Student name: Doniyor Musurmanov

Student ID: u3244564

Assessment: ST1 Capstone Project COLT1 2023

Date: 30/04/2023

**ST1 Capstone Project COLT1 2023**

**Credit Card Approval Predictor (GUI Application)**

* **TESTING:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| * **Test Case ID** | | ST1 Capstone Project | **Test Case Description** | | This is a Python code that creates a Graphical User Interface (GUI) using the tkinter module. The GUI consists of several frames, each containing various widgets such as labels, entry boxes, and option menus. The purpose of this GUI is to collect user inputs related to credit card approval prediction, such as gender, age, debt, marital status, industry, ethnicity, years employed, prior default, etc. The collected inputs are then passed to a credit card approval prediction model, which is imported from the cvd\_model module. | | | | | |
| **Created By** | | Doniyor | **Reviewed By** | |  | | **Version** | | 1.0 | |
|  |  |  |  |  |  |  |  |  |  |  |
| **QA Tester’s Log** | | N/A | | | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Tester's Name** | | Gurpreet | **Date Tested** | | 30.04.2023 | | **Test Case (Pass/Fail/Not Executed)** | | Pass | |
|  |  |  |  |  |  |  |  |  |  |  |
| **S #** | **Prerequisites (Precondition):** | | |  | **S #** | **Test Data** | | | | |
| 1 | The user must be informed to input the followings:  **Run the program** | | |  | 1 | For example, the user runs the program: | | | | |
| 2 | **'Gender':**  **'Age':**  **'Debt':**  **'Married':**  **'BankCustomer':**  **'Industry':**  **'Ethnicity':**  **'YearsEmployed':**  **'PriorDefault':**  **'Employed':**  **'CreditScore':**  **'DriversLicense':**  **'Citizen':**  **'ZipCode':**  **'Income':** | | |  | 2 | For example, the user inputs:  'Gender': Male  'Age': 30  'Debt': 0  'Married': Married  'BankCustomer': has a bank account  'Industry': Industrials  'Ethnicity': White  'YearsEmployed': 1  'PriorDefault': Yes  'Employed': Yes  'CreditScore': 1  'DriversLicense': No  'Citizen': By birth  'ZipCode': 202  'Income': 0 | | | | |
| 3 | **'Gender':**  **'Age':**  **'Debt':**  **'Married':**  **'BankCustomer':**  **'Industry':**  **'Ethnicity':**  **'YearsEmployed':**  **'PriorDefault':**  **'Employed':**  **'CreditScore':**  **'DriversLicense':**  **'Citizen':**  **'ZipCode':**  **'Income':** | | |  | 3 | For example, the user inputs:  'Gender': Male  'Age': 21  'Debt': 10  'Married': Single…  'BankCustomer': does not have a bank account  'Industry': Other  'Ethnicity': Black  'YearsEmployed': 1  'PriorDefault': No  'Employed': No  'CreditScore': 0  'DriversLicense': No  'Citizen': By birth  'ZipCode': 260  'Income': 0 | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Scenario** | Landscaping Cost Calculator |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step #** | **Step Details** | **Expected Results** | **Actual Results** | **Calculation** |
|
| 1 | **Run the program** | Algorithm Comparison  True label and Predicted label  Receiver Operating Characteristic  Credit card approval predictor app | As expected | Algorithm Comparison  True label and Predicted label  Receiver Operating Characteristic  Credit card approval predictor app |
| 2 | **'Gender':**  **'Age':**  **'Debt':**  **'Married':**  **'BankCustomer':**  **'Industry':**  **'Ethnicity':**  **'YearsEmployed':**  **'PriorDefault':**  **'Employed':**  **'CreditScore':**  **'DriversLicense':**  **'Citizen':**  **'ZipCode':**  **'Income':** | {{This prediction has an accuracy of:} 0.8913043478260869} {  1 - Your credit card is **approved**} | As expected | {{This prediction has an accuracy of:} 0.8913043478260869} {  1 - Your credit card is **approved**} |
| 3 | {{This prediction has an accuracy of:} 0.8623188405797102} {  } {0 - Your credit card is **not approved**} | As expected | {{This prediction has an accuracy of:} 0.8623188405797102} {  } {0 - Your credit card is **not approved**} |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Post-Condition** | The program ends |  |  |  |  |

* **The screenshots:**

|  |
| --- |
| **Test Case 1:** |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
| **Test Case 2:** |
|  |

|  |
| --- |
| **Test Case 3:** |
| **Graphical user interface, application  Description automatically generated** |

# Import the tkinter library and the cvd\_predictor function from the cvd\_model module  
import tkinter as tk  
from cvd\_model import \*  
  
  
# Define a class called CVD\_GUI for the credit card approval predictor application  
class CVD\_GUI:  
 def \_\_init\_\_(self):  
  
 # Create the main window for the GUI  
 self.main\_window = tk.Tk()  
 self.main\_window.title("CREDIT CARD APPROVAL PREDICTOR")  
  
 # Create 17 frames to group widgets  
 self.one\_frame = tk.Frame()  
 self.two\_frame = tk.Frame()  
 self.three\_frame = tk.Frame()  
 self.four\_frame = tk.Frame()  
 self.five\_frame = tk.Frame()  
 self.six\_frame = tk.Frame()  
 self.seven\_frame = tk.Frame()  
 self.eight\_frame = tk.Frame()  
 self.nine\_frame = tk.Frame()  
 self.ten\_frame = tk.Frame()  
 self.eleven\_frame = tk.Frame()  
 self.twelve\_frame = tk.Frame()  
 self.thirteen\_frame = tk.Frame()  
 self.fourteen\_frame = tk.Frame()  
 self.fifteen\_frame = tk.Frame()  
 self.sixteen\_frame = tk.Frame()  
 self.seventeen\_frame = tk.Frame()  
  
  
 # Create the widgets for one frame. (Title display)  
 self.title\_label = tk.Label(self.one\_frame, text='CREDIT CARD APPROVAL PREDICTOR',fg="green", font=("Helvetica", 18, 'bold'))  
 self.title\_label.pack()  
  
  
 # Create the widgets for two frame. (Gender input)  
 self.gender\_label = tk.Label(self.two\_frame, text='Gender:')  
 self.click\_gender\_var = tk.StringVar()  
 self.click\_gender\_var.set("Male")  
 self.gender\_inp = tk.OptionMenu(self.two\_frame,self.click\_gender\_var, "Male", "Female")  
 self.gender\_label.pack(side='left')  
 self.gender\_inp.pack(side='left')  
  
  
 # Create the widgets for three frame. (Age input)  
 self.age\_label = tk.Label(self.three\_frame, text='Age:')  
 self.age\_entry = tk.Entry(self.three\_frame, bg="white", fg="black", width = 10)  
 #self.age\_entry.insert(0,'50')  
 self.age\_label.pack(side='left')  
 self.age\_entry.pack(side='left')  
  
  
 # Create the widgets for four frame. (Debt input)  
 self.debt\_label = tk.Label(self.four\_frame, text='Debt:')  
 self.debt\_entry = tk.Entry(self.four\_frame, bg="white", fg="black")  
 #self.debt\_entry.insert(0,'150')  
 self.debt\_label.pack(side='left')  
 self.debt\_entry.pack(side='left')  
  
  
 # Create the widgets for five frame. (Married input)  
 self.married\_label = tk.Label(self.five\_frame, text='Married:')  
 self.click\_married\_var = tk.StringVar()  
 self.click\_married\_var.set("Single/Divorced/etc")  
 self.married\_inp = tk.OptionMenu(self.five\_frame, self.click\_married\_var, "Single/Divorced/etc", "Married")  
 self.married\_label.pack(side='left')  
 self.married\_inp.pack(side='left')  
  
   
 # Create the widgets for six frame. (Bank Customer input)  
 self.bank\_customer\_label = tk.Label(self.six\_frame, text='Bank Customer:')  
 self.click\_bank\_customer\_var = tk.StringVar()  
 self.click\_bank\_customer\_var.set("does not have a bank account")  
 self.bank\_customer\_inp = tk.OptionMenu(self.six\_frame, self.click\_bank\_customer\_var, "does not have a bank account", "has a bank account")  
 self.bank\_customer\_label.pack(side='left')  
 self.bank\_customer\_inp.pack(side='left')  
  
  
 # Create the widgets for seven frame. (Industry input)  
 self.industry\_label = tk.Label(self.seven\_frame, text='Industry:')  
 self.click\_industry\_var = tk.StringVar()  
 self.click\_industry\_var.set("Energy")  
 self.industry\_inp = tk.OptionMenu(self.seven\_frame, self.click\_industry\_var,"Energy", "Materials", "Industrials", "Consumer Discretionary", "Consumer Staples", "Other")  
 self.industry\_label.pack(side='left')  
 self.industry\_inp.pack(side='left')  
  
  
 # Create the widgets for eight frame. (Ethnicity input)  
 self.ethnicity\_label = tk.Label(self.eight\_frame, text='Ethnicity:')  
 self.click\_ethnicity\_var = tk.StringVar()  
 self.click\_ethnicity\_var.set("White")  
 self.ethnicity\_inp = tk.OptionMenu(self.eight\_frame, self.click\_ethnicity\_var,"White", "Black", "Asian", "Latino", "Other")  
 self.ethnicity\_label.pack(side='left')  
 self.ethnicity\_inp.pack(side='left')  
  
  
 # Create the widgets for nine frame. (Years Employed input)  
 self.years\_employed\_label = tk.Label(self.nine\_frame, text='Years Employed:')  
 self.years\_employed\_entry = tk.Entry(self.nine\_frame, bg="white", fg="black")  
 #self.years\_employed\_entry.insert(0,'100')  
 self.years\_employed\_label.pack(side='left')  
 self.years\_employed\_entry.pack(side='left')  
  
  
 # Create the widgets for ten frame. (Prior default input)  
 self.prior\_default\_label = tk.Label(self.ten\_frame, text='Prior default:')  
 self.click\_prior\_default\_var = tk.StringVar()  
 self.click\_prior\_default\_var.set("No")  
 self.prior\_default\_inp = tk.OptionMenu(self.ten\_frame, self.click\_prior\_default\_var, "No", "Yes")  
 self.prior\_default\_label.pack(side='left')  
 self.prior\_default\_inp.pack(side='left')  
  
  
 # Create the widgets for eleven frame. (Employed input)  
 self.employed\_label = tk.Label(self.eleven\_frame, text='Employed:')  
 self.click\_employed\_var = tk.StringVar()  
 self.click\_employed\_var.set("No")  
 self.employed\_inp = tk.OptionMenu(self.eleven\_frame, self.click\_employed\_var, "No", "Yes")  
 self.employed\_label.pack(side='left')  
 self.employed\_inp.pack(side='left')  
  
  
 # Create the widgets for twelve frame. (Credit Score input)  
 self.credit\_score\_label = tk.Label(self.twelve\_frame, text='Credit Score:')  
 self.credit\_score\_entry = tk.Entry(self.twelve\_frame, bg="white", fg="black")  
 #self.credit\_score\_entry.insert(0,'100')  
 self.credit\_score\_label.pack(side='left')  
 self.credit\_score\_entry.pack(side='left')  
  
  
 # Create the widgets for thirteen frame. (Drivers license input)  
 self.drivers\_license\_label = tk.Label(self.thirteen\_frame, text='Drivers license:')  
 self.click\_drivers\_license\_var = tk.StringVar()  
 self.click\_drivers\_license\_var.set("No")  
 self.drivers\_license\_inp = tk.OptionMenu(self.thirteen\_frame, self.click\_drivers\_license\_var, "No", "Yes")  
 self.drivers\_license\_label.pack(side='left')  
 self.drivers\_license\_inp.pack(side='left')  
  
  
 # Create the widgets for fourteen frame. (Citizen input)  
 self.citizen\_label = tk.Label(self.fourteen\_frame, text='Citizen:')  
 self.click\_citizen\_var = tk.StringVar()  
 self.click\_citizen\_var.set("By birth")  
 self.citizen\_inp = tk.OptionMenu(self.fourteen\_frame, self.click\_citizen\_var, "By birth", "By other means")  
 self.citizen\_label.pack(side='left')  
 self.citizen\_inp.pack(side='left')  
  
  
 # Create the widgets for fifteen frame. (Zip Code input)  
 self.zip\_code\_label = tk.Label(self.fifteen\_frame, text='Zip Code:')  
 self.zip\_code\_entry = tk.Entry(self.fifteen\_frame, bg="white", fg="black")  
 #self.zip\_code\_entry.insert(0,'100')  
 self.zip\_code\_label.pack(side='left')  
 self.zip\_code\_entry.pack(side='left')  
  
  
 # Create the widgets for sixteen frame. (Income input)  
 self.income\_label = tk.Label(self.sixteen\_frame, text='Income:')  
 self.income\_entry = tk.Entry(self.sixteen\_frame, bg="white", fg="black")  
 #self.income\_entry.insert(0,'100')  
 self.income\_label.pack(side='left')  
 self.income\_entry.pack(side='left')  
  
  
  
 #Create the widgets for seventeen frame = cca (Prediction of Credit Card Approval)  
 self.cca\_predict\_ta = tk.Text(self.seventeen\_frame,height = 10, width = 25,bg= 'light blue')  
  
 #Create predict button and quit button  
 self.btn\_predict = tk.Button(self.seventeen\_frame, text='Predict Credit Card Approval', command=self.predict\_cca)  
 self.btn\_quit = tk.Button(self.seventeen\_frame, text='Quit', command=self.main\_window.destroy)  
  
  
 self.cca\_predict\_ta.pack(side='left')  
 self.btn\_predict.pack()  
 self.btn\_quit.pack()  
  
 # Pack the frames.  
 self.one\_frame.pack()  
 self.two\_frame.pack()  
 self.three\_frame.pack()  
 self.four\_frame.pack()  
 self.five\_frame.pack()  
 self.six\_frame.pack()  
 self.seven\_frame.pack()  
 self.eight\_frame.pack()  
 self.nine\_frame.pack()  
 self.ten\_frame.pack()  
 self.eleven\_frame.pack()  
 self.twelve\_frame.pack()  
 self.thirteen\_frame.pack()  
 self.fourteen\_frame.pack()  
 self.fifteen\_frame.pack()  
 self.sixteen\_frame.pack()  
 self.seventeen\_frame.pack()  
  
  
 # Enter the tkinter main loop.  
 tk.mainloop()  
  
 def predict\_cca(self):  
 result\_string = ""  
  
 self.cca\_predict\_ta.delete(0.0, tk.END)  
  
 creditor\_gender = self.click\_gender\_var.get()  
 if(creditor\_gender == "Male"):  
 creditor\_gender = 1  
 else:  
 creditor\_gender = 0  
  
 creditor\_age = self.age\_entry.get()  
 creditor\_debt = self.debt\_entry.get()  
  
 creditor\_married = self.click\_married\_var.get()  
 if(creditor\_married == "Single/Divorced/etc"):  
 creditor\_married = 0  
 else:  
 creditor\_married = 1  
  
 creditor\_bank\_customer = self.click\_bank\_customer\_var.get()  
 if(creditor\_bank\_customer == "does not have a bank account"):  
 creditor\_bank\_customer = 0  
 else:  
 creditor\_bank\_customer = 1  
  
 creditor\_industry = self.click\_industry\_var.get()  
 if(creditor\_industry == "Energy"):  
 creditor\_industry = 0  
 elif(creditor\_industry == "Materials"):  
 creditor\_industry = 1  
 elif(creditor\_industry == "Industrials"):  
 creditor\_industry = 2  
 elif(creditor\_industry == "Consumer Discretionary"):  
 creditor\_industry = 3  
 else:  
 creditor\_industry = 4  
  
 creditor\_ethnicity = self.click\_ethnicity\_var.get()  
 if(creditor\_ethnicity == "White"):  
 creditor\_ethnicity = 0  
 elif(creditor\_ethnicity == "Black"):  
 creditor\_ethnicity = 1  
 elif(creditor\_ethnicity == "Asian"):  
 creditor\_ethnicity = 2  
 elif(creditor\_ethnicity == "Latino"):  
 creditor\_ethnicity = 3  
 else:  
 creditor\_ethnicity = 4  
  
  
 creditor\_years\_employed = self.years\_employed\_entry.get()  
  
 creditor\_prior\_default = self.click\_prior\_default\_var.get()  
 if(creditor\_prior\_default == "Yes"):  
 creditor\_prior\_default = 1  
 else:  
 creditor\_prior\_default = 0  
  
 creditor\_employed = self.click\_employed\_var.get()  
 if(creditor\_employed == "Yes"):  
 creditor\_employed = 1  
 else:  
 creditor\_employed = 0  
  
 creditor\_credit\_score = self.credit\_score\_entry.get()  
  
 creditor\_drivers\_license = self.click\_drivers\_license\_var.get()  
 if(creditor\_drivers\_license == "Yes"):  
 creditor\_drivers\_license = 1  
 else:  
 creditor\_drivers\_license = 0  
  
 creditor\_citizen = self.click\_citizen\_var.get()  
 if(creditor\_citizen == "By birth"):  
 creditor\_citizen = 1  
 else:  
 creditor\_citizen = 0  
  
 creditor\_zip\_code = self.zip\_code\_entry.get()  
 creditor\_income = self.income\_entry.get()  
  
  
  
  
 result\_string += "===creditor Diagnosis=== \n"  
 creditor\_info = (creditor\_gender,creditor\_age,creditor\_debt,creditor\_married,creditor\_bank\_customer, creditor\_industry, creditor\_ethnicity,creditor\_years\_employed,creditor\_prior\_default, creditor\_employed, creditor\_credit\_score, creditor\_drivers\_license, creditor\_citizen, creditor\_zip\_code, creditor\_income)  
  
  
 cca\_prediction = best\_model.predict([creditor\_info])  
 disp\_string = ("This prediction has an accuracy of:", str(model\_accuracy))  
  
 result = cca\_prediction  
  
 if(cca\_prediction == [0]):  
 result\_string = (disp\_string, '\n', "0 - Your credit card is not approved")  
 else:  
 result\_string = (disp\_string, '\n'+ "1 - Your credit card is approved")  
 self.cca\_predict\_ta.insert('1.0',result\_string)  
  
 # Predicted: 1 Actual: 1 Data: (1, 30, 0, 1, 1, 2, 0, 1, 1, 1, 1, 0, 0, 202, 0)  
 # Predicted: 0 Actual: 0 Data: (1, 21, 10, 0, 0, 5, 1, 1, 0, 0, 0, 0, 0, 260, 0)  
  
  
  
my\_cvd\_GUI = CVD\_GUI()