Dealing with Missing Values

In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
data=pd.read_csv("/Users/Vaibhav/Desktop/Academic_Performance.csv")
```

In [3]:

```
data.head()
```

Out[3]:

	STUDENT_ID	GENDER	PLACEMENT	HONOR_OPTED_OR_NOT	EDUCATION_TYPE
0	SB11201210000129	F	Yes	Yes	ACADEMIC
1	SB11201210000137	F	Yes	Yes	ACADEMIC
2	SB11201210005154	M	No	Yes	ACADEMIC
3	SB11201210007504	F	Yes	Yes	ACADEMIC
4	SB11201210007548	M	Yes	Yes	ACADEMIC

In [4]:

```
missing_values=data.isnull().sum()
missing_values
```

Out[4]:

```
STUDENT_ID             0
GENDER                22
PLACEMENT             15
HONOR_OPTED_OR_NOT    14
EDUCATION_TYPE        15
ACADEMIC_PROGRAM      34
COURSE 1 MARKS        11
COURSE 2 MARKS         8
COURSE 3 MARKS        14
COURSE 4 MARKS        14
COURSE 5 MARKS        22
PERCENTILE             0
OVEARLL_GRADE          0
dtype: int64
```

In [5]:

```
miss_values_per= data.isnull().sum()/len(data)*100
miss_values_per
```

Out[5]:

```
STUDENT_ID           0.000000
GENDER               0.177262
PLACEMENT            0.120861
HONOR_OPTED_OR_NOT   0.112803
EDUCATION_TYPE       0.120861
ACADEMIC_PROGRAM     0.273951
COURSE 1 MARKS       0.088631
COURSE 2 MARKS       0.064459
COURSE 3 MARKS       0.112803
COURSE 4 MARKS       0.112803
COURSE 5 MARKS       0.177262
PERCENTILE           0.000000
OVEARLL_GRADE        0.000000
dtype: float64
```

In [6]:

```
!pip install missingno
```

```
Requirement already satisfied: missingno in c:\users\vaibhav\anaconda3\lib
\site-packages (0.5.1)
Requirement already satisfied: numpy in c:\users\vaibhav\anaconda3\lib\sit
e-packages (from missingno) (1.20.1)
Requirement already satisfied: seaborn in c:\users\vaibhav\anaconda3\lib\s
ite-packages (from missingno) (0.11.1)
Requirement already satisfied: matplotlib in c:\users\vaibhav\anaconda3\li
b\site-packages (from missingno) (3.3.4)
Requirement already satisfied: scipy in c:\users\vaibhav\anaconda3\lib\sit
e-packages (from missingno) (1.6.2)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\vaibhav\anaco
nda3\lib\site-packages (from matplotlib->missingno) (1.3.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in
c:\users\vaibhav\anaconda3\lib\site-packages (from matplotlib->missingno)
(2.4.7)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\vaibhav\an
aconda3\lib\site-packages (from matplotlib->missingno) (2.8.1)
Requirement already satisfied: cycler>=0.10 in c:\users\vaibhav\anaconda3
\lib\site-packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: pillow>=6.2.0 in c:\users\vaibhav\anaconda3
\lib\site-packages (from matplotlib->missingno) (8.2.0)
Requirement already satisfied: six in c:\users\vaibhav\anaconda3\lib\site-
packages (from cycler>=0.10->matplotlib->missingno) (1.15.0)
Requirement already satisfied: pandas>=0.23 in c:\users\vaibhav\anaconda3
\lib\site-packages (from seaborn->missingno) (1.2.4)
Requirement already satisfied: pytz>=2017.3 in c:\users\vaibhav\anaconda3
\lib\site-packages (from pandas>=0.23->seaborn->missingno) (2021.1)
```

In [7]:

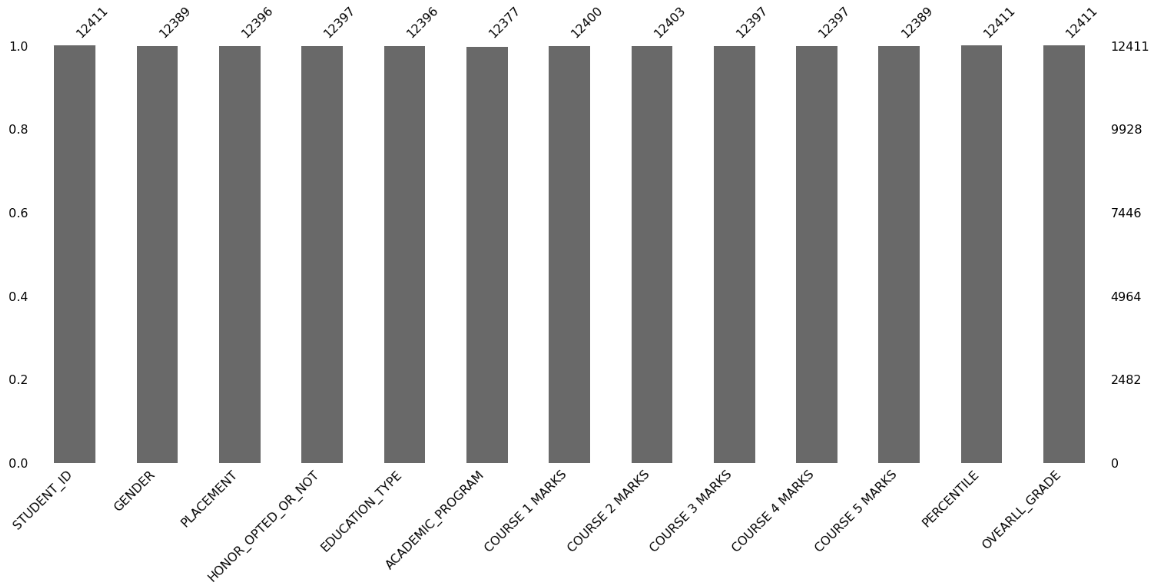
```
import missingno as msno
```

In [8]:

```
msno.bar(data)
```

Out[8]:

```
<AxesSubplot:>
```



Inputing the missing values

In [9]:

```
# Replacing null with mean
data['COURSE 1 MARKS']=data['COURSE 1 MARKS'].replace(np.NaN , data['COURSE 1 MARKS'].mea
```

In [10]:

```
data['COURSE 1 MARKS'].isnull().sum()
```

Out[10]:

```
0
```

In [11]:

```
# Replacing null with arbitrary value
data['COURSE 2 MARKS']=data['COURSE 2 MARKS'].fillna(0)
```

In [12]:

```
data['COURSE 2 MARKS'].isnull().sum()
```

Out[12]:

```
0
```

In [13]:

```
data
```

Out[13]:

	STUDENT_ID	GENDER	PLACEMENT	HONOR_OPTED_OR_NOT	EDUCATION_TYF
0	SB11201210000129	F	Yes	Yes	ACADEM
1	SB11201210000137	F	Yes	Yes	ACADEM
2	SB11201210005154	M	No	Yes	ACADEM
3	SB11201210007504	F	Yes	Yes	ACADEM
4	SB11201210007548	M	Yes	Yes	ACADEM
...	...	...	...	...	
12406	SB11201420568705	M	Yes	Yes	ACADEM
12407	SB11201420573045	M	Yes	Yes	ACADEM
12408	SB11201420578809	M	Yes	No	ACADEM
12409	SB11201420578812	F	Yes	Yes	ACADEM
12410	SB11201420583232	M	No	No	ACADEM

12411 rows × 13 columns

In [20]:

```
from sklearn.impute import SimpleImputer
imputer=SimpleImputer(strategy='most_frequent')

data['ACADEMIC_PROGRAM']=imputer.fit_transform(data['ACADEMIC_PROGRAM'].values.reshape(-1
```

In [21]:

```
data['ACADEMIC_PROGRAM'].isnull().sum()
```

Out[21]:

```
0
```

In [22]:

```
imputer=SimpleImputer(strategy='constant',fill_value='missing')

data['HONOR_OPTED_OR_NOT']=imputer.fit_transform(data['HONOR_OPTED_OR_NOT'].values.reshap
```

In [23]:

```
data['HONOR_OPTED_OR_NOT'].isnull().sum()
```

Out[23]:

```
0
```

In [24]:

```
data['HONOR_OPTED_OR_NOT'].unique()
```

Out[24]:

```
array(['Yes', 'No', 'missing'], dtype=object)
```

Dealing with Outliers

In [26]:

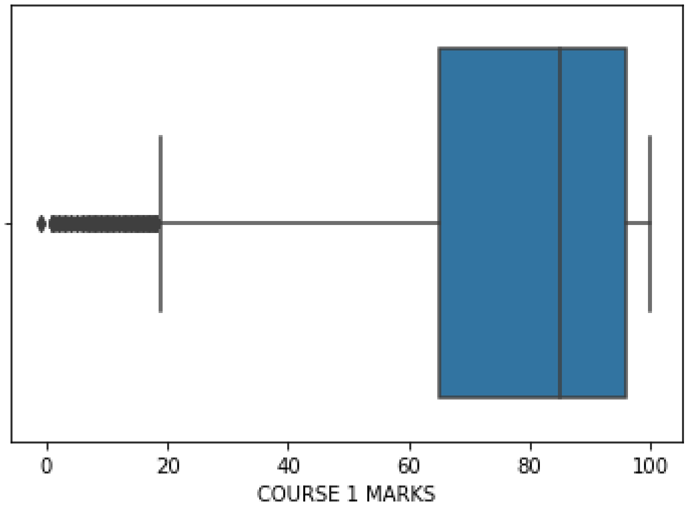
```
import seaborn as sns
import pandas as pd
import numpy as np
```

In [27]:

```
#data=pd.read_csv("/Users/Vaibhav/Desktop/Academic_Performance.csv")
sns.boxplot(data=data['COURSE 1 MARKS'],x=data['COURSE 1 MARKS'])
```

Out[27]:

```
<AxesSubplot:xlabel='COURSE 1 MARKS'>
```



In [28]:

```
data['COURSE 1 MARKS']
```

Out[28]:

```
0        71.0
1        97.0
2        17.0
3        65.0
4        94.0
         ... 
12406    88.0
12407    46.0
12408    98.0
12409    60.0
12410    83.0
Name: COURSE 1 MARKS, Length: 12411, dtype: float64
```

Detecting outliers using Z-scores

In [6]:

```
import numpy as np
outliers = []
def detect_outliers_zscore(arr):
    thres = 3
    mean = np.mean(arr)
    std = np.std(arr)
    # print(mean, std)
    for i in arr:
        z_score = (i-mean)/std
        if (np.abs(z_score) > thres):
            outliers.append(i)
    return outliers
marks_outliers = detect_outliers_zscore(data['COURSE 1 MARKS'])
print("Outliers from Z-scores method: ", marks_outliers)
```

```
Outliers from Z-scores method:  [4.0, 6.0, 3.0, 1.0, 5.0, 2.0, 8.0, 7.0,
2.0, 8.0, 6.0, 9.0, 2.0, 9.0, 8.0, 1.0, 9.0, 2.0, 2.0, 1.0, 6.0, 7.0, 4.0,
5.0, 9.0, 7.0, 9.0, 1.0, 2.0, 8.0, 5.0, 2.0, 8.0, 8.0, 1.0, 4.0, 7.0, 4.0,
7.0, 8.0, 3.0, 8.0, 5.0, 9.0, 7.0, 8.0, 7.0, 1.0, 9.0, 2.0, 7.0, 5.0, 3.0,
7.0, 3.0, 8.0, 6.0, 9.0, 8.0, 9.0, 6.0, 1.0, 7.0, 8.0, 1.0, 9.0, 1.0, 7.0,
8.0, 9.0, 6.0, 7.0, 7.0, 8.0, 4.0, 6.0, 6.0, 5.0, -1.0, 8.0, 8.0, 3.0, 1.
0, 3.0, 3.0, 2.0, 9.0, 8.0, 3.0, 6.0, 3.0, 2.0, 7.0, 8.0, 4.0, 8.0, 3.0,
7.0, 9.0, 9.0, 3.0, 7.0, 6.0, 1.0, 1.0, 1.0, -1.0, 9.0, 4.0, 8.0, 7.0, 1.
0, 6.0]
```

Another way to implement zscore.

In [30]:

```
high_thresh=data['COURSE 1 MARKS'].mean() + 3*data['COURSE 1 MARKS'].std()
high_thresh
```

Out[30]:

```
145.50435926680913
```

In [31]:

```
data['COURSE 1 MARKS'].max()
```

Out[31]:

```
100.0
```

In [32]:

```
data['COURSE 1 MARKS'].mean()
```

Out[32]:

```
77.3858870967742
```

In [33]:

```
data['COURSE 1 MARKS'].std()
```

Out[33]:

```
22.706157390011644
```

In [34]:

```
lowest_thresh=data['COURSE 1 MARKS'].mean() - 3*data['COURSE 1 MARKS'].std()
lowest_thresh
```

Out[34]:

```
9.267414926739264
```

In [35]:

```
newdf=data[(data['COURSE 1 MARKS']<lowest_thresh) | (data['COURSE 1 MARKS']>high_thresh)]
```

In [36]:

```
newdf['COURSE 1 MARKS']
```

Out[36]:

```
11       4.0
155      6.0
167      3.0
305      1.0
353      5.0
        ...
11995    4.0
12009    8.0
12068    7.0
12137    1.0
12246    6.0
Name: COURSE 1 MARKS, Length: 113, dtype: float64
```

Trimming

In [38]:

```
newdf1=data[(data['COURSE 1 MARKS']>lowest_thresh) & (data['COURSE 1 MARKS']<high_thresh)
newdf1
```

Out[38]:

	STUDENT_ID	GENDER	PLACEMENT	HONOR_OPTED_OR_NOT	EDUCATION_TYF
0	SB11201210000129	F	Yes	Yes	ACADEM
1	SB11201210000137	F	Yes	Yes	ACADEM
2	SB11201210005154	M	No	Yes	ACADEM
3	SB11201210007504	F	Yes	Yes	ACADEM
4	SB11201210007548	M	Yes	Yes	ACADEM
...	...	...	...	...	
12406	SB11201420568705	M	Yes	Yes	ACADEM
12407	SB11201420573045	M	Yes	Yes	ACADEM
12408	SB11201420578809	M	Yes	No	ACADEM
12409	SB11201420578812	F	Yes	Yes	ACADEM
12410	SB11201420583232	M	No	No	ACADEM

12298 rows × 13 columns

In [39]:

```
data.shape
```

Out[39]:

```
(12411, 13)
```

In [41]:

```
newdf1.shape
```

Out[41]:

```
(12298, 13)
```

Data Transformation

In [42]:

```
categorical_data=data.select_dtypes(exclude=[np.number])
categorical_data
```

Out[42]:

	STUDENT_ID	GENDER	PLACEMENT	HONOR_OPTED_OR_NOT	EDUCATION_TYF
0	SB11201210000129	F	Yes	Yes	ACADEM
1	SB11201210000137	F	Yes	Yes	ACADEM
2	SB11201210005154	M	No	Yes	ACADEM
3	SB11201210007504	F	Yes	Yes	ACADEM
4	SB11201210007548	M	Yes	Yes	ACADEM
...	...	...	...	...	
12406	SB11201420568705	M	Yes	Yes	ACADEM
12407	SB11201420573045	M	Yes	Yes	ACADEM
12408	SB11201420578809	M	Yes	No	ACADEM
12409	SB11201420578812	F	Yes	Yes	ACADEM
12410	SB11201420583232	M	No	No	ACADEM

12411 rows × 7 columns

In [43]:

```
categorical_data['PLACEMENT'].unique()
```

Out[43]:

```
array(['Yes', 'No', nan], dtype=object)
```

In [44]:

```
categorical_data.PLACEMENT.value_counts()
```

Out[44]:

```
Yes    9740
No     2656
Name: PLACEMENT, dtype: int64
```

In [45]:

```
categorical_data.PLACEMENT.replace({'Yes':1,'No':-1},inplace=True)
```

```
C:\Users\Vaibhav\anaconda3\lib\site-packages\pandas\core\series.py:4509: S
ettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://
pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
view-versus-a-copy)
  return super().replace(
```

In [39]:

```
categorical_data
```

Out[39]:

	STUDENT_ID	GENDER	PLACEMENT	HONOR_OPTED_OR_NOT	EDUCATION_TYF
0	SB11201210000129	F	1.0	Yes	ACADEM
1	SB11201210000137	F	1.0	Yes	ACADEM
2	SB11201210005154	M	-1.0	Yes	ACADEM
3	SB11201210007504	F	1.0	Yes	ACADEM
4	SB11201210007548	M	1.0	Yes	ACADEM
...	...	...	...	...	
12406	SB11201420568705	M	1.0	Yes	ACADEM
12407	SB11201420573045	M	1.0	Yes	ACADEM
12408	SB11201420578809	M	1.0	No	ACADEM
12409	SB11201420578812	F	1.0	Yes	ACADEM
12410	SB11201420583232	M	-1.0	No	ACADEM

12411 rows × 7 columns

In [46]:

```
categorical_data=categorical_data.drop('STUDENT_ID',axis=1)
```

In [47]:

```
categorical_data
```

Out[47]:

	GENDER	PLACEMENT	HONOR_OPTED_OR_NOT	EDUCATION_TYPE	ACADEMIC_PRO
0	F	1.0	Yes	ACADEMIC	INDUS ENGINE
1	F	1.0	Yes	ACADEMIC	INDUS ENGINE
2	M	-1.0	Yes	ACADEMIC	ELECTI ENGINE
3	F	1.0	Yes	ACADEMIC	INDUS ENGINE
4	M	1.0	Yes	ACADEMIC	INDUS ENGINE
...	...	...	...	...	
12406	M	1.0	Yes	ACADEMIC	MECHATR ENGINE
12407	M	1.0	Yes	ACADEMIC	INDUS ENGINE
12408	M	1.0	No	ACADEMIC	INDUS ENGINE
12409	F	1.0	Yes	ACADEMIC	r
12410	M	-1.0	No	ACADEMIC	INDUS ENGINE

12411 rows × 6 columns

In [50]:

```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()

for i in categorical_data:

    categorical_data[i] = label_encoder.fit_transform(categorical_data[i])

print("Label Encoded Data: ")

categorical_data.head()
```

```
Label Encoded Data: 
```

Out[50]:

	GENDER	PLACEMENT	HONOR_OPTED_OR_NOT	EDUCATION_TYPE	ACADEMIC_PROGRA
0	0	1	1	0	1
1	0	1	1	0	1
2	1	0	1	0	1
3	0	1	1	0	1
4	1	1	1	0	1

In [56]:

```
# One hot encoding
categorical_df=data.select_dtypes(exclude=[np.number])
categorical_df

one_hot_encoded_data = pd.get_dummies(categorical_df, columns = ['PLACEMENT', 'GENDER'])
print(one_hot_encoded_data)
```

```
             STUDENT_ID HONOR_OPTED_OR_NOT EDUCATION_TYPE  \
0      SB11201210000129                Yes       ACADEMIC
1      SB11201210000137                Yes       ACADEMIC
2      SB11201210005154                Yes       ACADEMIC
3      SB11201210007504                Yes       ACADEMIC
4      SB11201210007548                Yes       ACADEMIC
...                 ...                ...            ...
12406  SB11201420568705                Yes       ACADEMIC
12407  SB11201420573045                Yes       ACADEMIC
12408  SB11201420578809                 No       ACADEMIC
12409  SB11201420578812                Yes       ACADEMIC
12410  SB11201420583232                 No       ACADEMIC

               ACADEMIC_PROGRAM OVEARLL_GRADE  PLACEMENT_No  PLACEMENT_Yes
\
0        INDUSTRIAL ENGINEERING   FIRST CLASS             0              1
1        INDUSTRIAL ENGINEERING   THIRD CLASS             0              1
2        ELECTRONIC ENGINEERING   DISTINCTION             1              0
3        INDUSTRIAL ENGINEERING   FIRST CLASS             0              1
4        INDUSTRIAL ENGINEERING   FIRST CLASS             0              1
...                         ...           ...           ...            ...
12406  MECHATRONICS ENGINEERING   FIRST CLASS             0              1
12407    INDUSTRIAL ENGINEERING   FIRST CLASS             0              1
12408    INDUSTRIAL ENGINEERING   FIRST CLASS             0              1
12409                   missing   FIRST CLASS             0              1
12410    INDUSTRIAL ENGINEERING   THIRD CLASS             1              0

       GENDER_F  GENDER_M
0             1         0
1             1         0
2             0         1
3             1         0
4             0         1
...         ...       ...
12406         0         1
12407         0         1
12408         0         1
12409         1         0
12410         0         1

[12411 rows x 9 columns]
```

In [ ]: