
بسم الله الرحمن الرحيم

پروژه درس داده کاوی

یادگیری عمیق

استاد محترم درس

جناب آقای دکتر پاینده

دانشجو

آیدا اعلابیکی

در ابتدا بر خود واجب می‌دانم از زحمات استاد ارجمندم جناب آقای دکتر پاینده، کمال تشکر را به جا آورم.

داده‌های هزینه و درآمد سال 1389 خانوارهای کل کشور از سایت مرکز آمار ایران اخذ شده است.

هدف از این پروژه، پیدا کردن بهترین مدل پیشگویانه برای رده بندی خانوارهای شهری براساس هزینه و درآمد به دو رده کم درآمد و پر درآمد است.

بررسی داده‌ها

به طور کلی، اطلاعات هزینه و درامد خانوار به ۴ بخش، تقسیم شده‌است:

- خصوصیات اجتماعی اعضای خانوار
- مشخصات محل سکونت و تسهیلات و لوازم عمده زندگی
- هزینه های خوراکی و غیرخوراکی خانوار
- درآمدهای خانوار

داده ها شامل ۶۸متغیر و ۲۱۸۸نمونه است. در این گزارش پس از اماده‌سازی داده‌ها آن‌ها را به سه بخش آزمون، تست و اعتبارسنجی تقسیم می‌کنیم در برخی از مواقع به موجب نوع مدل از تست و اعتبارسنجی را بایکدیگر مخلوط می‌کنیم.

ردیف	تعریف متغیر	نام متغیر
خصوصیات اجتماعی اعضای خانوار		
۱	آدرس خانوار	Address
۲	کد استان	C.Ostan
۳	جنسیت سرپرست خانوار	Gender
۴	سن سرپرست خانوار	Age
۵	میزان سواد سرپرست خانوار	Savad
۶	سرپرست خانوار تحصیل می‌کند یا خیر؟	InEdu
۷	مدرک تحصیلی سرپرست خانوار	Edu
۸	وضعیت فعالیت سرپرست خانوار	Faaliat
۹	تعداد اعضای شاغل خانوار	T.shaghel
۱۰	تعداد اعضای خانوار	Tedad.a
۱۱	نحوه تصرف منزل مسکونی	T.M.S
۱۲	تعداد اتاق در اختیار	T.O
۱۳	سطح زیر بنای محل سکونت	S.Z
۱۴	نوع اسکلت بنای محل سکونت	N.S
۱۵	مصالح عمده بنای محل سکونت	Masleh
۱۶	اتومبیل شخصی	oto
۱۷	موتورسیکلت	mo
۱۸	دوچرخه	do
۱۹	رادیو	radio
۲۰	ضبط	zabt
۲۱	تلویزیون سیاه و سفید	TV.S

ردیف	تعریف متغیر	نام متغیر
۲۲	تلویزیون رنگی	TV.r
۲۳	انواع ویدئو، VCD و DVD	DVD
۲۴	انواع یارانه و تبلت	Pc
۲۵	تلفن همراه	mobile
۲۶	فریزر	freeizer
۲۷	یخچال	yakhchal
۲۸	یخچال فریزر	yakhchal.f
۲۹	اجاق گاز	gaz
۳۰	جارو برقی	jaro.b
۳۱	ماشین لباسشویی	m.lebas
۳۲	چرخ خیاطی	charkh.kh
۳۳	پنکه	panke
۳۴	کولر آبی متحرک	cooler.a
۳۵	کولر گازی متحرک	cooler.g
۳۶	ماشین ظرفشویی	m.zarf
۳۷	مایکروویو و انواع فرهای هالوژن دار	microfer
۳۸	آب لولهکشی	ab.l
۳۹	برق	bargh
۴۰	گاز لولهکشی	gaz.l
۴۱	تلفن ثابت	tel
۴۲	دسترسی به اینترنت	internet
۴۳	حمام	hamam

ردیف	تعریف متغیر	نام متغیر
۴۴	آشپزخانه	ashpazkhane
۴۵	کولر آبی ثابت	cooler.a.s
۴۶	برودت مرکزی	broodat.m
۴۷	حرارت مرکزی	hararat.m
۴۸	پکیج	package
۴۹	کولر گازی ثابت	cooler.g.s
۵۰	شبکه عمومی فاضلاب	fazelab
نوع سوخت عمده مصرفی خانوار		
۵۱	نوع سوخت برای پخت و پز	sookht.p
۵۲	نوع سوخت برای ایجاد گرما	sookht.g
۵۳	نوع سوخت برای تهیه آب گرم	sookht.ab
هزینه‌های خانوار		
۵۴	هزینه‌های بهداشتی خانوار در ماه گذشته	H_behdasht
۵۵	هزینه ارتباطات خانوار در ماه گذشته	H_Ertebatat
۵۶	هزینه‌های غذای آماده خانوار در ماه گذشته	H_Ghazayeamade
۵۷	هزینه‌های حمل و نقل خانوار در ماه گذشته	H_Hamlonaghl
۵۸	هزینه‌های متفرقه خانوار در ماه گذشته	H_kalavakhadamat
۵۹	هزینه‌های دخانیات خانوار در ماه گذشته	H_Khorakivadokhani
۶۰	هزینه‌های مبلمان خانوار در ماه گذشته	H_mobleman
۶۱	هزینه‌های مسکن خانوار در ماه گذشته	H_Maskan
۶۲	هزینه‌های نوشیدنی خانوار در ماه گذشته	H_Noshidani
۶۳	هزینه‌های تفریحات خانوار در ماه گذشته	H_Tafrihat

جدول 1- جدول داده هزینه خانوار

فراخوانی داده‌ها

ابتدا از کد زیر استفاده می‌کنیم ،داده‌ها و بسته‌های موردنیاز در طول تحلیل را فراخوانی می‌کنیم.

همچنین ستون آدرس را نیز حذف می‌کنیم.

```
library(readr)
library(tidyverse)
library(plyr); library(dplyr)
library(imputeMissings)
library(ggplot2)
library(hrbrthemes)
library(tidyr)
library(viridis)
library(ROSE)
Data <- as.data.frame(read_csv ("C /Users/acer/Desktop/Daramad khanevar "))
Data=Data[,-1]
```

پاک‌سازی داده‌ها

ابتدا با استفاده از کد زیر ستون‌هایی که بیش از ۴۰ درصد داده‌هایشان گمشده است کنار می‌گذاریم زیرا جانهی آن‌ها باعث خراب شدن داده‌ها می‌شود و داده‌ها بیش از حد تغییر می‌کنند.

```
Columns<-c()
for(i in 1:(ncol(Data)-4)){
  if(sum(is.na(Data[i]))/nrow(Data) > 0.4)
    Columns<-c(Columns,i)
}
Data=Data[-Columns]
summary(Data)
```

ستون‌های زیر در طی فرآیند بالا حذف شده‌اند.

```
 [1] "oto"         "motor"       "do"          "radio"       "zabt"
 [6] "TV.S"        "DVD"         "Pc"          "freeizer"    "yakhchal"
[11] "charkh.kh"   "panke"       "cooler.a"    "cooler.g"    "m.zarf"
```

```
[16] "microfer"      "ab.l"          "internet"      "hamam"         "broodat.m"
[21] "hararat.m"     "package"       "cooler.g.s"    "H_Noshidani"   "H_Pushak"
[26] "H_Tafrihat"    "H_Ghazayeamade"
```

می‌دانیم که وجود واریانس در متغیرها نشان‌دهنده حضور اطلاعات در آن‌ها می‌باشد و متغیری که واریانس صفر دارد عملاً بی‌ارزش است پس متغیرهایی با واریانس صفر را کنار می گذاریم.

```
Columns<-c()
for (i in 1:ncol(Data)) {
  if(length(unique(na.omit(Data[,i])))==1)
    Columns<-c(Columns,i)
}
Data=Data[-Columns]
summary(Data)
```

متغیرهای زیر نیز از جریان تحلیل حذف می‌شوند.

```
 [1] "TV.r"       "mobile"     "yakhchal.f" "gaz"        "jaro.b"     "m.lebas"
 [7] "bargh"      "gaz.l"      "tel"        "ashpazkhane" "cooler.a.s" "fazelab"
```

وضعیت گمشدگی باقی متغیرها را بررسی می کنیم.

```
     InEdu            Edu            Faaliat        T.shaghel        T.M.S
 NA's   :502     NA's   :502                    NA's   :594
      T.O             S.Z             N.S           Masleh         sookht.p
                                                  NA's   :598
   sookht.g       sookht.ab     H_Khorakivadokhani    H_Maskan
                                        NA's   :4
  H_mobleman        H_behdasht        H_Hamlonaghl      H_Ertebatat
    NA's   :178     NA's   :635       NA's   :105       NA's   :40
 H_kalavakhadamat      D_Mozd          D_Azad         D_Motefaraghe
    NA's   :120     NA's   :1205      NA's   :1494      NA's   :510
    D_Yarane
    NA's   :91
```

تعداد مقادیر گمشده هر متغیر زیر نام متغییر نوشته شده است و عدم حضور هیچ عبارتی در زیر نام متغیر نشان‌دهنده عدم وجود داده‌ی گمشده است.

جانهی متغیرها

در ادامه متوجه میشویم که تمام افرادی که بیسواد هستند سوالات "در حال تحصیل" و "مدرک تحصیل"را بی پاسخ رها کرده‌اند. لذا مقدار ویژگی‌ها فوق به ترتیب ۲ و ۰ خواهد بود.

```
#imputation of Edu and InEdu
temp = is.na(Data$InEdu)
unique(Data[temp ,]$Savad)
Data[temp ,]$Edu = 0
Data[temp ,]$InEdu = 2
rm(temp)
```

تبدیل متغیرهای Categorical به Factor

در ادامه متغیرهای رسته‌ای را تبدیل می‌کنیم.

```
#Convert Categorical Value to Factors
factors=c( "C.Ostan" ,"Tedad.a","Gender","Savad","InEdu","Edu","Faaliat",
        "T.shaghel","T.M.S","T.O","N.S","Masleh" , "sookht.p","sookht.g","sookht.ab")
for (i in factors) {
  Data[,i]=factor(Data[,i])
}
rm(factors,i)
```

جانهی متغیرهای رسته‌ای و پیوسته

به لحاظ منطقی تعداد اعضای خانوار با تعداد اعضای شاغل خانوار رابطه دارد در نتیجه مقادیر گمشده تعداد اعضای شاغل خانوار را باگروه‌بندی داده با توجه به تعداد اعضای خانوار و استفاده از مد متغیر تعداد اعضای شاغل خانوار در گروه مربوطه به‌دست می‌آوریم.

```
#a Function for finding modes
Modes <- function(x) {
  ux <- unique(x)
  tab <- tabulate(match(x, ux))
  ux[tab == max(tab)]
```

```
}
Modes_of_T.shaghel=Data %>%
  group_by(Tedad.a) %>%
  summarise(Mode = Modes(na.omit(T.shaghel)))
#With Respect to Modes_of_T.shaghel we Choose 1 for Null Value of T.shaghel
Data$T.shaghel[is.na(Data$T.shaghel)]=1
```

حال متغیر مصالح را نیز با توجه به متغیر استان جانهی می‌کنیم.

```
#Imputing Masaleh
Modes_of_Masleh=Data %>%
  group_by(C.Ostan) %>%
  summarise(Mode = Modes(na.omit(Masleh)))
Data$Masleh[is.na(Data$Masleh)]=1


rm(Modes,Modes_of_T.shaghel,Modes_of_Masleh)
```

مقدار یارانه سالانه برای هر فرد ۵۱۰۰۰۰۰ می‌باشد و با توجه به تعداد اعضای خانوار این مقدار را جانهی می‌کنیم.

```
#Impute Yaraneh
Data$D_Yarane[is.na(Data$D_Yarane)]=as.numeric(Data$Tedad.a[is.na(Data$D_Yarane)])*5100000
```

در پایان نیز تمام متغیرهای پیوسته که مقادیر گمشده دارند بااستفاده جنگل تصادفی رگرسیونی و استفاده از تمام متغیرهای باقی‌مانده جانهی می‌کنیم سپس مقادیر منفی حاصل از جنگل رگرسیونی را با میانگین متغیر مربوطه جانهی می‌کنیم

```
#Impute Coumtinios Values
Data=impute(Data, object = NULL, method = "randomForest", flag = FALSE)
#Correct negative value
Data$D_Azad[Data$D_Azad<0]=mean(Data$D_Azad[Data$D_Azad>0])
```

ساخت متغیر درآمد و رسته‌ای کردن آن با توجه به چندک‌ها

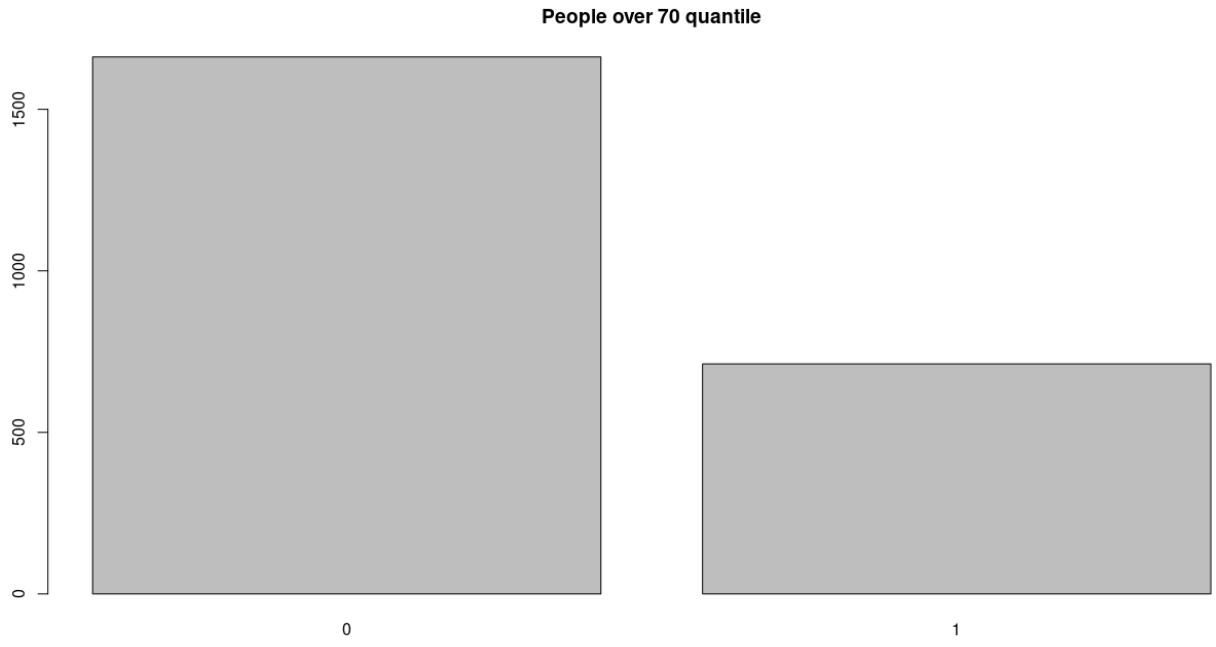
حال متغیر درآمد را با توجه به کد زیر می‌سازیم.

```
#Building Income
Data=Data %>% mutate(Daramad = D_Mozd+D_Mozd+D_Motefaraghe+D_Yarane)
#Build Categorical variables
```

```
Quantile=quantile(Data$Daramad,0.7)
Data <- Data %>%mutate(upquantile = case_when(
  .$Daramad %>% between(Quantile,max(Data$Daramad)+1) ~ 1,
  .$Daramad %>% between(0,Quantile) ~ 0))

Quantile=quantile(Data$Daramad,0.3)
Data <- Data %>%mutate(downquantile = case_when(
  .$Daramad %>% between(Quantile,max(Data$Daramad)+1) ~ 0,
  .$Daramad %>% between(0,Quantile) ~ 1))
```

نگاهی به یکی از دو متغیر پاسخ رسته ای تولیدشده می‌اندازیم.

```
#Barplots
barplot(table(Data$upquantile),main = "People over 70 quantile")
Data$upquantile=factor(Data$upquantile)
Data$downquantile=factor(Data$downquantile)
rm(Quantile)
```



مشاهده می‌کنیم که با یک کلاس‌بندی نامتوازن مواجهیم از بیش‌نمونه برداری برای متوازن کردن کلاس‌ها استفاده می‌کنیم.

```
rm(smp_siz)
#Solve inbalanced Problem
train=ovun.sample(upquantile ~ ., data = train, method = "over",N = 2200)$data
table(train$upquantile)# data balanced
```

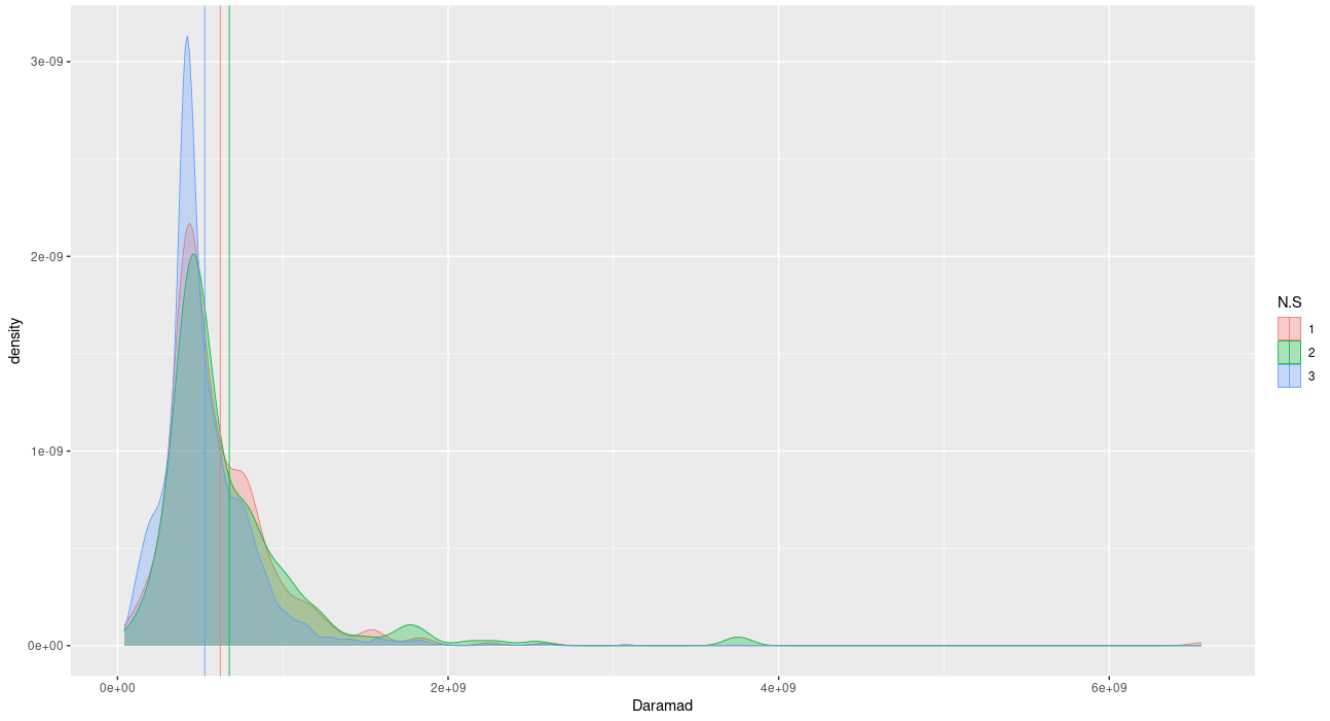
خروجی کد نشان‌دهنده‌ی این است که کلاس‌ها متوازن شده‌اند.

برخی از نمودارهای تولید شده از داده‌ها

توزیع درآمد تحت نوع اسکلت بنای محل سکونت

با استفاده از دستور زیر توزیع درآمد را تحت نوع اسکلت بنای محل سکونت رسم می‌کنیم.

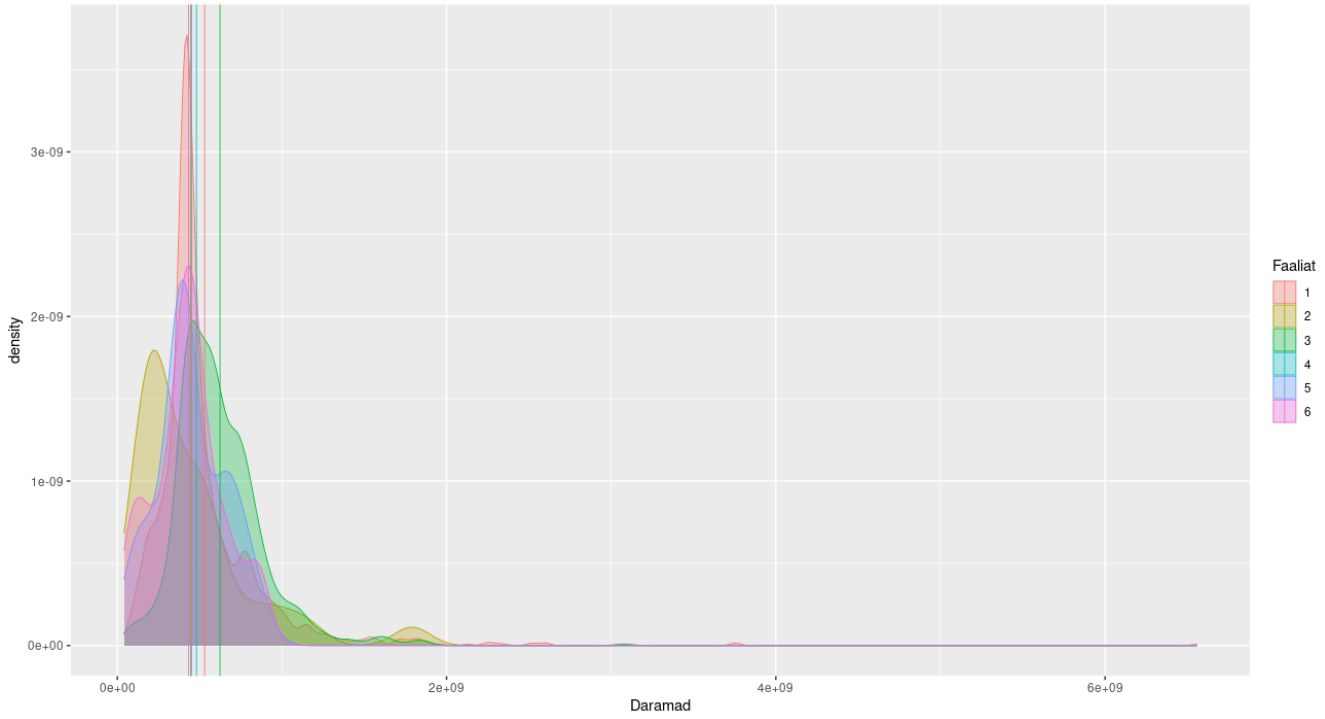
```
mu <- ddply(Data , "N.S", summarise , grp.mean=mean(Daramad))
ggplot(Data , aes(x=Daramad , color=N.S, fill=N.S)) +
  geom_density(alpha=0.3,size=.3)+
  geom_vline(data = mu, aes(xintercept = grp.mean , color = N.S), size=.3)
```



مشاهده می‌کنیم که توزیع تحت مقادیر مختلف متغیر نوع اسکلت بنای محل سکونت تقریباً یکسان است.

توزیع درآمد تحت وضعیت فعالیت سرپرست خانوار

```
mu <- ddply(Data , "Faaliat", summarise , grp.mean=mean(Daramad))
ggplot(Data , aes(x=Daramad , color=Faaliat, fill=Faaliat)) +
  geom_density(alpha=0.3,size=.3)+
  geom_vline(data = mu, aes(xintercept = grp.mean , color = Faaliat), size=.3)
```

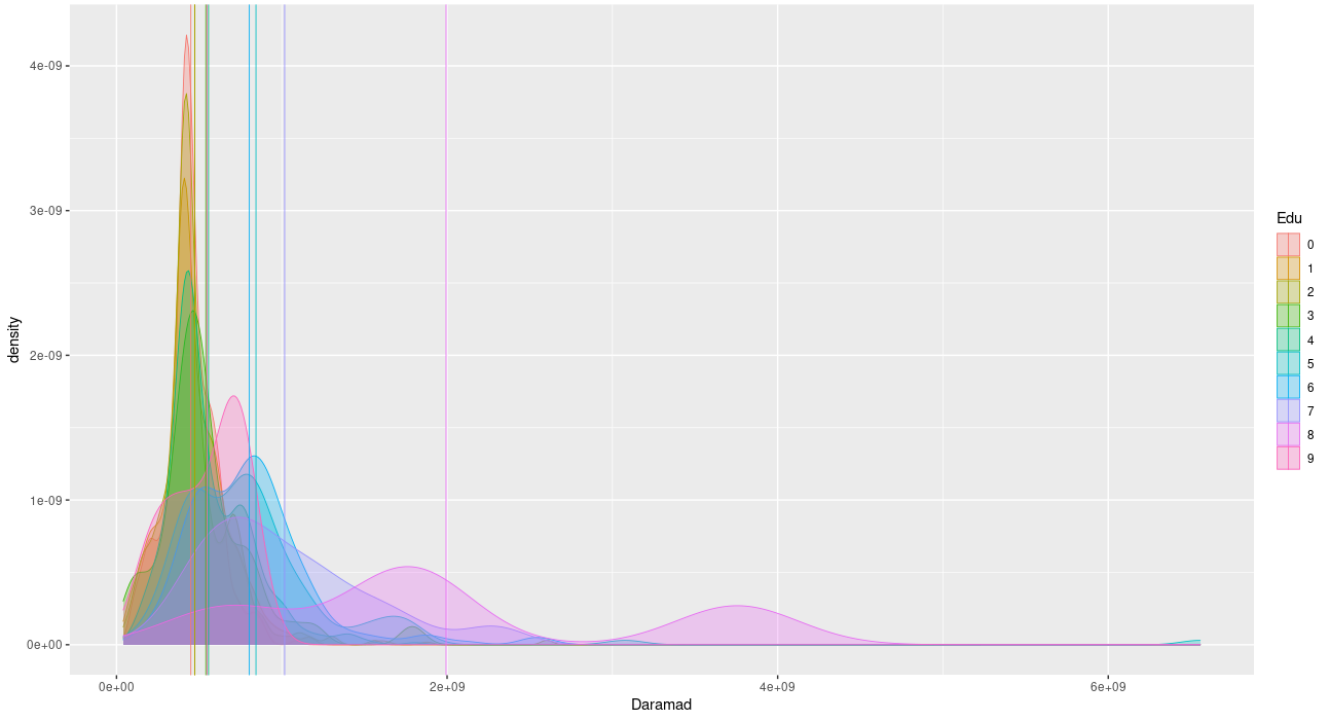


حال متغیرهای مربوط به درآمد را به طور کلی از داده‌ها کنار می‌گذاریم و تحلیل را ادامه می‌دهیم.

مشاهده می‌کنیم که تحت وضعیت فعالیت سرپرست خانوار توزیع درآمد متفاوت است و افرادی که در دسته ۲ قرار می‌گیرند، به‌طور میانگین درآمد کمتری دارند.

توزیع درآمد تحت مدرک تحصیلی سرپرست خانوار

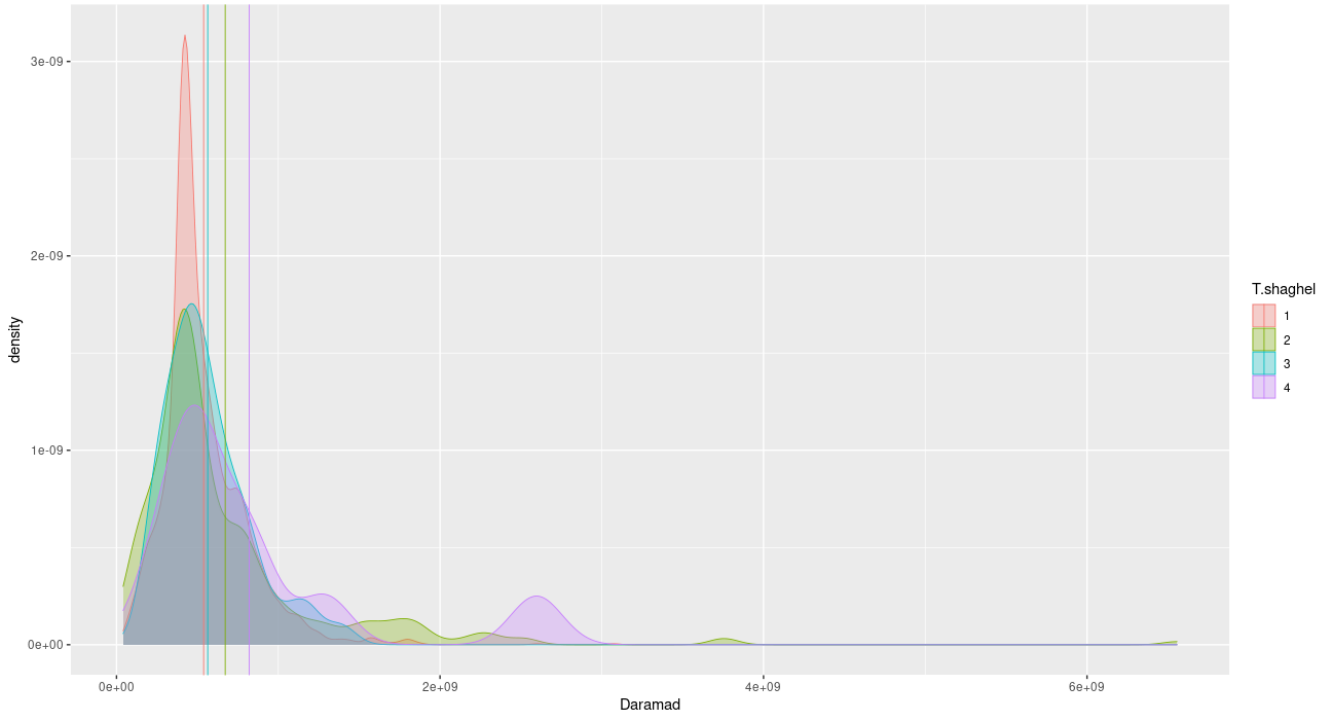
```
mu <- ddply(Data , "Edu", summarise , grp.mean=mean(Daramad))
ggplot(Data , aes(x=Daramad , color=Edu, fill=Edu)) +
  geom_density(alpha=0.3,size=.3)+
  geom_vline(data = mu, aes(xintercept = grp.mean , color = Edu), size=.3)
```



مشاهده می‌کنیم که توزیع درآمد تحت گروه ۸ بسیار متفاوت از دیگر گروه‌ها است و اعضای این گروه میانگین درآمد بالاتری نسبت به گروه‌های دیگر دارند و همچنین این توزیع دومدی است.

توزیع درآمد تحت تعداد شاغلین

```
mu <- ddply(Data , "T.shaghel", summarise , grp.mean=mean(Daramad))
ggplot(Data , aes(x=Daramad , color=T.shaghel, fill=T.shaghel)) +
  geom_density(alpha=0.3,size=.3)+
  geom_vline(data = mu, aes(xintercept = grp.mean , color = T.shaghel), size=.3)
```



مشاهده می‌کنیم که تحت متغیر تعداد شاغلان توزیع درآمد تغییر زیادی ندارد.

حال به انتخاب ویژگی‌های مناسب می‌پردازیم.

مدل‌سازی

تقسیم داده برای مدل‌سازی و بیش‌نمونه‌برداری (دهک ۷۰)

با استفاده از دستور زیر داده را تقسیم می‌کنیم و کلاس‌ها را متوازن می‌کنیم.

```
#Train , Test, Validation

smp_siz = floor(0.7*nrow(Data))

train_ind = sample(seq_len(nrow(Data)),size = smp_siz)  # Randomly identifies therows equal to sample size ( defined in previous instruction) from  all the rows of Smarket dataset and stores the row number in train_ind

train =Data[train_ind,] #creates the training dataset with row numbers stored in train_ind

remaind=Data[-train_ind,]

#test validation

smp_siz = floor(0.5*nrow(remaind))
```

test_ind = sample(seq_len(nrow(remaind)),size = smp_siz) # Randomly identifies therows equal to ،sample size (defined in previous instruction) from all the rows of Smarket dataset and stores the row ،number in train_ind

test =remaind[test_ind,] #creates the training dataset with row numbers stored in train_ind

valid=remaind[-test_ind,]

Feature Selection (دهک ۷۰)

برای انتخاب متغیرهای مناسب از قابلیت نمره‌دهی دسته‌بند جنگل تصادفی استفاده می‌کنیم.

ابتدا مسیله کلاس‌بندی را برای افراد بالای چندک ۷۰ انجام می‌دهیم.

برای این‌کار ابتدا از دسته‌بند جنگل تصادفی استفاده می‌کنیم و متغیرهای مناسب دسته‌بندی را استخراج می‌کنیم.

کد به شرح زیر است.

library(randomForest)

fit_rf = randomForest(train[1:24],train$upquantile, data=train)

#Accuracy

library(caret)

confusionMatrix(predict(fit_rf,remaind[1:24]),remaind$upquantile)

Create an importance based on mean decreasing gini

importance(fit_rf)[order(importance(fit_rf),decreasing = TRUE),]

barplot(importance(fit_rf)[order(importance(fit_rf),decreasing = TRUE),])

Names=names(importance(fit_rf)[importance(fit_rf)>20,])

خروجی به شرح زیر است.

Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	448	86
1	42	137

Accuracy : 0.8205

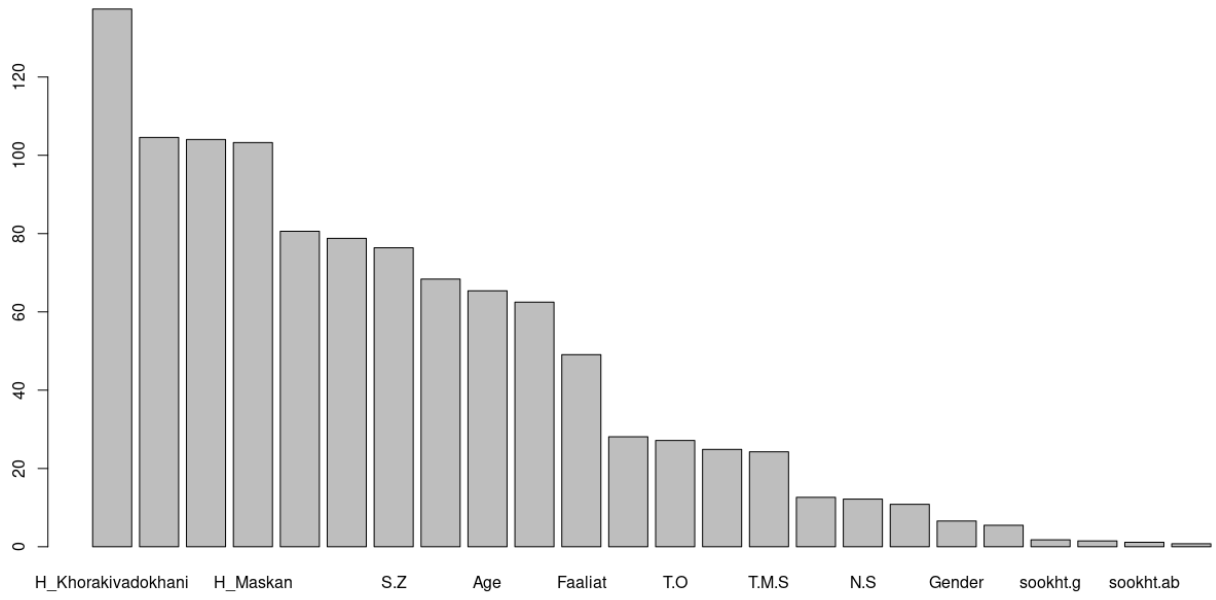
توجه کنید که در اینجا از ترکیب داده‌ی تست و اعتبارسنجی برای سنجش دقت عمل‌کرد مدل استفاده شده است و دقت قابل قبول ٪۸۲ حاصل شده است.

ویژگی‌های انتخاب شده توسط این مدل به شرح زیر است.

توجه کنید ویژگی‌هایی جداشده‌اند که میزان ضریب Gini بالای ۲۰ توسط دسته‌بند به آن‌ها داده‌شده‌است.

```
 [1] "C.Ostan"          "Tedad.a"          "Age"              "Edu"
 [5] "Faaliat"          "T.M.S"            "T.O"              "S.Z"
 [9] "H_Khorakivadokhani" "H_Maskan"       "H_mobleman"       "H_behdasht"
[13] "H_Hamlonaghl"     "H_Ertebatat"      "H_kalavakhadamat"
```

نمودار زیر میزان ضریب Gini را برای تمام متغیرها نشان می‌دهد.



حال مجموعه داده را باتوجه به ویژگی‌های حاصل شده حرص می‌کنیم.

```
train1=train[,c(Names,"upquantile")]
test1=test[,c(Names,"upquantile")]
valid1=valid[,c(Names,"upquantile")]
remaind1=remaind[,c(Names,"upquantile")]
```

مدل لجستیک (دهک ۷۰)

ابتدا مدل لجستیک را پیاده می‌کنیم.

از آنجا که پارامتری برای تنظیم شدن وجود ندارن مجموعه داده تست و اعتبارسنجی را مخلوط می‌کنیم.

```
model <- glm(upquantile~.,family=binomial(link='logit'),data=train1)
confusionMatrix(factor(ifelse(predict(model,remaind1[-16])>0,1,0)),remaind1$upquantile)
```

خروجی برای داده تست به شرح زیر است.

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 407  67
         1  71 168

               Accuracy : 0.8065
                 95% CI : (0.7755, 0.8348)
    No Information Rate : 0.6704
    P-Value [Acc > NIR] : 4.268e-16

                  Kappa : 0.5639

 Mcnemar's Test P-Value : 0.7984

            Sensitivity : 0.8515
            Specificity : 0.7149
         Pos Pred Value : 0.8586
         Neg Pred Value : 0.7029
             Prevalence : 0.6704
         Detection Rate : 0.5708
   Detection Prevalence : 0.6648
```

Balanced Accuracy : 0.7832

'Positive' Class : 0

مترهای موردنظر برای داده آموزش به شرح زیر است.

```
confusionMatrix(factor(ifelse(predict(model,train1[-16])>0,1,0)),train1$upquantile)
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 998 222
         1 186 794

               Accuracy : 0.8145
                 95% CI : (0.7977, 0.8306)
    No Information Rate : 0.5382
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.626

 Mcnemar's Test P-Value : 0.08314

            Sensitivity : 0.8429
            Specificity : 0.7815
         Pos Pred Value : 0.8180
         Neg Pred Value : 0.8102
             Prevalence : 0.5382
         Detection Rate : 0.4536
   Detection Prevalence : 0.5545
      Balanced Accuracy : 0.8122
```

'Positive' Class : 0

میزان دقت مدل ۸۰ درصد است و فاصله اطمینان نیز برای آن فراهم شده‌است.

مدل درخت تصمیم (دهک ۷۰)

مدل درخت تصمیم برخلاف مدل قبلی نیاز به انتخاب Hyper Parameters دارد که به شرح زیر هستند.

-minsplit: Set the minimum number of observations in the node before the algorithm perform a split

-minbucket: Set the minimum number of observations in the final note i.e. the leaf

-maxdepth: Set the maximum depth of any node of the final tree. The root node is treated a depth 0

حال با استفاده از مجموعه داده Validation مقادیر Hyper Parameter ها را تعیین می‌کنیم.

کد زیر این وظیفه را انجام می‌دهد.

```
ibrary(rpart)
library(rpart.plot)


accuracy_tune <- function(fit,data_test) {
  predict_unseen <- predict(fit, data_test, type = 'class')
  table_mat <- table(data_test$upquantile, predict_unseen)
  accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
  accuracy_Test
}
Minsplit=c(4,8,25,50,15)
Minbucket=c((round(Minsplit)+1)/3)
Maxdepth=c(3,25,5,7,9)
for (i in 1:length(Maxdepth)) {
  control <- rpart.control(minsplit = Minsplit[i],
                           minbucket = Minbucket[i],
                           maxdepth = Maxdepth[i])
  tune_fit <- rpart(upquantile~., data = train1, method = 'class', control = control)
```

```
  print(accuracy_tune(tune_fit,valid1))
  }
```

خروجی به شرح زیر است.

```
[1] 0.82
[1] 0.809
[1] 0.806
[1] 0.801
[1] 0.809
```

به نظر می‌رسد مقادیر maxdepth = 3، minbucket = round(5 / 3)، minsplit = 4 مناسب باشد، حال دقت مدل برروی داده آزمون را مشاهده می‌کنیم.

```
#best Choineon test set
control <- rpart.control(minsplit = 4,
                         minbucket = round(5 / 3),
                         maxdepth = 3,)
tune_fit <- rpart(upquantile~., data = train1, method = 'class', control = control)
print(accuracy_tune(tune_fit,test1))
```

خروجی به شرح زیر است.

```
Accuracy =  0.81
```

ریز جزییات مترها به شرح زیر است.

برای داده تست

```
confusionMatrix(predict(tune_fit, test1[-16], type = 'class'),test1$upquantile)
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 215  51
         1  27  63

               Accuracy : 0.812
```

95% CI : (0.7343, 0.8228)

No Information Rate : 0.6798

P-Value [Acc > NIR] : 1.574e-05

Kappa : 0.4671

Mcnemar's Test P-Value : 0.009208

Sensitivity : 0.8884

Specificity : 0.5526

Pos Pred Value : 0.8083

Neg Pred Value : 0.7000

Prevalence : 0.6798

Detection Rate : 0.6039

Detection Prevalence : 0.7472

Balanced Accuracy : 0.7205

'Positive' Class : 0

برای داده آموزش

confusionMatrix(predict(tune_fit, train1[-16], type = 'class'),train1$upquantile)

Confusion Matrix and Statistics

Reference

Prediction 0 1

0 1026 327

1 158 689

Accuracy : 0.7795

95% CI : (0.7616, 0.7967)

No Information Rate : 0.5382

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.5512

Mcnemar's Test P-Value : 2.375e-14

Sensitivity : 0.8666

Specificity : 0.6781

Pos Pred Value : 0.7583

Neg Pred Value : 0.8135

Prevalence : 0.5382

Detection Rate : 0.4664

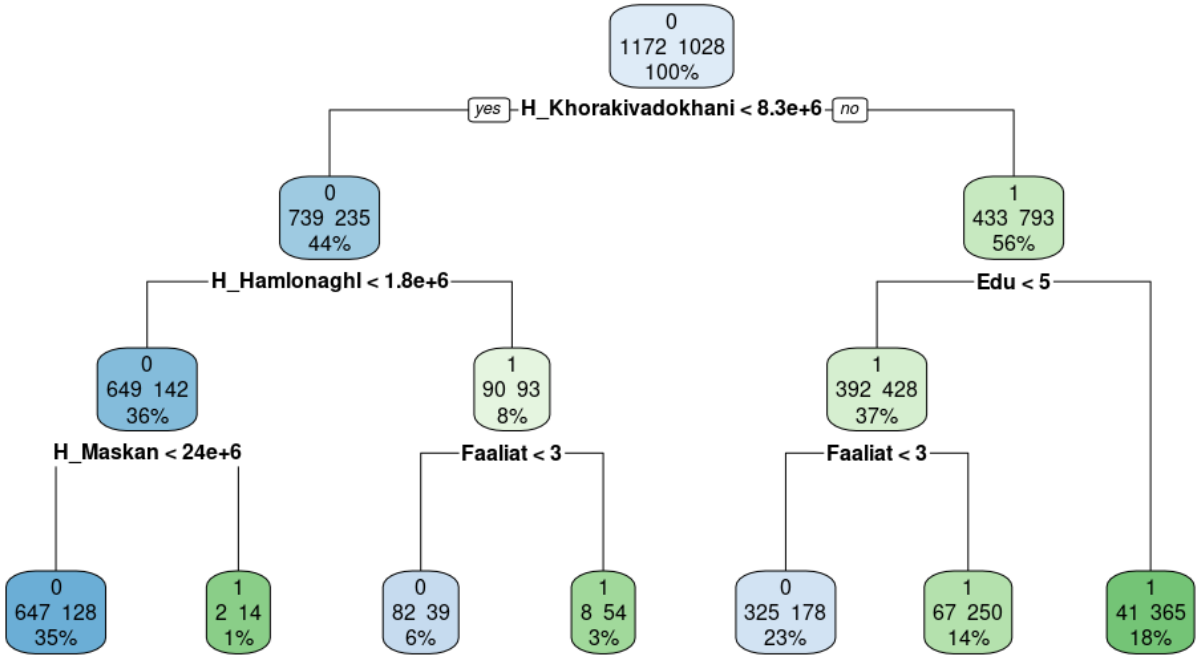
Detection Prevalence : 0.6150

Balanced Accuracy : 0.7724

'Positive' Class : 0

درخت را با دستور زیر رسم می‌کنیم.

```
rpart.plot(tune_fit, extra = 101)
```



درصدهای موجود در برگ‌ها درصدی از داده را نشان می‌دهند که به آن برگ تعلق دارد.

در هر عمیقی مجموع درصدها مساوی ۱۰۰ است.

مدل KNN (دهک ۷۰)

در ادامه مدل KNN را پیاده می‌کنیم.

```
#KNN 70 quantile
library(class)
Size=c(3,4,5,7,10,15,20,30,40,60,80,100,200,300,400,600,1000)

for (i in 1:length(Size)) {
  model <- knn(train = train1[-16], test = valid1[-16],cl=train1$upquantile, k=i)
  x=confusionMatrix(model,valid1$upquantile)$table
  accuracy_Test <- sum(diag(x)) / sum(x)
```

```
  print(accuracy_Test)}
[1] 0.7114846
[1] 0.7058824
[1] 0.6946779
[1] 0.6918768
[1] 0.67507
[1] 0.6498599
[1] 0.6694678
[1] 0.7030812
[1] 0.697479
[1] 0.697479
[1] 0.7142857
[1] 0.7282913
[1] 0.7338936
[1] 0.7366947
[1] 0.7338936
[1] 0.7226891
[1] 0.7254902
```

با توجه به میزان پیچیدگی مدل k=3 انتخاب مناسبی است.

عمل‌کرد مدل برروی داده تست را بررسی می‌کنیم.

```
model <- knn(train = train1[-16], test = test1[-16],cl=train1$upquantile, k=3)
confusionMatrix(model,test1$upquantile)
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 167  48
         1  75  66
```

Accuracy : 0.6545

95% CI : (0.6026, 0.7038)

No Information Rate : 0.6798

P-Value [Acc > NIR] : 0.85952

Kappa : 0.2532

Mcnemar's Test P-Value : 0.01906

Sensitivity : 0.6901

Specificity : 0.5789

Pos Pred Value : 0.7767

Neg Pred Value : 0.4681

Prevalence : 0.6798

Detection Rate : 0.4691

Detection Prevalence : 0.6039

Balanced Accuracy : 0.6345

'Positive' Class : 0

مشاهده می‌شود که مدل دقت خوبی ندارد.

عمل‌کرد مدل برروی داده آموزش

```
model <- knn(train = train1[-16], test = train1[-16],cl=train1$upquantile, k=3)
confusionMatrix(model,train1$upquantile)
```

Confusion Matrix and Statistics

Prediction	Reference 0	1
0	900	39
1	284	977

Accuracy : 0.8532

95% CI : (0.8377, 0.8677)

No Information Rate : 0.5382

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.7096

Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.7601

Specificity : 0.9616

Pos Pred Value : 0.9585

Neg Pred Value : 0.7748

Prevalence : 0.5382

Detection Rate : 0.4091

Detection Prevalence : 0.4268

Balanced Accuracy : 0.8609

'Positive' Class : 0

مدل شبکه عصبی (چندک ۷۰)

در ادامه از شبکه‌عصبی برای مدل کردن متغیر پاسخ استفاده می‌کنیم.

ابتدا باید داده‌های گسسته را به صورت One Hot کرد و متغیرهای پیوسته را Scale کرد.

سپس باید مدل را با دستور formula تعریف کرد.

کد زیر وظایف بالا را انجام می‌دهد.

```
library(neuralnet)
# fit neural network
```

```
pp=preProcess(train1, method = "range")
m <- model.matrix( ~ ., data = predict(pp,train1))
col_list <- paste(c(colnames(m[,-c(1,50)])),collapse="+")
col_list <- paste(c("upquantile1~",col_list),collapse="")
f <- formula(col_list)
pp=preProcess(valid1, method = "range")
m2 <- model.matrix( ~ ., data = predict(pp,valid1))
```

سپس باید تعداد لایه‌ها و نورون‌های شبکه را تنظیم کرد که برای این موضوع از داده اعتبارسنجی استفاده می‌کنیم، لایه‌های مختلفی که تست را روی ان‌ها انجام می‌دهیم ->۵،۱۲، (۱،۲)

```
Size=list(5,12,c(1,2))
for (i in 1:length(Size)) {
  nn=neuralnet(f,data=m, hidden=Size[[i]],act.fct = "logistic",linear.output = FALSE)
  Predict=predict(nn,m2[,-c(1,50)],rep=1)
  Predict=ifelse(Predict>0.5,1,0)
  table_mat <- table(valid1$upquantile, Predict)
  accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
  print(accuracy_Test)}
```

دقت مدل‌ها به شرح زیر است.

```
[1] 0.6778711
[1] 0.6890756
[1] 0.7030812
```

شبکه نهایی بهترین عمل‌کرد را داشته است.

حال عمل‌کرد این شبکه برروی داده تست را بررسی می‌کنیم.

```
nn=neuralnet(f,data=m, hidden=c(1,2),act.fct = "logistic",linear.output = FALSE)
pp=preProcess(test1, method = "range")
m <- model.matrix( ~ ., data = predict(pp,test1))
Predict=predict(nn,m[,-c(1,50)],rep=1)
Predict=ifelse(Predict>0.5,1,0)
```

جزییات تحت داده تست به شرح زیر است.

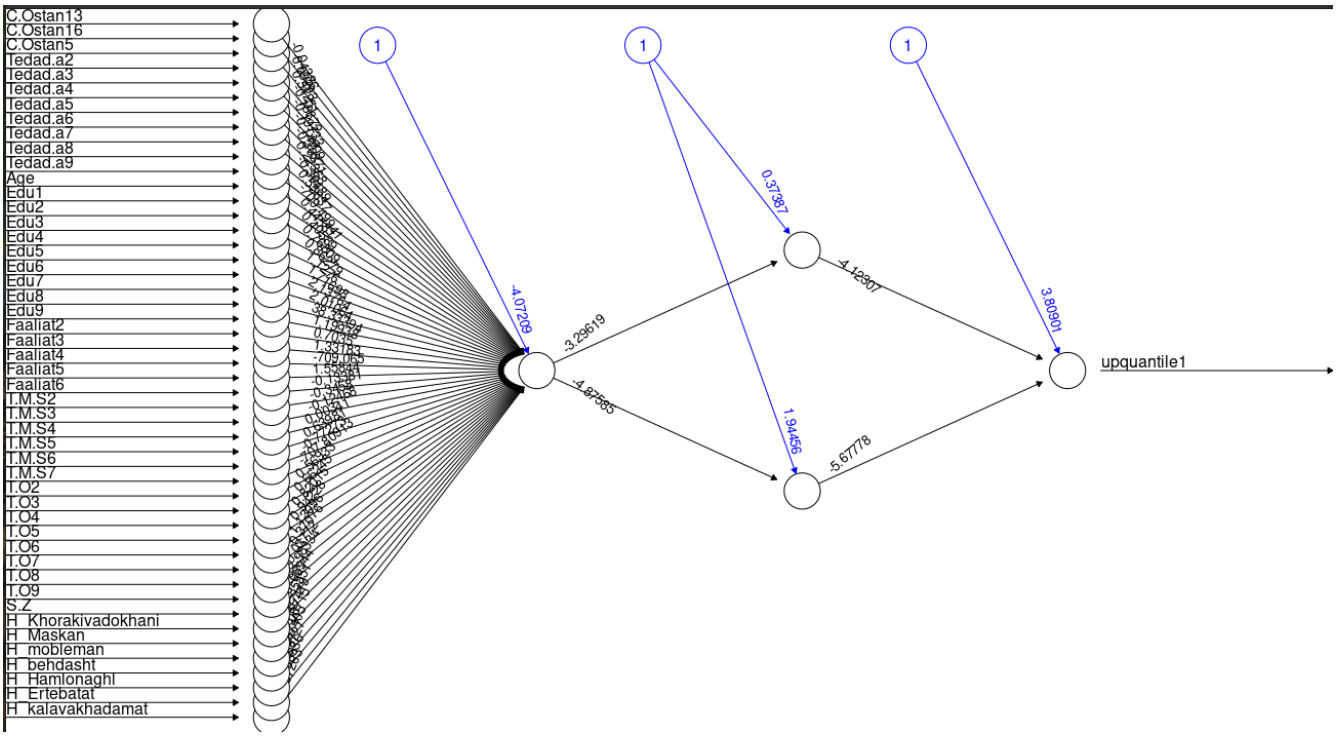
```
confusionMatrix(factor(Predict),test1$upquantile)


Confusion Matrix and Statistics


          Reference
Prediction   0   1
         0 136   9
         1 106 105


               Accuracy : 0.677
                 95% CI : (0.6257, 0.7253)
    No Information Rate : 0.6798
    P-Value [Acc > NIR] : 0.5702


                  Kappa : 0.3943


 Mcnemar's Test P-Value : <2e-16


            Sensitivity : 0.5620
            Specificity : 0.9211
         Pos Pred Value : 0.9379
         Neg Pred Value : 0.4976
             Prevalence : 0.6798
         Detection Rate : 0.3820
   Detection Prevalence : 0.4073
      Balanced Accuracy : 0.7415


       'Positive' Class : 0
```

جزییات تحت داده آموزش به شرح زیر است.

```
Predict=predict(nn,m[,-c(1,50)],rep=1)
Predict=ifelse(Predict>0.5,1,0)
confusionMatrix(factor(Predict),train1$upquantile)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1083  255
         1  101  761

               Accuracy : 0.8382
                 95% CI : (0.8221, 0.8533)
    No Information Rate : 0.5382
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6709

 Mcnemar's Test P-Value : 5.104e-16

            Sensitivity : 0.9147
            Specificity : 0.7490
         Pos Pred Value : 0.8094
         Neg Pred Value : 0.8828
             Prevalence : 0.5382
         Detection Rate : 0.4923
   Detection Prevalence : 0.6082
      Balanced Accuracy : 0.8319
```

'Positive' Class : 0

شبکه را با دستور plot(nn) رسم می‌کنیم، خروجی به شرح زیر است.



مدل‌سازی دهک ۳۰

ابتدا مشابه قسمت قبل داده را متوازن کرده و سپس به سه قسمت اموزش، تست، اعتبارسنجی تقسیم می‌کنیم.

کد زیراعمال بالا را انجام می‌دهد.

```
#Quantile 30

#Train , Test, Validation

smp_siz = floor(0.7*nrow(Data))
```

ابتدا مشابه حالت قبل داده‌ها را بالانس رده و به سه دسته مشابه قسمت قبل تقسیم می‌کنیم.

```
train_ind = sample(seq_len(nrow(Data)),size = smp_siz)  # Randomly identifies therows equal to sample size ( defined in previous instruction) from  all the rows of Smarket dataset and stores the row number in train_ind

train =Data[train_ind,] #creates the training dataset with row numbers stored in train_ind

remaind=Data[-train_ind,]

#test validation
```

```
smp_siz = floor(0.5*nrow(remaind))

test_ind = sample(seq_len(nrow(remaind)),size = smp_siz)  # Randomly identifies therows equal to sample size ( defined in previous instruction) from  all the rows of Smarket dataset and stores the row number in train_ind

test =remaind[test_ind,] #creates the training dataset with row numbers stored in train_ind

valid=remaind[-test_ind,]


rm(smp_siz)

#Solve inbalanced Problem 30 Quantile

train=ovun.sample(downquantile ~ ., data = train, method = "over",N = 2200)$data

table(train$downquantile)# data balanced
```

حال به انتخاب ویژگی‌های مناسب می‌پردازیم.(مشابه قسمت قبل)

کد زیر این وظیفه را انجام می‌دهد.

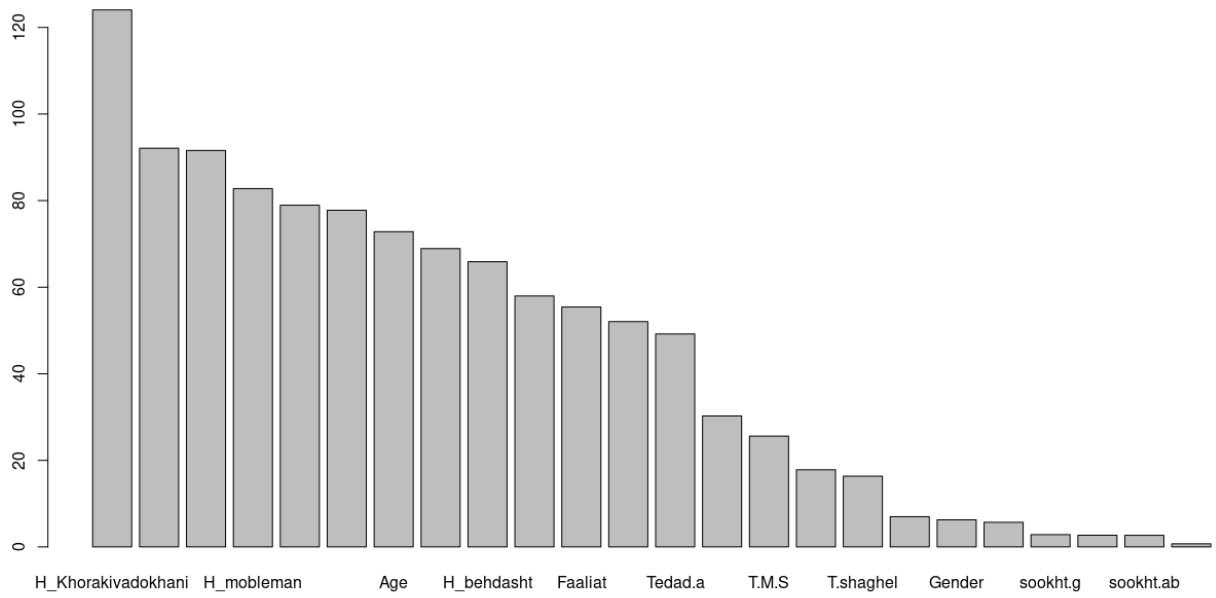
```
library(randomForest)

fit_rf = randomForest(train[1:24],train$downquantile, data=train)

#Accuracy

library(caret)

confusionMatrix(predict(fit_rf,remaind[1:24]),remaind$downquantile)

# Create an importance based on mean decreasing gini


importance(fit_rf)[order(importance(fit_rf),decreasing = TRUE),]

barplot(importance(fit_rf)[order(importance(fit_rf),decreasing = TRUE),])

Names=names(importance(fit_rf)[importance(fit_rf)>20,])
```

خروجی به شرح زیر است.

ویژگی‌های انتخاب شده به شرح زیر است.

```
[1] "C.Ostan"           "Tedad.a"           "Age"               "Edu"
 [5] "Faaliat"           "T.M.S"             "T.O"               "S.Z"
 [9] "H_Khorakivadokhani" "H_Maskan"          "H_mobleman"        "H_behdasht"
[13] "H_Hamlonaghl"      "H_Ertebatat"       "H_kalavakhadamat"
```

نمودار میله‌ای اهمیت متغیرها نیز به صورت زیر می‌باشد.



حال داده‌ها را باتوجه به ویژگی حاصل شده حرص می‌کنیم.

```
train1=train[,c(Names,"downquantile")]
test1=test[,c(Names,"downquantile")]
valid1=valid[,c(Names,"downquantile")]
remaind1=remaind[,c(Names,"downquantile")]
```

مدل لجستیک (دهک ۳۰)

ابتدا مدل لجستیک را پیاده می‌کنیم.

از آنجا که پارامتری برای تنظیم شدن وجود ندارن مجموعه داده تست و اعتبارسنجی را مخلوط می‌کنیم.

```
#Logestic Regression for ۳۰ quantile
model <- glm(downquantile~.,family=binomial(link='logit'),data=train1)
confusionMatrix(factor(ifelse(predict(model,remaind1[-16])>0,1,0)),remaind1$downquantile)
```

خروجی برای داده تست شرح زیر است.

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction   0   1
         0 356  49
         1 151 157

               Accuracy : 0.7195
                 95% CI : (0.685, 0.7522)
    No Information Rate : 0.7111
    P-Value [Acc > NIR] : 0.3264

                  Kappa : 0.4048

 Mcnemar's Test P-Value : 9.213e-13

            Sensitivity : 0.7022
            Specificity : 0.7621
         Pos Pred Value : 0.8790
         Neg Pred Value : 0.5097
             Prevalence : 0.7111
         Detection Rate : 0.4993
   Detection Prevalence : 0.5680
      Balanced Accuracy : 0.7322

       'Positive' Class : 0
```

جزییات برازش برروی داده آموزش به شرح زیر است.

```
confusionMatrix(factor(ifelse(predict(model,train1[-16])>0,1,0)),train1$downquantile)
```

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction   0   1
         0 873 244
         1 282 801

               Accuracy : 0.7609
                 95% CI : (0.7425, 0.7786)
    No Information Rate : 0.525
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.5214

 Mcnemar's Test P-Value : 0.1067

            Sensitivity : 0.7558
            Specificity : 0.7665
         Pos Pred Value : 0.7816
         Neg Pred Value : 0.7396
             Prevalence : 0.5250
         Detection Rate : 0.3968
   Detection Prevalence : 0.5077
      Balanced Accuracy : 0.7612

       'Positive' Class : 0
```

مدل درخت تصمیم (دهک 30)

مانند قسمت قبل مدل را پیاده می‌کنیم.

```
library(rpart)
library(rpart.plot)
```

```
accuracy_tune <- function(fit,data_test) {
  predict_unseen <- predict(fit, data_test, type = 'class')
  table_mat <- table(data_test$downquantile, predict_unseen)
  accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
  accuracy_Test
}
Minsplit=c(4,8,25,50,15)
Minbucket=c((round(Minsplit)+1)/3)
Maxdepth=c(3,25,5,7,9)
for (i in 1:length(Maxdepth)) {
  control <- rpart.control(minsplit = Minsplit[i],
                           minbucket = Minbucket[i],
                           maxdepth = Maxdepth[i])
  tune_fit <- rpart(downquantile~., data = train1, method = 'class', control = control)
  print(accuracy_tune(tune_fit,valid1))
}
```

خروجی به شرح زیر است.

```
[1] 0.7282913
[1] 0.7226891
[1] 0.7226891
[1] 0.7226891
[1] 0.7226891
```

مشاهده می‌کنیم که تفاوت آنچنانی بین مدل‌ها وجود ندارد درنتیجه ساده‌ترین مدل را انتخاب می‌کنیم.

```
#best Choineon test set
control <- rpart.control(minsplit = 4,
                         minbucket = round(5 / 3),
                         maxdepth = 3,)
tune_fit <- rpart(downquantile~., data = train1, method = 'class', control = control)
```

```
print(accuracy_tune(tune_fit,test))
rpart.plot(tune_fit, extra = 101)
```

دقت مدل به شرح زیر است.

```
[1] 0.6994382
```

جزییات عمل‌کرد مدل برروی داده تست به شرح زیر است.

```
confusionMatrix(predict(tune_fit, test1[-16], type = 'class'),test1$downquantile)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 165  28
         1  90  73

               Accuracy : 0.6685
                 95% CI : (0.617, 0.7173)
    No Information Rate : 0.7163
    P-Value [Acc > NIR] : 0.979

                  Kappa : 0.312

 Mcnemar's Test P-Value : 1.96e-08

            Sensitivity : 0.6471
            Specificity : 0.7228
         Pos Pred Value : 0.8549
         Neg Pred Value : 0.4479
             Prevalence : 0.7163
```

Detection Rate : 0.4635

Detection Prevalence : 0.5421

Balanced Accuracy : 0.6849

'Positive' Class : 0

جزییات عمل‌کرد مدل برروی داده آموزش به شرح زیر است.

```
confusionMatrix(predict(tune_fit, train1[-16], type = 'class'),train1$downquantile)
```

Confusion Matrix and Statistics

Prediction	Reference 0	1
0	838	261
1	317	784

Accuracy : 0.7373

95% CI : (0.7183, 0.7556)

No Information Rate : 0.525

P-Value [Acc > NIR] : < 2e-16

Kappa : 0.4746

Mcnemar's Test P-Value : 0.02216

Sensitivity : 0.7255

Specificity : 0.7502

Pos Pred Value : 0.7625

Neg Pred Value : 0.7121

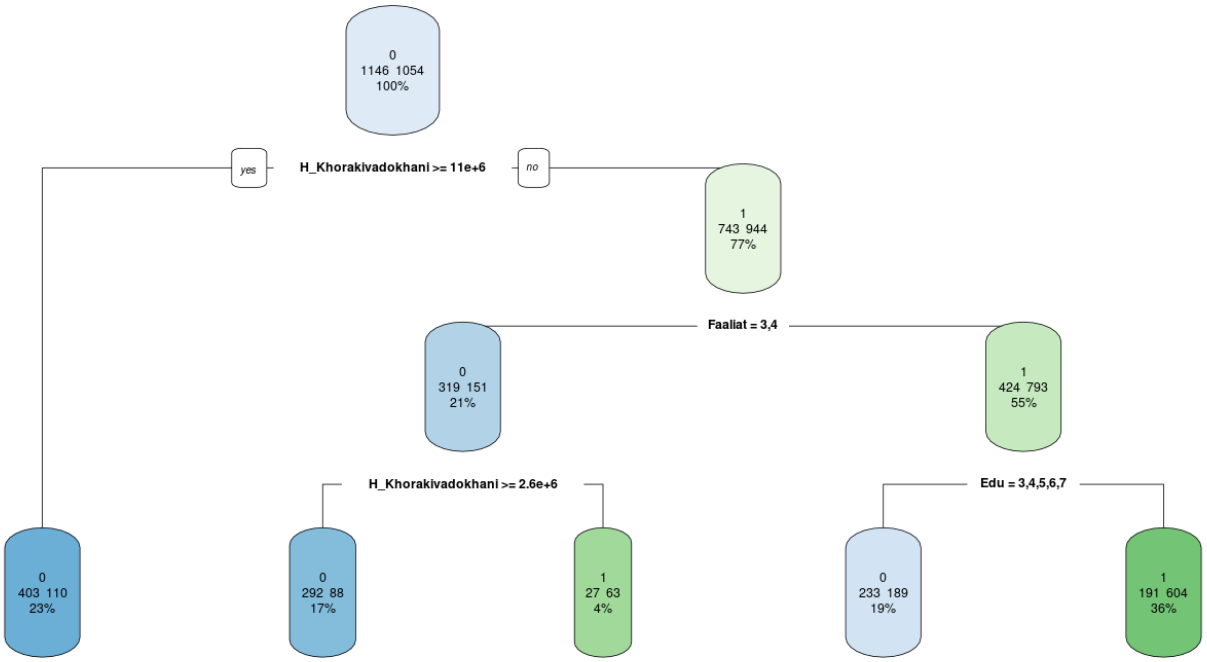
Prevalence : 0.5250

Detection Rate : 0.3809

Detection Prevalence : 0.4995

Balanced Accuracy : 0.7379

'Positive' Class : 0



مدل KNN (دهک 30)

```
library(class)
Size=c(3,4,5,7,10,15,20,30,40,60,80,100,200,300,400,600,1000)
for (i in 1:length(Size)) {
  model <- knn(train = train1[-16], test = valid1[-16],cl=train1$downquantile, k=i)
  x=confusionMatrix(model,valid1$downquantile)$table
  accuracy_Test <- sum(diag(x)) / sum(x)
  print(accuracy_Test)}
```

خروجی به شرح زیر است.

[1] 0.6722689

[1] 0.6302521

[1] 0.5742297

[1] 0.5882353

[1] 0.6190476

[1] 0.6302521

[1] 0.6162465

[1] 0.6022409

[1] 0.5742297

[1] 0.605042

[1] 0.605042

[1] 0.6246499

[1] 0.6190476

[1] 0.6190476

[1] 0.627451

[1] 0.6246499

[1] 0.6414566

به ظاهر k=3 انتخاب مناسبی است.

حال برروی داده تست ، مدل را پیاده می‌کنیم.

جزییات عمل‌کرد مدل برروی داده تست به شرح زیر است.

```
model <- knn(train = train1[-16], test = test1[-16],cl=train1$downquantile, k=3)
confusionMatrix(model,test1$downquantile)
```

Confusion Matrix and Statistics

```
          Reference
Prediction   0   1
         0 149  38
         1 106  63

               Accuracy : 0.5955
                 95% CI : (0.5425, 0.6469)
```

No Information Rate : 0.7163

P-Value [Acc > NIR] : 1

Kappa : 0.1729

Mcnemar's Test P-Value : 2.36e-08

Sensitivity : 0.5843

Specificity : 0.6238

Pos Pred Value : 0.7968

Neg Pred Value : 0.3728

Prevalence : 0.7163

Detection Rate : 0.4185

Detection Prevalence : 0.5253

Balanced Accuracy : 0.6040

'Positive' Class : 0

حال برروی داده آموزش دقت مدل را می‌سنجیم.

```
model <- knn(train = train1[-16], test = train1[-16],cl=train1$downquantile, k=3)
confusionMatrix(model,train1$downquantile)
```

Confusion Matrix and Statistics

Prediction	Reference 0	1
0	806	49
1	349	996

Accuracy : 0.8191

```
                95% CI : (0.8024, 0.835)
    No Information Rate : 0.525
    P-Value [Acc > NIR] : < 2.2e-16

                 Kappa : 0.6422

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.6978
            Specificity : 0.9531
         Pos Pred Value : 0.9427
         Neg Pred Value : 0.7405
             Prevalence : 0.5250
         Detection Rate : 0.3664
   Detection Prevalence : 0.3886
      Balanced Accuracy : 0.8255

       'Positive' Class : 0
```

مدل شبکه عصبی (چندک ۳۰)

مشابه قسمت قبل کد را پیاده‌سازی می کنیم.

```
library(neuralnet)
# fit neural network
pp=preProcess(train1, method = "range")
m <- model.matrix( ~ ., data = predict(pp,train1))
col_list <- paste(c(colnames(m[,-c(1,ncol(m))])),collapse="+")
col_list <- paste(c("downquantile1~",col_list),collapse="")
f <- formula(col_list)
pp=preProcess(valid1, method = "range")
Size=list(2,3,10)
```

```
m2 <- model.matrix( ~ ., data = predict(pp,valid1))
for (i in 1:length(Size)) {
  nn=neuralnet(f,data=m, hidden=Size[[i]],act.fct = "logistic",linear.output = FALSE)
  Predict=predict(nn,m2[,-c(1,ncol(m2))],rep=1)
  Predict=ifelse(Predict>0.5,1,0)
  table_mat <- table(valid1$downquantile, Predict)
  accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
  print(accuracy_Test)
}
```

خروجی به شرح زیر است.

```
[1] 0.7226891
[1] 0.6778711
[1] 0.7058824
```

به‌نظر می‌رسد که شبکه عصبی با ۲ نورون عمل‌کرد خوبی دارد. حال عمل‌کرد مدل برروی داده آموزش را می‌سنجیم.

```
nn=neuralnet(f,data=m, hidden=c(1,2),act.fct = "logistic",linear.output = FALSE)
Predict=predict(nn,m[,-c(1,50)],rep=1)
Predict=ifelse(Predict>0.5,1,0)
confusionMatrix(factor(Predict),train1$downquantile)

Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 882 244
         1 273 801

               Accuracy : 0.765
                 95% CI : (0.7467, 0.7826)
```

No Information Rate : 0.525

P-Value [Acc > NIR] : <2e-16

Kappa : 0.5294

Mcnemar's Test P-Value : 0.2182

Sensitivity : 0.7636

Specificity : 0.7665

Pos Pred Value : 0.7833

Neg Pred Value : 0.7458

Prevalence : 0.5250

Detection Rate : 0.4009

Detection Prevalence : 0.5118

Balanced Accuracy : 0.7651

'Positive' Class : 0

حال عمل‌کرد را برروی داده تست می‌سنجیم

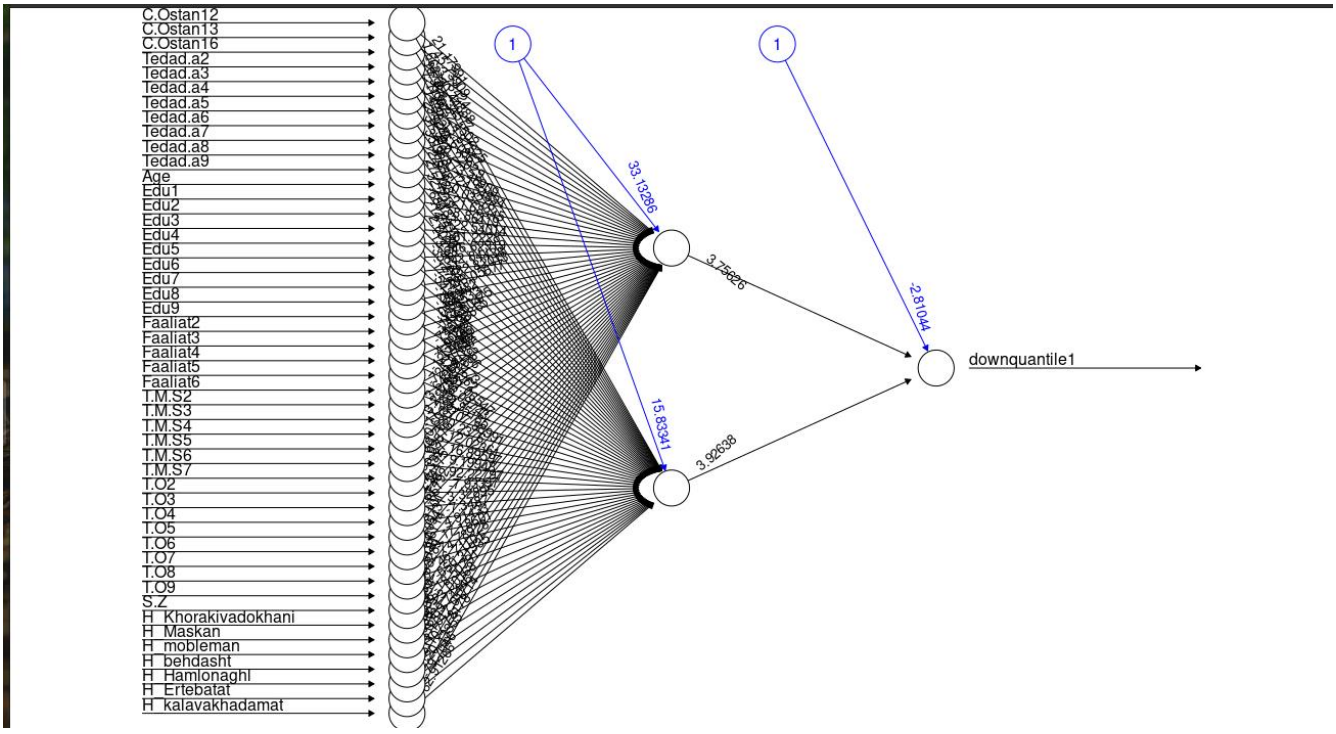
```
pp=preProcess(test1, method = "range")
m <- model.matrix( ~ ., data = predict(pp,test1))
Predict=predict(nn,m[,-c(1,50)],rep=1)
Predict=ifelse(Predict>0.5,1,0)
confusionMatrix(factor(Predict),test1$downquantile)
```

Confusion Matrix and Statistics

	Reference	
Prediction	0	1
0	245	77

```
  1  10  24

               Accuracy : 0.7556
                 95% CI : (0.7076, 0.7994)
    No Information Rate : 0.7163
    P-Value [Acc > NIR] : 0.05465

                  Kappa : 0.2481

 Mcnemar's Test P-Value : 1.484e-12

            Sensitivity : 0.9608
            Specificity : 0.2376
         Pos Pred Value : 0.7609
         Neg Pred Value : 0.7059
             Prevalence : 0.7163
         Detection Rate : 0.6882
   Detection Prevalence : 0.9045
      Balanced Accuracy : 0.5992

       'Positive' Class : 0
```

شکل مدل با دستور plot(nn) به شکل زیر است.


C.Ostan12
C.Ostan13
C.Ostan16
Tedad.a2
Tedad.a3
Tedad.a4
Tedad.a5
Tedad.a6
Tedad.a7
Tedad.a8
Tedad.a9
Age
Edu1
Edu2
Edu3
Edu4
Edu5
Edu6
Edu7
Edu8
Edu9
Faaliat2
Faaliat3
Faaliat4
Faaliat5
Faaliat6
T.M.S2
T.M.S3
T.M.S4
T.M.S5
T.M.S6
T.M.S7
T.O2
T.O3
T.O4
T.O5
T.O6
T.O7
T.O8
T.O9
S.Z
H_Khorakivadokhani
H_Maskan
H_mobleman
H_behdasht
H_Hamlonaghl
H_Ertebatat
H_kalavakhadamat
1
1
33.13286
15.83341
3.75696
3.92638
-2.81044
downquantile1