Project Title	**Human Faces (Object Detection)**
Skills take away From This Project	**Python, Analytics, statistics, Ploting, streamlit, machine learning, Deep Learning, GenAI**
Domain	**Computer vision**

Problem Statement:

Create a system to detect human faces in photos and videos quickly and accurately. It should work well in different lighting and angles, and handle face detection in real-time. This will improve uses in security, identity checks, and user interactions.

Business Use Cases:

1. **Security**: Monitor and identify people in public spaces.
2. **Access Control**: Secure access to buildings and devices.
3. **Retail**: Analyze customer emotions and demographics.
4. **Healthcare**: Track patient conditions and detect distress.
5. **Automotive**: Monitor driver attention and safety.
6. **Entertainment**: Enhance gaming and virtual reality experiences.

Approach:

Step 1: Data Preprocessing

- Remove duplicate or irrelevant images and correct any incorrect annotations.
- Scale images to a consistent size suitable for the model (e.g., 224x224 pixels).
- Adjust pixel values to a standard range (e.g., 0 to 1) for consistent input to the model.
- Apply techniques like rotation, flipping, and color adjustment to increase dataset diversity.

Step 2: Exploratory Data Analysis (EDA)

- **Image Count:** Determine the total number of images and faces in the dataset.
- **Face Count**: Calculate the number of faces per image to understand the density of faces.
- **Bounding Box Accuracy**: Check if bounding boxes are correctly placed around faces.
- **Label Consistency**: Ensure annotations are accurate and consistent.
- **Resize Requirements**: Identify if resizing or cropping is needed.
- **Resolution**: Evaluate the resolution of images to ensure they are clear and suitable for detection.

Step 3: Feature Engineering

- **Bounding Box Coordinates:** Use the coordinates of bounding boxes to define the location of faces in images.
- **Face Landmarks:** Extract facial landmarks (e.g., eyes, nose, mouth) for more detailed face detection.
- **Histogram Equalization**: Enhance image contrast to improve face visibility.
- **Normalization**: Scale pixel values to a standard range for uniform model input.
- **HOG (Histogram of Oriented Gradients)**: Extract edge and gradient information for face detection.
- **LBP (Local Binary Patterns)**: Capture texture patterns to identify facial features.

Step 4: Split Data into Training and Test Sets

- Divide your dataset into two parts:
- **Training set:** Used to train the classification model.
- **Test set:** Used to evaluate the model's performance on unseen data.

Step 5: Choose a Classification Model

- Select an appropriate algorithm based on your problem type, dataset size, and performance requirements:
- Common algorithms include YOLO, Faster R-CNN, Multi-task Cascaded Convolutional Networks

Step 6: Train the Model

- **Initialize Training**: Start the training process with your dataset.
- **Monitor Progress**: Track metrics such as loss, accuracy, and IoU (Intersection over Union) during training.
- **Validate**: Periodically evaluate the model on the validation set to ensure it's learning effectively.

Step 7: Evaluate Model Performance

- **Test Performance**: Assess the model's performance on the test set using metrics like precision, recall, F1-score, and mAP (mean Average Precision).
- **Check for Overfitting**: Compare performance on training and validation sets to identify potential overfitting.

Step 8: Model Deployment and Monitoring

- **Integrate**: Deploy the model into your application or system.
- **Test in Real-world Scenarios**: Validate performance in real-world conditions and make any necessary adjustments.

Step 9: Iterate and Improve

Model tuning and hyper parameter tuning.

Results:

Result should be in streamlit app as follows.

Sidebar Number	Sidebar Name	Function to do
1	Data	1. Show the dataset which you are using for model building. 2. Show the model performance metric dataset.
2	EDA - Visual	1. Do all kind of EDA analysis for the given dataset. 2. Plot the EDA plots and Statistics plots and show them in this menu. 3. Plot preferred are Plotly or seaborn, try to plot in plotly for interaction.
3	Prediction	1. Create a text or number input items in streamlit for selected featured which you have selected for model building. 2. Once the user has entered the value and hit the button of predict, you model should predict and show whether the customer is defaulted or not.

Project Evaluation metrics:

Precision, recall, accuracy and F1 score should be greater than (>) 85 %.

All required preprocessing has to be completed and feature selection should be done.

Technical Tags:

Open CV

Data Preprocessing

Feature Engineering

Model Training

Model Evaluation

Hyperparameter Tuning

Deep Learning

Data Set:

https://drive.google.com/drive/folders/1cNc_nWMmPgqwQSul47OxWyyX_6gKaK8n?usp=drive_link

Data Set Explanation:

A dataset for human face detection includes images with annotated bounding boxes around faces. Key components are diverse images with varied lighting, angles, and demographics. Annotations typically consist of bounding box coordinates and optionally facial landmarks. The dataset is split into training, validation, and test sets. Proper data preparation involves cleaning, balancing, and augmenting the dataset to enhance model performance. Data looks like layman.

Project Deliverables:

Submission Requirements
Source Code: The complete code used for data preprocessing, model training, and evaluation.
Documentation: A report detailing the methodology, analysis, results, and insights.
Presentation: A slide deck summarizing the project and key findings.
Model Files: The trained model ready for deployment.
README: Instructions on how to run the code and reproduce the results.

Project Guidelines:

Best Practices
Coding Standards: Standard code standard for Python code.
Version Control: Use Git for version control and regularly commit changes.
Documentation: Comment your code and provide clear explanations for your logic.
Collaboration: Use collaborative tools like GitHub or GitLab for team projects.

Timeline:

Analyse data EDA Ploting Building Model Open CV Model Selection	1 weeks
Classification	4 Days
Streamlit	3 Days
Total	2 weeks