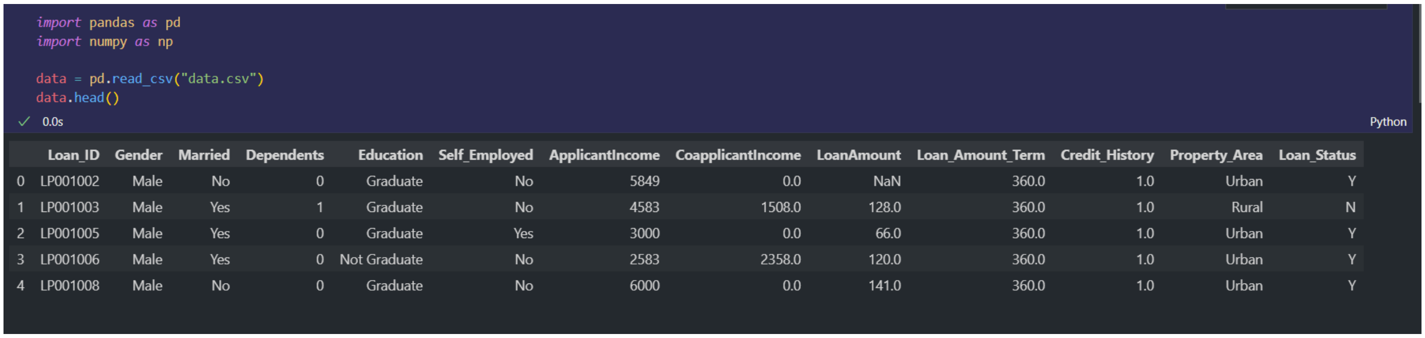
Machine Learning Assignment_2 [Find_S_Algorithm]
Name: Arya Chakraborty
Reg.No: 22MSD7020
Date: 2/23/2023

Question: **Find the most specific hypothesis for the loan prediction dataset (Kaggle) using Find-S Algorithm.**

Ans:

First, we load the required libraries & the dataset on the local machine.

Code & output:



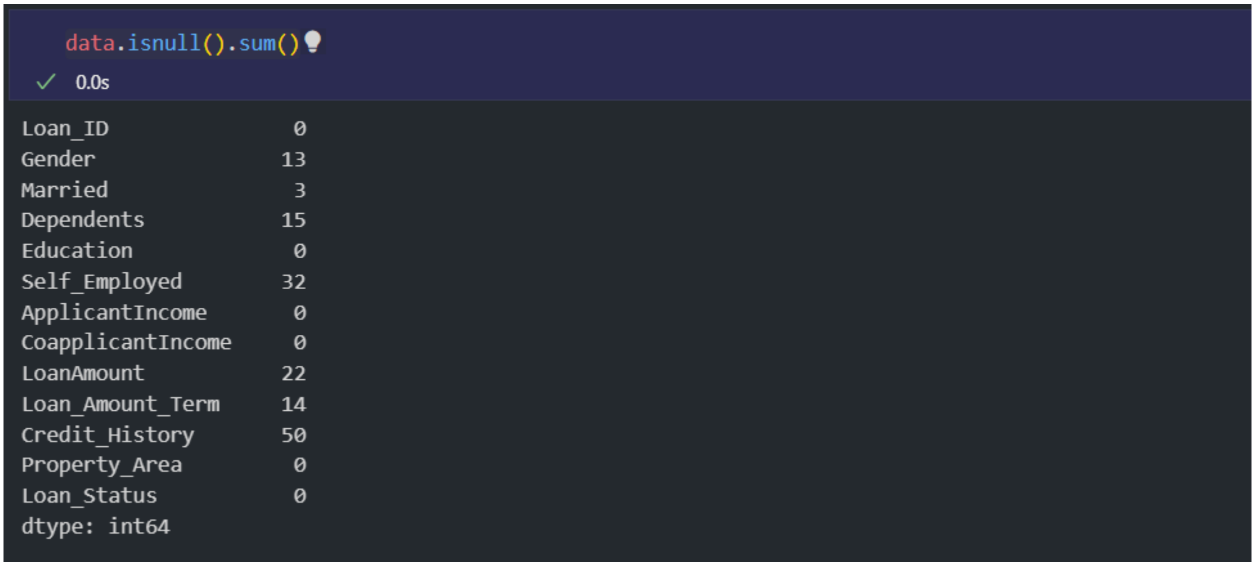
```
import pandas as pd
import numpy as np

data = pd.read_csv("data.csv")
data.head()
```

✓ 0.0s

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

Now we check for null values in the dataset before going further into the Algorithm.

Code & Output:



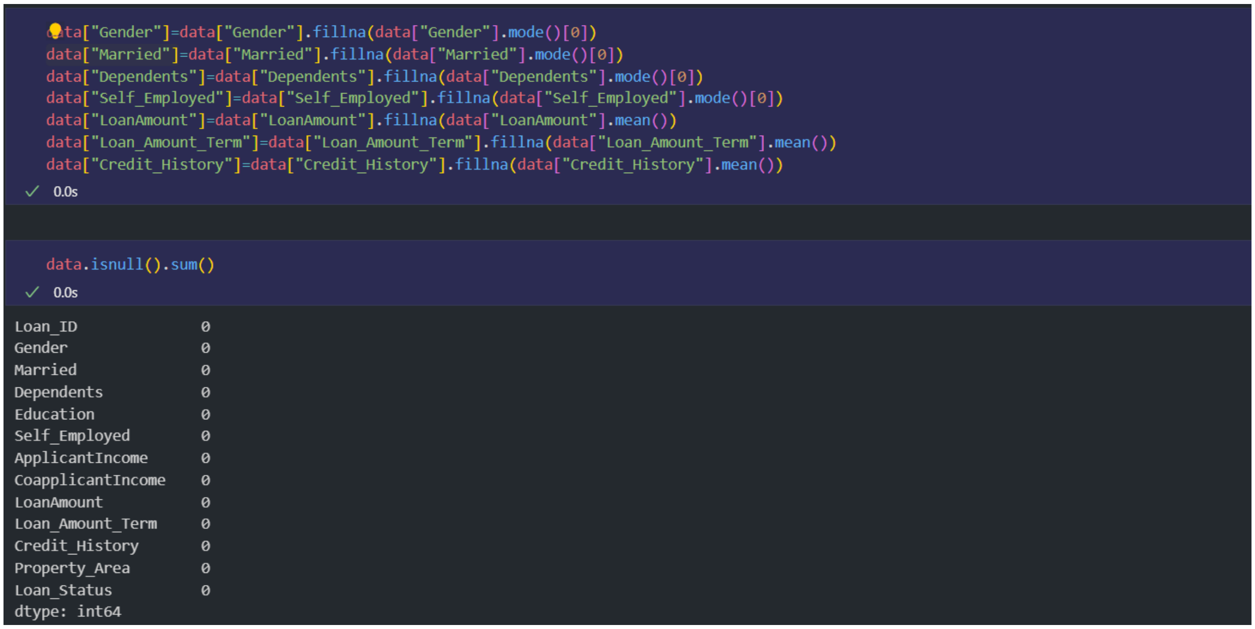
```
data.isnull().sum()
```

✓ 0.0s

```
Loan_ID               0
Gender               13
Married               3
Dependents           15
Education             0
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

Here we can see there are several columns with null values. Let's fill those empty cells according to the data types.

<u>Code & Output</u>:



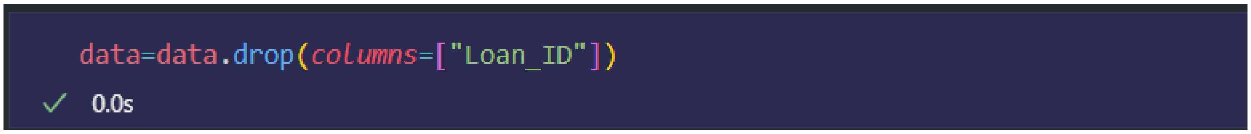
```
data["Gender"]=data["Gender"].fillna(data["Gender"].mode()[0])
data["Married"]=data["Married"].fillna(data["Married"].mode()[0])
data["Dependents"]=data["Dependents"].fillna(data["Dependents"].mode()[0])
data["Self_Employed"]=data["Self_Employed"].fillna(data["Self_Employed"].mode()[0])
data["LoanAmount"]=data["LoanAmount"].fillna(data["LoanAmount"].mean())
data["Loan_Amount_Term"]=data["Loan_Amount_Term"].fillna(data["Loan_Amount_Term"].mean())
data["Credit_History"]=data["Credit_History"].fillna(data["Credit_History"].mean())
✓ 0.0s
```

```
data.isnull().sum()
✓ 0.0s
```

```
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
Loan_Status          0
dtype: int64
```

Now we have no null values. We may drop the "Loan_ID" Column as it is not required for "Loan_Status" prediction.

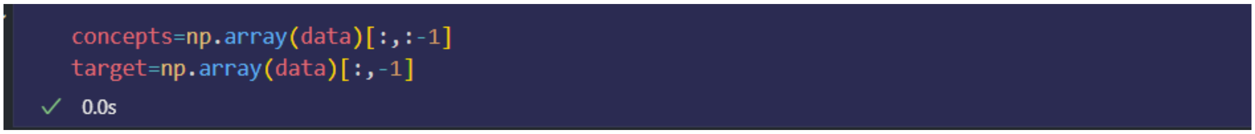
<u>Code & Output:</u>



```
data=data.drop(columns=["Loan_ID"])
✓ 0.0s
```

Now we divide the dataset into "Concepts" & "Target".

<u>Code & Output</u>:



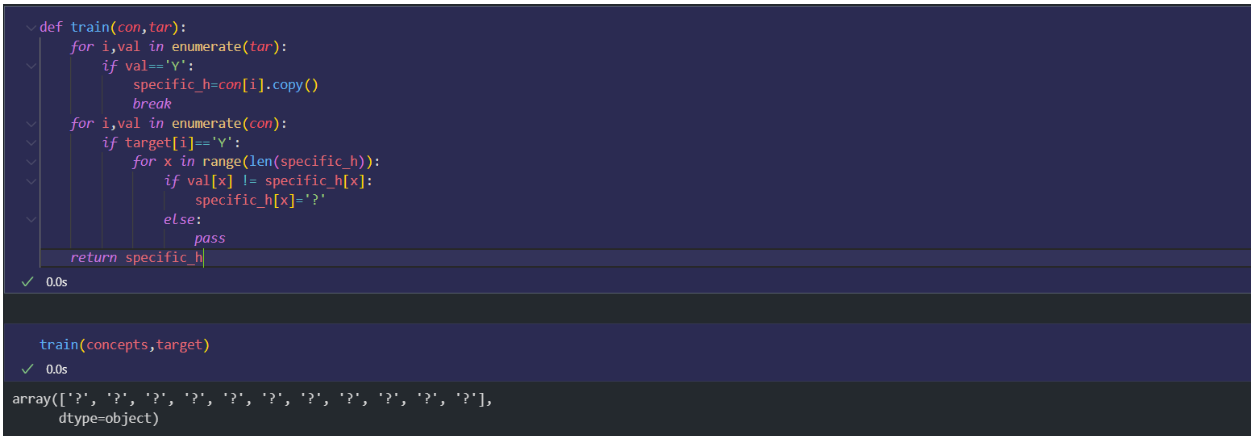
```
concepts=np.array(data)[:,:-1]
target=np.array(data)[:,-1]
✓ 0.0s
```

Algorithm for "Find_S_Algorithm":

1. We initialize 'h' with the most specific hypothesis, which is the first positive example in the dataset.

2. We check for each positive example. If the example is negative, we will move on to the next example but if it is a positive example we will consider it for the next step.
3. We will check if each attribute in the example is equal to the hypothesis value.
4. If the value matches, then no changes are made and If the value does not match, the value is changed to ‘?’.
5. We do this until we reach the last positive example in the data set.

Code & Output:



```
def train(con,tar):
    for i,val in enumerate(tar):
        if val=='Y':
            specific_h=con[i].copy()
            break
    for i,val in enumerate(con):
        if target[i]=='Y':
            for x in range(len(specific_h)):
                if val[x] != specific_h[x]:
                    specific_h[x]='?'
                else:
                    pass
    return specific_h
```

✓ 0.0s

```
train(concepts,target)
```

✓ 0.0s

```
array(['?', '?', '?', '?', '?', '?', '?', '?', '?', '?', '?'],
      dtype=object)
```

Result: Here we can see that in the resultant hypothesis, all are general hypothesis. We use this hypothesis to test and predict the “Loan_Status”.