附录

绘图部分用R编写，建模部分用python编写。

绘图部分

```
rm(list = ls())
library(dplyr)
library(reshape2)
library(GGally)
# for color
library(ggpubr)
library(RColorBrewer)

df <- read.csv("Admission_Predict.csv") %>%
  select('GRE.Score':'Chance.of.Admit')

names(df)[1] <- "GRE"
names(df)[2] <- "TOEFL"
names(df)[3] <- "Univ.Rating"
names(df)[6] <- "GPA"
names(df)[8] <- "Chance"
df$GPA <- round(df$GPA / 10 * 4, 2)

head(df, 3)
summary(df)

# 相关系数图
corrdata <- cor(df) %>%
  round(2) %>%
  melt

ggplot(corrdata, aes(x=Var1, y=Var2, fill=value, label=value)) +
  geom_tile(colour="black") +
  geom_text(size=4, colour="white") +
  coord_equal()

# 矩阵散点图
lowerFn <- function(data, mapping, method = "loess", ...) {
  ggplot(data = data, mapping = mapping) +
    geom_point(size=1)+#colour = "blue") +
    geom_smooth(method = method, color = "red", ...)+
    theme(panel.background = element_rect(fill = "white", colour = "grey20"))
}

diagFn <- function(data, mapping, method = "loess", ...) {
  p <- ggplot(data = data, mapping = mapping) +
    geom_histogram(colour = "black",size=0.1) +
    theme(panel.background = element_rect(fill = "white", colour = "grey20"))
```

```
  p
}

ggpairs(df,
        lower = list(continuous = wrap(lowerFn, method = "lm")),
        diag = list(continuous = wrap(diagFn)),
        upper = NULL) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.border = element_rect( colour = "black", fill = NA,size=0.3),
        axis.title=element_text(size=8,face="plain",color="grey30"),
        axis.text = element_text(size=8,face="plain",color="grey30"),
        strip.background = element_blank())

# GPA vs. Research
ggplot(df, aes(factor(Research), GPA),)+
  geom_boxplot(aes(fill = factor(Research)),notch = FALSE)+
  guides(fill=guide_legend(title="Research"))+
  geom_jitter(binaxis = "Research", position = position_jitter(0.2),stackdir =
"center",dotsize = 0.1)+
  scale_fill_manual(values=c(brewer.pal(7,"Set2")[c(1,2,4,5)])) +
  xlab("Research")

# Univ.Rating vs. Research
ggplot(df, aes(x = factor(Univ.Rating), y = Chance))+
  geom_boxplot(outlier.size = 1, aes(fill=factor(Research)),
               position = position_dodge(0.8),size=0.5) +
  guides(fill=guide_legend(title="Research"))+
  theme_minimal()+
  theme(axis.title=element_text(size=13,face="plain",color="black"),
        axis.text = element_text(size=11,face="plain",color="black"),
        panel.background=element_rect(colour="black",fill=NA),
        panel.grid.minor=element_blank(),
        legend.position="right",
        legend.background=element_rect(colour=NA,fill=NA),
        axis.ticks=element_line(colour="black"))+
  xlab("Univ.Rating")

# 重要性条形图
importance.df <- data.frame(importance=c(0.0744, 0.037, 0.0144, 0.0266, 0.024, 0.8122,
0.0114),
                            features = names(df)[1:7])
mydata2 <- arrange(importance.df,desc(importance))
importance.df$features <- factor(importance.df$features, levels = mydata2$features)

ggplot(data=importance.df,aes(y=importance, x=features))+
  geom_bar(stat="identity", color="black", width=0.6,fill="#FC4E07",size=0.25) +#"#00AFBB"
  scale_fill_manual(values=brewer.pal(9,"GnBu")[c(7:2)])+
  coord_flip()+
  theme(
    axis.title=element_text(size=15,face="plain",color="black"),
    axis.text = element_text(size=12,face="plain",color="black"),
    legend.title=element_text(size=13,face="plain",color="black"),
    legend.position = "right"# c(0.83,0.15)
```

```
)
```

建模部分

```
import graphviz
import numpy as np
import pandas as pd
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LassoCV, LinearRegression, RidgeCV
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor, export_graphviz

np.set_printoptions(precision=4, suppress=True)

df = pd.read_csv('Admission_Predict.csv')
df.loc[:, 'CGPA'] = df.loc[:, 'CGPA']/10*4 # 转GPA为四分制

X = df.iloc[:, 1:-1]
y = df.iloc[:, -1]
feature_names = ["GRE", "TOEFL", "University" , "SOP", "LOR", "GPA", "Research"]

# random forest for generate Feature Importances
rf_cls = RandomForestRegressor(n_estimators=1000, n_jobs=-1).fit(X, y)
print("features importances:")
print(feature_names)
print(rf_cls.feature_importances_)

OLS_reg = LinearRegression().fit(X, y)
print(f"OLS coef: {OLS_reg.intercept_:.4f}, {OLS_reg.coef_}")
predict = OLS_reg.predict(X)
print(f"OLS MSE: {mean_squared_error(y, predict):.6f}")



# 测试各种方法的性能
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1, test_size=0.2)

# regression tree
tree_reg = DecisionTreeRegressor(max_depth=4).fit(X_train, y_train)
predict = tree_reg.predict(X_test)
print(f"tree MSE: {mean_squared_error(y_test, predict):.6f}")

# OLS
OLS_reg = LinearRegression().fit(X_train, y_train)
print(f"OLS coef: {OLS_reg.intercept_:.4f}, {OLS_reg.coef_}")

predict = OLS_reg.predict(X_test)
mse = mean_squared_error(y_test, predict)
print(f"MSE: {mse:.6f}")

# ridge
alphas_to_test = np.linspace(0.01, 1, 100)
Ridge_reg = RidgeCV(alphas=alphas_to_test, cv=10).fit(X_train, y_train)
```

```
print(f"Ridge coef: {Ridge_reg.intercept_:.4f}, {Ridge_reg.coef_}")
predict = Ridge_reg.predict(X_test)
print(f"MSE: {mean_squared_error(y_test, predict):.6f}")

# lasso
alphas_to_test = np.linspace(0.001, 0.01, 10)
LASSO_reg = LassoCV(alphas=alphas_to_test).fit(X_train, y_train)

print(f"LASSO coef: {LASSO_reg.intercept_:.4f}, {LASSO_reg.coef_}")
predict = LASSO_reg.predict(X_test)
print(f"MSE: {mean_squared_error(y_test, predict):.6f}")

# 导出graphviz dot文件用于决策树可视化
export_graphviz(tree_reg,
                out_file="tree.dot",
                feature_names=feature_names,
                impurity=False,
                special_characters=True,
                filled=True, rounded=True)
```