Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних технологій факультету інформатики

Визначення успішності стартапу на основі машинного навчання

Текстова частина до курсової роботи
за спеціальністю «Інженерія програмного забезпечення» 121

Керівник курсової роботи
Доцент, кандидат фізико-математичних наук
Жежерун О. П.

_____

(підпис)

“____” _____ 2021 р.

Виконала студентка 4-го курсу
Хоменець В.С.
“____” _____ 2021 р.

Київ 2021

Календарний план виконання курсової роботи

Тема: Визначення успішності стартапу на основі машинного навчання

Календарний план виконання роботи:

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1.	Розгляд проблеми то способів її вирішення	Вересень-жовтень 2020р.	
2.	Пошук датасету	Жовтень-Листопад 2020р.	
3.	Вибір технологій та ознайомлення з ними	Грудень 2020р.	
4.	Розробка архітектури програми	Січень 2021р.	
5.	Створення моделі для передбачень	Лютий-березень 2021р.	
6.	Перегляд праці науковим керівником	Квітень 2021р.	
7.	Створення презентації роботи	Квітень 2021р.	
8.	Презентація роботи		

Студент Хоменець В.С _____

Керівник Жежерун О.П _____

“_____” _____ 2021 р.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

1. DS - Data Science
2. ML - Machine Learning
3. CRAN – Comprehensive R Archive Network
4. kNN – k-Nearest Neighbors

АНОТАЦІЯ

Робота присвячена створенню програмного забезпечення, яке на основі певних фактів про стартап, надасть передбачення про ймовірність його успіху. Передбачення успішності компанії стане корисним і для її засновників, і для її менеджменту, і для інвесторів. Для досягнення цієї мети було використано датасет, який містить інформацію про 472 підприємства. Кожна компанія має 116 характеристик. Було досліджено інформацію, яка зберігалася у датасеті, за допомогою побудови різноманітних графіків.

У проекті використано найпопулярніші на даний момент серед дата аналітиків і програмістів технології, які є добре задокументовані і загальнодоступні. Програмне забезпечення написане на мові програмування R, оскільки вона надає багато готових функції для роботи з неструктурованою інформацією.

ВСТУП

Багато людських ресурсів у світі марнується. У теперішній світовій економіці люди, у яких наявні кошти, визначають, яку саме ідею стартапу підтримати. Ці рішення найчастіше бувають дуже непрості, адже кошти обмежені, і неможливо підтримати усі ідеї.

В історії наявно дуже багато випадків, коли інвестори хибили, марнували капітал, тобто велику частину людських ресурсів. Яскравим прикладом цього є події 90-х років 20 століття. Тоді зародилася так звана dot-com bubble – економічна бульбашка, яка існувала через надмірну спекуляцію над інтернет-компаніями. Її результатом було потрачені нанівець мільярди доларів. Чому вона утворилася? Дослідники наводять багато причин, одна з основних це намагання компаніями застосувати новітні технології до своїх продуктів, навіть, якщо це ускладнювало користування ними. Проте також варто зазначити, що багато підприємств того часу змогли досягти тоді успіху і продовжують рости і сьогодні. Це такі компанії як Amazon.com, Adobe Systems та багато інших.

Чи можливо зменшити марнування людських ресурсів при виборі стартапу, в якого інвестувати? На даний момент існують технології, які дають можливість це зробити. За допомогою різноманітних методів машинного навчання можна досягнути високої точності передбачень.

Одним з найбільш затратних процесів у створенні моделі для передбачення є створення датасету, оскільки для цього необхідно досить тривалий час. У вільному доступі існують також уже зібрані датасети, які можна використати для своїх цілей. У даній практичній роботі було використано такий підхід.

Результатом виконання роботи стане програмний продукт, який зможе стати в нагоді працівникам, засновникам і інвесторам компанії, надаючи передбачення про її успішність на основі певних характеристик.

Дана робота складається з трьох частин.

У першій частині розповідається про використання мови програмування R для машинного навчання, про її синтаксис і середовище програмування.

Друга частина містить у собі дослідження про технології класифікації і регресії. Розглядаються причини і методи їх використання, а також різницю між ними.

Третя частина присвячена опису практичної частини даної роботи.

МОВА ПРОГРАМУВАННЯ R

1.1 Визначення

R — мова програмування і програмне середовище для статистичних обчислень, аналізу та зображення даних в графічному вигляді. [7]

1.2 Базовий опис

Мова програмування доступна для використання на всіх популярних операційних системах: різновидах UNIX платформ, Windows та MacOS. Її можна вільно використовувати у комерційних і некомерційних проектах, оскільки вона має ліцензію на розповсюдження GNU General Public License.

R, також відома як R lang, підтримує процедурну парадигму програмування. Вона надає можливість інтеграції з процедурами, які написані на мовах C, C ++, .Net, Python або FORTRAN.

Найчастіше серед працівників IT індустрії можна почути таке твердження: R – це мова програмування, яка написана математиками для виконання різних математичних операцій. Це дійсно так, на даній мові можна легко використовувати операції над множинами, обрахунки медіан та багато іншого. В основному R використовують дата аналітики для дослідження неструктурованої інформації, будуючи графічне відображення та створюючи звіти.

У мови програмування R існує багато причин її використовувати. Серед них можна виділити основні, які зображено нижче.

- Хороший інструмент для аналізу даних, їхньої візуалізації та створення моделей машинного навчання
- Гістограми, кругові діаграми, коробкові графіки – усе це доступно на мові R

- Можна використовувати різні статистичні методи, такі як класифікація, кластеризація та інші
- Відкрита для використання і підтримується на найпопулярніших операційних системах
- Велика кількість документації, наявно багато фахівців на трудовому ринку
- Доступно багато бібліотек для вирішення певних проблем, які можуть виникнути під час розробки проекту [11]

1.3 Історія

У 1975 – 1976 роках у світі з'явилася мова програмування S, метою якої було вирішувати проблеми в області статистики. У 80-х роках 19 століття Bell Laboratories, розробники мови S, зробили доступним вихідний код. Це у 1991 році дало можливість для Ross Ihaka та Robert Gentelman розпочати створення альтернативної реалізації S, яка спочатку мала назву S-PLUS. У 1993 році цей проект був оприлюднений. У 1995 році Martin Maechler переконав Ross Ihaka та Robert Gentelman зробити R вільнодоступним з ліцензію GNU General Public License. Згодом була створена команда R Development Core для управління подальшим розвитком R.

Дана мова програмування була названа R через дві причини:

1) Імена двох засновників Ross і Robert починаються з літери R
2) R є альтернативною реалізацією мови програмування S, назва якої складається з однієї літери

Перший офіційний реліз R вийшов у 1995 році. CRAN – мережа архівів мови R, було представлено 23 квітня 1997 року з 3 дзеркалами (місце, де можна завантажити архіви) та 12 пакетами. Стабільна бета-версія була випущена 29 лютого 2000 року. [7]

1.4 Середовище розробки

Для того, щоб програмувати на мові R декілька способів.

Один з варіантів, який є найбільш поширеним серед розробників, це завантаження середовища розробки R-Studio. Її завантажити можна з сайту https://rstudio.com/products/rstudio/download/. Вона йде вже з вбудованим інтерпретатором і має широкий функціонал, такий як, наприклад, побудова документів Word, HTML на основі формату R Markdown.

Іншим варіантом є завантаження плагіну для IntelliJ IDEA або PyCharm, який доступний за покликанням https://plugins.jetbrains.com/plugin/6632-r-language-for-intellij. Перевагами такого підходу є можливість використовувати всі корисні утиліти, які надає середовище програмування. Проте існують також і суттєві недоліки, серед яких, наприклад, відсутність підтримки деякий пакетів R. При використанні плагіну потрібно самостійно завантажити інтерпретатор, який доступний на офіційного сайті за покликанням https://cloud.r-project.org.

1.5 Синтаксис та типи даних

1.5.1 Коментарі

Для того, щоб вставити однорядковий коментар у коді, можна використати #.

```
company.character <- company.cleared[[2]]

# Приклад однорядкового коментаря

company.numeric <- create.last.funding.and.age.ratio.feature(company.numeric)
```

Рисунок 1

Для написання багаторядкового коментаря можна використати значення стрічки, яке ніяк не буде оброблятися.

```
company.character <- company.cleared[[2]]

"Приклад багаторядкового
коментаря"

company.numeric <- create.last.funding.and.age.ratio.feature(company.numeric)
```

Рисунок 2

1.5.2 Типи даних

R – мова динамічно типізована. Базові типи даних зображено нижче. [9]

- Logical (TRUE, FALSE)
- Numeric (1, 6.5)
- Integer (1L, 2L)
- Complex (1 + 2i, 3 + 5i)
- Character (“Привіт”, “світ”)
- Raw (“Сьогодні” 1 квітня = TRUE)

На базових типах побудовані типи-колекції, які зображені нижче.

- Vectors. Список однотипних елементів. Для створення використовується функція c(…)
- Lists. Список багатотипних елементів. Для створення використовується функція list(…)
- Matrices. Двовимірний датасет. Для створення використовується функція matrix(…)
- Arrays. Багатовимірний датасет. Для створення використовується функція array(…)
- Factors. Вектор, який зберігається разом з його унікальними значеннями. Для створення використовується функція factor(…)
- Data Frames. Структура для представлення таблиць. Для створення використовується функція data.frame()

1.5.3 Змінні

Для того, щоб присвоїти змінній значення існує три варіанти. [9]

1) Використовуючи <-

variable <- "Привіт"

2) Використовуючи ->

"Привіт" -> variable

3) Використовуючи =

variable = "Привіт"

Для видалення змінної з програмного середовища можна використати функцію rm(…).

rm(variable)

1.5.4 Оператори

- + (додавання)
- - (віднімання)
- * (множення)
- / (ділення)
- %% (остача від ділення)
- %/% (ціла частина від ділення)
- ^ (піднесення до степеню)
- >, <, ==, <=, >=, != (порівняння)
- & (логічний AND)
- | (логічний OR)
- ! (заперечення)
- && (логічний AND, при першому FALSE, далі не перевіряє)
- || (логічний OR, при першому TRUE, далі не перевіряє)
- : (інструмент для створення числових послідовностей)
- %in% (інструмент для перевірки чи елемент належить вектору)

- %*% (інструмент для множення матриці на її обернену матрицю) [9]

1.5.5 Цикли

В мові програмування цикли можна створювати за допомогою трьох ключових слів. [9]

- repeat. Виконує код певну кількість разів
- while. Виконує код, якщо певна умова дійсна
- for. Виконує код, якщо певна умова дійсна, перевірка відбувається в кінці циклу

Ключові слова для контролю циклів.

- break. Припиняє виконання циклу
- next. Припиняє виконання певної ітерації, переходить на наступну

1.5.6 Функції

Для створення функції потрібно вказати її ім’я і після ключового параметра function її параметри та тіло.

```
example.function <- function(text) {
  print(text)
}
```

Рисунок 3

1.6 R Markdown

1.6.1 Базовий опис

R Markdown – формат, який надає змогу створювати динамічно документи, у яких міститься проаналізована. У файлі R Markdown, який має розширення .rmd можна помістити код для виконання, а також опис про нього.

За допомогою R Markdown можна створювати звіти у різних форматах, таких як, наприклад, Word або HTML.

Даний інструмент використовувався у практичній курсової роботи і зарекомендував себе як дуже зручний.

1.6.2 Встановлення і використання

Для використання R Markdown спочатку потрібно встановити потрібний пакет пекедж.

```
install.packages("rmarkdown")
```

Потім потрібно створити файл з розширенням .rmd і для створення обробленого документа в R-Studio натиснути на копку Knit.

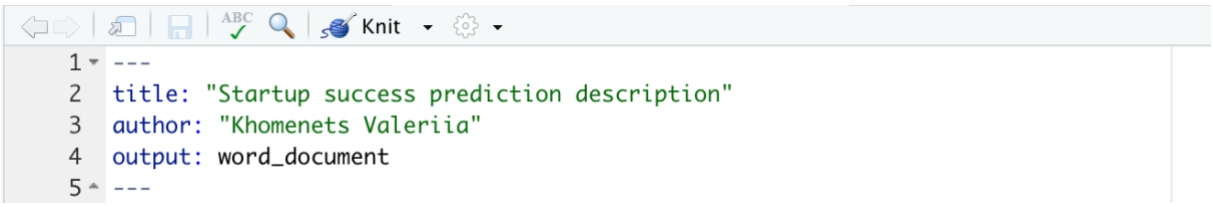

Рисунок 4

КЛАСИФІКАЦІЯ ТА РЕГРЕСІЯ

2.1 Класифікація

2.1.1 Визначення

Класифікація - це процес розбивання даних на задану кількість класів. Основною проблемою класифікації є визначення категорії, до якої належать нові дані. [11]

2.1.2 Базовий опис

Модель передбачень, яка створена за допомогою класифікації має на меті надати наближену функцію (f), яка з вхідних змінних (x), дасть на вихід дискретну змінну (y). [11]

Класифікація належить до категорії контрольованого навчання, при якому окрім вхідних даних, відомий також і результат їхнього оброблення.

Класифікацію можна розділити на три типи за кількістю класів, яких можуть набувати вихідна змінна.

1) Binary classification. Вихідні дані можуть приймати лише одне з двох значень. У практичній частині роботи це, наприклад, Success/Failure
2) Multi-class classification. Вихідні дані можуть приймати лише одне з багатьох значень. Наприклад, колір може бути або зелений, або червоний, але одночасно зеленим і червоним бути не може
3) Multi-label classification. Вихідні дані можуть приймати багато значень. Наприклад, відео може бути і історичним, і спортивним в один і той самий час

За часом навчання класифікацію можна розділити на два типи.

1) Класифікація з пізнім навчання. В такому випадку модель просто зберігає інформацію для тренування і чекає на інформацію для тестування. Передбачення робляться на основі найбільш

наближеної інформації для тренування. При такому підході час для тренування малий, проте для передбачення – великий. Прикладом класифікації з пізнім навчанням є kNN.

2) Класифікація з раннім навчанням. Модель обробляє інформацію для тренування одразу при її отриманні. При такому підході час для тренування великий, проте передбачення надається дуже швидко. Прикладом класифікації з раннім навчанням є Decision tree та Artificial Neural Networks.

2.1.3 Приклад використання

Існує безліч прикладів використання класифікації у світі.

Компанія Amazon щодо кожного замовлення, яке вони отримують, хоче передбачити: чи є це замовлення шахрайським? Вони мають деяку інформацію про кожне замовлення (його загальна вартість, чи відправляється замовлення на адресу, яку раніше використовував цей клієнт, чи адреса доставки така ж, як платіжна адреса власника кредитної картки). Amazon має багато даних про минулі замовлення, і знає, які з цих минулих замовлень були шахрайськими, а які ні. Тому вони вивчають закономірності, які допоможуть їм передбачити, коли надійдуть замовлення, чи є ці нові замовлення шахрайськими.

Gmail використовує класифікацію для побудови моделі, яка може передбачати чи є певний лист шкідливим. В цьому випадку листи, для яких уже визначено чи є вони спамом, можна використовувати як інформацію для навчання моделі. [12]

2.1.4 Decision Tree

Decision tree будує модель використовуючи структуру дерево. За допомогою вхідних атрибутів з їхніми класами Decision tree створює

послідовність правил, які можна потім використати для класифікації нової інформації.

Серед переваг даного методу класифікації є легкість для розуміння і візуалізації, невеликі затрати на підготовку інформації та можливість обробляти числові та інші категоричні типи даних. [13]

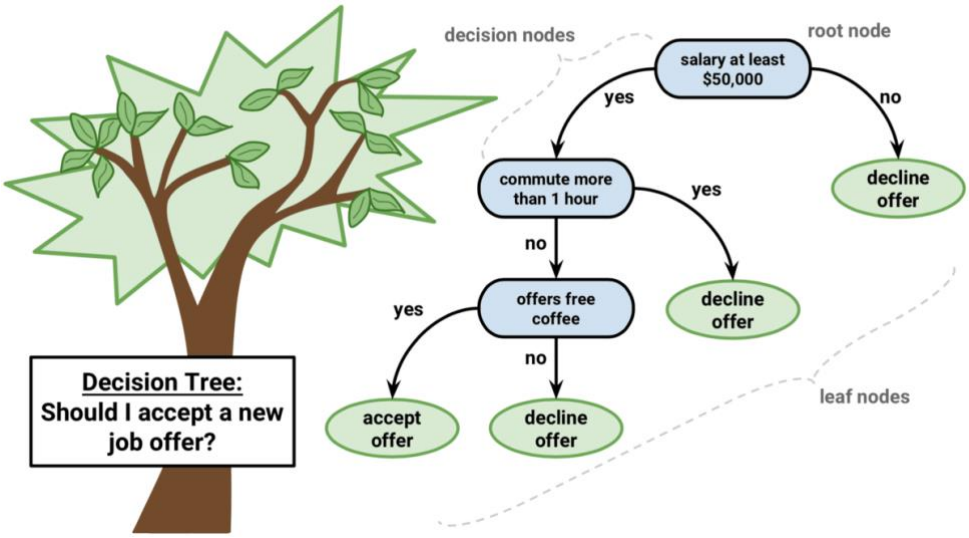

Рисунок 5 [16]

2.1.5 Random Forest

Random forest використовує декілька Decision Tree для створення передбачень з більшою точність. При отриманні інформації для тестування дерева видають певні результати, середнє значення яких буде кінцевим результатом. Для побудови моделі дереву надається інформація для тренування, яка є однакова за розмірами для всіх, проте з деякими перестановками для кожного дерева.

Серед переваг такого методу є зменшення надмірної кількості правильних передбачень під час тестування (over-fitting) та збільшення точності результату. [13]

2.1.6 Artificial Neural Networks

Artificial Neural Networks - це сукупність з’єднаних частин вводу та виводу, де кожне з’єднання має вагу. Дослідження даного методу класифікації було розпочато психологами та нейробіологами для розробки та перевірки обчислювальних аналогів нейронів. На етапі навчання мережа вчиться, регулюючи ваги, щоб мати можливість правильно класифікувати вхідну інформацію.

Основною перевагою Artificial Neural Networks є здатність показувати хороші результати на тестових даних. Дана модель добре працює з непідготовленою інформацією.

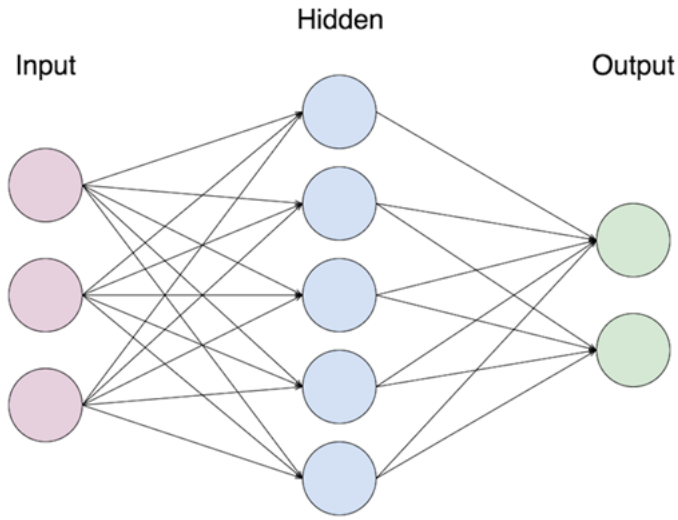

Рисунок 6 [17]

2.1.7 k-Nearest Neighbor

Оброблення даних для тренування відбувається під час запиту на отримання передбачення. Клас визначається через обрахунок найбільшої кількості однакових класів, яка зустрічається у k найближчих сусідів. Даний метод був використаний у практичній роботі для заповнення пропущених даних.

Даний алгоритм легкий для реалізації і високоефективний, якщо наявна велика кількість інформація для тренування. [13]

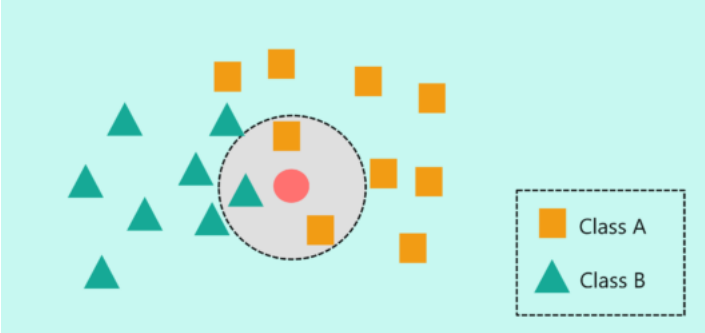

Рисунок 7 [18]

2.2 Регресія

2.2.1 Визначення

Регресія - це техніка, що використовується для моделювання та аналізу взаємозв'язку між залежною змінною та одною або багатьма незалежними змінними. [14]

2.2.2 Базовий опис

Регресія використовується для прогнозування числових змінних, які можуть набувати необмежену кількість значень. Наприклад, вона може використовуватися для встановлення взаємозв'язку між швидкісною їздою на автомобілі та кількістю дорожньо-транспортних пригод.

Загалом, кінцева модель, яка побудована за допомогою одного з алгоритмів регресії буде містити в собі певну математичну функцію.

Технік регресії є дуже багато, в даній роботі розглянуто дві найпопулярніші: Linear та Logistic Regression. Здебільшого алгоритми регресії відрізняються трьома характеристиками, які зображенні на малюнку нижче. [14]

Рисунок 8

2.2.3 Linear Regression

Linear Regression встановлює взаємозв'язок між залежною змінною (Y) та однією або кількома незалежними змінними (X), використовуючи пряму лінію, яка підходить найкраще (також відому як лінія регресії). У лінійній регресії залежна змінна може набувати значень з необмеженої множини, а незалежні змінні – з необмеженої або дискретної величин.

Дана техніка регресії може бути представлена рівнянням y = a + b * x.

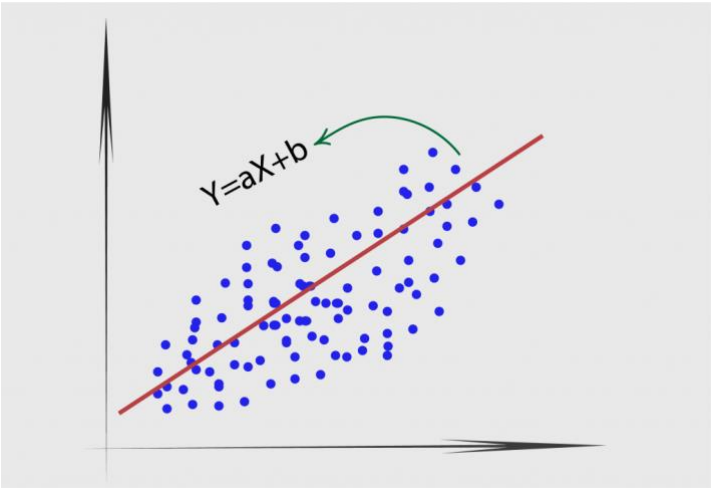

Рисунок 9 [19]

2.2.4 Logistic Regression

Logistic Regression використовується, щоб знайти ймовірність події, яка може набувати лише два значення (TRUE, FALSE). У практичні роботі, такий вид регресії застосовувався для передбачення успішності компанії (success/failure).

Logistic Regression використовує функцію log, оскільки така функція найбільше відповідає бінарному розподілу.

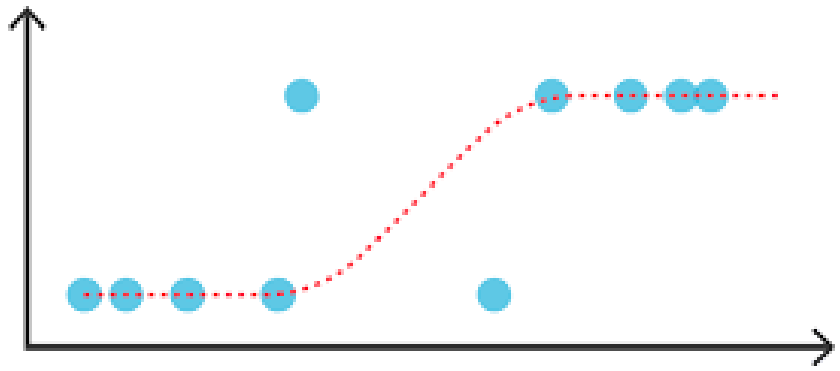

Рисунок 10 [20]

2.3 Порівняння класифікації та регресії

Класифікацію та регресії використовують для приблизно однієї і тієї самої цілі: створення моделі передбачень. Вони обидва для навчання моделі потребують вхідних і вихідних даних. Проте важливо розуміти, що класифікація та регресія суттєво відрізняються. Нижче наведено ключові відмінності.

- Залежна змінна у класифікації є дискретною, в той час як у регресії вона може набувати необмежену кількість значень. Logistic Regression дає передбачення для дискретної величини, проте воно надається у числовому форматі

- Для числових типів використовується регресія, а для інших типів з обмеженою кількістю категорій - класифікація
- Для передбачення зроблених за допомогою класифікації можна задавати їх точність [15]

2.4 Результати дослідження

Після дослідження алгоритмів класифікації та регресії стало зрозуміло, коли і за яких умов кожного з них краще застосовувати. Також було з’ясовано ключову різницю між ними.

ОПИС ПРАКТИЧНОЇ ЧАСТИНИ

3.1 Вступ

Практична частина присвячена створенню програмного забезпечення, яке дасть змого передбачати успіх стартапу. Для того, щоб такий функціонал можна було реалізувати, потрібно спочатку накопичити датасет з характеристиками компаній і результатами, які вони досягли. Після пошуків на різних інформаційних платформах було знайдено датасет, який знаходиться у відкритому для користування доступі. В ньому наявна інформація про 472 компанії, про кожну з яких описано 116 характеристик, які представлені в різних форматах: числовому, текстовому, логічному. Характеристики описують різні аспекти компанії, наприклад, історію інвестувань в неї та кількість співробітників. У практичній роботі буде виконано очищення даних (перетворення типів, заповнення пропущених значень), створення нових характеристик, обрання найважливіших змінних, а також створення моделі для передбачення успіху стартапу. Практична частина була виконана на мові програмування R, оскільки вона має багато переваг для оброблення неструктурованих даних, вона дає можливість досліджувати дані за допомогою побудови графіків, а також на ній реалізовано багато функцій для машинного навчання.

3.2 Бібліотеки

Для написання програми було використано бібліотеки, які зображено нижче. Вони містять функції для роботи з стрічками, для графічного відображення даних та для створення і тестування моделей.

```
library('stringr')
library('matrixStats')
library('VIM')
```

```
library(devtools)
install_github("vqv/ggbiplot")
library(ggbiplot)
library(ggplot2)
library(gcookbook)
library(caret)
library(randomForest)
library(devtools)
library(woe)
library(DBI)
library(caret)
library(ResourceSelection)
```

3.3 Ключові службові функції

Для створення даної програми було написано службові функції для оброблення інформації.

```
col.can.convert.to.two.stage.factor <- function(data.frame, col.name) {
  unique.values <- unique(data.frame[,col.name])
  unique.values.na.count <- sum(is.na(unique.values))
  unique.values.lenght <- length(unique.values)
  return(unique.values.na.count == 1 || unique.values.lenght == 2)
}

col.as.two.stage.factor <- function(data.frame, col.name) {
  data.frame <- col.without.whitespaces.tolower.by.name(data.frame, col.na
me)
  if (col.can.convert.to.two.stage.factor(data.frame, col.name)) {
    data.frame[,col.name] <- as.factor(data.frame[,col.name])
  }
  return(data.frame)
}

col.factor.transform.levels <- function(col.factor){
  col.factor.levels <- levels(col.factor)
  col.factor.levels.count <- length(col.factor.levels)
  if (col.factor.levels.count == 3 && "few" %in% col.factor.levels) {
    col.factor.levels <- c("none", "few", "many")
  } else if (col.factor.levels.count == 3 && "low" %in% col.factor.levels)
{
    col.factor.levels <- c("low", "medium", "high")
  } else if (length(col.factor.levels.count) == 4 && "low" %in% col.factor
.levels) {
    col.factor.levels <- c("none", "low", "medium", "high")
  }
  return(factor(col.factor, levels = col.factor.levels))
}

col.as.multi.factor <- function(col, col.name, is.ordered.name) {
```

```
  col <- col.without.whitespaces.tolower(col)
  if (is.ordered.name(col.name)) {
    col <- col.rename.for.order(col)
  }
  unique.values <- unique(col)
  col.factor <- as.factor(col)
  col <- col.factor.transform.levels(col.factor)
  return(col)
}

col.remove.outliers <- function(col) {
  col.quantile <- quantile(col, probs=c(.25, .75), na.rm = TRUE)
  right.parts <- 1.5 * IQR(col, na.rm = TRUE)
  col.result <- col
  col.result[col < (col.quantile[1] - right.parts)] <- NA
  col.result[col > (col.quantile[2] + right.parts)] <- NA
  return(col.result)
}

#columns constant variables
col.id.index <- 1
col.company.status.index <- 2
col.date.indexes <- c(13, 14)
col.number.indexes <- c(3:5, 10:11,15, 18:23,25,61,66,68:70,72,74,88,92,94
:96,98,99,102:116)
col.two.factor.indexes <- c(2, 12, 24, 27, 29:32, 34, 36, 38, 40:42, 44:53
, 55, 58, 63:64, 77:78, 81:86, 89:91)
col.multi.factor.indexes <- c(26, 28, 33, 35, 37, 39, 43, 54, 56, 57, 59,
60, 65, 67, 71, 73, 75, 76,79, 80, 87, 93, 97, 100, 101)
col.multi.factor.ordered.indexes <- c(26, 28, 43, 56, 57, 59, 67, 71, 73,
75,76, 79, 80, 87, 93)
col.characters.indexes <- c(6:9, 16:17, 62)
col.founders.skills.indexes <- c(27, 28, 29, 30, 31, 32, 33, 34, 35, 36)
col.not.applicable.no.confusion.index <- 37
```

3.4 Зчитування з файлу

Для початку зчитаємо з файлу з розширення .csv датасет і відобразимо усі колонки, які у ньому є.

```
company <- read.csv(file="~/Desktop/курсова/startupPrediction/data.csv", h
eader=TRUE, as.is=TRUE)
colnames(company)

##   [1] "Company_Name"
##   [2] "Dependent.Company.Status"
##   [3] "year.of.founding"
##   [4] "Age.of.company.in.years"
##   [5] "Internet.Activity.Score"
##   [6] "Short.Description.of.company.profile"
```

```
##   [7] "Industry.of.company"
##   [8] "Focus.functions.of.company"
##   [9] "Investors"
##  [10] "Employee.Count"
##  [11] "Employees.count.MoM.change"
##  [12] "Has.the.team.size.grown"
##  [13] "Est..Founding.Date"
##  [14] "Last.Funding.Date"
##  [15] "Last.Funding.Amount"
##  [16] "Country.of.company"
##  [17] "Continent.of.company"
##  [18] "Number.of.Investors.in.Seed"
##  [19] "Number.of.Investors.in.Angel.and.or.VC"
##  [20] "Number.of.Co.founders"
##  [21] "Number.of.of.advisors"
##  [22] "Team.size.Senior.leadership"
##  [23] "Team.size.all.employees"
##  [24] "Presence.of.a.top.angel.or.venture.fund.in.previous.round.of.inv
estment"
##  [25] "Number.of.of.repeat.investors"
##  [26] "Number.of..Sales.Support.material"
##  [27] "Worked.in.top.companies"
##  [28] "Average.size.of.companies.worked.for.in.the.past"
##  [29] "Have.been.part.of.startups.in.the.past."
##  [30] "Have.been.part.of.successful.startups.in.the.past."
##  [31] "Was.he.or.she.partner.in.Big.5.consulting."
##  [32] "Consulting.experience."
##  [33] "Product.or.service.company."
##  [34] "Catering.to.product.service.across.verticals"
##  [35] "Focus.on.private.or.public.data."
##  [36] "Focus.on.consumer.data."
##  [37] "Focus.on.structured.or.unstructured.data"
##  [38] "Subscription.based.business"
##  [39] "Cloud.or.platform.based.serive.product."
##  [40] "Local.or.global.player"
##  [41] "Linear.or.Non.linear.business.model"
##  [42] "Capital.intensive.business.e.g..e.commerce..Engineering.products
.and.operations.can.also.cause.a.business.to.be.capital.intensive"
##  [43] "Number.of..of.Partners.of.company"
##  [44] "Crowdsourcing.based.business"
##  [45] "Crowdfunding.based.business"
##  [46] "Machine.Learning.based.business"
##  [47] "Predictive.Analytics.business"
##  [48] "Speech.analytics.business"
##  [49] "Prescriptive.analytics.business"
##  [50] "Big.Data.Business"
##  [51] "Cross.Channel.Analytics..marketing.channels"
##  [52] "Owns.data.or.not...monetization.of.data..e.g..Factual"
##  [53] "Is.the.company.an.aggregator.market.place..e.g..Bluekai"
##  [54] "Online.or.offline.venture...physical.location.based.business.or.
online.venture."
##  [55] "B2C.or.B2B.venture."
##  [56] "Top.forums.like..Tech.crunch..or..Venture.beat..talking.about.th
e.company.model...How.much.is.it.being.talked.about."
##  [57] "Average.Years.of.experience.for.founder.and.co.founder"
##  [58] "Exposure.across.the.globe"
```

```
##  [59] "Breadth.of.experience.across.verticals"
##  [60] "Highest.education"
##  [61] "Years.of.education"
##  [62] "Specialization.of.highest.education"
##  [63] "Relevance.of.education.to.venture"
##  [64] "Relevance.of.experience.to.venture"
##  [65] "Degree.from.a.Tier.1.or.Tier.2.university."
##  [66] "Renowned.in.professional.circle"
##  [67] "Experience.in.selling.and.building.products"
##  [68] "Experience.in.Fortune.100.organizations"
##  [69] "Experience.in.Fortune.500.organizations"
##  [70] "Experience.in.Fortune.1000.organizations"
##  [71] "Top.management.similarity"
##  [72] "Number.of.Recognitions.for.Founders.and.Co.founders"
##  [73] "Number.of..of.Research.publications"
##  [74] "Skills.score"
##  [75] "Team.Composition.score"
##  [76] "Dificulty.of.Obtaining.Work.force"
##  [77] "Pricing.Strategy"
##  [78] "Hyper.localisation"
##  [79] "Time.to.market.service.or.product"
##  [80] "Employee.benefits.and.salary.structures"
##  [81] "Long.term.relationship.with.other.founders"
##  [82] "Proprietary.or.patent.position..competitive.position."
##  [83] "Barriers.of.entry.for.the.competitors"
##  [84] "Company.awards"
##  [85] "Controversial.history.of.founder.or.co.founder"
##  [86] "Legal.risk.and.intellectual.property"
##  [87] "Client.Reputation"
##  [88] "google.page.rank.of.company.website"
##  [89] "Technical.proficiencies.to.analyse.and.interpret.unstructured.da
ta"
##  [90] "Solutions.offered"
##  [91] "Invested.through.global.incubation.competitions."
##  [92] "Industry.trend.in.investing"
##  [93] "Disruptiveness.of.technology"
##  [94] "Number.of.Direct.competitors"
##  [95] "Employees.per.year.of.company.existence"
##  [96] "Last.round.of.funding.received..in.milionUSD."
##  [97] "Survival.through.recession..based.on.existence.of.the.company.th
rough.recession.times"
##  [98] "Time.to.1st.investment..in.months."
##  [99] "Avg.time.to.investment...average.across.all.rounds..measured.fro
m.previous.investment"
## [100] "Gartner.hype.cycle.stage"
## [101] "Time.to.maturity.of.technology..in.years."
## [102] "Percent_skill_Entrepreneurship"
## [103] "Percent_skill_Operations"
## [104] "Percent_skill_Engineering"
## [105] "Percent_skill_Marketing"
## [106] "Percent_skill_Leadership"
## [107] "Percent_skill_Data.Science"
## [108] "Percent_skill_Business.Strategy"
## [109] "Percent_skill_Product.Management"
## [110] "Percent_skill_Sales"
## [111] "Percent_skill_Domain"
```

```
## [112] "Percent_skill_Law"
## [113] "Percent_skill_Consulting"
## [114] "Percent_skill_Finance"
## [115] "Percent_skill_Investment"
## [116] "Renown.score"
```

3.5 Очищення інформації

Перша колонка таблиці містить значення Company1, Company2… Слово Company нам не потрібно для ідентифікації рядка, тому його варто забрати, а колонку перетворити у числовий тип. У деяких клітинках з таблиці зберігаються значення “No Info”, “unknown amount”, або просто воно відсутнє. В таких місця слід використовувати NA. Один стовпець містить значення “not applicable” та “no”, які несуть однакову інформацію. Тому їх було приведено до одного загального “no”. Регістр текстових значень у таблиці значення не має, тому переведемо усе в нижній регістр. Для зручності у доступі до потрібної інформації створено змінні, які містять значення назв різних стовпців.

```
company <- col.rename(company, col.id.index, "id")
company$id <- col.as.numeric(col.str.replace(company$id, "Company", ""))

company <- data.frame.remove.unknown(company, "No Info")
company <- data.frame.remove.unknown(company, "")
company <- data.frame.remove.unknown(company, "unknown amount")

company[,col.not.applicable.no.confusion.index] <- col.str.replace(company
[,col.not.applicable.no.confusion.index], "not applicable", "no")

company <- col.rename.all.tolower(company)

col.names <- names(company)
col.date.names <- col.names[col.date.indexes]
col.number.names <- col.names[col.number.indexes]
col.multi.factor.names <- col.names[col.multi.factor.indexes]
col.multi.factor.ordered.names <- col.names[col.multi.factor.ordered.index
es]
```

Деякі колонки містять лише дві категорії значень або лише одна клітинка у них має значення NA. У таких випадках стовпець було

перетворено на фактор. Також дати були приведені до одного загального формату, і, де можливо, колонки було конвертовано у числові типи.

```
company <- col.as.two.stage.factor(company, col.names[col.company.status.index])

date.format <- "%m/%d/%Y"
company$est..founding.date <- col.as.date(company$est..founding.date, date.format)
company$last.funding.date <- col.as.date(company$last.funding.date, date.format)
company$last.funding.date <- col.extract.year.from.date(company$last.funding.date)

company <- cols.as.numeric.round(company, col.number.indexes, 4)
company <- cols.as.two.stage.factors(company, col.names[col.two.factor.indexes])
```

Деякі стовпці можна було привести до багаторівневого фактору. Він може мати три типу рівнів, які наведено нижче:

1) none, few, mane

2) low, medium, high

3) none, low, medium, high

Для того, щоб некорисна інформація не впливала на передбачення, її потрібно усунути з таблиці. Тому було видалено колонки, у яких кількість відсутніх значень менша від 40%. Також дані було розділено за числовим і текстовим типами.

```
company <- cols.as.multi.factors(company, col.multi.factor.names, function(col.name) col.name %in% col.multi.factor.ordered.names)

company.percentage.missing <- data.frame.with.percent.missing(company)
company.cleaned.variables <- variables.with.percent.missing.less.than(company.percentage.missing, 40)
company.cleaned <- company[,company.cleaned.variables]
company.unused.variables <- variables.with.percent.missing.more.equal.than(company.percentage.missing, 40)
company.unused <- company[,company.unused.variables]

col.numeric.names <- col.except.names(c(col.number.names, col.date.names), company.unused.variables)
company.numeric <- company.cleaned[col.numeric.names]
```

```
col.character.names <- col.except.names(colnames(company.cleaned), col.num
eric.names)
company.character <- company.cleaned[col.character.names]
```

Було виконано один з важливих кроків під час очищення неструктурованих даних - видалення сторонніх значень. За допомогою kNN було заповнено пропущені значення в таблиці.

```
company.numeric <- data.frame.remove.outliers(company.numeric)
company.numeric <- kNN(company.numeric, imp_var = FALSE)
```

3.6 Створення характеристик

З наявних даних можна створити дві характеристики такі як відношення останньої дати фінансування до віку компанії та кількість інвесторів. До таблиці з числовими значеннями потрібно додати колонку значення успіху стартапу і привести її у числовий тип.

```
company.numeric <- create.last.funding.and.age.ratio.feature(company.numer
ic)
company.character <- create.investors.count.feature(company.character)

company.numeric$dependent.company.status <- col.factor.as.numeric.bool(com
pany.character$dependent.company.status)
```

3.7 Визначення найважливіших чинників успіху компанії

Потрібно дізнатися, які характеристики компанії найбільше впливають на її успіх, тому було проведено PCA для частини даних, яка відповідає за здібності засновників підприємств, а також проаналізовано інші аспекти стартапів, які можна побачити на графіках нижче.

```
col.founders.skills.pca <- prcomp(company.numeric[,col.founders.skills.ind
exes], center = TRUE, scale. = TRUE)
summary(col.founders.skills.pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
PC7
## Standard deviation     1.6670 1.1677 1.0780 0.97166 0.89468 0.85052 0.8
2069
## Proportion of Variance 0.2779 0.1363 0.1162 0.09441 0.08004 0.07234 0.0
6735
## Cumulative Proportion  0.2779 0.4142 0.5304 0.62486 0.70491 0.77724 0.8
4460
##                            PC8     PC9    PC10
## Standard deviation     0.74858 0.73012 0.67866
## Proportion of Variance 0.05604 0.05331 0.04606
## Cumulative Proportion  0.90063 0.95394 1.00000

ggbiplot(col.founders.skills
```

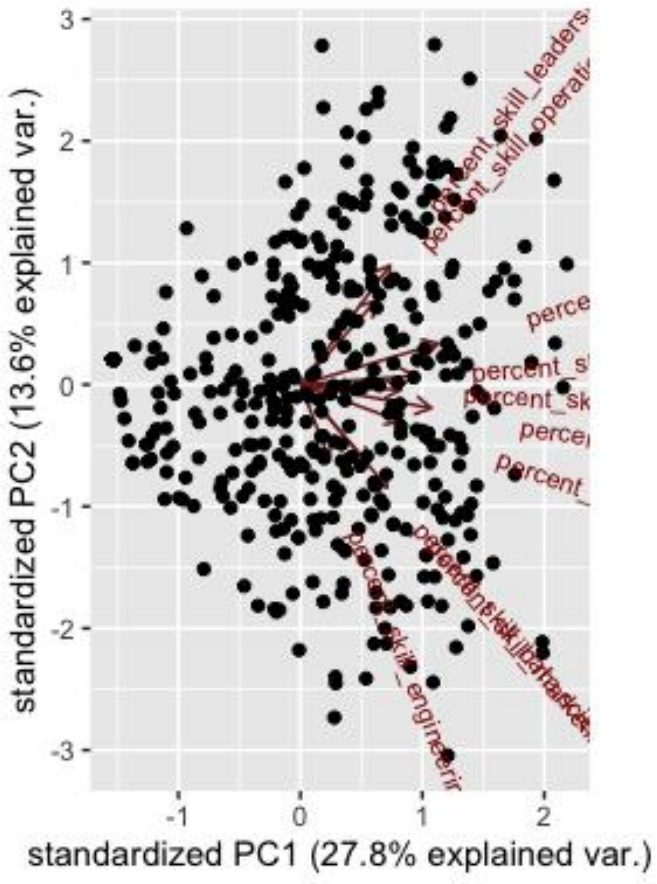

Рисунок 11

Коробковий графік для перегляду тенденцій кількості робітників у компанії.

```
boxplot(company.numeric$employee.count, ylab="Кількість співробітників")
```

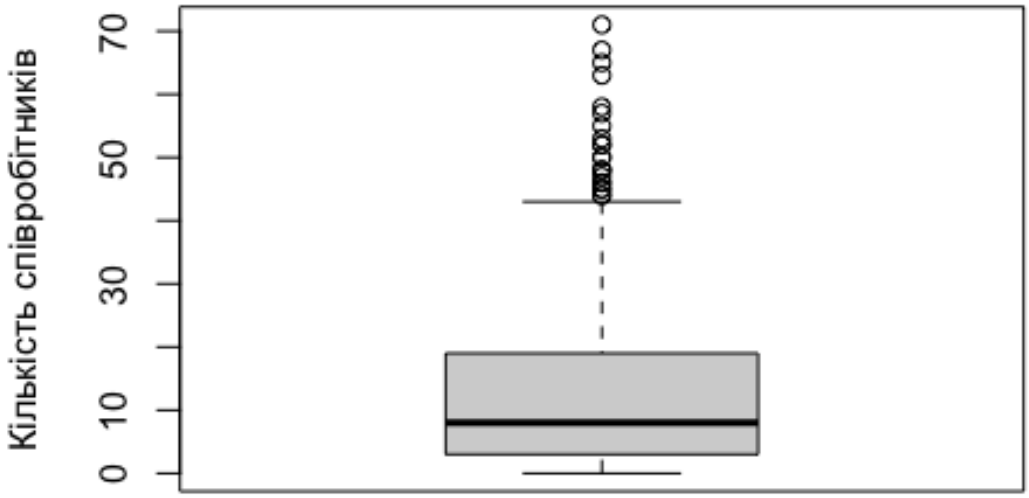

Рисунок 12

Коробковий графік для перегляду тенденцій кількості робітників у компаніях, які зазнали успіху або невдачі.

```
company.numeric$dependent.company.status <-company.character$dependent.company.status
col.numeric.names <- names(company.numeric)

# box plot to see difference in mean of team size w.r.t two categories of dependent
ggplot(company.numeric, aes(x=dependent.company.status, y=team.size.all.employees, fill=dependent.company.status)) +
  geom_boxplot(alpha=0.7) +
  scale_fill_manual(values=c("#FD3307", "#05D832"))
```

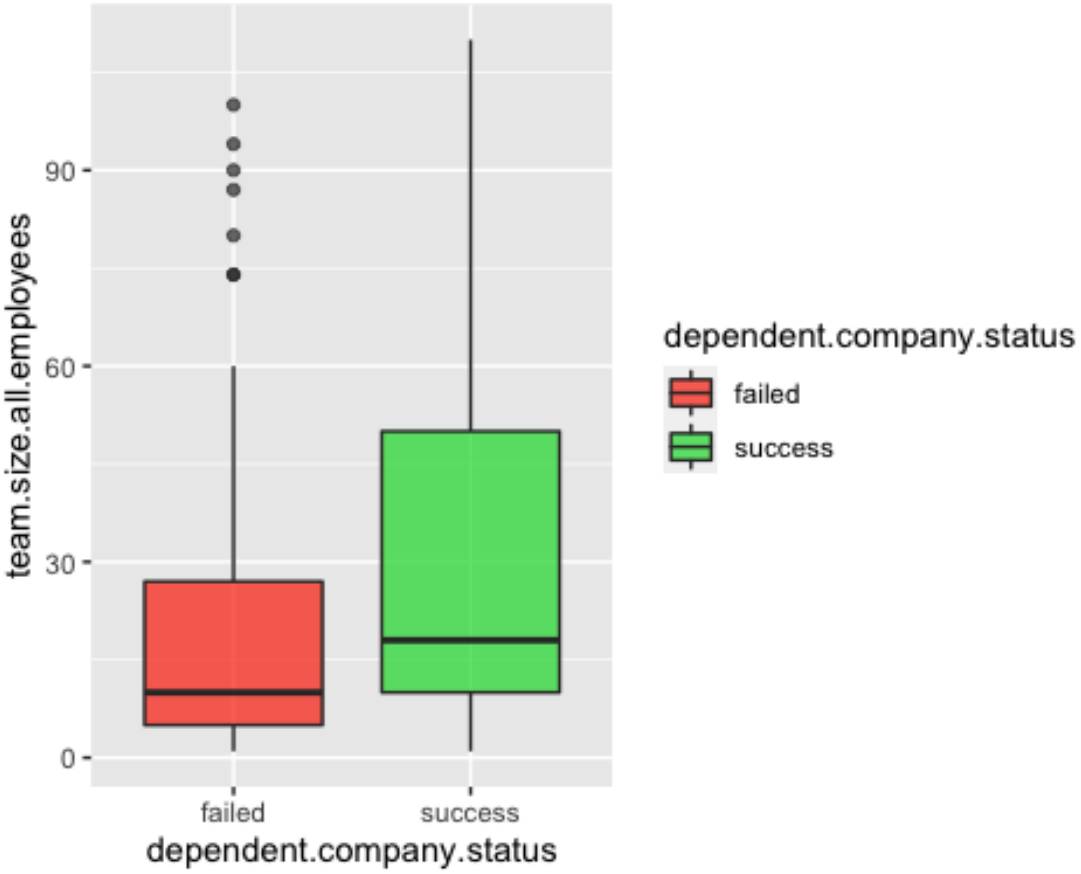

Рисунок 13

Графік для перегляду кількості первинних інвесторів компанії і її результату.

```
ggplot(company.numeric, aes(x = number.of.investors.in.seed, fill=company.
numeric$dependent.company.status)) +
  geom_density(position="identity", alpha=0.5) +
  scale_x_continuous(name = "К-сть первинних інвесторів") +
  scale_y_continuous(name = "К-сть стартапів") +
  scale_fill_manual(values=c("#FF0700", "#05D832")) +
  theme_classic()
```

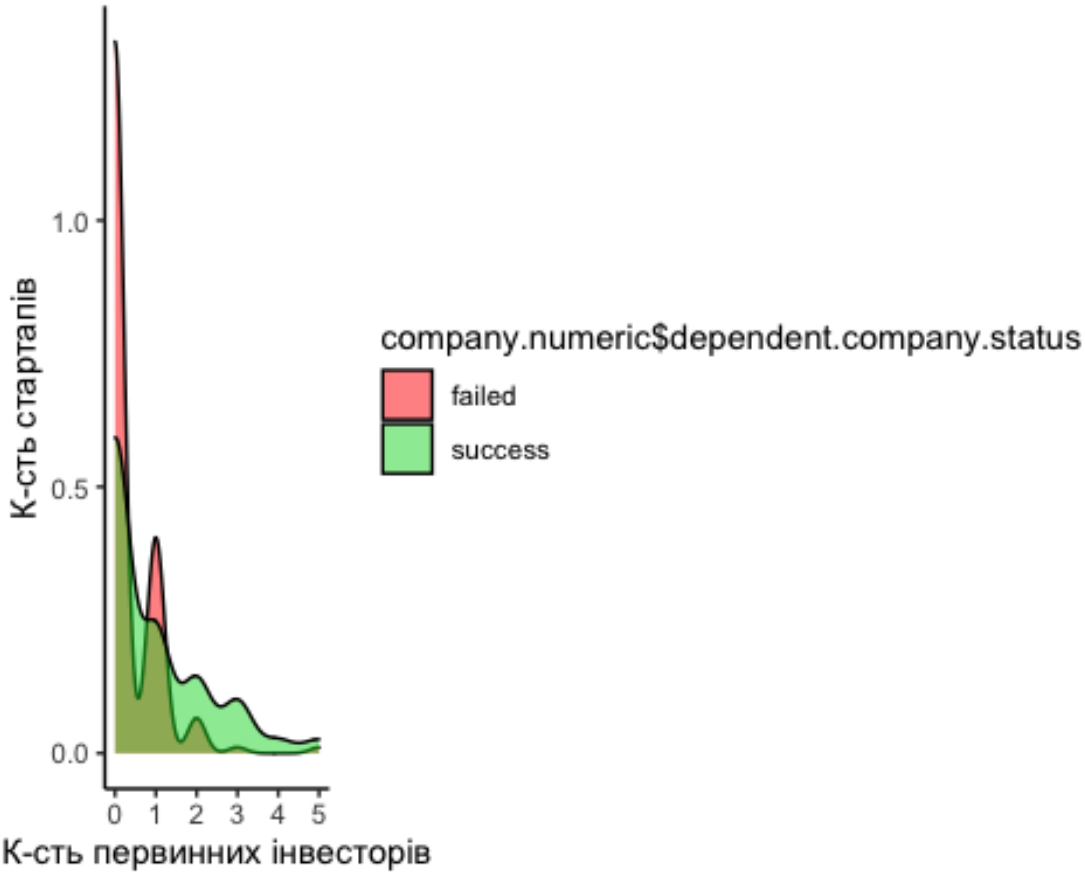

Рисунок 14

Графік для перегляду впливу наявності навичок програмування у засновників компанії на її результат.

```
ggplot(company.numeric, aes(x=percent_skill_engineering, fill=company.nume
ric$dependent.company.status)) +
  geom_histogram(binwidth = 1.5, position="identity", alpha=0.5) +
  scale_x_continuous(name = "% навичок програмування") +
  scale_y_continuous(name = "К-сть стартапів") +
  scale_fill_manual(values=c("#FF0700", "#05D832"))
```

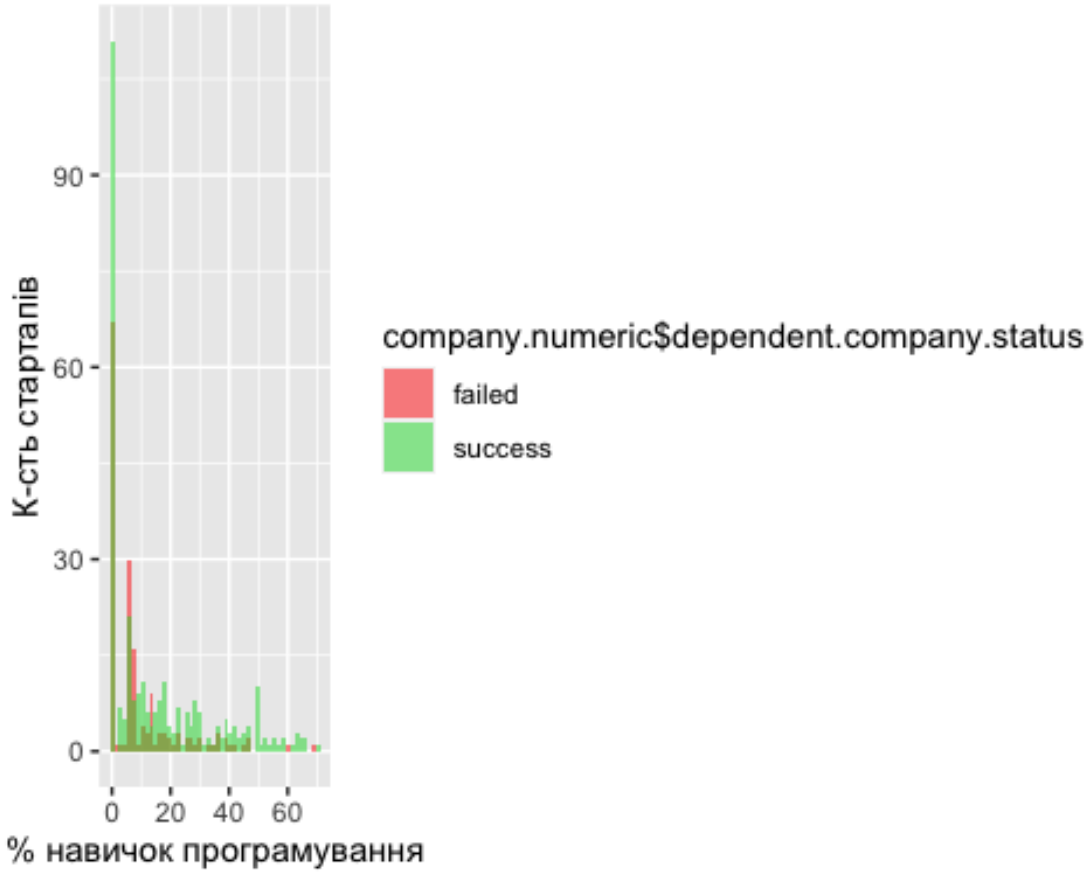

Рисунок 15

Перейдемо до визначення ключових характеристик за допомогою моделей RandomForest та Generalized linear model. Спочатку потрібно до таблиці з числовими типами додати колонку dependent.company.status та привести її в числовий тип. Створимо GLM і отримаємо важливість змінних за допомогою функції varImp. Створимо модель використовуючи RandomForest і отримаємо важливість змінних за допомогою функції

importance. Відфільтруємо характеристики за обрахованою важливість, для GLM використаємо нижній поріг 2, для RandomForest - 3. Об'єднаємо назви важливих колонок, отриманих з обох моделей, і отримаємо ці колонки за їх назвами.

```
company.numeric$dependent.company.status <- col.factor.as.numeric.bool(com
pany.character$dependent.company.status)

model.glm <- glm(formula = dependent.company.status~., data = company.nume
ric)
importance.glm <- varImp(model.glm)

col.founding.funding.dates.indexes <- c(42,43)
model.random.forest.data <- company.numeric[-col.founding.funding.dates.in
dexes]
model.random.forest <- randomForest(dependent.company.status~., data=model
.random.forest.data)

importance.random.forest <- importance(model.random.forest)

features.glm <- rownames(importance.glm)[apply(importance.glm, 1, function
(x) x > 2)]
features.random.forest <- rownames(importance.random.forest)[apply(importa
nce.random.forest, 1, function(x) x > 3)]

features <- c(features.glm, features.random.forest)
train.variable.selection <- company.numeric[, c(features, "dependent.compa
ny.status")]
```

Використаємо також підхід information value для того, щоб визначити найважливіші характеристики компанії, які впливають на її успіх. Відфільтруємо їх за мінімальним значенням 0.1, та максимальним - 0.5. Додаймо колонку з значеннями успіхів стартапів для того, щоб пізніше побудувати модель.

```
company.numeric$fail <- col.as.numeric(!company.numeric$dependent.company.
status)
col.founding.funding.dates.comp.status.indexes <- c(col.founding.funding.d
ates.indexes, 45)
information.value.data <- company.numeric[-col.founding.funding.dates.comp
.status.indexes]
inforation.value <- iv.mult(information.value.data, y="fail", summary=TRUE
)
```

```
inforation.value.without.lower <- inforation.value[which(inforation.value$
InformationValue>0.1),]
inforation.value.cleared <- inforation.value.without.lower[which(inforatio
n.value.without.lower$InformationValue<0.5),]
features.information.value <- inforation.value.cleared$Variable
train.information.value <- company.numeric[, c(features.information.value,
"dependent.company.status")]
```

Створимо три моделі для передбачення успіху стартапу. Першу модель побудуємо за допомогою відібраних при створенні GLM і моделі RandomForest змінних, другу - за допомогою змінних, відфільтрованих використовуючи InformationValue, третю - на змінних, які були визначені нами під час спостереження як основні.

```
model.variables.selection <- step(glm(dependent.company.status~., family =
binomial(link=logit), data = train.variable.selection))

model.variables.importance <- step(glm(dependent.company.status~., family
= binomial(link=logit), data = train.information.value))

model<-glm(formula = as.formula(paste("dependent.company.status~",
                                      paste(colnames(company.numeric)[c(6,
10,18,32,33,35,43,3)], collapse = "+"),
                    sep = "")),
              family = binomial(link = logit),
              data = company.numeric)
```

Обрахуємо коректність моделей за допомогою матриці невідповідності і відобразимо результат на графіках. Виконаємо це для всіх обчислених моделей, окрім однієї зі змінними, які були підібрані нами самостійно.

```
confusion.variables.selection <- confusionMatrix(as.factor(round(model.var
iables.selection$fitted.values)), as.factor(company.numeric$dependent.comp
any.status))

qplot(as.factor(company.numeric$dependent.company.status),
      as.factor(round(model.variables.importance$fitted.values)),
      colour= company.numeric$dependent.company.status, geom = c("boxplot"
, "jitter"),
      xlab = "Спостереження", ylab = "Передбачення") +
  scale_color_gradientn(colors = c("deepskyblue4", "dimgray"))
```

Рисунок 16

```
confusion.variables.importance <- confusionMatrix(as.factor(round(model.va
riables.selection$fitted.values)),
                                   as.factor(company.numeric$dependent.c
ompany.status))

qplot(as.factor(company.numeric$dependent.company.status),
      as.factor(round(model.variables.importance$fitted.values)),
      colour = company.numeric$dependent.company.status, geom = c("boxplot
", "jitter"),
      xlab = "Спостереження", ylab = "Передбачення") +
  scale_color_gradientn(colors = c("deepskyblue4", "dimgray"))
```

Рисунок 17

ВИСНОВОК

Після завершення виконання даної роботи, було створено модель, яка на основі певних характеристик стартапів надає передбачення про їхню успішність. На мою думку, такий продукт є актуальним у сучасному світі, адже він може вирішити проблеми працівників, засновників та інвесторів багатьох компаній, що дасть змогу колосально зекономи людські ресурси.

Курсова робота також допомогла розвинути знання про технології машинного навчання. Було розглянуто декілька найбільш популярних алгоритмів класифікації та регресії, що дає змогу використовувати кожного з них у місцях, де вони найбільш ефективні.

Вивчення мови програмування R є теж дуже важливо, оскільки використовуючи інструменти, які вона надає, можна створювати різноманітні застосування у галузі DS (Data Science) та ML (Machine Learning).

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. "Основи Data Science та Big Data. Python та наука про дані" Деві Сілен, Арно Мейсман, Мохамед Алі
2. "Введення до машинного навчання за допомогою Python" Андреас Мюлер, Сара Гвідо
3. "Самонавчаючі системи" С.І. Ніколаєнко, А.Л. Тулупьєв
4. "Математичні основи теорії машинного навчання та прогнозування" В.В. Вьюгін
5. "Машинне навчання" Брінк Х., Річардс Дж., Феверолф М.
6. "Python для складних задач. Наука про дані та машинне навчання" Плас Дж.В.
7. R - https://en.wikipedia.org/wiki/R_(programming_language)
8. R - https://www.tutorialspoint.com/r/r_overview.htm
9. R - https://www.w3schools.com/r/r_syntax.asp
10. R - https://www.r-project.org
11. Classification - https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623
12. Classification - https://matthew-brett.github.io/dsfe/chapters/09/classification
13. Classification - https://analyticsindiamag.com/7-types-classification-algorithms/
14. Regression - https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/
15. Regression - https://towardsdatascience.com/introduction-to-regression-analysis-9151d8ac14b3
16. Decision tree - https://medium.datadriveninvestor.com/decision-trees-lesson-101-f00dad6cba21
17. Neural network - https://webkid.io/blog/neural-networks-in-javascript/
18. kNN - https://www.excelr.com/blog/data-science/machine-learning-supervised/understanding-the-concept-of-knn-algorithm-using-r

19. Linear regression - https://www.machinelearningmindset.com/linear-regression-with-tensorflow/
20. Logistic regression - https://www.tibco.com/reference-center/what-is-logistic-regression