Faculty of Engineering and Technology

Department of Electrical and Computer Engineering

LINUX LABORATORY- ENCS3130

Section#2

PROJECT#1 "Shell Scripting"

Prepared by:

Abdallah Hamed 1202063

Nafe Abubaker 1200047

Instructor:

Dr. Mohammad Jubran

Instructor's Assistant:

Eng. Katia

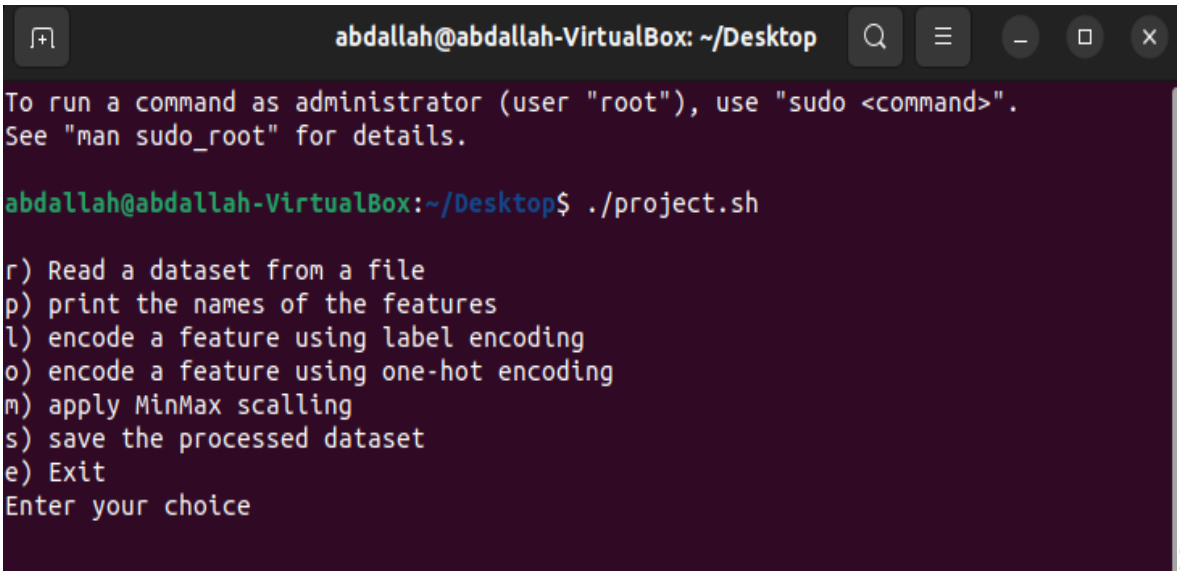
December 29, 2022

Main Idea of the code:

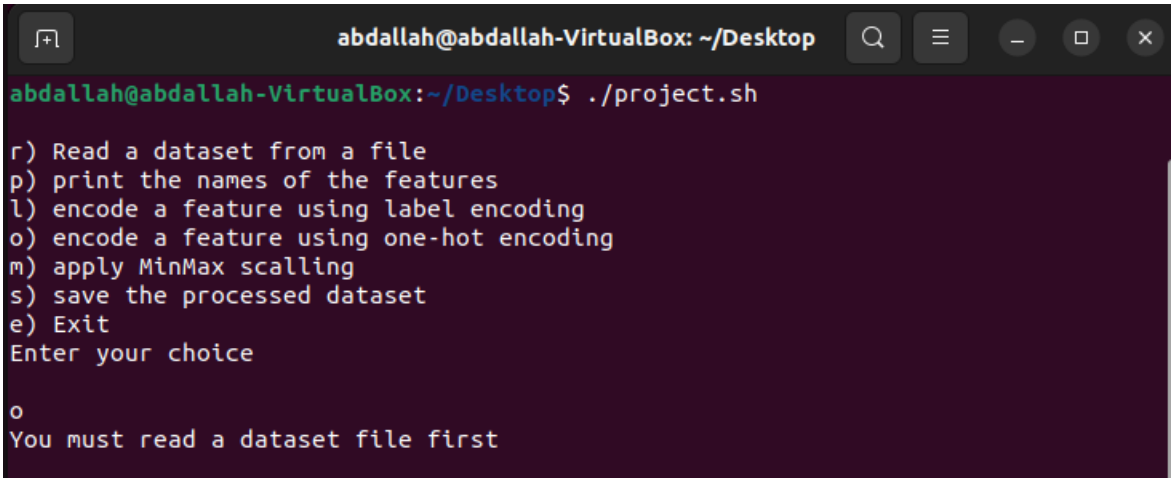
the purpose of this project is to process a dataset contained in a text file by performing feature encoding and scaling. The program, which is written in shell script, presents the user with a main menu from which they can select various tasks to be performed on the data. These tasks include reading the data from the file and encoding categorical data using label encoding or One-Hot encoding methods, scaling numerical data using a specified equation, and saving the modified dataset after completing these processes.

Program details:

Upon initial execution, the program will present a main menu to the user and the main menu will display a list of options outlining the various functions that the program is capable of performing:



If the user enters any character other than 'r' as their initial input, the program will instruct them that they must first read the data before proceeding:



Upon selecting the 'r' option from the main menu, the program will prompt the user to enter the name of the dataset file that they wish to read. Once the file has been successfully accessed, the program will output the contents of the dataset to the user:

Here is another sample of reading a various file with different data:

```
abdallah@abdallah-VirtualBox:~/Desktop$ ./project.sh

r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

r
Enter the name of the file
file2.sh
id;age;gender;height;weight;active;smoke;governorate;
1;33;male;130;83;no;yes;ramallah;
2;56;female;170;63;no;yes;Jericho;
3;25;male;167;98;yes;yes;nablus;
4;28;female;189;56;no;no;Jericho;
5;44;female;192;75;no;no;jerusalem;
```

Code for reading a file :

```
echo "Enter the name of the file"
read FileName
if [ -e "$FileName" ]
then
FileContents=$(cat "$FileName")
while read -r line
do
Lines+=("$line")
NumOfLines=$((NumOfLines + 1))
echo "$line"
done <<< "$FileContents"
```

```
# Count the number of words in the line
linee=$(echo "${Lines[0]}" | tr ";" " ")
word_count=$(echo "$linee" | wc -w) #How many words are in a line
for ((i = 0; i < $word_count; i++))
do
Categories+=($(echo "${Lines[0]}" | tr ';' ' '))
done
Chose_Read_First=1
else
echo "The file does not exist"
fi
;;
```

If the user selects the 'p' option from the main menu, the program will output the first line of the dataset file, which contains the feature names:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

p
id;age;gender;height;weight;active;smoke;governorate;
```

Code for printing the features :

```
#"printing"
if [ $Chose_Read_First -eq 0 ]
```

```
then
echo "You must read a dataset file first"
else
echo "${Lines[0]}"
fi
;;
```

Upon selecting the 'l' option from the main menu, the program will request that the user input the name of the feature that they wish to encode using the label encoding method. If the user inputs an invalid feature name or the feature does not exist in the dataset, the program will display ”wrong category “ message and return to the main menu:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

l
Please input the name of the categorical feature for label encoding
car
Wrong category!

r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
```

If the user specifies a categorical feature as the one to be encoded, the program will proceed to encode the feature by replacing the categorical data with numerical values. The program will then output each modified data value to the screen:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

l
Please input the name of the categorical feature for label encoding
governorate
jerusalem = 0
nablus = 1
ramallah = 2
Label encoding done..
```

```
id;age;gender;height;weight;active;smoke;governorate;
1;30;male;170;88;no;yes;2;
2;25;female;160;65;no;no;2;
3;28;male;165;72;yes;yes;1;
4;44;male;188;90;no;no;0;
5;60;female;166;70;no;no;0;
```

Here is another sample of label encoding for another file(file2.sh):

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

l
Please input the name of the categorical feature for label encoding
governorate
Jericho = 0
jerusalem = 1
nablus = 2
ramallah = 3
Label encoding done..
```

```
id;age;gender;height;weight;active;smoke;governorate;
1;33;male;130;83;no;yes;3;
2;56;female;170;63;no;yes;0;
3;25;male;167;98;yes;yes;2;
4;28;female;189;56;no;no;0;
5;44;female;192;75;no;no;1;
```

Code for label encoding:

```
#label Encoding"
if [ $Chose_Read_First -eq 0 ]
then
echo "You must read a dataset file first"
else
echo "Please input the name of the categorical feature for label encoding"
read Input_Categorical
#Checking if the categorical input exists
for ((i = 0; i < $word_count; i++))
 do
   if [ "${Categories[i]}" = "$Input_Categorical" ]
   then
        index=$((i + 1))
        CategoryExist=1
 fi
 done
#Displaying result to the user
if [ $CategoryExist = 0 ]
then
echo "Wrong category!"
elif [ $CategoryExist = 1 ]
then
truncate -s 0 test.txt # to empty the file
for ((i = 1; i < $NumOfLines; i++))
```

```
do
   Word=($(echo "${Lines[i]}" | cut -d ';' -f $index))
   WordInLine[i]="$Word"
   echo "${WordInLine[i]}" >> test.txt
done
#append the unique values to unique.txt
sort test.txt | uniq -i  > unique.txt
#reading from the file and storing in an array
readarray -t Variables < unique.txt
size=$(wc -l < unique.txt)
for ((i =0 ; i < size; i++))
do
z=${Variables[i]}
echo "$z = $i"
done
for ((i = 1; i < NumOfLines; i++))
do

kelme=($(echo "${Lines[i]}" | cut -d ';' -f $index))
for ((j = 0; j < $size; j++))
do
if [ "${Variables[j]}" = "$kelme" ]
then
Lines[i]=$(echo "${Lines[i]}" | sed 's/'$kelme'/'$j'/g')
```

```
fi

done

done

echo "Label encoding done.."

fi

fi

;;
```

When the user selects option ‘o’ for one-hot encoding, the program will prompt the user to enter the name of the feature to be encoded. If the user enters a non-existent or incorrectly spelled feature, a message will be displayed for the user and the program will return to the main menu:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

o
Enter the feature to be encoded ( One hot Encoding)
car
Feature does not exist!
```

If the user enters a correct feature, the program will then proceed to one-hot encode the feature, replacing the original feature with the encoded categorical data, as well as replacing the original data with the corresponding codes of 0 or 1:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

o
Enter the feature to be encoded ( One hot Encoding)
governorate
./project.sh: line 233: [: =: unary operator expected
./project.sh: line 233: [: =: unary operator expected
./project.sh: line 233: [: =: unary operator expected
One hot encoding done..

^Cabdallah@abdallah-VirtualBox:~/Desktop$ cat hani
id;age;gender;height;weight;active;smoke;jerusalem;nablus;ramallah;
1;30;male;170;88;no;yes;0;0;1;
2;25;female;160;65;no;no;0;0;1;
3;28;male;165;72;yes;yes;0;1;0;
4;44;male;188;90;no;no;1;0;0;
5;60;female;166;70;no;no;1;0;0;
abdallah@abdallah-VirtualBox:~/Desktop$
```

Here is another sample of encoding one-hot for another file(file2.sh):

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

o
Enter the feature to be encoded ( One hot Encoding)
governorate
./project.sh: line 233: [: =: unary operator expected
./project.sh: line 233: [: =: unary operator expected
./project.sh: line 233: [: =: unary operator expected
./project.sh: line 233: [: =: unary operator expected
One hot encoding done..

r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

e
You have not saved the dataset!
Exit without saving? (Answer yes/no)
no
PLease enter the filename to save the processed dataset into
text3
Saved --> text3
abdallah@abdallah-VirtualBox:~/Desktop$ cat t
text3    tst.txt
abdallah@abdallah-VirtualBox:~/Desktop$ cat text3
id;age;gender;height;weight;active;smoke;Jericho;jerusalem;nablus;ramallah;
1;33;male;130;83;no;yes;0;0;0;1;
2;56;female;170;63;no;yes;1;0;0;0;
3;25;male;167;98;yes;yes;0;0;1;0;
4;28;female;189;56;no;no;1;0;0;0;
5;44;female;192;75;no;no;0;1;0;0;
abdallah@abdallah-VirtualBox:~/Desktop$
```

Code for one-hot encoding :

```
#"one hot encoding"
if [ $Chose_Read_First -eq 0 ]
then
echo "You must read a dataset file first"
else
echo "Enter the feature to be encoded ( One hot Encoding)"
read F
for ((i = 0; i < $word_count; i++))
  do
     if [ "${Categories[i]}" = "$F" ]
     then
          indexxx=$((i + 1))
          CategoryExistt=1
  fi
  done
#Displaying result to the user
if [ $CategoryExistt = 0 ]
then
echo "Feature does not exist!"
elif [ $CategoryExistt = 1 ]
then
truncate -s 0 tst.txt # to empty the file
for ((i = 1; i < $NumOfLines; i++))
do
```

```
    jubran=($(echo "${Lines[i]}" | cut -d ';' -f $indexxx))
    WordinLine[i]="$jubran"
    echo "${WordinLine[i]}" >> tst.txt
done
#append the unique values to unique.txt
sort tst.txt | uniq -i  > unique1.txt
#reading from the file and storing in an array
readarray -t Variable < unique1.txt
size1=$(wc -l < unique1.txt)
#to delete the word chosen from the user and it's values
for (( i=0 ; i<$NumOfLines ; i++))
do
IFS=';' read -ra words <<< "${Lines[i]}"
# Delete the (category chosen from the user) from the array
unset words[$indexxx-1]
# Join the array back into a string
Lines[i]=$(IFS=';'; echo "${words[*]}")
Lines[i]="${Lines[i]}"';'
done
#to concatinate the values to the first line
for ((i=0 ; i<$size1 ; i++))
do
if [ "$i" -eq 0 ]
then
Lines[0]=${Lines[0]}${Variable[i]}
```

```
else
Lines[0]=${Lines[0]}";"${Variable[i]}
fi
done
Lines[0]=${Lines[0]}";"
readarray -t array < tst.txt
for ((i=0 ; i<$NumOfLines ; i++))
do
for ((j=0 ; j<$size1 ; j++))
do
if [ ${array[i]} = ${Variable[j]} ]
then
Lines[i+1]=${Lines[i+1]}"1;"
else
Lines[i+1]=${Lines[i+1]}"0;"
fi
done
done
echo "One hot encoding done.."
fi
fi
;;
```

When the user enters the value 'm', the program will prompt them to enter the name of a feature, and then use the minimum and maximum values of that feature to apply min-max scaling. If the user enters a category feature a message will appear to tell the user that this feature has to be encoded first, and the program will return to the main menu:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

m
Provide the name feature to be scaled
gender
this feature 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) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice
```

When the user enters a feature with numerical data that exists within the dataset, the code will evaluate the min-max scale(scale = $(xi - min) / (max - min)$for each data point and replace its original value with the scaled value and return to main menu:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

m
Provide the name feature to be scaled
age


MaxValue = 60
MinValue = 25
1;.14;male;170;88;no;yes;ramallah;
2;0;female;160;65;no;no;ramallah;
3;.08;male;165;72;yes;yes;nablus;
4;.54;male;188;90;no;no;jerusalem;
5;1.00;female;166;70;no;no;jerusalem;
```

Here is another sample of applying min-max scale after encoded a category feature for (file2.sh):

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

l
Please input the name of the categorical feature for label encoding
governorate
Jericho = 0
jerusalem = 1
nablus = 2
ramallah = 3
Label encoding done..

r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
```

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

m
Provide the name feature to be scaled
governorate


MaxValue = 3
MinValue = 0
1;1.001.00;male;11.000;81.00;no;yes;1.00;
2;56;female;170;63;no;yes;0;
3;.665;male;167;98;yes;yes;.66;
4;28;female;189;56;no;no;0;
5;44;female;.3392;75;no;no;.33;
```

Code for min-max scale:

```
#"minimizing"

if [ $Chose_Read_First -eq 0 ]

then
```

```
echo "You must read a dataset file first"
else
echo "Provide the name feature to be scaled"
read feature
   for ((i = 0; i < $word_count; i++))
         do
         if [ "${Categories[i]}" = "$feature" ]
         then
                indexx=$((i + 1))
    fi
    done
        mota8ayer=($(echo "${Lines[1]}" | cut -d ';' -f $indexx))
  # Test if the variable is an integer using expr
  if expr "$mota8ayer" : '^[0-9]*$' >/dev/null
  then
   for ((i = 0; i < $word_count; i++))
  do
    if [ "${Categories[i]}" = "$feature" ]
    then
        indexx=$((i + 1))
        MaxValue=($(echo "${Lines[1]}" | cut -d ';' -f $indexx ))
        MinValue=($(echo "${Lines[1]}" | cut -d ';' -f $indexx ))

           for ((j = 0; j < $NumOfLines; j++)) do
```

```
  if [ $(echo "${Lines[j]}" | cut -d ';' -f $indexx ) -gt $MaxValue ]
          then
       MaxValue=($(echo "${Lines[j]}" | cut -d ';' -f $indexx ))

   elif [ $(echo "${Lines[j]}" | cut -d ';' -f $indexx ) -lt $MinValue ]
       then
       MinValue=($(echo "${Lines[j]}" | cut -d ';' -f $indexx ))
       fi
done
fi
done #End of for loop
echo "MaxValue = $MaxValue"
echo "MinValue = $MinValue"
 for ((i = 1; i < $NumOfLines; i++))
 do
 Value=($(echo "${Lines[i]}" | cut -d';' -f $indexx))
 num1=$(($Value-$MinValue))
 num2=$(($MaxValue-$MinValue))
 NewValue=$(echo "scale=2; $num1/$num2" | bc) #math expressions
 #echo "$Value"
 #echo "$NewValue"
 Lines[i]=$(echo "${Lines[i]}" | sed 's/'$Value'/'$NewValue'/g')
 echo "${Lines[i]}"
 done
```

```
    else
    echo "this feature is categorical feature and must be encoded first"
    fi
        fi # if the user had checked the read first
;;
```

When the user enters 's', the function to save the edited dataset will be triggered, and the program will prompt the user to enter the name of the file they want to save the dataset in. The contents of the saved file will then be contained all operation that has been applied:

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

s
PLease enter the filename to save the processed dataset into
text.txt
```

```
^Cabdallah@abdallah-VirtualBox:~/Desktop$ cat text.txt
id;age;gender;height;weight;active;smoke;governorate;
1;.14;male;170;88;no;yes;2;
2;0;female;160;65;no;no;2;
3;.08;male;165;72;yes;yes;1;
4;.54;male;188;90;no;no;0;
5;1.00;female;166;70;no;no;0;
```

Code for saving :

```
#"saving"
if [ $Chose_Read_First -eq 0 ]
```

```
then
echo "You must read a dataset file first"
else
echo "PLease enter the filename to save the processed dataset into"
read AfterFile
for ((i=0 ;i<$NumOfLines ; i++))
do
echo ${Lines[i]} >> $AfterFile
done

Saved=1
fi
;;
```

When the user selects option 'e' before exiting the program, the program will check whether the edited dataset has been saved to a file. If it has not been saved, the program will inform the user and give them the option to save it by entering 'no' and specifying a file name, or to exit without saving by entering 'yes':

```
l
Please input the name of the categorical feature for label encoding
governorate
jerusalem = 0
nablus = 1
ramallah = 2
Label encoding done..

r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

e
You have not saved the dataset!
Exit without saving? (Answer yes/no)
```

```
r) Read a dataset from a file
p) print the names of the features
l) encode a feature using label encoding
o) encode a feature using one-hot encoding
m) apply MinMax scalling
s) save the processed dataset
e) Exit
Enter your choice

e
You have not saved the dataset!
Exit without saving? (Answer yes/no)
no
PLease enter the filename to save the processed dataset into
abd.txt
Saved --> abd.txt
abdallah@abdallah-VirtualBox:~/Desktop$
```

Code for exiting:

```
if [ $Saved -eq 0 ]
then
echo "You have not saved the dataset!"
echo "Exit without saving? (Answer yes/no)"
```

```
read op
if [ $op = "no" ]
then
echo "PLease enter the filename to save the processed dataset into"
read AfterFile
for ((i=0 ;i<$NumOfLines ; i++))
do
echo ${Lines[i]} >> $AfterFile
done
echo "Saved --> $AfterFile"
Saved=1
fi
else
echo "Thanks for using our program"
fi
;;
*)
echo "Invalid Option.Try the given ones"
;;
esac
done
```

Conclusion:

This project was useful because it allowed us to become familiar with various commands. Additionally, the project contained a wide range of

ideas that covered much of the material covered in the lab. Finally, it was enjoyable to work in a team as we were able to share ideas and learn about teamwork.