

CREDIT CARD APPROVAL USING MACHINE LEARNING

INTRODUCTION:

1.1 OVERVIEW

Credit risk as the board in banks basically centers around deciding the probability of a customer's default or credit decay and how expensive it will end up being assuming it happens. It is important to consider major factors and predict beforehand the probability of consumers defaulting given their conditions. Which is where a machine learning model comes in handy and allows the banks and major financial institutions to predict whether the customer, they are giving the loan to, will default or not. This project builds a machine learning model with the best accuracy possible using python. First we load and view the dataset. The dataset has a combination of both mathematical and non-mathematical elements, that it contains values from various reaches, in addition to that it contains a few missing passages. We preprocess the dataset to guarantee the AI model we pick can make great expectations. After the information is looking great, some exploratory information examination is done to assemble our instincts. Finally, we will build a machine learning model that can predict if an individual's application for a credit card will be accepted. Using various tools and techniques we then try to improve the accuracy of the model. This project uses Jupyter notebook for python programming to build the machine learning model. Using Data Analysis and Machine Learning, we attempted to determine the most essential parameters for obtaining credit card acceptance in this project.

1.2 PURPOSE

It uses personal information and data submitted by credit card applicants to predict the probability of future defaults and credit card borrowings. The bank is able to decide whether to issue a credit card to the applicant. Credit scores can objectively quantify the magnitude of risk.

2 LITERATURE SURVEY

2.1 Existing problem :

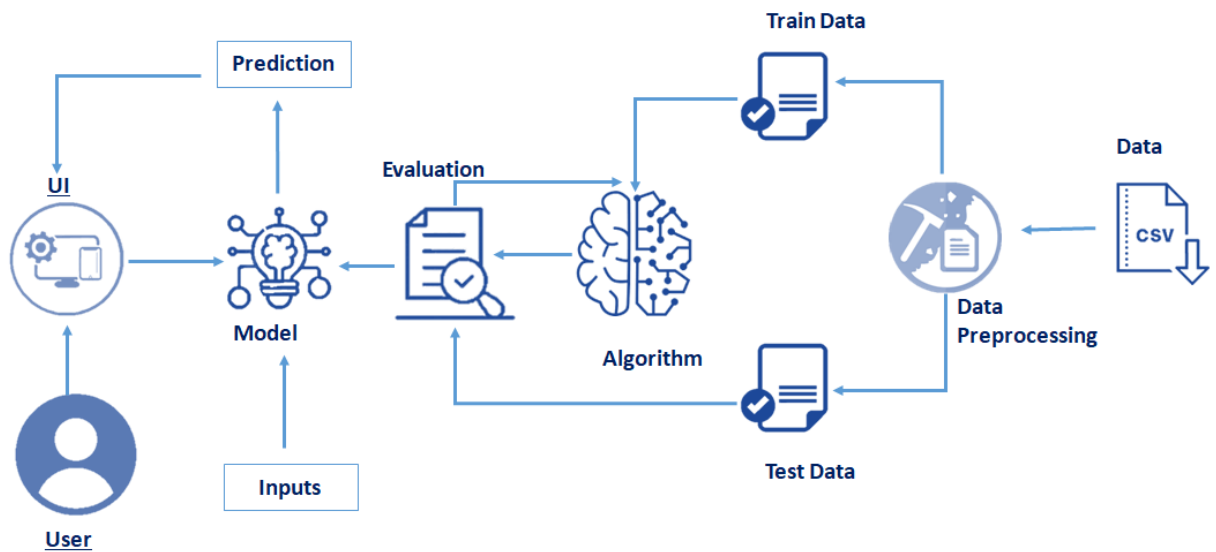
To apply machine learning model that can predict if an individual's application for a credit card will be accepted or not.

2.2 Proposed solution :

A good machine learning model should be able to accurately predict the status of the applications with respect to these statistics. Predicting if a credit card application will be approved or not is a classification task

3 THEORITICAL ANALYSIS

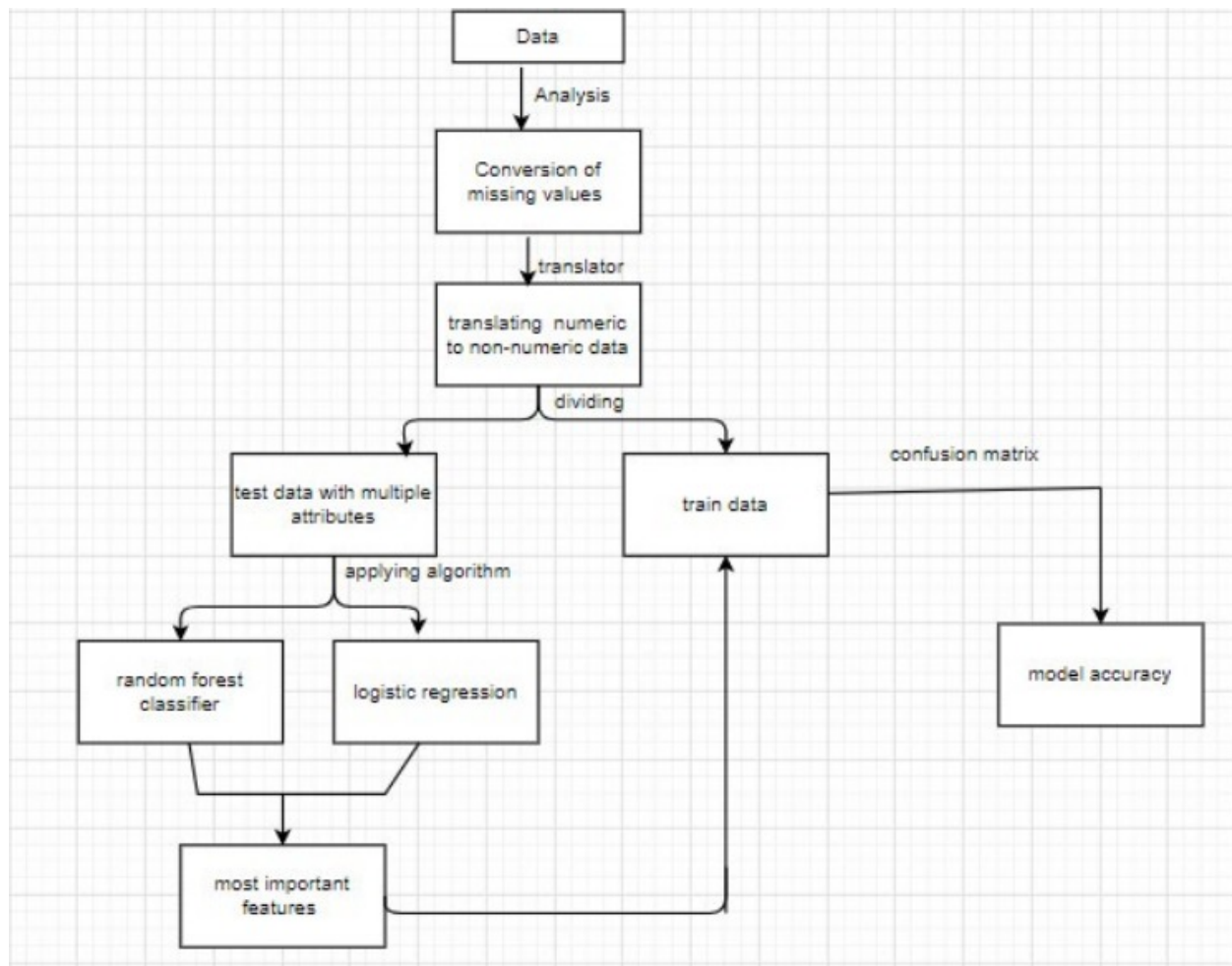
3.1 BLOCK DIAGRAM :



SOFTWARE REQUIREMENTS :

- ANACONDA NAVIGATOR
- JUPYTER
- SPYDER

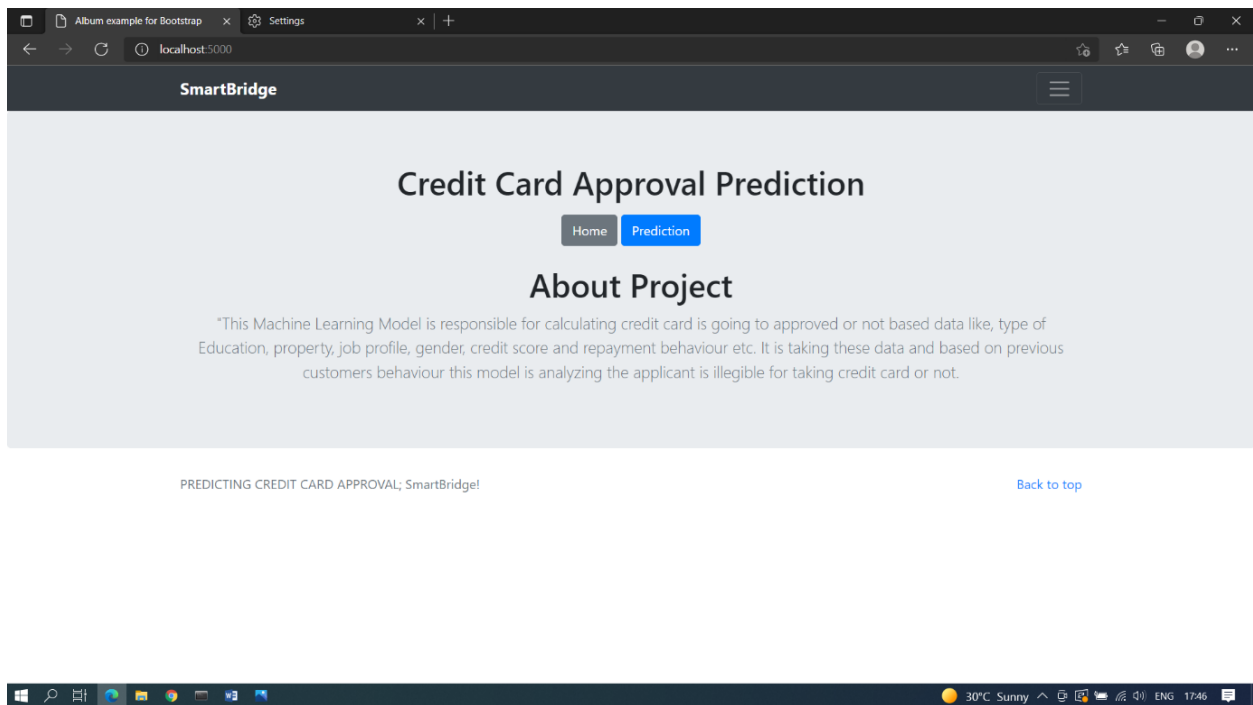
FLOW CHART:



4. PROJECT FLOW :

- DATA COLLECTION
- DATA VISUALIZATION
- DATA PRE-PROCESSING
- MODEL BUILDING
- APPLICATION BUILDING

5 RESULT:



Credit Card Approval Prediction

GENDER	OWN CAR OR NOT	
<input type="text" value="FEMALE"/>	<input type="text" value="NO"/>	
OWN REALSTATE	TOTAL ANNUAL INCOME	TYPE OF INCOME
<input type="text" value="NO"/>	<input type="text" value="454562"/>	<input type="text" value="Student"/>
EDUCATION	FAMILY STATUS	TYPE OF HOUSING
<input type="text" value="Higher education"/>	<input type="text" value="Married"/>	<input type="text" value="House / apartment"/>
DAYS BIRTH	DAYS EMPLOYED	FAMILY MEMBERS
<input type="text" value="16"/>	<input type="text" value="19"/>	<input type="text" value="6"/>
EMI PAID OFF	EMI OF PASTDUES	NUMBER OF LOANS
<input type="text" value="6"/>	<input type="text" value="4"/>	<input type="text" value="2"/>

Predict

You are "Not Eligible" for credit card

6 ADVANTAGES AND DISADVANTAGES

6.1 ADVANTAGES:

- Opportunity to build credit
- Earn rewards such as cash back or miles points
- Protection against credit card fraud
- Free credit score information
- No foreign transaction fees

6.2 DISADVANTAGES:

- Minimum due trap
- Hidden costs
- Ease of overuse
- High interest rate

7 CONCLUSION:

Currently, factors considered are regular details related to gender, age of the consumer, his/her credit reports and worthiness, yearly income, and the number of years he/she has been working. Further, to improve this work, various other factors or conditions can be considered like their history related to any offense and their assets which can be both physical and liquid cash. These features can improve the model to be more effective and can help the institutes to make better decisions so that they can avoid experiencing frauds and loss. Various classification algorithms can be used to build a

model and compare the rates or levels of accuracy to improve the model for better use.

8 FUTURE SCOPE :

From this initial analysis, we are able to conclude that the most significant factors in determining the outcome of a credit application are Employment, Income, Credit Score and Prior Default.

Based on these insights, we can work on building some predictive models. They can be used by analysts in financial sector and be incorporated to automate the credit approval process. These results can also serve as a source of information for the consumers.

9 APPENDIX:

Predicting Good or Bad From Credit Card Approvals

Application data contains information from clients when applying for credit card. The credit card data contains information on the length of time the credit card account was opened since the initial approval as well as the status of the loan during each month. The purpose of this machine learning task is to predict whether or not a client is good or bad based on the application/ credit card data. The target variable is not given so we will need to find a method to determine good or bad.

```
In [6]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
# from sklearn.metrics import SHOTETonek
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import classification_report, confusion_matrix, f1_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
```

```
In [7]: app = pd.read_csv("C:\\Users\\gatti_mhg711f\\Desktop\\Loan Approval Prediction\\Data\\application_record.csv")
credit = pd.read_csv("C:\\Users\\gatti_mhg711f\\Desktop\\Loan Approval Prediction\\Data\\credit_record.csv")
```

```
In [8]: app.head()
```

Out[8]:

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	N
0	5008804	M	Y	Y	0	427500.0	Working	Higher education	
1	5008805	M	Y	Y	0	427500.0	Working	Higher education	
2	5008806	M	Y	Y	0	112500.0	Working	Secondary / secondary special	
3	5008808	F	N	Y	0	270000.0	Commercial associate	Secondary / secondary special	
4	5008809	F	N	Y	0	270000.0	Commercial associate	Secondary / secondary special	

Desktop/Loan Approval Prediction x Credit Card Approval Prediction x

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
In [9]: credit.head()
```

```
Out[9]:
```

	ID	MONTHS_BALANCE	STATUS
0	50017111	0	X
1	50017111	-1	0
2	50017111	-2	0
3	50017111	-3	0
4	5001712	0	C

Data Visualization

```
In [10]: print("Number of people working status :")
print(app['OCCUPATION_TYPE'].value_counts())
sns.set(rc = {'figure.figsize':(18,6)})
sns.countplot(x='OCCUPATION_TYPE', data=app, palette = 'Set2')
```

Number of people working status :

Occupation Type	Count
Laborers	78240
Core staff	43007
Sales staff	41098
Managers	35487
Drivers	26990
High skill tech staff	17289
Accountants	15985
Medicine staff	13520
Cooking staff	8076
Security staff	7993
Cleaning staff	5845
Private service staff	3456
Low-skill Laborers	2140
Secretaries	2044
Waiters/barmen staff	1665
Realty agents	1041
HR staff	774
IT staff	604

```
Out[10]: <AxesSubplot:xlabel='OCCUPATION_TYPE', ylabel='count'>
```

31°C Mostly sunny 09:30 13-04-2022

Desktop/Loan Approval Prediction x Credit Card Approval Prediction x

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
Out[10]: <AxesSubplot:xlabel='OCCUPATION_TYPE', ylabel='count'>
```

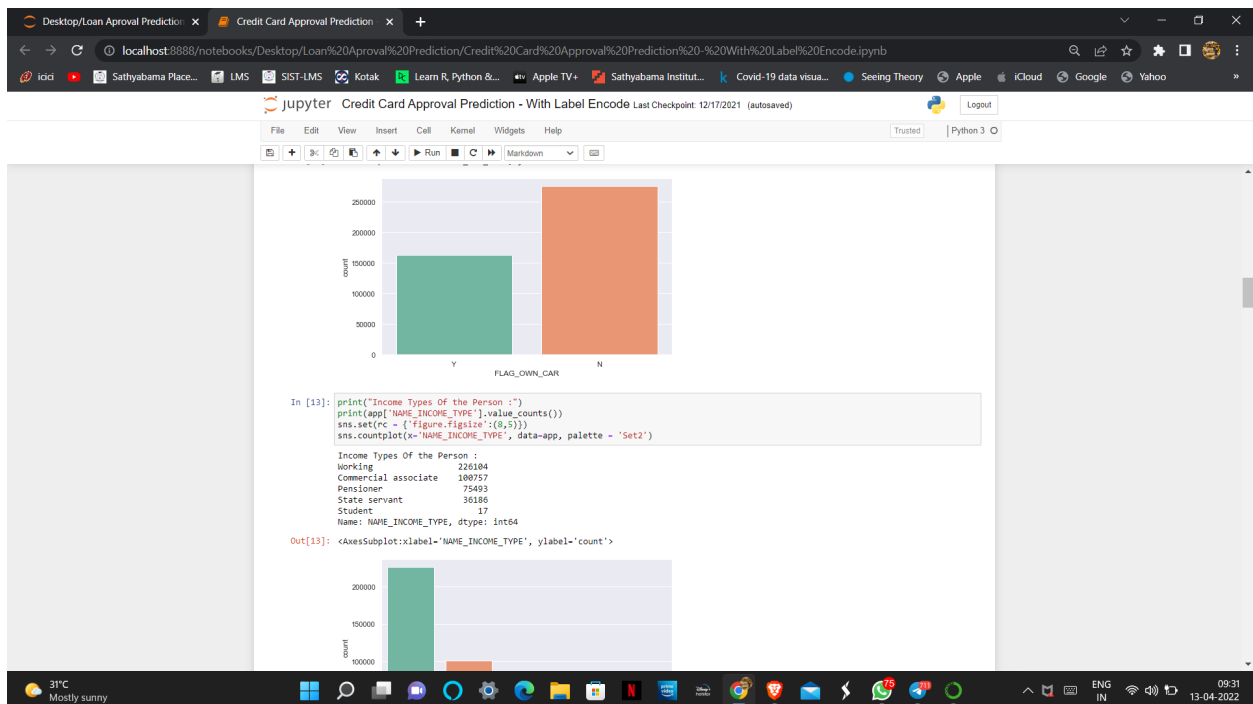
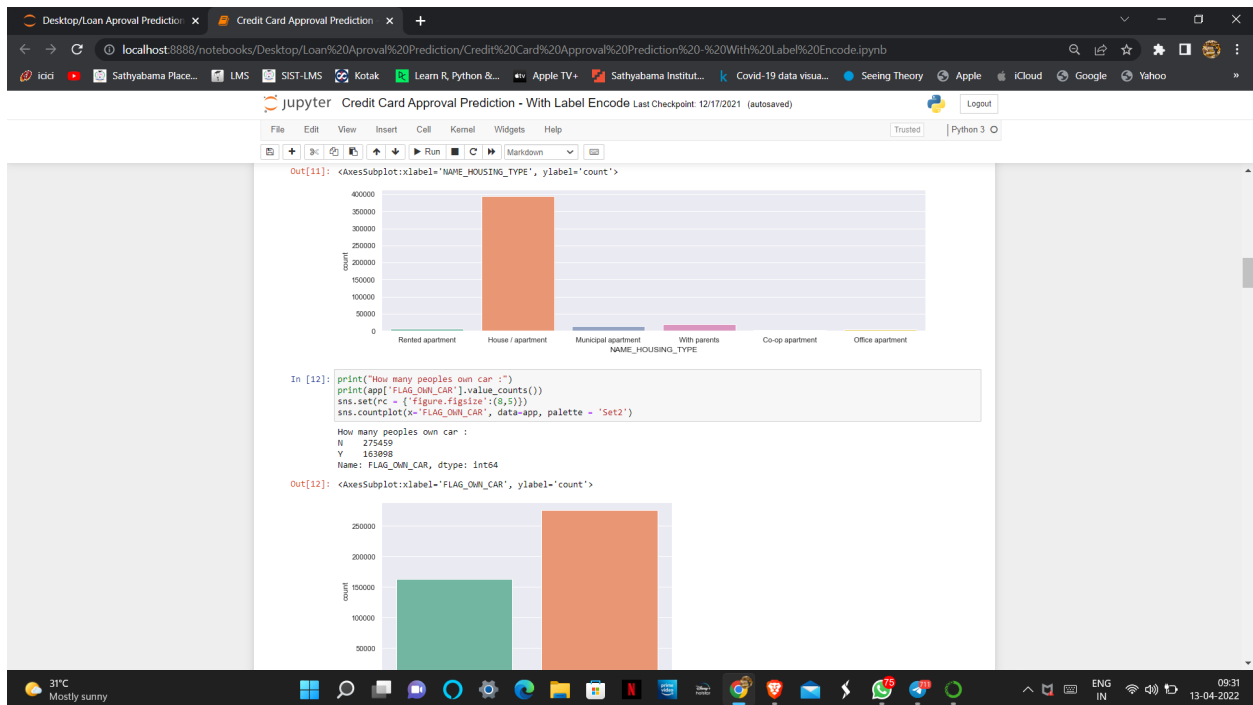
```
In [11]: print("Types of house of the peoples :")
print(app['NAME_HOUSING_TYPE'].value_counts())
sns.set(rc = {'figure.figsize':(15,4)})
sns.countplot(x='NAME_HOUSING_TYPE', data=app, palette = 'Set2')
```

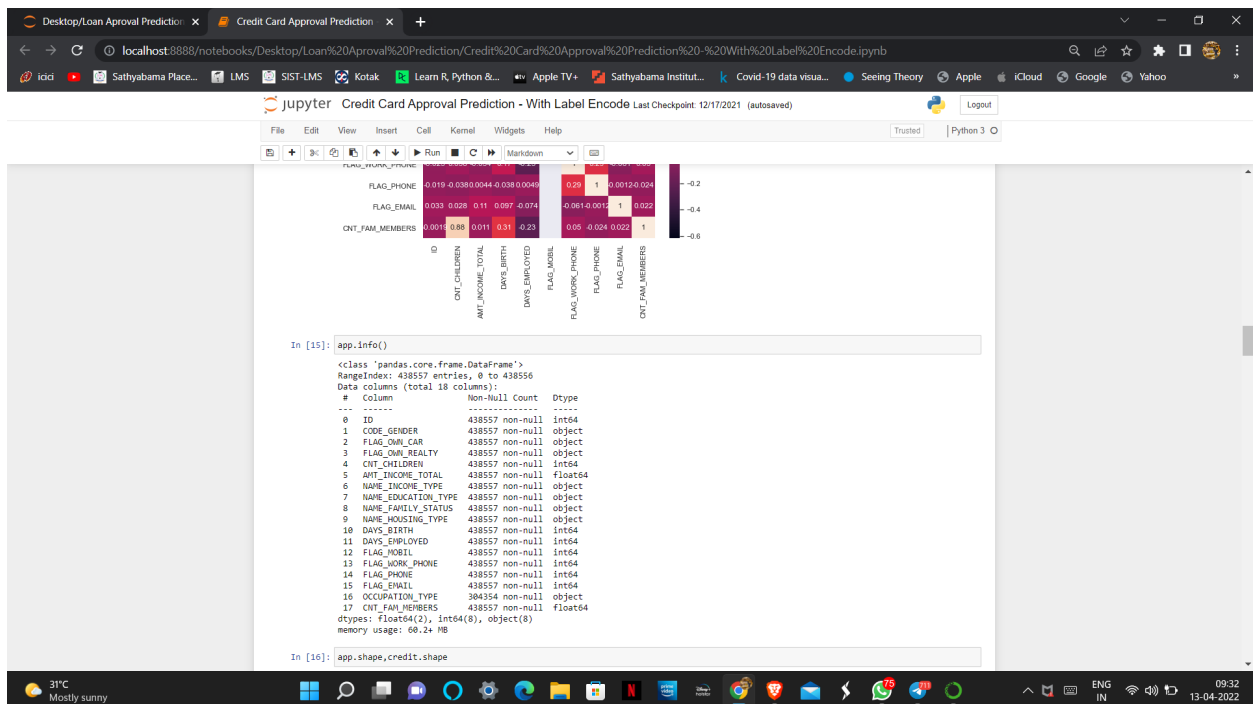
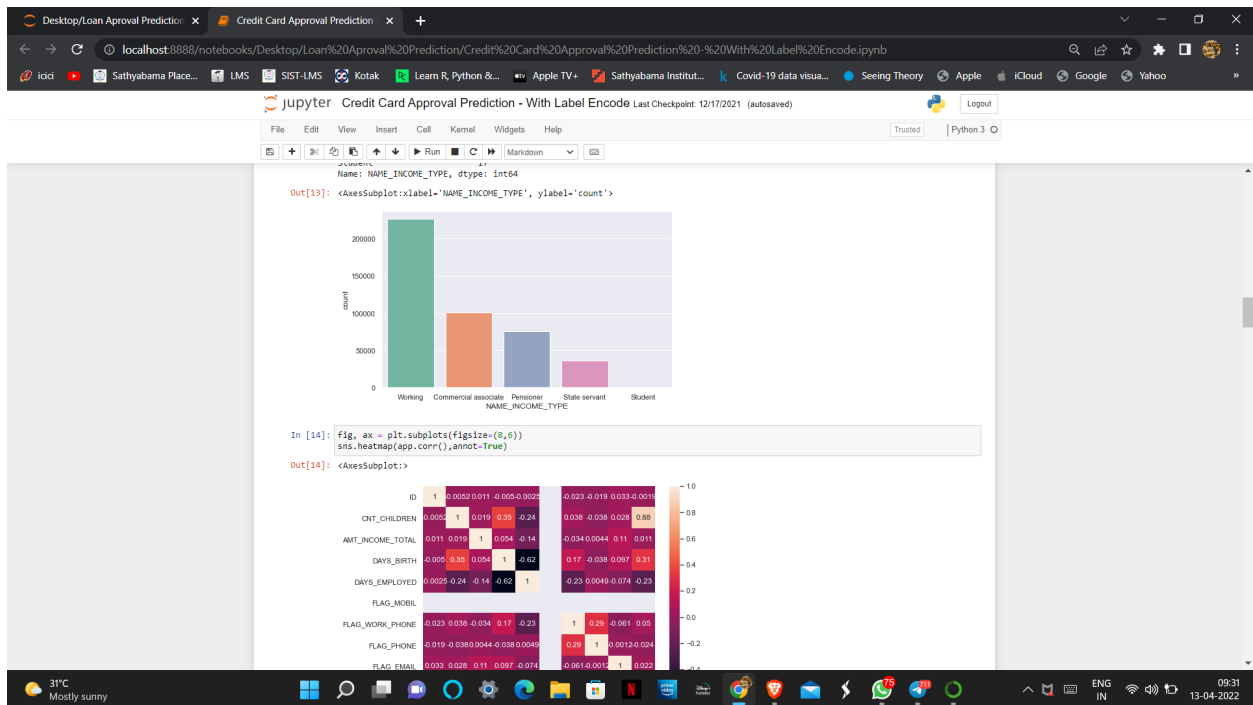
Types of house of the peoples :

House Type	Count
House / apartment	393831
With parents	19077
Municipal apartment	14214
Rented apartment	5974
Office apartment	3922
Co-op apartment	1539

```
Out[11]: <AxesSubplot:xlabel='NAME_HOUSING_TYPE', ylabel='count'>
```

31°C Mostly sunny 09:31 13-04-2022





Desktop/Loan Approval Prediction x Credit Card Approval Prediction x +

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 O

In [16]: app.shape, credit.shape

Out[16]: ((438557, 18), (1848575, 3))

In [17]: app.describe()

Out[17]:

	ID	CNT_CHILDREN	AMT_INCOME_TOTAL	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_EM
count	4.385570e+05	438557.000000	4.385570e+05	438557.000000	438557.000000	438557.0	438557.000000	438557.000000	438557.000000
mean	6.022176e-06	0.427390	1.875243e+05	-15997.904649	60563.875328	1.0	0.206133	0.287771	0.1082
std	5.716370e-05	0.724882	1.100899e+05	4185.030007	138767.799647	0.0	0.404527	0.452724	0.3106
min	5.008046e-06	0.000000	2.610000e+04	-25201.000000	-17531.000000	1.0	0.000000	0.000000	0.0000
25%	5.609375e-06	0.000000	1.215000e+05	-19483.000000	-3103.000000	1.0	0.000000	0.000000	0.0000
50%	6.047745e-06	0.000000	1.607805e+05	-15830.000000	-1487.000000	1.0	0.000000	0.000000	0.0000
75%	6.456871e-06	1.000000	2.250000e+05	-12514.000000	-371.000000	1.0	0.000000	1.000000	0.0000
max	7.99952e-06	19.000000	6.750000e+05	-7489.000000	365243.000000	1.0	1.000000	1.000000	1.0000

In [18]: app.isnull().sum()

Out[18]:

ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS	NAME_HOUSING_TYPE	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_EMAIL	OCCUPATION_TYPE	CNT_FAM_MEMBERS	dtype
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	int64

31°C Mostly sunny 09:32 13-04-2022

Desktop/Loan Approval Prediction x Credit Card Approval Prediction x +

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 O

In [19]:

```
def unique_values():
    a = app.CODE_GENDER.unique()
    print("-----CODE_GENDER-----")
    print(a)
    print()
    b = app.FLAG_OWN_CAR.unique()
    print("-----FLAG_OWN_CAR-----")
    print(b)
    print()
    c = app.FLAG_OWN_REALTY.unique()
    print("-----FLAG_OWN_REALTY-----")
    print(c)
    print()
    d = app.CNT_CHILDREN.unique()
    print("-----CNT_CHILDREN-----")
    print(d)
    print()
    e = app.NAME_INCOME_TYPE.unique()
    print("-----NAME_INCOME_TYPE-----")
    print(e)
    print()
    f = app.NAME_EDUCATION_TYPE.unique()
    print("-----NAME_EDUCATION_TYPE-----")
    print(f)
    print()
    g = app.NAME_FAMILY_STATUS.unique()
    print("-----NAME_FAMILY_STATUS-----")
    print(g)
    print()
    h = app.NAME_HOUSING_TYPE.unique()
    print("-----NAME_HOUSING_TYPE-----")
    print(h)
    print()
    i = app.OCCUPATION_TYPE.unique()
    print("-----OCCUPATION_TYPE-----")
    print(i)
    print()
    j = app.CNT_FAM_MEMBERS.value_counts()
    print("-----CNT_FAM_MEMBERS-----")
    print(j)
    print()
    return unique_values
```

In [20]: unique_values()

31°C Mostly sunny 09:32 13-04-2022

Desktop/Loan Approval Prediction: x Credit Card Approval Prediction x +

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 O

```
In [20]: unique_values()

-----CODE_GENDER-----
['M' 'F']

-----FLAG_OWN_CAR-----
['Y' 'N']

-----FLAG_OWN_REALTY-----
['Y' 'N']

-----CNT_CHILDREN-----
[ 0  1  2  3  4  5 14 19  7  9 12  6]

-----NAME_INCOME_TYPE-----
['Working' 'Commercial associate' 'Pensioner' 'State servant' 'Student']

-----NAME_EDUCATION_TYPE-----
['Higher education' 'Secondary / secondary special' 'Incomplete higher'
 'Lower secondary' 'Academic degree']

-----NAME_FAMILY_STATUS-----
['Civil marriage' 'Married' 'Single / not married' 'Separated' 'Widow']

-----NAME_HOUSING_TYPE-----
['Rented apartment' 'House / apartment' 'Municipal apartment'
 'With parents' 'Co-op apartment' 'Office apartment']

-----OCCUPATION_TYPE-----
[nan 'Security staff' 'Sales staff' 'Accountants' 'Laborers' 'Managers'
 'Drivers' 'Core staff' 'High skill tech staff' 'Cleaning staff'
 'Private service staff' 'Cooking staff' 'Low-skill Laborers'
 'Medicine staff' 'Secretaries' 'Walters/barmen staff' 'HR staff'
 'Realty agents' 'IT staff']

-----CNT_FAM_MEMBERS-----
2.0    233891
1.0     84492
3.0     77128
4.0     37356
5.0      5081
6.0       459
7.0       124
9.0         9
11.0         0
```

31°C Mostly sunny 09:32 13-04-2022

Desktop/Loan Approval Prediction: x Credit Card Approval Prediction x +

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 O

```
Out[20]: <function __main__.unique_values()>

In [21]: # dropping duplicate rows
app.drop_duplicates(subset = ['CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN',
                             'NAME_INCOME_TOTAL', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE',
                             'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'DAYS_BIRTH',
                             'DAYS_EMPLOYED', 'FLAG_MOBIL', 'FLAG_WORK_PHONE', 'FLAG_PHONE',
                             'FLAG_EMAIL', 'OCCUPATION_TYPE', 'CNT_FAM_MEMBERS'], keep = 'first', inplace = True)

app.shape

Out[21]: (90085, 18)
```

Data Cleaning

```
In [22]: def data_cleaning(data):
# Adding number of family members with number of children to get overall family members.
data['CNT_FAM_MEMBERS'] = data['CNT_FAM_MEMBERS'] + data['CNT_CHILDREN']

dropped_cols = ['FLAG_MOBIL', 'FLAG_WORK_PHONE', 'FLAG_PHONE',
                 'FLAG_EMAIL', 'OCCUPATION_TYPE', 'CNT_CHILDREN']
data = data.drop(dropped_cols, axis = 1)

#converting birth years and days employed to years.
data['DAYS_BIRTH'] = np.abs(data['DAYS_BIRTH'])/365 #Absolute
data['DAYS_EMPLOYED'] = data['DAYS_EMPLOYED']/365

#Cleaning up categorical values to lower the count of dummy variables.
housing_type = {'House / apartment': 'House / apartment',
                'With parents': 'With parents',
                'Municipal apartment': 'House / apartment',
                'Rented apartment': 'House / apartment',
                'Office apartment': 'House / apartment',
                'Co-op apartment': 'House / apartment'}

income_type = {'Commercial associate': 'Working',
               'State servant': 'Working',
               'Working': 'Working',
               'Pensioner': 'Pensioner',
               'Student': 'Student'}

education_type = {'Secondary / secondary special': 'secondary',
                  'Lower secondary': 'secondary',
```

31°C Mostly sunny 09:32 13-04-2022

Desktop/Loan Approval Prediction - Credit Card Approval Prediction

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

icdi Sathyabama Place... LMS SIST-LMS Kotak Learn R, Python &... Apple TV+ Sathyabama Institut... Covid-19 data visu... Seeing Theory Apple iCloud Google Yahoo

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```

    'Lower secondary':'secondary',
    'Higher education':'Higher education',
    'Incomplete higher':'Higher education',
    'Academic degree':'Academic degree')
family_status = {'Single / not married':'Single',
                 'Separated':'Single',
                 'Widow':'Single',
                 'Civil marriage':'Married',
                 'Married':'Married'})
data['NAME_HOUSING_TYPE'] = data['NAME_HOUSING_TYPE'].map(housing_type)
data['NAME_INCOME_TYPE'] = data['NAME_INCOME_TYPE'].map(income_type)
data['NAME_EDUCATION_TYPE'] = data['NAME_EDUCATION_TYPE'].map(education_type)
data['NAME_FAMILY_STATUS'] = data['NAME_FAMILY_STATUS'].map(family_status)
return data

In [23]: cleansed_app = data_cleansing(app)

In [24]: cleansed_app

Out[24]:

```

	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS
0	5006804	M	Y	Y	427500.0	Working	Higher education	
2	5006806	M	Y	Y	112500.0	Working	secondary	
3	5006808	F	N	Y	270000.0	Working	secondary	
7	5006812	F	N	Y	283500.0	Pensioner	Higher education	
10	5006815	M	Y	Y	270000.0	Working	Higher education	
...
43541	6837707	M	N	Y	202500.0	Working	Higher education	
43545	6839551	F	N	Y	99000.0	Pensioner	secondary	
43547	6839917	F	N	Y	180000.0	Pensioner	Higher education	
43552	6840104	M	N	Y	135000.0	Pensioner	secondary	
...

```

In [25]: # Data frame to analyze length of time since initial approval of credit card
# Shows number of past dues, paid off and no loan status.
grouped = credit.groupby('ID')

pivot_tb = credit.pivot(index = 'ID', columns = 'MONTHS_BALANCE', values = 'STATUS')
pivot_tb['open_month'] = grouped['MONTHS_BALANCE'].min()
pivot_tb['end_month'] = grouped['MONTHS_BALANCE'].max()
pivot_tb['window'] = pivot_tb['end_month'] - pivot_tb['open_month']
pivot_tb['window'] += 1 # Adding 1 since month starts at 0.

#Counting number of past dues, paid offs and no loans.
pivot_tb['paid_off'] = pivot_tb[pivot_tb.iloc[:,0:61] == 'C'].count(axis = 1)
pivot_tb['pastdue_1-9'] = pivot_tb[pivot_tb.iloc[:,0:61] == '0'].count(axis = 1)
pivot_tb['pastdue_30-59'] = pivot_tb[pivot_tb.iloc[:,0:61] == '1'].count(axis = 1)
pivot_tb['pastdue_60-89'] = pivot_tb[pivot_tb.iloc[:,0:61] == '2'].count(axis = 1)
pivot_tb['pastdue_90-119'] = pivot_tb[pivot_tb.iloc[:,0:61] == '3'].count(axis = 1)
pivot_tb['pastdue_120-149'] = pivot_tb[pivot_tb.iloc[:,0:61] == '4'].count(axis = 1)
pivot_tb['pastdue_over_150'] = pivot_tb[pivot_tb.iloc[:,0:61] == '5'].count(axis = 1)
pivot_tb['no_loan'] = pivot_tb[pivot_tb.iloc[:,0:61] == 'X'].count(axis = 1)
#Setting ID column to merge with app data.
pivot_tb['ID'] = pivot_tb.index

In [26]: pivot_tb.head()

Out[26]:

```

	MONTHS_BALANCE	-60	-59	-58	-57	-56	-55	-54	-53	-52	-51	...	window	paid_off	pastdue_1-9	pastdue_30-59	pastdue_60-89	pastdue_90-119	pastdue_120-149	pastdue_over_150	no_loan
ID																					
5001711	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	4	0	3	0	0	0	0	0	0
5001712	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	19	9	10	0	0	0	0	0	0
5001713	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	22	0	0	0	0	0	0	0	0
5001714	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	15	0	0	0	0	0	0	0	0
5001715	NaN	X	X	X	X	X	X	X	X	X	X	...	60	0	0	0	0	0	0	0	0

5 rows x 23 columns

Feature Engineering

Desktop/Loan Approval Prediction - Credit Card Approval Prediction

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

icdi Sathyabama Place... LMS SIST-LMS Kotak Learn R, Python &... Apple TV+ Sathyabama Institut... Covid-19 data visu... Seeing Theory Apple iCloud Google Yahoo

jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```

In [25]: # Data frame to analyze length of time since initial approval of credit card
# Shows number of past dues, paid off and no loan status.
grouped = credit.groupby('ID')

pivot_tb = credit.pivot(index = 'ID', columns = 'MONTHS_BALANCE', values = 'STATUS')
pivot_tb['open_month'] = grouped['MONTHS_BALANCE'].min()
pivot_tb['end_month'] = grouped['MONTHS_BALANCE'].max()
pivot_tb['window'] = pivot_tb['end_month'] - pivot_tb['open_month']
pivot_tb['window'] += 1 # Adding 1 since month starts at 0.

#Counting number of past dues, paid offs and no loans.
pivot_tb['paid_off'] = pivot_tb[pivot_tb.iloc[:,0:61] == 'C'].count(axis = 1)
pivot_tb['pastdue_1-9'] = pivot_tb[pivot_tb.iloc[:,0:61] == '0'].count(axis = 1)
pivot_tb['pastdue_30-59'] = pivot_tb[pivot_tb.iloc[:,0:61] == '1'].count(axis = 1)
pivot_tb['pastdue_60-89'] = pivot_tb[pivot_tb.iloc[:,0:61] == '2'].count(axis = 1)
pivot_tb['pastdue_90-119'] = pivot_tb[pivot_tb.iloc[:,0:61] == '3'].count(axis = 1)
pivot_tb['pastdue_120-149'] = pivot_tb[pivot_tb.iloc[:,0:61] == '4'].count(axis = 1)
pivot_tb['pastdue_over_150'] = pivot_tb[pivot_tb.iloc[:,0:61] == '5'].count(axis = 1)
pivot_tb['no_loan'] = pivot_tb[pivot_tb.iloc[:,0:61] == 'X'].count(axis = 1)
#Setting ID column to merge with app data.
pivot_tb['ID'] = pivot_tb.index

In [26]: pivot_tb.head()

Out[26]:

```

	MONTHS_BALANCE	-60	-59	-58	-57	-56	-55	-54	-53	-52	-51	...	window	paid_off	pastdue_1-9	pastdue_30-59	pastdue_60-89	pastdue_90-119	pastdue_120-149	pastdue_over_150	no_loan
ID																					
5001711	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	4	0	3	0	0	0	0	0	0
5001712	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	19	9	10	0	0	0	0	0	0
5001713	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	22	0	0	0	0	0	0	0	0
5001714	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	15	0	0	0	0	0	0	0	0
5001715	NaN	X	X	X	X	X	X	X	X	X	X	...	60	0	0	0	0	0	0	0	0

5 rows x 23 columns

Feature Engineering

Desktop/Loan Approval Prediction x Credit Card Approval Prediction x +

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

ici Sathyabama Place... LMS SIST-LMS Kotak Learn R, Python &... Apple TV+ Sathyabama Institut... Covid-19 data visu... Seeing Theory Apple iCloud Google Yahoo

Jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Feature Engineering

A ratio based method was used to create the target variable. For example, given a client with a time period of 60 months, if the client had paid off loan 40 times and was late 20 times, this would be considered a fairly good client given that there were more loans that were paid off on time compared to late payments. If a client had no loans throughout the initial approval of the credit card account, by default, this would be considered a good client as well. To identify a bad client, the number of past dues would exceed the number of loans paid off or if the client only has past dues. It may be better to incorporate a set difference between number of paid off loans and number of past dues. Meaning, there needs to be a significant gap between paid off loans and past dues. If a person has 50 past dues and 51 paid off loans, based on the ratio method, this would be considered good. However the difference is only 1 and this may not be a good sign of a good client. For simplicity sake, I will not adjust the algorithm further and keep it at ratio decisioning. Code is also not optimal, adjustment may be needed for the code to compute faster.

```
In [27]: def feature_engineering_target(data):
    good_or_bad = []
    for index, row in data.iterrows():
        paid_off = row['paid_off']
        over_1 = row['pastdue_1-29']
        over_30 = row['pastdue_30-59']
        over_60 = row['pastdue_60-89']
        over_90 = row['pastdue_90-119']
        over_120 = row['pastdue_120-149'] + row['pastdue_over_150']
        no_loan = row['no_loan']

        overall_pastdues = over_1+over_30+over_60+over_90+over_120

        if overall_pastdues == 0:
            if paid_off > no_loan or paid_off <= no_loan:
                good_or_bad.append(1)
            elif paid_off == 0 and no_loan == 1:
                good_or_bad.append(1)
        elif overall_pastdues != 0:
            if paid_off > overall_pastdues:
                good_or_bad.append(1)
            elif paid_off <= overall_pastdues:
                good_or_bad.append(0)
        elif paid_off == 0 and no_loan != 0:
            if overall_pastdues < no_loan or overall_pastdues >= no_loan:
                good_or_bad.append(0)
        else:
            good_or_bad.append(1)
```

31°C Mostly sunny 09:33 13-04-2022

Desktop/Loan Approval Prediction x Credit Card Approval Prediction x +

localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb

ici Sathyabama Place... LMS SIST-LMS Kotak Learn R, Python &... Apple TV+ Sathyabama Institut... Covid-19 data visu... Seeing Theory Apple iCloud Google Yahoo

Jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

There is data on clients in the credit data that intersect with the application data. the following is a merge between the two data frames given on the data of clients that exist in both data sets. Featured engineered additional columns from the credit data.

```
In [ ]: target = pd.DataFrame()
target['ID'] = pivot_tb.index
target['paid_off'] = pivot_tb['paid_off'].values
target['#_of_pastdues'] = pivot_tb['pastdue_1-29'].values+pivot_tb['pastdue_30-59'].values
+pivot_tb['pastdue_60-89'].values+pivot_tb['pastdue_90-119'].values
+pivot_tb['pastdue_120-149'].values+pivot_tb['pastdue_over_150'].values

target['no_loan'] = pivot_tb['no_loan'].values
target['target'] = feature_engineering_target(pivot_tb)
credit_app = cleansed_app.merge(target, how = 'inner', on = 'ID')
credit_app.drop('ID', axis = 1, inplace = True)

In [ ]: credit_app.head()

In [ ]: from sklearn.preprocessing import LabelEncoder

cg = LabelEncoder()
oc = LabelEncoder()
own_r = LabelEncoder()
it = LabelEncoder()
et = LabelEncoder()
fs = LabelEncoder()
ht = LabelEncoder()

credit_app['CODE_GENDER'] = cg.fit_transform(credit_app['CODE_GENDER'])
credit_app['FLAG_OWN_CAR'] = oc.fit_transform(credit_app['FLAG_OWN_CAR'])
credit_app['FLAG_OWN_REALTY'] = own_r.fit_transform(credit_app['FLAG_OWN_REALTY'])
credit_app['NAME_INCOME_TYPE'] = it.fit_transform(credit_app['NAME_INCOME_TYPE'])
credit_app['NAME_EDUCATION_TYPE'] = et.fit_transform(credit_app['NAME_EDUCATION_TYPE'])
credit_app['NAME_FAMILY_STATUS'] = fs.fit_transform(credit_app['NAME_FAMILY_STATUS'])
credit_app['NAME_HOUSING_TYPE'] = ht.fit_transform(credit_app['NAME_HOUSING_TYPE'])

In [ ]: print('CODE_GENDER', credit_app['CODE_GENDER'].unique())
print(cg.inverse_transform(list(credit_app['CODE_GENDER'].unique())))
print()
print('FLAG_OWN_CAR:', credit_app['FLAG_OWN_CAR'].unique())
print(oc.inverse_transform(list(credit_app['FLAG_OWN_CAR'].unique())))
```

31°C Mostly sunny 09:33 13-04-2022

```
Desktop/Loan Approval Prediction x Credit Card Approval Prediction x +
localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb
jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved)
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 O
In [ ]: print('CODE_GENDER', credit_app['CODE_GENDER'].unique())
print(cg.inverse_transform(list(credit_app['CODE_GENDER'].unique())))
print()
print('FLAG_OWN_CAR', credit_app['FLAG_OWN_CAR'].unique())
print(oc.inverse_transform(list(credit_app['FLAG_OWN_CAR'].unique())))
print()
print('FLAG_OWN_REALTY', credit_app['FLAG_OWN_REALTY'].unique())
print(own_r.inverse_transform(list(credit_app['FLAG_OWN_REALTY'].unique())))
print()
print('NAME_INCOME_TYPE', credit_app['NAME_INCOME_TYPE'].unique())
print(it.inverse_transform(list(credit_app['NAME_INCOME_TYPE'].unique())))
print()
print('NAME_EDUCATION_TYPE', credit_app['NAME_EDUCATION_TYPE'].unique())
print(et.inverse_transform(list(credit_app['NAME_EDUCATION_TYPE'].unique())))
print()
print('NAME_FAMILY_STATUS', credit_app['NAME_FAMILY_STATUS'].unique())
print(fs.inverse_transform(list(credit_app['NAME_FAMILY_STATUS'].unique())))
print()
print('NAME_HOUSING_TYPE', credit_app['NAME_HOUSING_TYPE'].unique())
print(ht.inverse_transform(list(credit_app['NAME_HOUSING_TYPE'].unique())))

In [ ]: x = credit_app.drop('target', axis = 1).columns
y = credit_app['target']
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.3, random_state = 0)

Imported SMOTE in case of high imbalance of target variable. However, it does not seem to be highly imbalanced so it is fine not to use an imbalanced
technique.

In [ ]: def logistic_reg(xtrain, xtest, ytrain, ytest):
l=LogisticRegression(solver='liblinear')
lr.fit(xtrain, ytrain)
ypred_lr=lr.predict(xtest)
print('***LogisticRegression***')
print('Confusion matrix:')
print(confusion_matrix(ytest, ypred))
print('Classification report:')
print(classification_report(ytest, ypred))

In [ ]: def random_forest(xtrain, xtest, ytrain, ytest):
rf=RandomForestClassifier()
rf.fit(xtrain, ytrain)
ypred_rf=rf.predict(xtest)
print('***RandomForestClassifier***')
```

```
Desktop/Loan Approval Prediction x Credit Card Approval Prediction x +
localhost:8888/notebooks/Desktop/Loan%20Approval%20Prediction/Credit%20Card%20Approval%20Prediction%20-%20With%20Label%20Encode.ipynb
jupyter Credit Card Approval Prediction - With Label Encode Last Checkpoint: 12/17/2021 (autosaved)
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 O
In [ ]: def g_boosting(xtrain, xtest, ytrain, ytest):
gb=GradientBoostingClassifier()
gb.fit(xtrain, ytrain)
ypred_gb=gb.predict(xtest)
print('***GradientBoostingClassifier***')
print('Confusion matrix:')
print(confusion_matrix(ytest, ypred))
print('Classification report:')
print(classification_report(ytest, ypred))

In [ ]: def d_tree(xtrain, xtest, ytrain, ytest):
dt=DecisionTreeClassifier()
dt.fit(xtrain, ytrain)
ypred_dt=dt.predict(xtest)
print('***DecisionTreeClassifier***')
print('Confusion matrix:')
print(confusion_matrix(ytest, ypred))
print('Classification report:')
print(classification_report(ytest, ypred))

In [ ]: def compare_model(xtrain, xtest, ytrain, ytest):
logistic_reg(xtrain, xtest, ytrain, ytest)
print("-"*100)
random_forest(xtrain, xtest, ytrain, ytest)
print("-"*100)
g_boosting(xtrain, xtest, ytrain, ytest)
print("-"*100)
d_tree(xtrain, xtest, ytrain, ytest)

In [ ]: compare_model(xtrain, xtest, ytrain, ytest)

In [ ]: dt=DecisionTreeClassifier()
dt.fit(xtrain, ytrain)
ypred_dt=dt.predict(xtest)

In [ ]: dt.predict([[1,1,1,112500,0,2,2,0,0,58,832877,-3.106849,2,0,7,7,16]])

In [ ]: import pickle
pickle.dump(dt, open("model.pkl", "wb"))
```