Article

Analyzing 'adults.csv' using Support Vector Machine (SVM) Algorithm

Faseeh U Rehman Qureshi

[1] E-mail: faseeh.quraishi.fq@gmail.com
[2] Tel.: +92-306-3551492

Abstract: A data-set named 'adults.csv' has been used to explore the possibility in predicting income level based on the individual's personal information as above or below $50,000 per year. This data-set was drawn from the 1994 United States Census Bureau data and involves individual's information such as qualification, occupation and much more. Still researchers try to research about why some individuals have more income than $50,000 while others have less. With the use of a modern data mining tools and an available data-set we take a look at which factors have impact at individual's income level and which factors do not. Support Vector Machine (SVM) has been implemented to predict the level of an income of an individual.

Keywords: Machine Learning; Classification; Support Vector Machine (SVM); Training; Testing; Kaggle

Citation: . *Reports* **2022**, *1*, 0.

Received:
Accepted:
Published:

1. Introduction

A data-set named 'adults.csv' has been used to explore the possibility in predicting income level based on the individual's personal information as above or below $50,000. The data-set is publicly available on a website called Kaggle.

This data-set was drawn from the 1994 United States Census Bureau data and involves using individual's information such as qualification, occupation and much more to predict whether he or she will earn more or less than $50,000 per year.

Machine learning is the study of computer algorithms that can improve automatically through experience and by the use of data. It is also seen as a part of artificial intelligence. The field of machine learning has allowed analysts to uncover insights from historical data and past events. Machine Learning is divided into three types:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

Currently, machine learning has been used in multiple fields and industries. For example, medical diagnosis, image processing, prediction, classification, learning association, regression.

This data-set has been studied and analyzed using Support Vector Machine (SVM). Support Vector Machine (SVM) is implemented using 'sklearn' library on Python. The prime objective is to analyze adults.csv to determine a correlation between different features the survival of passengers and characteristics of the passengers using machine learning algorithm - Support Vector Machine (SVM).

2. Implementation

2.1. Introduction to SVM

"Support Vector Machine" (SVM) is a machine learning algorithm that can be used for both classification and regression. SVMs are applied to find the line in two dimensions or the hyperplane in more than two dimensions in order to separate space into different classes.

We use SVM for classifying of genes, patients on the basis of genes and other biological problems. Protein fold and remote homology detection and handwriting recognition also use SVMs widely. The data collected is raw-data which is very likely to contain mistakes, unnecessary features, missing values and corrupt values. Before drawing any conclusions from the data we need to do some data pre-processing which involves feature engineering. Feature engineering process attempts to create additional relevant features from existing raw features in the data and to increase the predictive power of learning algorithms.

2.2. Data Pre-Processing

Our approach to solve the problem starts with collecting the raw data need to solve the problem and import the data-set into the working environment and do data pre-processing which includes data exploration and feature engineering then explore the data and prepare a model for performing analysis using machine learning algorithms and evaluate the model and re-iterate till we get satisfactory model performance then compare the results within the algorithm and select a model which gives a more accurate results. Firstly, Different libraries used throughout the project are imported.

```
import numpy as np
import pandas as pd
from sklearn import svm,metrics
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import seaborn as sns
```

Then, website. The data provided is in the form of a CSV (Comma Separated Value) file. The data consists of 32561 entries(record) in the data-set which is individual's sample with their associated features. For each individual there are 14 features, like Age, Type of Job, Qualification, Experience and many more shown in below.

```
missing_values = [" ?",np.nan]
adults_data= pd.read_csv('adults.csv', na_values = missing_values)
adults_data.head()
```

	Age	Type of Job	Qualification	Experience	Marital Status	Occupation	Unnamed: 6	Race	Gender	Unnamed: 9	Unnamed: 10	Unnamed: 11	Location	Salary
0	39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	40	Private	Assoc-voc	11	Married-civ-spouse	Craft-repair	Husband	Asian-Pac-Islander	Male	0	0	40	NaN	>50K
2	50	Self-emp-not-inc	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
3	38	Private	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
4	53	Private	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K

To start the exploration or data pre-processing, some information and few plots are made to get an overall idea for each attribute. Some of the plots and information have been shown below.

```
adults_data.shape
```

```
(32561, 14)
```

```
adults_data[['Type of Job','Qualification','Marital Status','Occupation','Unnamed: 6','Race','Gender','Location']].describe()
```

	Type of Job	Qualification	Marital Status	Occupation	Unnamed: 6	Race	Gender	Location
count	30725	32561	32561	30718	32561	32561	32561	31978
unique	8	16	7	14	6	5	2	41
top	Private	HS-grad	Married-civ-spouse	Prof-specialty	Husband	White	Male	United-States
freq	22696	10501	14976	4140	13193	27816	21790	29170

```
adults_data.describe()
```

	Age	Experience	Unnamed: 9	Unnamed: 10	Unnamed: 11
count	32561.000000	32561.000000	32561.000000	32561.000000	32561.000000
mean	38.581647	10.080679	1077.648844	87.303830	40.437456
std	13.640433	2.572720	7385.292085	402.960219	12.347429
min	17.000000	1.000000	0.000000	0.000000	1.000000
25%	28.000000	9.000000	0.000000	0.000000	40.000000
50%	37.000000	10.000000	0.000000	0.000000	40.000000
75%	48.000000	12.000000	0.000000	0.000000	45.000000
max	90.000000	16.000000	99999.000000	4356.000000	99.000000

```
adults_data[['Type of Job']].value_counts()
```

```
Type of Job
 Private             22696
 Self-emp-not-inc     2541
 Local-gov            2093
 State-gov            1298
 Self-emp-inc         1116
 Federal-gov           960
 Without-pay            14
 Never-worked            7
dtype: int64
```

```
adults_data[['Qualification']].value_counts()
```

```
Qualification
 HS-grad         10501
 Some-college     7291
 Bachelors        5355
 Masters          1723
 Assoc-voc        1382
 11th             1175
 Assoc-acdm       1067
 10th              933
 7th-8th           646
 Prof-school       576
 9th               514
 12th              433
 Doctorate         413
 5th-6th           333
 1st-4th           168
 Preschool          51
dtype: int64
```

```
adults_data[['Marital Status']].value_counts()
```

```
Marital Status
 Married-civ-spouse       14976
 Never-married            10683
 Divorced                  4443
 Separated                 1025
 Widowed                    993
 Married-spouse-absent      418
 Married-AF-spouse           23
dtype: int64
```

```
adults_data[['Occupation']].value_counts()
```

```
Occupation
 Prof-specialty       4140
 Craft-repair         4099
 Exec-managerial      4066
 Adm-clerical         3770
 Sales                3650
 Other-service        3295
 Machine-op-inspct    2002
 Transport-moving     1597
 Handlers-cleaners    1370
 Farming-fishing       994
 Tech-support          928
 Protective-serv       649
 Priv-house-serv       149
 Armed-Forces            9
dtype: int64
```

```
adults_data[['Unnamed: 6']].value_counts()
```

```
Unnamed: 6
 Husband           13193
 Not-in-family      8305
 Own-child          5068
 Unmarried          3446
 Wife               1568
 Other-relative      981
dtype: int64
```

```
adults_data[['Race']].value_counts()
```

```
Race
 White                 27816
 Black                  3124
 Asian-Pac-Islander     1039
 Amer-Indian-Eskimo      311
 Other                   271
dtype: int64
```

```
adults_data[['Gender']].value_counts()
```

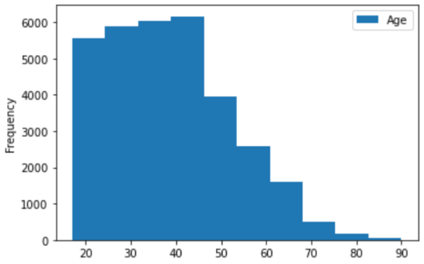
```
Gender
 Male      21790
 Female    10771
dtype: int64
```

```
adults_data[['Location']].value_counts()
```

```
Location
 United-States                 29170
 Mexico                          643
 Philippines                     198
 Germany                         137
 Canada                          121
 Puerto-Rico                     114
 El-Salvador                     106
 India                           100
 Cuba                             95
 England                          90
 Jamaica                          81
 South                            80
 China                            75
 Italy                            73
 Dominican-Republic               70
 Vietnam                          67
 Guatemala                        64
 Japan                            62
 Poland                           60
 Columbia                         59
 Taiwan                           51
 Haiti                            44
 Iran                             43
 Portugal                         37
 Nicaragua                        34
 Peru                             31
 Greece                           29
 France                           29
 Ecuador                          28
 Ireland                          24
 Hong                             20
 Trinadad&Tobago                  19
 Cambodia                         19
 Laos                             18
 Thailand                         18
 Yugoslavia                       16
```

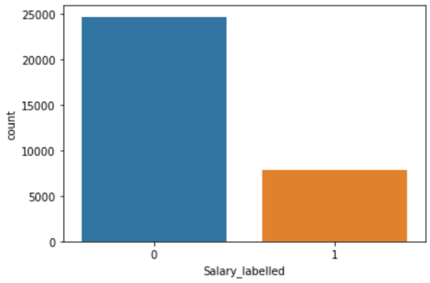
```
reduced_data[['Age']].plot(kind='hist')
#right skewed
```

```
<AxesSubplot:ylabel='Frequency'>
```



```
#reduced_data[['Salary_labelled']].plot(kind='hist')
sns.countplot(x="Salary_labelled", data= reduced_data)
# 0 = <=50k
# 1 = >50k
```

```
<AxesSubplot:xlabel='Salary_labelled', ylabel='count'>
```



After observing whole data some unrelated features were dropped.

```
reduced_data=adults_data.drop(columns=['Unnamed: 6','Unnamed: 9','Unnamed: 10','Unnamed: 11','Marital Status','Race'])
reduced_data.head()
```

	Age	Type of Job	Qualification	Experience	Occupation	Gender	Location	Salary
0	39	State-gov	Bachelors	13	Adm-clerical	Male	United-States	<=50K
1	40	Private	Assoc-voc	11	Craft-repair	Male	NaN	>50K
2	50	Self-emp-not-inc	Bachelors	13	Exec-managerial	Male	United-States	<=50K
3	38	Private	HS-grad	9	Handlers-cleaners	Male	United-States	<=50K
4	53	Private	11th	7	Handlers-cleaners	Male	United-States	<=50K

While observation it has been found that the data-set is not complete. There are various rows for which one or more fields are marked empty (specifically in Type of Job, Occupation and Location features). But the all of these are important features to predict the income level. In order to deal with the missing values forward filling technique is used.

```
reduced_data= reduced_data.fillna(method='ffill') #missing values are forward filled
#reduced_data= reduced_data.dropna()
reduced_data.head()
```

	Age	Type of Job	Qualification	Experience	Occupation	Gender	Location	Salary
0	39	State-gov	Bachelors	13	Adm-clerical	Male	United-States	<=50K
1	40	Private	Assoc-voc	11	Craft-repair	Male	United-States	>50K
2	50	Self-emp-not-inc	Bachelors	13	Exec-managerial	Male	United-States	<=50K
3	38	Private	HS-grad	9	Handlers-cleaners	Male	United-States	<=50K
4	53	Private	11th	7	Handlers-cleaners	Male	United-States	<=50K

```
reduced_data.isnull().sum()
```

```
Age              0
Type of Job      0
Qualification    0
Experience       0
Occupation       0
Gender           0
Location         0
Salary           0
dtype: int64
```

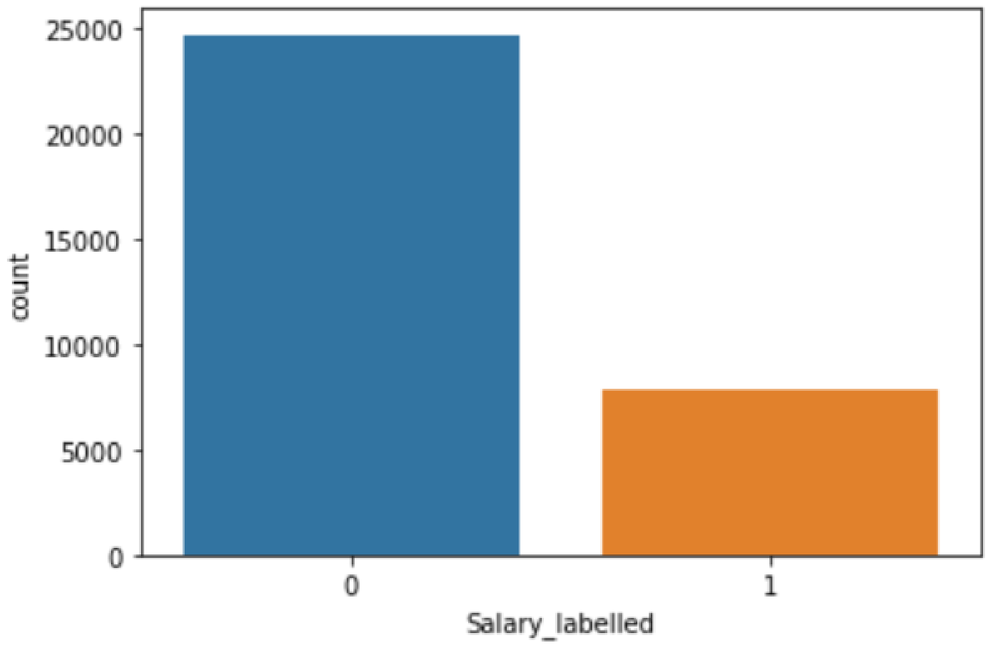
The features like Type of Job, Qualification, Occupation, Gender, Location, and Salary have been label encoded to fit the prediction model in a better manner.

```
labelencoder = LabelEncoder()
reduced_data['TypeOfJob_labelled']=labelencoder.fit_transform(reduced_data['Type of Job'])
reduced_data = reduced_data.drop(columns = ['Type of Job'])
reduced_data['Qualification_labelled']=labelencoder.fit_transform(reduced_data['Qualification'])
reduced_data = reduced_data.drop(columns = ['Qualification'])
reduced_data['Occupation_labelled']=labelencoder.fit_transform(reduced_data['Occupation'])
reduced_data = reduced_data.drop(columns = ['Occupation'])
reduced_data['Gender_labelled']=labelencoder.fit_transform(reduced_data['Gender'])
reduced_data = reduced_data.drop(columns = ['Gender'])
reduced_data['Location_labelled']=labelencoder.fit_transform(reduced_data['Location'])
reduced_data = reduced_data.drop(columns = ['Location'])
reduced_data['Salary_labelled']=labelencoder.fit_transform(reduced_data['Salary'])
reduced_data = reduced_data.drop(columns = ['Salary'])
reduced_data.head()

#label encoding of non-numeric columns
```

	Age	Experience	TypeOfJob_labelled	Qualification_labelled	Occupation_labelled	Gender_labelled	Location_labelled	Salary_labelled
0	39	13	6	9	0	1	38	0
1	40	11	3	8	2	1	38	1
2	50	13	5	9	3	1	38	0
3	38	9	3	11	5	1	38	0
4	53	7	3	1	5	1	38	0

A histogram is generated to determine the number of individuals earning more than \$50,000 vs. number of individuals who earn less than \$50,000.



From the histogram it is clear that the number of individuals who earn more than \$50,000 are more than the number of individuals who earn less than \$50,000.

Feature named 'Salary_labelled' has been selected as output y and dropped from input data-set which is named as X:

```
reduced_data.shape
```

```
(32561, 8)
```

```
X = reduced_data.drop(columns=['Salary_labelled'])
y = reduced_data['Salary_labelled']
print(X.shape, y.shape)
```

```
(32561, 7) (32561,)
```

X and y has been divided into 80% training set and 20% testing set i.e 'X_train', 'X_test', 'y_train' and 'y_test'.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,random_state=109)
```

```
X_train.shape
```

```
(26048, 7)
```

```
X_test.shape
```

```
(6513, 7)
```

The model is then trained:

```
model = svm.SVC(kernel='linear')
model.fit(X_train,y_train)
```

```
SVC(kernel='linear')
```

Lastly prediction is done

```
y_pred = model.predict(X_test)
y_pred
```

```
array([1, 0, 0, ..., 0, 1, 0])
```

3. Results

After training with the SVM algorithm, we have validated our trained algorithm with test data-set and measured the algorithm's performance with confusion matrix for validation. 80 percent of data is used as training data set and 20 percent of data is used as testing data set. It has been found that the accuracy rate and precision of predicting the income level using Support Vector Machine (SVM) algorithm are approximately 0.8050 (80.5%) and 0.6432 (64.32%) respectively.

```
confusion_matrix= confusion_matrix(y_test,y_pred)
TN,FP,FN,TP = confusion_matrix.ravel()
confusion_matrix
```

```
array([[4684,  310],
       [ 960,  559]], dtype=int64)
```

```
Accuracy=(TP+TN)/y_test.shape[0]
print("Accuracy:",Accuracy)
```

```
Accuracy: 0.8050053738676494
```

```
ErrorRate = 1- Accuracy
print('Error Rate:', ErrorRate)
```

```
Error Rate: 0.19499462613235063
```

```
Precision = TP/(TP+FP)
print('Precision:', Precision)
```

```
Precision: 0.6432681242807825
```

```
Recall = TP/(FN+TP)
print('Recall:', Recall)
```

```
Recall: 0.36800526662277816
```

Figure 1. This is a Confusion Matrix. From the confusion matrix, it has been observed that out of 6513 predictions, the prediction model using SVM algorithm has made 5207 i.e. 4648+559 valid predictions.

4. Conclusions

The analysis revealed interesting patterns across individual-level features. Factors such as Experience, Age and Location appeared to have an impact on income level. These conclusions, however, were derived from findings in the given data set. Support Vector Machine (SVM) proved to be the better algorithm for the adults income classification problem. The research also determined the features that are the most significant for the prediction.