```
You will use ‘Admission_Predict.csv’ for Midterm. This dataset includes the
data of the applicants of an academic program. Each application has a unique
serial number, which represents a particular student. The dataset contains
several parameters which are considered important during the application for
Masters Programs. The parameters included are :
1)  GRE Scores (out of 340)
2) TOEFL Scores (out of 120)
3) University Rating (out of 5)
4) Statement of Purpose (SOP) (out of 5)
5) Letter of Recommendation (LOR) Strength (out of 5)
6) Undergraduate GPA (out of 10)
7) Research Experience (either 0 or 1)
8) Chance of Admit (ranging from 0 to 1)
```

Q1: Download “Admission_Predict.csv” dataset and load it as ‘data’.

In [62]:
```
import pandas as pd
```

In [63]:
```
data = pd.read_csv("/content/Admission_Predict.csv")
```

Q2: Display the first three rows in this dataset.

In [64]:
```
data.head(3)
```

Out[64]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72

Q3: check the duplicate records. if yes then remove the duplicate records.

In [65]:
```
df = data
```

In [66]:
```
df.duplicated()
```

Out[66]:
```
0      False
1      False
2      False
3      False
4      False
       ...
395    False
396    False
397    False
398    False
399    False
Length: 400, dtype: bool
```

In [67]:
```
df.duplicated().sum()
```

Out[67]:
```
0
```

Q4: Are there any missing values in the dataset?

In [68]:
```
df.isnull()
```

Out[68]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...
395	False	False	False	False	False	False	False	False	False
396	False	False	False	False	False	False	False	False	False
397	False	False	False	False	False	False	False	False	False
398	False	False	False	False	False	False	False	False	False
399	False	False	False	False	False	False	False	False	False

400 rows × 9 columns

In [69]:
```
df.isnull().sum()
```

Out[69]:
```
Serial No.           0
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
Chance of Admit      0
dtype: int64
```

Q5. Remove the missing value.

In [69]:
```
#There are no missing values
```

Q6: - Display the structure of all variables

In [70]:
```
df.shape
```

Out[70]:
```
(400, 9)
```

Q7: Check the datatypes of the attributes.

In [71]:
```
df.dtypes
```

Out[71]:
```
Serial No.             int64
GRE Score              int64
TOEFL Score            int64
University Rating      int64
SOP                  float64
LOR                  float64
CGPA                 float64
Research               int64
Chance of Admit      float64
dtype: object
```

Q8: change the data type according to the discribution of the data set

In [72]:
```
#The data type does not need to be changed
```

Q9: Print the descriptive statistics of the admission data to understand the data a little better (min, max, mean, median, 1st and 3rd quartiles).

In [73]:
```
df.describe()
```

Out[73]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	R
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400
mean	200.500000	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0
std	115.614301	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0
min	1.000000	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0
25%	100.750000	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0
50%	200.500000	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1
75%	300.250000	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1
max	400.000000	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1

Q10: Use a histogram to assess the normality of the 'Chance.of.Admit' variable and explain whether it appears normally distributed or not and why?

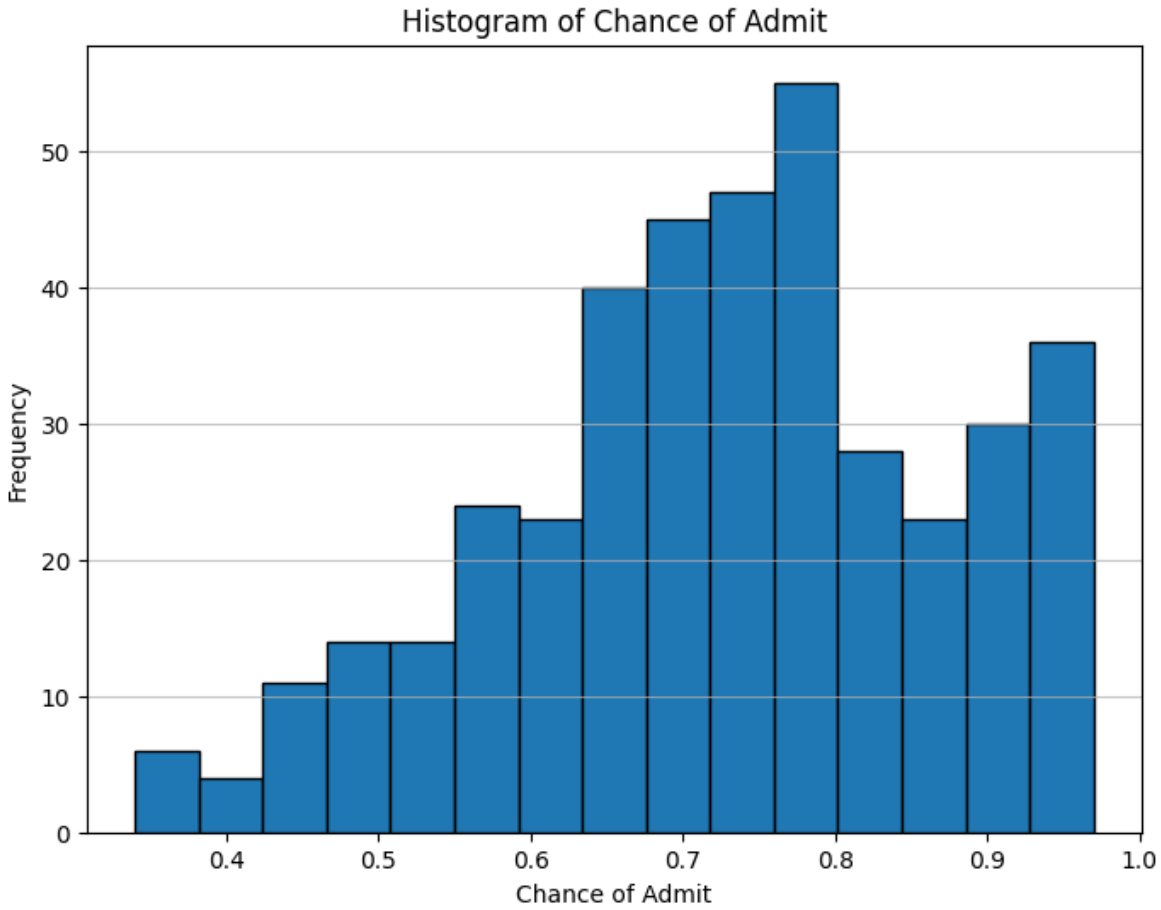
In [74]:
```
chance_of_admit_column = data['Chance of Admit ']
print(data.columns)
data['Chance of Admit ']

import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
plt.hist(chance_of_admit_column, bins=15, edgecolor='k')
plt.title('Histogram of Chance of Admit')
plt.xlabel('Chance of Admit')
plt.ylabel('Frequency')
plt.grid(axis='y', alpha=0.75)

# Show the histogram
plt.show()
```

```
Index(['Serial No.', 'GRE Score', 'TOEFL Score', 'University Rating', 'SOP',
       'LOR ', 'CGPA', 'Research', 'Chance of Admit '],
      dtype='object')
```



Q11: covert the categorical attribute into numeric using ine hot incoding method.

In [75]:
```
df = pd.get_dummies(df)
df.head(2)
```

Out[75]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76

Q12: Normalize the data set.

In [75]:
```
#There is no need to normalize the data set
```

Q13: Divide the dataset to training and test sets.

In [85]:
```
dfx = df.drop('Chance of Admit ', axis = 1)
```

In [86]:
```
data.head(2)
```

Out[86]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76

In [88]:
```
dfy = data['Chance of Admit ']
```

In [91]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [92]:
```
scaler = MinMaxScaler()
```

In [93]:
```
scaler.fit_transform(dfx)
```

Out[93]:
```
array([[0.        , 0.94      , 0.92857143, ..., 0.875     , 0.91346154,
        1.        ],
       [0.00250627, 0.68      , 0.53571429, ..., 0.875     , 0.66346154,
        1.        ],
       [0.00501253, 0.52      , 0.42857143, ..., 0.625     , 0.38461538,
        1.        ],
       ...,
       [0.99498747, 0.8       , 0.85714286, ..., 0.875     , 0.84935897,
        1.        ],
       [0.99749373, 0.44      , 0.39285714, ..., 0.75      , 0.63461538,
        0.        ],
       [1.        , 0.86      , 0.89285714, ..., 0.75      , 0.91666667,
        1.        ]])
```

In [94]:
```
dfx.columns
```

Out[94]:
```
Index(['Serial No.', 'GRE Score', 'TOEFL Score', 'University Rating', 'SOP',
       'LOR ', 'CGPA', 'Research'],
      dtype='object')
```

In [95]:
```
pd.DataFrame (scaler.fit_transform(dfx))
```

Out[95]:

	0	1	2	3	4	5	6	7
0	0.000000	0.94	0.928571	0.75	0.875	0.875	0.913462	1.0
1	0.002506	0.68	0.535714	0.75	0.750	0.875	0.663462	1.0
2	0.005013	0.52	0.428571	0.50	0.500	0.625	0.384615	1.0
3	0.007519	0.64	0.642857	0.50	0.625	0.375	0.599359	1.0
4	0.010025	0.48	0.392857	0.25	0.250	0.500	0.451923	0.0
...	...	...	...	...	...	...	...	...
395	0.989975	0.68	0.642857	0.50	0.625	0.625	0.717949	1.0
396	0.992481	0.70	0.535714	0.50	0.500	0.625	0.740385	1.0
397	0.994987	0.80	0.857143	0.75	1.000	0.875	0.849359	1.0
398	0.997494	0.44	0.392857	0.50	0.625	0.750	0.634615	0.0
399	1.000000	0.86	0.892857	0.75	1.000	0.750	0.916667	1.0

400 rows × 8 columns

In [96]:
```
pd.DataFrame(scaler.fit_transform(dfx),columns = dfx.columns)
```

Out[96]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
0	0.000000	0.94	0.928571	0.75	0.875	0.875	0.913462	1.0
1	0.002506	0.68	0.535714	0.75	0.750	0.875	0.663462	1.0
2	0.005013	0.52	0.428571	0.50	0.500	0.625	0.384615	1.0
3	0.007519	0.64	0.642857	0.50	0.625	0.375	0.599359	1.0
4	0.010025	0.48	0.392857	0.25	0.250	0.500	0.451923	0.0
...	...	...	...	...	...	...	...	...
395	0.989975	0.68	0.642857	0.50	0.625	0.625	0.717949	1.0
396	0.992481	0.70	0.535714	0.50	0.500	0.625	0.740385	1.0
397	0.994987	0.80	0.857143	0.75	1.000	0.875	0.849359	1.0
398	0.997494	0.44	0.392857	0.50	0.625	0.750	0.634615	0.0
399	1.000000	0.86	0.892857	0.75	1.000	0.750	0.916667	1.0

400 rows × 8 columns

In [97]:
```
scaler_dfx = pd.DataFrame(scaler.fit_transform(dfx),columns = dfx.columns)
scaler_dfx.head(10)
```

Out[97]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
0	0.000000	0.94	0.928571	0.75	0.875	0.875	0.913462	1.0
1	0.002506	0.68	0.535714	0.75	0.750	0.875	0.663462	1.0
2	0.005013	0.52	0.428571	0.50	0.500	0.625	0.384615	1.0
3	0.007519	0.64	0.642857	0.50	0.625	0.375	0.599359	1.0
4	0.010025	0.48	0.392857	0.25	0.250	0.500	0.451923	0.0
5	0.012531	0.80	0.821429	1.00	0.875	0.500	0.814103	1.0
6	0.015038	0.62	0.607143	0.50	0.500	0.750	0.448718	1.0
7	0.017544	0.36	0.321429	0.25	0.500	0.750	0.352564	0.0
8	0.020050	0.24	0.357143	0.00	0.250	0.125	0.384615	0.0
9	0.022556	0.66	0.571429	0.50	0.625	0.500	0.576923	0.0

In [98]:
```
dfy
```

Out[98]:
```
0      0.92
1      0.76
2      0.72
3      0.80
4      0.65
       ...
395    0.82
396    0.84
397    0.91
398    0.67
399    0.95
Name: Chance of Admit , Length: 400, dtype: float64
```

In [99]:
```
scaler_dfx.shape
```

Out[99]:
```
(400, 8)
```

In [100]:
```
from sklearn.model_selection import train_test_split
```

In [105]:
```
# Split the dataset into training and test sets (e.g., 80% training, 20% testi
input_train, input_test, output_train, output_test = train_test_split(scaler_d
```

In [106]:
```
input_train.shape[0]
```

Out[106]:
```
320
```

In [107]:
```
input_train
```

Out[107]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
93	0.233083	0.22	0.178571	0.25	0.500	0.500	0.346154	1.0
23	0.057644	0.88	0.964286	1.00	1.000	0.875	0.929487	1.0
299	0.749373	0.30	0.714286	0.50	0.500	0.625	0.592949	0.0
13	0.032581	0.34	0.607143	0.50	0.750	0.500	0.384615	1.0
90	0.225564	0.56	0.500000	0.25	0.750	0.750	0.358974	1.0
...	...	...	...	...	...	...	...	...
255	0.639098	0.34	0.642857	0.75	0.750	0.875	0.503205	0.0
72	0.180451	0.62	0.678571	1.00	1.000	1.000	0.849359	1.0
396	0.992481	0.70	0.535714	0.50	0.500	0.625	0.740385	1.0
235	0.588972	0.72	0.678571	1.00	0.875	0.750	0.778846	1.0
37	0.092732	0.20	0.464286	0.00	0.000	0.250	0.320513	0.0

320 rows × 8 columns

In [108]:
```
input_train,input_test,output_train,output_test
```

Out[108]:
```
(     Serial No.  GRE Score  TOEFL Score  University Rating    SOP   LOR
\
 93     0.233083       0.22     0.178571               0.25  0.500  0.500
 23     0.057644       0.88     0.964286               1.00  1.000  0.875
 299    0.749373       0.30     0.714286               0.50  0.500  0.625
 13     0.032581       0.34     0.607143               0.50  0.750  0.500
 90     0.225564       0.56     0.500000               0.25  0.750  0.750
 ..          ...        ...          ...                ...    ...    ...
 255    0.639098       0.34     0.642857               0.75  0.750  0.875
 72     0.180451       0.62     0.678571               1.00  1.000  1.000
 396    0.992481       0.70     0.535714               0.50  0.500  0.625
 235    0.588972       0.72     0.678571               1.00  0.875  0.750
 37     0.092732       0.20     0.464286               0.00  0.000  0.250

          CGPA  Research
 93   0.346154       1.0
 23   0.929487       1.0
 299  0.592949       0.0
 13   0.384615       1.0
```

Q14: Use the KNN algorithm to predict the quality of wine using its attributes.

Q15: Evaluate the model performance by computing Accuracy.