Q1 Assign a type to each of the following features manually as well as using a code if possible: (a) Job Title, (b) salary, (c ) company size

```
import pandas as pd
import numpy as np
import statistics as st
data = pd.read_csv('/content/job_salaries.csv', encoding = "windows-1252")

work_year = data['work_year']
work_year_mean = work_year.mean()
work_year = work_year.fillna(work_year_mean)
data['work_year'] = work_year

experience_level = data['experience_level']
print(experience_level.mode(),"\n")
experience_level = experience_level.fillna('SE')
data['experience_level'] = experience_level

employment_type = data['employment_type']
print(employment_type.mode(),'\n')
employment_type = employment_type.fillna('FT')
data['employment_type'] = employment_type

job_title = data['job_title']
print(job_title.mode(),'\n')
job_title = job_title.fillna('Data Scientist')
data['job_title'] = job_title

salary = data['salary']
salary_mean = salary.mean()
salary = salary.fillna(salary_mean)
data['salary'] = salary

salary_currency = data['salary_currency']
print(salary_currency.mode(),'\n')
salary_currency = salary_currency.fillna('USD')
data['salary_currency'] = salary_currency

salary_in_usd = data['salary_in_usd']
salary_in_usd_mean = salary_in_usd.mean()
salary_in_usd = salary_in_usd.fillna(salary_in_usd_mean)
data['salary_in_usd'] = salary_in_usd

employee_residence = data['employee_residence']
print(employee_residence.mode(),'\n')
employee_residence = employee_residence.fillna('US')
data['employee_residence'] = employee_residence

remote_ratio = data['remote_ratio']
```

```
remote_ratio_mean = remote_ratio.mean()
remote_ratio = remote_ratio.fillna(remote_ratio_mean)
data['remote_ratio'] = remote_ratio

company_location = data['company_location']
print(company_location.mode(),'\n')
company_location = company_location.fillna('US')
data['company_location'] = company_location

company_size = data['company_size']
print(company_size.mode(),'\n')
company_size = company_size.fillna('M')
data['company_size'] = company_size

print(data)
print(data.info())
```

```
0    SE
dtype: object

0    FT
dtype: object

0    Data Scientist
dtype: object

0    USD
dtype: object

0    US
dtype: object

0    US
dtype: object

0    M
dtype: object

     Unnamed: 0  work_year experience_level employment_type  \
0             0     2020.0               MI              FT
1             1     2020.0               SE              FT
2             2     2020.0               SE              FT
3             3     2020.0               MI              FT
4             4     2020.0               SE              FT
..          ...        ...              ...             ...
602         602     2022.0               SE              FT
603         603     2022.0               SE              FT
604         604     2022.0               SE              FT
605         605     2022.0               SE              FT
606         606     2022.0               MI              FT

                     job_title    salary salary_currency  salary_in_usd  \
0               Data Scientist   70000.0             EUR        79833.0
1    Machine Learning Scientist  260000.0             USD       260000.0
```

```
2             Big Data Engineer   85000.0             GBP       109024.0
3          Product Data Analyst   20000.0             USD        20000.0
4     Machine Learning Engineer  150000.0             USD       150000.0
..                          ...       ...             ...            ...
602               Data Engineer  154000.0             USD       154000.0
603               Data Engineer  126000.0             USD       126000.0
604                Data Analyst  129000.0             USD       129000.0
605                Data Analyst  150000.0             USD       150000.0
606                AI Scientist  200000.0             USD       200000.0

    employee_residence  remote_ratio company_location company_size
0                   DE             0               DE            L
1                   JP             0               JP            S
2                   GB            50               GB            M
3                   HN             0               HN            S
4                   US            50               US            M
..                 ...           ...              ...          ...
602                 US           100               US            M
603                 US           100               US            M
604                 US             0               US            M
```

Q2) Write a function to handle missing values in the dataset (e.g., any NA, NaN values), and demonstrate/discuss its functioning in the report.

```
import pandas as pd
data = pd.read_csv("/content/job_salaries.csv")

a = data['job_title']
print("Data type of job_title is",a.dtype)

b = data['salary']
print("Data type of salary is",b.dtype)

c = data['company_size']
print("Data type of company_size is",c.dtype)
```

```
Data type of job_title is object
Data type of salary is float64
Data type of company_size is object
```

Q3) Write a function to reduce noise (any error) in individual attributes and demonstrate/discuss its functioning in the report.

```
import pandas as pd
data = pd.read_csv("/content/job_salaries.csv")

work_year = data['work_year']
work_year_mean = work_year.mean()
work_year = work_year.fillna(work_year_mean)
```

```
data['work_year'] = work_year

salary = data['salary']
salary_mean = salary.mean()
salary = salary.fillna(salary_mean)
data['salary'] = salary

salary_in_usd = data['salary_in_usd']
salary_in_usd_mean = salary_in_usd.mean()
salary_in_usd = salary_in_usd.fillna(salary_in_usd_mean)
data['salary_in_usd'] = salary_in_usd

remote_ratio = data['remote_ratio']
remote_ratio_mean = remote_ratio.mean()
remote_ratio = remote_ratio.fillna(remote_ratio_mean)
data['remote_ratio'] = remote_ratio

print(data)
print(data.info())
```

```
     Unnamed: 0  work_year experience_level employment_type  \
0             0     2020.0               MI              FT
1             1     2020.0               SE              FT
2             2     2020.0               SE              FT
3             3     2020.0               MI              FT
4             4     2020.0               SE              FT
..          ...        ...              ...             ...
602         602     2022.0               SE              FT
603         603     2022.0               SE              FT
604         604     2022.0               SE              FT
605         605     2022.0               SE              FT
606         606     2022.0               MI              FT

                    job_title    salary salary_currency  salary_in_usd  \
0              Data Scientist   70000.0             EUR        79833.0
1    Machine Learning Scientist  260000.0             USD       260000.0
2             Big Data Engineer   85000.0             GBP       109024.0
3          Product Data Analyst   20000.0             USD        20000.0
4     Machine Learning Engineer  150000.0             USD       150000.0
..                          ...       ...             ...            ...
602              Data Engineer  154000.0             USD       154000.0
603              Data Engineer  126000.0             USD       126000.0
604               Data Analyst  129000.0             USD       129000.0
605               Data Analyst  150000.0             USD       150000.0
606               AI Scientist  200000.0             USD       200000.0

    employee_residence  remote_ratio company_location company_size
0                   DE             0               DE           L
1                   JP             0               JP           S
2                   GB            50               GB           M
3                   HN             0               HN           S
4                   US            50               US         NaN
..                 ...           ...              ...         ...
602                 US           100               US           M
```

```
603                US           100               US           M
604                US             0               US           M
605                US           100               US           M
606                IN           100               US           L

[607 rows x 12 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 607 entries, 0 to 606
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Unnamed: 0          607 non-null    int64
 1   work_year           607 non-null    float64
 2   experience_level    595 non-null    object
 3   employment_type     581 non-null    object
 4   job_title           583 non-null    object
 5   salary              607 non-null    float64
 6   salary_currency     604 non-null    object
 7   salary_in_usd       607 non-null    float64
 8   employee_residence  607 non-null    object
 9   remote_ratio        607 non-null    int64
 10  company_location    607 non-null    object
 11  company_size        571 non-null    object
dtypes: float64(3)  int64(2)  object(7)
```

Q4) Write a function to encode all the categorical features in the dataset according to the type of variable jointly, and demonstrate/discuss its functioning in the report.

```
import pandas as pd
import numpy as np
Super_store = pd.read_csv('job_salaries.csv')
df = pd.DataFrame(Super_store)
df.info()

print('/n /n /n /n')


from sklearn.preprocessing import LabelEncoder
lbcode = LabelEncoder()
def encoding(n):
  for i in n.columns:
      if n[i].dtypes == "object":
        n[i] = lbcode.fit_transform(n[i])
  return n
encoding(df)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 607 entries, 0 to 606
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Unnamed: 0          607 non-null    int64  
 1   work_year           606 non-null    float64
 2   experience_level    595 non-null    object 
 3   employment_type     581 non-null    object 
 4   job_title           583 non-null    object 
 5   salary              580 non-null    float64
 6   salary_currency     604 non-null    object 
 7   salary_in_usd       605 non-null    float64
 8   employee_residence  607 non-null    object 
 9   remote_ratio        607 non-null    int64  
 10  company_location    607 non-null    object 
 11  company_size        571 non-null    object 
dtypes: float64(3), int64(2), object(7)
memory usage: 57.0+ KB
/n /n /n /n
```

| | Unnamed: 0 | work_year | experience_level | employment_type | job_title | salary |
|---|---|---|---|---|---|---|
| **0** | 0 | 2020.0 | 2 | 2 | 22 | 70000.0 |
| **1** | 1 | 2020.0 | 3 | 2 | 41 | 260000.0 |
| **2** | 2 | 2020.0 | 3 | 2 | 7 | 85000.0 |
| **3** | 3 | 2020.0 | 2 | 2 | 47 | 20000.0 |
| **4** | 4 | 2020.0 | 3 | 2 | 38 | 150000.0 |
| **...** | ... | ... | ... | ... | ... | ... |
| **602** | 602 | 2022.0 | 3 | 2 | 17 | 154000.0 |
| **603** | 603 | 2022.0 | 3 | 2 | 17 | 126000.0 |
| **604** | 604 | 2022.0 | 3 | 2 | 12 | 129000.0 |

Q5

Write a function to normalize / scale the features either individually or jointly, and demonstrate/discuss its functioning in the report.

```
from sklearn.preprocessing import StandardScaler
import pandas as pd
import numpy as np
Super_store = pd.read_csv('/content/job_salaries.csv', encoding = "windows-1252")
df = pd.DataFrame(Super_store)
def normalize(n):
  scaler = StandardScaler()
  X = df.iloc[:,[n]]
```

```
  scaler.fit(X)
  data_set=pd.DataFrame(scaler.transform(X))
  return data_set
normalize(5)
```

| | 0 |
|---|---|
| **0** | -0.167715 |
| **1** | -0.047288 |
| **2** | -0.158207 |
| **3** | -0.199406 |
| **4** | -0.117009 |
| **...** | ... |
| **602** | -0.114473 |
| **603** | -0.132220 |
| **604** | -0.130319 |
| **605** | -0.117009 |
| **606** | -0.085317 |

607 rows × 1 columns

Q6) Write a function to create a random split of the data into three subsets (train, validation and test sets) in the ratio of 70:20:10 respectively. (a) Using numpy operation

```
import numpy as np

def slice(x):
  index_array = np.random.randint(df.shape[0], size=int(0.7*df, shape[0]))
  return x.iloc[index_array]
slice(df)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/ops/array_ops.py in
_na_arithmetic_op(left, right, op, is_cmp)
    165     try:
--> 166         result = func(left, right)
    167     except TypeError:

 15 frames 

TypeError: can't multiply sequence by non-int of type 'float'
```

```
from sklearn.model_selection import train_test_split
df_train, df_test= train_test_split(df, test_size=0.3, random_state = 2021)
df_train
```

| | Unnamed: 0 | work_year | experience_level | employment_type | job_title | salary |
|---|---|---|---|---|---|---|
| **150** | 150 | 2021.0 | SE | FT | Director of Data Science | 168000.0 |
| **314** | 314 | 2022.0 | MI | FT | Data Engineer | NaN |
| **78** | 78 | 2021.0 | MI | CT | ML Engineer | 270000.0 |
| **566** | 566 | 2022.0 | SE | FT | Data Analyst | 170000.0 |
| **491** | 491 | 2022.0 | MI | FT | Principal Data Analyst | 75000.0 |
| **...** | ... | ... | ... | ... | ... | ... |
| **44** | 44 | 2020.0 | MI | FT | Data Engineer | 88000.0 |
| | | | | | Machine | |

0s completed at 11:48 AM