МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

ЗВІТ

до індивідуального завдання №3

з дисципліни «Моделі статистичного навчання»

Виконав

студент групи ПМіМ-12:

Бордун Михайло

Перевірив:

Проф. Заболоцький Т. М.

Львів – 2021

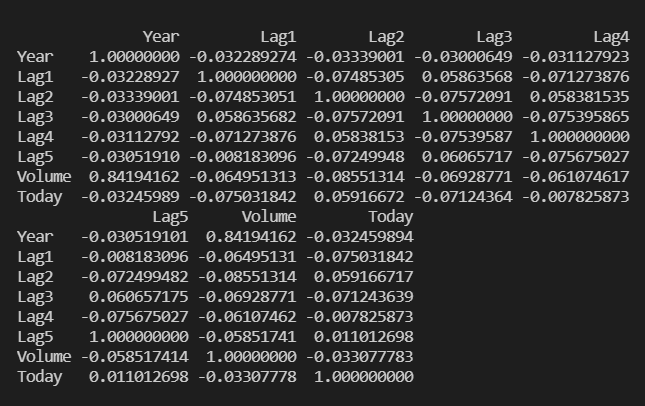
**Хід виконання**

**1.Аналіз даних Weekly, що містять 1 089 щотижневі дохідності за 21 рік, з початку 1990 р. до кінця 2010 року.**

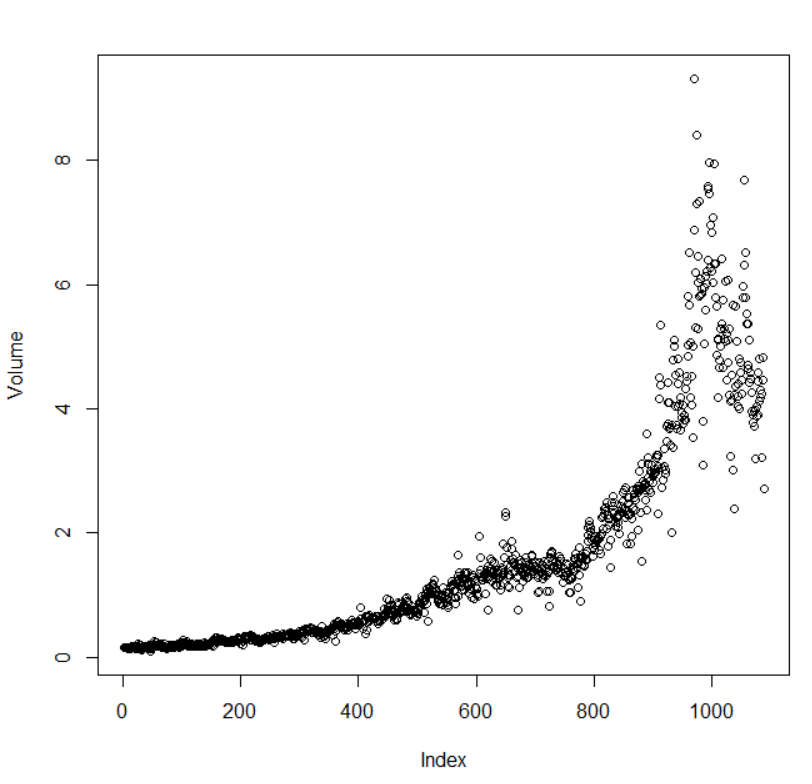
Розглянемо датасет Weekly. Для кожної дати наявна дохідність для попередніх 5 днів Lag1,…,Lag5. Змінна Volume містить дані про обсяг торгів попереднього дня у млн., Today – сьогоднішня дохідність, та змінна Direction, яка вказує чи зріс ринок чи впав.



**1.1** Розглянуто кореляції між змінними нашого датасету за допомогою функції cor().

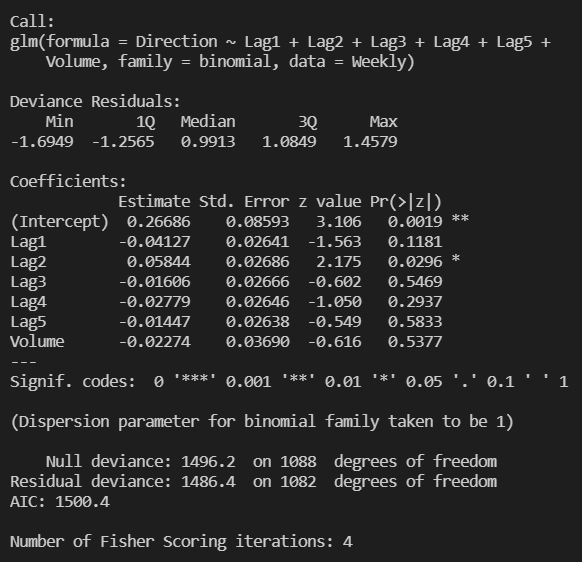
****

Як бачимо чітку кореляцію тільки між змінною Year та Volume. Можна представити це графічно за допомогою функції plot().

****

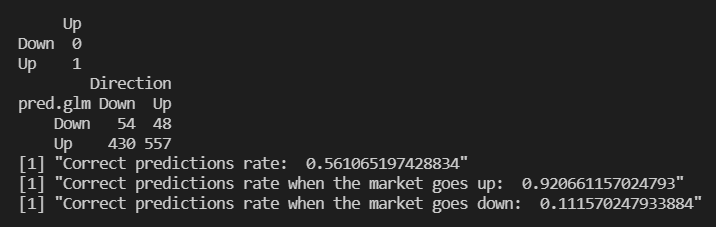
Бачимо, що змінна Volume зростає з плином часу, при чому загалом бачимо експоненціальний зріст.

**1.2** Побудовано логістичну регресію, де Direction – залежна змінна, а п’ять зміщенних дохідностей та змінна Volume незалежні.

****

Для нашої моделі бачимо, що найменше p-value відповідає змінній Lag2, тобто кореляція є суттєва. Тим паче, є наявний позитивний коефіцієнт біля цієї змінної, що вказує на те, що якщо позавчора ринок мав негативну дохідність, то сьогодні більше шансів зрости.

**1.3** Побудовано матрицю помилок та обчислено загальну частку правильних прогнозів.

****

Тобто, загальна частка правильних прогнозів становить 56.1%, що є досить поганим результатом прогнозування. Впродовж тижнів коли ринок йде вгору, модель правильно прогнозує у 92.1% випадків (557/(48+557))%. Впродовж тижнів коли ринок спадає, модель має рацію лише в 11.2% випадків (54/(54+430))%.

**1.4** Побудовано модель логістичної регресії з використанням навчальних даних з 1990 по 2008 рр., з єдиним предиктором Lag2.

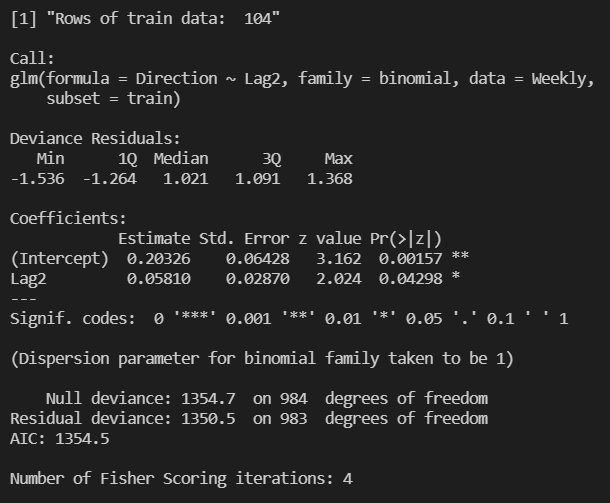
train = (Year >= 1990 & Year <= 2008)

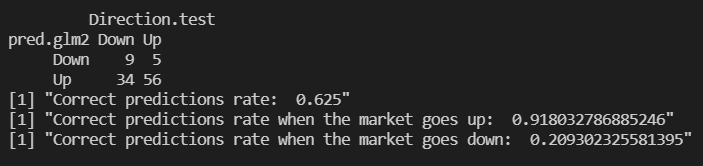
Weekly.test = Weekly[!train, ]

Direction.test = Direction[!train]

cat("\n")

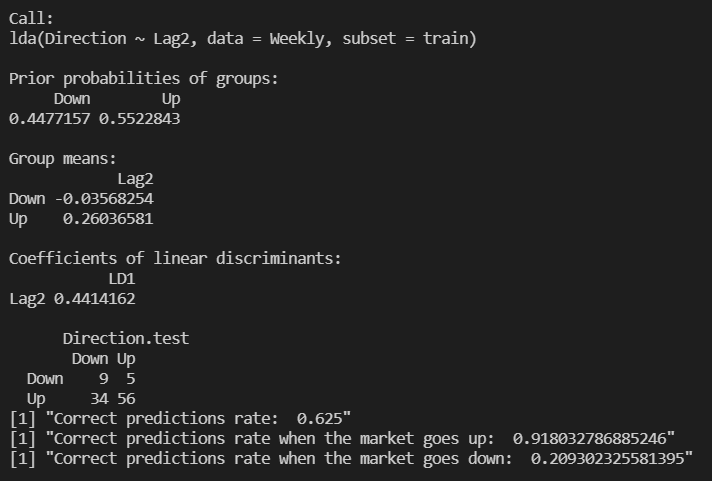
print(paste("Rows of train data: ", dim(Weekly.test)[1]))

****

****

З огляду на матрицю помилок, бачимо, що модель схильна позначати більшу частину тестових даних як підйом ринку. Загалом частка правильних прогнозів становить 62.5%. Впродовж тижнів коли ринок йде вгору, модель правильно прогнозує у 91.8% випадків (56/(56+5))%. Впродовж тижнів коли ринок спадає, модель має рацію лише в 20.9% випадків (9/(34+9))%.

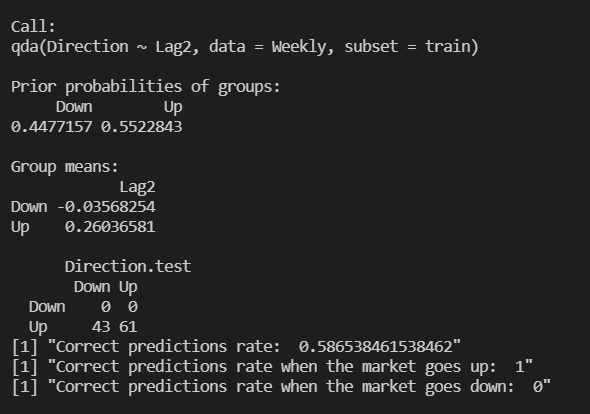
**1.5** Побудовано модель лінійного дискримінантного аналізу з використанням навчальних даних з 1990 по 2008 рр., з єдиним предиктором Lag2.

****

Як бачимо навчальні дані є досить добре розподілені і 55% вибірки позначено як зростання ринку. Ми також отримали групові середні, з яких бачимо, що позавчора коли ринок спадає, то дохідність є негативна і навпаки.

З огляду на матрицю помилок, бачимо, що модель схильна позначати більшу частину тестових даних як підйом ринку. Загалом частка правильних прогнозів становить 62.5%. Впродовж тижнів коли ринок йде вгору, модель правильно прогнозує у 91.8% випадків (56/(56+5))%. Впродовж тижнів коли ринок спадає, модель має рацію лише в 20.9% випадків (9/(34+9))%.

**1.6** Побудовано модель квадратичного дискримінантного аналізу з використанням навчальних даних з 1990 по 2008 рр., з єдиним предиктором Lag2.

****

Бачимо цікаву картину аналізуючи матрицю помилок. Тобто, модель позначає весь час тестові дані як підйом ринку. Загалом частка правильних прогнозів становить 58.7%. Впродовж тижнів коли ринок йде вгору, модель правильно прогнозує у 100% випадків і навпаки, коли ринок спадає, модель робить повністю хибні прогнози.

**1.7** Побудовано класифікатор K-найближчих сусідів з K=1 з використанням навчальних даних з 1990 по 2008 рр., з єдиним предиктором Lag2.

Нижче наведений код для формування перших двох аргументів для функції прогнозування. Тобто формуємо матриці з предиктора Lag2, які пов’язані чисто з навчальними даними та тестовими. Set.seed(1) встановлено для випадку, якщо кілька спостережень розглядатимуться як найближчі сусіди, то R буде випадковим чином вибирати потрібну кількість.

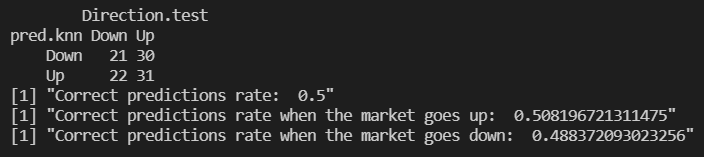
train.X = as.matrix(Lag2[train])

test.X = as.matrix(Lag2[!train])

Direction.train = Direction[train]

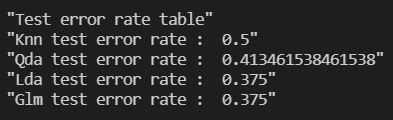
set.seed(1)

pred.knn = knn(train.X, test.X, Direction.train, k = 1)

****

З огляду на матрицю помилок бачимо, що частка правильних прогнозів становить 50%. Впродовж тижнів коли ринок йде вгору, модель правильно прогнозує у 50.8% випадків (31/(30+31))%. Впродовж тижнів коли ринок спадає, модель має рацію в 48.8% випадків (21/(22+21))%.

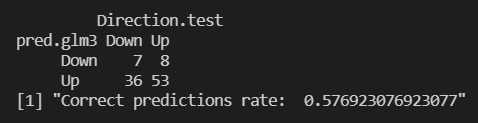
**1.8**

****

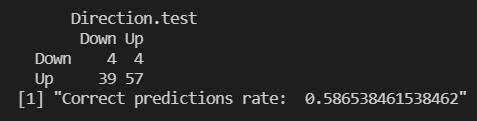
Вище наведено таблицю тестових помилок вищезгаданих моделей. Звідси очевидно, що найбільш відповідними для нашої вибірки виявились LDA та GLM моделі.

**1.9** Поекспериментував з різними комбінаціями предикторів, використовуючи у тому числі можливі перетворення та взаємодії.

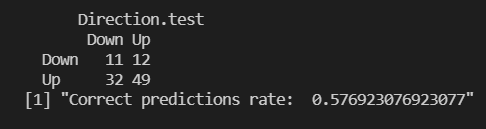
fit.glm3 =  glm(Direction ~ Lag2\*Lag1, data = Weekly, family = binomial, subset = train)

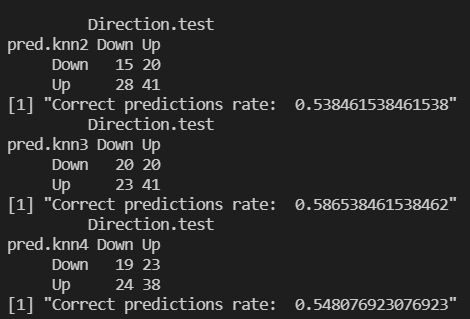
****

fit.lda2 = lda(Direction ~ Lag2:Lag1 + Lag1, data = Weekly, subset = train)

****

fit.qda2 = qda(Direction ~ Lag2 + abs(Lag2), data = Weekly, subset = train)

****

****

В наступних результатах подані матриці помилок для моделей К-найближчих сусідів відповідно із значеннями K= 5, 15, 30. Тобто для значення K=15 бачимо найкращу частку правильних прогнозів на тестових даних, а саме 58.7%, це найкращий результат прогнозування серед всіх попередньо розглянутих класифікаторів K-найближчих сусідів. Такого ж результату було досягнуто з використання LDA, де було використано змінну взаємодії та змінну Lag1. Але, бачимо, що використання змінних взаємодії не приносить покращень щодо прогнозування і базові моделі логістичної регресії та LDA мають найкращу точність (62.5%).

**2. Модель класифікації для передбачення, чи вибране авто має велике або низьке споживання газу на базі даних Auto.**

**2.1** Було створено двійкову змінну mpg01, яка містить 1, якщо mpg містить значення вище медіани, і 0, якщо mpg містить значення нижче його медіани. А також створено єдиний набір даних , що містить як mpg01, так і інші змінні з датасету Auto.

attach(autos)

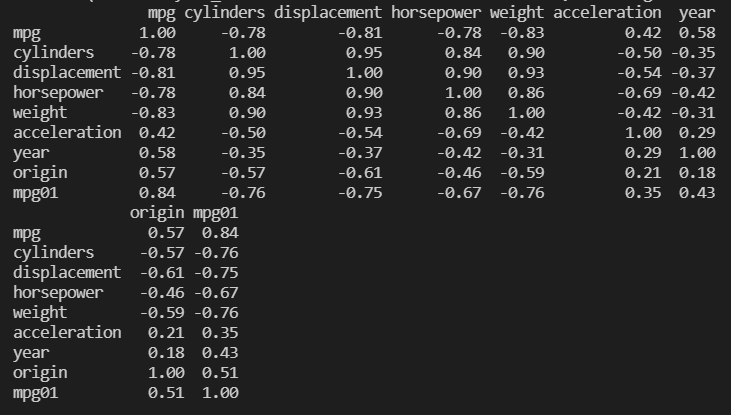
mpg01 = rep(0, length(mpg))

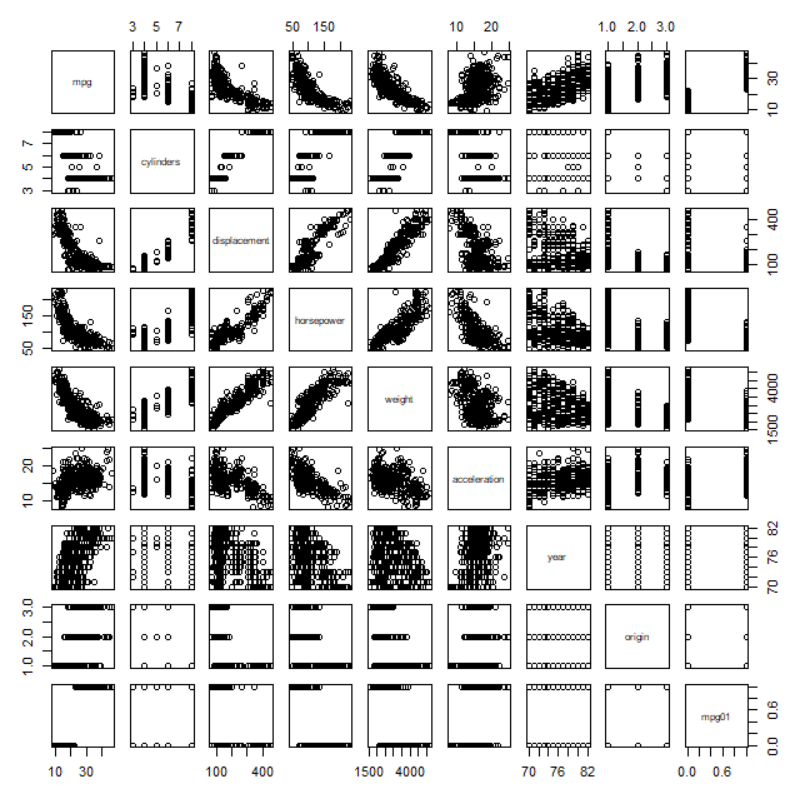
mpg01[mpg > median(mpg)] = 1

autos = data.frame(autos, mpg01)

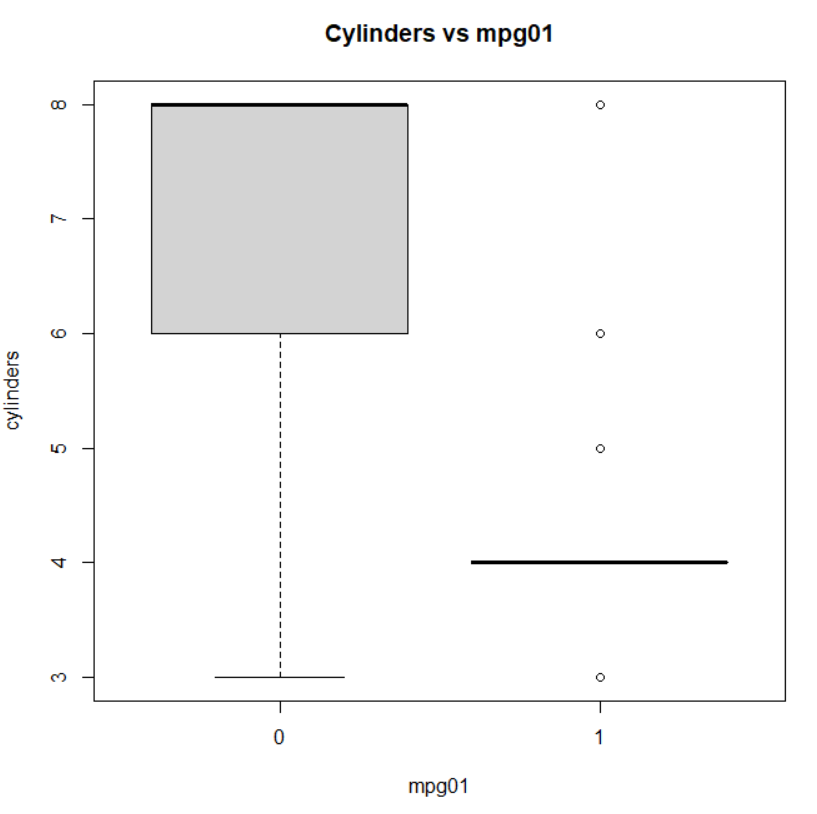
**2.2** Дослідимо змінні та їх залежності з датасету Auto. Для початку використано функцію cor(), щоб побачити настільки сильна чи слаба кореляція між змінними.

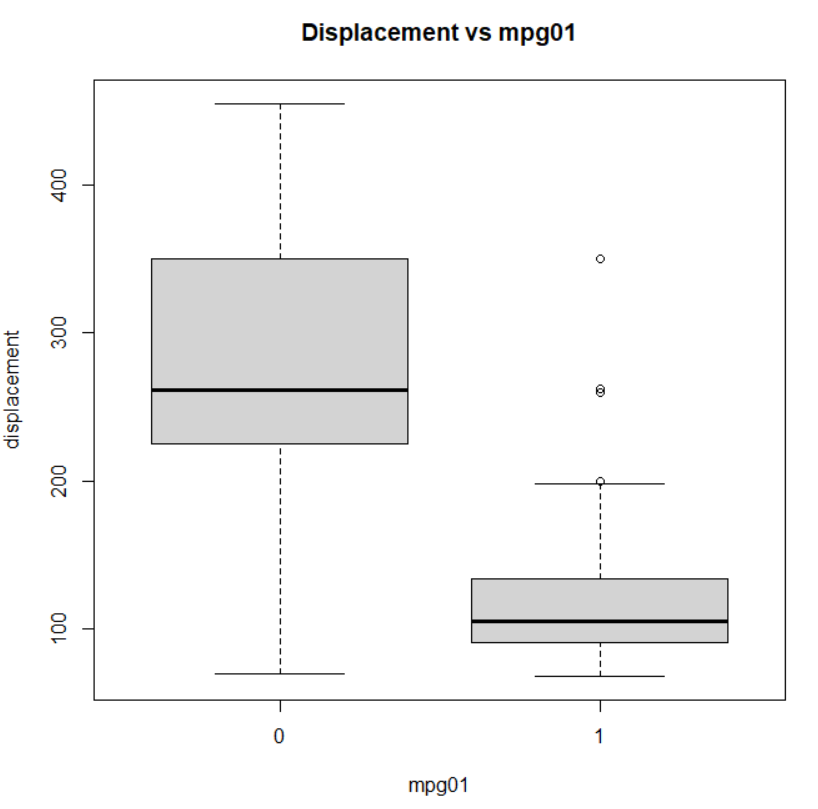
А також за допомогою функції pairs() виведено графічну залежність всіх змінних.

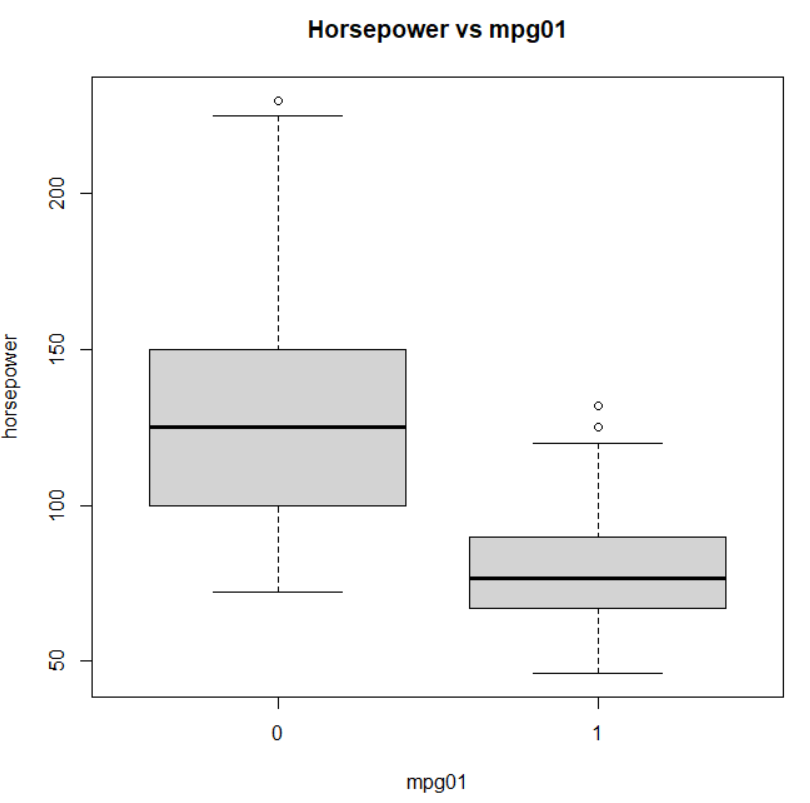


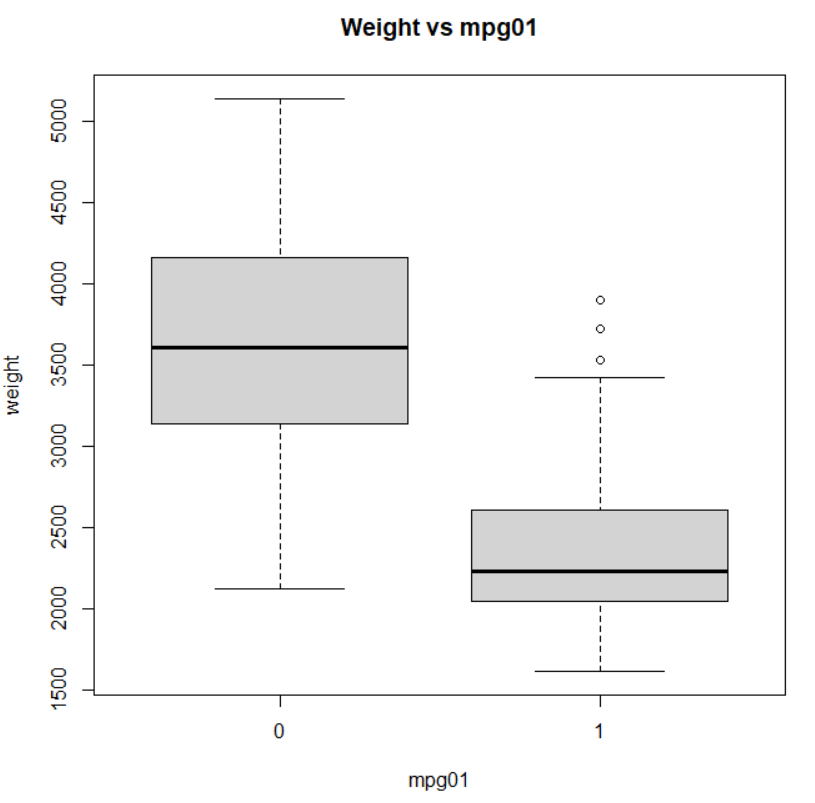


З числових даних бачимо, що є досить сильна залежність між mpg01 та змінними cylinders, displacement, horsepower та weight, але тут варто наголосити що це є від’ємна кореляція. Тобто далі будемо досліджувати детальніше вищезгадані змінні. Це було виконано з використанням функції boxplot().









Всі наведені вище коробчасті діаграми вказують, що справді є наявна від’ємна кореляція між змінними cylinders, displacement, horsepower, weight та нашою якісною змінною mpg01. Хоча варто сказати, що для всіх цих змінних існують значення при mpg01 = 1, які є більші за середнє при mpg01 = 0 (тобто коли mpg містить значення нижче його медіани), проте таких небагато.

**2.3** Розбито дані на начальний та тестовий набори. Це було виконано взявши посортовану вибірку year та розділивши на дві половини, де перша – це тренувальний набір, а друга відповідно тестовий.

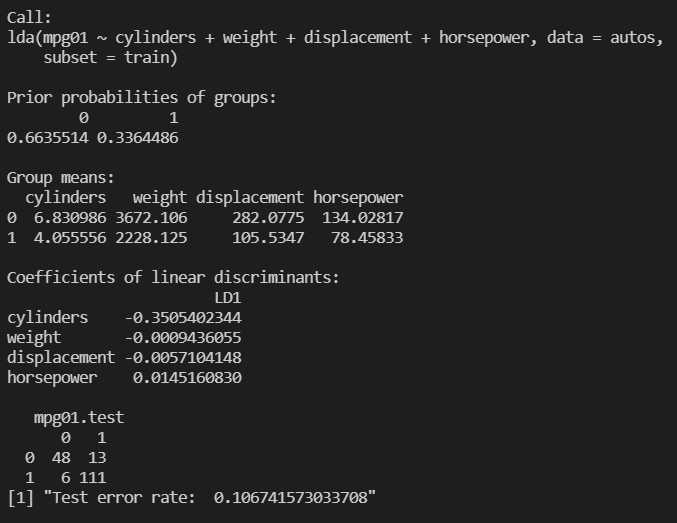
train = (year >= min(year) & year <= min(year) + (max(year) - min(year)) %/% 2)

autos.train = autos[train, ]

autos.test = autos[!train, ]

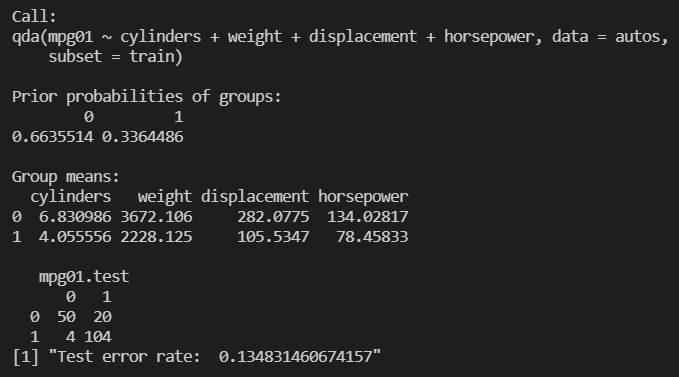
mpg01.test = mpg01[!train]

**2.4** Застосовано лінійний дискримінантний аналіз на навчальних даних. В ролі предикторів було взято найбільш залежні від mpg01 змінні, тобто cylinders, displacement, horsepower та weight.

****

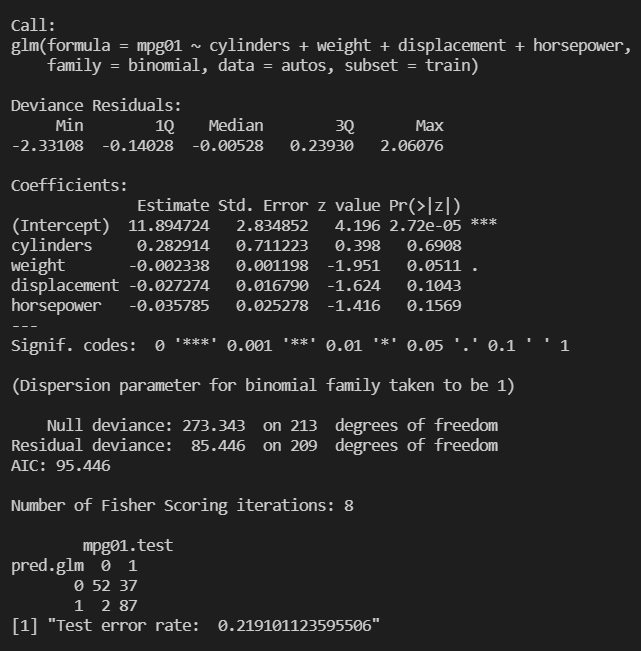
Як бачимо, тільки третина навчальних даних відповідає тому, що mpg містить значення вище своєї медіани, тобто вибірка не є ідеально розподілена. Також цікаво наголосити, що коефіцієнт лінійного дискримінанту для змінної cylinders становить -0.35, що значною мірою буде впливати на правило для прийняття рішень щодо прогнозування. Тестова помилка отриманої моделі є 10%, при чому вона не сильно відрізняється в залежності чи значення mpg є нижче за медіану чи навпаки.

**2.5** Застосовано квадратичний дискримінантний аналіз на навчальних даних. В ролі предикторів було взято найбільш залежні від mpg01 змінні.

****

Тестова помилка цієї моделі становить 13.5%, що є гіршим показником в порівнянні з LQA. Але тут як бачимо з матриці помилок прогнозування відбується дуже точно для mpg01=0, а саме тестова помилка становить (4/54)% = 7.4%, що звісно також означає про гірші показники прогнозування для значень змінної mpg01, які дорівнюють одиниці.

**2.6** Застосовано логістичну регресію на навчальних даних. В ролі предикторів було взято найбільш залежні від mpg01 змінні.

****

З огляду на наведені вище дані, бачимо що найменше p-value відповідає змінній weight, тобто кореляція є суттєва. Змінна cylinders виділяється за рахунок того, що є наявний позитивний коефіцієнт біля цієї змінної, однак через велике p-value неможливо говорити про зв’язок між залежною змінною в даній моделі. Всі інші коефіцієнти вказують на від’ємну кореляцію із залежною змінною. Тестова помилка становить 21.9%, що є найгіршим результатом прогнозування серед вищезгаданих моделей, при чому точність прогнозування для mpg01=0 сильно зростає, а саме (2/54)% = 3.7%.

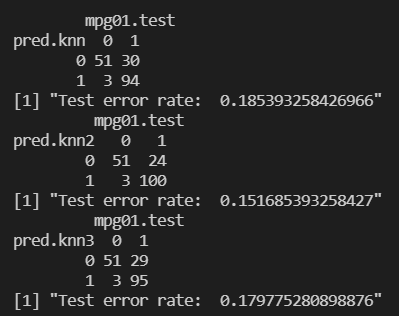
**2.7** Застосовано метод К-найближчих сусідів з різними значеннями для К на навчальних даних. Нижче наведений код для формування перших двох аргументів для функції прогнозування. Тобто за допомогою функції cbind() утворюється матриці з предикторами, які пов’язані чисто з навчальними даними та тестовими.

train.X = cbind(cylinders, weight, displacement, horsepower)[train, ]

test.X = cbind(cylinders, weight, displacement, horsepower)[!train, ]

mpg01.train = mpg01[train]

set.seed(1)

****

В наступних результатах подані матриці помилок для моделей К-найближчих сусідів відповідно із значеннями K= 1, 5, 15. Тобто для значення K=5 бачимо найкращу тестову помилку, а саме 15.2%, що є гіршим результатом ніж у моделях LDA та QDA. З огляду на те, що більшість навчальних даних відповідає тому, що mpg містить значення нижче своєї медіани, то і знову бачимо що дуже висока точність для прогнозування значень, де mpg01=0, а саме тестова помилка становить (3/54)% = 5.6%.

**3. Написання функцій.**

**3.1** Створено функцію Power(), що виводить результат піднесення 2 до 3-ої степені.

Power = function() {2^3}

print(paste("raising 2 to the 3rd power: ", Power()))



**3.2** Запрограмовано нову функцію Power2(), яка дозволяє передавати будь-які два числа, x і a, і виводить значення x^a.

print(paste("x: 5, a: 3 -> power: ", Power2(5, 3)))

****

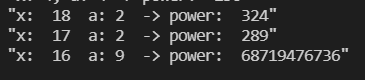
**3.3** Для демонстрації виконання функції Power2(), написано нижче наведений цикл з використання згенерованої вибірки значень для аргументів x та a.

for (x in sample(1:25, 3)) {

  a = sample(1:10, 1)

  print(paste("x: ", x, " a:", a, " -> power: ", Power2(x, a)))

}

****

**3.4**  Написано нову функцію Power3(), яка фактично повертає результат x^a як об'єкт R, а не просто друкує його на екран.

Power3 = function(x, a) {

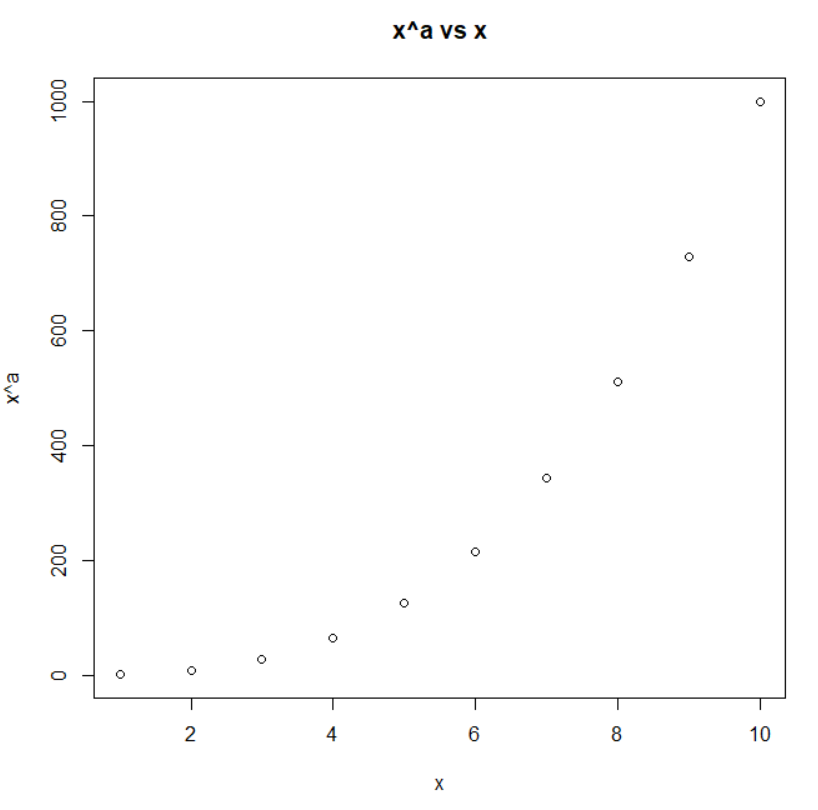
    result = x^a

    return(result)

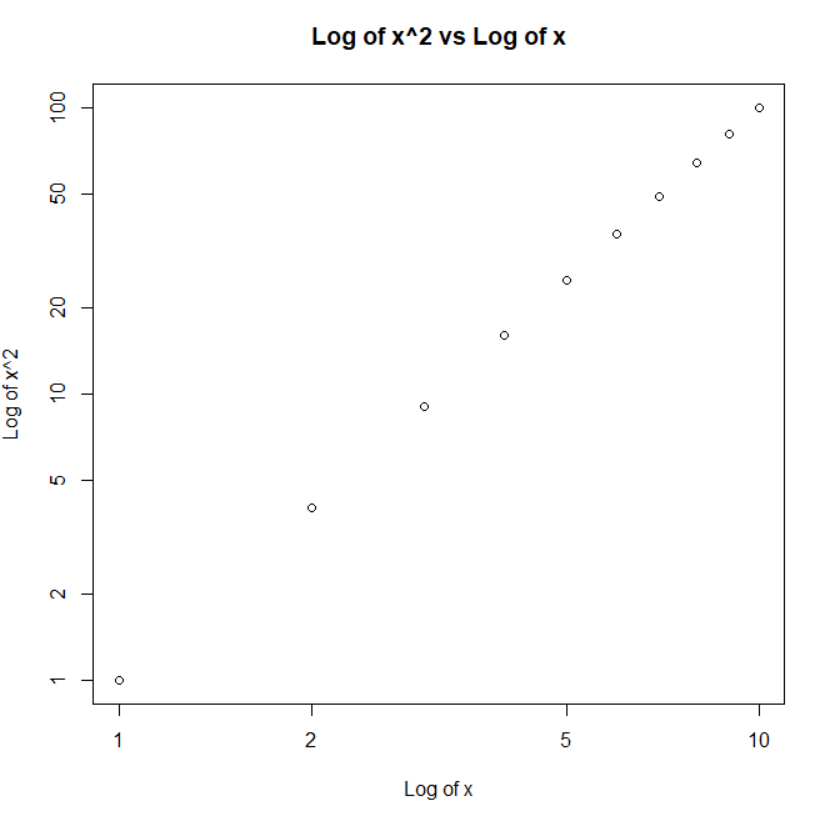
}

power\_res\_3\_3 = Power3(3, 3)

**3.5**  Використовуючи функцію Power3(), побудовано графік f(x) = x2. Як вибірку значень для осі абсцис взято діапазон цілих чисел від 1 до 10. Вісь ординат відповідно відображає x2.

****

Також розглянув можливість відображення обох осей в логарифмічній шкалі використовуючи аргумент log=”xy” у функції plot().

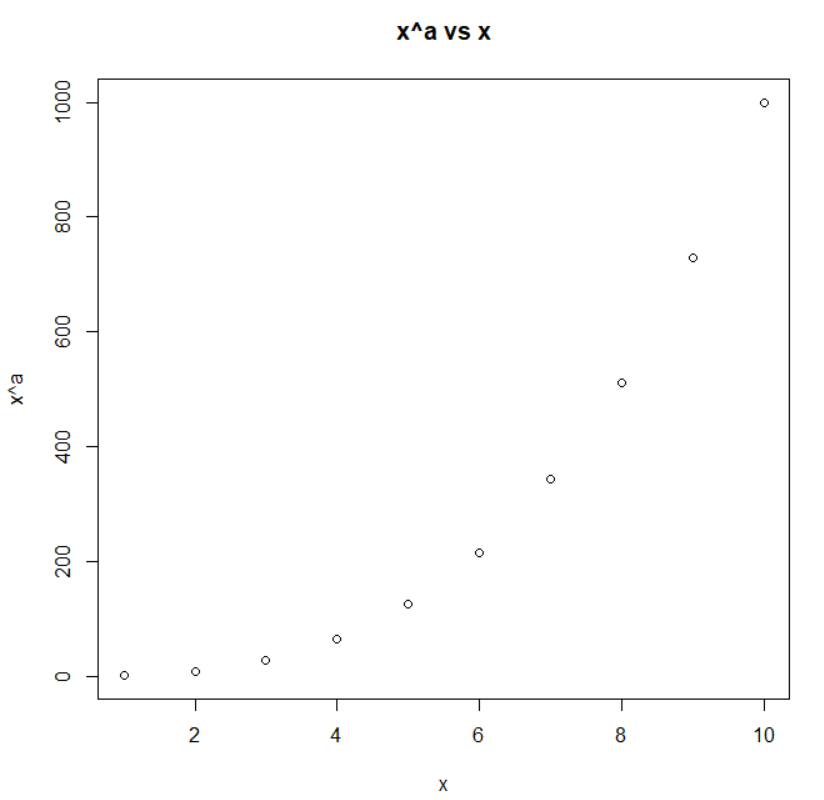
****

**3.6** Запрограмовано функцію PlotPower(), яка дозволяє будувати графік функції x^a для фіксованого a та для діапазону значень x. Виконано це завдяки використанню функції plot() та її аргументу для значення осі ординат як виклик функції Power3(x, a).

PlotPower = function(x, a) {

  plot(x, Power3(x, a), xlab = "x", ylab = "x^a", main = "x^a vs x")

}

****

**4. Модель класифікації для передбачення у вибраному районі рівня злочинності більшого чи меншого за медіану на основі даних Boston.**

Для початку нам потрібно визначити бінарну змінну crim01, яка містить 1, якщо crim містить значення вище медіани, і 0, якщо crim містить значення нижче його медіани. А також створено єдиний набір даних , що містить як crim01, так і інші змінні з датасету Boston.

Далі необхідно розбити дані на начальний та тестовий набори. Це було виконано взявши вибірку crim та розділивши на дві половини, де перша – це тренувальний набір, а друга відповідно тестовий.

Нижче також наведений код для дослідження логістичної регресії. В ролі предикторів було взято всі змінні, окрім crim01, crim, а також zn та rad для уникнення застережень про ідеальне розділення одиниць та нулів в target-змінній.

train = (crim %in% crim[1:(length(crim) %/% 2)])

Boston.train = Boston[train, ]

Boston.test = Boston[!train, ]

crim01.test = crim01[!train]

fit.glm = glm(crim01 ~. - crim01 - crim - zn - rad,

 data = Boston, family = binomial, subset = train)

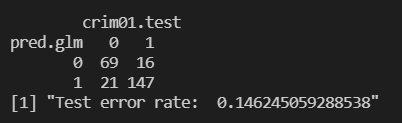
probs = predict(fit.glm, Boston.test, type = "response")

pred.glm = rep(0, length(probs))

pred.glm[probs > 0.5] = 1

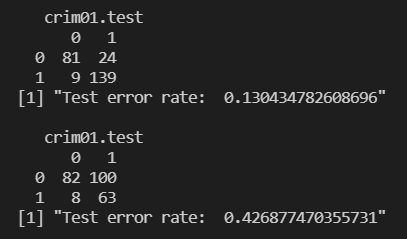
print(table(pred.glm, crim01.test))

print(paste("Test error rate: ", mean(pred.glm != crim01.test)))



Результат показує, що тестова помилка отриманої моделі є 14.6%, при чому з матриці помилок бачимо, що точність прогнозування у вибраному районі рівня злочинності більшого за медіану зростає, а саме помилка є (16/(147+16))% = 9.8%.

Розглянемо також моделі лінійного та квадратичного дискримінантного аналізу.



Верхня матриця помилок відноситься до LDA, нижня відповідно до QDA. Бачимо очевидну різницю, оскільки тестова помилка для LDA становить 13%, що робить цю модель також більш придатною в порівнянні з моделлю логістичної регресії для нашої задачі прогнозування. З іншого боку бачимо жахливу тестову помилку для моделі QDA, при чому знову розглядається сильний контраст у прогнозуванні, оскільки для рівня злочинності більшого за медіану набагато більше неправильних прогнозувань, тоді як для рівня злочинності меншого за медіану точність є більша ніж навіть у моделі LDA.

Далі розглянемо прогнозування з використанням моделі KNN. Нижче наведений код для формування перших двох аргументів для функції прогнозування. Тобто за допомогою функції cbind() утворюється матриці з предикторами, які пов’язані чисто з навчальними даними та тестовими.

train.X = cbind(indus, chas, nox, rm, age, dis,

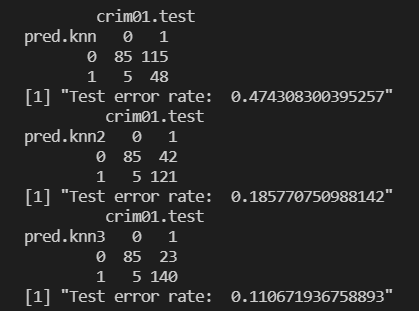
 tax, ptratio, black, lstat, medv)[train, ]

test.X = cbind(indus, chas, nox, rm, age, dis,

 tax, ptratio, black, lstat, medv)[!train, ]

crim01.train = crim01[train]

set.seed(1)



В наступних результатах подані матриці помилок для моделей К-найближчих сусідів відповідно із значеннями K= 1, 5, 15. Тобто для значення K=15 бачимо найкращу тестову помилку, а саме 11%, це найкращий результат прогнозування серед всіх попередньо розглянутих моделей. Найгірший результат прогнозування є для моделі з K=1, але це не означає, що є якась залежність між кількістю K та точністю, оскільки вже для значень K=100 результат був найгірший і складав точність майже 50%.