**Görev 1 (Kolay): "mtcars" Veri Setinde Basit Rastgele Bölme**

Veri setini %60 eğitim ve %40 test olarak böleceğiz. Eğitim ve test setlerindeki **"mpg"** (miles per gallon) ortalamalarının benzer olduğunu göstereceğiz.

```
# "mtcars" Veri Setinde Basit Rastgele Bölme
# Gerekli paket
install.packages("rsample")
library(rsample)

# Reproducible sonuçlar için
set.seed(123)

# mtcars veri seti
data("mtcars")

# Veriyi %60 eğitim ve %40 test olarak bölme
split <- initial_split(mtcars, prop = 0.6)
train_data <- training(split)
test_data <- testing(split)

# Eğitim ve test setlerindeki "mpg" ortalamaları
mean_train <- mean(train_data$mpg)
mean_test <- mean(test_data$mpg)

# Sonuçları raporlama
cat("Eğitim setindeki MPG ortalaması:", mean_train, "\n")
cat("Test setindeki MPG ortalaması:", mean_test, "\n")
```

Kodda, **initial_split()** fonksiyonu ile veriyi böldük. **mean()** fonksiyonu ile ortalamaları hesapladık.
Ortalama değerler birbirine yakınsa, bölme işleminin uygun olduğunu söyleyebiliriz.

**Görev 2 (Normal): "PimaIndiansDiabetes" Veri Setinde Stratified Sampling**

Veri setini **%70 eğitim ve %30 test** olarak stratified sampling yöntemiyle böleceğiz. **"diabetes"** değişkeninin dağılımının eğitim ve test setlerinde orijinal veriyle aynı olduğunu kontrol edeceğiz.

```
# "PimaIndiansDiabetes" Veri Setinde Stratified Sampling
# Gerekli paketler
install.packages("mlbench")
install.packages("rsample")
```

```
library(mlbench)
library(rsample)

# PimaIndiansDiabetes veri setini yükle
data("PimaIndiansDiabetes")

# Veriyi inceleyelim
str(PimaIndiansDiabetes)

# Reproducible sonuçlar için
set.seed(123)

# Stratified sampling ile veri bölme (%70 eğitim, %30 test)
split <- initial_split(PimaIndiansDiabetes, prop = 0.7, strata = "diabetes")
train_data <- training(split)
test_data <- testing(split)

# Orijinal veri setindeki diabetes dağılımı
orig_dist <- prop.table(table(PimaIndiansDiabetes$diabetes))

# Eğitim setindeki diabetes dağılımı
train_dist <- prop.table(table(train_data$diabetes))

# Test setindeki diabetes dağılımı
test_dist <- prop.table(table(test_data$diabetes))

# Sonuçları raporlama
cat("Orijinal Veri Seti Dağılımı:\n")
print(orig_dist)
cat("\nEğitim Seti Dağılımı:\n")
print(train_dist)
cat("\nTest Seti Dağılımı:\n")
print(test_dist)
```

**initial_split()** fonksiyonuna **strata = "diabetes"** ekleyerek **stratified sampling** gerçekleştirdik.
Sonuçları kontrol ederek, her iki setteki **"diabetes"** değişkeninin dağılımının orijinal veri setine benzediğini görmeliyiz.

**Görev 3 (Zor): "diamonds" Veri Setinde Cross Validation ve RMSE Hesabı**

**"diamonds"** veri setinde fiyat tahmini için **5 katlı cross validation** uygulayacağız.

Her kat için **Lineer Regresyon Modeli** eğitilecek ve **RMSE** hesaplanacak. Son olarak, **ortalama RMSE** raporlanacak.

```
# "diamonds" Veri Setinde Cross Validation ve RMSE Hesabı
# Gerekli paketler
install.packages("ggplot2")
install.packages("rsample")
install.packages("yardstick")
install.packages("dplyr")
install.packages("parsnip")

library(ggplot2)
library(rsample)
library(yardstick)
library(dplyr)
library(parsnip)

# diamonds veri setini yükle
data("diamonds")

# Reproducible sonuçlar için
set.seed(123)

# 5 katlı cross validation ayarı
cv_folds <- vfold_cv(diamonds, v = 5)

# Her kat için RMSE değerlerini tutacak bir vektör
rmse_values <- c()

# Cross validation döngüsü
for (i in seq_along(cv_folds$splits)) {
  # Eğitim ve test verilerini ayırma
  train_data <- training(cv_folds$splits[[i]])
  test_data <- testing(cv_folds$splits[[i]])

  # Lineer regresyon modeli oluşturma
  model <- linear_reg() %>%
    set_engine("lm") %>%
    fit(price ~ carat + depth + table + x + y + z, data = train_data)

  # Test setinde tahmin yapma
  predictions <- predict(model, test_data)$.pred

  # Test setindeki gerçek değerleri çıkarma
```

```
  actuals <- test_data$price

  # RMSE hesaplama
  rmse_value <- rmse_vec(actuals, predictions)

  # RMSE değerini saklama
  rmse_values <- c(rmse_values, rmse_value)
}

# Ortalama RMSE hesaplama
average_rmse <- mean(rmse_values)

# Sonuçları raporlama
cat("Her Katın RMSE Değerleri:\n")
print(rmse_values)
cat("\nOrtalama RMSE Değeri:", average_rmse, "\n")
```

**vfold_cv()** fonksiyonu ile 5 katlı cross validation yaptık.
Her kat için **linear_reg()** fonksiyonu ile lineer regresyon modeli eğittik.
**rmse_vec()** fonksiyonunu kullanarak her katın RMSE değerlerini hesapladık.
Ortalama RMSE'yi raporladık.