Covariate Creation

Christopher Lovell

Sunday, October 26, 2014

Introduction

There are two levels of covariate creation: transforming raw data in to a covariate, and transforming tidy covariates in to alternative forms.

Raw Data -> Covariates

The balancing act is summarisation vs. information loss. More knowledge of a system you have the better the job you will do. Whn in doubt, err on the side of more features.

Feature selection can be automated, but should be done with caution.

Tidy Covariates -> new covariates

Features you've already created on the dataset, transformed to make them more useful.

The original building of the covarites should only be performed on the training set (otherwise you're at risk of overfitting).

Example

First load the libraries and data, then split the Wage dataset in to training and testing sets.

```
library(ISLR)
library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
data(Wage)

inTrain <- createDataPartition(y=Wage$wage,p=0.7,list=FALSE)

training <- Wage[inTrain,]
testing <- Wage[-inTrain,]
```

If we take a look at the jobclass variable, it is split in to Industrial and Information classes:

```
table(training$jobclass)
```

```
##
##  1. Industrial 2. Information
##           1081           1021
```

These are factor, or qualitative, variables, and a machine learning algorithm may find them difficult to interpret in their extended current form, therefore we may wish to convert them to quantitative, or indicator, variables:

```
dummies <- dummyVars(wage ~ jobclass,data=training)
head(predict(dummies,newdata=training))
```

```
##        jobclass.1. Industrial jobclass.2. Information
## 231655                       1                       0
## 86582                        0                       1
## 11443                        0                       1
## 11141                        0                       1
## 448410                       1                       0
## 229379                       1                       0
```

Removing Zero Covariates

If a covariate is almost always true, for example 'Does the email contain characters?', you can remove it using the following:

```
nsv <- nearZeroVar(training,saveMetric=TRUE)
nsv
```

```
##            freqRatio percentUnique zeroVar   nzv
## year        1.008547    0.33301618   FALSE FALSE
## age         1.222222    2.85442436   FALSE FALSE
## sex         0.000000    0.04757374    TRUE  TRUE
## maritl      3.130152    0.23786870   FALSE FALSE
## race        8.564356    0.19029496   FALSE FALSE
## education   1.477612    0.23786870   FALSE FALSE
## region      0.000000    0.04757374    TRUE  TRUE
## jobclass    1.058766    0.09514748   FALSE FALSE
## health      2.562712    0.09514748   FALSE FALSE
## health_ins  2.238829    0.09514748   FALSE FALSE
## logwage     1.142857   19.07706946   FALSE FALSE
## wage        1.142857   19.07706946   FALSE FALSE
```

The sex variable is all Male, therefore it has no variability and can be thrown out. The same applies to the Region variable.

Spline basis

Basis functions allow non-linear regression fitting. they are contained in the *splines* package.

```
library(splines)
bsBasis <- bs(training$age,df=3)
tail(bsBasis)
```

```
##                 1         2          3
## [2097,] 0.3063341 0.4241549 0.19576382
## [2098,] 0.4430868 0.2436978 0.04467792
```

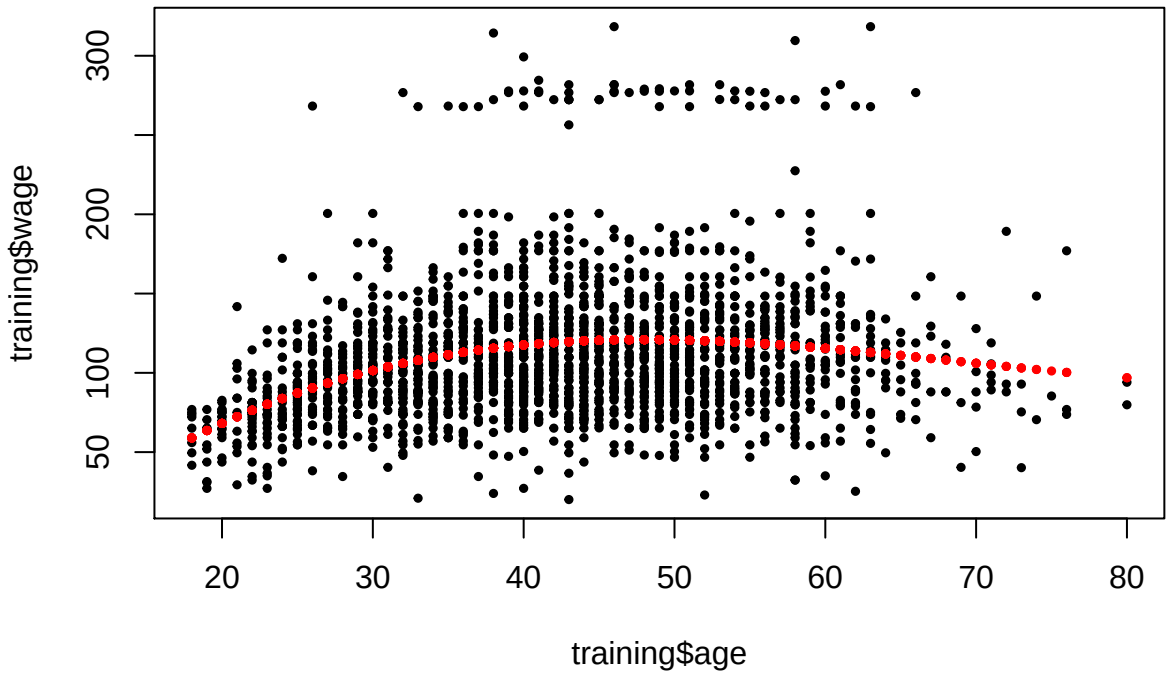
```
## [2099,] 0.2436978 0.4430868 0.26853748
## [2100,] 0.4060287 0.1184250 0.01151354
## [2101,] 0.3625256 0.3866940 0.13749119
## [2102,] 0.4241549 0.3063341 0.07374710
```

This function allows fitting of a third degree polynomial to the age data in the training set. It produces a dataframe with 3 columns, containing the original age data, age squared and age cubed.

Fitting curves with splines

You can then pass this *bsBasis* object to a linear fitting function against another variable in your training set, in this case wage. The plot shows the relationship between age and wage, along with a third order polynomial fit for the relationship.

```
lm1 <- lm(wage ~ bsBasis,data=training)
plot(training$age,training$wage,pch=19,cex=0.5)
points(training$age,predict(lm1,newdata=training),col="red",pch=19,cex=0.5)
```



Splines on the test set

Must create coivariates on the test set using the exact same procedure as the training.

```r
head(predict(bsBasis,age=testing$age))
```

```
##               1          2           3
## [1,] 0.00000000 0.00000000 0.000000000
## [2,] 0.23685006 0.02537679 0.000906314
## [3,] 0.36252559 0.38669397 0.137491189
## [4,] 0.44308684 0.24369776 0.044677923
## [5,] 0.01793746 0.20448709 0.777050955
## [6,] 0.43081384 0.29109043 0.065560908
```