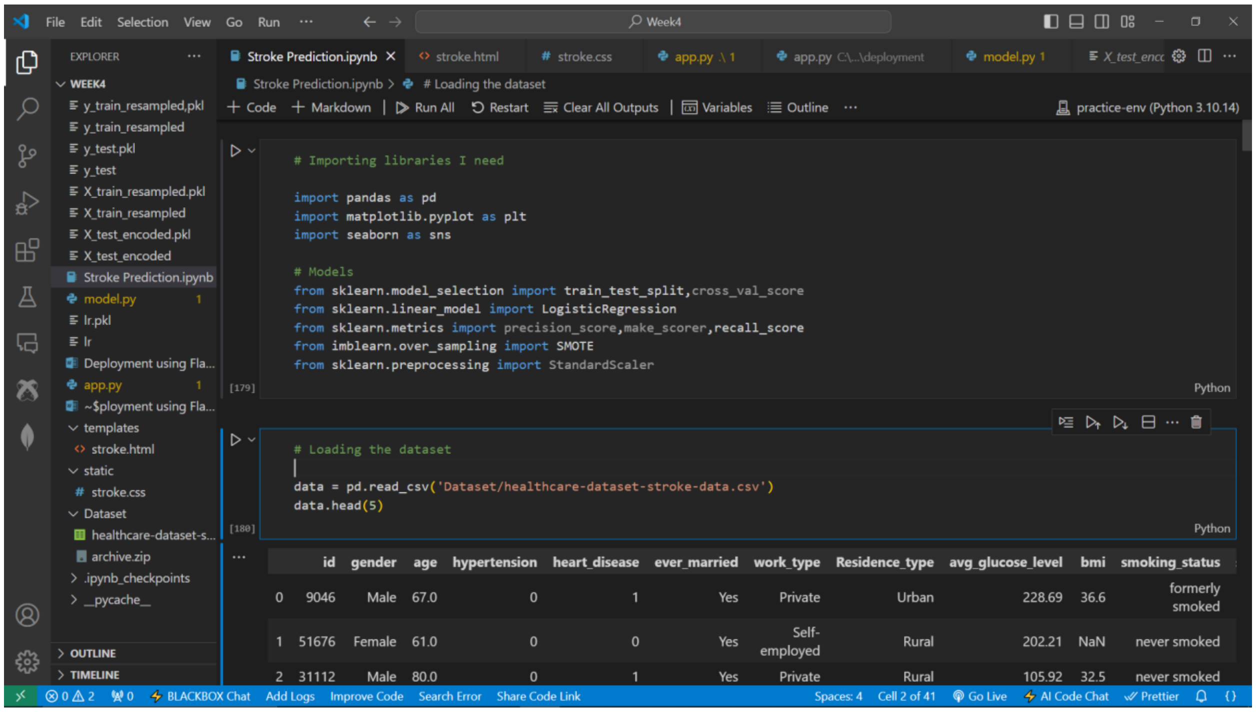
CLOUD AND API DEPLOYMENT

NAME: VIVIAN KERUBO MOSOMI

BATCH CODE: LISUM39

SUBMISSION DATE: 01/12/24

1.Loading the Data

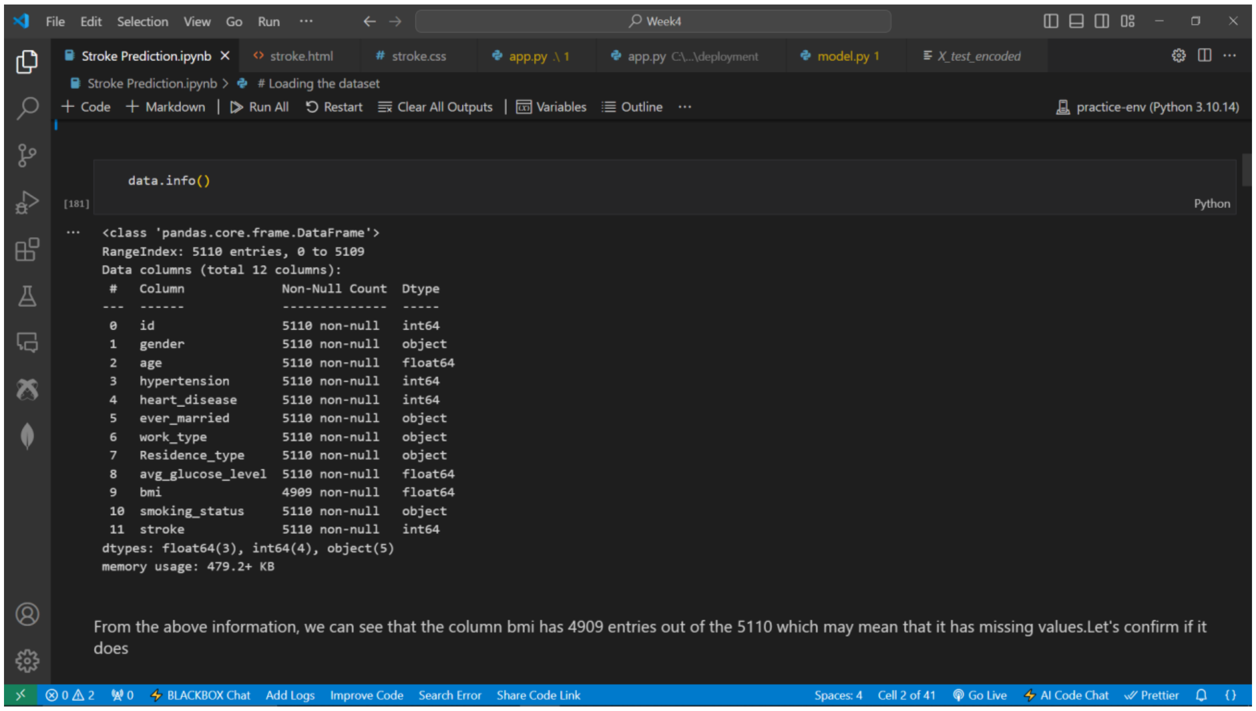


```python
# Importing libraries I need

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Models
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_score,make_scorer,recall_score
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import StandardScaler
```

```python
# Loading the dataset

data = pd.read_csv('Dataset/healthcare-dataset-stroke-data.csv')
data.head(5)
```

```
      id  gender   age  hypertension  heart_disease  ever_married  work_type  Residence_type  avg_glucose_level   bmi  smoking_status
0   9046    Male  67.0             0              1           Yes    Private           Urban             228.69  36.6        formerly
                                                                                                                             smoked
1  51676  Female  61.0             0              0           Yes      Self-           Rural             202.21   NaN    never smoked
                                                                     employed
2  31112    Male  80.0             0              1           Yes    Private           Rural             105.92  32.5    never smoked
```

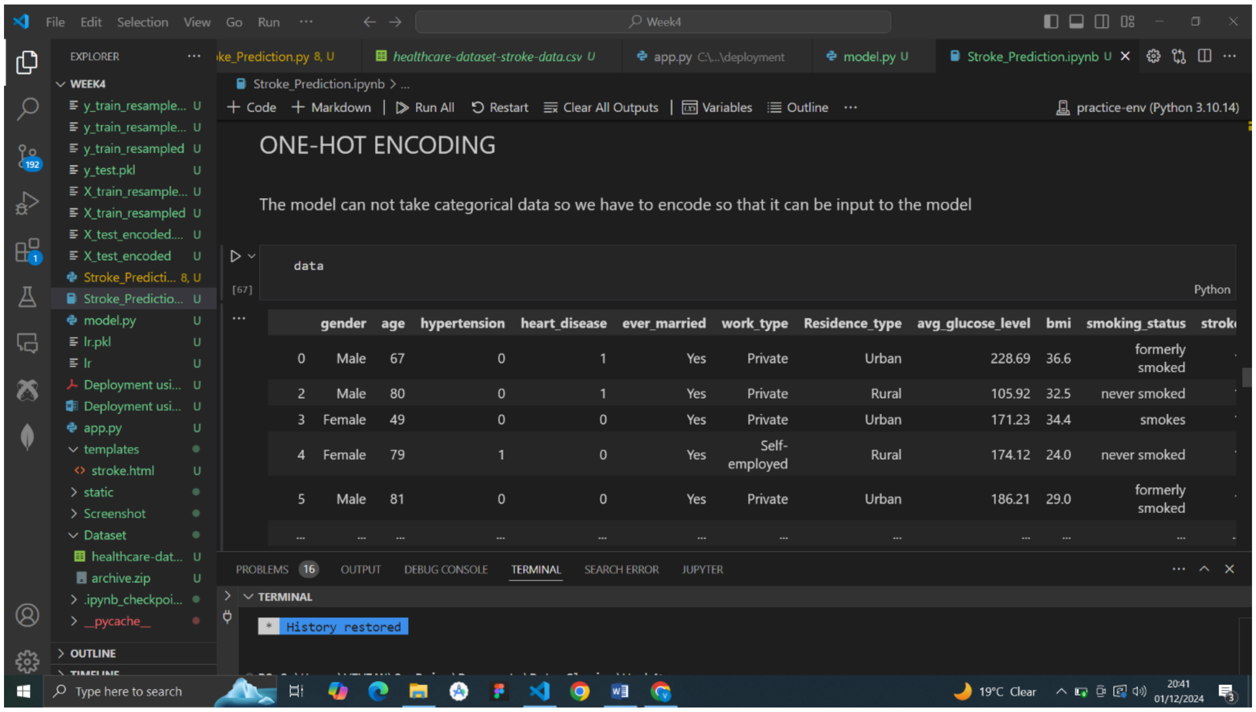
2.Data Preparation



```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   id                 5110 non-null   int64
 1   gender             5110 non-null   object
 2   age                5110 non-null   float64
 3   hypertension       5110 non-null   int64
 4   heart_disease      5110 non-null   int64
 5   ever_married       5110 non-null   object
 6   work_type          5110 non-null   object
 7   Residence_type     5110 non-null   object
 8   avg_glucose_level  5110 non-null   float64
 9   bmi                4909 non-null   float64
 10  smoking_status     5110 non-null   object
 11  stroke             5110 non-null   int64
dtypes: float64(3), int64(4), object(5)
memory usage: 479.2+ KB
```

From the above information, we can see that the column bmi has 4909 entries out of the 5110 which may mean that it has missing values.Let's confirm if it does



ONE-HOT ENCODING

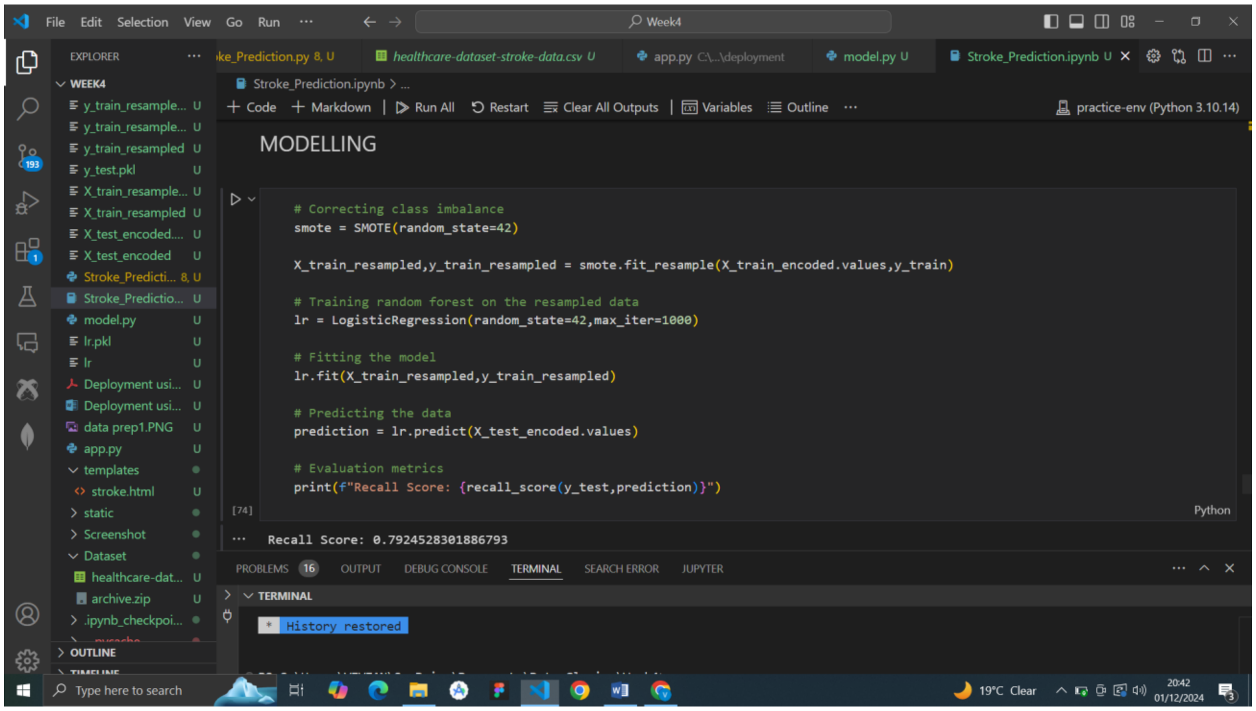
The model can not take categorical data so we have to encode so that it can be input to the model

```
data
```

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	strok
0	Male	67	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	
2	Male	80	0	1	Yes	Private	Rural	105.92	32.5	never smoked	
3	Female	49	0	0	Yes	Private	Urban	171.23	34.4	smokes	
4	Female	79	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	
5	Male	81	0	0	Yes	Private	Urban	186.21	29.0	formerly smoked	
...	...	...	...	...	...	...	...	...	...	...	

```
* History restored
```

3. Modelling


File Edit Selection View Go Run
Week4
EXPLORER
WEEK4
Stroke_Prediction.ipynb
+ Code + Markdown Run All Restart Clear All Outputs Variables Outline
practice-env (Python 3.10.14)
PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL SEARCH ERROR JUPYTER
History restored
Type here to search
19°C Clear
20:42 01/12/2024

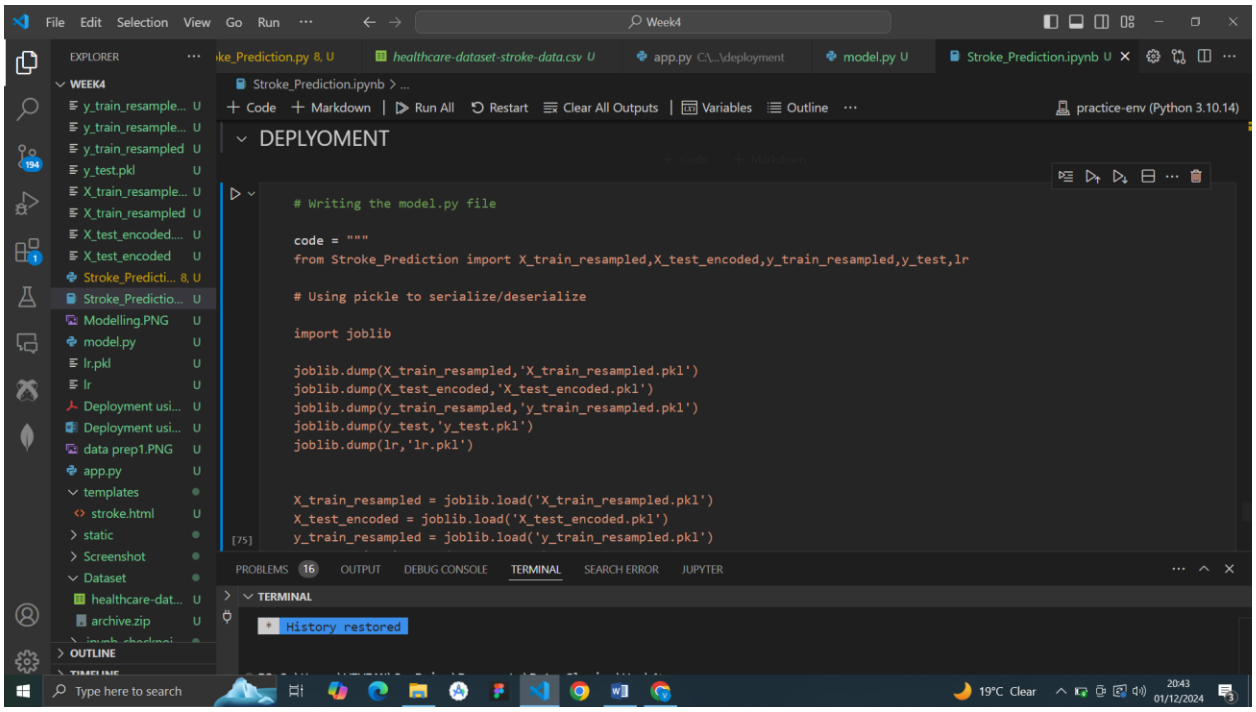
MODELLING

```
# Correcting class imbalance
smote = SMOTE(random_state=42)

X_train_resampled,y_train_resampled = smote.fit_resample(X_train_encoded.values,y_train)

# Training random forest on the resampled data
lr = LogisticRegression(random_state=42,max_iter=1000)

# Fitting the model
lr.fit(X_train_resampled,y_train_resampled)

# Predicting the data
prediction = lr.predict(X_test_encoded.values)

# Evaluation metrics
print(f"Recall Score: {recall_score(y_test,prediction)}")
```

[74]

```
Recall Score: 0.7924528301886793
```

4.Deployment

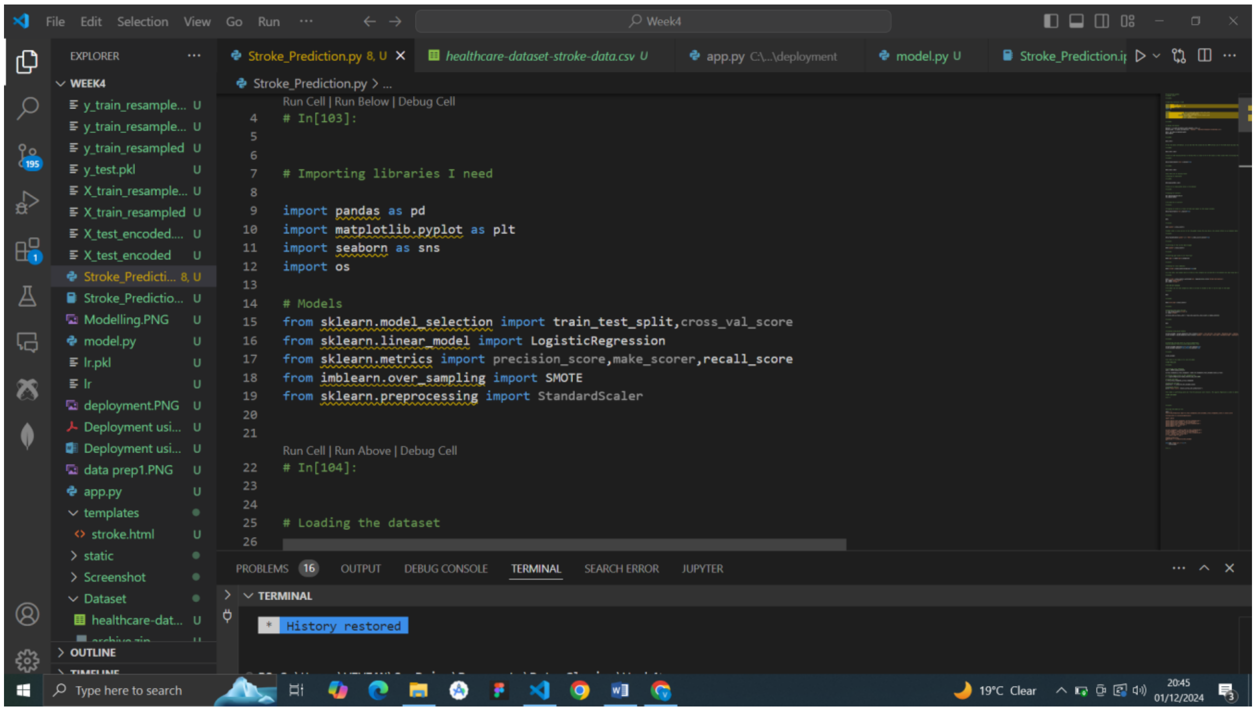

File Edit Selection View Go Run
Week4
EXPLORER
WEEK4
Stroke_Prediction.ipynb
+ Code + Markdown Run All Restart Clear All Outputs Variables Outline
practice-env (Python 3.10.14)
PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL SEARCH ERROR JUPYTER
History restored
Type here to search
19°C Clear
20:43 01/12/2024

DEPLYOMENT

```
# Writing the model.py file

code = """
from Stroke_Prediction import X_train_resampled,X_test_encoded,y_train_resampled,y_test,lr

# Using pickle to serialize/deserialize

import joblib

joblib.dump(X_train_resampled,'X_train_resampled.pkl')
joblib.dump(X_test_encoded,'X_test_encoded.pkl')
joblib.dump(y_train_resampled,'y_train_resampled.pkl')
joblib.dump(y_test,'y_test.pkl')
joblib.dump(lr,'lr.pkl')


X_train_resampled = joblib.load('X_train_resampled.pkl')
X_test_encoded = joblib.load('X_test_encoded.pkl')
y_train_resampled = joblib.load('y_train_resampled.pkl')
```

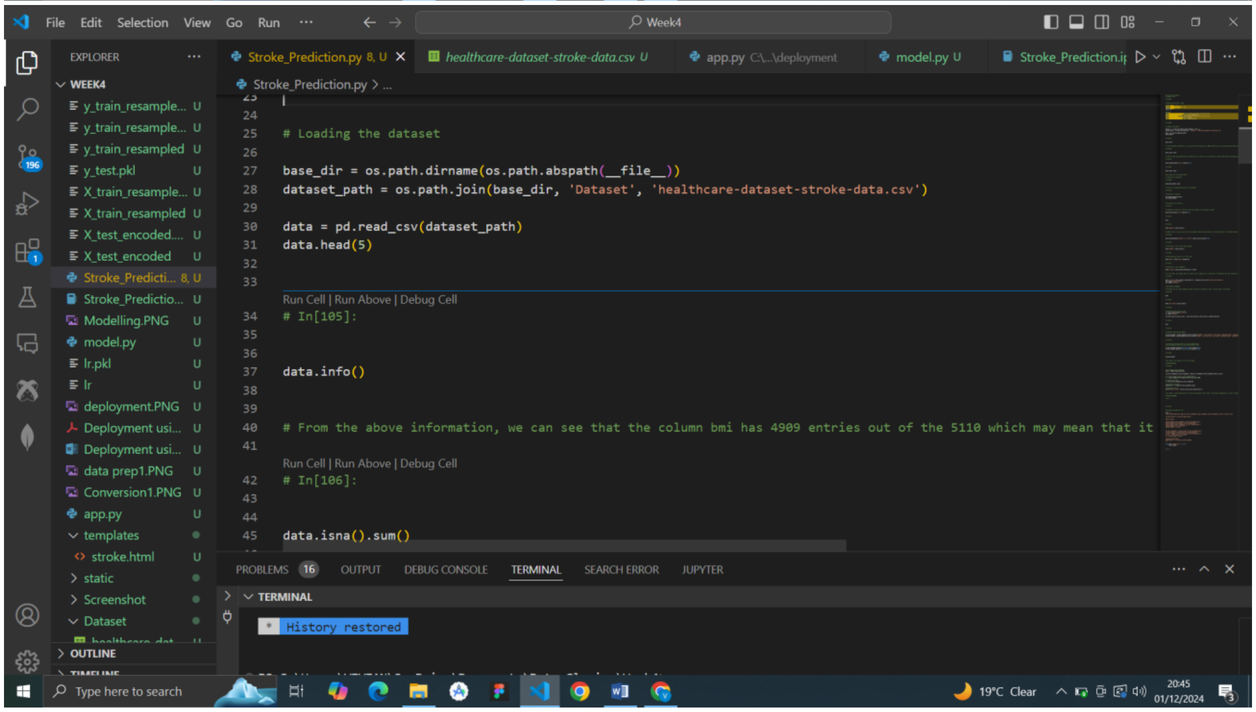
[75]

5.Converting ipynb to .py file



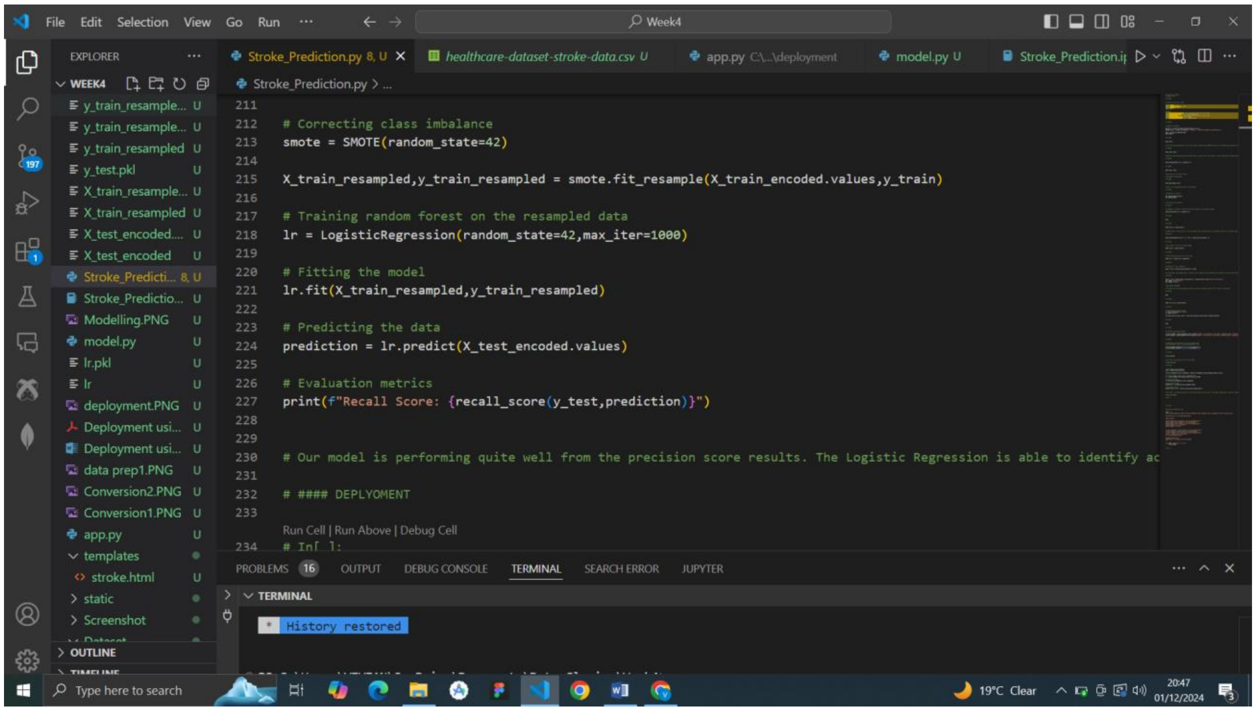
```
Run Cell | Run Below | Debug Cell
# In[103]:


# Importing libraries I need

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os

# Models
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_score,make_scorer,recall_score
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import StandardScaler


Run Cell | Run Above | Debug Cell
# In[104]:


# Loading the dataset
```

```
* History restored
```



```
# Loading the dataset

base_dir = os.path.dirname(os.path.abspath(__file__))
dataset_path = os.path.join(base_dir, 'Dataset', 'healthcare-dataset-stroke-data.csv')

data = pd.read_csv(dataset_path)
data.head(5)


Run Cell | Run Above | Debug Cell
# In[105]:


data.info()


# From the above information, we can see that the column bmi has 4909 entries out of the 5110 which may mean that it

Run Cell | Run Above | Debug Cell
# In[106]:


data.isna().sum()
```

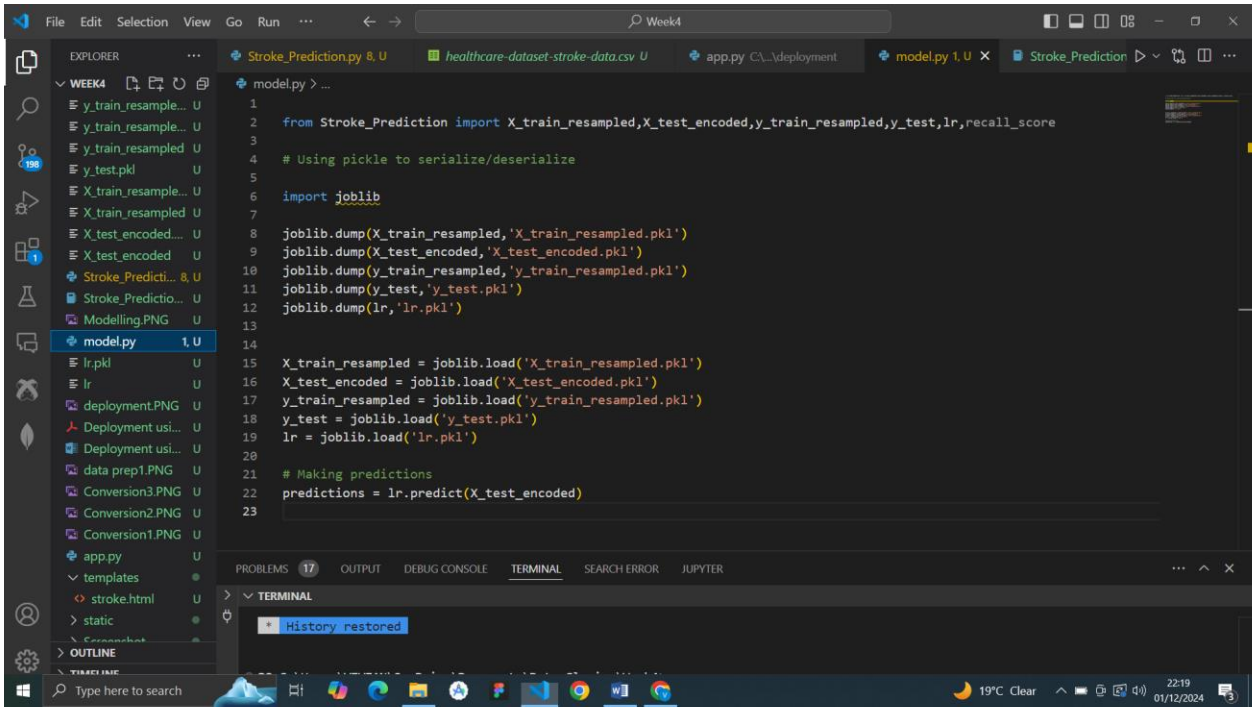
```
* History restored
```



```
# Correcting class imbalance
smote = SMOTE(random_state=42)

X_train_resampled,y_train_resampled = smote.fit_resample(X_train_encoded.values,y_train)

# Training random forest on the resampled data
lr = LogisticRegression(random_state=42,max_iter=1000)

# Fitting the model
lr.fit(X_train_resampled,y_train_resampled)

# Predicting the data
prediction = lr.predict(X_test_encoded.values)

# Evaluation metrics
print(f"Recall Score: {recall_score(y_test,prediction)}")


# Our model is performing quite well from the precision score results. The Logistic Regression is able to identify ac

# #### DEPLOYMENT

Run Cell | Run Above | Debug Cell
# In[ ]:
```

```
History restored
```

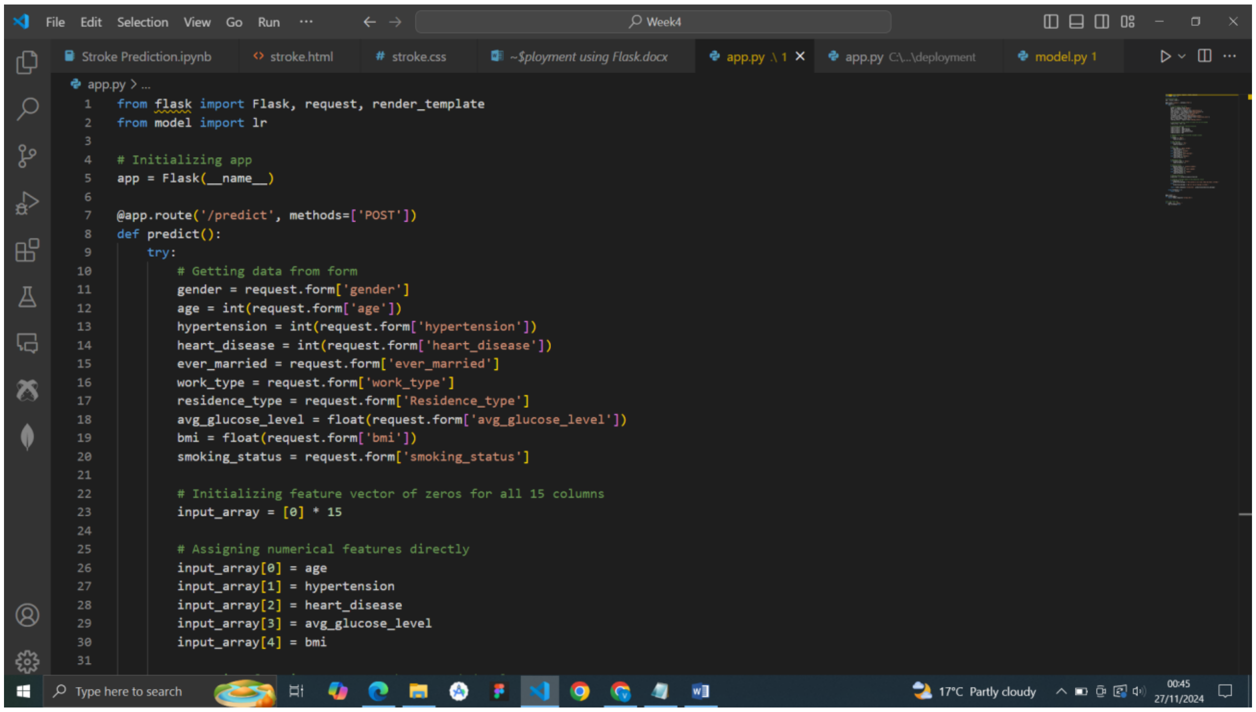
6. Predicting using serialized files



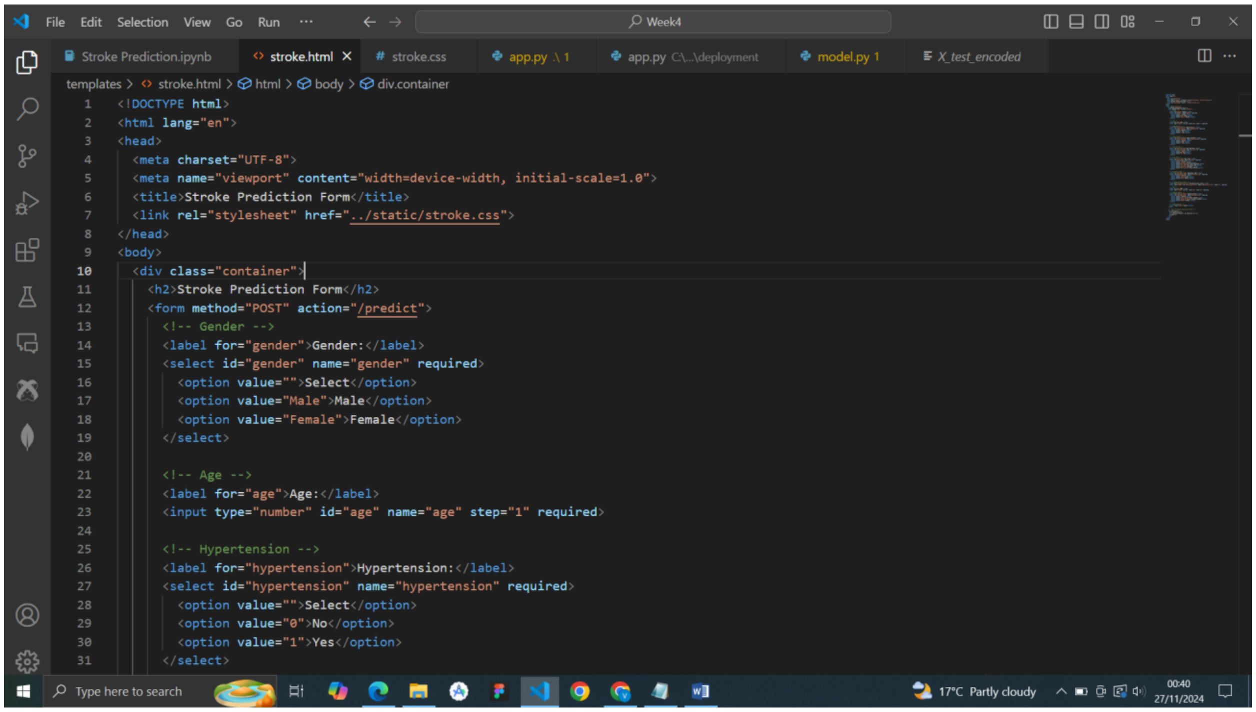
```

from Stroke_Prediction import X_train_resampled,X_test_encoded,y_train_resampled,y_test,lr,recall_score

# Using pickle to serialize/deserialize

import joblib

joblib.dump(X_train_resampled,'X_train_resampled.pkl')
joblib.dump(X_test_encoded,'X_test_encoded.pkl')
joblib.dump(y_train_resampled,'y_train_resampled.pkl')
joblib.dump(y_test,'y_test.pkl')
joblib.dump(lr,'lr.pkl')


X_train_resampled = joblib.load('X_train_resampled.pkl')
X_test_encoded = joblib.load('X_test_encoded.pkl')
y_train_resampled = joblib.load('y_train_resampled.pkl')
y_test = joblib.load('y_test.pkl')
lr = joblib.load('lr.pkl')

# Making predictions
predictions = lr.predict(X_test_encoded)
```

```
History restored
```

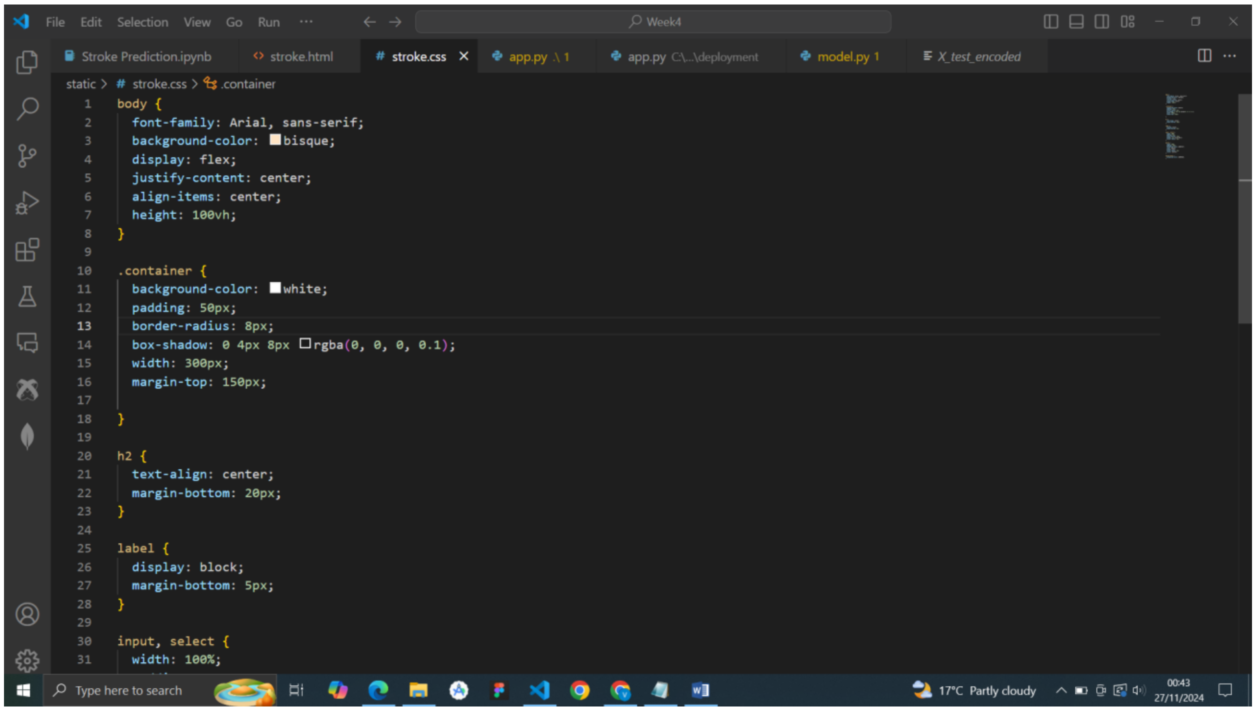
7. App.py, Html and Css Files



```python
from flask import Flask, request, render_template
from model import lr

# Initializing app
app = Flask(__name__)

@app.route('/predict', methods=['POST'])
def predict():
    try:
        # Getting data from form
        gender = request.form['gender']
        age = int(request.form['age'])
        hypertension = int(request.form['hypertension'])
        heart_disease = int(request.form['heart_disease'])
        ever_married = request.form['ever_married']
        work_type = request.form['work_type']
        residence_type = request.form['Residence_type']
        avg_glucose_level = float(request.form['avg_glucose_level'])
        bmi = float(request.form['bmi'])
        smoking_status = request.form['smoking_status']

        # Initializing feature vector of zeros for all 15 columns
        input_array = [0] * 15

        # Assigning numerical features directly
        input_array[0] = age
        input_array[1] = hypertension
        input_array[2] = heart_disease
        input_array[3] = avg_glucose_level
        input_array[4] = bmi

```

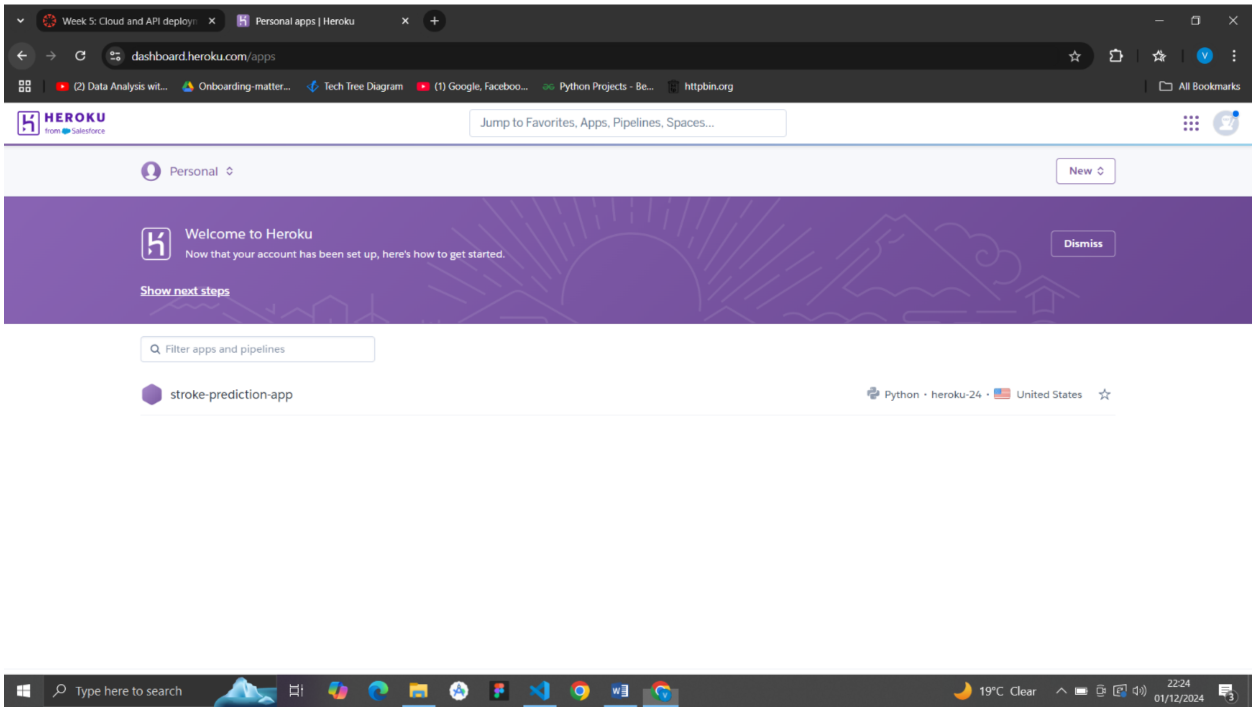


```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Stroke Prediction Form</title>
  <link rel="stylesheet" href="../static/stroke.css">
</head>
<body>
  <div class="container">
    <h2>Stroke Prediction Form</h2>
    <form method="POST" action="/predict">
      <!-- Gender -->
      <label for="gender">Gender:</label>
      <select id="gender" name="gender" required>
        <option value="">Select</option>
        <option value="Male">Male</option>
        <option value="Female">Female</option>
      </select>

      <!-- Age -->
      <label for="age">Age:</label>
      <input type="number" id="age" name="age" step="1" required>

      <!-- Hypertension -->
      <label for="hypertension">Hypertension:</label>
      <select id="hypertension" name="hypertension" required>
        <option value="">Select</option>
        <option value="0">No</option>
        <option value="1">Yes</option>
      </select>
```



```css
body {
  font-family: Arial, sans-serif;
  background-color: bisque;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

.container {
  background-color: white;
  padding: 50px;
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  width: 300px;
  margin-top: 150px;

}

h2 {
  text-align: center;
  margin-bottom: 20px;
}

label {
  display: block;
  margin-bottom: 5px;
}

input, select {
  width: 100%;
```

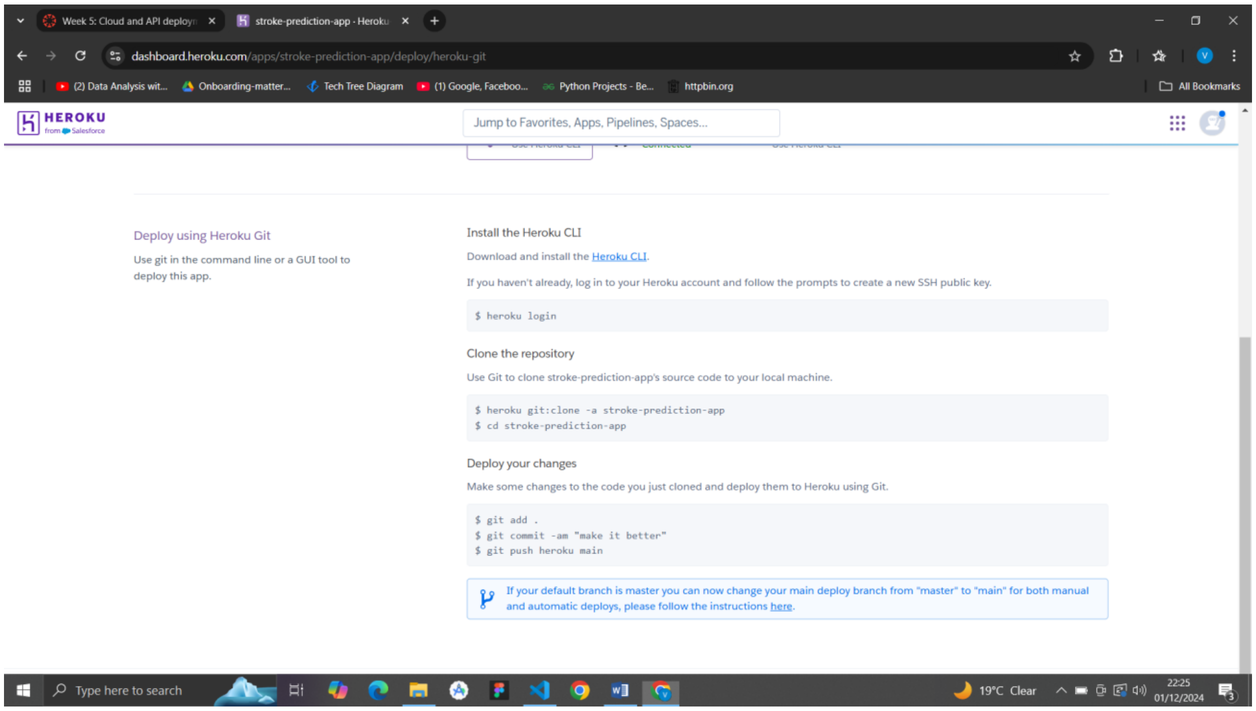
8.Login To Heroku



Welcome to Heroku
Now that your account has been set up, here's how to get started.

stroke-prediction-app   Python · heroku-24 · United States

9.Connect to the app created on Heroku


Deploy using Heroku Git

Use git in the command line or a GUI tool to deploy this app.

Install the Heroku CLI

Download and install the Heroku CLI.

If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

Clone the repository

Use Git to clone stroke-prediction-app's source code to your local machine.

```
$ heroku git:clone -a stroke-prediction-app
$ cd stroke-prediction-app
```

Deploy your changes

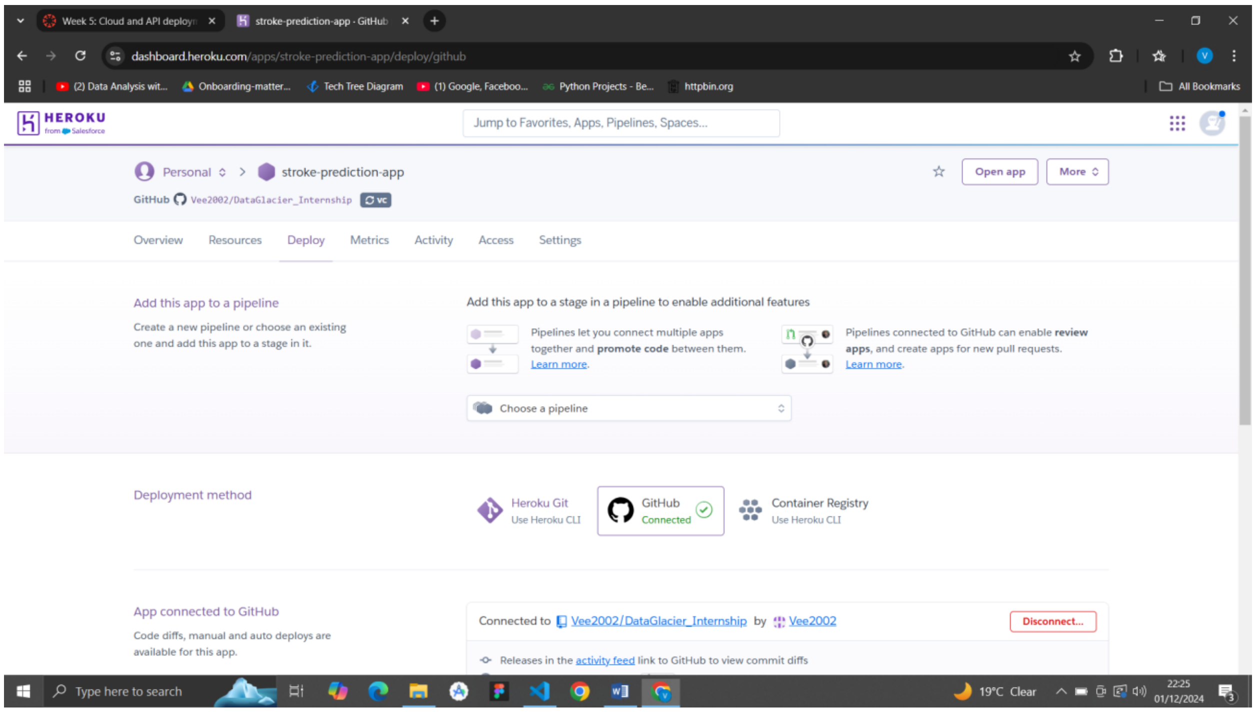
Make some changes to the code you just cloned and deploy them to Heroku using Git.

```
$ git add .
$ git commit -am "make it better"
$ git push heroku main
```

If your default branch is master you can now change your main deploy branch from "master" to "main" for both manual and automatic deploys, please follow the instructions here.

Install the Heroku CLI, and use git bash or command prompt to run the Heroku commands or connect using GitHub

10.Open the app


Personal > stroke-prediction-app

GitHub Vee2002/DataGlacier_Internship

Add this app to a pipeline

Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

Pipelines let you connect multiple apps together and promote code between them. Learn more.

Pipelines connected to GitHub can enable review apps, and create apps for new pull requests. Learn more.

Deployment method

Heroku Git – Use Heroku CLI; GitHub – Connected; Container Registry – Use Heroku CLI

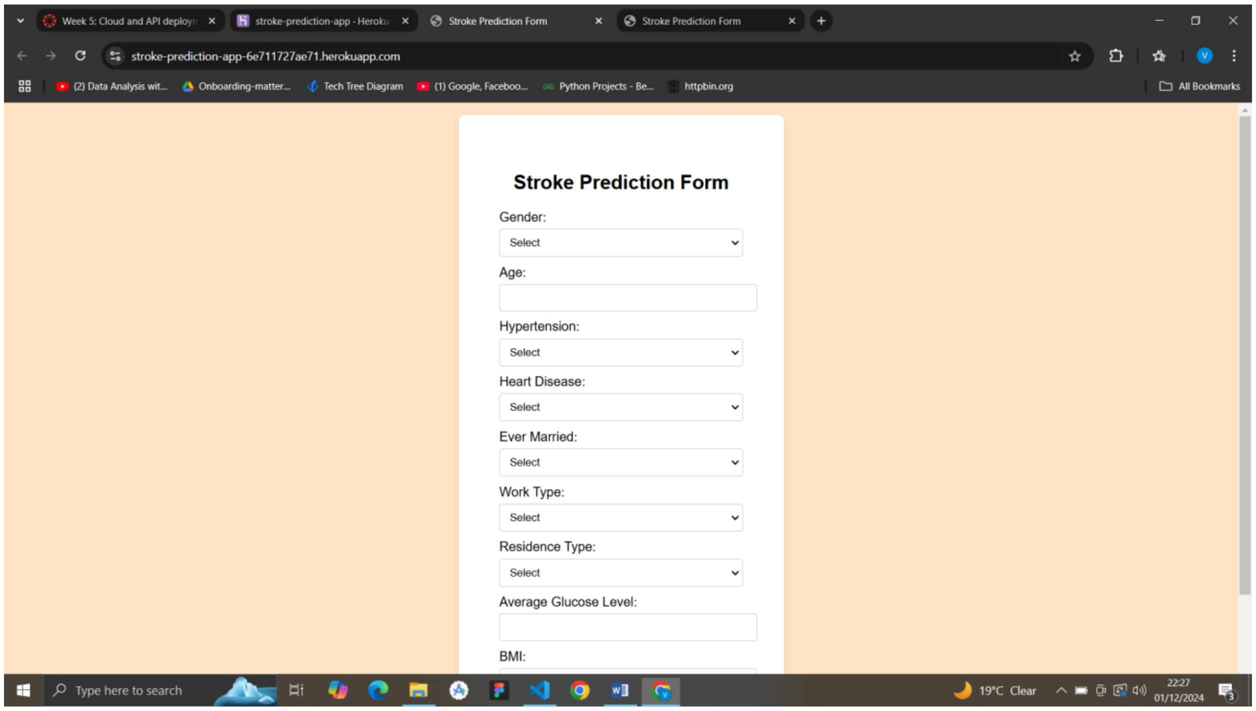
App connected to GitHub

Code diffs, manual and auto deploys are available for this app.

Connected to Vee2002/DataGlacier_Internship by Vee2002

Releases in the activity feed link to GitHub to view commit diffs

11.Load the app and fill in the form



stroke-prediction-app-6e711727ae71.herokuapp.com

Stroke Prediction Form

Gender: Select

Age:

Hypertension: Select

Heart Disease: Select

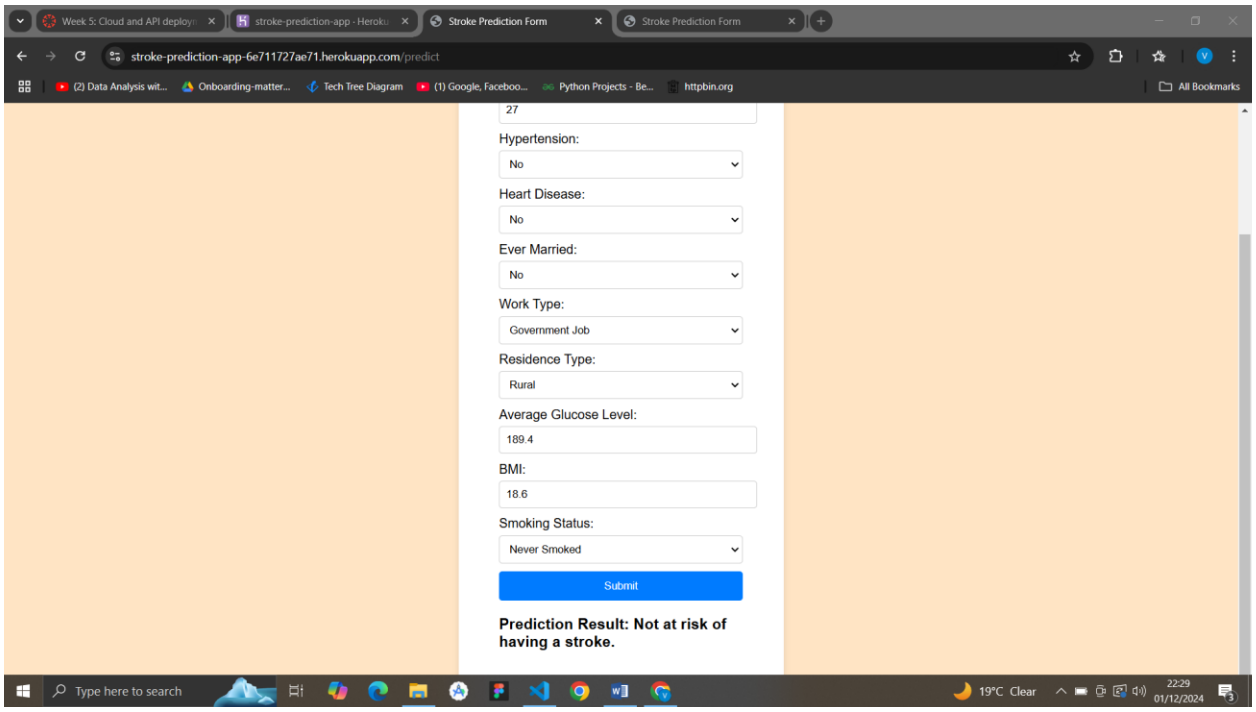
Ever Married: Select

Work Type: Select

Residence Type: Select

Average Glucose Level:

BMI:



stroke-prediction-app-6e711727ae71.herokuapp.com/predict

27

Hypertension: No

Heart Disease: No

Ever Married: No

Work Type: Government Job

Residence Type: Rural

Average Glucose Level: 189.4

BMI: 18.6

Smoking Status: Never Smoked

Submit

Prediction Result: Not at risk of having a stroke.

Our application is working well on Heroku.