EDA, FE and Classification Model (Census Income Dataset)

Aryan Shriva

Linkedin: https://www.linkedin.com/in/aryan-shriva-0811/ (https://www.linkedin.com/in/aryan-shriva-0811/)

GitHub https://github.com/AryanShriva/Machine-Learning (https://github.com/AryanShriva/Machine-Learning)

For the code please check out my Machine Learning repository on GitHub

1. EDA and FE

1. Data Profiling
2. Stastical analysis
3. Graphical Analysis
4. Data Cleaning
5. Data Scaling

2. Logistic Regression Model

1. Linear Regression Model
2. Performance metrics for above model
3. Hyper-Parameter Tuning for above model

3. Support Vector Classifier Model

1. Support Vector Classifier Model
2. Performance metrics for above model
3. Hyper-Parameter Tuning for above model

Dataset: https://archive.ics.uci.edu/ml/datasets/Census+Income (https://archive.ics.uci.edu/ml/datasets/Census+Income)

1.0 Importing required libraries

In [5]:
```
### Pandas and Numpy
import pandas as pd
import numpy as np

### Visualisation libraries
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

### For Q-Q Plot
import scipy.stats as stats

### To ignore warnings
import warnings
warnings.filterwarnings('ignore')
```

2.0 Importing Dataset and Data Cleaning

In [6]:
```
### importing both train and test dataset
column_names=['age', 'workclass', 'fnlwgt', 'education', 'education_num', 'm
              'race', 'sex', 'capital_gain', 'capital_loss', 'hours_per_week'
datasetp1=pd.read_csv('adult.csv', names=column_names, header=None)
datasetp2=pd.read_csv('adult1.csv', names=column_names, header=None)
```

In [7]:
```
datasetp1.head()
```

Out[7]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	ra
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	Wh
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	Wh
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	Wh
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Bla
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Bla

In [8]:
```
datasetp2.head()
```

Out[8]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	ra
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Bla
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	Wh
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	Wh
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Bla
4	18	?	103497	Some-college	10	Never-married	?	Own-child	Wh

In [9]:
```
### joining both dataset and resetting index
dataset=pd.concat([datasetp1, datasetp2])

dataset.reset_index(inplace=True)
```

In [10]:
```
## dropping index feature as it is not required
dataset.drop('index', axis=1,inplace=True)
```

In [11]:
```
### getting shape of dataset
### dataset has 48842 records and 15 features
dataset.shape
```

Out[11]:
```
(48842, 15)
```

In [12]:
```
### getting column names
dataset.columns
```

Out[12]:
```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education_num',
       'marital_status', 'occupation', 'relationship', 'race', 'sex',
       'capital_gain', 'capital_loss', 'hours_per_week', 'native_country',
       'salary'],
      dtype='object')
```

In [13]:
```
dataset.head()
```

Out[13]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	ra
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	Wh
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	Wh
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	Wh
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Bla
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Bla

2.1 Dataset Information

Data Set Information:

Extraction was done by Barry Becker from the 1994 Census database. A set of reasonably clean records was extracted using the following conditions: ((AAGE>16) && (AGI>100) && (AFNLWGT>1)&& (HRSWK>0))

Prediction task is to determine whether a person makes over 50K a year.

Attribute Information:

Listing of attributes:

1. 50K, <=50K.

2. age: continuous.
3. workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
4. fnlwgt: continuous.
5. education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
6. education-num: continuous.
7. marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
8. occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
9. relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

10. race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
11. sex: Female, Male.
12. capital-gain: continuous.
13. capital-loss: continuous.
14. hours-per-week: continuous.
15. native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

In [14]:
```
### getting count of values in salary feature which is dependent feature

### Data cleaning is required as there is additional . which is creating 2 m
dataset.salary.value_counts()
```

Out[14]:
```
 <=50K     24720
 <=50K.    12435
 >50K       7841
 >50K.      3846
Name: salary, dtype: int64
```

In [15]:
```
### checking duplicates in dataset
dataset[dataset.duplicated()].shape
```

Out[15]: (29, 15)

In [16]:
```
### dropping duplicates in dataset
dataset.drop_duplicates(inplace=True)
```

In [17]:
```
### confirming the duplicates removal
dataset[dataset.duplicated()].shape
```

Out[17]: (0, 15)

In [18]:
```
### replacing . in salary feature
dataset['salary']=dataset['salary'].str.replace('<=50K.','<=50K', regex=True
dataset['salary']=dataset['salary'].str.replace('>50K.','>50K', regex=True)
```

In [19]:
```
### salary feature is clean and is divided into two categories

dataset.salary.value_counts()
```

Out[19]:
```
 <=50K    37128
 >50K     11685
Name: salary, dtype: int64
```

In [20]:
```
### Checking null values in dataset
dataset.isnull().sum()
```

Out[20]:
```
age               0
workclass         0
fnlwgt            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
salary            0
dtype: int64
```

In [21]:
```
### checking unique categories in categorical features and unique values in

for feature in dataset.columns:
    print("Feature '{}' has these {} unique values\n".format(feature, datase
```

```
Feature 'age' has these [39 50 38 53 28 37 49 52 31 42 30 23 32 40 34 25 43 54
35 59 56 19 20 45
 22 48 21 24 57 44 41 29 18 47 46 36 79 27 67 33 76 17 55 61 70 64 71 68
 66 51 58 26 60 90 75 65 77 62 63 80 72 74 69 73 81 78 88 82 83 84 85 86
 87 89] unique values

Feature 'workclass' has these [' State-gov' ' Self-emp-not-inc' ' Private' ' Fe
deral-gov' ' Local-gov'
 ' ?' ' Self-emp-inc' ' Without-pay' ' Never-worked'] unique values

Feature 'fnlwgt' has these [ 77516  83311 215646 ... 173449  89686 350977] uniq
ue values

Feature 'education' has these [' Bachelors' ' HS-grad' ' 11th' ' Masters' ' 9t
h' ' Some-college'
 ' Assoc-acdm' ' Assoc-voc' ' 7th-8th' ' Doctorate' ' Prof-school'
 ' 5th-6th' ' 10th' ' 1st-4th' ' Preschool' ' 12th'] unique values

Feature 'education_num' has these [13  9  7 14  5 10 12 11  4 16 15  3  6  2  1
8] unique values

Feature 'marital_status' has these [' Never-married' ' Married-civ-spouse' ' Di
vorced'
 ' Married-spouse-absent' ' Separated' ' Married-AF-spouse' ' Widowed'] unique
values

Feature 'occupation' has these [' Adm-clerical' ' Exec-managerial' ' Handlers-c
leaners' ' Prof-specialty'
 ' Other-service' ' Sales' ' Craft-repair' ' Transport-moving'
 ' Farming-fishing' ' Machine-op-inspct' ' Tech-support' ' ?'
 ' Protective-serv' ' Armed-Forces' ' Priv-house-serv'] unique values

Feature 'relationship' has these [' Not-in-family' ' Husband' ' Wife' ' Own-chi
ld' ' Unmarried'
 ' Other-relative'] unique values

Feature 'race' has these [' White' ' Black' ' Asian-Pac-Islander' ' Amer-Indian
-Eskimo' ' Other'] unique values

Feature 'sex' has these [' Male' ' Female'] unique values

Feature 'capital_gain' has these [ 2174     0 14084  5178  5013  2407 14344 150
24  7688 34095  4064  4386
  7298  1409  3674  1055  3464  2050  2176   594 20051  6849  4101  1111
  8614  3411  2597 25236  4650  9386  2463  3103 10605  2964  3325  2580
  3471  4865 99999  6514  1471  2329  2105  2885 25124 10520  2202  2961
 27828  6767  2228  1506 13550  2635  5556  4787  3781  3137  3818  3942
   914   401  2829  2977  4934  2062  2354  5455 15020  1424  3273 22040
  4416  3908 10566   991  4931  1086  7430  6497   114  7896  2346  3418
  3432  2907  1151  2414  2290 15831 41310  4508  2538  3456  6418  1848
  3887  5721  9562  1455  2036  1831 11678  2936  2993  7443  6360  1797
  1173  4687  6723  2009  6097  2653  1639 18481  7978  2387  5060  1264
```

```
  7262  1731  6612] unique values

Feature 'capital_loss' has these [   0 2042 1408 1902 1573 1887 1719 1762 1564
2179 1816 1980 1977 1876
 1340 2206 1741 1485 2339 2415 1380 1721 2051 2377 1669 2352 1672  653
 2392 1504 2001 1590 1651 1628 1848 1740 2002 1579 2258 1602  419 2547
 2174 2205 1726 2444 1138 2238  625  213 1539  880 1668 1092 1594 3004
 2231 1844  810 2824 2559 2057 1974  974 2149 1825 1735 1258 2129 2603
 2282  323 4356 2246 1617 1648 2489 3770 1755 3683 2267 2080 2457  155
 3900 2201 1944 2467 2163 2754 2472 1411 1429 3175 1510 1870 1911 2465
 1421] unique values

Feature 'hours_per_week' has these [40 13 16 45 50 80 30 35 60 20 52 44 15 25 3
8 43 55 48 58 32 70  2 22 56
 41 28 36 24 46 42 12 65  1 10 34 75 98 33 54  8  6 64 19 18 72  5  9 47
 37 21 26 14  4 59  7 99 53 39 62 57 78 90 66 11 49 84  3 17 68 27 85 31
 51 77 63 23 87 88 73 89 97 94 29 96 67 82 86 91 81 76 92 61 74 95 79 69] uniqu
e values

Feature 'native_country' has these [' United-States' ' Cuba' ' Jamaica' ' Indi
a' ' ?' ' Mexico' ' South'
 ' Puerto-Rico' ' Honduras' ' England' ' Canada' ' Germany' ' Iran'
 ' Philippines' ' Italy' ' Poland' ' Columbia' ' Cambodia' ' Thailand'
 ' Ecuador' ' Laos' ' Taiwan' ' Haiti' ' Portugal' ' Dominican-Republic'
 ' El-Salvador' ' France' ' Guatemala' ' China' ' Japan' ' Yugoslavia'
 ' Peru' ' Outlying-US(Guam-USVI-etc)' ' Scotland' ' Trinadad&Tobago'
 ' Greece' ' Nicaragua' ' Vietnam' ' Hong' ' Ireland' ' Hungary'
 ' Holand-Netherlands'] unique values

Feature 'salary' has these [' <=50K' ' >50K'] unique values
```

Observations

1. All categorical features have space and dash which requires cleaning.
2. Some categorical features like workclass, occupation, and native_country have ? as a value, it also requires cleaning.

In [22]:
```
### checking datatypes and null values in dataset
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 48813 entries, 0 to 48841
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   age             48813 non-null  int64 
 1   workclass       48813 non-null  object
 2   fnlwgt          48813 non-null  int64 
 3   education       48813 non-null  object
 4   education_num   48813 non-null  int64 
 5   marital_status  48813 non-null  object
 6   occupation      48813 non-null  object
 7   relationship    48813 non-null  object
 8   race            48813 non-null  object
 9   sex             48813 non-null  object
 10  capital_gain    48813 non-null  int64 
 11  capital_loss    48813 non-null  int64 
 12  hours_per_week  48813 non-null  int64 
 13  native_country  48813 non-null  object
 14  salary          48813 non-null  object
dtypes: int64(6), object(9)
memory usage: 6.0+ MB
```

3.0 Numerical and Categorical features

3.1 Categorical features

In [23]:
```
### Getting categorical features in dataset
categorical_features=[feature for feature in dataset.columns if dataset[feat
print(categorical_features)
```

```
['workclass', 'education', 'marital_status', 'occupation', 'relationship', 'rac
e', 'sex', 'native_country', 'salary']
```

In [24]:
```
### Getting count of each category in each categorical feature
for feature in categorical_features:
    print("Feature Name: {}\n{}\n".format(feature ,dataset[feature].value_co
```

```
Feature Name: workclass
 Private             33879
 Self-emp-not-inc     3861
 Local-gov            3136
 ?                    2799
 State-gov            1981
 Self-emp-inc         1694
 Federal-gov          1432
 Without-pay            21
 Never-worked           10
Name: workclass, dtype: int64

Feature Name: education
 HS-grad         15777
 Some-college    10869
 Bachelors        8020
 Masters          2656
 Assoc-voc        2060
 11th             1812
 Assoc-acdm       1601
 10th             1389
 7th-8th           954
 Prof-school       834
 9th               756
 12th              656
 Doctorate         594
 5th-6th           508
 1st-4th           245
 Preschool          82
Name: education, dtype: int64

Feature Name: marital_status
 Married-civ-spouse       22372
 Never-married            16098
 Divorced                  6630
 Separated                 1530
 Widowed                   1518
 Married-spouse-absent      628
 Married-AF-spouse           37
Name: marital_status, dtype: int64

Feature Name: occupation
 Prof-specialty       6167
 Craft-repair         6107
 Exec-managerial      6084
 Adm-clerical         5608
 Sales                5504
 Other-service        4919
 Machine-op-inspct    3019
 ?                    2809
 Transport-moving     2355
 Handlers-cleaners    2071
 Farming-fishing      1487
```

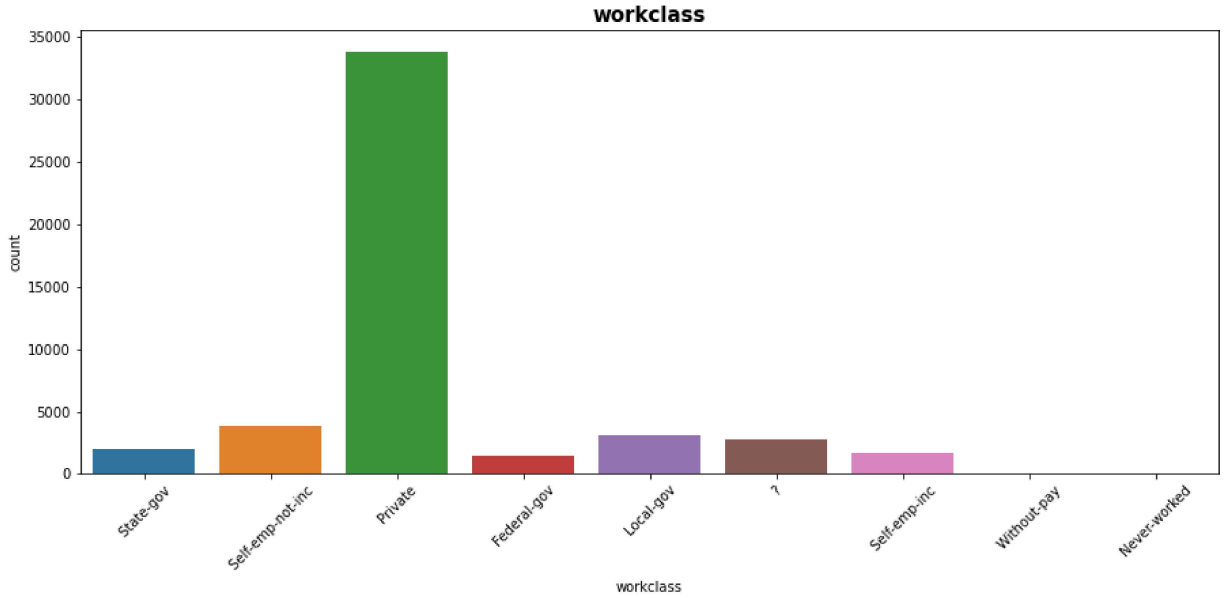
```
 Tech-support         1445
 Protective-serv       983
 Priv-house-serv       240
 Armed-Forces           15
Name: occupation, dtype: int64

Feature Name: relationship
 Husband           19709
 Not-in-family     12567
 Own-child          7576
 Unmarried          5124
 Wife               2331
 Other-relative     1506
Name: relationship, dtype: int64

Feature Name: race
 White                 41736
 Black                  4683
 Asian-Pac-Islander     1518
 Amer-Indian-Eskimo      470
 Other                   406
Name: race, dtype: int64

Feature Name: sex
 Male      32631
 Female    16182
Name: sex, dtype: int64

Feature Name: native_country
 United-States                 43810
 Mexico                          947
 ?                               856
 Philippines                     295
 Germany                         206
 Puerto-Rico                     184
 Canada                          182
 El-Salvador                     155
 India                           151
 Cuba                            138
 England                         127
 China                           122
 South                           115
 Jamaica                         106
 Italy                           105
 Dominican-Republic              103
 Japan                            92
 Poland                           87
 Guatemala                        86
 Vietnam                          86
 Columbia                         85
 Haiti                            75
 Portugal                         67
 Taiwan                           65
 Iran                             59
 Greece                           49
 Nicaragua                        49
 Peru                             46
```

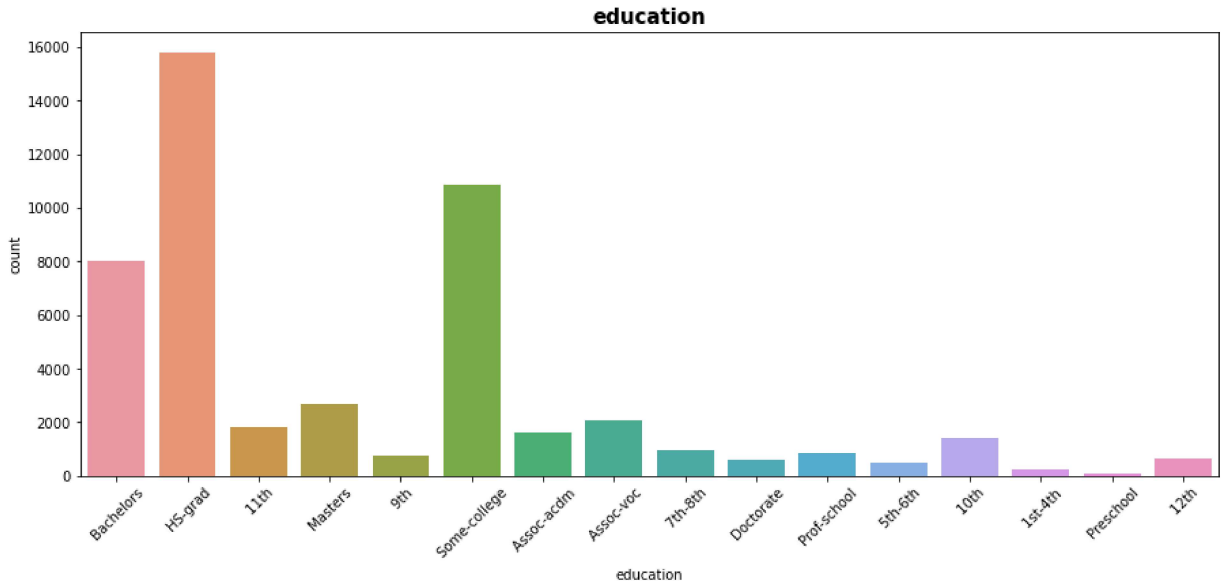
```
 Ecuador                         45
 France                          38
 Ireland                         37
 Hong                            30
 Thailand                        30
 Cambodia                        28
 Trinadad&Tobago                 27
 Laos                            23
 Yugoslavia                      23
 Outlying-US(Guam-USVI-etc)      23
 Scotland                        21
 Honduras                        20
 Hungary                         19
 Holand-Netherlands               1
Name: native_country, dtype: int64

Feature Name: salary
 <=50K    37128
 >50K     11685
Name: salary, dtype: int64
```

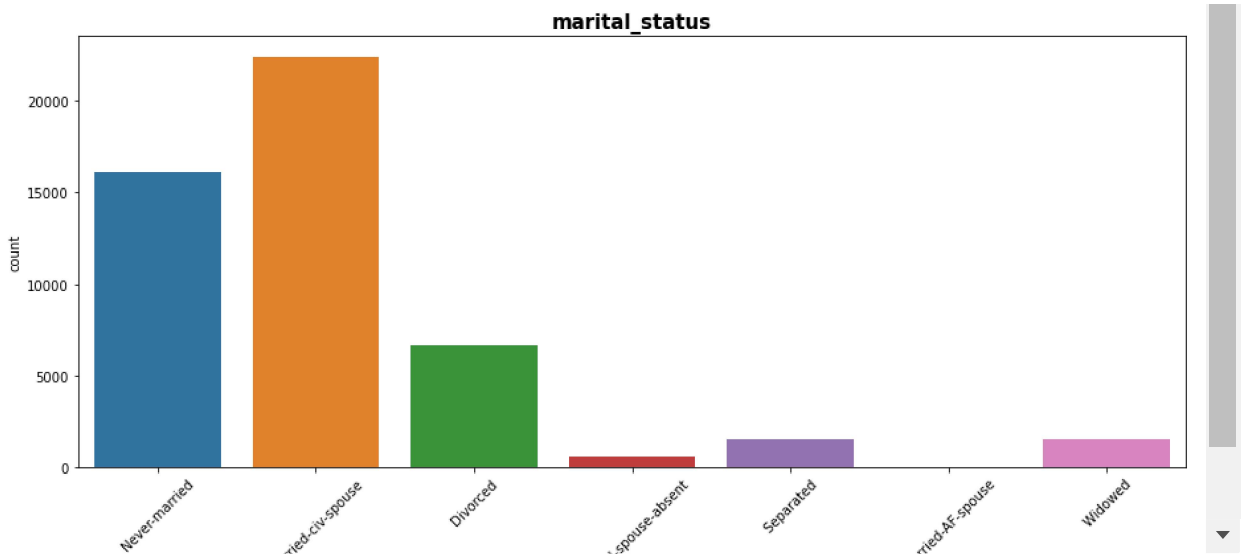
Observations

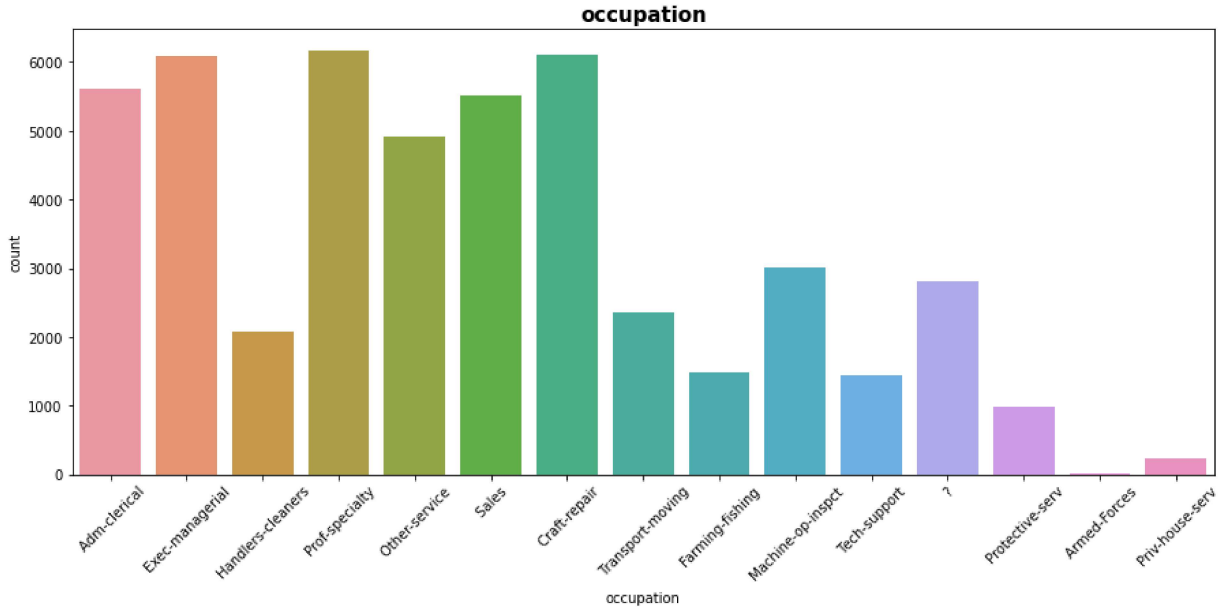
1. workclass has 2799 , occupation has 2809, and native_country has 856 ? as a value.

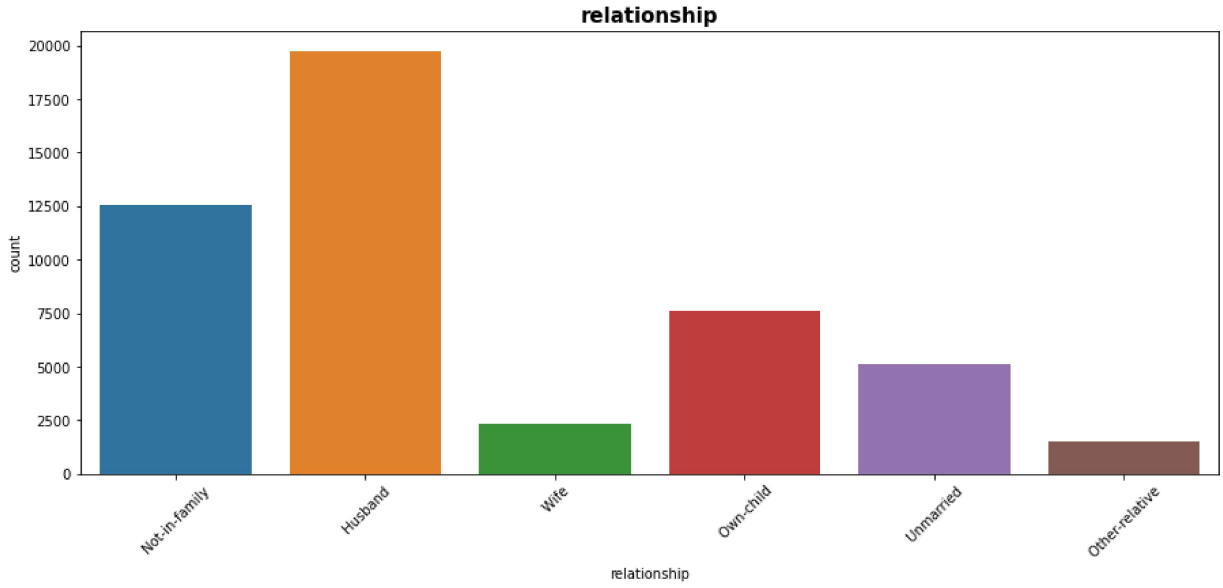
In [25]:
```
### Countplot to visualize the count of each category in each categorical fe
for feature in [feature for feature in categorical_features if feature not i
    plt.figure(figsize=(15,6))
    sns.countplot(data=dataset, x=feature)
    plt.title(feature, fontsize=15, weight='bold')
    plt.xticks(rotation=45)
    plt.show();
```

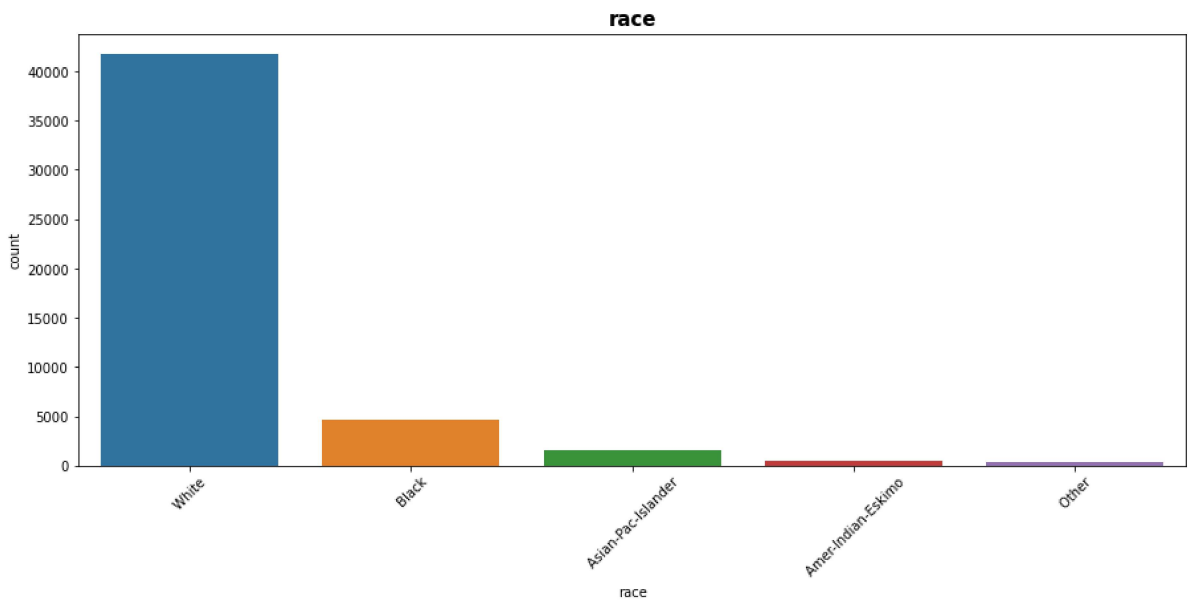


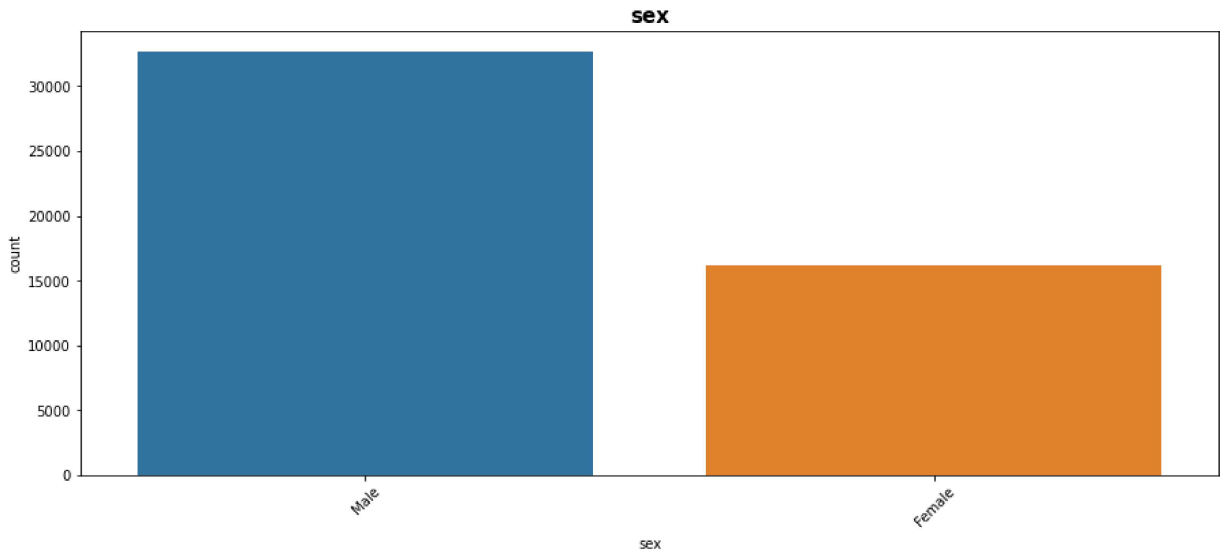


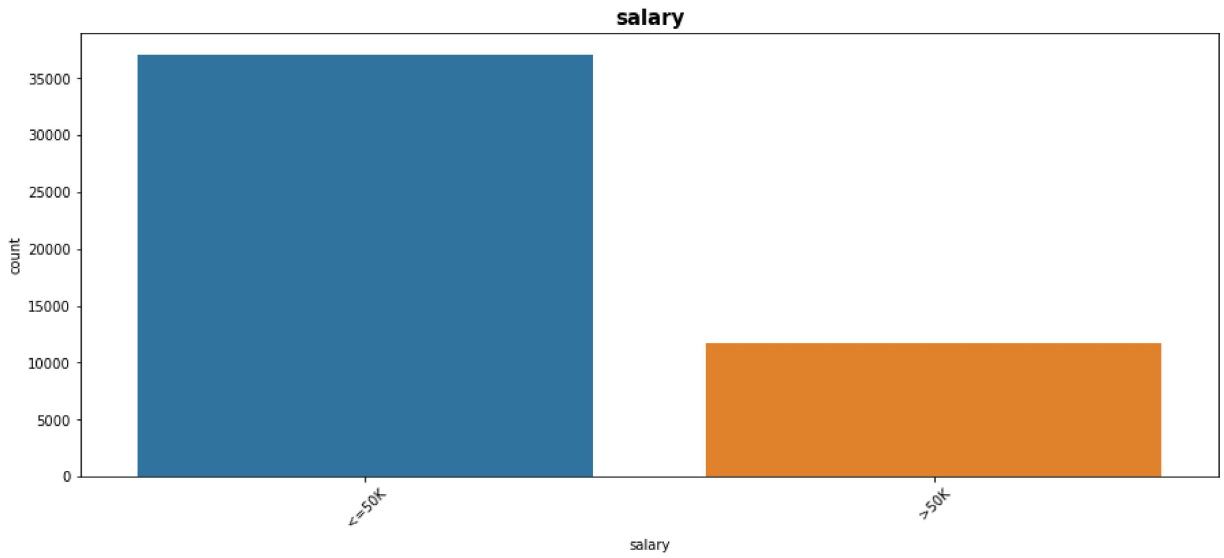

marital_status
count
20000
15000
10000
5000
0
Never-married
Divorced
Separated
Widowed


occupation
count
6000
5000
4000
3000
2000
1000
0
Adm-clerical
Exec-managerial
Handlers-cleaners
Prof-specialty
Other-service
Sales
Craft-repair
Transport-moving
Farming-fishing
Machine-op-inspct
Tech-support
?
Protective-serv
Armed-Forces
Priv-house-serv
occupation


relationship
count
20000
17500
15000
12500
10000
7500
5000
2500
0
Not-in-family
Husband
Wife
Own-child
Unmarried
Other-relative
relationship


race
count
40000
35000
30000
25000
20000
15000
10000
5000
0
White
Black
Asian-Pac-Islander
Amer-Indian-Eskimo
Other
race

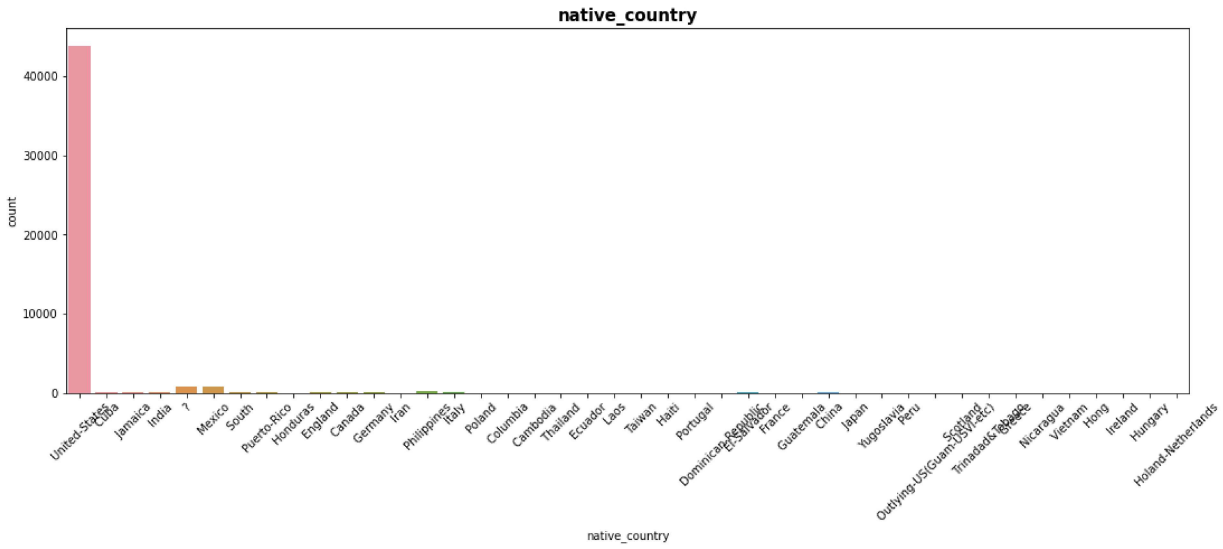

sex
count
30000
25000
20000
15000
10000
5000
0
Male
Female
sex


salary
count
35000
30000
25000
20000
15000
10000
5000
0
<=50K
>50K
salary

Observations

1. The workclass feature has around 33K Private Employees, all other categories has less than 5K Employees whereas Never worked and Without pay has least employees.
2. In education feature, HS-grad has highest (around 16K) person, followed by some college (around 10K) and Bachelors (around 8K) people whereas preschool has least person in armed forces.
3. In Marital Status feature, Married-civ-Spouse has highest (around 22K) person, followed by never married (around 16K) and Divorced (around 6K) people whereas Married-AF-Spouse has least person.
4. In occupation feature, Ex-managerial, prof-speciality, craft repair has almost same employees and is highest. This is followed by admin clerical, sales and other service. The least employees are in
5. Husbands around 20K, not in family 12K, own child 7k and unmarried 5K contrubutes to nearly 90 percent of records in relationship feature.
6. Around 41K people are white by race, followed by 5K black, and Others.
7. There are more male Employees(around 32K) than female employees(16K).
8. Persons having salary less than 50K(35K Persons) are more in number than Persons having salary more than 50K(13K persons)

In [26]:
```
plt.figure(figsize=(18,6))
sns.countplot(data=dataset, x='native_country')
plt.title('native_country', fontsize=15, weight='bold')
plt.xticks(rotation=45)
plt.show();
```



Observations

1. In native country feature, Most employees are from united states (around 44k). The rest can be clubbed in others category.

3.2 Numerical features

In [27]:
```
numerical_features=[feature for feature in dataset.columns if feature not in
print(numerical_features)
```

```
['age', 'fnlwgt', 'education_num', 'capital_gain', 'capital_loss', 'hours_per_w
eek']
```

In [28]:
```
for feature in numerical_features:
    print("Feature '{}' has these {} no. of unique values\n".format(feature,
```

```
Feature 'age' has these 74 no. of unique values

Feature 'fnlwgt' has these 28523 no. of unique values

Feature 'education_num' has these 16 no. of unique values

Feature 'capital_gain' has these 123 no. of unique values

Feature 'capital_loss' has these 99 no. of unique values

Feature 'hours_per_week' has these 96 no. of unique values
```

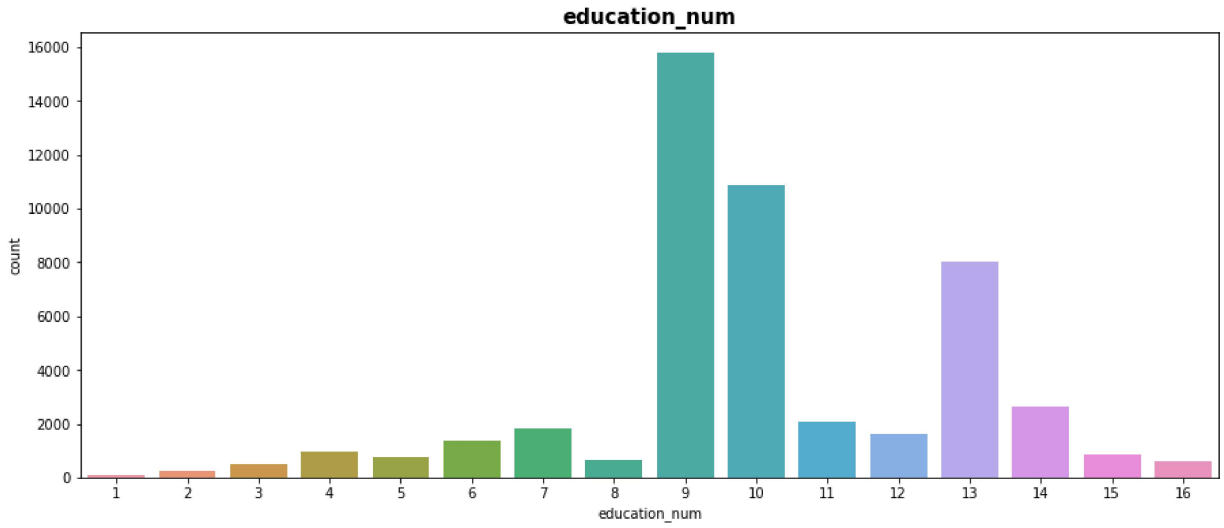
3.2.1 Discrete Numerical features

In [29]:
```
discreate_features=[feature for feature in numerical_features if dataset[fea
print(discreate_features)
```

```
['education_num']
```

3.2.2 Countplot of education_number

In [30]:
```
plt.figure(figsize=(15,6))
sns.countplot(data=dataset, x='education_num')
plt.title('education_num', fontsize=15, weight='bold')
plt.show();
```



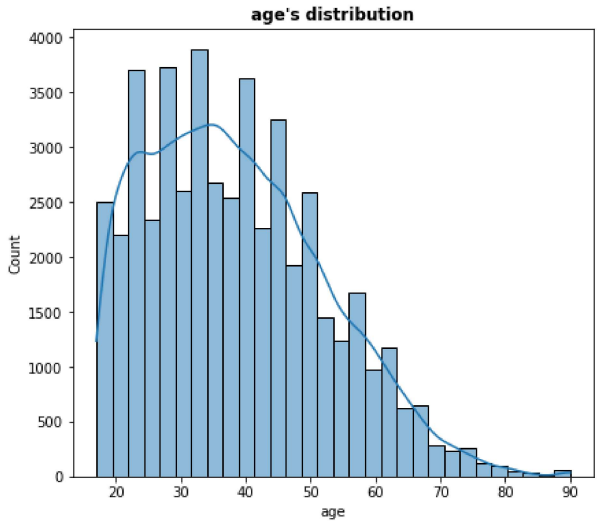
3.3 Continuous Numerical feature

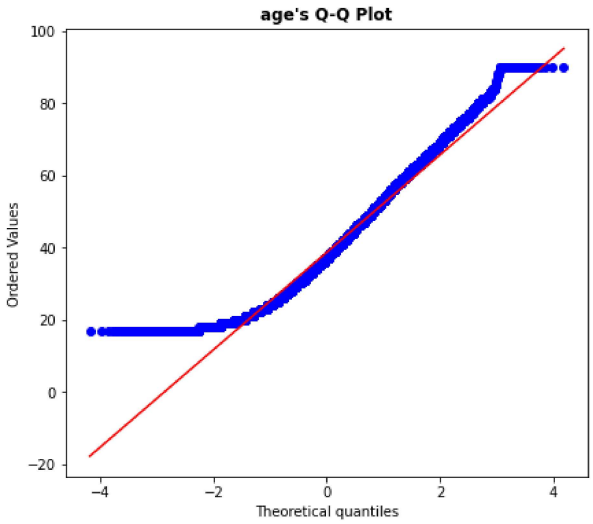
In [31]:
```
continuous_features=[feature for feature in numerical_features if feature no
print(continuous_features)
```

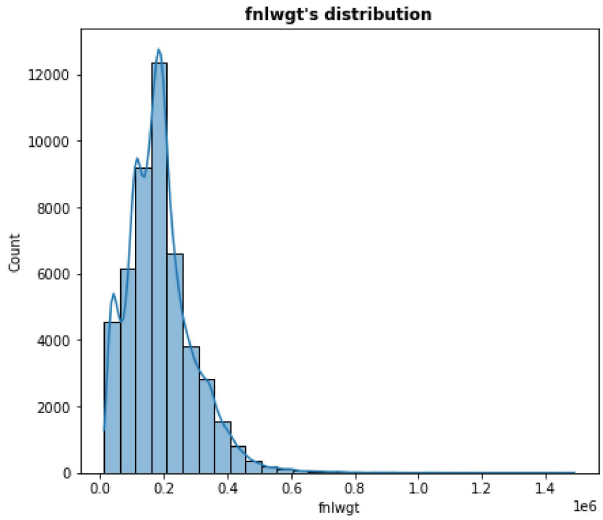
```
['age', 'fnlwgt', 'capital_gain', 'capital_loss', 'hours_per_week']
```

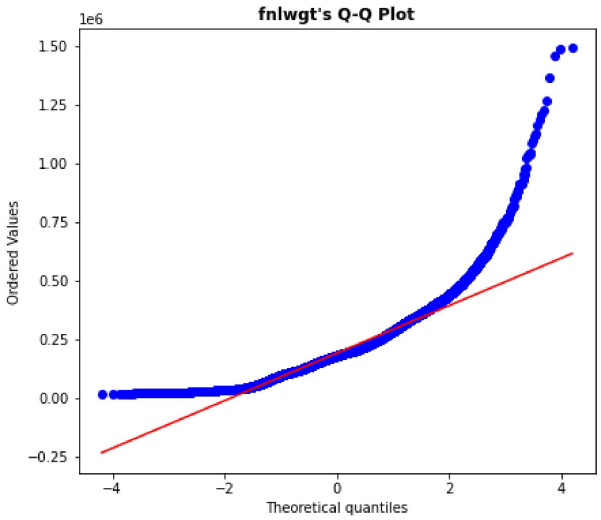
3.3.1 Distribution of Continuous Numerical Features

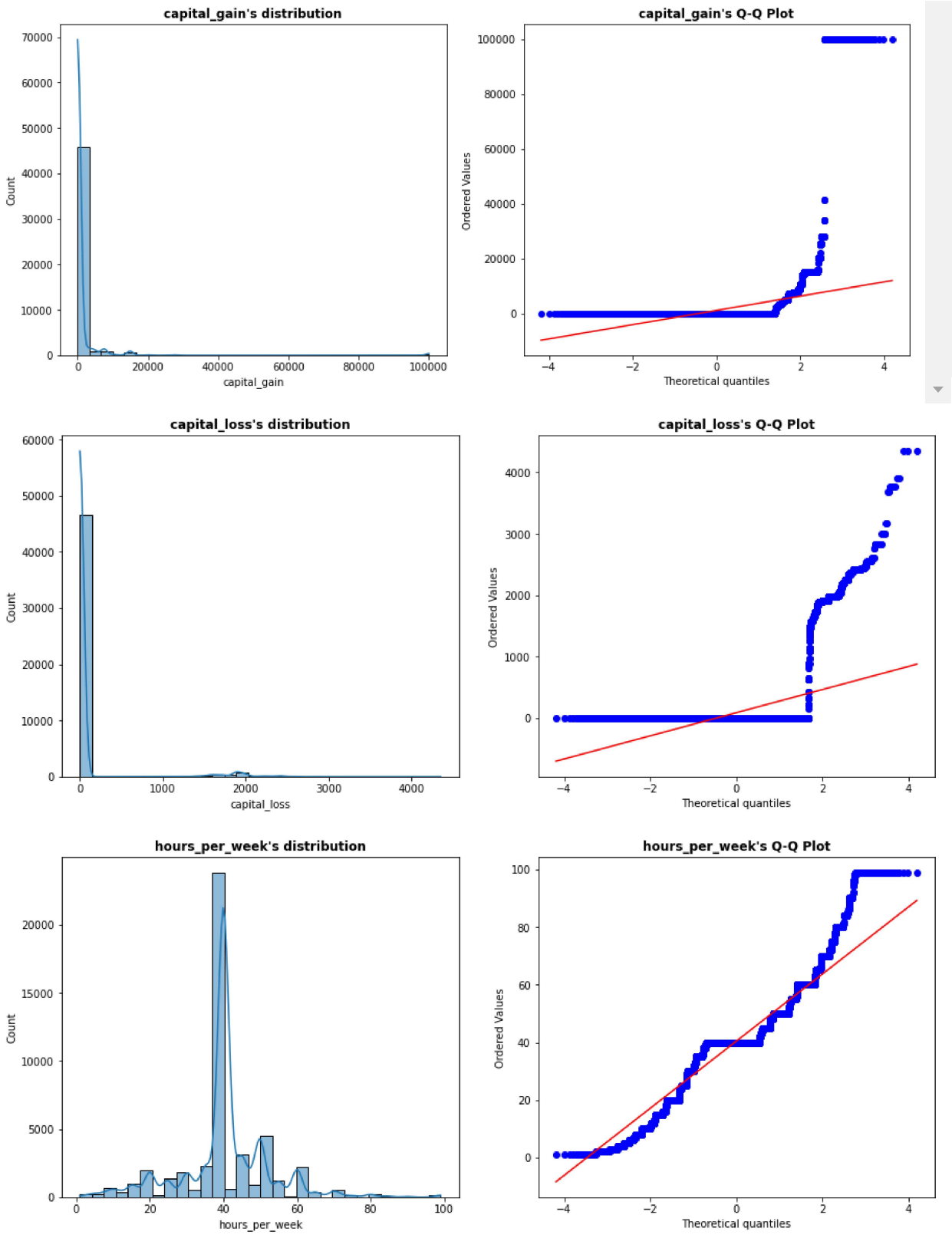
In [32]:
```
### Checking distribution of Continuous numerical features

for i in continuous_features:
    plt.figure(figsize=(15,6))
    plt.subplot(121)
    sns.histplot(data=dataset, x=i, kde=True, bins=30)
    plt.title("{}'s distribution".format(i),fontweight="bold")

    plt.subplot(122)
    stats.probplot(dataset[i], dist='norm', plot=plt)
    plt.title("{}'s Q-Q Plot".format(i),fontweight="bold")
    plt.show();
```





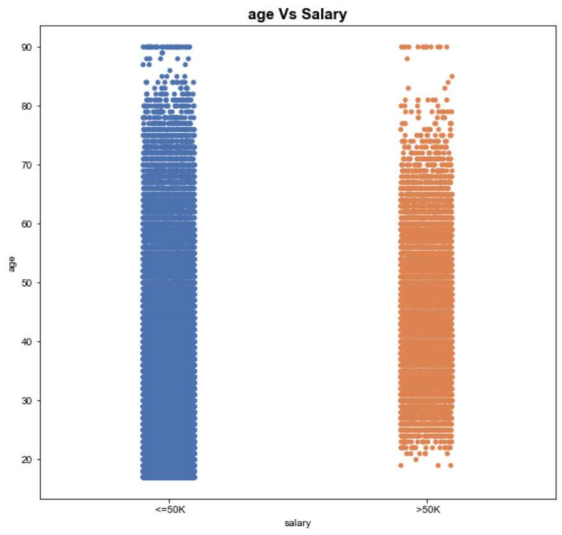


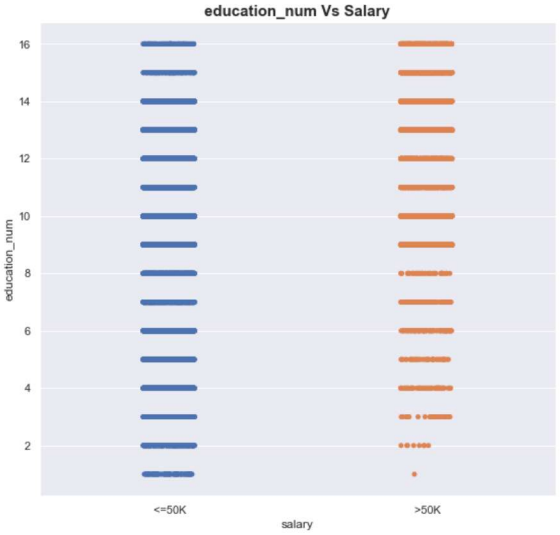


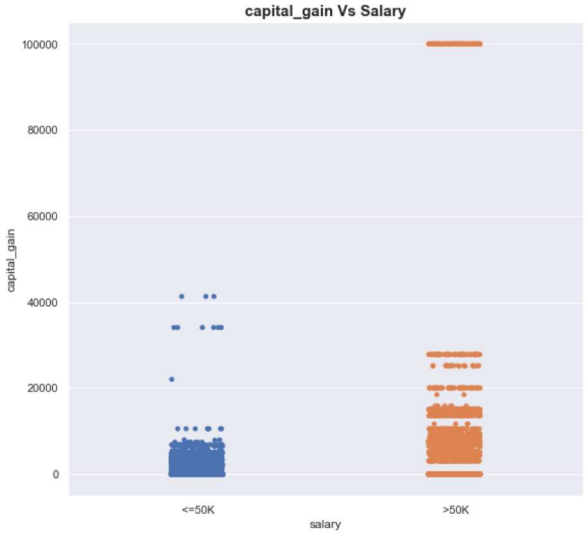


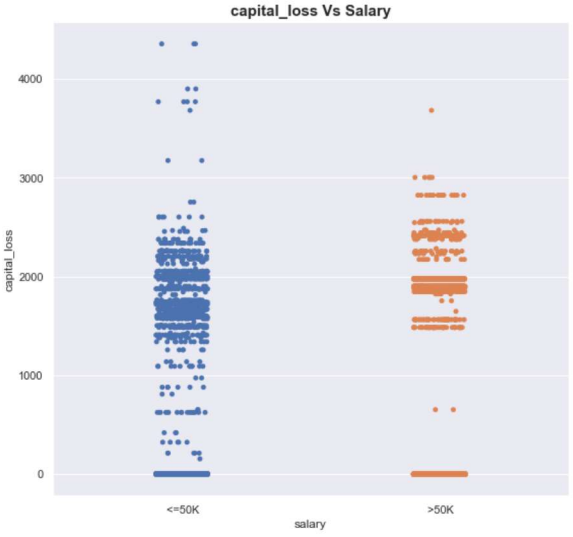
3.3.2 Continuous Numerical Features vs Dependent feature

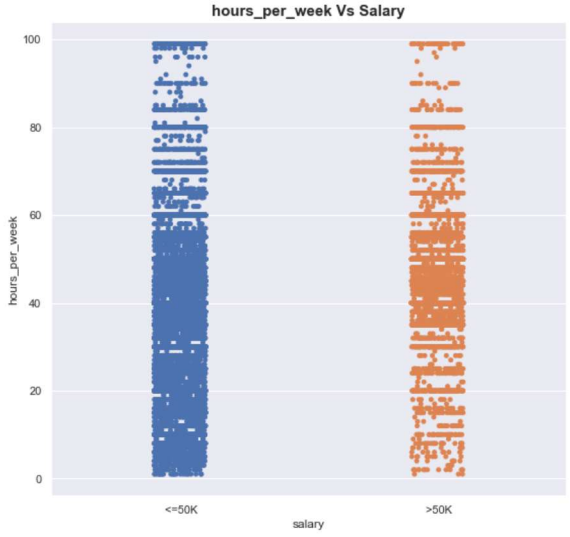
In [33]:
```
plt.figure(figsize=(20,60))
for i in enumerate(numerical_features):
    plt.subplot(6, 2, i[0]+1)
    sns.set(rc={'figure.figsize':(8,10)})
    sns.stripplot(data=dataset, y=i[1], x='salary')
    plt.title("{} Vs Salary".format(i[1]), fontsize=15, fontweight='bold')
```





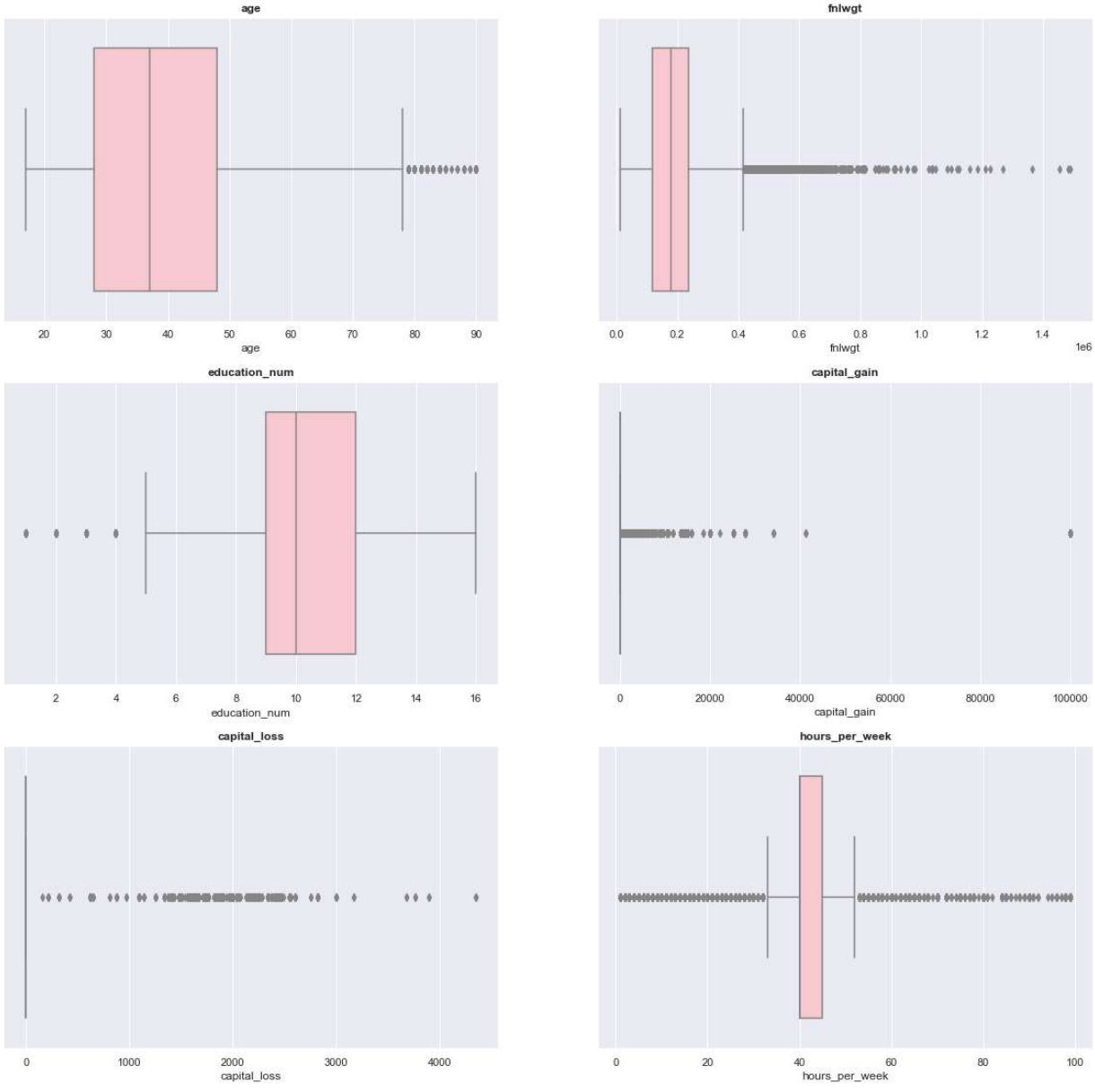






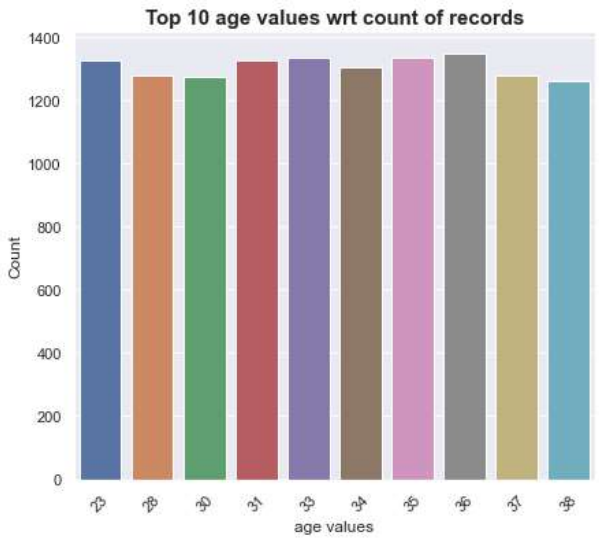
3.3.3 Checking Outliers

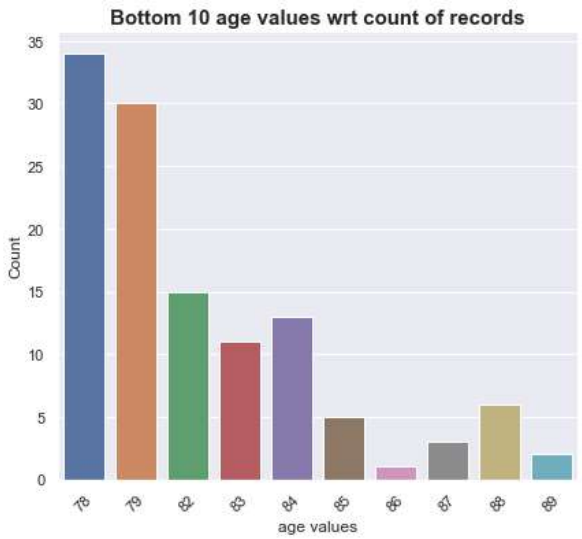
In [34]:
```
### Checking outliers in numerical features

plt.figure(figsize=(20,40))
for i in enumerate(numerical_features):
    plt.subplot(6, 2, i[0]+1)
    sns.set(rc={'figure.figsize':(10,6)})
    sns.boxplot(data=dataset, x=i[1], color='pink')
    plt.title("{}".format(i[1]), fontweight="bold")
```

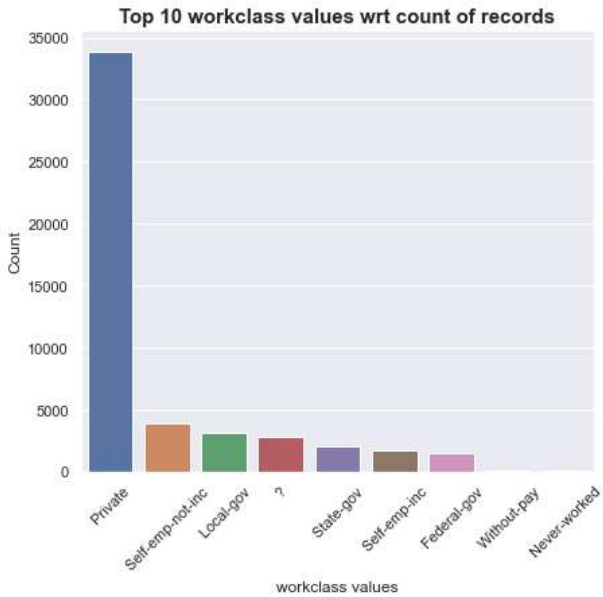


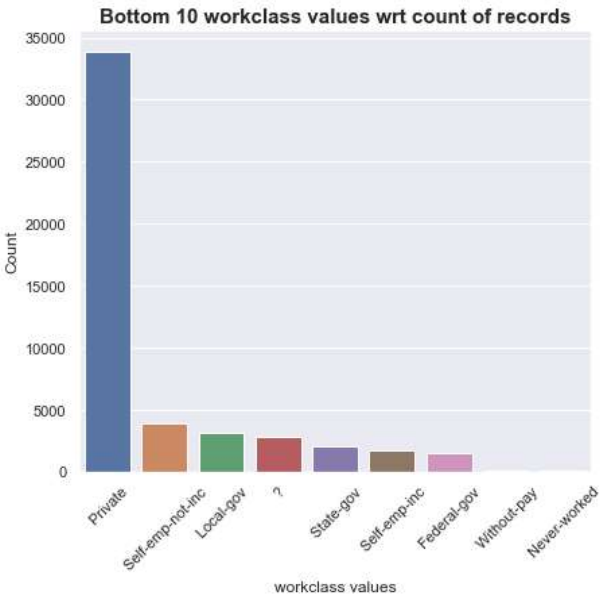
3.4 Top and Bottom 10 feature values wrt count of records

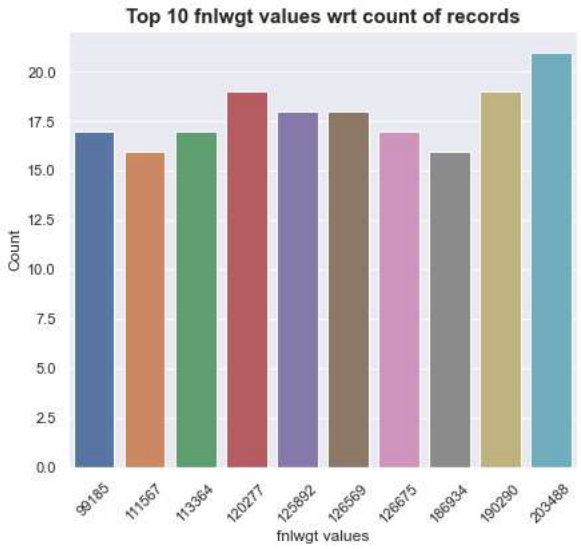
In [35]:
```
for feature in [feature for feature in dataset.columns if feature not in ['s
    plt.figure(figsize=(15,6))
    plt.subplot(121)
    sns.barplot(y=dataset[feature].value_counts()[:10], x=dataset[feature].v
    plt.ylabel('Count')
    plt.xlabel('{} values'.format(feature))
    plt.xticks(rotation=45)
    plt.title("Top 10 {} values wrt count of records".format(feature),fontsi

    plt.subplot(122)
    sns.barplot(y=dataset[feature].value_counts()[-10:], x=dataset[feature].
    plt.ylabel('Count')
    plt.xlabel('{} values'.format(feature))
    plt.xticks(rotation=45)
    plt.title("Bottom 10 {} values wrt count of records".format(feature),fon
    plt.show();
```

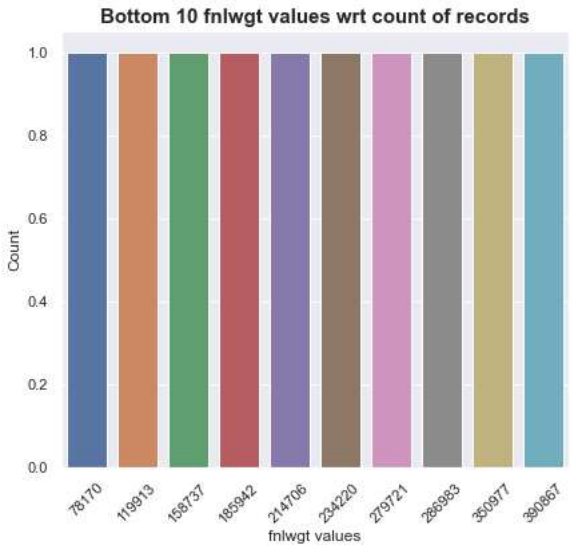


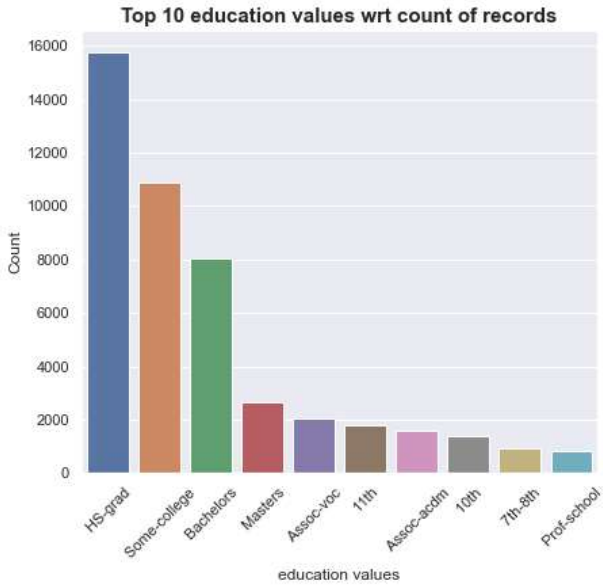


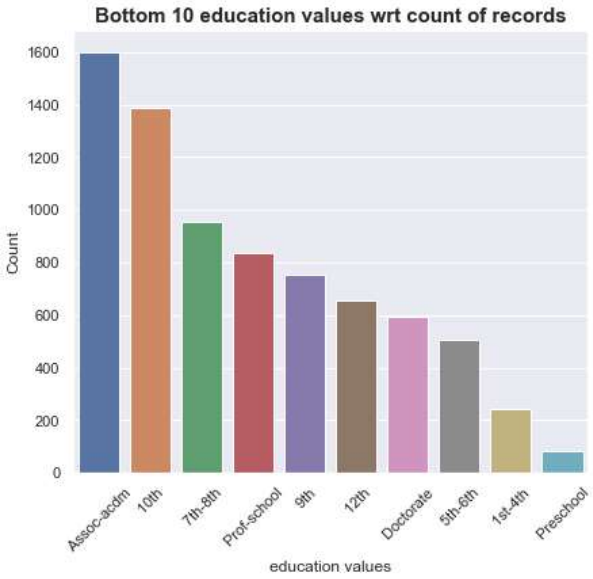


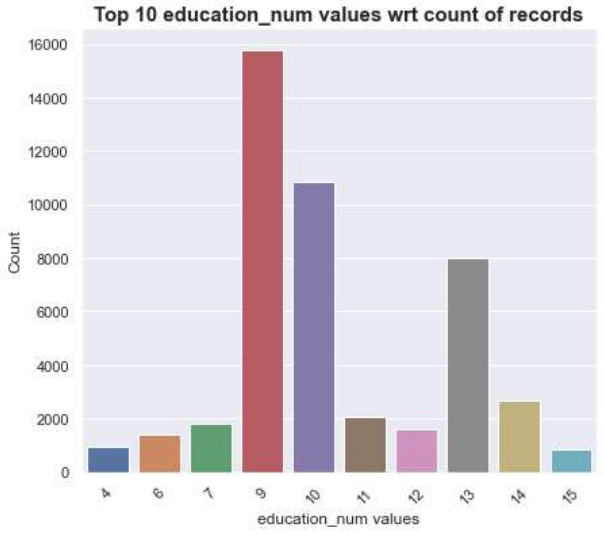


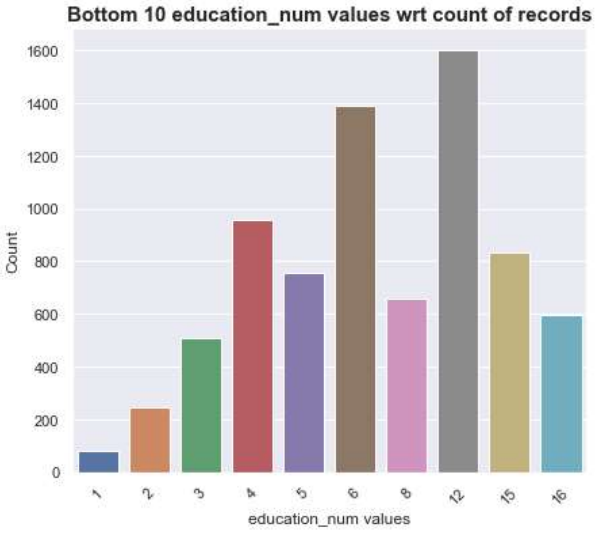

Top 10 fnlwgt values wrt count of records
Count
fnlwgt values
99185
111567
113364
120277
125892
126569
126675
186934
190290
203488

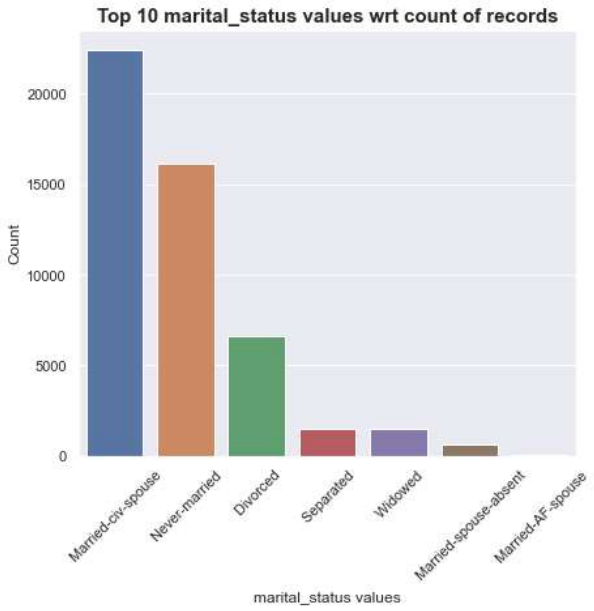

Bottom 10 fnlwgt values wrt count of records
Count
fnlwgt values
78170
119913
158737
185942
214706
234220
279721
286983
350977
390867

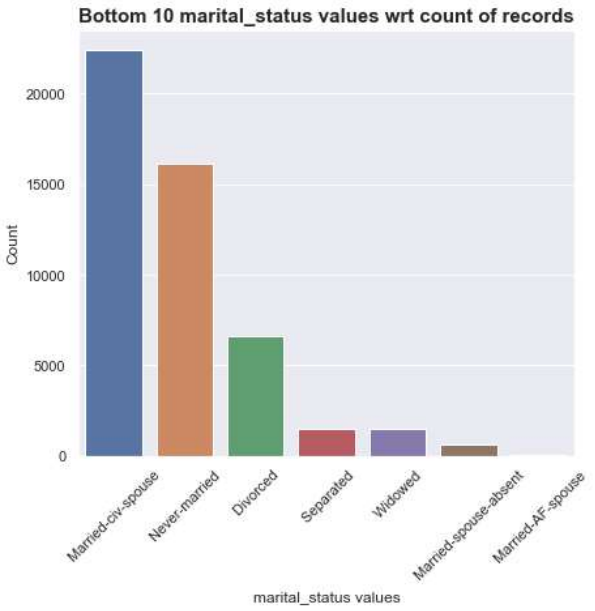

Top 10 education values wrt count of records
Count
education values
HS-grad
Some-college
Bachelors
Masters
Assoc-voc
11th
Assoc-acdm
10th
7th-8th
Prof-school

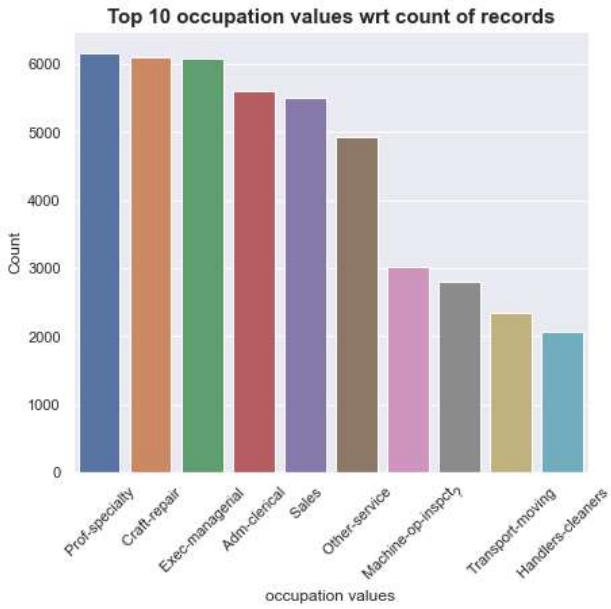

Bottom 10 education values wrt count of records
Count
education values
Assoc-acdm
10th
7th-8th
Prof-school
9th
12th
Doctorate
5th-6th
1st-4th
Preschool

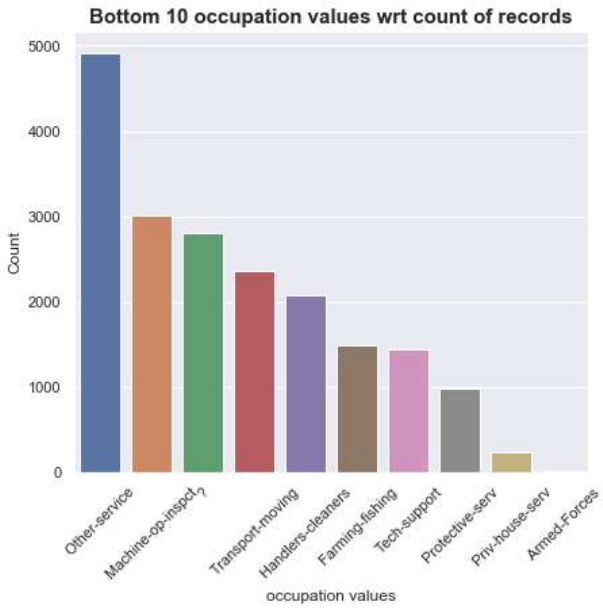

Top 10 education_num values wrt count of records
Count
education_num values
4
6
7
9
10
11
12
13
14
15

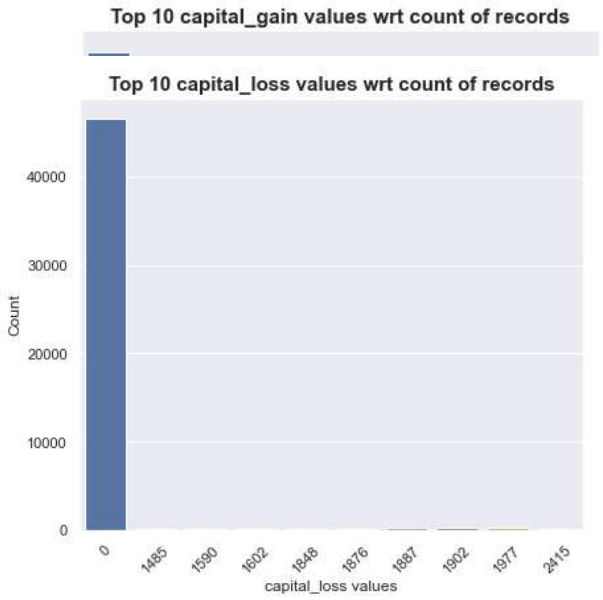

Bottom 10 education_num values wrt count of records
Count
education_num values
1
2
3
4
5
6
8
12
15
16

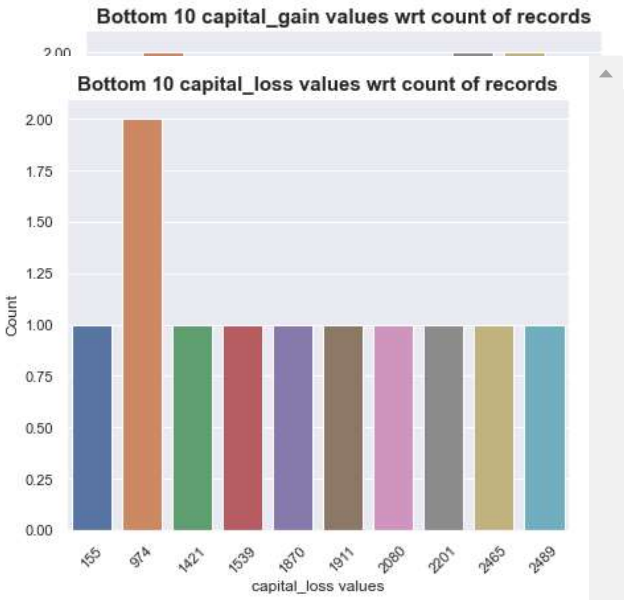

Top 10 marital_status values wrt count of records
Count
marital_status values
Married-civ-spouse
Never-married
Divorced
Separated
Widowed
Married-spouse-absent
Married-AF-spouse

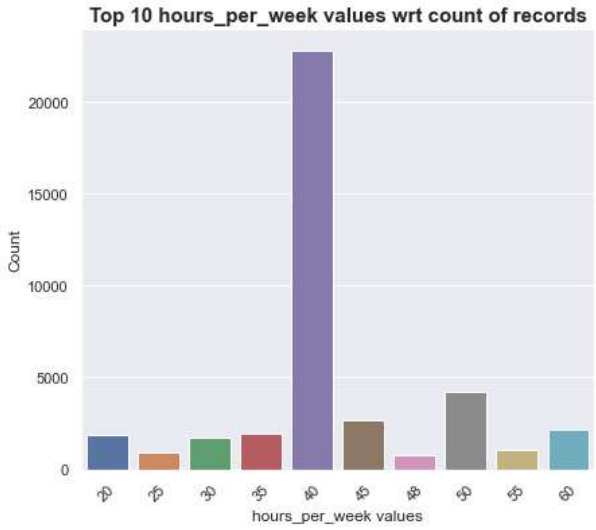

Bottom 10 marital_status values wrt count of records
Count
marital_status values
Married-civ-spouse
Never-married
Divorced
Separated
Widowed
Married-spouse-absent
Married-AF-spouse

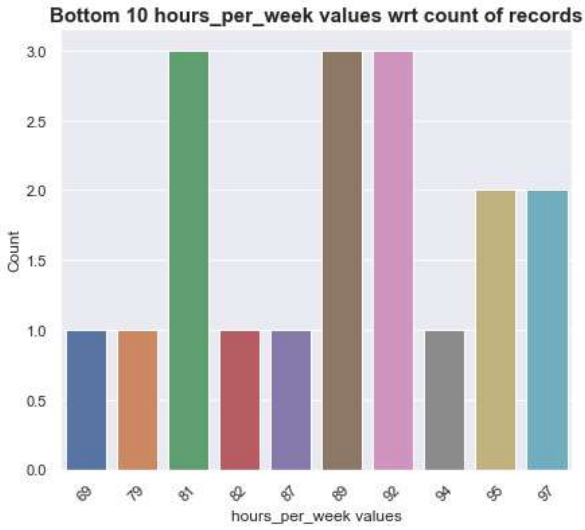

Top 10 occupation values wrt count of records
Count
occupation values
Prof-specialty
Craft-repair
Exec-managerial
Adm-clerical
Sales
Other-service
Machine-op-inspct
Transport-moving
Handlers-cleaners

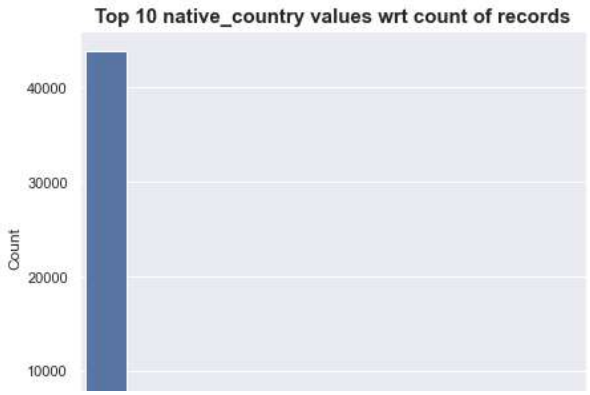

Bottom 10 occupation values wrt count of records
Count
occupation values
Other-service
Machine-op-inspct
Transport-moving
Handlers-cleaners
Farming-fishing
Tech-support
Protective-serv
Priv-house-serv
Armed-Forces

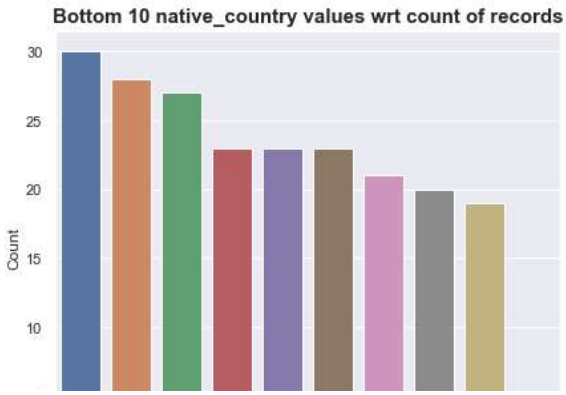

Top 10 capital_gain values wrt count of records
Count
capital_loss values
0
1485
1590
1602
1848
1876
1887
1902
1977
2415


Bottom 10 capital_gain values wrt count of records
Count
capital_loss values
155
974
1421
1539
1870
1911
2080
2201
2465
2489


Top 10 hours_per_week values wrt count of records
Count
hours_per_week values
20
25
30
35
40
45
48
50
55
60


Bottom 10 hours_per_week values wrt count of records
Count
hours_per_week values
69
79
81
82
87
89
92
94
95
97

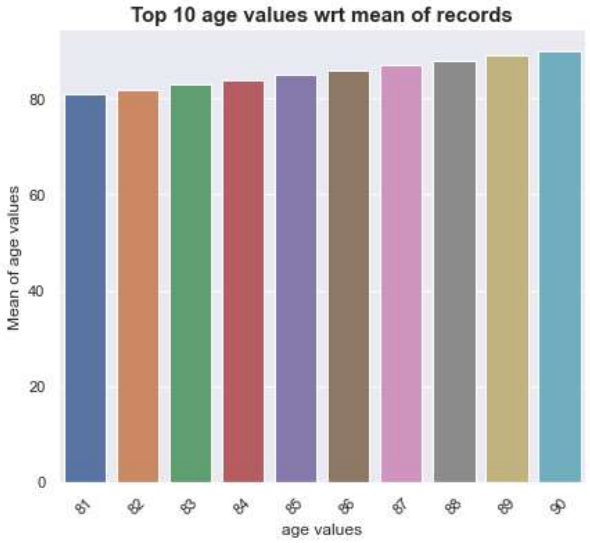


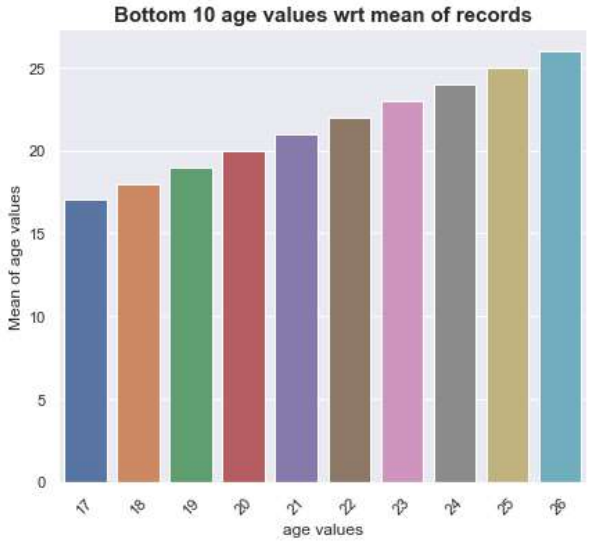


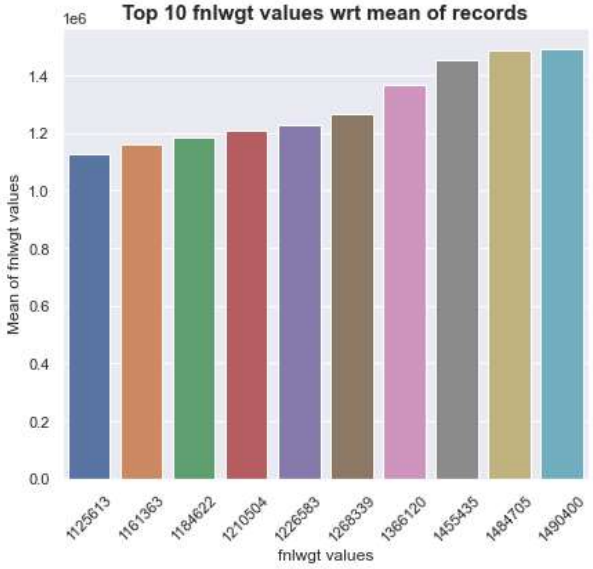
3.5 Top and Bottom 10 Numerical feature values wrt Mean of records

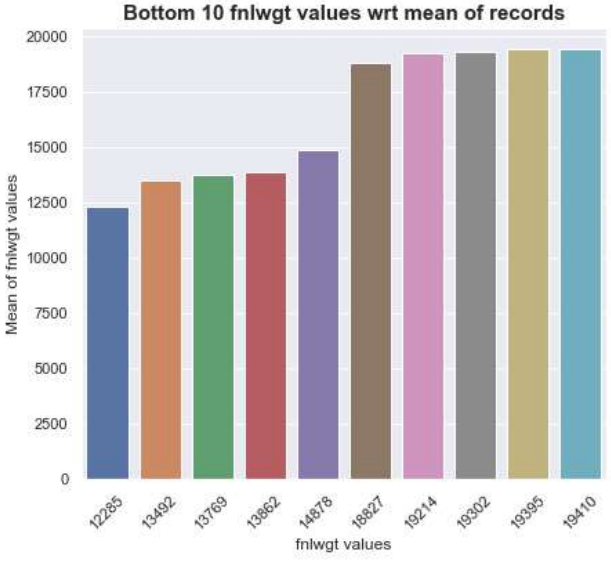
In [36]:

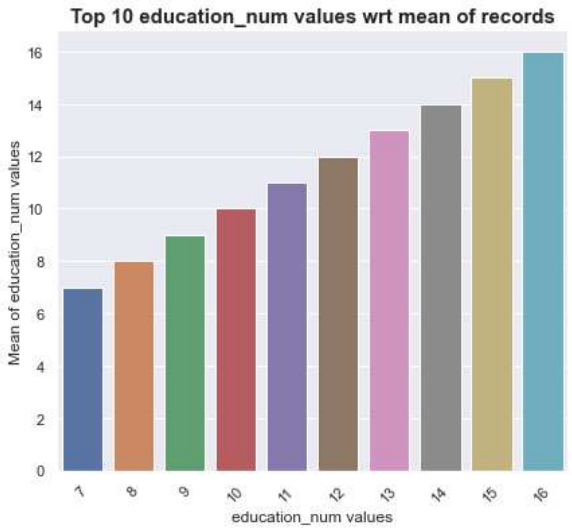
```
for feature in numerical_features:
    plt.figure(figsize=(15,6))
    plt.subplot(121)
    sns.barplot(y=dataset.groupby(feature)[feature].mean().sort_values(ascen
            x=dataset.groupby(feature)[feature].mean().sort_values(ascending
    plt.ylabel('Mean of {} values'.format(feature))
    plt.xlabel('{} values'.format(feature))
    plt.xticks(rotation=45)
    plt.title("Top 10 {} values wrt mean of records".format(feature),fontsiz

    plt.subplot(122)
    sns.barplot(y=dataset.groupby(feature)[feature].mean().sort_values(ascen
            x=dataset.groupby(feature)[feature].mean().sort_values(ascending
    plt.ylabel('Mean of {} values'.format(feature))
    plt.xlabel('{} values'.format(feature))
    plt.xticks(rotation=45)
    plt.title("Bottom 10 {} values wrt mean of records".format(feature),font
    plt.show();
```

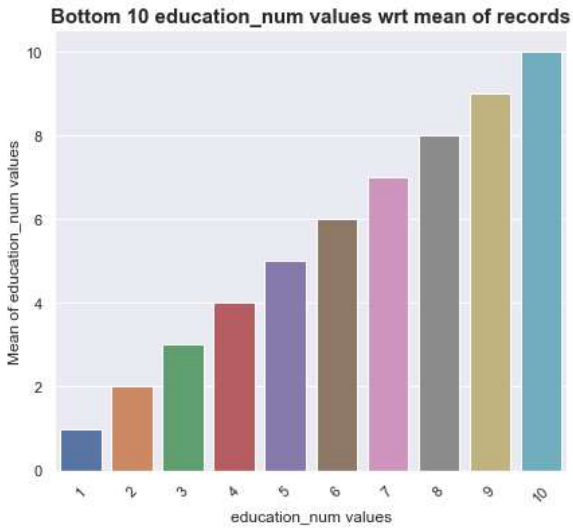


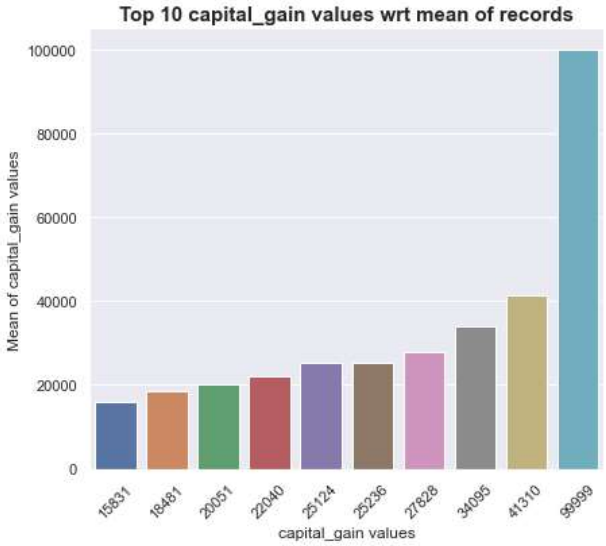


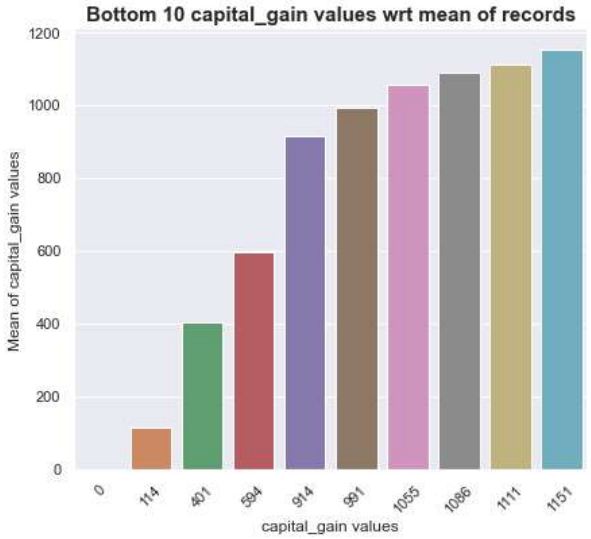


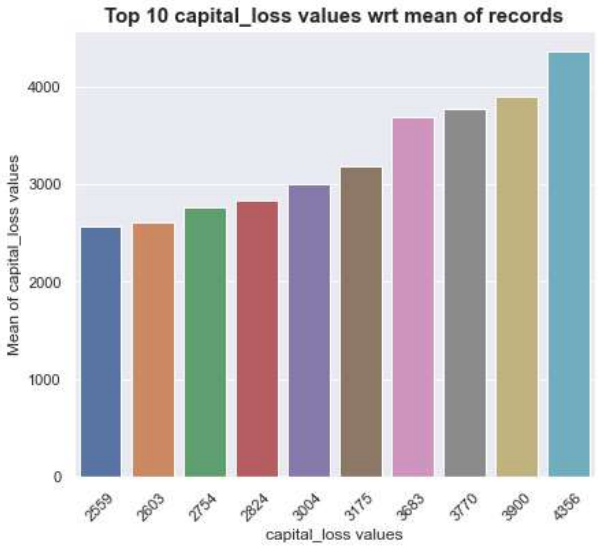


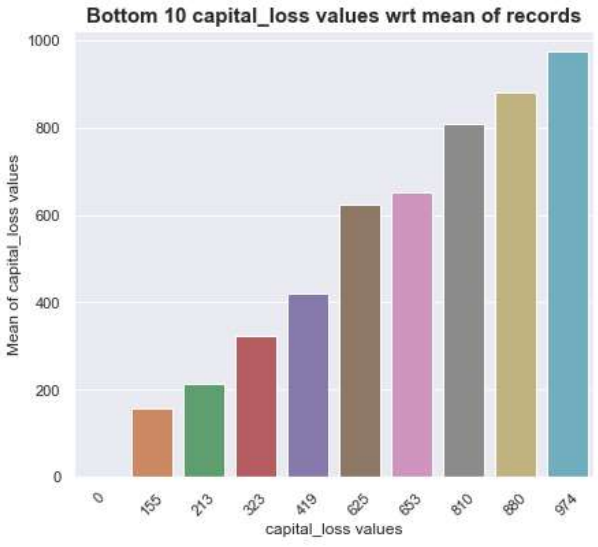

Top 10 education_num values wrt mean of records
Mean of education_num values
education_num values
7
8
9
10
11
12
13
14
15
16

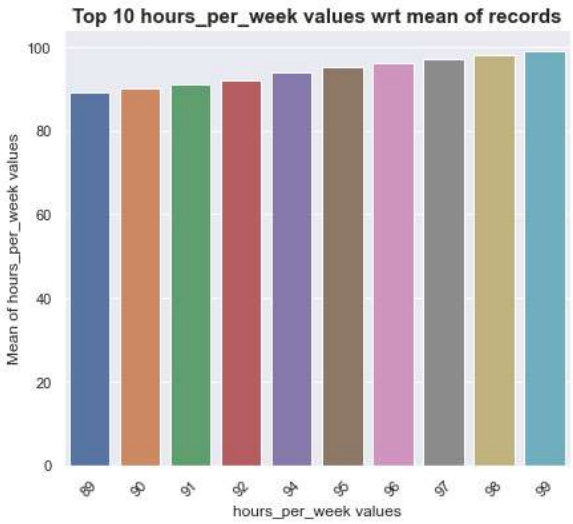

Bottom 10 education_num values wrt mean of records
Mean of education_num values
education_num values
1
2
3
4
5
6
7
8
9
10

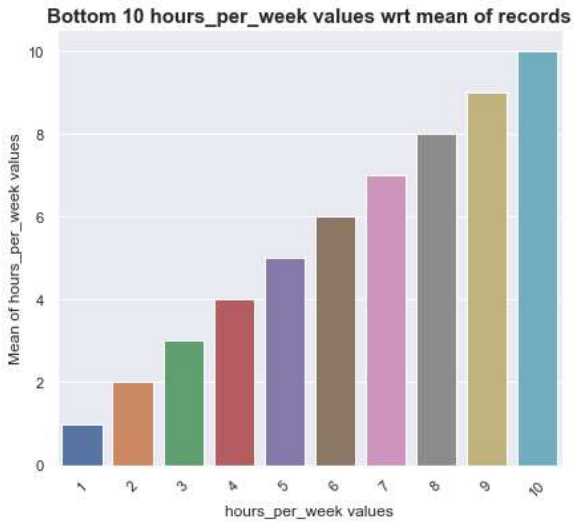

Top 10 capital_gain values wrt mean of records
Mean of capital_gain values
capital_gain values
15831
18481
20051
22040
25124
25236
27828
34095
41310
99999


Bottom 10 capital_gain values wrt mean of records
Mean of capital_gain values
capital_gain values
0
114
401
594
914
991
1055
1086
1111
1151


Top 10 capital_loss values wrt mean of records
Mean of capital_loss values
capital_loss values
2559
2603
2754
2824
3004
3175
3683
3770
3900
4356


Bottom 10 capital_loss values wrt mean of records
Mean of capital_loss values
capital_loss values
0
155
213
323
419
625
653
810
880
974

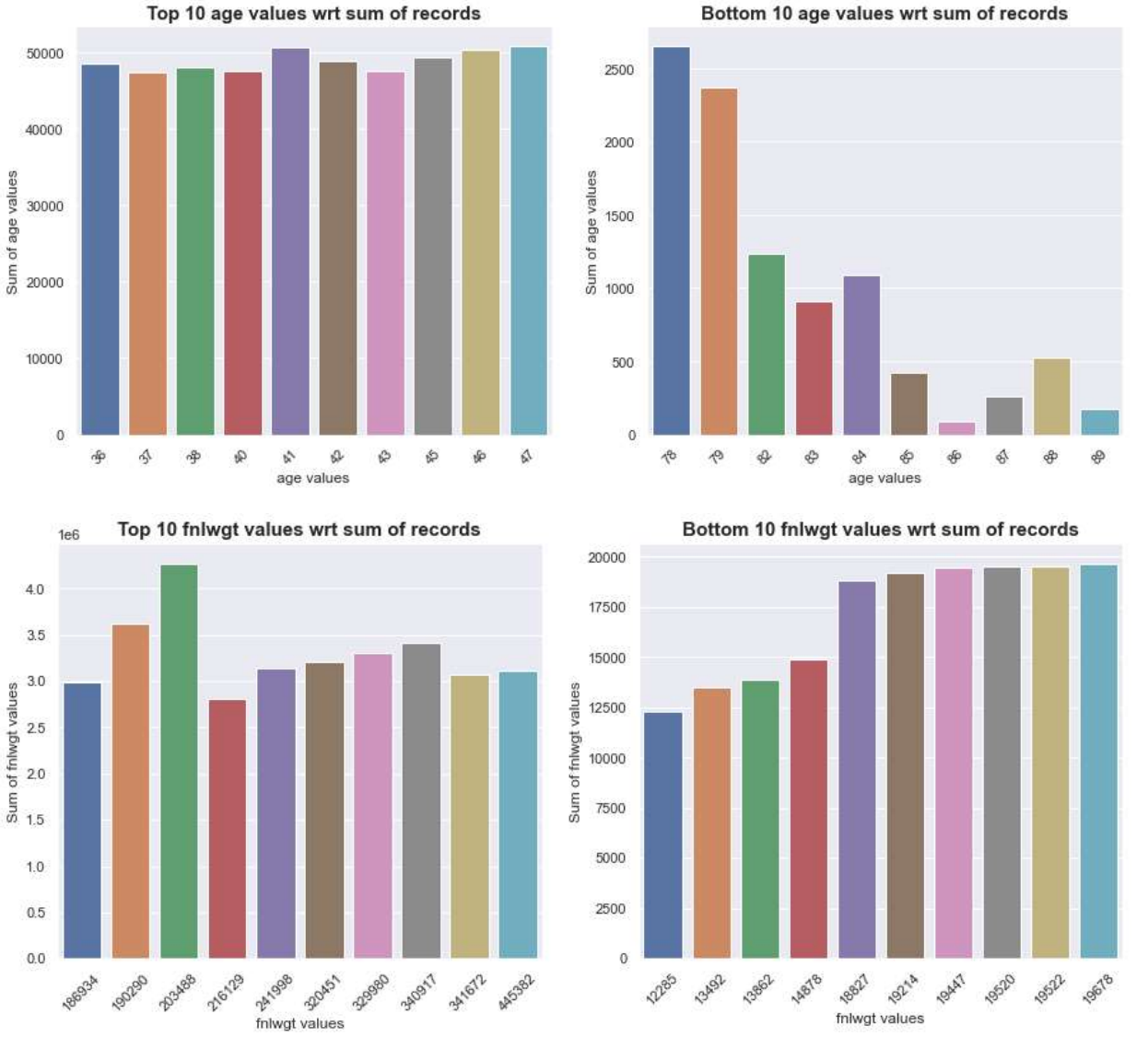


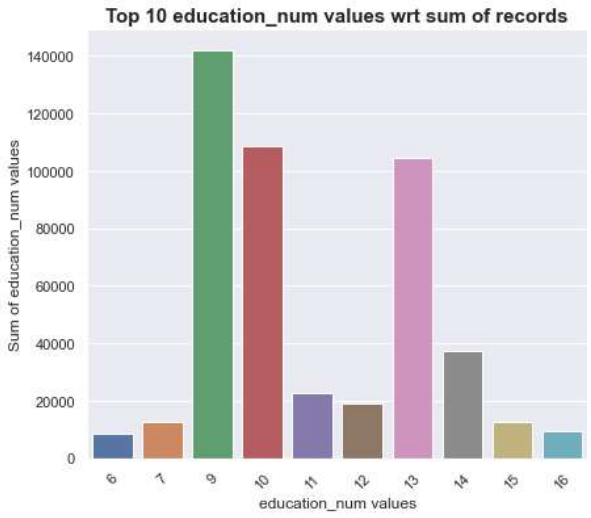


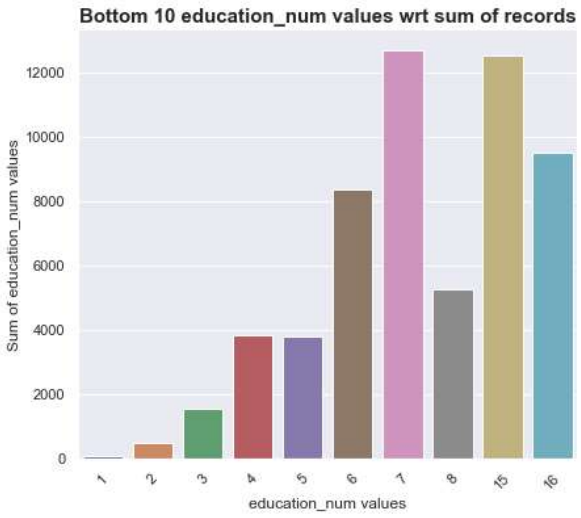
3.6 Top and Bottom 10 Numerical feature values wrt Sum of records

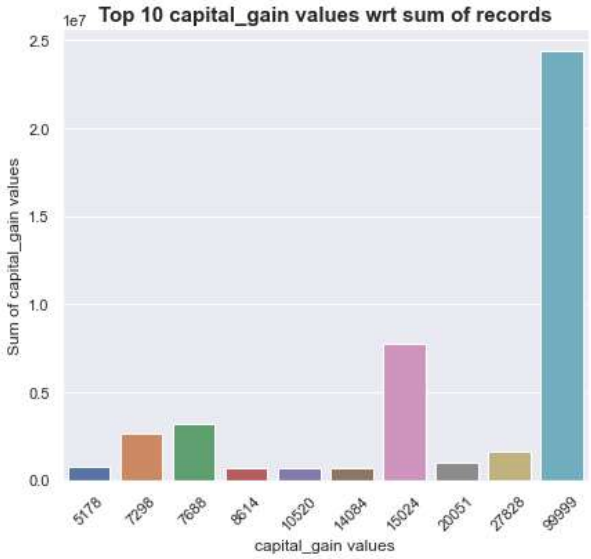
In [37]:

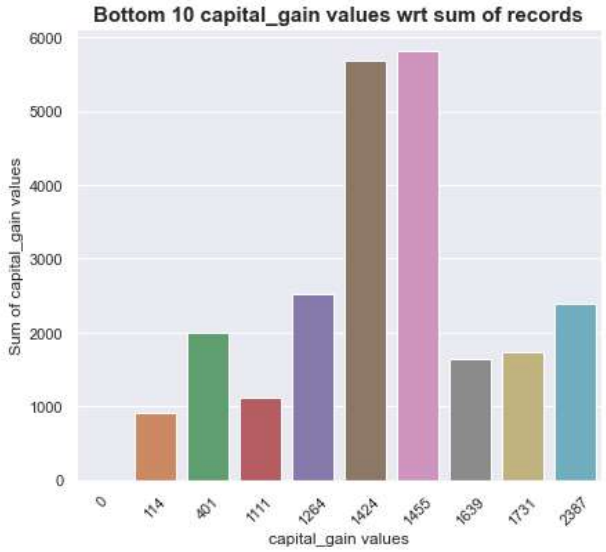
```
for feature in numerical_features:
    plt.figure(figsize=(15,6))
    plt.subplot(121)
    sns.barplot(y=dataset.groupby(feature)[feature].sum().sort_values(ascend
            x=dataset.groupby(feature)[feature].sum().sort_values(ascending=
    plt.ylabel('Sum of {} values'.format(feature))
    plt.xlabel('{} values'.format(feature))
    plt.xticks(rotation=45)
    plt.title("Top 10 {} values wrt sum of records".format(feature),fontsize

    plt.subplot(122)
    sns.barplot(y=dataset.groupby(feature)[feature].sum().sort_values(ascend
            x=dataset.groupby(feature)[feature].sum().sort_values(ascending=
    plt.ylabel('Sum of {} values'.format(feature))
    plt.xlabel('{} values'.format(feature))
    plt.xticks(rotation=45)
    plt.title("Bottom 10 {} values wrt sum of records".format(feature),fonts
    plt.show();
```

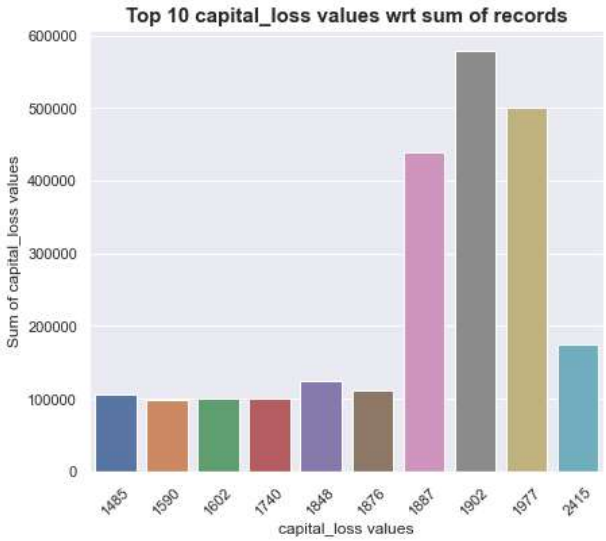


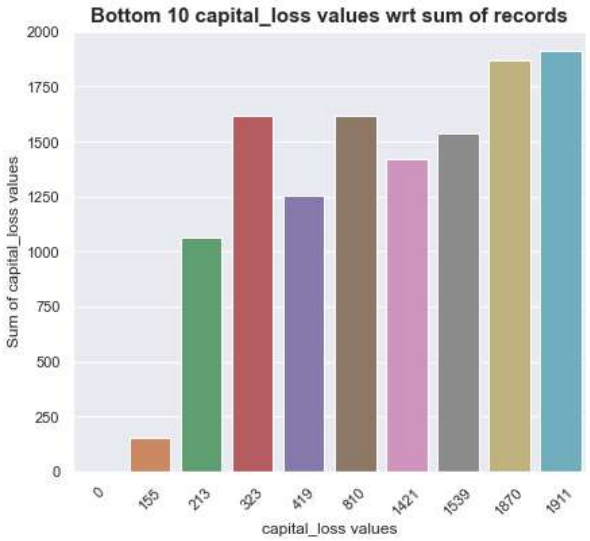

Top 10 education_num values wrt sum of records
Sum of education_num values
education_num values
6
7
9
10
11
12
13
14
15
16

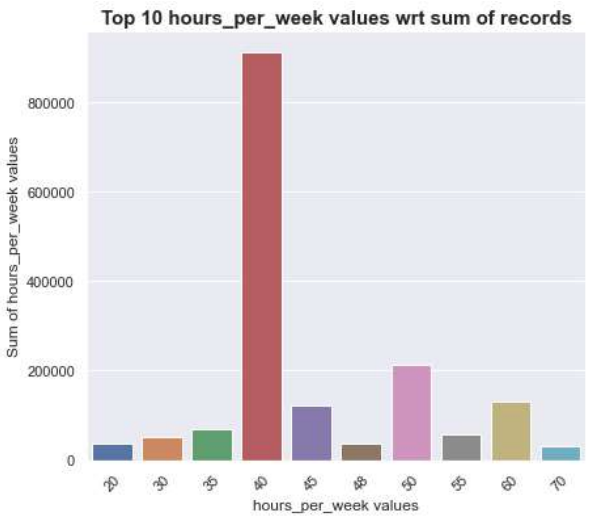

Bottom 10 education_num values wrt sum of records
Sum of education_num values
education_num values
1
2
3
4
5
6
7
8
15
16

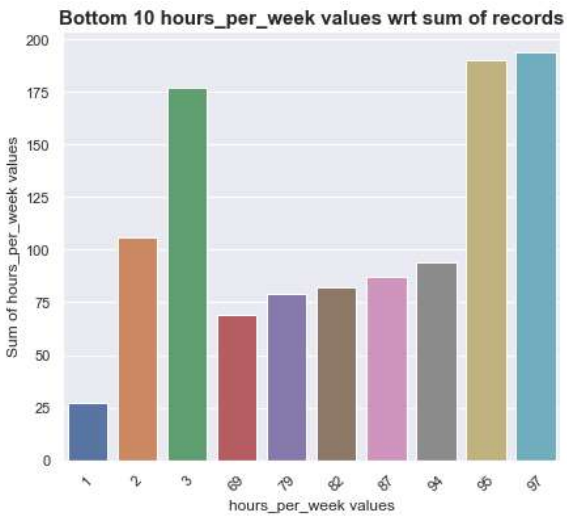

Top 10 capital_gain values wrt sum of records
1e7
Sum of capital_gain values
capital_gain values
5178
7298
7688
8614
10520
14084
15024
20051
27828
99999


Bottom 10 capital_gain values wrt sum of records
Sum of capital_gain values
capital_gain values
0
114
401
1111
1264
1424
1455
1639
1731
2367


Top 10 capital_loss values wrt sum of records
Sum of capital_loss values
capital_loss values
1485
1590
1602
1740
1848
1876
1887
1902
1977
2415


Bottom 10 capital_loss values wrt sum of records
Sum of capital_loss values
capital_loss values
0
155
213
323
419
810
1421
1539
1870
1911





4.0 Data Cleaning Continued

In [38]:
```
### Cleaning values in categorical features
for feature in categorical_features:
    print("Feature {}\n{}".format(feature, dataset[feature].unique()))
```

```
Feature workclass
[' State-gov' ' Self-emp-not-inc' ' Private' ' Federal-gov' ' Local-gov'
 ' ?' ' Self-emp-inc' ' Without-pay' ' Never-worked']
Feature education
[' Bachelors' ' HS-grad' ' 11th' ' Masters' ' 9th' ' Some-college'
 ' Assoc-acdm' ' Assoc-voc' ' 7th-8th' ' Doctorate' ' Prof-school'
 ' 5th-6th' ' 10th' ' 1st-4th' ' Preschool' ' 12th']
Feature marital_status
[' Never-married' ' Married-civ-spouse' ' Divorced'
 ' Married-spouse-absent' ' Separated' ' Married-AF-spouse' ' Widowed']
Feature occupation
[' Adm-clerical' ' Exec-managerial' ' Handlers-cleaners' ' Prof-specialty'
 ' Other-service' ' Sales' ' Craft-repair' ' Transport-moving'
 ' Farming-fishing' ' Machine-op-inspct' ' Tech-support' ' ?'
 ' Protective-serv' ' Armed-Forces' ' Priv-house-serv']
Feature relationship
[' Not-in-family' ' Husband' ' Wife' ' Own-child' ' Unmarried'
 ' Other-relative']
Feature race
[' White' ' Black' ' Asian-Pac-Islander' ' Amer-Indian-Eskimo' ' Other']
Feature sex
[' Male' ' Female']
Feature native_country
[' United-States' ' Cuba' ' Jamaica' ' India' ' ?' ' Mexico' ' South'
 ' Puerto-Rico' ' Honduras' ' England' ' Canada' ' Germany' ' Iran'
 ' Philippines' ' Italy' ' Poland' ' Columbia' ' Cambodia' ' Thailand'
 ' Ecuador' ' Laos' ' Taiwan' ' Haiti' ' Portugal' ' Dominican-Republic'
 ' El-Salvador' ' France' ' Guatemala' ' China' ' Japan' ' Yugoslavia'
 ' Peru' ' Outlying-US(Guam-USVI-etc)' ' Scotland' ' Trinadad&Tobago'
 ' Greece' ' Nicaragua' ' Vietnam' ' Hong' ' Ireland' ' Hungary'
 ' Holand-Netherlands']
Feature salary
[' <=50K' ' >50K']
```

In [39]:
```
### creating function to clean columns
def feature_cleaner_custom(in_dataset, in_feature_list):
    for feature in in_feature_list:
        in_dataset[feature]=in_dataset[feature].str.replace(' ','')
        in_dataset[feature]=in_dataset[feature].str.replace('-','_')
    return in_dataset
```

In [40]:
```
### cleaning values in features
dataset=feature_cleaner_custom(dataset, categorical_features)
```

In [41]:
```
### replacing ? with nan values
dataset.replace('?',np.nan, inplace=True)
```

In [42]:
```
### checking cleaned feature values
for feature in categorical_features:
    print("Feature {}\n{}".format(feature, dataset[feature].unique()))
```

```
Feature workclass
['State_gov' 'Self_emp_not_inc' 'Private' 'Federal_gov' 'Local_gov' nan
 'Self_emp_inc' 'Without_pay' 'Never_worked']
Feature education
['Bachelors' 'HS_grad' '11th' 'Masters' '9th' 'Some_college' 'Assoc_acdm'
 'Assoc_voc' '7th_8th' 'Doctorate' 'Prof_school' '5th_6th' '10th'
 '1st_4th' 'Preschool' '12th']
Feature marital_status
['Never_married' 'Married_civ_spouse' 'Divorced' 'Married_spouse_absent'
 'Separated' 'Married_AF_spouse' 'Widowed']
Feature occupation
['Adm_clerical' 'Exec_managerial' 'Handlers_cleaners' 'Prof_specialty'
 'Other_service' 'Sales' 'Craft_repair' 'Transport_moving'
 'Farming_fishing' 'Machine_op_inspct' 'Tech_support' nan
 'Protective_serv' 'Armed_Forces' 'Priv_house_serv']
Feature relationship
['Not_in_family' 'Husband' 'Wife' 'Own_child' 'Unmarried' 'Other_relative']
Feature race
['White' 'Black' 'Asian_Pac_Islander' 'Amer_Indian_Eskimo' 'Other']
Feature sex
['Male' 'Female']
Feature native_country
['United_States' 'Cuba' 'Jamaica' 'India' nan 'Mexico' 'South'
 'Puerto_Rico' 'Honduras' 'England' 'Canada' 'Germany' 'Iran'
 'Philippines' 'Italy' 'Poland' 'Columbia' 'Cambodia' 'Thailand' 'Ecuador'
 'Laos' 'Taiwan' 'Haiti' 'Portugal' 'Dominican_Republic' 'El_Salvador'
 'France' 'Guatemala' 'China' 'Japan' 'Yugoslavia' 'Peru'
 'Outlying_US(Guam_USVI_etc)' 'Scotland' 'Trinadad&Tobago' 'Greece'
 'Nicaragua' 'Vietnam' 'Hong' 'Ireland' 'Hungary' 'Holand_Netherlands']
Feature salary
['<=50K' '>50K']
```

In [43]:
```
### checking null values in dataset
dataset.isnull().sum()
```

Out[43]:
```
age                0
workclass       2799
fnlwgt             0
education          0
education_num      0
marital_status     0
occupation      2809
relationship       0
race               0
sex                0
capital_gain       0
capital_loss       0
hours_per_week     0
native_country   856
salary             0
dtype: int64
```

In [44]:
```
### custom nan replace function
def custom_cat_nan_replace(in_data, in_feature_list):
    for feature in in_feature_list:
        value=in_data[feature].mode()[0]
        in_data[feature]=in_data[feature].fillna(value)
    return in_data
```

In [45]:
```
### replacing nan values in categorical features
missing_value_cat=['workclass', 'occupation', 'native_country']

dataset=custom_cat_nan_replace(dataset, missing_value_cat )
```

In [46]:
```
### checking null values
dataset.isnull().sum()
```

Out[46]:
```
age               0
workclass         0
fnlwgt            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
salary            0
dtype: int64
```

In [47]:
```
### checking numerical features
for feature in numerical_features:
    print("Feature {}\n{}".format(feature, dataset[feature].unique()))
```

```
Feature age
[39 50 38 53 28 37 49 52 31 42 30 23 32 40 34 25 43 54 35 59 56 19 20 45
 22 48 21 24 57 44 41 29 18 47 46 36 79 27 67 33 76 17 55 61 70 64 71 68
 66 51 58 26 60 90 75 65 77 62 63 80 72 74 69 73 81 78 88 82 83 84 85 86
 87 89]
Feature fnlwgt
[ 77516  83311 215646 ... 173449  89686 350977]
Feature education_num
[13  9  7 14  5 10 12 11  4 16 15  3  6  2  1  8]
Feature capital_gain
[ 2174     0 14084  5178  5013  2407 14344 15024  7688 34095  4064  4386
  7298  1409  3674  1055  3464  2050  2176   594 20051  6849  4101  1111
  8614  3411  2597 25236  4650  9386  2463  3103 10605  2964  3325  2580
  3471  4865 99999  6514  1471  2329  2105  2885 25124 10520  2202  2961
 27828  6767  2228  1506 13550  2635  5556  4787  3781  3137  3818  3942
   914   401  2829  2977  4934  2062  2354  5455 15020  1424  3273 22040
  4416  3908 10566   991  4931  1086  7430  6497   114  7896  2346  3418
  3432  2907  1151  2414  2290 15831 41310  4508  2538  3456  6418  1848
  3887  5721  9562  1455  2036  1831 11678  2936  2993  7443  6360  1797
  1173  4687  6723  2009  6097  2653  1639 18481  7978  2387  5060  1264
  7262  1731  6612]
Feature capital_loss
[   0 2042 1408 1902 1573 1887 1719 1762 1564 2179 1816 1980 1977 1876
 1340 2206 1741 1485 2339 2415 1380 1721 2051 2377 1669 2352 1672  653
 2392 1504 2001 1590 1651 1628 1848 1740 2002 1579 2258 1602  419 2547
 2174 2205 1726 2444 1138 2238  625  213 1539  880 1668 1092 1594 3004
 2231 1844  810 2824 2559 2057 1974  974 2149 1825 1735 1258 2129 2603
 2282  323 4356 2246 1617 1648 2489 3770 1755 3683 2267 2080 2457  155
 3900 2201 1944 2467 2163 2754 2472 1411 1429 3175 1510 1870 1911 2465
 1421]
Feature hours_per_week
[40 13 16 45 50 80 30 35 60 20 52 44 15 25 38 43 55 48 58 32 70  2 22 56
 41 28 36 24 46 42 12 65  1 10 34 75 98 33 54  8  6 64 19 18 72  5  9 47
 37 21 26 14  4 59  7 99 53 39 62 57 78 90 66 11 49 84  3 17 68 27 85 31
 51 77 63 23 87 88 73 89 97 94 29 96 67 82 86 91 81 76 92 61 74 95 79 69]
```

5.0 Handling Rare categories

In [48]:
```
### Getting categories percentage in each features
for feature in categorical_features:
    print(dataset[feature].value_counts()/dataset.shape[0]*100)
```

```
Private             75.139819
Self_emp_not_inc     7.909778
Local_gov            6.424518
State_gov            4.058345
Self_emp_inc         3.470387
Federal_gov          2.933645
Without_pay          0.043021
Never_worked         0.020486
Name: workclass, dtype: float64
HS_grad         32.321308
Some_college    22.266609
Bachelors       16.430049
Masters          5.441173
Assoc_voc        4.220187
11th             3.712126
Assoc_acdm       3.279864
10th             2.845553
7th_8th          1.954397
Prof_school      1.708561
9th              1.548768
12th             1.343904
Doctorate        1.216889
5th_6th          1.040706
1st_4th          0.501915
Preschool        0.167988
Name: education, dtype: float64
Married_civ_spouse       45.832053
Never_married            32.978920
Divorced                 13.582447
Separated                 3.134411
Widowed                   3.109827
Married_spouse_absent     1.286543
Married_AF_spouse         0.075799
Name: marital_status, dtype: float64
Prof_specialty       18.388544
Craft_repair         12.511011
Exec_managerial      12.463893
Adm_clerical         11.488743
Sales                11.275685
Other_service        10.077234
Machine_op_inspct     6.184828
Transport_moving      4.824534
Handlers_cleaners     4.242722
Farming_fishing       3.046320
Tech_support          2.960277
Protective_serv       2.013808
Priv_house_serv       0.491672
Armed_Forces          0.030730
Name: occupation, dtype: float64
Husband           40.376539
Not_in_family     25.745191
Own_child         15.520456
Unmarried         10.497204
```

```
Wife              4.775367
Other_relative    3.085244
Name: relationship, dtype: float64
White                85.501813
Black                 9.593756
Asian_Pac_Islander    3.109827
Amer_Indian_Eskimo    0.962858
Other                 0.831746
Name: race, dtype: float64
Male      66.848995
Female    33.151005
Name: sex, dtype: float64
United_States                91.504312
Mexico                        1.940057
Philippines                   0.604347
Germany                       0.422019
Puerto_Rico                   0.376949
Canada                        0.372851
El_Salvador                   0.317538
India                         0.309344
Cuba                          0.282712
England                       0.260177
China                         0.249933
South                         0.235593
Jamaica                       0.217155
Italy                         0.215107
Dominican_Republic            0.211009
Japan                         0.188474
Poland                        0.178231
Guatemala                     0.176183
Vietnam                       0.176183
Columbia                      0.174134
Haiti                         0.153648
Portugal                      0.137259
Taiwan                        0.133161
Iran                          0.120869
Greece                        0.100383
Nicaragua                     0.100383
Peru                          0.094237
Ecuador                       0.092189
France                        0.077848
Ireland                       0.075799
Hong                          0.061459
Thailand                      0.061459
Cambodia                      0.057362
Trinadad&Tobago               0.055313
Laos                          0.047119
Yugoslavia                    0.047119
Outlying_US(Guam_USVI_etc)    0.047119
Scotland                      0.043021
Honduras                      0.040973
Hungary                       0.038924
Holand_Netherlands            0.002049
Name: native_country, dtype: float64
<=50K    76.061705
>50K     23.938295
Name: salary, dtype: float64
```

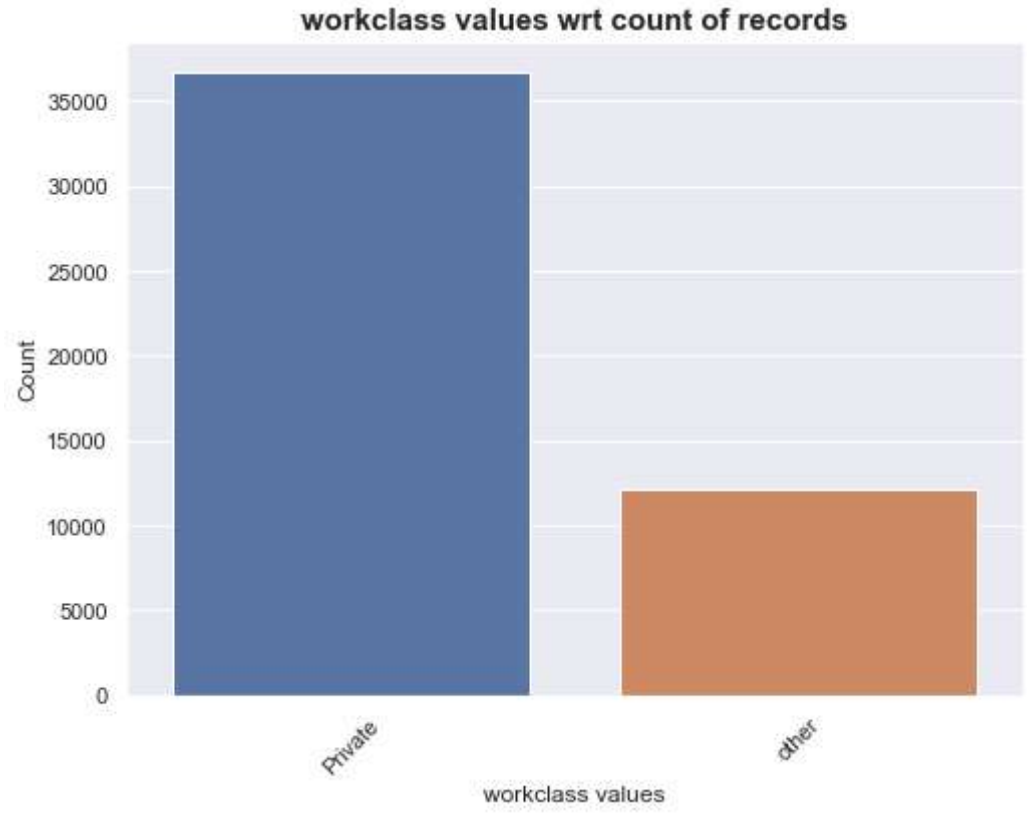
In [49]:
```
### Clubbing rare categories in custom Other category using less than 10 per
for feature in categorical_features:
    frequency=dataset[feature].value_counts(normalize=True)
    mapping=dataset[feature].map(frequency)
    dataset[feature]=dataset[feature].mask(mapping<0.1, 'other')
```

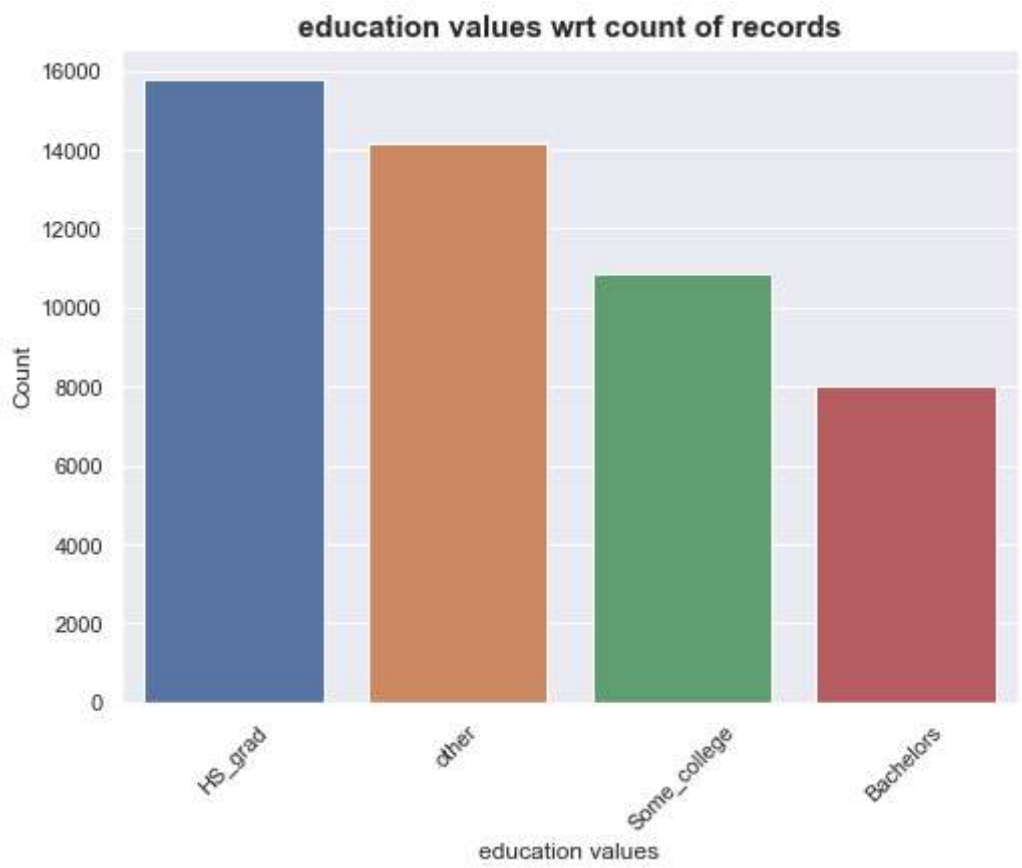
In [50]:
```
### Again checking categories percentage in each features
for feature in categorical_features:
    print(dataset[feature].value_counts()/dataset.shape[0]*100)
```

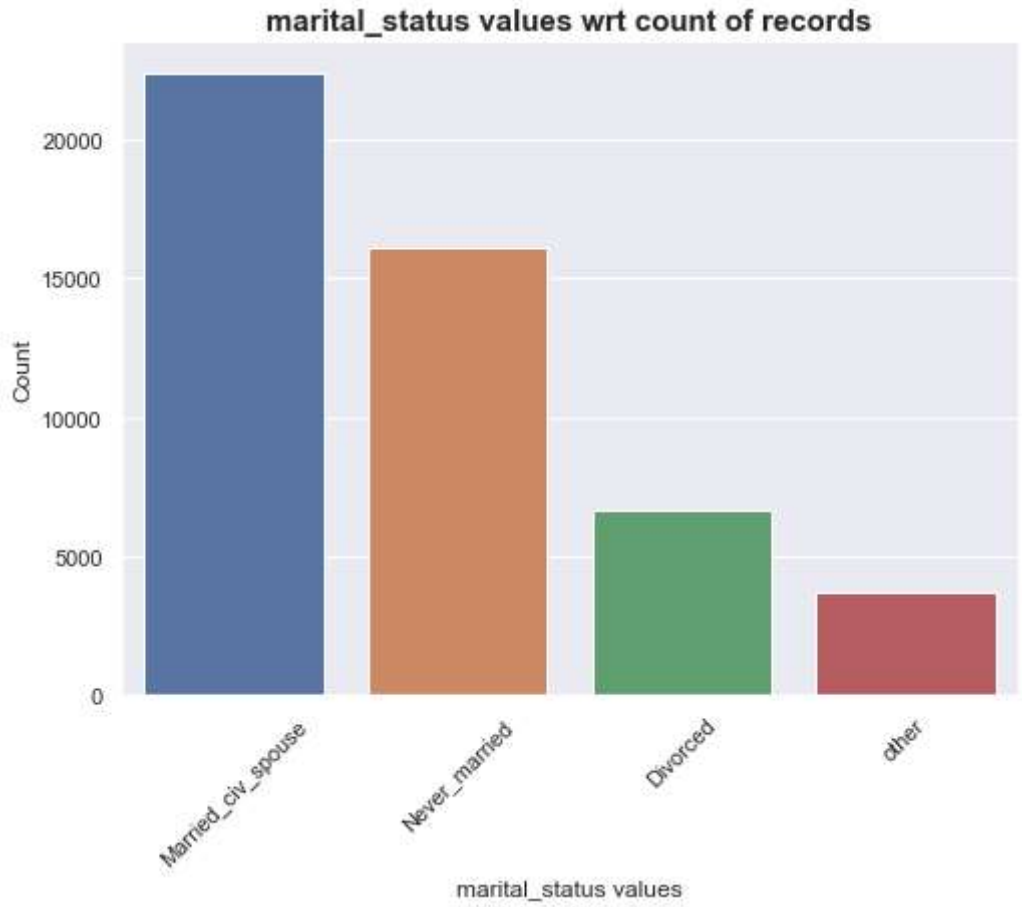
```
Private    75.139819
other      24.860181
Name: workclass, dtype: float64
HS_grad         32.321308
other           28.982033
Some_college    22.266609
Bachelors       16.430049
Name: education, dtype: float64
Married_civ_spouse    45.832053
Never_married         32.978920
Divorced              13.582447
other                  7.606580
Name: marital_status, dtype: float64
other              23.794891
Prof_specialty     18.388544
Craft_repair       12.511011
Exec_managerial    12.463893
Adm_clerical       11.488743
Sales              11.275685
Other_service      10.077234
Name: occupation, dtype: float64
Husband          40.376539
Not_in_family    25.745191
Own_child        15.520456
Unmarried        10.497204
other             7.860611
Name: relationship, dtype: float64
White    85.501813
other    14.498187
Name: race, dtype: float64
Male      66.848995
Female    33.151005
Name: sex, dtype: float64
United_States    91.504312
other             8.495688
Name: native_country, dtype: float64
<=50K    76.061705
>50K     23.938295
Name: salary, dtype: float64
```

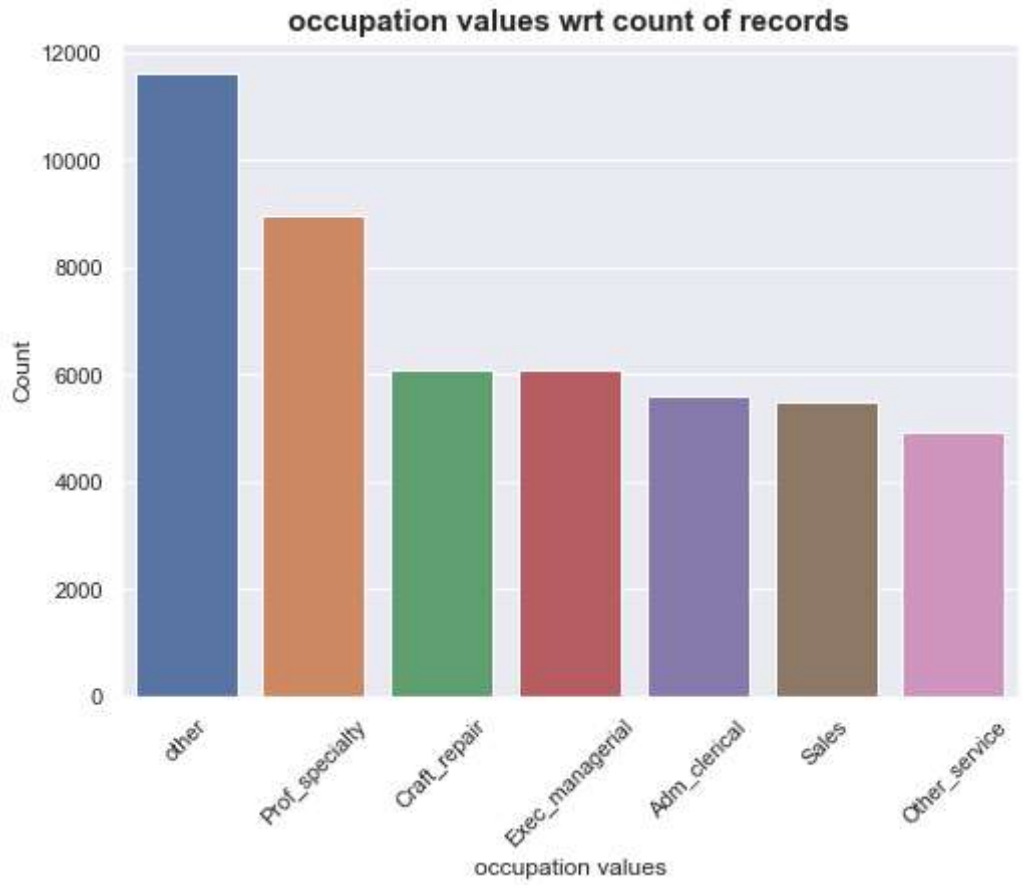
5.1 Visualizing Rare categories

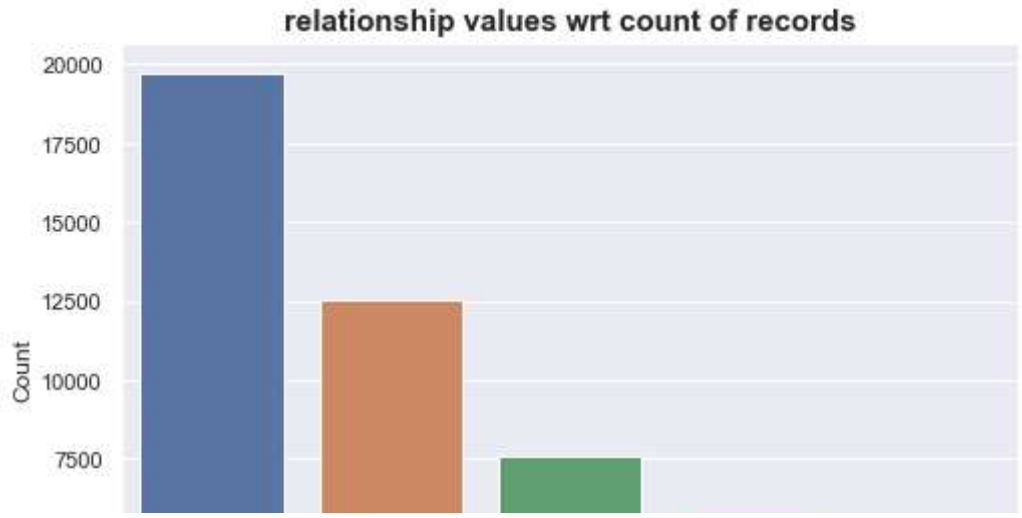
In [51]:
```
for feature in categorical_features:
    plt.figure(figsize=(8,6))
    sns.barplot(y=dataset[feature].value_counts(), x=dataset[feature].value_
    plt.ylabel('Count')
    plt.xlabel('{} values'.format(feature))
    plt.xticks(rotation=45)
    plt.title("{} values wrt count of records".format(feature),fontsize=15,
```

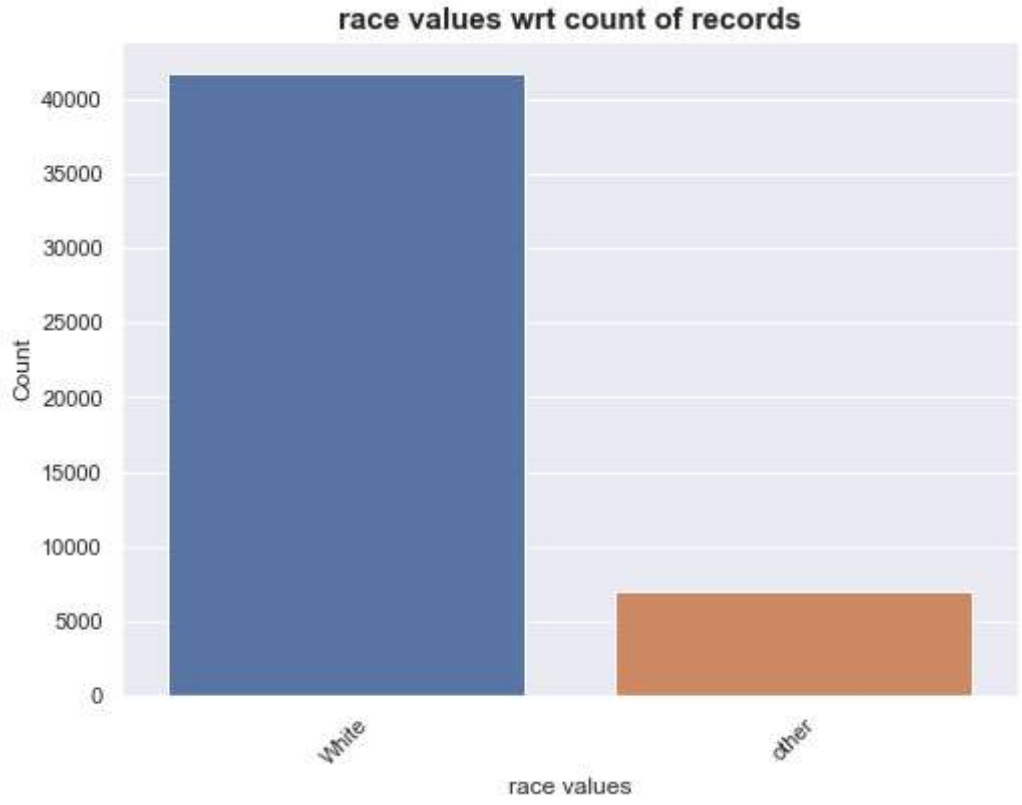


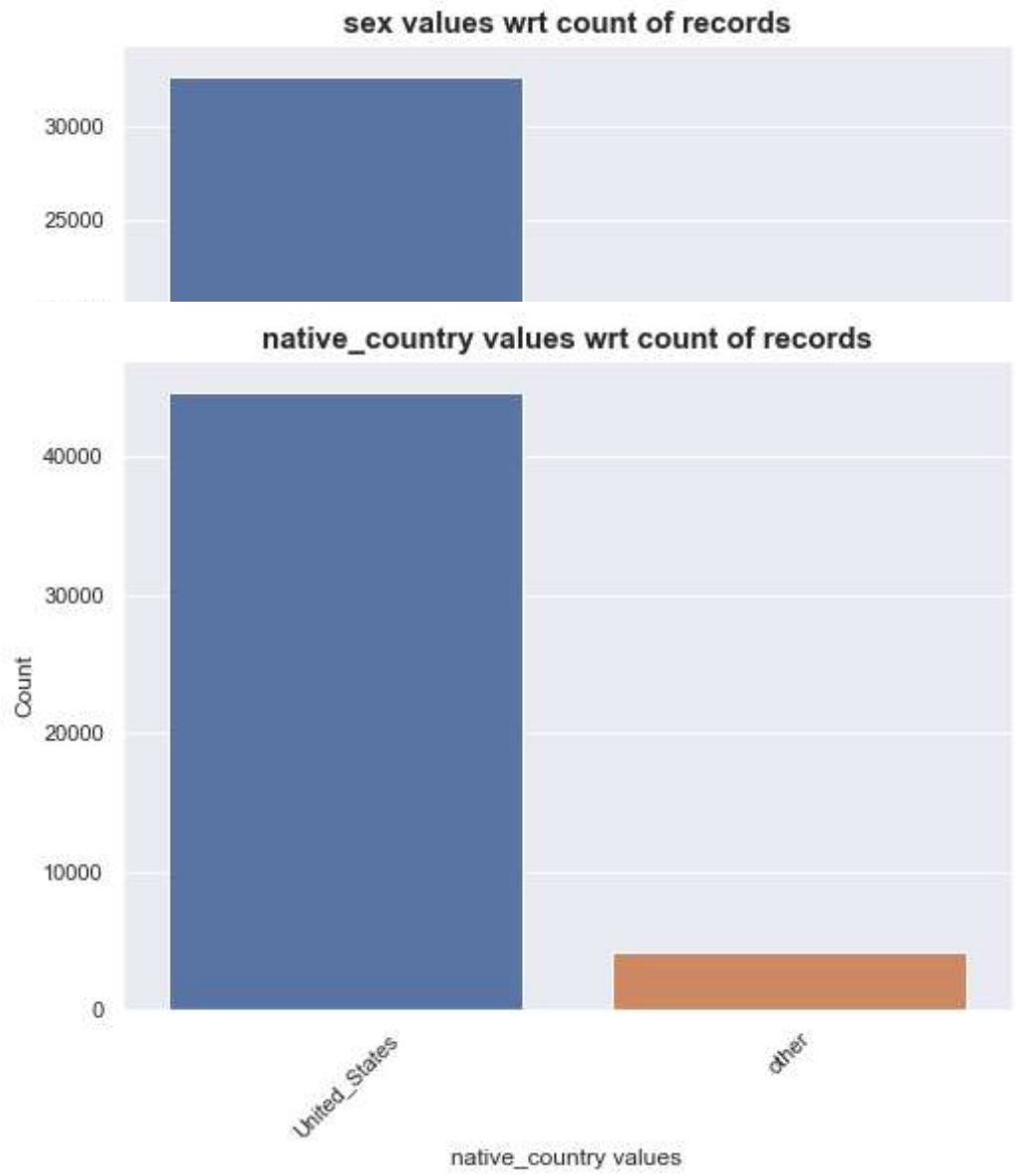

education values wrt count of records
16000
14000
12000
10000
8000
6000
4000
2000
0
Count
HS_grad
other
Some_college
Bachelors
education values

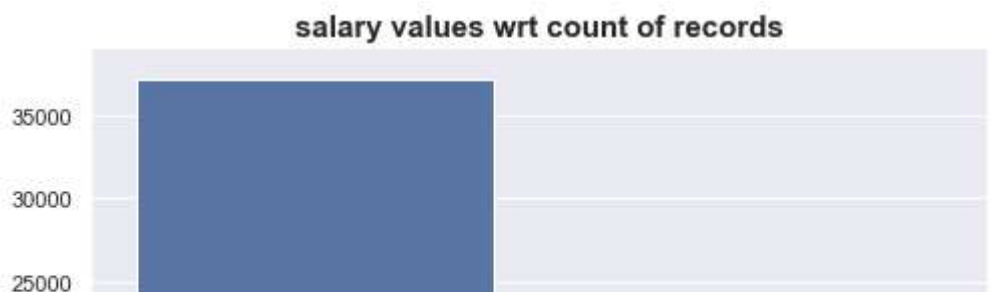

marital_status values wrt count of records
20000
15000
10000
5000
0
Count
Married_civ_spouse
Never_married
Divorced
other
marital_status values


occupation values wrt count of records
12000
10000
8000
6000
4000
2000
0
Count
other
Prof_specialty
Craft_repair
Exec_managerial
Adm_clerical
Sales
Other_service
occupation values


relationship values wrt count of records
Count
20000
17500
15000
12500
10000
7500


race values wrt count of records
Count
40000
35000
30000
25000
20000
15000
10000
5000
0
White
other
race values


sex values wrt count of records
30000
25000
native_country values wrt count of records
40000
30000
20000
10000
0
Count
United_States
other
native_country values



6.0 Encoding Dependent feature

In [52]:
```
dataset['salary']=dataset['salary'].replace('<=50K','0')
dataset['salary']=dataset['salary'].replace('>50K','1')
dataset['salary']=dataset['salary'].astype('int64')
```

In [53]:
```
dataset.head()
```

Out[53]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relatio
0	39	other	77516	Bachelors	13	Never_married	Adm_clerical	Not_in_
1	50	other	83311	Bachelors	13	Married_civ_spouse	Exec_managerial	Hus
2	38	Private	215646	HS_grad	9	Divorced	other	Not_in_
3	53	Private	234721	other	7	Married_civ_spouse	other	Hus
4	28	Private	338409	Bachelors	13	Married_civ_spouse	Prof_specialty	

7.0 Correlation and HeatMap

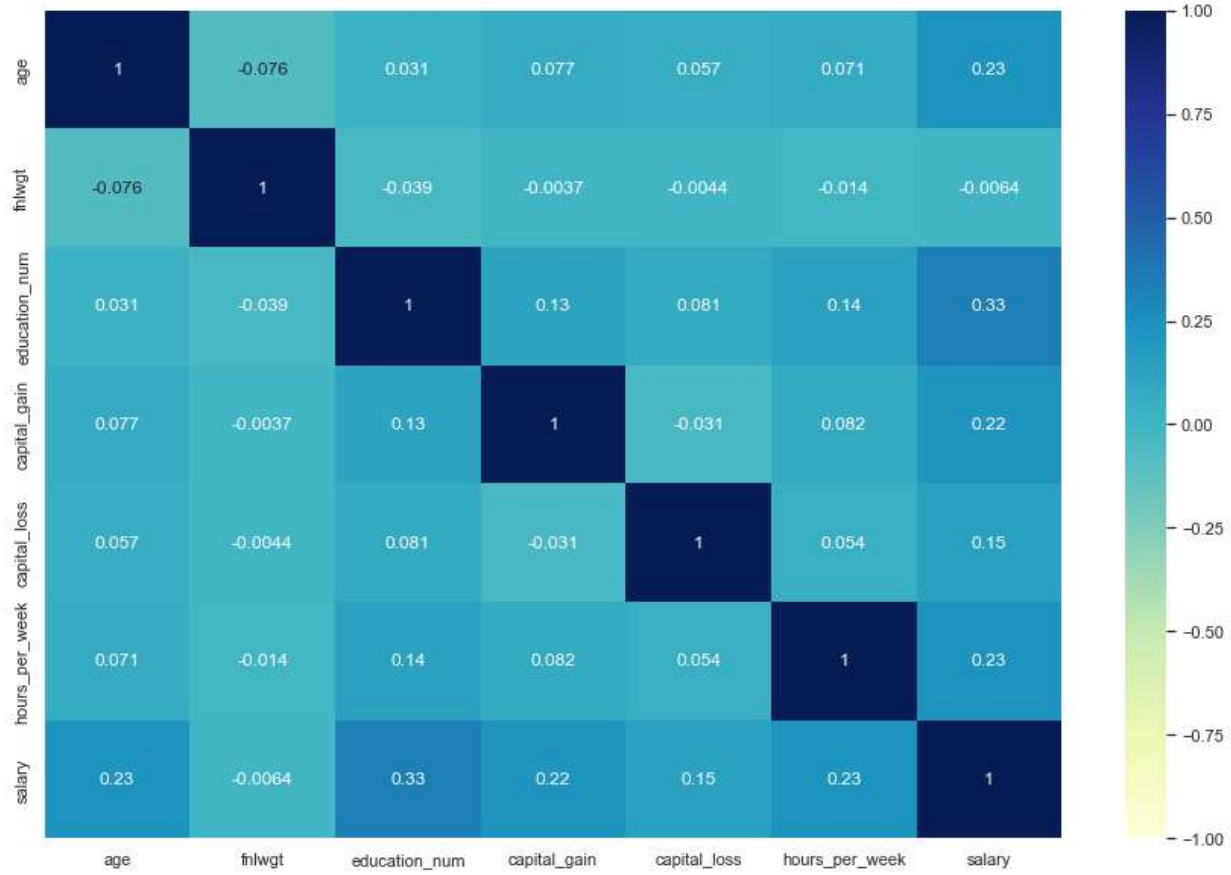
In [55]:
```
corr=dataset.corr()
corr
```

Out[55]:

	age	fnlwgt	education_num	capital_gain	capital_loss	hours_per_week
age	1.000000	-0.076474	0.030760	0.077221	0.056908	0.071322
fnlwgt	-0.076474	1.000000	-0.038798	-0.003713	-0.004375	-0.013516
education_num	0.030760	-0.038798	1.000000	0.125186	0.080969	0.143872
capital_gain	0.077221	-0.003713	0.125186	1.000000	-0.031460	0.082154
capital_loss	0.056908	-0.004375	0.080969	-0.031460	1.000000	0.054440
hours_per_week	0.071322	-0.013516	0.143872	0.082154	0.054440	1.000000
salary	0.230335	-0.006376	0.332746	0.223014	0.147527	0.227649

In [56]:
```
### Plotting heatmap for visualising the correlation between features
sns.set(rc={'figure.figsize':(15,10)})
sns.heatmap(data=corr, annot=True,  vmin=-1, vmax=1, cmap="YlGnBu")
```

Out[56]: <AxesSubplot:>



8.0 Uplodaing data to MongoDB

In [ ]:
```
### creating connection with MongoDB
import pymongo
client = pymongo.MongoClient("mongodb+srv://{username}:{password}@clustershu
```

In [ ]:
```
### creating database and collection in MongoDB
db=client['Census_income']
collection=db['Census_income_data']
```

In [ ]:
```
### Converting dataframe to dict so it can be uploaded to MongoDB
dataset.reset_index(inplace=True)
data_dict = dataset.to_dict("records")
```

In [ ]:
```
# Insert collection to MongoDB
collection.insert_many(data_dict)
```

In [ ]:
```
dataset.to_csv('Census_income_cleaned.csv')
```