Predictive Analysis Compitition

Siyuan Zhu

2023-12-4

Contents

1 Part I: Introduction

The primary question guiding this analysis is, "How do various features and conditions influence the sale price of a used car?" Our aim is to develop a predictive model for the response variable, the sale price, as a function of the given 79 predictor variables.

To do this, we will engage in extensive exploratory data analysis, focusing on identifying and eliminating variables that show little correlation with the sale price. Additionally, we will be attentive to removing predictors that exhibit high inter-correlations to avoid multicollinearity, which could inflate variance and lead to less accurate predictions.

A significant focus of my analysis is on handling missing data, employing tailored strategies for numerical and categorical variables. For numerical data, we use median imputation for minimal missing values and linear regression imputation for significant gaps. Categorical variables undergo mode imputation for minimal missing data and decision tree imputation for more substantial cases.

This report will document my approach, from data preprocessing to model selection and interpretation, offering a comprehensive view of how used car features add up to their price tag.

2 Part II: Data Processing and Analysis

2.1 Set the price in test dataset to be 0

```
test$price <- 0
```

2.2 Combine train and test datasets

```
train$dataset_type <- 'train'
test$dataset_type <- 'test'
idtest <- test[,1]
combine = rbind(train, test)
```

2.3 Dropping unnecessary variables

1. Dropping Id as it is unnecessary for the prediction process. 2. Dropping **description** and **major_options**since it's difficult to fit into a model 3. **wheel_system** and **wheel_system_display** are talking about the same thing, so dropping the latter one. Same for **transmission_display**

```
combine = combine[, -1]
combine <- subset(combine, select = -description)
combine <- subset(combine, select = -major_options)
combine <- subset(combine, select = -wheel_system_display)
combine <- subset(combine, select = -transmission_display)
```

2.4 Check Missing Data

```
colSums(is.na(combine) | combine == "")
```

```
##                make_name               model_name                trim_name
##                        0                        0                        9
##                body_type fuel_tank_volume_gallons                fuel_type
##                        0                        4                      669
##     highway_fuel_economy        city_fuel_economy                    power
##                     5398                     5398                     5538
##                   torque             transmission             wheel_system
##                     6249                     1526                      350
##         wheelbase_inches      back_legroom_inches     front_legroom_inches
##                        0                        0                        0
```

```
##           length_inches            width_inches           height_inches
##                       0                       0                       0
##             engine_type     engine_displacement              horsepower
##                     672                     389                     389
##            daysonmarket          exterior_color          interior_color
##                       0                       3                       0
##         maximum_seating                    year                   fleet
##                       0                       0                   23517
##           frame_damaged       franchise_dealer          franchise_make
##                   23517                       0                   12104
##           has_accidents                   isCab                  is_cpo
##                   23517                   23517                   46983
##                  is_new            listed_date           listing_color
##                       0                       0                       0
##                 mileage           owner_count                 salvage
##                    2245                   24741                   23517
##           seller_rating                   price            dataset_type
##                     677                       0                       0
```

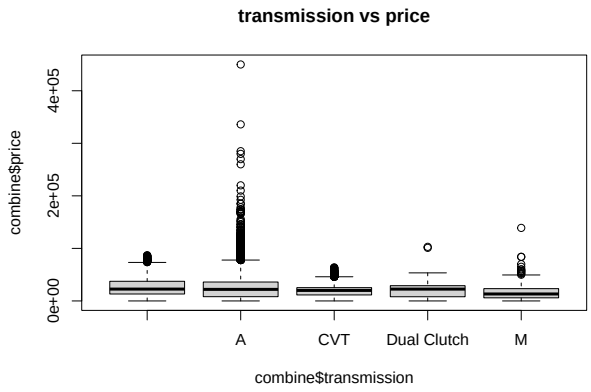
```
nrow(combine)
```

```
## [1] 50000
```

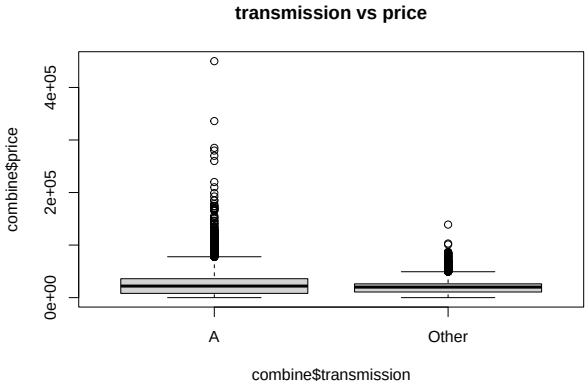
2.5 Consolidated similar types into 'Other'

transmission: The visualization indicates 'A' transmission type has a distinct price distribution compared to other categories. To refine the model, I've consolidated the varied but less differentiated transmission types into 'Other'.

```
boxplot(combine$price ~ combine$transmission, data = combine, main = "transmission vs price")
```




```
combine$transmission <- ifelse(combine$transmission == 'A', 'A', 'Other')
boxplot(combine$price ~ combine$transmission, data = combine, main = "transmission vs price")
```



2.6 Imputing Missing Data

Initially, for missing data handling, I adopted a straightforward approach: using the mode for categorical variables and the median for numerical variables. However, as the analysis progressed, it became clear that this method, while simple, was not optimizing the model's performance. A turning point came when I switched to predictive imputation methods. Specifically, for numerical variables with minimal missing values, I continued using the median, but for those with significant missing data, I employed linear regression imputation. Similarly, for categorical variables, I used mode imputation for minimal missing values and decision tree imputation for significant gaps. This nuanced approach notably improved the model's accuracy, as evidenced by a substantial reduction in the RMSE.The details are as followed:

2.6.0.1 Numerical variables

1. Variables with minimal NA values will be replaced by median, which is more robust to outliers than the mean.
2. NA values in **owner_count** will be replaced by 0, indicating no prevous owner.
3. Variables with about or more than 50% NA values will be removed.
4. Variables with significant missing values will use predictive imputation by using linear regression.

2.6.0.2 Categorical variables

1. Variables with minimal NA values will be replaced by mode (e.g., **exterior_color**)
2. Based on the introduction, NA values in **is_cpo** indicate not certified (False)
3. Variables with significant missing values will use predictive imputation by using decision tree.

```
combine$fuel_tank_volume_gallons[is.na(combine$fuel_tank_volume_gallons)] <- median(combine$fuel_tank_v

combine$owner_count[is.na(combine$owner_count)] = 0

non_missing <- combine[!is.na(combine$engine_displacement), ]
missing <- combine[is.na(combine$engine_displacement), ]
model <- lm(engine_displacement ~ engine_type + fuel_tank_volume_gallons + wheelbase_inches + length_in
predicted_values <- predict(model, newdata = missing)
combine$engine_displacement[is.na(combine$engine_displacement)] <- predicted_values
```

```r
non_missing <- combine[!is.na(combine$highway_fuel_economy), ]
missing <- combine[is.na(combine$highway_fuel_economy), ]
model <- lm(highway_fuel_economy ~ fuel_tank_volume_gallons + height_inches + engine_displacement, data
predicted_values <- predict(model, newdata = missing)
combine$highway_fuel_economy[is.na(combine$highway_fuel_economy)] <- predicted_values

non_missing <- combine[!is.na(combine$mileage), ]
missing <- combine[is.na(combine$mileage), ]
model <- lm(mileage ~ fuel_tank_volume_gallons + year + owner_count + height_inches + highway_fuel_econ
predicted_values <- predict(model, newdata = missing)
combine$mileage[is.na(combine$mileage)] <- predicted_values

combine$seller_rating[is.na(combine$seller_rating)] <- median(combine$seller_rating, na.rm = TRUE)

non_missing <- combine[!is.na(combine$horsepower), ]
missing <- combine[is.na(combine$horsepower), ]
model <- lm(horsepower ~ fuel_tank_volume_gallons + highway_fuel_economy + wheelbase_inches + engine_di
predicted_values <- predict(model, newdata = missing)
combine$horsepower[is.na(combine$horsepower)] <- predicted_values
```

```r
non_missing <- combine[!is.na(combine$transmission) & combine$transmission != "", ]
missing <- combine[is.na(combine$transmission)| combine$transmission == "", ]
model <- rpart(transmission ~ highway_fuel_economy + width_inches + height_inches + horsepower, data=no
predicted_values <- predict(model, newdata = missing, type = "class")
combine$transmission[is.na(combine$transmission) | combine$transmission == ""] <- predicted_values
```

```r
combine$exterior_color[combine$exterior_color  == "" | is.na(combine$exterior_color)] <- "Black"
combine$trim_name[combine$trim_name == "" | is.na(combine$trim_name)] = "SE FWD"

non_missing <- combine[!is.na(combine$wheel_system) & combine$wheel_system != "",  ]
missing <- combine[is.na(combine$wheel_system)| combine$wheel_system == "", ]
model <- rpart(wheel_system ~ fuel_tank_volume_gallons  + wheelbase_inches   , data=non_missing, method=
predicted_values <- predict(model, newdata = missing, type = "class")
combine$wheel_system[is.na(combine$wheel_system)| combine$wheel_system == ""] <- predicted_values
```

```r
combine$fuel_type[is.na(combine$fuel_type) | combine$fuel_type == "" ] <- "Gasoline"
combine$is_cpo[combine$is_cpo == ""] <- FALSE

non_missing <- combine[!is.na(combine$engine_type) & combine$engine_type != "",  ]
missing <- combine[is.na(combine$engine_type) | combine$engine_type == "", ]
model <- rpart(engine_type ~ fuel_tank_volume_gallons  + highway_fuel_economy + wheelbase_inches  + leng
predicted_values <- predict(model, newdata = missing, type = "class")
combine$engine_type[is.na(combine$engine_type) | combine$engine_type == ""] <- predicted_values
```

```r
non_missing <- combine[!is.na(combine$power) & combine$power != "", ]
missing <- combine[is.na(combine$power) | combine$power == "", ]
model <- rpart(power ~ fuel_tank_volume_gallons + highway_fuel_economy + horsepower + engine_displaceme
               data = non_missing,
               method = "class",
               control = rpart.control(maxdepth = 5))
predicted_values <- predict(model, newdata = missing, type = "class")
combine$power[is.na(combine$power) | combine$power == ""] <- predicted_values
```

```
non_missing <- combine[!is.na(combine$torque) & combine$torque != "", ]
missing <- combine[is.na(combine$torque) | combine$torque == "", ]
model <- rpart(torque ~ fuel_tank_volume_gallons + highway_fuel_economy + horsepower + engine_displacem
               data = non_missing,
               method = "class",
               control = rpart.control(maxdepth = 5))
predicted_values <- predict(model, newdata = missing, type = "class")
combine$torque[is.na(combine$torque) | combine$torque == ""] <- predicted_values
```

```
combine$franchise_make[is.na(combine$franchise_make) | combine$franchise_make == "" ] <- "Chevrolet"
combine <- subset(combine, select = -has_accidents)
combine <- subset(combine, select = -salvage)
combine <- subset(combine, select = -isCab)
combine <- subset(combine, select = -frame_damaged)
combine <- subset(combine, select = -fleet)
```

2.7 Transformed categorical variables into factors

```
Charcol <- names(combine[,sapply(combine, is.character)])
Charcol
```

```
##  [1] "make_name"        "model_name"       "trim_name"        "body_type"
##  [5] "fuel_type"        "power"            "torque"           "transmission"
##  [9] "wheel_system"     "engine_type"      "exterior_color"   "interior_color"
## [13] "franchise_dealer" "franchise_make"   "is_cpo"           "is_new"
## [17] "listed_date"      "listing_color"    "dataset_type"
```

```
combine$is_cpo <- ifelse(combine$is_cpo == TRUE | combine$is_cpo == "True", 1, 0)
combine$is_new <- ifelse(combine$is_new == TRUE| combine$is_new == "True", 1, 0)
combine[Charcol[-length(Charcol)]] <- lapply(combine[Charcol[-length(Charcol)]], as.factor)
combine[Charcol[-length(Charcol)]] <- lapply(combine[Charcol[-length(Charcol)]], function(x) as.integer
```

2.8 Correlation Matrix

```
#Correlation Matrix
train <- subset(combine, dataset_type == 'train')[, !(names(combine) %in% c('dataset_type'))]
numericVars <- which(sapply(combine, is.numeric))
numerical_data <- train[, numericVars]
cor_matrix <- cor(numerical_data, use = "pairwise.complete.obs")
cor_matrix
```

```
numerical_data |>
  pivot_longer(1:24,names_to = 'var',values_to = 'values')|>
  group_by(var)|>
```

```
  summarize(r = round(cor(train$price, values,use = "complete.obs"),2), p = round(cor.test(train$price,
  arrange(desc(abs(r)))
```

```
## # A tibble: 24 x 3
##    var                          r     p
##    <chr>                    <dbl> <dbl>
##  1 horsepower                0.64     0
##  2 torque                    0.5      0
##  3 power                     0.49     0
##  4 width_inches              0.46     0
##  5 fuel_tank_volume_gallons  0.44     0
##  6 length_inches             0.43     0
##  7 wheelbase_inches          0.43     0
##  8 engine_displacement       0.41     0
##  9 height_inches             0.4      0
## 10 highway_fuel_economy     -0.39     0
## # i 14 more rows
```

2.9 Removing variables with high collinearity

I am dropping a variable if two variables are highly correlated. For example, **highway_fuel_economy** and **city_fuel_economy** has a correlation of 0.91, so I will drop the latter one which has lower correlation with **price**.

```
combine <- subset(combine, select = -city_fuel_economy)
combine <- subset(combine, select = -length_inches)
```

2.10 Removing variable based on the output of Lasso

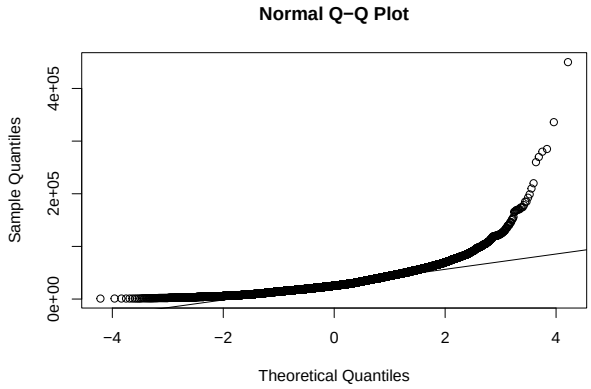
```
combine <- subset(combine, select = -listed_date)
```

2.11 Composing train and test sets

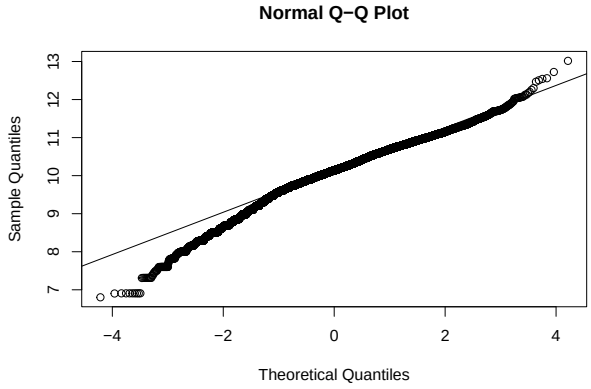
```
train <- subset(combine, dataset_type == 'train')[, !(names(combine) %in% c('dataset_type'))]
test <- subset(combine, dataset_type == 'test')[, !(names(combine) %in% c('dataset_type'))]
```

2.12 Normalizaing the response variable

```r
qqnorm(train$price)
qqline(train$price)
```




```r
train$price <- log(train$price)
qqnorm(train$price)
qqline(train$price)
```



3 Part III: Model Selection

In the quest for the most effective predictive model, my journey began with the exploration of several techniques, including LASSO regression, regression trees, Random Forest with ranger, SVM, and finally XGBoost. Each model offered unique insights, but it was the XGBoost model that consistently delivered the lowest RMSE, emerging as the most promising tool for the analysis. After fine-tuning various parameters within the XGBoost framework, I discovered that setting 'nrounds' to 3000 provided an optimal balance, significantly enhancing the model's accuracy. This iterative process of model selection and parameter optimization was critical in developing a robust predictive model for used car prices.

3.1 Final model (with lowest RMSE): XGBoost

```
set.seed(3184)
dtrain <- xgb.DMatrix(data = as.matrix(train[, -ncol(train)]), label = train[, ncol(train)])
dtest <- xgb.DMatrix(data = as.matrix(test[, -ncol(test)]))

params <- list(
    objective = "reg:squarederror",
    eta = 0.3,
    max_depth = 6,
    subsample = 0.8,
    colsample_bytree = 0.8
)
nrounds <- 2000

cv_results <- xgb.cv(
  params = params,
  data = dtrain,
  nrounds = nrounds,
  nfold = 5,
  metrics = "rmse",
  early_stopping_rounds = 50,
  verbose = TRUE
)
```

```
optimal_nrounds <- which.min(cv_results$evaluation_log$test_rmse_mean)
xgb_model <- xgb.train(params = params, data = dtrain, nrounds = 2000)
xgb_predict_test <- predict(xgb_model, dtest)
xgb_predict_test_transformed <- exp(xgb_predict_test)
submissionFile <- data.frame(id = idtest, price = xgb_predict_test_transformed)
write.csv(submissionFile, 'xgb_sample_submission.csv', row.names = FALSE)
```