DATA PREPROCESSING

1. Data Preprocessing is the process of transforming raw data into a format that is more suitable for analysis and modeling..
2. Data preprocessing involves cleaning the data,Normalization,Feature Engineering tasks..

In [2]:

```
#importing modules
import numpy as np
#imported numpy as np
import pandas as pd
#imported pandas module as pd
import matplotlib.pyplot as plt
#imported matplotlib.pyplot as plt alias
import seaborn as sns
#imported seaborn library as sns
import scipy
#imported scientific python module
import sklearn
#imported sklearn
import sklearn.preprocessing
#imported preprocessing from sklearn library
```

In [3]:

```python
#Loading the dataframe using pandas library
df = pd.read_excel('expenses.xlsx')
#df is the variable to store the expenses dataframe
df
#viewing the dataframe
```

Out[3]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain
0	39	Self-emp-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	15024
1	20	Private	Some-college	10	Never-married	Other-service	Own-child	White	Male	0
2	50	Private	Doctorate	16	Married-civ-spouse	Prof-specialty	Husband	White	Male	0
3	38	State-gov	HS-grad	9	Married-civ-spouse	Prof-specialty	Wife	White	Female	0
4	23	Local-gov	Bachelors	13	Never-married	Prof-specialty	Own-child	White	Female	0
...	...	...	...	...	...	...	...	...	...	...
4995	38	Private	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0
4996	26	Private	Some-college	10	Never-married	Tech-support	Own-child	White	Female	0
4997	20	Private	11th	7	Never-married	Transport-moving	Own-child	White	Male	0
4998	24	Private	HS-grad	9	Married-civ-spouse	Craft-repair	Husband	White	Male	0
4999	40	Private	HS-grad	9	Divorced	Craft-repair	Not-in-family	White	Male	0

5000 rows × 14 columns

In [4]:
```python
df.head()
#getting the 5 top records from dataframe
```

Out[4]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	cap
0	39	Self-emp-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	15024	
1	20	Private	Some-college	10	Never-married	Other-service	Own-child	White	Male	0	
2	50	Private	Doctorate	16	Married-civ-spouse	Prof-specialty	Husband	White	Male	0	
3	38	State-gov	HS-grad	9	Married-civ-spouse	Prof-specialty	Wife	White	Female	0	
4	23	Local-gov	Bachelors	13	Never-married	Prof-specialty	Own-child	White	Female	0	

In [6]:
```python
df.columns
#returns the column names of the dataframe
```

Out[6]:
```
Index(['age', 'workclass', 'education', 'education-num', 'marital-status',
       'occupation', 'relationship', 'race', 'sex', 'capital-gain',
       'capital-loss', 'hours-per-week', 'native-country', 'Expense'],
      dtype='object')
```

In [9]:
```python
#info function is used to get a concise summary of the dataframe
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 14 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   age             5000 non-null   int64 
 1   workclass       5000 non-null   object
 2   education       5000 non-null   object
 3   education-num   5000 non-null   int64 
 4   marital-status  5000 non-null   object
 5   occupation      5000 non-null   object
 6   relationship    5000 non-null   object
 7   race            5000 non-null   object
 8   sex             5000 non-null   object
 9   capital-gain    5000 non-null   int64 
 10  capital-loss    5000 non-null   int64 
 11  hours-per-week  5000 non-null   int64 
 12  native-country  5000 non-null   object
 13  Expense         5000 non-null   object
dtypes: int64(5), object(9)
memory usage: 547.0+ KB
```

In [11]:

```
df.describe()
#describe function is used to get some basic stastical details like mean,min,max,cou
```

Out[11]:

	age	education-num	capital-gain	capital-loss	hours-per-week
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.000000
mean	38.656000	10.065000	1104.080000	90.032800	40.566200
std	13.698292	2.558141	7579.674371	404.168991	12.154191
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	48.000000	12.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	3004.000000	99.000000

In [20]:

```
df.shape
#the shape function returns the Number of Rows and columns from the dataframe in tup
```

Out[20]:

```
(5000, 14)
```

Cleaning Data

1. Cleaning data: Cleaning data involves removal of duplicate values, handling the missing values and removal of outliers....
2. The goal of cleaning data is that the data is accurate,complete,consistent

In [27]:

```
#lets count the missing values or NaN values in the dataframe
df.isnull().sum()
#isnull function is used to check whether the the record is null or not
#the sum function returns the sum of the null values in the each column respectively
```

Out[27]:

```
age               0
workclass         0
education         0
education-num     0
marital-status    0
occupation        0
relationship      0
race              0
sex               0
capital-gain      0
capital-loss      0
hours-per-week    0
native-country    0
Expense           0
dtype: int64
```

In [34]:

```
#we can see this in percentage in respective columns
missing_values = df.isnull().sum()
#check for the missing values and sort the values in desc
total = df.isnull().sum().sort_values(ascending = False)
percent = ((df.isnull().sum()/df.shape[0]*100))
percent = percent.sort_values(ascending = False)
#concetanating the total missing values
missing_data = pd.concat([total,percent],axis = 1,
                        keys = ['Total Missing Values','Percentage of Missing values
#adding the Data types
missing_data['Data(Dtypes)'] = df[missing_data.index].dtypes
missing_data
#viewing the missing_data
```

Out[34]:

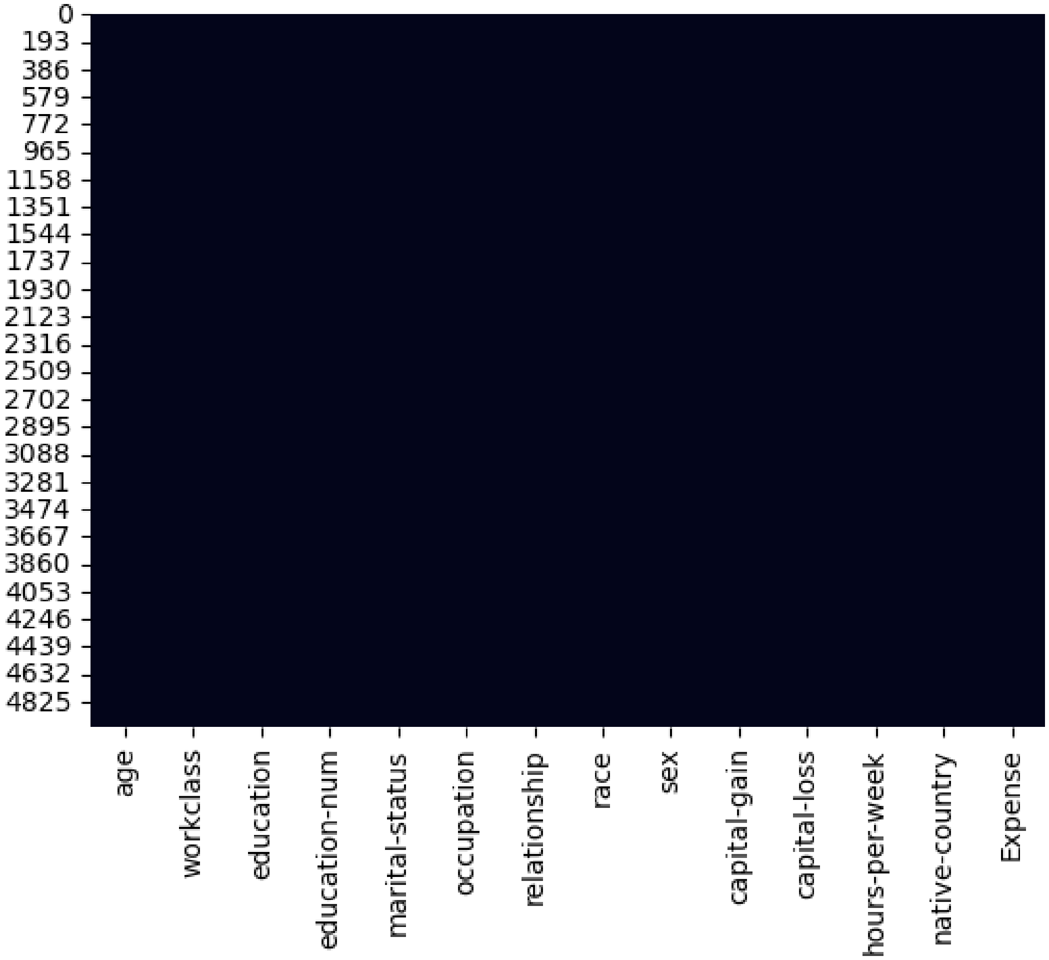
	Total Missing Values	Percentage of Missing values	Data(Dtypes)
age	0	0.0	int64
workclass	0	0.0	object
education	0	0.0	object
education-num	0	0.0	int64
marital-status	0	0.0	object
occupation	0	0.0	object
relationship	0	0.0	object
race	0	0.0	object
sex	0	0.0	object
capital-gain	0	0.0	int64
capital-loss	0	0.0	int64
hours-per-week	0	0.0	int64
native-country	0	0.0	object
Expense	0	0.0	object

In [35]:

```
#representing the null values in heatmap
import seaborn as sns
#imported seaborn library as sns
sns.heatmap(df.isnull(),cbar = False)
```

Out[35]:

```
<AxesSubplot:>
```



In [25]:

```
#As there No Null Values in the entire dataset...
#we don't need to handle the missing values
```

Removal of Outliers

In [37]:

```
#lets divide the numerical columns into another dataset
df_num = df.select_dtypes(include = [np.number])
df_num
#now df_num is the dataset which contains only the Numerical columns of the dataframe
```

Out[37]:

	age	education-num	capital-gain	capital-loss	hours-per-week
0	39	13	15024	0	50
1	20	10	0	0	40
2	50	16	0	1902	65
3	38	9	0	0	40
4	23	13	0	0	60
...	...	...	...	...	...
4995	38	9	0	0	40
4996	26	10	0	0	40
4997	20	7	0	0	60
4998	24	9	0	0	60
4999	40	9	0	0	45

5000 rows × 5 columns

In [44]:

```
df_num.columns
#the columns of Numerical columns
```

Out[44]:

```
Index(['age', 'education-num', 'capital-gain', 'capital-loss',
       'hours-per-week'],
      dtype='object')
```

In [42]:
```
#lets divide the Categorical columns from the dataset for further reference
from warnings import filterwarnings
#imported filterwarnings from warnings
filterwarnings('ignore')
#ignored the filterwarnings
df_cat = df.select_dtypes(include = [np.object])
#df_cat variable holds the Categorical columns from the dataframe
df_cat
#now df_cat is a separate dataset which holds the Categorical columns
```

Out[42]:

	workclass	education	marital-status	occupation	relationship	race	sex	native-country	Expense
0	Self-emp-inc	Bachelors	Married-civ-spouse	Exec-managerial	Husband	White	Male	United-States	>50K
1	Private	Some-college	Never-married	Other-service	Own-child	White	Male	United-States	<=50K
2	Private	Doctorate	Married-civ-spouse	Prof-specialty	Husband	White	Male	United-States	>50K
3	State-gov	HS-grad	Married-civ-spouse	Prof-specialty	Wife	White	Female	United-States	>50K
4	Local-gov	Bachelors	Never-married	Prof-specialty	Own-child	White	Female	United-States	<=50K
...	...	...	...	...	...	...	...	...	...
4995	Private	HS-grad	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	United-States	<=50K
4996	Private	Some-college	Never-married	Tech-support	Own-child	White	Female	United-States	<=50K
4997	Private	11th	Never-married	Transport-moving	Own-child	White	Male	United-States	<=50K
4998	Private	HS-grad	Married-civ-spouse	Craft-repair	Husband	White	Male	Mexico	>50K
4999	Private	HS-grad	Divorced	Craft-repair	Not-in-family	White	Male	United-States	<=50K

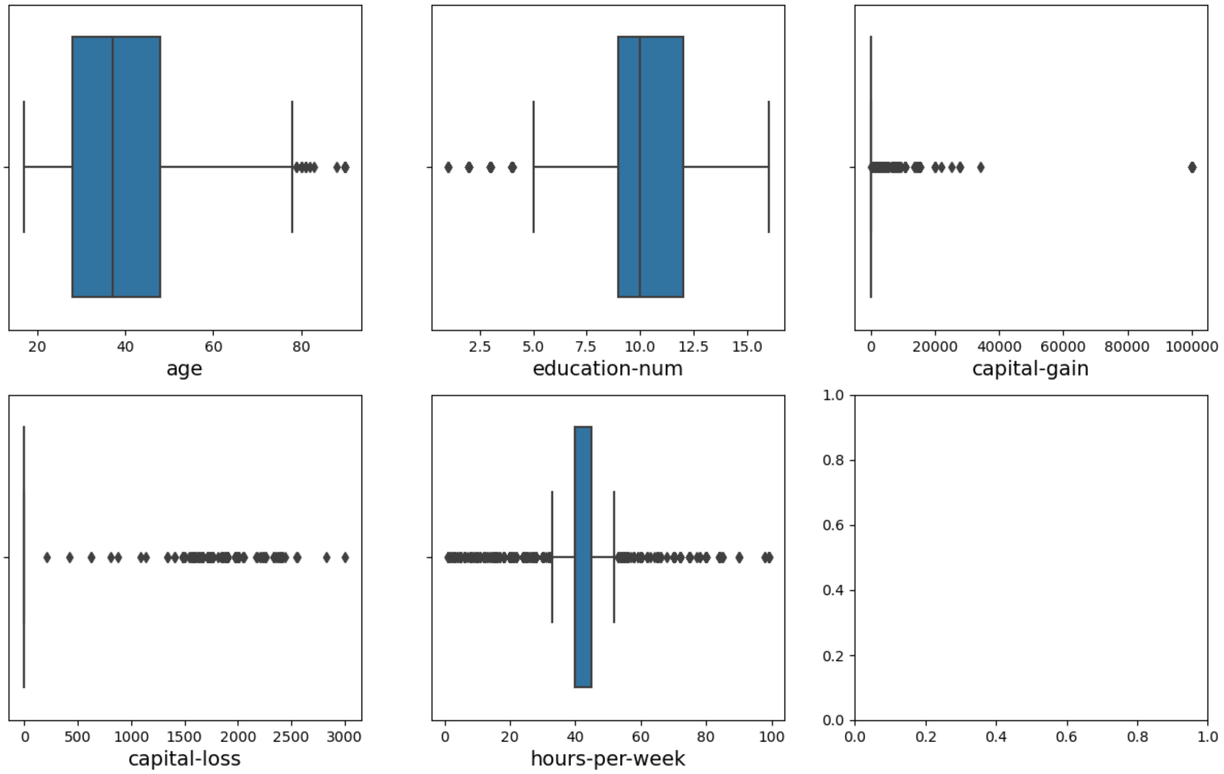
5000 rows × 9 columns

In [43]:
```
df_cat.columns
#the columns of Categorical dataset
```

Out[43]:
```
Index(['workclass', 'education', 'marital-status', 'occupation',
       'relationship', 'race', 'sex', 'native-country', 'Expense'],
      dtype='object')
```

In [59]:

```
#Now lets find the Outliers for the Numerical columns
#for this we need to use subplots() function
import matplotlib.pyplot as plt
import seaborn as sns
#import matplotlib and seaborn as plt and sns respectively
fig,ax = plt.subplots(2,3,figsize = (15,9))
for variable,subplot in zip(df_num.columns,ax.flatten()):
    z = sns.boxplot(x = df_num[variable],orient = 'h',whis = 1.5,ax = subplot)
    z.set_xlabel(variable,fontsize = 14)
```



In [60]:

```
#here we have seen a major outliers in the Numerical columns
#we need to remove the outliers using IQR method
#IQR means Inter-quartile Range
```

Removing Outliers using IQR Method

In [61]:

```
Q1 = df_num.quantile(0.25)
Q3 = df_num.quantile(0.75)
#we know that to find IQR
#The formula is
IQR = Q3 - Q1
IQR
#We got the IQR for respective columns
```

Out[61]:

```
age               20.0
education-num      3.0
capital-gain       0.0
capital-loss       0.0
hours-per-week     5.0
dtype: float64
```

In [62]:

```
# ~ : select all the rows which doesn't satisfy the condition
```

In [66]:

```
df_cleaned = df[~((df<(Q1-1.5*IQR))|(df>Q3+1.5*IQR)).any(axis = 1)]
#removed the outliers using the formula and
#storing the cleaned data in df_cleaned variable
```

In [67]:

```
df_cleaned
```

Out[67]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain
1	20	Private	Some-college	10	Never-married	Other-service	Own-child	White	Male	0
3	38	State-gov	HS-grad	9	Married-civ-spouse	Prof-specialty	Wife	White	Female	0
6	58	Private	Bachelors	13	Married-civ-spouse	Adm-clerical	Husband	White	Male	0
7	66	Private	HS-grad	9	Separated	Machine-op-inspct	Not-in-family	Black	Male	0
8	39	Self-emp-inc	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0
...	...	...	...	...	...	...	...	...	...	...
4993	28	Private	HS-grad	9	Never-married	Handlers-cleaners	Unmarried	White	Male	0
4994	35	Private	HS-grad	9	Divorced	Other-service	Not-in-family	White	Female	0
4995	38	Private	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0
4996	26	Private	Some-college	10	Never-married	Tech-support	Own-child	White	Female	0
4999	40	Private	HS-grad	9	Divorced	Craft-repair	Not-in-family	White	Male	0

3032 rows × 14 columns

In [68]:

```
df_cleaned.shape
```

Out[68]:

```
(3032, 14)
```

In [74]:

```
df_cleaned.columns
```

Out[74]:

```
Index(['age', 'workclass', 'education', 'education-num', 'marital-status',
       'occupation', 'relationship', 'race', 'sex', 'capital-gain',
       'capital-loss', 'hours-per-week', 'native-country', 'Expense'],
      dtype='object')
```

```
In [79]: #lets divide the numerical columns into another dataset
         df_num = df_cleaned.select_dtypes(include = [np.number])
         df_num
```

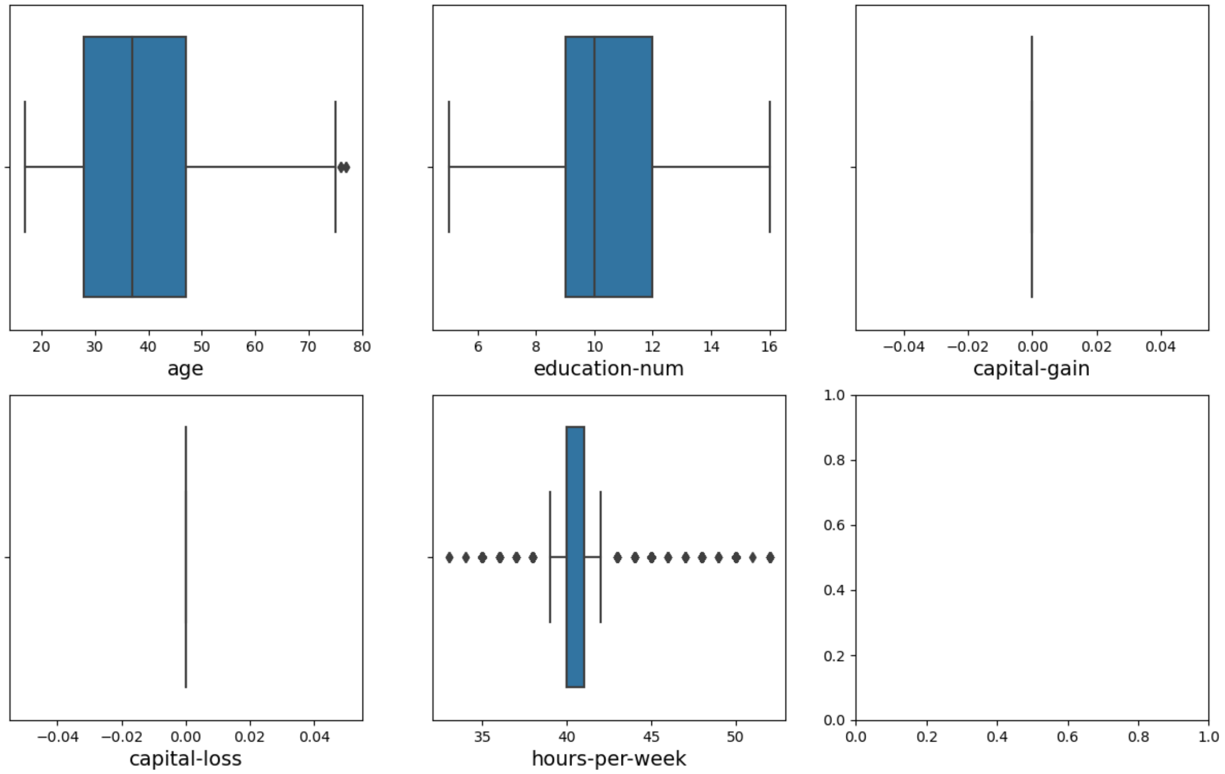
Out[79]:

	age	education-num	capital-gain	capital-loss	hours-per-week
1	20	10	0	0	40
3	38	9	0	0	40
6	58	13	0	0	40
7	66	9	0	0	40
8	39	9	0	0	40
...	...	...	...	...	...
4993	28	9	0	0	40
4994	35	9	0	0	35
4995	38	9	0	0	40
4996	26	10	0	0	40
4999	40	9	0	0	45

3032 rows × 5 columns

In [86]:

```
#Now lets find the Outliers for the Numerical columns
#for this we need to use subplots() function
import matplotlib.pyplot as plt
import seaborn as sns
#import matplotlib and seaborn as plt and sns respectively
fig,ax = plt.subplots(2,3,figsize = (15,9))
for variable,subplot in zip(df_num.columns,ax.flatten()):
    z = sns.boxplot(x = df_num[variable],orient = 'h',whis = 1.5,ax = subplot)
    z.set_xlabel(variable,fontsize = 14)
```



In [96]:

```

Q1 = df_num.quantile(0.25)
Q3 = df_num.quantile(0.75)
#we know that to find IQR
#The formula is
IQR = Q3 - Q1
IQR
#We got the IQR for respective columns
```

Out[96]:

```
age               19.0
education-num      3.0
capital-gain       0.0
capital-loss       0.0
hours-per-week     1.0
dtype: float64
```

In [97]:

```
df_ultra_cleaned = df_cleaned[~((df_cleaned<(Q1-1.5*IQR))|(df_cleaned>Q3+1.5*IQR)).a
#removed the outliers using the formula and
#storing the cleaned data in df_cleaned variable
```

In [98]:
```
df_ultra_cleaned
```

Out[98]:

	age	workclass	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain
1	20	Private	Some-college	10	Never-married	Other-service	Own-child	White	Male	0
3	38	State-gov	HS-grad	9	Married-civ-spouse	Prof-specialty	Wife	White	Female	0
6	58	Private	Bachelors	13	Married-civ-spouse	Adm-clerical	Husband	White	Male	0
7	66	Private	HS-grad	9	Separated	Machine-op-inspct	Not-in-family	Black	Male	0
8	39	Self-emp-inc	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0
...	...	...	...	...	...	...	...	...	...	...
4990	32	Private	HS-grad	9	Married-civ-spouse	Craft-repair	Husband	White	Male	0
4991	34	Private	Bachelors	13	Never-married	Sales	Own-child	Black	Female	0
4993	28	Private	HS-grad	9	Never-married	Handlers-cleaners	Unmarried	White	Male	0
4995	38	Private	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0
4996	26	Private	Some-college	10	Never-married	Tech-support	Own-child	White	Female	0

2000 rows × 14 columns

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

Out[99]:
```
(2000, 14)
```

In [101]:

```
#lets divide the numerical columns into another dataset
df_num = df_ultra_cleaned.select_dtypes(include = [np.number])
df_num
```

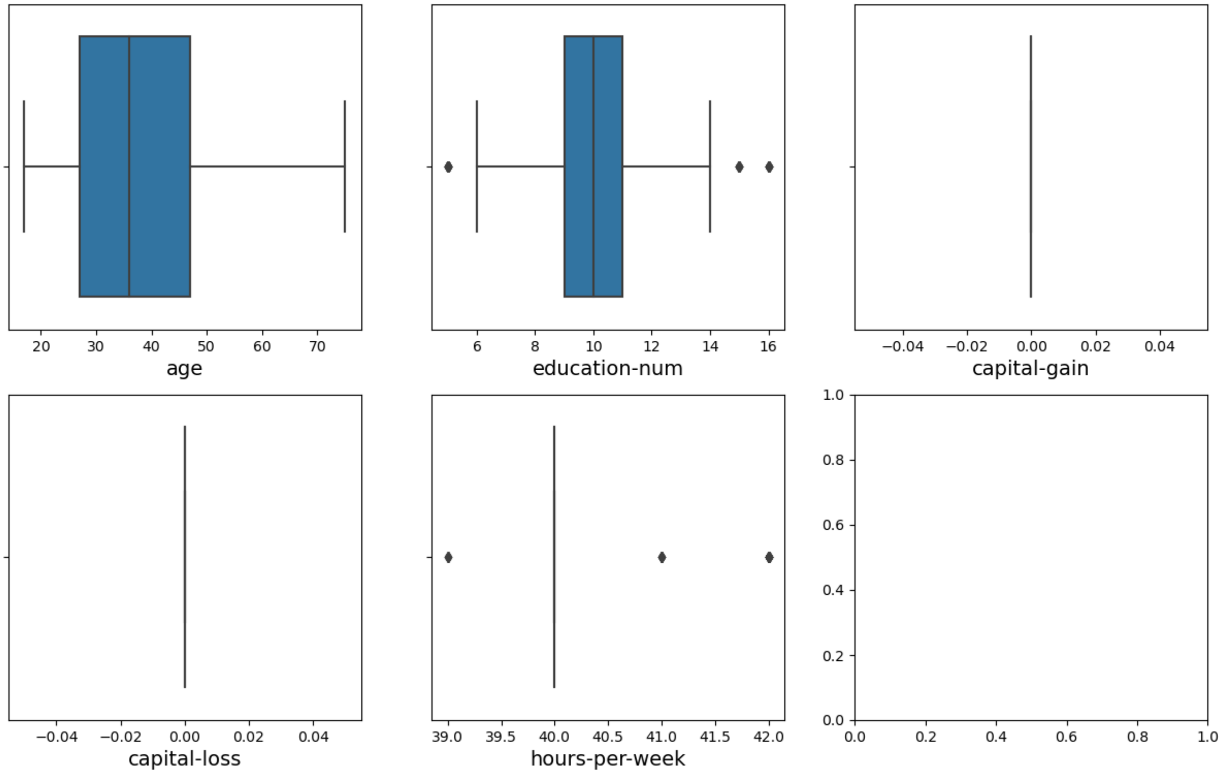
Out[101]:

	age	education-num	capital-gain	capital-loss	hours-per-week
1	20	10	0	0	40
3	38	9	0	0	40
6	58	13	0	0	40
7	66	9	0	0	40
8	39	9	0	0	40
...	...	...	...	...	...
4990	32	9	0	0	40
4991	34	13	0	0	40
4993	28	9	0	0	40
4995	38	9	0	0	40
4996	26	10	0	0	40

2000 rows × 5 columns

In [102]:

```
#Now lets find the Outliers for the Numerical columns
#for this we need to use subplots() function
import matplotlib.pyplot as plt
import seaborn as sns
#import matplotlib and seaborn as plt and sns respectively
fig,ax = plt.subplots(2,3,figsize = (15,9))
for variable,subplot in zip(df_num.columns,ax.flatten()):
    z = sns.boxplot(x = df_num[variable],orient = 'h',whis = 1.5,ax = subplot)
    z.set_xlabel(variable,fontsize = 14)
```



In [ ]: