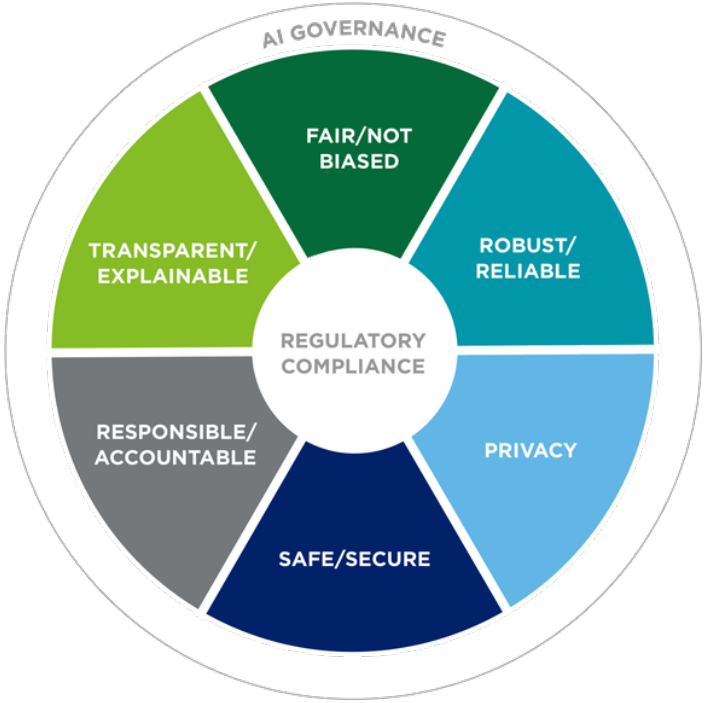
Deloitte.

Group E5

Capstone Project

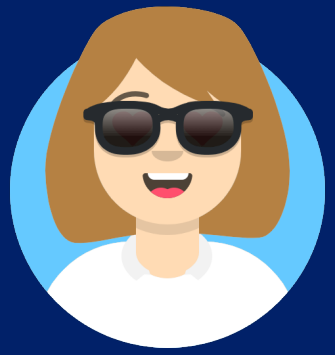
A requisite submission for the
completion of Deloitte AI Academy™



JIGSAW ACADEMY
a manipal group investee company

Meet the Team.

Chandrasekhar Gudipati



Shruti Maigur

Sushant Mudalgi

Sushanth Ganesh

Problem Statement.

Let’s see how various parameters affect loan default.

Problem Statement.

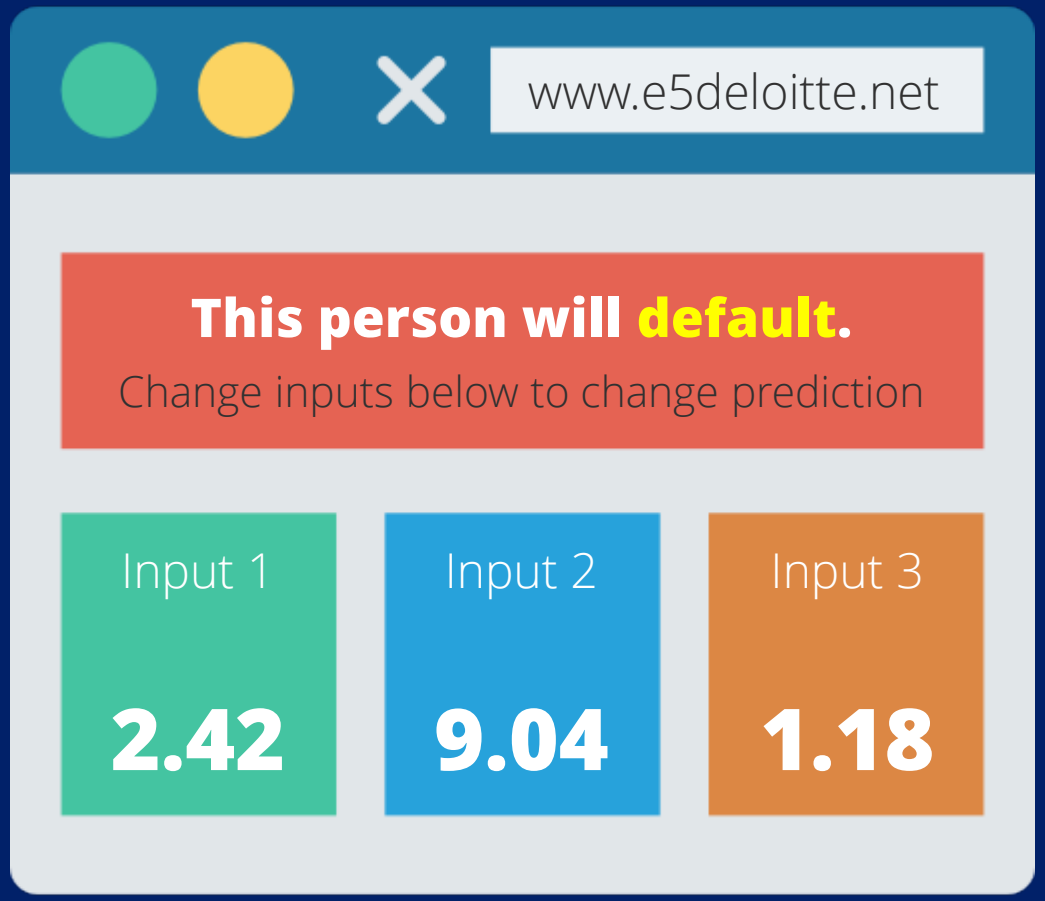
Hey, can you give me a loan with the following details,

Lorem Ipsum...

Let me check with the software brought to us by the folks at

Team E5

Problem Statement.



Individual Data Files.

Loan_details.csv

Branch_region_
mapping.csv

Loan_status.csv

Individual Data Files.

total_table.csv

Combined Data.

Loan_ id	disbursed_ amount	asset_ cost	ltv	branch_id	Date.of. Birth	Employment. Type	Disbursal Date	MobileNo_Avl _Flag	Aadhar_ flag
PAN_flag	VoterID_ flag	Driving_ flag	Passport_ flag	PERFORM_ CNS. SCORE	DELINQUENT .ACCTS.IN.LA ST.SIX.MONT HS	CREDIT. HISTORY. LENGTH	NO.OF_ INQUIRIES	loan_ default	region

Work Methodology.

Limited
Time

Unlimited
Exploration Possibilities

Work Methodology.

Waterfall vs Agile

Work Methodology.

Work Methodology.

- **Two meetings were held each week.**
- **Once one member presented his/her work, others gave constructive criticism.**
- **This feedback was used to improve results.**

So we'll let you in on our little Masterplan.

01 Exploration and Basic Visualization
> Poke the bear by going through the data and identifying patterns and correlations using visualizations and plots

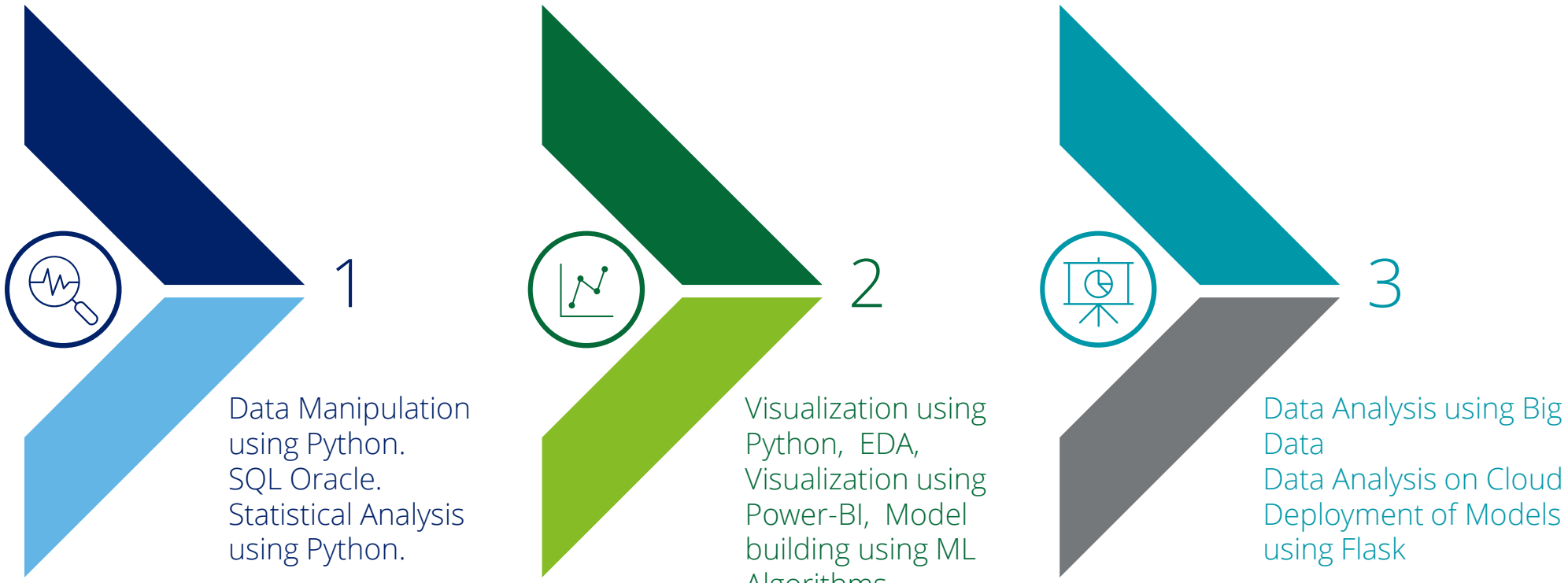
02 Interactive Visualizations
> Create Tableau and PowerBI Dashboards that allow for interactive visualizations

03 PreProcessing
> Preprocess the data based on the inferences made

04 AI/ML
> Create various AI models based on the inferences and select the best of the lot

05 Data Engineering & Cloud
> Upload them to the cloud for the world to enjoy

RoadMap.


1
Data Manipulation using Python.
SQL Oracle.
Statistical Analysis using Python.
2
Visualization using Python, EDA, Visualization using Power-BI, Model building using ML Algorithms
3
Data Analysis using Big Data
Data Analysis on Cloud
Deployment of Models using Flask


1
2
3

18 October 2021

Milestone **One**

Individual Contributions.

Chandrasekhar Gudipati	1.1, PowerPoint
Shruti Maigur	1.3
Sushant Mudalgi	1.2
Sushanth Ganesh	1.2

1.1

Data Manipulation using Python

1.2

SQL

1.3

Statistical Analysis using Python

1.1
Data Manipulation
using Python

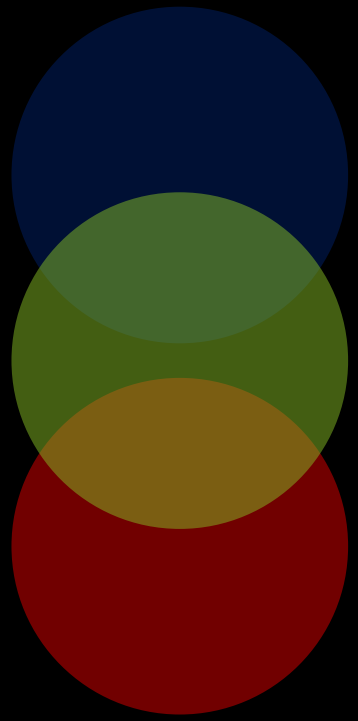
1.2
SQL

1.3
Statistical Analysis
using Python

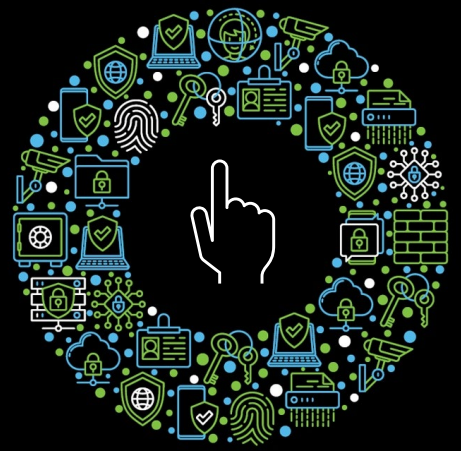
Chandrasekhar Gudipati

1.1 Data Manipulation using Python

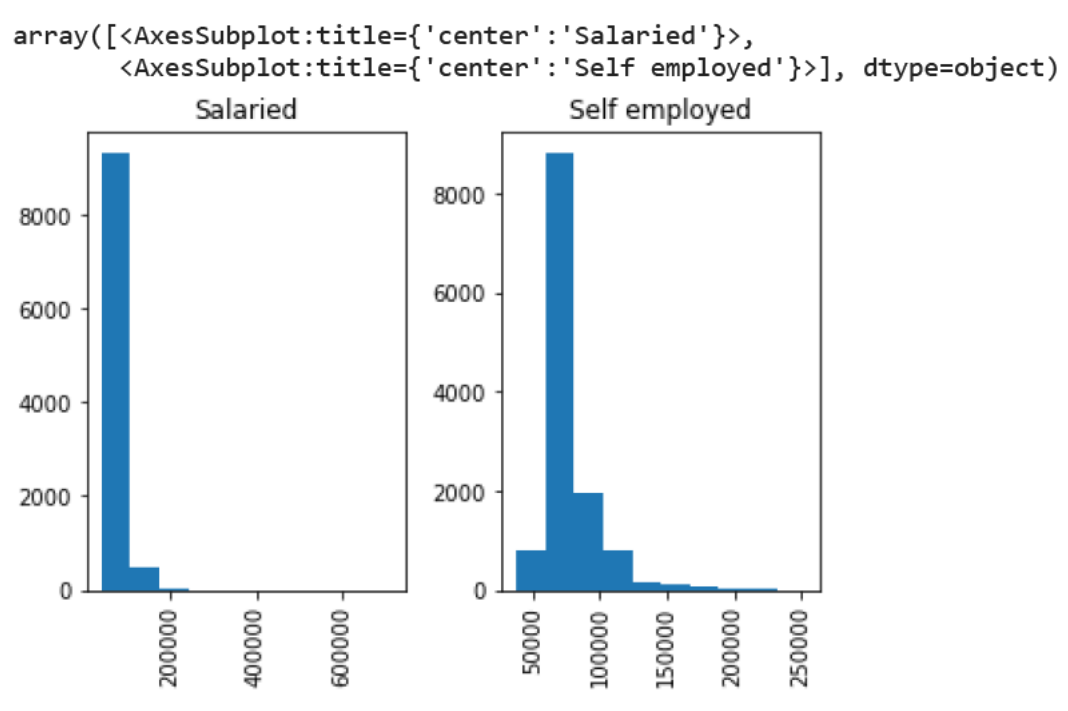
Table Merging.



To begin with, we combined the data of the three given csv files into one pandas dataframe called total_table. Here's how.

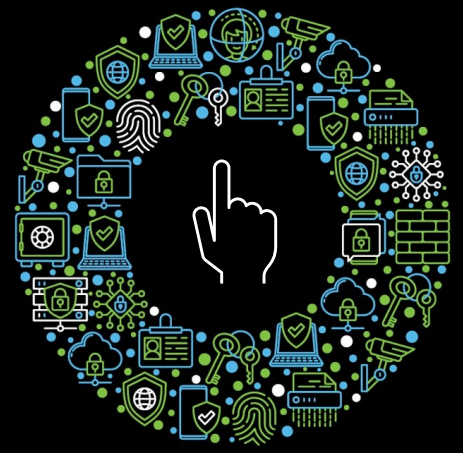


Looking Around.

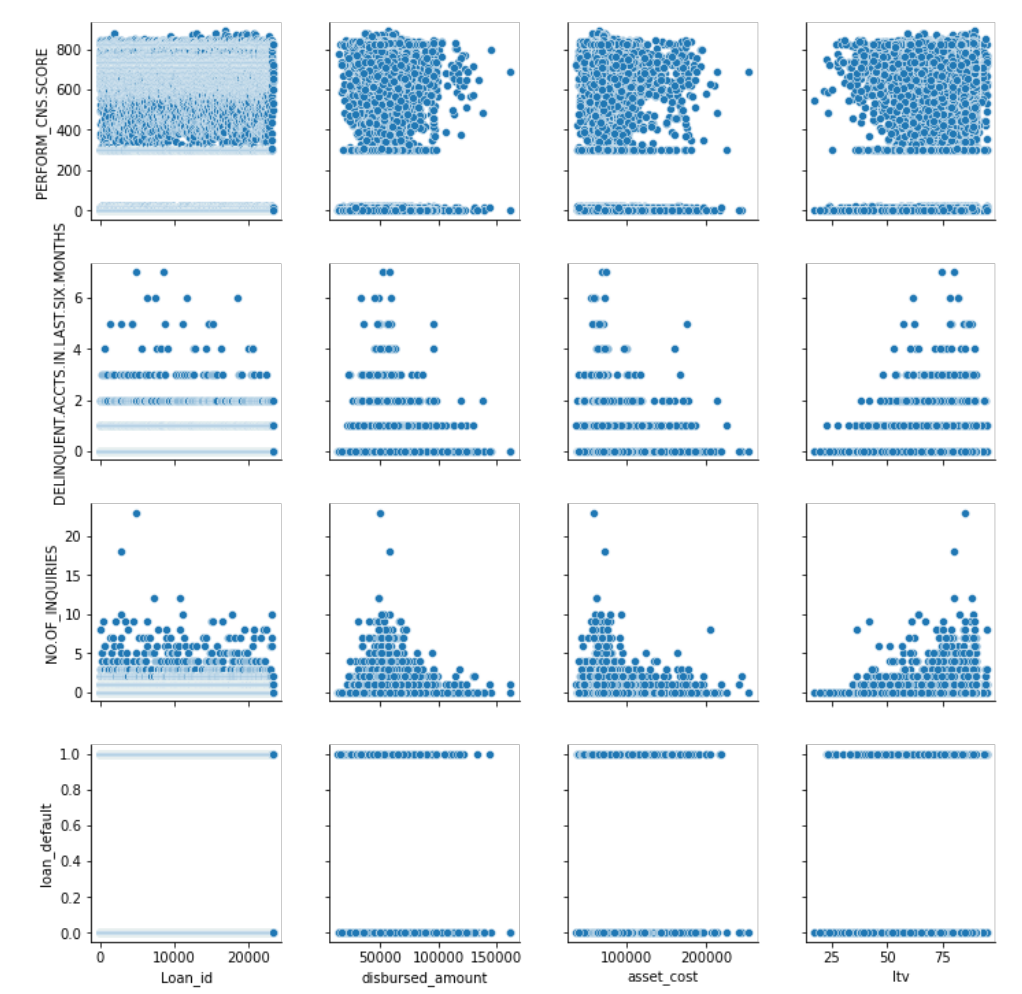


Mean, Median, Mode

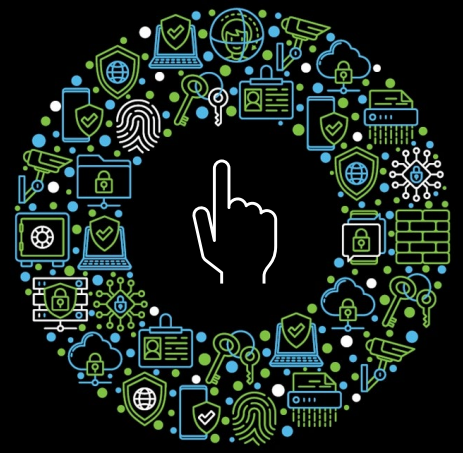
Then we began multiple analyses as alongside.
Click to begin.



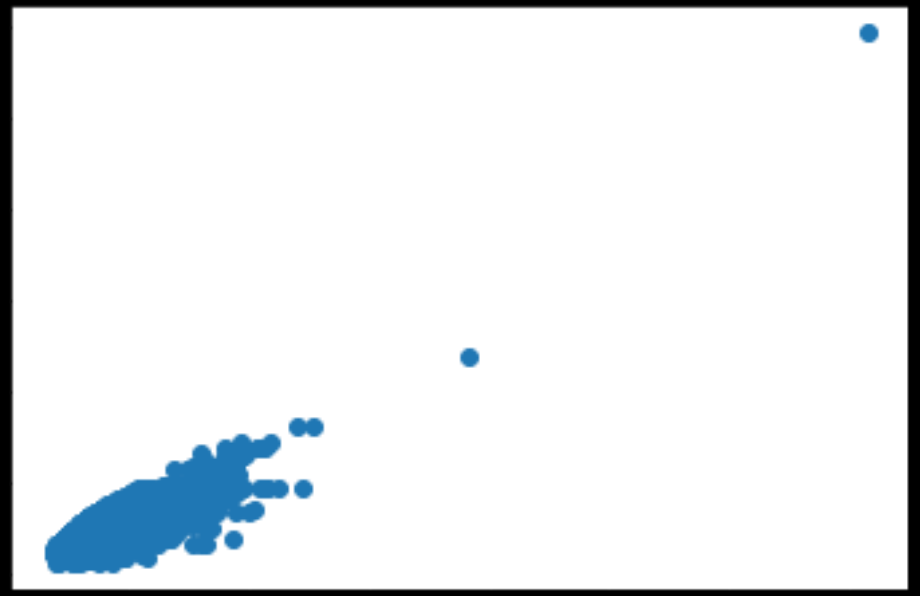
Outliers Outliers Everywhere.



Handling outliers is a job of perspective. One person's outlier is another's gold mine. Click to know more.

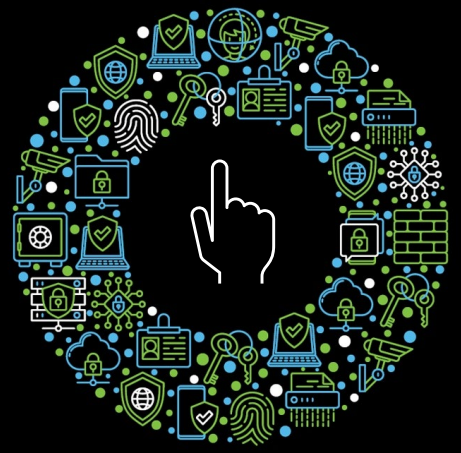


Outliers Outliers Everywhere.



Asset Cost (X) vs Disbursed Amount(Y)

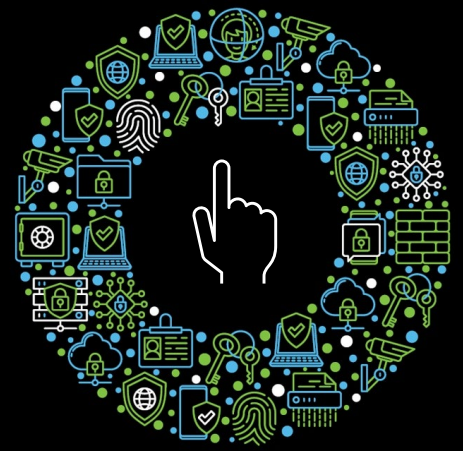
Both these outliers belong to bank 138. So guess where we started our detective work? Click to know more.



Reading between the Lines.

	Loan_id	disbursed_amount	asset_cost	ltv	branch_id	Aadhar_flag	PAN_flag
Loan_id	1.000000	-0.006935	-0.003720	-0.005458	0.001564	0.004079	0.002083
disbursed_amount	-0.006935	1.000000	0.752629	0.372385	0.024890	-0.017392	0.013246
asset_cost	-0.003720	0.752629	1.000000	-0.304288	0.023997	-0.089512	0.043821
ltv	-0.005458	0.372385	-0.304288	1.000000	0.010844	0.098845	-0.032283
branch_id	0.001564	0.024890	0.023997	0.010844	1.000000	-0.041416	0.026347
Aadhar_flag	0.004079	-0.017392	-0.089512	0.098845	-0.041416	1.000000	-0.188042
PAN_flag	0.002083	0.013246	0.043821	-0.032283	0.026347	-0.188042	1.000000
VoterID_flag	-0.004142	0.016849	0.084220	-0.091096	0.025477	-0.862401	0.164873
Driving_flag	-0.001820	-0.001977	0.015421	-0.021724	-0.014848	-0.287130	0.000230
Passport_flag	-0.004154	0.001242	-0.005971	0.010548	-0.011556	-0.082545	-0.002240
PERFORM_CNS.SCORE	0.003165	0.009414	-0.052328	0.088537	-0.016329	0.067788	0.022877
DELINQUENT.ACCTS.IN.LAST.SIX.MONTHS	-0.002406	0.020351	-0.002257	0.038652	-0.004391	0.022699	-0.000499
NO.OF_INQUIRIES	-0.005668	0.035428	-0.018974	0.087960	0.000974	0.028507	0.024491
loan_default	-0.010647	0.062805	0.002535	0.095423	0.037361	-0.050624	0.007771

We love the vibrant colours on this one. Click below for a chromatic feast. We understood a lot of patterns through these correlation plots.



1.1

Data Manipulation using Python

1.2

SQL

1.3

Statistical Analysis using Python

1.1

Data Manipulation using Python

1.2

SQL

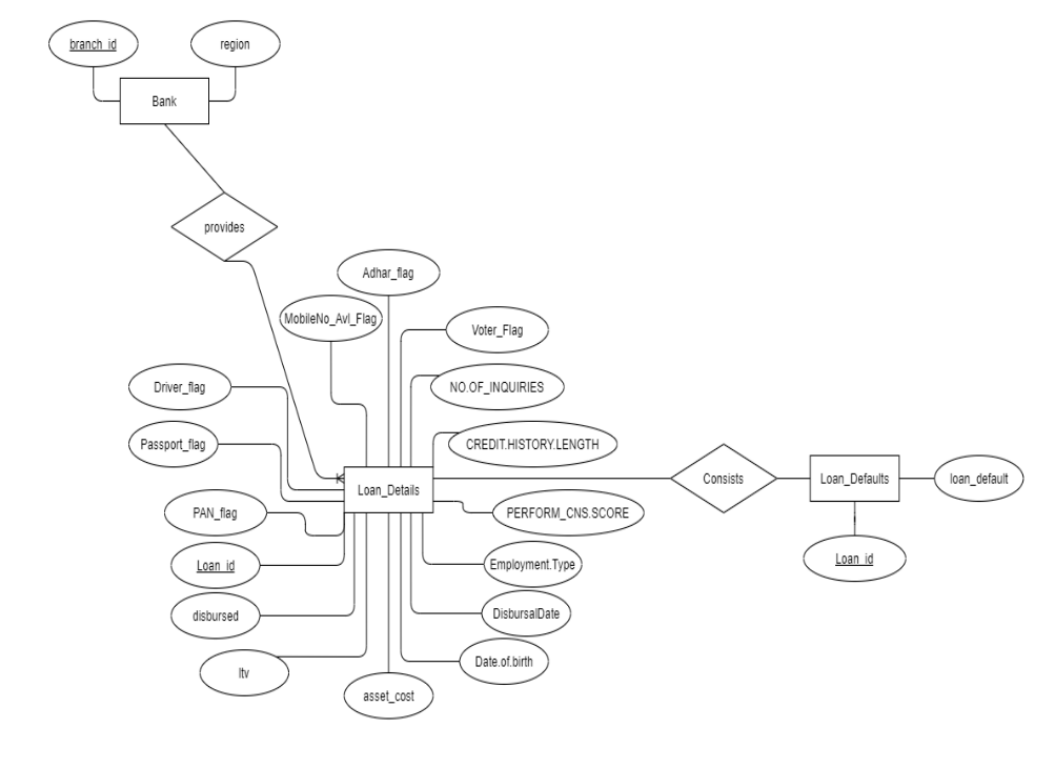
1.3

Statistical Analysis using Python

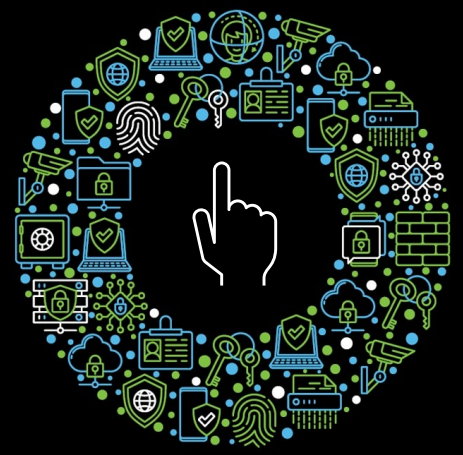
Sushant Mudalgi, Sushanth Ganesh

1.2 SQL

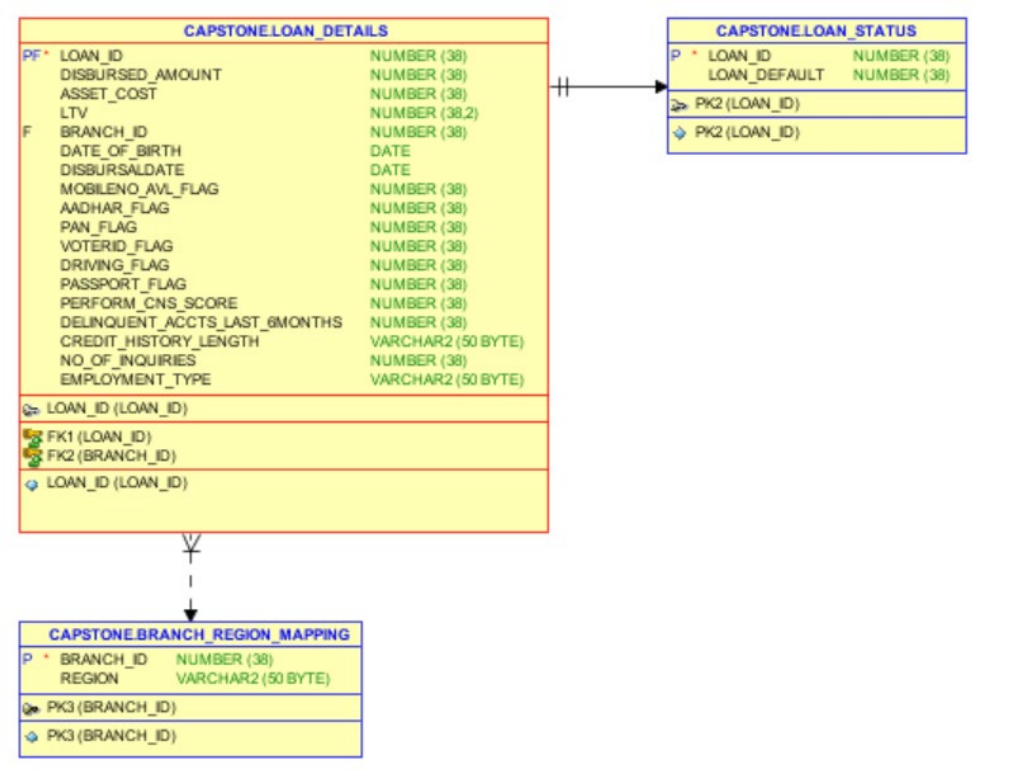
ER Diagram.



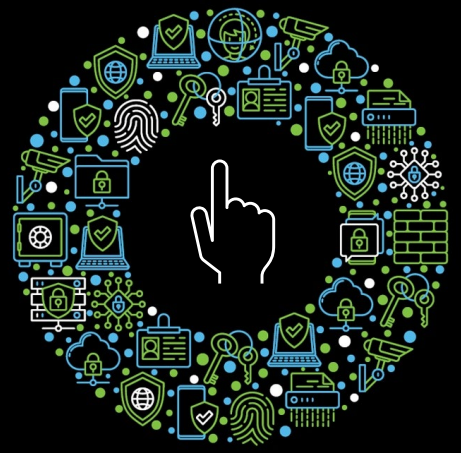
We started with an ER Diagram to get a basic understanding of our work. Click to begin.



Functioning.



This was generated from SQL Developer tool after we created the table and assigned the keys. Click to proceed.



1.1

Data Manipulation using Python

1.2

SQL

1.3

Statistical Analysis using Python

1.1

Data Manipulation using Python

1.2

SQL

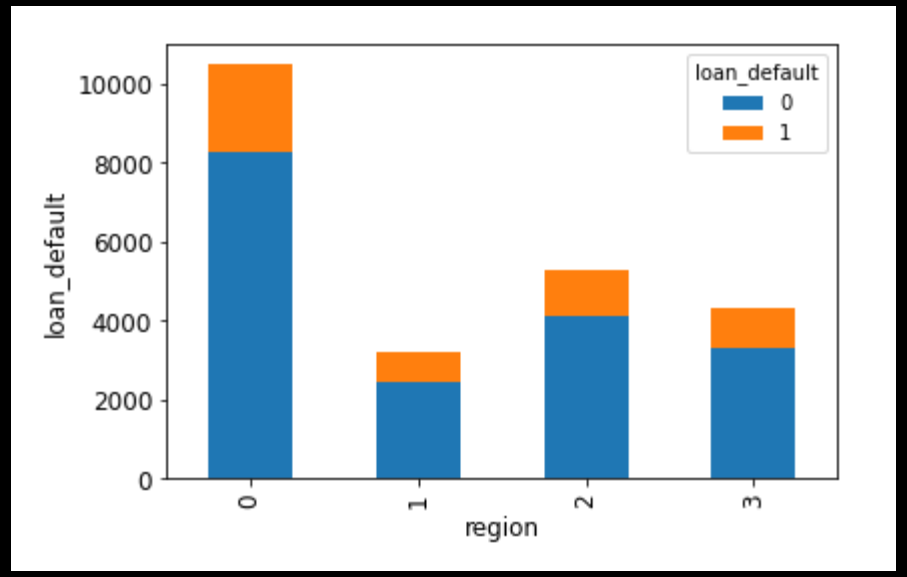
1.3

Statistical Analysis using Python

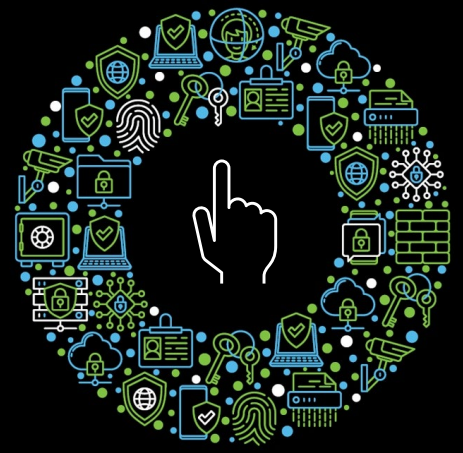
Shruti Maigur

1.3 Statistical Analysis using Python

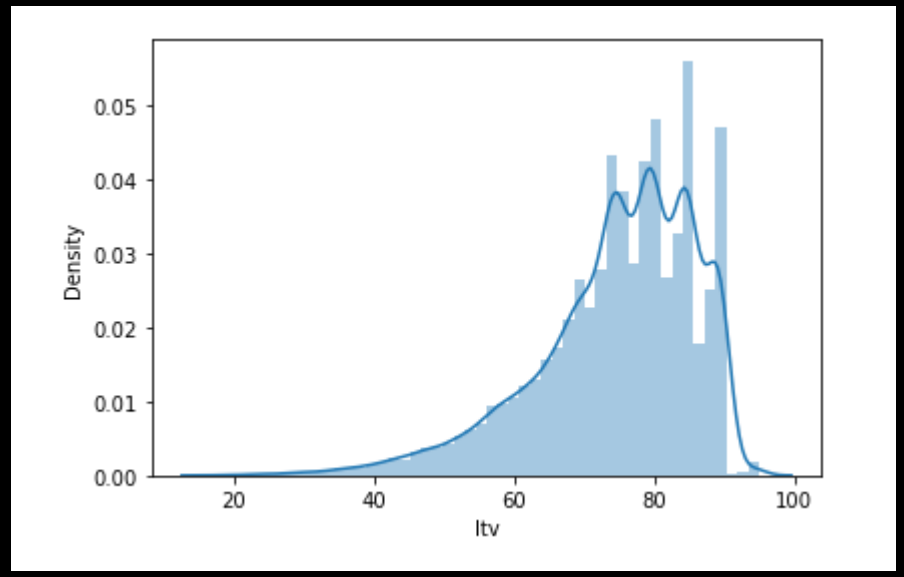
Descriptive Statistics.



To understand the data just a little more, we began descriptive statistical analyses. Click to know more.



Hypothesis Testing.

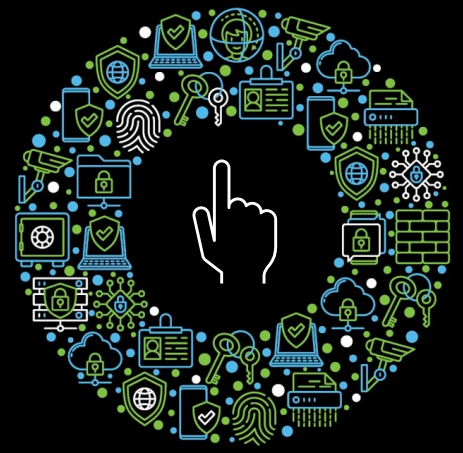



```
1  Data['ltv'].skew()
```

```
-1.0760494824413254
```

- Data is not normally distributed.
- Applying T-test on mean on pop and sample,
- p-value (0.1517) > alpha (0.05)

To make sure that all our assumptions were correct and to understand the attributes, we performed hypothesis testing. Click below.




1
2
3

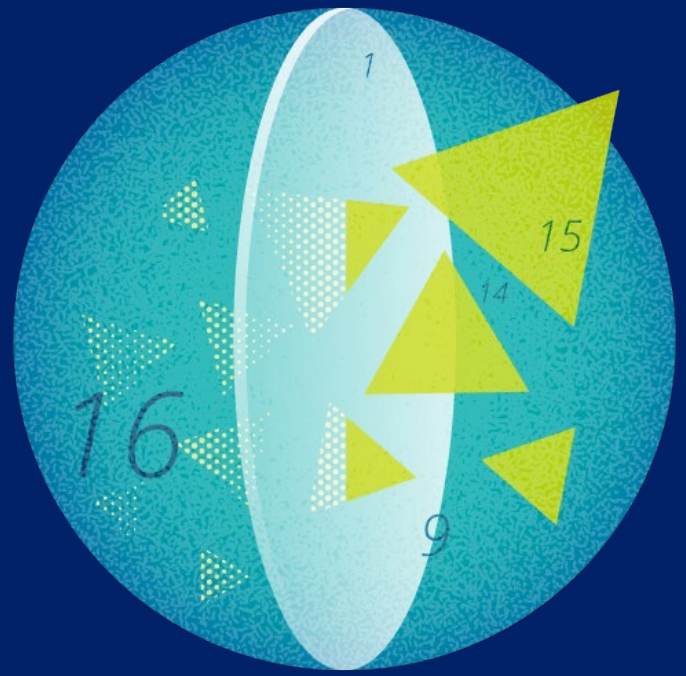
15 November 2021

Milestone **Two**

README.

E_5_milestone_2.zip

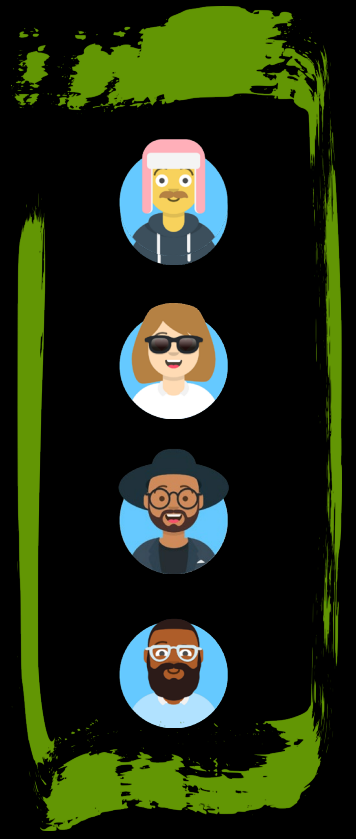
- E_5_milestone_2.pptx
- E_5_Task_2.1.html
- E_5_Task_2.2.html
- E_5_Task_2.3.pbix
- E_5_Task_2.4.html



Our final advice to the bank is
located at the end of this very PowerPoint presentation

Individual Contributions.

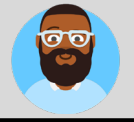
	CG	SMa	SMu	SG
2.1				
2.2				
2.3				
2.4				



Analyst Deck

Presentation by **Chandrasekhar Gudipati.**

2.1. Visualization using Python

	CG	SMa	SMu	SG
2.1				
2.2				
2.3				
2.4				

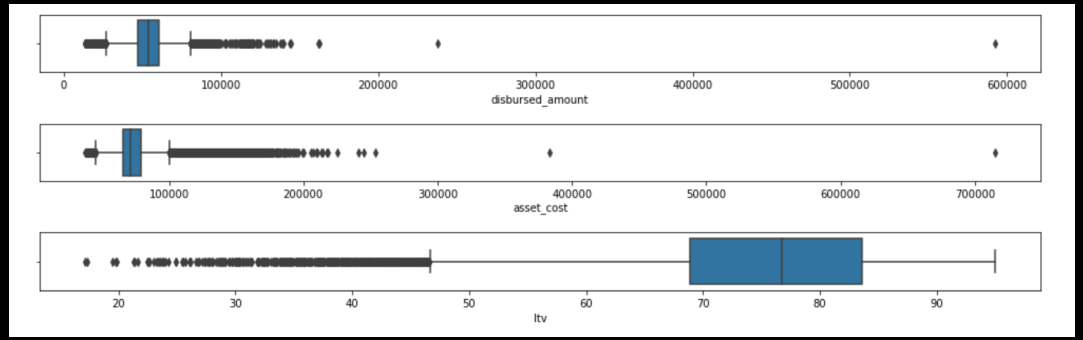
Presentation by **Chandrasekhar Gudipati.**

Analyst Deck

2.1. Visualization using Python

To Do List

- Percentage of default
- Imbalanced data
- Missing values
- Outliers
- Analyze default variable with demographic related data
- Association between default and other variables




```
ra=[]
comb = {}
for i in range(0,len(ma)):
    ra.append(ma[i]-mi[i])
#print(df1.columns[2])
print("range: \n")
for j in range(0,len(ra)):
    comb[df1.columns[j]] = ra[j]
    #print(df1.columns[j],"\t\t\t",ra[j])
#print(comb)
for key, value in comb.items():
    print(key, ' : ', value)
```

```
range:

Loan_id  :  23314.0
disbursed_amount  :  579091.0
asset_cost  :  677956.0
ltv  :  77.85000000000001
branch_id  :  260.0
MobileNo_Avl_Flag  :  0.0
Aadhar_flag  :  1.0
PAN_flag  :  1.0
VoterID_flag  :  1.0
Driving_flag  :  1.0
Passport_flag  :  1.0
PERFORM_CNS.SCORE  :  890.0
DELINQUENT.ACCTS.IN.LAST.SIX.MONTHS  :  7.0
NO.OF_INQUIRIES  :  23.0
loan_default  :  1.0
```

```
df1.quantile(q=0.25)
```

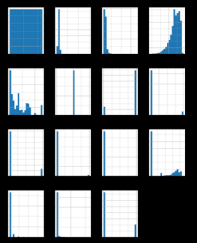
```
Loan_id                              5829.50
disbursed_amount                    46949.00
asset_cost                          65629.00
ltv                                    68.83
branch_id                              13.00
MobileNo_Avl_Flag                       1.00
Aadhar_flag                             1.00
PAN_flag                                0.00
VoterID_flag                            0.00
Driving_flag                            0.00
Passport_flag                           0.00
PERFORM_CNS.SCORE                       0.00
DELINQUENT.ACCTS.IN.LAST.SIX.MONTHS     0.00
NO.OF_INQUIRIES                         0.00
loan_default                            0.00
Name: 0.25, dtype: float64
```

```
df1.var()
```

```
df1.std()
```

```
df1.skew(skipna = True)
```

```
kurt = df1.kurt()
kurt
```



default: 5126
total: 23315
Default%: 21.985%

2.1. Visualization using Python

To Do List

- Percentage of default
- Imbalanced data
- Missing values
- Outliers
- Analyze default variable with demographic related data
- Association between default and other variables

```
total_deleted = total_table.dropna()
total_deleted
```

```
print("total_shape rows: ",total_table.shape[0])
print("total_shape rows: ",total_deleted.shape[0])
print("lost rows: ",total_table.shape[0]-total_deleted.shape[0])
print("lost rows percentage: ",((total_table.shape[0]-total_deleted.shape[0])/total_table.shape[0])*100)
```

```
total_shape rows:  23315
total_shape rows:  22545
lost rows:  770
lost rows percentage:  3.3025948959897065
```

Despite my reservations against dropping ***n/a*** values, we will proceed with **total_deleted** because that is what the task requires.

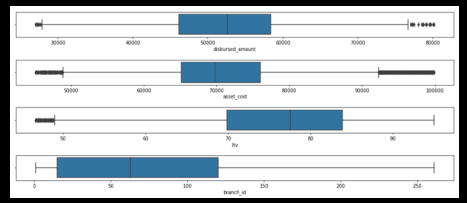
```
df2 = df1.dropna()
```

```
def Remove_Outlier_Indices(df):
    Q1 = df.quantile(0.25)
    Q3 = df.quantile(0.75)
    IQR = Q3 - Q1
    trueList = ~((df < (Q1 - 1.5 * IQR)) |(df > (Q3 + 1.5 * IQR)))
    return trueList
```

```
# Index List of Non-Outliers
nonOutlierList = Remove_Outlier_Indices(df2)

# Non-Outlier Subset of the Given Dataset
dfSubset = df2[nonOutlierList]
```

```
dfSubset2 = dfSubset.dropna()
dfSubset2
```

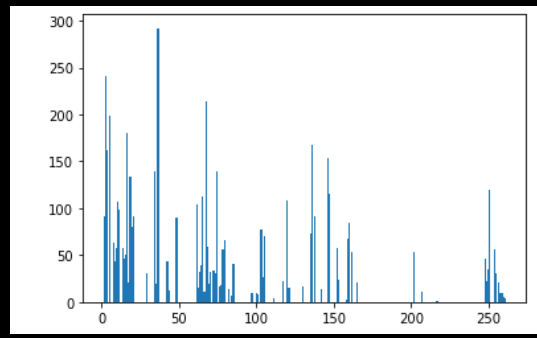


2.1. Visualization using Python

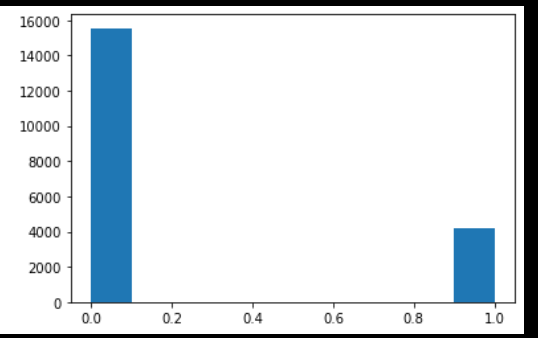
To Do List

- Percentage of default
- Imbalanced data
- Missing values
- Outliers
- Analyze default variable with demographic related data
- Association between default and other variables

branch_id (x) and count(loan_default) (y)



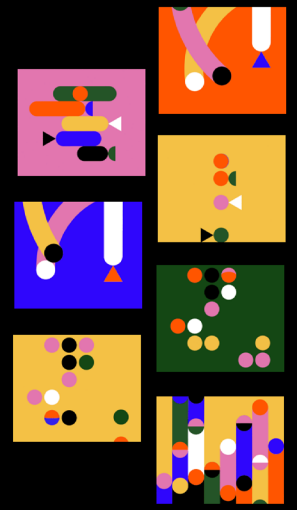
loan_default (x) and aadhaar_flag (y)



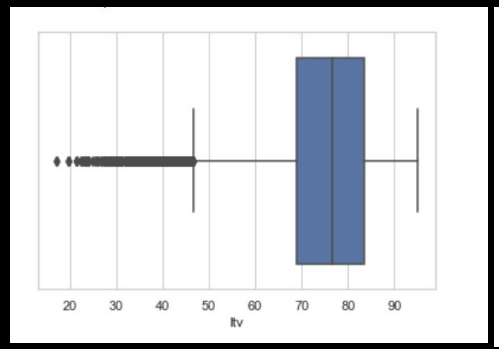

```
total_table.corr()['loan_default']
```

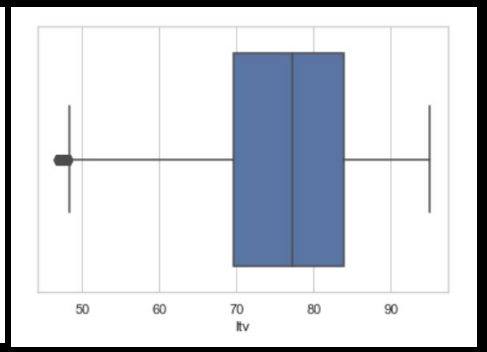
Loan_id	-0.010647
disbursed_amount	0.062805
asset_cost	0.002535
ltv	0.095423
branch_id	0.037361
MobileNo_Avl_Flag	NaN
Aadhar_flag	-0.050624
PAN_flag	0.007771
VoterID_flag	0.046974
Driving_flag	0.004321
Passport_flag	-0.010403
PERFORM_CNS.SCORE	-0.060836
DELINQUENT.ACCTS.IN.LAST.SIX.MONTHS	0.032289
NO.OF_INQUIRIES	0.042042
loan_default	1.000000

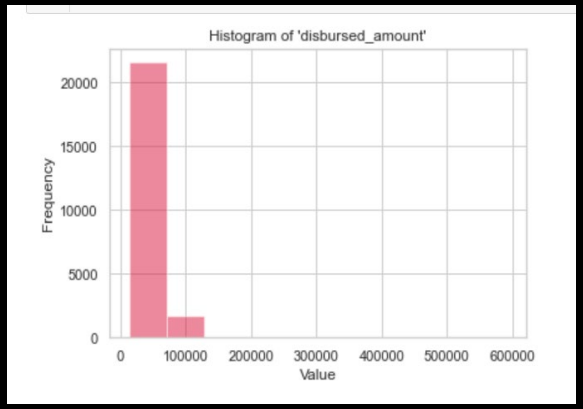
Name: loan_default, dtype: float64

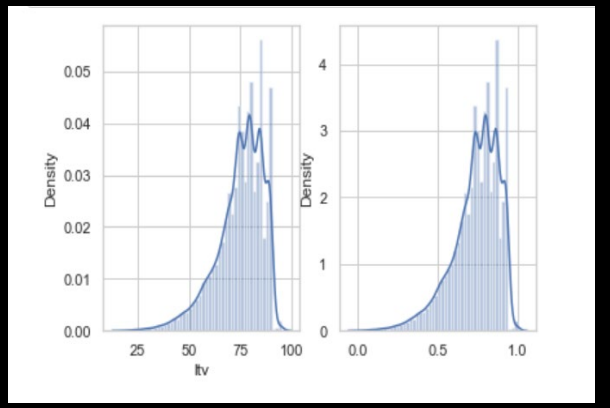


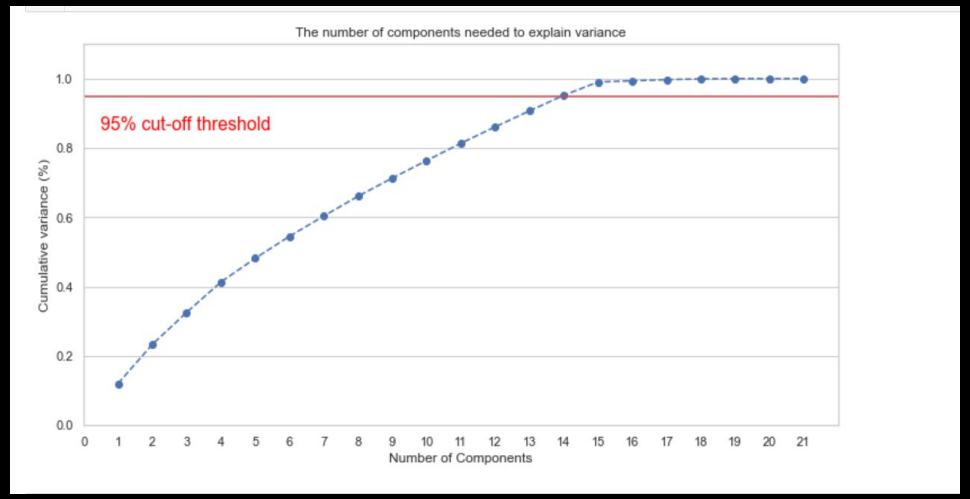
2.2. Exploratory Data Analysis

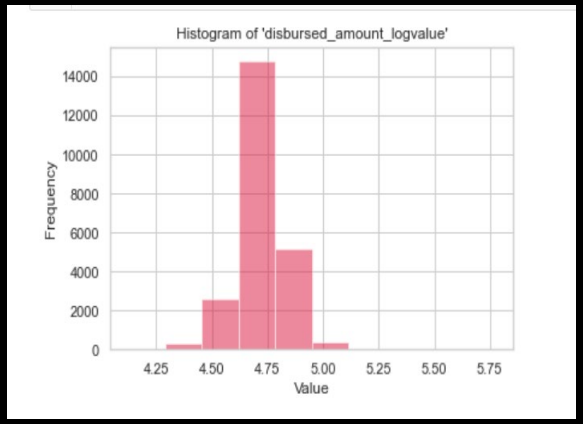


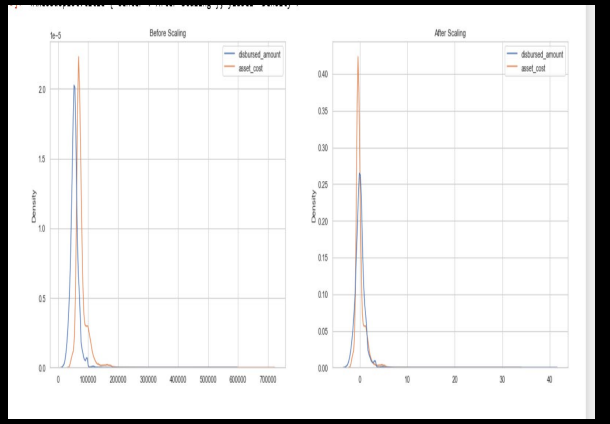




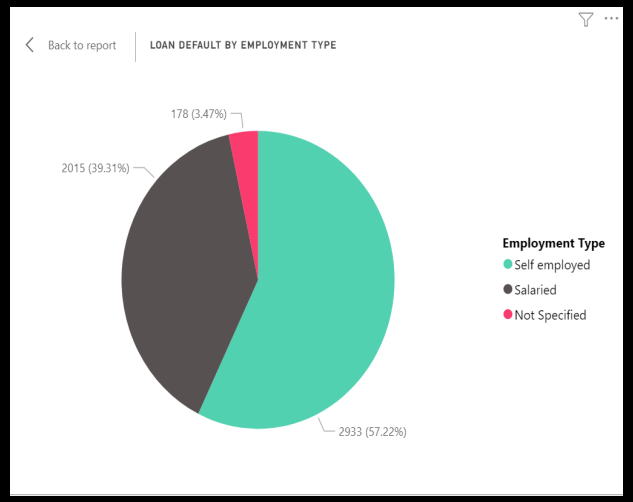


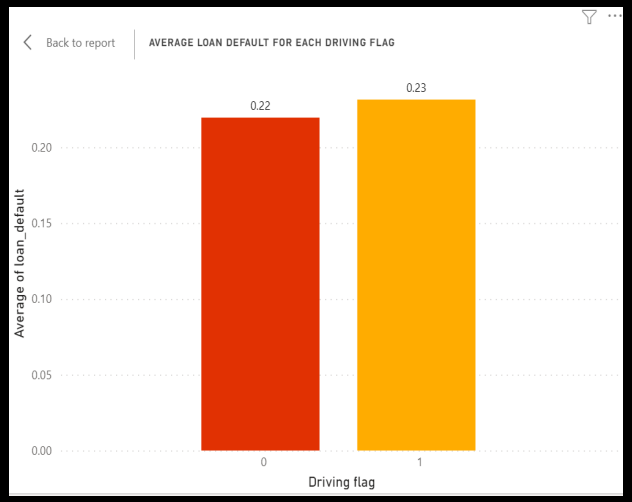


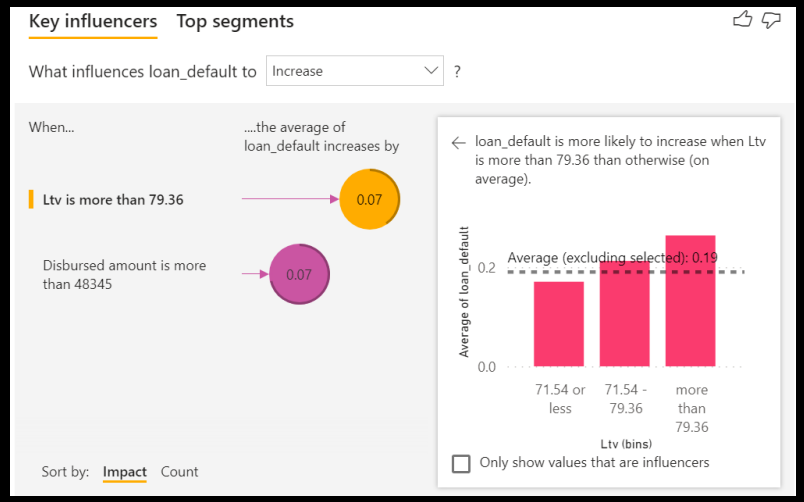




2.3. Visualization using Power-BI



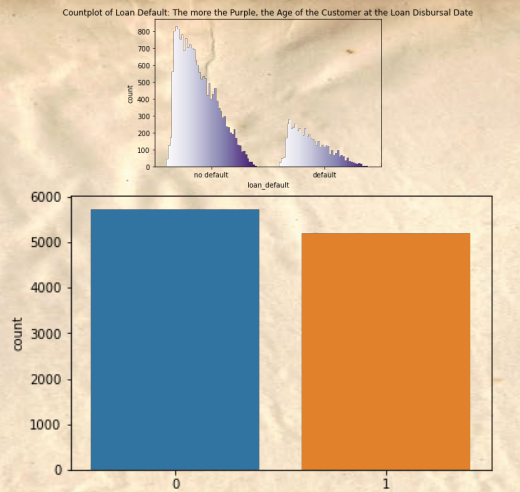




2.4. Model Building using ML Algos

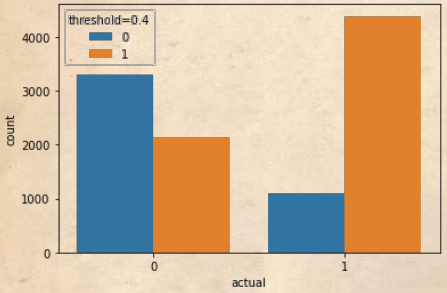
Model 1

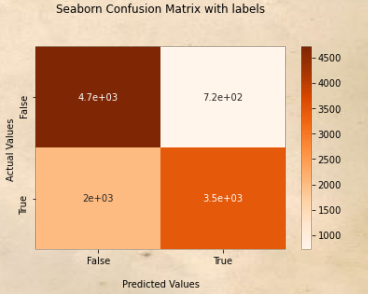
Logistic Regression



All hail Lord **SMOTE!**

Model 2 Linear Regression with Manual Thresholds

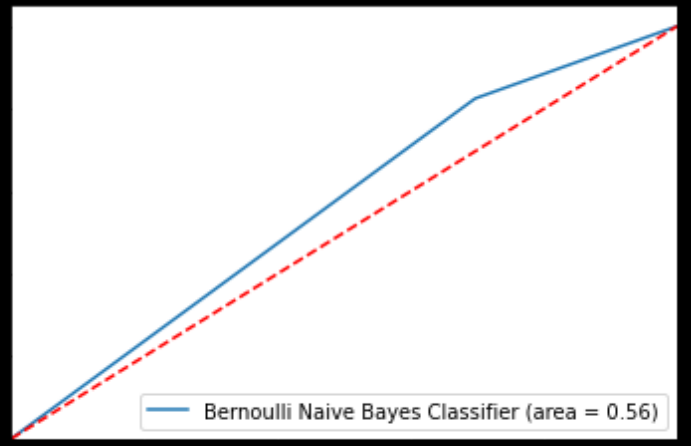


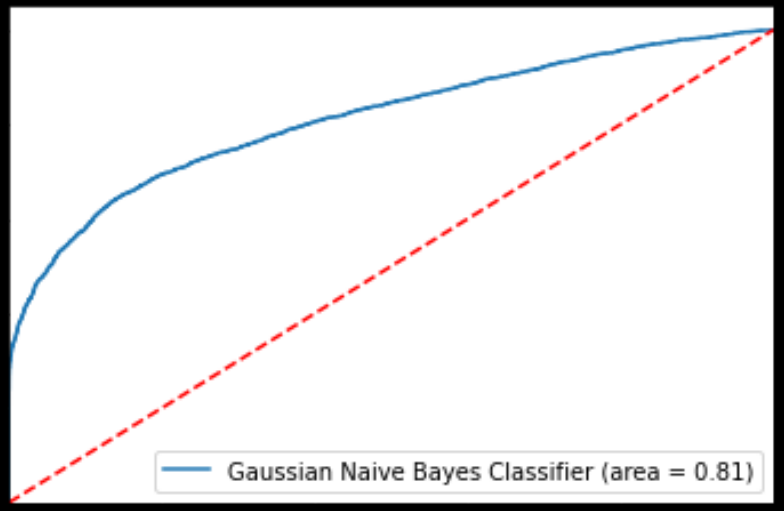


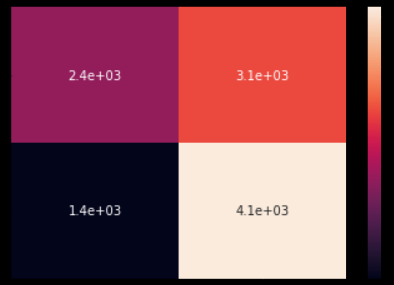
* This 'model' is just an ultra elaborate demonstration of the effects of changing parameters on results and its final results are not to be taken seriously

2.4. Model Building using ML Algos

Model 3:
Gaussian
Naïve Bayes
Classifier



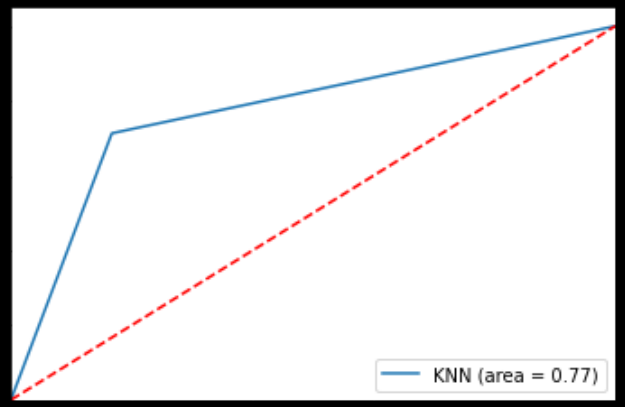




Model 4: Bernoulli
Naïve Bayes Classifier

2.4. Model Building using ML Algos

Model 5: K Nearest Neighbours

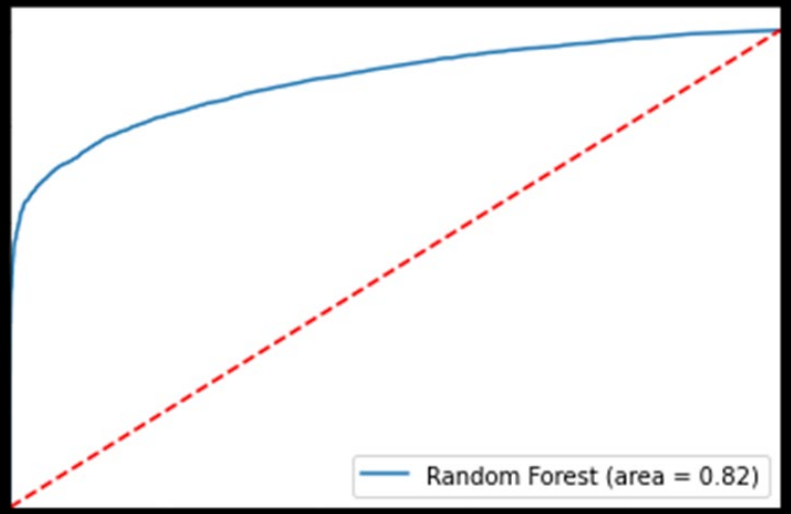


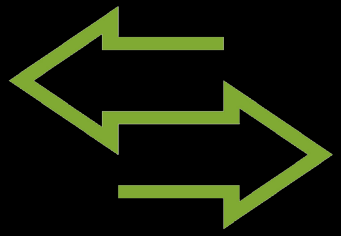
accuracy -> 0.6408

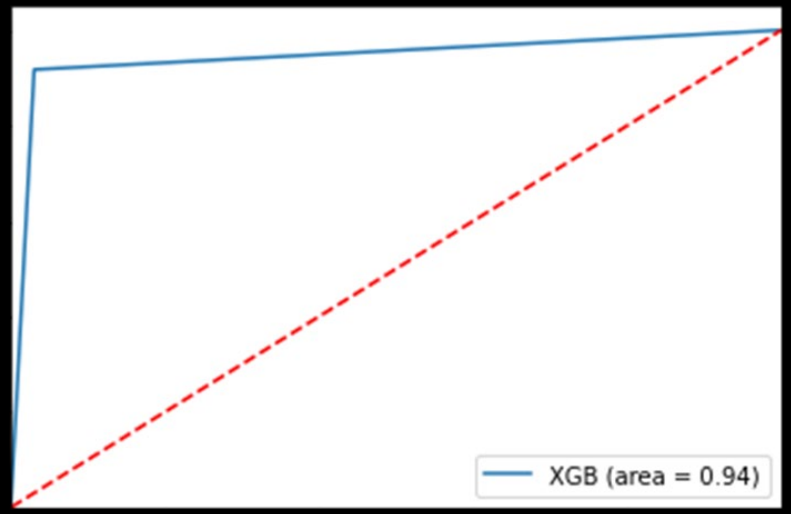
Model 6: Decision Trees Classifier

[[4014, 1425],
[2495, 2980]]

2.4. Model Building using ML Algos







accuracy -> 0.8201 [[4834, 605], [1358, 4117]]

accuracy -> 0.9422 [[5248, 191], [439, 5036]]

Model 7a: Random Forest

Model 7b: XGB

2.4. Comparisons

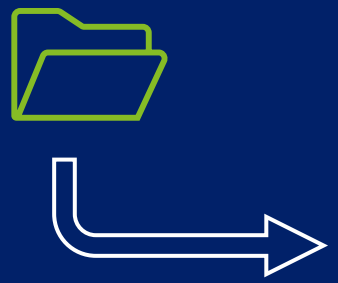
Model	Accuracy	Model	Accuracy
Logistic Regression	0.65	Decision Trees Classifier	0.64
Gaussian	0.81	Random Forest	0.82
Bernoulli	0.56	XGB	0.94
KNN	0.77		


1
2
3

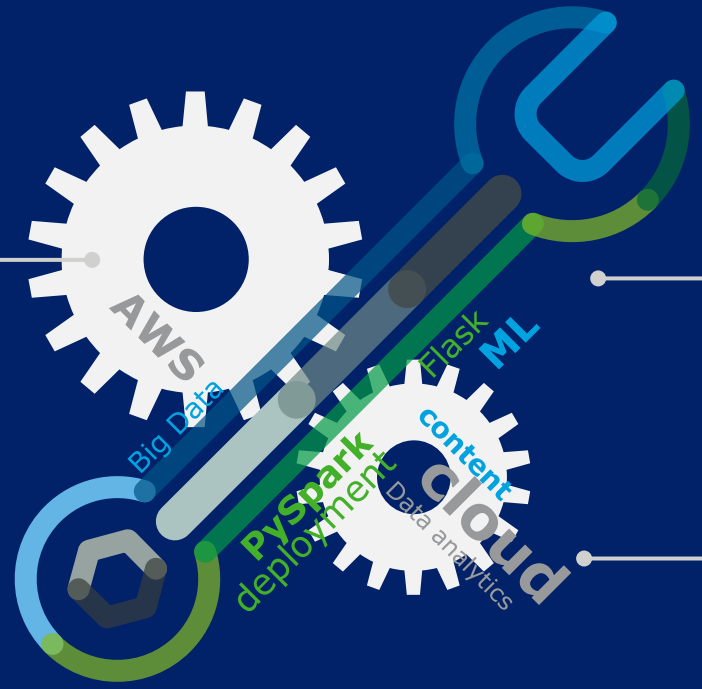
01 December 2021

Milestone **Three**

Read_me.



3.1
Data Analysis using Big Data Tools



3.2
Data Analysis on Cloud (AWS)

3.3
Deployment of Models using Flask

Task by **Shruti Maigur and Sushant Mudalgi.** Presentation by **Chandrasekhar Gudipati.**

3.1 Data Analysis using Big Data Tools.


```
df.describe()
```

```
DataFrame[summary: string, disbursed_amount: string, asset_cost: string, ltv: string, PERFORM_CNS_SCORE: string, b
ranch_id: string, MobileNo_Avl_Flag: string, Aadhar_flag: string, PAN_flag: string, VoterID_flag: string, Driving_
flag: string, Passport_flag: string, DELINQUENT_ACCTS_IN_LAST_SIX_MONTHS: string, NO_OF_INQUIRIES: string, loan_de
fault: string, Date_of_Birth_month: string, DisbursalDate_month: string, Date_of_Birth_year: string, DisbursalDate
_year: string, CREDIT_HISTORY_MONTHS_SCALED: string, age_at_disb_norm: string, Emp_Type: string, region_East: stri
ng, region_North: string, region_South: string, region_West: string]
```

```
spark.sql("select DISTINCT region from Data ").show()
```

```
+------+
|region|
+------+
| South|
|  East|
|  West|
| North|
+------+
```

```
spark.sql("select DISTINCT Employment_Type from Data ").show()
```

```
+---------------+
|Employment_Type|
+---------------+
|  Self employed|
|       Salaried|
+---------------+
```

```
defaults_perc_by_region=defaults_and_nonDefaults_by_region.loc[defaults_and_nonDefaults_by_region['loan_default'] =
= 1]
```

```
defaults perc by region['percentage']=(defaults perc by region['count']/defaults perc by region['count'].sum())*100
```

```
defaults perc by region
```

	region	loan_default	count	percentage
4	West	1	966	19.523040
5	South	1	1132	22.877930
6	North	1	724	14.632175
7	East	1	2126	42.966855

Saving output file in parquet format.

```
Customers_defaulted.coalesce(1).write.save("ParquetFiles/CustomersDefaulted.parquet", format="parquet")
```

Reading the data from the stored parquet file

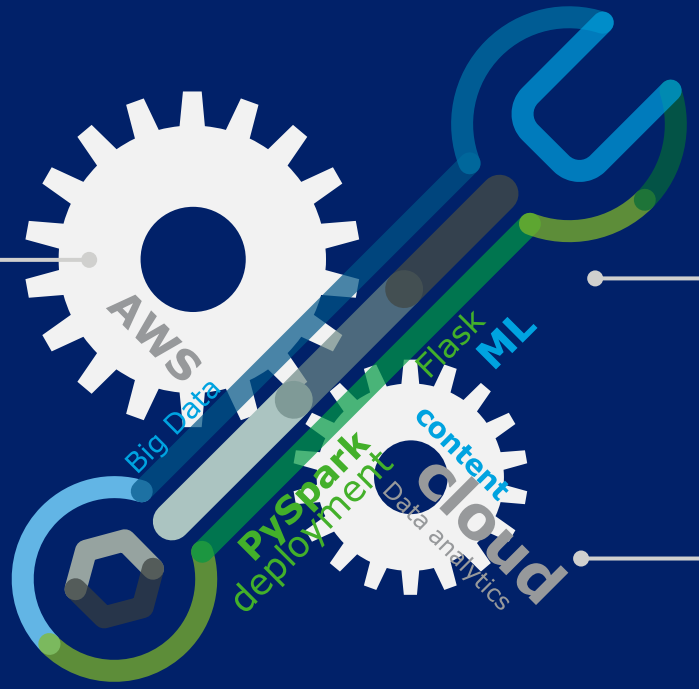
```
parquetFileDF = spark.read.parquet("ParquetFiles/CustomersDefaulted.parquet")
```

```
parquetFileDF.createOrReplaceTempView("DefaultersParquetFile")
```

```
spark.sql("select * from DefaultersParquetFile").show(1)
```

```
+--------------------------+---------------+---------+
|Defaulted_customers_loan_id|disbursed_amount|branch_id|
+--------------------------+---------------+---------+
|                         3|          55348|        2|
+--------------------------+---------------+---------+
only showing top 1 row
```

3.1
Data Analysis using Big Data Tools



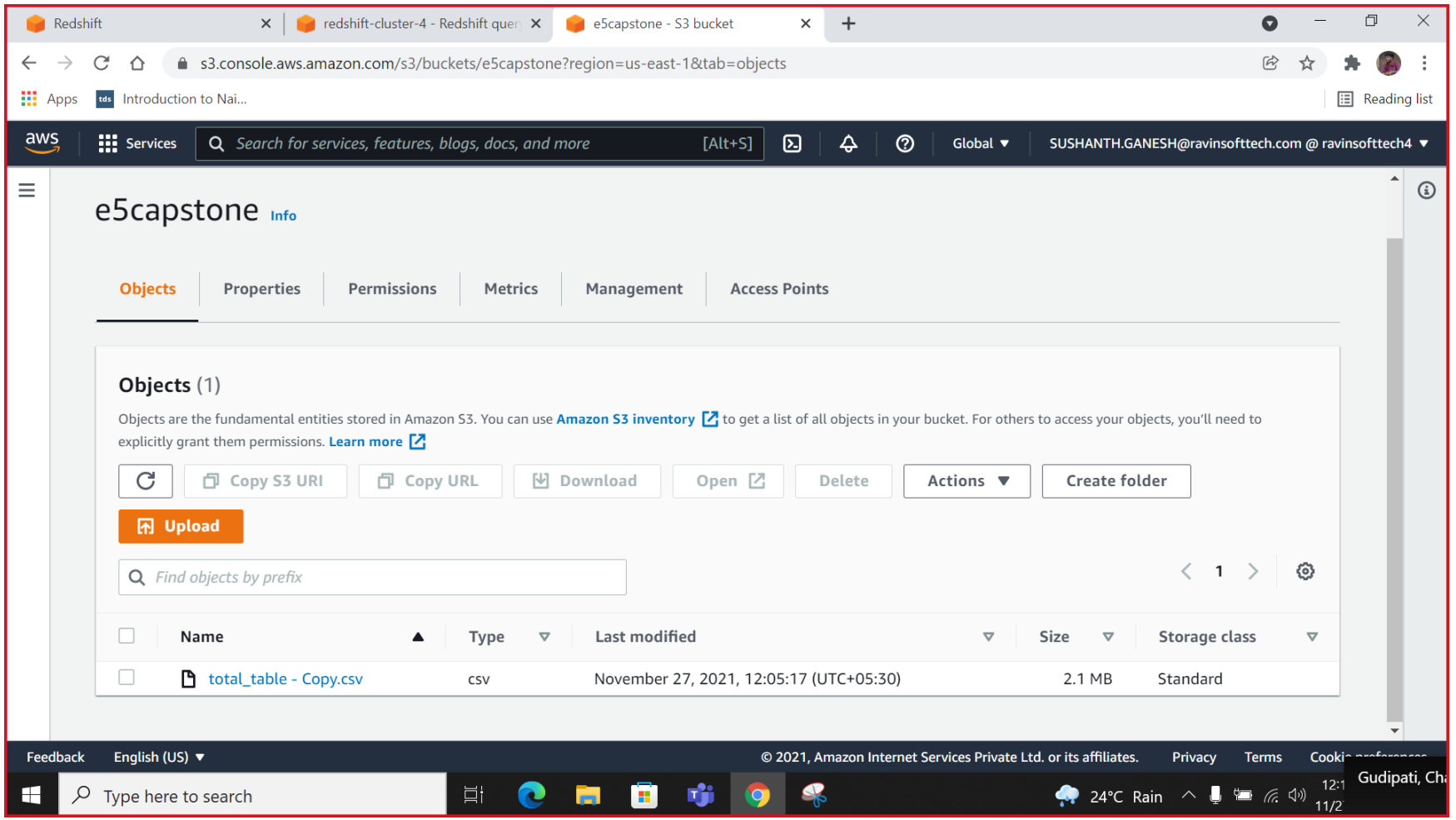
3.2
Data Analysis on Cloud (AWS)

3.3
Deployment of Models using Flask

Task by **Sushanth Ganesh and Chandrasekhar Gudipati.** Presentation by **Chandrasekhar Gudipati.**

Data Analysis on Cloud (AWS).

amazon S3

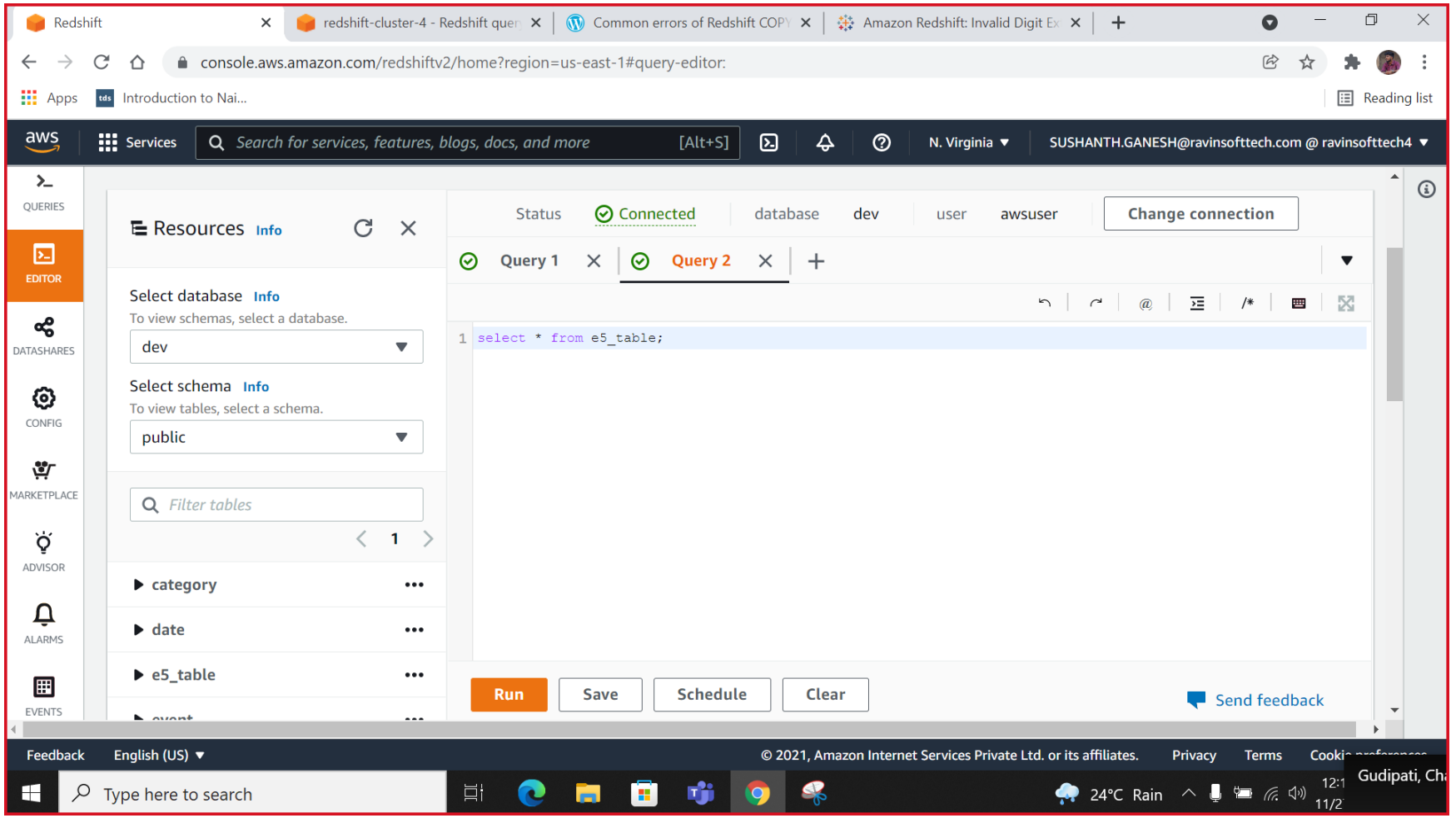


amazon S3

Uploading data to Amazon S3 (Simple Storage Service)

Data Analysis on Cloud (AWS).

Amazon Redshift

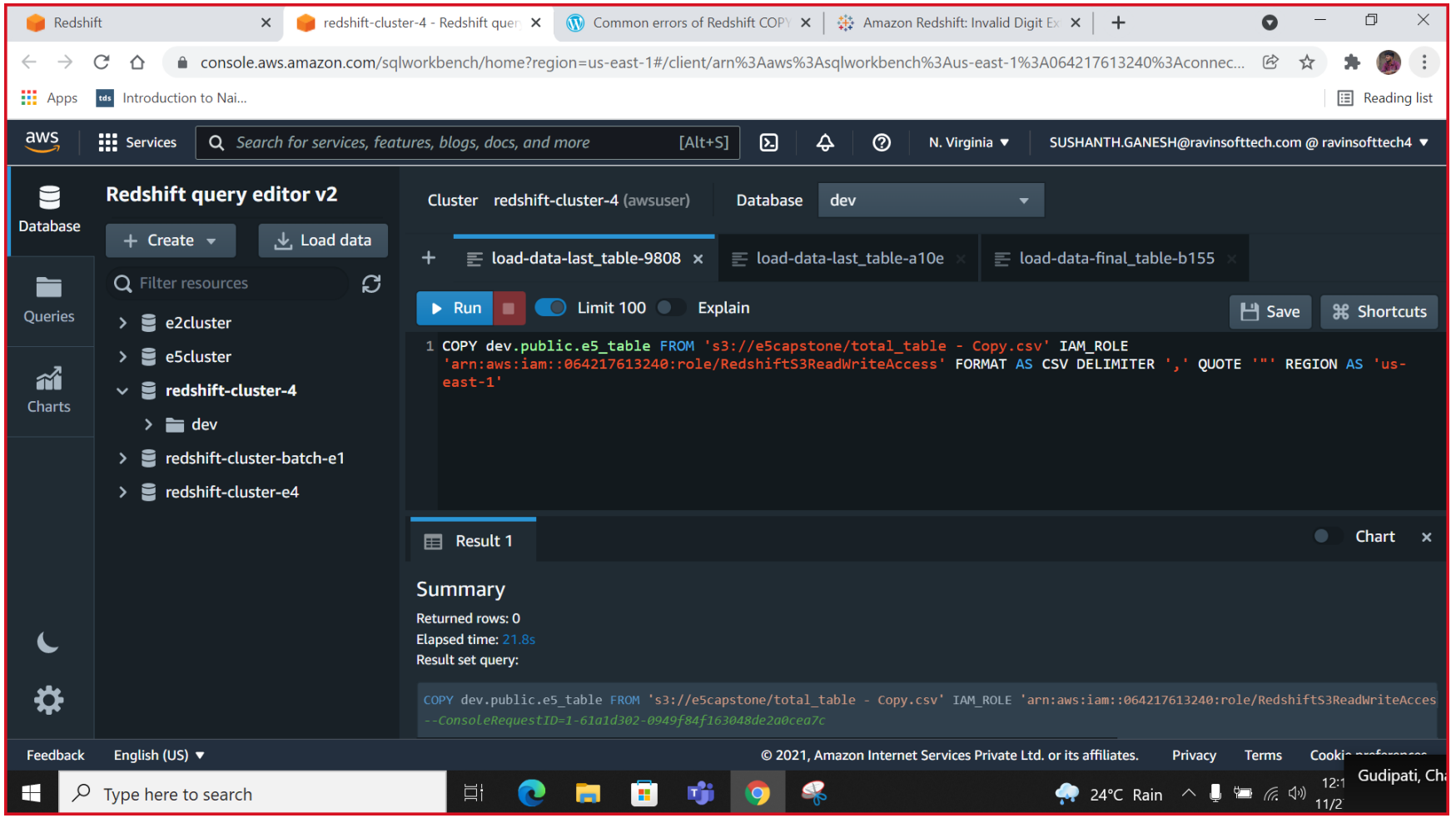


Amazon Redshift

Uploading data to Redshift (for large structured data, requires predefinition of table)

Data Analysis on Cloud (AWS).

Amazon **Redshift**

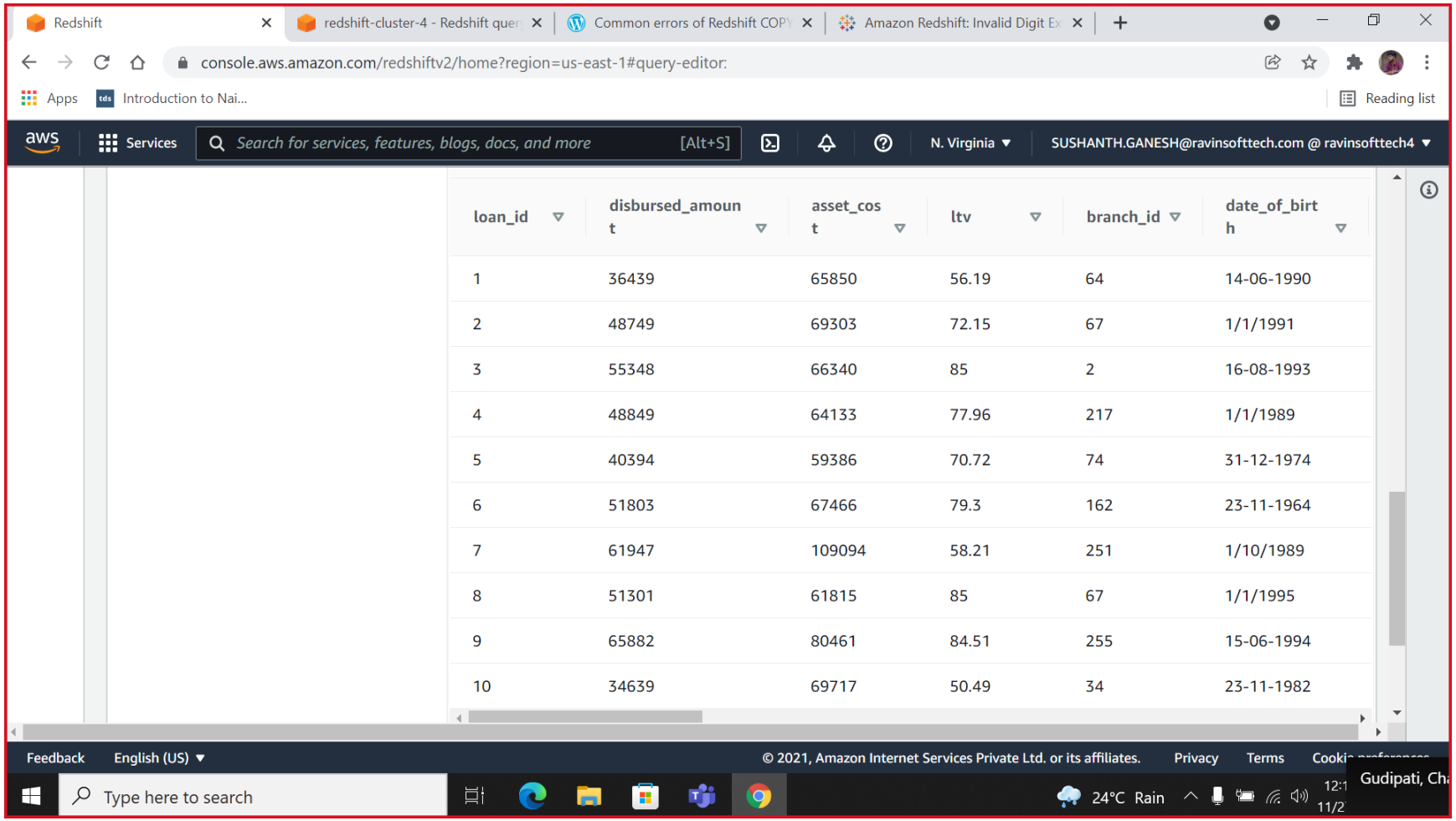


Amazon **Redshift**

Uploading data to Redshift (for large structured data, requires predefinition of table)

Data Analysis on Cloud (AWS).

Amazon **Redshift**



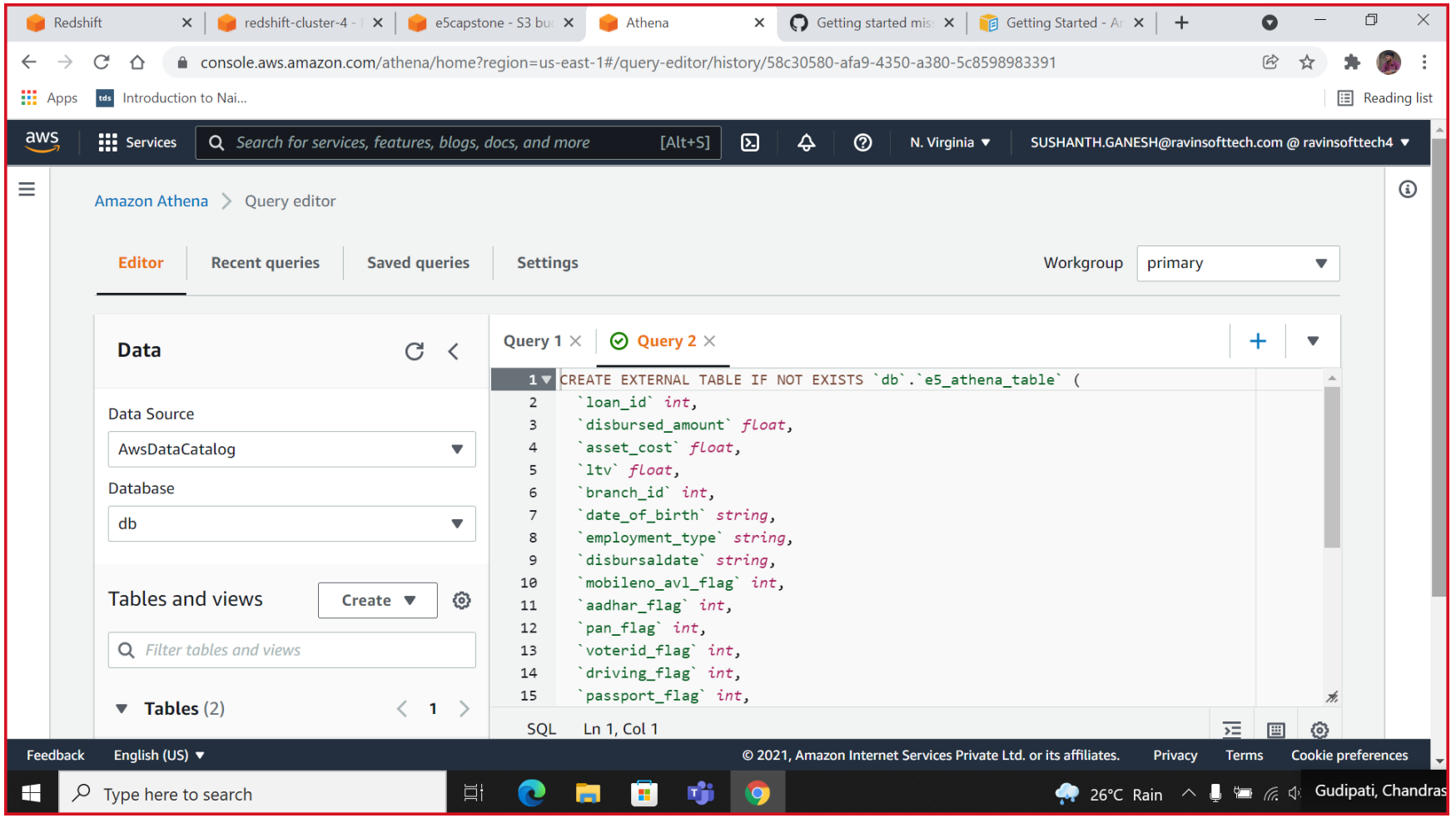
loan_id	disbursed_amount	asset_cost	ltv	branch_id	date_of_birth
1	36439	65850	56.19	64	14-06-1990
2	48749	69303	72.15	67	1/1/1991
3	55348	66340	85	2	16-08-1993
4	48849	64133	77.96	217	1/1/1989
5	40394	59386	70.72	74	31-12-1974
6	51803	67466	79.3	162	23-11-1964
7	61947	109094	58.21	251	1/10/1989
8	51301	61815	85	67	1/1/1995
9	65882	80461	84.51	255	15-06-1994
10	34639	69717	50.49	34	23-11-1982

Amazon **Redshift**

Uploading data to Redshift (for large structured data, requires predefinition of table)

Data Analysis on Cloud (AWS).

Amazon Athena

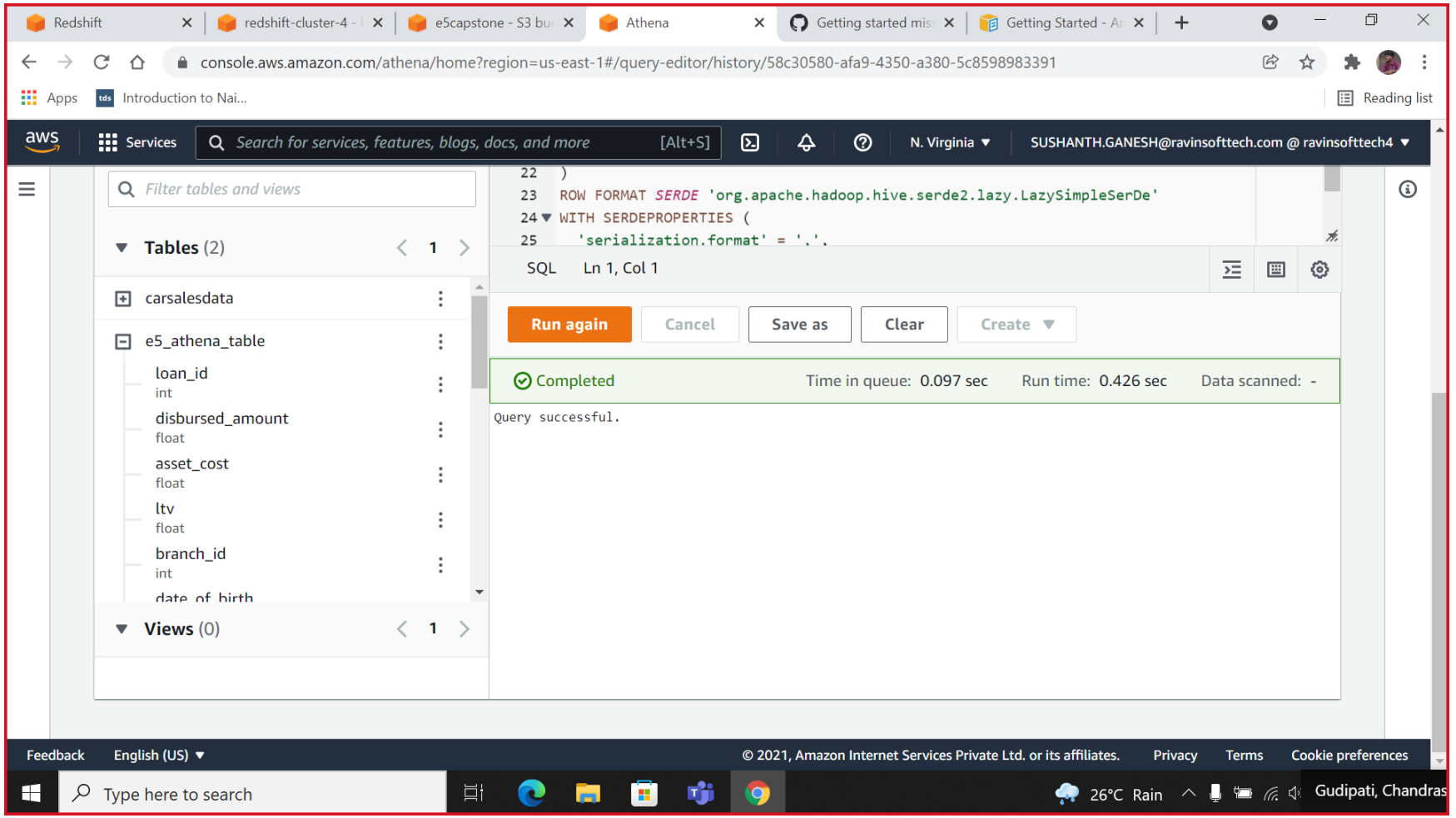


Amazon Athena

Uploading to Athena (serverless, useful even for unstructured and semi structured data, great for PowerBI integration)

Data Analysis on Cloud (AWS).

Amazon Athena

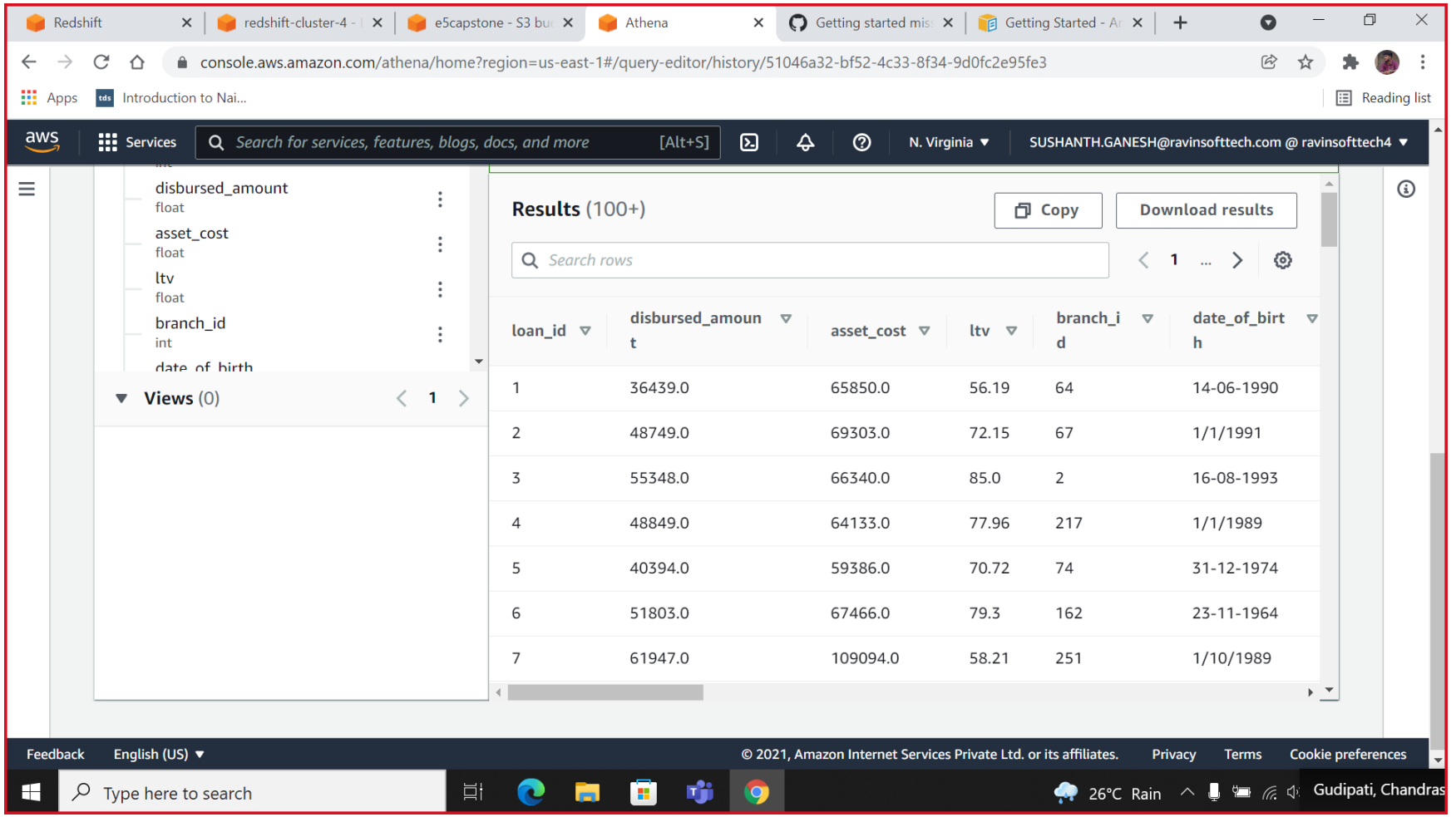


Amazon Athena

Uploading to Athena (serverless, useful even for unstructured and semi structured data, great for PowerBI integration)

Data Analysis on Cloud (AWS).

Amazon Athena

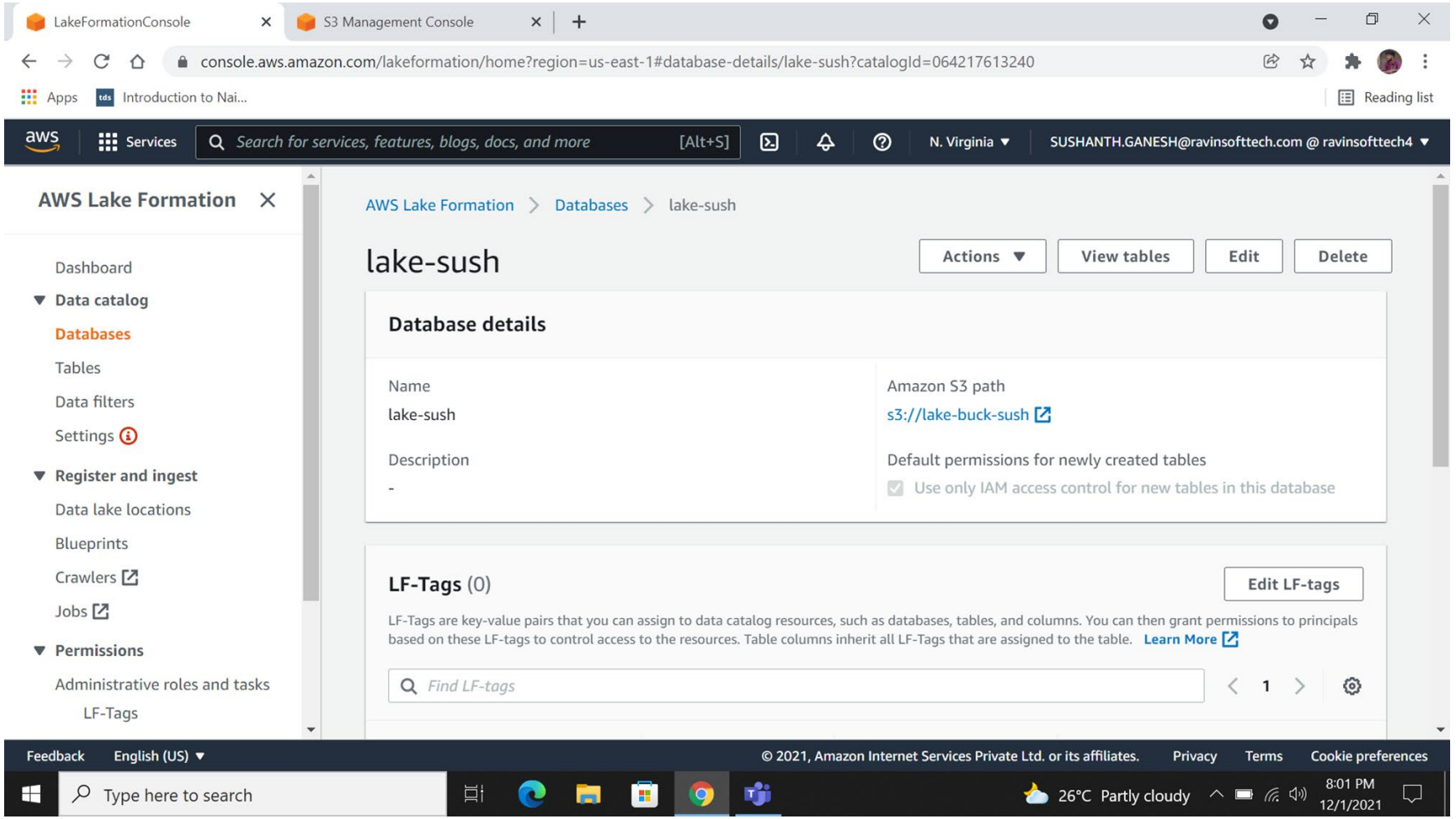


Amazon Athena

Uploading to Athena (serverless, useful even for unstructured and semi structured data, great for PowerBI integration)

Data Analysis on Cloud (AWS).

Datalake

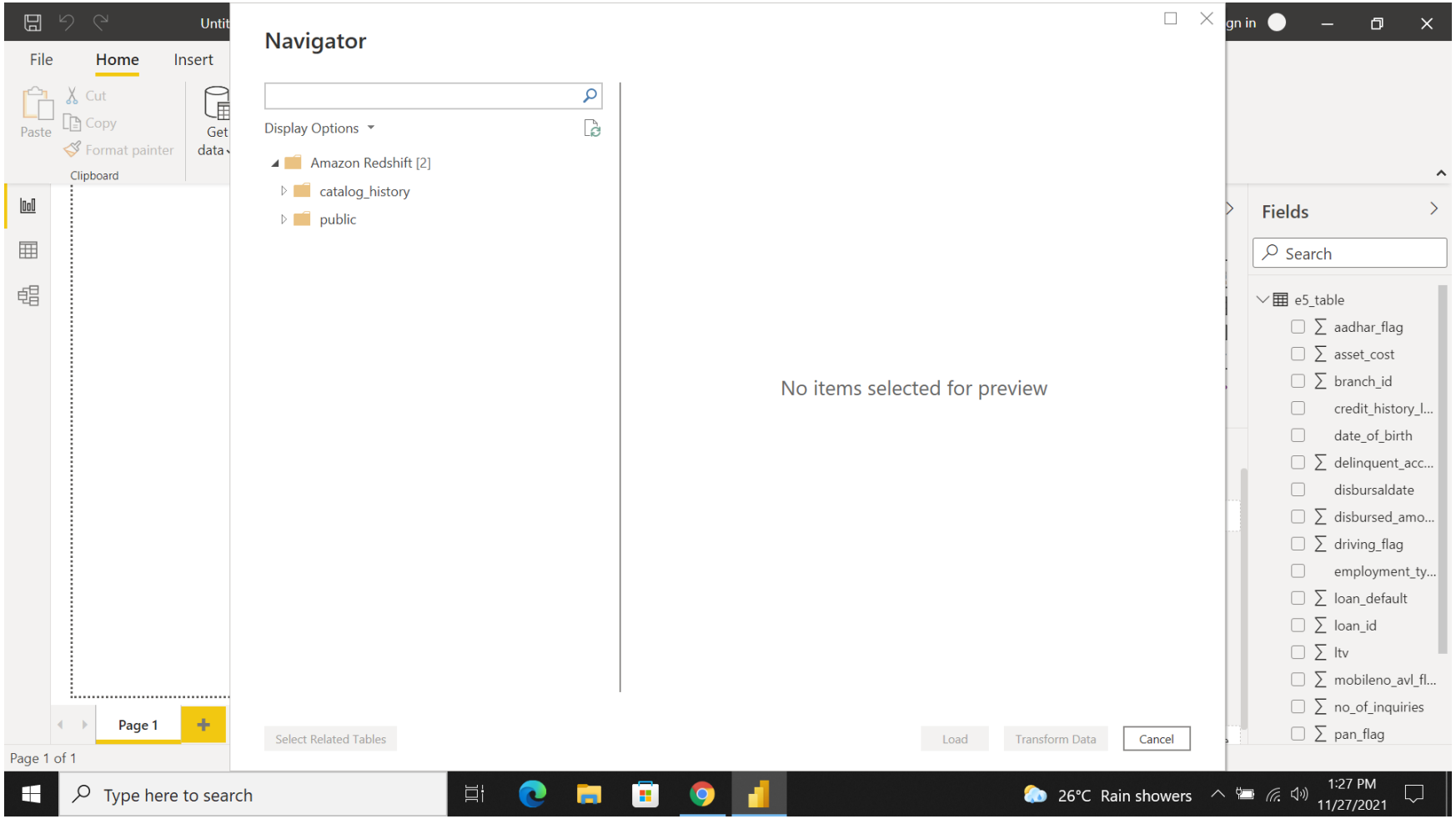


Datalake

Datalake is a centralized repository ready for large amounts of both structured and unstructured data

Data Analysis on Cloud (AWS).

Power BI

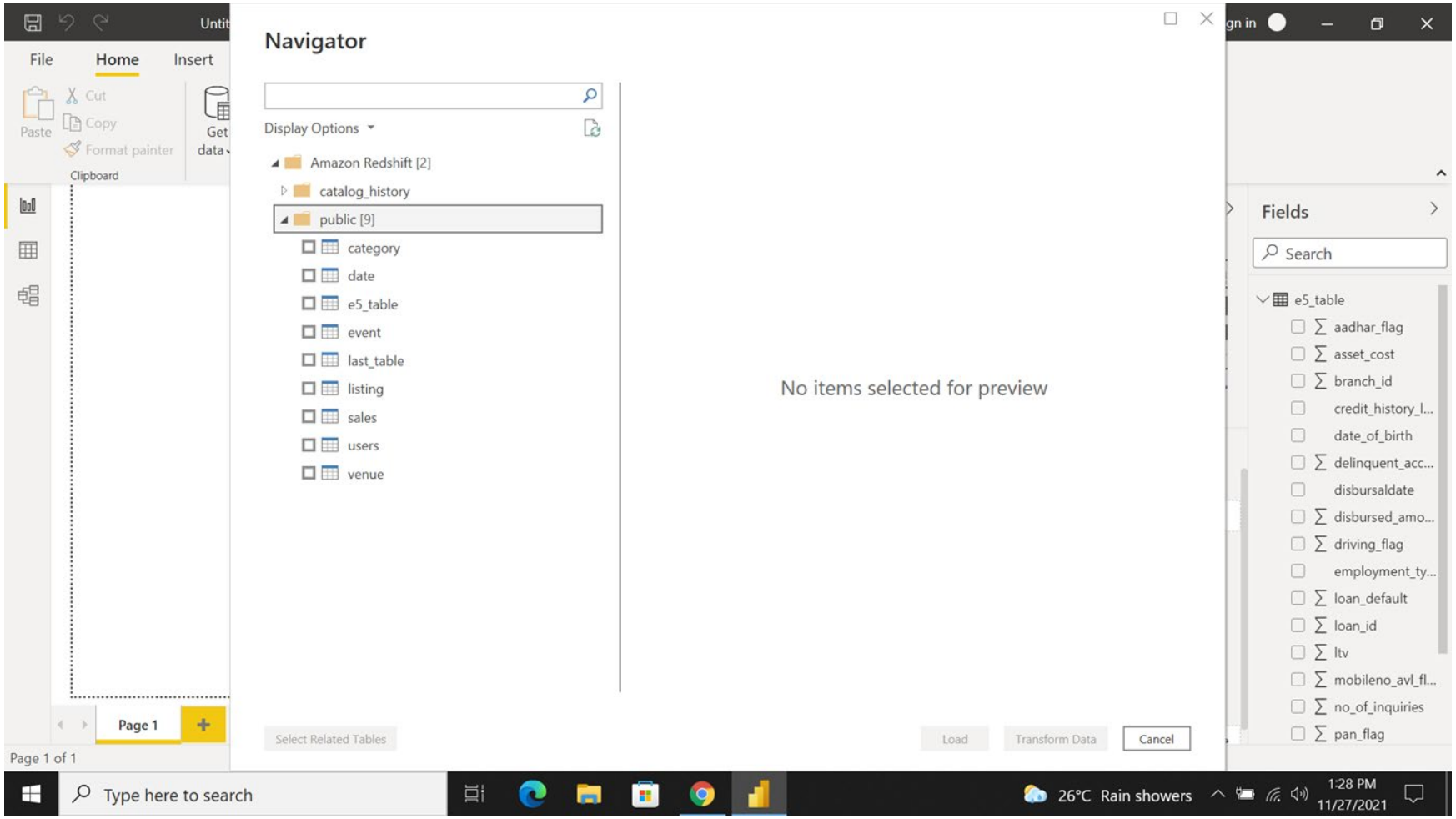


Power BI

Uploading to PowerBI

Data Analysis on Cloud (AWS).

Power BI



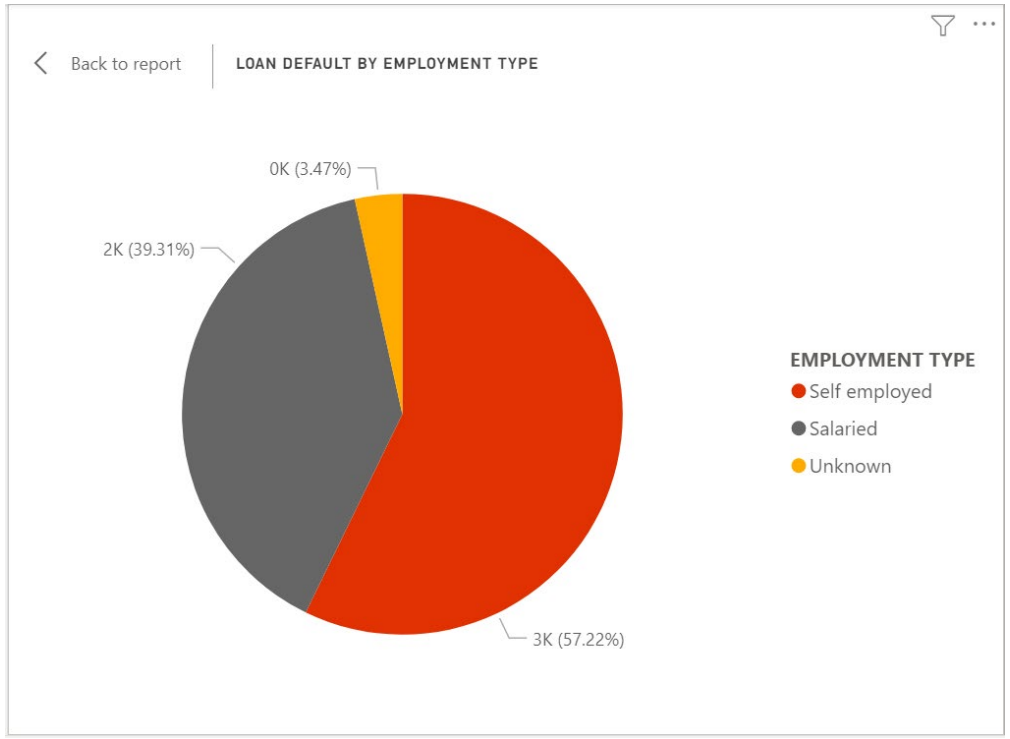
Power BI

Uploading to PowerBI

Data Analysis on Cloud (AWS).

1
2
3

Power BI

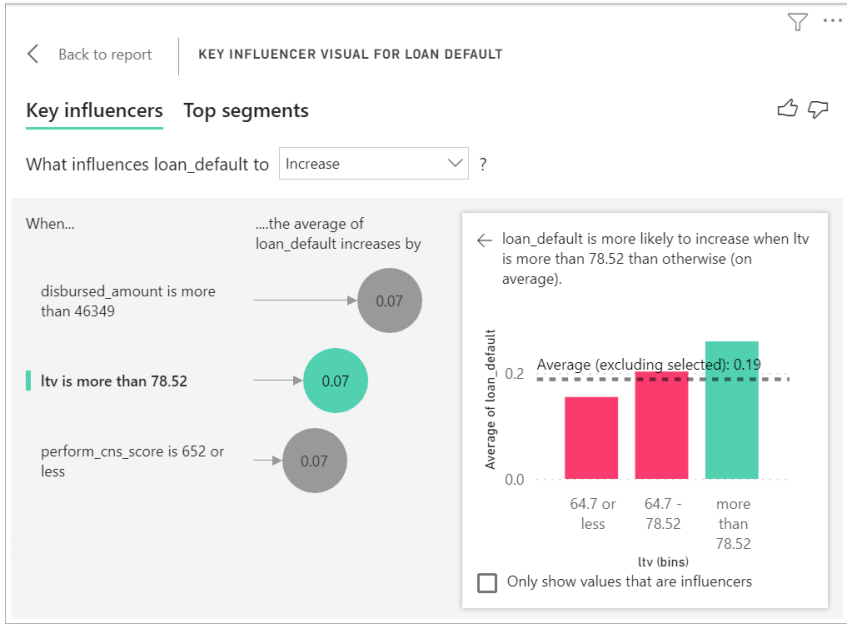


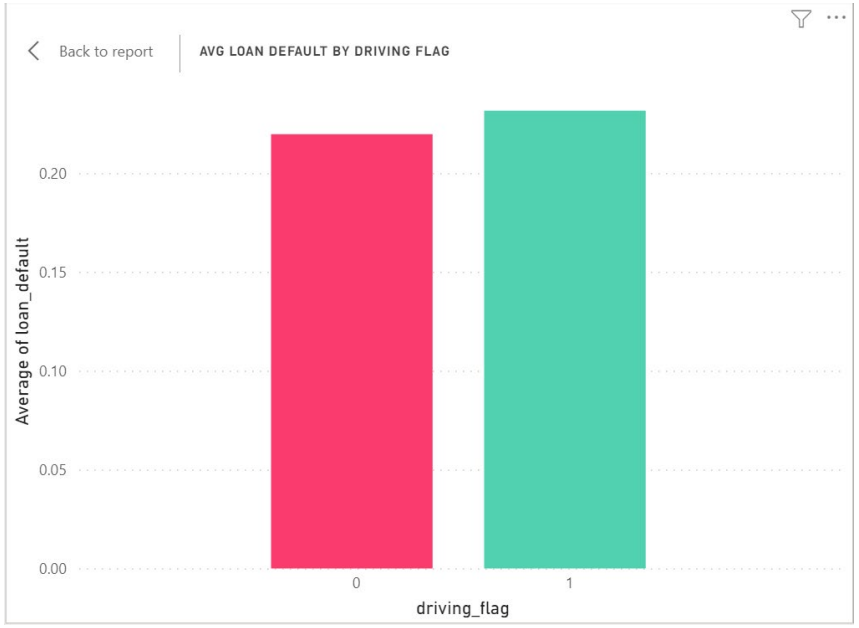
Power BI

Uploading to PowerBI

Data Analysis on Cloud (AWS).

Power BI

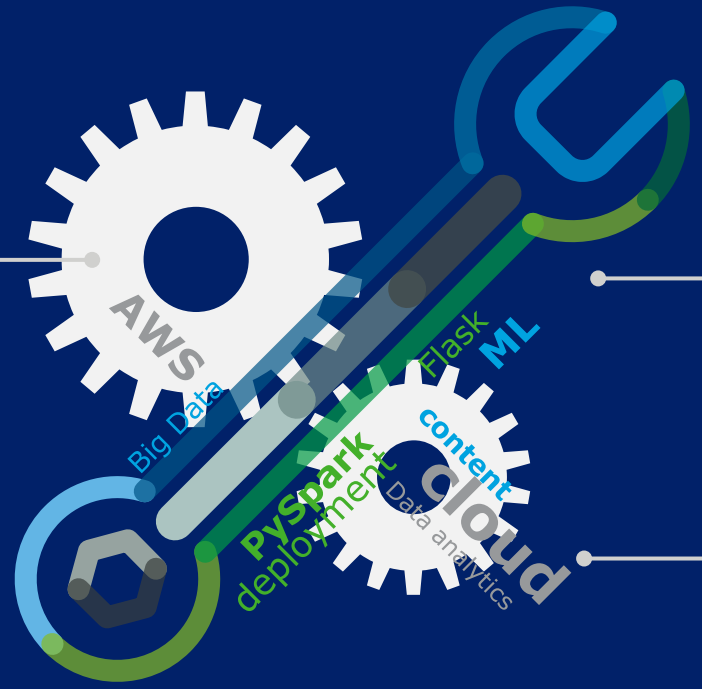




Power BI

Uploading to PowerBI

3.1
Data Analysis using Big Data Tools

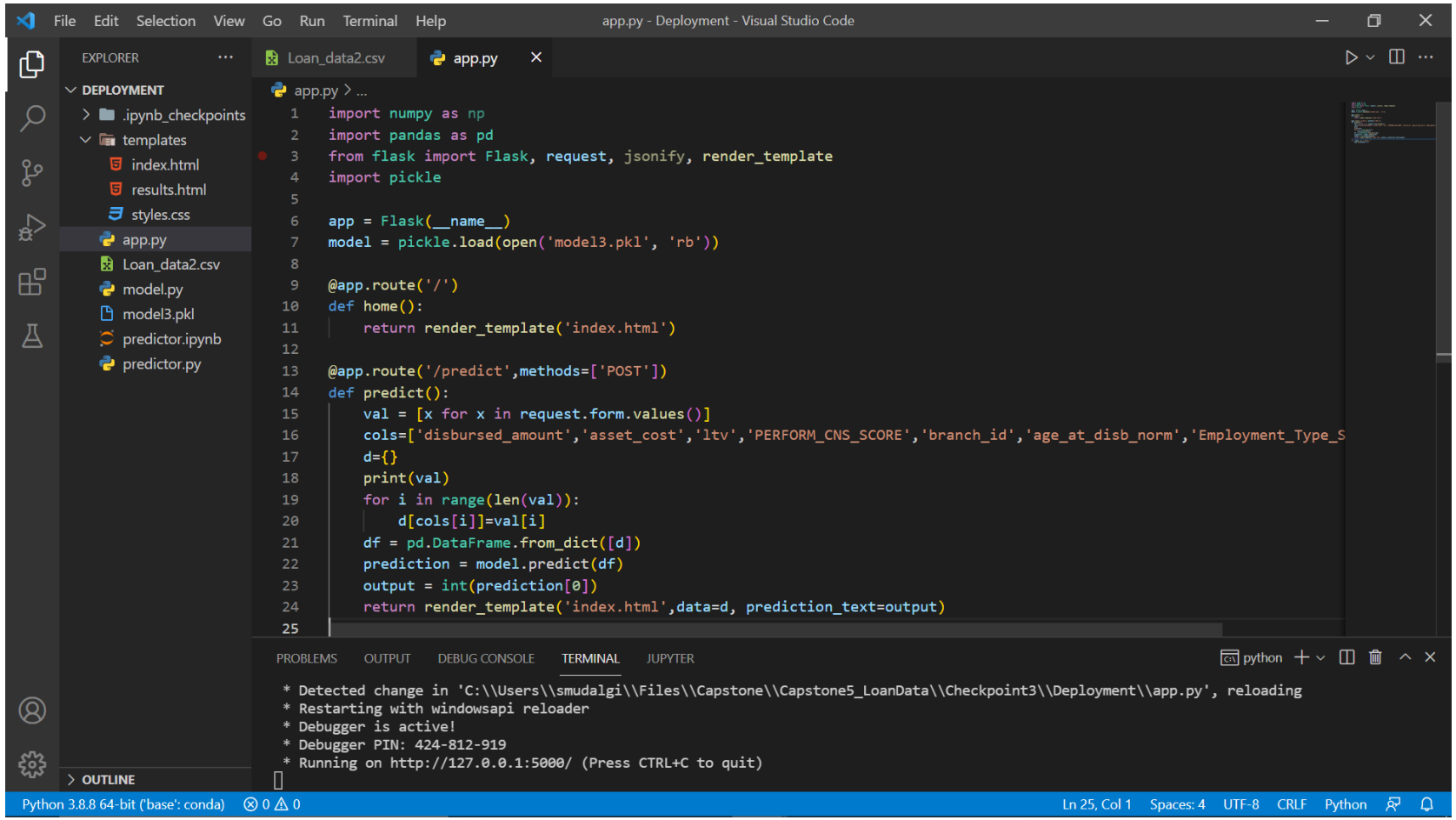


3.2
Data Analysis on Cloud (AWS)

3.3
Deployment of Models using Flask

Task by **The Fantastic Four.** Presentation by **Chandrasekhar Gudipati.**

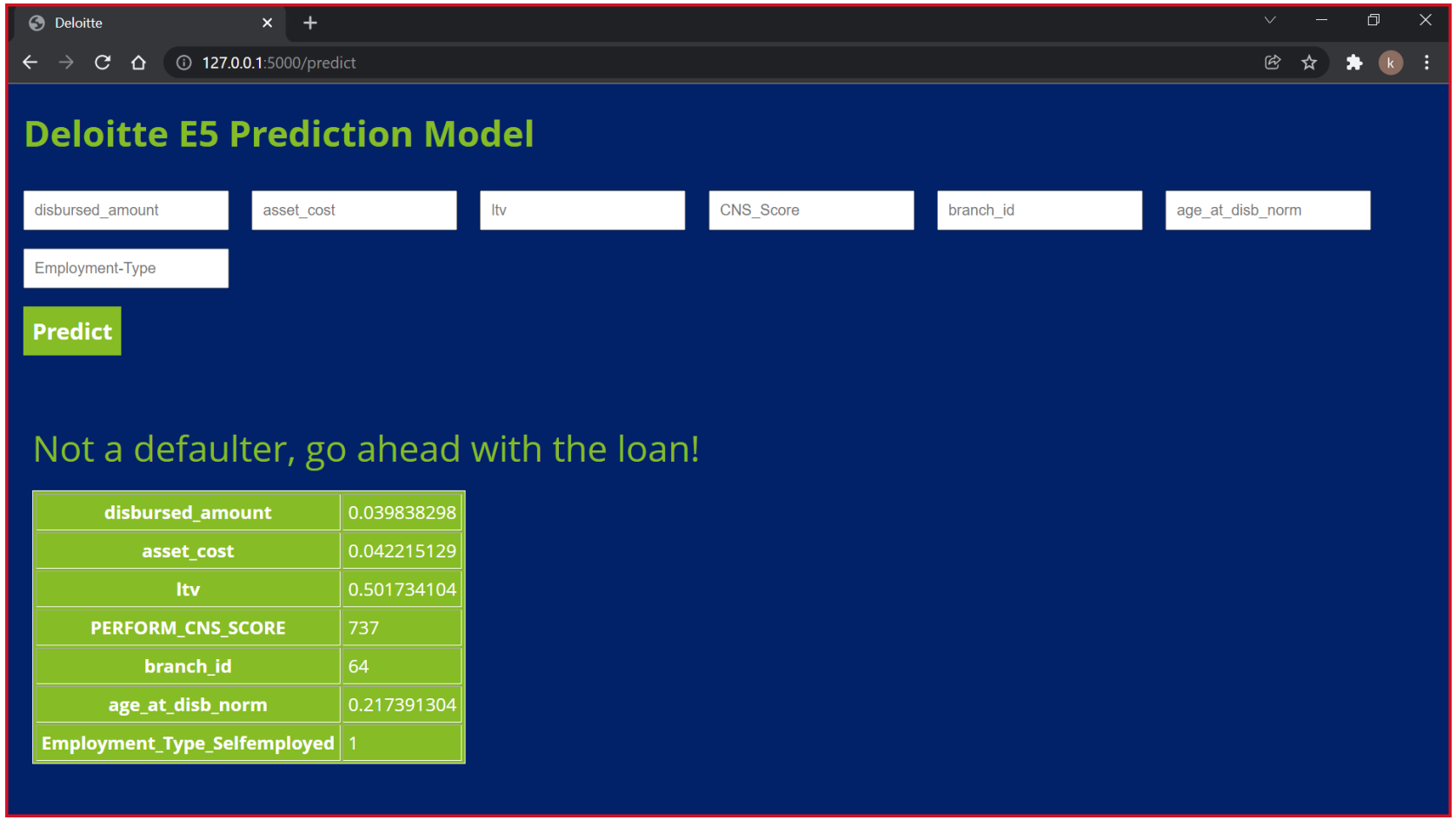
Deployment using Flask.



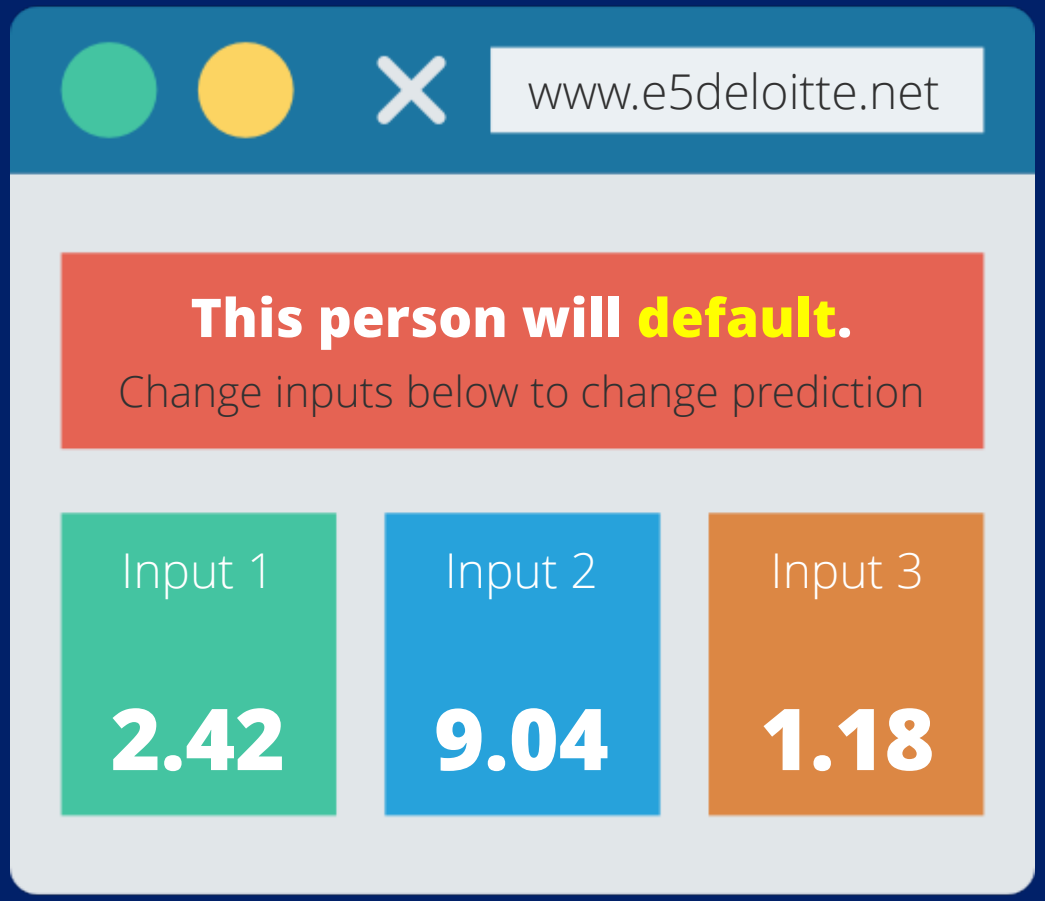
Flask

Flask

Deployment using Flask.



Problem Statement.



Pretty close, huh?

Our recommendations as Deloitte Analysts to the Bank

Always check your customer's loan to value ratio. It seems to always be the biggest player in predicting whether your customer will default. The more the ltv, the more likely the default (check task 2.4).

Use the XGB Model that we are leaving you with (check task 2.4) for loan default predictions. It gives the best results of all our models.

Prefer salaried employees over self-employed people as customers. They are more likely to have a stable income and hence less likely to default (check task 2.3).


1
2
3

The **End.**

Know us better.

Chandrasekhar Gudipati	Shruti Maigur	Sushant Mudalgi	Sushanth Ganesh

Any Questions?

Deloitte.

Thank **You**