Logistic Regression with R

Advanced Statistics - MSM ITB

Pima Indians Diabetes Dataset

Pima Indians Diabetes Dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

Pima Indians Diabetes Dataset

pregnant: Number of times pregnant

glucose: Plasma glucose concentration (glucose tolerance test)

pressure: Diastolic blood pressure (mm Hg)

triceps: Triceps skin fold thickness (mm)

Pima Indians Diabetes Dataset

insulin: 2-Hour serum insulin (mu U/ml)

mass: Body mass index (weight in kg/(height in m)\^2)

pedigree: Diabetes pedigree function

age: Age (years)

diabetes: Class variable (test for diabetes)

Import Data

- Please download the dataset via https://github.com/apriandito/advanced-statistics/blob/main/data/pima-indians.xlsx
- Upload to the data folder in your Rstudio cloud project directory
- Use the following command

Command: Import Data

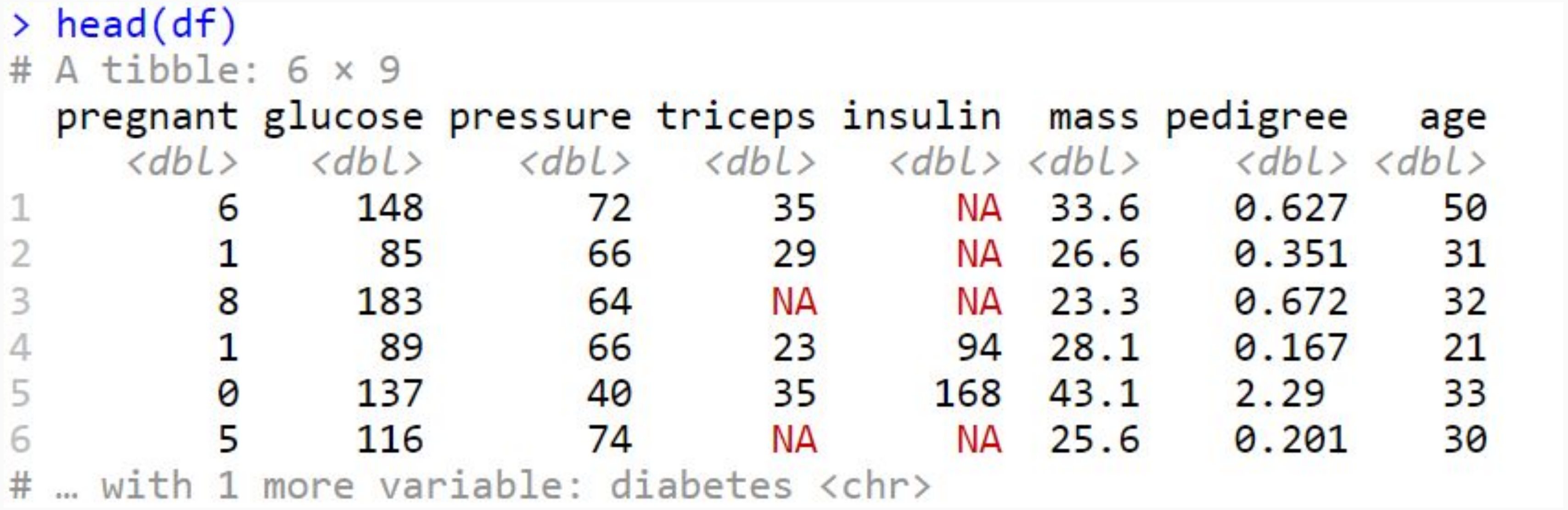
```
# Import Excel Data

df <- read_xlsx("data/pima-indians.xlsx")
```

Data Exploration

Please check the data that has been imported. You can use commands like head(), glimpse() from the dplyr package, skim() from the skimr package, or ggpairs() from the ggally package.

Data Exploration



```
> head(df)
# A tibble: 6 × 9
  pregnant glucose pressure triceps insulin  mass pedigree   age
     <dbl>   <dbl>    <dbl>   <dbl>   <dbl> <dbl>    <dbl> <dbl>
1        6     148       72      35      NA  33.6    0.627    50
2        1      85       66      29      NA  26.6    0.351    31
3        8     183       64      NA      NA  23.3    0.672    32
4        1      89       66      23      94  28.1    0.167    21
5        0     137       40      35     168  43.1    2.29     33
6        5     116       74      NA      NA  25.6    0.201    30
# … with 1 more variable: diabetes <chr>
```

Data Preprocessing

The results of data exploration show that there are empty values in the data, and some formats are not appropriate. please do the data processing using the following command

Command: Data Preprocessing

```
# Fix Data

df_fix <- df %>%

  dplyr::select(-c(triceps, insulin)) %>%

  mutate(diabetes = case_when(

    diabetes == "pos" ~ 1,

    diabetes == "neg" ~ 0

  )) %>%

  mutate(diabetes = as_factor(diabetes)) %>%

  drop_na()
```

Split Data

To start our modeling we will divide the data into train and test. please use the following command.

Command: Split Data

```
# Create Split Index

split_index <- sort(sample(nrow(df_fix), nrow(df_fix) * 0.7))


# Create Train Data

df_train <- df_fix[split_index, ]


# Create Test Data

df_test <- df_fix[-split_index, ]
```

Modeling

The modeling process can be performed using the following command. You can start by entering all the variables first, then adjust the variables accordingly by selecting only the significant variables.

Command: Modeling

```
# Create Logistic Regression Model

lr_model <- glm(diabetes ~ pregnant + glucose + mass +
pedigree, data = df_train, family = binomial)


# Check Model Summary

summary(lr_model)
```

Check Model and Assumption

The modeling process can be performed using the following command. You can start by entering all the variables first, then adjust the variables accordingly by selecting only the significant variables.

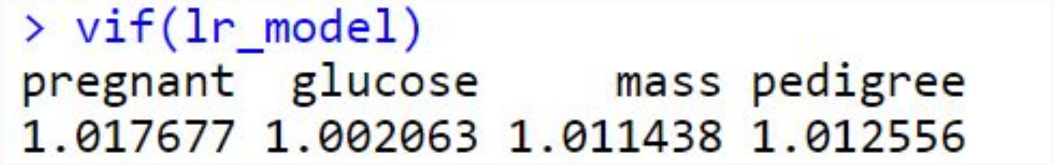
Command: Check Model and Assumption

```
# Plot Residual
plot(lr_model, 1)

# Plot QQ-plot
plot(lr_model, 2)

# Check Multicolinerity
vif(lr_model)
```

Check Model and Assumption



```
> vif(lr_model)
pregnant  glucose      mass pedigree 
1.017677 1.002063 1.011438 1.012556 
```

A rule of thumb for interpreting the variance inflation factor:

1 = not correlated.

Between 1 and 5 = moderately correlated.

Greater than 5 = highly correlated.

Model Evaluation

The modeling process can be performed using the following command. You can start by entering all the variables first, then adjust the variables accordingly by selecting only the significant variables.

Command: Model Evaluation (1)

```
# Make a prediction
pred <- predict(lr_model,
  newdata = df_test,
  type = "response"
)

# Convert to a Class
df_pred_class <- ifelse(pred > 0.5, 1, 0)
```

Command: Model Evaluation (2)

```
# Confusion Matrix
c_table <- table(
  actual = df_test[["diabetes"]],
  predicted = df_pred_class
)

# Get Performance Statistics
confusionMatrix(c_table)
```

Thank You