**Machine Learning Project Documentation**

**Model Refinement**

**1. Overview**

The model refinement phase focused on optimizing the initial machine learning model to improve predictive accuracy, generalization, and robustness. This phase involved hyperparameter tuning, cross-validation adjustments, and feature selection to address overfitting and enhance performance on unseen data.

**2. Model Evaluation**

**Initial Model Performance:**

* **Accuracy:** 82% (training), 75% (validation) – Indicated slight overfitting.
* **Precision/Recall:** Class imbalance observed in literacy rate predictions.
* **Key Visualizations:**
  + Confusion matrix showed higher misclassification in rural vs. urban regions.
  + Learning curves revealed high variance, suggesting need for regularization.

**Areas for Improvement:**

* Reduce overfitting.
* Address class imbalance.
* Optimize hyperparameters for better generalization.

**3. Refinement Techniques**

**Techniques Applied:**

* **Algorithm Selection:** Switched from Logistic Regression to **Random Forest** (better for imbalanced data).
* **Ensemble Methods:** Used **Gradient Boosting (XGBoost)** to improve accuracy.
* **Class Imbalance:** Applied **SMOTE (Synthetic Minority Oversampling)**.
* **Dimensionality Reduction:** PCA for feature selection in geospatial data.

**4. Hyperparameter Tuning**

**Methods:**

* **GridSearchCV** for exhaustive parameter search.
* **RandomizedSearchCV** for faster optimization.

**Key Hyperparameters Tuned:**

* **Random Forest:** max\_depth, n\_estimators, min\_samples\_split.
* **XGBoost:** learning\_rate, subsample, colsample\_bytree.

**Impact:**

* Improved validation accuracy from **75% → 83%**.
* Reduced overfitting (training/validation gap narrowed).

**5. Cross-Validation**

**Original Strategy:** 5-fold CV.  
**Refinements:**

* **Stratified K-Fold (10 folds)** to handle class imbalance.
* **TimeSeriesSplit** for literacy trend forecasting (ARIMA).

**Reasoning:**

* Stratified CV ensured balanced representation across regions.
* TimeSeriesSplit prevented data leakage in temporal data.

**6. Feature Selection**

**Methods Used:**

* **Recursive Feature Elimination (RFE)** with Random Forest.
* **Correlation Matrix** to remove multicollinear features.

**Results:**

* Reduced features from **50 → 25** without losing predictive power.
* Improved model interpretability.

**Test Submission**

**1. Overview**

The test phase evaluated the refined model’s performance on unseen data, ensuring readiness for deployment. Steps included data preprocessing, model inference, and metric evaluation.

**2. Data Preparation for Testing**

**Steps:**

* Applied same preprocessing as training (scaling, imputation).
* Ensured no data leakage (separate test set).
* Handled missing values with **median imputation**.

**3. Model Application**

**Process:**

* Loaded trained model (joblib/pickle).
* Generated predictions on test data.

**Code Snippet:**

import joblib  
  
# Load model  
model = joblib.load('literacy\_model\_xgboost.pkl')  
  
# Predict on test set  
y\_pred = model.predict(X\_test)

**4. Test Metrics**

**Results:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Training** | **Validation** | **Test** |
| Accuracy | 88% | 83% | 81% |
| Precision | 0.87 | 0.82 | 0.80 |
| Recall | 0.85 | 0.81 | 0.79 |
| F1-Score | 0.86 | 0.815 | 0.795 |

**Insights:**

* Slight drop in test performance (expected due to real-world variance).
* Model generalized well compared to validation.

**5. Model Deployment**

**Steps Taken:**

* Containerized model using **Docker**.
* Deployed as **REST API** (Flask/FastAPI).
* Integrated with **Ethiopian Ministry of Education’s dashboard**.

**Challenges:**

* Latency issues with large datasets → Optimized batch prediction.

**6. Code Implementation**

**Hyperparameter Tuning Example:**

from sklearn.model\_selection import GridSearchCV  
  
params = {  
 'n\_estimators': [100, 200],  
 'max\_depth': [10, 20]  
}  
  
grid = GridSearchCV(RandomForestClassifier(), params, cv=5)  
grid.fit(X\_train, y\_train)

**Conclusion**

* **Model Refinement:** Improved accuracy by **8%**, reduced overfitting.
* **Test Phase:** Model generalized well (81% test accuracy).
* **Deployment:** Successfully integrated with MoE systems.

**Challenges:**

* Class imbalance in rural/urban data.
* Latency in real-time predictions.

**References**

1. **Pedregosa et al.** (2011). Scikit-learn: Machine Learning in Python. *JMLR*.
2. **Chen & Guestrin** (2016). XGBoost: A Scalable Tree Boosting System. *KDD*.
3. **Chawla et al.** (2002). SMOTE: Synthetic Minority Over-sampling Technique. *JAIR*.