01_analyse_raw_data

January 16, 2020

1 EDA on raw data (after collect)

This notebook main goal is to : - Understand what is in the data : plot variables one by one, missing values, etc. - See which data are correlated - Removes useless variables : too much missing values, too correlated with other variables. - See if data is biased on protected attributes

1.1 Load packages

[1]:
```python
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import display, Markdown

# Custom helper package : https://github.com/Nathanlauga/MLHelper
import MLHelper.analyse.eda as eda
import MLHelper.utils as utils
```

[2]:
```python
PROJECT_PATH = '..'
```

1.2 Load data informations

[3]:
```python
df_details = pd.read_csv(f'{PROJECT_PATH}/01_collect/columns_informations.csv')
```

[4]:
```python
df_details
```

[4]:
```
  dataset           column   dtype  is_protected      new_name  \
0   adult              age   int64             1           NaN
1   adult        workclass  object             0           NaN
2   adult           fnlwgt   int64             0  final_weight
3   adult        education  object             0           NaN
4   adult    educational-num   int64             0           NaN
5   adult   marital-status  object             1           NaN
6   adult       occupation  object             0           NaN
7   adult     relationship  object             0           NaN
8   adult             race  object             1           NaN
```

```
9    adult          gender  object             1          NaN
10   adult    capital-gain   int64             0          NaN
11   adult    capital-loss   int64             0          NaN
12   adult  hours-per-week   int64             0          NaN
13   adult  native-country  object             1          NaN
14   adult          income  object             0          NaN

                                          description
0                                  Age of the person
1              Workclass of the person (e.g. Private)
2   final weight, which is the number of units in …
3                                     Education level
4                  Education level (numerical format)
5                                      Marital status
6                                    Field occupation
7                                Current relationship
8                                           Ethnicity
9                                      Female or Male
10                                    Gain of capital
11                                    Loss of capital
12                number of working hours during a week
13                                     Native country
14                        Target : income > 50K or not
```

1.3 Load data using data informations

[5]:

```python
dfs = {}
datasets = df_details['dataset'].unique()

for dataset in datasets:
    dataset_detail = df_details[df_details['dataset'] == dataset]

    dtypes, columns = dataset_detail['dtype'], dataset_detail['column']
    dtypes = pd.Series(dtypes.values,index=columns)

    parse_dates = list()
    for var, dtype in dtypes.iteritems():
        if 'datetime' in dtype:
            parse_dates.append(var)

    dtypes = dtypes.str.replace(r'datetime.*','str')

    dfs[dataset] = pd.read_csv(f'{PROJECT_PATH}/_data/{dataset}.csv',
                               dtype=dtypes.to_dict(),
                               parse_dates=parse_dates
                              )
```

```
    columns = np.where(dataset_detail['new_name'].isna(), columns,␣
↪dataset_detail['new_name'])
    dfs[dataset].columns = columns
```

1.4 Display shape & head

[6]:
```
for dataset in dfs:
    display(Markdown(f'#### {dataset}, {dfs[dataset].shape}'))
    display(dfs[dataset].head())
```

adult, (48842, 15)

```
   age  workclass  final_weight     education  educational-num  \
0   25    Private        226802          11th                7
1   38    Private         89814       HS-grad                9
2   28  Local-gov        336951    Assoc-acdm               12
3   44    Private        160323  Some-college               10
4   18          ?        103497  Some-college               10

       marital-status         occupation relationship   race  gender  \
0       Never-married  Machine-op-inspct    Own-child  Black    Male
1  Married-civ-spouse    Farming-fishing      Husband  White    Male
2  Married-civ-spouse    Protective-serv      Husband  White    Male
3  Married-civ-spouse  Machine-op-inspct      Husband  Black    Male
4       Never-married                  ?    Own-child  White  Female

   capital-gain  capital-loss  hours-per-week native-country income
0             0             0              40  United-States  <=50K
1             0             0              50  United-States  <=50K
2             0             0              40  United-States   >50K
3          7688             0              40  United-States   >50K
4             0             0              30  United-States  <=50K
```

1.5 Set datasets into differents variables

Here we just have one dataset `adult`, but if in the next versions we had some others datasets it's important to set them into distinct variables.

[7]:
```
adult = dfs['adult']
del dfs
```

1.6 Analyse : missing values

[8]:
```
help(eda.print_missing_values)
```

```
Help on function print_missing_values in module MLHelper.analyse.eda.eda:

print_missing_values(df)
    Show a bar plot that display percentage of missing values on columns that
have some.
    If no missing value then it use `display` & `Markdown` functions to indicate
it.

    Parameters
    ----------
    df: pd.DataFrame
        Dataframe to inspect
```

[9]:
```
display(Markdown('#### Missing values for adult dataset'))
eda.print_missing_values(adult)
```

Missing values for adult dataset No missing value.

1.7 Analyse : plot each variable

1.7.1 Using MLHelper

[10]:
```
adult = eda.remove_var_with_one_value(adult)
```

[11]:
```
help(eda.show_df_vars)
```

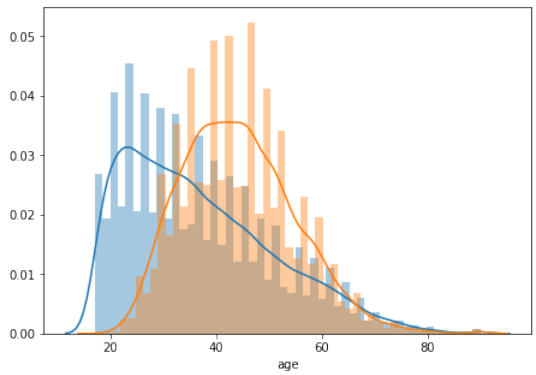
```
Help on function show_df_vars in module MLHelper.analyse.eda.eda:

show_df_vars(df, target=None)
    Show all variables with graphics to understand each variable.
    If target is set, complement visuals will be added to take a look on the
    influence that a variable can have on target

    Data type handle : categorical, numerical, datetime

    Parameters
    ----------
    df: pd.DataFrame
        Dataframe to inspect
    target: str (optional)
        Target column for classifier
```

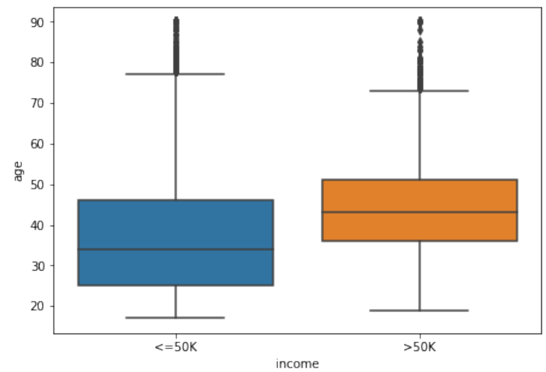
[12]:
```
eda.show_df_vars(df=adult, target='income')
```

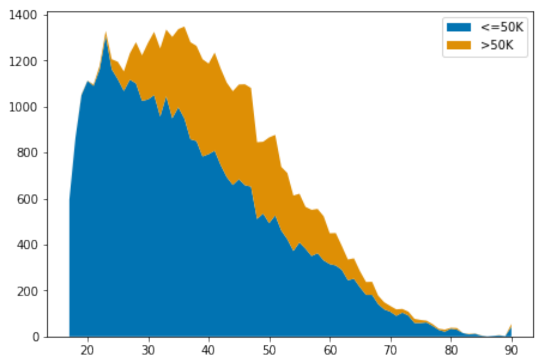
1.7.2 Numerical variables

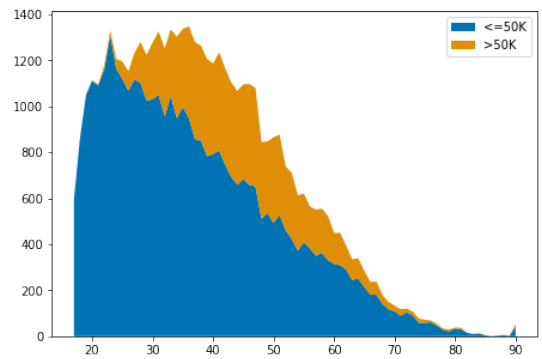
---

age : 0 nulls, 74 unique vals, most common: {36: 1348, 35: 1337}



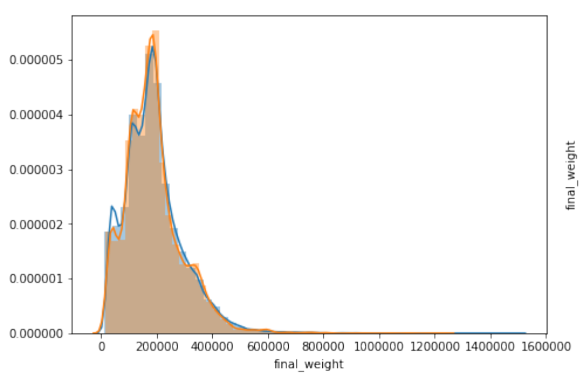


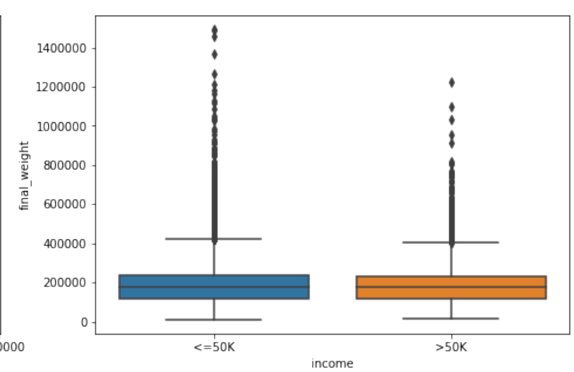


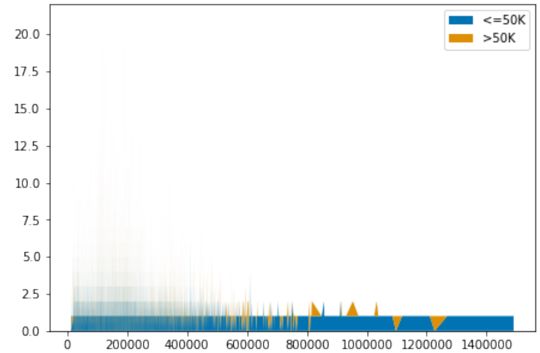


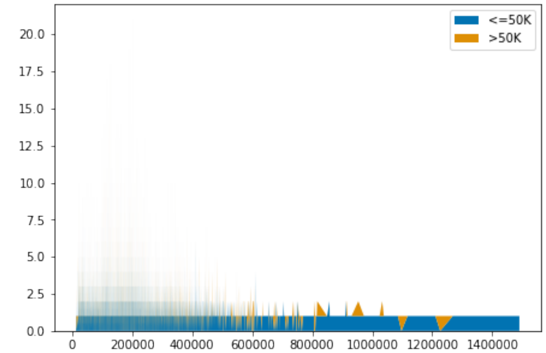
---

final_weight : 0 nulls, 28523 unique vals, most common: {203488: 21, 190290: 19}



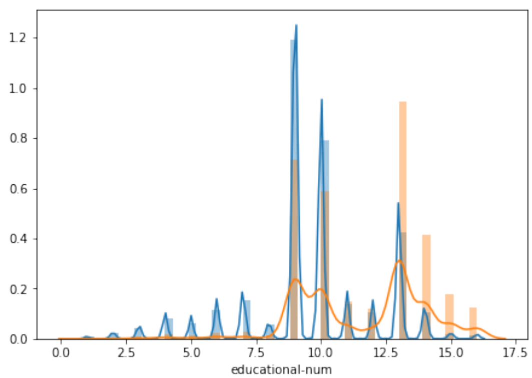


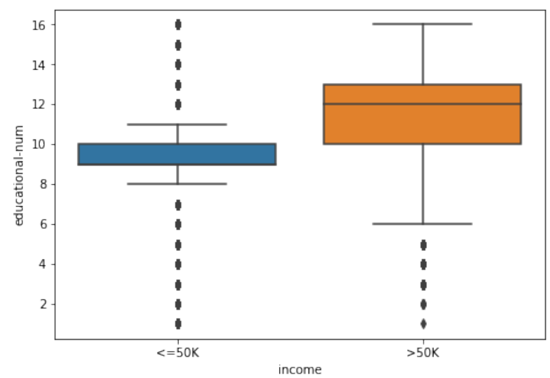


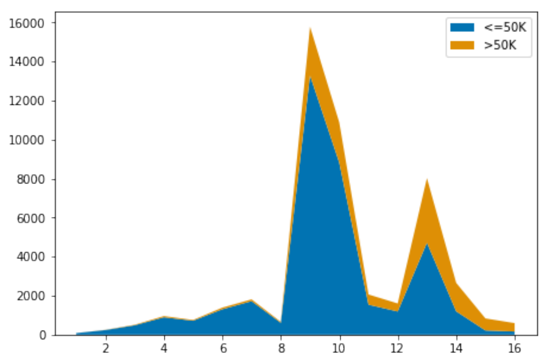


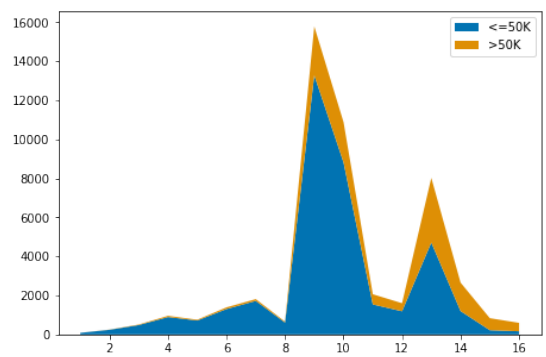
---

educational-num : 0 nulls, 16 unique vals, most common: {9: 15784, 10: 10878}



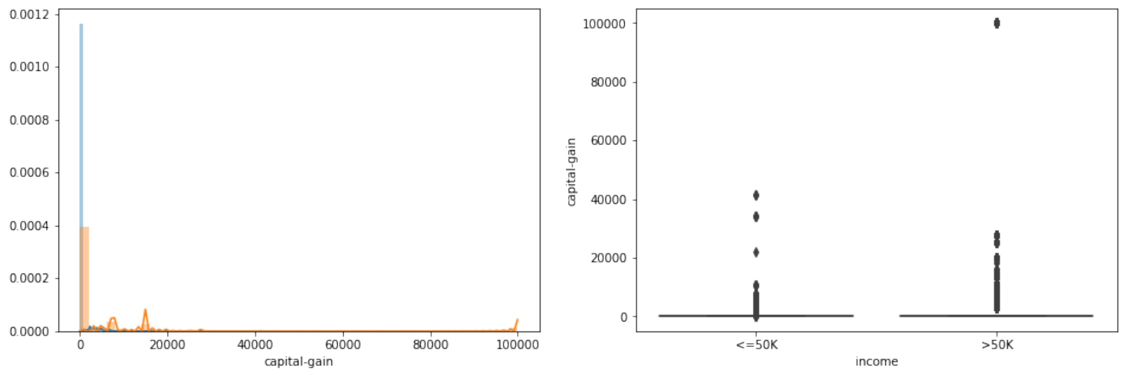


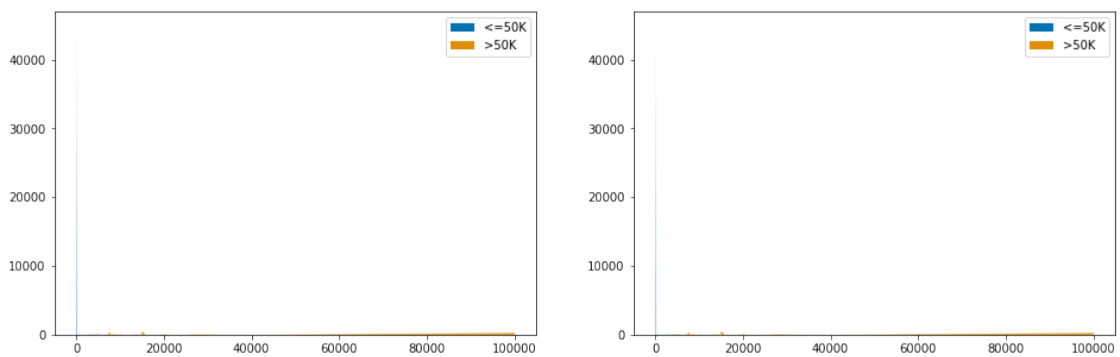




---

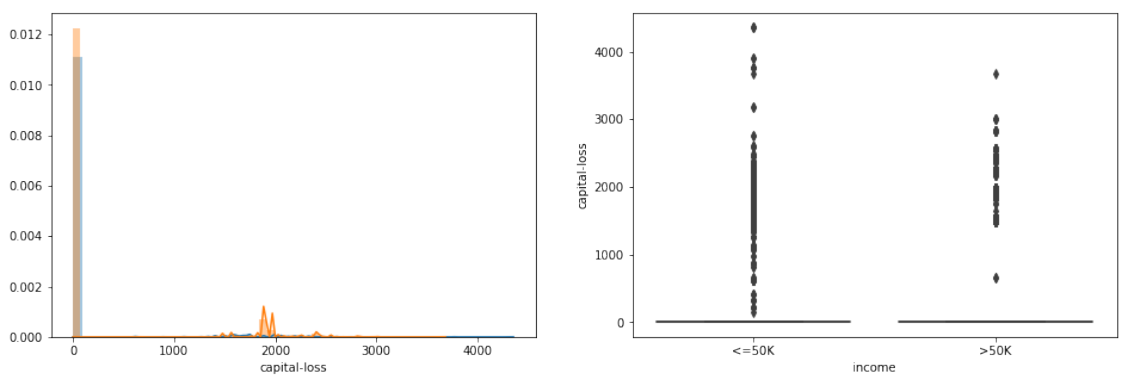
capital-gain : 0 nulls, 123 unique vals, most common: {0: 44807, 15024: 513}

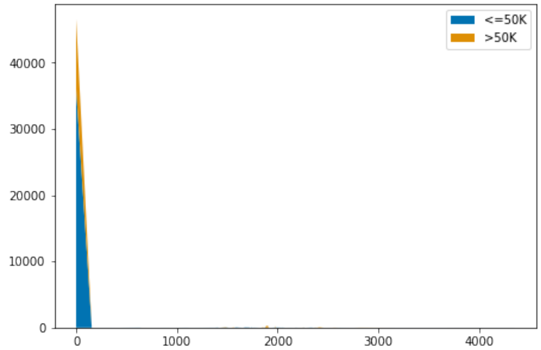


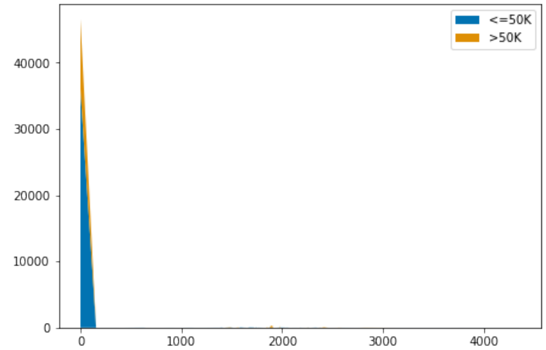


---

capital-loss : 0 nulls, 99 unique vals, most common: {0: 46560, 1902: 304}

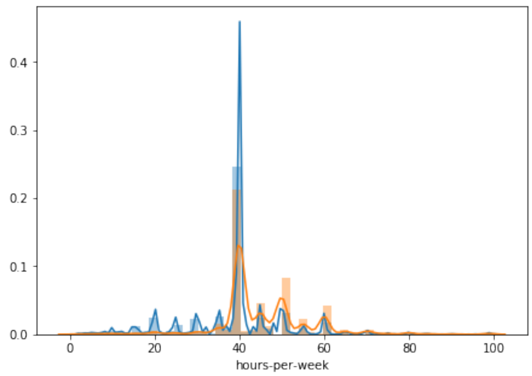


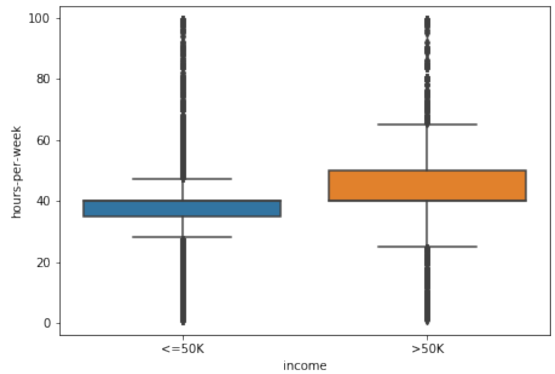


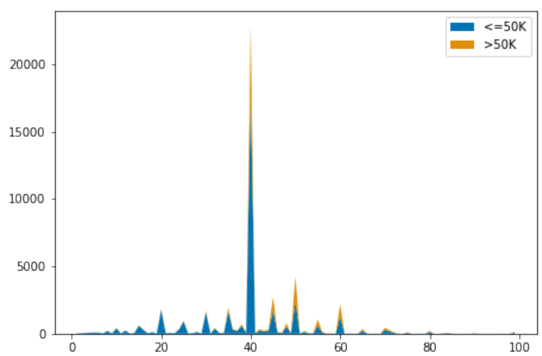


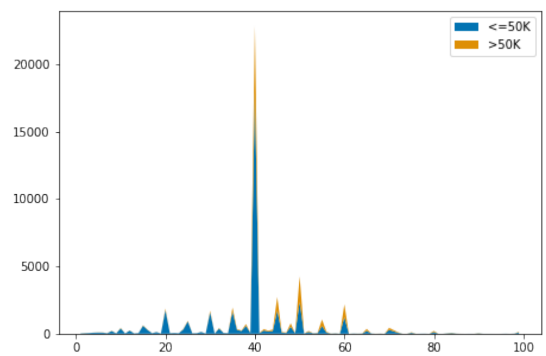
---

hours-per-week : 0 nulls, 96 unique vals, most common: {40: 22803, 50: 4246}





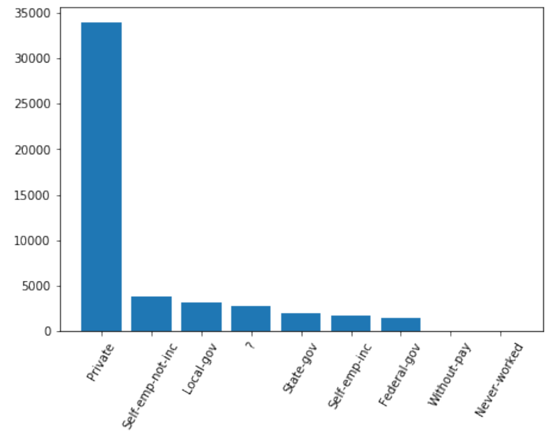


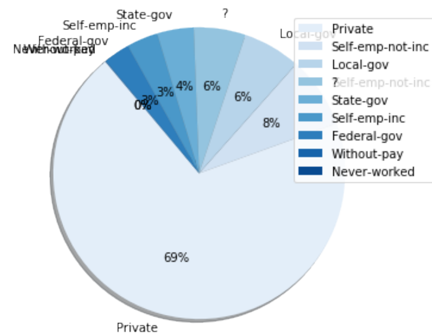


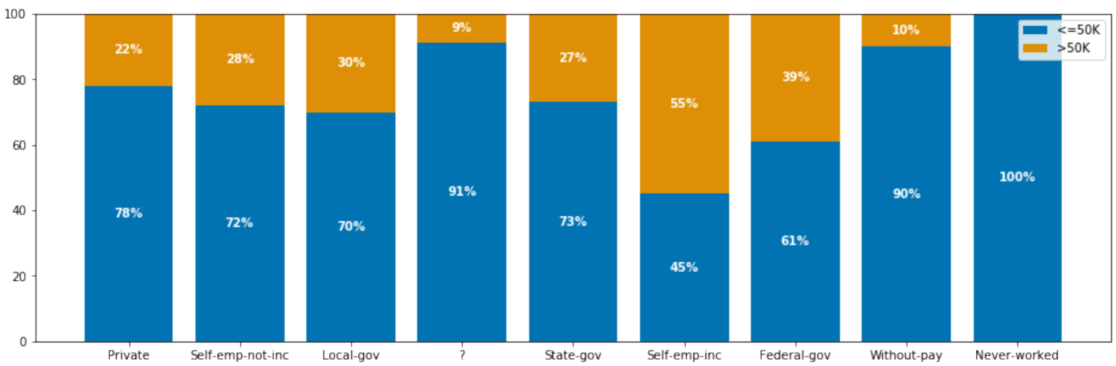
1.7.3 Categorical variables

---

workclass : 0 nulls, 9 unique vals, most common: {'Private': 33906, 'Self-emp-not-inc': 3862}

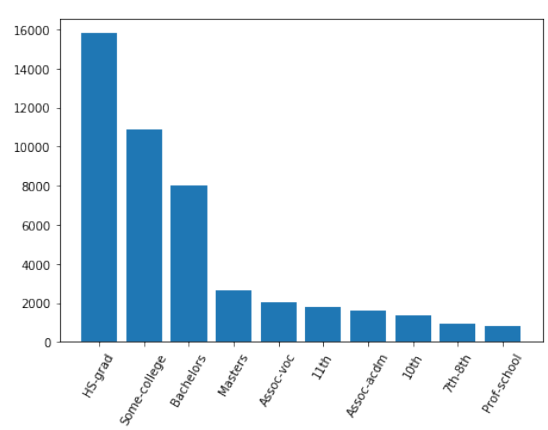


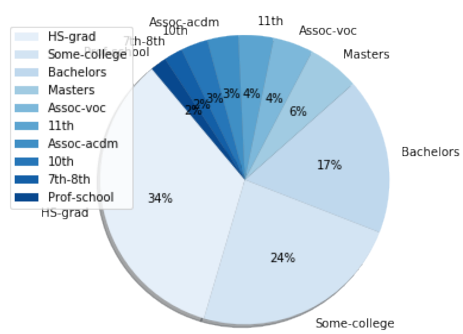


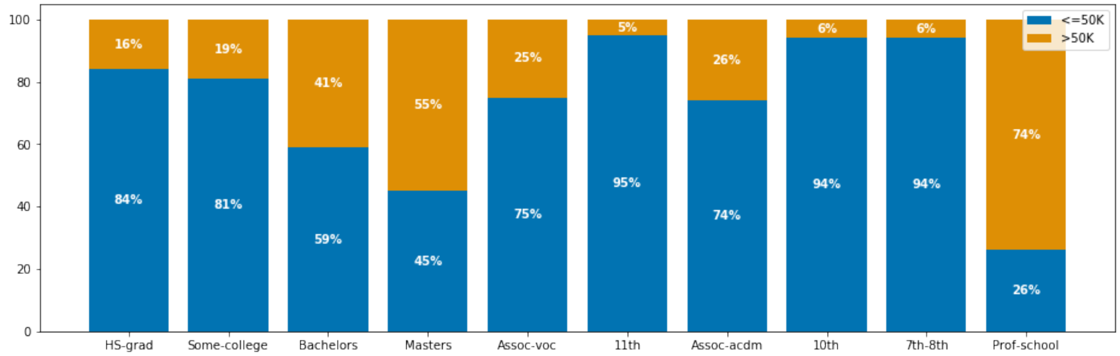


---

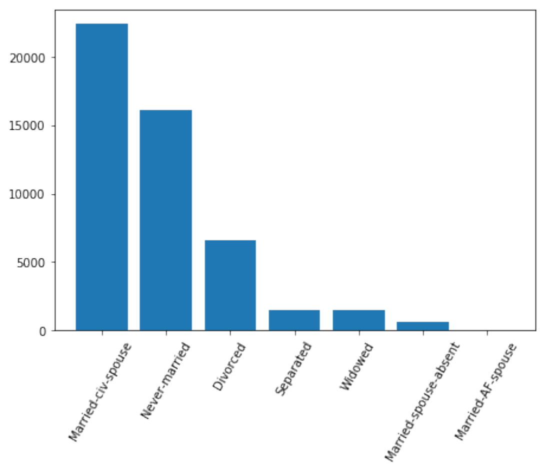
education : 0 nulls, 16 unique vals, most common: {‘HS-grad’: 15784, ‘Some-college’: 10878}

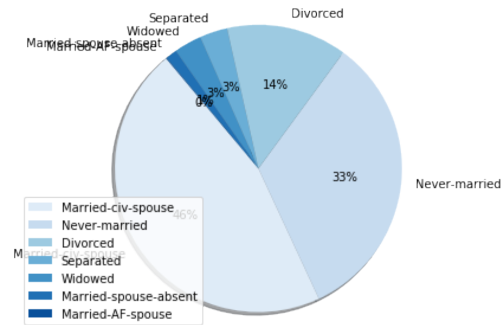


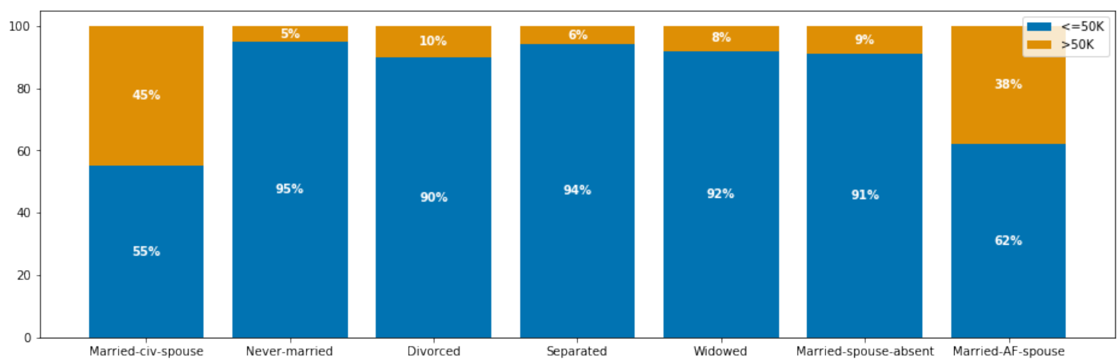




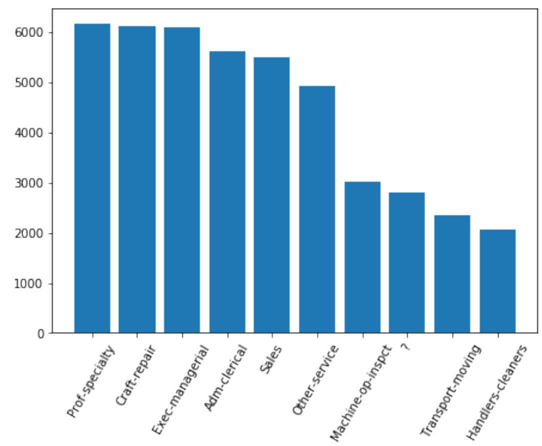
marital-status : 0 nulls, 7 unique vals, most common: {'Married-civ-spouse': 22379, 'Never-married': 16117}

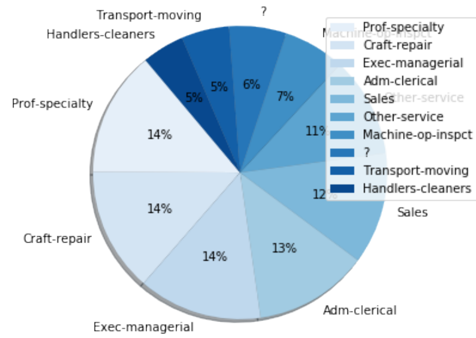


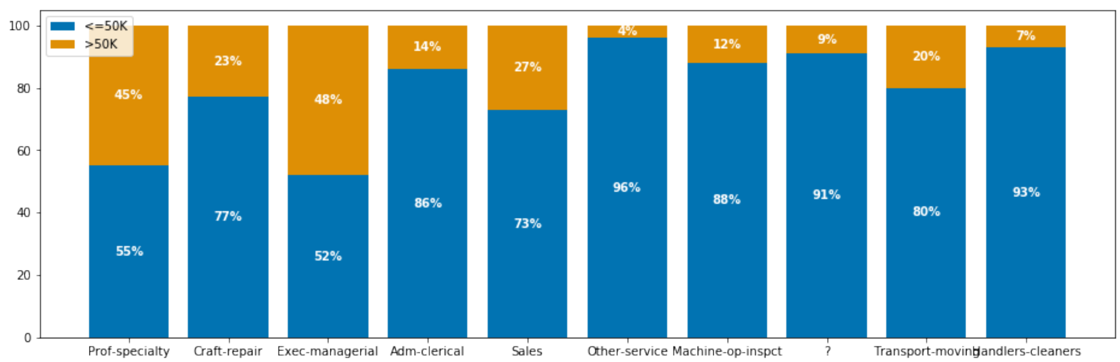




occupation : 0 nulls, 15 unique vals, most common: {'Prof-specialty': 6172, 'Craft-repair': 6112}

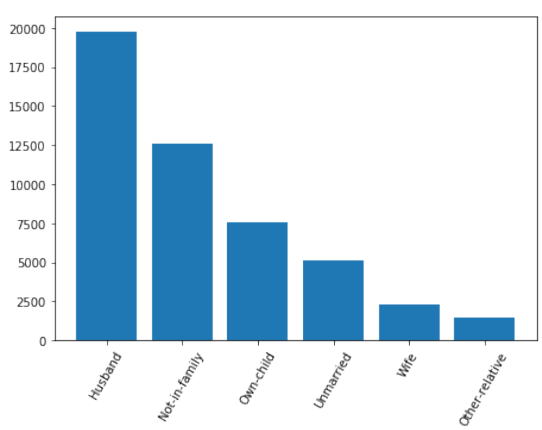


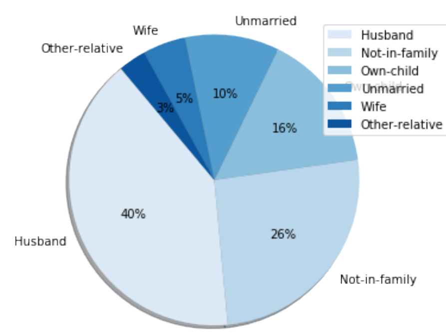


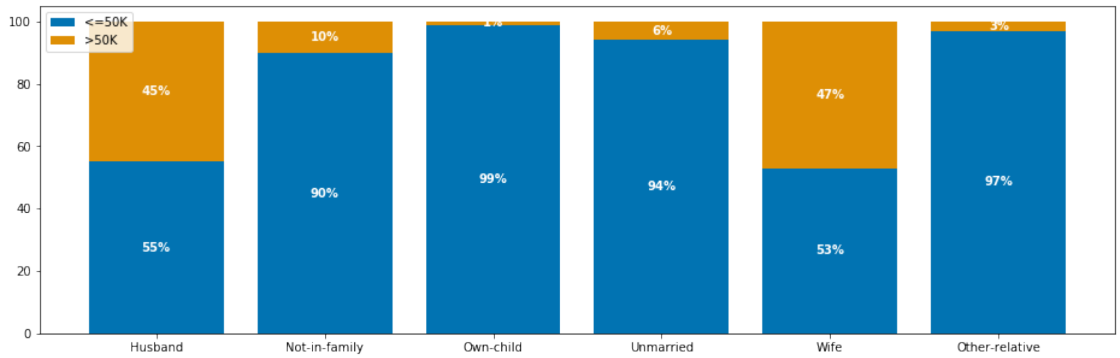


---

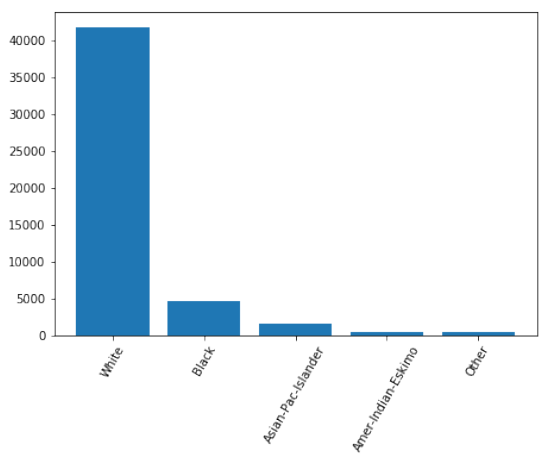
relationship : 0 nulls, 6 unique vals, most common: {'Husband': 19716, 'Not-in-family': 12583}

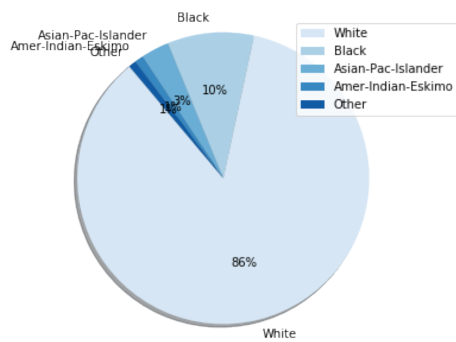


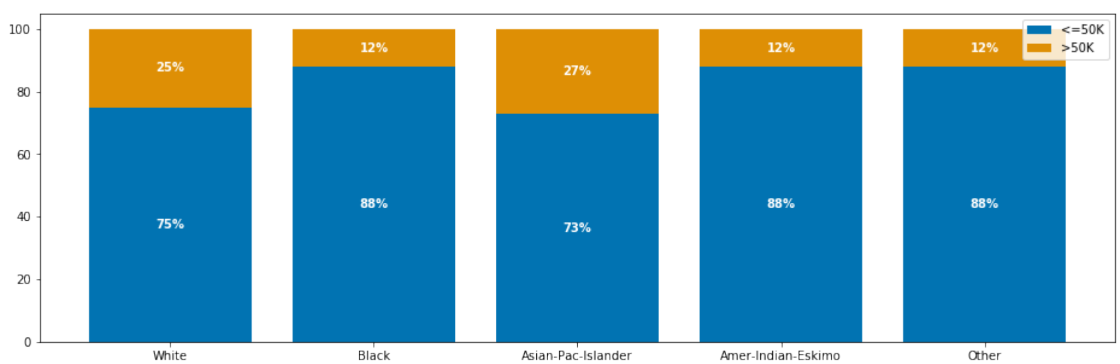




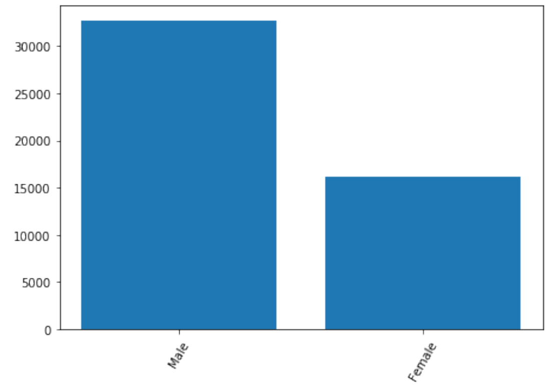
race : 0 nulls, 5 unique vals, most common: {'White': 41762, 'Black': 4685}

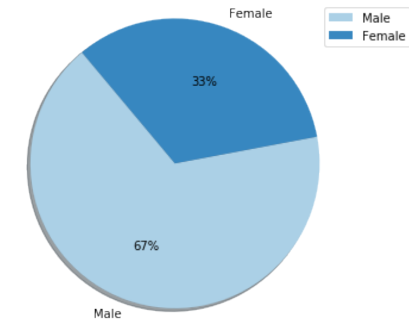


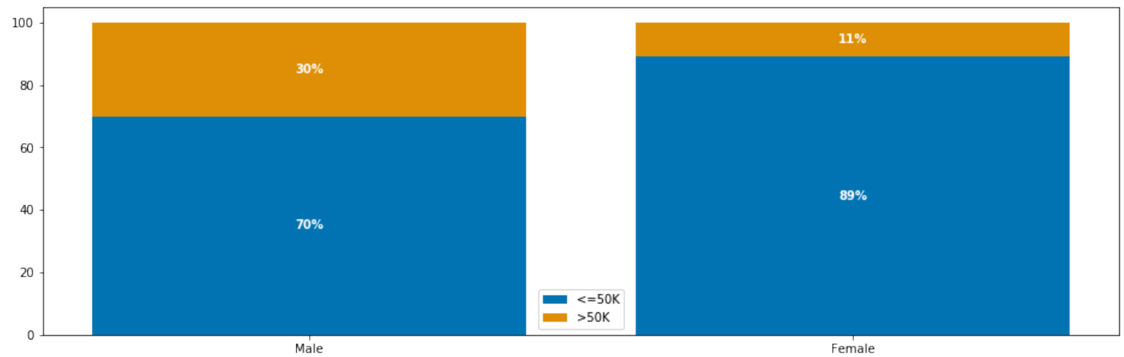




gender : 0 nulls, 2 unique vals, most common: {'Male': 32650, 'Female': 16192}

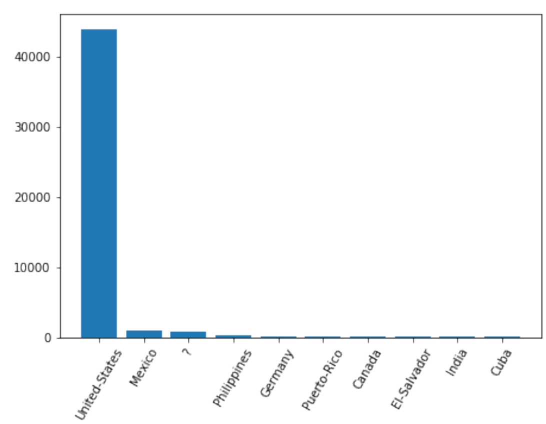


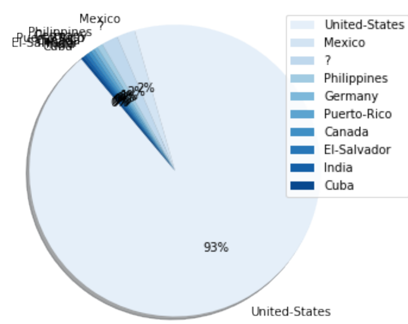


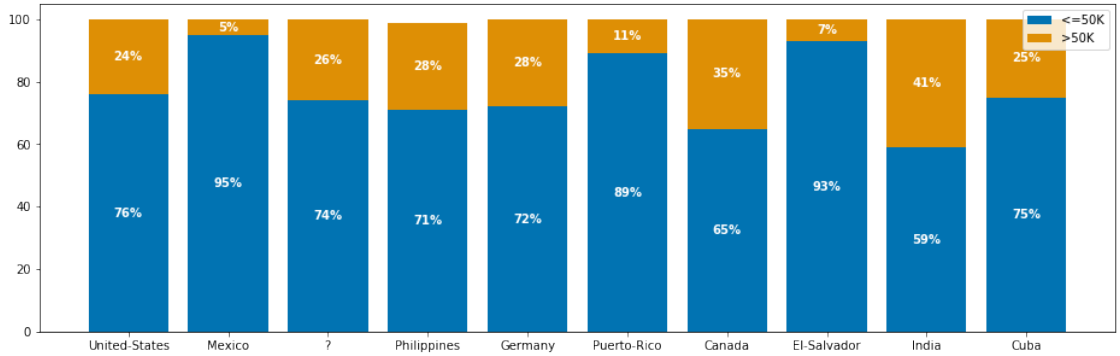


---

native-country : 0 nulls, 42 unique vals, most common: {'United-States': 43832, 'Mexico': 951}

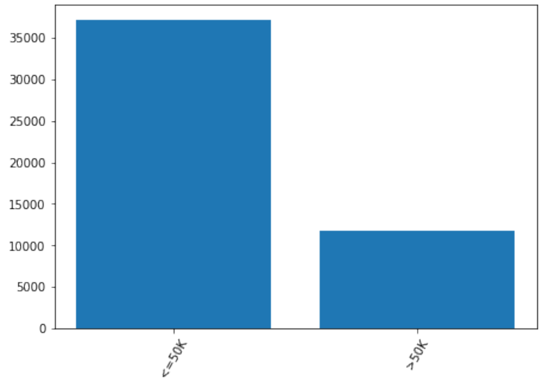


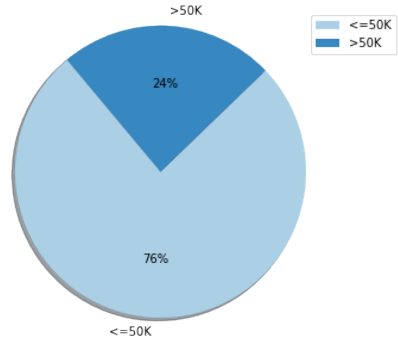


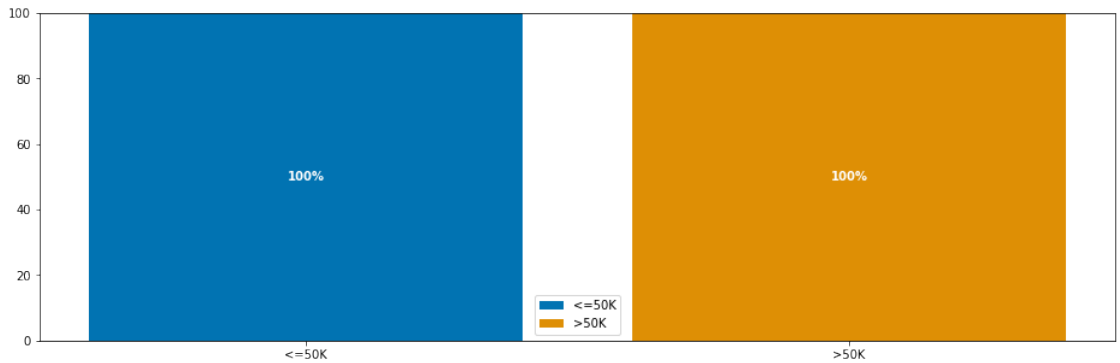


---

income : 0 nulls, 2 unique vals, most common: {'<=50K': 37155, '>50K': 11687}







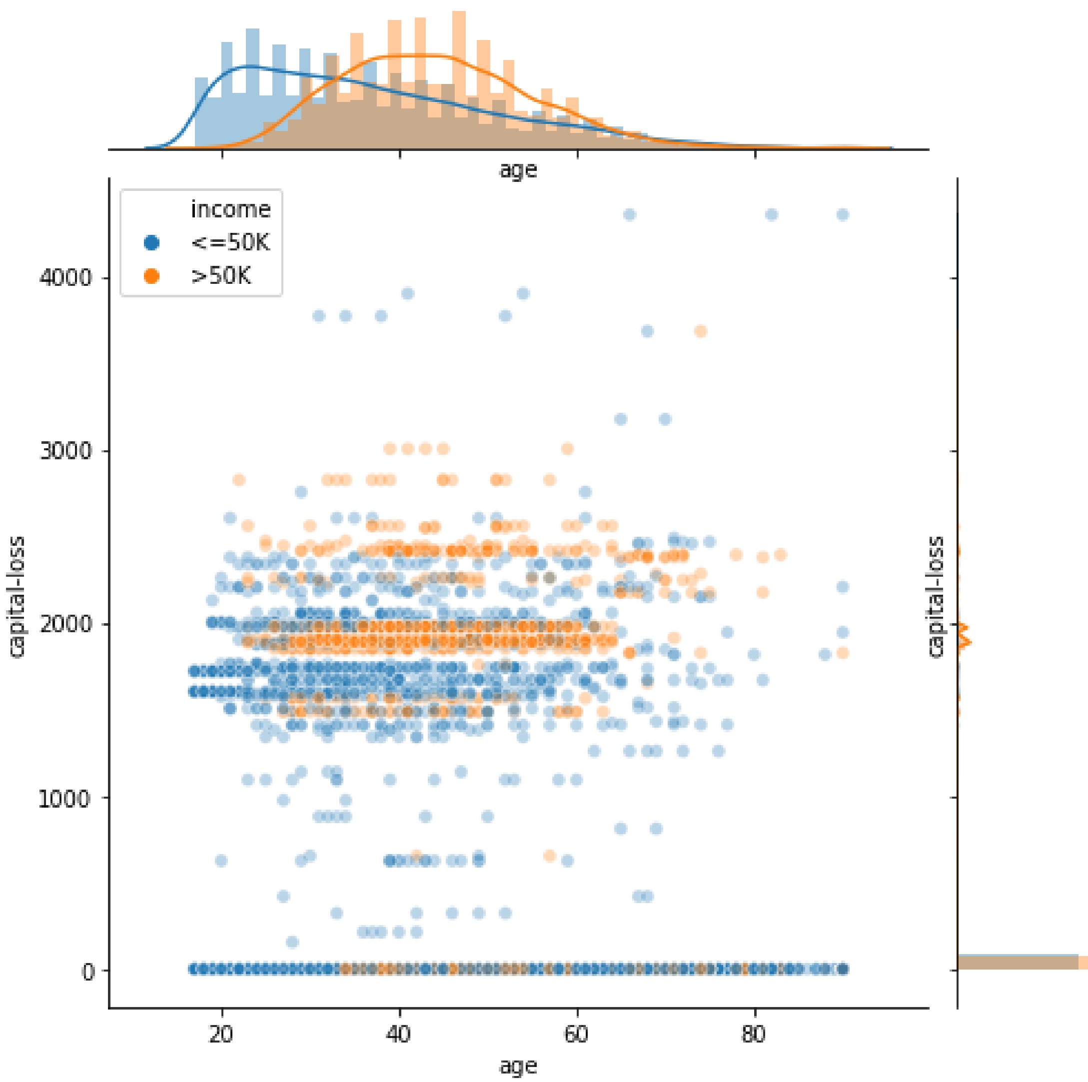
1.7.4 Datetime variables

[15]:
```
help(eda.show_df_numerical_relations)
```

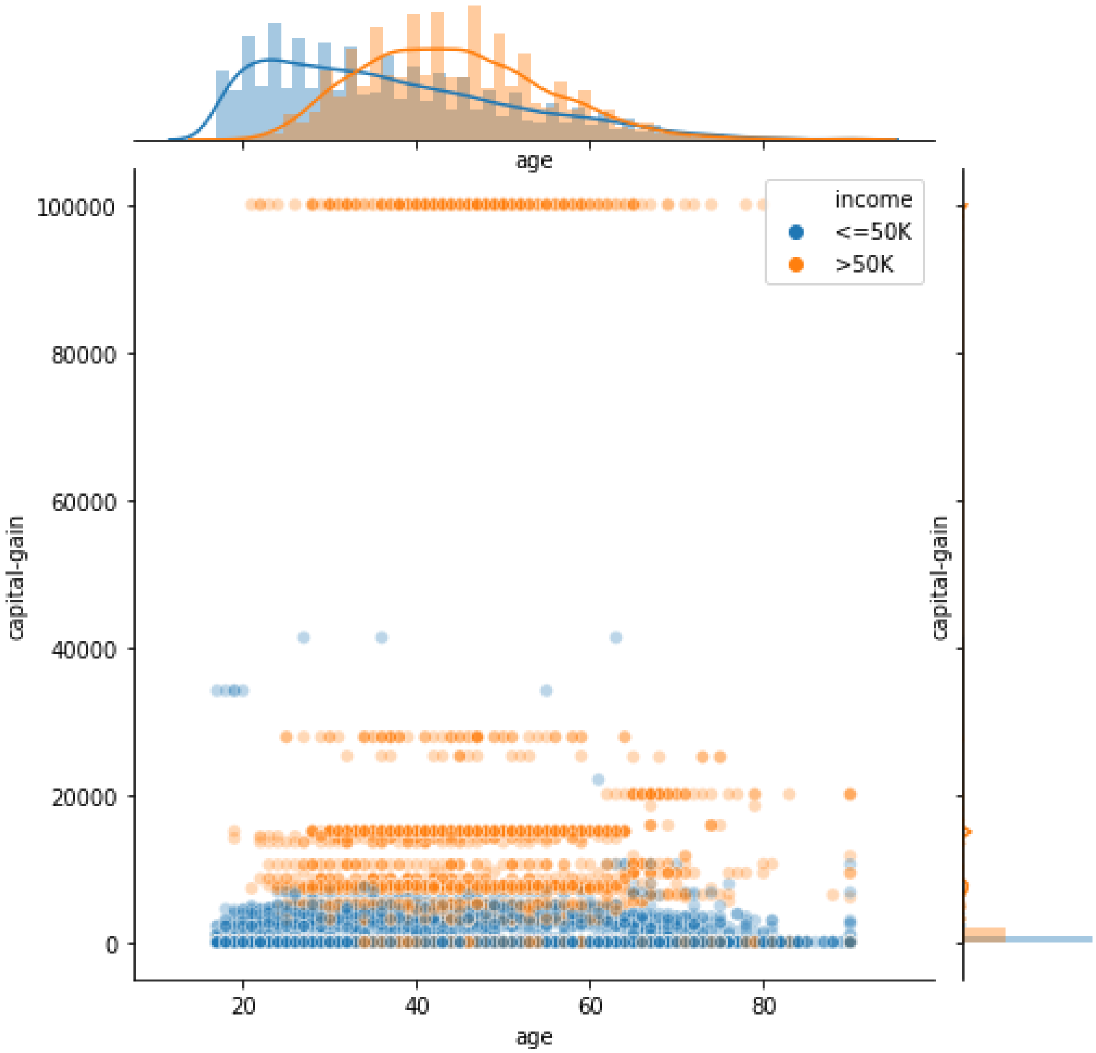
```
Help on function show_df_numerical_relations in module MLHelper.analyse.eda.eda:

show_df_numerical_relations(df, target=None)
    Show all numerical variables 2 by 2 with graphics understand their relation.
    If target is set, separate dataset for each target value.

    Parameters
    ----------
    df: pd.DataFrame
        Dataframe to inspect
    target: str (optional)
        Target column for classifier
```

[16]:
```
eda.show_df_numerical_relations(df=adult, target='income')
```

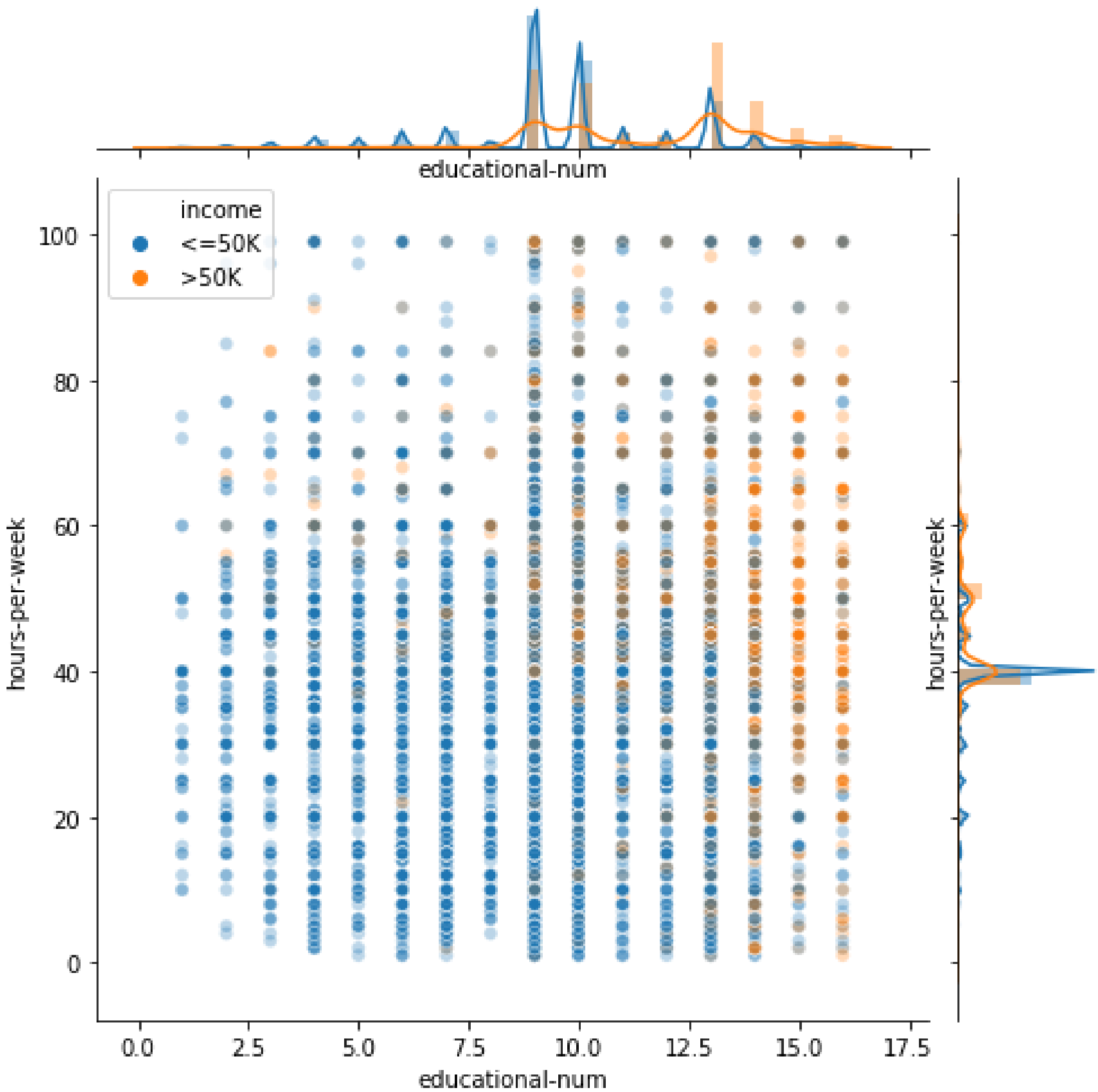
Joint plot for **age** & **capital-loss**



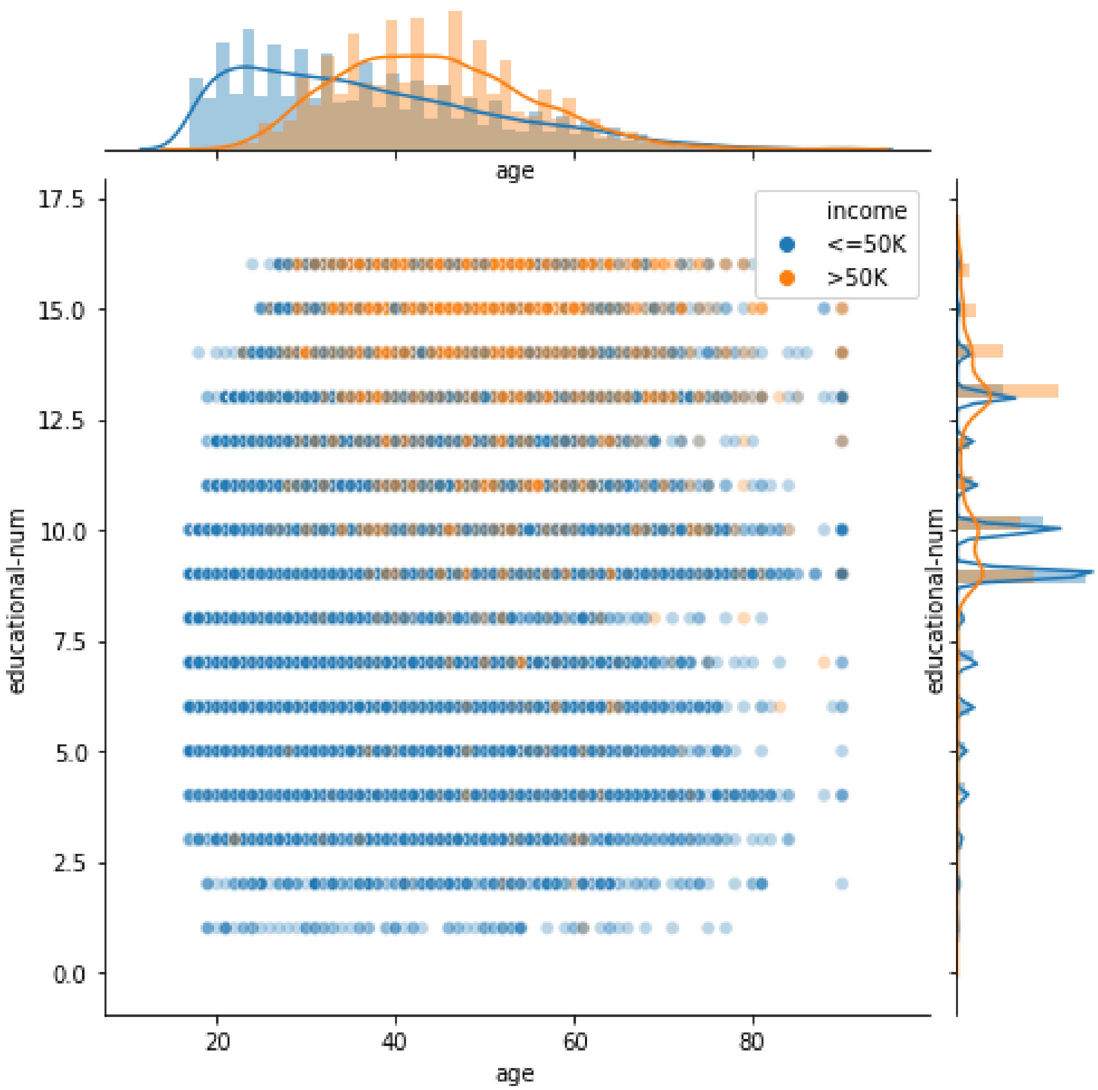
Joint plot for **age** & **capital-gain**



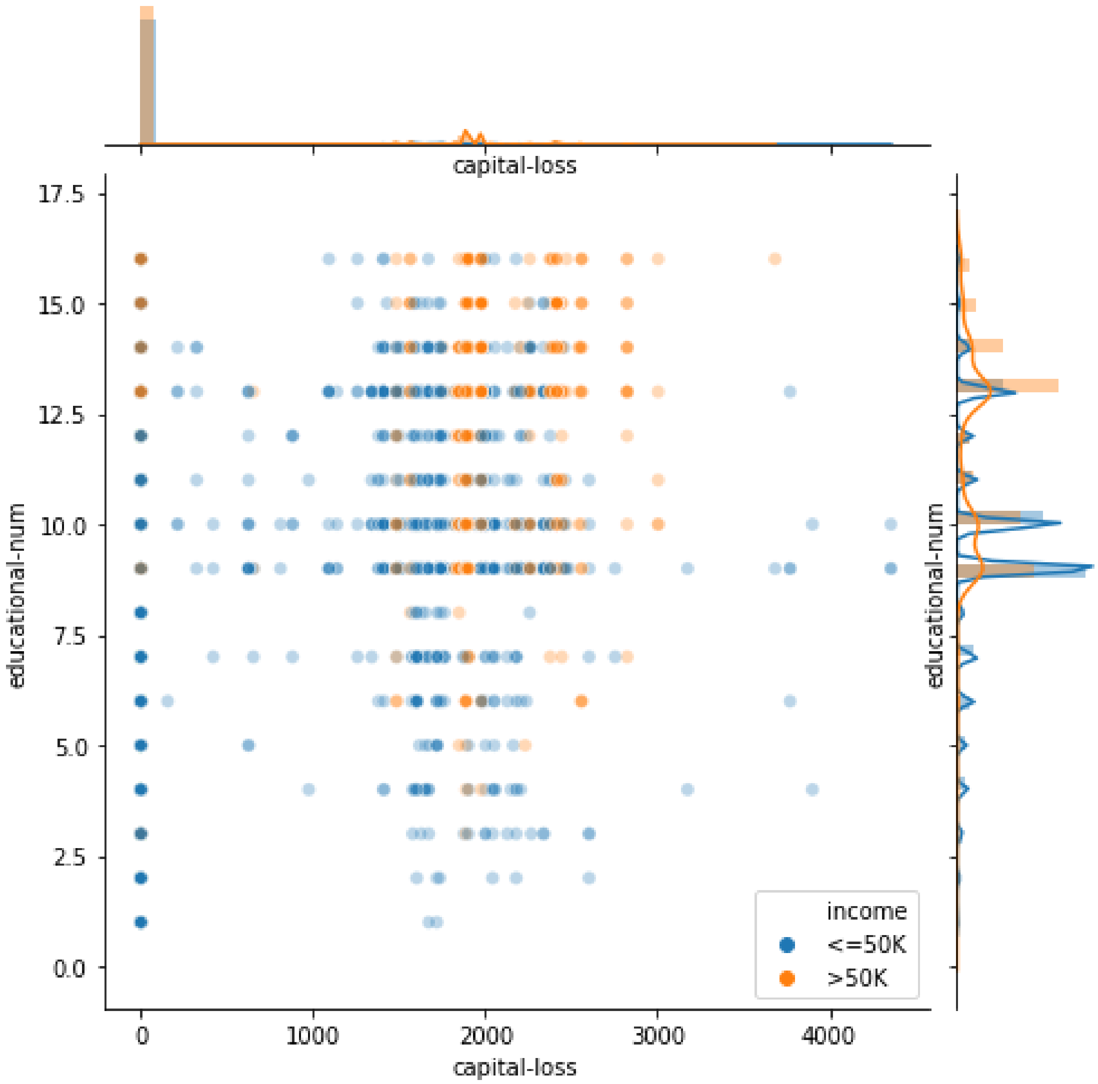
Joint plot for **educational-num** & **hours-per-week**



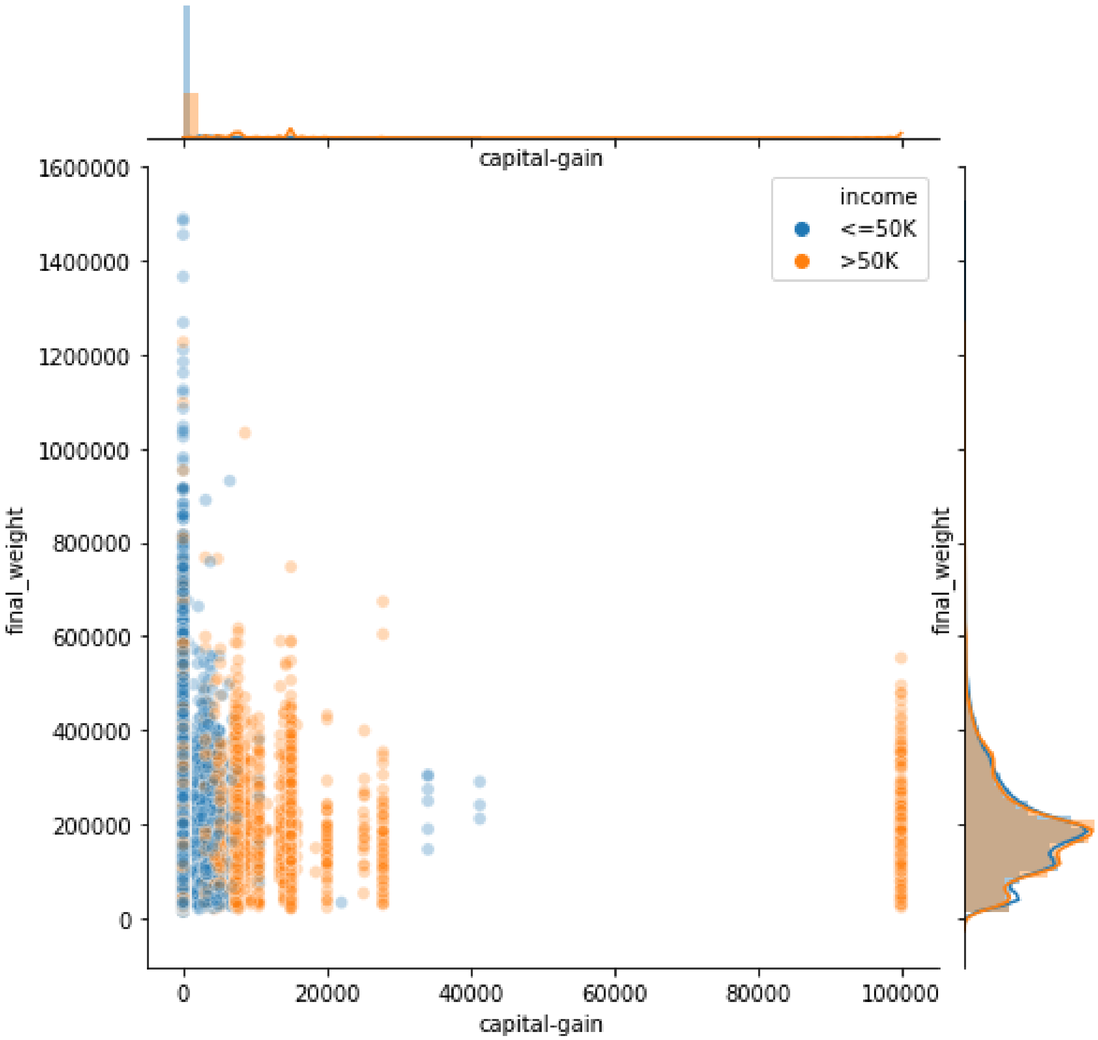
Joint plot for **age** & **educational-num**



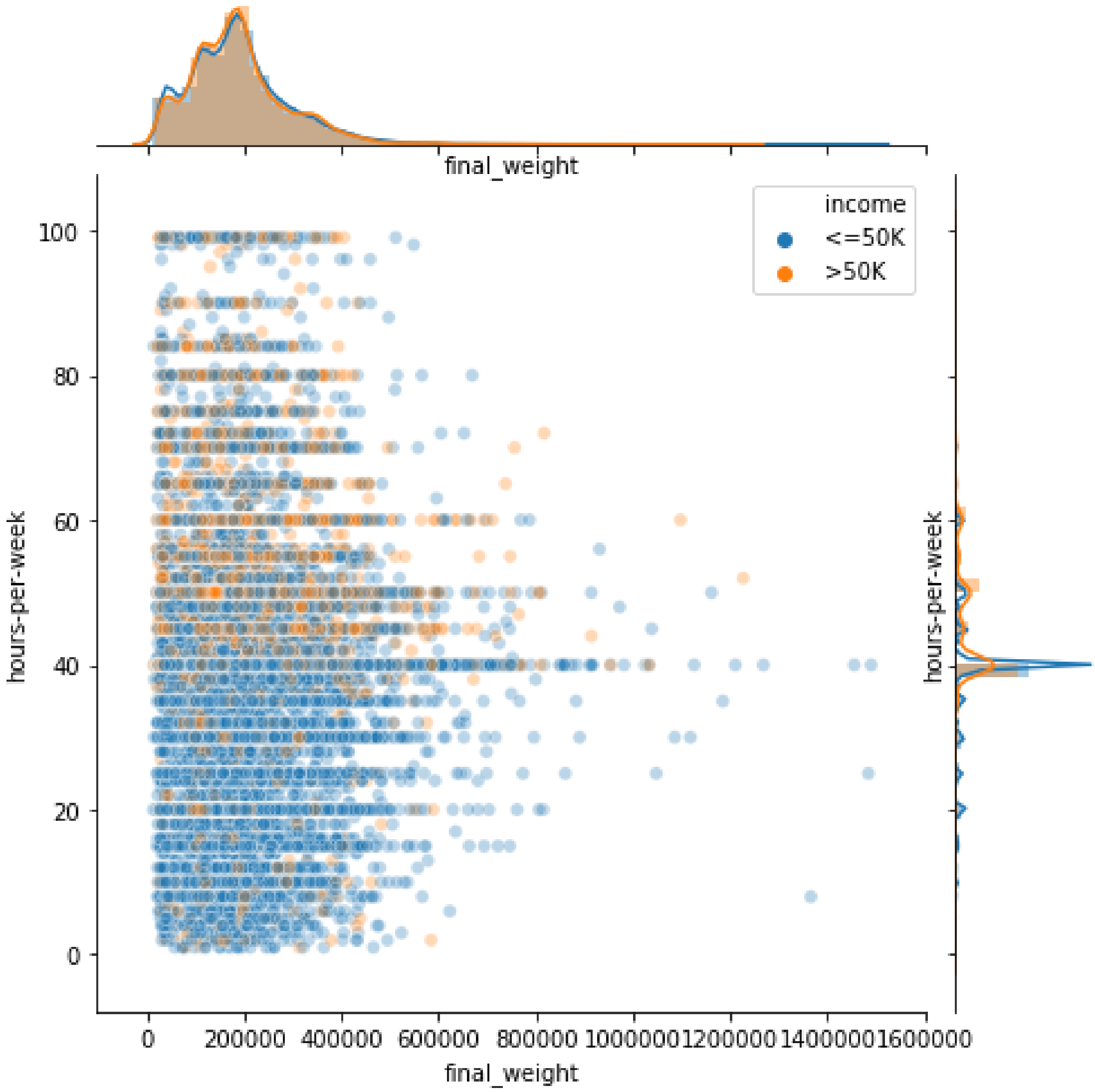
Joint plot for **capital-loss** & **educational-num**



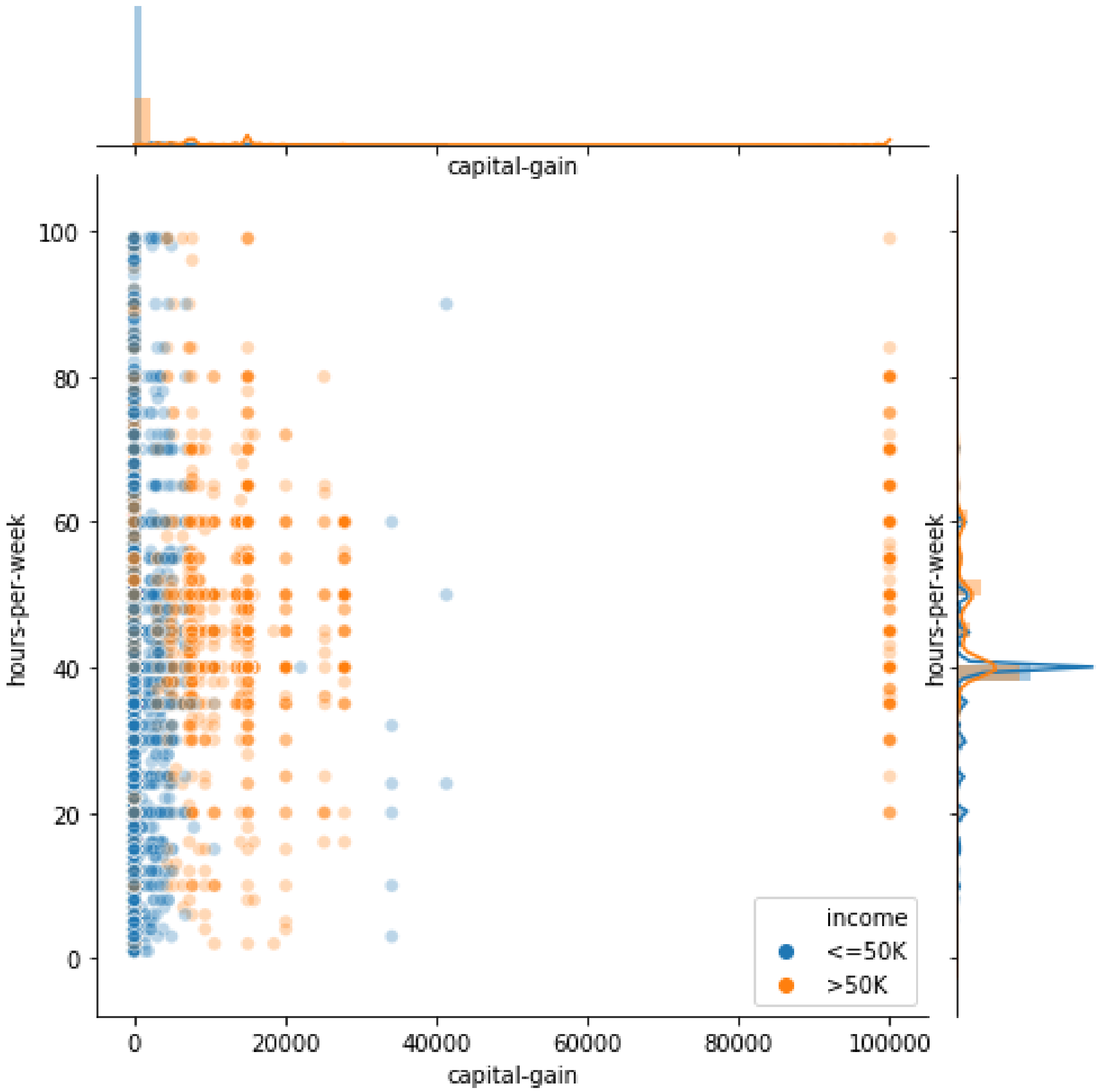
Joint plot for **capital-gain** & **final_weight**



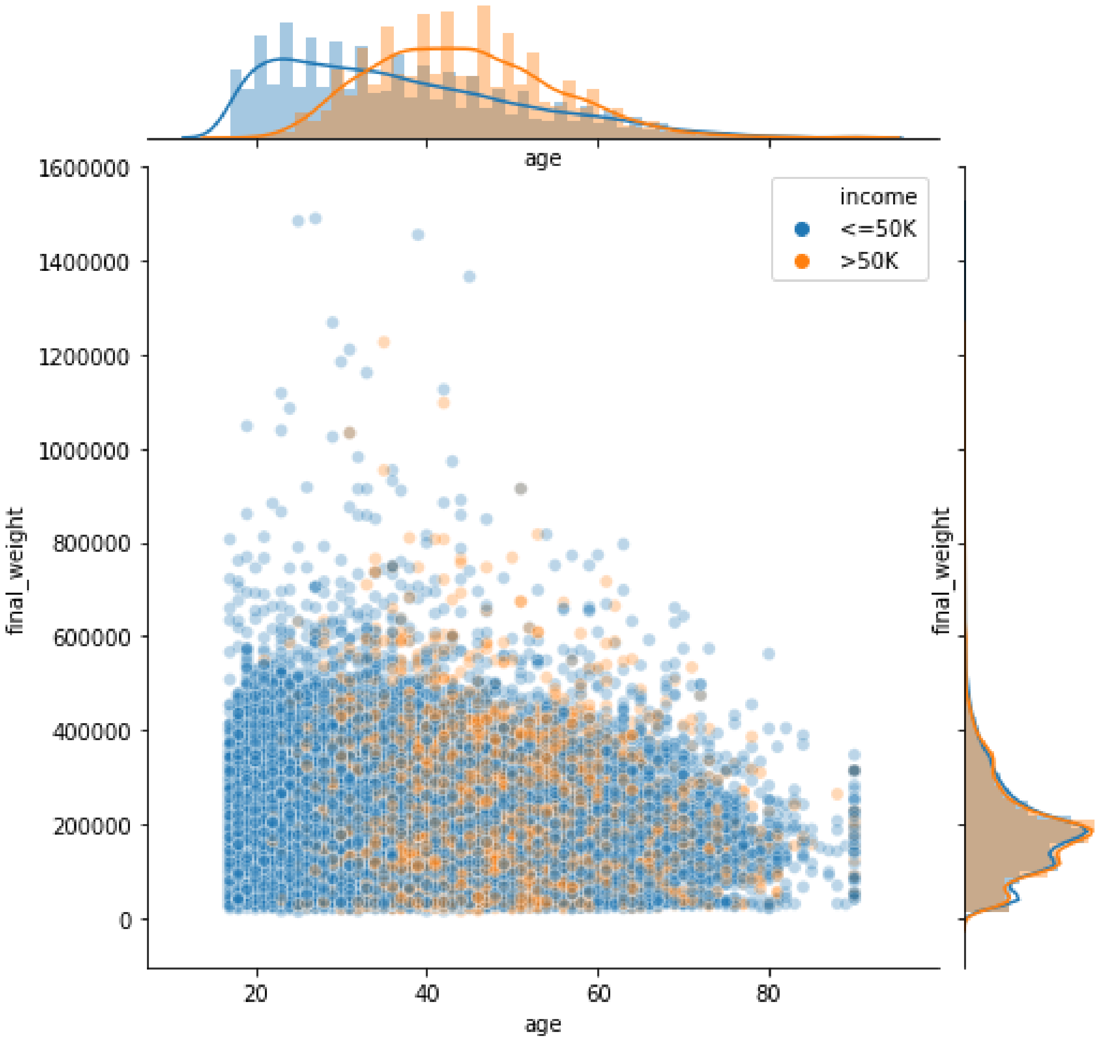
Joint plot for **final_weight** & **hours-per-week**



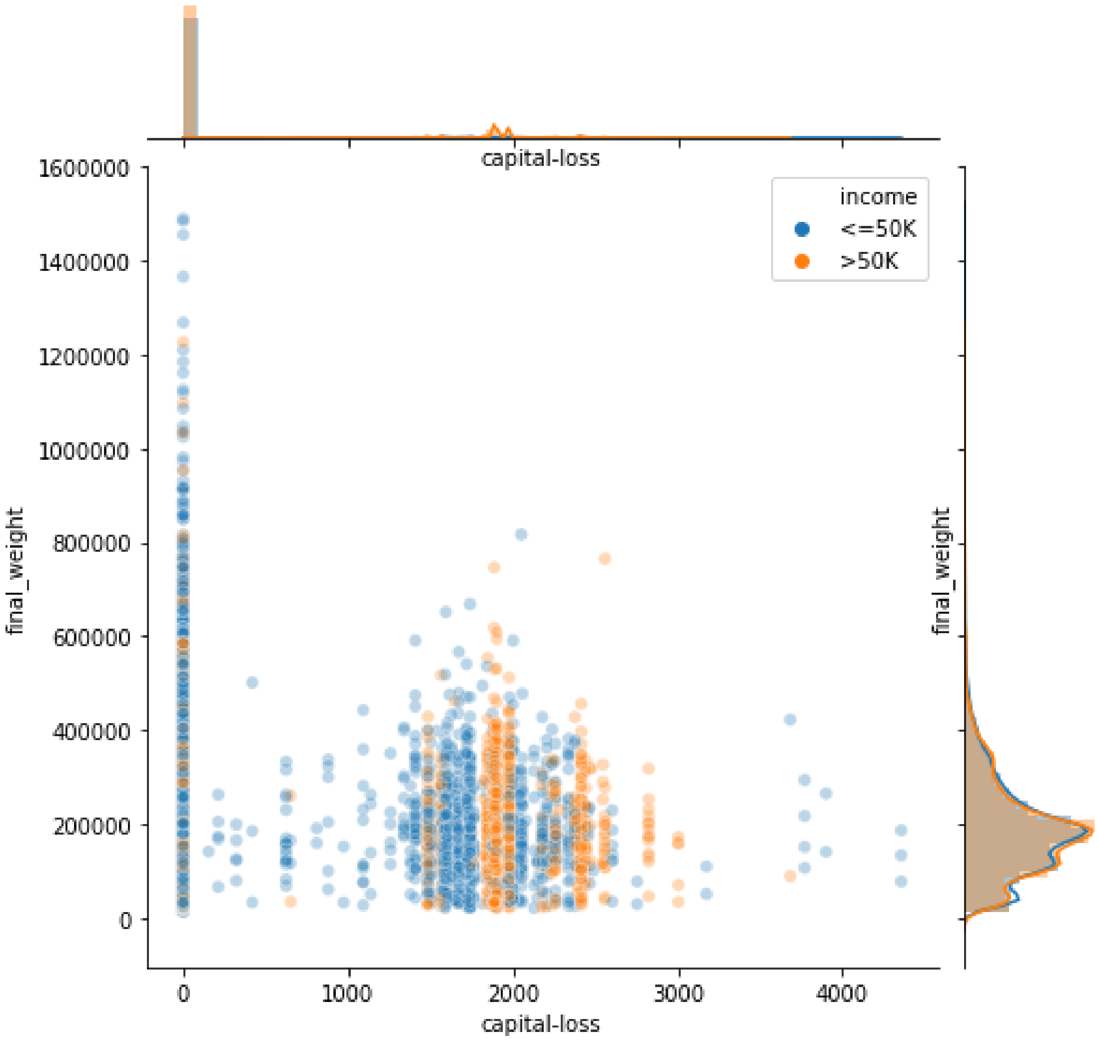
Joint plot for **capital-gain** & **hours-per-week**



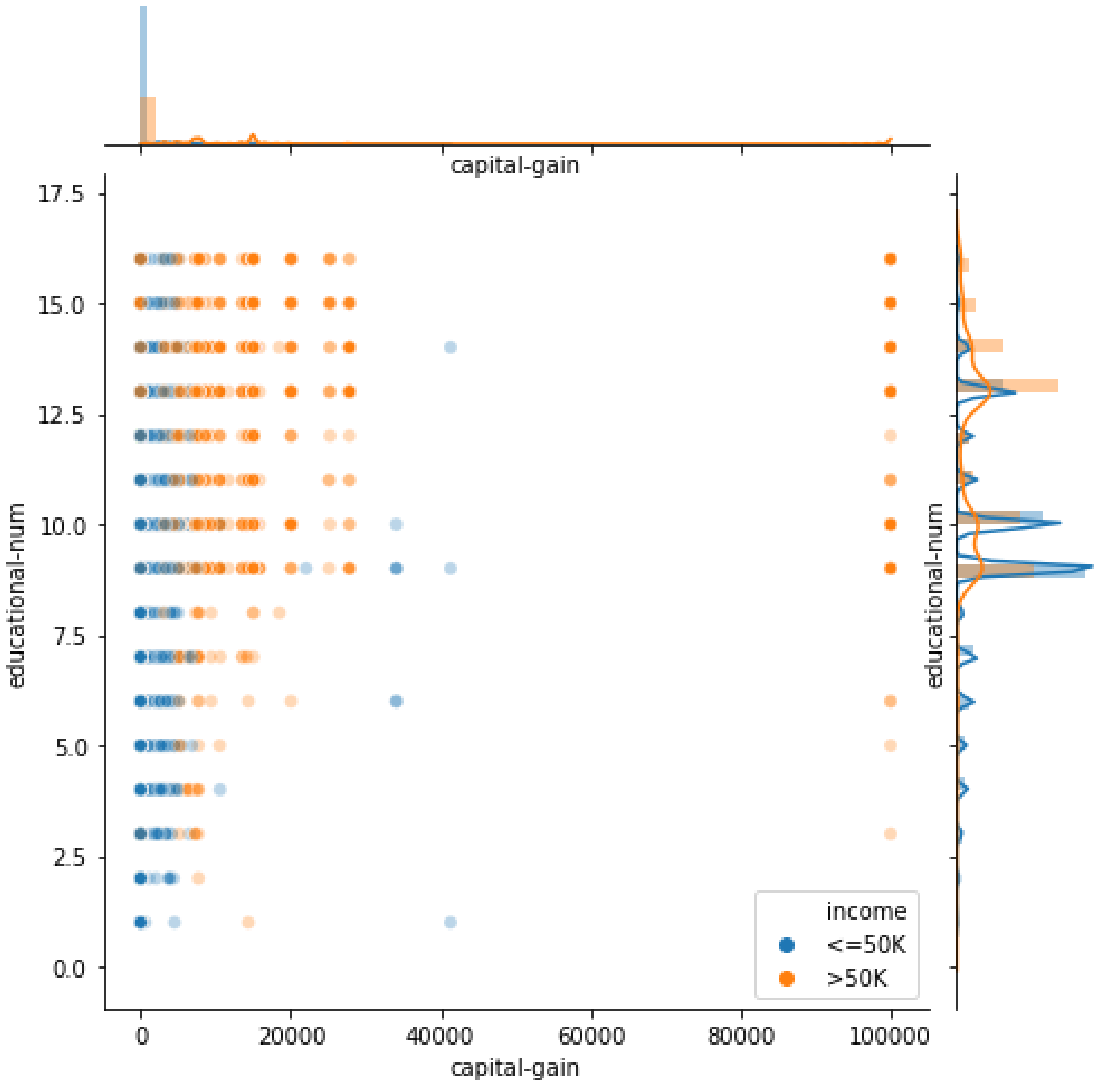
Joint plot for **age** & **final_weight**



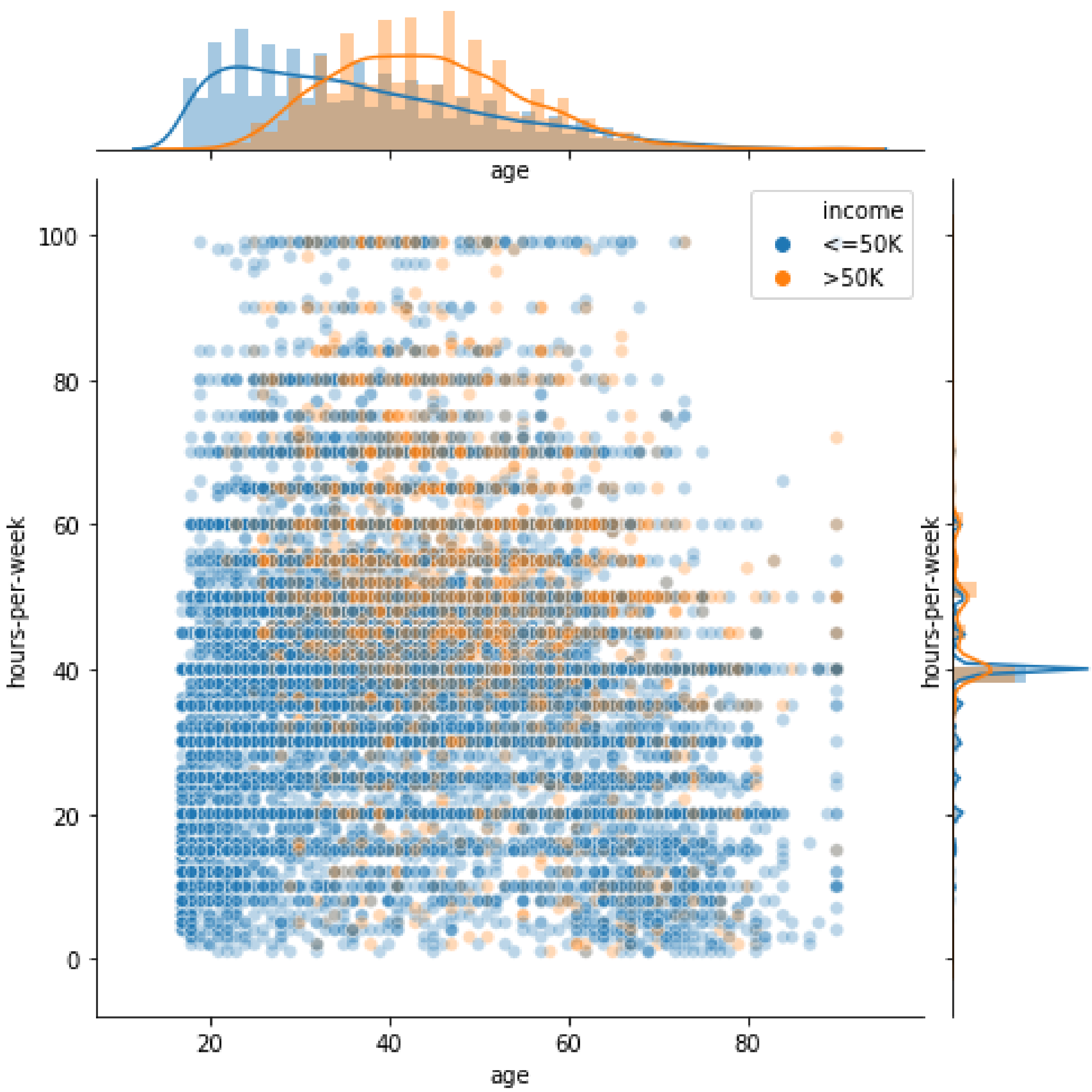
Joint plot for **capital-loss** & **final__weight**



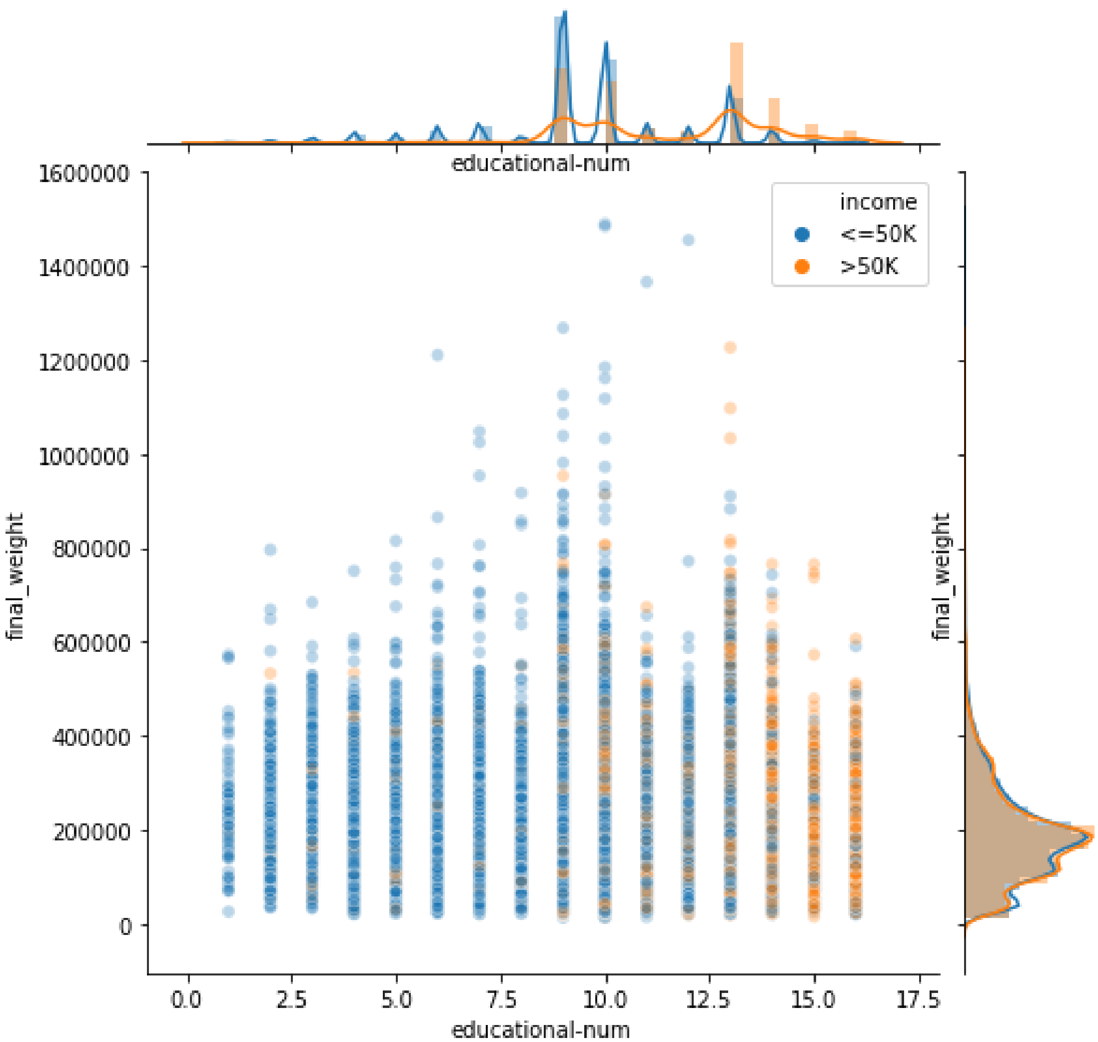
Joint plot for **capital-gain** & **educational-num**



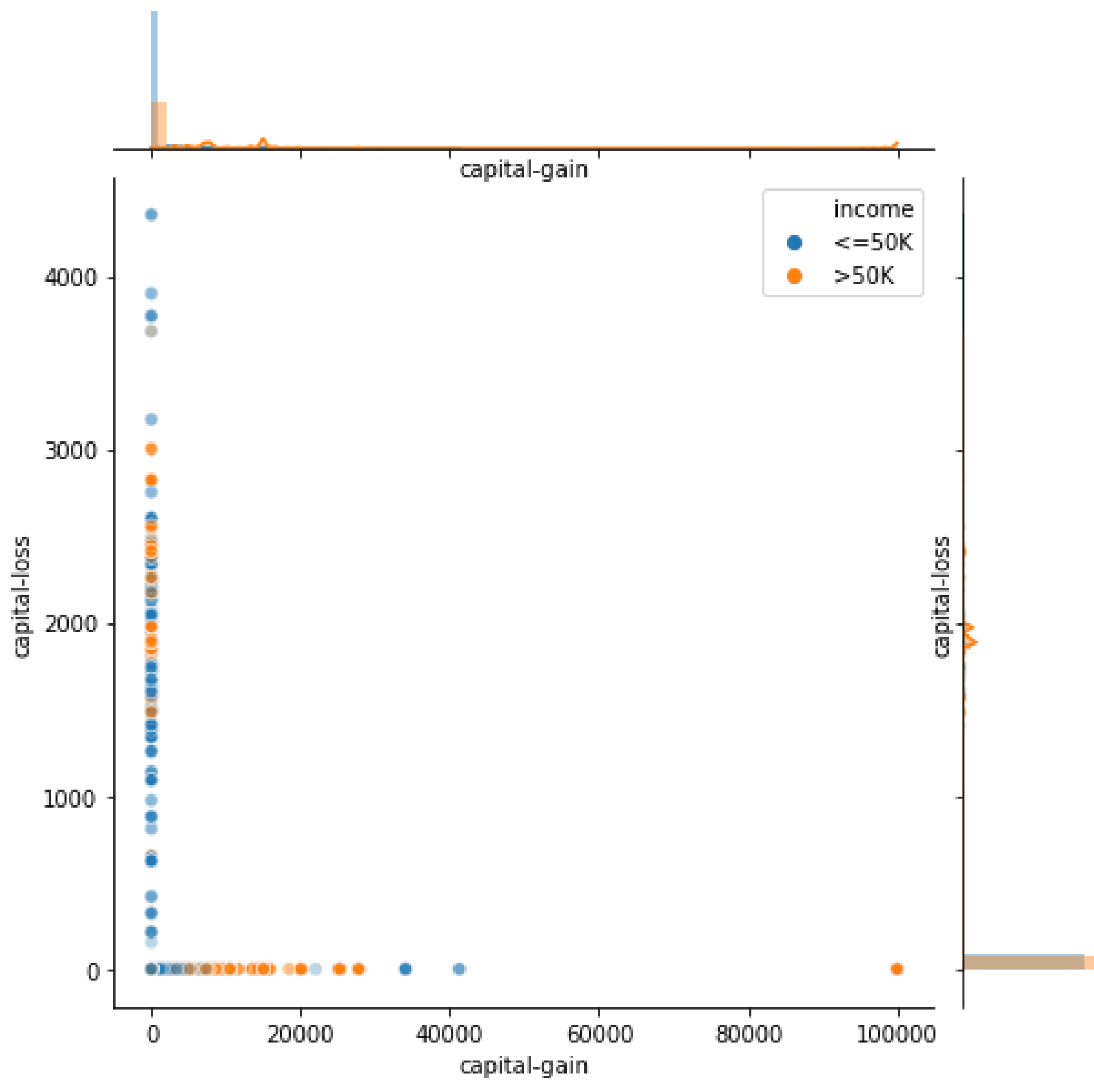
Joint plot for **age** & **hours-per-week**



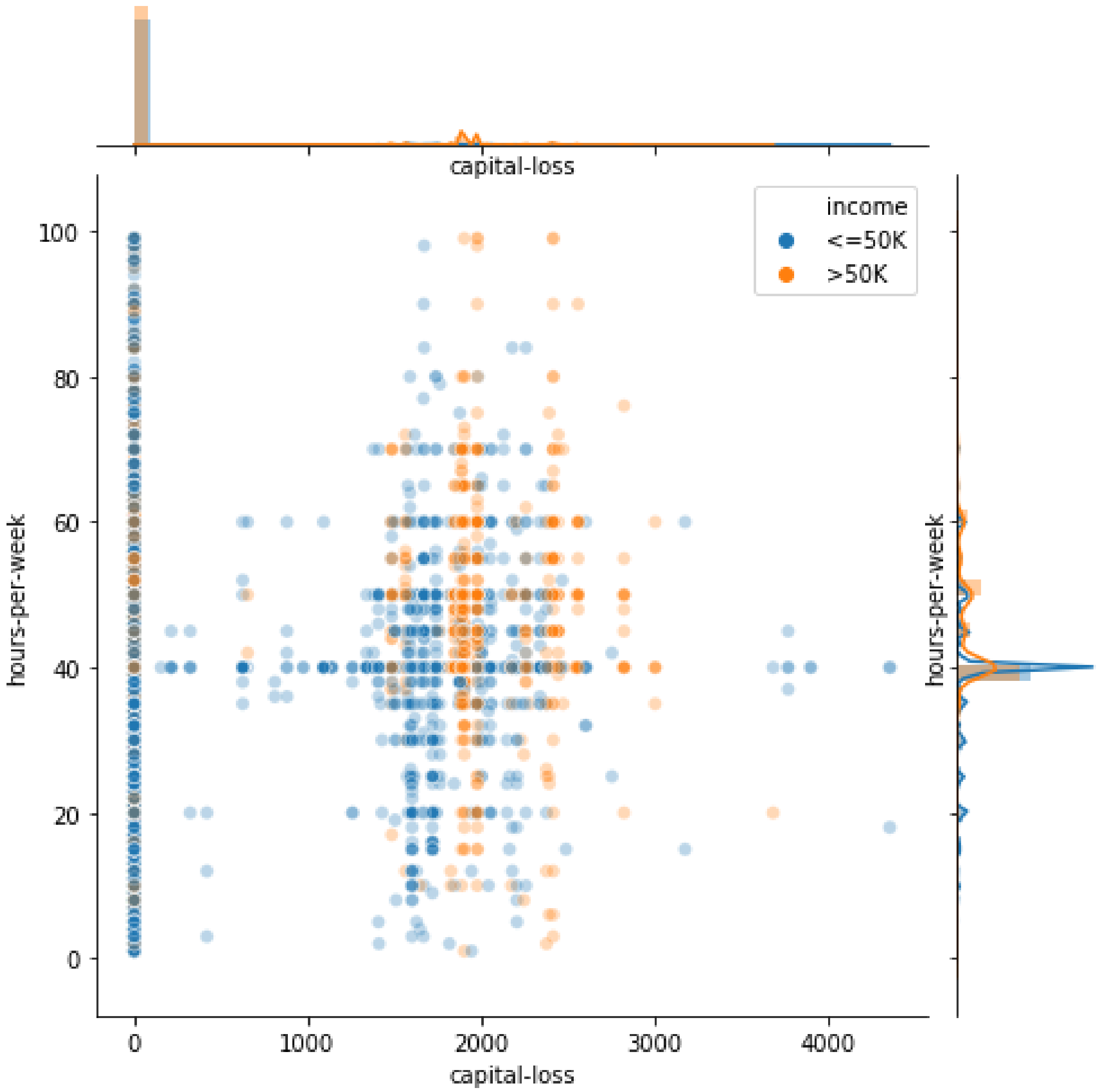
Joint plot for **educational-num** & **final_weight**



Joint plot for **capital-gain** & **capital-loss**



Joint plot for **capital-loss** & **hours-per-week**



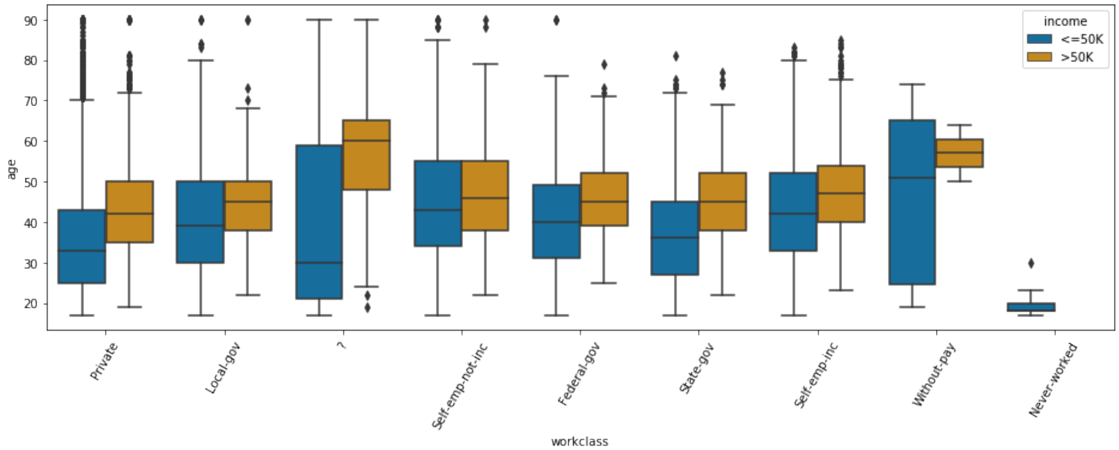
[17]:
```
help(eda.show_df_num_cat_relations)
```

```
Help on function show_df_num_cat_relations in module MLHelper.analyse.eda.eda:

show_df_num_cat_relations(df, target=None)
    Show boxplots for each pair of categorical and numerical variables
    If target is set, separate dataset for each target value.

    Parameters
    ----------
    df: pd.DataFrame
        Dataframe to inspect
    target: str (optional)
```

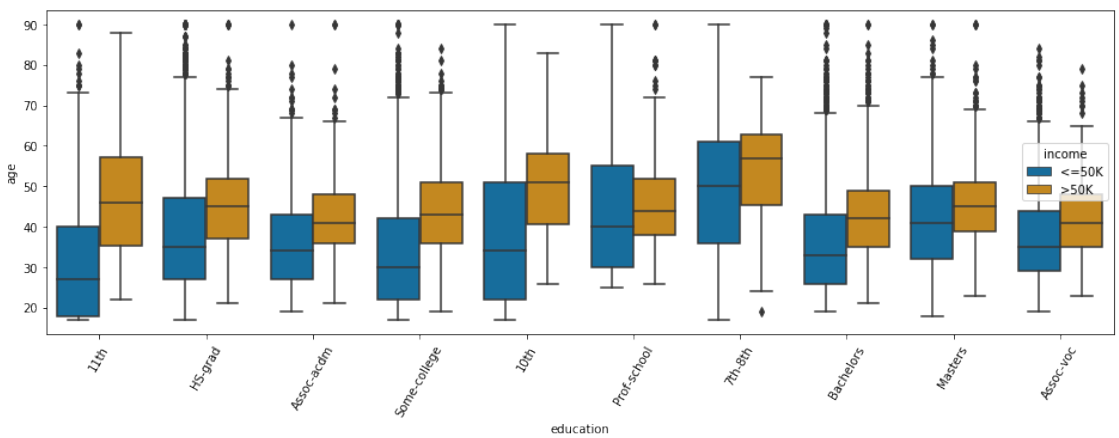
Target column for classifier

[18]:
```
eda.show_df_num_cat_relations(df=adult, target='income')
```

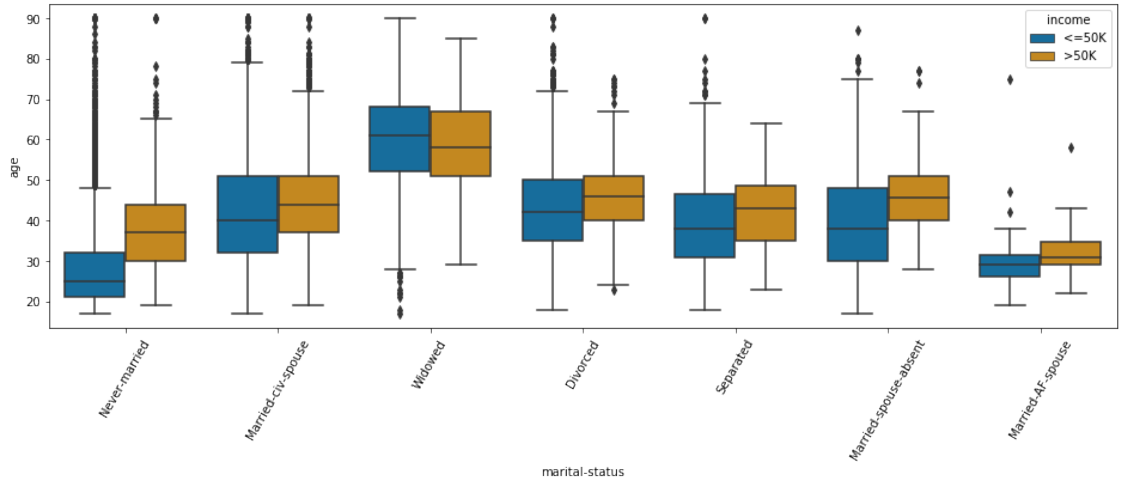
Box plot for **workclass** & **age**



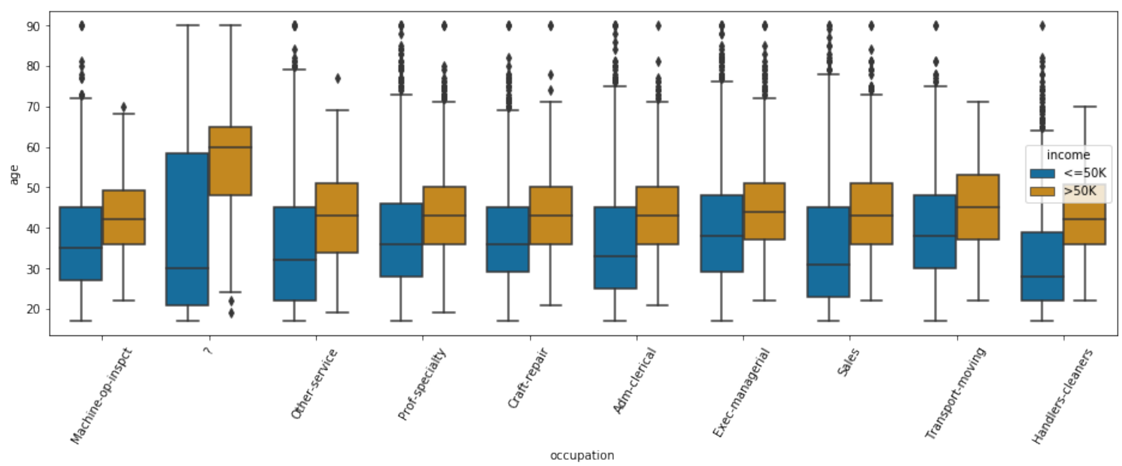
Box plot for **education** & **age**



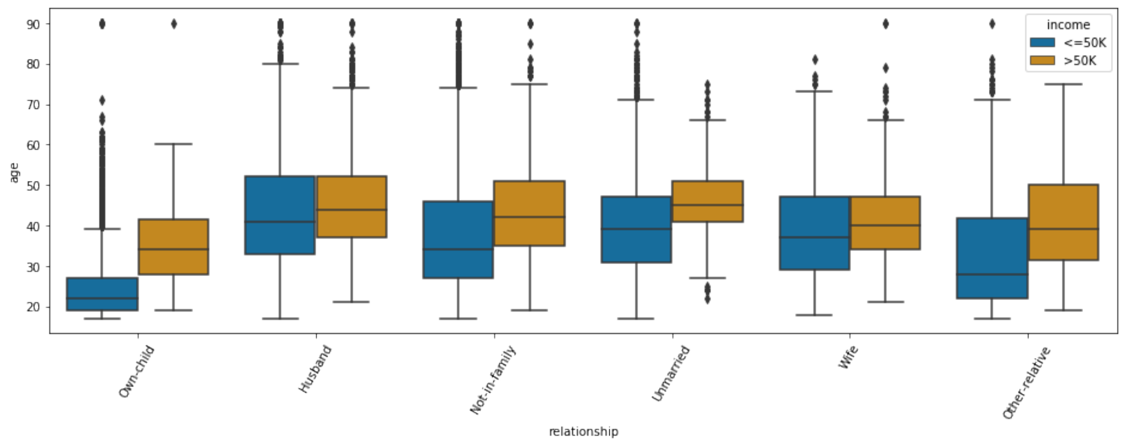
Box plot for **marital-status** & **age**



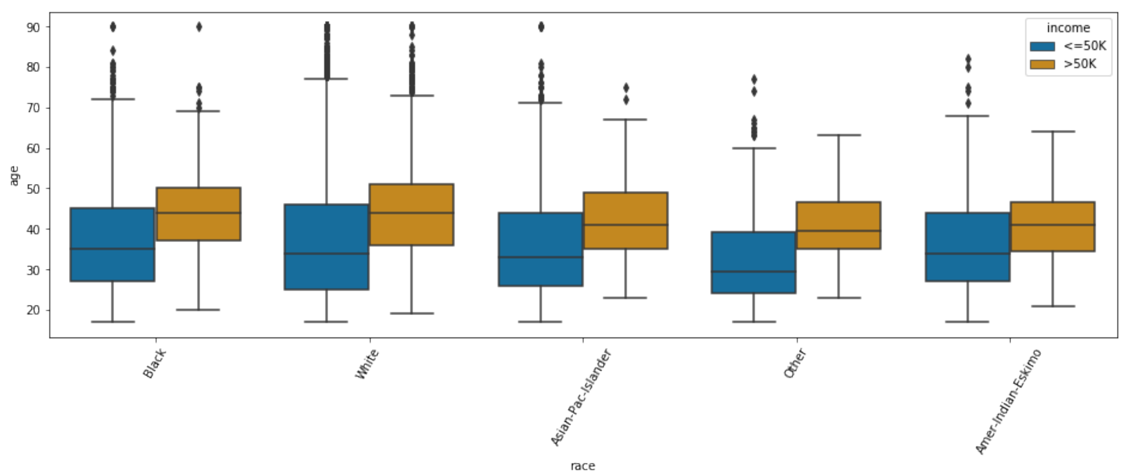
Box plot for **occupation** & **age**



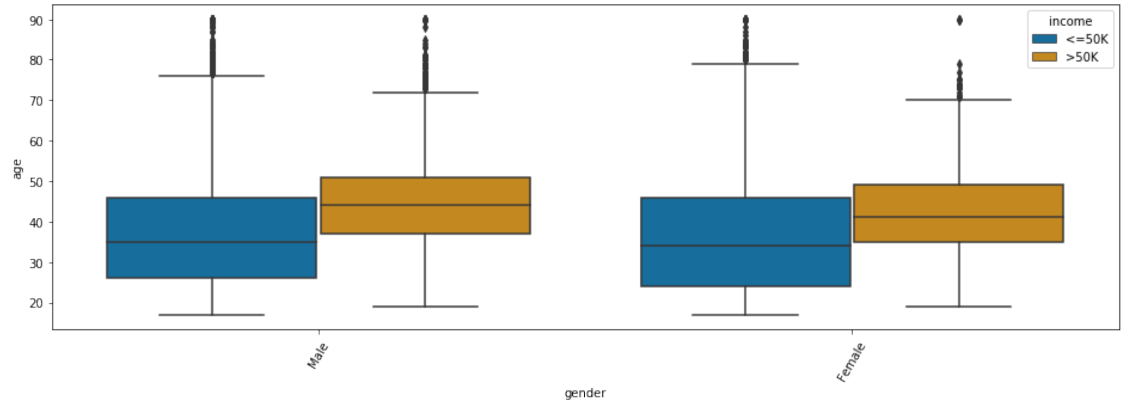
Box plot for **relationship** & **age**



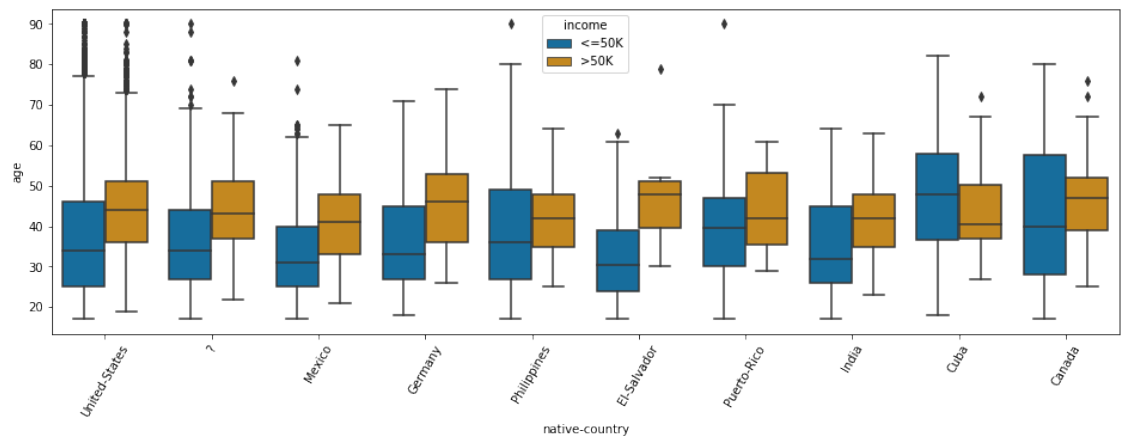
Box plot for **race** & **age**



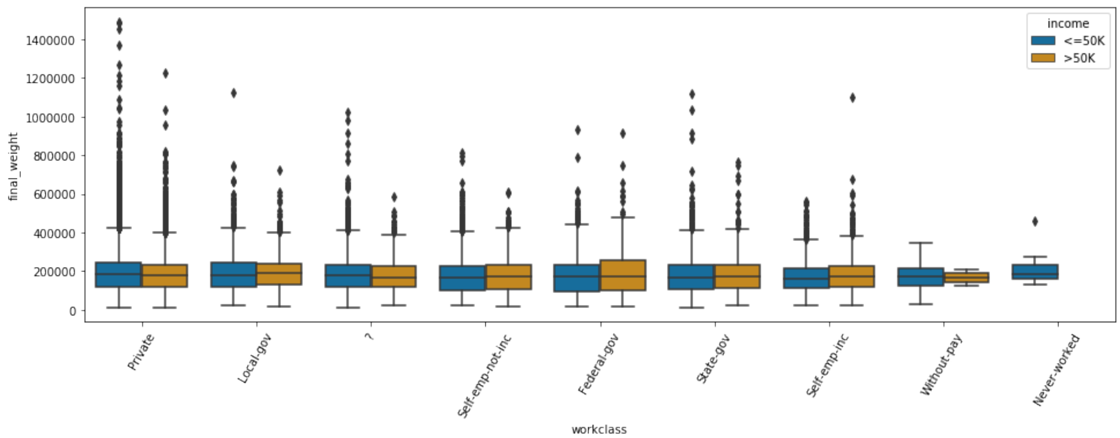
Box plot for **gender** & **age**



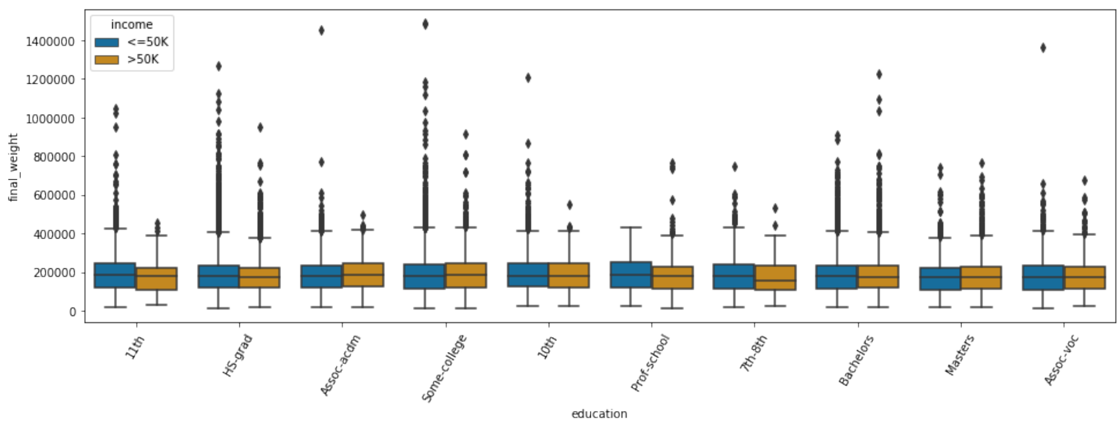
Box plot for **native-country** & **age**



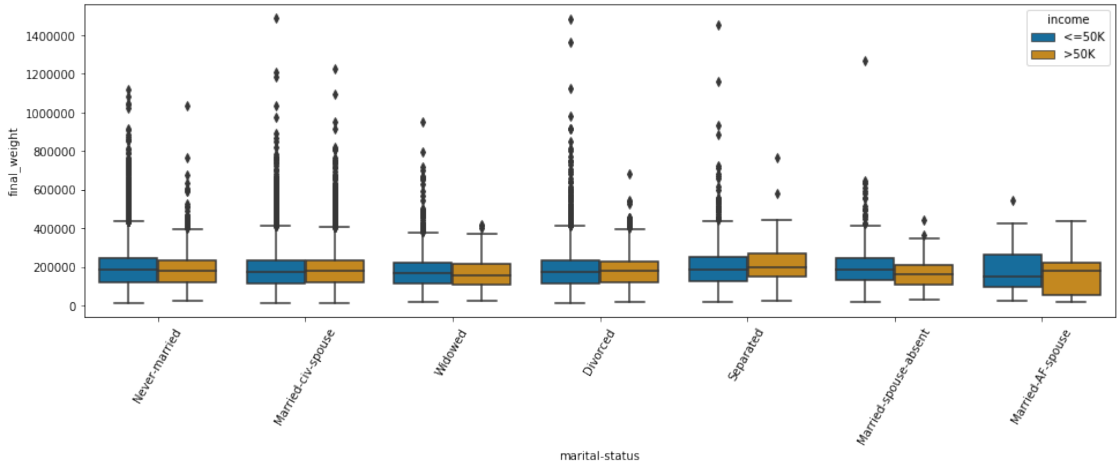
Box plot for **workclass** & **final_weight**



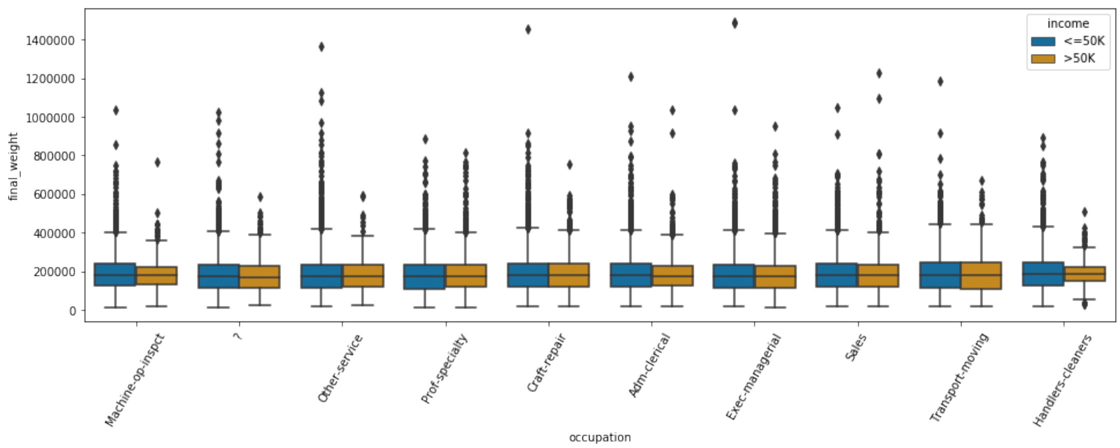
Box plot for **education** & **final_weight**



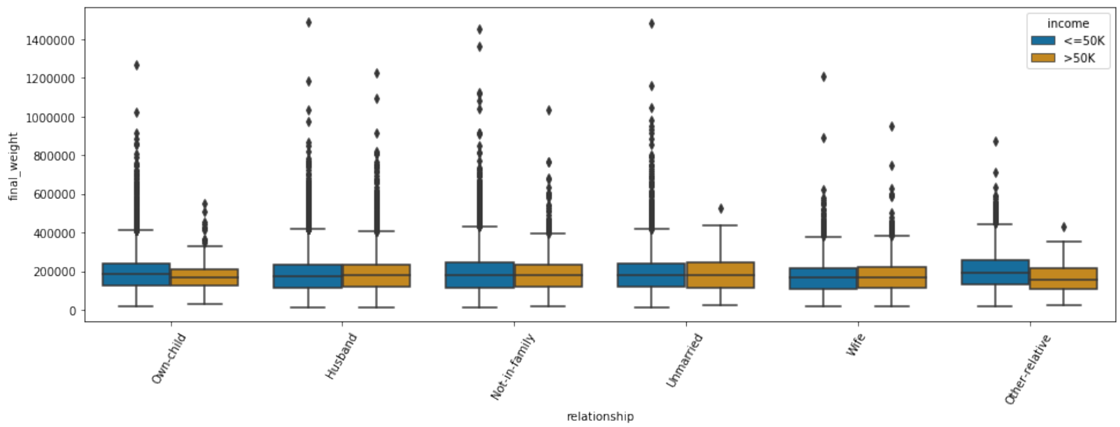
Box plot for **marital-status** & **final_weight**



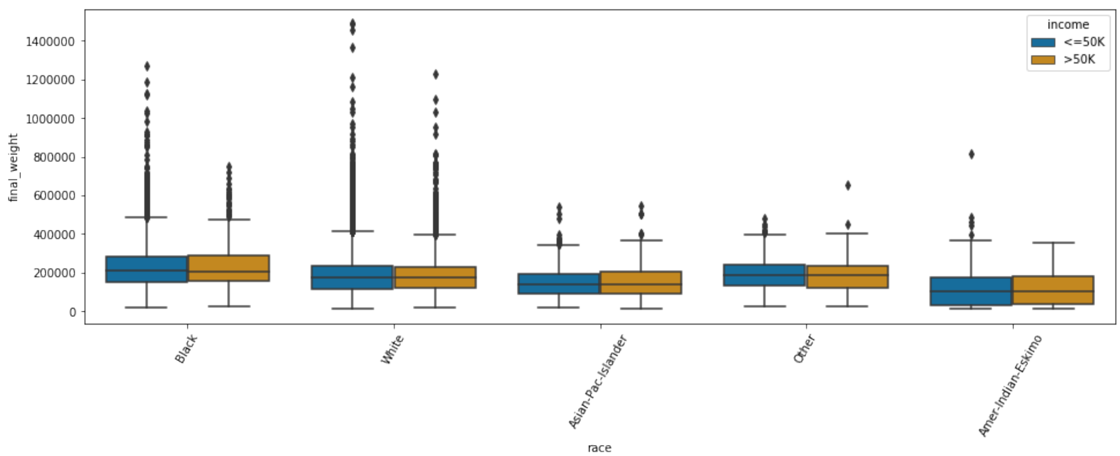
Box plot for **occupation** & **final_weight**



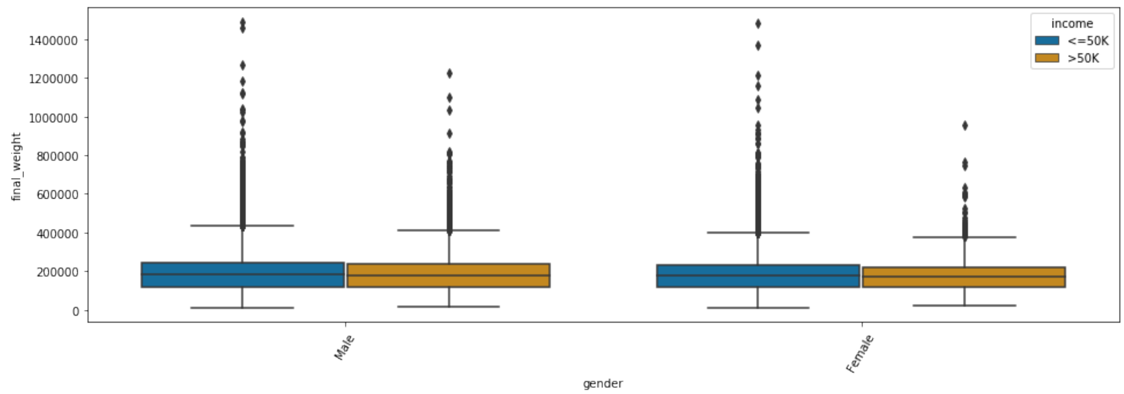
Box plot for **relationship** & **final_weight**



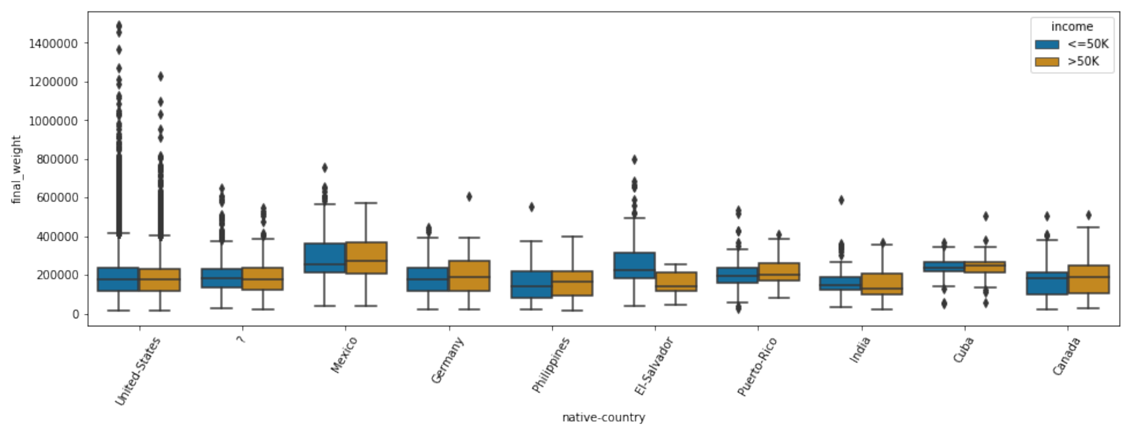
Box plot for **race** & **final_weight**



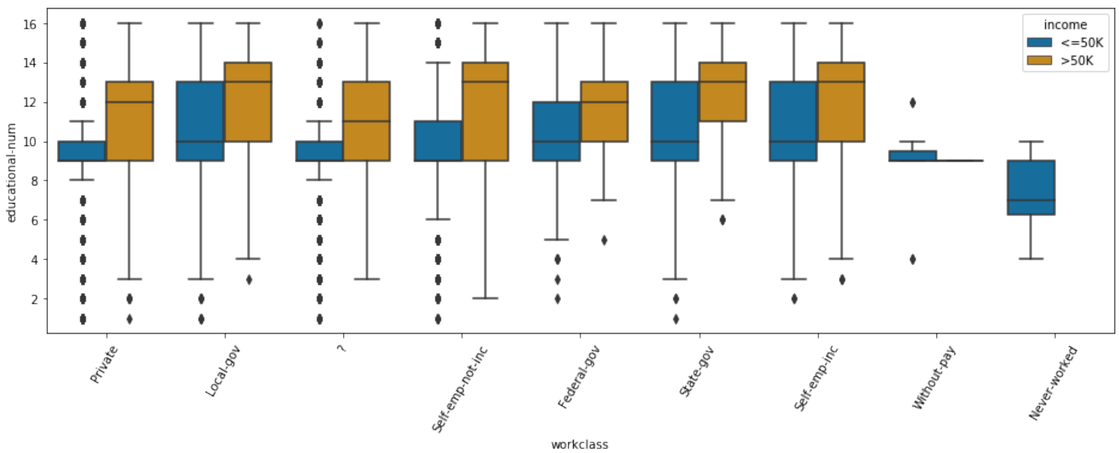
Box plot for **gender** & **final_weight**



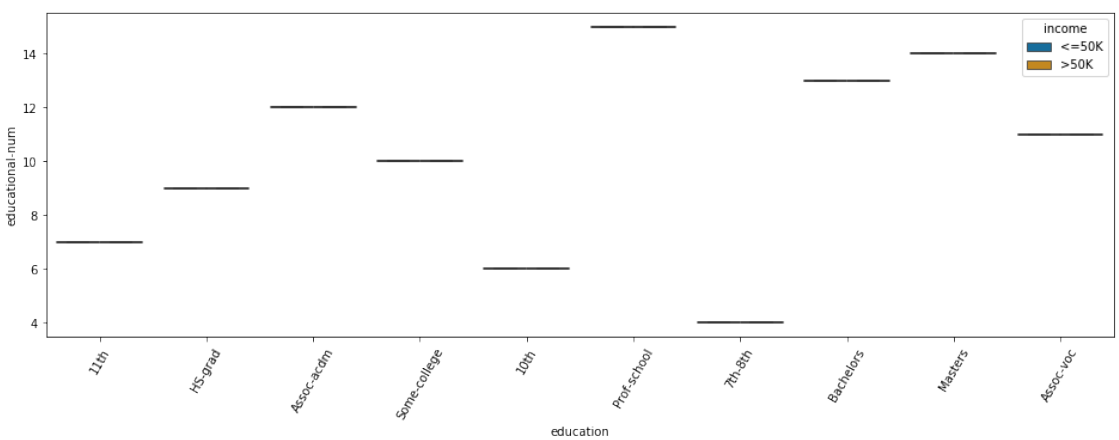
Box plot for **native-country** & **final_weight**



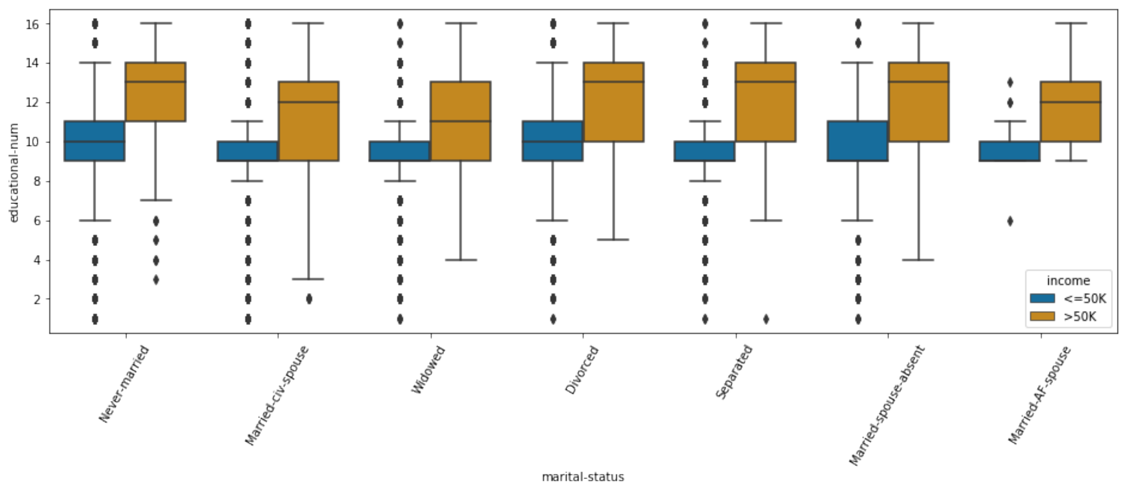
Box plot for **workclass** & **educational-num**



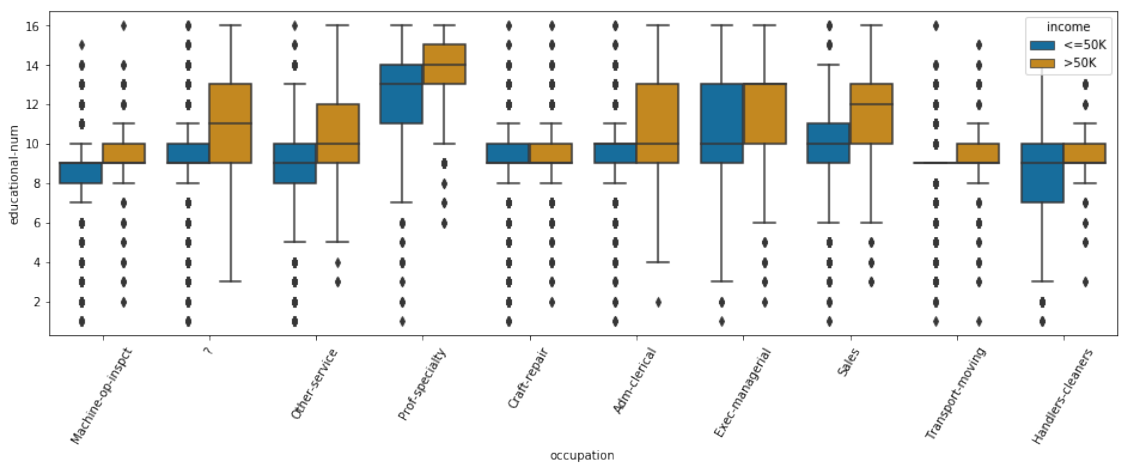
Box plot for **education** & **educational-num**



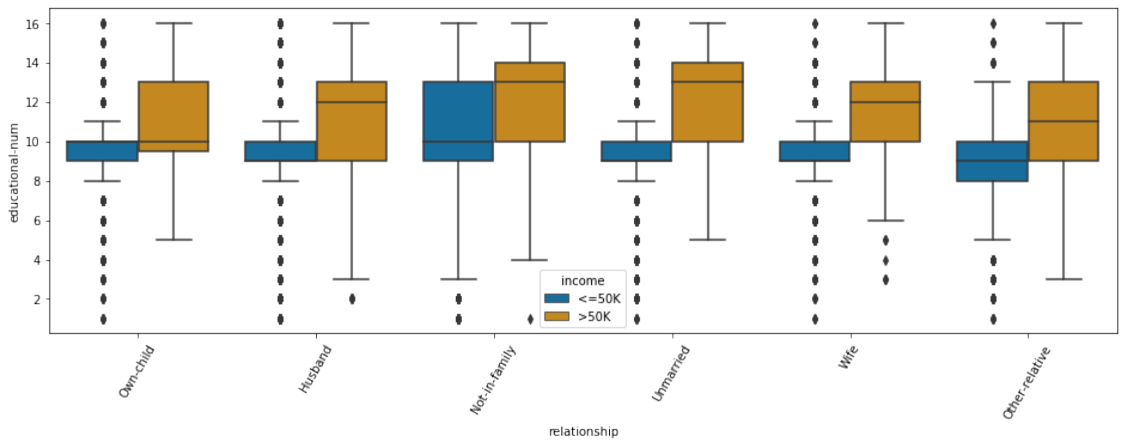
Box plot for **marital-status** & **educational-num**



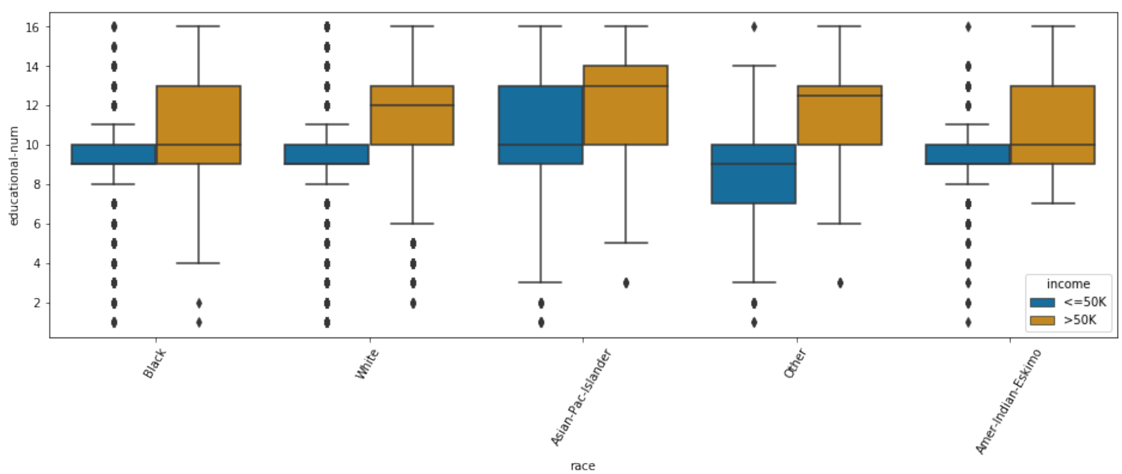
Box plot for **occupation** & **educational-num**



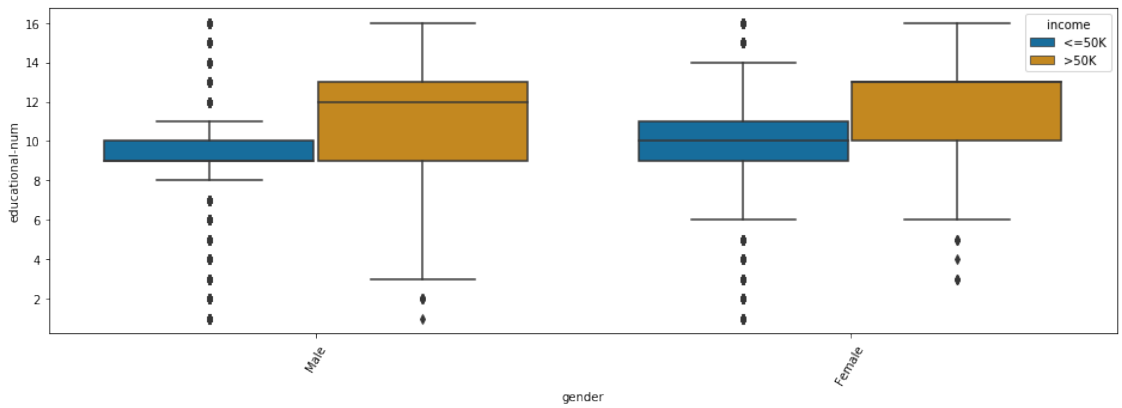
Box plot for **relationship** & **educational-num**



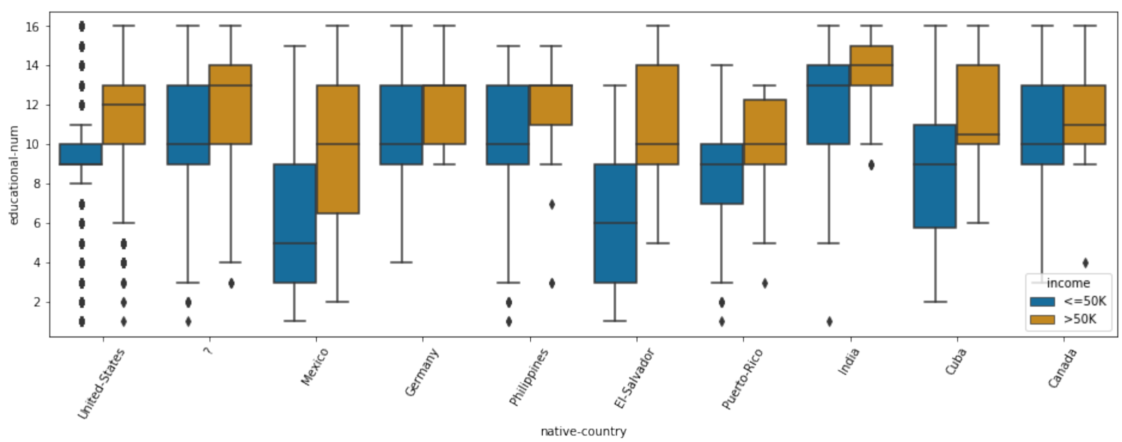
Box plot for **race** & **educational-num**



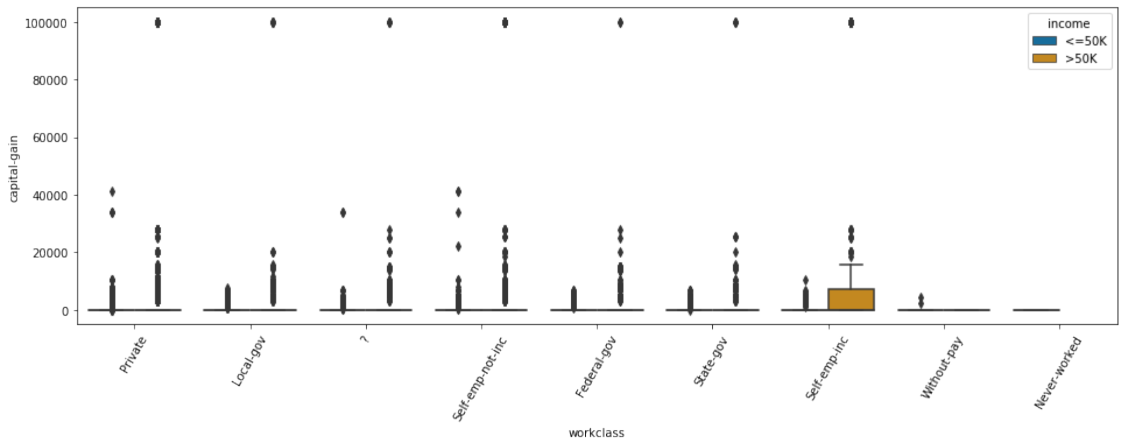
Box plot for **gender** & **educational-num**



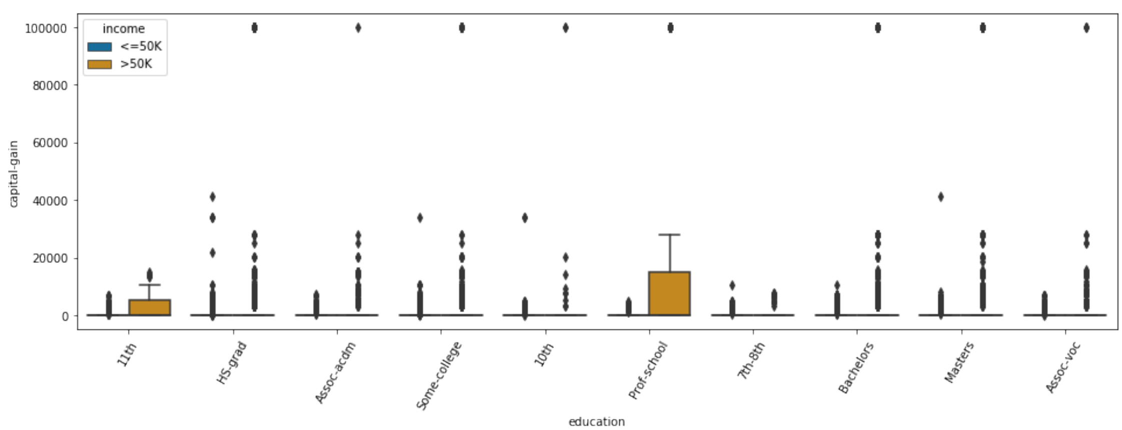
Box plot for **native-country** & **educational-num**



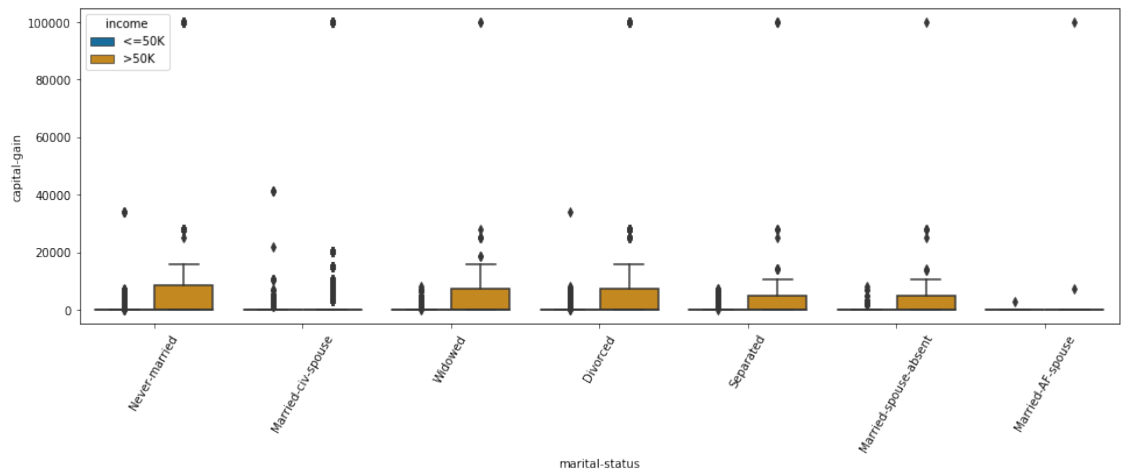
Box plot for **workclass** & **capital-gain**



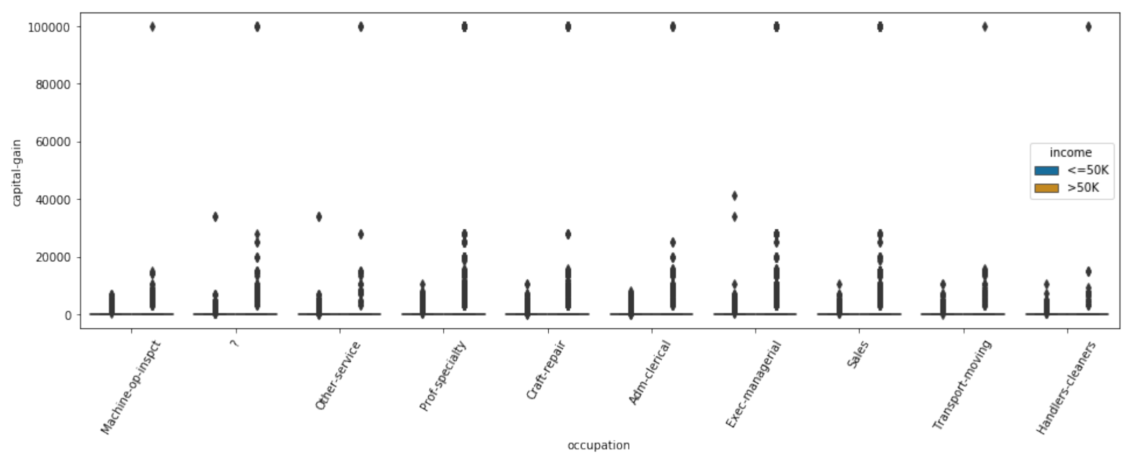
Box plot for **education** & **capital-gain**



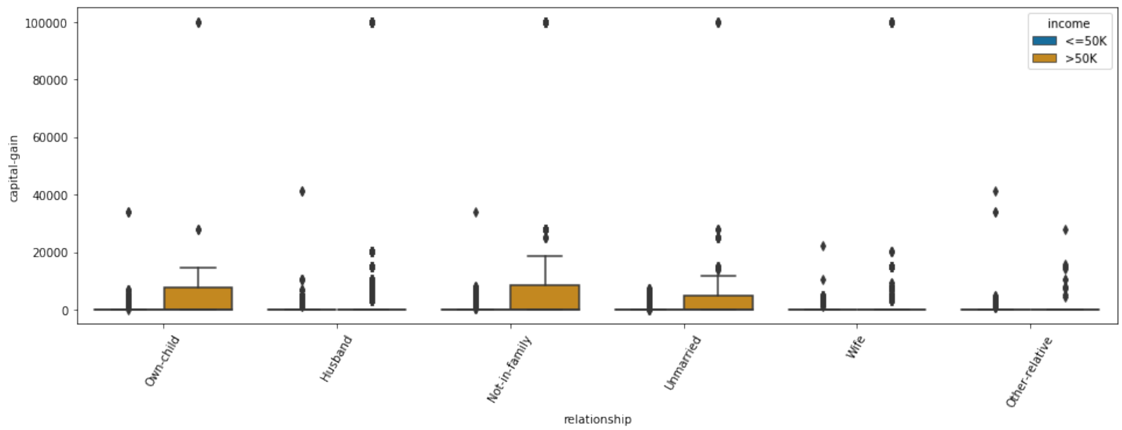
Box plot for **marital-status** & **capital-gain**



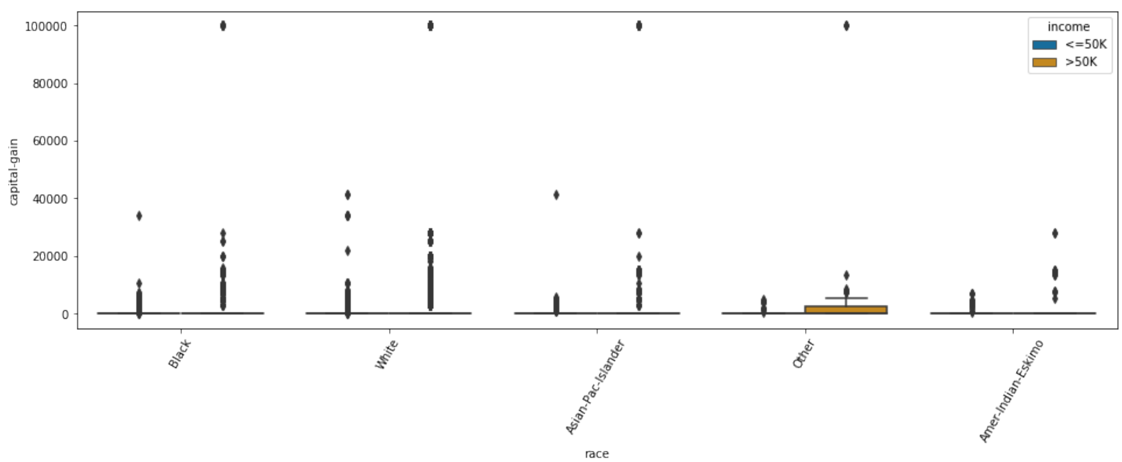
Box plot for **occupation** & **capital-gain**



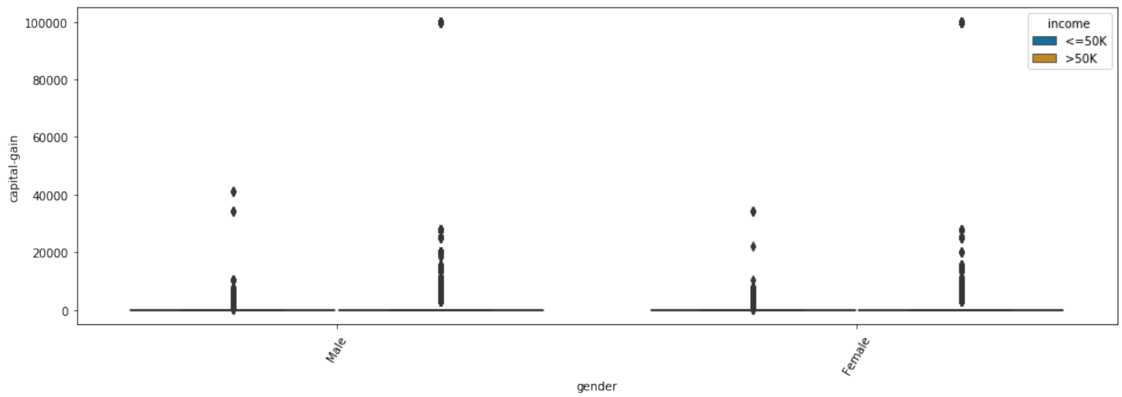
Box plot for **relationship** & **capital-gain**



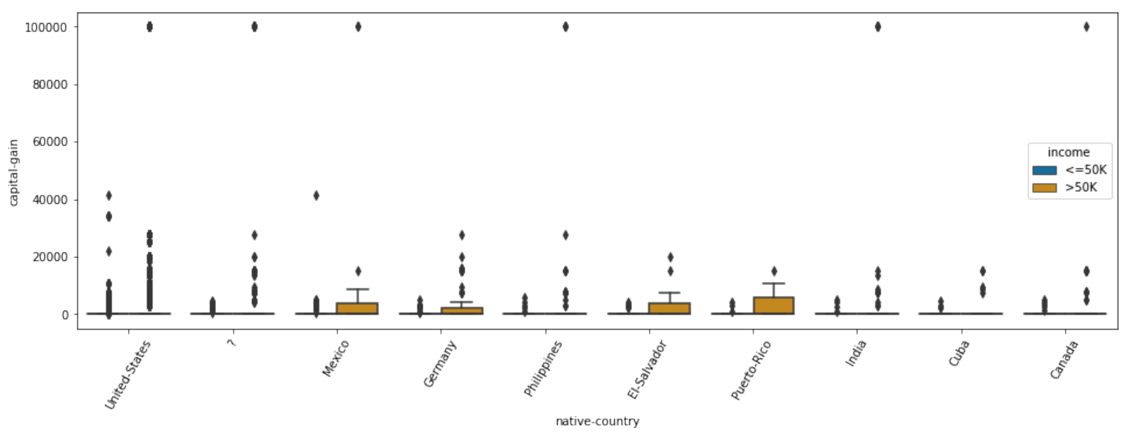
Box plot for **race** & **capital-gain**



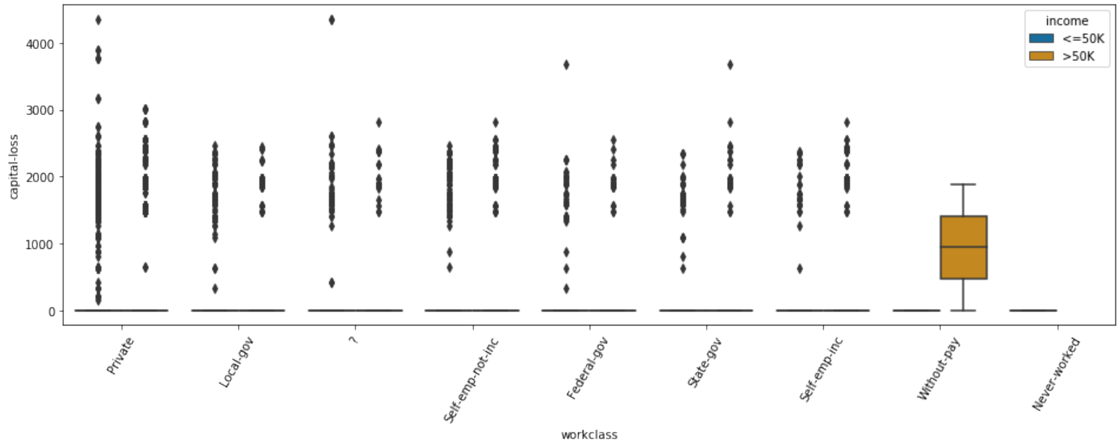
Box plot for **gender** & **capital-gain**



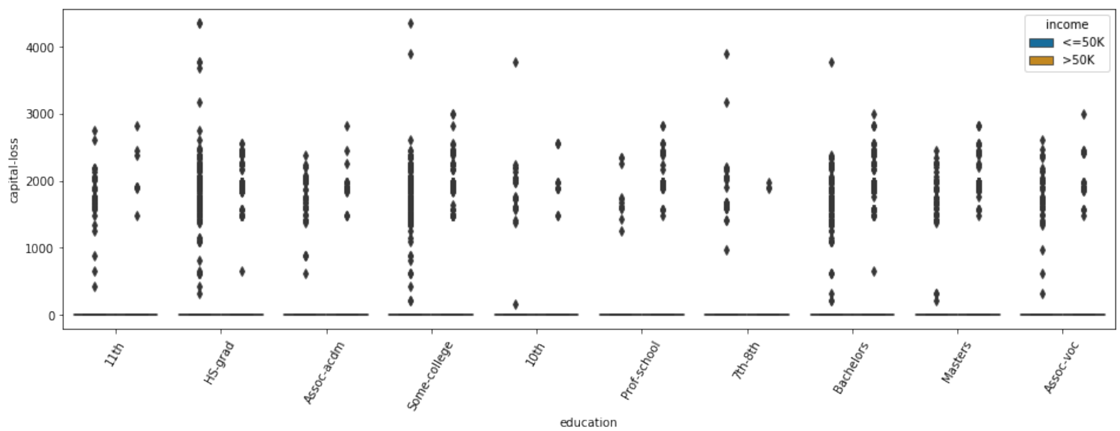
Box plot for **native-country** & **capital-gain**



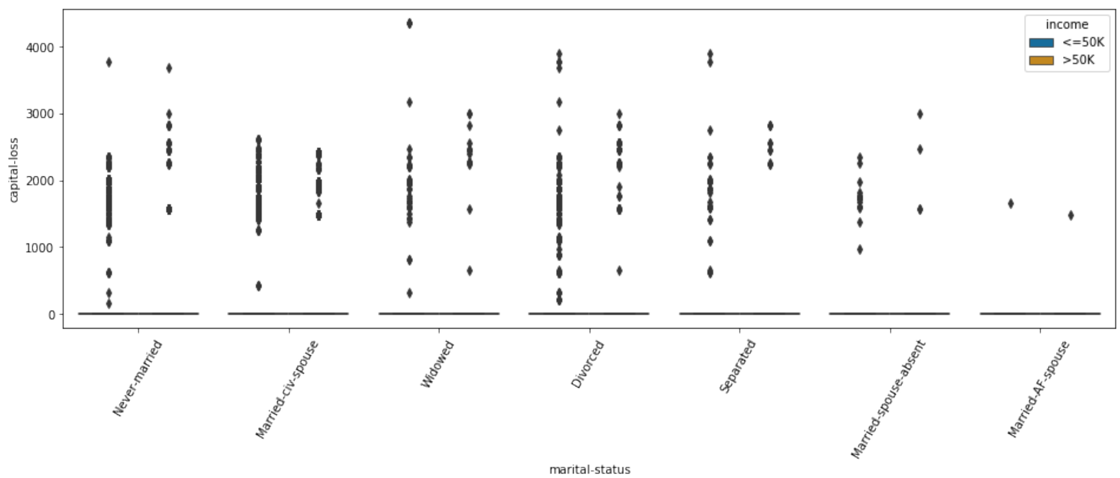
Box plot for **workclass** & **capital-loss**



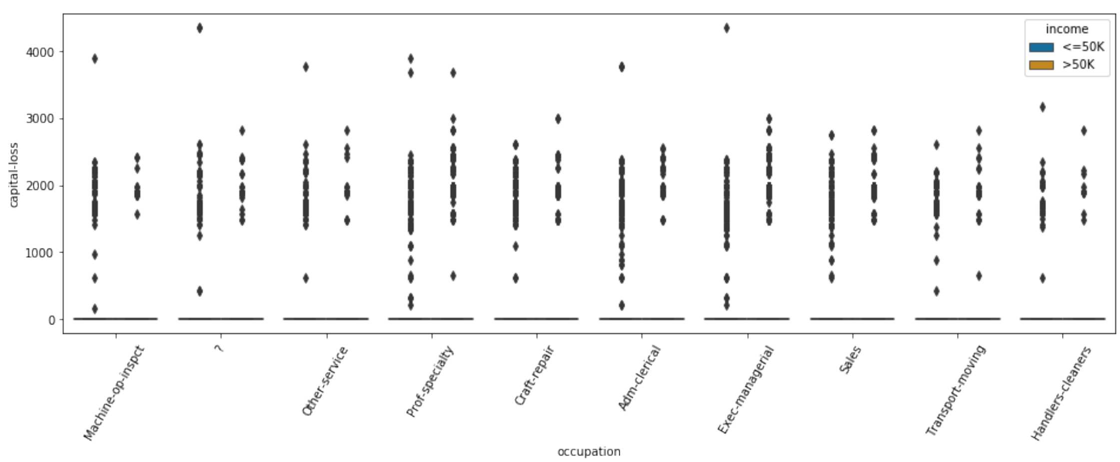
Box plot for **education** & **capital-loss**



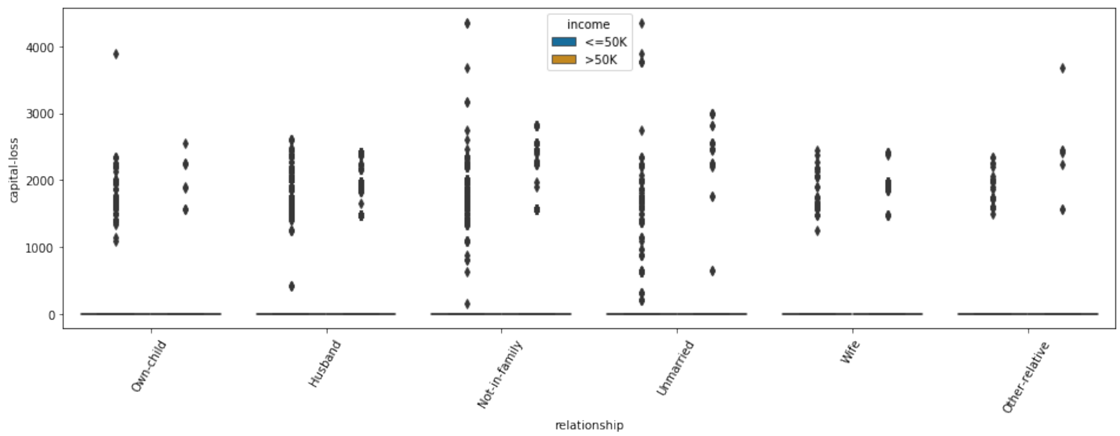
Box plot for **marital-status** & **capital-loss**



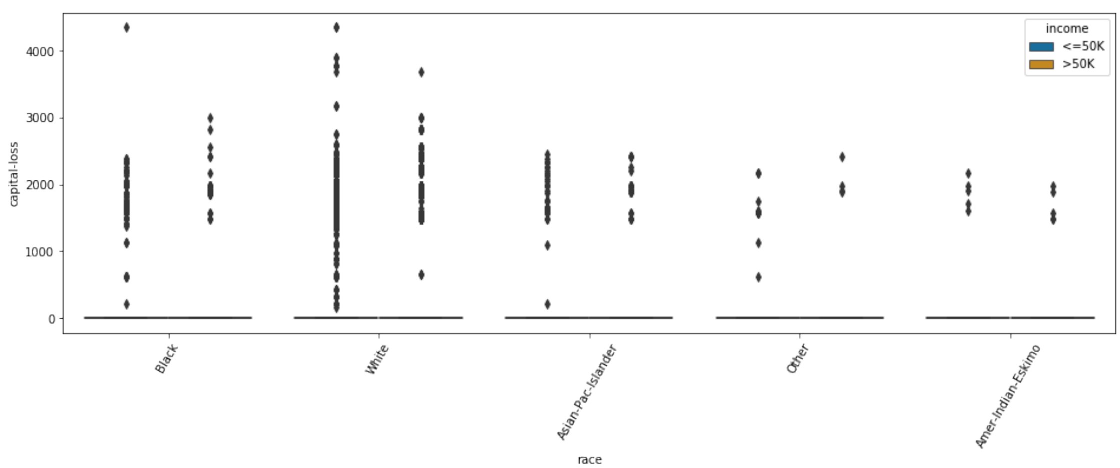
Box plot for **occupation** & **capital-loss**



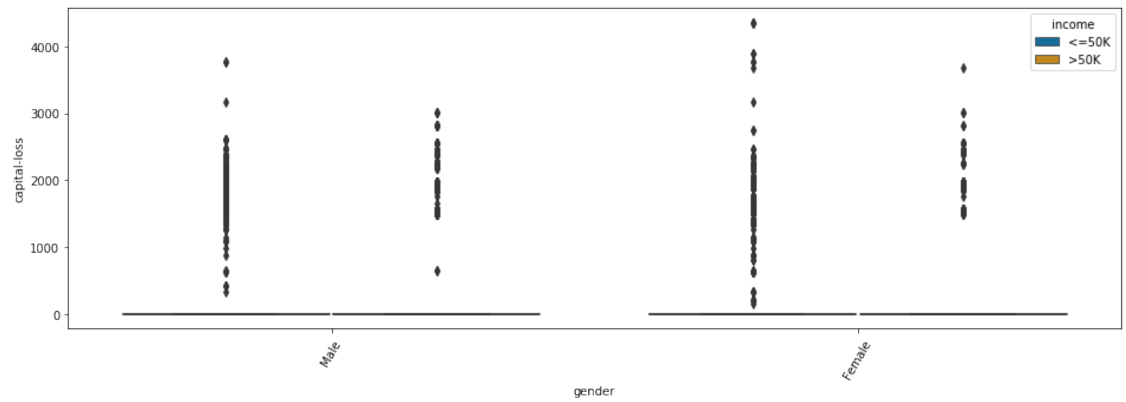
Box plot for **relationship** & **capital-loss**



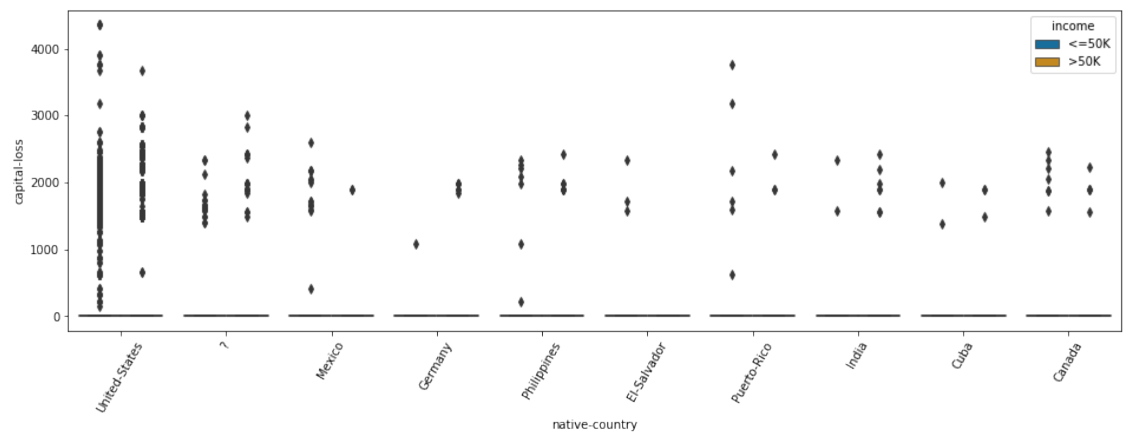
Box plot for **race** & **capital-loss**



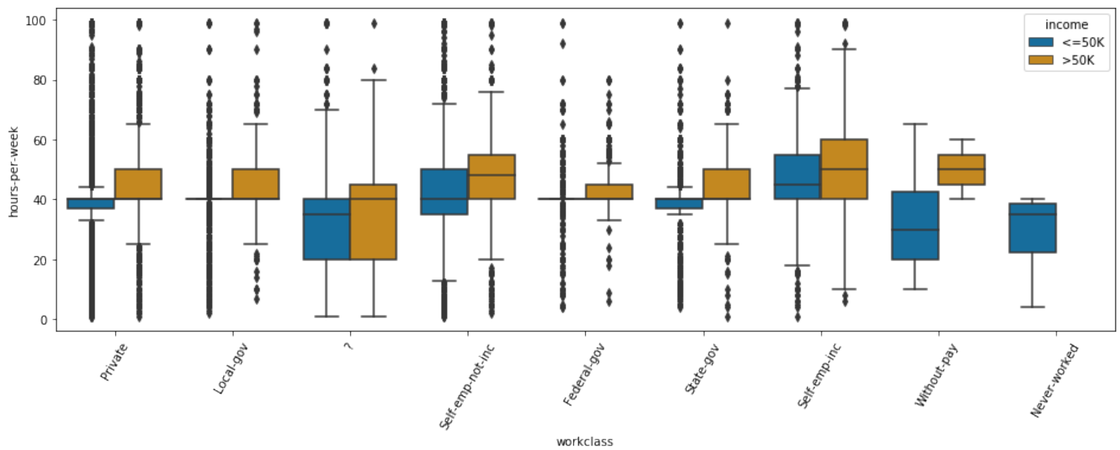
Box plot for **gender** & **capital-loss**



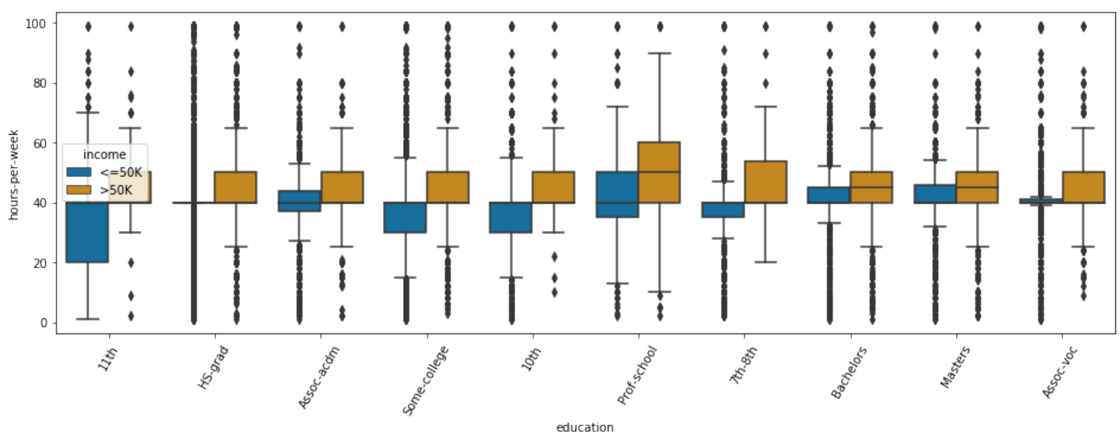
Box plot for **native-country** & **capital-loss**



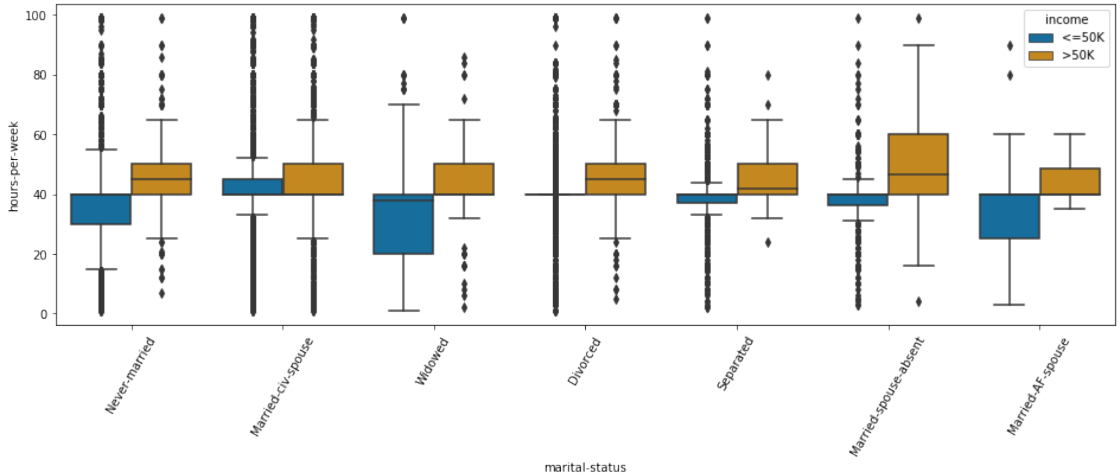
Box plot for **workclass** & **hours-per-week**



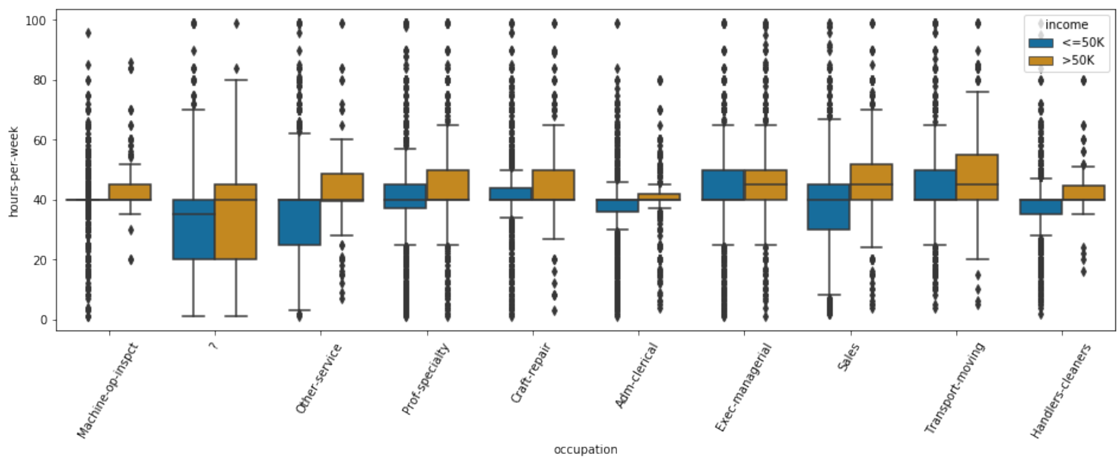
Box plot for **education** & **hours-per-week**



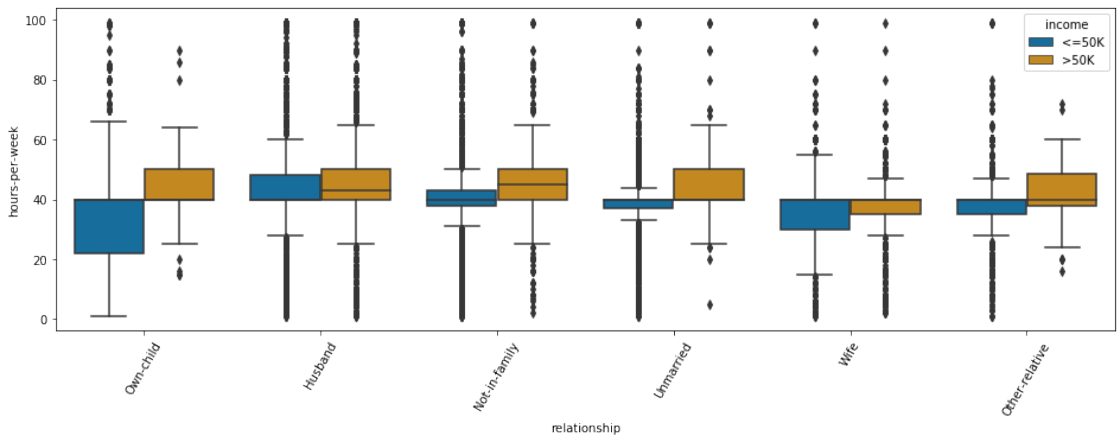
Box plot for **marital-status** & **hours-per-week**



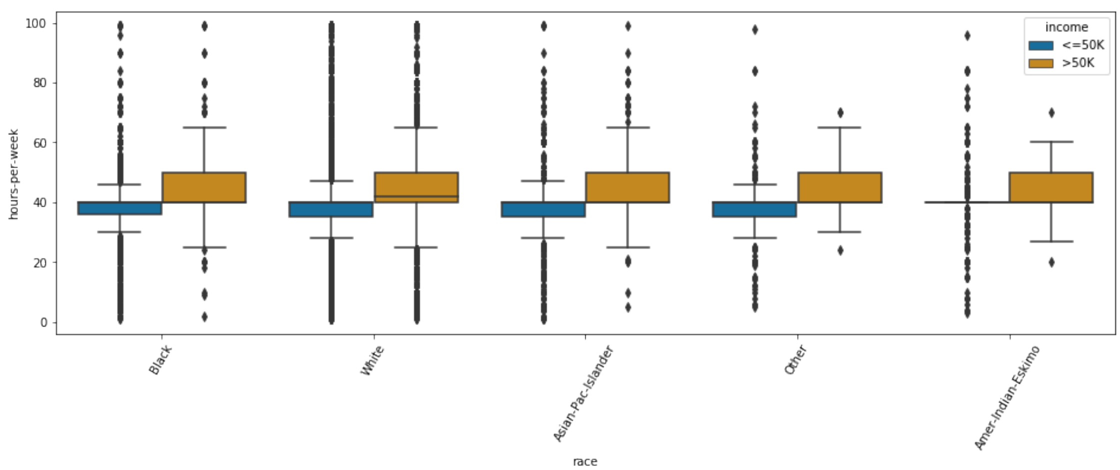
Box plot for **occupation** & **hours-per-week**



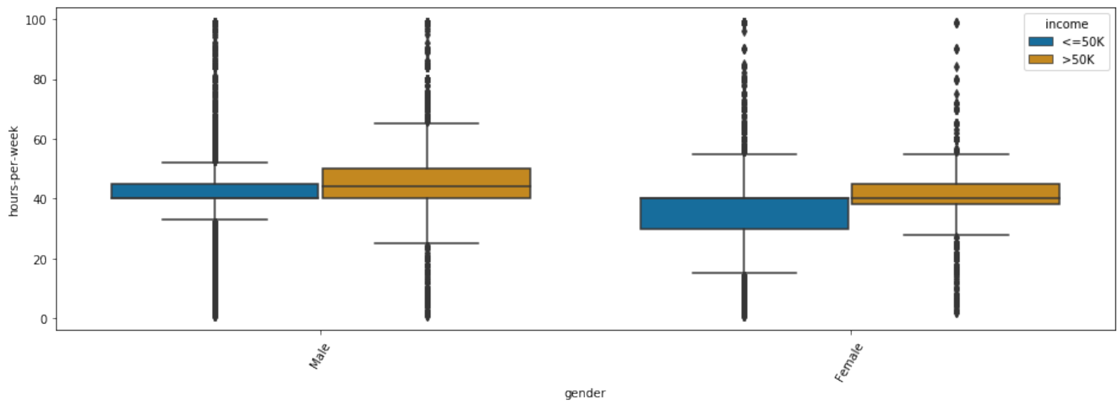
Box plot for **relationship** & **hours-per-week**



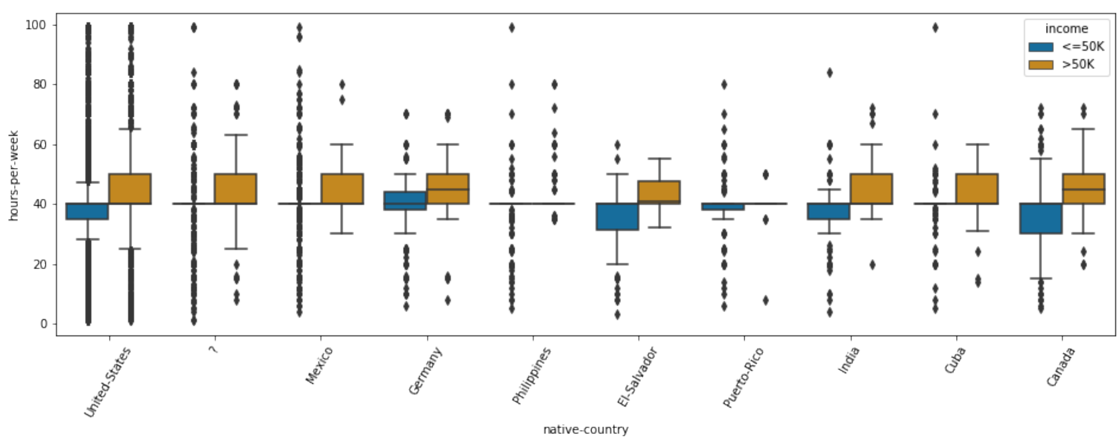
Box plot for **race** & **hours-per-week**



Box plot for **gender** & **hours-per-week**



Box plot for **native-country** & **hours-per-week**

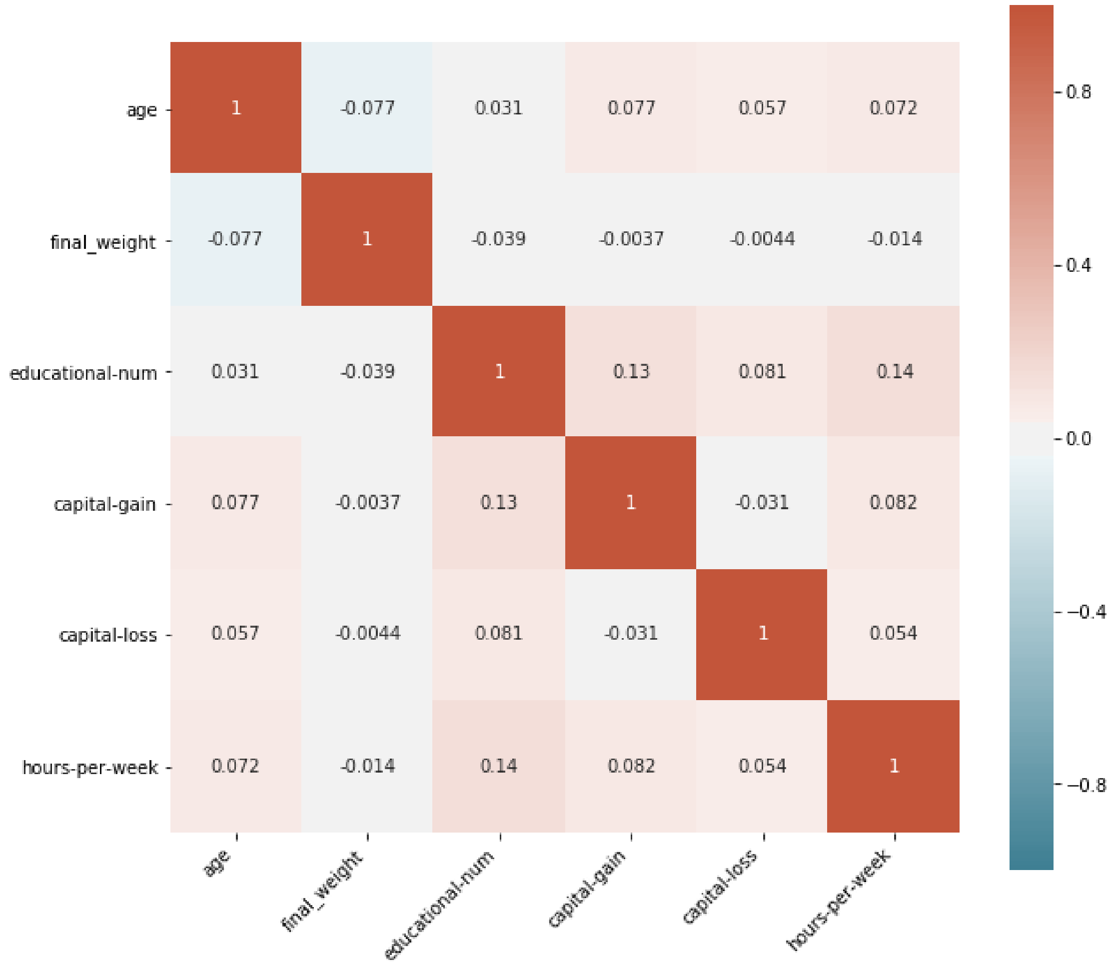


[19]:
```
help(eda.show_df_correlations)
```

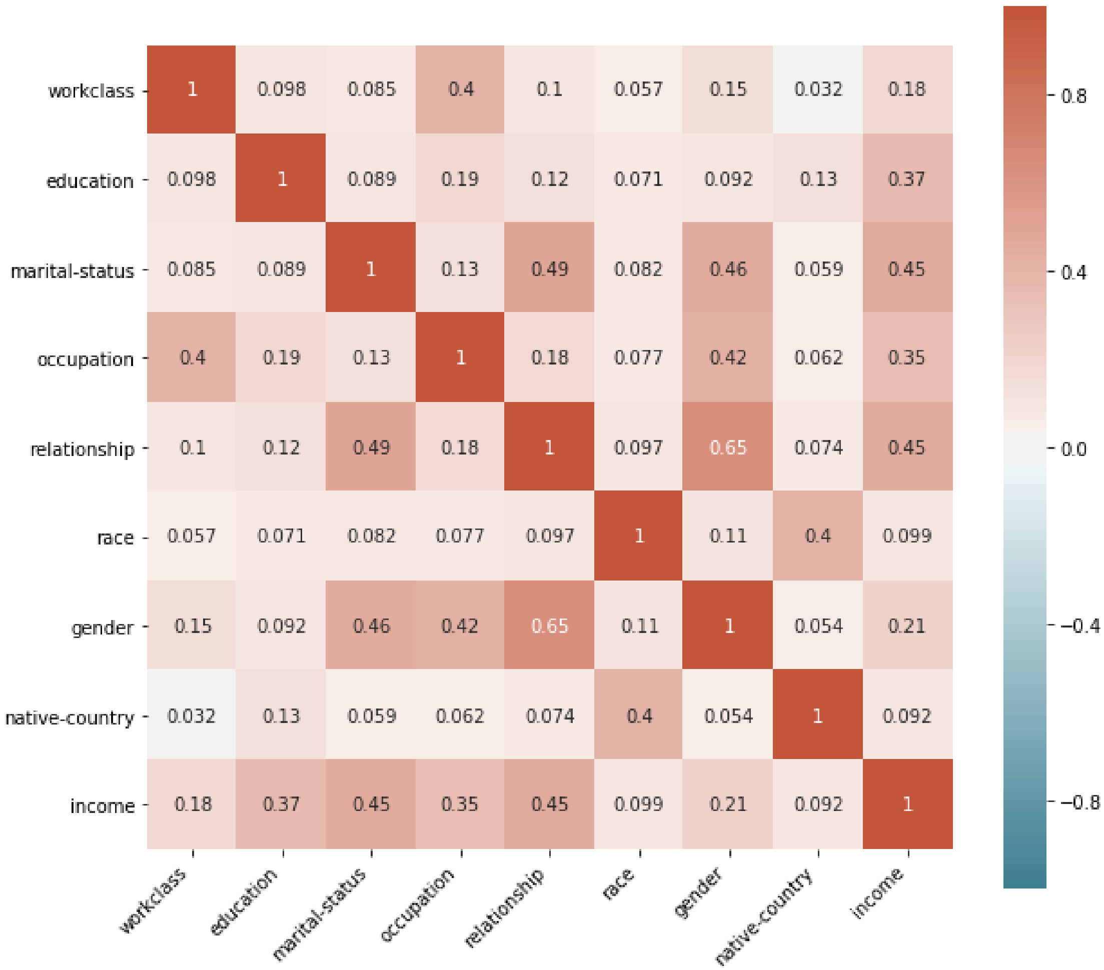
```
Help on function show_df_correlations in module MLHelper.analyse.eda.eda:

show_df_correlations(df)
    Show differents correlations matrix for 3 cases :
    - numerical to numerical (using Pearson coeff)
    - categorical to categorical (using Cramers V & Chi square)
    - numerical to categorical (discrete) (using Point Biserial)

    Parameters
    ----------
    df: pd.DataFrame
        Dataframe to inspect
```

[20]:
```
eda.show_df_correlations(df=adult)
```

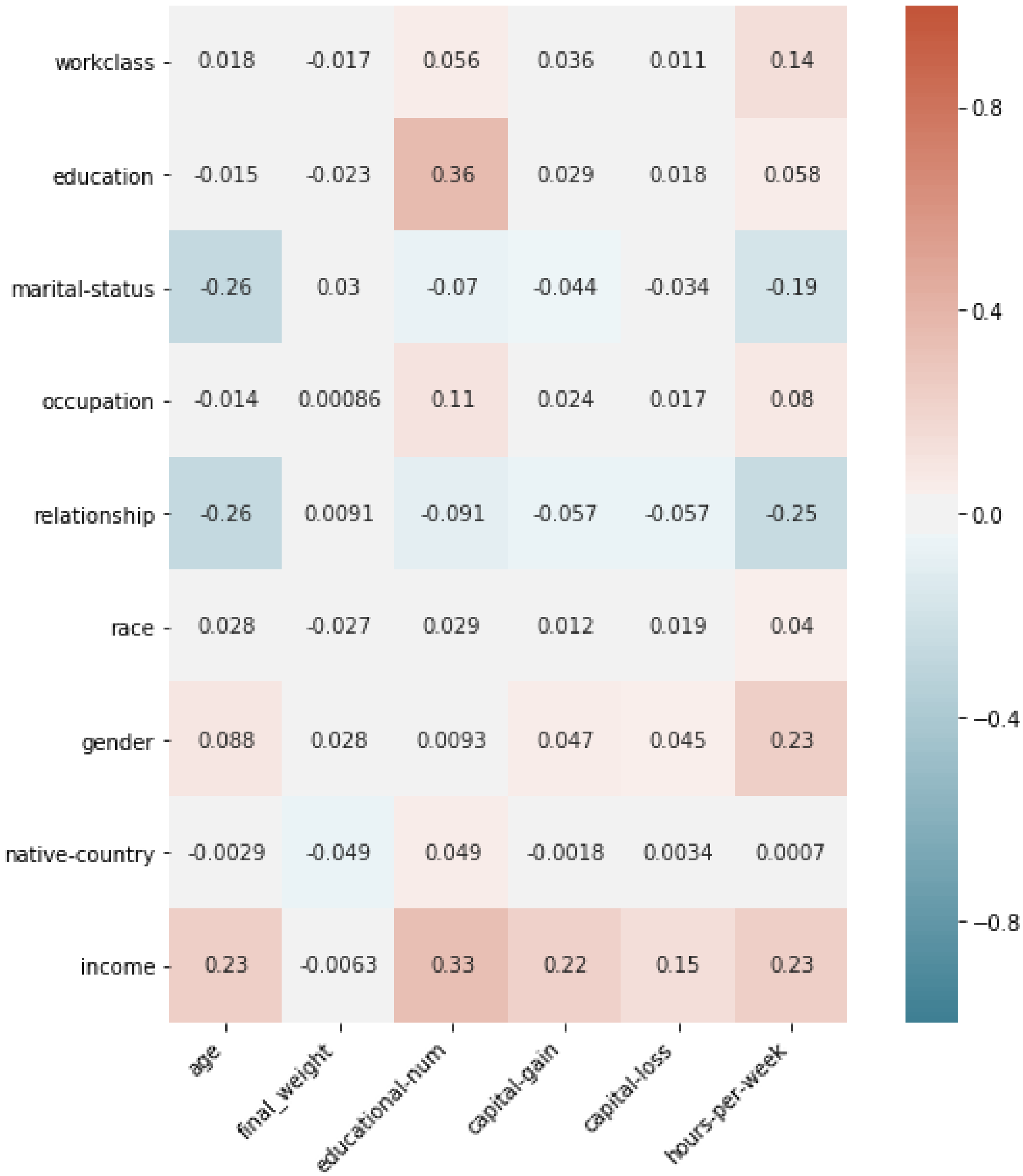
Pearson correlation matrix for numerical variables



Cramers V correlation matrix for categorical variables



Point Biserial correlation matrix for numerical & categorical variables




```
[ ]:
```

```
[ ]:
```

1.8 The end.

Thanks for reading. *Nathan*