Graduate Admission

2025-01-08

1. Load and Explore the Dataset

We are going to predict Chance of Admission on the basis of various predictors.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
data <- read.csv("adm_data.csv")

# View the first few rows of the dataset
head(data)
```

```
##   Serial.No. GRE.Score TOEFL.Score University.Rating SOP LOR CGPA Research
## 1          1       337         118                 4 4.5 4.5 9.65        1
## 2          2       324         107                 4 4.0 4.5 8.87        1
## 3          3       316         104                 3 3.0 3.5 8.00        1
## 4          4       322         110                 3 3.5 2.5 8.67        1
## 5          5       314         103                 2 2.0 3.0 8.21        0
## 6          6       330         115                 5 4.5 3.0 9.34        1
##   Chance.of.Admit
## 1            0.92
## 2            0.76
## 3            0.72
## 4            0.80
## 5            0.65
## 6            0.90
```

```
# Remove the first column
data <- data[, -1]

# Print basic statistics and structure
print("\nSummary Statistics:")
```

```
## [1] "\nSummary Statistics:"
```

```
print(summary(data))
```

```
##    GRE.Score      TOEFL.Score    University.Rating      SOP
```

```
##  Min.   :290.0   Min.   : 92.0   Min.   :1.000     Min.   :1.0
##  1st Qu.:308.0   1st Qu.:103.0   1st Qu.:2.000     1st Qu.:2.5
##  Median :317.0   Median :107.0   Median :3.000     Median :3.5
##  Mean   :316.8   Mean   :107.4   Mean   :3.087     Mean   :3.4
##  3rd Qu.:325.0   3rd Qu.:112.0   3rd Qu.:4.000     3rd Qu.:4.0
##  Max.   :340.0   Max.   :120.0   Max.   :5.000     Max.   :5.0
##       LOR             CGPA          Research      Chance.of.Admit
##  Min.   :1.000   Min.   :6.800   Min.   :0.0000   Min.   :0.3400
##  1st Qu.:3.000   1st Qu.:8.170   1st Qu.:0.0000   1st Qu.:0.6400
##  Median :3.500   Median :8.610   Median :1.0000   Median :0.7300
##  Mean   :3.453   Mean   :8.599   Mean   :0.5475   Mean   :0.7244
##  3rd Qu.:4.000   3rd Qu.:9.062   3rd Qu.:1.0000   3rd Qu.:0.8300
##  Max.   :5.000   Max.   :9.920   Max.   :1.0000   Max.   :0.9700
```

```
print("\nStructure of the Dataset:")
```

```
## [1] "\nStructure of the Dataset:"
```

```
print(str(data))
```

```
## 'data.frame':    400 obs. of  8 variables:
##  $ GRE.Score        : int  337 324 316 322 314 330 321 308 302 323 ...
##  $ TOEFL.Score      : int  118 107 104 110 103 115 109 101 102 108 ...
##  $ University.Rating: int  4 4 3 3 2 5 3 2 1 3 ...
##  $ SOP              : num  4.5 4 3 3.5 2 4.5 3 3 2 3.5 ...
##  $ LOR              : num  4.5 4.5 3.5 2.5 3 3 4 4 1.5 3 ...
##  $ CGPA             : num  9.65 8.87 8 8.67 8.21 9.34 8.2 7.9 8 8.6 ...
##  $ Research         : int  1 1 1 1 0 1 1 0 0 0 ...
##  $ Chance.of.Admit  : num  0.92 0.76 0.72 0.8 0.65 0.9 0.75 0.68 0.5 0.45 ...
## NULL
```

```
# Check for missing values
print("\nMissing Values in the Dataset:")
```

```
## [1] "\nMissing Values in the Dataset:"
```

```
print(colSums(is.na(data)))
```

```
##         GRE.Score       TOEFL.Score University.Rating               SOP
##                 0                 0                 0                 0
##              LOR              CGPA          Research   Chance.of.Admit
##                 0                 0                 0                 0
```

The dataset contains 400 observations and 8 variables: GRE.Score, TOEFL.Score, University.Rating, SOP, LOR, CGPA, Research, and Chance.of.Admit. There are no missing values across all columns as confirmed by colSums(is.na(data)), which returned zeros for each variable.

2. Visualize the Data

```
# Correlation matrix
correlation_matrix <- cor(data)
print("\nCorrelation Matrix:")
```

```
## [1] "\nCorrelation Matrix:"
```

```
print(correlation_matrix)
```

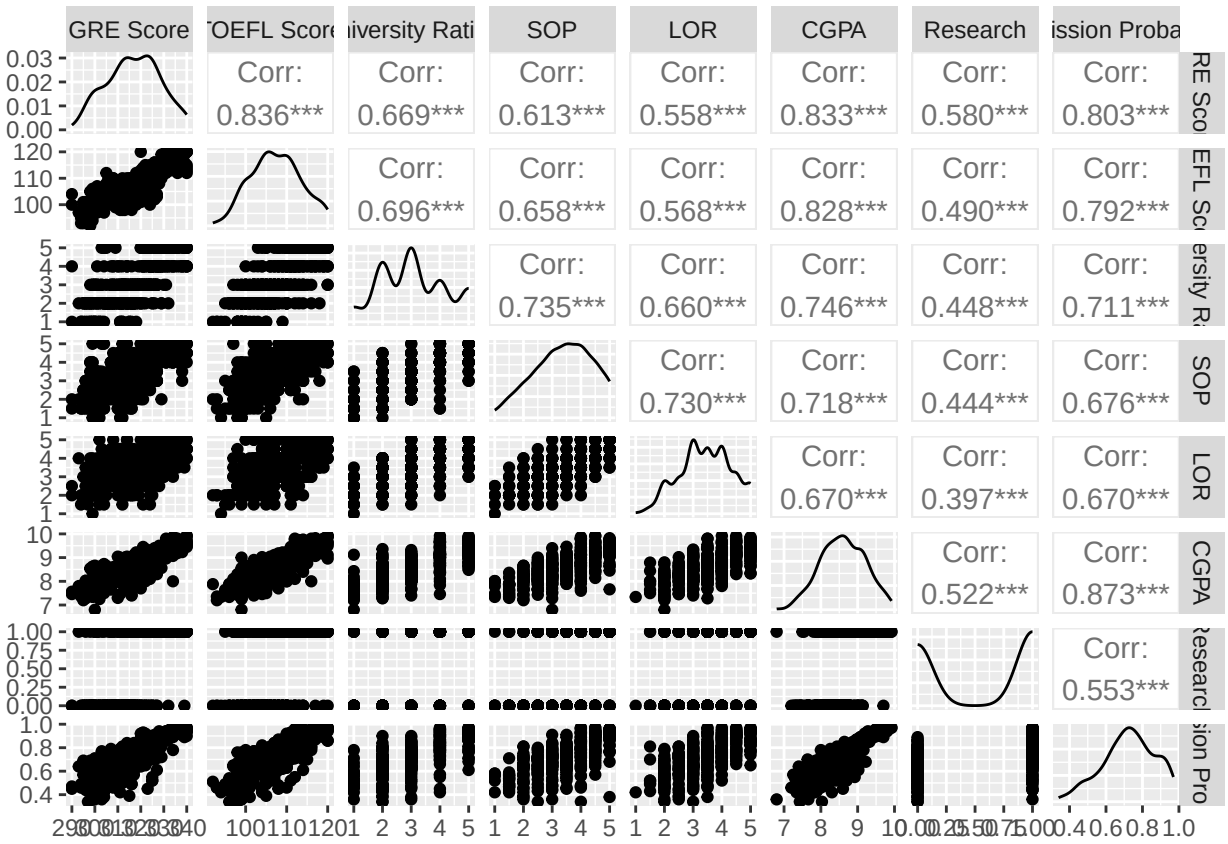
```
##                    GRE.Score TOEFL.Score University.Rating       SOP       LOR
## GRE.Score          1.0000000   0.8359768         0.6689759 0.6128307 0.5575545
## TOEFL.Score        0.8359768   1.0000000         0.6955898 0.6579805 0.5677209
## University.Rating  0.6689759   0.6955898         1.0000000 0.7345228 0.6601235
## SOP                0.6128307   0.6579805         0.7345228 1.0000000 0.7295925
## LOR                0.5575545   0.5677209         0.6601235 0.7295925 1.0000000
## CGPA               0.8330605   0.8284174         0.7464787 0.7181440 0.6702113
## Research           0.5803906   0.4898579         0.4477825 0.4440288 0.3968593
## Chance.of.Admit    0.8026105   0.7915940         0.7112503 0.6757319 0.6698888
##                         CGPA  Research Chance.of.Admit
## GRE.Score          0.8330605 0.5803906       0.8026105
## TOEFL.Score        0.8284174 0.4898579       0.7915940
## University.Rating  0.7464787 0.4477825       0.7112503
## SOP                0.7181440 0.4440288       0.6757319
## LOR                0.6702113 0.3968593       0.6698888
## CGPA               1.0000000 0.5216542       0.8732891
## Research           0.5216542 1.0000000       0.5532021
## Chance.of.Admit    0.8732891 0.5532021       1.0000000
```

```
# Custom column labels
custom_labels <- c("GRE Score", "TOEFL Score", "University Rating",
                   "SOP", "LOR", "CGPA", "Research", "Admission Probability")
colnames(data) <- custom_labels

# Scatterplot matrix
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(data[, sapply(data, is.numeric)],
        columnLabels = custom_labels,
        progress = FALSE)
```



These correlations can help in understanding the relationships between the features and their potential impact on the chance of admission. The correlation matrix shows that GRE.Score and TOEFL.Score are highly correlated (0.84), indicating a strong relationship between these two variables. Other pairs, such as CGPA and Chance.of.Admit (0.87), also exhibit notable correlations, suggesting that some predictors are related to each other in the dataset.

3. Split the Data into Training and Testing Sets

Cross-validation was used to assess the model's performance more reliably by splitting the data into 10 subsets (folds). In each iteration, the model was trained on 9 of the 10 folds and tested on the remaining fold. This process was repeated 10 times, each time using a different fold as the test set, ensuring that each data point was used for both training and testing. This helps mitigate issues like overfitting or underfitting, which could arise if the model were evaluated using a single training/test split.

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
set.seed(123)
index <- createDataPartition(data$`Admission Probability`, p = 0.8, list = FALSE)
train <- data[index, ]
test <- data[-index, ]
```

```r
# 4. Set up Cross-Validation
train_control <- trainControl(method = "cv", number = 10)  # 10-fold cross-validation
cv_model <- train(`Admission Probability` ~ .,
                  data = train,
                  method = "lm",
                  trControl = train_control)

# Print cross-validation results
print(cv_model)
```

```
## Linear Regression
##
## 322 samples
##   7 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 291, 290, 288, 291, 289, 290, ...
## Resampling results:
##
##   RMSE        Rsquared   MAE
##   0.06257349  0.8131451  0.04500347
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

5. Evaluate Model Performance on the Test Set

We evaluated the model on a separate test dataset (which was not used in the training or validation process) to simulate how the model would perform in real-world scenarios.

```r
predictions <- predict(cv_model, newdata = test)

# Mean Squared Error (MSE) on Test Set
mse <- mean((predictions - test$`Admission Probability`)^2)
cat("Mean Squared Error (MSE) on Test Set:", mse, "\n")
```

```
## Mean Squared Error (MSE) on Test Set: 0.004388214
```

```r
# Calculate RMSE on Test Set
rmse <- sqrt(mse)
cat("Root Mean Squared Error (RMSE) on Test Set:", rmse, "\n")
```

```
## Root Mean Squared Error (RMSE) on Test Set: 0.0662436
```

```r
# R-squared on Test Set
rsq <- 1 - sum((predictions - test$`Admission Probability`)^2) / sum((mean(test$`Admission Probability`)
cat("R-squared on Test Set:", rsq, "\n")
```

```
## R-squared on Test Set: 0.802751
```

The model shows good performance on the test set with an RMSE of 0.0662 and an R-squared of 0.8028, indicating that it explains over 80% of the variance in the data.

6. Checking for Multicollinearity (Variance Inflation Factor - VIF)

Multicollinearity refers to a situation in regression analysis where two or more predictor variables (independent variables) are highly correlated with each other. This can cause problems because the model struggles to distinguish between the individual effects of those correlated predictors on the dependent variable.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
vif_model <- lm(`Admission Probability`~ ., data = train)
vif(vif_model)
```

```
##         `GRE Score`        `TOEFL Score` `University Rating`                 SOP
##            4.820616            4.045572            2.846581            3.064486
##                LOR                CGPA            Research
##            2.368963            5.221852            1.640062
```

VIF values indicate the degree of multicollinearity between predictors. A VIF greater than 5-10 suggests that a variable is highly collinear with others in the model. Based on the values, none of the VIFs seem to be excessively high, indicating that multicollinearity is not a major concern in this model.