LABORATORY REPORT

Application Development Lab
(CS33002)

B.Tech Program in ECSc

Submitted By

Name:- Bhairav Ganguly

Roll No: 2230246

Kalinga Institute of Industrial Technology
(Deemed to be University)
Bhubaneswar, India

Spring 2024-2025

Table of Content

Experiment Number	2
Experiment Title	Machine Learning and Deep Learning for Cat and Dog Classification
Date of Experiment	9/1/2025
Date of Submission	20/1/2025

1. Objective:- To classify images as cats or dogs using machine learning models.

2. Procedure:-

1. Collect a labeled dataset of cat and dog images.
2. Preprocess images using OpenCV (resize, flatten, etc.).
3. Train ML models: SVM, Random Forest, Logistic Regression, CNN, and K-means
Clustering
4. Save the trained models.
5. Build a Flask backend to load models and handle image uploads.
6. Create a frontend with HTML/CSS for uploading images and selecting models.

3. Code:-

CNN and Random Forest models are in git hub:
https://github.com/bhairavganguly/AD_LAB2.git

flask:

```
from flask import Flask, request, jsonify, render_template
import cv2
import numpy as np
#from tensorflow.keras.models import load_model
import joblib
app = Flask(__name__)

# Load models
```

```
# cnn_model = load_model('my_model.h5')
rf_model = joblib.load('random_forest_model.joblib')
def preprocess_image(image, model_type):
    if model_type == 'cnn':
        img = cv2.resize(image, (256, 256))
        img = img.astype('float32') / 255.0
        img = img.reshape((1, 256, 256, 3))
    else:  # random forest
        img = cv2.resize(image, (64, 64))
        img = img.astype('float32') / 255.0
        img_flat = img.flatten().reshape(1, -1)
        return img_flat
    return img
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/predict', methods=['POST'])
def predict():
    try:
        file = request.files['file']
        model_type = request.form['model']
        # Read and preprocess image
        nparr = np.fromstring(file.read(), np.uint8)
        image = cv2.imdecode(nparr, cv2.IMREAD_COLOR)
        if image is None:
            return jsonify({'result': 'Error', 'confidence': 0})
        processed_image = preprocess_image(image, model_type)
        if model_type == 'cnn':
            prediction = cnn_model.predict(processed_image)[0][0]
            result = 'Dog' if prediction > 0.5 else 'Cat'
            confidence = float(prediction if prediction > 0.5 else 1 - prediction)
        else:  # random forest
            prediction = rf_model.predict(processed_image)[0]
            probability = rf_model.predict_proba(processed_image)[0]
            result = 'cat' if prediction == 0 else 'dog'  # Note the change here
to match your RF model
            confidence = float(probability[1] if prediction == 0 else
probability[0])

        return jsonify({
```

```
            'result': result,
            'confidence': confidence
        })
    except Exception as e:
        print(f"Error: {str(e)}")
        return jsonify({'result': 'Error', 'confidence': 0})
if __name__ == '__main__':
    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Cat VS Dog Image Classifier</title>
    <style>
        body {
            font-family: 'Roboto', sans-serif;
            margin: 0;
            padding: 0;
            background-color: #f4f4f9;
            color: #333;
        }
        .container {
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: center;
            min-height: 100vh;
            padding: 20px;
        }
        h1 {
```

```
    font-size: 2.5rem;
    margin-bottom: 20px;
    color: #4CAF50;
}
.upload-section {
    background: #fff;
    border-radius: 10px;
    padding: 20px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
    width: 100%;
    max-width: 400px;
    text-align: center;
}
input[type="file"] {
    display: none;
}
label {
    display: inline-block;
    background-color: #4CAF50;
    color: #fff;
    padding: 10px 20px;
    font-size: 1rem;
    border-radius: 5px;
    cursor: pointer;
    margin-bottom: 15px;
}
select, button {
    width: 100%;
    padding: 10px;
```

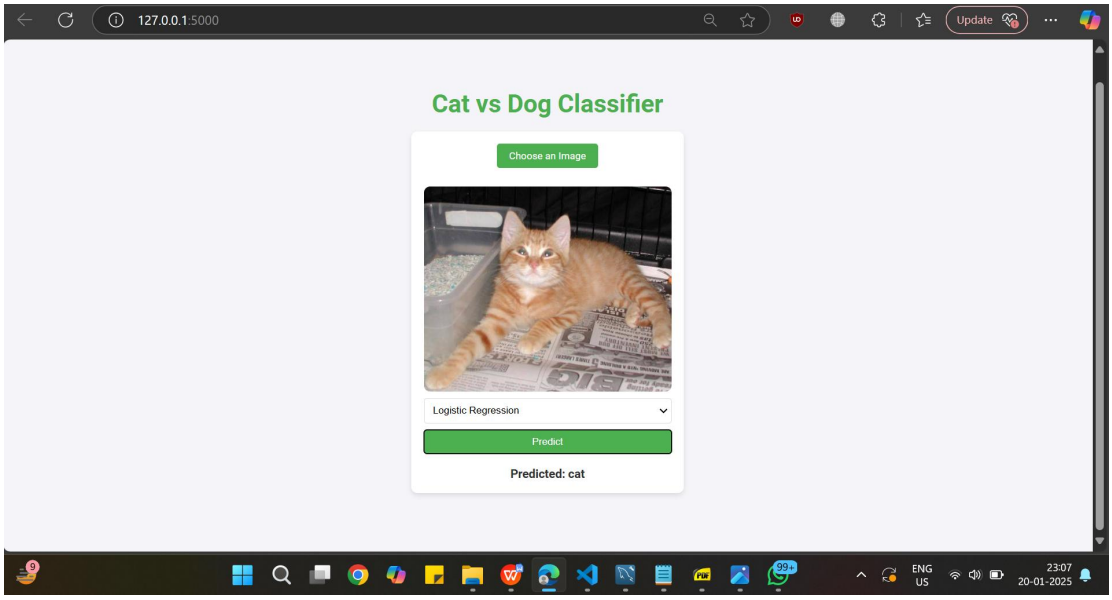
```
    margin-top: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
    font-size: 1rem;
}
button {
    background-color: #4CAF50;
    color: #fff;
    border: none;
    cursor: pointer;
    transition: background-color 0.3s;
}
button:hover {
    background-color: #45a049;
}
#imagePreview {
    margin-top: 15px;
    max-width: 100%;
    height: auto;
    border-radius: 10px;
    display: none;
}
.result {
    margin-top: 20px;
    font-size: 1.2rem;
    font-weight: bold;
}
@media (max-width: 480px) {
    h1 {
```

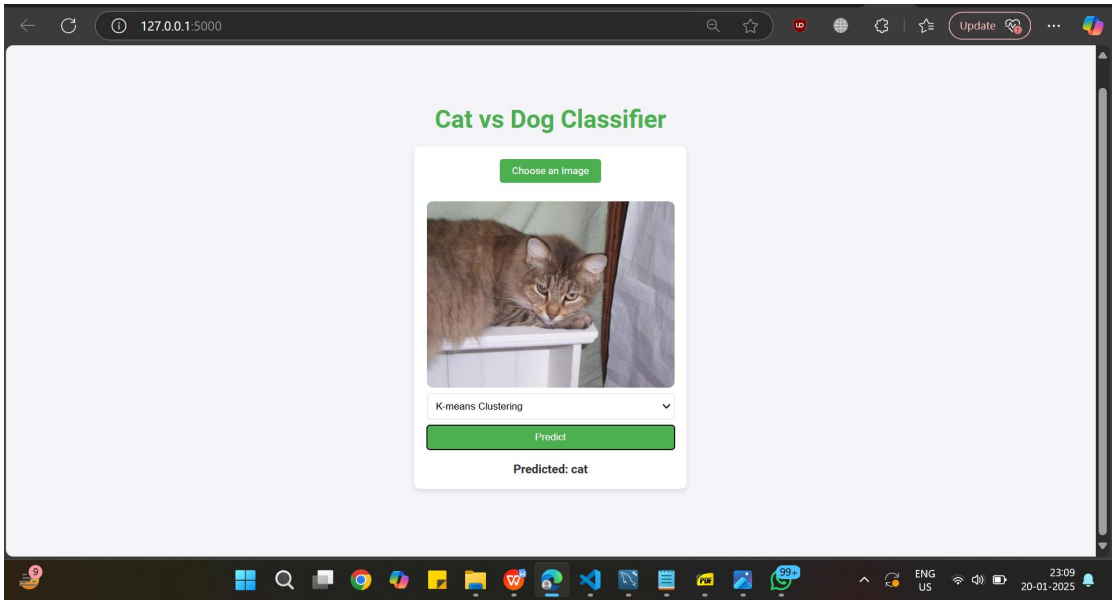
```
            font-size: 2rem;
        }
        .upload-section {
            padding: 15px;
        }
        label, select, button {
            font-size: 0.9rem;
        }
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>Cat vs Dog Classifier</h1>
    <div class="upload-section">
      <label for="imageUpload">Choose an Image</label>
      <input type="file" id="imageUpload" accept="image/*">
      <img id="imagePreview" alt="Image Preview">
      <select id="modelSelect">
        <option value="cnn">CNN Model</option>
        <option value="logistic">Logistic Regression</option>
        <option value="kmeans">K-means Clustering</option>
        <option value="random_forest">Random Forest</option>
      </select>
      <button onclick="predict()">Predict</button>
      <div id="result" class="result"></div>
    </div>
  </div>
```

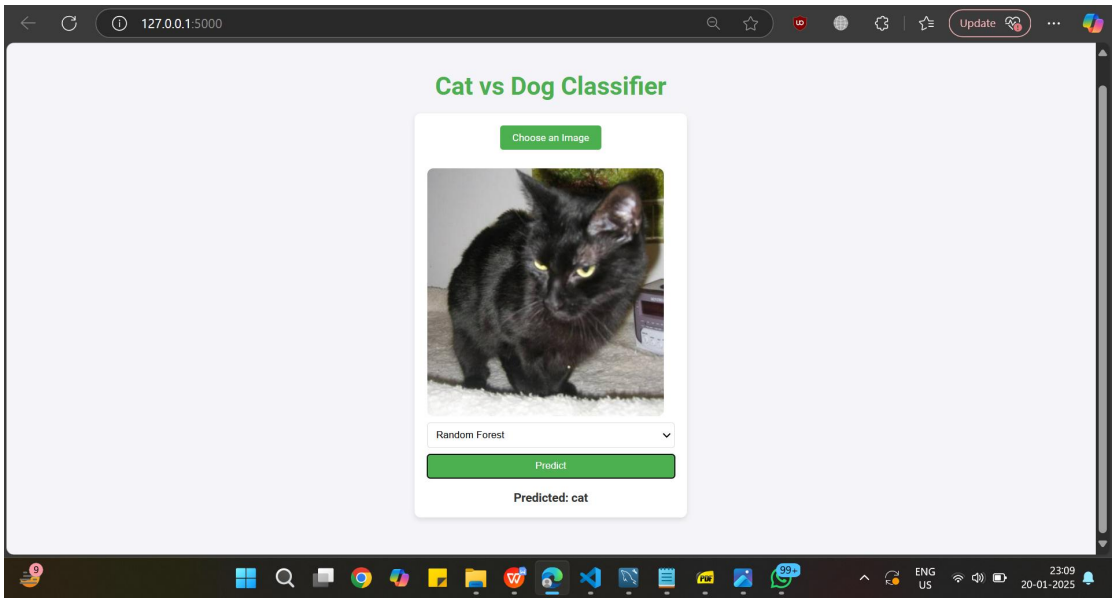
```
<script>

document.getElementById('imageUpload').addEventListener('change',
function(e) {
        const preview = document.getElementById('imagePreview');
        preview.style.display = 'block';
        preview.src = URL.createObjectURL(e.target.files[0]);
    });

    function predict() {
        const fileInput = document.getElementById('imageUpload');
        const modelSelect = document.getElementById('modelSelect');
        const resultDiv = document.getElementById('result');

        if (!fileInput.files[0]) {
            alert('Please select an image first');
            return;
        }

        const formData = new FormData();
        formData.append('file', fileInput.files[0]);
        formData.append('model', modelSelect.value);

        fetch('/predict', {
            method: 'POST',
            body: formData
        })
        .then(response => response.json())
        .then(data => {
```

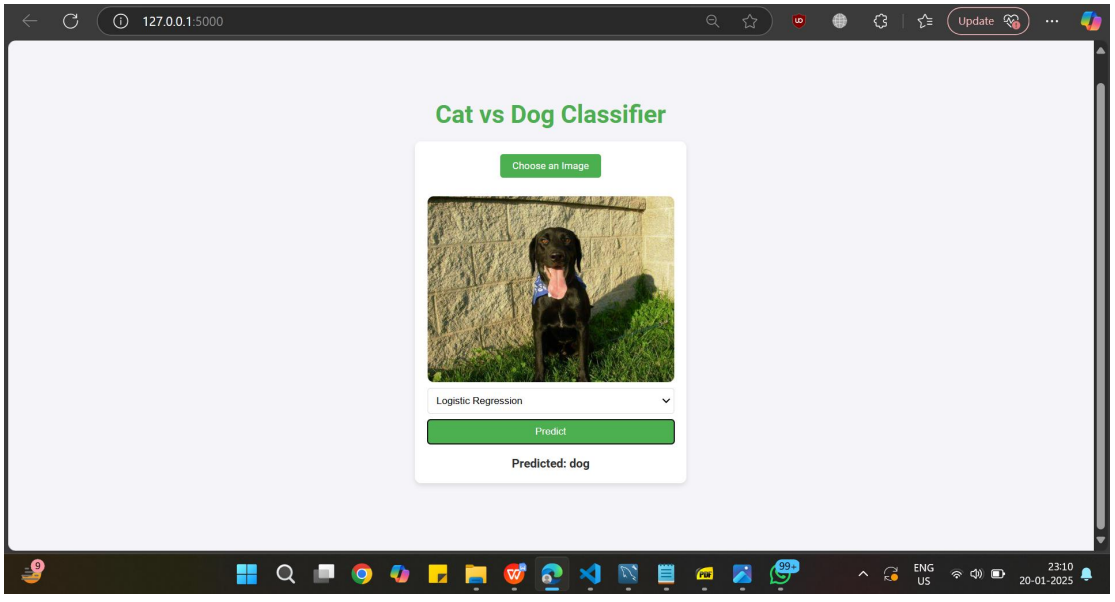
```
            resultDiv.textContent = `Predicted: ${data.result}`;
        })
        .catch(error => {
            console.error('Error:', error);
            resultDiv.textContent = 'Error processing image';
        });
    }
  </script>
</body>
</html>
```

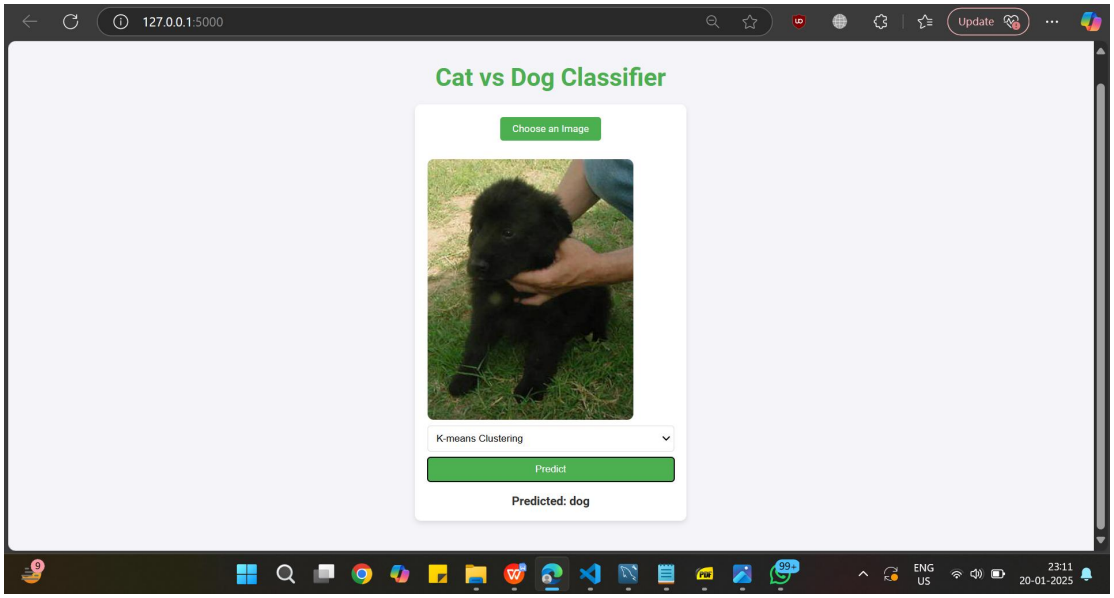
4. Results/Output:-

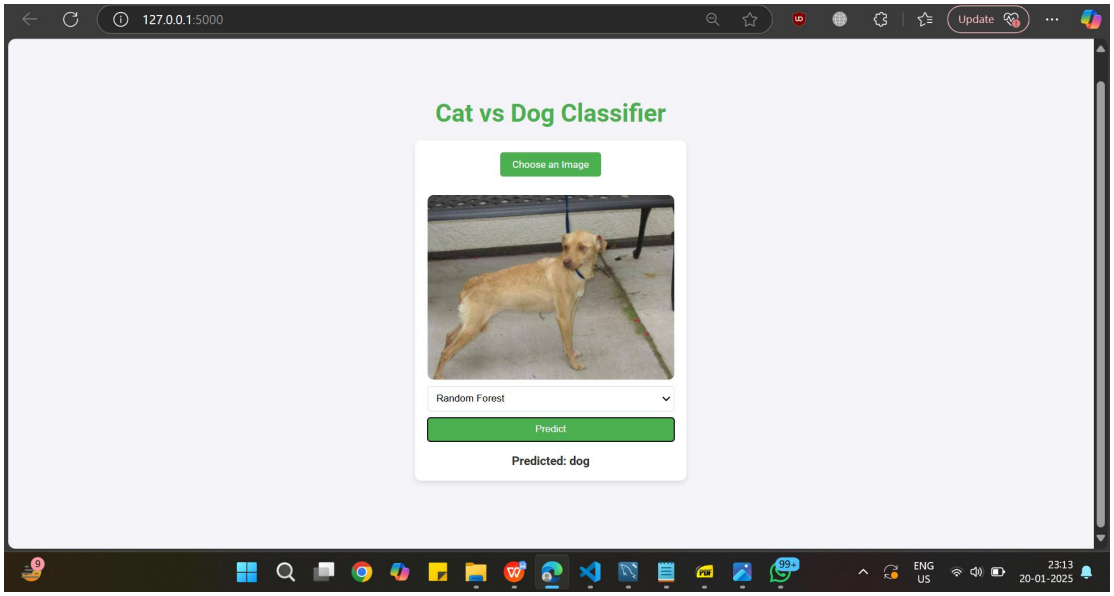



127.0.0.1:5000
Update
Cat vs Dog Classifier
Choose an Image
K-means Clustering
Predict
Predicted: cat
ENG
US
23:09
20-01-2025


127.0.0.1:5000
Update
Cat vs Dog Classifier
Choose an Image
Random Forest
Predict
Predicted: cat
ENG
US
23:09
20-01-2025


127.0.0.1:5000
Update
Cat vs Dog Classifier
Choose an Image
Logistic Regression
Predict
Predicted: dog
ENG
US
23:10
20-01-2025





5. Remarks:-

Signature of the Student
Bhairav Ganguly
(Name of the Student)

Signature of the Lab Coordinator
_____
(Name of the Coordinator)