Projet STA203

Alexandre Bodinier et Corentin SOUBEIRAN

27/04/2020

Classification supervisée de genres musicaux :

Partie 1: Analyse des données et régression logistique :

1. Analyse descriptive :

L'objet de ce projet est la classification supervisée de musiques en fonction de leur genre : Jazz ou Classique. Pour ceci, nous disposons de 6447 musiques différentes indépendantes, avec 191 variables qui illustrent 16 paramètres physiques différents. Ces paramètres sont principalement issus de l'analyse spectrale, *cepstral*[1] et temporelle. La répartition des données entre les deux genre est équitable avec 53% de musiques classiques.

```
## Classical      Jazz
##      3444      3003
```

Analyse uni et bi-varié : L'analyse univariée du jeu de données à l'aide de la fonction `summary` révèle des ordres de grandeurs très différents entre les variables (du fait que les paramètres physiques mesurés n'aient pas la même dimension). Cet ordre de grandeur va de 10^4 par exemple pour le *centroïde spectral*[2], 10^0 par exemple pour *l'enveloppe spectrale*[3] et 10^{-4} pour la *planaité spectral*[4].

L'analyse des corrélations à l'aide de la fonction `corrplot` montre des corrélations fortes autour de la diagonale, ce qui implique que les variables sont, de proche en proche, corrélées dans le jeu de données. Ceci est dû, comme nous l'expliquerons dans la suite, à la continuité du spectre. On trouve également un bon nombre de variables anticorrélées. Le fait que des variables soient trés correlées ou le contraire, peut amener à se demander si une reduction de dimension ne serait pas utile afin d'éviter un phénomène de *sur-apprentissage* ou de *non convergence* se traduisant par une variance élevée du modèle dû à la *sur-paramétrisation*.

Transformations log : Les paramètres *PAR_SC_V* et *PAR_ASC_V* correspondent respectivement à la variance du centroïde spectral et du centroïde du spectre audio. En regardant les données:

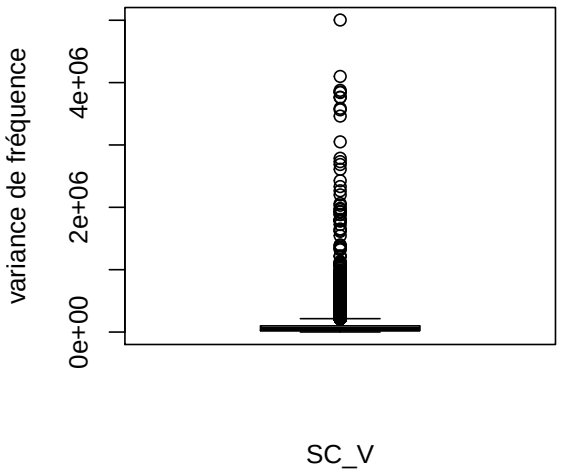
```
## [1] variance SC:
```

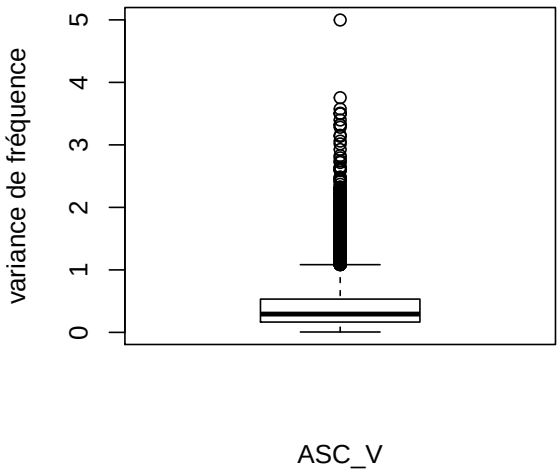
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     605   22714   46309  103732   99464 5003700
```

```
## [1] variance ASC:
```

[1] transformation du signal du domaine temporel à un domaine analogue
[2] spectral centroïd : centre de gravité spectral du signal, il représente le poids relatif des fréquences aiguës et graves
[3] ASE (audio spectrum enveloppe): courbe fréquence/amplitude du signal
[4] SFM: spectral flatness measure, qui représente le rapport signal sur bruit en dB

```
##      Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.005926 0.165035 0.292860 0.426270 0.532465 4.998000
```

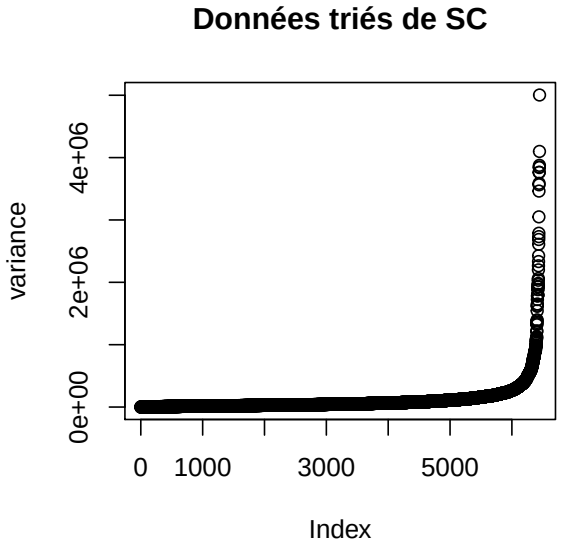


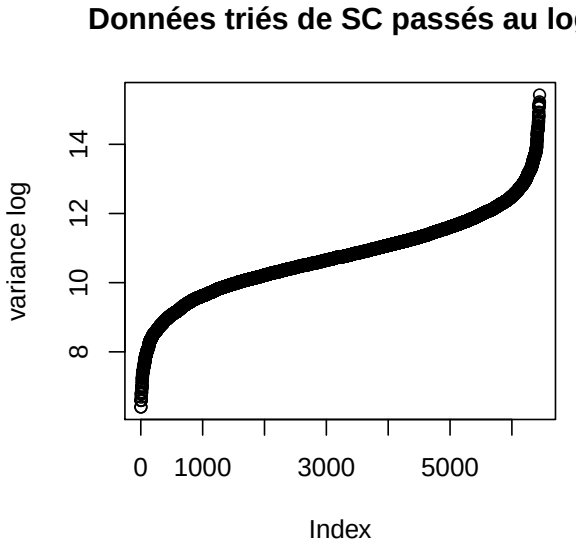


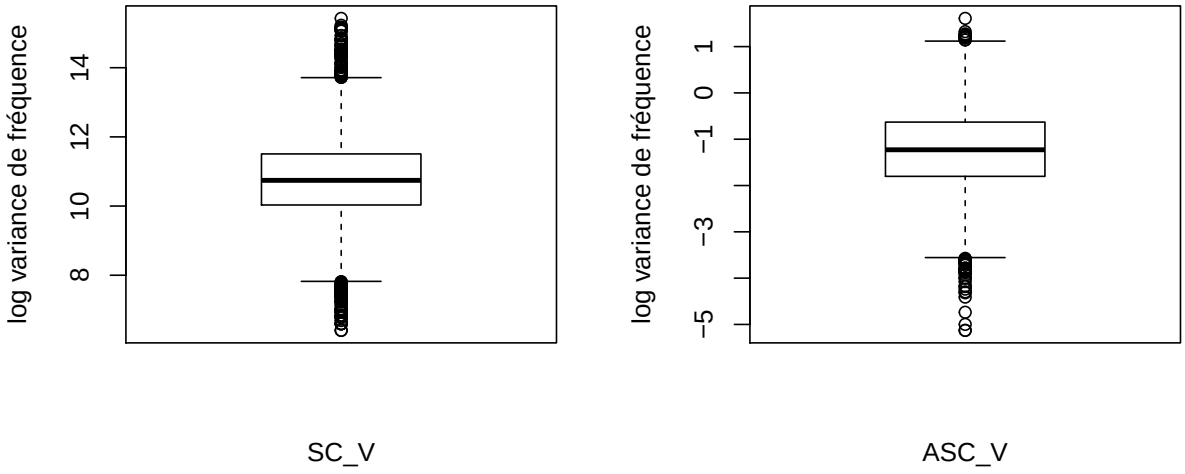
On remarque une forte étendue, confirmée par la médiane plus proche du minimum et par l'écrasement du `boxplot`.

En effet, en observant SC_V par exemple, on remarque une abondance de données pour les faibles valeurs de variance et quelques données de très forte variance. On remarque que la transformation log permet d'uniformiser la répartition des données. Il en est de même pour ASC_V.

Ainsi cette transformation rend utilisables les variables, et leurs répartitions plus exploitables.



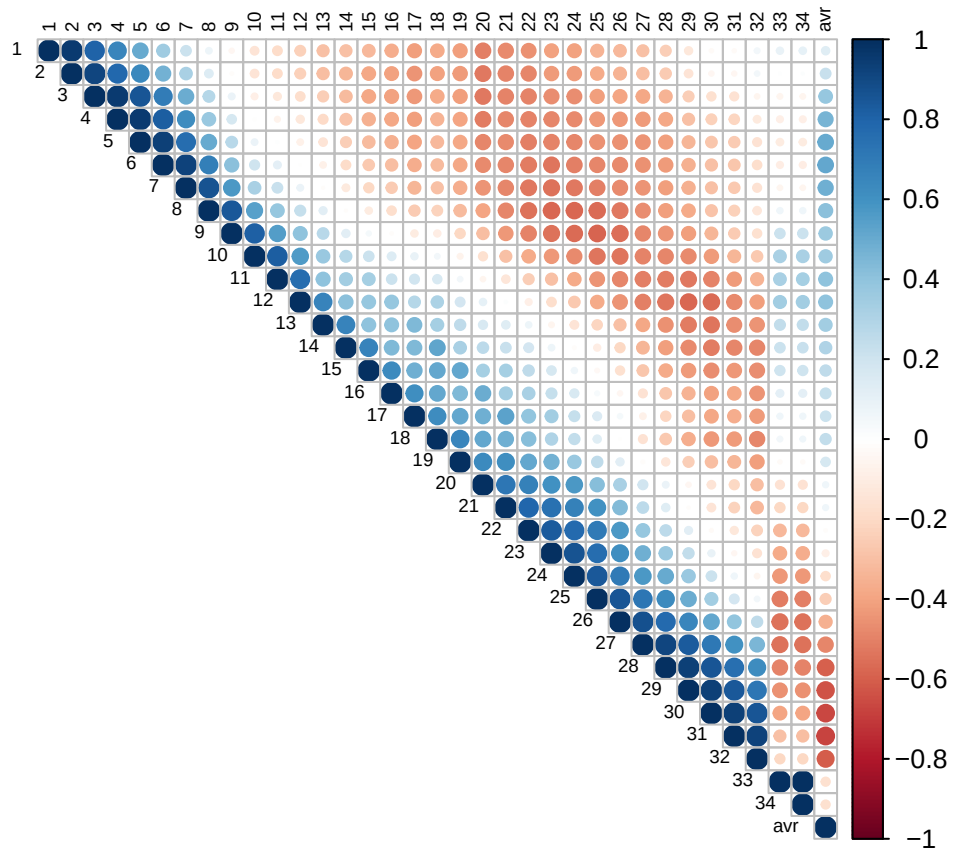


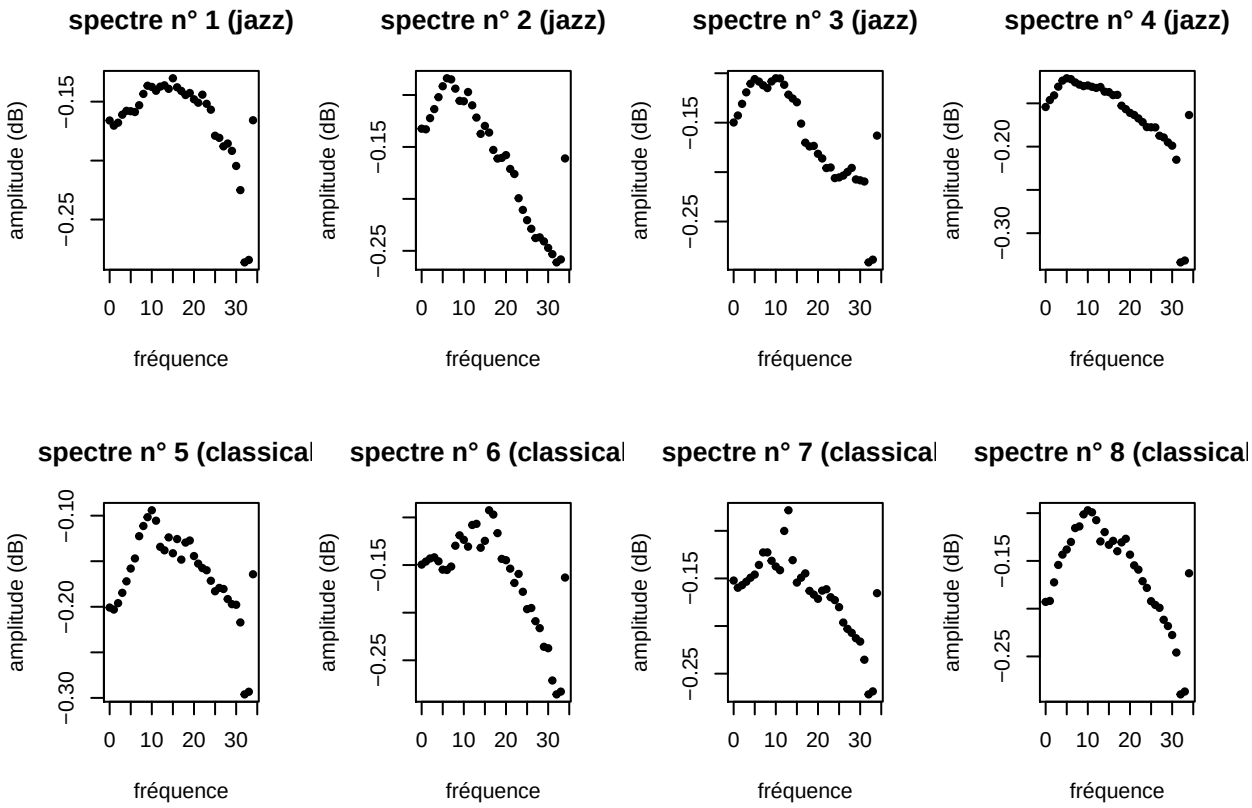


Variables 148 à 167 : On remarque que les variables 128-147 sont fortement corrélées ($> 99.9\%$) aux variables 148-167 une à une. Ce qui est confirmé dans l'annexe: les paramètres sont les mêmes ! Ces variables sont donc inutiles à priori puisqu'elles ne sont qu'une répétition.

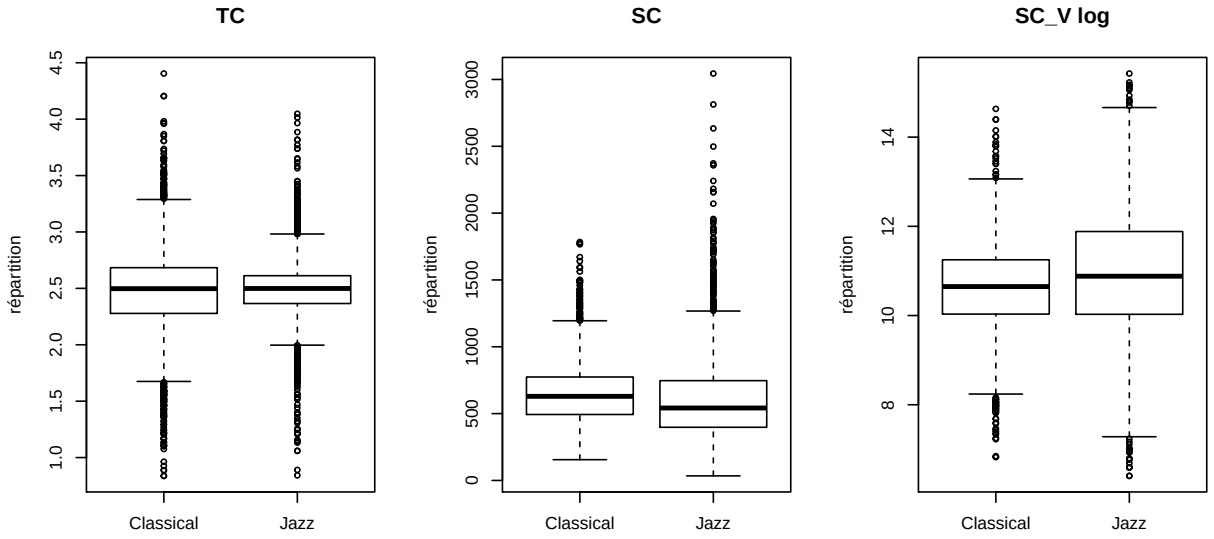
Analyse appronfondie de certaines variables Lors de l'analyse des correlations des variables, on remarque que pour un même paramètre, les correlations sont fortes, par exemple pour les variables du paramètre ASE. La correlation entre les variables PAR_ASE33 et PAR_ASE34 est de plus de $99,9\%$: il n'y a donc pas d'intérêt de garder les deux pour l'étude.

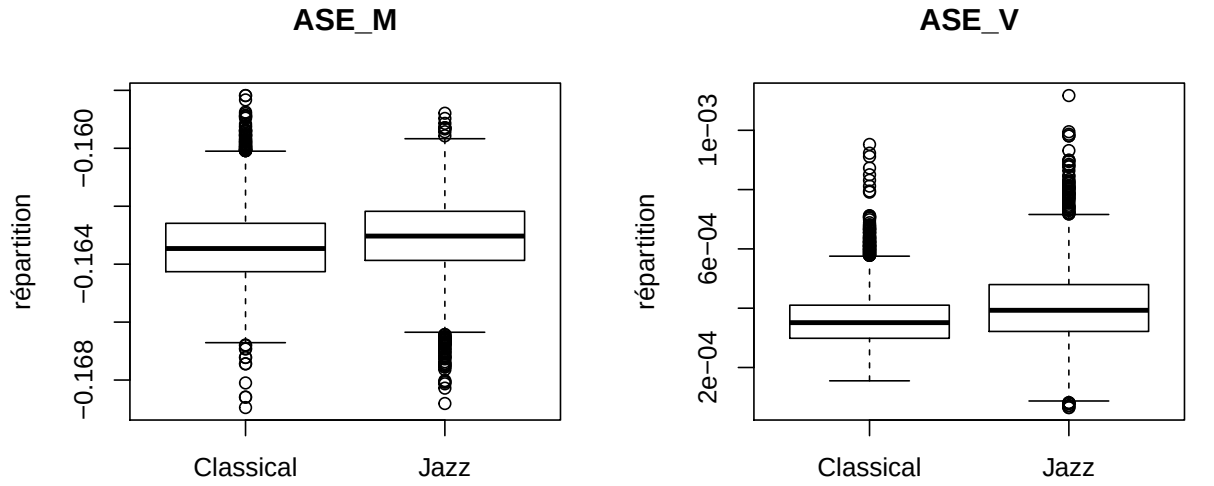
Les "ASE" représentent la forme de l'enveloppe spectrale. Il est donc normal que les variables soient similaires de proche en proche puisqu'il s'agit de bandes voisines. Ici la corrélation rend compte d'un phénomène physique : la continuité du spectre. Intuitivement on pourrait penser qu'il y ait deux familles de spectres, une certaine forme pour le Jazz et une autre pour le Classique, et alors on pourrait séparer les genres par leur forme de spectre ? Malheureusement, on voit que ce n'est pas un très bon séparateur puisque il existe des morceaux qui ont un spectre similaire et un genre pourtant différent (*cf. spectres 2 et 5* sur la figure suivante). C'est notamment pourquoi nous allons utiliser des agrégats (moyennes, variances, moyenne des variances) : (*cf suite*).

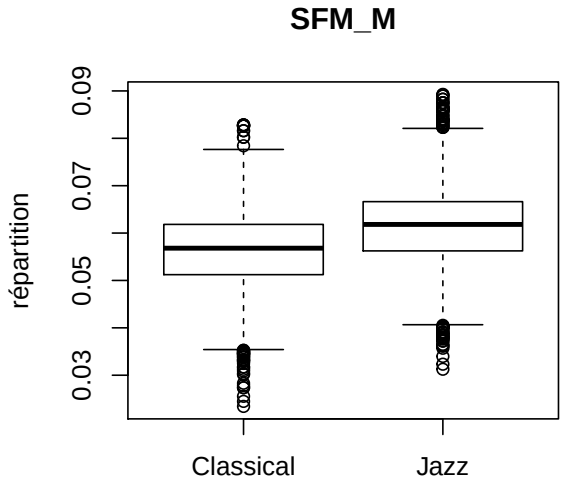

Correlation entre les variables d'ASE
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
avr
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1

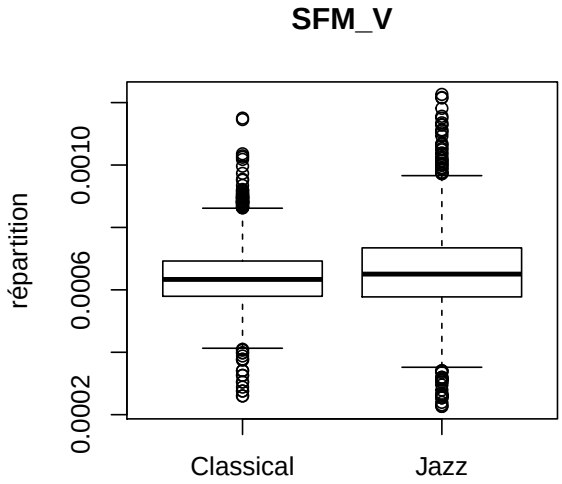


Représentons maintenant les diagrammes en moustache des différentes variables proposées: *TC*,*SC*,*SC_V* (et nous utiliseront plus précisément le *SC_V log* pour les raisons déjà citées),*ASE_M*, *ASE_V*,*SFM_M* et *SFM_V*. Ces données correspondent aux agrégats de variables pour les différents paramètres. Certains agrégats sont un bon levier de séparation puisqu'on observe des répartitions clairement distinctes (en termes de moyenne, quartiles, variance, étendue), notamment pour la *SFM_M*. Pour d'autres, c'est moins le cas (exmple de PAR_TC).


TC
répartition
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
Classical
Jazz
SC
répartition
0
500
1000
1500
2000
2500
3000
Classical
Jazz
SC_V log
répartition
8
10
12
14
Classical
Jazz


ASE_M
répartition
−0.168
−0.164
−0.160
Classical
Jazz
ASE_V
répartition
2e−04
6e−04
1e−03
Classical
Jazz





Bilan : Ainsi il semble que:

- Les variables 148 à 167 sont à retirer du modèle.
- On peut passer au log les variables SC_V et ACV_V pour plus de significativité.
- Les ASE individuelles (4-37) ne sont pas forcément différentes pour les deux genres en question, d'où la nécessité d'utiliser des agrégats (ASE_M, ASE_V, ...). Les corrélations rendent compte d'une phénomène physique.
- Il en est de même pour ASE_V 39-72, SFM 78-101, SFM 103-126.
- Nous n'avons pas de données manquantes.
- Nous n'avons vraisemblablement pas de valeurs aberrantes.

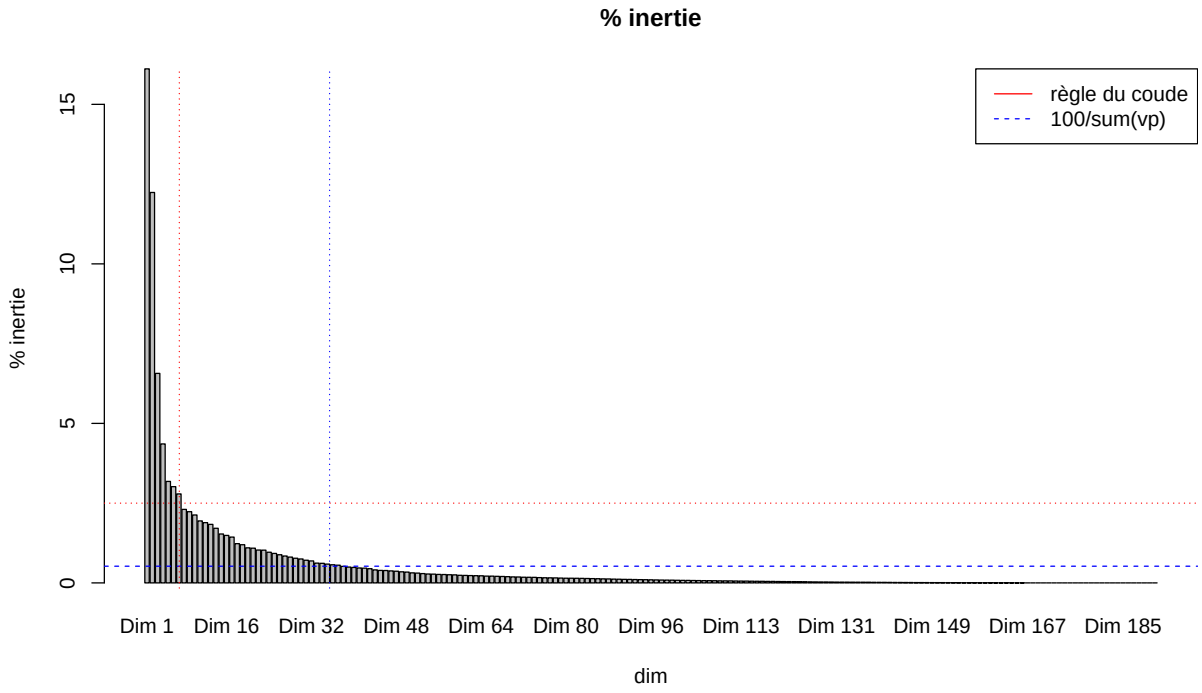
Modèle Logistique : Hypothèses du modèle logistique:

- Nous sommes bien dans un cadre de classification binaire : $\Omega = \{Classique, Jazz\}$.
- Nos variables sont indépendantes : on peut légitimement supposer l'indépendance entre chaque morceau.
- Pour appliquer ce modèle il faut $N_{echantillon} >> N_{variables}$. Ici nous avons un facteur 300, ce qui est suffisant.
- Pour ce qui est de la robustesse il faudra prêter attention aux variables sensibles.

Approfondissement : Avant de lancer un quelconque calcul, nous devons établir des modèles :

- Un premier modèle ne conservant que certains agrégats : ce sera le modèle *Mod0*.

- Un modèle avec beaucoup plus de variables, ces variables seront sélectionnées par intuition et déduction en observant notre jeu de données, en faisant des recherches sur la signification des différents paramètres et en se documentant sur le sujet. (On rassemble le plus de connaissances "terrain"). Ce sera notre modèle *ModT*.
- Afin de raffiner ce modèle T, nous allons nous séparer de variables en conservant que celles qui sont significatives à 5% c'est le modèle *Mod1*.
- Un modèle *Mod2* sera extrapolé de ModT en conservant les variables significatives à 20%.
- Un dernier modèle avec sélection par méthode AIC stepwise : ce sera *MoDAIC*
- Pour chacun des modèles, on utilisera un seuil s de décision maximisant l'*accuracy*, sans prendre en compte la $spécificité$ ou bien la $significativité$.



2. Définition de l'échantillon d'apprentissage :

Pour la suite, nous avons besoin de transformer le genre en variable binaire : False = Jazz, True = classique (n'y voyez aucun jugement de valeur)

```
set.seed(103)
train=sample(c(TRUE, FALSE), n, rep=TRUE, prob=c(2/3, 1/3))
Y = music$GENRE == 'Classical'
```

3. Estimation des modèles :

Modèle 0 : Dans ce modèle nous considérerons les variables agrégés suivantes (TC,SC,SC V,ASE M,ASE MV,SFM M,SFM MV).

```
##
```

```
## Call:
## glm(formula = Y_train ~ ., family = binomial, data = X_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5161  -0.9577   0.5071   0.9246   3.9107
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.664e+01  4.663e+00 -10.002  < 2e-16 ***
## PAR_TC      -1.090e-01  1.045e-01  -1.043    0.297
## PAR_SC       2.803e-04  2.220e-04   1.263    0.207
## PAR_SC_V    -3.032e-06  5.093e-07  -5.953 2.64e-09 ***
## PAR_ASE_M   -3.412e+02  2.889e+01 -11.809  < 2e-16 ***
## PAR_ASE_MV  -7.145e+03  4.975e+02 -14.362  < 2e-16 ***
## PAR_SFM_M   -1.037e+02  6.106e+00 -16.992  < 2e-16 ***
## PAR_SFM_MV   4.189e+02  4.002e+02   1.047    0.295
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5914.0  on 4277  degrees of freedom
## Residual deviance: 4864.4  on 4270  degrees of freedom
## AIC: 4880.4
##
## Number of Fisher Scoring iterations: 6

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

Modèle T : C'est ici que nous avons dû nous documenter pour acquérir des "connaissances terrain". Ainsi, nous avons développé une certaine intuition quant aux variables à conserver (notaments les agrégats, mais aussi des ASE pour des bandes de basse fréquence, des mesures RMS, des mesures SFM). Nous avons fait de la *data visualisation* pour consolider nos intuitions et pour observer quelles variables séparaient le mieux les deux genres.

Nous nous sommes ensuite interrogés sur les résultats qu'une PCA[5] pourrait nous fournir. Cette méthode à pour but de déterminer les axes principaux d'un jeu de donnée (combinaison linéaire de variables expliquant le mieux le jeu de données). En apliquant la règle de coude, on conserverait 7 combinaisons de variables qui d'ailleurs ont l'air de correspondre à celles du *modèle 0* (en terme de qualité de représentation et de contribution). En appliquant le critère $\frac{100}{\sum \lambda_i}$ nous sommes tentés de conserver les 42 premiers axes principaux.
Afin de retrouver quelles variables sélectionner, nous allons regarder lesquelles sont les mieux représentées et les plus contributives aux premiers axes principaux. Il s'agit de :

- PAR_SFM_M, PAR_SFM_MV

[5]Principal composent analysis

- PAR_SC, PAR_SC_V
- PAR_ASE_M, PAR_ASE_MV
- PAR_ASC, PAR_ASC_V
- PAR_PEAK_RMS_TOT
- PAR_ASS_V
- PAR_ASE1 à 8, PAR_ASE23-24 et PAR_ASE30
- PAR_THR_3RMS_TOT, PAR_THR_2RMS_TOT, PAR_THR_1RMS_TOT
- PAR_SFMV16, PAR_SFMV15

On retrouve un premier jeu de variables, correspondant aux variables agrégés. Cela conforte l'observation que nous avons fait sur ces variables. Un second jeu correspond à des ASE ou $SFMV$ pour certaines portion du spectre (basses fréquences, et hautes fréquences pour ASE, ce qui confirme l'intuition : les instruments des deux genres étant différents (plus de batteries, notament de cymbales pour le jazz, ce qui contribue aux hautes fréquences. Les différences de timbre des intruments s'expriment dans les basses et les aigus). C'est cet ensemble de 26 varaibles qui formera notre *modèle T*

```
##
## Call:
## glm(formula = Y_train ~ ., family = binomial, data = X_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9561  -0.5956   0.3160   0.7421   2.7390
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -2.772e+01  1.500e+01  -1.848 0.064605 .
## PAR_SFM_M        -7.051e+01  8.006e+00  -8.807  < 2e-16 ***
## PAR_SC           -1.993e-03  6.756e-04  -2.950 0.003173 **
## PAR_ASE1          1.086e+02  9.049e+00  12.000  < 2e-16 ***
## PAR_ASE2         -1.393e+02  1.292e+01 -10.785  < 2e-16 ***
## PAR_ASE3          4.522e+01  1.768e+01   2.558 0.010529 *
## PAR_ASE4         -4.666e+01  2.544e+01  -1.834 0.066638 .
## PAR_ASE5          2.164e+01  2.481e+01   0.872 0.383100
## PAR_ASE6          3.962e+00  1.989e+01   0.199 0.842094
## PAR_ASE7          1.958e+01  1.218e+01   1.608 0.107833
## PAR_ASE8          1.394e+00  6.094e+00   0.229 0.819126
## PAR_ASE24         1.421e+01  5.482e+00   2.592 0.009533 **
## PAR_ASE30        -2.770e+01  6.319e+00  -4.385 1.16e-05 ***
## PAR_ASE23         1.367e+01  5.308e+00   2.575 0.010012 *
## PAR_ASC           1.194e+00  2.860e-01   4.174 3.00e-05 ***
## PAR_ASC_V         1.644e-01  1.941e-01   0.847 0.397073
## PAR_SFM_MV       -2.224e+03  5.777e+02  -3.849 0.000119 ***
## PAR_ASE_M        -2.454e+02  8.581e+01  -2.860 0.004238 **
## PAR_ASE_MV       -2.867e+03  6.739e+02  -4.254 2.10e-05 ***
## PAR_PEAK_RMS_TOT  8.406e-02  4.374e-02   1.922 0.054625 .
## PAR_ASS_V        -5.828e+00  1.072e+00  -5.438 5.39e-08 ***
## PAR_THR_3RMS_TOT -2.066e+01  3.060e+01  -0.675 0.499663
## PAR_THR_2RMS_TOT  6.561e+01  1.255e+01   5.228 1.72e-07 ***
```

```
## PAR_THR_1RMS_TOT -2.972e+01  4.993e+00  -5.951 2.66e-09 ***
## PAR_SC_V         -7.099e-07  4.322e-07  -1.642 0.100500
## PAR_SFMV16        5.655e+02  6.027e+02   0.938 0.348100
## PAR_SFMV15       -1.321e+03  4.340e+02  -3.044 0.002338 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5914.0  on 4277  degrees of freedom
## Residual deviance: 3829.4  on 4251  degrees of freedom
## AIC: 3883.4
##
## Number of Fisher Scoring iterations: 6
```

Modèle 1 :

Pour *Mod1* nous ne conservons que les variables significatives à 5%. Cependant, ce n'est pas une très bonne méthode, puis que le fait d'enlever des variables modifie l'ensemble des résultats ainsi, il est possible qu'il reste tout de même des variables non significatives après la réduction. C'est justement ceci qui justifieras l'utilisation de la méthode AIC.

```
##
## Call:
## glm(formula = Y_train ~ ., family = binomial, data = X_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1258  -0.5940   0.3230   0.7434   2.6543
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -4.217e+01  1.343e+01  -3.140  0.00169 **
## PAR_SFM_M        -6.736e+01  7.847e+00  -8.584  < 2e-16 ***
## PAR_SC           -1.587e-03  6.197e-04  -2.562  0.01042 *
## PAR_ASE24         1.457e+01  5.424e+00   2.687  0.00722 **
## PAR_ASE1          1.080e+02  8.907e+00  12.123  < 2e-16 ***
## PAR_ASE2         -1.370e+02  1.269e+01 -10.795  < 2e-16 ***
## PAR_ASE3          2.977e+01  1.463e+01   2.035  0.04185 *
## PAR_ASE4         -1.237e+01  1.068e+01  -1.158  0.24691
## PAR_ASE7          2.824e+01  3.804e+00   7.422 1.15e-13 ***
## PAR_ASE30        -3.275e+01  5.885e+00  -5.564 2.64e-08 ***
## PAR_ASE23         1.346e+01  5.257e+00   2.561  0.01045 *
## PAR_ASC           1.057e+00  2.563e-01   4.124 3.73e-05 ***
## PAR_SFM_MV       -2.088e+03  5.594e+02  -3.733  0.00019 ***
## PAR_ASE_M        -3.297e+02  7.522e+01  -4.383 1.17e-05 ***
## PAR_ASE_MV       -2.674e+03  6.012e+02  -4.448 8.66e-06 ***
## PAR_ASS_V        -6.109e+00  1.042e+00  -5.861 4.61e-09 ***
## PAR_THR_2RMS_TOT  5.670e+01  9.619e+00   5.895 3.75e-09 ***
## PAR_THR_1RMS_TOT -3.050e+01  3.198e+00  -9.537  < 2e-16 ***
## PAR_SC_V         -6.119e-07  4.143e-07  -1.477  0.13970
## PAR_SFMV15       -1.029e+03  3.178e+02  -3.239  0.00120 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5914.0  on 4277  degrees of freedom
## Residual deviance: 3838.4  on 4258  degrees of freedom
## AIC: 3878.4
##
## Number of Fisher Scoring iterations: 5
```

Modèle 2 :

Même remarque que pour *mod1*, il n'est pas judicieux de procéder de la sorte (retirer l'ensemble des variables non significatives). On remarque dans les résultats que ces méthodes ne sont pas performantes (réduction de la précision). Et il reste encores des variables non significatives (SFMV16 par exemple).

```
##
## Call:
## glm(formula = Y_train ~ ., family = binomial, data = X_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1063  -0.6700   0.3981   0.8051   2.6132
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -2.422e+01  1.414e+01  -1.713 0.086793 .
## PAR_SFM_M        -5.561e+01  7.487e+00  -7.428 1.10e-13 ***
## PAR_SC           -3.687e-03  6.415e-04  -5.748 9.02e-09 ***
## PAR_ASE24         1.694e+01  5.235e+00   3.237 0.001209 **
## PAR_ASE5         -8.510e+00  3.160e+00  -2.694 0.007069 **
## PAR_ASE30        -2.356e+01  5.850e+00  -4.027 5.64e-05 ***
## PAR_ASE23         1.141e+01  5.095e+00   2.240 0.025080 *
## PAR_ASC           1.477e+00  2.581e-01   5.723 1.05e-08 ***
## PAR_ASC_V         1.300e-01  1.832e-01   0.709 0.478017
## PAR_SFM_MV       -1.397e+03  5.563e+02  -2.511 0.012041 *
## PAR_ASE_M        -2.183e+02  7.942e+01  -2.748 0.005992 **
## PAR_ASE_MV       -3.437e+03  6.499e+02  -5.288 1.24e-07 ***
## PAR_PEAK_RMS_TOT  5.269e-02  3.990e-02   1.320 0.186687
## PAR_ASS_V        -5.707e+00  9.454e-01  -6.036 1.58e-09 ***
## PAR_THR_3RMS_TOT -1.001e+01  2.951e+01  -0.339 0.734372
## PAR_THR_2RMS_TOT  6.199e+01  1.200e+01   5.167 2.37e-07 ***
## PAR_THR_1RMS_TOT -3.030e+01  4.874e+00  -6.217 5.05e-10 ***
## PAR_SC_V         -1.175e-07  3.806e-07  -0.309 0.757543
## PAR_SFMV16       -3.147e+01  5.907e+02  -0.053 0.957512
## PAR_SFMV15       -1.643e+03  4.231e+02  -3.883 0.000103 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5914.0  on 4277  degrees of freedom
## Residual deviance: 4129.9  on 4258  degrees of freedom
```

```
## AIC: 4169.9
##
## Number of Fisher Scoring iterations: 5
```

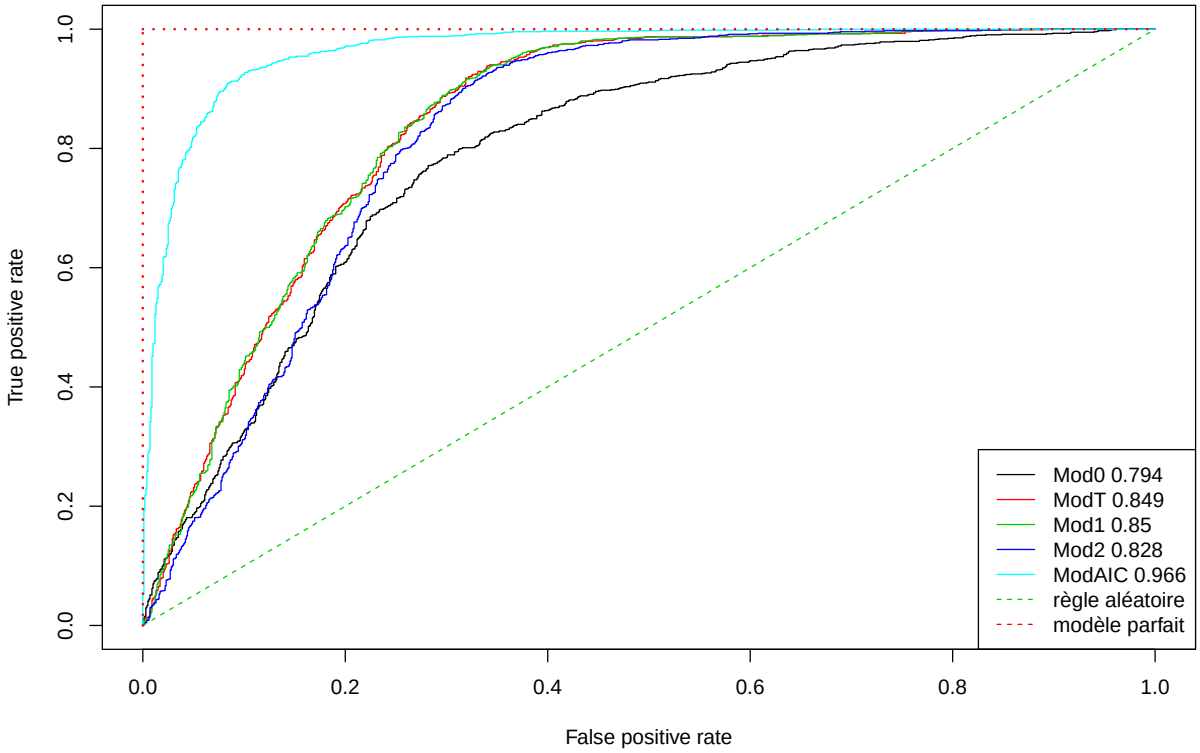
Modèle AIC :

L'objectif est de raffiner le modèle pas à pas. On souhaite diminuer le nombre de variables à prendre en compte afin de réduire le biais de notre estimateur. Pour se faire, on va pénaliser la déviance du modèle avec $2K$, K étant le nombre de variables du modèle. N'oublions pas de préciser qu'il est important de vérifier la cohérence des résultats donnés par la méthode (les variables sélectionnées), il est impossible de se fier les yeux fermés à un programme informatique.

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Question 4 : Visualisation des résultats

Nous allons maintenant comparer les différents tests en superposant leur courbes *ROC*, calculées sur l'échantillon de test.



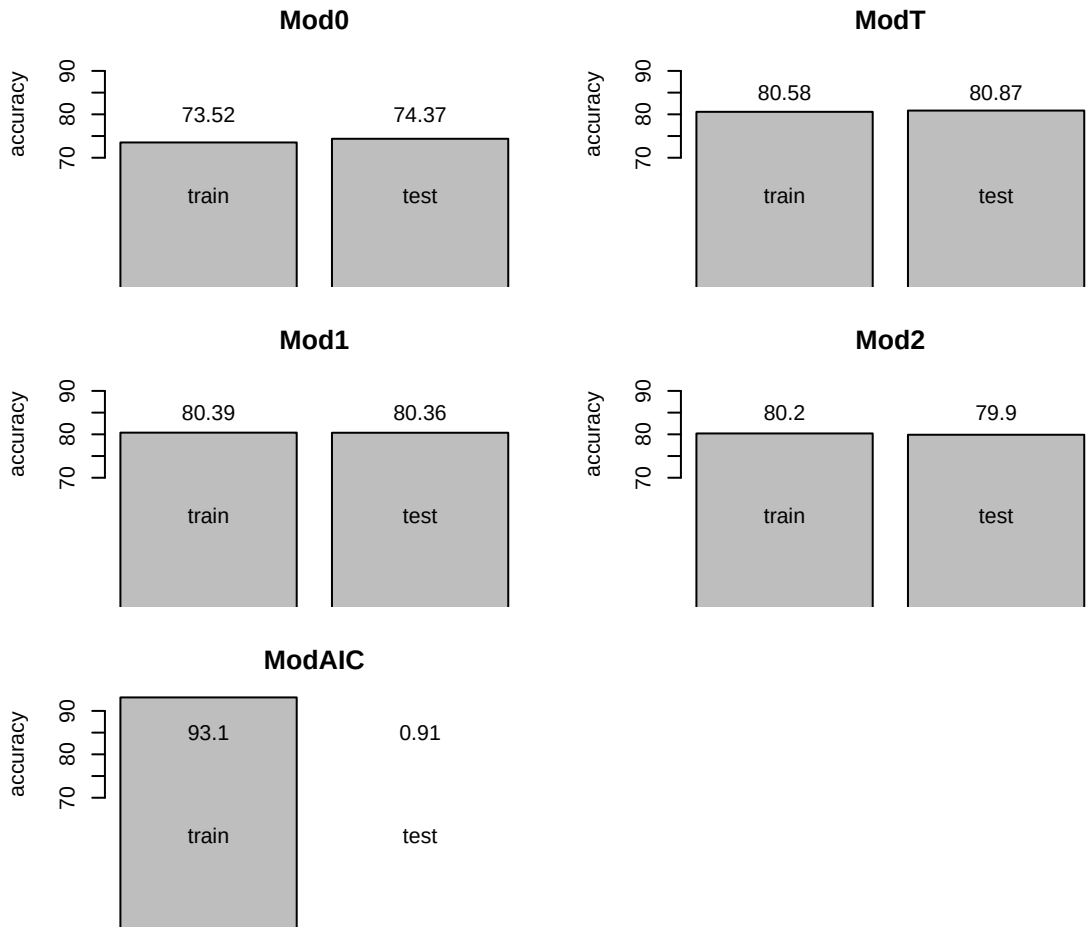
Nous pouvons tout d'abord à partir des résultats des regressions, comparer les modèles en fonction du *Akaike Information Criterion*, nous remarquons alors que ce critère classe de la même manière les modèles que le critère de comparaison global des courbes ROC (aire sous courbe ou AUC).

```
##        Modèle  AIC Accuracy
## test     Mod0 4880     74.4
## test.1   ModT 3883     80.9
## test.2   Mod1 3878     80.4
## test.3   Mod2 4170     79.9
## test.4 ModAIC 1865      0.9
```

Le modèle qui semble le plus pertinent est celui trouvé par le `stepAIC` qui se compose de 141 variables. Bien que ce modèle conserve de nombreuses variables, dont certaines très corrélées, il permet d'atteindre des performances proches de 97% au sens AUC. Notons également que le modèle $ModT$ est meilleur que les modèles *Mod1* et *Mod2*. En pratique cette méthode d'élimination de variable en fonction de la p-value est une mauvaise pratique à remplacer par une procedure de selection de variables pas à pas par *forward*, *backward* ou *stepwise (methode mixte)* sur critère AIC ou BIC.

Question 5 :

On calcule maintenant l'erreur sur les échantillons d'apprentissage et de test :



On remarque ainsi que le taux de bonnes réponses sur l'ensemble de test est proche de celle sur notre ensemble de validation. Cela signifie que l'on a évité le $sur - apprentissage$. Là où l'écart est le plus important est pour le modèle AIC, ce est logique puisque c'est le plus paramétrisé.

A ce stade, on choisit le modèle **ModAIC**. Intéressons nous à son adéquation. Le résultat du `summary(ModAic)` donne une déviance résiduelle de 1573.7 avec 4051 degrés de liberté. On calcul alors le quantile du Khi2 à 4051 ddl pour $\alpha = 5\%$:

```
qchisq(0.95,4051)
```

```
## [1] 4200.183
```

Ce quantile est plus grand que la déviance résiduelle, par conséquent on conserve $H0$ (le modèle). Il est donc adéquat avec un risque de seconde espèce inconnu.

Partie 2: Algorithme des K plus proches voisins :

1. KNN :

La methode des KNN (k-nearest neighbors) est une methode de classification supervisée.

Elle consiste à classer un individu non labélisé (en espace de dimension p = le nombre de paramètres) en fonction de la classe de ses k plus proches voisins labélisés. “Proche” est à prendre au sens de la norme euclidienne.

Autrement dit : à partir d'un jeu de données labellisé (le set d'apprentissage), on attribue un label aux individus d'un jeu inconnu en fonction de leur position dans l'espace. C'est la stratégie de “qui se ressemble s'assemble”. Dans le cas ou k=1 on donne à une nouvel individu le label de l'individu le plus proche de l'ensemble d'entrainement. Rappelons qu'ici, chaque individu est un morceau de musique.

2. Classification avec l'algorithme des KNN :

Jeu de données réduit : En utilisant les variables du modèle T, et avec le paramètres k = 1, on obtient une *accuracy* (taux de bonne réponses) de :

```
## [1] Accuracy=91.055786076533
```

On peut alors dresser la matrice de confusion de cette prédiction :

```
##      V   F
## 1 1113 135
## 2   59 862
```

Et calculer des indicateurs comme la spécificité et sensibilité :

```
## [1] sensibilité = 94.9658703071672
```

```
## [1] spécificité = 86.4593781344032
```

Jeu de données entier : Précédemment, nous avons appliqué la methode des KNN au jeu de donnée réduit sur 26 variables mais on pourrait s'intéresser à son resultat sur un jeu de variables bien plus grand, composé des 191 paramètres. On va tout de même retirer les variables 147 à 168 (qui sont des répétitions) et en appliquant la transformation log aux paramètres SC_V et ASC_V :

```
## [1] Accuracy = 94.8162111215834
```

On remarque alors un taux de bonnes réponses 3% plus élevé avec l'ensemble des variables plutot que sur le modèle réduit. Néamoins le temps de calcul est beaucoup plus long, en fonction du domaine d'application ce paramètre est à prendre en compte.

```
##      V   F
## 1 1123  89
## 2   21 889
```

```
## [1] sensibilité=98.1643356643357
```

```
## [1] spécificité=90.8997955010225
```

Il en va de même pour la sensibilité et la spécificité.

3.Commentaires :

On remarque que l'utilisation de l'ensemble des variables plutôt qu'une partie, améliore de 3% le taux de bonne réponse et atteint un taux de $94,8\%$. Contrairement aux méthodes de classification de la partie 1 ou 3, les KNN ne font pas intervenir de seuil et ne pouvons pas dresser de courbe ROC à proprement parler. En effet la méthode des KNN est une methode déterministe (non-paramètrique).

De plus, nous pouvons nous interroger sur la pertinence de prendre $k = 1$. En effet cela implique que seul le voisin le plus proche influe sur le choix de la classe. Nous pourrions vouloir augmenter ce nombre. Cependant que ce soit à partir de toutes les variables[6] ou de celles du modèle T, l'*accuracy* a une tendance à la baisse avec l'augmentation de k. A titre illustratif nous avons représenté cette accuracy pour les premières valeurs k. On remarque que pour de faibles valeurs de k, le taux est très variable puis se stabilise par la suite. Cela illustre le "curse of dimensionality" : nous avons un nombre élevé des paramètres, et dans cet espace de grande dimension, tous les individus deviennent proches. Il serait alors intelligent d'appliquer des poids aux variables pour ce KNN. Cela constitue une perspective d'amélioration.

Nous retenons de cette observation qu'il est préférable de ne considérer qu'un faible nombre de voisins (moins de 10). En l'état nous ne pouvons pas choisir à proprement dit une valeur de k, car nous n'auront alors plus accès à l'erreur de généralisation. Pour cela il est nécessaire de séparer l'échantillon en 3 parties: apprentissage, validation et test. L'ensemble de validation servant à calculer la valeur de k optimale, et l'ensemble de test à obtenir une erreur de généralisation. Cependant les modèles de la partie 1 utilisent 2/3 des données pour l'apprentissage et 1/3 pour le test.

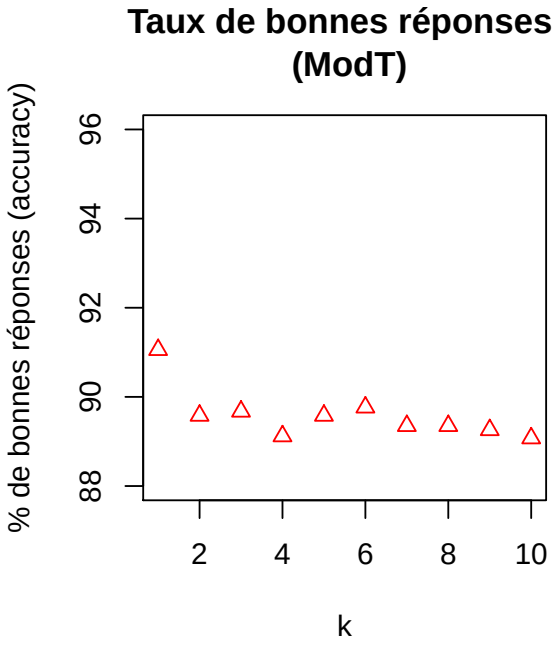
Le KNN est comme nous le constatons nettement plus performant (plus de 30%) avec des variables centrée réduites. En effet notre jeu de variable disposant de variables a ordre de grandeur différents, toutes n'ont pas la même importance sans cette étape préalable et nuit à la prédiction de la classe. Cela était également nécessaire dans les régressions, mais les méthodes appliquent elles même le re-scaling, ce qui n'est pas le cas de KNN

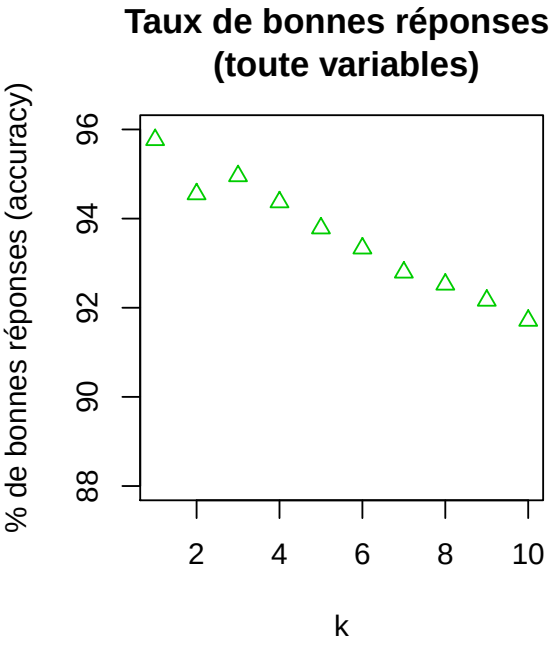
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

[6] hors variables 148 à 167 et en passant au log SC_V et ASC_V





Partie 3: Regression de Ridge

1. Intérêt:

La methode Ridge est une methode de régularisation en classification supervisée. Cette methode a généralement pour but de palier aux contraintes du "cadre des grandes dimensions" (jeu de données où le nombre de variables est plus important que le nombre d'observations). L'objectif est alors d'ajouter une contrainte afin de pouvoir malgré tout conserver nos variables explicatives malgré la non injectivité. L'inconvénient réside dans le fait que l'estimateur obtenu est biaisé mais que sa variance est meilleure que celle obtenue par les moindres carré. Il faut alors trouver un compromis biais/variance.

Notre étude vise la classification du genre des musiques et non la détermination des paramètres les plus significatifs différenciant les deux. Ainsi, nous ne cherchons pas forcément à éliminer des variables de notre modèle, c'est un avantage. Dans notre cas, le nombre de paramètres est de 191 pour 6447 individus, nous ne somme pas dans un cas avec un problème d'injectivité. Néanmoins nous avons certaines variables fortement corrélées, ce qui pose bien entendu problème dans le cadre d'une régression classique. C'est là que la regression de Ridge intervient puisqu'elle permet justement de traiter le cas de variables corrélées.

2. Regression Ridge :

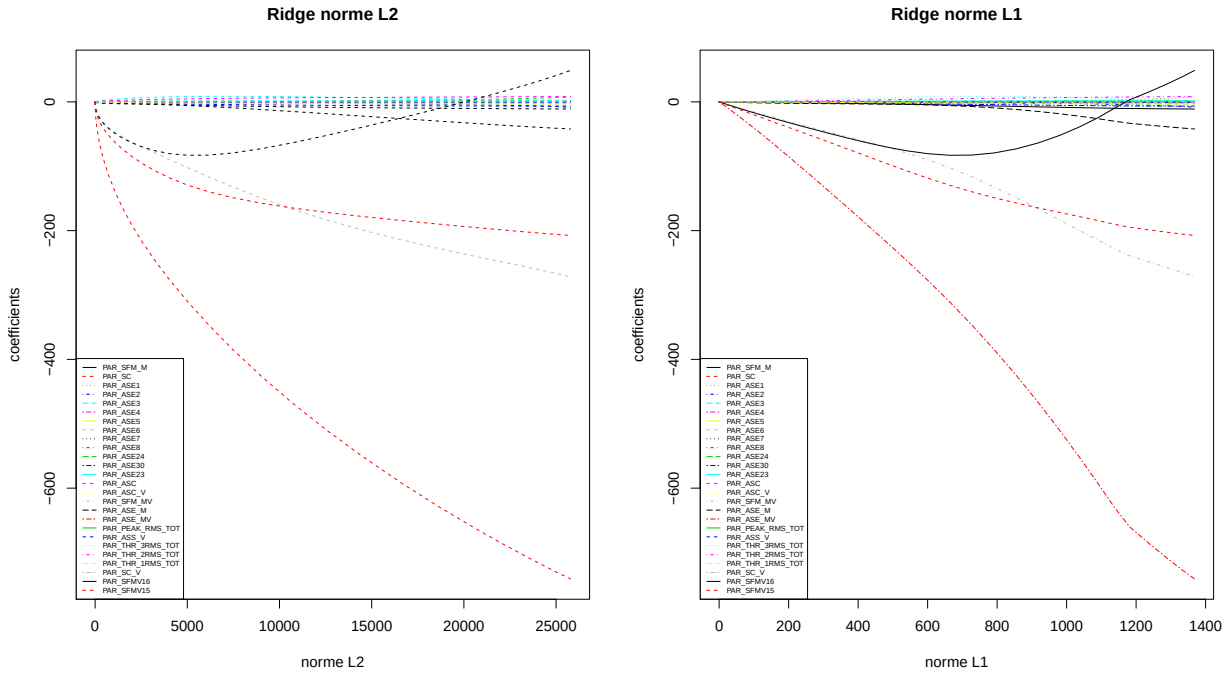
Pour des valeurs de λ très grandes (comme 10^{10}) les coefficients prédits tendent vers 0, à l'inverse, pour des valeurs faibles de λ et se rapprochant de 0 (comme 10^{-2}) l'estimateur de Ridge tend vers l'EMCO.

```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-2
```

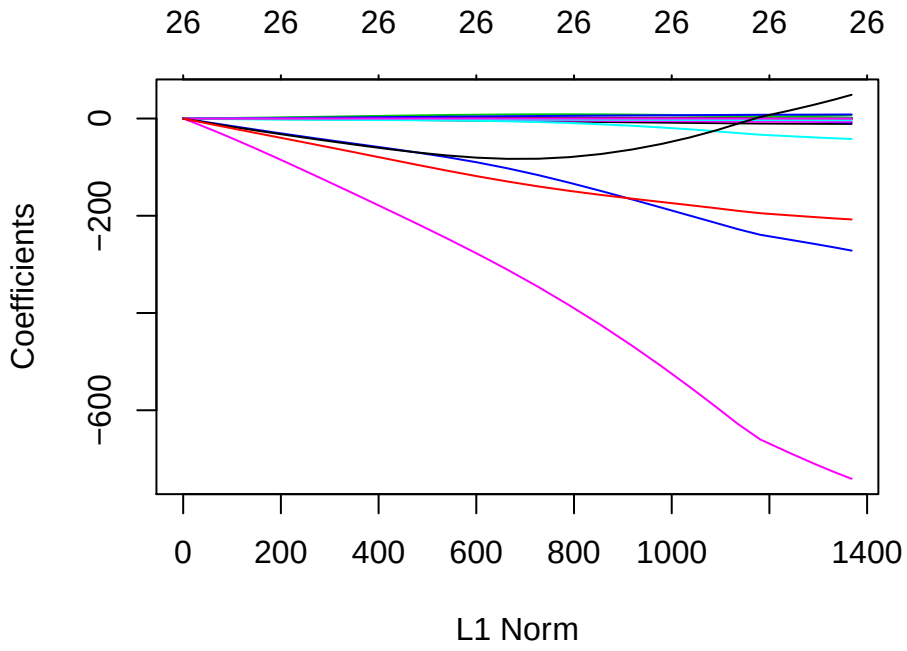
Traçons les graphiques des trajectoires des coordonnées de l'estimateur en fonction de la norme L1 ou L2 de cet estimateur pour les variables du modèle T:

```r
par(mfrow=c(1,2))
###### Norme L2
matplot(apply(coef.ridge^2,2,mean),t(coef.ridge), main="Ridge norme L2",
        col=1:length(names(X_train)), lty=length(names(X_train)), type="l",
        xlab="norme L2", ylab="coefficients")

legend("bottomleft", names(X_train),lty=1:length(names(X_train)),
       col=1:length(names(X_train)),cex=0.5)

###### Norme L1
matplot(apply(abs(coef.ridge),2,sum),t(coef.ridge), main="Ridge norme L1",
        col=1:length(names(X_train)),lty=1:length(names(X_train)),type="l",
        xlab="norme L1", ylab="coefficients")

legend("bottomleft", names(X_train),lty=1:length(names(X_train)),
       col=1:length(names(X_train)),cex=0.5)
```



Le résultat de la fonction *plot* est le suivant, il correspond aux trajectoires des coordonnées de l'estimateur en fonction de la norme L1. Le 26 correspondant au nombre de variables.

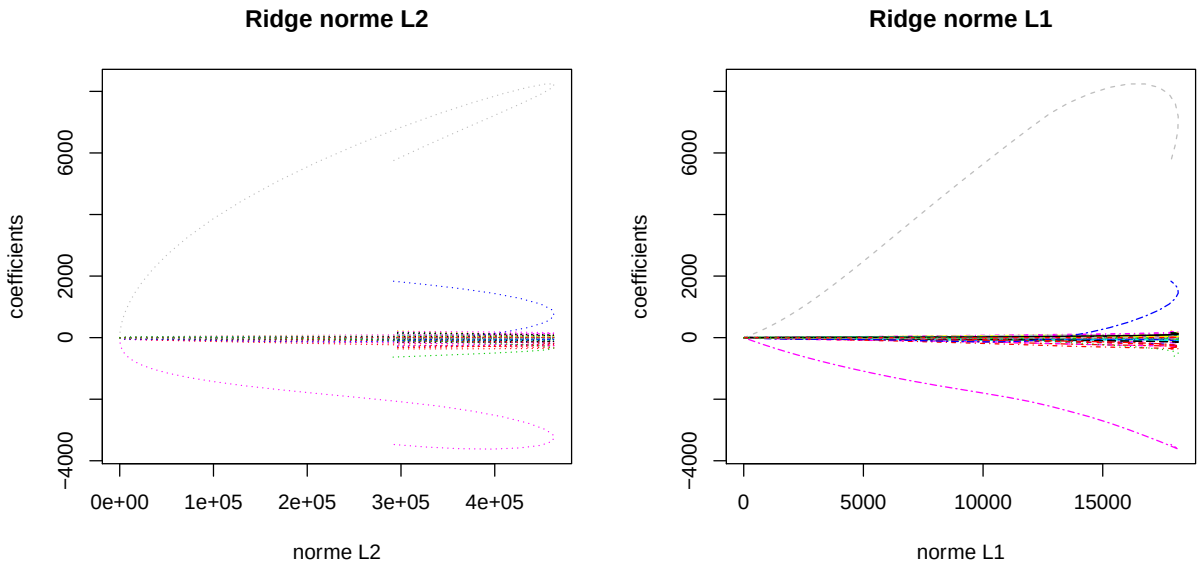
```r
###### Plot classique
par(mfrow=c(1,1))
plot(ridge.fit)
```



En conservant toutes les variables[7]

```
par(mfrow=c(1,2))
###### Norme L2
matplot(apply(coef.ridge^2,2,mean),t(coef.ridge), main="Ridge norme L2",
        col=1:length(names(X_train)), lty=length(names(X_train)), type="l",
        xlab="norme L2", ylab="coefficients")
# legend("bottomleft", names(X_train),lty=1:length(names(X_train)),
#        col=1:length(names(X_train)),cex=0.5)

###### Norme L1
matplot(apply(abs(coef.ridge),2,sum),t(coef.ridge), main="Ridge norme L1",
        col=1:length(names(X_train)),lty=1:length(names(X_train)),type="l",
        xlab="norme L1", ylab="coefficients")
```

[7] hors variables 148 à 167 et en passant au log SC_V et ASC_V




```
# legend("bottomleft", names(X_train),lty=1:length(names(X_train)),
#       col=1:length(names(X_train)),cex=0.5)
```

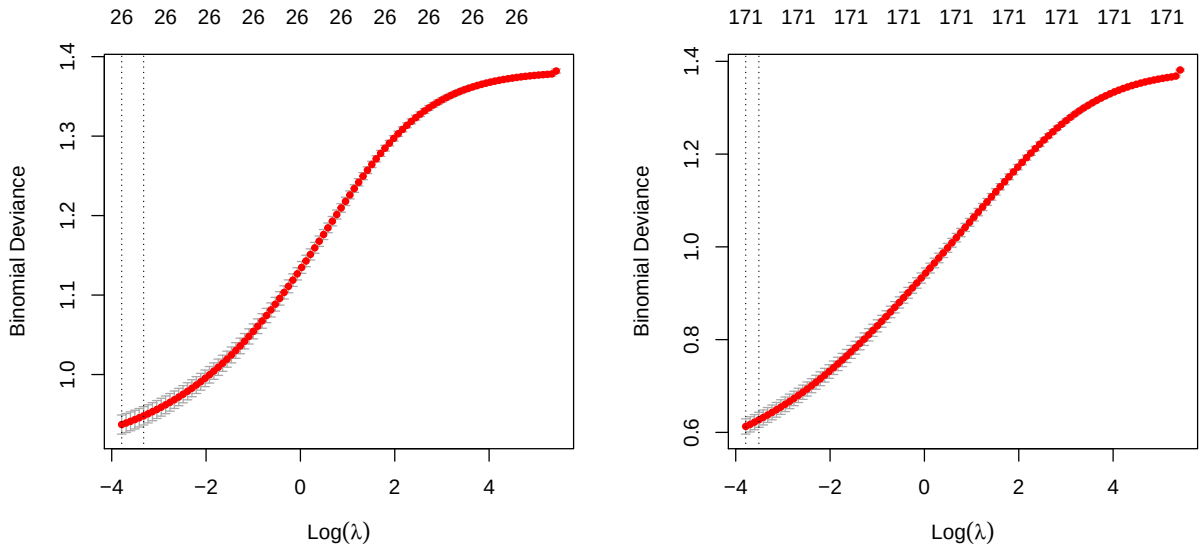
3. Validation croisée :

Nous appliquons une validation croisée à 10 segments pour deux modèles : d'une part au modèle T et d'autre part sur le modèle conservant toutes les variables[8].

La valitation croisée sur dix segments consiste à générer le modèle à partir de 9 parties de l'ensemble de données et de calculer sa performance sur le 10 ième. On fait cela 10 fois en alternant les plis. On fait alors la moyenne des erreurs de prédiction sur les 10 plis isolés.

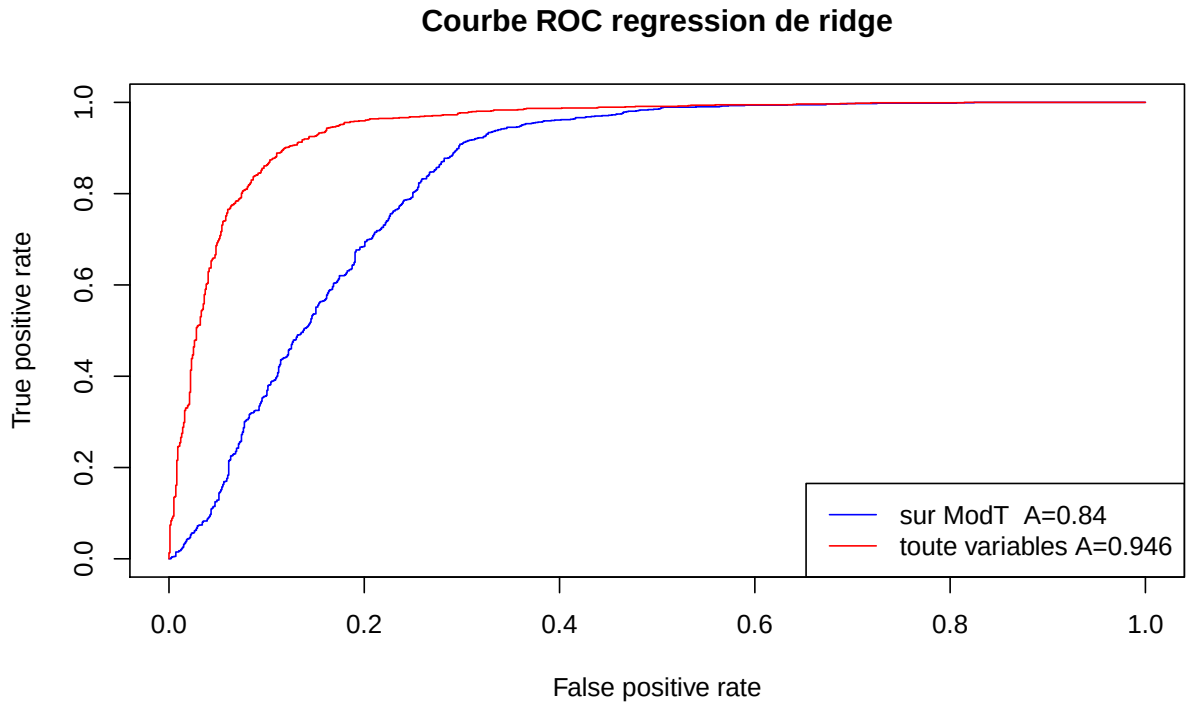
On obtient ainsi l'EQMP (erreur quadratique moyenne de prévision). Ainsi cette estimation de la performance calculée par validation croisée permet de selectionner un modèle (et donc un lambda dans le cas du modèle de Ridge) pour classifier nos données.

[8] hors variables 148 à 167 et en passant au log SC_V et ASC_V




```
## [1] EQM modT=2.21110587733226
```

```
## [1] EQM total=5.40653072890064
```

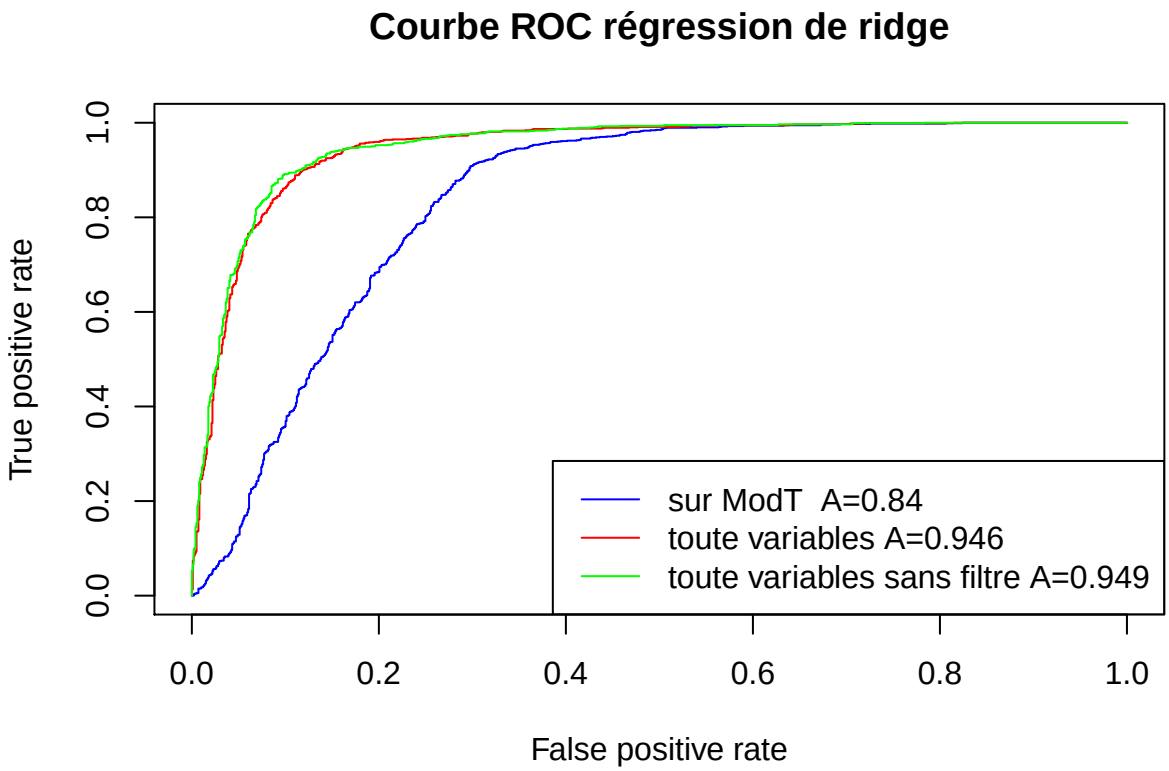


On remarque alors qu'avec une aire sous courbe de $94,6\%$ le modèle avec toute les variables[9] est intéressant pour la classification des genres musicaux.

[9] hors variables 148 à 167 et en passant au log SC_V et ASC_V

4. Utilisation de toutes les variables :

```
## [1] EQM total=5.69579698645104
```



On observe alors que la copie des données à une influence sur les performances qui sont légèrement meilleures. La copie des données donne alors plus de "poids" aux paramètres concernés, et si ces paramètres sont représentatifs pour notre ségrégation, ceci améliore notre classification.

```
##                              Modèle   AIC Accuracy
## 1                       ridge ModT  0.84    0.803
## 2      ridge toute variable filtrés 0.946    0.891
## 3 ridge toute variable sans filtres 0.949    0.895
```

Résumé de la démarche :

Rappel de l'objectif : Différencier les musiques Classiques des musiques de Jazz, sur la base de multiples mesures (191 variables à notre disposition pour 6447 individus). Nous sommes en stratégie supervisée car nous avons un set labélisé à disposition.

Nous commençons l'étude par une analyse descriptive du jeu de données. Un rapide coup d'oeil sur les variables nous permet de constater qu'elles mesurent 16 paramètres physiques caractéristiques d'enregistrements musicaux. Certaines variables sont des mesures par bande de fréquence (ASE par exemple) et d'autres sont des agrégats. L'analyse descriptive révèle que les ordres de grandeurs très variables, et, plus grave : de nombreuses variables sont très corrélées.

Nous enchaînons avec une étape de pré-processing : il s'agit d'appliquer une transformation log aux variables présentant une étendue très élevée et des valeurs extrêmes. On élimine les variables en double (147 à 168). Et une PCA va nous aider à identifier les variables les plus importantes.

Le cadre du problème (classification binaire) nous oriente directement vers une régression logistique. Nous élaborons différents modèles, que nous évaluons. Il s'avère que le plus performant est le modèle de sélection de variables par méthode *stepAIC*. Face au problème des variables très corrélées, nous choisissons d'appliquer ensuite une régression ridge, qui permet de palier à ce défaut. Les résultats sont très satisfaisants avec une précision de 91.33. Cependant, par nature, la régression logistique ne permet que de classifier deux genres maximum. Or, dans le challenge original, il s'agit de classifier de multiples genres musicaux. Ainsi, nous pensons à un nouveau modèle de classification : les KNN. Après avoir choisi les paramètres du modèle (limitation à un seul voisin notamment), on obtient des performances qui rivalisent totalement avec les régressions logistiques avec 94.82%.

Approfondissement :

Etant donné le grand nombre de variables, nous avons pensé à des méthodes alternatives adaptées à la grande dimension : les *SVM* et les *Random Forest*.

SVM :

Conclusion : La SVM fournit de meilleurs résultats que toutes les méthodes précédentes avec 94.7% de précision. A cela s'ajoute une efficacité algorithmique avantageuse contrairement à son concurrent le KNN.

```
##
## Call:
## svm(formula = Y_train ~ ., data = X_train, type = "C", kernel = "linear",
##     scale = F)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  973
```

```
##
## Call:
## svm(formula = Y_train ~ ., data = X_train, type = "C", kernel = "sigmoid",
##     scale = F)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  sigmoid
##        cost:  1
##      coef.0:  0
##
## Number of Support Vectors:  1298
```

```
##
## Call:
## svm(formula = Y_train ~ ., data = X_train, type = "C", kernel = "polynomial",
##     degree = 2, scale = F)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  1
##      degree:  2
##      coef.0:  0
##
## Number of Support Vectors:  1644
```

Random Forests :

Conclusion : De même que pour les SVM, les Random Forests fournissent des résultats assez bons. De plus, elles ont l'avantage d'être explicatives, ce qui est un avantage très intéressant si l'on veut comprendre pourquoi telle musique a été classifiée tel genre. C'est une méthode qui est beaucoup moins une boîte noire que pourrait l'être une régression logistique.

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

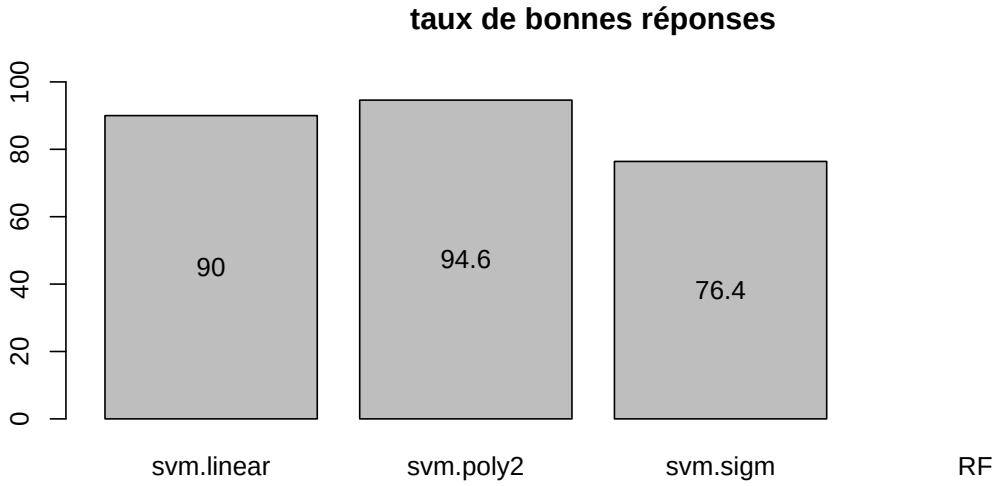
```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
##
## Call:
##  randomForest(formula = Y_train ~ ., data = X_train, ntree = 200)
##                Type of random forest: regression
##                      Number of trees: 200
## No. of variables tried at each split: 63
##
##           Mean of squared residuals: 0.0602218
##                     % Var explained: 75.82
```



Conclusion:

En conclusion, nous remarquons que le modèle le plus performant est celui issu de la méthode des KNN avec un taux de bonne réponses de $94,8\%$. Suivi de la régression de logistique du `stepwise` avec 91% puis Ridge avec $89,5\%$. Cependant il est faut remarquer que de se baser uniquement sur le taux de bonnes réponses n'est pas toujours un critère optimal. En effet dans certains tests il est pertinent de vouloir une plus grande spécificité (par exemple dans les test sérologiques) ou au contraire une plus grande sensibilité (par exemple lors d'une recherche sur un moteur de recherche d'une musique par exemple). Dans le cas de ce choix plus fin, il est nécessaire d'avoir une méthode paramétrique, les KNN sont alors hors-jeux car nous le pouvons pas choisir ce compromis même si la spécificité et sensibilité est bonne, par exemple dans le cas médical on préfère des spécificité de 95% ici elle n'est que de 90%.

Ainsi au sein des méthode paramétriques c'est le modèle du stepwise de *ModAIC* qui l'emporte avec une aire sous courbe la plus importante de $96,6\%$. Néanmoins si nous voulions généraliser notre étude dans le cadre du jeu de données qui initialement contient plus de 2 genres musicaux, il serait nécessaire de préférer l'algorithme des KNN.

C'est pour cela que nous avons approfondi notre étude avec deux méthodes adaptées à cette contrainte. D'une part la *SVM* à noyau polynomial de degré 2 qui est une méthode adaptée aux grandes dimensions et qui permet la classification multiple. Et d'autre part l'algorithme des *Random Forest*, qui permet d'ajouter de l'explicativité à la prise de décision d'une classe. Finalement ces deux méthodes donnent des performances comparables aux autres traitées dans ce sujet.

En terme de complexité algorithmique l'avantage des régressions logistique ou de ridge est qu'une fois les coefficients calculés et le seuil défini, la complexité est de l'ordre du nombre de variables. Contrairement aux KNN où la complexité est bien plus élevée car au moins quadratique en nombre d'individus. Les randoms forest quant à elles bénéficient d'une complexité logarithmique ! Ainsi en fonction de l'usage de cette classification et des ressources à disposition ce critère peut être déterminant.