НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
КАФЕДРА ІНФОРМАТИКИ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

КУРСОВА РОБОТА

з дисципліни «Аналіз даних в інформаційних системах»

на тему: «Аналіз впливу деяких факторів на рівень щастя населення країн. Класифікація країн за рівнем розвитку»

Студентів 2 курсу ІП-01 групи
Спеціальності: 121
«Інженерія програмного забезпечення»
Заранік Богдан Юрійович
Пашковський Євгеній Сергійович

«ПРИЙНЯВ» з оцінкою

_____

доц. Ліхоузова Т.А. / доц. Олійник Ю.О.

_____

___

Підпис Дата

Київ - 2022 рік

Національний технічний університет України “КПІ ім. Ігоря Сікорського”

Кафедра інформатики та програмної інженерії

Дисципліна Аналіз даних в інформаційно-управляючих системах

Спеціальність 121 "Інженерія програмного забезпечення"

Курс 2 Група ІП-01

Семестр 4

ЗАВДАННЯ

на курсову роботу студентів

Зараніка Богдана Юрійовича та Пашковського Євгенія Сергійовича

1.Тема: Аналіз впливу деяких факторів на рівень щастя населення країн.

2.Строк здачі студентом закінченої роботи 19.06.2022

3. Вхідні дані до роботи методичні вказівки до курсової робота, обрані дані з сайту

https://www.kaggle.com/datasets/virajkulkarni952/country-development-indicators

https://www.kaggle.com/datasets/mayzannilarthein44/world-happiness-report-2015-to-2022

https://www.kaggle.com/datasets/jamesvandenberg/renewable-power-generation?select=Country_Consumption_TWH.csv

https://www.kaggle.com/datasets/saleh846/causes-of-deaths-worldwide?select=age-between-5-and-14.csv

4.Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

1.Постановка задачі

2.Аналіз предметної області

3.Розробка сховища даних

4.Інтелектуальний аналіз даних

5.Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

6.Дата видачі завдання 16.04.2022

КАЛЕНДАРНИЙ ПЛАН

№п/п	Назва етапів курсової роботи	Термін виконання етапів роботи	Підписи керівника, студента
1.	Отримання теми курсової роботи	16.04.2022	
2.	Визначення зовнішніх джерел даних	20.04.2022	
3.	Пошук та вивчення літератури з питань курсової роботи	25.04.2022	
4.	Розробка моделі сховища даних	01.05.2022	
5.	Розробка ETL процесів	15.05.2022	
6.	Обґрунтування методів інтелектуального аналізу даних	20.05.2022	
7.	Застосування та порівняння ефективності методів інтелектуального аналізу даних	25.05.2022	
8.	Підготовка пояснювальної записки	15.06.2022	
9.	Здача курсової роботи на перевірку	19.06.2022	
10.	Захист курсової роботи	21.06.2022	

Студент _____ Заранік Богдан Юрійович

(підпис) (прізвище, ім’я, по батькові)

Студент _____ Пашковський Євгеній Сергійович

(підпис) (прізвище, ім’я, по батькові)

Керівник _____ доц. Ліхоузова Т.А

(підпис) (прізвище, ім’я, по батькові)

Керівник _____ доц. Олійник Ю.О.

(підпис) (прізвище, ім’я, по батькові)

"26" червня 2022 р.

АНОТАЦІЯ

Пояснювальна записка до курсової роботи: 31 сторінок, 30 рисунків, 8 посилань.

Об’єкт дослідження: інтелектуальний аналіз даних.

Предмет дослідження: створення програмного забезпечення, що дозволить аналізувати залежність рівня щастя від деяких параметрів розвитку країн, його прогнозування та тернарна класифікація країн за рівнем розвитку у залежності від вищезгаданих параметрів.

Мета роботи: проектування та реалізація сховища даних, ETL процесів та імплементація програмного забезпечення мовою Python для отримання даних зі сховища та їх подальшого аналізу, прогнозування та класифікації.

Курсова робота включає в себе: опис проектування, створення та заповнення сховища даних згідно з темою завдання за допомогою використання бібліотек мови Python та скриптів SQL, опис створення програмного забезпечення для інтелектуального аналізу даних, їх графічного представлення у вигляді графіків та гістограм та прогнозування за допомогою математичних моделей.

МОДЕЛЬ ПРОГНОЗУВАННЯ, КЛАСИФІКАЦІЯ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, ETL ПРОЦЕСИ.

ЗМІСТ

ВСТУП

Питання рівня щастя населення має першочергове значення у сучасному світі, що розвивається з величезною швидкістю. Від нього залежить рівень життя населення, темпи економічного розвитку країни, привабливість країни для спеціалістів сучасних професій із-за кордону, інвестиційна привабливість та інші.

Цей параметр є достатньо відносним, проте навіть його можна оцінити за певною шкалою. На рівень щастя населення впливають багато чинників, перш за все - економічні та соціальні.

Міжнародний індекс щастя (англ. Happy Planet Index) являє собою індекс, що відображає добробут людей та стан навколишнього середовища в різних країнах світу. Головне завдання індексу відобразити «реальний» добробут націй. Для порівняння рівня життя в різних країнах використовується значення ВВП на душу населення або ІЛР, але ці індекси не завжди можуть відобразити реальний стан речей. Зокрема порівняння значення ВВП на душу населення вважається недоречним, оскільки кінцева мета більшості людей не бути багатими, а бути щасливими та здоровими.

У нашій роботі ми задалися питанням, які саме чинники впливають на рівень щастя людей суттєво, а якими можна знехтувати.

Також, як відомо, країни поділяють на “розвинені”, “ті, що розвиваються” та “слабо розвинені”. За даними вибірок ми також маємо на меті класифікувати країни за цими трьома типами, оскільки дуже вірогідно, що параметри, що сильно впливають на рівень щастя населення країн, також будуть суттєво вливати на те, до якого класу розвиненості належить та чи інша країна. Отже, цю гіпотезу нам і потрібно перевірити шляхом створення математичної моделі класифікації даних.

В ролі системи керування сховищем даних для даної роботи буде виступати PostgreSQL, а мова програмування для реалізації застосунку – Python3.

1. ПОСТАНОВКА ЗАДАЧІ

Мета нашого дослідження - розробка ПО, що дозволяє виокремити із переліку параметрів, що впливають на рівень щастя населення, ті, що впливають суттєво, та розробити математичну модель прогнозування рівня щастя населення за даними конкретними параметрами. Також метою нашого дослідження є розробка моделі класифікації країн за рівнем розвитку.

Під час виконання курсової роботи необхідно виконати наступні завдання: Створення сховища даних типу «сніжинка».

Сховище даних повинне містити щонайменше 3 таблиць вимірів та 2 таблиць фактів.

Створення ETL процесів для завантаження даних до сховища, їх отримання зі самого сховища за допомогою запитів, а також оновлення та додавання даних до таблиць вимірів.

Реалізувати спроектоване сховище даних з використанням PostgreSQL версії 11.

Створення застосунку, що отримує вибірку даних зі створеного сховища, графічно відображає отримані дані, проводить їх інтелектуальний аналіз для отримання передбачення за допомогою різних моделей прогнозування.

Аналіз отриманих результатів, порівняння різних методів прогнозування на даній вибірці, отримання найоптимальнішого методу. Використати мову програмування Python 3 для реалізації застосунку.

Дані математичні моделі можуть суттєво допомогти економістам у перевірці своїх припущень щодо встановленого рівня щастя населення для певної країни та віднесення її до певного класу за рівнем розвиненості.

Результатом роботи алгоритму передбачення рівня щастя населення має бути певне дійсне число, що диференціює країну серед інших за вищезгаданим параметром.

Результатом роботи алгоритму класифікації країни за рівнем розвитку має бути один із трьох класів: “розвинена”, “та, що розвивається” та “слабо розвинена” - передбачуваний рівень розвитку даної країни.

2. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

На нашу думку, впливати на рівень щастя населення та рівень розвитку країни можуть такі чинники:

- ВВП на душу населення
- Рівень свободи
- Рівень довіри населення владі
- Щедрість населення
- Розповсюдженість наркотичних засобів
- Захворюваність на на деякі види захворювань
- Соціальна підтримка зі сторони держави
- Споживання електроенергії на душу населення
- Очікувана середня тривалість життя
- Площа країни проживання
- Загальна смертність на 10000 населення

Даний список параметрів, що впливають на рівень щастя населення, є неповним, оскільки на нього впливає безліч чинників, але досліджувати шукану залежніть ми будемо саме за ним, адже на нашу думку у списку присутня більшість параметрів, що так чи інакше складають оцінку рівня щастя.

У програмній системі буде реалізовано наступну функціональність, що включає в себе:

- створення ETL процесів для завантаження даних;
- створення датасету зі сховища у вигляді csv-файлу;
- графічне відображення отриманих результатів та їх аналіз.
- інтелектуальний аналіз даних;
- використання регресійних моделей прогнозування;
- використання математичних моделей класифікації даних(навчанні із вчителем);

Нами було знайдено 4 датасети.

1. World happiness report 2015-2022.

   Описує велику кількість показників, що впливають на рівень щастя населення різних країн по роках 2015-2022 та сам рівень щастя.

2. Country Electricity Consumption

   Описує параметр споживання електроенергії певної країни певного року .

3. Deaths Reasons

   Описує причини смертності населення певних країн по роках.

4. Population By Country

   Містить демографічний аналіз певних країн. Параметри такі як густина населення, кількість населення та інші.

3. ВІДБІР ТА АНАЛІЗ ДАНИХ

3.1. Візуалізація даних

Для кращого розуміння, давайте зробимо деяку візуалізацію нашого датасету. Ми знаємо, що наші дані містить колонки, наприклад, gdp, freedom, trust тож побудуємо гістограму, яка показує кількісний розподіл gdp, freedom, trust.

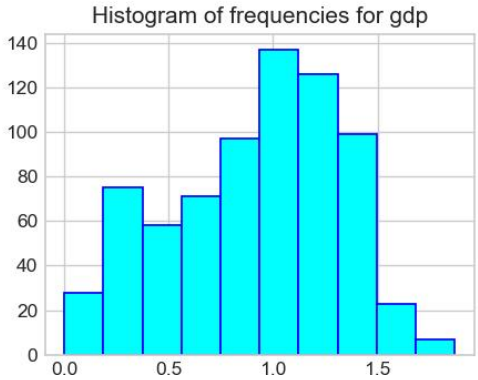

Рисунок 4.1 - Побудова гістограми для gdp

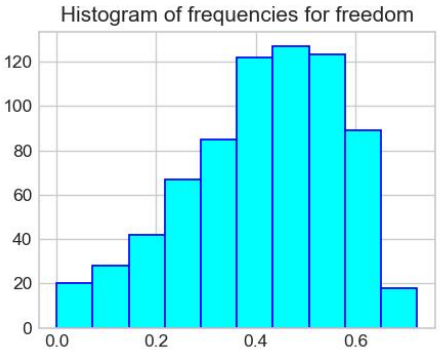

Рисунок 4.2 - Побудова гістограми для freedom

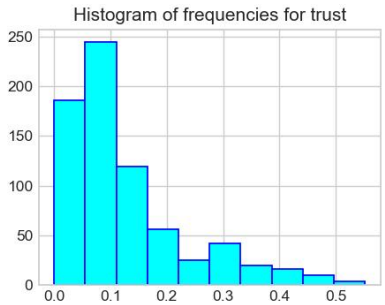

Рисунок 4.3 - Побудова гістограми для trust

Для більшої наглядності продемонструємо також діаграми розмаху деяких даних по колонках:

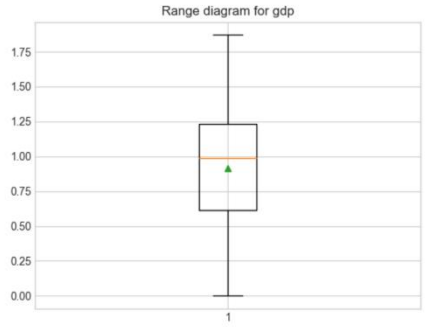

Рисунок 4.4 - Побудова діаграми розмаху для gdp

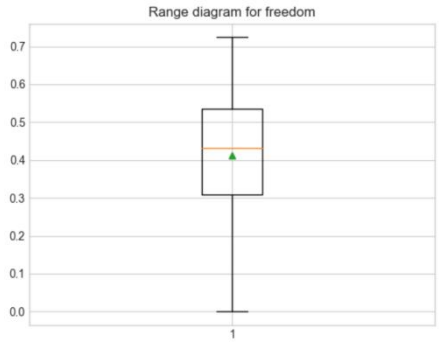

Рисунок 4.5 - Побудова діаграми розмаху для freedom

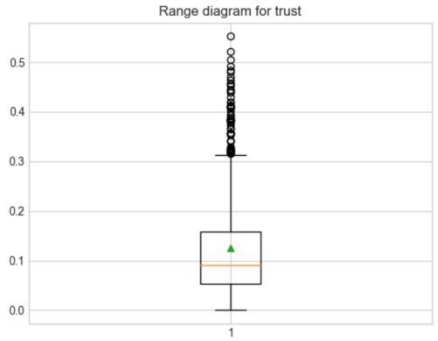

Рисунок 4.6 - Побудова діаграми розмаху для trust

3.2. Створення представлень бази даних

Для отримання зручних для обробки даних із новоствореного сховища даних створимо кілька представлень засобами СУБД мовою SQL.

Для початку створимо представлення, що описує звіт про рівень щастя населення країн світу протягом 2015-2019 років. У представленні присутня також нормалізація деяких полів. Наприклад, замість номінального показника споживання електроенергії, будемо використовувати нормалізоване значення споживання електроенергії на душу населення.

```sql
CREATE VIEW ADISCOURSEWORK.MAIN_VIEW AS
SELECT
       NAME COUNTRY_NAME,
       YEAR,
       GDP,
       SOCIAL_SUPPORT,
       FREEDOM,
         ENERGY_CONSUMPTION/(AVERAGE_POPULATION::REAL - (2020-YEAR) * C.NET_POPULATION_CHANGE)
             as ENERGY_CONSUMPTION_PER_CAPITA,
       TRUST,
       GENEROSITY,
         (TOTAL_DEATHS::REAL)/(AVERAGE_POPULATION::REAL - (2020-YEAR) * C.NET_POPULATION_CHANGE)
             AS DEATHS_PER_CAPITA,
         AREA,
         (AVERAGE_POPULATION::REAL - (2020-YEAR) * C.NET_POPULATION_CHANGE) / AREA
             AS POPULATION_DENSITY,
       HAPPINESS_SCORE,
       category
FROM ADISCOURSEWORK.HAPPINESS_REPORT
INNER JOIN ADISCOURSEWORK.DATE D ON D.ID = ADISCOURSEWORK.HAPPINESS_REPORT.DATE_ID
INNER JOIN ADISCOURSEWORK.COUNTRY C ON C.ID = ADISCOURSEWORK.HAPPINESS_REPORT.COUNTRY_ID
WHERE YEAR BETWEEN 2015 AND 2019;
```

Рисунок 4.7 - Скрипт для створення представлення, що описує звіт про рівень щастя

Далі створимо представлення, що описує вплив різних причин смертності на щастя протягом 2015-2019 років. У представленні присутня також нормалізація деяких полів. Наприклад, замість номінального показника споживання кількості смертей з тієї чи іншої причини, будемо використовувати нормалізоване значення кількості смертей з тієї чи іншої причини на душу населення.

```sql
CREATE VIEW ADISCOURSEWORK.DEATHS_REASONS_INFLUENCE_ON_HAPPINESS AS
    SELECT C.NAME AS COUNTRY_NAME,
           YEAR,
           DR.NAME AS REASON_NAME,
           COUNT / (AVERAGE_POPULATION::REAL - (2020-YEAR) * NET_POPULATION_CHANGE) * 10000 AS COUNT
    FROM ADISCOURSEWORK.DEATH_REPORT
    INNER JOIN ADISCOURSEWORK.COUNTRY C ON C.ID = DEATH_REPORT.COUNTRY_ID
    INNER JOIN ADISCOURSEWORK.DATE D ON D.ID = DEATH_REPORT.DATE_ID
    INNER JOIN ADISCOURSEWORK.DEATH_REASON DR ON DR.ID = DEATH_REPORT.REASON_ID
    WHERE YEAR BETWEEN 2015 AND 2019 AND COUNT IS NOT NULL;
```

Рисунок 4.8 - Скрипт для створення представлення, що описує вплив різних причин смертності на щастя

3.3. Підготовка даних до аналізу

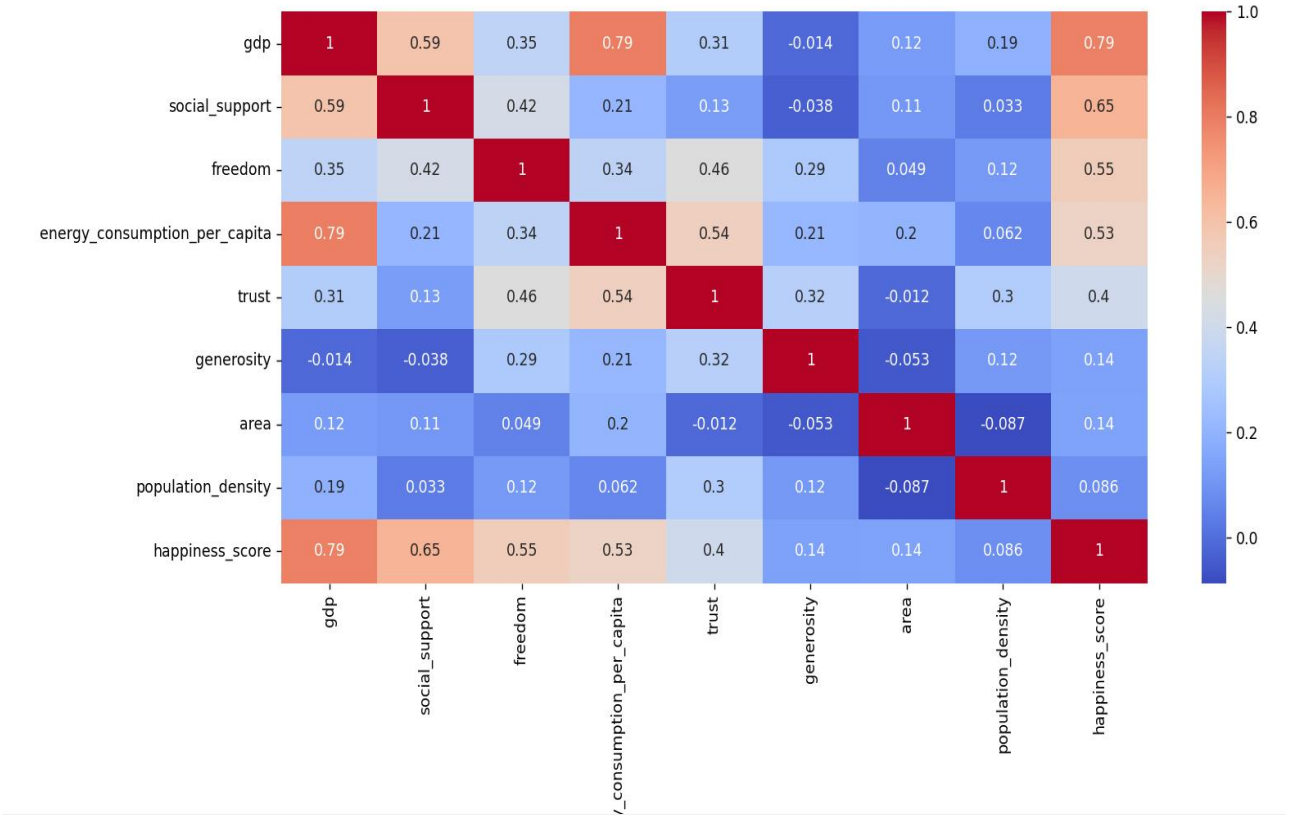

Рисунок 4.9 - Кореляційна матриця до добору параметрів

Оскільки параметри area (кореляція із happiness_score 0.062), generosity (кореляція із happiness_score 0.14) і population_density (кореляція із happiness_score -0.045) мають низьку значимість, то їх було видалено із аналізу.

Також у зв’язку з мультиколінеарністю було видалено параметр energy_consumption_per_capita (кореляція із gdp 0.79), social_support(кореляція із gdp 0.59).

Отже, у кінці кінців з представлення main_view залишись такі параметри(представлена матриця кореляції).

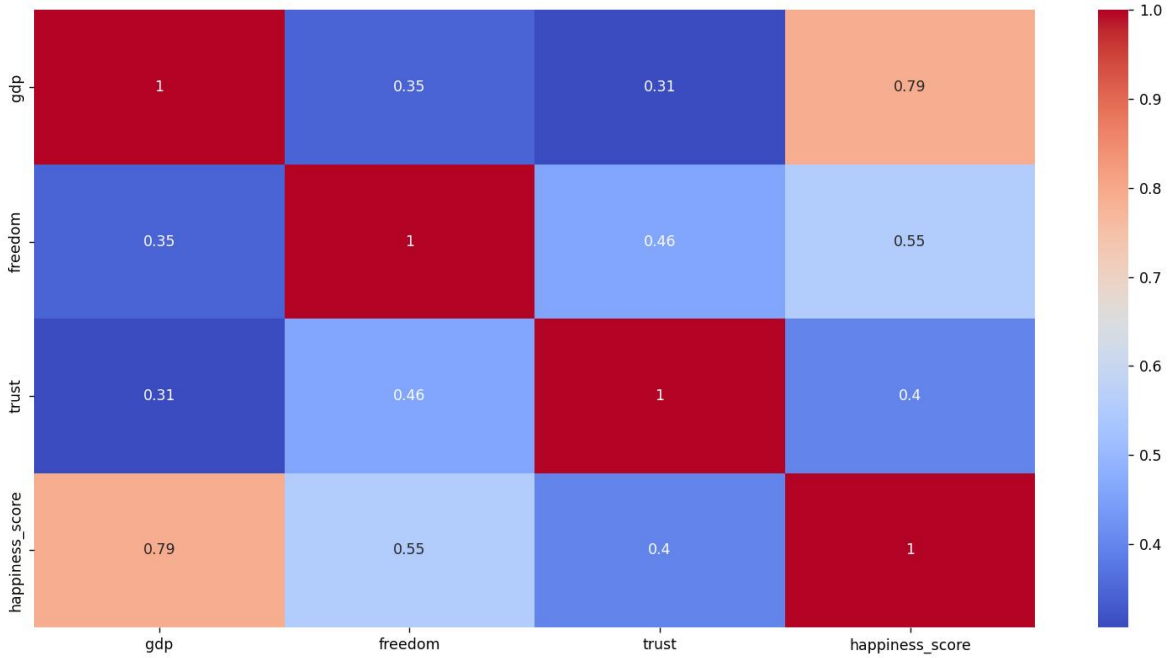

Рисунок 4.10 - Кореляційна матриця після добору параметрів

*Зауваження: happiness_score є не предиктором, а фактичними значенням, тому значення значення, наприклад, на перетині happiness_score та gdp, що рівне 0.79, є не великим значенням кореляції, яке потрібно прибрати, а значення залежності happiness_score від предиктора gdp.

Розглянемо інше представлення - представлення із причинами смертності. Проаналізувавши таблицю кореляції більш ніж 20 різних причин смертності і їх впливу на рівень щастя населення країн, прийшли до такого висновку, що варто вилучити з розгляду всі параметри, що не задовольняють двома умовам:

- значимість цих параметрів (кореляція з happiness_score) за модулем не менша за 0.3;
- кореляція між цими параметрами за модулем не більша 0.6;

Отримані параметри “примішаємо” до першого представлення і отримаємо кінцевий результат даних, підготовлених до інтелектуального аналізу.

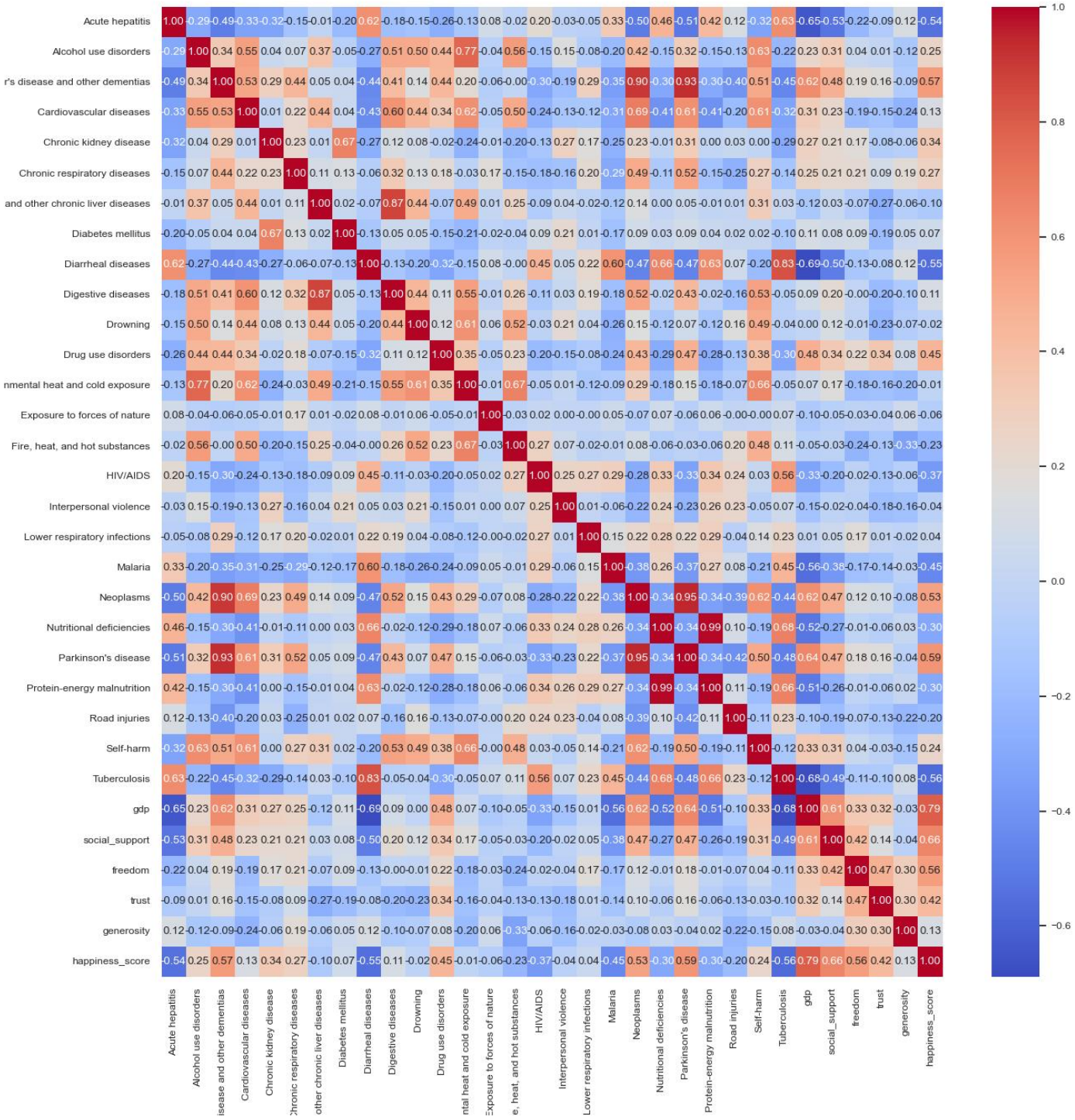

Рисунок 4.11 - Загальна кореляційна матриця до добору

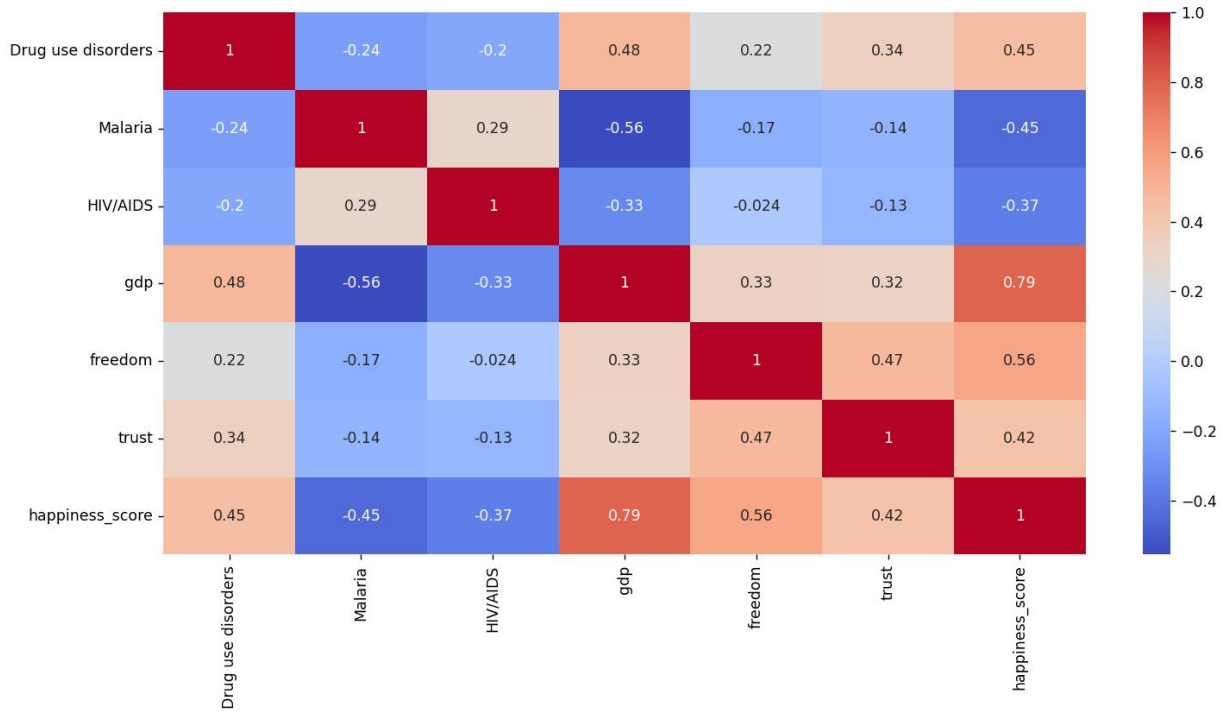

Рисунок 4.12 - Загальна кореляційна матриця після добору

Отже, результатом роботи над цим розділом стали дані, повністю підготовлені до інтелектуального аналізу.

4. ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ

4.1. Обґрунтування алгоритмів для побудови моделі класифікації

Класифікація - один із розділів машинного навчання, присвячений вирішенню наступного завдання. Є множина об'єктів (ситуацій), розділених певним чином на класи. Задано кінцева множина об'єктів, про котрі відомо, до яких класів вони належать. Ця множина називається навчальною вибіркою. Класова приналежність інших об'єктів невідома. Потрібно побудувати алгоритм, який здатен класифікувати довільний об'єкт із вихідної множини за одним із вищезгаданих класів. Існує багато алгоритмів класифікації. Ось кілька із них.

Логістична регресія – це багатовимірна регресія, загальною метою якої є аналіз зв'язку між кількома незалежними змінними (регресорами/провісниками) та залежною змінною. Бінарна логістична регресія використовується, якщо змінна двійкова, тобто змінна може приймати лише два значення. За допомогою логістичної регресії можна оцінити ймовірність подій для конкретного об’єкта (здоровий/хворий). У цьому випадку, якщо фактичним значення алгоритму регресії є не істинне значення, а ціле, можна розглядати класифікацію цього алгоритму, якщо кожне вихідне значення регресії відповідає певному класу.

Як відомо всі регресійні моделі можуть бути записані у вигляді формули:

$$y = F(x_1, x_2, ..., x_n).$$

Наприклад, в багатомірній лінійній регресії вважають, що залежна змінна є лінійною функцією незалежних змінних:

$$y = a + b_1 x_1 + b_2 x_2 + ... b_n x_n \qquad (4.1)$$

Цю модель можна використовувати для оцінки ймовірності результату події за допомогою стандартних коефіцієнтів регресії.

У випадку коли кількість різних варіантів результатів роботи логістичної регресії більша двох, використовують функції Soft Max, яка екстраполює частковий варіант регресії для двох можливих значень змінної,

Дерева рішень - це ще одна таксономія методів графічної організації, заснована на послідовному процесі прийняття рішень. Дерево рішень містить вузли рішень, кожен з яких має гілку для кожного альтернативного рішення. Шанси вузлів (випадкові величини) також з'являються в дереві, а корисність кожної гілки розраховується в листках кожної гілки. Очікувана корисність будь-якого рішення може бути розрахована на основі зваженої суми всіх гілок рішення для всіх листів цієї гілки.

Характерною рисою цієї математичної моделі є те, що вона визначає процес прийняття рішень так, що відображає кожне можливе рішення, попередні та наступні події чи інші рішення, а також наслідки кожного остаточного рішення.

Оскільки буква є частиною даних з однією цільовою ознакою, це буде передбачення з урахуванням описової ознаки.

Random forest - алгоритми машинного навчання, включаючи використання ансамблів дерев рішень. Алгоритм поєднує в собі дві основні ідеї: метод Бреймана і метод випадкового підпростору, запропонований Тін Кам Хо. Цей алгоритм використовується для задач класифікації, регресії та кластеризації. Основна ідея полягає у використанні великої кількості дерев рішень, кожне з яких само по собі дає дуже низьку якісну класифікацію, але завдяки їх великій кількості результати хороші. RF (Random Forest) — це набір ключових дерев. У задачах регресії їхні відповіді усереднюються, а в задачах на класифікацію рішення приймаються більшістю голосів.

Градієнтний бустінг - це техніка для побудови ансамблів, де прогнозні моделі будуються не окремо, а послідовно. Ця техніка використовує ідею, що наступна модель буде вчитися на помилках попередньої моделі. Імовірність їх появи в наступних моделях неоднакова, і частіше будуть ті моделі, які дають найбільшу похибку. Прогнози можна вибрати з широкого діапазону моделей, таких як дерева рішень, регресія, класифікатори тощо. Оскільки провісник вчиться на попередніх помилках, для отримання справжньої відповіді потрібно

менше часу. Але треба вибрати критерій обережної зупинки, інакше це призведе до перенавчання.

Мною було прийнято рішення використати кілька методів класифікації, а саме: логістична регресія, дерево рішень та random forest(рандомізований ліс). Метою використання одразу кількох методів є порівняння їх ефективності та вибір одного найбільш точного.

Оцінити точність класифікаційної моделі можна за критерієм R^2 (коефіцієнт детермінації), що дасть нам змогу зрозуміти, чи достатньо математична модель описує життєву.

Формула для розрахунку стандартної похибки залишків:

$$RSE = \arg\min \sum_{i=1}^{n} \left| y_i - \beta_0 - \sum_{j=1}^{k} X_{ij}\beta_j \right|^2 = \arg\min_{\beta} \left\| y = X\beta \right\|^2 \qquad (4.2)$$

4.2. Побудова і тренування моделі

Для побудови моделі нами було розроблено функцію classification_function, що одразу генерує, навчає, тестує та оцінює якість регресійної моделі.

```python
def classification_function():
    df = pd.read_csv('../data/the_most_main_view.csv', sep=',', decimal='.')
    df = df.drop(columns=["Unnamed: 0", 'country_name', 'year'])
    df = df[df['category'].isna() == False]

    # correlation
    corr = df.corr().round(2)
    seaborn.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns,
                    annot=True, cmap=seaborn.color_palette("coolwarm", as_cmap=True))
    plt.show()
    df.drop(columns='happiness_score', inplace=True)
    X_data = df.drop(columns=['category'], inplace=False)

    # transforms 'Least Developed Country' into 2, etc...(ASC alphabetic order)
    label_encoder = LabelEncoder()
    Y_data = label_encoder.fit_transform(df['category'])

    X_train, X_test, Y_train, Y_test = train_test_split(
        X_data.values,
        Y_data,
        test_size=0.3,
        random_state=5)

    # classification algorithm
    decisionTreeClassification(X_train, Y_train, X_test, Y_test)
    randomForest(X_train, Y_train, X_test, Y_test)
    logisticRegression(X_train, Y_train, X_test, Y_test)
    gradientBoosting(X_train, Y_train, X_test, Y_test)

    seaborn.pairplot(df, hue='category', height=2.5)
    plt.savefig('../data/images/test.png')
    plt.show()
    pass
```

Рисунок 4.1 - Вигляд функції regression

Для початку видалимо непотрібну колонку Unnamed: 0, яка містить порядковий номер рядка у датасеті і не представляє ніякої цінності, country, year, які не використовуються у аналізі, і видалимо усі рядки, де значення колонки category (категорія, рівень розвиненості країни) відсутнє.

```python
df = pd.read_csv('../data/the_most_main_view.csv', sep=',', decimal='.')
df = df.drop(columns=["Unnamed: 0", 'country_name', 'year'])
df = df[df['category'].isna() == False]
```

Рисунок 4.2 - Видалення непотрібних даних

Далі побудуємо матрицю кореляції даних.

```
# correlation
corr = df.corr().round(2)
seaborn.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns,
                annot=True, cmap=seaborn.color_palette("coolwarm", as_cmap=True))
plt.show()
```

Рисунок 4.3 - Матриця кореляції даних

Завантажимо предиктори у окрему змінну X_data.

```
X_data = df.drop(columns=['category'], inplace=False)
```

Рисунок 4.3 - Змінна предикторів X_data

За допомогою функції LabelEncoder закодуємо усі значення рівня розвитку країни унікальними для кожного рівня розвитку числами і завантажимо до змінної Y_data, що являє собою представлення категорії, але вже у числовому варіанті.

```
# transforms 'Least Developed Country' into 2, etc...(ASC alphabetic order)
label_encoder = LabelEncoder()
Y_data = label_encoder.fit_transform(df['category'])
```

Рисунок 4.3 - Змінна Y_data

Для побудови моделі спочатку розіб’ємо задану вибірку на навчальну та тестову частини у відношенні 3:7 відповідно. Надалі навчальну вибірку використовуватимемо для навчання моделі, а тестову - для перевірки якості моделі на нових для неї даних задля чистоти експерименту. У результаті роботи функції отримали такі змінні:

- Xtrain - предиктори навчальної вибірки
- Ytrain - результати навчальної вибірки
- Xtest - предиктори тестової вибірки
- Ytest - результати тестової вибірки

```
X_train, X_test, Y_train, Y_test = train_test_split(
    X_data.values,
    Y_data,
    test_size=0.3,
    random_state=5)
```

Рисунок 4.2 - Розбиття вибірки на навчальну та тестову

У кінці кінців викличемо кілька функції побудови та дослідження класифікаційної моделі:

```
# classification algorithm
decisionTreeClassification(X_train, Y_train, X_test, Y_test)
randomForest(X_train, Y_train, X_test, Y_test)
logisticRegression(X_train, Y_train, X_test, Y_test)
gradientBoosting(X_train, Y_train, X_test, Y_test)
```

Рисунок 4.2 - Виклики функцій побудови та дослідження класифікаційної моделі

Далі розглянемо реалізацію функції класифікації деревом рішень decisionTreeClassification. Спочатку модель навчається, потом виводиться оцінка на тестовому наборі даних за критерієм R^2. У кінці функція будує матрицю помилок (confusion matrix), що описує де та скільки у відсотках модель зробила помилки, а де виявилася коректною.

```
def decisionTreeClassification(X_train, Y_train, X_test, Y_test):
    dtree_model = DecisionTreeClassifier().fit(X_train, Y_train)
    dtree_predictions = dtree_model.predict(X_test)

    # accuracy
    print('R^2 score for decision tree= ', dtree_model.score(X_test, Y_test))

    # confusion matrix
    cm = confusion_matrix(Y_test, dtree_predictions, normalize='all')
    plot_confusion_matrix(cm, classes, title='Confusion matrix for DecisionTreeClassifier')
    return dtree_model
```

Рисунок 4.2 - Вигляд функції decisionTreeClassification

Confusion matrix для класифікації методом дерева рішень має наступний вигляд:

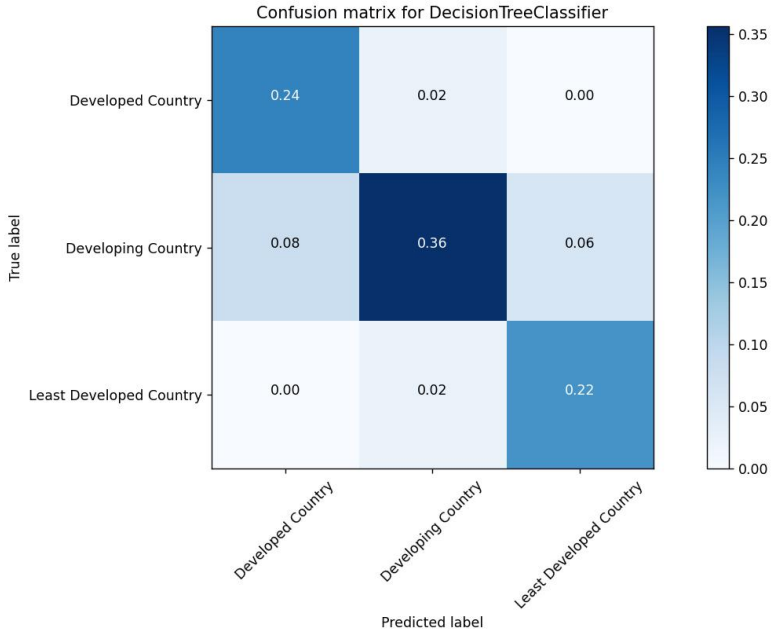

Рисунок 4.2 - Вигляд Confusion matrix для класифікації методом дерева рішень

Далі розглянемо реалізацію функції класифікації рандомізованим деревом рішень randomForest. Спочатку модель навчається, потом виводиться оцінка на тестовому наборі даних за критерієм R^2. У кінці функція будує матрицю помилок (confusion matrix), що описує де та скільки у відсотках модель зробила помилки, а де виявилася коректною.

```
def randomForest(X_train, Y_train, X_test, Y_test):
    random_forest = RandomForestClassifier().fit(X_train, Y_train)
    random_forest_prediction = random_forest.predict(X_test)

    # accuracy
    print('R^2 score for random forest: ', random_forest.score(X_test, Y_test))

    # confusion matrix
    cm = confusion_matrix(Y_test, random_forest_prediction, normalize='all')
    plot_confusion_matrix(cm, classes, title='Confusion matrix for RandomForestClassifier')
    return random_forest
```

Рисунок 4.2 - Вигляд функції randomForest

Confusion matrix для класифікації методом дерева рішень має наступний вигляд:

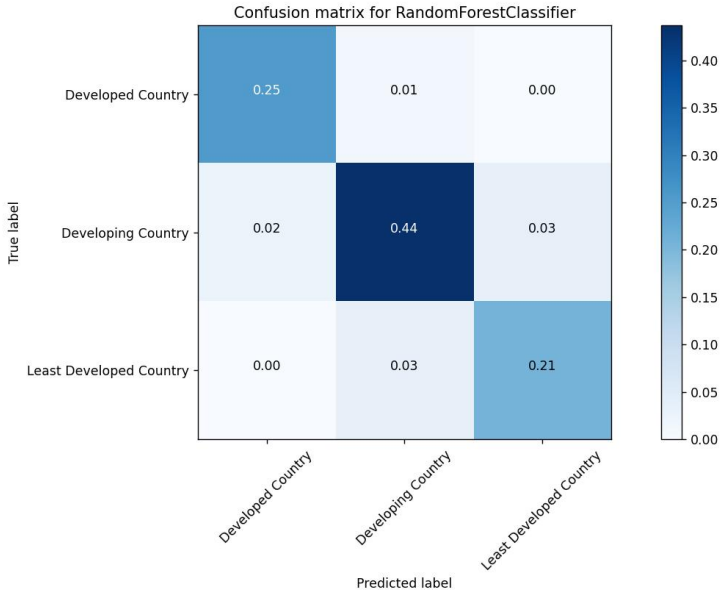

Рисунок 4.2 - Вигляд Confusion matrix для класифікації методом рандомізованого дерева рішень

Далі розглянемо реалізацію функції класифікації методом логістичної регресії із застосуванням функції softmax logisticRegression. Спочатку модель навчається, потом виводиться оцінка на тестовому наборі даних за критерієм R^2. У кінці функція будує матрицю помилок (confusion matrix), що описує де та скільки у відсотках модель зробила помилки, а де виявилася коректною.

```
def logisticRegression(X_train, Y_train, X_test, Y_test):
    soft_max = LogisticRegression(max_iter=500, random_state=0, multi_class='auto').fit(X_train, Y_train)
    soft_max_predict = soft_max.predict(X_test)

    # accuracy
    print('R^2 score for logistic regression: ', soft_max.score(X_test, Y_test))

    # confusion matrix
    cm = confusion_matrix(Y_test, soft_max_predict, normalize='all')
    plot_confusion_matrix(cm, classes, title='Confusion matrix for LogisticRegressionSoftMax')
    return soft_max
```

Рисунок 4.2 - Вигляд функції logisticRegression

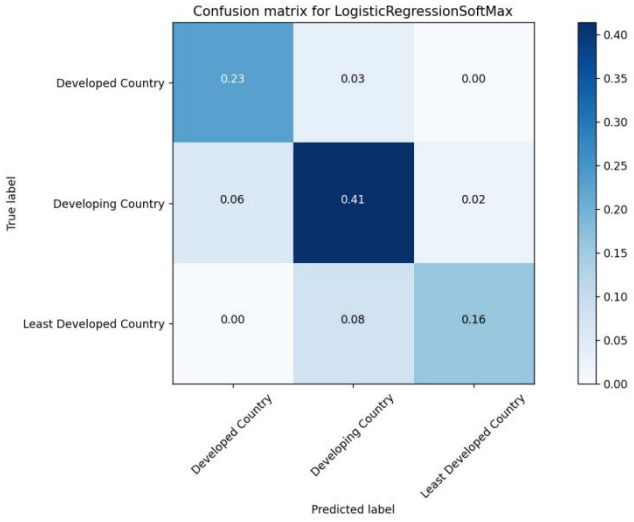

Рисунок 4.2 - Вигляд Confusion matrix для класифікації методом логістичної регресії

Наостанок проробимо усі ті самі дії над методом градієнтного бустінгу.

```
def gradientBoosting(X_train, Y_train, X_test, Y_test):
    gradient_boosting = GradientBoostingClassifier(learning_rate=1.0, random_state=0).fit(X_train, Y_train)
    gradientBoosting_predict = gradient_boosting.predict(X_test)

    # accuracy
    print('R^2 score for gradient_boosting: ', gradient_boosting.score(X_test, Y_test))

    # confusion matrix
    cm = confusion_matrix(Y_test, gradientBoosting_predict, normalize='all')
    plot_confusion_matrix(cm, classes, title='Confusion matrix for LogisticRegressionSoftMax')
    return gradient_boosting
```

Рисунок 4.2 - Вигляд функції gradientBoosting

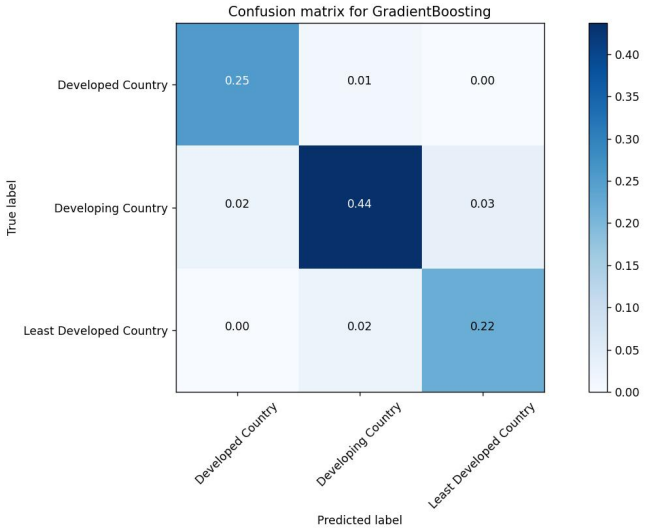

Рисунок 4.2 - Вигляд Confusion matrix для класифікації методом градієнтного бустінгу

У консолі маємо оцінки ефективності методів:

```
R^2 score for decision tree:  0.8045977011494253
R^2 score for random forest:  0.896551724137931
R^2 score for logistic regression:  0.8045977011494253
R^2 score for gradient_boosting:  0.9080459770114943
```

Рисунок 4.2 - Оцінки результатів роботи різних методів класифікації

Зрозуміло, що random forest та gradient boosting мають суттєву перевагу у точності над іншими досліджуваними методами. Тому для подальшого аналізу варто взяти або метод random forest або метод gradient boosting.

Отже, результатом роботи над цим розділом стала побудована модель класифікаційного аналізу, причому серед усіх протестованих вибрана найбільш точна. Досягнуто точності 86-91% за критерієм R^2.

Функція друку матриці конфузій наведено нижче:

```
def plot_confusion_matrix(cm, classes, title='Confusion matrix', cmap=plt.cm.Blues):
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], '.2f'),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.show()
    pass
```

Рисунок 4.2 - Функція побудови матриці конфузій

Для зручності дослідження варто побудувати графіки дискретних розподілів у залежності одразу від двох параметрів. Також точки, що відповідають реальним даним, розфарбовані у кольори, згідно з їх приналежністю до деякого класу.

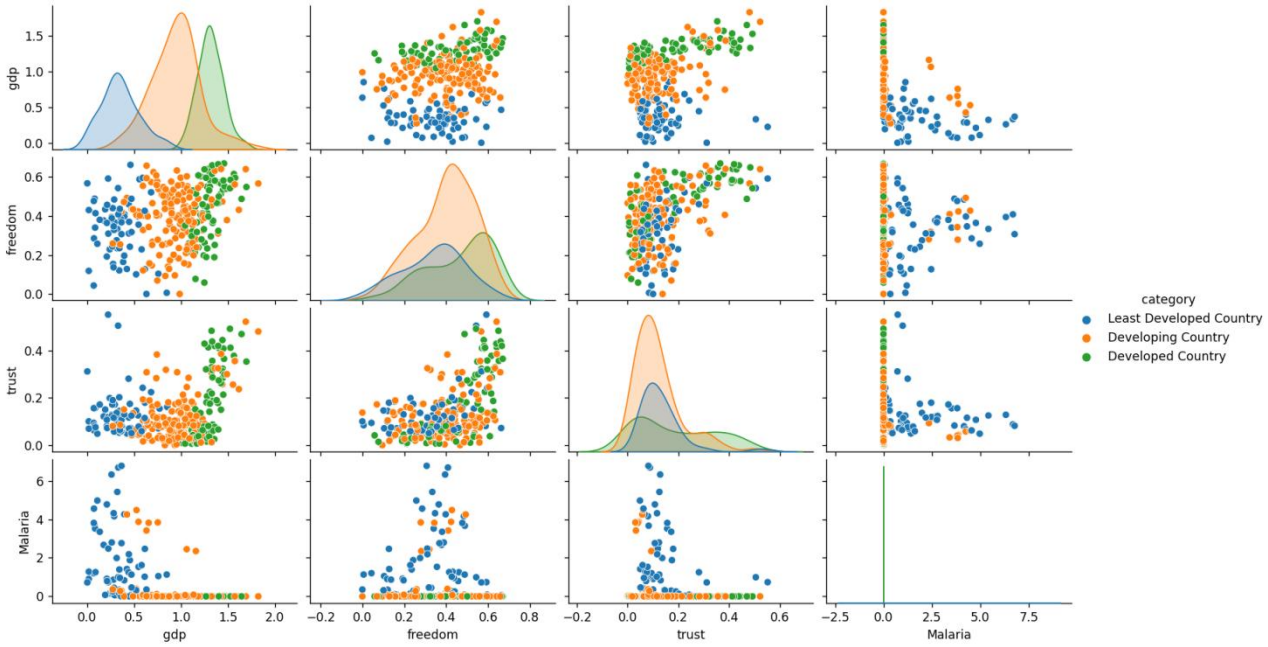

Рисунок 4.2 - Графіки дискретних розподілів за двома параметрами

5. ВИСНОВОК

В результаті виконання курсової роботи було розроблено алгоритм, який допомагає класифікувати країну за рівнем розвитку. Розглянуто основні підходи для реалізація такої моделі. Для реалізація поставленої задачі було використано мову програмування Python та різні бібліотеки: pandas, numpy, matlplot, seaborn, scikitlearn та інші.

У ході роботи було створено сховище даних, розроблено та застосовано ETL-процеси мовами SQL та Python. Усі скрипти було детально документовано у цьому звіті, а дані проілюстровано наочними графіками.

Було розроблено код, що передбачає за значеннями параметрів країн рівень її розвитку. Було реалізовано кілька моделей класифікації:

- логістична регресія
- градієнтний бустинг
- дерево рішень
- рандомізоване дерево рішень

Як показали досліди, найбільш точними виявилися методи рандомізованого дерева рішень та градієнтного бустингу. Менш точними, але теж достатньо оптимальними виявилися методи логістичної регресії у поєднанні із функцією SoftMax, що дозволяє вирішувати задачу, у якій відомо більш як 2 можливих значення факта, і метод звичайного дерева рішень.

На основі детального опису та проведеного аналізу предметної області інтелектуального аналізу даних для визначення рівня розвитку країни було отримано результати з високою точністю передбачення. Підтвердженням даних висновків є результати точності, яка складає до 0.91 для деяких видів класифікації на тестовому наборі даних. Результати досліджень показують, що дана модель може застосовуватися за призначенням і приносити користь. Отже, поставлені задачі були виконані.

ПЕРЕЛІК ПОСИЛАНЬ

1. Лекційні матеріали
2. Machine Learning. Coursera. Author: Andrew Ng
3. https://matplotlib.org /stable/tutorials/index
4. https://seaborn.pydata.org/generated/seaborn.heatmap.html
5. https://pandas.pydata.org/docs/
6. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

ДОДАТОК А ТЕКСТИ ПРОГРАМНОГО КОДУ

https://github.com/GeniusDP/CursachDataAnalysis - public repository with resources and code.