Problem Statement

In this assignment students need to predict whether a person makes over 50K per year or not from classic adult dataset using XGBoost. The description of the dataset is as follows:

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))

Attribute Information: Listing of attributes:

- Salary >50K, <=50K.
- age: continuous.
- workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- fnlwgt: continuous.
- 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.
- education-num: continuous.
- marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
- 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.
- relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.
- race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
- sex: Female, Male. -capital-gain: continuous.
- capital-loss: continuous.
- hours-per-week: continuous.
- 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 [1]:
```
import numpy as np
import pandas as pd
train_set = pd.read_csv('adult_train.csv', header = None)
test_set = pd.read_csv('adult_test.csv', header = None)
col_labels = ['age', 'workclass', 'fnlwgt', 'education', 'education_num', 'marital_status', 'occupation','relationship', 'race'
train_set.columns = col_labels
test_set.columns = col_labels
```

In [2]:
```python
# removing white spaces in the train dataset
train_set.replace('^\s+', '', regex=True, inplace=True) #front
train_set.replace('\s+$', '', regex=True, inplace=True) #end


# Replacing married and unmarried
train_set.replace(['Divorced', 'Married-AF-spouse',
              'Married-civ-spouse', 'Married-spouse-absent',
              'Never-married','Separated','Widowed'],
             ['not married','married','married','married',
              'not married','not married','not married'], inplace = True)
train_set.head()
```

Out[2]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week
0	39	State-gov	77516	Bachelors	13.0	not married	Adm-clerical	Not-in-family	White	Male	2174.0	0.0	40.0
1	50	Self-emp-not-inc	83311	Bachelors	13.0	married	Exec-managerial	Husband	White	Male	0.0	0.0	13.0
2	38	Private	215646	HS-grad	9.0	not married	Handlers-cleaners	Not-in-family	White	Male	0.0	0.0	40.0
3	53	Private	234721	11th	7.0	married	Handlers-cleaners	Husband	Black	Male	0.0	0.0	40.0
4	28	Private	338409	Bachelors	13.0	married	Prof-specialty	Wife	Black	Female	0.0	0.0	40.0

In [3]:
```
# removing white spaces in the test dataset
test_set.replace('^\s+', '', regex=True, inplace=True) #front
test_set.replace('\s+$', '', regex=True, inplace=True) #end
# Replacing married and unmarried
test_set.replace(['Divorced', 'Married-AF-spouse',
              'Married-civ-spouse', 'Married-spouse-absent',
              'Never-married','Separated','Widowed'],
             ['not married','married','married','married',
              'not married','not married','not married'], inplace = True)
test_set.head()
```

Out[3]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week
0	25	Private	226802	11th	7	not married	Machine-op-inspct	Own-child	Black	Male	0.0	0.0	40.0
1	38	Private	89814	HS-grad	9	married	Farming-fishing	Husband	White	Male	0.0	0.0	50.0
2	28	Local-gov	336951	Assoc-acdm	12	married	Protective-serv	Husband	White	Male	0.0	0.0	40.0
3	44	Private	160323	Some-college	10	married	Machine-op-inspct	Husband	Black	Male	7688.0	0.0	40.0
4	18	?	103497	Some-college	10	not married	?	Own-child	White	Female	0.0	0.0	30.0

In [4]:
```
# defining function for estimating missing values in each columns
def missing_value(df):
    miss=[]
    col_list=df.columns
    for i in col_list:
        missing=df[i].isnull().sum()
        miss.append(missing)
        list_of_missing=pd.DataFrame(list(zip(col_list,miss)))
    return list_of_missing
```

In [5]:
```
print("Training Set ======================")
print(missing_value(train_set))

print("Test Set ======================")
print(missing_value(test_set))
```

```
Training Set ======================
                 0  1
0              age  0
1        workclass  0
2           fnlwgt  0
3        education  0
4    education_num  1
5   marital_status  1
6       occupation  1
7     relationship  1
8             race  1
9              sex  1
10    capital_gain  1
11    capital_loss  1
12  hours_per_week  1
13  native_country  1
14      wage_class  1
Test Set ======================
                 0  1
0              age  0
1        workclass  0
2           fnlwgt  0
3        education  0
4    education_num  0
5   marital_status  0
6       occupation  0
7     relationship  1
8             race  1
9              sex  1
10    capital_gain  1
11    capital_loss  1
12  hours_per_week  1
13  native_country  1
14      wage_class  1
```

Don't See any null / missing value in the Train and Test Data Set

In [6]:
```python
# Checking the unique Values in the training dataset to check the correctness of data
print("Work Class ===", train_set.workclass.unique())
print("-"*100)
print("Age ===", train_set.age.unique())
print("-"*100)
print("fnlwgt ===", train_set.fnlwgt.unique())
print("-"*100)
print("education ===", train_set.education.unique())
print("-"*100)
print("education_num ===", train_set.education_num.unique())
print("-"*100)
print("marital_status ===", train_set.marital_status.unique())
print("-"*100)
print("occupation ===", train_set.occupation.unique())
print("-"*100)
print("relationship ===", train_set.relationship.unique())
print("-"*100)
print("race ===", train_set.race.unique())
print("-"*100)
print("sex ===", train_set.sex.unique())
print("-"*100)
print("capital_gain ===", train_set.capital_gain.unique())
print("-"*100)
print("capital_loss ===", train_set.capital_loss.unique())
print("-"*100)
print("hours_per_week ===", train_set.hours_per_week.unique())
print("-"*100)
print("native_country ===", train_set.native_country.unique())
print("-"*100)
print("wage_class ===", train_set.wage_class.unique())
```

```
Work Class === ['State-gov' 'Self-emp-not-inc' 'Private' 'Federal-gov' 'Local-gov' '?'
 'Self-emp-inc' 'Without-pay' 'Never-worked']
----------------------------------------------------------------------------------------------------
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]
----------------------------------------------------------------------------------------------------
fnlwgt === [ 77516  83311 215646 ... 115066 223751 354075]
----------------------------------------------------------------------------------------------------
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']
----------------------------------------------------------------------------------------------------
education_num === [13.  9.  7. 14.  5. 10. 12. 11.  4. 16. 15.  3.  6.  2.  1.  8. nan]
----------------------------------------------------------------------------------------------------
marital_status === ['not married' 'married' nan]
```

```
----------------------------------------------------------------------------------------------------
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' nan]
----------------------------------------------------------------------------------------------------
relationship === ['Not-in-family' 'Husband' 'Wife' 'Own-child' 'Unmarried' 'Other-relative'
 nan]
----------------------------------------------------------------------------------------------------
race === ['White' 'Black' 'Asian-Pac-Islander' 'Amer-Indian-Eskimo' 'Other' nan]
----------------------------------------------------------------------------------------------------
sex === ['Male' 'Female' nan]
----------------------------------------------------------------------------------------------------
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.    nan]
----------------------------------------------------------------------------------------------------
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.   nan]
----------------------------------------------------------------------------------------------------
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.
 nan]
----------------------------------------------------------------------------------------------------
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' nan]
```

```
-----------------------------------------------------------------------------------------------------
wage_class === ['<=50K' '>50K' nan]
```

By finding the uniques values we found "?" is suspicious in workclass, occupation and native_country

In [7]:

```
# Checking the unique Values in the test dataset to check the correctness of data
print("Work Class ===", test_set.workclass.unique())
print("-"*100)
print("Age ===", test_set.age.unique())
print("-"*100)
print("fnlwgt ===", test_set.fnlwgt.unique())
print("-"*100)
print("education ===", test_set.education.unique())
print("-"*100)
print("education_num ===", test_set.education_num.unique())
print("-"*100)
print("marital_status ===", test_set.marital_status.unique())
print("-"*100)
print("occupation ===", test_set.occupation.unique())
print("-"*100)
print("relationship ===", test_set.relationship.unique())
print("-"*100)
print("race ===", test_set.race.unique())
print("-"*100)
print("sex ===", test_set.sex.unique())
print("-"*100)
print("capital_gain ===", test_set.capital_gain.unique())
print("-"*100)
print("capital_loss ===", test_set.capital_loss.unique())
print("-"*100)
print("hours_per_week ===", test_set.hours_per_week.unique())
print("-"*100)
print("native_country ===", test_set.native_country.unique())
print("-"*100)
print("wage_class ===", test_set.wage_class.unique())
```

```
Work Class === ['Private' 'Local-gov' '?' 'Self-emp-not-inc' 'Federal-gov' 'State-gov'
 'Self-emp-inc' 'Without-pay' 'Never-worked']
----------------------------------------------------------------------------------------------------
Age === [25 38 28 44 18 34 29 63 24 55 65 36 26 58 48 43 20 37 40 72 45 22 23 54
 32 46 56 17 39 52 21 42 33 30 47 41 19 69 50 31 59 49 51 27 57 61 64 79
 73 53 77 80 62 35 68 66 75 60 67 71 70 90 81 74 78 82 83 85 76 84 89]
----------------------------------------------------------------------------------------------------
fnlwgt === [226802  89814 336951 ... 174525 161599 193494]
----------------------------------------------------------------------------------------------------
education === ['11th' 'HS-grad' 'Assoc-acdm' 'Some-college' '10th' 'Prof-school'
 '7th-8th' 'Bachelors' 'Masters' 'Doctorate' '5th-6th' 'Assoc-voc' '9th'
 '12th' '1st-4th' 'Preschool']
----------------------------------------------------------------------------------------------------
education_num === [ 7  9 12 10  6 15  4 13 14 16  3 11  5  8  2  1]
----------------------------------------------------------------------------------------------------
marital_status === ['not married' 'married']
```

```
--------------------------------------------------------------------------------------------------
occupation === ['Machine-op-inspct' 'Farming-fishing' 'Protective-serv' '?'
 'Other-service' 'Prof-specialty' 'Craft-repair' 'Adm-clerical'
 'Exec-managerial' 'Tech-support' 'Sales' 'Priv-house-serv'
 'Transport-moving' 'Handlers-cleaners' 'Armed-Forces' 'Craft']
--------------------------------------------------------------------------------------------------
relationship === ['Own-child' 'Husband' 'Not-in-family' 'Unmarried' 'Wife' 'Other-relative'
 nan]
--------------------------------------------------------------------------------------------------
race === ['Black' 'White' 'Asian-Pac-Islander' 'Other' 'Amer-Indian-Eskimo' nan]
--------------------------------------------------------------------------------------------------
sex === ['Male' 'Female' nan]
--------------------------------------------------------------------------------------------------
capital_gain === [    0.  7688.  3103.  6418.  7298.  3908. 14084.  5178. 15024. 99999.
  2597.  2907.  4650.  6497.  1055.  5013. 27828.  4934.  4064.  3674.
  2174. 10605.  3418.   114.  2580.  3411.  4508.  4386.  8614. 13550.
  6849.  2463.  3137.  2885.  2964.  1471. 10566.  2354.  1424.  1455.
  3325.  4416. 25236.   594.  2105.  4787.  2829.   401.  4865.  1264.
  1506. 10520.  3464.  2653. 20051.  4101.  1797.  2407.  3471.  1086.
  1848. 14344.  1151.  2993.  2290. 15020.  9386.  2202.  3818.  2176.
  5455. 11678.  7978.  7262.  6514. 41310.  3456.  7430.  2414.  2062.
 34095.  1831.  6723.  5060. 15831.  2977.  2346.  3273.  2329.  9562.
  2635.  4931.  1731.  6097.   914.  7896.  5556.  1409.  3781.  3942.
  2538.  3887.    nan]
--------------------------------------------------------------------------------------------------
capital_loss === [   0. 1721. 1876. 2415. 1887.  625. 1977. 2057. 1429. 1590. 1485. 2051.
 2377. 1672. 1628. 1902. 1602. 1741. 2444. 1408. 2001. 2042. 1740. 1825.
 1848. 1719. 3004. 2179. 1573. 2205. 1258. 2339. 1726. 2258. 1340. 1504.
 2559. 1668. 1974. 1980. 1564. 2547. 2002. 1669. 1617.  323. 3175. 2472.
 2174. 1579. 2129. 1510. 1735. 2282. 1870. 1411. 1911. 1651. 1092. 1762.
 2457. 2231. 2238.  653. 1138. 2246. 2603. 2392. 1944. 1380.   nan]
--------------------------------------------------------------------------------------------------
hours_per_week === [40. 50. 30. 32. 10. 39. 35. 48. 25. 20. 45. 47.  6. 43. 90. 54. 60. 38.
 36. 18. 24. 44. 56. 28. 16. 41. 22. 55. 14. 33. 37.  8. 12. 70. 15. 75.
 52. 84. 42. 80. 68. 99. 65.  5. 17. 72. 53. 29. 96. 21. 46.  3.  1. 23.
 49. 67. 76.  7.  2. 58. 26. 34.  4. 51. 78. 63. 31. 92. 77. 27. 85. 13.
 19. 98. 62. 66. 57. 11. 86. 59.  9. 64. 73. 61. 88. 79. 89. nan]
--------------------------------------------------------------------------------------------------
native_country === ['United-States' '?' 'Peru' 'Guatemala' 'Mexico' 'Dominican-Republic'
 'Ireland' 'Germany' 'Philippines' 'Thailand' 'Haiti' 'El-Salvador'
 'Puerto-Rico' 'Vietnam' 'South' 'Columbia' 'Japan' 'India' 'Cambodia'
 'Poland' 'Laos' 'England' 'Cuba' 'Taiwan' 'Italy' 'Canada' 'Portugal'
 'China' 'Nicaragua' 'Honduras' 'Iran' 'Scotland' 'Jamaica' 'Ecuador'
 'Yugoslavia' 'Hungary' 'Hong' 'Greece' 'Trinadad&Tobago'
 'Outlying-US(Guam-USVI-etc)' 'France' nan]
--------------------------------------------------------------------------------------------------
wage_class === ['<=50K.' '>50K.' nan]
```

By finding the uniques values we found "?" is suspicious in workclass, occupation and native_country

In [8]:
```
# Column wise unwanted data calculation like "?" in train data set
col_names = train_set.columns
num_data = train_set.shape[0]
for c in col_names:
    num_non = train_set[c].isin(["?"]).sum()
    if num_non > 0:
        print (c)
        print (num_non)
        print ("{0:.2f}%".format(float(num_non) / num_data * 100))
        print ("\n")
```

```
workclass
988
5.58%


occupation
991
5.60%


native_country
321
1.81%
```

In [9]:
```python
# Column wise unwanted data calculation like "?" in test data set
col_names = test_set.columns
num_data = test_set.shape[0]
for c in col_names:
    num_non = test_set[c].isin(["?"]).sum()
    if num_non > 0:
        print (c)
        print (num_non)
        print ("{0:.2f}%".format(float(num_non) / num_data * 100))
        print ("\n")
```

```
workclass
542
6.07%


occupation
543
6.08%


native_country
143
1.60%
```

In [10]:
```python
# Replacing all the "?" data of training and test to np.nan

all_data = [train_set, test_set]
for data in all_data:
    for i in data.columns:
        data[i].replace('?', np.nan, inplace=True)
    #data.dropna(inplace=True)
```

In [11]:
```
train_set.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17703 entries, 0 to 17702
Data columns (total 15 columns):
age               17703 non-null int64
workclass         16715 non-null object
fnlwgt            17703 non-null int64
education         17703 non-null object
education_num     17702 non-null float64
marital_status    17702 non-null object
occupation        16711 non-null object
relationship      17702 non-null object
race              17702 non-null object
sex               17702 non-null object
capital_gain      17702 non-null float64
capital_loss      17702 non-null float64
hours_per_week    17702 non-null float64
native_country    17381 non-null object
wage_class        17702 non-null object
dtypes: float64(4), int64(2), object(9)
memory usage: 2.0+ MB
```

In [12]:
```
test_set.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8933 entries, 0 to 8932
Data columns (total 15 columns):
age               8933 non-null int64
workclass         8391 non-null object
fnlwgt            8933 non-null int64
education         8933 non-null object
education_num     8933 non-null int64
marital_status    8933 non-null object
occupation        8390 non-null object
relationship      8932 non-null object
race              8932 non-null object
sex               8932 non-null object
capital_gain      8932 non-null float64
capital_loss      8932 non-null float64
hours_per_week    8932 non-null float64
native_country    8789 non-null object
wage_class        8932 non-null object
dtypes: float64(3), int64(3), object(9)
memory usage: 1.0+ MB
```

In [13]:
```
#train_set = train_set.applymap(str)
#train_set.info()
```

In [14]:
```
test_set.isnull().T.any().T.sum()
#test_set.isnull().T.any().T.sum()
#count = 0
#if test_set.isnull().any(axis=1):
#    count = count+1
#count
#test_set[test_set.isNaN().any(axis=1)]
```

Out[14]:
```
675
```

In [15]:
```
print(train_set.isnull().T.any().T.sum()*100/train_set.shape[0])
print(test_set.isnull().T.any().T.sum()*100/test_set.shape[0])
```

```
7.332090606111959
7.556252098958916
```

7.4 Percent of rows are affected by unusual character "?" in Training Set

7.5 Percent of rows are affected by unusual character "?" in Test Set

Deleting all such rows

In [16]:
```
print("Training Set",train_set.shape)
print("Test Set",test_set.shape)
```

```
Training Set (17703, 15)
Test Set (8933, 15)
```

In [17]:
```
# Deleting NaN Rows in train dataset
train_set.dropna( axis=0, inplace = True)
```

In [18]:
```
# Deleting NaN Rows in test dataset
test_set.dropna( axis=0, inplace = True)
```

In [19]:
```
print("Training Set",train_set.shape)
print("Test Set",test_set.shape)
```

```
Training Set (16405, 15)
Test Set (8258, 15)
```

In [20]:
```
train_set.head()
```

Out[20]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week
0	39	State-gov	77516	Bachelors	13.0	not married	Adm-clerical	Not-in-family	White	Male	2174.0	0.0	40.0
1	50	Self-emp-not-inc	83311	Bachelors	13.0	married	Exec-managerial	Husband	White	Male	0.0	0.0	13.0
2	38	Private	215646	HS-grad	9.0	not married	Handlers-cleaners	Not-in-family	White	Male	0.0	0.0	40.0
3	53	Private	234721	11th	7.0	married	Handlers-cleaners	Husband	Black	Male	0.0	0.0	40.0
4	28	Private	338409	Bachelors	13.0	married	Prof-specialty	Wife	Black	Female	0.0	0.0	40.0

In [21]:
```
# Let's convert wage_class to 0, 1
train_set1=train_set
train_set1.head()
```

Out[21]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week
0	39	State-gov	77516	Bachelors	13.0	not married	Adm-clerical	Not-in-family	White	Male	2174.0	0.0	40.0
1	50	Self-emp-not-inc	83311	Bachelors	13.0	married	Exec-managerial	Husband	White	Male	0.0	0.0	13.0
2	38	Private	215646	HS-grad	9.0	not married	Handlers-cleaners	Not-in-family	White	Male	0.0	0.0	40.0
3	53	Private	234721	11th	7.0	married	Handlers-cleaners	Husband	Black	Male	0.0	0.0	40.0
4	28	Private	338409	Bachelors	13.0	married	Prof-specialty	Wife	Black	Female	0.0	0.0	40.0

In [22]:
```python
# Encode the categorical features as numbers for training set
import sklearn.preprocessing as preprocessing
def number_encode_features(df):
    result = df.copy()
    encoders = {}
    for column in result.columns:
        if result.dtypes[column] == np.object:
            encoders[column] = preprocessing.LabelEncoder()
            result[column] = encoders[column].fit_transform(result[column])
    return result, encoders

# Calculate the correlation and plot it
encoded_train_set,encoders = number_encode_features(train_set1)
#sns.heatmap(encoded_data.corr(), square=True)
#plt.show()
encoded_train_set.head()
#encoders
```

Out[22]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	nati
0	39	5	77516	9	13.0	1	0	1	4	1	2174.0	0.0	40.0	
1	50	4	83311	9	13.0	0	3	0	4	1	0.0	0.0	13.0	
2	38	2	215646	11	9.0	1	5	1	4	1	0.0	0.0	40.0	
3	53	2	234721	1	7.0	0	5	0	2	1	0.0	0.0	40.0	
4	28	2	338409	9	13.0	0	9	5	2	0	0.0	0.0	40.0	

In [23]:
```python
train_set.head()
```

Out[23]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week
0	39	State-gov	77516	Bachelors	13.0	not married	Adm-clerical	Not-in-family	White	Male	2174.0	0.0	40.0
1	50	Self-emp-not-inc	83311	Bachelors	13.0	married	Exec-managerial	Husband	White	Male	0.0	0.0	13.0
2	38	Private	215646	HS-grad	9.0	not married	Handlers-cleaners	Not-in-family	White	Male	0.0	0.0	40.0
3	53	Private	234721	11th	7.0	married	Handlers-cleaners	Husband	Black	Male	0.0	0.0	40.0
4	28	Private	338409	Bachelors	13.0	married	Prof-specialty	Wife	Black	Female	0.0	0.0	40.0

In [24]:
```
train_set1.head()
```

Out[24]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week
0	39	State-gov	77516	Bachelors	13.0	not married	Adm-clerical	Not-in-family	White	Male	2174.0	0.0	40.0
1	50	Self-emp-not-inc	83311	Bachelors	13.0	married	Exec-managerial	Husband	White	Male	0.0	0.0	13.0
2	38	Private	215646	HS-grad	9.0	not married	Handlers-cleaners	Not-in-family	White	Male	0.0	0.0	40.0
3	53	Private	234721	11th	7.0	married	Handlers-cleaners	Husband	Black	Male	0.0	0.0	40.0
4	28	Private	338409	Bachelors	13.0	married	Prof-specialty	Wife	Black	Female	0.0	0.0	40.0

In [25]:
```
# Encode the categorical features as numbers for test set
import sklearn.preprocessing as preprocessing
def number_encode_features(df):
    result = df.copy()
    encoders = {}
    for column in result.columns:
        if result.dtypes[column] == np.object:
            encoders[column] = preprocessing.LabelEncoder()
            result[column] = encoders[column].fit_transform(result[column])
    return result, encoders

# Calculate the correlation and plot it
encoded_test_set,encoders = number_encode_features(test_set)
#sns.heatmap(encoded_data.corr(), square=True)
#plt.show()
encoded_test_set.head()
#encoders
```

Out[25]:

	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	nativ
0	25	2	226802	1	7	1	6	3	2	1	0.0	0.0	40.0	
1	38	2	89814	11	9	0	4	0	4	1	0.0	0.0	50.0	
2	28	1	336951	7	12	0	10	0	4	1	0.0	0.0	40.0	
3	44	2	160323	15	10	0	6	0	2	1	7688.0	0.0	40.0	
5	34	2	198693	0	6	1	7	1	4	1	0.0	0.0	30.0	

Feature Selection

In [26]:
```
encoded_train_set.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 16405 entries, 0 to 17701
Data columns (total 15 columns):
age               16405 non-null int64
workclass         16405 non-null int64
fnlwgt            16405 non-null int64
education         16405 non-null int64
education_num     16405 non-null float64
marital_status    16405 non-null int64
occupation        16405 non-null int64
relationship      16405 non-null int64
race              16405 non-null int64
sex               16405 non-null int64
capital_gain      16405 non-null float64
capital_loss      16405 non-null float64
hours_per_week    16405 non-null float64
native_country    16405 non-null int64
wage_class        16405 non-null int64
dtypes: float64(4), int64(11)
memory usage: 2.0 MB
```

In [27]:
```
import matplotlib.pyplot as plt
import seaborn as sns
hmap = encoded_train_set.corr()
plt.subplots(figsize=(12, 9))
sns.heatmap(hmap, vmax=.8,annot=True,cmap="BrBG", square=True);
```

Inferences:

- Married citizens with spouse have higher chances of earning more than those who're unmarried/divorced/widowed/separated.
- Males on an average make earn more than females.
- Higher Education can lead to higher income in most cases.
- Asian-Pacific-Islanders and white are two races that have the highest average income.

In [28]:

```
# col_labels = ['age', 'workclass', 'fnlwgt', 'education', 'education_num', 'marital_status', 'occupation','relationship', 'rac
from sklearn.model_selection import train_test_split
from sklearn import metrics

from xgboost import XGBClassifier

X2=encoded_train_set[['education_num','age','hours_per_week', 'capital_gain']].values
y2= encoded_train_set[['wage_class']].values

X2_train, X2_test, y2_train, y2_test = train_test_split(X2 ,y2, test_size=0.3, random_state=21, stratify=y2)

# fit model no training data
xgbc = XGBClassifier()
xgbc.fit(X2_train, y2_train)
prediction2=xgbc.predict(X2_test)


print('The accuracy of the xGB is',metrics.accuracy_score(prediction2,y2_test))
```

```
C:\Users\prashant_gupta1\AppData\Local\Continuum\anaconda3\lib\site-packages\sklearn\preprocessing\label.py:95: DataConversio
nWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for examp
le using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\prashant_gupta1\AppData\Local\Continuum\anaconda3\lib\site-packages\sklearn\preprocessing\label.py:128: DataConversi
onWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for exam
ple using ravel().
  y = column_or_1d(y, warn=True)

The accuracy of the xGB is 0.8232425843153189

C:\Users\prashant_gupta1\AppData\Local\Continuum\anaconda3\lib\site-packages\sklearn\preprocessing\label.py:151: DeprecationW
arning: The truth value of an empty array is ambiguous. Returning False, but in future this will result in an error. Use `arr
ay.size > 0` to check that an array is not empty.
  if diff:
```

In [29]:

```
# Final test Set

X3=encoded_test_set[['education_num','age','hours_per_week', 'capital_gain']].values
y3= encoded_test_set[['wage_class']].values

prediction3=xgbc.predict(X3)
print('The final accuracy of the xGB is',metrics.accuracy_score(prediction3,y3))
```

```
The final accuracy of the xGB is 0.8240494066359894

C:\Users\prashant_gupta1\AppData\Local\Continuum\anaconda3\lib\site-packages\sklearn\preprocessing\label.py:151: DeprecationW
arning: The truth value of an empty array is ambiguous. Returning False, but in future this will result in an error. Use `arr
ay.size > 0` to check that an array is not empty.
  if diff:
```