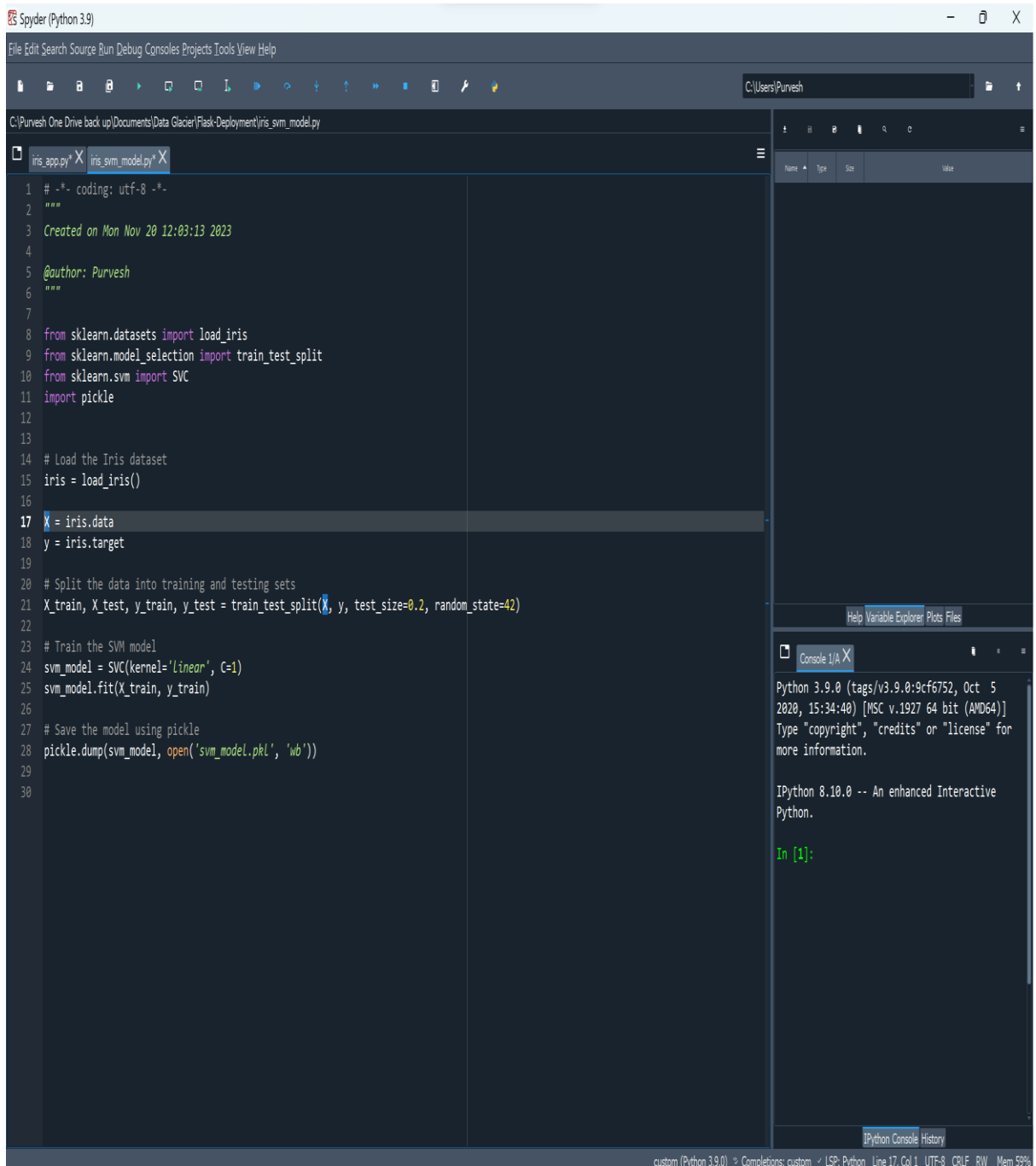


## Step-1 Train ML Model



The screenshot displays the Spyder Python IDE interface. The main editor window shows a Python script named `iris_svm_model.py` with the following code:

```
1  -*- coding: utf-8 -*-
2  """
3  Created on Mon Nov 20 12:03:13 2023
4
5  @author: Purvesh
6  """
7
8  from sklearn.datasets import load_iris
9  from sklearn.model_selection import train_test_split
10 from sklearn.svm import SVC
11 import pickle
12
13
14 # Load the Iris dataset
15 iris = load_iris()
16
17 X = iris.data
18 y = iris.target
19
20 # Split the data into training and testing sets
21 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
22
23 # Train the SVM model
24 svm_model = SVC(kernel='Linear', C=1)
25 svm_model.fit(X_train, y_train)
26
27 # Save the model using pickle
28 pickle.dump(svm_model, open('svm_model.pkl', 'wb'))
29
30
```

The right sidebar contains the Variable Explorer, which is currently empty. Below it is the IPython Console, showing the following output:

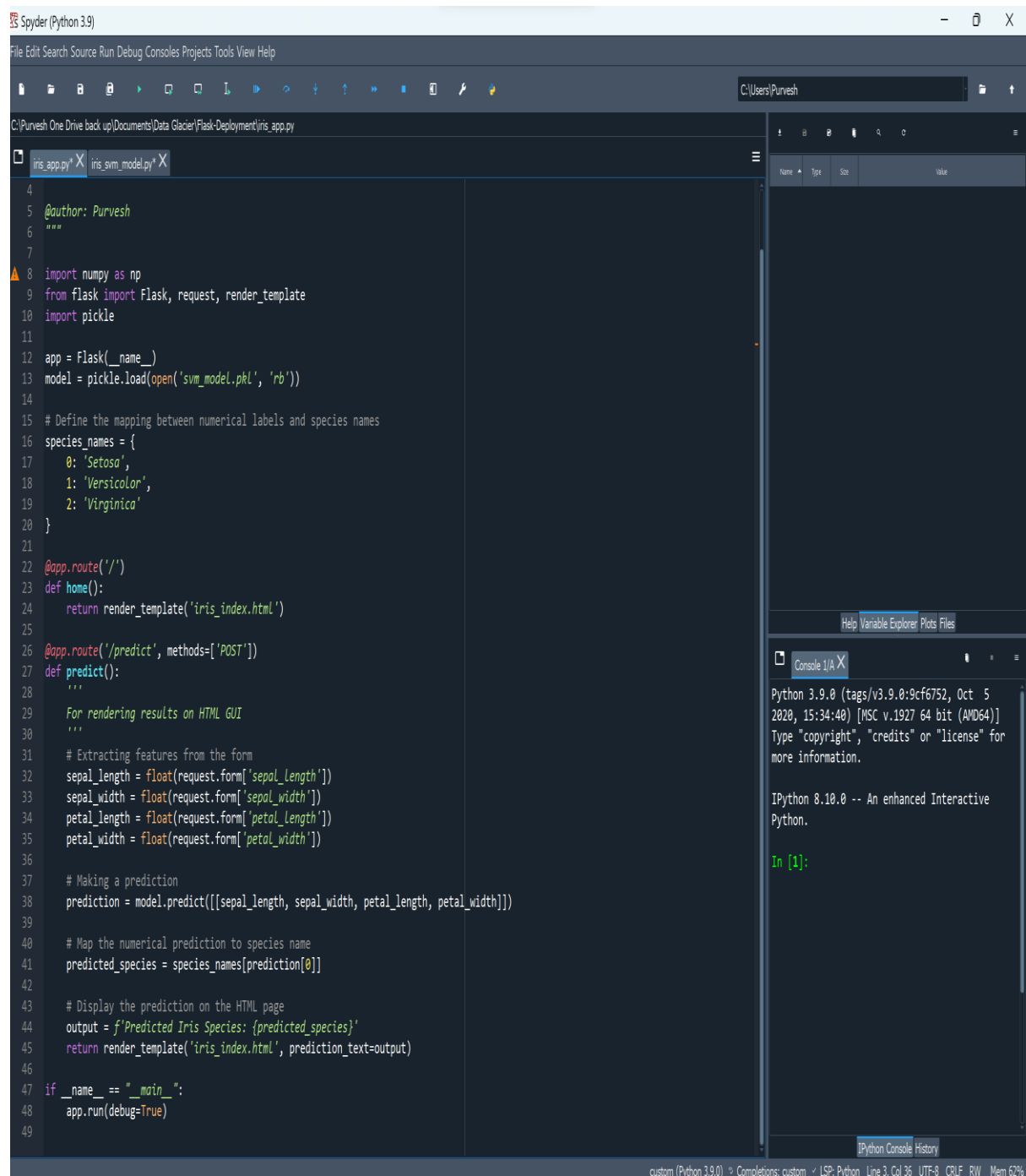
```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5
2020, 15:34:40) [MSC v.1927 64 bit (AMD64)]
Type "copyright", "credits" or "license" for
more information.

IPython 8.10.0 -- An enhanced Interactive
Python.

In [1]:
```

The status bar at the bottom indicates the current file is `custom (Python 3.9.0)`, with a completion count of `custom`, and the cursor is at `Line 17, Col 1` in the `iris_svm_model.py` file.

## Step – 2 Create Web Application using Flask



The screenshot shows the Spyder Python IDE interface. The main editor displays a Python file named `iris_app.py` with the following code:

```
4
5 @author: Purvesh
6 """
7
8 import numpy as np
9 from flask import Flask, request, render_template
10 import pickle
11
12 app = Flask(__name__)
13 model = pickle.load(open('svm_model.pkl', 'rb'))
14
15 # Define the mapping between numerical labels and species names
16 species_names = {
17     0: 'Setosa',
18     1: 'Versicolor',
19     2: 'Virginica'
20 }
21
22 @app.route('/')
23 def home():
24     return render_template('iris_index.html')
25
26 @app.route('/predict', methods=['POST'])
27 def predict():
28     """
29     For rendering results on HTML GUI
30     """
31     # Extracting features from the form
32     sepal_length = float(request.form['sepal_length'])
33     sepal_width = float(request.form['sepal_width'])
34     petal_length = float(request.form['petal_length'])
35     petal_width = float(request.form['petal_width'])
36
37     # Making a prediction
38     prediction = model.predict([[sepal_length, sepal_width, petal_length, petal_width]])
39
40     # Map the numerical prediction to species name
41     predicted_species = species_names[prediction[0]]
42
43     # Display the prediction on the HTML page
44     output = f'Predicted Iris Species: {predicted_species}'
45     return render_template('iris_index.html', prediction_text=output)
46
47 if __name__ == "__main__":
48     app.run(debug=True)
49
```

The right sidebar shows the Variable Explorer, Plots, and Files panels. The Console panel at the bottom displays the following output:

```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct 5 2020, 15:34:40) [MSC v.1927 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information.

IPython 8.10.0 -- An enhanced Interactive Python.

In [1]:
```

The status bar at the bottom indicates: custom (Python 3.9.0) % Completions: custom < LSP: Python Line 3, Col 36 UTF-8 CRLF RW Mem 62%

## Step – 3 Checking Flask Using Postman

The screenshot shows the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', 'API Network', and 'Explore'. The main workspace is titled 'My Workspace' and shows a 'New' button and an 'Import' button. The left sidebar contains 'Collections', 'Environments', and 'History'. The 'Collections' section shows a 'My first collection' with two folders: 'First folder inside collection' and 'Second folder inside collection'. The 'Environments' section shows a 'No Environment' selected. The 'History' section shows a list of requests. The main area displays a GET request to the endpoint `http://127.0.0.1:5000/predict?sepal_length=5.1&sepal_width=3.5&petal_length=1.4&petal_width=0.2`. The request is successful, returning a status of 200 OK. The response body is shown in the 'Body' tab, displaying a JSON object: `{ "predicted_species": "Setosa" }`. The status bar at the bottom shows 'Status: 200 OK', 'Time: 12 ms', and 'Size: 200 B'.

My Workspace **New** **Import** Overview Getting started GET http://127.0.0.1:5000/predict?sepal\_length=5.1&sepal\_width=3.5&petal\_length=1.4&petal\_width=0.2 No Environment

Collections + My first collection ☆

- First folder inside collection
  - GET
  - POST
  - GET
- Second folder inside collection
  - GET
  - GET

Create a collection for your requests

A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it.

Create Collection

GET http://127.0.0.1:5000/predict?sepal\_length=5.1&sepal\_width=3.5&petal\_length=1.4&petal\_width=0.2 Save Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> sepal_length	5.1		
<input checked="" type="checkbox"/> sepal_width	3.5		
<input checked="" type="checkbox"/> petal_length	1.4		
<input checked="" type="checkbox"/> petal_width	0.2		
Key	Value	Description	

Body Cookies Headers (5) Test Results Status: 200 OK Time: 12 ms Size: 200 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "predicted_species": "Setosa"
3 }
```

Online Find and replace Console Postbot Runner Start Proxy Cookies Trash

## Step – 4 Heroku: Linking Github and Deploying ML model

The screenshot shows the Heroku dashboard for an application named 'heroku-demo-week-5'. The browser's address bar shows the URL 'dashboard.heroku.com/apps/heroku-demo-week-5/deploy/github'. The dashboard has a header with the Heroku logo and a search bar. Below the header, there's a section titled 'Manual deploy' with a sub-section 'Deploy a GitHub branch'. This section includes a text box for 'Choose a branch to deploy' with 'main' selected, and a 'Deploy Branch' button. Below this, there's a 'Build main' section showing the deployment log. The log includes the following text: '1.11.4 six-1.16.0 threadpoolctl-3.2.0 tzdata-2023.3', 'Discovering process types', 'Procfile declares types -> web', 'Compressing...', 'Done: 153M', 'Launching...', 'Released v12', and a link to the deployed application: 'https://heroku-demo-week-5-e12e6184e606.herokuapp.com/ deployed to Heroku'. There's also a checkbox for 'Autoscroll with output' and a 'View build log' link. At the bottom, there's a 'Release phase' section and a 'Deploy to Heroku' button. The footer contains links for 'heroku.com', 'Blogs', 'Careers', 'Documentation', 'Support', 'Terms of Service', 'Privacy', 'Cookies', and '© 2023 Salesforce.com'.

Discussion: x Heroku | Si: x heroku-de: x Application: x ChatGPT x jinja2-pyth: x Week-5/re: x Jinja — Jin: x markupsaf: x +

dashboard.heroku.com/apps/heroku-demo-week-5/deploy/github

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Enable Automatic Deploys

Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

main

Deploy Branch

Receive code from GitHub

Build main 9cf5686d

```
1.11.4 six-1.16.0 threadpoolctl-3.2.0 tzdata-2023.3
----> Discovering process types
Procfile declares types -> web
----> Compressing...
Done: 153M
----> Launching...
Released v12
https://heroku-demo-week-5-e12e6184e606.herokuapp.com/ deployed to Heroku
```

☒ Autoscroll with output View build log

Release phase

Deploy to Heroku

heroku.com Blogs Careers Documentation Support

Terms of Service Privacy Cookies © 2023 Salesforce.com

## Step – 5 Testing Web App

The screenshot shows a web browser window with the address bar displaying `heroku-demo-week-5-e12e6184e606.herokuapp.com`. The browser tabs include 'Discussions', 'Heroku | Sign', 'heroku-demo', 'Loading...', 'jinja2-python', 'Week-5/require', 'Jinja - Jinja', and 'markupsafe'. The main content area has a blue background with a dark blue rounded rectangle in the center. Inside this rectangle, the title 'Predict Iris Species' is at the top. Below it are four input fields containing the values '4.6', '3.1', '1.5', and '0.2'. A blue 'Predict' button is positioned below the input fields. At the bottom of the rectangle, the text 'Predicted Iris Species: Setosa' is displayed.

Predict Iris Species

4.6

3.1

1.5

0.2

Predict

Predicted Iris Species: Setosa