NALAIYA THIRAN

PROJECT REPORT WEEK 3 REPORT

Project Title: **Fertilizers Recommendation System for Disease Prediction**

Mentor Name: Malathy EM

Team ID: PNT2022TMID53297

Team Size: 4

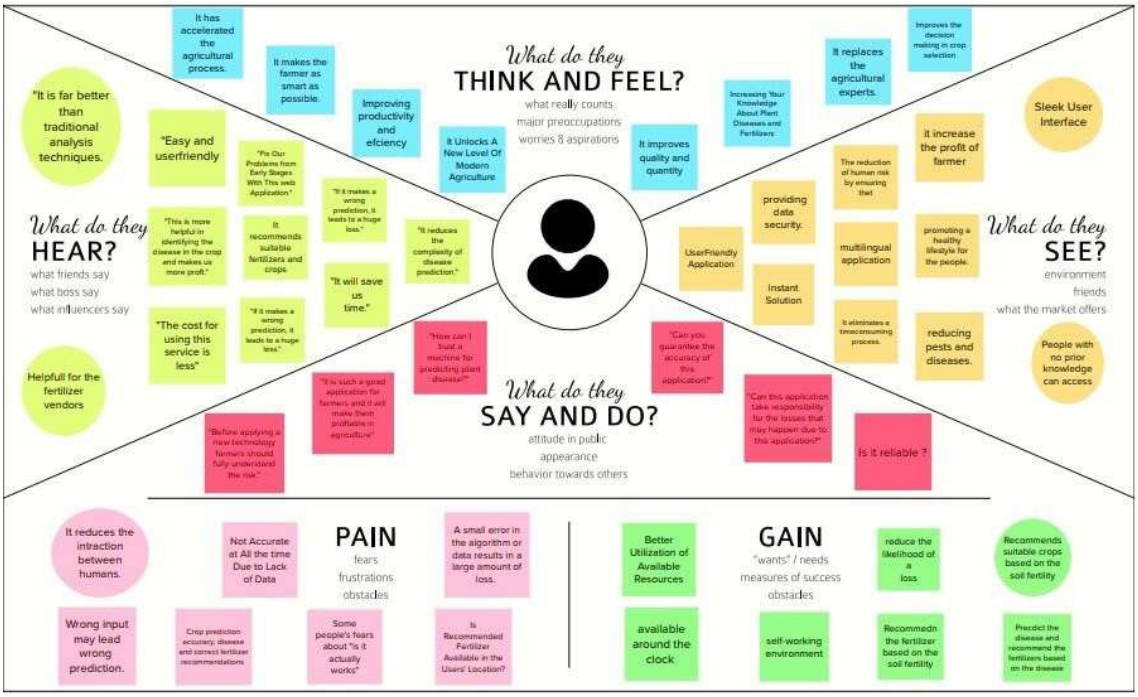
Team Leader: Santhosh M

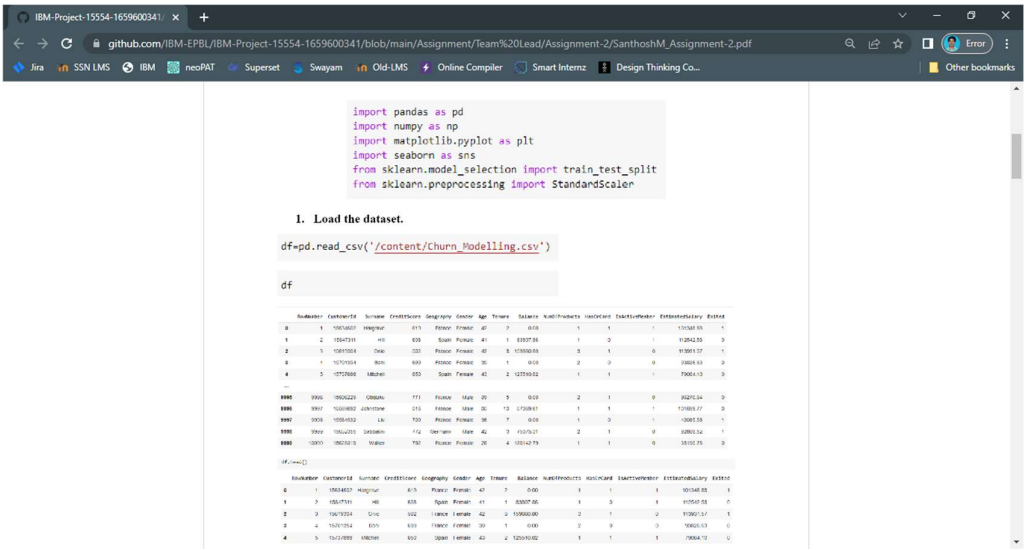
Team member-1: Iswarya M

Team member-2: Sabarivasan E

Team member-3: Navaneethakrishnan S

PHASE-III DESCRIPTION: EMPATHY MAP



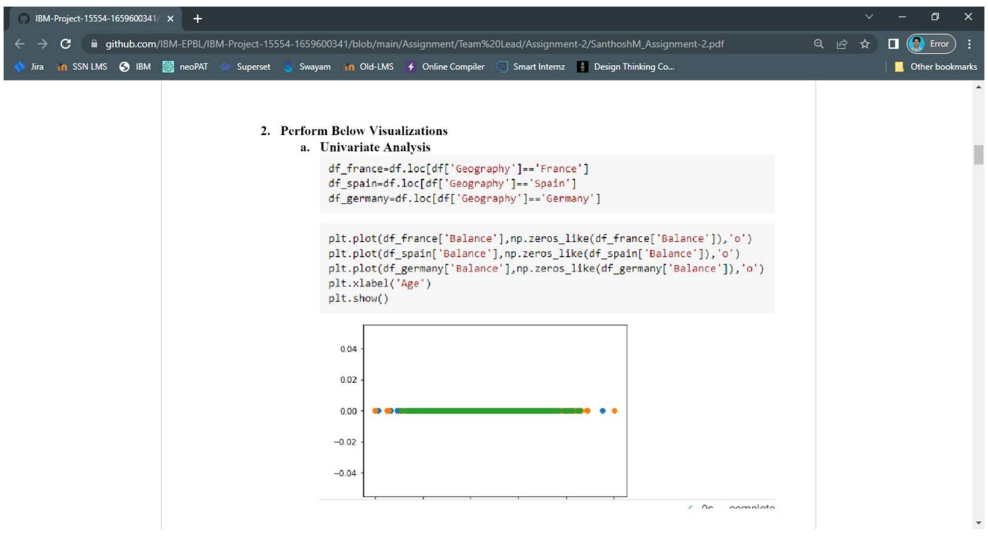


```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

1. **Load the dataset.**

```
df=pd.read_csv('/content/Churn_Modelling.csv')
```

```
df
```

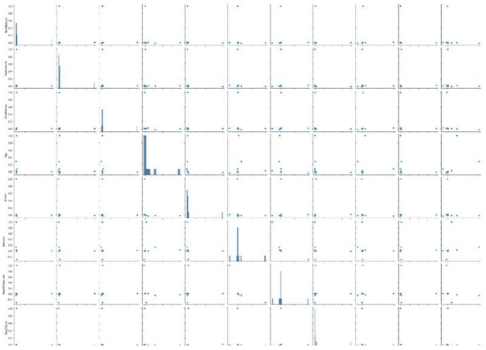
2. **Perform Below Visualizations**
   a. **Univariate Analysis**

```
df_france=df.loc[df['Geography']=='France']
df_spain=df.loc[df['Geography']=='Spain']
df_germany=df.loc[df['Geography']=='Germany']
```

```
plt.plot(df_france['Balance'],np.zeros_like(df_france['Balance']),'o')
plt.plot(df_spain['Balance'],np.zeros_like(df_spain['Balance']),'o')
plt.plot(df_germany['Balance'],np.zeros_like(df_germany['Balance']),'o')
plt.xlabel('Age')
plt.show()
```

3. Multivariate Analysis

```
sns.pairplot(df.corr())
```



4. Perform descriptive statistics on the dataset

```
df.describe(include='all')
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
count	10000.00000	1.000000e+04	10000	10000.000000	10000	10000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
unique	NaN	NaN	2932	NaN	3	2	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	Smith	NaN	France	Male	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	32	NaN	5014	5457	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	5000.50000	1.569094e+07	NaN	650.528800	NaN	NaN	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881
std	2886.89568	7.193619e+04	NaN	96.653299	NaN	NaN	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818
min	1.00000	1.556570e+07	NaN	350.000000	NaN	NaN	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000
25%	2500.75000	1.562853e+07	NaN	584.000000	NaN	NaN	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000
50%	5000.50000	1.569074e+07	NaN	652.000000	NaN	NaN	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000
75%	7500.25000	1.575323e+07	NaN	718.000000	NaN	NaN	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500
max	10000.00000	1.581569e+07	NaN	850.000000	NaN	NaN	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000

5. Handle the Missing values

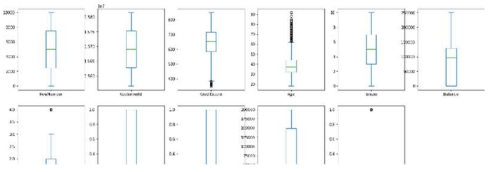
```
df.isnull()
```

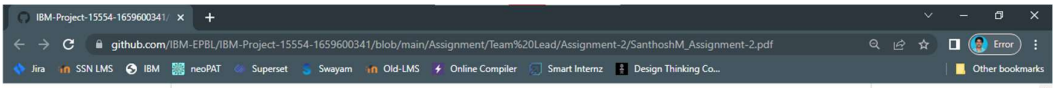
	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False
...														
9995	False	False	False	False	False	False	False	False	False	False	False	False	False	False
9996	False	False	False	False	False	False	False	False	False	False	False	False	False	False
9997	False	False	False	False	False	False	False	False	False	False	False	False	False	False
9998	False	False	False	False	False	False	False	False	False	False	False	False	False	False
9999	False	False	False	False	False	False	False	False	False	False	False	False	False	False

6. Find the outliers and replace the outliers

```
df.plot(subplots=True,layout=(4,6),kind='box',figsize=(22,18))
```

```
RowNumber            AxesSubplot(0.125,0.71587;0.110714x0.16413)
CustomerId         AxesSubplot(0.257857,0.71587;0.110714x0.16413)
CreditScore        AxesSubplot(0.390714,0.71587;0.110714x0.16413)
Age                AxesSubplot(0.523571,0.71587;0.110714x0.16413)
Tenure             AxesSubplot(0.656429,0.71587;0.110714x0.16413)
Balance            AxesSubplot(0.789286,0.71587;0.110714x0.16413)
NumOfProducts        AxesSubplot(0.125,0.518913;0.110714x0.16413)
HasCrCard          AxesSubplot(0.257857,0.518913;0.110714x0.16413)
IsActiveMember     AxesSubplot(0.390714,0.518913;0.110714x0.16413)
EstimatedSalary    AxesSubplot(0.523571,0.518913;0.110714x0.16413)
Exited             AxesSubplot(0.656429,0.518913;0.110714x0.16413)
dtype: object
```

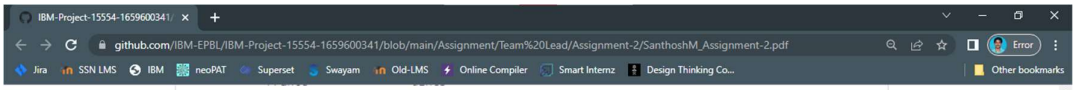




7. Check for Categorical columns and perform encoding

```
df.dtypes
```

```
RowNumber            int64
CustomerId           int64
Surname             object
CreditScore          int64
Geography           object
Gender              object
Age                  int64
Tenure               int64
Balance            float64
NumOfProducts        int64
HasCrCard            int64
IsActiveMember       int64
EstimatedSalary    float64
Exited               int64
dtype: object
```




```
Germany              uint8
Spain                uint8
Female               uint8
Male                 uint8
dtype: object
```

```
df.drop(["Surname"],axis=1,inplace=True)
```

8. Split the data into dependent and independent variables

```
dependent_data=df['Exited']
```

```
independent_data=df.drop('Exited',axis=1)
```

9. Scale the independent variables

```
#minmax scaler
df_norm = (independent_data-independent_data.min())/(independent_data.max()-independent_data.min())
```

```
df_norm.head()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	France	Germany	Spain	Female	Male
58	0.060060	0.229986	0.213553	0.615148	0.4	0.000000	0.000000	1.0	0.0	0.005731	0.0	0.0	1.0	1.0	0.0
86	0.062732	0.166305	0.552206	0.417730	1.0	0.000000	0.333333	1.0	1.0	0.873761	0.0	0.0	1.0	1.0	0.0
104	0.064657	0.166369	0.552760	0.069966	0.1	0.000000	0.000000	1.0	1.0	0.892208	0.0	0.0	1.0	1.0	0.0
158	0.070124	0.295542	0.503452	0.244828	0.6	0.620066	0.000000	0.0	1.0	0.823726	1.0	0.0	0.0	1.0	0.0
181	0.072102	0.166726	0.220789	0.069966	0.2	0.000000	0.333333	1.0	1.0	0.230051	1.0	0.0	0.0	0.0	1.0

10. Split the data into training and testing

```
X_train, X_test, y_train, y_test = train_test_split( independent_data, dependent_data, test_size=0.30, random_state=42)
```