TP1 - Logiciel R et prédiction de l'efflorescence algale

Apprentissage statistique
Chargé de TD : Christophe Denis

Pierre ALLAIN, Benoît CHOFFIN
25 octobre 2016

Question 1. (1.a) La commande lm de R gère les variables catégorielles en modélisant chaque modalité par une indicatrice (valant 1 si l'instance appartient bien à la catégorie en question et 0 sinon). La modalité choisie comme référence n'est quant à elle représentée par aucune variable. Les valeurs des coefficients correspondant aux différentes modalités doivent donc être interprétées relativement à la modalité de référence.

On utilise le coefficient de détermination ajusté (R^2_{adj}) pour mesurer la qualité d'ajustement par un modèle linéaire. Sa valeur est ici de 0.3204. Le coefficient de détermination ajusté tient compte du nombre de variables, contrairement au R^2 qui croît nécessairement avec le nombre de variables explicatives. Or, on sait qu'un excès de variables produit des modèles peu robustes.

(1.b) D'après les résultats du test de Fisher, les variables `Season`, `NH4` et `Chla` ne sont pas signficatives (même pour un seuil de 20%). Ces variables sont donc inutiles pour prévoir la variable `a1`.

(1.c) Les variables retenues pour le modèle final sont les suivantes : `size, mxPH, mnO2, NO3, NH4, PO4`. On rappelle que la variable `size` contient trois modalités, deux d'entre elles – small et medium – représentées chacune par une indicatrice, et la modalité de référence (large) qui n'apparaît pas. Le R^2 normal ne peut que diminuer, car on a retiré des variables au modèle précédent. Ce n'est donc pas un critère pertinent. En revanche, le R^2 ajusté a légèrement augmenté, dénotant une meilleure qualité d'ajustement par ce modèle linéaire.

Question 2. Pour compléter la commande destinée à nettoyer les données manquantes, on a utilisé la fonction knnImputation(), fournie par le package DMwR, qui remplace les valeurs manquantes d'une variable quantitative par la médiane des 10 valeurs de cette variable correspondantes aux observations les plus proches de celle qui contient la valeur

manquante. Dans le cas d'une variable catégorielle, c'est la modalité la plus fréquente parmi les 10 observations les plus proches qui est utilisée. On complète les commandes de prédiction de la manière suivante. Pour la prédiction basée sur le modèle de régression multiple, on choisit comme modèle celui déterminé dans la question (1.c) (noté `final.lm`) et comme données à prédire les 140 observations de test. De même, pour la prédiction basée sur l'arbre de décision : on choisit comme modèle celui de la question (2.a) noté `rt.a1` et on cherche à prédire la variable d'intérêt pour les 140 observations de test. On obtient alors les deux vecteurs de prédictions correspondant aux deux méthodes.

Question 3. Pour évaluer la qualité des prédictions fournies par les modèles précédents sur la variable `a1`, on utilise d'abord les deux lignes de codes suivantes :

```
> regr.eval(algae.sols$a1, lm.predictions.a1, train.y = algae[,"a1"])
> regr.eval(algae.sols$a1, rt.predictions.a1, train.y = algae[,"a1"])
```

La fonction `regr.eval` prend ici en premier argument le vecteur des vraies valeurs de la base de test de `a1` (donné par `algae.sols`), en deuxième argument les prédictions du modèle considéré pour la base de test, et en troisième argument le vecteur des vraies valeurs de la base d'entraînement pour le calcul de deux métriques (*NMSE* et *NMAE*). Cette étape d'évaluation des modèles sur une base de test est centrale car elle permet de mesurer leur surapprentissage.

Le tableau suivant reporte les principales métriques permettant de comparer les deux modèles entre eux sur la base de test (le *MAPE* n'est pas indiqué car certaines valeurs de `a1` étant nulles, son mode de calcul induit une valeur infinie pour la métrique) :

TABLE 1 – Comparaison des performances de prédiction des deux modèles selon différentes métriques (variable `a1` ; base de test)

Métriques / Modèles	MAE	MSE	RMSE	NMSE	NMAE
Régression linéaire multiple	12.52	**276.44**	**16.63**	**0.66**	0.78
Arbre de décision	**11.65**	285.74	16.90	0.68	**0.73**

MAE signifie *Mean Absolute Error*, *MSE*, *Mean Squared Error*, *RMSE*, *Root Mean Square Error*, *NMSE*, *Normalized Mean Squared Error*, *NMAE*, *Normalized Mean Absolute Error* et *MAPE*, *Mean Absolute Percentage Error*. Leurs formules de calcul sont les suivantes :

1. $MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i|$ avec $e_i = y_i - \hat{y}_i$, y_i la vraie valeur et $\hat{y}_i$ la valeur prédite.
2. $MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$
3. $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$

4. $MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{y_i-\hat{y}_i}{y_i}|$
5. $NMSE = \frac{\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}{\sum_{i=1}^{n}(y_i-\text{moy}(Y))^2}$ avec moy(Y) la moyenne des valeurs de la variable à prédire sur la base d'entraînement.
6. $NMAE = \frac{\sum_{i=1}^{n}|y_i-\hat{y}_i|}{\sum_{i=1}^{n}|y_i-\text{moy}(Y)|}$

On constate que, de manière générale, sur toutes les métriques considérées, les deux modèles ont des performances prédictives similaires et relativement médiocres.

Le modèle de régression linéaire multiple a un moins bon MAE que l'arbre de décision, mais son MSE et son RMSE sont meilleurs. Ces deux dernières métriques pénalisent plus les écarts extrêmes de prédiction, on peut donc supposer que l'arbre de décision a une distribution des erreurs plus éclatée que la régression multilinéaire.

On remarque également que, à l'inverse de la comparaison sur la base d'entraînement, l'arbre de décision n'a pas des performances uniformément meilleures sur toutes les métriques. En effet, les modèles d'arbre ont tendance à être particulièrement sujets au surapprentissage.

La figure suivante présente une comparaison des vraies valeurs et des valeurs prédites sur la base de test pour les deux modèles :

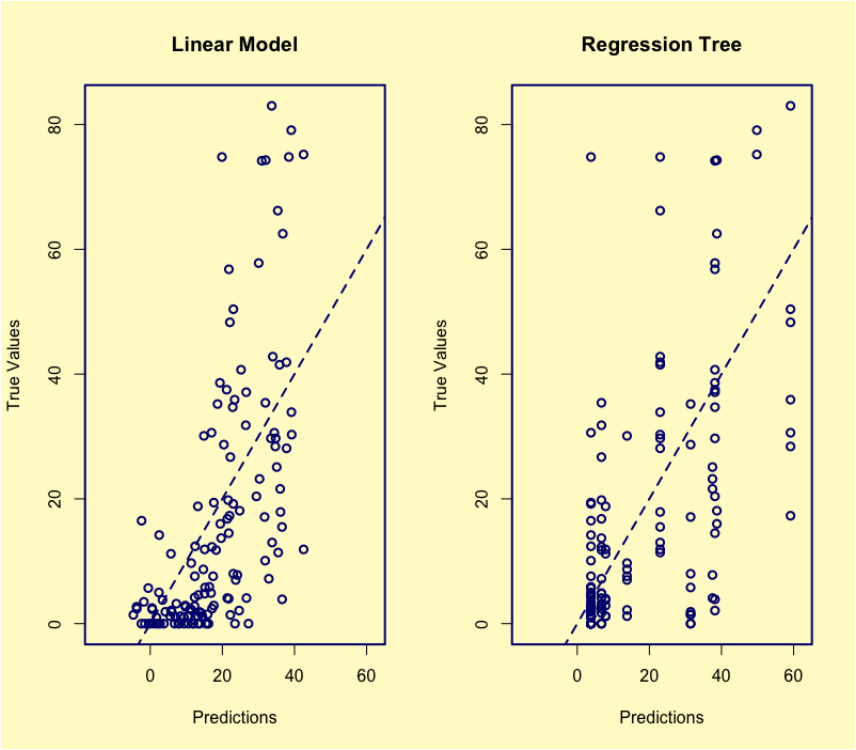

FIGURE 1 – Représentation des valeurs prédites et des vraies valeurs pour la variable `a1` (base de test)

Question 4. Le code utilisé pour générer les résultats à cette question peut être trouvé en annexe.

Les tableaux 2 à 7 donnent les résultats des modèles prédictifs pour chaque variable, de `a2` à `a7` :

TABLE 2 – Comparaison des performances de prédiction des deux modèles selon différentes métriques (variable `a2` ; base de test)

Modèles \ Métriques	MAE	MSE	RMSE	NMSE	NMAE
Régression linéaire multiple	**6.86**	**102.67**	**10.13**	**0.96**	**0.90**
Arbre de décision	7.09	117.71	10.85	1.10	0.93

TABLE 3 – Comparaison des performances de prédiction des deux modèles selon différentes métriques (variable `a3` ; base de test)

Modèles \ Métriques	MAE	MSE	RMSE	NMSE	NMAE
Régression linéaire multiple	**3.86**	**28.27**	**5.32**	**0.90**	**0.86**
Arbre de décision	4.39	40.64	6.38	1.29	0.98

TABLE 4 – Comparaison des performances de prédiction des deux modèles selon différentes métriques (variable `a4` ; base de test)

Modèles \ Métriques	MAE	MSE	RMSE	NMSE	NMAE
Régression linéaire multiple	1.88	**7.68**	**2.77**	**0.98**	**0.95**
Arbre de décision	**1.80**	8.43	2.90	1.07	0.91

TABLE 5 – Comparaison des performances de prédiction des deux modèles selon différentes métriques (variable `a5` ; base de test)

Modèles \ Métriques	MAE	MSE	RMSE	NMSE	NMAE
Régression linéaire multiple	5.44	**80.62**	**8.98**	**0.87**	0.89
Arbre de décision	**5.23**	90.21	9.50	0.98	**0.86**

TABLE 6 – Comparaison des performances de prédiction des deux modèles selon différentes métriques (variable `a6` ; base de test)

Modèles \ Métriques	MAE	MSE	RMSE	NMSE	NMAE
Régression linéaire multiple	**7.22**	145.60	12.07	0.81	0.85
Arbre de décision	6.82	**140.31**	**11.85**	**0.78**	**0.81**

TABLE 7 – Comparaison des performances de prédiction des deux modèles selon différentes métriques (variable `a7` ; base de test)

Modèles \ Métriques	MAE	MSE	RMSE	NMSE	NMAE
Régression linéaire multiple	2.90	24.00	4.90	1.11	1.08
Arbre de décision	**2.48**	**22.62**	**4.76**	**1.05**	**0.92**

De manière générale, les performances prédictives des régressions linéaires multiples sont meilleures que celles des arbres de décision, sauf pour les deux dernières variables `a6` et `a7`. Mais là encore, les métriques sont généralement similaires.

En outre, les modèles ont des performances différenciées selon la variable : ainsi, les deux modèles sont plus performants pour la variable `a4` que pour la variable `a6` (respectivement 2.77 et 2.90 pour le *RMSE* de la régression multilinéaire et de l'arbre de décision contre 12.07 et 11.85).

Enfin, les figures 2 à 7 comparent valeurs prédites et vraies valeurs (sur la base de test) pour les deux modèles sur chaque variable.

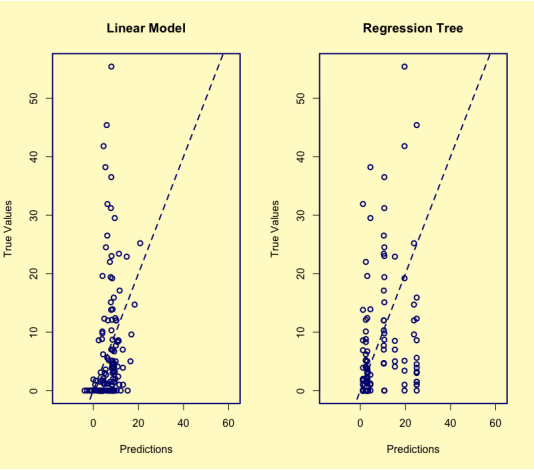

FIGURE 2 – Variable `a2`

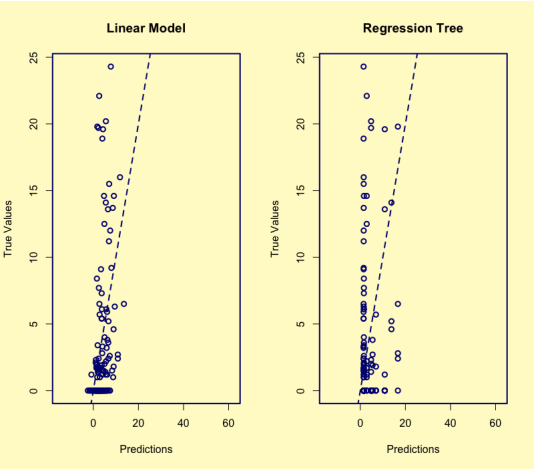

FIGURE 3 – Variable `a3`

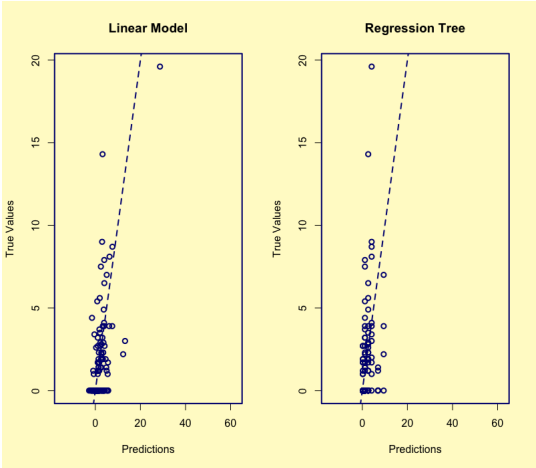

FIGURE 4 – Variable a4

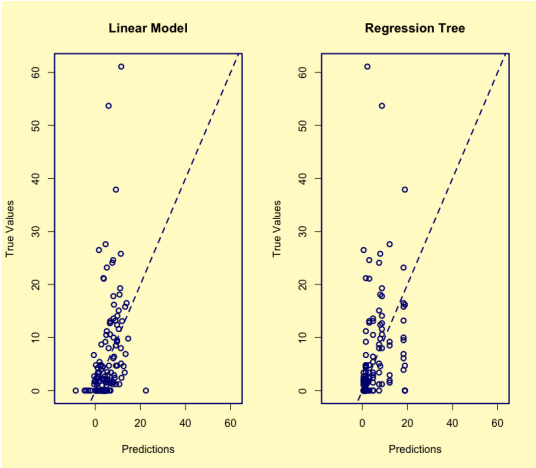

FIGURE 5 – Variable a5

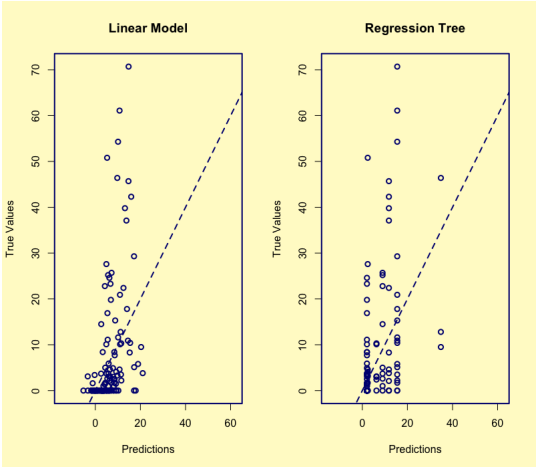

FIGURE 6 – Variable a6

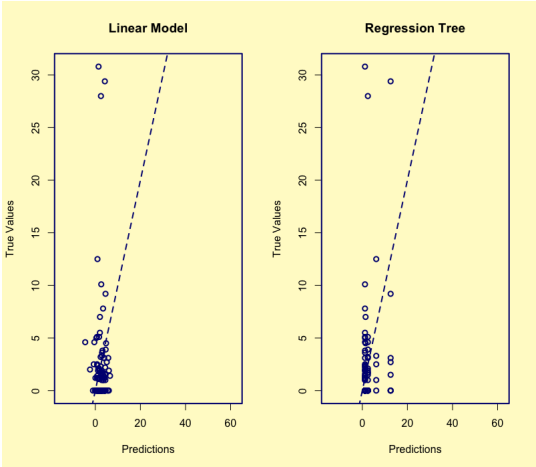

FIGURE 7 – Variable a7

Annexes - Code R

```
\begin{framed}
rm(list = ls())

#Ouverture des librairies
library(DMwR)

#D but du dataset
head(algae)

#R sum  stat des variables
summary(algae)

#Histogramme, estimateur    noyau de la densit  et QQ-plot
library(car)
op = par(mfrow=c(1,2))
hist(algae$mxPH, prob=T, xlab="",
     main="Histogram of maximum pH value",ylim=0:1)
lines(density(algae$mxPH,na.rm=T))
rug(jitter(algae$mxPH))
qqnorm(algae$mxPH,main="Normal QQ plot of maximum pH")
par(op)

#Boxplot conditionnelle    la variable cat gorielle size
library(lattice)
bwplot(size ~ a1, data=algae, ylab="River Size",xlab="Algal A1")

### Donn es manquantes ###
algae[!complete.cases(algae),]
nrow(algae[!complete.cases(algae),])

#Suppression des lignes avec NA --> cr ation de algae1
algae1 = na.omit(algae)
nrow(algae)
nrow(algae1)

#On remplace les NAs par valeurs plus fr quentes
algae2 = centralImputation(algae)
nrow(algae[!complete.cases(algae),])
nrow(algae2[!complete.cases(algae2),])

#Avec m thode stat
algae3 = knnImputation(algae, k =10, meth = "median")
nrow(algae[!complete.cases(algae),])
nrow(algae3[!complete.cases(algae3),])

### R gression lin aire multiple ###
algae = knnImputation(algae, k =10, meth = "median")
lm.a1 <- lm(a1 ~ ., data = algae[, 1:12])
summary(lm.a1)


anova(lm.a1)


final.lm = step(lm.a1)

summary(final.lm)

### Decision trees ###
algae = knnImputation(algae, k =10, meth = "median")
library(rpart)
rt.a1 = rpart(a1 ~ ., data = algae[, 1:12])
rt.a1

#Affichage de l'arbre de d cision obtenu :
par(lwd=2, col="red")
plot(rt.a1, compress=TRUE)
text(rt.a1, use.n=TRUE,col="blue")

#Affichage alternatif :
par(lwd=2, bg="lemonchiffon3")
prettyTree(rt.a1,col="navy",bg="lemonchiffon")

#Qualit  de pr diction :
lm.predictions.a1 = predict(final.lm, algae)
rt.predictions.a1 = predict(rt.a1, algae)
regr.eval(algae[, "a1"], rt.predictions.a1, train.y = algae[,"a1"])
regr.eval(algae[, "a1"], lm.predictions.a1, train.y = algae[,"a1"])

#Plot des valeurs pr dites vs observ es
par(mfrow = c(1, 2), col="navy", bg="lemonchiffon1")
plot(lm.predictions.a1, algae[, "a1"], main = "Linear Model",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)
plot(rt.predictions.a1, algae[, "a1"], main = "Regression Tree",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)

#On fait la m me chose, mais sur la base de test
summary(test.algae)

#Imputation de valeurs manquantes
test.algae = knnImputation(test.algae, k =10, meth = "median")
#Pr dictions sur base de test
lm.predictions.a1 = predict(final.lm,test.algae)
rt.predictions.a1 = predict(rt.a1,test.algae)

#Evaluation des performances des mod les sur la base de test
regr.eval(algae.sols$a1, lm.predictions.a1, train.y = algae[,"a1"])
regr.eval(algae.sols$a1, rt.predictions.a1, train.y = algae[,"a1"])

#Et maintenant sur les variables a2    a7
#a2
```

```
lm.a2 <- lm(a2 ~ ., data = algae[, c(1:11,13)])
final.lm = step(lm.a2)

rt.a2 = rpart(a2 ~ ., data = algae[, c(1:11,13)])

lm.predictions.a2 = predict(final.lm,test.algae)
rt.predictions.a2 = predict(rt.a2,test.algae)

regr.eval(algae.sols$a2, lm.predictions.a2,train.y = algae[,"a2"])
regr.eval(algae.sols$a2, rt.predictions.a2,train.y = algae[,"a2"])

par(mfrow = c(1, 2), col="navy", bg="lemonchiffon1")
plot(lm.predictions.a2, algae[, "a2"], main = "Linear Model",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)
plot(rt.predictions.a2, algae[, "a2"], main = "Regression Tree",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)


#a3
lm.a3 <- lm(a3 ~ ., data = algae[, c(1:11,14)])
final.lm = step(lm.a3)

rt.a3 = rpart(a3 ~ ., data = algae[, c(1:11,14)])

lm.predictions.a3 = predict(final.lm,test.algae)
rt.predictions.a3 = predict(rt.a3,test.algae)

regr.eval(algae.sols$a3, lm.predictions.a3,train.y = algae[,"a3"])
regr.eval(algae.sols$a3, rt.predictions.a3,train.y = algae[,"a3"])

par(mfrow = c(1, 2), col="navy", bg="lemonchiffon1")
plot(lm.predictions.a3, algae[, "a3"], main = "Linear Model",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)
plot(rt.predictions.a3, algae[, "a3"], main = "Regression Tree",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)

#a4
lm.a4 <- lm(a4 ~ ., data = algae[, c(1:11,15)])
final.lm = step(lm.a4)

rt.a4 = rpart(a4 ~ ., data = algae[, c(1:11,15)])

lm.predictions.a4 = predict(final.lm,test.algae)
rt.predictions.a4 = predict(rt.a4,test.algae)

regr.eval(algae.sols$a4, lm.predictions.a4,train.y = algae[,"a4"])
regr.eval(algae.sols$a4, rt.predictions.a4,train.y = algae[,"a4"])

par(mfrow = c(1, 2), col="navy", bg="lemonchiffon1")
plot(lm.predictions.a4, algae[, "a4"], main = "Linear Model",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)
plot(rt.predictions.a4, algae[, "a4"], main = "Regression Tree",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)

#a5
lm.a5 <- lm(a5 ~ ., data = algae[, c(1:11,16)])
final.lm = step(lm.a5)

rt.a5 = rpart(a5 ~ ., data = algae[, c(1:11,16)])

lm.predictions.a5 = predict(final.lm,test.algae)
rt.predictions.a5 = predict(rt.a5,test.algae)

regr.eval(algae.sols$a5, lm.predictions.a5,train.y = algae[,"a5"])
regr.eval(algae.sols$a5, rt.predictions.a5,train.y = algae[,"a5"])

par(mfrow = c(1, 2), col="navy", bg="lemonchiffon1")
plot(lm.predictions.a5, algae[, "a5"], main = "Linear Model",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)
plot(rt.predictions.a5, algae[, "a5"], main = "Regression Tree",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)

#a6
lm.a6 <- lm(a6 ~ ., data = algae[, c(1:11,17)])
final.lm = step(lm.a6)

rt.a6 = rpart(a6 ~ ., data = algae[, c(1:11,17)])

lm.predictions.a6 = predict(final.lm,test.algae)
rt.predictions.a6 = predict(rt.a6,test.algae)

regr.eval(algae.sols$a6, lm.predictions.a6,train.y = algae[,"a6"])
regr.eval(algae.sols$a6, rt.predictions.a6,train.y = algae[,"a6"])

par(mfrow = c(1, 2), col="navy", bg="lemonchiffon1")
plot(lm.predictions.a6, algae[, "a6"], main = "Linear Model",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)
plot(rt.predictions.a6, algae[, "a6"], main = "Regression Tree",
     xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
abline(0, 1, lty = 2)

#a7
lm.a7 <- lm(a7 ~ ., data = algae[, c(1:11,18)])
final.lm = step(lm.a7)

rt.a7 = rpart(a7 ~ ., data = algae[, c(1:11,18)])
```

```
209
210 lm.predictions.a7 = predict(final.lm,test.algae)
211 rt.predictions.a7 = predict(rt.a7,test.algae)
212
213 regr.eval(algae.sols$a7, lm.predictions.a7,train.y = algae[,"a7"])
214 regr.eval(algae.sols$a7, rt.predictions.a7,train.y = algae[,"a7"])
215
216 par(mfrow = c(1, 2), col="navy", bg="lemonchiffon1")
217 plot(lm.predictions.a7, algae[, "a7"], main = "Linear Model",
218      xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
219 abline(0, 1, lty = 2)
220 plot(rt.predictions.a7, algae[, "a7"], main = "Regression Tree",
221      xlab = "Predictions", ylab = "True Values", xlim=c(-15,62))
222 abline(0, 1, lty = 2)
223 \end{framed}
```