Indukcyjne metody analizy danych - Sprawozdanie z zajęć laboratoryjnych

April 15, 2020

Ćwiczenie 2. Indukcja drzew decyzyjnych C5.0 (C4.5) w R

Piotr Błoński 225959

Celem ćwiczenia było zapoznanie się z indukcjądrzew decyzyjnych C5.0 na platformie R. W tym celu wykorzystałem poniższe biblioteki:

[36]:
```
options(warn=-1) #Wyłączenie warningów aby w sprawozdaniu niebyły drukowane
library(tidyverse)
library(C50) #Drzewa decyzyjne C5.0
library(caret)#Pakiet do uczenia maszynowego
library(MLmetrics)# Pakiet zawierający metryki takie jak Fscore, Precision itp.
```

Należy zbudować model klasyfikatora na zbiorach danych: iris, wine, glass, seeds. Razem ze sprawozdaniem dołączone są pliki csv zawierające te zbiory.

[2]:
```
iris_data = read.csv(file = "iris.csv") #załaduj do iris_data dane Iris.
head(iris_data,3) # 3 pierwsze rekordy z datasetu
```

	Id <int>	SepalLengthCm <dbl>	SepalWidthCm <dbl>	PetalLengthCm <dbl>	PetalWidthCm <dbl>	Species <fct>
1	1	5.1	3.5	1.4	0.2	Iris-setosa
2	2	4.9	3.0	1.4	0.2	Iris-setosa
3	3	4.7	3.2	1.3	0.2	Iris-setosa

Jak widać dane Iris posiadają kolumne z klasą - Species która nas interesuje, w R jest faktorem. Absolutnie nie interesuje nas kolumna Id ponieważ od Id nie ma żadnego wpływu na to jakiego gatunku rośliną będzie dany Irys. Będzie trzeba w procesie uczenia pominąć tą kolumne.

[38]:
```
wine_data = read.csv(file = "wine.csv") #załaduj do wine_data dane Wine Quality
head(wine_data,3)
```

	fixed acidity <dbl>	volatile acidity <dbl>	citric acid <dbl>	residual sugar <dbl>	chlorides <dbl>	free sulfur dioxide <dbl>	total sulfur dioxide <dbl>	density <dbl>	pH <dbl>	sulphates <dbl>	alcohol <dbl>	quality <int>
1	7.4	0.70	0.00	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
2	7.8	0.88	0.00	2.6	0.098	25	67	0.9968	3.20	0.68	9.8	5
3	7.8	0.76	0.04	2.3	0.092	15	54	0.9970	3.26	0.65	9.8	5

[4]:
```
glass_data = read.csv(file = "glass.csv") #załąduj do glass_data dane Glass
head(glass_data,3)
```

	RI <dbl>	Na <dbl>	Mg <dbl>	Al <dbl>	Si <dbl>	K <dbl>	Ca <dbl>	Ba <dbl>	Fe <dbl>	Type <int>
1	1.52101	13.64	4.49	1.10	71.78	0.06	8.75	0	0	1
2	1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0	0	1
3	1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0	0	1

W danych glass i wine ostatnie kolumny niestety mają w R typ Int co powoduje błędy! Typu jak w [1]. Aby tego uniknąć w funkcji uczące będziemy przerabiać tą kolumne na factor.

[5]:
```
seed_data = read.csv(file = "seeds.csv") #załaduj do seed_data dane Seeds
head(seed_data,3)
```

	Area <dbl>	Perimeter <dbl>	compactness <dbl>	length of kernel <dbl>	width of kernel <dbl>	asymmetry coefficient <dbl>	lenght of kernel groove <dbl>	Type <int>
1	15.26	14.84	0.8710	5.763	3.312	2.221	5.220	1
2	14.88	14.57	0.8811	5.554	3.333	1.018	4.956	1
3	14.29	14.09	0.9050	5.291	3.337	2.699	4.825	1

W informacjach o danych Seeds, jest napisane że compactness jest zależne od Area i Perimeter jako: - compactness = $4 * pi * Area/Perimeter^2$
Dlatego też w procesie uczenia nie będziemy korzystać z tej kolumny za pomocą poniższej lini:

[6]:
```
column_to_drop <- c("compactness")
seed_data <- seed_data[ , !(names(seed_data) %in% column_to_drop)]
```

W trakcie wykonywania crossvalidacji za pomocą biblioteki caret okazało się że jedyne metryki jakie nam zwraca to Accuracy i Kappa. A interesują nas Fscore, Accuracy, Precision i Recall. W tym celu zgodnie z dokumentacja biblioteki caret [2], stworzyłem własną funkcje z metrką.

[7]:
```
metrics <- function(data, lev = NULL, model = NULL)
{
  f1_val <- F1_Score(y_pred = data$pred, y_true = data$obs, positive = lev[1])
  rec_val <-Recall(y_pred = data$pred, y_true = data$obs, positive = lev[1])
  sen_val = Sensitivity(y_pred = data$pred, y_true = data$obs, positive = lev[1])
  pre_val = Precision(y_pred = data$pred, y_true = data$obs, positive = lev[1])
  acc_val = Accuracy(y_pred = data$pred, y_true = data$obs)
  c(fScore = f1_val,Recall = rec_val, Sensitivity = sen_val, Precision =␣
 ↪pre_val,Accuracy=acc_val)
}
```

Funkcja ta jest używana podczas caret'owego train w celu walidacji modelu. Przy użyciu caretowej crossvalidacji powinna sie wykonywać dla wszystkich foldów i zwrócić wartość średnią.Funkcja ta wykorzystuje funkcje metryki dostępne z biblioteki MLmetrics.

Funkcja odpowiedzialna za uczenie modelu naszego drzewa decyzyjnego zwraca metryki i pomiary na modelu jako dataframe. Przymuje następujące parametry: - Param-names - jest to string który zawiera informacje jakie parametry modelu będzie posiadało nasze drzewo decyzyjne. Ten parametr używany jest tylko i wyłącznie jako nazwa identyfikacyjna pomiaru. - dataset - np iris_data - dataframe którym będziemy uczyć. - model_type - 'tree' lub 'rules' - parametr przekazywany do funkcji train w celu wybrania czy drzewo ma być rule-based czy nie. - straing_col_number - kolumna startowa którą od której chcemy zacząć podawać dane. Najczęściej 1 w przypadku gdyby jednak kolumna Id była jako pierwsza (np Iris) to można ustawić np 2. Ten parametr pojawił się tylko i wyłącznie dlatego że w trakcie pisania tego skryptu dość późno nauczyłem się dropować kolumny. - last_col_number - ostatnia kolumna - formula - formuła R np 'Species ~.' przekazywana do train - folds - ilość foldów w crossvalidacji.
Parametry opisane pod kodem: - Winnowing - parametr True / False - jest odpowiedzialny za to czy powinna zostać użyta feature selection. - Fuzzy - pamater True / False - odpowiedzialny za fuzzyThreshold - GlobalPruning - parametr True / False - odpowiedzialny jest za przycinanie końcowe drzewa w celu jego uproszczenia. Nazwa parametru to w rzeczywistości noGlobablPruning więc zaznaczenia na True oznacze brak przycinania.

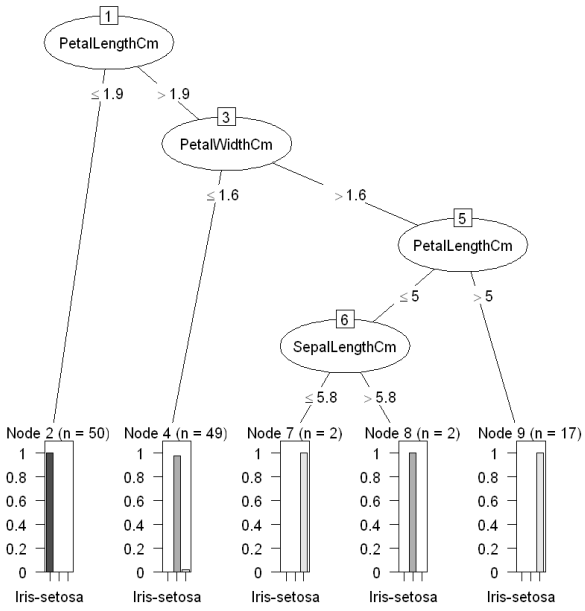
[41]:
```
TreeModel_caret <-␣
 ↪function(param_names,dataset,model_type,starting_col_number,last_col_number,formula,folds,win
){
    #Selekcja danych
    test = dataset$last_col_number
    dataset[,ncol(dataset)] = as.factor(dataset[,ncol(dataset)])
    y = dataset[,(last_col_number-1)] #class column
    index = createDataPartition(y=y, p=0.7, list=FALSE)
    train.set = dataset[index,starting_col_number:last_col_number]
    test.set = dataset[-index,starting_col_number:last_col_number]
    #Ustawianie parametrów i Control
    train.control <- trainControl(#https://www.rdocumentation.org/packages/C50/
 ↪versions/0.1.3/topics/C5.0Control
                  method = "cv",
                  number = folds,
                  savePredictions = "all",
                  summaryFunction = metrics)

    Control <- C5.0Control(
                  winnow = winnowing,
                  fuzzyThreshold = fuzzy,
                  noGlobalPruning = GlobalPruning)
    #uczenie
    tree <- train(
                  formula,
                  data=train.set,
                  method="C5.0",
                  control = Control,
                  tuneGrid = data.frame(trials = 1, model = c(model_type),␣
 ↪winnow = winnowing),
```

```
                    trControl = train.control)
    #wyciąganie metryk z results
    f1 = tree$results$fScore
    rec = tree$results$Recall
    sen = tree$results$Sensitivity
    acc = tree$results$Accuracy
    prec= tree$results$Precision
    size = tree$finalModel$size
    # przygotowanie danych do zwrócenia
    research_frame<-data.frame(param_names,f1,acc,rec,prec,size)
    names(research_frame)<-c("params","f1","acc","rec","prec","Tree_size")
    return(research_frame)
}
```

Najpierw jednak pokaże przykładowe ploty drzew decyzyjnych. Niestety wszystkich drzew nie moge umieścić w tym sprawozdaniu gdyż było by ich ponad 50.

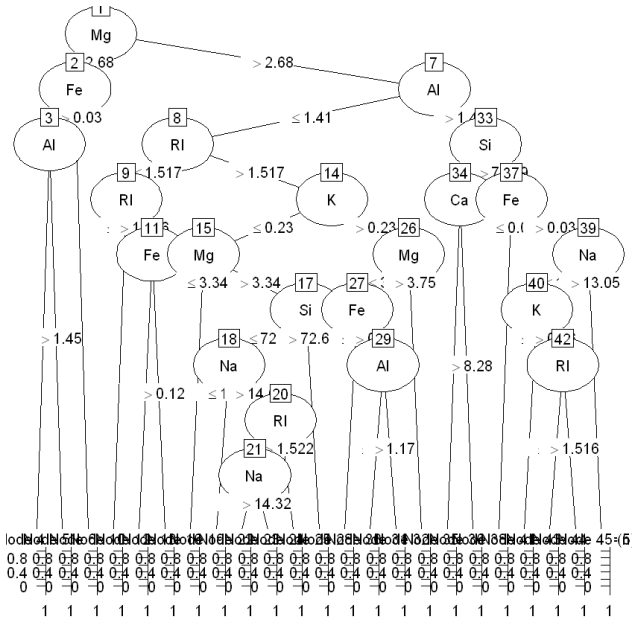
[42]:
```
in_train <- as.factor(sample(1:nrow(iris_data), size = (0.8*nrow(iris_data))))
train_data <- iris_data[ in_train,]
test_data  <- iris_data[-in_train,]
tree_mod_iris <- C5.0(x = train_data[, 2:5], y = train_data[,6])
plot(tree_mod_iris)
print(tree_mod_iris$size)
```

```
[1] 5
```



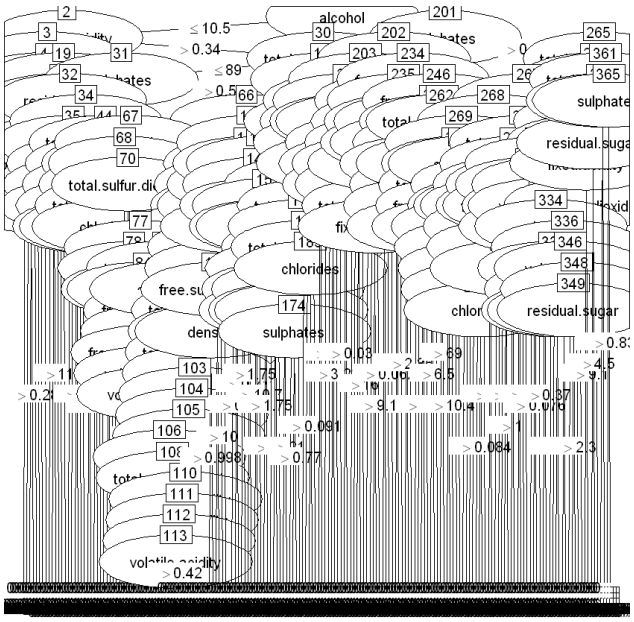
[43]:
```
in_train <- as.factor(sample(1:nrow(glass_data), size = (0.8*nrow(glass_data))))
glass_data[,ncol(glass_data)] <- as.factor(glass_data[,ncol(glass_data)])
train_data <- glass_data[ in_train,]
test_data  <- glass_data[-in_train,]
tree_mod_glass <- C5.0(x = train_data[, 1:9], y = train_data[,10])
plot(tree_mod_glass)
print(tree_mod_glass$size)
```

```
[1] 23
```



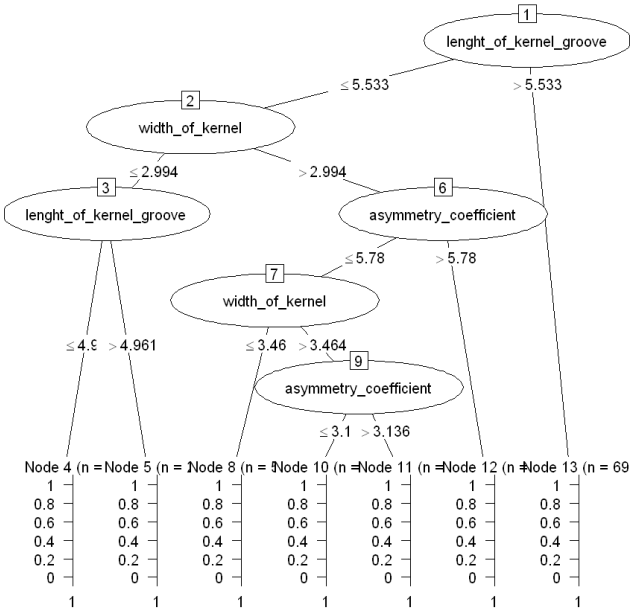
[44]:
```
in_train <- as.factor(sample(1:nrow(wine_data), size = (0.8*nrow(wine_data))))
wine_data[,ncol(wine_data)] <- as.factor(wine_data[,ncol(wine_data)])
train_data <- wine_data[ in_train,]
test_data  <- wine_data[-in_train,]
tree_mod_wine <- C5.0(x = train_data[, 1:11], y = train_data[,12])
plot(tree_mod_wine)
print(tree_mod_wine$size)
```

```
[1] 184
```



[45]:
```
in_train <- as.factor(sample(1:nrow(seed_data), size = (0.8*nrow(seed_data))))
seed_data[,ncol(seed_data)] <- as.factor(seed_data[,ncol(seed_data)])
train_data <- seed_data[ in_train,]
test_data  <- seed_data[-in_train,]
tree_mod_seed <- C5.0(x = train_data[, 1:6], y = train_data[,7])
plot(tree_mod_seed)
print(tree_mod_seed$size)
```

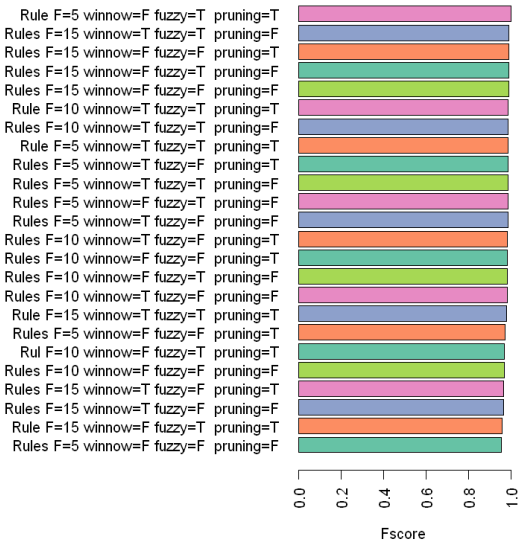
```
[1] 7
```

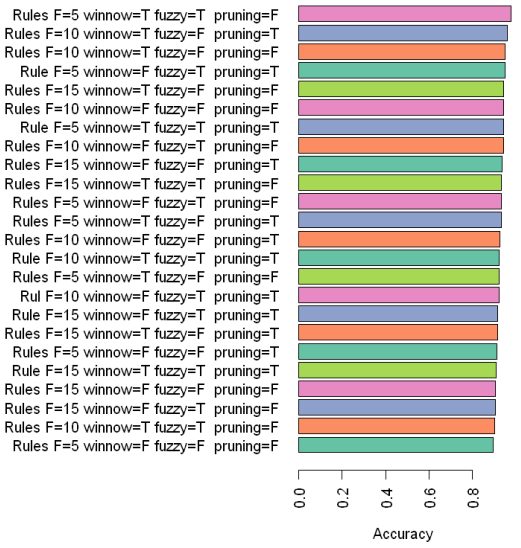


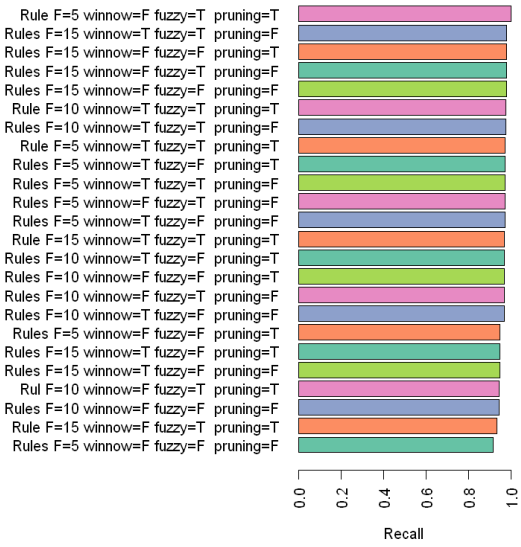
Plan badania: Dla każdego z foldów (5 , 10 , 15) sprawdzić wartość metryk i każdej kombinacji parametrów (winnow,noGlobalPunning,fuzzyThreshold.

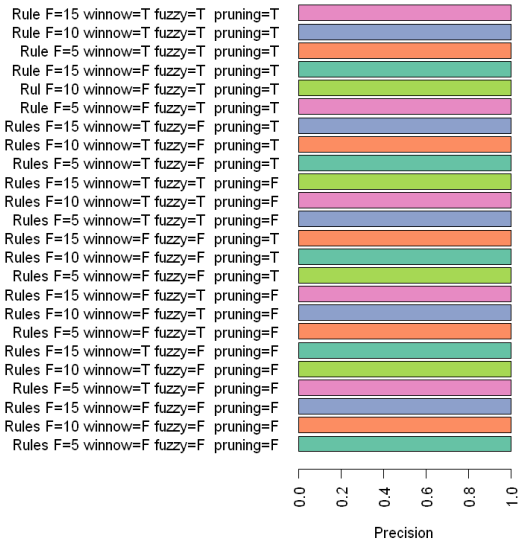
[47]:
```
print('Iris')
research(iris_data,2,6,Species~.)
```

```
[1] "Iris"
```

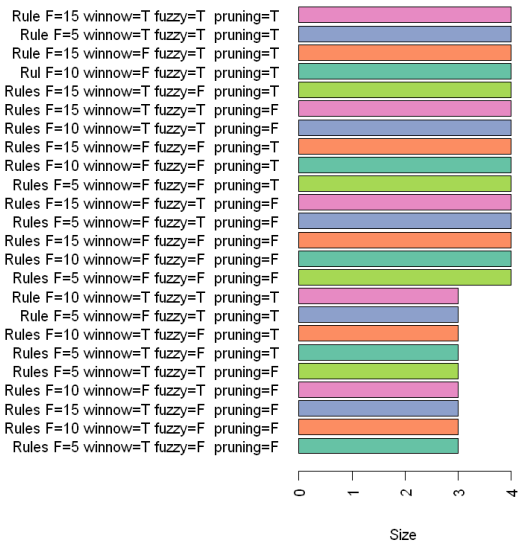





Rule F=5 winnow=F fuzzy=T pruning=T
Rules F=15 winnow=T fuzzy=T pruning=F
Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=T pruning=F
Rules F=15 winnow=F fuzzy=F pruning=F
Rule F=10 winnow=T fuzzy=T pruning=T
Rules F=10 winnow=T fuzzy=T pruning=F
Rule F=5 winnow=T fuzzy=T pruning=T
Rules F=5 winnow=T fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=F pruning=F
Rule F=15 winnow=T fuzzy=T pruning=T
Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=T pruning=F
Rules F=10 winnow=T fuzzy=F pruning=F
Rules F=5 winnow=F fuzzy=F pruning=T
Rules F=15 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=T fuzzy=F pruning=F
Rul F=10 winnow=F fuzzy=T pruning=T
Rules F=10 winnow=F fuzzy=F pruning=F
Rule F=15 winnow=F fuzzy=T pruning=T
Rules F=5 winnow=F fuzzy=F pruning=F
0.0
0.2
0.4
0.6
0.8
1.0
Recall

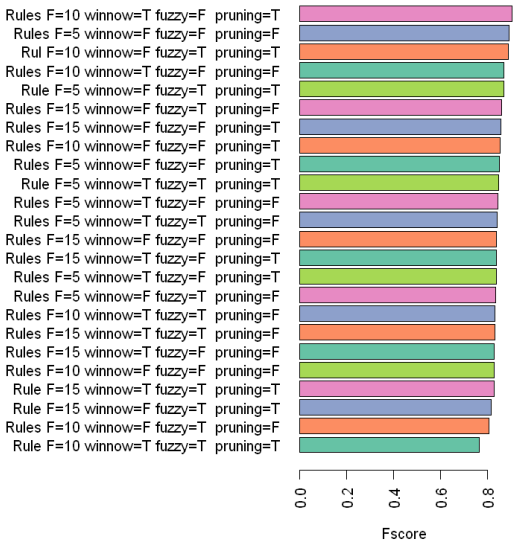

Rule F=15 winnow=T fuzzy=T pruning=T
Rule F=10 winnow=T fuzzy=T pruning=T
Rule F=5 winnow=T fuzzy=T pruning=T
Rule F=15 winnow=F fuzzy=T pruning=T
Rul F=10 winnow=F fuzzy=T pruning=T
Rule F=5 winnow=F fuzzy=T pruning=T
Rules F=15 winnow=T fuzzy=F pruning=T
Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=T fuzzy=T pruning=F
Rules F=10 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=T pruning=F
Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=F pruning=T
Rules F=5 winnow=F fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=T pruning=F
Rules F=10 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=F fuzzy=T pruning=F
Rules F=15 winnow=T fuzzy=F pruning=F
Rules F=10 winnow=T fuzzy=F pruning=F
Rules F=5 winnow=T fuzzy=F pruning=F
Rules F=15 winnow=F fuzzy=F pruning=F
Rules F=10 winnow=F fuzzy=F pruning=F
Rules F=5 winnow=F fuzzy=F pruning=F
0.0
0.2
0.4
0.6
0.8
1.0
Precision

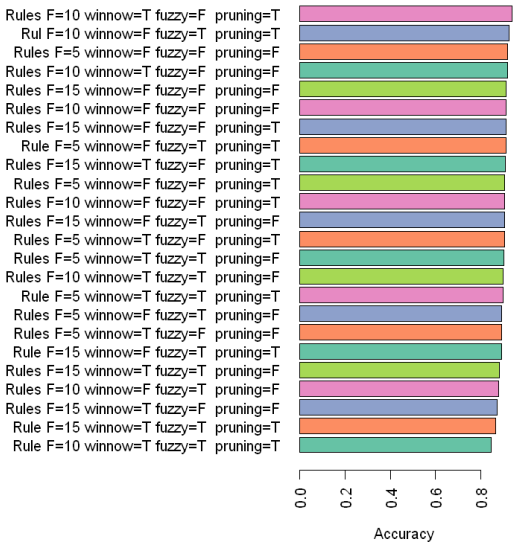
	params <fct>	f1 <dbl>	acc <dbl>	rec <dbl>	prec <dbl>	Tree_size <int>
4	Rules F=5 winnow=T fuzzy=F pruning=F	0.9846154	0.9242424	0.9714286	1	3
5	Rules F=10 winnow=T fuzzy=F pruning=F	0.9800000	0.9531313	0.9666667	1	3
6	Rules F=15 winnow=T fuzzy=F pruning=F	0.9644444	0.9452381	0.9444444	1	3
8	Rules F=10 winnow=F fuzzy=T pruning=F	0.9800000	0.9432828	0.9666667	1	3
13	Rules F=5 winnow=T fuzzy=T pruning=F	0.9846154	0.9809524	0.9714286	1	3
16	Rules F=5 winnow=T fuzzy=F pruning=T	0.9846154	0.9332035	0.9714286	1	3
17	Rules F=10 winnow=T fuzzy=F pruning=T	0.9800000	0.9609091	0.9666667	1	3
19	Rule F=5 winnow=F fuzzy=T pruning=T	1.0000000	0.9513853	1.0000000	1	3
23	Rule F=10 winnow=T fuzzy=T pruning=T	0.9857143	0.9249495	0.9750000	1	3
1	Rules F=5 winnow=F fuzzy=F pruning=F	0.9512821	0.8964502	0.9142857	1	4
2	Rules F=10 winnow=F fuzzy=F pruning=F	0.9657143	0.9442424	0.9416667	1	4
3	Rules F=15 winnow=F fuzzy=F pruning=F	0.9866667	0.9079365	0.9777778	1	4
7	Rules F=5 winnow=F fuzzy=T pruning=F	0.9846154	0.9337662	0.9714286	1	4
9	Rules F=15 winnow=F fuzzy=T pruning=F	0.9866667	0.9075397	0.9777778	1	4
10	Rules F=5 winnow=F fuzzy=F pruning=T	0.9712821	0.9150216	0.9464286	1	4
11	Rules F=10 winnow=F fuzzy=F pruning=T	0.9800000	0.9268182	0.9666667	1	4
12	Rules F=15 winnow=F fuzzy=F pruning=T	0.9866667	0.9388889	0.9777778	1	4
14	Rules F=10 winnow=T fuzzy=T pruning=F	0.9857143	0.9040404	0.9750000	1	4
15	Rules F=15 winnow=T fuzzy=T pruning=F	0.9866667	0.9341270	0.9777778	1	4
18	Rules F=15 winnow=T fuzzy=F pruning=T	0.9644444	0.9161376	0.9444444	1	4
20	Rul F=10 winnow=F fuzzy=T pruning=T	0.9657143	0.9239899	0.9416667	1	4
21	Rule F=15 winnow=F fuzzy=T pruning=T	0.9555556	0.9162698	0.9333333	1	4
22	Rule F=5 winnow=T fuzzy=T pruning=T	0.9846154	0.9441558	0.9714286	1	4
24	Rule F=15 winnow=T fuzzy=T pruning=T	0.9777778	0.9105820	0.9666667	1	4

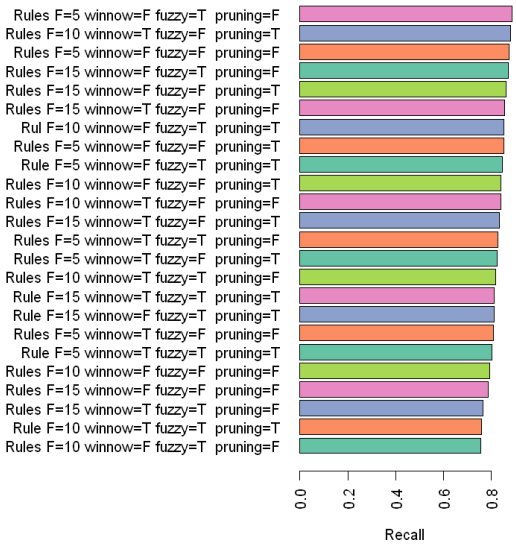


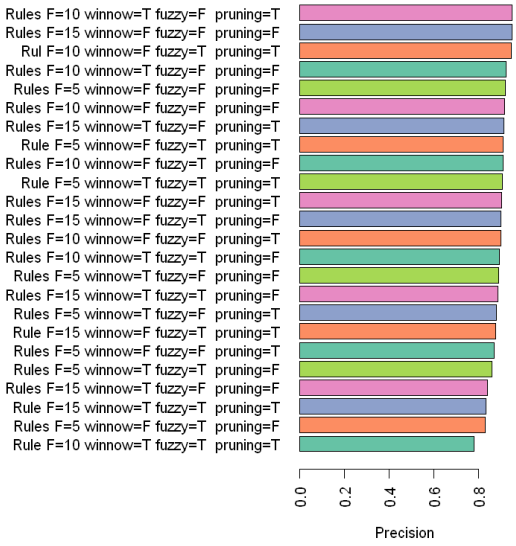
[48]:
```
print('Seeds')
research(seed_data,1,7,Type~.)
```

```
[1] "Seeds"
```

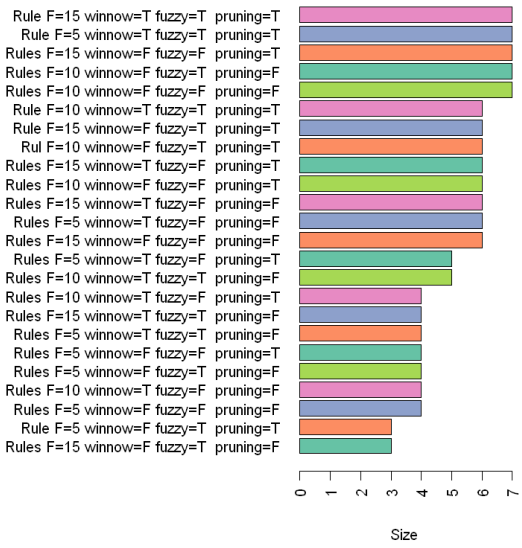





Rules F=5 winnow=F fuzzy=T pruning=F
Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=5 winnow=F fuzzy=F pruning=F
Rules F=15 winnow=F fuzzy=T pruning=F
Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=15 winnow=T fuzzy=F pruning=F
Rul F=10 winnow=F fuzzy=T pruning=T
Rules F=5 winnow=F fuzzy=F pruning=T
Rule F=5 winnow=F fuzzy=T pruning=T
Rules F=10 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=T fuzzy=F pruning=F
Rules F=15 winnow=T fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=F pruning=T
Rules F=10 winnow=T fuzzy=T pruning=F
Rule F=15 winnow=T fuzzy=T pruning=T
Rule F=15 winnow=F fuzzy=T pruning=T
Rules F=5 winnow=T fuzzy=F pruning=F
Rule F=5 winnow=T fuzzy=T pruning=T
Rules F=10 winnow=F fuzzy=F pruning=F
Rules F=15 winnow=F fuzzy=F pruning=F
Rules F=15 winnow=T fuzzy=T pruning=F
Rule F=10 winnow=T fuzzy=T pruning=T
Rules F=10 winnow=F fuzzy=T pruning=F
0.0
0.2
0.4
0.6
0.8
Recall

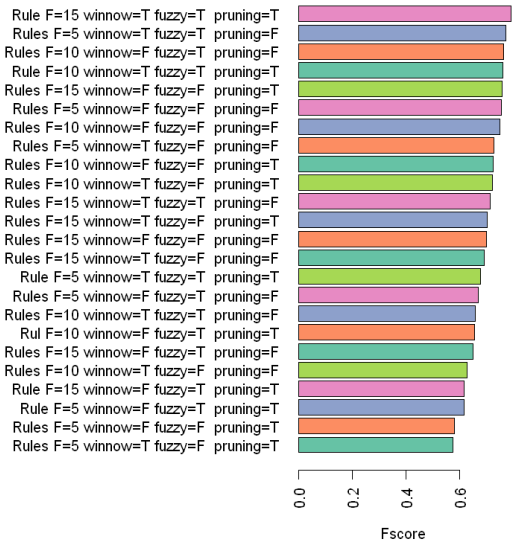

Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=F pruning=F
Rul F=10 winnow=F fuzzy=T pruning=T
Rules F=10 winnow=T fuzzy=F pruning=F
Rules F=5 winnow=F fuzzy=F pruning=F
Rules F=10 winnow=F fuzzy=F pruning=F
Rules F=15 winnow=T fuzzy=F pruning=T
Rule F=5 winnow=F fuzzy=T pruning=T
Rules F=10 winnow=F fuzzy=T pruning=F
Rule F=5 winnow=T fuzzy=T pruning=T
Rules F=15 winnow=F fuzzy=T pruning=T
Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=F pruning=F
Rules F=15 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=F pruning=T
Rule F=15 winnow=F fuzzy=T pruning=T
Rules F=5 winnow=F fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=T pruning=F
Rules F=15 winnow=T fuzzy=F pruning=F
Rule F=15 winnow=T fuzzy=T pruning=T
Rules F=5 winnow=F fuzzy=T pruning=F
Rule F=10 winnow=T fuzzy=T pruning=T
0.0
0.2
0.4
0.6
0.8
Precision

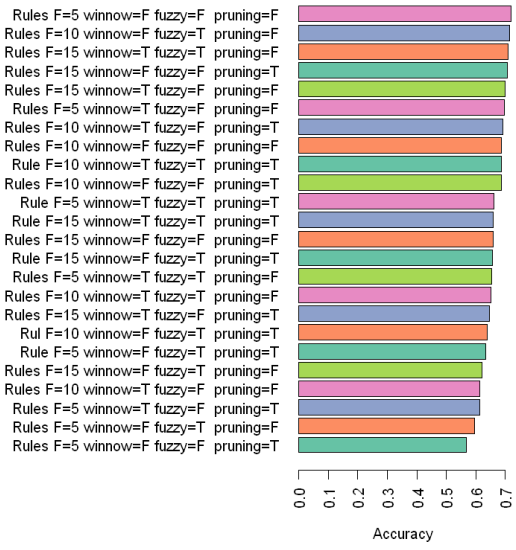
	params <fct>	f1 <dbl>	acc <dbl>	rec <dbl>	prec <dbl>	Tree_size <int>
9	Rules F=15 winnow=F fuzzy=T pruning=F	0.8593651	0.9056566	0.8722222	0.8996825	3
19	Rule F=5 winnow=F fuzzy=T pruning=T	0.8698574	0.9126437	0.8472727	0.9088267	3
1	Rules F=5 winnow=F fuzzy=F pruning=F	0.8910649	0.9195106	0.8757576	0.9189610	4
5	Rules F=10 winnow=T fuzzy=F pruning=F	0.8701515	0.9185119	0.8400000	0.9216667	4
7	Rules F=5 winnow=F fuzzy=T pruning=F	0.8338817	0.8927697	0.8888889	0.8277778	4
10	Rules F=5 winnow=F fuzzy=F pruning=T	0.8506366	0.9057471	0.8533333	0.8699134	4
13	Rules F=5 winnow=T fuzzy=T pruning=F	0.8393464	0.9010604	0.8288889	0.8578283	4
15	Rules F=15 winnow=T fuzzy=T pruning=F	0.8304989	0.8824579	0.7666667	0.8845238	4
17	Rules F=10 winnow=T fuzzy=F pruning=T	0.9053030	0.9402976	0.8800000	0.9500000	4
14	Rules F=10 winnow=T fuzzy=T pruning=F	0.8311328	0.8994048	0.8200000	0.8916667	5
16	Rules F=5 winnow=T fuzzy=F pruning=T	0.8363280	0.9050278	0.8244444	0.8784848	5
3	Rules F=15 winnow=F fuzzy=F pruning=F	0.8386243	0.9136700	0.7888889	0.9477778	6
4	Rules F=5 winnow=T fuzzy=F pruning=F	0.8424031	0.8923693	0.8109091	0.8886364	6
6	Rules F=15 winnow=T fuzzy=F pruning=F	0.8279365	0.8720539	0.8555556	0.8377778	6
11	Rules F=10 winnow=F fuzzy=F pruning=T	0.8540404	0.9057143	0.8400000	0.8988095	6
18	Rules F=15 winnow=T fuzzy=F pruning=T	0.8385714	0.9088552	0.8333333	0.9122222	6
20	Rul F=10 winnow=F fuzzy=T pruning=T	0.8876612	0.9254625	0.8533333	0.9440476	6
21	Rule F=15 winnow=F fuzzy=T pruning=T	0.8140212	0.8914815	0.8111111	0.8766667	6
23	Rule F=10 winnow=T fuzzy=T pruning=T	0.7634343	0.8461905	0.7600000	0.7783333	6
2	Rules F=10 winnow=F fuzzy=F pruning=F	0.8273810	0.9131548	0.7950000	0.9150000	7
8	Rules F=10 winnow=F fuzzy=T pruning=F	0.8046032	0.8794048	0.7550000	0.9085714	7
12	Rules F=15 winnow=F fuzzy=F pruning=T	0.8559788	0.9127946	0.8611111	0.9011111	7
22	Rule F=5 winnow=T fuzzy=T pruning=T	0.8462932	0.8993103	0.8036364	0.9054545	7
24	Rule F=15 winnow=T fuzzy=T pruning=T	0.8262472	0.8652525	0.8111111	0.8321429	7

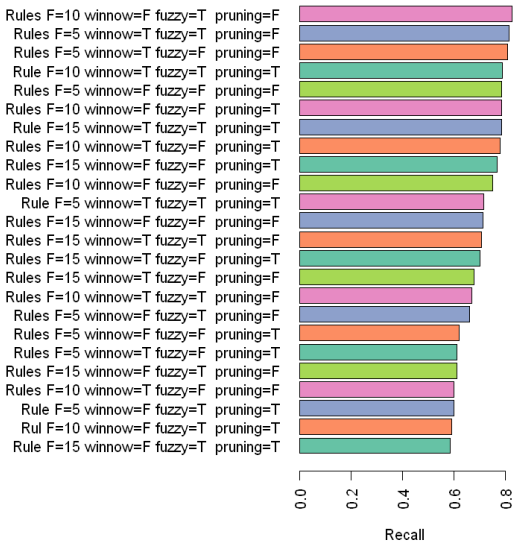


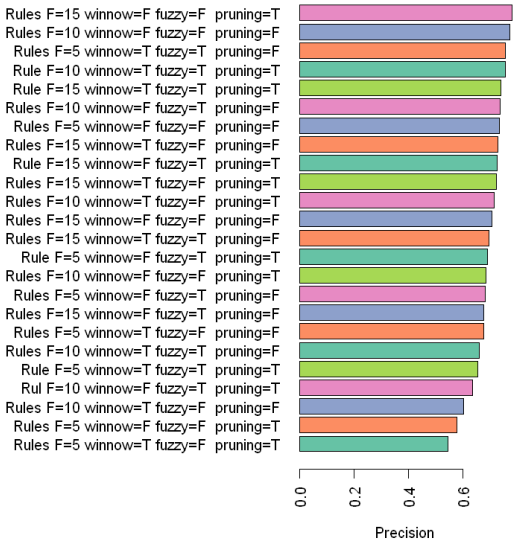
[49]:
```
print('Glass')
research(glass_data,1,10,Type~. )
```

```
[1] "Glass"
```

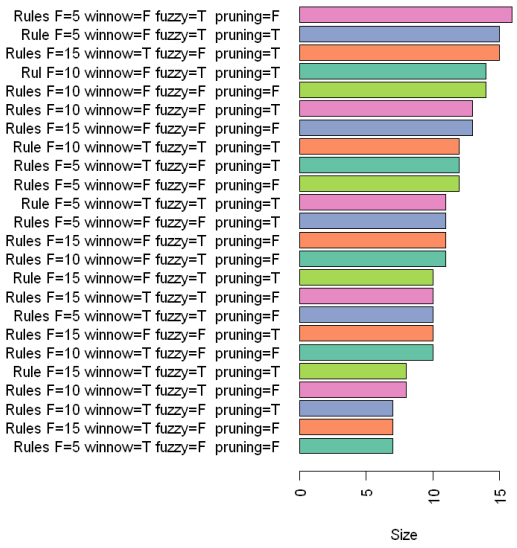





Rules F=10 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=F pruning=F
Rule F=10 winnow=T fuzzy=T pruning=T
Rules F=5 winnow=F fuzzy=F pruning=F
Rules F=10 winnow=F fuzzy=F pruning=T
Rule F=15 winnow=T fuzzy=T pruning=T
Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=F pruning=F
Rule F=5 winnow=T fuzzy=T pruning=T
Rules F=15 winnow=F fuzzy=F pruning=F
Rules F=15 winnow=T fuzzy=T pruning=F
Rules F=15 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=T fuzzy=F pruning=F
Rules F=10 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=F fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=T pruning=F
Rules F=10 winnow=T fuzzy=F pruning=F
Rule F=5 winnow=F fuzzy=T pruning=T
Rul F=10 winnow=F fuzzy=T pruning=T
Rule F=15 winnow=F fuzzy=T pruning=T
0.0
0.2
0.4
0.6
0.8
Recall

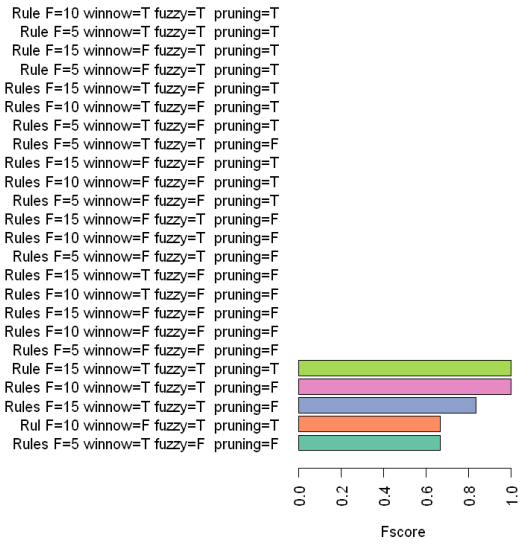

Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=F pruning=F
Rules F=5 winnow=T fuzzy=T pruning=F
Rule F=10 winnow=T fuzzy=T pruning=T
Rule F=15 winnow=T fuzzy=T pruning=T
Rules F=10 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=F fuzzy=F pruning=F
Rules F=15 winnow=T fuzzy=F pruning=F
Rule F=15 winnow=F fuzzy=T pruning=T
Rules F=15 winnow=T fuzzy=F pruning=T
Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=F pruning=F
Rules F=15 winnow=T fuzzy=T pruning=F
Rule F=5 winnow=F fuzzy=T pruning=T
Rules F=10 winnow=F fuzzy=F pruning=T
Rules F=5 winnow=F fuzzy=T pruning=F
Rules F=15 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=F pruning=F
Rules F=10 winnow=T fuzzy=T pruning=F
Rule F=5 winnow=T fuzzy=T pruning=T
Rul F=10 winnow=F fuzzy=T pruning=T
Rules F=10 winnow=T fuzzy=F pruning=F
Rules F=5 winnow=F fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=F pruning=T
0.0
0.2
0.4
0.6
Precision

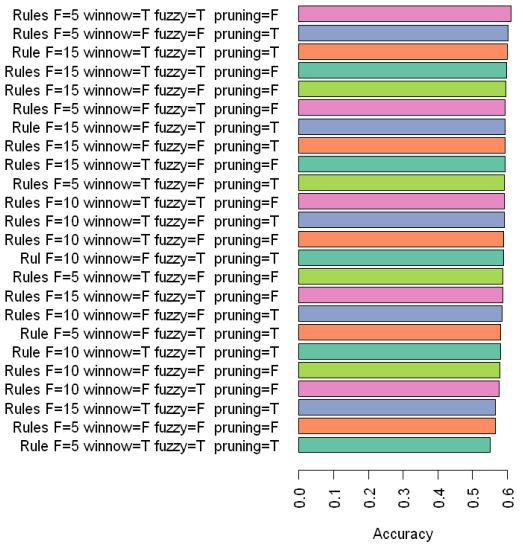
	params <fct>	f1 <dbl>	acc <dbl>	rec <dbl>	prec <dbl>	Tree_size <int>
4	Rules F=5 winnow=T fuzzy=F pruning=F	0.7277391	0.6966550	0.8090909	0.6773232	7
6	Rules F=15 winnow=T fuzzy=F pruning=F	0.6908466	0.6989562	0.6777778	0.7311111	7
17	Rules F=10 winnow=T fuzzy=F pruning=T	0.7227689	0.6904167	0.7800000	0.7168651	7
14	Rules F=10 winnow=T fuzzy=T pruning=F	0.6582900	0.6508899	0.6700000	0.6600000	8
24	Rule F=15 winnow=T fuzzy=T pruning=T	0.7930272	0.6597462	0.7833333	0.7411111	8
5	Rules F=10 winnow=T fuzzy=F pruning=F	0.6271605	0.6138130	0.6000000	0.6029101	10
12	Rules F=15 winnow=F fuzzy=F pruning=T	0.7571958	0.7077273	0.7666667	0.7833333	10
13	Rules F=5 winnow=T fuzzy=T pruning=F	0.7724211	0.6527253	0.8133333	0.7577778	10
15	Rules F=15 winnow=T fuzzy=T pruning=F	0.7127551	0.7087374	0.7055556	0.6976190	10
21	Rule F=15 winnow=F fuzzy=T pruning=T	0.6170899	0.6559790	0.5833333	0.7277778	10
8	Rules F=10 winnow=F fuzzy=T pruning=F	0.7645599	0.7145168	0.8266667	0.7376984	11
9	Rules F=15 winnow=F fuzzy=T pruning=F	0.6498299	0.6205387	0.6111111	0.6777778	11
10	Rules F=5 winnow=F fuzzy=F pruning=T	0.5800619	0.5665109	0.6200000	0.5800000	11
22	Rule F=5 winnow=T fuzzy=T pruning=T	0.6785590	0.6600371	0.7145455	0.6543323	11
1	Rules F=5 winnow=F fuzzy=F pruning=F	0.7567191	0.7206377	0.7854545	0.7340870	12
16	Rules F=5 winnow=T fuzzy=F pruning=T	0.5736916	0.6129403	0.6111111	0.5454545	12
23	Rule F=10 winnow=T fuzzy=T pruning=T	0.7613364	0.6869643	0.7866667	0.7571429	12
3	Rules F=15 winnow=F fuzzy=F pruning=F	0.7006803	0.6587205	0.7111111	0.7083333	13
11	Rules F=10 winnow=F fuzzy=F pruning=T	0.7255051	0.6858977	0.7850000	0.6847619	13
2	Rules F=10 winnow=F fuzzy=F pruning=F	0.7501515	0.6870238	0.7500000	0.7733333	14
20	Rul F=10 winnow=F fuzzy=T pruning=T	0.6558522	0.6379921	0.5900000	0.6353175	14
18	Rules F=15 winnow=T fuzzy=F pruning=T	0.7016402	0.6466667	0.7000000	0.7255556	15
19	Rule F=5 winnow=F fuzzy=T pruning=T	0.6166947	0.6323174	0.5977778	0.6907359	15
7	Rules F=5 winnow=F fuzzy=T pruning=F	0.6685046	0.5956322	0.6600000	0.6825758	16

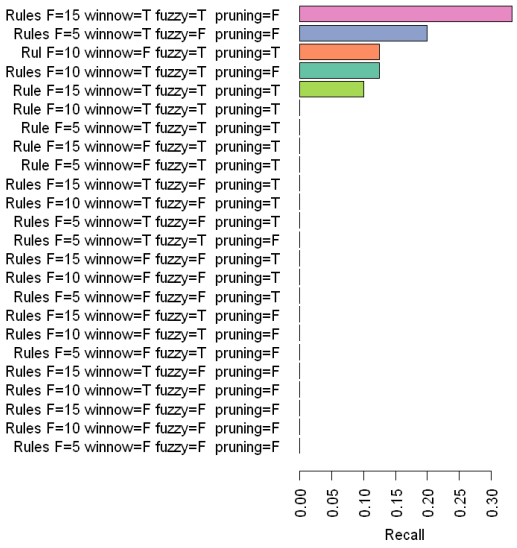


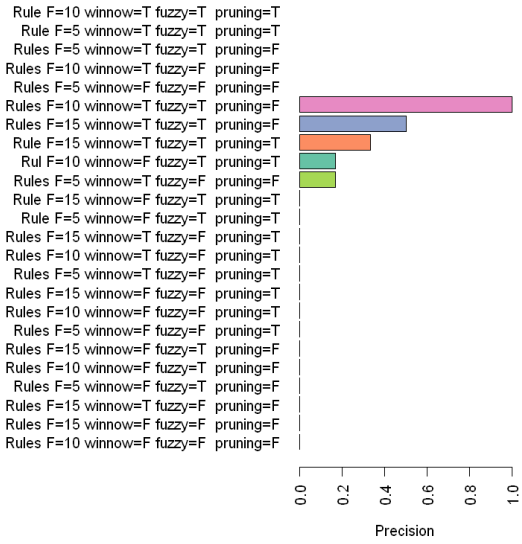
[50]:
```
print('Wine quality')
research(wine_data,1,12,quality~.)
```

```
[1] "Wine quality"
```

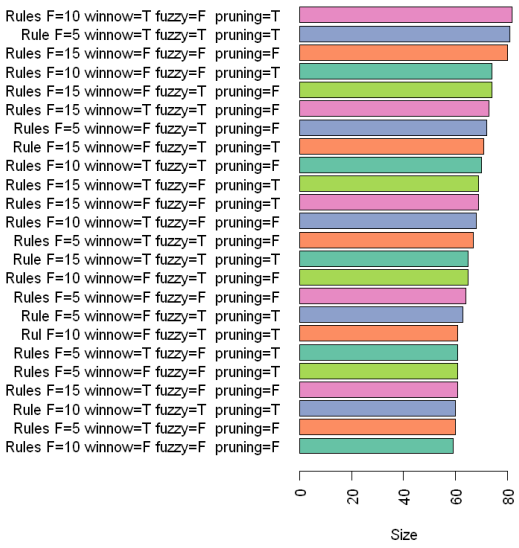





Rules F=15 winnow=T fuzzy=T pruning=F
Rules F=5 winnow=T fuzzy=F pruning=F
Rul F=10 winnow=F fuzzy=T pruning=T
Rules F=10 winnow=T fuzzy=T pruning=F
Rule F=15 winnow=T fuzzy=T pruning=T
Rule F=10 winnow=T fuzzy=T pruning=T
Rule F=5 winnow=T fuzzy=T pruning=T
Rule F=15 winnow=F fuzzy=T pruning=T
Rule F=5 winnow=F fuzzy=T pruning=T
Rules F=15 winnow=T fuzzy=F pruning=T
Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=T pruning=F
Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=F pruning=T
Rules F=5 winnow=F fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=T pruning=F
Rules F=10 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=F fuzzy=T pruning=F
Rules F=15 winnow=T fuzzy=F pruning=F
Rules F=10 winnow=T fuzzy=F pruning=F
Rules F=15 winnow=F fuzzy=F pruning=F
Rules F=10 winnow=F fuzzy=F pruning=F
Rules F=5 winnow=F fuzzy=F pruning=F
0.00
0.05
0.10
0.15
0.20
0.25
0.30
Recall


Rule F=10 winnow=T fuzzy=T pruning=T
Rule F=5 winnow=T fuzzy=T pruning=T
Rules F=5 winnow=T fuzzy=T pruning=F
Rules F=10 winnow=T fuzzy=F pruning=F
Rules F=5 winnow=F fuzzy=F pruning=F
Rules F=10 winnow=T fuzzy=T pruning=F
Rules F=15 winnow=T fuzzy=T pruning=F
Rule F=15 winnow=T fuzzy=T pruning=T
Rul F=10 winnow=F fuzzy=T pruning=T
Rules F=5 winnow=T fuzzy=F pruning=F
Rule F=15 winnow=F fuzzy=T pruning=T
Rule F=5 winnow=F fuzzy=T pruning=T
Rules F=15 winnow=T fuzzy=F pruning=T
Rules F=10 winnow=T fuzzy=F pruning=T
Rules F=5 winnow=T fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=F pruning=T
Rules F=10 winnow=F fuzzy=F pruning=T
Rules F=5 winnow=F fuzzy=F pruning=T
Rules F=15 winnow=F fuzzy=T pruning=F
Rules F=10 winnow=F fuzzy=T pruning=F
Rules F=5 winnow=F fuzzy=T pruning=F
Rules F=15 winnow=T fuzzy=F pruning=F
Rules F=15 winnow=F fuzzy=F pruning=F
Rules F=10 winnow=F fuzzy=F pruning=F
0.0
0.2
0.4
0.6
0.8
1.0
Precision

	params <fct>	f1 <dbl>	acc <dbl>	rec <dbl>	prec <dbl>	Tree_size <int>
2	Rules F=10 winnow=F fuzzy=F pruning=F	NaN	0.5782036	0.0000000	0.0000000	59
4	Rules F=5 winnow=T fuzzy=F pruning=F	0.6666667	0.5869362	0.2000000	0.1666667	60
23	Rule F=10 winnow=T fuzzy=T pruning=T	NaN	0.5808215	0.0000000	NaN	60
6	Rules F=15 winnow=T fuzzy=F pruning=F	NaN	0.5931423	0.0000000	0.0000000	61
10	Rules F=5 winnow=F fuzzy=F pruning=T	NaN	0.6022315	0.0000000	0.0000000	61
16	Rules F=5 winnow=T fuzzy=F pruning=T	NaN	0.5923418	0.0000000	0.0000000	61
20	Rul F=10 winnow=F fuzzy=T pruning=T	0.6666667	0.5895858	0.1250000	0.1666667	61
19	Rule F=5 winnow=F fuzzy=T pruning=T	NaN	0.5816187	0.0000000	0.0000000	63
1	Rules F=5 winnow=F fuzzy=F pruning=F	NaN	0.5656156	0.0000000	NaN	64
8	Rules F=10 winnow=F fuzzy=T pruning=F	NaN	0.5772336	0.0000000	0.0000000	65
24	Rule F=15 winnow=T fuzzy=T pruning=T	1.0000000	0.5993890	0.1000000	0.3333333	65
13	Rules F=5 winnow=T fuzzy=T pruning=F	NaN	0.6120293	0.0000000	NaN	67
5	Rules F=10 winnow=T fuzzy=F pruning=F	NaN	0.5896447	0.0000000	NaN	68
12	Rules F=15 winnow=F fuzzy=F pruning=T	NaN	0.5935395	0.0000000	0.0000000	69
18	Rules F=15 winnow=T fuzzy=F pruning=T	NaN	0.5664936	0.0000000	0.0000000	69
14	Rules F=10 winnow=T fuzzy=T pruning=F	1.0000000	0.5910375	0.1250000	1.0000000	70
21	Rule F=15 winnow=F fuzzy=T pruning=T	NaN	0.5939063	0.0000000	0.0000000	71
7	Rules F=5 winnow=F fuzzy=T pruning=F	NaN	0.5942742	0.0000000	0.0000000	72
15	Rules F=15 winnow=T fuzzy=T pruning=F	0.8333333	0.5974024	0.3333333	0.5000000	73
9	Rules F=15 winnow=F fuzzy=T pruning=F	NaN	0.5868618	0.0000000	0.0000000	74
11	Rules F=10 winnow=F fuzzy=F pruning=T	NaN	0.5860761	0.0000000	0.0000000	74
3	Rules F=15 winnow=F fuzzy=F pruning=F	NaN	0.5968108	0.0000000	0.0000000	80
22	Rule F=5 winnow=T fuzzy=T pruning=T	NaN	0.5513254	0.0000000	NaN	81
17	Rules F=10 winnow=T fuzzy=F pruning=T	NaN	0.5905675	0.0000000	0.0000000	82



1 Wnioski

1. Zawsze należy sprawdzać dany zbiór danych przez jego użyciem.
2. Drzewa decyzyjne można wizualizować przez co nie działają jak blackbox(np sieci NN).
3. Crossvalidacja pozwala na lepsze określenie dokladności modelu
4. Język R pozwala implementować różnego rodzaju uczenie maszynowe jednakże czasem potrafi być chaotyczny.
5. W przypadku kiepsko nauczonego modelu (w naszym przypadku Wine) pomimo accuracy na poziomie 60% recall potrafi być równy zero co skutku nieliczbami w pomiarach f1 itp.
6. Parametr globabPruning ma duży wpływ na końcowy rozmiar drzewa
7. W zależności od datasetu parametry dawały różne skuteczności modelu. Ale tylko dla nielicznych zbiór Wine dawał jakiekolwiek wyniki.
8. Dla wyższych wartości foldów w crosswalidacji osiągaliśmy lepsze wyniki na metrykach.
9. Wizualizacja dużych drzew może być bardzo nieczytelna z poziomu R.