adult_data_Classificatioin

April 24, 2018

```
In [64]: # ensemle model KN, naive(gaussiannb) , decisiontree on voting classifier
         import warnings
         warnings.filterwarnings('ignore')
         %matplotlib inline
         import matplotlib.pyplot as plt
         import pandas as pd
         import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.naive_bayes import GaussianNB
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn.ensemble import VotingClassifier
```

```
In [65]: indexes = ['age','workclass','fnlwgt','education','education-num','marital-status','occ
         'relationship','race','sex','capital-gain','capital-loss','hours-per-week','native-coun
```

```
In [66]: df = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult
         dff = df
```

```
In [67]: df.head(3)
```

```
Out[67]:    age          workclass  fnlwgt   education  education-num  \
         0   39          State-gov   77516   Bachelors             13
         1   50   Self-emp-not-inc   83311   Bachelors             13
         2   38            Private  215646     HS-grad              9

                 marital-status          occupation    relationship    race    sex  \
         0        Never-married        Adm-clerical   Not-in-family   White   Male
         1   Married-civ-spouse     Exec-managerial         Husband   White   Male
         2             Divorced   Handlers-cleaners   Not-in-family   White   Male

            capital-gain  capital-loss  hours-per-week  native-country  earned
         0          2174             0              40   United-States   <=50K
         1             0             0              13   United-States   <=50K
         2             0             0              40   United-States   <=50K
```

```
In [68]: df['workclass'] = df['workclass'].replace(' ?',df['workclass'].max())
         df['occupation']= df['occupation'].replace(' ?',df['occupation'].max())
```

```
In [69]: earned = df['earned']
```

```
In [70]: df = df.drop('earned',axis=1)
```

```
In [71]: df.head(3)
```

```
Out[71]:    age           workclass  fnlwgt   education  education-num  \
         0   39           State-gov   77516   Bachelors             13
         1   50    Self-emp-not-inc   83311   Bachelors             13
         2   38             Private  215646     HS-grad              9

                 marital-status          occupation    relationship    race    sex  \
         0        Never-married        Adm-clerical   Not-in-family   White   Male
         1   Married-civ-spouse     Exec-managerial         Husband   White   Male
         2             Divorced   Handlers-cleaners   Not-in-family   White   Male

            capital-gain  capital-loss  hours-per-week  native-country
         0          2174             0              40   United-States
         1             0             0              13   United-States
         2             0             0              40   United-States
```

```
In [72]: mylist = list(df.select_dtypes(include=['object']).columns)
         df = pd.get_dummies(df, prefix= mylist)
         df.head(3)
```

```
Out[72]:    age  fnlwgt  education-num  capital-gain  capital-loss  hours-per-week  \
         0   39   77516             13          2174             0              40
         1   50   83311             13             0             0              13
         2   38  215646              9             0             0              40

            workclass_ Federal-gov  workclass_ Local-gov  workclass_ Never-worked  \
         0                       0                     0                        0
         1                       0                     0                        0
         2                       0                     0                        0

            workclass_ Private             ...              native-country_ Portugal  \
         0                   0             ...                                     0
         1                   0             ...                                     0
         2                   1             ...                                     0

            native-country_ Puerto-Rico  native-country_ Scotland  \
         0                            0                         0
         1                            0                         0
         2                            0                         0

            native-country_ South  native-country_ Taiwan  native-country_ Thailand  \
         0                      0                       0                         0
         1                      0                       0                         0
         2                      0                       0                         0
```

```
           native-country_ Trinadad&Tobago  native-country_ United-States  \
        0                                0                              1
        1                                0                              1
        2                                0                              1

           native-country_ Vietnam  native-country_ Yugoslavia
        0                        0                           0
        1                        0                           0
        2                        0                           0

        [3 rows x 106 columns]
```

```
In [73]: earned = pd.get_dummies(earned)

In [74]: earned_less_50 = earned[' <=50K']

In [75]: train_x,test_x, train_y,test_y = train_test_split(df,earned_less_50)

In [76]: estimators = []

In [77]: clf1 = GaussianNB()

In [78]: estimators.append(('gaussiannb', clf1))

In [79]: clf2 = DecisionTreeClassifier()

In [80]: estimators.append(('decisiontree', clf2))

In [81]: clf3 = KNeighborsClassifier()

In [82]: estimators.append(('kneighbors',clf3))

In [83]: clf  = VotingClassifier(estimators)

In [84]: clf.fit(train_x,train_y)

Out[84]: VotingClassifier(estimators=[('gaussiannb', GaussianNB(priors=None)), ('decisiontree',
                     max_features=None, max_leaf_nodes=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_sa...owski',
                    metric_params=None, n_jobs=1, n_neighbors=5, p=2,
                    weights='uniform'))],
                  flatten_transform=None, n_jobs=1, voting='hard', weights=None)

In [85]: score = clf.score(test_x,test_y)

In [86]: predicted = clf.predict(test_x)
         mse = np.mean((predicted-test_y)**2)

In [87]: print('Score: ',score, 'MSE: ',mse)
```

```
Score:  0.8164844613683823 MSE:  0.18351553863161774
```

```
In [88]: # for i,j in zip(predicted,test_y):
         #     if i:
         #         print('>50K Predicted: %d Real: %d'%(i,j))
         #     else:
         #         print('<=50K Predicted: %d Real: %d'%(i,j))
```

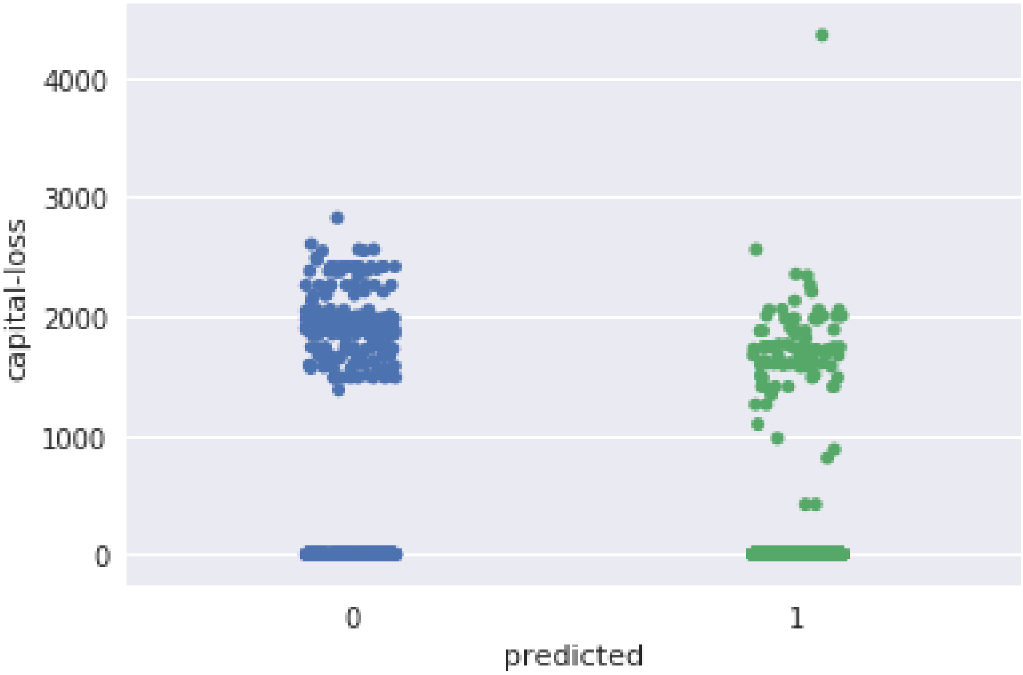
```
In [89]: test_x['predicted'] = predicted.copy()
         test_x.head()
```

```
Out[89]:        age  fnlwgt  education-num  capital-gain  capital-loss  hours-per-week  \
         13739   21  121889             10             0             0              20
         28826   21  205844             10             0             0              25
         20154   34  321787             10             0             0              40
         16438   59  140957             11             0             0              35
         24512   28  177955              7             0             0              40

                workclass_ Federal-gov  workclass_ Local-gov  workclass_ Never-worked  \
         13739                       0                     0                        0
         28826                       0                     0                        0
         20154                       0                     0                        0
         16438                       0                     0                        0
         24512                       0                     0                        0

                workclass_ Private    ...      native-country_ Puerto-Rico  \
         13739                   1    ...                                0
         28826                   1    ...                                0
         20154                   1    ...                                0
         16438                   0    ...                                0
         24512                   1    ...                                0

                native-country_ Scotland  native-country_ South  \
         13739                         0                      0
         28826                         0                      0
         20154                         0                      0
         16438                         0                      0
         24512                         0                      0

                native-country_ Taiwan  native-country_ Thailand  \
         13739                       0                         0
         28826                       0                         0
         20154                       0                         0
         16438                       0                         0
         24512                       0                         0

                native-country_ Trinadad&Tobago  native-country_ United-States  \
13739                                0                              1
28826                                0                              1
20154                                0                              1
16438                                0                              1
24512                                0                              0

       native-country_ Vietnam  native-country_ Yugoslavia  predicted
13739                        0                           0          1
28826                        0                           0          1
20154                        0                           0          1
16438                        0                           0          1
24512                        0                           0          1

[5 rows x 107 columns]
```

```
In [90]: import seaborn as sns
         sns.set(color_codes=True)
```

```
In [91]: sns.stripplot(x="predicted", y="capital-loss", data=test_x,jitter=True)
```

```
Out[91]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7b0ca92780>
```


capital-loss
predicted