جامعة بيرزيت

BIRZEIT UNIVERSITY

Linux shell project

ENCS3130

Student’s name and id:

Amr Halahla 1200902.

Manar Shawahni 1201086.

Dr.Mohammad Jubran.

Assistent: Mahmood Abu Awwad.

- This is the main script code that contains the menu and the case statement, also it contains some logic code which do the process of the first entered option by the user, where it should be either ‘r’ or ‘e’, and if not, it will print an error message then return to the main menu again.

```
# Set the initial value of the verified flag to false
verified=false

# Start the main loop
while [ 1 ]; do
  printf "\nr) read a dataset from a file\np) print the names of the features\nl) encode a feature using label encoding\no) enocde a feature using one-hot encoding\nm) apply MinMax scalling\ns) save the processed dataset\ne) exit\n"
  # ask user for the main option
  echo "Please enter your option"
  read option

  # Check if the option is 'r' or 'e'
  if [ "$option" = "r" -o "$option" = "e" ]; then
    # If the option is 'r' or 'e', set the verified flag to true
    verified=true
  fi

  # If the verified flag is false, print a message and continue to the next iteration of the loop
  # because the user should enter either r or e for the first time he run the script.
  if [ "$verified" = false ]; then
    echo "You must first read a dataset from a file"
    continue
  fi

  # Use a case statement to handle each menu option
  case "$option"
  in
    "r" ) # the case where the user should enter the name of the dataset source.
          echo "Please input the name of the dataset file"
          read dataset
          # check if the file exists.
          if [ ! -e "$dataset" ]
          then
              echo "$dataset Not Found!"
              continue
          fi
          # if the name entered exist, then start the format checking process.
          ./labels.sh "$dataset";;
    "p" ) ./case3.sh;;
    "l" ) ./case4.sh "$dataset";;
    "o" ) ./case5.sh "$dataset";;
    "m" ) ./case6.sh "$dataset";;
    "s" )
                saved=false
                echo "Please input the name of the file to save the processed dataset"
                read filename
                cat newdataset.txt > "$filename"
                saved=true
                echo "Data Saved To The File $filename"
                cat "$filename"
                ;;
    "e" )
                if [ "$saved" ];then # check if the processed data saved or not.
                        echo "Are you sure you want to exist"
                        read answer
                        if [ "$answer" = "yes" ];then
                                exit 10
                        else
                                continue
                        fi
                else
                        echo "The processed dataset is not saved. Are you sure you want to exist ?"
                        read answer
                        if [ "$answer" = "yes" ];then
                                exit 10
                        else
                                continue
                        fi
                fi
        ;;
esac
done
```

- **This capture below shows the result if the user started the program with wrong option (neither ‘r’ nor ‘e’):**

```
amr@amr-VirtualBox:~/Project$ ./menu.sh

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
l
You must first read a dataset from a file

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
```

- **This is the first dataset we used:**

```
id;age;gender;height;weight;active;blood;smoke;governorate;
1;20;male;171;55;yes;A+;no;Hebron;
2;20;female;158;55;yes;A-;yes;Jenin;
3;30;male;160;60;no;B+;no;Jerusalem;
4;19;female;162;55;yes;AB+;no;Ramallah;
5;13;female;142;37;no;O-;no;Jenin;
```

- **This is the second dataset we used:**

```
age;sex;bmi;children;smoker;region;charges;
19;female;27;0;yes;southwest;16884;
18;male;33;1;no;southeast;1725;
28;male;33;3;no;southeast;4449;
33;male;22;0;no;northwest;21984;
32;male;28;e;no;northwest;3866;
31;female;25;e;no;southeast;3756;
46;female;33;1;yes;southeast;8240;
```

DATASET 1:

- **Option ‘r’:**
  - **The code:**

```
"r" ) # the case where the user should enter the name of the dataset source.
      echo "Please input the name of the dataset file"
      read dataset
      # check if the file exists.
      if [ ! -e "$dataset" ]
      then
          echo "$dataset Not Found!"
          continue
      fi
      # if the name entered exist, then start the format checking process.
      ./labels.sh "$dataset";;
```

  - In case ‘r’, the user should enter the dataset file name.
  - The logic of the code firstly will check if the file exists or not, if not then it will print an error file not found.
  - If file exists, then the script ‘labels.sh’ will be executed.

‘**labels.sh**’ script:

```
case3.sh   case4.sh   encoded.txt   labels.sh   menu.sh   scale.txt
if [ $# -lt 1 ]; then # check if the user entered a parameter that represent the dataset file name.
  echo "Error: No dataset file provided"
  exit 1
fi
data=$1 # extract the register value (which is the name of the file)
line1=$(sed -n "1p" "$data" | tr ';' '\n') # split the first line by the delimeter ';' so we can then count the number of features.
feature_count=$(echo "$line1" | wc -l) # count the number of features.
# split the second line in order to be able to check if the number of feature's values matches the number of features
line2=$(sed -n "2p" "$data" | tr ';' '\n')
values_count=$(echo "$line2" | wc -l) # count feature's values
#echo "$values_count"
if [ "$feature_count" != "$values_count" ]; then # check if they are equal, if not then print an error data formating message
  echo "The format of the data in the dataset file is wrong"
  exit 11
else
  echo "Format is Clean"
fi
# save a copy of the main dataset file into another file to be processed.
cat "$data" > newdataset.txt
 > sc.txt # file to store the name of the features that have been encoded.
 > scale.txt # to store the features that have been encoded attatched with their codes.
```

- The main functionality of this script is to check the data formatting if it is correct or not.
- The formatting is correct if the number of features in the dataset is equal to the number of values attached with it.
- We made a copy of the dataset in order to perform all of the processes on it.

- Here the output for different tests for case ‘r’:

```
amr@amr-VirtualBox:~/Project$ ./menu.sh

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
r
Please input the name of the dataset file
wrong dataset
wrong dataset Not Found!

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
r
Please input the name of the dataset file
dataset.txt
Format is Clean

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
```

The output when we entered a wrong dataset file

Here the output of the correct dataset file name, with correct formatting.

```
Please enter your option
r
Please input the name of the dataset file
dataset.txt
The format of the data in the dataset file is wrong
```

we put a wrong format in dataset file

- **Option ‘p’:**
  - The code**:**

```
# this script is used to print the names of features.
line=$(head -n 1 newdataset.txt | tr ';' '\n') # split the first line of the file which contains the features names=> each feature's name at a line.
printf "You have these features\n"
echo "$line"
```

  - This script prints the name of the features that exists in the dataset file.

  - **The output for this case:**

```
Please enter your option
p
You have these features
id
age
gender
height
weight
active
blood
smoke
governorate
```

- **Option ‘l’:**
  - The code:

```
 > features.txt # file that will be used to store the name of categorical features.
sed -i '/^$/d' features.txt
dataset="$1" # get the name of the original dataset file
# initializing varibales
counter=1
fet_index=100
# Count the number of lines in the dataset file
numoflines=$(cat newdataset.txt | wc -l)

# Ask the user to input the name of a categorical feature for label encoding
echo "Please input the name of the categorical feature for label encoding"
read feature_name

# Extract the first line of the dataset file and store it in a file called line.txt
line=$(sed -n "1p" newdataset.txt)
echo "$line" > line.txt

# Count the number of fields(features) in line.txt
numf=$(cat line.txt | tr ';' '\n' | wc -l)
# Loop through each field in line.txt
while [ "$numf" -gt 1 ]
do
   # Extract the current field from the first line of the dataset file
   fet=$(echo "$line" | cut -d';' -f"$counter")
   if [ "$fet" = "$feature_name" ];then # check if the entered feature's name matches the feature at the current index.
        fet_index=$((counter)) # if yes, store the index (at which delimeter) of the target feature.
   fi
   # Extract the current field from the second line of the dataset file
   value=$(sed -n "2p" newdataset.txt | cut -d';' -f"$counter") # the value of the current feature

   # Check if the value of the current field is numeric or non-numeric
   if [[ ! "$value" =~ ^[0-9]+$ ]]; then # if its categorical, append the feature name into a file.
       echo "$fet" >> features.txt
   fi

   numf=$((numf-1))
   counter=$((counter+1))
done

if grep -qw "$feature_name" features.txt; then # verify that the entered feature is categorical or not (in order to be encoded).
  echo "Feature $feature_name exist in the file"
else
  echo "The name of categorical feature is wrong"
  exit 3
fi

count_fet=$(cat features.txt | wc -l)
 > fet_value.txt # a file in which the possible values for a specific feature are stored
while [ "$numoflines" -gt 1 ];do # loop through all lines in the dataset to extract the values
        fet_value=$(sed -n "$numoflines p" newdataset.txt | cut -d';' -f"$fet_index") # the value of the feature at the current data line.
        numoflines=$((numoflines-1))
        echo "$fet_value" >> fet_value.txt # append the value into the file.
done

sort  fet_value.txt | uniq > temp.txt # remove the dulicated values
 >codes.txt # a file in which the codes for the unique possible values are stored
cat temp.txt > fet_value.txt
sed -i '/^$/d' fet_value.txt # remove blank lines from the file
sed -i '/^$/d' codes.txt # remove blank lines from the file
#printf "Unique file :\n"
#cat fet_value.txt
counter=$(cat fet_value.txt | wc -l) # the number of unique values for the current specified feature.
code=0
while [ "$counter" -gt 0 ];do # loop through all values to attch it with their codes.
        value=$(sed -n "$counter p" fet_value.txt)
        echo "$code" >> codes.txt # store the unique codes in a file.
        code=$((code+1))
        counter=$((counter-1))
done
paste -d'=' fet_value.txt codes.txt > encoded.txt # concat each value with its own code in one file (value = code).
#cat encoded.txt
sed -i '/^$/d' newdataset.txt
numoflines=$(cat newdataset.txt | wc -l)
echo "" > encodedfile.txt # a file in which the encoded data will be stored temporarily
sed -i '/^$/d' encodedfile.txt
head -n 1 newdataset.txt > encodedfile.txt # the first line will not be encoded in label-encoding.
echo "" > newline.txt
sed -i '/^$/d' newline.txt
counter=2
 > scale_line.txt # This file will be used with the option "m," and it stores every feature that has been encoded with its codes.
echo "$feature_name" >> scale_line.txt # set feature name has been encoded.

echo "The values of the features' codes are encoded as follows :"
cat encoded.txt
```

```
echo "-------------------------------------------------"
while [ "$counter" -le "$numoflines" ];do # loop through each line in the dataset.
        newline=$(sed -n "$counter p" newdataset.txt > newline.txt) # extract current line from the dataset
        cat newline.txt | tr ';' '\n' > tmp.txt && mv tmp.txt newline.txt # splitting
        sed -i '/^$/d' newline.txt # remove blank lines
        linecount=$(cat encoded.txt| wc -l) # get the number of possible encoded values.
        while [ "$linecount" -gt 0 ];do # loop through all possible encoded values attached with their own codes.
                var=$(sed -n "$linecount p" encoded.txt | cut -d'=' -f1) # get the name of the  value
                code=$(sed -n "$linecount p" encoded.txt | cut -d'=' -f2) # get the code of the current value.
                base_value=$(sed -n "$fet_index p" newline.txt) # extract the pointed field(by the 'fet_index' variable) from the line of dataset.
                sed -i "$fet_index s/\b"$var"/"$code"/" newline.txt # replace the value by its specified code.
                linecount=$((linecount-1))
                if [ "$base_value" = "$var" ];then # check if this is the wanted value.
                        echo "$code" >> scale_line.txt # append the code so it will be used later.
                fi
        done
        cat newline.txt | tr '\n' ';' > tmp.txt && mv tmp.txt newline.txt # rearrange the encoded data line as the original format.
        cat newline.txt >> encodedfile.txt # store each encoded line to the encoded data file.
        printf "\n" >> encodedfile.txt
        counter=$((counter+1))
done
cat encodedfile.txt > newdataset.txt # store the encoded data to the new dataset file.
printf "The dataset after label encoding for the feature $feature_name : \n"
cat newdataset.txt # print the final result after encode the current feature.
cat scale_line.txt | tr '\n' ';' > tm.txt && mv tm.txt scale_line.txt # rearrange the file in a useful form.
sed -i '/^$/d' scale_line.txt
cat scale_line.txt >> scale.txt # append the encoded feature with its codes to a file.
printf "\n" >> scale.txt
echo "$feature_name" >> sc.txt # to assign that this feature has been encoded correctly.
```

- This script performs the label encoding process.
- The label encoding can only be performed on the categorical features.
- The algorithm we used can be summarized by some mainly steps:
  - Firstly, we stored the categorical features at a file.
  - Then, we extracted the unique possible values for the wanted feature.
  - We assign a code for each value.
  - The encoding process done on each line of the dataset separately.
  - At the end, we update the encoded feature in the dataset.
- The label encoding process can be performed on different features at different times at the same dataset.

- **Some results attached for different cases:**

```
Please enter your option
l
Please input the name of the categorical feature for label encoding
blood
Feature blood exist in the file
The values of the features' codes are encoded as follows :
A+=0
A-=1
AB+=2
B+=3
O-=4
--------------------------------------------------
The dataset after label encoding for the feature blood :
id;age;gender;height;weight;active;blood;smoke;governorate;
1;20;male;171;55;yes;0;no;Hebron;
2;20;female;158;55;yes;1;yes;Jenin;
3;30;male;160;60;no;3;no;Jerusalem;
4;19;female;162;55;yes;2;no;Ramallah;
5;13;female;142;37;no;4;no;Jenin;
```

Label encoding for the categorical feature ' blood '

```
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
l
Please input the name of the categorical feature for label encoding
smoke
Feature smoke exist in the file
The values of the features' codes are encoded as follows :
no=0
yes=1
--------------------------------------------------
The dataset after label encoding for the feature smoke :
id;age;gender;height;weight;active;blood;smoke;governorate;
1;20;male;171;55;yes;0;0;Hebron;
2;20;female;158;55;yes;1;1;Jenin;
3;30;male;160;60;no;3;0;Jerusalem;
4;19;female;162;55;yes;2;0;Ramallah;
5;13;female;142;37;no;4;0;Jenin;
```

Label encoding for the categorical feature ' smoke '

The dataset after label encoding for two features (blood, smoke).

- **The picture below shows two different cases at same time:**
  - **When the feature entered is not categorical.**
  - **When the user request to encode the feature more than one time.**

```
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
l
Please input the name of the categorical feature for label encoding
smoke
The name of categorical feature is wrong
```

The feature can be encoded only once, bcz since its encoded it will be regarded as a not categorical feature

- **Option ‘o’:**
  - The code**:**

```
> features.txt # a file in whcih the categorical features will be stored
sed -i '/^$/d' features.txt
dataset="$1"
sed -i '/^$/d' newdataset.txt
counter=1
fet_index=100
numoflines=$(cat newdataset.txt | wc -l)
# Ask the user to input the name of a categorical feature for label encoding
echo "Please input the name of the categorical feature for one hot encoding"
read feature_name
# Extract the first line of the dataset file and store it in a file called line.txt
line=$(sed -n "1p" newdataset.txt) # Extract the first line of the dataset.
echo "$line" > line.txt
numf=$(cat line.txt | tr ';' '\n' | wc -l) # count number of features in the dataset.
# Loop through each field in line.txt
while [ "$numf" -gt 1 ]
do
   # Extract the current field from the first line of the dataset file
   fet=$(echo "$line" | cut -d';' -f"$counter")
   if [ "$fet" = "$feature_name" ];then
        fet_index=$((counter))
   fi
   # Extract the current field from the second line of the dataset file
   value=$(sed -n "2p" newdataset.txt | cut -d';' -f"$counter")

   # Check if the value of the current field is numeric or non-numeric
   if [[ ! "$value" =~ ^[0-9]+$ ]]; then
       echo "$fet" >> features.txt # if the feature is categorical, append its name into the file.
   fi

   numf=$((numf-1))
   counter=$((counter+1))
done

if grep -qw "$feature_name" features.txt; then # verify if the entered feature is one of the categorical.
  echo "Feature $feature_name exist in the file"
else
  echo "The name of categorical feature is wrong"
  exit 3
fi

count_fet=$(cat features.txt | wc -l) # count number of categorical features.
> fet_value.txt
while [ "$numoflines" -gt 1 ];do
        fet_value=$(sed -n "$numoflines p" newdataset.txt | cut -d';' -f"$fet_index")
        numoflines=$((numoflines-1))
        echo "$fet_value" >> fet_value.txt # store the values of the current feature into a file.
done
#-------------------------
echo "$feature_name" > scale_line.txt # append the feature to the file at where the encoded features are stored.

sort  fet_value.txt | uniq > temp.txt && mv temp.txt fet_value.txt # sort and remove duplicated values.
cat fet_value.txt > copy.txt
# arrange the feature values as the wanted formatting to replace the feature name in tne dataset
cat fet_value.txt | tr '\n' ';' > tmp.txt && mv tmp.txt fet_value.txt
echo "" > encodedfile.txt # Each encoded line will be appended to this file.
sed -i '/^$/d' encodedfile.txt
values=$(cat fet_value.txt) # the encoding of the feature name
printf "\n"
sed -i '/^$/d' copy.txt
sed -i "s/\b"$feature_name";/"$values"/" newdataset.txt # replace the the feature name with its all possible unique values arranged.
#-------------------------------------------------------
numoflines=$(cat newdataset.txt | wc -l)
while [ "$numoflines" -gt 1 ];do # loop through the dataset.
        sed -n "$numoflines p" newdataset.txt > newline.txt # Exract each line from the dataset in order to be encoded.
        cat newline.txt | tr ';' '\n' > tmp.txt && mv tmp.txt newline.txt # split the current line into multiple lines.
        sed -i '/^$/d' newline.txt
        linecount=$(cat copy.txt | wc -l) # count number of possible values for the current feature.
        parameter=$(sed -n "$fet_index p" newline.txt) # the target value that must be encoding.(the value to be replaced by 1)
         > tempencoded.txt
         counter=1
        while [ "$counter" -le "$linecount" ];do # loop through all possible values to determine which one is the target.
                var=$(sed -n "$counter p" copy.txt) # extract a randomly value from the possible values.
                if [ "$parameter" = "$var" ];then # if we reached the target value, replace it by value 1 in the code.
                        echo "1" >> tempencoded.txt # append 1 to the result of one-hot code.
                        echo "1" >> scale_line.txt # store the code beside the name of the feature so it will be used later(case'm').
                else
                        echo "0" >> tempencoded.txt # if this is not the target, append 0 to the one-hot code.
                        echo "0" >> scale_line.txt

                fi
                counter=$((counter+1))
        done
        cat tempencoded.txt | tr '\n' ';' > tmp.txt && mv tmp.txt tempencoded.txt # rearrange the code to be in the form (0;0;1;0 ...)
        code=$(cat tempencoded.txt) # store the final result of the code for the current value
        #printf "code = %s\n" "$code"
        sed -i "$numoflines s/\b"$parameter";/"$code"/" newdataset.txt # replace the value by its own code in the dataset.
        numoflines=$((numoflines-1))
done
cat scale_line.txt | tr '\n' ';' > tm.txt && mv tm.txt scale_line.txt
sed -i '/^$/d' scale_line.txt
cat scale_line.txt >> scale.txt
printf "\n" >> scale.txt
echo "$feature_name distinct values are :"
cat copy.txt
cat newdataset.txt # show the dataset after encoding.
echo "$feature_name" >> sc.txt # append the feature name to the file where the encoded features are stored.
```

- This script performs the one-hot encoding process.
- The one-hot encoding can only be performed on the categorical features.
- The algorithm we used can be summarized by some mainly steps:
  - Firstly, we stored the categorical features at a file.
  - Then, we extracted the unique possible values for the wanted feature.
  - We replace the feature name by its unique possible values.
  - We assign ‘1’ to the correct value, otherwise we assign ‘0’ to the rest values.
  - The encoding process done on each line of the dataset separately.
  - At the end, we update the encoded feature in the dataset.
- The one-hot encoding process can be performed on different features at different times at the same dataset.

- **Some results attached for different cases:**

```
Please enter your option
o
Please input the name of the categorical feature for one hot encoding
governorate
Feature governorate exist in the file

governorate distinct values are :
Hebron
Jenin
Jerusalem
Ramallah
id;age;gender;height;weight;active;blood;smoke;Hebron;Jenin;Jerusalem;Ramallah;
1;20;male;171;55;yes;0;0;1;0;0;0;
2;20;female;158;55;yes;1;1;0;1;0;0;
3;30;male;160;60;no;3;0;0;0;1;0;
4;19;female;162;55;yes;2;0;0;0;0;1;
5;13;female;142;37;no;4;0;0;1;0;0;
```

One-hot encoding for the feature governorate

```
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
o
Please input the name of the categorical feature for one hot encoding
gedner
The name of categorical feature is wrong
```

Try to encode a feature that not exsist

```
Please input the name of the categorical feature for one hot encoding
gender
Feature gender exist in the file

gender distinct values are :
female
male
id;age;female;male;height;weight;active;blood;smoke;Hebron;Jenin;Jerusalem;Ramallah;
1;20;0;1;171;55;yes;0;0;1;0;0;0;
2;20;1;0;158;55;yes;1;1;0;1;0;0;
3;30;0;1;160;60;no;3;0;0;0;1;0;
4;19;1;0;162;55;yes;2;0;0;0;0;1;
5;13;1;0;142;37;no;4;0;0;1;0;0;
```

One-hot encoding for the feature ' gender '

The dataset after encoding some features using both encoding methods

```
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
o
Please input the name of the categorical feature for one hot encoding
governorate
The name of categorical feature is wrong
```

Can not encode a feature which has been already encoded

```
Please enter your option
p
You have these features
id
age
female
male
height
weight
active
blood
smoke
Hebron
Jenin
Jerusalem
Ramallah
```

Finally, the name of features in the dataset after one-hot encoding.

- **Option ‘m’:**
  - The code:

```
 > features.txt
 > numeric_features.txt
sed -i '/^$/d' features.txt
dataset="$1"
sed -i '/^$/d' "$dataset"
counter=1
fet_index=100
numoflines=$(cat "$dataset" | wc -l)
echo "Please input the name of the feature to be scaled"
read feature_name

# The code below is used to check whether the feature entered is categorical in its original case before determining
# whether it has been encoded or not.
line=$(sed -n "1p" "$dataset") # extract first line from the dataset
echo "$line" > line.txt
numf=$(cat line.txt | tr ';' '\n' | wc -l) # count number of features in order to able looping through it.
# Loop through each field in line.txt
while [ "$numf" -gt 1 ]
do
   fet=$(echo "$line" | cut -d';' -f"$counter") # exract feature name.
   if [ "$fet" = "$feature_name" ];then # verify if its the wanted feature.
        fet_index=$((counter))
   fi
   value=$(sed -n "2p" "$dataset" | cut -d';' -f"$counter")

   # Check if the value of the current field is numeric or non-numeric
   if [[ ! "$value" =~ ^[0-9]+$ ]]; then
       echo "$fet" >> features.txt
   else
        echo "$fet" >> numeric_features.txt
   fi
   numf=$((numf-1))
   counter=$((counter+1))
done
#-----------------------------------------------
# loop to determine if the feature entered isn't categorical, also to save its index.
if ! grep -qw "$feature_name" features.txt;then
        line1=$(head -n 1 newdataset.txt) # Obtain the first line of the dataset in order to determine which numerical feature is the target.
        echo "$line1" | tr ';' '\n' > line1.txt
        line1=$(cat line1.txt | wc -l)
        counter=1
        while [ "$counter" -le "$line1" ];do
                name=$(sed -n "$counter p" line1.txt)
                if [ "$name" = "$feature_name" ];then
                        numeric_index=$((counter)) # save the index of the numeric feature index.
                fi
                counter=$((counter+1))
        done
fi

#-----------------------------------------------
# If the feature is categorical, we must determine whether it has been encoded or not.
if grep -qw "$feature_name" features.txt; then
  echo "Feature $feature_name exist in the file"
  if grep -qw "$feature_name" sc.txt; then # grep through the file containing the names of encoded features.
        echo "The Feature $feature_name Has Been Encoded"
        # Extract only the codes (without the feature name) that are attached after the feature name.
        line=$(grep "$feature_name" scale.txt | cut -d';' -f2-)
        echo "$line" | tr ';' '\n' > scaledcode.txt # Save the codes in a file, one for each line.
        sed -i '/^$/d' scaledcode.txt # remove blanked lines.
        numofcodes=$(cat scaledcode.txt | wc -l)
        #cat scaledcode.txt
        #echo "$numofcodes"
        counter=1
        # Sort the codes to make it easier to find the minimum and maximum values.
        # Extract the min and max numbers
        min=$(sort -n scaledcode.txt | head -n 1)
        max=$(sort -n scaledcode.txt | tail -n 1)
         > vector.txt
        printf "[" >> vector.txt
        printf "max value = %d \nmin value = %d\n" $max $min # print the min and max values.
        while [ "$counter" -le "$numofcodes" ];do  # loop through all codes to be encoded
                xi=$(sed -n "$counter p" scaledcode.txt) # extract each code separately
                upper=$((xi-min)) #calculate the numerator.
                lower=$((max-min)) # calculat denominator.
                xiscale=$(echo "scale=1;$upper/$lower" | bc) # calculate the float result of dividing upper by lower.
                printf "%.2f" $xiscale >> vector.txt # insert the scaled code into the vector
                if [ "$counter" -ne "$numofcodes" ]; then
                        printf ", " >> vector.txt
                fi
                printf "x$counter scaled into %.2f \n" $xiscale
                counter=$((counter+1))
        done
        printf "]." >> vector.txt
        #print the scaled vector
        printf "x-sacled = "
        cat vector.txt
  else
```

```
        echo "this feature $feature_name is categorical feature and must be encoded first"
        exit 3
  fi
# case of numeric features.
elif grep -qw "$feature_name" numeric_features.txt; then
        > numeric_codes.txt # file at where the number feature's values are stored.
        echo "The Feature is Numeric"
        #echo "$numeric_index"
        counter=2
        numoflines=$(cat newdataset.txt | wc -l) # loop through the dataset to extract the values.
        while [ "$counter" -le "$numoflines" ];do
                value=$(sed -n "$counter p" newdataset.txt | cut -d';' -f$numeric_index)
                echo "$value" >> numeric_codes.txt # store the current value
                counter=$((counter+1))
        done
        counter=1
        # Extract the min and max numbers
        min=$(sort -n numeric_codes.txt | head -n 1)
        max=$(sort -n numeric_codes.txt | tail -n 1)
        printf "max value = %d \nmin value = %d\n" $max $min
         > vector.txt
        printf "[" >> vector.txt
        numoflines=$(cat numeric_codes.txt | wc -l)
        while [ "$counter" -le "$numoflines" ];do # loop through all codes.
                xi=$(sed -n "$counter p" numeric_codes.txt) # extract the current code
                upper=$((xi-min)) # calculate the numerator
                lower=$((max-min)) # calculate the denominator
                xiscale=$(echo "scale=1;$upper/$lower" | bc) # calculate the scale of the code.
                printf "%.2f" $xiscale >> vector.txt # insert the scaled value to the vector.
                if [ "$counter" -ne "$numoflines" ]; then
                        printf ", " >> vector.txt
                fi
                printf "x$counter scaled into %.2f\n" $xiscale
                counter=$((counter+1))
        done
        printf "]." >> vector.txt
        printf "x-sacled = "
        cat vector.txt # print the scaled vector.
else
        echo "the feature does not exist in the dataset."
fi
```

- First step, we checked if the feature is categorical or numeric.
- If the feature is numeric, we start the process of minmax scaling.
- If the feature is categorical, we should first check if it has been encoded.
    - If yes, then calculate the scaling vector of the feature.
    - If not, print that the feature must be encoded first.
- Print the scaling vector.

- **Some results attached for different cases:**

```
blood
Feature blood exist in the file
The Feature blood Has Been Encoded
max value = 4
min value = 0
x1 scaled into 0.00
x2 scaled into 0.20
x3 scaled into 0.70
x4 scaled into 0.50
x5 scaled into 1.00
x-sacled = [0.00, 0.20, 0.70, 0.50, 1.00].
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
m
Please input the name of the feature to be scaled
smoke
Feature smoke exist in the file
The Feature smoke Has Been Encoded
max value = 1
min value = 0
x1 scaled into 0.00
x2 scaled into 1.00
x3 scaled into 0.00
x4 scaled into 0.00
x5 scaled into 0.00
x-sacled = [0.00, 1.00, 0.00, 0.00, 0.00].
```

Apply MinMax scaling on the feature blood, and print the scaled vector.

MinMax scaling for the feature ' smoke' , which has been encoded before.

```
Please enter your option
m
Please input the name of the feature to be scaled
id
The Feature is Numeric
max value = 5
min value = 1
x1 scaled into 0.00
x2 scaled into 0.20
x3 scaled into 0.50
x4 scaled into 0.70
x5 scaled into 1.00
x-sacled = [0.00, 0.20, 0.50, 0.70, 1.00].
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
m
Please input the name of the feature to be scaled
gender
Feature gender exist in the file
this feature gender is categorical feature and must be encoded first
```

MinMax scaling on a numeric feature ' id '.

Try to get a MinMax scaling of a categorical feature that has not been encoded before.

- **If the feature that we entered is not in the dataset:**

```
Please enter your option
m
Please input the name of the feature to be scaled
wrongfeature
the feature does not exist in the dataset.

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
```

- **Option ‘s’:**

  - The code**:**

```
    "s" )
                saved=false
                echo "Please input the name of the file to save the processed dataset"
                read filename
                cat newdataset.txt > "$filename"
                saved=true
                echo "Data Saved To The File $filename"
                cat "$filename"
                ;;
```

  - **The output:**

```
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
s
Please input the name of the file to save the processed dataset
saved.txt
Data Saved To The File saved.txt
id;age;gender;height;weight;active;A+;A-;AB+;B+;O-;smoke;governorate;
1;20;male;171;55;yes;1;0;0;0;0;0;0;
2;20;female;158;55;yes;0;1;0;0;0;1;1;
3;30;male;160;60;no;0;0;0;1;0;0;2;
4;19;female;162;55;yes;0;0;1;0;0;0;3;
5;13;female;142;37;no;0;0;0;0;1;0;1;
```

- **Option ‘e’:**
  - The code:

```
    "e" )
                if [ "$saved" ];then
                        echo "Are you sure you want to exist"
                        read answer
                        if [ "$answer" = "yes" ];then
                                exit 10
                        else
                                continue
                        fi
                else
                        echo "The processed dataset is not saved. Are you sure you want to exist ?"
                        read answer
                        if [ "$answer" = "yes" ];then
                                exit 10
                        else
                                continue
                        fi
                fi
        ;;
esac
done
```

- **The output:**

```
Please enter your option
s
Please input the name of the file to save the processed dataset
saved.txt
Data Saved To The File saved.txt
id;age;gender;height;weight;active;A+;A-;AB+;B+;O-;smoke;governorate;
1;20;male;171;55;yes;1;0;0;0;0;0;0;
2;20;female;158;55;yes;0;1;0;0;0;1;1;
3;30;male;160;60;no;0;0;0;1;0;0;2;
4;19;female;162;55;yes;0;0;1;0;0;0;3;
5;13;female;142;37;no;0;0;0;0;1;0;1;

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
e
Are you sure you want to exist
no

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
e
Are you sure you want to exist
yes
amr@amr-VirtualBox:~/Project$
```

The case when the processed dataset is saved to file

```
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
e
The processed dataset is not saved. Are you sure you want to exist ?
no

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
e
The processed dataset is not saved. Are you sure you want to exist ?
yes
amr@amr-VirtualBox:~/Project$
```

The case when the processed dataset is not saved

DATASET 2:

❖ The output of option ‘r’ & ‘p’:

```
r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
r
Please input the name of the dataset file
tempdataset.txt
Format is Clean

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
p
You have these features
age
sex
bmi
children
smoker
region
charges
```

❖ The output of option ‘l’:

```
Please enter your option
l
Please input the name of the categorical feature for label encoding
sex
Feature sex exist in the file
The values of the features' codes are encoded as follows :
female=0
male=1
-----------------------------------------------
The dataset after label encoding for the feature sex :
age;sex;bmi;children;smoker;region;charges;
19;0;27;0;yes;southwest;16884;
18;1;33;1;no;southeast;1725;
28;1;33;3;no;southeast;4449;
33;1;22;0;no;northwest;21984;
32;1;28;e;no;northwest;3866;
31;0;25;e;no;southeast;3756;
46;0;33;1;yes;southeast;8240;
```

❖ The outputs of option ‘o’:

```
Please enter your option
o
Please input the name of the categorical feature for one hot encoding
region
Feature region exist in the file

region distinct values are :
northwest
southeast
southwest
age;sex;bmi;children;smoker;northwest;southeast;southwest;charges;
19;0;27;0;yes;0;0;1;16884;
18;1;33;1;no;0;1;0;1725;
28;1;33;3;no;0;1;0;4449;
33;1;22;0;no;1;0;0;21984;
32;1;28;e;no;1;0;0;3866;
31;0;25;e;no;0;1;0;3756;
46;0;33;1;yes;0;1;0;8240;
```

```
Please enter your option
p
You have these features
age
sex
bmi
children
smoker
northwest
southeast
southwest
charges
```

❖ The outputs of option ‘m’:

```
Please enter your option
m
Please input the name of the feature to be scaled
sex
Feature sex exist in the file
The Feature sex Has Been Encoded
max value = 1
min value = 0
x1 scaled into 0.00
x2 scaled into 1.00
x3 scaled into 1.00
x4 scaled into 1.00
x5 scaled into 1.00
x6 scaled into 0.00
x7 scaled into 0.00
x-sacled = [0.00, 1.00, 1.00, 1.00, 1.00, 0.00, 0.00].
```

```
Please enter your option
m
Please input the name of the feature to be scaled
region
Feature region exist in the file
The Feature region Has Been Encoded
max value = 1
min value = 0
x1 scaled into 0.00
x2 scaled into 1.00
x3 scaled into 0.00
x4 scaled into 0.00
x5 scaled into 1.00
x6 scaled into 0.00
x7 scaled into 1.00
x8 scaled into 0.00
x9 scaled into 0.00
x10 scaled into 1.00
x11 scaled into 0.00
x12 scaled into 0.00
x13 scaled into 0.00
x14 scaled into 1.00
x15 scaled into 0.00
x16 scaled into 0.00
x17 scaled into 1.00
x18 scaled into 0.00
x19 scaled into 0.00
x20 scaled into 0.00
x21 scaled into 1.00
x-sacled = [0.00, 1.00, 0.00, 0.00, 1.00, 0.00, 1.00, 0.00, 0.00, 1.00, 0.00, 0.00, 0.00, 1.00, 0.00, 0.00, 1.00, 0.00, 0.00, 0.00, 1.00].
```

```
Please enter your option
m
Please input the name of the feature to be scaled
sex
Feature sex exist in the file
this feature sex is categorical feature and must be encoded first

r) read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) enocde a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) exit
Please enter your option
m
Please input the name of the feature to be scaled
age
The Feature is Numeric
max value = 46
min value = 18
x1 scaled into 0.00
x2 scaled into 0.00
x3 scaled into 0.30
x4 scaled into 0.50
x5 scaled into 0.50
x6 scaled into 0.40
x7 scaled into 1.00
x-sacled = [0.00, 0.00, 0.30, 0.50, 0.50, 0.40, 1.00].
```

THANK YOU