In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
df = pd.read_csv('C:\\Users\\Sammed\\OneDrive\\Desktop\\Sammed\\Machine Learning\\Data-cleaning-for-beginners-using-pa
```

In [3]:
```
df.head()
```

Out[3]:

	Index	Age	Salary	Rating	Location	Established	Easy Apply
0	0	44.0	$44k-99k$	5.4	India,In	1999	TRUE
1	1	66.0	$55k-66k$	3.5	New York,Ny	2002	TRUE
2	2	NaN	$77k-89k$	-1.0	New York,Ny	-1	-1
3	3	64.0	$44k-99k$	4.4	India In	1988	-1
4	4	25.0	$44k-99k$	6.4	Australia Aus	2002	-1

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29 entries, 0 to 28
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Index        29 non-null     int64  
 1   Age          22 non-null     float64
 2   Salary       29 non-null     object 
 3   Rating       28 non-null     float64
 4   Location     29 non-null     object 
 5   Established  29 non-null     int64  
 6   Easy Apply   29 non-null     object 
dtypes: float64(2), int64(2), object(3)
memory usage: 1.7+ KB
```

In [5]:
```
df.isna().sum()
```

Out[5]:
```
Index          0
Age            7
Salary         0
Rating         1
Location       0
Established    0
Easy Apply     0
dtype: int64
```

In [6]:
```
count_of_minus_1 = (df == -1).sum()
print(count_of_minus_1)
```

```
Index          0
Age            0
Salary         0
Rating         4
Location       0
Established    5
Easy Apply     0
dtype: int64
```

In [7]:
```
df.duplicated().sum()
```

Out[7]:
```
0
```

Q1

In [8]:
```
df.replace('-1', float('nan'), inplace=True)
df.replace(-1.0, float('nan'), inplace=True)
```

In [9]:
```
from sklearn.impute import SimpleImputer
```

In [10]:
```
simpleImputer = SimpleImputer(strategy='most_frequent',missing_values = np.nan)
```

In [11]:
```
df['Age'] = simpleImputer.fit_transform(df[['Age']])
df['Rating'] = simpleImputer.fit_transform(df[['Rating']])
df['Established'] = simpleImputer.fit_transform(df[['Established']])
df['Easy Apply'] = simpleImputer.fit_transform(df[['Easy Apply']])
```

Q2

In [12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29 entries, 0 to 28
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Index        29 non-null     int64  
 1   Age          29 non-null     float64
 2   Salary       29 non-null     object 
 3   Rating       29 non-null     float64
 4   Location     29 non-null     object 
 5   Established  29 non-null     float64
 6   Easy Apply   29 non-null     object 
dtypes: float64(3), int64(1), object(3)
memory usage: 1.7+ KB
```

Q3 Adjusted, as it does not cause loss of data

In [13]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['Salary'] = le.fit_transform(df[['Salary']])
```

```
C:\Users\ayush\anaconda3\envs\Ayush_Walekar_MIT\lib\site-packages\sklearn\preprocessing\_label.py:116: DataConversion
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), fo
r example using ravel().
  y = column_or_1d(y, warn=True)
```

In [14]:
```
import seaborn as sns
```

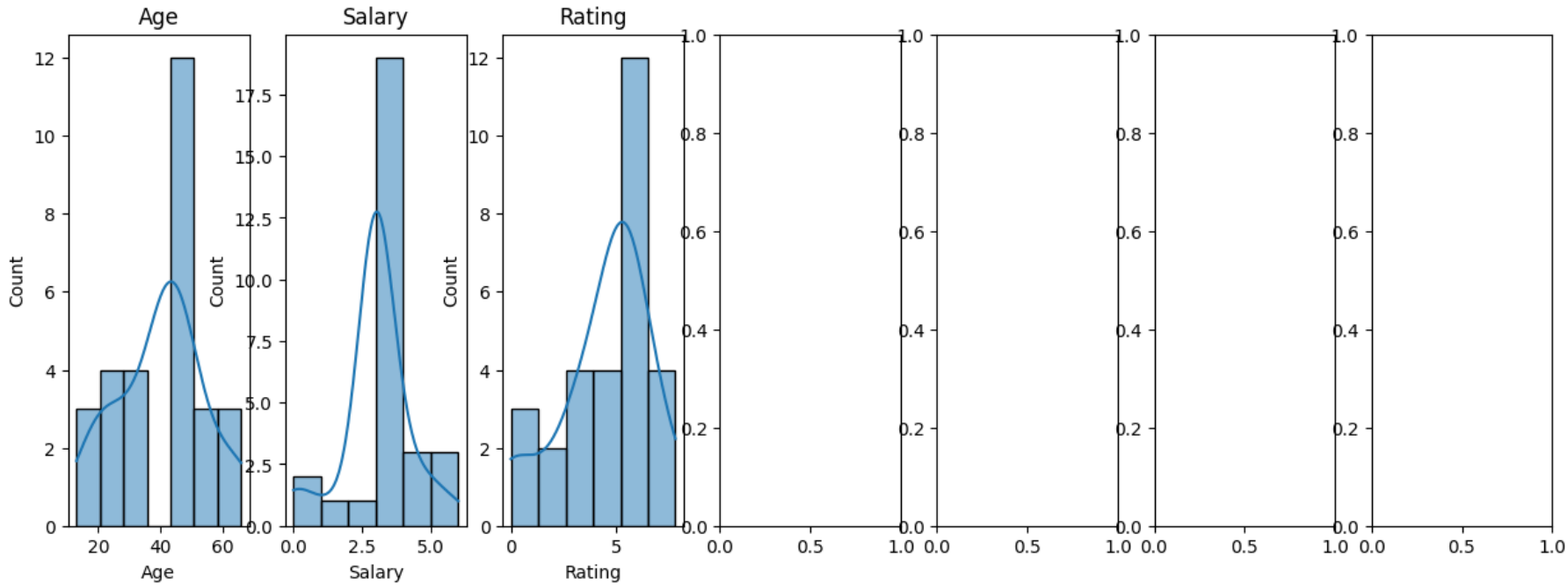
In [16]:
```
selected_target_cols = ['Age','Salary','Rating']
```

In [17]:
```
fig, axes = plt.subplots(nrows=1, ncols=len(df.columns), figsize=(15, 5))


for i, col in enumerate(selected_target_cols):

    if pd.api.types.is_numeric_dtype(df[col]):
        sns.histplot(df[col], kde=True, ax=axes[i])
        axes[i].set_title(f'{col}')


plt.show()
```



In [ ]:
```
```

Q4

In [18]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
df['Salary'] = scaler.fit_transform(df['Salary'].values.reshape(-1, 1))
```

In [19]:
```
df
```

Out[19]:

	Index	Age	Salary	Rating	Location	Established	Easy Apply
0	0	44.0	-0.028330	5.4	India,In	1999.0	TRUE
1	1	66.0	0.793230	3.5	New York,Ny	2002.0	TRUE
2	2	44.0	1.614789	5.4	New York,Ny	1999.0	TRUE
3	3	64.0	-0.028330	4.4	India In	1988.0	TRUE
4	4	25.0	-0.028330	6.4	Australia Aus	2002.0	TRUE
5	5	44.0	1.614789	1.4	India,In	1999.0	TRUE
6	6	21.0	-0.028330	0.0	New York,Ny	1999.0	TRUE
7	7	44.0	-0.028330	5.4	Australia Aus	1999.0	TRUE
8	8	35.0	-0.028330	5.4	New York,Ny	1999.0	TRUE
9	9	22.0	-0.028330	7.7	India,In	1999.0	TRUE
10	10	55.0	-2.493008	5.4	India,In	2008.0	TRUE
11	11	44.0	-2.493008	6.7	India,In	2009.0	TRUE
12	12	44.0	-0.028330	0.0	India,In	1999.0	TRUE
13	13	25.0	-0.028330	5.4	Australia Aus	2019.0	TRUE
14	14	66.0	-0.028330	4.0	Australia Aus	2020.0	TRUE
15	15	44.0	2.436349	3.0	Australia Aus	1999.0	TRUE
16	16	19.0	-1.671448	4.5	India,In	1984.0	TRUE
17	17	44.0	-0.028330	5.3	New York,Ny	1943.0	TRUE
18	18	35.0	-0.028330	6.7	New York,Ny	1954.0	TRUE
19	19	32.0	-0.028330	3.3	New York,Ny	1955.0	TRUE
20	20	44.0	-0.028330	5.7	New York,Ny	1944.0	TRUE
21	21	35.0	-0.028330	5.0	New York,Ny	1946.0	TRUE
22	22	19.0	0.793230	7.8	New York,Ny	1988.0	TRUE
23	23	44.0	-0.028330	2.4	New York,Ny	1999.0	TRUE
24	24	13.0	-0.028330	5.4	New York,Ny	1987.0	TRUE

	Index	Age	Salary	Rating	Location	Established	Easy Apply
25	25	55.0	-0.028330	0.0	Australia Aus	1980.0	TRUE
26	26	44.0	0.793230	5.4	India,In	1934.0	TRUE
27	27	52.0	-0.028330	5.4	India,In	1935.0	TRUE
28	28	44.0	-0.849889	3.4	Australia Aus	1932.0	TRUE

Q5 , Q10 , Q13

In [25]:
```
unique_locations = df['Location'].unique()
print(unique_locations)
```

```
['In' 'Ny' 'Aus']
```

In [26]:
```
df['Location'] = df['Location'].apply(lambda x: x.split(',')[-1].strip())
```

In [28]:
```
df['Location'].replace("Australia Aus","Aus",inplace=True)
df['Location'].replace("India In","In",inplace=True)
```

In [33]:
```
df
```

Out[33]:

	Index	Age	Salary	Rating	Location	Established	Easy Apply
0	0	44.0	-0.028330	5.4	In	1999.0	1
1	1	66.0	0.793230	3.5	Ny	2002.0	1
2	2	44.0	1.614789	5.4	Ny	1999.0	1
3	3	64.0	-0.028330	4.4	In	1988.0	1
4	4	25.0	-0.028330	6.4	Aus	2002.0	1
5	5	44.0	1.614789	1.4	In	1999.0	1
6	6	21.0	-0.028330	0.0	Ny	1999.0	1
7	7	44.0	-0.028330	5.4	Aus	1999.0	1
8	8	35.0	-0.028330	5.4	Ny	1999.0	1
9	9	22.0	-0.028330	7.7	In	1999.0	1
10	10	55.0	-2.493008	5.4	In	2008.0	1
11	11	44.0	-2.493008	6.7	In	2009.0	1
12	12	44.0	-0.028330	0.0	In	1999.0	1
13	13	25.0	-0.028330	5.4	Aus	2019.0	1
14	14	66.0	-0.028330	4.0	Aus	2020.0	1
15	15	44.0	2.436349	3.0	Aus	1999.0	1
16	16	19.0	-1.671448	4.5	In	1984.0	1
17	17	44.0	-0.028330	5.3	Ny	1943.0	1
18	18	35.0	-0.028330	6.7	Ny	1954.0	1
19	19	32.0	-0.028330	3.3	Ny	1955.0	1
20	20	44.0	-0.028330	5.7	Ny	1944.0	1
21	21	35.0	-0.028330	5.0	Ny	1946.0	1
22	22	19.0	0.793230	7.8	Ny	1988.0	1
23	23	44.0	-0.028330	2.4	Ny	1999.0	1
24	24	13.0	-0.028330	5.4	Ny	1987.0	1

	Index	Age	Salary	Rating	Location	Established	Easy Apply
25	25	55.0	-0.028330	0.0	Aus	1980.0	1
26	26	44.0	0.793230	5.4	In	1934.0	1
27	27	52.0	-0.028330	5.4	In	1935.0	1
28	28	44.0	-0.849889	3.4	Aus	1932.0	1

Q6 No

Q7 Q12 #### it contains boolean but we can encode them for better understanding

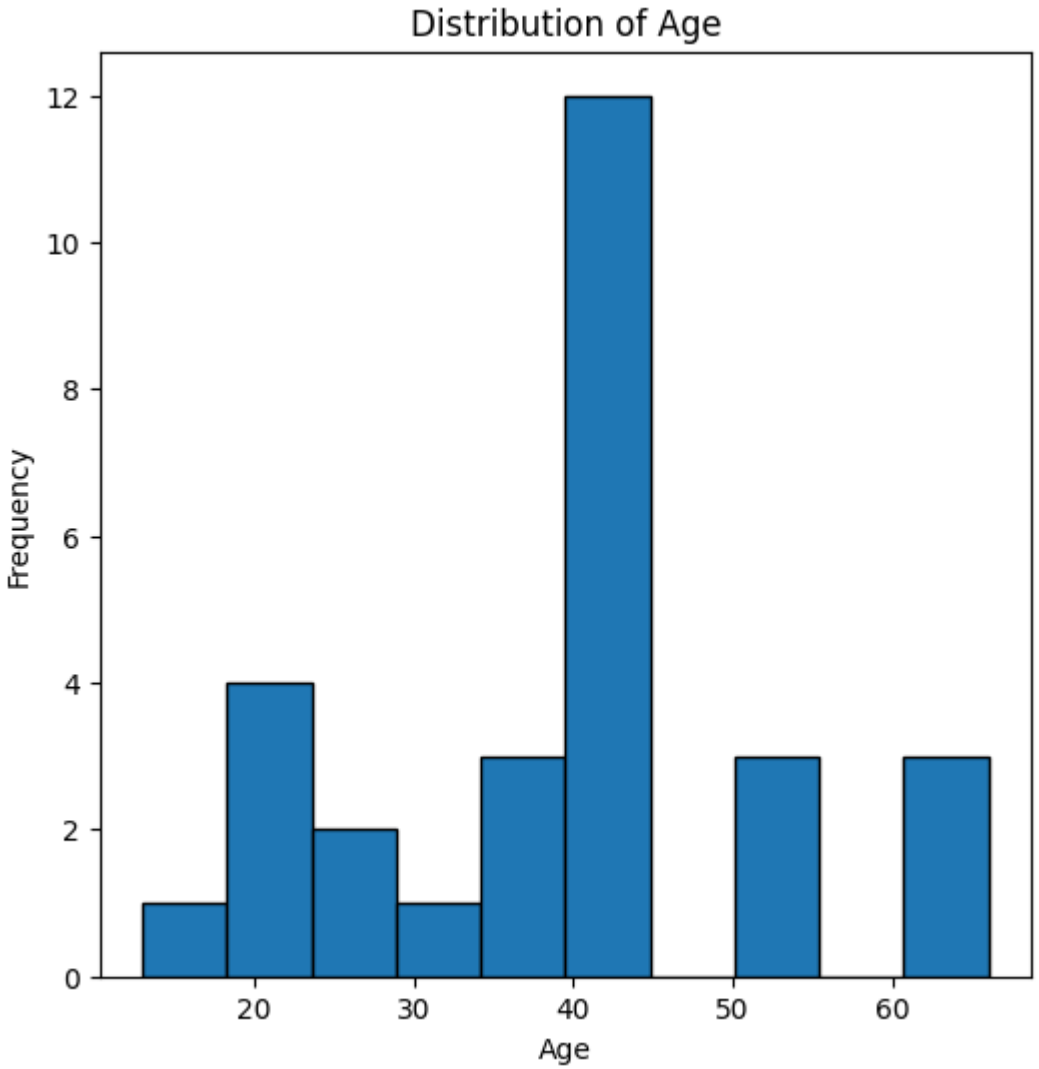
In [30]:
```
df['Easy Apply'].unique()
```

Out[30]:
```
array(['TRUE'], dtype=object)
```

In [32]:
```
df['Easy Apply'].replace('TRUE',1,inplace=True)
```

Q9

In [36]:
```
plt.figure(figsize=(6, 6))
plt.hist(df['Age'], bins=10, edgecolor='black')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```


Distribution of Age
Frequency
Age
0
2
4
6
8
10
12
20
30
40
50
60

In [37]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29 entries, 0 to 28
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Index        29 non-null     int64  
 1   Age          29 non-null     float64
 2   Salary       29 non-null     float64
 3   Rating       29 non-null     float64
 4   Location     29 non-null     object 
 5   Established  29 non-null     float64
 6   Easy Apply   29 non-null     int64  
dtypes: float64(4), int64(2), object(1)
memory usage: 1.7+ KB
```

Q14

In [38]:
```
df['Location'] = le.fit_transform(df[['Location']])
```

```
C:\Users\ayush\anaconda3\envs\Ayush_Walekar_MIT\lib\site-packages\sklearn\preprocessing\_label.py:116: DataConversion
Warning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), fo
r example using ravel().
  y = column_or_1d(y, warn=True)
```

Q15

In [41]:
```
df['Rating'] = scaler.fit_transform(df['Rating'].values.reshape(-1, 1))
```

In [42]:
```
df
```

Out[42]:

	Index	Age	Salary	Rating	Location	Established	Easy Apply
0	0	44.0	-0.028330	0.445207	1	1999.0	1
1	1	66.0	0.793230	-0.470126	2	2002.0	1
2	2	44.0	1.614789	0.445207	2	1999.0	1
3	3	64.0	-0.028330	-0.036547	1	1988.0	1
4	4	25.0	-0.028330	0.926961	0	2002.0	1
5	5	44.0	1.614789	-1.481809	1	1999.0	1
6	6	21.0	-0.028330	-2.156265	2	1999.0	1
7	7	44.0	-0.028330	0.445207	0	1999.0	1
8	8	35.0	-0.028330	0.445207	2	1999.0	1
9	9	22.0	-0.028330	1.553242	1	1999.0	1
10	10	55.0	-2.493008	0.445207	1	2008.0	1
11	11	44.0	-2.493008	1.071488	1	2009.0	1
12	12	44.0	-0.028330	-2.156265	1	1999.0	1
13	13	25.0	-0.028330	0.445207	0	2019.0	1
14	14	66.0	-0.028330	-0.229249	0	2020.0	1
15	15	44.0	2.436349	-0.711003	0	1999.0	1
16	16	19.0	-1.671448	0.011629	1	1984.0	1
17	17	44.0	-0.028330	0.397032	2	1943.0	1
18	18	35.0	-0.028330	1.071488	2	1954.0	1
19	19	32.0	-0.028330	-0.566476	2	1955.0	1
20	20	44.0	-0.028330	0.589734	2	1944.0	1
21	21	35.0	-0.028330	0.252506	2	1946.0	1
22	22	19.0	0.793230	1.601417	2	1988.0	1
23	23	44.0	-0.028330	-1.000055	2	1999.0	1
24	24	13.0	-0.028330	0.445207	2	1987.0	1

	Index	Age	Salary	Rating	Location	Established	Easy Apply
25	25	55.0	-0.028330	-2.156265	0	1980.0	1
26	26	44.0	0.793230	0.445207	1	1934.0	1
27	27	52.0	-0.028330	0.445207	1	1935.0	1
28	28	44.0	-0.849889	-0.518301	0	1932.0	1

In [ ]: