**Hair Color Recommendation Model Documentation**

**Introduction**

The Hair Color Recommendation Model is designed to suggest the best hair color based on a user’s features, including eye color, gender, skin color, hair type, and face shape. This model uses **XGBoost** (Extreme Gradient Boosting), a powerful machine learning algorithm, to make personalized hair color suggestions with high accuracy.

This document outlines the libraries used, methods applied, expected outcomes, feature-label relationships, model configuration, and other key components for understanding the model.

**1. Libraries Used and Their Purpose**

| **Library** | **Purpose** |
| --- | --- |
| pandas | For handling dataset loading, manipulation, and preprocessing. |
| sklearn.preprocessing.LabelEncoder | To encode categorical features into numerical values for modeling. |
| xgboost | To build the hair color recommendation model using gradient boosting. |
| joblib | For saving and loading the trained model and label encoders for future use. |
| matplotlib.pyplot | For visualizing data insights, such as feature importance and confusion matrices. |
| sklearn.model\_selection.StratifiedKFold | For cross-validation and splitting the dataset into training and testing sets. |
| sklearn.preprocessing.StandardScaler | For scaling numerical features to ensure model performance is not affected by magnitude differences. |
| sklearn.metrics | To evaluate the model’s performance using accuracy score, classification reports, and confusion matrix. |

**2. Why Use XGBoost?**

**XGBoost** is used for the following reasons:

* It is highly efficient, fast, and optimized for large datasets.
* It prevents overfitting through regularization techniques.
* It can model complex, non-linear relationships between features.
* It supports parallel processing, making the training process faster.
* It performs well in classification tasks, such as predicting hair color based on user features.

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

**SMOTE** (Synthetic Minority Over-sampling Technique) is applied to balance the dataset by:

* Generating synthetic samples for underrepresented hair color types.
* Ensuring the model does not bias its predictions toward more frequent classes.
* Improving model performance and generalization, especially for less common hair colors.

**4. Methods Used in the Model**

**4.1 Data Preprocessing**

* **Missing Value Imputation**: If applicable, missing values in features will be handled.
* **Label Encoding**: Converts categorical features (e.g., eye color, skin tone, gender) into numerical format using LabelEncoder().
* **Feature Scaling**: Normalizes numerical data to ensure that no single feature skews the results of the model due to differing magnitudes.

**4.2 Hyperparameter Tuning**

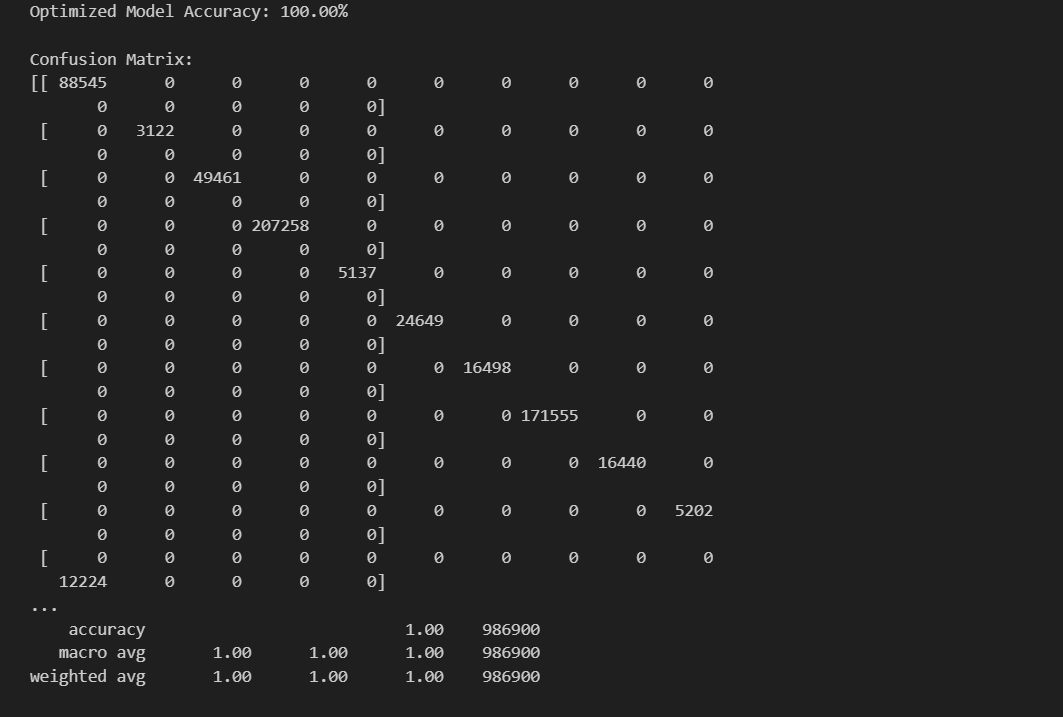
Manual fine-tuning of hyperparameters ensures optimal model performance:

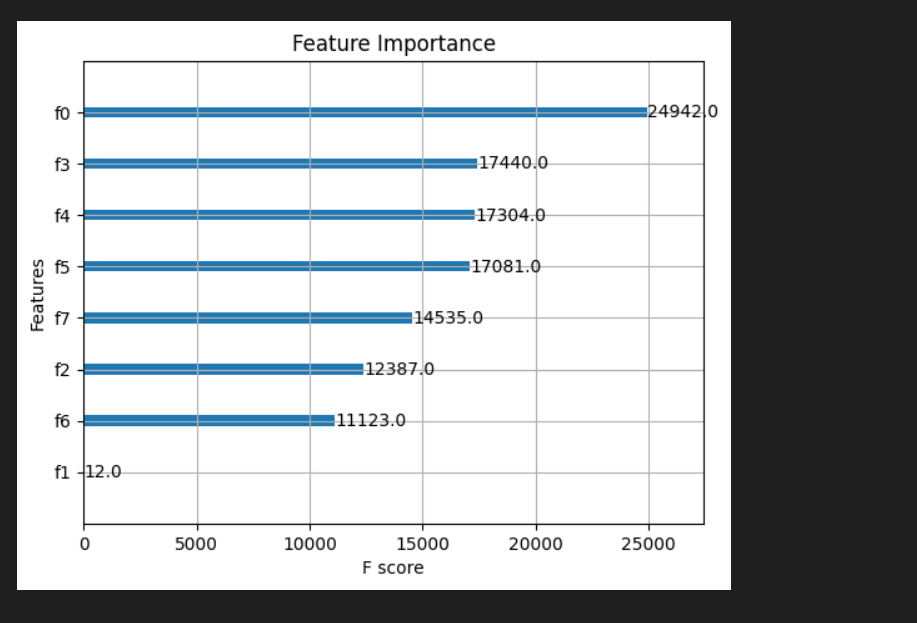
| **Hyperparameter** | **Purpose** |
| --- | --- |
| n\_estimators | Defines the number of trees in the model. |
| max\_depth | Controls the depth of each tree (higher values may cause overfitting). |
| learning\_rate | Determines the step size during optimization (lower values improve generalization). |
| subsample | Fraction of data used for each boosting round, helps prevent overfitting. |
| colsample\_bytree | Fraction of features used per tree to enhance diversity and reduce overfitting. |

**5. Expected Output of the Model**

The model will recommend the best hair color based on the user's input features:

* **Hair Color Recommendation** (e.g., "Soft Brown", "Golden Blonde", "Ash Black")
* **Hair Color Description** (e.g., "A rich brown shade that complements medium to fair skin tones.")
* **Accuracy**: The model is expected to achieve a high accuracy rate (above 90%) after proper tuning.





**6. How Features Work with Labels**

Each feature influences the hair color recommendations:

| **Feature** | **Impact on Hair Color Recommendation** |
| --- | --- |
| **Eye Color** | Determines complementary hair color tones that enhance the natural eye color. |
| **Skin Color** | Influences hair color recommendations that best suit different skin tones (fair, medium, dark). |
| **Hair Type** | Suggests hair colors based on texture, e.g., vibrant colors for straight hair, darker shades for curly hair. |
| **Face Shape** | Impacts the recommended hair color as it works with hairstyles that suit the shape of the face. |
| **Gender** | Certain colors may be traditionally associated with gender, influencing the final recommendation. |
| **Occasion** | For example, a "party" occasion might influence more vibrant hair colors, while a "professional" occasion may recommend more natural tones. |
| **Hair Color Recommendation** | Final output based on the combination of the above features. |

**7. Model Configuration**

* **Training Dataset**: 80% of the dataset for training.
* **Testing Dataset**: 20% of the dataset for testing.
* **Oversampling**: **SMOTE** is applied to balance the classes.
* **Algorithm**: **XGBoost**
* **Performance Metrics**:
  + Accuracy
  + Confusion Matrix
  + Classification Report

**8. Saving and Loading the Model**

The trained model and label encoders are saved using **joblib**, allowing the model to be deployed without retraining:

* **hair\_color\_model\_xgb.pkl** → Stores the trained model.
* **label\_encoders.pkl** → Stores label encoders for categorical features.
* **scaler.pkl** → Stores the scaler for numerical features (if applicable).

These files can be easily integrated into applications for real-time predictions.

**9. Making Predictions with New Data**

* New user attributes are encoded using the saved **label\_encoders.pkl**.
* The model predicts the best hair color based on these encoded features.
* The hair color is then decoded and presented as the recommendation.

**Example Output**:

* **Prediction 1**: Soft Brown - A rich brown shade ideal for medium to fair skin tones.
* **Prediction 2**: Golden Blonde - A light and warm shade perfect for individuals with fair skin and blue eyes.
* **Prediction 3**: Ash Black - A dark and cool-toned color suitable for individuals with dark skin tones.

**10. Summary**

* The model recommends hair colors based on various user features (eye color, skin tone, occupation, etc.).
* **XGBoost** provides fast and accurate recommendations.
* **SMOTE** helps balance the dataset for better generalization.
* Hyperparameter tuning ensures the model performs optimally.
* The model is saved and can be reused for future predictions in real-world applications.

**11. Future Improvements**

* Enhance model accuracy by incorporating additional features (e.g., age, hairstyle preferences).
* Expand the dataset to include more diverse hair colors and attributes.
* Explore deep learning approaches for better feature extraction and prediction.
* Develop a mobile app for easy user access and personalized recommendations.

**12. Questions**

💡 **Q1: Why do we encode categorical data?**  
📝 Machine learning models require numerical data for processing, so categorical data is converted using label encoding.

💡 **Q2: Why do we use XGBoost instead of other models?**  
📝 XGBoost is known for its speed, accuracy, and ability to prevent overfitting, making it ideal for classification tasks like hair color recommendations.

💡 **Q3: Why is the data split into training and testing sets?**  
📝 To assess how well the model generalizes to new, unseen data and ensure it’s not overfitting.

💡 **Q4: Why do we save the model?**  
📝 To reuse the trained model for future predictions without needing to retrain it.

💡 **Q5: What happens if we don’t balance the dataset?**  
📝 The model may bias its predictions toward more frequent hair color types, resulting in lower fairness and generalization.

💡 **Q6: Can this model work for all skin tones and hair colors?**  
📝 Yes, as long as the dataset is diverse and inclusive of various skin tones, colors, and features.

💡 **Q7: Can this model be improved?**  
📝 Yes, by adding more features, tuning the hyperparameters further, and considering deep learning for better feature extraction.