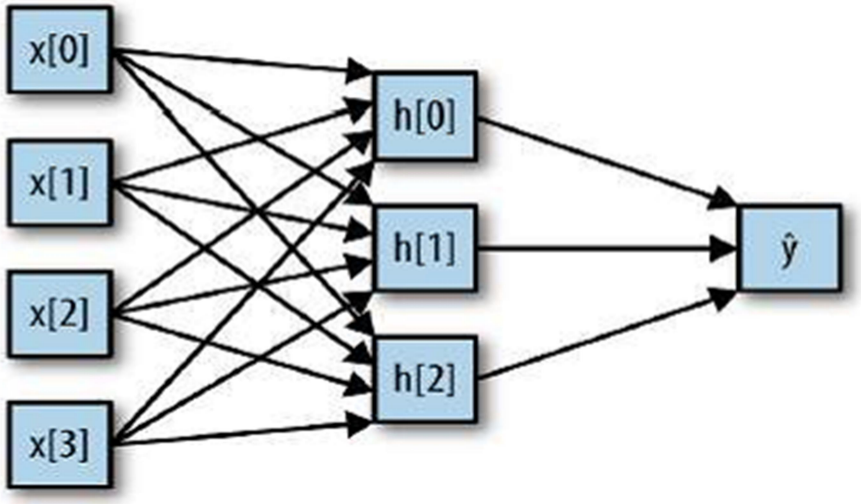


Рис.6.2. Ілюстрація багатошарового персептрона з одним прихованим шаром

У цій моделі набагато більше обчислюваних коефіцієнтів (які називаються вагами): коефіцієнт між кожним входом і кожним прихованим елементом (які утворюють прихований шар hiddenlayer) і коефіцієнт між кожним елементом прихованого шару і виходом (рис.6.3).

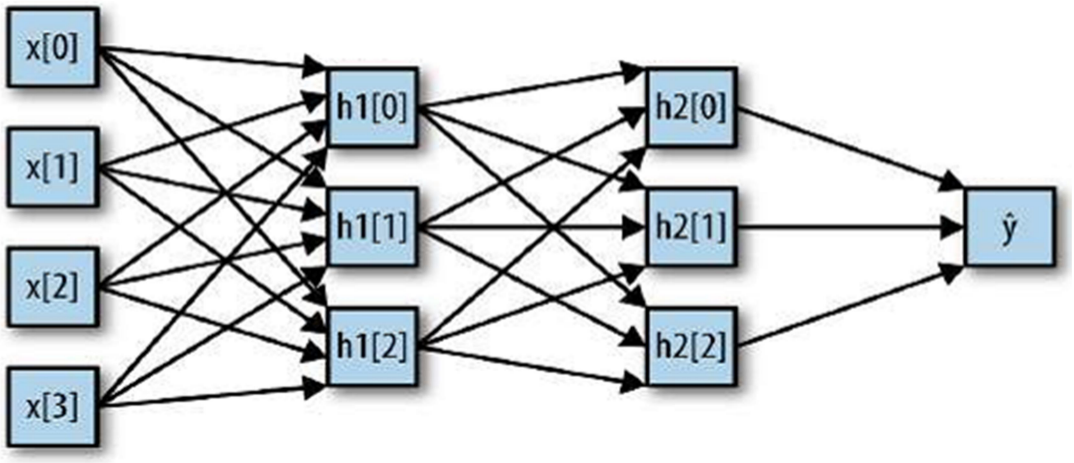


Рис.6.3. Багатошаровий персептрон з двома прихованими шарами

Коли відбувається обчислення зваженої суми входів для кожного елемента прихованого шару, до неї застосовується ***функція активації*** - зазвичай використовуються нелінійні функції випрямлений лінейний елемент (rectified linear unit або *relu*) або гіперболічний тангенс (hyperbolic tangent або *tanh*). Як результат отримуємо виходи нейронів прихованого шару. Ці проміжні виходи можуть вважатися нелінійними перетвореннями і комбінаціями первинних входів. Вони стають входами вихідного шару. Знову обчислюється зважена суму входів, застосовується функція активації і отримуються підсумкові значення цільової змінної. Функції активації relu і tanh показані на рис. 6.4. Relu відсікає значення нижче нуля, в той час як tanh приймає значення від -1 до 1 (відповідно для мінімального і максимального значень входів).

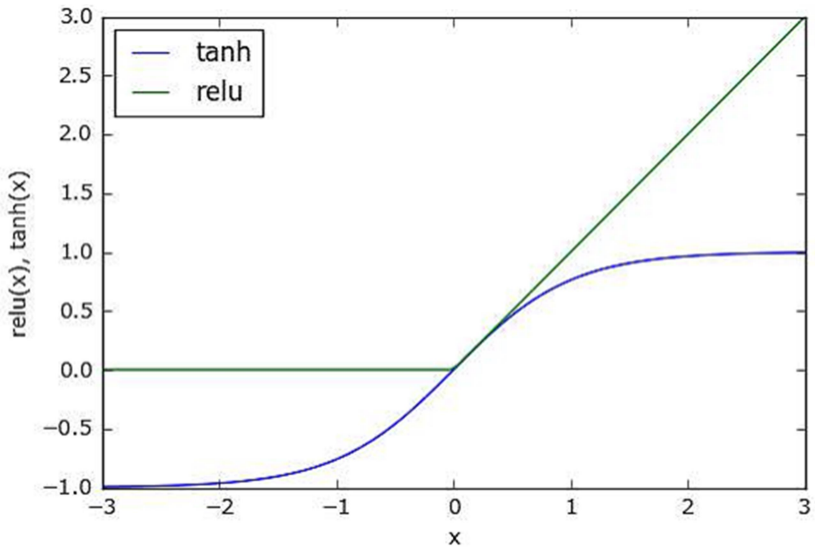


Рис.6.4_ Функція активації гіперболічний тангенс
і функція активації випрямленого лінійного елемента

$h[0] = \tanh(w[0,0] * x[0] + w[1,0] * x[1] + w[2,0] * x[2] + w[3,0] * x[3])$

$h[1] = \tanh(w[0,0] * x[0] + w[1,0] * x[1] + w[2,0] * x[2] + w[3,0] * x[3])$

$h[2] = \tanh(w[0,0] * x[0] + w[1,0] * x[1] + w[2,0] * x[2] + w[3,0] * x[3])$

$\hat{y} = v[0] * h[0] + v[1] * h[1] + v[2] * h[2]$

Тут w - ваги між входом x і прихованим шаром h, а v – вагові коефіцієнти між прихованим шаром h і виходом $y\hat{}$. Ваги v і w обчислюються за даними, x є вхідними ознаками, $y\hat{}$ - обчислений вихід, а h - проміжні обчислення.

Важливий параметр - кількість вузлів в прихованому шарі. Побудова великих нейронних мереж, що складаються з багатьох шарів обчислень, ввела термін «глибоке навчання» («**deep learning**»).

Важливою властивістю нейронних мереж є те, що їх ваги задаються випадковим чином перед початком навчання і випадкова ініціалізація впливає на процес навчання моделі. Це означає, що навіть при використанні одних і тих же параметрів можна отримати дуже різні моделі, задаючи різні стартові значення генератора псевдовипадкових чисел.

Одна з базових переваг нейронних мереж полягає в тому, що вони здатні обробляти інформацію, що міститься у великих обсягах даних, і будувати неймовірно складні моделі. При наявності достатнього часу обчислень, даних і ретельного налаштування параметрів нейронні мережі часто перевершують інші алгоритми машинного навчання. Але нейронні мережі, особливо великі нейронні мережі, як правило, вимагають тривалого часу навчання. Також вони вимагають попередньої обробки даних, наприклад, того, щоб все вхідні ознаки були виміряні в одному і тому ж масштабі, в ідеалі вони повинні мати нульове середнє і дисперсію 1.

PyTorch — бібліотека машинного навчання яка особливо корисна для розробки систем штучного інтелекту для комп'ютерного бачення, обробки природної мови та аналізу часових рядів. Розробляє PyTorch переважно група дослідження штучного інтелекту компанії Facebook. PyTorch є вільним та відкритим програмним забезпеченням.

PyTorch має таку високорівневу функціональність:

- Тензорні обчислення із значним прискоренням завдяки використанню розрахунків на графічних процесорах (GPU);

- зручна розробка глибинних нейронних мереж.

Поряд з Keras (TensorFlow) PyTorch є найбільш розповсюдженим в розробці систем глибинного навчання в дослідженнях та промисловому застосуванні.

Більш детально з особливостями PyTorch можна ознайомитись в офіційній документації та навчальних матеріалах за посиланням https://pytorch.org/tutorials/

Робоче завдання

1. Ознайомитись із навчальним відео по розробці штучної глибокої нейронної мережі за посиланням
How to Code A Deep Neural Network From Scratch | PyTorch Tutorial
2. Реалізувати запропоновану нейронну мережу, отримати результати з точності класифікації із параметрами за замовчуванням.
3. Поміняти параметр batch_size на 10, 100. Отримати результати з точності, зробити висновки.
4. Поміняти параметр learning_rate на 0.01, 0.0001. Отримати результати з точності, зробити висновки.
5. Поміняти кількість епох на 150, 350. Отримати результати з точності, зробити висновки.
6. Реалізувати глибоку нейронну мережу для класифікації зображень рукописних цифр, з використанням заздалегідь створених шарів в PyTorch з використанням настанов за цим посиланням.

Контрольні питання:

1. Що позначають параметри, вплив яких на точність класифікації досліджено в роботі?
2. Що таке “глибока нейронна мережа”?

ЛАБОРАТОРНА РОБОТА №7*

Реалізація методів машинного навчання на прикладі задачі біометричної ідентифікації особи за райдужною оболонкою ока

Мета роботи: Ознайомлення з засадами побудови біометричних систем ідентифікації особи за райдужною оболонкою ока, а також вивчення і реалізація методів машинного навчання. Виявлення і порівняння інформативних ознак райдужної оболонки ока, отриманих в різних координатних базисах, а також вибір методу класифікації, який забезпечує найбільшу точність при ідентифікації особи за райдужною оболонкою ока.

Загальні теоретичні відомості

Розвиток інформаційних технологій нерозривно пов'язаний з проблемою доступу до конфіденційної інформації або матеріальних об'єктів за допомогою ініціалізації користувача. Верифікація предмета ініціалізації (магнітної карти, ключа, бейджа, пароля) не підтверджує ідентичність особи, тому не може гарантувати, що вона дійсно має право доступу. З метою захисту від несанкціонованого доступу сучасні системи контролю доступу широко застосовують біометричні методи. Під біометричними технологіями розуміють автоматичні або автоматизовані методи розпізнавання особи за її біометричними характеристиками. Існують методи біометричної ідентифікації особи за геометрією руки, особливостями рис обличчя, формі вуха, відбитками пальців, голосу людини, електронного підпису. Одним з найбільш точних біометричних ознак з точки зору ймовірності розпізнавання є райдужна оболонка ока.

Існуючі методи розпізнавання райдужної оболонки ока використовують ознаки, отримані за допомогою просторових фільтрів, коефіцієнтів двовимірного перетворення Фур'є, двовимірного вейвлет-перетворення, методу головних компонент і ін. У деяких роботах використовують не самі коефіцієнти розкладів, а їх статистичні показники, такі як середнє значення, середньоквадратичне відхилення, ентропія, енергія. Однак подібні статистичні показники можуть істотно не відрізнятися для різних людей. Підходи до розпізнавання райдужної

оболонки ока, засновані на застосуванні двовимірного вейвлет-перетворення, відрізняються між собою видом материнської вейвлет-функції, кількістю рівнів вейвлет-розкладу, вибором різних комбінацій вейвлет-коефіцієнтів при формуванні вектора ознак.

Для сканування райдужної оболонки ока як правило використовується портативні камери, що працюють в ближньому інфрачервоному діапазоні (від 850 до 900 нм). Після виділення на зображенні ока і його райдужної оболонки необхідно знайти набір інформативних ознак (образ), який буде використовуватися для ідентифікації особи. При розпізнаванні особи цей образ порівнюється або єдиним еталоном (верифікація), або з усіма зареєстрованими еталонами (ідентифікація).

На етапі розпізнавання можливе порівняння як безпосередньо пікселів вхідного зображення райдужної оболонки ока і еталона, так і отримання інформативних ознак зображень райдужної оболонки ока в різних координатних базисах, що дозволяє значно знизити розмірність даних. Формування ознак можна формалізувати як перетворення T вихідного зображення I(x,y) до вектору ознак Y: Y=T{I(x,y)}, де T { . } — оператор перетворення. В якості оператора T для формування ознак райдужної оболонки ока може бути використано дискретне перетворення Фур'є, просторові фільтри, дискретне косинусне перетворення, вейвлет-перетворення та ін. В процесі формування ознак необхідно враховувати, що при класифікації важливі ті ознаки, які найбільш подібні всередині класу (тобто для однієї особи), і в той же час найбільш розрізняються між ознаками різних класів (різних осіб).

Для дискретного зображення розміром $M \times N$ двовимірний спектр за Фур'є визначається як

$$F(k,p) = \frac{1}{M \times N} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m,n) e^{-j2\pi(\frac{km}{M} + \frac{pn}{N})}$$

де k, p номери гармонічних спектральних компонент зображення, $k = \overline{0, M-1}, p = \overline{0, N-1}$ Спектр дискретного зображення - це періодична функція частот k, p з періодами, які дорівнюють розмірам зображення $F(k \pm qM, p \pm cN) = F(k,p)$ для цілих чисел q, c.

Дискретне косинусне перетворення подає зображення райдужної оболонки у вигляді суми гармонік з різною амплітудою та частотою. Двовимірне дискретне косинусне перетворення зображення $I(m, n)$ розміром $M*N$ визначається наступним чином

$$B_{pq} = \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m,n) \mathrm{Cos} \frac{\pi(2m+1)p}{2M} \mathrm{Cos} \frac{\pi(2n+1)q}{2N},$$

$$0 \le p \le M-1, \quad 0 \le q \le N-1, \quad \alpha_p = \begin{cases} 1/\sqrt{M}, & p = 0 \\ \sqrt{2/M}, & 1 \le p \le M-1 \end{cases}$$

$$\alpha_q = \begin{cases} 1/\sqrt{N}, & q = 0 \\ \sqrt{2/N}, & 1 \le q \le N-1 \end{cases}$$

Двовимірне вейвлет-перетворення отримується в результаті застосування одновимірного вейвлет-перетворення до рядків і стовпців зображення. Багаторівневий двовимірний вейвлет-розклад зображення райдужної оболонки ока виконується для отримання коефіцієнтів апроксимації та деталізації у трьох орієнтаціях: горизонтальній, вертикальній та діагональній. На кожному наступному етапі коефіцієнти апроксимації рівня *j* розкладаються для отримання 4 компонент: апроксимуючої складової рівня *j*+1 та трьох видів деталізуючих складових. Процедура повторюється до необхідного рівня розкладу. На рис.7.1 наведено приклади двовимірного вейвлет-розкладу зображення райдужної оболонки ока до 4 рівня з використанням вейвлет-функції Добеші 2 порядку.

Масштабуюча функція визначається як

$$\varphi_{j,x,y}(m,n) = 2^{j/2} \varphi(2^j m - x, 2^j n - y),$$

$$j = 0, 1, 2, ..., J-1, \quad x, y = 0, 1, 2, ..., 2^j - 1,$$

а вейвлет-функція чутлива до напрямку і може бути визначена як

$$\psi^i{}_{j,x,y}(m,n) = 2^{j/2} \varphi^i(2^j m - x, 2^j n - y), \; i = \{H, V, D\}.$$

Апроксимуюча складова на рівні *j* знаходиться як

$$W_\varphi(j,x,y) = \frac{1}{\sqrt{M \times N}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m,n) \varphi_{j,x,y}(m,n).$$

Деталізуючі складові $W_\psi^H, W_\psi^V, W_\psi^D$ (горизонтальні, вертикальні та діагональні):

$$W_\psi^i(j,x,y) = \frac{1}{\sqrt{M \times N}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m,n) \psi^i{}_{j,x,y}(m,n), \; i = \{H, V, D\}$$

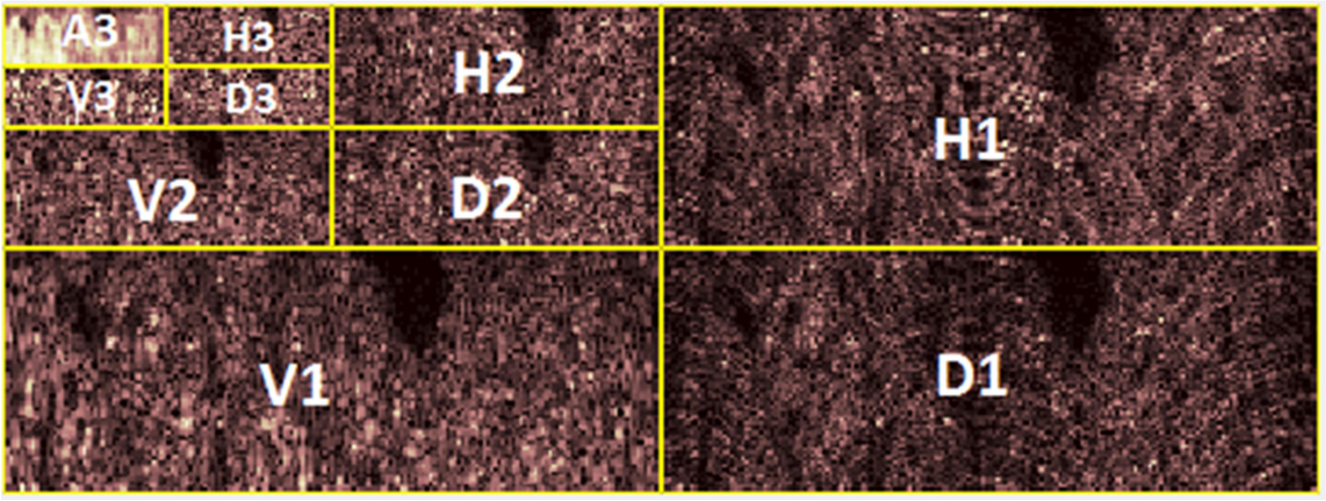


Рис.7.2. Вейвлет-коефіцієнти для зображення райдужної оболонки ока

Робоче завдання

ЧАСТИНА І – ПОПЕРЕДНЯ ОБРОБКА ТА АНАЛІЗ ЗОБРАЖЕННЯ РАЙДУЖНОЇ ОБОЛОНКИ ОКА

1. Завантажити з бази UBIRIS 24-бітові кольорові зображення райдужної оболонки ока з початковим розміром 800*600 пікселів для 2-3 різних осіб. Нормалізувати їх до розміру 450 * 450 пікселів.

2. Для одного з зображень побудувати окремо R, G, B матриці кольорового зображення у 2D та 3D поданні. Звернути увагу на перепади яскравості у зоні зіниці ока.

3. Перетворити початкове зображення у напівтонове. Також побудувати його у 2D та 3D поданні (рис.7.2)· Навести у вигляді графіку зріз матриці на рівні середини зіниці ока. Зробити висновок щодо застосування перепаду яскравості для визначення радіусу зіниці ока.

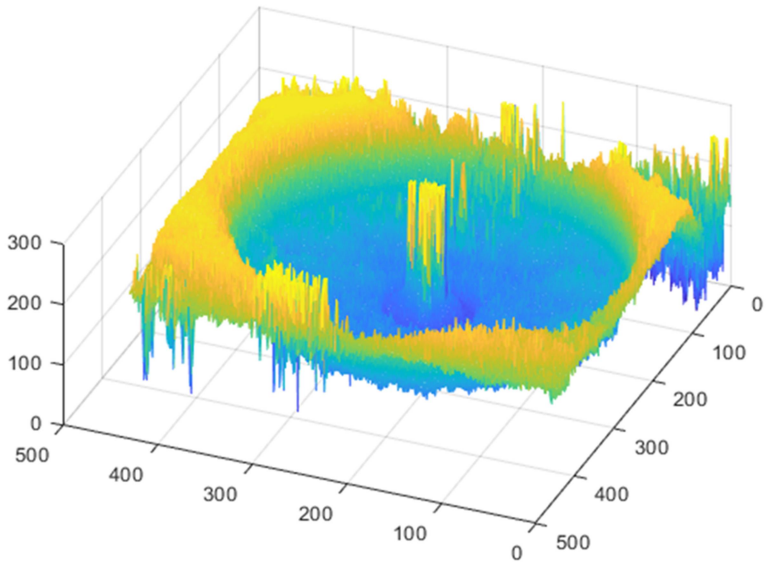
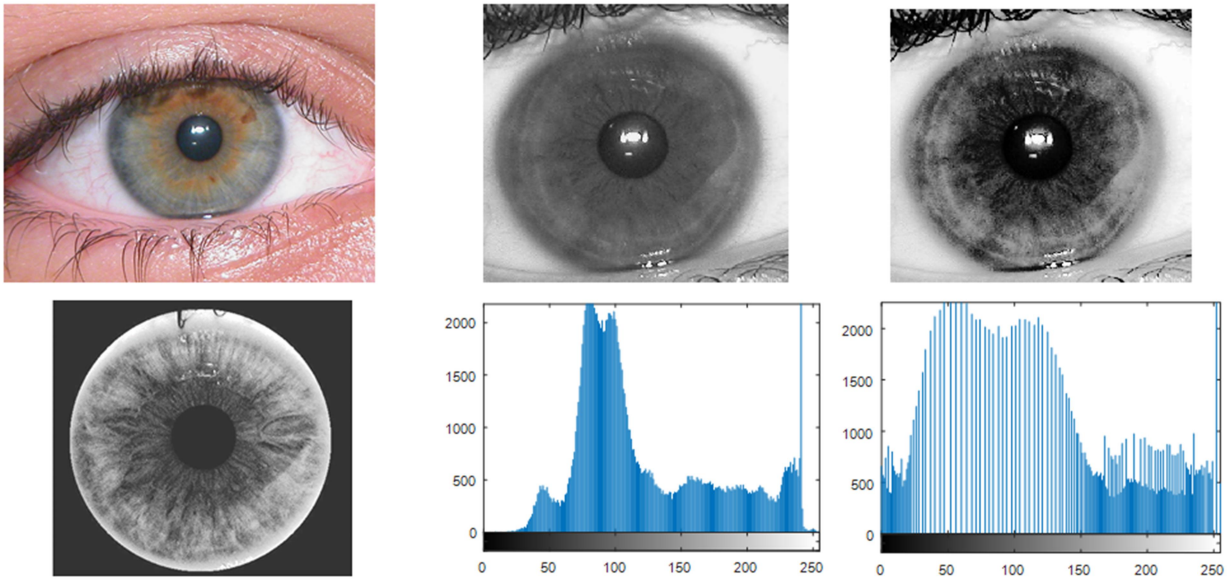


Рис. 7.2. Напівтонове зображення ока в 3D поданні

4. *Реалізувати наступні етапи обробки зображення:*

4.1. **Попередня обробка зображення з метою поліпшення його якості**

Так як дрібні деталі райдужної оболонки ока можуть бути погано помітні у разі низької контрастності зображення, на етапі попередньої обробки необхідно використати еквалізацію (вирівнювання) гістограми розподілу яскравостей зображення (рис. 7.3). По осі абсцис на гістограмі розподілу яскравостей зображення відкладено номери 256 градацій рівнів сірого, а по осі ординат - частота появи рівнів сірого в зображенні (кількість пікселів, які мають даний рівень сірого).



а) б) в)

Рис. 7.3. а) Початкове зображення та виділення райдужної оболонки шляхом сегментації зображення; б, в) зображення ока та гістограма розподілу яскравостей цього зображення до обробки (б) та після еквалізації гістограми (в)

4.2. **Подання зображення райдужної оболонки ока в полярній системі координат**

Приклад перетворення кільця райдужної оболонки з декартової системи координат у прямокутне зображення в полярній системі координат наведено на рис.7.4. Процес включає визначення центру та меж зіниці, знаходження радіусів кіл зіниці та райдужної оболонки, формування координат полярної системи та перетворення кожного пікселя райдужної оболонки з його подання в декартовій системі координат до представлення в полярній системі координат:

$$I(x(\rho,\theta),y(\rho,\theta)) => I(\rho,\theta) I(x(\rho,\theta),y(\rho,\theta)) => I(\rho,\theta),$$

$$x_p(\rho,\theta) = x_{p0}(\theta) + r_p * \mathrm{Cos}(\theta) x_p(\rho,\theta) = x_{p0}(\theta) + r_p * \mathrm{Cos}(\theta),$$

$$y_p(\rho,\theta) = y_{p0}(\theta) + r_p * \mathrm{Sin}(\theta) y_p(\rho,\theta) = y_{p0}(\theta) + r_p * \mathrm{Sin}(\theta),$$

$$x_i(\rho,\theta) = x_{i0}(\theta) + r_i * \mathrm{Cos}(\theta) x_i(\rho,\theta) = x_{i0}(\theta) + r_i * \mathrm{Cos}(\theta),$$

$$y_i(\rho,\theta) = y_{i0}(\theta) + r_i * \mathrm{Sin}(\theta) y_i(\rho,\theta) = y_{i0}(\theta) + r_i * \mathrm{Sin}(\theta),$$

де r_p та r_i відповідно радіуси зіниці та райдужної оболонки, $(x_p(\theta), y_p(\theta))$ та $(x_i(\theta), y_i(\theta))$ координати меж зіниці та райдужної оболонки у напрямку θ. Величина θ належить проміжку $[0; 2\pi]$, величина ρ проміжку $[0; 1]$.

Рис.7.4. Прямокутне зображення райдужної оболонки ока у полярній системі координат

5. Виділення зони інтересу на прямокутному зображенні райдужної оболонки ока в полярній системі координат

Досить часто частина райдужної оболонки закривається повіками. У цьому випадку в розгорнутому зображенні райдужної оболонки виникають спотворення, які викликаються шкірою верхніх та нижніх повік та вій. Отже, в якості сегменту для подальшої робот из зображенням використати область інтересу (ROI) зі сторони зіниці, яка визначається, як показано на рис.7.5.

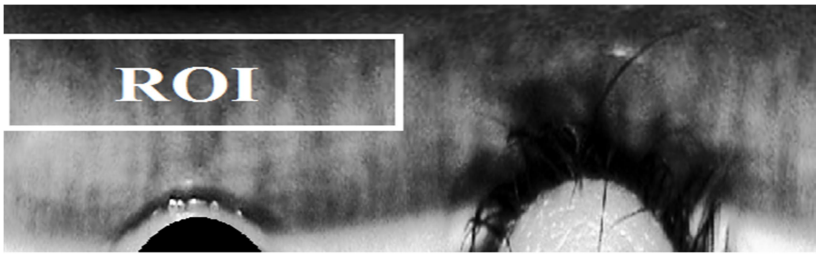


Рис.7.5. Виділення зони інтересу ROI на зображенні райдужної оболонки в полярній системі координат

5. **Визначити ознаки, які характеризують текстуру райдужної оболонки ока та застосовуються для її розпізнавання**

Розрахувати та побудувати у протоколі у вигляді 2D та 3D зображень (навести поруч інформацію щодо розмірності даних ознак):

- коефіцієнти двовимірного дискретного перетворення Фур'є,
- коефіцієнти двовимірного дискретного косинусного перетворення,
- коефіцієнти апроксимації та деталізації 2-го, 3-го і 4-го рівнів двовимірного вейвлет-перетворення)

Візуально порівняти результати для 2 різних людей.

ЧАСТИНА 2 – РЕАЛІЗАЦІЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ

1. Обрати 2 підходи до отримання ознак райдужної оболонки ока (наприклад, серед наступних ознак: коефіцієнти двовимірного дискретного перетворення Фур'є, коефіцієнти двовимірного дискретного косинусного перетворення, коефіцієнти апроксимації та деталізації 2-го, 3-го чи 4-го рівнів двовимірного вейвлет-перетворення, на вибір ознаки розраховувати для напівтонового зображення або для окремо взятих матриць R, G, B) .
2. Машинне навчання провести для зображень райдужної оболонки ока 50 осіб, по 5 різних зображень для кожної особи. Для оцінювання точності роботи класифікаторів використовувати перехресна перевірка, кросвалідацію, з розбиттям вибірки даних на 5 частин, з яких на 4-х проводити навчання, а 5ту застосовувати для тестування. Процедуру повторювати 5 разів таким чином, щоб кожна з п’яти частин використовувалася для тестування, а результати точності класифікації усереднювалися.
3. Для 2 різних наборів ознак проаналізувати результати машинного навчання низки класифікаторів, серед яких дискримінантний аналіз, метод k-найближчих сусідів, метод k-середніх, метод опорних векторів, дерева прийняття рішень та ін.
4. Навести у таблиці точність розпізнавання райдужної оболонки ока в залежності від набору ознак та типу класифікатора.
5. Проаналізувати матриці похибок класифікації. Зробити висновки.

Контрольні питання:

1. Наведіть приклади можливого формування ознак для задачі розпізнавання особи за райдужною оболонкою ока.
2. Навіщо застосовувати полярну систему координат при аналізі зображень райдужної оболонки ока?
3. Який метод машинного навчання дозволив вам отримати найкращі результати класифікації для розпізнавання особи за райдужною оболонкою ока?

ЛАБОРАТОРНА РОБОТА №8*

Реалізація методів машинного навчання на прикладі задачі розпізнавання голосу

Мета роботи: Ознайомлення з засадами побудови біометричних систем ідентифікації особи за голосом, а також вивчення і реалізація методів машинного навчання. Виявлення і порівняння інформативних ознак голосового сигналу, отриманих в різних координатних базисах, а також вибір методу класифікації, який забезпечує найбільшу точність при ідентифікації особи за голосом, а також для розпізнавання голосових команд користувача.

Загальні теоретичні відомості

Мова – це послідовність різних звуків. В свою чергу звук являє собою суперпозицію звукових коливань різних частот. Більшість методів розпізнавання голосових сигналів складаються з алгоритмів визначення набору ознак, який максимально точно описує властивості сигналу, а також з алгоритмів порівняння за цим набором ознак еталонних сигналів з бази даних та вхідного сигналу, який необхідно розпізнати. Слід розрізняти два основні напрямки розпізнавання голосових сигналів. Перший – розпізнавання мови – дає відповідь на запитання, що говорить людина. Другий напрямок – ідентифікація (верифікація) особи – дає відповідь на запитання, хто говорить. Інтенсивний розвиток систем голосової ідентифікації та верифікації особистості пояснюється актуальністю їх застосування в таких областях, як криміналістика, біометричний пошук, голосова верифікація користувачів, розмежування прав доступу до інформації за допомогою голосової біометрії. Унікальність голосу окремої людини пояснюється особливостями будови її голосових зв'язок, трахеї і носових порожнин, а також манерою вимови звуків, особливостями розташуванням зубів.

Біометрична ідентифікація людини за голосом може бути реалізована шляхом аналізу структури мовного сигналу, яка є послідовністю сплесків коливань різної амплітуди і частоти, а також пауз між ними. Завдання

розпізнавання ускладнюється присутністю у звуковому сигналі шумів оточуючої середи та каналу зв'язку, до яких відносяться спотворення мікрофона і каналу передачі, а також похибки кодування аудіосигналу. Для порівняння вхідного мовного сигналу і еталона використовується певна міра близькості між кожною парою порівнюваних параметрів вхідного мовного сигналу і еталона. При цьому можна порівнювати як безпосередньо відліки вхідного мовного сигналу та еталону після вирівнювання сигналів за початком мовлення, так і подання цих сигналів у різних координатних базисах, які дозволяють отримати інформативні ознаки мовних сигналів. Доцільно провести порівняння сигналу та еталону у часовій та частотній областях, а також проаналізувати схожість їх спектрограм, скейлограм, мелкепстрограм.

На рис.8.1а наведено еталонний мовний сигнал (прізвище та ім’я користувача, вимовлені особою). Для порівняння на рис. 8.1б наведено сигнал, що відповідає вимовленню тієї самої фрази, але іншою особою. Видно, що наведені фрагменти мають явно виражені періоди коливального характеру. Період коливань і їх характер є досить індивідуальними. У різних людей своєрідно різними є періоди основного тону та форма внутрішніх коливань перехідних процесів.

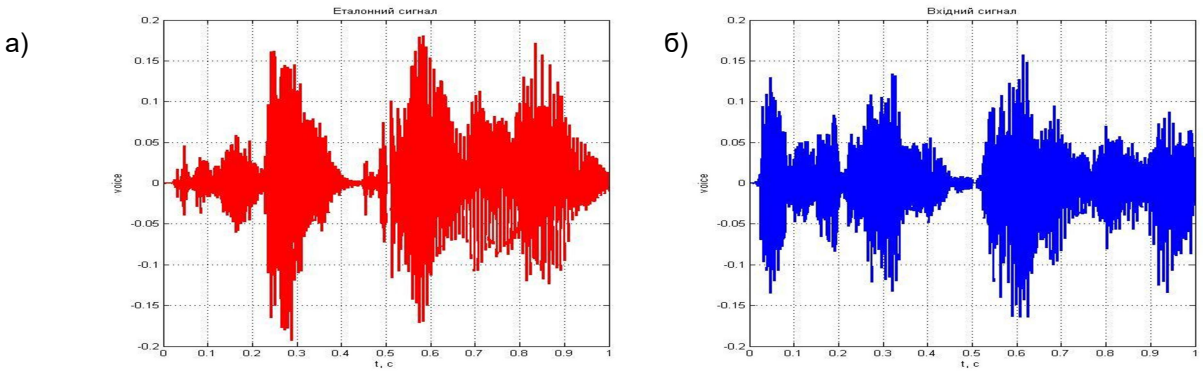


Рис.8.1. Мовні сигнали: а) еталонний сигнал (прізвище та ім’я, вимовлені особою); б) вхідний сигнал (така ж сама фраза, вимовлена іншою особою)

Найбільш простий шлях порівняння звукових сигналів - це порівняння звукового сигналу за формою хвилі. При цьому порівнюється значення сигналів

для кожного моменту часу. Критерієм оцінки близькості двох мовних сигналів може служити середньоквадратичне відхилення амплітуд сигналів RMS:

$$RMS = \sqrt{\frac{1}{N}\sum_{i=1}^{n}(E_i - S_i)^2},$$

де N – кількість відліків в еталонному та вхідному сигналах, що порівнюються.

Аналіз залежності спектральної густини потужності від частоти дозволяє дослідити відмінності еталонного та вхідного сигналів у частотній області. Так у спектрі еталонного сигналу, наведеного на рис.2а, основна спектральна густина потужності зосереджена в області до 250 Гц, домінуюча частота складає 119 Гц, також спостерігаються 2 значні піки на частотах 233 та 238 Гц. У спектрі вхідного сигналу, наведеного на рис.8.2б, є значний пік на частоті 144 Гц. Але найбільша різниця між спектрами двох сигналів полягає в тому, що у спектрі вхідного сигналу присутні значні високочастотні складові у діапазоні 250-450 Гц. Отримані результати свідчать про те, що порівняння спектрів доцільно проводити із застосуванням RMS метрики, розбиваючи спектр на частотні смуги.

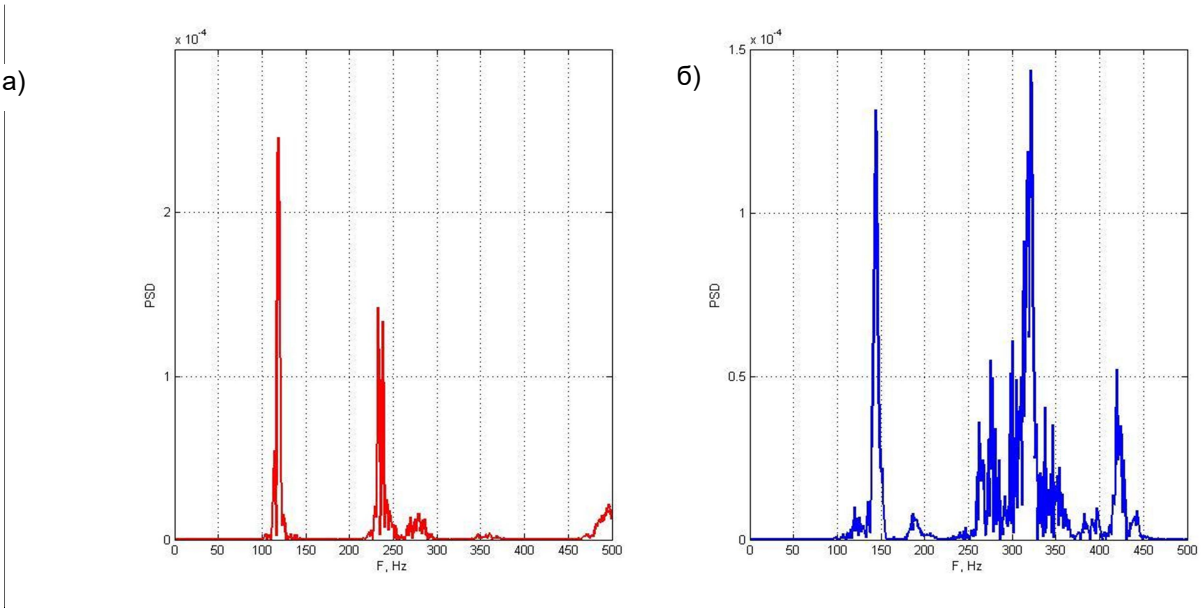


Рис.8.2. Спектральна густина потужності: а) еталонний сигнал (прізвище та ім’я, вимовлені особою); б) вхідний сигнал (така ж сама фраза, вимовлена іншою особою)

Якісно інший метод порівняння звукових сигналів грунтується на дослідженні частотних властивостей мовного сигналу, що змінюються у часі. Частотно-часовий аналіз проводиться шляхом порівняння спектрограм,

побудованих для еталонного сигналу та вхідних сигналів, що потребували розпізнавання. Для отримання спектрограм проводиться сегментація досліджуваного мовного сигналу з деяким кроком за часом dt, після чого для кожного з цих сегментів розраховується спектр (спектральна густина потужності) з використанням віконної функції Хемінга. При цьому вибір розміру сегментів обумовлюється компромісом між часовою і частотною роздільною здатністю: для більшого значення розміру сегмента спостерігається краща роздільна здатність по частоті, але гірша роздільна здатність у часі, і навпаки. Отримані спектри формують двовимірний масив (час, частота), який представляє собою спектрограму голосового сигналу (рис. 8.3). Порівняння спектрограм також доцільно проводити в окремих частотних чи часових смугах.

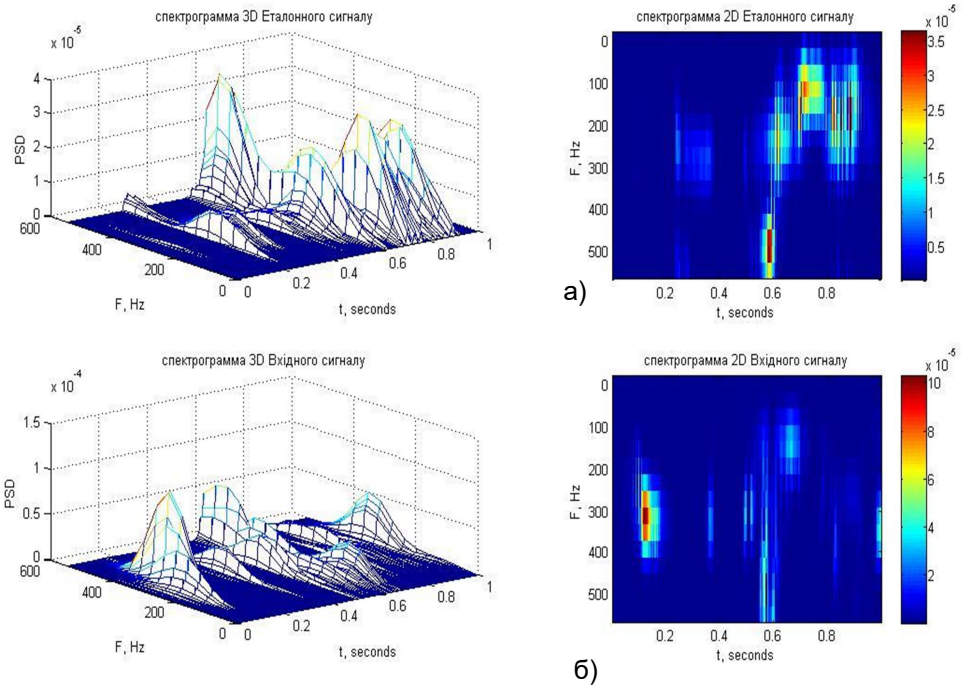


Рис.8.3. Спектрограма: а) еталонний сигнал (прізвище та ім’я, вимовлені особою); б) вхідний сигнал (така ж сама фраза, вимовлена іншою особою)

Мел-кепстральні коефіцієнти (MFCC)

Перший крок в аналізі мовних даних - це виділення ознак, які є значущими для ідентифікації лінгвістичного змісту і відкидання всіх інших ознак, що відповідають за шум і емоції. Шкала Мел співвідносить сприйняту частоту або висоту чистого тону (мел) з фактичної виміряної частотою (Гц). Люди набагато

краще розрізняють невеликі зміни висоти звуку на низьких частотах, ніж на високих. Ця залежність нелінійна і описується наступною формулою:

$$M(f) = 1127 * \ln(1 + f/700)$$

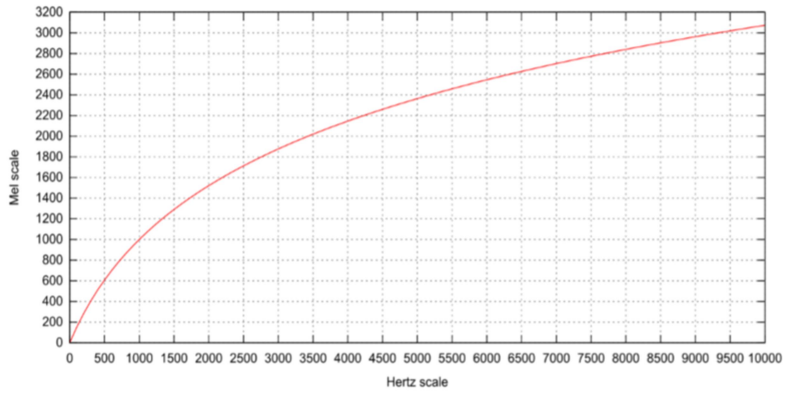


Рис.8.4. Залежність частоти від мел

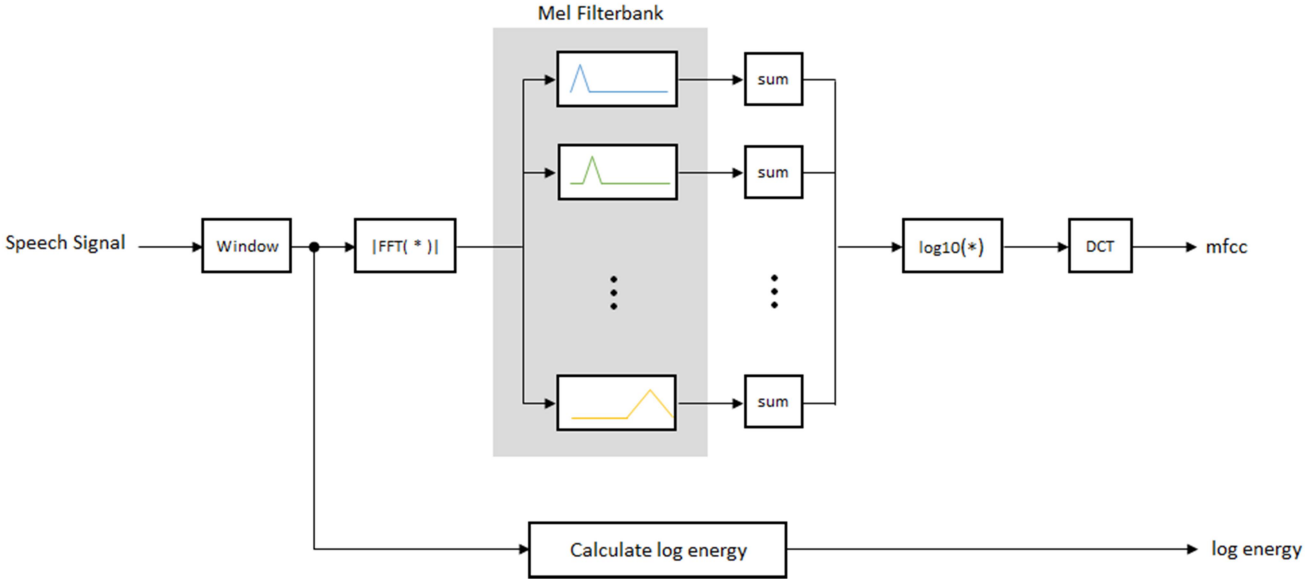


Рис.8.5. Обчислення мел-частотних кепстральних коефіцієнтів

Обчислення мел-частотних кепстральних коефіцієнтів включає в себе наступні кроки (рис.8.5):

1. Необхідно розділити вихідний сигнал на фрейми. Розмір фрейму зазвичай обирається від 20 до 40 мс, так як вважається, що мовний сигнал на цьому проміжку не сильно змінюється.

Наступні кроки застосовуються для кожного окремого фрейма.

2. Мовний сигнал є скінченим і не є періодичним, тому через розриви на його кінцях при застосуванні перетворення Фур'є проявляється ефект витоку. Для того,

щоб знизити його вплив на результат, кожен фрейм множиться на віконну функцію Хемінга. До результату застосовується дискретне перетворення Фур'є та розраховується періодограма для кожного фрейму.

3. Обчислюється блок мел-фільтрів (рис.8.6). Для цього трикутні фільтри (від 20 до 40) множаться на періодограму і сумуються. В результаті отримуються енергії набору фільтрів.

4. Отримані енергії логарифмуються. Це також мотивується людським слухом: ми не чуємо гучність в лінійному масштабі. Зазвичай, щоб подвоїти сприйняту гучність звуку, потрібно затратити в 8 разів більше енергії. Ця операція стиснення робить функції ближчими до того, що насправді чують люди.

5. Далі, використовуючи дискретне косинусное перетворення, отримуються мел-кепстральні коефіцієнти.

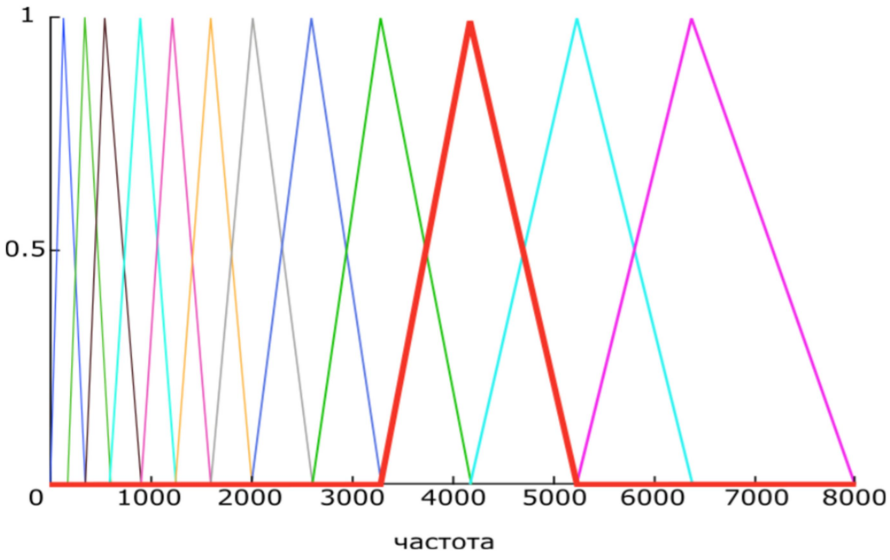


Рис.8.6. Фільтри “збираються” в області низьких частот, забезпечуючи більш високу роздільну здатність там, де це необхідно для розпізнавання

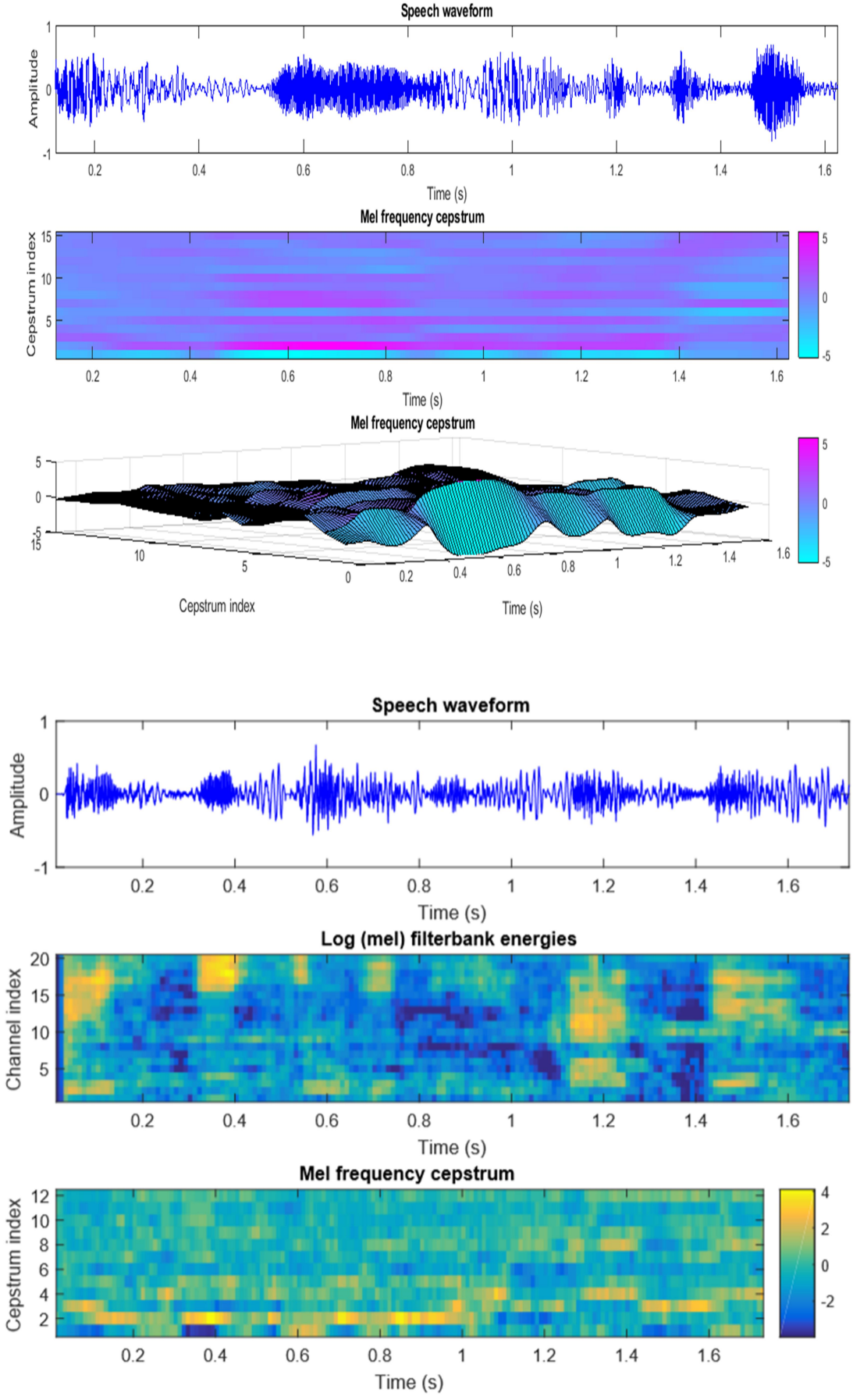


Рис.8.7. Приклади мел-кепстрального аналізу звукового сигналу

Робоче завдання

ЧАСТИНА І

ТЕКСТОЗАЛЕЖНА ІДЕНТИФІКАЦІЯ ОСОБИ ЗА ГОЛОСОМ

1. Кожному зі студентів навчальної групи записати 10 разів голосову команду "*ВИКОНАТИ ІДЕНТИФІКАЦІЮ*". Аудіо файли називати з використанням спільної частини (voice), свого прізвища (lukianenko) та номера реалізації (1) за прикладом: voice_lukianenko_1.m, voice_lukianenko_2.m, voice_lukianenko_3.m.…). Параметри запису обрати наступні: тривалість запису T=1.5 сек (середня тривалість 1 слова – 500 мс), частота дискретизації сигналу Fs=11025 Hz.

2. Створити загальну базу з сигналів всіх студентів навчальної групи.

3. Визначити ознаки, які характеризують звуковий сигнал та можуть застосовуються для його розпізнавання. Розрахувати та побудувати у протоколі для 2х різних людей частотний спектр сигналу, спектрограму, скейлограму, мел-кепстрограму записаних голосових сигналів (рис.8.7). Порівняти, зробити висновки.

4. Розрахувати ознаки для розпізнавання особи для навчальної вибірки даних (записи голосу студентів групи). Для можливості порівняння обрати ознаки, отримані 2-ма різними методами аналізу, наприклад, спектрально-часовий аналіз та мел-кепстральні коефіцієнти (довжина фреймів 20 мс, перекриття 50%).

5. Машинне навчання провести для звукових сигналів 10 осіб, по 10 реалізацій для кожної особи. Для оцінювання точності роботи класифікаторів використовувати перехресна перевірка, кросвалідацію, з розбиттям вибірки даних на 5 частин, з яких на 4-х проводити навчання, а 5ту застосовувати для тестування. Процедуру повторювати 5 разів таким чином, щоб кожна з п'яти частин використовувалася для тестування, а результати точності класифікації усереднювалися. Для 2 різних наборів ознак проаналізувати результати машинного навчання низки класифікаторів, серед яких дискримінантний аналіз, метод k-найближчих сусідів, метод k-середніх, метод опорних векторів, дерева прийняття рішень та ін.

6. Дослідити точність класифікації з використанням мел-кепстральних коефіцієнтів в залежності від обраної кількості коефіцієнтів, кількості фільтрів та інш. параметрів на вибір.

7. Навести у таблиці точність розпізнавання особи за текстозалежним голосовим сигналом в залежності від набору ознак та типу класифікатора. Проаналізувати матриці похибок класифікації. Зробити висновки.

8. Записати ще раз фразу "*ВИКОНАТИ ІДЕНТИФІКАЦІЮ*" (нова реалізація, яка не приймала участь в навчанні). Розрахувати для неї вектор ознак, але не використовувати мітку класу.

9. Написати код, який використовує передбачення класу для тестового голосового сигналу, та у разі класу "свій" виводить на екран привітання, а у разі "чужих" класів виводить на екран сповіщення про заборону доступу (рис.8.8).

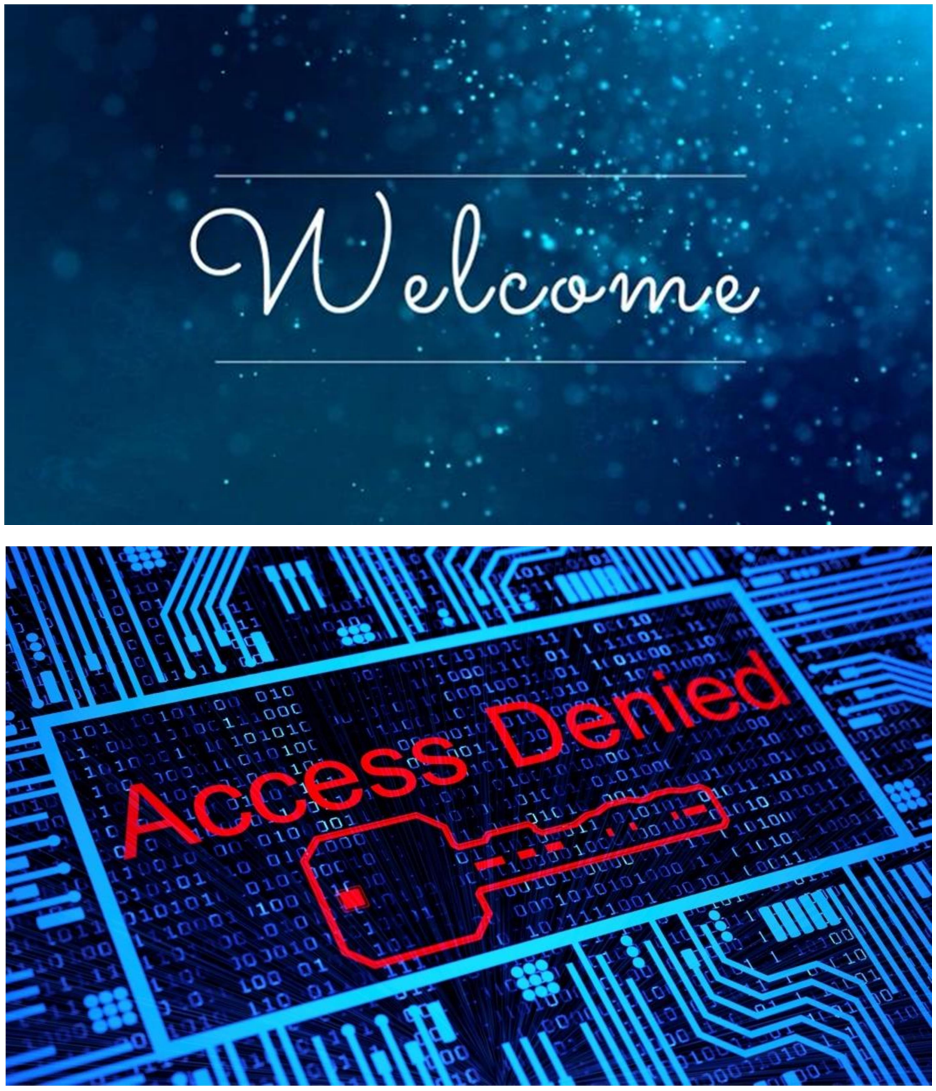


Рис.8.8. Приклад сповіщень розробленої системи як результат розпізнавання голосу користувача

ЧАСТИНА ІІ – РОЗПІЗНАВАННЯ КОМАНД КОРИСТУВАЧА ДЛЯ ГОЛОСОВОГО КЕРУВАННЯ

1. Кожному зі студентів навчальної групи засобами системи MATLAB записати по 10 разів кожну з набору голосових команд, наприклад "*ПОКАЗАТИ ЗОБРАЖЕННЯ*", "*ПРОГРАТИ АУДІОЗАПИС*", "*ПОБУДУВАТИ ГРАФІК*". Аудіо файли називати з використанням спільної частини (voice), типу команди (image, audio, fig) та номера реалізації (1) за прикладом: voice_image_1.m, voice_image_2.m, voice_image_3.m….). Параметри запису обрати наступні: тривалість запису T=1.2 сек (середня тривалість 1 слова – 500 мс), частота дискретизації сигналу Fs=11025 Hz.
2. Розрахувати ознаки для розпізнавання команди для навчальної вибірки даних (реалізації голосових команд). Для можливості порівняння обрати ознаки, отримані 2-ма різними методами аналізу.
3. Проаналізувати результати машинного навчання низки обраних класифікаторів. Проаналізувати матриці похибок класифікації. Зробити висновки.
4. Записати тестову команду (нова реалізація, яка не приймала участь в навчанні). Розрахувати для неї вектор ознак, але не використовувати мітку класу.
5. Написати код, який використовує передбачення класу для тестових команд, та в залежності від розпізнаної команди виводить на екран її реалізацію.

Контрольні питання:

1. Наведіть приклади можливого формування ознак для задачі розпізнавання особи за голосом або розпізнавання команд голосового керування.
2. Який метод машинного навчання дозволив вам отримати найкращі результати класифікації для розпізнавання особи за голосом, а також для розпізнавання команд голосового керування?

Список використаної літератури

1. M. Swamynathan, Mastering Machine Learning with Python in Six Steps, DOI 10.1007/978-1-4842-2866-1

2. Python для складних завдань. Наука про дані та машинне навчання | Вандер Плас Дж., 576 с., 2017, ISBN 978-5-4461-0914-2, 978-5-496-03068-7

3. Адреас Мюллер, Сара Гвідо: Вступ до машинного навчання за допомогою Python, 480 с., 2017, ISBN 978-5-9908910-8-1.

4. M. Swamynathan, Mastering Machine Learning with Python in Six Steps, DOI 10.1007/978-1-4842-2866-1

5. Nikhil Ketkar. Deep Learning with Python: A Hands-on Introduction. Apres, 2017, 226 p.

6. Aurélien Géron. Hands-On Machine Learning with ScikitLearn and TensorFlow. O'Reilly Media, 2017, 751 p.

7. Rangayyan, Rangaraj M. Biomedical Signal Analysis, Wiley (2015), IEEE Press series in biomedical engineering, DOI:10.1002/9781119068129.