Faculty of Engineering and Technology

Department of Computer Science

COMP338 - ARTIFICIAL INTELLIGENCE

Project #2 - ML

Obada Tahayna - 1191319

Prof. Mostafa Jarrar

January 24, 2023

Abstract

This project aimed to predict the type of dwelling unit a Palestinian family lives in, based on various features in a dataset collected from the Palestinian Expenditure and Consumption Survey in 2011 and 2014. The dataset was preprocessed to standardize missing values, unify currency, and remove irrelevant features. A total of 20 features were carefully selected through the use of Extra Trees Classifier and Decision Tree algorithm to determine feature importance. A Decision Tree model was built using these selected features and the accuracy was evaluated using both training and testing data. The end goal of this project was to accurately predict the type of dwelling unit a Palestinian family lives in and help to understand the factors that influence the type of housing they live in.

Table of Contents

Selected Features

Our dataset contains a variety of features that are divided into the following categories:

1. Sample type and identification information
2. Household and locality information
3. Housing and tenure information
4. Income and expenditure information
5. Consumption and purchasing habits
6. Agricultural and livestock information
7. Access to education and healthcare
8. Impact of the 2014 aggression on the household
9. Hosting of other families or individuals.

After carefully analyzing these features, The following features have been selected as the target and training features:

Target Feature

The chosen target feature is **H1 (What kind of dwelling unit does the family live in?)**, which is a categorical feature with the following possible values:

1. Villa
2. House
3. Apartment
4. Independent room
5. Tent
6. Barracks
7. Caravan
8. Shelter/ public house
9. Other

Training Features

Nineteen features were carefully selected for use in building the model. These features were chosen based on their potential impact on the target feature and through a series of steps as described below.

1. **H2: What is type of tenure?** (Full ownership, Mortgage/ loaned residence, Rented without furniture, Rented with furniture, For free, In exchange of labor, Other).
2. **H3_1: What is the value of monthly rent.**
3. **H4_1: What would be the estimated rent for this current month.**
4. **H6: Main building material for external walls?** (Clean stones, Stone and cement, Old stone, Bricks/ cement, Concrete, Other).
5. **H6_1: Main building material for the roof?** (Cement/ concrete, Metal, Wood, Asbestos/ cement fiber, Hay/ palm leaves, Other).

6. **H13_2: Is the service provided by public networks disrupted - Electricity** (Service continuously available, Service disrupted once a week, Service disrupted more than once a week, Service disrupted on daily basis (fixed hours of service a day), Network not operating, Other).
7. **H18_2: What is the main source of energy used in heating** (Gas, Kerosene, Electricity, Wood, Diesel, Coal, Other, None).
8. **H22_1: How many Private car are available to the house hold? (if non, write 0).**
9. **Region** (West Bank, Gaza).
10. **ID1: Province.**
11. **ID2: Residential area (التجمع السكاني)**
12. **Local Type** (Urban, Rural, Camp).
13. **C14_2: In the past 30 days and when the family did not have sufficient food or money to purchase food, how many times have you - Picked wild plants** (1 Never - 5 Always).
14. **C14_5: In the past 30 days and when the family did not have sufficient food or money to purchase food, how many times have you - Adults eaten smaller portion** (1 Never - 5 Always).
15. **C14_6: In the past 30 days and when the family did not have sufficient food or money to purchase food, how many times have you - Eaten stored food Purchased** (1 Never - 5 Always).
16. **C14_7: In the past 30 days (make sure you repeat the following phrase in asking all of the questions related to coping strategies) and when the family did not have sufficient food or money to purchase food, how many times have you - Picked wild plants** (1 Never - 5 Always).
17. **I03_1: After the aggression, has there been any change to the family income?** (Yes, It increased, Yes, decreased, No, remained unchanged, I don't know/ no answer).
18. **I06_01: What is the main source of household income** (Agriculture/ animal breeding/ fishing, Family business (other than agriculture), Government wage and salary, Private sector wage and salary, Wages from Israeli labor sectors, Transfers from Palestinian Territory (including pension), Transfers from abroad, International organizations (assistance), Social assistance, Salary from international organizations, National insurance (Jerusalem), Property Income, Other).
19. **T2_2A: During the second half 2014, have you or any of your family members faced difficulties that impeded your access to Farming land/ harvest** (No difficulty, Minor difficult, Big difficulty, NA, I don't know).

Work

Data Preprocessing

The preprocessing stage of this project was crucial in ensuring that the data was cleaned, ready for analysis, and only included the most relevant information. And it is a set of ordered steps that do the purpose

```
# Load the tidyverse and haven packages
library("tidyverse")
library("haven")
library("Hmisc")

# Set the working directory
setwd("D:\\Project2-ML")

# Use the functions to clean and prepare the data
data <-
read_and_convert_data_to_factor("dataset\\SefSec_2014_HH_weight
new.sav")
data <- replace_na_in_id1_and_localtype(data)
data <- convert_currencies(data)
data <- merge_h3_and_h4(data)
data <- convert_h3_h4_merged_to_categorical(data)
data <- keep_only_certain_columns(data)
data <- remove_text_columns(data)
data <- drop_na_columns_and_rows(data)
```

NULL Standardization

The dataset contained a variety of different representations for missing values, including NULL, NA, null, and empty entries. It was crucial to standardize these representations to ensure accurate and consistent analysis.

```
# Define a function to replace NULL, NA, and empty values with NA
replace_null_na_empty <- function(data) {
    data[grep("NULL", tolower(data), value = TRUE)] <- NA
    data[grep("NA", tolower(data), value = TRUE)] <- NA
    data[data == ""] <- NA
    return(data)
}
```

Features Merging

During the preprocessing stage, it was identified that the features 'ID1' and 'رمزالمحافظة' in the dataset referred to the same information, which is the number that indicates the county. To handle missing data, missing values in 'ID1' were replaced with the corresponding value in the 'رمزالمحافظة' column. Similarly, missing values in the 'Localtype' column were replaced with the corresponding value in the 'localitytype' column. This helped to ensure that the data was consistent and accurate.

```
# Define a function to replace NA or empty values in ID1 with the
value in the column "رمزالمحافظة"
replace_na_in_id1 <- function(data) {
  data$ID1 <- ifelse(is.na(data$ID1), data$`رمزالمحافظة`, data$ID1)
  return(data)
}

# Define a function to replace NA values in Localtype with the value
in the column "localitytype"
replace_na_in_localtype <- function(data) {
  data$Localtype <- ifelse(is.na(data$Localtype), data$localitytype,
data$Localtype)
  return(data)
}

# Define a function to replace NA or empty values in ID1 and
Localtype
replace_na_in_id1_and_localtype <- function(data) {
  data <- replace_na_in_id1(data)
  data <- replace_na_in_localtype(data)
  return(data)
}
```

In the dataset, the features 'H3_1'[1] and 'H4_1'[2] were found to not be together, either one of them only, 'H3_1' represents the value of the rent and 'H4_1' represents the estimated value for the current month; as a result, the two columns merged together.

```
# Define a function to create a new column "H3_H4_merged" that takes
the value of H3_1 if it is not NA, otherwise it takes the value of
H4_1
merge_h3_and_h4 <- function(data) {
  data$H3_H4_merged <- ifelse(is.na(data$H3_1), data$H4_1, data$H3_1)
  return(data)
}
```

[1] What is the value of monthly rent
[2] What would be the estimated rent for this current month

Currency unification

In order to ensure that our model would work effectively, it was necessary to unify the currency in the dataset. The columns 'H3_1' and 'H4_1' in the dataset, which represent the monthly rent, were accompanied by columns 'H3_2'[3] and 'H4_2'[4] respectively, which indicate the currency used (NIS, JOD, or USD). To unify the currency, the values in columns 'H3_1' and 'H4_1' were adjusted according to the currency indicated in columns 'H3_2' and 'H4_2' respectively. This helped to ensure that all values were consistent and in the same currency.

```
# Define a function to convert currencies to NIS
convert_currencies <- function(data) {
  JOD_ILS_in2014 <- 5.05313
  USD_ILS_in2014 <- 3.57763

  data$H3_1 <- ifelse(data$H3_2 == 2, data$H3_1*JOD_ILS_in2014,
                      ifelse(data$H3_2 == 3,
data$H3_1*USD_ILS_in2014,
                             data$H3_1))

  data$H4_1 <- ifelse(data$H4_2 == 2, data$H4_1*JOD_ILS_in2014,
                      ifelse(data$H4_2 == 3,
data$H4_1*USD_ILS_in2014,
                             data$H4_1))
  return(data)
}
```

Note that the survey was conducted in 2014, and the exchange rates used to unify the currency were the average rates in that year.[5]

Features Removal

During the preprocessing stage, several features that were not relevant to the analysis were removed. The following three functions were used for this purpose:

Remove Non-Certain Features

During this step, certain features were removed as they were deemed unnecessary for the analysis. Specifically, the features 'H3_1', 'H3_2', 'H4_1', and 'H4_2' were removed as they were already merged into a single feature. Additionally, the first 32 features were removed as they were found to have no impact on the target value and included information such as names, phone numbers, and ID numbers.

---

[3] Specify currency (for H3_1)
[4] Specify currency (for H4_1)
[5] https://exchangerates.org/jod/ils/in-2014 & https://exchangerates.org/usd/ils/in-2014

```
# Define a function to keep only certain columns
keep_only_certain_columns <- function(data) {
    data <- subset(data, select = -c(H3_1, H3_2, H4_1, H4_2))    #as
we have merged H3 and H4, so we can remove the original columns
    data <- subset(data, select = c("ID1", "ID2", "Region",
"Localtype", names(data)[-(1:32)]))
    return(data)
}
```

Remove Text Features

During this step, certain features were removed from the dataset as they were found to be irrelevant to the analysis. These features were identified as columns containing text data that are not numerical or categorical.

```
# Define a function to remove the columns that have text (not
categorical nor numerical data)
remove_text_columns <- function(data) {
  text_columns <- grep("text", names(data), ignore.case = TRUE)
  data <- data[, -text_columns]
  return(data)
}
```

Remove NA Features and Rows

In this step, only features and rows that contain all missing values were removed from the dataset.

```
# Define a function to remove the columns that have all values as
NA
drop_na_columns_and_rows <- function(data) {
  data <- data[,colSums(is.na(data))<nrow(data)]
  data <- data[rowSums(is.na(data)) != ncol(data), ]
  return(data)
}
```

Categorizing Numeric Data

The data in the H3 and H4 columns are numeric, which cannot be used directly in building a decision tree. To address this, the numeric data was converted into categorical intervals to be used in building the model.

```
# define a function to convert H3_H4_merged to categorical data
(intervals)
convert_h3_h4_merged_to_categorical <- function(data) {
  data$H3_H4_merged <- as.numeric(data$H3_H4_merged)
  data$H3_H4_merged <- cut2(data$H3_H4_merged, g = 10)
  data <- remove_punctuations(data)
  return(data)
}
```

Features Selection

With 840 features in the dataset, using all of them to build the decision tree is not an efficient approach. This would lead to a slow building process and slow predictions, as well as potentially yielding a high accuracy for the training data but poor accuracy for testing data, resulting in a poor model overall.

Therefore, 19 features were carefully chosen through a series of steps.

Extra Trees Classifier

The first step in selecting the features was to use the Extra Trees Classifier algorithm[6]. This algorithm is an ensemble method that builds multiple decision trees and outputs the feature importance of each feature. The features with the highest importance were then selected for further analysis.

```
from sklearn.ensemble import ExtraTreesClassifier
import os
import pandas as pd

# Set the working directory
os.chdir("D:\\Project2-ML")

# Read the .sav file and store it in the df dataframe
df = pd.read_spss("PreprocessedData.sav")

# Drop the rows with missing values (as the ExtraTreesClassifier
cannot handle missing values)
df = df.dropna()

# Define target and predictors
X = df.drop(columns=["H1"])
y = df["H1"]

# Convert the categorical variables to dummy variables
X = pd.get_dummies(X)

# fit the Extra Trees Classifier
clf = ExtraTreesClassifier()
clf = clf.fit(X, y)
# Get the feature importances
importances = clf.feature_importances_
# Get the feature names
feature_names = X.columns
```

[6] https://towardsdatascience.com/what-when-how-extratrees-classifier-c939f905851c

```
# Zip the feature importances and names together
importances_with_names = list(zip(importances, feature_names))

# Sort the features by importance
importances_with_names = sorted(
    importances_with_names, key=lambda x: x[0], reverse=True)
```

Then to print the features:

```
# Create an empty list to store the column names that match the words
in the file
matched_columns = []

# Read the original .sav file to get the column names
df_original = pd.read_spss("dataset\\SefSec_2014_HH_weight new.sav")
columns_names = df_original.columns.tolist()

index = 1   # index to print the features

for importance, feature_name in importances_with_names:
    if feature_name in columns_names and feature_name not in
matched_columns:
        matched_columns.append(feature_name)
        print(f"[{index}]{feature_name}: {importance}")
        index += 1

    else:
        last_underscore_index = feature_name.rindex("_")
        feature_name = feature_name[:last_underscore_index]
        if feature_name in columns_names and feature_name not in
matched_columns:
            matched_columns.append(feature_name)
            print(f"[{index}]{feature_name}: {importance}")
            index += 1
```

The code above will output a list of all features sorted by their importance as determined by the Extra Trees Classifier algorithm. The first 40 features, along with their importance values, are as follows:

[1] H12_3B: 0.007568536475346147
[2] H6_1: 0.004613781664578499
[3] H12_1A: 0.004535563206028545
[4] T3_4_1: 0.0044241753744211874
[5] H12_3A: 0.0036787875191988074

[6] H6: 0.003285466696073471
[7] C14_10: 0.003079190740988562
[8] H12_2A: 0.003059495841105018
[9] C14_12: 0.0028703582722227183
[10] BU227_B: 0.002824883277358046
[11] T3_3_2: 0.0028136867904518135
[12] H13_3A: 0.002782840732274329
[13] Region: 0.002549935874791556
[14] C14_7: 0.0024837085587592934
[15] C14_9_WB: 0.0024027828795652
[16] H13_1_1_A: 0.002187292340094045
[17] ID2: 0.002186396036493239
[18] H18_2: 0.0020833040800279394
[19] C14_5_WB: 0.0019923082823744757
[20] H13_1_2_A: 0.0019797837470506402
[21] H22_5: 0.0019739414148923144
[22] C14_10_WB: 0.001862213530864433
[23] C14_7_WB: 0.0018542456314255432
[24] Localtype: 0.0017100370152056327
[25] H23_1: 0.0016924547773258865
[26] H13_2: 0.0016694298879213736
[27] H18_4: 0.0016542756482433641
[28] C07_2_2: 0.0016353239004123863
[29] A7_7: 0.0015853914789918555
[30] H2: 0.001566714862903464
[31] T3_5_1: 0.0015583911814054095
[32] H22_6: 0.001543770140673582
[33] BU227_A: 0.0015179178448931742
[34] C14_8: 0.0014862983143711498
[35] T3_5_2: 0.001483621369116886
[36] T2_2A: 0.0014423222566027191
[37] A7_3: 0.0014419497088214775
[38] MR5: 0.0014222878201291028
[39] I06_6_B: 0.0013969478142561286
[40] C14_5: 0.0013732493765905942

Decision Tree

The second step in feature selection was building a decision tree using all available features. This provided insight into the most important variables by displaying the attribute usage as a percentage.

Combine Results

The results from the two methods used to determine feature importance were combined to select the most important features. The basic features were selected based on their orders from the two methods.

Check Accuracy on Decision Tree

To ensure that the best features were selected, a decision tree was built using the selected features, and the accuracy was evaluated using both training and testing data. Additional features were also tried from the two lists and the accuracy was re-evaluated. This process was repeated until the most accurate combination of features was determined.

Building the Decision Tree

The decision tree was constructed using the chosen features and specific parameters that were determined to be the most optimal through experimentation.

```
library("C50")

features <- c("H1", "C14_2", "C14_5", "C14_6", "C14_7", "C14_9",
"C14_10", "H2", "H6",
              "H6_1", "H13_2", "H18_2", "H22_1", "H3_H4_merged",
"ID1", "ID2", "I03_1",
              "I06_01", "Localtype", "Region", "T2_2A")

dataframe_subset <- data[, features]
dataframe_subset$H1 <- as.factor(dataframe_subset$H1)

ctrl = C5.0Control(sample = 0.7,
                   seed = 123,
                   winnow = T,
                   fuzzyThreshold = T,
                   CF = 0.4,
                   minCases = 6)

model <- C5.0(dataframe_subset[,-1], dataframe_subset$H1, control =
ctrl, trials = 3)
summary(model)

plot(model)
```

Numbers and Figures

Evaluation on training data (5770 cases):

Error = 19.4%

Accuracy = 80.6%

Evaluation on test data (2473 cases):

Error = 20.9%

Accuracy = 79.1%

Confusion matrix:

```
 (a)   (b)   (c)   (d)   (e)   (f)   (g)   (h)   (i)    <-classified as
----  ----  ----  ----  ----  ----  ----  ----  ----
 993                188                                 (a): class Apartment
       1                5                     2         (b): class Barracks
       1                                      1         (c): class Caravan
 298              946                          1         (d): class House
   3               13                                   (e): class Independent room
                    1                                   (f): class Other
                                                        (g): class Shelter/ public house
       1                2                    10         (h): class Tent
   4                    3                               (i): class Villa
```

Attribute usage:

```
 100.00% ID2
  99.95% H6
  82.60% H6_1
  76.33% H2
  72.84% I06_01
  52.08% T2_2A
  45.86% H22_1
  43.00% ID1
  39.76% H18_2
  26.15% H13_2
  10.57% Region
   7.04% H3_H4_merged
   3.83% I03_1
   0.49% C14_5
   0.33% C14_7
```