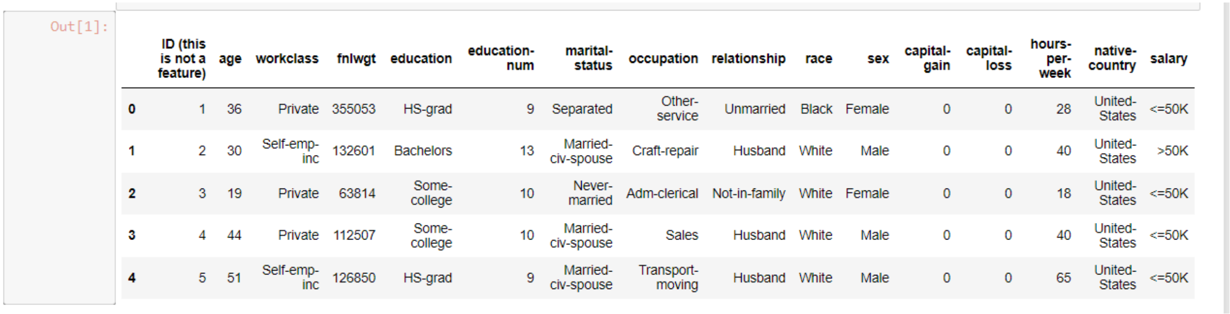
AIT 636: Final Project

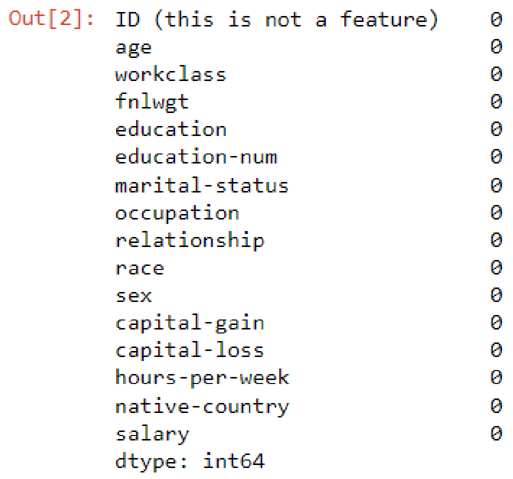
Name: Shubham Khaladkar



Out[1]:

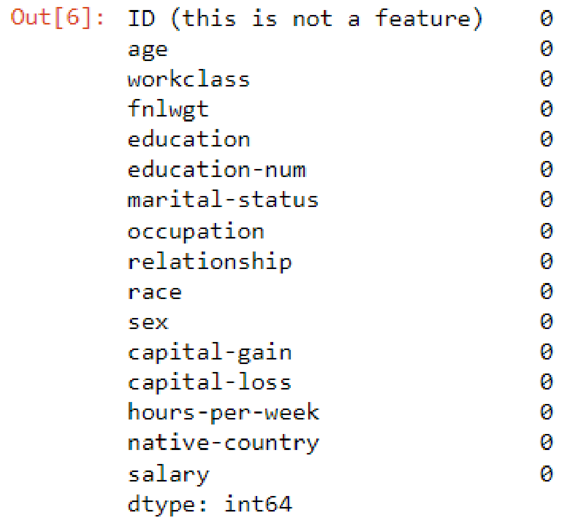
	ID (this is not a feature)	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	salary
0	1	36	Private	355053	HS-grad	9	Separated	Other-service	Unmarried	Black	Female	0	0	28	United-States	<=50K
1	2	30	Self-emp-inc	132601	Bachelors	13	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	40	United-States	>50K
2	3	19	Private	63814	Some-college	10	Never-married	Adm-clerical	Not-in-family	White	Female	0	0	18	United-States	<=50K
3	4	44	Private	112507	Some-college	10	Married-civ-spouse	Sales	Husband	White	Male	0	0	40	United-States	<=50K
4	5	51	Self-emp-inc	126850	HS-grad	9	Married-civ-spouse	Transport-moving	Husband	White	Male	0	0	65	United-States	<=50K

Data Cleaning – For training data (Checking ‘na’ or ‘?’ values)



```
Out[2]: ID (this is not a feature)    0
        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
```

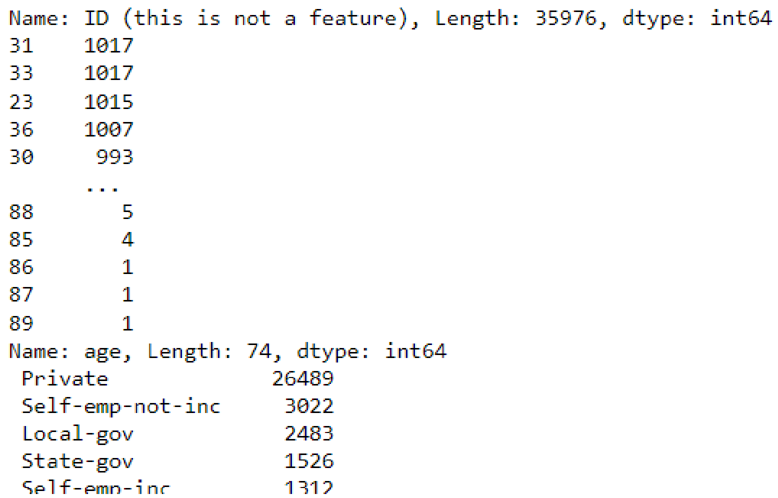
Data Cleaning – For testing data



```
Out[6]: ID (this is not a feature)    0
        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
```

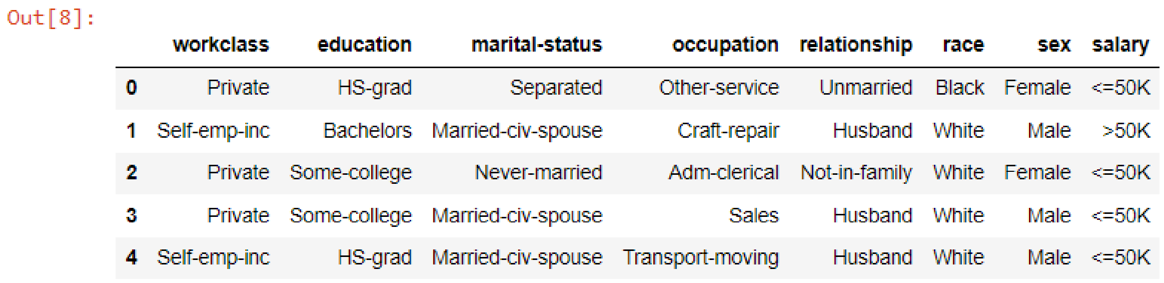
Feature engineering – For training data

Converting categorical data to numerical



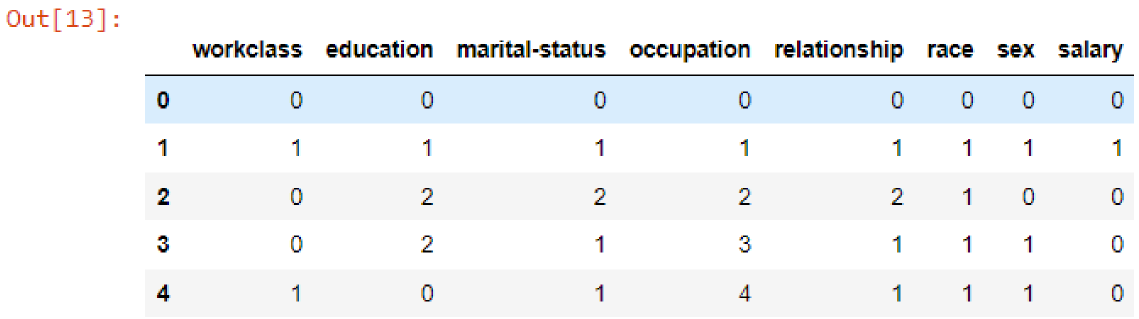
```
Name: ID (this is not a feature), Length: 35976, dtype: int64
31    1017
33    1017
23    1015
36    1007
30     993
      ...
88       5
85       4
86       1
87       1
89       1
Name: age, Length: 74, dtype: int64
 Private             26489
 Self-emp-not-inc     3022
 Local-gov            2483
 State-gov            1526
 Self-emp-inc         1312
```

Removing noise data (irrelevant data).



Out[8]:

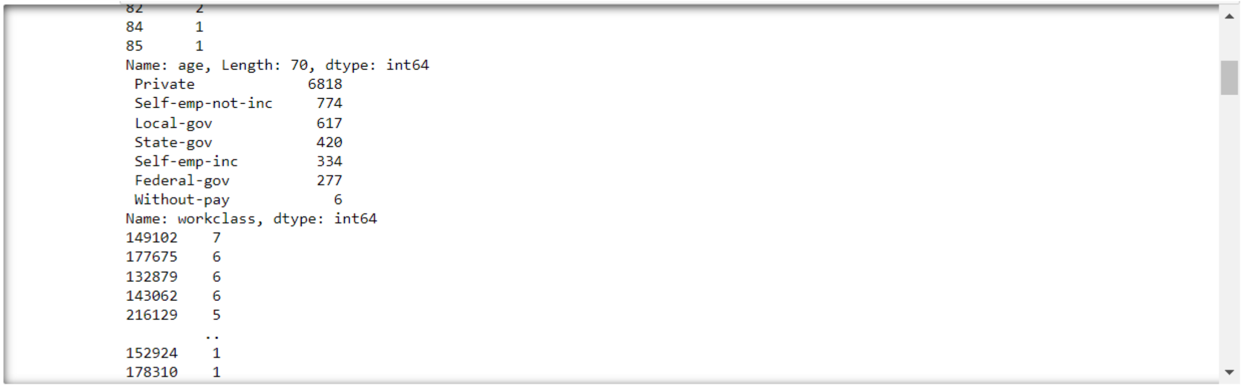
	workclass	education	marital-status	occupation	relationship	race	sex	salary
0	Private	HS-grad	Separated	Other-service	Unmarried	Black	Female	<=50K
1	Self-emp-inc	Bachelors	Married-civ-spouse	Craft-repair	Husband	White	Male	>50K
2	Private	Some-college	Never-married	Adm-clerical	Not-in-family	White	Female	<=50K
3	Private	Some-college	Married-civ-spouse	Sales	Husband	White	Male	<=50K
4	Self-emp-inc	HS-grad	Married-civ-spouse	Transport-moving	Husband	White	Male	<=50K



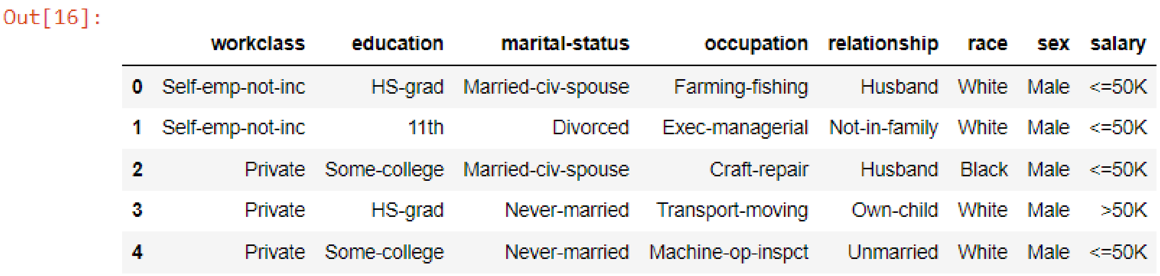
Out[13]:

	workclass	education	marital-status	occupation	relationship	race	sex	salary
0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1
2	0	2	2	2	2	1	0	0
3	0	2	1	3	1	1	1	0
4	1	0	1	4	1	1	1	0

Feature engineering – For training data

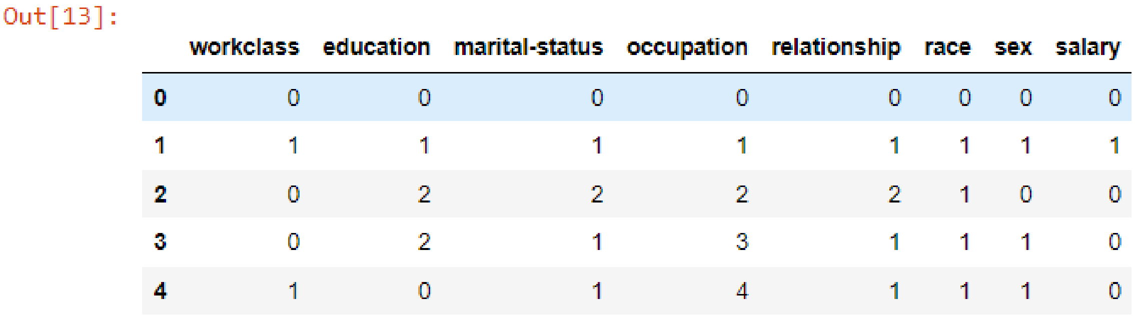


```
82      2
84      1
85      1
Name: age, Length: 70, dtype: int64
 Private             6818
 Self-emp-not-inc     774
 Local-gov            617
 State-gov            420
 Self-emp-inc         334
 Federal-gov          277
 Without-pay            6
Name: workclass, dtype: int64
149102    7
177675    6
132879    6
143062    6
216129    5
         ..
152924    1
178310    1
```



Out[16]:

	workclass	education	marital-status	occupation	relationship	race	sex	salary
0	Self-emp-not-inc	HS-grad	Married-civ-spouse	Farming-fishing	Husband	White	Male	<=50K
1	Self-emp-not-inc	11th	Divorced	Exec-managerial	Not-in-family	White	Male	<=50K
2	Private	Some-college	Married-civ-spouse	Craft-repair	Husband	Black	Male	<=50K
3	Private	HS-grad	Never-married	Transport-moving	Own-child	White	Male	>50K
4	Private	Some-college	Never-married	Machine-op-inspct	Unmarried	White	Male	<=50K

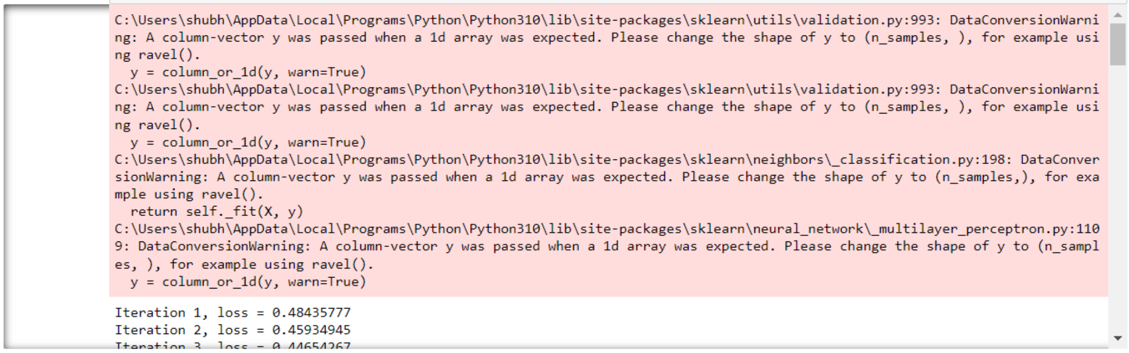


Out[13]:

	workclass	education	marital-status	occupation	relationship	race	sex	salary
0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1
2	0	2	2	2	2	1	0	0
3	0	2	1	3	1	1	1	0
4	1	0	1	4	1	1	1	0

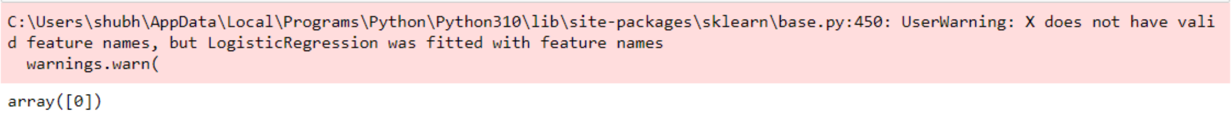
Classification Models

Training model using training data and predicting the salary on test data.



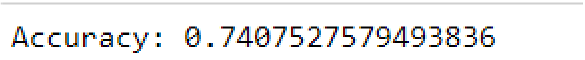
```
C:\Users\shubh\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\utils\validation.py:993: DataConversionWarni
ng: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\shubh\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\utils\validation.py:993: DataConversionWarni
ng: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  y = column_or_1d(y, warn=True)
C:\Users\shubh\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\neighbors\_classification.py:198: DataConver
sionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for exa
mple using ravel().
  return self._fit(X, y)
C:\Users\shubh\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\neural_network\_multilayer_perceptron.py:110
9: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sampl
es, ), for example using ravel().
  y = column_or_1d(y, warn=True)

Iteration 1, loss = 0.48435777
Iteration 2, loss = 0.45934945
Iteration 3, loss = 0.44654267
```

Predicting the label output for the variation [0,0,2,4,3,1,1].



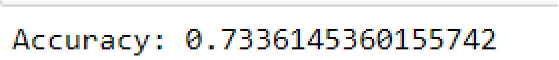
```
C:\Users\shubh\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have vali
d feature names, but LogisticRegression was fitted with feature names
  warnings.warn(

array([0])
```

Logistic Regression



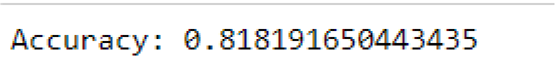
```
Accuracy: 0.7407527579493836
```

Perceptron

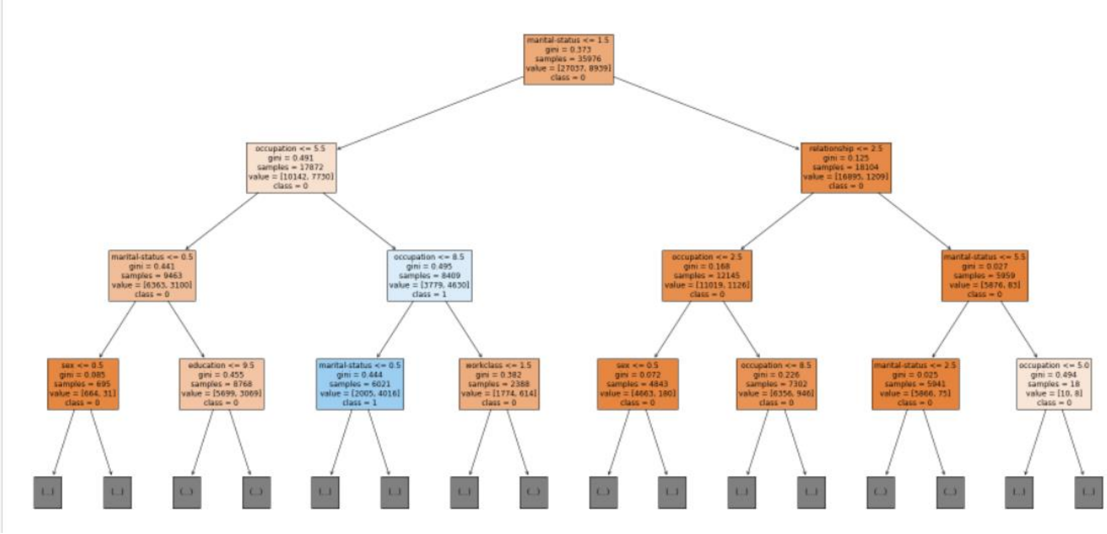


```
Accuracy: 0.7336145360155742
```

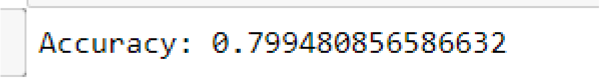
Decision Tree Classifier



```
Accuracy: 0.818191650443435
```

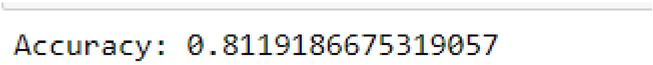


KNN Classifier



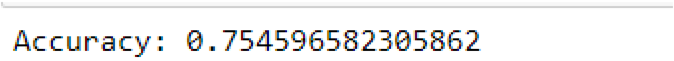
```
Accuracy: 0.799480856586632
```

MLP without PCA



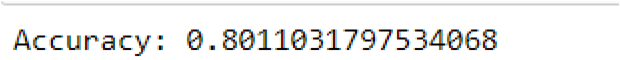
```
Accuracy: 0.8119186675319057
```

Linear SVC



```
Accuracy: 0.754596582305862
```

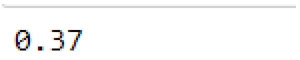
Non-linear SVC



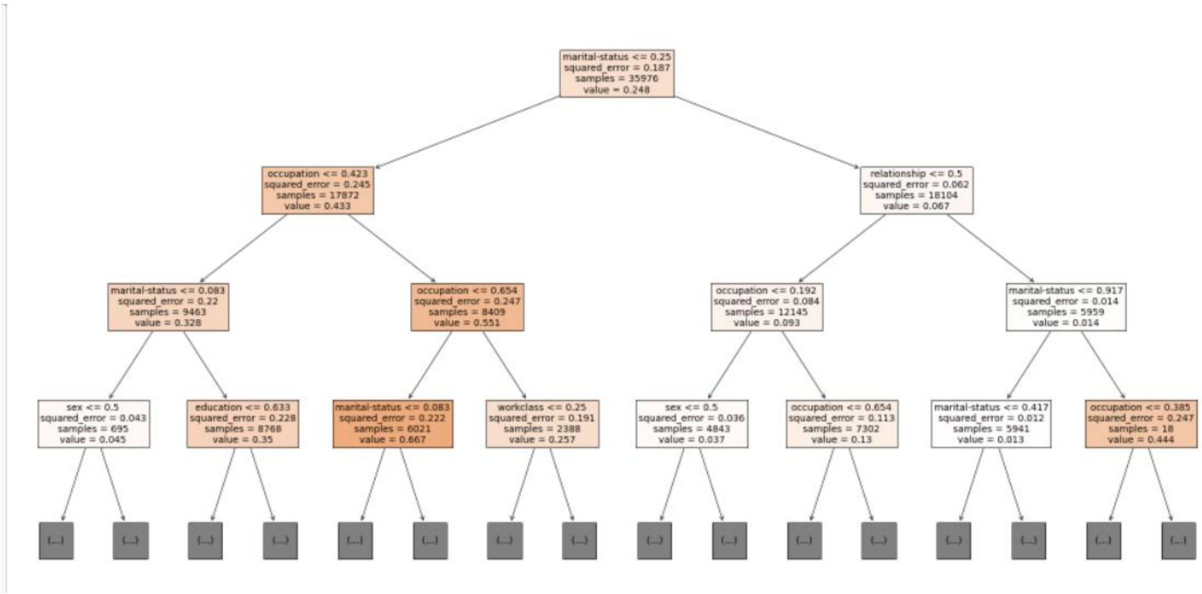
```
Accuracy: 0.8011031797534068
```

Regression Models

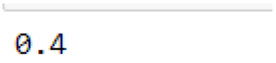
Decision Tree Regressor: Root Mean Square Error (RMSE)



```
0.37
```

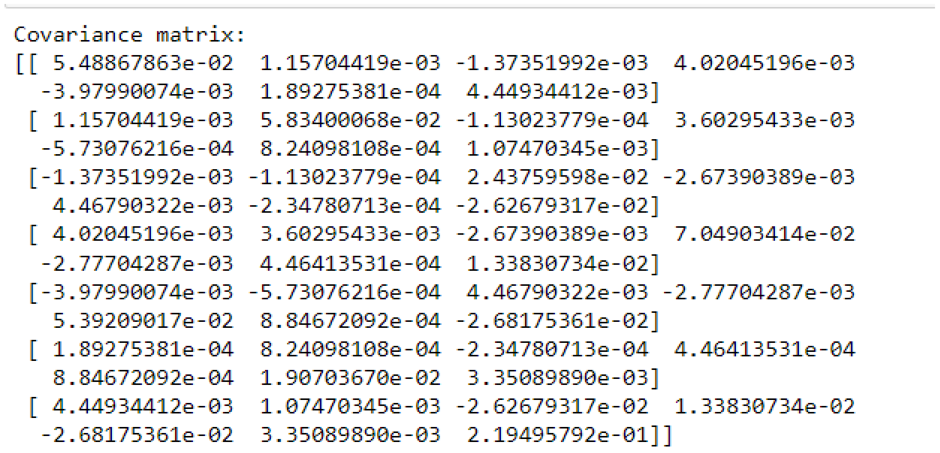


KNN Regressor: Root Mean Square Error (RMSE)

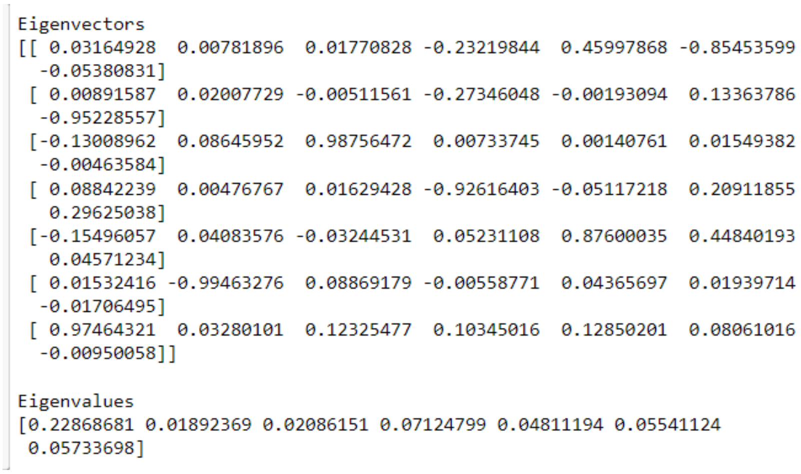


```
0.4
```

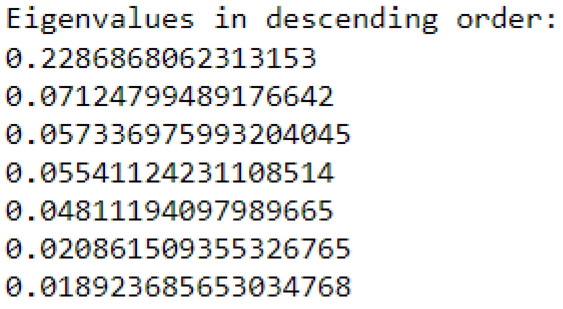
MLP classification model (using PCA)



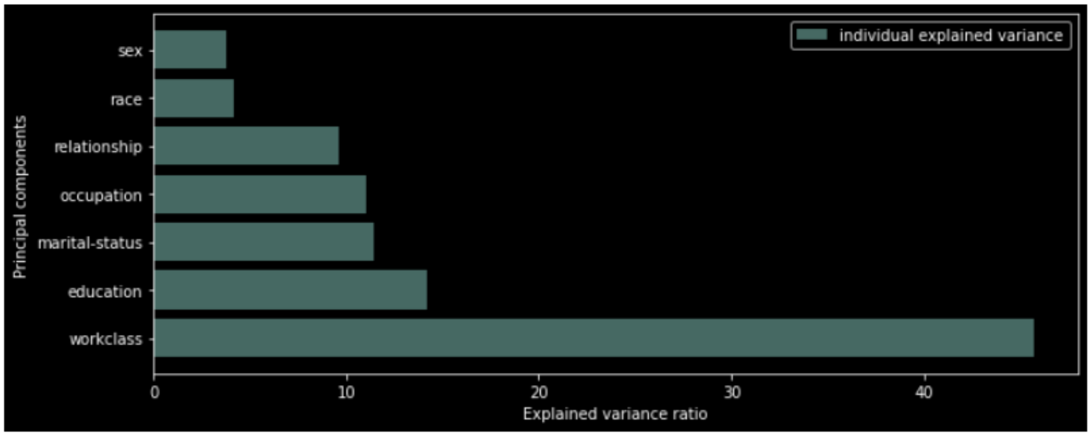
```
Covariance matrix:
[[ 5.48867863e-02  1.15704419e-03 -1.37351992e-03  4.02045196e-03
  -3.97990074e-03  1.89275381e-04  4.44934412e-03]
 [ 1.15704419e-03  5.83400068e-02 -1.13023779e-04  3.60295433e-03
  -5.73076216e-04  8.24098108e-04  1.07470345e-03]
 [-1.37351992e-03 -1.13023779e-04  2.43759598e-02 -2.67390389e-03
   4.46790322e-03 -2.34780713e-04 -2.62679317e-02]
 [ 4.02045196e-03  3.60295433e-03 -2.67390389e-03  7.04903414e-02
  -2.77704287e-03  4.46413531e-04  1.33830734e-02]
 [-3.97990074e-03 -5.73076216e-04  4.46790322e-03 -2.77704287e-03
   5.39209017e-02  8.84672092e-04 -2.68175361e-02]
 [ 1.89275381e-04  8.24098108e-04 -2.34780713e-04  4.46413531e-04
   8.84672092e-04  1.90703670e-02  3.35089890e-03]
 [ 4.44934412e-03  1.07470345e-03 -2.62679317e-02  1.33830734e-02
  -2.68175361e-02  3.35089890e-03  2.19495792e-01]]
```



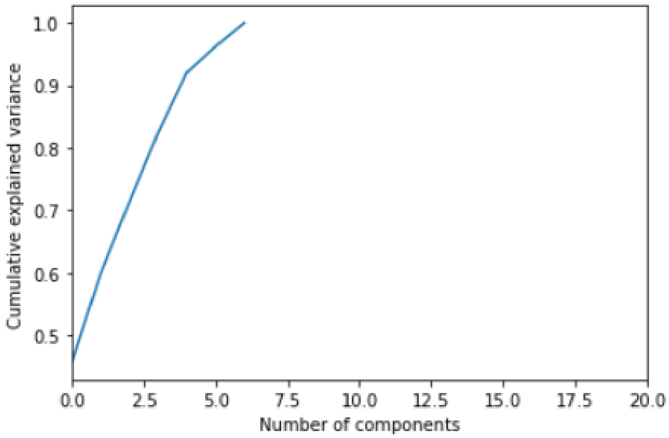
```
Eigenvectors
[[ 0.03164928  0.00781896  0.01770828 -0.23219844  0.45997868 -0.85453599
  -0.05380831]
 [ 0.00891587  0.02007729 -0.00511561 -0.27346048 -0.00193094  0.13363786
  -0.95228557]
 [-0.13008962  0.08645952  0.98756472  0.00733745  0.00140761  0.01549382
  -0.00463584]
 [ 0.08842239  0.00476767  0.01629428 -0.92616403 -0.05117218  0.20911855
   0.29625038]
 [-0.15496057  0.04083576 -0.03244531  0.05231108  0.87600035  0.44840193
   0.04571234]
 [ 0.01532416 -0.99463276  0.08869179 -0.00558771  0.04365697  0.01939714
  -0.01706495]
 [ 0.97464321  0.03280101  0.12325477  0.10345016  0.12850201  0.08061016
  -0.00950058]]

Eigenvalues
[0.22868681 0.01892369 0.02086151 0.07124799 0.04811194 0.05541124
 0.05733698]
```



```
Eigenvalues in descending order:
0.2286868062313153
0.07124799489176642
0.057336975993204045
0.05541124231108514
0.04811194097989665
0.020861509355326765
0.018923685653034768
```



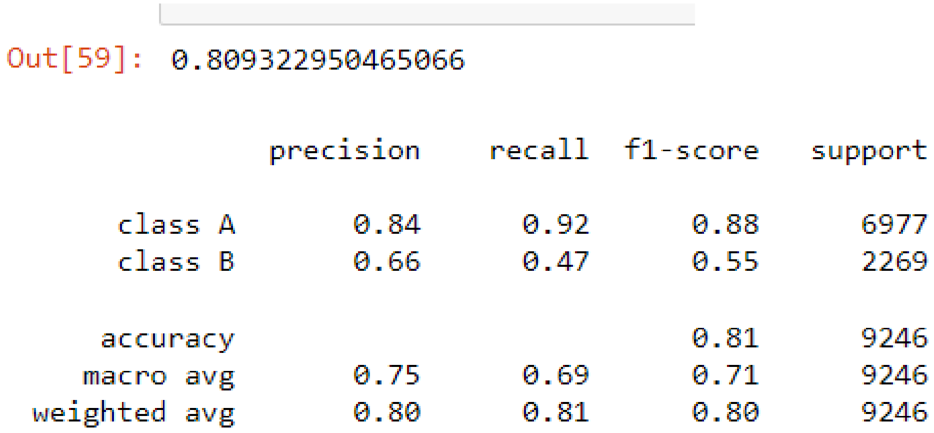
```
Text(0, 0.5, 'Cumulative explained variance')
```



```
Iteration 63, loss = 0.40867722
Iteration 64, loss = 0.40867126
Iteration 65, loss = 0.40718512
Iteration 66, loss = 0.40850798
Iteration 67, loss = 0.40817471
Iteration 68, loss = 0.40755855
Iteration 69, loss = 0.40830843
Iteration 70, loss = 0.40817951
Iteration 71, loss = 0.40944393
Iteration 72, loss = 0.40867412
Iteration 73, loss = 0.40816762
Iteration 74, loss = 0.40892878
Iteration 75, loss = 0.40793259
Iteration 76, loss = 0.40823860
Training loss did not improve more than tol=0.000100 for 10 consecutive epochs. Stopping.
```

```
: MLPClassifier(hidden_layer_sizes=(6, 5), learning_rate_init=0.01,
                random_state=5, verbose=True)
```

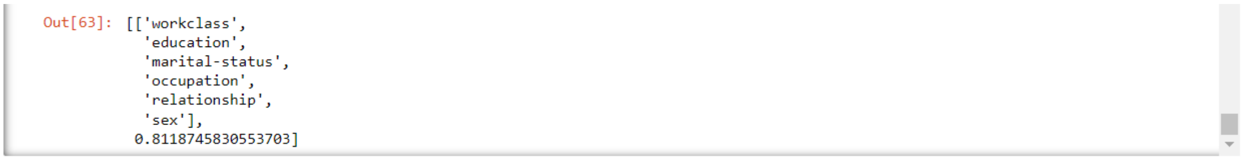
```
array([0.22156259, 0.15193462, 0.14549537, 0.13823179, 0.13287955,
       0.12305259])
```



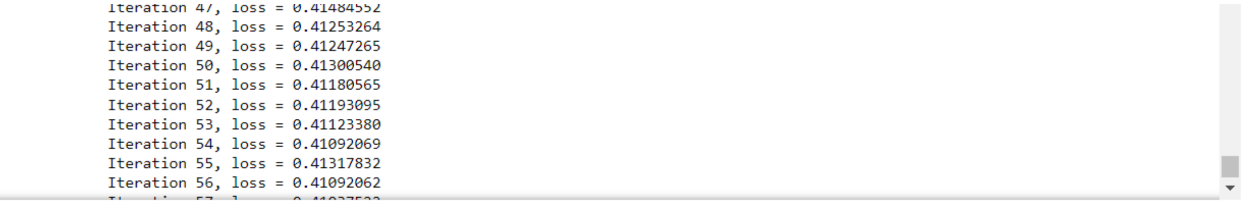
```
Out[59]: 0.809322950465066

              precision    recall  f1-score   support

     class A       0.84      0.92      0.88      6977
     class B       0.66      0.47      0.55      2269

    accuracy                           0.81      9246
   macro avg       0.75      0.69      0.71      9246
weighted avg       0.80      0.81      0.80      9246
```

MLP: Find best subset selection

For training data

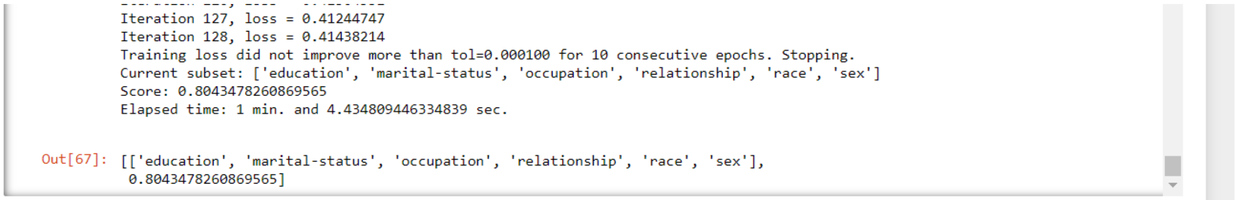


```
Out[63]: [['workclass',
           'education',
           'marital-status',
           'occupation',
           'relationship',
           'sex'],
          0.8118745830553703]
```

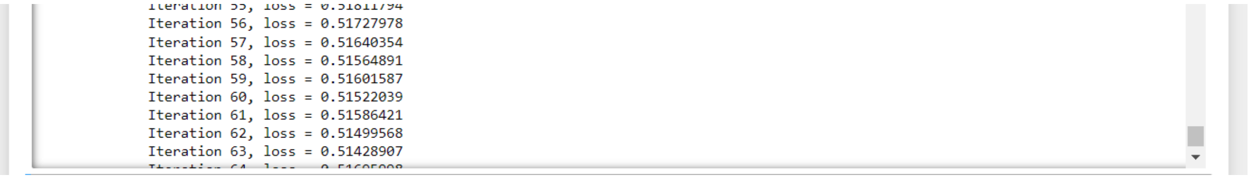


```
Iteration 47, loss = 0.41484552
Iteration 48, loss = 0.41253264
Iteration 49, loss = 0.41247265
Iteration 50, loss = 0.41300540
Iteration 51, loss = 0.41180565
Iteration 52, loss = 0.41193095
Iteration 53, loss = 0.41123380
Iteration 54, loss = 0.41092069
Iteration 55, loss = 0.41317832
Iteration 56, loss = 0.41092062
```

For testing data



```
Iteration 127, loss = 0.41244747
Iteration 128, loss = 0.41438214
Training loss did not improve more than tol=0.000100 for 10 consecutive epochs. Stopping.
Current subset: ['education', 'marital-status', 'occupation', 'relationship', 'race', 'sex']
Score: 0.8043478260869565
Elapsed time: 1 min. and 4.434809446334839 sec.


Out[67]: [['education', 'marital-status', 'occupation', 'relationship', 'race', 'sex'],
          0.8043478260869565]
```



```
Iteration 55, loss = 0.51811794
Iteration 56, loss = 0.51727978
Iteration 57, loss = 0.51640354
Iteration 58, loss = 0.51564891
Iteration 59, loss = 0.51601587
Iteration 60, loss = 0.51522039
Iteration 61, loss = 0.51586421
Iteration 62, loss = 0.51499568
Iteration 63, loss = 0.51428907
```