A4_juh3

2023-04-04

Question1

```
# load package and data
library(tidyverse)
library(tidymodels)
library(GGally)
library(reshape2)
library(vip)
tidymodels_prefer()
hd <- (
  read_csv('heart.csv',show_col_types = FALSE)
)
```

Description:

I found this dataset in kaggle which is a dataset for heart attack classification. This Heat Disease Dataset was compiled in 1988 and comprises four separate databases: Cleveland, Hungary, Switzerland, and Long Beach V. It comprises 76 different attributes, including the predicted attribute, but in all published experiments, only a subset of 14 attributes were utilized.

Initial investigation:

The attribute name is defined as following:

(1). Age : Age of the patient

(2). Sex : Sex of the patient

(3). exang: exercise induced angina (1 = yes; 0 = no)

(4). ca: number of major vessels (0-3)

(5). cp : Chest Pain type chest pain type

Value 1: typical angina Value 2: atypical angina Value 3: non-anginal pain Value 4: asymptomatic

(6). trtbps : resting blood pressure (in mm Hg)

(7). chol : cholestoral in mg/dl fetched via BMI sensor

(8). fbs : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)

(9). rest_ecg : resting electrocardiographic results

Value 0: normal Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV) Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria

(10). thalach : maximum heart rate achieved

(11). slp

(12). caa

(13). thall

(14). target : 0= less chance of heart attack 1= more chance of heart attack

```
# summary of data
hd |> glimpse() |> summary()
```

Firstly, there are 8 categorical attributes (sex, cp, fbs, restecg, exng, slp, caa, thall) and 5 numerical attributes (age, trtbps, chol, thalachh, oldpeak). There are some unusual:

1. The average cholesterol level (chol) is 246.3 and with 1st Qu. 211.0 and 3rd Qu.274.5, but the maximum is 564.0, this could be a outlier.
2. The maximum heart rate (thalachh) is 202.0 which is unusual.

```
# distribution of cholesterol level
hd |> ggplot(aes(x = chol)) +
  geom_histogram(binwidth = 5, fill = "blue", color = "black") +
  labs(x = "Cholesterol level", y = "Count", title = "Distribution of Attribute")

# distribution of old peak
hd |> ggplot(aes(x = oldpeak)) +
  geom_histogram(binwidth = 0.5, fill = "blue", color = "black") +
  labs(x = "Old peak", y = "Count", title = "Distribution of Attribute")




# heatmap for the correlation
hd_num <- hd[c('age','trtbps', 'chol', 'thalachh', 'oldpeak')]
hd_num_tbl <- as_tibble(hd_num)

corr_matrix <- hd_num_tbl |>
  cor()
corr_matrix <- round(corr_matrix,2)

ggplot(data = melt(corr_matrix), aes(x=Var1, y=Var2, fill=value)) +
  geom_tile() +
  geom_text(aes(Var2, Var1, label = value),
          color = "black", size = 4)
```

We can see cholesterol level is normally distributed and most have 0 old peak. By looking at the heat map of correlation for numerical attributes, we can see there's no highly correlated attributes, the age and maximum heart rate are negatively correlated.

preprocessing steps:

The categorical varaibles are represented by integer in the original dataset, we first convert it to factor. We checked there's no NA value. After that, we do train/test split which using %75 data as training set. Then, we create recipes: 1. performs a stepwise centering for numerical variables 2. sacle numerical variables 3. performs a stepwise feature selection operation based on the Near Zero Variance (NZV) criterion for numerical variables 4. one-hot endcoding for categorical variables.

```
# since categorical variable are represented by integers, we have to first convert it to character

hd[,"sex"] <- as.factor(hd$sex)
```

```
hd[,"cp"] <- as.factor(hd$cp)
hd[,"fbs"] <- as.factor(hd$fbs)
hd[,"restecg"] <- as.factor(hd$restecg)
hd[,"exng"] <- as.factor(hd$exng)
hd[,"slp"] <- as.factor(hd$slp)
hd[,"caa"] <- as.factor(hd$caa)
hd[,"thall"] <- as.factor(hd$thall)
hd[,"output"] <- as.factor(hd$output)
hd <- hd %>% drop_na()
hd |> glimpse()


# split data into train and test data
split <- initial_split(hd, prop = 0.75, strata = sex)
train_data <- training(split)
test_data <- testing(split)


# preprocessing data using recipe
hd_recipe <- (
  recipe(output ~., data = train_data)
  |> step_center(all_numeric())
  |> step_scale(all_numeric())
  |> step_nzv(all_numeric())
  |> step_dummy(all_nominal(), one_hot = TRUE,-output)
)

hd_prepped <- prep(hd_recipe)

hd_prepped |> bake(train_data) |> rsample::vfold_cv(v=10)
```

Model

The model I choose is random forest since the data has many features. And from previous step, we found that there may be some outliers in the data, random forest are good at handling them. Random Forest can provide feature importance measures, which can help understand which features are driving the predictions.

```
model_decision <- parsnip::rand_forest(mtry=tune(), min_n = tune(), trees = 1000) |>
  set_engine("ranger") |>
  set_mode("classification")
```

```
#create the work flow
hd_wflow <- workflow() |>
  add_recipe(hd_prepped) |>
  add_model(model_decision)
```

Parameter tuning

I tuned hyperparameters “mtry” and “min_n” in random forest. mtry controls the number of features that are used at each split of the decision tree. And “min_n” controls the minimum number of observations in the leaf of the tree.

```
tt<-tune_grid(
  hd_wflow,
  grid = 100,
```

```
  resamples = vfold_cv(train_data)
)


#select best parameter using accuracy
show_best(tt,metric = "accuracy")

#select best parmeter using roc
show_best(tt,metric = "roc_auc")


autoplot(tt)
```

Fitting the best model

Based on my our hyperparamter turing, we choose the highest roc_auc as the best parameter and refit to the data.

```
final_hd <- hd_wflow |> finalize_workflow(select_best(tt,metric = "roc_auc"))
final_hd




# last fit
hd_fit <- last_fit(final_hd,split)
hd_fit
# metric on the test set
collect_metrics(hd_fit)
```

The result is not bad. Acc 0.79, ROC_AUC 0.86.

Variable Importance plot

Then we plot the variable importance plot, we can see caa is the most important factor.

```
# fit model again, set imprtance = "permutation"
imp_hd <- model_decision |>
  finalize_model(select_best(tt)) |>
  set_engine("ranger", importance = "permutation")


workflow() |>
  add_recipe(hd_recipe) |>
  add_model(imp_hd) |>
  fit(train_data) |>
  pull_workflow_fit() |>
  vip(aesthetics = list(fill = "red"))
```

reference: https://juliasilge.com/blog/ikea-prices/