Logistic Regression

Arnab Maity - Modified by Justin Post

library(MASS)  
library(klaR)  
library(tidyverse)  
library(caret)  
library(rsample)  
library(ISLR2)  
library(knitr)  
library(AppliedPredictiveModeling)  
library(kableExtra)  
library(nnet)  
library(glmnet)

We’ve now discussed the main ideas associated with the predictive modeling tasks:

* Regression task:
  + numeric response variable
  + MSE or RMSE is the commonly used metric
* Classification task:
  + categorical response variable
  + accuracy, confusion matrix-based metrics, and log-loss are common metrics

Now we’ll talk about other models that are commonly used for the classification task!

# Logistic Regression

We talked about two major ways to create classification models:

* Models that directly try to model the conditional class probabilities,
* Generative models that model ’s distribution and use Bayes’ theorem to obtain estimated conditional class probabilities

Logistic regression takes the first approach.

* LR models the model the probabilities associated with the distribution of as functions of the data vector without actually specifying any distribution of .

Often, logistic regression models are used for their abilities to help us understand the role of the predictors in explaining the outcome (our discrimination task)

## Why not an MLR Model?

We might first ask, why can’t we use our usual MLR model for the classification setting?

Consider just a binary response

* What is the mean of the response?

   
 

Suppose you have a predictor variable as well, call it

* Given two values of we could model separate proportions
* For a continuous , we could consider a SLR model
* What’s wrong with this? Consider data about [heart disease](https://www4.stat.ncsu.edu/online/datasets/heart.csv)

library(tidyverse)  
heart\_data <- read\_csv("https://www4.stat.ncsu.edu/online/datasets/heart.csv") |>  
 filter(RestingBP > 0) |> #remove one value  
 mutate(HeartDiseaseFac = factor(HeartDisease))  
heart\_data |>   
 dplyr::select(HeartDiseaseFac, everything())

# A tibble: 917 × 13  
 HeartDiseaseFac Age Sex ChestPainType RestingBP Cholesterol FastingBS  
 <fct> <dbl> <chr> <chr> <dbl> <dbl> <dbl>  
 1 0 40 M ATA 140 289 0  
 2 1 49 F NAP 160 180 0  
 3 0 37 M ATA 130 283 0  
 4 1 48 F ASY 138 214 0  
 5 0 54 M NAP 150 195 0  
 6 0 39 M NAP 120 339 0  
 7 0 45 F ATA 130 237 0  
 8 0 54 M ATA 110 208 0  
 9 1 37 M ASY 140 207 0  
10 0 48 F ATA 120 284 0  
# ℹ 907 more rows  
# ℹ 6 more variables: RestingECG <chr>, MaxHR <dbl>, ExerciseAngina <chr>,  
# Oldpeak <dbl>, ST\_Slope <chr>, HeartDisease <dbl>

#Cholesterol has many values set to 0 so we ignore that

* Heart Disease is our response variable. We can look at the mean of the binary variable to estimate

heart\_data |>  
 group\_by(HeartDiseaseFac) |>   
 summarize(prop = n()/nrow(heart\_data))

# A tibble: 2 × 2  
 HeartDiseaseFac prop  
 <fct> <dbl>  
1 0 0.447  
2 1 0.553

* Of course, we may think that these probabilities may differ based on the values of a predictor variable

heart\_data |>  
 group\_by(HeartDiseaseFac) |>  
 summarize(mean\_Age = mean(Age),  
 mean\_RestingBP = mean(RestingBP))

# A tibble: 2 × 3  
 HeartDiseaseFac mean\_Age mean\_RestingBP  
 <fct> <dbl> <dbl>  
1 0 50.6 130.  
2 1 55.9 134.

We can try to fit our SLR model with Age as our predictor.

ggplot(heart\_data, aes(x = Age, y = HeartDisease, color = RestingBP)) +  
 geom\_point() +  
 geom\_smooth(method = "lm")

|  |
| --- |
|  |

That doesn’t quite look right… why? The points all lie on top of one another. Let’s fix that by jittering them.

ggplot(heart\_data, aes(x = Age, y = HeartDisease, color = RestingBP)) +  
 geom\_jitter() +  
 geom\_smooth(method = "lm")

|  |
| --- |
|  |

That’s better but still not clear. We really are modeling probabilities (proportions) at each value Age can take on. We can visualize this with small Age group means.

* Obtain proportion with heart disease for different Age groups

Age\_x <- seq(from = min(heart\_data$Age), to = max(heart\_data$Age), length = 20)  
heart\_data\_grouped <- heart\_data |>  
 mutate(Age\_groups = cut(Age, breaks = Age\_x)) |>  
 group\_by(Age\_groups) |>  
 summarize(HeartDisease\_mean = mean(HeartDisease), counts = n())  
heart\_data\_grouped

# A tibble: 20 × 3  
 Age\_groups HeartDisease\_mean counts  
 <fct> <dbl> <int>  
 1 (28,30.6] 0 4  
 2 (30.6,33.2] 0.444 9  
 3 (33.2,35.7] 0.333 18  
 4 (35.7,38.3] 0.424 33  
 5 (38.3,40.9] 0.286 28  
 6 (40.9,43.5] 0.303 66  
 7 (43.5,46.1] 0.361 61  
 8 (46.1,48.6] 0.52 50  
 9 (48.6,51.2] 0.494 81  
10 (51.2,53.8] 0.493 69  
11 (53.8,56.4] 0.550 129  
12 (56.4,58.9] 0.675 80  
13 (58.9,61.5] 0.745 98  
14 (61.5,64.1] 0.724 87  
15 (64.1,66.7] 0.647 34  
16 (66.7,69.3] 0.737 38  
17 (69.3,71.8] 0.667 12  
18 (71.8,74.4] 0.75 12  
19 (74.4,77] 0.714 7  
20 <NA> 0 1

* Now plot that data as well!

ggplot(data = heart\_data, aes(x = Age, y = HeartDisease)) +  
 geom\_jitter(aes(color = RestingBP)) +  
 geom\_point(data = heart\_data\_grouped, aes(x = Age\_x, y = HeartDisease\_mean, size = counts)) +  
 geom\_smooth(method = "lm", color = "Green")

`geom\_smooth()` using formula = 'y ~ x'

|  |
| --- |
|  |

Here SLR isn’t actually terrible! However, theoretically we have some issues.

* Response = success/failure (0/1)
  + We are modeling the average number of successes for a given . This is a probability!
    - predictions should never go below 0
    - predictions should never go above 1

The SLR model doesn’t require this! We just got lucky. Consider a much wider set of Age values.

ggplot(data = heart\_data, aes(x = Age, y = HeartDisease)) +  
 geom\_jitter(aes(color = RestingBP)) +  
 geom\_point(data = heart\_data\_grouped, aes(x = Age\_x, y = HeartDisease\_mean, size = counts)) +  
 geom\_smooth(method = "lm", color = "Green") +  
 xlim(0, 100)

`geom\_smooth()` using formula = 'y ~ x'

|  |
| --- |
|  |

The predicted probability (the value of the line) will go below 0 and above 1 sometimes!

Ideally, we should consider a model that requires this. One option is a logistic regression model.

## Logistic Regression Model

* The basic Logistic Regression model uses a *logistic function* to model the success probability:

This function never goes below 0 or above 1!

|  |
| --- |
|  |

This is actually a non-linear model! It is much harder to fit the model…

* The logistic regression model doesn’t have a closed form solution (maximum likelihood often used to fit parameters)

### Interpreting the Model

How can we relate this to our usual linear model?

* Back-solving shows the *logit* or *log-odds* of success is linear in the parameters

- Coefficient interpretation changes greatly from linear regression model!

- $\frac{P(success|X)}{1-P(success|X)}$ is call the odds of success  
- $\beta\_1$ represents a change in the log-odds of success

Let’s plot the logistic regression model that uses Age to predict HeartDisease:

|  |
| --- |
|  |

This is a sigmoid function that looks linear close up! Let’s zoom out:

|  |
| --- |
|  |

Nice!

### Including More Than One Predictor

In the MLR model we could easily add other numeric predictors, interactions, quadratics, and categorical predictors via the use of indicator variables. We can do this here too!

* Adding a dummy variable corresponding to a binary predictor just changes the ‘intercept’
  + Call Age
  + Let’s add a predictor for Sex ( if ‘Male’ and if “Female”)

|  |
| --- |
|  |

We can see that we don’t have parallel lines in this case. Let’s investigate the model fit a bit more.

* Similar to the lm() function to fit MLR models, glm() with family = "binomial" allows us to fit logistic regression models.

log\_reg\_fit <- glm(HeartDiseaseFac ~ Age + Sex,   
 data = heart\_data, family = "binomial")  
summary(log\_reg\_fit)

Call:  
glm(formula = HeartDiseaseFac ~ Age + Sex, family = "binomial",   
 data = heart\_data)  
  
Deviance Residuals:   
 Min 1Q Median 3Q Max   
-2.0594 -1.0661 0.6460 0.9366 2.2458   
  
Coefficients:  
 Estimate Std. Error z value Pr(>|z|)   
(Intercept) -4.631848 0.481062 -9.628 < 2e-16 \*\*\*  
Age 0.066476 0.008163 8.144 3.84e-16 \*\*\*  
SexM 1.638859 0.189346 8.655 < 2e-16 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for binomial family taken to be 1)  
  
 Null deviance: 1261.0 on 916 degrees of freedom  
Residual deviance: 1100.7 on 914 degrees of freedom  
AIC: 1106.7  
  
Number of Fisher Scoring iterations: 4

   
   
   
   
 

* If we fit an interaction term between Age and our Sex dummy variable, we essentially fit two separate logistic regression models

log\_reg\_fit\_int <- glm(HeartDiseaseFac ~ Age + Sex + Age\*Sex,   
 data = heart\_data, family = "binomial")  
summary(log\_reg\_fit\_int)

Call:  
glm(formula = HeartDiseaseFac ~ Age + Sex + Age \* Sex, family = "binomial",   
 data = heart\_data)  
  
Deviance Residuals:   
 Min 1Q Median 3Q Max   
-2.0760 -1.0580 0.6387 0.9357 2.1752   
  
Coefficients:  
 Estimate Std. Error z value Pr(>|z|)   
(Intercept) -4.20453 1.04975 -4.005 6.19e-05 \*\*\*  
Age 0.05871 0.01889 3.108 0.00188 \*\*   
SexM 1.12014 1.15531 0.970 0.33227   
Age:SexM 0.00951 0.02094 0.454 0.64978   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for binomial family taken to be 1)  
  
 Null deviance: 1261.0 on 916 degrees of freedom  
Residual deviance: 1100.5 on 913 degrees of freedom  
AIC: 1108.5  
  
Number of Fisher Scoring iterations: 4

* Let’s check out the visual of the fits

|  |
| --- |
|  |

* We can also include more than one numeric predictor and add in polynomial terms increases flexibility as well!
  + The plot below shows a model with a linear and quadratic term included

|  |
| --- |
|  |

### General Logistic Regression Model

Suppose we have a set of covariates and a binary response.

* We can model the conditional (posterior) probabilities as follows:
* Since we have only two classes for , we know . This means we can write the following:

Another way to write the same model is using the *log-odds*:

* The parameters quantify the impact of the covariates to the prediction of
* The group used in the denominator (class 0 in our formulation above) is called the *reference group*. The choice of reference group is arbitrary as the estimates of the conditional (posterior) probabilities are same.

### Fitting the Logistic Regression Model

For the MLR model we said we could use least squares or maximum likelihood to fit the model and the fit (estimated coefficients) would be the same.

* Here, the use of least squares doesn’t make sense
* Maximum likelihood still works! However, we don’t get a closed form solution as we did in the MLR case
  + We can show that the equation to find the ‘best’ ’s is equivalent to minimizing the negative log loss over the training data!

   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
   
 

# Inference on the Logistic Regression Model

## Classification Using the Logistic Regression Model

We estimate our conditional probabilities using the fitted model:

As with our model, we can predict the class for a particular observation via the following rule:

The item is classified in group if , otherwise in group .

* Again, in the binary case, this is equivalent to

Logistic regression can be performed using the glm() function in base R. **Make sure your response variable is a factor.** Use family = "binomial" to specify the logistic regression model.

* We use the same formula notation as before!
  + response ~ predictors
  + Separate predictor terms with +
  + I() or poly() for polynomial terms
  + pred1\*pred2 for an interaction
  + pred1:pred2 for main effects and an interaction
  + glm() automatically creates dummy variables

Let’s fit a logistic regression model for our HeartDisease response. We’ll use Age, RestingBP, and Sex as predictors.

# Logistic regression  
heart\_glm = glm(HeartDiseaseFac ~ Age + RestingBP + Sex,   
 family = "binomial",   
 data = heart\_data)

The estimated coefficients are as follows.

heart\_coef <- heart\_glm$coefficients  
heart\_coef

(Intercept) Age RestingBP SexM   
-5.348843019 0.063339913 0.006618919 1.648167845

We can interpret as the *log-odds* when both Age and RestingBP are zero and Sex is Female.

* We have odds of heart disease vs not having heart disease in this case.
* Of course, zero values for Age and RestingBP don’t make sense!

The estimated value of can be interpreted as the amount the log-odds will *change* due to one unit *increase* in Age while *keeping the RestingBP level fixed* for a Sex = Female.

* We expect a unit change in log-odds.
* Equivalently, our odds will change by a multiplicative factor of ()
  + In other words, increase by percent ()

Suppose we have a new sample with Age = 60 and RestingBP = 130. We can compute the estimated conditional (posterior) probabilities of heart disease for Males and Females as follows:

In R, we can simply use the predict() function to compute the probabilities shown above.

newx <- data.frame(Age = c(60, 60),  
 RestingBP = c(130, 130),  
 Sex = c("M", "F"))  
predict(heart\_glm,   
 newdata = newx,  
 type = "response")

1 2   
0.7231577 0.3344806

* We can see that a Male with these values of Age and RestingBP is classified as having heart disease but a female would not.

Let’s check out our confusion matrix.

# Confusion matrix  
confusionMatrix(table(heart\_data$HeartDiseaseFac,   
 predicted = ifelse(predict(heart\_glm, type = "response") > 0.5,  
 1, 0)  
))

Confusion Matrix and Statistics  
  
 predicted  
 0 1  
 0 226 184  
 1 105 402  
   
 Accuracy : 0.6848   
 95% CI : (0.6537, 0.7148)  
 No Information Rate : 0.639   
 P-Value [Acc > NIR] : 0.001996   
   
 Kappa : 0.3506   
   
 Mcnemar's Test P-Value : 4.47e-06   
   
 Sensitivity : 0.6828   
 Specificity : 0.6860   
 Pos Pred Value : 0.5512   
 Neg Pred Value : 0.7929   
 Prevalence : 0.3610   
 Detection Rate : 0.2465   
 Detection Prevalence : 0.4471   
 Balanced Accuracy : 0.6844   
   
 'Positive' Class : 0

As we have only a couple of predictors, we can plot the decision boundary of the classifier!

* Here we need to find all values of Age and RestingBP that have the same conditional (posterior) probability of being heart disease or not (once for males and then once for females).
* The boundary (for females) is given by all the solutions of the linear equation (the estimated formula of log-odds):

|  |
| --- |
| Decision boundary of a 2-class logistic regression based classifier. |

Now the confusion matrix for Females:

confusionMatrix(table(heart\_data |>   
 filter(Sex == "F") |>   
 pull(HeartDiseaseFac),   
 predicted = ifelse(predict(heart\_glm,   
 newdata = heart\_data |> filter(Sex == "F"),   
 type = "response") > 0.5,  
 1, 0)  
 ))

Confusion Matrix and Statistics  
  
 predicted  
 0 1  
 0 140 3  
 1 47 3  
   
 Accuracy : 0.7409   
 95% CI : (0.6731, 0.8012)  
 No Information Rate : 0.9689   
 P-Value [Acc > NIR] : 1   
   
 Kappa : 0.0547   
   
 Mcnemar's Test P-Value : 1.193e-09   
   
 Sensitivity : 0.7487   
 Specificity : 0.5000   
 Pos Pred Value : 0.9790   
 Neg Pred Value : 0.0600   
 Prevalence : 0.9689   
 Detection Rate : 0.7254   
 Detection Prevalence : 0.7409   
 Balanced Accuracy : 0.6243   
   
 'Positive' Class : 0

* The boundary (for males) is given by all the solutions of the linear equation (the estimated formula of log-odds):

|  |
| --- |
| Decision boundary of a 2-class logistic regression based classifier. |

Now the confusion matrix for Males:

confusionMatrix(table(heart\_data |>   
 filter(Sex == "M") |>   
 pull(HeartDiseaseFac),   
 predicted = ifelse(predict(heart\_glm,   
 newdata = heart\_data |> filter(Sex == "M"),   
 type = "response") > 0.5,  
 1, 0)  
 ))

Confusion Matrix and Statistics  
  
 predicted  
 0 1  
 0 86 181  
 1 58 399  
   
 Accuracy : 0.6699   
 95% CI : (0.6343, 0.7041)  
 No Information Rate : 0.8011   
 P-Value [Acc > NIR] : 1   
   
 Kappa : 0.2159   
   
 Mcnemar's Test P-Value : 2.985e-15   
   
 Sensitivity : 0.5972   
 Specificity : 0.6879   
 Pos Pred Value : 0.3221   
 Neg Pred Value : 0.8731   
 Prevalence : 0.1989   
 Detection Rate : 0.1188   
 Detection Prevalence : 0.3688   
 Balanced Accuracy : 0.6426   
   
 'Positive' Class : 0

* Perhaps an interaction term would help us here!

Note: **Ideally, we should use cross-validation, training-testing sets to estimate the accuracy, as we have learned before!**

## Odds and log-odds

Let’s investigate the concepts of *odds* and *log-odds* in more detail.

Reconsider the heart disease data with classes (1 and 0) and only one covariate, $X = `Age`$.

* The odds are defined as
  + If then the odds of class 1 are .
  + In contrast, if then the odds are
* Thus greater odds relate to higher posterior probability of class 1.
* In terms of log-odds,
  + Model implies that for a one unit increase in (Age), the log-odds will change by units
  + This change in log-odds does *not* depend on the value of , just that it increases by 1
  + Equivalently, due to one unit increase in , the *odds* gets multiplied by .

## Hypothesis testing and confidence intervals

Reconsider our logistic regression model with Age, RestingBP, and Sex. We can get a summary of the fit as follows.

summary(heart\_glm)

Call:  
glm(formula = HeartDiseaseFac ~ Age + RestingBP + Sex, family = "binomial",   
 data = heart\_data)  
  
Deviance Residuals:   
 Min 1Q Median 3Q Max   
-2.1088 -1.0748 0.6426 0.9212 2.3102   
  
Coefficients:  
 Estimate Std. Error z value Pr(>|z|)   
(Intercept) -5.348843 0.668422 -8.002 1.22e-15 \*\*\*  
Age 0.063340 0.008384 7.555 4.20e-14 \*\*\*  
RestingBP 0.006619 0.004205 1.574 0.115   
SexM 1.648168 0.190003 8.674 < 2e-16 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for binomial family taken to be 1)  
  
 Null deviance: 1261.0 on 916 degrees of freedom  
Residual deviance: 1098.3 on 913 degrees of freedom  
AIC: 1106.3  
  
Number of Fisher Scoring iterations: 4

* summary() of the fit produces -tests for coefficient of each covariate
  + This is an approximate (large sample) test
  + We test whether vs after accounting for the other predictors in the model
  + The test statistic is
  + This statistics approximately follows a distribution under . We reject for large (in absolute value) values.
  + P-value is

We can produce large sample confidence intervals for as

where is the quantile of the distribution. For example, a confidence intervals for is

We can interpret this intervals as follows:

* with every unit increase in the value of (Age in our example), we can expect an *increase in log-odds* by an amount of 0.0469067 to 0.0797731 (while holding RestingBP and Sex constant)
* Equivalently, every unit increase in the value of , the *odds* will be changed by a factor of 1.0480243 to 1.0830413 (while holding RestingBP and Sex constant)
  + In other words, the increase in odds will be between 4.802 percent and 8.304 percent

# Logistic regression with multiple classes

We can extend logistic regression presented for two classes to the case of multiple classes.

* We call this a *Multinomial Logistic Regression* model.
* Suppose we have classes, and we take the -th class as the reference. The log-odds of classes vs the reference class are modeled as follows:

Some algebra shows that the corresponding conditional (posterior) probabilities are as follows:

We can similarly build a classification rule as follows.

An item with covariate is predicted to be in class if the estimated probability is larger than the other posterior probabilities.

# Issues to consider

## Separation of Data

There are some situations where logistic regression might not perform well. One such situation is *complete (or quasi-complete) separation* of the data.

This situation happens when the outcome variable separates a predictor completely. This leads to perfect prediction of the outcome by the predictor.

* Consider the following data set with binary response and two predictors and .
* The figure below shows relationship between and the two predictors.
* In such a case, logistic regression may produce unreasonable over-inflated estimates of regression coefficients.

|  |
| --- |
| Simulated data set. The response separates the data completely – the boundary (dashed line) is . Negative values of corresponds to , and positive values corresponds to . |

glm(y ~ x1 + x2, family = binomial())

Warning: glm.fit: algorithm did not converge

Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

Call: glm(formula = y ~ x1 + x2, family = binomial())  
  
Coefficients:  
(Intercept) x1 x2   
 14.54 -1578.34 -1550.26   
  
Degrees of Freedom: 199 Total (i.e. Null); 197 Residual  
Null Deviance: 277.1   
Residual Deviance: 2.051e-06 AIC: 6

In general, if there is a linear combination that is completely separated by , logistic regression will fail to produce reasonable results! Figure shows one such example where the data is completely separated by the line .

* Although the examples above shows complete separation using continuous predictors, it is more like to happen when using categorical predictors coded by dummy variables. Small sample size might contribute to this problem as well.
  + In such situations, applying other classification methods (e.g., LDA) is preferred.

## Sample Size

Since logistic regression usually deals with binary outcome, often it requires a larger sample size that linear regression.

Multinomial logistic regression requires even more sample size than binary logistic regression due to the fact that it estimates parameters for multiple classes.

## Combinations Present in the Data

In the presence of categorical predictors, it might happen that there are some combinations of predictor and response values that are not present in the data.

In such a case, logistic fit may become unstable, or might even fail to converge.

# High-Dimensional Problems

When the number of predictors is larger than (or close to) the sample size , the methods described in this section suffer from numerical instability or simply can not be applied to the data. We saw these issues in the MLR case as well!

* We can apply similar strategies discussed for linear regression here as well: regularization/shrinkage and dimension reduction methods.
* Like linear regression, we can develop ridge, lasso and elastic net methods for logistic regression.
* All these methods are available in glmnet() package.
  + As before, these methods shrink the regression coefficients towards zero and can be used in high-dimensional setting.
  + We show the lasso based logistic regression fit of the heart disease data with the penalty parameter chosen by CV below.
  + We only include numeric predictors here but could add in our own dummy variables for each categorical predictor
  + Note: **glmnet() automatically scales the predictors before estimating the regression coefficients, and then outputs the coefficients in the original scale**

set.seed(1102)  
# CV to choose lambda  
logit\_cv <- cv.glmnet(x = as.matrix(heart\_data |>   
 dplyr::select(-starts\_with("Heart"), -Sex, -ChestPainType, -RestingECG, -ExerciseAngina, -ST\_Slope)),   
 y = heart\_data$HeartDiseaseFac,  
 family = binomial(),  
 alpha = 1)

|  |
| --- |
| CV results for logistic regression of wines data with lasso penalty. |

# Final fit with lambda chosen by 1-SE rule  
heart\_lasso <- glmnet(x = as.matrix(heart\_data |>   
 dplyr::select(-starts\_with("Heart"), -Sex, -ChestPainType, -RestingECG, -ExerciseAngina, -ST\_Slope)),  
 y = heart\_data$HeartDiseaseFac,  
 family = binomial(),  
 alpha = 1,  
 lambda = logit\_cv$lambda.1se)

# Estimated coefs  
coef(heart\_lasso)

7 x 1 sparse Matrix of class "dgCMatrix"  
 s0  
(Intercept) 2.8547940686  
Age 0.0005390516  
RestingBP .   
Cholesterol -0.0016404417  
FastingBS 0.6981182823  
MaxHR -0.0220519396  
Oldpeak 0.6536631089

Dimension reduction method like PCA can still be applied to the predictors before building classifiers!