exploration

December 25, 2018

Author: Renjini Ramadasan Nair

Description of the data: Source citation: The data represents the census data from the UCI machine learning database (http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data). Detailed information can be read by referring to the original publication (http://robotics.stanford.edu/~ronnyk/nbtree.pdf). There were 15 attributes, of which I have loaded 9 attributes/columns into a dataframe named 'adult' for further analyses. The 9 attributes were the age, workclass, fnlwgt, education-num, sex, capital-gain, capital-loss, hours-per-week and income columns.

```
In [84]: # Import the Numpy and Pandas libraries
         import numpy as np
         import pandas as pd
         import matplotlib.pyplot as plt
         import seaborn as sb
         from sklearn.preprocessing import StandardScaler
         from sklearn.cluster import KMeans
         from sklearn.ensemble import RandomForestClassifier
         from sklearn.linear_model import LogisticRegression
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import *

In [85]: # Download and load data from the url: http://archive.ics.uci.edu/ml/machine-learning
         url = "http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"
         adult1 = pd.read_csv(url, header=None)

         # Add column names
         adult1.columns = ["age", "workclass", "fnlwgt", "education", "education-num", "marita

         # Select the columns needed to fulfill the current objectives; save as a new datafram
         adult = adult1.loc[:, ['age', 'workclass', 'fnlwgt', 'education-num', 'sex', "capital-
```

0.0.1 Exploration

```
In [86]: adult.head()

Out[86]:    age          workclass  fnlwgt  education-num      sex  capital-gain  \
         0   39          State-gov   77516             13     Male          2174
```

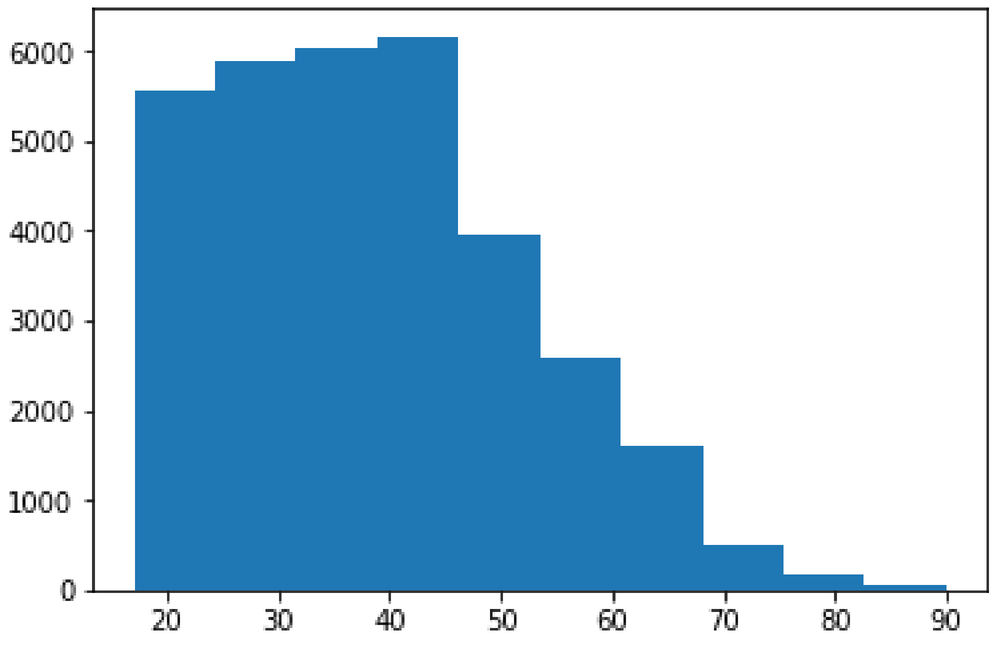
```
    1   50   Self-emp-not-inc   83311             13     Male              0
    2   38            Private  215646              9     Male              0
    3   53            Private  234721              7     Male              0
    4   28            Private  338409             13   Female              0

       capital-loss  hours-per-week  income
    0             0              40   <=50K
    1             0              13   <=50K
    2             0              40   <=50K
    3             0              40   <=50K
    4             0              40   <=50K
```

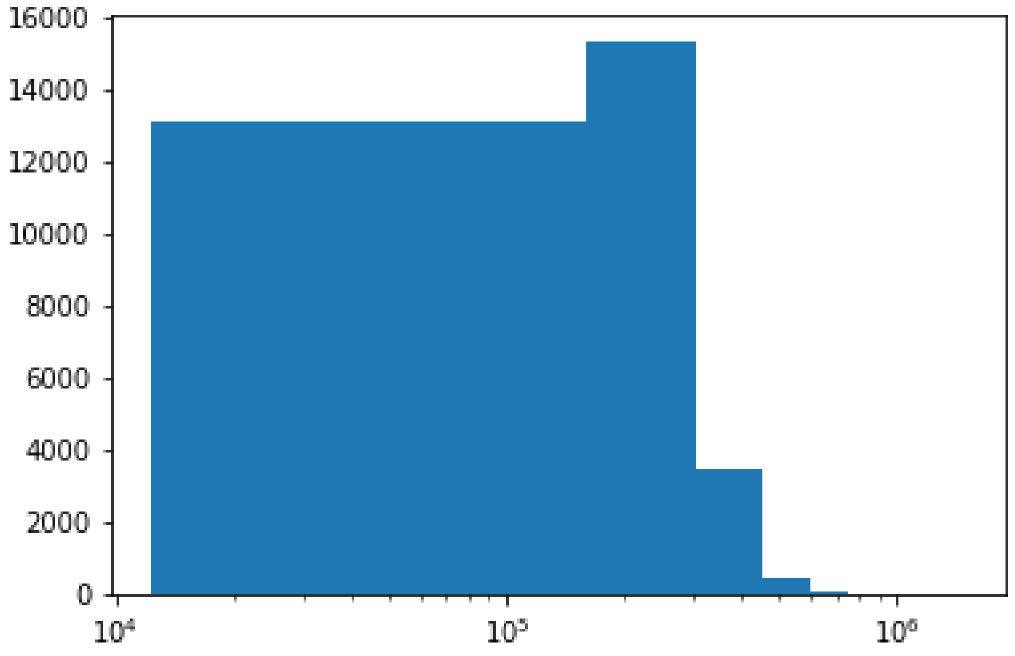
```
In [87]: adult.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 9 columns):
age               32561 non-null int64
workclass         32561 non-null object
fnlwgt            32561 non-null int64
education-num     32561 non-null int64
sex               32561 non-null object
capital-gain      32561 non-null int64
capital-loss      32561 non-null int64
hours-per-week    32561 non-null int64
income            32561 non-null object
dtypes: int64(6), object(3)
memory usage: 2.2+ MB
```

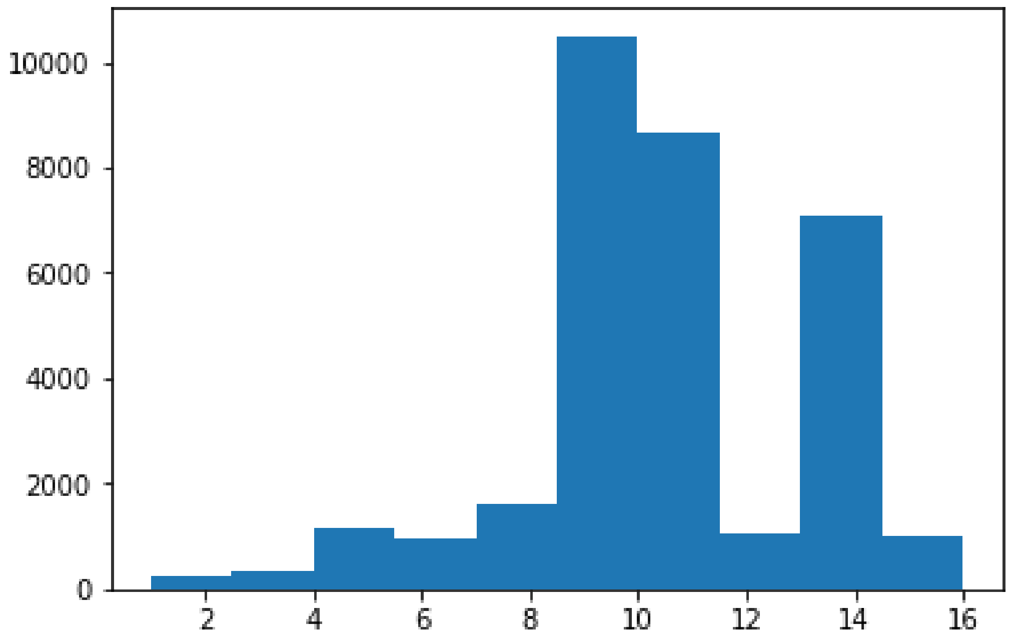
```
In [88]: plt.hist(data = adult, x = 'age');
```



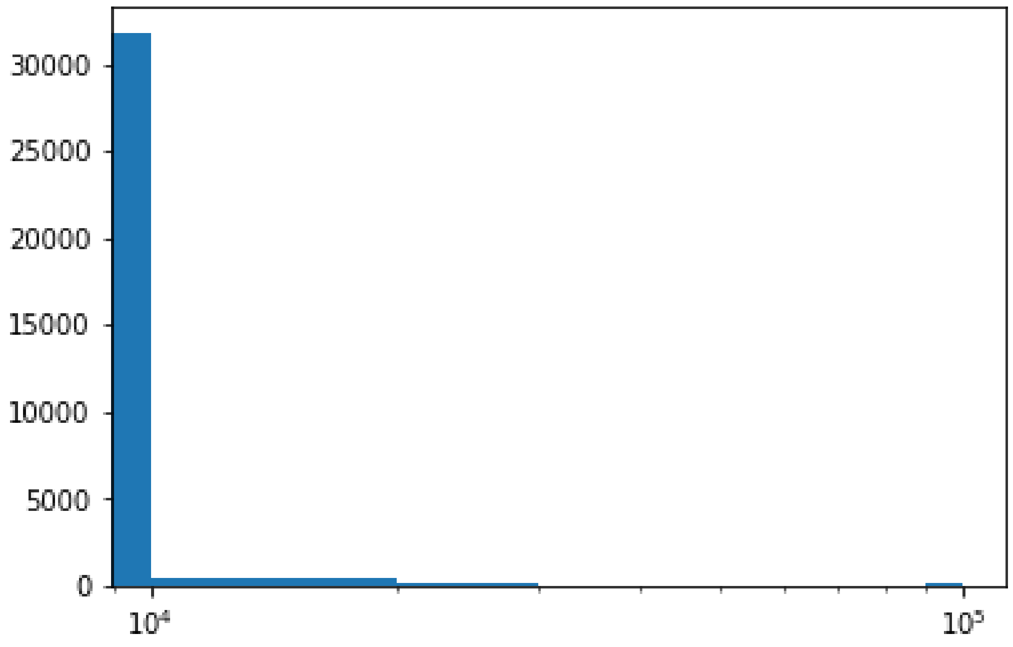
```
In [89]: plt.hist(data = adult, x = 'fnlwgt')
         plt.xscale('log');
```



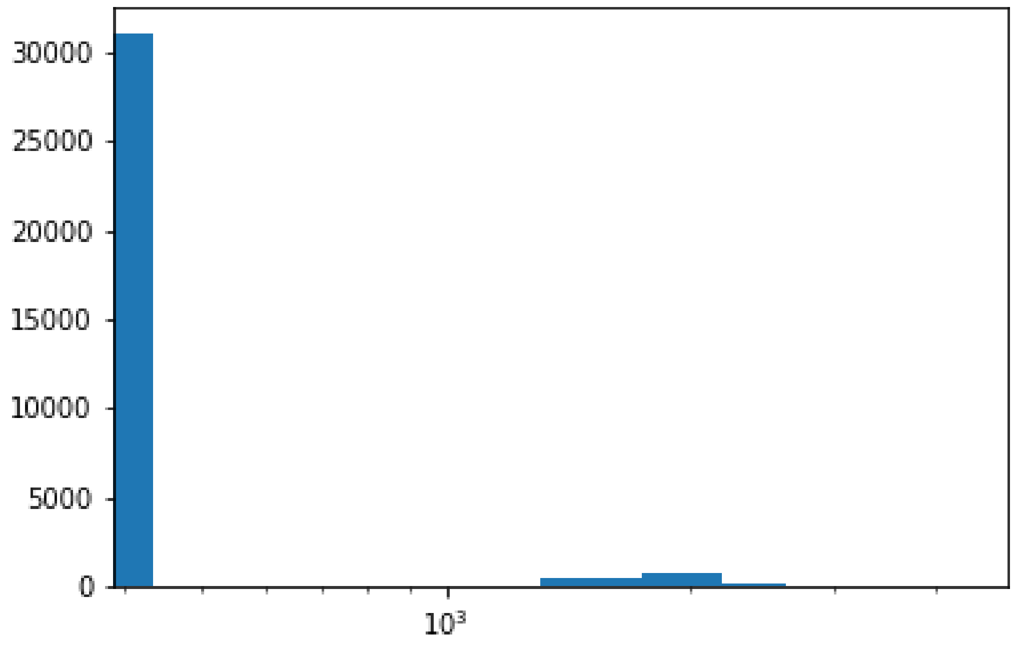
```
In [90]: plt.hist(data = adult, x = 'education-num');
```



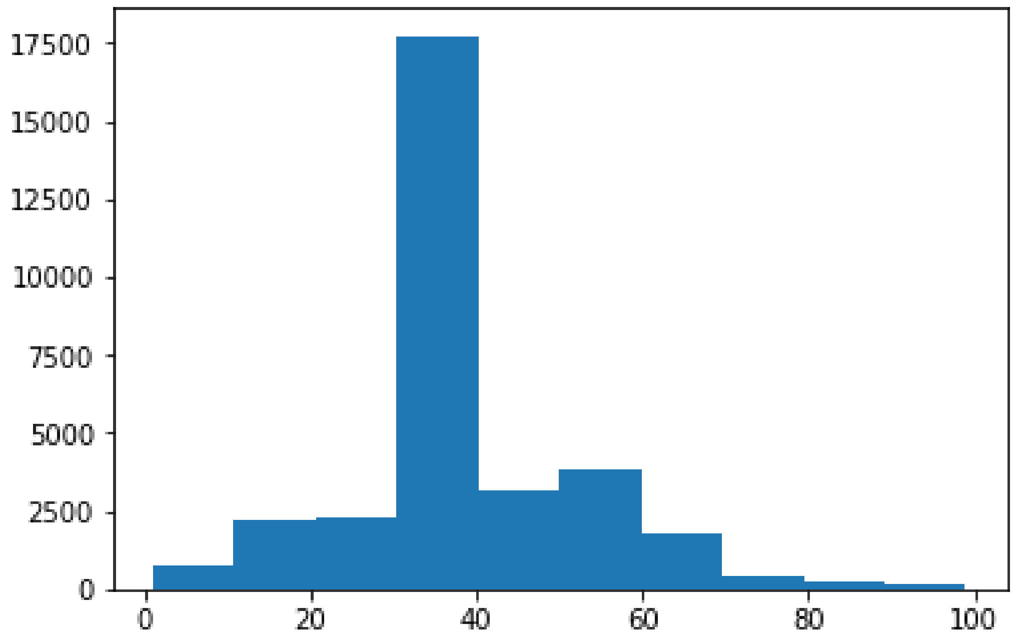
```
In [91]: plt.hist(data = adult, x = 'capital-gain')
         plt.xscale('log');
```



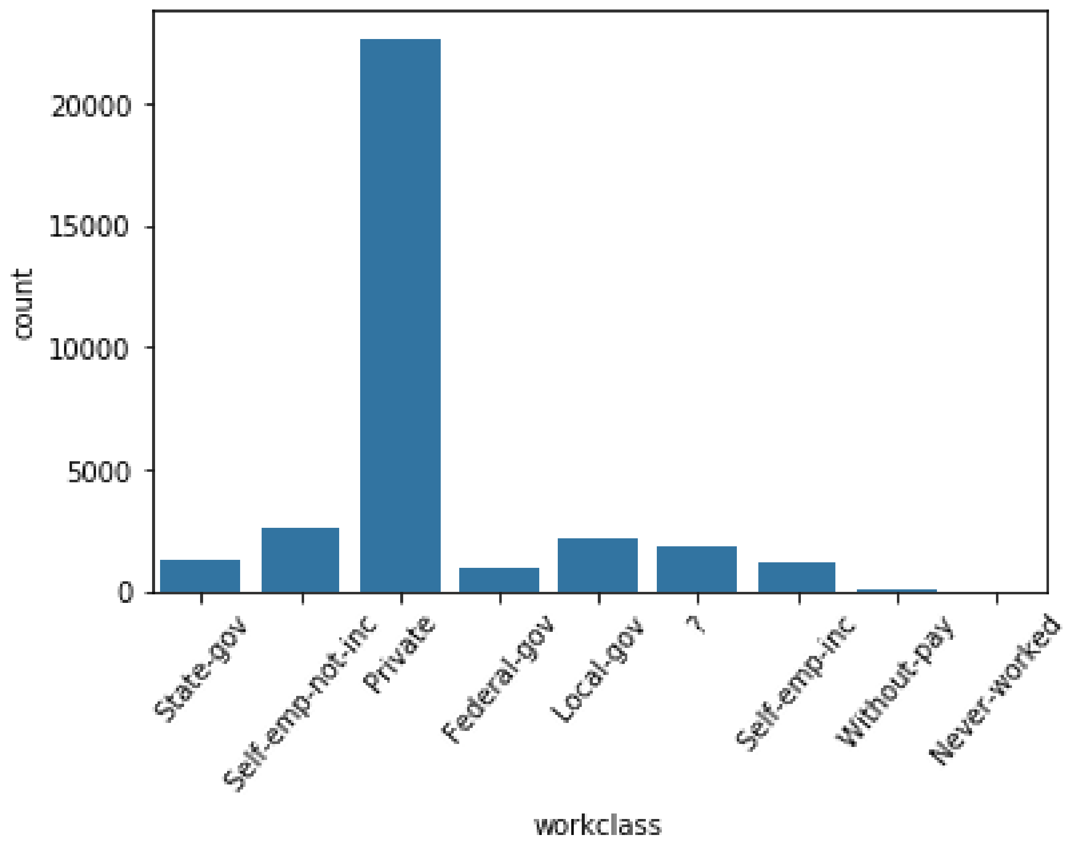
```
In [92]: plt.hist(data = adult, x = 'capital-loss')
         plt.xscale('log');
```



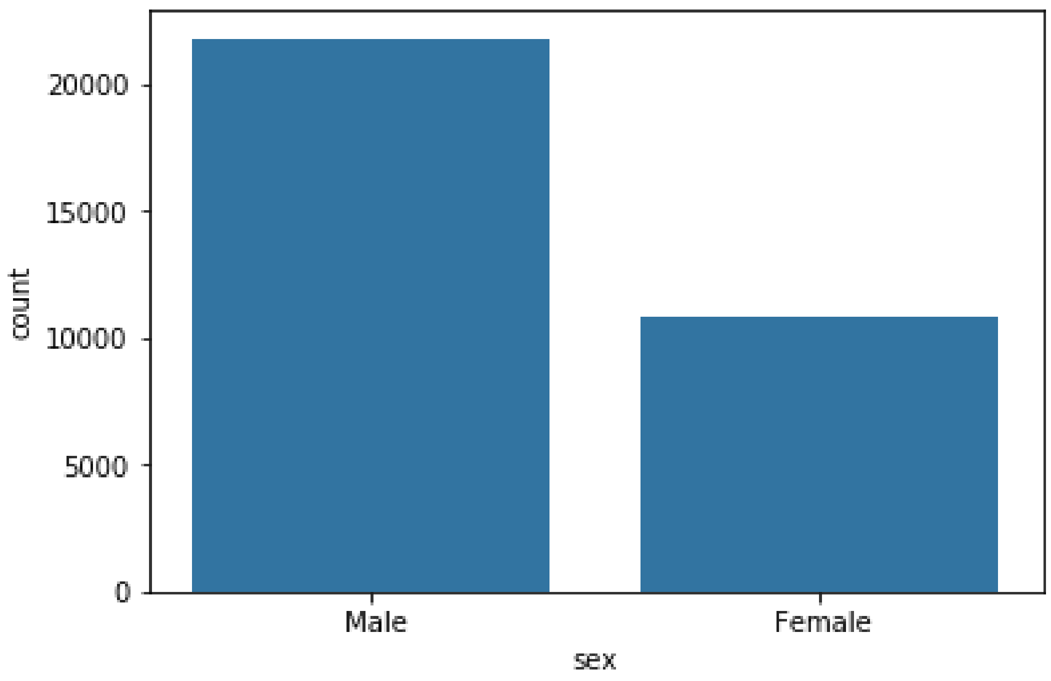
```
In [93]: plt.hist(data = adult, x = 'hours-per-week');
```



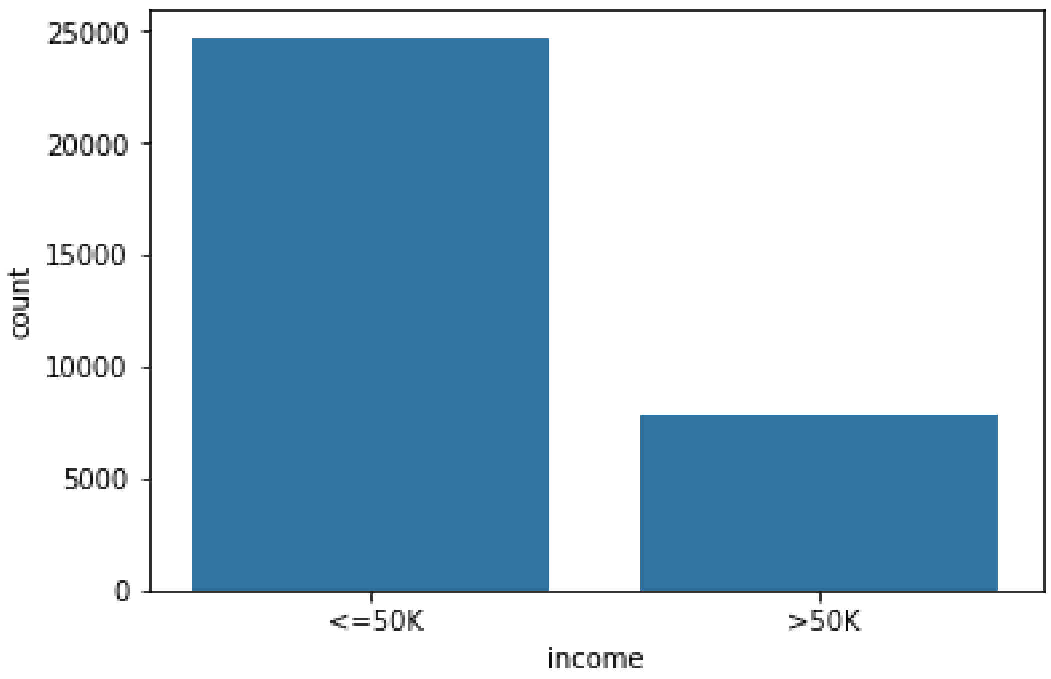
```
In [94]: sb.countplot(data = adult, x = 'workclass', color = sb.color_palette()[0])
         plt.xticks (rotation = 50);
```



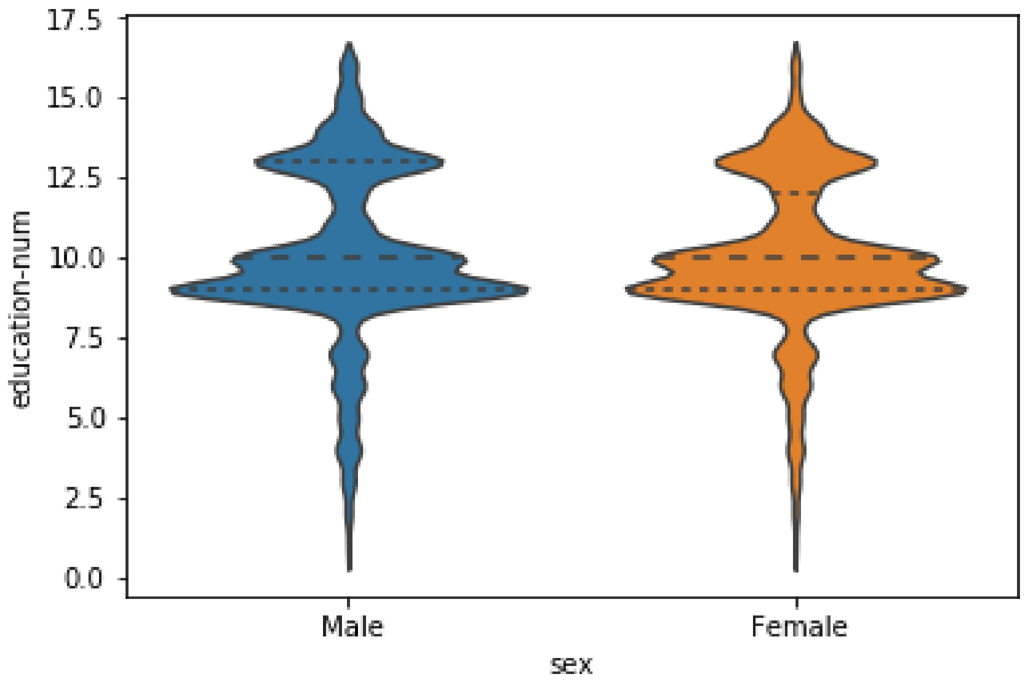
```
In [95]: sb.countplot(data = adult, x = 'sex', color = sb.color_palette()[0]);
```



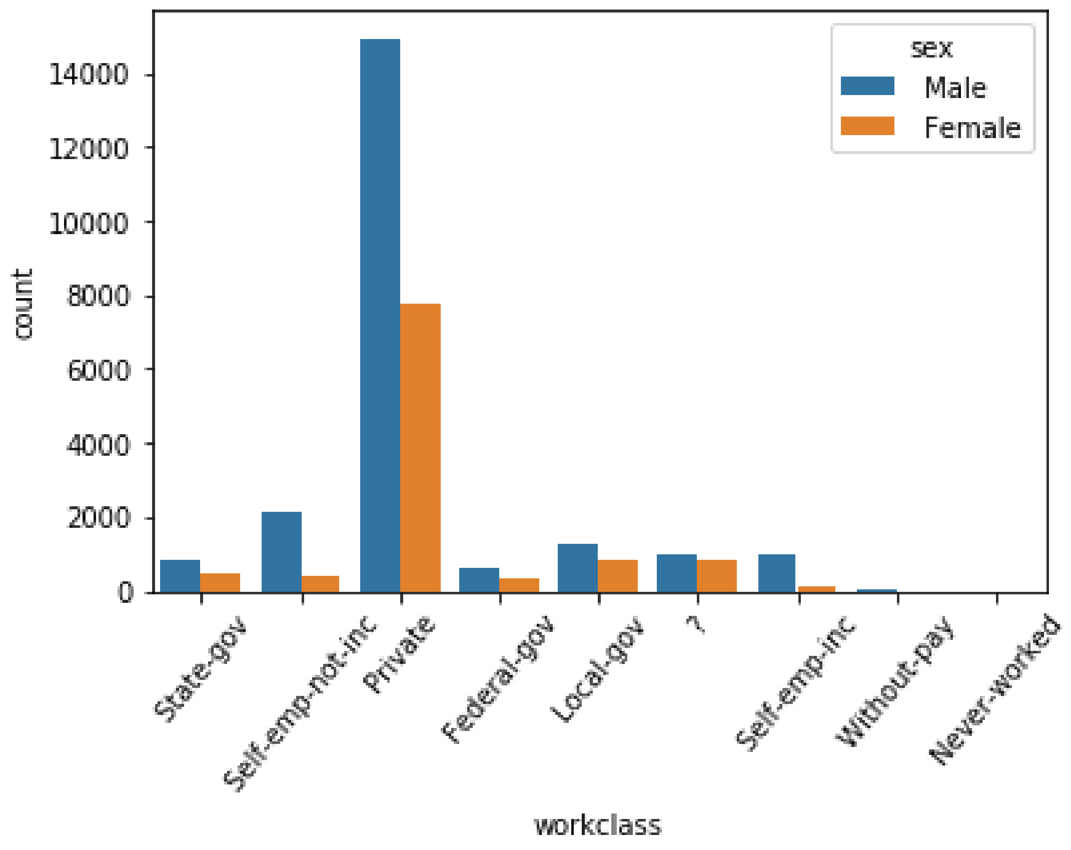
```
In [96]: sb.countplot(data = adult, x = 'income', color = sb.color_palette()[0]);
```



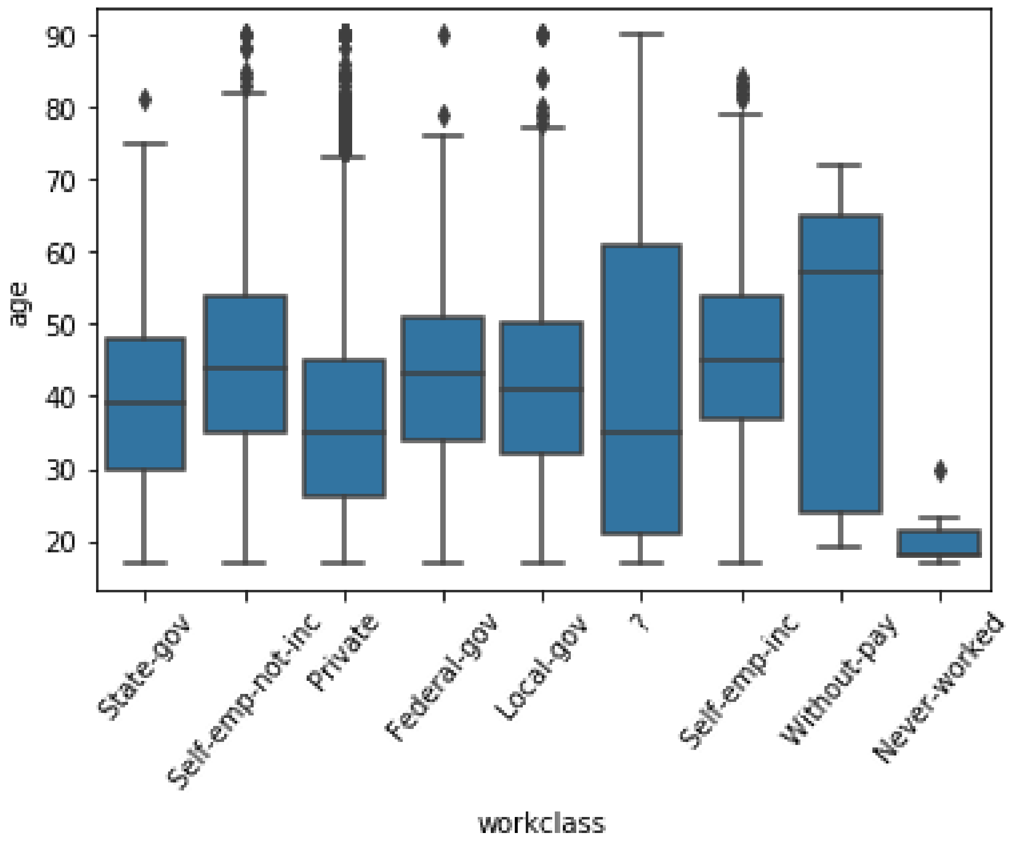
```
In [97]: # Sex related to education
         sb.violinplot(data= adult, x = 'sex', y= 'education-num', inner = 'quartile');
```



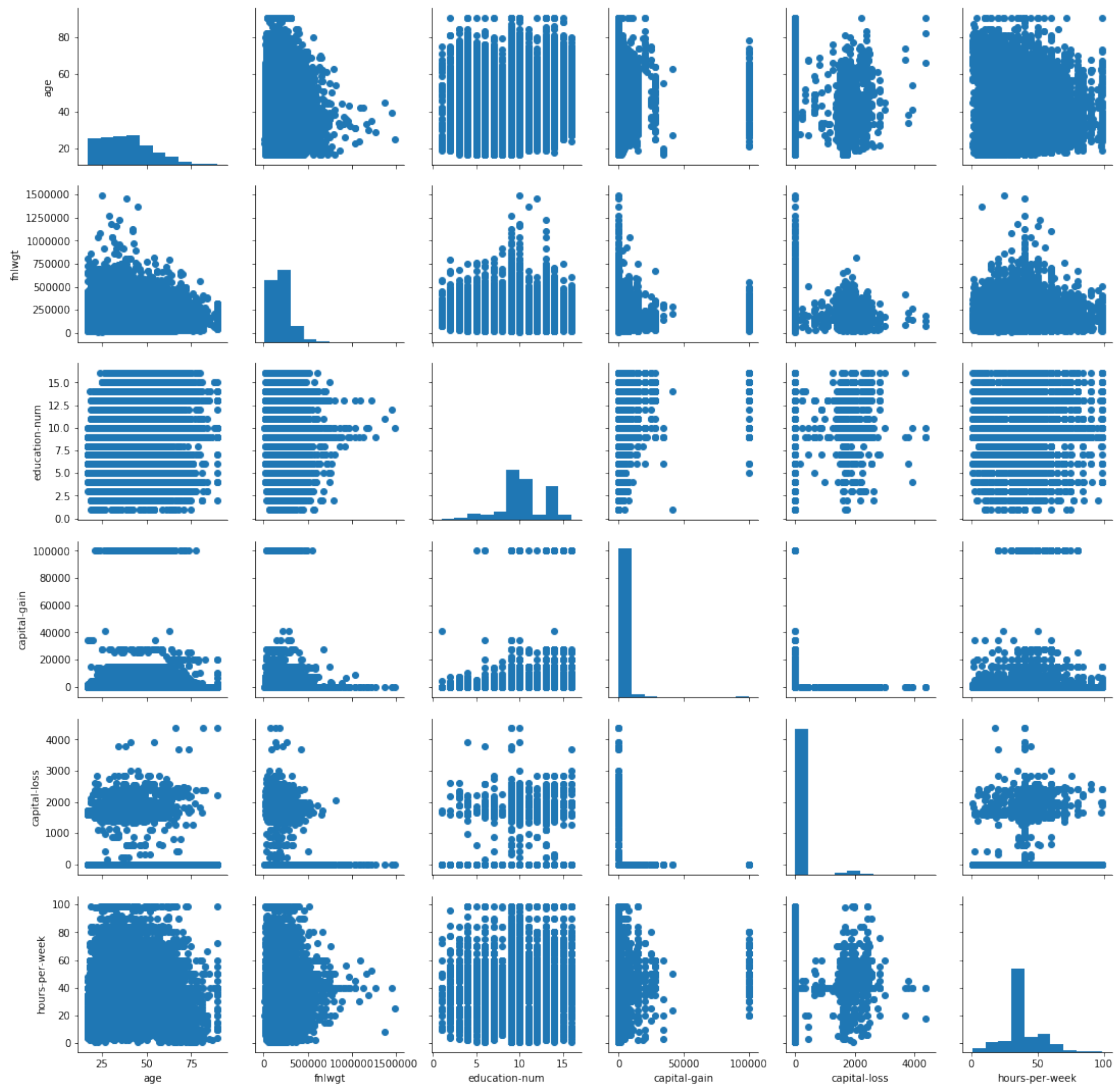
```
In [98]: # Sex related to employment
         sb.countplot(data= adult, x = 'workclass', hue= 'sex')
         plt.xticks(rotation = 50);
```



```
In [99]: # Age related to employment
         sb.boxplot(data= adult, x = 'workclass', y= 'age', color = sb.color_palette()[0])
         plt.xticks(rotation = 50);
```



```
In [100]: # Relations between the numerical columns
          g = sb.PairGrid(data = adult)
          g.map_diag(plt.hist)
          g.map_offdiag(plt.scatter);
```



Of the attributes, age, fnlwgt, capital-gain, capital-loss and hours-per-week were numerical attributes and the rest categorical. Initially there were 32561 rows. There were no null values in the numerical data columns, but were present in some categorical columns. Upon exploration it was seen that the education status of males and females followed a similar trend as the employment, being lower for women across job types.

0.0.2 Data Cleaning

I performed data cleaning by following the steps outlined below. The numerical column 'fnlwgt' had outliers, which were replaced with the median value, followed by z-standardization. A plotted histogram showed that this attribute has a more or less normal distribution. For the 'capital-gain' and 'capital-loss' columns, the rows with a few extreme values (>20000 for capital-gain and >3000 for capital loss) were removed. These two columns were also standardized by z-normalization. For the 'education-num', 'workclass' and 'sex' variables, the categories were decoded, imputed and consolidated. The 'education-num' variable was categorized into Primary (all education lesser than college), Bachelor, Master, Professional and Doctorate. The Doctoral was later consolidated into Professional. The 'workclass' column was categorized into Private, Government and Unemployed categories. The missing values in the 'workclass' column were imputed with the most frequent 'Private' value. The 'sex' column had two categories , 'Male' and 'Female'. Dummy variables were derived for the all three categorical variables and the parental columns deleted. The outliers in the age column were replaced with the median of the non-null values. For the 'age' and 'hours-per-week' columns, the values were binned into 1 (low), 2 (medium) and 3 (high) categories. The obsolete parent columns were deleted.

```
In [101]: # Function definitions

          # Function for checking for outliers in the numeric value columns
          def outlier(var):
              high = np.mean(var) + 2*np.std(var)
              low = np.mean(var) - 2*np.std(var)
              outliers = (var >= high) | (var <= low)
              return outliers

          # Function to bin numerical columns
          def bins(x, n):
              BinWidth = (max(x) - min(x))/n
              bound1 = float('-inf')
              bound2 = min(x) + 1 * BinWidth
              bound3 = min(x) + 2 * BinWidth
              bound4 = float('inf')
              Binned = np.array([" "]*len(x))
              Binned[(bound1 < x) & (x <= bound2)] = 1 # Low
              Binned[(bound2 < x) & (x <= bound3)] = 2 # Med
              Binned[(bound3 < x) & (x  < bound4)] = 3 # High
              return Binned

          # Function for z-standardization of a numerical column
          def norm(col):
              x = np.array(col).astype(float)
              X = pd.DataFrame(x)
              y = StandardScaler().fit(X).transform(X)
              return y

In [102]: # Remove/replace outliers for the numerical columns.

          # Replace outliers with median values for the numerical column, fnlwgt.
          outliers = outlier(adult['fnlwgt'])
          adult.loc[outliers, 'fnlwgt'] = np.median(adult.loc[:,"fnlwgt"])

          # Remove rows with outlier values; For the capital-gain and capital-loss variables
          # capital-gain
          adult.loc[:,"capital-gain"].sort_values()
          high1 = adult.loc[:,"capital-gain"] > 20000
```

```
        adult = adult.loc[~high1, :]
        # capital-loss
        adult.loc[:,"capital-loss"].sort_values()
        high2 = adult.loc[:,"capital-loss"] > 3000
        adult = adult.loc[~high2, :]
```

```
In [103]: # Normalize values for the numerical columns by z-standardization

        # Normalize the fnlwgt column; plot the normalized fnlwgt data.
        col = adult["fnlwgt"]
        adult["fnlwgt"] = norm(col)

        # Normalize the capital-gain column
        col = adult["capital-gain"]
        adult["capital-gain"] = norm(col)

        # Normalize the capital-loss column
        col = adult["capital-loss"]
        adult["capital-loss"] = norm(col)
```

```
In [104]: # Bin the numerical columns (age, hours-per-week)kmeans = KMeans(n_clusters=5).fit(a
        # age - Equal-width Binning using numpy
        x = np.array(adult['age'])
        adult['Binned_age'] = bins(x, 6)
        # hours-per-week - Equal-width Binning using numpy
        x = np.array(adult['hours-per-week'])
        adult['Binned_hours-per-week'] = bins(x, 6)

        # Delete obsolete column age and hours-per-week columns
        adult.drop('age', axis = 1, inplace = True)
        adult.drop('hours-per-week', axis = 1, inplace = True)
```

```
In [105]: # Decode education-num column, impute categories and consolidate
        # Decode and impute
        adult.loc[adult.loc[:, 'education-num'] <=12, 'education-num'] = 'Primary'
        adult.loc[adult.loc[:, 'education-num'] ==13, 'education-num'] = 'Bachelor'
        adult.loc[adult.loc[:, 'education-num'] ==14, 'education-num'] = 'Master'
        adult.loc[adult.loc[:, 'education-num'] ==15, 'education-num'] = 'Professional'
        adult.loc[adult.loc[:, 'education-num'] ==16, 'education-num'] = 'Doctorate'
        # Consolidate professional and doctorate into professional
        adult.loc[adult.loc[:, 'education-num'] =='Doctorate', 'education-num'] = 'Profession

        # Decode the other categorical variable, workclass; impute categories, consolidate a
        adult.loc[(adult.loc[:, 'workclass'] == ' State-gov'), 'workclass'] = 'Government'
        adult.loc[adult.loc[:, 'workclass'] == ' Self-emp-not-inc', 'workclass'] = 'Unemploye
        adult.loc[adult.loc[:, 'workclass'] == ' Federal-gov', 'workclass'] = 'Government'
        adult.loc[adult.loc[:, 'workclass'] == ' Local-gov', 'workclass'] = 'Government'
        adult.loc[adult.loc[:, 'workclass'] == ' Self-emp-inc', 'workclass'] = 'Private'
        adult.loc[adult.loc[:, 'workclass'] == ' Without-pay', 'workclass'] = 'Unemployed'
        adult.loc[adult.loc[:, 'workclass'] == ' Never-worked', 'workclass'] = 'Unemployed'
        adult.loc[adult.loc[:, 'workclass'] == ' Private', 'workclass'] = 'Private'
        # Private seems highest, so impute '?' with Private
        adult.loc[adult.loc[:, 'workclass'] == ' ?', 'workclass'] = 'Private'
        # Save the dataframe as a copy for plotting purpose
        adult2 = adult.copy()
        # add dummy variables
        adult.loc[:, "Government"] = (adult.loc[:, "workclass"] == "Government").astype(int)
        adult.loc[:, "Private"] = (adult.loc[: , "workclass"] == "Private").astype(int)
        adult.loc[:, "Unemployed"] = (adult.loc[:, "workclass"] == "Unemployed").astype(int)
        adult.drop('workclass', axis = 1, inplace = True)

        # add dummy variables for the categorical column education-num
        adult[['Bachelor', 'Master', 'Primary', 'Professional']] = pd.get_dummies(adult['edu
        # Delete the obsolete education-num column
        adult.drop('education-num', axis = 1, inplace = True)

        # add dummy variables for the sex column and delete parent column
        adult[['Female', 'Male']] = pd.get_dummies(adult['sex'])
        adult.drop('sex', axis = 1, inplace = True)

        # Replace the '<=50K' and '>50K' with 1 and 0 respectively, in the income column
        adult.loc[(adult.loc[:, 'income'] == ' <=50K'), 'income'] = 1
        adult.loc[(adult.loc[:, 'income'] == ' >50K'), 'income'] = 0
```

Following the data cleaning, there were 32297 rows and 15 columns in the ′adult′ dataframe. Of the 15, 3 are z-standardized numerical columns named ′fnlwgt′, ′capital-gain′ and ′capital-loss′. ′Binned_age′ and ′Binned_hours-per-week′ are binned columns of the original numerical attributes. There were 2, 3 and 4 columns derived by one-hot encoding of the ′sex′, ′workclass′ and ′education-num′ categorical columns respectively. The ′income′ column represents the expert labels, claasifying each instance into income <= or > than 50000.