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

Факультет інформатики та обчислювальної техніки
(повна назва інституту/факультету)

Кафедра автоматизованих систем обробки інформації і управління
(повна назва кафедри)

«На правах рукопису»

УДК *004.023*

«До захисту допущено»
В.о. завідувача кафедри
Олександр ПАВЛОВ
(підпис) (ініціали, прізвище)

«_____» _____ 2021 р.

Магістерська дисертація
на здобуття ступеня магістра

за освітньо-науковою програмою *«Інженерія програмного забезпечення комп'ютеризованих систем»*

зі спеціальності *121 «Інженерія програмного забезпечення»*

на тему *«Математичне та програмне забезпечення обробки ЕКГ сигналів»*

Виконала *студенткаVIкурсу ФІОТ, групи ІП-91мн*
Реутська Світлана Віталіївна
(прізвище, ім'я, по-батькові) (підпис)

Керівник *доц., к.т.н., викладач кафедри АСОІУ, К. І. Ліщук*
(посада, науковий ступінь, вчене звання, ініціали, прізвище)(підпис)

Рецензент *доц. каф. ТК, к.т.н., доц. О.І. Лісовиченко*
(посада, науковий ступінь, вчене звання, ініціали, прізвище) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студентка _____
(підпис)

Київ – 2021 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра автоматизованих систем обробки інформації і управління

Рівень вищої освіти *другий (магістерський)*
(повна назва)

Спеціальність *121 «Інженерія програмного забезпечення»*
(код та назва)

Освітньо-наукова програма *«Інженерія програмного забезпечення комп'ютеризованих систем»*
(повна назва)

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
Олександр ПАВЛОВ
(підпис) (ініціали, прізвище)
«_____» _____ 2021 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Реутській Світлані Віталіївні
(прізвище, ім'я, по-батькові)

1. Тема дисертації *Математичне та програмне забезпечення обробки ЕКГ сигналів*

науковий керівник *к.т.н., доц., викладач кафедри АСОІУ,Ліщук Катерина Ігорівна*
(посада, науковий ступінь, вчене звання, ПІБ)

затверджені наказом по університету *№ 910-с* від «*24*»*березня* 2020 р.

2. Термін подання студентом дисертації «*6*» *травня* 2021 р.

3. Об'єкт дослідження*Процес аналізу та класифікації даних для задачі обробкиЕКГ сигналів*

4. Предмет дослідження *Методи та засоби аналізу та класифікації даних для задачі обробки ЕКГ сигналів*

5. Перелік завдань, які необхідно розробити *Дослідити існуючі методи розпізнавання та аналізу ЕКГ сигналів; дослідити наявні алгоритми машинного навчання, що можуть використовуватися для аналізу ЕКГ; удосконалити існуючі підходидо параметризації ЕКГ даних;удосконалити існуючі підходи до аналізу ЕКГ даних штучним інтелектом;виконати експериментальні дослідження запропонованихрішень.*

6. Орієнтовний перелік графічного матеріалу*схема структурна класів, схема структурна бізнес-процесукласифікації ЕКГ,демонстраційний плакат, графікиефективності алгоритму.*

7. Орієнтовний перелік наукових публікацій *«Сучасні методи програмного виділення pqrst інтервалів в електрокардіограмі», «ECG signalprocessingbasedonlinguisticchainfuzzysets», «Підхід до виявлення аномалій в даних екг»*

8. Консультанти розділів дисертації

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

9. Дата видачі завдання «» 2020 р.

Календарний план

№ з/п	Назва етапу виконання магістерської дисертації	Термін виконання	Примітка
1	*Ознайомлення з предметною областю, вивчення літератури*	*01.12.2019*	
2	*Аналіз наявних методів та засобів автоматизаціїрозмітки даних*	*01.03.2020*	
3	*Постановка та формалізація задачі дослідження*	*20.03.2020*	
4	*Аналіз вимог до програмного забезпечення*	*25.03.2020*	
5	*Розробка методів та засобів для розв'язання поставлених задач*	*30.06.2020*	
6	*Розробка архітектури програмного забезпечення*	*01.11.2020*	
7	*Виконання експериментальних досліджень*	*01.02.2021*	
8	*Оформлення пояснювальної записки*	*15.04.2021*	
9	*Подання дисертації на попередній захист*	*20.04.2021*	
10	*Подання дисертації на рецензію*	*07.05.2021*	
11	*Подання дисертації на захист*	*15.05.2021*	

Студент _____ СвітланаРЕУТСЬКА
(підпис)(прізвище та ініціали)

Науковий керівник _____ КатеринаЛЩУК
(підпис)(прізвище та ініціали)

РЕФЕРАТ

Магістерська дисертація: 102 с., 39 рис., 13 табл., 3 додатки, 18 джерел

Актуальність теми.За статистичними даними основною причиною смертності серед людей працездатного віку є серцево-судинні захворювання.Люди, що входять в групу ризику, потребують допомоги шляхом надання послуг раннього виявлення. Для діагностики сердечно-судинних захворювань широко застосовується електрокардіографічний метод.Часто патологічні зміни в серці дуже швидко відображаються в інформаційному потоці. Зараз програмне забезпечення має піднімати тривогу, якщо пацієнт перебуває в ризикованому становищі. Проте на даний момент воно часто генерує помилкові тривоги в більше ніж 80% випадків. Якість відповіді такого аналізу можна вважати випадковою.

Саме тому вирішення задачі пошуку вдалого алгоритму для обробки ЕКГ сигналу та його аналізу є дуже актуальним.

Зв'язок роботи з науковимипрограмами, планами, темами. Робота виконувалась на кафедріавтоматизованих систем обробкиінформаціїіуправління в рамках теми «Інтелектуальніметодипрограмування, моделювання іпрогнозування з використаннямймовірністного і лінгвістичнихпідходів» (державнийреєстраційний номер 0117U000926)

Мета і завдання дослідження.Основна *мета*роботи полягає в розробці математичного забезпечення для аналізу ЕКГ сигналів та підбору найкращого алгоритму машинного навчання для класифікації цих даних, інтеграції розробленого алгоритму в відповідне програмне забезпечення,що дасть можливість підвищити точність та універсальністьалгоритмів розпізнавання хвороб у пацієнта по ЕКГ. Щоб досягнути поставленої мети необхідно забезпечити повний опис усіх ознак часового рядуза допомогою параметрів, а такожнаблизити алгоритм роботипрограми до алгоритму роботи мозку людини під час аналізу.

Для досягнення поставленої мети необхідно розв’язати комплекс наступнихвзаємопов’язаних**задач**:

– дослідити та порівняти наявні алгоритми аналізу тарозділення часових рядівЕКГна набір параметрів;

– дослідити та порівняти наявні підходи до класифікаціїчасових рядівЕКГна випадок хвороб;

– створити власні алгоритми розділення, аналізу та класифікаціїчасових рядівЕКГ, що будуть працювати з прийнятною точністю та охоплювати різні випадки аномалій в синусоїді;

– реалізувати запропоновані алгоритми у вигляді незалежних бібліотек;

– розробка програмного забезпечення для розпізнавання хвороб у пацієнта по ЕКГ;

– дослідити розроблені алгоритми на ефективність роботи.

Об’єкт досліджень. Процес аналізу ЕКГ, розбиття часових рядівна ознаки та їхкласифікації за цими ознаками.

Предмет досліджень.Алгоритми та методи аналізу та класифікаціїчасових рядівЕКГ на основі машинного навчання.

Наукова новизнаотриманих результатів полягає в створенні нового методу аналізу та класифікаціїчасових рядівсинусоїдального типу на прикладі ЕКГ даних із застосуванням штучного інтелекту, що має кращу ефективність.

Практичне значення отриманих результатів полягає застосуванні розробленого методув застосунку для аналізу ЕКГ даних.

Апробація результатів роботи. Результати роботи доповідались на «VIвсеукраїнській науково-практичній конференції молодих вчених та студентів «Інформаційні системи та технологій управління» (ІСТУ-2021), ConferenceonComputationallinguisticsandintelligentsystems(CoLInS 2021).

Публікації.Матеріали роботи опубліковані:

– взбірнику тез конференції «MODS 2020»РеутськаС. В., Баклан І.В., Олійник Ю.О., Ліщук К.І. «Підхід до виявлення аномалій в даних екг»;

– в збірнику тез ІІІМіжнародної науково-практичної конференції«Сучасні тенденції розвитку інформаційних систем і телекомунікаційних технологій»РеутськаС. В.«СУЧАСНІ МЕТОДИ ПРОГРАМНОГО ВИДІЛЕННЯ PQRST ІНТЕРВАЛІВ В ЕЛЕКТРОКАРДІОГРАМІ»;

– в збірнику статей конференції«ComputationalLinguisticsandIntelligentSystems» (CoLins2021)I. Baklan, A. Oliynyk, I. Mukha, K. Lishchuk, O. Gavrilenko, S. Reutska, A. Tsitsyliuk, Y. Oliynyk «ECG signalprocessingbasedonlinguisticchainfuzzysets».

Ключові слова: МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ,АНАЛІЗ ЧАСОВИХ РЯДІВ, КЛАСИФІКАЦІЯ ЧАСОВИХ РЯДІВ,ДЕТЕКЦІЯ АНОМАЛІЙ, ДЕРЕВО ПРИЙНЯТТЯ РІШЕНЬ, ВИПАДКОВИЙ ЛІС, ЛОГІСТИЧНА РЕГРЕСІЯ, СТОХАСТИЧНИЙ КООРДИНАТНИЙ ПІДЙОМ

ABSTRACT

Master's degree thesis: 102 pages, 39 figures, 13 tables, 3 attachments, 18 sources.

Relevance of the topic. According to statistics, the main cause of death among people of working age is cardiovascular disease. People at risk need help by providing early detection services. Electrocardiographic method is widely used to diagnose cardiovascular diseases. Often pathological changes in the heart are very quickly reflected in the information flow. Now the software should raise the alarm if the patient is at risk. However, at present it often generates false alarms in more than 80% of cases. The response quality of such an analysis can be considered random. That is why solving the problem of finding a successful algorithm for ECG signal processing and analysis is very important.

Scientific programs, plans, topics. The work was performed at the Department of Automated Information Processing and Control Systems within the theme "Intelligent Speech Recognition System in the study of foreign words based on machine learning".

The following objectives have been formulated to achieve this research objective:

Improve the accuracy and versatility of ECG disease recognition algorithms in patients by creating a new data analysis algorithm and selecting the best machine learning algorithm to classify these data. To achieve this goal it is necessary to provide a complete description of all the features of the time series with the help of parameters, as well as to bring the algorithm of the program to the algorithm of the human brain during the analysis.

To achieve this goal, the **following tasks** were formed:

– to investigate and compare the existing algorithms for analysis and division of EC time series into a set of parameters;

– to study and compare the available approaches to the classification of ECG time series in case of diseases;

– to create their own algorithms for separation, analysis and classification of ECG time series, which will work with acceptable accuracy and cover various cases of anomalies in the sine wave;

– to implement the proposed algorithms in the form of independent libraries that can be used in the future;

– to implement a program for the user;

– to investigate the created algorithms on efficiency of work.

The object of research. The process of ECG analysis, division of time series into signs and their classification by these signs.

The subject of research. Algorithms and methods of analysis and classification of ECG time series based on machine learning.

The scientific innovation is to create a new method of analysis and classification of sinusoidal time series on the example of ECG data using artificial intelligence, which has better efficiency.

The practical value of the obtained results is to apply the developed method in the application for ECG data analysis.

Approbation. The results were presented at the VI All-Ukrainian scientific-practical conference of young scientists and students "Information systems and management technologies" (ISTU-2021).

Scientific publications. Materials of work are published:

– in the collection of abstracts of the conference "MODS 2020" Reutska SV, Baklan IV, Oliynyk YO, Lishchuk KI "APPROACH TO DETECTION OF ANOMALIES IN ECG DATA";

– in the collection of abstracts of the III International scientific-practical conference "Modern trends in the development of information systems and telecommunications technologies" Reutska SV "MODERN METHODS OF SOFTWARE SELECTION OF PQRST INTERVALS IN ELECTROCARDIOGRAPHY;

– in the collection of articles of the conference "Computational Linguistics and Intelligent Systems" (CoLins2021) I. Baklan, A. Oliynyk, I. Mukha, K. Lishchuk, O. Gavrilenko, S. Reutska, A. Tsitsyliuk, Y. Oliynyk "ECG signal processing based on linguistic chain fuzzy sets".

Keywords: MACHINE LEARNING, NEURAL NETWORKS, TIME SERIES ANALYSIS, TIME SERIES CLASSIFICATION, DETECTION OF ANOMALIES, DECISION TREE, INCIDENTAL FOREST, LOGISTIC REGRESSION, STOCHASTIC COORDINATE RECOVERY

ЗМІСТ

ВСТУП

За статистичними даними основною причиною смертності серед людей працездатного віку є серцево-судинні захворювання. Основними факторами ризику таких хвороб є неправильне харчування, відсутність фізичних навантажень, паління та вживання алкогольних чи наркотичних речовин. Люди, що входять в групу ризику, потребують допомоги шляхом надання послуг раннього виявлення, консультування та лікування.

Вкрай важливим є саме процес вчасного діагностування проблеми у хворого. В сучасних клініках для діагностики сердечно-судинних захворювань широко застосовується електрокардіографічний метод. Електрокардіограма (далі ЕКГ) – зручний, дешевий, безболісний та абсолютно нешкідливий процес зняття даних про електричні поля, що створюються роботою серця. Фактично являє собою графічне представлення різниці потенціалів. Зазвичай в програму подається ЕКГ у вигляді так званого часового ряду. Часовий ряд - це зібраний в різні моменти часу статистичнийматеріал про значення будь-яких параметрів (в найпростішому випадку одного).

Сучасне програмне забезпечення має піднімати тривогу, якщо пацієнт перебуває в ризикованому становищі.Часто патологічні зміни в серці дуже швидко відображаються в інформаційному потоці, задовго до появи видимих симптомів захворювання. Такі відображення змін люди називають аномаліями – інтервалами в часових рядах, де поведінка синусоїди не відповідає нормальній поведінці.Важливість виявлення цих змін зумовлена тим фактом, що виявлення найменшої аномалії в даних може призвести до виявлення значних порушень у прикладній області, яку описує часовий ряд.

Наразі розвиток у сфері дослідження ЕКГ програмними методами вважається перспективним.Багато вчених намагається знайти дієвий спосіб автоматизувати процес виявлення аномалій. Проте, існуюче програмне забезпечення часто генерує помилкові

тривоги, має низьку точність прогнозування, може бути застосоване в дуже вузькому колі прикладних задач. Так відбувається через високу зашумленість сигналу, слабкість алгоритмів обробки та недостатню базу ознак відхилень роботи серця.

Тобто якість аналізу та класифікації за допомогою такого програмного забезпечення можна вважати незадовільною для використання в реальних умовах. Саме тому пошук вдалих алгоритмів для обробки ЕКГ сигналу та його аналізу стає першочерговою задачею в розвитку кардіологічного лікування.

Мета роботи:полягає в розробці математичного забезпечення для аналізу ЕКГ сигналів та підбору найкращого алгоритму машинного навчання для класифікації цих даних, інтеграції розробленого алгоритму в відповідне програмне забезпечення,що дасть можливість підвищити точність та універсальністьалгоритмів розпізнавання хвороб у пацієнта по ЕКГ.

1 ВИЯВЛЕННЯ ІНФОРМАЦІЙНИХ ПОТРЕБ ПРИКЛАДНОЇ ОБЛАСТІ

Список хвороб, що можна виявити під час дослідження електрокардіаграми, вражає:

- аритмія;
- міокардит;
- вроджений порок серця;
- фібриляція передсердь;
- тахікардія;
- серцева недостатність;
- серцевий напад.

Таким чином, незважаючи на те що ЕКГ-дослідження є одним їх найстаріших діагностичних методів, воно досі займає провідне місце серед застосовуваних методів діагностування. Знання основ електрокардіографії, вміння оцінити зубці, сегменти, інтервали електрокардіограми необхідно для швидкої діагностики захворювань.

Електрокардіографія відіграє допоміжну роль в діагностиці іїї результати повинні інтерпретуватися тільки з урахуванням клінічної картини захворювання і даних інших методів дослідження. Отже програмне забезпечення, що зможе проводити попередню діагностику пацієнтів, буде проміжною ланкою між пацієнтом та реальним лікарем, дозволить значно зменшити обсяг виконання необхідних досліджень, зменшити черги в медичні заклади.

Також важливим є те, що програмна реалізація аналізу ЕКГ даних може слугувати додатковим перестрахувальним фактором в діагностиці. Таким чином вірогідність помилки через людський фактор буде зведена до мінімуму.

1.1 Опис даних електрокардіограм

Дані зчитуються з пацієнта спеціальним приладом – електрокардіографом. Пацієнт має знаходитись в зручному положенні, лежачи, кінцівки мають бути злегка зігнутими та лежати на поверхні. Для запису ЕКГ, кабель пацієнта (його розгалужену частину) приєднують до кінцівок, накладаючи металеві пластини (електроди).

Швидкість запису ЕКГ важлива для аналізу протяжності сегментів сигналу. Зазвичай при швидкості запису 50 мм/с 1мм відповідає 0,02 с, при швидкості запису 25 мм/с 1 мм відповідає 0,04 с.

ЕКГ комплекс складається з сегментів, інтервалів та зубців. На рисунку 1.1 показано основний сегмент серцевого ритму, за яким проводяться дослідження. Зазвичай основний аналіз стану пацієнта проводиться саме за зубцями в такій послідовності: зубець P, інтервал P–Q, комплекс QRS і його зубці, сегмент ST, зубці T і U. Також виділяється лінія між зубцями U та P, яка має спеціальну назву: ізоелектрична.

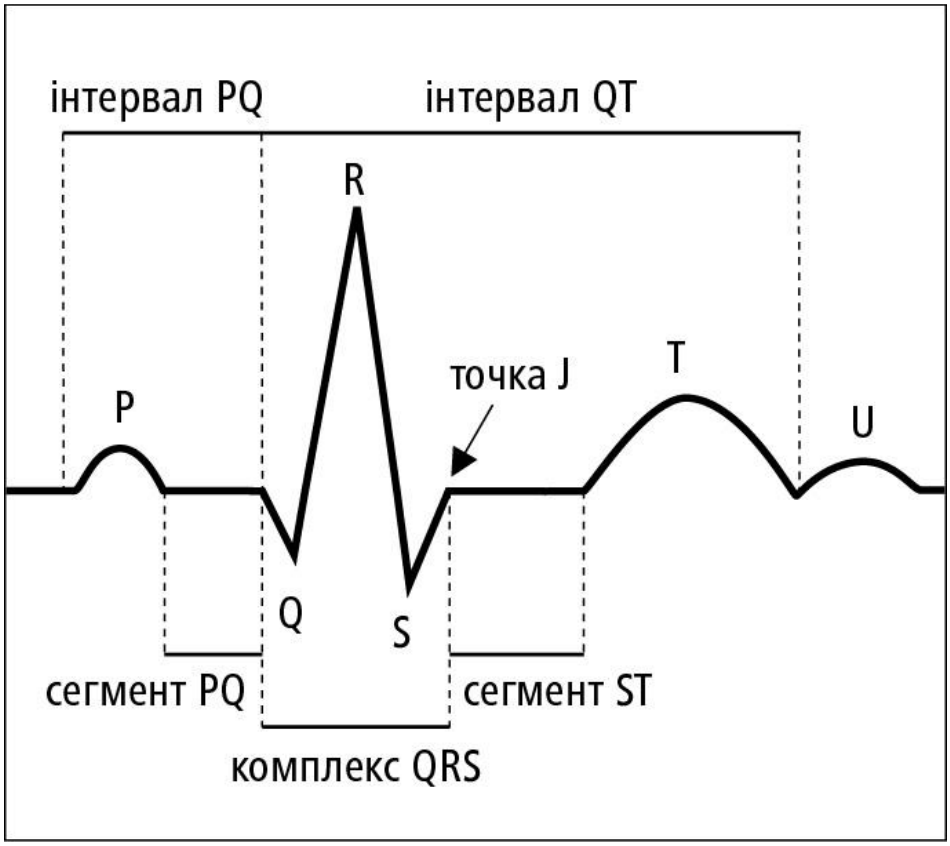


Рисунок 1.1 – Схема нормального запису ЕКГ сегменту

Головну увагу приділяють часовим вимірам інтервалів, формі зубців та співвідношенню їх амплітуд, аналізу морфології шлуночків. Нижче наведені принципи виміру основних критеріїв[1], за якими в подальшому аналізується серцевий ритм.

Частота серцевого ритму

Знайдемо середню відстань між сусідніми вершинами RR (за потреби можна взяти інший пік, що добре розпізнається). За умови запису точки ЕКГ в швидкості 50 мм/с, відстань між двома сусідніми точками графіку складатиме 0,002 с. Далі 60 секунд ділимо на отримане число і знаходимо частоту серцевого ритму за 1 хвилину.

Обчислюючи показник треба вважати ситуацію такою, де має місце серцева аномалія. Отже окрім підрахунку «правильної» частоти треба врахувати і ситуативну. Тому необхідно підрахувати кількість QRS інтервалів в середньому зустрічається за 6 секунд вимірів, а отриманий результат помножити на 10.

Регулярність серцевого ритму

В здорової людини RR інтервали однакові, проте допустиме відхилення до 10% в обидва боки.

Проводимість

Робота спеціальної провідної системи для швидкої передачі від джерела збудження до всього серця. Її неправильна робота описується трьома блокадами: синусовою (періодичне випадання інтервалів P – T), внутрішньо-передсердньою (подовження зубця P) та атріовентикулярною (подовження інтервала P–Q при випаданні Q – T).

Зубець P

Відповідає за скорочення лівого і правого передсердь. Заміряється тривалість зубця, його амплітуда. В ідеалі ці показники не мають перевищувати 0.12с та 0.2мВ відповідно.

Зубець Q

Показує початок поширення імпульсу по міжшлуночкової перегородки. Являє собою початкове негативне відхилення QRS, не має перевищувати чверті висоти R піку або продовжуватись більше 0.04с. Нормальний зубець Q не повинен бути зазубрений.

Зубець R

Відповідає роздачі імпульсу по правому і лівому міокарду шлуночків. Позитивне відхилення QRS, зумовлене збудженням міокарду правого та лівого шлуночків, має бути чітко вираженим. Нормальною вважається висота до 24мм.

Зубець S

Відображає закінчення розподілу збудження через міжшлуночкової перегородки. Являє собою друге негативне відхилення QRS, зумовлене збудженням базального відділу лівого шлуночків, не має перевищувати чверті висоти R піку або продовжуватись більше 0.03с. Є самим глибоким негативним зубцем на ЕКГ.

Зубець T

Показує процес реполяризації міокарда шлуночків (підйом шлункового комплексу в сегменті ST). Швидка кінцева реполяризація шлуночків від епікарда до ендокарду, не має перевищувати чверті висоти R піку, але може мати велику протяжність до 0.24с.

Інтервал P-Q

Показує тривалість проходження імпульсу через шлуночки, пучок Гіса і атріовентрикулярний вузол назад до шлуночків. Вимірюється від початку P до початку Q. Суть цього інтервалу: відображення затримки імпульсу. З віком інтервал PQ подовжується. У нормі інтервал продовжується 0,12с – 0,18с.

Інтервал Q – S

Вимірюється від початку Q до кінця **S**. Суть цього інтервалу: відображення деполяризації шлуночків. Тривалість може коливатись від 0.06 до 0.1с.

Інтервал S – T

Вимірюється від кінця **S**. до початку Т. Дуже важливий для виявленняаномалій.

Інтервал Q – T

Суть цього інтервалу: відображення збудження всіх шлуночків серця. Протяжність має складати не більше половини R – R відстані.

Індекс Маркауза

Співвідношення протяжності зубця Р до інтервалу Р–Q. У здорової людини становить 1.1 - 1.6.

1.2 Норми показників ЕКГ у людей в таблицях та критерії аналізу стану

Для ЕКГ результатів немає різниці стосовно статі.

Частота серцевих скорочень вважається здоровою якщо варіюється від 60 до 90 ударів в стані спокою.

Нормальні показники часових інтервалів пацієнта будуть мати приблизно наступні значення [2]:

Таблиця 1.1 – Порівняльний аналіз підходів до організації розмітки даних

Інтервал/пік	**Довжина**
PQ	0,12 – 0,2 с
QRS	0,06– 0,1 с
QT	<30% від RR 0,35 – 0,44 с
RR	0,6 – 0,66 с

Нормальні показники піків пацієнта будуть мати приблизно наступні значення, наведенні в таблиці 1.2:

Таблиця 1.2 – Порівняльний аналіз підходів до організації розмітки даних

Пік	**Довжина**	**Амплітуда**	**Додатково**
P	0,07–0,12 с	<2,5 мм	В деякихвипадкахможе бути від`ємним.
Q	<0,03 с	<4мм, < 25% R	
R	0,06 - 0,1 с	6 – 25 мм	
S	<0,06 с	2,5 –6 мм	Амплітуда зазвичай мала
T	0,16 – 0,24 с	2 – 6 мм	Завждидодатній

За допомогою електрокардіограми можна виявити ряд небезпечних захворювань серця.

Аритмія – патологічна хвороба, яка виражається порушенням частоти серцебиття, його ритму та послідовності. Виявляється за допомогою частоти та регулярності серцевого ритму.

Нормальні показники:

Частота серцевих скорочень від 60 до 90 ударів в стані спокою.

Патологічні показники:

Тахікардія: частота від 140 до 200 ударів за хвилину, вузькі Q – S комплекси.

Брадикардія: частота менше 60 ударів за хвилину, збереження правильного ритму, зубець Рчасто нашаровується на T.Q – S комплекс подовжується до 0,12 – 0,16 мс.

Фібриляція передсердь: частота від 90 до 140 ударів за хвилину, відсутність зубців Р, неоднакові та непередбачувані R – R інтервали при однакових Q – S.

Інфаркт міокарду – патологічна хвороба, при якій внаслідок нестачі кровообігу в серцевому м`язі виникають запальні процеси, що призводять до некрозу тканин. Фактично це є смертю клітин серця і може спричинити інвалідність або навіть смерть.

Патологічні показники:

Значні зміни інтервалу S – T та T–зубця в 2 або більше інтервалах підряд, які показують на появу вогнища некрозу тканин. Характерним показником може стати підйом сегмента S – T більше ніж на 2мм.

Поява патологій зубця Q в 2 або більше інтервалах підряд. Патологією вважається тривалість більше за норму на 0.03с.

Метаболічна кардіоміопатія – ураження серця без запальних процесів, причиною якого є порушення обміну речовин в організмі.

Патологічні показники:

Зміна інтервалу S – T, що має вигляд сходження до зубця T. Зубець T може бути сильно деформованим, від`ємним, низьким.

Збільшення часу інтервалу Q – T при загальному збільшенні амплітуди Q – S.

Гіпертрофія шлуночка – основний чинник смертності внаслідок хвороб серця.

Патологічні показники:

Лівого шлуночка: збільшені амплітуди S та R піків, від`ємний або від`ємно-додатній T зубець, інтервал S – T має низхідну депресію.

Правого шлуночка: збільшена амплітуди R піка, від`ємний T зубець, інтервал S – T має низхідну депресію.

Стенокардія – синдром, що спричинює біль та дискомфорт в області серця, лівої руки та грудини. Зумовлена нестачею кисню в міокарді.

Патологічні показники:

Депресія інтервалу S – T найчастіша ознака. Іноді разом з нею спостерігається підйом S – T відносно інших інтервалів.

Зменшення висоти R піку.

Перикардит – синдром, що викликає захворювання перикарда та тампонаду серця, накопичує рідину в його порожнині. Причиною є запалення.

Патологічні показники:

Елевація сегменту S – T. Також інколи спостерігається інверсія зубця T.

Депресія сегменту P – Q.

Зубці R та Q не змінюються в порівнянні з ЕКГ здорової людини.

Міокардит – запалення, що вражає кардіоміоцити, судини та тканини перикарду.

Патологічні показники:

Завжди гострі видимі зміни, спостерігаються не в окремих P – Tінтервалах, а ледь не в кожному. Найчастіше змінам підлягає інтервал S – T та пік T.

Зубці Q зазвичай лишаються незмінними в порівнянні з ЕКГ здорової людини.

1.3 Висновки до розділу

В даному розділі було описано форму, в якій представлені дані електрокардіограми. Було визначено основні опорні точки, інтервали та характеристики, на основі яких можна побудувати подальше дослідження ЕКГ на різні хвороби серця, за допомогою розбиття графіку на частини. Детальний опис допустимих меж характеристик діаграми дає можливість побудувати умови для вирішення питання «чи є хвороба?»

Також було описано захворювання, які можливо розпізнати за допомогою зняття електрокардіограми. Було визначено їх характерні кількісні ознаки, що можуть бути опорним апаратом для машинного алгоритму при визначенні хвороби пацієнта. В наведеній нижче таблиці хвороби серця згруповані за симптоматикою, яку видно на ЕКГ.

Таблиця 1.3 – Патологічні параметри ЕКГ

Параметр	Характеристика	Вірогідні захворювання
R - R	Нерівномірна відстань	миготлива аритмія серцева блокада

Продовження таблиці 1.3

R пік	>25 мм >0,1 с Подвійнийпік Відсутність Пилкообразність	потовщення міокарда передсердьмерехтіння передсердь
P – Q інтервал	>0.2 с <0.03с	атріовентрикулярнаблокада серця
Q–S інтервал	Вид флага Відсутність горизонтальної лінії	гіпертрофія міокарда шлуночків блокада ніжок пучка Гіса фібриляція шлуночків інфаркт міокарда
Q пік	>4 мм >0,03 с	інфарктміокарда
S пік	> 10 мм	Гіпертрофія лівого шлуночка.
S–T інтервал	>6 мм	стенокардія інфарктміокарда ішемічнахвороба
T пік	>50% R Гострий пік Подвійнийпік	перевантаженнясерця ішемічна хвороба інфарктміокарда

2 ВІДОМІ МЕТОДИ КЛАСИФІКАЦІЇ ТА АНАЛІЗУ ДАНИХ

Дослідження ЕКГ даних зазвичай ділять на 2 основних етапи:

– поверхневий аналіз графіку, виділення опорних точок, інтервалів та характеристик;

– глибокий аналіз на основі отриманих даних, порівняння з типовими даними з бази, формулювання висновків.

Останні 10 років дослідники так чи інакше намагались винайти алгоритм, який міг би з задовільною точністю визначити хворобу людини за ЕКГ діаграмою. Більшість з них ставила собі на меті задовільнити потреби в точному діагностуванні хвороб серця без втручання людини. В цьому розділі описані найвідоміші з цих методів.

2.1 Методи виділення опорних точок

Перший етапмаєвирішальну роль в точності такого аналізу. Для успішного знаходження всіх опорних точок на початку треба виділити R – Rінтервали. БезпомилковемашиннедетектуванняR– зубця є нетривіальноюзавдачею.

По-перше, на ЕКГ присутніспотворення, викликані шумами різного характеру:

– міографічнаінтерференція– високочастотніперешкоди, пов'язані з активністюм'язівтілапацієнта;

– низькочастотний дрейф ізолінії через поганий контакт міжелектродомкардіопідсилювача і шкіроюлюдиниабонедостатньоюобробкоюїїповерхні;

– наводкамережіелектроживлення.

По–друге, форма R–піків, їх частотні, амплітудні і часові характеристики мають великий ступінь варіативності, викликаний фізіологічними особливостями пацієнта або патологією серцево-судинної системи.

Метод локальних мінімумів та максимумів [3]є тривіальною спробою вирішення проблеми. Оскільки нам точно відомо, додатніми чи від`ємними мають бути опорні точки, можна шукати Rпіки, а потім, послідовно рухаючись в обидва боки, локальні мінімуми та максимуми. Детальний алгоритм пошуку наведений на рисунках 2.1 – 2.2.

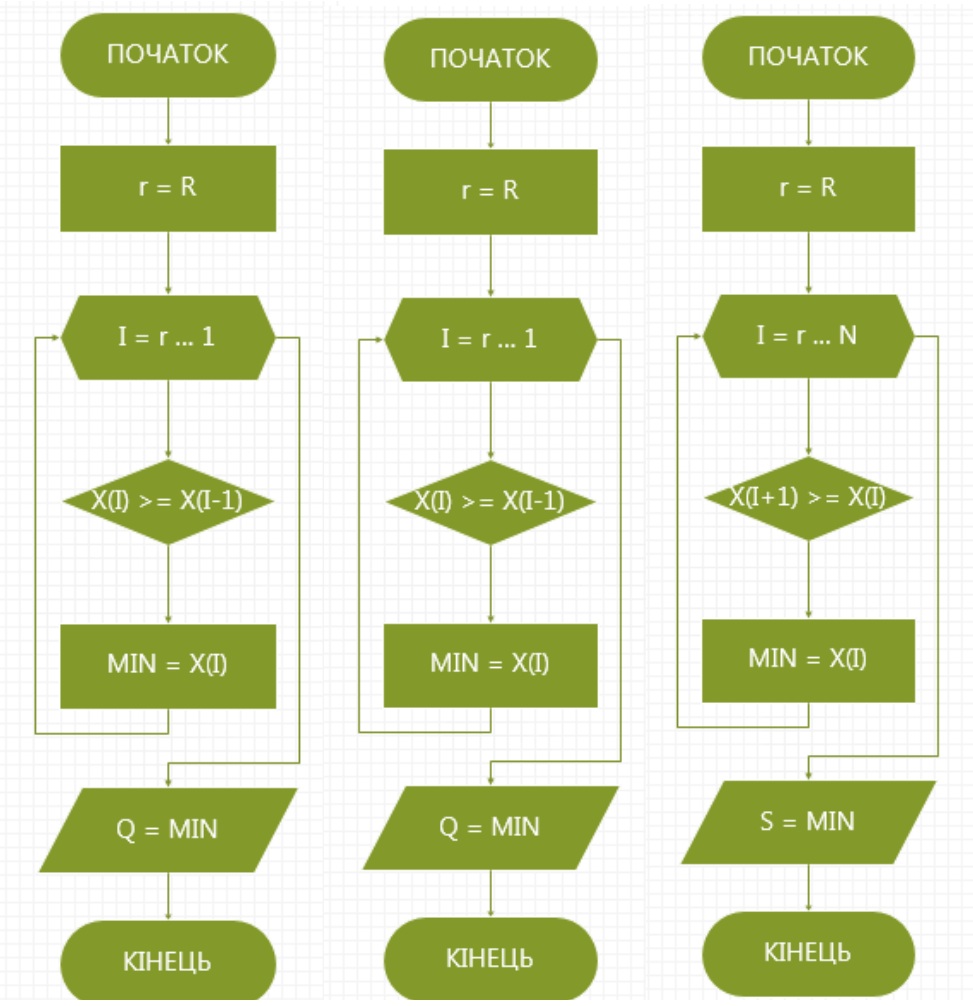


Рисунок 2.1 – Алгоритми пошуку опорнихточок Q, R, S

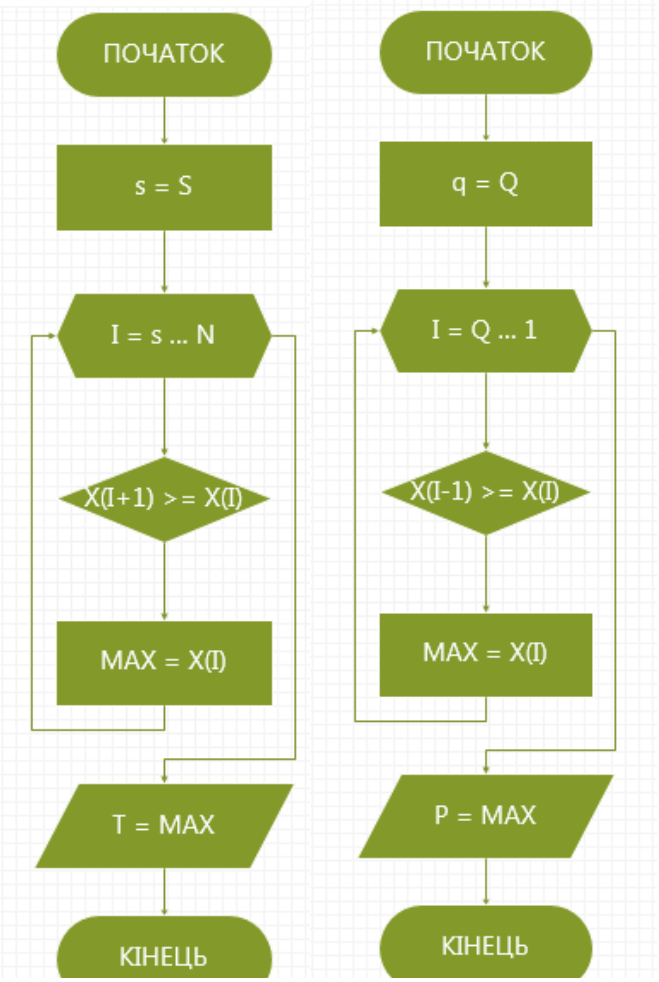


Рисунок 2.2 – Алгоритми пошуку опорнихточокP, T

Перевагиметоду:

– простота реалізації;

– не потребує великих ресурсів;

– добре виконує розпізнавання на ЕКГ здорової людини.

Недоліки методу:

– не здатний розпізнати аномальні ЕКГ хворого пацієнта;

– чутливий до зашумленості даних.

Алгоритм Пана−Томпкінса[4]базується на аналізі нахилу та ширині всередині Q – Sінтервалу.Він складається з наступної послідовності фільтрів і методів:

– фільтр нижніх частот;

– фільтр верхніх частот;

– знаходження похідної;

– зведення в квадрат;

– інтегрування;

– гранична процедура пошуку.

Фільтр нижніх частот описується формулою 2.1:

$$y(n) = 2*y(n-1) - y(n-2) + 1/32*[x(n) - 2*x(n-6) + x(n-12)] \quad (2.1)$$

Фільтр верхніх частот описується формулою 2.2:

$$y(n) = 2*y(n-1) - y(n-2) + 1/32*[x(n) - 2*x(n-6) + x(n-12)] \quad (2.2)$$

Похідна функція описується формулою 2.3:

$$y(n) = 1/8 * [2 * x(n) + x(n-1) - x(n-3) - 2 * x(n-4)] \quad (2.3)$$

Подальше зведення в квадрат робить результат позитивним і підсилює компоненти − комплексів.В результаті отримуємо новий ЕКГ, де підсилені високочастотні піки Q, R та S, а також зглажені низькочастотні зубці P та T.

Функція інтегрування (фільтр типу сковзуючого вікна) описується формулою:

$$y(n) = 1/N * [x(n-(N-1)) + x(n-(N-2)) + \cdots + x(n)] \quad (2.3)$$

де N – ширина вікна.

На рисунку 2.3 показано результат роботи методу:

Рисунок 2.3 – Метод Пана–Томпкінса

Переваги методу:

- простота реалізації;
- добре виконує розпізнавання на ЕКГ опорних точок Q, R та S.

Недоліки методу:

- посилені несиметричні Т вершини можуть бути зафіксовані як R піки.
- для визначення піків в вихідному сигналі потрібно враховувати затримку, створювану фільтрами.

Алгоритм підрахунку зміни знака [5] −ефективне рішення проблеми виявлення QRS-комплексів через простоту виявлення і підрахунку нульових перетинів.

На першому етапі сигнал фільтрується смуговим фільтром з нижньої і верхньої частотами зрізу − 18Гц і 35Гц відповідно.

Далі проводиться операція за формулою 2.4:

$$y(n) = sign(f(n))\ f(n^{**}2) \quad (2.4)$$

Підраховується оцінка амплітуди за формулою 2.5:

$$y(n) = \lambda * K(n-1) + (1 - \lambda * K) * |y(n)| * c (2.5)$$

де $\lambda K \in (0; 1)$ - фактор упущення, і параметр с позначає постійнепосилення, наприклад, с = 4.

Перетворюється сигнал на більш високочастотний за формулою 2.6:

$$z(n) = y(n) + b(n) (2.6)$$

Для підрахунку кількості перетинів застосовуєтьсяформула 2.7:

$$d(n) = [sign[z(n)] - sign[z(n-1)] / 2] (2.7)$$

Виявленняподійвиконується з використанням адаптивного порога Θрозраховується за формулою 2.8:

$$\Theta(n) = \lambda * \Theta(n-1) + (1 - \lambda) * d(n), \text{ де } \lambda\, \Theta \in (0; 1) (2.8)$$

На рисунку 2.4 показано результат роботи методу:

Рисунок 2.4 – Метод підрахунку зміни знака

Переваги методу:

- простота реалізації;
- добре виконує розпізнавання на ЕКГ опорних точок Q, R та S.

Недоліки методу:

– для визначення позиції піку в вихідному сигналі необхідно брати до уваги затримку смугового фільтра.

Алгоритм неперервного Вейвлет-перетворення[6] – нова техніка велектрокардіографії, що дає вдосконалені методи. Перевага цього перетворення – в здатності відзначати деталі ЕКГ–сигналу за оптимальний час.

Вейвлет-перетворення засноване на наборі функцій-аналізаторів, що розкладають ЕКГ-сигнал на послідовності коефіцієнтів. Такі функції–аналізатори (Вейвлети) - це узагальнена назва сімейств математичних функцій певної форми, які локальні за часом і по частоті, і в яких всі функції виходять з однієї базової (породжуючої) функції за допомогою її зрушень і розтягувань по осі часу.Кожен вейвлет має певну тривалість, положення в часі і смугу частот. В результаті перетворення вейвлет–коефіцієнти відповідають ЕКГ-компонентів на якомусь часовому відрізку і смузі частот.

Діляться на 2 великі групи:

– дискретні – використовується для обробки та стиснення сигналів, їх кодування;

– неперервні – використовуються для аналізу сигнала в наукових дослідженнях.

Вейвлет Морле (рисунок 2.5):

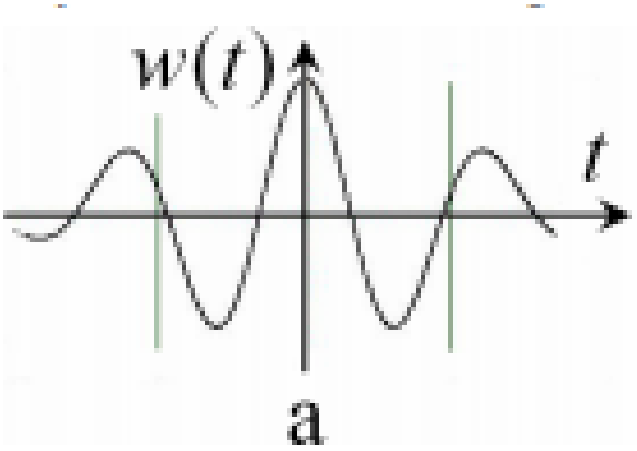


Рисунок 2.5 – Вейвлет Морле з масштабуванням 0.2

Вейвлет Haar(рисунок 2.6):

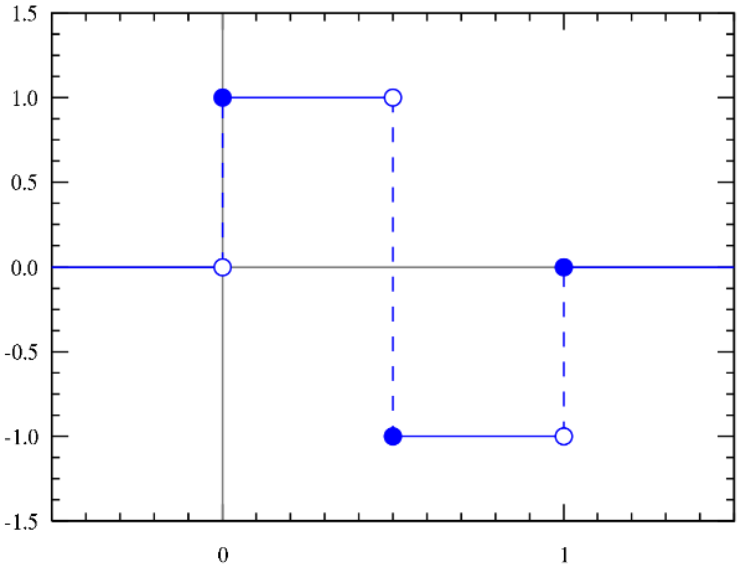


Рисунок 2.6 – Вейвлет Хаара

Вейвлет Гауса (рисунок 2.7):

Рисунок 2.7 – Вейвлети Гаусса різних порядків

Вейвлет «Мексиканська шляпка» (рисунок 2.8):

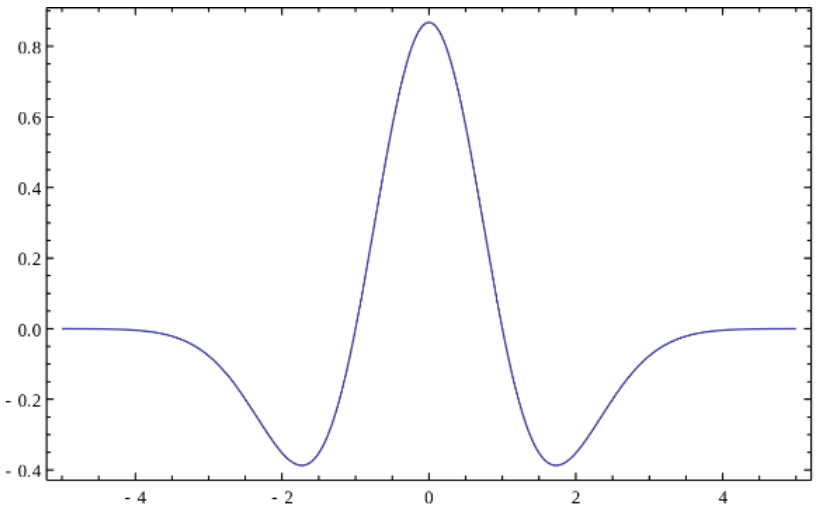


Рисунок 2.8 – Вейвлет MEXICANHAT

Переваги методу:

– сигнал добре ранжується як по часу, так і по частоті, можна виділити з нього лише ті рівні деталізації, що потрібно;

– варіативність функцій забезпечує широку сферу використання методу.

Недоліки методу:

– складність алгоритму.

2.2 Різноманітні підходи до глибинного дослідження числового ряду

Реальневиявленняаномалії для часовихрядів єяк і ранішескладнимзавданням. Це особливо актуально для періодичнихабоквазіперіодичнмх часовихрядів. Часовий ряд ЕКГ–яскравий приклад квазіперіодичнихсигналів у реальному світі. Впродовж багатьох років люди намагаються знайти найкращий алгоритм для визначення хвороби ЕКГ за допомогою машини, без втручання людини.

2.2.1 Прогнозування часових рядів

В 2019 році винайшли спосіб класифікування сигналу ЕКГ за допомогою нейронних мереж [7]. В статті був представлений алгоритм, що базувався на LongShort-TermMemory (LSTM) нейронних мережах.

Ідея була в тому, щоб навчити модель передбачувати наступні інтервали часового ряду. Інтуїція такого підходу полягає в тому, що звичайні квазіперіодичні закономірності в часових рядах ЕКГ повинні бути передбачуваними (допустимі лише незначні помилки), тоді як ненормальна поведінка повинна призвести до великих відхиленнях в прогнозах.

Для вивчення завдання прогнозування використовується LSTM архітектура з 2 шарами. Кожен шар складається з 64 юнітів. Модель LSTM навчається за допомогою визначенню Adamoptimizer (алгоритму середньоквадратичної помилки MSE). Інші функції, такі як log-cosh (логарифм гіперболічного косинуса) та MAE (середнє абсолютне значення помилки) також були взяті до уваги.

Для оцінювання аномальності часового ряду було використано Mahalanobisdistance.Великі помилки в одному або декількох вимірах призведуть до великих значеннь на відстані Махаланобіса. Тому відстань Махаланобіса можна використовувати як показник аномальності поведінки сигналу часового ряду.

На рисунку 2.9 наочно показано відстань махаланобісу в часі для вибраного сигналу ЕКГ до та після Window-BasedErrorCorrection алгоритму.

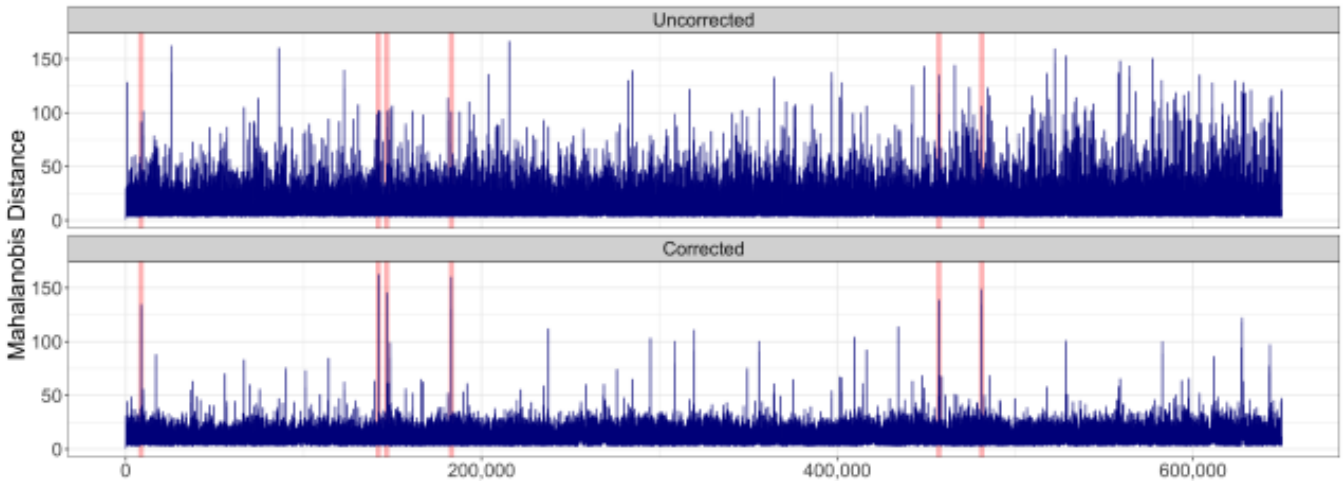


Рисунок 2.9 – Відстань махланобіса

При дослідженні аномалій перевіряються дві гіпотези:

– H0: У даних немає аномалій.

– H1: У даних є до ν відсотків аномалій.

Покроковий алгоритм аналізу наведений нижче.

1) Будемо вважати, що кожне k ∈{1, ..., m} є приблизно нормальним розподілом.

2) Ітеративно обчислюємо багатовимірний нормальний розподіл із середнім значенням μ та матрицею коваріації Σ.

3) Обчислюємо відмінності спостережуваного вектора порівняно з цим базовим багатовимірним нормальним розподілом, обчислюючи Mahalanobisdistance

4) Вибираємо найбільше значення відстань МахаланобісаMj і оцінюємо чи перевищує це значення 1 – α квантиль нормального розподілу.

5) Якщо так, спостереження вважається аномалією.

6) Процедура зупиняється, коли щонайбільше ν% даних були позначені як аномалія

Результати:

Подібно до звичайних класифікаційних завдань, похибка результатів вимірюється за допомогою основних 4 показників: кількість істинних позитивних відповідей (TP), хибнопо–зитивних (FP), хибно–негативних та істинно негативних (TN). В результаті отримуємо істинно-позитивний показник TPR, істинно-негативний показникFPR.

Вдалось досягти точності алгоритму 83%.

2.2.2 Класифікатор за піками та їх протяжністю

У статті [8] запропонований алгоритм ідентифікації ЕКГ, заснований на виявленні та тимчасовій локалізації максимумів модулів вейвлет-перетворення та класифікації кардіоциклів за допомогою нейромережі.

Задача класифікації кардіоциклів представляє себою завдання співставлення кардіоциклів з одним з чотирьох класів:

- без патології;
- блокада левоїножки пучка Гіса;
- блокада правої ножки пучка Гіса;
- шлуночкова екстрасистола.

На входи нейронів першого слою поступає 24 елементів-ознак (дивись таблицю 2.1):

Таблиця 2.1 – Ознаки класифікації кардіоциклів

Номер	**Ознака**
1	Тривалість QRS–комплексу (QRS)
2	Тривалість інтервалу PQ (PQ)

3	Тривалість інтервалу QT (QT)

Продовження таблиці 2.1

4	Тривалість попереднього RR-інтервалу
5	Тривалість наступного RR–інтервалу
6 - 15	Нормалізовані морфологічні ознаки сигналу ЕКГ між початком і кінцем QRS (піки та початок – кінець)
16 - 24	Нормалізовані морфологічні ознаки сигналу ЕКГ між кінцем QRSкомплекса і кінців Т–хвилі (піки та початок – кінець)

Виходи нейронів першого шару надходять на входи нейронів другого шару, а виходи нейронів другого шару формують вектор виходів мережі (дивись рисунок 2.10).

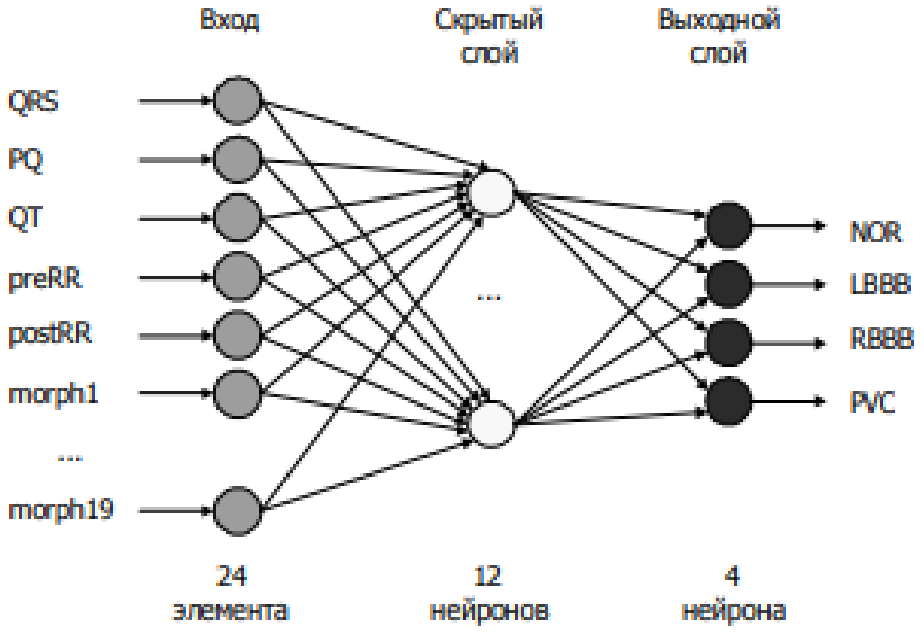


Рисунок 2.10 – Схема нейронної мережі

Дослідженню піддавалися записи цифрових сигналів ЕКГ постійно оновлюваного банку даних комплексних фізіологічних сигналів PhysioBank. База даних QTDB містить 105 записів ЕКГ в двох відведеннях по 15 хвилин і частотою

дискретизації 250 Гц. Кожен запис має як мінімум 30 анотованих ударів кмітливістю початку, піку і кінця P–хвилі і QRS–комплексу і мітками піку і кінця T–хвилі.

Результати:

Для оцінки якості роботи використовуються два показники: чутливість (здатність алгоритму давати правильний результат) і прогнозованість позитивного результату (позитивна прогнозованість).

Вдалось досягти того, що частка істинно позитивних випадків серед всіх позитивних випадків 99.7%.

У статті [9] провели схоже дослідження.

При формуванні вихідних даних був використаний структурований масив оцифрованих записів реальних фізіологічних сигналів і пов'язаних з ними даних для застосування біомедичних спільнотою в дослідженнях. Дані записи були отримані холтерівськогомоніторування в умовах стаціонару.

Для навчання штучної нейронної мережі використовувався алгоритм зворотного поширення помилки. Був підготовлений набір даних з 458 епізодів ЕКГ (Навчальна вибірка 50%, валідаційну вибірка 30%, і тестова вибірка 20%). Підготовлені вхідні дані подаються на вхідний прошарок нейронної мережі, вихідний шар якої діагностує стан пацієнта.

Вхідні параметри для мережі можна побачити на рисунку 2.11:

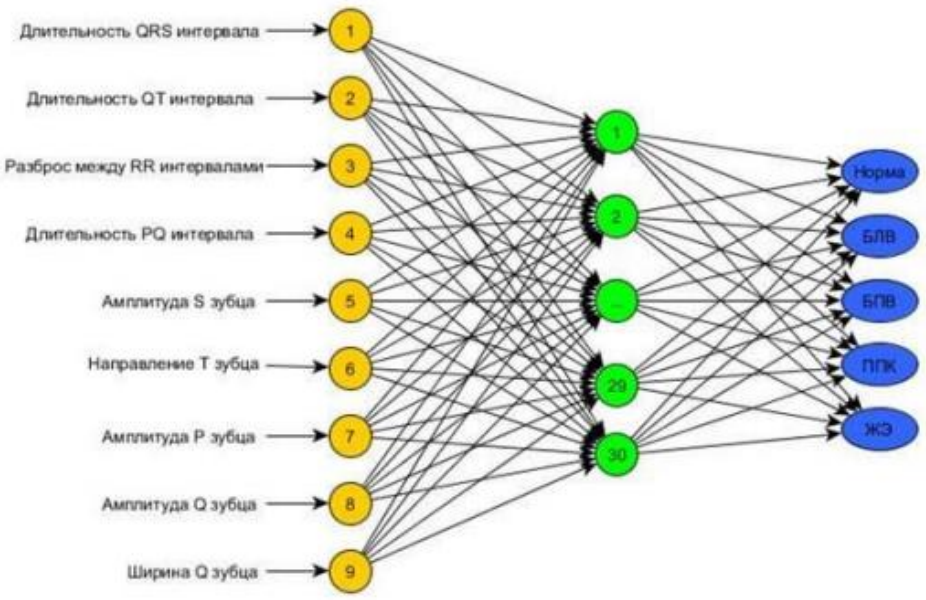


Рисунок 2.11 – Схема нейронної мережі

Для знаходження оптимального числа нейронів прихованого шару нейронної мережі зі структурою багатошарового персептрона, в роботі була проведена оцінка критеріїв ефективності роботи штучної нейронної мережі – чутливості, специфічності і помилки навчання.

Показники чутливості і специфічності в ідеальному випадку повинні прагнути до 100%. У реальних умовах при вирішенні задач діагностики, система повинна вибрати один з декількох варіантів діагнозу. При цьому по всіх варіантах діагнозу бажано мати значення критеріїв чутливості і специфічності системи по всіх варіантах діагнозу рівномірно розподіленими, але не нижче порогового значення, при якому результат не може вважатися достовірним.

Результати:

Специфічність запропонованої системи, перевірена дослідженням на контрольній групі без серцево–судинних захворювань, склала 81%.

Чутливість розробленої системи перевірялася на контрольній групі з серцево-судинними захворюваннями, склала 79%.

2.2.3 Класифікатор за допомогою методу кластеризації

У статті [10] для вдосконалення використовували медіанний фільтр та фільтр низьких частот продуктивність алгоритму класифікації стресів. Після застосування середнього фільтра, а для усунення шуму використовувався фільтр низьких частот.

Точки характеристик відповідно до різниці між інтервалом R–R та значенням R та S піків витягується з отриманого сигналу ЕКГ. Для класифікації сигналу ЕКГ пікове значення R-S встановлюється на 1,2 як орієнтир. Якщо воно більше цього значення, ЕКГ сигнал рухається вправо, а якщо він менше, він рухається вліво.

K-means кластеризація застосовується та класифікується на випадки стресу та відсутність стресу.Результатакої кластеризації можна побачити на рисунку 2.12:

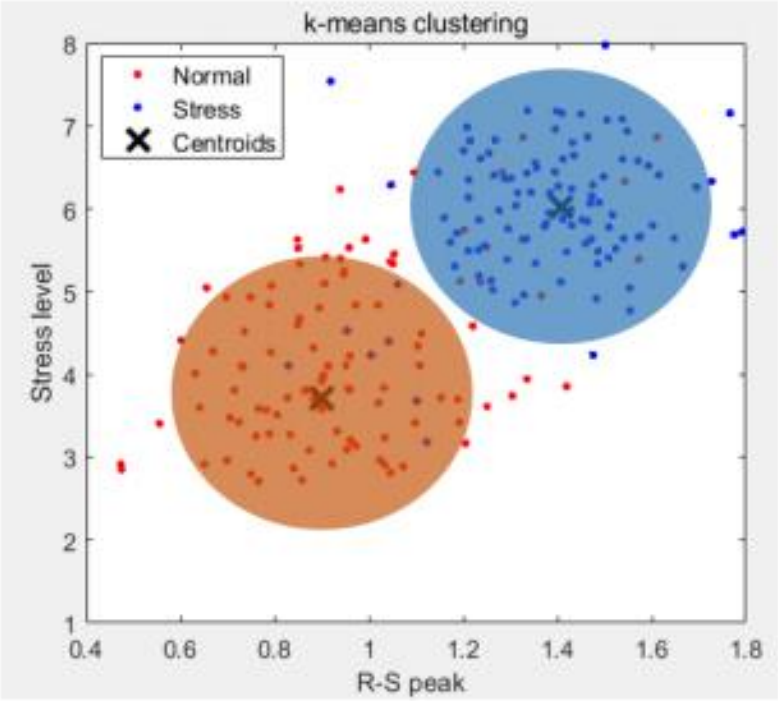


Рисунок 2.12 – Результат алгоритму K-means

Результати:

Точність алгоритму класифікації напружень запропоноване в цьому дослідженні було розраховано порівнянням усіх ЕКГ даних та відстані до всіх центроїдів. Ця точність була 85,07%.

У статті [9] автори пропонують підхід до вивчення ЕКГ без нагляду (Unsupervisedrepresentationlearningusingdeepgenerativemodels).

Запропонований підхід складається з двох завдань:

– навчити без нагляду алгоритм на основі ЕКГ даних;

– подальше завдання виявлення аномалій.

Модель, яка використовується для навчання, базується на варіаційному автокодері, параметризованом періодичними нейронними мережами.

Вхідними даними моделі є часовий ряд.

Філософія, що лежить в основі виявлення аномалій на основі автокодера, полягає у навчанні моделі на часових рядах з переважно нормальними шаблонами, так що він засвоює безліч нормальних значень даних.

Мета полягає у визначенні двох кластерів, що містять приховані коди нормального та аномального серцебиття ЕКГ. Ця стратегія передбачає, що більшість серцевих скорочень мають нормальний характер, і, отже, аномальні серцебиття будуть проектуватись по-різному в прихованому просторі по відношенню до нормальних.

Було розглянуто три алгоритми кластеризації:

– ієрархічну кластеризацію;

– спектральну кластеризацію;

– K-means.

Кластери підбираються до цих класів відповідно до їх розміру: один з більшою кількістю точок даних присвоюється нормальному класу, а інший -аномальний клас.

Результати:

AUC обчислюється для кривої ROC з відповідними істинними додатними іхибнопозитивними показниками.

Вдалось досягти точності алгоритму 92%.

2.2.4 Аналіз синусоїдального ритму за допомогою SVM

В статті [11] розглядаютьсяалгоритмивиділення і класифікаціїінформативнихознакелектрокардіосигнала.

Головною ідеєю дискретного вейвлет-перетворення (ДВП) є розбивка сигналу на двіскладові - грубу (аппроксимуючу) і уточнену (деталізуючу), з подальшимїхдробленням з метою змінирівнядекомпозиції сигналу. При цьому за нульовийрівеньдекомпозиціїприймається сам сигнал, а наступнірівніутворюютьнисхідне вейвлет-дерево того чиіншого виду. Точністьподання сигналу під час переходу на більшнизькірівнізнижується, але затез'являєтьсяможливість вейвлет-фільтраціїсигналів, видаленняшумів і ефективноїкомпресіїсигналів.

Розглянуто задачу класифікації для об'єктівдвохкласів.Нехайзадані:

– безлічнавчальнихоб'єктів, представлених векторами ознак;

– безлічвідповідей для навчальнихоб'єктів.

Рішення завдання класифікації зводиться до побудови такої функції (класифікатора), яка оптимальним чином розділяє точки з навчальної вибірки різних класів.

У методі SVM як функція обрана гіперплощина, відстані до якої найближчих точок-векторів, які носять назву опорних, обох класів рівні (дивись рисунок 2.13).

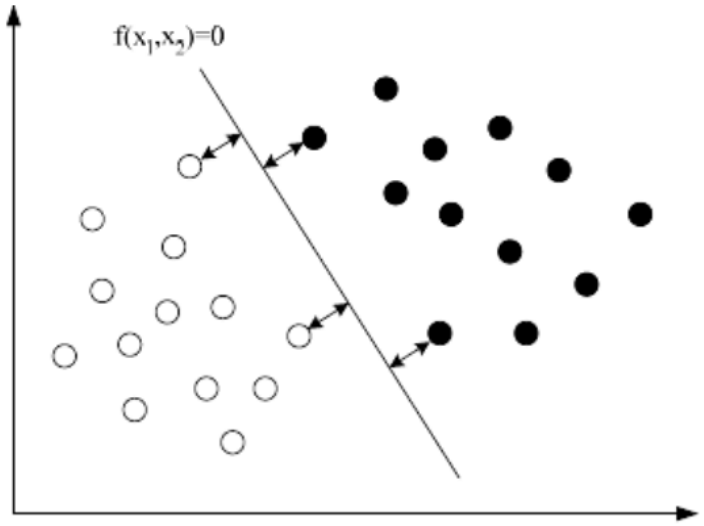


Рисунок 2.13 – SVM для двовимірного простору

На першому етапі з множини навчальних прикладів відбираються опорні вектори, на основі яких будується розділяючаплощина. Етап розпізнавання полягає в тому, що на вхід отриманого класифікатора подається приклад, про класову приналежність якого нічого не відомо. Класифікатор повинен видативідповідь, до якогокласуналежить вектор.

У даній роботі в якості аналізованих даних були обрані електрокардіосигналіинормального синусового ритму і електрокардіосигнали з аритмією.

Вхідними даними для вирішення задачі класифікації в даній роботі послужили частотні і тимчасові параметри варіабельності серцевого ритму, набори середнього квадратичного відхилення,деталізуючі і апроксимуючі коефіцієнти, отримані в результаті дискретного вейвлет розкладу (дивись таблицю 2.2).

Таблиця 2.2 – Ознаки класифікації серцевого ритму

Номер	**Ознака**
1	Пульс
2	Відхилення пульсу
3	Коефіцієнт варіації
4	Потужність спектру TP
5	Потужність спектру LF
6	Потужність спектру HF
7	Потужність спектру VLF
5	Стрес-індекс

Результати:

Післязакінчення етапу навчання на вхід був поданий електрокардіосигнал, про класовуприналежність якого нічого не було відомо. Результатом класифікації сталовіднесення даного сигналу до класу електрокардіосигналів зі здоровим ритмом, що є доказом правильності роботи системи.

Вдалось досягти точності алгоритму 84%.

2.3 Висновки до розділу

В даномурозділі були наведені основні методи, які зараз використовуються в дослідженні ЕКГ.

На першому етапі (виділення параметрів ЕКГ, опорних точок) найчастіше використовуються два алгоритми:

– локальних мінімумів та максимумів;

– вейвлет-перетворення.

Вони значно полегшують подальшу роботу з сигналом, звільняючи його від шуму та виявляючи певні ознаки, за якими найвигідніше порівнювати два інтервали між собою.

На другому етапі існує безліч варіантів досліджень. В таблиці 2.3 наведено порівняльну характеристику найпопулярніших методів. Кожен з них має певні ознаки, що вигідно виділяють його проміж інших. Проте жоден з них не є ідеальним варіантом, саме тому дослідження в цій області науки ідуть повним ходом. Зеленим кольором позначені риси, що суттєво вплинули на позитивний результат, а червоним ті, шо спричинили негативний.

Таблиця 2.3 – Ознаки класифікації серцевого ритму

	Прогнозування часових рядів	**Класифікаторза піками**	**Кластеризація**	**Аналіз на основі SVM**
Вхідні дані	Бази даних	Бази даних, вже з параметрами	Бази даних	Бази даних

Опорні точки	Попередні повні інтервали	Локальні мінімуми, максимуми та час	**Локальні мінімуми, максимуми та інші опорні значення**	**Частотні параметри**
Попередня обробка ряду	Ні	Ні	**Так**	**Так**
Головний алгоритм	LSTM, Melanobis distance	Нейроннамережа	K-means	SVM
Навчання	Без учителя	З учителем	Без учителя	З учителем
Складність програми	Складний	Складний	Простий	Простий
Класифікація	Однокласова	Багатокласова	Багатокласова	Однокласова
Опорний клас	Здорова ЕКГ	Кожна хвороба одночасно	Кожна хвороба одночасно	Здорова ЕКГ

Продовження таблиці 2.3

Конкретний діагноз	Ні	Так	Так	Ні
Точність	83%	80%	92%	84%

Проаналізувавши таблицю, можна виділити успішні риси кожного з алгоритмів та звести їх до єдиного списку:

– попередня обробка даних за допомогою вейвлетів та шумових фільтрів;

– орієнтація на виявлення хвороби, а не здорової ЕКГ;

– простота програми для можливості її подальшого використання для розширення;

– максимально детальні, відфільтровані та згруповані вхідні дані що мають різні значення для різних станів людини;

– багатокласова класифікація, як результат – конкретні діагнози;
– навчання з учителем.

3 РОЗРОБКА МЕТОДУ КЛАСИФІКАЦІЇ ТА АНАЛІЗУ ДАНИХ

Після аналізу існуючих алгоритмів розпізнавання електрокардіограм, було прийнято рішення розробити власний алгоритм, врахувавши всі переваги та недоліки існуючих методів.

3.1 Постановка завдання

Мета: головною метою магістерської дисертації є збільшення точності та варіативності аналізу ЕКГ даних на хвороби за рахунок використання інформативного набору вхідних даних та створення найбільш пристосованого до такого типу аналізу алгоритму їх обробки.

Призначення: створення програмного забезпечення, яке самостійно виконує виділення ключових характеристик ЕКГ інтервалів, аналізує їх на хворобливий стан та інформує користувача про ситуацію з пацієнтом.

Задачі:

– розробити алгоритм зчитування ЕКГ даних та їх розбиття на PQRSTінтервали;

– визначити ключові характеристики, за якими буде проводитись аналіз даних;

– за допомогою створеного алгоритму дослідити ЕКГ з кожною хворобою окремо та записати ключові характеристики в датасети;

– розробити алгоритм аналізу ЕКГ для конкретної хвороби із застосуванням машинного навчання;

– дослідити ефективність алгоритму;

– створити програмний інтерфейс для візуалізації результатів обробки ЕКГ пацієнта.

3.2 Метод модифікації вхідних даних

В якості навчальних та тестових даних було обрано датасети[12] у вигляді необроблених часових рядів.

Алгоритм фільтрації даних включає в себе два основних кроки:

– застосування вейвлет-перетворення;

– виявлення періоду сигналу та його розбиття.

3.2.1 Вейвлет Хаара

Перша проблема ЕКГ даних, які програма має обробити, це їх величина. В середньому за 20 хвилин зняття електрокардіограми в датасет поступає близько 300000 точок.

За допомогою вейвлет-перетворень стає можливим стиснути дані в декілька разів без втрати суттєвої інформації. **Вейвлет Хаара**(дивись рисунок 3.1) – один з перших і найбільш простих вейвлетів[17].

Він заснований на ортогональній системі функцій, запропонованої угорським математиком Альфредом Хара в 1909 році. Вейвлети Хаара ортогональні, добре локалізовані в просторі, але не є гладкими.

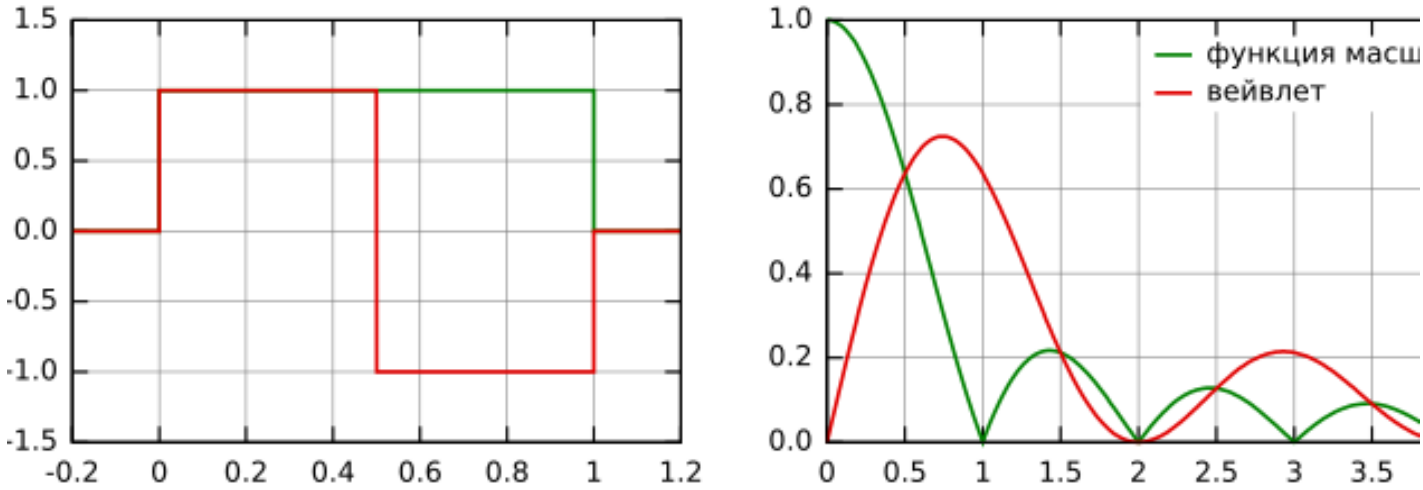


Рисунок 3.1 – Функція масштабування та її частотні складові

Функція задається формулою 3.1:

$$f(x) = \begin{cases} 1, & 0 < x < \frac{1}{2} \\ -1, & \frac{1}{2} < x \geq 1 \\ 0, & x > 1 \;||\; x < 0 \end{cases} \qquad (3.1)$$

Нехай є одновимірнийдискретнийвхідний сигнал S. Кожнійпарісусідніхелементівставляться у відповідність два числа (дивись формули 3.2 – 3.3).

$$a = \frac{S_{2i} + S_{2i+1}}{2} \qquad (3.2)$$

$$b = \frac{S_{2i} - S_{2i+1}}{2} \qquad (3.3)$$

Повторюючицюоперацію для кожного елементавихідного сигналу, на виходіотримують два сигнали, один з яких є огрубінняверсієювхідного сигналу, адругийміститьдеталізуєінформацію, необхідну для відновленнявихідного сигналу.

На кожному кроці перетворення сигнал розпадається на дві складові:

– наближення з більш низькою роздільною здатністю (апроксимацію);
– інформацію що дозволить деталізувати наближення.

Перетворення Хаара використовується для стиснення вхідних сигналів, компресії зображень, в основному кольорових і чорно-білих з плавними переходами. Ідеальний для картинок типу рентгенівських знімків. Даний вид архівації відомий досить давно і безпосередньо виходить з ідеї використання когерентності областей. Ступінь стиснення задається і варіюється в межах 5–100.

3.2.2 Алгоритм виявлення періоду

Після вейвлет-перетворення сигнал містить більш концентровані локальні мінімуми та максимуми. Алгоритм знаходження періодичності сигналу базується на побудові R-Rінтервалів (дивись рисунок 3.2).

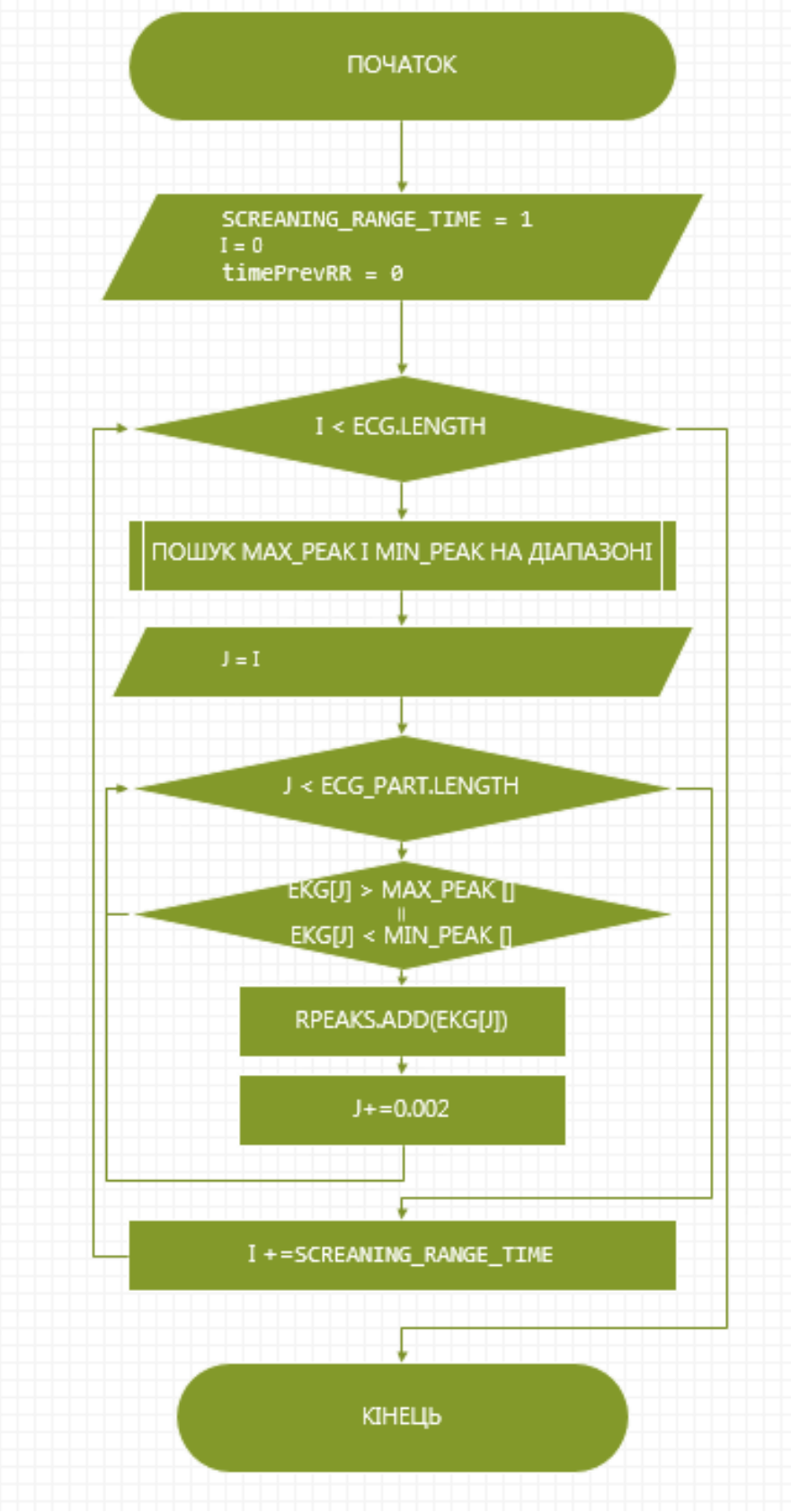


Рисунок 3.2 – Алгоритм поділу на R-Rінтервали

Загальну довжину ЕКГ ділимо на інтервали по 500 точок (зі сподіванням отримати по 3–4 R піки в кожному). Це дає змогу програмі уникнути помилок при визначенні локальних мінімумів на різних інтервалах при перепаді висот та зміні поведінки кривої. Також це дозволяє виявити сегменти для всіх п`яти фаз одним циклом.

Загалом фрагменти ЕКГ можна поділити на здорові (дивись рисунок 3.3) і хворі (дивись рисунок 3.4). Як видно, в деяких випадках наше загальне уявлення про PQRS комплекс може істотно відрізнятись від реального часового ряду.

Рисунок 3.3 – Приклад PQRSздорової людини

Рисунок 3.4 – Приклад викривлення PQRSхворої людини

Такі зміни можуть унеможливити визначення правильного R-R сегменту. Тому для даного випадку алгоритм тимчасво приймає за R-пік локальний мінімум на всьому інтервалі.

3.3 Методианалізу даних для проектування програмного забезпечення

В процесі знаходження опорних точок для формування датасетів з важливими ознаками було проаналізовано які саме характеристики частіше всього досліджуються лікарями при огляді ЕКГ пацієнтів в лікарнях.

3.3.1 Методаналізу числових даних

В таблиці 3.1 представлені основні опорні точки, які були знайдені методом визначення локальних мінімумів та максимумів (дивись розділ 2).

Таблиця 3.1 – Основні точки та причини їх появи в алгоритмі

Точка	**Причина**
P локальниймаксимум	Вказує наблокаду серця, перикардит, міокардит
Q локальний мінімум	Вказує на інфаркт
R локальниймаксимум	Вказує на хворобу передсердь
S локальний мінімум	Вказує на аритмію, ішемічну хворобу
T локальниймаксимум	Вказує на аритмію, стенокардію

Також на хворобливий стан може вказувати час перебігу тої чи іншої події в процесі серцебиття. В нижче зібрані основні часові характеристики кожного PQRSTінтервалу:

- час P відносно R по модулю;
- час Qвідносно R по модулю;
- час Sвідносно R по модулю;
- час Твідносно R по модулю;
- час PQ інтервал (від початку P горба до точки Q);

- час QSінтервал(від початку Q прогину до кінця S прогину);
- час STінтервал (від точки Sдо кінця T горба);
- швидкість серцебиття (в ударах на хвилину);
- середнє відхилення від швидкості серцебиття (аритмічність).

На рисунку 3.5 показано алгоритм розбиття кожної пари R-R інтервалів на числові показники:

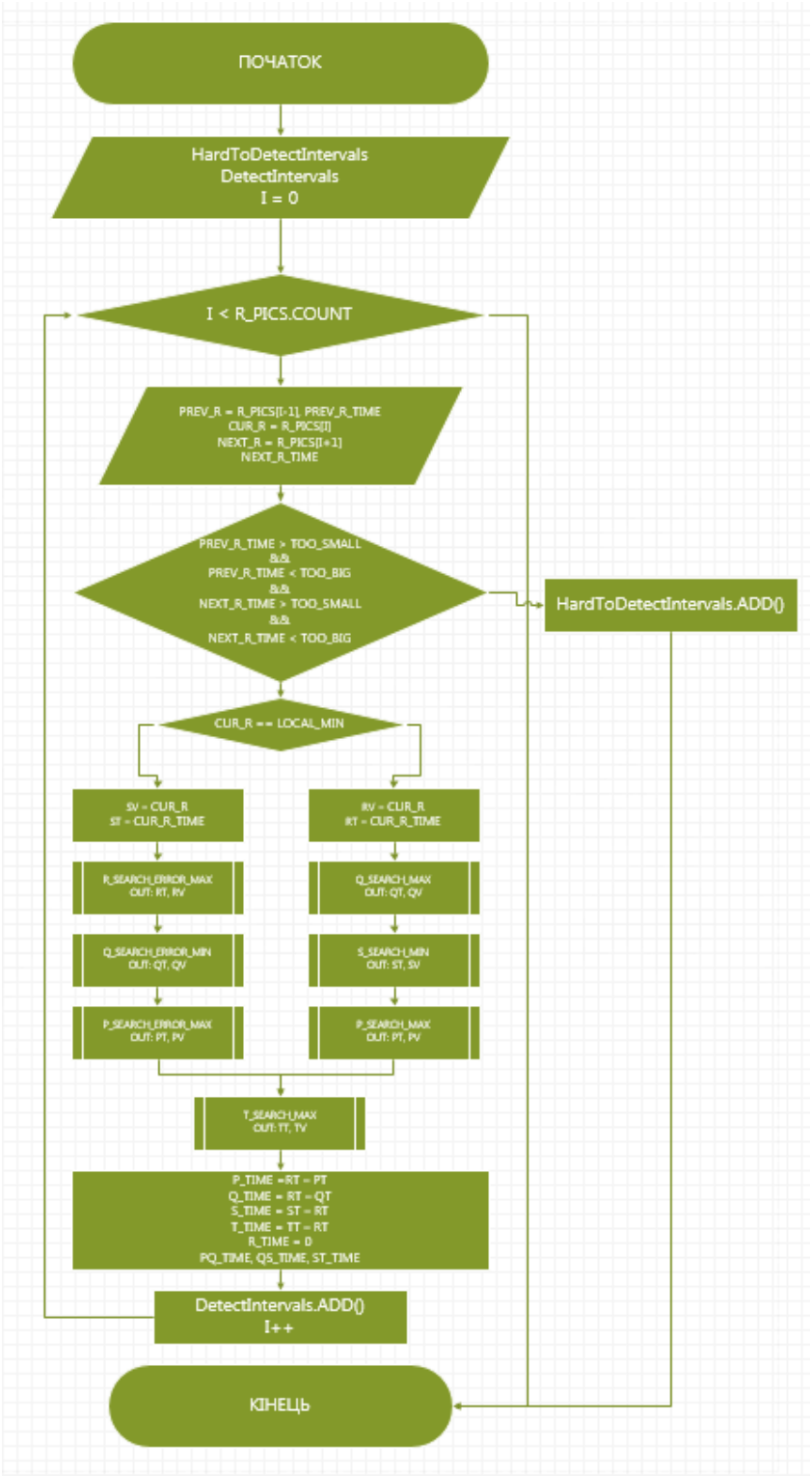


Рисунок 3.5 – Алгоритм розбиття ЕКГ на опорні точки інтервалів

3.3.1 Методаналізу інтервальних даних

Під час аналізу готових наборів PQRST інтервалів виникає проблема: два очевидно різних для людського ока інтервали програма може описати одними і тими самими характеристиками. Це виникає за рахунок великої розрідженості даних, що беруться до уваги (дивись рисунок 3.6).

Рисунок 3.6 –Ілюстрація низької інформативності пікових показників

Очевидно, що низька інформативність датасету призводить до визнання подібними абсолютно різних інтервалів. Часові показники дещо покращують ситуацію, проте як видно з інтервалу ST на рисунку, і вони не можуть забезпечити достатнього опису картини.

Рішення цієї проблеми може забезпечити розпізнавання перепадів значень ряду. Дослідивши поведінку кривої в розрізі одного PQRSTінтервалу, було помітно що частіше за все перепади значень траплялись в межах від 0 до 200 одиниць. Було прийнято рішення позначити перепади різного рівня (різницю між попереднім значенням інтервалу та поточним) літерами англійського алфавіту (дивись таблицю 3.2).

Таблиця 3.2 – Алфавітні позначки інтервальних перепадів

Перепади в більшу сторону	
A	0
B	1-2

C	3-5

Продовження таблиці3.2

D	6-9
E	10-15
F	16-25
G	26-35
H	36-45
I	46-55
J	56-85
K	86-109
L	110-149
M	150-199
N	>200
Перепади в меншу сторону	
O	>200
P	150-199
Q	110-149
R	86-109
S	56-85
T	36-55
U	26-35
V	16-25
W	10-15
X	6-9
Y	3-5
Z	1-2

ЕКГ інтервали, досліджені та розбиті на алфавітні позначки, виглядають наступним чином:

– PQ: “UBFBEZVJTZCX”;

– QS:“CZYWUYGFXWW”;

– ST: “YBEDVECYUJOJIGXFZ”.

На часовому рядку ЕКГ, оптимізованому за допомогою вейвлету Хаара, для опису пагорба чи прогалини на графіку розглянемо декілька типових сценаріїв обробки (дивись рисунок 3.7).

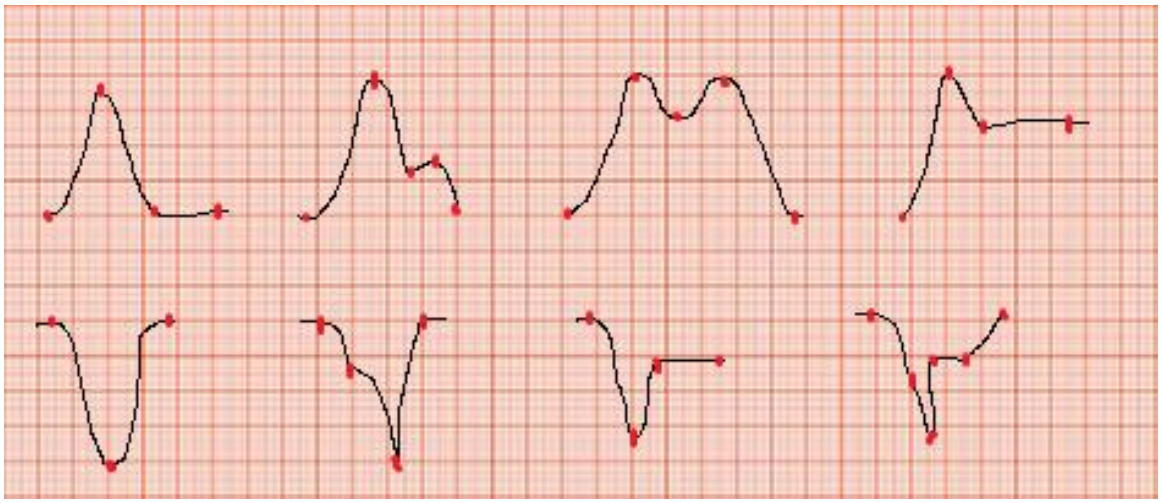


Рисунок 3.7 –Ілюстрація низької інформативності пікових показників

Очевидно, що для опису поведінки кривої на одній ділянці достатньо інформації від 4 до 6 точок.

Тому всі ряди з алфавітних позначок перетворимо на речення із слів по 3, 4 та 5 символів:

– PQ: “UBF UBFB UBFBE BFB BFBE BFBEZ FBE FBEZ FBEZV BEZ BEZV BEZVJ EZV EZVJ EZVJT ZVJ ZVJT ZVJTZ VJT VJTZ VJTZC JTZ JTZC JTZCX TZC TZCX ZCX UBFBEZVJTZCX”

– QS: “CZYCZYWCZYWUZYWZYWUZYWUYYWUYWUYYWUYGWUYWUYGWUYGFU

YGUYGFUYGFXYGFYGFXYGFXWGFXGFXWGFXWWFXWFXWWXWWCZYWUY
GFXWW”

– ST:
“YBEYBEDYBEDVBEDBEDVBEDVEEDVEDVEEDVECDVEDVECDVECYVECVEC
YVECYUECYECYUECYUJCYUCYUJCYUJOYUJYUJOYUJOJUJOUJOJUJOJIJOJJOJIJ
OJIGOJIOJIGOJIGXJIGJIGXJIGXFIGXIGXFIGXFZGXFGXFZXFZYBEDVECYUJOJIG
XFZ”

Таким чином звичайний набір чисел перетворюється на більш інформативний набір слів, який містить всі необхідні для аналізу програмою дані. Комбінація числових та інтервальних датасетів дозволяє охопити абсолютно всі моменти поведінки кривої, що можуть свідчити про хворобу.

3.4 Алгоритм класифікаціїчислових даних для проектування ПЗ

Для того щоб обрати алгоритм, який буде найкращим в прийнятті рішення щодо стану пацієнта, розглянемо процес обробки інформації в ЕКГ лікарем – людиною. В першу чергу лікарі дивляться на:

– висоту зубців;
– ширину горбів та прогалин;
– частоту R-Rінтервалу та стійкість ритму.

Плинність його думок під час аналізу ЕКГ на можливі діагнози можна описати за допомогою діаграми на рисунку 3.8:

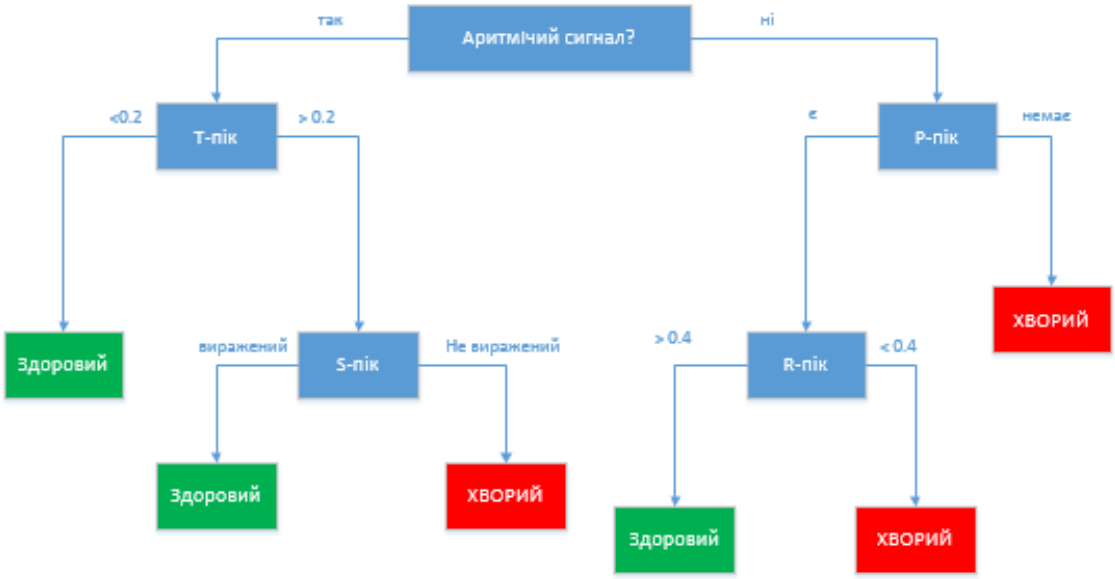


Рисунок 3.8 –Діаграма аналізу ЕКГ людиною

Така діаграма дуже нагадує приклади діаграм алгоритму машинного навчання DecisionTree або Дерево прийняття рішень (дивись рисунок 3.9).

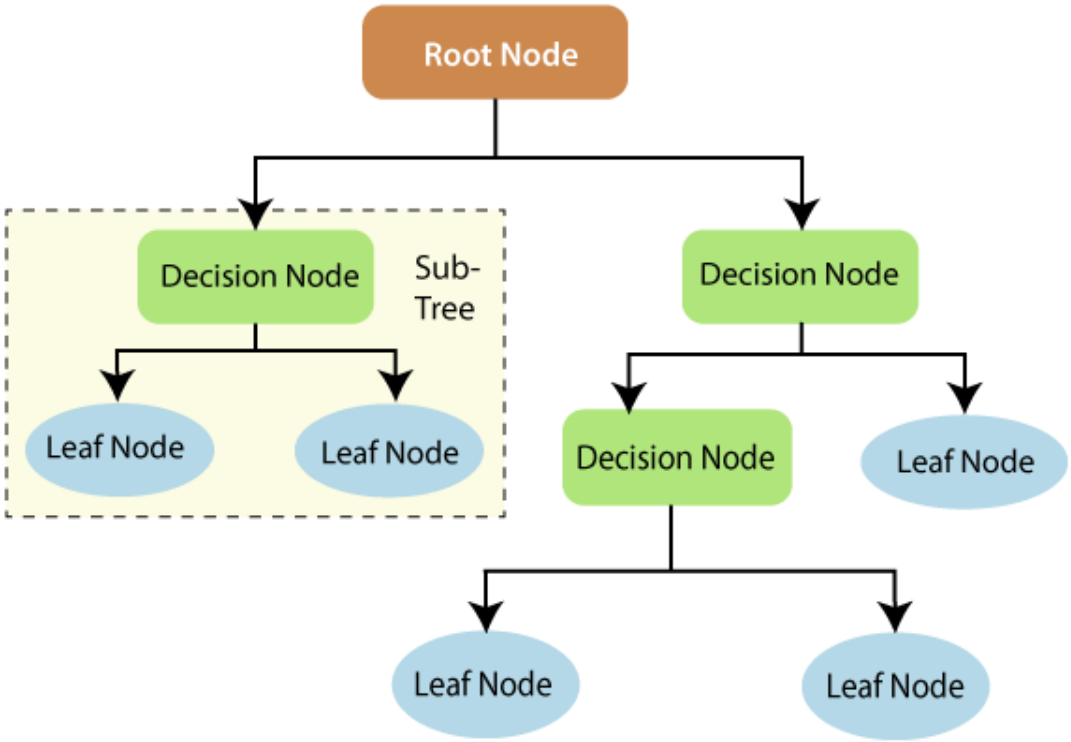


Рисунок 3.9 –Алгоритм «Дерево прийняття рішень»

DecisionTree складається з деревоподібної структуриде верхнім вузлом вважається корінь дерева, який рекурсивно розділений насерію вузлів від кореня до кінцевого вузла де рішеннядосягнуто.

Кожній з вершин дерева за винятком листя відповідає деяке питання, що має на увазі кілька варіантів відповідей. Залежно від обраного варіанта здійснюється перехід до вершини наступного рівня. Кінцевим вершинам поставлені у відповідність мітки, що вказують на відповідність об'єкта що розпізнається одному з класів.

Переваги використання алгоритму:

– дозволяють навчальні моделі з кількісними та якісними вхідними даними;

– захищені від надмірних змінних або змінних з високою кореляцією;

– мають дуже мало параметрів, на які можна налаштуватися під час навчання моделі;

– працює добре з викидами або відсутніми значеннями в наборі даних.

Недоліки використання алгоритму:

– погано обробляють шум;

– точно передбачає дані, подібні до навчальних, але погано ті, що суттєво відрізняються;

– схильні до перенавчання.

Випадковий ліс (або Randomforest)– покращена модифікація дерева прийняття рішень, що вирішує проблему з перенавчанням (дивись рисунок 3.10). Базові алгоритми навчаються на різних підмножинахознак, які також виділяються випадковим чином.

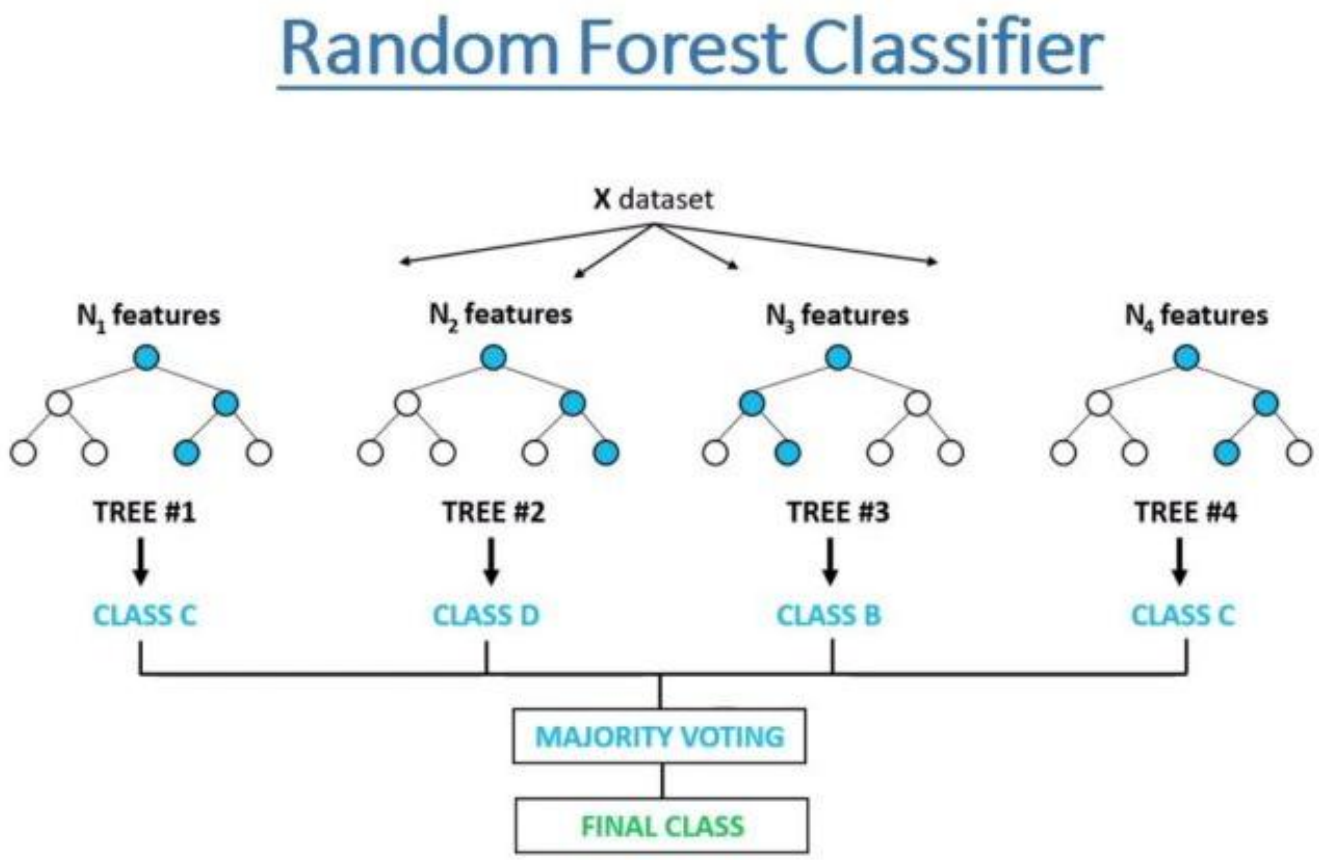


Рисунок 3.10 –Алгоритм «Випадковий ліс»

Ансамбль моделей, що використовують метод, можна побудувати, використовуючи наступний алгоритм:

– позначимо кількість об`єктів в датасетіN а кількість ознак в цих об`єктахM;

– позначимоL – вибірка окремих дерев прийняття рішень;

– для кожного дерева з L визначимо його власний dl – кількість ознак. dl має бути менше M;

– навчаємо кожну модель з L окремо;

– для формування передбачення об`єднуємо результати окремих дерев.

Для завдання класифікаціірішення обирається голосуванням за більшістю, а в задачі регресії – середнім значенням.

Переваги використання алгоритму[13]:

– має високу точність передбачення, на більшості завдань буде краще лінійних алгоритмів;

– не чутливий до масштабування та перетворення значень ознак;

– не вимагає ретельної настройки параметрів, добре працює «з коробки»;

– здатний ефективно обробляти дані з великим числом ознак і класів;

– не перенавчається, додавання дерев майже завжди тільки покращує композицію.

Недоліки використання алгоритму [13]:

– на відміну від одного дерева, результати випадкового лісу складніше інтерпретувати;

– алгоритм працює гірше багатьох лінійних методів, коли у вибірці дуже багато розріджених ознак (тексти);

– погано обробляє шум.

3.5 Алгоритм класифікації інтервальних даних для проектування ПЗ

Оскільки інтервали ЕКГ представлені у вигляді речень з окремими словами, то алгоритм для їх обробки має бути ефективним для обробки текстових даних.

Коли ми говоримо про дані, загальноприйнятою практикою є уявлення величин або категорій, що містять елементи з фіксованих списків (наприклад числа). Якщо текст – ценабір неструктурованих даних (слів) і створювався він як мова для взаємопорозуміння між людьми а не машинами, як можна перетворити його на зрозумілу програмі модель для класифікації чи прогнозування?

За допомогою NLP можна розбити неструктурований текст на більш короткі та структуровані інформаційні фрагменти. Спочатку весь текст розбивається на слова або речення. Далі корисно застосовувати механізм розбиття слів на окремі лексеми, що підвищить точність аналізу.

Такі лексеми називаються **N-грами**, або поєднання суміжних слів або букв довжиною n, які можна знайти у вихідному тексті.Основною перевагою n-грамів є те, що вони фіксують мовну структуру, перетворюють вільну мову на словник. Чим довший n-грам, тим більше контексту. Оптимальна довжина залежить від програми - якщо ваші n-грами занадто короткі, можна не вловити важливих відмінностей. З іншого боку, якщо вони занадто довгі - не побудується схема «загальних знань, алгоритм перенавчиться.

Далі можна застосовувати певний метод для кластеризації чи прогнозування результату. Для визначення чи хвора людина чи ні по її ЕКГ потрібен алгоритм, що класифікує дані на дві окремі групи. Одною з вдалих груп таких алгоритмів є логістична регресія.

Логістична регресія(дивись рисунок 3.11) - це метод статистичного аналізу, який використовується для прогнозування значення даних на основі попередніх спостережень. Отримана аналітична модель може враховувати безліч вхідних критеріїв.

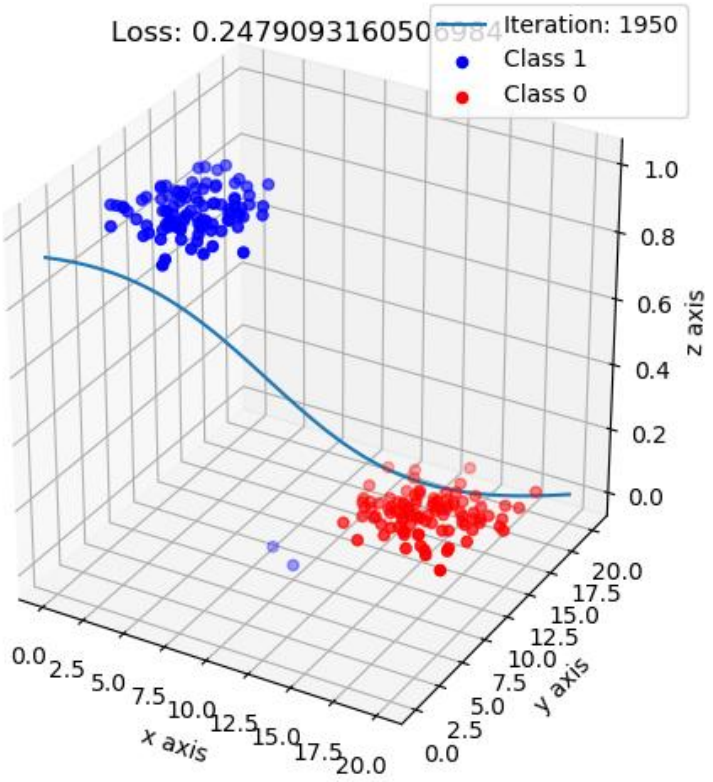


Рисунок 3.11 –Логістична регресіяв тривимірному просторі

Бінарна логістична регресія використовується тоді, коли залежна змінна (результат) може приймати лише два значення (зазвичай це «так» або «ні»). За допомогою логістичної регресії можна оцінювати вірогідність того, що подія настане для конкретного випробуваного (в випадку з ЕКГ датасетами, вірогідність захворювання).

В чому відмінність від лінійної регресії? Логістична регресія використовується, коли змінна відповіді є категоричною, а інша –коли змінна реакції безперервна, виражається в числовому еквіваленті.

Хоча створенняподібних моделей вважається добре вивченою та оптимізованою областюмашинного навчання та оптимізації, досягнення в області стохастичного градієнтазначно вдосконалили рівень якості за останні декілька років.

Велика кількість задач машинного навчання та обробки сигналів єпроблемою мінімізації втрат.Простим підходом для RLM є стохастичний градієнтний спуск (SGD).

Градієнтний спуск (рисунок 3.12) – метод виявлення мінімальних значень функціїпохибки. Мінімізація будь-якої функції– це пошук її глобального мінімуму. Ця функція використовується, щоб контролювати помилку в прогнозах машинного навчання.

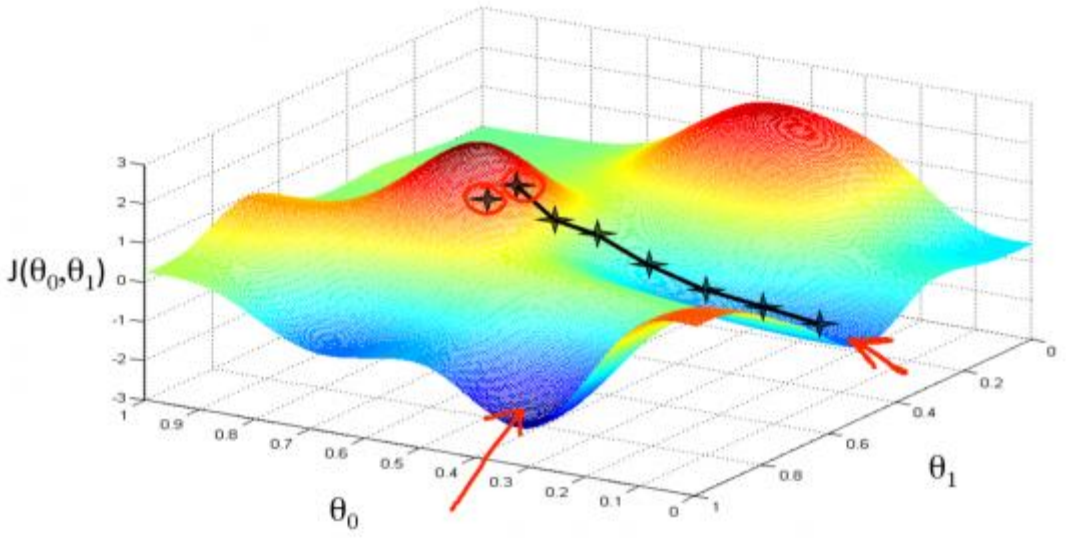


Рисунок 3.12 –Градієнтний спуск

Щоб знайти найглибшу помилку у функції, потрібно налаштувати параметри моделі. Нахил графічних функцій - похідна. Цейнахил завжди вказує на найближчу впадину.

Його час роботи не залежить від обсягу даних, і тому він дуже популярний при обробці великомасштабного набору даних. Однак він не має чіткого критерію зупинки і, займаючи відносно мало часу для отриманняприблизного рішення, надто повільно видає рішення точне.

Альтернативний підхід– SDCA**.**

Стохастичний подвійний координатний підйом (SDCA)[14] нещодавновиник як надсучасний метод вирішення широкомасштабнихпроблем з учитем. Фактично це просто процес максимізації функції правильних відповідей, а не мінімізації функціїпомилок.

Стохастичний градієнтний підйом відрізняється від звичайного градієнтного підйому тим, що градієнт на кожному кроці рахується не як сума градієнтів від кожного елементу набору, а як градієнт від одного, випадково обраного елемента. Якщо при повному градієнтному спуску результат завжди покращується на кожній ітерації, то при стохастичному функціонуванні результат може навіть знизитися, проте в загальному все одно буде прагнути до глобального максимуму.

Феномен подвійного підйому або спуску (дивись рисунок 3.13):

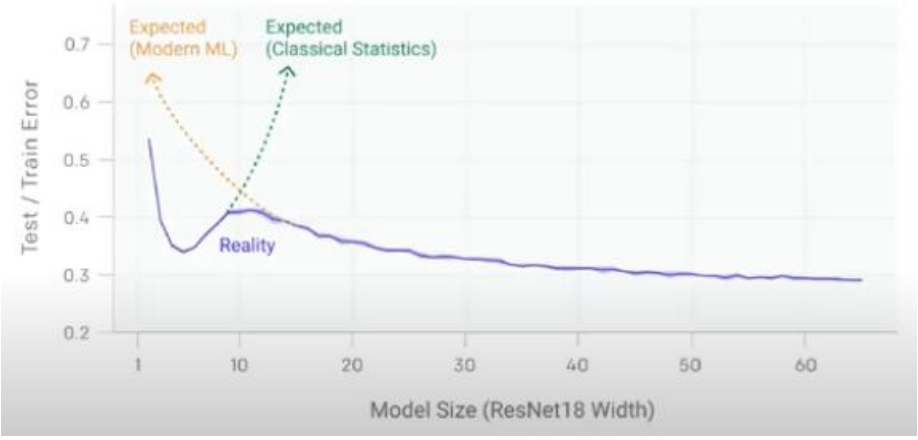


Рисунок 3.13 –Ефективність подвійного градієнтного спуску

4 ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДУ

Програмна реалізація має стати помічником лікаря в діагностуванні хвороб серця. Фактично вона має стати першою ланкою в етапі обстежування хворого. Після закінчення роботи програми лікар має отримати прогноз та найточніші ілюстрації частин ЕКГ, за якими цей прогноз було створено.

4.1 Вимоги до програмного забезпечення

Відповідно до поставленої задачі, для реалізації такого програмного забезпечення необхідно виконати наступні кроки:

– знайти базу даних, звідки можна витягнути ЕКГ з хворобами і без них, ЕКГ мають бути без попередньої обробки або спрощення;

– реалізувати запропонований алгоритм зчитування та фільтрації даних;

– реалізувати запропоновані алгоритми для аналізу ЕКГ за піковими значеннями, а саме: алгоритм пошуку локальних мінімумів та максимумів з урахуванням можливої аномалії, класифікатор на основі алгоритму випадкового лісу;

– реалізувати запропоновані алгоритми для аналізу ЕКГ за інтервальними значеннями, а саме: алгоритм перетворення інтервалів у речення, стохастичний подвійний координатний підйом;

– розробити окремі модулі, що відповідають за формування експертної системи на основі штучного інтелекту;

– розробити інтерфейс користувача для використання програмних алгоритмів;

– дослідити роботу запропонованих алгоритмів, оцінити їх точність та швидкість обробки даних;

– порівняти отримані результати з існуючими методами розпізнавання ЕКГ хвороб.

На основі сформульованих задач можна виділити деякі **функціональні вимоги** до програмного забезпечення.

Вимоги до програми користувача:

– зчитування ЕКГ даних з .csv файлів;

– розбиття ЕКГ даних на PQRSTінтервали;

– формування лінгвістичних ланцюгів на основі перепаді висот на часовому ряду;

– запис даних про PQRSTінтервали в датасети з розширенням .tsv;

– аналіз отриманої ЕКГ на основі відомих хвороб;

– вивід результатів аналізу на екран у вигляді графіків та передбачень.

Вимоги до експертної системи:

– зчитування ЕКГ даних з .tsv файлів;

– створення систем штучного інтелекту, що вчиться на даних з .tsv файлів за запропонованими алгоритмами;

– аналіз точності роботи штучного інтелекту.

Також можна виділити деякі **нефункціональні вимоги** до програмного забезпечення:

– можливість запуску програми на операційних системах Windows 7, 8, 10;

– інтуїтивний інтерфейс програми, зрозуміле подання інформації про результати;

– можливість використання експертних систем за межами основної програми користувача як бібліотеки.

Метою дослідженняє створення методу обробки та аналізу ЕКГ даних, що підвищить ефективність та збільшить точність в порівнянні з уже існуючими методами.

4.2 Засоби розробки

Під час розробки програмного забезпечення було вирішено використовувати мову C#. Такийвибір зумовлений наступними факторами:

– доступне об`єктно орієнтоване програмування;

– автоматична робота з пам`ятю;

– строга типізація;

– великий набір бібліотек, що задовольняють всі потреби програмного забезпечення.

Окрему увагу варто приділити наступним засобам .NETплатформи, що стали в нагоді в процесі розробки.

4.2.1 ML.NET[15]

Усі звикли, що зазвичай експерти з машинного навчання працюють з більш «історично типовими» мовами для цього виду діяльності, наприклад такими як Python. Проте не всі програмісти мають бажання використовувати чужу, незнайому їм мову, деяким просто не подобається Python.

На щастя і в інших мовах є аналоги, що дозволяють заглибитьсь у алгоритми машинного навчання. Однією з таких аналогів стала велична бібліотека ML.NET від Microsoft(дивись рисунок 4.1).

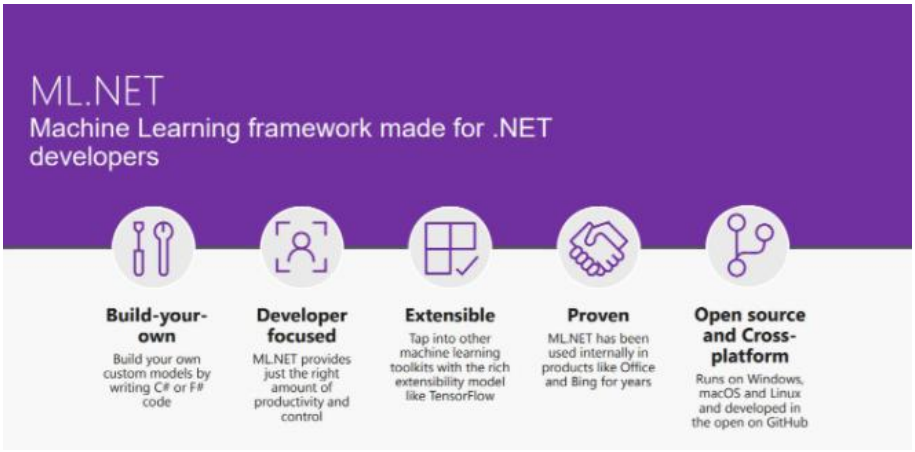


Рисунок 4.1 – Бібліотека для машинного навчання на C#

ML.NETнадаєпрограмістуінструменти для аналіти, формування прогнозів та створення моделей машинного навчання. Розробники можуть навчити модель самостійно або повторно використати існуючу модель з будь-якого джерела.Готова модель зберігається в зручному .zip форматі та може бути використана за потреби будь-якою програмою.

За допомогою ML.NET можна виконувати такі завдання:

– класифікація;
– регресія;
– виявлення аномалій;
– глибоке навчання;
– системи рекомендацій;
– обробка природних мов.

Класи бібліотеки мають зручні та інтуїтивні інтерфейси, для виведення результатів та оцінки ефективності машинного аналізу передбачено стандартну оцінку точності алгоритму.

За потреби базові алгоритми можна параметризувати, розширювати власними реалізаціями функцій. Бібліотека покрита документацією та має власний репозиторій на GitHub з прикладами реалізації найпопулярніших алгоритмів на таких відомих

задачах як: класифікація тюльпанів, пошук котів на картинках, виявлення негативних агресивних коментарів, прогнозування погоди, тощо.

4.2.2 WINDOWS FORMS[16]

Цедоволі стара технологія, що тягне свій початок ще від часів зародження мови C#. WindowsForms–легка для освоювання бібліотека, що має перевагу в простоті перед MFC та WCF.

Windows Forms– це інтерфейс для побудови програм для ПК на операційній системі Windows. Бібліотека пропонує один з найбільш продуктивних способів створення настільних програм за допомогою інструментів візуального дизайнера.Функціональні можливості, такі як перетягування та розміщення елементів керування, дозволяють зосередитись на реалізації програмної логіки та не витрачати багато часу на створення користувацького інтерфейсу.

Бібліотека має потужний інтерфейс для роботи з графіками, що є незамінним при написанні програми, яка працює з ЕКГ даними –**EssentialChart**. Це засіб для налаштування діаграм чи графіків, що виводяться на панель у вигляді, подібному до презентацій в PowerPoint. Ідеальне рішення для розробників, які хочуть додати вдосконалені, багатофункціональні та візуально привабливі діаграми до своїх програм Windows Forms.Діаграма використовується як засіб для графічного відображення двох значень. Наприклад, лінійна діаграма може бути використана у звітах про статистику здоров'я, де вона може відображати показники з часом.

Деякі ключові особливості основної діаграми наведені нижче:

– модель даних діаграми - об'єктна модель даних, що може бути заповнена як з файлу чи колекції, так і безпосередньо з таблиці в базі даних;

– існує можливість автоматичного прорахунку інтервалів будь-якого діапазону;

– є інструменти експорту готової діаграми у файли найпопулярніших форматів, здатних зберегти графічне представлення.

4.3 Архітектура ПЗ

Програма ділиться на три окремих незалежних додатки:

– ECGAnomalyDetection (програмне забезпечення для взаємодії з користувачем);

– ECGPeaksMLAnalize (програмне забезпечення, що навчає модель аналізу опорних точок);

– ECGWordsAnalize (програмне забезпечення, що навчає модель аналізу інтервалів у вигляді речень).

4.3.1 Програмне забезпечення для взаємодії з користувачем

Побудований за архітектурним шаблоном під назвою MVP (дивись рисунок 4.2).

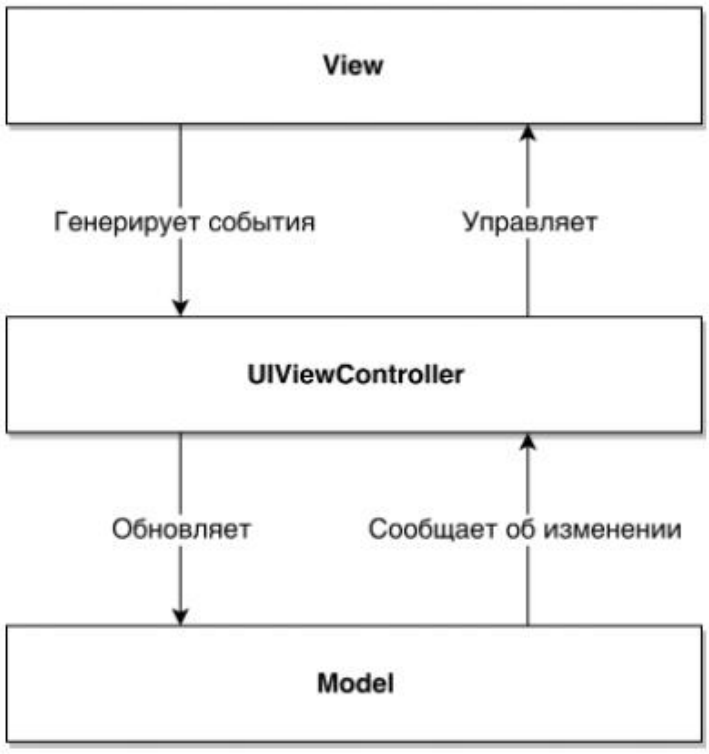


Рисунок 4.2 – Концепція шаблону Model-View-Presenter

Підхід дозволяє створювати абстракцію View-частини додатку. Для цього необхідно виділити інтерфейс з певним набором властивостей та методів.

Presenterотримує посилання на реалізацію інтерфейсу, підписується на події, що виникають (наприклад натиснення кнопки, завантаження файлу) і змінює модель.

В програмному забезпеченні роль View відіграє стандартна форма WindowsForms (дивись рисунок 4.3). На ній розташовані такі елементи, як:

– кнопка завантаження ЕКГ сигналу з файлу;
– кнопка перетворення ЕКГ даних на параметризовані списки;
– кнопка аналізу ЕКГ;
– представлення для ЕКГ графіків;
– радіокнопки для вибору хвороби для аналізу.

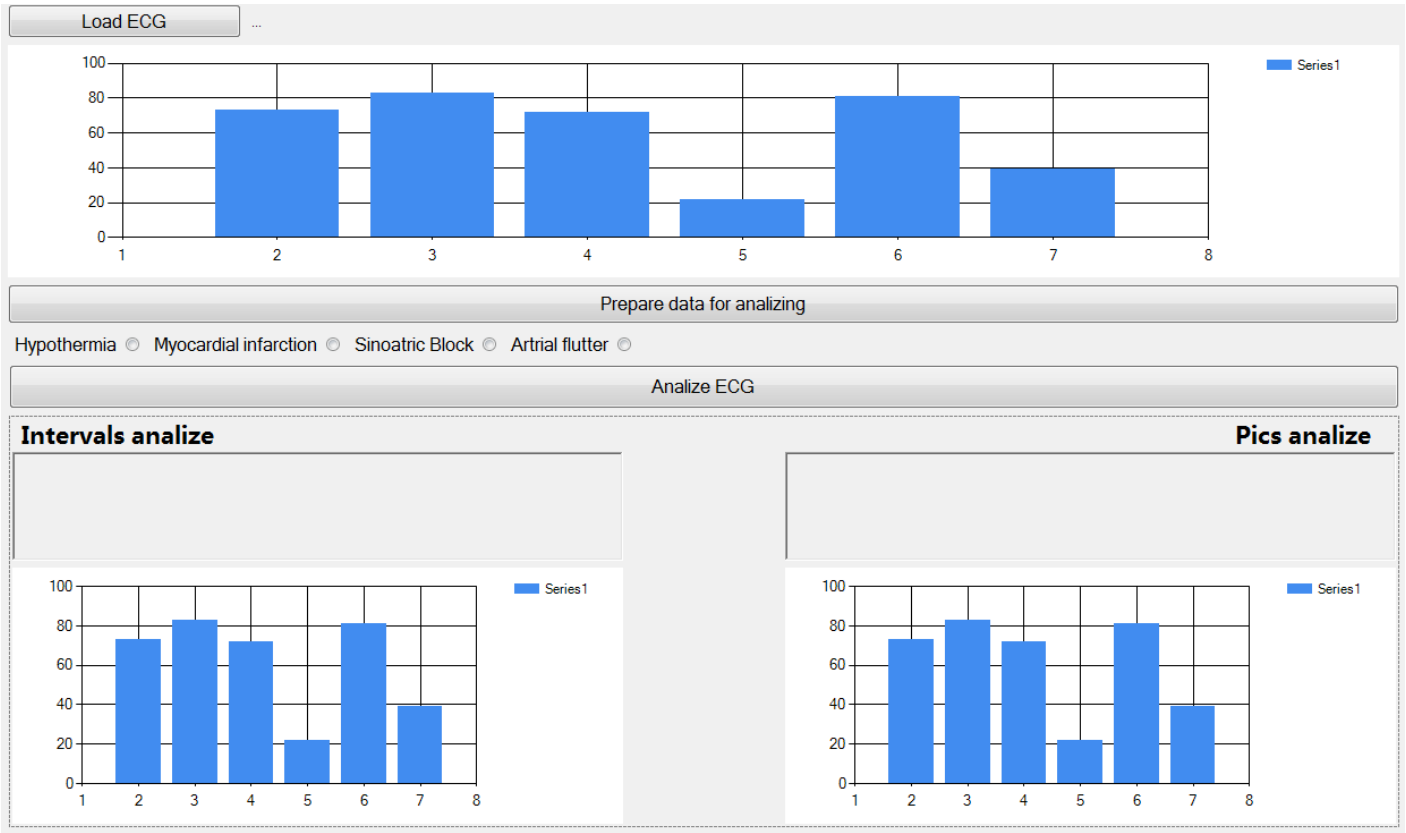


Рисунок 4.3 – Представлення View

В ролі Presenter виступає клас Screen. Під час взаємодії з формою, він перехоплює події, що виникають (а саме натиск на одну з кнопок) та змінює модель або витягує чи обробляє її дані.

Моделлю виступає клас ECGConverter, який зберігає в собі інформацію про повноцінні інтервали ЕКГ та формує їх параметризовані списки за потреби.

Також важливу роль відіграють допоміжні сервіси:

– FileManager – допомагає зберігати в файли новоутворені датасети;

– WaveletTransformation – за допомогою вейвлет перетворення зменшує кількість точок на графіку ЕКГ в чотири рази без втрати інформативності;

– PicksAnalizator – аналізує параметризовані списки ЕКГ інтервалів на наявність в них захворювання (інформація про локальні мінімуми та максимуми);

– IntervalsAnalizator – аналізує параметризовані списки ЕКГ інтервалів на наявність в них захворювання (інформація про перепади в графіку на інтервалах).

Повну інформацію про компоненти ECGAnomalyDetection додатку можна побачити на діаграмі класів (дивись рисунок 4.4).

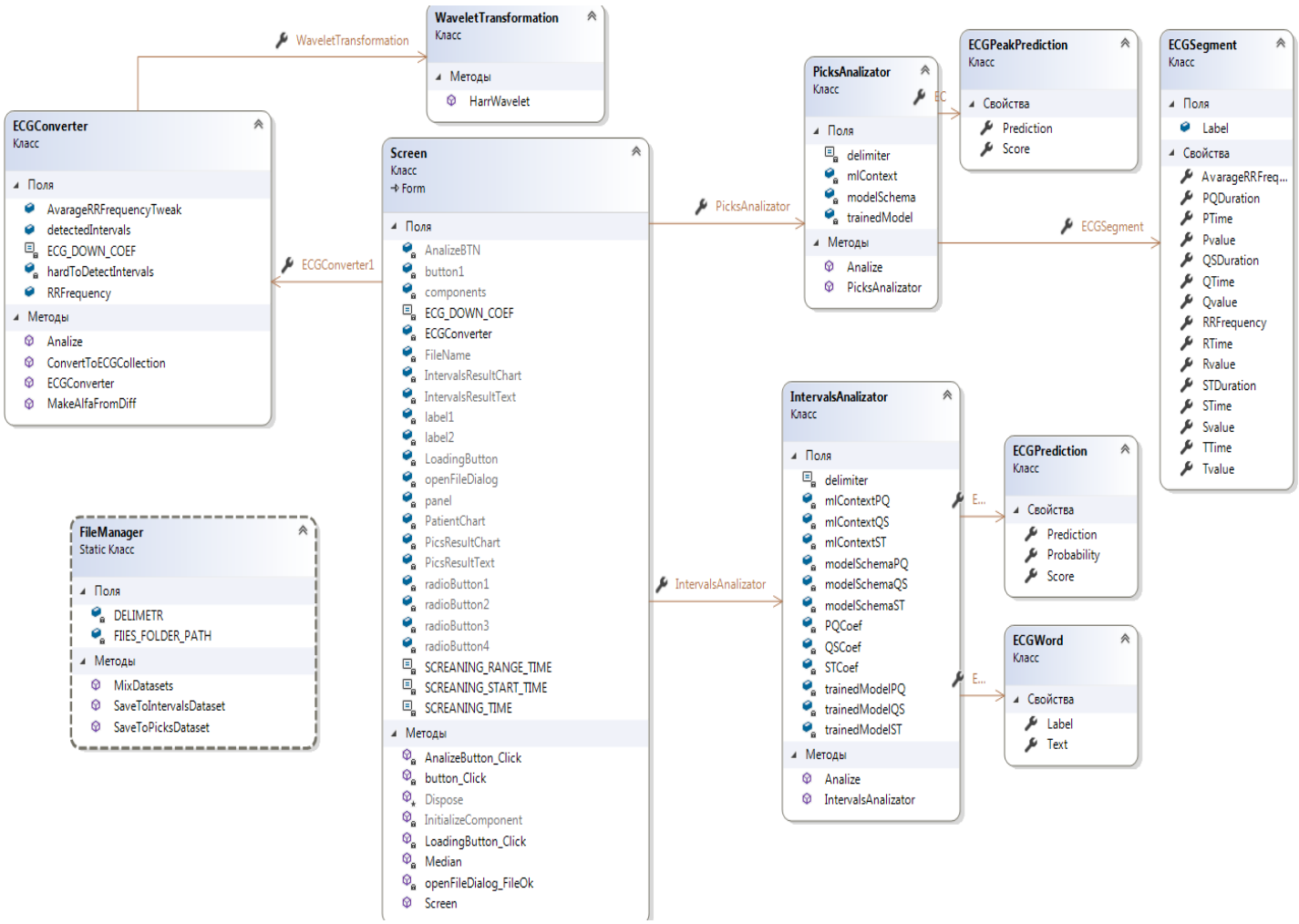


Рисунок 4.4 – Діаграма класів клієнтської частини програми

4.3.1 Бібліотеки для формування прогнозів щодо хвороби

Допоміжні бібліотеки, що мають своєю метою утворення штучного інтелекту, що аналізує ЕКГ параметри на наявність в них хвороб та видає прогноз щодо вірогідності захворювань у пацієнта.

Для створення штучного інтелекту в ECGPeaksMLAnalize використовується стандартний клас MlContext.Regression.Trainers.FastForest. Параметри його налаштування можна побачити в таблиці 4.1.

Таблиця 4.1 – Параметри налаштування алгоритму

NumberOfLeaves	Максимальна кількість листків у кожному дереві

	регресії
NumberOfTrees	Загальна кількість дерев рішень, які потрібно створити в ансамблі
MinimumExampleCountPerLeaf	Мінімальна кількість точок даних, необхідних для формування нового листя дерева

Для створення штучного інтелекту в ECGWordsAnalize використовується стандартний клас MlContext.BinaryClassification.Trainers.SdcaLogisticRegression. Параметри його налаштування можна побачити в таблиці 4.2.

Таблиця 4.2 – Параметри налаштування алгоритму

L1Regularization	Додає штрафну величину до функції мінімізації лише для великих коефіцієнтів
L2Regularization	Додає штрафну величину до функції мінімізації
MaximumNumberOfIterations	Максимальна кількість ітерацій для пошукумаксимуму

4.4 Інструкція користувача

Відкривши програму, користувач побачить перед собою панель з кнопками та пустими полями (дивись рисунок 4.5).

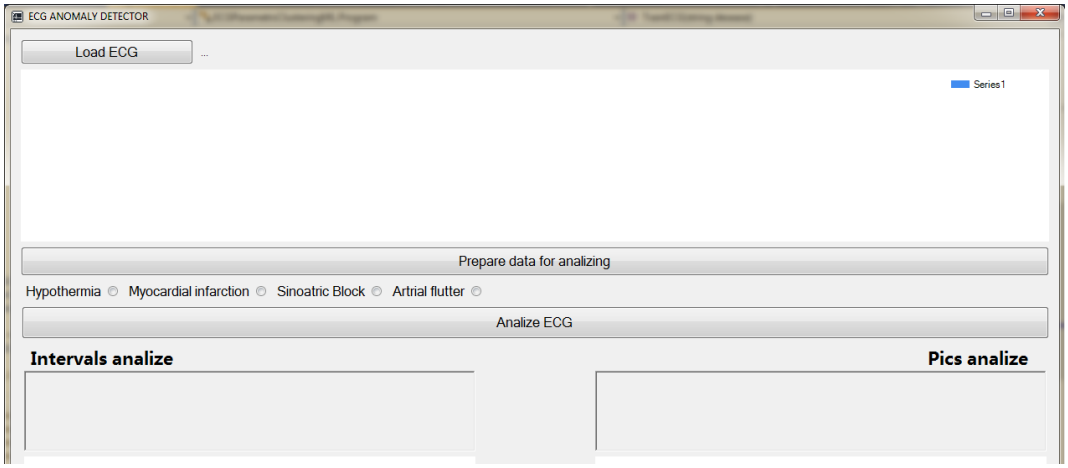


Рисунок 4.5 – Початок роботи з програмою

Для завантаження ЕКГ даних необхідно натиснути на кнопку «LOADECG» та вибрати файл у окремому вікні. Після цієї операції користувач побачить графічне представлення ЕКГ даних у вікні (дивись рисунок 4.6). За потреби він може скористатись функцією виділення області на графіку для ретельнішого перегляду.

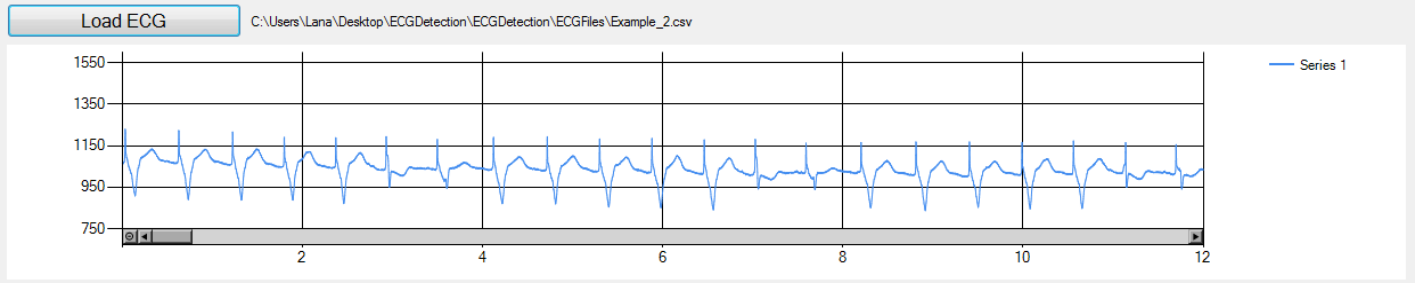


Рисунок 4.6 – ЕКГ графік

Після цього користувач має натиснути на кнопку «Preparedataforanalizing» та отримати у відповідь графічне підтвердження розбиття ЕКГ на R-Rпіки (дивись рисунок 4.7)

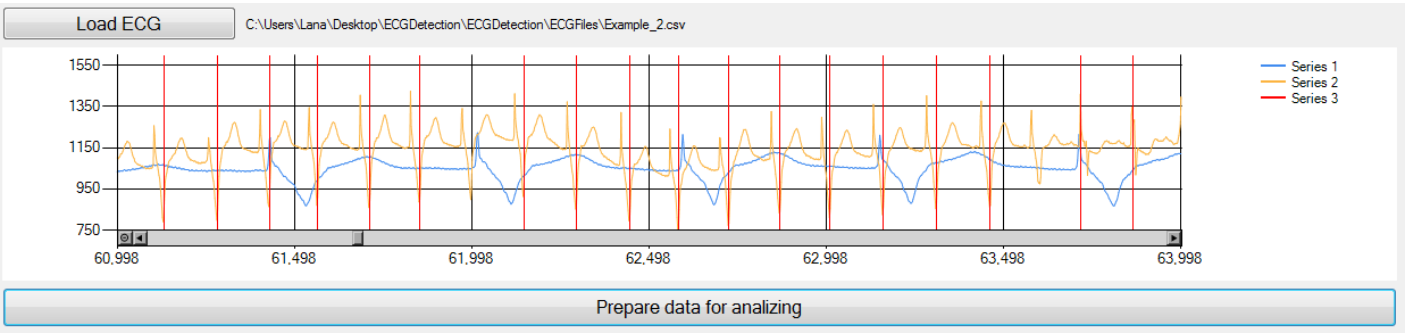


Рисунок 4.7 – Графікрозбиття ЕКГ на R-Rпіки

Далі користувач може обрати хворобу для аналізу та натиснути на кнопку «AnalizeECG». Через деякий час він отримує прогноз у поляхIntervalsanalize та PicksAnalize (дивись рисунок 4.8).

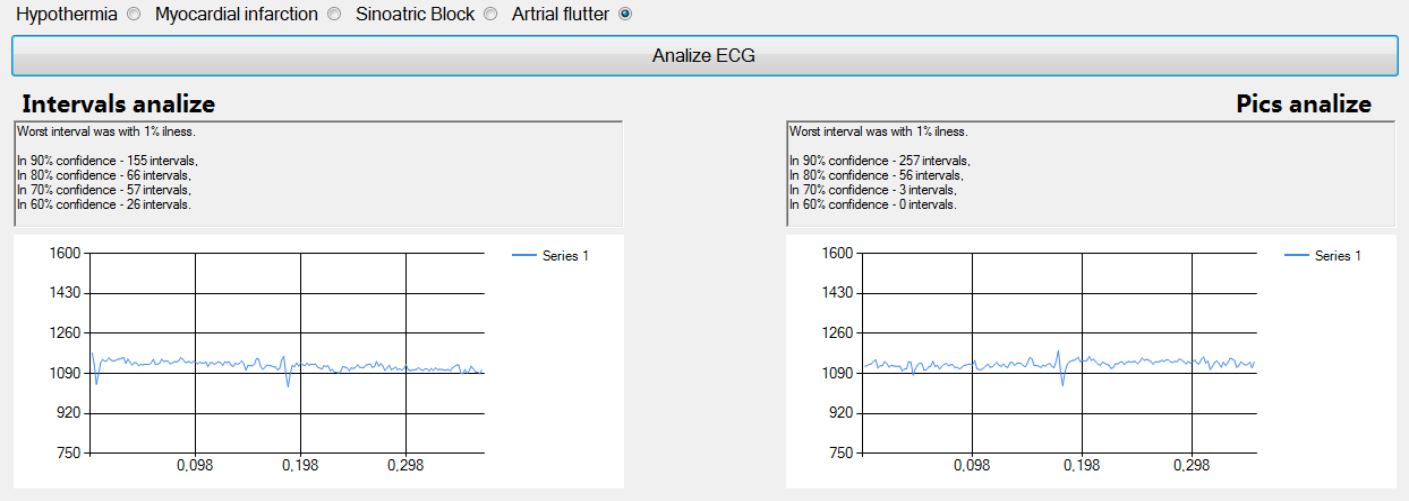


Рисунок 4.8 – Результати аналізу на хворобу

4.5 Висновки до розділу

За допомогою інструментів, таких як C#, WindowsForms та ML.NET, було розроблене програмне забезпечення, що реалізує метод класифікації та аналізу ЕКГ даних у вигляді часового ряду.

Програма користувача працює безпомилково, правильно обробляє та видає очікуваний та інформативний результат на наборах даних.

В результаті були сформовані дві незалежні бібліотеки: ECGPeaksMLAnalize та ECGWordsMLAnalize, що можуть бути використаними для потреб інших досліджень та вдосконалення створеного методу в майбутньому.

5 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ МЕТОДУ

Для того щоб виявити точність створеного алгоритму та порівняти його з уже існуючими аналогами за ефективністю, треба визначитись з тим, які метрики використовуватимуться при його аналізі.

5.1 Ключові критерії оцінки ефективності роботи методу

Першим критерієм оцінки буде можливість розпізнавання ключових параметрів ряду на правильному, зашумленому та хворому ЕКГ.

Така оцінка обчислюватиметься за формулою 5.1:

$$k_{\text{розпізнавання}} = \frac{n_{\text{оброблених даних}}}{n_{\text{загальне}}} \qquad (5.1)$$

Другим важливим критерієм є алгоритмічна складність аналізу даних, вона обчислюється за формулою 5.2:

$$O\big(f(n)\big) = 2 * \frac{n_{\text{заг.}}}{4} + \sqrt{\frac{n_{\text{заг.}}}{4}} \qquad (5.2)$$

Третім важливим критерієм є швидкість алгоритму класифікації даних. Вона вимірюється експериментальним шляхом.

Також додатково можна визначити **ефективність методів класифікації** часових рядів за допомогою машинного навчання.

Перш ніж заглибитися в концепцію точності, варто розрізняти помилки двох різних типів:

– помилка першого типу, або класифікація позитивного результату як негативного. Позначимо її FP;

– помилка другого типу, або класифікація негативного результату як позитивногоFN.

Усі передбачення, що видаються алгоритмом загалом можна поділити на чотири типи (дивись рисунок 5.1).

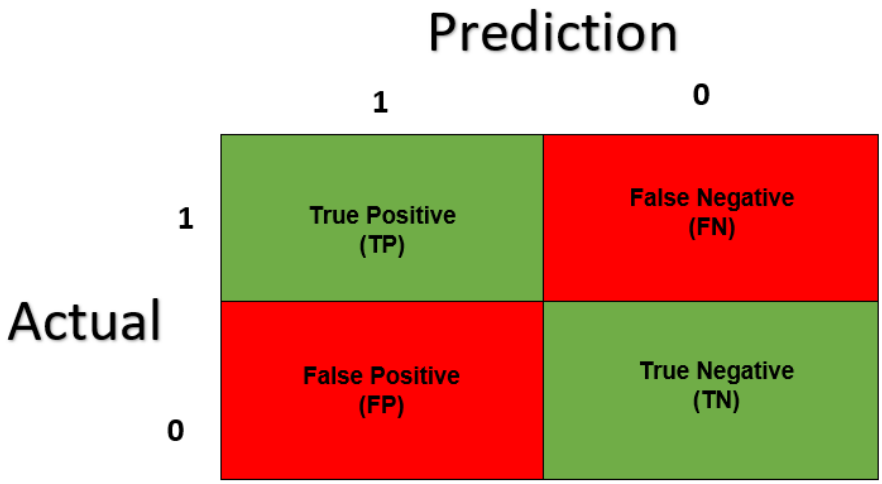


Рисунок 5.1 –Матриця оцінки результатів алгоритму класифікації

Тепер завдяки цій класифікації можна виділити ключові метрики оцінки алгоритму.

Чутливість алгоритму–здатністьалгоритму розпізнавати об`єкти з групи 1 серед усіх запропонованих.Таким чином, чутливість дозволяє побачити частку істинно-позитивних класифікацій щодо загального числа позитивних класифікацій.Обчислюється за формулою 5.3:

$$Sensitivity = \frac{TP}{TP+FN} \qquad (5.3)$$

Чутливістьхарактеризуєточністьмоделібінарноїкласифікаціїщодопозитивнихспостережень. Точністьмоделіщодонегативнихспостереженьвідображає величина, звана специфічністю.

Специфічність алгоритму–здатність алгоритму не вважатиоб`єкти з групи 0 об`єктами групи 1.Обчислюється за формулою 5.4:

$$Specific = \frac{TN}{TN+FP} \qquad (5.4)$$

Чутливість тим нижча, чим більше число помилково-негативних спостережень, тобто позитивних спостережень, помилково розпізнаних, як негативні. Верхня межа чутливості дорівнює 1, коли помилково-негативні спостереження відсутні. Модель, що володіє високою чутливістю, забезпечує більшу ймовірність правильного розпізнавання для позитивних прикладів.

Площа під ROCAUC–кривоюдозволяє знайти оптимальний баланс між чутливістю і специфічністю моделі, в якому чутливість і специфічність рівні 1, коли помилково-позитивні і хибно-негативні класифікації відсутні. Має наблизитись до одиниці.

Наприклад AUC=0,7можна інтерпретувати як імовірність 70%, що моделі вдасться розділити дані правильно.AUC=0–такийкласифікаторзавждирозпізнаєпозитивний приклад як негативний, тобтоймовірністьпомилки становить 100%.

На практиці в залежності від значення AUC ефективність моделі класифікується за таблицею 5.1:

Таблиця 5.1 – Оцінка значень AUC

Значення	**Оцінка**
<=0.5	Модель не працює, її ефективність нижча за звичайний рандом
>0.5 <=0.6	Модель можевидаватиправильнірезультати за наявностідужехарактернихознак у протестованомуекземплярі. Проте для більшостівипадків все щенепрацездатна.
>0.6 <=0.8	Модель працездатна, можевикористовуватись для задач де цынапомилки не є критично важливою. Наприклад, в контекстній рекламі, в ігрових проектах для демонстрації можливостей машинного навчання, у програмах підбору товарів для користувача тощо.
>0.8 <=0.94	Модель працює з хорошою точністю, може використовуватись для будь-яких цілей окрім тих, де ціною помилки є людське життя або втрата великих грошових сум.
>0.94	Ідеальна модель, може використовуватись будь де. Це те значення, до якого варто намагатись дійти при побудові будь якого алгоритму.

5.2 Оцінка ефективності роботи методу

Коефіцієнт розпізнавання було визначено на даних різної довжини та складності (дивись таблицю 5.2):

Таблиця 5.2 – Коефіцієнт розпізнавання ЕКГ

Складність даних	**Розмір в мілісекундах**	**Розмір в інтервалах**	**Кількість розпізнаних інтервалів**	**Коефіцієнт**
Легка (синусоїда нормального типу)	150000	491	434	88.39%
Середня (синусоїда неправильна, алеінтервали яскраво виражені)	150000	512	414	80.85%
Важка (синусоїда зашумлена, інтервали не виражені)	150000	414	316	76.32%
Легка (синусоїда нормального типу)	350000	1132	990	87.45%
Середня (синусоїда неправильна, але інтервали яскраво виражені)	350000	1184	974	82.26%
Важка (синусоїда зашумлена, інтервали не виражені)	350000	1025	712	69.46%

Легка (синусоїда нормального типу)	600000	1953	1523	77.98%

Продовження таблиці 5.2

Середня (синусоїда неправильна, але інтервали яскраво виражені)	600000	2021	1671	82.68%
Важка (синусоїда зашумлена, інтервали не виражені)	600000	1686	1079	63.99%

Як видно з таблиці, коефіцієнт розпізнавання майже ніколи не опускається нижче 75%. А це означає, що принаймні 3/4 від усього об`єму даних розпізнано як валідні інтервали та може бути подано на аналіз. Цього цілком достатньо, щоб сформувати діагноз.

Ефективність аналізатору числових характеристик:

– гіпотермія – 98.26%;

– інфаркт міокарда – 98.13%;

– синоатріальна блокада – 98.84%;

– мерехтіння передсердь – 99.02%.

Ефективність аналізатору інтервальних характеристик

В залежностівідзначущостіінтервалу для конкретноїхворобиефективність алгоритму можезнижатись (дивись таблицю 5.3). Тому найкращимрішенням буде аналіз ЕКГ по найкращійознаці.

Таблиця 5.3 – Ефективність інтервальних аналізаторів

Гіпотермія		
PQ	TP	88.46%

	FP	17.42%
	TN	80.53%

Продовження таблиці 5.3

	FN	4.77%
	ROC AUC	**94.01%**
QS	TP	81.08%
	FP	14.88%
	TN	85.85%
	FN	3.83%
	ROC AUC	**88.58%**
ST	TP	97.26%
	FP	6.31%
	TN	91.94%
	FN	1.44%
	ROC AUC	**98.53**
Інфаркт міокарда		
PQ	TP	89.71%
	FP	7.21%
	TN	95.34%
	FN	8.21%
	ROC AUC	**97.4%**
QS	TP	82.65%
	FP	15.64%
	TN	85.31%
	FN	10.12%
	ROC AUC	**90.87%**
ST	TP	87.83%

	FP	7.31%
	TN	95.17%

Продовження таблиці 5.3

	FN	6.13%
	ROC AUC	**96.73%**
Мерехтіння передсердь		
PQ	TP	85.52%
	FP	14.39%
	TN	85.71%
	FN	16.03%
	ROC AUC	**94.74%**
QS	TP	83.67%
	FP	23.94%
	TN	82.44%
	FN	2.16%
	ROC AUC	**83.15%**
ST	TP	97.24
	FP	4.35%
	TN	94.44
	FN	2.16%
	ROC AUC	**98.99%**
Синоатріальна блокада		
PQ	TP	84.81%
	FP	12.9%
	TN	89.92%
	FN	17.15%
	ROC AUC	**95.51%**

QS	TP	80.55
	FP	19.49%

Продовження таблиці 5.3

	TN	80.47
	FN	10.47%
	ROC AUC	**89.14%**
ST	TP	85.52%
	FP	14.39%
	TN	85.71%
	FN	16.03%
	ROC AUC	**93.53%**

Середнє значення ефективності для інтервальних показників становить близько **97.6%** по найкращому для аналізу інтервалу.

5.3 Висновки до розділу

Були сформовані основні критерії за якими виконується дослідження ефективності алгоритму. За цими критеріями було проаналізовано алгоритми класифікації числового ряду за піками та за інтервалами.

Отримані результати показані на рисунках 5.2 – 5.3.

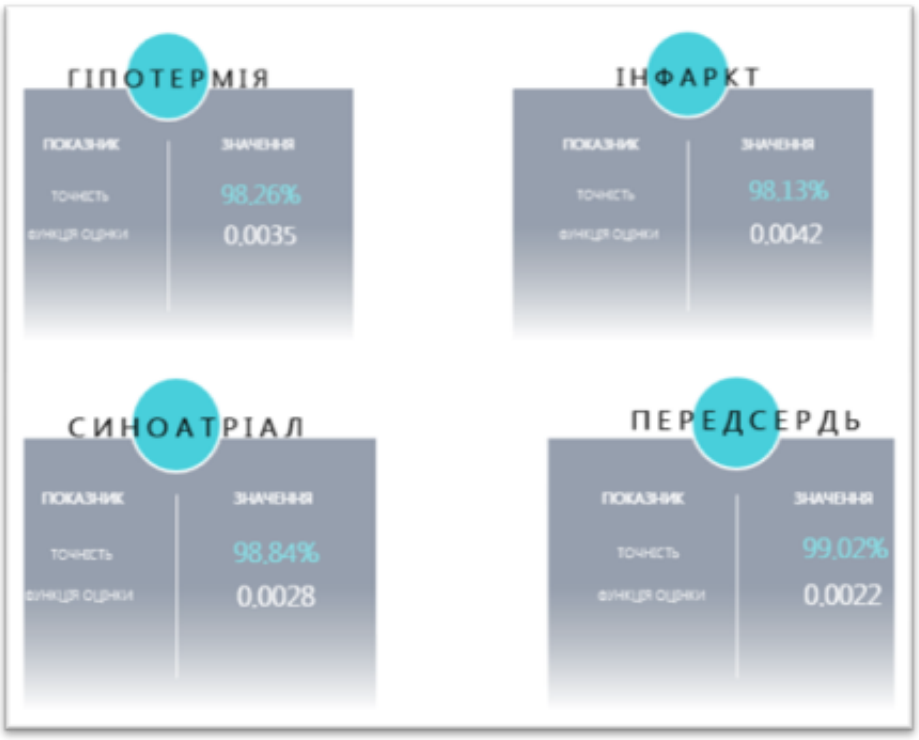


Рисунок 5.2 –Ефективність пікового методу

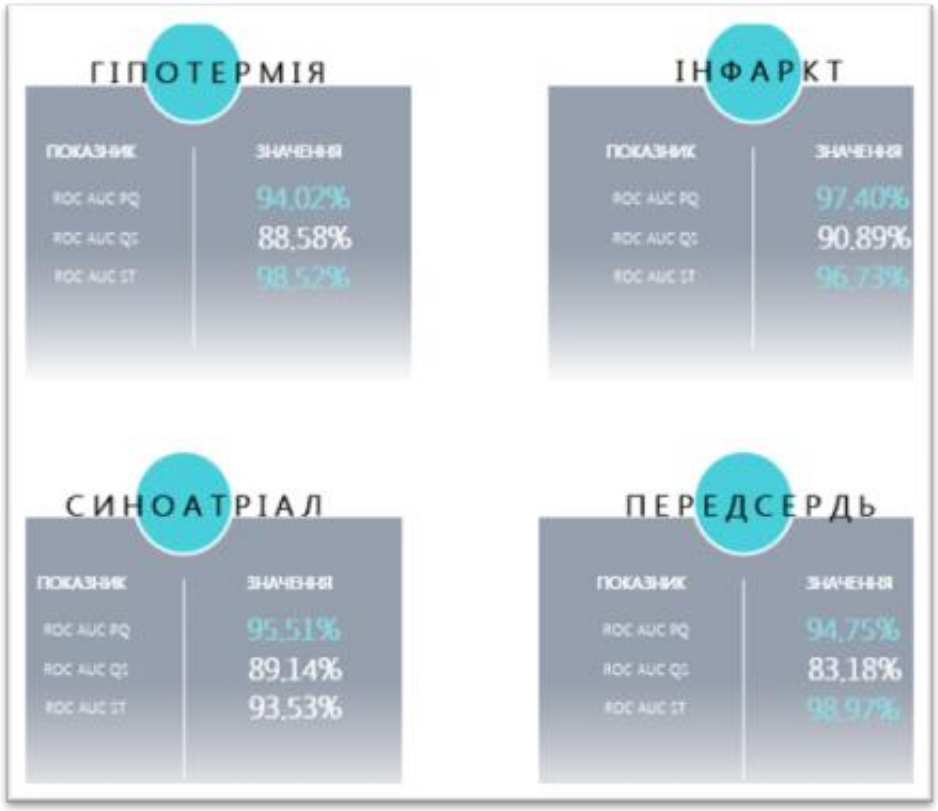


Рисунок 5.3 –Ефективність інтервального методу

Приріст ефективності алгоритму в порівнянні з аналогами становить 7.13% та 6.08% для числових та інтервальних показників відповідно (дивись рисунок 5.4).

Рисунок 5.4 –Приріст ефективності нового методу

ВИСНОВКИ

В магістерській роботі було проведено аналіз інформаційних потреб прикладної області (визначено які саме дані та ознаки є суттєвими для визначення ЕКГ аномалій).

Був зроблений критичний аналіз існуючих методіваналізу та класифікації даних на прикладі визначення аномалій ЕКГ, що представлене у вигляді часового ряду. На прикладі цих методів було проаналізовано основні сильні та слабкі сторони, визначено найкращі ознаки, що призводять до збільшення точності алгоритму.

На основі аналізу існуючих методів було запропонованота реалізованоалгоритм зчитування та фільтрації даних, розроблено та реалізовано алгоритми для аналізу часового ряду за піковими значеннями(знаходження локальних мінімумів та максимумів, класифікатор на основі алгоритму випадкового лісу), запропоновано та реалізовано алгоритми для аналізу часового рядуза інтервальними значеннями (перетворення інтервалів у речення, стохастичний подвійний координатний підйом).

Визначено технологічний стек та розроблено архітектуру програмного забезпечення. В результаті розробки програмного забезпечення було створено дві незалежні бібліотеки, що реалізують алгоритми створення штучних інтелектів, які здійснюють класифікацію часових рядів ЕКГ за піковими та інтервальними показниками відповідно. Вони можуть використовуватись стороннім програмним забезпеченням для аналізу ЕКГ даних.

В результаті аналізу ефективності розробленого методу аналізу та класифікації даних (на прикладі визначення аномалій ЕКГ) було отримано приріст в точності на 6% та на 7% за інтервальними та піковими показниками відповідно в порівнянні з методами–аналогами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Циммерман Ф. Клиническая электрокардиография, 2013.– 424 с.

2) Мешков А.П. Азбука клинической электрокардиографии,1998.– 149 с.

3) В.І. ДубровінКомп’ютерніметодиінтелектуальноїобробкиданих– Запоріжжя ЗНТУ, 2013.– 105 с.

4) Волосатова Т.М., Малышев А.П. Улучшение сигнала электрокардиограммы на основе алгоритма удаления дрейфа его изолинии//Интернет журнал «НАУКОВЕДЕНИЕ», 2017.–9 с.

5) Зотов Д.Д. Современные методы функциональной диагностики в кардиологии–СПбГПМА, 2000.–51 с.

6) Волосатова Т. М., Малышев А. П.Автоматизированная система анализа и интерпретации электрокардиосигнала//Радиооптика МГТУ им. Н. Э. Баумана, 2016.–18 с.

7) M. Thill, S.Däubener, W.Konen, T.Bäck Anomaly Detection in Electrocardiogram Readings with Stacked LSTM Networks, 2016.– 9 с.

8) Дубровин В. И., Твердохлеб Ю. В., Харченко В. В. Автоматизированная система анализа и интерпретации ЭКГ, 2014. – 8с.

9) Мустафаев А.Г., Темирбулатов М.А., Омаров Р.С Определение аномалий сердечного ритма и выявление заболевания сердца при помощи нейронных сетей, 2017. – 11с.

10) J. Pereira, M. Silveira Unsupervised Representation Learning and Anomaly Detection in ECG Sequences, 2017 – 18 с.

11) Волосатова Т.М.,Спасенов А.Ю.,ЛогуноваА.О. Автоматическая система анализа и интерпретациикардиосигнала, 2016 – 18 с.

12) MIT-BIH ArrhythmiaDatabase, [Електронний ресурс]. – Режим доступу: www.physionet.org/content/mitdb/1.0.0/

13) Кіншаков Е.В.Моделювання та прогнозування великих наборівданихзасобами машинного навчання, 2020. – 72с.

14) S. Shalev-Shwartz, T. Zhang Stochastic Dual Coordinate Ascent Methods for Regularized Loss Minimization, 2013. – 33с.

15) Чтотакое ML.NET и принципыработыэтойсистемы, [Електроннийресурс]. – Режимдоступу: https://docs.microsoft.com/ru-ru/dotnet/machine-learning/how-does-mldotnet-work

16) WindowsForms и интеллектуальные клиентские приложения, [Електроннийресурс]. – Режимдоступу: https://docs.microsoft.com/ru-ru/dotnet/desktop/winforms/windows-forms-overview?view=netframeworkdesktop-4.8

17) Мішура К.А. Виділення складного тренду сигналів на основі масштабно-часовихперетворень, 2020.– 129 с.

18) I. Baklan, A. Oliynyk, I. Mukha, K. Lishchuk, O. Gavrilenko, S. Reutska, A. Tsitsyliuk, Y. OliynykECG signalprocessingbasedonlinguisticchainfuzzyset//ComputationalLinguisticsandIntelligentSystems (CoLins2021), 2021. – 11с.

ДОДАТОКА. ПРОГРАМНИЙКОД

```
publicclassPicksAnalizator
    {
constchar delimiter = '\t';

DataViewSchemamodelSchema;
ITransformertrainedModel;
MLContextmlContext = newMLContext(seed: 4);

publicPicksAnalizator(string disease)
        {
trainedModel = mlContext.Model.Load($"C:\\{disease}_Picks.zip", outmodelSchema);
        }

publicKeyValuePair<string, List<double>>Analize(List<ECGItem>detectedIntervals, doubleRRFrequency,
doubleAvarageRRFrequencyTweak)
        {
            List<KeyValuePair<int, double>>problemsPeaks = new List<KeyValuePair<int, double>>();

for (inti = 0; i<detectedIntervals.Count; i++)
            {
ECGSegment test = newECGSegment()
                {
PQDuration = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].PQ.DurationTime, 2) * 100)),
QSDuration = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].QS.DurationTime, 2) * 100)),
STDuration = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].ST.DurationTime, 2) * 100)),
RRFrequency = Convert.ToSingle(Math.Floor(Math.Round(RRFrequency, 2) * 100)),
AvarageRRFrequencyTweak = Convert.ToSingle(Math.Floor(Math.Round(AvarageRRFrequencyTweak, 2) * 100)),
PTime = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].P.Time, 2) * 100)),
Pvalue = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].P.PicValue, 2) * 100)),
QTime = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].Q.Time, 2) * 100)),
Qvalue = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].Q.PicValue, 2) * 100)),
RTime = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].R.Time, 2) * 100)),
Rvalue = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].R.PicValue, 2) * 100)),
STime = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].S.Time, 2) * 100)),
Svalue = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].S.PicValue, 2) * 100)),
TTime = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].T.Time, 2) * 100)),
Tvalue = Convert.ToSingle(Math.Floor(Math.Round(detectedIntervals[i].T.PicValue, 2) * 100)),
                };
varpredEngine = mlContext.Model.CreatePredictionEngine<ECGSegment, ECGPeakPrediction>(trainedModel);
varresultprediction = predEngine.Predict(test);

if (resultprediction.Score> 0.6) problemsPeaks.Add(newKeyValuePair<int, double>(i,
resultprediction.Score));
            }
problemsPeaks.Sort((pair1, pair2) =>pair2.Value.CompareTo(pair1.Value));

stringresultString;
            List<double>resultInterval = null;

if (problemsPeaks.Count == 0) resultString = "Everything is good";
else
            {
resultString = $"Worst interval was with {Math.Round(problemsPeaks[0].Value, 2)}% ilness. \n\n" +
```

```
$"In 90% confidence - {problemsPeaks.Where(x =>x.Value>= 0.9).Count()} intervals, \n" +
$"In 80% confidence - {problemsPeaks.Where(x =>x.Value< 0.9 &&x.Value>= 0.8).Count()} intervals, \n" +
$"In 70% confidence - {problemsPeaks.Where(x =>x.Value< 0.8 &&x.Value>= 0.7).Count()} intervals, \n" +
$"In 60% confidence - {problemsPeaks.Where(x =>x.Value< 0.7 &&x.Value>= 0.6).Count()} intervals. \n";
resultInterval = detectedIntervals[problemsPeaks[0].Key].FullInterval;
            }

returnnewKeyValuePair<string, List<double>>(resultString, resultInterval);
}


 publicclassIntervalsAnalizator
     {
 constchar delimiter = '\t';

 ITransformertrainedModelPQ;
 ITransformertrainedModelQS;
 ITransformertrainedModelST;
 DataViewSchemamodelSchemaPQ;
 DataViewSchemamodelSchemaQS;
 DataViewSchemamodelSchemaST;
 MLContextmlContextPQ = newMLContext(seed: 1);
 MLContextmlContextQS = newMLContext(seed: 2);
 MLContextmlContextST = newMLContext(seed: 3);

 doublePQCoef;
 doubleQSCoef;
 doubleSTCoef;

 publicIntervalsAnalizator(string disease)
         {
 trainedModelPQ = mlContextPQ.Model.Load($"C:\\PQ_{disease}.zip", outmodelSchemaPQ);
 trainedModelQS = mlContextQS.Model.Load($"C:\\QS_{disease}.zip", outmodelSchemaQS);
 trainedModelST = mlContextST.Model.Load($"C:\\ST_{disease}.zip", outmodelSchemaST);

 var Lines_1 = File.ReadAllLines($"C:\\Words_ROC_values.tsv").ToList();
 PQCoef = Math.Round(Convert.ToDouble(Lines_1.Where(x =>x.Split(delimiter)[0] ==
 $"{disease}"&&x.Split(delimiter)[1] == "PQ").First().Split(delimiter)[2]), 3);
 QSCoef = Math.Round(Convert.ToDouble(Lines_1.Where(x =>x.Split(delimiter)[0] ==
 $"{disease}"&&x.Split(delimiter)[1] == "QS").First().Split(delimiter)[2]), 3);
 STCoef = Math.Round(Convert.ToDouble(Lines_1.Where(x =>x.Split(delimiter)[0] ==
 $"{disease}"&&x.Split(delimiter)[1] == "ST").First().Split(delimiter)[2]), 3);
 varsumCoef = PQCoef + QSCoef + STCoef;
 PQCoef = PQCoef / sumCoef;
 QSCoef = QSCoef / sumCoef;
 STCoef = STCoef / sumCoef;
         }

 publicKeyValuePair<string, List<double>>Analize(List<ECGItem>detectedIntervals)
         {
             List<KeyValuePair<int, double>>problemsIntervals = new List<KeyValuePair<int,
 double>>();

 for (inti = 0; i<detectedIntervals.Count; i++)
             {
 ECGWordtestPQ = newECGWord() { Text = detectedIntervals[i].PQ.Word };
 varpredEnginePQ = mlContextPQ.Model.CreatePredictionEngine<ECGWord, ECGPrediction>(trainedModelPQ);
 varresultpredictionPQ = predEnginePQ.Predict(testPQ);

 ECGWordtestQS = newECGWord() { Text = detectedIntervals[i].QS.Word };
```

```
varpredEngineQS = mlContextQS.Model.CreatePredictionEngine<ECGWord, ECGPrediction>(trainedModelQS);
varresultpredictionQS = predEngineQS.Predict(testQS);

ECGWordtestST = newECGWord() { Text = detectedIntervals[i].ST.Word };
varpredEngineST = mlContextST.Model.CreatePredictionEngine<ECGWord, ECGPrediction>(trainedModelST);
varresultpredictionST = predEngineST.Predict(testST);

var prediction = resultpredictionPQ.Probability * PQCoef + resultpredictionQS.Probability * QSCoef +
resultpredictionST.Probability * STCoef;

if (prediction > 0.6) problemsIntervals.Add(newKeyValuePair<int, double>(i, prediction));
            }

problemsIntervals.Sort((pair1, pair2) =>pair2.Value.CompareTo(pair1.Value));

stringresultString;
            List<double>resultInterval = null;

if (problemsIntervals.Count == 0) resultString = "Everything is good";
else
            {
resultString = $"Worst interval was with {Math.Round(problemsIntervals[0].Value, 2)}% ilness. \n\n"
+
$"In 90% confidence - {problemsIntervals.Where(x =>x.Value>= 0.9).Count()} intervals, \n" +
$"In 80% confidence - {problemsIntervals.Where(x =>x.Value< 0.9 &&x.Value>= 0.8).Count()} intervals,
\n" +
$"In 70% confidence - {problemsIntervals.Where(x =>x.Value< 0.8 &&x.Value>= 0.7).Count()} intervals,
\n" +
$"In 60% confidence - {problemsIntervals.Where(x =>x.Value< 0.7 &&x.Value>= 0.6).Count()} intervals.
\n";
resultInterval = detectedIntervals[problemsIntervals[0].Key].FullInterval;
            }

returnnewKeyValuePair<string, List<double>>(resultString, resultInterval);
        }


publicstaticclassFileManager
    {
staticstringFIlES_FOLDER_PATH = "C:\\";
staticstring DELIMETR = "\t";

publicstaticvoidMixDatasets(string name1, string name2)
        {
var Lines_1 = File.ReadAllLines($"{FIlES_FOLDER_PATH}{name1}.tsv").ToList();
var Lines_2 = File.ReadAllLines($"{FIlES_FOLDER_PATH}{name2}.tsv").ToList();

var header = Lines_1[0];
            Lines_2.RemoveAt(0);
            Lines_1.RemoveAt(0);

            List<string>LinesAll = new List<string>(Lines_1);
LinesAll.AddRange(Lines_2);

            Random random = newRandom();
LinesAll = LinesAll.OrderBy(x =>random.Next()).ToList();

var result = newStringBuilder();
result.AppendLine(header);
foreach (var item inLinesAll)
```

```
            {
if (item.Length> 5) result.AppendLine(item);
            }

File.WriteAllText($"{FIlES_FOLDER_PATH}{name2}_withGood.tsv", result.ToString());
        }

publicstaticvoidSaveToIntervalsDataset(string name, intisWrong, string disease,
List<ECGItem>detectedIntervals)
        {
varcsvWordsPQ = newStringBuilder();
varcsvWordsQS = newStringBuilder();
varcsvWordsST = newStringBuilder();

csvWordsPQ.AppendLine($"Label{DELIMETR}Text");
foreach (var item indetectedIntervals)
            {
if (item.PQ.Word.Length> 0)
                {
varnewPQParamsLine = $"{isWrong}{DELIMETR}{item.PQ.Word}";
csvWordsPQ.AppendLine(newPQParamsLine);
                }
if (item.QS.Word.Length> 0)
                {
varnewQSParamsLine = $"{isWrong}{DELIMETR}{item.QS.Word}";
csvWordsQS.AppendLine(newQSParamsLine);
                }
if (item.ST.Word.Length> 0)
                {
varnewSTParamsLine = $"{isWrong}{DELIMETR}{item.ST.Word}";
csvWordsST.AppendLine(newSTParamsLine);
                }
            }

            File.WriteAllText($"{FIlES_FOLDER_PATH}Words_{name}_PQ_{disease}.tsv",
csvWordsPQ.ToString());
            File.WriteAllText($"{FIlES_FOLDER_PATH}Words_{name}_QS_{disease}.tsv",
csvWordsQS.ToString());
            File.WriteAllText($"{FIlES_FOLDER_PATH}Words_{name}_ST_{disease}.tsv",
csvWordsST.ToString());
        }

publicstaticvoidSaveToPicksDataset(string name, intisWrong, string disease,
List<ECGItem>detectedIntervals, doubleRRFrequency, doubleAvarageRRFrequencyTweak)
        {
varcsvParams = newStringBuilder();

csvParams.AppendLine($"Label" +
$"{DELIMETR}PTime{DELIMETR}Pvalue{DELIMETR}" +
$"QTime{DELIMETR}Qvalue{DELIMETR}" +
$"RTime{DELIMETR}Rvalue{DELIMETR}" +
$"STime{DELIMETR}Svalue{DELIMETR}" +
$"TTime{DELIMETR}Tvalue{DELIMETR}" +
$"PQDuration{DELIMETR}QSDuration{DELIMETR}STDuration" +
$"{DELIMETR}RRFrequency{DELIMETR}AvarageRRFrequencyTweak");

foreach (var item indetectedIntervals)
            {
varnewcsvParamsLine = $"{isWrong}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.P.Time, 2) * 100)}" +
```

```
$"{DELIMETR}{Math.Floor(Math.Round(item.P.PicValue, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.Q.Time, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.Q.PicValue, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.R.Time, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.R.PicValue, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.S.Time, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.S.PicValue, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.T.Time, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.T.PicValue, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.PQ.DurationTime, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.QS.DurationTime, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(item.ST.DurationTime, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(RRFrequency, 2) * 100)}" +
$"{DELIMETR}{Math.Floor(Math.Round(AvarageRRFrequencyTweak, 2) * 100)}";
csvParams.AppendLine(newcsvParamsLine);
            }

            File.WriteAllText($"{FIlES_FOLDER_PATH}Pics_{name}_{disease}.tsv",
csvParams.ToString());
        }


publicclassECGConverter
    {
privateconstint ECG_DOWN_COEF = 950;
        List<List<double>>hardToDetectIntervals;

public List<ECGItem>detectedIntervals;
publicdoubleRRFrequency;
publicdoubleAvarageRRFrequencyTweak;

publicECGConverter(List<double>FullECG, List<double>RRTime, List<int>RIds, doubleAVGIntervalTime)
        {
detectedIntervals = new List<ECGItem>();
hardToDetectIntervals = new List<List<double>>();

RRFrequency = 0;
AvarageRRFrequencyTweak = 0;

ConvertToECGCollection(FullECG, RRTime, RIds, AVGIntervalTime);
        }

publicWaveletTransformationWaveletTransformation
        {
get =>default;
set
            {
            }
        }

publicvoidConvertToECGCollection(List<double>FullECG, List<double>RRTime, List<int>RIds,
doubleAVGIntervalTime)
        {

for (inti = 1; i<RIds.Count - 1; i++)
            {
intprevId = RIds[i - 1];
intcurId = RIds[i];
intnextId = RIds[i + 1];
```

```
intSId = 0, QId = 0, PId = 0, RId = 0, TId = 0;
doubleSVal = 0, QVal = 0, PVal = 0, RVal = 0, TVal = 0;

doubleleftTime = RRTime[i - 1];
doublerightTime = RRTime[i];

boolleftIsTooSmall = (leftTime< 0.75 * AVGIntervalTime);
boolleftIsTooBig = (leftTime> 1.25 * AVGIntervalTime);

if (leftIsTooBig || leftIsTooSmall)
                {
hardToDetectIntervals.Add(FullECG.GetRange(prevId, curId - prevId));
if (leftIsTooSmall) continue;

leftTime = AVGIntervalTime;
                }
boolrightIsTooSmall = (rightTime< 0.75 * AVGIntervalTime);
boolrightIsTooBig = (rightTime> 1.25 * AVGIntervalTime);

if (rightIsTooSmall || rightIsTooBig)
                {
if (rightIsTooSmall) continue;
rightTime = AVGIntervalTime;
                }

RRFrequency += leftTime;
AvarageRRFrequencyTweak += Math.Abs(leftTime - AVGIntervalTime) * 10;

doubleRSign = FullECG[curId];
ECGItem item = newECGItem();
item.P = newECGPoint();
item.Q = newECGPoint();
item.R = newECGPoint();
item.S = newECGPoint();
item.T = newECGPoint();


intsearchId = SId;
intupCount = 0;

if (RSign<FullECG[curId + 2] || RSign<FullECG[curId - 2])
                {
}

if (RId == 0)
                {
RId = curId;
RVal = FullECG[curId];
                }

if (QId == 0)
                {
#regionFindQ

QId = RId;
QVal = FullECG[RId];
searchId = RId;
upCount = 0;
```

```
while (upCount< 2)
                    {
searchId--;

if (FullECG[searchId] >QVal) upCount++;
else
                        {
QVal = FullECG[searchId];
upCount = 0;
                        }
                    }
QId = searchId + 2;

#endregion
                }
if (SId == 0)
                {
#regionFindS

SId = curId;
SVal = FullECG[curId];
searchId = curId;
upCount = 0;

while (upCount< 2)
                    {
searchId++;

if (FullECG[searchId] >SVal) upCount++;
else
                        {
SVal = FullECG[searchId];
upCount = 0;
                        }
                    }
SId = searchId - 2;
#endregion
                }
if (TId == 0)
                {
#regionFindT

TId = SId;
TVal = FullECG[SId];
doubleTValPrev = FullECG[SId];
searchId = SId;
upCount = 0;

while (upCount< 5)
                    {
searchId++;

if (FullECG[searchId] <TValPrev)
                        {
upCount++;
                        }
else
                        {
TVal = FullECG[searchId];
upCount = 0;
```

```
                        }

TValPrev = FullECG[searchId];
                    }
TId = searchId - 5;
#endregion
                }
if (PId == 0)
                {
#regionFindP

PId = QId;
PVal = FullECG[QId];
doublePValPrev = FullECG[QId];
searchId = QId;
upCount = 0;

while (upCount< 3)
                    {
searchId--;

if (FullECG[searchId] <PValPrev)
                        {
upCount++;
                        }
else
                        {
PVal = FullECG[searchId];
upCount = 0;
                        }

PValPrev = FullECG[searchId];
                    }
PId = searchId - 3;
#endregion
                }
item.P.PicValue = 0;
item.Q.PicValue = 0;
item.R.PicValue = 0;
item.S.PicValue = 0;
item.T.PicValue = 0;

item.P.Time = 0;
item.Q.Time = 0;
item.R.Time = 0;
item.S.Time = 0;
item.T.Time = 0;

item.P.PicValue = PVal - ECG_DOWN_COEF;
item.Q.PicValue = QVal - ECG_DOWN_COEF;
item.R.PicValue = RVal - ECG_DOWN_COEF;
item.S.PicValue = SVal - ECG_DOWN_COEF;
item.T.PicValue = TVal - ECG_DOWN_COEF;

item.P.Time = (RId - PId) * 0.002;
item.Q.Time = (RId - QId) * 0.002;
item.R.Time = 0;
item.S.Time = (SId - RId) * 0.002;
item.T.Time = (TId - RId) * 0.002;
```

```
item.PQ = newECGInterval();
item.QS = newECGInterval();
                item.ST = newECGInterval();

item.FullInterval = FullECG.GetRange(prevId, nextId - prevId);

intPQBegin = PId - Convert.ToInt32((QId - PId) / 2);
doublePQTime = (QId - PQBegin) * 0.2;
stringPQWord = "";
                StringBuilder PQSb = newStringBuilder();

for (int s = PQBegin; s <= QId; s += 2)
                {
int diff = Convert.ToInt32(FullECG[s]) - Convert.ToInt32(FullECG[s + 1]);

PQSb.Append(MakeAlfaFromDiff(diff));
                }
if (PQSb.Length> 3)
                {
for (int s = 0; s <= PQSb.Length - 3; s++)
                    {
PQWord += PQSb[s].ToString() + PQSb[s + 1].ToString() + PQSb[s + 2].ToString() + " ";

if (PQSb.Length> 4 && (s + 3 <PQSb.Length))
PQWord += PQSb[s].ToString() + PQSb[s + 1].ToString() + PQSb[s + 2].ToString() + PQSb[s +
3].ToString() + " ";
if (PQSb.Length> 4 && (s + 4 <PQSb.Length))
PQWord += PQSb[s].ToString() + PQSb[s + 1].ToString() + PQSb[s + 2].ToString() + PQSb[s +
3].ToString() + PQSb[s + 4].ToString() + " ";
                    }
                }
PQWord += PQSb.ToString();

intQSEnd = SId + Convert.ToInt32((SId - QId) / 3);
intQSBegin = QId - Convert.ToInt32((SId - QId) / 3);
doubleQSTime = (QSEnd - QSBegin) * 0.2;
stringQSWord = "";
                StringBuilder QSSb = newStringBuilder();

for (int s = QSBegin; s <= QSEnd; s += 2)
                {
int diff = Convert.ToInt32(FullECG[s]) - Convert.ToInt32(FullECG[s + 1]);

QSSb.Append(MakeAlfaFromDiff(diff));
                }
if (QSSb.Length> 3)
                {
for (int s = 0; s <= QSSb.Length - 3; s++)
                    {
QSWord += QSSb[s].ToString() + QSSb[s + 1].ToString() + QSSb[s + 2].ToString() + " ";

if (QSSb.Length> 4 && (s + 3 <QSSb.Length))
QSWord += QSSb[s].ToString() + QSSb[s + 1].ToString() + QSSb[s + 2].ToString() + QSSb[s +
3].ToString() + " ";
if (QSSb.Length> 4 && (s + 4 <QSSb.Length))
QSWord += QSSb[s].ToString() + QSSb[s + 1].ToString() + QSSb[s + 2].ToString() + QSSb[s +
3].ToString() + QSSb[s + 4].ToString() + " ";
                    }
                }
QSWord += QSSb.ToString();
```

```
intSTBegin = QSEnd;
intSTEnd = TId + Convert.ToInt32((TId - STBegin) / 2);
doubleSTTime = (STEnd - STBegin) * 0.2;
stringSTWord = "";
                StringBuilder STSb = newStringBuilder();

for (int s = STBegin; s <= STEnd; s += 2)
                {
int diff = Convert.ToInt32(FullECG[s]) - Convert.ToInt32(FullECG[s + 1]);

STSb.Append(MakeAlfaFromDiff(diff));
                }
if (STSb.Length> 3)
                {
for (int s = 0; s <= STSb.Length - 3; s++)
{
STWord += STSb[s].ToString() + STSb[s + 1].ToString() + STSb[s + 2].ToString() + " ";

if (STSb.Length> 4 && (s + 3 <STSb.Length))
STWord += STSb[s].ToString() + STSb[s + 1].ToString() + STSb[s + 2].ToString() + STSb[s +
3].ToString() + " ";
if (STSb.Length> 4 && (s + 4 <STSb.Length))
STWord += STSb[s].ToString() + STSb[s + 1].ToString() + STSb[s + 2].ToString() + STSb[s +
3].ToString() + STSb[s + 4].ToString() + " ";
}
}
STWord += STSb.ToString();

item.PQ.DurationTime = 0;
item.QS.DurationTime = 0;
item.ST.DurationTime = 0;

item.PQ.DurationTime = PQTime;
item.PQ.Word = PQWord;

item.QS.DurationTime = QSTime;
item.QS.Word = QSWord;

item.ST.DurationTime = STTime;
item.ST.Word = STWord;

detectedIntervals.Add(item);
            }

AvarageRRFrequencyTweak = AvarageRRFrequencyTweak / detectedIntervals.Count;
RRFrequency = 10 / (RRFrequency / detectedIntervals.Count);
        }
publicstringMakeAlfaFromDiff(int diff)
        {
string res = "";

if (diff == 0) res = "A";
if (diff > 0)
            {
if(diff < 3) res = "B";
elseif (diff < 6) res = "C";
elseif (diff < 10) res = "D";
elseif (diff < 16) res = "E";
```

```
elseif (diff < 26) res = "F";
elseif (diff < 36) res = "G";
elseif (diff < 46) res = "H";
elseif (diff < 56) res = "I";
elseif (diff < 86) res = "J";
elseif (diff < 110) res = "K";
elseif (diff < 150) res = "L";
elseif (diff < 200) res = "M";
else res = "N";
            }
if (diff < 0)
            {
if (diff > -3) res = "Z";
elseif (diff > -6) res = "Y";
elseif (diff > -9) res = "X";
elseif (diff > -16) res = "W";
elseif (diff > -26) res = "V";
elseif (diff > -36) res = "U";
elseif (diff > -56) res = "T";
elseif (diff > -86) res = "S";
elseif (diff > -110) res = "R";
elseif (diff > -150) res = "Q";
elseif (diff > -200) res = "P";
else res = "O";
            }

return res;
        }

public List<KeyValuePair<string, List<double>>>Analize(string disease)
{
PicksAnalizatorpicksAnalizator = newPicksAnalizator(disease);
IntervalsAnalizatorintervalsAnalizator = newIntervalsAnalizator(disease);

            List<KeyValuePair<string, List<double>>> returns = new List<KeyValuePair<string,
List<double>>>()
            {
intervalsAnalizator.Analize(detectedIntervals),
picksAnalizator.Analize(detectedIntervals, RRFrequency, AvarageRRFrequencyTweak)
};

return returns;
        }


classProgram
    {
conststring delimiter = "\t";

staticvoid Main(string[] args)
        {
            StringBuilder sb = newStringBuilder();
sb.AppendLine($"Disease{delimiter}Interval{delimiter}ROC");

sb.AppendLine(TrainECG("gypotermia"));
sb.AppendLine(TrainECG("infarktMiocarda"));
sb.AppendLine(TrainECG("synoatrialBlock"));
sb.AppendLine(TrainECG("trepetPeredserd"));
        }
```

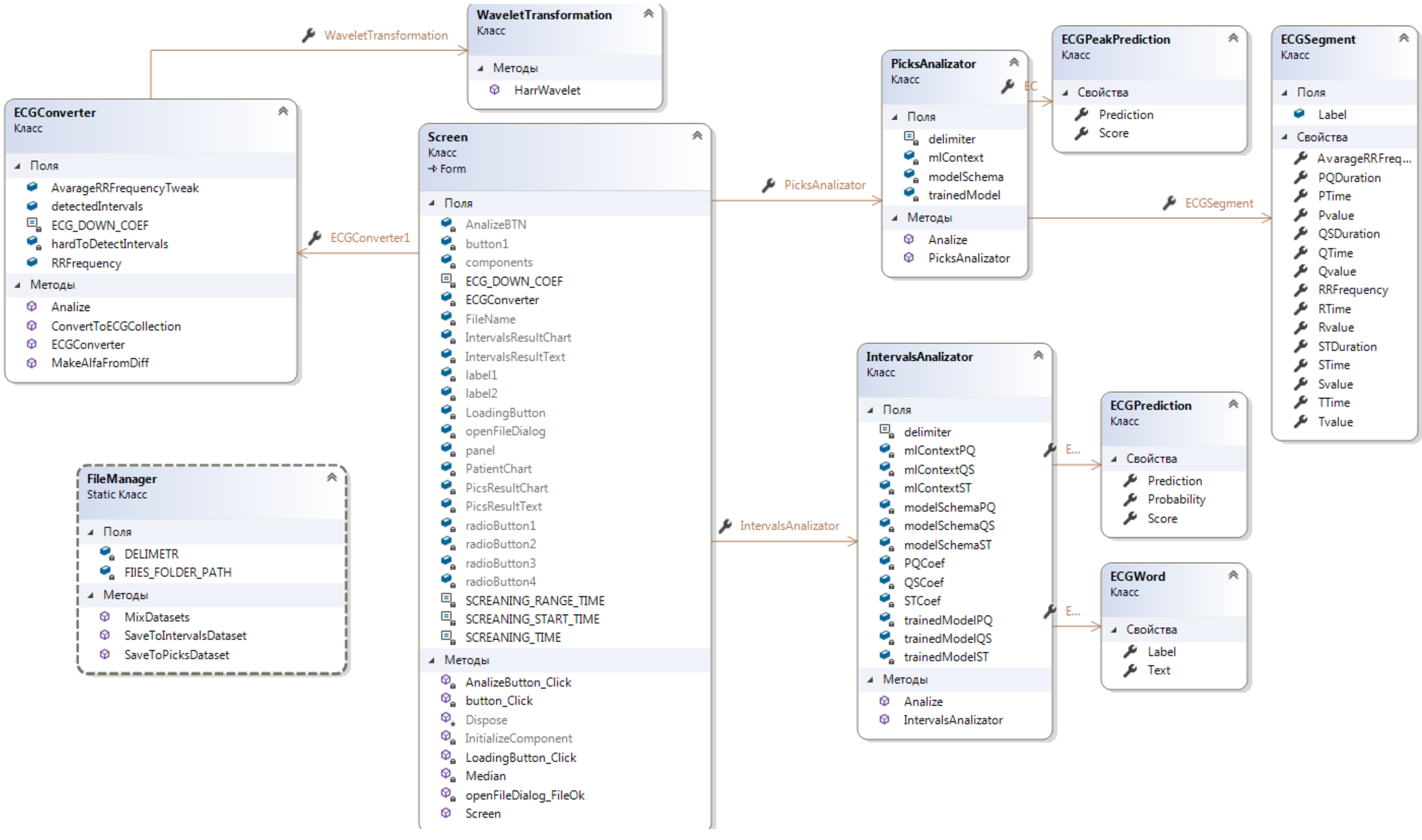
```
publicstaticstringTrainECG(stringdesease)
        {
varmlContext = newMLContext(seed: 0);

IDataViewdataView =
mlContext.Data.LoadFromTextFile<ECGSegment>($"C:\\Pics_ECGDataset_{desease}_withGood.tsv",
hasHeader: true);

TrainTestDatatrainTestSplit = mlContext.Data.TrainTestSplit(dataView, testFraction: 0.2);
IDataViewtrainingData = trainTestSplit.TrainSet;
IDataViewtestData = trainTestSplit.TestSet;


vardataProcessPipeline = mlContext.Transforms.Concatenate("Features",
nameof(ECGSegment.PTime),
nameof(ECGSegment.Pvalue),
nameof(ECGSegment.QTime),
nameof(ECGSegment.Qvalue),
nameof(ECGSegment.Rvalue),
nameof(ECGSegment.STime),
nameof(ECGSegment.Svalue),
nameof(ECGSegment.TTime),
nameof(ECGSegment.Tvalue),
nameof(ECGSegment.PQDuration),
nameof(ECGSegment.QSDuration),
nameof(ECGSegment.STDuration),
nameof(ECGSegment.RRFrequency),
nameof(ECGSegment.AvarageRRFrequencyTweak)
                                                                    )
.AppendCacheCheckpoint(mlContext);

var trainer = mlContext.Regression.Trainers.FastForest(labelColumnName: "Label", featureColumnName:
"Features", numberOfLeaves: 7, numberOfTrees: 14, minimumExampleCountPerLeaf: 20);
vartrainingPipeline = dataProcessPipeline.Append(trainer);
ITransformertrainedModel = trainingPipeline.Fit(dataView);

var predictions = trainedModel.Transform(testData);
var metrics = mlContext.Regression.Evaluate(data: predictions, labelColumnName: "Label",
scoreColumnName: "Score");

mlContext.Model.Save(trainedModel, trainingData.Schema, $"C:\\{desease}_Picks.zip");


Console.WriteLine($"\n\n\n================== ML FOR :{desease} ==================\n");
Console.WriteLine($"RSquared: {metrics.RSquared}\n");
Console.WriteLine($"LossFunction: {metrics.LossFunction}");
Console.WriteLine($"MeanAbsoluteError: {metrics.MeanAbsoluteError}");
Console.WriteLine($"MeanSquaredError: {metrics.MeanSquaredError}");
Console.WriteLine($"RootMeanSquaredError: {metrics.RootMeanSquaredError}");

return$"{desease}{delimiter}{metrics.MeanSquaredError}";
        }
    }


classProgram
    {
conststring delimiter = "\t";

staticvoid Main(string[] args)
```

```
        {
            StringBuilder sb = newStringBuilder();
sb.AppendLine($"Disease{delimiter}Interval{delimiter}ROC");

sb.AppendLine(TrainECG("PQ_gypotermia"));
sb.AppendLine(TrainECG("QS_gypotermia"));
sb.AppendLine(TrainECG("ST_gypotermia"));

sb.AppendLine(TrainECG("PQ_infarktMiocarda"));
sb.AppendLine(TrainECG("QS_infarktMiocarda"));
sb.AppendLine(TrainECG("ST_infarktMiocarda"));

sb.AppendLine(TrainECG("PQ_synoatrialBlock"));
sb.AppendLine(TrainECG("QS_synoatrialBlock"));
sb.AppendLine(TrainECG("ST_synoatrialBlock"));

sb.AppendLine(TrainECG("PQ_trepetPeredserd"));
sb.AppendLine(TrainECG("QS_trepetPeredserd"));
sb.AppendLine(TrainECG("ST_trepetPeredserd"));

File.WriteAllText($"C:\\Words_ROC_values.tsv", sb.ToString());
        }

staticstringTrainECG(stringdesease)
        {
varmlContext = newMLContext(seed: 1);

IDataViewdataView =
mlContext.Data.LoadFromTextFile<ECGWord>($"C:\\Words_ECGDataset_{desease}_withGood.tsv", hasHeader:
true);

TrainTestDatatrainTestSplit = mlContext.Data.TrainTestSplit(dataView, testFraction: 0.2);
IDataViewtrainingData = trainTestSplit.TrainSet;
IDataViewtestData = trainTestSplit.TestSet;

vardataProcessPipeline = mlContext.Transforms.Text.FeaturizeText(outputColumnName: "Features",
inputColumnName: nameof(ECGWord.Text));

var trainer = mlContext.BinaryClassification.Trainers.SdcaLogisticRegression(labelColumnName:
"Label", featureColumnName: "Features", l2Regularization: 0.000006f, l1Regularization: 0.000013f,
maximumNumberOfIterations: 10000);
vartrainingPipeline = dataProcessPipeline.Append(trainer);

ITransformertrainedModel = trainingPipeline.Fit(trainingData);

var predictions = trainedModel.Transform(testData);
var metrics = mlContext.BinaryClassification.Evaluate(data: predictions, labelColumnName: "Label",
scoreColumnName: "Score");

mlContext.Model.Save(trainedModel, trainingData.Schema, $"C:\\{desease}.zip");

Console.WriteLine($"\n\n\n================== ML FOR :{desease} ==================\n");
Console.WriteLine($"ROC AUC: {metrics.AreaUnderRocCurve}\n");
Console.WriteLine($"ACURACY: {metrics.Accuracy}");
Console.WriteLine($"NegativePrecision: {metrics.NegativePrecision}");
Console.WriteLine($"NegativeRecall: {metrics.NegativeRecall}");
Console.WriteLine($"PositivePrecision: {metrics.PositivePrecision}");
Console.WriteLine($"PositiveRecall: {metrics.PositiveRecall}");
```

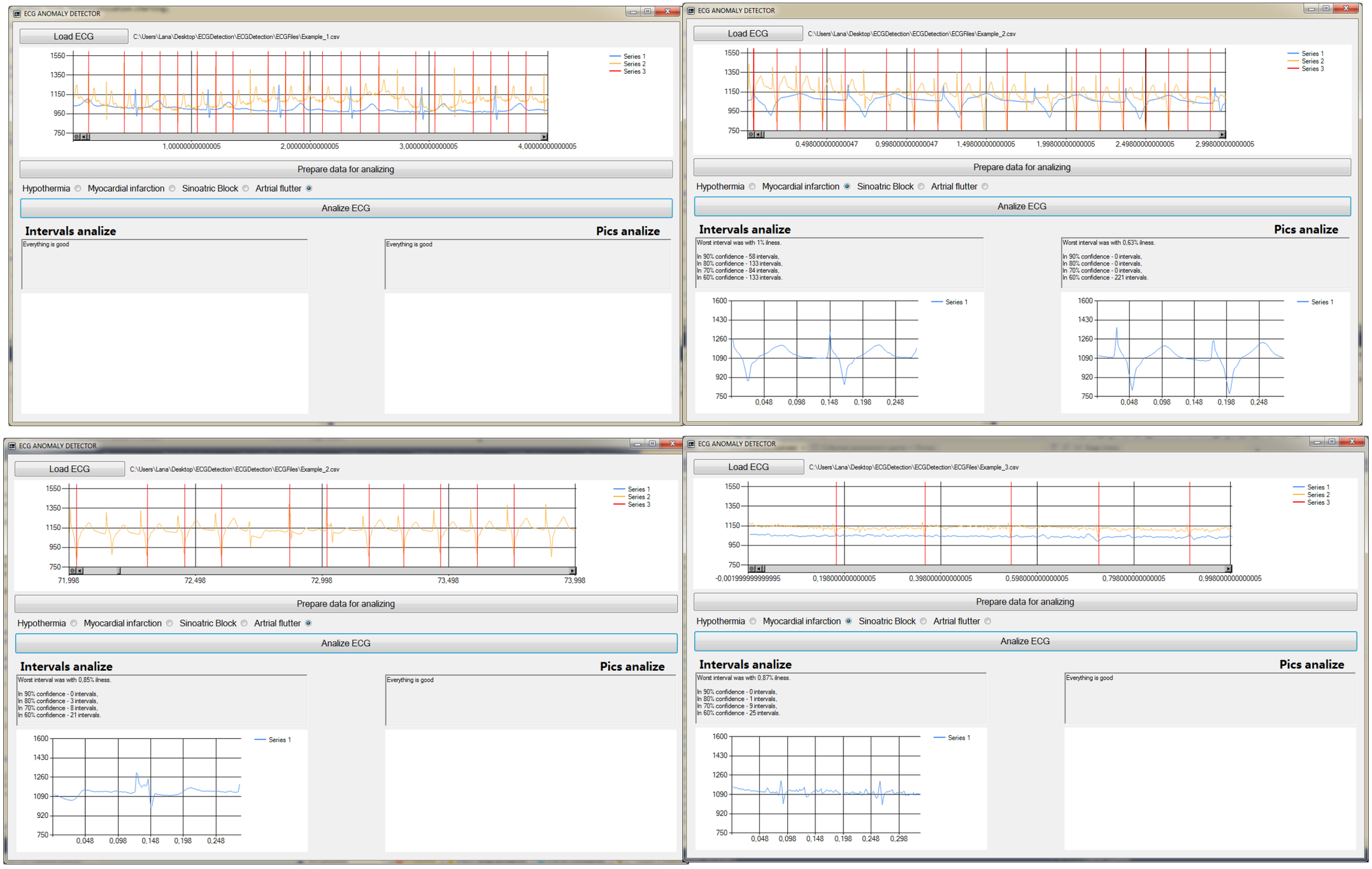
```
return$"{desease.Split("_")[1]}{delimiter}{desease.Split("_")[0]}{delimiter}{metrics.AreaUnderRocCur
ve}";
}
    }
```

ДОДАТОК Б. ГРАФІЧНИЙ МАТЕРІАЛ

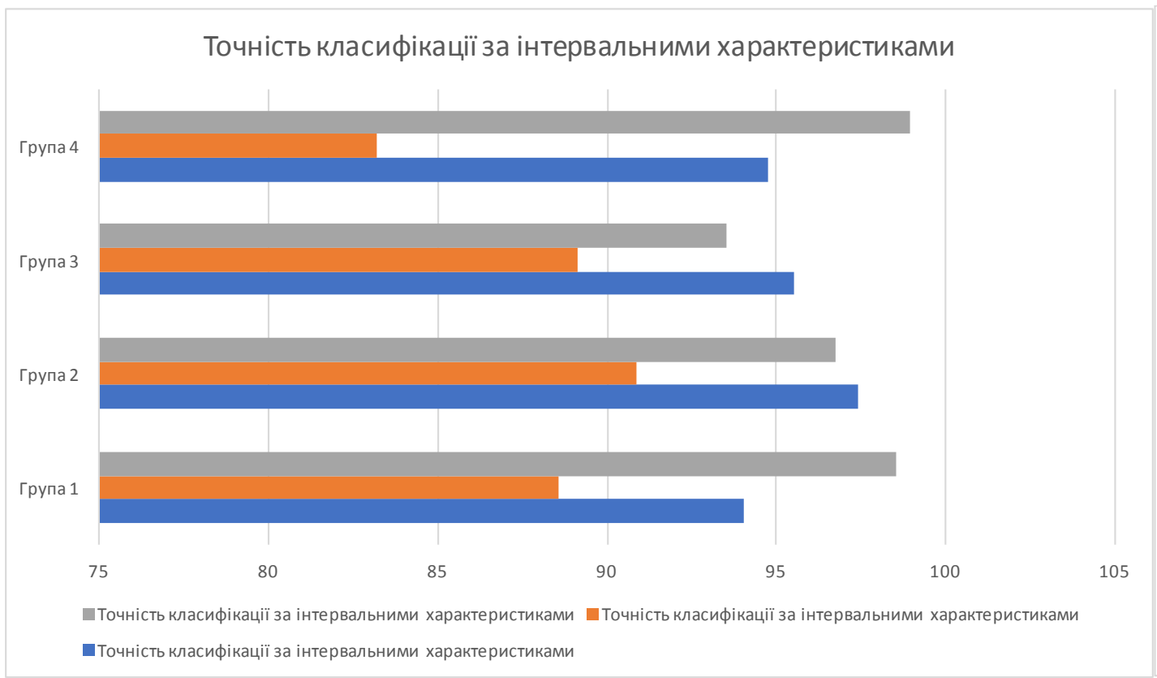
Схема структурна класів

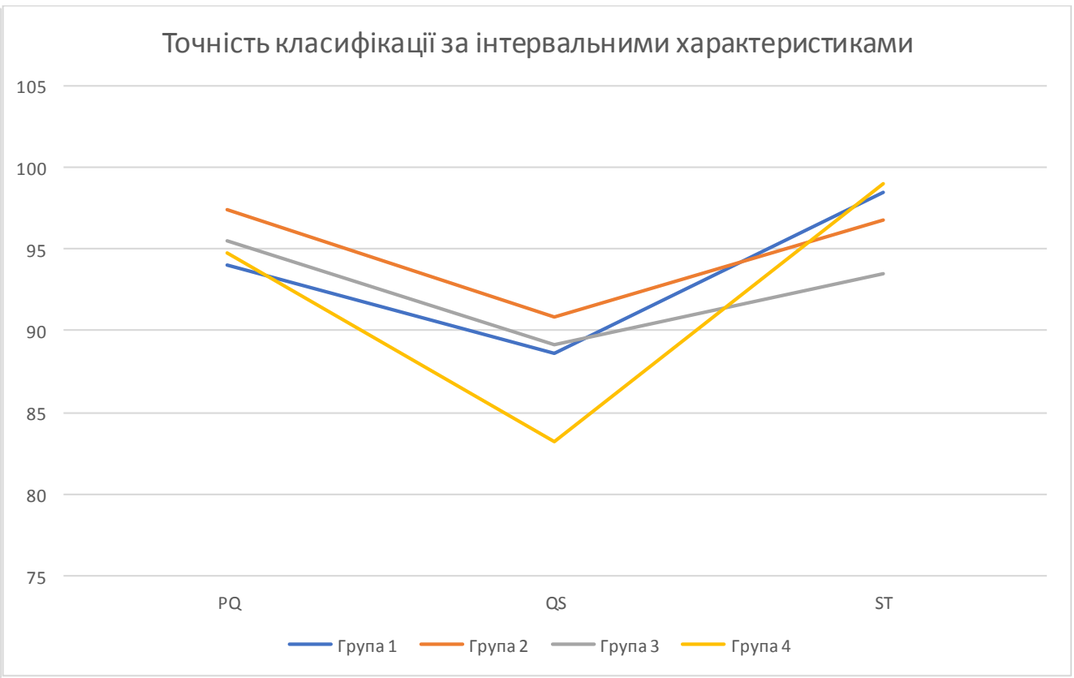


Демонстраційний плакат



Графіки ефективності алгоритму





UNICHECK

Ім'я користувача:
Попенко Володимир Дмитрович

Дата перевірки:
16.05.2021 21:58:41 EEST

Дата звіту:
16.05.2021 22:37:18 EEST

ID перевірки:
1007877591

Тип перевірки:
Doc vs Internet + Library

ID користувача:
77149

Назва документа: **Reutska_magistr_ip91mn_2**

Кількість сторінок: **80** Кількість слів: **10987** Кількість символів: **85637** Розмір файлу: **2.44 MB** ID файлу: **1007972222**

6.84%
Схожість

Найбільша схожість: **1.03%** з джерелом з Бібліотеки (ID файлу: **1005727512**)

2.68% Джерела з Інтернету	44	Сторінка 82
6.18% Джерела з Бібліотеки	289	Сторінка 83

0.13% Цитат

Цитати	2	Сторінка 84

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	6