Python Flask Intro

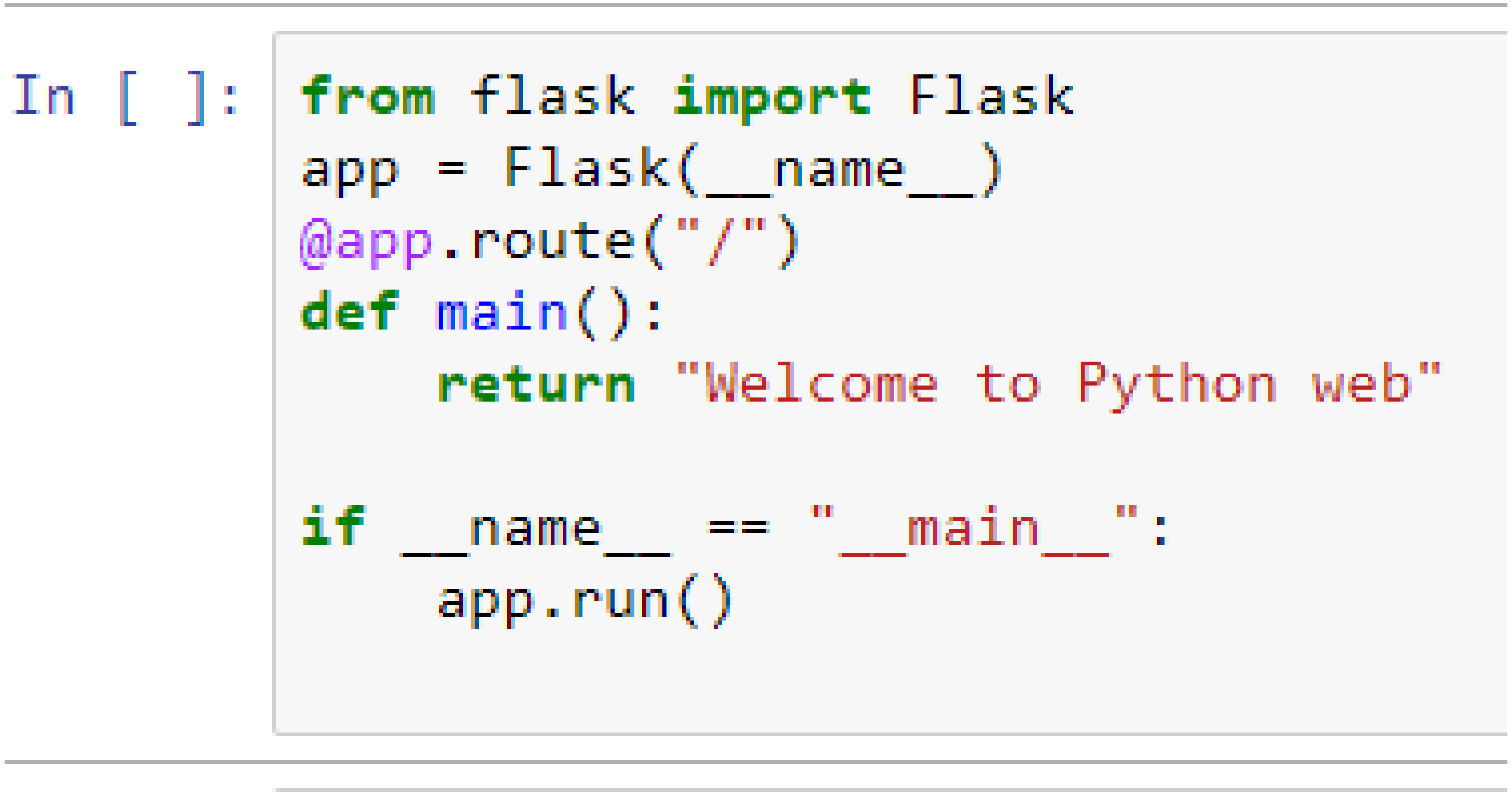
By JJ

Flask Framework

- All Flask applications need to create an application instance from class **Flask**.
- The Flask web server passes all client requests to this object.
- It uses a protocol called *Web Server Gateway Interface* (WSGI).
- Usually the form of creating this object:

```
from flask import Flask
app = Flask(__name__)
```

Create a simple Python script



```python
from flask import Flask
app = Flask(__name__)
@app.route("/")
def main():
    return "Welcome to Python web"

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

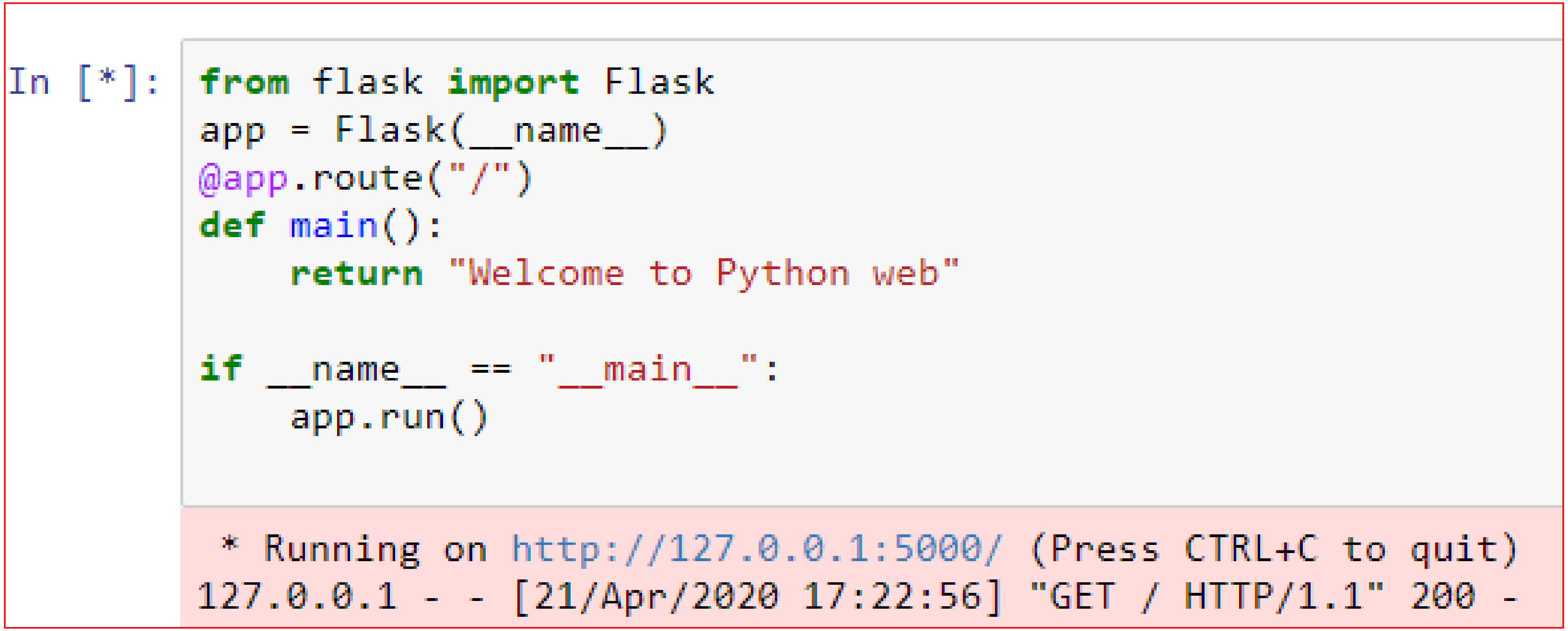
Flask constructor

- The only required argument to the Flask class constructor is the name of the main module or package of the application.
- For most applications, Python’s __name__ variable is the correct value as shown in previous slide
- As we know the __name__ == __main__   is to prevent the app to call its run() if this file was imported by other Python file.
- If we this file directly then app.run() will be executed

Flask route: mapping URL to Python function

- This application instance app keeps a **mapping** of **client URLs** to **Python functions**.
- Different URL maps to different Python functions.
- In Flask the mapping is called a *route*.
- The previous slide the website root (/) will map to Python function main()
- We can create other mappings by creating other functions and URL patterns.

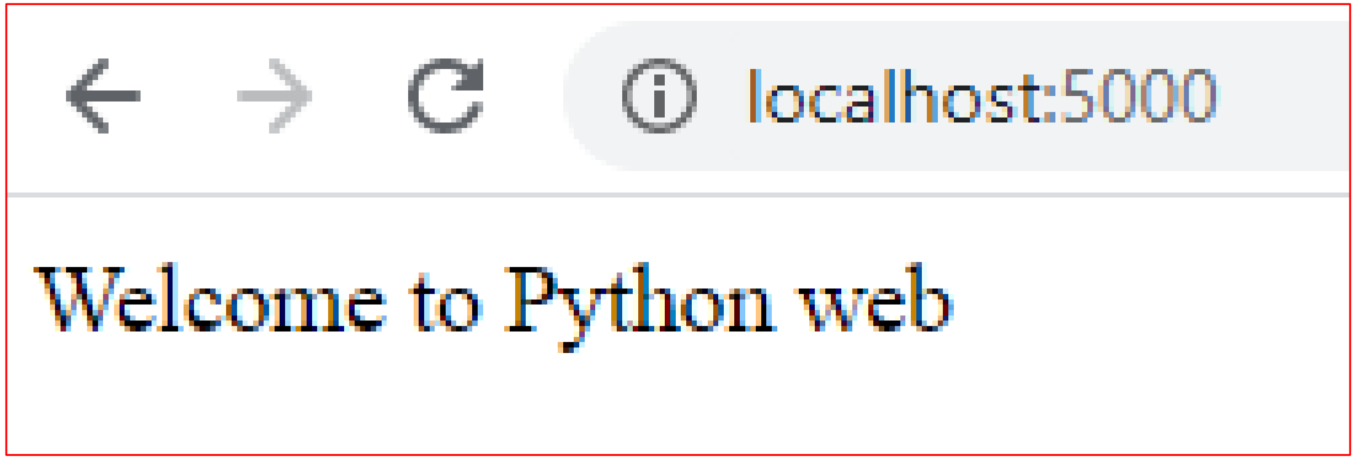
Execute the script and run Flask web server on port 5000 on Jupyter notebook



```
from flask import Flask
app = Flask(__name__)
@app.route("/")
def main():
    return "Welcome to Python web"

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

```
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [21/Apr/2020 17:22:56] "GET / HTTP/1.1" 200 -
```

Flask is now running on localhost at port 5000

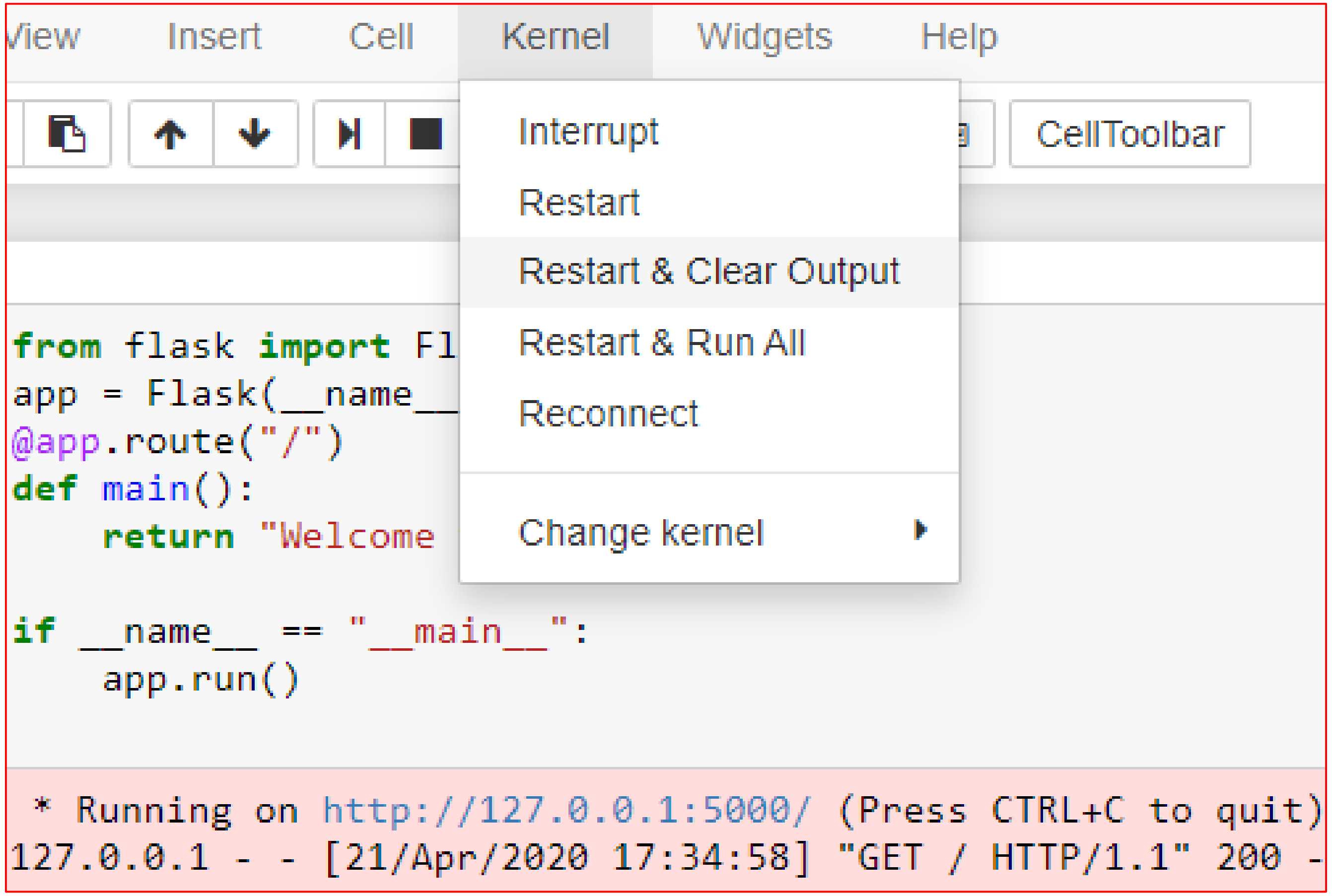


Welcome to Python web

Port number in Flask app

- You can specify the port to different port number note for this time port 0-1023 are reserved. So don’t use.
- Say we run app.run(debug=True, port=3000). Then the port number on user’s url shall be localhost:3000

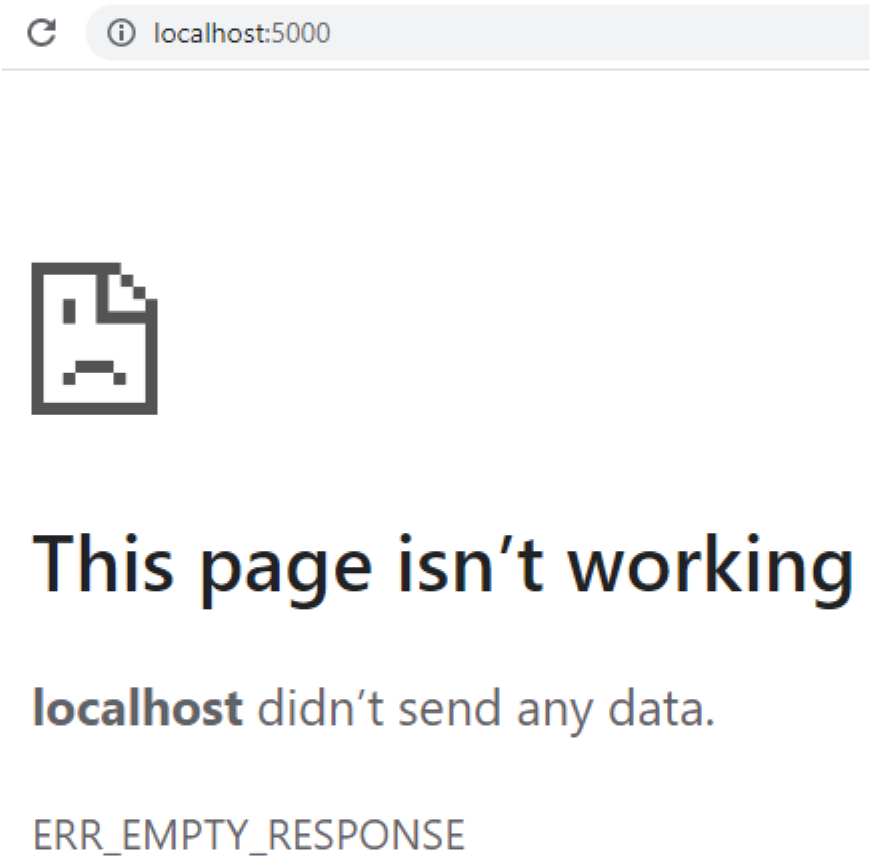
Clear and restart Kernel to stop the server on Jupyter



```
from flask import Fl
app = Flask(__name__
@app.route("/")
def main():
    return "Welcome

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

```
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [21/Apr/2020 17:34:58] "GET / HTTP/1.1" 200 -
```

After stop the server

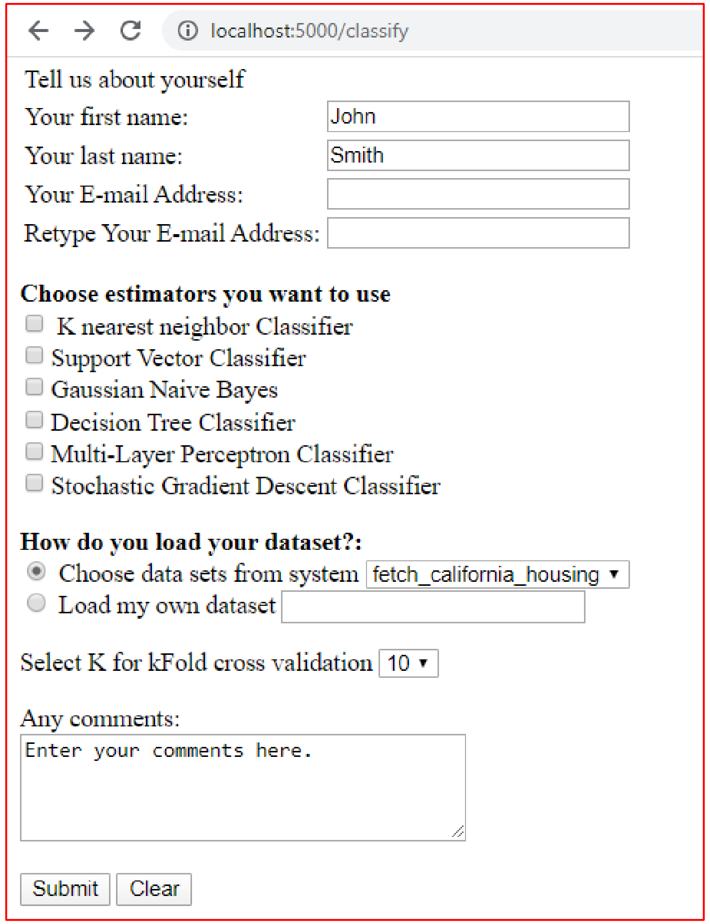


This page isn’t working

localhost didn’t send any data.

ERR_EMPTY_RESPONSE

Assuming we have form like the following



Tell us about yourself

Your first name:	John
Your last name:	Smith
Your E-mail Address:	
Retype Your E-mail Address:	

Choose estimators you want to use

□ K nearest neighbor Classifier
□ Support Vector Classifier
□ Gaussian Naive Bayes
□ Decision Tree Classifier
□ Multi-Layer Perceptron Classifier
□ Stochastic Gradient Descent Classifier

How do you load your dataset?:

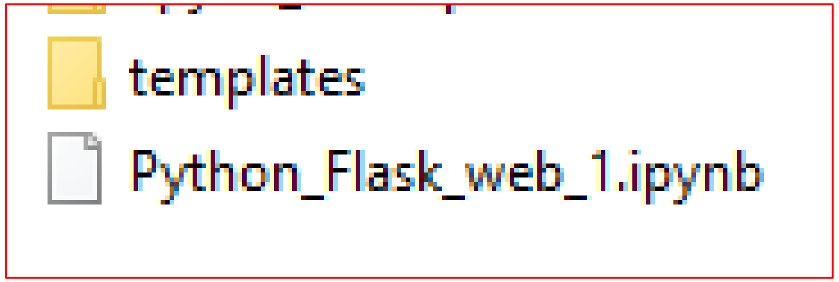
● Choose data sets from system fetch_california_housing ▾
○ Load my own dataset

Select K for kFold cross validation 10 ▾

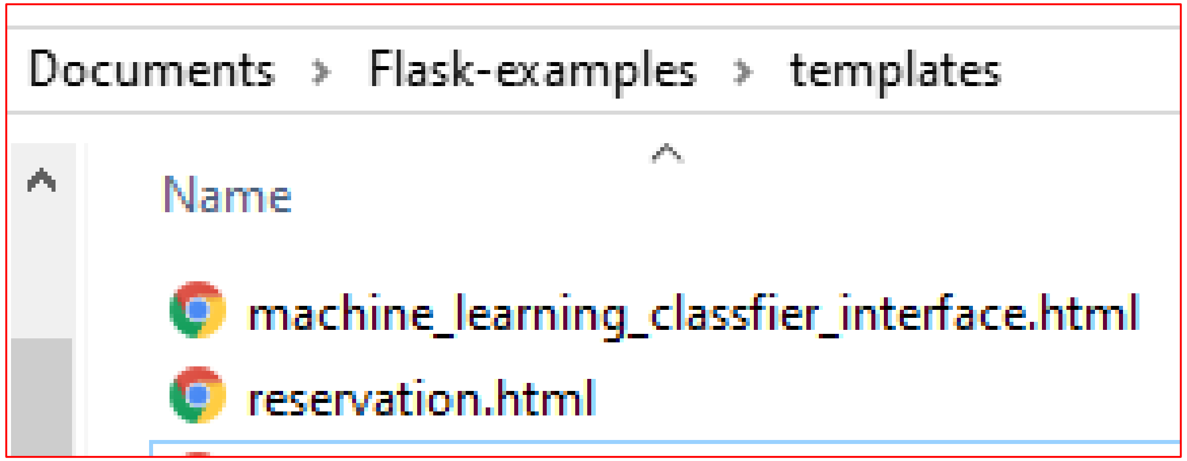
Any comments:

```
Enter your comments here.
```

Let’s create a sub-folder call templates and put this form into this folder. Python_Flask_web_1 notebook contains the Flask app we created



templates
Python_Flask_web_1.ipynb



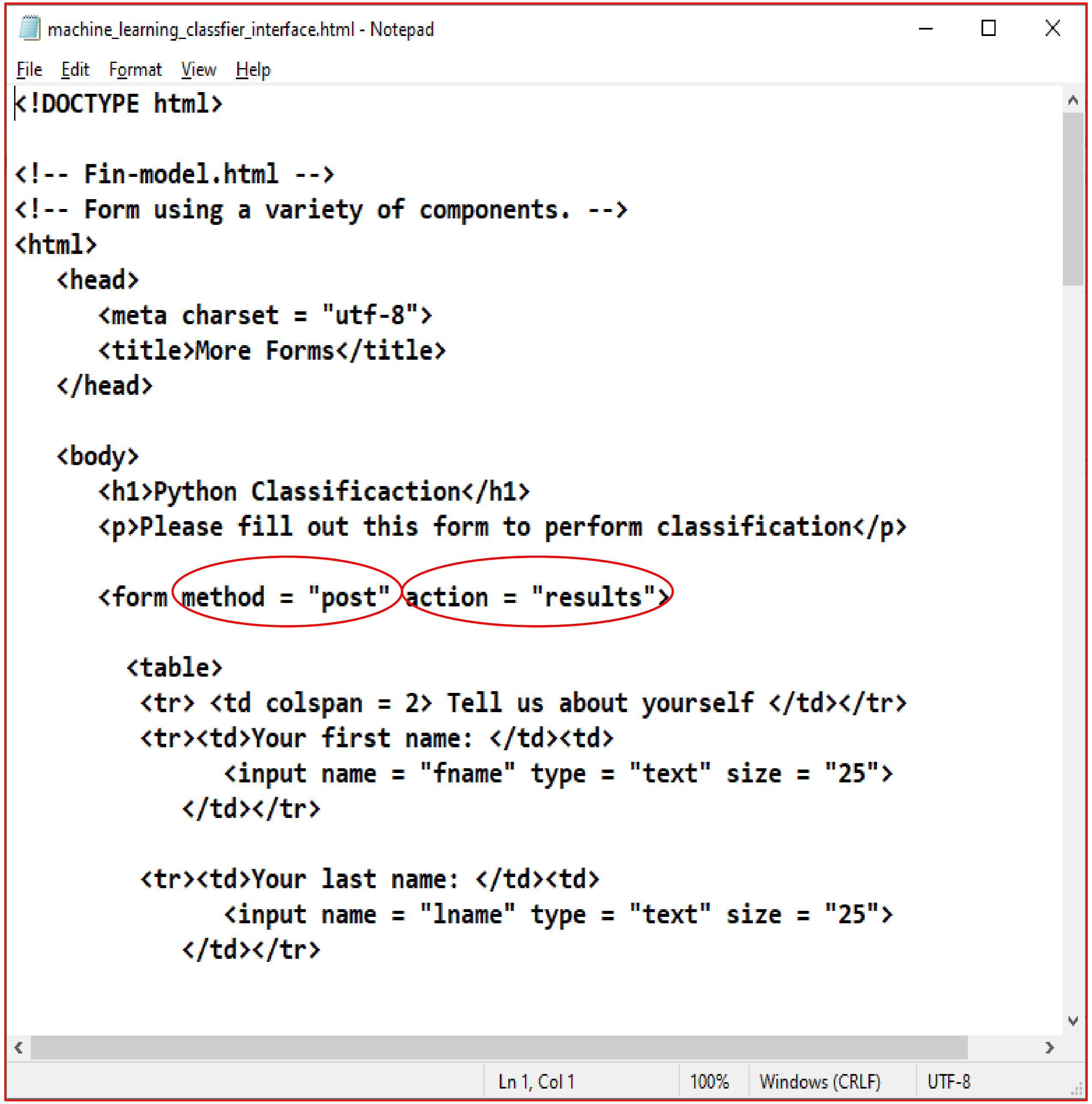
Documents › Flask-examples › templates

Name

machine_learning_classfier_interface.html
reservation.html

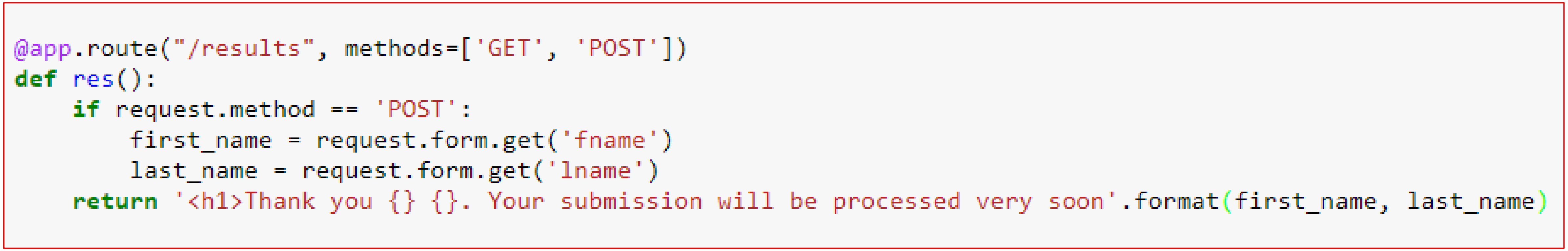
The HTML file content.

Method POST is used and action is results



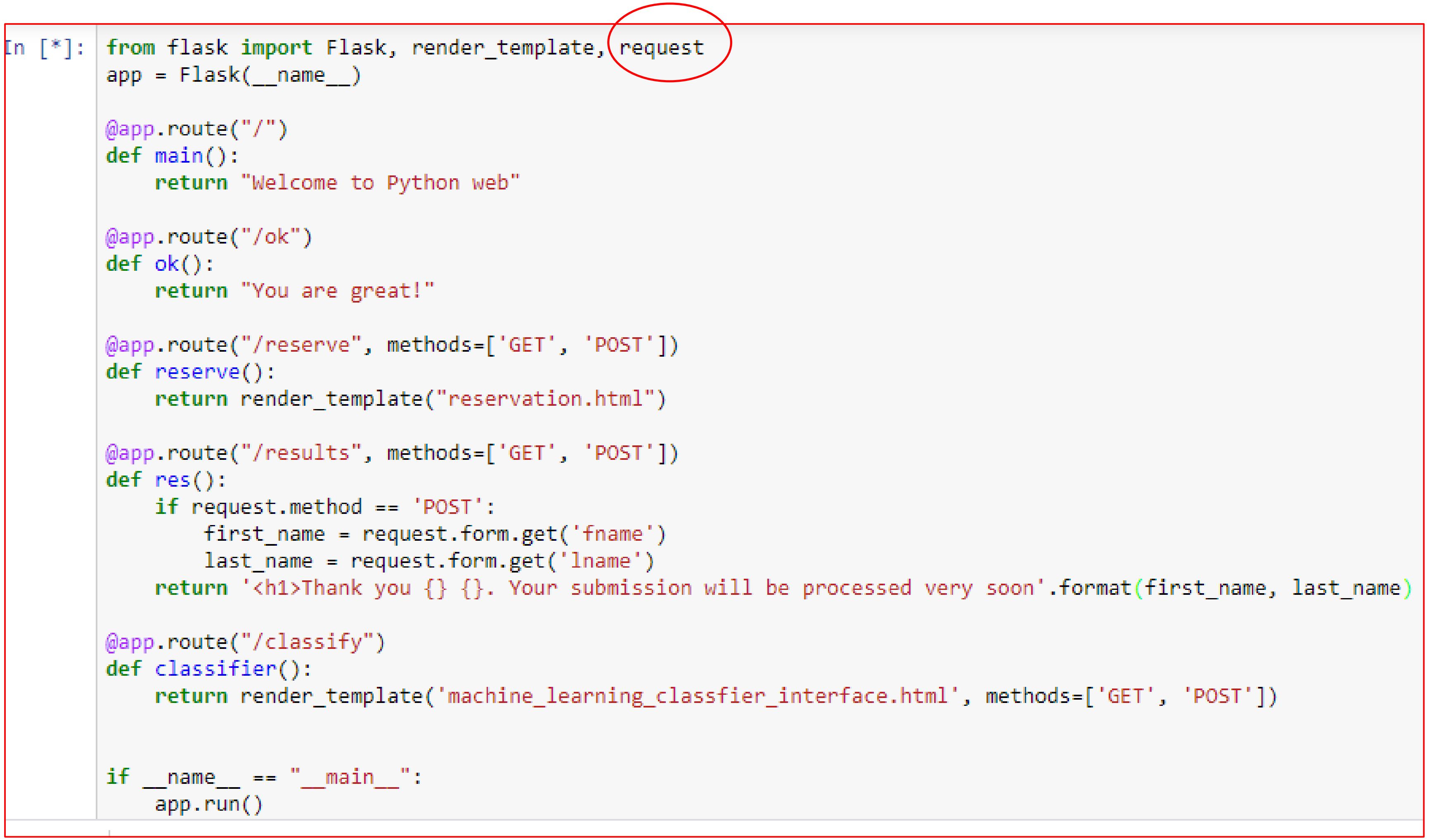
```
<!DOCTYPE html>

<!-- Fin-model.html -->
<!-- Form using a variety of components. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>More Forms</title>
   </head>

   <body>
      <h1>Python Classificaction</h1>
      <p>Please fill out this form to perform classification</p>

      <form method = "post" action = "results">

        <table>
         <tr> <td colspan = 2> Tell us about yourself </td></tr>
         <tr><td>Your first name: </td><td>
               <input name = "fname" type = "text" size = "25">
            </td></tr>

         <tr><td>Your last name: </td><td>
               <input name = "lname" type = "text" size = "25">
            </td></tr>
```

We map the url pattern results as the following (we only process first and last name)



```python
@app.route("/results", methods=['GET', 'POST'])
def res():
    if request.method == 'POST':
        first_name = request.form.get('fname')
        last_name = request.form.get('lname')
    return '<h1>Thank you {} {}. Your submission will be processed very soon'.format(first_name, last_name)
```

Since we retrieve request data so we also import *request* module

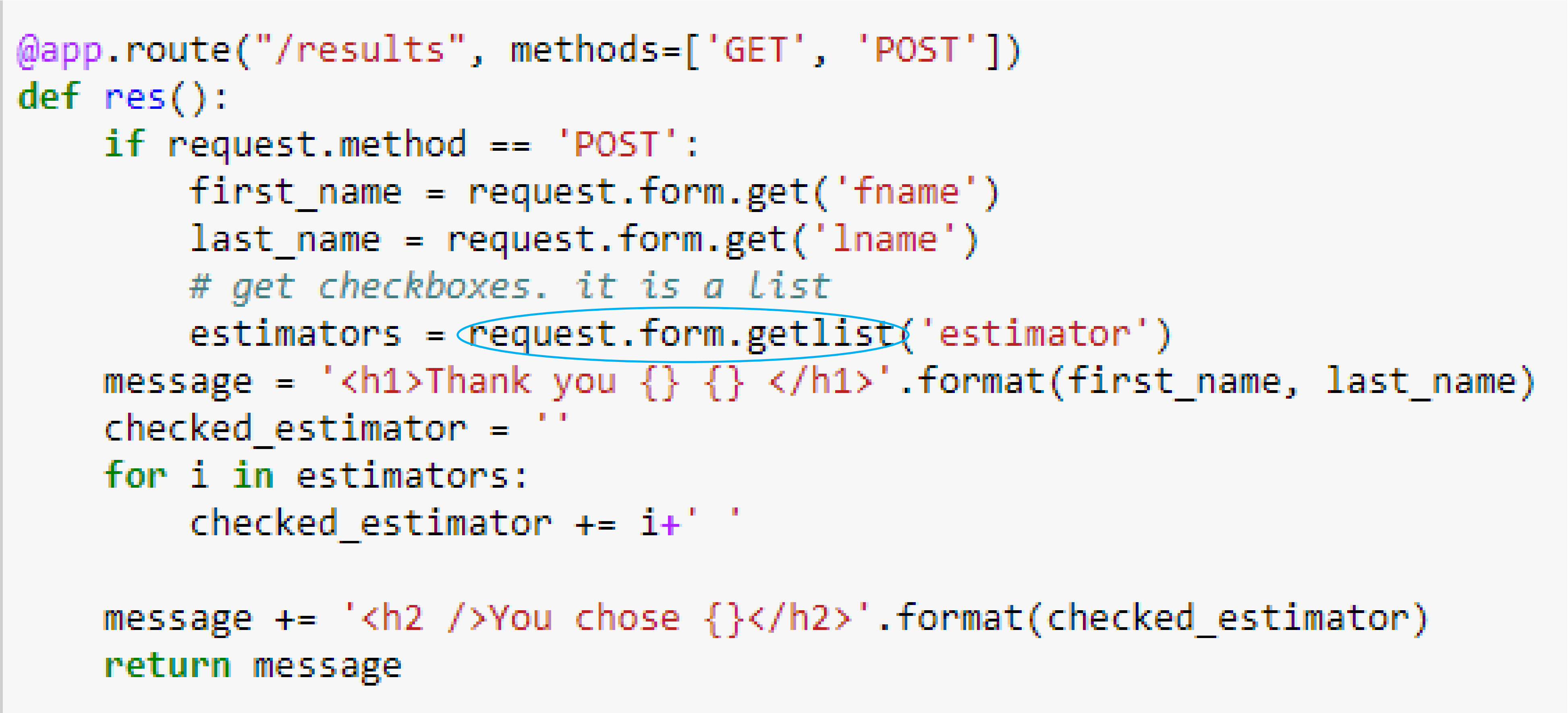


```
In [*]: from flask import Flask, render_template, request
        app = Flask(__name__)

        @app.route("/")
        def main():
            return "Welcome to Python web"

        @app.route("/ok")
        def ok():
            return "You are great!"

        @app.route("/reserve", methods=['GET', 'POST'])
        def reserve():
            return render_template("reservation.html")

        @app.route("/results", methods=['GET', 'POST'])
        def res():
            if request.method == 'POST':
                first_name = request.form.get('fname')
                last_name = request.form.get('lname')
            return '<h1>Thank you {} {}. Your submission will be processed very soon'.format(first_name, last_name)

        @app.route("/classify")
        def classifier():
            return render_template('machine_learning_classfier_interface.html', methods=['GET', 'POST'])


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

How do we retrieve data from client?

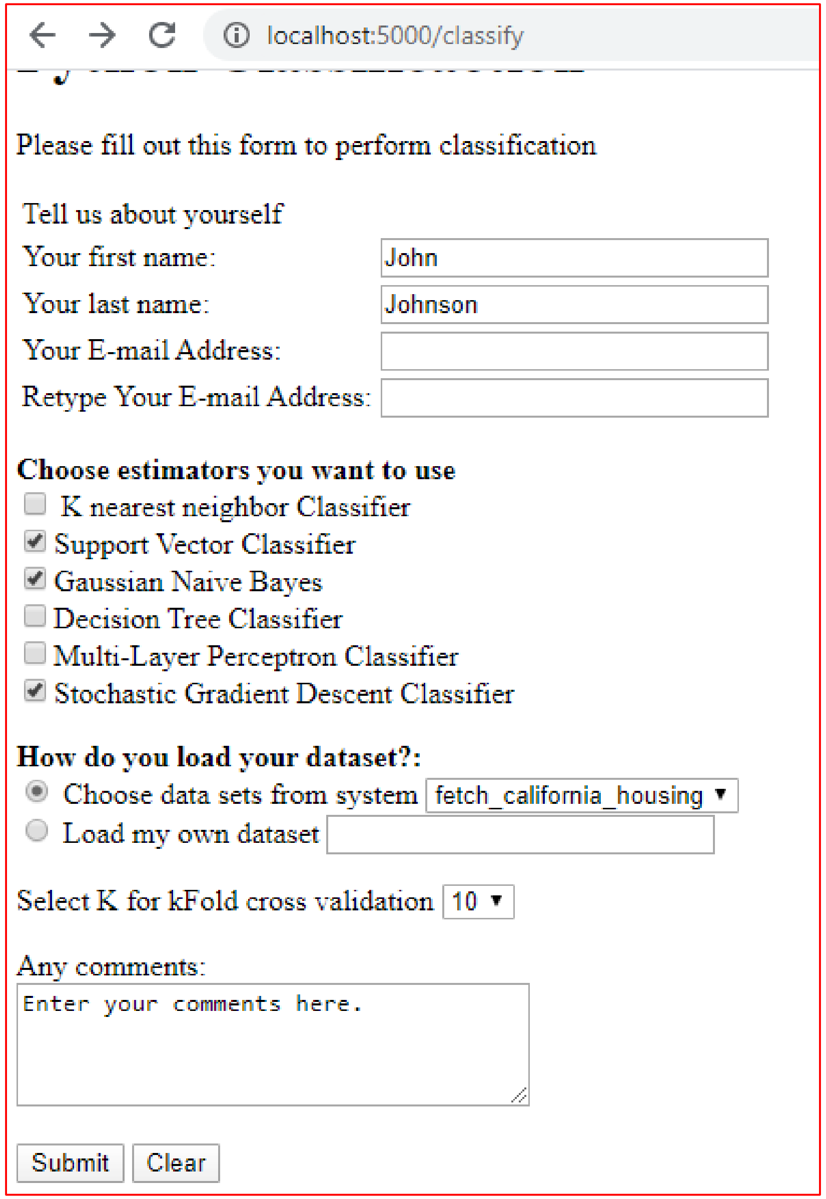
- request.args: the key/value pairs in *the URL query string* (after ?)
- request.form: the key/value pairs in the body, from a HTML post form, or JavaScript request that isn't JSON encoded
- request.files: the files in the body, which Flask keeps separate from form. HTML forms must use enctype=multipart/form-data or files will not be uploaded.
- request.values: combined args and form, preferring args if keys overlap
- request.json: parsed JSON data. The request must have the application/json content type, or use request.get_json(force=True) to ignore the content type.

Let’s modify it a little bit to handle check list by using getlist()



```python
@app.route("/results", methods=['GET', 'POST'])
def res():
    if request.method == 'POST':
        first_name = request.form.get('fname')
        last_name = request.form.get('lname')
        # get checkboxes. it is a list
        estimators = request.form.getlist('estimator')
    message = '<h1>Thank you {} {} </h1>'.format(first_name, last_name)
    checked_estimator = ''
    for i in estimators:
        checked_estimator += i+' '

    message += '<h2 />You chose {}</h2>'.format(checked_estimator)
    return message
```

Fill in first and last names and check some estimators



localhost:5000/classify

Please fill out this form to perform classification

Tell us about yourself

Your first name: John

Your last name: Johnson

Your E-mail Address:

Retype Your E-mail Address:

Choose estimators you want to use

- □ K nearest neighbor Classifier
- ☑ Support Vector Classifier
- ☑ Gaussian Naive Bayes
- □ Decision Tree Classifier
- □ Multi-Layer Perceptron Classifier
- ☑ Stochastic Gradient Descent Classifier

How do you load your dataset?:

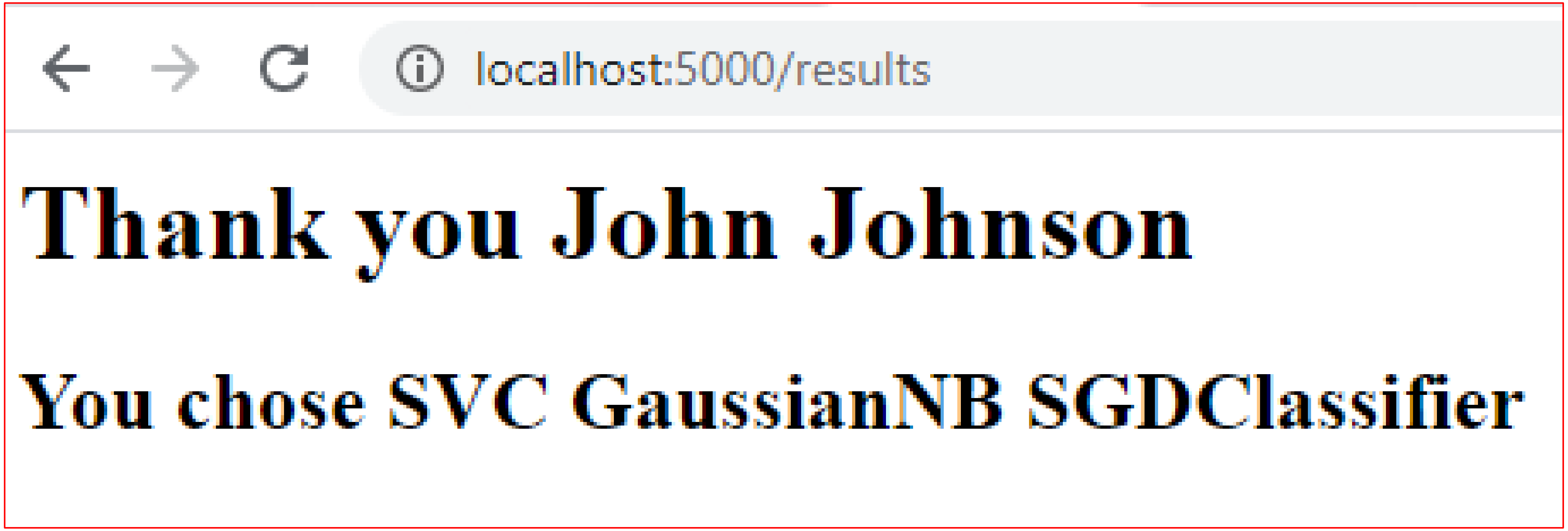
- ● Choose data sets from system fetch_california_housing ▾
- ○ Load my own dataset

Select K for kFold cross validation 10 ▾

Any comments:

Enter your comments here.

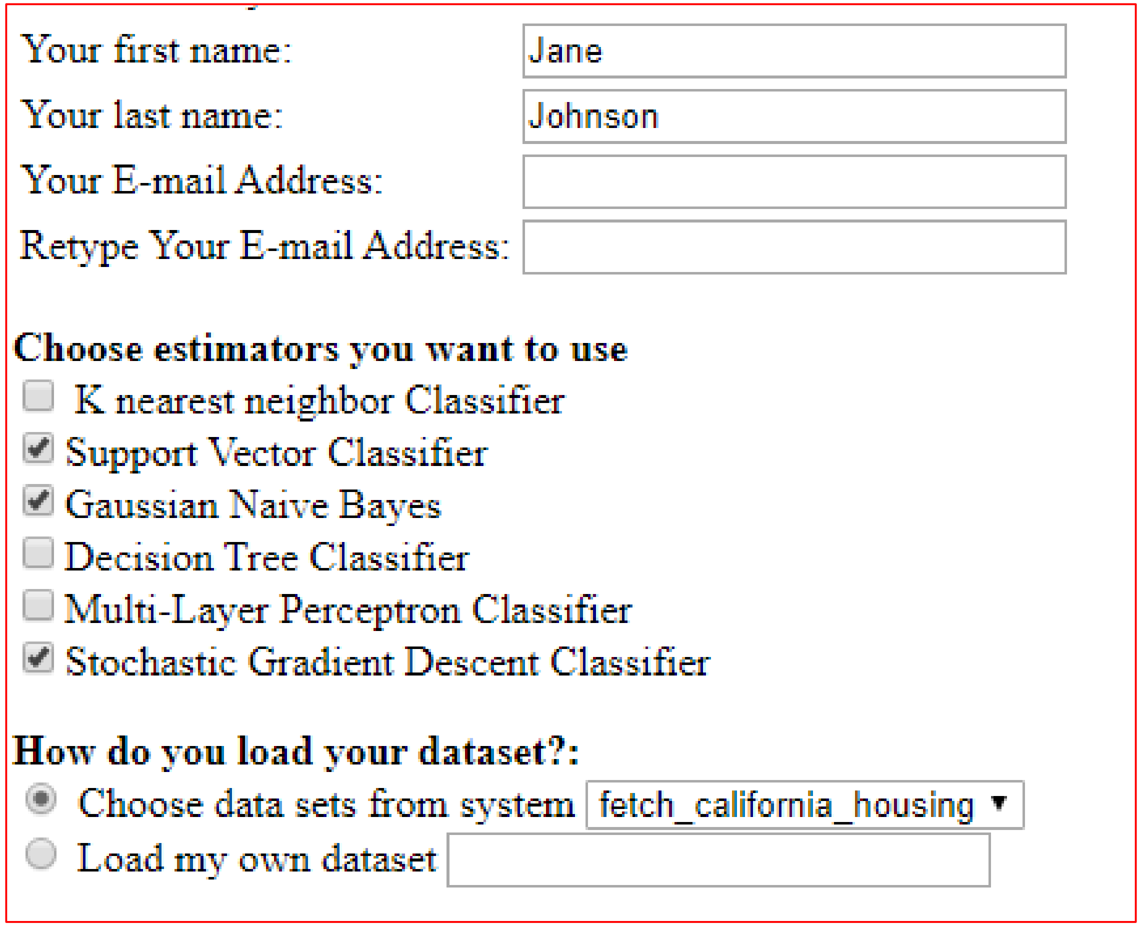
Output



Thank you John Johnson

You chose SVC GaussianNB SGDClassifier

Now let’s handle radio button



Your first name: Jane

Your last name: Johnson

Your E-mail Address:

Retype Your E-mail Address:

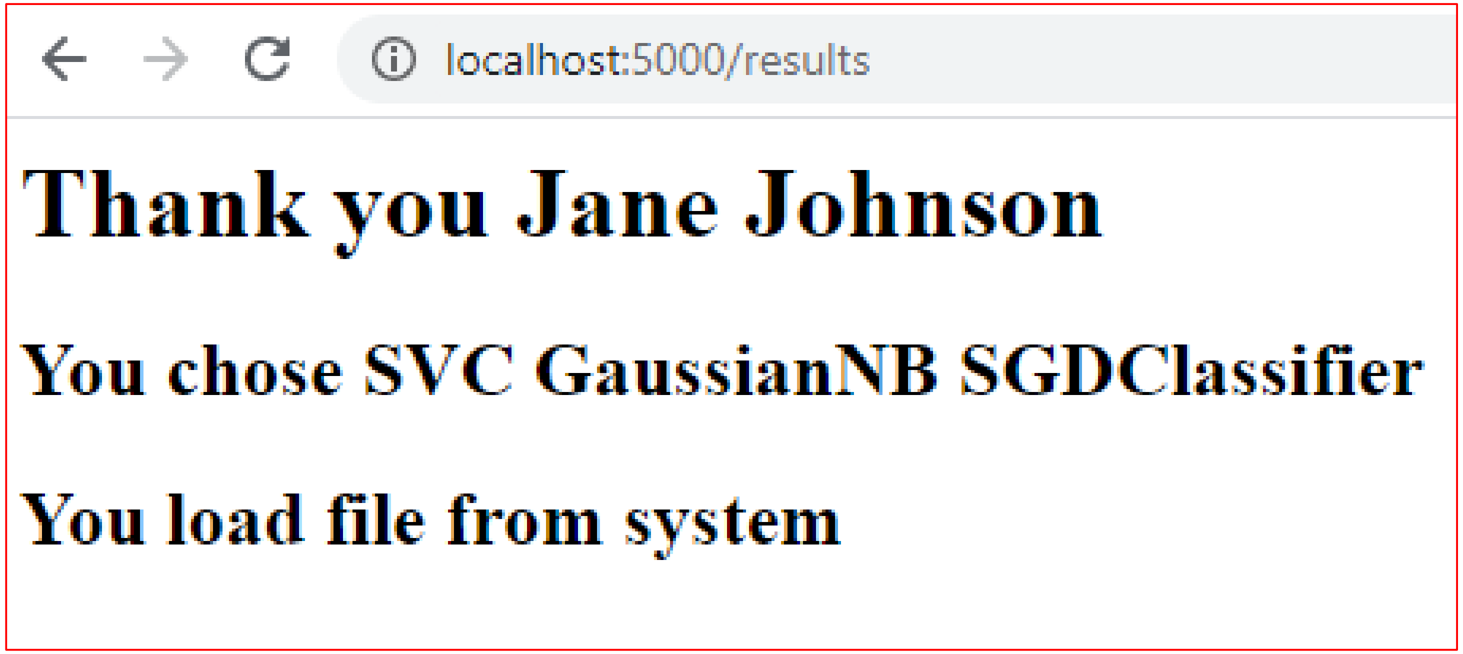
Choose estimators you want to use

☐ K nearest neighbor Classifier
☑ Support Vector Classifier
☑ Gaussian Naive Bayes
☐ Decision Tree Classifier
☐ Multi-Layer Perceptron Classifier
☑ Stochastic Gradient Descent Classifier

How do you load your dataset?:

● Choose data sets from system fetch_california_housing ▼
○ Load my own dataset

Output

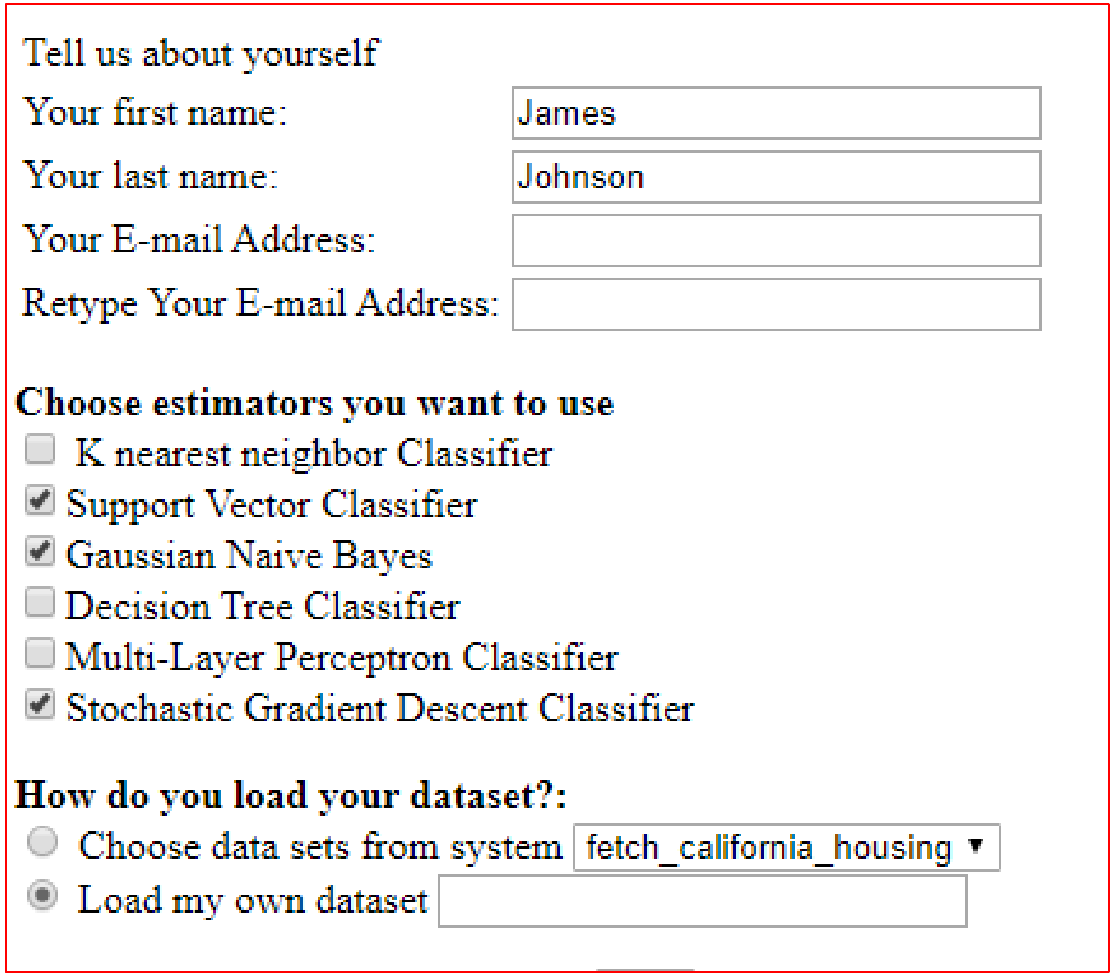


Thank you Jane Johnson

You chose SVC GaussianNB SGDClassifier

You load file from system

Click Load my own dataset



Tell us about yourself

Your first name:	James
Your last name:	Johnson
Your E-mail Address:	
Retype Your E-mail Address:	

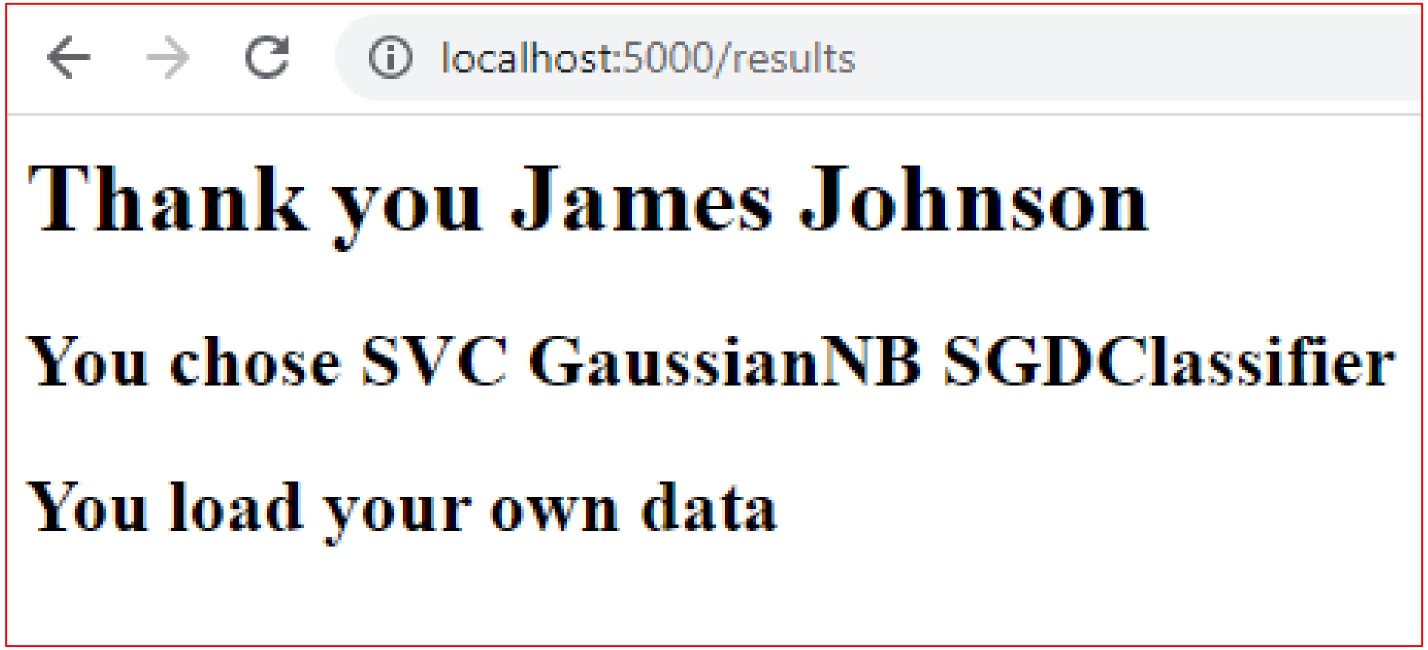
Choose estimators you want to use

- ☐ K nearest neighbor Classifier
- ☑ Support Vector Classifier
- ☑ Gaussian Naive Bayes
- ☐ Decision Tree Classifier
- ☐ Multi-Layer Perceptron Classifier
- ☑ Stochastic Gradient Descent Classifier

How do you load your dataset?:

- ○ Choose data sets from system fetch_california_housing ▼
- ● Load my own dataset

Output

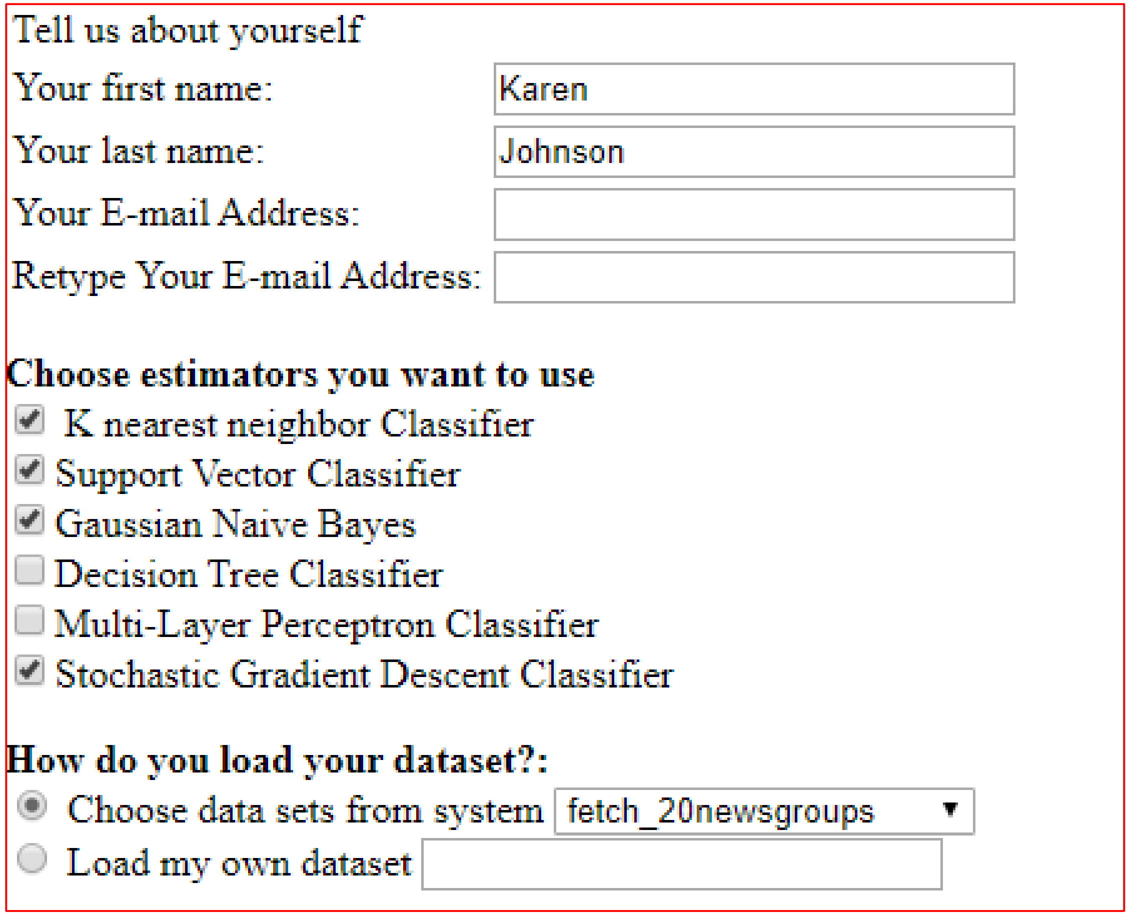


Thank you James Johnson

You chose SVC GaussianNB SGDClassifier

You load your own data

Let’s handle drop down list



Tell us about yourself

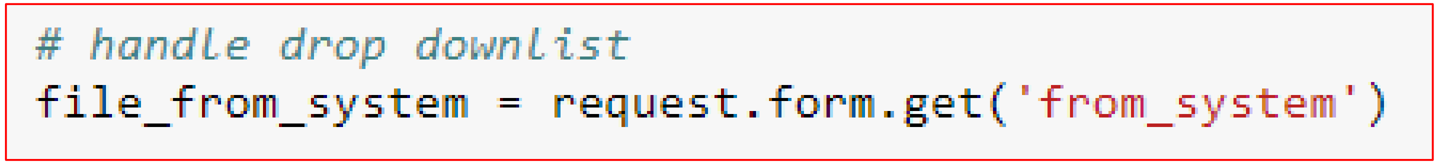
Your first name:	Karen
Your last name:	Johnson
Your E-mail Address:	
Retype Your E-mail Address:	

Choose estimators you want to use

☑ K nearest neighbor Classifier
☑ Support Vector Classifier
☑ Gaussian Naive Bayes
□ Decision Tree Classifier
□ Multi-Layer Perceptron Classifier
☑ Stochastic Gradient Descent Classifier

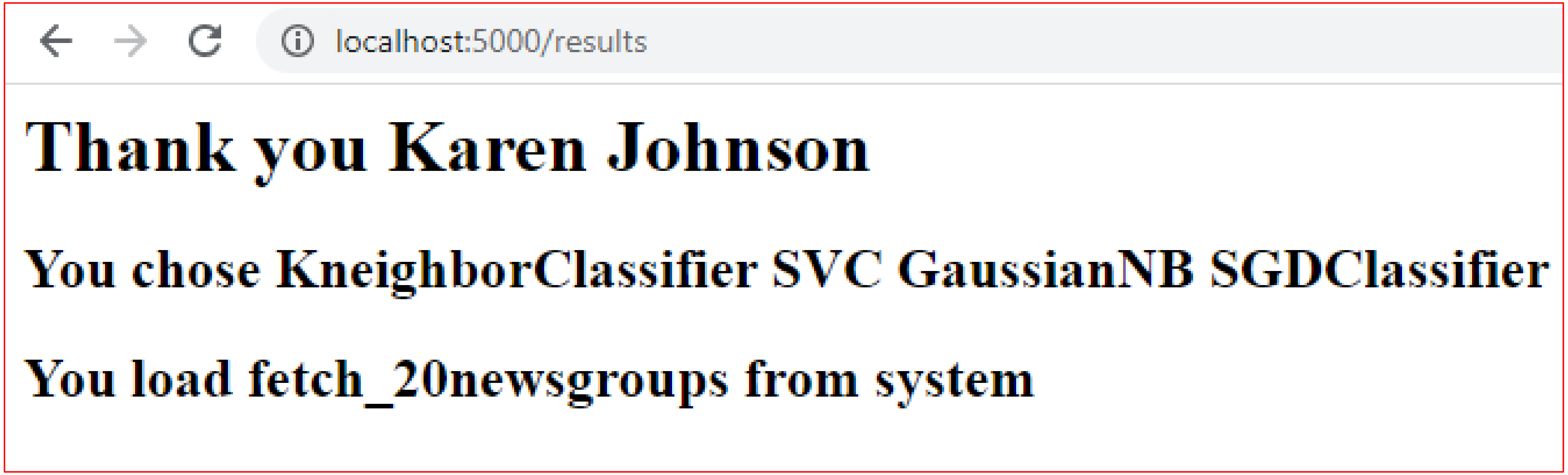
How do you load your dataset?:

● Choose data sets from system fetch_20newsgroups ▼
○ Load my own dataset



```
# handle drop downlist
file_from_system = request.form.get('from_system')
```

Output



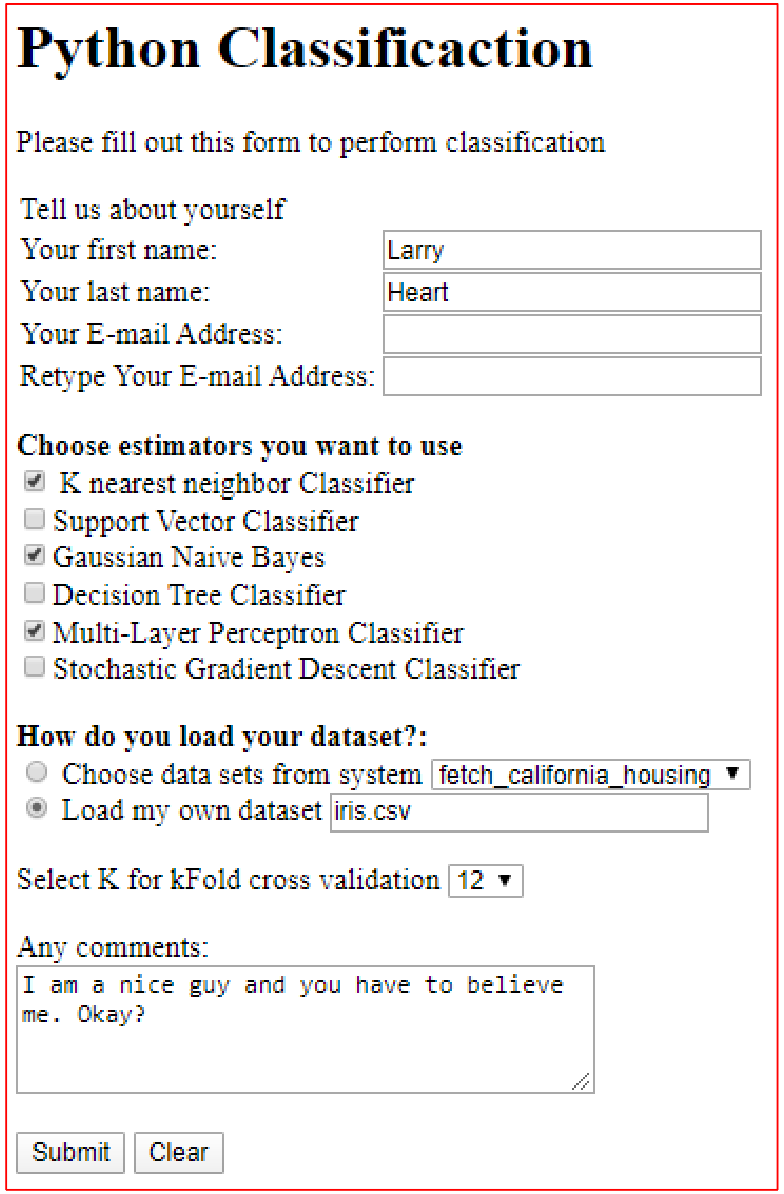
Thank you Karen Johnson

You chose KneighborClassifier SVC GaussianNB SGDClassifier

You load fetch_20newsgroups from system

Handle textarea

```python
# get comments
comments = request.form.get('comments')
```



Python Classificaction

Please fill out this form to perform classification

Tell us about yourself

Your first name:	Larry
Your last name:	Heart
Your E-mail Address:	
Retype Your E-mail Address:	

Choose estimators you want to use

☑ K nearest neighbor Classifier
□ Support Vector Classifier
☑ Gaussian Naive Bayes
□ Decision Tree Classifier
☑ Multi-Layer Perceptron Classifier
□ Stochastic Gradient Descent Classifier

How do you load your dataset?:

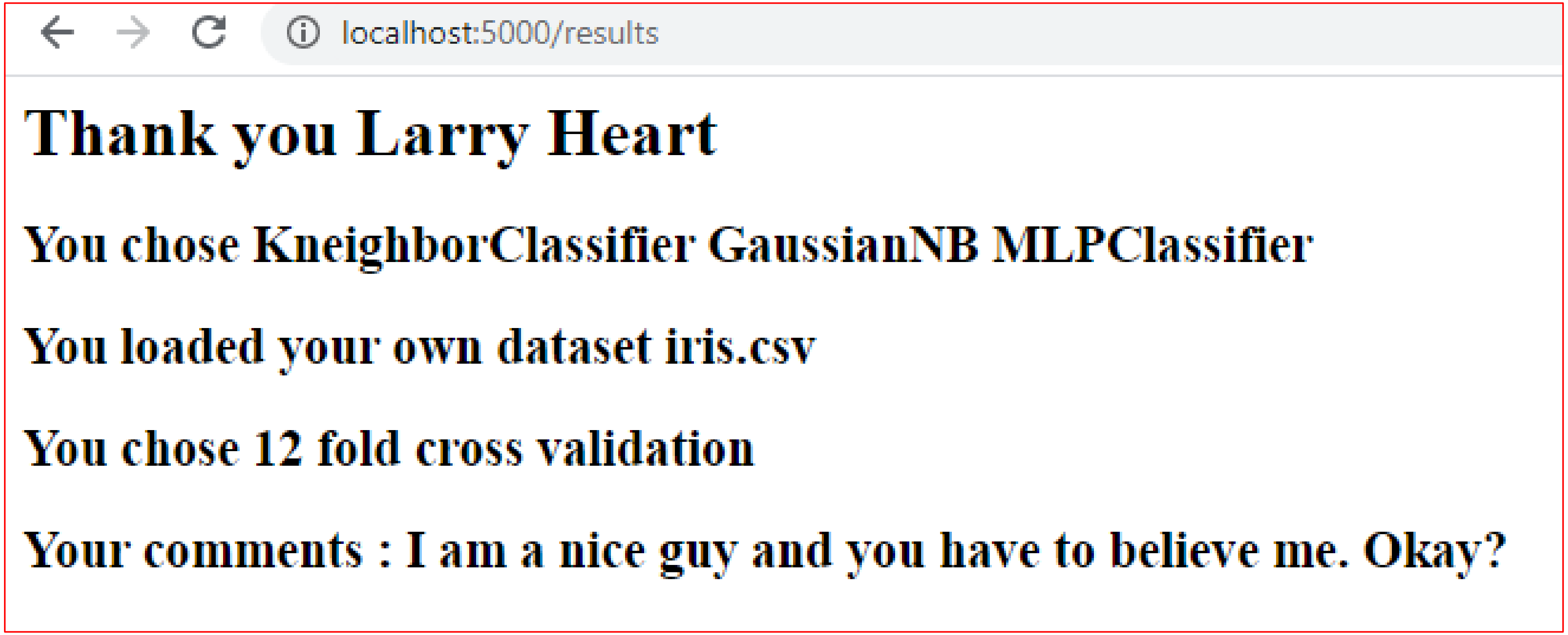
○ Choose data sets from system fetch_california_housing ▼
● Load my own dataset iris.csv

Select K for kFold cross validation 12 ▼

Any comments:

```
I am a nice guy and you have to believe
me. Okay?
```

The output



Thank you Larry Heart

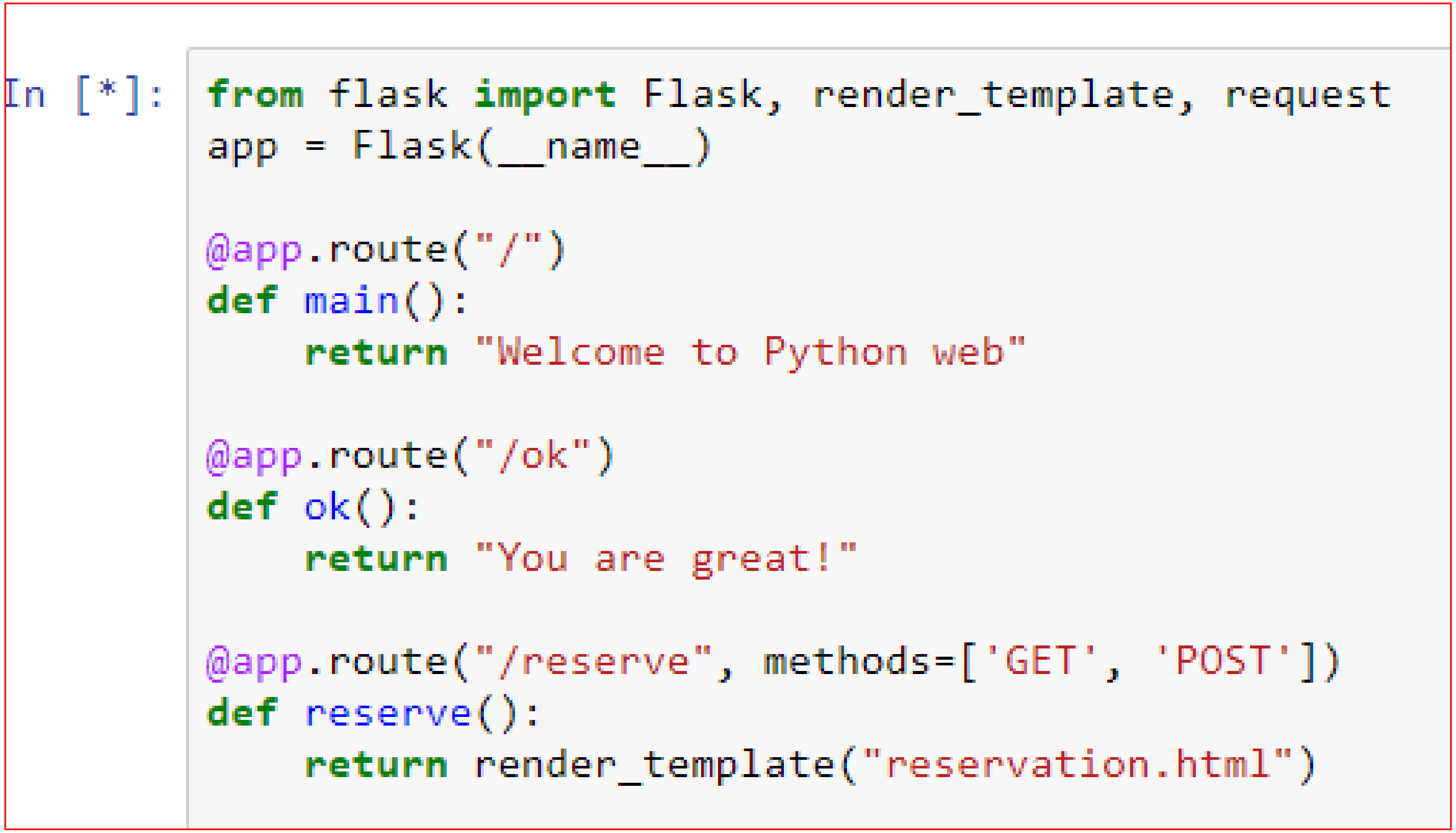
You chose KneighborClassifier GaussianNB MLPClassifier

You loaded your own dataset iris.csv

You chose 12 fold cross validation

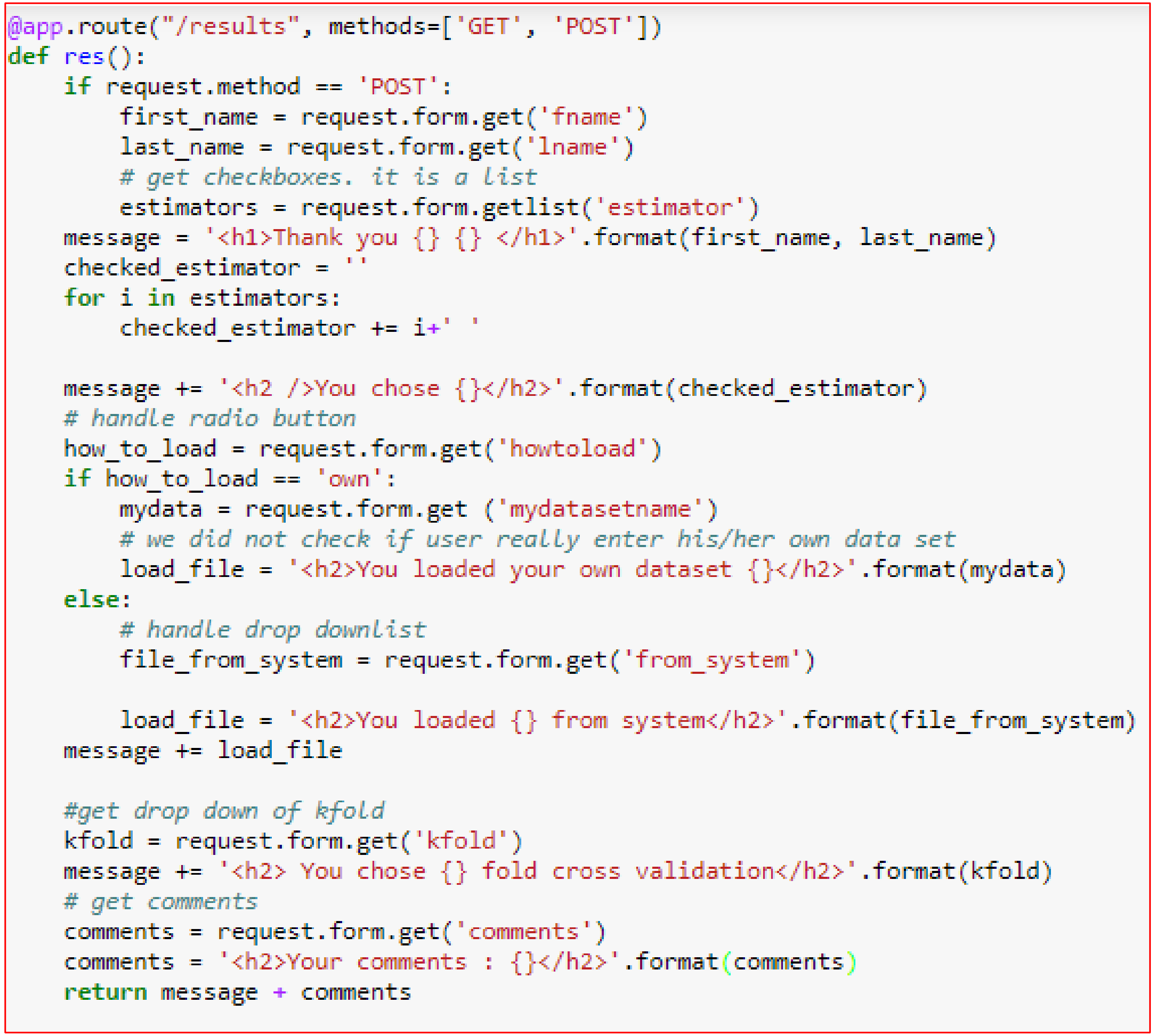
Your comments : I am a nice guy and you have to believe me. Okay?

Final Python Source code



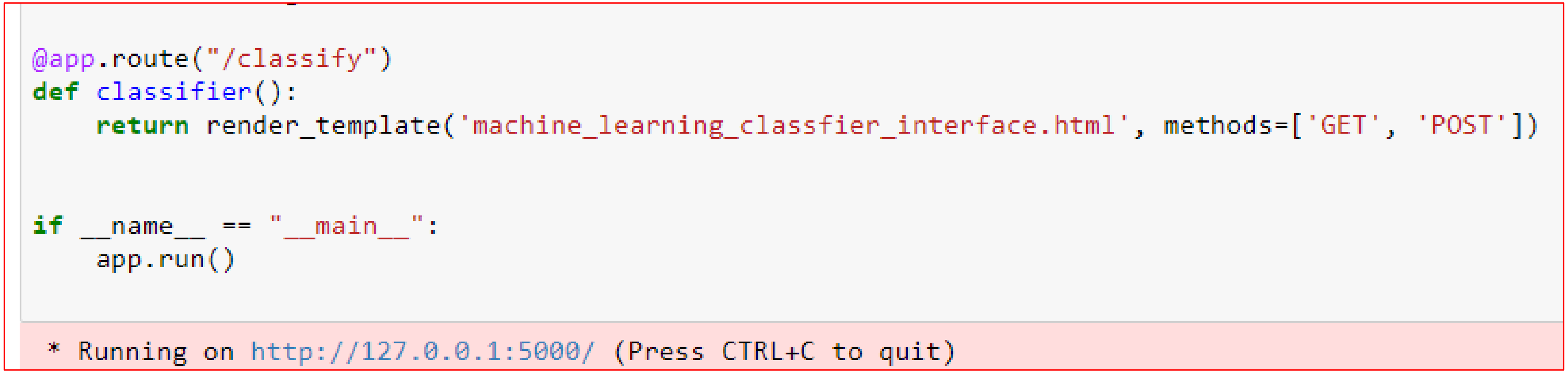
```
from flask import Flask, render_template, request
app = Flask(__name__)

@app.route("/")
def main():
    return "Welcome to Python web"

@app.route("/ok")
def ok():
    return "You are great!"

@app.route("/reserve", methods=['GET', 'POST'])
def reserve():
    return render_template("reservation.html")
```

Final Python Source code to handle post request



```
@app.route("/results", methods=['GET', 'POST'])
def res():
    if request.method == 'POST':
        first_name = request.form.get('fname')
        last_name = request.form.get('lname')
        # get checkboxes. it is a list
        estimators = request.form.getlist('estimator')
    message = '<h1>Thank you {} {} </h1>'.format(first_name, last_name)
    checked_estimator = ''
    for i in estimators:
        checked_estimator += i+' '

    message += '<h2 />You chose {}</h2>'.format(checked_estimator)
    # handle radio button
    how_to_load = request.form.get('howtoload')
    if how_to_load == 'own':
        mydata = request.form.get ('mydatasetname')
        # we did not check if user really enter his/her own data set
        load_file = '<h2>You loaded your own dataset {}</h2>'.format(mydata)
    else:
        # handle drop downlist
        file_from_system = request.form.get('from_system')

        load_file = '<h2>You loaded {} from system</h2>'.format(file_from_system)
    message += load_file

    #get drop down of kfold
    kfold = request.form.get('kfold')
    message += '<h2> You chose {} fold cross validation</h2>'.format(kfold)
    # get comments
    comments = request.form.get('comments')
    comments = '<h2>Your comments : {}</h2>'.format(comments)
    return message + comments
```

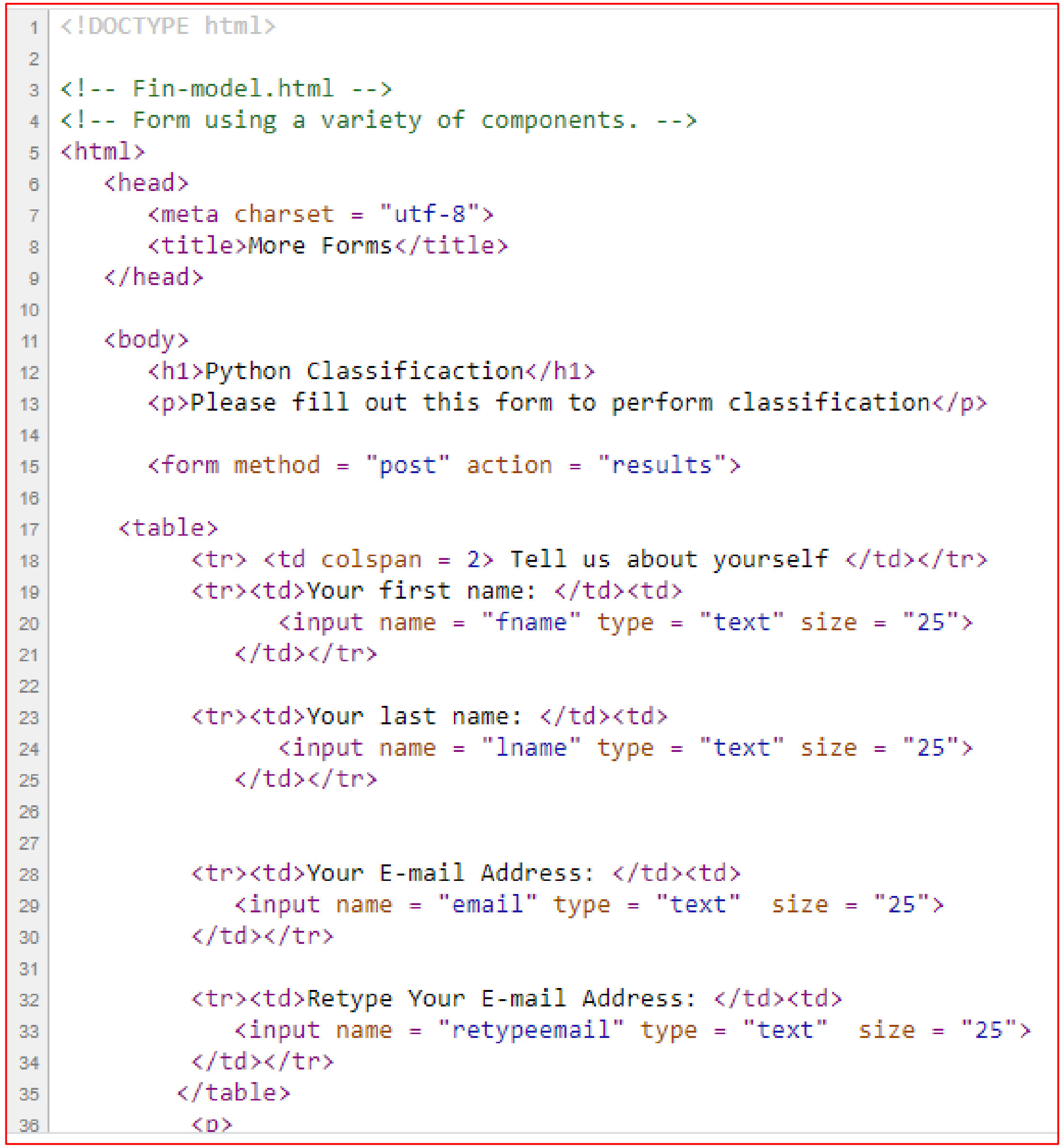
Python source code



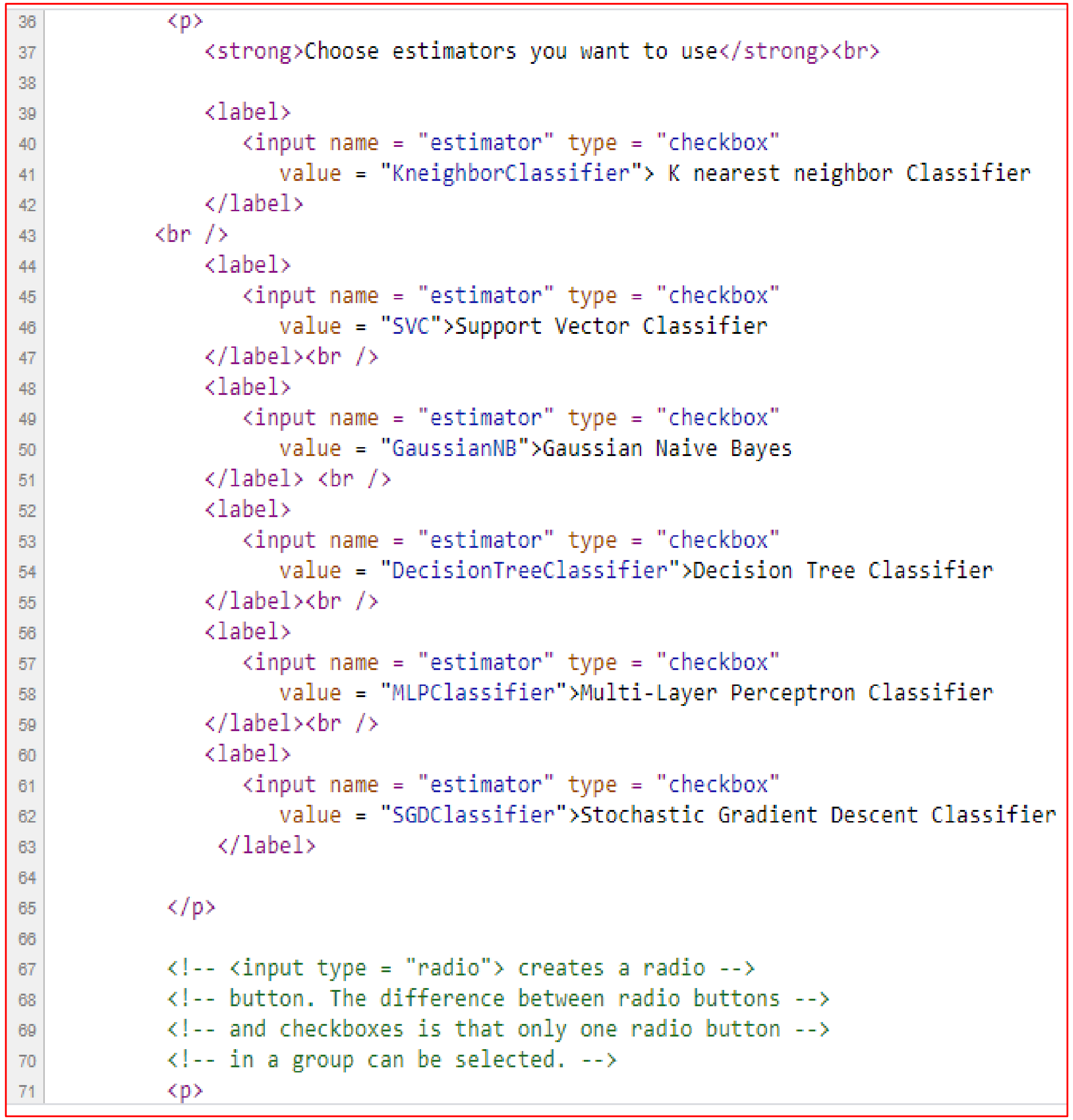
```
@app.route("/classify")
def classifier():
    return render_template('machine_learning_classfier_interface.html', methods=['GET', 'POST'])


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

```
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

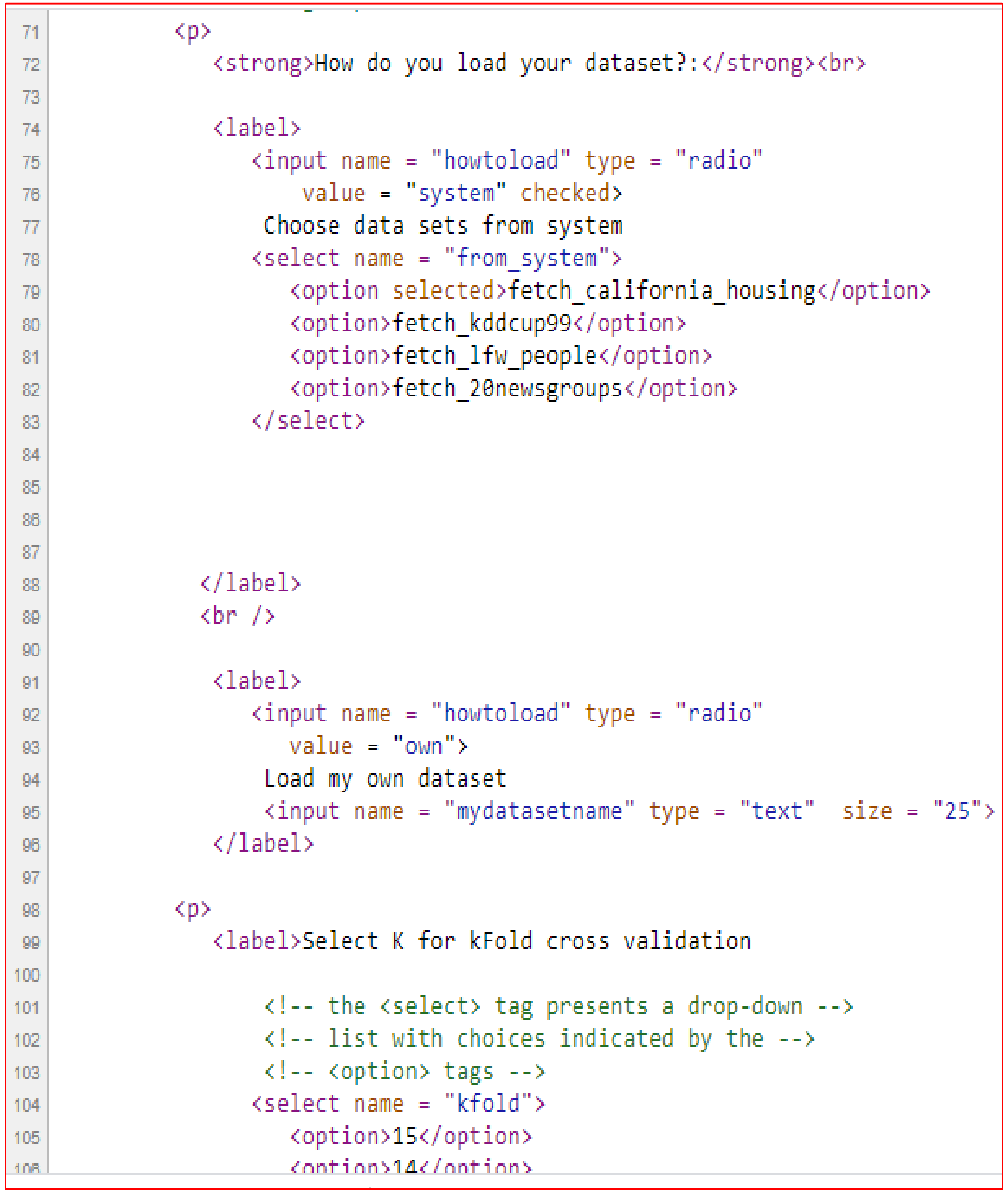
HTML Source code



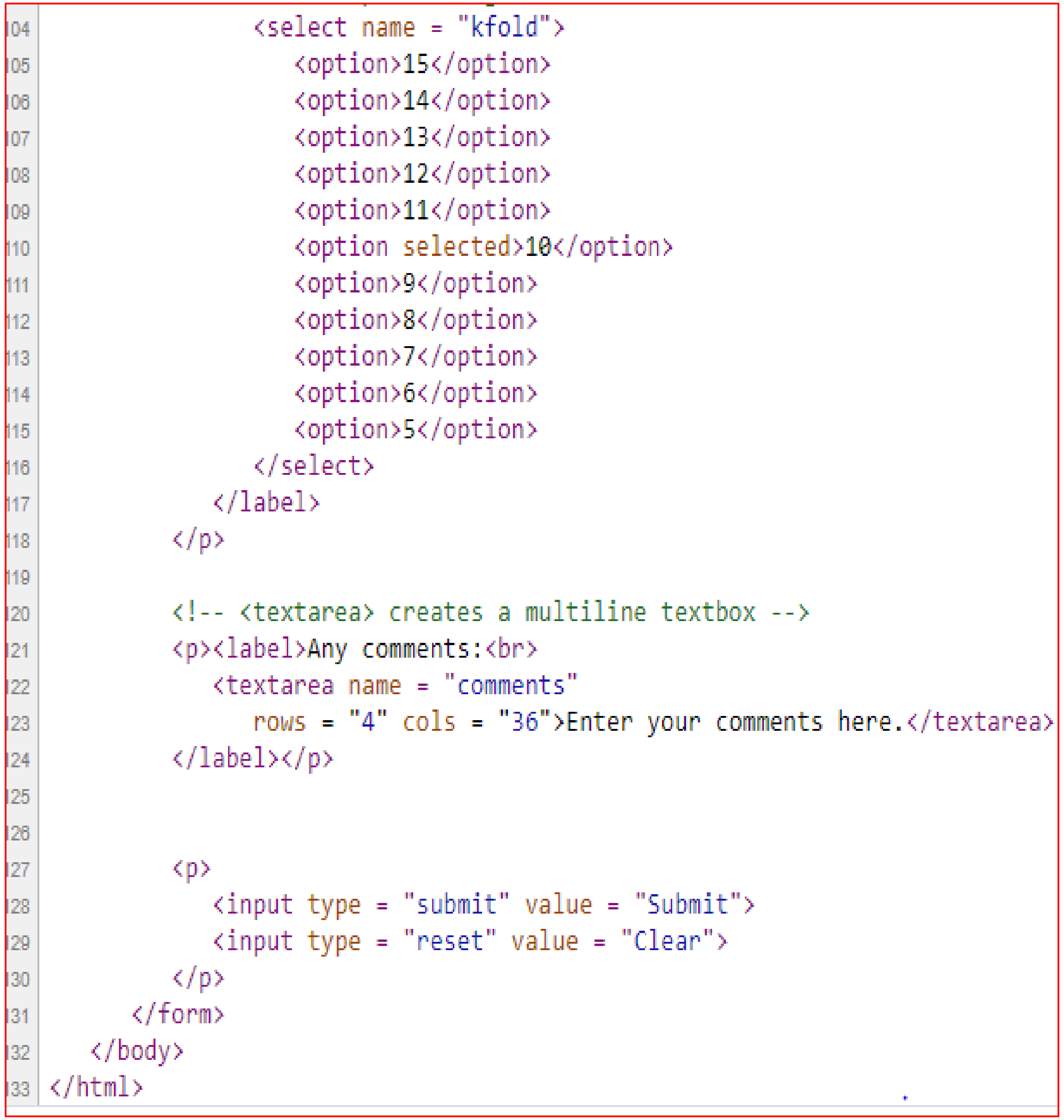
```html
<!DOCTYPE html>

<!-- Fin-model.html -->
<!-- Form using a variety of components. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>More Forms</title>
   </head>

   <body>
      <h1>Python Classificaction</h1>
      <p>Please fill out this form to perform classification</p>

      <form method = "post" action = "results">

    <table>
         <tr> <td colspan = 2> Tell us about yourself </td></tr>
         <tr><td>Your first name: </td><td>
               <input name = "fname" type = "text" size = "25">
            </td></tr>

         <tr><td>Your last name: </td><td>
               <input name = "lname" type = "text" size = "25">
            </td></tr>


         <tr><td>Your E-mail Address: </td><td>
            <input name = "email" type = "text"  size = "25">
         </td></tr>

         <tr><td>Retype Your E-mail Address: </td><td>
            <input name = "retypeemail" type = "text"  size = "25">
         </td></tr>
        </table>
         <p>
```



```html
         <p>
            <strong>Choose estimators you want to use</strong><br>

            <label>
               <input name = "estimator" type = "checkbox"
                  value = "KneighborClassifier"> K nearest neighbor Classifier
            </label>
        <br />
            <label>
               <input name = "estimator" type = "checkbox"
                  value = "SVC">Support Vector Classifier
            </label><br />
            <label>
               <input name = "estimator" type = "checkbox"
                  value = "GaussianNB">Gaussian Naive Bayes
            </label> <br />
            <label>
               <input name = "estimator" type = "checkbox"
                  value = "DecisionTreeClassifier">Decision Tree Classifier
            </label><br />
            <label>
               <input name = "estimator" type = "checkbox"
                  value = "MLPClassifier">Multi-Layer Perceptron Classifier
            </label><br />
            <label>
               <input name = "estimator" type = "checkbox"
                  value = "SGDClassifier">Stochastic Gradient Descent Classifier
             </label>

         </p>

         <!-- <input type = "radio"> creates a radio -->
         <!-- button. The difference between radio buttons -->
         <!-- and checkboxes is that only one radio button -->
         <!-- in a group can be selected. -->
         <p>
```



```
         <p>
            <strong>How do you load your dataset?:</strong><br>

            <label>
               <input name = "howtoload" type = "radio"
                   value = "system" checked>
                Choose data sets from system
               <select name = "from_system">
                  <option selected>fetch_california_housing</option>
                  <option>fetch_kddcup99</option>
                  <option>fetch_lfw_people</option>
                  <option>fetch_20newsgroups</option>
               </select>




           </label>
           <br />

            <label>
               <input name = "howtoload" type = "radio"
                  value = "own">
                Load my own dataset
                <input name = "mydatasetname" type = "text"  size = "25">
            </label>

         <p>
            <label>Select K for kFold cross validation

                <!-- the <select> tag presents a drop-down -->
                <!-- list with choices indicated by the -->
                <!-- <option> tags -->
               <select name = "kfold">
                  <option>15</option>
                  <option>14</option>
```



```
               <select name = "kfold">
                  <option>15</option>
                  <option>14</option>
                  <option>13</option>
                  <option>12</option>
                  <option>11</option>
                  <option selected>10</option>
                  <option>9</option>
                  <option>8</option>
                  <option>7</option>
                  <option>6</option>
                  <option>5</option>
               </select>
            </label>
         </p>

         <!-- <textarea> creates a multiline textbox -->
         <p><label>Any comments:<br>
            <textarea name = "comments"
               rows = "4" cols = "36">Enter your comments here.</textarea>
         </label></p>


         <p>
            <input type = "submit" value = "Submit">
            <input type = "reset" value = "Clear">
         </p>
      </form>
   </body>
</html>
```