МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Львівський національний університет імені Івана Франка
Факультет електроніки та комп'ютерних технологій
Кафедра радіоелектронних і комп’ютерних систем

Допустити до захисту

Завідувач кафедри
_____ доцент Оленич І. Б.
«____»_____ 2023 р.

Курсова робота

Магістр
(освітній ступінь)

Прогнозування ризику виникнення депресії та розладів психічного здоров'я методами машинного навчання

Виконала:
Студент V курсу групи ФЕІ-54
спеціальності 121 – Комп’ютерні науки

_____ Новакова Є. С.

Науковий керівник:
_____ Гура В. Т.
«____»_____ 2023 р.

Львів — 2023

АНОТАЦІЯ

Дана кваліфікаційна робота присвячена застосуванню сучасних методів машинного навчання для передбачення депресії в людей, використовуючи виділені ознаки. Робота включає аналіз різних алгоритмів машинного навчання, таких як класифікація, глибоке навчання та ансамбль, з метою знаходження найефективнішого підходу.

В процесі дослідження були використані методи вилучення та попередньої обробки даних, навчання ML-класифікаторів, класифікація та виявлення ризику, а також оцінка ефективності моделей. Для аналізу алгоритмів використовувалась середа Colaboratory, а саме були розглянуті алгоритми, такі як Логістична регресія, KNN класифікація, Дерево рішень, Random forest та Stacking.

Результатом роботи є аналіз прогнозування з використанням різних алгоритмів, а також графіки, що демонструють ефективність та точність кожного з алгоритмів. Основною метою дослідження є виявлення найкращого алгоритму для даної задачі прогнозування депресії.

Робота складається з вступу, чотирьох розділів та висновку.

Курсова робота містить: 46 сторінок, 19 рисунків, 1 таблицю, список використаних джерел.

ANNOTATION

This qualification work is devoted to the application of modern machine learning methods for predicting depression in humans using selected features. The work includes the analysis of various machine learning algorithms such as classification, deep learning, and ensemble to find the most effective approach.

The research involved data extraction and preprocessing, ML classifier training, classification and risk detection, and model performance evaluation. The Colaboratory environment was used to analyze the algorithms, namely, algorithms such as Logistic Regression, KNN classification, Decision Tree, Random forest, and Stacking were considered.

The result of the work is an analysis of forecasting using different algorithms, as well as graphs demonstrating the efficiency and accuracy of each algorithm. The main goal of the study is to identify the best algorithm for the task of predicting depression.

The paper consists of an introduction, four chapters and a conclusion.

The term paper contains: 46 pages, 19 figures, 1 table, and a list of references.

ЗМІСТ

ВСТУП
РОЗДІЛ 1 ОПИС ПРОБЛЕМИ
1.1 Актуальність теми
1.2 Використання машинного навчання для вирішення проблеми
1.2 Мета та завдання дослідження
РОЗДІЛ 2 ОСНОВИ ПОБУДОВИ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ
2.1 Методи машинного навчання для прогнозування
2.1.1 Логістична регресія
2.1.2 KNN класифікація
2.1.3 Дерево рішень (Decision Tree)
2.1.4 Random Forest
2.1.5 Stacking
РОЗДІЛ 3 РОЗРОБКА МОДЕЛІ ПРОГНОЗУВАННЯ
3.1 Збір та підготовка даних
3.2 Дослідження даних
3.3 Scaling and fitting
3.4 Створення моделей
3.5 Нейронна модель та передбачення
ВИСНОВОК
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ
ЛІСТИНГ ПРОГРАМИ

ВСТУП

В останні роки технології знаходяться в постійному розвитку, що дозволяє працювати та знаходити роботу необмежено в часі для людей у всьому світі. Однак, цей швидкий розвиток супроводжується напруженим графіком роботи, який може негативно впливати на психічне здоров'я людей. Особливо в контексті пандемії психічне здоров'я стало однією з найактуальніших проблем, а стрес, самотність і депресія зростали протягом останнього року.

Діагноз психічного здоров'я є складним завданням, оскільки люди не завжди готові говорити про свої проблеми або не можуть ясно виражати свої почуття та емоції. У цьому контексті машинне навчання стає незамінним інструментом для діагностики різних психічних розладів. Воно надає лікарям та фахівцям з психічного здоров'я платформу для аналізу великого обсягу даних про пацієнтів та створення персоналізованого підходу до лікування відповідно до медичної ситуації кожного пацієнта.

Метою даної роботи є прогнозування ризику виникнення депресії та розладів психічного здоров'я з використанням різних моделей машинного навчання. Дослідження спрямоване на визначення найточніших моделей машинного навчання для класифікації даних і прогнозування ризику виникнення психічних розладів.

У роботі буде розглянута загальна модель діагностики депресії, що включає в себе вилучення та попередню обробку даних, навчання ML-класифікаторів, класифікацію та виявлення ризику, а також оцінку ефективності. Також буде проведений огляд різних алгоритмів машинного навчання для діагностики депресії, включаючи класифікацію, глибоке навчання та ансамбль.

Вивчення та аналіз цих моделей допоможуть отримати більше уявлення про те, як машинне навчання може бути використане для прогнозування ризику виникнення депресії та розладів психічного здоров'я. Результати дослідження можуть мати важливе значення для розробки ефективних стратегій попередження та управління психічним здоров'ям на робочому місці, зокрема для визначення ризикових груп та розробки персоналізованих підходів до лікування та підтримки.

Отже, розглядаючи значення машинного навчання та його потенціал у сфері психічного здоров'я, проведемо детальний аналіз моделей та алгоритмів машинного навчання для прогнозування ризику виникнення депресії та розладів психічного здоров'я.

РОЗДІЛ 1 ОПИС ПРОБЛЕМИ

1.1 Актуальність теми

Сучасний спосіб життя має психологічний вплив на свідомість людей, що спричиняє емоційний стрес і депресію. Депресія - це переважаючий психічний розлад, що впливає на мислення та розумовий розвиток людини. За даними ВООЗ, приблизно 1 мільярд людей мають психічні розлади і понад 300 мільйонів людей у світі страждають від депресії. Депресія переважає в суїцидальних думках людини. Щорічно близько 800 000 людей вчиняють самогубство. Тому це вимагає комплексного реагування для вирішення проблеми психічного здоров'я. Консультування та психологічна терапія можуть допомогти в боротьбі з депресією.

Деякі з основних розладів психічного здоров'я, такі як хронічні захворювання, біполярний розлад і шизофренія, не виникають раптово з нізвідки; вони часто розвиваються з часом і дають симптоми, які можна розпізнати на ранніх стадіях. Таких розладів можна уникнути або контролювати їх більш успішно. Якщо виявити аномальні психічні стани на ранніх стадіях захворювання, можна надати додаткове лікування та догляд. Отже, судження про психічний стан людини на основі її зовнішнього вигляду чи поведінки - це складна психологічна наука, яку ще належить механізувати. Хоча скринінгові тести існують, через часові та фінансові обмеження вони не можуть бути застосовані до великих груп населення. Крім того, процедури, засновані на діагностиці, мають непередбачуваний наслідок - відлякування нездорових людей від участі. Як наслідок, психологічні проблеми часто залишаються непоміченими або нелікованими.

Тривога і депресія є серйозними проблемами громадського здоров'я, від яких страждають люди в усьому світі. Вони зачіпають людей різного віку, від дітей до людей похилого віку, включаючи як чоловіків, так і жінок. Тривожні та депресивні розлади мають широкий спектр впливу на здоров'я та самопочуття. Вони спричиняють різноманітні соматичні симптоми, такі як гастрит, кислотний рефлюкс, серцебиття, безсоння або гіперсоння, тремор, значну втрату або збільшення ваги, а також різні психосоціальні прояви, такі як пригнічений настрій, соціальна замкнутість, зниження продуктивності на робочому місці, суїцидальні думки або спроби суїциду та відсутність концентрації. Депресія і тривога є значними факторами ризику для цілого ряду інших розладів способу життя, таких як ішемічна хвороба серця, гіпертонія, діабет, ненавмисні та навмисні нещасні випадки.

Як наслідок, люди втрачають доступ до економічних і соціальних можливостей, що призводить до низької якості життя. Економічна напруга є масовим і часто не піддається кількісній оцінці проявом, що сприяє утворенню порочного кола бідності та поганого здоров'я.

Машинне навчання стало надійним інструментом для аналізу цих даних. Машинне навчання - це застосування передових імовірнісних і статистичних методів для створення комп'ютерів, які можуть самостійно навчатися на основі даних.

У таких галузях, як біоінформатика, ML сприяв значному прогресу, дозволяючи проводити швидкий і масштабований аналіз складних даних. Дані про психічне здоров'я також досліджуються за допомогою подібних аналітичних інструментів, що має потенціал для покращення результатів лікування пацієнтів, а також для кращого розуміння психологічних захворювань та їх лікування.

1.2 Використання машинного навчання для вирішення проблеми

Машинне навчання (ML) спрямоване на створення алгоритмів, які мають здатність навчатися сприймати складні закономірності. Ця здатність допомагає знаходити рішення для нових проблем, використовуючи попередні дані та рішення. Застосування методів ML в охороні здоров'я виявилося прагматичним, оскільки вони можуть обробляти величезну кількість різнорідних даних і надавати ефективні клінічні рекомендації. Підходи на основі ML забезпечують ефективне розуміння психічних станів і допомагають фахівцям з психічного здоров'я у прийнятті прогностичних рішень. Методи ML сприяють прогнозуванню та діагностиці в галузі охорони здоров'я, генеруючи інформацію з неструктурованих медичних даних. Результати прогнозування допомагають виявити захворювання з високим ризиком у пацієнтів для раннього лікування.

При психічних розладах методи ML допомагають оцінити потенційні поведінкові біомаркери, щоб допомогти фахівцям у галузі охорони здоров'я прогнозувати непередбачувані випадки психічних розладів і призначати ефективні результати лікування. Ці методи допомагають візуалізувати та інтерпретувати складні медичні дані. Візуалізація допомагає розробити ефективну гіпотезу щодо діагностики психічних розладів.

Склад симптомів, пов'язаних з психічними розладами, такими як депресія, можна легко виявити і передбачити, використовуючи методи ML. Тому діагностичний підхід на основі ML видається ефективним вибором для прогностичного аналізу. У секторі охорони здоров'я основні сфери, що використовуються для вилучення спостережень, пов'язаних з психічними розладами, за допомогою ML, можна класифікувати як сенсори, текст, структуровані дані та мультимодальні технологічні

взаємодії. Дані з датчиків можна аналізувати за допомогою мобільних телефонів та аудіосигналів. Текстові джерела можуть бути вилучені через соціальні мережі, текстові повідомлення та клінічні записи.

Структуровані дані - це дані, отримані зі стандартних скринінгових шкал, опитувальників та медичних карток. Мультимодальні технологічні взаємодії включають дані про взаємодію людини з повсякденним технологічним обладнанням, роботами та віртуальними агентами. Підходи ML можуть бути використані для діагностики психічних розладів. Більшість досліджень аналізують дані Twitter та дані сенсорів мобільних пристроїв для виявлення розладів настрою. Аналіз сенсорних даних, клінічних медичних записів і даних текстових повідомлень може допомогти передбачити тяжкість психічних розладів і суїцидальну поведінку. Були проведені різні дослідження, покликані допомогти медичним фахівцям у виявленні депресії та багатьох інших психічних розладів. Психічні розлади охоплюють широкий спектр психічних захворювань.

1.2 Мета та завдання дослідження

Метою дослідження є оцінка ефективності восьми різних алгоритмів машинного навчання для класифікації набору даних, пов'язаних з проблемами психічного здоров'я, за кількома критеріями точності. Дослідження спрямоване на з'ясування, які алгоритми найточніше класифікують ці дані та можуть бути корисні для прогнозування ризику психічних розладів.

Завдання дослідження включають наступні кроки:

1. Збір та підготовка даних: Зібрати відповідні дані про особисті, психологічні, соціальні та інші фактори, що можуть бути пов'язані з ризиком депресії та розладів психічного здоров'я. Виконати

очищення, обробку та підготовку цих даних для використання в моделі машинного навчання.

2. Вибір моделі та алгоритму машинного навчання: Вибрати підходящу модель машинного навчання, таку як логістична регресія, дерева рішень чи нейронні мережі, та відповідний алгоритм для прогнозування ризику депресії та розладів психічного здоров'я.
3. Побудова моделі: Навчити модель на підготовлених даних та зробити її прогнози щодо ризику виникнення депресії та розладів психічного здоров'я.
4. Оцінка та покращення моделі: Оцінити ефективність моделі за допомогою метрик, таких як точність, чутливість. Врахувати можливості покращення моделі, включаючи вибір оптимальних гіперпараметрів, застосування ансамблевих методів або використання інших алгоритмів машинного навчання.
5. Валідація та перевірка моделі: Перевірити модель на незалежних тестових даних для оцінки її загальної здатності до прогнозування ризику депресії та розладів психічного здоров'я. Зробити порівняння з існуючими методами та дослідженнями у цій області.

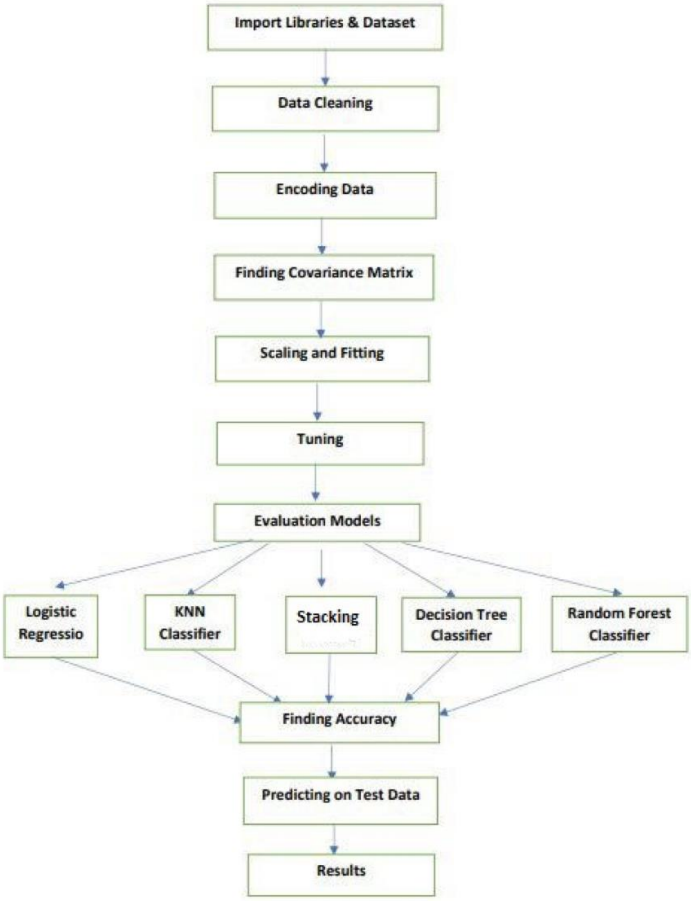

Рис. 1 Кроки створення моделі

РОЗДІЛ 2 ОСНОВИ ПОБУДОВИ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ

2.1 Методи машинного навчання для прогнозування

Методи машинного навчання для прогнозування відіграють важливу роль в багатьох сферах, включаючи фінанси, медицину, кліматологію, маркетинг, логістику та багато інших. Ці методи дозволяють аналізувати великі обсяги даних, виявляти складні зв'язки та шаблони і робити прогнози на основі цих відкриттів.

Одним з найпоширеніших методів машинного навчання є навчання з учителем. У цьому методі модель навчається на основі вхідних даних і відповідних міток (ярликів), що дозволяє їй зробити прогнози для нових невідомих прикладів. У цьому контексті використовуються алгоритми, такі як лінійна регресія, рішучі дерева, метод опорних векторів (SVM) та нейронні мережі. Ці алгоритми засновані на різних математичних моделях та підходах і можуть бути використані для прогнозування числових значень (наприклад, цін на акції) або класифікації об'єктів на кілька категорій (наприклад, визнання образів).

Інший метод - навчання без учителя - використовується для виявлення складних структур та патернів у навчальних даних без наявності міток. Алгоритми кластеризації, такі як k-середніх, алгоритми зменшення розмірності, наприклад, метод головних компонентів (PCA), та асоціативні правила є прикладами методів навчання без учителя. Вони дозволяють групувати схожі об'єкти разом, знаходити взаємозв'язки між змінними та здійснювати зведення до меншої кількості репрезентативних змінних.

Крім того, глибоке навчання є однією з найбільш просунутих галузей машинного навчання, яка використовує нейронні мережі з

багатьма шарами для вирішення складних завдань прогнозування. Використовуючи алгоритми, такі як згорткові нейронні мережі (CNN) та рекурентні нейронні мережі (RNN), глибоке навчання може аналізувати великі обсяги даних з високою точністю, зокрема у сферах обробки зображень, обробки природної мови та голосового розпізнавання.

Методи машинного навчання для прогнозування постійно розвиваються і вдосконалюються. Використання цих методів дозволяє покращити точність прогнозів, здійснювати раннє виявлення важливих трендів та патернів, а також розробляти персоналізовані рекомендації та стратегії для різних галузей та ситуацій. Це робить їх незамінним інструментом для прийняття рішень і покращення результатів у багатьох сферах діяльності.

2.1.1 Логістична регресія

Логістична регресія є одним із найпоширеніших методів у сфері машинного навчання, особливо в задачах класифікації. Вона використовується для прогнозування категоріальних змінних, де вихідні дані можуть бути класифіковані на два або більше класів.

Логістична регресія також може бути використана для прогнозування ймовірності подій або ризиків. Наприклад, в медичній діагностиці вона може бути використана для прогнозування ймовірності розвитку певного захворювання на основі медичних ознак пацієнта.

Однією з переваг логістичної регресії є її інтерпретованість. Вагові коефіцієнти моделі можуть бути проаналізовані, що дозволяє зрозуміти вплив кожної вхідної ознаки на прогнозований результат. Це може бути корисно для виявлення ключових факторів, що впливають на результат, і для прийняття обґрунтованих рішень.

Основна ідея логістичної регресії полягає у використанні логістичної функції для визначення ймовірності належності об'єкта до певного класу. Логістична функція приймає вхідні дані і видає значення між 0 і 1, що можна інтерпретувати як ймовірність. Зазвичай використовується логістична функція sigmoid:

$$sigmoid(z) = 1 / (1 + exp(-z))$$

де z - лінійна комбінація вхідних ознак і вагових коефіцієнтів моделі.

Процес навчання логістичної регресії полягає в знаходженні оптимальних значень вагових коефіцієнтів, які найкраще відповідають навчальним даним. Це виконується шляхом мінімізації функції втрат, такої як логарифмічна функція втрати (loss-function):

$$J(w) = -(1/m) * \Sigma [y * log(h(x)) + (1-y) * log(1 - h(x))]$$

де

w - вектор вагових коефіцієнтів,

m - кількість навчальних прикладів,

y - цільова змінна,

h(x) - значення, прогнозоване моделлю для вхідного прикладу x.

Оптимізація вагових коефіцієнтів здійснюється за допомогою градієнтного спуску, який шукає мінімум функції втрат шляхом ітеративного оновлення вагових коефіцієнтів.

Логістична регресія має кілька важливих переваг. Вона є відносно простим інтерпретованим методом, добре працює з великими наборами даних і може бути застосована для багатьох типів класифікаційних задач. Крім того, вона може бути розширена на багатокласову класифікацію за допомогою стратегій, таких як один проти решти (one-vs-rest) або один проти одного (one-vs-one).

Однак, варто враховувати, що логістична регресія має свої обмеження. Вона припускає лінійну роздільну здатність даних і може бути вразливою до викидів або недостатньо інформативних ознак. Крім того,

вона може мати проблему з узгодженістю, якщо вхідні ознаки сильно корельовані.

У загальному контексті логістична регресія є потужним інструментом прогнозування для багатьох класифікаційних завдань. Її простота, ефективність та інтерпретованість роблять її популярним вибором для багатьох дослідників і практиків у сфері машинного навчання.

2.1.2 KNN класифікація

K-Nearest Neighbors (KNN) є одним з найпростіших і популярних методів машинного навчання для класифікації і прогнозування. Він використовується для вирішення задач, де необхідно визначити клас нового об'єкта на основі схожості з його найближчими сусідами.

Основна ідея KNN полягає в тому, що об'єкт класифікується на основі класів його k найближчих сусідів. Коли новий об'єкт потрібно класифікувати, він порівнюється з усіма іншими об'єктами у тренувальному наборі даних за допомогою метрики відстані, такої як Евклідова відстань чи Манхеттенська відстань. Класифікація нового об'єкта здійснюється шляхом голосування ближніх сусідів - більшість класів з найближчих сусідів визначає приналежність об'єкта до класу.

KNN використовується в багатьох галузях, зокрема в медицині, фінансах, маркетингу та рекомендаційних системах. Він особливо ефективний у випадках, коли дані мають просту структуру, низьку кількість ознак і коли схожі об'єкти відносно одного класу тенденційно згруповані.

Перевага KNN полягає в його незалежності від функціональної форми даних і його відносно простій реалізації. Також він може використовуватися як базовий метод для порівняння з більш складними алгоритмами. Однак, він може мати обмеження в ефективності з обробкою

великих обсягів даних і може бути чутливим до шуму та нерелевантних ознак.

Для успішного застосування KNN важливо правильно встановити значення параметра k. Занадто мале значення k може призвести до перенавчання моделі, тоді як занадто велике значення k може призвести до упущення локальних залежностей. Вибір оптимального значення k вимагає експериментування та крос-валідації моделі.

У підсумку, KNN є простим, ефективним і інтуїтивно зрозумілим методом класифікації для прогнозування. Він широко використовується через свою простоту і здатність враховувати схожість між об'єктами. Правильне встановлення значення k і обробка даних можуть покращити результати KNN і зробити його потужним інструментом для прогнозування у багатьох галузях.

2.1.3 Дерево рішень (Decision Tree)

Дерево рішень є одним з основних методів машинного навчання, який використовується для класифікації та прогнозування. Цей метод використовує структуру дерева, де кожен вузол представляє рішення на основі певного набору ознак, а гілки відповідають можливим результатам цього рішення. Кожен листовий вузол дерева представляє окремий клас або прогнозоване значення.

Процес побудови дерева рішень полягає у рекурсивному розбитті тренувального набору даних на піднабори за допомогою різних критеріїв, таких як індекс Джині, ентропія або помилка класифікації. На кожному кроці алгоритм вибирає найкращий розбиток, який максимізує чистоту або інформаційний приріст. Процес продовжується досягнення певної умови зупинки, такої як досягнення максимальної глибини дерева або мінімальної кількості об'єктів у листових вузлах.

Однією з переваг дерева рішень є його інтерпретованість. Візуалізація дерева рішень дозволяє легко зрозуміти логіку, за якою приймаються рішення. Він також добре працює з числовими та категоріальними ознаками без потреби у попередній обробці даних. Крім того, дерева рішень можуть бути використані для вирішення задач класифікації та регресії.

Однак, дерева рішень також мають свої обмеження. Вони можуть бути схильні до перенавчання, особливо якщо дозволяють складнішу структуру дерева. Це може привести до низької загальної здатності узагальнення моделі на нові дані. Для запобігання перенавчанню можна використовувати методи обрізки дерева, регуляризацію або ансамблювання дерев.

У випадку задач прогнозування, дерева рішень можуть бути використані для класифікації об'єктів на основі набору ознак. Наприклад, в медицині дерева рішень можуть допомагати в класифікації пацієнтів на групи з різним ризиком захворювань або в прогнозуванні ефективності лікування на основі показників пацієнта.

Загалом, дерева рішень є потужним інструментом для прогнозування, який може бути використаний в різних галузях. Вони прості у розумінні, інтерпретації та реалізації. Однак, для досягнення оптимальних результатів важливо уважно налаштувати параметри моделі та контролювати перенавчання.

2.1.4 Random Forest

Random Forest (випадковий ліс) є одним з найпопулярніших методів машинного навчання, використовуваних для прогнозування в багатьох областях. Він є типом ансамблю дерев рішень, де кілька дерев приймають

рішення незалежно один від одного, і результати об'єднуються для отримання кінцевого прогнозу.

Random Forest використовує поняття "bagging" (випадковий вибір замість "bootstrapping") для створення набору випадкових підвибірок тренувальних даних замість використання всіх доступних даних. Кожне дерево будується на основі цих випадкових підвибірок і випадково вибирає лише деякі ознаки для розгляду при кожному розбитті. Це дозволяє випадковому лісу бути різноманітним і зменшує його схильність до перенавчання.

Процес побудови Random Forest складається з наступних етапів:

1. Вибір випадкових підвибірок даних з тренувального набору.
2. Побудова дерева рішень для кожної випадкової підвибірки шляхом рекурсивного розбиття даних на основі вибраних ознак та критеріїв якості, таких як ентропія або індекс Джині.
3. Повторення пунктів 1-2 декілька разів для отримання кількох дерев.
4. Комбінування результатів дерев узагальненням для отримання кінцевого прогнозу.

Однією з основних переваг Random Forest є його висока точність прогнозування. Він може добре працювати з великими наборами даних, включаючи дані з числовими та категоріальними ознаками. Він також здатний виявляти та враховувати важливі ознаки у наборі даних.

Крім того, Random Forest може оцінювати важливість ознак, що дозволяє визначити, які ознаки найбільше сприяють прогнозуванню. Це дає змогу зрозуміти, які фактори мають велике вплив на модель і які можуть бути ігноровані.

Random Forest також ефективний при роботі з пропущеними даними та викидами, оскільки він може обробляти ці ситуації без значного впливу на точність моделі.

Застосування Random Forest включає широкий спектр задач прогнозування, включаючи класифікацію, регресію, кластеризацію та виявлення аномалій. Він знаходить своє застосування в багатьох галузях, таких як медицина, фінанси, маркетинг, біологія та багато інших.

В цілому, Random Forest є потужним і гнучким методом машинного навчання, який надає високу точність прогнозування та може використовуватися в різних сферах для розв'язання різноманітних задач прогнозування.

2.1.5 Stacking

Stacking є одним з методів ансамблювання в машинному навчанні, який використовується для покращення прогнозів моделей шляхом комбінування їх результатів. Він базується на ідеї використання навчених моделей як "метамоделей", які комбінують прогнози вхідних моделей, що дозволяє отримати більш точний та надійний прогноз.

Основна ідея стекінгу полягає в тому, що замість того, щоб використовувати лише одну модель для прогнозування, ми використовуємо кілька базових моделей. Спочатку тренується набір базових моделей на тренувальних даних. Потім на цих моделях застосовується валідація з перехресною перевіркою (cross-validation), де дані розділяються на кілька фолдів, і кожен фолд використовується як тестовий набір для оцінки прогнозів.

Після цього створюється "метадані" для навчання метамоделі. Метадані - це прогнози базових моделей на валідаційному наборі даних. Ці прогнози використовуються як вхідні ознаки для метамоделі, разом з

вихідними значеннями, які потрібно прогнозувати. Метамодель навчається на цих метаданих, щоб зробити фінальний прогноз.

Один з ключових аспектів стекінгу полягає в тому, що базові моделі повинні бути різноманітними, тобто використовувати різні алгоритми та/або різні набори ознак. Це дозволяє отримати різноманітні та незалежні прогнози, що покращує роботу метамоделі.

Переваги стекінгу включають:

1. Висока точність: стекінг може покращити точність прогнозування в порівнянні з окремими моделями.
2. Гнучкість: можна використовувати різні типи моделей як базові та метамодель, що дозволяє адаптувати стекінг до конкретної задачі.
3. Універсальність: стекінг може бути застосований до різних типів задач прогнозування, включаючи класифікацію, регресію та кластеризацію.

Однак, стекінг має свої виклики та обмеження:

1. Складність: стекінг вимагає більшої обчислювальної потужності та часу для тренування моделей та метамоделі.
2. Ризик перенавчання: можлива схильність до перенавчання, особливо якщо набір даних є обмеженим.
3. Вразливість до помилок базових моделей: якщо базові моделі виконують погано, це може негативно позначитись на роботі метамоделі.

Взагалі, стекінг є потужним інструментом для покращення прогнозів моделей. Він дозволяє комбінувати сильні сторони різних моделей для отримання кращих результатів прогнозування. Застосування стекінгу може бути особливо корисним у випадках, коли точність прогнозу має

велике значення, або коли базові моделі мають різні особливості та сильні сторони.

РОЗДІЛ 3 РОЗРОБКА МОДЕЛІ ПРОГНОЗУВАННЯ

3.1 Збір та підготовка даних

Етап збору даних - це процес отримання вихідних даних, необхідних для побудови моделі передбачення. На цьому етапі важливо визначити, яка саме інформація потрібна для вирішення конкретної задачі передбачення. Потім необхідно ідентифікувати джерела даних, з яких можна отримати необхідну інформацію. Це можуть бути бази даних, веб-сторінки, API або файли даних. Якщо дані знаходяться у вигляді файлів, їх необхідно завантажити та зберегти у відповідному форматі, наприклад, CSV, JSON або SQL.

Після завантаження даних важливо розібратися з їх структурою та характеристиками. Розглянути атрибути, їх типи та інформацію, яку вони містять. Також слід перевірити якість даних, виявити та виправити можливі пропуски, аномалії, помилки або дублікати. Для цього можна використовувати методи очищення даних, заповнення пропущених значень, видалення дублікатів або нормалізацію даних.

Якщо вихідні дані містять багато атрибутів, можна вибрати лише ті, які найбільше впливають на задачу передбачення. Це може включати виключення неінформативних ознак або використання методів відбору ознак, таких як кореляційний аналіз, важливість ознак за алгоритмами машинного навчання або залучення експертів.

Загальний підхід до збору даних полягає в тому, щоб отримати необхідну інформацію з відповідних джерел, перевірити їх якість та структуру, а також вибрати найбільш важливі ознаки для вашої моделі передбачення. Це забезпечить вам чисті, якісні та релевантні дані, які можуть суттєво покращити ефективність вашої моделі передбачення.

Для виконання роботи я використовую дані в форматі csv. Який містить такі стовпці даних (Рис. 2)

```
RangeIndex: 1259 entries, 0 to 1258
Data columns (total 27 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Timestamp                  1259 non-null   object
 1   Age                        1259 non-null   int64
 2   Gender                     1259 non-null   object
 3   Country                    1259 non-null   object
 4   state                      744 non-null    object
 5   self_employed              1241 non-null   object
 6   family_history             1259 non-null   object
 7   treatment                  1259 non-null   object
 8   work_interfere             995 non-null    object
 9   no_employees               1259 non-null   object
 10  remote_work                1259 non-null   object
 11  tech_company               1259 non-null   object
 12  benefits                   1259 non-null   object
 13  care_options               1259 non-null   object
 14  wellness_program           1259 non-null   object
 15  seek_help                  1259 non-null   object
 16  anonymity                  1259 non-null   object
 17  leave                      1259 non-null   object
 18  mental_health_consequence  1259 non-null   object
 19  phys_health_consequence    1259 non-null   object
 20  coworkers                  1259 non-null   object
 21  supervisor                 1259 non-null   object
 22  mental_health_interview    1259 non-null   object
 23  phys_health_interview      1259 non-null   object
 24  mental_vs_physical         1259 non-null   object
 25  obs_consequence            1259 non-null   object
 26  comments                   164 non-null    object
```

Рис. 2 Структура даних

Наступним кроком після збору йде очищення даних. Етап очищення даних є критичним кроком у процесі підготовки даних для моделі передбачення. На цьому етапі, ми проводимо ретельний аналіз набору даних з метою виявлення та виправлення проблем, які можуть вплинути на якість та надійність наших результатів.

Перший крок - виявлення пропущених значень. Ми перевіряємо кожну змінну в наборі даних, щоб виявити, чи містить вона пропущені значення. Якщо такі значення є, ми вирішуємо, як їх обробити - видаляти рядки з пропущеними значеннями, заповнювати їх середніми або медіанними значеннями або застосовувати інші стратегії заповнення.

Другий крок - виявлення аномалій. Ми аналізуємо розподіл значень кожної змінної та шукаємо значення, які відхиляються від типового розподілу. Це можуть бути викиди або значення, які лежать за межами допустимого діапазону. Виявлені аномалії можуть бути видалені або оброблені в залежності від контексту завдання.

Третій крок - виправлення помилок у даних. Ми перевіряємо дані на наявність помилок, таких як орфографічні або синтаксичні помилки, некоректні значення, які не відповідають контексту. Ці помилки можуть бути виправлені або замінені на правильні значення.

Четвертий крок - видалення дублікатів. Ми перевіряємо набір даних на наявність дублікатів - рядків, що мають ідентичні значення у всіх змінних. Дублікати можуть спотворювати результати моделі, тому вони повинні бути видалені для забезпечення точності.

П'ятий крок - нормалізація та шкалювання даних. Залежно від використовуваних моделей, може бути потрібно нормалізувати дані до одного масштабу або шкалювати їх до конкретного діапазону значень. Це забезпечить стабільність та рівномірність обробки даних моделлю.

Шостий крок - кодування категоріальних змінних. Якщо набір даних містить категоріальні змінні, такі як рівень освіти або тип продукту, ці змінні потрібно закодувати у числовий формат, який може бути зрозумілий для моделі. Це можна зробити за допомогою технік, таких як one-hot encoding або label encoding.

Сьомий крок - вибір ознак. На цьому етапі важливо вибрати підмножину ознак, які найбільше впливають на цільову змінну. Це допоможе знизити розмірність даних та поліпшити ефективність моделі.

Восьмий крок - розділення набору даних на навчальну та тестову вибірку. Для оцінки ефективності моделі необхідно розділити дані на дві частини: навчальну вибірку, на якій модель буде тренуватися, і тестову вибірку, яка буде використовуватись для перевірки точності передбачень моделі.

В цілому, етап очищення даних перед побудовою моделі передбачення вимагає від нас уважності та систематичного підходу. Це допомагає забезпечити якість, надійність та ефективність моделі, а також забезпечує нам достовірні результати інтерпретації та прийняття рішень.

Отже використовуючи описані методи, перевірю датасет на наявність пропущених даних (Рис. 3)

```
#missing data
total = train_df.isnull().sum().sort_values(ascending=False)
percent = (train_df.isnull().sum()/train_df.isnull().count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
missing_data.head(20)
print(missing_data)
```

```
                           Total   Percent
comments                    1095  0.869738
state                        515  0.409055
work_interfere               264  0.209690
self_employed                 18  0.014297
seek_help                      0  0.000000
obs_consequence                0  0.000000
mental_vs_physical             0  0.000000
phys_health_interview          0  0.000000
mental_health_interview        0  0.000000
supervisor                     0  0.000000
coworkers                      0  0.000000
phys_health_consequence        0  0.000000
mental_health_consequence      0  0.000000
leave                          0  0.000000
anonymity                      0  0.000000
Timestamp                      0  0.000000
wellness_program               0  0.000000
Age                            0  0.000000
benefits                       0  0.000000
tech_company                   0  0.000000
remote_work                    0  0.000000
no_employees                   0  0.000000
treatment                      0  0.000000
family_history                 0  0.000000
Country                        0  0.000000
Gender                         0  0.000000
care_options                   0  0.000000
```

Рис. 3 Пропущені дані

Отже, виконаю очистку даних заміню наявні значення NaN (Рис. 4)

Cleaning NaN

```
[19] # Assign default values for each data type
     defaultInt = 0
     defaultString = 'NaN'
     defaultFloat = 0.0

     # Create lists by data tpe
     intFeatures = ['Age']
     stringFeatures = ['Gender', 'Country', 'self_employed', 'family_history', 'treatment', 'work_interfere',
                       'no_employees', 'remote_work', 'tech_company', 'anonymity', 'leave', 'mental_health_consequence',
                       'phys_health_consequence', 'coworkers', 'supervisor', 'mental_health_interview', 'phys_health_interview
                       'mental_vs_physical', 'obs_consequence', 'benefits', 'care_options', 'wellness_program',
                       'seek_help']
     floatFeatures = []

     # Clean the NaN's
     for feature in train_df:
         if feature in intFeatures:
             train_df[feature] = train_df[feature].fillna(defaultInt)
         elif feature in stringFeatures:
             train_df[feature] = train_df[feature].fillna(defaultString)
         elif feature in floatFeatures:
             train_df[feature] = train_df[feature].fillna(defaultFloat)
         else:
             print('Error: Feature %s not recognized.' % feature)
     train_df.head()
```

Рис.4 Очистка даних

Стовпець gender містить багато значень які несуть одну й ту саму інформацію, що можна оптимізувати привівши значення стовпця до одного уніфікованого вигляду. Використовуючи описаний вище метод, пропущені значення age заміню на середнє значення з різних вікових категорій.

Наступним важливим кроком є кодування категоріальних даних. Відповідно, дослідивши дані, можна зробити висновок, що можемо замінити дані на числові, що буде більш зрозумілим для моделі. Стовпець country буде важко змінити на числові дані, але він не несе велику кількість інформації тому ми можемо відкинути його. (Рис. 5, 6)

```
#Encoding data
labelDict = {}
for feature in train_df:
    le = preprocessing.LabelEncoder()
    le.fit(train_df[feature])
    le_name_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
    train_df[feature] = le.transform(train_df[feature])
    # Get labels
    labelKey = 'label_' + feature
    labelValue = [*le_name_mapping]
    labelDict[labelKey] =labelValue

for key, value in labelDict.items():
    print(key, value)
```

```
label_Age [18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
label_Gender ['female', 'male', 'trans']
label_Country ['Australia', 'Austria', 'Belgium', 'Bosnia and Herzegovina', 'Br
label_self_employed ['No', 'Yes']
label_family_history ['No', 'Yes']
label_treatment ['No', 'Yes']
label_work_interfere ["Don't know", 'Never', 'Often', 'Rarely', 'Sometimes']
label_no_employees ['1-5', '100-500', '26-100', '500-1000', '6-25', 'More than
label_remote_work ['No', 'Yes']
label_tech_company ['No', 'Yes']
label_benefits ["Don't know", 'No', 'Yes']
label_care_options ['No', 'Not sure', 'Yes']
label_wellness_program ["Don't know", 'No', 'Yes']
label_seek_help ["Don't know", 'No', 'Yes']
label_anonymity ["Don't know", 'No', 'Yes']
```

Рис. 5 Кодування даних

	Age	Gender	self_employed	family_history	treatment	work_interfere	no_employees	remote_work	tech_company
0	19	0	0	0	1	2	4	0	1
1	26	1	0	0	0	3	5	0	0
2	14	1	0	0	0	3	4	0	1
3	13	1	0	1	1	2	2	0	1
4	13	1	0	0	0	1	1	1	1

Рис. 6 Отриманий датасет після кодування

3.2 Дослідження даних

Процес дослідження даних є важливим кроком у розумінні характеристик набору даних перед побудовою моделі передбачення. Цей процес допомагає нам з'ясувати структуру даних, виявити залежності між змінними і зрозуміти, як ці залежності можуть вплинути на модель.

Один з перших кроків у процесі дослідження даних - аналіз статистичних характеристик. Ми розглядаємо основні статистичні показники, такі як середнє значення, медіана, стандартне відхилення та квартилі. Це дозволяє нам отримати загальне уявлення про розподіл значень кожної змінної.

Далі, ми можемо побудувати матрицю коваріації (covariance matrix). Ця матриця вказує на ступінь взаємозв'язку між парами змінних. Коваріація показує, які змінні змінюються разом, а кореляція - наскільки сильно і в якому напрямку вони пов'язані. Позитивна коваріація вказує на пряму залежність, тобто коли одна змінна збільшується, інша також збільшується. Негативна коваріація показує зворотну залежність, тобто коли одна змінна збільшується, інша зменшується. Коваріація близька до нуля вказує на відсутність залежності між змінними. Матриця коваріації дозволяє нам виявити загальні залежності між змінними та виключити непотрібні змінні з подальшого аналізу.

Крім цього, ми можемо провести кореляційний аналіз для виявлення залежностей між змінними. Це може бути виконано за допомогою матриці кореляції або графіків кореляції, таких як теплова карта (heatmap). Це дозволяє нам визначити, які змінні мають найсильніші зв'язки між собою і які можуть бути корисними для подальшої моделювання.

Для датасету побудую матрицю коваріації для дослідження залежностей різних характеристик. (Рис. 7) Також створю матрицю

кореляцій для різних типів лікування та догляду за людьми з депресією. (Рис. 8)

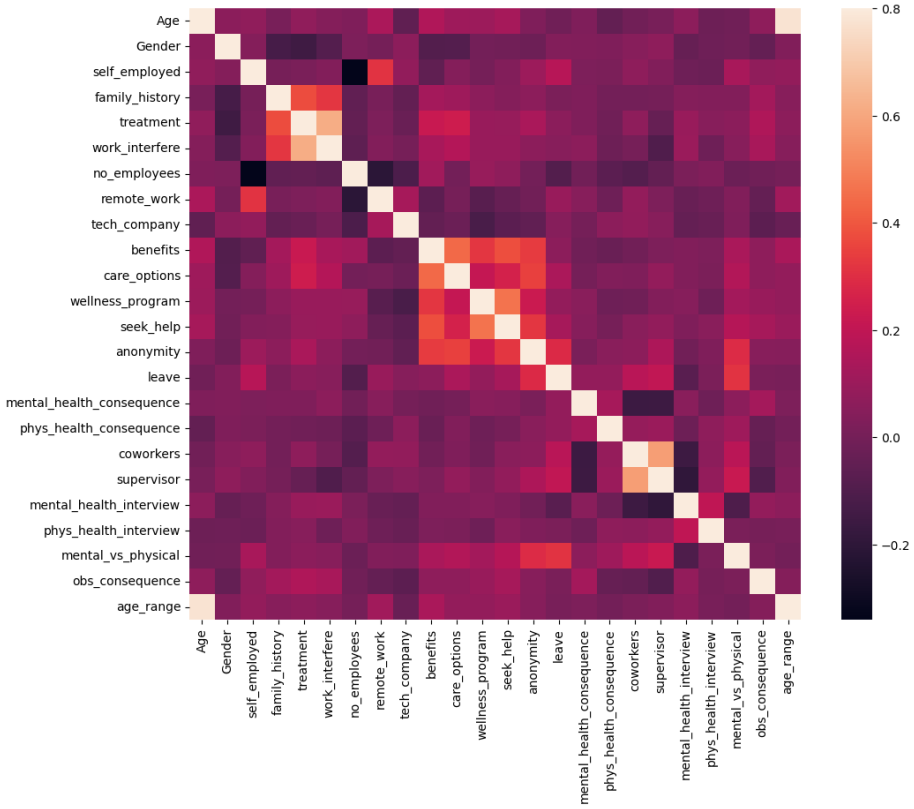

Рис. 7 Матриця коваріації досліджуваних характеристик

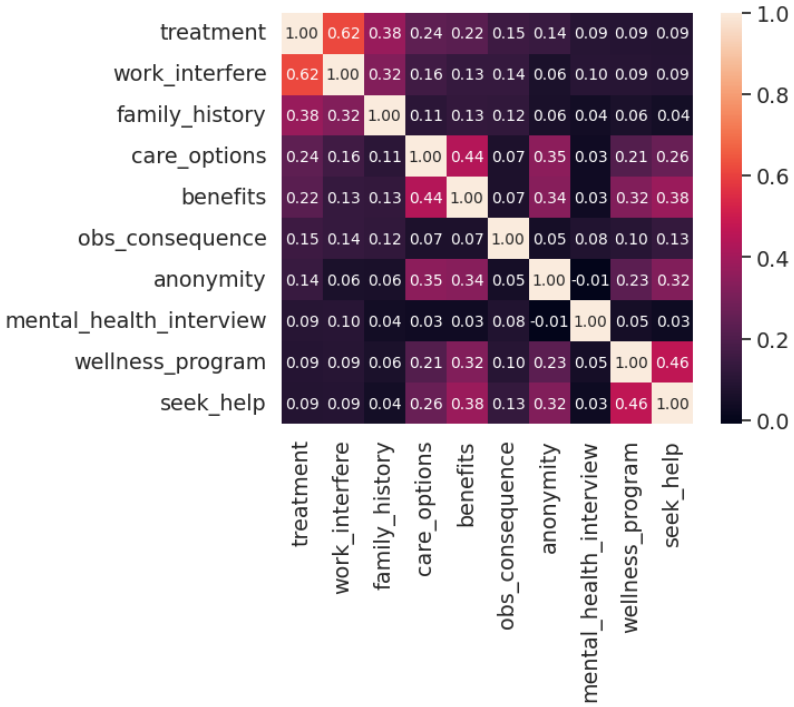

Рис. 8 Кореляція різних видів лікування

Окрім цього, ми можемо побудувати графіки для візуалізації залежностей. Наприклад, графік розсіювання (scatter plot) дозволяє нам перевірити наявність лінійних або нелинійних залежностей між двома змінними. Ми також можемо побудувати графіки розподілу (distribution plots), які демонструють розподіл значень кожної змінної. Графіки boxplot допомагають виявити викиди (outliers) та відстежити розподіл значень за категоріями.

Важливо відмітити, що процес дослідження даних не є лінійним і може включати багато ітерацій. Ми можемо змінювати підхід до аналізу, додавати нові графіки, перевіряти різні аспекти даних, щоб отримати якомога більше інсайтів перед побудовою моделі передбачення.

Отже, побудую графік розподілу людей за віком (Рис. 8, 9), з якого можна побачити, що вибірка осіб старше 35 років є невелика, що в подальшому може бути покращено, опрацьовуванням та додаванням більшої кількості даних для аналізу.

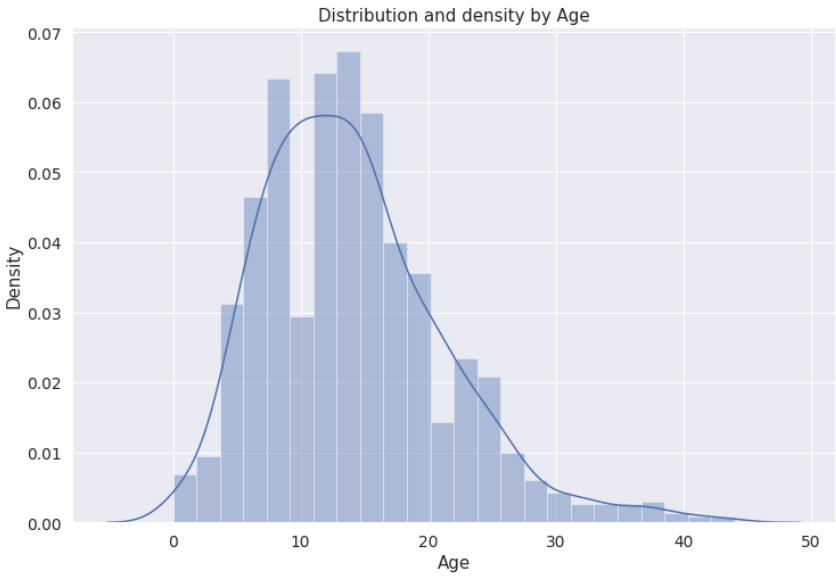

Рис. 8 Графік розподілу людей за віком

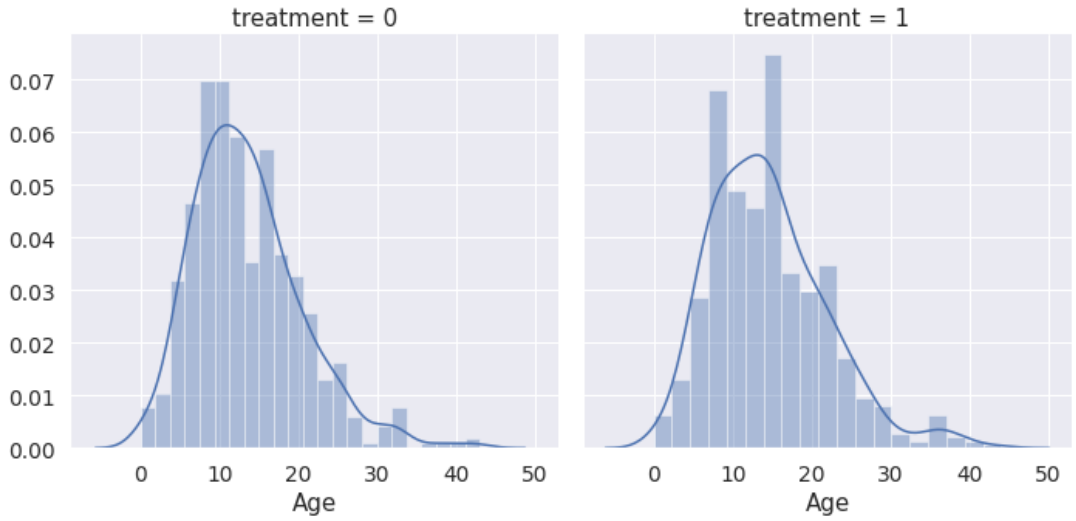

Рис. 9 Графік розподілу людей за віком в залежності чи проходили вони лікування

Також наведу приклад графіку імовірності депресії, чи будь якого іншого психічного захворювання, в залежності від статі. (Рис. 10) На графіку також можна побачити проблему кількості вхідних даних.

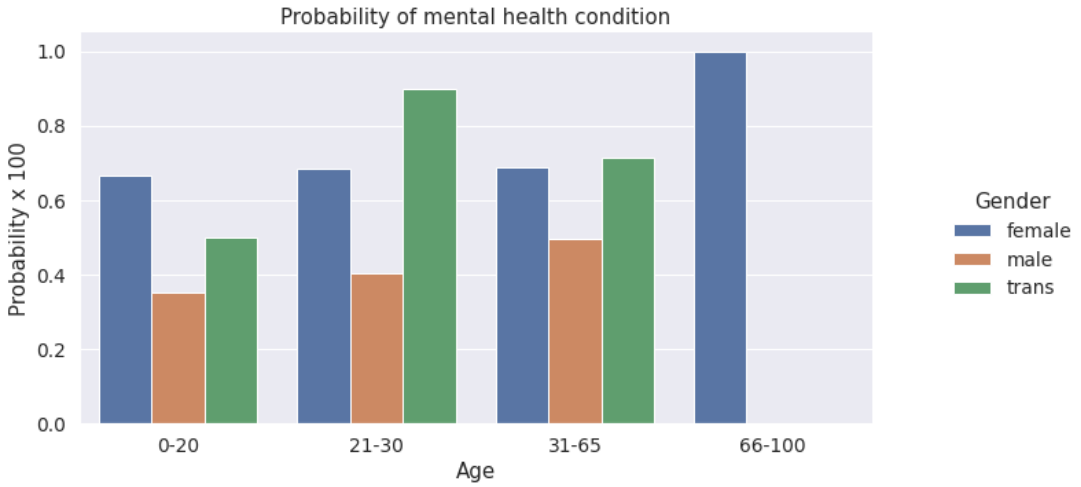

Рис. 10 Графік імовірності психічних захворювань залежно від статі

3.3 Scaling and fitting

Scaling і fitting є двома важливими концепціями у машинному навчанні та попередній обробці даних. Пояснення кожного терміну:

1. Scaling: Scaling відноситься до процесу перетворення даних в певний діапазон або розподіл. Воно виконується для забезпечення подібної шкали для всіх ознак або змінних, що може бути корисним для багатьох алгоритмів машинного навчання. Шкалювання особливо корисне, коли ознаки мають різні одиниці виміру або діапазони, оскільки це допомагає запобігти тому, щоб деякі ознаки домінували над іншими лише через їх шкалу.
   Існує кілька технік шкалювання, таких як:
   - Стандартизація (або нормування за z-оцінкою): Вона перетворює дані таким чином, щоб вони мали середнє значення рівне нулю і одиничну дисперсію. Це досягається шляхом віднімання середнього значення і поділу на стандартне відхилення.
   - Мінімаксне Scaling: Воно перетворює дані на певний діапазон, зазвичай від 0 до 1. Це включає віднімання мінімального значення і поділ на розмах (максимальне значення мінус мінімальне).
   - Robust scaling: Воно схоже на стандартизацію, але використовує медіану та міжквартильний розмах замість середнього значення та стандартного відхилення. Це робить його більш стійким до викидів.
2. Вибір методу шкалювання залежить від характеру даних і вимог алгоритму машинного навчання, який використовується.
3. Fitting: Процес пристосування відноситься до навчання моделі машинного навчання на заданому наборі даних. Воно включає пошук оптимальних параметрів або коефіцієнтів, які найкраще відображають залежність між вхідними ознаками і цільовою змінною. Модель "вчиться" з даних під час процесу пристосування, налаштовуючи свої параметри на основі певного алгоритму

оптимізації (наприклад, градієнтного спуску) та визначеної функції втрат або цілі.
Процес пристосування змінюється залежно від типу алгоритму машинного навчання. Наприклад:

- У лінійній регресії пристосування включає пошук найкращої прямої, яка мінімізує суму квадратів різниць між прогнозованими та фактичними значеннями.
- У деревах рішень пристосування включає визначення оптимальних розбиттів та правил прийняття рішень, які мінімізують невпорядкованість або максимізують приріст інформації.
- У нейронних мережах пристосування включає налаштування ваг та зсувів за допомогою методу зворотного поширення помилки та градієнтного спуску для мінімізації помилки між прогнозованими та фактичними виходами.

4. Після того, як модель налаштована на тренувальних даних, її можна використовувати для прогнозування на нових, невидимих даних.

Підсумовуючи, шкалювання перетворює дані на певну шкалу або розподіл, а пристосування включає навчання моделі на даних, щоб знайти оптимальні параметри для передбачення цільової змінної.

Отже, для даного набору даних, необхідно обрати найбільш значущі ознаки, які використовуватимуться для передбачення. Але попередньо виконаю scaling, та стандартизую значення age, використовуючи мінімаксне перетворення, яке суттєво відрізняється від інших значень. Наступним кроком, розіб'ю дані на тренувальний та тестовий сет.

Для виокремлення ознак використаю метод лісу дерев рішень. Метод виокремлення ознак за допомогою Random Forest та feature_importances використовується для оцінки важливості кожної ознаки у задачі машинного навчання. Цей метод дає змогу визначити, які

ознаки мають найбільший вплив на результат моделі. Цей метод дозволяє ідентифікувати найбільш важливі ознаки у задачі та зрозуміти, як вони впливають на результати моделі. Ця інформація може бути корисною для подальшого вдосконалення моделі, виключення неважливих ознак або вибору оптимального підмножини ознак для використання. (Рис. 11)

```
# Build a forest and compute the feature importances
forest = ExtraTreesClassifier(n_estimators=250,
                              random_state=0)

forest.fit(X, y)
importances = forest.feature_importances_
std = np.std([tree.feature_importances_ for tree in forest.estimator
             axis=0)
indices = np.argsort(importances)[::-1]

labels = []
for f in range(X.shape[1]):
    labels.append(feature_cols[f])

# Plot the feature importances of the forest
plt.figure(figsize=(12,8))
plt.title("Feature importances")
plt.bar(range(X.shape[1]), importances[indices],
       color="r", yerr=std[indices], align="center")
plt.xticks(range(X.shape[1]), labels, rotation='vertical')
plt.xlim([-1, X.shape[1]])
plt.show()
```

Рис. 11 Виділення ознак методом Random Forest

Побудую графік з виділеними ознаками та їх важливістю. (Рис. 12)

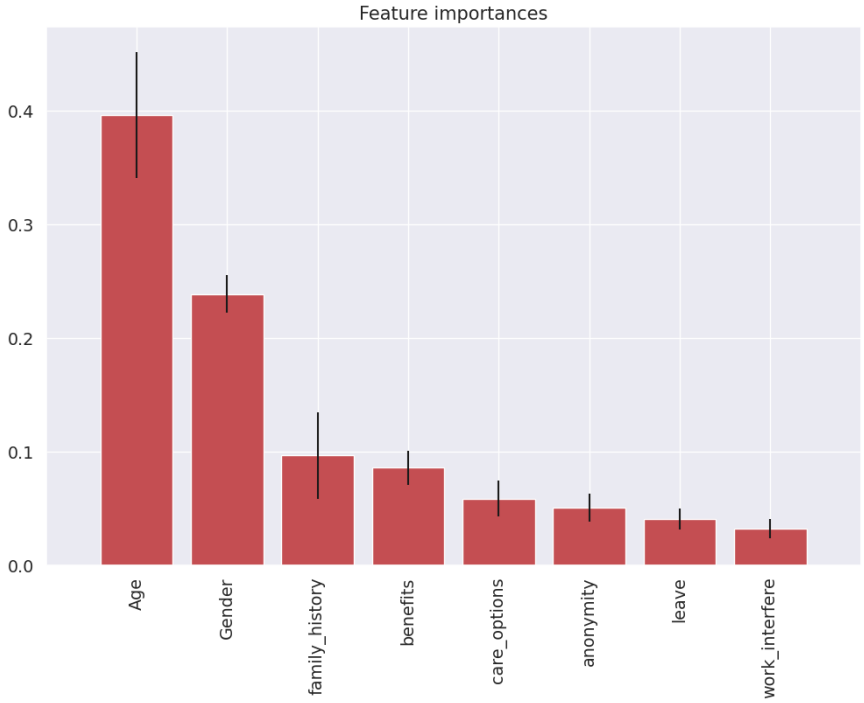

Рис. 12 Графік виділених ознак

3.4 Створення моделей

Логістична регресія - це відомий алгоритм машинного навчання, який належить до підходу керованого навчання. У цьому методі ми прогнозуємо конкретну залежну змінну з набору незміщених змінних. Логістична регресія використовується для прогнозування виходу конкретних структурованих змінних. Отже, результатом має бути категоричне або дискретне значення. Це може бути 0 або 1, "так" або "ні", "правда" або "неправда" тощо, але він дає імовірнісні значення, які знаходяться десь між 0 і 1, замість того, щоб давати точні значення, як 0 і 1.

ROC-крива (Receiver Operating Characteristic curve) є інструментом для візуалізації та оцінки ефективності моделі класифікації. Вона дозволяє

визначити торгову офф та зв'язок між чутливістю (true positive rate) та специфічністю (true negative rate) моделі при різних порогових значеннях.

ROC-крива будується шляхом розрахунку чутливості та специфічності для різних порогових значень, застосованих до вихідних прогнозів моделі. Чутливість відображає відсоток правильно визначених позитивних випадків (true positives) відносно загальної кількості позитивних випадків, тоді як специфічність відображає відсоток правильно визначених негативних випадків (true negatives) відносно загальної кількості негативних випадків.

ROC-крива графічно представляється на площині з осі чутливості на вісі y та осі 1-специфічності на вісі x. Ідеальна ROC-крива проходить через точки (0, 0) та (1, 1) і відповідає моделі з ідеальною чутливістю та специфічністю. Зазвичай ROC-крива представлена лінією, яка лежить над діагональною лінією (шансова модель) і вгору в сторону верхнього лівого кута.

Оцінка ефективності моделі здійснюється на основі площі під ROC-кривою, відомої як площа під кривою (AUC - Area Under the Curve). Чим більше площа під кривою, тим краща модель класифікації. Значення AUC може варіюватися від 0 до 1, де 0 відповідає моделі з нульовою прогностичною здатністю, а 1 відповідає ідеальній моделі з повною прогностичною здатністю.

ROC-крива є важливим інструментом для порівняння та вибору моделей класифікації, особливо в ситуаціях, коли класи незбалансовані або коли важливо керувати торговою офф між чутливістю та специфічністю.

Для логістичної моделі, яка має точність 79,63% побудую ROC (Рис. 13) криву для оцінки ефективності передбачення моделі. AUC area - площа під кривою дорівнює 0,79, що говорить про якість моделі класифікації.

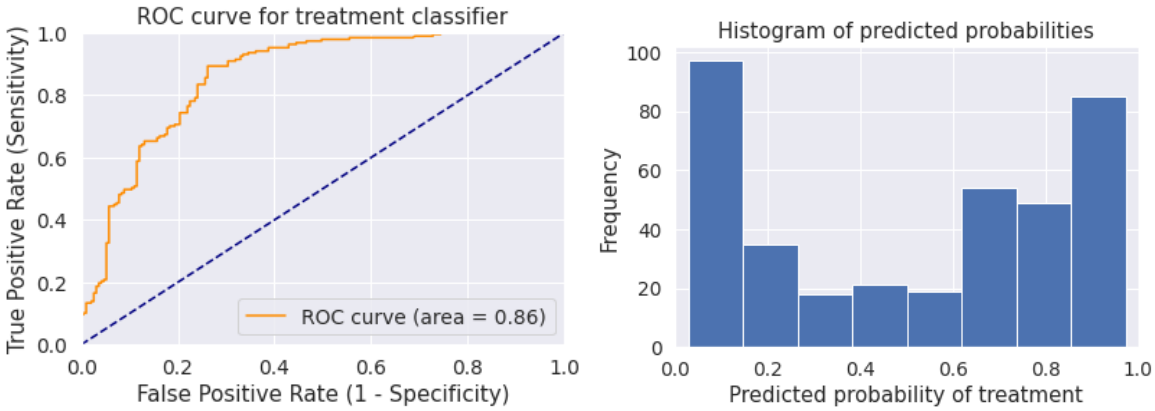

Рис. 13 ROC крива для Логістичної регресії

K-Nearest Neighbour - це базовий алгоритм машинного навчання, який базується на техніці Supervised Learning. У методі KNN існуючі випадки та нові випадки/дані будуть схожими. KNN - це непараметричний алгоритм, який не робить жодних припущень щодо підкреслених даних або їх розподілу. А також він працює з декількома класами.

Для моделі KNN, яка має точність 80,4% побудую ROC (Рис. 14) криву для оцінки ефективності передбачення моделі. AUC area - площа під кривою дорівнює 0,805.

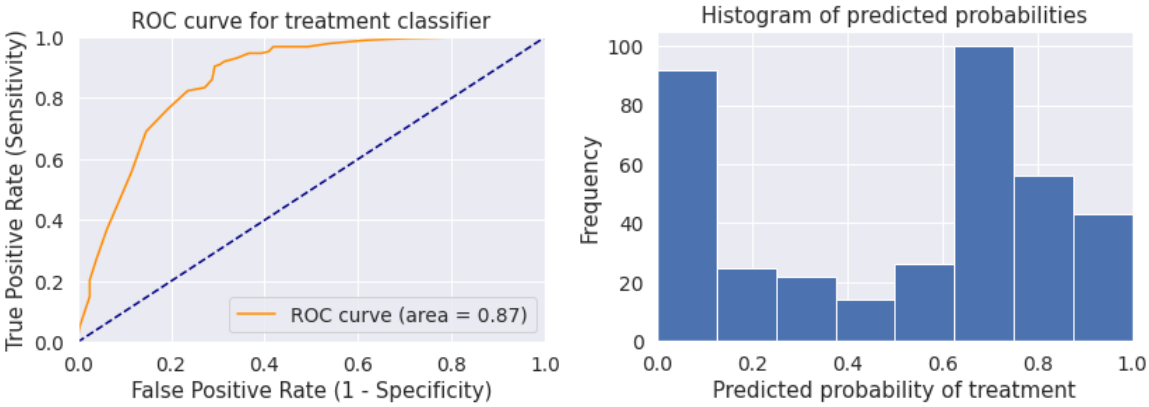

Рис. 14 ROC крива для моделі KNN

Дерево рішень (Decision Tree) - це широко розповсюджена техніка керованого машинного навчання, яка використовується в інтелектуальному аналізі даних. Дерево рішень - це діаграма, яку люди

використовують для ілюстрації статистичної ймовірності або для визначення послідовності подій, дій чи результатів.

Побудую ROC (Рис. 15) криву для оцінки ефективності передбачення моделі дерево рішень, точність моделі 80,68%. AUC area - площа під кривою дорівнює 0,808.

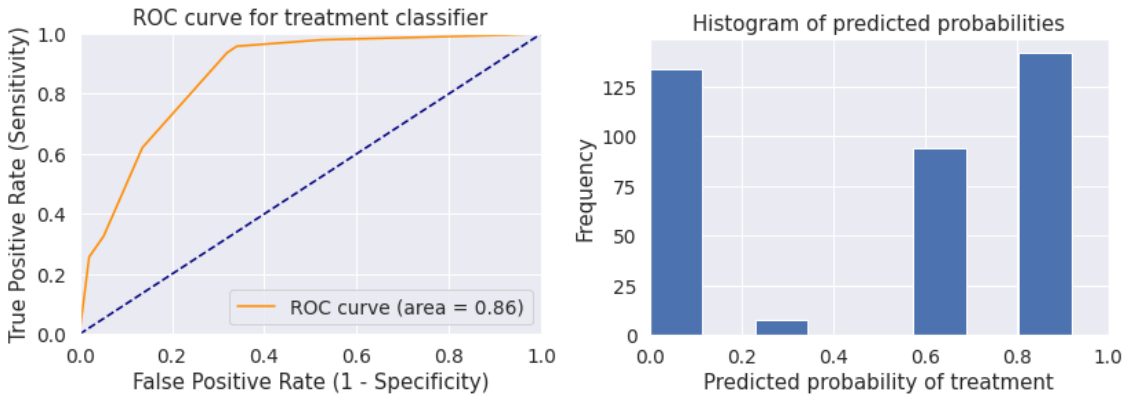

Рис. 15 ROC крива для Дерева рішень

Випадковий ліс (Random Forest) - це метод, який вирішує проблеми класифікації, а також регресії, і базується на підході керованого машинного навчання. Однак найчастіше його використовують для класифікації. Оскільки він об'єднує багато дерев рішень для створення "лісу" і подає до них випадкові ознаки з вхідного набору даних, він називається випадковим лісом.

Для моделі випадкового лісу, яка має точність 81,2% побудую ROC (Рис. 16) криву для оцінки ефективності передбачення моделі. AUC area - площа під кривою дорівнює 0,813.

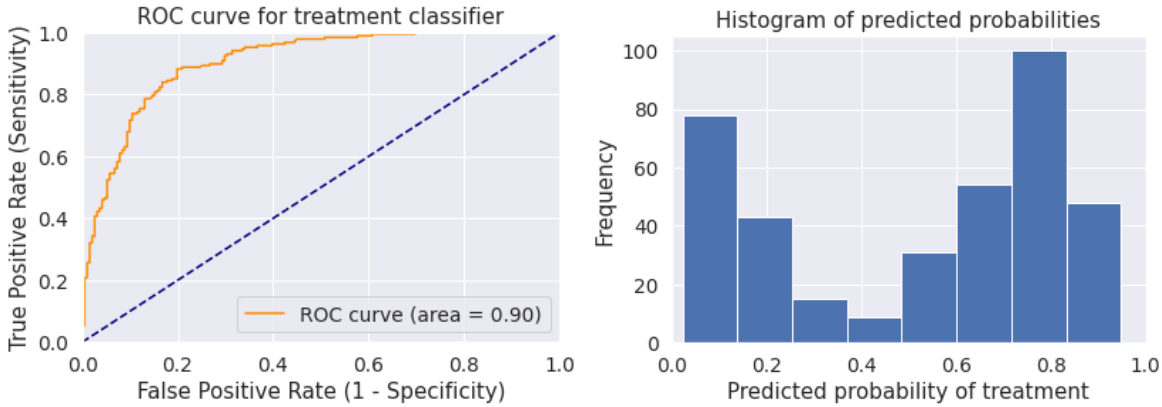

Рис. 16 ROC крива для моделі Випадковий ліс

Stacked Generalization, або просто "Стекінг", - це алгоритм ансамблю машинного навчання. Подібно до пакування та бустінгу, він передбачає об'єднання прогнозів від багатьох моделей машинного навчання на одному наборі даних.

Побудую ROC (Рис. 17) криву для оцінки ефективності передбачення моделі Stacking, точність моделі 81,7%. AUC area - площа під кривою дорівнює 0,817.

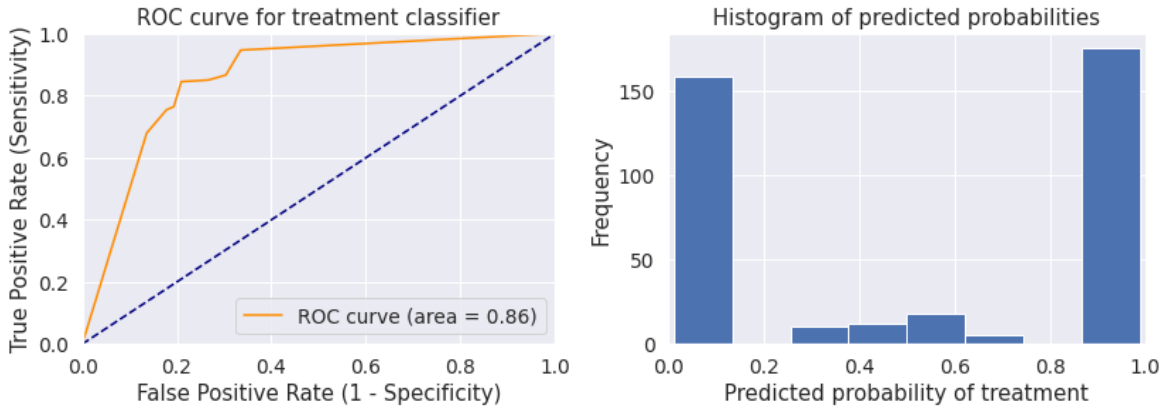

Рис. 17 ROC крива для моделі Stacking

Це дослідження визначило п'ять методів машинного навчання: класифікатор найближчого сусіда (KNN), логістична регресія, дерево рішень, стекінг та випадковий ліс. Ми оцінили їхню точність у виявленні

проблем з психічним здоров'ям. Спочатку ми виконали класифікатори, які включали всі 27 атрибутів, що були ідентифіковані з текстових документів, а потім ми виконали його, включивши 8 ознак, які були відібрані за допомогою алгоритму відбору ознак. Точність даного тестового набору для класифікатора - це відсоток екземплярів тестового набору, які були правильно класифіковані за допомогою класифікатора.

В таблиці 1 показано, що точність стекування є більш точною порівняно з іншими класифікаторами. Точність будь-якого класифікатора залежить від того, наскільки добре класифікатор класифікує набір даних, який тестується. Ми виміряли це за допомогою площі під робочою кривою приймача. На ROC-кривій ідеальний тест буде мати площу 1, а нікчемний тест - 0,5. Наведені вище рисунки ілюструють графіки залежності п'яти класифікаторів від значень ROC-площі. Я помітила, що ці класифікатори точніше прогнозують стан психічного здоров'я, ніж інші класифікатори, оскільки ROC-площа всіх використаних класифікаторів знаходиться в межах від 0,8 до 0,9.

Метод	**Точність (%)**
Логістична регресія	79, 63
KNN класифікація	80,42
Дерево рішень	80,68
Випадковий ліс	81,2
Stacking	81,7

Таблиця 1. Результати роботи, точність для кожного з методів

3.5 Нейронна модель та передбачення

Провівши дослідження даних, виділивши ознаки, та розділивши датасет на навчальний та тренувальний, створю DNN(Deep Neural Network) для класифікації даних та передбачення депресії. Виділю ознаки (features) та побудую DNN модель, яка складається з 2 шарів по 10 нейронів, використаю оптимізатор: tf.train.ProximalAdagradOptimizer, який є варіантом оптимізатора з проксимальним методом Адаграда. Оптимізатор відповідає за оновлення вагових коефіцієнтів моделі для мінімізації функції втрати. Також додам силу регуляризації, яка додає штраф до втрати моделі залежно від абсолютних значень вагових коефіцієнтів. Це допомагає контролювати перенавчання моделі та покращує її загальну узагальнюючу здатність. (Рис. 18)

```
# Define Tensorflow feature columns
age = tf.feature_column.numeric_column("Age")
gender = tf.feature_column.numeric_column("Gender")
family_history = tf.feature_column.numeric_column("family_history")
benefits = tf.feature_column.numeric_column("benefits")
care_options = tf.feature_column.numeric_column("care_options")
anonymity = tf.feature_column.numeric_column("anonymity")
leave = tf.feature_column.numeric_column("leave")
work_interfere = tf.feature_column.numeric_column("work_interfere")
feature_columns = [age, gender, family_history, benefits, care_options, anonymity, leave, work_interfere]
```

ntiate an Estimator

```
# Build a DNN with 2 hidden layers and 10 nodes in each hidden layer.
model = tf.estimator.DNNClassifier(feature_columns=feature_columns,
                                   hidden_units=[10, 10],
                                   optimizer=tf.train.ProximalAdagradOptimizer(
                                     learning_rate=0.1,
                                     l1_regularization_strength=0.001
                                   ))
```

Рис. 18 Налаштування для DNN моделі

DNNClassifier є класом моделі нейронної мережі, який використовується для класифікації даних. DNN в даному контексті означає Deep Neural Network (глибока нейронна мережа). Цей клас моделі надає можливість побудувати та навчити глибоку нейронну мережу для вирішення задач класифікації.

DNNClassifier використовується для вирішення задач, де потрібно прогнозувати категоріальну змінну або класифікувати об'єкти в певні категорії. Він може бути застосований в різних областях, включаючи комп'ютерне зору, обробку природної мови, фінанси та інші.

DNNClassifier використовує глибоку нейронну мережу, яка складається з багатьох шарів нейронів. Кожен шар містить набір нейронів, які виконують обчислення над вхідними даними та передають результати наступному шару. Глибока нейронна мережа здатна виявляти складні залежності в даних та вирішувати складні задачі класифікації.

Для використання DNNClassifier необхідно навчити модель на тренувальних даних, вказавши архітектуру мережі, кількість шарів, кількість нейронів у кожному шарі, функції активації та інші параметри. Після тренування модель може бути використана для передбачення класу нових даних на основі навчених знань.

DNNClassifier є потужним інструментом для класифікації даних, особливо в ситуаціях, коли дані мають складні залежності та неоднорідність. Він дозволяє створювати глибокі нейронні мережі з багатьма шарами та настроювати їх для досягнення високої точності класифікації.

Протестую модель, виконаю передбачення та перевірю її точність. В результаті точність DNN моделі складає 79,6%.

Щоб підбити підсумки побудую графік з отриманими результатами, які я отримала використовуючи вищеописані методи для передбачення депресії по заданих виділених параметрах (Рис. 19).

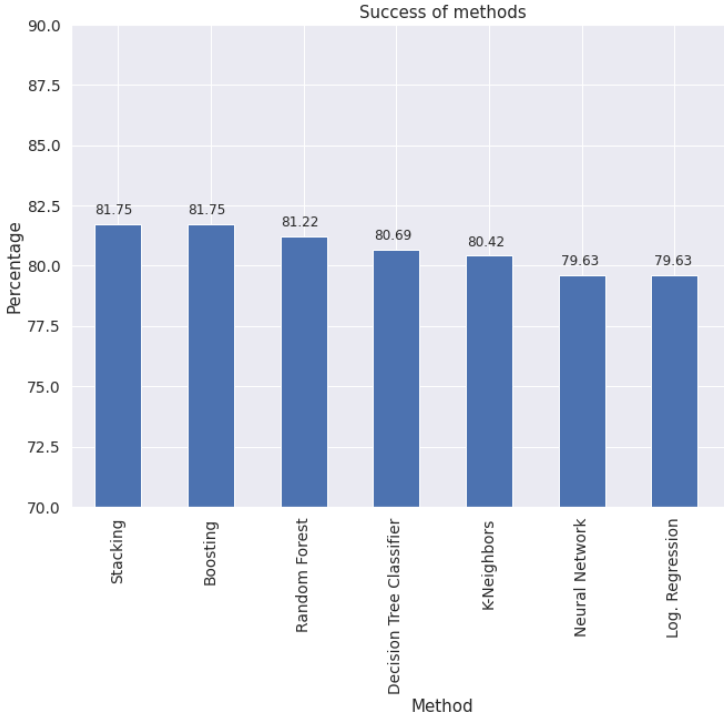

Рис. 19 Порівняння різних методів передбачення

ВИСНОВОК

Оскільки існує багато доступних методів машинного навчання, дуже важливо провести порівняльний аналіз цих методів, щоб визначити найкращий для конкретної області дослідження. У даній курсовій роботі було порівняно п'ять різних методів машинного навчання для класифікації набору даних, що стосується проблем психічного здоров'я.

Результати дослідження показали, що всі п'ять методів машинного навчання досягають високої точності в класифікації. Усі класифікатори мають точність, що перевищує 79%. Хоча використаний набір даних був досить обмеженим, ці результати свідчать про потенціал методів машинного навчання в класифікації проблем психічного здоров'я.

Крім того, було виявлено, що використання ансамблевих класифікаторів, зокрема KNN, SVM та Random Forest, значно підвищує ефективність прогнозування психічного здоров'я з точністю до 90%. Це підкреслює важливість використання комплексних моделей, які комбінують декілька класифікаторів для отримання більш точних результатів.

Це дослідження зосереджувалося на перевагах машинного навчання для поліпшення ідентифікації психічного здоров'я та діагностики різних психічних захворювань. Застосування методів машинного навчання може допомогти вчасно виявляти такі стани, як хвороба Альцгеймера, депресія та шизофренія, що є важливим для забезпечення ефективного лікування та підтримки пацієнтів.

Загалом, машинне навчання має великий потенціал у покращенні клінічних та дослідницьких досліджень в галузі психічного здоров'я. Воно дозволяє отримати нові інсайти та розуміння про причини, діагностику та лікування психічних захворювань, що сприяє підвищенню благополуччя та якості життя пацієнтів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Stevie Chancellor, Eric PS Baumer, and Munmun De Choudhury. Who is the" human" in human-centered machine learning: The case of predicting mental health from social media. Proceedings of the ACM on Human-Computer Interaction, 3(CSCW):1–32, 2019.
2. Model stacking to improve prediction and variable importance robustness for soft sensor development. [Електронний ресурс] Режим доступу: https://www.sciencedirect.com/
3. Predicting Anxiety, Depression and Stress in Modern Life using Machine Learning Algorithm. [Електронний ресурс] Режим доступу: https://www.sciencedirect.com/

ЛІСТИНГ ПРОГРАМИ

```
train_df = pd.read_csv('survey.csv')
print(train_df.shape)
print(train_df.describe())
print(train_df.info())
#missing data
total = train_df.isnull().sum().sort_values(ascending=False)
percent =
(train_df.isnull().sum()/train_df.isnull().count()).sort_values(ascendi
ng=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total',
'Percent'])
missing_data.head(20)
print(missing_data)
#dealing with missing data
train_df.drop(['comments'], axis= 1, inplace=True)
train_df.drop(['state'], axis= 1, inplace=True)
train_df.drop(['Timestamp'], axis= 1, inplace=True)

train_df.isnull().sum().max() #just checking that there's no missing
data missing...
train_df.head(5)
defaultInt = 0
defaultString = 'NaN'
defaultFloat = 0.0

# Create lists by data tpe
intFeatures = ['Age']
stringFeatures = ['Gender', 'Country', 'self_employed',
'family_history', 'treatment', 'work_interfere',
                'no_employees', 'remote_work', 'tech_company',
'anonymity', 'leave', 'mental_health_consequence',
                'phys_health_consequence', 'coworkers', 'supervisor',
'mental_health_interview', 'phys_health_interview',
                'mental_vs_physical', 'obs_consequence', 'benefits',
'care_options', 'wellness_program',
                'seek_help']
floatFeatures = []

# Clean the NaN's
for feature in train_df:
```

```
    if feature in intFeatures:
        train_df[feature] = train_df[feature].fillna(defaultInt)
    elif feature in stringFeatures:
        train_df[feature] = train_df[feature].fillna(defaultString)
    elif feature in floatFeatures:
        train_df[feature] = train_df[feature].fillna(defaultFloat)
    else:
        print('Error: Feature %s not recognized.' % feature)
train_df.head()
```

Повний код програми можна знайти за посиланням: [Colaboratory]