Détermination de modèles linéaires et prédiction de variables quantitatives et qualitatives

EMPIS - RABHI

Contents

I - Prédiction d'une variable quantitative

1. Présentation des données

Le jeu de données contient 20 variables et 200 observations.

On cherche à prédire la variable Salary.

Extrait des données :

	AtBat	Hits	HmRun	Runs	RBI	Walks	Years	CAtBat	CHits	CHmRun	CRuns
-Alan Ashby	315	81	7	24	38	39	14	3449	835	69	321
-Alvin Davis	479	130	18	66	72	76	3	1624	457	63	224
-Andre Dawson	496	141	20	65	78	37	11	5628	1575	225	828
-Andres Galarraga	321	87	10	39	42	30	2	396	101	12	48
-Alfredo Griffin	594	169	4	74	51	35	11	4408	1133	19	501
-Al Newman	185	37	1	23	8	21	2	214	42	1	30

	CRBI	CWalks	League	Division	PutOuts	Assists	Errors	Salary	NewLeague
-Alan Ashby	414	375	N	W	632	43	10	475.0	N
-Alvin Davis	266	263	A	W	880	82	14	480.0	A
-Andre Dawson	838	354	N	E	200	11	3	500.0	N
-Andres Galarraga	46	33	N	E	805	40	4	91.5	N
-Alfredo Griffin	336	194	A	W	282	421	25	750.0	A
-Al Newman	9	24	N	E	76	127	7	70.0	A

2. Recherche du meilleur modèle

Modèle linéaire complet :

```
modlin=lm(Salary~.,data=train_Hitters)
```

Il existe plusieurs méthodes pour trouver le meilleur modèle.

On va utiliser la méthode de l'AIC, il y a plusieurs options :

- backward : débute avec toutes les variables candidates et supprime la variable en suivant le critère d'adéquation du modèle choisi puis répète ce processus tant que d'autres variables peuvent être supprimées.
- forward : débute avec aucune variable candidate et ajoute la variable à l'aide d'un critère d'ajustement du modèle choisi dont l'inclusion donne l'amélioration la plus statistiquement significative puis répète ce processus jusqu'à ce que aucune n'améliore le modèle dans une mesure statistiquement significative.
- both : une combinaison de ce qui précède, testant à chaque étape les variables à inclure ou à exclure.

Mise en place de la méthode :

```
modselect_a=stepAIC(modlin,~.,trace=TRUE,direction=c("backward"))
modselect_b=stepAIC(modlin,~.,trace=TRUE,direction=c("forward"))
modselect_c=stepAIC(modlin,~.,trace=TRUE,direction=c("both"))
```

Regardons maintenant les AIC obtenus avec les différentes options :

AIC(modselect_a) = 2873.5065114

AIC(modselect_b) = 2887.9073264

AIC(modselect_c) = 2873.5065114

On sélectionne le modèle qui a l'AIC le plus petit (ici 2873.5065114) :

Salary ~ AtBat + Hits + Walks + CAtBat + CRuns + CRBI + CWalks + Division + PutOuts + Assists

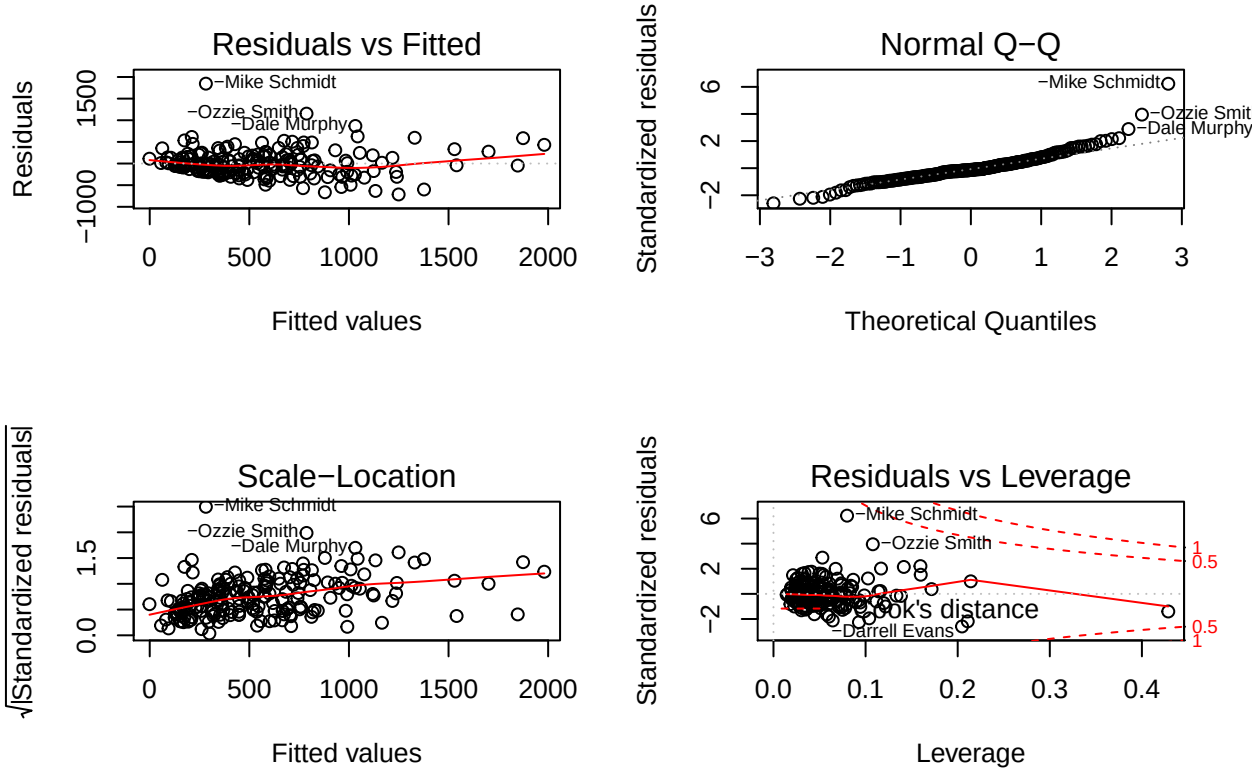
Définition du modèle :

```
model1<-lm(Salary ~ AtBat + Hits + Walks + CAtBat + CRuns + CRBI +
             CWalks +  Division + PutOuts + Assists,data=train_Hitters)
```

Voici les caractéristiques de notre modèle :

```
##  (Intercept)        AtBat         Hits        Walks       CAtBat
##  168.9376916   -2.8331643    8.8559520    6.1391263   -0.2043903
##        CRuns         CRBI       CWalks    DivisionW      PutOuts
##    1.5953069    1.0768414   -0.7515933 -119.5206406    0.4029913
##      Assists
##    0.5618160
```

Représentation des caractéristiques du modèle :



Residuals vs Fitted : on a des résidus majoritairement répartis autour d'une ligne horizontale sans motifs distincts donc on n'a pas de relation non linéaire. Donc l'hypothèse que le modèle choisi est adéquat est validé.

Normal Q-Q : les résidus suivent bien une ligne droite et ne s'écartent pas considérablement. Donc, les résidus sont normalement distribués.

Scale-Location : les résidus sont répartis aléatoirement et la ligne ne présente aucun angle, elle est presque horizontale. Donc, les résidus sont répartis de manière égale le long des plages de prédicteurs.

3. Evaluation de la qualité de ce modèle sur l'ensemble d'apprentissage et de test

Calcul du RMSE :

```
predTrain <- predict(model1,newdata=train_Hitters)
errors<-predTrain-train_Hitters$Salary
rmse_T <- sqrt(mean(errors^2))
```

Sur les données d'apprentissage, on obtient un RMSE de : 300.3137451

Sur les données de test, on obtient un RMSE de : 342.8452973

Evaluons maintenant la qualité de ce modèle sur l'ensemble d'apprentissage en utilisant MSEP sur une procédure de vérification croisée :

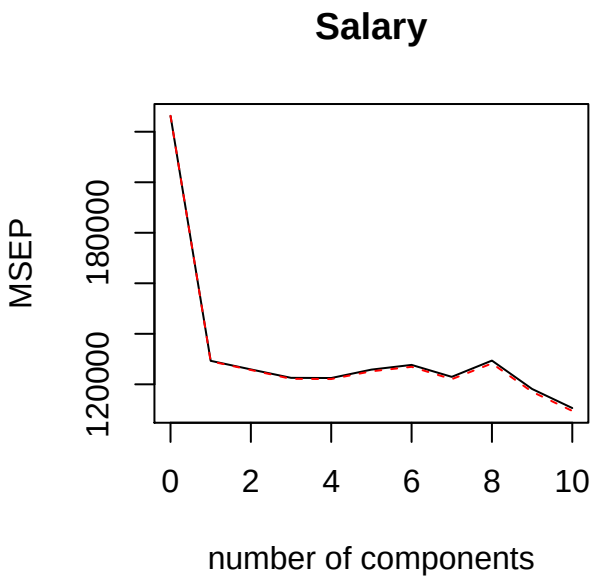
```
set.seed(42)
index <- sample(1:nrow(train_Hitters),50)
train_Hitters_cv <- train_Hitters[index,]
```

```
model_caret <- train(Salary ~ AtBat + Hits + Walks + CAtBat + CRuns +
                       CRBI + CWalks + Division + PutOuts + Assists ,
                     data = train_Hitters_cv,method = "lm",
                     trControl = trainControl(
                       method = "cv", number =10,
                       verboseIter = TRUE)
)
prediction_model_c <- predict(model_caret,data=train_Hitters_cv)
pcr_fit <- pcr(Salary~ AtBat + Hits + Walks + CAtBat + CRuns +
                 CRBI + CWalks + Division + PutOuts + Assists,
               data = train_Hitters, scale = TRUE, validation = "CV")
```

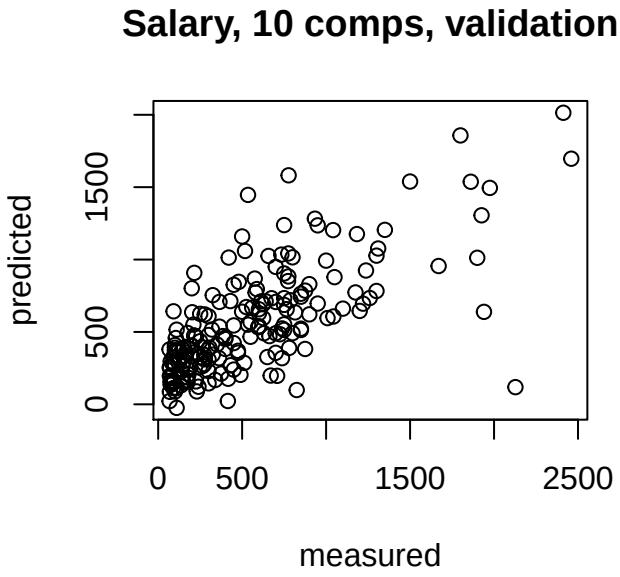
```
model_caret
```

```
## Linear Regression
##
## 50 samples
## 10 predictors
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 45, 43, 45, 45, 45, 44, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   279.1267  0.6414183  219.2121
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
validationplot(pcr_fit, val.type = "MSEP")
```



```
predplot(pcr_fit)
```



II - Prédiction d'une variable qualitative

On cherche maintenant à prédire une variable qualitative

1. Présentation des données

Extrait des données à disposition :

	Y	X.1	X.2	X.3	X.4	X.5	X.6	X.7	X.8
V1	0	0.7733437	-2.438405	-0.4825622	-2.7211350	-1.2170580	0.8278092	1.342604	0.0570417
V2	0	-0.0781778	-2.415754	0.4127717	-2.8251460	-0.6262365	0.0544882	1.429498	-0.1202486
V3	0	-0.0844692	-1.649739	-0.2413075	-2.8752860	-0.8894054	-0.0274740	1.159300	0.0156765
V4	0	0.9656140	-2.380547	0.6252965	-1.7412560	-0.8453664	0.9496868	1.093801	0.8197358
V5	0	0.0756639	-1.728785	0.8526265	0.2726953	-1.8413700	0.3279359	1.251219	0.7714499
V6	0	0.4588163	-2.875286	0.1358412	0.4053984	-2.0826470	0.1378471	1.733530	0.3964244

2. Recherche du meilleur modèle

On cherche le meilleur modèle en utilisant la méthode lasso :

```
model3<-glmnet(data_Khan$X, data_Khan$Y, family = "binomial", alpha =1)
```

On calcule le RMSE :

```
predTrain3 <- predict(model3,data_Khan$X,type='response')
errors3<-predTrain3-data_Khan$Y
rmse_T3 <- sqrt(mean(errors3**2))
```

On obtient le résultat suivant : 0.1923336

-> RMSE (Root-Mean-Square-Error) est une mesure fréquemment utilisée pour représenter l'écart entre les valeurs prédites et les valeurs réelles. Ici, nous avons un RMSE égal à 0.19, donc notre modèle est bon.

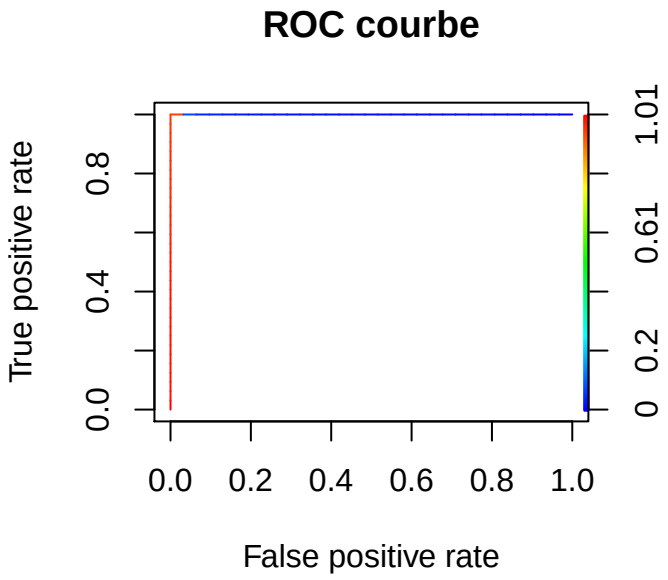
3. Évaluation de la qualité de ce modèle à l'aide de la courbe ROC et l'AUC sur l'ensemble d'apprentissage et en utilisant une procédure de validation croisée.

On utilise la procédure de validation croisée afin de trouver le lambda qui minimise l'erreur. On l'utilise ensuite pour créer le modèle "md_lasso".

Courbe ROC (Receiver operating characteristic) : graphique représentant les performances d'un modèle de classification pour tous les seuils de classification. Cette courbe trace le taux de vrais positifs en fonction du taux de faux positifs.

```
cv_lasso <- cv.glmnet(data_Khan$X, data_Khan$Y, family = "binomial", nfold = 10,
                      type.measure = "deviance",alpha = 1, paralle = TRUE)
md_lasso <- glmnet(data_Khan$X, data_Khan$Y, family = "binomial",
                   lambda = cv_lasso$lambda.1se, alpha = 1)
P_lasso <-predict(md_lasso,data_Khan$X, type = "response")

pred_lasso <- prediction(P_lasso,data_Khan$Y)
roc_lasso <- performance(pred_lasso,"tpr", "fpr")
plot(roc_lasso,colorize = TRUE,main="ROC courbe")
```




```
auc_lasso <- performance( pred_lasso, "auc")@y.values[[1]]
```

AUC signifie "aire sous la courbe ROC". Cette valeur mesure l'intégralité de l'aire à deux dimensions située sous l'ensemble de la courbe ROC (par calculs d'intégrales) de (0,0) à (1,1), en gros c'est un résumé de la précision du modèle avec un seul chiffre.

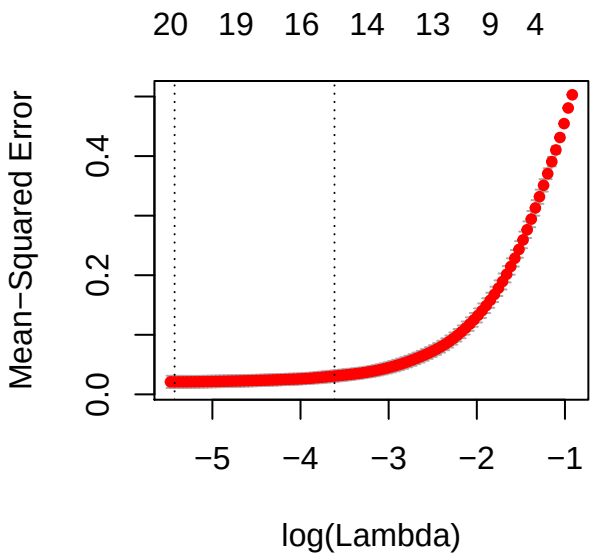
Remarque : l'AUC est pertinente pour évaluer la qualité d'un modèle parce qu'elle est :

- invariante d'échelle
- indépendante des seuils de classification

-> **On trouve ici l'AUC = 1 donc le modèle est bon.**

4. Evaluation de l'erreur de classification erronée sur le kit d'apprentissage en utilisant une procédure de validation croisée.

On utilise plot pour tracer la courbe de validation croisée produite par cv.glmnet.



De plus, on voulait profiter encore de "cv.out1" puisque la fonction cv.glmnet à part l'objet de classe "cv.glmnet" et "lambda", retourne aussi un objet "cvm"" qui est un vecteur de longueur = length(lamba) et contient l'ensemble des erreurs : The mean cross-validated error.

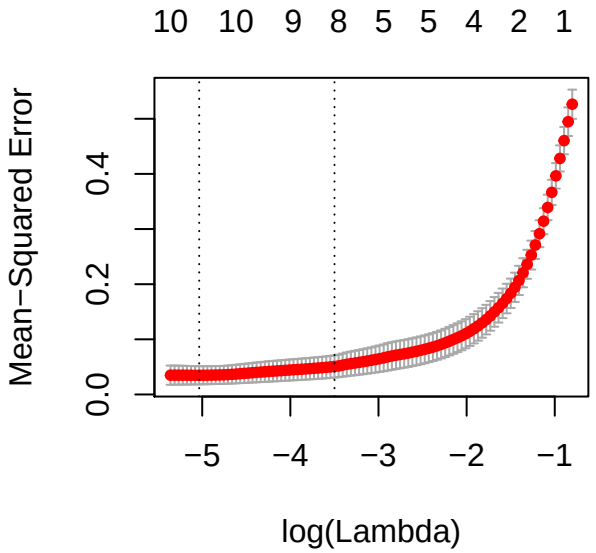
Mais aussi un objet "cvsd" qui présente ces erreurs d'une manière standardisée.

```
cv.out1$cvsd
```

Extrait :

```
## [1] 0.003336644 0.003664383 0.003753824 0.004939437 0.006409641 0.008005432
```

5. Evaluation de l'erreur de classification sur les données de test



Conclusion

Au cours de ce projet, nous avons eu l'occasion d'exploiter différentes librairies (MASS, caret, pls, glmnet, ROCR, caTools), d'analyser des données, de mettre en place des modèles linéaires et de comprendre les mécanismes pour réaliser la prédiction des données.