Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Радіотехнічний факультет

Кафедра радіоконструювання та виробництва радіоапаратури

**Дипломна робота**

**на тему: «Методи классифікації нот в акустичному сигналі»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Керівник: | |  |  | Виконав | Красницький |
| Шпилька О.О. | |  |  | Група: | РТ-71 |
|  | |  |  |  | |
| Комісія: | |  |  | (ПІБ) | |
|  |  |  |  |  | |
|  |  |  |  | Дата | Підпис |

Київ — 2021 р.

Анотація

Дипломна робота викладена на 60 сторінках, містить 22 ілюстрацій, 2 таблиці та 4 додатки.

Метою цього проекту є огляд базових класифікаторів та створення алгоритму классифікації акустичних сигналів. А саме, такий класифікатор буде визначати яка нота грає на аудиозаписі.

Анотация

Дипломная работа изложена на 60 страницах, имеет 22 иллюстрации, 2 таблицы и 4 приложения.

Целью

фів

фів

Перелік умовних позначень

SVM –support vector machines

БК – Байесовський класифікатор

FFT – Fast Fourier transform

Змі**с**т

[Анотація 2](#_Toc72676912)

[Анотация 3](#_Toc72676913)

[Перелік умовних позначень 4](#_Toc72676914)

[Зміст 5](#_Toc72676915)

[Вступ 7](#_Toc72676916)

[1 Теорія музики 8](#_Toc72676917)

[1.1 Теорія нот 8](#_Toc72676918)

[1.2 Фізика тембру 12](#_Toc72676919)

[2 Теорія класифікації 18](#_Toc72676920)

[2.1 Основи класифікації 18](#_Toc72676921)

[2.2 Байесівска класифікація. Гаусовський наївний баєсів класифікатор 19](#_Toc72676922)

[2.3 SVM 23](#_Toc72676923)

[2.4 Машинне навчання. Перцептрон 29](#_Toc72676924)

[3 Дослідження методів классифікації 31](#_Toc72676925)

[3.1 Вибір параметрів вхідного сигналу для класифікації 31](#_Toc72676926)

[3.1.1 Домен сигналу 31](#_Toc72676927)

[3.1.2 Довжина сигналу 31](#_Toc72676928)

[3.1.3 Використання інтерполяції спектру 33](#_Toc72676929)

[3.2 Емпіричні методи класифікації 35](#_Toc72676930)

[3.3 Гаусівський наївний баєсів класифікатор (Gaussian naïve Bayes) 37](#_Toc72676931)

[3.4 SVM класифікатор 39](#_Toc72676932)

[3.5 Застосування машинного навчання з підкріпленням 40](#_Toc72676933)

Вступ

# Теорія музики

## Теорія нот

Нота - це графічна одиниця музикальної нотації [1] або музична елементарна одиниця композиції. Будь-яка класична композиція складається з нот (звуків) і пауз.

Фізично ноти є вібрації в межах діапазону звукових частот людини. Всього нот 7: . Слід додати, що стандартом є представлення нот латинецею, а саме послідовність наведена вище буде мати вигляд:

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 використовуюється певний ряд інтервалів. В Таблиці 1.1 показана класифікація стандартних інтервалів.

Таблиця 1.1 – Класифікація простих інтервалів [2]

|  |  |  |
| --- | --- | --- |
| 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 [3], 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 альтернаціі - бекар, який нівелює дію дієза і 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 :

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 рисунку 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 рази, нота пересувається на квінту, тобтно на 5 ступенів. Другий – множення чи ділення частоти ноти на зсуває ноту на октаву вище чи нижче відповідно. Третій – щоб побудувати частотний ряд потрібна хоча б одна єталонна частота. Але в даній системі є один ньюанс. Якщо враховувати наявність нот з приставкою бекар або бімоль в октаві, то в інтервалі квінти має бути 8 з 12ти нот.

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

Тоді після множення частоти на отримаємо ноту 5-ої ступені, а саме Мі другої октави. Щоб перенести ноту Мі на першу октаву поділемо її частоту на два.

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

Замість піднімання на 5 ступенів, можна спустити ноту на 5 ступенів поділивши на

Аналогічно підрахую інші ноти

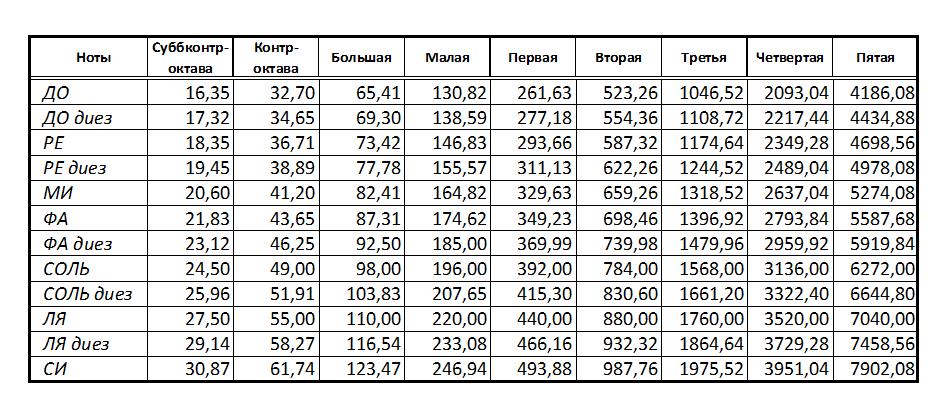
Якщо розглянути квінту від ноти Сі, то бачимо що до ноти Фа є 5 ступенів, але немає 8ми з 12 нот, що було вказано як ньюанс вище. Це означає, що збільшивши частоту ноти Сі в отримуємо ноту «Фа дієз». Таким чином можемо знайти всі 12 нот октави.

Зменшуючи частоти в від ноти Фа отримуємо бімолі.

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».

Найпоширеніший з систем розподілу частот – рівномірно темперований лад [4]. Така система будується значно простіше. Вона стверджує, що частоти нот інтервал між якими дорівнює полутону відносяться як . За стандарт частоти прийнята нота Ля першої октави з частотою 440Гц. На Таблиці 1.2 представлена відповідність нотам їх частот за системою рівномірно темперованого ладу.

Таблиця 1.2 – Частоти нот кожної октави



Неважко помітити, що зі збільшенням октави частоти нот збільшуються як показова функція з основою рівною 2. На Рисунку 1.4 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.

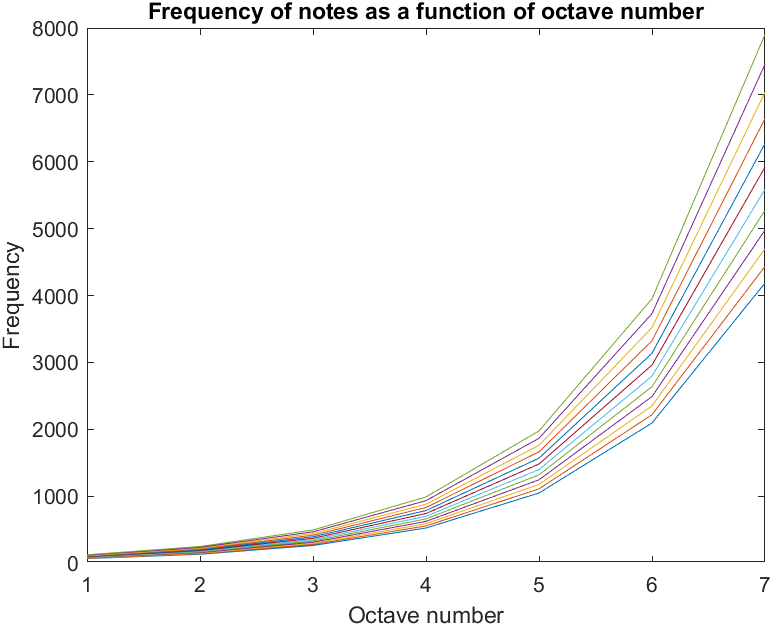


Рисунок 1.2 – Графік відповідністі нотам октав частот

## Фізика тембру

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 видававзвук ноти, його певна частина повинна задовольняти 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 рисунку 1.3.



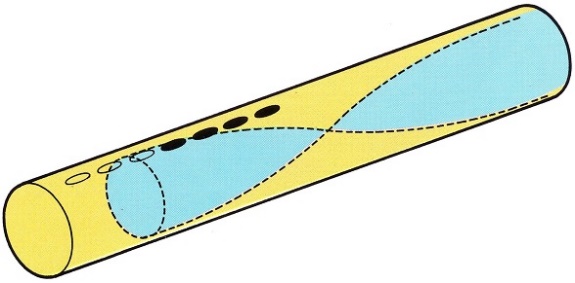
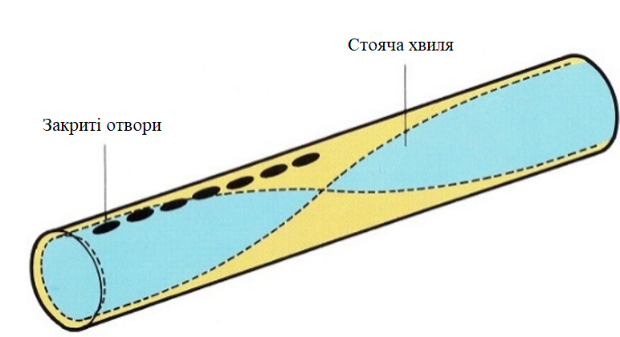


Рисунок 1.3 – Формування стоячої хвилі в трубі в залежності від стану клапонів[5]

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 рисунку 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»[6]. 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 тоні з часом.

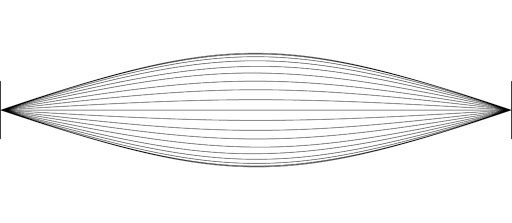


Рисунок 1.5 – Коливання струни на основному тоні в перерізах часу

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 Рисунку 1.6.

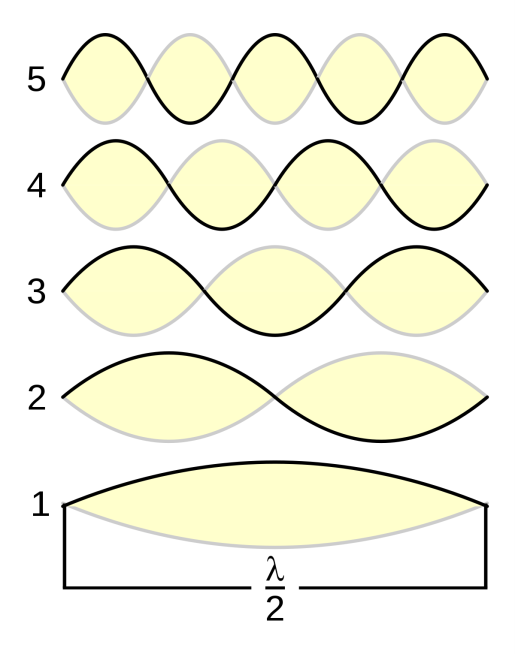


Рисунок 1.6 – Зображення довжин хвиль, що збуджуються в струні

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 Рисунку 1.7 можна побачити розподіл амплітуд обертонів деяких інструментів.

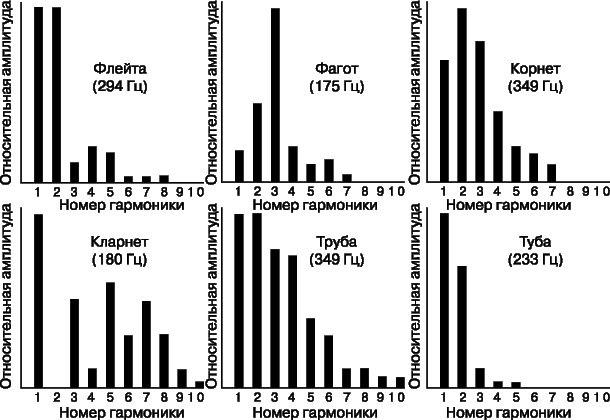


Рисунок 1.7 – Спектральні показники тембру флейти, фаготу, корнету, кларнету, труби та туби

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

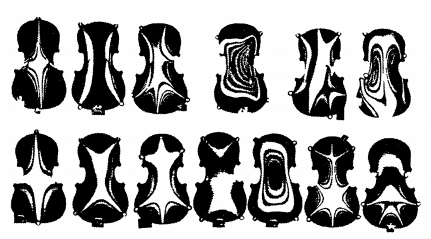


Рисунок 1.8 – Форми коливань верхньої і нижньої деки скрипки[7]

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 Рисунку 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 Рисунку 1.10 можна подивитися на часові спектрограми флейти і гобоя, а на Рисунку 1.11 на відношення обертонів гітарної струни в часі.

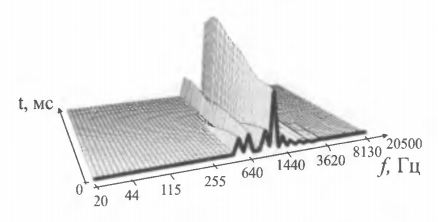
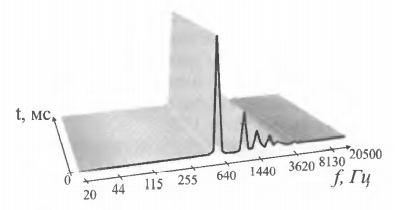


Рисунок 1.10 – Часові спектрограмми зліва направо: флейта, гобой[7]

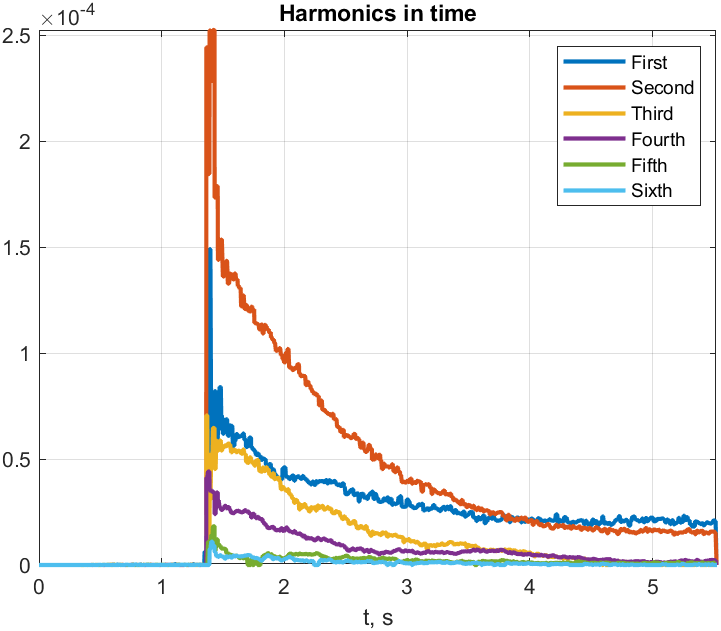


Рисунок 1.11 – Відношення обертонів до основнго тону в часі при збуджені струни гітари

# Теорія класифікації

## 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 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. [книжка по статистике, забыл какая]

Нехай - множина об'єктів, - множина номерів класів. Тоді існує певна невідома залежність , значення якої відомі тільки для скінченної вибірки для навчання . Задачею класифікації є побудова алгоритма що здатен класифікувати довільный об’єкт .

Характеристикою є відображення , де – множина допустимих значень характеристик. Об’єкт може бути представленим через вектор характеристик так, що . Характеристики можна ототожнювати із самими об'єктами. Нехай кожен об’єкт множини має ознак , а множина класів містить класів .

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 , де значно менше кількості слів в словнику. Так слова розділяються на під-змісти, синоніми знаходяться ближче один до одного в 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.

## Байесівска класифікація. Гаусовський наївний баєсів класифікатор

Почнемо з ймовірнісного формулювання задачі класифікації. Припускається що множина пар «об'єкт, клас» є ймовірнісним простором з невідомою ймовірнісною мірою . Є скінченна навчальна вибірка спостережень , згенерована згідно з ймовірнісною мірою . Необхідно побудувати алгоритм здатний класифікувати довільный об’єкт .

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 . 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 в 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. [11]

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 максимальна.

Приймаючи за множину класів, де один клас, X за множину об’єктів, за об’єкт, а за алгоритм класифікації, рішення про належність об’єкту до класу може бути визначено за формулою (1.1)

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

Згідно з теоремою Байеса вираз апостеріорної ймовірності вище може бути переписано в більш явному вигляді (1.2).

|  |  |
| --- | --- |
|  | () |

де – ймовірність що у об’єкта класу будуть ознаки ;

– апріорна ймовірність зустрічі классу серед множини об’єктів;

– апріорна ймовірність з якою зустрічається такий набір ознак.

При пошуку максимуму з формули константами з виразу апостеріорної ймовірності (1.2) можна знехтувати. Так, не залежить від класу, отже може бути відкинутим. Якщо розглядати на практиці, то частіше всього , так як фотографії пікселі яких повністю співпадають не зустрічаються, якщо тільки одна фотографія не зроблена з першої. (?)

Застосовуючи наївну БК припустимо, що ознаки , не залежать одне від одного. Тоді чисельник формули (1.2) можна спростити.

де – 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.

Неперервні залежності, такі як , як правило, оцінюються через нормальний розподіл. Тобто на практиці БК використовує підходи Гаусовського класифікатора[12] (надалі ГК). Принцип ГК полягає в тому, що залежності нам відомі заздалегідь і ми знаємо що їх форма розподілу – гаусівська (1.5). Такий підход є популярним, так як багато природніх процесів, що можуть класифікуватись мають нормальний розподіл параметрів.

де – математичне очікування класу для ознаки ;

– середнеквадратичне відхилення тої ж ознаки.

В якості математичного очікування і дисперсії обчислюються середнє арифметичне (1.6) і середнє квадратичне відхилення (1.7) відповідно.

де M – кількість об’єктів з навчальної вибірки класу , відповідно – i-та ознака p-го об’єкту.

Відповідно для класифікації Наївним БК з використанням принципів ГК потрібно буде зберігти математичних очікувань та середньоквадратічніх відхілень. Тобто на кожен клас з множени потрібно зберігти параметри нормального розподілу для кожної ознаки .

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[13]. Згідно статті[14] видно, що в порівнянні з деревом ухвалення рішення та методом k-найближчих сусідів наївний БК показав, що він значно класифікує з вищою точністью та в значно меншій мірі залежить від параметрів оптимізації, хоча займає більше часу на обрахунки.

## SVM

SVM, який ще називають методом опорних векторів або опорно-векторними машинами, 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.[15]

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 характеристиками, то такий об’єкт можна представити як точку в вимірному просторі. На Рисунку 2.1 показан приклад зображення об’єктів двох класів, що описуються двома ознаками – . Відповідно цифрові значення ознак можуть виступати координатами точок як об’єктів.

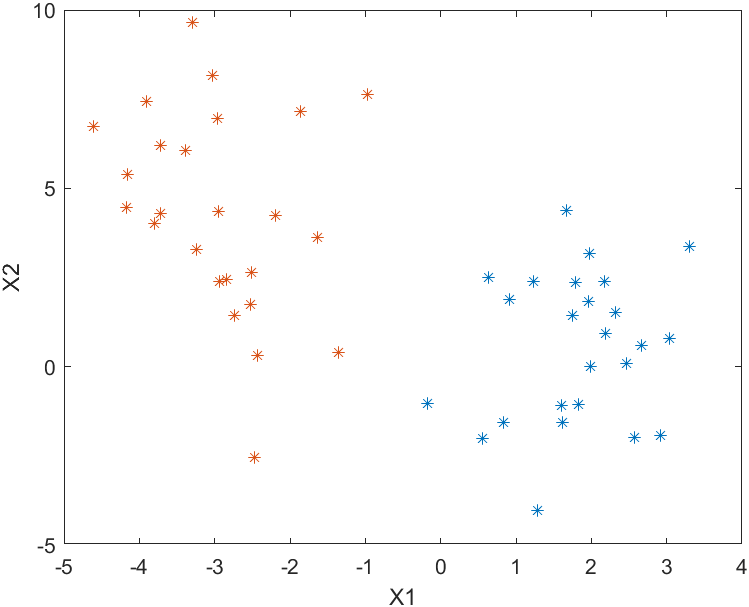


Рисунок 2.1 – Приклад зображення об’єкту з двома ознаками як точки на площині

Метод опорних векторів полягає в тому, що регресивно може бути побудована така площина, що буде побудована рівновіддалено до найближчих точок кожного классу. Вона буде розділяти точки 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 вимірного об’єкту класифікатор SVM будує 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.

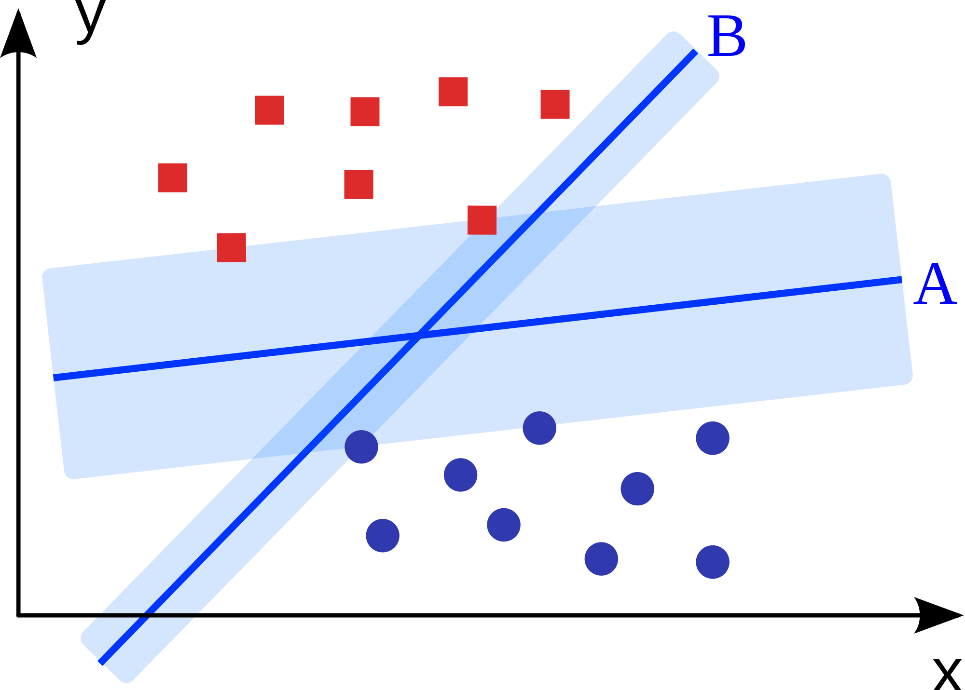


Рисунок 2.2 – Зображення оптимальної (А) та неоптимальної (В) роздільної гіперплощини

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 Рисунку 2.2 можна побачити данні двох класів – червоні та сині точки, що описуються двома ознаками - . 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 точок вигляду , де приймає значення 1 або -1, що відповідає першому або другому классу, а кожна точка є -мірним вектором ознак. Задача методу знайти максимально розділову гіперплощину яка відділяє групу точок , для яких , від групи точок, для яких , і визначається таким чином, що відстань між цією гіперплощиною та найближчою точкою з кожної з груп є максимальною. Гіперплощину можна записати як рівняння (1.8)

де – вектор нормалі до гіперплощини;

– 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 (1.9). Графік з розділенням та вказівками на паралельні площини можна побачити на Рисунку 2.3.

З геометричної точки зору, відстанню між цими двома гіперплощинами є , тож для максимізації відстані між ними нам треба мінімізувати .

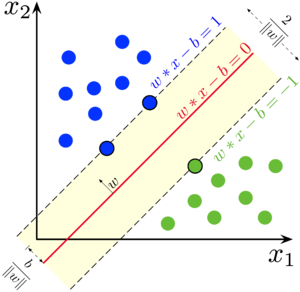


Рисунок 2.3 – Оптимальна розділяє гіперплощина в та вказывки на паралельні гіперплощини для визначення проміжку [15]

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

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 , які лежать найближче до неї. Ці 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

Така функція помилки поверне нуль в тому випадку, якщо нерівність (1.10) задовільняється. Для інших випадків значення помилки буде пропорційним до відстані від розділення. В такому разі мінімізувати треба функцію (1.11), а не .

де – параметр що визначає важливість між двома критеріями – збільшення проміжку та задовільнення умови (1.10).

Одним із методів класифікації данних, що не є лінійно розділенними – перехід до більших вимірів, 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 Рисунку 2.4. Для того, щоб перетворити такі дані в ті, що будуть лінійно розділенними в вищому вимірі використовують ядрові функції . Тоді лінійна гіперплощина у вищому вимірі ознак може бути нелінійною функцією в первинному просторі де потребужться класифікувати дані.

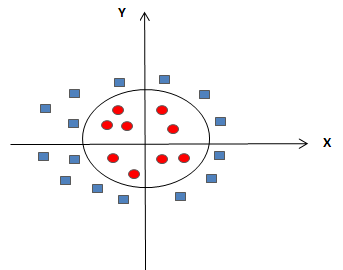


Рисунок 2.4 – Приклад нерозділених лінійно даних [16]

Найпоширеніші ядрові функції :

* Поліноміальне однорідне.
* Поліноміальне неоднорідне.
* Ґаусова радіально-базисна функція. .
* Сигмоїдне (гіперболічний тангенс).

де – відповідні вектори ознак.

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

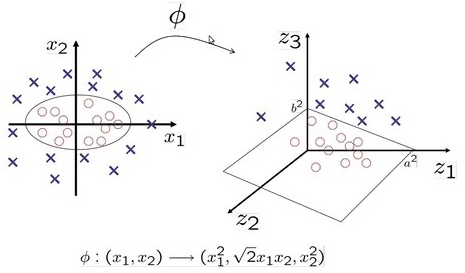


Рисунок 2.5 – Результат збільшення розмірності данних задля відокремлення данних гіперплощиною [16]

Ще одна проблема що може виникнути при використанні данного методу – це багатоклассова 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 класів використовують 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: «A» 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. [17]

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.

Проте, Райан Ріфкін та Альдебаро Клаутау мають наукову статтю, що доказує більшу ефективність останнього методу. [18]

# Дослідження методів классифікації

## Вибір параметрів вхідного сигналу для класифікації

### 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 – .

Швидке перетворення Фур'є [10] – це алгоритм, який дозволяє скоротити час розрахунку дискретного перетворення Фур'є. Тоді як звичайне дискретне перетворення Фур'є виконує розрахунки за час , швидке перетворення Фур'є справляється за

### 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 Рисунку 3.1.

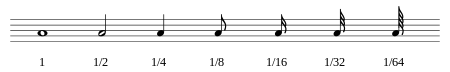


Рисунок 3.1 – Зображення нот різної тривалості

Тривалість ноти є відносною велечиною, поки не заданий темп 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 200.

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

де – тривалість четвертної ноти;

– 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 мен яскраві по тембру, то візьмемо для підрахунків найнижчу частоту дискретизації. Тоді отримаємо, що найкоротша нота буде записана кількістю відліків

(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 або ???)

### Використання інтерполяції спектру

Маючи кількість відліків найменшої ноти і частоту дискретизації можемо знайти частотний крок в спектрі.

Перше, що можна помітити - це частотний крок, який більший ніж крок між деякими нотами. З Таблиці 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 . 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 Рисунку 2.2.

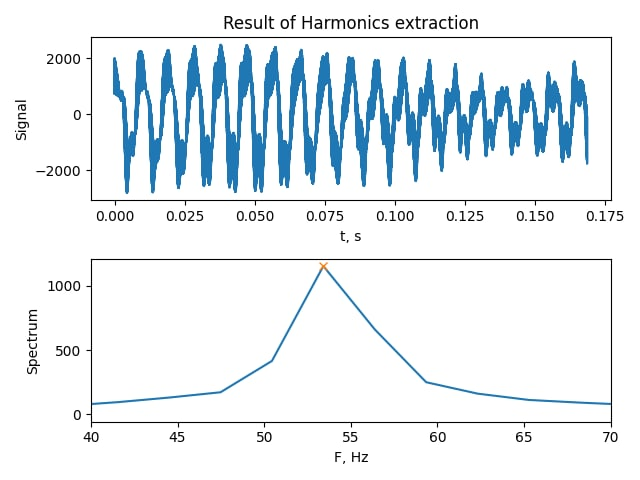
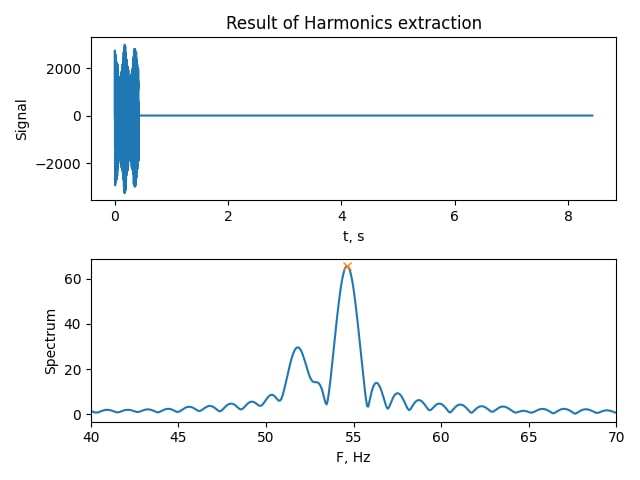
 

Рисунок 3.2 – Сигнали та спектри звучання двох сусідніх низькочастотних нот. Перший випадок - акустичний сигнал в якому закладено звук двох нот узятий з такою довжиною, щоб частотний крок перевищував відстань між нотами в спектрі. Другий випадок - той же сигнал був доповнений нулями в часовій області,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

## 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 повністью схаратеризована базовою частотою, відповідну частоту для кожної ноти можна подивитися на Таблиці 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 кратні 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. Тому найбільшу ініціюючу амплітуду отримують 2га та 3тя гармоніка, що і можна побачити на Рисунку 1.13, 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 , що буде показано пізніше, 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 Рисунку 3.1. 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.

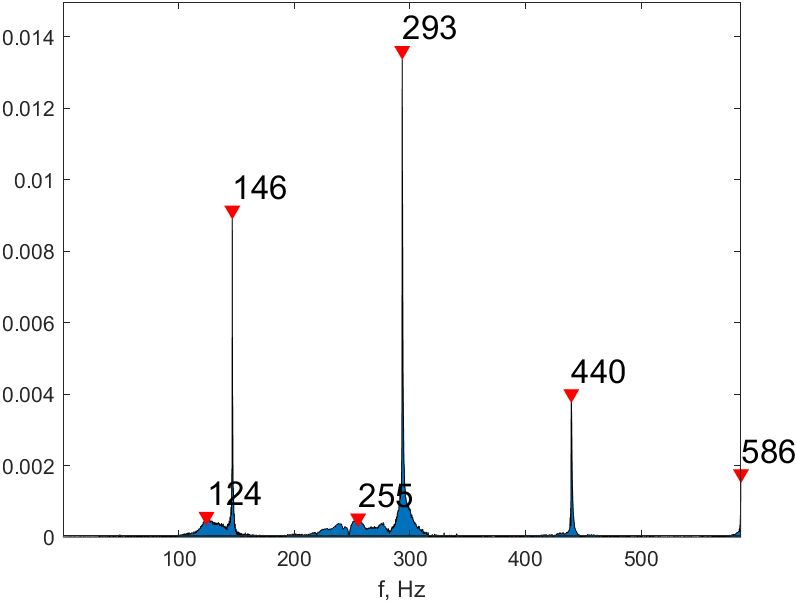


Рисунок 3.3 – Спектр ноти Ре малої октави гітари з низьким пороговим значенням

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 Рисунку 3.2 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 єтапі – потрапляють в другий. Таким чином можна визначити найменшу частоту коливання і класифікувати сигнал.

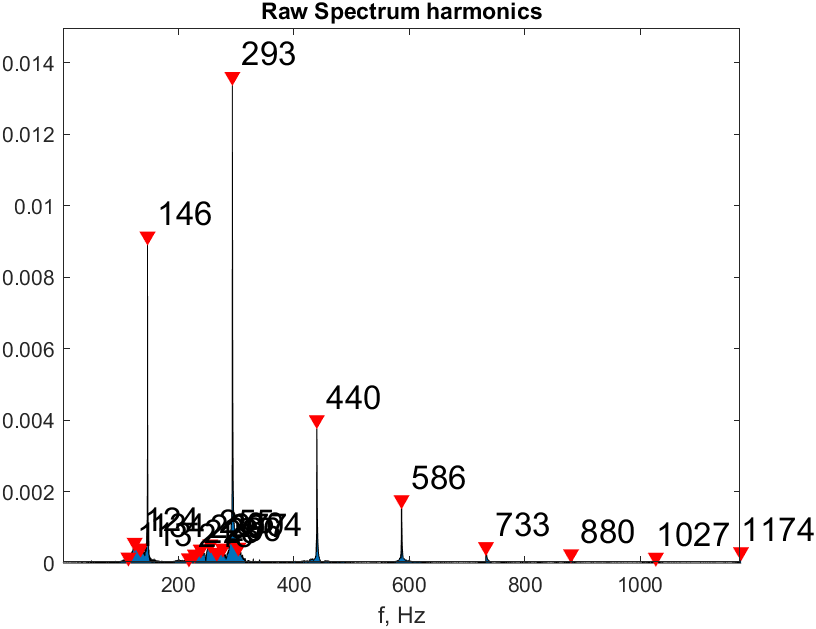
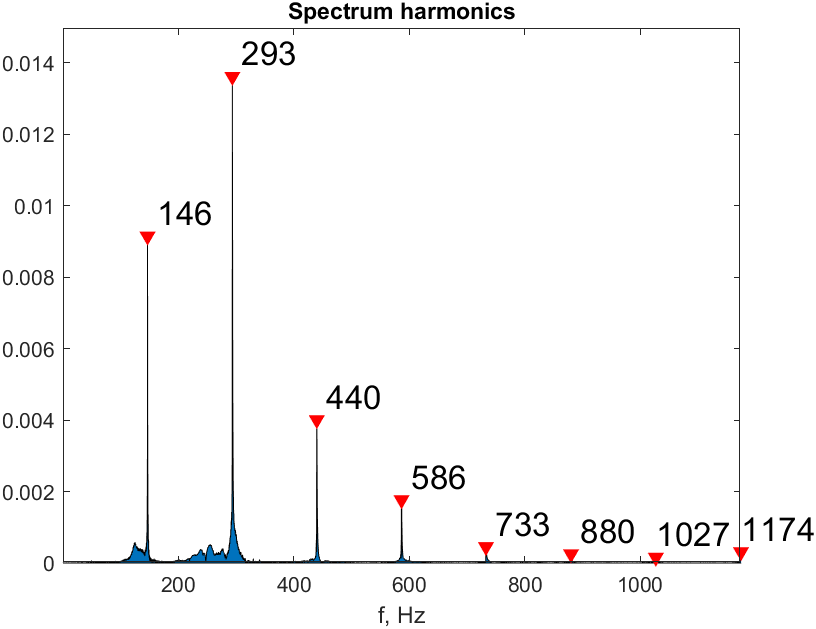
 

Рисунок 3.4 – Спектр ноти Ре малої октави гітари. Зліва спектр виділяє всі локальні мінімуми більші за деяке низбке порогове значення. Справа локальні максимуми фільтруються по принципу кратності

Проте, як було сказано раніше, зі збільшенням частоти коливання збільшується і коефіцієнт затухання. Тому, починаючи з третьої (не перевірено) 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 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 Таблиці 1.2.

Далі будується матриця така, що

де – 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.

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, що нижче нуля. Для даної задачі була використана матриця показана на (1.14)

Таким чином, якщо матриця є

де – це абсолютне значення амплітуди інтерпольованого спектру на частоті ноти октави .

То матриця

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

На Рисунку 3.5 можна побачити відображення матриці , 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.

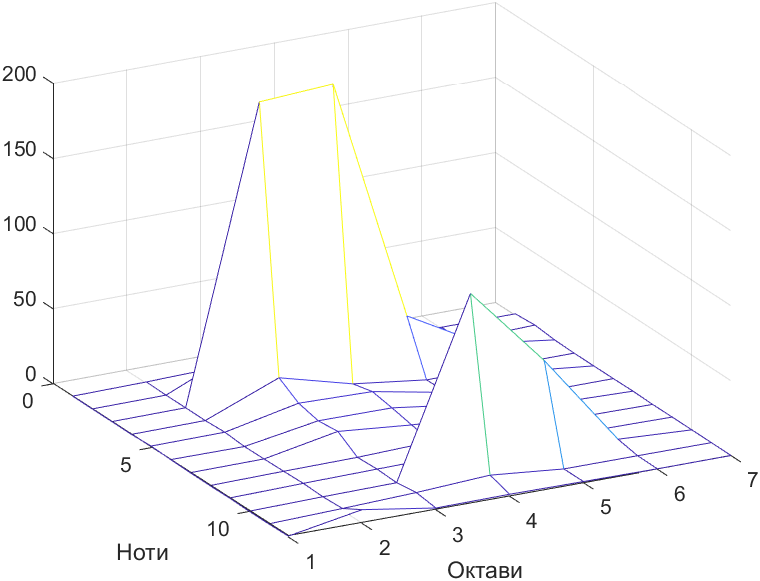
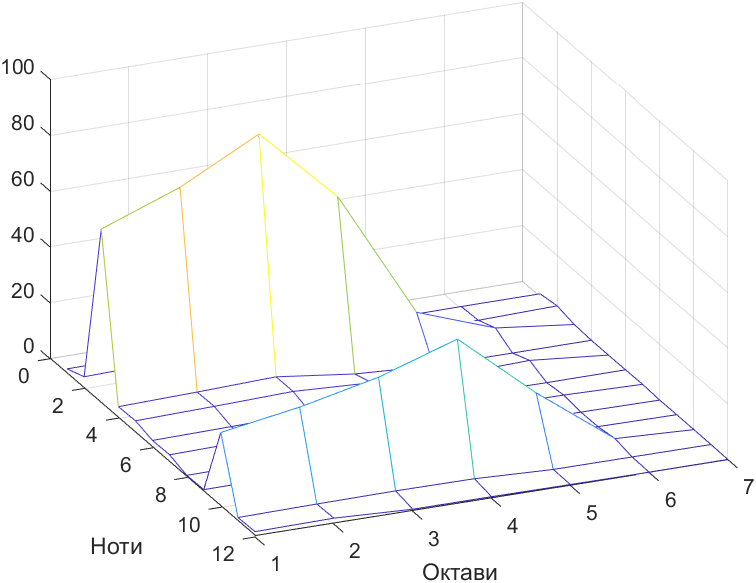


Рисунок 3.5 – Матриця М для 3ої ноти 3ої октави

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 , зображену на Рисунку 3.6. Згідно з такою матрицею 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 можна здійснити за допомогою підналаштування вектору .



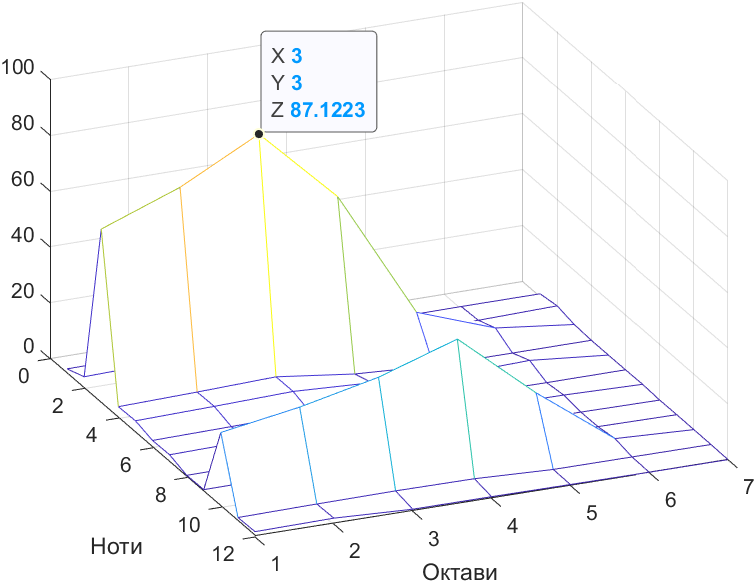


Рисунок 3.6 – Зображення матриці Ms з чітким виявленням ноти

Прийнявши ширину вікна як , використовуючи запис з 84ма нотами, де кожна нота звучить рівно 2 секунди, починаючи з контр-октави та закінчуючи четвертою октавою, при частоті дискретизації , маємо 86 акустичних сигналів на ноту та 7224 сигналів взагалом. Результат класифікації 7224 сигналів з запису зображений на Рисунку 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 .

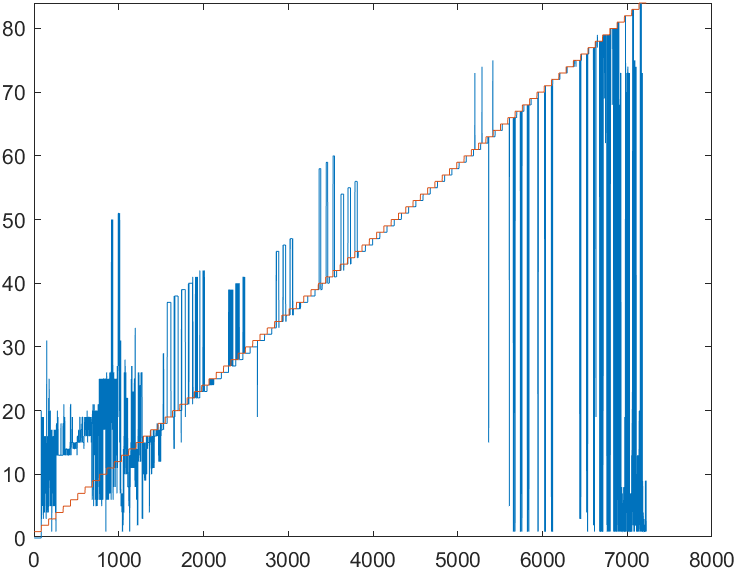


Рисунок 3.7 – Результат класифікації емпіричним методом на основі сум октав

## 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 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. Таким чином, якщо спектральний відлік 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, но 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 (1.3) 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 збільшилася. На Рисунку 3.8 можна побачити розподіл дисперсій для кожної ноти відповідно до кожного відліку спектру.

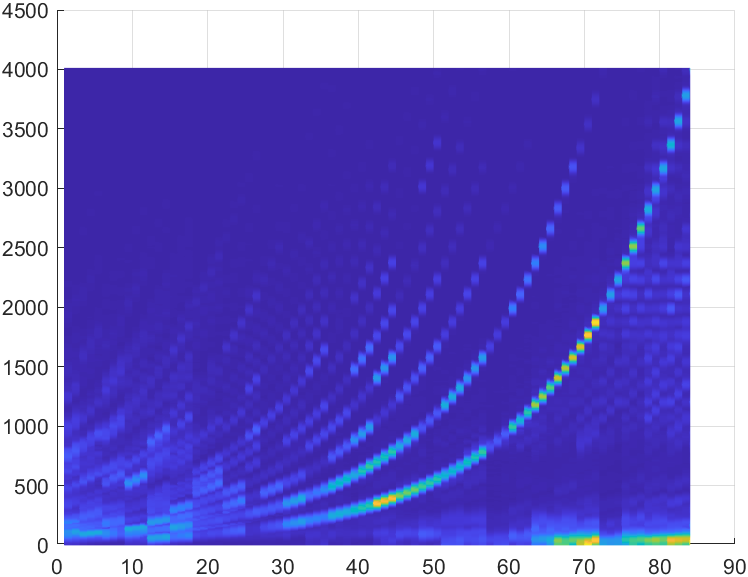
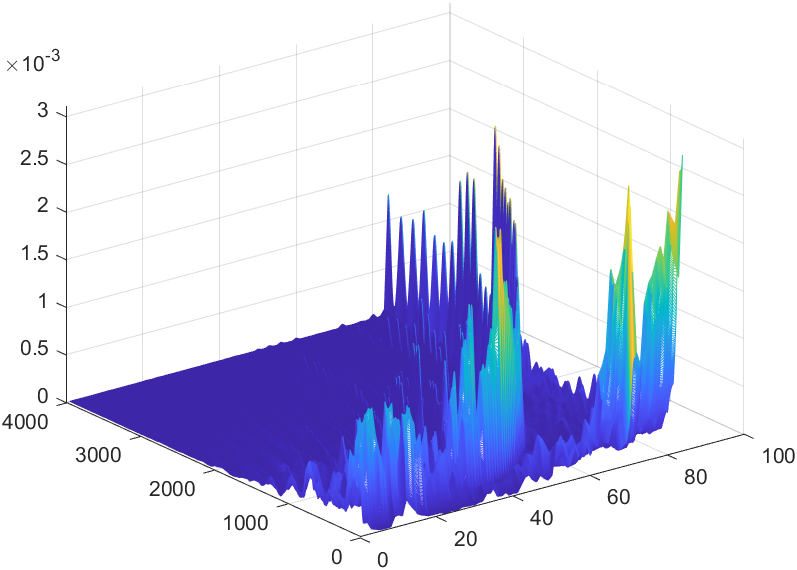


Рисунок 3.8 – Зображення дисперсії для кожного відліку спектру для 84 класифікації

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

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

Відповідно час на підрахунки для одного сигналу з тестової вибірки

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

Таким чином тестова вибірка буде підрахована за час

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 (1.4), де добуток ймовірностей був замінений суммою логарифмів. З урахуванням жорсткої апроксимації гауса через перевірку відстані за правилом 3 сігма та послідовним присвоюванням ймовірності , був застрахований випадок де логарифм видавав бі від’ємну нескінченність.

Результат класифікування тестової группи сигналів можна побачити на Рисунку 3.9. Відсоток помилки становить 8.5%. Час на підрахунок становить 56 хвилин.

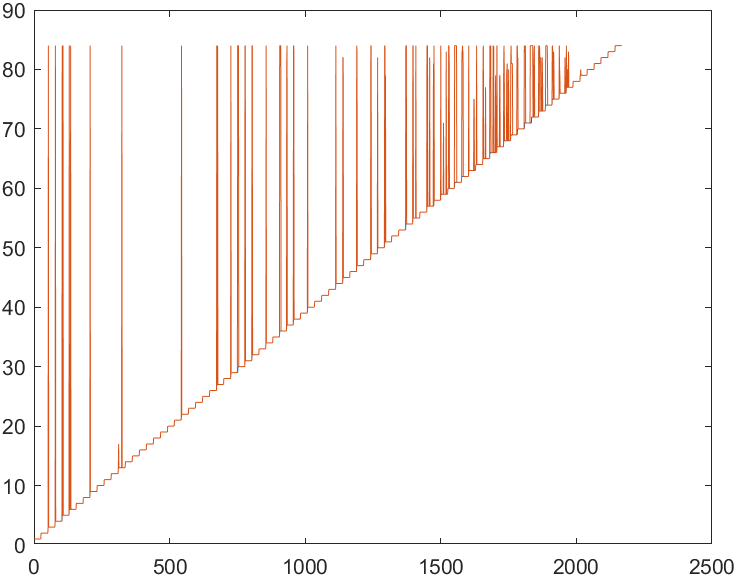


Рисунок 3.9 – Результат класифікації БК по спектру

Згідно формули (1.16) 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 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 , яких у свою чергу всього 84. Фактично, використовуючи матрицю замість спектру – класифікатор ігнорує всі відліки, що не є фактичними відповідностями до нот, тобто є менш важливими. В той час як використання матриці замість 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 наведено в Додатку В.

В той час як при тренуванні БК на матриці було отримано відсоток помилки 76%, при використанні матриці маємо результат зображений на РИС1, 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 відсоток помилки може бути зменшиний.

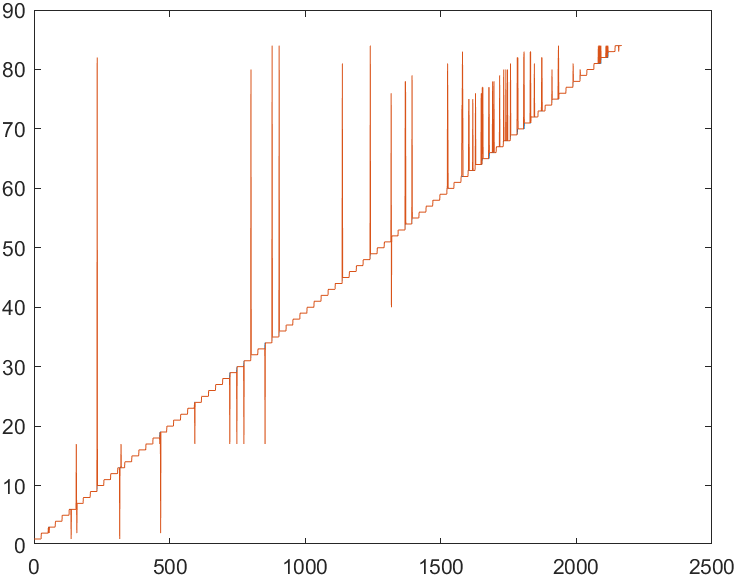


Рисунок 3.10 – Результат класифікації БК по матриці Ms

## SVM класифікатор

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. Класифікатор SVM на основі спектральних відліків тренерувався 4хв 46с, прокласифікував тестовий ряд сигналів за 1хв 28с, проте показав якість в 86% помилок.

Застосування емпірично виведеної матриці для навчання та класифікування не змінило вихідні характеристики. Час навчання становить 25с, час класифікації ряду тестових сигналів становить 6с, але відсоток помилок 87%. Результат класифікації таким класифікатором можна побачити на Рисунку 3.11. Пояснити такий результат можна тим, що в реалізації класифікатору була використана вбудована в MATLAB функцыя навчання SVM моделі і можливо не була підібрана така ядрова функція, яка б надавала можливість якісної класифікації.

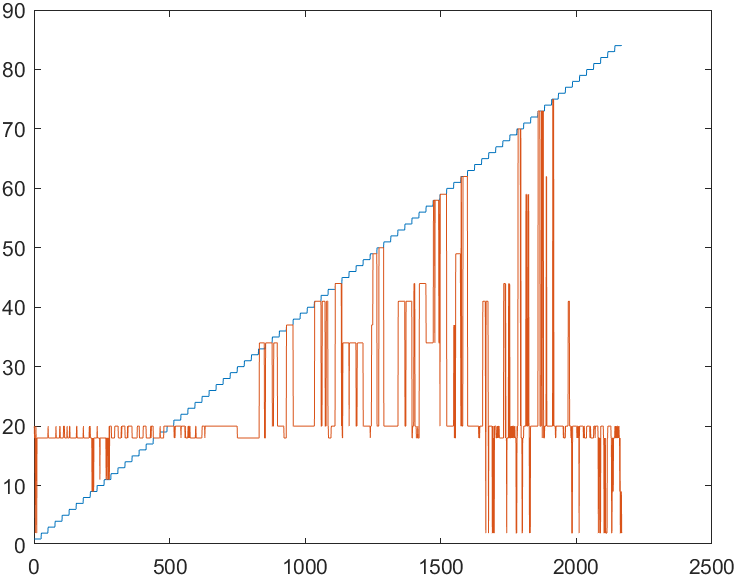


Рисунок 3.11 – Результат класифікації SVM по матриці Ms

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 ./ sum(XTest\_ds, 2);

XTrain\_ds = XTrain\_ds ./ 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]); %

% 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) + '%')

References

[5] Статья про тона инструментов

<http://information-technology.ru/sci-pop-articles/23-physics/268-kak-poluchayut-raznye-tona>

[6] Пособие – «Колебание струны». Математическая физика

<http://gukitkafmi.narod.ru/files/INShitov/string.pdf>

[7] Учебник – Музыкальная акустика, Ирина Алдошина и Рой Приттс

[8] Регрессионный анализ –

<https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7>

[9] Метод опорных векторов –

[https://neerc.ifmo.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4\_%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D1%8B%D1%85\_%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BE%D0%B2\_(SVM)#:~:text=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%20%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D1%8B%D1%85%20%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BE%D0%B2%20(%D0%B0%D0%BD%D0%B3%D0%BB.,%D1%80%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D1%8F%D1%8E%D1%89%D0%B5%D0%B9%20%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D1%8B%20%D0%B2%D1%8B%D0%B1%D0%BE%D1%80%D0%BA%D0%B8%20%D0%BE%D0%BF%D1%82%D0%B8%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%BC%20%D1%81%D0%BF%D0%BE%D1%81%D0%BE%D0%B1%D0%BE%D0%BC](https://neerc.ifmo.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D1%8B%D1%85_%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BE%D0%B2_(SVM)#:~:text=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%20%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D1%8B%D1%85%20%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BE%D0%B2%20(%D0%B0%D0%BD%D0%B3%D0%BB.,%D1%80%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D1%8F%D1%8E%D1%89%D0%B5%D0%B9%20%D0%BE%D0%B1%D1%8A%D0%).

[12] Гаусовський класифікатор –

<https://learnmachinelearning.wikia.org/ru/wiki/%D0%93%D0%B0%D1%83%D1%81%D1%81%D0%BE%D0%B2_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80>

[13] База данних для касифікування вина

<https://archive.ics.uci.edu/ml/datasets/Wine>

[14] Стаття про застосування Байесівського класифікатора

<https://blog.uiam.sk/the-naive-probability-is-an-expert/>

[15] SVM –

<https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D0%B8%D1%85_%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D1%96%D0%B2>

[16] Стаття про використання SVM на мові R –

<https://www.listendata.com/2017/01/support-vector-machine-in-r-tutorial.html>

[17] Емпіричне порівняння методів багатокласової класифікації SVM –

<http://www.keerthis.com/multiclass_mcs_kaibo_05.pdf>

[18] Дослідження порівняння методів багатокласової класифікації SVM –

<https://www.jmlr.org/papers/volume5/rifkin04a/rifkin04a.pdf>