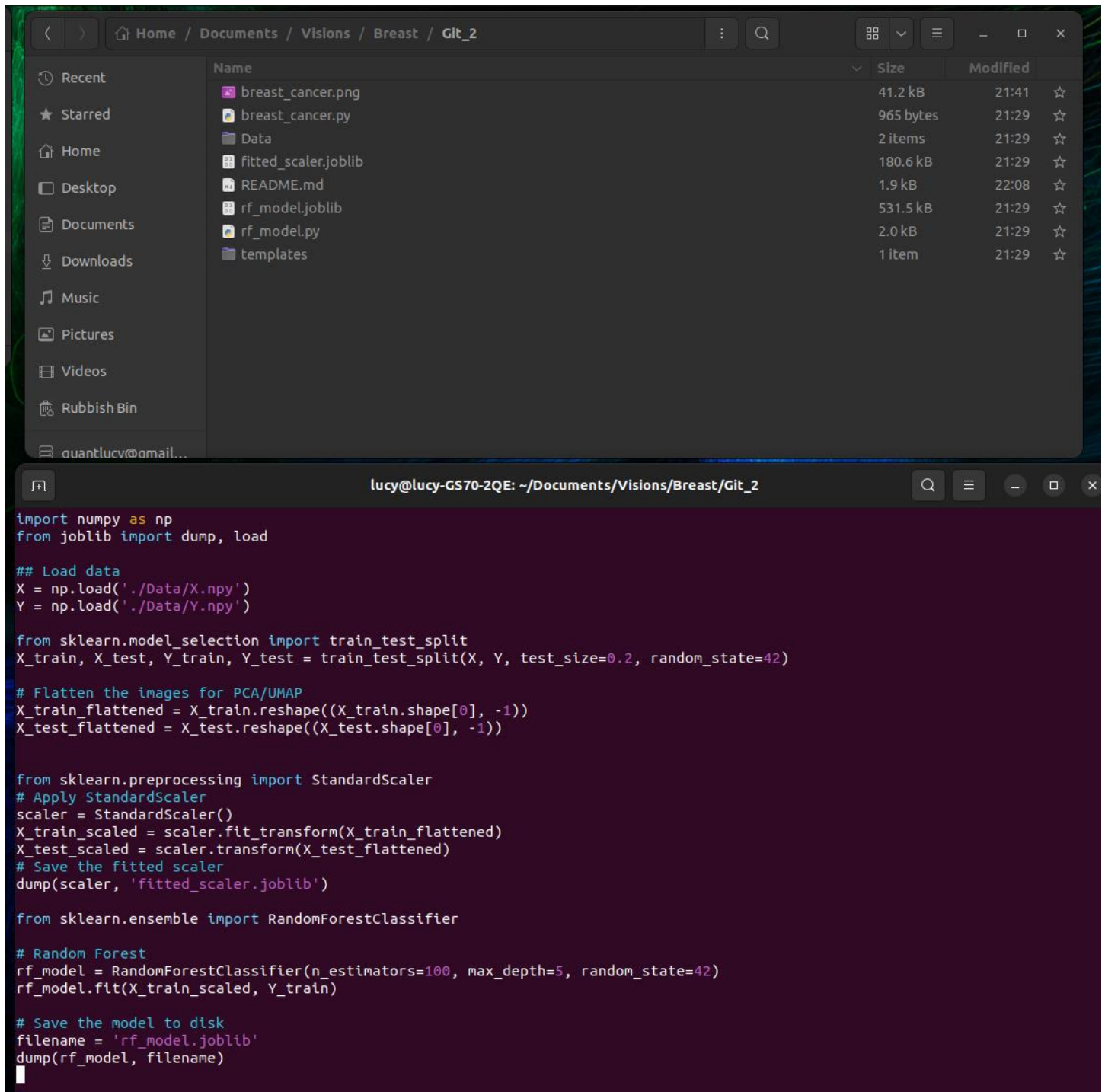


Breast Cancer

FILES STRUCTURE & ML MODEL



The image shows a file explorer window and a terminal window. The file explorer window displays the contents of the directory `~/Documents/Visions/Breast/Git_2`. The terminal window shows the Python code used to train the model.

File Explorer Window:

| Name | Size | Modified |
|----------------------|-----------|----------|
| breast_cancer.png | 41.2 kB | 21:41 |
| breast_cancer.py | 965 bytes | 21:29 |
| Data | 2 items | 21:29 |
| fitted_scaler.joblib | 180.6 kB | 21:29 |
| README.md | 1.9 kB | 22:08 |
| rf_model.joblib | 531.5 kB | 21:29 |
| rf_model.py | 2.0 kB | 21:29 |
| templates | 1 item | 21:29 |

Terminal Window:

```
lucy@lucy-GS70-2QE: ~/Documents/Visions/Breast/Git_2
import numpy as np
from joblib import dump, load

## Load data
X = np.load('./Data/X.npy')
Y = np.load('./Data/Y.npy')

from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

# Flatten the images for PCA/UMAP
X_train_flattened = X_train.reshape((X_train.shape[0], -1))
X_test_flattened = X_test.reshape((X_test.shape[0], -1))

from sklearn.preprocessing import StandardScaler
# Apply StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train_flattened)
X_test_scaled = scaler.transform(X_test_flattened)
# Save the fitted scaler
dump(scaler, 'fitted_scaler.joblib')

from sklearn.ensemble import RandomForestClassifier

# Random Forest
rf_model = RandomForestClassifier(n_estimators=100, max_depth=5, random_state=42)
rf_model.fit(X_train_scaled, Y_train)

# Save the model to disk
filename = 'rf_model.joblib'
dump(rf_model, filename)
```

FLASK IMPLEMENTATION

```
lucy@lucy-GS70-2QE: ~/Documents/Visions/Breast/Git_2
from flask import Flask, request, jsonify, render_template
from joblib import load
from sklearn.preprocessing import StandardScaler
import numpy as np
import os

import base64
from io import BytesIO
import matplotlib.pyplot as plt
import matplotlib
matplotlib.use('Agg')

app = Flask(__name__)

# Load the model and scaler at app startup
model_path = os.path.join('/home/lucy/Documents/Visions/Breast/Git_1', 'rf_model.joblib')
model = load(model_path)
scaler_path = '/home/lucy/Documents/Visions/Breast/Git_1/fitted_scaler.joblib'
scaler = load(scaler_path)

@app.route('/')
def index():
    # Display the form for user input
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    try:
        image_index = int(request.form['image_index'])
        X = np.load('./Data/X.npy') # Load data

        print("Selected image shape:", X[image_index].shape) # Should print (50, 50, 3)

        # Flatten the image for prediction
        image_flat = X[image_index].reshape(-1) # Flatten the image
        print("Flattened image shape:", image_flat.shape) # Should print (7500,)

        # Transform the data using the pre-fitted scaler
        image_scaled = scaler.transform([image_flat]) # Make sure it's a 2D array for scaling

        # Predict using the loaded model
        prediction = model.predict(image_scaled)
        result = 'No Cancer' if prediction[0] == 0 else 'Cancer'

        # Convert image to Base64 for displaying
        fig, ax = plt.subplots()
        ax.imshow(X[image_index]) # Use original shape for displaying
        ax.axis('off')
        buf = BytesIO()
        plt.savefig(buf, format='png')
        plt.close(fig)
        image_base64 = base64.b64encode(buf.getvalue()).decode('utf-8')
        buf.close()

        return jsonify({'prediction': result, 'image': image_base64})
    except Exception as e:
        return jsonify({'error': str(e)}), 500

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

GITHUB DEPLOYMENT

github.com/LucyNowacki/CancerDetector

Course: School... Course(s) Slownik angiel... Dashboard

LucyNowacki / CancerDetector

Type to search

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

CancerDetector Public

Pin Unwatch 1 Fork 0 Star 0

main 1 Branch 0 Tags

Go to file Add file Code

LucyNowacki Update README with dataset image 16d9ec2 · 10 minutes ago 6 Commits

| | | |
|----------------------|----------------------------------|----------------|
| Data | Initial commit | 49 minutes ago |
| templates | Initial commit | 49 minutes ago |
| .README.md.swp | Initial commit | 49 minutes ago |
| README.md | Update README with dataset image | 10 minutes ago |
| breast_cancer.png | Add breast cancer dataset image | 25 minutes ago |
| breast_cancer.py | Initial commit | 49 minutes ago |
| fitted_scaler.joblib | Initial commit | 49 minutes ago |
| rf_model.joblib | Initial commit | 49 minutes ago |
| rf_model.py | Initial commit | 49 minutes ago |

README

CancerDetector

Overview

`CancerDetector` is a Flask-based web application that uses a machine learning model to predict the presence of breast cancer in histopathological images.

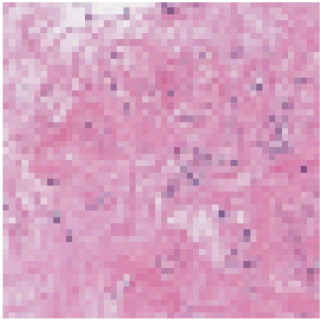
127.0.0.1:5000

Course: School... Course(s) Slownik angiel... Dashboard

Breast Cancer Detector

Enter Image Index (0-1109):

Prediction: No Cancer



Model Details

The application employs a Random Forest classifier, which has been trained on a standardized dataset of 50x50 pixel histopathological images. Each image is labeled with one of two possible categories: `Cancer` or `No Cancer`.

Installation

To get `CancerDetector` up and running on your local machine, follow these instructions:

Prerequisites

- Python 3.8 or higher

About

Detects breast cancer on images.

Readme

Activity

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

Python 67.3%

HTML 32.7%

Suggested workflows

Based on your tech stack

SLSA Generic generator

Configure

Generate SLSA3 provenance for your existing release workflows

Pylint

Configure

Lint a Python application with pylint.

Python package

Configure

Create and test a Python package on multiple Python versions.





More workflows

Dismiss suggestions

START APP

```
lucy@lucy-GS70-2QE: ~/Documents/Visions/Breast/Git_2
(lucy) lucy@lucy-GS70-2QE:~/Documents/Visions/Breast/Git_2$ export FLASK_APP=rf_model.py
(lucy) lucy@lucy-GS70-2QE:~/Documents/Visions/Breast/Git_2$ flask run
* Serving Flask app 'rf_model.py'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [28/Apr/2024 22:24:08] "GET / HTTP/1.1" 200 -
Selected image shape: (50, 50, 3)
Flattened image shape: (7500,)
127.0.0.1 - - [28/Apr/2024 22:24:15] "POST /predict HTTP/1.1" 200 -
```

[←](#) [→](#) [↻](#) [ⓘ](#) 127.0.0.1:5000

 Course: School...  Course(s)  Słownik angielski...  Dashboard

Breast Cancer Detector

Enter Image Index (0-1109):

Prediction: Cancer

