Making predictions with model and pipeline

Group F

2024-10-30

Load the saved model and preprocessing pipeline

```
# Load the saved model and preprocessing pipeline
randomForest_model <- readRDS("random_forest_model.rds")
preprocessing_pipeline <- readRDS("preprocessing_pipeline.rds")
```

Import frequency encoding attributes

```
# Import frequency encoding attributes
title_frequency <- read.csv("title_frequency.csv")
department_frequency <- read.csv("department_frequency.csv")
country_frequency <- read.csv("country_frequency.csv")
```

Sample data used to make predictions with

```
# Define a sample single record as a data frame
# Use one of these to make predictions with different value

# This record should output "Eligible"
new_record <- data.frame(
  Title = "CORRECTIONAL OFFICER",
  Department.Name = "CORRECTIONS & REHABILITATION",
  Annual.Salary = 54620,
  Gross.Pay.Last.Paycheck = 2502,
  Gross.Year.To.Date = 48025,
  Gross.Year.To.Date...FRS.Contribution = 46617,
  year_of_birth = 1976,
  marital_status = "married",
  Country_id = "52770",
  Education = "Masters",
  Occupation = "Prof.",
  household_size = 2,
  yrs_residence = 4
)
```

```r
#This record should output "Non Eligible"
# new_record <- data.frame(
#   Title = "CUSTODIAL WORKER 2",
#   Department.Name = "PARKS, RECREATION AND OPEN SPACES",
#   Annual.Salary = 18439.46,
#   Gross.Pay.Last.Paycheck = 1574.1,
#   Gross.Year.To.Date = 22011.04,
#   Gross.Year.To.Date...FRS.Contribution = 21378.11,
#   year_of_birth = 1951,
#   marital_status = "married",
#   Country_id = "52789",
#   Education = "Masters",
#   Occupation = "Prof.",
#   household_size = 2,
#   yrs_residence = 4
# )
```

Cleaning and preprocessing

```r
# Feature Engineering: Calculate 'Age' and add it to new_record
new_record <- new_record %>%
  mutate(Age = as.integer(year(today()) - year_of_birth))

new_record$Eligible <- 1

# Encoding (frequency encoding using pre-calculated tables)
new_record$Frequency_Title <- ifelse(new_record$Title %in%
                                 title_frequency$Title,
                                 title_frequency[new_record$Title], 0)
new_record$Frequency_Department <-
  ifelse(new_record$Department.Name %in%
           names(department_frequency),
           department_frequency[new_record$Department.Name], 0)
new_record$Frequency_Country_ID <-
  ifelse(new_record$Country_id %in%
           names(country_frequency),
           country_frequency[new_record$Country_id], 0)

# One-hot encode marital status
new_record$Marital_Status_married <-
  ifelse(new_record$marital_status == "married", 1, 0)
new_record$Marital_Status_single <-
  ifelse(new_record$marital_status == "single", 1, 0)
new_record$Marital_Status_divorced <-
  ifelse(new_record$marital_status == "divorced", 1, 0)
new_record$Marital_Status_widowed <-
  ifelse(new_record$marital_status == "widowed", 1, 0)

new_record$Education_Bach <-
  ifelse(new_record$Education == "Bach.", 1, 0)
new_record$Education_Masters <-
```

```
  ifelse(new_record$Education == "Masters", 1, 0)
new_record$Education_HS <-
  ifelse(new_record$Education == "HS-grad", 1, 0)

new_record$Occupation_Cleric <-
  ifelse(new_record$Occupation == "Cleric.", 1, 0)
new_record$Occupation_Prof <-
  ifelse(new_record$Occupation == "Prof.", 1, 0)
new_record$Occupation_Exec <-
  ifelse(new_record$Occupation == "Exec.", 1, 0)
new_record$Occupation_Sales <-
  ifelse(new_record$Occupation == "Sales", 1, 0)

# Set minimum value threshold for Box-Cox compatibility
new_record <- new_record %>%
  mutate(across(where(is.numeric), ~ ifelse(. <= 0, 0.001, .)))

# Ensure new_record has all columns required by the preprocessing pipeline
required_columns <- names(preprocessing_pipeline$mean)
new_record <- new_record %>% select(all_of(required_columns))

# Convert columns to numeric if necessary
new_record <- new_record %>% mutate(across(everything(), as.numeric))
```

Apply the preprocessing pipeline

```
# Apply the preprocessing pipeline
processed_new_record <- predict(preprocessing_pipeline, newdata = new_record)
```

Make a prediction using the random forest model

```
# Make a prediction using the random forest model
prediction <- predict(randomForest_model, newdata = processed_new_record)

predVal <- ifelse(prediction == "0.743001202264081",
                  "Eligible", "Not Eligible")
# Print the predicted value
print(prediction)
```

```
##                 1
## 0.743001202264081
## Levels: -1.34588580748694 0.743001202264081
```

```
print(predVal)
```

```
## [1] "Eligible"
```