Indagine Sulla Potabilità dell’Acqua

MASTER IN AI&DS UNICAL-TREE BASED MODELS

Corsisti:
Giovanni Tripicchio
Marta Calasso
Mariachiaria Mannoccio
Daniele Cristofori

Indice

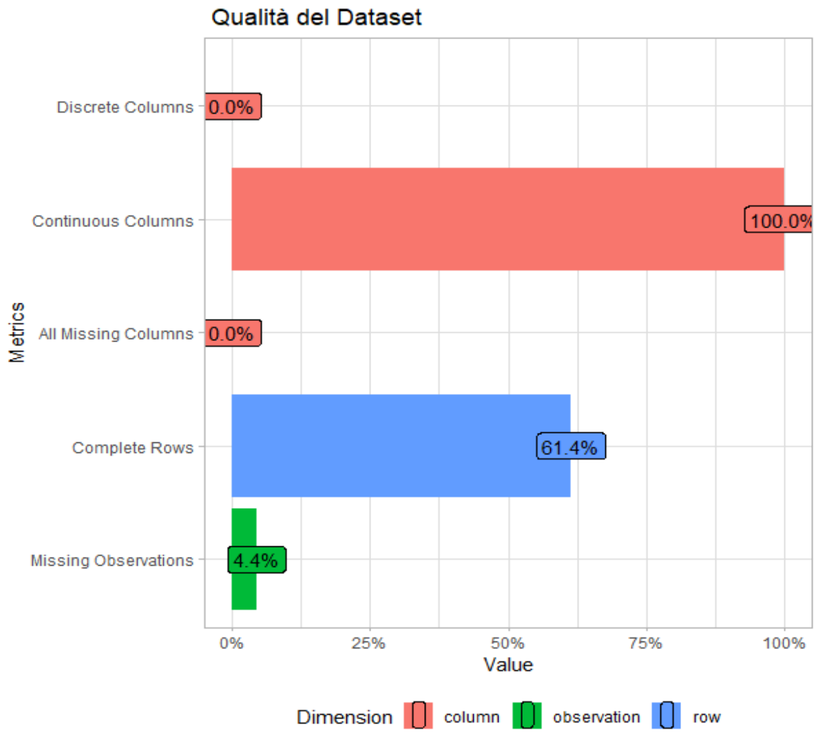
Analisi del Dataset

Il dataset preso in esame, denominato “Water Potability”, è un file open source disponibile su Kaggle, composto da 3276 righe e 10 colonne che fornisce informazioni su diverse caratteristiche chimiche dell’acqua:

- pH
- Durezza
- Solidi sospesi totali
- Clorammine
- Solfato
- Conducibilità
- Carbonio organico
- Trihalometani
- Torbidità
- Potabilità

Lo scopo principale del progetto è valutare la potabilità dell’acqua in base a tali caratteristiche chimiche; pertanto la variabile target binomiale sarà la Potabilità .

Volendo approfondire l’analisi sulla qualità del dataset:

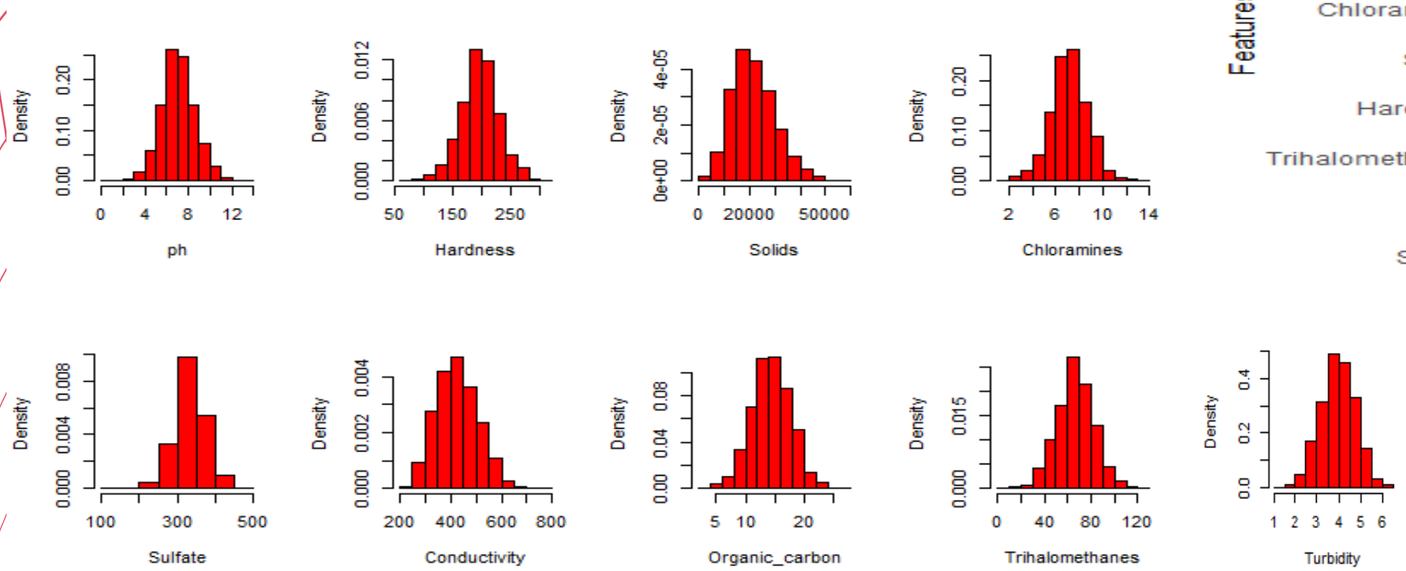


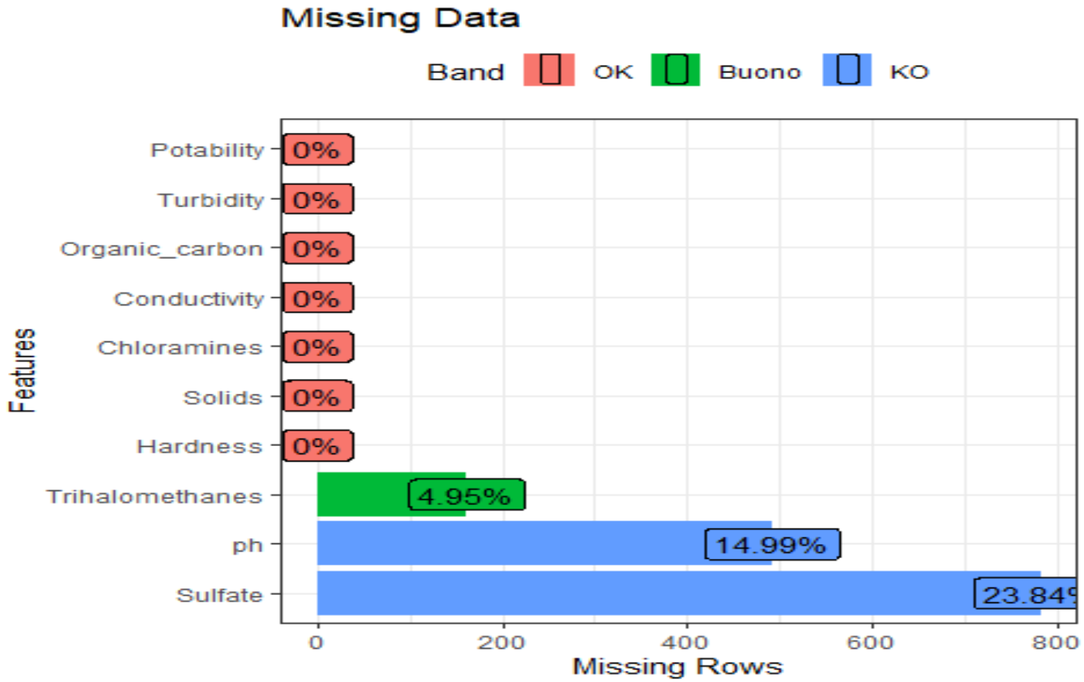
Emerge la mancanza del 4,4% dei valori, che determina il 61,4% di righe complete.

Analisi del Dataset

Decidiamo di rimuovere le osservazioni corrispondenti. Il file così modificato si riduce a 2011 osservazioni distribuite e 10 variabili.

- La maggior parte delle variabili mostra una distribuzione simmetrica con un picco centrale, il che indica che i valori tendono ad essere distribuiti uniformemente intorno alla media.
- Alcune variabili, come i solidi disciolti e i trialometani, mostrano code lunghe, indicando la presenza di outlier o valori estremi.





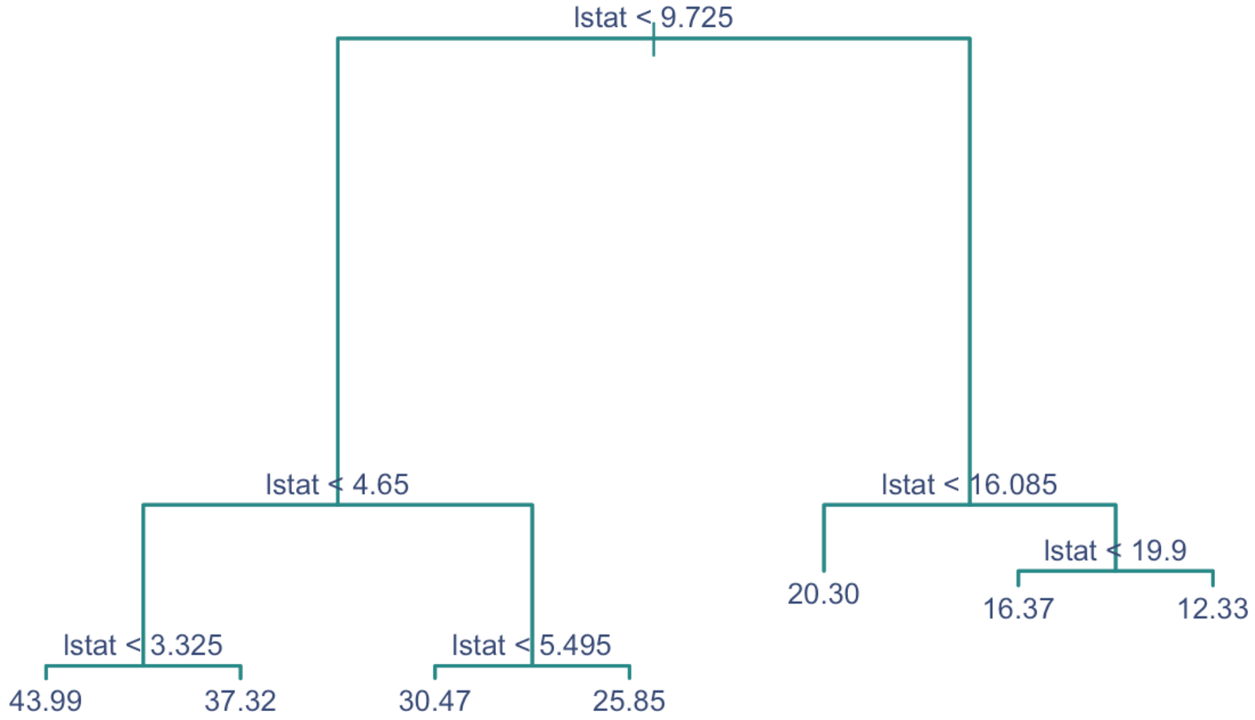
Modelli Predittivi

Un albero decisionale è un algoritmo di apprendimento supervisionato utilizzato sia per tecniche di classificazione che di regressione. Si compone di una struttura ad albero gerarchica, che consiste di un nodo radice, di rami, nodi interni e nodi foglia. I rami in uscita dal nodo radice alimentano i nodi interni, noti anche come nodi decisionali. Sulla base delle funzionalità disponibili, entrambi i tipi di nodi conducono valutazioni per formare sottoinsiemi che sono rappresentati di nodi foglia o nodi terminali.

L'apprendimento dell'albero decisionale prevede l'utilizzo di una strategia che consiste nell'identificazione dei punti di divisione ottimali all'interno di un albero, questo processo di suddivisione viene ripetuto in modo ricorsivo dall'alto verso il basso.

La scelta dei modelli predittivi da applicare al dataset si è incentrata su:

- **Modello Cart - Classification And Regression Trees**
- **Random Forest**
- **Boosting**



Parametri training e test

Il dataset è stato suddiviso in modo che il 70% delle osservazioni venga utilizzato per l'addestramento e il 30% per il test. Possiamo notare che è garantito un corretto bilanciamento tra i valori della variabile di interesse:

Train →

0	1
836	572

Test →

0	1
364	239

Il bilanciamento è cruciale per evitare bias nel modello predittivo, assicurando che entrambi i valori della variabile target (0 e 1) siano rappresentati adeguamente sia nel training set che nel test set.

Viene impostato il Seed al fine di garantire la riproducibilità della suddivisione.

La variabile «Potability» viene convertita in fattore al fine di trattare il problema come una classificazione, con livelli «NO» e «YES»

```
set.seed(11)
## Dividi il dataset in training set e test set in 70% training set e 30% test set
trainIndex_0 <- createDataPartition(data$Potability, p = .7,list = FALSE,times = 1)

# per bilanciare positivi e negativi

imbal_train_1 <- data[ trainIndex_0,]
imbal_test_1 <- data[ -trainIndex_0,]


table(data$Potability)

table(imbal_train_1$Potability)
table(imbal_test_1$Potability)

# fattorizzazione della variabile outcome del train set
imbal_train_1$Potability <- as.factor(imbal_train_1$Potability)
levels(imbal_train_1$Potability) <- c("NO", "YES")



# fattorizzazione della variabile outcome del test set
imbal_test_1$Potability <- as.factor(imbal_test_1$Potability)
levels(imbal_test_1$Potability) <- c("NO", "YES")
```

Cart

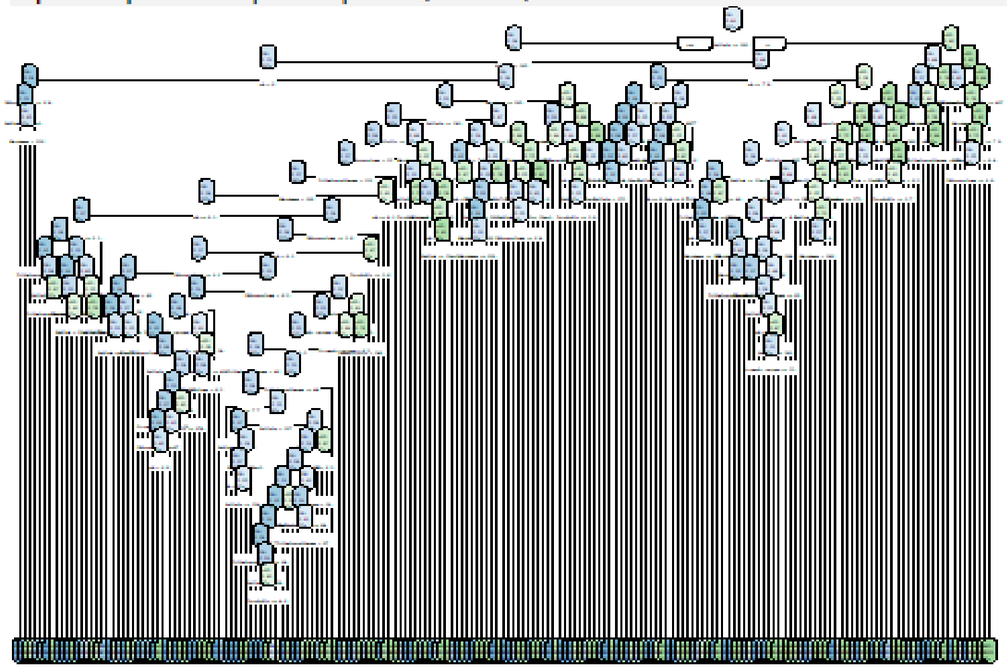
Applicazione del Modello Cart

L'obiettivo dell'algoritmo CART è suddividere ricorsivamente il dataset di train in sottoinsiemi omogenei in modo da massimizzare la purezza delle classi (problemi di classificazione) o minimizzare l'errore nella previsione della variabile target, (problema di regressione).

1. **Scelta della variabile predittiva e della soglia di divisione**
2. **Divisione del dataset**
3. **Calcolo della purezza o dell'errore**
4. **Ripetizione ricorsiva**
5. **Costruzione dell'albero.**

TEST 1: Applicazione modello CART

```
set.seed(11)
Tree1 <- rpart(Potability ~ ., data=imbal_train_1 ,method = "class",
               control=rpart.control(minsplit=5,cp=0))
rpart.plot::rpart.plot(Tree1)
```



Confusion Matrix

TRAIN

predicted	actual NO	YES
NO	801	38
YES	35	534

0.9481534

TEST

predicted	actual NO	YES
NO	234	121
YES	130	118

0.5837479

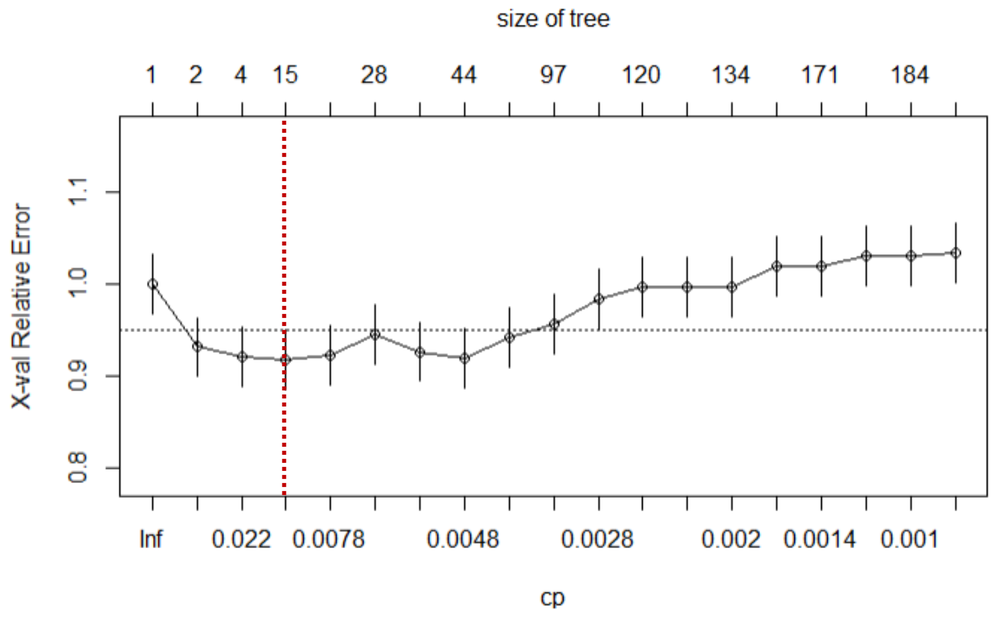
Albero troppo complesso e include rumore e dettagli irrilevanti. Questo significa che il modello non è riuscito ad apprendere abbastanza dai dati di addestramento e non è in grado di generalizzare bene sui dati che non ha mai visto prima, come quelli del test set.

Cart

Pruning dell'albero

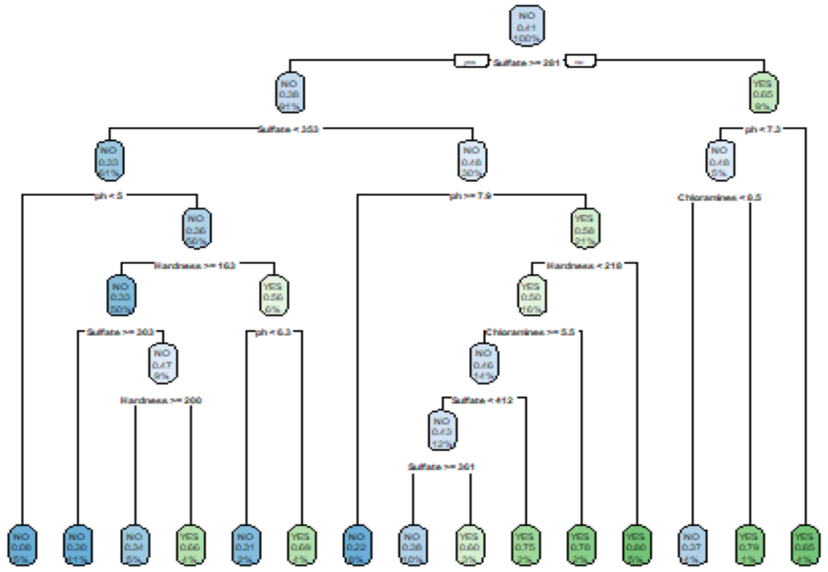
Per ridurre la complessità degli alberi si ricorre al pruning dell' albero attraverso il **parametro di complessità** (CP).
Il valore di CP minimizza l'errore di classificazione sul set di dati di addestramento identificando il numero di split a cui fermarsi .

```
rpart::plotcp(Tree1)
```




```
Tree1p <- prune(Tree1, cp = Tree1$cptable[which.min(Tree1$cptable[,"xerror"]),"CP"])
rpart.plot::rpart.plot(Tree1p)
```

Pruning



L'albero potato è più semplice e meno suscettibile all'overfitting rispetto all'albero non potato.

Cart

Risultati albero potato

```
predettoP.tr <- predict(Tree1p,  type = "class")
table(predicted = predettoP.tr, actual = imbal_train_1$Potability)
```

```
##  TRAIN   actual
## predicted  NO YES
##       NO  746 311
##       YES  90 261
```

```
# Accuracy for train set
mean(imbal_train_1$Potability == predettoP.tr)
```

```
## [1] 0.7151989
```

→ **Albero non potato 0.95**

```
predettoP.te <- predict(Tree1p,  type = "class", newdata = imbal_test_1)
table(predicted = predettoP.te, actual = imbal_test_1$Potability)
```

```
##   TEST
##           actual
## predicted  NO YES
##       NO  303 168
##       YES  61  71
```

```
# Accuracy for test set
mean(imbal_test_1$Potability == predettoP.te)
```

```
## [1] 0.6202322
```

→ **Albero non potato 0.58**

Random Forest

Il Random Forest combina diversi alberi decisionali in una “foresta”, ciascuno dei quali viene addestrato su un sottoinsieme casuale dei dati di addestramento. L’output finale del modello -ottenuto mediante il processo di *Bagging*- è la media delle previsioni di tutti gli alberi (per la regressione) o voto di maggioranza(per la classificazione).
Durante la costruzione di ciascun albero decisionale, viene selezionato un sottoinsieme casuale(*mtry*) delle variabili predittive per la ricerca della migliore divisione in ogni nodo dell’albero.

Questo aiuta a rendere i singoli alberi meno correlati tra loro e quindi a ridurre la varianza complessiva del modello. Questo lo rende più adatto all’applicazione a una vasta gamma di dataset senza la necessità di complesse ottimizzazioni dei parametri.

Si osserva che quando la dimensione del sottoinsieme casuale delle variabili considerate (*mtry*) corrisponde al numero totale delle variabili predittive, il *Random Forest* corrisponde al *Bagging* .
In conclusione il *Random Forest* è noto per la sua robustezza, flessibilità e ottime prestazioni su molti tipi di dati e problemi di apprendimento automatico.

Random Forest

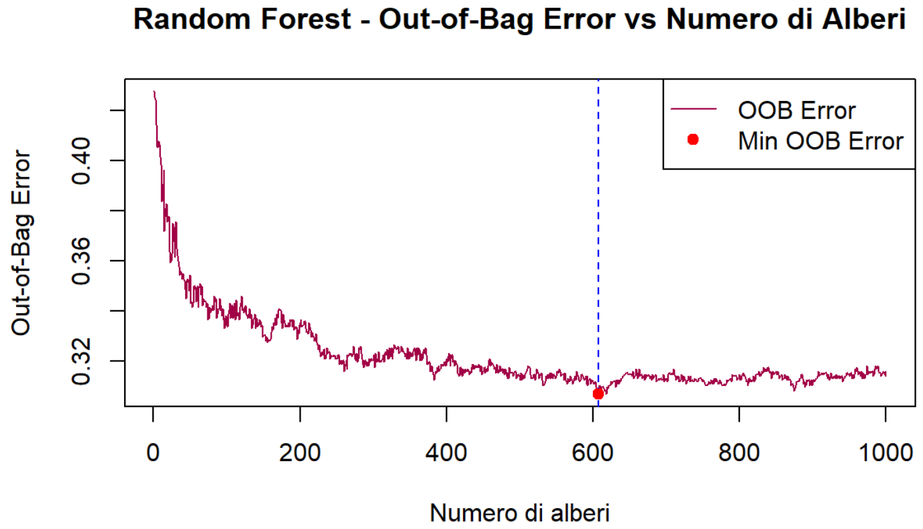
Abbiamo realizzato un primo tentativo utilizzando i dati di *train* , lasciando *mtry* ai valori di default(radqp=3) e impostando il numero di alberi pari a 1000. L'output ci da un' indicazione sui dettagli utilizzati per un modello. L' errore out-of-bag (OOB), è l'errore medio per ciascun campione bootstrap calcolato utilizzando le previsioni degli alberi che non contengono quel campione nel rispettivo campione bootstrap.

Si è calcolato infine l'*Accuracy* sul *test, dove si può osservare come il modello faccia fatica a predire quando l'acqua è potabile*.

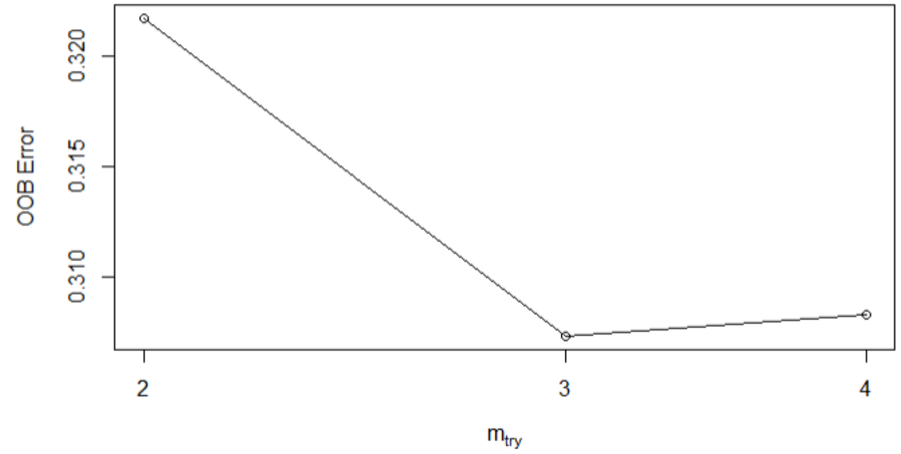
```
No. of variables tried at each split: 3

        OOB estimate of  error rate: 31.25%
Confusion matrix:
     NO YES class.error
NO  725 111   0.1327751
YES 329 243   0.5751748
[1] "OOB Accuracy: 0.6875"
Confusion Matrix and Statistics

          Reference
Prediction  NO YES
       NO  309 141
       YES  55  98

               Accuracy : 0.675
                 95% CI : (0.636, 0.7122)
    No Information Rate : 0.6036
    P-Value [Acc > NIR] : 0.0001728

                  Kappa : 0.276

 Mcnemar's Test P-Value : 1.268e-09

            Sensitivity : 0.8489
            Specificity : 0.4100
         Pos Pred Value : 0.6867
         Neg Pred Value : 0.6405
             Prevalence : 0.6036
         Detection Rate : 0.5124
   Detection Prevalence : 0.7463
      Balanced Accuracy : 0.6295

       'Positive' Class : NO
```

Random Forest

Il parametro *mtry* controlla il numero di variabili casuali considerate in ciascuna divisione del nodo durante la costruzione degli alberi nel *Random Forest*. Aumentando o diminuendo il valore di *mtry*, si influenza la complessità degli alberi e la correlazione tra di essi. È conveniente effettuare un' ottimizzazione dei parametri per ricondursi a un valore ottimale di parametri da utilizzare.



Misurare l'errore OOB per diversi valori di *mtry* consente di valutare come la variazione del numero di variabili considerate in ciascuna divisione influisce sull'accuratezza del modello.
In questo caso OOB risulta minimizzato per un numero di alberi pari a circa 600.



Il valore ottimale di mtry corrisponde al punto in cui l'errore OOB è minimo. Nel nostro caso, il valore di mtry che corrisponde all'errore OOB più basso è 3.

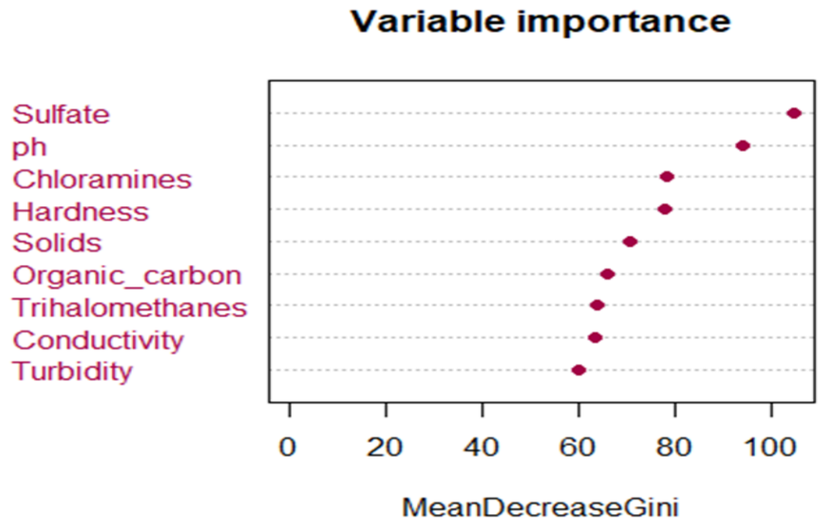
Random Forest

L'importanza delle variabili può essere valutata misurando quanto la loro presenza o assenza nelle divisioni dell'albero migliora la capacità predittiva complessiva del modello.
Il *Gini Impurity* è una misura di disordine o impurità all'interno di un insieme di dati.

In un contesto di classificazione binaria, il Gini *Impurity* di un nodo misura la probabilità che un elemento scelto casualmente dall'insieme sia etichettato in modo scorretto se venisse etichettato casualmente secondo la distribuzione delle etichette nel nodo.

Il "*mean decrease Gini*" è invece una misura dell'importanza delle variabili, che quantifica quanto la suddivisione delle variabili nelle divisioni degli alberi riduca il Gini impurity medio dei nodi.

```r
#Variable Importance
varImpPlot(Model.rf, main="Variable importance", pch = 19, color="#A20045")
```



Boosting

Il boosting: modello di machine learning per l'apprendimento automatico esamble. Il boosting addestra una serie di alberi in modo sequenziale, addestrati uno alla volta, e ogni albero successivo cerca di correggere gli errori dei modelli precedenti.
A differenza del Bagging o del Random forest, gli alberi del Boosting sono corti con alto bias e bassa varianza.

Vantaggi:

- Solitamente produce modelli più potenti, poiché ciascun modello successivo corregge gli errori dei modelli precedenti.

- Funziona bene con variabili categoriali e numeriche.

- Può ottenere prestazioni migliori rispetto a Random Forest su problemi più complessi.

Boosting

Il boosting è ampiamente utilizzato perché è molto efficace nell'affrontare una vasta gamma di problemi di apprendimento automatico e può produrre modelli molto accurati.

```
set.seed(11)
model_gbm_2 = gbm(Potability ~.,
                  data = imbal_train_1,
                  distribution = "multinomial",
                  shrinkage = 0.3,
                  interaction.depth = 3,
                  n.trees = 150)
model_gbm_2
```

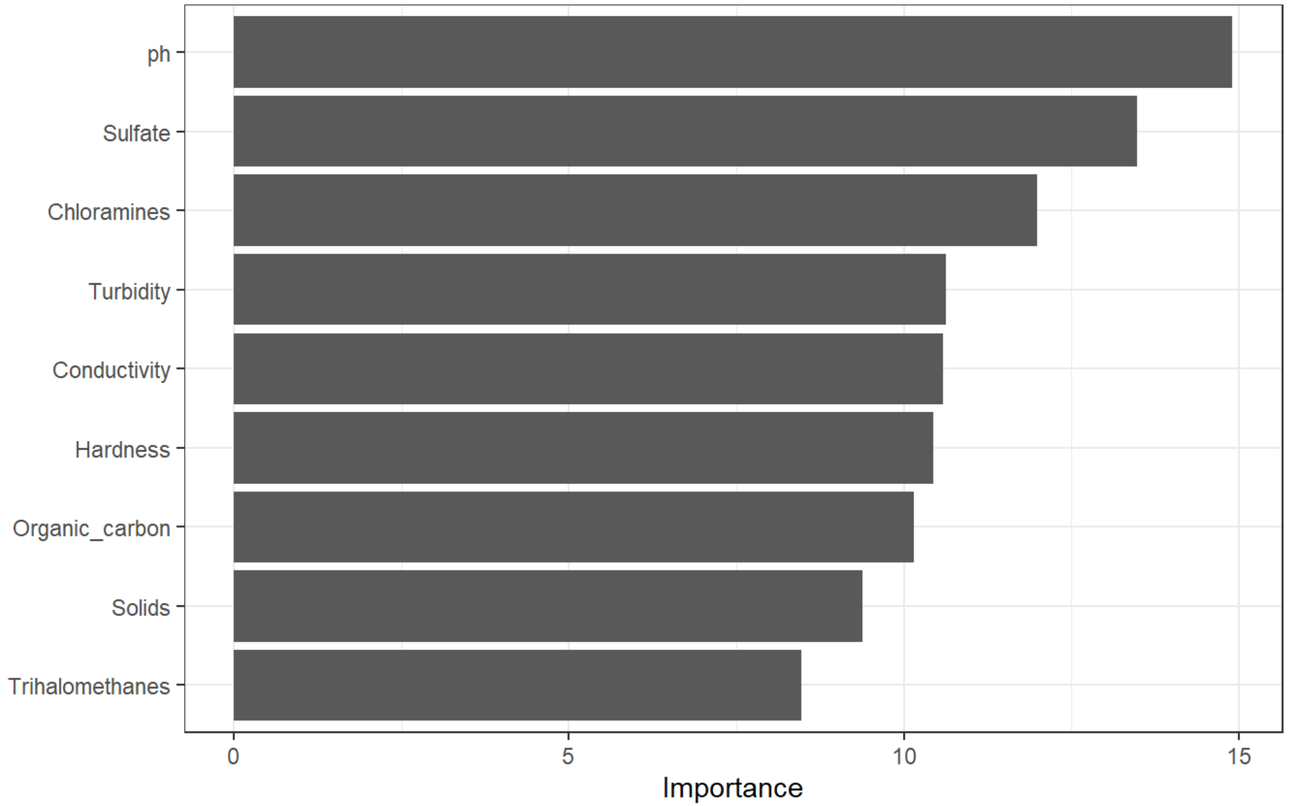
```
gbm(formula = Potability ~ ., distribution = "multinomial", data =
imbal_train_1,
    n.trees = 150, interaction.depth = 3, shrinkage = 0.3)
A gradient boosted model with multinomial loss function.
150 iterations were performed.
There were 9 predictors of which 9 had non-zero influence.
```

Boosting

```
Confusion Matrix and Statistics

          Reference
Prediction  NO YES
       NO  258 106
       YES 119 120

               Accuracy : 0.6269
                 95% CI : (0.5869, 0.6656)
    No Information Rate : 0.6252
    P-Value [Acc > NIR] : 0.4846

                  Kappa : 0.2129

 Mcnemar's Test P-Value : 0.4237

            Sensitivity : 0.6844
            Specificity : 0.5310
         Pos Pred Value : 0.7088
         Neg Pred Value : 0.5021
             Prevalence : 0.6252
         Detection Rate : 0.4279
   Detection Prevalence : 0.6036
      Balanced Accuracy : 0.6077

       'Positive' Class : NO
```

Feature importance:

```
library(vip)
vip::vip(model_gbm_2)+theme_bw()
```



Risultati e Conclusioni

La previsione sulla potabilità dell'acqua ha ottenuto risultati differenti rispetto ai diversi modelli di alberi decisionali applicati:

1)Cart:

Accuracy : 0.62 con intervallo di confidenza 95% (0.58, 0.66)

2) Random Forest:

Accuracy : 0.68 con intervallo di confidenza 95% (0.64, 0.71)

3)Boosting:

Accuracy : 0.63 con intervallo di confidenza 95% (0.59, 0.67)

Le features più influenti per questo problema di classificazione sono **ph** e **Sulfate.**