CSE 587: DATA INTENSIVE COMPUTING

Project Phase #3

Name: Viditha Wudaru **UBID:** 50442670

Name: Jayavani Akkiraju **UBID:** 50442863

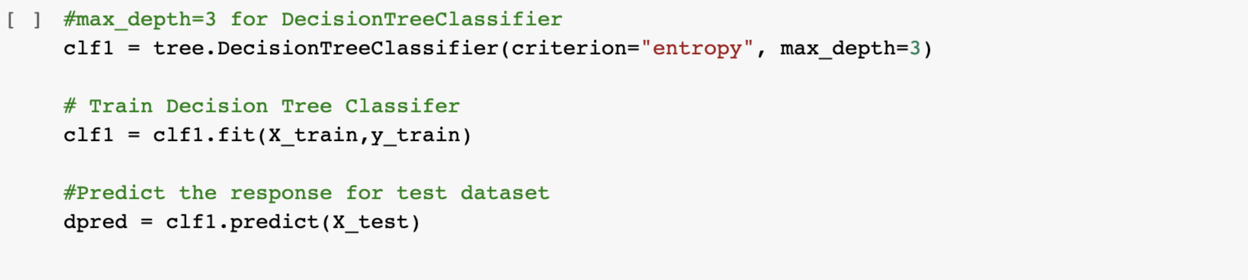
Introduction:
Our objective with the Adult census income dataset is to predict the income level to be <=50k or >50k based on factors such as occupation, work hours, education, country, etc. Phase 1 involved choosing the dataset and preprocessing it, while phase 2 involved training the model by fitting various classification algorithms, such as Decision Trees, Naive Bayeses, etc. The third phase of the project integrates the front-end with the back-end model prediction. An intuitive user interface that provides input fields to generate the required output (income level).

Overview of Model's Accuracy:

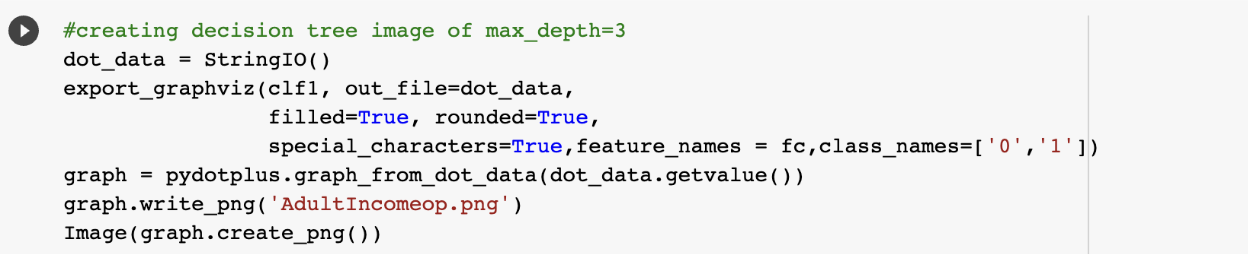
Model	Metric	Train Accuracy	Test Accuracy
Logistic Regression	solver-'lbfgs'	82.5	83.0
Naive Bayes	model-Gaussian NB	80.9	80.9
Decision Tree	max_depth-3	97.5	81.7
KNN	n_neighbors-39	84.4	84.2
SVC	kernel-'rbf'	80.2	84.9
SGD	loss-'log_loss'	79.1	82.6
AdaBoost	model-Adaboost	86.2	86.0

We can see that the training accuracy of the Decision Tree is the highest among all. So, we are importing the decision tree fit into a pickle file which can be used to predict the results. The user interface (UI) is built using HTML and the styles are added to the HTML file using CSS in our application.

Decision Tree Tuning:
With this fit of the decision tree using criterion as entropy and max_depth = 3, we achieved the best train accuracy.



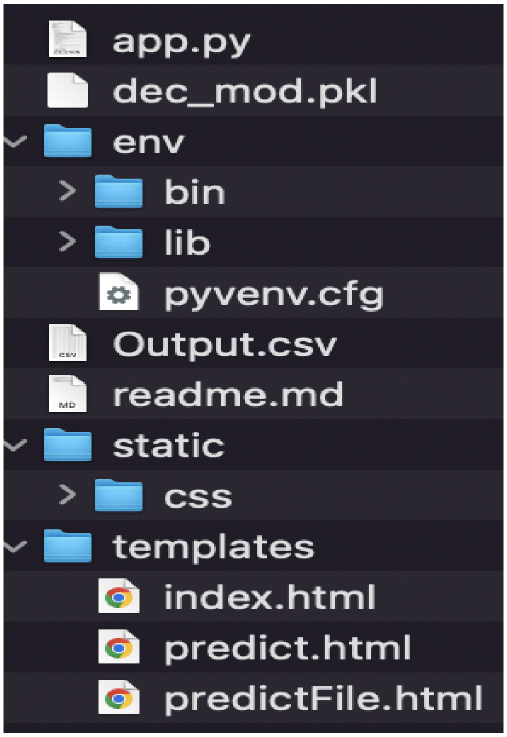
```
[ ] #max_depth=3 for DecisionTreeClassifier
    clf1 = tree.DecisionTreeClassifier(criterion="entropy", max_depth=3)

    # Train Decision Tree Classifer
    clf1 = clf1.fit(X_train,y_train)

    #Predict the response for test dataset
    dpred = clf1.predict(X_test)
```



```
#creating decision tree image of max_depth=3
dot_data = StringIO()
export_graphviz(clf1, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,feature_names = fc,class_names=['0','1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('AdultIncomeop.png')
Image(graph.create_png())
```

Folder Structure:

Building a complete model: The folder structure of the project looks as follows:



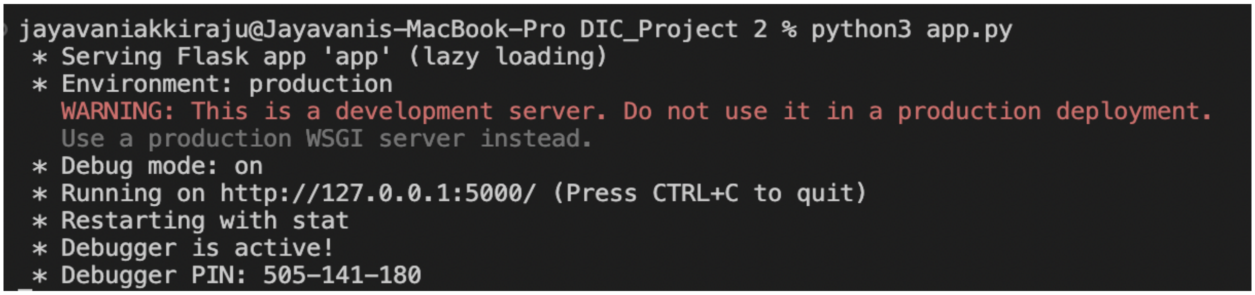
```
  app.py
  dec_mod.pkl
v env
    > bin
    > lib
      pyvenv.cfg
  Output.csv
  readme.md
v static
    > css
v templates
      index.html
      predict.html
      predictFile.html
```

The app.py contains the necessary Python code to route the client's request. Basically, it's implemented using the Flask framework. Python-based Flask is a framework for building web applications. Python serializes and deserializes objects using the Pickle module. A Python object can be converted to a byte stream for storage or transmission. Dec_mod.pkl is the pickle module file that imports the decision tree model into the code being run.

Under the static folder, there is a CSS folder, which contains the style sheet files.

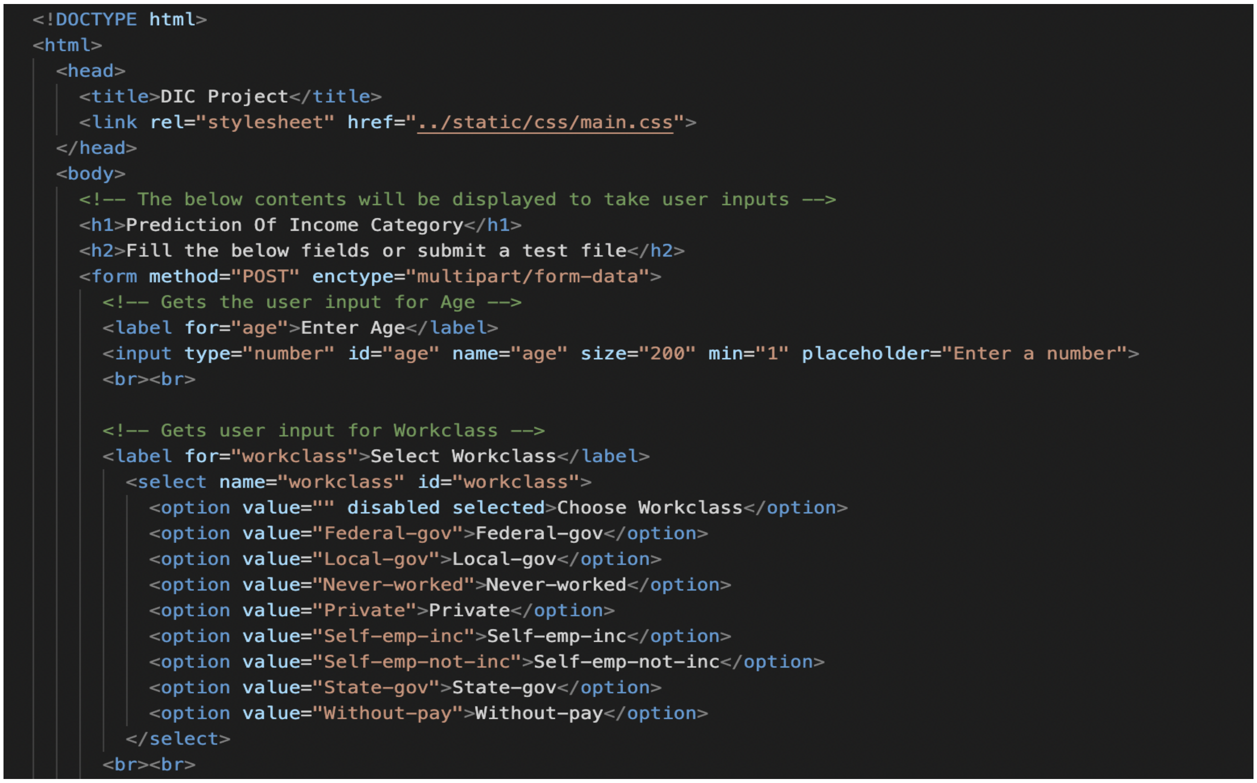
Under the templates folder, all the required HTML files that contain the front-end display code are given.

Running the app.py file on the terminal in the VSCode editor using the following command:
>>**python3 app.py**
We can see below that the application is running on localhost.
http://127.0.0.1:5000/



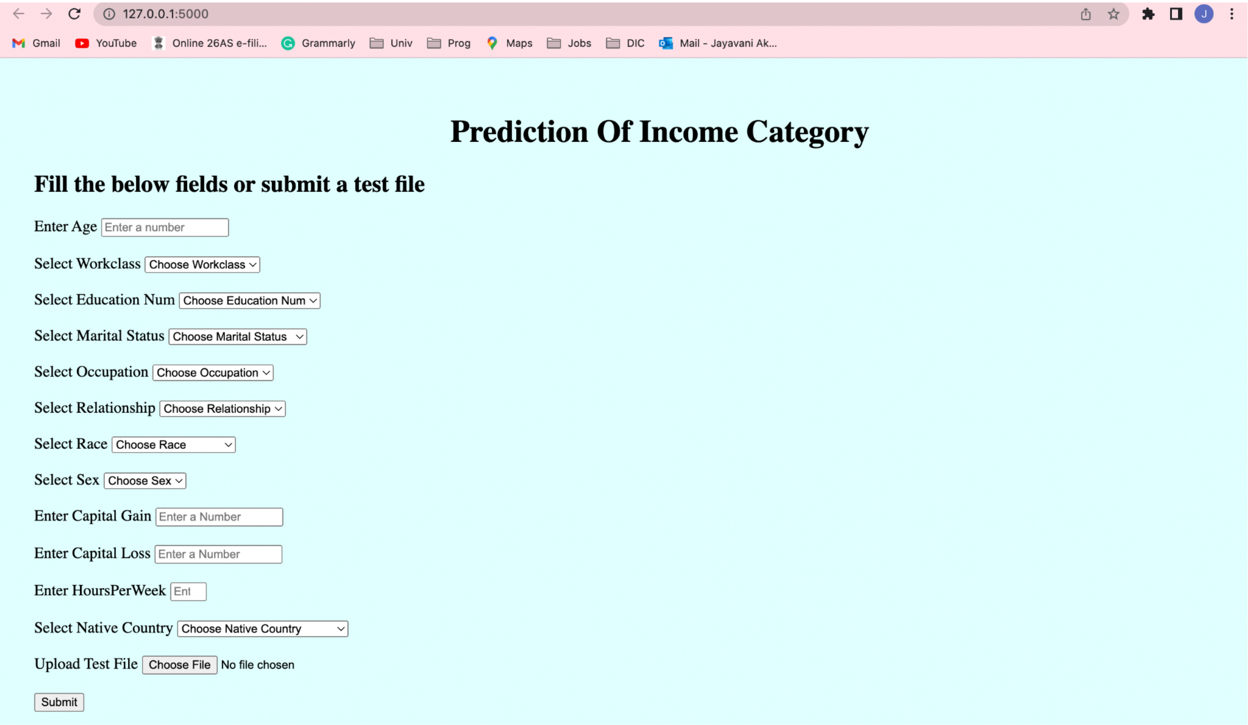
```
jayavaniakkiraju@Jayavanis-MacBook-Pro DIC_Project 2 % python3 app.py
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 505-141-180
```

User Interface Code:



```
<!DOCTYPE html>
<html>
  <head>
    <title>DIC Project</title>
    <link rel="stylesheet" href="../static/css/main.css">
  </head>
  <body>
    <!-- The below contents will be displayed to take user inputs -->
    <h1>Prediction Of Income Category</h1>
    <h2>Fill the below fields or submit a test file</h2>
    <form method="POST" enctype="multipart/form-data">
      <!-- Gets the user input for Age -->
      <label for="age">Enter Age</label>
      <input type="number" id="age" name="age" size="200" min="1" placeholder="Enter a number">
      <br><br>

      <!-- Gets user input for Workclass -->
      <label for="workclass">Select Workclass</label>
        <select name="workclass" id="workclass">
          <option value="" disabled selected>Choose Workclass</option>
          <option value="Federal-gov">Federal-gov</option>
          <option value="Local-gov">Local-gov</option>
          <option value="Never-worked">Never-worked</option>
          <option value="Private">Private</option>
          <option value="Self-emp-inc">Self-emp-inc</option>
          <option value="Self-emp-not-inc">Self-emp-not-inc</option>
          <option value="State-gov">State-gov</option>
          <option value="Without-pay">Without-pay</option>
        </select>
      <br><br>
```

The index.html file is used to build the user interface initially to input the fields.



Prediction Of Income Category

Fill the below fields or submit a test file

Enter Age Enter a number

Select Workclass Choose Workclass

Select Education Num Choose Education Num

Select Marital Status Choose Marital Status

Select Occupation Choose Occupation

Select Relationship Choose Relationship

Select Race Choose Race

Select Sex Choose Sex

Enter Capital Gain Enter a Number

Enter Capital Loss Enter a Number

Enter HoursPerWeek Ent

Select Native Country Choose Native Country

Upload Test File Choose File No file chosen

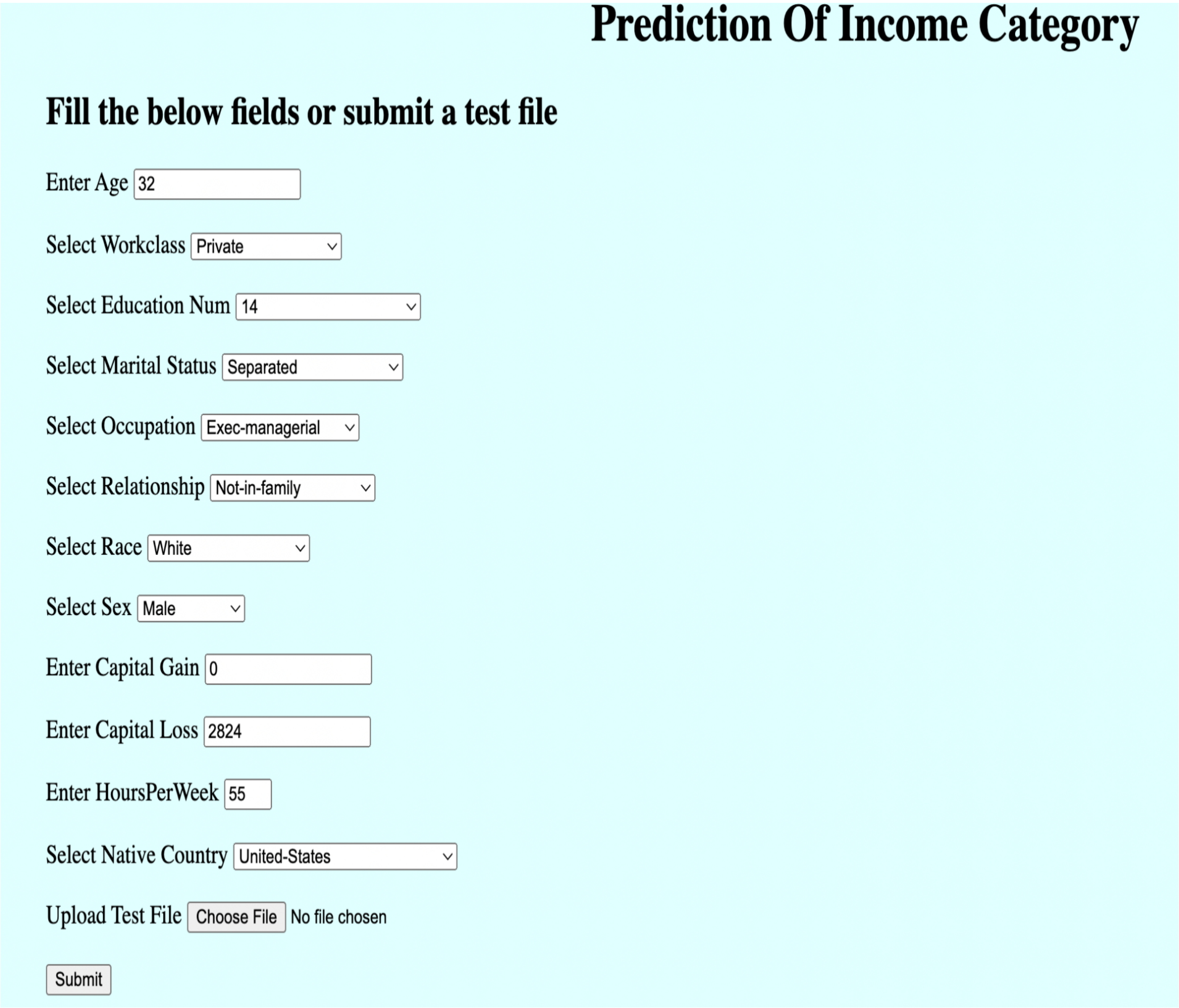
Submit

There are two ways in which we can predict the outcome.
i. By filling in the fields
ii. By uploading the test file

The fields provided are:
Age: The age of a person given as an input
Workclass: [Federal-gov, Local-gov, Never-worked, Private, Self-emp-inc, Self-emp-not-inc, State-gov, Without-pay]
Education Num: [Range: 1 to 16]
Marital status: ['Divorced': 0, 'Married-AF-spouse': 1, 'Married-civ-spouse': 2, 'Married-spouse-absent': 3, 'Never-married': 4, 'Separated': 5, 'Widowed': 6]
Occupation: ['Adm-clerical': 0, 'Armed-Forces': 1, 'Craft-repair': 2, 'Exec-managerial': 3, 'Farming-fishing': 4, 'Handlers-cleaners': 5, 'Machine-op-inspct': 6, 'Other': 7, 'Other-service': 8, 'Priv-house-serv': 9, 'Prof-specialty': 10, 'Protective-serv': 11, 'Sales': 12, 'Tech-support': 13, 'Transport-moving': 14]
Relationship: ['Husband': 0, 'Not-in-family': 1, 'Other-relative': 2, 'Own-child': 3, 'Unmarried': 4, 'Wife': 5]
Race: ['Amer-Indian-Eskimo': 0, 'Asian-Pac-Islander': 1, 'Black': 2, 'Other': 3, 'White': 4]
Sex: ['Female': 0, 'Male': 1]
Capital Gain: [Min : 0]

Capital Loss: [Min : 0]
HoursPerWeek: [Min : 1, Max : 99]
Native Country: ['Cambodia': 0, 'Canada': 1, 'China': 2, 'Columbia': 3, 'Cuba': 4, 'Dominican-Republic': 5, 'Ecuador': 6, 'El-Salvador': 7, 'England': 8, 'France': 9, 'Germany': 10, 'Greece': 11, 'Guatemala': 12, 'Haiti': 13, 'Holand-Netherlands': 14, 'Honduras': 15, 'Hong': 16, 'Hungary': 17, 'India': 18, 'Iran': 19, 'Ireland': 20, 'Italy': 21, 'Jamaica': 22, 'Japan': 23, 'Laos': 24, 'Mexico': 25, 'Nicaragua': 26, 'Outlying-US(Guam-USVI-etc)': 27, 'Peru': 28, 'Philippines': 29, 'Poland': 30, 'Portugal': 31, 'Puerto-Rico': 32, 'Scotland': 33, 'South': 34, 'Taiwan': 35, 'Thailand': 36, 'Trinadad&Tobago': 37, 'United-States': 38, 'Unknown': 39, 'Vietnam': 40, 'Yugoslavia': 41]

1. Manually giving input in the fields provided:



Prediction Of Income Category

Fill the below fields or submit a test file

Enter Age 32

Select Workclass Private

Select Education Num 14

Select Marital Status Separated

Select Occupation Exec-managerial

Select Relationship Not-in-family

Select Race White

Select Sex Male

Enter Capital Gain 0

Enter Capital Loss 2824

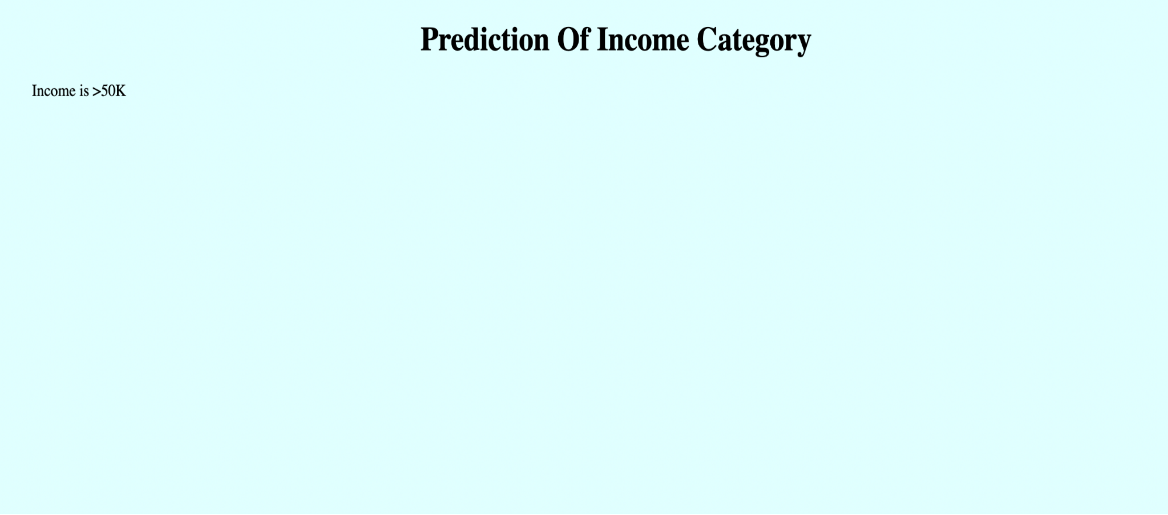
Enter HoursPerWeek 55

Select Native Country United-States

Upload Test File No file chosen

We have provided the fields with the required information to generate a response on whether the salary could be <=50k or >50k.

After clicking on submit button, the output that is displayed is as follows:

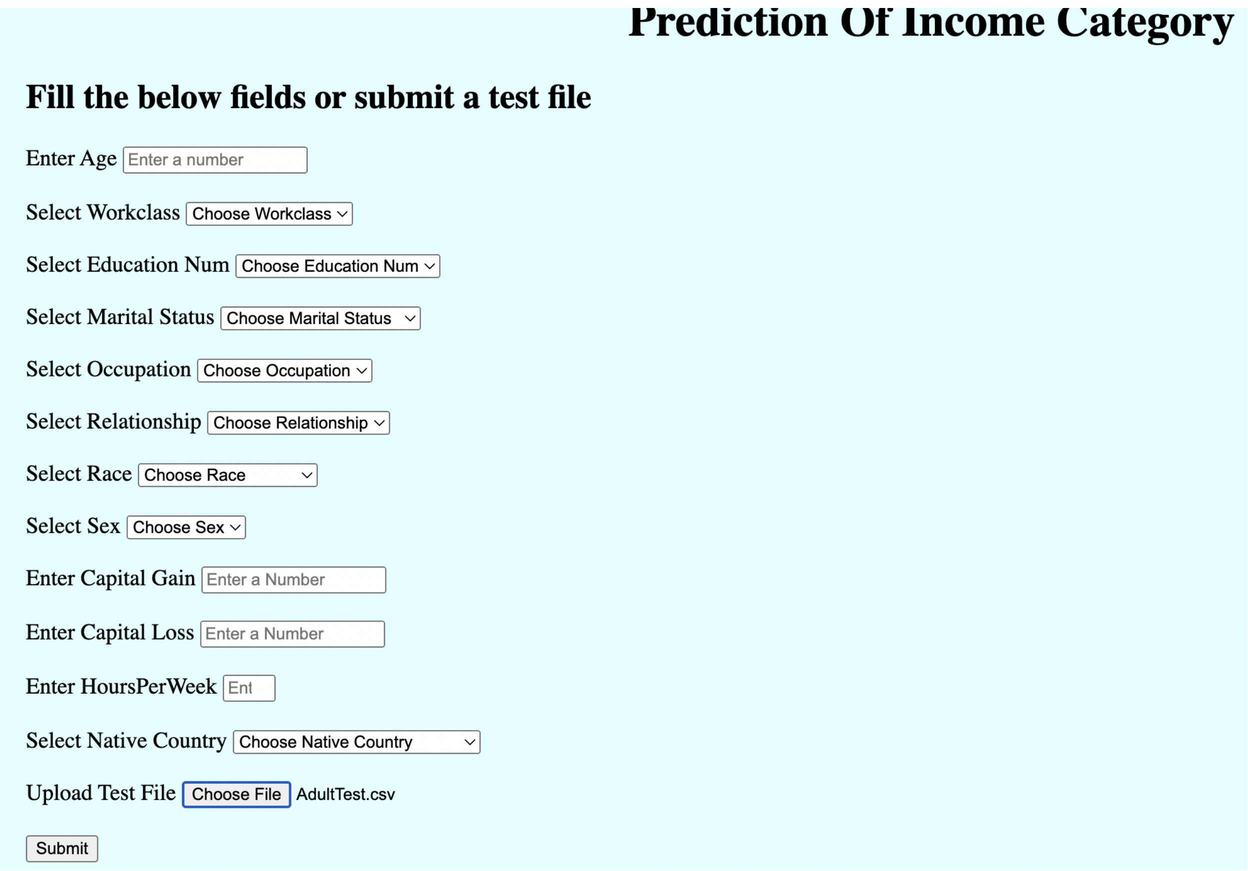

Prediction Of Income Category

Income is >50K

The output is generated in a new HTML page instead of on the same input page giving the fields. This makes it interactive to the user as well as it gets redirected to a new page of HTML.

For the above-given input, the income is predicted as >50k.

2. Uploading a test file:


Prediction Of Income Category

Fill the below fields or submit a test file

Enter Age Enter a number

Select Workclass Choose Workclass

Select Education Num Choose Education Num

Select Marital Status Choose Marital Status

Select Occupation Choose Occupation

Select Relationship Choose Relationship

Select Race Choose Race

Select Sex Choose Sex

Enter Capital Gain Enter a Number

Enter Capital Loss Enter a Number

Enter HoursPerWeek Ent

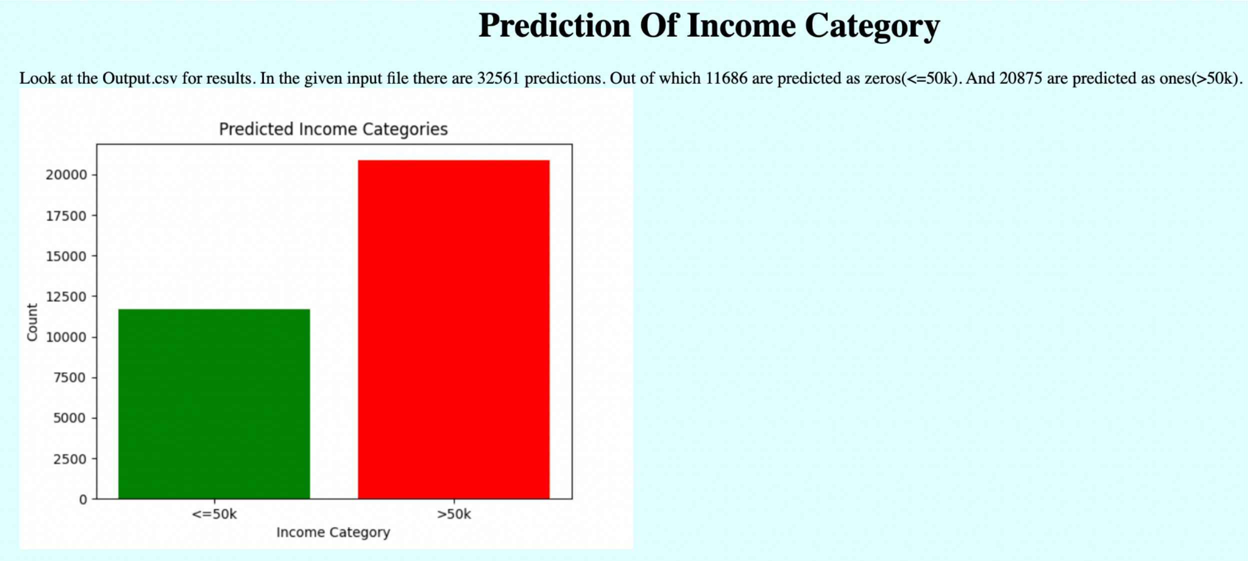
Select Native Country Choose Native Country

Upload Test File Choose File AdultTest.csv

Submit

Uploaded a test file AdultTest.csv. The test file contains 32561 rows.
We can see that 11686 rows are predicted as 0 (<=50k) in green and 20875 rows are predicted as 1 (>50k) in red out of a total of 32561 rows.

Visualization:



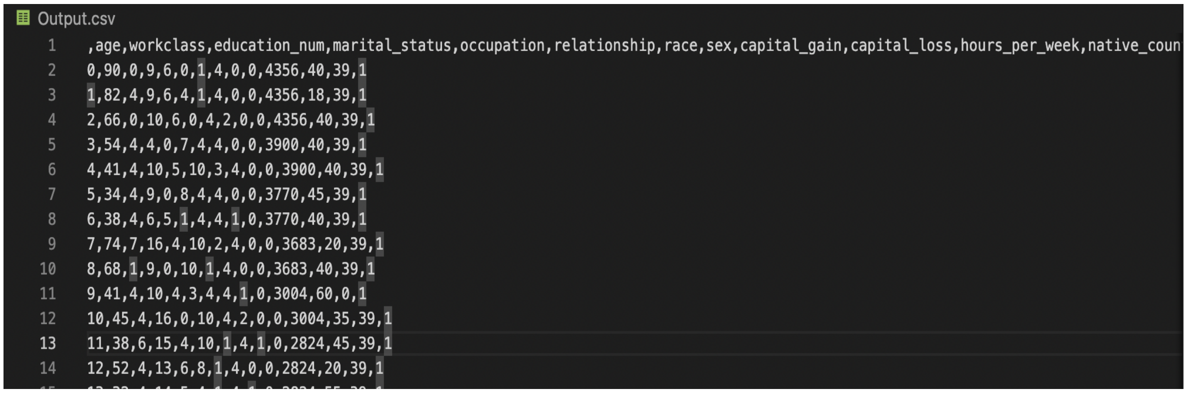
Prediction Of Income Category

Look at the Output.csv for results. In the given input file there are 32561 predictions. Out of which 11686 are predicted as zeros(<=50k). And 20875 are predicted as ones(>50k).

A bar graph displaying the count of the predicted income category. We can see that the frequency of the target being 1 (>50k) is more than 0 (<=50k).

The X-axis of the bar graph represents the income category. It is a binary class variable where 0 represents <=50k as the income range and 1 represents >50k as the income range.
The Y-axis represents the count/frequency of the binary output variable.

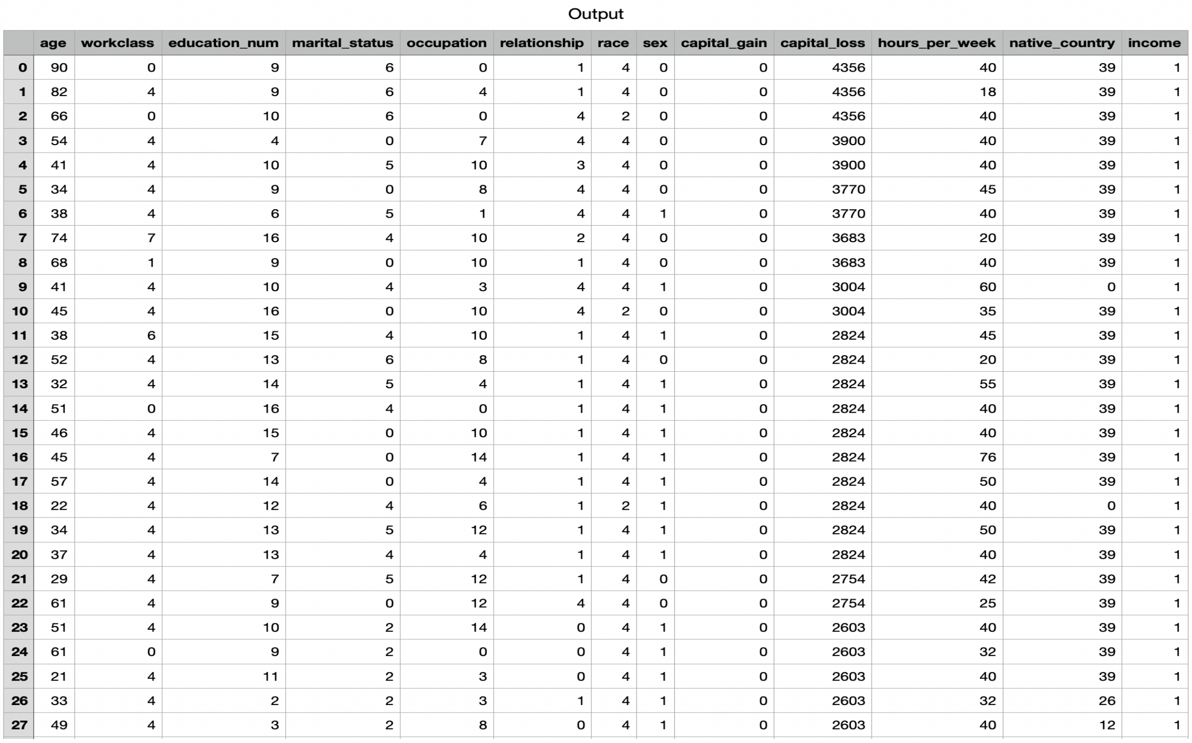
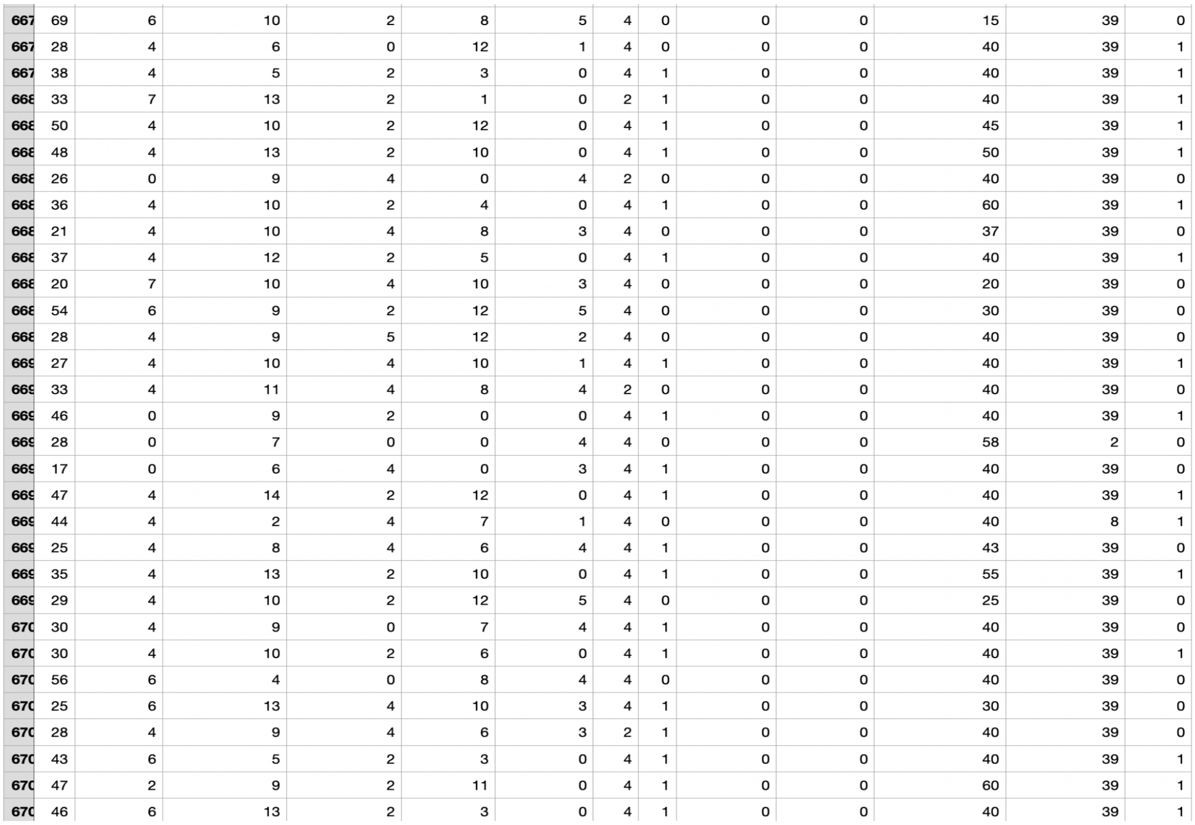
Output.csv can be downloaded as a part of the output after uploading a test CSV file.



Output.csv

```
,age,workclass,education_num,marital_status,occupation,relationship,race,sex,capital_gain,capital_loss,hours_per_week,native_coun
0,90,0,9,6,0,1,4,0,0,4356,40,39,1
1,82,4,9,6,4,1,4,0,0,4356,18,39,1
2,66,0,10,6,0,4,2,0,0,4356,40,39,1
3,54,4,4,0,7,4,4,0,0,3900,40,39,1
4,41,4,10,5,10,3,4,0,0,3900,40,39,1
5,34,4,9,0,8,4,4,0,0,3770,45,39,1
6,38,4,6,5,1,4,4,1,0,3770,40,39,1
7,74,7,16,4,10,2,4,0,0,3683,20,39,1
8,68,1,9,0,10,1,4,0,0,3683,40,39,1
9,41,4,10,4,3,4,4,1,0,3004,60,0,1
10,45,4,16,0,10,4,2,0,0,3004,35,39,1
11,38,6,15,4,10,1,4,1,0,2824,45,39,1
12,52,4,13,6,8,1,4,0,0,2824,20,39,1
```

The output CSV file contains all the fields that are mapped into numerics.

Output

	age	workclass	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country	income
0	90	0	9	6	0	1	4	0	0	4356	40	39	1
1	82	4	9	6	4	1	4	0	0	4356	18	39	1
2	66	0	10	6	0	4	2	0	0	4356	40	39	1
3	54	4	4	0	7	4	4	0	0	3900	40	39	1
4	41	4	10	5	10	3	4	0	0	3900	40	39	1
5	34	4	9	0	8	4	4	0	0	3770	45	39	1
6	38	4	6	5	1	4	4	1	0	3770	40	39	1
7	74	7	16	4	10	2	4	0	0	3683	20	39	1
8	68	1	9	0	10	1	4	0	0	3683	40	39	1
9	41	4	10	4	3	4	4	1	0	3004	60	0	1
10	45	4	16	0	10	4	2	0	0	3004	35	39	1
11	38	6	15	4	10	1	4	1	0	2824	45	39	1
12	52	4	13	6	8	1	4	0	0	2824	20	39	1
13	32	4	14	5	4	1	4	1	0	2824	55	39	1
14	51	0	16	4	0	1	4	1	0	2824	40	39	1
15	46	4	15	0	10	1	4	1	0	2824	40	39	1
16	45	4	7	0	14	1	4	1	0	2824	76	39	1
17	57	4	14	0	4	1	4	1	0	2824	50	39	1
18	22	4	12	4	6	1	2	1	0	2824	40	0	1
19	34	4	13	5	12	1	4	1	0	2824	50	39	1
20	37	4	13	4	4	1	4	1	0	2824	40	39	1
21	29	4	7	5	12	1	4	0	0	2754	42	39	1
22	61	4	9	0	12	4	4	0	0	2754	25	39	1
23	51	4	10	2	14	0	4	1	0	2603	40	39	1
24	61	0	9	2	0	0	4	1	0	2603	32	39	1
25	21	4	11	2	3	0	4	1	0	2603	40	39	1
26	33	4	2	2	3	1	4	1	0	2603	32	26	1
27	49	4	3	2	8	0	4	1	0	2603	40	12	1
667	69	6	10	2	8	5	4	0	0	0	15	39	0
667	28	4	6	0	12	1	4	0	0	0	40	39	1
667	38	4	5	2	3	0	4	1	0	0	40	39	1
668	33	7	13	2	1	0	2	1	0	0	40	39	1
668	50	4	10	2	12	0	4	1	0	0	45	39	1
668	48	4	13	2	10	0	4	1	0	0	50	39	1
668	26	0	9	4	0	4	2	0	0	0	40	39	0
668	36	4	10	2	4	0	4	1	0	0	60	39	1
668	21	4	10	4	8	3	4	0	0	0	37	39	0
668	37	4	12	2	5	0	4	1	0	0	40	39	1
668	20	7	10	4	10	3	4	0	0	0	20	39	0
668	54	6	9	2	12	5	4	0	0	0	30	39	0
668	28	4	9	5	12	2	4	0	0	0	40	39	0
669	27	4	10	4	10	1	4	1	0	0	40	39	1
669	33	4	11	4	8	4	2	0	0	0	40	39	0
669	46	0	9	2	0	0	4	1	0	0	40	39	1
669	28	0	7	0	0	4	4	0	0	0	58	2	0
669	17	0	6	4	0	3	4	1	0	0	40	39	0
669	47	4	14	2	12	0	4	1	0	0	40	39	1
669	44	4	2	4	7	1	4	0	0	0	40	8	1
669	25	4	8	4	6	4	4	1	0	0	43	39	0
669	35	4	13	2	10	0	4	1	0	0	55	39	1
669	29	4	10	2	12	5	4	0	0	0	25	39	0
670	30	4	9	0	7	4	4	1	0	0	40	39	0
670	30	4	10	2	6	0	4	1	0	0	40	39	1
670	56	6	4	0	8	4	4	0	0	0	40	39	0
670	25	6	13	4	10	3	4	1	0	0	30	39	0
670	28	4	9	4	6	3	2	1	0	0	40	39	0
670	43	6	5	2	3	0	4	1	0	0	40	39	1
670	47	2	9	2	11	0	4	1	0	0	60	39	1
670	46	6	13	2	3	0	4	1	0	0	40	39	1

The output file contains the class label printed as 0 (<=50k)or 1 (>50k).

Users:

Prediction Model: An individual's income can be estimated using the dataset based on their demographic and employment characteristics. Using this method can be useful for financial institutions, marketing firms, and other organizations looking to target people based on their income level.

Researchers: Income levels can be studied using the dataset by examining demographic, education, and employment factors. It can be useful to researchers studying social inequality and economic mobility.

Policies: Government policies aimed at reducing poverty can be evaluated using the dataset. The information can be useful for policymakers and advocacy groups.

Public use: Individuals can use this application to determine which education and career choices will result in higher incomes. Educators, career counselors, and job placement agencies can benefit from this.

Recommendations:

The income gap between different categories has increased significantly in recent years. A variety of inputs are used to classify them into groups of <=50K or >50K. Many policies are recommended and people are instructed on how to enroll and ensure their future for those earning less than $50K. Citizens welfare will be better understood thereby. A reliable model for detecting overpayment can be built and used to test whether an individual is being overpaid or underpaid. Also, it can be used for marketing purposes, so specific advertisements can be sent to users according to their income level.

Instructions:

Below are the instructions to run the phase3 application:

1. Install the latest version of python3 and pip3
   To check the installed version of Python, open the terminal and enter: python3 --version
2. Use VS Code and open the folder phase 3. Open the terminal in the same path and type the below commands
   a. pip3 install virtualenv
   b. virtualenv env
   c. source env/bin/activate
3. To run application type: python3 app.py in the terminal.
   It displays the server at which the app is running which is generally http://127.0.0.1:5000
   Open http://127.0.0.1:5000 in any browser and you can see the application is up and running.

You have 2 options here:

1. Enter details manually and submit to get the prediction

2. Upload test file and submit, a Output.csv file gets created in phase3 folder. Open the file and see the predictions for various inputs

Name	**UBIT#**	**UBID**
Viditha Wudaru	50442670	vidithaw@buffalo.edu
Jayavani Akkiraju	50442863	jayavani@buffalo.edu

References:

1. https://github.com/jakerieger/FlaskIntroduction/tree/master/templates
2. https://github.com/pallets/flask/tree/main/tests/test_apps
3. https://www.w3schools.com/html/default.asp
4. https://flask.palletsprojects.com/en/2.3.x/patterns/wtforms/?highlight=request%20form%20get
5. https://flask.palletsprojects.com/en/2.3.x/quickstart/
6. https://www.w3schools.com/tags/att_input_size.asp
7. https://towardsdatascience.com/deploy-a-machine-learning-model-using-flask-da580f84e60c#:~:text=Pickle%20is%20used%20to%20serializing,be%20used%20by%20the%20server
8. https://medium.com/analytics-vidhya/deploy-machine-learning-model-using-flask-to-heroku-beginners-part-2-6db424211e3d
9. https://www.kaggle.com/discussions/getting-started/60835
10. https://www.w3docs.com/snippets/css/how-to-create-a-placeholder-for-an-html5-select-box-by-using-only-html-and-css.html
11. https://pythonbasics.org/read-csv-with-pandas/
12. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.to_numpy.html
13. https://flask.palletsprojects.com/en/2.3.x/patterns/fileuploads/
14. https://numpy.org/doc/stable/reference/generated/numpy.ndarray.astype.html
15. https://www.askpython.com/python/string/python-string-float-conversion
16. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html
17. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html
18. https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.bar.html
19. https://docs.python.org/3/library/io.html
20. https://docs.python.org/3/library/base64.html
21. https://www.tutorialspoint.com/matplotlib/matplotlib_bar_plot.htm

22. https://www.python-graph-gallery.com/3-control-color-of-barplots
23. https://www.w3resource.com/pandas/series/series-count.php#
24. https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.savefig.html
25. https://www.geeksforgeeks.org/python-seek-function/
26. https://www.brookings.edu/research/the-polarization-of-job-opportunities-in-the-u-s-labor-market-implications-for-employment-and-earnings/
27. https://sites.pitt.edu/~naraehan/python3/pickling.html