gender=c("male","female")

> gender

[1] "male" "female"

> x1=c(1,3,5,7,9)

> x1

[1] 1 3 5 7 9

> seq(from=1, to=7 ,by=0.5)

[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0

> seq(from=1, to=7 ,by=1)

[1] 1 2 3 4 5 6 7

> rep(1,times=5)

[1] 1 1 1 1 1

> rep(5,times=6)

[1] 5 5 5 5 5 5

> rep("harun",times=4)

[1] "harun" "harun" "harun" "harun"

> a=c(1:6) c matris demek

> a

[1] 1 2 3 4 5 6

> a[3]

[1] 3

> a[2]=9

> a

[1] 1 9 3 4 5 6

> a[-2]

[1] 1 3 4 5 6 #2. eleman hariç tüm elemanları getirir

> y=c(1,3,5,7,9)

> y

[1] 1 3 5 7 9

> y[c(1,5)] 1. ve 5. elemanları alır

[1] 1 9

> y[y<6]

[1] 1 3 5

> a=matrix(c(1,2,3,4,5,6), nrow = 3, byrow=FALSE)

> a matris oluştururken matrix diye başlanır

[,1] [,2]

[1,] 1 4

[2,] 2 5

[3,] 3 6

[,1] [,2] [,3] [,4]

[1,] 1 2 3 4

[2,] 5 6 7 8

[3,] 9 10 11 12

> b[1,2]

[1] 2

> mat[c(1,3),2] yani [1,2] [3,2] yi çıkartır

[1] 2 8

> mat[2,]

[1] 4 5 6 2. satırın tamamı

> mat[,3]

[1] 3 6 9 3. sütunun tamamı

> mat[,3]+5

[1] 8 11 14 3. sütuna 5 ekledim

> mat[2,2]=10

> mat

[,1] [,2] [,3]

[1,] 1 2 3

[2,] 4 10 6

[3,] 7 8 9

> data1

> DataToExport veriyi import ettim

# A tibble: 8 x 4

Subject Age Gender Score

<chr> <dbl> <chr> <dbl>

1 Dave.Andreychuk 53 male 80.5

2 Jon.Stewart 54 male 82.1

3 Jane.Doe 38 female 75.9

4 Amelia.Earhart 119 female 90

5 Donald.Trump 70 male -25.5

6 Sidney.Crosby 28 male 87.2

7 Oprah.Winfrey 62 female 88.8

8 Steve.Jobs 61 male 91.1

> DataToExport[1,1]

# A tibble: 1 x 1

Subject

<chr>

1 Dave.Andreychuk [1,1] dave dir dedi

> DataToExport[c(1:8),1] 1 den 8 e kadar 1. sütunu al

# A tibble: 8 x 1

Subject

<chr>

1 Dave.Andreychuk

2 Jon.Stewart

3 Jane.Doe

4 Amelia.Earhart

5 Donald.Trump

6 Sidney.Crosby

7 Oprah.Winfrey

8 Steve.Jobs

> DataToExport[c(1:8),c(1,2)] 1 den 8 e kadar 1. ve 2. sütunu al

# A tibble: 8 x 2

Subject Age

<chr> <dbl>

1 Dave.Andreychuk 53

2 Jon.Stewart 54

3 Jane.Doe 38

4 Amelia.Earhart 119

5 Donald.Trump 70

6 Sidney.Crosby 28

7 Oprah.Winfrey 62

8 Steve.Jobs 61

> DataToExport[c(1:8),c(1,3)] 1 den 8 e kadar 1. ve 3. sütunları göster dedim

# A tibble: 8 x 2

Subject Gender

<chr> <chr>

1 Dave.Andreychuk male

2 Jon.Stewart male

3 Jane.Doe female

4 Amelia.Earhart female

5 Donald.Trump male

6 Sidney.Crosby male

7 Oprah.Winfrey female

8 Steve.Jobs male

rm(data1) data1 tablosunu verilerden kaldırır.

kişi\_maaş[1] kişi maaşın 1.sütununu getirir.

> dim(kişi\_maaş)

[1] 4 2 dim bize tablonun boyutunu verir kaç satır kaç sütun

> dim(DataToExport)

[1] 8 4 8 satır 4 sütun

> head(DataToExport) head komutu verinin ilk 6 satırını görmemizi sağlar.

# A tibble: 6 x 4

Subject Age Gender Score

<chr> <dbl> <chr> <dbl>

1 Dave.Andreychuk 53 male 80.5

2 Jon.Stewart 54 male 82.1

3 Jane.Doe 38 female 75.9

4 Amelia.Earhart 119 female 90

5 Donald.Trump 70 male -25.5

6 Sidney.Crosby 28 male 87.2

> tail(DataToExport) tail komutu son 6 satırı görmemizi sağlar.

# A tibble: 6 x 4

Subject Age Gender Score

<chr> <dbl> <chr> <dbl>

1 Jane.Doe 38 female 75.9

2 Amelia.Earhart 119 female 90

3 Donald.Trump 70 male -25.5

4 Sidney.Crosby 28 male 87.2

5 Oprah.Winfrey 62 female 88.8

6 Steve.Jobs 61 male 91.1

> dim(LungCapData2) lungcap datası 654 satır ve 5 sütunlu değişkenden oluşmaktadır.

[1] 654 5

> LungCapData2[1,c(1:5)] lungcap datasının ilk satırını aldım.

# A tibble: 1 x 5

Age LungCap Height Gender Smoke

1 9 3.12 57 female no

> LungCapData2[c(5,6,7,8,9),] c den sonra 2. değişkeni boş bırakırsan 5,6,7.. bu satırların tüm değerlerini gösterir.

> LungCapData2[5:7,] 5 ten 7 ye kadar tüm sütunları getirir.

Age LungCap Height Gender Smoke

1 9 3.68 57 male no

2 8 5.01 61 female no

3 6 3.76 58 female no

> names(LungCapData2)

[1] "Age" "LungCap" "Height" "Gender" "Smoke" names(lungdata) başlık isimlerini verir.

> names(LungCapData2[1]) ilk başlığı verir.

[1] "Age"

> mean(LungCapData2$Age) lungcapdata dan ortalamayı almasını söylüyoruz

[1] 9.931193 hangi değişkenin ortalamasını alacaksak $ işareti kullanmak zorundayız.

> LungCapData2$Age lungcapdata dan yaşı çeker hepsini matrise alır.

> attach(LungCapData2) r da lungcapdata yı hafızaya alır ve tanır.

artık mean(Age) diyebiliriz Age i artık tanıyo

sadece Age dersek bütün nesneler gelir.

> class(Height) class modülü ile değişken türünü görebiliriz.

[1] "numeric"

> class(Gender) cinsiyet veri türünün karakter olduğunu gördük

[1] "character"

> class(DataToExport$Gender) bu da attach kullanmadan dolar işareti kullanarak hafızaya alır

[1] "character"

attach ile LungCapData$Age aynı şeylerdir.

> factor(DataToExport$Gender) factor yani kategorik değişken gender cinsiyet character di kategorik olarak değiştirdim.

[1] male male female female male male female male

Levels: female male buradada levelleri seviyeleri var

> class(DataToExport$Gender)

[1] "factor"

> levels(LungCapData2$Gender) katgeorik değişkenlerin ne olduğunu bulmak için levels

[1] "female" "male"

> DataToExport$Gender=as.factor(DataToExport$Gender) \*\*değişken türünü değiştirmek için lungdata$age=as.factor(lungdata$age)

> class(DataToExport$Gender)

[1] "factor"

> levels(DataToExport$Gender)

[1] "female" "male"

summary(lungCapData2)=verinin özetini veirir.

> LungCapData2$Smoke=as.factor(LungCapData2$Smoke)

> summary(LungCapData2)

Age LungCap Height Gender Smoke

Min. : 3.000 Min. : 0.373 Min. :46.00 female:318 no :589

1st Qu.: 8.000 1st Qu.: 3.943 1st Qu.:57.00 male :336 yes: 65

Median :10.000 Median : 5.643 Median :61.50

Mean : 9.931 Mean : 5.910 Mean :61.14

3rd Qu.:12.000 3rd Qu.: 7.356 3rd Qu.:65.50

Max. :19.000 Max. :15.379 Max. :74.00

sigara içenlerin özeti

> summary(LungCapData2$Smoke)

no yes

589 65

yaş için özet

> summary(LungCapData2$Age)

Min. 1st Qu. Median Mean 3rd Qu. Max.

3.000 8.000 10.000 9.931 12.000 19.000

length(Age)

725 yaşın satırını verdi

> Age[11:14]

[1] 6 8 8 8 11 den 14 e kadar yaş değerlerini verir

> LungCapData2[11:14,]

Age LungCap Height Gender Smoke 11 den 14 e kadar olan tabolyu verir.

11 6 2.806 53.0 female no

12 8 3.205 54.0 male no

13 8 4.579 58.5 female no

14 8 4.354 60.5 male no

== işareti nesnelere değer atamak için kullanılır.

= matematiksel eşit demek.

> mean(Age[Gender=="female"]) Kadınların yaş ortalaması

[1] 9.842767

> mean(Age[Gender=="male"]) Erkeklerin yaş ortalaması.

[1] 10.01488

> erkekler=LungCapData2[Gender=="male",] sadece erkeklerin verilerini gösterir.

> mean(Age[Smoke=="yes"]) sigara içenlerin yaş ort

[1] 13.52308

> mean(Age[Smoke=="no"]) sigara içmeyenlerin yaş ortalaması

[1] 9.534805

> mean(Age[Smoke=="yes" & Gender=="1"]) sigara içen erkeklerin özeti

[1] 13.25641

> mean(Age[Smoke=="yes" & Gender=="2"]) sigara içen kadınların özeti.

[1] 13.92308

ERKEKLERİN VERİ ÖZETİ.

> erkeközet=summary(LungCapData2[Gender=="male",])

> erkeközet

Age LungCap Height Gender Smoke

Min. : 3.00 Min. : 0.388 Min. :47.00 female: 0 no :310

1st Qu.: 8.00 1st Qu.: 4.026 1st Qu.:57.00 male :336 yes: 26

Median :10.00 Median : 5.818 Median :62.00

Mean :10.01 Mean : 6.437 Mean :62.03

3rd Qu.:12.00 3rd Qu.: 8.604 3rd Qu.:67.50

KADINLARIN VERİ ÖZETİ.

> kadınözet

Age LungCap Height Gender Smoke

Min. : 3.000 Min. :0.373 Min. :46.00 female:318 no :279

1st Qu.: 8.000 1st Qu.:3.845 1st Qu.:57.50 male : 0 yes: 39

Median :10.000 Median :5.458 Median :61.00

Mean : 9.843 Mean :5.354 Mean :60.21

3rd Qu.:12.000 3rd Qu.:6.975 3rd Qu.:63.50

Max. :19.000 Max. :9.505 Max. :71.00

> e=summary(LungCapData2[Gender==1,c(3,5)])

Height Smoke

Min. :46.00 no :279 sigara içen erkeklerin boy ve sigara içme durumları

1st Qu.:57.50 yes: 39

Median :61.00

Mean :60.21

3rd Qu.:63.50

Max. :71.00

> a=LungCapData2[,c(1,4)] erkek ve kadınların yaşları cinsiyetleri tablosu

15 YAŞINDAN BÜYÜK ERKEKLER

> buyukerkek15=LungCapData2[Gender=="male" & Age>15,]

> buyukerkek15

Age LungCap Height Gender Smoke

609 19 13.306 72.0 male no

611 16 9.064 68.0 male yes

612 17 11.287 70.0 male no

621 17 7.246 67.0 male yes

626 18 10.660 68.0 male no

627 17 12.172 70.5 male no

629 17 8.218 69.0 male yes

632 17 14.899 73.0 male no ... Diye devameder hepsini almadım.

Mantık komutları True False.

> temp=Age>15 temp değişkeni atadım ve yaş 15 ten büyüktür dedim

> temp[1:5] 1 ile 5 arasını döndürdüm hiçbişey yazmasaydım hepsini getirirdi.

[1] FALSE FALSE FALSE FALSE FALSE

erkeklerin (>15) yaşını numeric yapmak 15 ten büyük ise 1,0 döndür

temp1=as.numeric(LungCapData2$Age>15)

> temp1[1:10]

[1] 0 0 0 0 0 0 0 0 0 ilk 10 nin hiçbiri 15 ten büyük değil.

> LungCapData2[1:10,]

Age LungCap Height Gender Smoke

1 9 3.124 57.0 female no

2 8 3.172 67.5 female no

3 7 3.160 54.5 female no

4 9 2.674 53.0 male no

5 9 3.685 57.0 male no

6 8 5.008 61.0 female no

7 6 3.757 58.0 female no

8 6 2.245 56.0 female no

9 8 3.961 58.5 female no

10 9 3.826 60.0 female no

> sigaraiçenkadın=kadınlar[kadınlar$Smoke=="yes",] ilk önce kadınlar tablosu oluşturmuştuk

> sigaraiçenkadın[1:8,] şimdide sigara içen kadınları çektik.

Age LungCap Height Gender Smoke

332 14 4.708 66.0 female yes

358 14 8.284 64.0 female yes

366 13 7.624 61.0 female yes

381 12 5.152 63.5 female yes

384 14 7.222 65.0 female yes

388 10 5.161 66.0 female yes

403 12 9.505 69.5 female yes

414 13 5.797 62.5 female yes

> a=sigaraiçenerkekler[1:5,] cbind tabloları birleştirir fakat aynı satır sayılarına sahip olmalılar.

> b=sigaraiçenkadın[1:5,]

> c=cbind(a,b)

Age LungCap Height Gender Smoke Age LungCap Height Gender Smoke

191 9 3.859 58 male yes 14 4.708 66.0 female yes

369 11 3.082 60 male yes 14 8.284 64.0 female yes

370 14 9.871 72 male yes 13 7.624 61.0 female yes

372 13 12.367 69 male yes 12 5.152 63.5 female yes

422 13 12.268 68 male yes 14 7.222 65.0 female yes

> chisq.test(LungCapData2$Gender,LungCapData2$Smoke) kikare testi p<0,05 sigara içmek cinsiyete bağlıdır.

Pearson's Chi-squared test with Yates' continuity correction

data: LungCapData2$Gender and LungCapData2$Smoke

X-squared = 3.2504, df = 1, p-value = 0.0714

dataya veri ekleme

> bilgi=data.frame(ad=character(),soyad=character(),yas=numeric()) bilgi adında data frame oluşturduk

> bilgi=rbind(bilgi,c("mehmet","kırkıncı",70)) rbind fonksiyonuyla veri ekledik.

> harun=c("harun","bakırcı",21)

> bilgi=rbind(bilgi,harun) böylede ekleyebiliriz

cbind vektörleri matrix e çevirir hepsini aynı değişken türünde yazar data frame öyledeğil

> L=x+q vektörleri toplayıp dataframe e ekleyebiliriz.

> t=data.frame(a,b,c) a,b,c data frame i

> t

a b c > avg=apply(X=t,MARGIN = 2,FUN=mean,na.rm=TRUE) hepsinin ort.

1 1 6 4 a b c

2 2 7 5 3.500000 6.500000 3.166667

3 3 8 3

4 4 9 1 > max=apply(t, MARGIN = 2,FUN=max,na.rm=TRUE)

5 5 5 3 a b c

6 6 4 3 6 9 5

if statemant

> if (mean(Age)>4) {print("büyük")} else {print("küçük")}

[1] "büyük"

> a=data.frame(ad="",soyad="",yas="") data frame başlık oluşturma

> names(a)=c("isim","soyisim","yas") data frame başlık değiştirme.

FAKTÖRLERİN DEĞERLERİNİ DEĞİŞTİRMEK

> a=c("harun","harun","mert","harun")

> levels(a)

[1] "harun" "mert"

> levels(a)[1]="kaan"

> levels(a)

[1] "kaan" "mert"

> a

[1] kaan kaan mert kaan

Levels: kaan mert

iki faktörlü test

> t.test(t\_test$extra~t\_test$groups)

Welch Two Sample t-test

data: t\_test$extra by t\_test$groups

t = -1.8608, df = 17.776, p-value = 0.07939

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-3.3654832 0.2054832

sample estimates:

mean in group 1 mean in group 2

0.75 2.33

ıd extra groups

<dbl> <dbl> <dbl>

1 1 0.7 1

2 2 -1.6 1

3 3 -0.2 1

4 4 -1.2 1

5 5 -0.1 1

6 6 3.4 1

7 7 3.7 1

8 8 0.8 1

9 9 0 1

10 10 2 1

11 11 1.9 2

12 12 0.8 2

13 13 1.1 2

14 14 0.1 2

15 15 -0.1 2

16 16 4.4 2

17 17 5.5 2

18 18 1.6 2

19 19 4.6 2

20 20 3.4 2

> names(sleepwide)[]

[1] "ıd" "extra" "groups" "ıd.1" "extra.1" "groups.1"

> names(sleepwide)[3]="grup1"

> names(sleepwide)[4]="ıd"

> names(sleepwide)[6]="grup2" grupların isimlerini değiştirmek.

> apply(data.frame(t\_test$groups,t\_test$extra),MARGIN = 2,FUN = mean) data frame e t testten grup ve extrayı aldım

t\_test.groups t\_test.extra

1.50 1.54

EŞ ÖRNEKLEM T-TEST

> t.test(t\_test$extra~t\_test$groups,paired=TRUE)

Paired t-test

data: t\_test$extra by t\_test$groups

t = -4.0621, df = 9, p-value = 0.002833

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-2.4598858 -0.7001142

sample estimates:

mean of the differences

-1.58

ONE SAMPLE T-TEST t\_test benim veri setim

> t.test(t\_test$extra,mu=0) fakat t testi yapmak için yine t\_test yazıyoruz

One Sample t-test

data: t\_test$extra

t = 3.413, df = 19, p-value = 0.002918

alternative hypothesis: true mean is not equal to 0

95 percent confidence interval:

0.5955845 2.4844155

sample estimates:

mean of x

1.54

> a=0.565966 yuvarlama komutu

> round(a,2)

[1] 0.57

> a=c(0.56586,0.236585,0.2548555,0.36669) yuvarlama komutu

> round(a,2)

[1] 0.57 0.24 0.25 0.37

str fonksiyonu

grup1 extra1 grup2 extra2 dataframe hakkında bilgi verir.

1 kadın 0.7 erkek 1.9 dataframe özelliklerini yazar.

2 kadın -1.6 erkek 0.8

3 kadın -0.2 erkek 1.1

4 kadın -1.2 erkek 0.1

5 kadın -0.1 erkek -0.1

6 kadın 3.4 erkek 4.4

7 kadın 3.7 erkek 5.5

8 kadın 0.8 erkek 1.6

9 kadın 0.0 erkek 4.6

10 kadın 2.0 erkek 3.4

> str(a)

'data.frame': 10 obs. of 4 variables:

$ grup1 : Factor w/ 2 levels "kadın","erkek": 1 1 1 1 1 1 1 1 1 1

$ extra1: num 0.7 -1.6 -0.2 -1.2 -0.1 3.4 3.7 0.8 0 2

$ grup2 : Factor w/ 2 levels "kadın","erkek": 2 2 2 2 2 2 2 2 2 2

$ extra2: num 1.9 0.8 1.1 0.1 -0.1 4.4 5.5 1.6 4.6 3.4

aynı başlıklara sahip başka bir data.frame oluşturup

tabloları birleştirebilirsin

b=data.frame(ıd="21",extra=1.5,groups=2)

detach(lungcap) lungcap i hafızadan siliyor. attach ın tersi.

yukarıdaki toollardan sesion save workspace yaptıktan sonra kaydedilen dosyada ls() yazarsan kayıtlı dosyaları verir.

rm(list=ls()) tüm çalışma sayfalarını kaldırır.

> plot(Age,col="red") kırmızı noktalarla yaş plotu oluşturur.

> plot(LungCapData2$Age[Smoke=="no"],col="blue") sigara içmeyenlerin yaş grafiği

> data.frame(grup1,aa=c(1,2,3,4,5,6,7,8,9,5))

ıd extra groups aa

1 1 0.7 1 1

2 2 -1.6 1 2 grup1 data.frame ine aa sütunu ekledi.

3 3 -0.2 1 3

4 4 -1.2 1 4

5 5 -0.1 1 5

6 6 3.4 1 6

7 7 3.7 1 7

8 8 0.8 1 8

9 9 0.0 1 9

10 10 2.0 1 5

getwd() çalışma sayfasını nereye kaydettiğini gösterir.

wiev(mtcars) r ın içindeki mtcars dosyası açılır

> arabaveri=subset(mtcars,select = c("mpg","cyl","wt","am")) mtcars dosyasından birkaç veri çekerek arabaveri tablosu oluşturduk

> colnames(arabaveri)=c("yol","silindir","ağırlık","vites") tablonun sütun adlarını değiştirdim.

GGPLOT FONKSİYONU

aes estetik kelimesi demek şekillerin özellikleri

ggplot(veri seti ,aes(x ekseni için değişken ,y ekseni için değişken))

geom\_bar()=çizgi grafiği

geom\_poin()=nokta graf.

geom\_line()=doğru grafiği

geom\_histogram()=histogram grafiği

geom\_boxplot()=boxplot grafiği.

geom\_smooth()=regresyon grafiği.

GRAFİK OLUŞTURMA

> grafigim=ggplot(arabaveri,aes(ağırık,yol)) sadece grafik tablosunun eksenlerini oluşturur. özelliklerini

> grafigim+geom\_point() hangi grafiği ekleyeceksen o grafiğin adını birleştirmen lazım ör. grafiğim+geom\_histogram() gibi.

> grafigim2=ggplot(arabaveri,aes(silindir,yol)) kırmızı noktalarla grafik oluşturur.

> grafigim2+geom\_point()

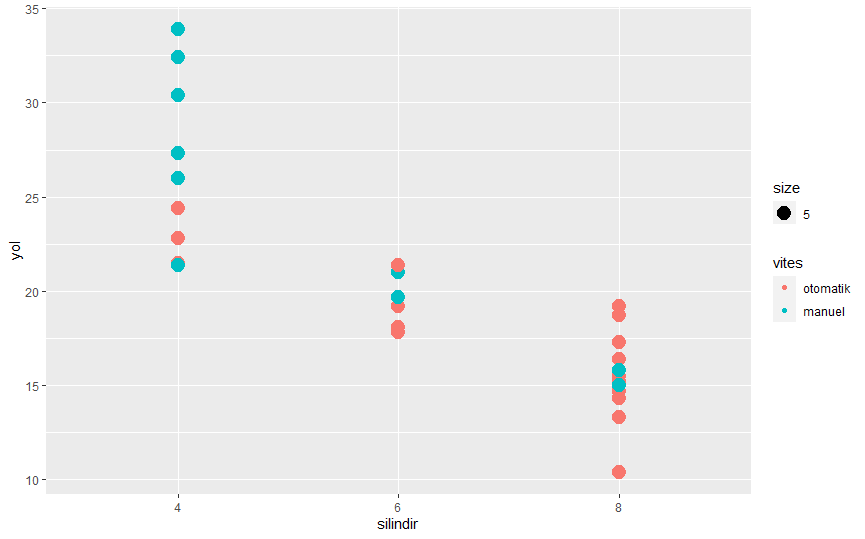
> grafigim2+geom\_point(colors="red") shape=17 grafik noktalarının şekillerini belirler 17 üçgen demek artık

> grafigim2+geom\_point(color="red",shape=17) grafif noktaları üçgen şeklinde olur.

> grafigim2+geom\_point(color="red",shape=15,size=5) size=5 yaparsak noktaların büyüklüğünü belirler.

> grafigim2+geom\_point(aes(colour=vites),size=5) grafiğe vites değişkenini de ekler otomatik manuel diye ayırır.

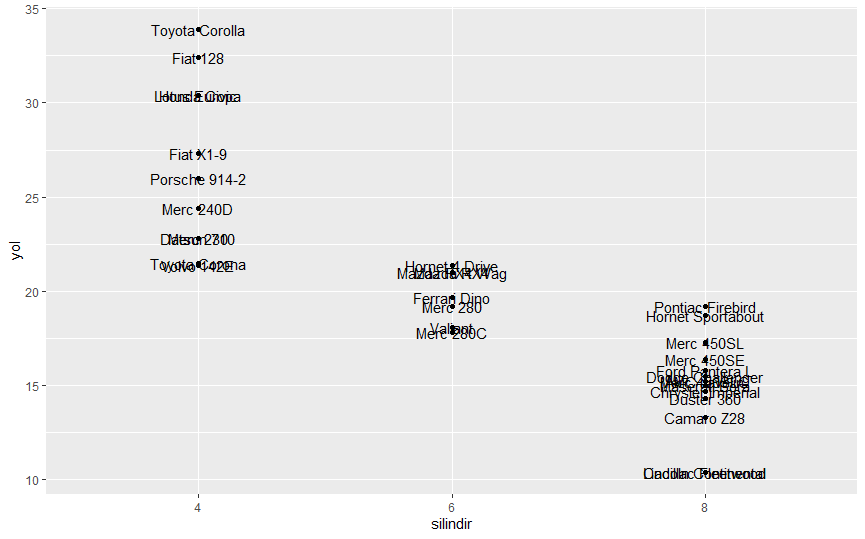
herbir katman için + sembolü kullanıyoruz



grafigim2=ggplot(arabaveri,aes(silindir,yol,label=rownames(arabaveri))) plotumun tasarımına yeni bir label faktorü ekliyorum.

grafigim2+geom\_point()+geom\_text()

hangi grafikle çalıştıracağımı belirtiyorum. grafikte araç isimlerini yansıtır. etiketli grafik oluşturdum.



Grafikte çakışan noktaları görebilmek için

> grafigim2=ggplot(arabaveri,aes(silindir,yol))

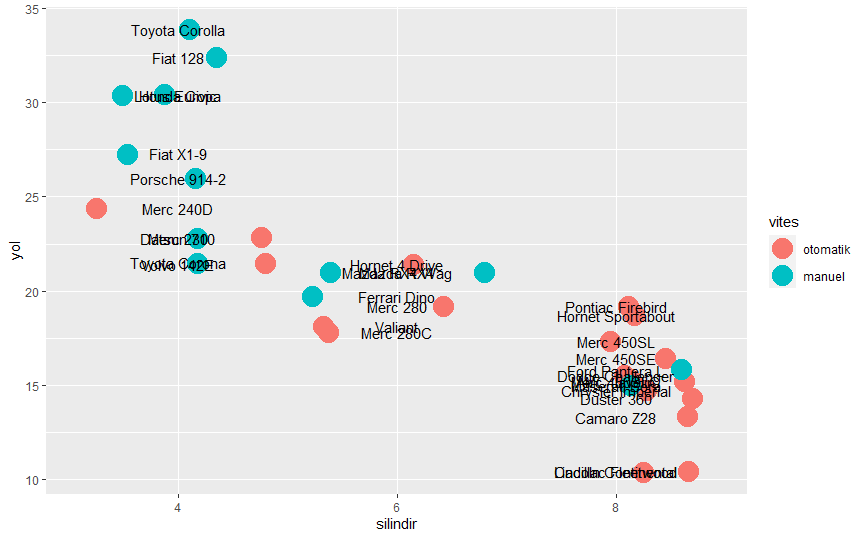
> grafigim2+geom\_point(position = "jitter") jitter toolunu ekliyorum çakışan noktalar yanyana geliyor.

> grafigim2=ggplot(arabaveri,aes(silindir,yol,label=rownames(arabaveri))) Noktalar ve isimleri gelir. vites değişkeni gelmez

> grafigim2+geom\_point(position = "jitter",size=5)+geom\_text()

> grafigim2=ggplot(arabaveri,aes(silindir,yol,label=rownames(arabaveri))) vites değişkenine göre nokta isim grafiği.

> grafigim2+geom\_point(aes(color=vites),position = "jitter",size=7)+geom\_text()



saçılım grafiği iki kategorik değişkenin incelenmesi için kullanılır.

grafik çizmek için ggplot2 paketini aktif etmek gereklidir.

sacılım+geom\_point()+labs(x="Taç yaprağı boyu",y="Taç yaprağı eni")

eksenlere isim verebilmek için labs fonksiyonuyla etiket yapıştırırız.

sacılım+geom\_point(size=5,color="red",shape=21) noktalar ile ilgili işlemler burada yapılır.

> sacılım=ggplot(iris,aes(iris$taçyaprağı\_boyu,iris$taç\_yaprağı\_eni,color=türü))

> sacılım+geom\_point()

color=türü diyerek tür değişkenlerinin farklı renkte oluşmasını sağlarız.

akciğer kapasitesi ile boy arasındakiilişkiyi cinsiyet kategorik değişkenini dikkate alarak değerlendirdik.

> lungcapgrafik=ggplot(LungCapData2,aes(Height,LungCap,color=Gender))

> lungcapgrafik+geom\_point(size=4,position = "jitter")

Yaş ile Akciğer kapasitesi arasındaki ilişki cinsiyet göz önüne alınarak oluşturulmuştur.

> lungcapgrafik=ggplot(LungCapData2,aes(Age,LungCap,color=Gender))

> lungcapgrafik+geom\_point(size=5,position = "jitter")+labs(x="YAŞ",y="AKCİĞER KAPASİTESİ")

geom\_smooth grafiğe regresyon doğrusu ekler.

> grafigim=ggplot(iris,aes(taçyaprağı\_boyu,taç\_yaprağı\_eni))

> grafigim+geom\_point(color="blue")+geom\_smooth(method="lm",colour="red")

method regresyon doğrusunun düz çizgi olmasını color ise regresyon çizgisinin rengini özelleştirir.

çizginin arkasındaki gölgede %95 güven aralığını temsil eder.

kategorilere göre regresyon doğrusu

> grafigim=ggplot(iris,aes(taçyaprağı\_boyu,taç\_yaprağı\_eni,color=türü))

> grafigim+geom\_point(color="blue")+geom\_smooth(method="lm") türü kolonunu eklememiz yeterli.

BAR CHART

> ortNot=c(80,90,85,95,60,100)

> sınıf=c("A","A","B","B","C","C")

> cinsiyet=c("K","E","K","E","K","E")

> matNot=data.frame(ortNot,sınıf,cinsiyet)

> bar=ggplot(matNot,aes(x=sınıf,y=ortNot,fill=cinsiyet)) 3.bir değişken olduğu için cinsiyeti böyle ekledik

> bar+geom\_bar(stat="identity") a,b,c sınıfındaki erkek kız notlarını üst üste 3 farklı sınıfta gösterir.

> bar+geom\_bar(stat="identity",position = "dodge") position ise cinsiyetleri üstüste değilde yanyana kutularda belirtir.

sıra isim

1 5 harun

2 6 ahmet null fonksiyonu sütun siler.

3 7 sadık

4 8 kemal

> a[,1]=NULL

> a

isim

1 harun

2 ahmet

3 sadık

4 kemal

Mean(a,na.rm=TRUE) dersen kayıp verileri baz almaz

İs.na(exam) veride kayıp olup olmadığını kontrol eder.

Sum(is.na(exam)) boş satrları sayar.

Example[is.na(example)]=0 dersen boş değerlere 0 atar.

for(i in 1:10){print(i)}

çalışma mantığı for(){}

for(i in list("harun","bakırcı","samsun","istanbul")){if (i =="harun"){print("bravooo")}

}

Liste oluşturma ve veri ekleme

> a=list("harun","bakırcı",20)

a=list(a,"ben")

ben nesnesi listeye eklenir

a=list(DataToExport$Age)

for(i in a){print(a)}

53 54 38 119 70 28 62 61

> for(i in DataToExport$Age){if (i>60){print(i)}}

[1] 119

[1] 70

[1] 62

[1] 61

> sd(DataToExport$Age) standart sapma formülü

[1] 27.19211

B=10 20 10 30 25 10 b deki 10 ları say

> length(which(b==10))

[1] 3

a b isimler a dataframe i

1 5 15 harun

2 10 16 mehmet

3 20 23 kerem

4 30 32 ihsan

> length(which(a$isimler=="harun"))

[1] 1

İsimlelrdeki harun sayısı

Sadece harunun bulunduğu satırları al

> a[a$isimler=="harun",]

a b isimler

1 5 15 harun

a b isimler a dataframe si

1 5 15 harun

2 10 16 mehmet

3 20 23 kerem

4 30 32 ihsan

5 15 10 harun

> a[a$isimler=="harun",]

a b isimler

1 5 15 harun

5 15 10 harun

> length(DataToExport) değişken sayısını verir.

[1] 4

Yaşı 60 tan büyük olanların sayısı

> length(which(DataToExport$Age>60))

[1] 4

Datatoexport ta erkeklerin yaşlaro toplamı

> sum(DataToExport$Age[DataToExport$Gender=="male"])

[1] 266

Veriyi frekanslara dönüştürür.

> a=c(10,20,10,20,10,15,13,16,17)

> table(a)

a

10 13 15 16 17 20

3 1 1 1 1 2

> a=c("harun","berk","harun",10,20,10,10,"harun")

> table(a)

a

10 20 berk harun

3 1 1 3

> tablo=data.frame(kişi,notlar)

> tablo

kişi notlar

1 berk 20

2 burcu 60

3 yağmur 15

4 ali 85

5 sıla 92

> length(which(tablo$notlar>54)) 54 ten büyük not al [1] 3 kişi sayısı

b=data.frame(GNO=krediler\*notlar/sum(krediler))

b değişkenine gno yu hesapladık kredi\*not/toplam kredi

plot(otosayısı)

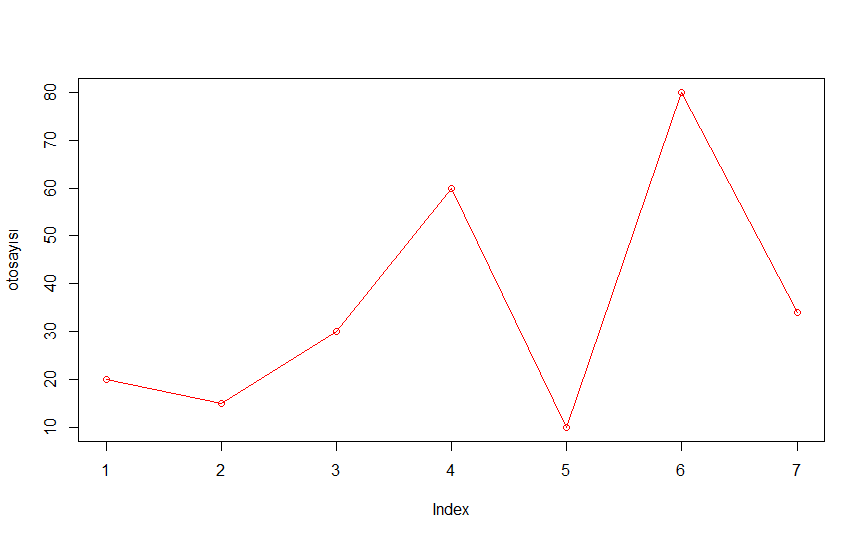
nokta grafiği oluşturur fakat dataframe nin değil vektörün nokta grafiği

plot(otosayısı,type="o",color="blue")

noktaları birleştirir çizgi grafiği yapar. Tek koordinat alır x y almaz

otosayısı=c(20,15,30,60,10,80,34)

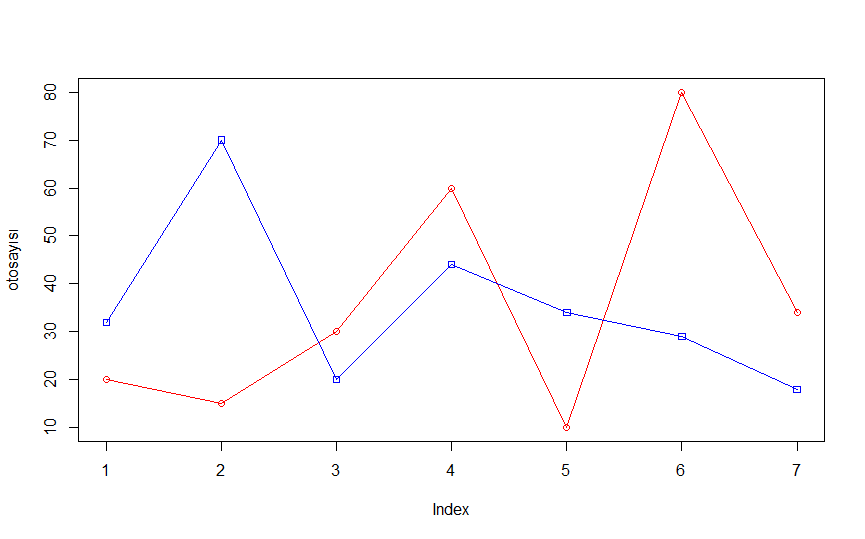
sadece bunun grafiği



sadece c vektörü üstüne line ile çizgi ekleyebilirsin

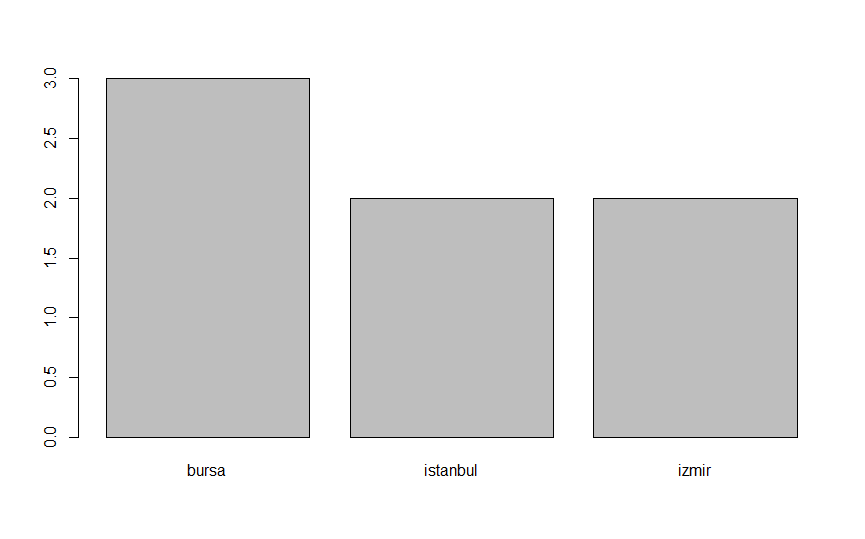
tır\_sayısı=c(32,70,20,44,34,29,18)

lines(tır\_sayısı,type="o",pch=22,col="blue")



Histogram grafiği (BARPLOT)

x=c("bursa","bursa","bursa","izmir","izmir","istanbul","istanbul")

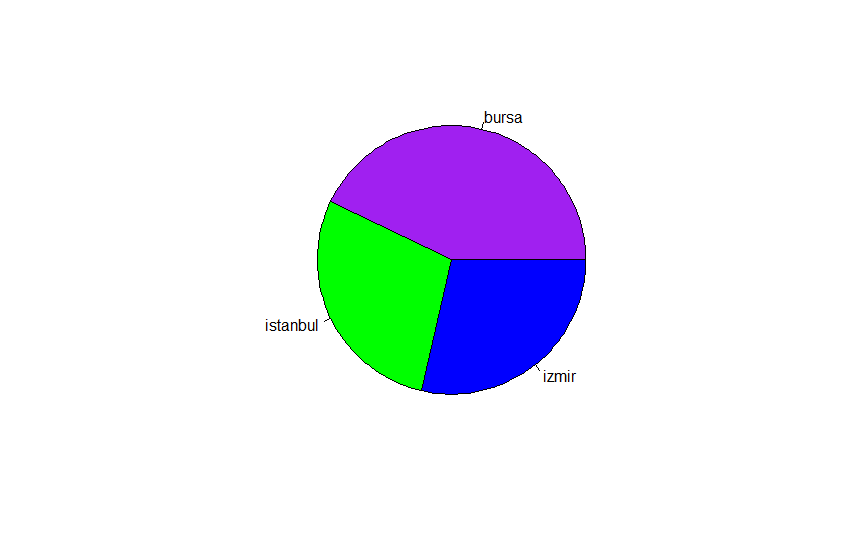


bursa istanbul izmir

3 2 2

> pie(xfrekansları,col=c("purple","green","blue"))

Pasta grafiği



x=c(1,1,1,4,2,9,2,6,4,8,5,4)

[1] 1 1 1 4 2 9 2 6 4 8 5 4

> x=c(x,5,0,6,7,9,1)

[1] 1 1 1 4 2 9 2 6 4 8 5 4 5 0 6 7 9 1

X e yeni veri ekleme

> xfrekansları

x

bursa istanbul izmir

3 2 2

> cumsum(xfrekansları)

bursa istanbul izmir

3 5 7

Cumsum(x) kümülatif toplam yapar

X in kümülatif frekansını al grafiğini çiz

> barplot(cumsum(table(x)),col=c("purple","green","blue","white","yellow"))

Ki kare tablosu

X= içiyor içmiyor içiyor içmiyor içiyor içmiyor içiyor içmiyor içiyor içmiyor

Levels: içiyor içmiyor

Y=kadın erkek kadın erkek kadın erkek kadın erkek kadın erkek

Levels: kadın erkek

Table(x,y)

kadın erkek

içiyor 5 0

içmiyor 0 5

a=1 3 5 6 7 5 4 9 2 sort sıralama

> sort(a)

[1] 1 2 3 4 5 5 6 7 9

table(x,y,a)

, , a = lise

y

x kadın erkek

içiyor 0 0

içmiyor 5 0

, , a = üniversite

y

x kadın erkek

içiyor 0 5

içmiyor 0 0

> for(i in 1:length(a)){print(a[i])}

[1] "harun"

[1] "berk"

[1] "kamil"

[1] "mehmet"

İ in 1:length(a) dedim 1 den uzunluğa kadar sadece length(a) olmaz

satıcı satışlar cumsatış frekanssat. cumfrekans

1 harun 100 100 0.08771930 0.0877193

2 berk 150 250 0.13157895 0.2192982

3 kamil 200 450 0.17543860 0.3947368

4 harun 100 550 0.08771930 0.4824561

5 berk 150 700 0.13157895 0.6140351

6 mehmet 50 750 0.04385965 0.6578947

7 harun 120 870 0.10526316 0.7631579

8 berk 180 1050 0.15789474 0.9210526

9 kamil 90 1140 0.07894737 1.0000000

a=satıcı isimleri

a="harun" "berk" "kamil" "mehmet"

> for(i in 1:length(a)){print(cumsum(satışlar[satıcı==a[i]]))}

[1] 100 200 320

[1] 150 300 480

[1] 200 290

[1] 50

Satıcıların kümülatif toplamlarını verir. Mükemmell

> for(i in 1:length(a)){print(length(which(satıcı==a[i])))}

[1] 3

[1] 3 kişilerin kaç adet satış yaptığını veririr.

[1] 2

[1] 1

> for(i in 1:length(a)){print(sum(satışlar[satıcı==a[i]]))}

[1] 320

[1] 480 kişilerin toplam satışları

[1] 290

[1] 50

[1] "harun" "berk" "kamil" "mehmet" "murat"

> data.entry(a)

Veri tablosu açılır veri girişi yapabilirsin.

Dataexport=edit(dataexport) deyip açılan veri giriş setinden giriş yapıp kaydedebilirsin.

Edit komutuyla veriye 2 kere tıklayıp değiştiebilirsin.

C değişkenine for döngüsüyle nesne ekledik.

> c=c("harun","bakırcı")

> for(i in 1:5){c=c(c,i)}

> c

[1] "harun" "bakırcı" "1" "2" "3" "4" "5"

Abs()mutlak değer round(sayı,2) yuvarlama komutu rnorm rastgele sayı üret

> round(abs(rnorm(5)),2)

[1] 0.18 0.73 1.54 1.08 0.94

Barplotta ana başlık main e atanır names.arg ise çubukların altılarındaki isimleri gösterir.

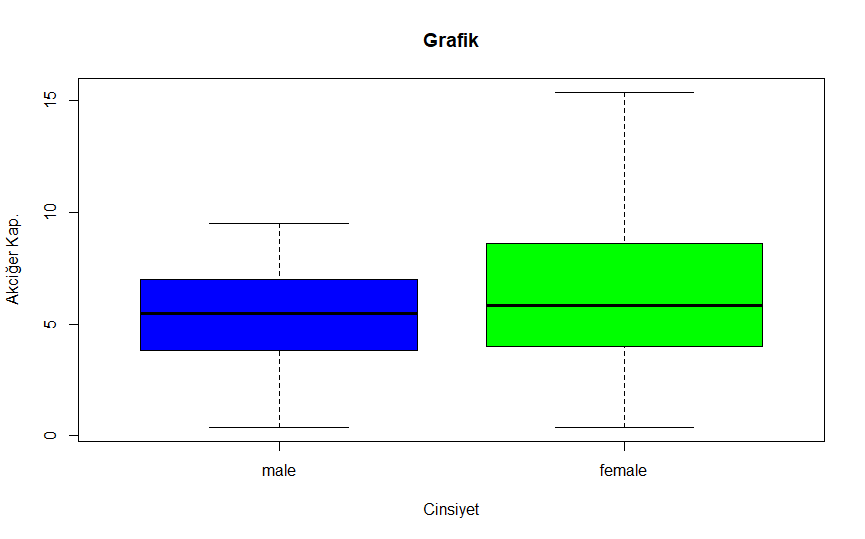
Yüzde diye kadın erkek frenkans tablosu oluşturdum.

> barplot(yüzde,main="Grafik",xlab="Cinsiyet",ylab="%",names.arg=c("kadın","erkek"))

İki değişkenin boxplotta gösterilmesi.

> boxplot(LungCap~Gender,main="Grafik",xlab="Cinsiyet",ylab="Akciğer Kap.",",col=c("blue","green")

)



İsim="harun" "ahmet" "sadık" "mert" "cem"

> nchar(isim)

[1] 5 5 5 4 3

İsimlerin kaçar karakterden oluştuğunu sayar

İsim="harun" "ahmet" "sadık" "mert" "cem"

> grep("mert",isim)

[1] 4

Mert in kaçıncı sırada olduğunu gösterir.

İsim="harun" "ahmet" "sadık" "mert" "cem"

> paste(isim,c(5,6,7,8,9))

[1] "harun 5" "ahmet 6" "sadık 7" "mert 8" "cem 9"

Paste birleştirme işlemi yapar

> paste("harun","mert")

[1] "harun mert"

> paste("harun","mert","kaan")

[1] "harun mert kaan"

> paste("harun","mert","kaan",sep="5")

[1] "harun5mert5kaan"

Aralara 5 ekler

a b c dataframe miz

1 10 a şimdi bunu unstack yapacağız.

2 20 a

3 30 b

4 50 b

5 60 b

6 80 b

> unstack(c) a sütununu ayrı b sütunnun ayrı verdi.

$a

[1] 10 20

$b

[1] 30 50 60 80

> genişver=data.frame(unstack(c))

Altalta olan c veri tabanını frame e dönüştürür

a b

1 10 30

2 20 50

3 10 60

4 20 80

> stack(genişver,select=c("a","b"))

values ind

1 10 a

2 20 a

3 10 a

4 20 a

5 30 b

6 50 b

7 60 b

8 80 b

R ile inputbox girdisi:

> t=readline("lütfen isminizi giriniz:")

lütfen isminizi giriniz:HARUN BAKIRCI

> t

[1] "HARUN BAKIRCI"

Rastgele sayı üretme

> runif(5,min=5,max=20)

[1] 14.717929 7.980028 7.573066 7.520780 18.593329

Min 5 max 20 olan 5 sayı üret.

> kişiadı=readline("merhaba adınzı giriniz lütfen:")

merhaba adınzı giriniz lütfen:harun

> print(paste("hoşgeldiniz",kişiadı,"bey"))

[1] "hoşgeldiniz harun bey"

> str(LungCapData2)

tibble [654 x 5] (S3: tbl\_df/tbl/data.frame)

$ Age : num [1:654] 9 8 7 9 9 8 6 6 8 9 ...

$ LungCap: num [1:654] 3.12 3.17 3.16 2.67 3.68 ...

$ Height : num [1:654] 57 67.5 54.5 53 57 61 58 56 58.5 60 ...

$ Gender : Factor w/ 2 levels "male","female": 1 1 1 2 2 1 1 1 1 1 ...

$ Smoke : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...

Döngü 5 kere döner eğer giridiğimiz sayılar 5 ten büyükse sayı 5 ten büyüktür der.

for(i in 1:5){t=as.numeric(readline("bir sayı giriniz:"))

if(t>5){print(paste(t," sayısı 5 ten büyüktür"))}

}

for(i in 1:5){t=as.numeric(readline("bir sayı giriniz:"))

if(t>5){print(paste(t," sayısı 5 ten büyüktür"))}

else{

print(paste(t, "sayısı 5 ten küçük veya eşittir."))

}}

For ve if statement

For(i>5){}

İf(i=5){}

Else if(i<5){}

İf(i=5){}else{}

for(i in c(1,7,3,6,4,9)){

if(i>5){

b=b+i

}

}

print(b) sonuç 22’dir.

Çift sayıları topla cevap 10’dur.

for(i in c(1,7,3,6,4,9)){

if(i%%2==0){

b=b+i

}

}

print(b)

11%%3 11 in 3’e bölümünden kalan nedir.

Break komutu döngüden çıkar

for(i in c(1,7,3,6,4,9)){

if(i%%2==0){

b=b+i

break()

}}

print(b) çift sayıyı bulduğu zaman döngüden çıkar cevap 6.

b=list()

for(i in c("ahmet","kemal","tarık","harun","kemal")){

if(i=="kemal"){

b=paste(b,i)

}

} print(b) çıktısı " kemal kemal"

> c=c("ahmet","kemal","tarık","harun","kemal")

> grep("kemal",c)

[1] 2 5

Bir değişkenin faktörlerini değiştirmek için ilk önce character sonra factor e dönüştrür.

> nrow(tablo) nrow son satırı verir.

[1] 5

!= r da eşit değildir işareti.

for(i in 1:nrow(başarı2)){ burada c ye i yi yazdırdık yani çift olanların sırasını yazdırdı

if(başarı2[i,2]%%2==0){

c=c(c,i) çıktısı 1 3 5

}}

Çift olan değerleri istersek.

for(i in 1:nrow(başarı2)){ bu ise çift olan değerleri verir direkt

if(başarı2[i,2]%%2==0){ çıktısı 80 90 50

c=c(c,başarı2[i,2])

}}

Eğer sayıysa is.numeric(xx)

if ( test\_expression1) {

statement1

} else if ( test\_expression2) {

statement2

} else if ( test\_expression3) {

statement3

} else {

statement4

}

İf(x>20){} else if(x<10){}

Diye gider else if önceki if in {} yanından başlar her zaman.

a=0

message("Harun Banka Hoşgeldiniz:")

islem=readline("para eklemek içi 1'e

para çekmek için 2'ye basınız:")

if(as.numeric(islem==1)){para=as.numeric(readline("lütfen miktar giriniz:"))

a=a+para} else if(as.numeric(islem==2)){para=as.numeric(readline("lütfen miktar giriniz:"))

a=a-para}

> A=sample(c("harun","ahmet","kemal","nadir"),70,replace=TRUE)

Atadığım vektörden 70 adet üretir ve karıştırır.

Vektörler sütun veya satır oluşturur.

pt\_data[c(1, 3), c(2, 4)]

temperature gender

1 98.1 MALE

3 101.4 MALE

**pt\_data[-2, c(-1, -3, -5, -6)] 2. Satır ve 1,3,5,6. Sütunlar hariç**

> c=1 3 5 6 7

> -c=-1 -3 -5 -6 -7

> a=readline("lütfen isminizi giriniz:")

> print(paste("merhaba",a,"bey","hoşgeldiniz"))

[1] "merhaba harun bey hoşgeldiniz"

> summary(tablo[c("kişi","notlar")])

> summary(tablo[c(1,2)])

> a=data.frame(ort.not=mean(tablo$notlar),medya.not=median(tablo$notlar))

> a

ort.not medya.not

1 54.4 60 tablo oluşturma

> range(tablo$notlar) min max u verir.

[1] 15 92

> data.frame(tablo$notlar,dif=c(0,diff(tablo$notlar)))

tablo.notlar dif

1 20 0

2 60 40 bir öncekinin farkını alır.

3 15 -45

4 85 70

5 92 7

> diff(range(tablo$notlar))

[1] 77 en düşük notla en yüksek notun farkını alır.

Listeden adı harun olanları sil

a=c("harun","mert","kemal","harun","harun","cemal","harun")

for(i in 1:7){

if(a[i]=="harun"){

a=a[-i]

}}

> rbind(c("kadın",ort=kadınort),c("erkek",ort=erkekort))

ort

[1,] "kadın" "45"

[2,] "erkek" "81"

C= 20 80 95 180 272 342

> mydata=ifelse(c>80,c("older"),c("younger"))

> mydata

[1] "younger" "younger" "older" "older" "older" "older"

[1] 20 80 95 180 272 342

> which(F>80)

[1] 3 4 5 6

b=c("ilkokul", "lise", "üniversite", "lise" ,"ilkokul")

k=grep("lise",b)

b=b[-k]

b=c("ilkokul" ,"üniversite" ,"ilkokul")

&& ve demek

a <- 10

b <- 100

**c** <- 50

**if**(a>b && a>c){

**print**("A is bigger than others!")

} **else** **if**(b>a && b>c){

**print**("B is bigger than others!")

} **else** {

**print**("C is the biggest one!")

}

|  |  |
| --- | --- |
| **Operator** | **Description** |
| **<** | less than |
| **<=** | less than or equal to |
| **>** | greater than |
| **>=** | greater than or equal to |
| **==** | exactly equal to |
| **!=** | not equal to |
| **!x** | Not x |
| **x | y** | x OR y |
| **x & y** | x AND y |
| **isTRUE(x)** | test if X is TRUE |

I veya o operatörü demek

x <- c(1:10)  
x[(x>8) | (x<5)]  
# yields 1 2 3 4 9 10

r tarih saat r ın tarih algılama biçimi araya “-“ koyar.

İlk önce lubridate kütüphanesini çağır

Library(lubridate)

ymd(20110512) yıl ay gün formatında alır.

[1] "2011-05-12"

> ymd(c("20110516","20200613","20150411"))

[1] "2011-05-16" "2020-06-13" "2015-04-11"

> dmy(05062011)

[1] "2011-06-05"

> mdy(05062015)

[1] "2015-05-06"

Excel Fonksiyonlarıı

**Basic Arithmetic Functions**

| **EXCEL FORMULA** | **R FUNCTION** |
| --- | --- |
| COUNTA | length |
| CEILING, ROUNDUP | ceiling |
| FLOOR, ROUNDDOWN | floor |
| EXP | exp |
| LOG, LN | log |

**Basic Text Functions**

| **EXCEL FORMULA** | **R FUNCTION** |
| --- | --- |
| CONCATENATE | paste |
| LEFT, MID, RIGHT | substr |
| LOWER | tolower |
| UPPER | toupper |
| LEN | nchar |
| FIND | grepl, grep and others |
| REPLACE | gsub and others |

**More Formulas**

| **EXCEL FORMULA** | **R FUNCTION** |
| --- | --- |
| ISNA | is.na |
| MATCH | which |
| REPT | rep |
| RANK | rank |
| VLOOKUP | merge ([sort of](https://www.rforexcelusers.com/vlookup-in-r/)) |

**Excel Features**

Here are some Excel features and some of their close counterparts in R:

| **EXCEL FEATURE** | **R FUNCTION(S)** |
| --- | --- |
| Format Cells | as.Date, as.numeric, as.character (sort of) |
| Navigating / inspecting table | head, tail, str, summary, dim, nrow |
| Text to Columns | strsplit (sort of) |

> l="harun"

> length(l) length kaç adet veri oldugu

[1] 1

> nchar(l) nchar kaç karakter olduğu excel len

[1] 5

l= "harun" "bakırcı" "samsun"

> nchar(l)

[1] 5 7 6

l= "harun" "bakırcı" "samsun"

> gsub("harun","mert",l)

[1] "mert" "bakırcı" "samsun"

gsub replace demek harun olanları mert yap

l="harun"

> substr(l,1,4) parça al fun.

[1] "haru"

Ncol(tablo) 5 kaç sütun olduğu

Kategorik normallik testi

> by(LungCap,Gender,shapiro.test)

Gender: male

Shapiro-Wilk normality test

data: dd[x, ]

W = 0.98532, p-value = 0.002505

-----------------------------------------------------

Gender: female

Shapiro-Wilk normality test

data: dd[x, ]

W = 0.95673, p-value = 2.126e-08

Geom\_bar() fonksiyonu sadece frekans tablosu çıkarır.

Vektörleri isimlendirebiliriz

> ort

[1] 25461.52 25461.52

> names(ort)=c("kadın","erkek")

> ort

kadın erkek

25461.52 25461.52

Fonksiyon oluşturma

> karealma=function(x,y){x^2+y}

> karealma(3,5)

[1] 14

> karhesap=function(kar,gider){print(paste("karınız:",kar-gider))}

> karhesap(100,50)

[1] "karınız: 50"

> karhesap(100,26)

[1] "karınız: 74"

> tablo$GNO=round(tablo$GNO,2)

> select(tablo,notlar,GNO)

> filter(tablo,substr(kişi,1,2)=="ha")

kişi notlar krediler GNO cins

1 harun 70 3 3.5 erkek

2 hasan 80 4 3.0 erkek

3 hakan 75 2 4.0 erkek

> filter(tablo,substr(kişi,1,2)=="ha" & tablo$kredi>2)

kişi notlar krediler GNO cins

1 harun 70 3 3.5 erkek

2 hasan 80 4 3.0 erkek

Notları büyükten küçüğe sıraladı

> arrange(tablo,desc(notlar))

kişi notlar krediler GNO cins

1 sıla 92 2 2.25 erkek

2 ali 85 3 3.15 kadın

3 hasan 80 4 3.00 erkek

4 hakan 75 2 4.00 erkek

5 harun 70 3 3.50 erkek

6 burcu 60 2 2.50 kadın

7 berk 20 2 3.33 kadın

8 yağmur 15 3 3.75 kadın

Böyle yazarsan küçükten büyüğe sıralar.

> arrange(tablo,notlar)

kişi notlar krediler GNO cins

1 yağmur 15 3 3.75 kadın

2 berk 20 2 3.33 kadın

3 burcu 60 2 2.50 kadın

4 harun 70 3 3.50 erkek

5 hakan 75 2 4.00 erkek

6 hasan 80 4 3.00 erkek

7 ali 85 3 3.15 kadın

8 sıla 92 2 2.25 erkek

Tablodan 3 tane rastgele örnek alır.

> sample\_n(tablo,3)

kişi notlar krediler GNO cins

1 berk 20 2 3.33 kadın

2 hasan 80 4 3.00 erkek

3 hakan 75 2 4.00 erkek

L=”harun babaa”

> str\_locate(l,"baba") baba nın kaçla başlayıp kaçla bittiğini söyler.

start end

[1,] 7 10

> str\_detect(l,"k") l de k harfinin olup olmadığını sorgular.

[1] FALSE

> l="harun baba"

> str\_locate(l,"ba")

start end

[1,] 7 8

> str\_locate(l,"ba")[1]

[1] 7

Grupları kendi aralarında karşılaştırırken standart hata eklenmeli ort + std err

Sadece grup beliritilecek ise ortalama + standart sapma

Library(psych)

Describe(tablo) tanımlayıcı istatistikler

Describe.by(tablo,grouped=cins) kategorik olarak sunar.

Korelasyon testi

> cor.test(gelir,reklam)

Pearson's product-moment correlation

data: gelir and reklam

t = 0.85595, df = 48, p-value = 0.3963

alternative hypothesis: true correlation is not equal to 0

95 percent confidence interval:

-0.1612373 0.3877282

sample estimates:

cor

0.1226134

Kikare testi

> chisq.test(sigara,bölgeler)

Pearson's Chi-squared test

data: sigara and bölgeler

X-squared = 0.37454, df = 2, p-value = 0.8292

> kikaresatışlar

sigara

bölgeler evet hayır

marmara 7 9

karadeniz 11 10

güneydoğu 7 6

anova

ilk önce aov analizi yap 0,05> ise tukey değilse tamhane

TukeyHSD(aov(gelir~bölgeler))

Kruskal Wallis

> kruskal.test(gelir~bölgeler)

Kruskal-Wallis rank sum test

data: gelir by bölgeler

Kruskal-Wallis chi-squared = 6.2158, df = 2, p-value = 0.04469

> select(tablo,kişi,krediler,notlar)

Select komutu tabloyu sıraya göre seçer yani ilk sütun kişi 2. Krediler3. Notlar

Notları 1. Sütuna almak istersem ilk notları yazarım.

> match("burcu",tablo$kişi)

[1] 2

Kişilerde burcunun olupolmadığını denetledim.

Contains tabloda sütunlarda i harfi içerenleri alır.

> select(tablo,contains("i"))

kişi krediler cins

1 berk 2 kadın

2 burcu 2 kadın

3 yağmur 3 kadın

4 ali 3 kadın

5 sıla 2 erkek

6 harun 3 erkek

7 burcu 2 kadın

Notları başa alır geri kalanları yanına ekler.

> select(tablo,notlar,everything())

notlar kişi krediler GNO cins

1 20 berk 2 3.3 kadın

2 60 burcu 2 2.5 kadın

3 15 yağmur 3 3.8 kadın

4 85 ali 3 3.1 kadın

Tablodan hangileri burcudur.

> grep("burcu",tablo$kişi)

[1] 2 7

> tablo[grep("u",tablo$kişi),] kişilerde u gecenler

kişi notlar krediler GNO cins

2 burcu 60 2 2.5 kadın

3 yağmur 15 3 3.8 kadın

6 harun 70 3 3.5 erkek

7 burcu 65 2 25.0 kadın

> tablo[grep("ur",tablo$kişi),] kişilerde ur gecen

kişi notlar krediler GNO cins

2 burcu 60 2 2.5 kadın

3 yağmur 15 3 3.8 kadın

7 burcu 65 2 25.0 kadın

b ile başlayanlar

> filter(tablo,substr(tablo$kişi,1,1)=="b")

kişi notlar krediler GNO cins

1 berk 20 2 3.3 kadın

2 burcu 60 2 2.5 kadın

3 burcu 65 2 25.0 kadın

Ekstra bir sütun oluşturma

> mutate(tablo,frenot=notlar/sum(notlar))

kişi notlar krediler GNO cins frenot

1 berk 20 2 3.3 kadın 0.04914005

2 burcu 60 2 2.5 kadın 0.14742015

3 yağmur 15 3 3.8 kadın 0.03685504

4 ali 85 3 3.1 kadın 0.20884521

5 sıla 92 2 2.2 erkek 0.22604423

6 harun 70 3 3.5 erkek 0.17199017

7 burcu 65 2 25.0 kadın 0.15970516

Nonparametrik 2 grup testi wilcox

Non parametrik eş örneklem de wilcox

Erkeklerin not ve gno özeti

> summary(tablo[tablo$cins=="erkek",c(2,4)])

notlar GNO

Min. :55.0 Min. :2.170

1st Qu.:70.0 1st Qu.:2.200

Median :80.0 Median :3.050

Mean :76.4 Mean :2.804

3rd Qu.:85.0 3rd Qu.:3.100

Max. :92.0 Max. :3.500

A değişkenine notlar krediler gno aldım

a=data.frame(tablo$notlar,tablo$krediler,tablo$GNO)

hepsinin ort aldım

> apply(X=a,MARGIN = 2,FUN = mean,na.rm=T)

notlar krediler GNO

58.400 2.500 2.829

> a=tablo[,c("notlar","krediler","GNO")]

> apply(X=a, MARGIN=2,FUN = median)

notlar krediler GNO

62.500 2.500 2.775

Yuvarlanmış hali

> round(apply(X=a, MARGIN=2,FUN = median),2)

notlar krediler GNO

62.50 2.50 2.78

Gruplara göre gelir boxplotu

> ggplot(satışlar)+geom\_boxplot(aes(y=gelir,col=bölgeler))

> hist(Age) tek değişkenlidir yaşın frekansını verir

|  |
| --- |
| > boxplot(Age) tek değişkenlidir yasın boxplotunu verir  > tablo[grep("ha",tablo$kişi),]  kişi notlar krediler GNO cins  6 harun 70 3 3.50 erkek  8 hakan 55 3 2.17 erkek  11 harun 55 2 2.15 erkek  13 hasan 50 2 2.75 erkek |
| > tibble(tablo)  Tibble tablonun daha kolay okunmasını sağlıyor (görüntü olarak)  İçinde ar gecen veriler  > tablo[grep("ar",tablo$kişi),]  kişi notlar krediler GNO cins  6 harun 70 3 3.50 erkek  11 harun 55 2 2.15 erkek  End\_with, constains, start\_with gibi operatörler sadece  başlıklarda filtreleme yapar  tablo %>% group\_by(cins) %>% summarise(mean(notlar))  1 kadın 44.6  2 erkek 64.6  tablo %>% group\_by(cins) %>% summarise(length(notlar)) |
| |  | | --- | |  | |

Hangi sütunda kaç adet eksik veri var

> sapply(v,function(x) sum(is.na(x)))

d b

3 2

> xtabs(~bölgeler+sigara,data=satislar)

sigara

bölgeler evet hayır

marmara 7 9

karadeniz 11 10

güneydoğu 7 6

fiyatı 135000 tl olan evi 95000 yap

> ev\_fiyat[grep(135000,ev\_fiyat)]=95000

REGRESYON İLE AKCİĞER KAPASİTESİ TAHMİNLEME

> regresyon.sonuc=lm(LungCap~.,data = LungCapData2)

Yeni akciğer kapasitesi diye data frame oluşturuyoum başlıklar kesinlikle lungcapdata2 ile aynı olmalı büyük küçük harf duyarlı

> yeni\_lung (bunu yeni oluşturdum lungcap yok ben tahmin edicem)

Age Height Gender Smoke

1 6 60 male yes

2 7 52 male no

3 8 68 female no

> sonuc=predict(regresyon.sonuc,yeni\_lung)

Yeni akciğer değerlerim

> sonuc

1 2 3

4.302404 2.259885 7.929294

Tahmin diye yeni bir sütun oluşturdum ve yeni\_lung ile birleştirdim.

> yeni\_lung$tahmin=sonuc

> yeni\_lung

Age Height Gender Smoke tahmin

1 6 60 male yes 4.302404

2 7 52 male no 2.259885

3 8 68 female no 7.929294

YENİ TAHMİN

> yeni\_lung

Age Height Smoke Gender

1 8 55 no female

2 8 61 no female

3 6 63 no male

4 9 66 yes female

5 5 59 no female

6 7 66 yes male

7 6 54 yes male

> predict(regresyon.sonuc,yeni\_lung)

> yeni\_lung$tahmin=predict(regresyon.sonuc,yeni\_lung)

> yeni\_lung

Age Height Smoke Gender tahmin

1 8 55 no female 3.865516

2 8 61 no female 5.741106

3 6 63 no male 5.501938

4 9 66 yes female 7.238886

5 5 59 no female 4.526326

6 7 66 yes male 6.374522

7 6 54 yes male 2.426814

Coefficients katsayılar demek

Spss tablosu ile birebir değerler verir.

> regresyon.sonuc=lm(LungCap~.,data = LungCapData2)

> summary(regresyon.sonuc)

Call:

lm(formula = LungCap ~ ., data = LungCapData2)

Residuals:

Min 1Q Median 3Q Max

-4.1297 -0.7510 0.0268 0.7676 5.7614

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -15.37092 0.66852 -22.993 < 2e-16 \*\*\*

Age 0.19653 0.02847 6.904 1.21e-11 \*\*\*

Height 0.31260 0.01427 21.901 < 2e-16 \*\*\*

Genderfemale 0.47131 0.09962 4.731 2.74e-06 \*\*\*

Smokeyes -0.26174 0.17776 -1.472 0.141

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.237 on 649 degrees of freedom

Multiple R-squared: 0.7754, Adjusted R-squared: 0.774

F-statistic: 560 on 4 and 649 DF, p-value: < 2.2e-16

Kategorik değişkenleri numerik seç ve değerleri tanımla ör 1=kadın 2=erkek gibi

Spss’te sütunu seç dürbüne bas değerleri raplace et.

Spss’teki mann whitney u testi r da wilcoxon testidir.

Chocran q bağımlı örneklem t testi

Birbirine bağımlı 3 veya daha fazla bağımlı örneği incelememizi sağar 0 1 diye kodlanır.

Ör 3 farklı cipsi 15 kişiye sunuyoruz tercihlerini alıyoruz 1.beğendi 2. Beğenmedi 3. Beğendi gibi

Sonra bu cipslerin birbirlerine olan üstünlüklerini karşılaştırıyoruz. Non-parametrik testtir.

Hepsi cips

Tüketici 3 cipside deniyor 3’ünün hakkında yorum yapıyor.

McNemar bağımlı örneklem t testi 2’li

Aynı tüketici iyi tarım ve organik tarımı seçip seçmdeiğini yorumluyor örneğin 1.kişi iyi tarım evet organik tarım hayır

2.kişi iyi tarım hayır organik tarım hayır.

b=as.vector(b)

b=c()

for(i in 1:15){

if(tablo$notlar[i]>50 ){

b=c(b,"geçti")

}else {b=c(b,"kaldı")

}

}

print(b)

ki kare testi

> summary(xtabs(~cins+sonuc,data = tablo))

> table(tablo$sonuc)

geçti kaldı

9 6

> mean(filter(tablo,sonuc=="geçti",cins=="erkek")$notlar)

[1] 72.83333

Regresyon doğrulu grafik

> ggplot(mpg)+geom\_point(aes(x=displ,y=hwy,color=class))+geom\_smooth(data=mpg ,aes(x=displ,y=hwy))

> arrange(tablo,yaş) yaş kısıtına göre sıralama yap

| sembolü veya anlamına gelir notu 50 den buyuk yada yaşı 25 ten buyuk olan kişiler diye sıralanır.

> filter(tablo,notlar>50 | yaş>25 )

kişi Yaş cins krediler GNO notlar sonuc

1 burcu 24 kadın 2 2.50 60 geçti

2 ali 26 erkek 3 3.10 85 geçti

3 sıla 23 erkek 2 2.20 92 geçti

4 harun 30 erkek 3 3.50 70 geçti

5 burcu 20 kadın 2 2.50 65 geçti

6 hakan 20 erkek 3 2.17 55 geçti

7 berkan 19 erkek 2 3.05 80 geçti

8 harun 27 erkek 2 2.15 55 geçti

9 hasan 26 erkek 2 2.75 50 kaldı

10 hüseyin 27 erkek 3 2.85 30 kaldı

11 yasemin 18 kadın 2 2.89 75 geçti

Örneğin, iki saatten fazla gecikmeli (varışta veya kalkışta) olmayan uçuşlar bulmak istiyorsanız, aşağıdaki iki filtreden birini kullanabilirsiniz:

filter(flights, !(arr\_delay > 120 | dep\_delay > 120))

filter(flights, arr\_delay <= 120, dep\_delay <= 120)

df

*<chr>* tibble(df) vektörü tek sütüna çevirir.

1 harun

2 15

3 20

4 12

5 4

> arrange(tablo,desc(yaş)) büyükten küçüğe sıralar.

Tidyverse fonksiyonlarında her zaman ilk önce veri seti adı yazılır.

mutate () her zaman veri kümenizin sonuna yeni sütunlar ekler, böylece yeni değişkenleri görebilmemiz için daha dar bir veri kümesi oluşturarak başlayacağız.

Mutate tüm tabloyu gösterirken transmutate sadece filtrelediğin sütunları gösterir.

Yalnızca yeni değişkenleri korumak istiyorsanız, transmute () işlevini kullanın

Ortalamayı notlardan çıkarır yeni sütun oluşturur.

> mutate(tablo,ort\_fark = round(notlar-mean(notlar),2))

kişi Yaş cins krediler GNO notlar sonuc ort\_fark

1 berk 21 kadın 2 3.30 20 kaldı -35.27

2 burcu 24 kadın 2 2.50 60 geçti 4.73

3 yağmur 22 kadın 3 3.80 15 kaldı -40.27

4 ali 26 erkek 3 3.10 85 geçti 29.73

5 sıla 23 erkek 2 2.20 92 geçti 36.73

Transmute sadece o sütunu verir.

> transmute(tablo,ort\_fark = round(notlar-mean(notlar),2))

ort\_fark

1 -35.27

2 4.73

3 -40.27

4 29.73

5 36.73

6 14.73

Sayıların küçükten büyüğe sırasını gösteriyor.

> min\_rank(tablo$notlar)

[1] 2 9 1 14 15 11 10 7 13 5 7 4 6 3 12

> tablo$notlar

[1] 20 60 15 85 92 70 65 55 80 42 55 35 50 30 75

Ls() r da bulunan değişkenleri tabloları gösterir.

R’da tarihleri yıl ay gün diye yazarız

> v=as.Date("2020/04/11")

> v

[1] "2020-04-11"

> class(v)

[1] "Date"

> today=as.Date('2020/04/5')

> weekdays(today)

[1] "Pazar"

> x

[1] 10 11 12 13

> range(x) range-> aralık verir.

[1] 10 13

> range(x)[2]

[1] 13

Rastgele arada (sample)

sample(500:1000,5)

[1] 770 676 916 583 946

Benzersiz değerleri getir.

> unique(v)

[1] "harun" "bakırcı" "istanbul" "samsun"

**#######Çok önemli######**

**Aggerate fonksiyonlarıyla group by kullanımı.**

> aggregate(tablo$notlar, by=list(Category=tablo$kişi), FUN=sum)

Category x

1 ali 85

2 berk 20

3 berkan 80

4 burcu 125

5 hakan 55

6 harun 125

7 hasan 50

8 hüseyin 30

9 pelin 42

10 sıla 92

11 yağmur 50

12 yasemin 75

for(i in 1:NROW(tablo)){

if(tablo$not[i]>=mean(tablo$not)){

sonuc=c(sonuc,"geçti")

}

else {

sonuc=c(sonuc,"kaldı")

}

}

tablo%>%group\_by(kişi)%>%summarise(ort\_not=sum(not))

not adedini sayar.

tablo%>%group\_by(kişi)%>%summarise(say\_not=n())

> tablo%>%group\_by(kişi)%>%summarise(mak=max(not))

> filter(tablo%>%group\_by(kişi)%>%summarise(mak=max(not)),mak>30)

# A tibble: 10 x 2

kişi mak

*<chr>* *<dbl>*

1 ali 85

2 berkan 80

3 burcu 65

4 hakan 55

5 harun 90

6 hasan 50

7 pelin 42

8 sıla 92

9 yağmur 60

10 yasemin 75

Tablom bir ki kare tablosudur.

Margin.table(tablom,2) tablonun sütunlarının toplanmasını sağla

> margin.table(tablom,2)

beğen

E H

5 3

Tablom bir ki kare tablosudur.

Margin.table(tablom,1) tablonun satırlarının toplanmasını sağla

> margin.table(tablom,1)

> tablom

beğen

cins EVET HAYIR

ERKEK 3 2

KADIN 2 1

> margin.table(tablom,1)

cins

ERKEK KADIN

5 3

> margin.table(tablom,2)

beğen

EVET HAYIR

5 3

> summarise(tablo,ort=mean(not))

ort

1 59.4

> summarise(filter(tablo,kişi=="harun"),satış\_sayısı=n())

satış\_sayısı

1 4

For(i in 10){

İf(a>10){B=”geçti”}

Else{b=”kaldı”}

}

gapminder %>%

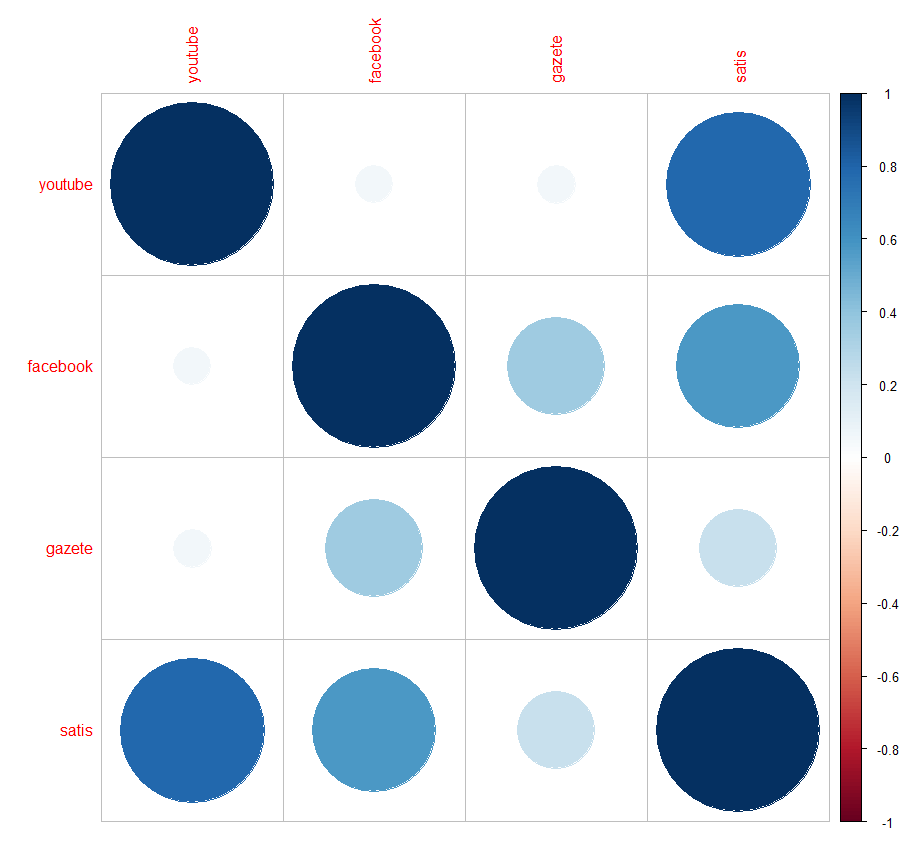
filter(year == 2007) %>%

mutate(lifeExpMonths = 12 \* lifeExp) %>%

arrange(desc(lifeExpMonths))

lineer regresyonda bağımlı değişken sayısal bağımsız değişken ise sürekli ya da kategorik olabilir.

Sonuç yani bağımlı değişken kategorik ise lojistik regresyon yapılır.



kor=cor(marketing)

> library(corrplot)

> corrplot(kor,"circle")

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 11.17397 0.67548 16.542 <2e-16 \*\*\*

facebook\_reklam 0.20250 0.02041 9.921 <2e-16 \*\*\*

estimate katsayılar demek

intercept sabit

burada satis ve facebook u lm yaptım

aslında satis=11.17+0.2025\*facebook

eğer facebook a hiç reklam vermezsem satış miktarım 11.17 olur

eğer facebook a 1000 dolarlık bir reklam verirsem satış miktarım 202 artar.

2.2e-16 demek 2\*10 üzeri -16 demek yani çok küçük.

İlk önce F istatistiğin p değerine bakılır yani testin p değeri

Eğer F istatistiğinin p değeri 0,05 ten küçük ise test anlamlıdır.

Call:

lm(formula = satis\_sayisi ~ youtube\_reklam + facebook\_reklam +

gazete, data = marketing)

Coefficients:

(Intercept) youtube\_reklam facebook\_reklam gazete

3.526667 0.045765 0.188530 -0.001037

İntercept yani katsayılar bağımsız değişkende meydana gelen 1 birimlik artış bağımlı yani sonuç değişkenimi şu kadar etkiler.

Yani youtube a verilen 1000 dolarlık reklamın 46 adet satışa sebep olacağı açıklanır.

Gazete değişkeninin p değerinin anlamsız olduğunu gördük bu yüzden gazete değişkenini modelden çıkarıp modeli yeniden kuruyoruz.

Coefficients:

(Intercept) youtube\_reklam facebook\_reklam gazete

3.526667 0.045765 0.188530 -0.001037

Aslında satis= 3.526 + youtube\*0.0457 + facebook\*0.188

Residual standard error: 2.018 on 197 degrees of freedom

Multiple R-squared: 0.8972, Adjusted R-squared: 0.8962

F-statistic: 859.6 on 2 and 197 DF, p-value: < 2.2e-16

Residual std error artık standart hata 2.018 demek

Satis değişkenim en fazla 2.018 sapar.

Her bir satışta 0.119 sapma olur

> 2.018/mean(marketing$satis\_sayisi)

[1] 0.1199263

F istatistiği 0.05’den küçük ise modelde en az bir değikenin sonuc değişkenini etkilediği söylenir.

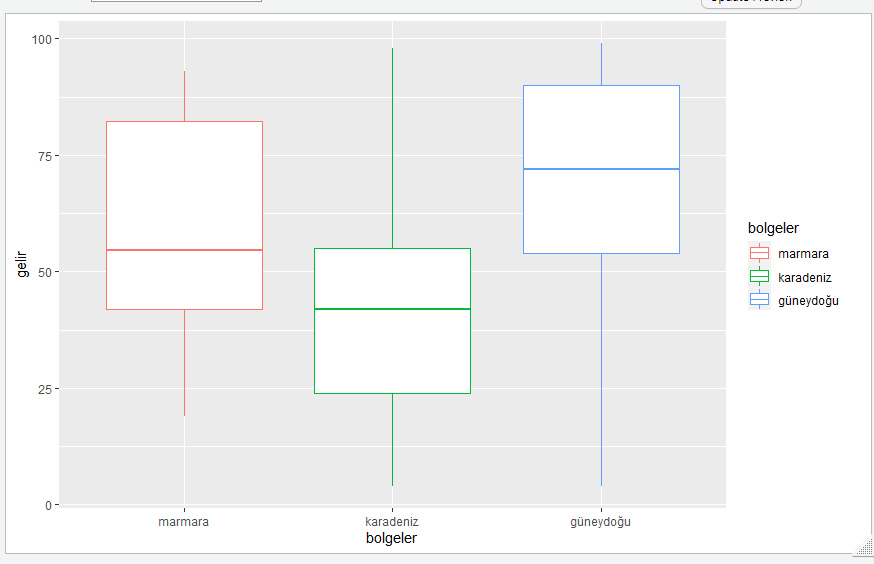
Na.rm=T na’ları kaldır.

Doğrusal regresyon lm()

Lojistik kategorik regresyon ise glm()

glm(formula = Class ~ Cell.shape, family ="binomial", data = bc)

> ggplot(satislar)+geom\_boxplot(aes(x=bolgeler,y=gelir,color=bolgeler))



> floor(40/6) aşağı yuvarlar.

[1] 6

> 50%%6

[1] 2

> ggplot(LungCapData2)+geom\_point(aes(x=Age,y=LungCap,color=Smoke,size=5),position = "jitter")+xlab("Yaş")+ylab("Akciğer Kapasitesi")

**> symptoms <- factor(c("SEVERE", "MILD", "MODERATE"),**

**levels = c("MILD", "MODERATE", "SEVERE"),**

**ordered = TRUE)**

The resulting symptoms factor now includes information about the order we

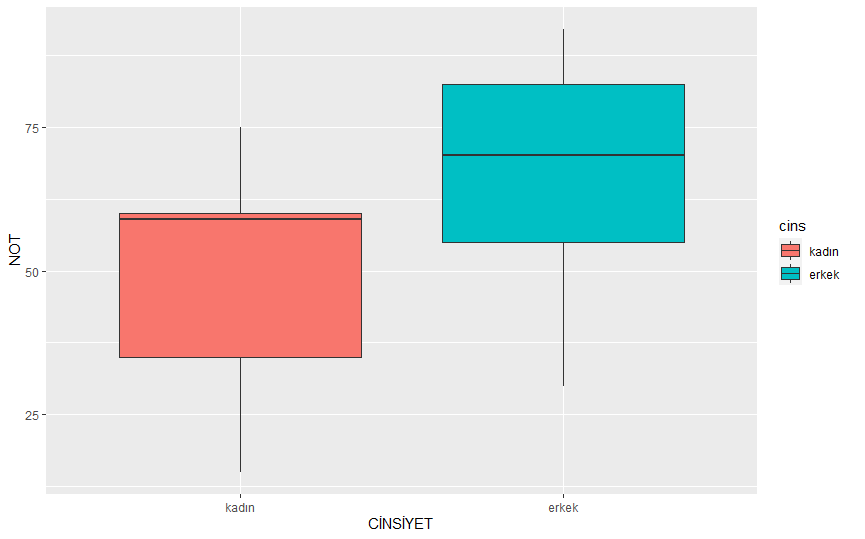
requested. Unlike our prior factors, the levels value of this factor are separated

by < symbols, to indicate the presence of a sequential order from mild to severe:

**> symptoms**

**[1] SEVERE MILD MODERATE**

**Levels: MILD < MODERATE < SEVERE**

> ggplot(tablo)+geom\_boxplot(aes(x=cins,y=not,fill=cins))+xlab("CİNSİYET")+ylab("NOT")

> tablo%>%group\_by(cins)%>%summarise(ort\_not=mean(not))

# A tibble: 2 x 2

cins ort\_not

*<fct>* *<dbl>*

1 kadın 47.9

2 erkek 68.8

> satislar%>%group\_by(bolgeler)%>%summarise(ort\_gelir=mean(gelir))

# A tibble: 3 x 2

bolgeler ort\_gelir

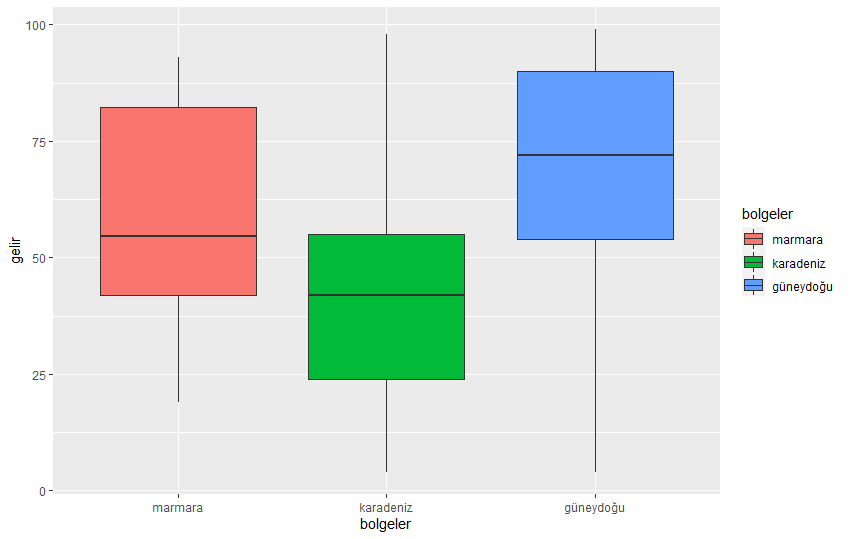
*<fct>* *<dbl>*

1 marmara 59.1

2 karadeniz 41.9

3 güneydoğu 65.2

> ggplot(satislar)+geom\_boxplot(aes(x=bolgeler,y=gelir,fill=bolgeler))



> satislar%>%group\_by(bolgeler)%>%summarise(satis\_sayisi=n())

# A tibble: 3 x 2

bolgeler satis\_sayisi

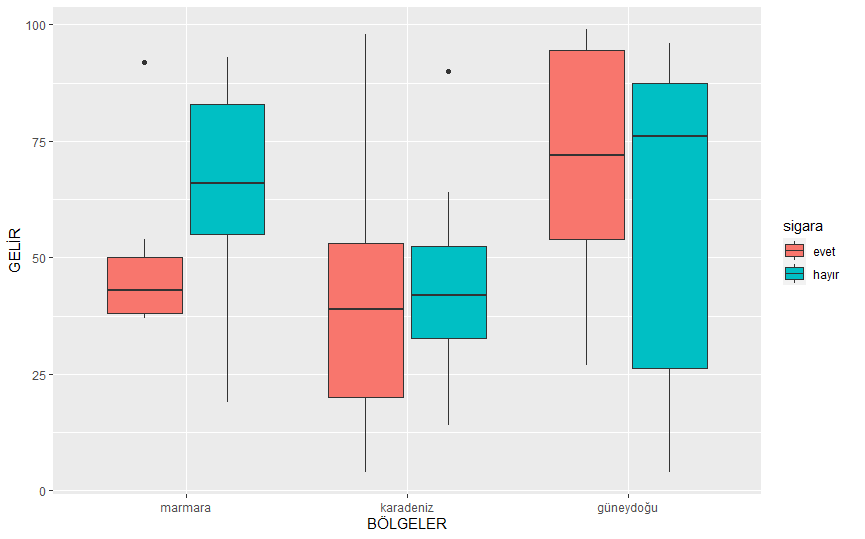
*<fct>* *<int>*

1 marmara 16

2 karadeniz 21

3 güneydoğu 13

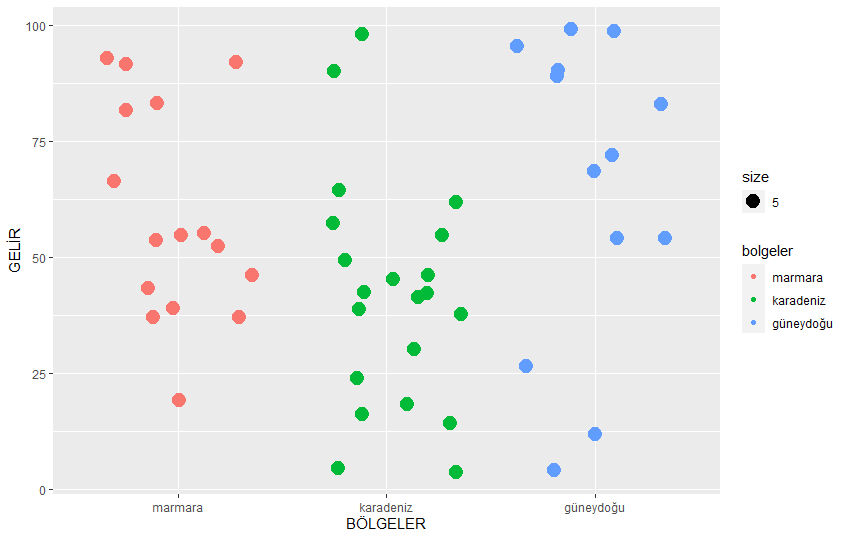
> ggplot(satislar)+geom\_boxplot(aes(x=bolgeler,y=gelir,fill=sigara))+xlab("BÖLGELER")+ylab("GELİR")



> ggplot(satislar)+geom\_boxplot(aes(x=bolgeler,y=gelir,fill=sigara))+xlab("BÖLGELER")+ylab("GELİR")+theme\_classic()

Theme\_classic() temayı değiştirir.

> ggplot(satislar)+geom\_point(aes(x=bolgeler,y=gelir,color=bolgeler,size=5,position = "jitter")+xlab("BÖLGELER")+ylab("GELİR")



for (i in 1:6){

a=readline("sayı girin:")

a=as.numeric(a)

if(a>5){

print("a 5'ten büyüktür")

}

else if (a<=5){

print("a 5'e eşit veya büyüktür")

}

}

a=readline("kişi giriniz:")

for (i in grep(a,tablo$kişi)){

print(paste(a,i,".","sıradadır"))

}

isim=function(a,kisi){

a=as.integer(a)

for(i in 1:a){

print(kisi)

}

}

bakiye=1000

while (islem<5) {

islem=readline("para çekmek için 1'e

para yatırmak için 2'ye

para göndermek için 3'e basın:")

islem=as.numeric(islem)

if (islem==1){

miktar=readline("miktar giriniz:")

miktar=as.numeric(miktar)

if(miktar<bakiye){

bakiye=bakiye-miktar

print(paste(bakiye,"TL'dir"))

}

}

}

LOJİSTİK REGRESYON

glm(Gender~.,family = binomial,data = LungCapData2)

ilk önce library(caret) çağır.

McNemar Testi DOGRUU

> mcnemar.test(mcnemar$oncesi,mcnemar$sonrası)

McNemar's Chi-squared test with continuity correction

data: mcnemar$oncesi and mcnemar$sonrası

McNemar's chi-squared = 0.47093, df = 1, p-value = 0.4926

> cor.test(LungCapData2$LungCap,LungCapData2$Age,method = "pearson")

Pearson's product-moment correlation

data: LungCapData2$LungCap and LungCapData2$Age

t = 29.533, df = 652, p-value < 2.2e-16

alternative hypothesis: true correlation is not equal to 0

95 percent confidence interval:

0.7216444 0.7874568

sample estimates:

cor

0.756459

> cor.test(LungCapData2$LungCap,LungCapData2$Age,method = "spearman")

Spearman's rank correlation rho

data: LungCapData2$LungCap and LungCapData2$Age

S = 9397713, p-value < 2.2e-16

alternative hypothesis: true rho is not equal to 0

sample estimates:

rho

0.7984229

MANN WHİTNEY U TEST

> wilcox.test(Age~Gender,data = LungCapData2)

Wilcoxon rank sum test with continuity correction

data: Age by Gender

W = 51900, p-value = 0.5256

alternative hypothesis: true location shift is not equal to 0

ONE SAMPLE T TEST

> t.test(LungCapData2$Age,mu=10)

One Sample t-test

data: LungCapData2$Age

t = -0.59569, df = 653, p-value = 0.5516

alternative hypothesis: true mean is not equal to 10

95 percent confidence interval:

9.704381 10.158005

sample estimates:

mean of x

9.931193

> wilcox.test(LungCapData2$Age,LungCapData2$LungCap,paired = T)

Wilcoxon signed rank test with continuity correction

data: LungCapData2$Age and LungCapData2$LungCap

V = 213988, p-value < 2.2e-16

alternative hypothesis: true location shift is not equal to 0

apply fonksiyonu margın=2 ise sütunlarda işlem yapar

MARGIN = 1 ise satırlarda işlem yapar.

xveri=c("c","ab","ac","bc","ab","abc","abc")

xveri[grep("a",xveri)] içinde “a” harfi olanlar

tekrar eden verileri duplicated() fonk. İle bulabilirim.

> xveri[duplicated(xveri)]

[1] "ab" "abc"

> paste("mehmet","akif")

[1] "mehmet akif"

> paste("mehmet","akif",sep = "-")

[1] "mehmet-akif"

> paste("mehmet","akif",sep = "")

[1] "mehmetakif"

strsplit("harun-mehmet-kasım-cenk-hasan-ismail-hüseyin",split = "-")

[[1]]

[1] "harun" "mehmet" "kasım" "cenk" "hasan" "ismail" "hüseyin"

“hir” parçasının hangi isimlerde geçtiğini verir.

> grep("hir",c("nehir","şehir","harun","kasım")

+ )

[1] 1 2

> regexpr("hir","nehir") 3 ten başlar 3 karakter son

[1] 3 sonra biter

attr(,"match.length")

[1] 3

> gregexpr("ur","burdur")

[[1]]

[1] 2 5

attr(,"match.length")

[1] 2 2

> gregexpr("ur","burdursur")

[[1]]

[1] 2 5 8

attr(,"match.length")

[1] 2 2 2

> filter(tablo,kişi=="harun" | kişi=="burcu")

kişi Yaş cins krediler GNO not sonuc

1 burcu 24 kadın 2 2.50 60 geçti

2 harun 30 erkek 3 3.50 70 geçti

3 burcu 20 kadın 2 2.50 65 geçti

4 harun 27 erkek 2 2.15 55 kaldı

5 harun 24 erkek 4 2.25 80 geçti

6 burcu 25 kadın 5 2.65 59 kaldı

7 harun 29 erkek 5 3.70 90 geçti

H 2.harf önemli değil sonraki harf s olsun.

> tablo$kişi[grep("h.s",tablo$kişi)]

[1] "hasan" "hüseyin"

> grep("r.a",c("ankara","bursa","ardahan"))

[1] 2 3

R da tarihler yıl-ay-gun olarak girilir.

As.date() olarak tanımla.

> aile1

[1] "1995-02-14" "1991-08-16"

> aile2

[1] "1998-05-12" "1993-11-13"

> difftime(aile1,aile2)

Time differences in days

[1] -1183 -820

İki tarih arasındaki gun farkını verir.

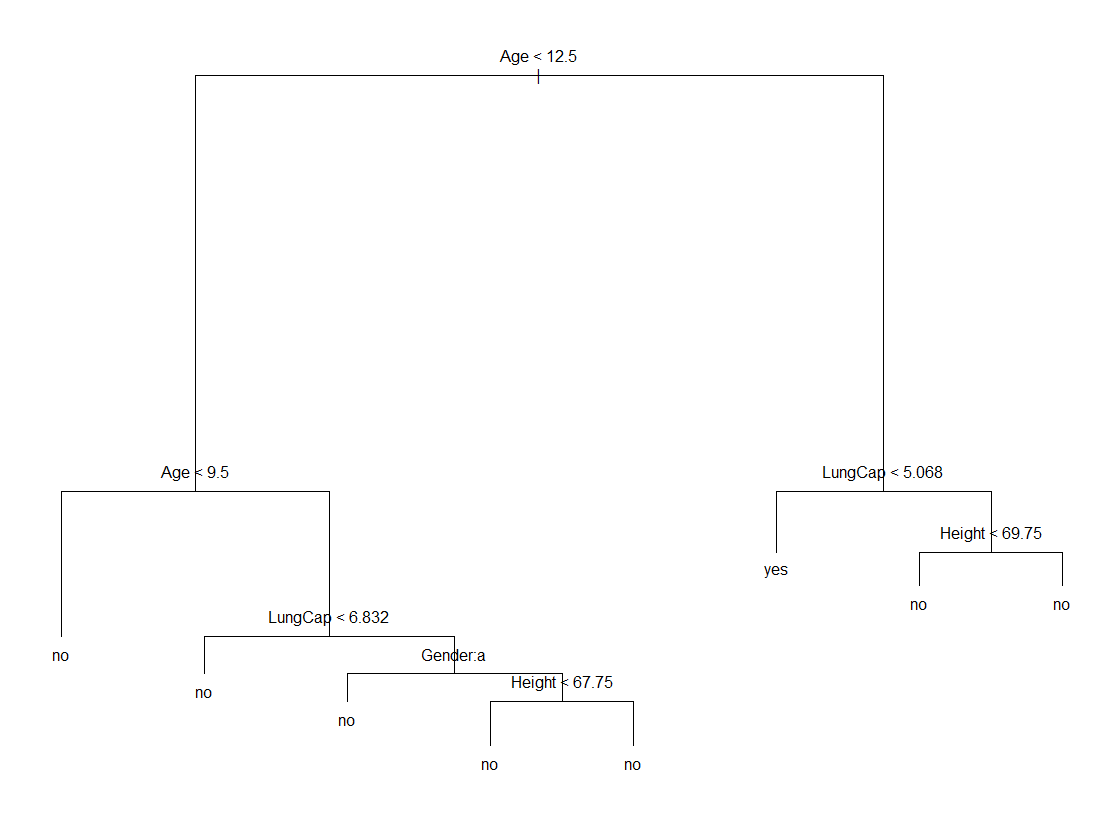
> b=ifelse(a>=27.5,"üstünde veya eşit","altında")

KARAR AĞACI DESİCİON TREE

> tree(Smoke~.,data = LungCapData2)

> plot(tree(Smoke~.,data = LungCapData2))

> text(tree(Smoke~.,data = LungCapData2))



> tree(vites~.,data = arabaveri)

> plot(tree(vites~.,data = arabaveri))

> text(tree(vites~.,data = arabaveri))

> filter(arabaveri,ağırlık>3,18,yol<15,1)

yol silindir ağırlık vites

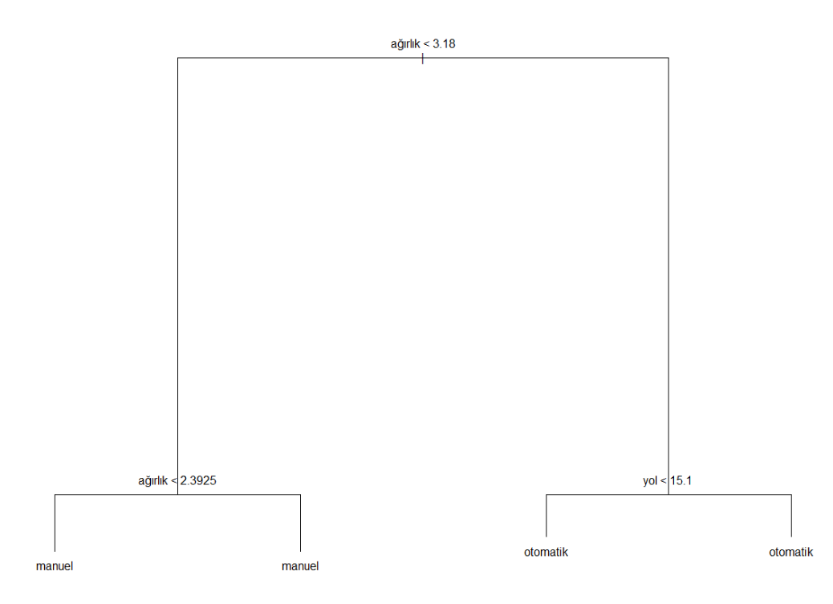
1 14.3 8 3.570 otomatik

2 10.4 8 5.250 otomatik

3 10.4 8 5.424 otomatik

4 14.7 8 5.345 otomatik

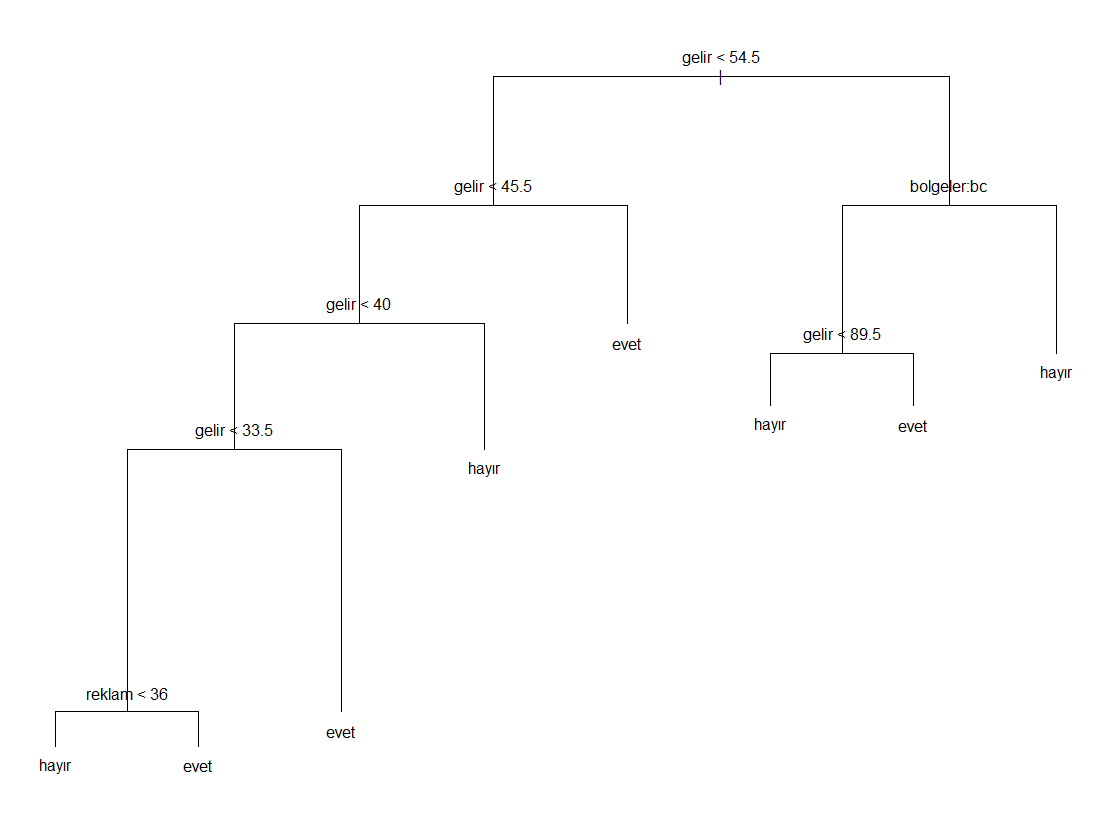
5 13.3 8 3.840 otomatik



> tree(sigara~.,data = satislar)

> plot(tree(sigara~.,data = satislar))

> text(tree(sigara~.,data = satislar))



Karar ağacı tahminleme

Tahmin etmeden önce library(class)

> sonuc=knn(cbind(veri$boy,veri$kilo),c(185,65),veri$cinsiyet)

> sonuc

[1] k

Levels: e k

K=3 dedim en yakın 3 kişiyi kontrol et dedim

K=3 demeseydim k=1 olarak alırdı.

> sonuc=knn(cbind(veri$boy,veri$kilo),c(165,65),veri$cinsiyet,k=3)

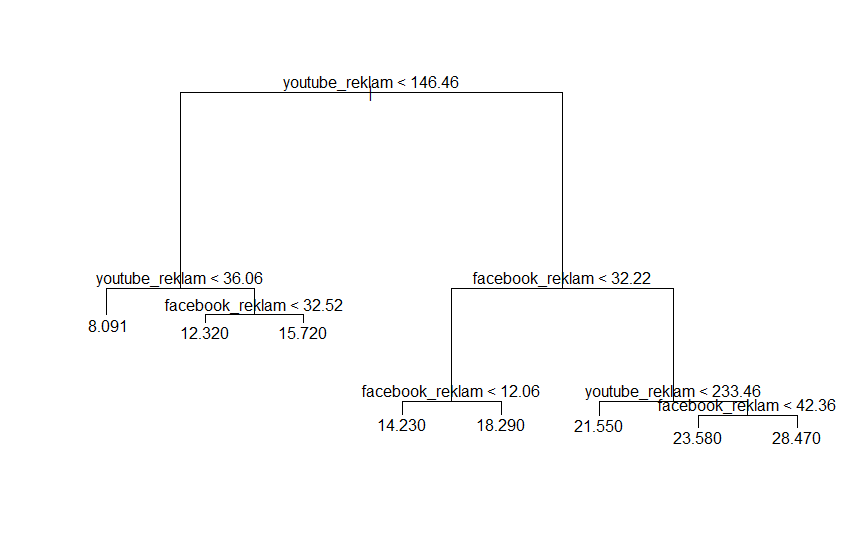
> sonuc

[1] e

Levels: e k

> plot(tree(satis\_sayisi~.,data = marketing))

> text(tree(satis\_sayisi~.,data = marketing))



sonuc=knn(cbind(LungCapData2$Age,LungCapData2$LungCap,LungCapData2$Height

),c(8,3.5,59),LungCapData2$Smoke,k=3)

> sonuc

[1] no

Levels: no yes

> sonuc1=knn(cbind(LungCapData2$Age,LungCapData2$LungCap,LungCapData2$Height),a[,c(1:3)],LungCapData2$Smoke,k=3)

> a$sigara\_sonuc=sonuc1

> a

Age LungCap Height Gender sigara\_sonuc

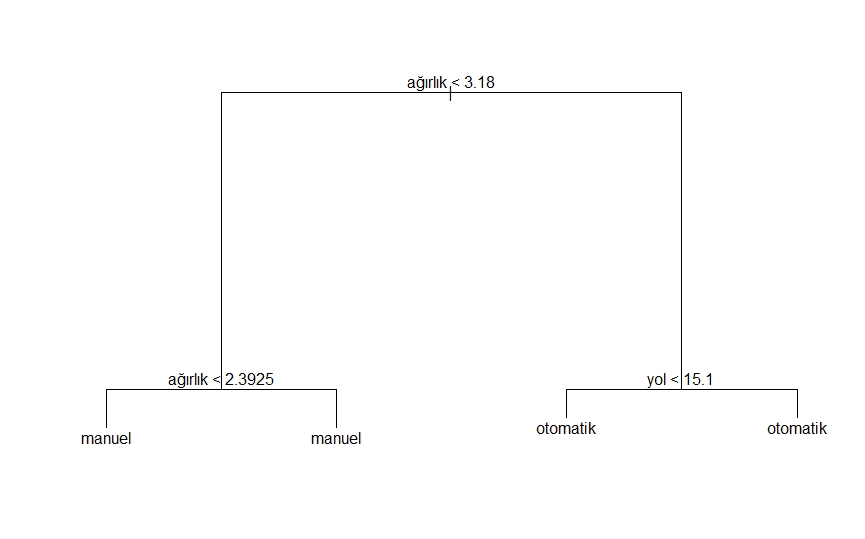
1 8 4.01 55 male no

2 9 3.88 60 female no

**REGRESYON VE KARAR AĞACI İLİŞKİSİ**

> plot(tree(vites~.,data = arabaveri))

> text(tree(vites~.,data = arabaveri))



Call:

glm(formula = vites ~ ., family = binomial, data = arabaveri)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.91154 -0.29298 -0.03574 0.21541 1.96021

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 23.92836 14.17738 1.688 0.0915 .

yol -0.09851 0.35135 -0.280 0.7792

silindir6 3.00979 2.51067 1.199 0.2306

silindir8 4.98194 3.50934 1.420 0.1557

ağırlık -8.17801 3.34965 -2.441 0.0146 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 43.230 on 31 degrees of freedom

Residual deviance: 14.588 on 27 degrees of freedom

AIC: 24.588

Number of Fisher Scoring iterations: 7

Binary lojistik regresyon tahmini

predict(model, newdata, type="response")

0.2361081

Tahmin edilen olasılık 0.24'tür.

LİNEER REGRESYON TAHMİNLEMESİ

Yeni oluşturacağım data da subject olmayacak ve kuracağım lm() modelinde de subject olmayacak

Yeni veri setimde önceden gender cinsiyet değişkenini as.factor() olarak tanımlayacağım sonra level lerini gireceğim

Modeli şimdi kurabilirim

yeni\_data=data.frame(Subject=c("harun","mert"),Age=c(32,45),Gender=c("male","male"))

lm() modelinde subject i çıkarıyorum

yeni veri setimde ki kategorik değişkenleri belirliyorum

> yeni\_data$Gender=as.factor(yeni\_data$Gender)

> levels(yeni\_data$Gender)=c("male","female")

> predict(lm(Score~Age+Gender,data = DataToExport),yeni\_data[,-1])

1 2

69.56765 65.58938

| **Subject** | | **Age** | **Gender** | | **Score** | |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | |  |  | |  |
| **1** | Dave.Andreychuk | | 53 | male | | 80.5 |
| **2** | Jon.Stewart | | 54 | male | | 82.1 |
| **3** | Jane.Doe | | 38 | female | | 75.9 |

R square si %78 olan lungcapdata2’m

Age LungCap Height Gender Smoke

1 9 3.124 57.0 male no

2 8 3.172 67.5 male no

3 7 3.160 54.5 male no

4 9 2.674 53.0 female no

Lungcap akciğer kapasitesini tahmin edeceğim datam

> yeni\_data

Age Height Gender Smoke lungcap yok tahmin edicem

1 9 55 male yes

2 10 62 male yes

3 11 58 female no

Faktörleri tanımladım

> yeni\_data$Smoke=as.factor(yeni\_data$Smoke)

> levels(yeni\_data$Smoke)=c("yes","no")

> yeni\_data$Gender=as.factor(yeni\_data$Gender)

> levels(yeni\_data$Gender)=c("male","female")

Yeni lungcap’lerimi tahmin ettim

> lung\_sonuc=predict(lm(LungCap~.,data = LungCapData2),yeni\_data)

> lung\_sonuc

1 2 3

4.062044 6.446760 4.659847

> yeni\_data$lung\_tahmin=lung\_sonuc

> yeni\_data

Age Height Gender Smoke lung\_tahmin

1 9 55 female no 4.062044

2 10 62 female no 6.446760

3 11 58 male yes 4.659847

BİTTİ.

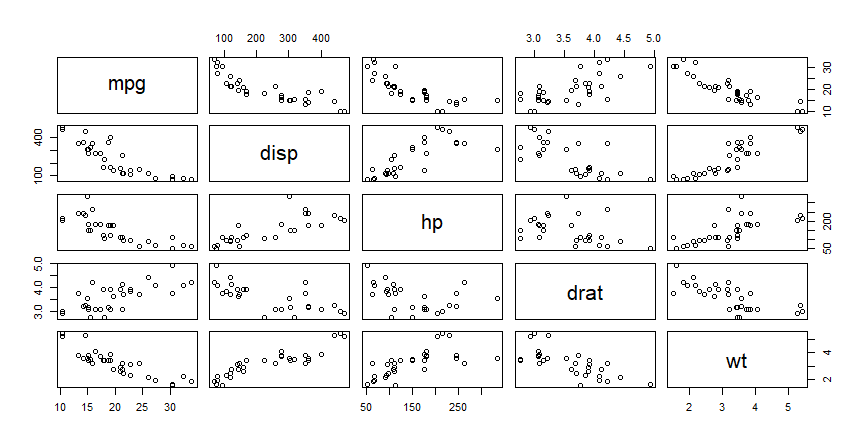
Birden fazla değişkenin normallik testi

shapiro.test(c(LungCapData2$LungCap,LungCapData2$Age))

lungcap datasından 10 örnek al

> sample\_n(LungCapData2,10)

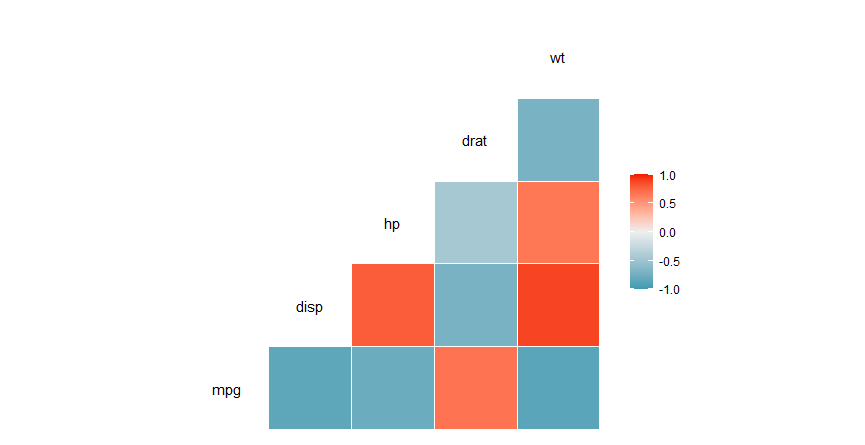
> plot(df)



Daha iyi bir görsel için

> library(GGally) df mtcars datasının bir alt kümesi

> ggcorr(df)

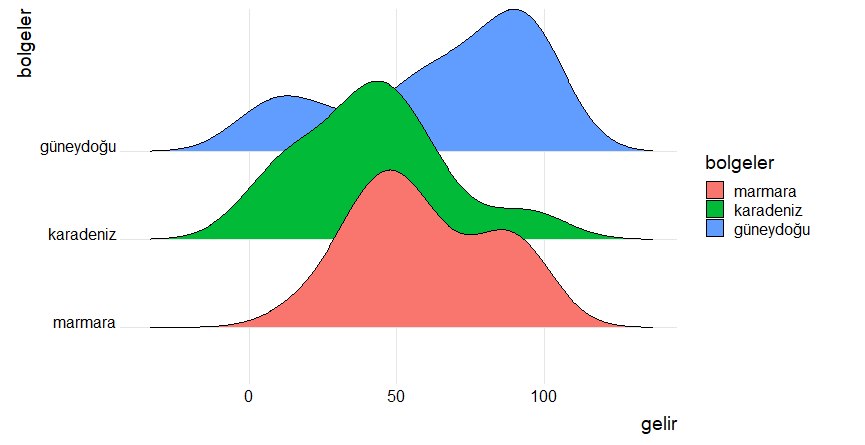


> library(ggridges)

ggplot(satislar,aes(x=gelir,y=bolgeler,fill=bolgeler)) +

geom\_density\_ridges()+

theme\_ridges()



ggplot(iris,aes(Sepal.Length,Sepal.Width))+

geom\_point(size=3,

shape=21,

stroke=2,

color="black",

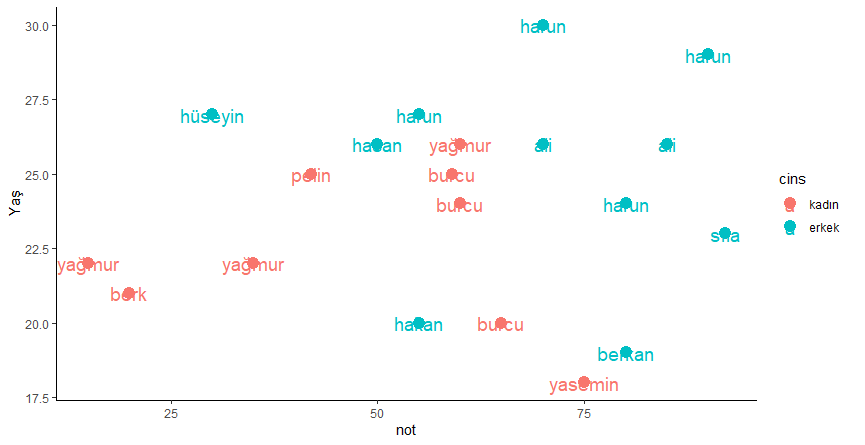
fill="orange")+

geom\_smooth(method = lm,color="red")

ggplot(tablo,aes(x=not,y=Yaş,color=cins))+

geom\_point(size=4)+

geom\_text(label=tablo$kişi,size=5)



Geom\_label() geom\_text()’teki isimleri kutucuk içerisine alır.

knn(cbind(LungCapData2$Age,LungCapData2$Height,LungCapData2$Gender

),c(8,60,factor("male")),LungCapData2$Smoke,k=3)

SORU (\*\*\*SOR ÖNEMLİİİ\*\*\*)

Eğer değişken anlamlı değilse oluşturulan karar ağacında yer almıyor.

Eksik veri bulma

sapply(LungCapData2, function(x) sum(is.na(x)))

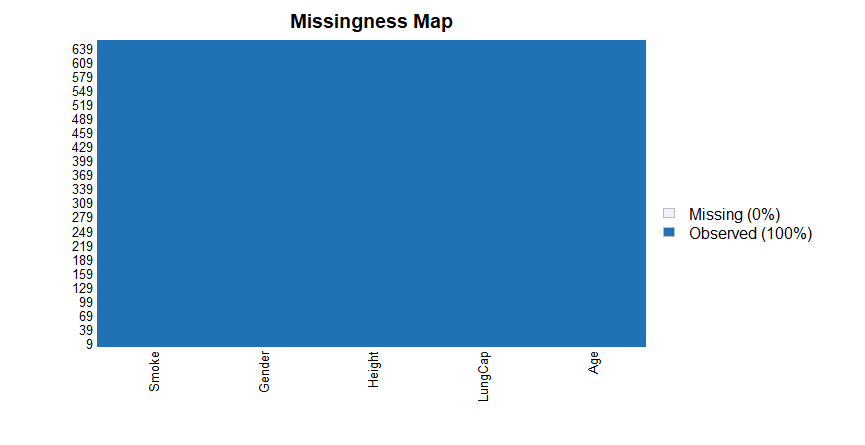
Age LungCap Height Gender Smoke

0 0 0 0 0

Eksik veri bulma

Library(Amelia)

missmap(LungCapData2)



Hangi satırlarda eksik veri olduğunu tespit ettikten sonra

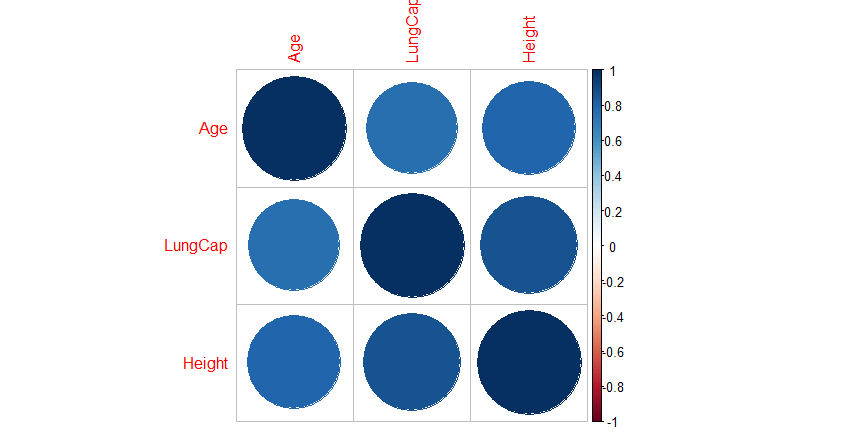
Veri= veri[!(is.na(veri$ca veya operatörü is.na(veri$tha1)),]

Çalıştırırsak eksik verilerimiz modelden atılmış olur.

library(corrplot)

cor(LungCapData2[,c(1,2,3)])

corrplot(cor(LungCapData2[,c(1,2,3)]),method = "circle")



MAKİNE ÖĞRENMESİ

Makine öğrenmesi algoritmaları uygulanmadan önce veri seti uygulama ve test diye ikiye ayrılır %20 %80 oranında ayırabiliriz.

Eğitim verisiyle model kurulur.

Test verisiyle model test edilir.

Library(caret) kütüphanesiyle veriyi böleriz.

Rastgele seçilen değerleri sabitlemek için set.seed(100)

Fonksiyonu kullanılır.

Veriyi bölmek için createDataPartition

Eğitim\_indis= createDataPartition(y=veri$kh,p=0.80,list=False)

Eğitim\_veri=veri[eğitim\_indis,]

Test\_veri=veri[-eğitim\_indis,]

library(caret)

set.seed(100)

egitim\_indis=createDataPartition(y=LungCapData2$Smoke,p=0.80,list = FALSE)

egitim\_veri=LungCapData2[c(egitim\_indis),]

test\_veri=LungCapData2[-egitim\_indis,]

model=glm(Smoke~.,family = binomial,data = egitim\_veri)

summary(model)

lr\_olasılık=predict(model,newdata = test\_veri,

type="response")

head(lr\_olasılık)

lr\_tahmin=ifelse(lr\_olasılık>0.5,"yes","no")

mean(lr\_tahmin==test\_veri$Smoke)

#### mean(lr\_tahmin==test\_veri$Smoke) sonucu 0.9076

#yani yeni veriyi tahmin etme olasılığı %91

İlk önce eğitim verisinin tahmin olasılığını buluyoruz.

Sonra test verisinin tahmin olasılığını buluyoruz.

Karşılaştırıyoruz yakın ise test güvenilirdir.

EĞİTİM VERİSİ

\*\*\*Eğitim verisi ile test verisini tahmin et.

library(caret)

library(tidyverse)

head(LungCapData2)

str(LungCapData2)

sapply(LungCapData2,function(x) sum(is.na(LungCapData2)))

set.seed(100)

egitim\_indis=createDataPartition(y=LungCapData2$Smoke,p=0.8,list = FALSE)

egitim\_veri=LungCapData2[c(egitim\_indis),]

test\_veri=LungCapData2[-egitim\_indis,]

attach(LungCapData2)

model\_egitim=glm(Smoke~.,family = binomial,data = egitim\_veri)

summary(model\_egitim)

test\_tahmin=predict(model\_egitim,test\_veri,type = "response")

head(test\_tahmin)

test\_sonuc=ifelse(test\_tahmin>0.5,"yes","no")

head(test\_sonuc)

summary(test\_sonuc)

class(test\_sonuc)

test\_sonuc=as.factor(test\_sonuc)

levels(test\_sonuc)

mean(test\_veri$Smoke==test\_sonuc)

> mean(test\_veri$Smoke==test\_sonuc)

[1] 0.9076923

TEST VERİSİ

\*\*\*Test verisi ile eğitimi tahmin et

model\_test=glm(Smoke~.,family = binomial,data = test\_veri)

summary(model\_test)

egitim\_tahmin=predict(model\_test,egitim\_veri,type = "response")

head(egitim\_tahmin)

egitim\_sonuc=ifelse(egitim\_tahmin>0.5,"yes","no")

head(egitim\_sonuc)

summary(egitim\_sonuc)

class(egitim\_sonuc)

egitim\_sonuc=as.factor(egitim\_sonuc)

levels(egitim\_sonuc)

mean(egitim\_veri$Smoke==egitim\_sonuc)

> mean(egitim\_veri$Smoke==egitim\_sonuc)

[1] 0.8854962

knn(cbind(LungCapData2$Age,LungCapData2$LungCap,

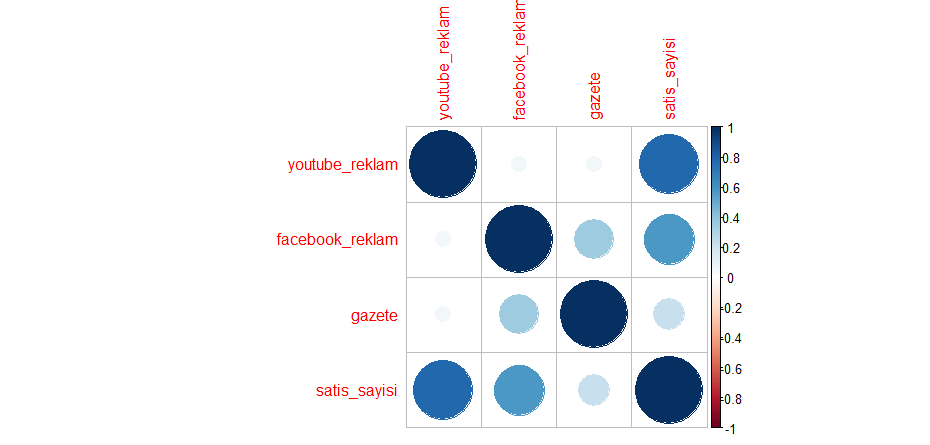
LungCapData2$Height,factor(LungCapData2$Gender)),

c(9,3.48,65,factor("male")),LungCapData2$Smoke,k=3)

filter(LungCapData2,Age<=10,Gender=="male",Height<=65,Height>60,Age>8)

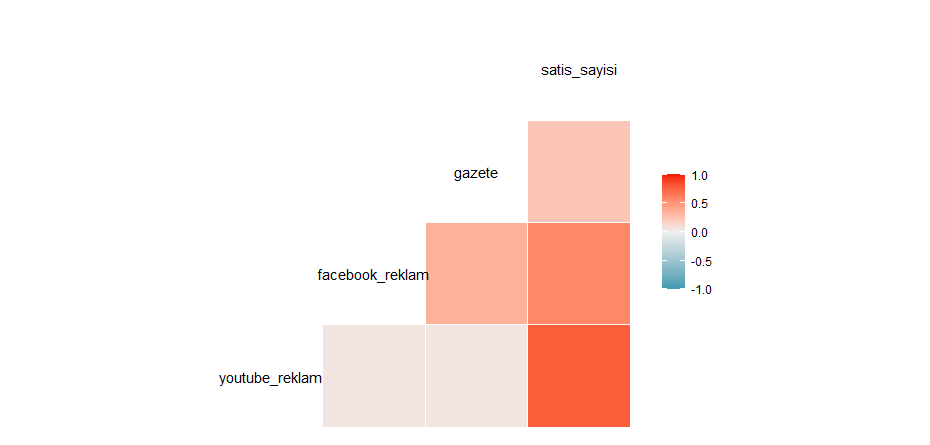
library(corrplot)

corrplot(cor(marketing))



library(GGally)

ggcorr(marketing)



range(tam.veri$Survived,na.rm = T)

içerisinde “b” harfi geçen kişiler

> tablo[grep("b",tablo$kişi),]

kişi Yaş cins krediler GNO not sonuc

1 berk 21 kadın 2 3.30 20 kaldı

2 burcu 24 kadın 2 2.50 60 geçti

7 burcu 20 kadın 2 2.50 65 geçti

9 berkan 19 erkek 2 3.05 80 geçti

17 burcu 25 kadın 5 2.65 59 kaldı

18 abdullah 26 erkek 4 3.10 70 geçti

> tablo[grep("i",tablo$kişi),]

kişi Yaş cins krediler GNO not sonuc

4 ali 26 erkek 3 3.10 85 geçti

10 pelin 25 kadın 3 2.17 42 kaldı

14 hüseyin 27 erkek 3 2.85 30 kaldı

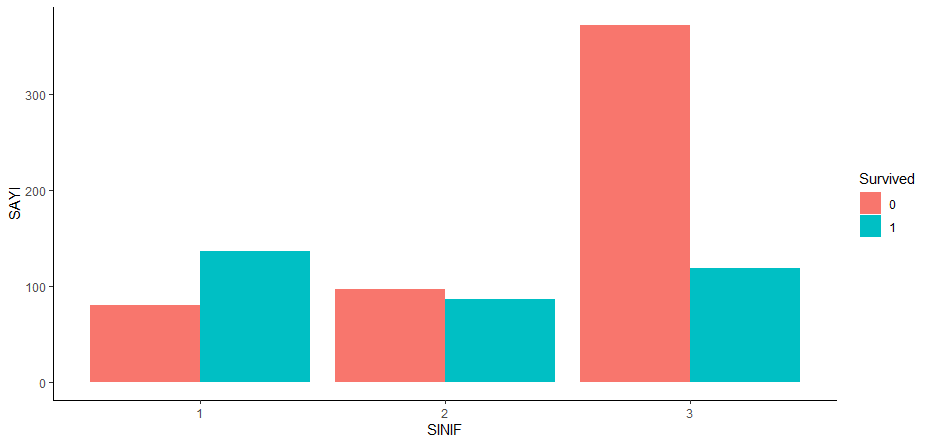
15 yasemin 18 kadın 2 2.89 75 geçti

#iki değişkenimde kategorik ise geom\_bar() kullanırım

ggplot(tam.veri[1:891,])+geom\_bar(aes(x=Pclass,fill=Survived),position = "dodge")+

theme\_classic()+xlab("SINIF")+

ylab("SAYI")



\*\*Birinci sınıftakilerin çoğu hayatta kalmış.

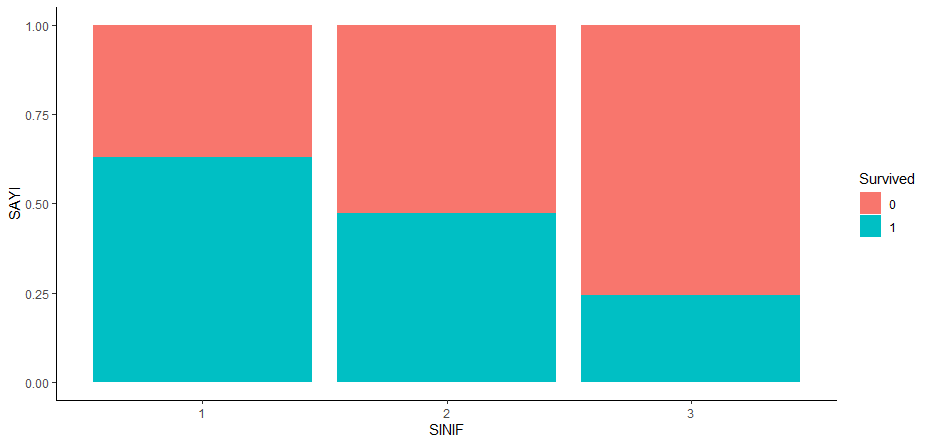
\*\*Yüzde olarak görmek istersem eğer.

ggplot(tam.veri[1:891,])+geom\_bar(aes(x=Pclass,fill=Survived),

position = "fill")+

theme\_classic()+xlab("SINIF")+

ylab("SAYI")



Birinci sınıftaki yolcuların %65’i hayatta kalmış.

Geom\_line() iki scale değişkenin grafiğini çizer.

Easyggplot kütüphanesi görsellerin tek bir karede yer almasını sağlayacak.

Eksik veri toplamı

sum(is.na(tam.veri$Age))

APPLY FONKSİYONU

İlk önce veri ismi yazılır daha sonra MARGIN = 2 sütunlarda işlem yapılacağı anlamına gelir MARGIN = 1 ise satırlar işlem yapılacağı anlamına gelir.

apply(tablo[,c(2,4,5)],MARGIN = 2,function(x) sum(x))

apply(tam.veri , MARGIN = 2 ,function(x) sum(is.na(x)))

tam.veri[is.na(tam.veri$Age),]

! işareti tersine döndürür. True ise False yapar.

Yaşı NA olmayan kişiler

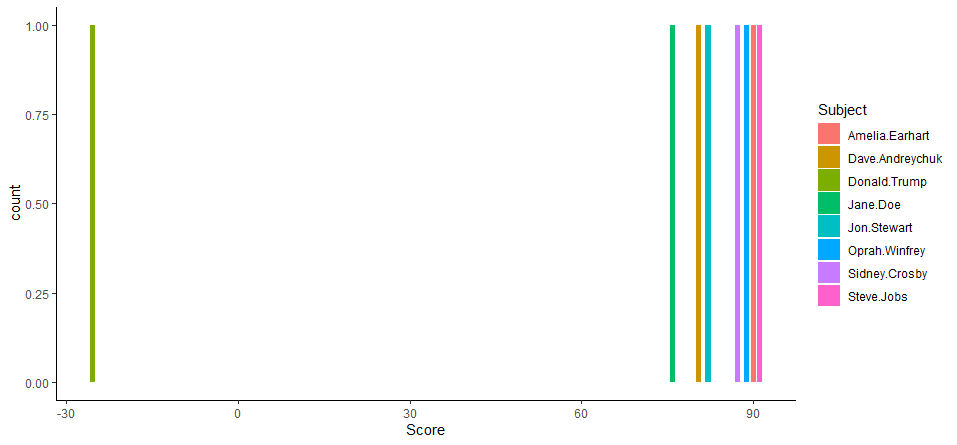
tam.veri[!is.na(tam.veri$Age),]

tablo$kişi[duplicated(tablo$kişi)]

unique(tablo$kişi[duplicated(tablo$kişi)])

ggplot(DataToExport)+geom\_bar(aes(x=Score,fill=Subject,main="GRAFİK"))+

theme\_classic()



Numeric ondalıklı innteger ise tam sayı demektir.

**RFM SKORLAMA**

#Recencey–7 en son 1 hafta önce alışveriş yaptım.

#Frequency-2 en son 1 sene önce ve 1 hafta önce alışveriş yaptım.

#Monetary-500 2 alışverişimde toplam 500 liralık alışveriş yaptım.

* Örneğin kategorilere ayıracağım mesela son 1 hafta içerisinde alışveriş yapanlar 5 puan son 1 ay içerisinde alışveriş yapanlar 4 puan gibi puanlandıracağım
* Aynı şeyi frekans değeri içinde yapacağım örneğin çok sık alışveriş yapan müşterime 5 puan daha az alışveriş yapan müşterime 4 puan vs..
* Aynı durum alışveriş tutarı içinde geçerli
* Ör herhangi bir müşterinin RFM skoru **553** bu demek oluyorki bu müşteri sürekli alışveriş yapan bir müşteri fakat çok yüksek miktarlarda alışveriş yapmıyor neden?

Bazen integer olan değerleri direk faktöre çevirmek problem olabilir bu yüzden ilk karaktere sonra faktöre çeviriyorum.

CDNOW\_sample$ID=as.factor(as.character(CDNOW\_sample$ID))

Tarihtede aynı şekilde tarih eğer integer olarak görünüyor ise ilk önce karaktere sonra date’e çevir.

**\*\*\*** CDNOW\_sample$TARİH=as.Date(as.character(CDNOW\_sample$TARİH),"%Y%m%d")

%Y%m%d yıl ay gün olarak formatladım **çok önemliii**.

Bugünün tarihi : Sys.Date()

Müşterinin en son ne zaman alışveriş yaptığı

CDNOW\_sample%>%group\_by(ID)%>%summarise(max(CDNOW\_sample$TARİH))

Müşterilerin yaptıkları satış sayısı

CDNOW\_sample%>%group\_by(ID)%>%summarise(satış\_sayısı=n())

colnames(CDNOW\_sample)

CDNOW\_sample=CDNOW\_sample[,-1]

colnames(CDNOW\_sample)=c("ID","TARİH","ADET","TUTAR")

head(CDNOW\_sample)

str(CDNOW\_sample)

CDNOW\_sample$ID=as.factor(CDNOW\_sample$ID)

CDNOW\_sample$TARİH=as.character(CDNOW\_sample$TARİH)

CDNOW\_sample$TARİH=as.Date(CDNOW\_sample$TARİH,"%Y%m%d")

library(tidyverse)

refDay=max(CDNOW\_sample$TARİH)

rfm\_recency=CDNOW\_sample%>%group\_by(ID)%>%summarise(Recency=as.numeric(refDay)-as.numeric(max(TARİH)))

rfm\_frequency=CDNOW\_sample%>%group\_by(ID)%>%summarise(satış\_sayısı=n())

rfm\_frequency=as.data.frame(rfm\_frequency)

rfm\_monetary = CDNOW\_sample%>%group\_by(ID)%>%summarise(Toplam\_Satış=sum(TUTAR))

rfm=merge(rfm\_recency,rfm\_frequency,by="ID")

rfm=merge(rfm,rfm\_monetary,by="ID")

quantile(rfm$Toplam\_Satış)

rankM=cut(rfm$Toplam\_Satış,breaks=c(0,20,45,105,1000,6600))

levels(rankM)

levels(rankM)=c(1,2,3,4,5)

levels(rankM)

head(rankM)

quantile(rfm$Recency)

#recency benim müşterimin en son ne zaman alışveriş yaptığıydı bu yüzden

#recency değeri düşük olan müşterim yüksek puan alacak

rankR=cut(rfm$Recency,breaks = c(0,60,217,473,506,550))

levels(rankR)

levels(rankR)=c(5,4,3,2,1)

quantile(rfm$satış\_sayısı)

sum(rfm$satış\_sayısı==1)

#müşterilerimin 1205 tanesi sadece 1 kez satış yapmış

rankF=cut(rfm$satış\_sayısı,breaks=c(0,1,2,3,7,60))

levels(rankF)

levels(rankF)=c(1,2,3,4,5)

rfmScores=cbind(rfm$ID,rankR,rankF,rankM)

rfmScores=data.frame(rfmScores)

head(rfmScores)

names(rfmScores)[1]="ID"

filter(rfmScores,ID=="136")

names(rfmScores)=c("ID","R","F","M")

filter(rfmScores,R==5,F==1,M==4)

**İnner join fonksiyonu**

left\_join(rfm\_recency,rfm\_frequency,by="ID") merge ile aynı

merge(rfm\_recency,rfm\_frequency,by="ID")

aynı şeyler merge ve left\_join

**ÇOOKKK ÖNEMLİİİİ**

urun\_ıd=data.frame(ID=1:6,urun\_adı=c("bisküvi","çikolata","gazoz","cips","kola","kek"))

urun\_detay=data.frame(ID=sample(1:6,50,replace = T),TUTAR=sample(20:500,50))

merge(urun\_ıd,urun\_detay,by="ID")

sum(rfm$satış\_sayısı==1)

satış sayısı 1 olanları topla

satış sayısını 5 faktöre bölüyorum

rankF=cut(rfm$satış\_sayısı,breaks=c(0,1,2,3,60))

**tarih fonksiyonu**

a=c(20200423,20200622)

a=as.character(a)

a=as.Date(a,"%Y%m%d")

a=c(11041999,23062020,22062020)

a=as.character(a)

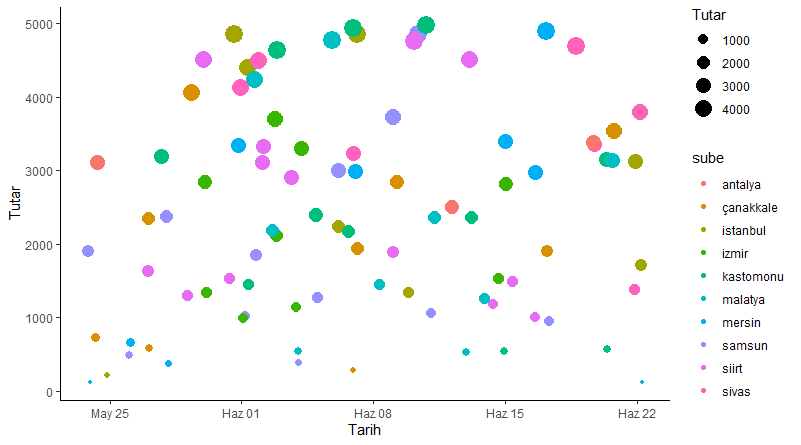
a=as.Date(a,"%d%m%y")

left\_join(musteri\_bilgi,satis\_detay,by="musteri\_ıd")%>%group\_by(sube)%>%summarise(toplam=sum(Tutar))

baloncuk grafiği

ggplot(a)+geom\_point(aes(x=Tarih,y=Tutar,color=sube,size=Tutar),position = "jitter")+

theme\_classic()



df=df[df$product!="NULL",]

df deki product sütunundaki “NULL” değerlerini kaldır.

orderID product

1 440 a

2 664 c

3 972 f

5 562 b

6 335 a

7 229 e

8 405 e

9 139 a

10 937 d bu şekildeki bir veriyi

df2=df%>%group\_by(orderID)%>%mutate(product=paste0(product,collapse = ";"))

fonksiyonu ile bu şekle soktuk.

orderID product

*<fct>* *<chr>*

1 440 a;a

2 664 c;a;d;a

3 972 f;b;b;g;b;a;a

4 562 b;f;b;a;f;g

5 335 a;a;e;b;g;h;b

6 229 e;a;f

Fakat burada tekrar eden veriler bulunduğundan bunların kaldırılması gerekir onu da fonksiyonu ile :

Unique(df2) yapılabilir.

!duplicated ile de yapılabilir ayni işlev.

df2=df2[!base::duplicated(df2$orderID),]

base demek duplicated fonksiyonuna ait başka pakette fonksiyonlar da olduğu için ben base deki yani r’ın kendisindeki duplicated fonskiyonunu kullandım.

nchar(df2$product[3])

13

3.ürün kaç karakter demek

> max(nchar(df2$product))

[1] 27

27 karakterin 13 ü noktalı virgül olduğu için ben en fazla 14 ürün satmışım.

Subset() fonksiyonunda ben istemediğim durumları belirtiyorum fonksiyon bana istemediğim durumların dışındaki yani istediğim durumları biçip veriyi çıkartıyor.

Subset örneği filter’a benzer pek fark yok.

Df 5000 satırlık veri seti

> subcategories=filter(df,product!="NULL")

> subcategories2=arules::subset(df$product,df$product!="NULL")

> head(subcategories)

orderID product

1 549 f

2 495 e

3 874 b

4 674 g

5 505 a

6 294 a

> head(subcategories2)

[1] "f" "e" "b" "g" "a" "a"

Arules::subset() arulesten subset demek.

filter(satislar,bolgeler!="karadeniz")

> save(a,file = "R a listesi.Rdata")

> getwd()

[1] "C:/Users/hatun/Documents"

Save ile dosyamızı Rdata uzantısı ile kaydederiz.

Getwd() nereye kaydettiğimizi söyler.

**LİSTELER**

> a=list(ad=c("harun","mehmet"),soyad="bakırcı",yas=20)

> a

$ad

[1] "harun" "mehmet"

$soyad

[1] "bakırcı"

$yas

[1] 20

> a$ad

[1] "harun" "mehmet"

Load () komutu, zaten bir .RData dosyasına kaydedilmiş olan veri yapılarını yeniden oluşturur. Önceki kodda kaydettiğimiz mydata.RData dosyasını yüklemek için şunu yazın:> load ("mydata.RData")

Prop.table olasılıklı table

> round(prop.table(table(arabalar$model)),2)\*100

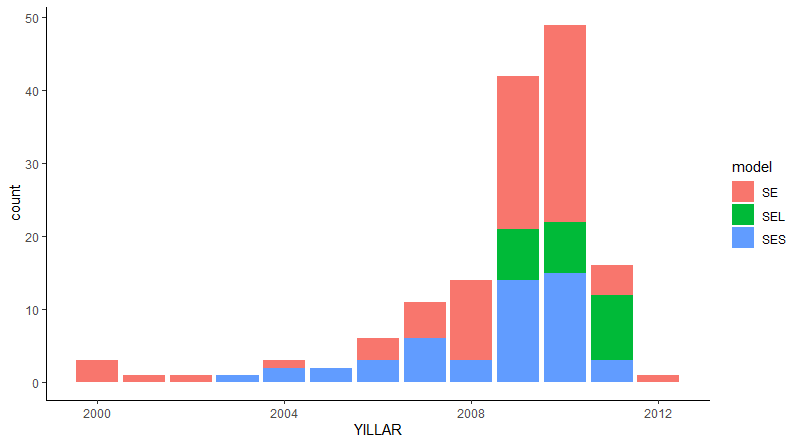
SE SEL SES

52 15 33

ggplot(arabalar)+

geom\_bar(aes(x=year , fill=model))+xlab("YILLAR")+

theme\_classic()



Kİ KARE TABLOSU

> library(gmodels)

> CrossTable(arabalar$model,arabalar$transmission)

Cell Contents

|-------------------------|

| N |

| Chi-square contribution |

| N / Row Total |

| N / Col Total |

| N / Table Total |

|-------------------------|

Total Observations in Table: 150

| arabalar$transmission

arabalar$model | AUTO | MANUAL | Row Total |

---------------|-----------|-----------|-----------|

SE | 63 | 15 | 78 |

| 0.190 | 1.108 | |

| 0.808 | 0.192 | 0.520 |

| 0.492 | 0.682 | |

| 0.420 | 0.100 | |

---------------|-----------|-----------|-----------|

SEL | 22 | 1 | 23 |

| 0.287 | 1.670 | |

| 0.957 | 0.043 | 0.153 |

| 0.172 | 0.045 | |

| 0.147 | 0.007 | |

---------------|-----------|-----------|-----------|

SES | 43 | 6 | 49 |

| 0.034 | 0.196 | |

| 0.878 | 0.122 | 0.327 |

| 0.336 | 0.273 | |

| 0.287 | 0.040 | |

---------------|-----------|-----------|-----------|

Column Total | 128 | 22 | 150

**KNN EN YAKIN KOMŞU ALGORİTMASI**

> a=knn(cbind(arabalar$year,arabalar$price,arabalar$mileage,arabalar$color,

+ arabalar$transmission),c(2010,20000,7500,factor("Silver"),factor("AUTO")),

+ arabalar$model,k=3)

> a

[1] SEL

Levels: SE SEL SES

Kategorik değişkenleri geom\_bar() ile görselleştir.

Scale değişkenleri geom\_point() scatter plotlar ile görselleştir.

table(satislar$bolgeler,satislar$sigara)

apply(table(satislar$bolgeler,satislar$sigara),

MARGIN = 1,

function(x) sum(x))

transmute()sadece o değişkeni filitreleyip alır.

Mutate dersen bütün veri setine değişken ekler.

yıl = transmute(CDNOW\_sample,YIL=substring(TARİH,1,4))

SORU:

Knn doğrusal olmayan modeller için mi kullanılır ?

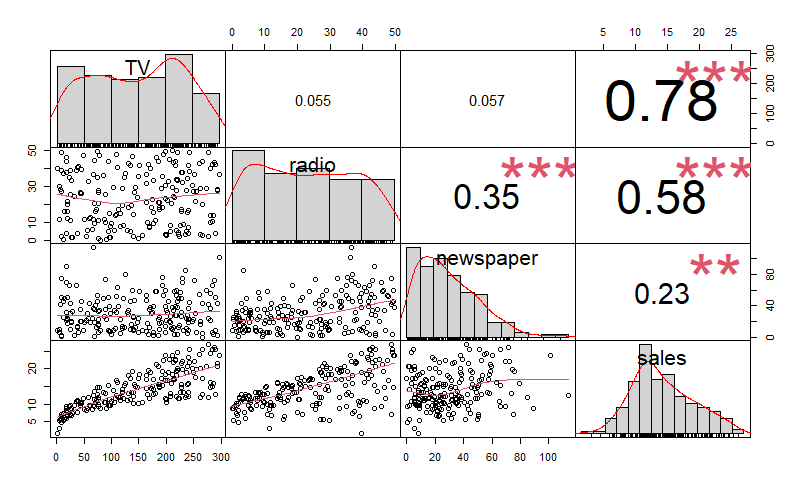
Gözetimli öğrenme y yani sonuç değişkenimin bulunduğu durum

Gözetimsiz öğrenme sonuç değişkeninin olmadığı uzaklık ile bulunan öğrenme modelidir.

pairs(df) bütün değişkenlerin birbiriyle olan ilişkilerini gösterir.

chart.Correlation(df)

değişknler arasındakikorelasyonu verir.



**GÜVEN ARALIĞI**

İnterval=”confidence”

> predict(lm\_model,yeni\_gozlem,interval = "confidence",levels=0.99)

fit lwr upr

1 17.81139 17.31781 18.30496

Cevap 17.811 en kötü .317 en iyi 18.304

Residuls artıklar bazen hata diye de geçebiliyor.

Makine öğrenmesi aslında artık optimizasyonudur. Yani gerçek değerlerden ne kadar uzakta olduğumuzu söyler.

Gerçek değerler ile tahmin edilen değerler arasındaki farktır.

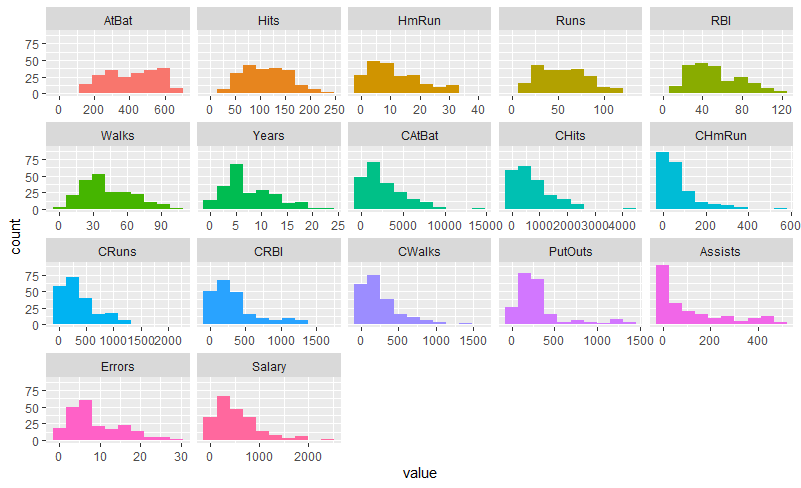
> hitters=na.omit(Hitters)

Na.omit eksik verileri siler.

Değişkenlerin histogram grafiği

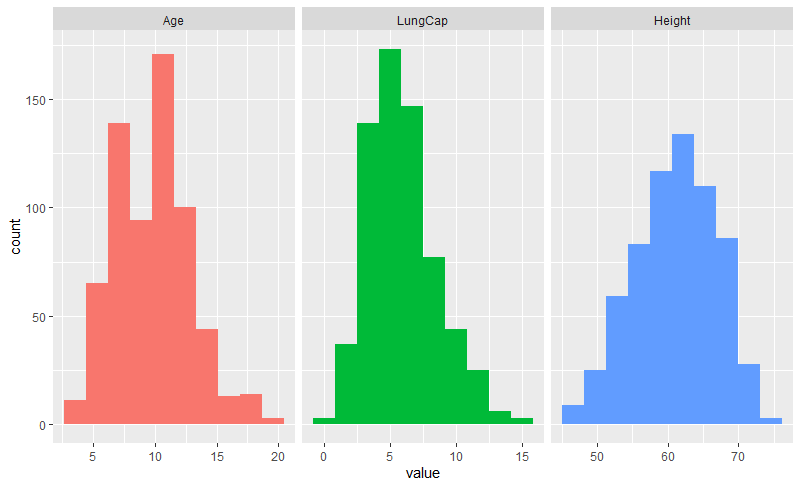
Funmodeling kütüphanesi ile çalıştırılır.

plot\_num(training)



Değişken seçmeye gerek yok kendisi ayarlıyor zaten.

plot\_num(LungCapData2)



> head(predict(lm\_fit, lm\_fit training veri setine + train\_x)) ait.

> head(lm\_fit$fitted.values)

İki fonksiyonda aynı işlevi görür. Yani modelin kendi içerisinde tahmin edilmesini temsil eder.

Eğitim hatasının hesaplanması

> defaultSummary(data.frame(obs=training$Salary,

+ pred=lm\_fit$fitted.values))

RMSE Rsquared MAE

289.3602571 0.5995248 209.9738029

Burada obs ve pred diye tanımlamamız gerekli ve de değişkenlerin boyulatı aynı olmalı yoksa çalışmaz.

Test hatasının hesaplanması biribirlerine yakın olmaları beklenir.

> defaultSummary(data.frame(obs=test\_y,

+ pred=predict(lm\_fit,test\_x)))

**LOJİSTİK REGRESYON ÖRNEK:**

library(tidyverse)

library(caret)

ilk başta bütün modele summary(glm()) yap anlamsız değişkenleri modelden çıkar eğitim ve test modellerini de bu değişkenler üzerinden kur ben bu veri setinde LungCap değişken p değeri 0.07 gibi olduğu için modele ekledim normalde eklememem lazım ama pek bir fark çıkmadı sonuçlar hemen hemen aynıydı.

Ör. eğitim\_model ve test\_model şöyle olmalıydı:

Eğitim\_model=glm(Smoke~Age+Gender+Height,family=”binomial”,data=df\_egitim)

egitim\_model=glm(Smoke~.,family = "binomial",data = df\_egitim)

tahmin\_test=predict(egitim\_model,df\_test[,-5],type="response")

tahmin\_test=ifelse(tahmin\_test>0.5,1,0)

tahmin\_test=ifelse(tahmin\_test==1,"evet","hayır")

mean(df\_test$Smoke==tahmin\_test)

> mean(df\_test$Smoke==tahmin\_test)

[1] 0.9076923

**Aynısını test veri setine uyguluyorum.**

test\_model=glm(Smoke~.,data = df\_test,family = "binomial")

tahmin\_egitim=predict(test\_model,df\_egitim[,-5],type="response")

tahmin\_egitim=ifelse(tahmin\_egitim>0.5,1,0)

tahmin\_egitim=ifelse(tahmin\_egitim==1,"evet","hayır")

table(tahmin\_egitim)

mean(tahmin\_egitim==df\_egitim$Smoke)

> mean(tahmin\_egitim==df\_egitim$Smoke)

[1] 0.8854962

> sapply(LungCapData2[,c(1,2,3)],median,na.rm=T)

Age LungCap Height

10.0000 5.6425 61.5000

> apply(LungCapData2[,c(1,2,3)],MARGIN = 2,function(x) median(x))

Age LungCap Height

10.0000 5.6425 61.5000

**KARAR AGACI TAHMİN ORANI:**

karar\_agac\_egitim=tree(Smoke~.,data = df\_egitim)

plot(karar\_agac\_egitim)

text(karar\_agac\_egitim)

tahmin\_test=predict(karar\_agac\_egitim,df\_test[,-5])

tahmin\_test=ifelse(tahmin\_test>0.5,1,0)

mean(tahmin\_test==df\_test$Smoke)

karar\_agac\_test=tree(Smoke~.,data = df\_test)

plot(karar\_agac\_test)

text(karar\_agac\_test)

tahmin\_egitim=predict(karar\_agac\_test,df\_egitim[,-5])

tahmin\_egitim=ifelse(tahmin\_egitim>0.5,1,0)

mean(tahmin\_egitim==df\_egitim$Smoke)

Tahmin ve test veri setlerimizin her ikisi içinde tahmin değeri 0.5 yani hemen hemen yakın çıktığı için kabul edilebilir bir değerdir.

text(karar\_agac,cex=1.2) cex yazıların boyutunu ayarlar.

Her zaman riskli durumu yani bağımlı sonuç değişkenini 0,1 şeklinde kodla.

**KARAR AGACI PREDİCT**

boy\_kilo=data.frame(kilo=sample(45:95,50,replace = T),

boy=sample(160:195,50,replace = T),

cinsiyet=sample(c("E","K"),50,replace = T))

indeks=createDataPartition(boy\_kilo$cinsiyet,

p=0.80,

list = F)

set.seed(100)

test\_boykilo=boy\_kilo[-c(indeks),]

egitim\_boykilo=boy\_kilo[c(indeks),]

egitim\_agac=tree(cinsiyet~.,data = egitim\_boykilo)

plot(tree(cinsiyet~.,data = egitim\_boykilo))

text(tree(cinsiyet~.,data = egitim\_boykilo))

tahmin\_test=predict(egitim\_agac,test\_boykilo)

tahmin\_test=ifelse(tahmin\_test>0.5,1,0)

tahmin\_test=ifelse(tahmin\_test==1,"E","K")

mean(tahmin\_test==test\_boykilo$cinsiyet)

test\_agac=tree(cinsiyet~.,data = test\_boykilo)

tahmin\_egitim=predict(test\_agac,egitim\_boykilo)

tahmin\_egitim=ifelse(tahmin\_egitim>0.5,1,0)

tahmin\_egitim=ifelse(tahmin\_egitim==1,"E","K")

mean(tahmin\_egitim==egitim\_boykilo$cinsiyet)

KNN ve DT hemen hemen birbirlerine yakın algoritmalardır.

KNN nin tahmin gücü yüksektir.

Eğer girdiler bağımsız değişkenler çoğu kategorik ise DT kullanılır.

Girdilerin bağımsız değişkenlerin çoğu sayısal ise knn kullanılır sonuçlar hemen hemen aynıdır.

ggplot(LungCapData2)+

geom\_point(aes(x=Age,y=LungCap,color=Gender,size=Smoke),position = "jitter")+

theme\_classic()



> cbind(a,ort=round(ort[,2],2),min=min[,2],mak=round(mak[,2],2))

Gender medyan ort min mak

1 female 5.458 5.35 0.373 9.51

2 male 5.818 6.44 0.388 15.38

**KNN TAHMİN**

boy\_kilo=data.frame(kilo=sample(45:95,50,replace = T),

boy=sample(160:195,50,replace = T),

cinsiyet=sample(c("E","K"),50,replace = T))

test=boy\_kilo[1:15,]

train=boy\_kilo[16:50,]

knn\_tahmin=knn(train[,c(1,2)],

test[,c(1,2)],

train$cinsiyet,

k=3)

> knn\_tahmin

[1] E E K E E K E E E E E K E K K

Levels: E K

> mean(knn\_tahmin==test$cinsiyet)

[1] 0.4

%40 tahmin eder.

Aşağıda görüldüğü gibi 9 kişiyi yanlış 6 kişiyi doğru tahmin etmiş.

> data.frame(test,tahmin=knn\_tahmin)

kilo boy cinsiyet tahmin

1 84 181 K E

2 85 192 K E

3 48 179 K K

4 50 172 E E

5 59 163 K E

6 60 170 K K

7 46 174 K E

8 76 179 K E

9 84 193 E E

10 65 187 E E

11 65 186 K E

12 57 190 E K

13 61 192 K E

14 69 169 E K

15 56 182 K K

Knn yapacağım zaman değişkenlerimin değerleri birbirlerinden aşırı uzak olmamalı ör. 1.değişkenim 0-50 arasında değişirken 2.değişkenim 0-1 arasında değişiyorsa knn yapamam bunları normalleştirmem lazım

Yani ortalamsını hepsinden çıkarıtorum std ye bölüyorum

Kısaca bu şekilde olmalı artık birbirlerinden çokta uzak değiller.

Scale() fonk. Kullanarak yapıyoruz.

a=c(10,100,5,6,150,290,3,2,500,1550)

b=scale(a)

**K-EN YAKIN KOMŞU SİGORTA TAHMİNİ**

karavan2=scale(karavan[,-86])

test=1:1000

train.X=karavan2[-test,]

test.X=karavan2[test,]

train.Y=karavan$Purchase[-test]

test.Y=karavan$Purchase[test]

tahmin=knn(train.X,test.X,train.Y,k=5)

> table(tahmin,test.Y)

test.Y

tahmin No Yes

No 930 55

Yes 11 4

Modelin tahmin gücü:

4/(4+11)

Toplam 15 kişinin sigorta alacağını tahmin etmişim fakat bunların 4’ü gerçekten sigorta almış diğer 11’i almamış.

Tahmin gücü: 4/15 = %26 düşük bir tahmin.

Model %6.6 hata ile çalışıyor yani sigorta alan 100 kişinin 6 sına almıyor diyor.

Zaten sigorta alan kişi sayısı %6’ydı.:D

> mean(test.Y!=tahmin)

[1] 0.066

lm\_fit=lm(salary~.,data = training)

lm\_fit$fitted.values

lm\_fit$fitted.values demek modelin tahmin ettiği değerler demek.

> defaultSummary(data.frame(obs=training$salary,

+ pred=lm\_fit$fitted.values))

RMSE Rsquared MAE

280.6643738 0.5867654 204.8242299

Defaultsummary ile modelin eğitim hatasını buluruz.

training$salary gerçek değerler

lm\_fit$fitted.values modelin tahmin ettiği değerler.

> defaultSummary(data.frame(obs=training$salary,

+ pred=lm\_fit$fitted.values))

Defaultsummary fonksiyonunda muhakkak obs ve pred değerleri bildirilmeli yoksa model çalışmaz.

Burada training$salary yani eğitim hatasını buldum şimdi obs yerine testi koyup test hatasını bulucam

Default summary test veya eğitim yani modlin hatasını bulur.

Eğitim hatası yukarıda yazıyor 0.58

Obs ve pred aynı boyutta olmalı.

**TEST HATASI**

> defaultSummary(data.frame(obs=test\_Y,

+ pred=predict(lm\_fit,test\_X)))

RMSE Rsquared MAE

402.5950121 0.3646558 240.5045799

Test hatası eğitim hatasına göre daha düşük.

RMSE test hatası Rsquared ise modelin testi tahmin olasılığı

**Eş örneklem**

t.test(a,test\_Y,paired = T)

wilcox.test(a,test\_Y,paired = T)

**\*\*\***bir kategorik ve bir scale değişkeni boxplot ile ifade ederiz.

names(training)[20]="Salary"

**BAGIMSIZ ÖRNEKLEM T TEST:**

library(DescTools)

> SignTest(a,mu=6500)

One-sample Sign-Test

data: a

S = 0, number of differences = 12, p-value = 0.0004883

alternative hypothesis: true median is not equal to 6500

96.1 percent confidence interval:

1 500

sample estimates:

median of the differences

65

a=rbind(summary(satislarr$gelir[satislarr$bolgeler=="marmara"]), summary(satislarr$gelir[satislarr$bolgeler=="güneydoğu"]),

summary(satislarr$gelir[satislarr$bolgeler=="karadeniz"]))

> rownames(a)=c("marmara","karadeniz","guneydogu")

> a

Min. 1st Qu. Median Mean 3rd Qu. Max.

marmara 19 42 54.5 59.06250 82.25 93

karadeniz 4 54 72.0 65.23077 90.00 99

guneydogu 4 24 42.0 41.85714 55.00 98

profiling\_num(tablo$not[tablo$cins=="erkek"])

profiling\_num(tablo$not[tablo$cins=="kadın"])

b=rbind(profiling\_num(tablo$not[tablo$cins=="erkek"]),

profiling\_num(tablo$not[tablo$cins=="kadın"]))

rownames(b)=c("erkek","kadın")

profilingnumber sürekli değişkenlerin özetini verir.

**TAHMİN ETTİĞİM DEĞERLERİN GÜVEN ARALIĞI:**

predict(lm\_fit,test\_X,interval="confidence")

RMSE hata kareler ortalaması demek değer başına yapacağım hata miktarıdır.

> shapiro.test(b$fark\_kare)

Shapiro-Wilk normality test

data: b$fark\_kare

W = 0.26772, p-value = 1.601e-14

> median(b$fark\_kare)

[1] 26731.99

> sqrt(median(b$fark\_kare))

[1] 163.4992

Fark\_kare’m normal dağılmamış bu yüzden medyanın kare kökünü aldım.

Xgboost algortiması bu aykırı değerlerin hangileri olduğunu tespit edip traşlar.

\*\*\*Doğrusal regresyonda hatalar normal dağılır.

\*\*\*Bağımsız değişkenler arasında çoklu doğrusal ilişki problemi yoktur.

\*\*\*Değişkenler arasında ör. 3 değişken arasında yüksek korelasyon varsa bunları tek bir değişkene düşürürüz.

Modeli kurabilmek için train\_x ve train\_y verilerinin birlikte bulunduğu training veri seti kurarız

Lm(salary~.,data=training)

pair tüm numeric değişkenler arasındaki korelasyon grafiğini verir.

**Aynı şeyler.**

> head(lm\_fit$fitted.values)

1 3 4 5 6 7

454.22066 1168.86255 481.74074 599.29014 173.78268 63.46355

> head(predict(lm\_fit,train\_x))

1 3 4 5 6 7

454.22066 1168.86255 481.74074 599.29014 173.78268 63.46355

**CV CROSS VALİDATİON**

•Validasyon; kullanılan yöntemin doğru ve kesin olarak sürekli bir şekilde

bekleneni gerçekleştirdiğinin kanıtlanması için yapılması gereken işlemleri kapsar.

•Valide edilmiş yöntemle çalışıldığında yapılan işlemin kesin,

doğru, spesifik, tutarlı, geçerli ve güvenilir olduğu garanti edilmiş olur.

* Predict fonksiyonu ile test veri setinin tahmini değerleri bulunur.
* defaultSummary(dataframe(obs=…,pred=….) fonksiyonu ile model ve test veri setlerinin hata oranları bulunur.
* Traincontrol(method=”cv”,number=10) fonksiyonu ile validasyon yöntemi seçilir.
* Number kaç katlı olacağını belirler.
* > lm\_val\_fit=train(x=train\_x,y=train\_y,

+ method="lm",

+ trControl=ctrl) train fonksiyonu ile validasyon modeli kurulur. Validasyon modeliyani lm\_val\_fit() aynen yazdırılır.

ÖZET

defaultSummary(data.frame(obs=test\_y,

pred=predict(lm\_fit,test\_x)))

**#test hatası hesaplanır.**

defaultSummary(data.frame(obs=train\_y,

pred=lm\_fit$fitted.values))

**#train eğitim hatası hesaplanır.**

ctrl=trainControl(method = "cv",

number = 10)

**#kurulacak olan validasyon yöntemi seçilir.**

lm\_val\_fit=train(x=train\_x,y=train\_y,

method="lm",

trControl=ctrl)

**#train fonksiyonu ile validasyon modeli kurulur.**

> lm\_val\_fit Lm\_val\_fit direkt yazdırılır.

Linear Regression

212 samples

19 predictor

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 192, 191, 190, 190, 191, 192, ...

Resampling results:

RMSE Rsquared MAE

317.9676 0.4806153 233.5473

Tuning parameter 'intercept' was held constant at a value of TRUE

> which(tablo$kişi=="harun")

[1] 6 11 16 19

sample(y) y vektörünü karıştırır.

KOLMOGROV SMİRNOV TESTİ:

Bağımlı değişkeni kategoriye göre ikiye ayır.

A=LungCapData2$LungCap[LungCapData2$Smoke==0]

B=LungCapData2$LungCap[LungCapData2$Smoke==1]

Sonra test et.

> ks.test(A,B)

Two-sample Kolmogorov-Smirnov test

data: A and B

D = 0.43456, p-value = 5.002e-10

alternative hypothesis: two-sided

**KARAR AGACI DefaultSummary**

a=iris[35:80,]

b=predict(tree(Species~.,data = iris),a[,-5])

s=c(1:46)

for(i in 1:46){

if(b[i,1]==1){

s[i]="setosa"

}

}

for(i in 1:46){

if(b[i,2]==1){

s[i]="versicolor"

}

}

for(i in 1:46){

if(b[i,3]==1){

s[i]="virginicia"

}

}

s=as.factor(s)

defaultSummary(data.frame(obs=a$Species,

pred=s))

data.frame(gercek=a$Species,tahmin=s)

library(gmodels)

CrossTable(a$Species,s)

> summary(glm(NewLeague~.,data = df\_hitters,family = "binomial"))

Call:

glm(formula = NewLeague ~ ., family = "binomial", data = df\_hitters)

Deviance Residuals:

Min 1Q Median 3Q Max

-2.6275 -0.2997 -0.1716 0.3586 2.8797

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -3.8409438 1.2486928 -3.076 0.0021 \*\*

AtBat 0.0129471 0.0089404 1.448 0.1476

Hits -0.0396329 0.0352049 -1.126 0.2603

HmRun -0.0644966 0.0830658 -0.776 0.4375

Runs 0.0024917 0.0398330 0.063 0.9501

RBI 0.0135420 0.0353105 0.384 0.7013

Walks 0.0120756 0.0260852 0.463 0.6434

Years -0.0636429 0.1526161 -0.417 0.6767

CAtBat -0.0023125 0.0019408 -1.192 0.2334

CHits 0.0199723 0.0102261 1.953 0.0508 .

CHmRun 0.0267149 0.0208756 1.280 0.2006

CRuns -0.0157892 0.0094444 -1.672 0.0946 .

CRBI -0.0142036 0.0097793 -1.452 0.1464

CWalks 0.0024483 0.0049143 0.498 0.6184

LeagueN 5.7477794 0.6313033 9.105 <2e-16 \*\*\*

DivisionW 0.0360110 0.5621488 0.064 0.9489

PutOuts -0.0001604 0.0010506 -0.153 0.8787

Assists -0.0007483 0.0031309 -0.239 0.8111

Errors -0.0501351 0.0587880 -0.853 0.3938

Salary -0.0001639 0.0007463 -0.220 0.8262

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 363.22 on 262 degrees of freedom

Residual deviance: 117.99 on 243 degrees of freedom

AIC: 157.99

Number of Fisher Scoring iterations: 6

Böyle bir modeli ancak knn veya desicion tree ile tahminleriz.

**\*\*\*** Veri çok karışık ise değişken sayısı cok fazla ise **KNN** algoritmasını kullanırız ama veri çok karışık değil ise **DESİCİON TREE** daha iyi tahminleme yapar.

> subset(data,YAS=="GENC")

YAS GELIR KREDI\_SKORU BILGISAYAR\_ALDI

1 GENC YUKSEK ORTALAMA HAYIR

2 GENC YUKSEK YUKSEK HAYIR

8 GENC ORTA ORTALAMA HAYIR

9 GENC DUSUK ORTALAMA EVET

11 GENC ORTA YUKSEK EVET

> filter(data,YAS=="GENC")

YAS GELIR KREDI\_SKORU BILGISAYAR\_ALDI

1 GENC YUKSEK ORTALAMA HAYIR

2 GENC YUKSEK YUKSEK HAYIR

3 GENC ORTA ORTALAMA HAYIR

4 GENC DUSUK ORTALAMA EVET

5 GENC ORTA YUKSEK EVET

> filter(data,YAS=="GENC",GELIR=="ORTA")

YAS GELIR KREDI\_SKORU BILGISAYAR\_ALDI

1 GENC ORTA ORTALAMA HAYIR

2 GENC ORTA YUKSEK EVET

Değişkeni Numeric character olarak tanımlama

a=c(0,1,0,0,0,1,1,0,0)

library(lfactors)

a=lfactor(a,levels = 0:1,labels = c("hayır","evet"))

ordinal sırala

temperature\_vector <- c("High", "Low", "High", "Low", "Medium")

# Specify that they are ordinal variables with the given levels

factor\_temperature\_vector <- factor(temperature\_vector, order = TRUE, levels = c("Low", "Medium", "High"))

factor\_temperature\_vector

**DESTEK VEKTÖR MAKİNESİ (SVM)**

\*gözetimli öğrenme algoritmasıdır.

\*her türlü veri tipini sınıflandırır.

Destek vektör makinesi(SVM) çok kullanışlı bir sınıflandırma algoritmasıdır.

Predict() ile tahmin ettikten sonra ifelse() dönüşümü yapmaya gerek yok zaten kendisi 0,1 diye sınıflandırıyor tahmin gücü çok yüksek.

Tune edilmesi gereken 2 tane parametresi var.

1. Gamma
2. Cost

library(e1071)

library(caret)

svm(Smoke~.,data=train)

summary(svm(Smoke~.,data=train))

predict(svm(Smoke~.,data=train),test\_x)

> defaultSummary(data.frame(obs=test\_y,

+ pred=predict(svm(Smoke~.,data=train),test\_x)))

Accuracy Kappa

0.9 0.0

Svm de sonuç %90 validasyon yapılmadı.

prop.table(table(test\_y,predict(svm(Smoke~.,data=train),test\_x)))

aynı veriyi karar ağacında tahmin ettim

library(tree)

tree(Smoke~.,data = train)

predict(tree(Smoke~.,data = train),test\_x)

s=c(1:length(test\_y))

for(i in 1:130){

if(predict(tree(Smoke~.,data = train),test\_x)[i,1]>=0.5){

s[i]=1

}

else if(predict(tree(Smoke~.,data = train),test\_x)[i,1]<0.5){

s[i]=0

}

} > defaultSummary(data.frame(obs=test\_y,

+ pred=s))

Accuracy Kappa

0.8615385 0.2307692

Sonuç %86.15 validasyon yapılmadı.

Accuracy değeri bir kategorik değişkenin açıklanma doğruluğudur.

Rsquare ise bağımsız değişkenlerin numeric bağımlı değişkeni % kaç oranda açıkladığıdır.

Yani accuracy %85 kanser veya %85 kanser değil der.

Rsquare ise değişkenler maaşınızın %50 sini açıklar maaşnızın %50 si bu değişkenlere bağlıdır der.

> round(prop.table(table(gercek=df\_hitters$NewLeague,tahmin=b))\*100,2)

tahmin

gercek A N

A 49.43 4.18

N 3.04 43.35

A grubuna ait olanların 49.43’ünü doğru bilmiş 4.18 kişiyi yanlış bilmiş

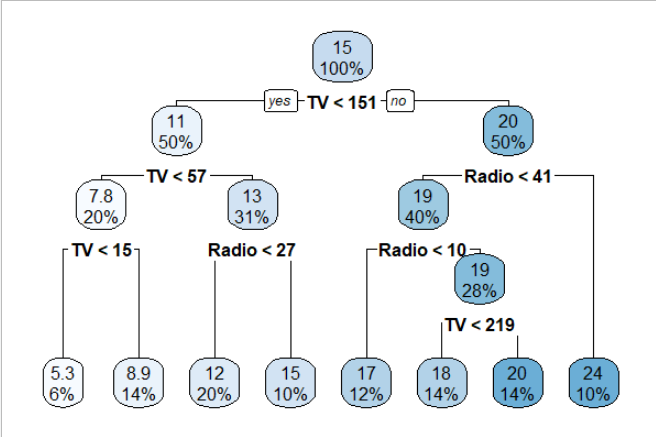
B grubuna ait olanların 43.35’ini doğru bilmiş 3.04’ünü yanlış bilmiş.

grep("Division",names(df\_hitters))

karar ağacı oluştururken RPART fonksiyonu kullan çünkü burada hem değişkenlerin önem seviyelerini görebiliriz hem de bu fonksiyon kendiliğinden 10 katlı cross validation işlemi yapabiliyor.

Rpart.plot() işlevi yüzdelikte verir.

rpart.plot(rpart(Sales~.,data = advertising),cex=1)



> filter(tablo,substr(kişi,1,1)=="h",

+ krediler>3)

kişi Yaş cins krediler GNO not sonuc

1 harun 24 erkek 4 2.25 80 geçti

2 harun 29 erkek 5 3.70 90 geçti

Group\_by() kullanımı

> filter(tablo%>%group\_by(kişi)%>%

+ summarise(top\_not=sum(not)),

+ substr(kişi,1,1)=="h",

+ top\_not>50)

`summarise()` ungrouping output (override with `.groups` argument)

# A tibble: 2 x 2

kişi top\_not

*<chr>* *<dbl>*

1 hakan 55

2 harun 295

ÖZET İSTATİSTİKLER:

k=tablo%>%group\_by(kişi)%>%summarise(n=n(),

min=min(not),

max=max(not),

ort=mean(not),

medyan=median(not))

> head(arrange(k,desc(n)))

kişi n min max ort medyan

1 harun 4 55 90 73.75 75

2 burcu 3 59 65 61.33 60

3 yağmur 3 15 60 36.67 35

4 abdullah 1 70 70 70.00 70

5 ali 1 85 85 85.00 85

6 berk 1 20 20 20.00 20

Faktörleştirme:

> not

[1] 20 60 15 85 92 70 65 55 80 42 55 35 50 30 75 80 59 70 90 60

> cut(not,breaks = c(0,16,32,48,64,92),labels = c("A","B","C","D","E"))

[1] B D A E E E E D E C D C D B E E D E E D

Levels: A B C D E

R da küçük eşit işareti =< değil <= böyledir

**McNemar Testi:**

> mcnemar.test(tahmin\_test,test$Smoke)

McNemar's Chi-squared test with continuity correction

data: tahmin\_test and test$Smoke

McNemar's chi-squared = 4.2667, df = 1, p-value = 0.03887

**çok fark var kullanıcıların fikirleri ciddi derecede değişmiş.**

factor(c("A","B","A","B","B","C"),levels = c("A","B","C"),ordered = T)

ordered=T

> okul=lfactor(sample(c(1,2,3),30,replace = T),levels = c(1,2,3),

+ labels = c("ilkokul","lise","üniversite"))

>

> okul=factor(okul,ordered = T)

> okul

[1] üniversite lise lise üniversite üniversite ilkokul lise üniversite

[9] ilkokul üniversite ilkokul lise lise ilkokul ilkokul ilkokul

[17] üniversite ilkokul ilkokul lise üniversite lise ilkokul lise

[25] lise ilkokul üniversite üniversite ilkokul ilkokul

Levels: ilkokul < lise < üniversite

**BİNOM DAĞILIMI**

**\*\*\*Bernoulli eger bir durumun sadece 2 olasılığı varsa ve bu olay sadece 1 kez tekrar ediliyorsa**

**bu bernoulli dağılımıdır.**

**\*\*\*Eğer Bernoullli dağılımı 1'den çok kez tekrar ediliyorsa bu Binom dağılımıdır.**

**Bir para 7 kez atılıyor 3 kez tura gelme olasılığı nedir ?**

> dbinom(3,7,prob = 0.50)

[1] 0.2734375

**Bir hastanede bir hastalıktan iyileşme olasılığı 0.07 dir. Bu hastaneye bu hastalıktan gelen 10 kişinin**

1. **İkisinin iyileşme olasılığı ?**
2. **Altısının iyileşme olasılığı nedir ?**
3. **En fazla ikisinin iyileşme olasılığı nedir ?**
4. **En az üçünün iyileşme olasılığı nedir ?**

> dbinom(2,10,prob = 0.07)

[1] 0.1233878

> dbinom(6,10,prob = 0.07)

[1] 1.848159e-05

> sum(dbinom(0:2,10,prob = 0.07))

[1] 0.9716579

> 1-sum(dbinom(0:3,10,prob = 0.07))

[1] 0.003576138

**POİSON DAĞILIMI**

**Bir maçta atılan gol sayısının 1.5 ortalamaya sahip olduğu biliniyorsa bu maçın golsüz bitme olasılığı nedir ?**

> dpois(0,lambda = 1.5)

[1] 0.2231302

**Bir bölgedeki günlük yanlış alarm sayısı ortalama 2’dir.**

**Yanlış alarm sayısının poison dağılımına uyduğu varsayılırsa;**

**a) Gelecek ilk günde hiç yanlış alarm almama olasılığı ?**

**b) Gelecek ilk günde ikiden fazla yanlış alarm alma olasılığı ?**

**c) En az 1 yanlış alarm alma olasılığı ?**

**d) En çok 1 yanlış yangın alarmı alma olasılığı nedir ?**

> dpois(0,2)

[1] 0.1353353

> 1-sum(dpois(0:2,2))

[1] 0.3233236

> 1-sum(dpois(0:1,2))

[1] 0.5939942

> dpois(0:1,2)

[1] 0.1353353 0.2706706

> sum(dpois(0:1,2))

[1] 0.4060058

**Bir mağazaya yarım saatte ortalama 12 müşteri gelmektedir. Bu mağazaya mesai saatleri içerisinde herhangi bir 5 dakikada hiç müşteri gelmeme olasılığı nedir ?**

30 dakikada 12 kişi gelirse ise 5 dakikada 2 kişi gelir.

> dpois(0,2)

[1] 0.1353353

**NORMAL DAĞILIM**

Ort=500 sd=100 P(X>624) ?

> 1-sum(dnorm(0:624,500,100))

[1] 0.1065652

**ÜSTEL DAĞILIM**

Saat ve uzunluk km metre mil bunlar hep sürekli değişkenler 10 kişi 5 adet bunlar kesikli değişkenler

1-e^(-lambda\*x)

F(10) demek P(X<10) demek.

Bir makinenin ortalama bozulma süresi 100 saattir. Derse üstel dağılıma girer çünkü zaman süreklidir.

Eger kesikli olsaydı poison kullanırdım.

Ortalamam süre ise üstel kesikli sayı ise poisson kullanırım.

**Bir kişinin ortalama taksi bekleme süresi 4 dk olduguna göre bu kişinin 3 dk’dan fazla taksi bekleme olasılığı nedir ?**

fonk.=1-e^(-lam\*x)

lam=1/4 x=3 P(X>3)=1-P(3)

1-(1-e^(-3/4)) cevap= %47.

**Bir araba ortalama 10000 mil yol gider. Bu arabanın 5000 millik yolu pili değiştirmeden tamamlama olasılığı nedir ?**

> 1-(1-(2.71^-0.5)) yani 5000’den daha fazla yol gitme olasılığı

[1] 0.6074567

**Bir öğrenci sınavda saatte ortalama 20 soru cevaplamaktadır. Öğrencinin ilk soruyu 5 dk yada daha uzun sürede çözme olasılığı ?**

P(X>5)=1- P(X<5) lam=1/3 1 soru 3 dk da çözülür

1-(1-e^(-lam\*x)) =

> 2.71^(-5/3)

[1] 0.1898386

**BİRDEN FAZLA DEĞİŞKENİN NORMALLİK VARSAYIMI**

> apply(LungCapData2[,c(1:2)],MARGIN = 2,FUN = shapiro.test)

$Age

Shapiro-Wilk normality test

data: newX[, i]

W = 0.97801, p-value = 2.404e-08

$LungCap

Shapiro-Wilk normality test

data: newX[, i]

W = 0.97052, p-value = 3.391e-10

\*\*Eger tüm sütunlara işlem yapacaksan apply kullan

**TANIMLAYICI İSTATİSTİKLER**

> ta=as.data.frame(rbind(apply(LungCapData2[,c(1:3)],MARGIN = 2,FUN = min),

+ apply(LungCapData2[,c(1:3)],MARGIN = 2,FUN = max),

+ round(apply(LungCapData2[,c(1:3)],MARGIN = 2,FUN = mean),2)))

> rownames(ta)=c()

> rownames(ta)=c("min","max","mean")

>

> ta

Age LungCap Height

min 3.00 0.373 46.00

max 19.00 15.379 74.00

mean 9.93 5.910 61.14

**MANOVA**

**MANOVA** :Çok değişkenli varyans analizi.

Her gruptaki veriler ayrı ayrı normal dağılmalıdır.

Manova anovayı tektek yapmak yerine bir işlemde yapmaktır.

> Y=cbind(arabalar$price,arabalar$mileage)

> summary.aov(manova(Y~arabalar$model))

Response 1 :

Df Sum Sq Mean Sq F value Pr(>F)

arabalar$model 2 276375070 138187535 17.268 1.837e-07 \*\*\*

Residuals 147 1176358866 8002441

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Response 2 :

Df Sum Sq Mean Sq F value Pr(>F)

arabalar$model 2 9.0183e+09 4509126112 6.6645 0.001696 \*\*

Residuals 147 9.9459e+10 676590523

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Ayrı ayrı yapılacağına tek satırda çözdü.

Ama bütün değişkenler kendi içerisinde normal dağılmalı bütün gruplarda.

Eksik veri: eger değişkende %20 den fazla eksik veri var ise o değişkeni sil.

Eger eksik veri %20’den az ise o zaman

* Silmek
* Basit atama
* Tahmine dayalı atama

Eger eksik veri türü kategorik ise yeni bir bilinmeyen kategorisi oluşturulur. Onun üzerinden analiz edilir.

Na girilmeyen veri demek.

Null de aynı eksik veri ama null kendi başına bir değişken türü yani nasıl integer character var ise Null diye de bir değişken türü var Na gibi o da eksik veri demek birbirlerinin yerine kullanılabiliyor.

Ortak noktaları bilinmeyen değerler içinde ikiside kullanılır.

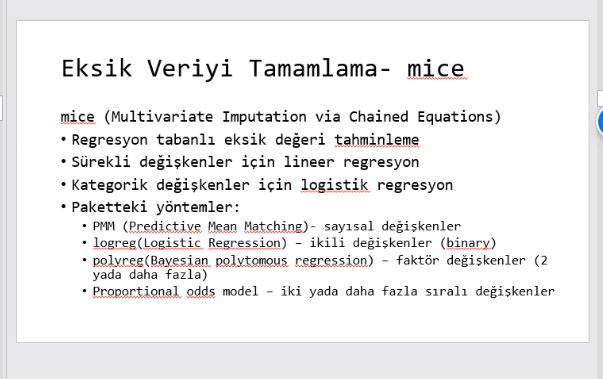
Eksik veriyi doldurmak en iyi yöntem diye bir şey yok

Eger veri doğrusal ise yani bütün değişkenler aynı anda alçalıp yükseliyorsa doğrusal interpolasyon kullanılabilir. Herhangi bir değişkeni seçip ona göre atayabilirsin.

Eger veri doğrusal değilse maksimum beklenti metodunu kullanabilirsin.

Tahminleyeceğim değerlerin birbirlerinden farklı olmasını istiyorsam jacknife metodunu kullan.

**Mice paketi**



Sapply Apply :

Sapply da function(x) sum(x) “,” virgül koyma

Apply veri,margın,FUN=sum

> sapply(df\_hitters,function(x) sum(is.na(x)/length(x)))

**AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits**

**0 0 0 0 0 0 0 0 0**

**CHmRun CRuns CRBI CWalks League Division PutOuts Assists Errors**

**0 0 0 0 0 0 0 0 0**

**Salary NewLeague**

**0 0**

> matrix(c(1,2,3,4,5,6),c(2,3))

[,1] [,2] [,3]

[1,] 1 3 5

[2,] 2 4 6

> grep("harun",tablo$kişi)

[1] 6 11 16 19

**> summary(filter(tablo,cins=="erkek",sonuc=="geçti"))[-c(2,5),]**

**kişi Yaş cins krediler GNO not**

**Length:7 Min. :19.00 kadın:0 Min. :2.000 Min. :2.200 Min. :70.0**

**Mode :character Median :26.00 Median :3.000 Median :3.100 Median :80.0**

**Mean :25.29 Mean :3.286 Mean :2.986 Mean :81.0**

**Max. :30.00 Max. :5.000 Max. :3.700 Max. :92.0**

**sonuc**

**geçti:7**

çeyreklikleri özetten çıkardım

Dersten geçen ve isminin harfi “b” olan kişileri bul.

> filter(tablo,sonuc=="geçti",substr(kişi,2,2)=="b")

kişi Yaş cins krediler GNO not sonuc

1 abdullah 26 erkek 4 3.1 70 geçti

Dersten geçen ve isminin harfi “b” olmayan kişileri bul.

> filter(tablo,sonuc=="geçti",substr(kişi,2,2)!="b")

kişi Yaş cins krediler GNO not sonuc

1 burcu 24 kadın 2 2.50 60 geçti

2 ali 26 erkek 3 3.10 85 geçti

3 sıla 23 erkek 2 2.20 92 geçti

4 harun 30 erkek 3 3.50 70 geçti

5 burcu 20 kadın 2 2.50 65 geçti

6 berkan 19 erkek 2 3.05 80 geçti

7 yasemin 18 kadın 2 2.89 75 geçti

8 harun 24 erkek 4 2.25 80 geçti

9 harun 29 erkek 5 3.70 90 geçti

10 yağmur 26 kadın 5 3.12 60 geçti

> tablo%>%group\_by(kişi)%>%summarise(min=min(not),

+ max=max(not),

+ mean=mean(not),

+ sum=sum(not))

`summarise()` ungrouping output (override with `.groups` argument)

# A tibble: 13 x 5

kişi min max mean sum

*<chr>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*

1 abdullah 70 70 70 70

2 ali 85 85 85 85

3 berk 20 20 20 20

4 berkan 80 80 80 80

5 burcu 59 65 61.3 184

6 hakan 55 55 55 55

7 harun 55 90 73.8 295

8 hasan 50 50 50 50

9 hüseyin 30 30 30 30

10 pelin 42 42 42 42

11 sıla 92 92 92 92

12 yağmur 15 60 36.7 110

13 yasemin 75 75 75 75

Multivariate manova

Üstündeki univariate ancova demektir.

Ancova da değişken kontrol altına alınır yani ör:

2 farklı tedavi yönteminin depresyon üzerindeki etkisini inceleyeceğim ama bu veriler farklı zamanlarda alındığı için hava değişkeni kontrol altına alınmak zorunda havanın kontrol edildiği durumda hangi tedavi yöntemi depresyon üzerinde daha etkilidir.

\*\*\*Eger birden fazla bağımlı değişkenim var ise yapmam gereken test manovadır. Tek tek anova yaparsam benim tip 1 hatam artar bu yüzden manova yaparım.

Birden fazla bağımlı değişken var ise manova yap.

T testlerinde ben gruplar arasında fark olup olmadığını denetlerken regresyonda değişkenlerin sonuç değişkenini ne kadar açıkladığını inceliyorum.

Skewness curtosis basıklık çarpıklık.

Eger 1 adet bağımsız 1 adet bağımlı değişenim var ise tek yönlü varyans analizi one way anova

Eger 2 adet bağımsız değişenim 1 adet bağımlı değişkenim var ise 2 yönlü varyans analizi denir two way anova

Manova çok değişkenli varyans analizi.

Covaryans analizi demek ancova demek.

Anova Ancova Manova Mancova

Veri setinin en geniş özetini istediğim özette aldım.

> rbind(min=apply(mtcars,2,min),

+ max=apply(mtcars,2,max),

+ mean=apply(mtcars,2,mean))

mpg cyl disp hp drat wt qsec vs am gear carb

min 10.40000 4.0000 71.1000 52.0000 2.760000 1.51300 14.50000 0.0000 0.00000 3.0000 1.0000

max 33.90000 8.0000 472.0000 335.0000 4.930000 5.42400 22.90000 1.0000 1.00000 5.0000 8.0000

mean 20.09062 6.1875 230.7219 146.6875 3.596563 3.21725 17.84875 0.4375 0.40625 3.6875 2.8125

eger bir gruptaki değişkenin dağılımını inceleseydim group\_by() ile min max mean özetlerini çıkarırdım.

Fonksiyon

> a=function(a,b){

+ return(a+b-a^2)

+ }

> a(10,20)

[1] -70

Fonksiyon

> a=function(a,b){

+ c=(a+b-a^2)

+ c=as.character(c)

+ print(paste(c,"harun bakırcı"))

+ }

> a(10,20)

[1] "-70 harun bakırcı"

> t=c(10,20,NA,15,NA,6,8)

> m=c()

> for(i in 1:length(t)){

+ if(is.na(t[i])==TRUE){

+ m=c(m,i)

+ }

+ }

>

> m

[1] 3 5

subset(LungCapData2,select = c(Age,Gender))

knn fonksiyonunda predict işlemine gerek yok zaten kendisi hesaplıyor.

Algoritma su sekilde çalışıyor:

Knn(cbind(bağımsız değişkenler),test edilecek verinin bağımsız değişkenleri,bağımlı değişken ,k değeri)

**Kruskall Wallis**

> kruskal.test(Sepal.Length~Species,data = iris)

Kruskal-Wallis rank sum test

data: Sepal.Length by Species

Kruskal-Wallis chi-squared = 96.937, df = 2, p-value < 2.2e-16

Veriler normal dağılmamış.

**NORMALLEŞTİRME İŞLEMİ**

Öncelikle şunu açıklayayım normalleştirme ile scale etmek yani ölçeklendirmek ayrı şeyler.

1. Kategorik değişkenler normalleştirilmez
2. Veriyi log10 ve karekök alarak normalleştirebiliriz
3. Log10 ile yapacaksak eger veride negatif sıfır olmamalı bunu engellemek için eger negatif ise veriyi 1 yapmak adına her değişkene ekleme yaparız örneğin en küçük değer -6 ise tüm verilere 7 ekleriz
4. Aynı işlemler karekök içinde geçerli karekök alırken – ve 0 değer olmaz burada da veriye değer ekleriz 1 yaparız.
5. 1/değişken olarakta dönüşüm yapabiliriz.
6. Karesini alarak da dönüştürme işlemi yapabiliriz.

**Ör:**

a=c(10,1,2,105,296,1,3,2,65)

shapiro.test(a) veri normal dağılmıyor.

log\_10\_a=log10(a)

shapiro.test(log\_10\_a) veri artık normal dağılıyor.

> shapiro.test(a)

Shapiro-Wilk normality test

data: a

W = 0.6334, p-value = 0.0002358

> shapiro.test(log\_10\_a)

Shapiro-Wilk normality test

data: log\_10\_a

W = 0.86649, p-value = 0.1127

Karekök alma ve logaritma işlemlerini veri setinin değişkenlerine dene uygun olana göre normalleştir.

> shapiro.test(iris$Sepal.Length)

Shapiro-Wilk normality test

data: iris$Sepal.Length

W = 0.97609, p-value = 0.01018

> shapiro.test(sqrt(iris$Sepal.Length))

Shapiro-Wilk normality test

data: sqrt(iris$Sepal.Length)

W = 0.98064, p-value = 0.03283

> shapiro.test(log10(iris$Sepal.Length))

Shapiro-Wilk normality test

data: log10(iris$Sepal.Length)

W = 0.98253, p-value = 0.05388

Knn algoritmasında değişken önem düzeyleri söz konusu değil çünkü uzaklık matrisinden yararlanıyor.

Soran=Yararlı araç ipucu beni bu sorunun öznel olduğu konusunda uyardı, ancak öyle olduğunu sanmıyorum. KNN tipi bir durumda herhangi bir özelliğin önemini belirleyip belirleyemeyeceğinizi teorik bir perspektiften belirtmek oldukça objektif olmalıdır.

Yapabileceğinizi sanmıyorum, çünkü eğer veriler doğru bir şekilde ölçeklenirse, herhangi bir değişkene bakılmaksızın sadece hepsi birbirine yakın olan noktalar olacaktır, benim anlayışıma göre, hangi noktaların komşu olduğunu belirlemek için hepsi eşit derecede yararlı olmalıdır. hiper boyutlu uzay. Yeni bir noktanın tüm k noktalarının tek bir sınıfta olması durumunda, değişkenlerin hangisine yakın olduğu önemli değildir, sadece p -boyutlu uzayda bunlardan yeterince yakın olduğu için a komşu.

Anladığım doğru mu? Ya da hangi değişkenlerin mesafe üzerinde en büyük mutlak etkiye sahip olduğunu (bu sadece değişkenlerin standart sapması olabilir ...) / hangi noktaların komşu olarak etiketlendiğini bir şekilde kurtarabilir misiniz?

Soruyorum çünkü veri bilimcisi olmayan patronum, KNN sonuçlarım için hangi özelliklerin en önemli olduğunu söylememi istedi ve bunun makul bir şekilde mümkün olmadığını söylerken doğru olduğumdan emin olmak istiyorum (örneğin, knn sonuçlarında bir gerileme yapmak dışında) ve bu ikinci el önem değerlerini kullanarak).

Cevap=Haklısınız, KNN değişkenlerin önemi veya katsayıları için bir öngörü sağlamaz. Katsayıları hesaplamak için regresyon (veya rasgele orman) gibi başka bir model çalıştırmak önerdiğiniz alternatifin daha mantıklı bir yaklaşım olduğunu düşünüyorum. Benzer bir soru zaten de istendi [Stack Exchange](https://stackoverflow.com/questions/29865750/how-to-calculate-knn-variable-importance-in-r)

**Multinominal regresyon :**

library(nnet)

multinom(Species~.,data = iris)

predict(multinom(Species~.,data = iris),

iris[,c(1:4)])

defaultSummary(data.frame(obs=iris$Species,

pred=predict(multinom(Species~.,data = iris),

iris[,c(1:4)])))

multinom(formula = Species ~ ., data = iris)

Coefficients:

(Intercept) Sepal.Length Sepal.Width Petal.Length Petal.Width

versicolor 18.69037 -5.458424 -8.707401 14.24477 -3.097684

virginica -23.83628 -7.923634 -15.370769 23.65978 15.135301

Residual Deviance: 11.89973

AIC: 31.89973

Versicolor ve virginica için tahminleme yapmış hiç de doğrusal bir model değil.

**Random forest:**

> randomForest(Species~.,data = iris)

Call:

randomForest(formula = Species ~ ., data = iris)

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 2

OOB estimate of error rate: 4.67%

Confusion matrix:

setosa versicolor virginica class.error

setosa 50 0 0 0.00

versicolor 0 47 3 0.06

virginica 0 4 46 0.08

hata oranı %4.67 baya başarılı bir model.

Setosa sınıfını hatasız tahmin ediyor.

Diğer grupları da %6 , %8 hata ile tahmin ediyor.

> names(rf)

[1] "call" "type" "predicted" "err.rate" "confusion"

[6] "votes" "oob.times" "classes" "importance" "importanceSD"

[11] "localImportance" "proximity" "ntree" "mtry" "forest"

[16] "y" "test" "inbag" "terms"

Değişkenlerin önem düzeyleri.

> rf$importance

MeanDecreaseGini

Sepal.Length 10.053422

Sepal.Width 2.278759

Petal.Length 42.608860

Petal.Width 44.314679

Karşılaştırma matrisi:

> confusionMatrix(rf$predicted,iris$Species)

Confusion Matrix and Statistics

Reference

Prediction setosa versicolor virginica

setosa 50 0 0

versicolor 0 47 3

virginica 0 3 47

Overall Statistics

Accuracy : 0.96

95% CI : (0.915, 0.9852)

No Information Rate : 0.3333

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.94

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: setosa Class: versicolor Class: virginica

Sensitivity 1.0000 0.9400 0.9400

Specificity 1.0000 0.9700 0.9700

Pos Pred Value 1.0000 0.9400 0.9400

Neg Pred Value 1.0000 0.9700 0.9700

Prevalence 0.3333 0.3333 0.3333

Detection Rate 0.3333 0.3133 0.3133

Detection Prevalence 0.3333 0.3333 0.3333

Balanced Accuracy 1.0000 0.9550 0.9550

Accuracy modelin tahmin gücü

Sensitivity ise sınıfların doğru açıklanma yüzdeleri.

Svm de tahminlemeyi data.frame() üzerinden yap.

predict(svmm,

data.frame(Sepal.Length=c(5.7),

Sepal.Width=c(2.9),

Petal.Length=c(4.2),

Petal.Width=c(1.3)))

Garson fonksiyonu sadece 1 çıkış düğümü olan modeller için kullanılır.

Yani bağımszı değişkenler sürekli ise garson fonksiyonu kullanılabilir.

Eger kategorik ise yani birden fazla çıkışı varsa kullanılamaz.

Doğrusal olmayan regresyon modellerinde sürekli değişkenlerin logaritmaları alınarak model dogrusallaştırılabilir.

Kategorik değişkenlere karışma onlara müdahale edilmez. Bir de değişkenlerde negatif veya 0 değeri bulunmamalı bu yüzden en küçük negatif değer ör -6 ise değişkenin bütün verilerine +7 ekle

In tabanında da böyle log10 tabanında da böyle

Normal olmayan verileri bu sekilde normalleştirebiliriz.

Ornegin anova yapacağız değişkenler normal değil ise bu sekilde normalleştir.

Trendaz akademi:

Dogrusallik varsayimi saglamiyorsa verinin logaritmasi alinarak donusum yapilabilir. Yine saglamiyorsa non parametrik testler uygulanabilir.

Ancova:

> anova(lm(gelir~reklam+factor(bolgeler),data = satislarr))

Analysis of Variance Table

Response: gelir

Df Sum Sq Mean Sq F value Pr(>F)

reklam 1 1427 1426.80 2.0754 0.1565

factor(bolgeler) 2 5279 2639.71 3.8397 0.0287 \*

Residuals 46 31624 687.48

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Gelirin bölgeler üzerindeki dağılımını reklam değişkenini tutarak değerlendir.

Bölgeler p=0.0287 old. İçin anlamlı bir fark vardır.

Model tuning de tüne edilecek parametreleri seçiyoruz kaç değişkene bakması gerektiğini söylüyoruz.

Çok değişkenli regresyonda validasyon yaparken train()

Fonksiyonuna tüne\_grid eklemiyorum

Tüne\_grid değişken ayarlamak demek random forest knn svm gibi algoritmalarda modelin en iyi değişken sayısını seçmesi için tüne ediyoruz.

Tune\_grid() hiperparametreleri optimize etmiş oluyoruz.

**Model parametreleri:** veri içerisinden bulduğumuz değişken katsayılarıdır.

**Model hiperparametreleri:** daha sağlıklı parametreler bulabilmemiz için modele verdiğimiz sayılardır örneğin knn algoritmasında en iyi parametreleri bulabilmem için k değerim yani hiperparametrem ne olsun.

**Model tuning :**hiperparametreler aracılığıyla parametreleri iyileştirme modeli iyileştrime anlamına gelir.

Yapay sinir ağlarıda kategorik değişkenlerde numeriğe cevirilmeli

For example, assume there was a “Gender” column having values “Male” and “Female” originally.

$ Gender                          : Factor w/ 2 levels “Female”,”Male”: 2 2 2 1 2 2 1 2 2 1 …

After the conversion, two more dummy variables will be added as below.

$ GenderFemale                 : num  0 0 0 1 0 0 1 0 0 1 …  
$ GenderMale                      : num  1 1 1 0 1 1 0 1 1 0 …

Eger yapay sinir ağlarında stepmax ile ilgili hata alırsan stepmax değerini düşük yap

Error in plot.nn(neuralnet(yol ~ ., data = a, hidden = 1, stepmax = 0.001)) :

weights were not calculated

Ek olarak: Warning message:

Algorithm did not converge in 1 of 1 repetition(s) within the stepmax.

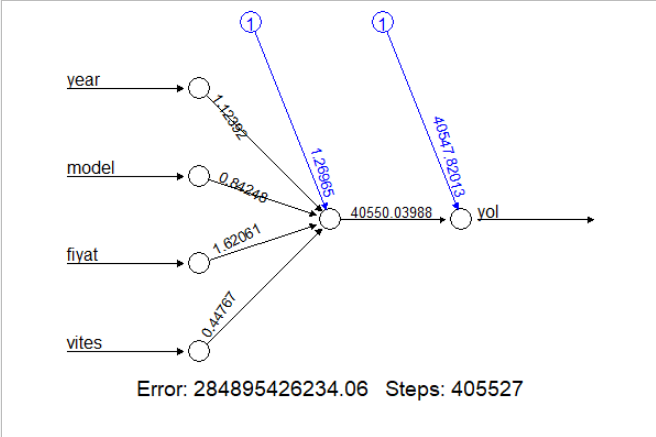
Böyle bir hata alırsan

Stepmax’ı arttır.

plot(neuralnet(yol~.,data = a,

hidden = 1,

stepmax =1e+08))

yıl değişkeninin katsayısı 1.2392

model değişkeninin katsayısı 0.64248

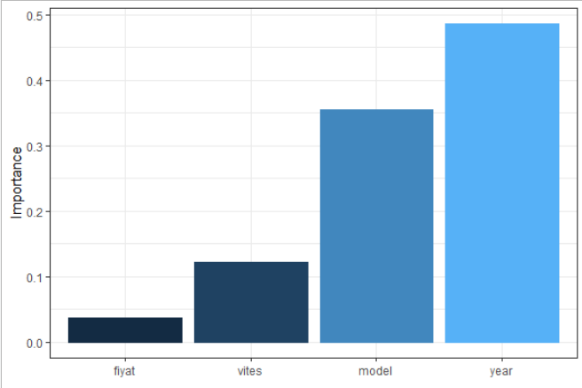
ilk mavi çizgi ise modelin sabiti:1.2695

değişken önem düzeyleri:

> garson(neuralnet(yol~.,data = a,

+ hidden = 1,

+ stepmax =1e+08))

yapay sinir ağında modeli kurmadan önce değişkenleri min max dönüşümü ile ölçeklendir.

doğrusal regresyonda anlamlı çıkan değişkenler yapay sinir ağında anlamsız çıkabililr çünkü yapay sinir ağları doğrusal olmayan modellerde kullanılır.

Min max doğrusallaştırmadır.

Yapay sinir ağlarında kategorik tahminleme yaparken bağımlı değişken integer olarak kalır bağımsız değişkenler numerictirler ve de min max donuşumu yapılır kategorik değişkenler dahil.

Tahminleme defaultsummary değilde table matris üzerinden el ile hesaplanır.

ysa\_tahmin=compute(neuralnet(vites~.,data = a,

hidden = 1),a[,-5])

ysa\_tahmin=ifelse(ysa\_tahmin$net.result>1.5,2,1)

defaultSummary(data.frame(obs=a$vites,

pred=ysa\_tahmin))

table(tahmin=ysa\_tahmin,

gercek=a$vites)

yapay sinir ağının %60 ile tahmin ettiği veriyi svm %61 tahmin etti

a$vites=as.factor(a$vites)

a$model=as.factor(a$model)

a$year=as.factor(a$year)

library(e1071)

svm(vites~.,data = a)

summary(svm(vites~.,data = a))

defaultSummary(data.frame(obs=a$vites,

pred=predict(svm(vites~.,data = a),

a[,c(1:4)])))

scale fonksiyonuyla değişkenin standart sapmasını düşürürüz eger değişkenlerin standart sapmaları çok yüksekse scale edebilirsin

> sd(scale(suc\_verisi$Assault))

[1] 1

> sd(log(suc\_verisi$Assault))

[1] 0.568251

> sd(suc\_verisi$Assault)

[1] 83.33766

Scale=ortalamdan çıkartıp standart sapmaya böler.

Min max dönüşümü de scale gibi bir ölçeklendirme yöntemidir.

Min max dönüşümü 0 ile 1 arasında değerler alır.

Yapay sinir ağında tahmin edilecek kategorik değişken ise min max dönüşümü yap 0-1 aralığına çek

Diğer değişkenlere sadece scale ile ölçeklendir.

Yapay sinir ağlarında bütün değişkenler sayısaldır.

Neuralnet fonksiyonunda ~. Nokta kullanılmıyormuş değişkenleri tek tek eklemen gerek (araştır)

Yanlılık parametresi b0 yani sabit katsayı.

Boyut küçültme değişken sayısının azaltılması yani bazı değişkenlerin katsayılarının sıfır yapılması.

Değişken sayısı ne kadar az olursa açıklama o kadar kolay olur.

PCA :temel bileşen analizi 100 değişkenli veriyi 2 değişken ile tanımlamak.

Bu analiz edilen değişkenlerinde regresyonunun alınması PCR temel bileşen regresyonu.

Ortaya çıkan değişkenler birbirleriyle ilişkisi olmayan değişkenlerdir. Pcr çoklu bağlantı problemini de çözer.

Çok boyutluluk probleminden kurtulunur. Çok boyutluluk problemi veri sayısı değişken sayısından azdır.

İki tane çözüm var ya değişkenler arasındaki korelasyonu bulup yüksek korelasyonlu değişşkenlerin birini ikisini modelden çıkarmak ya da temel bileşen analizi yapmak fonksiyonu pls

**PCR (temel bileşen analizi)**

library(pls)

> summary(pcr(satis\_sayisi~.,

+ data=marketing,

+ scale=TRUE,

+ validation="CV"))

Data: X dimension: 200 3

Y dimension: 200 1

Fit method: svdpc

Number of components considered: 3

VALIDATION: RMSEP

Cross-validated using 10 random segments.

(Intercept) 1 comps 2 comps 3 comps

CV 6.277 5.013 2.871 2.067

adjCV 6.277 4.996 2.866 2.063

TRAINING: % variance explained

1 comps 2 comps 3 comps

X 45.70 78.47 100.00

satis\_sayisi 37.71 80.14 89.72

bileşenlerin varyans açıklama oranları da verilmiş.

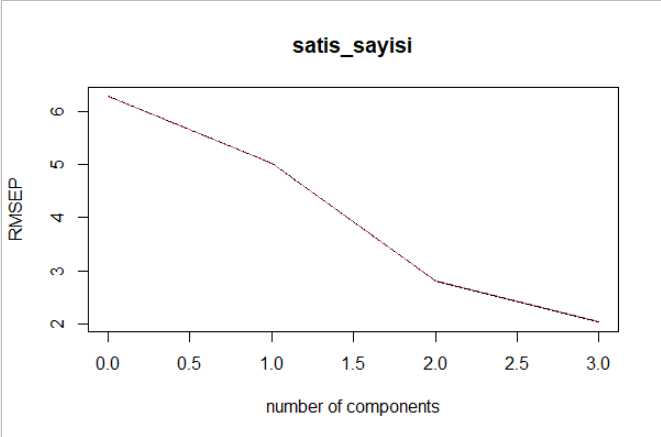
En az hatayı da 3 bileşenli regresyon veriyor.

> validationplot(pcr(satis\_sayisi~.,

+ data=marketing,

+ scale=TRUE,

+ validation="CV"))

Bileşen sayısı ve hata kareler toplamı

Model tune ederken yazılan preProc(c(“scale”,”center”))

Modeli ölçeklendir anlamına geliyor.

Corelasyon fonksiyonu

library(psych)

pairs.panels(iris[,c(-5)])

BORUTA paketi değişken seçimi

Boruta(Species~.,data = iris,

doTrace=3) 3 sınıfım olduğu için 3 dedim.

Sonuç:

Boruta performed 9 iterations in 0.3969269 secs.

4 attributes confirmed important: Petal.Length, Petal.Width,

Sepal.Length, Sepal.Width;

No attributes deemed unimportant.

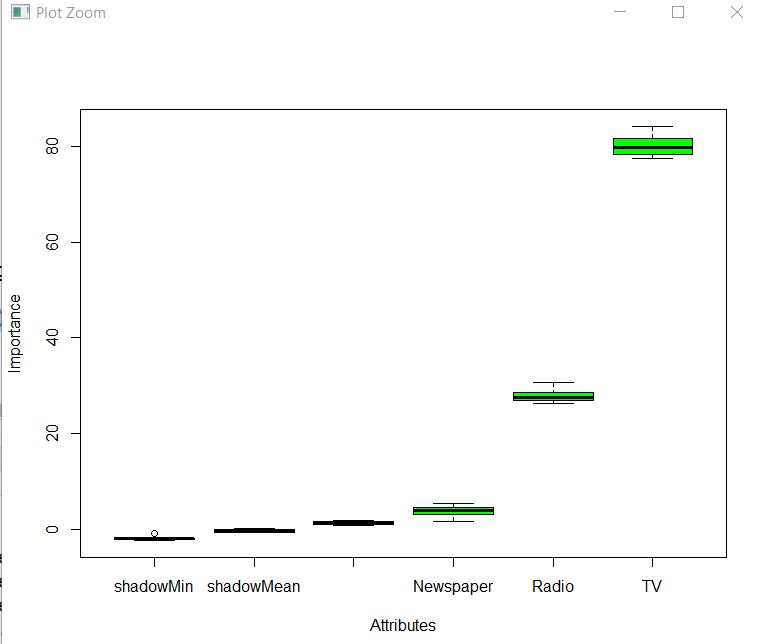
Boruta 0,3969269 saniyede 9 iterasyon gerçekleştirdi.

4 özellik önemli olduğunu doğruladı: Petal.Length, Petal.Width, Sepal.Length, Sepal.Width;

Önemsiz sayılan niteliklerin hiçbiri.

Değişken önemi:

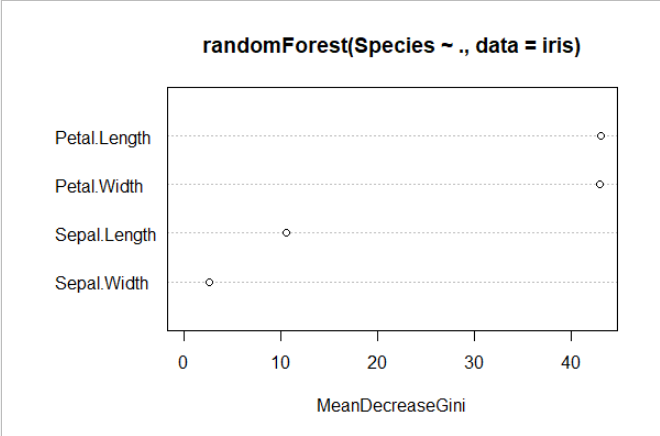
plot(Boruta(Sales~.,data = advertising))



Randomforest değişken önemi :

randomForest(Species~.,data = iris)

varImpPlot(randomForest(Species~.,data = iris))



Boruta(Species~.,data = iris,doTrace=3)

plot(Boruta(Species~.,data = iris,doTrace=3))

borutayı her modele yapabilirsin borutada da aynı sonucu verdi.

Bu importance fonksiyonu da randomforest algoritmasının içerisinde.

> m=randomForest(Species~.,data = iris)

> importance(m,data=iris)

MeanDecreaseGini

Sepal.Length 10.478885

Sepal.Width 2.389879

Petal.Length 43.351059

Petal.Width 43.057050

Caret kütüphanesindeki varImp fonksiyonu.

> varImp(m,data=iris)

Overall

Sepal.Length 10.478885

Sepal.Width 2.389879

Petal.Length 43.351059

Petal.Width 43.057050

VarImp ve importance random forest için

Svm için

s=fit(Species~.,data = iris,model = "svm")

imp=Importance(s,data = iris)

boruta değişken seçiminde yeşil kutular confirmed kabul edilen sarı kutular tentetive geçici olanlar kırmızı kutular unconfirmed kabul edilmeyen değişkenlerdir.

> getSelectedAttributes(Boruta(NewLeague~.,data = df\_hitters))

[1] "AtBat" "Runs" "CAtBat" "CHits" "League" "Salary"

Getselect önemli değişkenleri verir.

Attribution stats ise değişkenlerin önem düzeylerini verir.

> attStats(Boruta(NewLeague~.,data = df\_hitters))

meanImp medianImp minImp maxImp normHits decision

AtBat 3.8518759 3.9147931 0.73251216 6.4167534 0.78787879 Confirmed

Hits 2.7528452 2.9060131 0.28449971 5.0899951 0.58585859 Tentative

HmRun 2.8852466 2.9810477 1.00267327 4.8920235 0.61616162 Tentative

Runs 5.2502518 5.2891251 2.72709692 7.1395438 0.96969697 Confirmed

RBI 3.3143553 3.4412400 0.86734559 6.2695097 0.68686869 Confirmed

Walks 2.7980639 2.8310473 -0.48948962 5.0561347 0.61616162 Tentative

Years 0.9700790 1.3925407 -1.25739657 2.4184341 0.00000000 Rejected

CAtBat 3.6979941 3.7416483 0.93219235 6.6467868 0.76767677 Confirmed

CHits 3.7984374 3.8233595 0.78509212 6.6377755 0.79797980 Confirmed

CHmRun 2.6124866 2.7286144 -0.74360156 5.5287842 0.52525253 Tentative

CRuns 3.2750531 3.2478884 0.63901128 6.0677511 0.69696970 Confirmed

CRBI 3.3100152 3.2867245 -0.01959474 5.6439650 0.72727273 Confirmed

CWalks 1.8094594 1.8831590 0.28958160 3.2263071 0.03030303 Rejected

League 49.8930987 50.4588901 42.41031992 55.6307289 1.00000000 Confirmed

Division -0.7664839 -0.8635304 -1.80670563 0.4545372 0.00000000 Rejected

PutOuts 3.0689965 3.1505450 0.22795810 5.5661717 0.63636364 Tentative

Assists -0.4823977 -0.3136419 -2.84710045 1.2496632 0.00000000 Rejected

Errors 1.0728453 0.9108966 -0.88979293 2.6709282 0.02020202 Rejected

Salary 3.7783660 3.7913524 1.25553257 6.4070121 0.78787879 Confirmed

En önemli değişken League

**DEĞİŞKEN SEÇİMİ ÖRNEK PROJE:**

> summary(lm(Age~.,data = LungCapData2))

Call:

lm(formula = Age ~ ., data = LungCapData2)

Residuals:

Min 1Q Median 3Q Max

-5.1278 -1.0781 -0.1987 0.9526 8.0142

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -5.70255 1.21466 -4.695 3.26e-06 \*\*\*

LungCap 0.34814 0.05043 6.904 1.21e-11 \*\*\*

Height 0.25370 0.02299 11.036 < 2e-16 \*\*\*

Gendermale -0.58301 0.13290 -4.387 1.34e-05 \*\*\*

Smokeevet -1.81728 0.22600 -8.041 4.23e-15 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.646 on 649 degrees of freedom

Multiple R-squared: 0.6914, Adjusted R-squared: 0.6895

F-statistic: 363.6 on 4 and 649 DF, p-value: < 2.2e-16

Yukarıdaki linear model çıktısı

Şimdi boruta ile tahminleme yapacağım.

> Boruta(Age~.,data = LungCapData2)

Boruta performed 9 iterations in 1.205771 secs.

4 attributes confirmed important: Gender, Height, LungCap, Smoke;

No attributes deemed unimportant.

\*\*\* bütün değişkenler önemli

> boruta=Boruta(Age~.,data = LungCapData2)

> getSelectedAttributes(boruta)

[1] "LungCap" "Height" "Gender" "Smoke"

\*\*\* seçilmesi gereken değişkenler.

> attStats(boruta)

meanImp medianImp minImp maxImp normHits decision

LungCap 29.537211 29.406014 27.708991 31.999949 1.0000000 Confirmed

Height 38.546852 38.638174 36.482126 40.779707 1.0000000 Confirmed

Gender 5.704314 5.768917 3.696013 8.584266 0.8421053 Confirmed

Smoke 24.564843 24.412477 22.077035 26.608601 1.0000000 Confirmed

Değişken önem düzeyleri.

Random forest örneği:

> random\_model=randomForest(Age~.,data = LungCapData2)

> importance(random\_model)

IncNodePurity

LungCap 1320.02387

Height 1397.11333

Gender 28.24841

Smoke 394.47743

Veri setine eger doğrusal model yapamıyorsan o zaman eksik veri yoksa eger direkt boruta fonksiyonunu uygulayıp (Bir vektördeki önemli değişkenlerin son bir listesini kaydetmek için getSelectedAttributes () işlevini kullanın .) getselect ile önemli değişkenlerin listesini al. Sınıflandırma ve regresyon problemlerinde çalışır.

**SVM PROJESİ:**

> summary(glm(vites~.,data = arabaveri,family = "binomial"))

Call:

glm(formula = vites ~ ., family = "binomial", data = arabaveri)

Deviance Residuals:

Min 1Q Median 3Q Max

-1.91154 -0.29298 -0.03574 0.21541 1.96021

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) 23.92836 14.17738 1.688 0.0915 .

yol -0.09851 0.35135 -0.280 0.7792

silindir6 3.00979 2.51067 1.199 0.2306

silindir8 4.98194 3.50934 1.420 0.1557

ağırlık -8.17801 3.34965 -2.441 0.0146 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 43.230 on 31 degrees of freedom

Residual deviance: 14.588 on 27 degrees of freedom

AIC: 24.588

Number of Fisher Scoring iterations: 7

Görüldüğü gibi model doğrusal değil bu yüzden svm algoritmasını deneyeceğim.

Ama öncelikle boruta paketiyle hangi özellikleri seçececeğimi belirleyeceğim.

> Boruta(vites~.,data = arabaveri)

Boruta performed 99 iterations in 1.143531 secs.

2 attributes confirmed important: ağırlık, yol;

No attributes deemed unimportant.

1 tentative attributes left: silindir;

Ağırlık ve yol anlamlı değişkenlerim. Silindir geçici değişkenim aslında alabilirim ama almayacağım.

> getSelectedAttributes(Boruta(vites~.,data = arabaveri))

[1] "yol" "ağırlık"

Bunlarda seçeceğim değişkenler.

> attStats(Boruta(vites~.,data = arabaveri))

meanImp medianImp minImp maxImp normHits decision

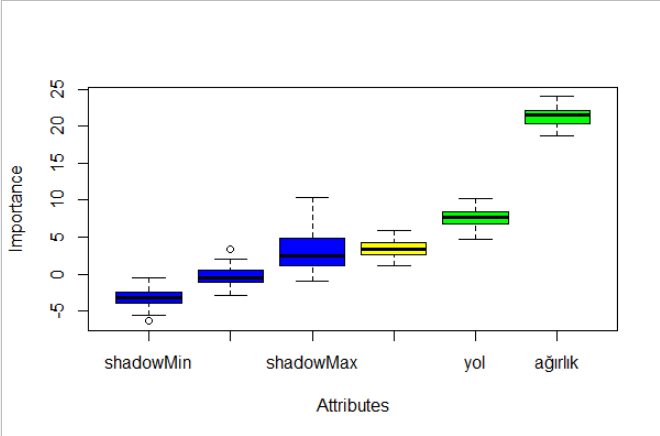
yol 7.364275 7.582570 4.7936561 10.402520 0.9555556 Confirmed

silindir 3.198753 3.281967 0.9451126 5.154371 0.7111111 Confirmed

ağırlık 21.573316 21.491973 18.5503214 24.183811 1.0000000 Confirmed

değişken önem düzeyleri en önemli değişkenim ağırlık sonraki yol.

plot(Boruta(vites~.,data = arabaveri))

silindir sarı renkte geçici olarak kabul görür.

svm(vites~yol+ağırlık,data=arabaveri)

eger silindir değişkenini modele eklemez isem

> defaultSummary(data.frame(obs=arabaveri$vites,

+ pred=predict(svm(vites~yol+ağırlık,data=arabaveri),

+ arabaveri[,c(1,3)])))

Accuracy Kappa

0.9062500 0.8032787

Doğruluk oranım % 91

> defaultSummary(data.frame(obs=arabaveri$vites,

+ pred=predict(svm(vites~.,data=arabaveri))))

Accuracy Kappa

0.8437500 0.6638655

Silindir değişkenini modele eklersem modelin doğruluk oranı %84’e iner.

> table(tahmin=predict(svm(vites~yol+ağırlık,data=arabaveri),

+ arabaveri[,c(1,3)]),

+ gerçek=arabaveri$vites)

gerçek

tahmin otomatik manuel

otomatik 18 2

manuel 1 11

otomatiği 1/19 %5 hata ile tahmin ediyor.

Manueli ise 2/13 %15 hata ile tahmin ediyor.

> random\_model=randomForest(vites~yol+ağırlık)

> random\_model

Call:

randomForest(formula = vites ~ yol + ağırlık)

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 1

OOB estimate of error rate: 12.5%

Confusion matrix:

otomatik manuel class.error

otomatik 17 2 0.1052632

manuel 2 11 0.1538462

random forest algoritmasında ise %87.5 dogruluk oranı var svm’den daha az ve otomatiği %10 hata ile manueli ise %15 hata ile doğru tahmin ediyor.

**Ölçeklendirme Neden Yapılır.**

**Neden Ölçeklendirme**

Veri kümeniz çoğu zaman büyüklük, birim ve aralık bakımından oldukça değişken özellikler içerecektir. Ancak, makine öğrenimi algoritmalarının çoğu hesaplamalarında iki veri noktası arasındaki Öklediyen mesafesini kullandığından, bu bir sorundur.

Yalnız bırakılırsa, bu algoritmalar yalnızca birimleri ihmal eden özelliklerin büyüklüğünü alır. Sonuçlar, 5kg ve 5000gms gibi farklı birimler arasında büyük farklılıklar gösterecektir. Mesafe hesaplamalarında yüksek büyüklükteki özellikler, düşük büyüklükteki özelliklere göre çok daha ağır olacaktır.Bu etkiyi bastırmak için, tüm özellikleri aynı büyüklük düzeyine getirmemiz gerekiyor. Bu, ölçeklendirme ile elde edilebilir.

**Ne Zaman Ölçeklendirilmeli**

Burada uyguladığım temel kural, mesafeyi hesaplayan veya normallik varsayan herhangi bir algoritmadır. **özelliklerinizi ölçeklendirin !!!**

Özellik ölçeklemenin önemli olduğu bazı algoritma örnekleri şunlardır:

Öklid mesafe ölçüsü olan **k-en yakın komşular** büyüklüklere duyarlıdır ve bu nedenle tüm özelliklerin eşit olarak tartılması için ölçeklenmelidir.

**Temel Bileşen Analizi (PCA)** gerçekleştirilirken ölçeklendirme çok önemlidir . PCA, özellikleri maksimum varyansla elde etmeye çalışır ve yüksek büyüklükteki özellikler için varyans yüksektir. Bu, PCA'yı yüksek büyüklük özelliklerine doğru çarpıtır.

**Gradyan inişini** ölçeklendirerek hızlandırabiliriz . Bunun nedeni, θ değerinin hızlı bir şekilde küçük aralıklarda ve büyük aralıklarda yavaşça alçalması ve dolayısıyla değişkenler çok düzensiz olduğunda optimum seviyeye verimsiz bir şekilde salınmasıdır.

**Ağaç tabanlı modeller** , mesafeye dayalı modeller değildir ve çeşitli özellik aralıklarını işleyebilir. Bu nedenle ağaçları modellerken Ölçeklendirme gerekli değildir.

**Lineer Discriminant Analysis (LDA), Naive Bayes** gibi algoritmalar **,** tasarım gereği bunu idare edecek şekilde donatılmıştır ve buna göre özelliklere ağırlık verir. Bu algoritmalarda bir özellik ölçeklendirme gerçekleştirmenin pek bir etkisi olmayabilir.

Umarım özellik ölçeklemenin **neden** , **nasıl** ve **ne zaman yapıldığını** anlamışsınızdır.

Ağaç tabanlı algoritmalarda ölçeklendirmeye gerek yoktur. Bayes dahil.

Fakat mesafe ölçümü yapan algoritmalarda kesinlikle ölçeklendirmek gerekir.

Örneğin knn veya svm veya yapay sinir ağı algoritmasını kullanacaksak kesinlikle ölçeklendirmemiz gerekir.

Ölçeklendirdikten sonra modelin ne kadar doğru tahmin yaptığını anlarız.

Scale fonksiyonu demek bildiğimiz normalleştirme demek x-xort/sd

Knn ölçeklendirilmiş ve ölçeklendirilmemiş tahminin doğruluğu:

A veri seti scale edilmiş

Advertising veri seti scale edilmeiştir.

> defaultSummary(data.frame(obs=advertising$Sales,

+ pred=as.numeric(as.vector(knn(cbind(TV,Radio,Newspaper),advertising[,c(1:3)],

+ advertising$Sales,k=2)))))

RMSE Rsquared MAE

10.90508826 0.04705647 9.61050000

> defaultSummary(data.frame(obs=a$Sales,

+ pred=knn\_fit))

RMSE Rsquared MAE

0.3212033 0.9015228 0.1966353

Scale edilmiş veri seti %90 tahmin gücüne sahip.

En sonunda yeni veri geldiğinde tahminlemeyi scale edilmemiş model üzerinden yap.

Doğrusal olmayan modelde düz bir eğri uydurulmadığından değişkenlerin p değerleri olmaz.

Doğrusal olmayan modelde katsayıda olmaz.

TentativeRoughFix(Boruta(Sales~.,data = advertising))

Bu fonksiyon geçici değişkenlerin anlamlılık kazanıp kazanmayacağını test eder.

**Normalizasyon** makine öğrenmesi projelerinde data hazırlama aşamasında yaygın olarak kullanılan bir işlemdir. Normalizasyonun amacı veri setindeki sayısal değişkenlerin değerlerini, değerler aralığındaki farklılıkları bozmadan ortak bir ölçeğe dönüştürmektir. Her veri setinin normalizasyon işlemine ihtiyacı yoktur. Sadece farklı aralıkta dağılan sayısal değişkenleri olduğunda gereklidir.

Örneğin, yaş (x1) ve gelir (x2) olmak üzere iki özellik içeren bir veri kümesini düşünün. Yaş 0–100 arasında değişirken, gelir 20,000 -500.000 arasında değişmektedir. Bu iki özellik çok farklı aralıklardadır. Örneğin, çok değişkenli doğrusal regresyon gibi bir analiz yaptığımızda, gelir değişkeni daha büyük değeri nedeniyle sonucu daha fazla etkileyecektir. Ancak bu durum gelir değişkenin daha önemli bir tahminleyici olduğu anlamına gelmez. Bunu önlemek amacıyla Normalizasyon işlemi yapmamız gerekmektedir.

Normalizasyon işlemi sonucunda tüm sayısal veriler 0 ile 1 arasında değer alacaktır. Eğer negatif veriler varsa -1 ile 1 arasında dağılacaktır.

Normalizasyon formülü (x-minx)/(maxx-minx)

Z puanı z skoru (x-xort)/sd(x) \*\*\* önemli.

Eger veri setindeki değişkenler arasındaki fark aşırı yüksek ise normalizasyon yap değişkenlerin katsayıları ve rsquare si değişmiyor. Lungcap data seti değişkenler arasındaki fark çok fazla açık değildi normalizasyon yapmama rağmen sonuç aynı cıktı.

Dönüşüm türleri:

Normalizasyon min max dönüşümüdür.

Standartizasyon ort standart sapma dönüşümü.

Analiz edebilmemiz için değişkenlerin birbirlerine yakın olması gerekiyor örneğin tecrübe 1-10 arasında değişirken maaş 2000-10000 arasında değişiyorsa tecrübedeki değişim neredeyse maaştaki değişimi hiç etkilemeyecek bu yüzden veriler birbirlerine yaklaştırılmalı yani scale yada normalize edilmeli.

Veriler birbirleriyle aynı aralıkta ifade edilmeli.

Normalizasyon işlemleri arasında bir fark yoktur.

apply(LungCapData2[,c(1:3)],2,normalizas)

bu şekilde yap. Hepsini seçip normalizas(lungcapdata) deme bu yanlış.

Değişken önemi:

Boruta da olabilir bu farklı bir yöntem.

library(caret)

set.seed(100)

rPartMod <- train(Class ~ ., data=trainData, method="rpart")

rpartImp <- varImp(rPartMod)

print(rpartImp)

Eger regresyon modelinde vif değeri 5 ten veya 10 dan büyük ise o değişkeni modelden çıkar yoksa çoklu bağlantı problemi yaşarsın.

vif için faraway paketini kullan.

> vif(model)

unemployed femlab marriage birth military

2.252888 3.613276 2.864864 2.585485 1.249596

Çoklu bağlantı ve aşırı uyum sorununu önlemek için ridge regresyon kullan.

Tüm korelasyonları görmek için

pairs.panels(mydata)

library(psych)

k-means ve knn algoritmaları mesafeye dayalı olduğu için scale kullan. Ve bu algoritmalarda sadece numeric değişkenleri kullan.

Eger ölçeklendirmezsen değeri büyük olan değişken etrafında kümelenir. Tek bir değişken tüm değişkenlere hakim olur.

> sd=apply(advertising,2,sd)

> mean=apply(advertising,2,mean)

> rbind(sd,mean)

TV Radio Newspaper Sales

sd 85.85424 14.84681 21.77862 5.283892

mean 147.04250 23.26400 30.55400 15.130500

varyanslar veya standart sapmalar birbirlerinden çok uzaklarsa ölçeklendirme yapıyorum tabi bunu uzaklığı dikkate alan algoritmalar için yapıyorum.

Cluster kümeleme demek.

**BAYES ALGORİTMASI**

**#Bayes Teoremi**

yas\_9\_evet=nrow(filter(LungCapData2,Age==9,Smoke=="evet"))/length(LungCapData2$Age)

ak\_8\_9\_icen=nrow(filter(LungCapData2,LungCap<=9,LungCap>=8,Smoke=="evet"))/length(LungCapData2$Age)

bayes\_olasılığı=(yas\_9\_evet\*ak\_8\_9\_icen)/(nrow(filter(LungCapData2,Smoke=="evet"))/length(LungCapData2$Smoke))

**#YASI GENC KREDİ SKORU YUKSEK GELİRİ ORTA BİRİSİNİN BİLGİSAYAR ALMA OLASILIĞI NEDİR**

yas\_genc=NROW(filter(data,YAS=="GENC",BILGISAYAR\_ALDI=="EVET"))/length(data$BILGISAYAR\_ALDI)

kredi\_yuksek=NROW(filter(data,KREDI\_SKORU=="YUKSEK",BILGISAYAR\_ALDI=="EVET"))/length(data$KREDI\_SKORU)

gelir\_orta= NROW(filter(data,GELIR=="ORTA",BILGISAYAR\_ALDI=="EVET"))/length(data$GELIR)

BİL\_ALMA\_BAYES=(kredi\_yuksek\*gelir\_orta\*yas\_genc)/(NROW(filter(data,BILGISAYAR\_ALDI=="EVET"))/length(data$BILGISAYAR\_ALDI))

df=df[df$product!="NULL",]

#iki fonksiyon da aynı benzersiz değerleri kaldırır.

df2=df2[!base::duplicated(df2),]

df2=unique(df2)

> paste(c("harun","bakırcı","samsun","istanbul"),collapse = "-")

[1] "harun-bakırcı-samsun-istanbul"

a[grep(1,a)]="kötü"

a[grep(2,a)]="orta"

a[grep(3,a)]="iyi"

a[grep(4,a)]="çokiyi"

b=factor(b,levels = 1:4,labels = c("ilkokul","lise","üniversite","doktora") )

hitters’dan Hits grubuna 7 tane boş veri ata yani var olanla değiştir.

hitters[sample(1:nrow(hitters),7),"Hits"]=NA

hitters[sample(1:nrow(hitters),2),"Runs"]=NA

hitters[sample(1:nrow(hitters),10),"RBI"]=NA

> hitters[!complete.cases(hitters),]

Kayıp veri olan değerleri verir yani her satırda illa bir kayıp veri var.

> hitters[complete.cases(hitters),]

Dolu verileri getirir.

select(inner\_join(urun\_detay,urun\_ıd),ID,urun\_adı)

eger doğruluk yüksekse ozellik seçimine gerek yok.

Youtube dr cevabı:

Denemekten zarar gelmez. Doğruluğu artırmaya yardımcı oluyorsa, kullanmalıdır. Doğruluk gelişmezse, göz ardı edilebilir.

c[sample(1:30,8)]=NA c vektörünün rastgele 8 değişkenine NA atadım.

> a[!is.na(c),] c sütunundaki eksik verileri sildim yani c sutununda eksik olmayan verileri seç.

b c

2 11 56 eger na.omit(a) deseydim butun değişkenlerdeki

3 155 25 eksik verileri silerdi.

5 165 37

6 146 60

7 153 12

8 87 85

9 94 86

10 121 62

11 197 94

13 114 85

> tablo[tablo$kişi=="harun" & tablo$not>40 & tablo$Yaş>25,]

kişi Yaş cins krediler GNO not sonuc

6 harun 30 erkek 3 3.50 70 geçti

11 harun 27 erkek 2 2.15 55 kaldı

19 harun 29 erkek 5 3.70 90 geçti

> a[is.na(c),] c sütunundaki eksik verileri göster

Eger ! kullansaydım c sütununda eksik olmayan verileri göster anlamına gelirdi.

b c

1 186 NA

4 169 NA

12 59 NA

16 49 NA

17 26 NA

18 61 NA

19 93 NA

26 111 NA

normalde runs değişkeninde de eksik veri vardı fakat ben onu sayma dedim.

> sapply(hitters[!is.na(hitters$Runs),],function(x) sum(is.na(x)))

AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun

0 7 0 0 9 0 0 0 0 0

CRuns CRBI CWalks League Division PutOuts Assists Errors Salary NewLeague

0 0 0 0 0 0 0 0 59 0

Hitters veri setinde runs ve salary değişkeninden eksik verileri al gerisi kalsın.

> hitters[!is.na(hitters$Runs) & !is.na(hitters$Salary),]

| veya anlamında.

> LungCapData2[LungCapData2$Gender=="female" | LungCapData2$Age>10,]

Oransal olarak eksik veri:

> round(sapply(hitters,function(x) sum(is.na(x))/length(x)),2)

AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits CHmRun

0.00 0.02 0.00 0.03 0.03 0.00 0.00 0.00 0.00 0.00

CRuns CRBI CWalks League Division PutOuts Assists Errors Salary NewLeague

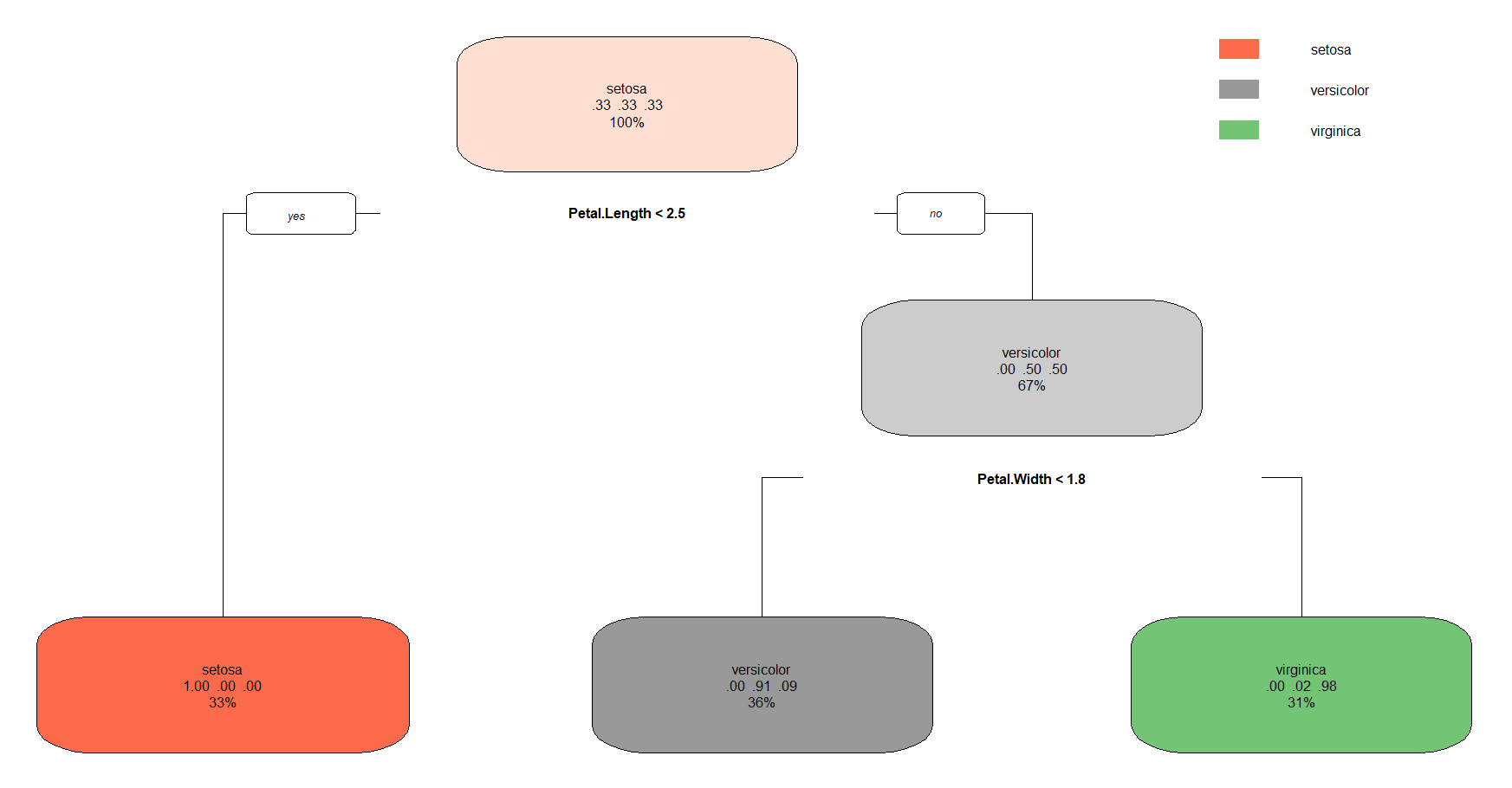
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.18 0.00

Eksik veri %20’nin üzerindeyse o değişkeni silebilirsin.

Rpart fonksiyonu ile de random forest modeli oluşturabilirsin. rpart.plot ile rpart.plor(model) yazarsan buyuk bir ağaç oluşur.

> model=rpart(Species~.,data = iris)

> rpart.plot(model)



> a=c()

> for(i in 1:length(hitters$Salary)){

+ if(is.na(hitters$Salary[i])==TRUE){

+ a=c(a,i)

+ }

+ }

>

> a

[1] 1 16 19 23 31 33 37 39 40 42 43 45 49 53 58 65 67 70 72 78 81 84 95 98 102

[26] 104 105 106 107 115 126 139 145 151 158 159 161 170 172 174 198 200 204 209 211 226 229 236 247 251

[51] 254 255 271 284 293 299 303 306 317

Maaş değişkenindeki boş veri satırları bu şekilde de bulunabilir.

> hitters[!is.na(hitters$Salary),]

Eksik verileri medyan ile doldurmak:

Medyan ile doldurdum çünkü maaş değişkeni normal dağılmıyordu.

> for(i in 1:length(hitters$Salary)){

+ if(is.na(hitters$Salary[i])==TRUE){

+ hitters$Salary[i]=median(hitters$Salary,na.rm = T)

+ }

+ }

Kişi bazında eksik veri:

> data.frame(eksik\_veri=apply(hitters,1,function(x) sum(is.na(x))/length(x)))

eksik\_veri

-Andy Allanson 0.05

-Alan Ashby 0.00

-Alvin Davis 0.00

-Andre Dawson 0.00

-Andres Galarraga 0.00

-Alfredo Griffin 0.00

-Al Newman 0.00

-Argenis Salazar 0.05

hitters$eksik=a eksik verileri değişken olarak ekledim.

Eksik veri değeri %10’dan küçük olanları seç.

> hitters[hitters$eksik<0.10,]

> colSums(is.na(hitters))

AtBat Hits HmRun Runs RBI Walks Years CAtBat CHits

0 7 0 9 10 0 0 0 0

CHmRun CRuns CRBI CWalks League Division PutOuts Assists Errors

0 0 0 0 0 0 0 0 0

Salary NewLeague eksik\_veri

59 0 0

hitters$Salary=NULL salary değişkenini siler.

select(hitters,-c(Salary))

> hitters$RBI[is.na(hitters$RBI)]

[1] NA NA NA NA NA NA NA NA NA NA

> which(tablo$kişi=="harun")

[1] 6 11 16 19

> which(tablo$kişi=="harun" & tablo$sonuc=="geçti")

[1] 6 16 19

Which satır bulur. İndeks bulur.

Hangi değerlerin NULL olduğunu buldurdum.

> sapply(hitters,function(x) which(is.na(x)))

$AtBat

integer(0)

$Hits

[1] 25 89 177 185 230 242 258

$HmRun

integer(0)

$Runs

[1] 127 194 203 212 238 244 262 296 300

Listeye madde ekleme.

> a=list(ad=c("harun","muhammed"),

+ şehir=c("samsun","istanbul","ankara"))

> b=readline("lütfen bir şehir giriniz:")

lütfen bir şehir giriniz:sivas

> a$şehir=c(a$şehir,b)

> a

$ad

[1] "harun" "muhammed"

$şehir

[1] "samsun" "istanbul" "ankara" "sivas"

Sapply liste döndürür.

Normalleştirme çeşitleri.

1-)Min-Max Normalizasyon

2-)Decimal Scaling Yöntemi

3-) Z Score Normalizasyon

Library(Rcmdr) r commander’ı çalıştırır.

**KOLMOGROV SMİRNOV TESTİ**

Gözlem sayısı eger 30’un üzerindeyse normallik varsayımı için bu testi kullan.

Sub komutu:

Aranılan, yerine, konulan, veri seti.

a=LungCapData2$Smoke

sub("evet","içiyor",a)

boşluk görünce ayır.

> a="harun bakırcı samsun istanbul başakşehir istanbul bakırcı istanbul"

> b=strsplit(a," ")

> table(b)

b

bakırcı başakşehir harun istanbul samsun

2 1 1 3 1

“-“ görünce ayır.

> a="merhaba-arkadaşlar-şimdi-sizlere-veri-setimi-tanıttıracağım"

> b=strsplit(a,"-")

> table(b)

b

arkadaşlar merhaba setimi sizlere şimdi tanıttıracağım veri

1 1 1 1 1 1 1

Which ile veri dönüşümü:

a=LungCapData2$Smoke

a=as.character(a)

a[which(a=="evet")]=1

a[which(a=="hayır")]=2

a=as.numeric(a)

aynı işlem:

a=LungCapData2$Smoke

a=as.character(a)

a[a=="evet"]=1

a[a=="hayır"]=2

a=as.numeric(a)

edit komutuyla açılan frame’de veri girişi yapabilirsin.

> a=edit(a)

> a

Kitap\_Adi Adet

1 harun 10

2 berk 8

3 mahmut 17

4 sadık 16

**ANCOVA ÖR:**

result <- aov(mpg~hp\*am,data = input)

print(summary(result))

mpg ve hp numeric am vites türü 0 1 kategorik değişkendir.

Milin yani mpg’nin vites türü üzerindeki etkisini beygiri yani hp’yi kontrol altına alarak test ediyorum.

**Df Sum Sq Mean Sq F value Pr(>F)**

**hp 1 678.4 678.4 77.391 1.50e-09 \*\*\***

**am 1 202.2 202.2 23.072 4.75e-05 \*\*\***

**hp:am 1 0.0 0.0 0.001 0.981**

**Residuals 28 245.4 8.8**

**---**

**Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’**

Bu sonuç, her iki durumda da p değeri 0,05'ten az olduğundan, hem beygir gücü hem de şanzıman türünün galon başına mil üzerinde önemli etkiye sahip olduğunu göstermektedir. Ancak bu iki değişken arasındaki etkileşim, p değeri 0.05'ten fazla olduğu için anlamlı değildir.

Spss geneal linear model -> univariate ancova demek

Spss geneal linear model -> multivariate manova mancova

**EKSİK VERİ**

**apply(hitters,2,function(x) round(sum(is.na(x))/length(x),2))**

**degisken\_eksik=data.frame(eksik=apply(hitters,2,function(x) round(sum(is.na(x))/length(x),2)))**

**filter(degisken\_eksik,eksik!=0)**

**apply(hitters,1,function(x) round(sum(is.na(x))/length(x),2))**

**satır\_eksik=data.frame(eksik=apply(hitters,1,function(x) round(sum(is.na(x))/length(x),2)))**

**filter(satır\_eksik,eksik!=0 & eksik>0.05)**

RBI değişkenini boş bırkanları silmek istersen boş bırakanların listesi:

which(is.na(hitters$RBI))

ordered:

ses.order <- **ordered**(ses, levels = **c**("low", "middle", "high"))

factor veya numeric veya integer olan değişkenleri bulmak:

> if(is.factor(hitters[,i])){

+ print(i)

+ }

> for(i in 1:21){

+ if(is.factor(hitters[,i])){

+ print(i)

+ }

+ }

[1] 14

[1] 15

[1] 20

Aykırı değerler destek vektör makinelerinin tahminleme aşamasında büyük öneme sahip.

Tek değişkenli aykırı değişkenleri bulmak için en güzel metodlar:

* Boxplot
* Histogram
* Standart sapma
* Standart normal dağılım

Standart sapmadan 1 2 3 standart sapma uzaklaşan değerlere aykırı değer diyebiliriz.

Örneğin elimizde Tofaş araçlarla ilgili bir veri seti var. Bir tane tofaşımız bir millet vekilimizin ismine ait plakaya sahip olsun ve bu aracın fiyatı 100000 tl olsun şimdi bizim bu değişkeni bu modelde kullanmamıza gerek yok modeli bozar. Milyonda bir gözlenecek bir olayı normalmiş gibi kabul edip veri setinin içerisinde tutmanın bir anlamı yoktur.

Birden fazla değişkeni sınıfını değiştirmek:

apply(b,2,function(x) as.integer(x))

mutate\_all(b,scale) alttaki ile aynı.

apply(b,2, function(x) scale(x))

**AYKIRI DEĞER**

aykırı değer tek değişkenli ise boxplot ile bulabiliriz. Boxplotun dışarısında kalan değerler aykırı değerdir.

Aykırı değer tespiti:

aykırı\_deger=boxplot.stats(a)$out

aykırı değerlerin değerlerini gösterir. İndekslere erişmek için:

which(a%in%boxplot.stats(a)$out) aykırı değer indekslerini verir.

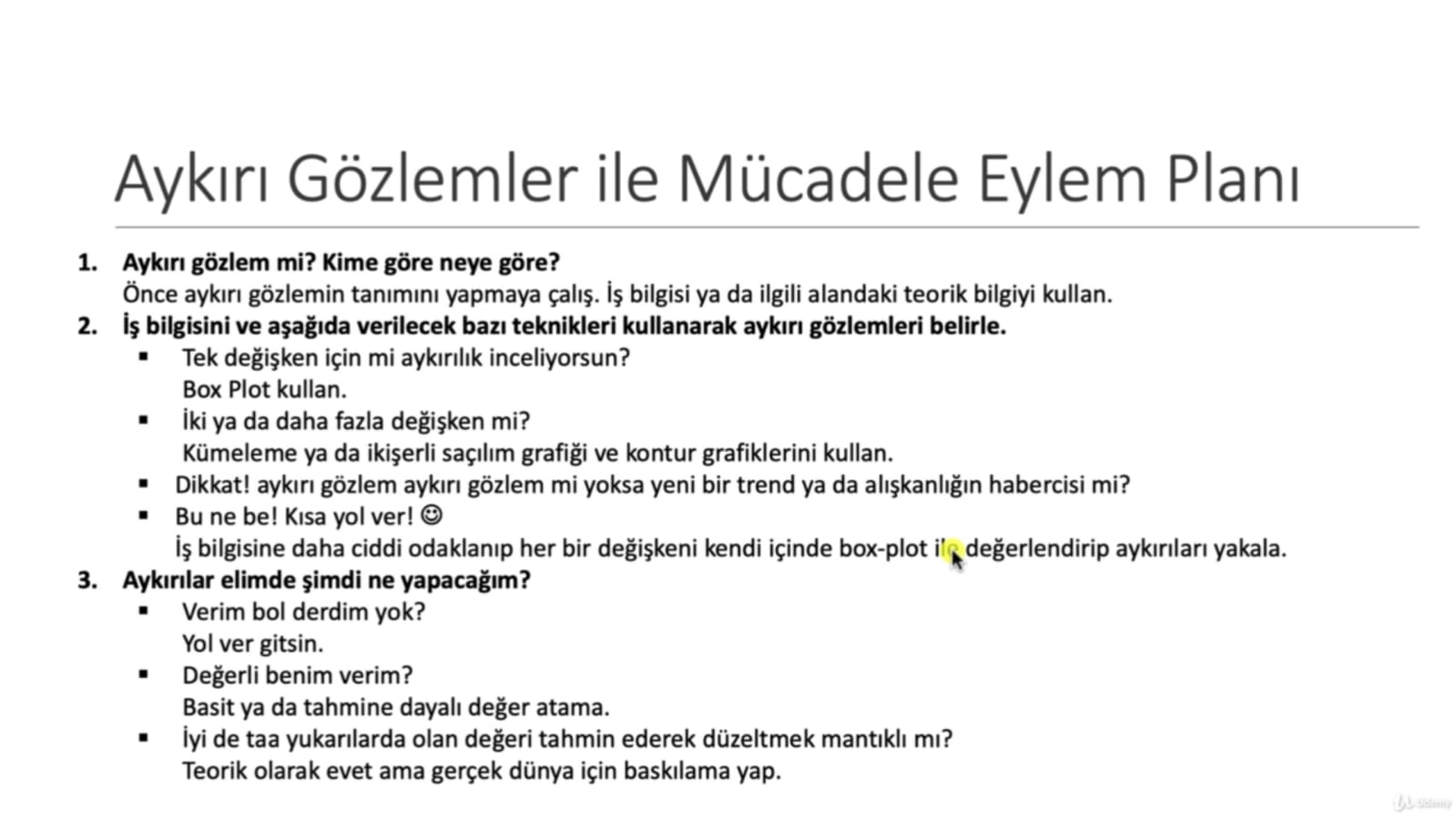
For dongusu ile de bulunabilir.

for(i in aykırı\_deger){

print(which(a==i))

}

Eger gözlem sayım çok fazla ise silebilirim ortalama medyan atayabilirim ya da tahminleme yapabilirim.



Eksik verilerin ortalama ile doldurulması.

a$tecrube[which(is.na(a$tecrube))]=mean(tecrube)

mcnemar testi bağımlı kategorik verileri kıyaslar

mcnemar.test(ilk,son)

paired sample ise bağımlı nicel verileri kıyaslar

chisq.test ise iki kategorik değişken arasındaki ilişkiyi inceler.

**ÇOK ÖNEMLİİ**

****

**K-means algoritması**

#k-means algoritması

library(cluster)

library(tidyverse)

library(factoextra)

k\_mean\_data=USArrests

head(k\_mean\_data)

k\_mean\_data=scale(k\_mean\_data)

apply(USArrests,2,function(x) sd(x))

distance=get\_dist(k\_mean\_data)

# eyaletler arası uzaklıklara erişebiliyoruz.

fviz\_dist(distance)

#centers'ı yani küme sayısını kendimiz atıyoruz.

k2=kmeans(k\_mean\_data,centers = 2,nstart = 25)

attributes(k2)

as.data.frame(k2$cluster)

fviz\_cluster(k2,data = k\_mean\_data)+theme\_classic()

#gorsel çalışmaz veri setini değiştirdim geomlabel mantıgına bak

ggplot(k\_mean\_data,aes(UrbanPop,Murder,color=factor(kumeler),label=eyalet))+geom\_label()+theme\_classic()

head(USArrests)

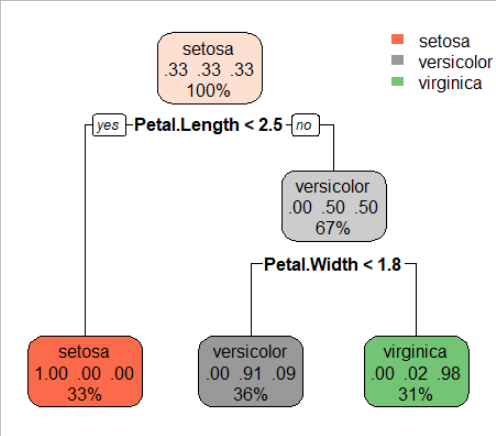
apply(USArrests[,c(1,3)],2,function(x) shapiro.test(x))

cor.test(USArrests$Murder,USArrests$UrbanPop,method = "pearson")

1.sonucta %100 setosa diye tahminler

2.sonucta %91 versicolor %9 virginica diye tahminler

3.sonucta %98 virginica %2 versicolor diye tahmin ediyor.



Aykırı değer bul:

for(i in 1:100){

for(j in boxplot.stats(a$yas)$out){

if(a$yas[i]==j){

p=c(p,which(a$yas==j))

}

}

}

Yukarıdaki ile aynı.

for(i in boxplot.stats(a$yas)$out){

print(which(a$yas==i))

}

**Knn validasyon:**

ctrl=trainControl(method = "cv",

number = 10)

knn\_val=train(x=l[,c(3,4,5)],

y=l$egitim,

trControl=ctrl,

method="knn",

preProc=c("scale","center"))

linear regresyon olsa method yerine “lm” yaz.

> importance(randomForest(a[,-c(1,2)],a$egitim,importance = TRUE))

ilkokul lise üniversite MeanDecreaseAccuracy MeanDecreaseGini

tecrube 1.068081 4.928601 10.049469 9.25155366 16.091006

yas 6.634742 5.903908 12.252805 14.39233841 20.164051

maas 9.788698 6.817878 3.144353 10.97403818 24.719927

cinsiyet 2.681035 -6.473414 2.908815 -0.06426885 3.766426

**Svm validasyon:**

svm\_tune=train(p[,c(1,2,4)],p$price,

trControl=ctrl,

preProc=c("center","scale"),

method="svmRadial",

tuneLength=14)

**Lm validasyon:**

lm\_val\_fit=train(x=arabalar[,c(1,2,4)],

y=arabalar$price,

trControl=ctrl,

method="lm")

defaultSummary(data.frame(obs=arabalar$price,

pred=predict(lm\_val\_fit,

arabalar[,c(1,2,4)])))

**xgboost algoritması:**

#eta öğrenme oranı.

#nrounds verinin üzerinden geçme sayısı. Agac sayısı da olabilir.

#iteration zaten yenilemek demek.

#nrounds'u 2 yaptığımız için 2 tur döner.

#nrounds yani iterasyon sayısı arttıkça hatamız azalıyor.

xgboost\_fit=xgboost(data = as.matrix(advertising[,-4]),

label = advertising$Sales,

booster="gblinear",

max.depth=2,

eta=1,

nthread=2,

nrounds =42)

predict(xgboost\_fit,as.matrix(advertising[,-4]))

defaultSummary(data.frame(obs=advertising$Sales,

pred=predict(xgboost\_fit,as.matrix(advertising[,-4]))))

imp\_xgb=xgb.importance(model = xgboost\_fit)

xgb.plot.importance(imp\_xgb)

**MANCOVA:**

> summary.aov(aov(cbind(arabalar$price,arabalar$mileage)~model\*year,data=arabalar))

Response 1 :

Df Sum Sq Mean Sq F value Pr(>F)

model 2 276375070 138187535 72.8579 <2e-16 \*\*\*

year 12 915781448 76315121 40.2363 <2e-16 \*\*\*

model:year 8 19700153 2462519 1.2983 0.2502

Residuals 127 240877265 1896671

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Response 2 :

Df Sum Sq Mean Sq F value Pr(>F)

model 2 9.0183e+09 4509126112 18.4276 9.404e-08 \*\*\*

year 12 6.7080e+10 5590016581 22.8449 < 2.2e-16 \*\*\*

model:year 8 1.3025e+09 162812269 0.6654 0.7211

Residuals 127 3.1076e+10 244693777

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

**İki yönlü anova:**

> summary(aov(price~model+year,data = arabalar))

Df Sum Sq Mean Sq F value Pr(>F)

model 2 276375070 138187535 71.59 <2e-16 \*\*\*

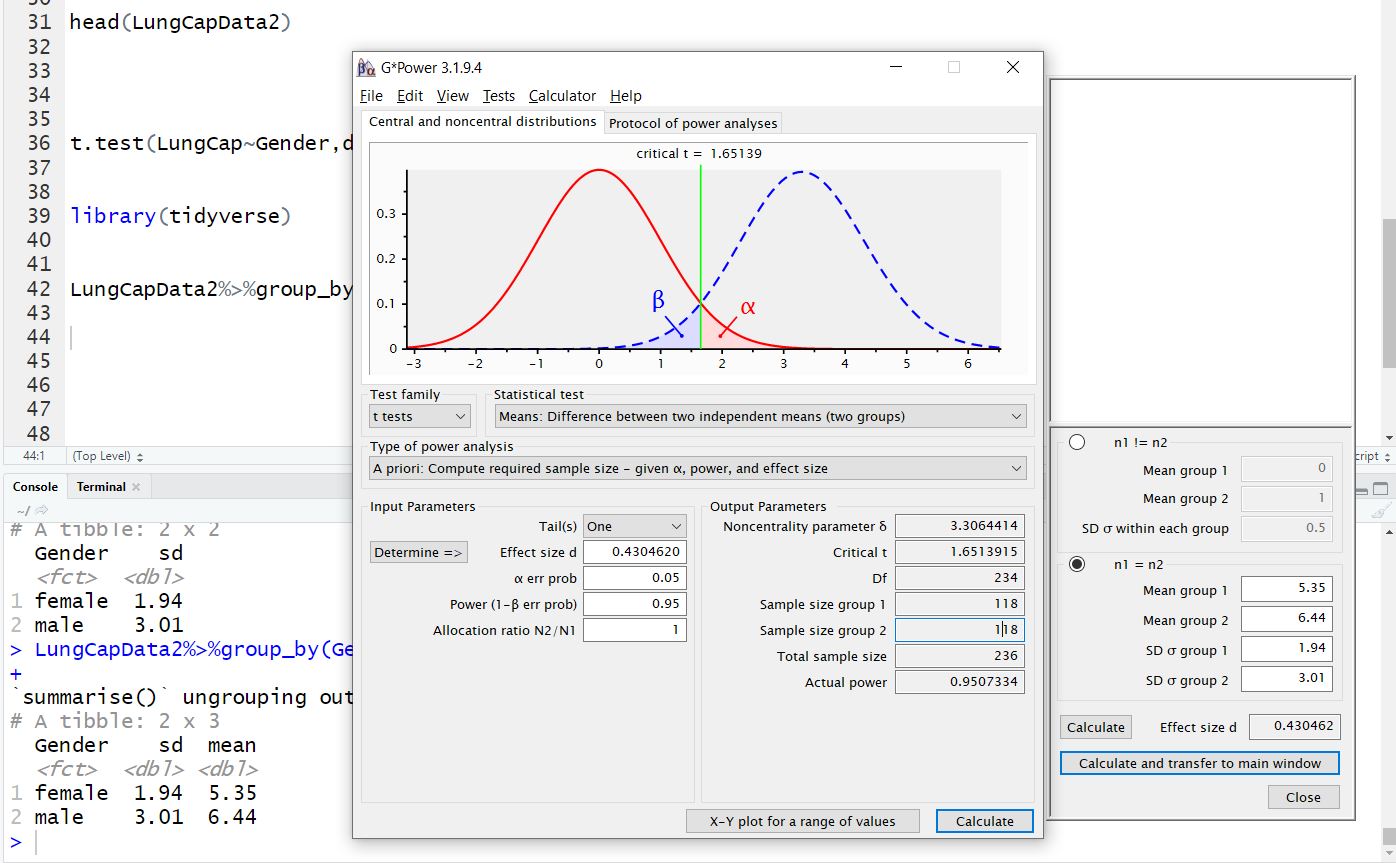
year 12 915781448 76315121 39.54 <2e-16 \*\*\*

Residuals 135 260577418 1930203

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

**Two Sample T test GPower Analizi:**



Total sample size en az alınması gereken örneklem sayısıdır.

Tail’i 2 seçeriz çünkü 2 grup arasındaki farkı arıyoruz.

Allcation ratio eger grup sayıları birbirlerine eşit ise 1 yaz.

Örneğin a grubunda 50 kisi b grubunda 40 kisi varsa allocation ratio

40/50’dir.

Eger gruplar birbirlerine eşit ise n1=n2 degilse n1!=n2 deriz.

Örneğin bir veri analizi eğitiminin kullanıcılar üzerinde etkili olup olmadığını ölçmek istiyorum bunun için GPower ile etki büyüklüğünü seçerim daha sonra örneklem sayısını öğrenip teste skarım.

Güven aralığı örneklem sayısı aynı seydir.

## **Kovaryans**

Olasılık teorisi ve istatistikte, kovaryans iki değişkenin birlikte ne kadar değiştiklerinin ölçüsüdür. Kovaryans, iki rastgele değişkenin beraber değişimlerini inceleyen bir istatistiktir.

Kovaryans iki değişkenin birlikte ne kadar değiştiğini inceleyen bilim dalıdır.

**Yanlılık**

Tahmin edilen değerler ile gerçek değerler arasındaki farktır.

Veri setinde bazı değerleri çok iyi bazı değerleri çok kötü tahmin etmesi yanlılığa sebep olur.

Bazı değerler tahmin edilen değerlere çok uzak ise yanlılık yüksektir.

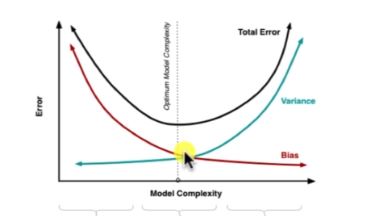
Varyans değişkenliktir. Yani doğru ne kadar değişken.



Soldaki resimde eğri herkese uymuyor varyans düşük yanlılık yüksek tahminler doğrudan çok uzakta.

Sağdaki resimde varyans yüksek yanlılık düşük neredeyse herkese mavi boncuk vermişiz. Modelin esnekliği yüksektir.

Modelin esnekliği varyanstır.



Yanlılık azalırken hata azalıyor varyans artıyor. En alt noktadan sonra varyans ile beraber hata oranı da artıyor.

apply(predict(lm(Sales~.,data = advertising),

advertising[,-4],

interval = "confidence"),

2,

function(x) mean(x))

Temel Bileşen Analizi (PCR):

Çok boyutluluk lanetine çözüm bulur p>n

eger coklu baglanti problemi varsa iki cozumumuz var biri degiskenleri silmek digeri pcr uygulamak. Eger değişken sayımız gözlem sayımızdan buyuk ise yine temel bileşen analizi uygularız.

library(pls)

model\_pcr=pcr(Salary~.,data=df\_hitters,scale=TRUE,validation="CV")

summary(model\_pcr)

#Bu validasyon plot ile butun bileşenlerin hata karelerini gösterir en düşük hatayı veren bileşeni

#buradan görebiliriz.

validationplot(model\_pcr,val.type = "MSE")

defaultSummary(data.frame(obs=df\_hitters$Salary,

pred=as.vector(model\_pcr$fitted.values)))

defaultSummary(data.frame(obs=df\_hitters$Salary,

pred=as.vector(predict(model\_pcr,df\_hitters[,],ncomp = 1:2))))

defaultSummary(data.frame(obs=df\_hitters$Salary,

pred=as.vector(predict(model\_pcr,df\_hitters[,],ncomp = 17))))

#Model tune etme işlemi.

#tuneLength max 20 bileşen demek.

pcr\_tune=train(df\_hitters[,-19],df\_hitters$Salary,

method="pcr",

preProc=c("scale","center"),

trControl=ctrl,

tuneLength=20)

defaultSummary(data.frame(obs=df\_hitters$Salary,

pred=as.vector(predict(pcr\_tune,df\_hitters))))

e1071 paketi içerisindeki naivebayes algoritması predict fonksiyonu ile direkt olarak faktör tahmini yapar dönüştürmeye gerek kalmaz.

Birden fazla görseli tek bir sayfada göstermek:

Gridextra kütüphanesi grid.arrange() fonksiyonu içerisine bütün grafikler yazılır.

library(gridExtra)

grid.arrange(fviz\_cluster(kmeans(k\_mean\_data,centers = 2,nstart = 25),k\_mean\_data)+theme\_classic(),

fviz\_cluster(kmeans(k\_mean\_data,centers = 3,nstart = 25),k\_mean\_data)+theme\_classic(),

fviz\_cluster(kmeans(k\_mean\_data,centers = 4,nstart = 25),k\_mean\_data)+theme\_classic(),

fviz\_cluster(kmeans(k\_mean\_data,centers = 5,nstart = 25),k\_mean\_data)+theme\_classic(),

ggplot(adult2)+geom\_bar(aes(x=income,fill=occupation)))

başlık ekleme ggtitle(“baslık”)

apply(wine\_clustering\_,2,function(x) table(x))

knnImputation ile eksik değer ataması yapmak:

library(DMwR) paketi içerisindeki knnImputation işleviyle yapabiliriz. Eksik veri bulunan değişkenlerimiz kategorikte olabilir numerikte olabilir.

missForest kütüphanesini kullanarak randomforest algoritması ile de tahminleme yapabiliriz.

Sapply liste dondurur:

l=sapply(hitters,function(x) which(is.na(x)))

En iyi eksik veri dldurma işlemi hangisi ?

Kendimiz veri setine suni eksik veri atarız. Daha sonra knn ya da randomforest ile tahminleme yaparız bu veri setimiz için hangisi daha az hata ortaya çıkarıyorsa o algoritmayı kullanırız.

rpart.plot(rpart\_model,cex=1)

cex argümanı ağaç üzerindeki yazıların byutlarını margin ise ağaçların dal uzunluklarını ayarlar

rpart yani karar ağacı fonksiyonu otomatik olarak cross validation işlemi uygular.

Tahmin edilen değerler ile gerçek değerlerin karşılaştırılması.

ggplot(advertising,size=6)+geom\_point(aes(x=Sales,y=predict(rpart\_model,advertising[,-4]),color="red"),position = "jitter")+

theme\_classic()

tune\_grid=hiperparametre belirlemek demek yani en iyi sonucu veren k değeri nedir en iyi sonucu veren ağaç budaması nedir

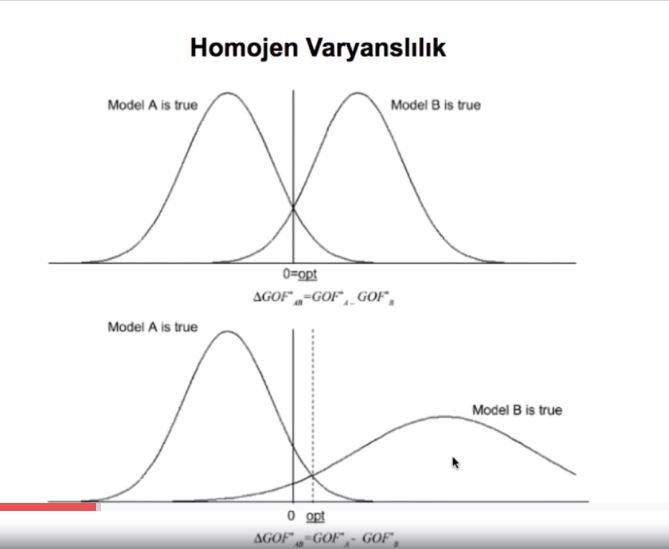
kolmogrov smirnov testi varyansı yüksek olan verilerde daha iyi ayarlama yapıyor ama shapiro wilk ise tamamen normalliği test etmek için vard.

En güçlü sonuçları veren shapiro wilk testidir.

p<0.05 yüzde 95 güven aralığı demek.

shapiro wilk testi buyuk veri de iyi tahminleme yapamıyor 300 500 1000 ornek üzerindeki buyuk verilerde doğru sonuclayamıyor bu yüzden veri setinin içerisinden belli bir ornek alıp bakacağım 100-200 200-300 300-400.

Arası değerleri kontrol edeceğim.

Histogram yani normallik verinin frekansını verir.

Alttaki görselde varyanslar homojen dağılmamış. Soldaki verinin varyansı sağdakine gore daha dusuk.

**ANOVA:**

library(car)

head(iris)

#ilk önce aykırı değerleri kontrol ediyorum aykırı deger varsa anova yapamam.

#sonra normalliği test ediyorum normallik varsayımını sağlamam gerek.

by(iris$Sepal.Length,iris$Species,shapiro.test)

#verinin dağılımı normal

boxplot(iris$Sepal.Length)

#aykırı deger yok

#şimdide varyans testime bakıyorum eger test sonucum 0.05 ten buyukse homojenliği test etmeyecegim.

#direkt olarak gruplar arsında fark yok diyebilirim.

aov(Sepal.Length~Species,data = iris)

summary(aov(Sepal.Length~Species,data = iris))

#gruplar arasında fark var simdi hangi testi seçecegimi belirlemek için varyansların homojenliğini

#kontrol edicem.

library(car)

leveneTest(formula(iris$Sepal.Length~iris$Species))

#varyanslar homojen değil bu yuzden tukey yerine tamhane ile çoklu karşılaştırma yapacagım.

library(PMCMRplus)

tamhane= tamhaneT2Test(aov(Sepal.Length~Species,data = iris))

ggplot(iris)+geom\_boxplot(aes(x=Species,y=Sepal.Length,fill=Species))+theme\_classic()

anova testinde aykırı değer olmamalı veri normal dağılmalı varyanslar homojense tukey değilse tamhane testi kullanılmalı

levene yani homojenlik testi car paketi içerisinde.

Df yani degree of freedom serbestlik derecesi demek 4 grubum varsa df 3 tur 3 grubum varsa df 2 dir bir tanesini baz alıp test eder. N-1

Subset veri setini boler:

subset(iris,subset = Species=="versicolor" | Species=="setosa")

sonra bu değişkeni karaktere cevir daha sonra as.factor() e cevir.

Eger veri normal dağılıyorsa homjenliği test etmek için bartlet testi kullanılır.

bartlett.test(b$Sepal.Length~b$Species)

independent sample two testindede homojenlik kontrol edilir.

Ttest 1.varsayım gruplar normal dağılmalı

2.varsayım varyanslar homojen dağılmalı.

Confiedence interval güven aralığı :

Welch Two Sample t-test

data: b$Sepal.Width by b$Species

t = 9.455, df = 94.698, p-value = 2.484e-15

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

0.5198348 0.7961652

sample estimates:

mean in group setosa mean in group versicolor

3.428 2.770

Bu iki grup arasındaki fark %95 olasılıkla 0.519 ile 0.796 arasında olacaktır.

t.test(b$Sepal.Width~b$Species,mu=0.7)

Welch Two Sample t-test

data: b$Sepal.Width by b$Species

t = -0.60351, df = 94.698, p-value = 0.5476

alternative hypothesis: true difference in means is not equal to 0.7

95 percent confidence interval:

0.5198348 0.7961652

sample estimates:

mean in group setosa mean in group versicolor

3.428 2.770

H0=İki grup arasındaki fark 0.7 dir.

H1=İki grup arasındaki fark 0.7 den farklıdır.

p>0.05 h0 kabul h1 red. Aradaki fark gerçekten 0.7 dir %95

rpart tahminleme:

> predict(rpart(Species~.,data = train),test[,-5])

> b=c(1:30)

> for(i in 1:30){

+ if(predict(rpart(Species~.,data = train),test[,-5])[i,1]>0.5){

+ b[i]="setosa"

+ }

+ }

> for(i in 1:30){

+ if(predict(rpart(Species~.,data = train),test[,-5])[i,2]>0.5){

+ b[i]="versicolor"

+ }

+ }

> for(i in 1:30){

+ if(predict(rpart(Species~.,data = train),test[,-5])[i,3]>0.5){

+ b[i]="virginica"

+ }

+ }

> defaultSummary(data.frame(obs=test$Species,

+ pred=b))

Accuracy Kappa

0.90 0.85

Randomforest ile tahminleme yaparsan el ile düzeltmek zorunda kalmazsın ve rpart ile aynı sonucu veriyor.

Rpart fonksiyonunu görselleştirme yaparken kullan rpart.plot ve rpart tahminleme yaparken randomforest ile tahminleme yap.

Multinominal Regresyon:

library(nnet)

multinom(Species~.,data = iris)

defaultSummary(data.frame())

predict(multinom(Species~.,data = iris),iris[,-5])

defaultSummary(data.frame(obs=iris$Species,

pred=predict(multinom(Species~.,data = iris),iris[,-5])))

%98 tahminledi güzel rakam.

Değişken isminin indeksini bulmak:

> which(colnames(iris)=="Species")

[1] 5

> model=multinom(model~.,data = arabalar)

> defaultSummary(data.frame(obs=arabalar$model,

+ pred=predict(model,arabalar)))

Accuracy Kappa

0.6866667 0.4684461

Multinominal regresyon %68 tahmin ederken destek vektör makinesi %56 tahminleyebildi.

Multinominal lojistik regresyonda bizim eger 3 grubumuz varsa 2 tane lojistik regresyon oluşturarak tahminler.

Multiinomial regresyon yaparken ilk once bağımlı degisken ile bağımsız değişkenler arasındaki ilişkiye bakıp anlamlı değişkenleri seçebiliriz.

Ornegin bağımlı değişkenle bağımsız degisken arasındaki ki-kare testi anlamsız ise o değişkeni modele ekleme.

Multinominal regresyon değişken önem düzeyleri:

> varImp(multinom(race~.,data = adult2))

**KOLMOGROV SMİRNOV TESTİ:**

**library(RcmdrMisc)**

normalityTest(age~race,test = "lillie.test",data = adult2)

Multinomial regresyon direkt sınıflandırma problemlerinde direkt olarak sonucu verir. Fakat lojistik regresyon olasılık verir. Eğer predict yaparken type=class dersen lojistik regresyonda sınıfları verir.

**Eş Örneklem Testi:**

> t.test(Sepal.Length,Sepal.Width,paired = T)

Paired t-test

data: Sepal.Length and Sepal.Width

t = 34.815, df = 149, p-value < 2.2e-16

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

2.627874 2.944126

sample estimates:

mean of the differences

2.786

İki ölçüm arasında fark vardır. Bu fark istatistiksel olarak 2.786’dır. %95 güven düzeyinde 2.62-2.94 arasındadır.

**Kolmogrov-Smirnov Testi:**

> normalityTest(~Sepal.Width,test = "lillie.test",data = iris)

Lilliefors (Kolmogorov-Smirnov) normality test

data: Sepal.Width

D = 0.10566, p-value = 0.0003142

> normalityTest(~Sepal.Length,test = "lillie.test",data = iris)

Lilliefors (Kolmogorov-Smirnov) normality test

data: Sepal.Length

D = 0.088654, p-value = 0.005788

**Normal Dağılmayan Eş örneklem testi:**

> wilcox.test(Sepal.Length,Sepal.Width,paired = TRUE)

Wilcoxon signed rank test with continuity correction

data: Sepal.Length and Sepal.Width

V = 11325, p-value < 2.2e-16

alternative hypothesis: true location shift is not equal to 0

**Multinomial Regresyonda p\_value hesaplama:**

zskoru\_multinomial=summary(model)$coefficients/summary(model)$standard.errors

p\_value=(1-pnorm(abs(zskoru\_multinomial\_regresyon),0,1))\*2

**p\_value Hesaplama:**

> z=summary(model)$coefficients/summary(model)$standard.error

> p\_value=(1-pnorm(abs(z),0,1))\*2

**p\_value Hesaplama:**

aer paketindeki coeftest fonksiyonu ile de p\_value hesaplanabilir.

> library(AER)

> coeftest(model2)

z test of coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -2.116966 1.575143 -1.3440 0.1790

Age -0.222944 0.051911 -4.2947 1.749e-05 \*\*\*

LungCap 0.328833 0.070857 4.6408 3.470e-06 \*\*\*

Height 0.031715 0.032206 0.9847 0.3248

Smokeevet 0.573903 0.310322 1.8494 0.0644 .

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Sınıflandırma problemlerinde her grup için neredeyse eşit sayıda örneklem al.

gözlem sayısı fazla olan grubun doğruluk oranı daha fazla olur.

**Lojistik Regresyon Tune İşlemi:**

> model=glm(Smoke~Age+Gender+Height,family = "binomial",data = l)

İlk once bir model oluştur ve anlamlı değişkenleri seç

Daha sonra bu modeli tune et.

> model\_tune\_glm=train(Smoke~Age+Gender+Height,

+ data = l,

+ method="glm",

+ trControl=ctrl)

**Multinomial Regresyon Projesi:**

**# Multinomial regresyonda tune işlemini decay parametresii ayarlamak için yapıyorum.**

model=multinom(Species~.,data = iris)

coeftest(model)

#modeldeki anlamlı değişkenleri seçtim.

tah=predict(model,iris)

#modelin doğruluk oranına baktım.

defaultSummary(data.frame(obs=iris$Species,

pred=tah))

**# modeli valide ettim. Valide ettiğim modeldeki decay parametresini buldum.**

model\_tune=train(Species~.,

data = iris,

method = "multinom",

trControl=ctrl)

**# Modeli decay parametresiyle birlikte bir daha kuruyorum.**

model\_tuned=multinom(Species~.,

data = iris,

decay=1e-04)

**# en son bulduğum nihai model ile tahminleme yapıyorum.**

defaultSummary(data.frame(obs=iris$Species,

pred=predict(model\_tuned,iris)))

**Lojistik Regresyon Model Tune:**

> model=lm(Age~.,data = LungCapData2)

**# Model Tune işlemi.**

**# Değişkenlerimi seçtim hepsi anlamlıymış şimdide bu değişkenlerle modeli tune edeceğim**

> model=lm(Age~.,data = LungCapData2)

> model\_tune=train(LungCapData2[,-1],

+ LungCapData2$Age,

+ mehodt="lm",

+ trControl=ctrl)

**# Normal Model tahminlemesi:**

> defaultSummary(data.frame(obs=LungCapData2$Age,

+ pred=predict(model,LungCapData2)))

RMSE Rsquared MAE

1.6396324 0.6914286 1.2589499

**# Tune edilmiş modelin tahminlemesi:**

> defaultSummary(data.frame(obs=LungCapData2$Age,

+ pred=predict(model\_tune,LungCapData2)))

RMSE Rsquared MAE

1.1224525 0.8613489 0.8547388

**TARİH:**

library(lubridate)

ymd("2020-11-16")

dmy("15-06-2020")

ymd(dmy("15-06-2020"))

bugun=today()

sample(seq(as.Date('2019/06/18'), as.Date('2019/10/24'), by="day"), 250,replace = T)

ilk önce modeli oluştur daha sonra anlamlı değişkenleri seç.

Tune ettiğinde modelde anlamsız değişkenleri kullanmamaya dikkat et.

**Tüm Değişkenlere Kolmogrov-Smirnov Testi:**

apply(marketing,2,function(x) normalityTest(~x,test = "lillie.test",data = marketing))

**Svm algoritması da Knn gibi mesafe metriğine dayalı olduğu için scale ederken ve model tune ederken problem yaşayacağımız için Kategorik değişkenleri modele eklemiyoruz.**

**Svm Tune:**

#Svm ile tahminleme

test\_x=test\_x[,-c(14,15,19)]

train\_x=train\_x[,-c(14,15,19)]

training=training[,-c(14,15,19)]

#Svm de Knn gibi kategorik veri kabul etmez.

library(e1071)

svm\_fit=svm(Salary~.,data = training)

names(svm\_fit)

svm\_fit$epsilon

svm\_fit$cost

svm\_fit$coefs

defaultSummary(data.frame(obs=test\_y,

pred=predict(svm\_fit,test\_x)))

defaultSummary(data.frame(obs=train\_y,

pred=predict(svm\_fit,train\_x)))

svm\_fit\_tune=train(train\_x,train\_y,

method="svmRadial",

trControl=ctrl,

preProc=c("center","scale"))

defaultSummary(data.frame(obs=train\_y,

pred=predict(svm\_fit\_tune,train\_x)))

defaultSummary(data.frame(obs=test\_y,

pred=predict(svm\_fit\_tune,test\_x)))

**Welch-Anova Testi:**

Eğer veri normal fakat varyanslar eşit değilse welch anova yap.

# Varynas homojenliği kontrol etmek için levene testini kullanacağım

#için car paketini çağırıyorum.

library(car)

# Normallik testi için bu paketi çağırdım

library(RcmdrMisc)

#veri normal dağılıyor.

normalityTest(Sepal.Length~Species,test = "lillie.",data = iris)

#Varyanlasr eşit değil

leveneTest(Sepal.Length~Species,data = iris)

# Welch Anova yapabilmek için rstatix paketini kullanıyorum.

library(rstatix)

welch\_anova=welch\_anova\_test(Sepal.Length~Species,data = iris)

# Anova testimin P value değerine bakıyorum.

welch\_anova$p

**#Karar Ağacı**

indeks=createDataPartition(advertising$Sales,

list = F,

p=0.80)

test=advertising[-c(indeks),]

train=advertising[c(indeks),]

test\_x=dplyr::select(test,-Sales)

test\_y=test$Sales

train\_x=dplyr::select(train,-Sales)

train\_y=train$Sales

training=data.frame(train\_x,Sales=train\_y)

library(rpart.plot)

library(rpart)

rpart\_fit=rpart(Sales~.,data = training)

defaultSummary(data.frame(obs=test\_y,

pred=predict(rpart\_fit,test\_x)))

defaultSummary(data.frame(obs=train\_y,

pred=predict(rpart\_fit,train\_x)))

plot(rpart\_fit,margin = 0.1)

text(rpart\_fit,cex=1)

rpart.plot(rpart\_fit)

tune\_grid=data.frame(

cp=seq(0,0.05,len=25)

)

rpart\_fit\_tune=train(Sales~.,

method="rpart",

preProc=c("scale","center"),

tuneGrid=tune\_grid,

data=training)

rpart\_fit\_tune$bestTune

rpart\_fit\_tune$finalModel

defaultSummary(data.frame(obs=test\_y,

pred=predict(rpart\_fit,test\_x)))

defaultSummary(data.frame(obs=test\_y,

pred=predict(rpart\_fit\_tune,test\_x)))

rpart.plot(rpart\_fit\_tune$finalModel,cex=1)

plot(test\_y,predict(rpart\_fit\_tune,test\_x))

abline(0,1)

**#RandomForest algoritmasında tune edilmesi gereken iki**

**#parametre var biri mtry diğeri ntree**

**Çoklu Kolmogrov-Smirnov Test**

apply(iris[,-5],2,function(x) normalityTest(x,test = "lillie.test"))

**Normalleştirme:**

**#veri normal dağılmamış.**

> normalityTest(Salary~League,test = "lillie.test",data = df)

--------

League = A

Lilliefors (Kolmogorov-Smirnov) normality test

data: Salary

D = 0.15365, p-value = 1.492e-08

--------

League = N

Lilliefors (Kolmogorov-Smirnov) normality test

data: Salary

D = 0.14676, p-value = 6.662e-07

--------

p-values adjusted by the Holm method:

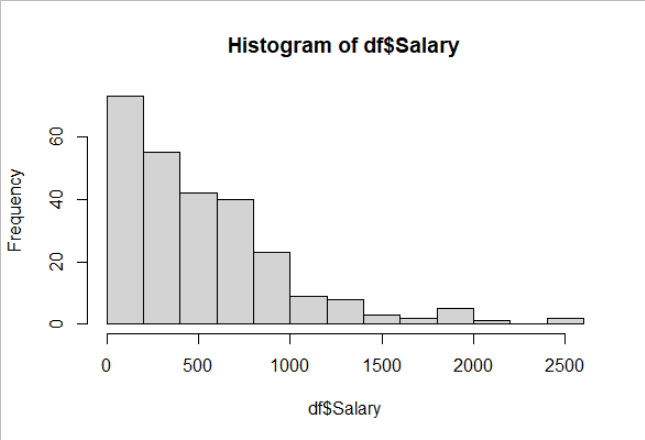
unadjusted adjusted

A 1.4921e-08 2.9843e-08

N 6.6619e-07 6.6619e-07

Verinin dağılımı bu şekilde:

> hist(df$Salary)



Şimdide logaritmasını alacağım log10 tabanında:

> normalityTest(log10(df$Salary)~League,test = "lillie.test",data = df)

--------

League = A

Lilliefors (Kolmogorov-Smirnov) normality test

data: log(df$Salary)

D = 0.071244, p-value = 0.08072

--------

League = N

Lilliefors (Kolmogorov-Smirnov) normality test

data: log(df$Salary)

D = 0.10478, p-value = 0.001947

--------

p-values adjusted by the Holm method:

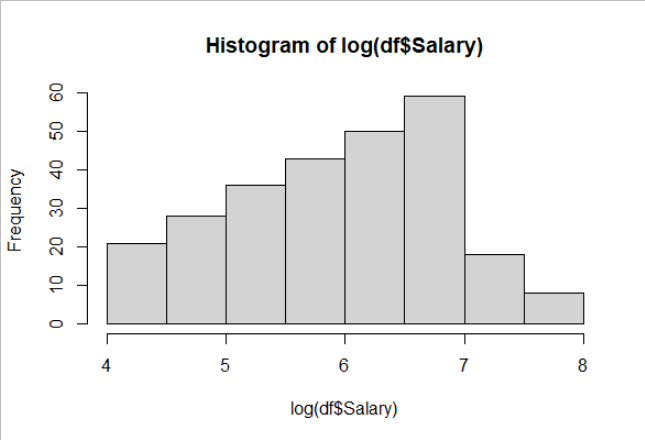
unadjusted adjusted

A 0.0807160 0.0807160

N 0.0019474 0.0038949

Biraz daha normalleşti.

Şimdide grafiği tekrardan inceleyelim:



**Not:** Logaritmik dönüşüm yaparken değerler 0 dan buyuk olmalı bu yüzden en kucuk değeri kontrol ederim or: en kucuk değerim -3 ise tum verilerime +4 eklerim

Ayrıca normalliği her grup kombinasyonu için yapman lazım. Verilerin normallikten biraz kaymışsa yine anova kullanabilirsin. Ama çok anormal dağılım varsa o zaman kruskal wallis testi kullanabilirsin.

Verinin skewness değerinin normlizasyon işlemi üzerinde etkisi yoktur sadece değişkende negatif veya 0 değeri olmasın yeter.

**Değişkenin sütun numarasını bulmak:**

which(colnames(a)=="League")

which(colnames(a)=="Division")

**PROJE HİTTERS:**

normalliği test et normal değilse ve ort medyan birbirine yakın ise aykırı değerleri ve eksik verileri medyan ile doldur.

**Aykırı değer tespit etmek değer atamak ya da silmek:**

boxplot(a$Errors)

boxplot.stats(a$Errors)$out

d=c()

for(i in 1:length(boxplot.stats(a$Errors)$out)){

c=which(a$Errors==boxplot.stats(a$Errors)$out[i])

c=as.integer(c)

d=c(d,c)

}

a[d,]$Errors=median(a$Errors)

normalityTest(~Hits,test = "lillie.test",data = a)

a$Hits[which(is.na(a$Hits))]=median(a$Hits,na.rm = T)

knn ile tahminleme veya eksik veri doldurması yapacaksak eger k değeri sınıflandırma problemleri içi değişken sayısının karekökü regresyon problemleri için değişken sayısı/3

bütün eksik değerleri ortalama medyan ile doldurduktan sonra :

which(is.na(a$Salary))

normalityTest(~Salary,test = "lillie.test",data = a)

summary(a$Salary)

a$Salary[which(is.na(a$Salary))]=median(a$Salary,na.rm = T)

maaş değişkenini knnimputation fonksiyonu ile doldurdum dikkat edelim bu fonksiyon kategorik değişken kabul etmiyor:

değişken sayım 20 olduğu için k değerini 7 olarak belirledim.

> knnImputation(data = a[,-c(14,15,20)],k=7,scale=T)

Bu atadığım değerleri tekrardan veri setime eklediğimi belirtmem gerek.

> a[,-c(14,15,20)]=knnImputation(data = a[,-c(14,15,20)],k=7,scale=T)

Eğer regresyon svm randomforest desicion tree gibi algoritmalarla model oluşturabiliyorsan sakın neuralnetworkse bulaşma

Eger sınıflandırma problemi çözeceksen sınıf sayıları mutlaka birbirlerine yakın olmalı.

Aykırı Değerleri değişkenden silmek:

a=df$Salary

boxplot(a)

boxplot.stats(a)$out[1]

for(i in 1:11){

b=c(b,which(a==boxplot.stats(a)$out[i]))

}

a=a[-b]

başka bir yol:

> for(i in boxplot.stats(a)$out){

+ a=a[-which(a==i)]

+ }

**DataFrame’deki değişkenden eksik veri silme:**

veri=data.frame(Maas=df$Salary,Lig=df$NewLeague)

for(i in boxplot.stats(veri$Maas)$out){

veri=veri[-which(veri$Maas==i),]

}

Gruplar Normal dağılıyorsa homojenlik için bartlet.test() yap

Varyanslar eşit değilse welch testleri yap

**Varyanslar eşit t test**

> t.test(Maas~Lig,var.equal=T,data = veri)

Two Sample t-test

data: Maas by Lig

t = -0.053517, df = 250, p-value = 0.9574

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-86.98106 82.37902

sample estimates:

mean in group A mean in group N

472.2517 474.5527

**Varyanslar eşit değil t test**

> t.test(Maas~Lig,var.equal=F,data = veri)

Welch Two Sample t-test

data: Maas by Lig

t = -0.05354, df = 245.33, p-value = 0.9573

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-86.95379 82.35175

sample estimates:

mean in group A mean in group N

472.2517 474.5527

Eğer veriler normal dağılmıyorsa varyansa bakmıyorum.

T testleri veya anova için

1. Veriler normal dağılmalı

Veriler normal dağılmıyorsa direk nonparametric test yap

Normal dağılmıyorsa varyans kontrol etmeden wilcox ya da kruskal Wallis testi yap.

Eğer normal dağılıyorsa varyanslar homojense ttes(var.equal=T)

Varyanslar eşit değilse ttest(var.equal=F)

1. Varyanslar eşit olmalı
2. Aykırı değer olmamalı

**Eksik Veri Silmek:**

veri=veri[-which(is.na(veri$yas)),]

**Aykırı Değer Silme :**

**\*\*\*En küçük aykırı değerin üzerindeki değerleri sil.**

boxplot.stats(veri$yas)$out

min(boxplot.stats(veri$yas)$out)

veri=veri[-which(veri$yas>67),]

**zaman serileri analizinde:**

**analizin faydaları:**

1. Aykırı gözlemleri yakalar.
2. Eksik verileri tamamlar.
3. Zaman serisi analiz yöntemleri:
4. 1.1 SMA toplam hareketli ortalama yöntemi
5. 1.2 CMA kümülatif ortalama yöntemi gibi yöntemler.

Bu yöntemler zaman serisi grafiklerini çok güzel düzenliyor.

Eksik verileri knnimputation ile ata.

Subset fonksiyonu veriyi istediğin parçaya ayırır.

arules::subset(iris,Species!="setosa")

b=data.frame(orderID=sample(c(1:1000),5000,replace = T),

product=sample(c("NULL","a","b","c","d","e","f","g","h"),5000,replace = T,

prob=c(0.15,0.65,0.3,0.15,0.1,0.1,0.2,0.1,0.15)))

veriyi yeniden subset ile parçaladım.

subcategories=arules::subset(b$product,b$product!="NULL")

transiction=arules::subset(b$orderID,b$product!="NULL")

**Regular Expressions:**

> gsub("[a]","e","harun")

[1] "herun"

**Birliktelik Analizi:**

b=data.frame(orderID=sample(c(1:1000),5000,replace = T),

product=sample(c("NULL","a","b","c","d","e","f","g","h"),5000,replace = T,

prob=c(0.15,0.65,0.3,0.15,0.1,0.1,0.2,0.1,0.15)))

subcategories=arules::subset(b$product,b$product!="NULL")

transiction=arules::subset(b$orderID,b$product!="NULL")

transaction\_by\_subcategories=split(subcategories,transiction)

basket=as(transaction\_by\_subcategories,"transactions")

rules=apriori(basket, parameter = list(support=0.00003,confidence=0.01))

rules\_df=data.frame(lhs=labels(lhs(rules)),rhs=labels(rhs(rules)),rules@quality)

rules\_df$lhs=gsub("[{}]","",rules\_df$lhs)

rules\_df$rhs=gsub("[{}]","",rules\_df$rhs)

**Birliktelik Analizi:**

set.seed(11)

veri=data.frame(MusteriID=sample(1:1000,5000,replace = T),

Egitim=sample(c("spss","java","python","sas","excel","R","Power BI",

"R ile Makine Öğrenmesi",

"Modern R programlama","A-Z R Programlama"),5000,replace = T))

library(tidyverse)

library(arules)

trans=split(veri$Egitim,veri$MusteriID)

trans=as(trans,"transactions")

kurallar=apriori(trans,parameter = list(support=0.00003,confidence=0.01))

kurallar\_df=data.frame(lhs=labels(lhs(kurallar)),rhs=labels(rhs(kurallar)),kurallar@quality)

kurallar\_df=gsub("[{}]","",kurallar\_df$lhs)

kurallar\_df=gsub("[{}]","",kurallar\_df$rhs)

arrange(kurallar\_df,desc(kurallar\_df$confidence))

> arrange(kurallar\_df,desc(kurallar\_df$confidence))

lhs rhs support confidence

1 {Modern R programlama,python,R,R ile Makine Öğrenmesi,sas} {excel} 0.005025126 1.0000000

2 {A-Z R Programlama,excel,java,Power BI,python} {Modern R programlama} 0.002010050 1.0000000

3 {excel,java,Modern R programlama,Power BI,sas} {A-Z R Programlama} 0.005025126 1.0000000

4 {Modern R programlama,python,R,R ile Makine Öğrenmesi,sas,spss} {excel} 0.003015075 1.0000000

5 {excel,java,Modern R programlama,python,R,spss} {sas} 0.004020101 1.0000000

6 {A-Z R Programlama,excel,Modern R programlama,python,R,spss} {sas} 0.001005025 1.0000000

7 {java,Modern R programlama,python,R,R ile Makine Öğrenmesi,sas} {excel} 0.002010050 1.0000000

8 {excel,java,Modern R programlama,python,R,R ile Makine Öğrenmesi} {sas} 0.002010050 1.0000000

9 {A-Z R Programlama,excel,Modern R programlama,python,R ile Makine Öğrenmesi,sas} {R} 0.001005025 1.0000000

10 {A-Z R Programlama,Modern R programlama,python,R,R ile Makine Öğrenmesi,sas} {excel} 0.001005025 1.0000000

11 {excel,java,Modern R programlama,Power BI,python,sas} {R} 0.001005025 1.0000000

115 A-Z R Programlama,excel,java,Modern R programlama,Power BI,R,R ile Makine Öğrenmesi,sas spss 1.0000000

116 A-Z R Programlama,java,Modern R programlama,Power BI,R,R ile Makine Öğrenmesi,sas,spss excel 1.0000000

117 A-Z R Programlama,excel,java,Power BI,R,R ile Makine Öğrenmesi,sas,spss Modern R programlama 1.0000000

118 excel,Power BI,R,R ile Makine Öğrenmesi,spss java 0.8750000

119 excel,java,Modern R programlama,python,spss sas 0.8571429

120 Modern R programlama,Power BI,R,R ile Makine Öğrenmesi,sas A-Z R Programlama 0.8571429

121 excel,java,python,R,spss sas 0.8571429

122 excel,Modern R programlama,python,R,spss sas 0.8333333

123 excel,java,Modern R programlama,python,R sas 0.8333333

124 A-Z R Programlama,excel,java,python,R ile Makine Öğrenmesi Modern R programlama 0.8333333

125 excel,java,Modern R programlama,R ile Makine Öğrenmesi,sas spss 0.8333333

126 excel,java,Modern R programlama,Power BI,R,spss A-Z R Programlama 0.8333333

Lhs=alınan eğitim rhs=verilen eğitim

> NROW(unique(kurallar\_df$lhs))

[1] 907

> NROW(unique(kurallar\_df$lhs,kurallar\_df$rhs))

[1] 987

T testlerinde eger grup varyansları eşit değilse kendisi direkt olarak welch wilcoxon testi yapar.

> t.test(Height~Smoke,data = LungCapData2)

Welch Two Sample t-test

data: Height by Smoke

t = 11.615, df = 114.84, p-value < 2.2e-16

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

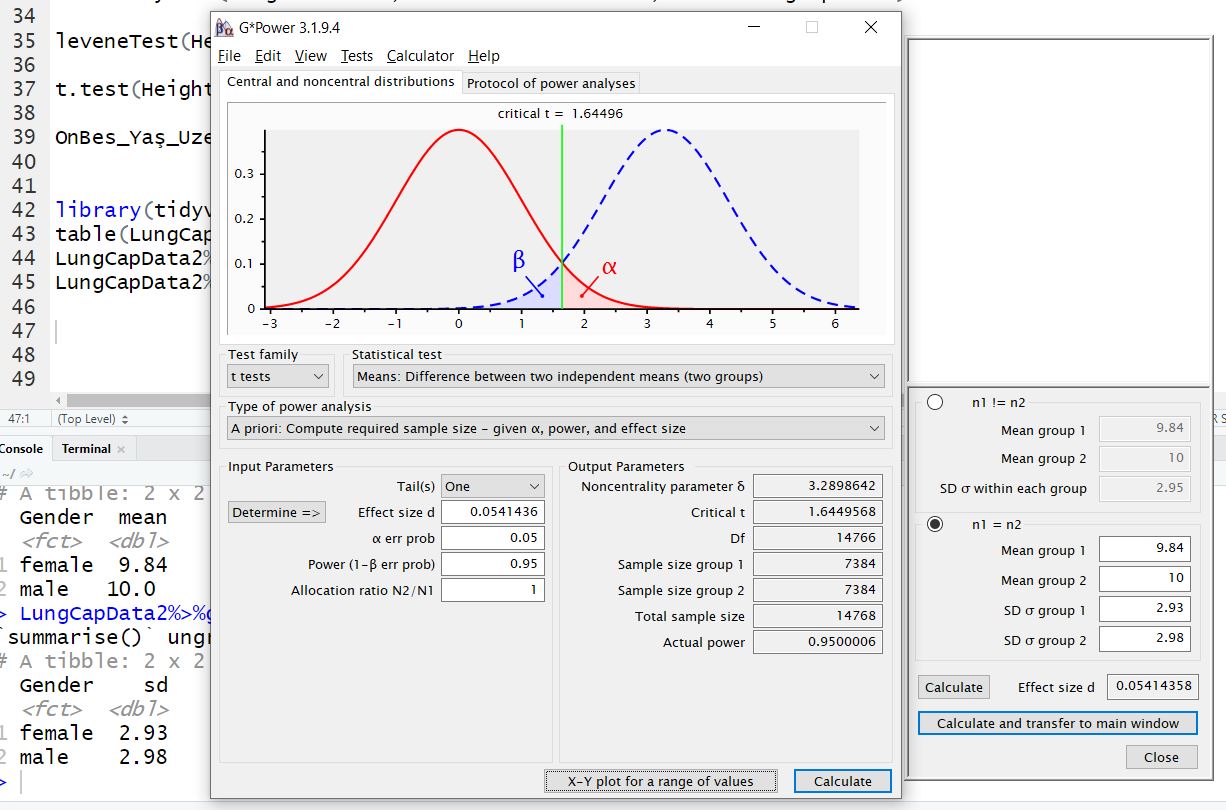
4.430258 6.251968

sample estimates:

mean in group 0 mean in group 1

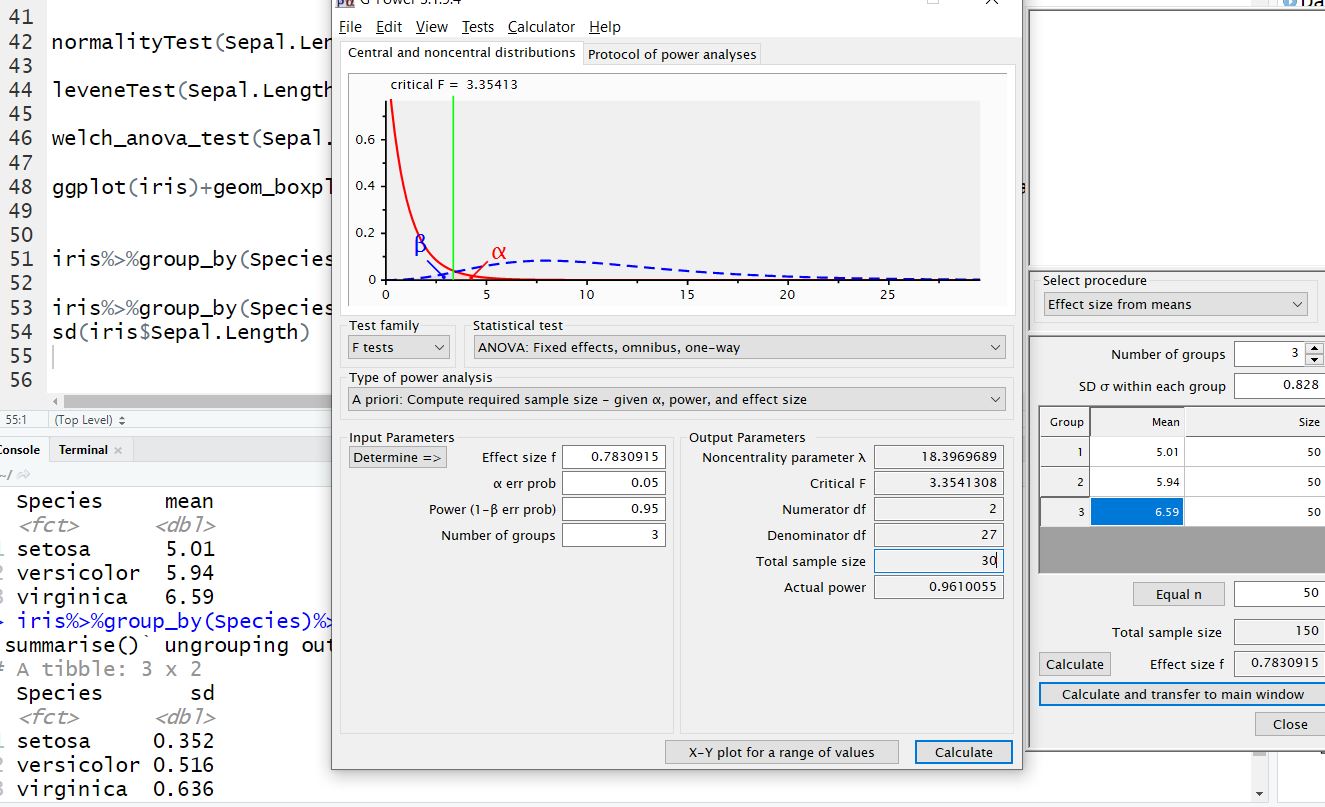
65.95385 60.61273

**Two Sample T-test Güç Analizi alpha beta değerleri:**



Bu iki grubun ortalamaları birbirine çok yakın bu yüzden effect size çok düşük çıktı bu iki grup arasında fark var diyebilmem için her gruptan en az 7384 adet veri toplamam lazım.

**One way Anova Güç Analizi alpha beta değerleri:**



Açımlayıcı Faktör Analizi:

Birbiriyle ilişkili olan kolonları tek bir faktör altında topluyor.

Örneğin fizik matematik kimya tarih biyoloji coğrafya gibi kolonlarım olsun bu kolonları iki kolon ile açıklıyor sözel ve sayısal olarak iki kolon.

* Multicolinetery çoklu doğrusallık problemlerini ortadan kaldırır.
* Vif degerleride bunu yapar.

> gapminder%>%mutate(gdp=gdpPercap\*pop)%>%

+ filter(year==2007)%>%

+ arrange(desc(gdp))

gapminder%>%filter(year==2007)%>%mutate(lifeExpMonths=12 \* lifeExp)%>%arrange(desc(lifeExpMonths))

tolower lower lowercase downcase

> a="harun"

> str\_replace(a,"un","ben")

[1] "harben"

> gsub("un","ben",a)

[1] "harben"

Güven aralıklarını almak istedim ve ortalamam fark 0’dır dedim.

Ortalama fark 3.999’tür bu değer de güven aralığı içerisinde olduğu için fark yoktur.

Güven aralığı demek -3 ile 10 arası fark anlamlı değildir demek.

> wilcox.test(breaks~wool,data = warpbreaks,mu=0,

+ conf.int=TRUE)

Wilcoxon rank sum test with continuity correction

data: breaks by wool

W = 431, p-value = 0.2531

alternative hypothesis: true location shift is not equal to 0

95 percent confidence interval:

-2.999939 9.999948

sample estimates:

difference in location

3.999927

Aralarındaki fark 15’tir dedim hayır aradaki fark 15 değildir dedi.

> wilcox.test(breaks~wool,data = warpbreaks,mu=15,

+ conf.int=TRUE)

Wilcoxon rank sum test with continuity correction

data: breaks by wool

W = 180, p-value = 0.001442

alternative hypothesis: true location shift is not equal to 15

95 percent confidence interval:

-2.999939 9.999948

sample estimates:

difference in location

3.999927

Aralarındaki fark 15’den küçüktür dedim. Evet gerçekten aradaki fark 3.999 gerçekten 15’ten küçük.

> wilcox.test(breaks~wool,data = warpbreaks,mu=15,

+ conf.int=TRUE,

+ alternative="less")

Wilcoxon rank sum test with continuity correction

data: breaks by wool

W = 180, p-value = 0.0007208

alternative hypothesis: true location shift is less than 15

95 percent confidence interval:

-Inf 9.000023

sample estimates:

difference in location

3.999927

Greater aradaki fark 15’den büyüktür. H0 aradaki fark -15’ten büyük değildir H1 aradaki fark -15’ten büyüktür.

> wilcox.test(breaks~wool,data = warpbreaks,mu=-15,

+ conf.int=TRUE,

+ alternative="greater")

Wilcoxon rank sum test with continuity correction

data: breaks by wool

W = 652.5, p-value = 3.231e-07

alternative hypothesis: true location shift is greater than -15

95 percent confidence interval:

-1.99999 Inf

sample estimates:

difference in location

3.999927

> normalityTest(Sepal.Length~Species,test = "lillie.test",data = a)

--------

Species = setosa

Lilliefors (Kolmogorov-Smirnov) normality test

data: Sepal.Length

D = 0.11486, p-value = 0.09693

--------

Species = versicolor

Lilliefors (Kolmogorov-Smirnov) normality test

data: Sepal.Length

D = 0.096241, p-value = 0.2942

--------

p-values adjusted by the Holm method:

unadjusted adjusted

setosa 0.096932 0.19386

versicolor 0.294170 0.29417

>

> leveneTest(Sepal.Length~Species,data = a)

Levene's Test for Homogeneity of Variance (center = median)

Df F value Pr(>F)

group 1 8.1727 0.005196 \*\*

98

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

>

> t.test(Sepal.Length~Species,data = a,var.equal=F,mu=-0.7,

+ alternative="less")

Welch Two Sample t-test

data: Sepal.Length by Species

t = -2.602, df = 86.538, p-value = 0.005451

alternative hypothesis: true difference in means is less than -0.7

95 percent confidence interval:

-Inf -0.7830302

sample estimates:

mean in group setosa mean in group versicolor

5.006 5.936

Aradaki fark dikkat edersen setosa-versicolor bu yüzden fark negatif çıktı.

H0=gruplar arasındaki fark -0.7’ye eşit veya büyüktür.

H1=gruplar arasındaki fark 0.7’den düşüktür.

P<0.05 H0 red H1 kabul aradaki fark -0.7’den düşüktür.

Max güven aralığı noktam -0.78303 yani en fazla fark -0.78303 evet gerçekten -0.7’den küçüktür.

> t.test(Sepal.Length~Species,data = a,var.equal=F,mu=-0.8,

+ alternative="greater")

Welch Two Sample t-test

data: Sepal.Length by Species

t = -1.4707, df = 86.538, p-value = 0.9275

alternative hypothesis: true difference in means is greater than -0.8

95 percent confidence interval:

-1.07697 Inf

sample estimates:

mean in group setosa mean in group versicolor

5.006 5.936

H0: gruplar arasındaki fark -0.8’e eşit veya küçüktür..

H1: gruplar arasındaki fark -0.8’den büyüktür.

p>0.05 H0 kabul H1 red

gruplar arasındaki fark -0.8’den büyük değildir.

Evet gruplar arasındaki fark -0.930 yani -0.8’den küçük bir değerdir.

-1.076 Inf min güven aralığı -1.076 yani -0.8’den büyük değildir.

> wilcox.test(Salary~Division,data = df,mu=1,conf.int=T)

Wilcoxon rank sum test with continuity correction

data: Salary by Division

W = 10059, p-value = 0.02171

alternative hypothesis: true location shift is not equal to 1

95 percent confidence interval:

14.99994 179.99998

sample estimates:

difference in location

90.00003

#H0:gruplar arasındaki fark 1'e eşit.

#H1:gruplar arasındaki fark 1'e eşit değildir.

#p<0.5 H0 red H1 kabul.

# %95 güvenilirlik içerisinde gruplar arasındaki fark 14.99-179.99 arasındadır.

> wilcox.test(Salary~Division,data = df,mu=50,conf.int=T,

+ alternative="less")

Wilcoxon rank sum test with continuity correction

data: Salary by Division

W = 9280.5, p-value = 0.8496

alternative hypothesis: true location shift is less than 50

95 percent confidence interval:

-Inf 165

sample estimates:

difference in location

90.00003

#H0:gruplar arasındaki fark 50’ye eşit veya büyüktür..

#H1:gruplar arasındaki fark 50'den azdır.

#p>0.05 H1 red H0 kabul. Aradaki fark 50’ye eşit veya büyüktür.

# -Inf 165 maksimum noktam 165’tir. 165’e kadar benim güven aralığımdır.

**Boxplot ile aykırı değer silme yöntemi:**

boxplot(a$LungCap)

boxplot.stats(a$LungCap)$out

min(boxplot.stats(a$LungCap)$out)

a=a[a$LungCap<12.493,]

funmodeling paketi ile kategorik ve sürekli değişkenlerin dağılımlarını kontrol etmek.

plot\_num(default)+theme\_classic()

freq(default$student)

**Roc eğrisi oluşturma:**

Auc accuracy demektir.

roc(test\_y ~ predict(glm(default~.,data = training,family = "binomial"),

test\_x,

type="response"),

plot=TRUE,

print.auc=TRUE)

knn(train = train\_x[,-1] ,test = test\_x[,-1],train\_y,k=3)

**Knn Model Tuning :**

knn(train = train\_x[,-1] ,test = test\_x[,-1],train\_y,k=3)

#kategorik olan sütunları -1 diyerek çıkardım.

defaultSummary(data.frame(obs=test\_y,

pred=knn(train = train\_x[,-1] ,test = test\_x[,-1],train\_y,k=3)))

knn\_grid=data.frame(k=1:20)

train(train\_x[,-1],train\_y,

method="knn",

tuneGrid=knn\_grid,

preProc=c("center","scale"))

**conf.it=TRUE güven aralığını verir.**

> wilcox.test(income~default,data = a,conf.int=TRUE)

Wilcoxon rank sum test with continuity correction

data: income by default

W = 1551835, p-value = 0.1663

alternative hypothesis: true location shift is not equal to 0

95 percent confidence interval:

-444.5684 2619.4361

sample estimates:

difference in location

1076.677

Eğer aykırı değerler boxplot’un altındaysa max değerden büyük olanları al

> boxplot.stats(a$balance[a$default=="Yes"])$out

[1] 780.1726 698.5674 652.3971

>

> boxplot(a$balance[a$default=="Yes" & a$balance>780.2])

Eğer aykırı değerler boxplot’un üzerindeyse min değerin altındakileri al

**Homojenlik testi nedir ?**

**İki grup arasındaki farkın test edildiği istatistik testleri homojen varyanslılık varsayımı neyi kontrol eder?**

**Gruplar içerisindeki değişimin birbirine ne kadar benzer olduğunu kontrol eder.**

**Randomforest tune işlemi:**

train(default~.,data = a,

method="rf",

length=14,

trControl=trainControl(method = "cv",

number = 10))

**çok değişkenli normallik testi:**

> a%>%group\_by(default)%>%summarise(cok\_degiskenli\_normallik=mshapiro\_test(cbind(balance,income)))

`summarise()` ungrouping output (override with `.groups` argument)

# A tibble: 2 x 2

default cok\_degiskenli\_normallik$statistic $p.value

*<fct>* *<dbl>* *<dbl>*

1 No 0.990 0.0189

2 Yes 0.969 0.00000381

**Çok değişkenli varyans homojenlik testi:**

Bütün değişkenleri tek tek de kontrol edebilirsin ama bu yöntem daha kolay.

> library(heplots) bu paketi kullan.

> bartlettTests(a[c("balance","income")],a$default)

Bartlett's Tests for Homogeneity of Variance

Chisq df Pr(>Chisq)

balance 48.9105 1 2.679e-12 \*\*\*

income 0.4707 1 0.4926

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> leveneTests(a[c("balance","income")],a$default)

Levene's Tests for Homogeneity of Variance (center = median)

df1 df2 F value Pr(>F)

balance 1 655 50.3896 3.288e-12 \*\*\*

income 1 655 2.2744 0.132

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Varyanslar homojen değildir.

**Not:** Varyans bir değişkenin ne kadar değiştiğini verirken kovaryans 2 tane değişkenin birlikte nasıl değiştiğini verir.

**İki yönlü varyans analizi:**

\* veya + da kullanabilirsin \* her ikisinin kovaryansını da verir.

> summary(aov(income~default+student,data=a))

Df Sum Sq Mean Sq F value Pr(>F)

default 1 2.104e+08 2.104e+08 2.697 0.101

student 1 6.910e+10 6.910e+10 885.668 <2e-16 \*\*\*

Residuals 654 5.102e+10 7.802e+07

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> summary(aov(income~default\*student,data=a))

Df Sum Sq Mean Sq F value Pr(>F)

default 1 2.104e+08 2.104e+08 2.694 0.101

student 1 6.910e+10 6.910e+10 884.704 <2e-16 \*\*\*

default:student 1 2.247e+07 2.247e+07 0.288 0.592

Residuals 653 5.100e+10 7.810e+07

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

**Cochran q testi:**

Non-parametric testtir. 2’den fazlı bağımlı değişken olduğu durumda kullanılır.

* 1. olarak kodlanır.

Baharatlı – sade – çeşnili

H0: Cips çeşitleri eşit derecede tercih edilmektedir.

H1: Cips çeşitleri arasında eşit derecede tercih edilmektedir.

Bir kişiye 3 çeşit cips tattırılır. Ve sonuçlar analiz edilir.

**Mcnemar test:**

Nonparametric testtir. 2 bagımlı kategorik grup arasındaki farkı inceler.

Örneğin müşterilere 2 çeşit tatlı tattırılıyor.

Tuzlu çekirdek – Tuzsuz çekirdek

H0: Tuzlu çekirdek ve Tuzsuz çekirdek arasında fark yoktur.

H1: Tuzlu çekirdek ve Tuzsuz çekirdek arasında fark vardır.

> mcnemar.test(tuzlu,tuzsuz)

McNemar's Chi-squared test with continuity correction

data: tuzlu and tuzsuz

McNemar's chi-squared = 1.5577, df = 1, p-value = 0.212

H0 kabul tuzlu ve tuzsuz çekirdek arasında fark yoktur.

Conf.level=0.90 %90 güvenilirlik düzeyinde analiz et.

> wilcox.test(LungCapData2$Height~LungCapData2$Gender,conf.int=TRUE,conf.level=0.90)

Wilcoxon rank sum test with continuity correction

data: LungCapData2$Height by LungCapData2$Gender

W = 43843, p-value = 7.215e-05

alternative hypothesis: true location shift is not equal to 0

90 percent confidence interval:

-2.999986 -1.000053

sample estimates:

difference in location

-2

KORELASYON TESTLERİ

Pearson : parametric Veriler Normal dağılır.

Kendall : Non-parametric Veriler Normal dağılmaz.

Spearman: Non-parametric Veriler normal dağılmaz. Ve ordinaldir.

**Eksik veri satır oranında:**

Eksik veri atamadan önce aykırı değerleri veri setinden sil.

Eğer ortalamaya dayalı atama yapacaksan ilk önce aykırı değerleri veri setinden çıkar daha sonra değişken bazında(satır bazında değil sütun bazında) Eğer eksik veri oranı çok fazla değilse ortalama atayabilirsin.

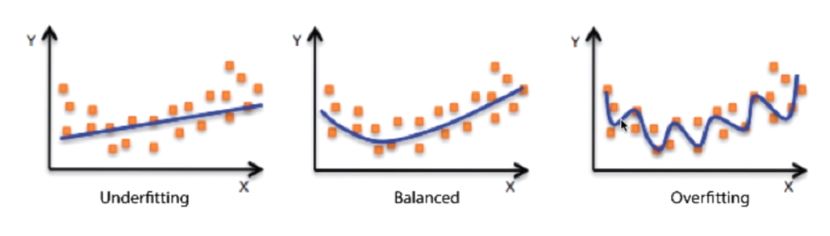
Satır bazında NA değeri oranı

eksik\_veri\_satır\_oranı=data.frame(eksik\_veri\_satır\_oranı=apply(hitters,1,function(x) round(sum(is.na(x))/length(x),2)))

**overtraining:**

overfitting train setini çok iyi tahmin eder fakat dışarıdan gelen veriyi çok iyi tahmin edemez.

aşırı öğrenmedir varyans çok yüksektir. Train verisini %95 tahmin ederken test veri setinde iyi tahminler yapamıyor.



**Bias ve Variance:**

Varyans ve yanlılık:

Underfitting çok kötü tahmin etmesi overfitting çok iyi tahmin etmesi.

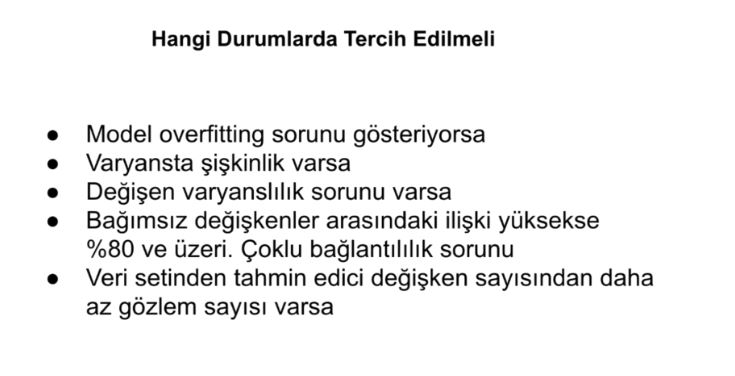
**Ridge regresyon:**

Eger test setim ile train setimin tahmin oranları arasında çok büyük fark varsa yani overtraining varsa ridge regresyon kullanıyorum.

En iyi modeli geliştrimeye çalışıyoruz. Fakat train seti üzerindeki değil test seti üzerindeki tahminlerimi geliştirmeye çalışıyorum.

Lambda değerimi çok iyi ayarlamam gerek. En önemli kısım budur. Bu lambda değerimi cross validation yöntemi ile belirlerim.

Modelin rmse hata kareler ortalamsını düşüren lambda değeridir eğer lambda değerim sıfır olursa normal regresyondan farkım olmaz bu yüzden bu lambda değerimi çok iyi bir şekilde ayarlamam gerekir.



Eğer değişken sayım gözlem sayımdan fazla ise örneğin gen projelerinde değişken sayısı örneklem sayısından fazladır. bu problemlerle karşılaşırsam ridge regresyon kullanacağım.

Ridge regresyonu glm.net paketi ile oluşturacağız.

Diğer regresyon modellerinden farklı olarak veri setini standartlaştırmam gerek.

Standartlaştırmayadabilirim. Bir standartlaştırılmış veri üzerinden bir de standartlaştırılmaış veri üzerinden model kur eğer arada uçurum varsa uygun olanı seç. Yani numeric değişkenlere std ort dönüşümü yap.

preProcess() fonksiyonu değişkenleri standartlaştırır. Method=c(“center”,”scale”)

standartlaştırma işlemleri:

* min-max dönüşümü
* ortalama standart sapma dönüşümü

lm fonksiyonunu kullanırken değişkenimi sadece faktör olarak bıraksam yeter fakat ridge regresyon yaparken bu faktör değişkenlerimi dummy değişkenine dönüştürmem gerekiyor.

model.matrix(LungCap~.,data = d) bu formül ile dummy değişkeni oluşturuyorum.

Bu model matrix fonksiyonu Sepal.Length değişkenini almaz sonradan veri setine Sepal.Length değişkenini de eklemelisin. Yani sadece bağımsız değişkenleri alır.

d=model.matrix(Sepal.Length~.,data = iris)

glmnet(train\_x,train\_y, alpha=0, lambda = 0.5)

Eger ridge regresyon yapmak istersen bu alpha değeri=0 olmalı lambda değişkenimi modeli valide ederken ayarlayacağım.

modelRidge$df demek katsayısı 0 olmayan değişken sayısı demek ki zaten ridge regresyon bu katsayıları sıfır yapmaz lasso regresyon bu katsayıları 0 yapar.

Değişkenlerin beta değerleri yani katsayıları.

> modelRidge$beta

6 x 1 sparse Matrix of class "dgCMatrix"

s0

(Intercept) .

Sepal.Width 0.53911379

Petal.Length 0.38628899

Petal.Width 0.08723965

Speciesversicolor 0.11352677

Speciesvirginica 0.10789241

modelRidge in beta sabiti katsayısı.

> modelRidge$a0

s0

2.586233

Dev.ratio r2 gibi bir değer.

> modelRidge

Call: glmnet(x = train\_x, y = train\_y, alpha = 0, lambda = 0.05)

Df %Dev Lambda

1 5 83.34 0.05

**Ridge Regresyon:**

library(caret)

library(glmnet)

library(tidyverse)

library(corrplot)

#bu d veri setinde sepal length bağımlı değişkeni yoktur sadece bağımsız değişkenleri alır

#bu yüzden bu d veri seti benim bağımsz değişkenlerimi içeriyor.

d=model.matrix(Sepal.Length~.,data = iris)

set.seed(1)

indeks=sample(1:nrow(d),size=0.8\*nrow(d))

train\_x=d[indeks,]

test\_x=d[-indeks,]

train\_y=iris$Sepal.Length[indeks]

test\_y=iris$Sepal.Length[-indeks]

#Ridge regresyon modeli.

modelRidge=glmnet(train\_x,train\_y,alpha = 0,lambda = 0.01)

R2(predict(modelRidge,train\_x),train\_y)

RMSE(predict(modelRidge,train\_x),train\_y)

MAE(predict(modelRidge,train\_x),train\_y)

defaultSummary(data.frame(obs=train\_y,

pred=as.vector(predict(modelRidge,train\_x))))

defaultSummary(data.frame(obs=test\_y,

pred=as.vector(predict(modelRidge,test\_x))))

modelRidge$beta

modelRidge$a0

modelRidge$lambda

train(train\_x,train\_y,

model="ridge",

trControl = trainControl(method = "cv",number = 10))

########## Lambda değeri için cross validation

####### Lambda değerleri oluşturuyorum

lambdas=10^seq(3,-2,by = -0.1)

#### nfolds cross validation numarası

modelRidgecv=cv.glmnet(train\_x,train\_y,alpha = 0,lambda = lambdas,nfolds = 10)

#ideal lambda değerim lambda.min dir.

modelRidgecv$lambda.min

defaultSummary(data.frame(obs=train\_y,

pred=as.vector(predict(modelRidgecv,train\_x))))

defaultSummary(data.frame(obs=test\_y,

pred=as.vector(predict(modelRidgecv,test\_x))))

**Ridge ve Lasso regresyon farkı Ridge regresyon değişkenlerin katsayılarını 0’a yaklaştırırken Lasoo regresyon bu değerleri 0 yapar. Lasso regresyon önemsiz değişkenleri modelden çıkartırken Ridge regresyon sadece bu değişkenlerin değerlerini düşürür.**

Lasso regresyon uygularken çok fazla değişkenimiz oluyor Lasso regresyon bu değişkenlerin önemsiz olanlarını modelden çıkartıyor.

Ridge regresyonda bu katsayıları sıfıra yaklaştırır.



Çok fazla değişkenin varsa gen problemi gibi örneğin 10 tane fare topladık ve bu farelerden bir sürü gen aldık. Değişken sayımız çok fazla ise bu durumda lasso veya ridge regresyon tercih edilir. Daha çok lasso regresyon tercih edilir çünkü değişken sayısı çok fazla lasso regresyon bu düşük katsayıları 0 yapar.

Dosya okuma:

Dosyalar hazır olarak session 🡪 set working directory 🡪 choose directory ile

Dosyayı almak istediğin konumu seçebilirsin.

> read.csv("heart.csv") bu şekilde dosya okunur.

Modelleri karşılaştırırken Residual Deviance değerlerine bakılır residual deviance değeri düşük olan model daha iyi tahminleme yapar.

Değişken önem düzeyleri(Multinomial regresyon)

Caret paketi içerisinde varİmp işlevi ile yapılabilir.

> confusionMatrix(predict(model\_multinomial,test),test$cp)

####Önemli not sınıflandırma problemlerinde hem test hem de

####train veri setlerimde bütün sınıflardan aynı sayıda örneklem alamam gerek.

**Model Tuning işlemi katsayıları düzenler.**

**Tanımlayıcı istatistikler:**

library(psych)

> describeBy(heart$chol,heart$cp)

Descriptive statistics by group

group: 0

vars n mean sd median trimmed mad min max range skew kurtosis se

X1 1 143 250.13 51.54 248 248.1 54.86 131 409 278 0.46 0.26 4.31

-------------------------------------------------------------------------------

group: 1

vars n mean sd median trimmed mad min max range skew kurtosis se

X1 1 50 244.78 43.23 238 243.38 48.93 157 342 185 0.24 -0.71 6.11

-------------------------------------------------------------------------------

group: 2

vars n mean sd median trimmed mad min max range skew kurtosis se

X1 1 87 243.17 60.26 233 238.03 44.48 126 564 438 1.96 8.01 6.46

Ancova da + kullan kovaryans etmek istediğin değişkeni grup değişkeninden önce yaz

İki yönlü varyans analizinde interaction için \* işareti kullan.

**İki Yönlü Anova:**

#####İki yönlü anova:

library(rstatix)

heart%>%group\_by(cp,fbs)%>%summarise(shapiro=shapiro.test(chol)$p.value)

bartlett.test(heart$chol ~ interaction(heart$cp,heart$fbs))

aov(heart$chol ~heart$fbs \* heart$cp)

summary(aov(heart$chol ~heart$fbs \* heart$cp))

anova\_test(data = heart,dv=chol,between = c(fbs,cp))

#####fbs ve cp interation ından birleşiminde nasıl bir sonuc olusturuyor ona bakarım. Cp 3 grup fbs 2 grup 0-0 1-0 1-1 gibi 8 farklı grubun etkileşimine bakarım.

summary(aov(trestbps~fbs\*cp,data = heart))

anova\_test(data=heart,dv=trestbps,between = c(cp,fbs))

summary(aov(trestbps~fbs\*cp\*restecg,data = heart))

İki yönlü anova ve ancova birleşimi:

Yas değişkenini tutarak iki yönlü anova

> summary(aov(chol~ï..age+cp\*fbs,data = heart))

Df Sum Sq Mean Sq F value Pr(>F)

ï..age 1 36614 36614 13.416 0.0003 \*\*\*

cp 2 1079 540 0.198 0.8207

fbs 1 638 638 0.234 0.6290

cp:fbs 2 582 291 0.107 0.8988

Residuals 273 745034 2729

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Yaşı tutmazsak anlamlı:

> summary(aov(chol~cp\*fbs,data = heart))

Df Sum Sq Mean Sq F value Pr(>F)

cp 2 2924 1462.2 0.513 0.599

fbs 1 35 34.5 0.012 0.912

cp:fbs 2 171 85.5 0.030 0.970

Residuals 274 780818 2849.7

**İki yönlü anova:**

####Tukey testi

identify\_outliers(data.frame(heart$trestbps))

df\_new=heart%>%filter(trestbps<172)

df\_new=heart%>%mutate(birlesim=paste(cp,"-",fbs))

###veriler normal dağılmalı.

df\_new%>%group\_by(cp,fbs)%>%

summarise(shapiro=shapiro.test(trestbps)$p.value)

#varyanslar homojen olmalı.

bartlett.test(trestbps~interaction(cp,fbs),data = df\_new)

anova\_1=aov(trestbps~cp\*fbs,data = df\_new)

summary(anova\_1)

TukeyHSD(anova\_1)

tukey\_hsd(anova\_1)

Multinomial Regresyon:

########## Multinomial regresyon:

library(AER)

library(caret)

library(rstatix)

library(nnet)

library(tidyverse)

heart=read.csv("heart.csv")

str(heart)

heart=heart[,-which(names(heart)=="target")]

which(colnames(heart)=="cp")

table(heart$cp)

### cp değişkeninde 3 türü veri sayısı çok az oldugu için modelden çıkarıyorum.

heart=heart[-which(heart$cp==3),]

heart=heart%>%mutate(

cp=as.factor(cp),

slope=as.factor(slope),

ca=as.factor(ca),

thal=as.factor(thal),

restecg=as.factor(restecg))

table(heart$ca)

table(heart$restecg)

table(heart$cp)

str(heart)

#Kategori gözlem sayıları birbirlerine yaklaşık olmalılar.

table(heart[,"cp"])

names(table(heart[,"cp"]))

min(table(heart[,"cp"]))

length(table(heart$cp))

trainTestsplit=function(data,dvName,seed){

tbl=table(data[,dvName])

classes = names(tbl)

minClass=min(tbl)

lengthClass=length(tbl)

train=data.frame()

test=data.frame()

for(i in 1:lengthClass){

selectedClass=data[,dvName]==classes[i]

set.seed(seed)

sampleIndeks=sample(1:nrow(data[selectedClass,]),size=minClass\*0.8)

train=rbind(train,data[selectedClass,][sampleIndeks,])

test=rbind(test,data[selectedClass,][-sampleIndeks,])

}

return(list(train,test))

}

trainTestsplit(heart,"cp",125)[[2]]

train=trainTestsplit(heart,"cp",125)[[1]]

test=trainTestsplit(heart,"cp",125)[[2]]

table(heart$cp)

str(train)

head(train)

table(test$cp)

par(mfrow=c(2,2))

plot(train$cp,train$ï..age,main="Age")

plot(train$cp,train$trestbps,main="Trestbps")

plot(train$cp,train$chol,main="Chol")

plot(train$cp,train$thalach,main="Thalach")

table(train$slope)

chisq.test(train$cp,train$ca)

chisq.test(train$cp,train$slope)

chisq.test(train$cp,train$exang)

chisq.test(train$cp,train$fbs)

chisq.test(train$cp,train$thal)

library(corrplot)

dev.off() #yukarıdaki par(mfrow=c(2,2)) kodunu kapatır..

corrplot(cor(train[,-c(2,3,6,7,9,11,12,13)]))

cor(train$thalach,train$ï..age)

library(e1071)

model\_multinomial=multinom(cp~.,data = train)

model\_multinomial$fitted.values

head(model\_multinomial$fitted.values)

library(Boruta)

Boruta(cp~.,data = heart)

getSelectedAttributes(Boruta(cp~.,data = heart))

defaultSummary(data.frame(obs=train$cp,

pred=predict(model\_multinomial,train)))

defaultSummary(data.frame(obs=train$cp,

pred=predict(multinom(cp~thalach+exang+oldpeak+

slope+ca+thal,data = train),train)))

confusionMatrix(predict(model\_multinomial,test),test$cp)

confusionMatrix(predict(model\_multinomial,train),train$cp)

###modeller üzerinde tahminlemeler.

######### yeni\_test verisi oluşturma.

#setseed alma farklı sonuclar istiyorum çünkü

test[test$cp==0,]

####1 kategorisinden zaten 10 tane oldugu için 1 kategorisini almadım.

yeni\_test=rbind(test[test$cp==1,],

test[test$cp==0,][sample(1:nrow(test[test$cp==0,]),10),],

test[test$cp==2,][sample(1:nrow(test[test$cp==2,]),10),])

table(yeni\_test$cp)

confusionMatrix(predict(model\_multinomial,test),test$cp)

confusionMatrix(predict(model\_multinomial,train),train$cp)

confusionMatrix(predict(model\_multinomial,yeni\_test),yeni\_test$cp,mode = "prec\_recall")

####Önemli not sınıflandırma problemlerinde hem test hem de

####train veri setlerimde bütün sınıflardan aynı sayıda örneklem alamam gerek.

#########Model Tuning işlemi

###Multinomial regresyonda decay parametresi tune edilir.

model\_multinom\_tune=train(train[,-3],train$cp,

method="multinom",

trControl=trainControl(method = "cv",

number = 10))

confusionMatrix(predict(model\_multinom\_tune,train),train$cp)

attributes(model\_multinom\_tune)

model\_multinom\_tune$bestTune

model\_multinom\_tune$finalModel

plot(model\_multinom\_tune)

#####best tune en iyi decay parametre mi buldum yeni bir model oluşturuyorum.

model\_multinom\_tune$bestTune

modelTuned=multinom(cp~.,data = train,decay=0.1)

predict(modelTuned,train)

defaultSummary(data.frame(obs=train$cp,

pred=predict(modelTuned,train)))

defaultSummary(data.frame(obs=yeni\_test$cp,

pred=predict(modelTuned,yeni\_test)))

confusionMatrix(predict(modelTuned,train),train$cp)

confusionMatrix(predict(modelTuned,yeni\_test),yeni\_test$cp)

**Ridge Regresyon modeli:**

######################################################################

############lineer model##############################################

library(caret)

library(car)

library(tidyverse)

library(corrplot)

lm(mpg~.,data = mtcars)

summary(lm(mpg~.,data = mtcars))

set.seed(11)

indeks=sample(1:nrow(mtcars),nrow(mtcars)\*0.75)

train=mtcars[indeks,]

test=mtcars[-indeks,]

##########lineer model###########################

#######çoklu baplantı problemi var #######

corrplot(cor(mtcars[,c(1,3,4,5,6,7)]))

lm\_model=lm(mpg~.,data = train)

summary(lm\_model)

####train veri setim tahmin oranı 0.92

defaultSummary(data.frame(obs=train$mpg,

pred=predict(lm\_model,train)))

####test veri setim tahmin oranı 0.33

defaultSummary(data.frame(obs=test$mpg,

pred=predict(lm\_model,test)))

######## model overfittinge uğradı ridge regresyon yapacağım ##############

library(glmnet)

mtcars\_ridge=model.matrix(~.,data = mtcars)

####intercept değişkenini atıyorumm #########

mtcars\_ridge=mtcars\_ridge[,-1]

view(mtcars\_ridge)

train=mtcars\_ridge[indeks,]

test=mtcars\_ridge[-indeks,]

test\_x=test[,-1]

test\_y=test[,1]

train\_x=train[,-1]

train\_y=train[,1]

model\_Ridge=glmnet(x=train\_x,y=train\_y,alpha = 0,data=train)

#######tune edilmemiş modelimin train tahmin oranı 0.5214 ##########

defaultSummary(data.frame(obs=train\_y,

pred=as.vector(predict(model\_Ridge,train\_x))))

#######tune edilmemiş modelimin test tahmin oranı 0.4585 ##########

defaultSummary(data.frame(obs=test\_y,

pred=as.vector(predict(model\_Ridge,test\_x))))

########lambda parametresini bulabilmek için modeli tune ediyorum #########

model\_ridge\_tune=cv.glmnet()

lambdas=10^seq(-2,2,by=0.05)

model\_ridge\_tune=cv.glmnet(train\_x,train\_y,alpha = 0,lambda = lambdas, nfolds = 10)

model\_ridge\_tune$lambda.min

#######yeni train tahmin oranım 0.8666 #########

defaultSummary(data.frame(obs=train\_y,

pred=as.vector(predict(model\_ridge\_tune,train\_x))))

#######yeni test tahmin oranım 0.8211 ##########

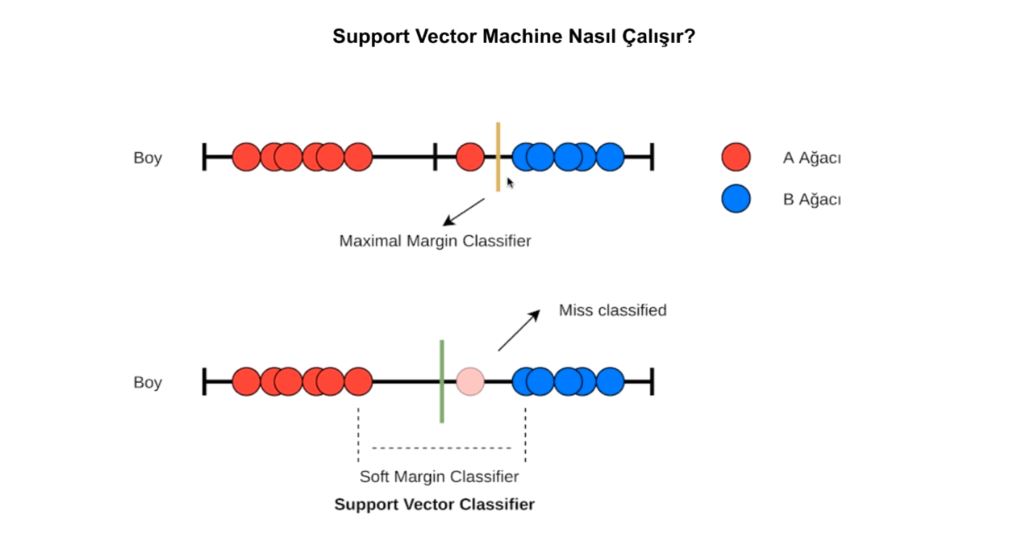
defaultSummary(data.frame(obs=test\_y,

pred=as.vector(predict(model\_ridge\_tune,test\_x))))

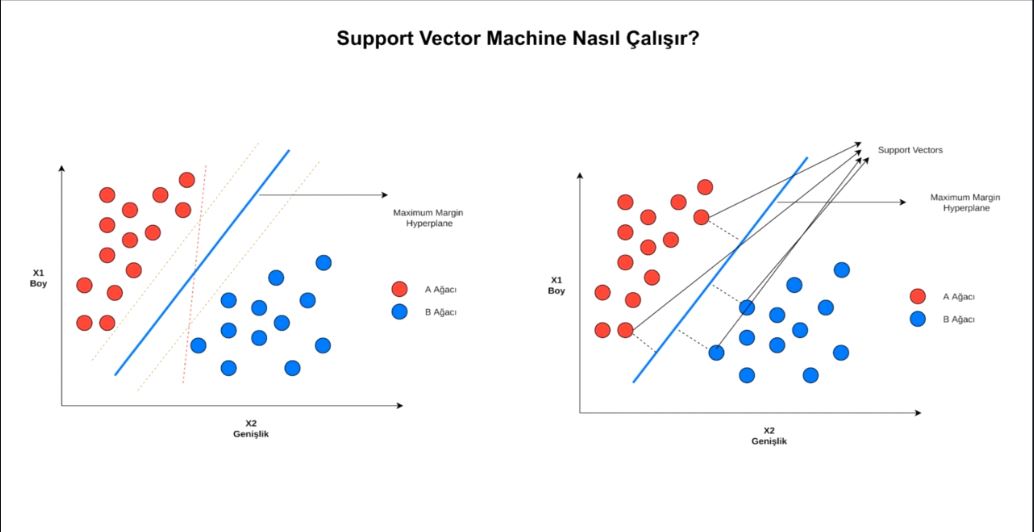
Ridge ve lasso regresyon modelin karmaşıklığını kompleksliliğini azaltır. Katsayıları düşürür.

Ekk yani en küçük kareler yöntemi lineer modeldir.

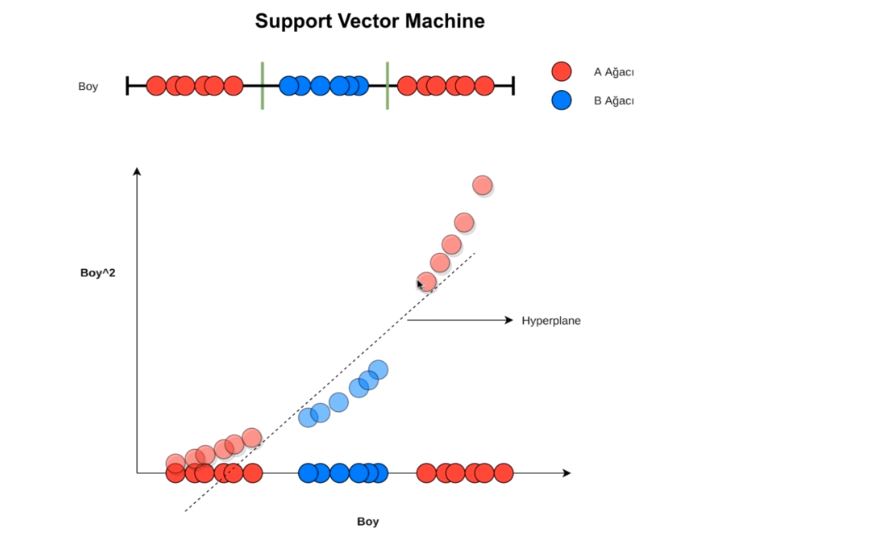
**Destek Vektör Makineleri (SVM):**

****

**Support vector classifier.**

****

İki tane hyperplane olmaz bir tane hyperplane olur.

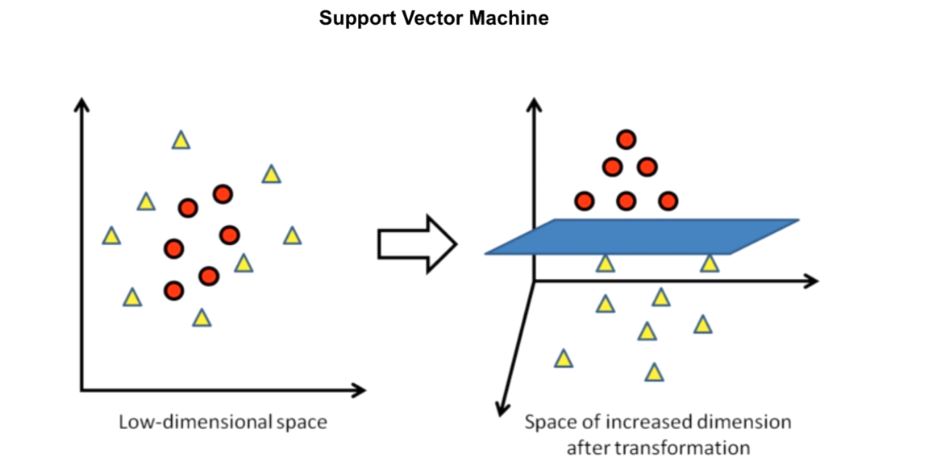


sasasasa

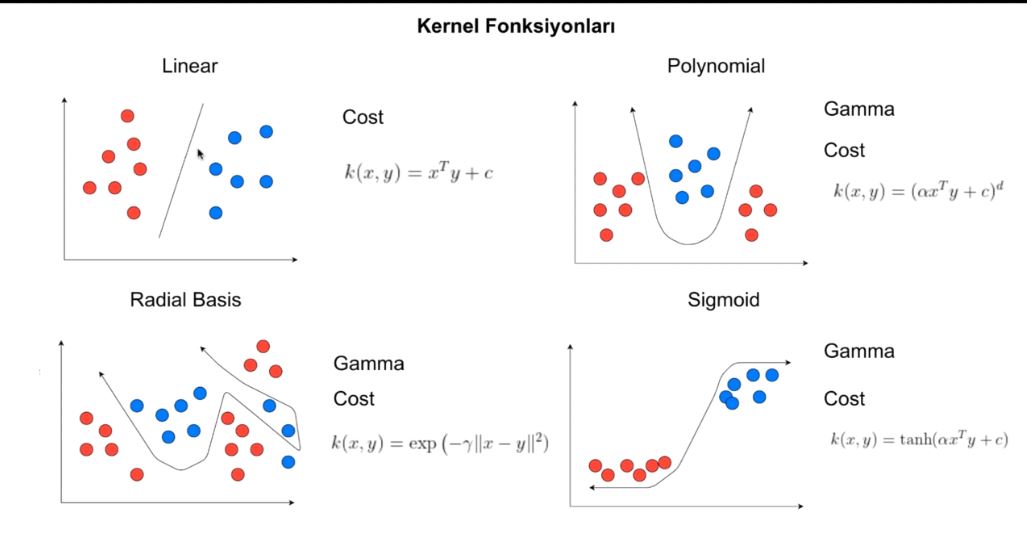
Burada 2 tane hyperplane var bir tane olmalı işte burada support vector machine olur.

İki tane hyperplane olmaz bu yüzden yeni bir değişken oluşturuyorum boyun karesini alıyorum yani boyut arttırıyorum.

sasaasa



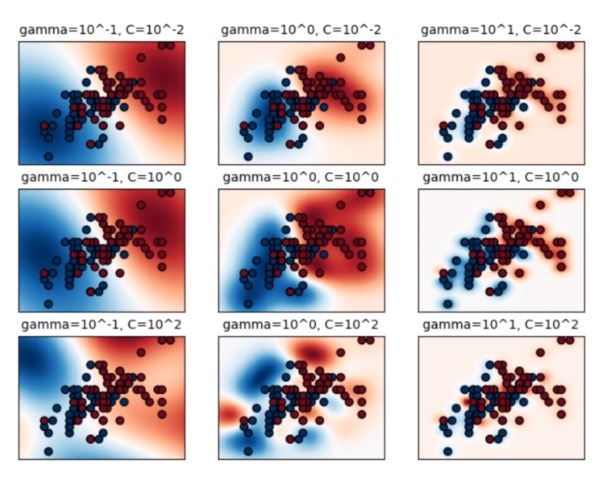
Kernel fonksiyonlarına göre model kurarız:



Bizim ağırlıklı olarak kullanacağımız fonksiyon radial basis olacaktır.

Gamma ve c parametreleri tune edilmeli.

Gamma değeri saga doğru arttıkça model lineerlikten radiallığıa dönüyor.



**SVM PROJESİ :**

İyi ve kötü huylu tümörü tespit etmek için kullanılır.

library(caret)

breast\_cancer=read.csv("breast cancer.csv")

str(breast\_cancer)

modeldata=breast\_cancer[,-c(which(names(breast\_cancer)=="X" | names(breast\_cancer)=="id"))]

str(modeldata)

names(modeldata)

names(modeldata)

selectedVars=c( "diagnosis" ,"radius\_mean", "texture\_mean" ,

"perimeter\_mean" , "area\_mean" , "smoothness\_mean" ,

"compactness\_mean" , "concavity\_mean" , "concave.points\_mean" ,

"symmetry\_mean" , "fractal\_dimension\_mean", "radius\_se" ,

"texture\_se" , "perimeter\_se" , "area\_se" ,

"smoothness\_se" , "compactness\_se" ,"concavity\_se" ,

"concave.points\_se" ,"symmetry\_se" , "fractal\_dimension\_se" )

modeldataSelected=modeldata[,selectedVars]

length(selectedVars)

dim(modelselectedVars)

head(modelselectedVars)

###########Görselleştirmeler ##############

#veriler lineer mi radial mı diye birkaç değişkene bakmak istedim.

##lineer bir çizgi çizsem iyi veya kötü tümorü birbirinden ayırabilirim gibi duruyor.

#radial da deneyebilirim.

plot(modeldataSelected$radius\_mean,

modeldataSelected$texture\_mean,

pch=19,

col=c("blue","orange")[as.factor(modeldataSelected$diagnosis)])

plot(modeldataSelected$area\_mean,

modeldataSelected$perimeter\_mean,

pch=19,

col=c("blue","orange")[as.factor(modeldataSelected$diagnosis)],main = "Grafik")

plot(modeldataSelected$smoothness\_mean,

modeldataSelected$compactness\_mean,

pch=19,

col=c("blue","orange")[as.factor(modeldataSelected$diagnosis)],main = "Grafik")

plot(modeldataSelected$radius\_mean,

modeldataSelected$compactness\_mean,

pch=19,

col=c("blue","orange")[as.factor(modeldataSelected$diagnosis)],main = "Grafik")

#ikili karşılaştırmalara bakıyorum ve radial basis'ın daha mantıklı oldugunu gordum.

#radial veya lineer kullanabiliriz. polinomal ve sigmoid fonksiyonunu kullanmayacağım.

#################Model Oluşturma ####################

library(e1071)

#diagnosis faktöre çevirmeli. kanser iyi huylu mu kötü huylu mu ?

modeldataSelected$diagnosis=as.factor(modeldataSelected$diagnosis)

table(modeldataSelected$diagnosis)

set.seed(125)

trainIndex=sample(1:nrow(modeldataSelected),size= nrow(modeldataSelected) \*0.8)

trainSet=modeldataSelected[trainIndex,]

testSet=modeldataSelected[-trainIndex,]

table(trainSet$diagnosis)

table(testSet$diagnosis)

###kernel fonksiyonu default olarak radial gelir değiştirmek istersen düzeltebilirsin.

##gamma ve cost parametrelerini sonra ayarlayacağız.

modelLinear=svm(diagnosis~.,data = trainSet,kernel="linear")

#56 tane support vector kullanmış hyperplane en yakın noktalar support vectorlerimdir.

#referans alınacak noktalar.

modelRadial=svm(diagnosis~.,data = trainSet,kernel="radial")

#56 tane support vector kullanmış hyperplane en yakın noktalar support vectorlerimdir.

summary(modelLinear)

summary(modelRadial)

#destek vektörlerimin katsayıları regresyonda her bir değişkenin katsayısı vardı burada destek

#vektörlerin katsayıları vardır.

modelRadial$coefs

modelLinear$coefs

#################Modeller üzerinde tahminler. #################

predLinear=predict(modelLinear,testSet)

predRadial=predict(modelRadial,testSet)

defaultSummary(data.frame(obs=testSet$diagnosis,

pred=predLinear))

defaultSummary(data.frame(obs=testSet$diagnosis,

pred=predRadial))

confusionMatrix(predLinear,testSet$diagnosis)

confusionMatrix(predRadial,testSet$diagnosis)

#bu matristeki %95 CI güven aralığını verir.

#bu matristeki kappa istatistiği tahmin edilen ile gercek deger ne kadar benzer.

#bu matristeki mc nemar testi yanlışlar birbirlerine ne kadar benzer.

confusionMatrix(predRadial,testSet$diagnosis,positive = "M")

#positive=m diyerek pozitif classımı belirleyebilirim.

#radial basis fınksiyonu ile kurduğum model B sınıfını yani iyi huylu tümoru daha iyi tahmin etti.

confusionMatrix(predLinear,testSet$diagnosis,mode = "prec\_recall")

confusionMatrix(predRadial,testSet$diagnosis,mode="prec\_recall")

###olasılık türünden tahmin sonucları.

### her bir gözlemi tahmin ettiği olasılığı verir. ##################

modelRadialP=svm(diagnosis~.,data = trainSet,kernel="radial",probability=TRUE)

modelLinearP=svm(diagnosis~.,data = trainSet,kernel="linear",probability=TRUE)

predLinearP=predict(modelLinearP,testSet,probability = T)

predRadialP=predict(modelRadialP,testSet,probability = T)

attr(predLinearP,"probabilities")

plot(modelRadial,testSet,radius\_mean~texture\_mean)

plot(modelLinear,testSet,radius\_mean~texture\_mean)

plot(modelLinear,testSet,perimeter\_mean~area\_mean)

plot(modelRadial,testSet,perimeter\_mean~area\_mean)

######Model tuning işlemi Gamma ve C(cost) parametrelerinin belirlenmesi. #######

##e1071 içerisindeki tune fonksiyonu kullanılır.

modelLinearTune=tune(method = "svm",

diagnosis~.,data = trainSet,

kernel="linear",

ranges = list(gamma=2^(-2:2),

cost=2^(-2:4)),

tunecontrol = tune.control(cross = 5))

modelRadialTune=tune(method = "svm",

diagnosis~.,data = trainSet,

kernel="radial",

ranges = list(gamma=2^(-2:2),

cost=2^(-2:4)),

tunecontrol = tune.control(cross = 5))

##en iyi modeli seçmek için modelLinearTune$best.model seçmem gerek

predLinearTune=predict(modelLinearTune$best.model,testSet)

predRadialTune=predict(modelRadialTune$best.model,testSet)

confusionMatrix(predLinearTune,testSet$diagnosis)

confusionMatrix(predRadialTune,testSet$diagnosis)

Soru :

Hocam bu veri setimizde bağımlı değişkenimizin dışında hiç kategorik veri almadık. bunun sebebi svm algoritmasının mesafe metriğini baz almasından kaynaklı mıdır ?  bu veri setine kategorik değişkenleri direk ekleyebilyoruz dediniz ? bildiğim kadarıyla ağaç tabanlı modellerde bu değişkenleri faktör olarak alabilriz fakat mesafe metriğini baz alan algoritmalarda faktör yerine dummy değişkenleri kullanırız diye biliyorum. ?  dummy değişkenleri ridge lasso regresyon, knn  ve yapay sinir ağlarında kullanılıyor diye biliyorum bunların dışında nerelerde kullanırız  ?

Çooook teşekkür ederim....

Cevap:

Merhabalar,  
  
Tabii ki kategorik verileri de kullanabiliriz. Bu kısımda şunu unutmamak gerek; Doğrusal Regresyon gibi makine öğrenmesi modellerinde verdiğimiz faktör değişkenleri zaten fonksiyon tarafından otomatik olarak Dummy değişkenlere çevriliyor. Mantık olarak zaten çevrilmesi gerek yoksa kategorik değerler ile model kurulamaz sayısal olması gerek. Tabii bu durum Karar Ağaçlarında farklı çünkü onda kurulan modellerde kategorik değişkendeki frekanslardan yola çıkılarak olasılık değeri bulunur. Ama dediğim gibi doğrusal regresyon, svm, neural networks bunlarda verilerin sayısal olması gerek. Kimi R fonksiyonları bunu otomatik olarak çevirirken kimileri bizim çevirip o şekilde modele eklememizi isteyebilir. Sanırım svm fonksiyonu da otomatik olarak çeviriyor. Eğer çevirmiyorsa biz de mode.matrix veya model.frame fonksiyonlarını kullanarak çevirebiliriz.

Bu çevirme işleminde tek dikkat edilmesi gereken kısım. Intercept'in atılması yani formülün sağ tarafı -1 olması. Örneğin aşağıdaki kod gibi. Çünkü SVM intercept'i kullanmıyor.

1. mf <- model.frame(y ~ . - 1, train)

**Karar agacları çookk onemliiiii:**

## Boyle çalıştırırsam olasılık verir.

predict(ModelEnropi,testSet)

## ben sınıfları tahmin etmesini istiyorum.

predict(ModelEnropi,testSet,type = "class")

veri setindeki toplam aykırı değer sayısı

s=0

for(i in 1:length(colnames(a))){

if(is.numeric(a[,i])==TRUE){

print(length(boxplot.stats(a[,i])$out))

s=s+length(boxplot.stats(a[,i])$out)

}

}

print(s)

Karar Ağaçlarında Tahminleme yaparken type fonksiyonunu kullan.

> predict(tree(Outcome~.,data = diabeties),diabeties)

0 1

1 0.3913043 0.60869565

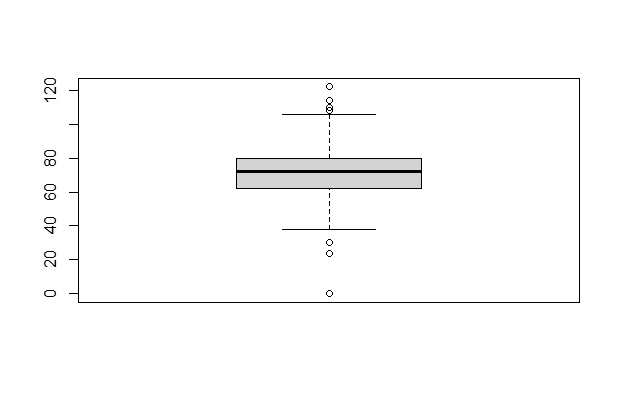
2 0.8181818 0.18181818

Class belirlemezsen olasılık verir.

> predict(tree(Outcome~.,data = diabeties),diabeties,type = "class")

[1] 1 0 1 0 1 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 1

Class belirlersen değer verir.



Bu şekilde alttan ve üstten aykırı değere sahipsen

> table(boxplot.stats(diabeties$BloodPressure)$out)

0 24 30 108 110 114 122

35 1 2 2 3 1 1

Bu fonksiyonu kullan görüldüğü gibi aykırı olmayan değerler 30 ve 108 arasındaki değerler.

Bunları almak için de

> diabeties[diabeties$BloodPressure>30 & diabeties$BloodPressure<108,]

Bu kodu yazarız. Bu değerleri dolduracaksak eğer aykırı değerlerin dışındaki verinin ortalamasını veya medyanını kullan.

Bu şekilde

> mean(diabeties[diabeties$BloodPressure>30 & diabeties$BloodPressure<110,]$BloodPressure)

[1] 72.30759

Aykırı değerleri bu aykırı olmayan değerlerin ortalamasıyla doldurabilirim daha mantıklı

Yapay sinir ağları Derin Öğrenme nesne tanımlama Projesi:

library(keras)

library(EBImage)

#read images

#kayıt yerime fotoğrafları attım.

#setwd() getwd().

#kayıt yerimi burası yaptım.

setwd('/Users/hatun/Documents/resimler')

pics=c("p1.jpg","p2.jpg","p3.jpg","p4.jpg","p5.jpg","p6.jpg",

"c1.jpg","c2.jpg","c3.jpg","c4.jpg","c5.jpg","c6.jpg")

mypic=list()

for(i in 1:12){

mypic[[i]]=readImage(pics[i])

}

#Explore

print(mypic[[1]])

#resimleri ekrana yazdırır.

display(mypic[[8]])

summary(mypic[[1]])

#pixelimin h,stogramı.

hist(mypic[[1]])

#buradan bakarsak bu resimlerin boyutları farklıbunları aynı boyuta getirmem gerek.

str(mypic)

#resize.

for(i in 1:12){

mypic[[i]]=resize(mypic[[i]],28,28)

}

#evet işte şimdi bu resimlerimi boyutlarını eşitleyebildim.

display(mypic[[1]])

display(mypic[[2]])

#evet buradada görüldüğü üzere bütün resimlerim aynı boyutta.

str(mypic)

#reshape.

28\*28\*3

#keras paketiyle yeniden şekillendirme yapıyorum.

for(i in 1:12){

mypic[[i]]=array\_reshape(mypic[[i]],c(28,28,3))

}

str(mypic)

#satır birleştirme yapacağım çünkü bunlar listede Row bind##

trainx=NULL

for(i in 1:5){trainx=rbind(trainx,mypic[[i]])}

#6. ve 12. verilerimi test olarak alacağım.

for(i in 7:11){trainx=rbind(trainx,mypic[[i]])}

str(trainx)

testx=rbind(mypic[[6]],mypic[[12]])

# 0 uçak 1 araba.

#train setimin etiketlerini gireceğim.

trainy=c(0,0,0,0,0,1,1,1,1,1)

#testime 6. ve 12. yi aldım 6. araba 12. uçak.

testy=c(0,1)

#one hot endcoing.

trainLabels=to\_categorical(trainy)

testLabels=to\_categorical(testy)

#Model oluşturuyorum katman sayıları belirliyorum hidden layerları.

model=keras\_model\_sequential()

dim(trainx) #2352 adet sutunum var bu yuzdem girdi şeklimi 28\*28\*3 = 2352 diye tanımlıyorum

# 3 tane katman oluşturdum ilkinde 256 nöron var ikincisinde 128 nöron var.

#en sonunda 2 tane çıktı değişkenim oldugu için units 2 olarak belirledim.

model%>%

layer\_dense(units = 256,activation = "relu",input\_shape = c(2352))%>%

layer\_dense(units = 128,activation = "relu")%>%

layer\_dense(units = 2,activation = "softmax")

summary(model)

#compile modeli derliyorum.

#sonuc değişkenim 2 sonuclu oldugu için binary\_crossentropy diyorum.

model%>%compile(loss="binary\_crossentropy",

optimizer=optimizer\_rmsprop(),

metrics=c("accuracy"))

#fit model

#validation split verinin %80'ini train %20'sini test için kullan.

history=model%>%fit(trainx,

trainlabels,

epochs=30,

batch\_size=32,

validation\_split=0.2)

plot(history)

#evulation ve tahminleme train seti için.

# %90 tahminleyebildi.

model%>%evaluate(trainx,trainLabels)

pred=model%>%predict\_classes(trainx)

table(tahmin=pred,gercek= trainy)

#olasılıkları görelim.

prob=model%>%predict\_proba(trainx)

prob

cbind(prob,tahmin=pred,gercek=trainy)

plot(model)

display(mypic[[11]])

#evaluation test hatası tahminleme

model%>%evaluate(testx,testLabels)

pred\_test=model%>%predict\_classes(testx)

table(Tahminlenen=pred\_test,gercek=testy)

display(mypic[[6]])

display(mypic[[12]])

##### R ile derin öğrenme.

library(keras)

library(EBImage)

#read images

#kayıt yerime fotoğrafları attım.

#setwd() getwd().

#kayıt yerimi burası yaptım.

setwd('/Users/hatun/Documents/resimler')

pics=c("p1.jpg","p2.jpg","p3.jpg","p4.jpg","p5.jpg","p6.jpg","p7.jpg",

"c1.jpg","c2.jpg","c3.jpg","c4.jpg","c5.jpg","c6.jpg","c7.jpg")

mypic=list()

for(i in 1:14){

mypic[[i]]=readImage(pics[i])

}

#Explore

display(mypic[[7]])

display(mypic[[14]])

display(mypic[[3]])

print(mypic[[2]])

#resimleri ekrana yazdırır.

display(mypic[[8]])

summary(mypic[[1]])

#pixelimin histogramı.

hist(mypic[[2]])

#buradan bakarsak bu resimlerin boyutları farklıbunları aynı boyuta getirmem gerek.

str(mypic)

#resize.

for(i in 1:14){

mypic[[i]]=resize(mypic[[i]],28,28)

}

#evet işte şimdi bu resimlerimi boyutlarını eşitleyebildim.

display(mypic[[1]])

display(mypic[[2]])

display(mypic[[7]])

#evet buradada görüldüğü üzere bütün resimlerim aynı boyutta.

str(mypic)

#reshape.

28\*28\*3

#keras paketiyle yeniden şekillendirme yapıyorum.

for(i in 1:14){

mypic[[i]]=array\_reshape(mypic[[i]],c(28,28,3))

}

str(mypic)

#satır birleştirme yapacağım çünkü bunlar listede Row bind##

trainx=NULL

for(i in 1:5){trainx=rbind(trainx,mypic[[i]])}

#6. ve 12. verilerimi test olarak alacağım.

for(i in 8:12){trainx=rbind(trainx,mypic[[i]])}

str(trainx)

testx=rbind(mypic[[6]],mypic[[7]],mypic[[13]],mypic[[14]])

# 0 uçak 1 araba.

#train setimin etiketlerini gireceğim.

trainy=c(0,0,0,0,0,1,1,1,1,1)

#testime 6. 7. ve 13. ve 14 yi aldım 6. uçak 7. uçak. 13. 14 araba

testy=c(0,0,1,1)

#one hot endcoing.

trainLabels=to\_categorical(trainy)

testLabels=to\_categorical(testy)

#Model oluşturuyorum katman sayıları belirliyorum hidden layerları.

model=keras\_model\_sequential()

dim(trainx) #2352 adet sutunum var bu yuzdem girdi şeklimi 28\*28\*3 = 2352 diye tanımlıyorum

# 3 tane katman oluşturdum ilkinde 256 nöron var ikincisinde 128 nöron var.

#en sonunda 2 tane çıktı değişkenim oldugu için units 2 olarak belirledim.

model%>%

layer\_dense(units = 256,activation = "relu",input\_shape = c(2352))%>%

layer\_dense(units = 128,activation = "relu")%>%

layer\_dense(units = 2,activation = "softmax")

summary(model)

#compile modeli derliyorum.

#sonuc değişkenim 2 sonuclu oldugu için binary\_crossentropy diyorum.

model%>%compile(loss="binary\_crossentropy",

optimizer=optimizer\_rmsprop(),

metrics=c("accuracy"))

#fit model

#validation split verinin %80'ini train %20'sini test için kullan.

#modeli eğitiyorum yani valide ediyorum.

history=model%>%fit(trainx,

trainLabels,

epochs=30,

batch\_size=32,

validation\_split=0.2)

plot(history)

#evulation ve tahminleme train seti için. eğitim hatası

# %90 tahminleyebildi.

model%>%evaluate(trainx,trainLabels) #modelin egitim üzerindeki performansı.

pred=model%>%predict\_classes(trainx) #egitim verisini tahmin edelim.

table(tahmin=pred,gercek= trainy)

#olasılıkları görelim.

prob=model%>%predict\_proba(trainx)

prob

cbind(prob,tahmin=pred,gercek=trainy)

plot(model)

display(mypic[[6]])

display(mypic[[7]])

display(mypic[[13]])

#evaluation test hatası tahminleme

model%>%evaluate(testx,testLabels)

pred\_test=model%>%predict\_classes(testx)

table(Tahminlenen=pred\_test,gercek=testy)

display(mypic[[6]])

display(mypic[[12]])

display(mypic[[13]])

**Derin Öğrenme Kategorical 3 sınıflı yani binary değil:**

library(keras)

library(EBImage)

setwd('/Users/hatun/Documents/resimler')

pics=c("p1.jpg","p2.jpg","p3.jpg","p4.jpg","p5.jpg","p6.jpg","p7.jpg",

"c1.jpg","c2.jpg","c3.jpg","c4.jpg","c5.jpg","c6.jpg","c7.jpg")

ev=paste0("ev",c(1:7),".jpg")

pics=c(pics,ev)

length(pics)

mypic=list()

for(i in 1:21){

mypic[[i]]=readImage(pics[i])

}

display(mypic[[20]])

display(mypic[[18]])

#Explore

display(mypic[[7]])

display(mypic[[14]])

display(mypic[[3]])

print(mypic[[2]])

#resimleri ekrana yazdırır.

display(mypic[[8]])

display(mypic[[12]])

summary(mypic[[1]])

#pixelimin histogramı.

hist(mypic[[2]])

hist(mypic[[19]])

#buradan bakarsak bu resimlerin boyutları farklıbunları aynı boyuta getirmem gerek.

str(mypic)

#resize.

for(i in 1:21){

mypic[[i]]=resize(mypic[[i]],28,28)

}

display(mypic[[19]])

#evet işte şimdi bu resimlerimi boyutlarını eşitleyebildim.

display(mypic[[1]])

display(mypic[[2]])

display(mypic[[7]])

#evet buradada görüldüğü üzere bütün resimlerim aynı boyutta.

str(mypic)

#reshape.

28\*28\*3

#keras paketiyle yeniden şekillendirme yapıyorum.

for(i in 1:21){

mypic[[i]]=array\_reshape(mypic[[i]],c(28,28,3))

}

str(mypic)

#satır birleştirme yapacağım çünkü bunlar listede Row bind##

trainx=NULL

for(i in 1:5){trainx=rbind(trainx,mypic[[i]])}

trainx[1]

#6,7 ve 13, 14 20 ve 21 verilerimi test olarak alacağım.

for(i in 8:12){trainx=rbind(trainx,mypic[[i]])}

str(trainx)

for(i in 15:19){trainx=rbind(trainx,mypic[[i]])}

dim(trainx)

testx=rbind(mypic[[6]],mypic[[7]],mypic[[13]],mypic[[14]],mypic[[20]],mypic[[21]])

dim(testx)

# 0 uçak 1 araba. 2 ev

#train setimin etiketlerini gireceğim.

trainy=c(0,0,0,0,0,1,1,1,1,1,2,2,2,2,2)

#testime 6. 7. ve 13. ve 14 20 21 i test olarak aldım.

testy=c(0,0,1,1,2,2)

pics

#one hot endcoing.

trainLabels=to\_categorical(trainy)

testLabels=to\_categorical(testy)

#Model oluşturuyorum katman sayıları belirliyorum hidden layerları.

model=keras\_model\_sequential()

dim(trainx) #2352 adet sutunum var bu yuzdem girdi şeklimi 28\*28\*3 = 2352 diye tanımlıyorum

# 3 tane katman oluşturdum ilkinde 256 nöron var ikincisinde 128 nöron var.

#en sonunda 2 tane çıktı değişkenim oldugu için units 2 olarak belirledim.

model%>%

layer\_dense(units = 256,activation = "relu",input\_shape = c(2352))%>%

layer\_dense(units = 128,activation = "relu")%>%

layer\_dense(units = 3,activation = "softmax")

summary(model)

#compile modeli derliyorum.

#sonuc değişkenim 2 olsaydı kayıp fonksiyonumu yani loss= binary\_crossentropy diyecektim.

#fakat 3 tane sınıfım oldugu için loss=categorical\_crossentropy diyorum.

model%>%compile(loss="categorical\_crossentropy",

optimizer=optimizer\_rmsprop(),

metrics=c("accuracy"))

#fit model

#validation split verinin %80'ini train %20'sini test için kullan.

#modeli eğitiyorum yani valide ediyorum. batch görüntü sayım 32 de girebilirim.

#şu an kayıp fonksiyonumu iyileştirmeye çalışıyorum.

history=model%>%fit(trainx,

trainLabels,

epochs=30,

batch\_size=21,

validation\_split=0.2)

plot(history)

#evulation ve tahminleme train seti için. eğitim hatası

# %80 tahminleyebildi.

model%>%evaluate(trainx,trainLabels) #modelin egitim üzerindeki performansı.

# bence gerçek hayat projeleri için güzel bir rakam

#biz de zaten bu resimleri internetten kendimiz aldık hazır veri seti değil

#bence çok iyi bir olasılık

pred=model%>%predict\_classes(trainx) #egitim verisini tahmin edelim.

table(tahmin=pred,gercek= trainy)

#olasılıkları görelim.

prob=model%>%predict\_proba(trainx)

prob

cbind(prob,tahmin=pred,gercek=trainy)

display(mypic[[6]])

display(mypic[[7]])

display(mypic[[13]])

#evaluation test hatası tahminleme

model%>%evaluate(testx,testLabels)

pred\_test=model%>%predict\_classes(testx)

table(Tahminlenen=pred\_test,gercek=testy)

# biraz overfitting problemi var gibi daha fazla görüntü vermem gerek.

# modele daha fazla görüntü vermem gerek.

display(mypic[[6]])

display(mypic[[12]])

display(mypic[[13]])

> paste("p",c(1:5),".jpg")

[1] "p 1 .jpg" "p 2 .jpg" "p 3 .jpg" "p 4 .jpg" "p 5 .jpg"

> paste0("p",c(1:5),".jpg")

[1] "p1.jpg" "p2.jpg" "p3.jpg" "p4.jpg" "p5.jpg"

Paste boşluklubirleştirir. Paste0 ise boşluksuz birleştirir.

> starwars%>%select(name,height,mass)%>%

+ rename(weight=mass,

+ isim=name)

# A tibble: 87 x 3

isim height weight

*<chr>* *<int>* *<dbl>*

1 Luke Skywalker 172 77

2 C-3PO 167 75

> LungCapData2%>%

+ select(Age,LungCap,Gender)%>%

+ rename(yas=Age,Akciger\_Kap=LungCap,cinsiyet=Gender)%>%

+ mutate(cinsiyet=recode(cinsiyet,

+ male="erkek",

+ female="kadın"))

# A tibble: 654 x 3

yas Akciger\_Kap cinsiyet

*<dbl>* *<dbl>* *<fct>*

1 9 3.12 kadın

2 8 3.17 kadın

3 7 3.16 kadın

4 9 2.67 erkek

> starwars%>%select(name,gender,hair\_color,height)%>%

+ drop\_na(hair\_color)

# A tibble: 82 x 4

name gender hair\_color height

*<chr>* *<chr>* *<chr>* *<int>*

1 Luke Skywalker masculine blond 172

2 Darth Vader masculine none 202

Eksik veri doldurma.

starwars%>%select(name,mass,hair\_color,gender)%>%

mutate(hair\_color=replace\_na(hair\_color,"none"))%>%

mutate(mass=replace\_na(mass,mean(starwars$mass,na.rm = T)))%>%

mutate(gender=replace\_na(gender,"cinsiyetsiz"))

starwars%>%select(name,mass,hair\_color,gender)%>%

mutate(hair\_color=replace\_na(hair\_color,"none"))%>%

mutate(mass=replace\_na(mass,mean(starwars$mass,na.rm = T)))%>%

mutate(gender=replace\_na(gender,"cinsiyetsiz"))%>%

rename(sac\_rengi=hair\_color)%>%

mutate(sac\_rengi=recode(sac\_rengi,

none="bilinmeyen"))

**r dan excele veri aktarımı:**

isim="harun bakırcı excele dsya aktarımı"

isim

write.table(substr(isim,2,100),'excel\_data\_at.csv',sep = ",")

**iris veri seti shiny:**

# Import libraries

library(shiny)

library(data.table)

library(randomForest)

library(shinythemes)

# Read in the RF model

model <- readRDS("model.rds")

####################################

# User interface #

####################################

ui <- fluidPage(

theme = shinytheme("darkly"),

shinythemes::themeSelector(),

# Page header

p(h1(strong('Harun BAKIRCI ile Veri Analizi'))),

# Input values

sidebarPanel(

img(src = "dank.png",height = 50,width = 200),

br(),

br(),

h3('Girdi Degiskenleri'),

numericInput("Sepal.Length",

label = "Sepal Length",

value = 5.1),

numericInput("Sepal.Width",

label = "Sepal Width",

value = 3.6),

numericInput("Petal.Length",

label = "Petal Length",

value = 1.4),

numericInput("Petal.Width",

label = "Petal Width",

value = 0.2),

actionButton("submitbutton", "Tahmin Et",

class = "btn btn-primary"),

),

mainPanel(

img(src = "harun.png",height = 140,width = 140,Position="right"),

br(),

tags$label(h3('Status/Output')), # Status/Output Text Box

verbatimTextOutput('contents'),

tableOutput('tabledata'),

HTML('<iframe width="500" height="300" src="https://www.youtube.com/embed/IEeF7UgrW9w" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe>')

# Prediction results table

)

)

####################################

# Server #

####################################

server<- function(input, output, session) {

# Input Data

datasetInput <- reactive({

df <- data.frame(

Name = c("Sepal Length",

"Sepal Width",

"Petal Length",

"Petal Width"),

Value = as.character(c(input$Sepal.Length,

input$Sepal.Width,

input$Petal.Length,

input$Petal.Width)),

stringsAsFactors = FALSE)

Species <- 0

df <- rbind(df, Species)

input <- transpose(df)

write.table(input,"input.csv", sep=",", quote = FALSE, row.names = FALSE, col.names = FALSE)

test <- read.csv(paste("input", ".csv", sep=""), header = TRUE)

Output <- data.frame(Prediction=predict(model,test), round(predict(model,test,type="prob"), 3))

print(Output)

})

# Status/Output Text Box

output$contents <- renderPrint({

if (input$submitbutton>0) {

isolate("Hesaplama Tamamlandi.")

} else {

return("Sunucu Hesaplama icin Hazir")

}

})

# Prediction results table

output$tabledata <- renderTable({

if (input$submitbutton>0) {

isolate(datasetInput())

}

})

}

####################################

# Create the shiny app #

####################################

shinyApp(ui = ui, server = server)

**Modelin kodları:**

# Importing libraries

library(RCurl) # for downloading the iris CSV file

library(randomForest)

library(caret)

# Importing the Iris data set

iris <- read.csv(text = getURL("https://raw.githubusercontent.com/dataprofessor/data/master/iris.csv") )

# Performs stratified random split of the data set

TrainingIndex <- createDataPartition(iris$Species, p=0.8, list = FALSE)

TrainingSet <- iris[TrainingIndex,] # Training Set

TestingSet <- iris[-TrainingIndex,] # Test Set

write.csv(TrainingSet, "training.csv")

write.csv(TestingSet, "testing.csv")

TrainSet <- read.csv("training.csv", header = TRUE)

TrainSet <- TrainSet[,-1]

# Building Random forest model

model <- randomForest(Species ~ ., data = TrainSet, ntree = 500, mtry = 4, importance = TRUE)

# Save model to RDS file

saveRDS(model, "model.rds")

sapply(xtrain, class)

sapply(xtest, class)

yeni bir dataframe oluştururken as.dataframe değilde dataframe olarak atama yap.

df <- data.frame (isim = c ("Jon", "Bill", "Maria"),

yaş = c (23,41,32))

library(writexl)

s=data.frame(a=c(10),

b=c(10),

c=100,

d="no",

e="male",

f="no")

names(LungCap)=c("LungCap","Age","Height","Smoke","Gender","Cancer")

names(s)=c("LungCap","Age","Height","Smoke","Gender","Cancer")

setwd("C:/Users/Harun/Desktop/LungCap Tahmin")

write\_xlsx(s,"LungCap.xlsx")

> iris%>%mutate(Virginica=ifelse(species=="virginica",1,0),

+ Setosa=ifelse(species=="setosa",1,0),

+ Versicolor=ifelse(species=="versicolor",1,0))

# A tibble: 150 x 8

sepal\_length sepal\_width petal\_length petal\_width species Virginica Setosa Versicolor

<dbl> <dbl> <dbl> <dbl> <fct> <dbl> <dbl> <dbl>

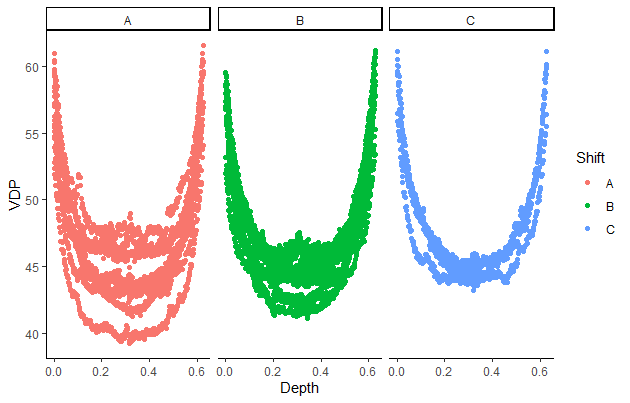
1 5.1 3.5 1.4 0.2 setosa 0 1 0

2 4.9 3 1.4 0.2 setosa 0 1 0

3 4.7 3.2 1.3 0.2 setosa 0 1 0

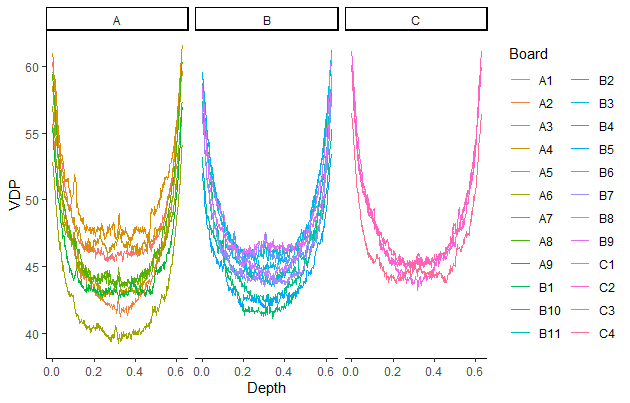
ggplot(VDP)+geom\_point(aes(x=Depth,y=VDP,color=Shift))+theme\_classic()+

facet\_grid(vars(),vars(Shift))



ggplot(VDP)+geom\_line(aes(x=Depth,y=VDP,color=Board))+theme\_classic()+facet\_grid(vars(),

vars(Shift))



Overfitting problemi :

Overfitting problemi varsa gözlem sayısını arttır.

Your dataset is quite small to train a [machine learning](https://intellipaat.com/blog/what-is-machine-learning/) model properly. You should collect more data, then you would have less chance to overfit. The adequate amount of data helps machine learning models to find patterns easily.

There are some parameters of random forest that can be tuned for the model’s better performance.

* **n\_estimators**: The more trees, the less likely the algorithm is to overfit. So try increasing this parameter. The lower this number, the closer the model is to a decision tree, with a restricted feature set.
* **max\_features**: You should try reducing this number. This defines how many features each tree is randomly assigned.
* **max\_depth**: This parameter will reduce the complexity of the learned models, lowering over fitting risk.
* **min\_samples\_leaf**: Try setting these values greater than one. This has a similar effect to the max\_depth parameter, it means the branch will stop splitting once the leaves have that number of samples each.

Kappa İstatistiği:

**Cohen'in kappa katsayısı** iki değerleyici arasındaki karşılaştırmalı uyuşmanın güvenirliğini ölçen bir [istatistik](https://tr.wikipedia.org/wiki/%C4%B0statistik) yöntemidir.[[1]](https://tr.wikipedia.org/wiki/Cohen%27in_kappa_katsay%C4%B1s%C4%B1#cite_note-1) Cohen'in kappa ölçüsü her biri *N* tane maddeyi *C* tane birbirinden karşılıklı hariç olan kategoriye ayıran iki değerleyicinin arasında bulunan uyuşmayı ölçer.

fviz\_cluster(cluster\_model,whole\_sales)+theme\_classic()+

geom\_label(label=whole\_sales$sınıf)

> iris\_bul = function(x){

+ iris[iris$species==x,]

+ }

> iris\_bul("setosa")

# A tibble: 50 x 5

sepal\_length sepal\_width petal\_length petal\_width species

<dbl> <dbl> <dbl> <dbl> <fct>

1 5.1 3.5 1.4 0.2 setosa

2 4.9 3 1.4 0.2 setosa

ggplot(iris,aes(x=sepal\_length,y=sepal\_width,color=species,size=petal\_length))+geom\_point()+theme\_classic()+

ggtitle("İRİS Dağılım Grafiği")+xlab("Sepal Length") +ylab("Sepal Width")

Faktör Ve Numeric Değişken Kontrolü :

nonFactorCols <- sapply(df1, function(col) !is.factor(col))

# or, more efficiently

nonFactorCols <- !sapply(df1, is.factor)

# or, even more efficiently

nonFactorCols <- !factorCols

Model Parametreleri :

model=lm(sepal\_length~.,data = iris)

a=predict(model,iris)

R2(a,iris$sepal\_length)

MAE(a,iris$sepal\_length)

RMSE(a,iris$sepal\_length)

caret::varImp(model)

a=c("hsrun","harun","bakirci","mert","baran","istanbul","ankara","istinye")

> grep("ist",a)

[1] 6 8

> ad[c(grep("eneme",ad))]

[1] "denemey" "deneme test"

Ad vektorunun ilk üç basamağını al

> substr(ad,1,3)

[1] "den" "yen" "gay" "mmm" "han" "has" "Flo" "Bak" "bee" "sco" "tes" "Şii" "Tes" "qqq"

[15] "Yaz" "den" "tes" "ttt" "ppp" "qq" "t3" "r2" "ww" "uyy" "ekl" "vv" "Yiğ" "Tes"

rnorm(n = 10,mean = 20,sd = 5)

summary(lm(Fresh~. -1 ,data = tum\_satislar))

-1 yazarsam katsayıyı almaz.

> matrix(c(3:26),4)

[,1] [,2] [,3] [,4] [,5] [,6]

[1,] 3 7 11 15 19 23

[2,] 4 8 12 16 20 24

[3,] 5 9 13 17 21 25

[4,] 6 10 14 18 22 26

> seq(from = 1,to = 5,by = 0.5)

[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0

filter(flights, month %in% c(11, 12))

filter(flights, !(arr\_delay > 120 | dep\_delay > 120))

başlığı “del” içeren kolonları seç

> flights%>%select(contains("del"))

dep\_delay arr\_delay

<dbl> <dbl>

1 2 11

2 4 20

3 2 33

4 -1 -18

5 -6 -25

Dep ile başlayan sütunları seç

> flights%>%select(starts\_with("dep"))

# A tibble: 336,776 x 2

dep\_time dep\_delay

<int> <dbl>

1 517 2

2 533 4

3 542 2

is\_not =data.frame(

isim=c("harun","ahmet"),

not = c(10,20)

)

İsim kolonunu adlar olarak değiştirdim

is\_not%>%rename(adlar= isim)

air\_time ve time\_hour değişkenlerini başa alır diğerlerini ardına.

flights%>%select(air\_time,time\_hour,everything())

flights%>%select(starts\_with("time"),contains("dep"))

time\_hour dep\_time sched\_dep\_time dep\_delay

<dttm> <int> <int> <dbl>

1 2013-01-01 05:00:00 517 515 2

2 2013-01-01 05:00:00 533 529 4

> 10%%3 %% mode operatörü kalanı verir

[1] 1

> 10%/%3 %/% içerisinde kaç tane 3 olduğunu verir.

[1] 3

> 10%/%4

[1] 2

> cumsum(sayilar)

[1] 89 126 160 249 293 372 405 489 524 594 668 710 748 768 796

[16] 816 860 947 1017 1057

> round(cummean(sayilar),2)

[1] 89.00 63.00 53.33 62.25 58.60 62.00 57.86 61.12 58.22 59.40 60.73 59.17

[13] 57.54 54.86 53.07 51.00 50.59 52.61 53.53 52.85

Min\_rank veriyi sıralar.

y=c(3,2,4,5,5,8)

min\_rank(y)

cbind(y,sıra = min\_rank(y))

y sıra

[1,] 3 2

[2,] 2 1

[3,] 4 3

[4,] 5 4

[5,] 5 4

[6,] 8 6

b= b[-which(is.na(b$dep\_delay)),] dep\_delay deki na değerleri sil.

Veriyi özetlemek için summarize(dplyr) paketini kullan.

summarize(flights,

delay=mean(dep\_delay,na.rm = TRUE),

min =min(dep\_delay,na.rm = TRUE),

mak = max(dep\_delay,na.rm = TRUE),

std=sd(dep\_delay,na.rm = TRUE)

)

# A tibble: 1 x 4

delay min mak std

<dbl> <dbl> <dbl> <dbl>

1 12.6 -43 1301 40.2

> notcancelled%>%group\_by(year,month,day)%>%summarise(

+ mean=mean(dep\_delay)

+ )

`summarise()` regrouping output by 'year', 'month' (override with `.groups` argument)

# A tibble: 365 x 4

# Groups: year, month [12]

year month day mean

<int> <int> <int> <dbl>

1 2013 1 1 11.4

2 2013 1 2 13.7

3 2013 1 3 10.9

x=sample(1:10,20,replace = TRUE)

cut(x = x, breaks = c(0,4,8,10),labels = c("1","2","3"))

deger donusum

1 9 3

2 9 3

3 10 3

4 3 1

5 5 2

6 1 1

7 3 1

df\_unique=unique(df)

recode(LungCap$Sigara\_durumu, eski değer yeni değer olarak kullanılır.

"yes" = 0,

"no"=1)

[1] 1 0 1 1 1 1 1 1 1 1 1 1 1 1

> apply(a, MARGIN = 2,function(x) normalityTest(~x,test = "lillie.test",data = a))

$oncesi

Lilliefors (Kolmogorov-Smirnov) normality test

data: x

D = 0.10706, p-value = 0.006626

$sonrasi

Lilliefors (Kolmogorov-Smirnov) normality test

data: x

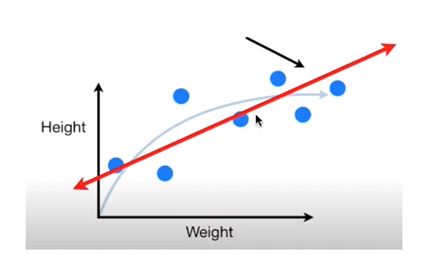
D = 0.11694, p-value = 0.001813

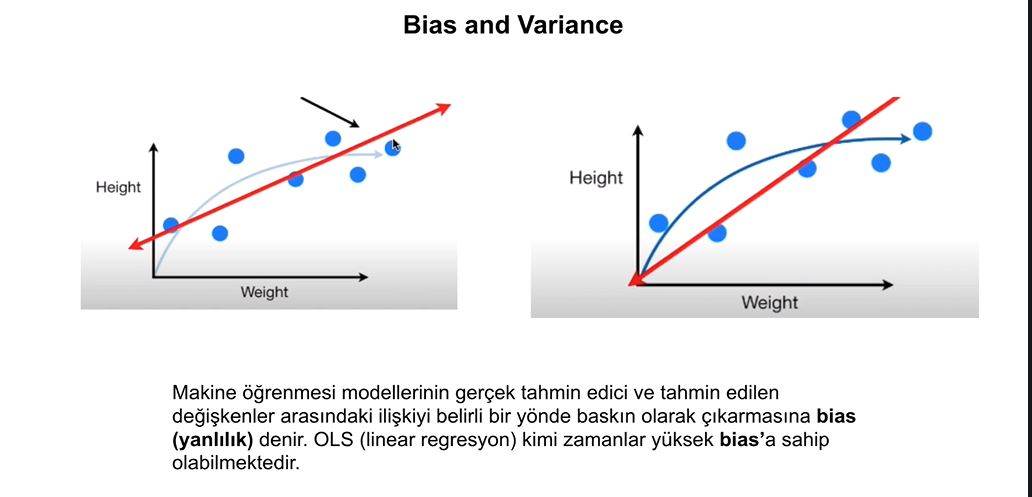
apply(a,MARGIN = 2,function(x) shapiro.test(x))

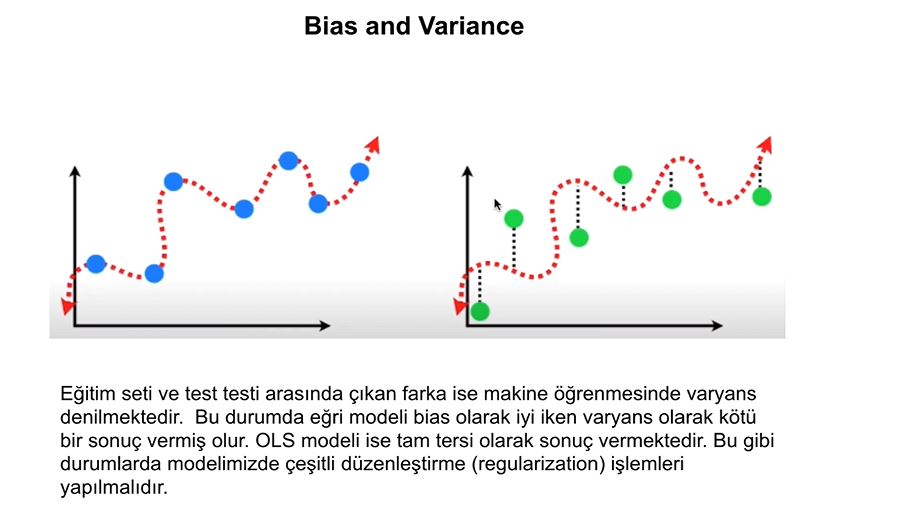
by(lungcap$Boy,lungcap$Cinsiyet,shapiro.test)

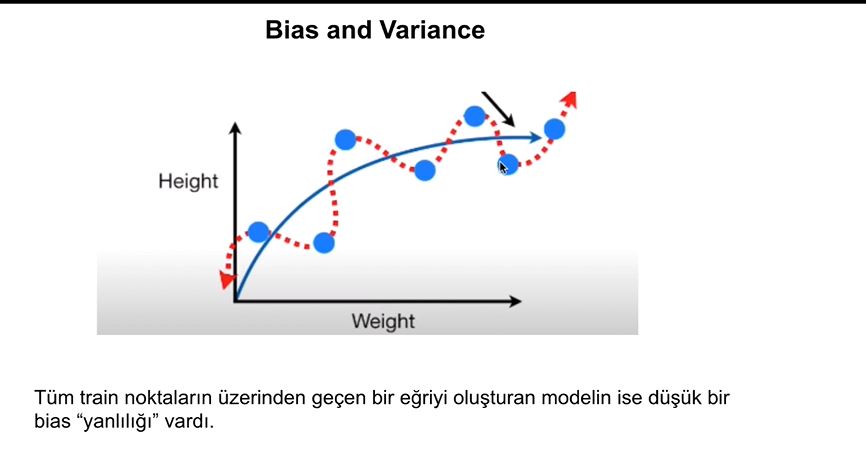
t.test(oncesi,sonrasi,paired = TRUE,var.equal = TRUE,conf.level = 0.95)

EKK en küçük kareler metodu yani regresyon bir eğri çizmez doğru çizer





Regresyon çizgisi her zaman düz olur.



Polinom regresyon

\*\*\*\*\* Önemli

Validasyon model parametrelerini optimize etmek için kullanılır.

Aynı zamanda overfitting yani aşırı öğrenme problemini çözmeye yarar.

Yanlılığı azaltır.

Underfitting demek modelin kötü tahmin etmesi demek. Model eğitim veri setinde iyi tahmin edemiyorsa underfitting olmuş demektir.

**Karar Agaçları :**

Doğrusal değildir.

Hem regresyon hem sınıflandırma problemlerinde kullanılır.

Multiclass problemlere uygundur.

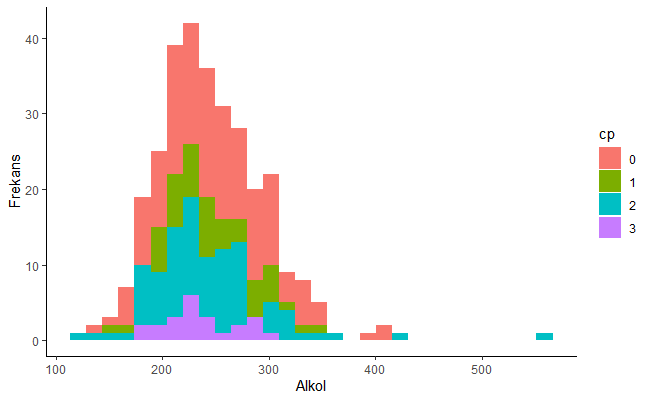
Root karar ağacı en üstü.başlangıç noktası.

Mahalanobis distance çok değişkenli aykırı değer tespiti.

ggplot(heart)+geom\_histogram(aes(x=chol,fill = cp))+

theme\_classic()+

xlab("Alkol")+

 ylab("Frekans")

iki yönlü anova için veriler normal dağılmalı varyanslar eşit olmalı.

# Two-way ANOVA with interaction effect# These two calls are equivalent

res.aov3<-aov(len~supp\*döşe,data=my\_data)

res.aov3<-aov(len~supp+dose+supp:dose,data=my\_data)

summary(res.aov3)

Spss de iki yönlü anova General model -> Univariate kısmında

Manova ise Multivariate kısmından yapılır

Çoklu bağlantı problemini ortadan kaldırır.

Çok yönlü manova da vardır.

Varyanslar homojen olmalı veriler normal dağılmalı.

İki yönlü anova

> summary(aov(age~cp+restecg + cp:restecg ,data = heart))

Df Sum Sq Mean Sq F value Pr(>F)

cp 3 818 272.7 3.487 0.0162 \*

restecg 2 535 267.5 3.420 0.0340 \*

cp:restecg 4 644 161.1 2.060 0.0861 .

Residuals 293 22913 78.2

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

APİ DEN VERİ AL:

library(jsonlite)

library(httr)

res = GET("https://jsonplaceholder.typicode.com/posts/1")

rawToChar(res$content)

data = fromJSON(rawToChar(res$content))

names(data)

data = as.data.frame(data)

View(as.data.frame(data))

```

```{r}

res = GET("https://jsonplaceholder.typicode.com/comments?postId=1")

rawToChar(res$content)

data=fromJSON(rawToChar(res$content))

```

```{r}

res = GET("https://jsonplaceholder.typicode.com/posts")

fromJSON(rawToChar(res$content))

recode yerine ifelse kullanılabilir.

a%>%mutate(

sex = recode(

sex,

kadın = 0, önce eski değer sonra yeni değer

erkek = 1

)

)

isimler[complete.cases(isimler),] eksik olmayan verileri getirir.

isimler[!complete.cases(isimler),] eksik verileri getirir.

a%>%filter(is.na(dep\_time))%>%select(dep\_time,dep\_delay,origin)

cinsiyet%>%mutate(

cinsiyet = recode(

cinsiyet,

male = "erkek",

female = "kadın"

)

)

Ozet tablo çıkarmak.

summarize(a,

ort\_yıl = mean(year),

ort\_kal\_dak = mean(arr\_time,na.rm=TRUE),

min\_gun = min(day)

)

library(odbc)

library(RSQLite)

con <- dbConnect(odbc(),

Driver = "SQL Server",

Server = "localhost\\SQLEXPRESS",

Database = "Northwind",

Trusted\_Connection = "True")

dbListTables(con)

dbGetQuery(con,"select \* from Nakliyeciler")

dbGetQuery(con,"select \* from urunler")

dbGetQuery(con,"select \* from kategoriler")

dbGetQuery(con,"select \* from urunler u inner join kategoriler k on k.KategoriId = u.KategoriId")

dbGetQuery(con,"select u.urunadi , k.kategoriadi from urunler u inner join kategoriler k on k.KategoriId = u.KategoriId")

dbGetQuery(con,"select PersonelID , satis\_tutari = sum(Nakliyeucreti) from satislar group by PersonelID having sum(Nakliyeucreti) >8000 order by satis\_tutari desc ")

dbGetQuery(con,"select \* from urunler where urunadi like 'ch%' and tedarikciID = 2 ")

dbGetQuery(con , "select urunadi , toplam\_tutar = sum(birimfiyati) / (select sum(birimfiyati) from urunler ) from urunler group by urunadi order by toplam\_tutar desc")

dbGetQuery(con , "select \* from musteriler")

dbGetQuery(con , "select \* from satislar")

dbGetQuery(con , "select m.MusteriID , toplam\_satis = sum(s.Nakliyeucreti),

satis\_yuzdesi = sum(s.Nakliyeucreti) / (select sum(nakliyeucreti) from satislar) ,

satis\_adedi = count(s.musteriID) from musteriler m inner join satislar s on s.MusteriID = m.MusteriID group by m.musteriID order by toplam\_satis desc")

dbGetQuery(con,"use makinelerim") #makinelerim veri tabanına bağlandı

dbListTables(con) #makinelerim veri tabanındaki tabloları listeledi.

dbGetQuery(con, "select \* from makinebilgi")

dbSendQuery(con , "delete makinebilgi where parcaadi = 'Python' or parcaadi = 'r studio' ") #ikisini de sildi.

dbSendQuery(con , "insert into makinebilgi values(

'R',76,85,90,2,'2021-06-14'

)")

dbGetQuery(con , "select \* from makinebilgi")

dbSendQuery(con , "update makinebilgi set parcaadi = 'R Studio' where parcaadi = 'R' ")

SQL server’a kullanıcı adı ve şifre ile bağlanma

library(odbc)

con <- dbConnect(odbc(),

Driver = "SQLServer",

Server = "localhost\\SQLEXPRESS",

Database = "fabrika",

UID = "admin",

PWD = rstudioapi::askForPassword('377Harun'),

Port = 1433)

One sample T-test

t.test(a,mu = 20,conf.level = 0.99)

One Sample t-test

data: a

t = -51.59, df = 724, p-value < 2.2e-16

alternative hypothesis: true mean is not equal to 20

99 percent confidence interval:

11.94277 12.71102

sample estimates:

mean of x

12.3269

**T-Test**

**bartlett.test(Yas~Sigara , data = lungcap)**

Bartlett test of homogeneity of variances

data: Yas by Sigara

Bartlett's K-squared = 19.885, df = 1, p-value = 8.223e-06

t.test(Yas~Sigara,data = lungcap ,conf.level = 0.99 , mu = -3, alternative = "less",var.equal = FALSE)

Welch Two Sample t-test

data: Yas by Sigara

t = 0.75473, df = 123.33, p-value = 0.7741

alternative hypothesis: true difference in means is less than -3

99 percent confidence interval:

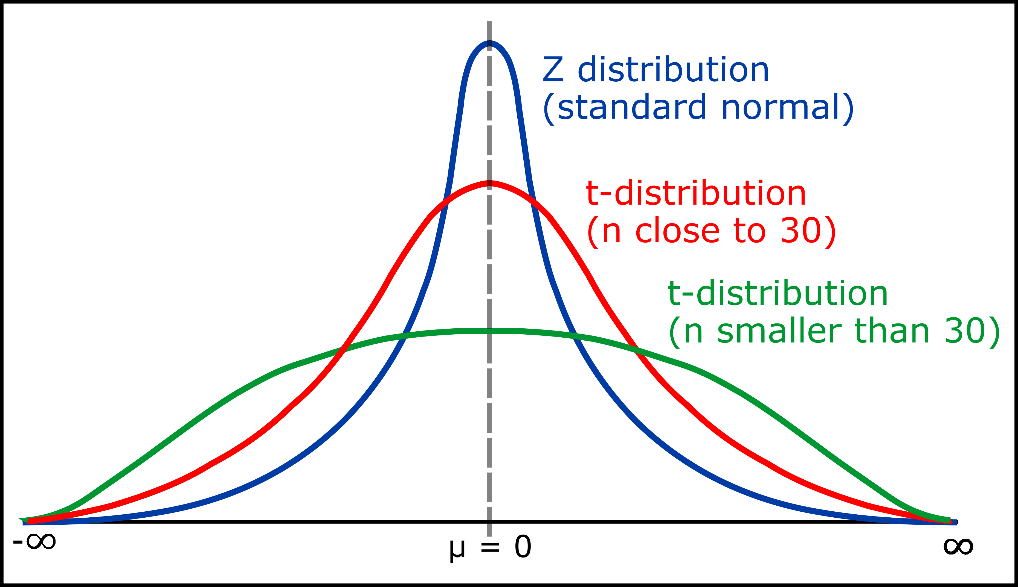
-Inf -1.943403

sample estimates:

mean in group no mean in group yes

12.03549 14.77922

T dağılımının standart sapması daha fazladır genelde örneklem sayısı az iken kullanılır örnek 30 un altındayken kullanılır.



mean(heart$chol)

sonuclar = c(1:50)

for(i in 1:50){

ornek = sample(heart$chol , 50 )

p\_value = t.test(ornek , mu = 250)$p.value

if(p\_value<0.05){

sonuclar[i] = "Red"

}

else if(p\_value>0.05){

sonuclar[i] = "Kabul"

}

}

prop.table(table(sonuclar))

sonuclar

Kabul Red

0.9 0.1

Bütün değişkenleri faktöre çevirmek.

which(substr(names(Train\_yeni),1,2) == "fe")

for( i in names(Train\_yeni[,which(substr(names(Train\_yeni),1,2) == "fe")]) ){

Train\_yeni[,i] = as.factor(Train\_yeni[,i])

}

Min split dallanırken en az kaç gözleme bakayım örneğin min split 20 olsun en son 20 gözlemim yoksa o zaman karar vermem gerek leaf node a ulaşırım.

Max depth kök ile yaprak arasındaki en uzun mesafe

Min bucket karar verirken leaf node da minimum kaç tane kararım olmalı bunu gösteriyor. = round(min.split/3)

Veri seti ne kadar büyük ise model kurma aşaması da o kadar uzun sürüyor.

Örneğin 200 k satır 77 değişkene sahip bir karar ağacı 1 saatten aşağıya oluşmaz.

**APİ’DEN VERİ AL VERİ GÖNDER**

library(httr)

library(jsonlite)

res = GET("https://jsonplaceholder.typicode.com/users")

fromJSON(rawToChar(res$content))

#bu istek ile veriyi gönderdim.

res=POST("http://kasimadalan.pe.hu/kisiler/insert\_kisiler.php",

body = data.frame(kisi\_ad = "harun",

kisi\_tel= "05306305656"),

encode = c("multipart", "form", "json", "raw"))

#Ridge Regresyon

lapply(a,function(x) dim(x))

$test

[1] 99 14

$train

[1] 407 14

#overfitting problemi var. ridge regresyon yap hem aşırı öğrenme hem de

#çoklu baplantı problemini çöz

#Ridge regresyon

library(glmnet)

a = model.matrix(~.,data = a)[,-1]

indeks = createDataPartition(a[,6] , list = FALSE,p = 0.80)

train = a[indeks,]

test = a[-indeks ,]

train\_x = train[,-6]

train\_y = train[,6]

test\_x = test[,-6]

test\_y = test[,6]

modelRidge = glmnet(y=train\_y ,x= train\_x, alpha = 0 , lambda = 0.5)

modelRidge$beta

modelRidge$a0

R2(predict(modelRidge, train\_x),train\_y)

RMSE(predict(modelRidge, train\_x),train\_y)

MAE(predict(modelRidge, train\_x),train\_y)

testHatasi = defaultSummary(

data.frame(

obs = test\_y,

pred = as.vector(predict(modelRidge, test\_x))

)

)

trainHatasi = defaultSummary(

data.frame(

obs = train\_y,

pred = as.vector(predict(modelRidge, train\_x))

)

)

rbind(testHatasi,trainHatasi)

lambdas = 10^seq(-2,3,by = 0.1)

modelRidgeCv = cv.glmnet(train\_x,train\_y , lambda = lambdas , alpha= 0 , nfolds = 10)

modelRidgeCv$lambda.min

R2(predict(modelRidgeCv, train\_x),train\_y)

RMSE(predict(modelRidgeCv, train\_x),train\_y)

MAE(predict(modelRidgeCv, train\_x),train\_y)

testHatasi = defaultSummary(

data.frame(

obs = test\_y,

pred = as.vector(predict(modelRidgeCv, test\_x))

)

)

trainHatasi = defaultSummary(

data.frame(

obs = train\_y,

pred = as.vector(predict(modelRidgeCv, train\_x))

)

)

rbind(testHatasi,trainHatasi)

gercek = train\_y

tahmin = predict(modelRidgeCv , train\_x)

tahmin = as.vector(tahmin)

sonuclar = data.frame("gercek" = gercek , "tahmin" = tahmin)

head(sonuclar)

sonuclar$fark = sonuclar$gercek - sonuclar$tahmin

sonuclar$Mutlak = abs(sonuclar$gercek- sonuclar$tahmin)

sonuclar$Karesi = sonuclar$fark^2

#aynı değerler.

sqrt(mean(sonuclar$Karesi))

RMSE(tahmin,gercek)

#aynı değerler.

MAE(gercek,tahmin)

mean(sonuclar$Mutlak)

pairs(train)

gamma ve c parametreleri hyperplane i nereye yerleştireceğime karar verir.

çok önemlidir. bu parametlere doğru ayarlanmalıdır.

c parametresi hyperplane in her iki sınıfa olan uzaklığını temsil eder.

örnegin c yuksek ise hyper plan her iki sınıfa da cok yakın dar bir alandadır.

c orta ise her iki sınıfada orta derece de uzaklıktadır

c dusuk ise hyperplan her iki sınıfa da çok uzaktır.

#Sadece numeric veya kategorik değişkenleri seçmek

select\_if(df,is.factor)

select\_if(df,is.numeric)

cbind(select\_if(df,is.factor) , select\_if(df,is.character))

**#Eksik veri içeren ve içermeyen değişkenler**

miss = as.data.frame(apply(df,MARGIN = 2 , function(x) sum(is.na(x))))

colnames(miss) = c("e\_sayi")

miss

rownames(miss)

miss$Degisken = rownames(miss)

rownames(miss) = c()

miss

eksik\_degis= miss[which(miss$e\_sayi>0),"Degisken"]

eksik\_degis = as.vector(eksik\_degis)

#eksik veri iceren degiskenler

df%>%select(eksik\_degis)

#eksik veri içermeyen değişkenler

df%>%select(!eksik\_degis)

#FALIM kelimesini içeren sastırları getir.

filter(df, grepl("FALIM",df$ITEMNAME))

strings <- c('abcd', 'dabc', 'abcabc',"harunabcharun","harunabc")

pattern <- '^abc'

icerir='abc'

sagdan='abc$'

print (grep(pattern, strings))

strings[grep(pattern, strings)]

strings[grep(icerir, strings)]

strings[grep(sagdan, strings)]

SparklyR big data

library(sparklyr)

library(dplyr)

library(caret)

library(tidyverse)

library(nycflights13)

library(DBI)

sc=spark\_connect(master = "local")

src\_tbls(sc) #burada tablo olup olmamasını inceliyoruz.

copy\_to(sc , df ,"df\_spark") #spark'a aktaracagım veri setidir.

f\_tbl=copy\_to(sc , df ,"df\_spark",overwrite = TRUE)

str(f\_tbl)

f\_tbl$ITEMNAME #null dondu cunku

#bu fonksiyonlar çalışmıyor dplyr backendi çalışır ama apply calışmadı

head(f\_tbl)

f\_tbl[1:10,] #bu çalışmadı

#dplyr sparklyR a full backend destek sağlar.

f\_tbl%>%filter(grepl("FALIM",ITEMNAME))

colnames(f\_tbl)

f\_tbl%>%select(c("ITEMNAME","PRICE")) #matris olarak seçtiğimde gelmedi.

dbGetQuery(sc, "show databases")

dbGetQuery(sc, "show tables")

dbGetQuery(sc,"select \* from df\_spark where ITEMNAME like '%FALIM%' and BRANCH like '%İstanbul%' ")

dbGetQuery(sc,"select ITEMNAME, count(ITEMNAME) as Sayi from df\_spark group by ITEMNAME

order by Sayi desc")

model = ml\_linear\_regression(f\_tbl,TUTAR~.)

f\_tbl%>%select(TUTAR)

summary(model)

predicted = ml\_predict(model , f\_tbl)

predicted%>%select(prediction)

view(predicted)

class(predicted)

tahmin = predicted%>%select(prediction)

gercek = predicted%>%select(TUTAR)

model=ml\_linear\_regression(f\_tbl,TUTAR~.)

model\_randomForest = ml\_random\_forest(f\_tbl , ITEMNAME~.)

summary(model)

R2 ve Düzeltilmiş R2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Düzeltilmiş R2 genelde daha küçüktür.

Modele yeni değişkenler eklense bile etkilenmeyecek olan düzeltilmiş R2 değeridir. Başka bir açıdan eğer R2 ve Düzeltilmiş R2 kare değerleri arasında çok fark var ise demek ki gereksiz fazla anlamsız değişken kullanıyoruz. Düzeltilmiş R2 anlamlılığı düşük olan değişkenleri hesaba katmaz.

Modelde ilgili ilgisiz anlamlı anlamsız ne kadar değişken var ise R2 değeri artacaktır.

F istatistiği modelin anlamlılığına ilişkin istatistiki değer.

Prob(F-statistic) modelin anlamlılığına ilişkin p-value değeridir.

Standart hata standart sapma/ kök n

Randomforest ve Karar ağaçları:

Randomforest’da birden fazla karar ağacı kullanıp bir orman oluşturuyorum.

Her bir karar ağacımdaki bağımsız değişkenler rastgele oluşturuluyor.

Her bir ağaçtan bir tahmin alıyorum.

En fazla hangi sonuç geldiyse onu kabul ediyorum.

Bu oluşturduğum karar ağaçlarında rastgele featurelarım var.

Her bir ağacın sıraları farklı oluyor.

Karar ağacındaki regresyon problemlerinde bağımlı değişkeni kendisi sınıflara ayırıp öyle tahminleme yapıyor.

Root node ağaca en çok bilgi kazanımı olan değişkendir.

Desicion node karar node’u yani yaş<25 cinsiyet=E gibi

Branch ağaçtaki dal true false diyen dal.

Parent node child node’un üstünde olan karar node’u

Min split = her bir karar node’unda min kaç tane gözlemim olsun.

Örneğin min split 20 olarak belirledim bir karar node’una geldiğimde o karar node’unda 20 tane gözlem yok ise dallanmayı durdururum. Daha aşağı inmem.

Min bucket = karar verebilmem için leaf node’umda min bucket kadar gözlemim olması gerek örneğin min split 20 tamam 20 tane aldım ve aşağı indim ama o leaf node umda min bucket kadar gözlem yok ise karar veremem.

Max depth =root ile leaf node’um arasının ne kadar derin olacağına karar verir. Bu da en fazla gözlem sayım -1 kadar olur. Ve bu max depth min split ile orantılıdır. Mesela bir karar node’unda 20 tane gözlem varsa aşağı in diyorum bu da benim max depth’imi etkiliyor.

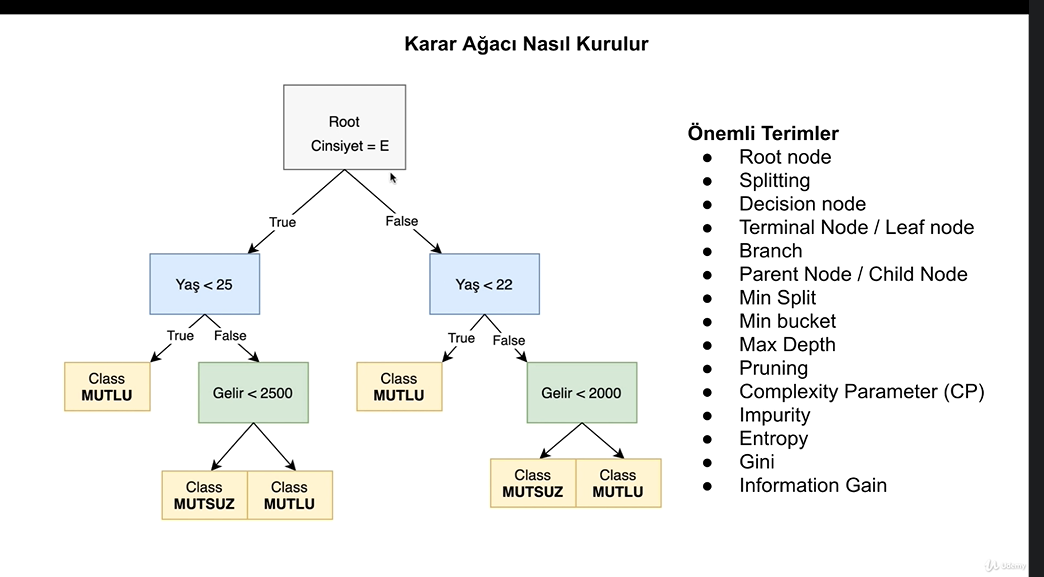
Pruning = budama anlamına gelir. Karar ağacı çok büyük ise complexity artar bu da benim yanlış karar vermeme sebep olur. Bu yüzden ağacım sade olmalı bunun için pruning yani budama işlemleri yapıyorum.

Complexity Parameter = tuning parametremizdir. Anlamı ben bu karar ağacına her yeni karar eklediğimde tahmin gücüm 0.01 artmalı.

Entropy ve Gini bilgi kazanımı için kullanılır.

İmpurity = karşıtlık parametresi. Bu karşıtlığı ölçmek için entropy ve gini indeksleri kullanılır. Örneğin karar node’um yaş<25 yaş<25 olasılığı 0.5 yaş>25 olasılığı 0.5 ama yaş<25 0.8 yaş>25 0.2 ben burada bu impurity nin her zaman düşük olmasını beklerim bu impurity bildiğimiz entropy diyebiliriz.

Karar ağacı Hiper-Parametreleri



#Model tuning

#eger ben train fonksiyonumda method argümanına bunları verirsem

#hangi hiper parametreleri tune eder bana onu gösteriyor.

modelLookup("lm")

modelLookup("svmRadial")

modelLookup("svmLinear")

modelLookup("knn")

modelLookup("rpart") #cp parametresini tune eder.

modelLookup("rpart2") #maxdepth'i tune eder.

library(e1071) #•buradaki tune.rpart fonksiyonunu kullanıcam

#bu tune.rpart fonksiyonu ile min split ve min bucket parametrelerini de tune edebilirim.

trControl = trainControl(method = "cv",number = 5,search = "random")

trControl2 = trainControl(method = "cv",number = 5,search = "grid")

modelCP = train(sales~. ,

data=Advertising,

method="rpart",

tuneLength=20,

trControl=trControl

)

modelMaxDep = train(sales~. ,

data=Advertising,

method="rpart2",

tuneLength=20,

trControl=trControl

)

#random search yaparsam tuneLength grid search yaparsam tuneGrid kullan

#Grid search büyük veri setlerinde her parametreyi tek tek denediği için

#random search'e istinaden daha maliyetli bu yüzden gercek hayatta genelde

#random search kullanılır.

modelCP = train(sales~. ,

data=Advertising,

method="rpart",

tuneGrid=expand.grid(cp=seq(0,1,0.1)), #0'dan 1'e git 0.1 art

trControl=trControl2

)

modelMaxDep = train(sales~. ,

data=Advertising,

method="rpart2",

tuneGrid=expand.grid(maxdepth =3:20),

trControl=trControl2

)