НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

лабораторної роботи №2

з дисципліни «[СИСТЕМИ ГЛИБИННОГО НАВЧАННЯ](https://classroom.google.com/c/NjIxMzk4NjAyNjI4)»

на тему

«Розробка програмного забезпечення для реалізації ймовірнісної  
нейронної мережі PNN»

|  |  |
| --- | --- |
| Виконав:  Студент групи КМ-03  Мітченко А.Д. | Перевірив:  доцент  Терейковський І. А. |
|  |  |
|  |  |
|  |  |

Київ — 2023

ЗМІСТ

[Теоретичні відомості 3](#_Toc150269750)

[Умова завдання: 5](#_Toc150269751)

[Скріншоти та опис програми 6](#_Toc150269752)

[ДОДАТОК А ТЕКСТ ПРОГРАМИ 7](#_Toc150269753)

[Завдання : 7](#_Toc150269754)

# Теоретичні відомості

Probabilistic neural networks  (PNN) — це тип штучної нейронної мережі прямого зв’язку, яка тісно пов’язана з оцінкою щільності ядра (KDE) через вікно Парзена, яке асимптотично наближається до оптимальної мінімізації ризику Байєса. Ця техніка широко використовується для оцінки умовної щільності класу (також відомої як ймовірність) у завданнях машинного навчання, таких як контрольоване навчання.

Изображение выглядит как диаграмма, линия, круг

Автоматически созданное описание

Рис. 1 Структура NN

Изображение выглядит как Шрифт, линия, диаграмма, рукописный текст

Автоматически созданное описание

Рис. 2 Parzen window density estimation

Изображение выглядит как текст, рукописный текст, доска, Шрифт

Автоматически созданное описание

Рис. 3 Формула яка розраховує вхідний сигнал для нейронів шару додавання.

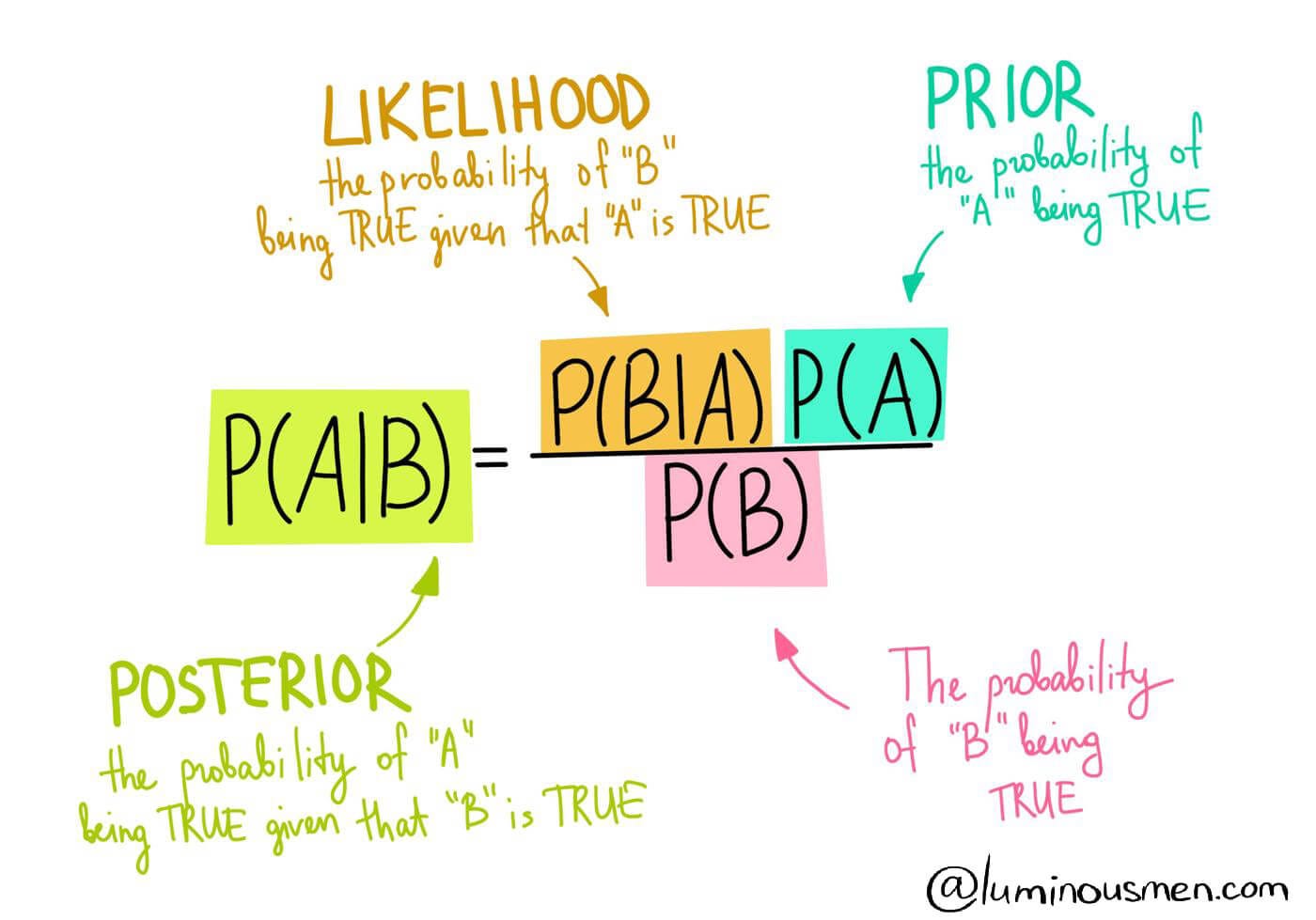


Рис 4. Теорема Байєса.

Нейронна мережа, яку представив Шпехт, складається з чотирьох рівнів:

1. Вхідний рівень: особливості точок даних (або спостережень)
2. Шар шаблону: розрахунок PDF умовного класу
3. Рівень підсумовування: підсумовування шаблонів між класами
4. Вихідний рівень: Перевірка гіпотези з максимальною апостериорною ймовірністю (MAP)

PNN мають кілька переваг, включаючи їх здатність обробляти дані великої розмірності, легкість навчання (особливо для малих і середніх наборів даних) і їх притаманний імовірнісний характер, який може забезпечити оцінки невизначеності для класифікації. Але є і обмеження. PNN можуть не підходити для надзвичайно великих наборів даних через вимоги до пам’яті та обчислень. Вони також можуть бути чутливими до вибору функції ядра та пропускної здатності у вікні Parzen.

Загалом PNN є корисним інструментом для завдань класифікації шаблонів, але їх придатність залежить від конкретної проблеми та набору даних. Вони використовувалися в різних програмах, зокрема для розпізнавання символів, медичної діагностики та контролю якості.

## Умова завдання:

Розробити програмне забезпечення для реалізації мережі PNN, що призначена для апроксимації функції у=х1+х2. Передбачити режими навчання та розпізнавання.

# Скріншоти та опис програми

Изображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

# ДОДАТОК А ТЕКСТ ПРОГРАМИ

## Завдання :

Розробити програмне забезпечення для реалізації мережі PNN, що призначена для апроксимації функції у=х1+х2. Передбачити режими навчання та розпізнавання.

# Імпорт нампая для створення масиву

import numpy as np

# Створюємо функцію яка розраховує вхідний сигнал для нейронів шару додавання.

def gaussian\_kernel(x, data\_point):

squared\_diff = np.sum((x - data\_point) \*\* 2)

return np.exp(-squared\_diff / (2 \* sigma \*\* 2))

# Тренуємо на навчальній вибірці.

def predict(x):

class\_sums = [0] \* (max(y\_train) + 1)

for i in range(len(X\_train)):

kernel\_value = gaussian\_kernel(x, X\_train[i])

class\_sums[y\_train[i]] += kernel\_value

predicted\_class = np.argmax(class\_sums)

return predicted\_class

# Я обрав сігму за 1 для спрощення розрахунків, але можна булоб обрати сігму за 0.1.

sigma = 1.0

X\_train = np.array([[1, 3], [3, 3], [3, 4], [4, 5]])

y\_train = np.array([4, 6, 7, 9])

X\_test = np.array([[2, 2], [3, 3]])

for x in X\_test:

predicted\_class = predict(x)

print(f"For input {x}, PNN predicts y = {predicted\_class}")