Sri Lanka Institute of Information Technology

Predict the Income of a Person Based On Different Factors

Machine Learning Mini Project

Assignment

K.T. Ramasinghe

IT 15070418

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Problem Statement

In this scenario we predict the income of person by using their personal information. It is more helpful for the employees and non-employees to get an idea about their income by their information. Using this they can get value for themselves without asking someone or searching. Nowadays everyone looking for a job which they can get paid more so they have no idea about how actually can earn before joining any company.

Income prediction is more valuable for employees who currently earn money as well as students who willing to be hired. So they can analyze the requirements and be prepared or they can change their path for the expected income. When predict the income we have to consider many aspects.

In this assignment I had use adults data set of different region person dataset. Then we can predict the income of different region person more accurately. This data set income divided into 2 classes so then we can check weather ones income is increase 50K (>50K) or less than or equal to 50K (<=50K) per year.

2 Methodology

To do this assignment firstly we need to select dataset. It is challenging task because there are lot of data sets in the web and had to go through deferent data sets and identify the algorithm to use on that each datasets. I selected adult data set and we can predict the income. After I selected the data set I had to select the algorithm that we use to predict the income of a person. I go through different algorithms and selected the decision tree algorithm to predict the salary using the dataset.

2.1 Data Collection

2.1.1 Data Set

The dataset is taken from Kaggle.com to do this machine learning assignment. This data set contain the person income is higher or less than $50K/year based on their different attributes. This dataset has 32 561 observation and 15 features of each person information.

Source of Dataset	https://www.kaggle.com/uciml/adult-census-income/downloads/adult-census-income.zip/3. [1]
Number of instances	32561
Number of attributes	15
Data type	Multivariate
Attribute type	Integer / Factor
Related Task	Classification

Table 2. 1 – Characteristics of dataset

2.1.2 Description of attributes

In the data set 15 features are recorded for every person. All the details about the features are listed in below table [2].

Attribute	Values	Type	Description
age	Min : 17 Max : 90	Numeric	Age of the person
workclass	Federal-gov, Local-gov, Never-worked, Private, Self-emp-inc, Self-emp-not-inc, State-gov, Without-pay	Categorical	Person class of work
fnlwgt	Min : 12285 Max : 1484705	Numeric	Final weight of how much of the population it represents
education	10th, 11th, 12th, 1st-4th, 5th-6th, 7th-8th, 9th, Assoc-acdm, Assoc-voc, Bachelors, Doctorate, HS-grad, Masters, Preschool, Prof-school, Some-college	Categorical	Education level of person

material_status	Divorced, Married-AF-spouse, Married-civ-spouse, Married-spouse-absent, Never-married, Separated, Widowed	Categorical	Material status of person
occupation	Adm-clerical, Armed-Forces, Craft-repair, Exec-managerial, Farming-fishing, Handlers-cleaners, Machine-op-inspct, Other-service, Priv-house-serv, Prof-specialty, Protective-serv, Sales, Tech-support, Transport-moving	Categorical	Occupation of person
relationship	Husband, Not-in-family, Other-relative, Own-child, Unmarried, Wife	Categorical	Type of relationship
race	Amer-Indian-Eskimo, Asian-Pac-Islander, Black, Other, White	Categorical	Race of the person
sex	Female, Male		Sex of the person
capital_gain	Min: 0 Max: 99999	Numeric	Capital gain obtained
capital_loss	Min: 0 Max: 4356	Numeric	Capital lost
hour_per_week	Min: 1 Max: 99	Numeric	Average number of hour working per week
native_country	Cambodia, Canada, China, Columbia, Cuba, Dominican-Republic, Ecuador, El-Salvador, England, France, Germany, Greece, Guatemala, Haiti, Holand-Netherlands, Honduras, Hong, Hungary, India, Iran, Ireland, Italy, Jamaica, Japan, Laos, Mexico, Nicaragua, Outlying-US(Guam-USVI-etc), Peru, Philippines, Poland, Portugal, Puerto-Rico, Scotland, South, Taiwan, Thailand, Trinadad&Tobago, United-States, Vietnam, Yugoslavia	Categorical	Country of origin
income	<=50K, >50K	Categorical	Income level of the person

Table 2. 2 – Description of attribute

2.2 Decision Tree Algorithm

Decision tree is a way of making decision like we made decisions in day to day life. [3] This algorithm is a supervised learning algorithm that use to solve classification and regression problem to make decisions on previously trained values. In the tree internal node represent a "test" on an attribute, each branch represent the outcome of the test and each leaf node represent a class label. The paths from root to leaf represent classification rules.

Decision trees split data into different branches based on certain conditions. In the binary tree there are several key components.

- **Root node** – Represent the entire dataset.
- **Leaf node** – Hold a class label
- **Branch** – Is an outcome of the test
- **Internal/Non-leaf node**–Test on an attribute or classification rule

There are several ways to find the best attribute to be split data. In a decision tree we get best attribute as the root node.

- Entropy

Entropy is the degree of randomness of elements or measure of impurity. This help to split our dataset correct way. Mathematically entropy can be represent as:

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

If the data set is order we get less entropy otherwise we get high entropy. Entropy is 0 if all sample of a node belong to the same class and entropy maximal when we have a uniform class distribution.

3 Implementation

Use python for implement the algorithm. To do the classification had to use many python libraries for different tasks.

3.1 Data Loading and Cleaning

To load the data set used pandas library in python. In the library used read_csv function to load the dataset and store the data frame in ‘df’ variable.

```
# read data file and put data into object
df = pd.read_csv('adult_dataset_new.csv')
```

```
# visualize data
df.head()
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female	0	4356	40	Unite
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	Unite
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female	0	4356	40	Unite
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	Unite
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	Unite

Figure 3. 1 – Read data and visualize sample data set

After loaded the dataset can see some data features are missing those missing features represent by using ‘?’ mark. So we have to filter those missing data otherwise our final result will be affected by those missing values.

```
# remove missng values workclass, occupation and native country columns
df = df[df['workclass'] != '?']
df = df[df['occupation'] != '?']
df = df[df['native.country'] != '?']
df.head()
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.c
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female	0	4356	18	Unitec
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female	0	3900	40	Unitec
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female	0	3900	40	Unitec
5	34	Private	216864	HS-grad	9	Divorced	Other-service	Unmarried	White	Female	0	3770	45	Unitec
6	38	Private	150601	10th	6	Separated	Adm-clerical	Unmarried	White	Male	0	3770	40	Unitec

Figure 3. 2 - After cleaned null values of the data set

3.2 Data Preparation

Now we have the filtered data set which don't have the null values for the features. After that we have to select categorical features and apply labels to that categorical data. To remove the categorical data from the data frame (df) variable and concatenate the labeled data to with non-categorical data.

```
# select all categorical variables
df_categorical = df.select_dtypes(include=['object'])
df_categorical.head()
```

	workclass	education	marital.status	occupation	relationship	race	sex	native.country	income
1	Private	HS-grad	Widowed	Exec-managerial	Not-in-family	White	Female	United-States	<=50K
3	Private	7th-8th	Divorced	Machine-op-inspct	Unmarried	White	Female	United-States	<=50K
4	Private	Some-college	Separated	Prof-specialty	Own-child	White	Female	United-States	<=50K
5	Private	HS-grad	Divorced	Other-service	Unmarried	White	Female	United-States	<=50K
6	Private	10th	Separated	Adm-clerical	Unmarried	White	Male	United-States	<=50K

```
# apply label endcorder to df_categorical variable
le = preprocessing.LabelEncoder()
df_categorical = df_categorical.apply(le.fit_transform)
df_categorical.head()
```

	workclass	education	marital.status	occupation	relationship	race	sex	native.country	income
1	2	11	6	3	1	4	0	38	0
3	2	5	0	6	4	4	0	38	0
4	2	15	5	9	3	4	0	38	0
5	2	11	0	7	4	4	0	38	0
6	2	0	5	0	4	4	1	38	0

```
# concat df_categorical with original df
df = df.drop(df_categorical.columns, axis=1)
df = pd.concat([df, df_categorical], axis=1)
df.head()
```

	age	fnlwgt	education.num	capital.gain	capital.loss	hours.per.week	workclass	education	marital.status	occupation	relationship	race	sex	native.cour
1	82	132870	9	0	4356	18	2	11	6	3	1	4	0	
3	54	140359	4	0	3900	40	2	5	0	6	4	4	0	
4	41	264663	10	0	3900	40	2	15	5	9	3	4	0	
5	34	216864	9	0	3770	45	2	11	0	7	4	4	0	
6	38	150601	6	0	3770	40	2	0	5	0	4	4	1	

Figure 3. 3 – Dataset after select categorical data and convert to numerical data

Now our data set has only numerical values to the features so it's easy to apply algorithm to the numerical values. After we converted categorical values to numerical we build the model and split data into training and test.

3.3 Split data

Before train the algorithm we have to split data into train and test. To do that we remove the income from the data set and assign data to variable X and assign income to the variable y. To split data train and test we use train_test_split function in the Sklearn library. In this function we configure test_size as 0.30 that means 30% of our data set use to test after the train the decision tree.

```
# putting feature variable to X
X = df.drop('income',axis=1)

# Putting response variable to y
y = df['income']

# splitting data into train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state = 99)
```

Figure 3. 4 – Split data to train and test

Now we split data to train the secision tree and test the tree. After we devicded the data we have to fitting / modeling the tree from DecisionTreeClassifier which is a function in the sklearn library in python. We have to set Hyperparameters to draw the tree.

```
# Fitting the decision tree with hyperparameters
# max_depth which is 5 so that we can plot and read the tree.
dt_default = DecisionTreeClassifier(criterion="entropy",
                                    random_state=100,
                                    max_depth=5,
                                    min_samples_leaf=5)
dt_default.fit(X_train, y_train)
```

Figure 3. 5 – Fitting the tree with hyper parameter

4 Results

4.1 Classification Report

The classification report will visualize the precision, recall, F1, and support score for the model. This report show the representation of the main classification matrices on a per class basis.

Classification Report :

	precision	recall	f1-score	support
0	0.86	0.96	0.90	6867
1	0.78	0.50	0.61	2182
micro avg	0.85	0.85	0.85	9049
macro avg	0.82	0.73	0.76	9049
weighted avg	0.84	0.85	0.83	9049

Figure 4. 1 – Classification Report

- Precision - Accuracy of positive predictions
- Recall - Fraction of positives that were correctly identified
- F1 score - Weighted harmonic mean of precision and recall [4]
- Support - Number of samples that represent for each classes

.

4.2 Confusion Matrix

Confusion matrix is a table used to describe the performance of a classification model. [5] In this scenario we predict the person income based on different features. This matrix is a measurement for machine learning classification problems. This table explain the two classes with different combination of predicted and actual values.

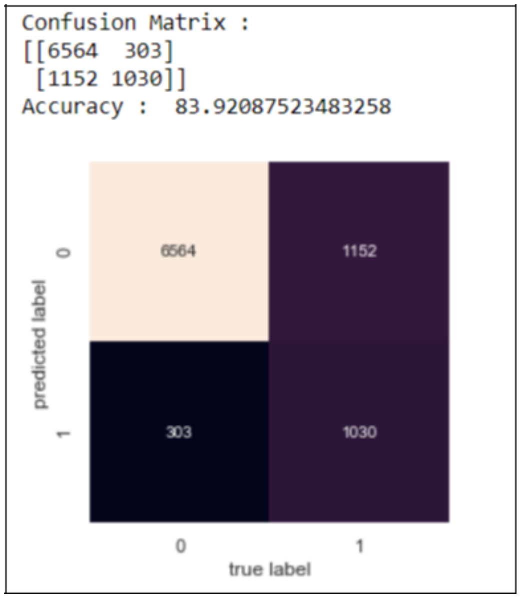


Figure 4. 2 – Confusion matrix

4.3 Decision Tree

In the decision tree we get the root node as the best future. We use entropy for the select best attribute selection. We can use Gini index also for plotting the tree. As the root node we get highest entropy and split data from that feature.

Using graphviz and pydot library in the python we can draw and export the decision tree. So that we can using that tree assume and predict decisions.

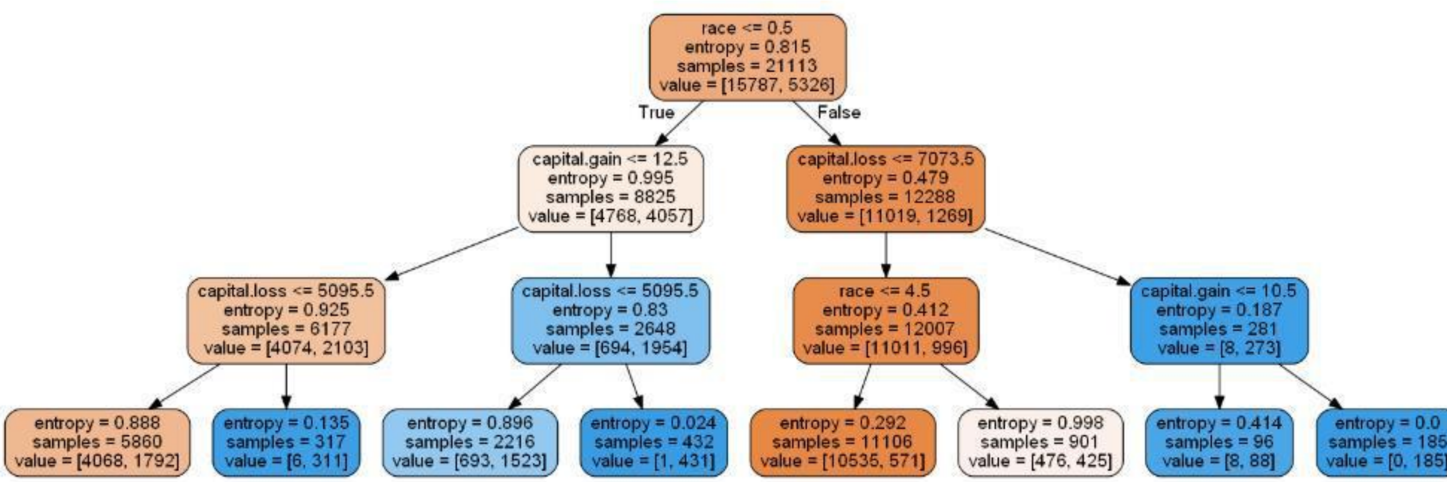


Figure 4. 3 – Sample Decision tree

5 Conclusion

We can conclude that we can predict person salary for 83.93 % accuracy using this algorithm. We split data to train the model and test the model. We get 70% data to train the model and 30% data to test the model.

After we trained the model our algorithm can predict the income of a person for 83% accuracy for their different features. Somehow we could be get wrong prediction of 17% because of the error rate.

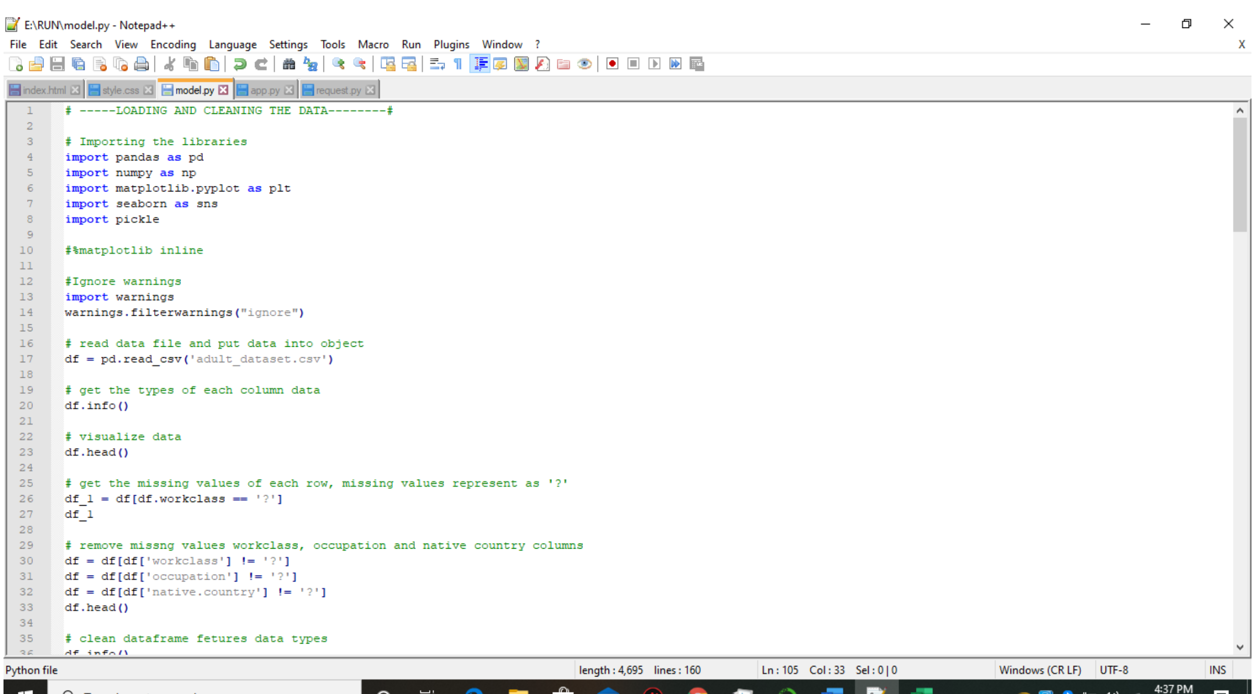
```
print("Accuracy : " , accuracy_score(y_test,y_pred_default) * 100)

Confusion Matrix :
[[6564  303]
 [1151 1031]]
Accuracy :  83.93192617968836
```

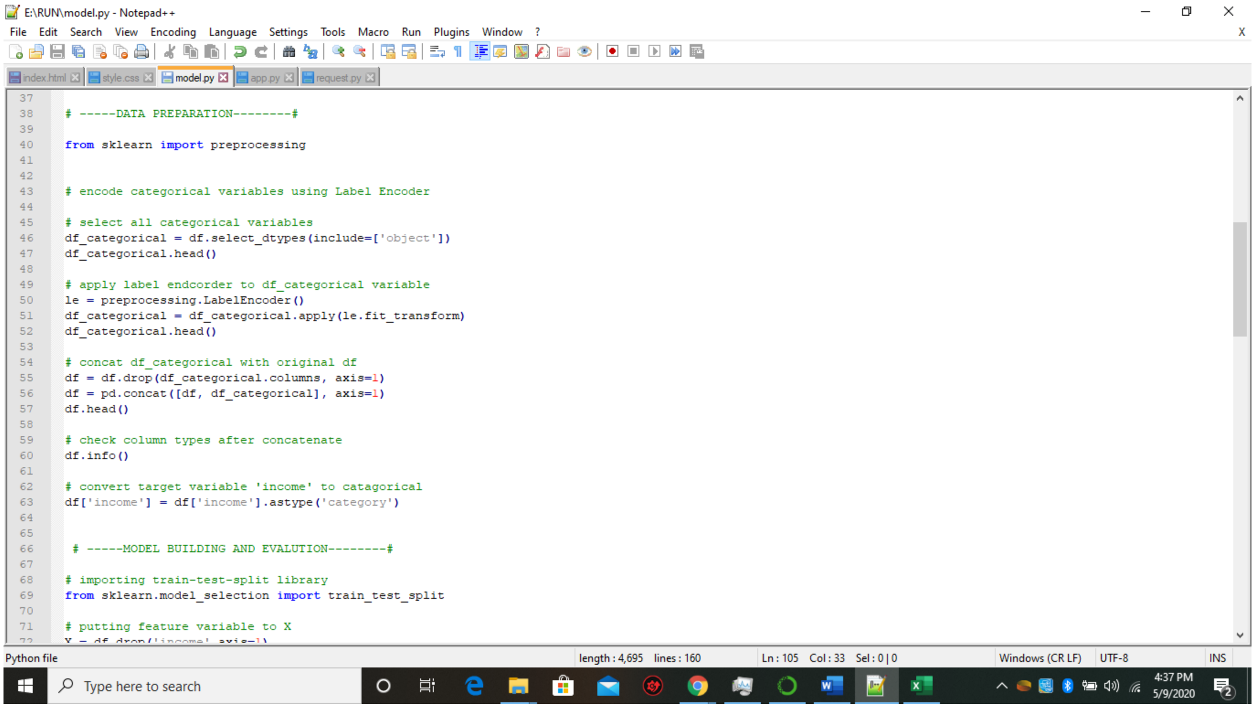
Figure 5. 1 – Accuracy of the algorithm

Code Implementation

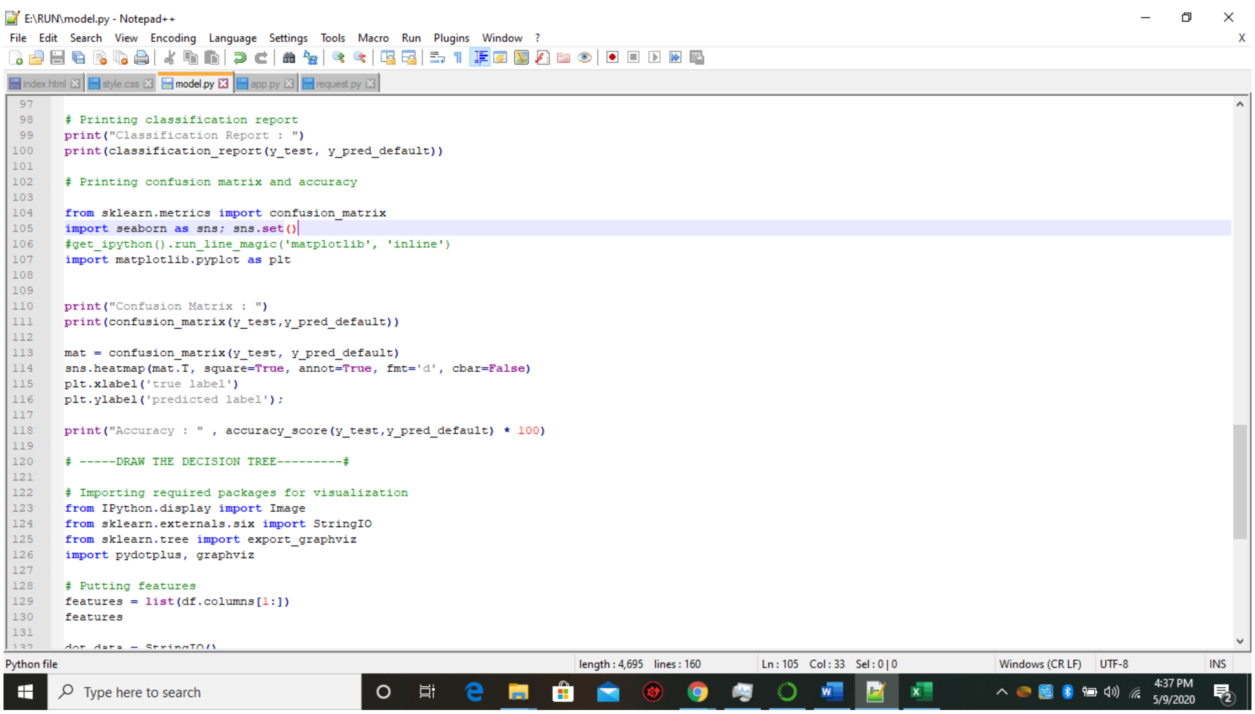
This project is included with 4 parts,

1. model.py — This contains code for the machine learning model to predict sales in the third month based on the sales in the first two months.

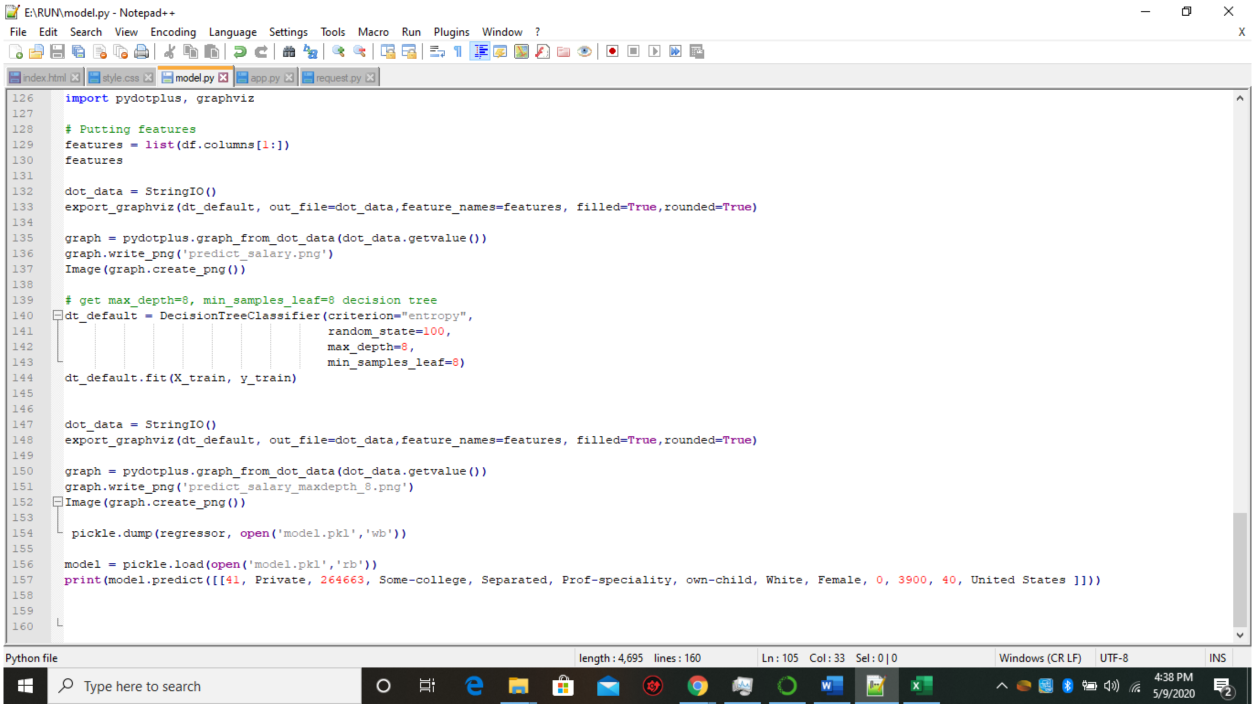




```python
# -----DATA PREPARATION--------#

from sklearn import preprocessing


# encode categorical variables using Label Encoder

# select all categorical variables
df_categorical = df.select_dtypes(include=['object'])
df_categorical.head()

# apply label endcorder to df_categorical variable
le = preprocessing.LabelEncoder()
df_categorical = df_categorical.apply(le.fit_transform)
df_categorical.head()

# concat df_categorical with original df
df = df.drop(df_categorical.columns, axis=1)
df = pd.concat([df, df_categorical], axis=1)
df.head()

# check column types after concatenate
df.info()

# convert target variable 'income' to catagorical
df['income'] = df['income'].astype('category')


 # -----MODEL BUILDING AND EVALUTION--------#

# importing train-test-split library
from sklearn.model_selection import train_test_split

# putting feature variable to X
X = df.drop('income',axis=1)
```

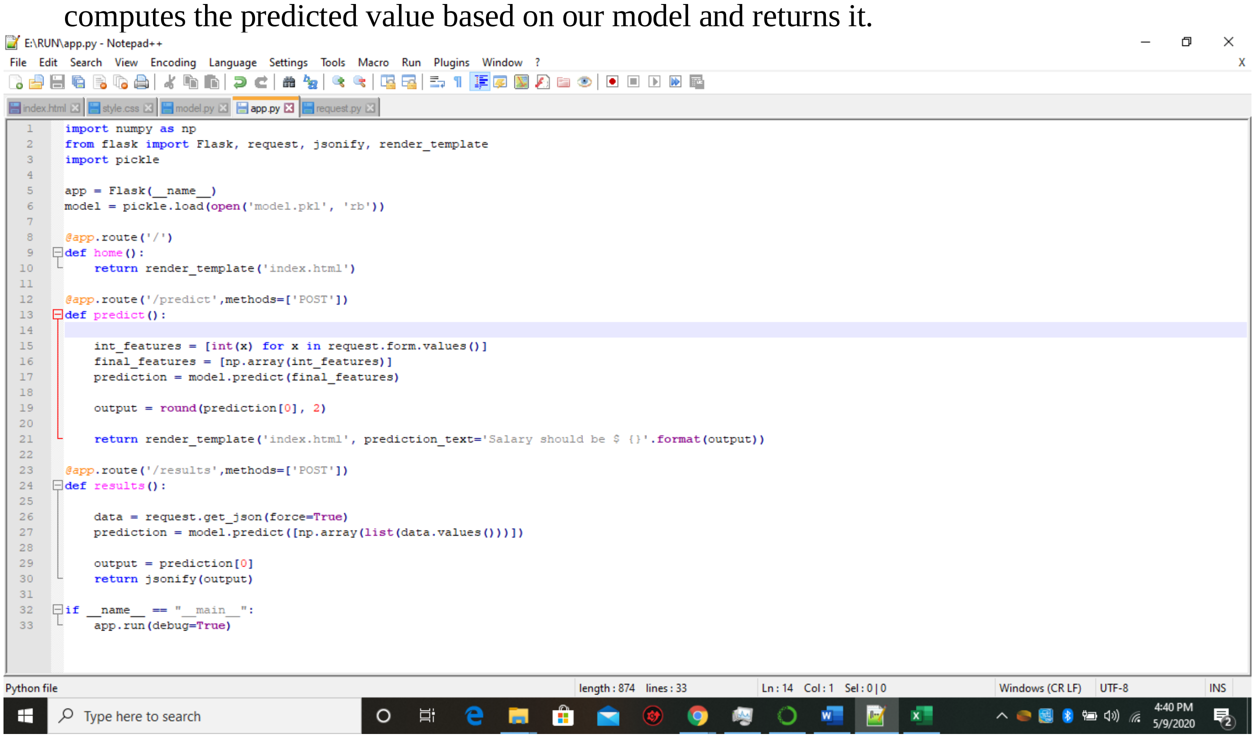


```python

# Printing classification report
print("Classification Report : ")
print(classification_report(y_test, y_pred_default))

# Printing confusion matrix and accuracy

from sklearn.metrics import confusion_matrix
import seaborn as sns; sns.set()
#get_ipython().run_line_magic('matplotlib', 'inline')
import matplotlib.pyplot as plt


print("Confusion Matrix : ")
print(confusion_matrix(y_test,y_pred_default))

mat = confusion_matrix(y_test, y_pred_default)
sns.heatmap(mat.T, square=True, annot=True, fmt='d', cbar=False)
plt.xlabel('true label')
plt.ylabel('predicted label');

print("Accuracy : " , accuracy_score(y_test,y_pred_default) * 100)

# -----DRAW THE DECISION TREE---------#

# Importing required packages for visualization
from IPython.display import Image
from sklearn.externals.six import StringIO
from sklearn.tree import export_graphviz
import pydotplus, graphviz

# Putting features
features = list(df.columns[1:])
features

dot_data = StringIO()
```



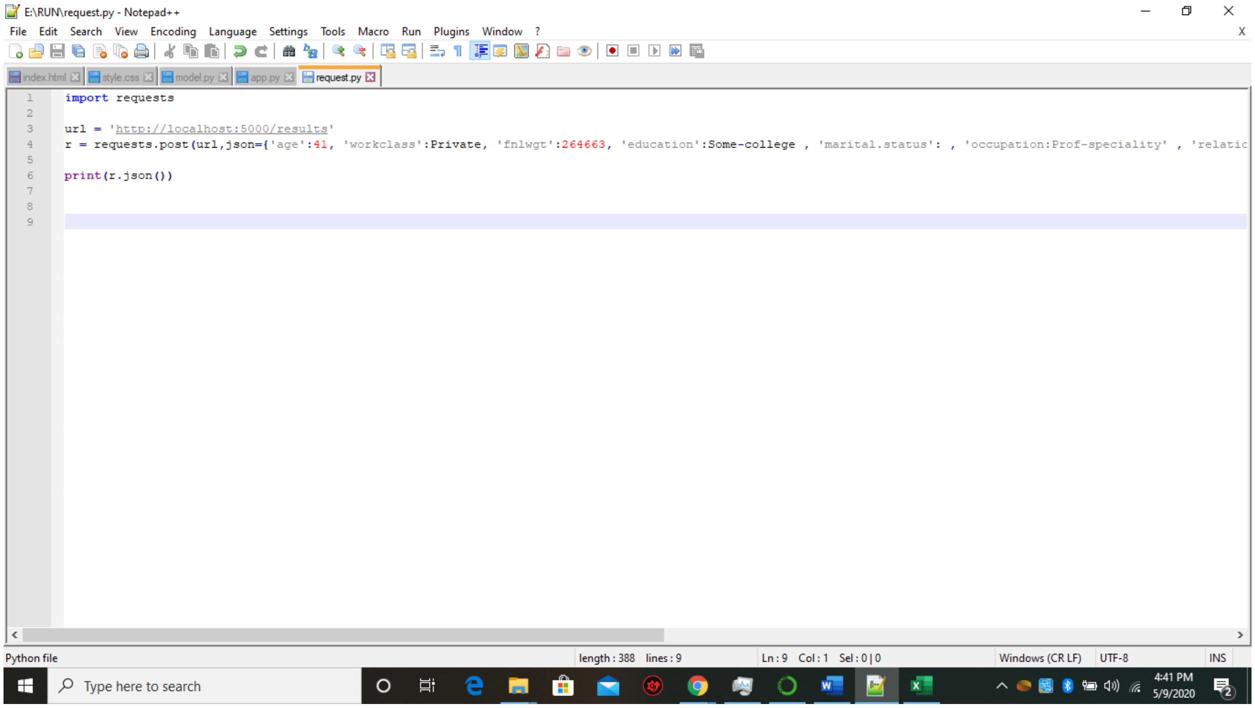
```python
import pydotplus, graphviz

# Putting features
features = list(df.columns[1:])
features

dot_data = StringIO()
export_graphviz(dt_default, out_file=dot_data,feature_names=features, filled=True,rounded=True)

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('predict_salary.png')
Image(graph.create_png())

# get max_depth=8, min_samples_leaf=8 decision tree
dt_default = DecisionTreeClassifier(criterion="entropy",
                                    random_state=100,
                                    max_depth=8,
                                    min_samples_leaf=8)
dt_default.fit(X_train, y_train)


dot_data = StringIO()
export_graphviz(dt_default, out_file=dot_data,feature_names=features, filled=True,rounded=True)

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('predict_salary_maxdepth_8.png')
Image(graph.create_png())

 pickle.dump(regressor, open('model.pkl','wb'))

model = pickle.load(open('model.pkl','rb'))
print(model.predict([[41, Private, 264663, Some-college, Separated, Prof-speciality, own-child, White, Female, 0, 3900, 40, United States ]]))
```

2. app.py — This contains Flask APIs that receives sales details through GUI or API calls, computes the predicted value based on our model and returns it.

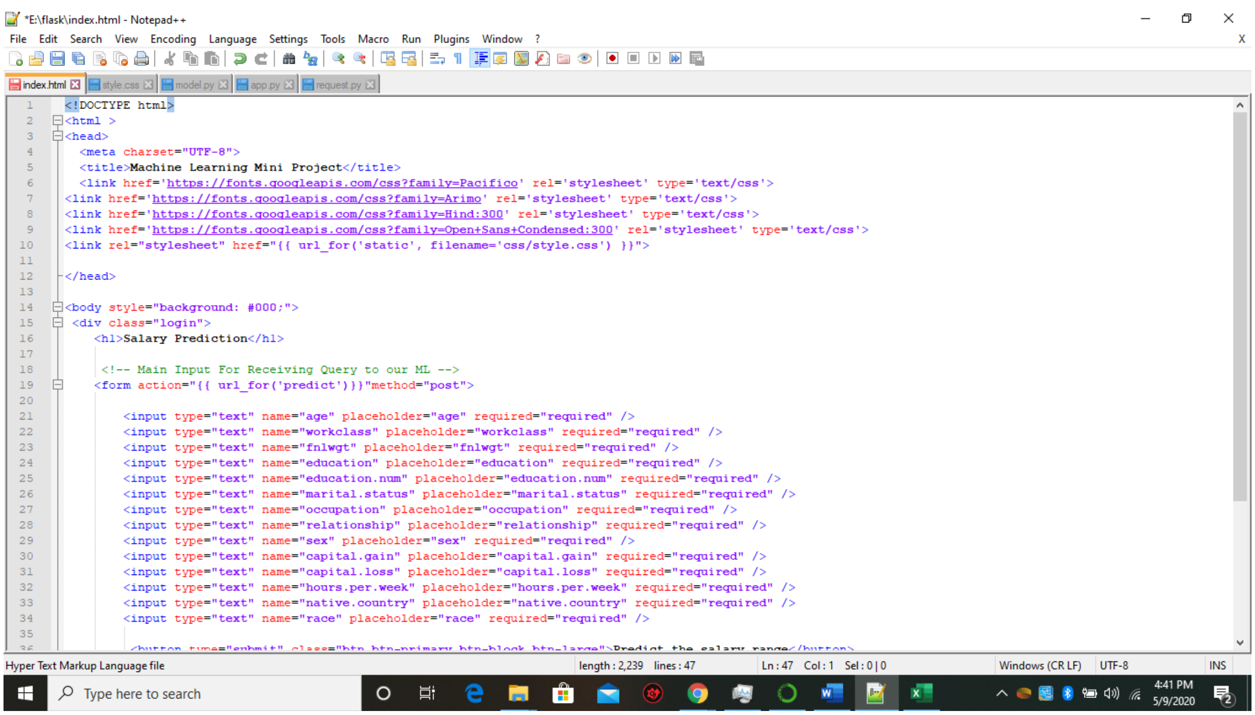


```python
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

app = Flask(__name__)
model = pickle.load(open('model.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict',methods=['POST'])
def predict():

    int_features = [int(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output = round(prediction[0], 2)

    return render_template('index.html', prediction_text='Salary should be $ {}'.format(output))

@app.route('/results',methods=['POST'])
def results():

    data = request.get_json(force=True)
    prediction = model.predict([np.array(list(data.values()))])

    output = prediction[0]
    return jsonify(output)

if __name__ == "__main__":
    app.run(debug=True)
```

3. request.py — This uses requests module to call APIs defined in app.py and displays the returned value.



4. HTML/CSS — This contains the HTML template and CSS styling to allow user to enter sales detail and displays the predicted sales in the third month.

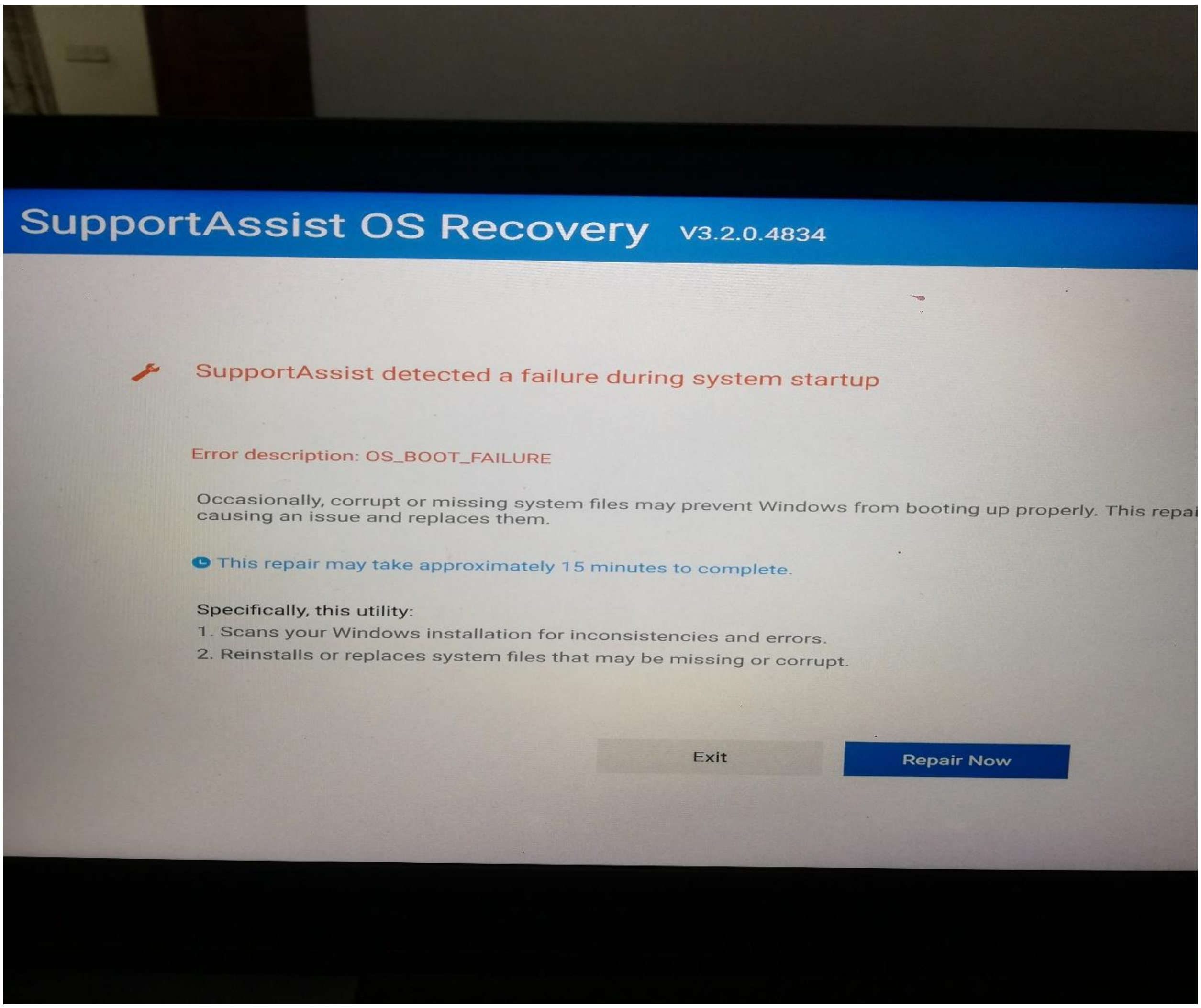


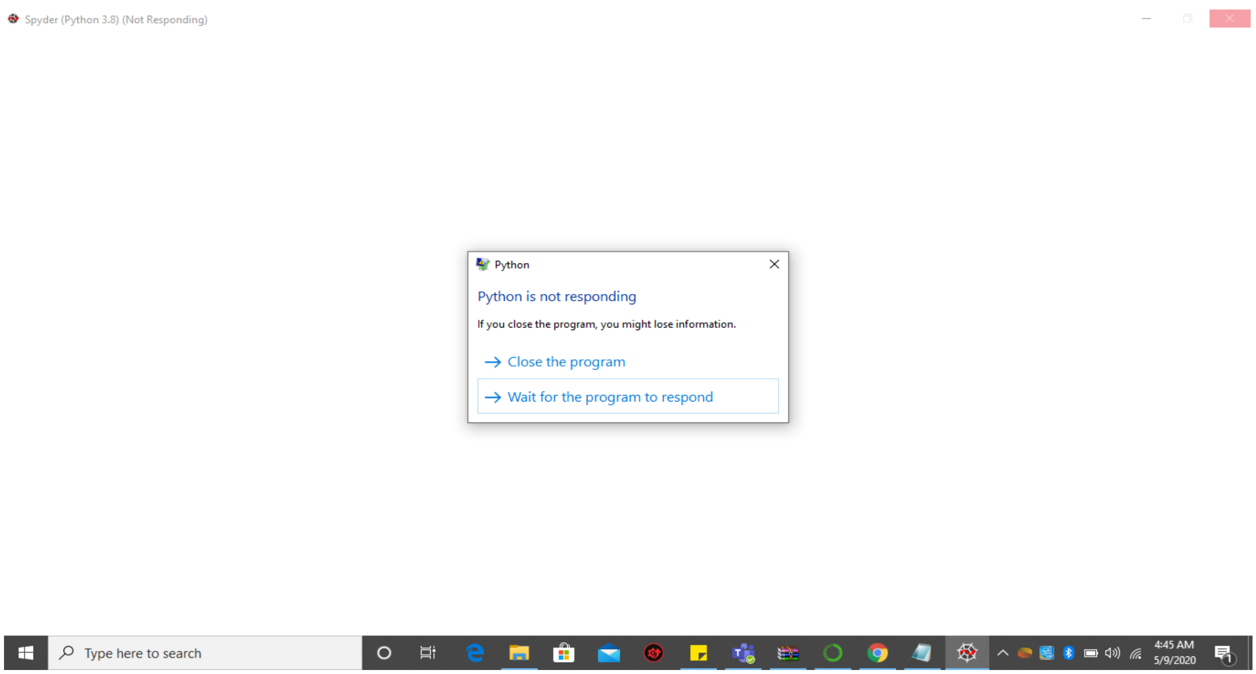
```html
<body style="background: #000;">
 <div class="login">
	<h1>Salary Prediction</h1>

	 <!-- Main Input For Receiving Query to our ML -->
	<form action="{{ url_for('predict')}}"method="post">

		<input type="text" name="age" placeholder="age" required="required" />
		<input type="text" name="workclass" placeholder="workclass" required="required" />
		<input type="text" name="fnlwgt" placeholder="fnlwgt" required="required" />
		<input type="text" name="education" placeholder="education" required="required" />
		<input type="text" name="education.num" placeholder="education.num" required="required" />
		<input type="text" name="marital.status" placeholder="marital.status" required="required" />
		<input type="text" name="occupation" placeholder="occupation" required="required" />
		<input type="text" name="relationship" placeholder="relationship" required="required" />
		<input type="text" name="sex" placeholder="sex" required="required" />
		<input type="text" name="capital.gain" placeholder="capital.gain" required="required" />
		<input type="text" name="capital.loss" placeholder="capital.loss" required="required" />
		<input type="text" name="hours.per.week" placeholder="hours.per.week" required="required" />
		<input type="text" name="native.country" placeholder="native.country" required="required" />
		<input type="text" name="race" placeholder="race" required="required" />

		 <button type="submit" class="btn btn-primary btn-block btn-large">Predict the salary range</button>

	</form>

   <br>
   <br>
   {{ prediction_text }}

 </div>
</body>
</html>
```

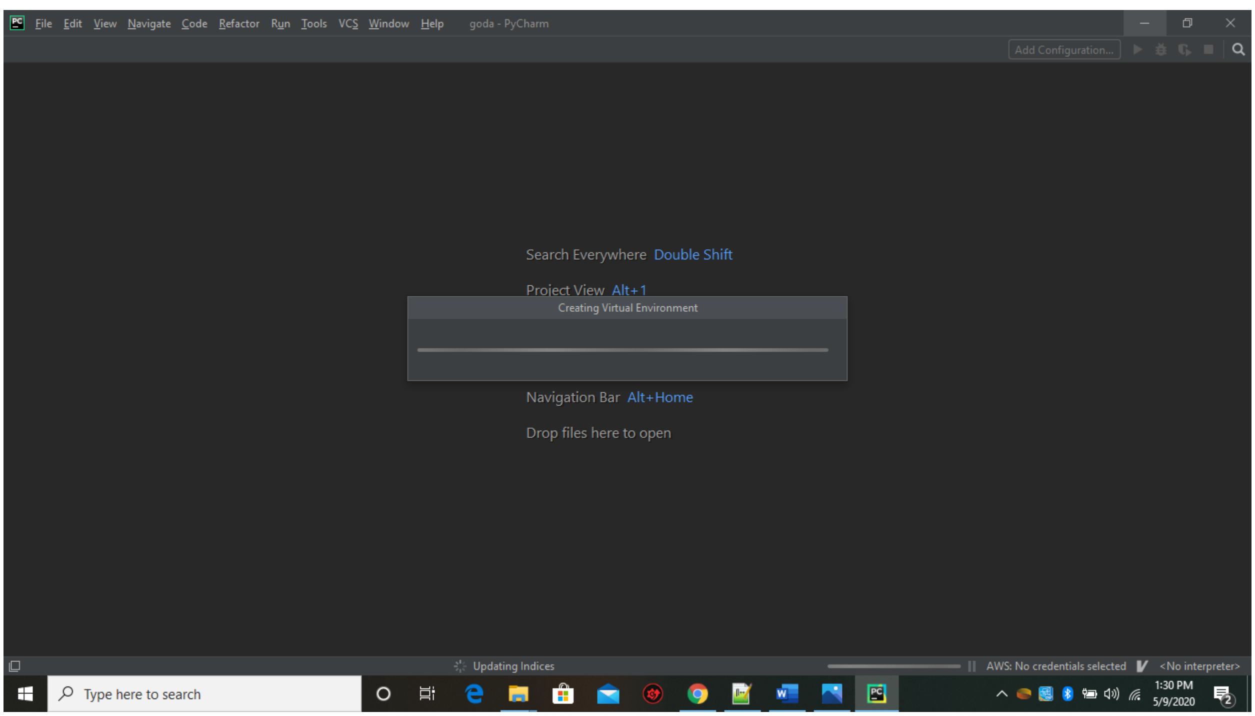
Obstacles.

The Laptop which I used to complete this assignment had got a mother board failure few days ago. I was unable to repair it due to the privilege situation in my area. However I could find another laptop. The problem which I had to face is the existing software are force shut down at once. I was able to train the model using jupyterNote book. But the time I am completing this document I can't open any software such as pycharm or spyder. As well as anaconda prompt is taking vey long time to open. Nearly about 1 hour. As I think the RAM of my existing lap is not enough. It's a 4GB ram. Extremely Sorry for the inconvenience caused.

Here are photos of my broken laptop.




Spyder (Python 3.8) (Not Responding)
Python
Python is not responding
If you close the program, you might lose information.
Close the program
Wait for the program to respond
Type here to search
4:45 AM
5/9/2020


File Edit View Navigate Code Refactor Run Tools VCS Window Help
goda - PyCharm
Add Configuration...
Search Everywhere Double Shift
Project View Alt+1
Creating Virtual Environment
Navigation Bar Alt+Home
Drop files here to open
Updating Indices
AWS: No credentials selected
<No interpreter>
Type here to search
1:30 PM
5/9/2020

References

[1] "Kaggle," [Online]. Available: https://www.kaggle.com/uciml/adult-census-income/downloads/adult-census-income.zip/3.

[2] "medium," [Online]. Available: https://medium.com/cracking-the-data-science-interview/decision-trees-how-to-optimize-my-decision-making-process-e1f327999c7a.

[3] "thatascience," [Online]. Available: https://thatascince.com/learn-machine-learning/gini-entropy/.

[4] "muthu," [Online]. Available: https://muthu.co/understanding-the-classification-report-in-sklearn/.

[5] "dataschool," [Online]. Available: https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/.

[6] L. Weynars, "lwmachinelearning," [Online]. Available: https://lwmachinelearning.wordpress.com/portfolio/adults-data-set/.