НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Радіотехнічний факультет

Радіотехнічних пристроїв та систем

До захисту допущено:

Завідувач кафедри

_____ Сергій ЖУК

«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою

«Радіотехнічні інформаційні технології»

спеціальності 172 «Телекомунікації та радіотехніка»

на тему: «Методи класифікації нот в акустичному сигналі»

Виконав:
студент IV курсу, групи РТ-71
Красницький Микита Олександрович _____

Керівник:
к.т.н.
Шпилька Олександр Олександрович _____

Рецензент:
к.т.н., доцент
Сушко Ірина Олександрівна _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2021 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Радіотехнічний факультет
Радіотехнічних пристроїв та систем

Рівень вищої освіти – перший (бакалаврський)
Спеціальність – **172 «Телекомунікації та радіотехніка»**
Освітньо-професійна програма «**Радіотехнічні інформаційні технології**»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій ЖУК
«___»_____2021 р.

ЗАВДАННЯ
на дипломну роботу студенту
Красницькому Микиті Олександровичу

1. Тема роботи «Методи класифікації нот в акустичному сигналі», керівник роботи Шпилька Олександр Олександрович, кандидат технічних наук, доцент, затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Термін подання студентом роботи 15 червня 2021р.

3. Вихідні дані до роботи _____
Теоретичні викладки методів класиіфкації_____
Аудиозапис, що містить музичні ноти_____
_____

4. Зміст роботи _____
Теоретичні засади музичного сигналу_____
Методи вирішення задач класифікації_____
Налаштування та дослідження методів классифікації_____

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)
Презентація в електроному вигляді з 16 слайдів_____
_____
_____

6. Консультанти розділів роботи*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 12 квітня 2021р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Теоретичні засади музики	21.02.2021	
2	Методи теорії класифікації	21.04.2021	
3	Налаштування БК	12.05.2021	
4	Налаштування SVM	24.05.2021	
5	Дослідження та висновки	12.06.2021	

Студент  Микита КРАСНИЦЬКИЙ

Керівник  Олександр ШПИЛЬКА

* Якщо визначені консультанти. Консультантом не може бути зазначено керівника дипломної роботи.

АНОТАЦІЯ

Дипломна робота викладена на 55 сторінках, містить 28 ілюстрацій, 2 таблиці та 4 додатки.

В даній роботі наведено частотне визначення нот, фізичні процеси що супроводжують збудження ноти в музичному інструменті та їх прояви в цифровому акустичному сигналі. Надалі ці теоретичні знання були використані для розробки емпіричних методів класифікації нот. Також теорія нот була використана для таких задач: вибір домену сигналу (частотного чи часового), вибір ширини вікна перетворення Фурье та інше.

Розглянуто популярні класифікатори, а саме Гаусівський Байєсів класифікатор та класифікатор на базі методу опорних векторів. Порівняно їх точності класифікації для задачі класифікації нот акустичного сигналу. Проведено експерименти з емпіричними методами класифікації та експерименти з поєднаннями класичних методів з емпіричними.

Ключові слова: Байесів класифікатор, Гаусів наївний Байесів класифікатор, SVM класифікатор, застосування машинного навчання, машинне навчання, класифікації цифрових акустичних сигналів, теорія нот, спектр, тембр, емпіричні методи класифікації.

ANNOTATION

Diploma project is presented on 55 pages, contains 28 illustrations, 2 tables and 4 appendices.

This paper describes the frequency definition of a note, physical processes which carry out realization of a note in a musical instrument and their displays in a digital acoustic signal. Later, this theoretical knowledge was used to develop empirical methods for classifying notes. Note theory was also used for the following tasks: signal domain selection (frequency or time), Fourier transform window width selection, and more.

The Gaussian Bayesian classifier and the SVM classifier are considered. Their classification accuracy for the problem of notes classification is compared. Experiments with empirical methods of classification and experiments with combinations of classical methods with empirical ones are carried out.

Keywords: Bayesian classifier, Gaussian naive Bayesian classifier, SVM classifier, application of machine learning, machine learning, classification of digital acoustic signals, note theory, spectrum, timbre, empirical methods of classification.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

SVM –support vector machines

FFT – Fast Fourier transform

БК – Байесівський класифікатор

ГК – Гаусівський класифікатор

Зміст

Анотація 4

Annotation 5

Перелік умовних позначень 6

Зміст 7

Вступ 9

1 Теортичні засади музичного сигналу 10

1.1 Теорія нот 10

1.2 Фізичні аспекти тембру звучання 15

2 Методи вирішення задач класифікації 22

2.1 Основи класифікації 22

2.2 Байесівска класифікація. Гаусовський наївний баєсів класифікатор 23

2.3 Метод опороних векторів 27

3 Налаштування методів классифікації 35

3.1 Вибір параметрів вхідного сигналу для класифікації 35

3.1.1 Домен сигналу 35

3.1.2 Довжина сигналу 35

3.1.3 Використання інтерполяції спектру 37

3.2 Емпіричні методи класифікації 39

3.3 Гаусівський наївний баєсів класифікатор 46

3.4 Класифікатор на базі методу опорних векторів 50

ВИСНОВКИ 53

Перелік джерел посилання 55

Додаток А 56

Додаток Б 58

Додаток В 61

Додаток Г 65

ВСТУП

Актуальність теми. Насьогодні машинне навчання має значний попит. Можна спостерігати зростаючий тренд збільшення сфер застосування машинного навчання та розвиток методів машинного навчання.

Дана робота є актуальною, так як вона розбирає математичні основи популярних на сьогодні та перевірених часом методів, а саме Баєсівського класифікатора та класифікатора на базі методу опорних векторів (SVM).

Ціль та задачі. Цілью даної роботи є проведення налаштування та порівняння точністних характеристик класифікаторів.

Для досягнення поставленої цілі необхідно виконати наступні задачі:

1) Провести аналіз музичних композицій та визначити ознаки для використання під час класифікації.
2) Освоїти теорію методу байєсовської класифікації та класифікації на базі методу опорних векторів.
3) Розробити емпіричний алгоритм класифікації музичної композиції шляхом зваженної суми спектральних компонент.
4) Провести дослідження щодо характеристик, що будуть використовуватись під час навчання Байєсівського класифікатора.
5) Провести дослідження щодо характеристик, що будуть використовуватись під час навчання класифікатора на базі методу опорних векторів.

Об’єкт дослідження - цифровий акустичний сигнал в якому записана нотна послідовність або композиція.

Предмет дослідження - методи використовуючи які отримуємо класифікацію акустичних сигналів. Предметами є: емпіричні класифікатори, Байесівський класифікатор, класифікатор SVM.

1 ТЕОРТИЧНІ ЗАСАДИ МУЗИЧНОГО СИГНАЛУ

1.1 Теорія нот

Нота - це графічна одиниця музикальної нотації або музична елементарна одиниця композиції, що містить інформацію про тривалість та висоту музичного звуку. Будь-яка класична композиція складається з нот і пауз.

Фізично ноти є вібрації в межах діапазону звукових частот, що можуть сприйматися людиною. Існує 7 основних нот: До, Ре, Мі, Фа, Соль, Ля, Сі. Слід додати, що стандартом є представлення нот латинецею, а саме послідовність наведена вище буде мати вигляд: C, D, E, F, G, A, B.

Півтон - найменший інтервал в традиційній і академічній музиці Європи. Ноти Мі і Фа, а так само ноти Сі і До знаходяться на відстані півтону, тоді як інші сусідні ноти знаходяться на відстані тону.

В теорії музики використовуються певний ряд інтервалів. В такими як бемоль і дієз, в нотній нотації позначають ноту яка на пів тона нижче або вище відповідно. Для цього існують окремі ноти, які називаються через основні 7, наприклад «До дієз» або «Ля бемоль». В силу того що 2 пари нот від самого початку були з відстанню півтону - до початкових 7ми нот додається тільки 5. Разом отримуємо 12 нот на октаву. показана класифікація стандартних інтервалів.

Таблиця 1.1 – Класифікація простих інтервалів

Кількість ступенів	Назва	Кількість тонів
1	чиста Прима	0
2	мала Секунда / велика Секунда	0,5 / 1
3	мала Терція / велика Терція	1,5 / 2
4	чиста Кварта / збільшена Кварта	2,5 / 3
5	зменшена Квінта / чиста Квінта	3 / 3,5
6	мала Секста / велика Секста	4 / 4,5
7	мала Септима / велика Септима	5 / 5,5
8	чиста Октава	6

Знаками альтерації, такими як бемоль і дієз, в нотній нотації позначають ноту яка на пів тона нижче або вище відповідно. Для цього існують окремі ноти, які називаються через основні 7, наприклад «До дієз» або «Ля бемоль». В силу того що 2 пари нот від самого початку були з відстанню півтону - до початкових 7ми нот додається тільки 5. Разом отримуємо 12 нот на октаву.

Варто зауважити що дієз і бемоль взаємозамінні. Таким чином «До дієз» і «Ре бемоль» одна і та ж нота. Наряді з дієзом та бемолем є знак альтернації - бекар, який нівелює дію дієза або бемоля на обрану ноту.

Фізично нота – це механічне коливання на визначеній частоті. Існують такі системи та способи розподілу частот нот:

1. Піфагорійський стрій
2. Чистий лад
3. Рівномірно темперований лад

Розберу піфагорійський стрій, що був запропонований Піфагором близько 550 до н.е. Теорію такого ладу пов'язують з теорією піфагорейської школи гармоніки. Такий лад зазвичай представляється у вигляді квінтового чи квартового ланцюгу. Наприклад як ланцюг з 6 чистих квінт від основного звуку – ноти F:

$$F — C — G — D — A — E — H \tag{1.1}$$

Слід зауважити, що між нотами послідовності (1.1) інтервал є постійним і він дорівнює квінті. Так, піфагорійський стрій створив початки того, що сьогодні називається квінтовим кругом та використовується для написання послідовності нот. Оскільки інтервал квінти відповідає відношенню 2:3 або інтервалу в 5 ступенів, то між кожною сусідньою парою нот послідовності (1.1) пропущені 3 ноти. Зображення інтервалу чистої квінти можна побачити на Рисунок 1.1.

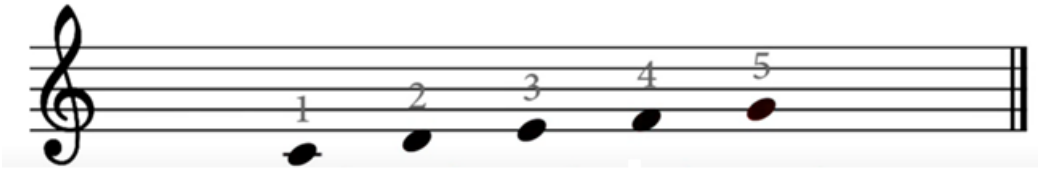

Рисунок 1.1 – Зображення інтервалу квінти на нотному стані

Приклад отримання частотного ряду нот однієї октави. Щоб отримати частоти нот октави достатньо мати на увазі декілька фактів. Перший – при збільшенні частоти в $\frac{3}{2}$ рази, нота пересувається на квінту, тобтно на 5 ступенів. Другий – множення чи ділення частоти ноти на 2 зсуває ноту на октаву вище чи нижче відповідно. Третій – щоб побудувати частотний ряд потрібна хоча б одна еталонна частота. Але в даній системі є один ньюанс. Якщо враховувати наявність нот з приставкою дієз або бімоль в октаві, то в інтервалі квінти має бути 8 з 12ти нот.

Нехай ми маємо еталонну частоту ноти Ля першої октави – 440Гц.

$$f_{A1} = 440\text{Гц}$$

Тоді після множення частоти на $\frac{3}{2}$ отримаємо ноту 5-ої ступені, а саме Мі другої октави. Щоб перенести ноту Мі на першу октаву поділемо її частоту на два.

$$f_{E1} = \frac{f_{E2}}{2} = \frac{f_{A1} * \frac{3}{2}}{2} = \frac{660}{2} = 330\text{Гц}$$

Застосовуючи той самий принцип до ноти Мі першої октави отримуємо ноту Сі першої октави.

$$f_{B1} = f_{E1} * \frac{3}{2} = 495\text{Гц}$$

Замість піднімання на 5 ступенів, можна спустити ноту на 5 ступенів поділивши на $\frac{3}{2}$.

$$f_{D1} = \frac{f_{A1}}{\frac{3}{2}} = 293{,}3\text{Гц}$$

Аналогічно підрахую інші ноти

$$f_{G1} = 2 * f_{G0} = 2 * \frac{f_{D1}}{\frac{3}{2}} = 391\text{Гц}$$

$$f_{C1} = \frac{f_{G1}}{\frac{3}{2}} = 260\text{Гц}$$

$$f_{F1} = 2 * f_{F0} = 2 * \frac{f_{C1}}{\frac{3}{2}} = 347{,}7\text{Гц}$$

Якщо розглянути квінту від ноти Сі, то бачимо що до ноти Фа є 5 ступенів, але немає 8ми з 12 нот, що було вказано як ньюанс вище. Це означає, що збільшивши частоту ноти Сі в $\frac{3}{2}$ отримуємо ноту «Фа дієз». Таким чином можемо знайти всі 12 нот октави.

$$f_{F\#1} = f_{B1} * \frac{3}{2} = 742{,}5\text{Гц}$$

Зменшуючи частоти в $\frac{3}{2}$ від ноти Фа отримуємо бімолі.

$$f_{Bb1} = \frac{f_{F1}}{\frac{3}{2}} = 462\text{Гц}$$

Так можна отримати частоти всіх нот. Щодо недоліків такої системи – це її незамкненість. Частоти нот збільшених на півтону не рівні наступній ноті змешеній на пів тону, тобто «До дієз» не тотожній до «Ре бімоль».

$$f_{C\#1} = 277{,}6\text{Гц} \neq f_{Db1} = 273{,}9\text{Гц}$$

Найпоширеніший з систем розподілу частот – рівномірно темперований лад. Така система будується значно простіше. Вона стверджує, що частоти нот інтервал між якими дорівнює полутону відносяться як 1: $\sqrt[12]{2}$. За стандарт частоти прийнята нота Ля першої октави з частотою 440Гц. На Таблиця 1.2 представлена відповідність нотам їх частот за системою рівномірно темперованого ладу.

Таблиця 1.2 – Частоти нот кожної октави

Ноты	Суббконтр-октава	Контр-октава	Большая	Малая	Первая	Вторая	Третья	Четвертая	Пятая
ДО	16,35	32,70	65,41	130,82	261,63	523,26	1046,52	2093,04	4186,08
ДО диез	17,32	34,65	69,30	138,59	277,18	554,36	1108,72	2217,44	4434,88
РЕ	18,35	36,71	73,42	146,83	293,66	587,32	1174,64	2349,28	4698,56
РЕ диез	19,45	38,89	77,78	155,57	311,13	622,26	1244,52	2489,04	4978,08
МИ	20,60	41,20	82,41	164,82	329,63	659,26	1318,52	2637,04	5274,08
ФА	21,83	43,65	87,31	174,62	349,23	698,46	1396,92	2793,84	5587,68
ФА диез	23,12	46,25	92,50	185,00	369,99	739,98	1479,96	2959,92	5919,84
СОЛЬ	24,50	49,00	98,00	196,00	392,00	784,00	1568,00	3136,00	6272,00
СОЛЬ диез	25,96	51,91	103,83	207,65	415,30	830,60	1661,20	3322,40	6644,80
ЛЯ	27,50	55,00	110,00	220,00	440,00	880,00	1760,00	3520,00	7040,00
ЛЯ диез	29,14	58,27	116,54	233,08	466,16	932,32	1864,64	3729,28	7458,56
СИ	30,87	61,74	123,47	246,94	493,88	987,76	1975,52	3951,04	7902,08

Неважко помітити, що зі збільшенням октави частоти нот збільшуються як показникова функція з основою рівною 2. На Рисунок 1.2 кожним кольором показана функціональна залежність частоти від октави для кожної окремої ноти. Синя найнища лінія - До, зелена найвища - Сі. З малюнка видно, що зі збільшенням октави збільшується абсолютне значення частотної відстані між нотами, хоча відносна відстань залишається сталою за принципом рівномірно темперованого ладу.

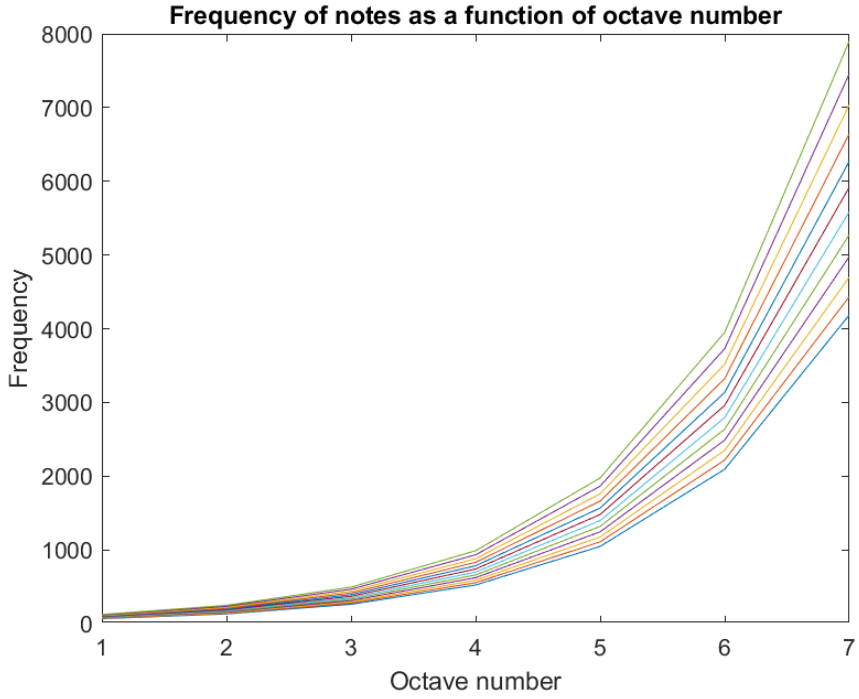

Рисунок 1.2 – Графік відповідності нотам октав частот

1.2 Фізичні аспекти тембру звучання

Ноти повністю визначаються частотами, але звучання ноти однієї і тієї ж частоти на різних інструментах звучить з різним забарвленням. Це називається тембром.

Почнемо з того, що аби музичний інструмент видавав звук ноти, його певна частина повинна задовільняти фізичних умов до механічних коливань на частоті ноти.

У струнних інструментів такими елементами є струни. Їх довжина визначає довжину половини довжини хвилі механічного коливання яку вони можуть збудити. В той час як натяг, матеріал та інші характеристики струни визначають швидкість поширення цієї хвилі. Таким чином струни однакової довжини збуджують звукові коливання різних частот.

У духових музичних інструментів, таких як орган, використовуються труби. Довжина труби підбирається так, що б у неї вклалася повна довжина стоячої хвилі що буде збуджуватися. Повітряний потік збуджує в трубі механічні коливання.

У таких інструментах, як кларнет, частота задається закриттям / відкриттям клапанів, що визначає довжину стоячої хвилі всередині труби, що показано на Рисунок 1.3.

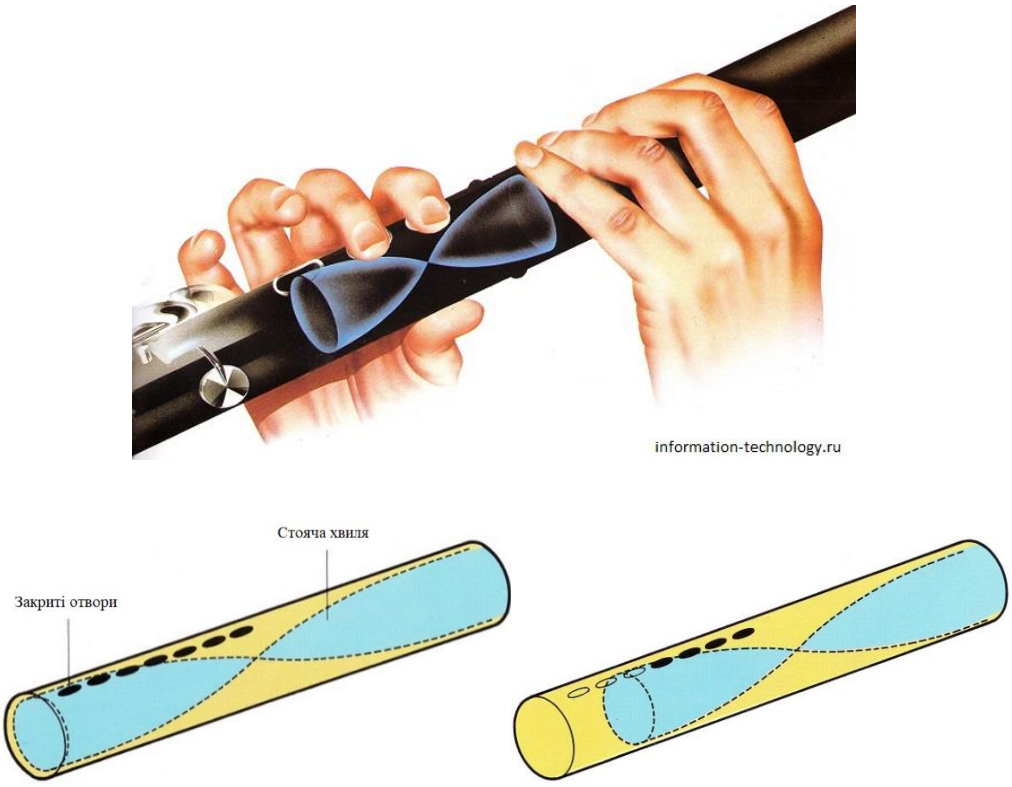

Рисунок 1.3 – Формування стоячої хвилі в трубі в залежності від стану клапонів[1]

Розглянемо поняття тембру більш докладно на прикладі струнних інструментів з точки зору більшої наочності. Струна є довгий відрізок гнучкого матеріалу, що знаходиться в натягнутому стані, завдяки чому може вільно коливатися. Перш за все треба відмітити, що у струни є 2 вузла. На гітарі ці вузли розташовані на нижньому і верхньому поріжку відповідно. Ці ділянки гітари можна побачити на Рисунок 1.4.

Рисунок 1.4 – Конструктивні частини класичної гітари

Таким чином при збудженні струни вона буде коливатися такими механічними хвилями, вузли яких лягають в вузли на порожках. Математичні підстави цього факту можна знайти з посібника математичної фізики в розділі «Свободные колебания струны с закрепленными концами»[2]. Таким чином, самий низькочастотний звук, що видає збуджена струна, виходить у хвилі з найбільшою довжиною хвилі. І половина довжини такої хвилі дорівнює довжині всієї струни. Такий тон називають основним, у нього два вузла і максимальна амплітуда коливання по середині струни. На Рисунок 1.5 можна подивитися як змінюється стан струни при коливаннях основного тону з часом.

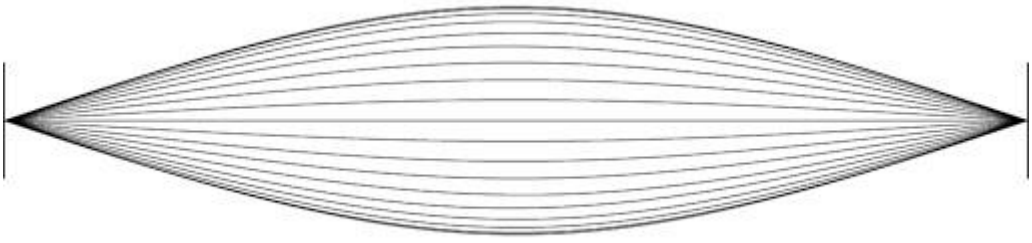

Рисунок 1.5 – Коливання струни на основному тону в перерізах часу

Окрім основного тону струна коливається усіма іншими хвилями у яких відношення довжини струни і половини довжини хвилі дають ціле число. Нескладно прийти до того, що ці хвилі будуть кратні по частоті основного тону, що можна побачити на Рисунок 1.6.

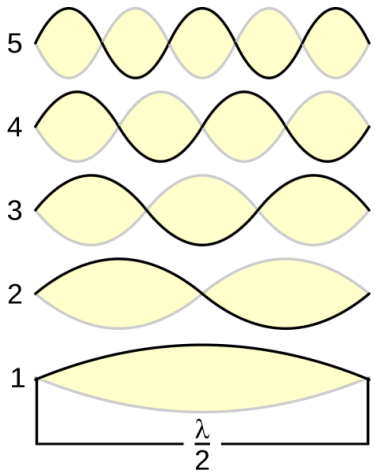

Рисунок 1.6 – Зображення довжин хвиль, що збуджуються в струні

Тони, що є кратними по частоті основному тону називають обертонами або гармоніками. Вони мають деякі властивості: частотна відстань між обертонами рівна частоті основного тону, зазвичай амплітуда основного тону є найвища.

Так як розподіл амплітуд обертонів щодо основного тону є індивідуальним для кожного інструменту, отримуємо тембри інструментів. Так, на Рисунок 1.7 можна побачити розподіл амплітуд обертонів деяких інструментів.

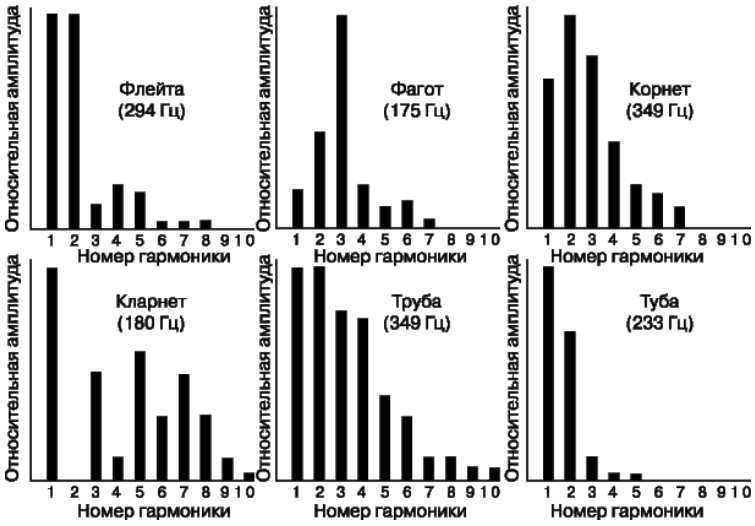

Рисунок 1.7 – Спектральні показники тембру флейти, фаготу, корнету, кларнету, труби та туби

Так само варто згадати що і сам інструмент може посилювати деякі обертони. Наприклад якщо дека скрипки резонує з певним тоном - він буде посилен в спектрі. Картини коливань верхньої і нижньої деки можна побачити на Рисунок 1.8.

Рисунок 1.8 – Форми коливань верхньої і нижньої деки скрипки[3]

Вирізи в деці скрипки, звані ефи, створені саме для посилення явища резонансу корпусу. У той же час смужки дерева, звані обичайками, які скріплюють верхню і нижню площину деки, утворюють особливі форми. Геометрія склепінь, їх товщина і розподіл так само впливає на силу звуку і на тембр інструмента. Конструкцію скрипки можна побачити на Рисунок 1.9.

Рисунок 1.9 – Конструктивні елементи скрипки

Розподіл амплітуд не залишається постійним у часі, без вимушеної сили, такої як постійний потік повітря, чи постіне пересування смичка. Відношення обертонів до основного тону може змінюватися в часі. Так, на Рисунок 1.10 можна подивитися на часові спектрограми флейти і гобоя, а на Рисунок 1.11 на відношення обертонів гітарної струни в часі.

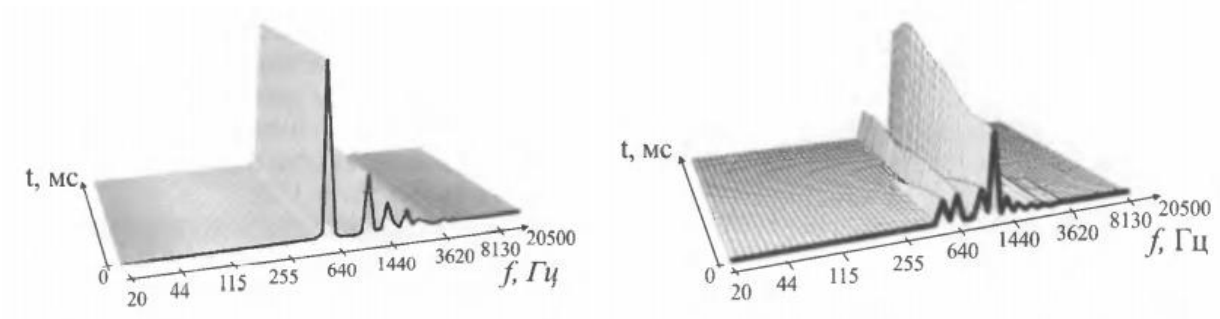

Рисунок 1.10 – Часові спектрограмми зліва направо: флейта, гобой[3]

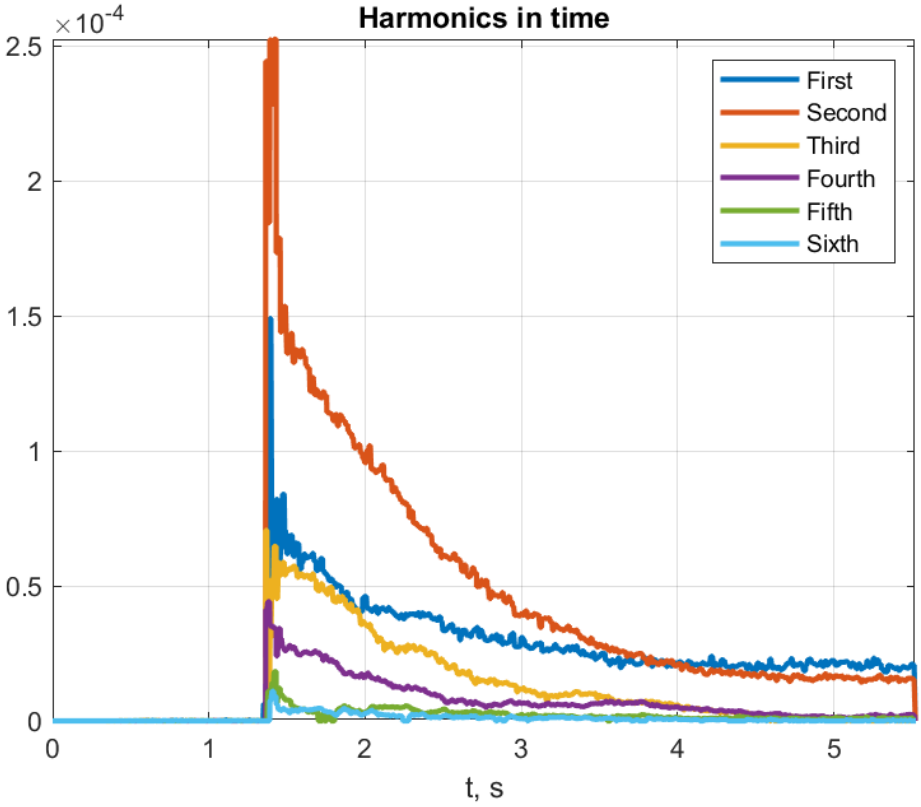

Рисунок 1.11 – Відношення обертонів до основнго тону в часі при збуджені струни гітари

2 МЕТОДИ ВИРІШЕННЯ ЗАДАЧ КЛАСИФІКАЦІЇ

2.1 Основи класифікації

Основна задача класифікації є задача розділення об’єктів за певними характеристиками з множини об’єктів на певні класи. Класифікатором називається алгоритм, що навчаючись на скінченній группі об’єктів з заздалегіть відомими классами, може класифікувати довільний об’єкт з вхідної множини. Таку группу об’єктів для навчання називають навчальною вибіркою. Класифікація – процес присвоювання заданному об’єкту класу з множини класів.

В математичній статистиці задачі класифікації називаються також задачами дискретного аналізу. В машинному навчанні завдання класифікації вирішується, як правило, за допомогою методів штучної нейронної мережі при постановці експеримента у вигляді навчання з учителем.

Математична модель – абстракція реальності, де елементи, що цікавлять дослідника замінени відповідними відносинами між математичними об’єектами. Математичні моделі в описі яких використовуються випадкові величини називають ймовірнисними або стохастичними. Будь-яка модель є спрощеним представленням реальності. Мистецтво моделювання полягає в знанні того, що де і як можна спростити.

Нехай X - множина об'єктів, Y - множина номерів класів. Тоді існує певна невідома залежність $y^*: X \to Y$, значення якої відомі тільки для скінченної вибірки для навчання $X^m = \{(x_1, y_1), \dots, (x_m, y_m)\}$. Задачею класифікації є побудова алгоритма $a: X \to Y$, що здатен класифікувати довільный об’єкт $x \in X$.

Характеристикою є відображення $f: X \to D_f$, де D_f – множина допустимих значень характеристик. Об’єкт може бути представленим через вектор характеристик $f_1, \dots, f_n$ так, що $x = (f_1(x), \dots, f_n(x))$. Характеристики можна ототожнювати із самими об'єктами. Нехай кожен об’єкт множини X має n ознак $x = (f_1(x), \dots, f_n(x))$, а множина класів Y містить k класів $y_j \in Y, j = 0, \dots, k$.

Так, характеристикою фотографії може бути матриця пікселей, що містять значення яскравості для кожного кольору. Слово може бути вредставлено як

бінарний вектор характеристик, в якому одиниця стоїть на місці відповідному до слова. Такий метод незручний в використанні, бо потребує вектори довжиною в повну кількість слів словника для кожного слова. Альтернативним та більш бажаним є представлення слова в просторі $\mathbb{R}^n$, де n значно менше кількості слів в словнику. Так слова розділяються на під-змісти, синоніми знаходяться ближче один до одного в n мірному просторі, антоніми знаходяться далі.

Основна функціональність штучних нейронних мемрежей загалом поділяється на класифікацію та регресію. Нейронні мережі не використовують апріорних знань про об’єкти для класифікації, натомість мережа методом навчання з тренеровочної вибірки дізнається про закономірності між об’єктами. Такі класифікатори відрізняються від інших рекордною точністю та універсальністью однієї мережі до множини проблем класифікації. Так, система що класифікує фотографії автомобілів може бути змінена на систему, що класифікує фотографії тварин тільки зміною тренеровочної вибірки та зміною назв класів. Недоліками такого класифікатора може бути потреба в великій бази данних для навчання та тестування, великі часові затрати на навчання.

Є багато методів класифікацій, роздивимось найбільш популярні.

2.2 Байесівска класифікація. Гаусовський наївний баєсів класифікатор

Почнемо з ймовірнісного формулювання задачі класифікації. Припускається що множина пар «об'єкт, клас» $X \times Y$ є ймовірнісним простором з невідомою ймовірнісною мірою P. Є скінченна навчальна вибірка спостережень $X^m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, згенерована згідно з ймовірнісною мірою P. Необхідно побудувати алгоритм $a: X \to Y$, здатний класифікувати довільный об’єкт $x \in X$.

Байесівський класифікатор – йомвірнісний класифікатор, що використовує теорему Байеса для визначення ймовірності приналежності об’єкта до одного з класів. Наївний байесівський класифікатор (надалі БК) працює при припущенні незалежності ознак об’єкта $f_i \neq g(f_j) \; \forall \, i, j \in 0 \ldots n, i \neq j$. Наївний БК іноді

працює краще ніж нейронні мережі, що спостерігається в випадках з обеженою вибіркою.

Тобто, якщо аналізуючи ознаки об'єкта можна однозначно визначити, до якого класу він належить, байєсів класифікатор повідомить ймовірність приналежності до цього класу. У проміжних же випадках, коли об’єкт може з різною ймовірністю належати до різних класів, результатом роботи класифікатора буде вектор, компоненти якого є ймовірностями приналежності до того чи іншого класу.

Як видно з означення, ідеальний байєсів класифікатор є оптимальним алгоритмом класифікування. Його результат не може бути поліпшений, тому що у всіх випадках, коли можлива однозначна відповідь, він його дасть — а в тих випадках, коли відповідь неоднозначна, результат кількісно характеризує міру цієї неоднозначності.

Разом з тим, в оптимальності криється і основний недолік ідеального байєсового класифікатора: для його побудови потрібна вибірка, що містить всі можливі комбінації ознак — а розмір такої вибірки експоненціально зростає із зростанням числа ознак. Для подолання описаної вище проблеми на практиці використовують наївний БК — класифікатор, побудований на основі припущення про незалежність змінних, обмежившись лише впливом кожної змінної окремо на приналежність образу до одного з класів.

Байєсівський класифікатор заснований на принципі максимуму апостеріорної ймовірності. Для об'єкта класифікації обчислюються функції правдоподібності кожного з класів, по ним обчислюються апостеріорні ймовірності класів. Об'єкт відноситься до того класу, для якого апостериорная ймовірність $P(y_j|x)$ максимальна.

Приймаючи Y за множину класів, де y_j один клас, X за множину об’єктів, x_q за об’єкт, а $a(x)$ за алгоритм класифікації, рішення про належність об’єкту до класу може бути визначено за формулою (2.1)

$$a(x_q) = y_q = arg \max_{j \in \{0,..,k\}} P(y_j|x_q) \quad (2.1)$$

Іншими словами алгоритм зводиться до максимуму апостеріорної ймовірності. Клас, ймовірність пренадлежності до якого буде максимальним зі всієї множини класів для конкретного об'єкту x_q буде дорівнювати y_q.

Згідно з теоремою Байеса вираз апостеріорної ймовірності вище може бути переписано в більш явному вигляді (2.2).

$$P(y_j|\boldsymbol{x}_q) = \frac{P(\boldsymbol{x}_q|y_j)P(y_j)}{P(\boldsymbol{x}_q)} \tag{2.2}$$

де $P(\boldsymbol{x}_q|y_j)$ – ймовірність що у об'єкта класу y_j будуть ознаки $\boldsymbol{x}_q$;

$P(y_j)$ – апріорна ймовірність зустрічі классу y_j серед множини об'єктів;

$P(\boldsymbol{x}_q)$ – апріорна ймовірність з якою зустрічається такий набір ознак.

При пошуку максимуму з формули (2.1) константами з виразу апостеріорної ймовірності (2.2) можна знехтувати. Так, $P(\boldsymbol{x}_q)$ не залежить від класу, отже може бути відкинутим.

Застосовуючи наївну БК припустимо, що ознаки $x_{\bar{\imath}}, \bar{\imath} = 0 \dots N-1$, не залежать одне від одного. Тоді чисельник формули (2.2) можна спростити.

$$P(y_j|\boldsymbol{x}_q) = P(y_j) * \prod_{i=1}^{N} P(x_{q,i}|y_j) \tag{2.3}$$

де $\boldsymbol{x}_q = (x_{q1}, x_{q2}, \dots, x_{qN})$ – вектор ознак, що є взаємозамінним поняттям до об'єкту.

А що б уникнути надмірно малих чисел при множенні великого числа ймовірностей – праву частину можна прологарифмувати, так як максимум від цього не зміниться.

$$P(y_j|x_q) = \log P(y_j) + \sum_{i=1}^{N} logP(x_{q,i}|y_j) \tag{2.4}$$

Неперервні залежності, такі як $P(x_{q,i}|y_j)$, як правило, оцінюються через нормальний розподіл. Тобто на практиці БК використовує підходи Гаусовського класифікатора[5] (надалі ГК). Принцип ГК полягає в тому, що залежності $P(x_{q,i}|y_j)$ нам відомі заздалегідь і ми знаємо що їх форма розподілу – гаусівська

(2.5). Такий підхід є популярним, так як багато природніх процесів, що можуть класифікуватись мають нормальний розподіл параметрів.

$$P(x_{q,i}|y_j) = N\left(m_{y_{j,i}}, \sigma_{y_{j,i}}\right) = \frac{\exp\left(-\frac{\left(x_{q,i} - m_{y_{j,i}}\right)^2}{2 * \sigma_{y_{j,i}}^2}\right)}{\sqrt{2\pi\sigma_{y_{j,i}}}} \quad (2.5)$$

де m_{y_j} – математичне очікування класу y_j для ознаки $x_{q,i}$;

$\sigma_{y_{j,i}}$ – середнеквадратичне відхилення тої ж ознаки.

В якості математичного очікування і дисперсії обчислюються середнє арифметичне (2.6) і середнє квадратичне відхилення (2.7) відповідно.

$$m_{y_{j,i}} = \frac{1}{M}\sum_{p=1}^{M} x_{p,i} \quad (2.6)$$

$$\sigma_{y_{j,i}} = \frac{1}{M}\sum_{p=1}^{M}\left(x_{p,i} - m_{y_{j,i}}\right)^2 \quad (2.7)$$

де M – кількість об’єктів з навчальної вибірки класу y_j;

$x_{p,i}$ – і-та ознака р-го об’єкту.

Відповідно для класифікації Наївним БК з використанням принципів ГК потрібно буде зберігти $k \times n$ математичних очікувань та середньоквадратічніх відхілень. Тобто на кожен клас з множени Y потрібно зберігти параметри нормального розподілу для кожної ознаки $x_{q,i}$.

Такий метод широко застосовується для класифікації об’єктів, ознаки яких слабо зв’язані один з одним. Тобто, для тих, ознаки яких можна прийняти незалежними без особливих втрат точнсті.

Розглянемо приклад класифікації. Для навчання класифікатора можна взяти базу данних ознак вин, таких як процент алкоголю, вміст яблучної кислоти, вміст і лужність золи та іншні[6]. Згідно статті[7] видно, що в порівнянні з деревом ухвалення рішення та методом k-найближчих сусідів наївний БК показав, що він

класифікує з значно вищою точністю та в значно меншій мірі залежить від параметрів оптимізації, хоча займає більше часу на обрахунки.

2.3 Метод опороних векторів

Метод опорних векторів (Support vector machines SVM) є одним з класифікаційних методів, що наряді з БК є представником класичного машинного навчання з учителем. Данний метод є однією з найбільш популярних методологій навчання по прецедентах, запропонованої В. Н. Вапніком в 1963 році.[4]

В класичному представлені метод застосовується для бінарної класифікації, тобто коли об’єкти поділені на 2 класи. Метод класифікації працює з n-мірними вимірами. В загальному вигляді, якщо об’єкт описується n характеристиками, то такий об’єкт можна представити як точку в n вимірному просторі. На Рисунок 2.1 показан приклад зображення об’єктів двох класів, що описуються двома ознаками – $(x1, x2)$. Відповідно цифрові значення ознак можуть виступати координатами точок як об’єктів.

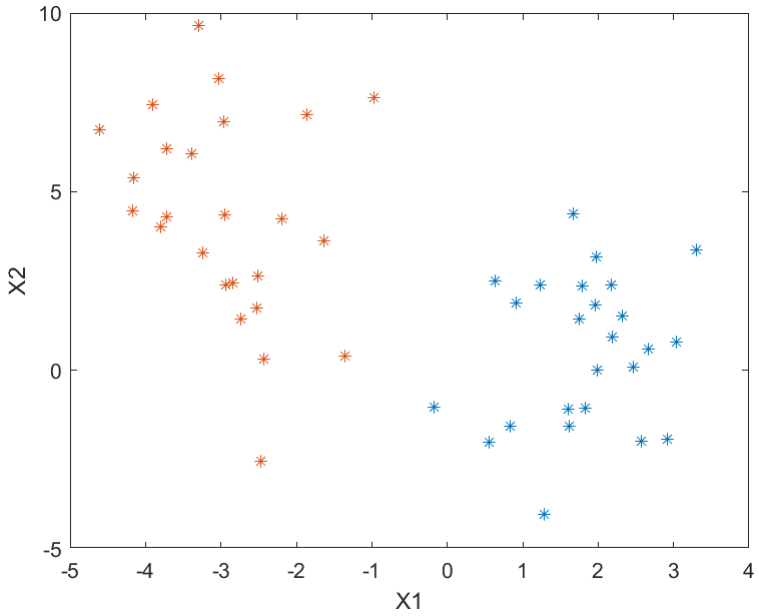

Рисунок 2.1 – Приклад зображення об’єкту з двома ознаками як точки на площині

Метод опорних векторів полягає в тому, що регресивно може бути побудована така площина, що буде рівновіддалена до найближчих точок кожного

классу. Вона буде розділяти точки n вимірного простору на два півпростори, що може бути використано для класифікації нових об’єктів.

Для заданого набору тренувальних зразків, кожен із яких відмічено як належний до однієї чи іншої з двох категорій, алгоритм тренування SVM будує модель, яка відносить нові зразки до однієї чи іншої категорії, роблячи це неймовірнісним бінарним лінійним класифікатором.

В загальному вигляді, для n вимірного об’єкту класифікатор SVM будує $n - 1$ вимірну гіперплощину, що розділяє точки і є рівновіддаленою до двох классів. По-перше слід сказати, що не всі дані можуть бути розділені гіперплощиною. По-друге, якщо така площина існує, скоріш за все існує множина можливих гіперплощин що будуть розділяти дані. Так підходимо до визначення критерію вибору гіперплощини.

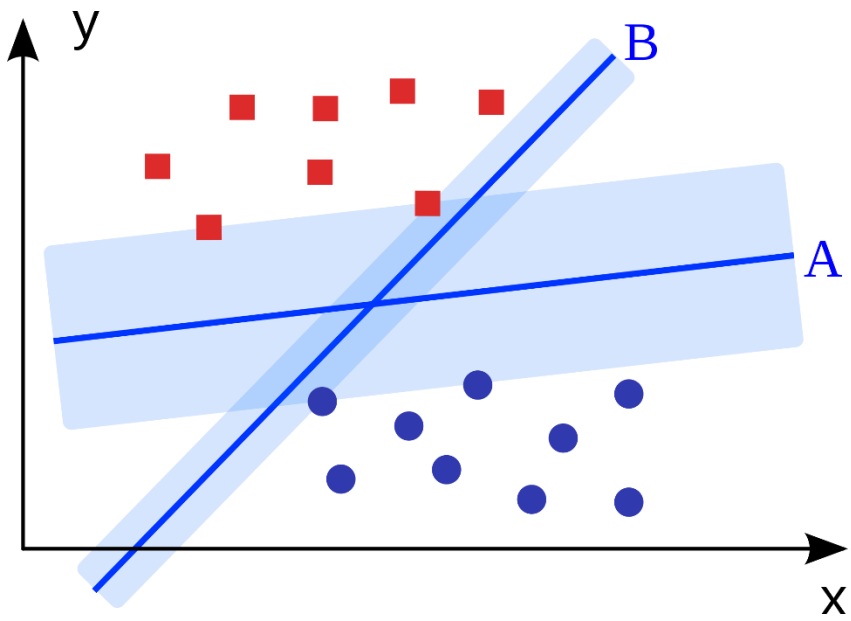

Рисунок 2.2 – Зображення оптимальної (A) та неоптимальної (B) роздільної гіперплощини

Одним із варіантів розумного вибору найкращої гіперплощини є такий, який пропонує найбільший проміжок, або розділення між двома класами. Тож обирається гіперплощина таким чином, щоби відстань від неї до найближчих точок даних з кожного боку була максимальною. Така гіперплощина, якщо вона

існує, відома як максимально розділова гіперплощина, а лінійний класифікатор, що вона його визначає, — як максимально розділовий класифікатор. На Рисунок 2.2 можна побачити данні двох класів – червоні та сині точки, що описуються двома ознаками - (x, y). На рисунку зображені дві розділяючі гіперплощини, А та В, від яких побудовані проміжки до найближчих об'єктів кожного классу. Оскільки проміжок від гіперплощини А більший – таке розділення є оптимальним для данного класифікатора.

Алгоритм розглянутий вище є алгоритм максимально розділової гіперплощини, запропонований Володимиром Вапником у 1963 році. Такий підхід працює тільки якщо тренувальні дані є лінійно роздільними. Такий підхід називається жорстким розділенням.

Розглянемо детальніше математичну основу такого методу. Нехай в нас є тренувальний набір даних з m точок вигляду $(x_1, y_1), \dots, (x_m, y_m)$, де y_i приймає значення 1 або -1, що відповідає першому або другому классу, а кожна точка x_i є -мірним вектором ознак. Задача методу знайти максимально розділову гіперплощину яка відділяє групу точок x_i, для яких $y_i = 1$, від групи точок, для яких $y_i = -1$, і визначається таким чином, що відстань між цією гіперплощиною та найближчою точкою x_i з кожної з груп є максимальною. Гіперплощину можна записати як рівняння (2.8)

$$w * x - b = 0 \tag{2.8}$$

де w – вектор нормалі до гіперплощини;

$\frac{b}{\|w\|}$ – параметр що визначає зсув гіперплощини відносно початку координат.

Щоб визначити проміжки, або margin, будують дві паралельні гіперплощини до роздільної так, щоб вони мали найближчу точку одного з классів. Рівнння для таких паралельних гіперплощин показані в формулі (2.9). Графік з розділенням та вказівками на паралельні площини можна побачити на Рисунок 2.3.

$$w * x - b = 1$$
$$w * x - b = -1 \tag{2.9}$$

З геометричної точки зору, відстанню між цими двома гіперплощинами є $\frac{2}{\|w\|}$, тож для максимізації відстані між ними нам треба мінімізувати $\|w\|$.

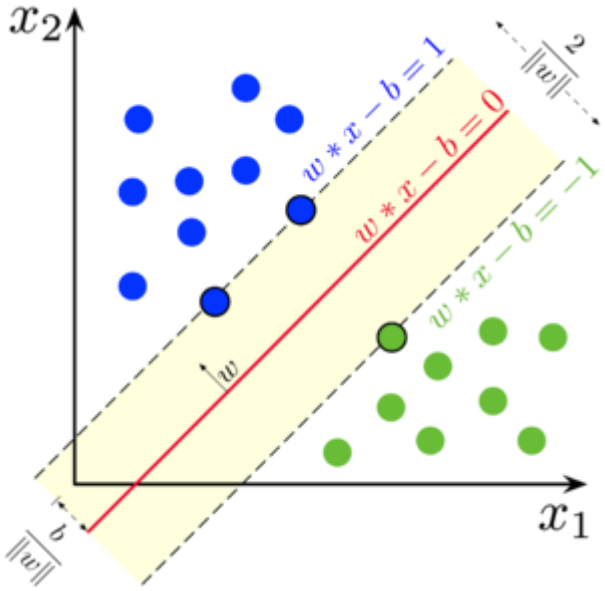

Рисунок 2.3 – Оптимальна роздільна гіперплощина в $\mathbb{R}^2$ та вказівки на паралельні гіперплощини для визначення проміжку [4]

Оскільки ми також маємо завадити потраплянню точок даних до розділення, ми додаємо наступне обмеження

$$y_i(w * x_i - b) \geq 1, i \in \{1, \dots, m\} \quad (2.10)$$

Ці обмеження стверджують, що кожна точка даних мусить лежати з правильного боку розділення.

Очевидним, але важливим наслідком цього геометричного опису є те, що максимально розділова гіперплощина повністю визначається тими x_i, які лежать найближче до неї. Ці x_i називають опорними векторами.

Також очевидним є те, що не всі дані можуть бути розділені лінійною функцією, тобто гіперплощиною. Для розширення SVM на випадки, в яких дані не є лінійно роздільними вводять поняття завісної функції втрат, що використовується для навчання класифікаторів в машинному навчанні. Оскільки класифікатор є бінарним, функцію втрат можно записати як

$$\ell = \max(0, 1 - y_i(w * x_i - b))$$

Така функція помилки поверне нуль в тому випадку, якщо нерівність (2.10) задовільняється. Для інших випадків значення помилки буде пропорційним до відстані від розділення. В такому разі мінімізувати треба функцію (2.11), а не $\|w\|$.

$$\left[\frac{1}{n}\sum_{i=0}^{n}\max(0, 1 - y_i(w * x_i - b))\right] + \lambda\|w\|^2 \qquad (2.11)$$

де λ – параметр що визначає важливість між двома критеріями – збільшення проміжку та задовільнення умови (2.10).

Одним із методів класифікації данних, що не є лінійно розділенними – перехід до більш вимірного простору, де класифікатору буде зручніше розділити дані гіперплощиною.

Такий підхід є дуже популярним, та часто зустрічається на практиці. Роздивимось приклад даних, що не є лінійно розділенними. Такі дані зображені на Рисунок 2.4. Для того, щоб перетворити такі дані в ті, що будуть лінійно розділенними в вищому вимірі використовують ядрову функцію $k(x, y)$. Тоді лінійна гіперплощина у вищому вимірі ознак може бути нелінійною функцією в первинному просторі де потребужться класифікувати дані.

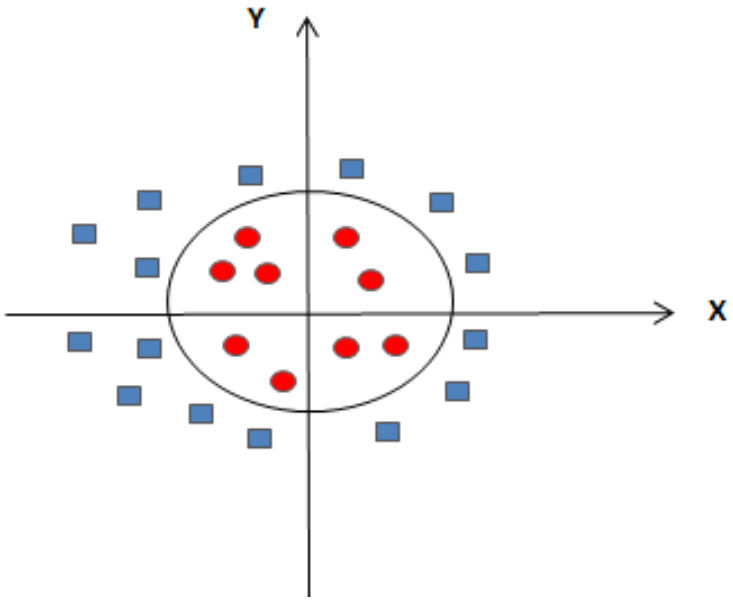

Рисунок 2.4 – Приклад нерозділених лінійно даних [8]

Найпоширеніші ядрові функції :

- Поліноміальне однорідне. $k(x_i, x_j) = (x_i \cdot x_j)^d$
- Поліноміальне неоднорідне. $k(x_i, x_j) = (x_i \cdot x_j + 1)^d$
- Ґаусова радіально-базисна функція. $k(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right)$, для $\gamma > 0$.
- Сигмоїдне (гіперболічний тангенс). $k(x_i, x_j) = \tanh(kx_i \cdot x_j + c)$, для деяких $k > 0$ та $c < 0$

де (x_i, x_j) – відповідні вектори ознак.

Так, застосовуючи однорідну поліноміальну ядрову функцію для збільшення розмірності данних показаних на Рисунок 2.4, маємо дані що є лінійно розділеними. Результат схожого перетворення даних зі збільшенням виміру ознак та розділення гіперплощиною можна побачити на Рисунок 2.5.

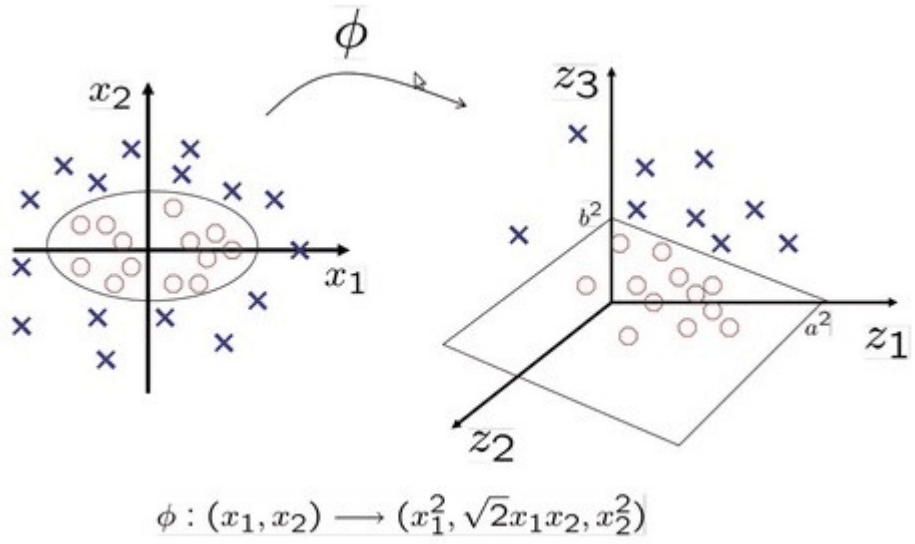

Рисунок 2.5 – Результат збільшення розмірності данних задля відокремлення данних гіперплощиною [8]

Ще одна проблема що може виникнути при використанні данного методу – це багатокласcова классифікація. Для застосування SVM до задачі багатокласового розділення данних можна використати одне з двох популярних рішень які називаються – «один-проти-всіх» та «один-проти-одного».

В стратегії «один-проти-всіх» для Z класів використовують Z послідовних бінарних класифікаторів які визначають чи дані належать до певного класу, чи не належать. Припустимо, що існує 4 класи, нехай вони називаються А, Б, С і Д. Тоді будуються 4 бінарні SVM класифікатори: «А» проти «не А», «Б» проти «не Б» і так далі. Обирається додатній клас, що має найбільшу відсталь до гіперплощини.

Емпіричні досліди Каі-Бо Даун та Сатія Керті показують, що саме такий метод дає найкращі результати. [9]

Стратегія «один-проти-одного» заключається в тому, що будуються пари класифікаторів з всіма можливими парами класів. Якщо розглядати той самий приклад з чотирма класами, то буде побудовано 10 класифікаторів: «А» проти «Б», «А» проти «В», «А» проти «С» і так далі. Відповідно кількість класифікаторів в такому випадку буде визначатися по формулі нижче, що значно дає значно більше класифікаторів ніж попередній метод.

$$\sum_{i=1}^{N} i - 1$$

Проте, Райан Ріфкін та Альдебаро Клаутау мають наукову статтю, що доказує більшу ефективність останнього методу. [10]

3 НАЛАШТУВАННЯ МЕТОДІВ КЛАССИФІКАЦІЇ

3.1 Вибір параметрів вхідного сигналу для класифікації

3.1.1 Домен сигналу

Розглядаючи детальніше задачу класифікації акустичних сигналів і приймаючи до уваги фізичні особливості звучання ноти можна помітити, що саме частота звуку визначить його ноту. В такому випадку доречніше перейти з часового домену в частотний і аналізувати ділянку сигналу по його спектру.

Є кілька методів переходу в частотний домен, наприклад використовуючи Швидке Перетворення Фур'є (FFT) або вейвлет-перетворення. Вейвлет перетворення дають часово-частотну картину на якій можна чітко побачити в який момент які частоти з'явилися, що вирішило б проблему захоплення ділянки сигналу з двома нотами. Однак, в більшості випадків віконне перетворення Фур'є може замінити більш складне для аналізу вейвлет перетворення. Отже треба визначити довжину вікна віконного перетворення Фур'є – $NFFT$.

Швидке перетворення Фур'є – це алгоритм, який дозволяє скоротити час розрахунку дискретного перетворення Фур'є. Тоді як звичайне дискретне перетворення Фур'є виконує розрахунки за час $O(N^2)$, швидке перетворення Фур'є справляється за $O(NlogN)$.

3.1.2 Довжина сигналу

Нагадаю, що мета даної класифікації – визначити кожну ноту композиції. Виходячи з цього можна визначити необхідну довжину часової вирізки як таку, в яка охопить найкоротшу ноту композиції.

Основними длительностями нот є ціла (біла нота без штилю) і її половинні поділу: половина (біла зі штилем), чверть (чорна зі штилем), восьма (чорна зі штилем і одним прапором), шістнадцята (чорна зі штилем і двома прапорами), тридцять другий (чорна зі штилем і трьома прапорами) і т. д. Інші ноти, такі як шістедят четверта, зустрічаються дуже рідко. Зображення таких нот можна подивитись на Рисунок 3.1.

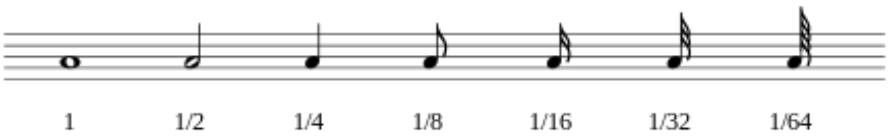

Рисунок 3.1 – Зображення нот різної тривалості

Тривалість ноти є відносною велечиною, поки не заданий bpm композиції. Візьмемо стандартний темп 4/4 для аналізу, що відповідає 4 долям в такті, кожна доля якої відповідає ноті з тривалістью одна четверта. В такому випадку ціла нота буде мати 4 долі і одна ціла нота може заповнювати цілий такт. Темп, або bpm (beats per minute, удари в хвилину) характеризує скільки четвертних нот буде зіграно за хвилину. Стандартним значенням є 120, що використовується повсякчас коли автор не указує bpm в композиції. Максимальний темп, що відповідає дуже швидкій композиції, має значення біля 216.

Отже, оскільки задачею класифікації є визначення нот композиції, визначемо мінімальну тривалість ноти, після чого визначемо мінімально необхідну довжину акустичного сигналу. Проведу підрахунки для максимального темпу.

Припускаючи що найшвидшею нотою композиції є одна тридцять друга маємо наструмну тривалість ноти

$$t_4 = \frac{1}{bpm/60} = 277мс$$

$$t_{32} = \frac{t_4}{8} = \ 35мс$$

де t_4 – тривалість четвертної ноти;

t_{32} – тривалість тридцять другої ноти.

У звукозаписуючих пристроїв є ряд стандартних частот дискретизації: 44,1 кГц, 48 кГц та 96 кГц. Оскільки ноти вище четвертої октави зустрічаються рідко, бо вони менш яскраві по тембру, то візьмемо для підрахунків найнижчу частоту дискретизації. Тоді отримаємо, що найкоротша нота буде записана кількістю відліків

$$N_s^* = t_{32} * F_s = 1653$$

Приймаючи до уваги особливості роботи fft, кількість відліків сигналу буде доповнена нулями так, що би отпримати степінь двійки. Задля прискорення процесу fft можемо змінити вікно на $N_s = 1024$ так, щоб у вікно точно попала найкоротша нота.

3.1.3 Використання інтерполяції спектру

Маючи кількість відліків найменшої ноти і частоту дискретизації $F_s = 44.1$кГц можемо знайти частотний крок в спектрі.

$$df_s^* = \frac{F_s}{N_s^*} = 27\text{Гц}$$

Перше, що можна помітити - це частотний крок, який більший ніж крок між деякими нотами. З Таблиця 1.2 можна прорахувати, що крок між нотами змінюється з 4Гц до 444Гц починаючи з великої і закінчуючи п'ятої октавою.

В такому випадку можемо спостерігати ефект злиття сусідніх нот низьких октав. Тобто різні ноти будуть збуджувати одні і ті самі спектральні відліки, що зробить ноти нерозрізними між собою. Згідно теореми Котельнікова, інформація про коливання не втрачається, ящко частота найквіста більша за максимальну частоту спектра. Оскільки ноти низьких октав мають частоту нижчу за половину частоти дискретизації, то інформацію зі спектру можна витягнути. Цю проблему може вирішити інтерполяція спектру.

Інтерполяцію спектра можна отримати нескладним чином - додавши нулів в кінець сигналу, який буде перетворений в фур'є. Таким чином ми штучно збільшуємо кількість відліків N_s^*. Однак у такого методу є свої недоліки. Головний з них - це пульсації в спектрі. Вони пояснюються тим, що еквівалентною дією до додавання нулів є застосування віконного перетворення Фур'є до великого сигналу, в якому прямокутної функцією Хемминга, шляхом перемноження, був виділений наш аналізований сигнал.

Відповідно до теореми спектрального аналізу, якщо в часовій області сигнали перемножити, то в спектральній області спектри цих сигналів згорнуться. Виходить що від кожної синусоїдальної гармоніки, що була зображена дельта-

імпульсом в спектрі тепер буде розходитися функція сінк, що може перекрити низькорівневі високочастотні гармоніки пульсаціям. Злиття спектру сусідній нот нижніх октав та результат інтерполяції спектру можна побачити на Рисунок 3.2.

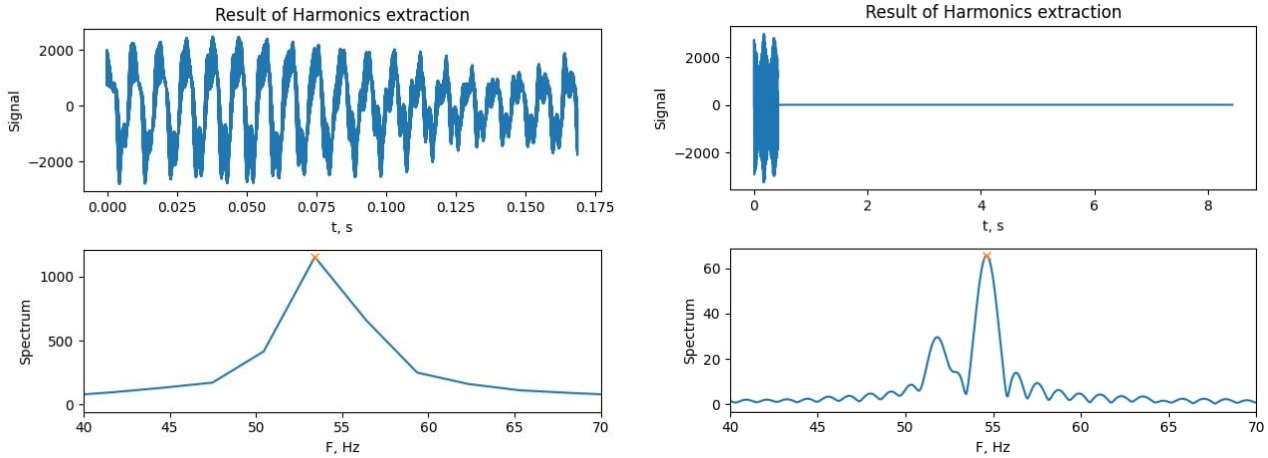

Рисунок 3.2 – Сигнали та спектри звучання двох сусідніх низькочастотних нот. Перший випадок - акустичний сигнал в якому закладено звук двох нот узятий з такою довжиною, щоб частотний крок перевищував відстань між нотами в спектрі. Другий випадок - той же сигнал був доповнений нулями в часовій області, що дозволило розрізнити спектральні складові нот.

Вибір кількості нулів, що мають бути добавлені в кінець сигналу має єдину вимогу – інтерполювати спектр так, що б кожна нота була розділена. Оскільки найменша частотна відстань буде між нотами найнижчої октави, а саме між до та до дієз суб-контр октави, 16.35Гц та 17.32Гц, було вирішено отримати частотний крок рівний 1Гц. Для цього було вирішено додавати стільки нулів, щоб кількість відліків сигналу біла рівна частоті дискретизації.

У такому випадку кількість нульових відліків, що має бути добавлена до акустичного сигналу

$$N_0 = f_s - 1653 = 42447$$

3.2 Емпіричні методи класифікації

У даній роботі в якості об'єктів для класифікації представлені акустичні сигнали. Вектором характеристик сигналу може бути як відліки сигналу в часовому домені, відліки спектру сигналу чи будь-яке інше доцільне цифрове представлення об’єкту. Нагадаю, що нота може бути повністю схаратеризована базовою частотою, відповідну частоту для кожної ноти можна подивитися на Таблиця 1.2. З точки зору повної видповідності частота-нота доцільно обрати представленя об’єкта через спектральні відліки.

Виходячи з фізичних основ звучання нот музичними інструментами можемо побачити закономірність, що є базова, найнища по частоті гармоніка, і вищі кратні їй по частоті. Оскільки зі збільшенням частоти коливання, наприклад струни, збільшується коефіцієнт затухання, то можемо припустити що базова частота в спектрі буде мати найбільше значення, що дозволяє використати найбільш простий метод класифікації – пошук максимуму по спектру з подальшим визначенням індексу.

Однак, в багатьох інструментах, наприклад в гітарі, струни збуджуються не в місці максимуму базової частоти. Це обусловлено тим, що в такому разі всі парні гармоніки не будуть збуджені та звучання буде менш яскравим, так як всі парні гармоніки мають вузол в центрі струни. Тому резонаторний отвір, що відповідає бажаному місцю для збудженню струни, знаходиться приблизно на ¼ довжини струни від нижнього поріжку. Через це найбільшу ініціюючу амплітуду отримують 2га та 3тя гармоніка, що і можна побачити на Рисунок 1.11, де до 4тої секунди друга гармоніка мала найбільшу амплітуду. Після 4тої секунди, відповідно до найменшого коефіцієнту затухання, базова гармоніка лідує по амплітуді. Це пояснює чому найпростіші тюнери, що використовують для настроювання музикальних інструментів, потребують деякий час звучання ноти

перед тим як тюнер видасть результат про базову частоту. Але для виявлення нот, тривалість яких може бути трохи більша за 31.25мс, такого методу недостатньо.

Другим найбільш тривіальним методом є пошук максимума з мінімальною частотою. Таким чином якщо буде знайден локальний максимум вище заданого порогового значення, що буде нижче по частоті ніж попередньо знайдений локальний максимум – вибір класу для присвоєння буде змінений. Для критики такого методу достатньо поглянути на спектр ноти Ре малої октави зображеної на Рисунок 3.3. На ньому бачимо виділені гармоніки, що мають кратні частоти – 146Гц, 293Гц, 440Гц, 586Гц. А також бачимо шумові компоненти, що утворюють локальні максимуми на частотах 124гц та 255Гц відповідно. Алгоритмом описаним вище було б обрано частоту 124Гц, що було б помилковим рішенням.

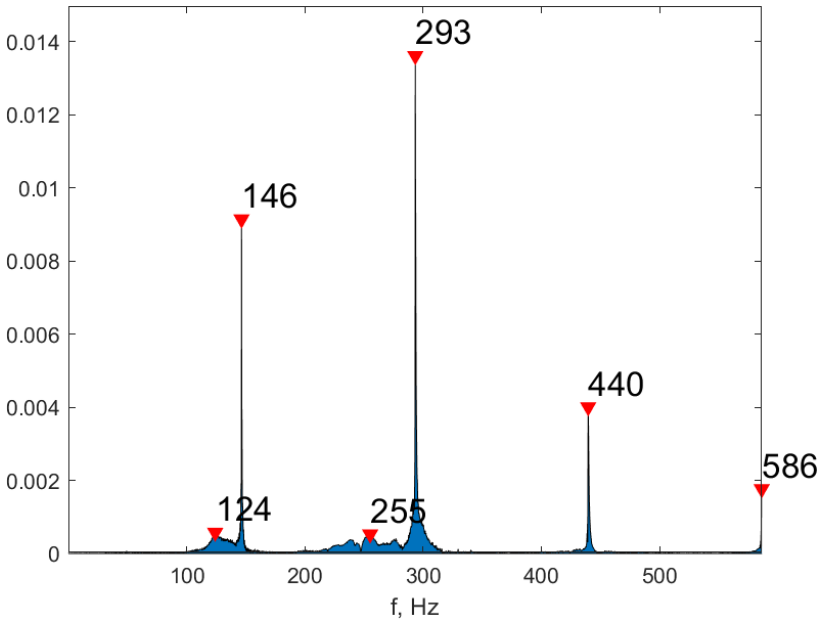

Рисунок 3.3 – Спектр ноти Ре малої октави гітари з низьким пороговим значенням

Рішенням даної проблеми такого підходу може бути класифікатор з двох частин. Перша шукає локальні максимуми, а друга рахує які з цих максимумів є кратні по частоті. На Рисунок 3.4 можна побачити, як при більш низькому пороговому значенню були виділені 8 гармонічних коливань відповідних до ноті

Ре та шумові всплески. Після відбору группи гармонік, що має найбільшу кількість кратних максимумів знайдених в першому етапі – потрапляють в другий. Таким чином можна визначити найменшу частоту коливання і класифікувати сигнал.

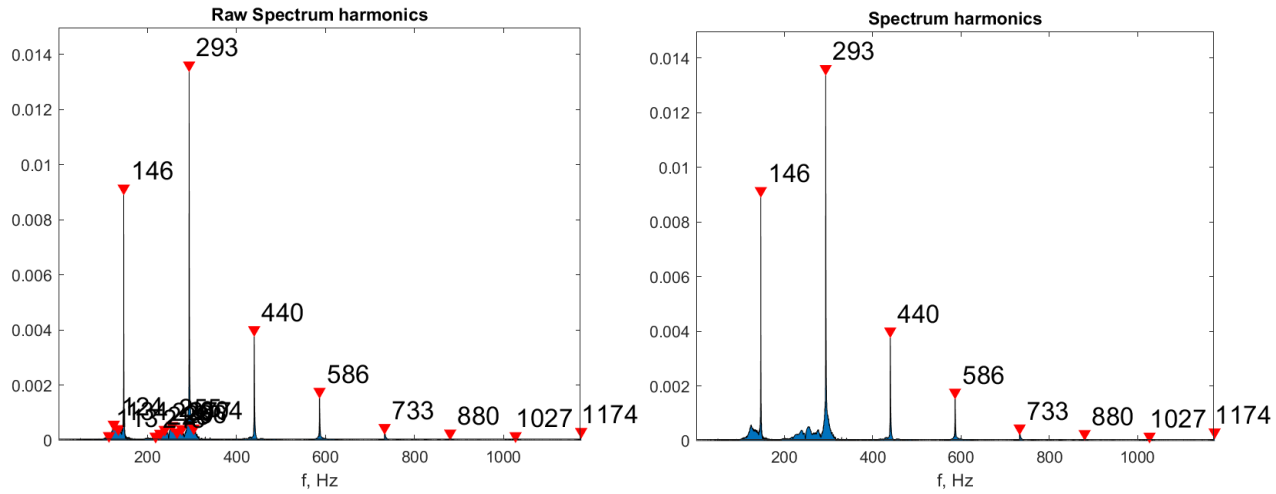

Рисунок 3.4 – Спектр ноти Ре малої октави гітари. Зліва спектр виділяє всі локальні мінімуми більші за обране низьке порогове значення. Справа локальні максимуми фільтруються по принципу кратності

Проте, як було сказано раніше, зі збільшенням частоти коливання збільшується і коефіцієнт затухання. Тому, починаючи з третьої октави кількість кратних по частоті локальних максимумів шумових компонент може бути рівним чи більшим ніж кількість виявлених гармонічних компонент. Зі збільшенням частоти такий метод показує себе гірше.

Був знайдений ще один емпіричний метод, що паказав себе краще ніж попередні. Принцип роботи такого методу заснований на знайденні сум кратних частот для кожної ноти, після чого за максимальним значенням буде визначена нота. Реалізацію цього методу можна знайти в Додатку А.

Алгоритм працює наступним чином. Спочатку будується матриця M розмірністю кількість октав на кількість нот в октаві. Для даного запису взяли 7 октав, з контр-октави до четвертої октави. Кількість нот в октаві постійна і рівна 12. В таку матрицю вносяться частоти відповідних нот, взятих з Таблиця 1.2.

Далі будується матриця V така, що

$$M_s = V * M \tag{3.1}$$

де M_s – матриця ромірністю кількість октав на кількість нот в октаві, де кожній комірці відповідає сумма октав, що є кратними для відповідної ноти. Ця сумма множиться на відповідний коефіцієнт для усереднення.

Очевидно, що при сумуванні всіх кратних частот найбільші сумми будуть на низьких частотах. Так, якщо звучить нота Ля першої октави – 440Гц, то сумми для нот Ля малої, більшої, контр та суббконтр октав будуть однакові та будуть дорівнювати максимуму. А з урахунком шуму, найбільше значення буде на ноті Ля суббконтр октави. Тому було вирішено помножити сумми на відповідні коефіцієнти k_j, що нижче нуля. Для даної задачі була використана матриця V показана на (3.2)

$$V = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} * [k_1 \quad k_2 \quad k_3 \quad k_4 \quad k_5 \quad k_6 \quad k_7] =$$

$$= \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \frac{1}{7} & \frac{1}{6} & \frac{1}{5} & \frac{1}{4} & \frac{1}{3} & \frac{1}{2} & 1 \end{bmatrix} \tag{3.2}$$

Таким чином, якщо матриця M є

$$M = \begin{bmatrix} a_{1,1} & \cdots & a_{1,12} \\ \vdots & \ddots & \vdots \\ a_{7,1} & \cdots & a_{7,12} \end{bmatrix}$$

де $a_{i,j}$ – це абсолютне значення амплітуди інтерпольованого спектру на частоті ноти i октави j.

То матриця M_s

$$M_s = \begin{bmatrix} \sum_{i=1}^{7} k_1 * a_{i,1} & \sum_{i=1}^{7} k_1 * a_{i,2} & \cdots & \sum_{i=1}^{7} k_1 * a_{i,12} \\ \sum_{i=2}^{7} k_2 * a_{i,1} & \sum_{i=2}^{7} k_2 * a_{i,2} & & \sum_{i=2}^{7} k_2 * a_{i,12} \\ \vdots & & \ddots & \vdots \\ k_7 * a_{7,1} & k_7 * a_{7,2} & \cdots & k_7 * a_{7,12} \end{bmatrix} \quad (3.3)$$

Так можемо отримати нормовані коефіцієнтами k_j сумми кратних октав для кожної ноти та класифікувати за максимальною суммою матриці M_s. Демонстрацію того, що сумма октав допомогає краще визначити ноту можна побачити при порівнянні двох рисунків - Рисунок 3.5 та Рисунок 3.6.

На Рисунок 3.5 можна побачити відображення матриці M, а саме абсолютні значення амплітуд спектру для частот кожної ноти. З цього малюнку чітко видно, що грає 3тя нота 3тої октави, так як вона є найнизькочастотним максимумом. Однак її гармоніка, що знаходиться на 4тій октаві 3тій ноті має вищу амплітуду.

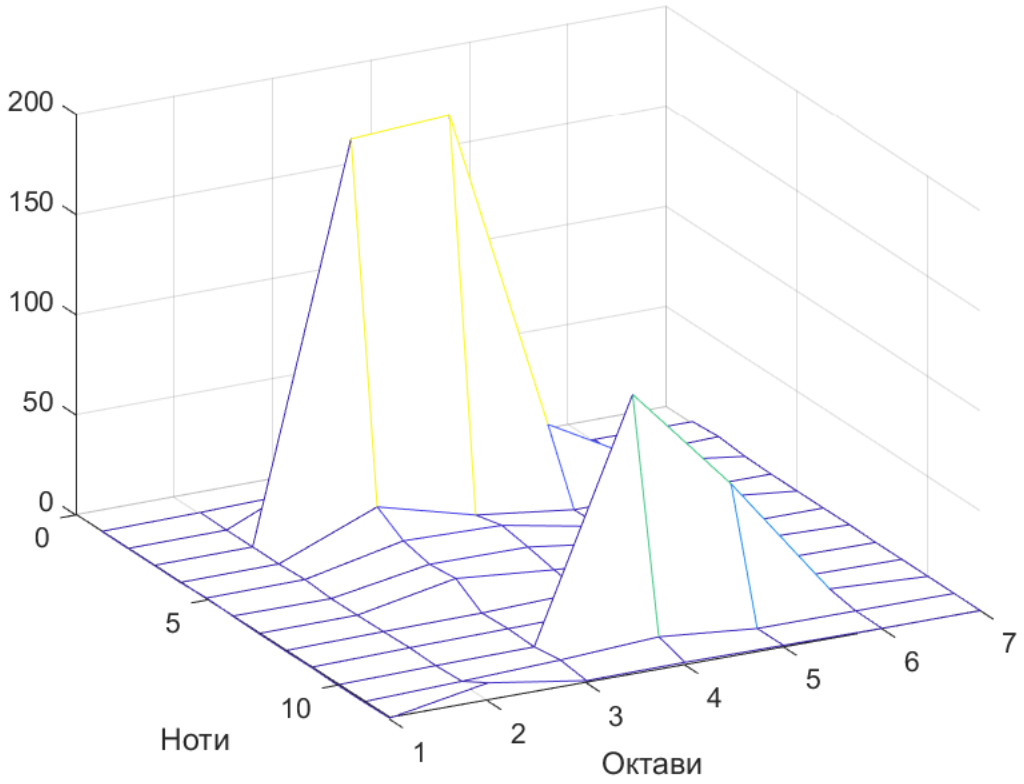

Рисунок 3.5 – Матриця M для 3ої ноти 3ої октави

Пік на 4тій октаві 10тій ноті можна пояснити тим, що не всі кратні частоти лежать через октаву від базової частоти. Як відомо з теорії нот, частоти нот зростають як показникова функція, в той час як кратні гармоніки знаходяться на фіксованій відстані одна від одної – на частоті базового тону. Таким чином, наприклад нота Ля малої октави – 220Гц, збудить як ноту Ля першої октави – 440Гц, так і ноту Мі першої октави – 660Гц. В цьому і полягає недолік даного класифікатора, він ігнорує кратні частоти що знаходяться не на октаві. Проте, частіше всього найбільша амплітуда припадає на перші 2 гармоніки після основного тону.

Таким чином, якщо просумувати таку матрицю по октавам отримаємо матрицю M_s, зображену на Рисунок 3.6. Згідно з такою матрицею M_s пошуком максимуму можна знайти точну ноту, що була в акустичному сигналі. Як видно з зображення, матриця отримала високі, проти нижчі значення амплітуди для найнижчих октав 3ої ноти. Коррекцію рівнів амплітуд в матриці M_s можна здійснити за допомогою підналаштування вектору $k = [k_1, \dots, k_7]$.

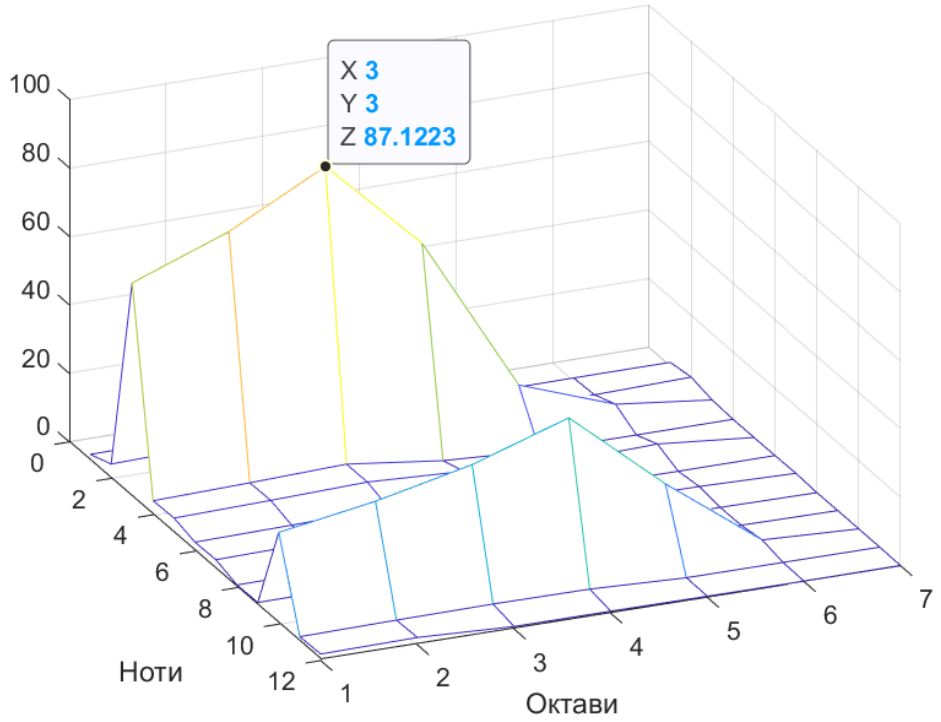

Рисунок 3.6 – Зображення матриці Ms з чітким виявленням ноти

Прийнявши ширину вікна як $NFFT = 1024$, використовуючи запис з 84ма нотами, де кожна нота звучить рівно 2 секунди, починаючи з контр-октави та закінчуючи четвертою октавою, при частоті дискретизації $F_s = 44{,}1$кГц, маємо 86 акустичних сигналів на ноту та 7224 сигналів взагалом. Результат класифікації 7224 сигналів з запису зображений на Рисунок 3.7. Синім кольором зображений результат класифікацій, в той час як жовтим зображена реальна нота. Відсоток помилки при такому методі класифікування становить 32.2%. Як видно, більшість помилок припадає на низькі частоти, що може бути пояснено через АЧХ мікрофону, що змінює амплітудні складові спектра та заважає класифікувати. Помилки на високих частотах можуть бути пояснені тим, що всі складові кратних частот на високих октавах прагнуть до нуля, в той час як нижні октави охоплюють в суммі як шум, так і високочастотну ноту. Помилки на високих частотах можуть бути зменшені підлаштовуванням вектору $k = [k_1, \dots, k_7]$.

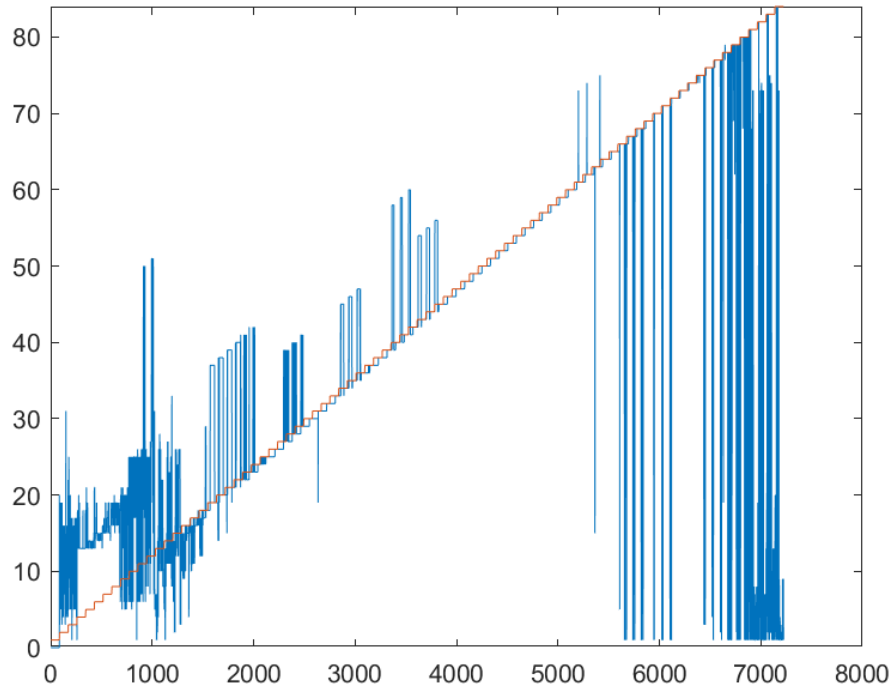

Рисунок 3.7 – Результат класифікації емпіричним методом на основі сум октав

3.3 Гаусівський наївний баєсів класифікатор

Гаусів БК – є одним з видів машинного навчання, як і метод SVM, тому при побудові алгоритму такого класифікатора слід підготувати дані для навчання та для тестування. Для всіх класифікаторів, включно з емпіричнми, використовується однаковий аудиофайл запису 84 нот піаніно починаючи з контр-октави та закінчуючи четвертою октавою. Таким чином маючи однакові вхідні дані простіше порівняти вихідні результати класифікаторів. Як було сказано раніше, при використанні вікна $NFFT = 1024$ маємо 7224 акустичних сигналів. Розбиваючи ці дані на тренувальні та тестові як 7:3 маємо 5057 сигналів для навчання та 2167 сигналів для тренування. Повний алгоритм такого класифікатора наведен в Додатку Б.

Першим підходом до застосування БК було обрання вектора ознак як абсолютні значення спектру сигналів. Таким чином для кожної з 84 нот БК побудував гаусівський розподіл, а саме – знайшов математичне очікування та дисперсію для кожного спектрального відліку. Отже, якщо значення спектрального відліку знаходиться під Гаусівським колоколом, тобто близько до середнього арифметичного, то ймовірність належності сигналу до відповідного класу висока. Таким чином, якщо перевіряється сигнал на ноту Ля першої октави, то значення спектральних відліків ноти До першої октави мають бути близькими до нуля. Відповідно, якщо сигнал має ноту До першої октави, то вірогідність цього відліку буде близитись до нуля, що згідно формулі (2.3) приблизить загальну вірогідність належності сигналу до класу Ля першої октави до нуля.

Було прийнято рішенно до нормалізування спектру перед обробкою таким чином, щоб сумма всіх відліків була рівна одиниці. Це невілює вплив зміни гучності ноти, залишаючи зміну відношення гармонік незмінним. Таким чином було отримано підвищення точності класифікації.

Також, в силу того що зі збільшенням частоти збільшується і коефіцієнт затухання, і якщо при низькооктавних нотах в спектрі можна виявити десяток кратних гармонік, для нот 4тої-5тої октави можна виявити 2-3 кратні гармоніки. Тому було вирішено подавати в класифікатор лише де-яку частину спектру.

Оскільки в даному записі максимальна частота ноти це 3951Гц, то було вирішено обрізати спектр до 4кГц. Слід зауважити, що додавання інтерполяції в спектр значно збільшило кількість спектральних відліків, що мають бути пораховані в гаусівських розподілах, тобто це значно збільшило час класифікації, проте якість класифікування аналогічно зросла. На Рисунок 3.8 можна побачити розподіл дисперсій для кожної ноти відповідно до кожного відліку спектру.

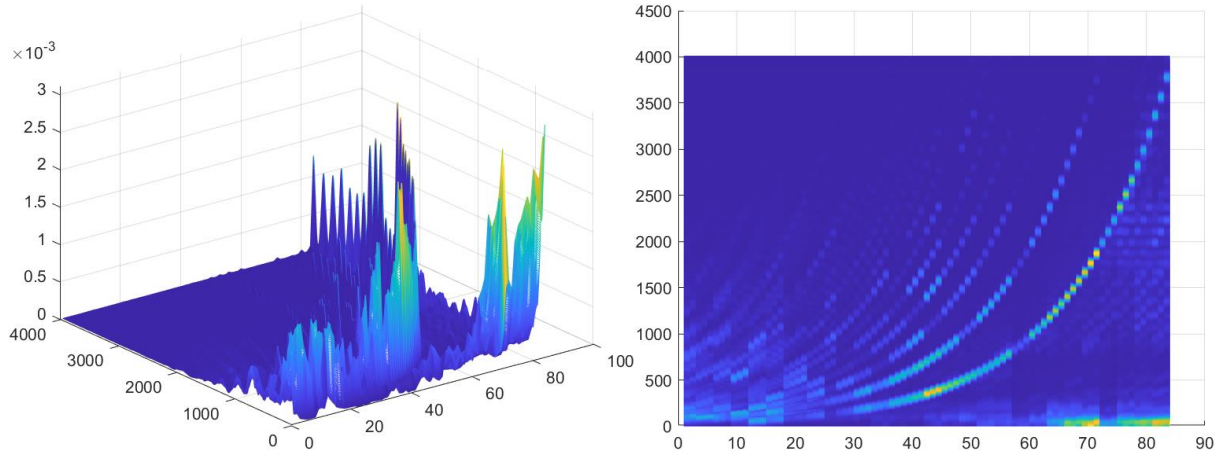

Рисунок 3.8 – Зображення дисперсії для кожного відліку спектру для 84 класифікації

В додаток було знайдено, що використання віконних функцій значно змешнило пульсації, що були визвані боковими пелюстками сінку, що в свою чергу зменшило коливання більшої кількості відліків спектру та покращило якість класифікації. В даній роботі було використано вікно Хаммінгу, що зменшує рівні бокових пелюсток більше ніж на -40дБ.

Основним недоліком такого методу є величезні затрати часу. Час на підрахунок одного гаусу в MATLAB становить приблизно 0,008с. Тестовий набір сигналів складається з 2167. Для кожного з сигналів підраховуються вірогідності належності до одного з 84 класів. При підрахунку вірогідності належності до одного класу треба порахувати стільки гаусівських функцій, скільки відліків спектру подається на вхі, після чого всі ці значення перемножаться. Було вирішено подавати спектр до 4кГц, отже, якщо подати першу $\frac{1}{11}$ інтерпольованого

спектру, отримаємо 4009 відліків відповідно до максимальної частоти – 4кГц. Отже кількість Гаусівських функцій що будуть підраховані

$$N_{\text{БК1}} = 4009 * 84 * 2167 = 729\,750\,252 \quad (3.4)$$

Якщо прийняти час виконання підрахунку однієї Гаусівської функції як 0,008с, то час затрачений на класифікації тестового пакету

$$t_{\text{БК1}} = 0{,}008\text{с} * N_{\text{БК1}} = 1622 \text{ годин}$$

Відповідно час на підрахунки для одного сигналу з тестової вибірки

$$t_{\text{БКОС}} = \frac{t_{\text{БК1}}}{2167} = 45\text{хв}$$

Було вирішено грубо спростити підрахунки гауса, що зменшить час підрахунку в 1300 разів. Виходячи з провила 3 сігми, яке стерджує, що вірогідність того що випадкова величина буде відстояти від математичного очікування на число більше за 3 сігми з вірогідністю 0.28%. Було вирішено додати перевірку чи не виходить значення за 3 сігми, якщо виходить – прирівняти вірогідність до 0.1%. Використовуючи такий підхід було отримано час на класифікацію одного сигналу

$$t_{\text{БКОС}} = 2\text{с}$$

Таким чином тестова вибірка буде підрахована за час

$$t_{\text{БК1}} = 2\text{с} * 2167 = 1\text{год } 12\text{хв}$$

Стовкнувшись з проблемою малих чисел, що вірогідності належності класу настільки малі, що не вкладаються в 8ми байтне число MATLAB, довелося використовувати підхід описаний в формулі (2.4), де добуток ймовірностей був замінений суммою логарифмів. З урахуванням жорсткої апроксимації гауса через перевірку відстані за правилом 3 сігма та послідовним присвоюванням ймовірності 0.1%, був застрахований випадок де логарифм видавав бі від’ємну нескінченність.

Результат класифікування тестової группи сигналів можна побачити на Рисунок 3.9. Відсоток помилки становить 8,5%. Час на підрахунок становить 56 хвилин.

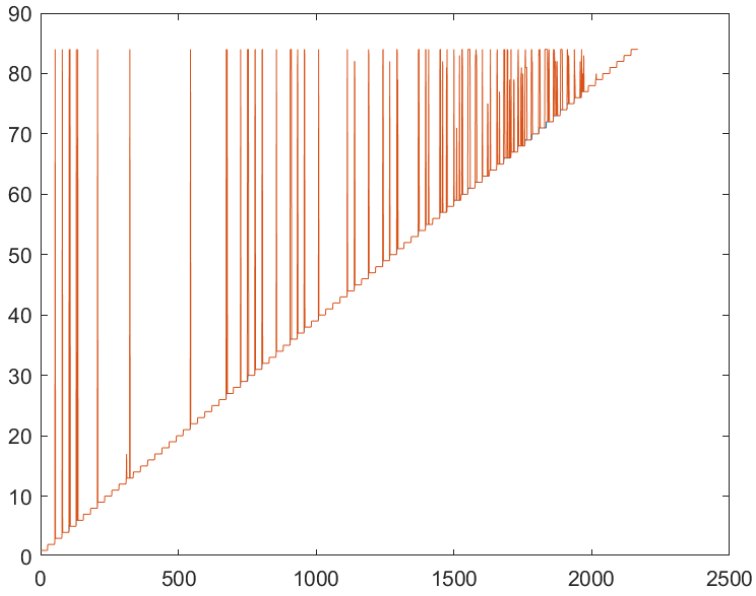

Рисунок 3.9 – Результат класифікації БК по спектру. Помилка 8,5%

Згідно формули (3.4) найбільша складова що формує кількість Гаусівських функцій – це довжина вхідного вектору ознак чи спектральних відліків. Було знайдено, що при суміщенні емпіричного методу основаному на суммах октав та формуванні матриці Ms з Гаусівським БК значно збільшується як швидкість виконання класифікації, так і її якість.

Збільшення швидкості виконання алгоритму пояснюється тим, що замість 4009 відліків БК приймає лише значення матриці Ms, яких у свою чергу всього 84. Фактично, використовуючи матрицю M замість спектру – класифікатор ігнорує всі відліки, що не є фактичними відповідностями до нот, тобто є менш важливими. В той час як використання матриці Ms замість M робить визначення ноти набагато точніше для випадків, коли амплітуда основного тону не є глобально максимальною в спектрі серед кратних гармонік. Код реалізації БК основаного на матриці Ms наведено в Додатку В.

В той час як при тренуванні БК на матриці M було отримано відсоток помилки 76%, при використанні матриці Ms маємо результат зображений на Рисунок 3.10, що відповідає відсотку помилки – 3,3%. Час підрахунків становить 2хв 40с. Отже поєднання емпіричного методу класифікування з класичним Гаусівським БК дало найбільш якісний результат класифікування. Слід зауважити

що доцільним підлаштуванням вектору $k = [k_1, \dots, k_7]$ відсоток помилки може бути зменшиний.

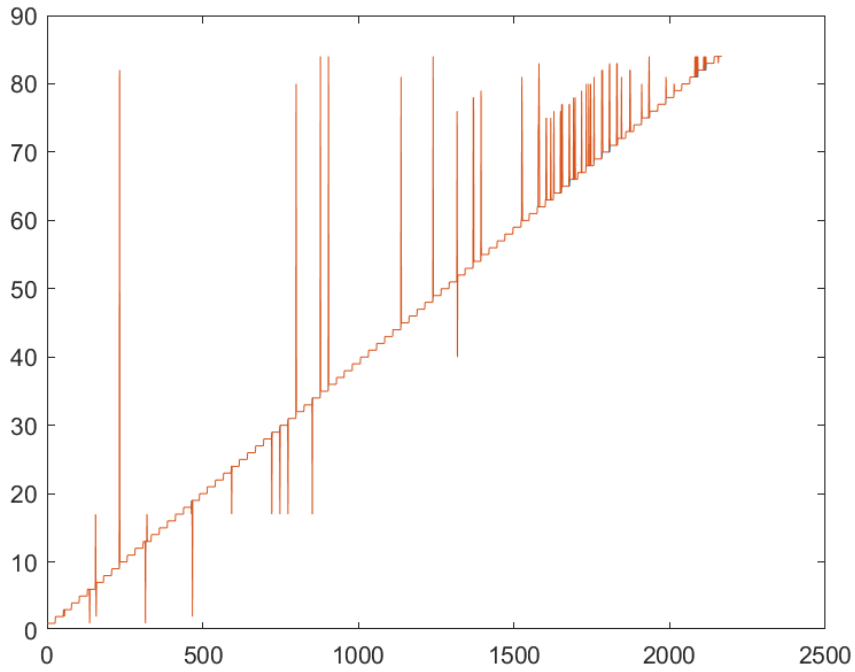

Рисунок 3.10 – Результат класифікації БК по матриці Ms. Помилка 3,3%

3.4 Класифікатор на базі методу опорних векторів

SVM, так само як і Гаусів БК є видом машинного навчання. підхід до тренування та навчання такого класифікатора абсолютно тотожній до БК. На вхід класифікатора подавалися 4009 відліків спектру, відношення тренувальних данних до тестових 7:3, спектри пронормовані так, щоб сумма відліків була рівна одиниці.

Замість побудови Гаусівських функцій класифікатор SVM побудує ряд бінарних класифікаторів точок 4009 мірного простору.

Слід також зауважити знатну чутливість до масштабування вектору ознак. Так, при мастабуванні вектору за принципом «сумма елементів вектора рівна одиниці» було отримано результат класифікації з помилкою в 86%, тоді як з використанням нормалізації за принципом «всі елементи вектору мають лежати в межах від 0 до 1» було отримано результат з помилкою в 1,1%. В той час як БК

показує протилежну залежність, при першому способі нормалізації – 3% помилки, при другому – 10%.

Застосування SVM до спектральних відліків показало результат в 1,1% помилки класифікації, що можна побачити на Рисунок 3.11.

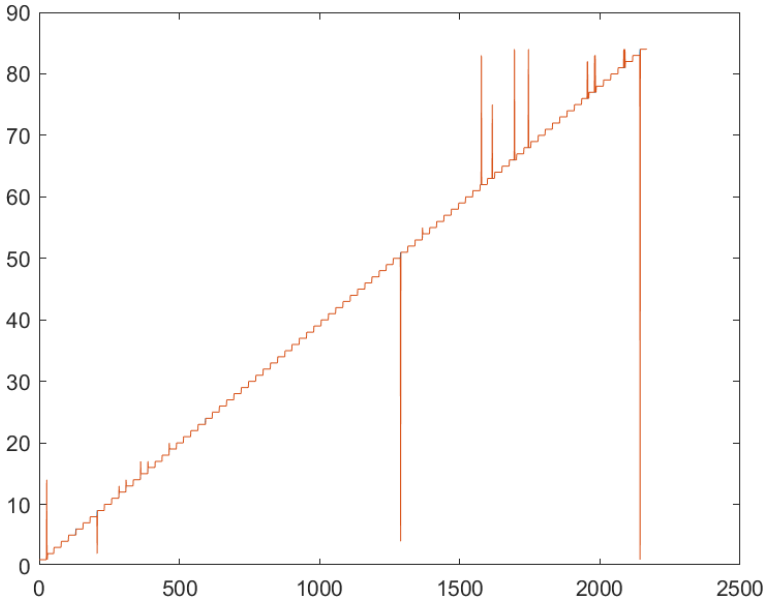

Рисунок 3.11 – Результат класифікації SVM по спектру. Помилка 1,1%

Застосування емпірично виведеної матриці Ms для навчання та класифікування значно зменшили час класифікації. Час навчання становить 25с, час класифікації ряду тестових сигналів становить 6с. Результат класифікації таким класифікатором показує дещо гірший відсоток помилки – 1,8%, що можна побачити на Рисунок 3.12. В той час як використання матриці M дає результат в 3% помилки. Отже застосування методу сум октав допомогає зменшити час розрахунку для SVM ціною зменшення точності класифікації.

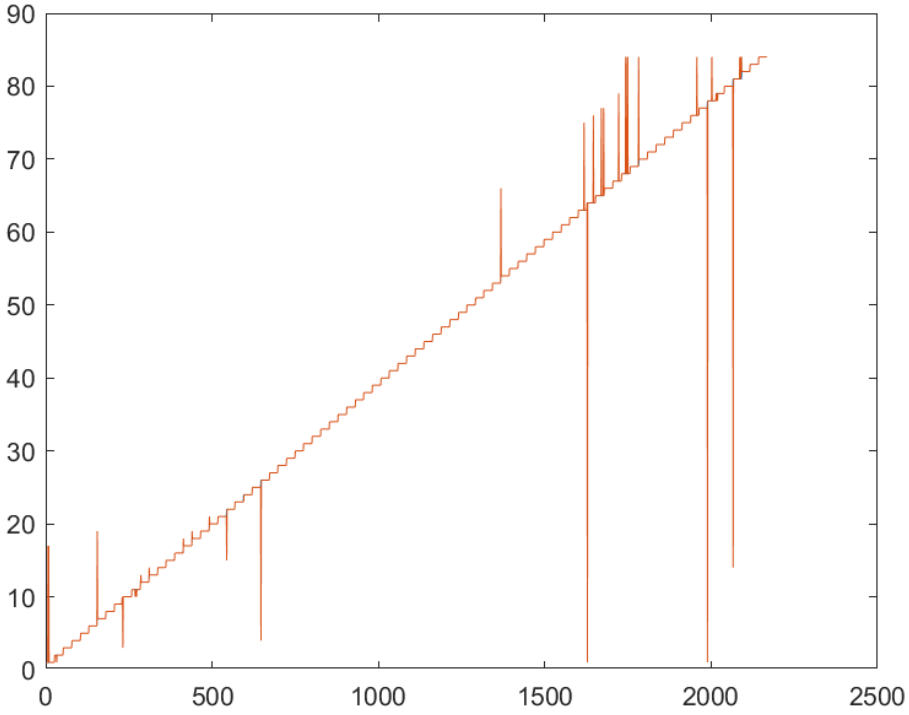

Рисунок 3.12 – Результат класифікації SVM по матриці Ms. Помилка 1,8%

ВИСНОВКИ

Отже, існує ряд можливих рішень проблеми класифікації нот в цифровому акустичному сигналі використовуючи машинне навчання. Вони різняться як точністю класифікації, способом попередньої обробки сигналів і часом обробки.

Так, емпірично розроблений метод, що базується на сумі кратних гармонік, показує результат в 32% помилок. Невелика точність може пояснюватися тим, що дані суми мають усереднюватися, що є окремою і складною задачею, так як тембри інструментів не є тотожніми.

Метод наївного БК з гаусівською апроксимацією ЩЙ потребував найбільшого часу на розрахунки, так як для кожного елементу вхідного вектору ознак було розраховано гаусівську функцію. Проте, класифікатор показав результат в 8,5% помилки аналізуючи перші 4009 відліки інтерпольованого спектру вирізаного сигналу. А в поєднанні з емпіричним методом класифікації, з заміною вхідного вектору ознак на матрицю Ms класифікатор показав результат в 3,2% помилки.

Метод SVM показав найкращі показники точності. В той час як заміна вектору ознак з спектральних відліків на матрицю Ms для БК дав приріст в точності, для методу SVM такий підхід показав зворотній ефект. Так само потребувався інший метод нормалізації вхідних даних. Тоді як БК потребував нормалізації вхідних даних за принципом «сума елементів вектору дорівнює одиниці», метод SVM показав кращі результати при нормалізації за принципом «кожен елемент має бути в межах від 0 до 1», що було реалізовано шляхом ділення вектору на максимальний елемент.

Так, при класифікувані на основі спектральних відліків за першим принципом нормалізації було отримано результат з 86% помилки, за другим принципом – 1,1%.

При використанні матриці Ms в якості вектора ознак знизився як час класифікації, так і точність, в той час як для БК такий підхід підвищив показник

точності. Метод SVM показав помилку в 1,8% при використанні матриці Ms, та 3% при використанні матриці M. Отже метод більш чустливий та точний.

Подальший розвиток класифікації може бути виконаний шляхом використання статистичних залежностей між нотами та використання методу максимальної правдоподібності до класифікованої послідовності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Как получают разные тона? [Електронний ресурс]: Как получают разные тона? / Режим доступу: http://information-technology.ru/sci-pop-articles/23-physics/268-kak-poluchayut-raznye-tona. – Назва за екрану.

2. Семченок М. С. Колебания струны / М. С. Семченок, І. Н. Щитов. – Санкт-Петербург: Кафедра математики та інформатики, 2010. – 43 с.

3. Алдошина І. А. Музыкалькая акустика / І. А. Алдошина, Р. Приттс. – Санкт-Петербург: Комозитор, Санкт-Петербург, 2006. – 712 с.

4. Метод опорных векторов (SVM) [Електронний ресурс] – Режим доступу до ресурсу: https://lnnk.in/hQaX.

5. Гауссов классификатор [Електронний ресурс] – Режим доступу до ресурсу: https://lnnk.in/h7aJ.

6. Wine Data Set [Електронний ресурс] – Режим доступу до ресурсу: https://archive.ics.uci.edu/ml/datasets/Wine.

7. The Naive Probability Is the Expert on Wine Quality [Електронний ресурс] – Режим доступу до ресурсу: https://blog.uiam.sk/the-naive-probability-is-an-expert/.

8. Support Vector Machine simplified using R [Електронний ресурс] – Режим доступу до ресурсу: https://www.listendata.com/2017/01/support-vector-machine-in-r-tutorial.html.

9. Duan K. Which Is the Best Multiclass SVM Method? [Електронний ресурс] / K. Duan, S. Sathiya Keerthi – Режим доступу до ресурсу: http://www.keerthis.com/multiclass_mcs_kaibo_05.pdf.

10. Rifkin R. In Defense of One-Vs-All Classification [Електронний ресурс] / R. Rifkin, A. Klautau. – 2004. – Режим доступу до ресурсу: https://www.jmlr.org/papers/volume5/rifkin04a/rifkin04a.pdf.

ДОДАТОК А

Код MATLAB з реалізацією емпіричного методу класифікування за матрицею Ms

```
clear all
close all


% 1. Завантаження сигналу з послідовних 84 нот
% Завантаження музичного запису 88 нот в змінну z
path = "D:\Desktop\Studie\Diploma\GIT Matlab\Audio data\wav";
path1 = path + "\Sasha\Steinway Grand Piano 70.wav";
[y, Fs] = audioread(path1);
z = y(:,1);
clear path path1 y


NOTES  = 84;                    % Кількість нот
NFFT   = 1024;                  % Ширина вікна
N_zero = Fs - NFFT;             % Кількість нулів для інтерполяції
Zer    = zeros(N_zero, 1);      % Вектор нулів для інтерполяції
N      = floor(2/(NFFT/Fs));    % Кількість сигналів для однієї ноти


data = reshape(z,4*Fs,[]);      % Відокремлюю кожну ноту по рядкам
data = data(1:(N*NFFT),:);      % Округлюю кількість семплів для зручності
data = data(:, 1:87);           % Забираю зайву ноту До 5тої октави
data = data(:,4:end);           % Откидываю первые 3 ноты суб-контр октавы
clear z


% 2. Формування таблиці частот що відповідають 84 нотам
Tab_F_1 = [65.41   69.3   73.42   77.78   82.41   87.31   92.5   98   103.83   110   116.54
123.47];
k = 2.^(0:6) / 2;
Tab_F = k' .* repmat(Tab_F_1, 7, 1);
Indexes = floor(Tab_F);          % Таблиця індексів нот для інтерпольованого спектру
clear Tab_F_1 Tab_F k


% 3. Підготовка матриць M, V, Ms
M  = zeros(7, 12);
Ms = M;
v1 = [1 1 1 1 1 1 1 ] * 1/7;
v2 = [0 1 1 1 1 1 1 ] * 1/6;
v3 = [0 0 1 1 1 1 1 ] * 1/5;
v4 = [0 0 0 1 1 1 1 ] * 1/4;
v5 = [0 0 0 0 1 1 1 ] * 1/3;
v6 = [0 0 0 0 0 1 1 ] * 1/2;
v7 = [0 0 0 0 0 0 1 ];
V  = [v1; v2; v3; v4; v5; v6; v7];
clear v1 v2 v3 v4 v5 v6 v7
```

```
% 4. Формування вектору міток Y
Y = 1:NOTES;
Y = repmat(Y, N, 1);
Y = reshape(Y, 1, [])';


% 5. Використання класифікатору
p_error   = 0;                    % Змінна для зберігання кількості помилок
notes_lin = zeros(1, N*NOTES);    % Вектор для запису результатів класифікації


for i = 1:NOTES
    for j = 1:N

        % Обирається сигнал довжиною в NFFT з data
        n1 = (j-1)*NFFT + 1;
        n2 = (j)*NFFT;
        S  = data(n1:n2, i);

        % До сигналу додаються нулі для інтерполяції
        S2 = vertcat(Zer, S);

        % Знаходиться спектр та підраховується матриця Ms
        Spec = abs(fft(S2));
        M    = Spec(Indexes);
        Ms   = V * M;

        % Знаходиться індекс максимального елемента матриці Ms та
        % визначається нота, що записується в вектор notes_lin
        ar_ind = N*(i-1) + j;
        [~, Ind] = max(Ms',[],[1 2], 'linear');
        notes_lin(ar_ind) = Ind;

        if Ind ~= Y(ar_ind)
            p_error = p_error + 1;
        end
    end
end


figure
plot(notes_lin)
hold on
plot(Y)
ylim([0, 84])


p_error = 100 * p_error / (N*NOTES);
disp('Error rate: ' + string(p_error) + '%')
```

ДОДАТОК Б

Код MATLAB з реалізацією Гуасівського БК на основі спектру

```
clear all
close all


% 1. Завантаження сигналу з послідовних 84 нот
% Завантаження музичного запису 88 нот в змінну z
path = "D:\Desktop\Studie\Diploma\GIT Matlab\Audio data\wav";
path1 = path + "\Sasha\Steinway Grand Piano 70.wav";
[y, Fs] = audioread(path1);
z = y(:,1);
clear path path1 y


NOTES  = 84;                    % Кількість нот
NFFT   = 1024;                  % Ширина вікна
N_zero = Fs - NFFT;             % Кількість нулів для інтерполяції
N      = floor(2/(NFFT/Fs));    % Кількість сигналів для однієї ноти
ds     = 11;                    % В скільки разів обрізати спектр


data = reshape(z,4*Fs,[]);      % Відокремлюю кожну ноту по рядкам
data = data(1:(N*NFFT),:);      % Округлюю кількість семплів для зручності
data = data(:, 1:87);           % Забираю зайву ноту До 5тої октави
data = data(:,4:end);           % Откидываю первые 3 ноты суб-контр октавы
clear z

% 2. Формування вектору міток Y та вектору об'єктів X
Y = zeros(N*NOTES, 1);
X = zeros([N*NOTES, NFFT+N_zero]);    % З інтерполяцією
% X = zeros([N*NOTES, NFFT]);         % Без інтерполяції


for k=1:NOTES
    N1 = N * (k-1) + 1;
    N2 = N1 + N - 1;
    notes = reshape(data(:,k),NFFT,N)';

    % Використовую віконну функцію Хаммінгу
    h = hamming(NFFT)';
    notes = notes .* h;

    X(N1:N2, 1:NFFT) = notes;
    Y(N1:N2) = k;
end


% 3. Розділяю дані на тренувальні та тестові
cv = cvpartition(Y,'HoldOut',0.30);
trainInds = training(cv);
testInds  = test(cv);
```

```
XTrain = X(trainInds,:);
YTrain = Y(trainInds);
XTest  = X(testInds,:);
YTest  = Y(testInds);
clear trainInds testInds X Y data


% 4. Проводжу попередню обробку сигналів
[STest, MTest]   = size(XTest);
[STrain, MTrain] = size(XTrain);


% Переводжу сигнали в частотний домен
XTest_spec  = abs(fft(XTest'))';
XTrain_spec = abs(fft(XTrain'))';


% Кількість відліків спектру після обрізання в векторах
vtest  = floor(1:MTest/ds);
vtrain = floor(1:MTrain/ds);


% Обрізаю спектри в ds разів
XTest_ds  = zeros([STest, length(vtest)]);
XTrain_ds = zeros([STrain, length(vtrain)]);
XTest_ds(:, vtest)  = XTest_spec(:, vtest);
XTrain_ds(:, vtest) = XTrain_spec(:, vtest);


% Нормалізація спектрів
XTest_ds  = XTest_ds ./ sum(XTest_ds, 2);
XTrain_ds = XTrain_ds ./ sum(XTrain_ds, 2);
clear XTrain XTest XTest_spec XTrain_spec


% 5. Знаходження параметрів Гаусівських розподілів
mean_ds = zeros([length(vtrain), NOTES]); % Масив для мат очікувань
std_ds  = zeros([length(vtrain), NOTES]); % Масив для дисперсій
for num_notes = 1 : NOTES
    % Отримую всі сигнали, що належать до заданої ноти
    ind   = find(YTrain==num_notes);
    X_cur = XTrain_ds(ind, :);
    % Знаходжу мат очікування та дисперсію окремих відліків по сигналам
    mean_ds(:,num_notes) = mean(X_cur,1)';
    std_ds(:,num_notes)  = std(X_cur,1,1)';
end


% 6. Використання класифікатору
p_error  = 0;                    % Змінна для зберігання кількості помилок
my_notes = zeros([STest 1]);     % Вектор для запису результатів класифікації
for j = 1:STest

    test_sig = XTest_ds(j, :);      % Обирається єдиний сигнал для класифікації
```

```
    test_label = YTest(j);          % Запам'ятовується його мітка

    Prob_note = ones([NOTES, 1]);   % Вектор вірогідності належності до кожної ноти

    for i = 1:NOTES

        for k = 1:length(vtrain)

            x = test_sig(k);           % Обирається єдиний відлік сигналу
            std_cur = std_ds(k, i);    % Беруться відповідні параметри Гаусу
            mean_cur = mean_ds(k, i);  %

            % Використовується грубе апроксимування Гаусу для великих
            % відстаней від математичного очікування
            if abs(x-mean_cur) > 3*std_cur
                p = 0.001;
            else
                p = gaussmf(x, [std_cur mean_cur]);
            end

            % Знаходяться вірогідності для кожної ноти
            Prob_note(i) = Prob_note(i) + log(p);

        end

    end

    % Обирається нота за максимальною вірогідністю
    note = find(Prob_note==max(Prob_note));

    % У випадку декількох максимумів - обирається перший
    my_notes(j) = note(1);

    % перевірка на помилковий результат
    if my_notes(j) ~= YTest(j)
        p_error = p_error + 1;
    end

end


figure
plot(YTest)
hold on
plot(my_notes)
hold off


p_error = 100 * p_error / STest;
disp('Error rate: ' + string(p_error) + '%')
```

ДОДАТОК В

Код MATLAB з реалізацією Гуасівського БК на основі емпіричного методу з матрицею Ms

```
clear all
close all


% 1. Завантаження сигналу з послідовних 84 нот
% Завантаження музичного запису 88 нот в змінну z
path = "D:\Desktop\Studie\Diploma\GIT Matlab\Audio data\wav";
path1 = path + "\Sasha\Steinway Grand Piano 70.wav";
[y, Fs] = audioread(path1);
z = y(:,1);
clear path path1 y


NOTES  = 84;                    % Кількість нот
NFFT   = 1024;                  % Ширина вікна
N_zero = Fs - NFFT;             % Кількість нулів для інтерполяції
N      = floor(2/(NFFT/Fs));    % Кількість сигналів для однієї ноти
ds     = 11;                    % В скільки разів обрізати спектр


data = reshape(z,4*Fs,[]);      % Відокремлюю кожну ноту по рядкам
data = data(1:(N*NFFT),:);      % Округлюю кількість семплів для зручності
data = data(:, 1:87);           % Забираю зайву ноту До 5тої октави
data = data(:,4:end);           % Откидываю первые 3 ноты суб-контр октавы
clear z


% 2. Формування таблиці частот що відповідають 84 нотам
Tab_F_1 = [65.41  69.3  73.42  77.78  82.41  87.31  92.5  98  103.83  110  116.54
123.47];
k = 2.^(0:6) / 2;
Tab_F = k' .* repmat(Tab_F_1, 7, 1);
Indexes = floor(Tab_F);               % Таблиця індексів нот для інтерпольованого
спектру
clear Tab_F_1 Tab_F k


% 3. Підготовка матриць M, V, Ms
M = zeros(7, 12);
Ms = M;
v1 = [1 1 1 1 1 1 1 ] * 1/7;
v2 = [0 1 1 1 1 1 1 ] * 1/6;
v3 = [0 0 1 1 1 1 1 ] * 1/5;
v4 = [0 0 0 1 1 1 1 ] * 1/4;
v5 = [0 0 0 0 1 1 1 ] * 1/3;
v6 = [0 0 0 0 0 1 1 ] * 1/2;
v7 = [0 0 0 0 0 0 1 ];
V = [v1; v2; v3; v4; v5; v6; v7];
clear v1 v2 v3 v4 v5 v6 v7
```

```
% 4. Формування вектору міток Y та вектору об'єктів X
Y = zeros(N*NOTES, 1);
X = zeros([N*NOTES, NFFT+N_zero]);    % З інтерполяцією
% X = zeros([N*NOTES, NFFT]);         % Без інтерполяції


for k=1:NOTES
    N1 = N * (k-1) + 1;
    N2 = N1 + N - 1;
    notes = reshape(data(:,k),NFFT,N)';

    % Використовую віконну функцію Хаммінгу
    h = hamming(NFFT)';
    notes = notes .* h;

    X(N1:N2, 1:NFFT) = notes;
    Y(N1:N2) = k;
end


% 5. Розділяю дані на тренувальні та тестові
cv = cvpartition(Y,'HoldOut',0.30);
trainInds = training(cv);
testInds  = test(cv);


XTrain = X(trainInds,:);
YTrain = Y(trainInds);
XTest  = X(testInds,:);
YTest  = Y(testInds);
clear trainInds testInds X Y data


% 6. Проводжу попередню обробку сигналів
[STest, MTest]   = size(XTest);
[STrain, MTrain] = size(XTrain);


% Переводжу сигнали в частотний домен
XTest_spec  = abs(fft(XTest'))';
XTrain_spec = abs(fft(XTrain'))';


% Кількість відліків спектру після обрізання в векторах
vtest  = floor(1:MTest/ds);
vtrain = floor(1:MTrain/ds);


% Обрізаю спектри в ds разів
XTest_ds  = zeros([STest, length(vtest)]);
XTrain_ds = zeros([STrain, length(vtrain)]);
XTest_ds(:, vtest)  = XTest_spec(:, vtest);
XTrain_ds(:, vtest) = XTrain_spec(:, vtest);
```

```
% Нормалізація спектрів
XTest_ds  = XTest_ds ./ sum(XTest_ds, 2);
XTrain_ds = XTrain_ds ./ sum(XTrain_ds, 2);
clear XTrain XTest XTest_spec XTrain_spec


% 7. Знаходження параметрів Гаусівських розподілів матриць Ms
mean_M = zeros([NOTES 7 12]);   % Масив для мат очікувань
std_M  = zeros([NOTES 7 12]);   % Масив для дисперсій


for i = 1:NOTES
    % Отримую всі сигнали, що належать до заданої ноти
    ind = find(YTrain==i);
    X_note = XTrain_ds(ind, :);

    Ncuts = length(ind);
    temp_M = zeros([Ncuts 7 12]);
    for k = 1:Ncuts
        x = X_note(k, :);       % Обираю єдиний сигнал даної ноти
        M  = x(Indexes);        % Отримаю матрицю Ms для цього сигналу
        Ms = V * M;             %
        temp_M(k, :, :) = Ms;   % Зберігаю матриці Ms обраної ноти в масив
    end
    mean_M(i, :, :) = mean(temp_M, 1);  % Отримую параметри Гаусівських розподілів
    std_M(i, :, :)  = std(temp_M, 1);   % Зі збережених матриць Ms для обраної ноти
end


% 8. Використання класифікатору
p_error  = 0;                   % Змінна для зберігання кількості помилок
my_notes = zeros([STest 1]);    % Вектор для запису результатів класифікації
for j = 1:STest

    test_sig = XTest_ds(j, :);      % Обирається єдиний сигнал для класифікації
    test_label = YTest(j);          % Запам'ятовується його мітка


    M = test_sig(Indexes);          % Отримую матрицю Ms для обраного сигналу
    Ms = V * M;                     %

    Prob_note = ones([NOTES, 1]);   % Вектор вірогідності належності до кожної ноти

    for i = 1:NOTES

        for k = 1:(7*12)

            m = Ms(k);                % Обирається відлік матриці Ms
            std_cur  = std_M(i, k);   % Беруться відповідні параметри Гаусу
            mean_cur = mean_M(i, k);  %

            % Рахується гаусівський розподіл
            p = gaussmf(m, [std_cur mean_cur]);

            % Знаходиться вірогідність належності до кожної ноти
```

```
            Prob_note(i) = Prob_note(i) * p;
        end

    end

    % Обирається нота за максимальною вірогідністю
    note = find(Prob_note==max(Prob_note));

    % У випадку декількох максимумів - обирається перший
    my_notes(j) = note(1);

    % перевірка на помилковий результат
    if my_notes(j) ~= test_label
        p_error = p_error + 1;
    end

end


figure
plot(YTest)
hold on
plot(my_notes)
hold off


p_error = 100 * p_error / STest;
disp('Error rate: ' + string(p_error) + '%')
```

ДОДАТОК Г

Код MATLAB з використанням SVM на матриці Ms

```
clear all
close all


% 1. Завантаження сигналу з послідовних 84 нот
% Завантаження музичного запису 88 нот в змінну z
path = "D:\Desktop\Studie\Diploma\GIT Matlab\Audio data\wav";
path1 = path + "\Sasha\Steinway Grand Piano 70.wav";
[y, Fs] = audioread(path1);
z = y(:,1);
clear path path1 y


NOTES  = 84;                    % Кількість нот
NFFT   = 1024;                  % Ширина вікна
N_zero = Fs - NFFT;             % Кількість нулів для інтерполяції
N      = floor(2/(NFFT/Fs));    % Кількість сигналів для однієї ноти
ds     = 11;                    % В скільки разів обрізати спектр


data = reshape(z,4*Fs,[]);      % Відокремлюю кожну ноту по рядкам
data = data(1:(N*NFFT),:);      % Округлюю кількість семплів для зручності
data = data(:, 1:87);           % Забираю зайву ноту До 5тої октави
data = data(:,4:end);           % Откидываю первые 3 ноты суб-контр октавы
clear z


% 2. Формування таблиці частот що відповідають 84 нотам
Tab_F_1 = [65.41  69.3  73.42  77.78  82.41  87.31  92.5  98  103.83  110  116.54
123.47];
k = 2.^(0:6) / 2;
Tab_F = k' .* repmat(Tab_F_1, 7, 1);
Indexes = floor(Tab_F);               % Таблиця індексів нот для інтерпольованого
спектру
clear Tab_F_1 Tab_F k


% 3. Підготовка матриць M, V, Ms
M = zeros(7, 12);
Ms = M;
v1 = [1 1 1 1 1 1 1 ] * 1/7;
v2 = [0 1 1 1 1 1 1 ] * 1/6;
v3 = [0 0 1 1 1 1 1 ] * 1/5;
v4 = [0 0 0 1 1 1 1 ] * 1/4;
v5 = [0 0 0 0 1 1 1 ] * 1/3;
v6 = [0 0 0 0 0 1 1 ] * 1/2;
v7 = [0 0 0 0 0 0 1 ];
V = [v1; v2; v3; v4; v5; v6; v7];
clear v1 v2 v3 v4 v5 v6 v7
```

```
% 4. Формування вектору міток Y та вектору об'єктів X
Y = zeros(N*NOTES, 1);
X = zeros([N*NOTES, NFFT+N_zero]);    % З інтерполяцією
% X = zeros([N*NOTES, NFFT]);         % Без інтерполяції


for k=1:NOTES
    N1 = N * (k-1) + 1;
    N2 = N1 + N - 1;
    notes = reshape(data(:,k),NFFT,N)';

    % Використовую віконну функцію Хаммінгу
    h = hamming(NFFT)';
    notes = notes .* h;

    X(N1:N2, 1:NFFT) = notes;
    Y(N1:N2) = k;
end


% 5. Розділяю дані на тренувальні та тестові
cv = cvpartition(Y,'HoldOut',0.30);
trainInds = training(cv);
testInds  = test(cv);


XTrain = X(trainInds,:);
YTrain = Y(trainInds);
XTest  = X(testInds,:);
YTest  = Y(testInds);
clear trainInds testInds X Y data


% 6. Проводжу попередню обробку сигналів
[STest, MTest]   = size(XTest);
[STrain, MTrain] = size(XTrain);


% Переводжу сигнали в частотний домен
XTest_spec  = abs(fft(XTest'))';
XTrain_spec = abs(fft(XTrain'))';


% Кількість відліків спектру після обрізання в векторах
vtest  = floor(1:MTest/ds);
vtrain = floor(1:MTrain/ds);


% Обрізаю спектри в ds разів
XTest_ds  = zeros([STest, length(vtest)]);
XTrain_ds = zeros([STrain, length(vtrain)]);
XTest_ds(:, vtest)  = XTest_spec(:, vtest);
XTrain_ds(:, vtest) = XTrain_spec(:, vtest);


% Нормалізація спектрів
```

```
XTest_ds  = XTest_ds ./ max(XTest_ds, [], 2);   %sum(XTest_ds, 2);
XTrain_ds = XTrain_ds ./ max(XTrain_ds, [], 2); %sum(XTrain_ds, 2);
clear XTrain XTest XTest_spec XTrain_spec


% 7. Формую тренувальні та тестові матриці Ms
for i = 1:STrain
    x = XTrain_ds(i, :);        % Обираю єдиний тренувальний сигнал
    M = x(Indexes);             % Заповнюю матрицю Ms
    Ms_Train(i, :, :) = V * M;  %
end


for i = 1:STest
    x = XTest_ds(i, :);         % Обираю єдиний тестовий сигнал
    M = x(Indexes);             % Заповнюю матрицю Ms
    Ms_Test(i, :, :) = V * M;   %
end


Ms_Train = reshape(Ms_Train, [STrain, 7*12]);   % Перетворення матриці в вектор ознак
Ms_Test  = reshape(Ms_Test, [STest, 7*12]);     %

% Нормалізація вхідного вектору ознак
Ms_Train = Ms_Train ./ max(Ms_Train, [], 2);
Ms_Test  = Ms_Test ./ max(Ms_Test, [], 2);


% 8. Використання класифікатору
Mdl_freq = fitcecoc(Ms_Train,YTrain);           % Навчання моделі
label    = predict(Mdl_freq,Ms_Test);           % Класифікування

figure
plot(YTest)
hold on
plot(label)
hold off

p_error = 0;
for i = 1:STest
    if YTest(i) ~= label(i)
        p_error = p_error + 1;
    end
end
p_error = 100 * p_error/STest;
disp('Error rate: ' + string(p_error) + '%')
```