Università della Calabria

Dipartimento di Economia, Statistica e Finanza "*Giovanni Anania*"

Corso di Laurea Magistrale in Statistica e Informatica per le Decisioni e le Analisi di Mercato

Reti neurali e dispositivi intelligenti. Applicazione sui sensor data di uno smartphone.

Docente

Prof. Paolo Carmelo Cozzucoli

Studenti

Tucci Marco Matr. 214216

Ruffo Pietro Matr. 214241

Anno Accademico 2020/2021

INTRODUZIONE

I device che utilizziamo quotidianamente, come smartphone o smartwatch, sono in grado di rilevare, analizzare e trasmettere informazioni riguardanti, ad esempio, segnali corporei – come segni vitali – e dati ambientali.
I dispositivi così intesi hanno un grande potenziale per essere sfruttati in diversi campi di applicazione che spaziano da quello medico, alla sicurezza, passando per il tempo libero e l'intrattenimento. Tali dispositivi sono versatili e quindi adattabili a diversi sistemi ed applicazioni. Oltre a questo, gli utenti li possono indossare continuamente, a stretto contatto con i loro corpi, così che i dati individuali possano essere costantemente raccolti, e possano essere fornite immediatamente risposte e notifiche.
Alla base delle funzionalità fornite da molti di questi dispositivi vi sono le reti neurali.
Le reti neurali sono sempre più importanti all'interno dei dispositivi mobili, in quanto permettono di sfruttare il Machine Learning per creare dei sistemi flessibili, in grado di imparare e di migliorarsi con il tempo, superando molti dei limiti delle tecnologie attuali.
L'analisi realizzata si propone di mostrare una semplice applicazione delle reti neurali ai dispositivi mobili con l'obiettivo di manifestarne l'enorme potenziale.

L'obiettivo del progetto è quello di costruire una rete neurale supervisionata in grado di rilevare se una persona sta correndo o camminando in base ai dati dei sensori raccolti da un dispositivo iOS.
Il dataset a disposizione contiene 4000 campioni di dati di sensori raccolti dall'accelerometro e dal giroscopio di un iPhone 5c con un intervallo di 10 secondi e una frequenza di ~5.4/secondo.
Un accelerometro è uno strumento di misura in grado di misurare l'accelerazione, effettuando il calcolo della forza rilevata rispetto alla massa dell'oggetto. Un giroscopio è, invece, un dispositivo fisico rotante che serve a misurare l'inclinazione dello smartphone.
Questi dati sono rappresentati dalle seguenti variabili (ogni variabile contiene i dati del sensore per uno degli assi del sensore):

- accelerazione_x
- accelerazione_y
- accelerazione_z
- giro_x
- giro_y
- giro_z

La variabile "activity" funge da variabile di output e riflette le seguenti attività:

- "0": camminare
- "1": correre

Il dataset contiene una colonna denominata "wrist" che rappresenta il polso sul quale il dispositivo è stato posizionato per raccogliere il campione:

- "0": polso sinistro
- "1": polso destro

Inoltre, il set di dati contiene le colonne "data", "ora" e "nome utente" che forniscono informazioni sulla data esatta, l'ora e l'utente che ha raccolto queste misurazioni.

Le variabili sulle quali verrà posta l'enfasi sono quelle relative ai diversi assi dimensionali dei sensori e l'activity, che nello specifico rappresenta il target.

La possibilità di poter sfruttare le informazioni relative ad una variabile di output pone l'analisi nel contesto dell'apprendimento supervisionato. In particolare, si propone di risolvere un problema di classificazione binaria: si vuole capire, sulla base dei dati rilevati dai sensori, se l'attività dell'utente rientra in una delle due categorie, camminare ovvero correre.

DESCRIZIONE DEL DATASET DI ADDESTRAMENTO E PRE-PROCESSAMENTO DEI DATI

Il software utilizzato per condurre l'analisi è "RStudio". Le librerie adoperate sono le seguenti: "neuralnet", "NeuralNetTools" e "caret".

Il dataset di addestramento a disposizione contiene 4000 osservazioni e presenta le seguenti 11 variabili:

- Date
- Time
- Username
- Wrist
- Activity (1 indica correre e 0 camminare)
- Acceleration_x
- Acceleration_y
- Acceleration_z
- Gyro_x
- Gyro_y
- Gyro_z

```
> dim(my_data)
[1] 4000   11

> head(my_data)
       date                time username wrist activity acceleration_x
1 2017-7-17 20:14:32:684908986   viktor     1        1         0.1304
2  2017-7-1 18:12:11:604216992   viktor     0        1         0.6485
3 2017-6-30 20:54:12:439673006   viktor     0        1        -0.5367
4 2017-7-16 14:46:43:635783016   viktor     1        0        -0.3166
5  2017-7-6  16:3:26:684176981   viktor     0        0         0.1409
6  2017-7-2 19:58:23:373413980   viktor     0        1         0.3447
  acceleration_y acceleration_z  gyro_x  gyro_y  gyro_z
1        -0.6496         0.2983 -0.7034 -0.5678 -1.1277
2         0.0212        -0.6460 -0.4591  0.9264  1.3082
3        -0.3479         0.2677  1.3178 -0.6878 -0.2581
4        -1.3171        -0.2849 -0.5361 -0.2142  0.6944
5        -0.7736        -0.0884 -1.2900  0.2583 -1.2345
6        -0.6172        -0.0260  1.6312 -1.3866 -0.2773
```

Dal dataset si estraggono le 6 variabili che costituiranno gli input della rete neurale (accelaration_x, accelaration_y, accelaration_z, gyro_x, gyro_y, gyro_z) e la variabile di output (activity). Tutte le variabili selezionate sono di tipo numerico. Per ognuna di esse si riportano le principali misure descrittive.

```
> head(my_data)
     activity acceleration_x acceleration_y acceleration_z  gyro_x  gyro_y  gyro_z
2463        1        -2.8928        -0.2228        -0.5819 -2.3331 -2.2071 -1.2718
2511        0        -0.3447        -1.1210        -0.3949 -0.2035  1.4917  0.4575
2227        0         0.3637        -0.7933        -0.0987 -0.0215  0.8079  0.5723
526         1        -0.2315         0.4813        -0.1984 -0.3022  1.6448  0.5257
```

```
195         0         0.5584        -1.1192        -0.4042  0.0572 -0.4700 -0.0036
2986        0        -0.2492        -0.9252        -0.2247  0.4978 -1.1279 -0.6571


> summary(my_data)
    activity      acceleration_x     acceleration_y     acceleration_z
 Min.   :0.0000   Min.   :-4.30030   Min.   :-2.0100   Min.   :-2.89940
 1st Qu.:0.0000   1st Qu.:-0.37643   1st Qu.:-1.0328   1st Qu.:-0.36988
 Median :0.0000   Median :-0.02670   Median :-0.7618   Median :-0.21500
 Mean   :0.4993   Mean   :-0.06346   Mean   :-0.5630   Mean   :-0.31506
 3rd Qu.:1.0000   3rd Qu.: 0.35440   3rd Qu.:-0.2492   3rd Qu.:-0.08468
 Max.   :1.0000   Max.   : 4.04590   Max.   : 2.4185   Max.   : 1.31650
     gyro_x             gyro_y             gyro_z
 Min.   :-3.65460   Min.   :-5.44440   Min.   :-9.48000
 1st Qu.:-0.89267   1st Qu.:-0.63990   1st Qu.:-1.36902
 Median : 0.00030   Median : 0.03400   Median : 0.02975
 Mean   : 0.01506   Mean   : 0.02609   Mean   : 0.02011
 3rd Qu.: 0.86572   3rd Qu.: 0.69358   3rd Qu.: 1.40775
 Max.   : 4.53630   Max.   : 4.45330   Max.   : 6.94650

> str(my_data)
'data.frame':  4000 obs. of  7 variables:
 $ activity      : int  1 0 0 1 0 0 0 0 1 0 ...
 $ acceleration_x: num  -2.893 -0.345 0.364 -0.232 0.558 ...
 $ acceleration_y: num  -0.223 -1.121 -0.793 0.481 -1.119 ...
 $ acceleration_z: num  -0.5819 -0.3949 -0.0987 -0.1984 -0.4042 ...
 $ gyro_x        : num  -2.3331 -0.2035 -0.0215 -0.3022 0.0572 ...
 $ gyro_y        : num  -2.207 1.492 0.808 1.645 -0.47 ...
 $ gyro_z        : num  -1.2718 0.4575 0.5723 0.5257 -0.0036 ...
```

Occorre “preparare” il dataset che permetterà alla rete di apprendere come risolvere al meglio il problema.

Prima di tutto, per far sì che nel dataset non sia presente alcun tipo di struttura dei dati, sono state randomizzate le osservazioni.

È stata verificata sia l’assenza di valori mancanti che di multicollinearità tra le variabili esplicative.

```
> table(is.na.data.frame(my_data))

FALSE
28000

> cor(my_data[,c(2:ncol(my_data))])
              acceleration_x acceleration_y acceleration_z      gyro_x      gyro_y
acceleration_x     1.00000000    -0.27483172    -0.53861758 -0.06391566 -0.04175623
acceleration_y    -0.27483172     1.00000000     0.09452244  0.02898744  0.07967053
acceleration_z    -0.53861758     0.09452244     1.00000000  0.04349350  0.02536683
gyro_x            -0.06391566     0.02898744     0.04349350  1.00000000  0.10902718
gyro_y            -0.04175623     0.07967053     0.02536683  0.10902718  1.00000000
gyro_z            -0.10376917    -0.02204610     0.08250210  0.33043440  0.29575738
                   gyro_z
acceleration_x -0.1037692
acceleration_y -0.0220461
acceleration_z  0.0825021
gyro_x          0.3304344
gyro_y          0.2957574
gyro_z          1.0000000
```

Il dataset è risultato essere bilanciato rispetto ai valori della variabile risposta.

```
> table(activity)
activity
   0    1
2003 1997
```

Per evitare problemi di convergenza all'algoritmo di ottimizzazione e renderlo più efficiente, si è proceduto alla normalizzazione dei dati di addestramento rispetto al campo di variazione della variabile: i valori sono ora compresi tra 0 e 1.

```
> summary(dati_scaled)
    activity      acceleration_x   acceleration_y   acceleration_z
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.0000   1st Qu.:0.4701   1st Qu.:0.2207   1st Qu.:0.6000
 Median :0.0000   Median :0.5120   Median :0.2819   Median :0.6367
 Mean   :0.4993   Mean   :0.5076   Mean   :0.3268   Mean   :0.6130
 3rd Qu.:1.0000   3rd Qu.:0.5577   3rd Qu.:0.3976   3rd Qu.:0.6676
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
     gyro_x           gyro_y           gyro_z
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.:0.3372   1st Qu.:0.4854   1st Qu.:0.4938
 Median :0.4462   Median :0.5535   Median :0.5789
 Mean   :0.4480   Mean   :0.5527   Mean   :0.5783
 3rd Qu.:0.5519   3rd Qu.:0.6201   3rd Qu.:0.6628
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

Infine, dopo aver settato un seme per la riproducibilità dei risultati, è stato effettuato uno split del dataset in training dataset (70%) e test dataset (30%), in modo tale da poter addestrare e validare correttamente i modelli di rete neurale.

```
> set.seed(123)
> index<-sample(1:nrow(my_data),round(0.70*nrow(my_data)))

> train_data<-as.data.frame(my_data[index,])
> train_data_scaled<-as.data.frame(dati_scaled[index,])

> test_data<-as.data.frame(my_data[-index,])
> test_data_scaled<-as.data.frame(dati_scaled[-index,])
```

ADDESTRAMENTO E VALIDAZIONE DEI MODELLI

1. MODELLI DI RETE NEURALE

Segue la forma funzionale adottata per addestrare i modelli di rete neurale.

```
> f
activity ~ acceleration_x + acceleration_y + acceleration_z +
    gyro_x + gyro_y + gyro_z
```

La funzione di attivazione scelta è la funzione logistica perché si tratta di risolvere un problema di classificazione binaria. Non è stata scelta la funzione tangente iperbolica in quanto non si desidera utilizzare un gradiente troppo forte.

Le reti che sono state costruite sono tutte di tipo feed-forward. Per addestrare la rete, cioè stimare e aggiornare i pesi (parametri) del modello, si procede con un processo di back-propagation, che viene realizzato tramite il metodo iterativo di ottimizzazione del gradiente discendente.

I tre modelli di rete neurale proposti sono i seguenti:

- Rete neurale (*model*) con 1 strato nascosto avente 5 neuroni, e con livello di errore pari a 0.01
- Rete neurale (*model2*) con 2 strati nascosti aventi ciascuno 5 neuroni, e con livello di errore pari a 0.01
- Rete neurale (*model3*) con 1 strato nascosto avente 5 neuroni, e con livello di errore pari a 0.001

Dopo aver addestrato il primo modello (*model*) sul training dataset, questo è stato testato sul dataset di test, ottenendo delle previsioni (valori compresi tra 0 e 1) che sono state poi arrotondate in modo da ottenere valori pari a 0 o 1. La validazione è stata effettuata tramite la matrice di confusione e diverse statistiche.

Il modello, nel complesso, mostra un'ottima capacità di risolvere il problema; in particolare, con 19 casi mal classificati, presenta un valore di accuracy pari a 0.9842.

```
> conf
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 597  11
         1   8 584

               Accuracy : 0.9842
                 95% CI : (0.9754, 0.9904)
    No Information Rate : 0.5042
```

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

                  Kappa : 0.9683

 Mcnemar's Test P-Value : 0.6464

            Sensitivity : 0.9868
            Specificity : 0.9815
         Pos Pred Value : 0.9819
         Neg Pred Value : 0.9865
             Prevalence : 0.5042
         Detection Rate : 0.4975
   Detection Prevalence : 0.5067
      Balanced Accuracy : 0.9841

       'Positive' Class : 0
```

Procedendo con la validazione del modello di rete neurale *model2*, si ottiene un valore di accuracy pari a 0.9825 (le unità mal classificate sono 21).

```
> conf2
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 595  11
         1  10 584

               Accuracy : 0.9825
                 95% CI : (0.9734, 0.9891)
    No Information Rate : 0.5042
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.965

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9835
            Specificity : 0.9815
         Pos Pred Value : 0.9818
         Neg Pred Value : 0.9832
             Prevalence : 0.5042
         Detection Rate : 0.4958
   Detection Prevalence : 0.5050
      Balanced Accuracy : 0.9825

       'Positive' Class : 0
```

Nonostante tale rete neurale abbiamo 5 neuroni in più, il modello addestrato presenta delle performance leggermente inferiori rispetto a quelle ottenute con il modello precedente (più semplice). Questo è dovuto al fatto che, nella maggior parte dei casi, per risolvere problemi non complessi (come quello proposto), un secondo o un terzo strato nascosto è raramente utile ed un solo strato intermedio permette un miglior funzionamento della rete.

Altri motivi potrebbero ricercarsi in rules-of-thumb (per determinare il numero di strati nascosti ed il loro numero di neuroni) individuate e definite grazie a sperimentazioni e intuizioni da parte di diversi autori.[1]

Un altro aspetto che è stato tenuto in considerazione è il problema dell'over-fitting: per il problema abbastanza semplice da risolvere, si è evitato di usare 2 o più strati nascosti aventi un numero di nodi alto, in modo tale da non avere un eccesso di parametri nel modello, che avrebbe potuto determinare una bontà di adattamento molto alta ma innaturale e risultati poco soddisfacenti nella fase di validazione.

Per i vari motivi finora esposti, tra i due modelli è stato scelto *model* con un solo strato nascosto avente 5 neuroni.

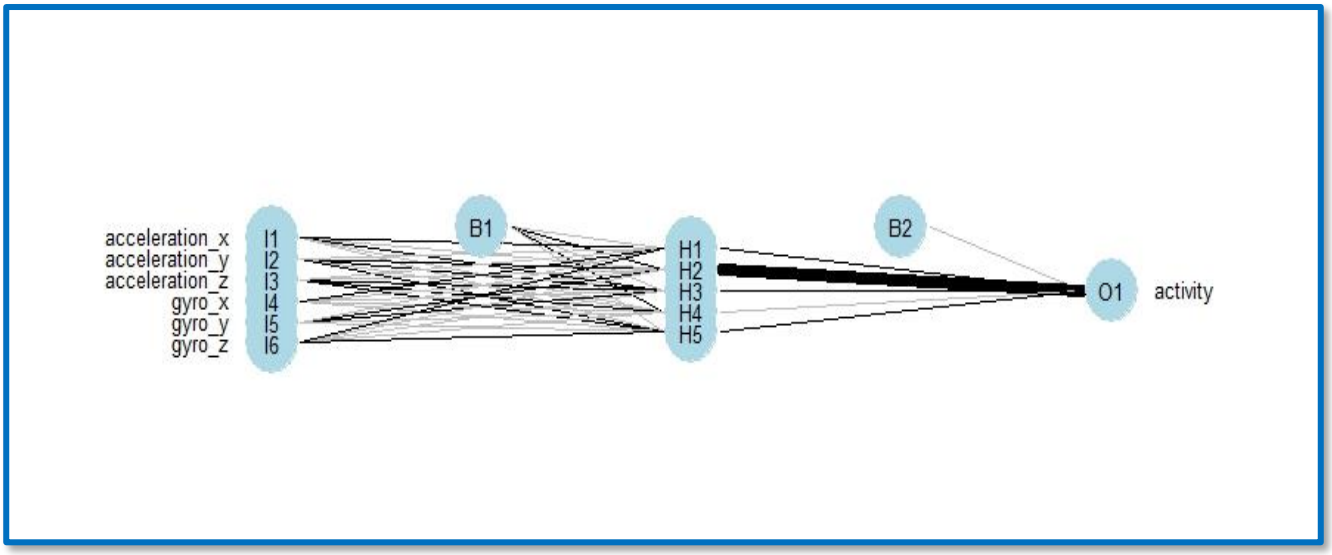
Il modello di rete neurale *model3* differisce da *model* per un più basso livello di errore imposto (0.001); *model3* presenta un valore di accuracy pari a 0.9883 (con 14 casi mal classificati), che è maggiore rispetto al valore ottenuto in *model*.

```
> conf3
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 600   9
         1   5 586

               Accuracy : 0.9883
                 95% CI : (0.9805, 0.9936)
    No Information Rate : 0.5042
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9767

 Mcnemar's Test P-Value : 0.4227

            Sensitivity : 0.9917
            Specificity : 0.9849
         Pos Pred Value : 0.9852
         Neg Pred Value : 0.9915
             Prevalence : 0.5042
         Detection Rate : 0.5000
   Detection Prevalence : 0.5075
      Balanced Accuracy : 0.9883

       'Positive' Class : 0
```

Dal momento che nel complesso le prestazioni sono migliori rispetto a *model* e che non si ha alcun aumento della complessità del modello (stesso numero di parametri di *model*), si preferisce *model3*.

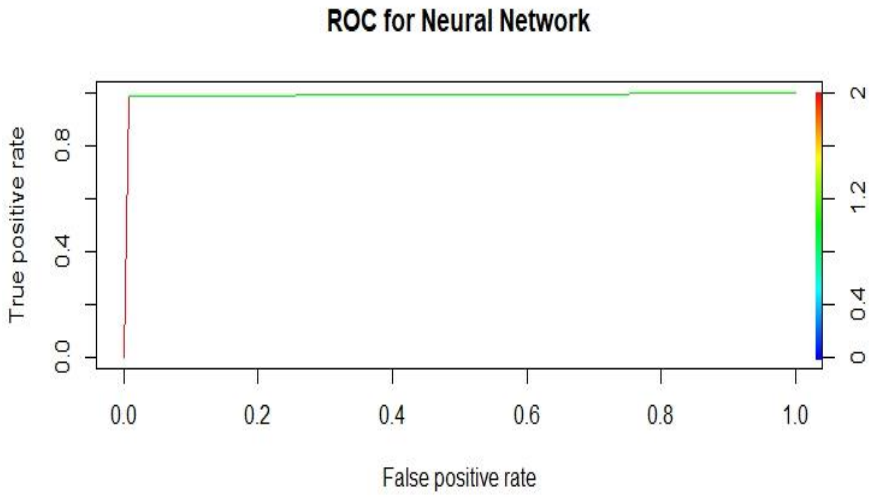
Segue il grafico relativo al modello di rete neurale scelto.

[1] un riferimento a quanto detto è esposto sul sito web https://www.heatonresearch.com/2017/06/01/hidden-layers.html



Il numero di parametri del modello è pari alla somma del numero di pesi sinaptici delle connessioni tra i nodi e del numero di pesi dei bias: 6*5 + 5*1 + 1*5 + 1*1 = 41.

Dal seguente grafico, è possibile osservare la curva ROC (curva Operativa Caratteristica) del modello. La curva ROC, che costituisce il complementare della funzione potenza, permette di valutare classificatori binari, in quanto è un indicatore della capacità del modello di risolvere in maniera adeguata il problema di classificazione binaria.



L'area sottostante la curva ROC (AUC, acronimo dei termini inglesi "Area Under the Curve") è pari a 0.9883, pertanto il modello può essere considerato buono.

```
> performance(f_prediction,"auc")@y.values
[[1]]
[1] 0.9883047
```

2. MODELLO DI REGRESSIONE LOGISTICA

Per valutare l'effettiva superiorità in termini di performance predittiva della rete neurale supervisionata, si considera un metodo antagonista che abbia come scopo quello di assolvere al medesimo compito ergo risolvere lo stesso problema.
Il metodo concorrenziale realizzato è rappresentato da un modello di regressione logistica, in cui la variabile dipendente è "activity", mentre le variabili indipendenti corrispondono ai nodi di input della rete neurale, ovvero i dati dei sensori raccolti dall'accelerometro e dal giroscopio nelle tre dimensioni x, y e z.

```
> mod<-glm(formula=f,data=train_data_scaled,family=binomial(link=logit))
> summary(mod)

Call:
glm(formula = f, family = binomial(link = logit), data = train_data_scaled)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.6910  -0.5441   0.0004   0.2430   3.6137

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)     -5.8248     0.8153  -7.144 9.04e-13 ***
acceleration_x   4.8977     0.7382   6.634 3.26e-11 ***
acceleration_y  31.1356     1.4174  21.967  < 2e-16 ***
acceleration_z  -9.9118     0.7122 -13.916  < 2e-16 ***
gyro_x           0.3202     0.4767   0.672 0.501778
gyro_y          -1.8489     0.5795  -3.191 0.001419 **
gyro_z           2.3183     0.6273   3.696 0.000219 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3881.6  on 2799  degrees of freedom
Residual deviance: 1816.8  on 2793  degrees of freedom
AIC: 1830.8

Number of Fisher Scoring iterations: 7
```

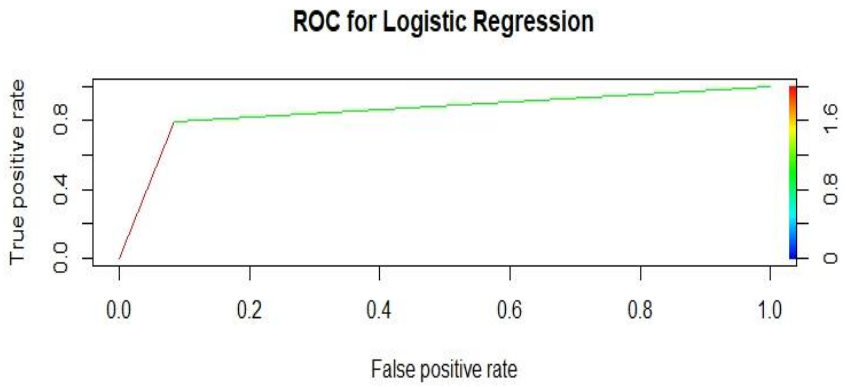
Dall'output del modello, si osserva che tutte le variabili esplicative inserite sono statisticamente significative (con p-value al di sotto della soglia 0.01), ad eccezione di "gyro_x" (con p-value pari a 0.50). I valori rilevati dal giroscopio sull'asse delle x non risultano utili a spiegare la dipendente.

Dalla matrice di confusione relativa al modello logit, è possibile ricavare un valore di accuracy pari a 0.8542: il modello è in grado di classificare correttamente i dati nell'85.4% dei casi.

```
> conf_reg
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 554 124
         1  51 471
```

```
               Accuracy : 0.8542
                 95% CI : (0.8329, 0.8737)
    No Information Rate : 0.5042
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.708

 Mcnemar's Test P-Value : 5.248e-08

            Sensitivity : 0.9157
            Specificity : 0.7916
         Pos Pred Value : 0.8171
         Neg Pred Value : 0.9023
             Prevalence : 0.5042
         Detection Rate : 0.4617
   Detection Prevalence : 0.5650
      Balanced Accuracy : 0.8536

       'Positive' Class : 0
```

Anche per questo modello si costruisce la curva ROC allo scopo di valutare al meglio le performance predittive della regressione logistica e di poter così eseguire un confronto completo con la rete neurale supervisionata.



L'area sottostante la curva ROC è, in questo caso, pari a 0.8536, per tanto il modello logit può essere considerato "adeguato".

```
> performance(f_prediction_reg,"auc")@y.values
[[1]]
[1] 0.8536496
```

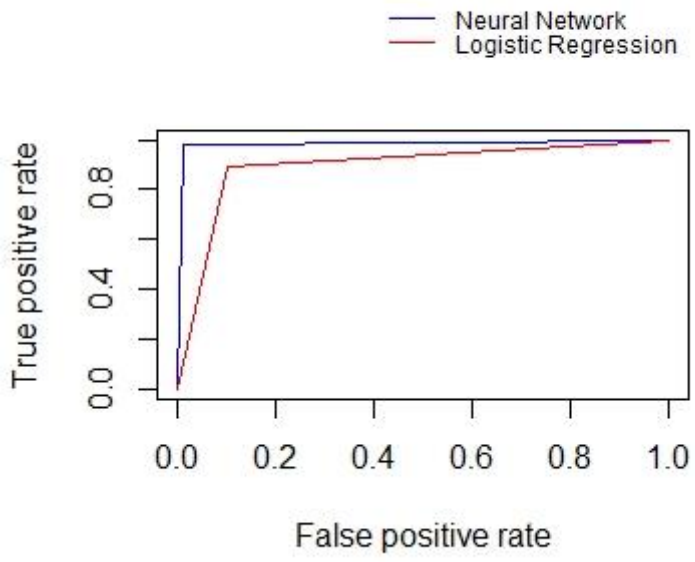
3. CONFRONTO TRA RETE NEURALE E REGRESSIONE LOGISTICA

A questo punto si procede con il confronto tra il modello di rete neurale scelto (*model3*) ed il modello di regressione logistica costruito.

Considerando la misura di accuracy, si nota che, con un valore quasi prossimo a 0.99, la rete fornisce classificazioni migliori rispetto al modello di regressione.

```
> conf3
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 600   9
         1   5 586

               Accuracy : 0.9883

> conf_reg
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 554 124
         1  51 471

               Accuracy : 0.8542
```

Analogo risultato può essere riscontrato graficamente per mezzo delle curve ROC dei due modelli.



L’area sotto la curva ROC è più grande per il modello di rete neurale (0.9883 vs 0.8536), infatti la curva del modello di rete neurale domina la curva del modello di regressione logistica.

A conclusione del confronto, è possibile affermare che sarà più opportuno rendere operativa la rete neurale (piuttosto che il modello logit) in modo da poter risolvere al meglio il problema proposto di classificazione binaria (il soggetto corre o cammina) per nuovi futuri campioni di dati.

CONCLUSIONI

Questo era solo un piccolo esempio di come le reti neurali possano essere impiegate per analizzare l'enorme quantità di dati rilevati dai dispositivi indossabili.
I dispositivi indossabili sono in grado di offrire prestazioni sempre migliori: ci permettono di rimanere sempre aggiornati e connessi, di rilevare parametri fisici e biologici – come la pressione e la frequenza cardiaca, il livello di ossigenazione del sangue, il tipo e la qualità dell'attività fisica, la durata e la qualità del sonno, … – di fornirci, quindi, informazioni sul nostro stato di salute e benessere, di spronarci a mantenere buone abitudini, e addirittura di avvisare dei contatti di emergenza nel caso rilevassero situazioni pericolose per la nostra salute. Sotto questo punto di vista, quindi, tali strumenti, monitorando i nostri parametri vitali, ci permettono di migliorare la qualità della vita, di salvaguardare il più possibile la salute e di estendere le capacità degli altri dispositivi – come smartphone e computer – che utilizziamo quotidianamente.