LABORATORY REPORT

Application Development Lab (CS33002)

B.Tech Program in ECSc

Submitted By

Name:-Anubrata Mukhopadhyay

Roll No: 2230311

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	14/01/25
Date of Submission	21/01/25

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:-

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 = 'Dog' if prediction == 0 else 'Cat'  # 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)
```

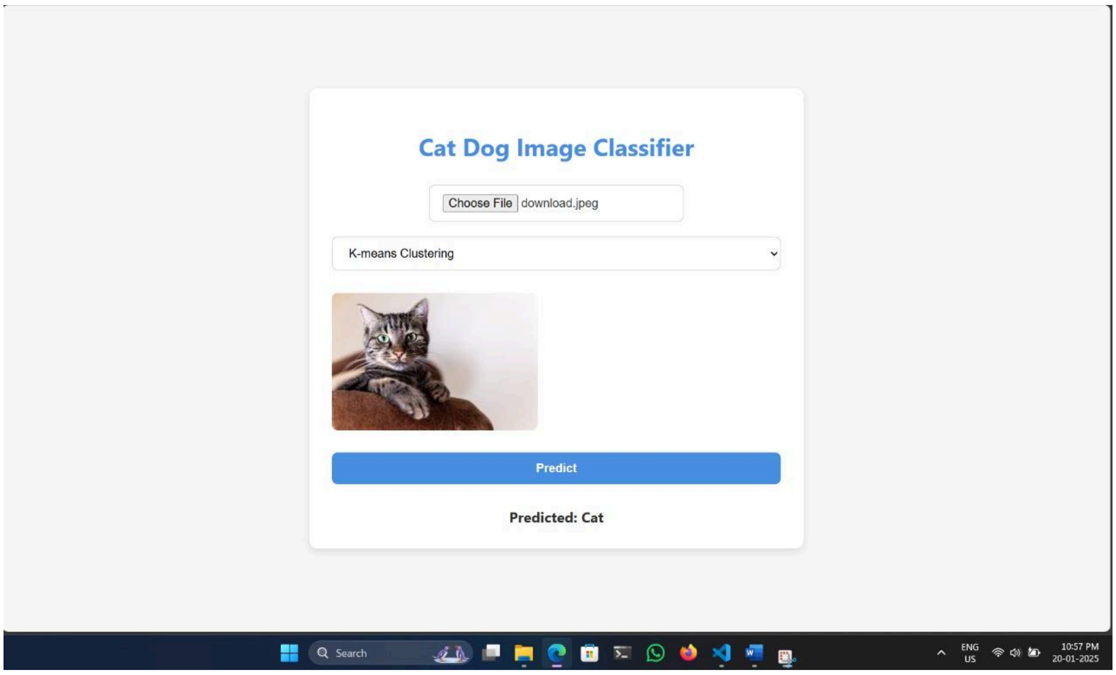
index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Cat Dog Classifier</title>
    <style>
        body {
            font-family: Arial;
            max-width: 800px;
            margin: 0 auto;
            padding: 20px;
        }
        .container {
            text-align: center;
        }
        select, button {
            margin: 10px;
            padding: 8px;
        }
        #imagePreview {
            max-width: 300px;
            margin: 10px auto;
        }
        .result {
            font-weight: bold;
            margin-top: 20px;
        }
        .confidence {
            color: #666;
            font-size: 0.9em;
            margin-top: 5px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Cat Dog Image Classifier</h2>
        <input type="file" id="imageUpload" accept="image/*">
        <br>
        <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>

        <br>

        <img id="imagePreview" style="display: none;">

        <br>

        <button onclick="predict()">Predict</button>

        <div id="result" class="result"></div>

        <div id="confidence" class="confidence"></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');
const confidenceDiv = document.getElementById('confidence');

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';
    confidenceDiv.textContent = '';
});
```

```
        }
    </script>
</body>
</html>
```

4. Results/Output:-

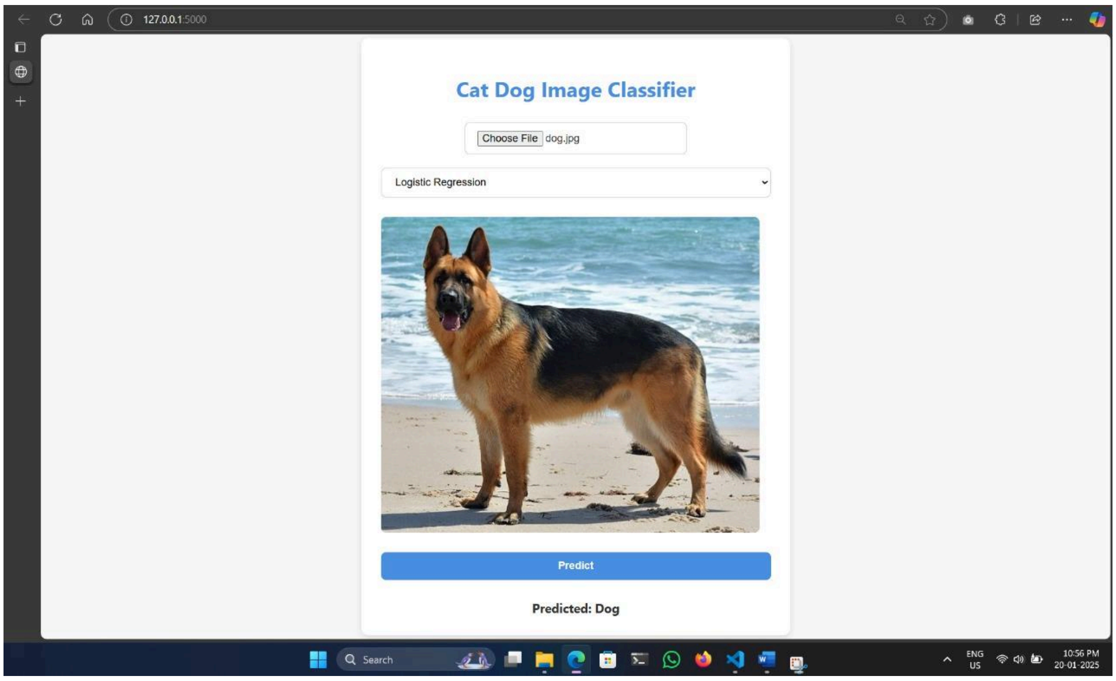

Cat Dog Image Classifier

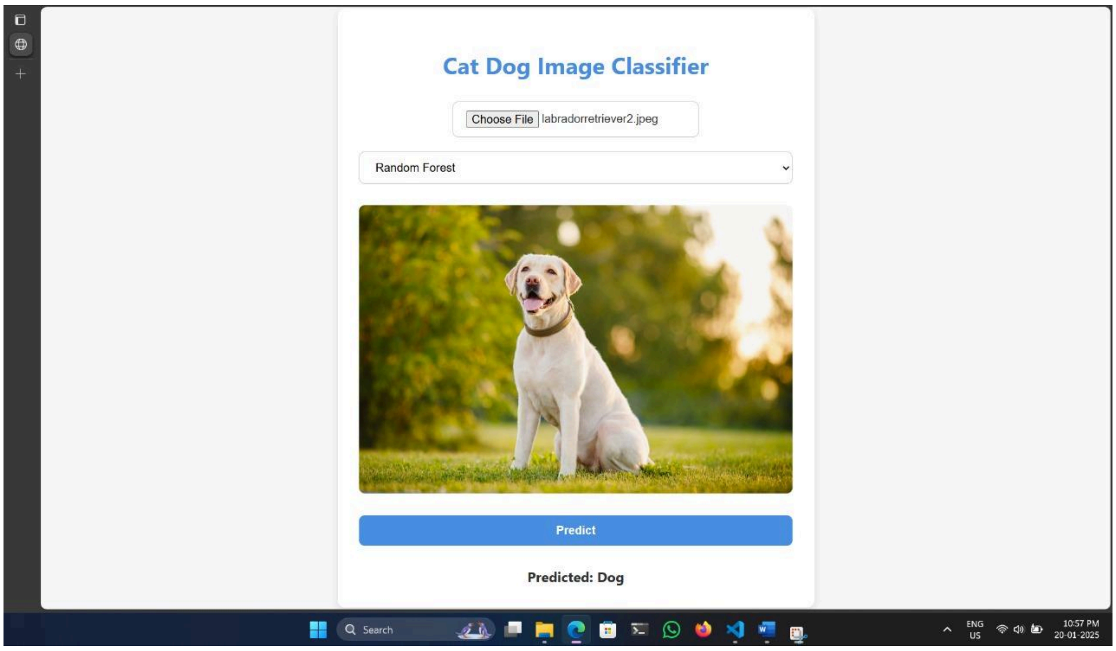
Choose File download.jpeg

K-means Clustering

Predict

Predicted: Cat





GITHUB LINK -

https://github.com/Anubrata11/Application-Designs-Lab

5. Remarks:-

Signature of the Student	Signature of the Lab Coordinator
______________________	______________________________________
(Name of the Student)	(Name of the Coordinator)