

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Фізико-математичний факультет**

Кафедра математичної фізики та диференціальних рівнянь

«На правах рукопису»

УДК 519.6, 368.2

До захисту допущено:

Завідувач кафедри

_____ Володимир ГОРБАЧУК

«01» грудня 2021 р.

**Магістерська дисертація
на здобуття ступеня магістра
за освітньо-професійною програмою «Страхова та фінансова математика»
зі спеціальності 111 «Математика»
на тему: «Математичні та програмні засоби системи автострахування»**

Виконав:

студент II курсу магістратури, групи ОМ-02мп

Рясик Ігор Сергійович _____

Керівник:

доктор фізико-математичних наук, професор,

Швець Олександр Юрійович _____

Рецензент:

провідний науковий співробітник, доктор фізико-математичних наук, професор,

Мазко Олексій Григорович

Інститут математики НАН України _____

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

Київ – 2021 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Фізико-математичний факультет

Кафедра математичної фізики та диференціальних рівнянь

Рівень вищої освіти – другий (магістерський)

Спеціальність – 111 «Математика»

Освітньо-професійна програма «Страхова та фінансова математика»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Володимир ГОРБАЧУК

«20» вересня 2021 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Рясику Ігорю Сергійовичу

1. Тема дисертації: «Математичні та програмні засоби системи автострахування», науковий керівник дисертації Швець Олександр Юрійович, доктор фізико-математичних наук, професор, затверджені наказом по університету від «26» березня 2021 р. № 901-с
2. Термін подання студентом дисертації: «01» грудня 2021 року
3. Об'єкт дослідження: страховий поліс у автострахуванні
4. Предмет дослідження: методи розрахунку страхового полісу в автострахуванні
5. Перелік завдань, які потрібно розробити
 - 5.1. ознайомлення з існуючими та популярними скоринговими системами;
 - 5.2. вибір математичних методів розв'язання поставлених задач;
 - 5.3. реалізування математичного та програмного забезпечення скорингової системи автострахування;
 - 5.4. проведення експериментальних випробувань розробленої системи;

5.5. формулювання висновків з отриманих результатів.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу 17 слайдів

7. Дата видачі завдання: «20» вересня 2021 р.

Календарний план

№ з/п	Назва етапів виконання дисертаційної роботи	Термін виконання етапів роботи	Примітка
1	Огляд літератури за тематикою та збір даних	20.09.2021 — 26.09.2021	Виконано
2	Проведення порівняльного аналізу методів	26.09.2021 — 02.10.2021	Виконано
3	Підготовка матеріалів першого розділу роботи	03.10.2021 — 07.10.2021	Виконано
4	Розроблення математичного забезпечення	08.10.2021 — 23.10.2021	Виконано
5	Підготовка матеріалів другого розділу роботи	24.10.2021 — 28.10.2021	Виконано
6	Підготовка матеріалів третього розділу роботи	29.10.2021 — 04.11.2021	Виконано
7	Розроблення програмного забезпечення	05.11.2021 — 19.11.2021	Виконано
8	Підготовка матеріалів четвертого розділу роботи	20.11.2021 — 24.11.2021	Виконано
9	Оформлення пояснювальної записки	25.11.2021 — 30.11.2021	Виконано

Студент _____

Ігор РЯСИК

Керівник роботи _____

Олександр ШВЕЦЬ

РЕФЕРАТ

Магістерську дисертацію виконано на 62 аркушах, вона містить 17 слайдів для проектора, 16 першоджерел та 2 додатка. У роботі наведено 16 рисунків та 4 таблиць.

Метою дисертаційної роботи є мінімізація втрат та ризиків, підвищення якості клієнтської бази для автострахових компаній. У роботі проведено огляд подібних систем та популярних підходів даної області, які використовуються в різних компаніях. На основі сформованих вимог було проаналізовано різні методи реалізації поставленої задачі та з них вибрано такі методи - робота алгоритму логістичної регресії для бінарної класифікації для 3 параметрів, візуалізація результатів роботи методами кривих Ендрюса та матриць розсіювання.

Розроблено десктоп систему, на основі мови Python з застосуванням графічного інтерфейсу Tkinter, також виконано випробування розробленої системи.

Ключові слова: скоринг, Python, телематика, логістична регресія, багатовимірний аналіз, кореляція.

ABSTRACT

The master's dissertation is made on 62 sheets, it contains 17 slides for a projector, 16 primary sources and 2 appendices. The paper presents 16 figures and 4 tables.

The main purpose of the dissertation is to minimize losses and risks, improve the quality of the customer base for auto insurance companies. The paper reviews similar systems and popular approaches in this field, which are used in different companies. On the basis of the formed requirements various methods of realization of the set task were analyzed and from them such methods are chosen - work of algorithm of logistic regression for binary classification for 3 parameters, visualization of results of work by methods of Andrews curves and scattering matrices.

Developed a desktop system based on Python using the graphical interface Tkinter, also tested the developed system.

Key words: scoring, Python, telematics, logistic regression, multidimensional analysis, correlation.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ	7
ВСТУП	8
1. ПОСТАНОВКА ЗАДАЧІ	9
2. ОГЛЯД ІСНУЮЧИХ СИСТЕМ	10
2.1 Розвиток скорингу у страхуванні	10
2.2 Оцифрування систем скорингу	11
2.3 Приклади існуючих рішень	13
2.4 Висновки до розділу	17
3. ОПИС МАТЕМАТИЧНИХ АЛГОРИТМІВ СИСТЕМИ	18
3.1 Методи візуалізації багатовимірних даних	18
3.2 Динамічне навчання. Логістична регресія	21
3.3 Висновки до розділу	29
4. ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМУ ТА СИСТЕМИ	30
4.1 Опис програми та аналізу даних	30
4.2 Перевірка на тестовій групі	36
4.3 Висновки до розділу	40
ВИСНОВКИ	41
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	42
Додаток А Лістинг програми	45
Додаток Б Ілюстративний матеріал	54

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Скоринг - метод класифікації позичальників на групи для оцінки їх кредитоспроможності та рівня кредитного ризику на основі кредитної історії та соціально-демографічних характеристик.

Телематика - Об'єднання телебачення з комп'ютерними пристроями для інтегрування обробки і передавання інформації.

ПЗ - програмне забезпечення.

ВСТУП

На сьогоднішній день страхування є однією з домінуючих галузей фінансів. Одним із найпопулярніших напрямків страхування у світі є автострахування. Беручи до уваги поточні передові технології будь-який бізнес на конкурентному ринку прагне отримати якомога більше винагороди, зазнаючи якомога менше збитків. Виходячи з цього на хвилі широкого застосування штучного інтелекту і математичних рішень штучного інтелекту, популярні страхові компанії беруть приклад з банків і впроваджують системи скорингу клієнтів. В секторі страхування перевіряються шанси клієнта не зазнати страхового випадку і великих збитків для компанії, в відмінності з банками, де перевіряється кредитоспроможність. Такі системи аналізують історичні дані відомих страхових компаній. Експерти аналізують фактори та особисті характеристики кожного клієнта, формують репрезентативні дані, далі на основі вхідної інформації система, що створена аналітиками та інженерами, приймає відповідне рішення. Чим краща система з точки зору якості, тим більше розвивається підприємство з точки зору прибутку. Саме для таких цілей і будується система скорингу клієнтів.

1. ПОСТАНОВКА ЗАДАЧІ

Метою дисертаційної роботи є мінімізація втрат та ризиків, підвищення якості клієнтської бази для автострахових компаній. Для досягнення цієї мети потрібно виконати такі завдання:

1. Ознайомитись з існуючими та популярними скоринговими системами
2. Проаналізувати закономірності між ознаками в історичних даних
3. Підібрати правильний технічний стек для розробки математичного та програмного забезпечення.
4. Реалізувати та протестувати на різних групах скорингову систему для автострахування

Розроблювана автоматизована система повинна виконувати такі функції:

1. Будувати найкращу апроксимацію на основі характеристик осіб відомої вибірки за допомогою математичного алгоритму машинного навчання, надавши бінарну оцінку щодо полісу клієнту
2. Переглянути графічне відображення тестових даних, загальну поведінку заданої групи та проаналізувати адекватність роботи

Дана система повинна мати оффлайн сервісно-орієнтовану архітектуру, тому повинна бути доступною з будь-якої операційної системи, що підтримує .exe файли.

2. ОГЛЯД ІСНУЮЧИХ СИСТЕМ

2.1 Страховий рейтинг. Розвиток скорингу у страхуванні

Скоринг - це інструмент для перевірки платоспроможності клієнта.

За даними Інституту страхової інформації, мета кожної страхової компанії - якомога точніше співвідносити тарифи на страхові поліси з фактичною вартістю відшкодувань. Якщо страховики встановлюють надто високі ставки, вони втратять частку ринку перед конкурентами, які точніше відповідали ставки очікуваним витратам. Якщо вони встановлять занадто низькі ставки, вони втратять гроші.

Цей безперервний пошук точності корисний як для споживачів, так і для страхових компаній. Більшість споживачів виграють, тому що вони не субсидують людей, які страждають гіршими страховими ризиками - людей, які частіше подають претензії, ніж вони.

Комп'ютеризація даних принесла більше точності, швидкості та ефективності підприємствам усіх видів. У сфері страхування кредитна інформація використовувалася протягом десятиліть, щоб допомогти аналітикам вирішити, приймати чи відхиляти заявки на страхування. Нові досягнення в інформаційних технологіях привели до розробки страхових балів, які дозволяють страховикам краще оцінювати ризик майбутніх відшкодувань.

Страховий рейтинг - це числовий рейтинг, заснований на кредитній історії людини. Актуарні дослідження показують, що те, як людина керує своїми фінансовими справами, про що свідчить страховий бал, є хорошим прогнозом страхових відшкодувань. Страхові бали використовуються, щоб допомогти страховикам розрізняти нижчі та вищі страхові ризики і таким чином

отримувати страхові внески, рівні ризику, який вони приймають. За статистикою, люди, які мають низький страховий бал, частіше подають претензію. Страхові бали не включають дані про расу чи дохід, оскільки страховики не збирають цю інформацію від претендентів на страхування.

Скоринг у банківському секторі використовується давно, особливо це помітно при роботі з клієнтами. Банк перевіряє активність клієнта на різних сайтах, його транзакції, місця відвідувань частих локацій і запити в пошукових системах.

В страхуванні скоринг молодший, і це пов'язано з банками. Згідно поправок до 218-ФЗ «Про кредитні історії» у 2014 році страхові компанії змогли переглядати кредитні історії своїх клієнтів. На основі кредитних історій клієнтів почали розробляти скорингові системи для страхової галузі [1]. Зараз у багатьох страхових компаній скоринговий бал - один з головних параметрів договору. Але навіть на сьогоднішній день відсутність узагальнених процесів і єдиних баз даних впливають на впровадження єдиної скорингової системи в сфері страхування. Беручи до уваги автострахування, можна сказати, що останній час все більше з'являються гнучкі тарифи завдяки скорингу. Для компаній вже немає вимог в дільбі збитків по всім клієнтам, пропонуючи дорогі тарифи надійним і ризикованим водіям.

2.2 Оцифрування систем скорингу

Оцифрування (діджиталізація) - це способи приведення будь-якого різновиду інформації в цифрову форму.

Велика кількість клієнтів страхових компаній - це активні користувачі

інтернету і усіх сучасних ІТ-рішень. Виходячи з цього кожна компанія орієнтується різними стратегіями.

Таблиця 2.1 - Орієнтири компаній

Більшість компаній	Частині випадки компаній	Меншість компаній
Орієнтир на вартість: 1. Ціноутворення в залежності від економічних чинників 2. Актурарная функція 3. Витрати на ризики 4. Розподіл витрат	Орієнтир на конкурентів: 1. Ціноутворення 2. Позиціонування 3. Знижки 4. Динамічне ціноутворення	Орієнтир на клієнта: 1. Ціноутворення в залежності від цінності для споживача 2. Сегментація 3. Кастомізація продукту 4. Канальна диференціація

Із розвитком технологій сучасні користувачі мають ряд властивостей серед яких наступні:

1. Підвищене очікування
2. Висока інформованість
3. Суб'єктивна довіра
4. Має велику кількість варіантів для порівняння і вибору
5. Вільна позиція в соціальних медіа

На рисунку 2.1 зображені інструменти над керуванням метрик.

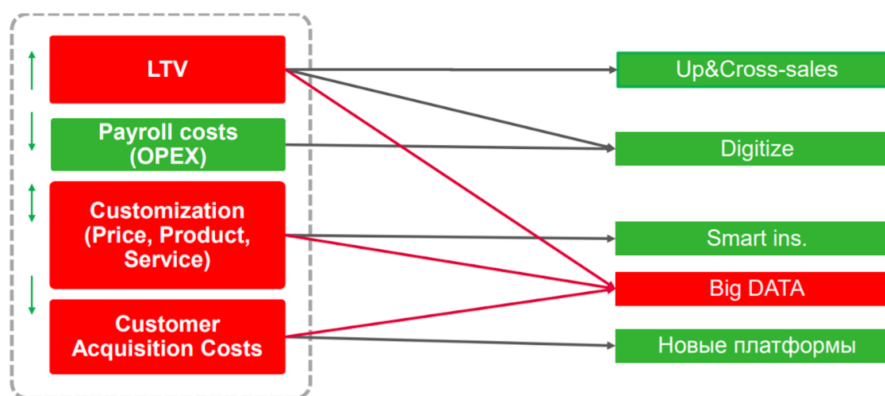


Рисунок 2.1 - Інструменти над керуванням основних метрик

Зазвичай, благонадійні клієнти найімовірніше приходять або повертаються в компанію, що застосовує скорингову систему при розрахунку вартості, тому що, вони зможуть отримати вигіднішу ставку завдяки своїй хорошій кредитній історії.

2.3 Приклади існуючих рішень

Для отримання великих обсягів даних в автострахованні застосовують можливості телематики. Наприклад, розрахунок гнучкої ставка по КАСКО проводиться беручи до уваги дані про кількість ризикових випадків на дорозі, стиль водіння, і взагалі всього водійського бекграунду автовласника.

По-перше, розберемо приклад роботи компанії intouch.

Основні параметри для аналізу:

1. Гнучкість поворотів
2. Швидкість
3. Довжина шляху
4. Місцезнаходження

5. Час використання

Зазвичай 3-4 місяці достатньо для формування стійкої статистичної оцінки. Цього періоду вистачає для повноцінного розуміння поведінки стилю та інших звичок водіння.

Інструкція роботи:

1. Купівля КАСКО
2. Встановлення телематичного модулю
3. Аналіз різних параметрів водіння
4. Присвоєння скорингових балів
5. Нарахування знижки до 20%

На рисунку 2.2 зображено принцип роботи телематики в скорингу [3].



Рисунок 2.2 - Принцип роботи телематики в скоринговій системі

Інтерфейс на стороні клієнта зображено на рисунку 2.3. На стороні клієнта присутні панель календару для водія, а також його маршрут [3].

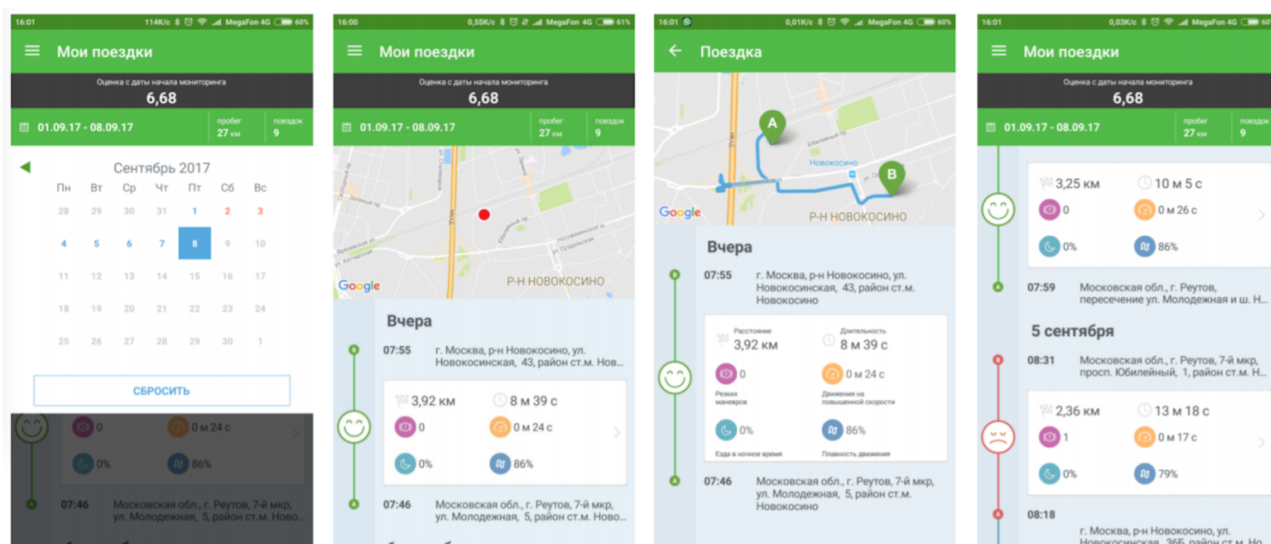


Рисунок 2.3 - Інтерфейс на стороні клієнта

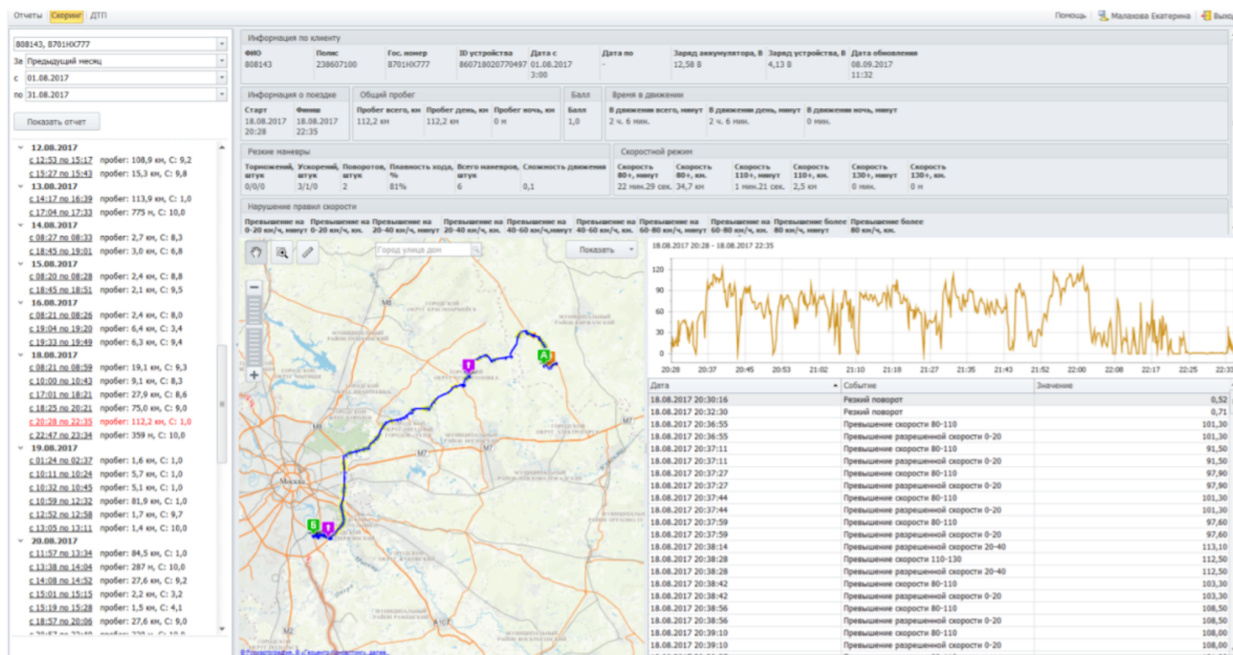


Рисунок 2.4 - Інтерфейс на стороні адміністратора компанії [3]

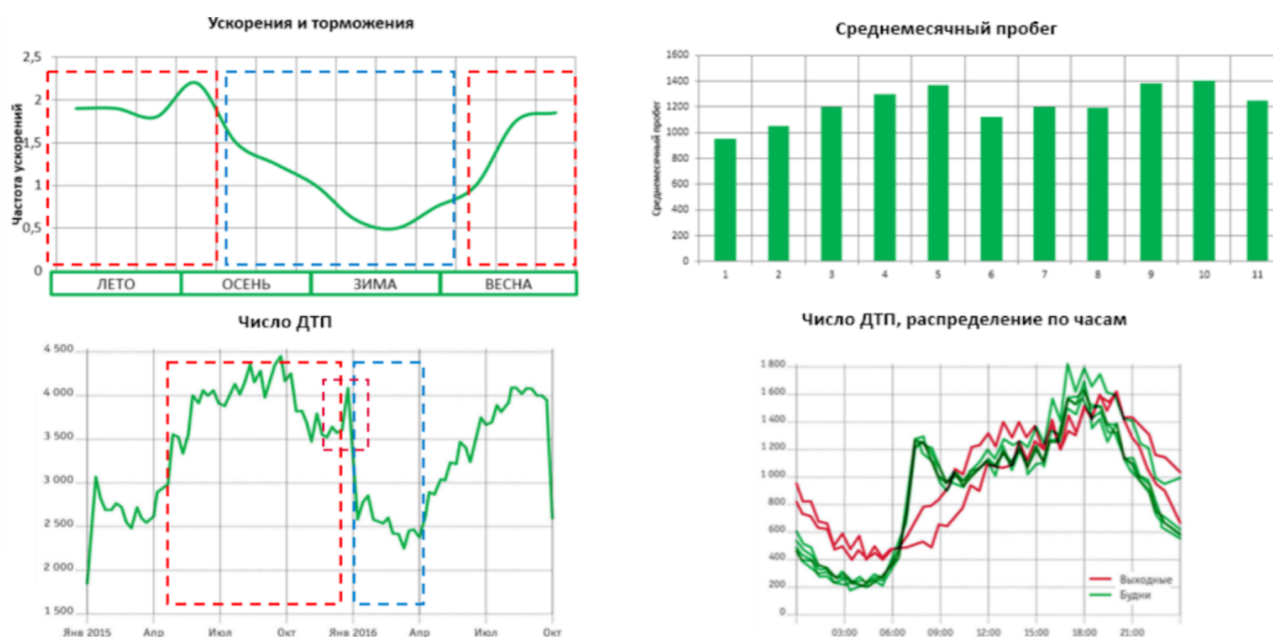


Рисунок 2.5 - Интерфейс статистики на стороне администратора компании [3]

По-друге, розберемо приклад роботи компанії Raxel Telematics. Компанія сама розробила програму розумної страховки з штучним інтелектом, що працює з хмарними технологіями. Вони створили пристрій, що здатний спрогнозувати ризик аварії. Прогноз відбувається за рахунок зібраної раніше статистики про стиль керування. Пізніше компанія Raxel отримала можливість безкоштовно використовувати додаткові можливості сервісу Microsoft Azure [4].

По-третє, розберемо приклад роботи компанії «АХА Страхування». Це чудовий та поки унікальний приклад в українському страховому ринку. Вони реалізували програму розумного автострахування під назвою «КАСКО Смарт». Принцип наступний: під'єднання телематичного пристрою до страхувального авто, через спеціальний роз'єм зчитуються показники водіння, а скорингова система підраховує скоринговий рейтинг через встроєний додаток за благополучне водіння, але пристрій не зберігає дані про локацію клієнта та не передає особисті дані в свою базу даних [5].

2.4 Висновки до розділу

Отже, в поточному розділі було розглянуте впровадження до скорингу в страхуванні, напрями та положення до сучасних технологій, систематизація систем скорингу, різні приклади компаній, що використовують телематичний скоринг.

3. ОПИС МАТЕМАТИЧНИХ АЛГОРИТМІВ СИСТЕМИ

3.1 Методи візуалізації багатовимірних даних

Матриця розсіювання - це оцінка матриці коваріації. Матриця розсіювання використовується також з ціллю зменшення розмірності. Коли є k змінних матриця матиме k стовпців і k рядків, тобто матрицю X_k .

На рисунку 3.1 зображено матриця розсіювання [6]

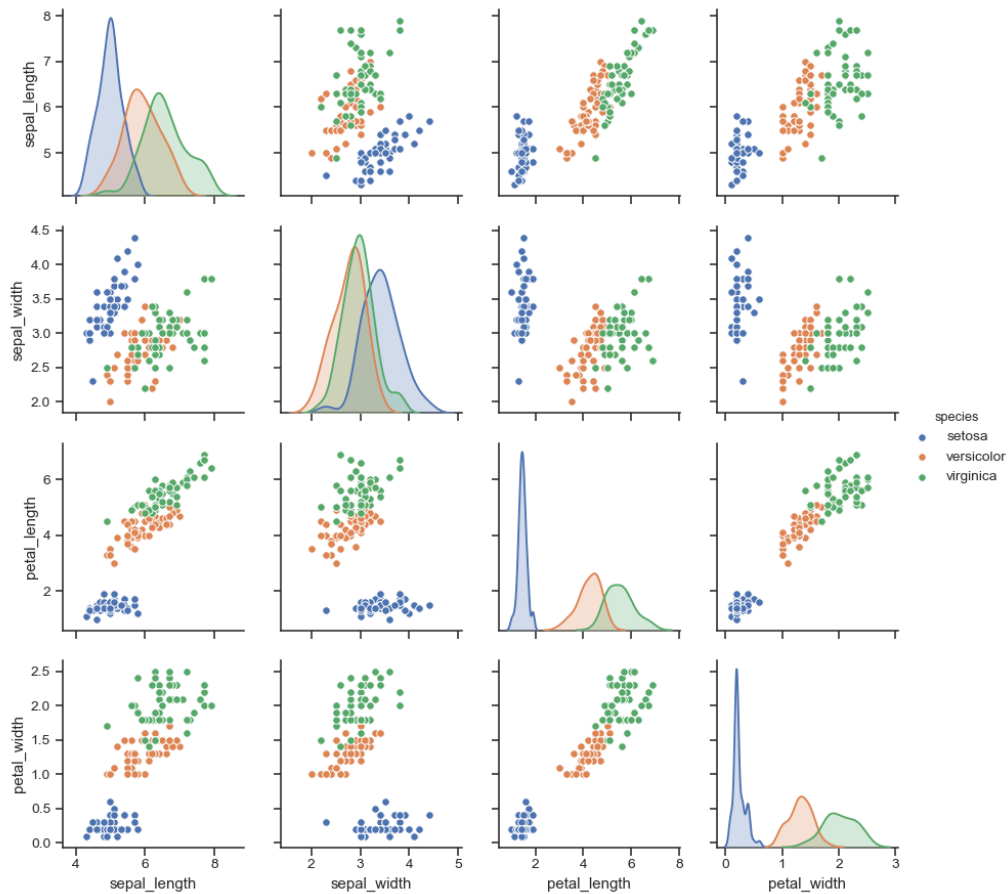


Рисунок 3.1 - матриця розсіювання

Матриця діаграми розсіювання - це матриця точкових діаграм. Вона

використовується для відображення відносин (двовимірних) між сполуками числових змінних. Будь-яка точкова діаграма матриці зображає відношення між двома змінними. Це дозволяє показати багато відносин на одній діаграмі.

Матриця діаграми розсіювання створена з одного графіка попереднього перегляду та таблиці маленьких графіків. Ці маленькі графіки є попарними діаграмами розсіювання.

Точкова діаграма (діаграма розсіювання) - діаграма, що базується на декартовій системі координат та відображає значення двох змінних для певного набору даних [8]. Набор даних показаний у вигляді набору точок. Будь-який набір має значення однієї змінної, це нам говорить про те, що він визначає положення на горизонтальній та вертикальній осях. Таку діаграма теж називають розсіювальною (граф розкиду).

На точковій діаграмі кожний елементарній одиниці набору даних відповідає точка, її координати дорівнюють значенням двох будь-яких інших параметрів цього спостереження. Якщо передбачається корелювання параметрів, то часто значення залежного відкладається по вертикальній осі, а незалежного - по горизонтальній. Суть діаграми розсіювання - продемонструвати наявність або відсутність залежності між двома змінними.

Матриця діаграми розсіювання рахується за наступною формулою:

$$S = \sum_{k=1}^n (x_k - m)(x_k - m)^T$$

$$m = \frac{1}{n} \sum_{k=1}^n x_k,$$

де m – вектор середніх.

У візуалізації даних крива Ендрюса - це ще один популярний спосіб візуалізації структури у великих розмірних даних. Часто це згорнута, не ціла версія радарної діаграми Кент-Ківіат, або згладжена версія паралельної ділянки координат [9].

Значення X_d - це розмірна точка даних, за умовою, що вона є елементом d вимірного простору дійсних чисел. Представимо об'ємні дані з числом для кожного з їх розмірів:

$$x = \{x_1, x_2, \dots, x_d\}$$

Щоб візуалізувати їх, крива Ендрюса визначає кінцевий ряд Фур'є:

$$f_x(t) = \frac{x_1}{\sqrt{2}} + x_2 \sin \sin(t) + x_3 \cos \cos(t) + x_4 \sin \sin(2t) + x_5 \cos \cos(2t) + x_6 \sin \sin(3t) + \dots$$

Далі така функція будується для $-\pi < t < \pi$. Тепер кожну точку даних можна розглядати як лінію між $-\pi$ і π . Розглянемо формулу проекції точки даних на вектор:

$$\left(\frac{1}{\sqrt{2}}, \sin(t), \cos(t), \sin(2t), \cos(2t), \dots \right)$$

У випадку наявності структури даних, криві Ендрюса явно це показують. Такі криві використовуються в різних галузях. Наприклад, частина з них використовується в візуалізації навчанні штучних нейронних мережах, виявленні періодів, контролі якості продукції, видатків у часових рядах та аналізу листування. Проекція на коло приводить до радарної діаграми.

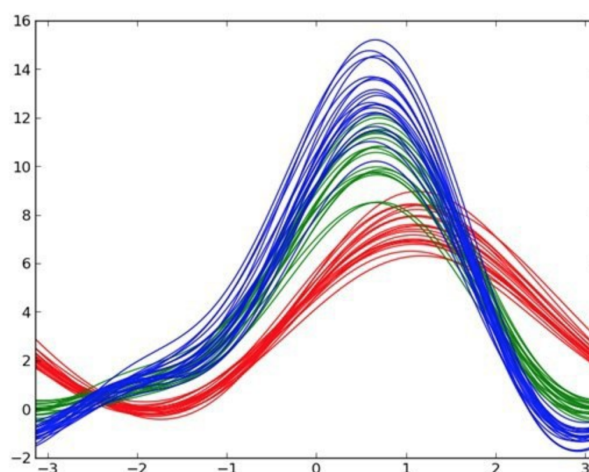


Рисунок 3.2 - криві Ендрюса на множині даних видів ірису

Розглянемо лінії, що відповідають схожим значенням, вони в свою чергу мають схожі формули. Кожна з точок завжди буде відповідати одній лінії на графіці, тому що, розмір даних немає ніяких значень.

Для візуалізації багатовимірних даних також існують і інші методи, наприклад, такі як: радарні діаграми, метод головних компонент та інші.

Криві Ендрюса працюють з високими розмірностями, і це є головною перевагою над матрицями діаграм розсіювання, але так як в задачі використовується невелика кількість змінних матриця розсіювання є також не поганим методом для графічної візуалізації поточної задачі.

3.2 Динамічне навчання. Логістична регресія

Машинне навчання - це спосіб імітувати результат роботи природного (людського) інтелекту за допомогою алгоритмів та програм. Для побудови таких методів використовуються засоби теорії графів, математичної статистики, теорії вірогідностей, оптимізації методів та різні технології роботи з даними в цифровій формі.

Для побудови найкращої апроксимації існує багато відомих методів. Ось деякі з них:

1. Логічні моделі (дерева)
2. Наївний байєсів класифікатор
3. Загальні лінійні моделі (для класифікації логістична регресія)
4. SVM

5. Глибокі нейронні мережі

Ці алгоритми широко застосовуються в багатьох прикладних сферах, таких як: біоінформатика, фільтрація спаму, комп'ютерний зір, геоаналіз, обробка штучної мови тощо.

Обґрунтуємо чому в рамках даної задачі було вирішено використовувати логістичну регресію.

Дерева рішень - це логічний алгоритм. Саме дерево містить в собі «листя» і «гілки». Цільова функція залежить від записаних атрибутів на «гілках» (ребрах) дерева рішення, а в інших вузлах знаходяться атрибути, за якими розрізняються випадки. Для класифікації нових випадків потрібно пройти по дереву до листа і видати відповідне значення. Такі дерева часто використовуються в статистичному аналізі даних. Алгоритм не потребує попередньої обробки даних та має просту інтерпретацію, але в той же час є нестійким.

Наївний байєсів класифікатор - це алгоритм, який використовує теорему Байєса для класифікації об'єктів. Класифікатори наївних байєсів передбачають сильну або наївну незалежність між атрибутами точок даних, а це говорить про те, що вони при частичних випадках, можуть навчатися динамічно дуже ефективно. Для оцінки параметрів наївних Байєсових моделей використовують метод максимальної правдоподібності. Плюсами наївного байєсівського класифікатора є мала кількість даних необхідних для навчання, класифікації та оцінки параметрів. Мінусом є строге припущення незалежності ознак (на практиці це часто порушується).

SVM – машина опорних векторів. Метою машинного алгоритму опорних векторів є знаходження гіперплощини в N-вимірному просторі (N - кількість ознак), що чітко класифікує точки даних. Щоб розділити два класи точок даних, можна вибрати багато можливих гіперплощин. Зазвичай знаходять максимальну

відстань між точками даних обох класів. Збільшення межі відстані надає деяке посилення, щоб майбутні точки даних можна було класифікувати з більшою впевненістю.

Глибокі нейронні мережі - окремий великий клас алгоритмів штучного інтелекту. Множина дочірніх алгоритмів має різні специфікації та топології. Вивчення даних на таких моделях виходить за обсяги звичайного машинного навчання та переходить у іншу тематику з назвою - глибоке навчання. Недолік даних моделей в витратності ресурсів. Немає смислу використовувати з малим об'ємом даних.

Логістична регресія - це відповідний регресійний аналіз, який слід проводити, коли залежна змінна є дихотомічною (бінарною). Як і всі регресійні аналізи, логістична регресія є прогнозним аналізом. Логістична регресія використовується для опису даних і пояснення зв'язку між однією залежною двійковою змінною та однією або кількома номінальними, порядковими, інтервальними або незалежними змінними на рівні співвідношення. В методі логістичної регресії не проводиться прогнозування значення числової змінної виходячи з вибірки вихідних значень, на відміну від звичайної регресії. Значенням функції є ймовірність того, що поточне початкове значення належить до певного класу. Для простоти, припустимо, що у нас є тільки два класи і ймовірність того, що деяке значення належить до певного класу. Виходячи з цього результат логістичної регресії завжди знаходиться в інтервалі $[0, 1]$.

Ідея логістичної регресії полягає в тому, що простір вихідних значень може бути розділеним на два відповідних класа області лінійним кордоном. Для двох вимірів це просто пряма лінія без вигинів. Для трьох - площина, і т. д. Така межа задається в залежності від навчального алгоритму та наявних вихідних даних. Для коректної роботи точки вихідних даних повинні розділятися лінійним кордоном на дві області. [10]

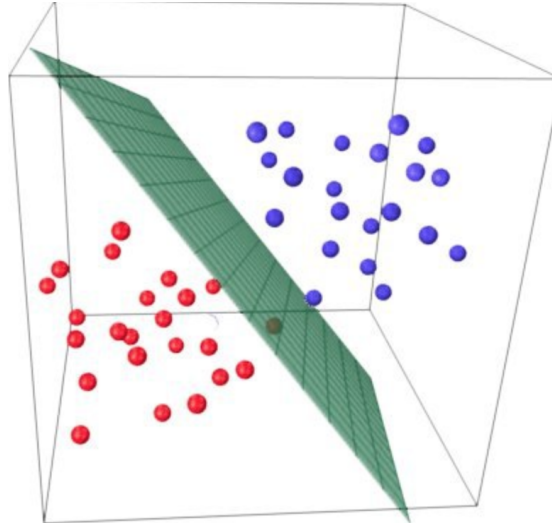


Рисунок 3.3 - Лінійний класифікатор

Задана вибірка - це множина пар (x_i, y_i) , в яких опис i -го елемента $x_i \in R^n$, и значення залежної змінної $y \in \{0, 1\}$ [11].

Розпишемо модель логістичної регресії, з залежністю між вільними і залежна змінними

$$y = \text{logit}^{-1}(z) + \varepsilon = \frac{1}{1 + \exp(-z)} + \varepsilon,$$

де $z = b_0 + \sum_{j=1}^n b_j x_j$.

Прийmemo позначення $p_i = f(b_i x_i)$, вектор $b = [b_0, \dots, b_n]^T$

Для зручності подальшого викладу позначимо вибірку вільних змінних як

$$X = \begin{bmatrix} 1 & x_1^T & 1 & x_m^T \end{bmatrix}$$

Знайdemo таке значення вектора параметрів, яке б приносило мінімум нормі вектора невязок:

$$S = \|y - p\|^2 = \sum_{i=1}^m (y_i - p_i)^2$$

Задовільні параметри розташовуються послідовно завдяки ітераційному методу найменших квадратів з використанням зважування елементів вибірки. Наведений нижче алгоритм заснований на алгоритмі Ньютона-Рафсона[14].

Спочатку алгоритму задаються параметри початкового наближення (скаляр):

$$b_0 = \log \frac{\tilde{y}}{1-\tilde{y}}, \text{ де } \tilde{y} = \frac{1}{m} \sum_{i=1}^m y_i$$

середнє значення вибірки залежною змінною і значення $b_j = 0$ для $j = 1 \dots n$.

Далі рекурсивно-ітеративно відбуваються наступні дії:

1. Обчислюється змінна $z = X_b$
2. Обчислюється оновлене значення вибірки

$$p = \frac{1}{1+\exp(-z)}$$

3. Обраховується вектор значень

$$u = z + y - pw, \text{ де } w = p(1 - p)$$

4. Обраховується вектор вагів значень
5. Вирішується задача найменших квадратів із зважуванням елементів вибірки.

Як тільки норма різниці векторів параметрів на кожній ітерації перевищуватиме задану константу процедура зупиниться.

$$|b^{next} - b^{previous}|^2 \leq \Delta_b$$

На рисунку 3.4 зображена логістична крива

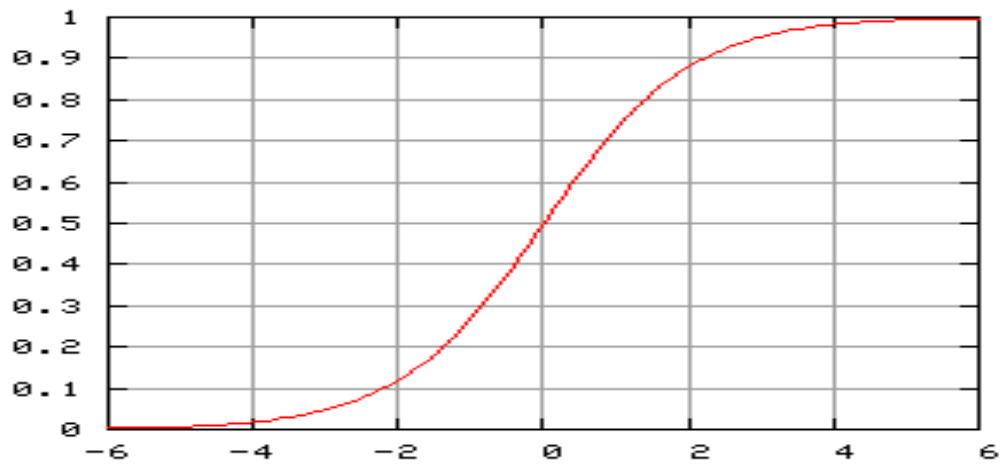


Рисунок 3.4 - Приклад логістичної кривої

Базуючись на цій функції робиться припущення ймовірності приналежності об'єкта до класу.

Якщо $y = 1$:

$$P\{y = 1 | x\} = f(z)$$

Якщо $y = 0$:

$$P\{y = 0 | x\} = 1 - f(z) = 1 - f(\theta^T X)$$

$$f(z) = \frac{1}{1 + e^{-z}}$$

$$z = \theta^T x = \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n$$

Розглянемо ϕ -ю розподілу y при заданому x (розподіл Бернуллі):

$$P\{x\} = f(\theta^T x)^y (1 - f(\theta^T x))^{1-y}$$

$$y \in \{0, 1\}$$

Максимізація функції правдоподібності еквівалентна максимізації її логарифма. Максимізувати функцію можна одним із методів, таких як: метод Ньютона, метод градієнтного спуску або стохастичний градієнтний спуск. 3

точки зору оптимізації вирішується задача мінімізації функції втрат. В поточному рішенні логістична регресія буде оптимальним вирішенням поставленої задачі. Об'єм тренувальних даних не великий. Кількість параметрів невелика, змінні незалежні. Кореляція не спостережується. Класи приймаються лінійно подільні, що і є умовою для використання лінійного класифікатора.

Порівняємо деякі алгоритми класифікації із логістичною регресією по метриці точності, часу тренування і виконання алгоритму.

Спершу, порівняємо алгоритм на основі даних MNIST (70000 навчальних прикладів, 784 розмірність, 10 класів)

Таблиця 3.1 - Порівняння алгоритмів на основі даних MNIST

Алгоритм	Точність	Час тренування	Час виконання
Логістична регресія	90.98%	265.1802s	0.0543s
БШП	98.01%	78.4591s	0.4020s
Згорткова мережа	96.54%	392.0012s	1.1980s
Дерево рішень	66.01%	3.1451s	0.0358s
Naive Bayes	57.24%	0.3798s	0.9156s

Отже, у задачах комп'ютерного зору логістична модель справляється досить добре по точності, але поступається глибоким нелінійним моделям. В той же час програє майже всім алгоритмам в часі тренування через великий об'єм даних.

Порівняємо алгоритм на основі даних IRIS (150 навчальних прикладів, 4

розмірність, 3 кількість класів)

Таблиця 3.2 - Порівняння алгоритмів на основі даних IRIS

Алгоритм	Точність	Час тренування	Час виконання
Логістична Регресія	91.84%	0.0012s	0.0002s
БШП	88.50%	1.5739s	0.0058s
Логістична регресія	89.25%	0.0016s	0.0001s
Метод опорних векторів, лінійний	87.54%	0.0005s	0.0003s
Наївний байєс	91.65%	0.0011s	0.0004s

Отже, логістична регресія з налаштуванням регуляризації працює краще ніж з стандартне значення. Проявляє гарну точність і швидкість виконання програми на відміну від нелінійних моделей.

Логістична регресія як метод машинного навчання вбудована у пакети різних мов програмування:

1. Java - високорівнева програмна мова. Реалізація у пакеті SPSS
2. Matlab - мова для математичних рішень. Реалізація у пакетах Statistics і Machine learning Toolbox
3. R - мова для математичних рішень та аналізу даних. Реалізація у пакеті CRAN

4. Python - високорівнева програмна мова. Часто використовується для рішення задач штучного інтелекту. Реалізація у пакетах statsmodels і scikit-learn

В даній роботі вибір пав на використання мови програмування Python з модулем scikit-learn через його зручність, підтримку розробниками та універсальність.

3.3 Висновки до розділу

У цьому розділі було розглянуто та проаналізовано математичні підходи і алгоритми для розв'язання поставленої задачі. Для побудови найкращої апроксимації обрано та обґрунтовано вибір алгоритму логістичної регресії. Обрано мову програмування Python для вирішення поставленої задачі.

4. ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМУ ТА СИСТЕМИ

4.1 Опис програми та аналізу даних

Для поставленої задачі основним інструментом була обрана мова програмування Python. В даній роботі були використані бібліотеки:

1. Scikit-learn (машинне навчання)
2. Matplotlib, pandas (візуалізація)
3. Tkinter (розробка графічного інтерфейсу)

Формат вхідних даних - json. У json міститься колекція дописів з потенційними клієнтів.

Будується клас на основі атрибутів особистих даних особи:

1. Ідентифікації (ідентифікатор, ПІБ, дата народження)
2. Для моделі (стаж водіння, наявність поганих звичок та ціна транспортного засобу який страхується)

Вихідні дані відображаються у форматі листбоксу на графічному полі. Приклад даних на рисунку 4.1

```

{
  "Collection": [
    {
      "Id": "000001",
      "Cost": 52010,
      "DriveExp": 1,
      "BadHabbits": 'No',
      "FullName": "Ivanov Oleg ",
      "Birthday": "1998-03-13"
    },
    {
      "Id": "000002",
      "Cost": 51000,
      "DriveExp": 2,
      "BadHabbits": 'Yes',
      "FullName": "Vando Ostop",
      "Birthday": "1993-12-17"
    },
    {
      "Id": "000003",
      "Cost": 49500,
      "DriveExp": 2,
      "BadHabbits": 'No',
      "FullName": "Kravchuk Anna",
      "Birthday": "1987-04-22"
    },
    {
      "Id": "000004",
      "Cost": 4060,
      "DriveExp": 29,
      "BadHabbits": 'Yes',
      "FullName": "Ivanov Oleg ",
      "Birthday": "1963-07-09"
    }
  ]
}

```

Рисунок 4.1 - Формат вихідних даних

У разі позитивного результату ціна полісу обчислюється як:

$$\text{ціна полісу} = \frac{1}{2} * \text{ціна авто} + \frac{1000}{\text{стаж водія}}$$

Графічний додаток реалізований як:

ПЗ має чотири кнопки для роботи (кнопка запуску алгоритму, дві для графіків та кнопка виходу). Зліва по центру є листбок - поле у якому після нажаття кнопки запуску алгоритму буде міститись результат роботи ПЗ. Після натискання кнопки запуску користувач може обрати у спливаючому вікні комп'ютера файл із вхідними даними json формату (Choose File). Після

закінчення роботи алгоритму користувач може переглянути одну з графічних опцій, що при натиску на одну із кнопок з графіками (Draw Andrews, Draw Scatter) надає можливість побачити графічну картину. Закрити додаток можна кнопкою для закриття додатку (Cancel).

Вигляд вищезазначеного інтерфейсу знаходиться на рисунку 4.2

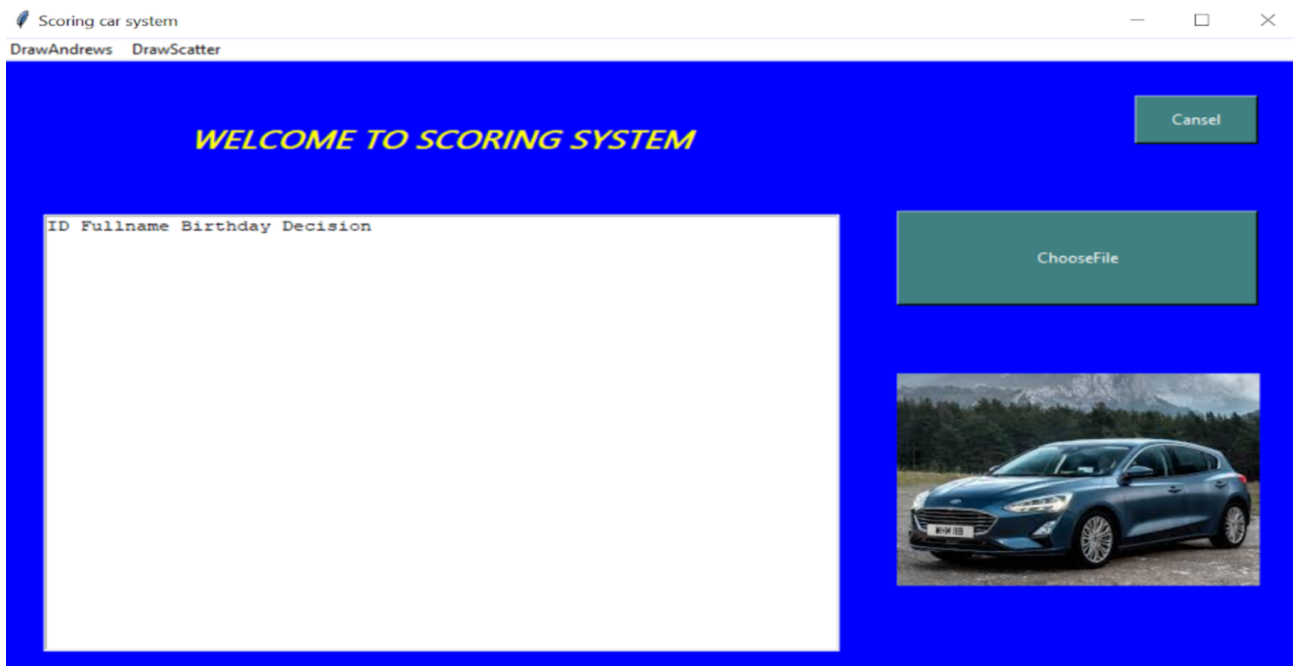


Рисунок 4.2 - Клієнтський вигляд додатку

Тренування алгоритму відбувається на вибірці із 20 реальних прикладах із страхових баз. Це був реєстр премій компанії ТАС 5 річної давнини. У ньому відображено параметри моделі та розмір збитку причинений клієнтом у страховій події. Якщо ризик більше 0 для вибірки обирається параметр 0, так як це є негативним випадком для видачі полісу. Так формується статистична картина, що репрезентативно відображає якість та ризик потенційних клієнтів на подібних компаніях.

У даній моделі були обрані наступні три значимі характеристики: стаж водія (у роках), вартість авто та наявність поганих звичок.

При огляді та тренуванні даних були виявлені наступні властивості:

1. Ціна розподілена за рівномірним законом $\sim \text{Uniform}(3000, 53000)$
2. Стаж водія також за рівномірним законом $\sim \text{Uniform}(0, 30)$
3. Погані звички мають 25% людей

Досліджувані параметри не корелюють тому не має сенсу в процедурі декорелювання. Побудована матриця кореляції шляхом емпіричної оцінки матриці коваріації:

$$[1 \ -0.38 \ -0.16]$$

$$[-0.38 \ 1 \ 0.19]$$

$$[-0.16 \ 0.19 \ 1]$$

Отримано модель розмірністю 3 ознакового простору із двома неперервними ознаками та одним бінарним.

На рисунку 4.3 зображено тренувальний набір на площині шляхом пониження розмірності методу головних компонент.

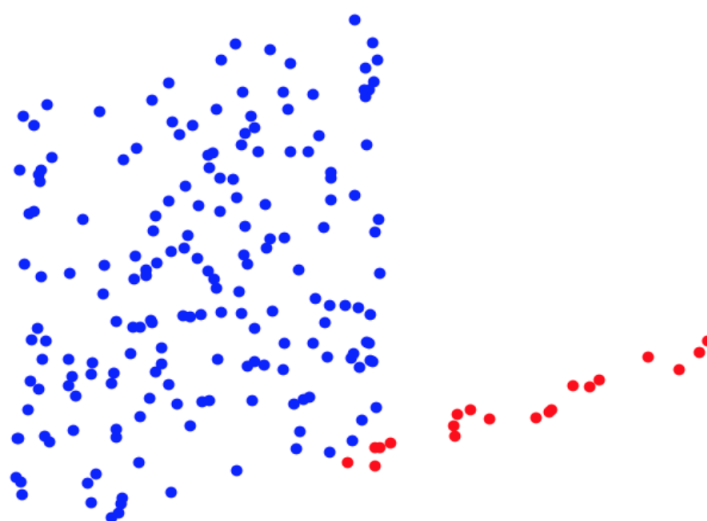


Рисунок 4.3 - Проекція на площину тренувальної вибірки

Виявлено незбалансований набір, коли екземплярів одного класу набагато більше ніж іншого. Також замічено, що люди з дорогим авто, малим стажем водіння та поганими звичками є потенційно ризикованими клієнтами. В поточній вибірці спостерігається випадок лінійної сепарабельності класів, тому і була використана логістична регресія, як лінійний алгоритм.

Завершальний етап підготовки даних - це кодування та нормалізація ознак в один масштаб, так як лінійний алгоритм може зазнати дисбалансу коефіцієнтів.

Одним з найосновніших етапів побудови апроксимаційної моделі є налаштування її параметрів.

L_2 penalty - це вид регуляризації вагових параметрів (регуляризація Тихонова) [16]. Це випадок, коли ми вимагаємо від коефіцієнтів щоб їх норма не перевищувала рамки, тому, функцію втрат штрафуємо регуляризуючим доданком.

class weights - у функції втрат вносимо пропорційність для навчальних прикладів, ця методика

$\text{tol } 10^{-4}$ - критерій зупинки алгоритму. Беремо значення за замовчуванням.

Максимальну кількість ітерацій - 100. Величина вибірки середня і для навчальної задачі нам треба не багато часу щоб натренувати модель.

Метод мінімізації обираємо LBFGS - так як ми маємо невелику вибірку і кількість параметрів навчання усього 3. Задля більш високої точності ми можемо дозволити використати квазіньютонівський метод і на кожній ітерації порахувати Гессіан для функції втрат.

C - константа налаштування регуляризації. Обираємо так щоб основна функція була сильніша за штраф у C раз. Була підібрана із побудови кросс валідаційної кривої шляхом перехресної перевірки для вибору оптимального значення константи для моделі.

На рисунку 4.4 зображена тестова кросс валідаційна крива

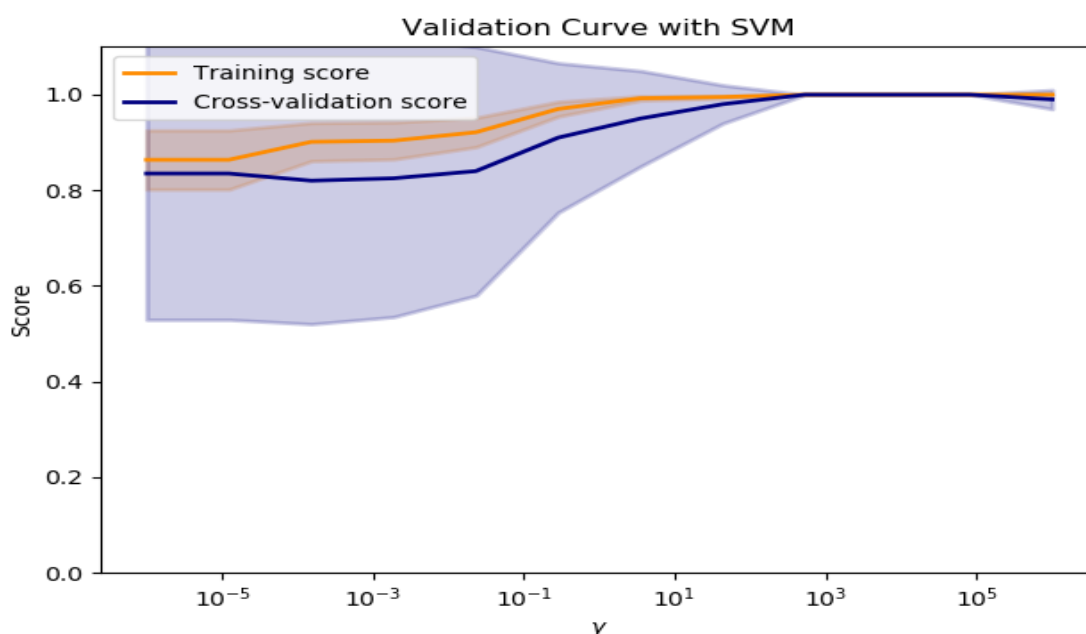


Рисунок 4.5 - Кросс валідаційні криві для підбирання константи C

Як бачимо крива найбільше значення приймає коли C в діапазоні від 100 до 100000. Ми обираємо значення 100. Перехресна перевірка є зручним емпіричним шляхом для обрання найкращої моделі. У якості метрики точності було обрано звичайну точність.

4.2 Перевірка на тестовій групі

Результати роботи на тестових даних проілюстровані на рисунку 4.5

ID	Fullname	Birthday	Decision
000020	- Butenko Anastasia	- 1997-05-09	- Not Give policy
000019	- Ivanova Olga	- 1964-10-23	- Give policy-15006.0
000018	- Kruglova Anastasia	- 1990-07-25	- Give policy-17275.5
000017	- Kubchunskui Vitalii	- 1970-03-31	- Give policy-24520.0
000016	- Vakhunskia Alla	- 1995-02-17	- Give policy-11418.5
000015	- Makovets Olga	- 1971-11-19	- Give policy-18852.5
000014	- Kalamarchuk Viktoria	- 1980-06-10	- Give policy-14552.0
000013	- Krabenko Danilo	- 1974-03-05	- Give policy-7937.0
000012	- Ivanov Oleg	- 1969-09-01	- Give policy-7554.5
000011	- Vandenko Olexandr	- 1973-05-01	- Give policy-6066.0
000010	- Rudenko Kateryna	- 1980-03-31	- Give policy-22867.5
000009	- Berenchuk Nazar	- 1980-07-13	- Give policy-5575.5
000008	- Kalamar Nadia	- 1987-08-26	- Give policy-5018.5
000007	- Andruhovich Artur	- 1970-09-28	- Give policy-3645.0
000006	- Markovets Olga	- 1974-03-31	- Give policy-3432.0
000005	- Goncharuk Mark	- 1990-10-13	- Give policy-3122.0
000004	- Ivanov Oleg	- 1963-07-09	- Give policy-2064.0
000003	- Kravchuk Anna	- 1987-04-22	- Not Give policy
000002	- Vando Ostap	- 1993-12-17	- Not Give policy
000001	- Ivanov Oleg	- 1998-03-13	- Not Give policy

Рисунок 4.5 - Оброблені дані на тестовій групі

На рисунку 4.6 зображені криві Ендрюса результату алгоритму на тестовій групі. Як ми бачимо в даному випадку алгоритм спрацював із деякими неточностями. Тому візуальна перевірка має сенс.

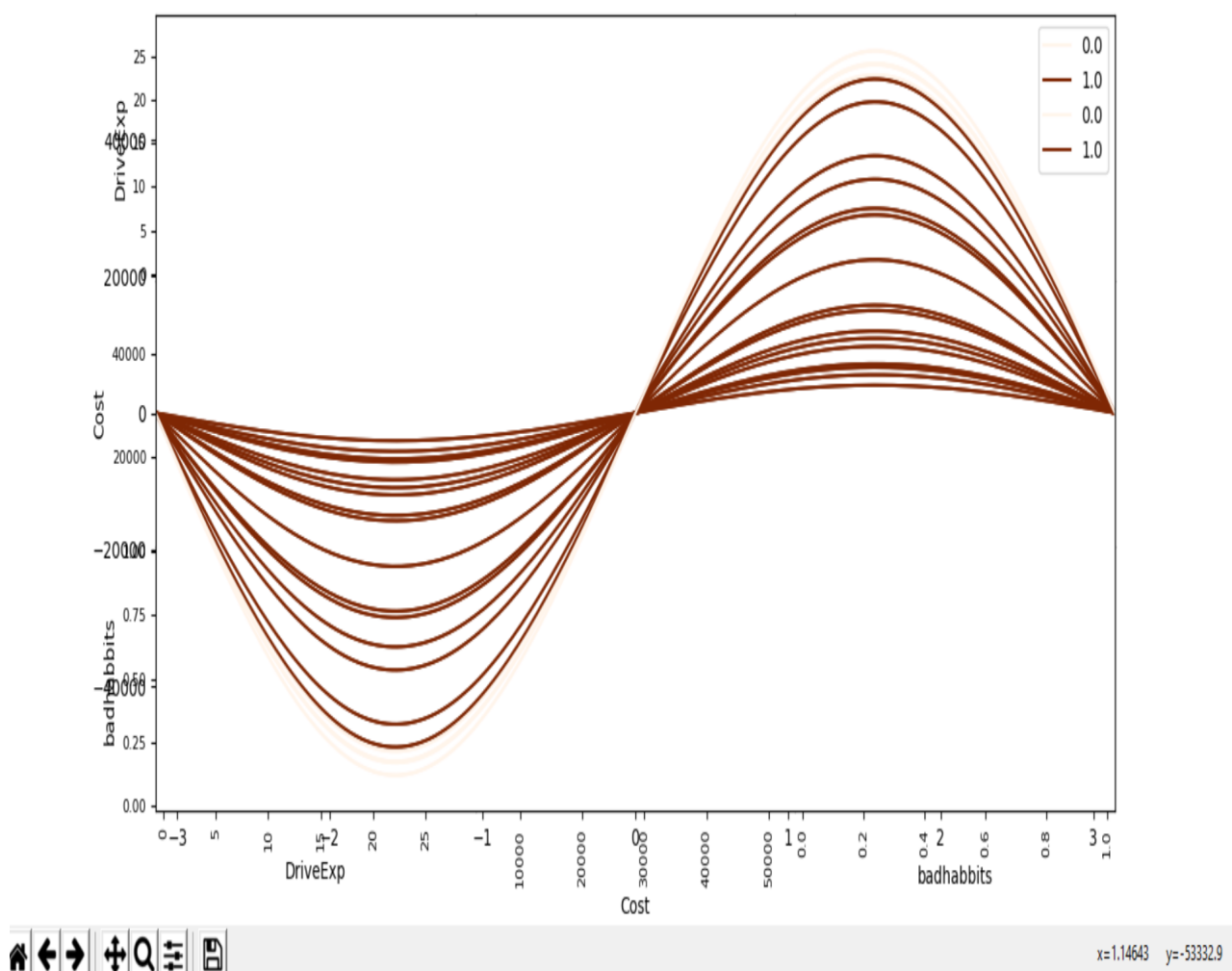


Рисунок 4.6 - Криві Ендрюса

На рисунку 4.7 зображено матрицю розсіювання результатів роботи на тестовій групі. По діагоналі було використано апроксимацію розподілу методом ядерної оцінки щільності розподілу.

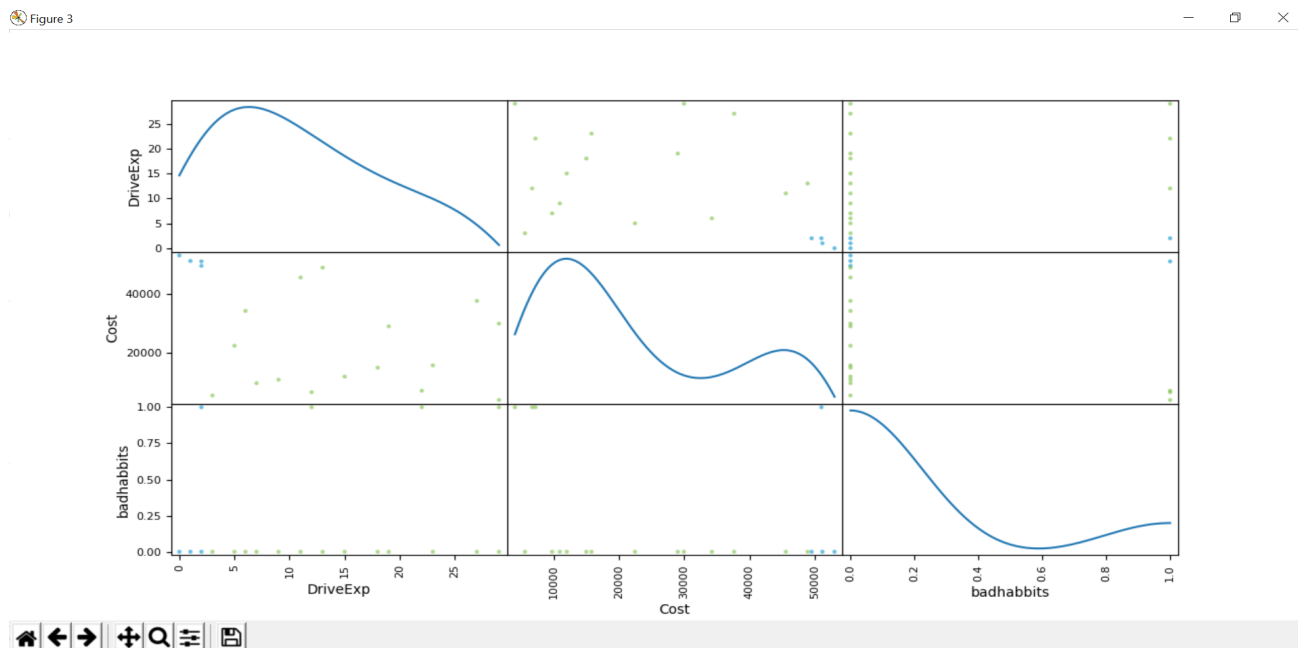


Рисунок 4.7 - Матриця розсіювання тестової групи

Таблиця 4.1 - Тестова група із результатами програми та очікуваними

ID	Cost	BadHabbits	DriveExp	Expected	Calculated
1	63012	No	0	0	0
2	41003	Yes	2	0	0
3	47503	No	1	1	0
4	3081	Yes	28	1	1
5	4500	No	3	1	1
6	8400	No	11	1	1
7	9300	No	20	1	1
8	8652	No	7	1	1
9	11723	No	9	1	1
10	55555	No	11	1	1
11	13000	Yes	15	1	1
12	13889	No	18	1	1
13	16021	No	23	1	1
14	28000	No	19	1	1
15	39543	Yes	27	1	1
16	12435	No	5	1	1
17	38686	No	13	1	1
18	35235	No	6	1	1
19	39934	No	29	1	1
20	61256	No	10	1	1

У вигляді таблиці порівняємо очікувані та реальні результати виконання.

У даних прикладах наша точність має значення $accuracy = 0.95$, але для не збалансованих класів звичайна точність є зміщеною оцінкою класифікатора. Тому порахувавши наступні показники ми можемо побачити більш детальну картину:

$$Tp = 16;$$

$$Tn = 2$$

$$Fp = 0$$

$$Fn = 1$$

Порахуємо на точність і повноту

$$precision = Tp / (Tp + Fp) = 1$$

$$recall = Tp / (Tp + Fn) = 0.93$$

Обрахуємо коефіцієнт кореляції Метьюза [15]:

$$MCC = (Tp * Tn - Fp * Fn) / \sqrt{(Tp + Fp)(Tp + Fn)(Tn + Fp)(Tn + Fn)} = 0.78$$

Отже, ми бачимо що наша модель є доволі непоганою, але не досконалою - для благонадійних клієнтів модель іноді може помилятися і відносити їх до групи збиткових клієнтів.

4.3 Висновки до розділу

У цьому розділі було описано реалізацію ПЗ системи, інструменти для роботи з алгоритмом та вхідними даними, графічного інтерфейсу, досліджено вибір тренувальних даних та їх статистичну структуру. По отриманим даним проводилось додаткове налаштування алгоритму і тестування контрольних прикладів на певній групі.

ВИСНОВКИ

Для виконання поставленої задачі спершу було проведено ознайомлення з існуючими та сучасними скоринговими системами.

По-друге, в результаті аналізу закономірностей між ознаками в історичних даних було проаналізовано основні принципи та функціонал скорингових систем.

По-третє, було реалізовано скорингову систему автострахування на основі прикладного математичного та ПЗ.

В ході виконання дисертаційної роботи було створено скорингову систему, що дозволяє на основі отриманих даних визначати бінарну оцінку щодо надання полісу клієнту. Використаний стек технологій: Python3 з використанням бібліотек - matplotlib, scikit-learn, pandas, tkinter.

Система дозволяє переглянути графічне відображення тестових даних, загальну поведінку заданої групи на прикладі кривих Ендрюса та матриць розсіювання. Результат роботи системи - це ПЗ з побудованою апроксимацією за допомогою математичного алгоритму машинного навчання на основі характеристик осіб відомої вибірки, що надає бінарну оцінку полісу клієнту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Страховой скоринг на основе кредитной истории [Электронный ресурс] – Режим доступа до ресурсу:
<https://insuranceconference.ru/2015/files/presentations/VolkovA.pdf>
2. Что такое скоринг в автостраховании и как он может работать на клиента [Электронный ресурс] – Режим доступа до ресурсу:
<https://www.ins-team.ru/news/scoring>
3. Диджитализация страховых продуктов: опыт использования Big Data [Электронный ресурс] Режим доступа до ресурсу:
<https://bosfera.ru/pdf/17.%20%D0%9F%D0%B0%D0%B2%D0%B5%D0%B%D0%91%D1%83%D1%82%D0%B5%D0%BD%D0%BA%D0%BE.pdf>
4. Клуб аккуратных водителей: как заработать на «умном» каско [Электронный ресурс] – Режим доступа до ресурсу:
https://amp.rbc.ru/rbcnews/own_business/23/06/2016/576a62769a79474a1a1abcb1
5. Программа «КАСКО Смарт» от «АХА Страхование» позволяет клиентам экономить до 60% [Электронный ресурс] – Режим доступа до ресурсу:
<https://forinsurer.com/news/19/06/12/36909>
6. Scatterplot Matrix [Электронный ресурс] Режим доступа до ресурсу:
https://seaborn.pydata.org/examples/scatterplot_matrix.html

7. Матрица диаграммы рассеяния [Электронный ресурс] – Режим доступа до ресурсу:
<https://pro.arcgis.com/ru/pro-app/help/analysis/geoprocessing/charts/scatter-plot-matrix.htm>
8. Сеньо П. С.(2007).Теорія ймовірностей та математична статистика(вид. 2-ге, перероб. і доп.). Київ:Знання. с.446.
9. Andrews, David F. (1972). "Plots of High-Dimensional Data".International Biometric Society.18(1): 125–136.JSTOR2528964
- 10.Logistic Regression (for dummies) [Электронный ресурс] – Режим доступа до ресурсу:
<https://codesachin.wordpress.com/2015/08/16/logistic-regression-for-dummies/>
- 11.Логистическая регрессия (пример) [Электронный ресурс] – Режим доступа до ресурсу:
http://www.machinelearning.ru/wiki/index.php?title=%D0%9B%D0%BE%D0%B3%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D1%8F_%28%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80%29
- 12.Comparing Classifiers [Электронный ресурс] – Режим доступа до ресурсу:
<https://martin-thoma.com/comparing-classifiers/>
- 13.Логістична регресія [Электронный ресурс] – Режим доступа до ресурсу:
https://uk.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D1%96%D1%81%D1%82%D0%B8%D1%87%D0%BD%D0%B0_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%96%D1%8F
- 14.Метод Ньютона - Рафсона [Электронный ресурс] – Режим доступа до ресурсу:
<https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B>

[4_%D0%9D%D1%8C%D1%8E%D1%82%D0%BE%D0%BD%D0%B0](#)

15. Коэффициент корреляции Мэтьюса [Электронный ресурс] – Режим доступа до ресурсу:

https://star-wiki.ru/wiki/Matthews_correlation_coefficient

16. Метод регуляризации Тихонова [Электронный ресурс] – Режим доступа до ресурсу:

https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%80%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8_%D0%A2%D0%B8%D1%85%D0%BE%D0%BD%D0%BE%D0%B2%D0%B0

ДОДАТКИ

Додаток А Лістинг програми

```

#!/usr/bin/env python
# -*- coding: utf-8 -*-
#
# GUI module generated by PAGE version 5.2
# in conjunction with Tcl version 8.6

import sys
import matplotlib.pyplot as plt
import pandas as pd

from marshmallow_dataclass import dataclass
from typing import List, Set, Dict, Tuple, Optional
import json

import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import MinMaxScaler
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.pipeline import Pipeline

import scipy.stats as st

try:
    import Tkinter as tk
except ImportError:
    import tkinter as tk

```

```

        try:
            import ttk
            py3 = False
        except ImportError:
            import tkinter.ttk as ttk
            py3 = True

from PIL import Image, ImageTk

import os.path
from tkinter import filedialog

@dataclass
class DataModel:
    Id: str
    Cost: float
    DriveExp: float
    FullName: str
    BadHabbits: str
    Birthday: str

@dataclass
class DataModelCollection:
    Collection: List[DataModel]

def vp_start_gui():
    '''Starting point when module is the main routine.'''
    global val, w, root
    global prog_location
    prog_call = sys.argv[0]
    prog_location = os.path.split(prog_call)[0]
    root = tk.Tk()
    top = Toplevel1(root)
    root.mainloop()

```

```

w = None

def create_Toplevel1(rt, *args, **kwargs):
    '''Starting point when module is imported by another module.
    Correct form of call: 'create_Toplevel1(root, *args, **kwargs)' .'''
    global w, w_win, root
    global prog_location
    prog_call = sys.argv[0]
    prog_location = os.path.split(prog_call)[0]
    #rt = root
    root = rt
    w = tk.Toplevel (root)
    top = Toplevel1 (w)
    return (w, top)

def destroy_Toplevel1():
    root.destroy()

class Toplevel1:
    def Calculate(self):
        self._Frame = []
        self.Listbox1.delete(0, 'end')
        self.Listbox1.insert(0, 'ID Fullname Birthday Decision')
c1 = ColumnTransformer(transformers=[('code1', OneHotEncoder(), [2])])
c2 = MinMaxScaler(feature_range=(-1, 1))

Model = LogisticRegression(penalty='l2', C=100, solver='lbfgs',
                           class_weight='balanced')

Model = Pipeline(steps=[('code1', c1), ('code2', c2), ('neur', Model)])

X1 = st.uniform.rvs(loc=0, scale=2, size=20)
X2 = st.uniform.rvs(loc=2, scale=30, size=180)
X = np.concatenate((X1, X2))

Z1 = st.uniform.rvs(loc=50000, scale=53000, size=20)

```

```

Z2 = st.uniform.rvs(loc=3000, scale=53000, size=180)
Z = np.concatenate((Z1, Z2))

W1 = ['No' for i in range(10)] + ['Yes' for i in range(10)]
W2 = ['No' for i in range(140)] + ['Yes' for i in range(40)]
W = np.array((W1 + W2))

X = np.array([list(X), list(Z), list(W)])
X = X.transpose()
y1 = [0 for i in range(20)]
y2 = [1 for i in range(180)]
y = np.array(y1 + y2)

Model.fit(X, y)

filename = filedialog.askopenfilename()
with open(filename, "r") as myfile:
    data = myfile.read().replace('\n', '')
    j_object = json.loads(data)
inputDataModel = DataModelCollection.Schema().load(j_object)
    counter = 1
    for i in inputDataModel.Collection:
        try:
prediction = Model.predict([[i.DriveExp, i.Cost, i.BadHabbits]])[0]
self._Frame.append([i.DriveExp, i.Cost, i.BadHabbits, prediction])
            if prediction == 1:
self.ListBox1.insert(counter, '{0} - {1} - {2} - {3}'.format(i.Id,
i.FullName,

str( np.datetime64(i.Birthday, 'ns'))[:10],

'Give policy' + "-" + str(

int(i.Cost) / 2 + int(1000 / (int(

i.DriveExp)+0.1))))))
                else:

```

```

self.ListBox1.insert(counter, '{0} - {1} - {2} - {3}'.format(i.Id,
                                                             i.FullName,

                                                             str( np.datetime64(i.Birthday, 'ns'))[:10],

                                                             'Not Give policy'))
    except Exception as err:
        err = str(err)
self.ListBox1.insert(counter, "Errorr " + err)
        counter + -1

def Andrews(self):
f = pd.DataFrame(np.array(self._Frame), columns= ['a','b','c','d'])
    def tr(x_):
        if x_=="Yes":
            return 1
        else:
            return 0

    def toint(x_):
        return float(x_)

    f['c'] = f['c'].map(tr)
    f['a'] = f['a'].map(toint)
    f['b'] = f['b'].map(toint)
    f['d'] = f['d'].map(toint)
pd.plotting.andrews_curves(f, class_column='d', colormap = 'Oranges')
    plt.show()

def Scatter(self):
f = pd.DataFrame(np.array(self._Frame), columns=['a', 'b', 'c', 'd'])
    def tr(x_):
        if x_=="Yes":
            return 1
        else:
            return 0

```

```

def toint(x_):
    return float(x_)

f['badhabbits'] = f['c'].map(tr)
f['DriveExp'] = f['a'].map(toint)
f['Cost'] = f['b'].map(toint)
f['d'] = f['d'].map(toint)
color_wheel = {1: "#0392cf",
               2: "#7bc043",
               3: "#ee4035"}
colors = f['d'].map(lambda x: color_wheel.get(x + 1))
pd.plotting.scatter_matrix(f[['DriveExp', 'Cost', 'badhabbits']], color=colors,
                           alpha=0.6, figsize=(15, 15), diagonal='kde')

plt.show()

pass

```

```

def __init__(self, top=None):
    '''This class configures and populates the toplevel window.
        top is the toplevel containing window.'''
    _bgcolor = '#d9d9d9' # X11 color: 'gray85'
    _fgcolor = '#000000' # X11 color: 'black'
    _compcolor = '#d9d9d9' # X11 color: 'gray85'
    _ana1color = '#d9d9d9' # X11 color: 'gray85'
    _ana2color = '#ececec' # Closest X11 color: 'gray92'
    font13 = "-family {Segoe UI Emoji} -size 17 -weight bold " \
             "-slant italic"

    top.geometry("928x549+649+59")
    top.minsize(176, 1)

```

```

        top.maxsize(1924, 1020)
        top.resizable(1, 1)
        top.title("Scoring car system")
        top.configure(background="#0000ff")

        self.Label1 = tk.Label(top)
self.Label1.place(relx=0.69, rely=0.492, height=185, width=259)
        self.Label1.configure(background="#0000ff")
        self.Label1.configure(disabledforeground="#a3a3a3")
        self.Label1.configure(foreground="#000000")
        photo_location = os.path.join(prog_location, "CARA.jpg")
        global _img0
        _img0 = ImageTk.PhotoImage(file=photo_location)
        self.Label1.configure(image=_img0)
        self.Label1.configure(text=''Label'')

        self.Label2 = tk.Label(top)
self.Label2.place(relx=0.032, rely=0.018, height=113, width=568)
        self.Label2.configure(activebackground="#f0f0f0f0f0f0")
        self.Label2.configure(background="#0000ff")
        self.Label2.configure(disabledforeground="#a3a3a3")
        self.Label2.configure(font=font13)
        self.Label2.configure(foreground="#ffff00")
        self.Label2.configure(highlightcolor="#0000ff")
        self.Label2.configure(text=''WELCOME TO SCORING SYSTEM'')

        self.Button1 = tk.Button(top)
self.Button1.place(relx=0.69, rely=0.237, height=82, width=258)
self.Button1.configure(command=lambda :Toplevel1.Calculate(self))
        self.Button1.configure(activebackground="#ecec")
        self.Button1.configure(activeforeground="#408080")
        self.Button1.configure(background="#408080")
        self.Button1.configure(disabledforeground="#a3a3a3")
        self.Button1.configure(foreground="#ffffff")
        self.Button1.configure(highlightbackground="#d9d9d9")
        self.Button1.configure(highlightcolor="black")
        self.Button1.configure(pady="0")

```

```

        self.Button1.configure(text='''ChooseFile''')

    self.Cansel = tk.Button(top, command=lambda : root.destroy())
    self.Cansel.place(relx=0.873, rely=0.055, height=42, width=88)
        self.Cansel.configure(activebackground="#ecec"
        self.Cansel.configure(activeforeground="#000000")
        self.Cansel.configure(background="#408080")
        self.Cansel.configure(disabledforeground="#a3a3a3")
        self.Cansel.configure(foreground="ffffff")
    self.Cansel.configure(highlightbackground="#d9d9d9")
        self.Cansel.configure(highlightcolor="black")
        self.Cansel.configure(pady="0")
        self.Cansel.configure(text='''Cancel''')

        self.Listbox1 = tk.Listbox(top)
        self.Listbox1.insert(0, 'ID Fullname Birthday Decision')
    self.Listbox1.place(relx=0.033, rely=0.242, relheight=0.69
                        , relwidth=0.612)
        self.Listbox1.configure(background="white")
    self.Listbox1.configure(disabledforeground="#a3a3a3")
        self.Listbox1.configure(font="TkFixedFont")
    self.Listbox1.configure(foreground="#000000")

self.menubar = tk.Menu(top, font="TkMenuFont", bg=_bgcolor, fg=_fgcolor)
    top.configure(menu = self.menubar)

        self.menubar.add_command(
            activebackground="#ecec",
            activeforeground="#000000",
            background="#d9d9d9",
            foreground="#000000",
            command=lambda: Toplevel1.Andrews(self),
            label="DrawAndrews")
    self.menubar.add_command(
        activebackground="#ecec",
        activeforeground="#000000",
        background="#d9d9d9",

```

```

        foreground="#000000",
        command=lambda: Toplevel1.Scatter(self),
        label="DrawScatter")

        self.Label3 = tk.Label(top)
self.Label3.place(relx=0.787, rely=0.838, height=37, width=166)
        self.Label3.configure(background="#0000ff")
        self.Label3.configure(disabledforeground="#a3a3a3")
        self.Label3.configure(foreground="#ffff00")

        self.Label4 = tk.Label(top)
self.Label4.place(relx=0.916, rely=0.893, height=37, width=48)
        self.Label4.configure(background="#0000ff")
        self.Label4.configure(disabledforeground="#a3a3a3")
        self.Label4.configure(foreground="#ffff00")
        self.Label4.configure(text='''2021''')

if __name__ == '__main__':
    vp_start_gui()

```

Додаток Б Ілюстративний матеріал

Дисертаційна робота
на здобуття ступеня магістра
за освітньо-професійною програмою
«Страхова та фінансова математика»
спеціальності 111 «Математика»
на тему:

«Математичні та програмні засоби системи автострахування»

Студент: Рясик І. С.

Група: ОМ-02мп

Керівник: доктор фіз.-мат. наук, професор Швець О. Ю.

1

Рисунок Б.1 – Слайд 1

Мета роботи

Метою даної дисертаційної роботи є мінімізація витрат та ризиків, підвищення надійності клієнтської бази даних для невеликих компаній у сфері автострахування.

2

Рисунок Б.2 – Слайд 2

Актуальність роботи

Впродовж останніх десятиліть важливою складовою глобальної економіки різних країн є сектор страхування. А однією із найпоширеніших сфер страхування у світі, в тому числі і в Україні є автострахування. Принцип компаній один - зібрати побільше страхових внесків та провести поменше виплат. Страхові компанії беруть приклад із банків і впроваджують в себе системи скорингу клієнта. Якщо в випадку банків перевіряється платоспроможність, то в секторі страхування шанси клієнта не потрапити в страхову подію і не понести величезні збитки для компанії. Такі системи аналізують історичні дані відомих клієнтів. Чим система краще по якості тим більше розвивається компанія у прибутку.

3

Рисунок Б.3 – Слайд 3

Постановка задачі

Для досягнення мети потрібно виконати такі завдання:

1. Ознайомитись з існуючими та сучасними скоринговими системами;
2. Проаналізувати закономірності між ознаками в тренувальних даних;
3. Реалізувати прикладне програмне забезпечення скорингової системи автострахування та проаналізувати роботу на тестовій групі.

Розроблена система повинна виконувати такі функції:

1. На основі характеристик осіб відомої вибірки даних побудувати найкращу апроксимацію за допомогою математичного алгоритму машинного навчання;
2. Надати можливість безпосередньо користувачу системи подивитись графічне відображення тестових даних та проаналізувати адекватність роботи або переглянути загальну поведінку заданої групи.

4

Рисунок Б.4 – Слайд 4

Список літератури

1. Страховой скоринг на основе кредитной истории [Электронный ресурс] – Режим доступа до ресурсу: <https://insuranceconference.ru/2015/files/presentations/VolkovA.pdf>
2. Что такое скоринг в автостраховании и как он может работать на клиента [Электронный ресурс] – Режим доступа до ресурсу: <https://www.ins-team.ru/news/scoring>
3. Диджитализация страховых продуктов: опыт использования Big Data [Электронный ресурс] Режим доступа до ресурсу: <https://bosfera.ru/pdf/17.%20%D0%9F%D0%B0%D0%B2%D0%B5%D0%BB%20%D0%91%D1%83%D1%82%D0%B5%D0%BD%D0%BA%D0%BE.pdf>
4. Клуб аккуратных водителей: как заработать на «умном» каско [Электронный ресурс] – Режим доступа до ресурсу: https://amp.rbc.ru/rbcnews/own_business/23/06/2016/576a62769a79474a1a1abcbl
5. Программа «КАСКО Смарт» от «АХА Страхование» позволяет клиентам экономить до 60% [Электронный ресурс] – Режим доступа до ресурсу: <https://forinsurer.com/news/19/06/12/36909>
6. Scatterplot Matrix [Электронный ресурс] Режим доступа до ресурсу: https://seaborn.pydata.org/examples/scatterplot_matrix.html
7. Матрица диаграммы рассеяния [Электронный ресурс] – Режим доступа до ресурсу: <https://pro.arcgis.com/ru/pro-app/help/analysis/geoprocessing/charts/scatter-plot-matrix.html>

5

Рисунок Б.5 – Слайд 5

Список літератури

8. Сеньо П. С.(2007). Теорія ймовірностей та математична статистика(вид. 2-ге, перероб. і доп.). Київ:Знання. с.446.
9. Andrews, David F. (1972). "Plots of High-Dimensional Data". International Biometric Society. 18(1): 125–136. JSTOR2528964
10. Logistic Regression (for dummies) [Электронный ресурс] – Режим доступа до ресурсу: <https://codesachin.wordpress.com/2015/08/16/logistic-regression-for-dummies/>
11. Логистическая регрессия (пример) [Электронный ресурс] – Режим доступа до ресурсу: http://www.machinelearning.ru/wiki/index.php?title=%D0%9B%D0%BE%D0%B3%D0%B8%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D1%8F_%28%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80%29
12. Comparing Classifiers [Электронный ресурс] – Режим доступа до ресурсу: <https://martin-thoma.com/comparing-classifiers/>
13. Логістична регресія [Электронный ресурс] – Режим доступа до ресурсу: https://uk.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D1%96%D1%81%D1%82%D0%B8%D1%87%D0%BD%D0%B0_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%96%D1%8F

6

Рисунок Б.6 – Слайд 6

Список літератури

14. Метод Ньютона - Рафсона [Електронний ресурс] – Режим доступу до ресурсу:
https://uk.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%9D%D1%8C%D1%8E%D1%82%D0%BE%D0%BD%D0%B0

15. Коэффициент корреляции Мэттьюса [Електронний ресурс] – Режим доступу до ресурсу:
https://star-wiki.ru/wiki/Matthews_correlation_coefficient

16. Метод регуляризации Тихонова [Електронний ресурс] – Режим доступу до ресурсу:
https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D1%80%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8_%D0%A2%D0%B8%D1%85%D0%BE%D0%BD%D0%BE%D0%B2%D0%B0

7

Рисунок Б.7 – Слайд 7

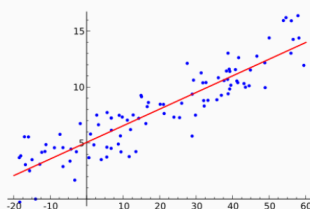
Екранні форми



8

Рисунок Б.8 – Слайд 8

Логістична регресія



Логістична регресія - це статистичний регресійний аналіз, який слід проводити, коли залежна змінна є дихотомічною (бінарною), тобто може набувати тільки двох значень (0 або 1). Як і всі регресійні аналізи, логістична регресія є прогнозним аналізом.

Нехай є деяка випадкова величина Y , що може набувати лише двох значень, які, як правило, позначаються цифрами 0 і 1. Нехай ця величина залежить від деякої множини пояснювальних змінних. Залежність Y від $x_1, \dots, x_n$ можна визначити ввівши додаткову змінну y^* , де $y^* = \theta^T x = \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n + \varepsilon$.

$$\text{Тоді: } Y = \begin{cases} 0, & y^* \leq 0 \\ 1, & y^* > 0 \end{cases}$$

При визначенні логістичної моделі стохастичний доданок ε вважається випадковою величиною з логістичним розподілом ймовірностей. Відповідно для певних конкретних значень змінних $x^* = x_1^*, \dots, x_n^*$ одержується відповідне значення y^* і ймовірність того, що $Y=1$ така:

$$p(Y=1) = p(y^* > 0) = p(\theta^T x^* + \varepsilon > 0) = p(\varepsilon > -\theta^T x^*) = p(\varepsilon \leq \theta^T x^*) = \Lambda(\theta^T x^*).$$

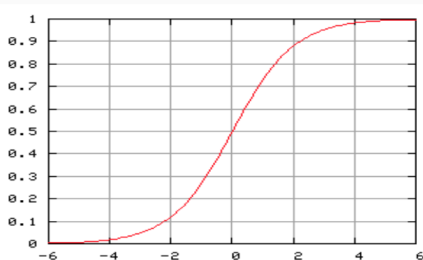
Передостання рівність випливає з симетричності логістичного розподілу

[13]

9

Рисунок Б.9 – Слайд 9

Логістична функція



Логістична функція - функція розподілу логістичного розподілу.

Позначається Λ . Логістична ф-я має наступний вигляд:

$$\Lambda(x) = \frac{e^x}{1 + e^x} = \frac{1}{1 + e^{-x}}$$

Таким чином для конкретного значення x^i випадкова величина Y^i , має розподіл Бернуллі: $Y^i \sim B(1, \Lambda(\theta^T x^i))$.

Логістична модель задовольняє наступній умові:

$$\ln \frac{p(1|x)}{1-p(1|x)} = \ln \frac{p(1|x)}{p(0|x)} = \theta_0 + \theta_1 x_1 + \dots + \theta_j x_j$$

$$\text{Оцінювання параметрів моделі: } \log L(\theta) = \sum_{i=1}^m \log \Pr\{Y = Y^{(i)} | x = x^{(i)}\} = \sum_{i=1}^m Y^{(i)} \log \Lambda(\theta^T x^{(i)}) + (1 - Y^{(i)}) \log(1 - \Lambda(\theta^T x^{(i)})). \quad [13]$$

$$\text{Задача зводиться до розв'язання системи рівнянь: } x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \quad [14]$$

10

Рисунок Б.10 – Слайд 10

Програмне забезпечення

Дана система може бути реалізована у багатьох мовах програмування.

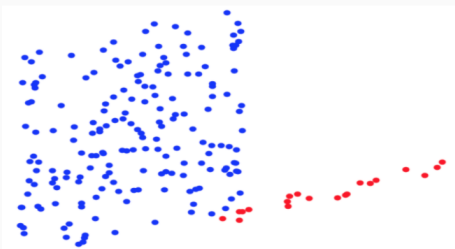
Найпоширеніші випадки використання:

1. R - мова для математичних розрахунків та аналізу даних. Реалізація у пакеті CRAN; 
2. Matlab - мова для математичних розрахунків у багатьох інженерних сферах від механіки до аналізу даних. Реалізація у пакеті Statistics and Machine learning Toolbox; 
3. Python - високорівнева програмна мова. Використовується у різних сферах. Реалізація у пакетах statsmodels, scikit-learn (дана реалізація буде використана). 

11

Рисунок Б.11 – Слайд 11

Опис вибірки. Вхідні дані



При розробці системи були використані дані із страхової компанії ТАС. Тренування алгоритму відбувається на вибірці із 20 реальних прикладах із страхових баз. Це був реєстр премій компанії ТАС 5 річної давнини. Так формується статистична картина, що репрезентативно відображає якість та ризик потенційних клієнтів на подібних компаніях.

У даній моделі були обрані наступні три значимі характеристики: стаж водія (у роках), вартість авто та наявність поганих звичок.

Було виявлено незбалансований набір, коли екземплярів одного класу набагато більше ніж іншого. Також замічено, що люди з дорогим авто, малим стажем водіння та поганими звичками є потенційно ризикованими клієнтами. В поточній вибірці спостерігається випадок лінійної сепарабельності класів, тому і була використана логістична регресія, як лінійний алгоритм.

12

Рисунок Б.12 – Слайд 12

Перевірка алгоритму

ID	Cost	BadHabbits	DriveExp	Expected	Calculated
1	63012	No	0	0	0
2	41003	Yes	2	0	0
3	47503	No	1	1	0
4	3081	Yes	28	1	1
5	4500	No	3	1	1
6	8400	No	11	1	1
7	9300	No	20	1	1
8	8652	No	7	1	1
9	11723	No	9	1	1
10	5555	No	11	1	1
11	13000	Yes	15	1	1
12	13889	No	18	1	1
13	16021	No	23	1	1
14	28000	No	19	1	1
15	39543	Yes	27	1	1
16	12435	No	5	1	1
17	38686	No	13	1	1
18	35235	No	6	1	1
19	39934	No	29	1	1
20	61256	No	10	1	1

Алгоритм був протестований та перевірений на тестовій групі.

В ній містяться явно позитивні, явно негативні та складні приклади.

Було досліджено результати роботи алгоритму на різних метриках.

Його слабкі та сильні сторони в рамках даної роботи.

13

Рисунок Б.13 – Слайд 13

Перевірка алгоритму. Оцінка якості

У протестованому прикладі точність результату має значення accuracy = 0.95, але для не збалансованих класів звичайна точність є зміщеною оцінкою класифікатора. Тому порахувавши наступні показники ми можемо побачити більш детальну картину:

Tr (true positive, безперечно ризиковий клієнт) = 16; Tn (true negative, не ризиковий клієнти, визначені алгоритмом) = 2;

Fp (false positive, ризикові клієнти, визначені алгоритмом, коли вони ними не являються) = 0;

Fn (false negative, ризикові клієнти, коректно визначені алгоритмом) = 1.

Порахуємо на точність і повноту:

precision = $Tr / (Tr + Fp) = 1$; recall = $Tr / (Tr + Fn) = 0.93$.

Обрахуємо коефіцієнт кореляції Метьюза [15]:

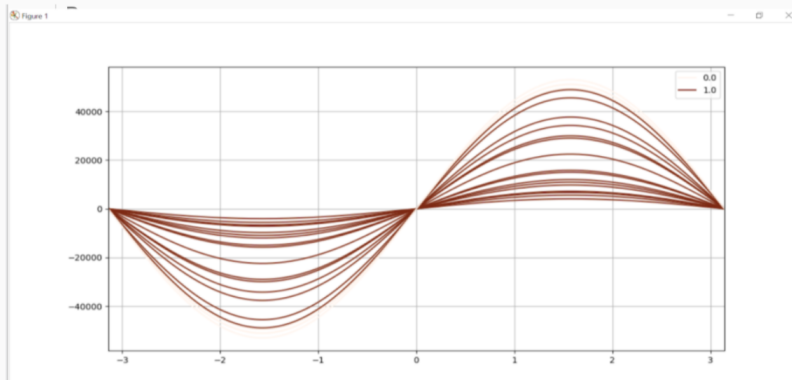
MCC = $(Tr * Tn - Fp * Fn) / \sqrt{(Tr + Fp)(Tr + Fn)(Tn + Fp)(Tn + Fn)} = 0.78$

Отже, ми бачимо що наша модель є доволі непоганою, але не досконалою - для благонадійних клієнтів модель іноді може помилятися і відносити їх до групи збиткових клієнтів.

14

Рисунок Б.14 – Слайд 14

Візуалізація роботи на тестовій вибірці



На рисунку зображено результат алгоритму на кривих Ендрюса. Одна крива - один клієнт. Клієнти, що належить до одного класу утворюють жмутик.

1, 0 - класи

1 - ризикований клієнт

0 - не ризикований клієнт

Чим жмутик темніше, тим більш ризикований клієнт.

15

Рисунок Б.15 – Слайд 15

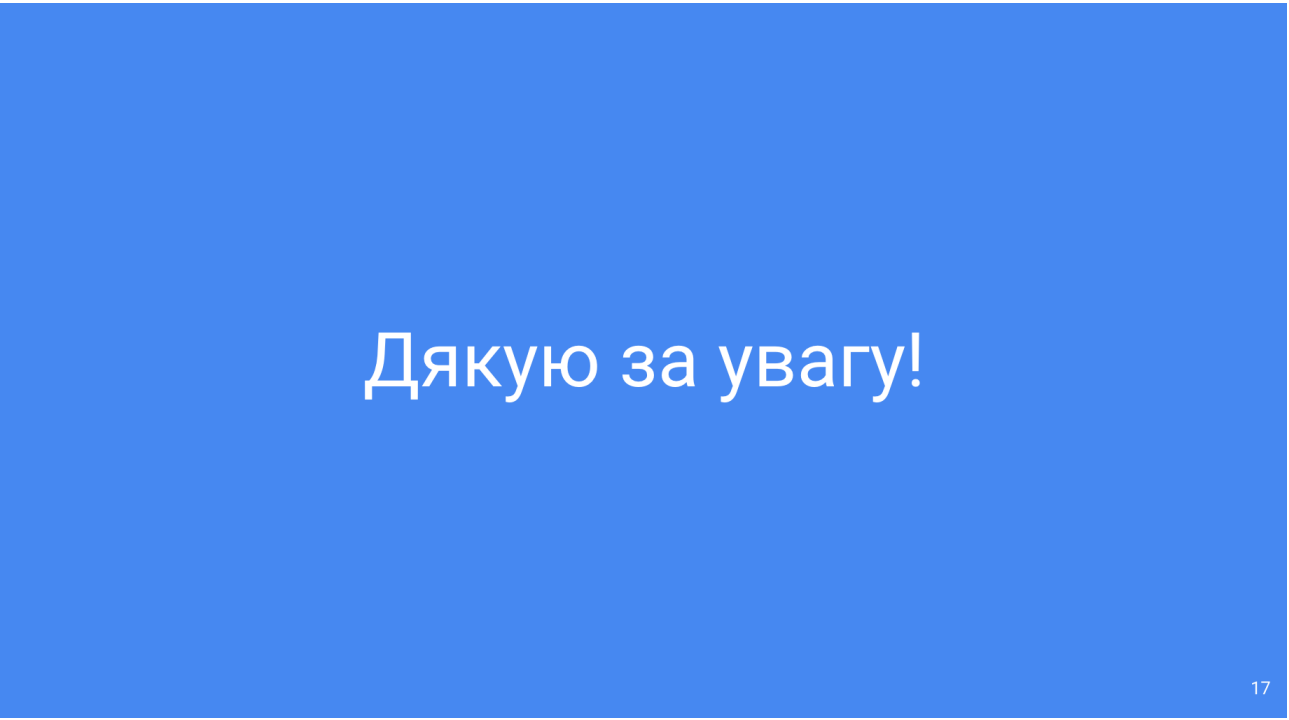
Висновки

В рамках дисертаційного проектування були виконані та описані наступні ключові пункти:

1. Введення в розвиток страхового скорингу в сучасному світі, розглянуті підходи для вирішення даного питання та приклади компаній, які розробили підходи в автострахованні;
2. Досліджені математичний апарат який є в даній тематиці, обрання та опис алгоритму для реалізації даної задачі;
3. Загальний опис розробки програмної реалізації для проектування додатку: програмні інструменти, загальна архітектура, дослідження та обґрунтування результатів роботи.

16

Рисунок Б.16 – Слайд 16



Дякую за увагу!

17

Рисунок Б.17 – Слайд 17