DATA 621 - HW3

Andrew Bowen, Glen Davis, Shoshana Farber, Joshua Forster, Charles Ugiagbe

2023-10-23

Homework 3 - Logistic Regression

Overview

In this homework assignment, you will explore, analyze and model a data set containing information on crime for various neighborhoods of a major city. Each record has a response variable indicating whether or not the crime rate is above the median crime rate (1) or not (0).

Your objective is to build a binary logistic regression model on the training data set to predict whether the neighborhood will be at risk for high crime levels. You will provide classifications and probabilities for the evaluation data set using your binary logistic regression model. You can only use the variables given to you (or, variables that you derive from the variables provided).

Below is a short description of the variables of interest in the data set:

Column	Description
`zn`	proportion of residential land zoned for large lots (over 25000 square feet) (*predictor variable*)
`indus`	proportion of non-retail business acres per suburb (*predictor variable*)
`chas`	a dummy var. for whether the suburb borders the Charles River (1) or not (0) (*predictor variable*)
`nox`	nitrogen oxides concentration (parts per 10 million) (*predictor variable*)
`rm`	average number of rooms per dwelling (*predictor variable*)
`age`	proportion of owner-occupied units built prior to 1940 (*predictor variable*)
`dis`	weighted mean of distances to five Boston employment centers (*predictor variable*)
`rad`	index of accessibility to radial highways (*predictor variable*)
`tax`	full-value property-tax rate per $10,000 (*predictor variable*)
`ptratio`	pupil-teacher ratio by town (*predictor variable*)
`lstat`	lower status of the population (percent) (*predictor variable*)
`medv`	median value of owner-occupied homes in $1000s (*predictor variable*)
`target`	**whether the crime rate is above the median crime rate (1) or not (0) (*response variable*)**

Data Loading

Let's load in the training dataset.

```
train_df <- read.csv('https://raw.githubusercontent.com/ShanaFarber/businessAnalyticsDataMiningDATA621/m

head(train_df)  # preview data
```

```
##   zn indus chas   nox    rm   age    dis rad tax ptratio lstat medv target
## 1  0 19.58    0 0.605 7.929  96.2 2.0459   5 403    14.7  3.70 50.0      1
## 2  0 19.58    1 0.871 5.403 100.0 1.3216   5 403    14.7 26.82 13.4      1
## 3  0 18.10    0 0.740 6.485 100.0 1.9784  24 666    20.2 18.85 15.4      1
## 4 30  4.93    0 0.428 6.393   7.8 7.0355   6 300    16.6  5.19 23.7      0
## 5  0  2.46    0 0.488 7.155  92.2 2.7006   3 193    17.8  4.82 37.9      0
## 6  0  8.56    0 0.520 6.781  71.3 2.8561   5 384    20.9  7.67 26.5      0
```

Data Exploration

```
dim(train_df)
```

```
## [1] 466  13
```

The dataset consists of 466 observations of 13 variables. There are 12 predictor variables and one response variable (`target`).

All of the columns in the dataset are numeric, but the predictor variable `chas` is a dummy variable, as is the response variable `target`. We recode them as factors.

```
train_df <- train_df |>
    mutate(chas = as.factor(chas), target = as.factor(target))
```

Let's take a look at the summary statistics for the variables in the dataset.

```
summary(train_df)
```

```
##        zn              indus        chas         nox               rm
##  Min.   :  0.00   Min.   : 0.460   0:433   Min.   :0.3890   Min.   :3.863
##  1st Qu.:  0.00   1st Qu.: 5.145   1: 33   1st Qu.:0.4480   1st Qu.:5.887
##  Median :  0.00   Median : 9.690           Median :0.5380   Median :6.210
##  Mean   : 11.58   Mean   :11.105           Mean   :0.5543   Mean   :6.291
##  3rd Qu.: 16.25   3rd Qu.:18.100           3rd Qu.:0.6240   3rd Qu.:6.630
##  Max.   :100.00   Max.   :27.740           Max.   :0.8710   Max.   :8.780
##       age              dis              rad              tax
##  Min.   :  2.90   Min.   : 1.130   Min.   : 1.00   Min.   :187.0
##  1st Qu.: 43.88   1st Qu.: 2.101   1st Qu.: 4.00   1st Qu.:281.0
##  Median : 77.15   Median : 3.191   Median : 5.00   Median :334.5
##  Mean   : 68.37   Mean   : 3.796   Mean   : 9.53   Mean   :409.5
##  3rd Qu.: 94.10   3rd Qu.: 5.215   3rd Qu.:24.00   3rd Qu.:666.0
##  Max.   :100.00   Max.   :12.127   Max.   :24.00   Max.   :711.0
##     ptratio          lstat            medv       target
```

```
##  Min.   :12.6   Min.   : 1.730   Min.   : 5.00   0:237
##  1st Qu.:16.9   1st Qu.: 7.043   1st Qu.:17.02   1:229
##  Median :18.9   Median :11.350   Median :21.20
##  Mean   :18.4   Mean   :12.631   Mean   :22.59
##  3rd Qu.:20.2   3rd Qu.:16.930   3rd Qu.:25.00
##  Max.   :22.0   Max.   :37.970   Max.   :50.00
```

We can see the mean, median, and interquartile ranges for each of the numeric variables in the dataset.

There do not appear to be any NA values in the dataset. Let's validate this.

```
sum(is.na(train_df))
```

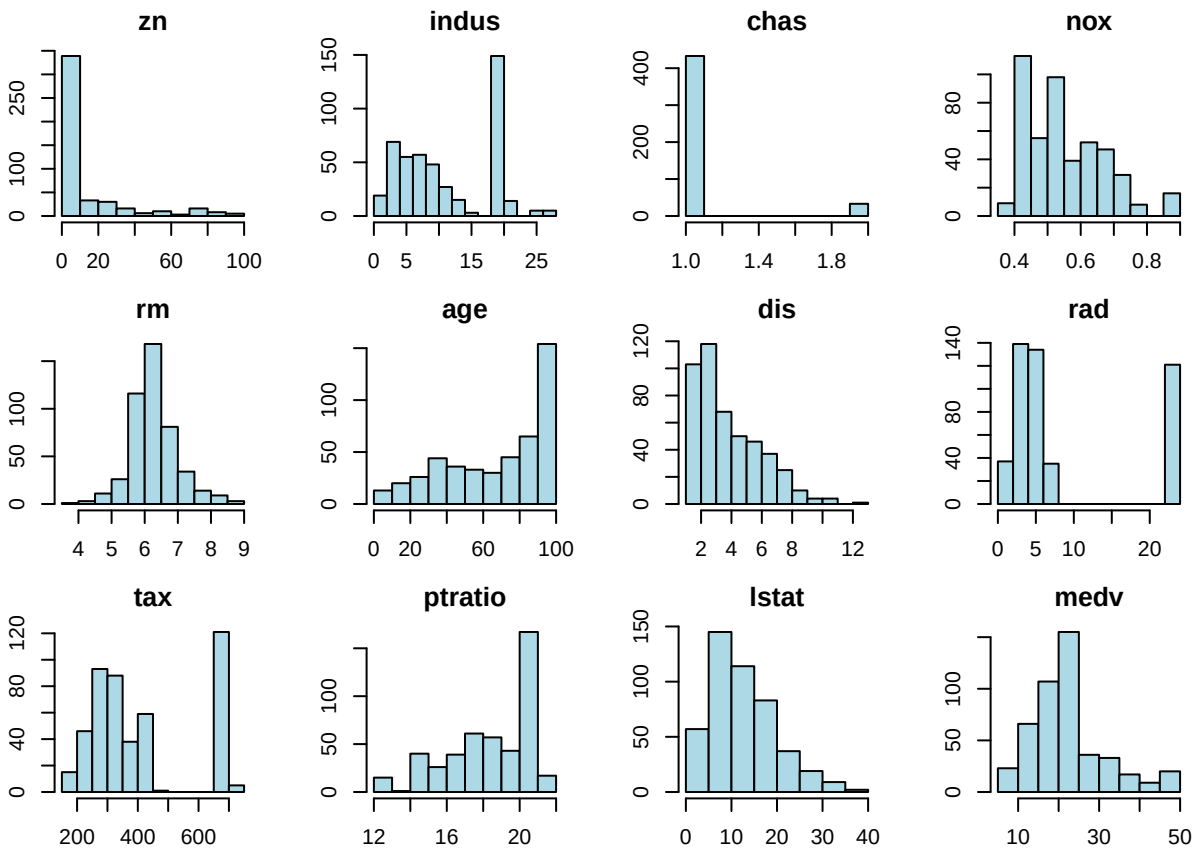
```
## [1] 0
```

There are no null values in the training dataset.

Let's take a look at the distributions for the predictor variables.

```
par(mfrow=c(3,4))
par(mai=c(.3,.3,.3,.3))

variables <- names(train_df)
factors <- c("chas", "target")
for (i in 1:(length(variables)-1)) {
    if (variables[i] %in% factors){
        hist(as.numeric(train_df[[variables[i]]]), main = variables[i],
             col = "lightblue")
    }else{
        hist(train_df[[variables[i]]], main = variables[i], col = "lightblue")
    }
}
```



`rm` appears to be normally distributed. `zn`, `dis`, and `lstat` appear to be skewed to the right. All other columns seem to be multimodal.

The distribution for `chas` appears degenerate at first glance. It looks like a near-zero variance predictor, which we can confirm using the `nearZeroVar` function from the `caret` package.

```
nzv <- nearZeroVar(train_df |> select(-target), saveMetrics = TRUE)
knitr::kable(nzv)
```

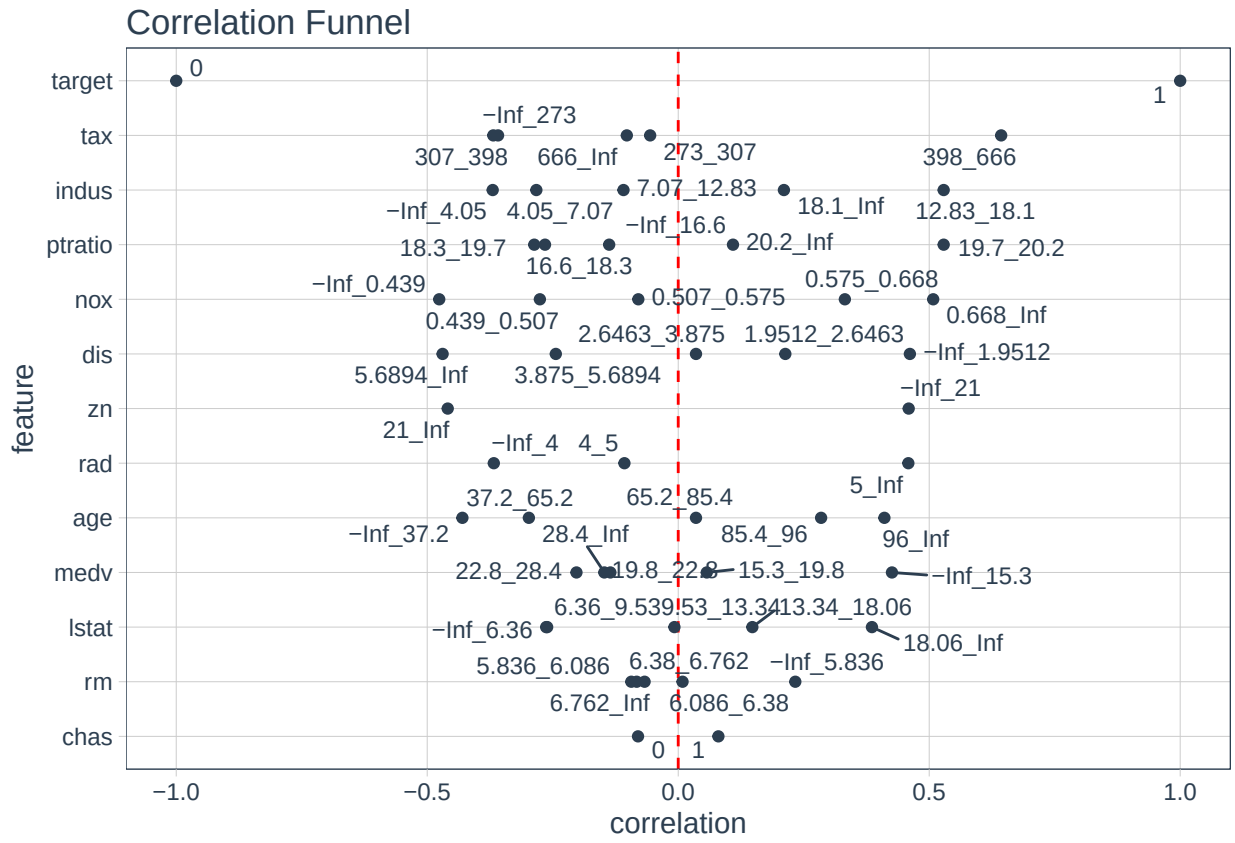
	freqRatio	percentUnique	zeroVar	nzv
zn	16.142857	5.5793991	FALSE	FALSE
indus	4.321429	15.6652361	FALSE	FALSE
chas	13.121212	0.4291845	FALSE	FALSE
nox	1.176471	16.9527897	FALSE	FALSE
rm	1.000000	89.9141631	FALSE	FALSE
age	10.500000	71.4592275	FALSE	FALSE
dis	1.000000	81.5450644	FALSE	FALSE
rad	1.110092	1.9313305	FALSE	FALSE
tax	3.457143	13.5193133	FALSE	FALSE
ptratio	4.000000	9.8712446	FALSE	FALSE
lstat	1.000000	90.9871245	FALSE	FALSE
medv	2.142857	46.7811159	FALSE	FALSE

The percentage of unique values, `percentUnique`, in the sample for this predictor is less than the typical threshold of 10 percent, but there is a second criterion to consider: the `freqRatio`. This measures the

frequency of the most common value (0 in this case) to the frequency of the second most common value (1 in this case). The `freqRatio` value for this predictor is less than the typical threshold of 19 (i.e. 95 occurrences of the most frequent value for every 5 occurrences of the second most frequent values). So it is not considered a near-zero variance predictor. Neither are any of the other predictors.

We produce a correlation funnel to visualize the relationships between our predictors and our response.

```
train_df_binarized <- train_df |>
    binarize(n_bins = 5, thresh_infreq = 0.01, name_infreq = "OTHER",
             one_hot = TRUE)
train_df_corr <- train_df_binarized |>
    correlate(target__1)
train_df_corr |>
    plot_correlation_funnel()
```



The correlation funnel plots the most important features towards the top. The variable `chas` is the least correlated to `target` by the Pearson Correlation coefficient. The correct coefficient to use to understand the strength of the relationship between two categorical variables is actually the ϕ coefficient. If one of the categorical variables had more than two categories, we would need to calculate ϕ using the formula for Cramer's V (also called Cramer's ϕ). However, in the special case that both categorical variables are binary, the value of the Cramer's V coefficient will actually be equal to the value of the Pearson Correlation coefficient. So either formula actually results in the same value for ϕ. We prove this below.

```
cramersv <- round(cramersv(train_df |> select(all_of(factors))), 5)
pearson <- round(cor(as.numeric(train_df$chas), as.numeric(train_df$target), method = "pearson"), 5)
(cramersv == pearson)
```

```
## [1] TRUE
```

The value for ϕ is 0.08004 regardless of the formula used to calculate it, and this very low value indicates very little correlation between `chas` and `target`.