Istanbul Okan University

Software Engineering Department

CENG 394 Data Mining

Term Project

Mustafa Doruk ÇİL 200218006

Project Aim

In a bank branch, obtaining credit loan type according to the customer data and developing a hypothesis using the data.

Data Set

Data set consists of 1000 columns and 10 attributes. Dataset is in csv format. Each attribute separated with “,”.

```
ID,Age,Sex,Job,Housing,Saving accounts,Checking account,Credit amount,Duration,Purpose
0,67,male,2,own,little,little,1169,6,radio/TV
1,22,female,2,own,little,moderate,5951,48,radio/TV
2,49,male,1,own,little,little,2096,12,education
3,45,male,2,free,little,little,7882,42,furniture/equipment
4,53,male,2,free,little,little,4870,24,car
5,35,male,1,free,little,little,9055,36,education
6,53,male,2,own,quite rich,,2835,24,furniture/equipment
7,35,male,3,rent,little,moderate,6948,36,car
8,61,male,1,own,rich,little,3059,12,radio/TV
9,28,male,3,own,little,moderate,5234,30,car
10,25,female,2,rent,little,moderate,1295,12,car
11,24,female,2,rent,little,little,4308,48,business
12,22,female,2,own,little,moderate,1567,12,radio/TV
13,60,male,1,own,little,little,1199,24,car
14,28,female,2,rent,little,little,1403,15,car
```

(Photo 1: Dataset snapshot)

The attributes are the following

ID: An identificator of each value. Data type is integer. (This attribute does not used in the data mining study, this attribute is important for CRUD (Create, Read, Update and Delete) operations).

Age: Identifies customer age. Data type is integer. The attribute takes values between 20 and 70.

Sex: Identifies gender of the customer. Data type is string. The attribute takes one of the following values: “male”, “female”.

Job: Shows the number of job of each customer. Data type is integer. The attribute takes one of the following values: 0, 1, 2, 3.

Housing: Shows the customer housing status. Data type is string. The attribute takes one of the following values: “own”: indicates customer has his/her own house,

“rent”: indicates customer has been living in a rented house,

“free”: indicates there is no information about customer has house or not.

Saving Accounts: Shows the customer saving account status. Data type is string. The attribute takes one of the following values:

“little”: indicates the customer has little savings,

“moderate”: indicates the customer has middle (not little not much) savings,

“quite”: indicates the customer has much savings,

“rich”: indicates the customer has very much savings

Checking Accounts: Shows the customer checking account status. Data type is string. The attribute takes one of the following values:

“little”: indicates the customer has little savings,

“moderate”: indicates the customer has middle (not little not much) savings,

“quite”: indicates the customer has much savings,

“rich”: indicates the customer has very much savings

Credit Amount: Shows credit loan amount. Data type is integer. The attributes takes values between 1 and 20000.

Duration: Shows the credit loan duration in terms of the month. Data type is integer. The attribute takes values between 6 and 56.

Purpose: Shows the credit loan purpose. This attribute is target attribute (We want to predict this attribute values according to training set data). Data type is string. The attribute takes one of the following values:

“radio/TV”, ”education”, “furniture/equipment”, “car”, “business”, “domestic appliances “, “vacation/others”.

700 data is used for training, 300 data is used for testing.

Dataset does not need data preprocessing.

Used Technologies

Python Programming Language (Version: 3.11.9): Python is very popular programming language for data science and machine learning because it has very useful libraries.

Numpy (Version 1.26.0): Used for building decision tree.

Pandas (Version: 2.2.1): Used for training the dataset, testing the dataset.

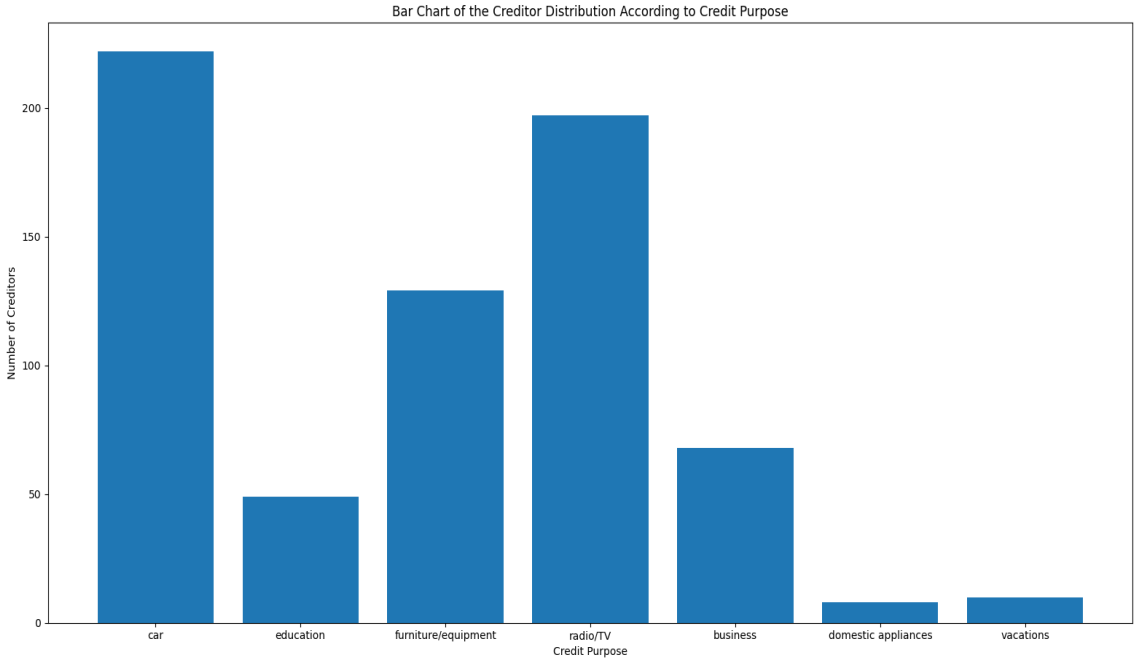
Matplotlib(Version: 3.8.0): Used for plotting charts for data visualization.

Data and Charts

According to training dataset (which includes 700 data) these informations, graphs and inferences has been gained.

Credit Purpose	Number of Creditors
Car	222
Education	49
Furniture/Equipment	129
Radio/TV	197
Business	68
Domestic Appliances	8
Vacation/Others	10

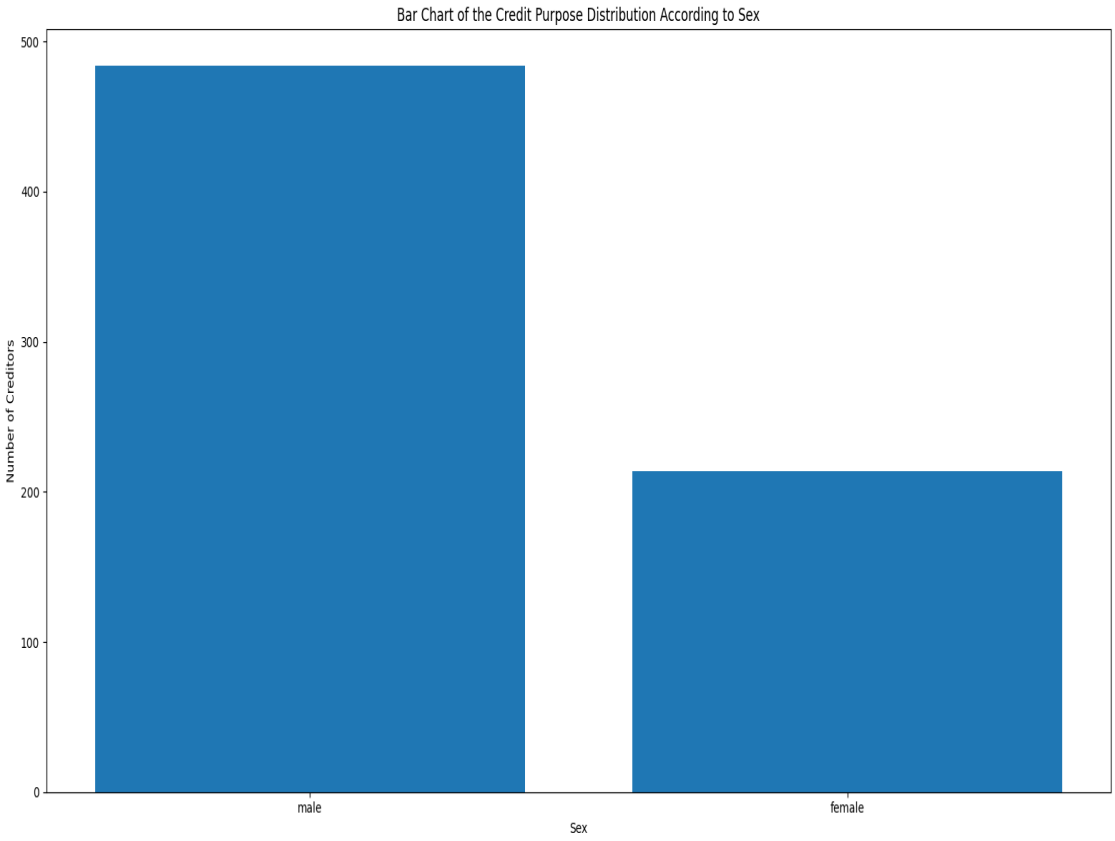
(Data 1: Creditor Distribution According to Credit Purpose)



(Photo 2: Bar Chart of the Creditor Distribution According to Credit Purpose)

Sex	Number of Creditors
Male	484
Female	216

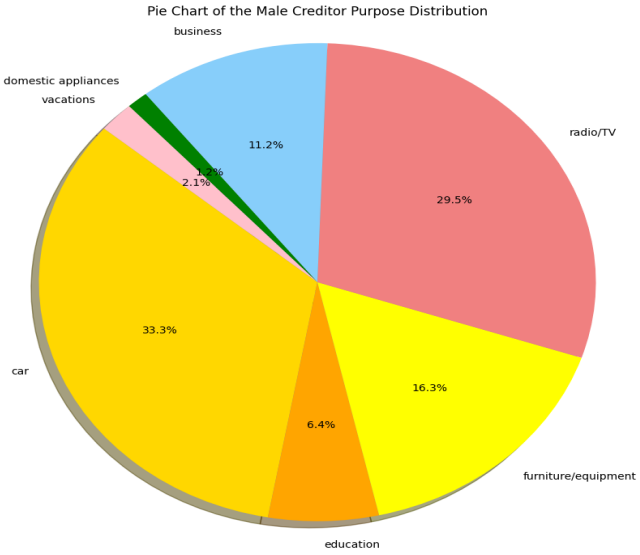
(Data 2: Number of Creditors According to Sex)



(Photo 3: Bar Chart of the Creditor Distribution According to Sex)

Car	Education	Furniture/Equipment	Radio/TV	Business	Domestic Appliances	Vacation/Others
161	31	79	143	54	6	10

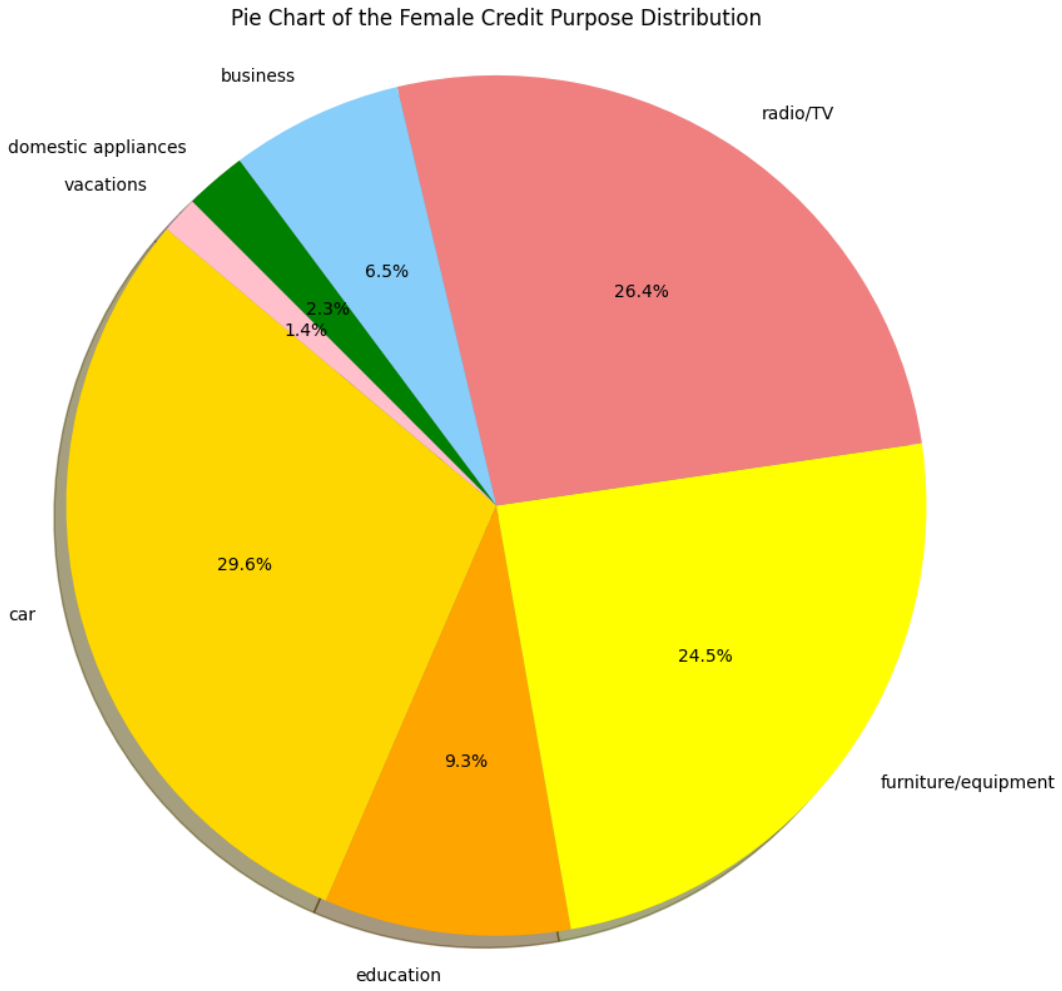
(Data 3: Male Creditor Distribution)



(Photo 4: Pie Chart of the Male Creditor Purpose Distribution)

Car	Education	Furniture/Equipment	Radio/TV	Business	Domestic Appliances	Vacation/Others
64	20	53	57	14	5	3

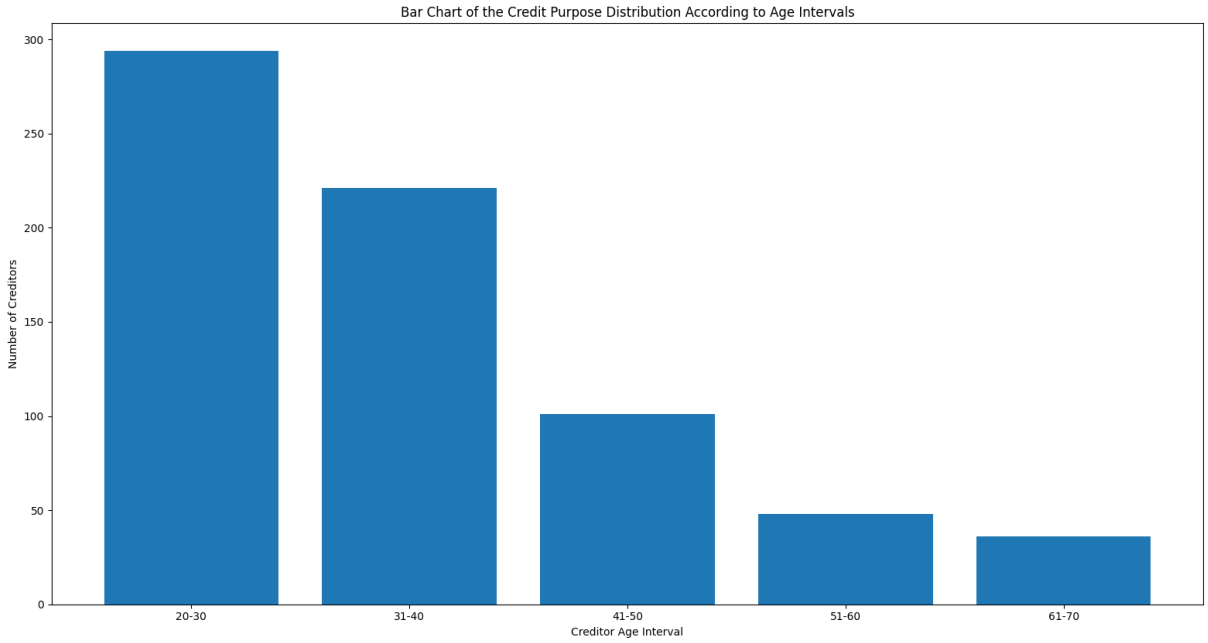
(Data 4: Female Creditor Distribution)



(Photo 5: Pie Chart of the Female Creditor Purpose Distribution)

Creditor Age Interval	Number of Creditors
20-30	294
31-40	221
41-50	101
51-60	48
61-70	36

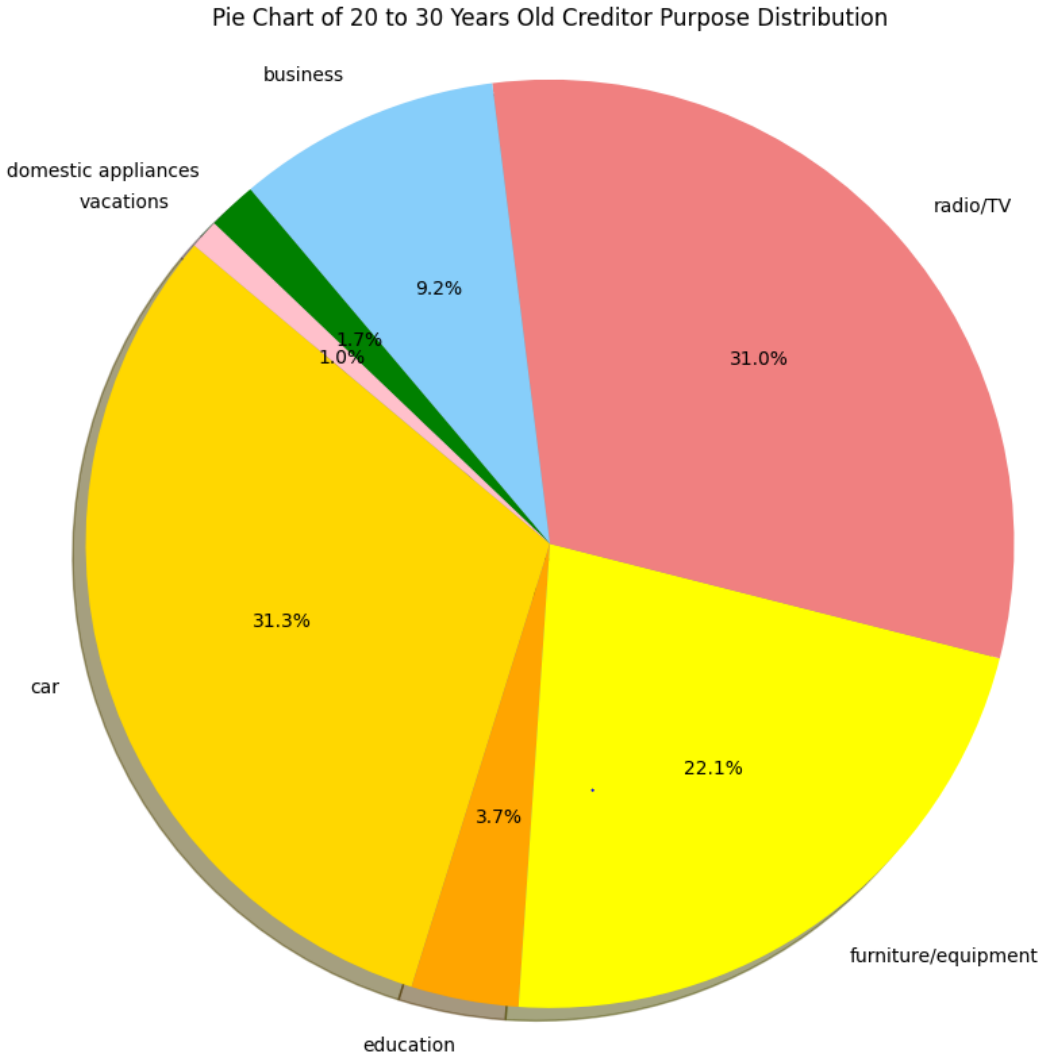
(Data 5: Creditor Age Distribution)



(Photo 6: Bar Chart of the Credit Purpose Distribution According to Age Intervals)

Car	Education	Furniture/Equipment	Radio/TV	Business	Domestic Appliances	Vacation/ Appliances
92	11	65	91	27	5	3

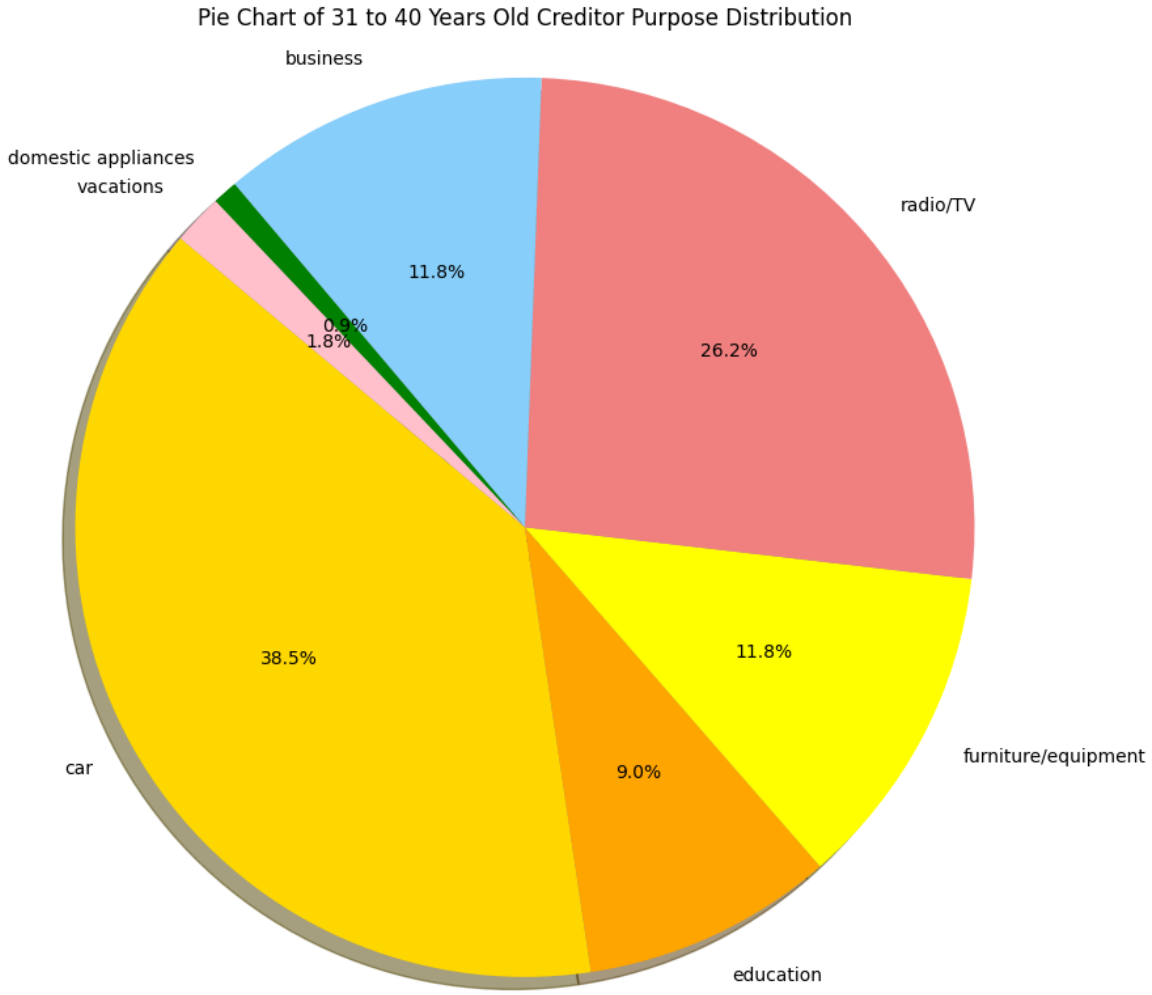
(Data 6: 20 to 30 Years Old Creditor Purpose Distribution)



(Photo 7: Pie Chart of 20 to 30 Years Old Creditor Purpose Distribution)

Car	Education	Furniture/Equipment	Radio/TV	Business	Domestic Appliances	Vacation/ Appliances
85	20	26	58	26	2	4

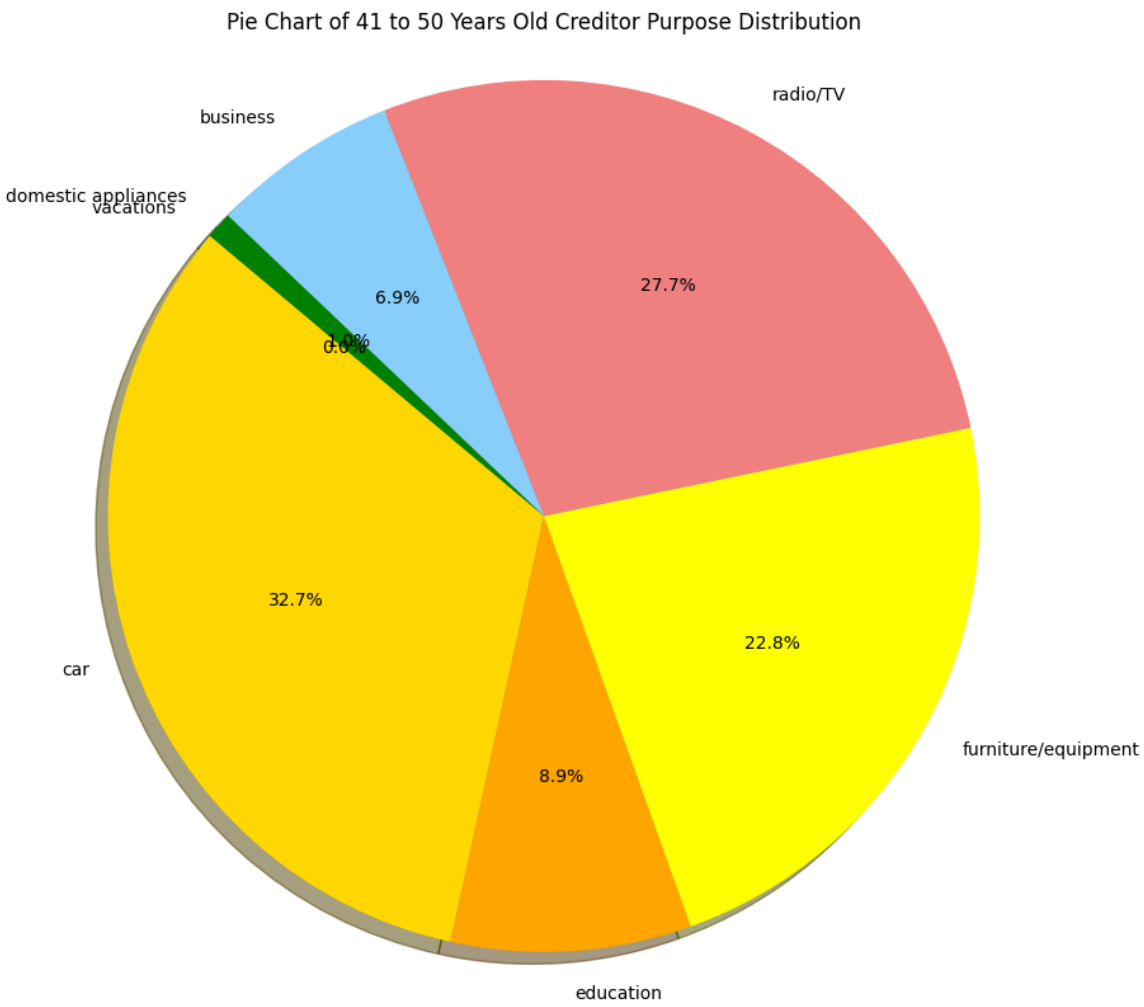
(Data 7: 31 to 40 Years Old Creditor Purpose Distribution)



(Photo 8: Pie Chart of 31 to 40 Years Old Creditor Purpose Distribution)

Car	Education	Furniture/Equipment	Radio/TV	Business	Domestic Appliances	Vacation/ Appliances
33	9	23	28	7	1	0

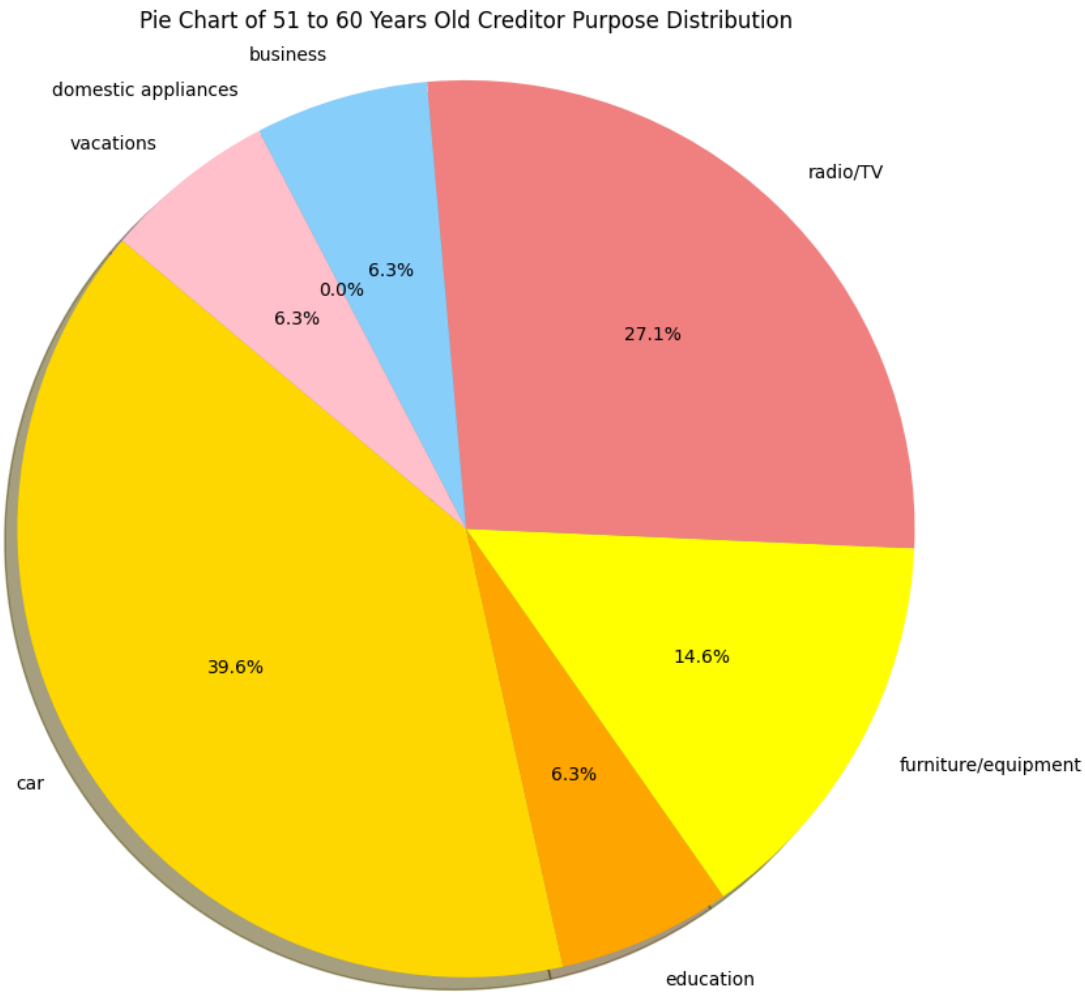
(Data 8: 41 to 50 Years Old Creditor Purpose Distribution)



(Photo 9: Pie Chart of 41 to 50 Years Old Creditor Purpose Distribution)

Car	Education	Furniture/Equipment	Radio/TV	Business	Domestic Appliances	Vacation/ Appliances
19	3	7	13	3	0	3

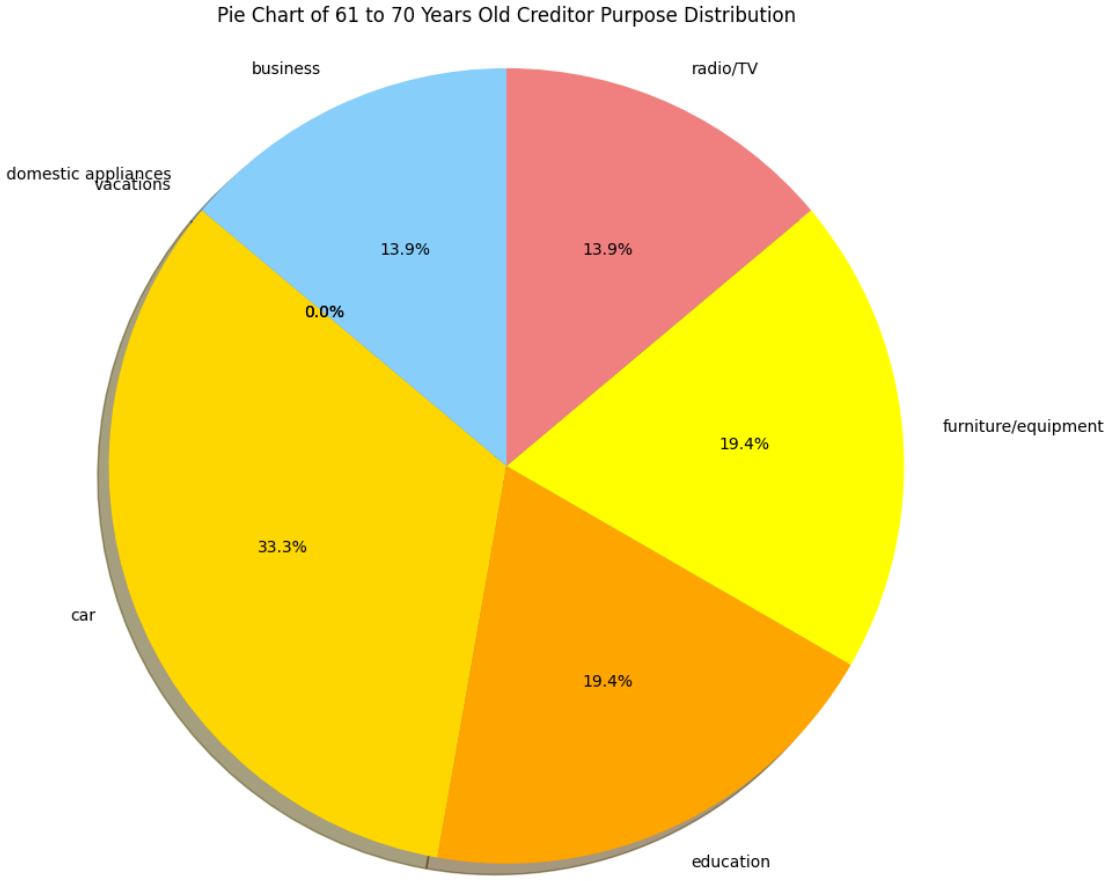
(Data 9: 51 to 60 Years Old Creditor Purpose Distribution)



(Photo 10: Pie Chart of 51 to 60 Years Old Creditor Purpose Distribution)

Car	Education	Furniture/Equipment	Radio/TV	Business	Domestic Appliances	Vacation/ Appliances
12	7	7	5	5	0	0

(Data 10: 61 to 70 Years Old Creditor Purpose Distribution)



(Photo 11: Pie Chart of 61 to 70 Years Old Creditor Purpose Distribution)

Coding

```python
import pandas as pd

#Get data from csv file
def get_data(dataset_path,attribute_names):
    data_set = pd.read_csv(dataset_path,skiprows=1,
                           names=attribute_names)
    return data_set
```

(Photo 12: Code of Reading CSV File with Pandas)

```python
import numpy as np

#Calculates entropy according to an attribute
def calculate_entropy(data):
    classes, counts = np.unique(data, return_counts=True)
    probability = counts/len(data)
    entropy = -np.sum(probability * np.log2(probability))

    return entropy
```

(Photo 13: Code of Calculating Entropy with Numpy)

```python
#calculates information gain according to an attribute
def calculate_information_gain(data, feature_name, class_name):
    total_entropy = calculate_entropy(data[class_name])

    feature_values, feature_counts = np.unique(data[feature_name], return_counts=True)
    weighted_entropy = np.sum([(feature_counts[i] / len(data)) *
                               calculate_entropy(data[data[feature_name]
                                                      == feature_values[i]][class_name])
                                                      for i in range(len(feature_values))])

    information_gain = total_entropy - weighted_entropy

    return information_gain
```

(Photo 14: Information Gain Calculation)

```python
#Builds decision tree according to data, attributes and target attribute
def build_decision_tree(data, features, class_name):
    # If all samples have the same class label, return a leaf node with that class label
    if len(np.unique(data[class_name])) == 1:
        return np.unique(data[class_name])[0]

    # If there are no more features to split on, return the most common class label
    if len(features) == 0:
        return np.unique(data[class_name])[np.argmax(np.unique(data[class_name],
                                                               return_counts=True)[1])]

    # Find the feature that provides the highest information gain
    information_gains = [calculate_information_gain(data, feature, class_name)
                         for feature in features]
    best_feature_index = np.argmax(information_gains)
    best_feature = features[best_feature_index]
```

```python
#Create a decision tree in dictionary data structure
decision_tree = {best_feature: {}}
features = [feature for feature in features if feature != best_feature]

for value in np.unique(data[best_feature]):
    subdata = data[data[best_feature] == value]
    subtree = build_decision_tree(subdata, features, class_name)
    decision_tree[best_feature][value] = subtree

return decision_tree
```

(Photo 15 & 16: Decision Tree Creation)

```python
#Testing function
def test(test_data):
    #Assign an variable which contains number of correct predictions
    correct_predictions = 0

   #Start a loop for each item of test data
    for index, row in test_data.iterrows():
        #Get actual target attribute value
        actual_class = row["Purpose"]

        #Predict target attribute according to credit amount, age sex and job
        predicted_class = predict_class(row["Credit amount"],
                                        row["Age"],
                                        row["Sex"],
                                        row["Job"])
```

```python
    #If actual class is equal to predicted class
    #It means that decision tree algorithm has predicted correctly
    if(actual_class == predicted_class):
        #Increase correct predictions +1
        correct_predictions+=1

#Get test data size for calculating accuracy
test_data_size = len(test_data)

#Calculate accuracy percentage
accuracy = float(correct_predictions/test_data_size)*100

#Print the testing results
print(f"Test Sample Size: {test_data_size}")
print(f"Number of correct predictions: {correct_predictions}")
print(f"Accuracy(%): {accuracy}")
```

(Photo 17 & 18: Test Dataset Algorithm Code)

```python
import matplotlib.pyplot as plt

#Bar Chart Draw Function
def draw_bar_chart(xAxisValues,yAxisValues,xLabel,yLabel,chart_title):
    #Create Bars According to Inputs
    plt.bar(xAxisValues,yAxisValues)

    #Specify X Axis Label
    plt.xlabel(xLabel)

    #Specify Y Axis Label
    plt.ylabel(yLabel)

    #Specify Bar Chart Title
    plt.title(chart_title)

    #Show the Chart
    plt.show()
```

(Photo 19: Bar Chart Draw Function Code)

```python
#Pie Chart Chart Draw Function
def draw_pie_chart(labels,percentages,chart_title):
    #Specify Pie Colors
    colors = ["gold", "orange", "yellow","lightcoral", "lightskyblue", "green", "pink"]

    #Plot the Pie Chart
    plt.figure(figsize=(10, 10))

    #Specify Pie Settings
    plt.pie(percentages, labels=labels, colors=colors, autopct='%1.1f%%',
            shadow=True, startangle=140)
    plt.axis("equal")

    plt.title(chart_title)

    #Display the Chart
    plt.show()
```

(Photo 20: Bar Chart Draw Function Code)

Used Algorithm

Decision tree algorithm used for this algorithm. Decision tree is an classification algorithm. So according to the data attributes we are going to classify the data into a class. The decision tree is in the decision_tree.txt file in dictionary data structure. Credit amount is enough for identifying most of the data. Other data can be identified according to sex, age and and job attribute.

Hypothesis

F: Abbreviation of facts. This hypothesis depends on mainly 4 facts.

F1: Males are uses more credit loan compared with females.

F2: Both males and females are using credit loans mainly for car.

F3: Younger people uses more credit loan compared with older people.

F4: In all age intervals car is the most preferred credit loan purpose but second the mostly preffered credit loan purpose changes accordingly age intervals.

Decision Tree Rules

The decision tree rules reduced to 8 with optimization and categorization of the data.

```
if(credit_amount == 1474):
    if(sex=="female"):
        return "furniture/equipment"
    else:
        return "car"
```

(Photo 21: Decision Tree Rule 1&2)

```
elif (credit_amount == 2978):
    if("Saving accounts" == "little"):
        return "business"
    else:
        return "furniture/equipment"
```

(Photo 22: Decision Tree Rule 3&4)

```
elif (credit_amount == 2028):
    if(number_of_job == 1):
        return "furniture/equipment"
    else:
        return "car"
```

(Photo 23: Decision Tree Rule 5&6)

```
else:
    #If target value has childs (if it has dictionary in it)
    #It means that there exists an nested dictionary, we need to
    #check other attributes for describing value
    #(In all other situations credit amount and
    #age is enough for getting target class)
    if(isinstance(decision_tree["Credit amount"][credit_amount],dict)):
        return decision_tree["Credit amount"][credit_amount]["Age"][age]

    #If target value has no child it means that
    #Credit amount is enough for getting target class
    else:
        return decision_tree["Credit amount"][credit_amount]
```

(Photo 24: Decision Tree Rule 7&8)

Hypothesis Accuracy

According to the test sample (which includes 300 data) the hypothesis accuracy is 92.66%.

```
Test Sample Size: 300
Number of correct predictions: 278
Accuracy(%): 92.66666666666666
```

(Photo 25: Accuracy Calculation with Python)

Project Github Repository

https://github.com/zThyphon/Data-Mining-Term-Project