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 успіху фільмів за їх оцінюванням та оглядами»

Студента 2 курсу ІП-14 групи

Спеціальності: 121

«Інженерія програмного забезпечення»

Радзівіло В.А.

«ПРИЙНЯВ» з оцінкою

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис                    Дата

Київ - 2023 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-14 Семестр 4

**ЗАВДАННЯ**

**на курсову роботу студента**

|  |
| --- |
| Радзівіло Валерії Артемівни |

|  |  |
| --- | --- |
| 1.Тема роботи | Аналіз успіху фільмів за їх оцінюванням та оглядами |
|  | |
|  | |

|  |  |
| --- | --- |
| 2.Строк здачі студентом закінченої роботи | 03.06.2022 |

|  |  |
| --- | --- |
| 3. Вхідні дані до роботи | методичні вказівки до курсової робота, обрані дані з сайту |
| <https://www.kaggle.com/datasets/thedevastator/rotten-tomatoes-top-movies-ratings-and-technical> | |
|  | |

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 | 30.03.2023 |  |
| 2. | Визначення зовнішніх джерел даних | 5.04.2023 |  |
| 3. | Пошук та вивчення літератури з питань курсової роботи | 10.04.2023 |  |
| 4. | Розробка моделі сховища даних | 15.04.2023 |  |
| 4. | Розробка ETL процесів | 27.04.2023 |  |
| 5. | Обґрунтування методів інтелектуального аналізу даних | 01.05.2023 |  |
| 6. | Застосування та порівняння ефективності методів інтелектуального аналізу даних | 12.05.2023 |  |
| 7. | Підготовка пояснювальної записки | 21.05.2023 |  |
| 8. | Здача курсової роботи на перевірку | 03.06.2023 |  |
| 9. | Захист курсової роботи | 05.06.2023 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  |  | Радзівіло В.А. |
|  | (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) |

"5" червня 2023 р.

**АНОТАЦІЯ**

Пояснювальна записка до курсової роботи: 25 сторінок, 20 рисунки, 4 таблиці, 8 посилань.

Об’єкт дослідження: інтелектуальний аналіз даних.

Предмет дослідження: створення програмного забезпечення, що проводить аналіз даних з подальшим прогнозуванням та графічним відображенням результатів.

Мета роботи: проектування та реалізація сховища даних та ETL процесів, а також реалізація програмного забезпечення для отримання даних зі сховища та їх подальшого аналізу та прогнозування.

Дана курсова робота включає в себе: опис проектування, створення та заповнення сховища даних за даною задачею за допомогою фізичної моделі бази даних, опис створення програмного забезпечення для інтелектуального аналізу даних, їх графічного відображення та прогнозування за допомогою різних моделей.

СХОВИЩЕ ДАННИХ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, ФІЗИЧНА МОДЕЛЬ БАЗИ ДАНИХ, ELT ПРОЦЕСИ, LINEAR REGRESSION, RANDOM FOREST REGRESSOR, LASSO REGRESSION.

**Зміст**

[**Вступ** 5](#_Toc136596692)

[**1.Постановка задачі** 6](#_Toc136596693)

[**2.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ** 8](#_Toc136596694)

[**3.РОЗРОБКА СХОВИЩА ДАНИХ** 9](#_Toc136596695)

[3.1 Розробка моделі сховища даних 9](#_Toc136596696)

[3.2 Завантаження даних за допомогою ETL процесів 13](#_Toc136596697)

[3.3 Перевірка даних 13](#_Toc136596698)

[3.4 Поділ даних 17](#_Toc136596699)

[**4.ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ** 18](#_Toc136596700)

[4.1 Обґрунтування вибору методів інтелектуального аналізу даних 18](#_Toc136596701)

[4.2 Аналіз отриманих результатів для методу Linear Regression 19](#_Toc136596702)

[4.3 Аналіз отриманих результатів для методу Random Forest Regressor 20](#_Toc136596703)

[4.4 Аналіз отриманих результатів для методу Lasso Regression 21](#_Toc136596704)

[4.5 Порівняння отриманих результатів методів 22](#_Toc136596705)

[**ВИСНОВКИ** 24](#_Toc136596706)

[**ПЕРЕЛІК ПОСИЛАНЬ** 25](#_Toc136596707)

[**Додаток А Тексти програмного коду** 26](#_Toc136596708)

**Вступ**

Фільми мають важливе значення в житті людей. Вони є джерелом культурної спадщини, що допомагає зберегти та передати цінності й ідеї наступним поколінням. Фільми можуть надихати на нові ідеї, стимулювати наші почуття та емоції, допомагати розв'язувати проблеми та приймати важливі рішення. Вони також допомагають краще зрозуміти світ, у якому ми живемо, та почуватися зв'язаними з іншими людьми. Фільми навчають новим навичкам, збагачують нашу освіченість та можуть слугувати засобом розваги та відпочинку, знімаючи стрес і покращуючи настрій.

Аналіз даних в області фільмів може мати дуже значний вплив на розуміння споживачів та ефективність кінематографічної індустрії. Використання аналізу даних може допомогти кінокомпаніям визначити популярність різних жанрів та створити більш успішні фільми, залучаючи більше глядачів. Крім того, аналіз даних може допомогти зрозуміти, які елементи фільму найбільш сприймаються глядачами, такі як сюжет, головні герої, музика та багато іншого. Це може допомогти виробникам фільмів створювати більш привабливі продукти для своєї аудиторії. Крім того, аналіз даних може допомогти кінотеатрам і компаніям з продажу фільмів оптимізувати свої бізнес-процеси, збільшити прибуток та поліпшити якість обслуговування своїх клієнтів. Таким чином, використання аналізу даних в області фільмів може допомогти покращити якість продукту, збільшити прибуток та покращити взаємодію між кінематографічною індустрією та глядачами.

В рамках даної курсової роботи розроблено сховище даних відеоігор, на основі фізичної моделі бази даних, функціонал якої було розроблено за допомогою SQL скриптів, а саму роботу з базою даних представлено через реалізацію програмного забезпечення для імплементації ETL процесів та інтелектуального аналізу обраних даних.

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

**1.Постановка задачі**

Під час виконання курсової роботи необхідно виконати наступні завдання:

Створення сховища даних типу «сніжинка». Структура курсової роботи узгоджена з керівником. Створення ETL процесів для завантаження даних до сховища, їх отримання зі самого сховища за допомогою запитів.

Реалізувати спроектоване сховище даних з використанням MySQL.

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.

Для прогнозування буде використано методи Linear Regression, Random Forest Regressor, Lasso Regression.

Застосування найоптимальнішої моделі прогнозування на згладжувальних даних та порівняння отриманих результатів з прогнозуванням на початкових даних. Аналіз отриманих результатів, порівняння різних методів прогнозування на даній вибірці, отримання найоптимальнішого методу.

Використати мову програмування Python 3 для реалізації застосунку.

Вхідними даними будуть інформація про фільми та їх оцінювання на сайті Rotten Tomatoes.

**2.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ**

Сучасна кінематографія є однією з найбільш динамічно-розвиваючих галузей світової економіки, що пов'язана з розробкою, виробництвом та дистрибуцією фільмів. Фільми є популярною формою розваги та масової культури, яка привертає все більше глядачів. Успіх фільму залежить від багатьох елементів, таких як сценарій, режисура, акторська гра, монтаж та багато іншого. Аналіз даних може допомогти кінокомпаніям визначити популярність різних жанрів та створити більш успішні фільми. Крім того, аналіз даних може допомогти зрозуміти, які елементи фільму найбільш сприймаються глядачами, що дозволить створювати більш привабливі продукти для своєї аудиторії. Таким чином, якісна розробка та вибір елементів має ключову роль в створенні будь-якого фільму.

У програмному забезпеченні буде реалізовано наступну функціональність, що включає в себе:

* проектування сховища даних;
* створення ETL процесів для завантаження і оновлення даних;
* створення вибірки даних з сховища;
* інтелектуальний аналіз даних;
* використання декількох моделей прогнозування даних;
* прогнозування перспектив оцінювання фільмів;
* графічне відображення отриманих результатів та їх аналіз.

# **3.РОЗРОБКА СХОВИЩА ДАНИХ**

## 3.1 Розробка моделі сховища даних

Для вирішення поставленної перед нами задачі був обраний «rotten\_tomatoes\_top\_movies». Даний набір даних складається з оцінок та інформації про 1610 фільмів. Також були створенні додаткові таблиці для групування даних.

Таблиця 1 – Таблиця виміру мов фільмів dim\_languages

|  |  |
| --- | --- |
| Назва поля | Опис поля |
| idlanguages | Унікальний ідентифікатор мови |
| name | Назва мови |

Таблиця 2 – Таблиця виміру жанрів фільмів dim\_genres

|  |  |
| --- | --- |
| Назва поля | Опис поля |
| idgenres | Унікальний ідентифікатор жанру |
| name | Назва жанру |

Таблиця 3 – Таблиця виміру мов фільмів dim\_rating

|  |  |
| --- | --- |
| Назва поля | Опис поля |
| idrating | Унікальний ідентифікатор вікового рейтингу |
| rating\_name | Назва рейтингу |

Таблиця 4 – Таблиця фактів рейтингу фільмів від rotten\_tomatoes rotten\_tomatoes\_top\_movies

|  |  |
| --- | --- |
| Назва поля | Опис поля |
| aspect\_ratio | Розширення кадру |
| consensus | Висновок від критиків, щодо фільму |
| crew | Команда зйомки |
| critic\_score | Оцінка від критиків |
| director | Режисер |
| genre | Жанр |
| id | Код фльму |
| link | Посилання на огляд |
| people\_score | Оцінка від глядачів |
| producer | Продюсер |
| production\_co | Компанія виробництва |
| rating | Вікові обмеження на фільм |
| release\_date\_streaming | Дата релізу на стрімінгові платформи |
| release\_date \_theaters | Дата релізу в кінотеатрі |
| runtime | Хронометраж фільму |
| sound\_mix | Технологія використана для запису звуку |
| synopsis | Короткий опис фільму |
| title | Назва фільму |
| total\_ratings | Кількість оцінок |
| total\_revievs | Кількість оглядів |
| type | Тип фільму |
| view\_the\_collection | Колекція якій належить фільм |
| writer | Сценарист |
| year | Рік виготовлення фільму |

Створення таблиці виміру мов фільмів dim\_languages в сховищі даних:

CREATE TABLE `dim\_languages` (

`idlanguages` int NOT NULL AUTO\_INCREMENT,

`name` varchar(45) NOT NULL,

PRIMARY KEY (`idlanguages`),

UNIQUE KEY `name\_UNIQUE` (`name`)

)

Створення таблиці виміру жанрів фільмів dim\_genres в сховищі даних:

CREATE TABLE `dim\_genres` (

`idgenres` int NOT NULL AUTO\_INCREMENT,

`name` varchar(45) NOT NULL,

PRIMARY KEY (`idgenres`)

)

Створення таблиці виміру вікового рейтингу фільмів dim\_rating в сховищі даних:

CREATE TABLE `dim\_rating` (

`idrating` int NOT NULL AUTO\_INCREMENT,

`rating\_name` varchar(45) NOT NULL,

PRIMARY KEY (`idrating`),

UNIQUE KEY `rating\_name\_UNIQUE` (`rating\_name`)

)

Створення таблиці фактів rotten tomatoes оцінки фільмів fact\_rotten\_tomatoes в сховищі даних:

CREATE TABLE `rotten\_tomatoes\_top\_movies` (

`id` int NOT NULL AUTO\_INCREMENT,

`title` varchar(68) NOT NULL,

`year` int NOT NULL,

`synopsis` text,

`critic\_score` int NOT NULL,

`people\_score` int DEFAULT NULL,

`consensus` varchar(528) DEFAULT NULL,

`total\_reviews` int NOT NULL,

`total\_ratings` varchar(16) NOT NULL,

`type` varchar(25) NOT NULL,

`rating` varchar(112) NOT NULL,

`genre` varchar(63) NOT NULL,

`original\_language` varchar(24) NOT NULL,

`director` varchar(144) NOT NULL,

`producer` varchar(537) NOT NULL,

`writer` varchar(125) NOT NULL,

`release\_date\_theaters` varchar(20) NOT NULL,

`release\_date\_streaming` date NOT NULL,

`box\_office\_gross\_usa` varchar(7) NOT NULL,

`runtime` varchar(6) NOT NULL,

`production\_co` varchar(565) NOT NULL,

`sound\_mix` varchar(77) NOT NULL,

`aspect\_ratio` varchar(34) NOT NULL,

`view\_the\_collection` varchar(25) NOT NULL,

`crew` varchar(1045) NOT NULL,

`link` varchar(96) NOT NULL,

PRIMARY KEY (`id`)

)

## 3.2 Завантаження даних за допомогою ETL процесів

Для заповнення сховища даних відеоігор було реалізовано наступне:

1. Було зведено дані до одного типу. А саме
   1. Для таблиці виміру rotten\_tomatoes\_top\_movies мова була зведена до одного формату - до коду з таблиці dim\_language або 32 – інша мова, жанри були зведені до коду з таблиці dim\_genre або 24 – інший жанр, вікові рейтинги – аналогічно, самі таблиці були отримані за допомоги stored procedures : get\_languages, create\_dim\_genres, get\_rating відповідно.
   2. Для таблиці rotten\_tomatoes\_top\_movies runtime було зведено до кількості хвилин, яку триває фільм за допомоги функції transform\_runtime\_to\_minutes
   3. Для таблиці rotten\_tomatoes\_top\_movies ‘total\_ratings’ було зведено до мінімальної кількості оцінок за допомоги функції change\_total\_ratings
   4. Для таблиці rotten\_tomatoes\_top\_movies жанр був зведений до першого з перелічених в сеті за допомоги функції create\_dim\_genres
2. Було отримано дані з Stage зони для заповнення сховища даних. А саме:
   1. За допомоги функції read\_data було прочитано файл rotten\_tomatoes\_top\_movies, звідки йде основний датасет з оцінками, інформацією про фільми.
   2. За допомоги функції read\_data\_semi було прочитано файли dim\_genre, dim\_rating, dim\_language – допоможні таблиці для групування даних.

## 3.3 Перевірка даних

Для роботи з даними на мові Python ми використовуємо бібліотеку «pandas».

Для початку ми зчитуємо дані з файлу та виводимо основну інформацію про

наш датафрейм

def read\_data(path):

    # Reads the file with the given delimiter ","

    data = pd.read\_csv(path, sep=',')

    return data

def check\_data(data):

    # Checks the data

    print("\n Data check")

    print(data.head(10))

    print(data.info())

    print(data.describe())

    # Count na

    print("\n Count na")

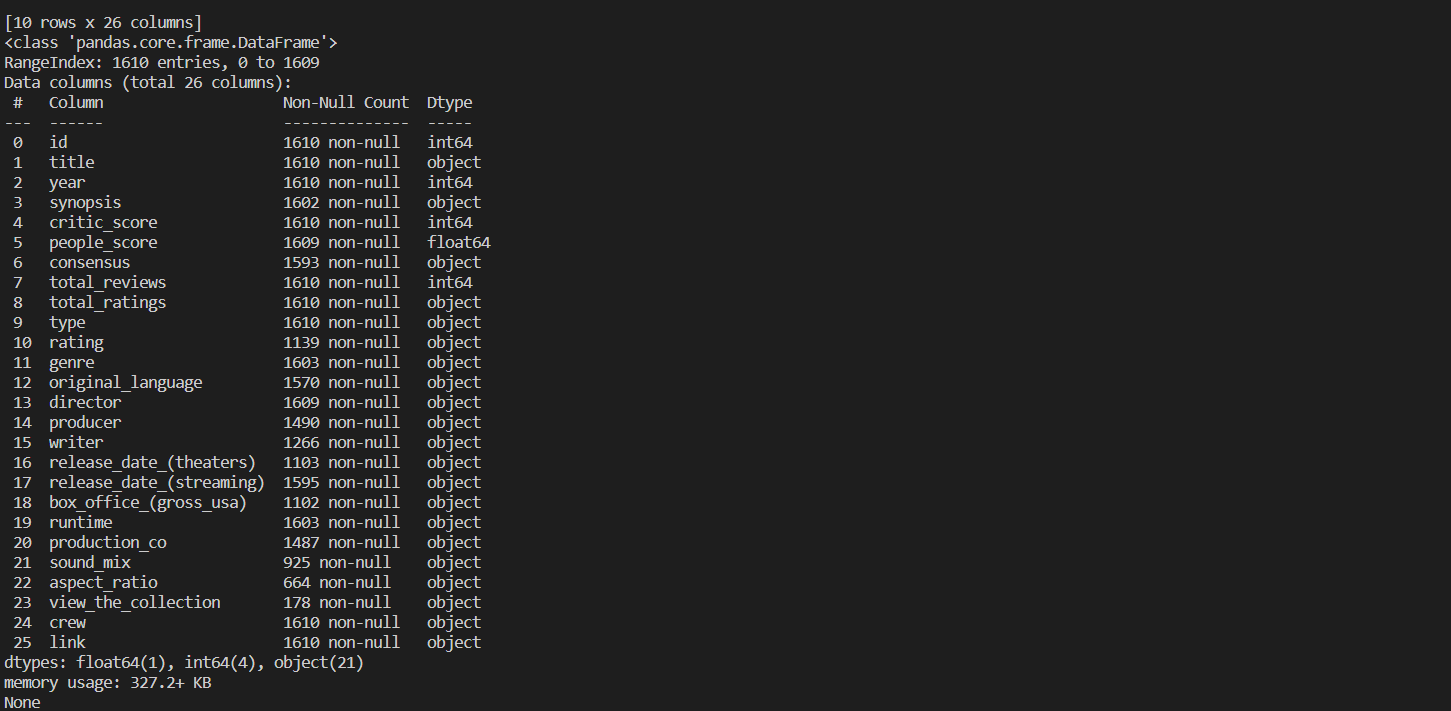
    print(data.isna().sum())

data\_path\_rotten\_tomatoes  = 'data/rotten\_tomatoes\_top\_movies.csv'

dataset\_rotten\_tomatoes = read\_data(data\_path\_rotten\_tomatoes)

check\_data(dataset\_rotten\_tomatoes)

Рисунок 3.1 – Код для виведення загальної інформації про датафрейм



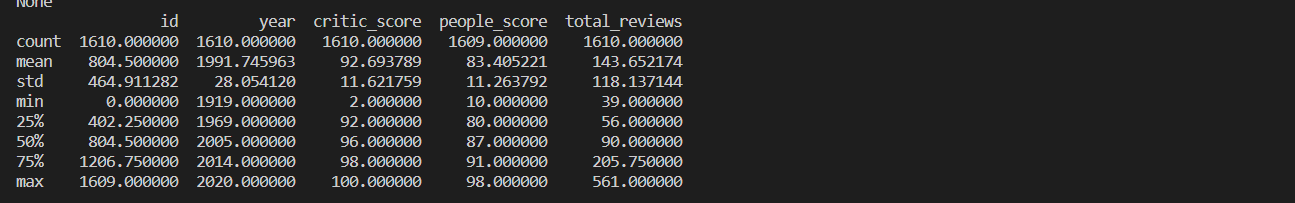




Рисунок 3.2 – Результат виконання коду, вивід загальної інформації про датафрейм

За цією інформацією, можна сказати, що відсутні значення менші за нуль, але є рядки з пропущенними значеннями. Також присутні декілька колонок, що є не релевантними для нашої задачі, тому їх варто прибрати, вони мають забагато пропущенних значень або унікальну інформацію, таку як назва.

dataset\_rotten\_tomatoes=dataset\_rotten\_tomatoes.drop(columns=['title','type','box\_office\_(gross\_usa)','view\_the\_collection','aspect\_ratio','sound\_mix','release\_date\_(theaters)','production\_co','consensus','synopsis','crew','link','director','producer','writer','release\_date\_(streaming)'])

Рисунок 3.3 - Код видалення непотрібних колонок

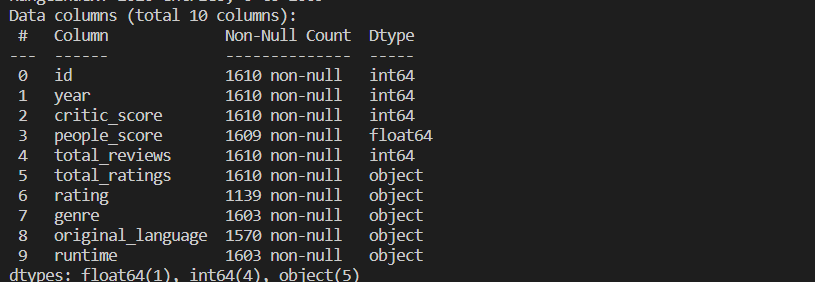


Рисунок 3.4 – Дані про колонки, що залишились

Після цього треба позбутись пропущенних значень та перетворити або згупувати дані для легшого аналізу. Спочатку, для колонок 'genre', 'rating', ‘original\_language’ треба замінити на відповідний код з таблиць dim\_genre, dim\_rating, dim\_language

    data\_path\_genre = 'data/dim\_genre.csv'

    dataset\_genre = read\_data\_semi(data\_path\_genre)

    data\_path\_rating = 'data/dim\_rating.csv'

    dataset\_rating = read\_data\_semi(data\_path\_rating)

    data\_path\_language = 'data/dim\_language.csv'

    dataset\_language = read\_data\_semi(data\_path\_language)

    dataset\_rotten\_tomatoes['genre'] = dataset\_rotten\_tomatoes['genre'].apply(change\_genre, dataset\_genre=dataset\_genre)

    dataset\_rotten\_tomatoes['rating'] = dataset\_rotten\_tomatoes['rating'].apply(change\_rating, dataset\_rating=dataset\_rating)

dataset\_rotten\_tomatoes['original\_language'] = dataset\_rotten\_tomatoes['original\_language'].apply(change\_original\_language, dataset\_language=dataset\_language)

Рисунок 3.5 – Код для знаходження відповідних кодів до полів.

Наступним кроком буде привести колонки ‘total\_ratings', 'runtime' до виду цілих чисел, для цього кількість оглядів буде переведена в мінімальну з зазначених кількостей оглядів, а хрономентраж фільму приведений до хвилин.

    dataset\_rotten\_tomatoes['total\_ratings'] = dataset\_rotten\_tomatoes['total\_ratings'].apply(change\_total\_ratings)

    dataset\_rotten\_tomatoes['runtime'] = dataset\_rotten\_tomatoes['runtime'].apply(transform\_runtime\_to\_minutes)

Рисунок 3.6 – Код для приведення кількості оцінювань та хронометражу фільму до цілого числа

Останній крок – заповнити пропущенні значення середніми. У випадках 'runtime' та 'people\_score' найкращим вирішенням проблеми пустих полів буде заповнити середніми значеннями.

    dataset\_rotten\_tomatoes['runtime'].fillna(dataset\_rotten\_tomatoes['runtime'].mean(), inplace=True)

    dataset\_rotten\_tomatoes['people\_score'].fillna(dataset\_rotten\_tomatoes['people\_score'].mean(), inplace=True)

    dataset\_rotten\_tomatoes['people\_score'] = dataset\_rotten\_tomatoes['people\_score'].astype(int)

Рисунок 3.7 – Код для заповнення середніми значеннями пропущенні поля хронометражу та оцінки від людей

Фінальна таблиця буде мати такий вигляд:

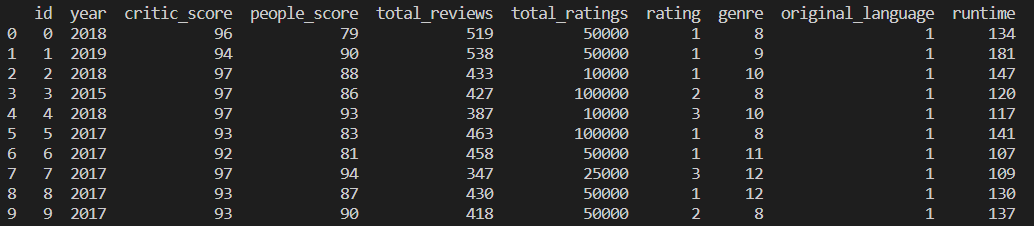


Рисунок 3.8 – Фінальний вигляд аналізованої таблиці після перевірки даних.

## 3.4 Поділ даних

Останнім кроком ми ділимо дані на тренувальні та тестові для подальшої

роботи з методами класифікації (рис 3.9).

    X = dataset\_rotten\_tomatoes[['year', 'people\_score', 'total\_reviews', 'total\_ratings',  'runtime', 'genre', 'rating', 'original\_language']]

    y = dataset\_rotten\_tomatoes['critic\_score']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

Рисунок 3.9 – Поділ даних на інформацію та результат

Щоб уникнути оверфіту, ми розділили наш набір даних на навчальні та тестові, а саме на 80% тренувальних та 20% даних, на яких буде проводитися тестування. Це дасть нам краще уявлення про те, як наші методи працюють на етапі тестування. Таким чином наші методи тестуються на невидимих даних, для кращого розуміння їх коректності використання для розв’язання поставленої нами задачі.

# **4.ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ**

## 4.1 Обґрунтування вибору методів інтелектуального аналізу даних

Аналіз фільмів зазвичай вимагає використання інтелектуального аналізу даних для виявлення тенденцій та моделей. Це допомагає кінокомпаніям прогнозувати популярність та прибутковість фільмів, а також визначати ті елементи, які найбільше сприймаються глядачами.

Мною було обрано два методи для прогнозування, а в подальшому і порівняні їх – це методи Linear Regression, Random Forest Regression та Lasso Regression.

Вибір був зроблений на основі суттєвих факторів, що полегшують проведення прогнозування. Linear Regression - це алгоритм, який базується на контрольованій навчальній області машинного навчання. Він успадковує лінійну залежність між його вхідними змінними та єдиною вихідною змінною, де вихідна змінна має суцільний характер. Завдяки своїй простоті він широко застосовується для моделювання прогнозів та умовиводів. Основна увага приділяється аналізу даних та попередній обробці даних. Отже, він має справу з різними даними, не турбуючись про деталі моделі. Проти, він працює ефективно, коли дані нормально поширюються. Таким чином, для ефективного моделювання необхідно уникати колінеарності.

Random Forest — це керований алгоритм машинного навчання, який складається з алгоритмів дерева рішень. Цей алгоритм застосовується в різних галузях, таких як банківська справа та електронна комерція, для прогнозування поведінки та результатів. Такий алгоритм є точнішим за алгоритм дерева рішень, він забезпечує ефективний спосіб обробки відсутніх даних. Також може створити обґрунтований прогноз без налаштування гіперпараметрів.

Lasso Regression - моделі зазвичай використовуються, щоб отримати швидке уявлення про те, які характеристики важливі для прогнозування змінної результату. Основна перевага регресійної моделі LASSO полягає в тому, що вона має можливість встановити коефіцієнти для функцій, які вона не вважає цікавими, до нуля. Це означає, що модель робить деякий автоматичний вибір функцій, щоб вирішити, які функції слід, а які не слід включати самостійно. Однак, відомо, що оцінки, отримані моделями LASSO, є відносно нестабільними, а це означає, що вони можуть сильно змінюватися під час навчання на дещо інших наборах даних.

## 4.2 Аналіз отриманих результатів для методу Linear Regression

Створюємо модель використовуючи метод Linear Regression і вираховуємо точність моделі, використовуючи коефіцієнт детермінації (R-квадрат), який показує, наскільки добре модель описує залежність між змінними. Чим ближче значення R-квадрат до 1, тим краще модель описує залежність.

def linear\_regression(X\_train, X\_test, y\_train, y\_test):

    model = LinearRegression()

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

    r2 = r2\_score(y\_test, y\_pred)

    print("Linear Regression R-squared: ", r2)

    scatter\_plot(y\_test, y\_pred, 'Linear Regression')

    return r2

Рисунок 4.1 – Створення оцінка та вивід графіку для Linear Regression

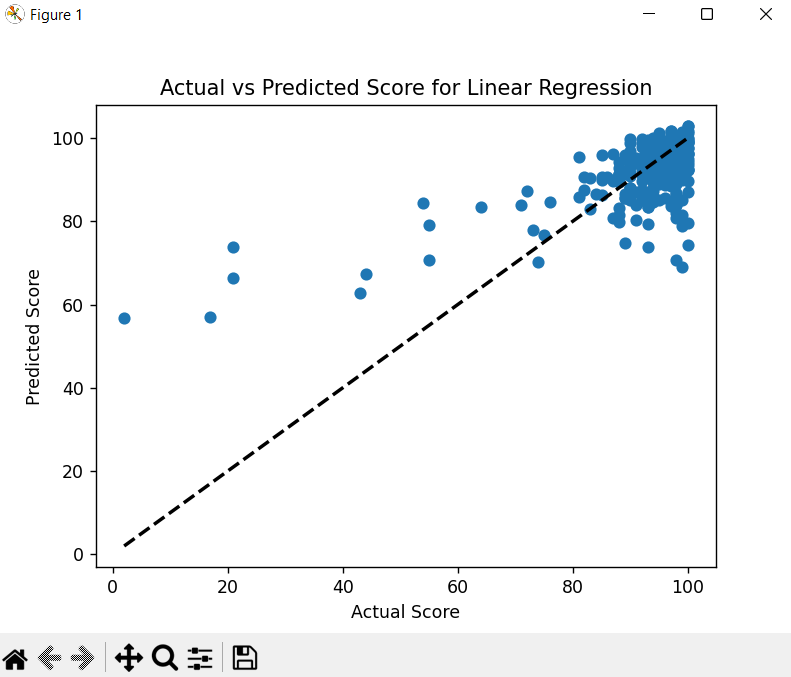
****

Рисунок 4.2 – Аналіз отриманої моделі за допомоги діаграми розсіювання Linear Regression.

****

Рисунок 4.3 – Коефіцієнт детермінації для Linear Regression.

З результатів бачимо, що ми отримали досить середню точність, не зважаючи на простоту і часто вживаність моделі. З діаграми розсіювання (рис 4.2) можна помітити, що модель краще впоралась з оцінюванням більше за 80. Значення коефіцієнту детермінації вказує на точність, даний коефіцієнт нам знадобиться для порівняння трьох моделей.

## 4.3 Аналіз отриманих результатів для методу Random Forest Regressor

Створюємо модель використовуючи метод Random Forest Regressor і вираховуємо точність моделі, використовуючи коефіцієнт детермінації.

def random\_forest\_regression(X\_train, X\_test, y\_train, y\_test):

    model = RandomForestRegressor()

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

    r2 = r2\_score(y\_test, y\_pred)

    print("Random Forest Regression R-squared: ", r2)

    scatter\_plot(y\_test, y\_pred, 'Random Forest Regression')

    return r2

Рисунок 4.4 – Створення оцінка та вивід графіку для Random Forest Regressor

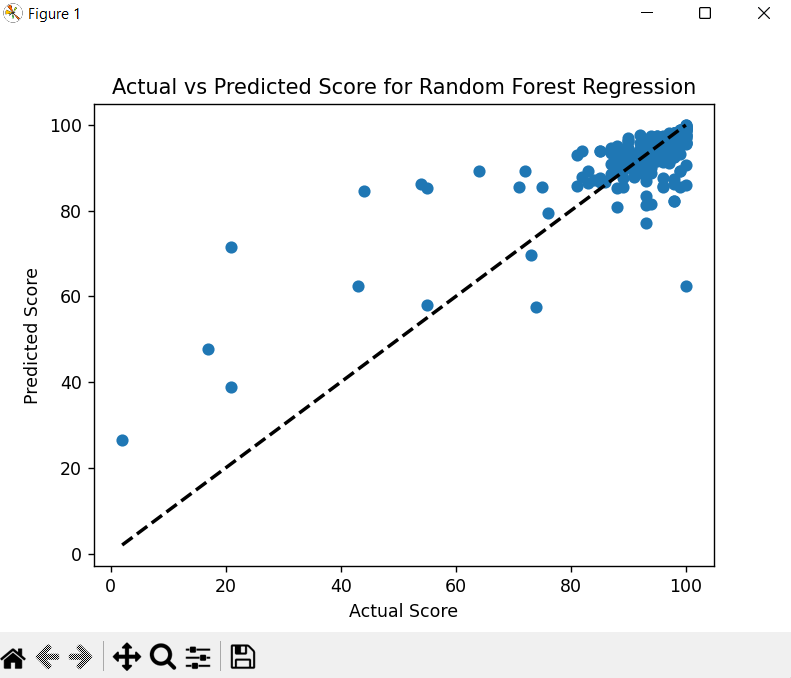


Рисунок 4.5 - Аналіз отриманої моделі за допомоги діаграми розсіювання Random Forest Regressor.

****

Рисунок 4.6 – Коефіцієнт детермінації для Random Forest Regressor.

Дана модель краще за попередню себе показала на різних значеннях, і набагато краще впоралась зі значеннями більші за 40. За коефіцієнтом детермінації в свою чергу можна визначити, що залежність змінних у Random Forest Regressor є більшою, тобто модель показала себе краще за попередню.

## 4.4 Аналіз отриманих результатів для методу Lasso Regression

Створюємо модель використовуючи метод Lasso Regression і вираховуємо точність моделі, використовуючи коефіцієнт детермінації.

def lasso\_regression(X\_train, X\_test, y\_train, y\_test):

    model = Lasso()

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

    r2 = r2\_score(y\_test, y\_pred)

    print("Lasso Regression R-squared: ", r2)

    scatter\_plot(y\_test, y\_pred, 'Lasso Regression')

    return r2

Рисунок 4.7 - Створення оцінка та вивід графіку для Lasso Regression

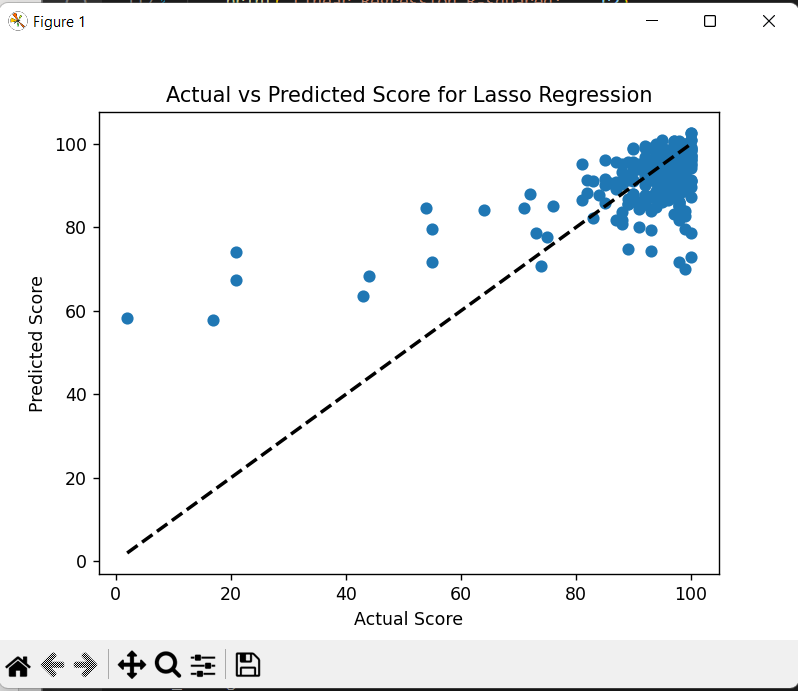


Рисунок 4.8 - Аналіз отриманої моделі за допомоги діаграми розсіювання Lasso Regression.

****

Рисунок 4.9 – Коефіцієнт детермінації для Lasso Regression.

За отриманими результатами можна прийти висновку, що Lasso Regression в даній задачі показало себе краще за Linear Regression, але набагато гірше за Random Forest Regressor. Модель, як і перша, найкраще справляється для значень більше 50. З коефіцієнту детермінації можна визначити, що цей метод є більш підходящим для цієї задачі за Linear Regression.

## 4.5 Порівняння отриманих результатів методів

Проаналізувавши окремо кожен із методів, що були мною використані під час прогнозування захворюваності на підготовлених даних, варто провести

порівняння даних методів.

def visualize\_regression\_results(mse\_scores):

    models = ['Linear Regression', 'Random Forest Regression','Lasso Regression']

    plt.bar(models, mse\_scores)

    plt.xlabel('Regression Models')

    plt.ylabel('R-square')

    plt.title('Comparison of Regression Models')

    plt.show()

Рисунок 4.10 – Порівняння точності методів Linear Regression, Random Forest Regressor та Lasso Regression

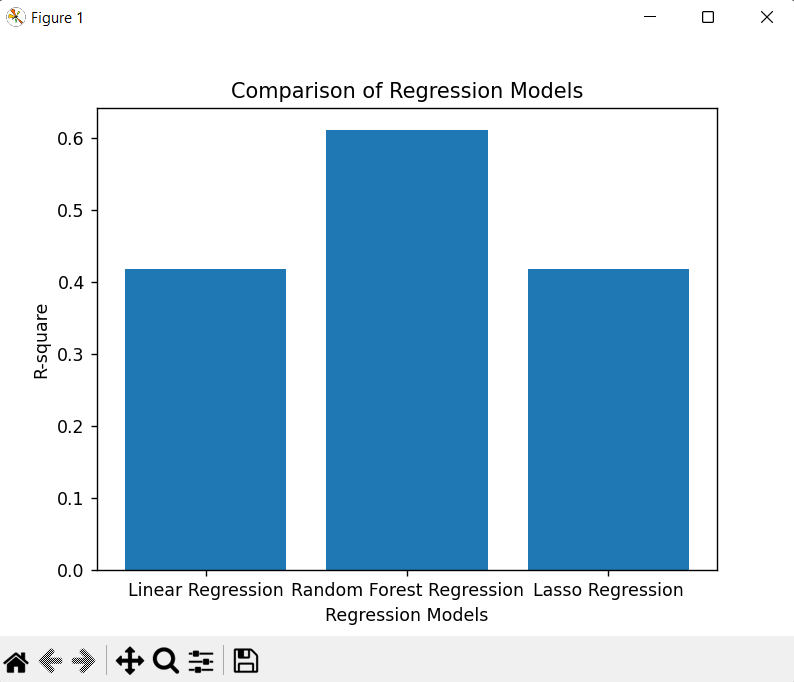
****

Рисунок 4.11 – Діаграма порівняння усіх регресійних моделей.

Отже, маємо, що максимально близьким до точності прогнозування в даних, на яких тренувалися ці методи, досяг Random Forest Regressor. Проте для прогнозування високих оцінок усі 3 методи справились однаково. Зважаючи на те, що оцінювання не може бути лише високим для даної задачі найбільш оптимальним і зручним буде використання Random Forest Regressor.

# **ВИСНОВКИ**

В результаті виконання курсової роботи було розроблено сховище даних відеоігор типу „сніжинка”. Реалізовано ETL процеси для завантаження даних, структуровану програму, яка б обробляла вхідні дані за допомогою інтелектуального аналізу з певних результатів вибірки та візуально відображала пророблений результат у ході статичних операцій. Для реалізації поставленої задачі було використано MySQL, мова програмування Python.

На основі детального опису та проведеного аналізу предметної області інтелектуального аналізу даних для прогнозування оцінювання критиків для нового фільму було отримано результати аналізу вибірки даних на мову, жанр, вікові обмеження та інші. Підтвердженням даних висновків є результати коефіцієнтів детермінації, а для найкращий результат - 0,64. Результати досліджень показують, що все таки кращою для прогнозування буде модель Random Forest Regressor, яка однаково добре справляється як з низьким, так і з високим оцінюванням. Linear Regression та Lasso Regression так само добре впорались для високих оцінок, але набагато гірше з низькими, сильно завищуючи їх. В деяких ситуаціях більше ніж в 2 рази, що є неприпустимим, тому що такі моделі прирівнюють погані фільми до хороших за оцінюванням, тому є непоказовими для людей з сфери кінематографії, тому що не показує які поєднання ключових характеристик буде оцінено гірше за інші.

Отже, поставлені задачі були виконані, а також планується розширення функціоналу для взаємодії з більшим сховищем даних фільмів, і прогнозування успіху фільму за ключовими параметрами, такими як мова, жанр, вікові обмеження, рік, хронометраж.

# **ПЕРЕЛІК ПОСИЛАНЬ**

1. Документація мови програмування Python. [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.python.org/3/>
2. Інформація про лінійну регресію [Електронний ресурс]- Режим доступу до ресурсу: [Проста лінійна регресія — Вікіпедія (wikipedia.org)](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%81%D1%82%D0%B0_%D0%BB%D1%96%D0%BD%D1%96%D0%B9%D0%BD%D0%B0_%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%96%D1%8F)
3. Бібліотека Pandas. [Електронний ресурс] – Режим доступу до ресурсу: <https://pandas.pydata.org/docs/>
4. Бібліотека Matplotlib. [Електронний ресурс] – Режим доступу до ресурсу: https://matplotlib.org/stable/
5. Бібліотека Sklearn. [Електронний ресурс] – Режим доступу до ресурсу: <https://scikit-learn.org/stable/user_guide.html>
6. Інформація про Random Forest. [Електронний ресурс] – Режим доступу до ресурсу: <https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/>
7. Інформація про Lasso Regression. [Електронний ресурс] – Режим доступу до ресурсу: <https://crunchingthedata.com/when-to-use-lasso/>
8. Датасет в Kaggle [Електронний ресурс] – Режим доступу до ресурсу: <https://www.kaggle.com/datasets/thedevastator/rotten-tomatoes-top-movies-ratings-and-technical>

**Додаток А Тексти програмного коду**

*студента групи ІП-14 ІІ курсу*

*Радзівіло В.А.*

(Обсяг програми (документа), арк., Кб)

*28 арк, 64 Кб*

(Вид носія даних)

*SSD*

(Найменування програми (документа))

*Тексти програмного коду* прогнозування доходів та факторів на популярність індустрії відеоігор

Python code:

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.metrics import r2\_score

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestRegressor

from sklearn.linear\_model import Lasso

def read\_data(path):

    # Reads the file with the given delimiter ","

    data = pd.read\_csv(path, sep=',')

    return data

def read\_data\_semi(path):

    # Reads the file with the given delimiter ";"

    data = pd.read\_csv(path, sep=';')

    return data

def check\_data(data):

    # Checks the data

    print("\n Data check")

    print(data.head(10))

    print(data.info())

    print(data.describe())

    # Count na

    print("\n Count na")

    print(data.isna().sum())

def transform\_runtime\_to\_minutes(runtime):

    if pd.isnull(runtime):

        return 108

    else:

        runtime = str(runtime)

        if 'h' in runtime:

            hours, minutes = runtime.split('h')

            hours = int(hours.strip())

            if len(minutes) !=0:

                minutes = int(minutes.strip().replace('m', ''))

            else:

                minutes = 0

            total\_minutes = hours \* 60 + minutes

        else:

            minutes = int(runtime.strip().replace('m', ''))

            total\_minutes = minutes

        return total\_minutes

def change\_genre(genre, dataset\_genre):

    genre = str(genre)

    if ',' in genre:

        type = genre.split(',')[0]

        id = dataset\_genre[dataset\_genre['name'] == type]['id'].values

    else:

        id = dataset\_genre[dataset\_genre['name'] == genre]['id'].values

    if id is None or len(id) == 0:

        return 24

    if len(id)>0:

        return id[0]

    return id

def change\_rating(rating, dataset\_rating):

    #if rating is nan then return 6

    if pd.isnull(rating):

        return 6

    rating = str(rating)

    if ' ' in rating:

        type = rating.split(' ')[0]

        #find in column name in dataset\_genre 'type' and return id

        id = dataset\_rating[dataset\_rating['rating\_name'] == type]['idrating'].values

    else:

        id = dataset\_rating[dataset\_rating['rating\_name'] == rating]['idrating'].values

    if id is None or len(id) == 0:

        return 6

    if len(id)>0:

        return id[0]

    else:

        return id

def change\_total\_ratings(total):

    if pd.isnull(total):

        return total

    else:

        total = str(total)

        if 'Fewer' in total:

            total = total = total.split('Fewer than ')[1]

        if '+' in total:

            total = total.replace('+', '')

        if ' ' in total:

            total = total.split(' ')[0]

        total = total.replace(',', '')

        return int(total)

def change\_original\_language(language, dataset\_language):

    if pd.isnull(language):

        return 32

    else:

        language = str(language)

        if ' ' in language:

            language = language.split(' ')[0]

        id = dataset\_language[dataset\_language['name'] == language]['id'].values

        if id is None or len(id) == 0:

            return 32

        if len(id)>0:

            return int(id[0])

        return id

def linear\_regression(X\_train, X\_test, y\_train, y\_test):

    # Linear regression

    model = LinearRegression()

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

    #score with r-squared

    r2 = r2\_score(y\_test, y\_pred)

    print("Linear Regression R-squared: ", r2)

    scatter\_plot(y\_test, y\_pred, 'Linear Regression')

    return r2

def random\_forest\_regression(X\_train, X\_test, y\_train, y\_test):

    # Random Forest regression

    model = RandomForestRegressor()

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

    r2 = r2\_score(y\_test, y\_pred)

    print("Random Forest Regression R-squared: ", r2)

    scatter\_plot(y\_test, y\_pred, 'Random Forest Regression')

    return r2

def lasso\_regression(X\_train, X\_test, y\_train, y\_test):

    # Lasso regression

    model = Lasso()

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

    r2 = r2\_score(y\_test, y\_pred)

    print("Lasso Regression R-squared: ", r2)

    scatter\_plot(y\_test, y\_pred, 'Lasso Regression')

    return r2

def visualize\_regression\_results(mse\_scores):

    models = ['Linear Regression', 'Random Forest Regression','Lasso Regression']

    plt.bar(models, mse\_scores)

    plt.xlabel('Regression Models')

    plt.ylabel('R-square')

    plt.title('Comparison of Regression Models')

    plt.show()

def scatter\_plot(y\_test, y\_pred, name):

    plt.scatter(y\_test, y\_pred)

    plt.xlabel('Actual Score')

    plt.ylabel('Predicted Score')

    plt.title('Actual vs Predicted Score for '+name)

    plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'k--', lw=2)

    plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

    # read file

    print("\n Reading data dataset\_rotten\_tomatoes")

    data\_path\_rotten\_tomatoes  = 'data/rotten\_tomatoes\_top\_movies.csv'

    dataset\_rotten\_tomatoes = read\_data(data\_path\_rotten\_tomatoes)

    data\_path\_genre = 'data/dim\_genre.csv'

    dataset\_genre = read\_data\_semi(data\_path\_genre)

    data\_path\_rating = 'data/dim\_rating.csv'

    dataset\_rating = read\_data\_semi(data\_path\_rating)

    data\_path\_language = 'data/dim\_language.csv'

    dataset\_language = read\_data\_semi(data\_path\_language)

    dataset\_rotten\_tomatoes=dataset\_rotten\_tomatoes.drop(columns=['title','type','box\_office\_(gross\_usa)','view\_the\_collection','aspect\_ratio','sound\_mix','release\_date\_(theaters)','production\_co','consensus','synopsis','crew','link','director','producer','writer','release\_date\_(streaming)'])

    dataset\_rotten\_tomatoes['genre'] = dataset\_rotten\_tomatoes['genre'].apply(change\_genre, dataset\_genre=dataset\_genre)

    dataset\_rotten\_tomatoes['rating'] = dataset\_rotten\_tomatoes['rating'].apply(change\_rating, dataset\_rating=dataset\_rating)

    dataset\_rotten\_tomatoes['original\_language'] = dataset\_rotten\_tomatoes['original\_language'].apply(change\_original\_language, dataset\_language=dataset\_language)

    dataset\_rotten\_tomatoes['total\_ratings'] = dataset\_rotten\_tomatoes['total\_ratings'].apply(change\_total\_ratings)

    dataset\_rotten\_tomatoes['runtime'] = dataset\_rotten\_tomatoes['runtime'].apply(transform\_runtime\_to\_minutes)

    # change na

    dataset\_rotten\_tomatoes['runtime'].fillna(dataset\_rotten\_tomatoes['runtime'].mean(), inplace=True)

    dataset\_rotten\_tomatoes['people\_score'].fillna(dataset\_rotten\_tomatoes['people\_score'].mean(), inplace=True)

    dataset\_rotten\_tomatoes['people\_score'] = dataset\_rotten\_tomatoes['people\_score'].astype(int)

    check\_data(dataset\_rotten\_tomatoes)

    # check\_data(dataset\_rotten\_tomatoes)

    #     # Select features and target variable

    X = dataset\_rotten\_tomatoes[['year', 'people\_score', 'total\_reviews', 'total\_ratings',  'runtime', 'genre', 'rating', 'original\_language']]

    y = dataset\_rotten\_tomatoes['critic\_score']

    # Split the dataset into training and testing sets

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

    scores = [

    linear\_regression(X\_train, X\_test, y\_train, y\_test),

    random\_forest\_regression(X\_train, X\_test, y\_train, y\_test),

    lasso\_regression(X\_train, X\_test, y\_train, y\_test)

    ]

    # Visualize the results

    #This code will generate a bar plot comparing the mean squared error (MSE) for each regression model. The lower the MSE, the better the model's performance.

    visualize\_regression\_results(scores)

MySQL Stored Procedures:  
get\_languages:

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_languages`()

BEGIN

SELECT DISTINCT trim\_first\_word(original\_language,"(")

FROM rotten\_tomatoes\_top\_movies

where original\_language!="" and original\_language!="1";

END

get\_rating:

CREATE DEFINER=`root`@`localhost` PROCEDURE `get\_rating`()

BEGIN

SELECT DISTINCT rating

FROM rotten\_tomatoes\_top\_movies

where rating!="";

END

create\_dim\_genres:

CREATE DEFINER=`root`@`localhost` PROCEDURE `create\_dim\_genres`()

BEGIN

insert into dim\_genre (name)

select distinct trim\_first\_word(s.genre,",")

from rotten\_tomatoes\_top\_movies as s;

END