Predictive Modeling

Series 10

Exercise 10.1

In this exercise, we will look at the `College` data set which tracks demographic characteristics of college applications in the USA. The dataset has the following structure:

- `Private`: If the applicant is from a public or private high school
- `Apps`: Number of applications received
- `Accept`: Number of applicants accepted
- `Enroll`: Number of new students enrolled
- `Top10perc`: New students from top 10% of high school class
- `Top25perc`: New students from top 25% of high school class
- `F.Undergrad`: Number of full-time undergraduates
- `P.Undergrad`: Number of part-time undergraduates
- `Outstate`: Out-of-state tuition
- `Room.Board`: Room and board costs
- `Books`: Estimated book costs
- `Personal`: Estimated personal spending
- `PhD`: Percent of faculty with Ph.D.'s
- `Terminal`: Percent of faculty with terminal degree
- `S.F.Ratio`: Student/faculty ratio
- `perc.alumni`: Percent of alumni who donate
- `Expend`: Instructional expenditure per student
- `Grad.Rate`: Graduation rate

You can read in the data by using

```r
college <- read.csv("college.csv")
# or using the ISLR package
library(ISLR)
college <- ISLR::College

# inspect the data using
View(college)
```

Our goal is to predict the number of applications (`Apps`) received by using the other variables in the dataset.

a) Split the data set into a training and a test set.

b) Fit a linear model using least squares and best subset selection on the training set, and report the test error obtained. **Hint:** Use `regsubsets` from the `leaps` library to find the best subset using BIC

c) Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

d) Fit a lasso model on the training set, with λ chosen by cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates. **Hint:** Use `cv.glmnet` with different α from the `glmnet` library to calculate a cross-validated Ridge and Lasso regression

Exercise 10.2

In this exercise, we will examine the `Boston` data set which contains housing values in 506 suburbs of Boston and a variety of census values. The dataset is constructed as follows:

- `crim`: Per capita crime rate by town
- `zn`: Proportion of residential land zoned for lots over 25'000 square feet
- `indus`: Proportion of non-retail business acres per town
- `chas`: Charles River dummy variable (=1 if tract bounds river, 0 otherwise)
- `nox`: Nitrogen oxides concentration (parts per 10 million)
- `rm`: Average number of rooms per dwelling
- `age`: Proportion of owner-occupied units built prior to 1940
- `dis`: Weighted mean of distances to five Boston employment centres
- `rad`: Index of accessibility to radial highways
- `tax`: Full-value property-tax rate per $10,000
- `ptratio`: Pupil-teacher ratio by town
- `lstat`: Lower status of the population (percent)
- `medv`: Median value of owner-occupied homes in $1000s

You can read in the data by means of

[1]:
```
boston <- read.csv('boston.csv')
# or using the ISLR2 package
boston <- boston <- ISLR2::Boston

# inspect the data using
View(boston)
```

We will try to predict the per capita crime rate (`crim`) in the data set via the other variables.

a) Split the data set into a training and a test set.

b) Fit a linear model using least squares and best subset selection on the training set, and report the test error obtained.

c) Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

d) Fit a lasso model on the training set, with λ chosen by crossvalidation. Report the test error obtained, along with the number of non-zero coefficient estimates.

Exercise 10.3

In this exercise, we will take a look at some model agnostic feature importance metrics for the `College` dataset

a) Plot the Partial Dependence Plot for each variable on the training and test set

b) Plot the Accumulated Local Effects for each variable on the training and test set

c) Plot the Permutation Feature Importance for each variable on the training and test set

d) Analyse the various feature importance metrics, if and why they highlight different features as significant

Hint: All these plots can be generated with the `iml` package

Exercise 10.4

In this exercise, we will have look at a toolbox called `caret` which can be used to test a variety of models (classification, regression, and anomaly detection) on a dataset and get an intuition about which models may be followed up.

In contrast to the `PyCaret` package which can be used to gain an overview over various models in a short timespan, the `caret` package for `R` is much more designed to be a pipeline for machine learning.

Carry out a regression analysis on the `College` data set with the target variable `Apps` on the training set, and compare your results especially with respect to the feature importance on the training and test data set.

Result Checker

Predictive Modeling

Solutions to Series 10

Solution 10.1 Transform the predictor variable `Private` to a factor variable.

Read in the data and split it into training and test set

```
library(caTools)
library(ISLR)
college <- ISLR::College
set.seed(42)
college$Private <- as.factor(college$Private)
sample <- sample.split(college$Apps, SplitRatio = 0.7)
train <- college[sample == TRUE, ]
test <- college[sample == FALSE, ]
```

Let us find the best subset of predictor variables.

```
library(leaps)
best_subset <- regsubsets(Apps~.,
    data = train,
    nbest = 1,       # For each number of predictors how many models
                     # should be recorded
    nvmax = NULL,    # NULL for no limit on number of variables
    force.in = NULL,
    force.out = NULL,
    method = "exhaustive"
)
summary_best_subset <- summary(best_subset)
num_predictors <- which.min(summary_best_subset$bic)
num_predictors

## [1] 8
```

The best subset we found on the basis of the Bayesian Information Criterion consists of 8 predictor variables, let us fit a linear model with these eight predictor variables.

```r
predictors = names(which(summary_best_subset$which[8, ]))
predictors
```

```
## [1] "(Intercept)" "PrivateYes"  "Accept"
## [4] "Enroll"      "Top10perc"   "Outstate"
## [7] "Room.Board"  "PhD"         "Expend"
```

```r
lin_reg02 <- lm(Apps ~ Private + Accept + Enroll + Top10perc +
    Outstate + Room.Board + PhD + Expend, data = train)
summary(lin_reg02)
```

```
##
## Call:
## lm(formula = Apps ~ Private + Accept + Enroll + Top10perc + Outstate +
##     Room.Board + PhD + Expend, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5393.7  -420.5     3.8   308.7  7690.2
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -78.25577  278.60400  -0.281 0.778907
## PrivateYes   -579.48119  163.93251  -3.535 0.000443 ***
## Accept          1.67565    0.04637  36.139  < 2e-16 ***
## Enroll         -0.80120    0.13780  -5.814 1.05e-08 ***
## Top10perc      35.96435    3.97772   9.041  < 2e-16 ***
## Outstate       -0.09702    0.02187  -4.437 1.11e-05 ***
## Room.Board      0.20086    0.05851   3.433 0.000644 ***
## PhD           -11.80305    3.71160  -3.180 0.001558 **
## Expend          0.07806    0.01326   5.889 6.89e-09 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1071 on 534 degrees of freedom
## Multiple R-squared:  0.9287,Adjusted R-squared:  0.9276
## F-statistic:   869 on 8 and 534 DF,  p-value: < 2.2e-16
```

We determine the residual sum of squares on the test the following value as follows:

```
y_pred = predict(lin_reg02, newdata = test)
RSS <- sum((test$Apps - y_pred)^2)
formatC(RSS, format = "e", digits = 4)

## [1] "2.4527e+08"
```

Thus, the RSS on the test set is given by 2.4526595×10^8.

For the Lasso and ridge regression we have to prepare our data as matrices

```
library(glmnet)

## Loading required package:  Matrix

## Loaded glmnet 4.1-1

X_train <- model.matrix(~. - 1, data = subset(train, select = -Apps))
y_train <- subset(train, select = Apps)
X_test <- model.matrix(~. - 1, data = subset(test, select = -Apps))
y_test <- subset(test, select = Apps)
```

In order to fit a regression model, we need to find an appropriate value for the regularization parameter λ. By means of cross-validation, we determine the regularization parameter for Ridge regression as follows:

```
mod_cv_ridge = cv.glmnet(x = X_train, y = y_train$Apps,
    alpha = 0, nfolds = 5)

mod_cv_ridge$lambda.min

## [1] 376.2027

y_pred = predict(mod_cv_ridge, newx = X_test)
RSS <- sum((y_test$Apps - y_pred)^2)
formatC(RSS, format = "e", digits = 4)

## [1] "2.5288e+08"
```

We thus find a value of the regularization parameter given by 376.2027063 and an RSS on the test set of 2.5288354×10^8.

Similarly, when changing the α to 1, we fit the Lasso model.

```
mod_cv_lasso = cv.glmnet(x = X_train, y = y_train$Apps,
    alpha = 1, nfolds = 5)

coef(mod_cv_lasso)

## 19 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) -424.69517672
## PrivateNo        .
## PrivateYes       .
## Accept          1.38008535
## Enroll           .
## Top10perc      14.67561609
## Top25perc        .
## F.Undergrad      .
## P.Undergrad      .
## Outstate         .
## Room.Board       .
## Books            .
## Personal         .
## PhD              .
## Terminal         .
## S.F.Ratio        .
## perc.alumni      .
## Expend          0.02571111
## Grad.Rate        .

mod_cv_lasso$lambda.min

## [1] 2.203427
```

with a regularization parameter value of 2.2034271 and the following non-zero predictor variables

- **Accept**
- **Top10perc**

- **Expend**

```
y_pred = predict(mod_cv_lasso, newx = X_test)
RSS <- sum((y_test$Apps - y_pred)^2)
formatC(RSS, format = "e", digits = 4)

## [1] "2.6618e+08"
```

The RSS is given by 2.6618424×10^8.

Solution 10.2

Read in the data set and split it into training and test set

```
library(caTools)
library(MASS)
boston <- Boston
set.seed(42)
sample <- sample.split(boston$crim, SplitRatio = 0.7)
train <- boston[sample == TRUE, ]
test <- boston[sample == FALSE, ]
```

We want to determine the best subset of predictor variables:

```
library(leaps)
best_subset <- regsubsets(crim ~ . - 1, data = train, nbest = 1,
    nvmax = NULL, force.in = NULL, force.out = NULL, method = "exhaustive
summary_best_subset <- summary(best_subset)
num_predictors <- which.min(summary_best_subset$bic)
num_predictors

## [1] 2
```

The best subset of predictor variables as determined on the basis of the Bayesian Information Criterion consists of 2 predictor variables.

```
predictors = names(which(summary_best_subset$which[2, ]))
predictors

## [1] "(Intercept)" "rad"          "medv"
```

```
lin_reg02 <- lm(crim ~ rad + medv, data = train)
summary(lin_reg02)

##
## Call:
## lm(formula = crim ~ rad + medv, data = train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -8.951 -1.805 -0.670  0.747 75.129
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.48521    1.37227   1.811    0.071 .
## rad          0.55246    0.04908  11.256  < 2e-16 ***
## medv        -0.18240    0.04600  -3.965  8.9e-05 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.424 on 351 degrees of freedom
## Multiple R-squared:  0.3777,Adjusted R-squared:  0.3741
## F-statistic: 106.5 on 2 and 351 DF,  p-value: < 2.2e-16
```

We determine the residual sum of squares (RSS) on the test set as follows:

```
y_pred = predict(lin_reg02, newdata = test)
RSS <- sum((test$crim - y_pred)^2)
formatC(RSS, format = "e", digits = 4)

## [1] "2.4366e+03"
```

The value of the RSS on the test set is given by of 2436.5753415.

For the Lasso and ridge regression we have to prepare our data set as matrices

```
library(glmnet)
X_train <- model.matrix(~. - 1, data = subset(train, select = -crim))
y_train <- subset(train, select = crim)
X_test <- model.matrix(~. - 1, data = subset(test, select = -crim))
y_test <- subset(test, select = crim)
```

We then fit a Ridge regression model as follows:

```
mod_cv_ridge = cv.glmnet(x = X_train, y = y_train$crim,
    alpha = 0, nfolds = 5)

coef(mod_cv_ridge)

## 14 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  2.062699101
## zn          -0.002844941
## indus        0.023684391
## chas        -0.123228040
## nox          1.484747033
## rm          -0.096771641
## age          0.004976946
## dis         -0.071659798
## rad          0.031361350
## tax          0.001483516
## ptratio      0.056106369
## black       -0.001775967
## lstat        0.026835783
## medv        -0.018303052

mod_cv_ridge$lambda.min

## [1] 0.554208

y_pred = predict(mod_cv_ridge, newx = X_test)
RSS <- sum((y_test$crim - y_pred)^2)
formatC(RSS, format = "e", digits = 4)

## [1] "4.7604e+03"
```

The value of the regularization parameter λ is given by 0.554208. The RSS on the test set is 4760.4039792.

By changing the value of α to 1, we fit a Lasso model to the data:

```
mod_cv_lasso = cv.glmnet(x = X_train, y = y_train$crim,
    alpha = 1, nfolds = 5)
coef(mod_cv_lasso)

## 14 x 1 sparse Matrix of class "dgCMatrix"
##                     1
## (Intercept) 1.4969043
## zn          .
## indus       .
## chas        .
## nox         .
## rm          .
## age         .
## dis         .
## rad         0.2346782
## tax         .
## ptratio     .
## black       .
## lstat       .
## medv        .

mod_cv_lasso$lambda.min

## [1] 0.03027238
```

The best λ value is found to be 0.0302724 and the following predictor variables are non-zero:

- **rad**

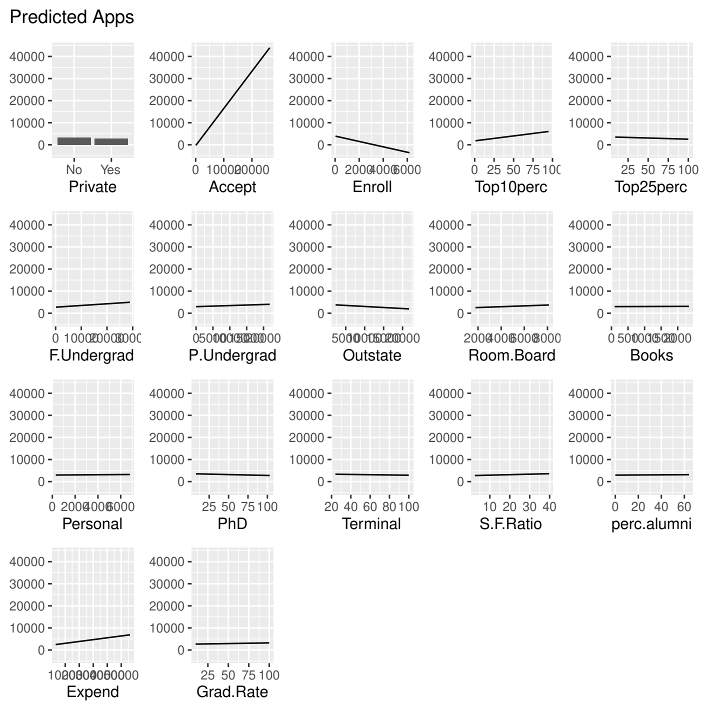
```
y_pred <- predict(mod_cv_lasso, newx = X_test)
RSS <- sum((y_test$crim - y_pred)^2)
formatC(RSS, format = "e", digits = 4)

## [1] "3.8754e+03"
```

The RSS on the test set is given by 3875.4399355 .

Solution 10.3

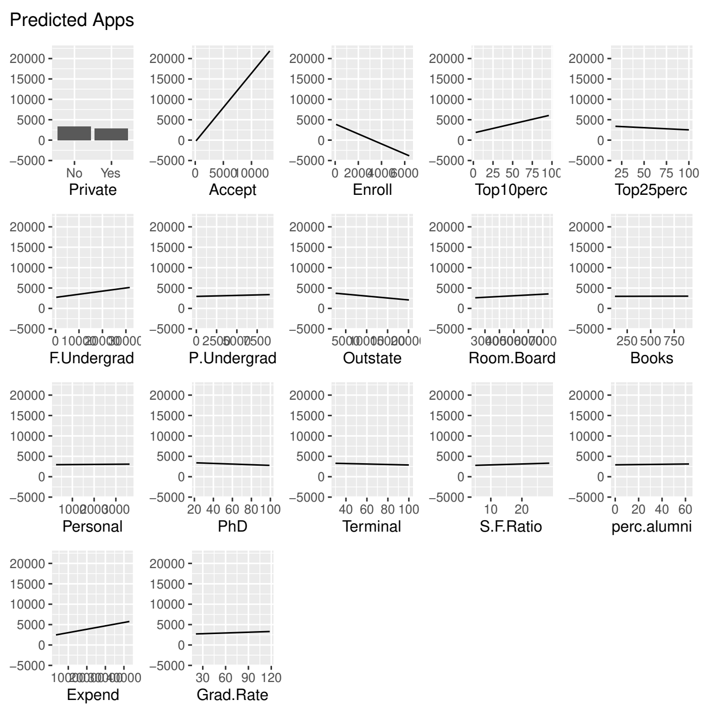
```r
library(iml)
library(caTools)
library(ISLR)
college <- ISLR::College
set.seed(42)
college$Private <- as.factor(college$Private)
sample <- sample.split(college$Apps, SplitRatio = 0.7)
train <- college[sample == TRUE, ]
test <- college[sample == FALSE, ]
lin_reg01 <- lm(Apps ~ ., data = train)
mod <- Predictor$new(lin_reg01, data = train)
mod_test <- Predictor$new(lin_reg01, data = test)
```

```
eff <- FeatureEffects$new(mod, method = "pdp")
eff$plot()
```



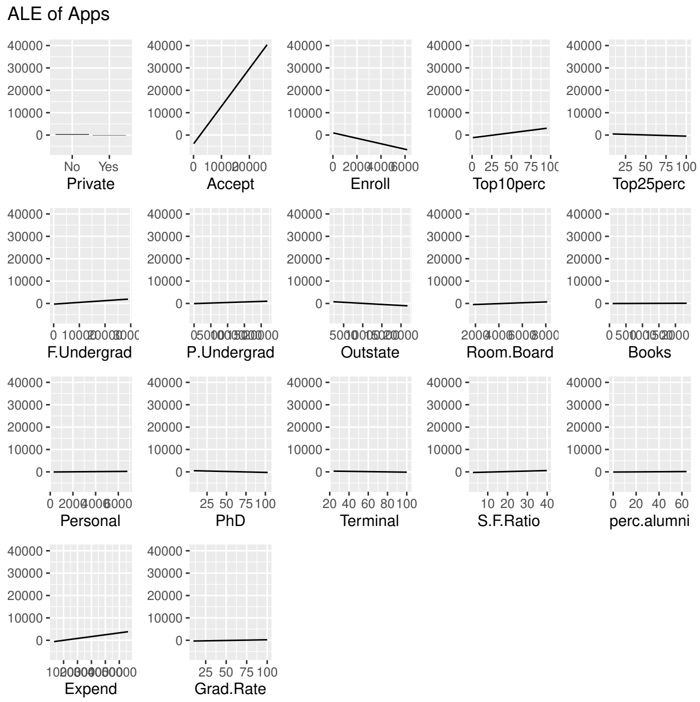
(a) Partial Dependence Plot on the training set.

```
eff_test <- FeatureEffects$new(mod_test, method = "pdp")
eff_test$plot()
```



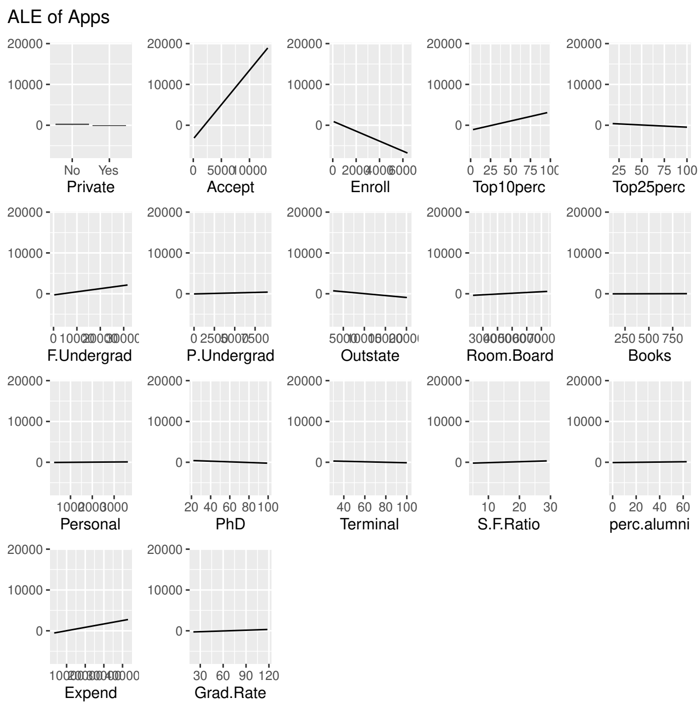
(b) Partial Dependence Plot on the test set.

```
eff <- FeatureEffects$new(mod, method = "ale")
eff$plot()
```



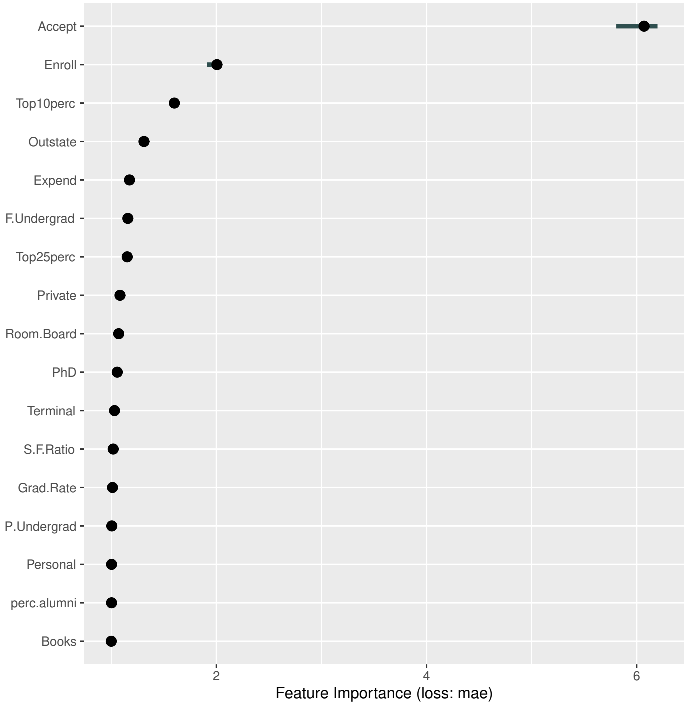
(a) Accumulated Local Effects on the training set.

```
eff_test <- FeatureEffects$new(mod_test, method = "ale")
eff_test$plot()
```



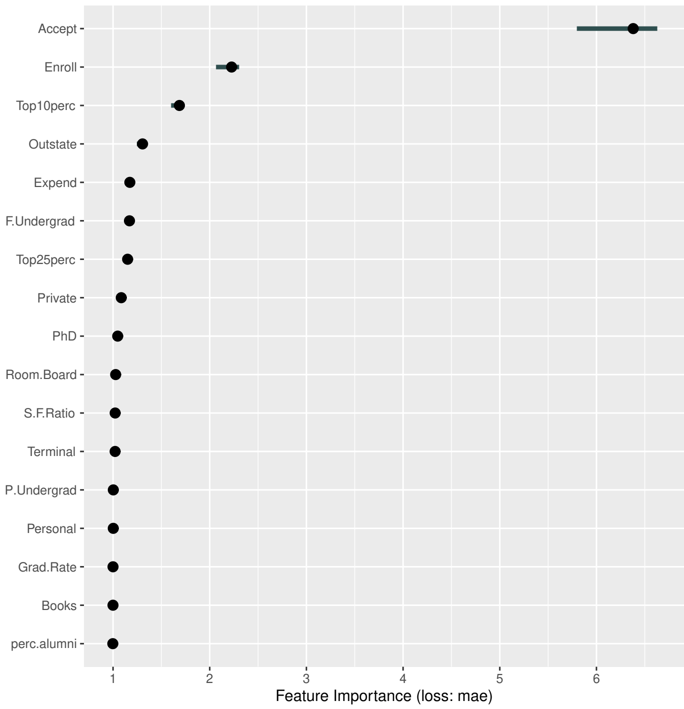
(b) Accumulated Local Effects on the test set.

```r
imp <- FeatureImp$new(mod, loss = "mae")
plot(imp)
```



(a) Permutation Feature Importance on the training set.

```r
mod_test <- Predictor$new(lin_reg01, data = test)
imp_test <- FeatureImp$new(mod_test, loss = "mae")
plot(imp_test)
```



(b) Permutation Feature Importance on the test set.

As we can see, most agnostic metrics agree on the importance of `Accept`, `Enroll`, `Top10perc`, and `Outstate`.

Solution 10.4 Let us use the train and test sets we created in exercise 1 to construct a cross-validated linear regression model using `caret`

[1]:
```
library(caret)

# Specify cross-validation as training method with 10 folds
train_control <- trainControl(method='cv', number=10)
# Other models can be found in https://rdrr.io/cran/caret/man/models.html
model <- train(Apps ~ ., data=train, method='lm', trControl=train_control)

# print model parameters
model
# and get the classic coefficients and metrics summary
summary(model)

# get the feature importance
vimp <- varImp(model)
vimp
plot(vimp)
```

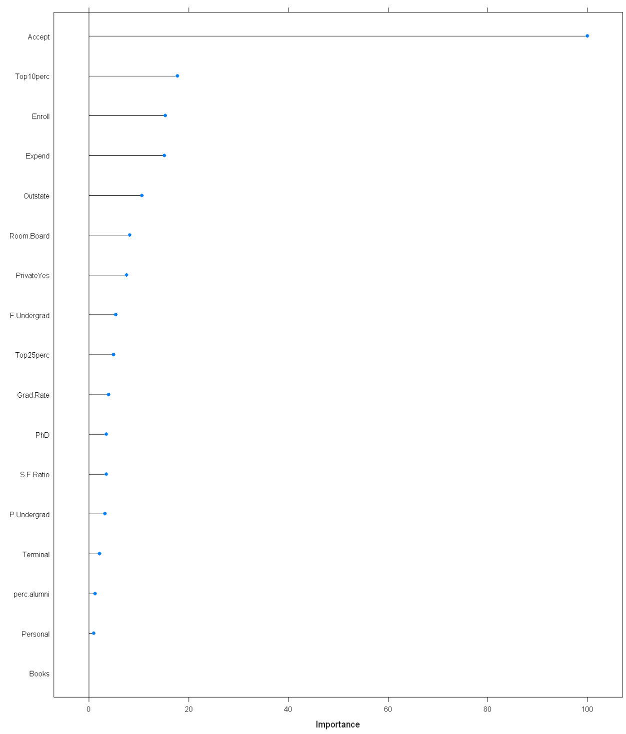


Figure 4: `Caret` feature importance on the linear regression model.

New data can be predicted as usual:

[1]:
```
test$predicted <- predict(model, newdata = test)
cor(log(test$Apps), test$predicted)

SSR = sum((test$Apps - test$predicted)**2)
formatC(SSR, format = "e", digits = 4)
```

Try out some other models from **Caret** and get an intuition about the opportunities this package offers.