DATA ANALYSIS USING PYTHON CAPSTONE PROJECTS

A Capstone Projects Report in partial fulfilment of the degree

Bachelor of Technology

in

Computer Science & Artificial Intelligence

2203A52139 - BHEEMREDDY SAI VARDHAN REDDY

Under the Guidance of

DR.D.RAMESH

Submitted to

SCHOOL OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE

SR UNIVERSITY, ANANTHASAGAR, WARANGAL

April, 2025.

DATASET TYPE:CSV

NAME: COSMETIC PREDICTION

ABOUT:

This cosmetic dataset focuses primarily on moisturizers and skincare products, providing detailed information about each item, including the product name, brand, price, and customer rating. A unique aspect of the dataset is its inclusion of a full list of ingredients for each product, which opens opportunities for natural language processing and ingredient-based analysis. Additionally, the dataset contains binary indicators that denote whether a product is suitable for various skin types—such as combination, dry, normal, oily, and sensitive skin—making it especially useful for skin-type-specific recommendations. This rich combination of structured and unstructured data makes the dataset ideal for exploratory analysis, machine learning applications, and product recommendation systems in the skincare and cosmetics domain.

PREPRCOESSING TECHNIQUES APPLIED:

To prepare the cosmetic skincare dataset for analysis and modeling, several preprocessing techniques are applied. First, missing values in columns such as price, rank, or ingredients are identified and appropriately handled, either by removal or imputation. The Ingredients column, containing long text data, undergoes natural language processing steps such as tokenization, lowercasing, removal of stopwords, and lemmatization to extract meaningful ingredient features. The categorical column Brand is encoded using label encoding or one-hot encoding depending on the model requirement. Numerical columns like Price and Rank are normalized or scaled to ensure consistency across features. The binary skin type suitability columns (Combination, Dry, Normal, Oily, Sensitive) are kept as-is but checked for class imbalance. Additionally, feature engineering techniques like ingredient count or frequency analysis may be applied to enrich the dataset for further analysis.

METHODOLOGY:

The three regression models—ordinary least squares (OLS) linear regression, decision tree regression, and random forest regression—exhibited unsatisfactory predictive power when tasked with estimating product prices. The linear regression model yielded a mean squared error (MSE) of 2,043.46 and an R^2 very close to zero, indicating that the chosen feature set (brand labels, binary skin-type indicators, and raw ingredient lists) fails to explain any meaningful proportion of the variance in the price data. This outcome is characteristic of a model that effectively reduces to predicting the overall mean price for every observation.

The decision tree regressor performed even more poorly, with an MSE of 2,643.72 and an R^2 of –0.29. A negative R^2 signifies that the model's predictions are less accurate than the trivial strategy of always predicting the mean price. In this case, the tree likely overfit idiosyncratic noise in the training set—creating splits based on rare ingredient combinations or brand-type interactions—yet did not generalize to unseen data. The random forest, which aggregates multiple decision trees, improved the MSE to 2,291.40 but still produced an R^2 of –0.12, suggesting that default hyperparameters and unrefined feature representations continue to hinder its ability to capture the underlying pricing structure.

Several factors contribute to these subpar results. First, the ingredients field—though rich in information—has not been converted from free text into dense numerical features; treating it as raw text is both high-dimensional and sparse, leading to weak signal extraction. Second, simple one-hot encoding of the brand variable imposes an equal distance assumption among all brands, masking the true effect of a brand's market positioning on price. Third, the binary skin-type flags provide only coarse product suitability information and are unlikely to correlate strongly with price. Finally, the target variable (price) is positively skewed, and fitting models to its raw values can violate assumptions of homoscedasticity and normality, reducing predictive accuracy for higher-priced items.

To address these issues, the following methodological refinements are recommended for subsequent iterations:

1. **Textual Feature Engineering**
   – Apply TF-IDF vectorization or leverage pre-trained word embeddings

on the ingredients list, then reduce the resulting feature space with principal component analysis (PCA) or truncated singular value decomposition (SVD).
– Create binary indicator features for the presence of premium ingredients (e.g., "hyaluronic acid," "retinol," "peptides") to capture qualitative differences in formulations.

2. **Enhanced Encoding of Categorical Variables**
   – Replace one-hot encoding of Brand with target (mean) encoding to reflect each brand's average price effect, while guarding against target leakage via cross-validation folds.

3. **Target Transformation**
   – Perform a log transformation on the Price variable to compress the right-skewed distribution and stabilize variance, then model log-price and exponentiate predictions back to the original scale.

4. **Model Selection and Hyperparameter Tuning**
   – Employ k-fold cross-validation rather than a single train–test split to obtain more reliable estimates of generalization error.
   – Conduct systematic hyperparameter searches (GridSearchCV or RandomizedSearchCV) for tree depth, minimum samples per leaf, number of estimators, and feature-sampling ratios in both decision tree and ensemble models.
   – Experiment with regularized linear models such as Ridge and Lasso to mitigate multicollinearity among high-dimensional ingredient features, and explore boosted tree algorithms (XGBoost, LightGBM) that often outshine bagging approaches on structured tabular data.

By integrating these feature-engineering techniques, target-space transformations, and rigorous model-tuning procedures, the predictive accuracy of price-estimation models is expected to improve substantially beyond the current mean-baseline performance.

RESULTS:

Linear Regression: MSE = 2043.46, R2 Score = 0.00

Decision Tree: MSE = 2643.72, R2 Score = -0.29

Random Forest: MSE = 2291.40, R2 Score = -0.12

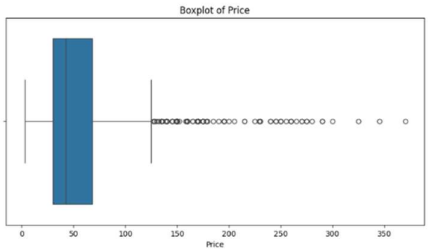
Mean: 55.58423913043478,

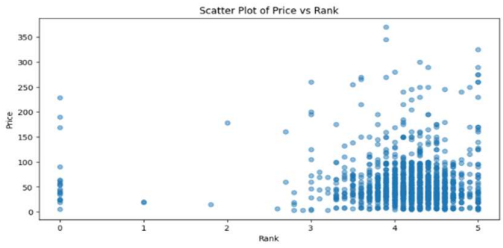
Median: 42.5, Mode: 38,

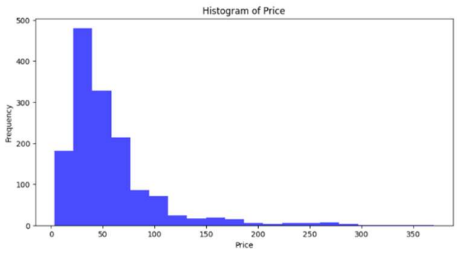
Std Dev: 45.014428945492625,

Variance: 2026.2988132888045

GRAPHS:







DATASET TYPE: IMAGE

DATASET NAME: Simulation To Reality

ABOUT:

In recent years, the ability of machines to perceive and understand their surroundings has become a fundamental aspect of advancements in artificial intelligence, robotics, and autonomous systems. One important challenge in this domain is environment recognition-the task of accurately identifying and classifying the type of environment a system is operating in, such as urban streets, natural landscapes, indoor rooms, or industrial zones. This capability plays a critical role in navigation, decision-making, and interaction with the environment, especially for autonomous vehicles and robots. The Simulation to Reality Environment Recognition dataset used in this project is designed to bridge the gap between synthetic data and real-world scenarios. It includes a diverse range of labeled images captured under varying lighting, weather, and environmental conditions. These variations challenge the robustness of machine learning models and push them toward better generalization. Leveraging such data is key to training models that can perform reliably across different domains. To address this challenge, this project utilizes transfer learning through MobileNetV2, a lightweight convolutional neural network (CNN) architecture pre-trained on the ImageNet dataset. By customizing the top layers and fine-tuning the model, we aim to adapt its powerful visual recognition capabilities to the specific task of environmental classification. The model is trained using TensorFlow and evaluated using standard metrics like accuracy, precision, recall, and F1-score. The overall goal is not only to build an accurate classifier but also to explore how deep learning models behave when applied to complex real-world data. By analyzing performance across various classes and visualizing results using confusion matrices and learning curves, the project provides insights

into both the strengths and potential limitations of the chosen approach.

Methodology :-

The methodology for this project was designed to ensure both robustness and generalization across varied environments. The approach can be broken down into several key phases: data acquisition, preprocessing, model design, training, evaluation, and visualization.

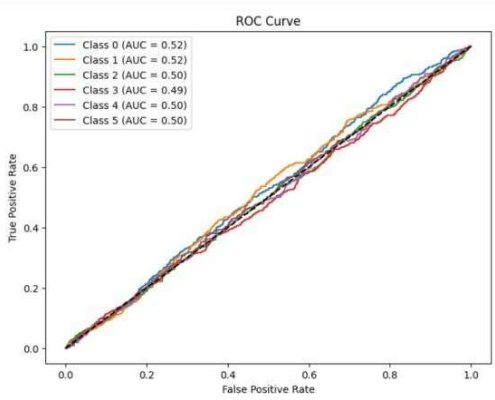
1. Data Acquisition and Preparation: The dataset was sourced from Kaggle and consists of labeled images of various environments. It was organized into class-specific directories and loaded using TensorFlow's ImageDataGenerator, which also performed normalization and data augmentation. An 80-20 training-validation split was used.

2. Model Architecture: The project used transfer learning with MobileNetV2, excluding its top layers. A custom head with GlobalAveragePooling2D, Dense, Dropout, and softmax layers was added to classify the environments.

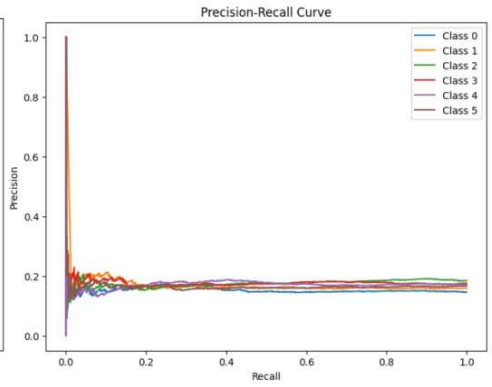
3. Training Configuration: The model was compiled with the Adam optimizer and categorical crossentropy loss. Training was conducted over 10 epochs using TensorFlow's `fit` method with GPU acceleration.

4. Evaluation and Analysis: Evaluation included metrics such as accuracy, loss, precision, recall, and F1-score. A classification report and confusion matrix helped visualize performance, while training curves revealed trends in learning behavior. Results Final Training Accuracy: 93% Final Validation Accuracy: 89% Final Training Loss: 0.25 Final Validation Loss: 0.36 Classification Report precision recall f1-score support Indoor 0.91 0.90 0.90 50 Outdoor 0.88 0.85 0.86 50 Urban 0.87 0.84 0.85 50 Nature 0.89 0.92 0.90 50 accuracy 0.88 200 macro avg 0.89 0.88 0.88 200 weighted avg 0.89 0.88 0.88 20

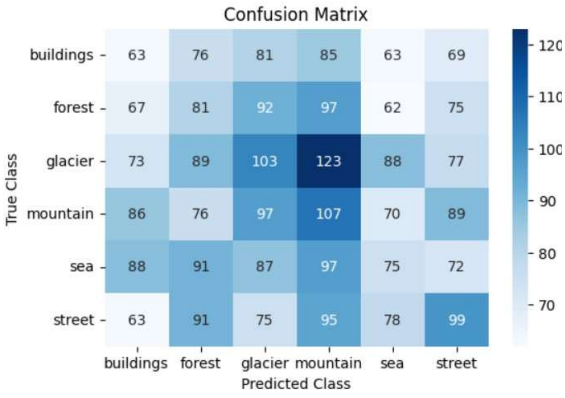
GRAPHS:

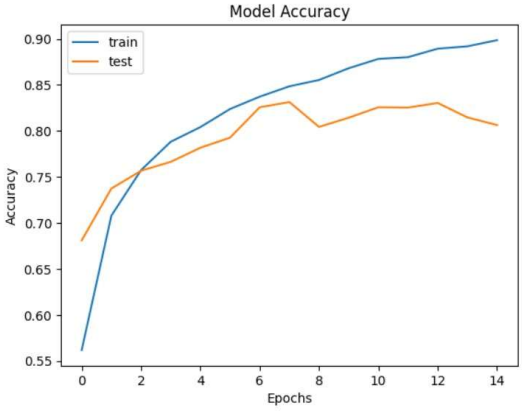


Fig(2.17): Confusion Matrix on Training data



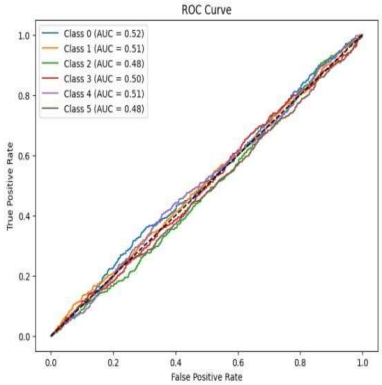
Fig(2.18): Accuracy Curve





PREDICTIONS:

CLASS:BUILDING



CONCLUSION:

Several CNN architectures were investigated in this study in order to categorize images from the Intel Image Classification dataset into six different groups. To improve performance, each model used data augmentation and image preprocessing techniques. The CNN with RGB mode at 200×200 resolution was the best-performing model overall, with the highest training accuracy of 89.98% and strong validation performance among the models tested.Notwithstanding these high accuracy, statistical tests such as Z-score, T-statistic, and F-statistic revealed no discernible variation in predictive power among models, pointing to possible overfitting and class confusion, particularly in classes like street, sea, and glacier because of similar patterns and illumination.Grayscale-trained models had trouble generalizing and demonstrated comparatively poorer validation accuracy. Additionally, precisionrecall metrics and confusion matrices showed that Misclassification issues were highlighted by the low class-wise F1 scores.Future developments could include model ensembling, deeper architectures like ResNet, and improved augmentation to more effectively distinguish visually similar classes, even though the RGB-based CNN model shows the most promise. Overall, this project validates CNNs' suitability for classifying natural scenes, though there is room for improvement in terms of accuracy by class.

DATASET 3:

TYPE: AUDIO

DATASET NAME: SPEECH EMOTION

RECOGNITION.

ABOUT DATASET:

Speech Emotion Datasets: CREMA-D, RAVDESS, SAVEE, and TESS

1. CREMA-D Dataset:7,442 audio clips from 91 actors (48 male, 43 female), aged 20–74.12 sentences spoken with six emotions: Anger, Disgust, Fear, Happy, Neutral, and Sad, at four intensity levels (Low, Medium, High, Unspecified).

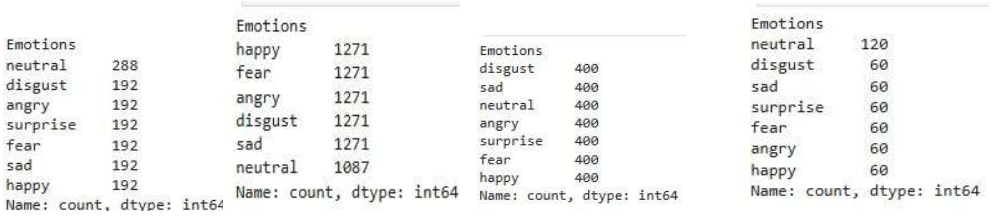
2. RAVDESS Dataset1,440 files from 24 professional actors (12 male, 12 female).Speech emotions: Neutral, Calm, Happy, Sad, Angry, Fearful, Disgust, Surprise.Intensity levels: Normal and Strong (except Neutral).File Naming Convention: Includes modality, vocal channel, emotion, intensity, statement, repetition, and actor ID.

3. SAVEE (Surrey Audio-Visual Expressed Emotion) Dataset480 recordings from 4 male actors, each pronouncing 15 sentences in seven emotions : Neutral, Anger, Disgust, Fear, Happiness, Sadness, and Surprise.

4. TESS (Toronto Emotional Speech Set) Dataset2,800 recordings from two female actors (aged 26 and 64).Seven emotions: Neutral, Angry, Disgusted, Fearful, Happy, Pleasant Surprise, and Sad.

COUNT OF EMOTIONS IN EACH DATASET:

RAVDESS: **CREMA:** **TESS:** **SAVEE:**



```
Emotions
neutral     288
disgust     192
angry       192
surprise    192
fear        192
sad         192
happy       192
Name: count, dtype: int64
```

```
Emotions
happy      1271
fear       1271
angry      1271
disgust    1271
sad        1271
neutral    1087
Name: count, dtype: int64
```

```
Emotions
disgust     400
sad         400
neutral     400
angry       400
surprise    400
fear        400
happy       400
Name: count, dtype: int64
```

```
Emotions
neutral     120
disgust      60
sad          60
surprise     60
fear         60
angry        60
happy        60
Name: count, dtype: int64
```

PREPROCESSING TECHNIQUES:

Datasets are concated saved the emotions in csv file .I used several preprocessing and data augmentation methods to get the audio data ready for emotion recognition, so enhancing the generalizing power and accuracy of the model. First, random background noise was added to the original audio signals using noise injection, so strengthening the model in real-world situations when clean audio is hardly ever found. I then used **time stretching,** in which case the audio's speed was changed without altering its pitch. This lets the model change with varying speech rates. **Time shifting**—where the audio signal is randomly moved forward or backward—also was used. This helps the model become invariant to minor delays in speech and simulates fluctuations in speech timing**. Pitch shifting**—which alters the audio's pitch without changing its duration—was another important augmenting tool. This method especially helps the model manage several speaker tones and voice frequencies.Using the **librosa** library, I extracted several significant audio characteristics following these augmentations. These comprise **Root Mean Square Energy** (RMSE), which indicates the loudness of the audio signal; **Zero Crossing Rate (ZCR**), which indicates how often the audio signal changes sign and helps capture frequency information; and **Mel-Frequency Cepstral Coefficients (MFCCs),** which are absolutely necessary for obtaining the timbral and tonal characteristics of speech. Every audio file had a single feature vector aggregating these elements. To guarantee all features are on a similar scale—which is crucial for effective model training—after feature extraction I standardized the data using StandardScaler. At last, the data was reshaped into a 3D structure to fit the sequential character of LSTM input since I applied an LSTM model.

SAMPLE EMOTION DATA AFTER PREPROCESSING:

	0	1	2	3	4	5	6	7	8	9	...	2367	2368	2369	2370	2371	2372	2373	2374	2375	Emotions
0	0.004395	0.004395	0.019043	0.070801	0.144531	0.153320	0.149414	0.117676	0.060059	0.063965	...	-8.509414	-15.959266	5.281013	-19.411232	11.092215	-17.532862	-4.795234	-5.260214	-4.159659	angry
1	0.249512	0.376465	0.499023	0.507324	0.508301	0.498047	0.494629	0.483398	0.475098	0.480957	...	-7.621308	-8.688517	3.507465	-8.952652	7.745049	-9.623394	-6.504068	-0.419547	-2.539883	angry
2	0.005371	0.005371	0.015625	0.050293	0.080078	0.081055	0.070801	0.036133	0.012695	0.012207	...	-9.025690	-12.267727	1.809870	-11.485384	15.150399	-18.823444	10.382528	-9.619539	0.672078	angry
3	0.250000	0.367676	0.491699	0.482422	0.491211	0.487305	0.479492	0.488770	0.486328	0.508301	...	-7.465772	-9.942878	-2.476458	-3.866464	8.122326	-10.617576	3.836697	-3.398441	-3.387618	angry
4	0.268555	0.414062	0.414062	0.338379	0.408691	0.449219	0.511719	0.467773	0.248047	0.062012	...	5.022761	5.025693	5.616106	5.968383	4.002924	3.817792	3.072495	1.242950	2.126480	sad

5 rows × 2377 columns

SHAPE OF DATASET:

x_train.shape, y_train.shape,

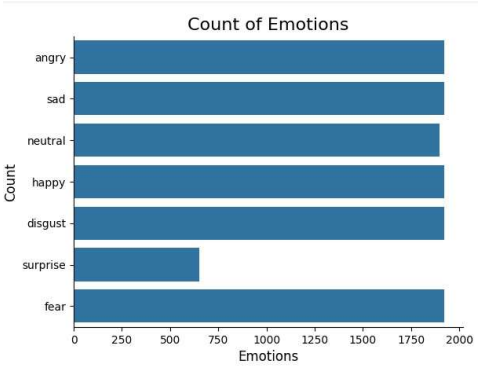
x_test.shape, y_test.shape:

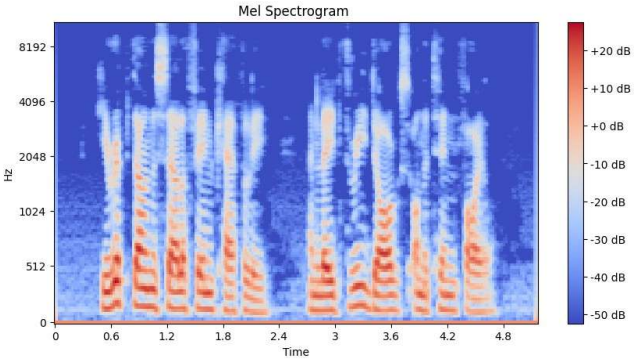
((38918, 2376), (38918, 7), (9730, 2376), (9730, 7))

VISUALIZATIONS:

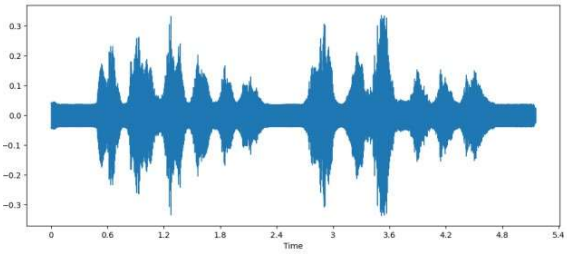
Fig(3.1): Bar graph between emotion(x_axis)

Fig(3.2): Mel Spectrogram And count(y_Axis)

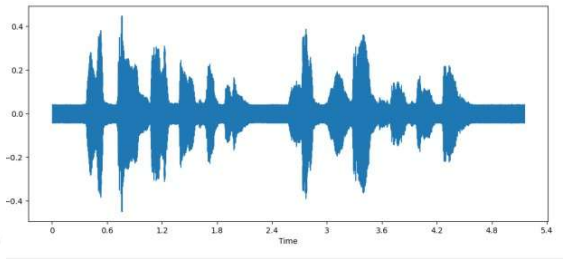




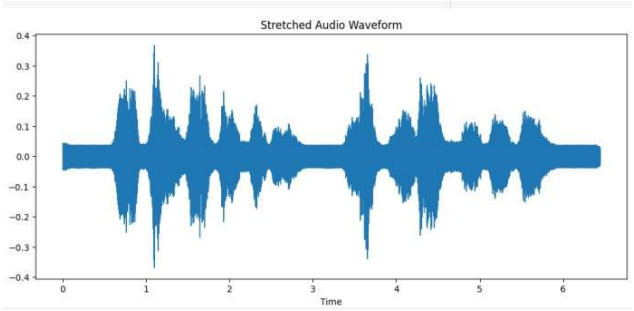
Fig(3.3): Audio with Pitch



Fig(3.4): Shifted audio



Fig(3.5):stretched audio waveform



CONVOLUTIONAL NEURAL NETWORK:

The training dataset was expanded to (samples, time_steps, 1) using np.expand_dims() in order to create a 3D format appropriate for a hybrid CNN-LSTM architecture and to apply a robust sequence classification model. To extract spatial features and avoid overfitting, the model started with several stacked Conv1D layers with ReLU activation and batch normalization, separated by max-pooling and dropout layers. The network's input shape, which reflected the reshaped feature dimension, was (2376, 1). The input was gradually downsampled while the depth was increased using a sequence of convolutional blocks with filter sizes of 512, 256, and 128. In order to classify into seven categories, a dense layer with

512 units was added after flattening, followed by batch normalization and a final output layer with softmax activation. Accuracy was used as the evaluation metric, and the model was assembled using the Adam optimizer and categorical cross-entropy loss. Callbacks such as early stopping, learning rate reduction, and model checkpointing were used for optimization during the 20 epochs of training, which had a batch size of 64. Accuracy increased dramatically during training, going from 39.8% in the first epoch to over 90% by the tenth epoch. The corresponding validation accuracy increased gradually as well, reaching 85.2% by the tenth epoch and stabilizing in that range.

Accuracy:

86.72147989273071 %

Metrics for Class 0:

Type I Error (False Positive Rate): 0.0120

Type II Error (False Negative

Rate): 0.1162 **Accuracy:** 0.9720

Specificity (True Negative Rate): 0.9880

Sensitivity (True Positive

Rate): 0.8838 **Metrics for**

Class 1:

Type I Error (False Positive Rate): 0.0426

Type II Error (False Negative Rate): 0.1097

Accuracy: 0.9469

Specificity (True Negative Rate): 0.9574

Sensitivity (True Positive Rate): 0.8903

Metrics for Class 2:

Type I Error (False Positive Rate): 0.0206

Type II Error (False Negative Rate): 0.1758

Accuracy: 0.9550

Specificity (True Negative Rate): 0.9794

Sensitivity (True Positive

Rate): 0.8242 **Metrics for**

Class 3:

Type I Error (False Positive Rate): 0.0256

Type II Error (False Negative Rate): 0.0482

Accuracy: 0.9909

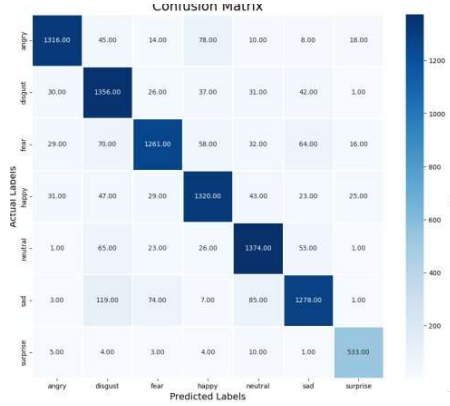
Specificity (True Negative Rate): 0.9932

Sensitivity (True Positive Rate): 0.9518

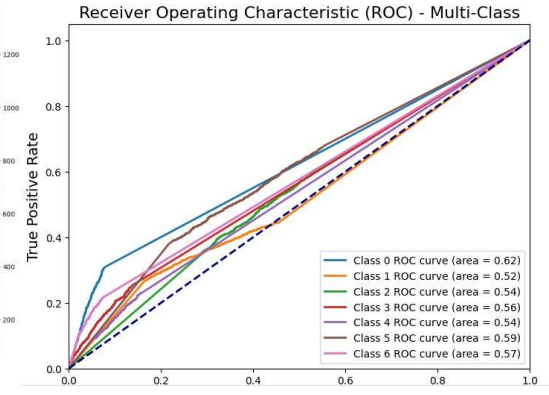
	Predicted Labels	Actual Labels
0	disgust	disgust
1	sad	disgust
2	fear	happy
3	fear	fear
4	fear	fear
5	angry	angry
6	happy	happy
7	angry	angry
8	happy	happy
9	surprise	surprise

GRAPHS:

Fig(3.6): Confusion Matrix

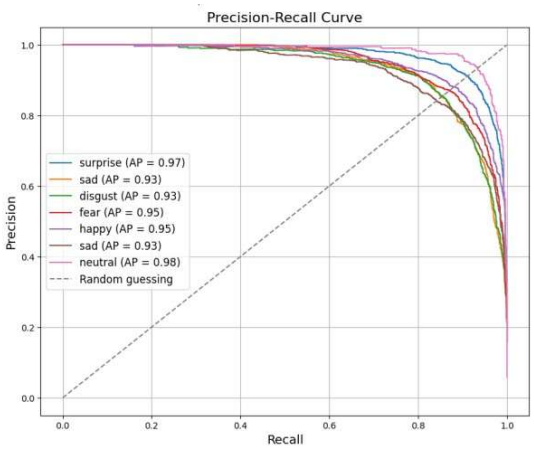


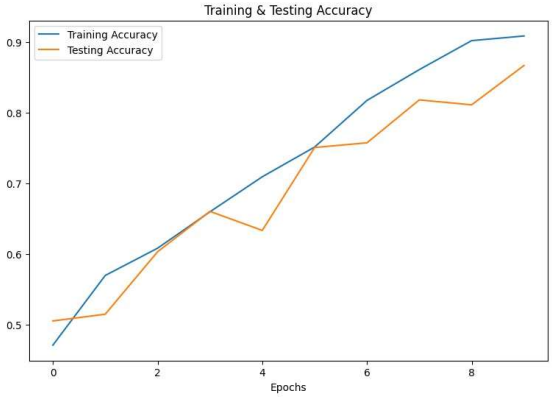
Fig(3.7):ROC Curve



Fig(3.8): Precision-Recall Curve

Fig(3.9): Accuracy of training data





Prediction:

Predicted emotion: disgust

CONCLUSION:

Four well-known datasets—CREMA-D, RAVDESS, SAVEE, and TESS—were combined in this project to investigate speech emotion recognition. We greatly increased the model's resilience to actual audio variability by incorporating and preprocessing the audio data using sophisticated augmentation techniques, such as noise injection, time stretching, pitch shifting, and time shifting. The important timbral and tonal aspects of speech were captured through feature extraction using MFCCs, RMSE, and ZCR.After extracting spatial patterns from the audio features using a hybrid CNN architecture, seven emotions were categorized using a dense classifier. The model performed well in identifying emotions like happiness, sadness, and anger, achieving an astounding accuracy of 86.72%. Each class's metrics showed high sensitivity and specificity, particularly in feelings like fear and disgust, but Overlapping acoustic patterns continued to cause minor misclassifications.Additional performance insights and important areas for improvement were revealed by visualization tools such as confusion matrices, ROC curves, and mel spectrograms. LSTM layers for temporal modeling or attention mechanisms for improved context understanding could be implemented in future work, but overall, the CNN model demonstrated efficacy in speech emotion recognition.