1.COMPANY PROFILE

COMPANY NAME:-KARUNADU TECHNOLOGIES PRIVATE LIMITED

TYPE: PRIVATELY HELD

COMPANY CATEGORY: PRODUCT/SERVICE BASED

Karunadu Technologies Private Limited offers a wide range of Embedded Design Services that are intended towards transforming an idea into a complete product.

Vision and Mission of the company is to empower unskilled individual with knowledge, skills and technical competencies in the field of information technology and embedded engineering which assist them to escalate as integrate individuals contributing to company's and Nation's growth. Provide cost effective and reliable solutions to customers across latest technologies. and offer scalable end-to-end application and management solutions and provide cost effective highly scalable products for varied verticals

Products of the companies: IOT Development Board , FPGA Development Board , IR Automation Board, Advance Security system, Time Tracking and Billing and Expense Tracker

2.Purpose & Objectives

Purpose:-

1. A bank’s profit or a loss depends to a large extent on loans i.e. whether the customers are paying back the loan or defaulting. By predicting the loan defaulters, the bank can reduce its non-performing assets. This makes the study of this phenomenon very important.

2. The company wants to automate the loan eligibility process (real-time) based on customer detail provided while filling out online application forms. These details are Gender, Marital Status, Education, number of Dependents, Income, Loan Amount, Credit History, and others.

3. To automate this process, they have provided a dataset to identify the customer segments that are eligible for loan amounts so that they can specifically target these customers.

Objectives:-

Main objective of this project includes:

1. Speedy

2. Immediate

3. Simple approach to sanction the loan.

3. Tools/Technology Used

Python: -

Figure:-3.1

Python is an interpreted high-level programming language for general-purpose programming.

Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace

Python Tool Used:-

ANACONDA DISTRIBUTION:-The world's most popular open-source Python distribution platform.

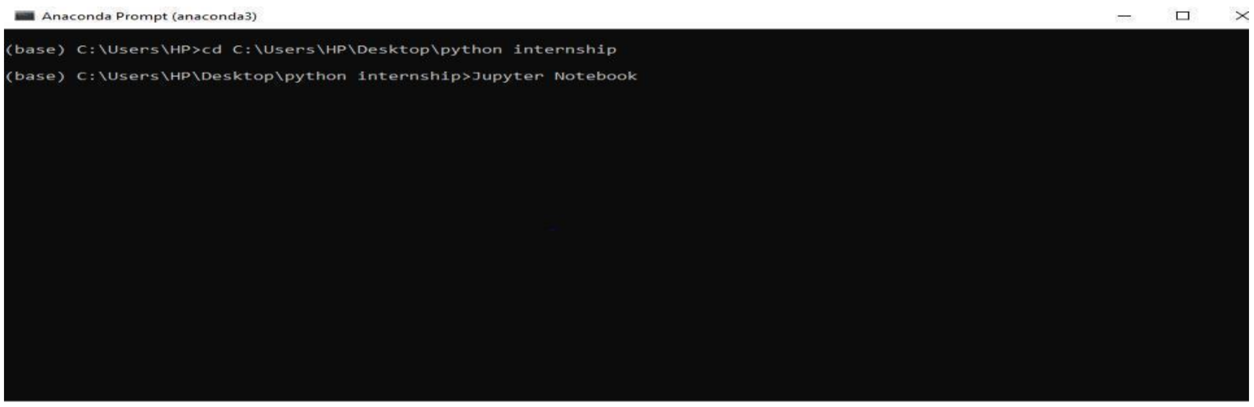
1. Open Source:- Access the open-source software you need for projects in any field, from data visualization to robotics.
2. User-friendly:-With our intuitive platform, you can easily search and install packages and create, load, and switch between environments.
3. Trusted:-Our securely hosted packages and artifacts are methodically tested and regularly updated.

Features of Anaconda:

<u>Jupyter Notebook:</u>

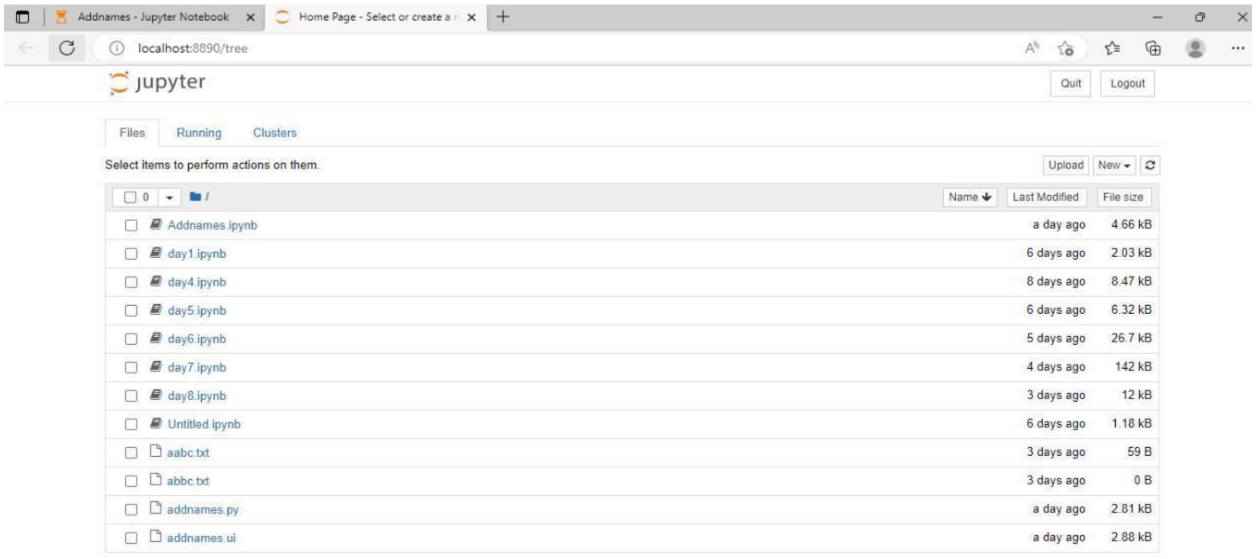
Figure:-3.2

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.



```
(base) C:\Users\HP>cd C:\Users\HP\Desktop\python internship
(base) C:\Users\HP\Desktop\python internship>Jupyter Notebook
```

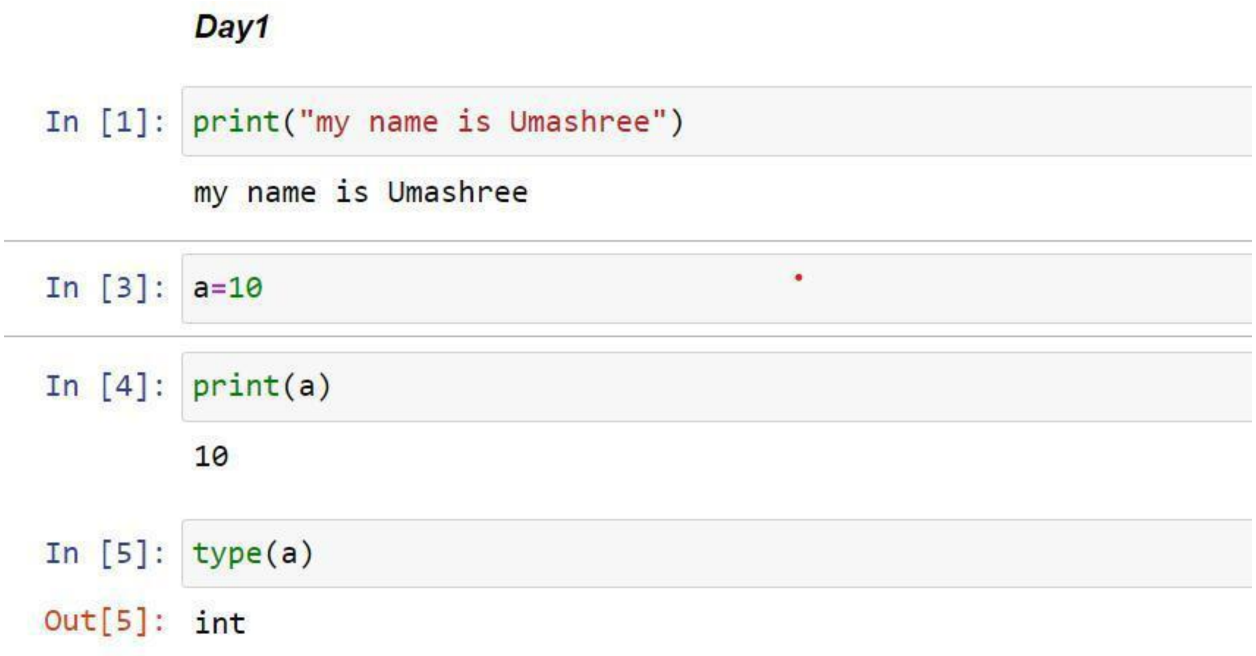
Figure:-3.3



Select items to perform actions on them.

Name	Last Modified	File size
Addnames.ipynb	a day ago	4.66 kB
day1.ipynb	6 days ago	2.03 kB
day4.ipynb	8 days ago	8.47 kB
day5.ipynb	6 days ago	6.32 kB
day6.ipynb	5 days ago	26.7 kB
day7.ipynb	4 days ago	142 kB
day8.ipynb	3 days ago	12 kB
Untitled.ipynb	6 days ago	1.18 kB
aabc.txt	3 days ago	59 B
abbc.txt	3 days ago	0 B
addnames.py	a day ago	2.81 kB
addnames.ui	a day ago	2.88 kB

Figure:-3.4



```
Day1

In [1]: print("my name is Umashree")

        my name is Umashree

In [3]: a=10

In [4]: print(a)

        10

In [5]: type(a)

Out[5]: int
```

Figure:-3.5

Jupyter Lab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning.

DATA SCIENCE:-

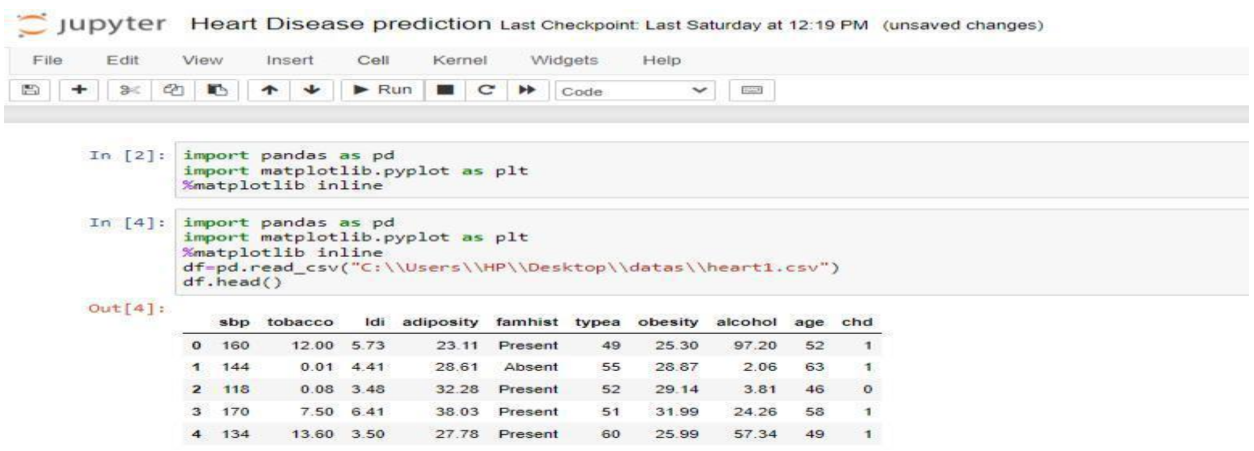
Figure:-3.6

“Data science” is just about as broad of a term as they come. It may be easiest to describe what it is by listing its more concrete components:

1) Data exploration & analysis:-. Included here: Pandas; NumPy; SciPy; a helping hand from Python’s Standard Library.

2) Data visualization:- A pretty self-explanatory name. Taking data and turning it into something colorful. Included here: Matplotlib; Seaborn; Datashader; others.

3) Classical machine learning:- Conceptually, we could define this as any supervised or unsupervised learning task that is not deep learning (see below). Included here: Scikit-Learn, StatsModels.


jupyter Heart Disease prediction Last Checkpoint: Last Saturday at 12:19 PM (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help
Run
Code

```
In [2]: import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
```

```
In [4]: import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
        df=pd.read_csv("C:\\Users\\HP\\Desktop\\datas\\heart1.csv")
        df.head()
```

```
Out[4]:
   sbp  tobacco   ldi  adiposity  famhist  typea  obesity  alcohol  age  chd
0  160    12.00  5.73      23.11  Present     49    25.30    97.20   52    1
1  144     0.01  4.41      28.61   Absent     55    28.87     2.06   63    1
2  118     0.08  3.48      32.28  Present     52    29.14     3.81   46    0
3  170     7.50  6.41      38.03  Present     51    31.99    24.26   58    1
4  134    13.60  3.50      27.78  Present     60    25.99    57.34   49    1
```

Figure:-3.7

MACHINE LEARNING:-

Figure:-3.8

Machine learning is a subset of artificial intelligence in the field of computer science that often uses statistical techniques to give computers the ability to "learn" (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed.

Various Python libraries used in the project:-

Pandas:- Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

Figure:-3.9

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc. In this tutorial, we will learn the various features of Python Pandas and how to use them in practice.

Machine Learning Algorithms:-

1. <u>Linear Regression</u>-

Linear regressions one of the supervised Machine learning algorithms in Python that observes continuous features and predicts an outcome. Depending on whether it runs on a single variable or on many features, we can call it simple linear regression or multiple linear regression.

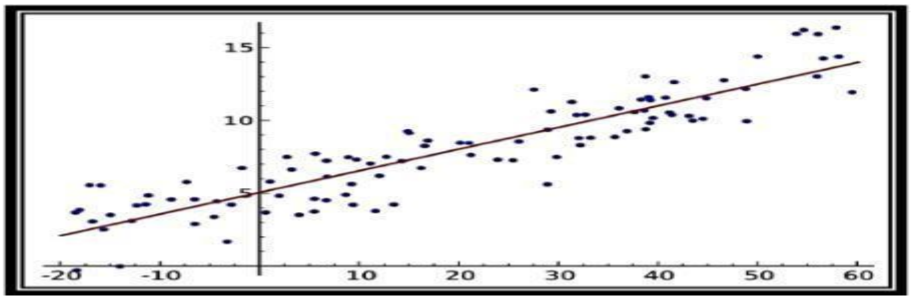


Figure:-3.10

2. <u>Logistic Regression</u> -Logistic regression is a supervised classification is unique Machine Learning algorithms in Python that finds its use in estimating discrete values like 0/1, yes/no, and true/false.

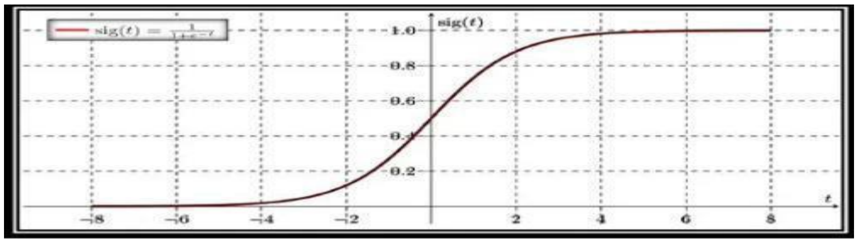


Figure:-3.11

3<u>. Decision Tree</u> –A decision tree falls under supervised Machine Learning Algorithms in Python and comes of use for both classification and regression- although mostly for classification.

This model takes an instance, traverses the tree, and compares important features with a determined conditional statement.

4. <u>Support Vector Machine (SVM):-</u>

SVM is a supervised classification is one of the most important Machines Learning algorithms in Python, that plots a line that divides different categories of your data. In this ML algorithm, we calculate the vector to optimize the line

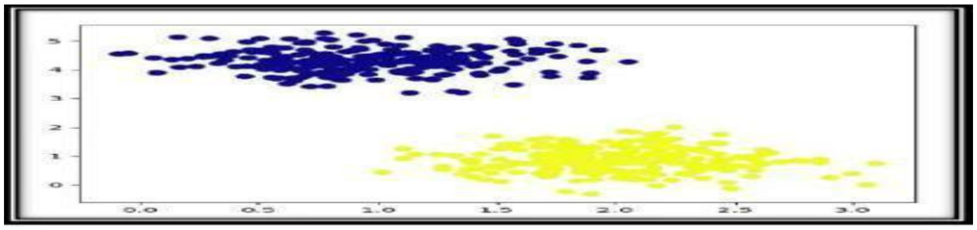


Figure:-3.12

5. K-Means Algorithm -

k-Means is an unsupervised algorithm that solves the problem of clustering. It classifies data using a number of clusters. The data points inside a class are homogeneous and heterogeneous to peer groups. k -means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining.

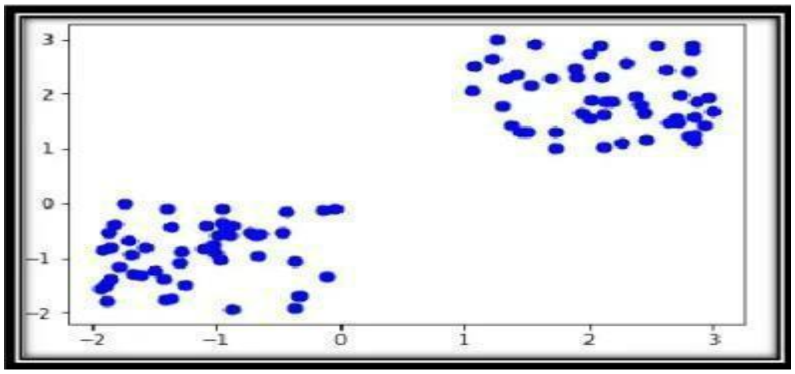


Figure:-3.13

4. Code Development

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'loansanction.ui'

# Created by: PyQt5 UI code generator 5.15.4

# WARNING: Any manual changes made to this file will be lost when pyuic5 is

# run again. Do not edit this file unless you know what you are doing.

from PyQt5 import QtCore, QtGui, QtWidgets

class Ui_Form(object):

    def setupUi(self, Form):

        Form.setObjectName("Form")

        Form.resize(1548, 870)

        Form.setStyleSheet("color: rgb(7, 7, 7);\n"

"background-color: rgb(255, 255, 255);\n"

"font: 16pt \"MS Shell Dlg 2\";") self.prediction =

        QtWidgets.QLabel(Form)

        self.prediction.setGeometry(QtCore.QRect(210, 0, 711,

        61)) self.prediction.setStyleSheet("color: rgb(3, 3, 3);\n"

"background-color: rgb(255, 255, 255);\n"

"font: 28pt \"MS Shell Dlg 2\";")

        self.prediction.setObjectName("prediction")

        self.gender = QtWidgets.QLabel(Form)

        self.gender.setGeometry(QtCore.QRect(170, 150, 121,

        31)) self.gender.setStyleSheet("color: rgb(0, 0, 0);\n"
```

```
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
        self.gender.setObjectName("gender") self.married =
        QtWidgets.QLabel(Form)
        self.married.setGeometry(QtCore.QRect(160, 210, 131, 31))
        self.married.setStyleSheet("color: rgb(11, 11, 11);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
        self.married.setObjectName("married")
        self.dependents = QtWidgets.QLabel(Form)
        self.dependents.setGeometry(QtCore.QRect(120, 270, 181, 41))
        self.dependents.setStyleSheet("color: rgb(12, 12, 12);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
        self.dependents.setObjectName("dependents")
        self.education = QtWidgets.QLabel(Form)
        self.education.setGeometry(QtCore.QRect(140, 340, 161, 41))
        self.education.setStyleSheet("color: rgb(4, 4, 4);\n"
"font: 16pt \"MS Shell Dlg 2\";\n"
"background-color: rgb(255, 255, 255);")
        self.education.setObjectName("education")
        self.self_employed = QtWidgets.QLabel(Form)
        self.self_employed.setGeometry(QtCore.QRect(80, 410, 221, 41))
        self.self_employed.setStyleSheet("color: rgb(12, 12, 12);\n"
"background-color: rgb(255, 255, 255);\n"
```

```
"font: 16pt \"MS Shell Dlg 2\";")
        self.self_employed.setObjectName("self_employed")
        self.applicantincome = QtWidgets.QLabel(Form)
        self.applicantincome.setGeometry(QtCore.QRect(60, 480, 251, 31))
        self.applicantincome.setStyleSheet("color: rgb(6, 6, 6);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
        self.applicantincome.setObjectName("applicantincome")
        self.coapplicantincome = QtWidgets.QLabel(Form)
        self.coapplicantincome.setGeometry(QtCore.QRect(30, 530, 291, 41))
        self.coapplicantincome.setStyleSheet("color: rgb(7, 7, 7);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
        self.coapplicantincome.setObjectName("coapplicantincome")
        self.loanamount = QtWidgets.QLabel(Form)
        self.loanamount.setGeometry(QtCore.QRect(120, 600, 191, 41))
        self.loanamount.setStyleSheet("color: rgb(4, 4, 4);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
        self.loanamount.setObjectName("loanamount")
        self.loan_amount_term = QtWidgets.QLabel(Form)
        self.loan_amount_term.setGeometry(QtCore.QRect(20, 670, 291, 41))
        self.loan_amount_term.setStyleSheet("color: rgb(11, 11, 11);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
```

```
self.loan_amount_term.setObjectName("loan_amount_term")
self.credit_history = QtWidgets.QLabel(Form)
self.credit_history.setGeometry(QtCore.QRect(70, 750, 231, 41))
self.credit_history.setObjectName("credit_history")
self.property_loan = QtWidgets.QLabel(Form)
self.property_loan.setGeometry(QtCore.QRect(640, 100, 221, 41))
self.property_loan.setObjectName("property_loan")
self.loan_status = QtWidgets.QLabel(Form)
self.loan_status.setGeometry(QtCore.QRect(670, 170, 191, 51))
self.loan_status.setObjectName("loan_status")
self.gender_2 = QtWidgets.QLineEdit(Form)
self.gender_2.setGeometry(QtCore.QRect(300, 150, 261, 41))
self.gender_2.setObjectName("gender_2")
self.married_2 = QtWidgets.QLineEdit(Form)
self.married_2.setGeometry(QtCore.QRect(300, 210, 261, 41))
self.married_2.setObjectName("married_2")
self.dependents_2 = QtWidgets.QLineEdit(Form)
self.dependents_2.setGeometry(QtCore.QRect(300, 270, 251, 41))
self.dependents_2.setObjectName("dependents_2")
self.education_2 = QtWidgets.QLineEdit(Form)
self.education_2.setGeometry(QtCore.QRect(310, 340, 261, 41))
self.education_2.setObjectName("education_2")
self.self_employed_2 = QtWidgets.QLineEdit(Form)
self.self_employed_2.setGeometry(QtCore.QRect(310, 410, 261, 41))
self.self_employed_2.setObjectName("self_employed_2")
self.applicantincome_2 = QtWidgets.QLineEdit(Form)
self.applicantincome_2.setGeometry(QtCore.QRect(310, 480, 261, 41))
self.applicantincome_2.setObjectName("applicantincome_2")
self.coapplicantincome_2 = QtWidgets.QLineEdit(Form)
self.coapplicantincome_2.setGeometry(QtCore.QRect(320, 540, 261, 41))
self.coapplicantincome_2.setObjectName("coapplicantincome_2")
self.loanamount_2 = QtWidgets.QLineEdit(Form)
self.loanamount_2.setGeometry(QtCore.QRect(320, 601, 261, 41))
self.loanamount_2.setObjectName("loanamount_2")
self.loan_amount_term_2 = QtWidgets.QLineEdit(Form)
self.loan_amount_term_2.setGeometry(QtCore.QRect(320, 671, 261, 41))
self.loan_amount_term_2.setObjectName("loan_amount_term_2")
self.credit_history_2 = QtWidgets.QLineEdit(Form)
self.credit_history_2.setGeometry(QtCore.QRect(320, 750, 281, 41))
self.credit_history_2.setObjectName("credit_history_2")
self.property_loan_2 = QtWidgets.QLineEdit(Form)
self.property_loan_2.setGeometry(QtCore.QRect(882, 100, 201, 41))
self.property_loan_2.setObjectName("property_loan_2")
self.loan_status_2 = QtWidgets.QLineEdit(Form)
self.loan_status_2.setGeometry(QtCore.QRect(872, 180, 201, 41))
self.loan_status_2.setObjectName("loan_status_2")
self.submit = QtWidgets.QPushButton(Form)
self.submit.setGeometry(QtCore.QRect(820, 330, 151, 61))
self.submit.setObjectName("submit")
self.submit.clicked.connect(self.pred)
```

```
        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)


    def pred(self):
        gen=self.gender_2.text()
        mar=self.married_2.text()
        dep=self.dependents_2.text()
        edu=self.education_2.text()
        se=self.self_employed_2.text()
        ai=self.applicantincome_2.text()
        ci=self.coapplicantincome_2.text()
        la=self.loanamount_2.text()
        lat=self.loan_amount_term_2.text()
        ch=self.credit_history_2.text()
        pl=self.property_loan_2.text()
        import pandas as pd
        path="C:\\Users\\mithu\\OneDrive\\Desktop\\Data\\train.csv"
        data=pd.read_csv(path)
        import sklearn
        from sklearn.preprocessing import LabelEncoder
        le_Gender=LabelEncoder()
        le_Dependents=LabelEncoder()
        le_Loan_Status=LabelEncoder()
        le_Married=LabelEncoder()
        le_Education=LabelEncoder()
```

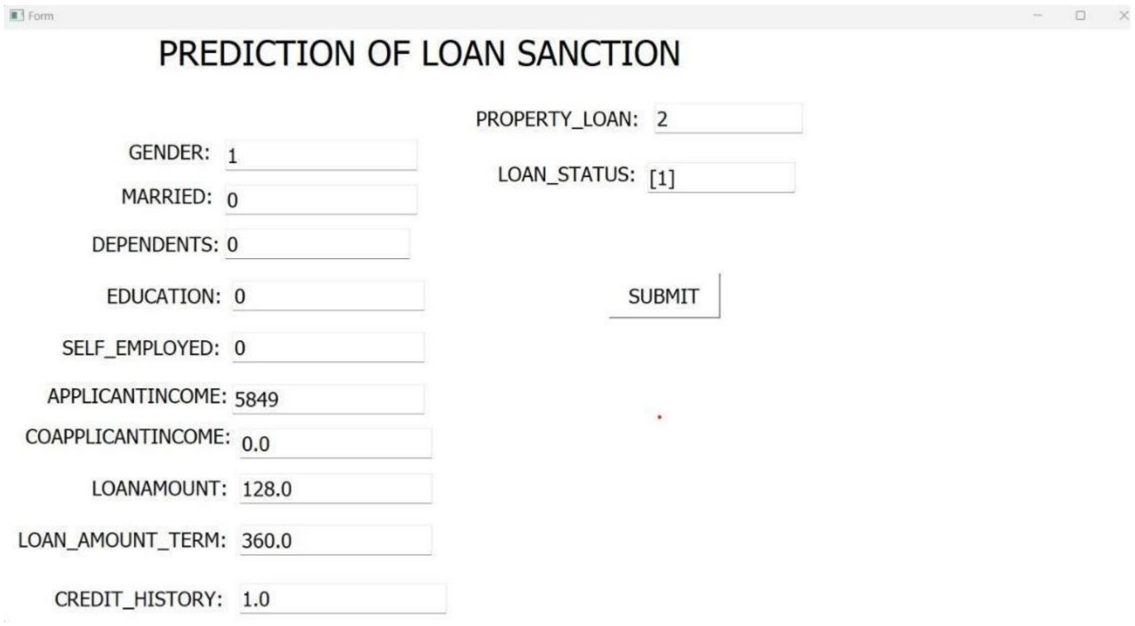
```
le_Self_Employed=LabelEncoder()
le_Property_Area=LabelEncoder()
print(data['Gender'].unique())
print(data['Dependents'].unique())
print(data['Loan_Status'].unique())
print(data['Married'].unique())
print(data['Education'].unique())
print(data['Self_Employed'].unique())
print(data['Property_Area'].unique())
print(data)

data['Gender_n']=le_Gender.fit_transform(data['Gender'])
data['Dependents_n']=le_Gender.fit_transform(data['Dependents'])
data['Loan_Status_n']=le_Loan_Status.fit_transform(data['Loan_Status'])
data['Married_n']=le_Loan_Status.fit_transform(data['Married'])
data['Education_n']=le_Loan_Status.fit_transform(data['Education'])
data['Self_Employed_n']=le_Loan_Status.fit_transform(data['Self_Employed'])
data['Property_Area_n']=le_Loan_Status.fit_transform(data['Property_Area'])
data['Gender']=data['Gender_n']
data['Dependents']=data['Dependents_n']
data['Loan_Status']=data['Loan_Status_n']
data['Married']=data['Married_n']
data['Education']=data['Education_n']
data['Self_Employed']=data['Self_Employed_n']
```

```
data['Property_Area']=data['Property_Area_n']

print(data)

medianvals=data.Gender.median()

print(medianvals)

data.Gender=data.Gender.fillna(medianvals)

medianval=data.LoanAmount.median()

print(medianval)

data.LoanAmount=data.LoanAmount.fillna(medianval)

medianvall=data.Self_Employed.median()

print(medianvall)

data.Self_Employed=data.Self_Employed.fillna(medianvall)

medianvalat=data.Loan_Amount_Term.median()

print(medianvalat)

data.Loan_Amount_Term=data.Loan_Amount_Term.fillna(medianvalat)

medianvalch=data.Credit_History.median()

print(medianvalch)

data.Credit_History=data.Credit_History.fillna(medianvalch)
print(data)
output=data.drop(['Loan_ID','Gender','Married','Dependents','Education','Self_Employed','ApplicantIncome','CoapplicantIncome','LoanAmount','Loan_Amount_Term','Credit_History','Property_Area','Gender_n','Dependents_n','Loan_Status_n','Married_n','Education_n','Self_Employed_n','Property_Area_n','Loan_Status_n'],'columns')
inputs=data.drop(['Loan_ID','Loan_Status','Gender_n','Loan_Status_n','Married_n','Education_n','Self_Employed_n','Property_Area_n','Loan_Status_n','Dependents_n'],'columns')
print(output)

print(inputs)

import sklearn
```

```
        from sklearn.model_selection import train_test_split
        x_train,x_test,y_train,y_test=train_test_split(inputs,output,train_size=0.8)
        print(x_train)
        print(x_test)
        print(y_train)
        print(y_test)
        from sklearn.linear_model import LogisticRegression
        model=LogisticRegression()
        model.fit(x_train,y_train)
        y_pred=model.predict(x_test)
        print(y_pred)
        print(x_test)
        from sklearn.metrics import confusion_matrix
        cm=confusion_matrix(y_test,y_pred)
        print(cm)
        res=model.predict([[gen,mar,dep,edu,se,ai,ci,la,lat,ch,pl]])
        print(res)
        self.loan_status_2.setText(str(res))
    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.prediction.setText(_translate("Form", "PREDICTION OF LOAN SANCTION"))
        self.gender.setText(_translate("Form", "GENDER:"))
        self.married.setText(_translate("Form", "MARRIED:"))
        self.dependents.setText(_translate("Form", "DEPENDENTS:"))
        self.education.setText(_translate("Form", "EDUCATION:"))
        self.self_employed.setText(_translate("Form", "SELF_EMPLOYED:"))
        self.applicantincome.setText(_translate("Form", "APPLICANTINCOME:"))
        self.coapplicantincome.setText(_translate("Form", "COAPPLICANTINCOME:"))
        self.loanamount.setText(_translate("Form", "LOANAMOUNT:"))
        self.loan_amount_term.setText(_translate("Form", "LOAN_AMOUNT_TERM:"))
        self.credit_history.setText(_translate("Form", "CREDIT_HISTORY:"))
        self.property_loan.setText(_translate("Form", "PROPERTY_LOAN:"))
        self.loan_status.setText(_translate("Form", "LOAN_STATUS:"))
        self.submit.setText(_translate("Form", "SUBMIT"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()
    ui.setupUi(Form)
    Form.show()
    sys.exit(app.exec_())
```

4.1.OUTPUT

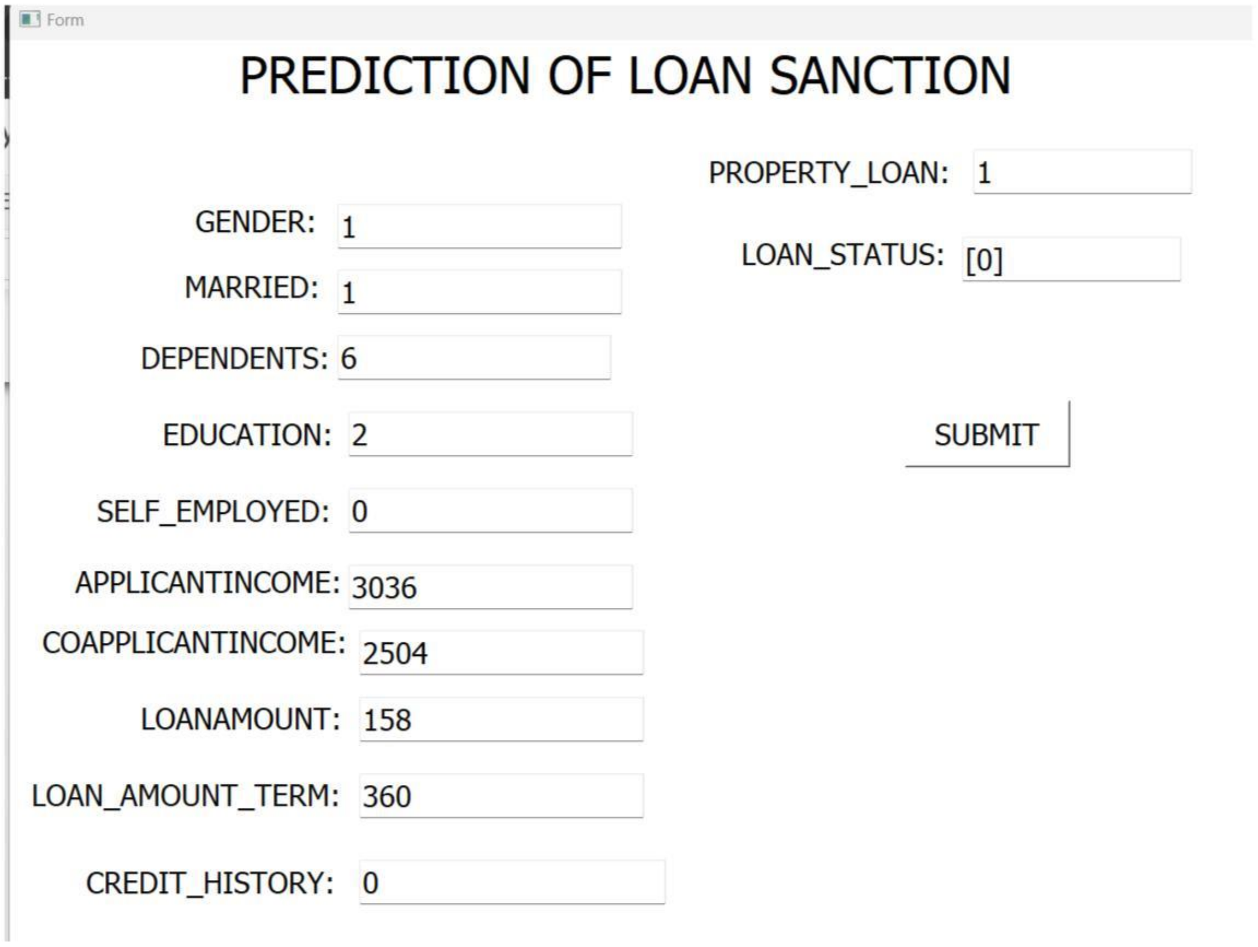
OUTPUT



PREDICTION OF LOAN SANCTION

GENDER:	1
MARRIED:	0
DEPENDENTS:	0
EDUCATION:	0
SELF_EMPLOYED:	0
APPLICANTINCOME:	5849
COAPPLICANTINCOME:	0.0
LOANAMOUNT:	128.0
LOAN_AMOUNT_TERM:	360.0
CREDIT_HISTORY:	1.0
PROPERTY_LOAN:	2
LOAN_STATUS:	[1]

Figure:-4.1.1



PREDICTION OF LOAN SANCTION

GENDER:	1
MARRIED:	1
DEPENDENTS:	6
EDUCATION:	2
SELF_EMPLOYED:	0
APPLICANTINCOME:	3036
COAPPLICANTINCOME:	2504
LOANAMOUNT:	158
LOAN_AMOUNT_TERM:	360
CREDIT_HISTORY:	0
PROPERTY_LOAN:	1
LOAN_STATUS:	[0]

Figure:-4.1.2

5.SELF LEARNING

1. Information processing.
2. Teamwork.
3. Planning/prioritizing.
4. Decision making/problem solving and
5. Verbal communication.

Self-Learning Starts With Planning
First, you have to plan and decide on what would add the most value to you. For example, if you already have a career goal in mind, think of the skills required and either learn a new one, or hone existing ones.

1.If you don't already have a career in mind, I urge you to decide on a job you're open to pursue and pick up some relevant skills. In the process, you'll also learn whether you enjoy the work, which is still a valuable outcome.

2. Read Widely and In Depth, Take Notes Read books relevant to your interests and gather different perspectives. You'll be surprised; soft skills can be learnt through books too.

3. Online Courses and Websites Utilise the free courses available. I share methods to make use of existing deals to obtain free certification.

4. OpenCourseWare Reputable Universities have made their lectures, assignments, and quizzes available. Some resources for you.

5. Watch YouTube (Seriously!)YouTube videos can be a source of valuable information and guidance.

6. Own a Project :-Projects are the best way to practice and showcase your skills, be it related to finance, software engineering, data science, communications, marketing and more.

7. Participate In Relevant Competitions:-There are ongoing competitions that you can compete in and gain recognition for.

6.Conclusion

I believe the trial has shown conclusively that it is both possible and desirable to use python as the principal teaching language:

It is a flexible tool that allows both the teaching of traditional procedural programmimg and modern OOP. It can be used to teach a large number of transferable skills.It is a real-world programming language and is used in academia and commercial world.It is free as in both cost and source code."

The practical implementation of Python in machine learning projects and tasks has made the work easier for developers, data scientists, and machine learning engineers. Python can be easily used to analyze and compose available data, which also makes it one of the most popular languages in data science.

7.REFERENCE

[1] Karunaduprivatelimited(2022)@Harish

[2] https://www.abacademies.org/articles/prediction-of-sales-based-on-an-effective-advertising-media-sale-data-a-python-implementation-approach-11437.html

[3]https://www.cse.scu.edu/~m1wang/projects/Mining_advertisementClickThroughRatePrediction_18s.pdf

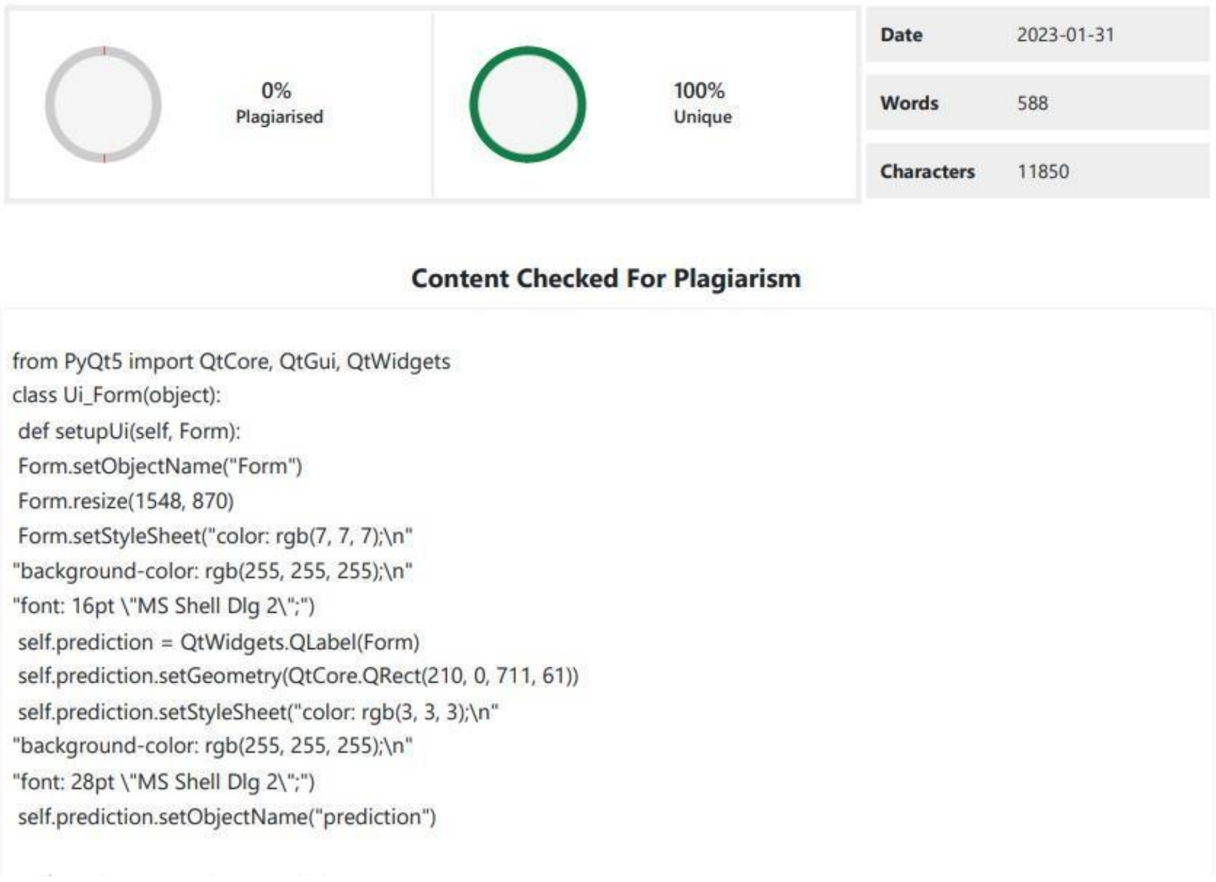
[4] https://medium.com/@kongbayan/linear-regression-sales-prediction-from-advertising-platform-using-python-ac60b07aab4f

[5]Prediction of sales based on an effective advertisement media sales data :a python implementation approach-Academy of Strategic Management Journal. Volume 20, issue 2,2021.

[6]https://www.researchgate.net/publication/350398584_COMMERCIALS_SALES_PREDICTION_USING_MULTIPLE_LINEAR_REGRESSION

[7] https://medium.com/@kongbayan/linear-regression-sales-prediction-from-advertising-platform-using-python-ac60b07aab4f

8.PLAGIARISM REPORT



0%
Plagiarised

100%
Unique

Date	2023-01-31
Words	588
Characters	11850

Content Checked For Plagiarism

```
from PyQt5 import QtCore, QtGui, QtWidgets
class Ui_Form(object):
 def setupUi(self, Form):
 Form.setObjectName("Form")
 Form.resize(1548, 870)
 Form.setStyleSheet("color: rgb(7, 7, 7);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 16pt \"MS Shell Dlg 2\";")
 self.prediction = QtWidgets.QLabel(Form)
 self.prediction.setGeometry(QtCore.QRect(210, 0, 711, 61))
 self.prediction.setStyleSheet("color: rgb(3, 3, 3);\n"
"background-color: rgb(255, 255, 255);\n"
"font: 28pt \"MS Shell Dlg 2\";")
 self.prediction.setObjectName("prediction")
```