Data pre-processing effect on the performance of machine learning algorithm

Project Report submitted in partial fulfilment of
The requirements for the degree of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

Of

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, WEST BENGAL

By

CHANCHALA KUMARI ----------------------------- Roll no→10900216059
ALOK BHOWMIK--------------------------------------Roll no→ 10900216079
ANKESH KUMAR ------------------------------------- Roll no→10900216072
CHIRASHREE KUNDU -------------------------------- Roll no→10900216057

Under the guidance of

MR CHANDAN BANERJEE

DEPARTMENT OF INFORMATION TECHNOLOGY

NETAJI SUBHASH ENGINEERING COLLEGE
TECHNO CITY, GARIA, KOLKATA – 700 152
Academic year of pass out 2019-20

CERTIFICATE

This is to certify that this project report titled **Data pre-processing effect on the performance of machine learning algorithm** submitted in partial fulfilment of requirements for award of the degree Bachelor of Technology (B. Tech) in Information Technology of MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY is a faithful record of the original work carried out by,

CHANCHALA KUMARI, **Roll no.**→10900216059 **Regd. No.** →161090110158 of 2016-17

ALOK BHOWMIK , **Roll no.**→10900216079 , **Regd. No.** →161090110138 of 2016-17

ANKESH KUMAR , **Roll no.**→10900216072 , **Regd. No.** →161090110145 of 2016-17

CHIRASHREE KUNDU , **Roll no.**→10900216057, **Regd. No.** →161090110160 of 2016-17

Under my guidance and supervision.

It is further certified that it contains no material, which to a substantial extent has been submitted for the award of any degree in any institute or has been published in any form, except the assistances drawn from other sources, for which due acknowledgement has been made.

____

Date:………..

Guide's signature

MR CHANDAN BANERJEE

Sd/__________________

Head of the Department

INFORMATION TECHNOLOGY
NETAJI SUBHASH ENGINEERING COLLEGE
TECHNO CITY, GARIA, KOLKATA – 700 152

DECLARATION

We hereby declare that this project report titled

Data pre-processing effect on the performance of machine learning algorithm
is our own original work carried out as a under graduate student in Netaji Subhash Engineering College except to the extent that assistances from other sources are duly acknowledged.

All sources used for this project report have been fully and properly cited. It contains no material which to a substantial extent has been submitted for the award of any degree in any institute or has been published in any form, except where due acknowledgement is made.

Student's names:	Signatures:	Dates:
Chanchala Kumari ………………………..	………………………..	…………………..
Alok Bhowmik ………………………..	………………………..	…………………..
Ankesh Kumar ………………………..	………………………..	…………………..
Chirashree Kundu ………………………..	………………………..	…………………..

CERTIFICATE OF APPROVAL

We hereby approve this dissertation titled

Data pre-processing effect on the performance of machine learning algorithm

carried out by

CHANCHALA KUMARI, **Roll no.**→10900216059 **Regd. No.** →161090110158 of 2016-17

ALOK BHOWMIK , **Roll no.**→10900216079 , **Regd. No.** →161090110138 of 2016-17

ANKESH KUMAR , **Roll no.**→10900216072 , **Regd. No.** →161090110145 of 2016-17

CHIRASHREE KUNDU , **Roll no.**→10900216057, **Regd. No.** →161090110160 of 2016-17

under the guidance of

MR CHANDAN BANERJEE

of Netaji Subhash Engineering College, Kolkata in partial fulfilment of requirements for award of the degree Bachelor of Technology (B. Tech) in INFORMATION TECHNOLOGY of

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY

Date:………..

Examiners' signatures:

1. …………………………………………..
2. …………………………………………..
3. …………………………………………..

Acknowledgement and/or Dedication

We would like to express our heartiest gratitude to our guide and mentor, Mr Chandan Banerjee, who gave us this golden opportunity of being a part of this wonderful and booming project. Along with that, we convey our heartiest regard to all professors of IT department who have always been there to guide us in spite of their busy time schedule.

Data pre-processing effect on the performance of machine learning algorithm

has helped us in understanding the basics machine learning algorithms in python environment for efficient the accuracy.

Finally we would also like to thank our parents along with our friends and acquaintances without whose cooperation and motivation this project wouldn't have been finalized within the limited time frame.

Along with that, we convey our heartiest regard to the professors who are ought to review this project of mine, having considered this to be worthy enough to occupy a slot in their busy time schedule.

Chanchala Kuamari

Alok Bhowmik

Ankesh Kumar

Chirashree Kundu

Dated:…………………

Abstract

Data pre-processing is a major and essential stage whose main goal is to obtain final data sets that can be considered correct and useful for further data mining algorithms. This paper summarizes the most influential data pre-processing algorithms according to their usage, popularity and extensions proposed in the specialized literature. For each algorithm, we provide a description, a discussion on its impact, and a review of current and further research on it. These most influential algorithms cover missing values imputation, noise filtering, dimensionality reduction (including feature selection and space transformations), instance reduction (including selection and generation), discretization and treatment of data for imbalanced pre-processing.

They constitute all among the most important topics in data pre-processing research and development. This paper also presents an illustrative study in two sections with different data sets that provide useful tips for the use of pre-processing algorithms. In the first place, we graphically present the effects on two benchmark data sets for the pre-processing methods.

CONTENTS

Chapter- 1:

Introduction

Machine learning:

Machine Learning is a current application of AI based around the idea that we should really just be able to give machines access to data and let them learn for themselves.

ML need large volume of data to find the pattern in them and learn

As machine learns from data, there are so many problems to learn from the data.So we write some algorithm to take care for that and

Getting computers to program themselves and also teaching them to make decision using data.

Data pre-processing

is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. Data -gathering methods are often loosely controlled, resulting in out-of- range values (e.g., Income: −100), impossible data combinations (e.g., Sex: Male, Pregnant: Yes), missing values, etc. Analysing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running an analysis. Often, data pre-processing is the most important phase of a machine learning project.

Data cleansing or **data cleaning**

It is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data. Data cleansing may be performed interactively with data wrangling tools, or as batch processing through scripting. Data pre-processing focuses on one of the most meaningful issues within the famous Knowledge Discovery from Data process. Data will likely have inconsistencies, errors, out of range values, impossible data combinations, missing values or most substantially, data is not suitable to start a DM process. In addition, the growing amount of data in current business applications, science, industry and academia, demands to the requirement of more complex mechanisms to analysis it.

Aim of the project

The aim of this project is to observe the effect of machine learning algorithm on the pre processed data. That is here in our project we will compute the predictive result by using the raw data and by using the processed data and then we will compare the predictive result between the raw data and processed data. So, we have taken one example of loan prediction for this comparison. We have taken one data set. The dataset is extracted from the official sites. . With the help of machine learning algorithm, using python as core we can predict if an applicant will get approved or not for the loan..This data seprediction using the features present in the dataset. The dataset is extracted from the official sites. With the help of machine learning algorithm, using python as core we can predict the suspected person should be arrested or not de

Dataset

Loan_ID	Gender	Married	Depender	Education	Self_Empl	Applicantl	Coapplica	LoanAmou	Loan_Amc	Credit_His	Property_Area
LP001015	Male	Yes	0	Graduate	No	5720	0	110	360	1	Urban
LP001022	Male	Yes	1	Graduate	No	3076	1500	126	360	1	Urban
LP001031	Male	Yes	2	Graduate	No	5000	1800	208	360	1	Urban
LP001035	Male	Yes	2	Graduate	No	2340	2546	100	360		Urban
LP001051	Male	No	0	Not Gradu	No	3276	0	78	360	1	Urban
LP001054	Male	Yes	0	Not Gradu	Yes	2165	3422	152	360	1	Urban
LP001055	Female	No	1	Not Gradu	No	2226	0	59	360	1	Semiurban
LP001056	Male	Yes	2	Not Gradu	No	3881	0	147	360	0	Rural
LP001059	Male	Yes	2	Graduate		13633	0	280	240	1	Urban
LP001067	Male	No	0	Not Gradu	No	2400	2400	123	360	1	Semiurban
LP001078	Male	No	0	Not Gradu	No	3091	0	90	360	1	Urban
LP001082	Male	Yes	1	Graduate		2185	1516	162	360	1	Semiurban
LP001083	Male	No	3+	Graduate	No	4166	0	40	180		Urban
LP001094	Male	Yes	2	Graduate		12173	0	166	360	0	Semiurban
LP001096	Female	No	0	Graduate	No	4666	0	124	360	1	Semiurban
LP001099	Male	No	1	Graduate	No	5667	0	131	360	1	Urban
LP001105	Male	Yes	2	Graduate	No	4583	2916	200	360	1	Urban
LP001107	Male	Yes	3+	Graduate	No	3786	333	126	360	1	Semiurban
LP001108	Male	Yes	0	Graduate	No	9226	7916	300	360	1	Urban

This is the actual dataset in which we are going to find out the accuracy using logistic regression ,before coming to this we must have a clear cut view on the topic so first began with that and after finding a perfect dataset from this data we will apply our algorithm.

Chapter- 2:

Categorical variable

In statistics, a **categorical variable** is a variable that can take on one of a limited, and usually fixed, number of possible values, assigning each individual or other unit of observation to a particular group or nominal category on the basis of some qualitative property. In computer science and some branches of mathematics, categorical variables are referred to as enumerations or enumerated types. Commonly (though not in this article), each of the possiblevalues of a categorical variable is referred to as a level.

is thestatistical data typeconsisting of categorical variables orof data that has been converted into that form, for example as grouped data. More specifically, categorical data may derive from observations made of qualitative data that are summarised as counts or cross tabulations, or fromobservations of quantitative data grouped within given intervals. Often, purely categorical data are summarised in the form of a contingency table.

Categorical data remove

- Null value remove :

  fillna() **function** to fill out the missing values in the given series object using forward fill (ffill) **method**. Output : ... **fillna**() **function** to fill out the missing values in the given series object. We will use forward fill **method** to fill out the missing values.

```
In [13]: loan.isnull().sum()

Out[13]: Loan_ID              0
         Gender               0
         Married              0
         Dependents           0
         Education            0
         Self_Employed        0
         ApplicantIncome      0
         CoapplicantIncome    0
         LoanAmount           0
         Loan_Amount_Term     0
         Credit_History       0
         Property_Area        0
         dtype: int64
```

- Convert to Number:

Label Encoder: It is used to transform non-numerical labels to numerical labels (or nominal categorical variables). Numerical labels are always between 0 and n classes-

```
]: from sklearn.preprocessing import LabelEncoder
lc=LabelEncoder()
x=lc.fit_transform(loan['Property_Area'])
x=pd.DataFrame(x)
loan['Property_Area']=x
loan.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount
0	0	1	1	0	0	0	5720	0	110
1	1	1	1	1	0	0	3076	1500	126
2	2	1	1	2	0	0	5000	1800	208
3	3	1	1	2	0	0	2340	2546	100
4	4	1	0	0	1	0	3276	0	78

Data visualization

Data visualization is the discipline of trying to understand data by placing it in a visual context

Python offers multiple great graphing libraries that come packed with lots of different features

Can be done with the help of

SEABORN

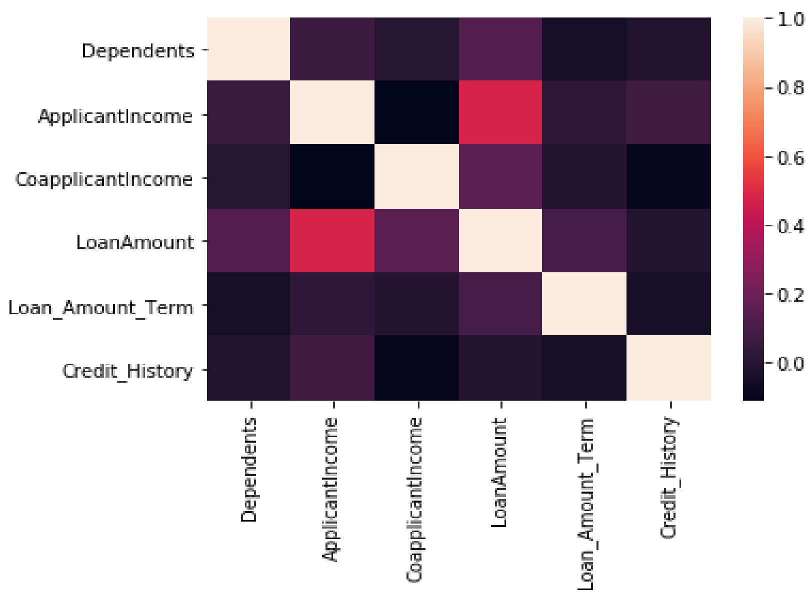
MATPOTLIB

SEABORN

heat map :heat map is seaborn graph . It is a two-dimensional representation of data in which values are represented by colors. A simple heat map provides an immediate visual summary of information. More elaborate heat maps allow the viewer to understand complex data sets.

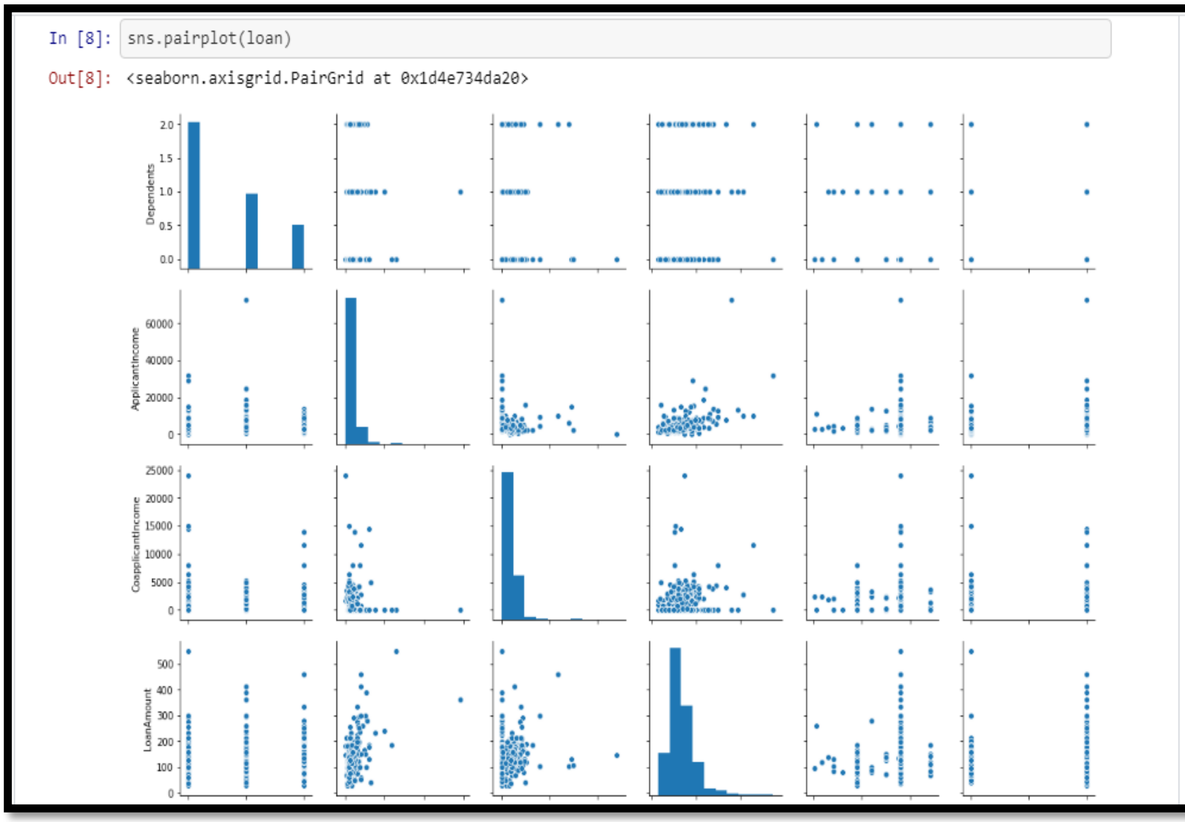
There can be many ways to display heat maps, but they all share one thing in common -- they use color to communicate relationships between data values that would be would be much harder to understand if presented numerically in a spreadsheet

```
In [7]: import seaborn as sns
        sns.heatmap(c)
        plt.show()
```



MATPOTLIB

Pairplot: It is a pair wise relationships in a dataset. This function creates a grid of Axes such that each variable in data will by shared in the y-axis across a single row and in the x-axis across a single column.



Pair plot will only plot the variables which are numerical. The variables which are of String type, by default pair plot won't plot automatically.

If i want plot that non -integer variable in graph then I have to explicitly mention in parameter

Combine Levels

- **Combine levels:** To avoid redundant levels in a categorical variable and to dealwith rare levels, we can simply combine the different levels. There are various methods of combining levels. Here are commonly used ones:

Using Business Logic: It is one of the most effective method of combining levels. It makes sense also to combine similar levels into similar groups based on domain or business experience. For example, we can combine levels of a variable “zip code” at state or district level.

Zip Code	District
110044	South Delhi
110048	South Delhi
110049	South Delhi
110006	North Delhi
110007	North Delhi
110058	West Delhi
110059	West Delhi
110063	West Delhi
110064	West Delhi

Using frequency or response rate: Combining levels based on business logic is effective but we may always not have the domain knowledge. Imagine, you are given a data set from Aerospace Department, US Govt. How would you apply business logic here? In such cases, we combine levels by considering the frequency distribution or response rate.

To combine levels using their frequency, we first look at the frequency distribution of each level and combine levels having frequency less than 5% of total observation (5% is standard but you can change it based on distribution).
This is an effective method to deal with rare levels.

We can also combine levels by considering the response rate of each level. We can simply combine levels having similar response rate into same group.

Based on Frequency

Levels	Frequency	New_Level
HA001	9%	HA001
HA002	12%	HA002
HA003	4%	New
HA004	1%	New
HA005	3%	New
HA006	11%	HA006
HA007	1%	New
HA008	4%	New
HA009	10%	HA009
HA010	4%	New
HA011	8%	HA011
HA012	12%	HA012
HA013	3%	New
HA014	11%	HA014
HA015	2%	New
HA016	4%	New
HA017	0%	New

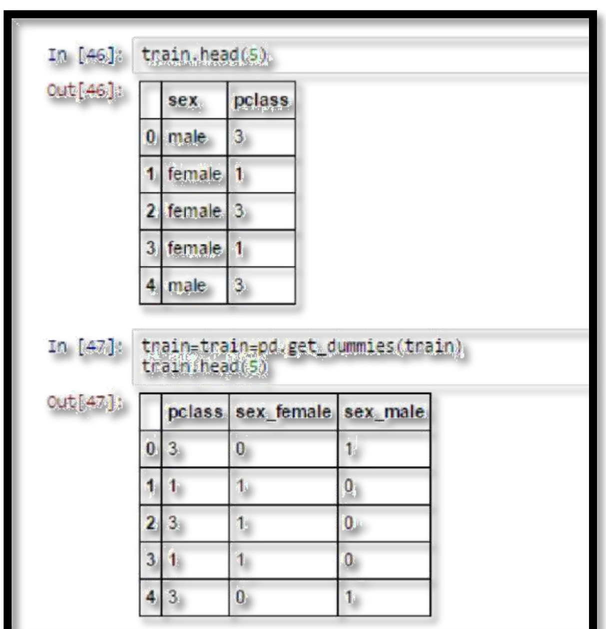
Based on Response Rate

Levels	Response_Rate	New_Level
HA014	98%	1
HA001	97%	1
HA003	93%	1
HA009	81%	2
HA015	75%	3
HA010	73%	3
HA006	66%	4
HA017	60%	4
HA007	49%	5
HA004	36%	6
HA005	31%	6
HA012	28%	7
HA008	25%	7
HA013	23%	7
HA016	22%	7
HA002	21%	8
HA011	5%	9

Based on Frequency and Response Rate

Levels	Frequency	Response_Rate	New_Level1	New_Level2
HA014	11%	98%	1	1
HA001	9%	97%	1	1
HA003	4%	93%	1	1
HA009	10%	81%	2	2
HA015	2%	75%	3	2
HA010	4%	73%	3	2
HA006	11%	66%	4	4
HA017	0%	60%	4	4
HA007	1%	49%	5	4
HA004	1%	36%	6	4
HA005	3%	31%	6	4
HA012	12%	28%	7	7
HA008	4%	25%	7	7
HA013	3%	23%	7	7
HA016	4%	22%	7	7
HA002	12%	21%	8	8
HA011	8%	5%	9	9

Dummy Coding

- **Dummy Coding:** Dummy coding is a commonly used method for convertinga categorical input variable into continuous variable. 'Dummy', as the name suggests is a duplicate variable which represents one level of a categorical variable. Presence of a level is represent by 1 and absence is represented by 0. For every level present, one dummy variable will be created. Look at the representation below to convert a categorical variable using dummy variable.
- 

In [46]:
```
train.head(5)
```
Out[46]:

	sex	pclass
0	male	3
1	female	1
2	female	3
3	female	1
4	male	3

In [47]:
```
train=train=pd.get_dummies(train)
train.head(5)
```
Out[47]:

	pclass	sex_female	sex_male
0	3	0	1
1	1	1	0
2	3	1	0
3	1	1	0
4	3	0	1

Chapter- 3:

IMPUTER

3.1 WHAT IS IMPUTER?

In statistics, **imputation** is the process of replacing missing data with substituted values. When substituting for a data point, it is known as "unit imputation"; when substituting for a component of a data point, it is known as "item imputation".

There are three main problems that missing data causes: missing data can introduce a substantial amount of bias, make the handling and analysis of the data more arduous, and create reductions in efficiency. Because missing data can create problems for analysing data, imputation is seen as a way to avoid pitfalls involved with list wise deletion of cases that have missing values.

That is to say, when one or more values are missing for a case, most statistical packages default to discarding any case that has a missingvalue, which may introduce bias or affect the representativeness of the results.

Imputation preserves all cases by replacing missing data with an estimated value based on other available information. Once all missing values have been imputed, the data set can then be analysed using standard techniques for complete data. Imputation theory is constantly developing and thus requires consistent attention to new information regarding the subject.

Chapter- 4:

Cross Validation

WHAT IS CROSS VALIDATION ?

Cross Validation is a technique which involves reserving a particular sample of a dataset on which you do not train the model. Later, you test your model on this sample before finalizing it.

Here are the steps involved in cross validation:

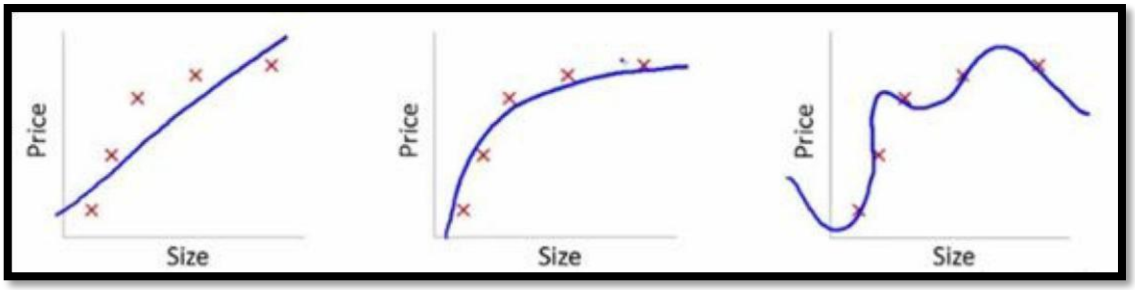
1. You *reserve* a sample data set
2. Train the model using the remaining part of the dataset
3. Use the reserve sample of the test (validation) set. This will help you in gauging the effectiveness of your model's performance. If your model delivers a positive result on validation data, go ahead with the current model. It rocks!

The validation set approach

In this approach, we reserve 50% of the dataset for validation and the remaining 50% for model training. However, a major disadvantage of this approach is that since we are training a model on only 50% of the dataset, there is a huge possibility that we might miss out on some interesting information about the data which will lead to a higher bias.

4.3 Why do models lose stability?

Let's understand this using the below snapshot illustrating the fit of various models:



Here, we are trying to find the relationship between size and price. To achieve this, we have taken the following steps:

1. We've established the relationship using a linear equation for which the plots have been shown. The first plot has a high error from training data points. Therefore, this will not perform well on either public or the private leader board. This is an example of "**Under fitting".** In this case, our model fails to capture the underlying trend of the data
2. In the second plot, we just found the right relationship between price and size, i.e., low training error and generalization of the relationship
3. In the third plot, we found a relationship which has almost zero training error. This is because the relationship is developed by considering each deviation in the data point (including noise), i.e., the model is too sensitive and captures random patterns which are present only in the current dataset.
   This is an example of "**Over fitting**". In this relationship, there could be a high deviation between the public and private leader boards.

Chapter- 5:

FEATURE SELECTION

WHAT IS FEATURE SELECTION ?

In machine learning and statistics, **feature selection**, also known as **variable selection**, **attribute selection** or **variable subset selection**, is the process of selecting a subset of relevant features (variables, predictors) for use in model construction. Feature selection techniques are used for four reasons:

- simplification of models to make them easier to interpret by researchers/users,
- shorter training times,
- to avoid the curse of dimensionality,
- enhanced generalization by reducing over fitting (formally,
- reduction of variance

The central premise when using a feature selection technique is that the data contains many features that are either *redundant* or *irrelevant*, and can thus be removed without incurring much loss of information.*Redundant* or *irrelevant* features are two distinct notions, since one relevant feature may beredundant in the presence of another relevant feature with which it is strongly correlated.

EXAMPLES

EXAMPLE:

```
import sklearn.feature_selection

select = sklearn.feature_selection.SelectKBest(k=20)
selected_features = select.fit(X_train, y_train) indices_selected =
selected_features.get_support(indices=True)
colnames_selected = [X.columns[i] for i in indices_selected]

X_train_selected = X_train[colnames_selected]
X_test_selected = X_test[colnames_selected]
```

Chapter- 6:

LOGISTIC REGRESSION

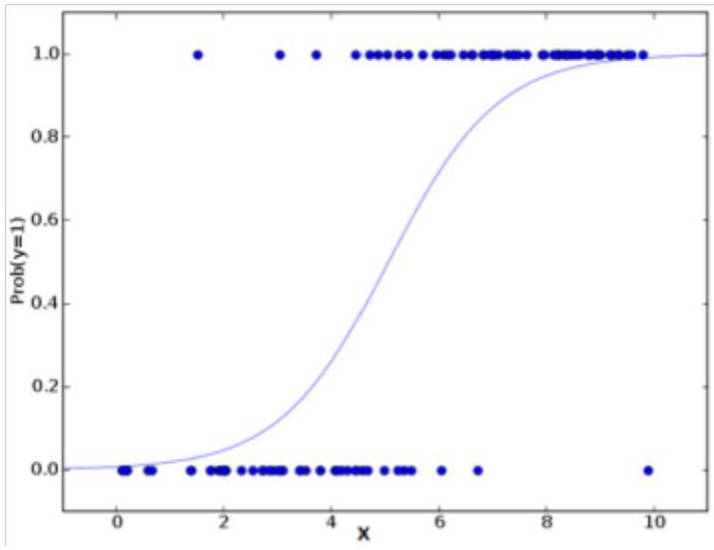
WHAT IS LOGISTIC REGRESSION ?

Logistic Regression is a classification algorithm. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables. To represent binary / categorical outcome, we use dummy variables. You can also think of logistic regression as a special case of linear regression when the outcome variable is categorical, where we are using log of odds as dependent variable. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function.

Equation and graphs

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta(\text{Age})$$

This is the equation used in Logistic Regression. Here (p/1-p) is the odd ratio. Whenever the log of odd ratio is found to be positive, the probability of success is always more than 50.



Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

1 / (1 + e^-value),Where e is the base of the natural logarithms (Euler's number or the EXP() function in your spreadsheet) and value is the actual numerical value that you want to transform.

PROPOSED ALGORITHM

Step1

import numpy as np

import pandas as pd

import seaborn as sb

import matplotlib.pyplot as plt

sklearn.linear_model import LogisticRegression

from sklearn.model_selection import

train_test_split

from sklearn.metrics import classification_report,accuracy_score

step2

loan=pd.read_csv("C:\\datasets\\train.csv")
loan.head(10)

after implementing all library function, we introduced all value in a perfect set that must be free from error after all these steps like visualization and data wrangling, we came to our final steps that is algorithm of logistic regression.

step3

from sklearn.linear_model import from
sklearn.model_selection import
train_test_split '''ts_score=[]
import numpy as np
for j in range(100):
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=j,test_size=0.1

5) lr=LogisticRegression().fit(x_train,y_train)

ts_score.append(lr.score(x_test,y_test)) j=ts_score.index(np.max(ts_score))'''

By applying this algorithm we get that at j=11 we are getting a accuracy and when this value is again put into the same algorithm we get a accuracy score of 73.9%

EXPERIMENTAL RESULT

Model without data preprocessing: 0.7395930562684261

Now if we do the data cleaning process i.e. if you eliminate the unwanted parameters from the data set which does not directly dependent on loan approval process we can get a more accurate model.

So , we have eliminated some unwanted parameters like ‘Loan_ID’,'Married','Dependents','Self_Employed','ApplicantIncome', 'Co_applicant_ Income','Loan_Amount_Term

We worked with the following parameters which are directly dependent parameters for loan approval

```
In [5]: new_loan_df.head()
Out[5]:
```

	Gender	Education	LoanAmount	Credit_History	Property_Area	Loan_Status
0	Male	Graduate	NaN	1.0	Urban	Y
1	Male	Graduate	128.0	1.0	Rural	N
2	Male	Graduate	66.0	1.0	Urban	Y
3	Male	Not Graduate	120.0	1.0	Urban	Y
4	Male	Graduate	141.0	1.0	Urban	Y

We have followed the same process and applied the same algorithm with the above data set and got a more accurate model with accuracy 0.85204 95597

So, model with data preprocessing:
0.852049559743873

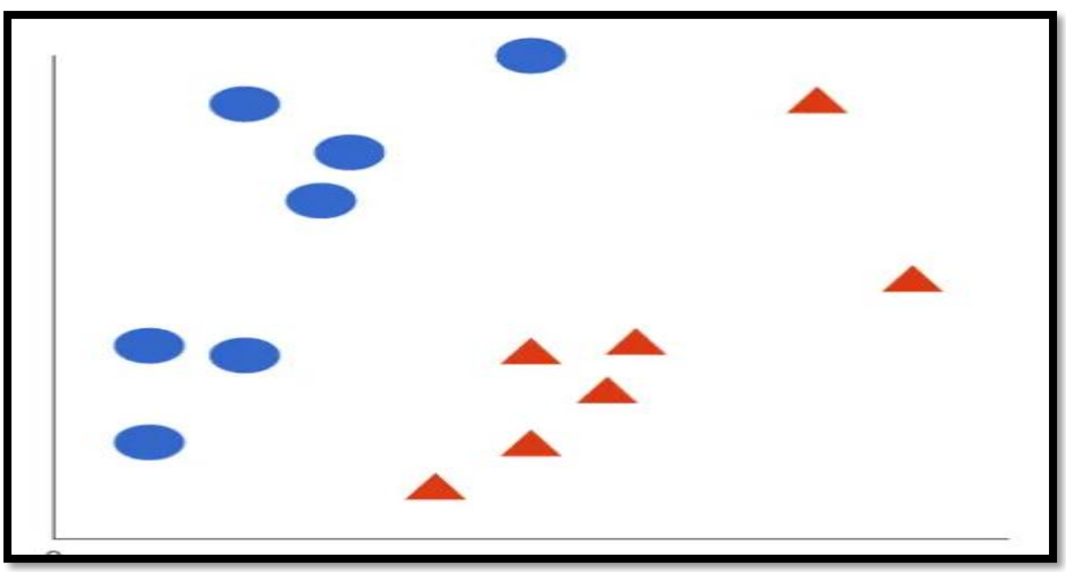
Model improvement of preprocessing: 12.75271774731987%

Support Vector Machine

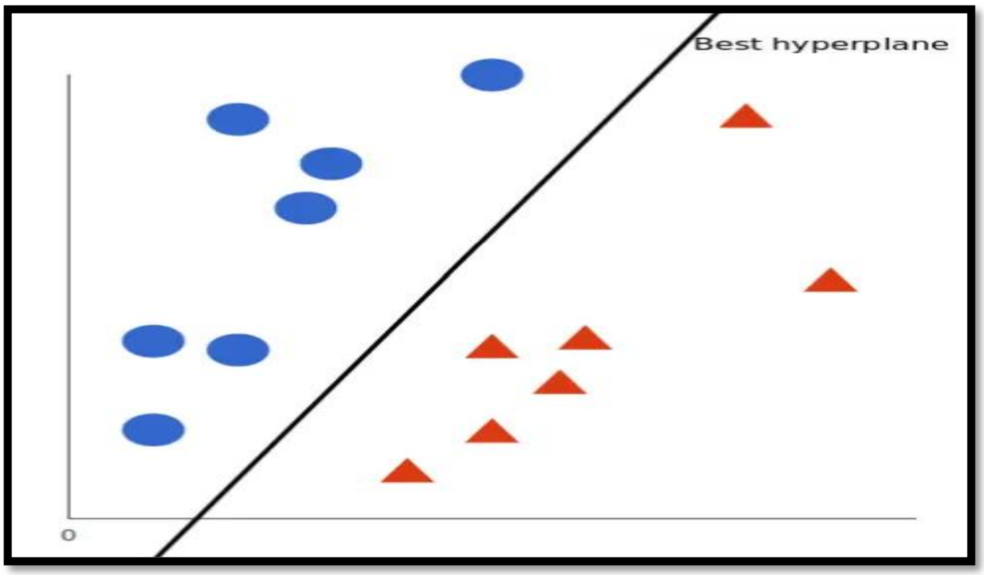
- a fast and dependable classification algorithm that performs very well with a limited amount of data
- A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyper plane
- given labeled training data (supervised learning), the algorithm outputs an optimal hyper plane which categorizes new examples.
- In two dimensional space this hyper plane is a line dividing a plane in two parts where in each class lay in either side.

How does SVM work?

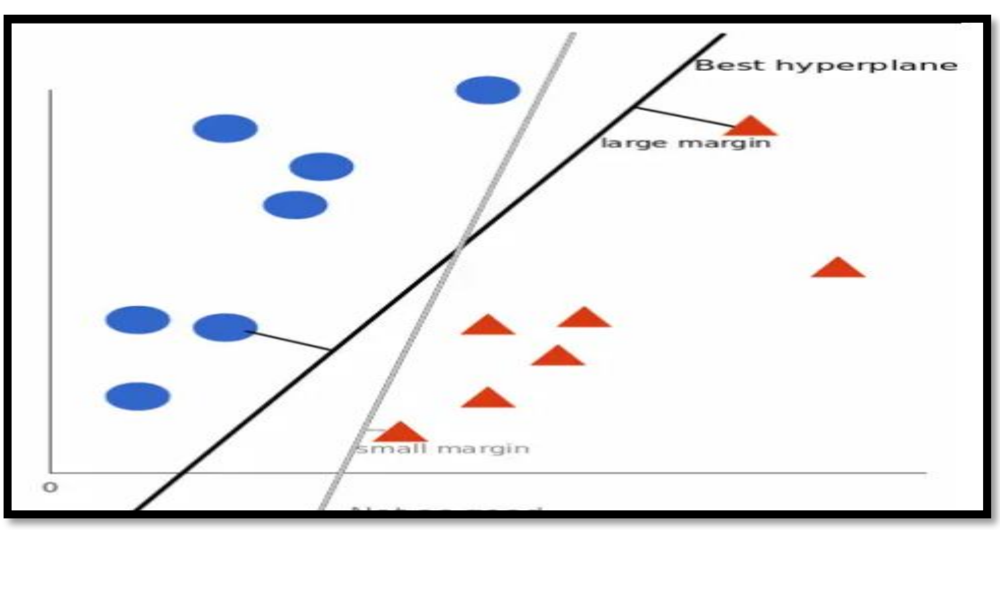
- Let's imagine we have two tags: *red* and *blue*, and our data has two features: *x* and *y*. We want a classifier that, given a pair of *(x,y)* coordinates, outputs if it's either *red* or *blue*.



- A support vector machine takes these data points and outputs the hyper plane (which in two dimensions it's simply a line) that best separates the tags. This line is the **decision boundary**: anything that falls to one side of it we will classify as *blue*, and anything that falls to the other as *red*.



- But, what exactly is *the best* hyper plane? For SVM, it's the one that maximizes the margins from both tags. In other words: the hyper plane (remember it's a line in this case) whose distance to the nearest element of each tag is the largest.



PROPOSED ALGORITHM

Step1

import numpy as np

import pandas as pd

import seaborn as sb

import matplotlib.pyplot as plt

from sklearn.linear_model import svm from

sklearn.model_selection import

train_test_split

from sklearn.metrics import classification_report,accuracy_score

step2

loan=pd.read_csv("C:\\datasets\\train.csv")
loan.head(10)

after implementing all library function, we introduced all value in a perfect set that must be free from error after all these steps like visualization and data wrangling, we came to our final steps that is algorithm of svm.

step3

from sklearn import svm

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.3,random_state=48)

clf=svm.SVC(kernel='linear')
clf.fit(x_train,y_train)
y_pred=clf.predict(x_test)
accuracy_score(y_test,y_pred)

By applying this algorithm we get that at j=27 we are getting a accuracy and when this value is again put into the same algorithm we get a accuracy score of 78.37%

EXPERIMENTAL RESULT

Model without data preprocessing: 0.7895930562684261

Now if we do the data cleaning process i.e. if you eliminate the unwanted parameters from the data set which does not directly dependent on loan approval process we can get a more accurate model.

So , we have eliminated some unwanted parameters like ‘Loan_ID’,'Married','Dependents','Self_Employed','ApplicantIncome', 'Co_applicant_ Income','Loan_Amount_Term

We worked with the following parameters which are directly dependent parameters for loan approval

```
In [5]: new_loan_df.head()
Out[5]:
```

	Gender	Education	LoanAmount	Credit_History	Property_Area	Loan_Status
0	Male	Graduate	NaN	1.0	Urban	Y
1	Male	Graduate	128.0	1.0	Rural	N
2	Male	Graduate	66.0	1.0	Urban	Y
3	Male	Not Graduate	120.0	1.0	Urban	Y
4	Male	Graduate	141.0	1.0	Urban	Y

We have followed the same process and applied the same algorithm with the above data set and got a more accurate model with accuracy 0.73520 495597

So, model with data preprocessing: 0.732049559743873

Model improvement of preprocessing: 5.75271774731987%

K Nearest Neighbours

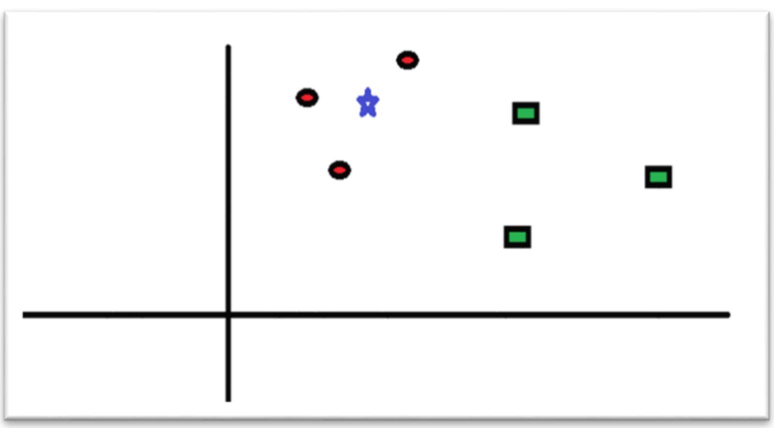
- KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry

- KNN algorithms use a data and classify new data points based on a similarity measures (e.g. distance function).

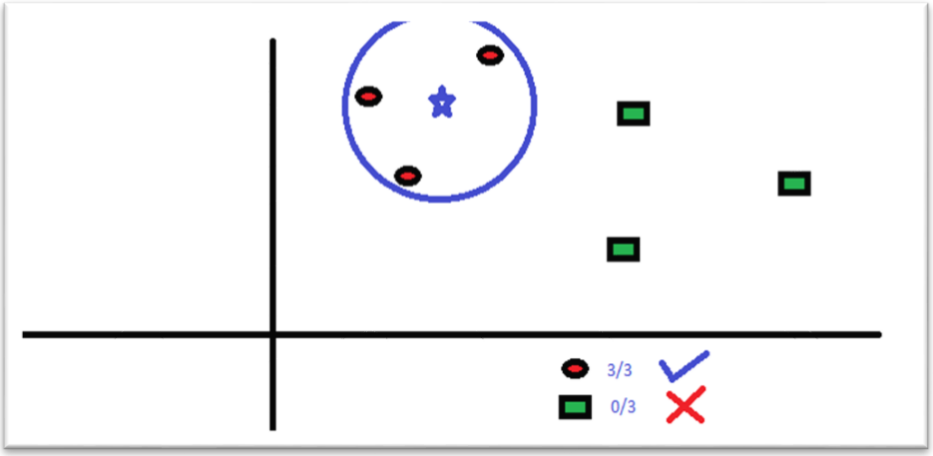
- Classification is done by a majority vote to its neighbours.
  The data is assigned to the class which has the most nearest neighbor

HOW DOES IT WORK?

Following is a spread of red circles (RC) and green squares (GS)



We intend to find out the class of the blue star (BS) . BS can either be RC or GS and nothing else. The “K” is KNN algorithm is the nearest neighbors we wish to take vote from. Let’s say K = 3. Hence, we will now make a circle with BS as center just as big as to enclose only three data points on the plane. Refer to following diagram for more details:



The three closest points to BS is all RC. Hence, with good confidence level we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbour went to RC. The choice of the parameter K is very crucial in this algorithm.

Choosing the right value for K

To select the K that's right for your data, we run the KNN algorithm several times with different values of K and choose the K that reduces the number of errors we encounter while maintaining the algorithm's ability to accurately make predictions when it's given data it hasn't seen before.
As we decrease the value of K to 1, our predictions become less stable

$$\boxed{\begin{aligned} \mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) &= \sqrt{(q_1-p_1)^2+(q_2-p_2)^2+\cdots+(q_n-p_n)^2} \\ &= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}. \end{aligned}}$$

Where q1 to qn represent the attribute values for one observation and p1 to pn represent the attribute values for the other observation.

PROPOSED ALGORITHM

Step1

import numpy as np

import pandas as pd

import seaborn as sb

import matplotlib.pyplot as plt

from sklearn.linear_model import KNeighborsClassifier
from sklearn.model_selection import

train_test_split

from sklearn.metrics import classification_report,accuracy_score

step2

loan=pd.read_csv("C:\\datasets\\train.csv")
loan.head(10)

after implementing all library function, we introduced all value in a perfect set that must be free from error after all these steps like visualization and data wrangling, we came to our final steps that is algorithm of svm.

step3

from sklearn import svm

```
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=j,test_size=.03)
lr=KNeighborsClassifier()
 lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
ts_score.append(accuracy_score(y_test,y_pred))
j=ts_score.index(np.max(ts_score))
y_pred=classifier.predict(x_test)
accuracy_score(y_test,y_pred)
```

By applying this algorithm we get that at j=27 we are getting a accuracy and when this value is again put into the same algorithm we get a accuracy score of 83.37%

EXPERIMENTAL RESULT

Model without data preprocessing: 0.8395930562684261

Now if we do the data cleaning process i.e. if you eliminate the unwanted parameters from the data set which does not directly dependent on loan approval process we can get a more accurate model.

So , we have eliminated some unwanted parameters like 'Loan_ID','Married','Dependents','Self_Employed','ApplicantIncome', 'Co_applicant_ Income','Loan_Amount_Term

We worked with the following parameters which are directly dependent parameters for loan approval

```
In [5]: new_loan_df.head()
Out[5]:
```

	Gender	Education	LoanAmount	Credit_History	Property_Area	Loan_Status
0	Male	Graduate	NaN	1.0	Urban	Y
1	Male	Graduate	128.0	1.0	Rural	N
2	Male	Graduate	66.0	1.0	Urban	Y
3	Male	Not Graduate	120.0	1.0	Urban	Y
4	Male	Graduate	141.0	1.0	Urban	Y

We have followed the same process and applied the same algorithm with the above data set and got a more accurate model with accuracy 0.61204 95597

So, model with data preprocessing: 0.612049559743873

COMPARATIVE STUDY

Unprocessed data	Processed data
Unprocessed data are those data which are not cleaned.	Processed data are those data which are cleaned
No categorical variables are removed	Categorical variables are removed
No missing words are filled	Imputer helps in filling those missing words
Outliers are not removed	Outliers are removed by Tukey method
Cross validation is done to divide the datasets	Cross validation is done to divide the datasets
No feature selection	Feature selection is done to select relevant features and to shorten the dimensionality of datasets
Prediction of outcomes is done by logistic regression	Prediction of outcomes is done by logistic regression
AUC of model with data without Preprocessing in logistic regression: **0.732049559743873**	**AUC of model with data Preprocessing in logistic regression:** **0.859593056268426l**

Chapter- 8:

User interface

A **user interface**, also called a "UI" or simply an "**interface**," is the means in which a person controls a software application or hardware device. ... Nearly all software programs have a graphical **user interface**, or GUI. This means the program includes graphical controls, which the **user** can select using a mouse or keyboard

user interface, also sometimes called a human-computer interface, comprises both hardware and software components. It handles the interaction between the user and the system.

There are different ways of interacting with computer systems which have evolved over the years. There are five main types of user interface:

- command line (cli)
- graphical user interface (GUI)
- menu driven (mdi)
- form based (fbi)
- natural language (nli)

Here we have used GUI as the interface to implement our project so that a user can give some basic information and get to know if he / she can get the loan or not.

GUI (Graphical User Interface).

Stands for "**Graphical User Interface**" and is pronounced "gooey." It is a **user interface** that includes **graphical** elements, such as windows, icons and buttons. The term was created in the 1970s to distinguish **graphical interfaces** from text-based ones, such as command line **interfaces**.

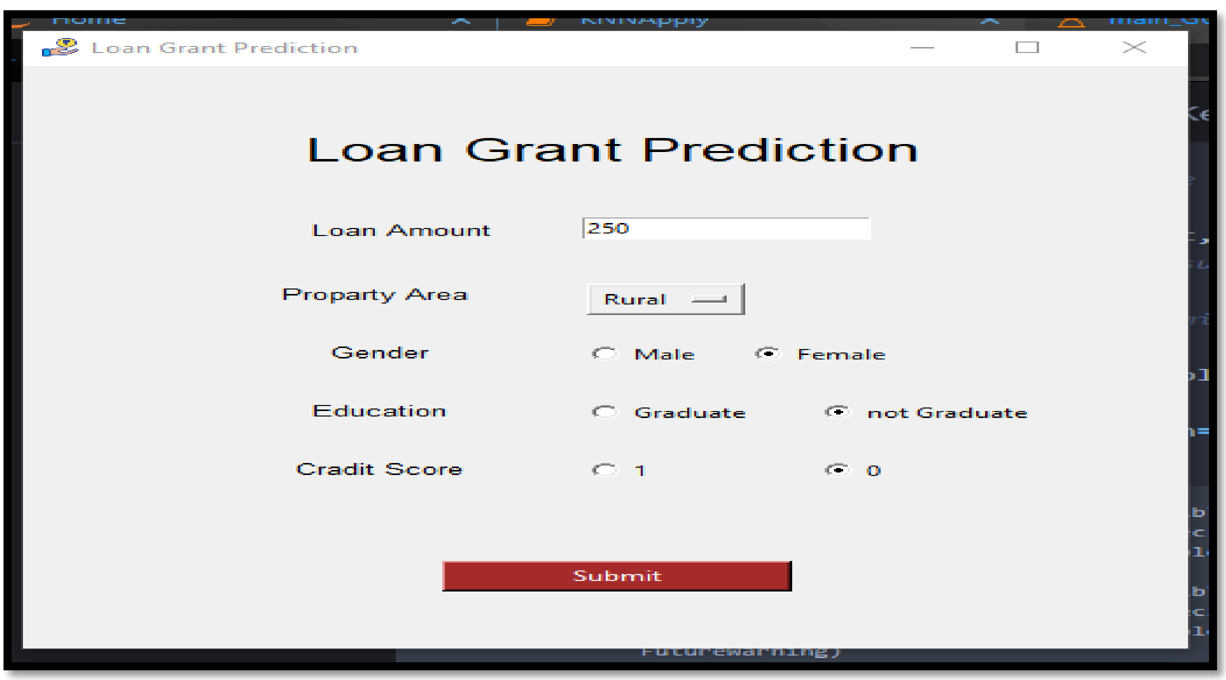
Some popular, modern **graphical user interface examples** include Microsoft Windows, macOS, Ubuntu Unity, and GNOME Shell for desktop environments, and Android, Apple's iOS, BlackBerry OS, Windows 10 Mobile, Palm OS-WebOS, and Firefox OS for smartphones.

Equipment

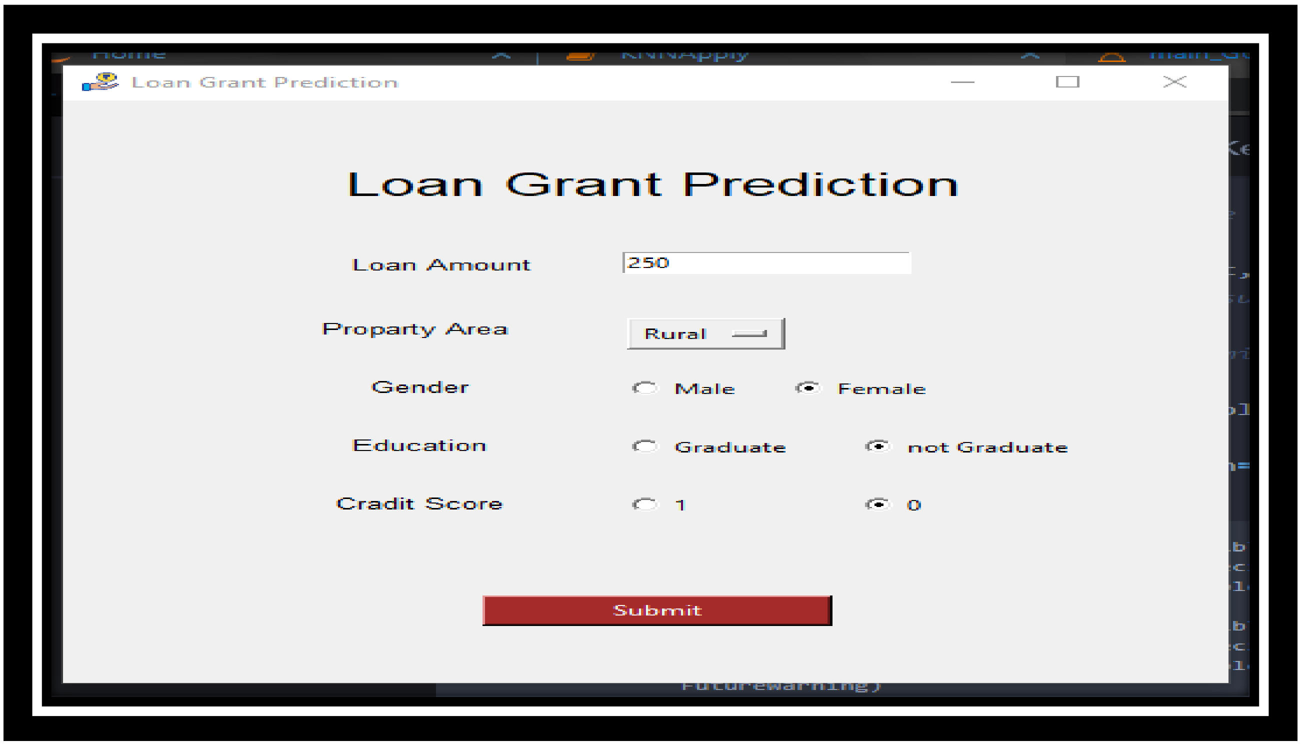
- **Tkinter package of python**: **Tkinter** is **Python's** de-facto standard GUI (Graphical User Interface) package. It is a thin object-oriented layer on top of Tcl/Tk. **Tkinter** is not the only GuiProgramming toolkit for **Python**. It is however the most commonly used one.

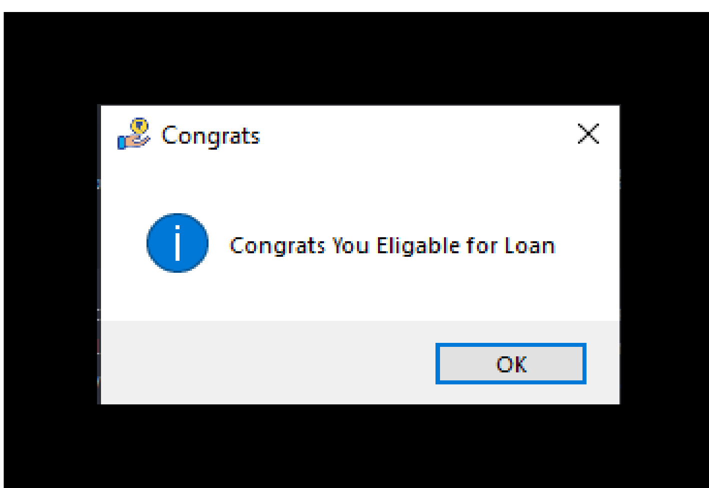
- **Jupyter Notebook**: The **Jupyter Notebook** is an open source web **application** that you can **use** to create and share documents that contain live code, equations, visualizations, and text. **Jupyter Notebook** is maintained by the people at Project **Jupyter**.

LOAN PREDICTION USER INTERFACE



These are the basic information which the user have to submit. The backend code will execute the algorithm which predict the most accurate probability of getting loan. There will be one pop up as soon as the user will press the submit button which will say if the user can get the loan or not.





Chapter- 9:

Research Work

Many factors affect the success of Machine Learning algorithm in a given project. The representation and quality of the dataset used for training the model is most important. If there is irrelevant and much redundant information present or noisy and unreliable data, then the predictive accuracy and knowledge discovery from the dataset become less accurate.

This study aims for an empirical assessment of the effectiveness of data preprocessing on the predictive accuracy of the algorithm. By selecting relevant instances, we have removed irrelevant as well as noisy or redundant data. So that high quality data will lead to high quality result. In order to reduce the prediction errors and improve efficiency, we have carefully selected those data field necessary according to the characteristics of Machine learning algorithm.

This paper presents the illustrative study of the Loan grant prediction by applying various algorithm to predict accuracy with different datasets i.e. original dataset from Kaggle and the refined dataset with selective data. By applying Logistics Regression, we found out that predictive accuracy of the refined dataset is more than the unrefined data.
Applying Logistics Regression on refined dataset with selective data predictive accuracy comes 85.78%.
Applying Logistics regression on original dataset, predictive accuracy comes at 73.21%.
By above result, we can clearly see that predictive accuracy of the refined dataset is more.

Chapter- 10:

CONCLUSION

Data pre-processing is very important in data mining process. Certain data cleaning techniques usually are not applicable to all kinds of data. Deduplication and data linkage are important tasks in the pre- processing step for many data mining projects. It is important to improve data quality before data is loaded into data warehouse. Locating approximate duplicates in large databases is an important part of data management and plays a critical role in the data cleaning process. In this research wok, a framework is designed to clean duplicate data for improving data quality and also to support any subject oriented data. Only few cleaning methods are implemented in the existing data cleaning techniques. However, those existing techniques are good in some part of cleaning process. For example duplicate elimination cleaning tools are suited for data elimination process and similarity cleaning tools is well suited for field similarity and record similarity.

With the help of data preprocessing or data wrangling ,we tried to enhance the performance of machine learning algorithm . We removed the categorical variables because machine learning do not understand alphabetical values .. Interaction variables increases the dimensionality of datasets.

Feature selection is used to select the best feature and to reduce the dimensionality of datasets . Cross Validation is a technique which involvesreserving a particular sample of a dataset on which you do not train the model. Later, you test your model on this sample before finalizing it. The logistics regression is used to predict the dependent variables and rock_auc curve is used to check the quality of model performance..

REFERENCES

1.)https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-onlogistic-regression-in-r/

2.)https://www.analyticsvidhya.com/blog/2015/11/improve-modelperformance-cross-validation-in-python-r/

3.)https://en.m.wikipedia.org/wiki/KNN

4.)https://en.m.wikipedia.org/wiki/Feature_selection

5.) https://www.kaggle.com/altruistdelhite04/loan-prediction-problem-dataset

6.)https://www.analyticsvidhya.com/blog/2015/11/easy-methods-dealcategorical-variables-predictive-modeling/

7.)Multivariate Data analysis book by Joseph, W/7th edition