---
title: "R Notebook"
output: html_notebook

Reading dataset:
```{r}
Data <- read.csv("Daegu_Real_Estate_data.csv")
Data
```
Cleaning:
```{r}
dim(Data)
Data <- na.omit(Data)
dim(Data)
```

 column N_elevators to binary value IS_elevator:
```{r}
names(Data)[names(Data) == "N_elevators"] <- "IS_elevator"
Data$IS_elevator <- ifelse(Data$IS_elevator > 0, 1, 0)
```

Now we can use Is_elevator column to build logistic regression model:
```{r}
dir_logistic <- list()
dir_logistic$fit <- glm(IS_elevator ~ . - IS_elevator,
                    family = binomial, data = Data)
summary(dir_logistic$fit)
```

Z racji na niezbierzność modelu podczas budowy modelu ze wszystkimi zmiennymi, proces selekcji został przeprowadzony ręcznie zgodnie z przypuszczeniami autora co do przydatnosci i istotności danych zmiennych w modelu.
```{r}
dir_logistic <- list()
dir_logistic$fit <- glm(IS_elevator ~ YearBuilt + SalePrice + Floor + N_APT +
N_SchoolNearBy.Total. + N_FacilitiesNearBy.Total. + N_Parkinglot.Ground. +
N_Parkinglot.Basement.,
                    family = binomial, data = Data)
summary(dir_logistic$fit)
```

Model został zbudowany po 11 iteracjach.

Zmienna YearBuilt jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest ujemny a więc zmiejsza prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna SalePrice jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna Floor jest nie istotna - można ją wykluczyć z modelu.

Zmienna N_APT jest istona. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest ujemny a więc zmiejsza prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_SchoolNearBy.Total. jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodani a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_FacilitiesNearBy.Total. jest istotna. Niska wartość satystyki p daje nam

wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest ujemny a więc zmiejsza prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_Parkinglot.Ground. jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_Parkinglot.Basement. jest istotna.Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Analiza dewiancji modelu pokazuje, że model znacząco różni się od modelu zerowego co jest porządanym rezultatem. Pdobne wnioski można wyciagnąć z porówania wyniku kryterium AIC z modelem zerowym.

Model bez zmiennej Floor:

```{r}
dir_logistic <- list()
dir_logistic$fit <- glm(IS_elevator ~ YearBuilt + SalePrice + N_APT +
N_SchoolNearBy.Total. + N_FacilitiesNearBy.Total. + N_Parkinglot.Ground. +
N_Parkinglot.Basement.,
                  family = binomial, data = Data)
summary(dir_logistic$fit)
```

Po usunięciu zmiennej Krytermiu AIC oraz dewiancja podobna choć nieznacznie większa.

Dodanie zmiennych do modelu:

```{r}
dir_logistic <- list()
dir_logistic$fit <- glm(IS_elevator ~ YearBuilt + SalePrice + N_APT +
N_SchoolNearBy.Total. + N_FacilitiesNearBy.Total. + N_Parkinglot.Ground. +
N_Parkinglot.Basement. + N_manager + AptManageType,
                  family = binomial, data = Data)
summary(dir_logistic$fit)
```

Końcowo metodą prób i błędów udało się uzyskać model lepszy od poprzeniego (niższa wartość kryterium AIC oraz niższa wartość dewiancji)

Analiza wyników:

Zmienna YearBuilt jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest ujemny a więc zmiejsza prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna SalePrice jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_APT jest istona. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest ujemny a więc zmiejsza prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_SchoolNearBy.Total. jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodani a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_FacilitiesNearBy.Total. jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_Parkinglot.Ground. jest istotna. Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_Parkinglot.Basement. jest istotna.Niska wartość satystyki p daje nam wysoką

grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna N_manager jest istotna.Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest ujemny a więc zmniejsza prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Zmienna AptManageTypeself_management jest istotna.Niska wartość satystyki p daje nam wysoką grarancje, że współczynik jest różny od zera. Współczynnik jest dodatni a więc zwiększa prawdopodobieństwo istnienia windy w budynku w którym znajduje się mieszkanie.

Modele addytywne GAM:

```{r}
library(gam)
```

```{r}
fit_gam_bf <- gam(IS_elevator ~ s(YearBuilt, df = 5) + s(SalePrice, df = 5) + s(N_APT, df = 5) + s(N_SchoolNearBy.Total., df = 5) +
    s(N_FacilitiesNearBy.Total., df = 5) + s(N_Parkinglot.Ground., df = 5) + s(N_Parkinglot.Basement., df = 5) +
    s(N_manager, df = 5) + AptManageType, family = binomial(), data = Data)
summary(fit_gam_bf)
```

Znacznie lepsze wyniki od standarodwego modelu regresji logistycznej Dewiancja równa 0 i kryterium informayjnce AIC 83

```{r}
par(mfrow = c(1, 3))
plot(fit_gam_bf, col = "red", se = TRUE)
```

```{r}
library(tree)
```

Zamiana na dane kategoryczne:

```{r}
Data$IS_elevator <- factor(Data$IS_elevator, levels = c(0, 1), labels = c("No", "Yes"))
```

```{r}
categorical_vars <- c("HallwayType", "HeatingType", "AptManageType", "SubwayStation","TimeToBusStop","TimeToSubway")

dummy_data <- model.matrix(~., data = Data[, categorical_vars])

preprocessed_data <- cbind(Data[, -which(names(Data) %in% categorical_vars)], dummy_data)

print(preprocessed_data)
```

formuła

```{r}
add_backticks = function(x) {
    paste0("`", x, "`")
}

x_lm_formula = function(x) {
    paste(add_backticks(x), collapse = " + ")
}

build_lm_formula = function(x, y){
```

```
    if (length(y)>1){
        stop("y jest różne od 1")
    }
    as.formula(
        paste0("`",y,"`", " ~ ", x_lm_formula(x))
    )
}


get_lm_formula <- function(data) {
  columns <- colnames(data)

  y_cols <- columns[1]
  x_cols <- columns[2:length(columns)]

  formula <- build_lm_formula(x_cols, y_cols)

  return(formula)
}



formula <- get_lm_formula(preprocessed_data)
formula

```

```{r}
# Split the data into training and testing sets
set.seed(123)
train_index <- sample(nrow(preprocessed_data), 0.7 * nrow(preprocessed_data))
train_data <- preprocessed_data[train_index, ]
test_data <- preprocessed_data[-train_index, ]
```

Drzewo decyzyjne:
```{r}
IS_elevator_tree <- tree(IS_elevator ~ YearBuilt + YrSold + MonthSold + Size.sqf. +
Floor
                      +N_Parkinglot.Ground. + N_Parkinglot.Basement. + N_APT+ N_manager
+ N_FacilitiesNearBy.PublicOffice. + N_FacilitiesNearBy.Hospital. +
N_FacilitiesNearBy.Dpartmentstore. + N_FacilitiesNearBy.Mall. + N_FacilitiesNearBy.ETC.
+ N_FacilitiesNearBy.Park. + N_SchoolNearBy.Elementary. + N_SchoolNearBy.Middle. +
N_SchoolNearBy.High. + N_SchoolNearBy.University. + N_FacilitiesInApt +
N_FacilitiesNearBy.Total. + N_SchoolNearBy.Total. + HallwayTypemixed +
HallwayTypeterraced + HeatingTypeindividual_heating + AptManageTypeself_management +
SubwayStationBanwoldang + SubwayStationDaegu + SubwayStationKyungbuk_uni_hospital +
SubwayStationno_subway_nearby, data = train_data)
summary(IS_elevator_tree)
```



```{r}
plot(IS_elevator_tree)
text(IS_elevator_tree, pretty = 0)
```
```{r}
print(IS_elevator_tree)
```



```{r}
library(tree)
predictions <- predict(IS_elevator_tree, test_data, type = "class")
confusionMatrix(predictions, test_data$IS_elevator)
```
Uzyskany model drzewa decyzyjnego użył do budowy drzewa tylko następujące zmienne:"N_SchoolNearBy.Middle.", "N_manager", N_Parkinglot.Ground.". I są to zmienne które są najbardziej istotne w klasyfikacji tego czy winda istnieje w budynku.

Uzyskane wyniki wskazują na idelane dopasowanie do danych, i bezbłędne rozpoznanie wszystkich klas w zbiorze testowym. Wynik ten jednak wydaje się być niemożliwym do uzyskania w rzeczywistiści. Upatrujemy przyczyne w zbiorze danych i być może zbyt małej ilości rekorów danych, rónież nie równym podziale w kategorii jest/nie ma windy.

Las losowy:

```{r}
install.packages("randomForest") # Install the package
library(randomForest) # Load the package
```

```{r}
# Train the random forest model
rf_model <- randomForest(IS_elevator ~  YearBuilt + YrSold + MonthSold + Size.sqf. + Floor
                         +N_Parkinglot.Ground. + N_Parkinglot.Basement. + N_APT+ N_manager + N_FacilitiesNearBy.PublicOffice. + N_FacilitiesNearBy.Hospital. + N_FacilitiesNearBy.Dpartmentstore. + N_FacilitiesNearBy.Mall. + N_FacilitiesNearBy.ETC. + N_FacilitiesNearBy.Park. + N_SchoolNearBy.Elementary. + N_SchoolNearBy.Middle. + N_SchoolNearBy.High. + N_SchoolNearBy.University. + N_FacilitiesInApt + N_FacilitiesNearBy.Total. + N_SchoolNearBy.Total. + HallwayTypemixed + HallwayTypeterraced + HeatingTypeindividual_heating + AptManageTypeself_management + SubwayStationBanwoldang + SubwayStationDaegu + SubwayStationKyungbuk_uni_hospital + SubwayStationno_subway_nearby, data = train_data, ntree = 50, mtry = 6)
```

```{r}
# Plot the error vs the number of trees graph
plot(rf_model)
```

```{r}
predictions <- predict(rf_model, test_data, type = "class")

# Ocenianie dopasowania modelu
confusion_matrix <- confusionMatrix(predictions, test_data$IS_elevator)
accuracy <- confusion_matrix$overall['Accuracy']
precision <- confusion_matrix$byClass['Precision']
recall <- confusion_matrix$byClass['Recall']
specificity <- confusion_matrix$byClass['Specificity']
f1_score <- confusion_matrix$byClass['F1']

# Wyświetlenie wyników
print(confusion_matrix)
cat(paste0("Accuracy: ", accuracy, "\n"))
cat(paste0("Precision: ", precision, "\n"))
cat(paste0("Recall: ", recall, "\n"))
cat(paste0("Specificity: ", specificity, "\n"))
cat(paste0("F1 Score: ", f1_score, "\n"))
```

Wyniki lasu losowego są takie podobne do pojedynczego drzewa. Idelane dopasowanie do danych, poprawne przewidzenie wszystkich przykładów. Jak zostało wspomniane wcześniej, wynik taki wydaje się być zbyt idelany, uaptrujemy winy w danych, (zbyt mała ilość danych, zbyt duża dysproporcja w klasie (ilość) yes/no)