**Haircut Recommendation Model Documentation**

**Introduction**

The haircut recommendation model is built using machine learning techniques to suggest the best haircut based on a user’s attributes such as gender, face shape, hair type, occupation, and other relevant features. The model leverages **XGBoost (Extreme Gradient Boosting)**, an advanced machine-learning algorithm, to provide highly accurate recommendations.

This document explains the **libraries used**, **model selection**, **methods applied**, **expected outcomes**, **feature-label relationships**, **model configuration**, and other important aspects to help students understand the workflow.

**1. Libraries Used and Their Purpose**

| **Library** | **Purpose** |
| --- | --- |
| pandas | To handle dataset loading, manipulation, and preprocessing. |
| imblearn.over\_sampling.SMOTE | To balance the dataset by generating synthetic data points for underrepresented classes. |
| sklearn.model\_selection.train\_test\_split | To split the dataset into training and testing sets. |
| sklearn.metrics | To evaluate the model’s performance using accuracy score, classification reports, and confusion matrix. |
| xgboost | The machine learning library used to build the haircut recommendation model. |
| matplotlib.pyplot & seaborn | To visualize data insights, including confusion matrices. |
| joblib | To save and load the trained model and label encoders for later use. |

**2. Why Use XGBoost?**

**XGBoost (Extreme Gradient Boosting)** is chosen because:

* It is **highly efficient** and optimized for speed and performance.
* It **handles missing values** and large datasets well.
* It can **identify complex relationships** between features.
* It **prevents overfitting** using regularization.
* It **supports parallel computing**, making training faster.

**3. Why Use SMOTE for Data Balancing?**

SMOTE (Synthetic Minority Over-sampling Technique) is used because:

* It **generates synthetic samples** to balance the dataset.
* It **reduces bias** toward majority classes, ensuring better generalization.
* It **improves model performance**, especially when some haircuts are underrepresented.

**4. Methods Used in the Model**

**4.1 Data Preprocessing**

* **Missing Value Imputation:**
  + Handled using SimpleImputer with the "most frequent" strategy for categorical features.
* **Label Encoding:**
  + Converts categorical features into numerical format using LabelEncoder().
* **Feature Scaling:**
  + Normalizes numerical data (if present) using a scaler.

**4.2 Hyperparameter Tuning**

Instead of using default values, we manually **fine-tune the hyperparameters**:

| **Hyperparameter** | **Purpose** |
| --- | --- |
| n\_estimators | Defines the number of trees in the model. |
| max\_depth | Controls the depth of each tree (higher values can cause overfitting). |
| learning\_rate | Determines the step size of updating weights (lower values lead to better generalization). |
| subsample | Fraction of data used for each boosting round (helps prevent overfitting). |
| colsample\_bytree | Fraction of features used per tree (improves diversity in trees). |

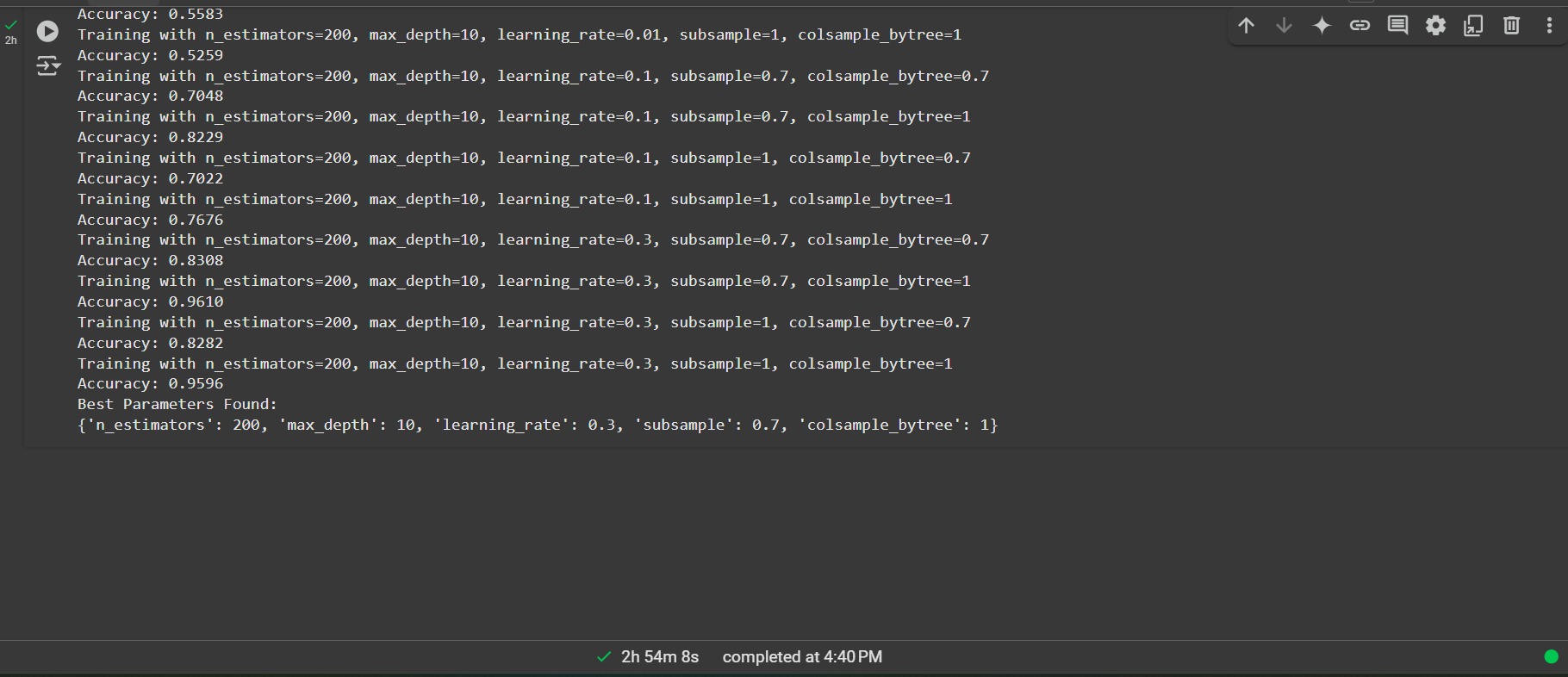
**5. Expected Output of the Model**

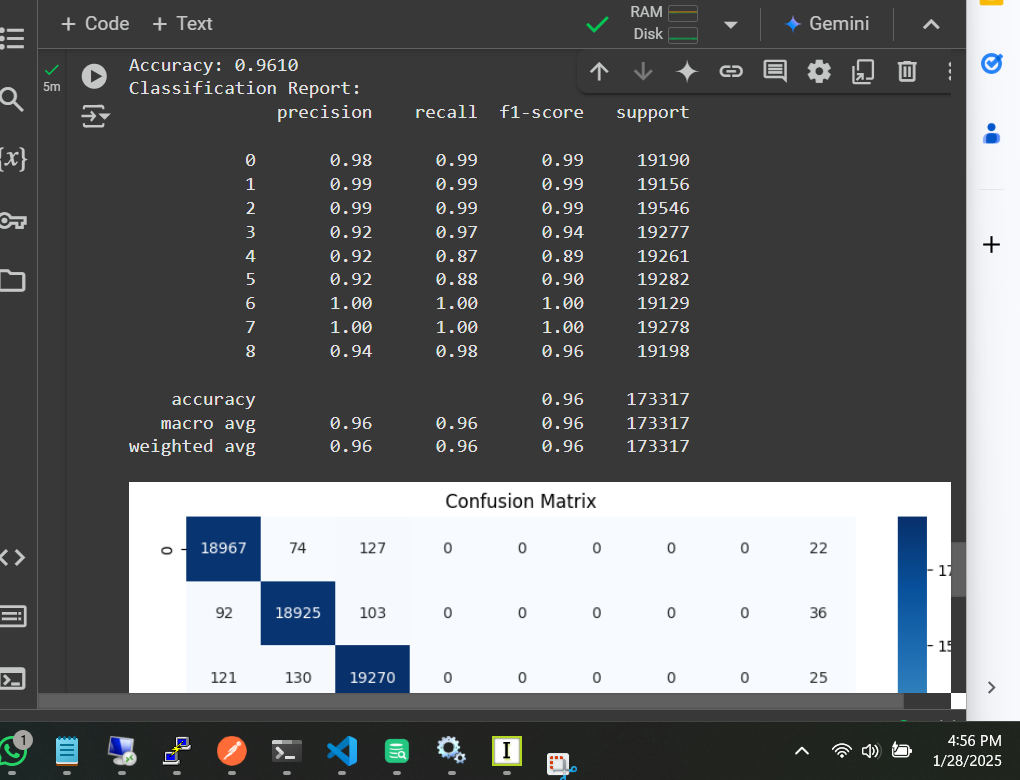
The model predicts **which haircut suits the user best** based on their features.

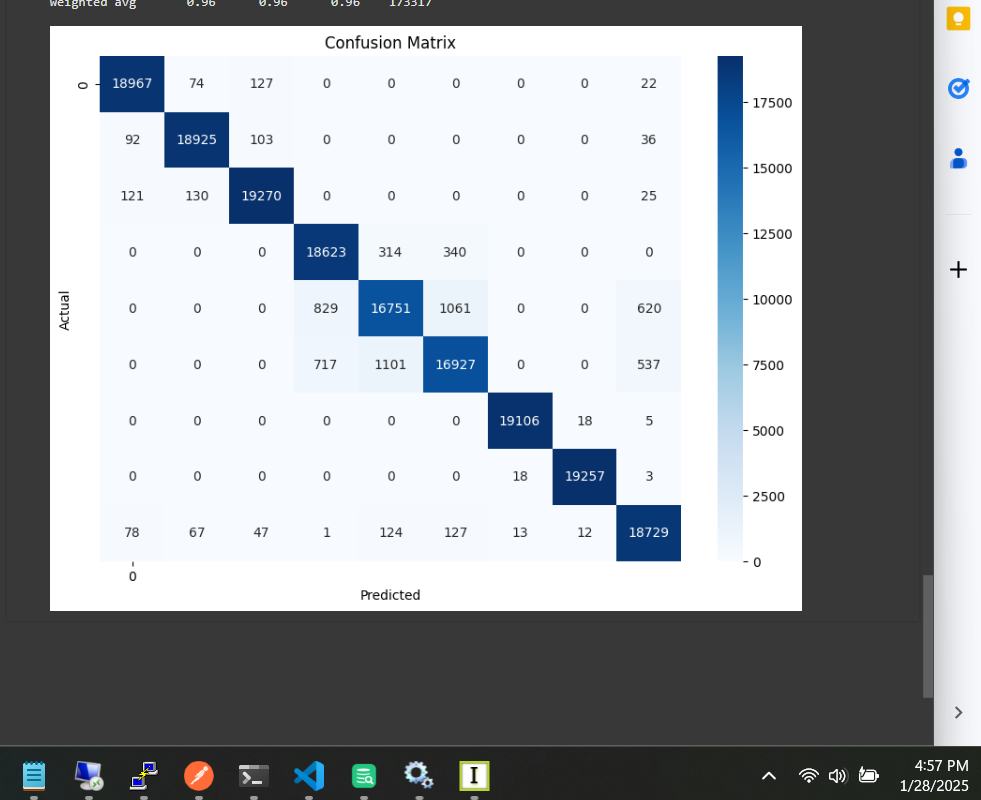
Output includes:

* **Haircut Name** (e.g., "Layered Cut", "Undercut")
* **Haircut Description** (e.g., "A stylish, modern cut for a creative look.")

Model **accuracy** is expected to be above **90%** after tuning.







**6. How Features Work with Labels**

Each feature influences the model’s predictions:

| **Feature** | **Impact on Haircut Recommendation** |
| --- | --- |
| Gender | Different haircuts are suitable for different genders. |
| Occupation | Creative jobs may favor trendy cuts, while formal jobs favor professional styles. |
| Hair Type | Curly hair works well with some styles but not others. |
| Face Shape | Determines which haircut complements facial features. |
| Multi-Tonal | Highlights may affect the choice of style. |
| Eyebrow Shape | Influences haircut harmony with face shape. |
| Lip Shape | Affects the perceived balance of a person’s face. |
| Look Type | Determines if the user prefers a natural, bold, or classic style. |
| Occasion | Helps recommend a casual, formal, or edgy style. |

**7. Model Configuration**

* **Training Dataset:** 80%
* **Testing Dataset:** 20%
* **Oversampling:** SMOTE applied
* **Algorithm Used:** XGBoost
* **Performance Metrics:**
  + Accuracy
  + Confusion Matrix
  + Classification Report

**8. Saving and Loading the Model**

The trained model and encoders are **saved using joblib**, so they can be reused without retraining.

**Files Saved:**

* haircut\_recommendation\_model.pkl → Stores the trained model.
* label\_encoders.pkl → Stores label encoders for categorical features.
* scaler.pkl → Stores the scaler for numerical features (if used).

This allows the **model to be deployed in real-world applications** like web apps or mobile apps.

**9. Making Predictions with New Data**

* New user attributes are **encoded** using label\_encoders.pkl.
* The model predicts the **best haircut** based on input features.
* The haircut name and description are retrieved from a **predefined dictionary**.

Example Prediction Output:

Prediction 1: Layered Cut - A stylish, modern cut for a creative look.

Prediction 2: Undercut - Casual yet stylish with a sharp contrast.

Prediction 3: Crew Cut - Simple, no-nonsense hairstyle.

**10. Summary**

* The model **recommends haircuts** based on various **user features**.
* It uses **XGBoost**, a **powerful machine learning algorithm**.
* **SMOTE** balances the dataset to improve accuracy.
* **Hyperparameter tuning** ensures **optimal performance**.
* The model is **saved and reusable** for real-world applications.
* Predictions are made using a **pretrained model** and **encoded user input**.

**11. Future Improvements**

* **Enhance model accuracy** with additional features (e.g., age, cultural background).
* **Expand dataset** with more diverse haircut images.
* **Integrate deep learning** for better feature extraction.
* **Develop a mobile app** for easy user access.

**12. Questions**

💡 **Q1: Why do we encode categorical data?**  
📝 Because machine learning models work with numerical data, not text.

💡 **Q2: Why do we use XGBoost instead of other models?**  
📝 XGBoost is fast, accurate, and prevents overfitting better than traditional models.

💡 **Q3: Why do we split data into training and testing sets?**  
📝 To evaluate how well the model performs on unseen data.

💡 **Q4: Why do we save the model?**  
📝 To reuse it later without retraining, saving time and resources.

💡 **Q5: What happens if we don’t balance the dataset?**  
📝 The model might favor common haircuts and ignore rare ones, reducing fairness.

💡 **Q6: Can this model work for all hair types?**  
📝 Yes, as long as the dataset has diverse examples.

💡 **Q7: Can this model be improved?**  
📝 Yes, by adding more features, better tuning, and using deep learning.