COPD Risk Prediction Using Machine Learning

Vishnu Sai Pasanth
M. Praneeth

December 3, 2025

1 Introduction

This project aims to build and compare machine learning models to predict the risk of Chronic Obstructive Pulmonary Disease (COPD) using a dataset of 44,553 patient records containing demographic, biochemical, and clinical variables. The F1-score was selected as the primary evaluation metric due to class imbalance. Models were trained, tuned, and evaluated; final predictions were submitted to Kaggle.

2 Exploratory Data Analysis and Preprocessing

2.1 Initial inspection

The dataset required minimal cleaning:

- No missing values or duplicate patient IDs.
- Three categorical variables: `sex`, `oral_health_status`, and `tartar_presence`.
- `oral_health_status` contained a single value ("Y") and was removed.
- `patient_id` was removed because it carries no predictive signal.

Categorical variables were encoded as numeric (e.g., `M=1, F=0`; `Y=1, N=0`). After encoding, all features were numeric.

2.2 Distribution and skewness

Histograms and skewness analysis showed:

- Strong right-skewness for biochemical and enzyme features (e.g., triglycerides, GGT, AST, ALT).
- Vision and hearing metrics displayed discrete, highly skewed distributions.

- Anthropometrics (height, weight, waist) and blood pressure were approximately symmetric.

Skewness values above 5 were observed for some enzyme markers — expected in clinical datasets. No transformations (log/Box–Cox) were applied because models used (SVM, NN, XGBoost) handle non-normal distributions robustly, and the skewed values were clinically meaningful.

2.3 Outlier capping experiment

An IQR-based capping (winsorization) class was implemented for experimentation:

```
class OutlierCapper:
    def _init_(self, series):
        q1 = series.quantile(0.25)
        q3 = series.quantile(0.75)
        iqr = q3 - q1
        self.lower = q1 - 1.5 * iqr
        self.upper = q3 + 1.5 * iqr

    def cap(self, value):
        if value < self.lower:
            return self.lower
        elif value > self.upper:
            return self.upper
        else:
            return value
```

This replaces values outside the Tukey whiskers with the whisker bounds rather than deleting rows. The capped dataset was used for EDA experiments (before/after plots), but not for the final modeling pipeline because:

- Extreme biochemical values were clinically plausible and not measurement errors.
- Capping alters genuine clinical variability and could bias models.
- Downstream models used are robust to skew and outliers.

2.4 Scatterplots, pairplots and separability

Scatterplots and pairplots for selected feature groups showed heavy overlap between COPD-positive and COPD-negative classes. No single feature provided linear separability, motivating nonlinear models (RBF SVM, Neural Networks, XGBoost).

2.5 Correlation analysis

A correlation heatmap revealed physiologically coherent clusters (blood pressure, lipid markers, anthropometrics) and no harmful multicollinearity; individual feature correlation with the label was modest.

2.6 Feature scaling and split

All numeric features were standardized using `StandardScaler` fitted on training data only to avoid leakage. An 80–20 stratified train–validation split was used for all experiments unless otherwise specified (e.g., the 20% dataset experiments described later).

3 Models Implemented

Models trained and compared:

- Logistic Regression
- Linear SVM
- RBF Kernel SVM
- Neural Network (scikit-learn MLPClassifier)
- TensorFlow Neural Network (lightweight)
- XGBoost (gradient boosting trees)

F1-score was the main evaluation metric.

4 Results Before Hyperparameter Tuning

4.1 Logistic Regression (baseline)

- Train F1: ≈ 0.71
- Validation F1: ≈ 0.66

4.2 Linear SVM (baseline)

- Train F1: ≈ 0.72
- Validation F1: ≈ 0.67

4.3 Neural Network (baseline)

- Train F1: ≈ 0.71
- Validation F1: ≈ 0.69

4.4 RBF SVM (manual baseline)

A basic RBF SVM trained with:

$$C = 2.0, \quad \gamma = 0.01$$

gave:

- Train F1: 0.706
- Validation F1: 0.665

This baseline showed the need for hyperparameter tuning.

5 Hyperparameter Tuning Results

This section lists the tuning results for the major models.

5.1 Logistic Regression (tuned)

$$C = 5$$

- Validation F1: 0.667

5.2 Linear SVM (tuned)

$$C = 0.01$$

- Validation F1: 0.674

5.3 Tuned RBF SVM (full dataset)

A randomized search over (C, γ) returned:

```
Best Params (RBF SVM): {'gamma': np.float64(0.01),
                        'C': np.float64(4.6415888336127775)}
```

Performance reported for this tuned RBF SVM (full dataset):

- RBF SVM Train F1: 0.70622
- RBF SVM Valid F1: 0.68453

Note: these are the empirical model metrics obtained during tuning and cross-validation for the full dataset.

5.4 Neural Network (tuned)

Using randomized search on the MLPClassifier, the best configuration obtained for the primary (full) experiments was:

$$\text{hidden\_layer\_sizes} = (128, 64, 32), \quad \text{activation} = \tanh, \quad \alpha = 0.001, \quad \text{learning\_rate\_init} = 0.001, \quad \text{max\_ite}$$

- Train F1: 0.7344
- Validation F1: 0.6935
- Kaggle Public F1: 0.724

5.5 XGBoost (tuned)

Grid search (108 candidates, 3-fold CV) returned:

$$\text{n\_estimators} = 200, \quad \text{learning\_rate} = 0.1, \quad \text{max\_depth} = 9, \quad \gamma = 0.1, \quad \text{colsample\_bytree} = 1.0.$$

Performance:

- Train F1: 0.9557
- Validation F1: 0.7372

XGBoost provided the highest validation F1 in cross-validation.

5.6 TensorFlow Neural Network

A lightweight TensorFlow feedforward model (two hidden layers) was trained for comparison. Due to limited tuning and compute time, this model achieved:

- Validation F1: 0.676
- Kaggle Public Score: 0.687

6 20% Dataset Experiments (Low-data regime)

To study robustness in a low-data regime, a stratified 20% subset of the dataset was created, then split 80–20 for training/validation.

6.1 Tuned RBF SVM on 20% dataset

Fast tuning on the 20% subset produced parameters close to:

$$C \approx 4.64, \quad \gamma \approx 0.01$$

and yielded:

- Train F1: 0.7163
- Validation F1: 0.6804

6.2 Tuned Neural Network on 20% dataset

Best configuration found for the small dataset:

$$\text{hidden\_layers} = (128, 64), \quad \alpha = 0.01, \quad \text{max\_iter} = 300$$

Observed performance:

- Train F1: 0.6952
- Validation F1: 0.667

6.3 Discussion: SVM vs NN in low-data regime

The RBF SVM outperformed the Neural Network on the 20% subset because:

- SVMs (with RBF kernel) optimize a convex problem and are often more data-efficient.
- Neural Networks require larger datasets to reliably learn many parameters and avoid unstable training.
- With reduced data, neural networks can underfit or produce high variance, while SVMs generalize better from fewer examples.

7 Final Comparison of Tuned Models

The table below shows the final validation F1-scores for each tuned model (full-dataset tuning unless otherwise noted).

Model	**Final Validation F1**
Logistic Regression (tuned)	0.667
Linear SVM (tuned)	0.674
RBF SVM (tuned, full dataset)	0.68453
Neural Network (tuned, full dataset)	0.6935
TensorFlow Neural Network	0.676
XGBoost (tuned)	**0.7372**

Table 1: Final comparison of tuned models (validation F1).

8 Conclusions

- Multiple classifiers were implemented and compared across a full dataset and a low-data (20%) regime.
- The tuned Neural Network achieved the best Kaggle public score (0.724) and competitive validation performance (0.681) on the full dataset.

- XGBoost obtained the highest validation F1 (0.7372) during cross-validation.
- Tuned RBF SVM provided strong results, especially in the 20% low-data regime where it outperformed the Neural Network due to better data efficiency.
- Outlier capping and other preprocessing steps were implemented and evaluated but were not included in the final modeling set because extreme values were clinically plausible.