PROFESSIONAL TRAINING REPORT

at

Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfillment of the requirements for the

award of

B.Tech (Bachelor in Technology) Bachelor of Engineering Degree
in
Information Technology

By

AKULA SRAVANTH SATYA RAMKI
REG. NO. 39120119

DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

NOVEMBER 2021

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI– 600119
www.sathyabama.ac.in

DEPARTMENT OF INFORMATION TECHNOLGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **AKULA SRAVANTH SATYA RAMKI(Reg. No: 39120119)** who carried out the project entitled “**South German Credit based on machine learning**” under my supervision from June 2021 to November2021.

Internal Guide

Dr. R.M. Gomathi, M.Tech., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on____________________

InternalExaminer **ExternalExaminer**

DECLARATION

I , **A.S.S.RAMKI** hereby declare that the project report entitled "**South German Credit based on machine learning"** done by me under the guidance of **Dr. R.M. Gomathi, M.Tech., Ph.D.,** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Information technology.

DATE:

PLACE: **SIGNATURE OF THECANDIDATE**

ACKNOWLEDGEMENT

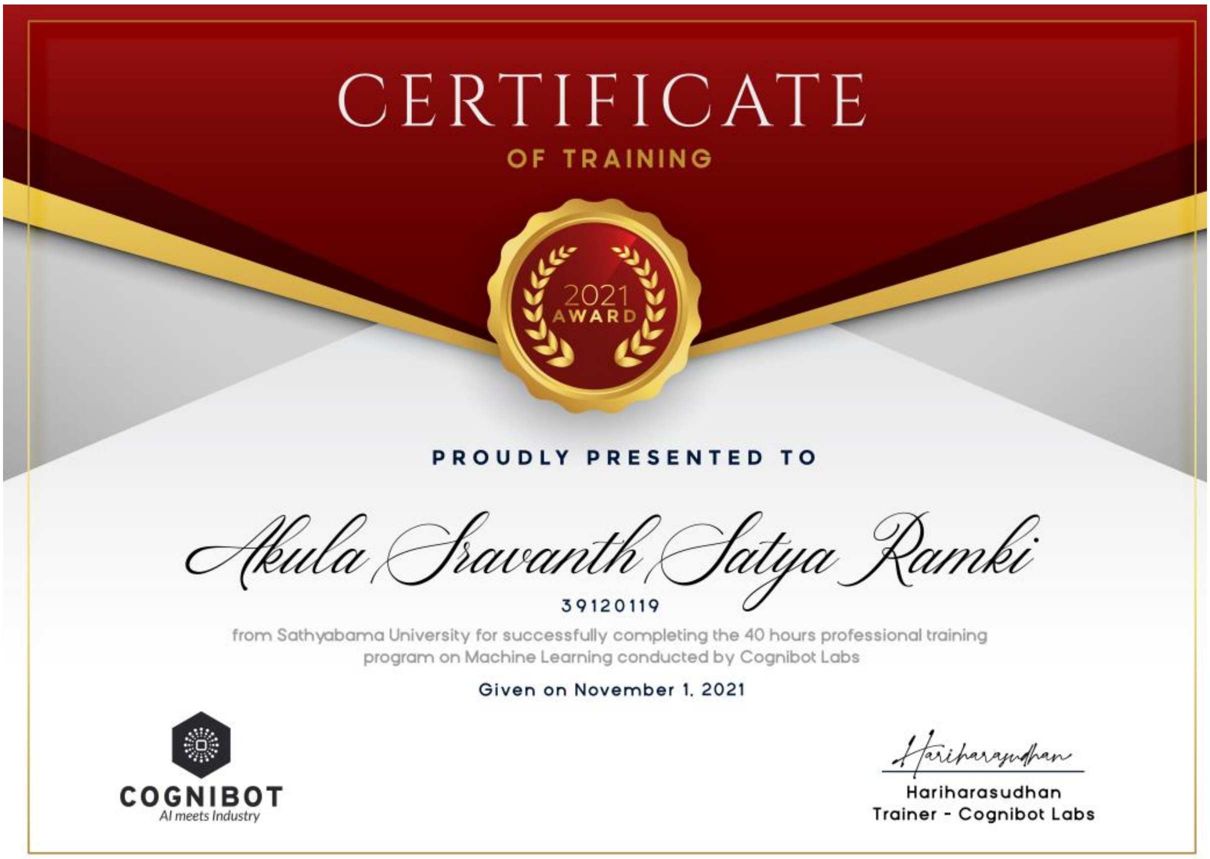
I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D**, **Dean**, School of Computing, **Dr. S. Vigneshwari, M.E., Ph.D. and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of **Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. R.M. Gomathi, M.Tech., Ph.D.,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Nonteaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



CERTIFICATE

OF TRAINING

PROUDLY PRESENTED TO

Akula Sravanth Satya Ramki

39120119

from Sathyabama University for successfully completing the 40 hours professional training program on Machine Learning conducted by Cognibot Labs

Given on November 1, 2021

COGNIBOT
AI meets Industry

Hariharasudhan

Hariharasudhan
Trainer - Cognibot Labs

Abstract

It is a dare to the bank to give the loan to someone because, it is based on the persons the applicant’s demographic and socio-economic profiles.

When a bank receives a loan application, based on the applicant’s profile the bank has to make a decision regarding whether to go ahead with the loan approval or not. If the applicant is a good credit risk, i.e. is likely to repay the loan, then not approving the loan to the person results in a loss of business to the bank.If the applicant is a bad credit risk, i.e. is not likely to repay the loan, then approving the loan to the person results in a financial loss to the bank.It may be assumed that the second risk is a greater risk, as the bank (or any other institution lending the money to a untrustworthy party) had a higher chance of not being paid back the borrowed amount.So its on the part of the bank or other lending authority to evaluate the risks associated with lending money to a customer.In business terms, we try to minimize the risk and maximize of profit for the bank. To minimize loss from the bank’s perspective, the bank needs a decision rule regarding who to give approval of the loan and who not to. An applicant’s demographic and socio-economic profiles are considered by loan managers before a decision is taken regarding his/her loan application.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1:INTRODUCTION

1.1 SOUTH GERMAN CREDIT

This Dataset is only for the bank for the loan. When a bank receives a loan application, based on the applicant’s profile the bank has to make a decision regarding whether to go ahead with the loan approval or not. Two types of risks are associated with the bank’s decision

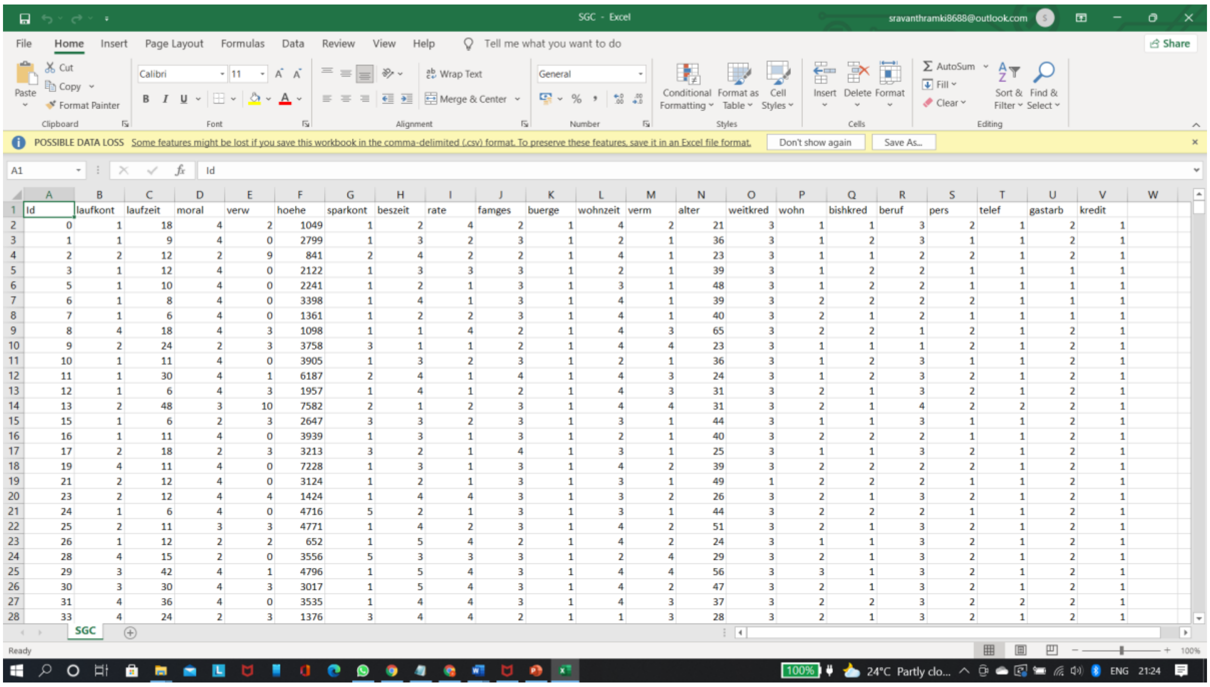
- If the applicant is a good credit risk, i.e. is likely to repay the loan .If the applicant is a bad credit risk, i.e. is not likely to repay the loan
- This study aims at adressing this classification problem by using the the applicant’s demographic and socio-economic profiles of **south german credit data** to examine the risk of lending loan to the customer.

we try to minimize the risk and maximize of profit for the bank. To minimize loss from the bank’s perspective, the bank needs a decision rule regarding who to give approval of the loan

1.2 ESTIMATING DATASET INFORMATION

- The given dataset contains 800 rows and 22 columns. The first 21 columns are features, the last column contains the classification label of ‘0's and ‘1’.

  Dataset is given below (Fig.1.2)



POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

A1 Id

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	Id	laufkont	laufzeit	moral	verw	hoehe	sparkont	beszeit	rate	famges	buerge	wohnzeit	verm	alter	weitkred	wohn	bishkred	beruf	pers	telef	gastarb	kredit
2	0	1	18	4	2	1049	1	2	4	2	1	4	2	21	3	1	1	3	2	1	2	1
3	1	1	9	4	0	2799	1	3	2	3	1	2	1	36	3	1	2	3	1	1	2	1
4	2	2	12	2	9	841	2	4	2	2	1	4	1	23	3	1	1	2	2	1	2	1
5	3	1	12	4	0	2122	1	3	3	3	1	2	1	39	3	1	2	2	1	1	1	1
6	5	1	10	4	0	2241	1	2	1	3	1	3	1	48	3	1	2	2	1	1	1	1
7	6	1	8	4	0	3398	1	4	1	3	1	4	1	39	3	2	2	2	2	1	1	1
8	7	1	6	4	0	1361	1	2	2	3	1	4	1	40	3	2	1	2	1	1	1	1
9	8	4	18	4	3	1098	1	1	4	2	1	4	3	65	3	2	2	1	2	1	2	1
10	9	2	24	2	3	3758	3	1	1	2	1	4	4	23	3	1	1	1	2	1	2	1
11	10	1	11	4	0	3905	1	3	2	3	1	2	1	36	3	1	2	3	1	1	2	1
12	11	1	30	4	1	6187	2	4	1	4	1	4	3	24	3	1	2	3	2	1	2	1
13	12	1	6	4	3	1957	1	4	1	2	1	4	3	31	3	2	1	3	2	1	2	1
14	13	2	48	3	10	7582	2	1	2	3	1	4	4	31	3	2	1	4	2	2	2	1
15	15	1	6	2	3	2647	3	3	2	3	1	3	1	44	3	1	1	3	1	1	2	1
16	16	1	11	4	0	3939	1	3	1	3	1	2	1	40	3	2	2	2	1	1	2	1
17	17	2	18	2	3	3213	3	2	1	4	1	3	1	25	3	1	1	3	2	1	2	1
18	19	4	11	4	0	7228	1	3	1	3	1	4	2	39	3	2	2	2	2	1	2	1
19	21	2	12	4	0	3124	1	2	1	3	1	3	1	49	1	2	2	2	1	1	2	1
20	23	2	12	4	4	1424	1	4	4	3	1	3	2	26	3	2	1	3	2	1	2	1
21	24	1	6	4	0	4716	5	2	1	3	1	3	1	44	3	2	2	2	1	1	2	1
22	25	2	11	3	3	4771	1	4	2	3	1	4	2	51	3	2	1	3	2	1	2	1
23	26	1	12	2	2	652	1	5	4	2	1	4	2	24	3	1	1	3	2	1	2	1
24	28	4	15	2	0	3556	5	3	3	3	1	2	4	29	3	2	1	3	2	1	2	1
25	29	3	42	4	1	4796	1	5	4	3	1	4	4	56	3	3	1	3	2	1	2	1
26	30	3	30	4	3	3017	1	5	4	3	1	4	2	47	3	2	1	3	2	1	2	1
27	31	4	36	4	0	3535	1	4	4	3	1	4	3	37	3	2	2	3	2	2	2	1
28	33	4	24	2	3	1376	3	4	4	2	1	1	3	28	3	2	1	3	2	1	2	1

Fig 1.1 - excel sheet of the given south german credit dataset

1.3 COMMON MACHINE LEARNING ALGORITHMS AND GOALS

Then the The variety of machine learning algorithms are classified into three categories as

follows –

Supervised learning algorithms model the relationship between features (independent variables) and a label (target) from given set of observations. Then the model is used to predict the label of new observations using the features. Depending on the characteristics of the target variable i.e., it can be either be classification(discrete variable) or regression (continuous variable) the task is further engaged.

.

Unsupervised learning finds the structures in unlabeled data.

Reinforcement learning works on action-reward principle. An agent learned to reach the goal by continuously calculating the rewards that it gained from the actions

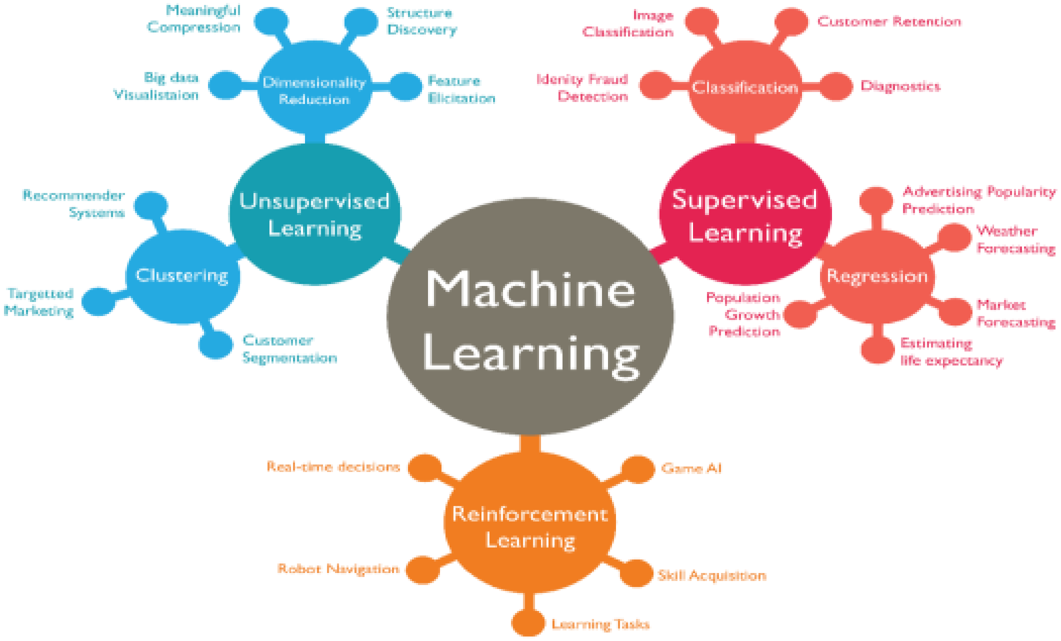


Fig 1.2 - types of machine learning along with the field of use

ALGORITHMS

1. Linear Regression is a supervised learning algorithm and tries to be a bridge between a continuous target variable and one or more independent variables by fitting a linear equation to the data. For choosing this algorithm, there needs be a linear relation between independent and target variable. As scatter plot shows the positive correlation between an independent variable(x-axis) and dependent variable (y-axis).

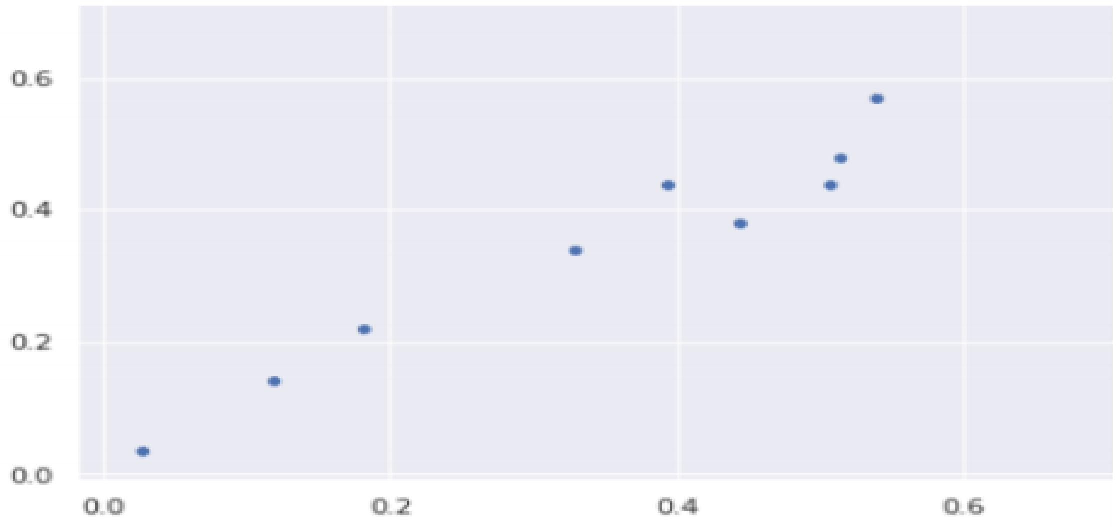


Fig 1.3- Linear regression scatter plot

This would try to put regression line to represent relations. Common technique is ordinary-least squares (OLS). As a result, we could get a regression line as a outcome by minimizing sum square of distance between data points and regression line.

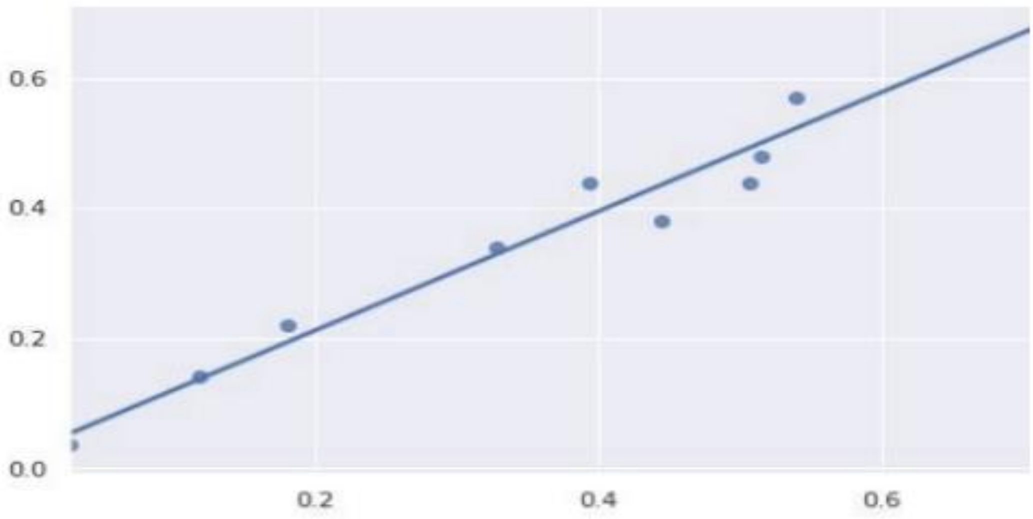


Fig 1.4- Linear regression scatter plot with regression line

2. Naïve Bayes is a supervised learning algorithm used for classification

problems, also called as Naïve Bayes Classifier. It assumes that features are independent of each other and there is no correlation between features. As assumption of features being uncorrelated is the reason for the name “naïve”.

Equaton:

$$\boldsymbol{p(A|B)} = \frac{\mathbf{p(A).\,p(B|A)}}{\mathbf{p(B)}} \qquad \boldsymbol{(Bayes'\ Theorem)}$$

p(A|B): Probability of event A given event B has already occurred

p(B|A): Probability of event B given event A has already occurred

p(A): Probability of event A

p(B): Probability of event B

3. Logistic Regression is a supervised learning algorithm which is mostly used for binary classification problems. Even when regression contradicts with classification, here the spot is for logistic that refers to logistic function which does the classification task. It is simple but effective classification algorithms most commonly used for binary classification problems. Logistic function also known as sigmoid function.

Equation:

$$Sigmoid\ Function\!:\ y = \frac{1}{1 + e^{-x}}$$

Logistic regression takes linear equation as input and uses sigmoid function and logs odds to perform a binary problem. As result s shape graph will be the output

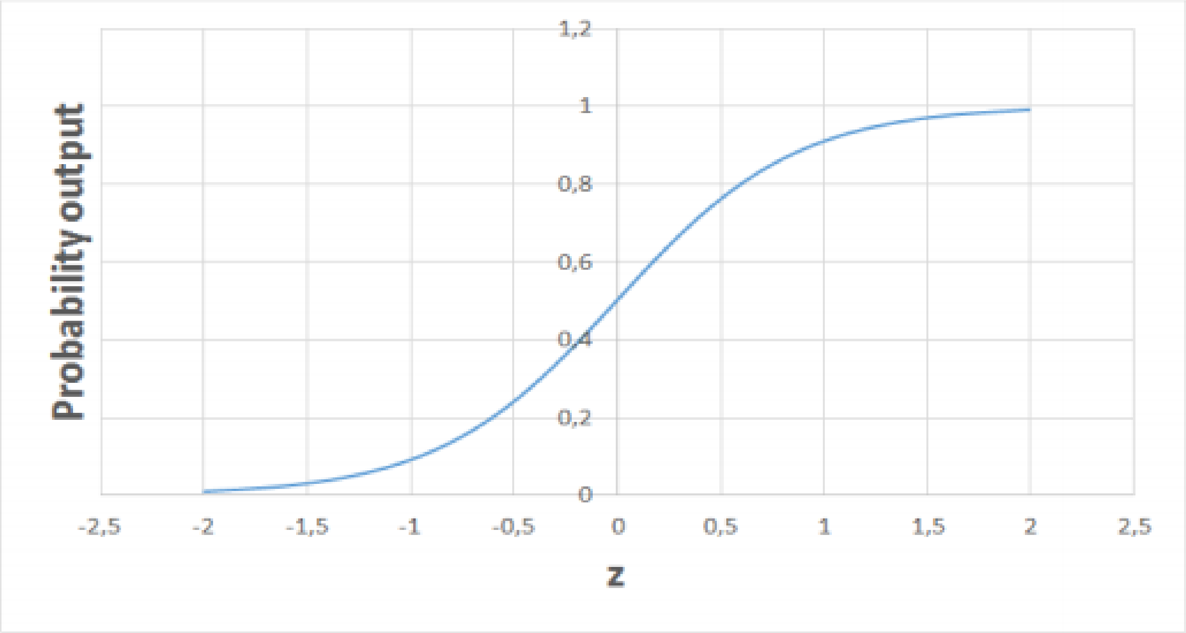


Fig 1.5 - Logistic regression with probability output in s shape

4. Decision Trees build upon continuously to partition the data. The aim of decision tree is to increase the predictiveness as much as possible at each stage so that the model keeps gaining information about the dataset. Randomly splitting will not give us valuable insight into dataset. The purity of node is inversely proportional to distribution of different classes in that node. Overfitting model would be too specific model and not be generalize well. Though it achieves high accuracy with training set but poorly on new. The depth of the tree is controlled by max_depth parameter for decision tree algorithm in scikit-learn. Is also suitable to work on a mixture of feature data types.

Example

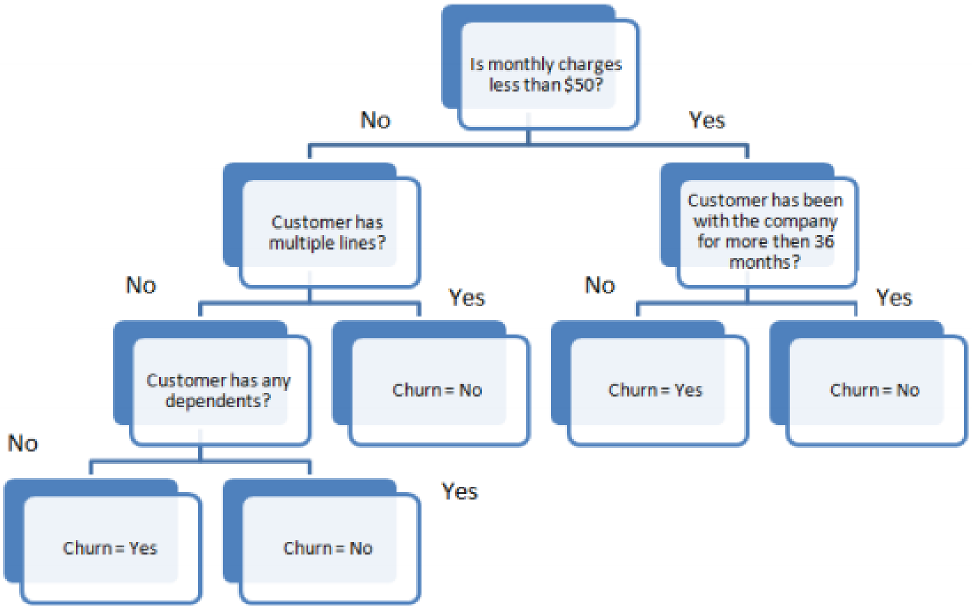

Is monthly charges less than $50?
No
Yes
Customer has multiple lines?
Customer has been with the company for more then 36 months?
No
Yes
No
Yes
Customer has any dependents?
Churn = No
Churn = Yes
Churn = No
No
Yes
Churn = Yes
Churn = No

Fig 1.6- Example for decision tree

5. Random Forest is an ensemble of many decision trees. They are built using a method called bagging where decision trees are used as parallel estimators. When used in classification problem, the result will be based on majority of vote received from each decision tree.

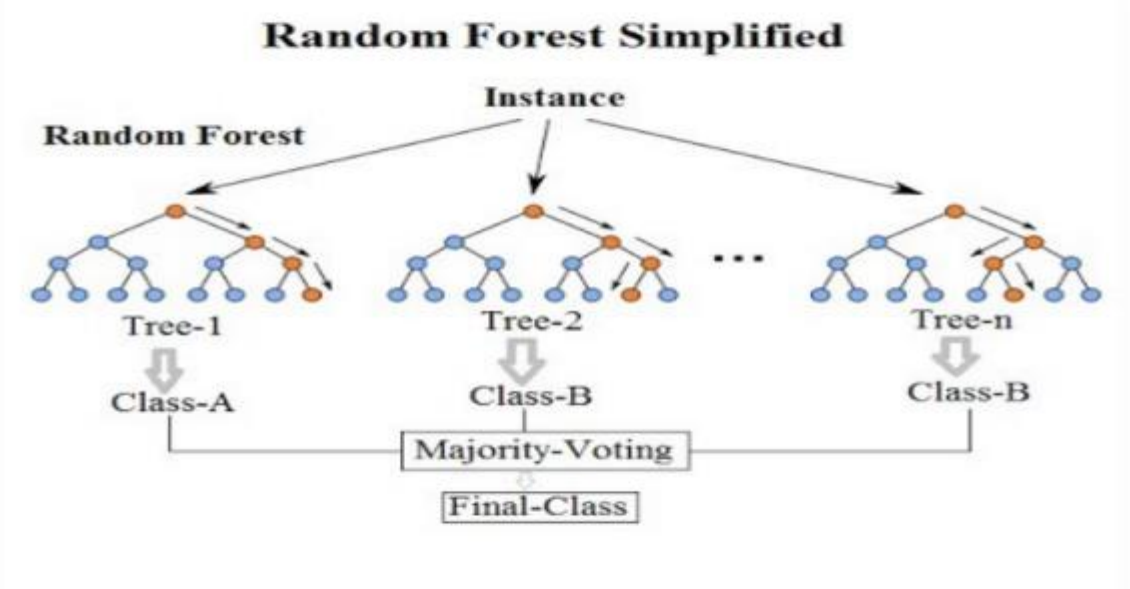


Fig 1.7- Random forest outlier

6. K-means Clustering is a way to group of set of data points in a way that similar data points are together. Thus, they look for dissimilarities or similarities among data points. It is an unsupervised learning so there is no label associated with data points. They try to find the underlying structures of the data. Clustering is not Classification.

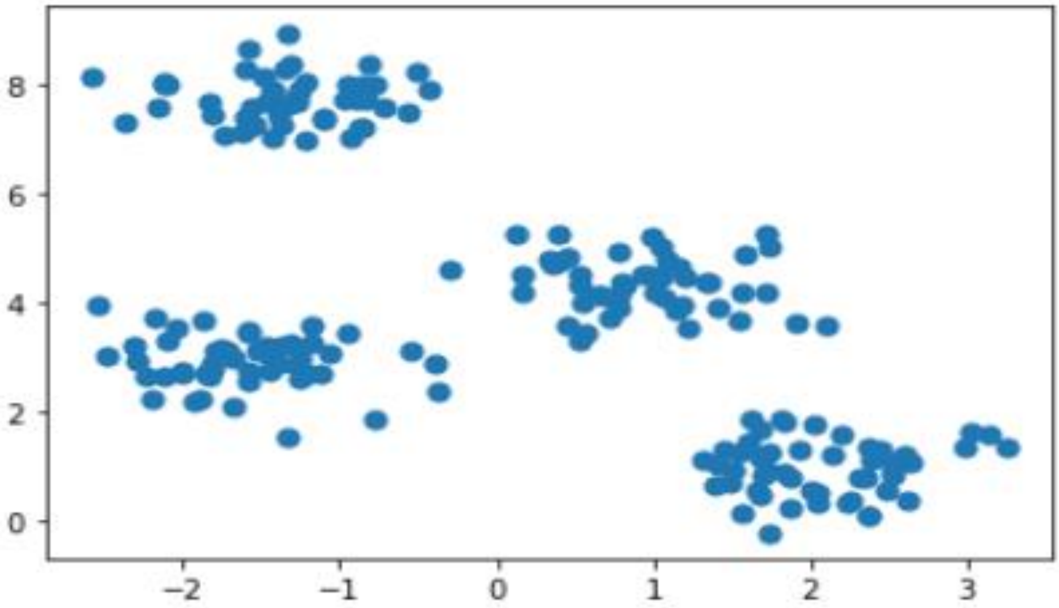


Fig 1.8-scatter plot on K- means clustering

CHAPTER-2: AIM AND SCOPE OF PRESENT INVESTIGATION

2.1 AIM:

To predict the Credit Risk in the given South German credit Data set using features from the given Data set

2.2 SCOPE:

In our daily life , all banks are Thinking the only thing about the loan. Means, for giving loan to the person is it good to give or bad based on their demographic and socio-economic profiles.

In the given data set ,

1. Data set characteristics are Multivariate
2. Attribute charaecteristics are interger,real
3. Associated task – Classification
4. no.of instances – 800
5. no.of attributes – 22 (one attribute is not usefull)

Then, it will be 21 Attributes

2.3 ATTRIBUTES INFORMATION :

1. Id = Id of individual entries, for evaluation
2. laufkont = status
   1 : no checking account
   2 : ... < 0 DM
   3 : 0<= ... < 200 DM
   4 : ... >= 200 DM / salary for at least 1 year
3. laufzeit = duration
4. moral = credit_history
   0 : delay in paying off in the past
   1 : critical account/other credits elsewhere
   2 : no credits taken/all credits paid back duly 3 : existing credits paid back duly till now
   4 : all credits at this bank paid back duly
5. verw = purpose 0 : others
   1 : car (new)
   2 : car (used)
   3 : furniture/equipment 4 : radio/television

5 : domestic appliances 6 : repairs
7 : education
8 : vacation
9 : retraining
10 : business

6. hoehe = amount
7. sparkont = savings
   1 : unknown/no savings account 2 : ... < 100 DM
   3 : 100 <= ... < 500 DM
   4 : 500 <= ... < 1000 DM
   5 : ... >= 1000 DM
8. beszeit = employment_duration
   1 : unemployed
   2 : < 1 yr
   3 : 1 <= ... < 4 yrs 4 : 4 <= ... < 7 yrs 5 : >= 7 yrs
9. rate = installment_rate
   1 : >= 35
   2 : 25 <= ... < 35 3 : 20 <= ... < 25 4 : < 20
10. famges = personal_status_sex
    1 : male : divorced/separated
    2 : female : non-single or male : single 3 : male : married/widowed
    4 : female : single
11. buerge = other_debtors
    1 : none
    2 : co-applicant 3 : guarantor
12. wohnzeit = present_residence
    1 : < 1 yr
    2 : 1 <= ... < 4 yrs 3 : 4 <= ... < 7 yrs 4 : >= 7 yrs
13. verm = property
    1 : unknown / no property
    2 : car or other
    3 : building soc. savings agr./life insurance 4 : real estate
14. alter = age
15. weitkred = other_installment_plans
    1 : bank
    2 : stores 3 : none
16. wohn = housing
    1 : for free 2 : rent
    3 : own
17. bishkred = number_credits 1 : 1
    2 : 2-3 3 : 4-5 4 : >= 6
18. beruf = job
    1 : unemployed/unskilled - non-resident
    2 : unskilled - resident
    3 : skilled employee/official
    4 : manager/self-empl./highly qualif. employee
19. pers = people_liable
    1 : 3 or more 2 : 0 to 2
20. telef = telephone
    1 : no
    2 : yes (under customer name)
21. gastarb = foreign_worker 1 : yes 2 : no
22. *kredit (target column)* = credit_risk 0 : bad 1 : good

2.4 MISSING ATTRIBUTES : (denoted by "null")

```
Data columns (total 21 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   status                  800 non-null    int64
 1   duration                800 non-null    int64
 2   credit_history          800 non-null    int64
 3   purpose                 800 non-null    int64
 4   amount                  800 non-null    int64
 5   savings                 800 non-null    int64
 6   employment_duration     800 non-null    int64
 7   installment_rate        800 non-null    int64
 8   personal_status_sex     800 non-null    int64
 9   other_debtors           800 non-null    int64
 10  present_residence       800 non-null    int64
 11  property                800 non-null    int64
 12  age                     800 non-null    int64
 13  other_installment_plans 800 non-null    int64
 14  housing                 800 non-null    int64
 15  number_credits          800 non-null    int64
 16  job                     800 non-null    int64
 17  people_liable           800 non-null    int64
 18  telephone               800 non-null    int64
 19  foreign_worker          800 non-null    int64
 20  credit_risk             800 non-null    int64
dtypes: int64(21)
memory usage: 131.4 KB
```

Threfore there is no null values in the data set

2.5 DISTRIBUTION OF ATTRIBUTE NUMBER 21 : credit Risk

```
Good 1    600
Bad  0    200
Name: credit_risk, dtype: int64
```

2.6 DATA PREPARATION

In this project as a backbone tool python is used to carry out machine learning concepts. With the help of a software called Anaconda Navigator, a jupyter notebook is launched where it is already installed along with the navigator. jupyter notebook is an open-source web application that allows to create and share documents and has live code and also visualization.

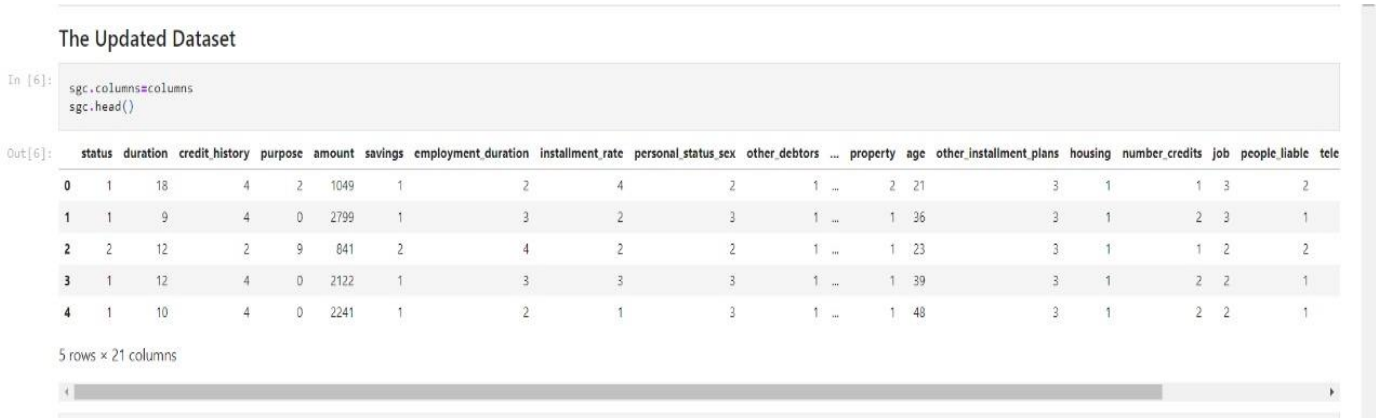
After importing the required libraries the dataset will be read in the note book with help of data frame (two-dimensional labeled data structure with columns of potentially different types) and read_csv(desired file type).

Reading Dataset :

sgc=pd.read_csv(r"F:\machine learning by cognibot\my project tech phanthons\SGC.csv”)

The Dataset which is provided by them is in the language in German and we again modified it into the English(u.k)

The dataset provided cannot always be a fully valued set, in that case we need to prepare the data in such a way the machine understands what is the value that has been entered.



The Updated Dataset

In [6]:
```
sgc.columns=columns
sgc.head()
```

Out[6]:

	status	duration	credit_history	purpose	amount	savings	employment_duration	installment_rate	personal_status_sex	other_debtors	...	property	age	other_installment_plans	housing	number_credits	job	people_liable	tele
0	1	18	4	2	1049	1	2	4	2	1	...	2	21	3	1	1	3	2	
1	1	9	4	0	2799	1	3	2	3	1	...	1	36	3	1	2	3	1	
2	2	12	2	9	841	2	4	2	2	1	...	1	23	3	1	1	2	2	
3	1	12	4	0	2122	1	3	3	3	1	...	1	39	3	1	2	2	1	
4	1	10	4	0	2241	1	2	1	3	1	...	1	48	3	1	2	2	1	

5 rows × 21 columns

Fig 2.1 – Updated Dataset and Modified Dataset

Check the values for null using isnull() function.

```
status                    0
duration                  0
credit_history            0
purpose                   0
amount                    0
savings                   0
employment_duration       0
installment_rate          0
personal_status_sex       0
other_debtors             0
present_residence         0
property                  0
age                       0
other_installment_plans   0
housing                   0
number_credits            0
job                       0
people_liable             0
telephone                 0
foreign_worker            0
credit_risk               0
dtype: int64
```

The dataset is now ready for creating a machine learning model.

CHAPTER- 3 : EXPERIMENTAL OR MATERIAL AND METHODS, ALGORITHMS USED

The given data set is in the form of Classification algorithm.so,we used classification types to predict the accuracy.

3.1 Types of classification algorithms used :

Decision Tree

3.2 *Decision tree Algorithm* :

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
- It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

3.3 IMPORTED LIBRARIES

Libraries are collections of prewritten code that users can use to optimize tasks. In project as python is used for implementation tool, it has the most libraries as compared to other programming languages. More than of 60% machine learning developers use and goes for python as it is easy to learn. As python has comparatively large collection of libraries let’s l0ook at the libraries that came in handy for mammographic dataset.

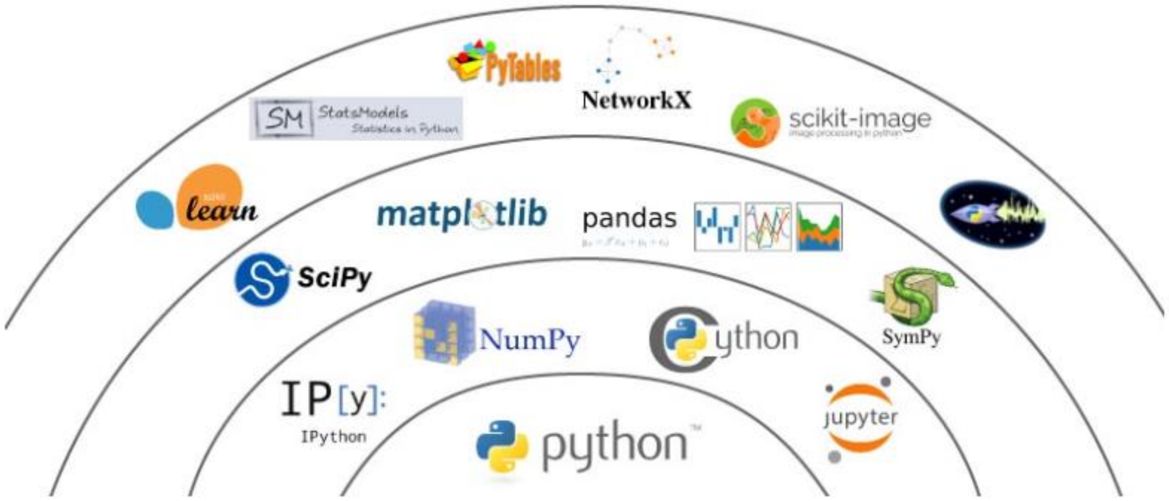


Fig 3.1- various python libraries for Machine Learning

LIBRARIES USED:

1. Pandas is a widely-used data analysis and manipulation library for python. It provides a lot of functions and methods that expedite the data analysis and preprocessing steps. IT also provides fast, flexible and expressive data structures working with relational or labeled or both easy and intuitive. Considered as fundamental high-level building block in performing practical, real-world data analysis in python. Has powerful tools like DataFrame and Series for analyzing.

2. Numpy stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of countless of routines for processing those arrays. Using this mathematical and logical operations on arrays can be performed. The difference in using Numpy from pandas is, it works on numerical data whereas pandas on tabular data.

3. sklearn stands for Scikit-learn, a machine learning library. It is imported for various classification, regression and clustering algorithms including k-means, random forest, support vector machines, gradient boosting and DBSCAN. It is designed using libraries Numpy and Scipy.

From the sklearn library and from the tree inside the library DecisionTreeClassifier. It is a class capable of performing multi-class classifier on a dataset. When compared with other classifiers, DecisionTreeClassifier takes input as two arrays: an array X, aparse or dense, of shape(n_samples, n_features) holding training samples and an array Y of integer values, shape (n_samples), holding class labels for training sample.

From sklearn another one called model_selection for training and testing the model imports train_test_split. It is a method setting a blueprint to analyze data and the using it to measure new data. Selecting a proper model allows to generate accurate results while making prediction. For proceeding, we need to train the model by using a specific dataset and test the model against another dataset.

By default, sklearn train_test_split will make random partitions for two subsets. We can also specify a random state for the operation. First, we need to split the dataset and then allocate the size for train and test. For this mammographic dataset we need train size as 80% (0.80) and test size as 20% (0.20) with the random state of 100.

4. Seaborn is a library built on top of matplotlib. It used for data visualization and exploratory data analysis. They work easily with dataframes and pandas library. The graphs created can also be customized easily. It provides default styles and color palettes to make statistical plots more attractive. Also closely integrated to the data structures from pandas

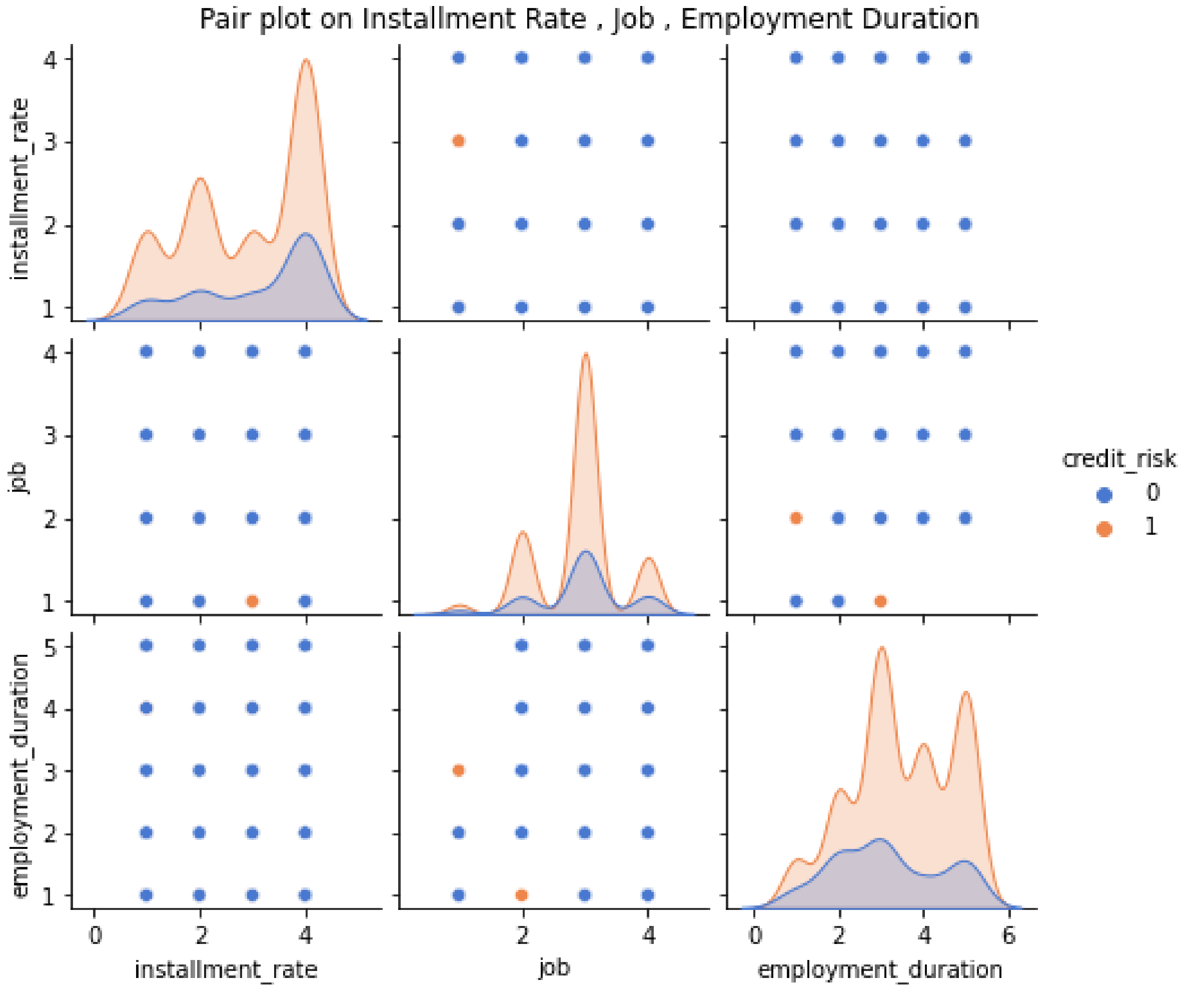


Fig 3.2- Example implementation of Seaborn library

5. Matplotlib.pyplot is a state-based interface to matplotlib. It provides a MATLAB-like way of plotting. It make changes to figures.

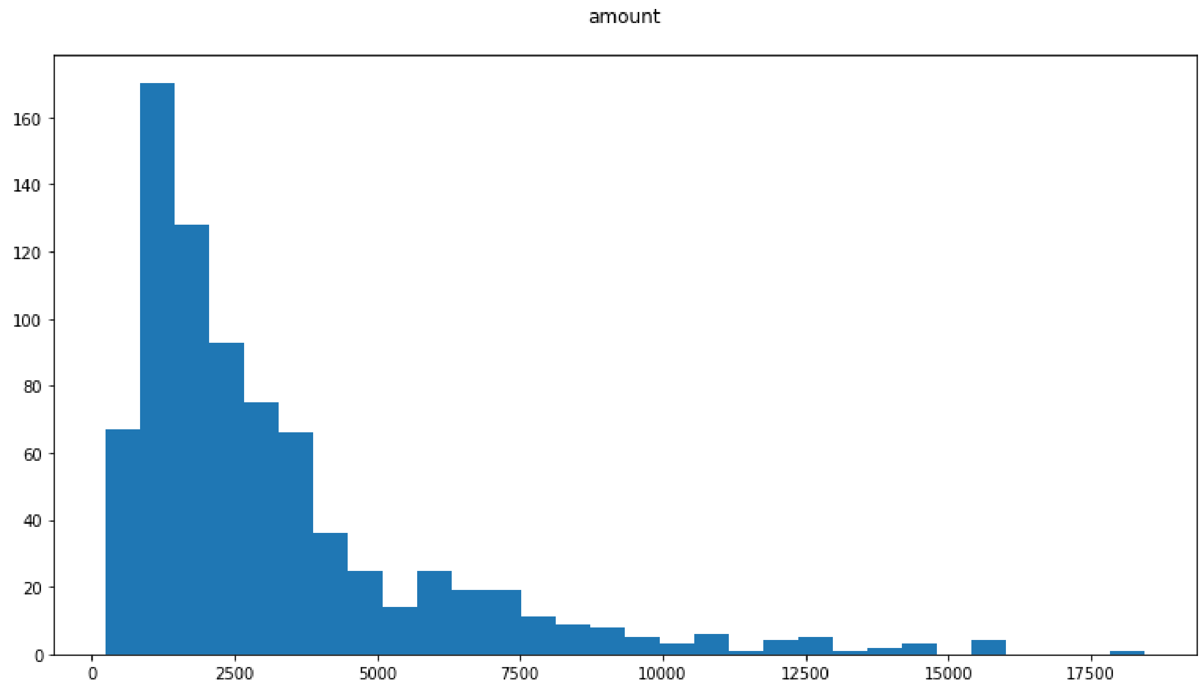
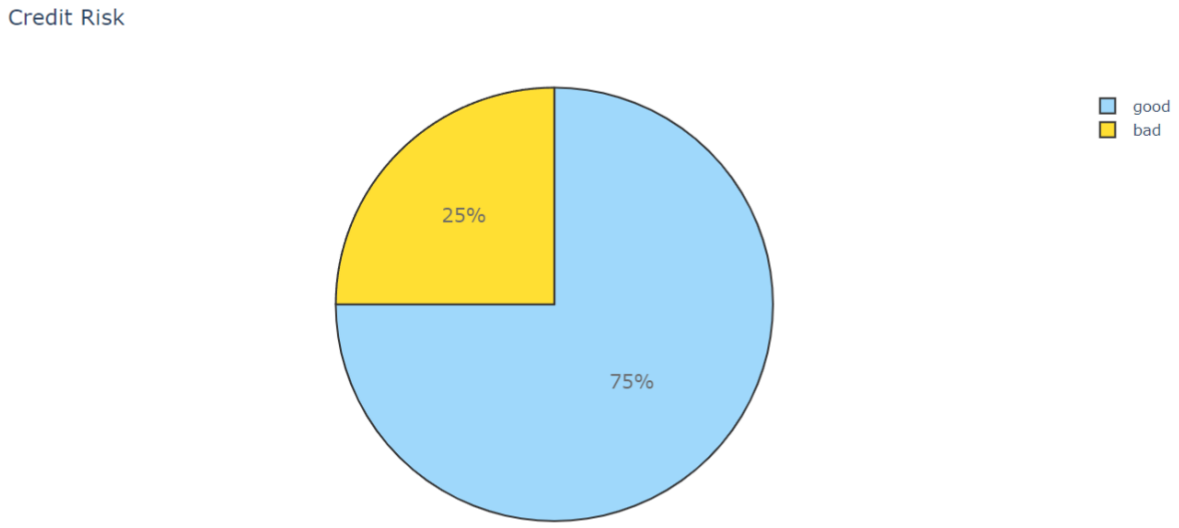


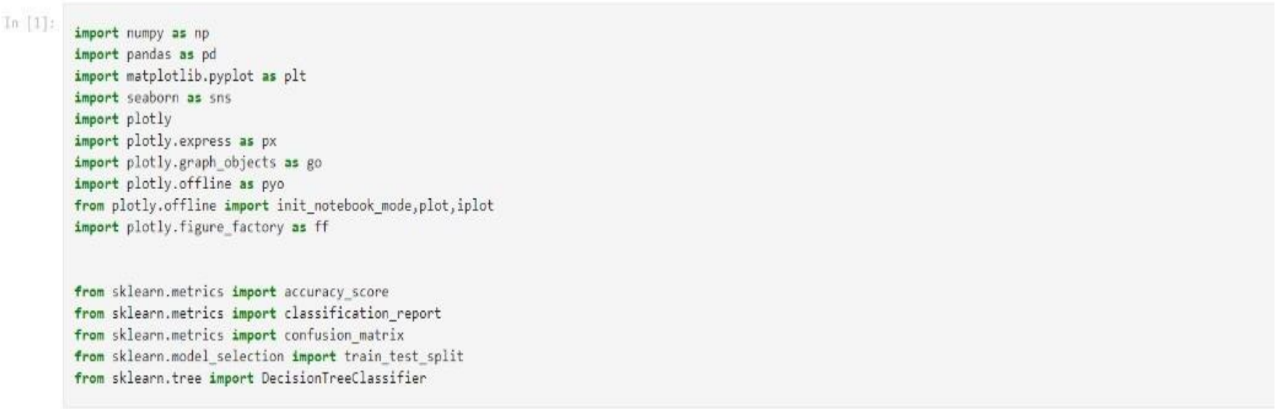
Fig 3.3 Hist plot

6. plotly:

The Plotly library is an interactive open-source library. Plotly makes data visualization and understanding simple and easy.

Fig 3.4 -plotly plot



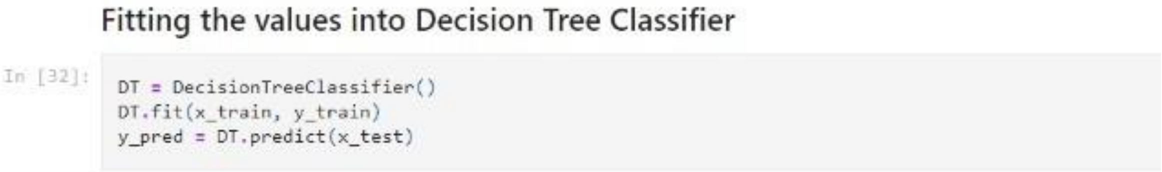


```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly
import plotly.express as px
import plotly.graph_objects as go
import plotly.offline as pyo
from plotly.offline import init_notebook_mode,plot,iplot
import plotly.figure_factory as ff


from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

Fig 3.5- Imported libraries in jupyter notebook

3.4 Implementation of Decision Tree :

1. Import the packages and classes you need.
2. Provide data to work with and eventually do appropriate transformations.
3. Create a regressor model and fit it with existing data.
4. Check the results of model fitting to know whether the model is satisfactory.
5. Apply the model for predictions.



Fitting the values into Decision Tree Classifier

```
DT = DecisionTreeClassifier()
DT.fit(x_train, y_train)
y_pred = DT.predict(x_test)
```

Fig 3.4.3 Testing and Training data

CHAPTER-4

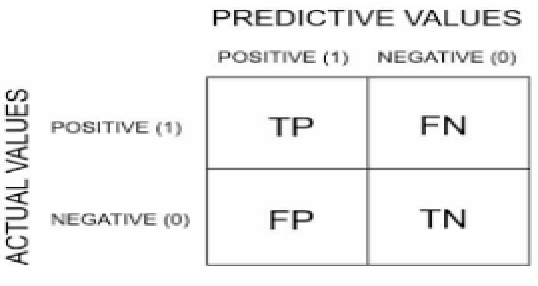
RESULTS AND PERFORMANCE ANALYSIS

4.1 TRAINING AND ACCURACY (MODEL ANALYSIS)

Confusion matrix is to evaluate the accuracy of a classification. This visual metric plots the number of predictions made for each class for each possible class in a table, with each row corresponding to the actual labels and each column corresponding to a prediction. It is beneficial for detecting which actual classes are being detected the most, and what predicted classes are being misclassified as (Bhardwaj and Tiwari, 2015; Liu et al., 2009). To further highlight the misclassifications and compare predictions with other classifiers, the confusion matrices are normalized to show a percentage rather than a count.

The resulted confusion matrix has 45 incorrect prediction i.e., (25 + 20 = 45).

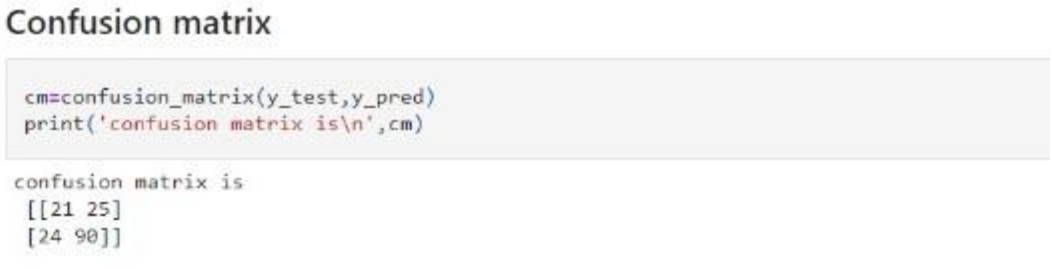
From the matrix of y_test and y_pred with the help of metrics library the incorrect prediction is the outcome



		PREDICTIVE VALUES	
		POSITIVE (1)	NEGATIVE (0)
ACTUAL VALUES	POSITIVE (1)	TP	FN
	NEGATIVE (0)	FP	TN

Fig 4.1- outlier of confusion matrix

For the given dataset:



Confusion matrix

```
cm=confusion_matrix(y_test,y_pred)
print('confusion matrix is\n',cm)
```

```
confusion matrix is
 [[21 25]
 [24 90]]
```

Accuracy is an evaluation metric would be misleading as it would not be representative of how well the classifier fitted the data. Additionally, in breast cancer detection, detecting FPs and FNs is primordial to avoid interpreting malignant as benign and vice versa, an interpretation which could harm the patient and eventually lead to their death.

Accuracy = T P + T N/ P + N

For instance, if a dumb classifier that always classifies an image as “normal” is created, it would achieve 64.28% accuracy on the mini-MIAS dataset despite never picking up abnormal cases. Therefore, a mixture of additional metrics should be used to assess how well the model learns the mammograms data and generalize to unseen cases

Precision

corresponds to the number of correct positive predictions showing the model’s ability to avoid labelling negative instances as positive.

Precision = T P /T P + F P

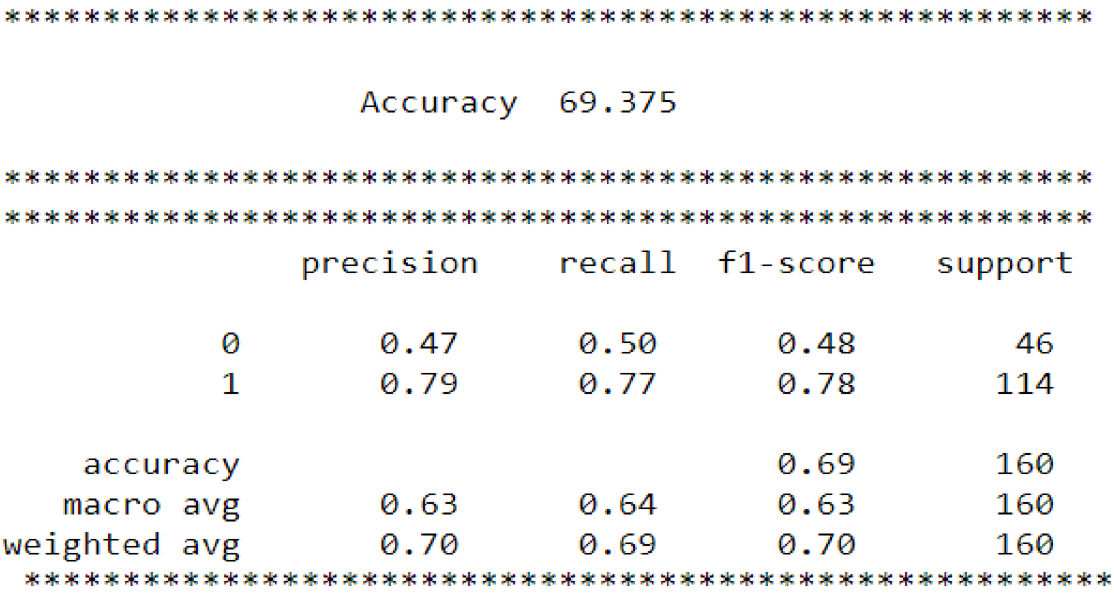
Recall

is the number of positive instances that are correctly predicted showing how well the model can find all positive instances.

Recall = T P/ T P + F

F1 Score

The F1 Score is the 2*((precision*recall)/(precision+recall)). It is also called the F Score or the F Measure. Put another way, the F1 score conveys the balance between the precision and the recall.



```
********************************************************

                  Accuracy  69.375

********************************************************
********************************************************
               precision    recall  f1-score    support

           0       0.47      0.50      0.48        46
           1       0.79      0.77      0.78       114

    accuracy                           0.69       160
   macro avg       0.63      0.64      0.63       160
weighted avg       0.70      0.69      0.70       160
 ********************************************************
```

FIGURE 4.2 ACCURACY,PRECISION,F1-SCORE,RECALL

CHAPTER-5

5.1 SUMMARY AND CONCLUSIONS

- We have modelled the South German Credit Data set using Decision Tree with the *accuracy* *69.375*
- In this Project the credit risk is mainly based on the status of the account,Amount,credit history,purpose,age,gender, installment rate and the remaining rows are showing less impact on the project.
- In the future we can add more categories, ploting more graphs and make it more user friendly and improves its quality. I and my team members are interested in studying these methods and implement in another Machine learning Algorithms .

REFERNCES

[1] UCI Machine Learning Repository: Ionosphere Data Set

https://drive.google.com/file/d/1LmZKbq3TPG5KkTGarsoHvhXpAvNb7u4J/view?usp=sharing

[2] For Learning Python Language
https://www.learnpython.org/

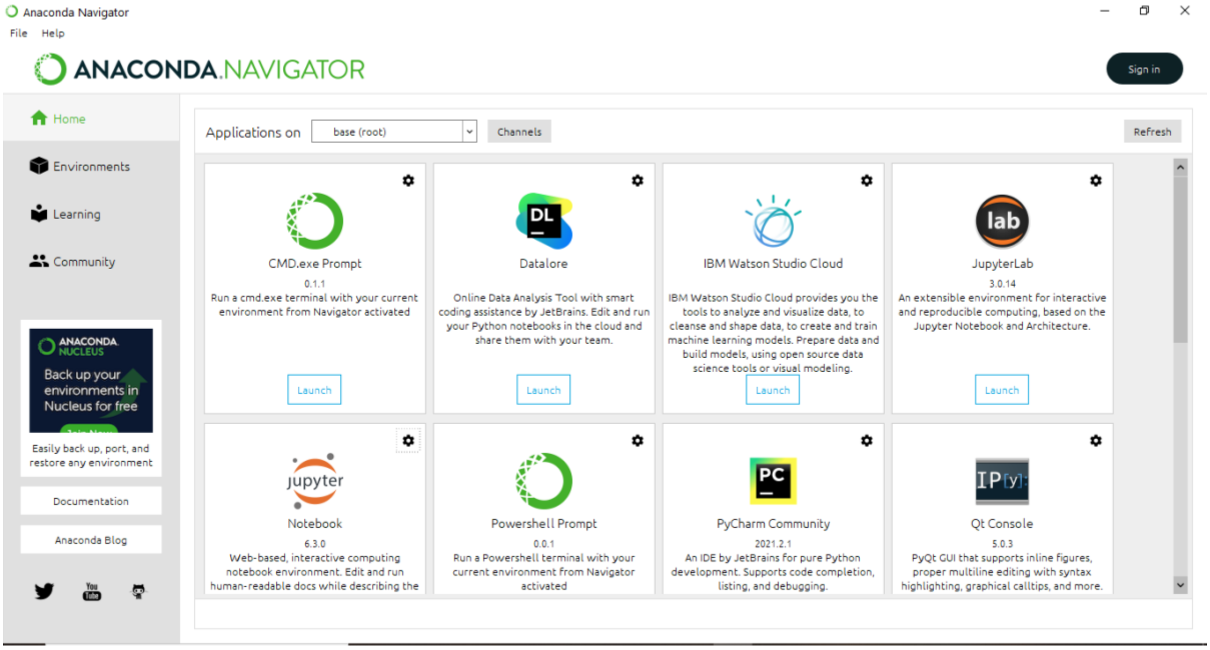
[3] Scipy Lectures Notes
http://scipy-lectures.org/_downloads/ScipyLectures-simple.pdf

[4] Git Hub
https://github.com/dabeaz-course/practical-python/blob/master/Notes/Contents.md

[5] Numpy
https://numpy.org/doc/stable/user/quickstart.html

WORKING ENVIRONMENT

ANACONDA NAVIGATOR is desktop GUI used to launch applications and also manage packages in one place. Outlook



Run a cmd.exe terminal with your current environment from Navigator activated

Online Data Analysis Tool with smart coding assistance by JetBrains. Edit and run your Python notebooks in the cloud and share them with your team.

IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the

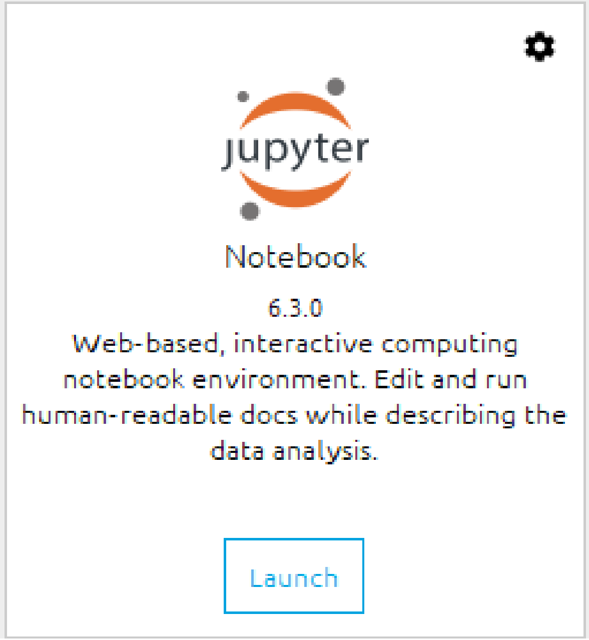
Run a Powershell terminal with your current environment from Navigator activated

An IDE by JetBrains for pure Python development. Supports code completion, listing, and debugging.

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

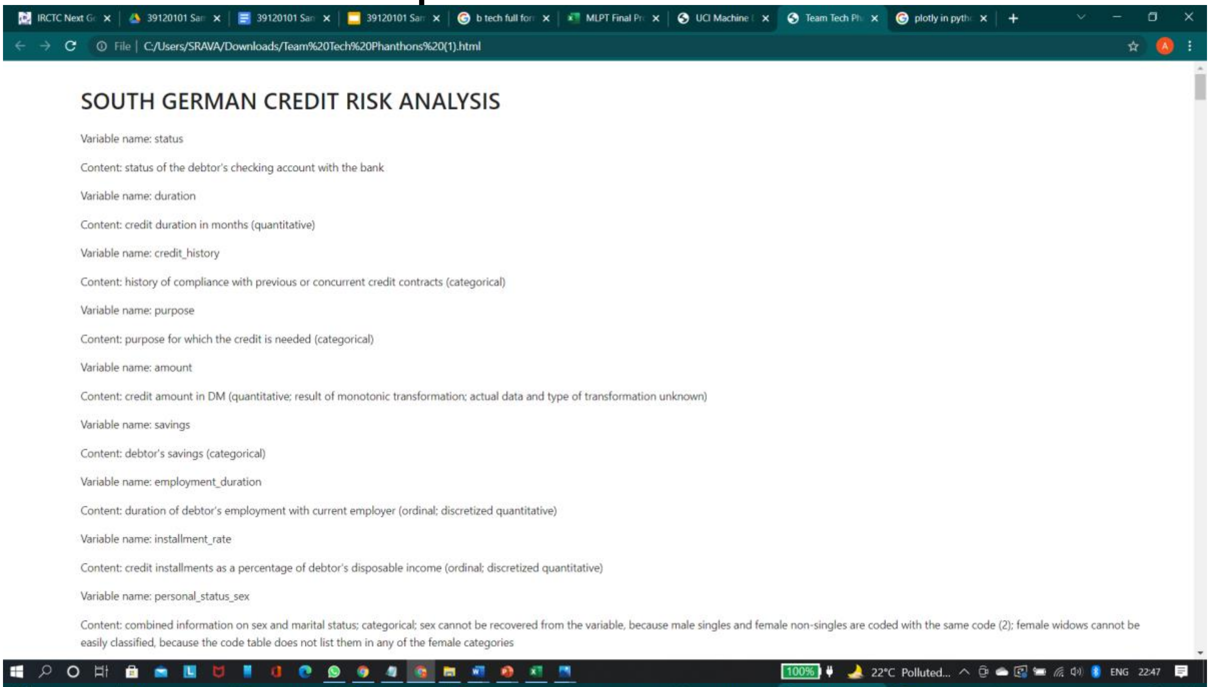
CODING ENVIRONMENT

Jupyter notebook from the anaconda navigator is launched along with all the preinstalled packages for python.



Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

A.Screenshots and Outputs



SOUTH GERMAN CREDIT RISK ANALYSIS

Variable name: status

Content: status of the debtor's checking account with the bank

Variable name: duration

Content: credit duration in months (quantitative)

Variable name: credit_history

Content: history of compliance with previous or concurrent credit contracts (categorical)

Variable name: purpose

Content: purpose for which the credit is needed (categorical)

Variable name: amount

Content: credit amount in DM (quantitative; result of monotonic transformation; actual data and type of transformation unknown)

Variable name: savings

Content: debtor's savings (categorical)

Variable name: employment_duration

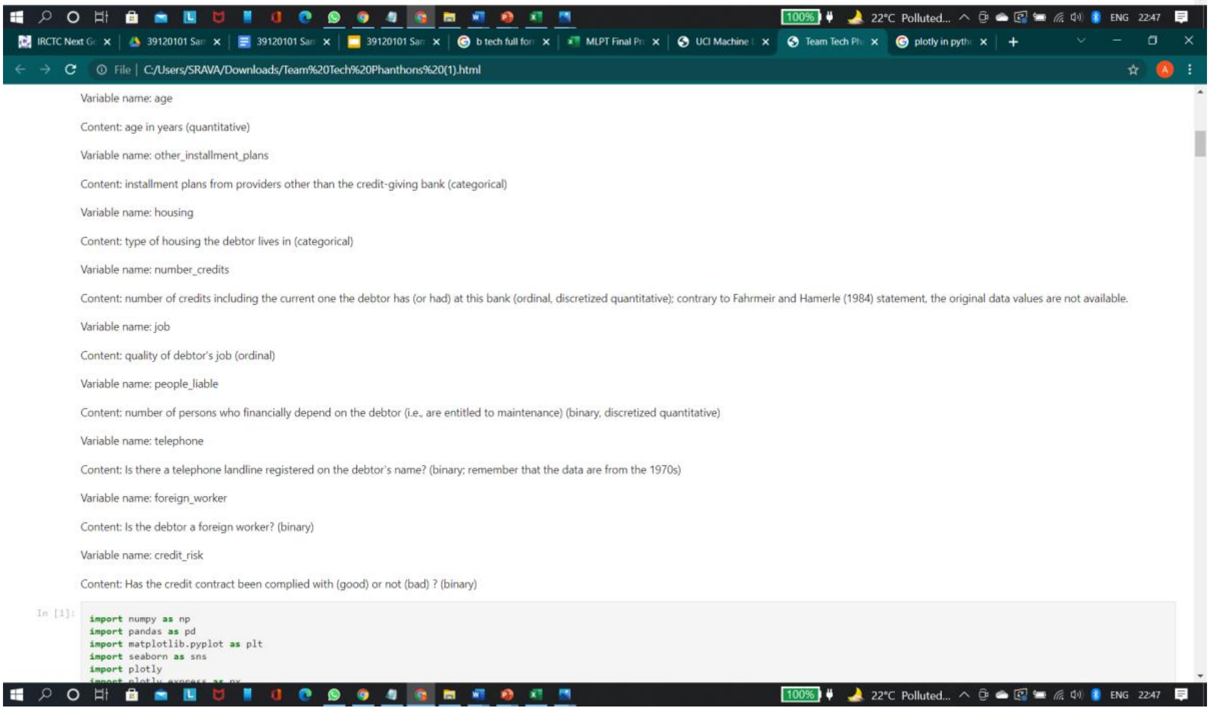
Content: duration of debtor's employment with current employer (ordinal; discretized quantitative)

Variable name: installment_rate

Content: credit installments as a percentage of debtor's disposable income (ordinal; discretized quantitative)

Variable name: personal_status_sex

Content: combined information on sex and marital status; categorical; sex cannot be recovered from the variable, because male singles and female non-singles are coded with the same code (2); female widows cannot be easily classified, because the code table does not list them in any of the female categories



Variable name: age

Content: age in years (quantitative)

Variable name: other_installment_plans

Content: installment plans from providers other than the credit-giving bank (categorical)

Variable name: housing

Content: type of housing the debtor lives in (categorical)

Variable name: number_credits

Content: number of credits including the current one the debtor has (or had) at this bank (ordinal, discretized quantitative); contrary to Fahrmeir and Hamerle (1984) statement, the original data values are not available.

Variable name: job

Content: quality of debtor's job (ordinal)

Variable name: people_liable

Content: number of persons who financially depend on the debtor (i.e., are entitled to maintenance) (binary, discretized quantitative)

Variable name: telephone

Content: Is there a telephone landline registered on the debtor's name? (binary; remember that the data are from the 1970s)

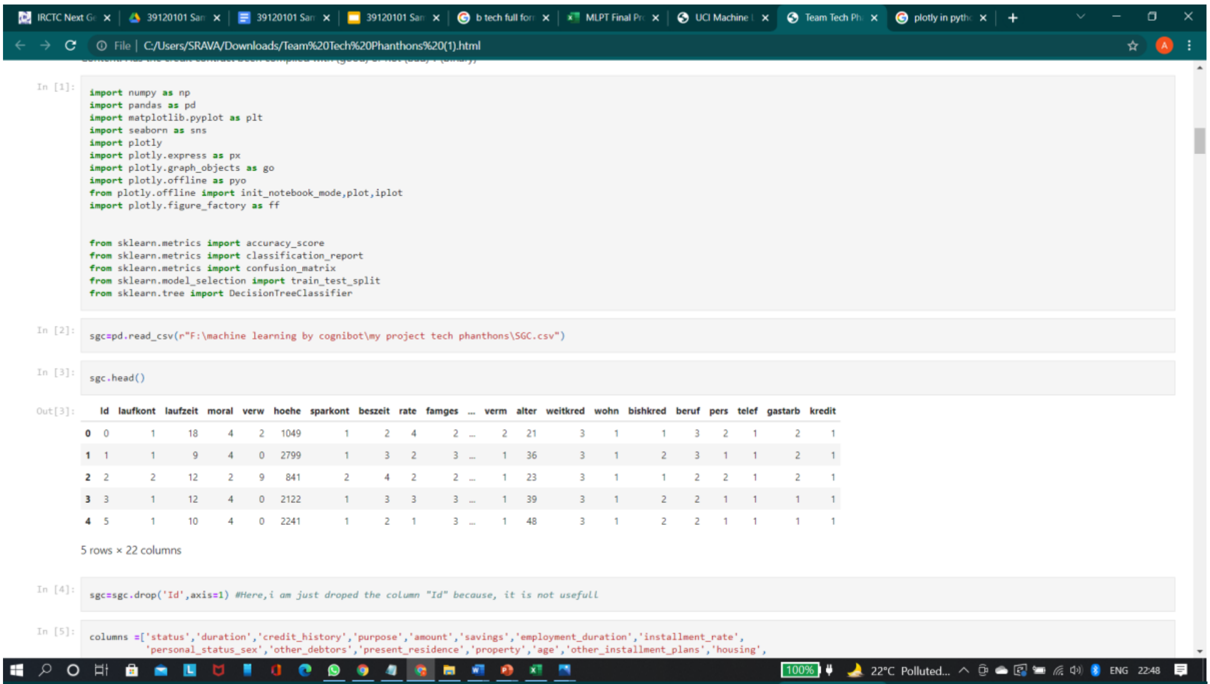
Variable name: foreign_worker

Content: Is the debtor a foreign worker? (binary)

Variable name: credit_risk

Content: Has the credit contract been complied with (good) or not (bad) ? (binary)

```
In [1]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly
import plotly.express as px
```



In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly
import plotly.express as px
import plotly.graph_objects as go
import plotly.offline as pyo
from plotly.offline import init_notebook_mode,plot,iplot
import plotly.figure_factory as ff


from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [2]:

```
sgc=pd.read_csv(r"F:\machine learning by cognibot\my project tech phanthons\SGC.csv")
```

In [3]:

```
sgc.head()
```

Out[3]:

	Id	laufkont	laufzeit	moral	verw	hoehe	sparkont	beszeit	rate	famges	...	verm	alter	weitkred	wohn	bishkred	beruf	pers	telef	gastarb	kredit
0	0	1	18	4	2	1049	1	2	4	2	...	2	21	3	1	1	3	2	1	2	1
1	1	1	9	4	0	2799	1	3	2	3	...	1	36	3	1	2	3	1	1	2	1
2	2	2	12	2	9	841	2	4	2	2	...	1	23	3	1	1	2	2	1	2	1
3	3	1	12	4	0	2122	1	3	3	3	...	1	39	3	1	2	2	1	1	1	1
4	5	1	10	4	0	2241	1	2	1	3	...	1	48	3	1	2	2	1	1	1	1

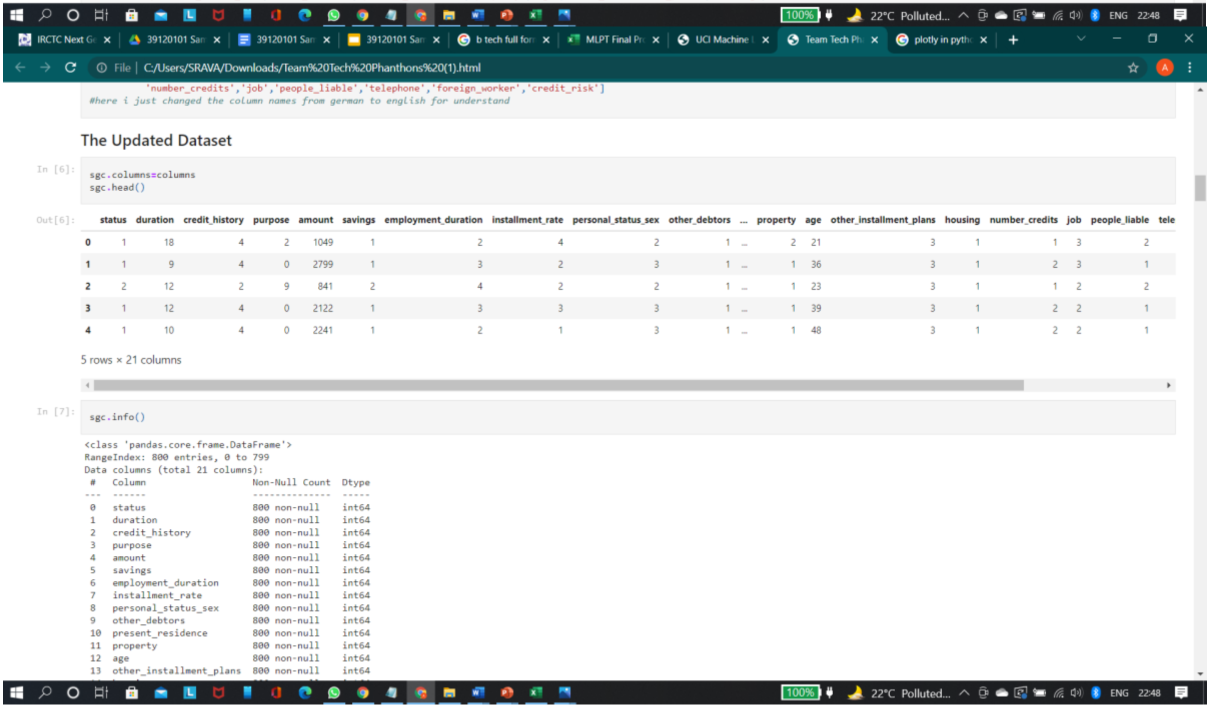
5 rows × 22 columns

In [4]:

```
sgc=sgc.drop('Id',axis=1) #Here,i am just droped the column "Id" because, it is not usefull
```

In [5]:

```
columns =['status','duration','credit_history','purpose','amount','savings','employment_duration','installment_rate',
          'personal_status_sex','other_debtors','present_residence','property','age','other_installment_plans','housing',
```



```
          'number_credits','job','people_liable','telephone','foreign_worker','credit_risk']
#here i just changed the column names from german to english for understand
```

The Updated Dataset

In [6]:

```
sgc.columns=columns
sgc.head()
```

Out[6]:

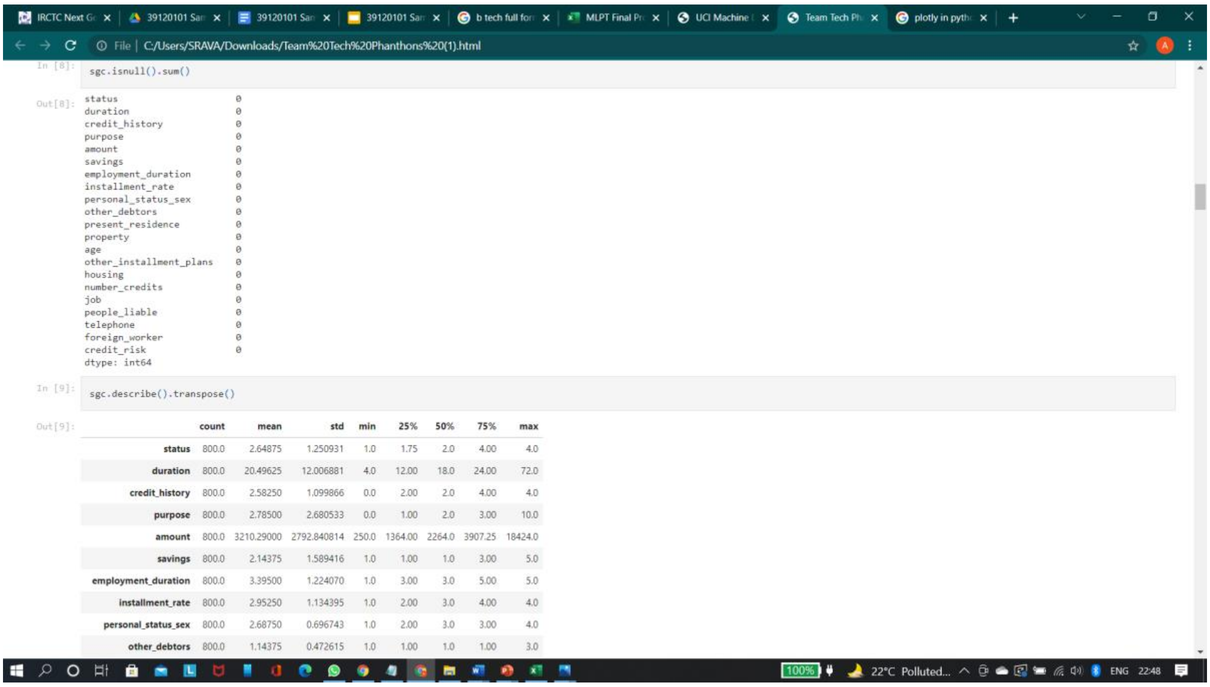
	status	duration	credit_history	purpose	amount	savings	employment_duration	installment_rate	personal_status_sex	other_debtors	...	property	age	other_installment_plans	housing	number_credits	job	people_liable	tele
0	1	18	4	2	1049	1	2	4	2	1	...	2	21	3	1	1	3	2	
1	1	9	4	0	2799	1	3	2	3	1	...	1	36	3	1	2	3	1	
2	2	12	2	9	841	2	4	2	2	1	...	1	23	3	1	1	2	2	
3	1	12	4	0	2122	1	3	3	3	1	...	1	39	3	1	2	2	1	
4	1	10	4	0	2241	1	2	1	3	1	...	1	48	3	1	2	2	1	

5 rows × 21 columns

In [7]:

```
sgc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 800 entries, 0 to 799
Data columns (total 21 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   status                   800 non-null    int64
 1   duration                 800 non-null    int64
 2   credit_history           800 non-null    int64
 3   purpose                  800 non-null    int64
 4   amount                   800 non-null    int64
 5   savings                  800 non-null    int64
 6   employment_duration      800 non-null    int64
 7   installment_rate         800 non-null    int64
 8   personal_status_sex      800 non-null    int64
 9   other_debtors            800 non-null    int64
 10  present_residence        800 non-null    int64
 11  property                 800 non-null    int64
 12  age                      800 non-null    int64
 13  other_installment_plans  800 non-null    int64
```



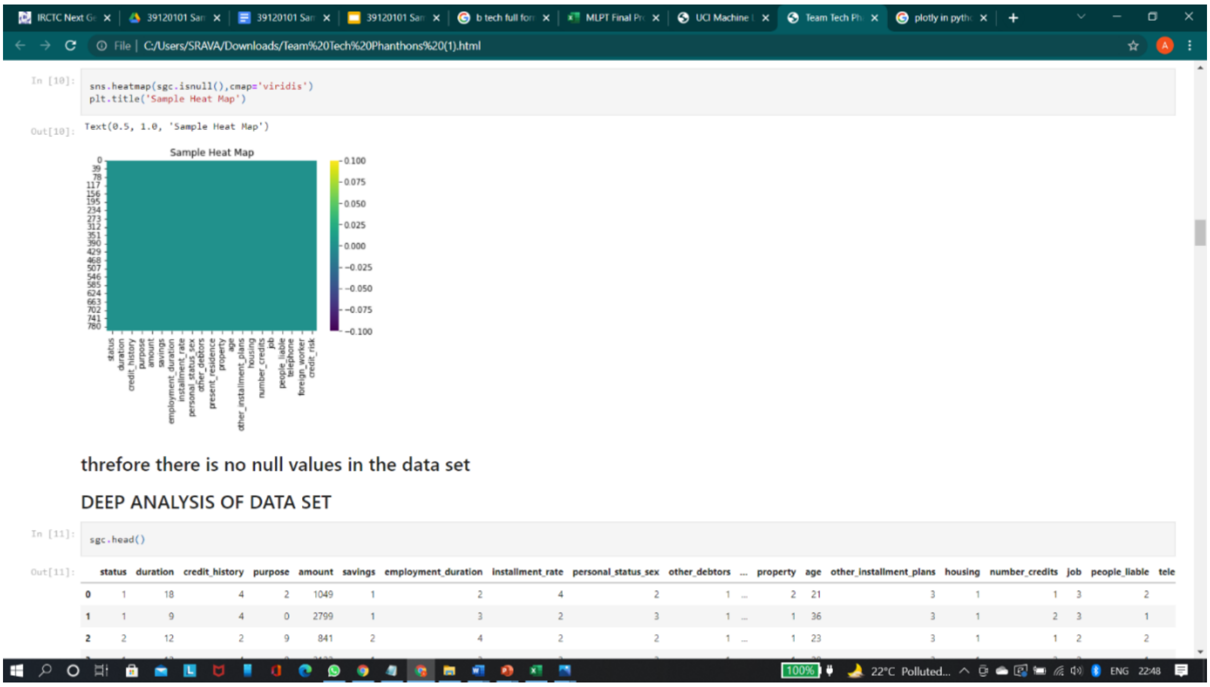
In [8]:
```
sgc.isnull().sum()
```

Out[8]:
```
status                     0
duration                   0
credit_history             0
purpose                    0
amount                     0
savings                    0
employment_duration        0
installment_rate           0
personal_status_sex        0
other_debtors              0
present_residence          0
property                   0
age                        0
other_installment_plans    0
housing                    0
number_credits             0
job                        0
people_liable              0
telephone                  0
foreign_worker             0
credit_risk                0
dtype: int64
```

In [9]:
```
sgc.describe().transpose()
```

Out[9]:

	count	mean	std	min	25%	50%	75%	max
status	800.0	2.64875	1.250931	1.0	1.75	2.0	4.00	4.0
duration	800.0	20.49625	12.006881	4.0	12.00	18.0	24.00	72.0
credit_history	800.0	2.58250	1.099866	0.0	2.00	2.0	4.00	4.0
purpose	800.0	2.78500	2.680533	0.0	1.00	2.0	3.00	10.0
amount	800.0	3210.29000	2792.840814	250.0	1364.00	2264.0	3907.25	18424.0
savings	800.0	2.14375	1.589416	1.0	1.00	1.0	3.00	5.0
employment_duration	800.0	3.39500	1.224070	1.0	3.00	3.0	5.00	5.0
installment_rate	800.0	2.95250	1.134395	1.0	2.00	3.0	4.00	4.0
personal_status_sex	800.0	2.68750	0.696743	1.0	2.00	3.0	3.00	4.0
other_debtors	800.0	1.14375	0.472615	1.0	1.00	1.0	1.00	3.0



In [10]:
```
sns.heatmap(sgc.isnull(),cmap='viridis')
plt.title('Sample Heat Map')
```

Out[10]:
```
Text(0.5, 1.0, 'Sample Heat Map')
```

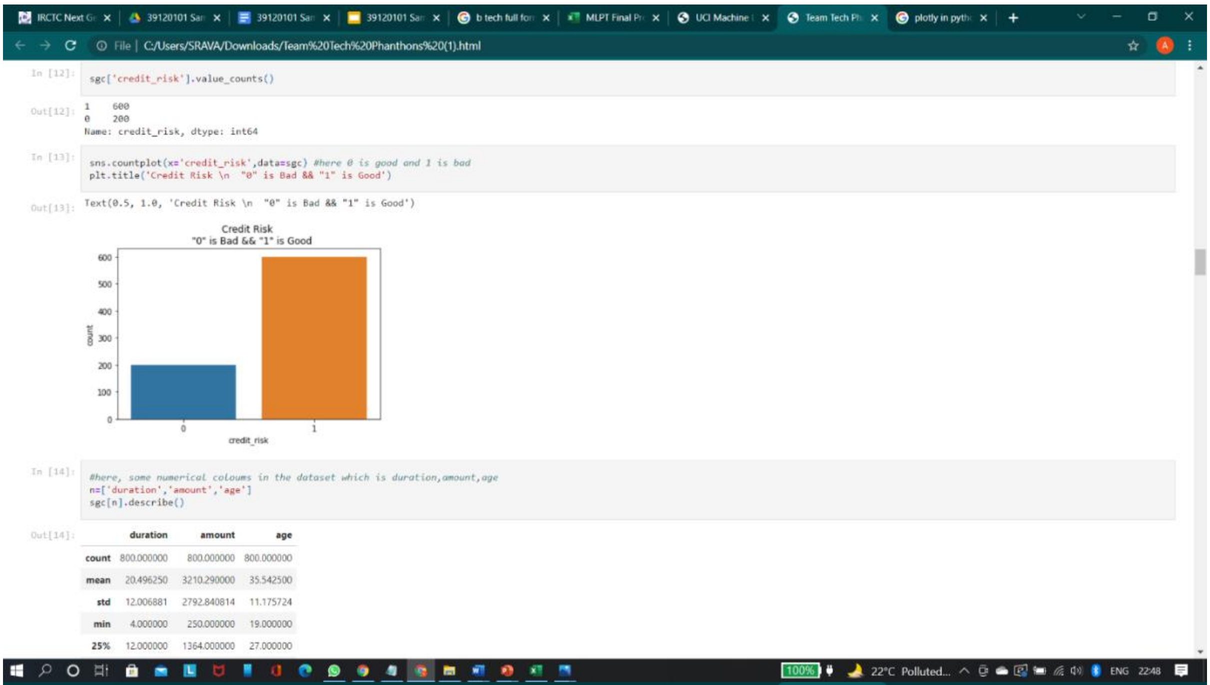
threfore there is no null values in the data set

DEEP ANALYSIS OF DATA SET

In [11]:
```
sgc.head()
```

Out[11]:

	status	duration	credit_history	purpose	amount	savings	employment_duration	installment_rate	personal_status_sex	other_debtors	...	property	age	other_installment_plans	housing	number_credits	job	people_liable	tele
0	1	18	4	2	1049	1	2	4	2	1	...	2	21	3	1	1	3	2	
1	1	9	4	0	2799	1	3	2	3	1	...	1	36	3	1	2	3	1	
2	2	12	2	9	841	2	4	2	2	1	...	1	23	3	1	1	2	2	



```
In [12]: sgc['credit_risk'].value_counts()
```

```
Out[12]: 1    600
         0    200
         Name: credit_risk, dtype: int64
```

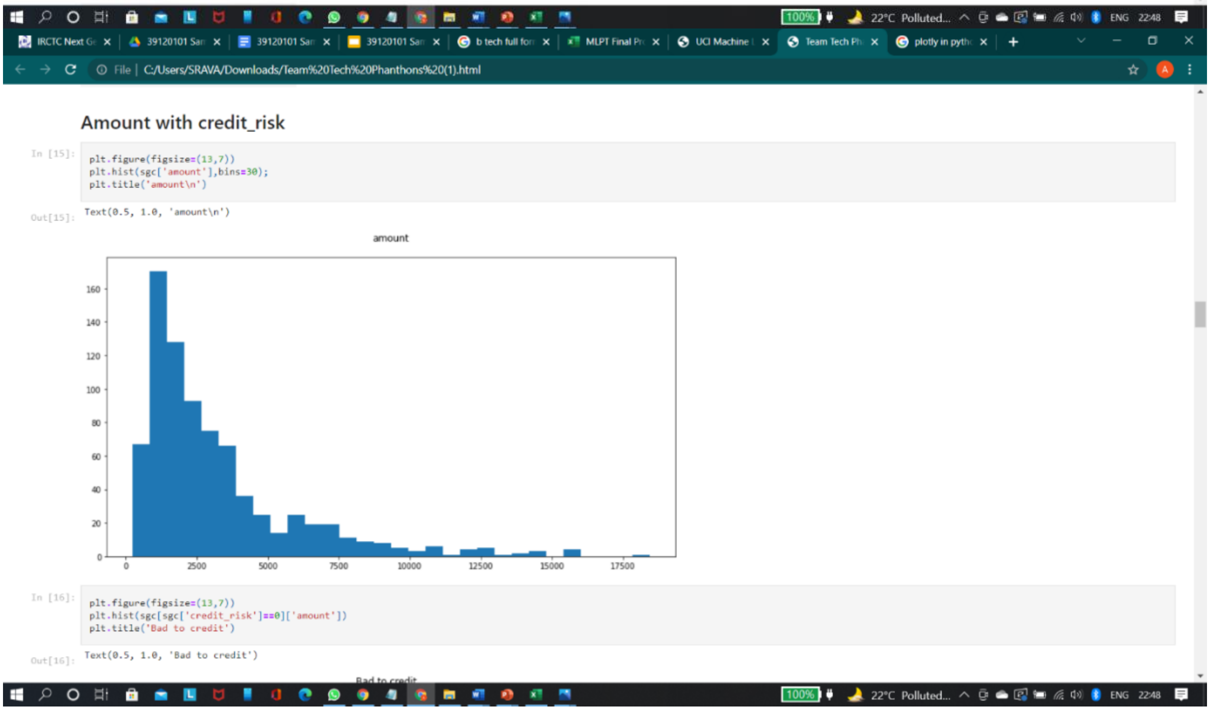
```
In [13]: sns.countplot(x='credit_risk',data=sgc) #here 0 is good and 1 is bad
         plt.title('Credit Risk \n  "0" is Bad && "1" is Good')
```

```
Out[13]: Text(0.5, 1.0, 'Credit Risk \n  "0" is Bad && "1" is Good')
```

```
In [14]: #here, some numerical coloums in the dataset which is duration,amount,age
         n=['duration','amount','age']
         sgc[n].describe()
```

Out[14]:

	duration	amount	age
count	800.000000	800.000000	800.000000
mean	20.496250	3210.290000	35.542500
std	12.006881	2792.840814	11.175724
min	4.000000	250.000000	19.000000
25%	12.000000	1364.000000	27.000000



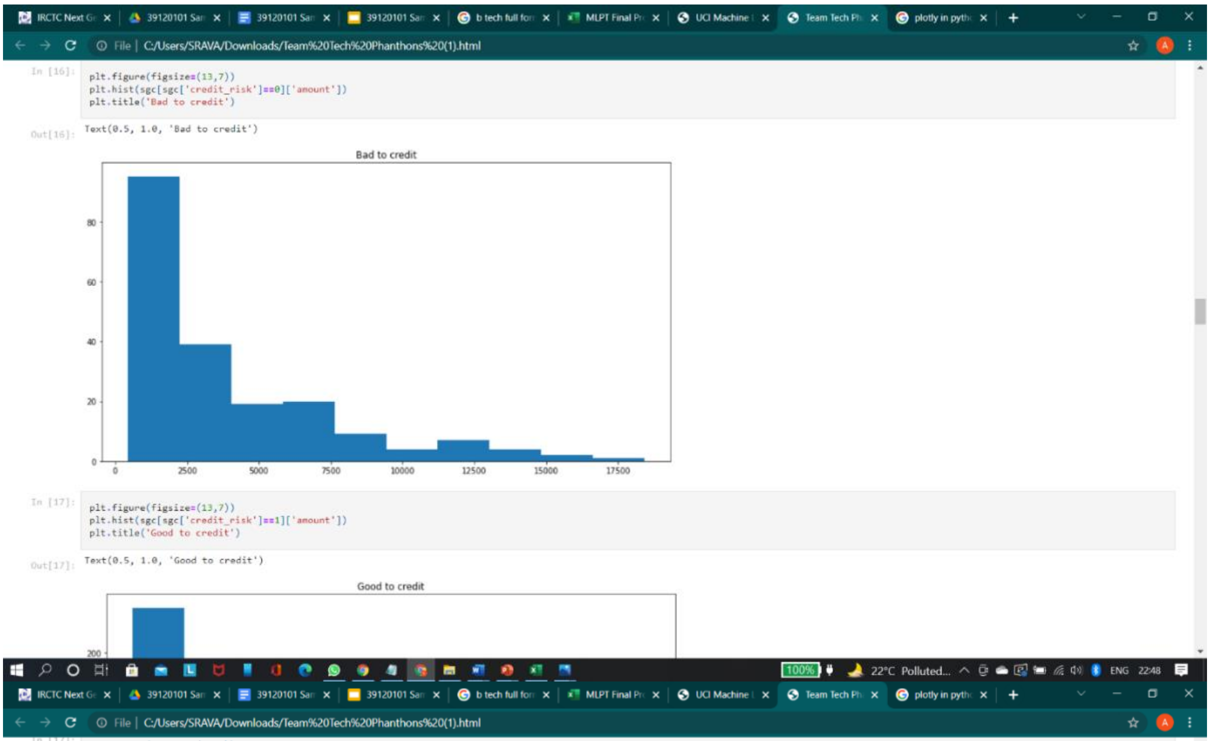
Amount with credit_risk

```
In [15]: plt.figure(figsize=(13,7))
         plt.hist(sgc['amount'],bins=30);
         plt.title('amount\n')
```

```
Out[15]: Text(0.5, 1.0, 'amount\n')
```

```
In [16]: plt.figure(figsize=(13,7))
         plt.hist(sgc[sgc['credit_risk']==0]['amount'])
         plt.title('Bad to credit')
```

```
Out[16]: Text(0.5, 1.0, 'Bad to credit')
```



In [16]:

```
plt.figure(figsize=(13,7))
plt.hist(sgc[sgc['credit_risk']==0]['amount'])
plt.title('Bad to credit')
```

Out[16]:

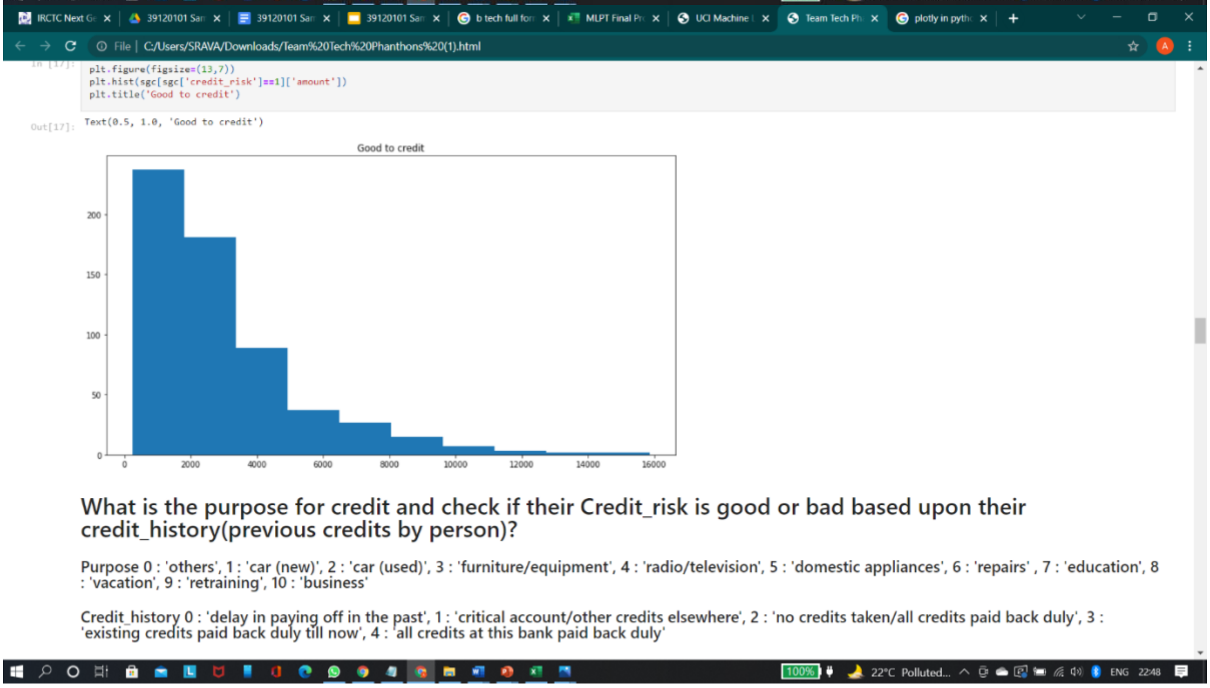
```
Text(0.5, 1.0, 'Bad to credit')
```

In [17]:

```
plt.figure(figsize=(13,7))
plt.hist(sgc[sgc['credit_risk']==1]['amount'])
plt.title('Good to credit')
```

Out[17]:

```
Text(0.5, 1.0, 'Good to credit')
```



```
plt.figure(figsize=(13,7))
plt.hist(sgc[sgc['credit_risk']==1]['amount'])
plt.title('Good to credit')
```

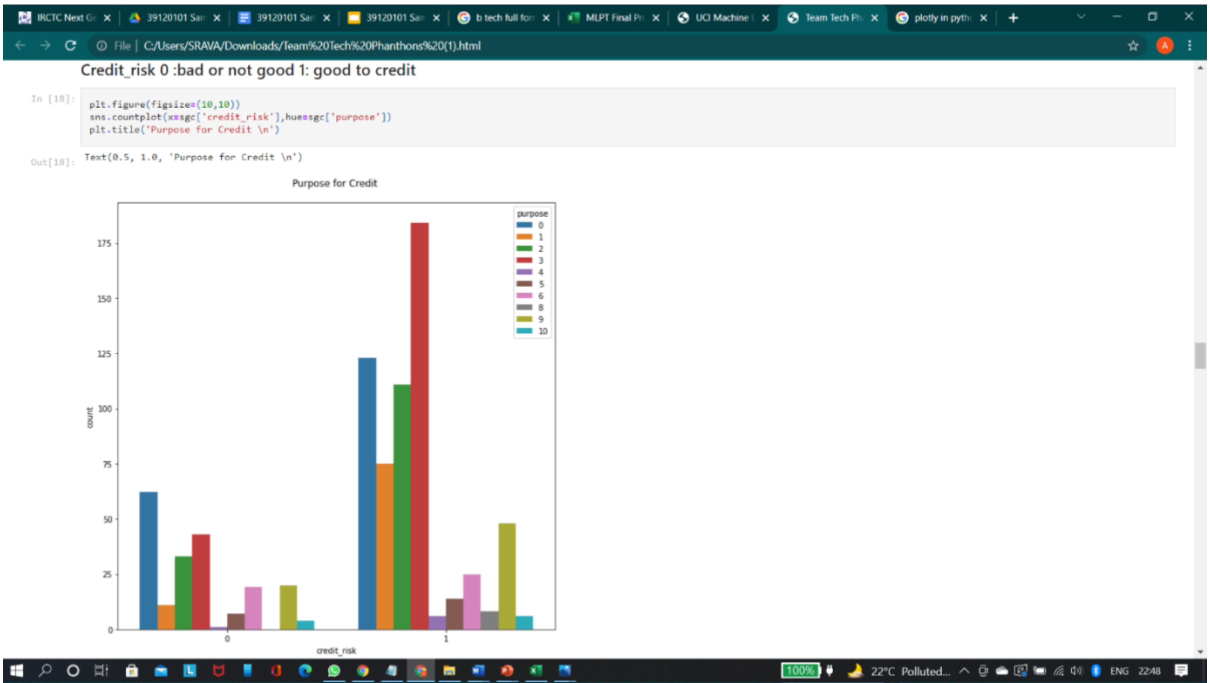
Out[17]:

```
Text(0.5, 1.0, 'Good to credit')
```

What is the purpose for credit and check if their Credit_risk is good or bad based upon their credit_history(previous credits by person)?

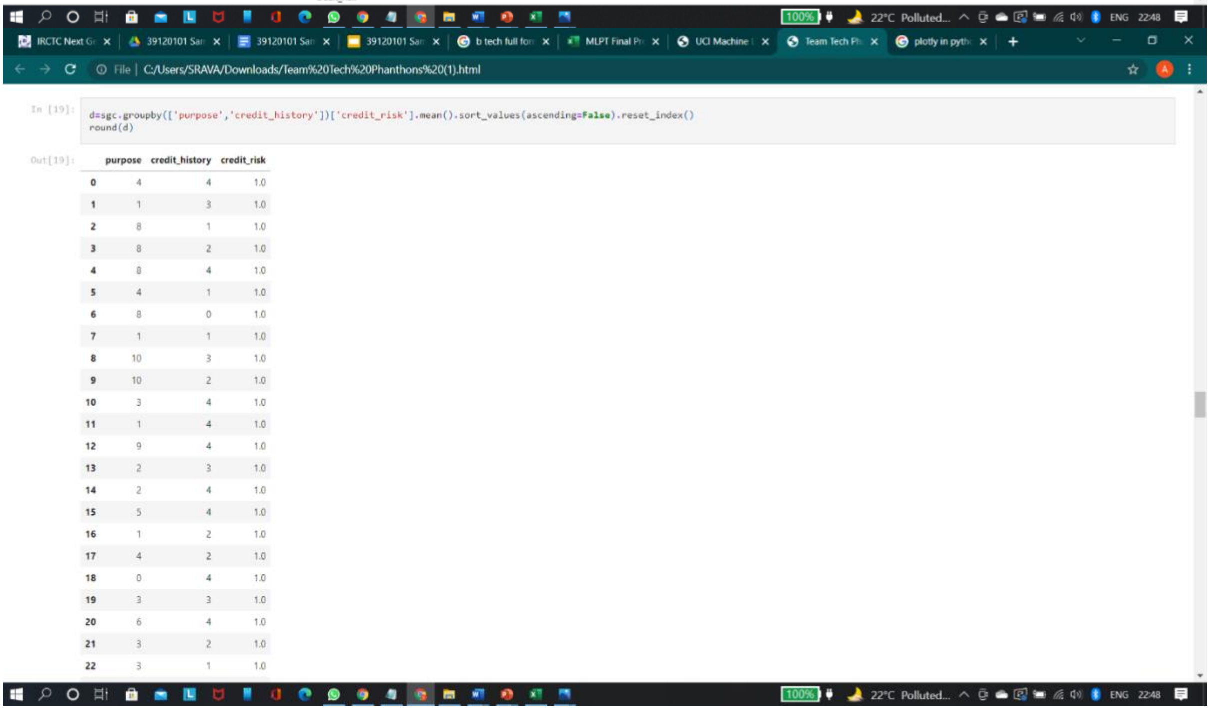
Purpose 0 : 'others', 1 : 'car (new)', 2 : 'car (used)', 3 : 'furniture/equipment', 4 : 'radio/television', 5 : 'domestic appliances', 6 : 'repairs' , 7 : 'education', 8 : 'vacation', 9 : 'retraining', 10 : 'business'

Credit_history 0 : 'delay in paying off in the past', 1 : 'critical account/other credits elsewhere', 2 : 'no credits taken/all credits paid back duly', 3 : 'existing credits paid back duly till now', 4 : 'all credits at this bank paid back duly'



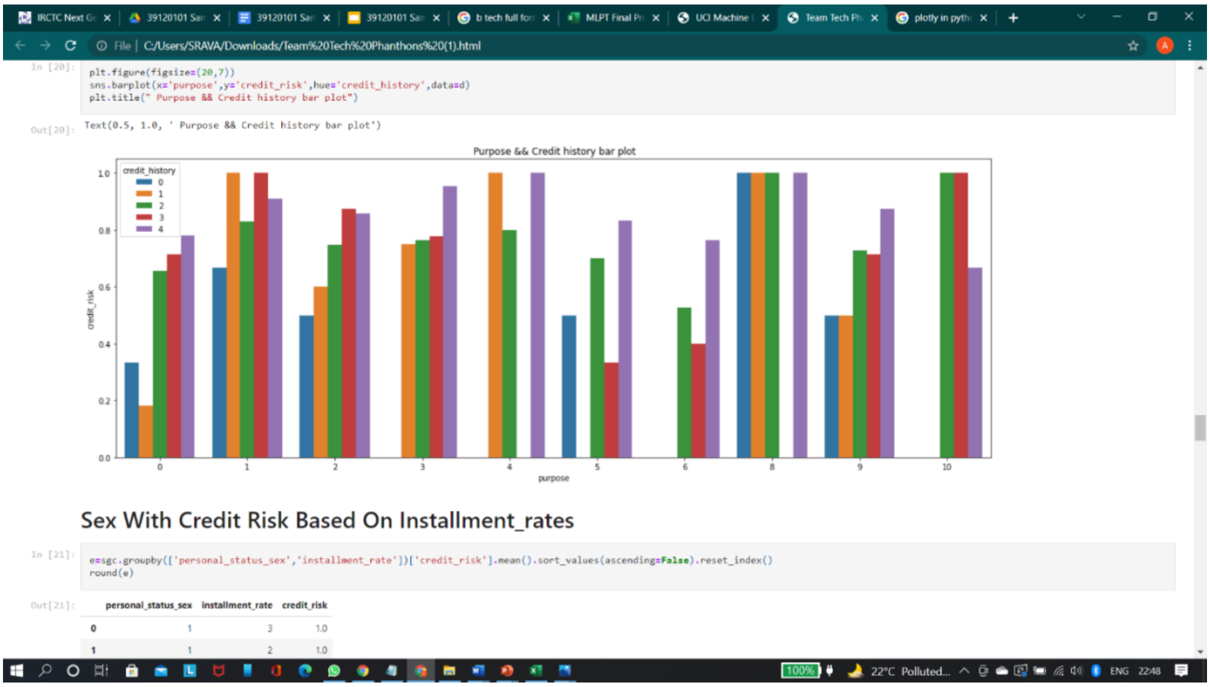
Credit_risk 0 :bad or not good 1: good to credit

```
In [18]: plt.figure(figsize=(10,10))
         sns.countplot(x=sgc['credit_risk'],hue=sgc['purpose'])
         plt.title('Purpose for Credit \n')

Out[18]: Text(0.5, 1.0, 'Purpose for Credit \n')
```



```
In [19]: d=sgc.groupby(['purpose','credit_history'])['credit_risk'].mean().sort_values(ascending=False).reset_index()
         round(d)

Out[19]:
```

	purpose	credit_history	credit_risk
0	4	4	1.0
1	1	3	1.0
2	8	1	1.0
3	8	2	1.0
4	8	4	1.0
5	4	1	1.0
6	8	0	1.0
7	1	1	1.0
8	10	3	1.0
9	10	2	1.0
10	3	4	1.0
11	1	4	1.0
12	9	4	1.0
13	2	3	1.0
14	2	4	1.0
15	5	4	1.0
16	1	2	1.0
17	4	2	1.0
18	0	4	1.0
19	3	3	1.0
20	6	4	1.0
21	3	2	1.0
22	3	1	1.0



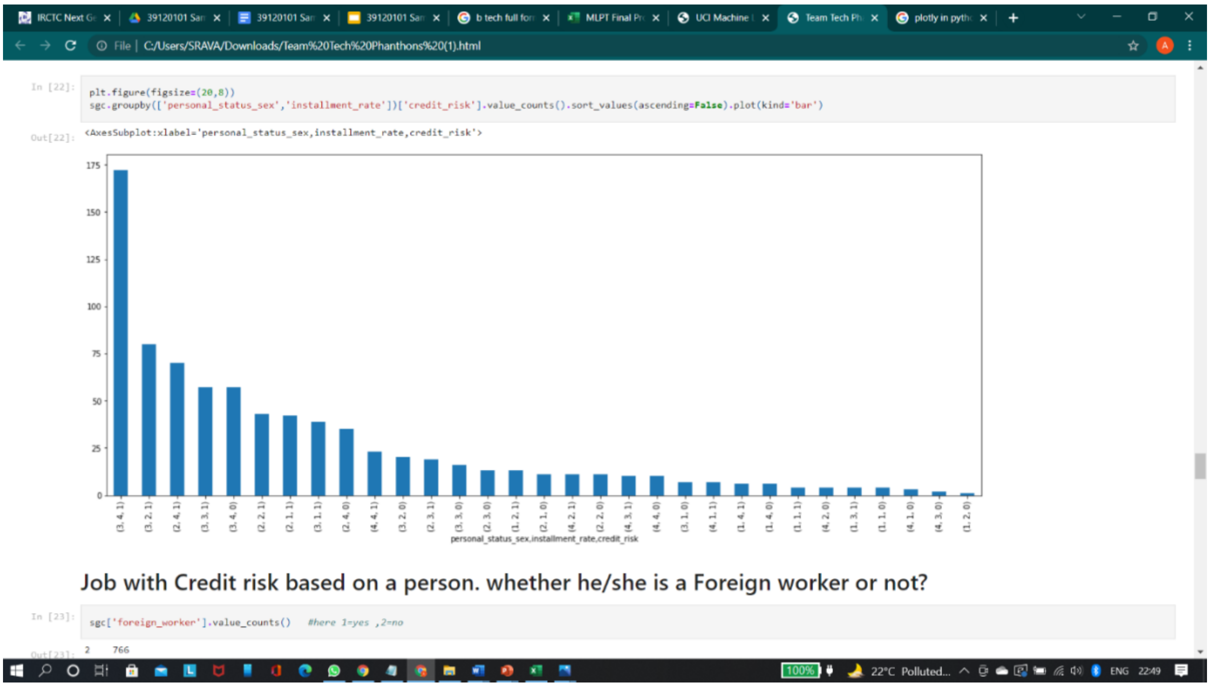
```
In [20]: plt.figure(figsize=(20,7))
         sns.barplot(x='purpose',y='credit_risk',hue='credit_history',data=d)
         plt.title(" Purpose && Credit history bar plot")

Out[20]: Text(0.5, 1.0, ' Purpose && Credit history bar plot')
```

Sex With Credit Risk Based On Installment_rates

```
In [21]: e=sgc.groupby(['personal_status_sex','installment_rate'])['credit_risk'].mean().sort_values(ascending=False).reset_index()
         round(e)
```

Out[21]:

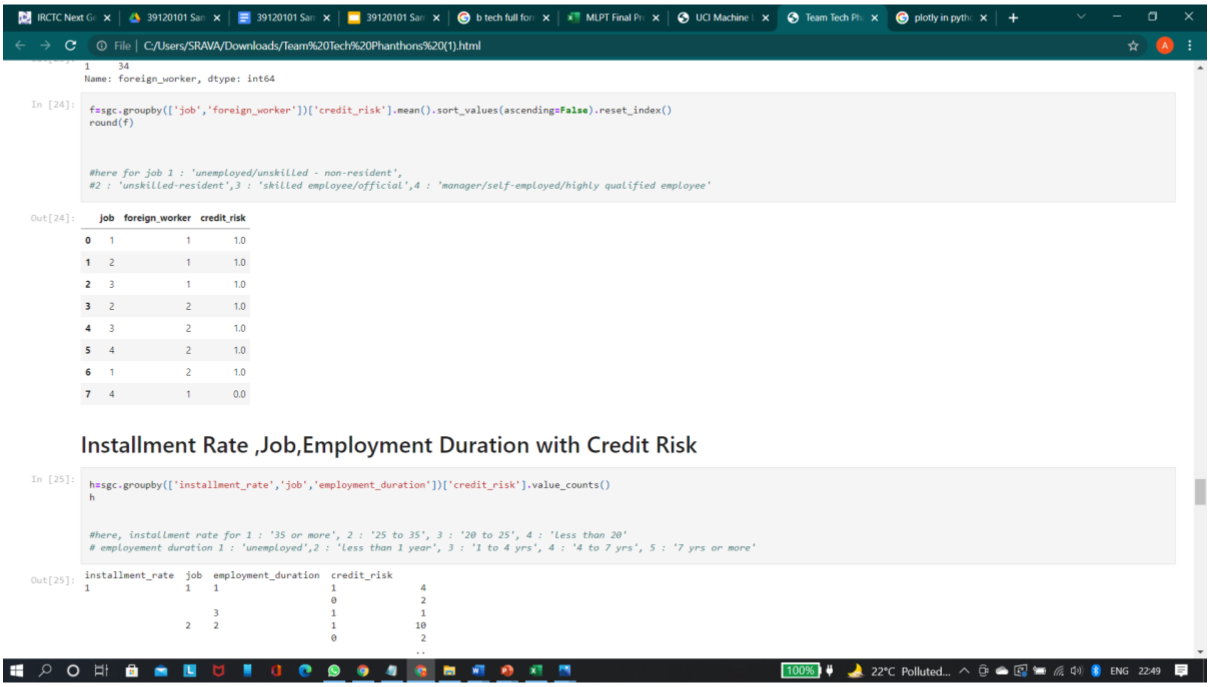
	personal_status_sex	installment_rate	credit_risk
0	1	3	1.0
1	1	2	1.0



```
In [22]: plt.figure(figsize=(20,8))
         sgc.groupby(['personal_status_sex','installment_rate'])['credit_risk'].value_counts().sort_values(ascending=False).plot(kind='bar')

Out[22]: <AxesSubplot:xlabel='personal_status_sex,installment_rate,credit_risk'>
```

Job with Credit risk based on a person. whether he/she is a Foreign worker or not?

```
In [23]: sgc['foreign_worker'].value_counts()   #here 1=yes ,2=no

Out[23]: 2     766
```



```
1      34
Name: foreign_worker, dtype: int64
```

In [24]:

```
f=sgc.groupby(['job','foreign_worker'])['credit_risk'].mean().sort_values(ascending=False).reset_index()
round(f)



#here for job 1 : 'unemployed/unskilled - non-resident',
#2 : 'unskilled-resident',3 : 'skilled employee/official',4 : 'manager/self-employed/highly qualified employee'
```

Out[24]:

	job	foreign_worker	credit_risk
0	1	1	1.0
1	2	1	1.0
2	3	1	1.0
3	2	2	1.0
4	3	2	1.0
5	4	2	1.0
6	1	2	1.0
7	4	1	0.0

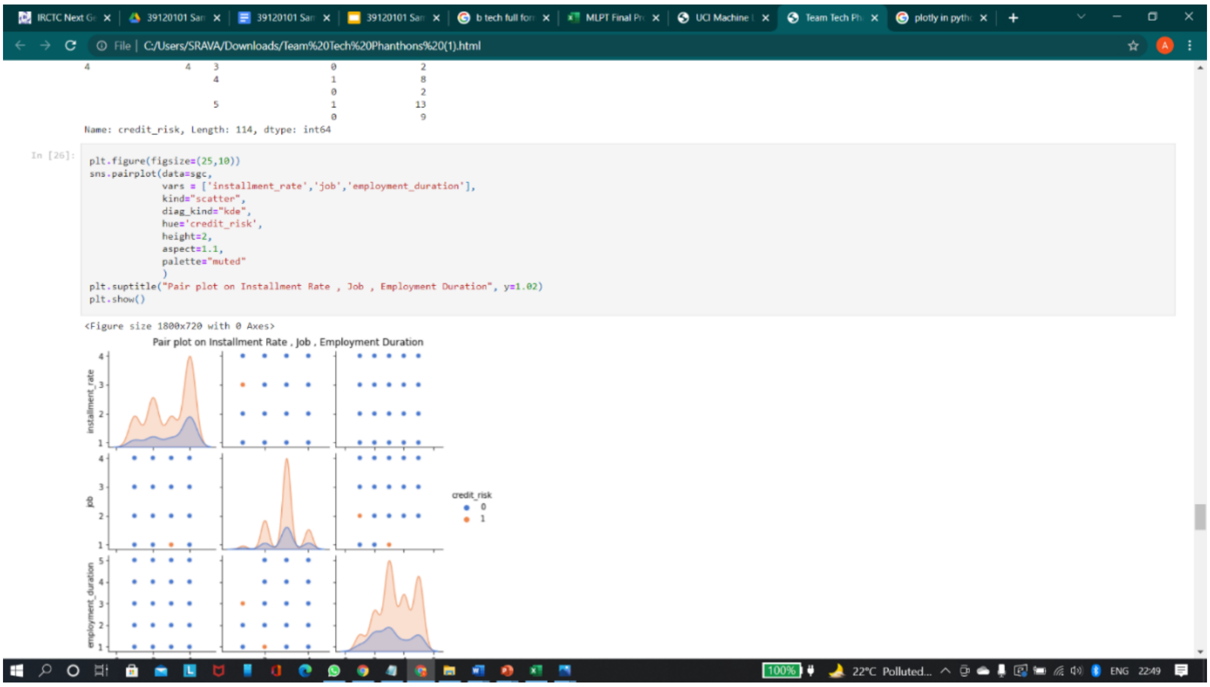
Installment Rate ,Job,Employment Duration with Credit Risk

In [25]:

```
h=sgc.groupby(['installment_rate','job','employment_duration'])['credit_risk'].value_counts()
h


#here, installment rate for 1 : '35 or more', 2 : '25 to 35', 3 : '20 to 25', 4 : 'less than 20'
# employement duration 1 : 'unemployed',2 : 'less than 1 year', 3 : '1 to 4 yrs', 4 : '4 to 7 yrs', 5 : '7 yrs or more'
```

Out[25]:

```
installment_rate  job  employment_duration  credit_risk
1                 1    1                    1              4
                                            0              2
                       3                    1              1
                  2    2                    1             10
                                            0              2
                                                          ..
```

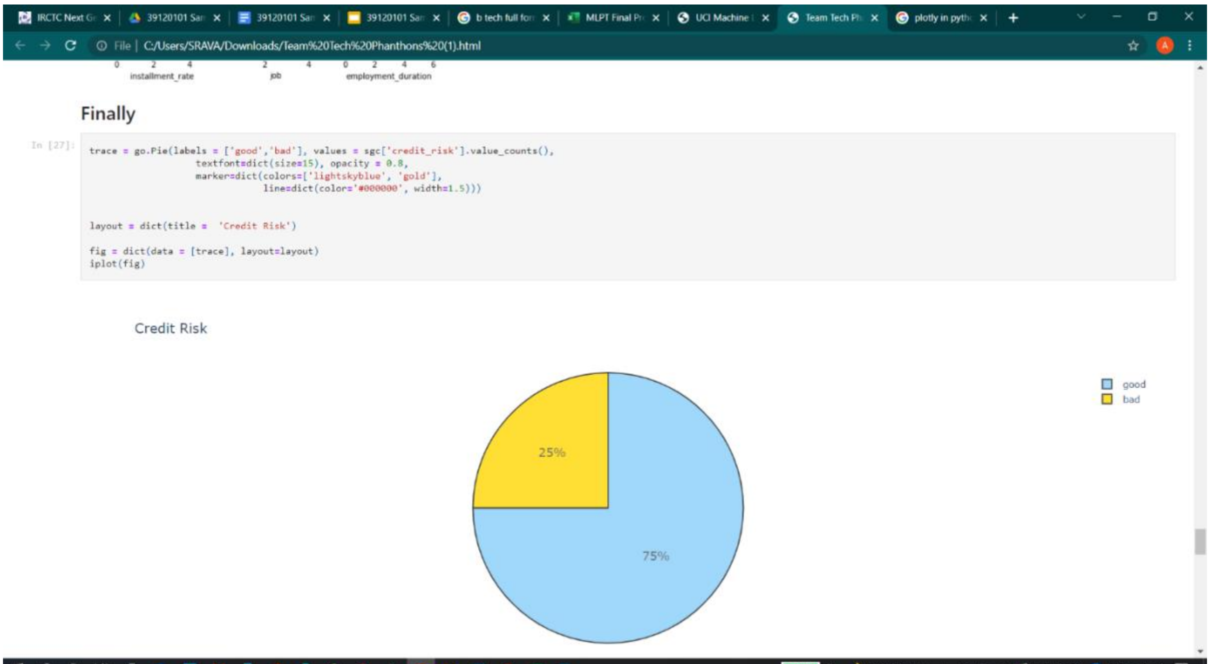


```
4                 4    3                    0              2
                       4                    1              8
                                            0              2
                       5                    1             13
                                            0              9
Name: credit_risk, Length: 114, dtype: int64
```

In [26]:

```
plt.figure(figsize=(25,10))
sns.pairplot(data=sgc,
             vars = ['installment_rate','job','employment_duration'],
             kind="scatter",
             diag_kind="kde",
             hue='credit_risk',
             height=2,
             aspect=1.1,
             palette="muted"
             )
plt.suptitle("Pair plot on Installment Rate , Job , Employment Duration", y=1.02)
plt.show()
```

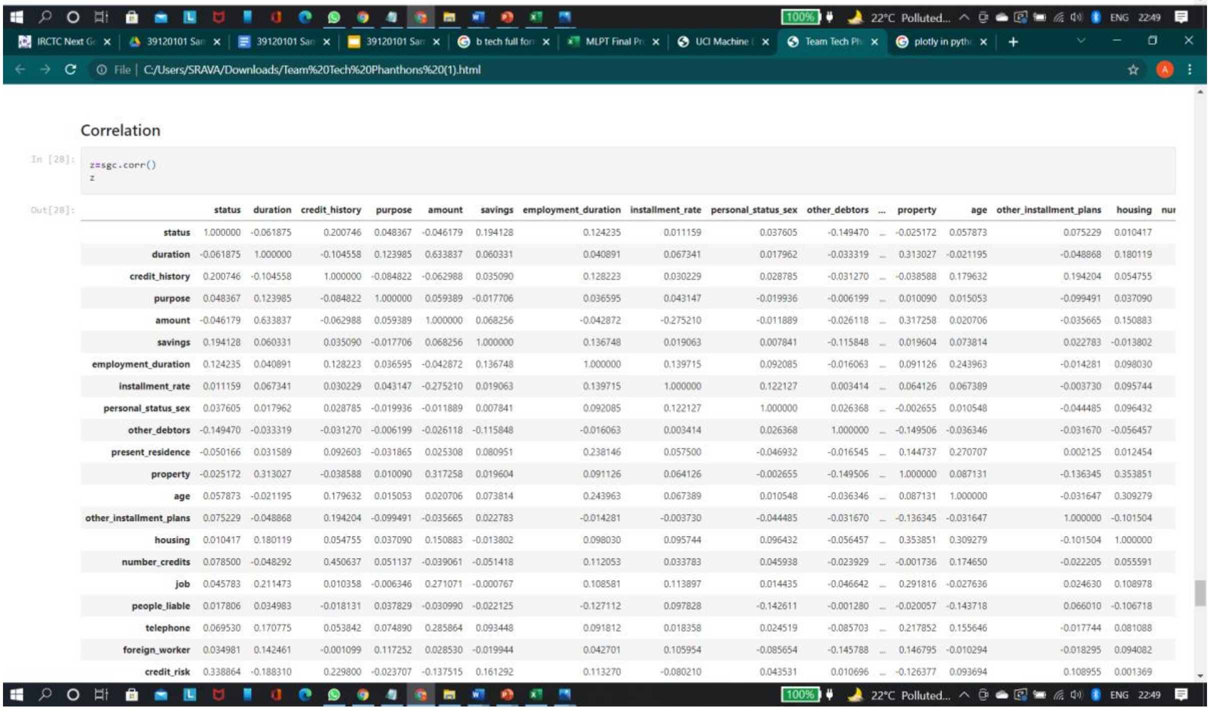
```
<Figure size 1800x720 with 0 Axes>
```



Finally

```
In [27]:
trace = go.Pie(labels = ['good','bad'], values = sgc['credit_risk'].value_counts(),
                    textfont=dict(size=15), opacity = 0.8,
                    marker=dict(colors=['lightskyblue', 'gold'],
                                line=dict(color='#000000', width=1.5)))


layout = dict(title =  'Credit Risk')

fig = dict(data = [trace], layout=layout)
iplot(fig)
```

Credit Risk

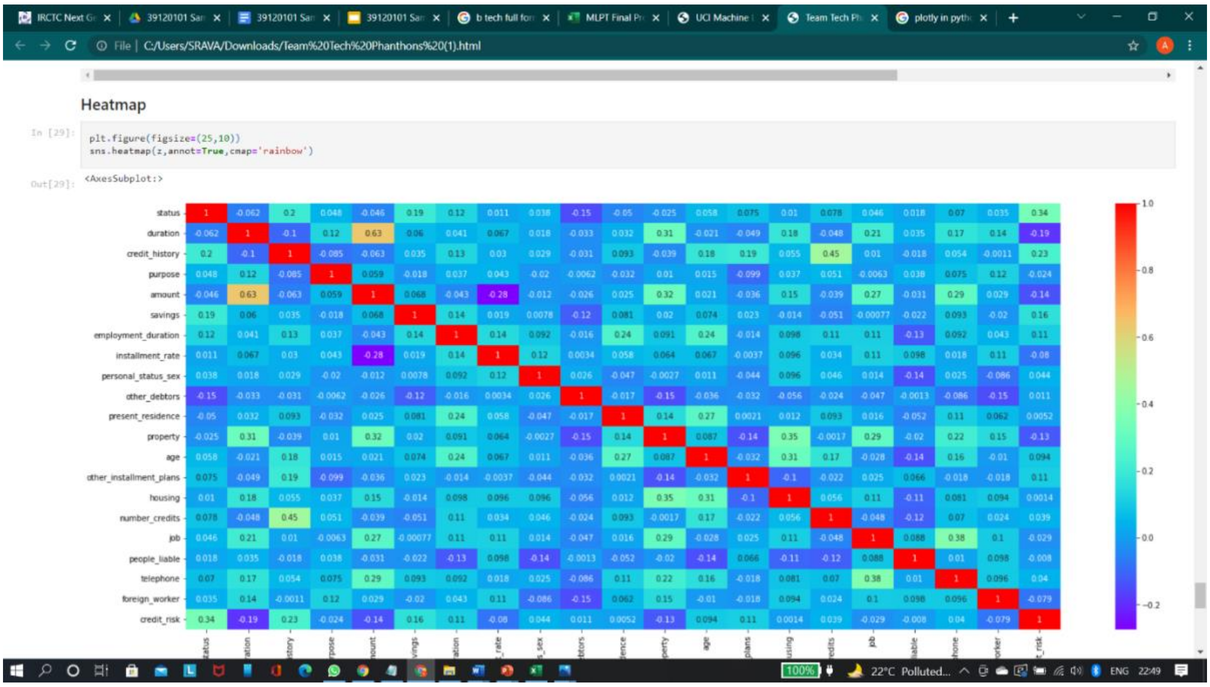


Correlation

```
In [28]:
z=sgc.corr()
z
```

Out[28]:

	status	duration	credit_history	purpose	amount	savings	employment_duration	installment_rate	personal_status_sex	other_debtors	...	property	age	other_installment_plans	housing	nu
status	1.000000	-0.061875	0.200746	0.048367	-0.046179	0.194128	0.124235	0.011159	0.037605	-0.149470	...	-0.025172	0.057873	0.075229	0.010417	
duration	-0.061875	1.000000	-0.104558	0.123985	0.633837	0.060331	0.040891	0.067341	0.017962	-0.033319	...	0.313027	-0.021195	-0.048868	0.180119	
credit_history	0.200746	-0.104558	1.000000	-0.084822	-0.062988	0.035090	0.128223	0.030229	0.028785	-0.031270	...	-0.038588	0.179632	0.194204	0.054755	
purpose	0.048367	0.123985	-0.084822	1.000000	0.059389	-0.017706	0.036595	0.043147	-0.019936	-0.006199	...	0.010090	0.015053	-0.099491	0.037090	
amount	-0.046179	0.633837	-0.062988	0.059389	1.000000	0.068256	-0.042872	-0.275210	-0.011889	-0.026118	...	0.317258	0.020706	-0.035665	0.150883	
savings	0.194128	0.060331	0.035090	-0.017706	0.068256	1.000000	0.136748	0.019063	0.007841	-0.115848	...	0.019604	0.073814	0.022783	-0.013802	
employment_duration	0.124235	0.040891	0.128223	0.036595	-0.042872	0.136748	1.000000	0.139715	0.092085	-0.016063	...	0.091126	0.243963	-0.014281	0.098030	
installment_rate	0.011159	0.067341	0.030229	0.043147	-0.275210	0.019063	0.139715	1.000000	0.122127	0.003414	...	0.064126	0.067389	-0.003730	0.095744	
personal_status_sex	0.037605	0.017962	0.028785	-0.019936	-0.011889	0.007841	0.092085	0.122127	1.000000	0.026368	...	-0.002655	0.010548	-0.044485	0.096432	
other_debtors	-0.149470	-0.033319	-0.031270	-0.006199	-0.026118	-0.115848	-0.016063	0.003414	0.026368	1.000000	...	-0.149506	-0.036346	-0.031670	-0.056457	
present_residence	-0.050166	0.031589	0.092603	-0.031865	0.025308	0.080951	0.238146	0.057500	-0.046932	-0.016545	...	0.144737	0.270707	0.002125	0.012454	
property	-0.025172	0.313027	-0.038588	0.010090	0.317258	0.019604	0.091126	0.064126	-0.002655	-0.149506	...	1.000000	0.087131	-0.136345	0.353851	
age	0.057873	-0.021195	0.179632	0.015053	0.020706	0.073814	0.243963	0.067389	0.010548	-0.036346	...	0.087131	1.000000	-0.031647	0.309279	
other_installment_plans	0.075229	-0.048868	0.194204	-0.099491	-0.035665	0.022783	-0.014281	-0.003730	-0.044485	-0.031670	...	-0.136345	-0.031647	1.000000	-0.101504	
housing	0.010417	0.180119	0.054755	0.037090	0.150883	-0.013802	0.098030	0.095744	0.096432	-0.056457	...	0.353851	0.309279	-0.101504	1.000000	
number_credits	0.078500	-0.048292	0.450637	0.051137	-0.039061	-0.051418	0.112053	0.033783	0.045938	-0.023929	...	-0.001736	0.174650	-0.022205	0.055591	
job	0.045783	0.211473	0.010358	-0.006346	0.271071	-0.000767	0.108581	0.113897	0.014435	-0.046642	...	0.291816	-0.027636	0.024630	0.108978	
people_liable	0.017806	0.034983	-0.018131	0.037829	-0.030990	-0.022125	-0.127112	0.097828	-0.142611	-0.001280	...	-0.020057	-0.143718	0.066010	-0.106718	
telephone	0.069530	0.170775	0.053842	0.074890	0.285864	0.093448	0.091812	0.018358	0.024519	-0.085703	...	0.217852	0.155646	-0.017744	0.081088	
foreign_worker	0.034981	0.142461	-0.001099	0.117252	0.028530	-0.019944	0.042701	0.105954	-0.085654	-0.145788	...	0.146795	-0.010294	-0.018295	0.094082	
credit_risk	0.338864	-0.188310	0.229800	-0.023707	-0.137515	0.161292	0.113270	-0.080210	0.043531	0.010696	...	-0.126377	0.093694	0.108955	0.001369	

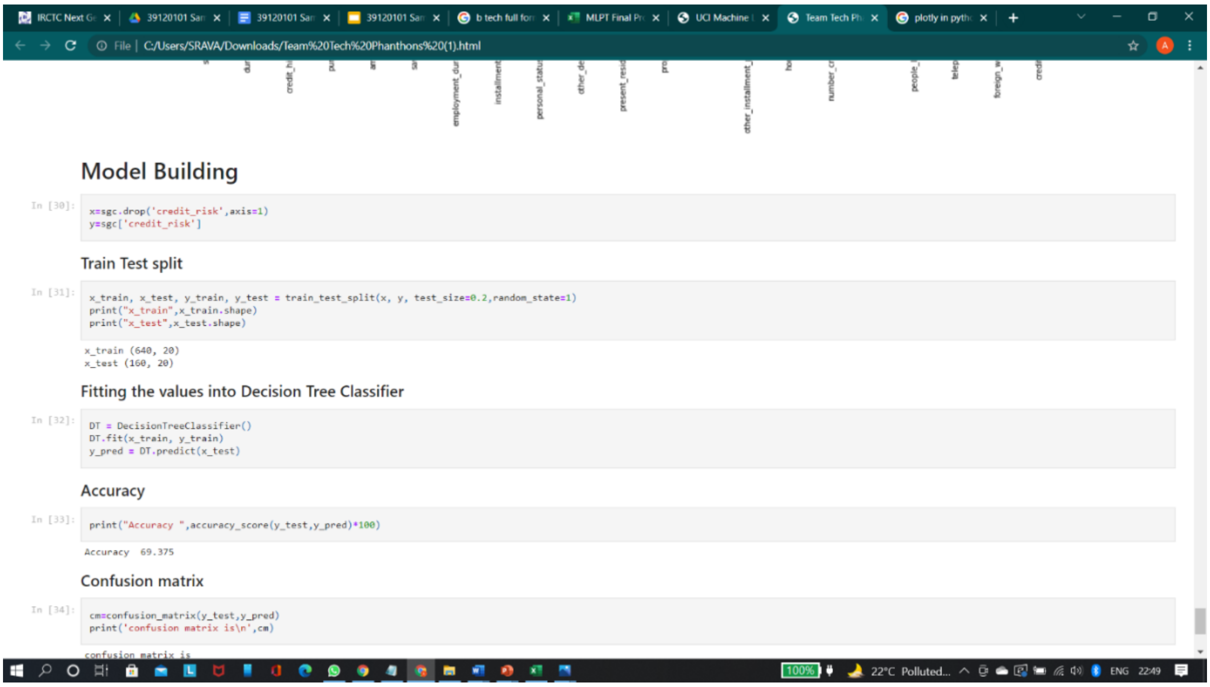


Heatmap

In [29]:
```
plt.figure(figsize=(25,10))
sns.heatmap(z,annot=True,cmap='rainbow')
```

Out[29]:
```
<AxesSubplot:>
```

	status	ation	story	rpose	nount	vings	ation	t_rate	s_sex	btors	dence	perty	age	plans	using	edits	job	liable	hone	orker	t_risk
status	1	-0.062	0.2	0.048	-0.046	0.19	0.12	0.011	0.038	-0.15	-0.05	-0.025	0.058	0.075	0.01	0.078	0.046	0.018	0.07	0.035	0.34
duration	-0.062	1	-0.1	0.12	0.63	0.06	0.041	0.067	0.018	-0.033	0.032	0.31	-0.021	-0.049	0.18	-0.048	0.21	0.035	0.17	0.14	-0.19
credit_history	0.2	-0.1	1	-0.085	-0.063	0.035	0.13	0.03	0.029	-0.031	0.093	-0.039	0.18	0.19	0.055	0.45	0.01	-0.018	0.054	-0.0011	0.23
purpose	0.048	0.12	-0.085	1	0.059	-0.018	0.037	0.043	-0.02	-0.0062	-0.032	0.01	0.015	-0.099	0.037	0.051	-0.0063	0.038	0.075	0.12	-0.024
amount	-0.046	0.63	-0.063	0.059	1	0.068	-0.043	-0.28	-0.012	-0.026	0.025	0.32	0.021	-0.036	0.15	-0.039	0.27	-0.031	0.29	0.029	-0.14
savings	0.19	0.06	0.035	-0.018	0.068	1	0.14	0.019	0.0078	-0.12	0.081	0.02	0.074	0.023	-0.014	-0.051	-0.00077	-0.022	0.093	-0.02	0.16
employment_duration	0.12	0.041	0.13	0.037	-0.043	0.14	1	0.14	0.092	-0.016	0.24	0.091	0.24	-0.014	0.098	0.11	0.11	-0.13	0.092	0.043	0.11
installment_rate	0.011	0.067	0.03	0.043	-0.28	0.019	0.14	1	0.12	0.0034	0.058	0.064	0.067	-0.0037	0.096	0.034	0.11	0.098	0.018	0.11	-0.08
personal_status_sex	0.038	0.018	0.029	-0.02	-0.012	0.0078	0.092	0.12	1	0.026	-0.047	-0.0027	0.011	-0.044	0.096	0.046	0.014	-0.14	0.025	-0.086	0.044
other_debtors	-0.15	-0.033	-0.031	-0.0062	-0.026	-0.12	-0.016	0.0034	0.026	1	-0.017	-0.15	-0.036	-0.032	-0.056	-0.024	-0.047	-0.0013	-0.086	-0.15	0.011
present_residence	-0.05	0.032	0.093	-0.032	0.025	0.081	0.24	0.058	-0.047	-0.017	1	0.14	0.27	0.0021	0.012	0.093	0.016	-0.052	0.11	0.062	0.0052
property	-0.025	0.31	-0.039	0.01	0.32	0.02	0.091	0.064	-0.0027	-0.15	0.14	1	0.087	-0.14	0.35	-0.0017	0.29	-0.02	0.22	0.15	-0.13
age	0.058	-0.021	0.18	0.015	0.021	0.074	0.24	0.067	0.011	-0.036	0.27	0.087	1	-0.032	0.31	0.17	-0.028	-0.14	0.16	-0.01	0.094
other_installment_plans	0.075	-0.049	0.19	-0.099	-0.036	0.023	-0.014	-0.0037	-0.044	-0.032	0.0021	-0.14	-0.032	1	-0.1	-0.022	0.025	0.066	-0.018	-0.018	0.11
housing	0.01	0.18	0.055	0.037	0.15	-0.014	0.098	0.096	0.096	-0.056	0.012	0.35	0.31	-0.1	1	0.056	0.11	-0.11	0.081	0.094	0.0014
number_credits	0.078	-0.048	0.45	0.051	-0.039	-0.051	0.11	0.034	0.046	-0.024	0.093	-0.0017	0.17	-0.022	0.056	1	-0.048	-0.12	0.07	0.024	0.039
job	0.046	0.21	0.01	-0.0063	0.27	-0.00077	0.11	0.11	0.014	-0.047	0.016	0.29	-0.028	0.025	0.11	-0.048	1	0.088	0.38	0.1	-0.029
people_liable	0.018	0.035	-0.018	0.038	-0.031	-0.022	-0.13	0.098	-0.14	-0.0013	-0.052	-0.02	-0.14	0.066	-0.11	-0.12	0.088	1	0.01	0.098	-0.008
telephone	0.07	0.17	0.054	0.075	0.29	0.093	0.092	0.018	0.025	-0.086	0.11	0.22	0.16	-0.018	0.081	0.07	0.38	0.01	1	0.096	0.04
foreign_worker	0.035	0.14	-0.0011	0.12	0.029	-0.02	0.043	0.11	-0.086	-0.15	0.062	0.15	-0.01	-0.018	0.094	0.024	0.1	0.098	0.096	1	-0.079
credit_risk	0.34	-0.19	0.23	-0.024	-0.14	0.16	0.11	-0.08	0.044	0.011	0.0052	-0.13	0.094	0.11	0.0014	0.039	-0.029	-0.008	0.04	-0.079	1



Model Building

In [30]:
```
x=sgc.drop('credit_risk',axis=1)
y=sgc['credit_risk']
```

Train Test split

In [31]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,random_state=1)
print("x_train",x_train.shape)
print("x_test",x_test.shape)
```

```
x_train (640, 20)
x_test (160, 20)
```

Fitting the values into Decision Tree Classifier

In [32]:
```
DT = DecisionTreeClassifier()
DT.fit(x_train, y_train)
y_pred = DT.predict(x_test)
```

Accuracy

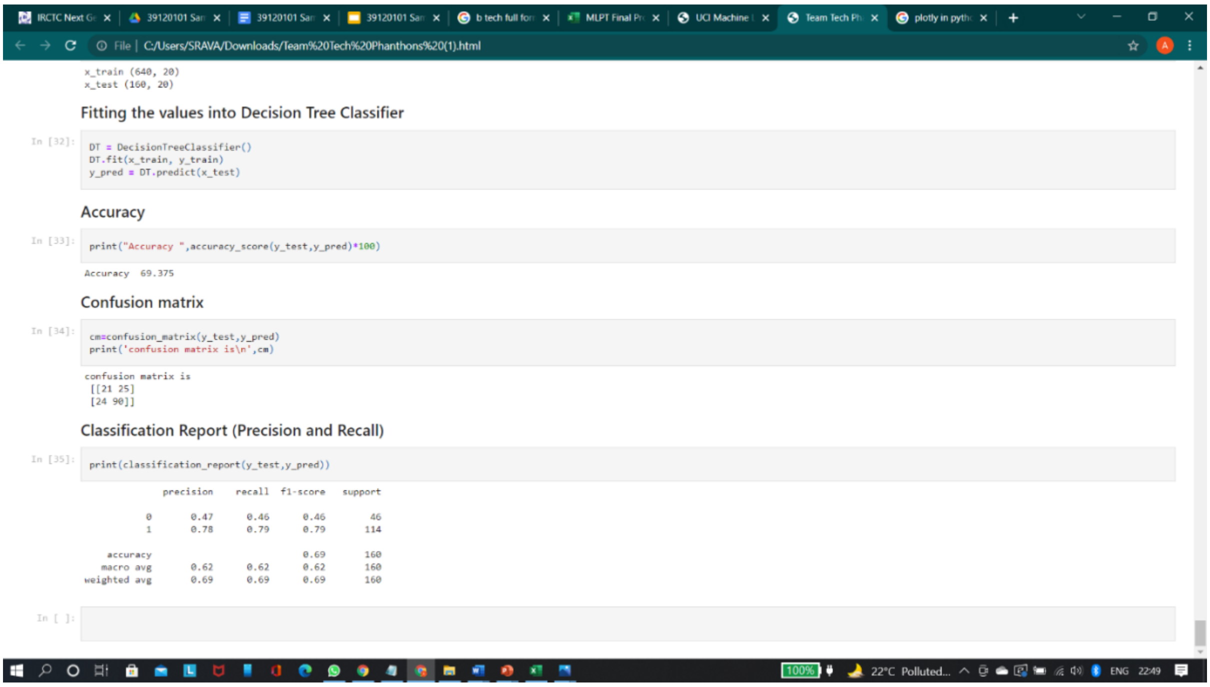
In [33]:
```
print("Accuracy ",accuracy_score(y_test,y_pred)*100)
```

```
Accuracy  69.375
```

Confusion matrix

In [34]:
```
cm=confusion_matrix(y_test,y_pred)
print('confusion matrix is\n',cm)
```

```
confusion matrix is
```



```
x_train (640, 20)
x_test (160, 20)
```

Fitting the values into Decision Tree Classifier

In [32]:
```
DT = DecisionTreeClassifier()
DT.fit(x_train, y_train)
y_pred = DT.predict(x_test)
```

Accuracy

In [33]:
```
print("Accuracy ",accuracy_score(y_test,y_pred)*100)
```

```
Accuracy  69.375
```

Confusion matrix

In [34]:
```
cm=confusion_matrix(y_test,y_pred)
print('confusion matrix is\n',cm)
```

```
confusion matrix is
 [[21 25]
 [24 90]]
```

Classification Report (Precision and Recall)

In [35]:
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.47      0.46      0.46        46
           1       0.78      0.79      0.79       114

    accuracy                           0.69       160
   macro avg       0.62      0.62      0.62       160
weighted avg       0.69      0.69      0.69       160
```

In [ ]:

SOURCE CODE :

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly
import plotly.express as px
import plotly.graph_objects as go
import plotly.offline as pyo
from plotly.offline import init_notebook_mode,plot,iplot
import plotly.figure_factory as ff


from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
sgc=pd.read_csv(r"F:\machine learning by cognibot\my project tech
phanthons\SGC.csv")
                                                            In [3]:
sgc.head()
sgc=sgc.drop('Id',axis=1) #Here,i am just droped the column "Id"
because, it is not usefull
                                                            In [5]:
columns
=['status','duration','credit_history','purpose','amount','savings','empl
oyment_duration','installment_rate',

'personal_status_sex','other_debtors','present_residence','property','
age','other_installment_plans','housing',

'number_credits','job','people_liable','telephone','foreign_worker','cre
dit_risk']
#here i just changed the column names from german to english for
understand
sgc.columns=columns
sgc.head()
sgc.info()
sgc.isnull().sum()
sgc.describe().transpose()
sns.heatmap(sgc.isnull(),cmap='viridis')
plt.title('Sample Heat Map')
sgc['credit_risk'].value_counts()
sns.countplot(x='credit_risk',data=sgc) #here 0 is good and 1 is bad
plt.title('Credit Risk \n  "0" is Bad && "1" is Good')
#here, some numerical coloums in the dataset which is
duration,amount,age
n=['duration','amount','age']
sgc[n].describe()
plt.figure(figsize=(13,7))
plt.hist(sgc['amount'],bins=30);
plt.title('amount\n')
plt.figure(figsize=(13,7))
plt.hist(sgc[sgc['credit_risk']==0]['amount'])
plt.title('Bad to credit')
plt.figure(figsize=(13,7))
plt.hist(sgc[sgc['credit_risk']==1]['amount'])
plt.title('Good to credit')
plt.figure(figsize=(10,10))
sns.countplot(x=sgc['credit_risk'],hue=sgc['purpose'])
plt.title('Purpose for Credit \n')
d=sgc.groupby(['purpose','credit_history'])['credit_risk'].mean().sort_
values(ascending=False).reset_index()
round(d)
plt.figure(figsize=(20,7))
sns.barplot(x='purpose',y='credit_risk',hue='credit_history',data=d)
plt.title(" Purpose && Credit history bar plot")
e=sgc.groupby(['personal_status_sex','installment_rate'])['credit_ris
k'].mean().sort_values(ascending=False).reset_index()
round(e)
plt.figure(figsize=(20,8))
sgc.groupby(['personal_status_sex','installment_rate'])['credit_risk'].
value_counts().sort_values(ascending=False).plot(kind='bar')
sgc['foreign_worker'].value_counts()   #here 1=yes ,2=no
f=sgc.groupby(['job','foreign_worker'])['credit_risk'].mean().sort_valu
es(ascending=False).reset_index()
round(f)
```

```
#here for job 1 : 'unemployed/unskilled - non-resident',
#2 : 'unskilled-resident',3 : 'skilled employee/official',4 : 'manager/self-employed/highly qualified employee'
h=sgc.groupby(['installment_rate','job','employment_duration'])['credit_risk'].value_counts()
h


#here, installment rate for 1 : '35 or more', 2 : '25 to 35', 3 : '20 to 25', 4 : 'less than 20'
# employement duration 1 : 'unemployed',2 : 'less than 1 year', 3 : '1 to 4 yrs', 4 : '4 to 7 yrs', 5 : '7 yrs or more'
plt.figure(figsize=(25,10))
sns.pairplot(data=sgc,
            vars = ['installment_rate','job','employment_duration'],
            kind="scatter",
            diag_kind="kde",
            hue='credit_risk',
            height=2,
            aspect=1.1,
            palette="muted"
            )
plt.suptitle("Pair plot on Installment Rate , Job , Employment Duration", y=1.02)
plt.show()
trace = go.Pie(labels = ['good','bad'], values = sgc['credit_risk'].value_counts(),
                  textfont=dict(size=15), opacity = 0.8,
                  marker=dict(colors=['lightskyblue', 'gold'],
                              line=dict(color='#000000', width=1.5)))


layout = dict(title =  'Credit Risk')

fig = dict(data = [trace], layout=layout)
iplot(fig)
```

Correlation

```
z=sgc.corr()
z
```

Heatmap

```
plt.figure(figsize=(25,10))
sns.heatmap(z,annot=True,cmap='rainbow')
```

Model Building

```
x=sgc.drop('credit_risk',axis=1)
y=sgc['credit_risk']
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.2,random_state=1)
print("x_train",x_train.shape)
print("x_test",x_test.shape)

DT = DecisionTreeClassifier()
DT.fit(x_train, y_train)
y_pred = DT.predict(x_test)
print("Accuracy ",accuracy_score(y_test,y_pred)*100)
cm=confusion_matrix(y_test,y_pred)
print('confusion matrix is\n',cm)
print(classification_report(y_test,y_pred))
```

--THE END--