Name: Sangeeth Kumar V
Register No: 16BIS0072

# Artificial Intelligence with Python

Lab Task – 02 (L39 & L40)

*Prof Hemprasad Yashwant Patil*

**Question:**

The Census Income Data Set (Income_data.txt) presents a challenge of predicting whether income exceeds $50K/yr based on census data. The data is stored in Income_data.txt file. There are two distinct classes namely '<=50K' and '>50K'. Apply necessary pre-processing steps on this data and state them clearly. Note that there should not be more than 70% data in train. The model has to be built which will accurately classify the test data. Use various classification techniques and indicate the results in tabular form. Perform a benchmarking analysis of the results.

**Solution:**

Sample dataset:

```
39, State-gov, 77516, Bachelors, 13, Never-married, Adm-clerical,
Not-in-family, White, Male, 2174, 0, 40, United-States, <=50K

50, Self-emp-not-inc, 83311, Bachelors, 13, Married-civ-spouse,
Exec-managerial, Husband, White, Male, 0, 0, 13, United-States,
<=50K
```

**SVM and LinearSVM classifier**

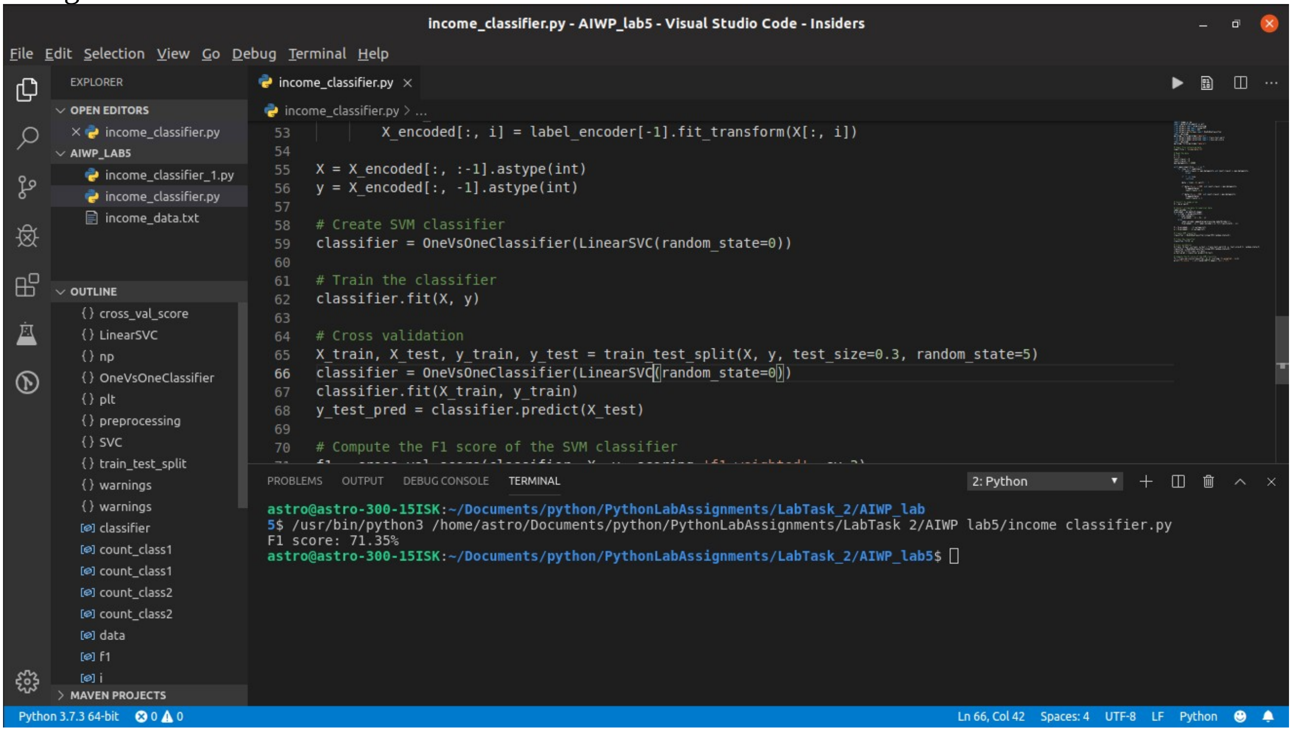
Code:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn.svm import LinearSVC
from sklearn.svm import SVC
from sklearn.multiclass import OneVsOneClassifier
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
import warnings
warnings.filterwarnings("ignore")

# Input file containing data
input_file = 'income_data.txt'

# Read the data
X = []
y = []
count_class1 = 0
count_class2 = 0
max_datapoints = 25000

with open(input_file, 'r') as f:
```

```
for line in f.readlines():
if count_class1 >= max_datapoints and count_class2 >= max_datapoints:
break

if '?' in line:
continue

data = line[:-1].split(', ')

if data[-1] == '<=50K' and count_class1 < max_datapoints:
X.append(data)
count_class1 += 1

if data[-1] == '>50K' and count_class2 < max_datapoints:
X.append(data)
count_class2 += 1

# Convert to numpy array
X = np.array(X)

# Convert string data to numerical data
label_encoder = []
X_encoded = np.empty(X.shape)
for i,item in enumerate(X[0]):
if item.isdigit():
X_encoded[:, i] = X[:, i]
else:
label_encoder.append(preprocessing.LabelEncoder())
X_encoded[:, i] = label_encoder[-1].fit_transform(X[:, i])

X = X_encoded[:, :-1].astype(int)
y = X_encoded[:, -1].astype(int)

# Create SVM classifier
classifier = OneVsOneClassifier(LinearSVC(random_state=0))

# Train the classifier
classifier.fit(X, y)

# Cross validation
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=5)
classifier = OneVsOneClassifier(LinearSVC(random_state=0))
classifier.fit(X_train, y_train)
y_test_pred = classifier.predict(X_test)

# Compute the F1 score of the SVM classifier
f1 = cross_val_score(classifier, X, y, scoring='f1_weighted', cv=3)
print("F1 score: " + str(round(100*f1.mean(), 2)) + "%")
```

## Output:

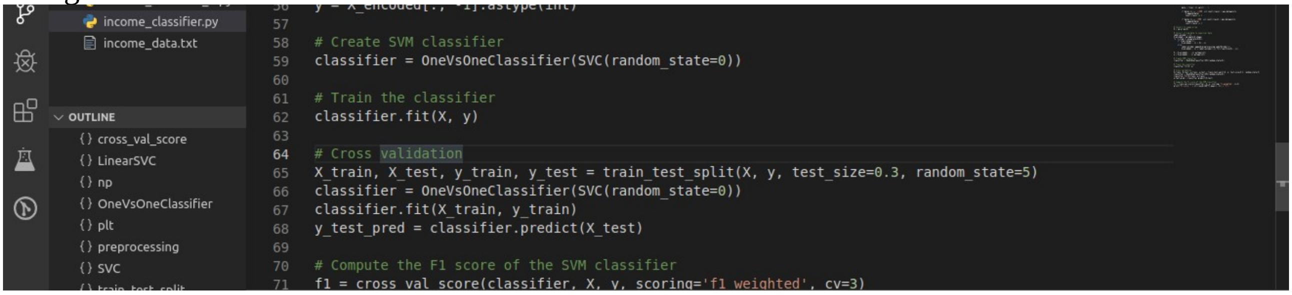
Using LinearSVC



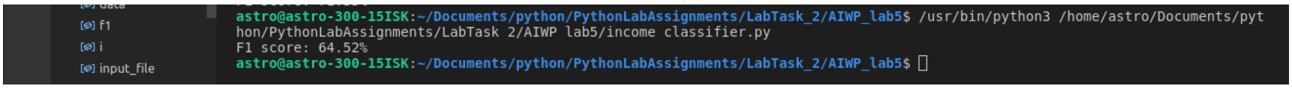
```
        X_encoded[:, i] = label_encoder[-1].fit_transform(X[:, i])

X = X_encoded[:, :-1].astype(int)
y = X_encoded[:, -1].astype(int)

# Create SVM classifier
classifier = OneVsOneClassifier(LinearSVC(random_state=0))

# Train the classifier
classifier.fit(X, y)

# Cross validation
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=5)
classifier = OneVsOneClassifier(LinearSVC(random_state=0))
classifier.fit(X_train, y_train)
y_test_pred = classifier.predict(X_test)

# Compute the F1 score of the SVM classifier
```

```
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_2/AIWP_lab
5$ /usr/bin/python3 /home/astro/Documents/python/PythonLabAssignments/LabTask 2/AIWP lab5/income classifier.py
F1 score: 71.35%
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_2/AIWP_lab5$
```

Using SVC



```
y = X_encoded[:, -1].astype(int)

# Create SVM classifier
classifier = OneVsOneClassifier(SVC(random_state=0))

# Train the classifier
classifier.fit(X, y)

# Cross validation
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=5)
classifier = OneVsOneClassifier(SVC(random_state=0))
classifier.fit(X_train, y_train)
y_test_pred = classifier.predict(X_test)

# Compute the F1 score of the SVM classifier
f1 = cross_val_score(classifier, X, y, scoring='f1_weighted', cv=3)
```



```
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_2/AIWP_lab5$ /usr/bin/python3 /home/astro/Documents/pyt
hon/PythonLabAssignments/LabTask 2/AIWP lab5/income classifier.py
F1 score: 64.52%
astro@astro-300-15ISK:~/Documents/python/PythonLabAssignments/LabTask_2/AIWP_lab5$
```

## Comparison:

F1 Score

| | |
|---|---|
| LinearSVC: | 71.35% |
| SVC: | 64.52% |

-----------------------------------------------------------------------------------------------------