Pi Greco Team

Relazione finale

Applicazione tecniche di regolarizzazione su un dataset contenente informazioni mediche riguardo gli attacchi di cuore e analisi predittiva

Si riporta inoltre, in appendice a questo documento, l'applicazione della regressione beta sul dataset "Admission Predict" la cui variabile dipendente è la percentuale di possibilità di essere ammesso all'università e l'applicazione a scopo dimostrativo di un algoritmo di machine learning (Random Forest Classifier) sul dataset "Heart Attack Prediction" la cui variabile dipendente binaria denota se si avrà un attacco di cuore o meno.

Membri del gruppo:

Marco Greco | 242455 | Ing. Informatica

Francesco Bonacci | 242713 | Ing. Informatica

Davide Musitano | 242716 | Ing. Informatica

Bruno Guzzo | 242504 | Ing. Informatica

Giuseppe Mallamo | 242767 | Ing. Informatica

Fabrizio Tempo | 247195 | Ing. Informatica

Piercosimo Nicolò Baffa | 242609 | Ing. Informatica

Modello per la previsione degli attacchi di cuore

Descrizione del dataset

Il dataset che abbiamo scelto è un dataset di informazioni mediche raccolte da 303 pazienti che hanno avuto o meno un attacco di cuore inerenti ad una visita medica da sforzo, per tale motivo si riportano anche altri valori come la presenza di malattie del sangue, il colesterolo, il risultato dell'elettrocardiogramma, etc.

Il dataset contiene 14 features di cui si riporta una descrizione del seguente elenco puntato:

- **age**: Età del paziente
- **sex**: Sesso del paziente
- **cp**: Tipo di dolore cardiaco
    - **0**: Asintomatico
    - **1**: Angina atipica
    - **2**: Non angina
    - **3**: Normale angina
- **trtbps**: Pressione sanguigna del paziente
- **chol**: Colesterolo del paziente
- **restecg**: Risultato dell'elettrocardiogramma
    - **0**: Ipertrofia del ventricolo sinistro
    - **1**: Normale
    - **2**: Curva ST-T abnormale
- **thalachh**: Battito cardiaco massimo raggiunto dal paziente
- **exng**: Angina (dolore al petto) dovuto all'esercizio fisico
- **oldpeak**: Depressione nella curva ST-T dovuta al riposo
- **slp**: Pendenza della curva ST-T indotta dall'esercizio fisico
    - **0**: Discendente
    - **1**: Piatta
    - **2**: Crescente
- **caa**: Numero di vasi cardiaci non ostruiti
    - **0**: Tutti i vasi cardiaci ostruiti
    - **1**: Un solo vaso cardiaco libero
    - **2**: Due vasi cardiaci liberi
    - **3**: Tre vasi cardiaci liberi
    - **4**: Tutti (quattro) i vasi cardiaci liberi
- **thall**: Malattia sanguigna chiamata Talassemia
    - **0**: valore nullo, nessuna informazione

    - **1**: difetto fisso del flusso sanguigno
    - **2**: flusso sanguigno normale (Talassemia assente)
    - **3**: difetto reversibile del flusso sanguigno
- **output**: Attacco di cuore
    - **0**: No
    - **1**: Si

Il nostro obiettivo è quello di stimare il miglior modello per la previsione di un possibile attacco di cuore sulla base delle informazioni di un paziente, per far ciò adoperiamo il modello lineare con le tecniche di regolarizzazione e cross validation viste durante il corso. Per una visione più dettagliata di tutte le operazioni effettuate rimandiamo alla visione dello script R (linguaggio R) in allegato a questo documento.

Import del dataset e prima analisi

Per prima cosa impostiamo il seed (seme) in modo da avere gli stessi risultati ad ogni esecuzione dello script, leggiamo i dati presenti nel file “heart.csv” (presente il allegato), facciamo una prima analisi dei dati importati e verifichiamo che non vi siano valori nulli.

```
DataSet <- read.csv(file = path, sep = ",", header = TRUE)
dim(DataSet)
describe(DataSet)
head(DataSet)

# Confermiamo che non ci sono valori mancanti
sum(is.na(DataSet))
```

```
> dim(DataSet)
[1] 303  14
> describe(DataSet)
DataSet

 14  Variables      303  Observations
--------------------------------------------------------------------------------------------------------------------------------
age
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     303        0       41    0.999    54.37    10.36     39.1     42.0     47.5     55.0     61.0     66.0     68.0

lowest : 29 34 35 37 38, highest: 70 71 74 76 77
--------------------------------------------------------------------------------------------------------------------------------
sex
       n  missing distinct     Info      Sum     Mean      Gmd
     303        0        2    0.649      207   0.6832   0.4343

--------------------------------------------------------------------------------------------------------------------------------
cp
       n  missing distinct     Info     Mean      Gmd
     303        0        4    0.866    0.967    1.105

Value          0     1     2     3
Frequency    143    50    87    23
Proportion 0.472 0.165 0.287 0.076
--------------------------------------------------------------------------------------------------------------------------------
trtbps
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     303        0       49    0.995    131.6    19.32      108      110      120      130      140      152      160

lowest :  94 100 101 102 104, highest: 174 178 180 192 200
--------------------------------------------------------------------------------------------------------------------------------
chol
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     303        0      152        1    246.3    55.95    175.0    188.0    211.0    240.0    274.5    308.8    326.9

lowest : 126 131 141 149 157, highest: 394 407 409 417 564
--------------------------------------------------------------------------------------------------------------------------------
fbs
       n  missing distinct     Info      Sum     Mean      Gmd
     303        0        2    0.379       45   0.1485   0.2538

--------------------------------------------------------------------------------------------------------------------------------
 restecg
        n  missing distinct     Info     Mean      Gmd
      303        0        3     0.76   0.5281   0.5274

 Value          0     1     2
 Frequency    147   152     4
 Proportion 0.485 0.502 0.013
 --------------------------------------------------------------------------------------------------------------------------------
 thalachh
        n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
      303        0       91        1    149.6    25.77    108.1    116.0    133.5    153.0    166.0    176.6    181.9

 lowest :  71  88  90  95  96, highest: 190 192 194 195 202
 --------------------------------------------------------------------------------------------------------------------------------
 exng
        n  missing distinct     Info      Sum     Mean      Gmd
      303        0        2     0.66       99   0.3267   0.4414

 --------------------------------------------------------------------------------------------------------------------------------
 oldpeak
        n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
      303        0       40    0.964     1.04    1.225      0.0      0.0      0.0      0.8      1.6      2.8      3.4

 lowest : 0.0 0.1 0.2 0.3 0.4, highest: 4.0 4.2 4.4 5.6 6.2
 --------------------------------------------------------------------------------------------------------------------------------
 slp
        n  missing distinct     Info     Mean      Gmd
      303        0        3    0.798    1.399   0.6291

 Value          0     1     2
 Frequency     21   140   142
 Proportion 0.069 0.462 0.469
 --------------------------------------------------------------------------------------------------------------------------------
 caa
        n  missing distinct     Info     Mean      Gmd
      303        0        5    0.795   0.7294    1.005

 lowest : 0 1 2 3 4, highest: 0 1 2 3 4

 Value          0     1     2     3     4
 Frequency    175    65    38    20     5
 Proportion 0.578 0.215 0.125 0.066 0.017
 --------------------------------------------------------------------------------------------------------------------------------
```

```
thall
       n  missing distinct     Info     Mean      Gmd
     303        0        4    0.778    2.314   0.6125

Value          0     1     2     3
Frequency      2    18   166   117
Proportion 0.007 0.059 0.548 0.386
-------------------------------------------------------------------------------------------------------------------
output
       n  missing distinct     Info      Sum     Mean      Gmd
     303        0        2    0.744      165   0.5446   0.4977

-------------------------------------------------------------------------------------------------------------------
> head(DataSet)
  age sex cp trtbps chol fbs restecg thalachh exng oldpeak slp caa thall output
1  63   1  3    145  233   1       0      150    0     2.3   0   0     1      1
2  37   1  2    130  250   0       1      187    0     3.5   0   0     2      1
3  41   0  1    130  204   0       0      172    0     1.4   2   0     2      1
4  56   1  1    120  236   0       1      178    0     0.8   2   0     2      1
5  57   0  0    120  354   0       1      163    1     0.6   2   0     2      1
6  57   1  0    140  192   0       1      148    0     0.4   1   0     1      1
```

Dal risultato delle operazioni effettuate possiamo trarre le seguenti conclusioni:

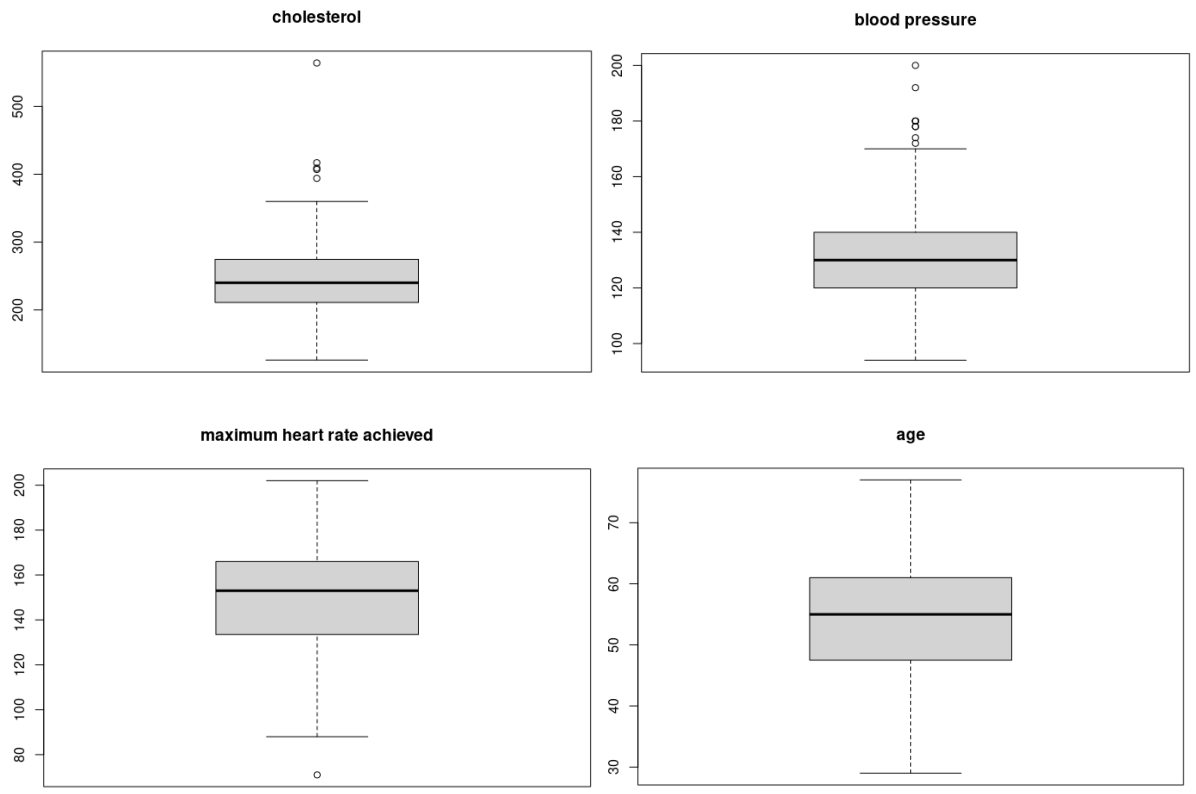
1. Non ci sono valori mancanti nel dataset,
2. È necessario effettuare la standardizzazione dei regressori in quanto ci sono diversi regressori con unità di misura e ordine di grandezza differenti, ad esempio colesterolo, pressione e battito cardiaco massimo. A questo proposito è bene ricordare che la funzione glmnet, usata in seguito per applicare le tecniche di regolarizzazione, effettuate di default la standardizzazione dei regressori.
3. Sono presenti alcuni outliers nei regressori del colesterolo e della pressione cardiaca. Possiamo intuire che si tratta di pazienti con valori clinici critici.

Per evitare di perdere informazioni importanti al fine della previsione evitiamo di eliminare gli outliers dal momento che potrebbero costituire un'importante base di conoscenza in fase di addestramento del modello (stima dei parametri dei modelli sul test set).

Visualizzazione degli outliers

Per visualizzare bene la come tali outliers sono disposti riportiamo di seguito alcuni boxplot.

```r
# Osserviamo la presenza di outlier nei valori del colesterolo, nello specifico
# un valore oltre 500.
boxplot(DataSet$chol, main = "cholesterol")

# Osserviamo la presenza di outlier nei valori della pressione saguigna, nello specifico
# si rileva la presenza valori oltre 170.
boxplot(DataSet$trtbps, main = "blood pressure")

# Osserviamo la presenza di un singolo outlier nei valori massimi del battito cardiaco,
# in particolare un valore minore di 80.
boxplot(DataSet$thalachh,  main = "maximum heart rate achieved")

# Non si rileva la presenza di outlier nei valori dell'età
boxplot(DataSet$age,  main = "age")
```



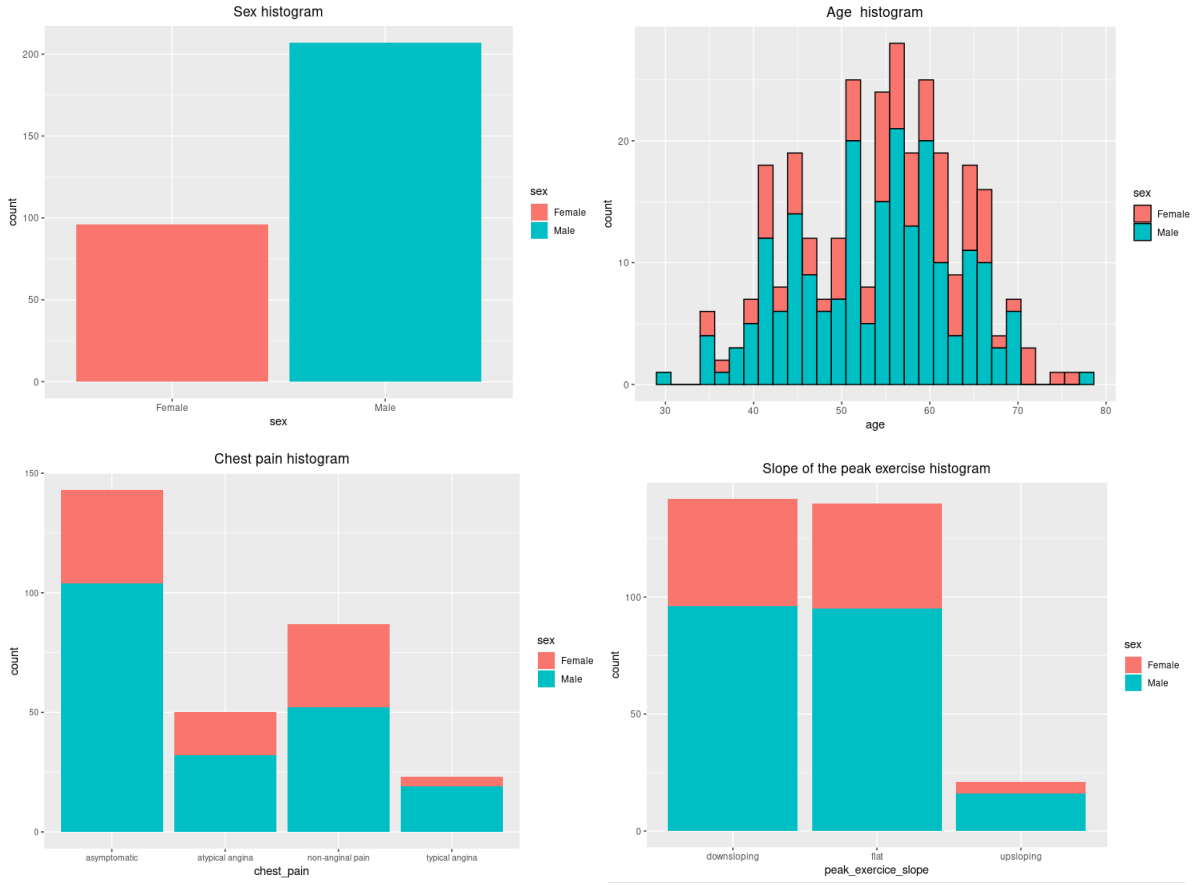
È quindi evidente la presenza di outliers che ci indicano valori anomali di pressione e colesterolo in determinati pazienti

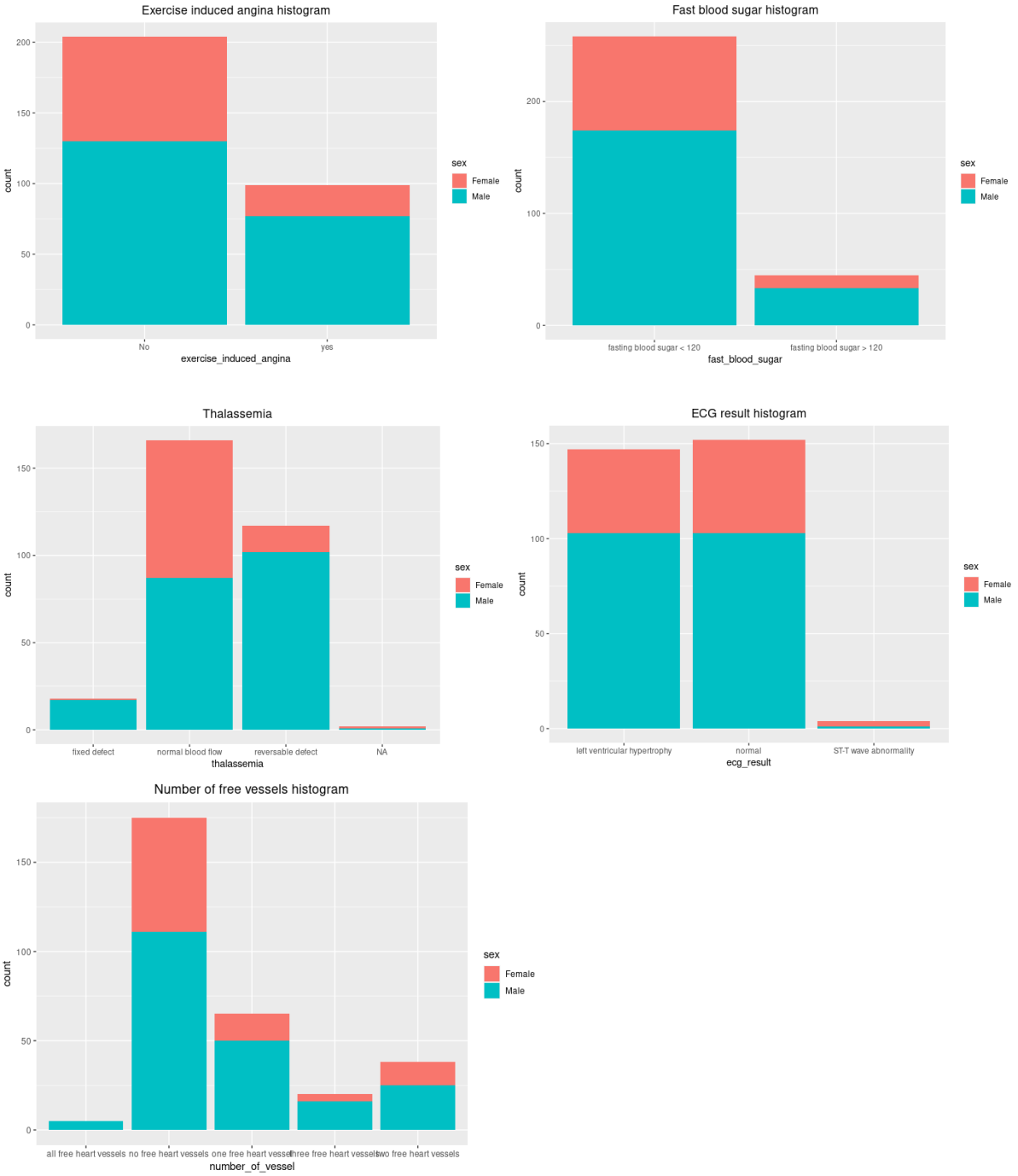
Grafici esplorativi

Per comprendere meglio il significato e la distribuzione dei regressori con cui abbiamo a che fare è utile visionare alcuni grafici quali istogrammi e densità, a tal proposito si è scelto di utilizzare la libreria ggplot2 in modo da poter realizzare suddetti grafici in relazione al sesso dei pazienti e avere così una maggiore compresione dei dati a nostra disposzione. Si è reso tuttavia necessario effettuare una preparazione dei dati a tale scopo che per motivi di brevità non riportiamo in questo documento.

Dal momento che il codice per realizzare i grafici di nostro interesse è del tutto analogo riportiamo solo alcuni sporadici esempi rimandando comunque ad una visione completa dello script R in allegato.

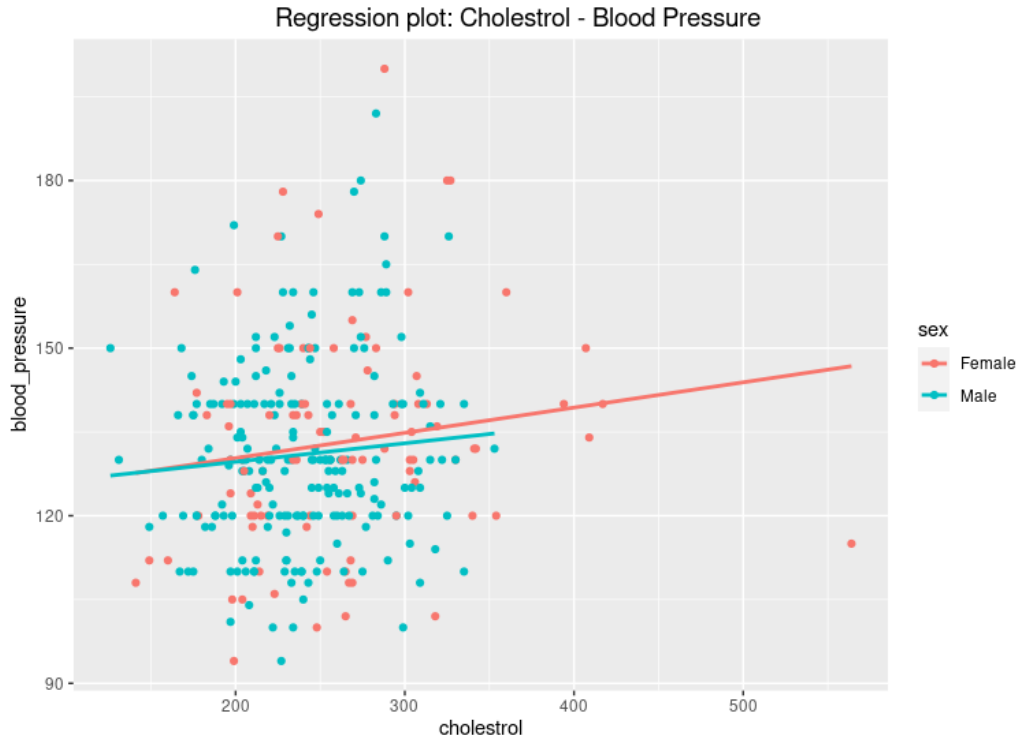
```
# Age histogram
ggplot(data = DataSetForGraph, aes(x = age, fill = sex)) +
  geom_histogram(col = "black") +
  ggtitle("Age  histogram") +
  theme(plot.title = element_text(hjust = 0.5))
```

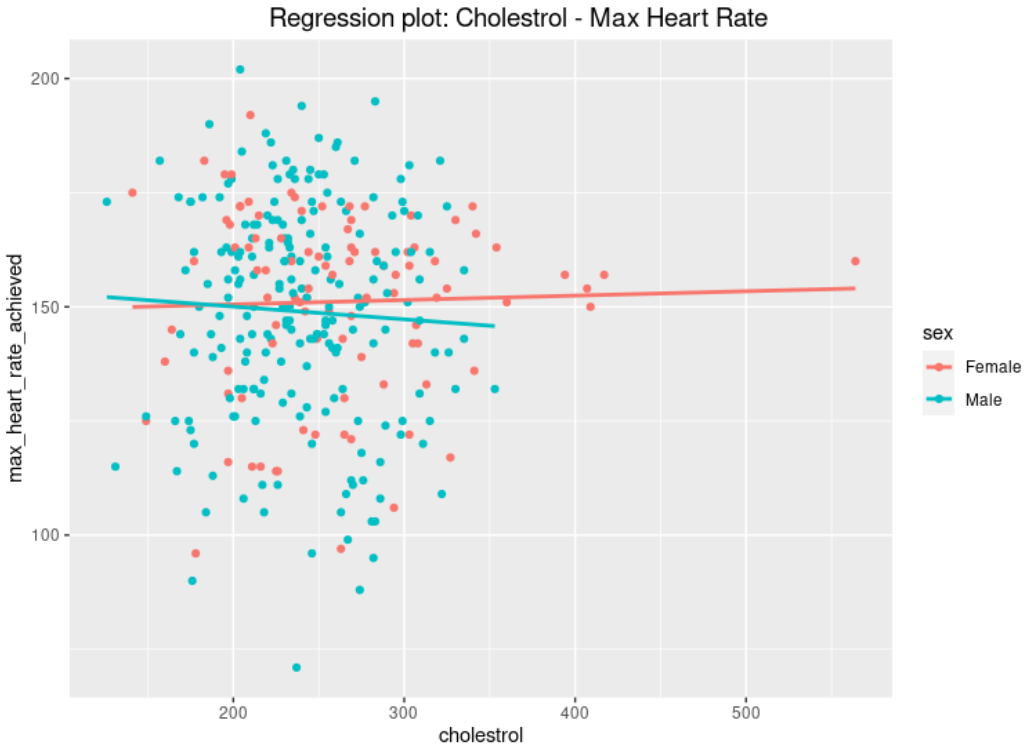




Questi istogrammi mettono in luce quale siano le differenze nei valori clinici dei pazienti in base al loro sesso oltre a mostrare quale dei due sessi è maggornamente afflitto, ad esempio, da Talassemia.

Osserviamo ora alcuni grafici in cui si riportano i valori di alcune coppie di regressori e la retta di regressione su tali valori al fine poter osservare per grandi linee quali relazioni intercorrono tra essi.

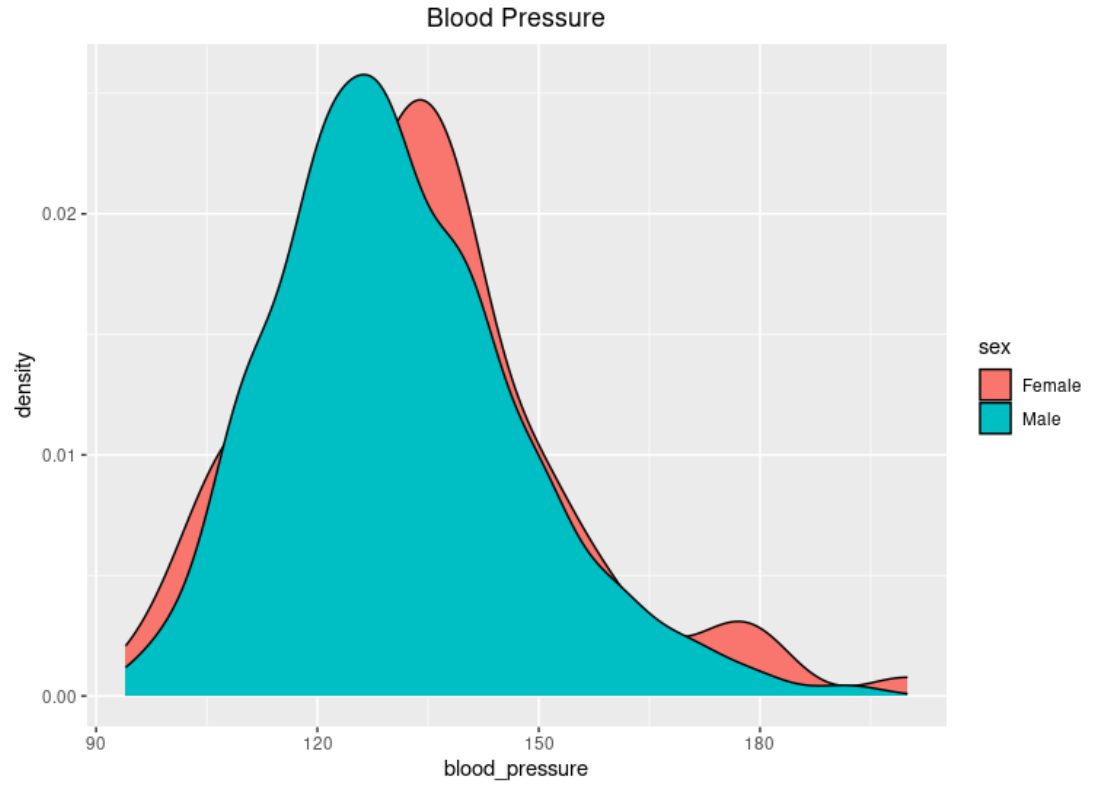

Regression plot: Cholestrol - Blood Pressure
blood_pressure
180
150
120
90
200
300
400
500
cholestrol
sex
Female
Male

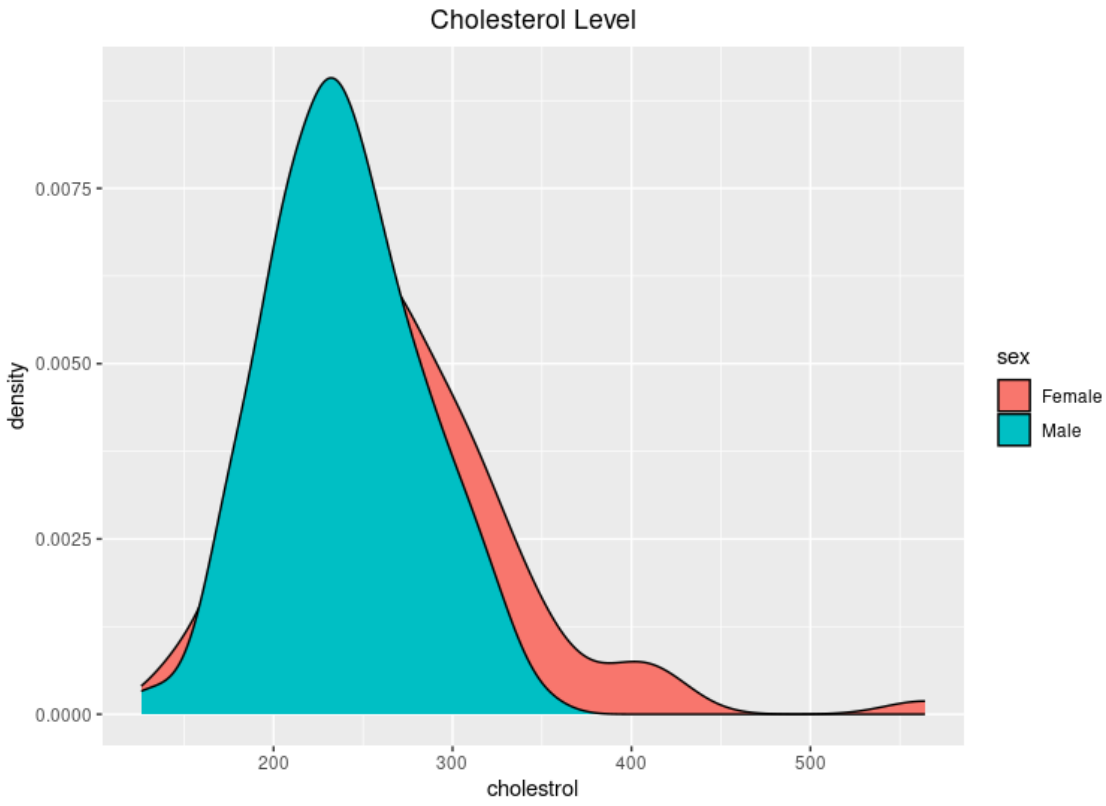

Regression plot: Cholestrol - Max Heart Rate
max_heart_rate_achieved
200
150
100
200
300
400
500
cholestrol
sex
Female
Male

Da questi tre grafici possiamo dedurre come vi sia una correlazione positiva tra il colesterolo e l'età per entrambi i sessi come anche tra colesterolo e pressione sanguigna. Questo risultato ci fornisce un indizio importante in quanto potremmo essere in presenza di multicollinearità. In seguito cercheremo di determinare proprio la presenza di quest'ultima.

Infine riportiamo anche la densità (ricostruita) della pressione sanguigna e del colesterolo al fine di comprendere bene come essi si distribuiscono nella nostra popolazione di esempio.

```
# Density plot of blood pressure
ggplot(DataSetForGraph, aes(x = blood_pressure, fill = sex)) +
  geom_density() +
  ggtitle("Blood Pressure") +
  theme(plot.title = element_text(hjust = 0.5))
```





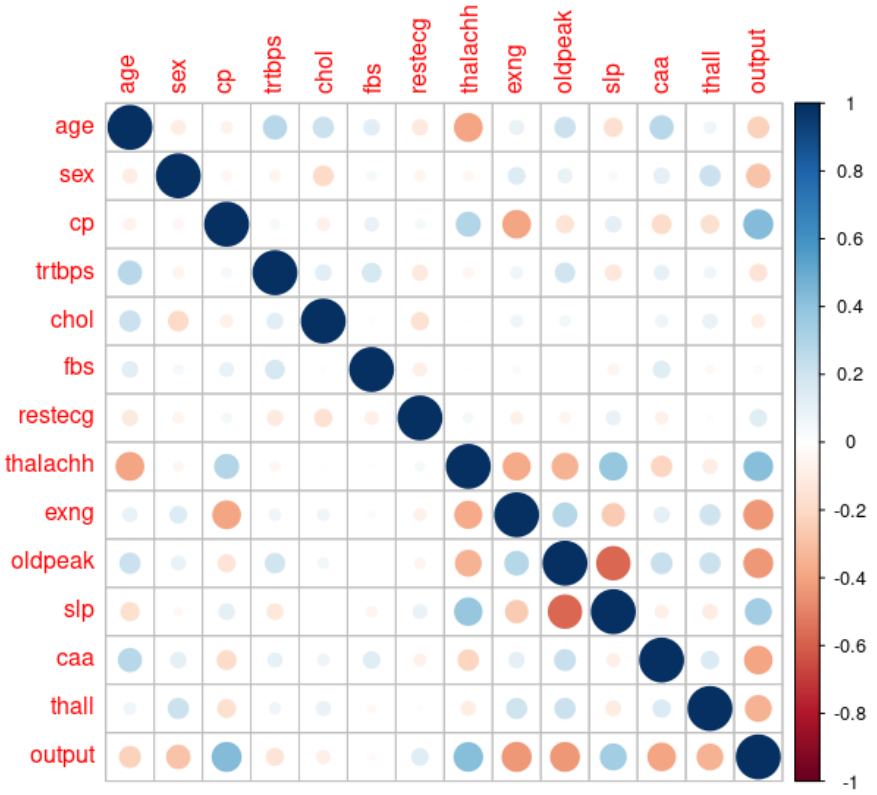
Dai precedenti grafici si evince come sia per la pressione sanguigna che per il colesterolo vi siano alcune differenze nella distribuzione dei valori per i due sessi. Nel caso della pressione sanguigna è possibile osservare proprio la presenza di due picchi separati che ci informano di due valori di pressione più

frequenti diversi nei due sessi, al contrario nel caso del colesterolo è possibile osservare una dilatazione maggiore della distribuzione sul sesso femminile, deduciamo quindi che le donne presentano una maggior varianza nei valori del colesterolo rispetto agli uomini.

Multicollinearità e correlazione tra i regressori

Come anticipato nella sezione dei grafici esplorativi abbiamo osservato qualche indizio di possibile correlazione tra le regressioni. Per comprendere se ci troviamo in una situazione di multicollinearità calcoliamo la matrice di correlazione dei regressioni e visualizziamola in modo da comprendere l'entità delle eventuali correlazioni.

```
DataSet.cor = cor(DataSet)
corrplot(DataSet.cor)
DataSet.cor
```



Osserviamo la presenza di una moderata correlazione tra varie coppie di regressori quindi possiamo concludere che siamo in una situazione di media multicollinearità, per questo motivo è necessario utilizzare le tecniche di regolarizzazione studiate in modo da ovviare al problema.

Analisi del modello con tutti i regressori

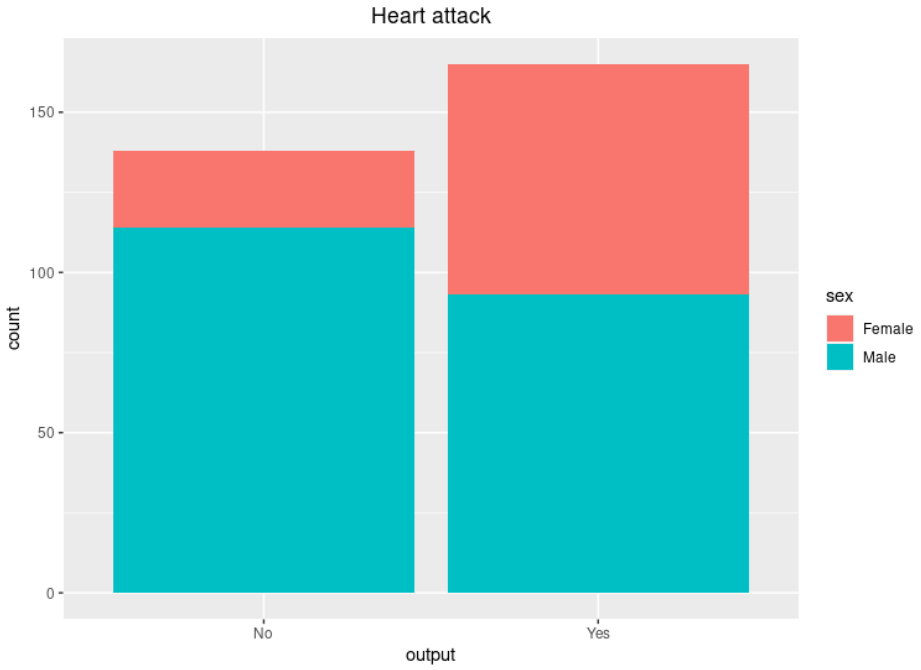
Stimiamo ora un primo modello lineare, con tutti i regressori senza la rimozione degli outliers.

```
attach(DataSet)
model <- lm(output~., data = DataSet)
summary(model)
```

```
Call:
lm(formula = output ~ ., data = DataSet)

Residuals:
     Min       1Q   Median       3Q      Max
-0.94748 -0.21270  0.06608  0.25022  0.93509

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.8288987  0.2929344   2.830 0.004987 **
age         -0.0008204  0.0026962  -0.304 0.761129
sex         -0.1959956  0.0471429  -4.157 4.24e-05 ***
cp           0.1127034  0.0223816   5.036 8.40e-07 ***
trtbps      -0.0019910  0.0012573  -1.583 0.114407
chol        -0.0003535  0.0004217  -0.838 0.402545
fbs          0.0173736  0.0596669   0.291 0.771125
restecg      0.0498480  0.0399228   1.249 0.212819
thalachh     0.0030193  0.0011304   2.671 0.007988 **
exng        -0.1440459  0.0513689  -2.804 0.005387 **
oldpeak     -0.0587887  0.0229269  -2.564 0.010847 *
slp          0.0789788  0.0423896   1.863 0.063453 .
caa         -0.1006022  0.0218565  -4.603 6.25e-06 ***
thall       -0.1190392  0.0356550  -3.339 0.000952 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3542 on 289 degrees of freedom
Multiple R-squared:  0.5175,    Adjusted R-squared:  0.4958
F-statistic: 23.85 on 13 and 289 DF,  p-value: < 2.2e-16
```

Possiamo osservare come il modello lineare con tutti i regressori abbia un basso adattamento ai dati (R^2 pari a a 0.4959) e numerosi regressori statisticamente poco significativi alla spiegazione del fenomeno. Inoltre, notiamo la presenza di uno standard error abbastanza alto pari a 0.3542.

Per comprendere meglio com’è distribuita la variabile dipendente, ovvero se il paziente ha avuto o meno un attacco di cuore si riporta anche l’istogramma della variabile dipendente (binaria).



Applicazione delle tecniche di regolarizzazione e cross validation

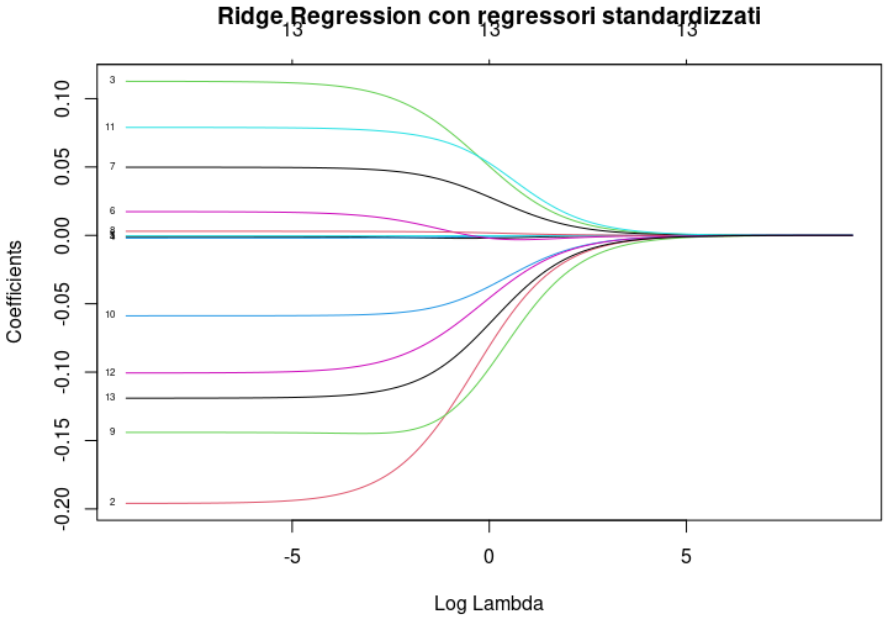
Al fine di stimare il miglior modello per la previsione applichiamo le tecniche di regolarizzazione per ovviare al problema della multicollinearità e le tecniche di cross validation al fine di determinare il miglior valore di lambda che corrisponde al minor MSE di test. Le tecniche di regolarizzazione usate sono Ridge Regression, LASSO ed Elsatic-Net mentre come tecnica di cross validation usiamo solo la K-fold con valore di K pari a 10.

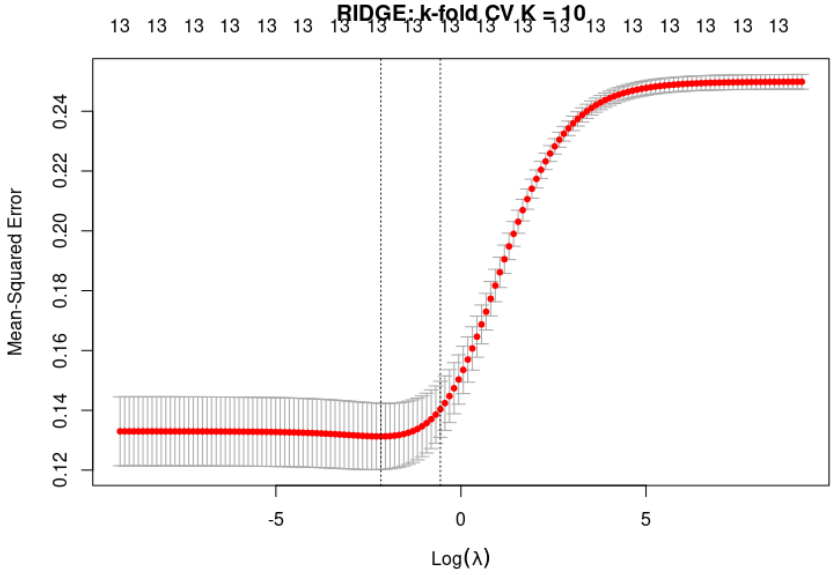
Come prima cosa definiamo una griglia di valori di lambda da usare e separiamo il vettore della variabile dipendente dalla matrice dei regressori.

```
# discretizzazione di lambda in una sequenza decrescente di valori
qq <- seq(4, -4, length = 150)
griglia = 10^qq

# divisione tra matrice dei regressori e vettore variabile dipendente
n <- nrow(DataSet)
names(DataSet)
xx <- DataSet[, -14]
x <- as.matrix(xx)
dim(x)
names(x)
y <- DataSet$output
```

Applichiamo la Ridge Regression con la K-fold cross validation e generiamo i grafici del regularization-path e dell'andamento dell'MSE di test al variare di lambda.

```r
# Di default glmnet() standardizza i regressori
ridgeAllLambda = glmnet (x, y, alpha = 0, lambda = griglia)
coef(ridgeAllLambda)

# Grafico andamento dei regressori rispetto al lambda
plot(ridgeAllLambda, main = "Ridge Regression con regressori standardizzati", xvar = "lambda", label = TRUE)

# Applichiamo la K-Fold cross-validation con K = 10 (default)
ridgeKfold10 = cv.glmnet(x, y, lambda = griglia, alpha = 0)
plot(ridgeKfold10, main = "RIDGE: k-fold CV K = 10")

# Estraiamo il lambda minimo a cui corrisponde la minore media deli MSE calcolati sul test-set
ridgeBestLambda <- ridgeKfold10$lambda.min
ridgeBestLambda

# Utilizziamo il lambda migliore per stimare il modello finale
ridgeModBestLambda = glmnet(x, y, alpha = 0, lambda = ridgeBestLambda)
coef(ridgeModBestLambda)[, 1]
```





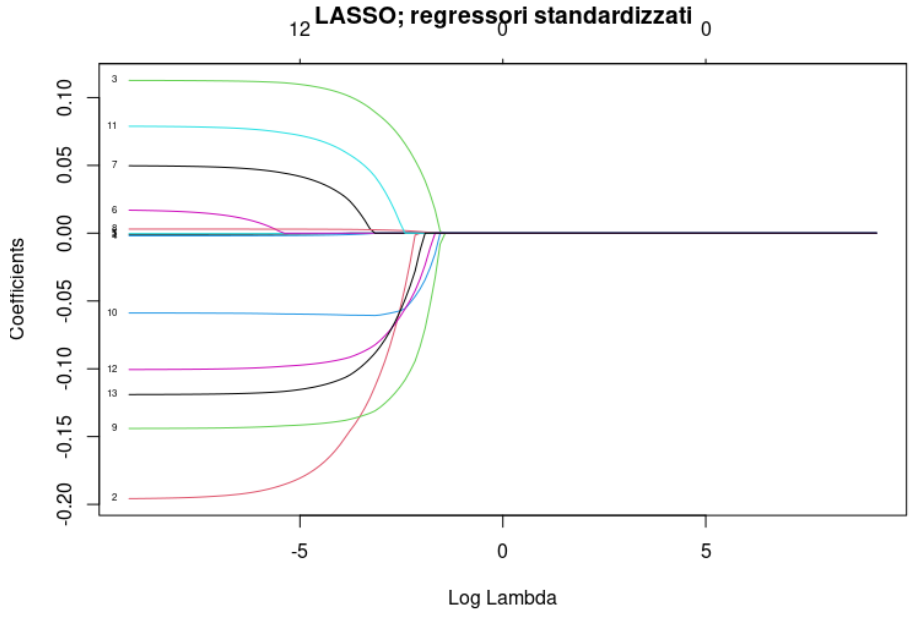
Dal regularization-path è possibile osservare come all'aumentare di lambda la Ridge Regression spinga tutti i coefficienti di regressione a zero senza effettuare alcuna selezione su di essi. i coefficienti di regressione ottenuti stimando il modello con il lambda corrispondente al minore standard error sono i seguenti.

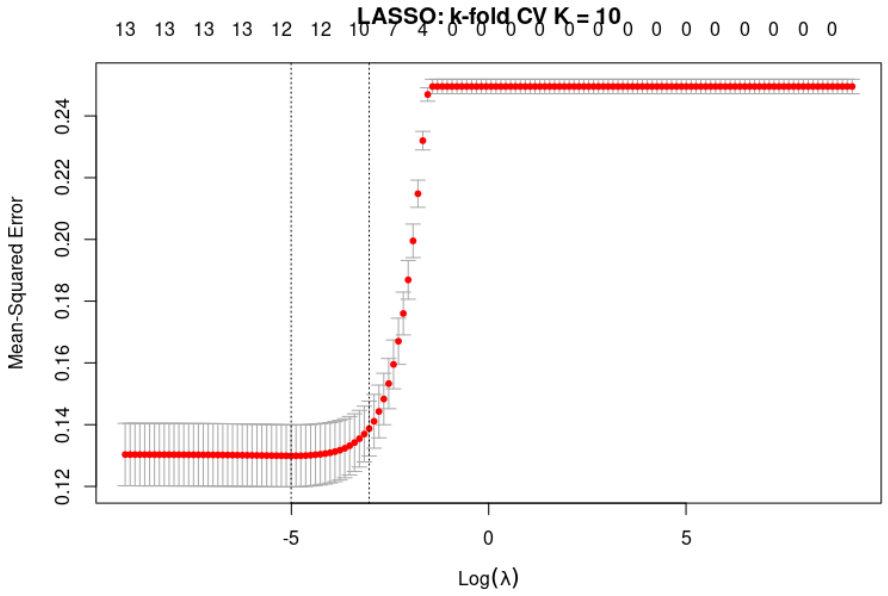
```
> coef(ridgeModBestLambda)[, 1]
  (Intercept)           age           sex            cp        trtbps          chol           fbs
0.8078875984 -0.0015562241 -0.1661713066  0.0967457032 -0.0016065749 -0.0002873219  0.0103582631
     restecg      thalachh          exng       oldpeak           slp           caa         thall
0.0461050784  0.0027738182 -0.1431794137 -0.0560984991  0.0746522584 -0.0869345914 -0.1095590308
```

Possiamo osservare come nessuno dei regressori sia stato posto a zero anche se alcuni, come trtbps, presentano valori prossimi allo zero.

Passiamo ora all'applicazione della tecnica di regolarizzazione LASSO con la K-fold cross validation e come fatto per la Ridge Regression generiamo i grafici del regularization-path e dell'MSE di test.

```
# Usiamo la funzione glmnet() ponendo alpha = 1 per la regressione LASSO con l'intero dataset
LassoAllLambda = glmnet (x, y, alpha = 1, lambda = griglia)
# È possibile osservare come la tecnica LASSO, anche per valori piccoli di lambda, pone alcuni regressori a zero
plot(LassoAllLambda, main = "LASSO; regressori standardizzati", xvar = "lambda", label = TRUE)
coef(LassoAllLambda)[, 1]
dim(coef(LassoAllLambda))

# Applichiamo la K-Fold cross-validation con K = 10 (default)
lassoKfold10 = cv.glmnet(x, y, lambda = griglia, alpha = 1)
plot(lassoKfold10, main = "LASSO: k-fold CV K = 10")
lassoBestLambda <- lassoKfold10$lambda.min
# Il lambda minimo a cui corrisponde il più piccolo valore del MSE test
lassoBestLambda

# Stima del modello di regressione LASSO con il lambda.min
lassoModBestLambda = glmnet(x, y, alpha = 1, lambda = lassoBestLambda)
coef(lassoModBestLambda)[, 1]
```





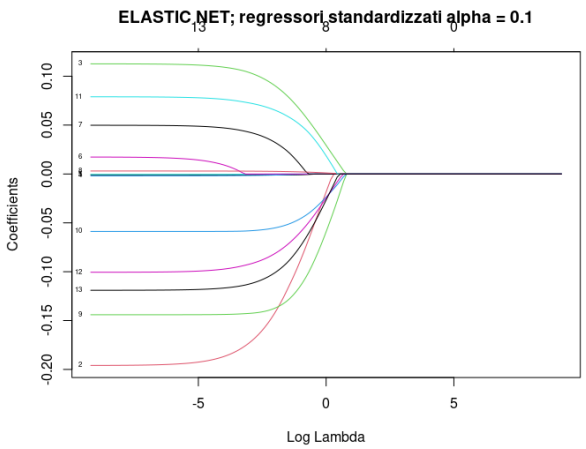
A differenza della Ridge Regression con la tecnica LASSO è evidente come essa applichi una selezione ai coefficienti di regressione anche per valori piccoli di la lambda. Infatti, osservando i coefficienti di regressione del modello stimato con il valore di lambda trovato dall'applicazione della K-fold cross validation è possibile osservare proprio come alcuni regressori presentino valori prossimi allo zero ed uno in particolare, fbs, sia stato azzerato.

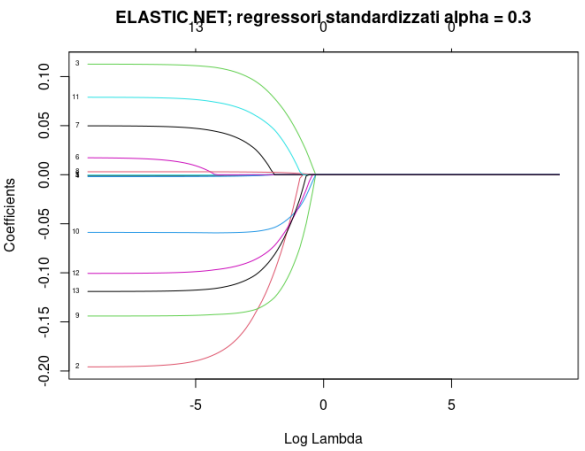
```
> coef(lassoModBestLambda)[, 1]
  (Intercept)           age           sex            cp        trtbps          chol           fbs
0.7529111976 -0.0005520523 -0.1807353973  0.1097970165 -0.0016746926 -0.0002453918  0.0000000000
     restecg      thalachh          exng       oldpeak           slp           caa         thall
0.0419392032  0.0029586311 -0.1414859779 -0.0596648455  0.0721134749 -0.0973305615 -0.1152977198
```

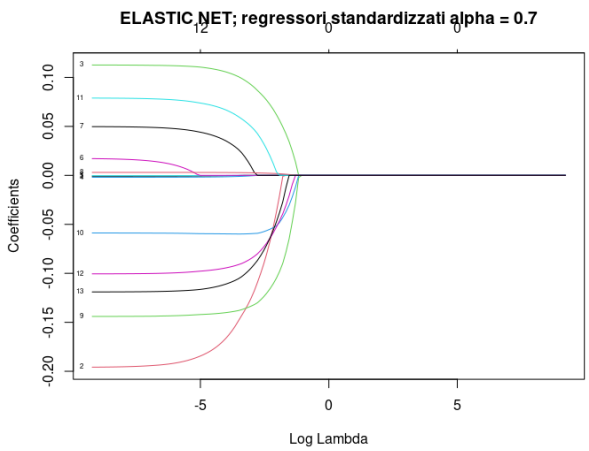
Infine, applichiamo la tecnica di regolarizzazione Elastic-Net con diversi valori di alfa quali 0.1, 0.3, 0.5, 0.7 e 0.9 e, come nei casi precedenti, generiamo i grafici del regularization path, dell'MSE di test e stampiamo a video i coefficienti di regressione ottenuti stimando il modello con il lambda corrispondente al miglior MSE di test trovato dalla K-fold cross validation.

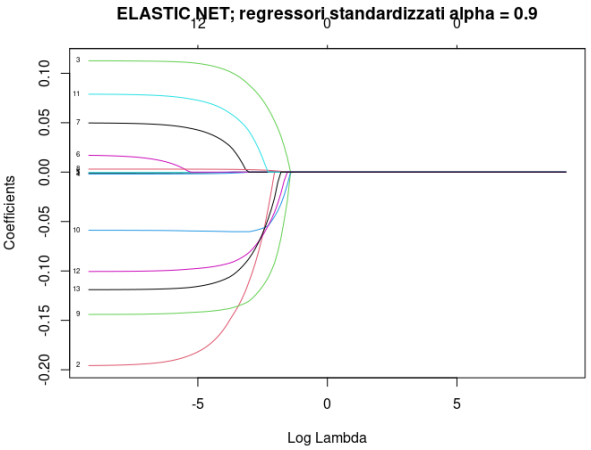
Per prima cosa generiamo tutti i grafici del regularization path usando la funzione glmnet con diversi valori di alfa.

```
ElasticNetAllLambda01 <- glmnet(x, y, lambda = griglia, alpha = .1)
plot(ElasticNetAllLambda01, main = "ELASTIC NET; regressori standardizzati alpha = 0.1", xvar = "lambda", label = TRUE)

ElasticNetAllLambda03 <- glmnet(x, y, lambda = griglia, alpha = .3)
plot(ElasticNetAllLambda03, main = "ELASTIC NET; regressori standardizzati alpha = 0.3", xvar = "lambda", label = TRUE)

ElasticNetAllLambda07 <- glmnet(x, y, lambda = griglia, alpha = .7)
plot(ElasticNetAllLambda07, main = "ELASTIC NET; regressori standardizzati alpha = 0.7", xvar = "lambda", label = TRUE)

ElasticNetAllLambda09 <- glmnet(x, y, lambda = griglia, alpha = .9)
plot(ElasticNetAllLambda09, main = "ELASTIC NET; regressori standardizzati alpha = 0.9", xvar = "lambda", label = TRUE)
```









Stimiamo ora i vari modelli con diverso alfa con il miglior valore di lambda (corrispondente all'MSE di test più basso) e osserviamo i coefficienti di regressione ottenuti.

Con alfa pari a 0.1 otteniamo il seguente modello.

```
# k-fold CROSS VALIDATION per ELASTIC NET
elasticNet01Kfold10 = cv.glmnet(x, y, lambda = griglia, alpha = 0.1)
plot(elasticNet01Kfold10, main = "Elastic Net alpha = 0.1: k-fold CV K = 10")
elasticNet01BestLambda <- elasticNet01Kfold10$lambda.min
elasticNet01BestLambda
elasticNet01ModBestLambda = glmnet(x, y, alpha = 0.1, lambda = elasticNet01BestLambda)
coef(elasticNet01ModBestLambda)[, 1]
```

```
> coef(elasticNet01ModBestLambda)[, 1]
  (Intercept)           age           sex            cp        trtbps          chol           fbs
0.7475909660 -0.0010374265 -0.1645561257  0.1002053154 -0.0014677441 -0.0002120123  0.0000000000
     restecg      thalachh          exng       oldpeak           slp           caa         thall
0.0401111597  0.0028031856 -0.1417714881 -0.0575401077  0.0707128724 -0.0895003171 -0.1097516131
```

Anche in questo caso osserviamo come alcuni regressori abbiano valori prossimi alla zero ma solo uno, fbs, abbia valore nullo.

Con alfa pari a 0.3 otteniamo il seguente modello.

```
# Con alpha = 0.3
elasticNet03Kfold10 = cv.glmnet(x, y, lambda = griglia, alpha = 0.3)
plot(elasticNet03Kfold10, main = "Elastic Net alpha = 0.3: k-fold CV K = 10")
elasticNet03BestLambda <- elasticNet03Kfold10$lambda.min
elasticNet03BestLambda
elasticNet03ModBestLambda = glmnet(x, y, alpha = 0.3, lambda = elasticNet03BestLambda)
coef(elasticNet03ModBestLambda)[, 1]
```

```
> coef(elasticNet03ModBestLambda)[, 1]
  (Intercept)           age           sex            cp        trtbps          chol           fbs
0.7507551357 -0.0006967587 -0.1759553373  0.1070329878 -0.0016080273 -0.0002330827  0.0000000000
     restecg      thalachh          exng       oldpeak           slp           caa         thall
0.0413286647  0.0029121747 -0.1418953131 -0.0591263099  0.0716496271 -0.0951221434 -0.1137800962
```

Con alfa pari a 0.5 otteniamo il seguente modello.

```
# Con alpha = 0.5
elasticNet05Kfold10 = cv.glmnet(x, y, lambda = griglia, alpha = 0.5)
plot(elasticNet05Kfold10, main = "Elastic Net alpha = 0.5: k-fold CV K = 10")
elasticNet05BestLambda <- elasticNet05Kfold10$lambda.min
elasticNet05BestLambda
elasticNet05ModBestLambda = glmnet(x, y, alpha = 0.5, lambda = elasticNet05BestLambda)
coef(elasticNet05ModBestLambda)[, 1]
```

```
> coef(elasticNet05ModBestLambda)[, 1]
  (Intercept)           age           sex            cp        trtbps          chol           fbs
0.6888330376 -0.0004661816 -0.1639007509  0.1042612698 -0.0013823958 -0.0001375913  0.0000000000
     restecg      thalachh          exng       oldpeak           slp           caa         thall
0.0347475700  0.0028535322 -0.1401398402 -0.0594092531  0.0665182358 -0.0933435505 -0.1101183605
```

Con alfa pari a 0.7 otteniamo il seguente modello.

```
# Con alpha = 0.7
elasticNet07Kfold10 = cv.glmnet(x, y, lambda = griglia, alpha = 0.7)
plot(elasticNet07Kfold10, main = "Elastic Net alpha = 0.7: k-fold CV K = 10")
elasticNet07BestLambda <- elasticNet07Kfold10$lambda.min
elasticNet07BestLambda
elasticNet07ModBestLambda = glmnet(x, y, alpha = 0.7, lambda = elasticNet07BestLambda)
coef(elasticNet07ModBestLambda)[, 1]
```

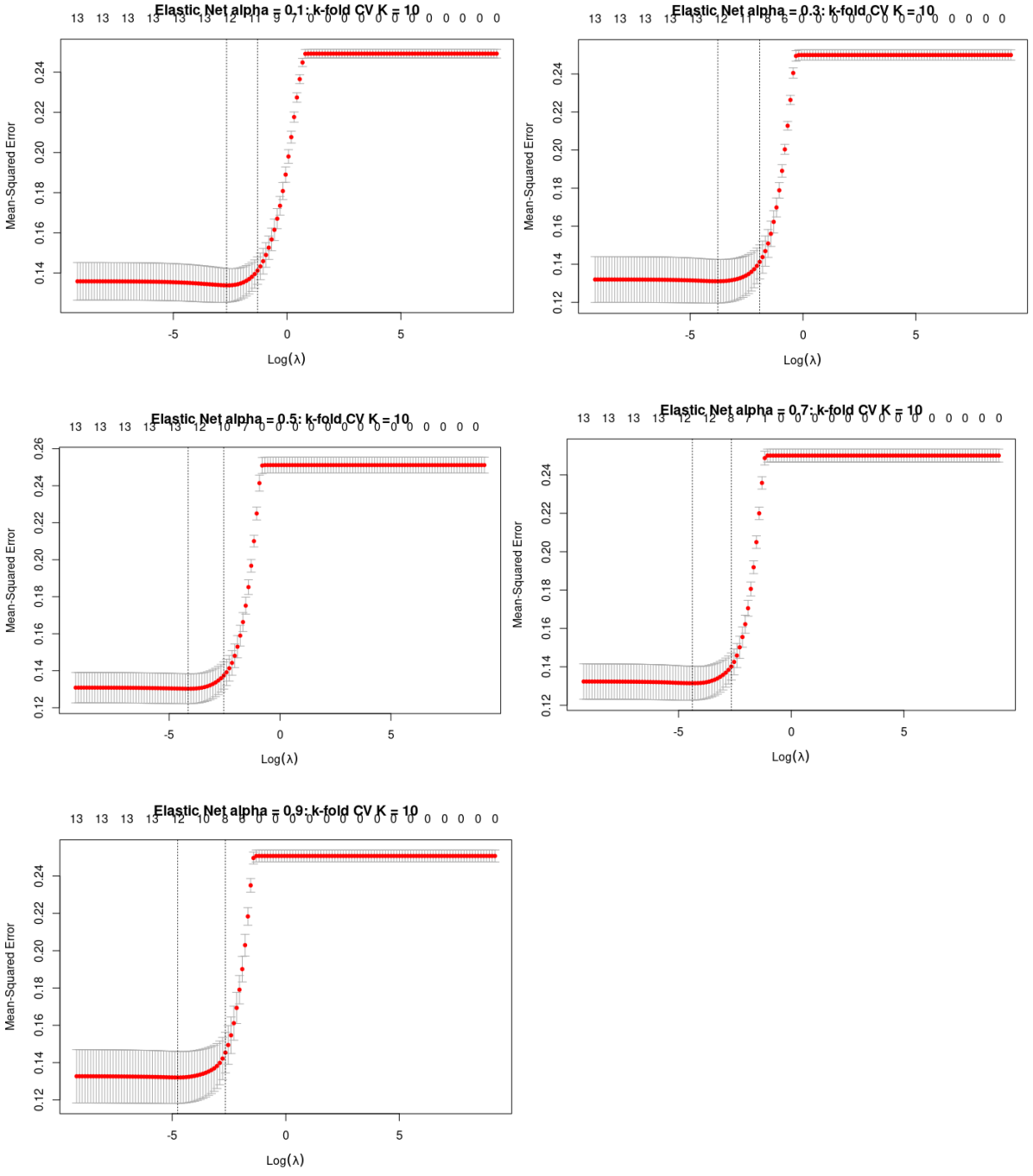
```
> coef(elasticNet07ModBestLambda)[, 1]
  (Intercept)           age           sex            cp        trtbps          chol           fbs
0.7324742381 -0.0005229015 -0.1754145290  0.1080819101 -0.0015819774 -0.0002108438  0.0000000000
     restecg      thalachh          exng       oldpeak           slp           caa         thall
0.0396647201  0.0029261104 -0.1411057796 -0.0596251998  0.0703045618 -0.0961124912 -0.1136944409
```

Con alfa pari a 0.9 otteniamo il seguente modello.

```
# Con alpha = 0.9
elasticNet09Kfold10 = cv.glmnet(x, y, lambda = griglia, alpha = 0.9)
plot(elasticNet09Kfold10, main = "Elastic Net alpha = 0.9: k-fold CV K = 10")
elasticNet09BestLambda <- elasticNet09Kfold10$lambda.min
elasticNet09BestLambda
elasticNet09ModBestLambda = glmnet(x, y, alpha = 0.9, lambda = elasticNet09BestLambda)
coef(elasticNet09ModBestLambda)[, 1]
```

```
> coef(elasticNet09ModBestLambda)[, 1]
  (Intercept)           age           sex            cp        trtbps          chol           fbs
0.7424244396 -0.0005269476 -0.1782895354  0.1090936458 -0.0016310523 -0.0002282519  0.0000000000
     restecg      thalachh          exng       oldpeak           slp           caa         thall
0.0408000923  0.0029449628 -0.1412606059 -0.0596845447  0.0711995905 -0.0968487298 -0.1145684184
```

Per avere una maggiore comprensione di quanto è stato effettuato si riportano anche i grafici dell'andamento dell'MSE di test per ciascuno dei modelli presi in esame.



Confronto fra i modelli ottenuti e selezione del modello migliore per la previsione

Confrontiamo ora i modelli ottenuti applicando le tecniche di regolarizzazione e selezioniamo il modello migliore ai fini della previsione. Per scegliere il modello migliore ci basiamo sull'MSE di test prendendo quello che corrisponde ad un valore più basso

```
# SCELTA del modello sulla base del minor MSE test
# Per confrontare i modelli bisogna estrarre MSE più piccolo a cui corrisponde il lambda minimo
mseMinRR <- ridgeKfold10$cvm[ridgeKfold10$lambda == ridgeKfold10$lambda.min]
mseMinLASSO <- lassoKfold10$cvm[lassoKfold10$lambda == lassoKfold10$lambda.min]
mseMinEN01 <- elasticNet01Kfold10$cvm[elasticNet01Kfold10$lambda == elasticNet01Kfold10$lambda.min]
mseMinEN03 <- elasticNet03Kfold10$cvm[elasticNet03Kfold10$lambda == elasticNet03Kfold10$lambda.min]
mseMinEN05 <- elasticNet05Kfold10$cvm[elasticNet05Kfold10$lambda == elasticNet05Kfold10$lambda.min]
mseMinEN07 <- elasticNet07Kfold10$cvm[elasticNet07Kfold10$lambda == elasticNet07Kfold10$lambda.min]
mseMinEN09 <- elasticNet09Kfold10$cvm[elasticNet09Kfold10$lambda == elasticNet09Kfold10$lambda.min]
vettMSE <- cbind(mseMinLASSO, mseMinRR, mseMinEN01, mseMinEN03, mseMinEN05, mseMinEN07, mseMinEN09)
vettMSE
minMSE <- min(vettMSE)
minMSE
```

```
> vettMSE
     mseMinLASSO  mseMinRR mseMinEN01 mseMinEN03 mseMinEN05 mseMinEN07 mseMinEN09
[1,]   0.1319808 0.1293068  0.1295624  0.1316002   0.133604   0.130485  0.1322443
> minMSE <- min(vettMSE)
> minMSE
[1] 0.1293068
```

Come si evince dai risultati ottenuti il modello con l'MSE più basso risulta essere quello ottenuto con la tecnica di regolarizzazione Ridge Regression, dunque useremo questo modello per effettuare delle previsioni su alcuni pazienti fittizi. Nello specifico consideriamo tre pazienti maschi di età crescente con diverse patologie e quadri clinici.

```r
# Person 1
# Age: 20
# Sex: Male (1)
# Chest pain: Asymptomatic (0)
# Blood pressure: 120
# Cholesterol: 80
# Fasting blood sugar: <120 (0)
# ECG result: Normal (1)
# Maximum heart rate achieved: 130
# Exercise induced angina: No (0)
# ST depression induced by exercise relative to rest: No (0)
# Slope of the peak exercise ST segment: Downsloping (0)
# Number of major vessels: All free (4)
# Thalassemia: Normal blood flow (2)
male20data <- c(20, 1, 0, 120, 80, 0, 1, 130, 0, 0, 0, 4, 2)
male20output <- predict(bestPredictionModel, male20data, type = "response")
male20output
```

```
> male20data <- c(20, 1, 0, 120, 80, 0, 1, 130, 0, 0, 0, 4, 2)
> male20output <- predict(bestPredictionModel, male20data, type = "response")
> male20output
           s0
[1,] 0.1497164
```

In questo primo caso, come ci aspettiamo, otteniamo un valore basso della variabile dipendente dato il quadro clinico praticamente perfetto e la giovane età del paziente.

```r
# Person 2
# Age: 50
# Sex: Male (1)
# Chest pain: Atypical angina (1)
# Blood pressure: 170
# Cholesterol: 200
# Fasting blood sugar: >120 (1)
# ECG result: Having ST-T wave abnormality (2)
# Maximum heart rate achieved: 180
# Exercise induced angina: Yes (1)
# ST depression induced by exercise relative to rest: No (0)
# Slope of the peak exercise ST segment: Upsloping (2)
# Number of major vessels: Two free (2)
# Thalassemia: Fixed defect (1)
male50data <- c(50, 1, 1, 170, 200, 1, 2, 180, 1, 0, 2, 1, 1)
male50output <- predict(bestPredictionModel, male50data, type = "response")
male50output
```

```
> male50data <- c(50, 1, 1, 170, 200, 1, 2, 180, 1, 0, 2, 1, 1)
> male50output <- predict(bestPredictionModel, male50data, type = "response")
> male50output
           s0
[1,] 0.7307008
```

Nel secondo caso la situazione si fa più critica dal momento che siamo in presenza di un paziente di mezza età con numerosi valori clinici elevati, quali pressione e colesterolo ma anche con due vasi cardiaci otturati e Talassemia.

In questo caso ci aspettiamo un valore predetto dal modello molto più alto del precedente paziente.

```
# Person 3
# Age: 70
# Sex: Male (1)
# Chest pain: Typical angina (3)
# Blood pressure: 170
# Cholesterol: 230
# Fasting blood sugar: >120 (1)
# ECG result: Having ST-T wave abnormality (2)
# Maximum heart rate achieved: 160
# Exercise induced angina: Yes (1)
# ST depression induced by exercise relative to rest: No (0)
# Slope of the peak exercise ST segment: Upsloping (2)
# Number of major vessels: No free vessels (0)
# Thalassemia: Fixed defect (1)
male70data <- c(70, 1, 3, 170, 230, 1, 2, 160, 1, 0, 2, 0, 1)
male70output <-predict(bestPredictionModel, male70data, type = "response")
male70output
```

```
> male70data <- c(70, 1, 3, 170, 230, 1, 2, 160, 1, 0, 2, 0, 1)
> male70output <-predict(bestPredictionModel, male70data, type = "response")
> male70output
           s0
[1,] 0.9702227
```

In quest'ultimo caso la situazione è molto più critica data l'elevata età del paziente e la presenza di un quadro clinico decisamente critico, infatti esso presenta ostruzioni in tutti i vasi cardiaci, un elevato colesterolo e altri parametri decisamente inadeguati. Come atteso, il modello fornisce una previsione del tutto allarmante con un elevato valore della variabile dipendente (predetta) che sta ad indicarci un alto rischio di attacco di cuore.

In conclusione, il valore predetto dal modello, inserendo opportunamente i dati, può essere visto come la probabilità di un paziente di avere un attacco di cuore a partire da alcuni semplici valori clinici. Seppur banale, questo esempio ci fornisce un'importante indicazione riguardo l'utilità dei metodi di apprendimento statistico nella vita di tutti i giorni.

Appendice 1: addestramento modello predittivo Random Forest Classifier

Ora, come approfondimento addestriamo un modello di classificazione noto nell'apprendimento automatico: il Random Forest Classifier.

Siccome i regressori presentano scale differenti è necessario effettuare la standardizzazione sui dati. Lo facciamo sui regressori che però non presentano una scala limitata (valori binari/ternari/quaternari) in quanto può alterare di molto i valori originari, la distribuzione può essere sbilanciata, e la relazione tra i valori di ciascuna variabile potrebbe essere importante per la previsione.

```
#regressori da standardizzare
x_toscale <- x[, c("age", "trtbps", "chol", "thalachh", "oldpeak")]
#regressori da non standardizzare
dropped <- x[,  c("sex", "cp", "fbs", "restecg", "exng", "slp", "caa", "thall")]

head(x_toscale)

scaled_regressors <- scale(x_toscale) #standardizziamo

head(scaled_regressors)

cleaned_regressors <- cbind(scaled_regressors, dropped, label = y) #dataset completo standardizzato

head(cleaned_regressors)
```

```
> head(x_toscale)
     age trtbps chol thalachh oldpeak
[1,]  63    145  233      150     2.3
[2,]  37    130  250      187     3.5
[3,]  41    130  204      172     1.4
[4,]  56    120  236      178     0.8
[5,]  57    120  354      163     0.6
[6,]  57    140  192      148     0.4
```

```
> head(scaled_regressors)
            age      trtbps        chol    thalachh    oldpeak
[1,]  0.9506240  0.76269408 -0.25591036  0.01541728  1.0855423
[2,] -1.9121497 -0.09258463  0.07208025  1.63077374  2.1190672
[3,] -1.4717230 -0.09258463 -0.81542377  0.97589950  0.3103986
[4,]  0.1798773 -0.66277043 -0.19802967  1.23784920 -0.2063639
[5,]  0.2899839 -0.66277043  2.07861109  0.58297496 -0.3786180
[6,]  0.2899839  0.47760118 -1.04694656 -0.07189928 -0.5508722
```

```
> head(cleaned_regressors)
            age      trtbps        chol    thalachh    oldpeak sex cp fbs restecg exng slp caa thall label
[1,]  0.9506240  0.76269408 -0.25591036  0.01541728  1.0855423   1  3   1       0    0   0   0     1     1
[2,] -1.9121497 -0.09258463  0.07208025  1.63077374  2.1190672   1  2   0       1    0   0   0     2     1
[3,] -1.4717230 -0.09258463 -0.81542377  0.97589950  0.3103986   0  1   0       0    0   2   0     2     1
[4,]  0.1798773 -0.66277043 -0.19802967  1.23784920 -0.2063639   1  1   0       1    0   2   0     2     1
[5,]  0.2899839 -0.66277043  2.07861109  0.58297496 -0.3786180   0  0   0       1    1   2   0     2     1
[6,]  0.2899839  0.47760118 -1.04694656 -0.07189928 -0.5508722   1  0   0       1    0   1   0     1     1
```

Ora ai fini dell'addestramento e dell'analisi della performance del modello dividiamo il dataset in training set e test set, con una rispettiva suddivisione pari al 70% e 30%.

```
#divisione in training set e test set
index <- sample(1:nrow(cleaned_regressors), 0.7*nrow(cleaned_regressors))
training_set <- cleaned_regressors[index, ]
y_training <- training_set[, "label"] #var dipendente training
training_set <- training_set[,-14]
test_set <- cleaned_regressors[-index, ]
y_test <- test_set[, "label"] #var dipendente test
test_set <- test_set[, -14] #eliminiamo la variabile dipendente dal test set
```

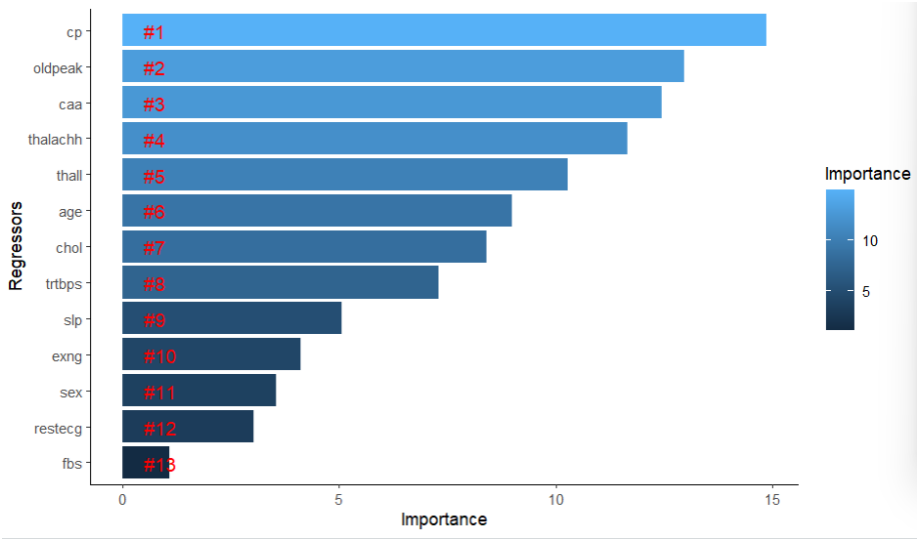
```
> dim(training_set)
[1] 212  13
> dim(test_set)
[1] 91 13
```

Addestriamo il modello

```
rf_model = randomForest(as.factor(y_training)~. , data = training_set, ntree =450, importance = TRUE)
```

Adesso effettuiamo, prima di misurare le prestazioni del nostro modello, un'analisi su quanto sia importante ciascun regressore ai fini della predizione.

```
importance <- importance(rf_model)

varImportance <- data.frame(Variables = row.names(importance),
                            Importance = round(importance[ ,'MeanDecreaseGini'],2))

rankImportance <- varImportance %>%
  mutate(Rank = paste0('#',dense_rank(desc(Importance))))

# Visualizziamo il tutto con ggplot
ggplot(rankImportance, aes(x = reorder(Variables, Importance),
                           y = Importance, fill = Importance)) +
  geom_bar(stat='identity') +
  geom_text(aes(x = Variables, y = 0.5, label = Rank),
            hjust=0, vjust=0.55, size = 4, colour = 'red') +
  labs(x = 'Variables') +
  coord_flip() +
  theme_classic()
```



Analizziamo ora la matrice di confusione:

```
predictions <- predict(rf_model, newdata = test_set)

cm2<-confusionMatrix(predictions, as.factor(y_test))
cm2
```

```
          Reference
Prediction  0  1
         0 30  2
         1 11 48
```

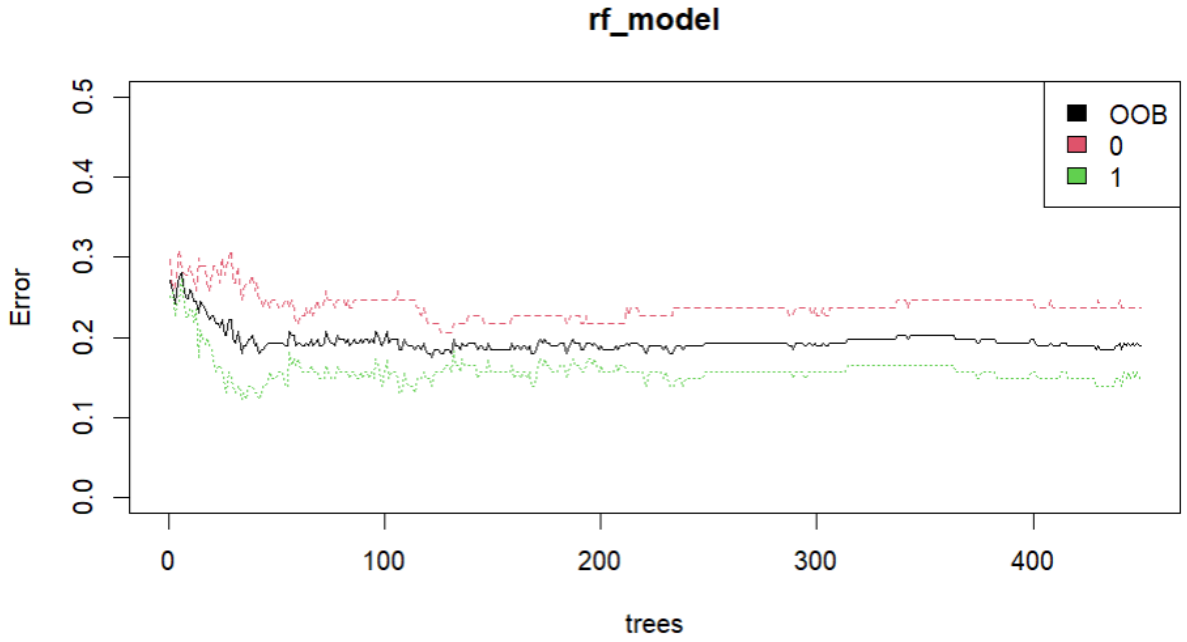
Notiamo che il nostro modello predice in modo corretto che il paziente non avrà l'attacco di cuore (0) 30 volte su 41, mentre che avrà un attacco di cuore (1) 48 volte su 50.

Quindi quando predice che un paziente avrà un attacco di cuore molto probabilmente avrà ragione! A differenza di quando predice che il paziente non l'avrà. E questo non è il massimo, in quanto sarebbe meglio il contrario.

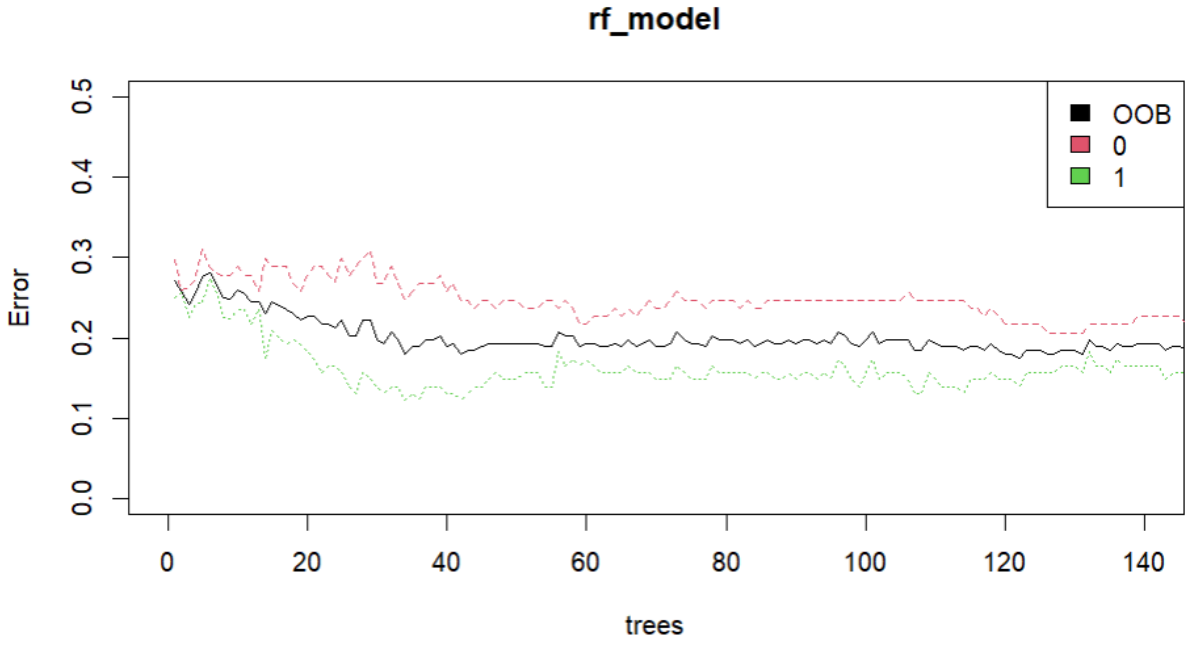
```
               Accuracy : 0.8571
                 95% CI : (0.7681, 0.9217)
    No Information Rate : 0.5495
    P-Value [Acc > NIR] : 3.897e-10

                  Kappa : 0.7056

 Mcnemar's Test P-Value : 0.0265

            Sensitivity : 0.7317
            Specificity : 0.9600
         Pos Pred Value : 0.9375
         Neg Pred Value : 0.8136
             Prevalence : 0.4505
         Detection Rate : 0.3297
   Detection Prevalence : 0.3516
      Balanced Accuracy : 0.8459
```

Andiamo a visualizzare ora il grafico che rappresenta la curva di errore, mostrando quindi la percentuale di errori di classificazione commessi dal modello in base al numero di alberi nella foresta.

numero alberi compreso tra 0 e 450:



zoommiamo su numero di alberi compreso tra 0 e 140:



Appendice 2: applicazione della regressione Beta

Descrizione del dataset

Al fine di mostrare un piccolo esempio di stima di un modello con la regressione Beta usiamo un dataset contenente informazioni su 400 studenti che applicano per entrare al college. Le informazioni contenute sono, ad esempio, il rating dell'università, il punteggio all'esame TOEFL dello studente e la probabilità che lo studente venga ammesso in quella università.

Il dataset contiene 9 features descritte nel seguente elenco puntato:

- **Serial.No**: Numero della riga
- **GRE.Score**: Rating dell'università stilato dal programma GRE (prevede un massimo di 340 punti)
- **TOEFL.Score**: Punteggio all'esame di inglese TOEFL sostenuto dallo studente (prevede un massimo di 120 punti)
- **University.rating**: Rating dell'università (valori interi da 1 a 5)
- **SOP**: Punteggio assegnato alla lettera di presentazione (valori continui da 0 a 5)
- **LOR**: Punteggio assegnato alla lettera di raccomandazione (valori continui da 0 a 5)
- **CGPA**: Indicatore delle performance dello studente nella sua carriera scolastica (valori continui da 0 a 10)
- **Research**: Indica se l'università effettua anche ricerca (1 = SI, 0 = NO)
- **Chance.of.Admit**: Probabilità dello studente di essere accettato in quell'università.

Il nostro obiettivo è quello di definire un modello con la regressione Beta che spieghi la probabilità di ammissione al college in funzione degli altri regressori.

Import del dataset e prima analisi

Per prima cosa leggiamo i dati presenti nel file "Admission_Predict.csv" (presente in allegato), facciamo una prima analisi dei dati importati e verifichiamo che non vi siano valori nulli.

```r
path2 <- paste(getwd(), "/Documenti/GitHub/ms-sl-2022/", "dataset/Admission_Predict.csv", sep = "", collapse = NULL)
dataSetBeta <- read.csv(file = path2, sep = ",", header = TRUE)
attach(dataSetBeta)
dim(dataSetBeta)
describe(dataSetBeta)
head(dataSetBeta)

# Confermiamo che non ci sono valori mancanti
sum(is.na(dataSetBeta))
```

```
dataSetBeta

 9  Variables      400  Observations
---------------------------------------------------------------------------------------------------------------------------------------------------------------
Serial.No.
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     400        0      400        1    200.5    133.7    20.95    40.90   100.75   200.50   300.25   360.10   380.05

lowest :   1   2   3   4   5, highest: 396 397 398 399 400
---------------------------------------------------------------------------------------------------------------------------------------------------------------
GRE.Score
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     400        0       49    0.999    316.8    13.15      298      300      308      317      325      332      336

lowest : 290 293 294 295 296, highest: 336 337 338 339 340
---------------------------------------------------------------------------------------------------------------------------------------------------------------
TOEFL.Score
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     400        0       29    0.997    107.4    6.933       98       99      103      107      112      116      118

lowest :  92  93  94  95  96, highest: 116 117 118 119 120
---------------------------------------------------------------------------------------------------------------------------------------------------------------
University.Rating
       n  missing distinct     Info     Mean      Gmd
     400        0        5    0.934    3.087    1.269

lowest : 1 2 3 4 5, highest: 1 2 3 4 5

Value          1     2     3     4     5
Frequency     26   107   133    74    60
Proportion 0.065 0.268 0.332 0.185 0.150
---------------------------------------------------------------------------------------------------------------------------------------------------------------
SOP
       n  missing distinct     Info     Mean      Gmd
     400        0        9     0.98      3.4    1.142

lowest : 1.0 1.5 2.0 2.5 3.0, highest: 3.0 3.5 4.0 4.5 5.0

Value        1.0   1.5   2.0   2.5   3.0   3.5   4.0   4.5   5.0
Frequency      6    20    33    47    64    70    70    53    37
Proportion 0.015 0.050 0.082 0.117 0.160 0.175 0.175 0.132 0.092
---------------------------------------------------------------------------------------------------------------------------------------------------------------

LOR
       n  missing distinct     Info     Mean      Gmd
     400        0        9    0.973    3.453    1.016

lowest : 1.0 1.5 2.0 2.5 3.0, highest: 3.0 3.5 4.0 4.5 5.0

Value        1.0   1.5   2.0   2.5   3.0   3.5   4.0   4.5   5.0
Frequency      1     7    38    39    85    73    77    45    35
Proportion 0.002 0.018 0.095 0.098 0.212 0.182 0.192 0.112 0.088
---------------------------------------------------------------------------------------------------------------------------------------------------------------
CGPA
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     400        0      168        1    8.599   0.6804    7.640    7.860    8.170    8.610    9.062    9.382    9.601

lowest : 6.80 7.20 7.25 7.28 7.30, highest: 9.80 9.82 9.87 9.91 9.92
---------------------------------------------------------------------------------------------------------------------------------------------------------------
Research
       n  missing distinct     Info      Sum     Mean      Gmd
     400        0        2    0.743      219   0.5475   0.4967

---------------------------------------------------------------------------------------------------------------------------------------------------------------
Chance.of.Admit
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90      .95
     400        0       60    0.999   0.7244    0.162     0.46     0.52     0.64     0.73     0.83     0.92     0.94

lowest : 0.34 0.36 0.38 0.39 0.42, highest: 0.93 0.94 0.95 0.96 0.97
---------------------------------------------------------------------------------------------------------------------------------------------------------------
```

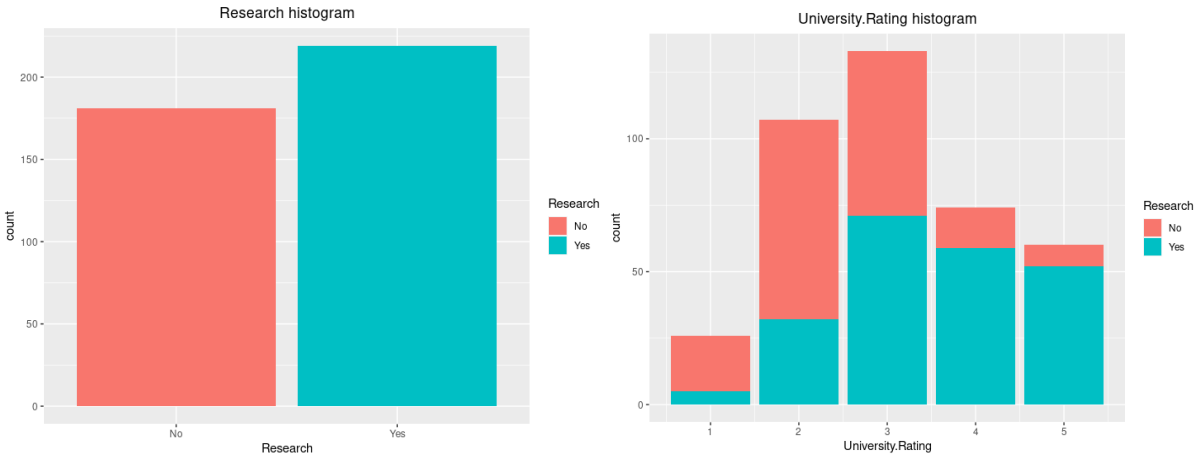
Possiamo concludere che non vi sono valori nulli, dunque procediamo ad una analisi grafica dei regressori al fine di comprendere meglio la distribuzione dei regressori.

Grafici esplorativi

Generiamo ora alcuni grafici dei regressori mediante la liberia ggplot2 la quale ci permette di aggregare i dati in base ad un particolare regressore, in questo caso Research, e generare istogrammi e densità categorizzando per tale regressore.

Dal momento che il modo per generare suddetti grafici è del tutto analogo riportiamo solo alcuni esempi. Per una visione completa del come sono stati generati questi grafici si rimanda alla visione dello script R in allegato.

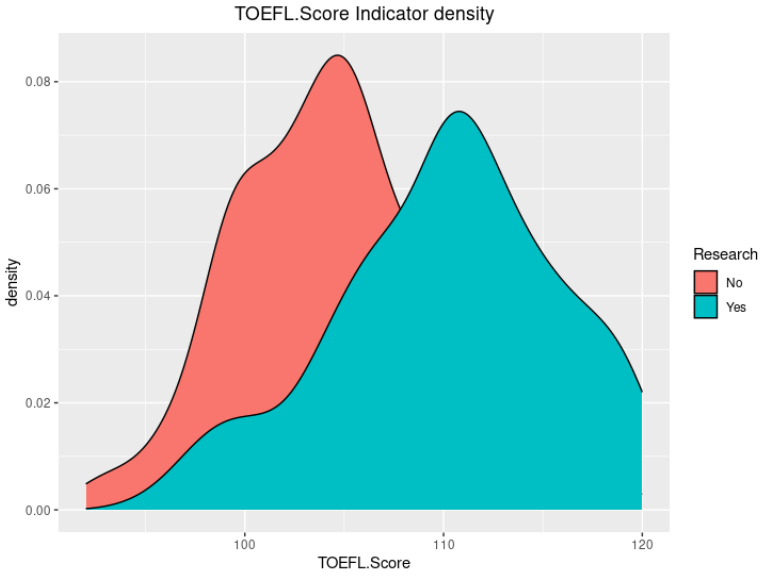
```
dataSetBetaForGraph = dataSetBeta%>%
  mutate(Research = recode(Research, "0" = "No", "1" = "Yes"))

# Istogramma ricerca si/no
ggplot(dataSetBetaForGraph, aes(x = Research, fill = Research)) +
  geom_bar() +
  ggtitle("Research histogram") +
  theme(plot.title = element_text(hjust = 0.5))

# Istogramma rating universitario
ggplot(dataSetBetaForGraph, aes(x = University.Rating, fill = Research)) +
  geom_bar() +
  ggtitle("University.Rating histogram") +
  theme(plot.title = element_text(hjust = 0.5))
```

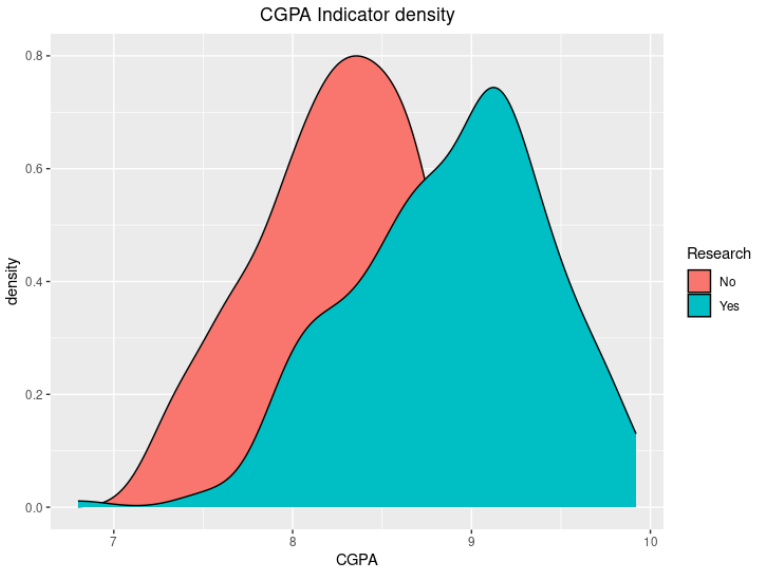


Gli istogrammi precedenti ci danno un'indicazione sul come sono distribuite le università nel dataset ed in particolare ci informano che le università con un rating maggiore sono anche quelle in cui si effettuano attività di ricerca più frequentemente.

```
# Desnità score GRE
ggplot(dataSetBetaForGraph, aes(x = GRE.Score, fill = Research)) +
  geom_density() +
  ggtitle("GRE Indicator density") +
  theme(plot.title = element_text(hjust = 0.5))
```


GRE Indicator density
density
0.04
0.03
0.02
0.01
0.00
290
300
310
320
330
340
GRE.Score
Research
No
Yes


TOEFL.Score Indicator density
density
0.08
0.06
0.04
0.02
0.00
100
110
120
TOEFL.Score
Research
No
Yes


CGPA Indicator density
density
0.8
0.6
0.4
0.2
0.0
7
8
9
10
CGPA
Research
No
Yes

I precedenti grafici riportano le densità di alcuni regressori nelle quali è evidente un'importante differenza in base al fatto o meno che l'università presa in esame svolga attività di ricerca. Nello specifico le università con un rating GRE più alto rientrano tra quelle che svolgono attività di ricerca, infatti è possibile osservare come la densità delle università che non svolgono ricerca si addensi sulla parte sinistra del grafico. È evidente inoltre che gli studenti con un maggiore punteggio scolastico (TOEFL e CGPA) applichino maggiormente nelle università in cui si svolge attività di ricerca, questo diventa sempre più evidente al crescere della performance scolastica degli studenti.

Stima di un modello di regressione Beta

Al fine di stimare un modello di regressione Beta il linguaggio R mette a disposizione la funzione *betareg* che dati la variabile dipendente ed i regressori genera un modello di regressione Beta. A titolo di esempio stimiamo il modello con tutti i regressori ed effettuiamo una prima analisi.

```r
# Modello con tutti i regressori
betaModel <- betareg(formula = Chance.of.Admit~
                       GRE.Score+
                       TOEFL.Score+
                       University.Rating+
                       SOP+
                       LOR+
                       CGPA+
                       Research, data = dataSetBeta)
summary(betaModel)# Pseudo R-squared: 0.8276
```

```
Call:
betareg(formula = Chance.of.Admit ~ GRE.Score + TOEFL.Score + University.Rating + SOP + LOR + CGPA + Research, data = dataSetBeta)

Standardized weighted residuals 2:
    Min      1Q  Median      3Q     Max
-3.6652 -0.4978  0.1272  0.7178  2.6319

Coefficients (mean model with logit link):
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      -9.918099   0.655891 -15.122  < 2e-16 ***
GRE.Score         0.010043   0.003045   3.298 0.000974 ***
TOEFL.Score       0.017876   0.005759   3.104 0.001907 **
University.Rating  0.044933   0.025416   1.768 0.077072 .
SOP              -0.028670   0.028432  -1.008 0.313269
LOR               0.107825   0.029164   3.697 0.000218 ***
CGPA              0.627918   0.062844   9.992  < 2e-16 ***
Research          0.122944   0.040769   3.016 0.002564 **

Phi coefficients (precision model with identity link):
      Estimate Std. Error z value Pr(>|z|)
(phi)   45.514      3.196   14.24   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Type of estimator: ML (maximum likelihood)
Log-likelihood: 563.2 on 9 Df
Pseudo R-squared: 0.8276
Number of iterations: 18 (BFGS) + 2 (Fisher scoring)
```

Possiamo osservare come il modello presenti un buon adattamento ai dati ma anche due regressori statisticamente poco significativi, per questa ragione possiamo rimuoverli ottenendo un modello a complessità minore.

```r
# Eliminiamo i regressori University.Rating e SOP dato che sono meno statisticamente significativi
betaModel2 <- betareg(formula = Chance.of.Admit~
                        GRE.Score+
                        TOEFL.Score+
                        LOR+
                        CGPA+
                        Research, data = dataSetBeta)
summary(betaModel2) # Pseudo R-squared: 0.8253
```

```
Call:
betareg(formula = Chance.of.Admit ~ GRE.Score + TOEFL.Score + LOR + CGPA + Research, data = dataSetBeta)

Standardized weighted residuals 2:
    Min      1Q  Median      3Q     Max
-3.6657 -0.4989  0.1107  0.6863  2.6253

Coefficients (mean model with logit link):
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -10.201962   0.619363 -16.472  < 2e-16 ***
GRE.Score     0.010310   0.003049   3.382 0.000720 ***
TOEFL.Score   0.018928   0.005628   3.363 0.000771 ***
LOR           0.108741   0.025459   4.271 1.94e-05 ***
CGPA          0.642341   0.060816  10.562  < 2e-16 ***
Research      0.122903   0.040795   3.013 0.002589 **

Phi coefficients (precision model with identity link):
      Estimate Std. Error z value Pr(>|z|)
(phi)    45.14       3.17   14.24   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Type of estimator: ML (maximum likelihood)
Log-likelihood: 561.5 on 7 Df
Pseudo R-squared: 0.8253
Number of iterations: 16 (BFGS) + 2 (Fisher scoring)
```