In [ ]:

```
'''
Feature engineering is the pre-processing step of ML
which transforms raw df into features (input) to our model

Feature engineering means creating new features from existing one's

e.g

from fullname we can create firstname,mniddlename and lastname

from marks1 and marks2 we can create total marks by taking there sum.


Outlier managenent means handling the df which is an outsider compared to others

e.g

realworld example will be

0.Red ball in group of blue

1.Driving a car on footpath or on wrong side

2.A foreigner from outside of India


programming example

1.Stack Overflow

2.ArrayIndexOutOfBounds

'''
```

In [1]:

```
# import pandas library and dfset

import pandas as pd

filename = 'p5_feature_engineering.csv'

df = pd.read_csv(filename)
df
```

Out[1]:

	rno	firstname	middlename	lastname	marks1	marks2
0	25	shivam	bhimling	limbhare	60	60
1	33	sankalp	santosh	oswal	60	60
2	35	umarkhan	zaheerkhan	pathan	50	50
3	43	mahesh	ramesh	patil	60	60
4	44	manjit	ganesh	patil	60	60
5	56	bhavesh	satish	shete	60	60
6	100	NAN	NaN	NO	100	10
7	110	fname	mname	lname	100	100

In [2]:

```
# creating new columns from existing columns

df['total_marks'] = df['marks1'] + df['marks2']

df
```

Out[2]:

	rno	firstname	middlename	lastname	marks1	marks2	total_marks
0	25	shivam	bhimling	limbhare	60	60	120
1	33	sankalp	santosh	oswal	60	60	120
2	35	umarkhan	zaheerkhan	pathan	50	50	100
3	43	mahesh	ramesh	patil	60	60	120
4	44	manjit	ganesh	patil	60	60	120
5	56	bhavesh	satish	shete	60	60	120
6	100	NAN	NaN	NO	100	10	110
7	110	fname	mname	lname	100	100	200

In [3]:

```
# remove unwanted columns from dfset
df = df.drop(columns=['marks1','marks2'])

df.head(10) # to print only first 10 rows
```

Out[3]:

	rno	firstname	middlename	lastname	total_marks
0	25	shivam	bhimling	limbhare	120
1	33	sankalp	santosh	oswal	120
2	35	umarkhan	zaheerkhan	pathan	100
3	43	mahesh	ramesh	patil	120
4	44	manjit	ganesh	patil	120
5	56	bhavesh	satish	shete	120
6	100	NAN	NaN	NO	110
7	110	fname	mname	lname	200

In [4]:

```
# students fullname from given details

df['fullname'] = df['firstname'] + df['middlename'] + df['lastname']

df
```

Out[4]:

	rno	firstname	middlename	lastname	total_marks	fullname
0	25	shivam	bhimling	limbhare	120	shivambhimlinglimbhare
1	33	sankalp	santosh	oswal	120	sankalpsantoshoswal
2	35	umarkhan	zaheerkhan	pathan	100	umarkhanzaheerkhanpathan
3	43	mahesh	ramesh	patil	120	maheshrameshpatil
4	44	manjit	ganesh	patil	120	manjitganeshpatil
5	56	bhavesh	satish	shete	120	bhaveshsatishshete
6	100	NAN	NaN	NO	110	NaN
7	110	fname	mname	lname	200	fnamemnamelname

In [5]:

```
df.info() # dfframe information columns name and there dftypes
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   rno          8 non-null      int64 
 1   firstname    8 non-null      object
 2   middlename   7 non-null      object
 3   lastname     8 non-null      object
 4   total_marks  8 non-null      int64 
 5   fullname     7 non-null      object
dtypes: int64(2), object(4)
memory usage: 512.0+ bytes
```

In [6]:

```
# Outlier management

# here rno for div A is from 1 to 64
first = 1
last = 64

print(df['rno'].isin([i for i in range(first,last)]))
```

```
0     True
1     True
2     True
3     True
4     True
5     True
6    False
7    False
Name: rno, dtype: bool
```

In [7]:

```
# drop the outliers from dfframe


df = df.drop(index=7) # remove outlier by index number
df = df.dropna()      # remove null values

df
```

Out[7]:

	rno	firstname	middlename	lastname	total_marks	fullname
0	25	shivam	bhimling	limbhare	120	shivambhimlinglimbhare
1	33	sankalp	santosh	oswal	120	sankalpsantoshoswal
2	35	umarkhan	zaheerkhan	pathan	100	umarkhanzaheerkhanpathan
3	43	mahesh	ramesh	patil	120	maheshrameshpatil
4	44	manjit	ganesh	patil	120	manjitganeshpatil
5	56	bhavesh	satish	shete	120	bhaveshsatishshete

In [ ]:

```
'''
What is One Hot Encoding?

A one hot encoding is a representation of categorical variables as binary vectors.

This first requires that the categorical values be mapped to integer values.

Then, each integer value is represented as a binary vector that is all zero values
except the index of the integer, which is marked with a 1.

i.e.
if we have n values
a index value  have assigned 1
and n-1 values have assigned 0

'''
```

In [8]:

```
print(df['rno'].unique()) # check for unique value
print(df['rno'].value_counts().sum()) # total no of rno
```

```
[25 33 35 43 44 56]
6
```

In [ ]:

```
# One Hot Encoding
```

In [9]:

```
one_hot_encoded_df = pd.get_dummies(df, columns = ['rno'])
print(one_hot_encoded_df)

from sklearn.preprocessing import OneHotEncoder
enc = OneHotEncoder()
enc
```

```
  firstname  middlename  lastname  total_marks                 fullname  \
0    shivam    bhimling  limbhare          120    shivambhimlinglimbhare
1   sankalp     santosh     oswal          120       sankalpsantoshoswal
2  umarkhan  zaheerkhan    pathan          100  umarkhanzaheerkhanpathan
3    mahesh      ramesh     patil          120         maheshrameshpatil
4    manjit      ganesh     patil          120         manjitganeshpatil
5   bhavesh      satish     shete          120        bhaveshsatishshete

   rno_25  rno_33  rno_35  rno_43  rno_44  rno_56
0       1       0       0       0       0       0
1       0       1       0       0       0       0
2       0       0       1       0       0       0
3       0       0       0       1       0       0
4       0       0       0       0       1       0
5       0       0       0       0       0       1
```

Out[9]:

```
OneHotEncoder()
```

In [15]:

```
# Converting type of columns to category
df['rno']=df['rno'].astype('category')
df['total_marks']=df['total_marks'].astype('category')


#Assigning numerical values and storing it in another columns
df['rno_new']=df['rno'].cat.codes
df['total_marks_new']=df['total_marks'].cat.codes


#Create an instance of One-hot-encoder
enc=OneHotEncoder()

#Passing encoded columns
'''
NOTE: we have converted the enc.fit_transform()
method to array because the fit_transform method
of OneHotEncoder returns SpiPy sparse matrix
this enables us to save space when we
have huge  number of categorical variables
'''
enc_df=pd.DataFrame(enc.fit_transform(df[['rno_new','total_marks_new']]).toarray())

#Merge with main dataframe df by using join

New_df=df.join(enc_df)

print(New_df)
```

```
  rno firstname  middlename  lastname total_marks                  fullnam
e  \
0  25    shivam    bhimling  limbhare         120    shivambhimlinglimbhar
e
1  33   sankalp     santosh     oswal         120       sankalpsantoshoswa
l
2  35  umarkhan  zaheerkhan    pathan         100  umarkhanzaheerkhanpatha
n
3  43    mahesh      ramesh     patil         120         maheshrameshpati
l
4  44    manjit      ganesh     patil         120         manjitganeshpati
l
5  56   bhavesh      satish     shete         120        bhaveshsatishshet
e

   rno_new  total_marks_new  natural_log  logarithm_tobase2    0    1    2
\
0        0                1          0.0                0.0  1.0  0.0  0.0
1        1                1          0.0                0.0  0.0  1.0  0.0
2        2                0         -inf               -inf  0.0  0.0  1.0
3        3                1          0.0                0.0  0.0  0.0  0.0
4        4                1          0.0                0.0  0.0  0.0  0.0
5        5                1          0.0                0.0  0.0  0.0  0.0

     3    4    5    6    7
0  0.0  0.0  0.0  0.0  1.0
1  0.0  0.0  0.0  0.0  1.0
2  0.0  0.0  0.0  1.0  0.0
3  1.0  0.0  0.0  0.0  1.0
```

```
4  0.0  1.0  0.0  0.0  1.0
5  0.0  0.0  1.0  0.0  1.0
```

In [11]:

```
1  # log transform
```

In [17]:

```
1  import numpy as np
2
3  # Calculate natural logarithm on  'total_marks_new' column
4
5  df['natural_log'] = np.log(df['rno_new'])
6
7  df   # Show the dataframe
```

```
C:\Users\UmarKhan pathan\anaconda3\lib\site-packages\pandas\core\arraylike.p
y:397: RuntimeWarning: divide by zero encountered in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

Out[17]:

	rno	firstname	middlename	lastname	total_marks	fullname	rno_new	tota
0	25	shivam	bhimling	limbhare	120	shivambhimlinglimbhare	0	
1	33	sankalp	santosh	oswal	120	sankalpsantoshoswal	1	
2	35	umarkhan	zaheerkhan	pathan	100	umarkhanzaheerkhanpathan	2	
3	43	mahesh	ramesh	patil	120	maheshrameshpatil	3	
4	44	manjit	ganesh	patil	120	manjitganeshpatil	4	
5	56	bhavesh	satish	shete	120	bhaveshsatishshete	5	

In [18]:

```
# Calculate logarithm to base 2 on  'total_marks_new' column

df['logarithm_tobase2'] = np.log2(df['rno_new'])

df   # Show the dataframe
```

```
C:\Users\UmarKhan pathan\anaconda3\lib\site-packages\pandas\core\arraylike.p
y:397: RuntimeWarning: divide by zero encountered in log2
  result = getattr(ufunc, method)(*inputs, **kwargs)
```

Out[18]:

	rno	firstname	middlename	lastname	total_marks	fullname	rno_new	tota
0	25	shivam	bhimling	limbhare	120	shivambhimlinglimbhare	0	
1	33	sankalp	santosh	oswal	120	sankalpsantoshoswal	1	
2	35	umarkhan	zaheerkhan	pathan	100	umarkhanzaheerkhanpathan	2	
3	43	mahesh	ramesh	patil	120	maheshrameshpatil	3	
4	44	manjit	ganesh	patil	120	manjitganeshpatil	4	
5	56	bhavesh	satish	shete	120	bhaveshsatishshete	5	

In [ ]:

```

```