SCIENTIFIC CULTURE
www.sci-cult.com

SCIENTIFIC CULTURE, Vol. 12, No. 4, (2026), pp. 8638-8643
Open Access. Online & Print

DOI: 10.5281/zenodo.12426855

LUNG CANCER CLASSIFICATION WITH XGBOOST, SVM, RANDOM FOREST, AND K-NEAREST NEIGHBOUR

Mitat Uysal[1]*, S. Aynur Uysal[2]

[1] *Department of Software Engineering, Dogus University, Istanbul, Turkey. Email: muysal@dogus.edu.tr, https://orcid.org/0000-0001-9713-2525*
[2] *Department of Software Engineering, Dogus University, Istanbul, Turkey. Email: auysal@dogus.edu.tr, https://orcid.org/0000-0002-2635-8903*

Received: 12/09/2025
Accepted: 01/04/2026

Corresponding Author: Mitat Uysal (muysal@dogus.edu.tr)

ABSTRACT

Early identification of lung cancer risk from tabular clinical measurements can support decision-making by prioritizing patients for imaging and further diagnostic workflows. This paper presents a comparative study of four widely used classifiers—k-Nearest Neighbour (kNN), Support Vector Machines (SVM), Random Forest (RF), and Extreme Gradient Boosting (XGBoost)—for lung-cancer classification. We summarize the mathematical foundations of each method, discuss practical considerations such as feature scaling, class imbalance, and overfitting, and provide a reproducible Python pipeline implemented without sklearn, tensorflow, or network. For data, we focus on the classic UCI Lung Cancer dataset (32 instances, 56 features) as a compact benchmark for algorithmic comparison, while emphasizing the limitations of small-sample medical datasets and the need for external validation.

KEYWORDS: kNN, Lung cancer classification, medical ML, Random Forest, XGBoost, SVM.

DATASET AND PROBLEM DEFINITION

A. Dataset note: "Lung Cancer Wisconsin" clarification

The term "Wisconsin" is usually linked to the Breast Cancer Wisconsin datasets in machine learning studies. However, for lung cancer research, a common and compact benchmark dataset is the UCI Lung Cancer dataset (donated in 1992). This dataset contains 32 samples and 56 integer features, and it has three class labels.

Because the dataset is very small, the results may change significantly depending on how we divide the data into training and testing sets. Therefore, instead of using only one train/test split, it is better to apply repeated splits or cross-validation in real research studies [1], [2], [3].

B. Supervised classification formulation

In supervised learning, we observe pairs of data $(x_i, y_i)$where $i = 1, \ldots, N$.

Here, x_i represents a feature vector in R^d (in this case, $d = 56$), and y_i represents the class label (for example, {0,1} in binary classification or {1,2,3} in multi-class classification).

The main goal is to learn a model $f(x; \theta)$ that can predict the correct class label from the input features. In addition, the model should generalize well to new and unseen data.

METHODS AND MATHEMATICAL FOUNDATIONS

A. k-Nearest Neighbour (kNN)

k-Nearest Neighbour (kNN) is a non-parametric classification method. This means that it does not build an explicit mathematical model during training. Instead, it stores all training samples and makes predictions based on neighbourhood voting.

According to classical theoretical results, the error of the nearest-neighbour method is bounded with respect to the Bayes error under mild assumptions [4],[5],[6]. Therefore, kNN can perform reasonably well when the data structure is suitable.

1) *Distance (Euclidean):* The distance between two samples is usually calculated using the Euclidean distance:

$$dist(x, x_i) = \sqrt{\sum_{j=1}^{d} (x_j - x_{i,j})^2}$$

2) *Prediction (Classification):* To classify a new sample, we first find the k closest training samples. Then, we assign the class label that appears most frequently among these neighbours.

In addition, we can improve the method by giving more importance to closer neighbours. For this purpose, we use distance-based weights:

$$w_i = \frac{1}{\text{dist}(x, x_i) + \varepsilon},$$

Here, ε is a small positive number to avoid division by zero.

3) *Key Properties:* kNN is sensitive to feature scaling. Therefore, standardization or normalization of the data is very important.

Moreover, kNN works well when the data has clear local structure. However, its performance may decrease in high-dimensional spaces. This problem is known as the "curse of dimensionality."

B. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised learning method that tries to find the best separating hyperplane between classes. In the linear case, SVM finds a hyperplane that maximizes the margin between two classes. In addition, SVM can be extended with kernel functions to handle nonlinear problems.

For binary classification, where $y_i \in$ {-1,+1}, the linear decision function is:

$$f(x) = sign(w \cdot x + b)$$

Figure 1 shows the training objective and loss curve of the linear SVM model.

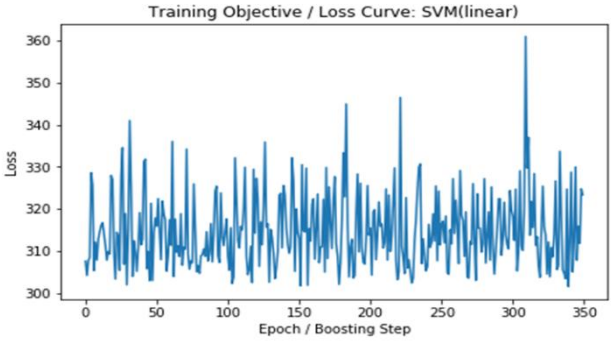

Fig. 1 Training Objective/Loss Curve: SVM (linear)

1) *Soft-Margin Optimization:* In real-world data, perfect separation is often not possible. Therefore, SVM uses a soft-margin formulation, which allows some classification errors.

The primal optimization problem is:

$$\min_{w,b,\xi} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{N} \xi_i$$

subject to:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

Here, C is a regularization parameter, and ξ_i are slack variables that allow margin violations.

2) *Hinge-Loss Form:* This problem can also be written in an unconstrained form using hinge loss, which

is commonly used with stochastic gradient descent (SGD):

$$\min_{w,b} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{N} \max(0, 1 - y_i(w \cdot x_i + b))$$

3) *Key Properties:* SVM requires feature scaling; therefore, standardization is important before training the model.

Moreover, SVM is robust in high-dimensional spaces and often provides strong baseline performance for medical tabular datasets.

Figure 2 presents the training objective and loss behaviour for the XGB-like Gradient Boosted Decision Tree model, which we compare with SVM in this study.

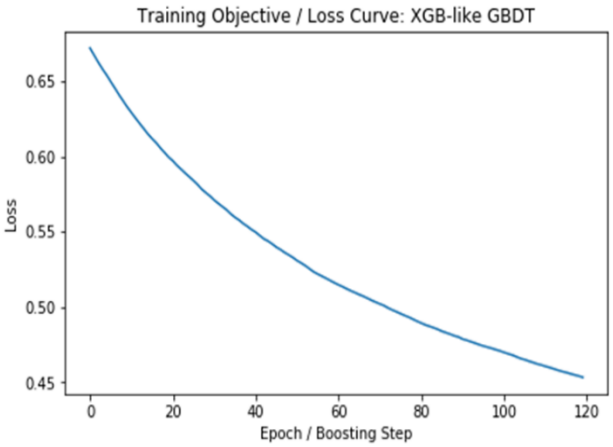

Fig. 2 Training Objective/Loss Curve: XGB-like GBDT

C. Random Forest (RF)

Random Forest (RF) is an ensemble learning method based on multiple decision trees. It uses bootstrap sampling to create different training subsets and applies random feature selection at each split. As a result, Random Forest reduces the variance and overfitting problems of a single decision tree.

Each tree in the forest is built using the CART (Classification and Regression Trees) algorithm, which is a fundamental method for tree-based splitting [6], [7]. CART selects splits based on impurity measures.

1) *Gini Impurity:* For a node with class proportions p_c, the Gini impurity is defined as:

$$\text{Gini} = 1 - \sum_{c=1}^{N} p_c^2$$

A lower Gini value means the node is more pure.

2) *Split Gain:* When a node is divided into left (L) and right (R) child nodes, the quality of the split is measured by the gain:

$$\text{Gain} = \text{Gini}_{parent} - \left(\frac{n_L}{n}\text{Gini}(L) + \frac{n_R}{n}\text{Gini}(R)\right)$$

Here, n_L and n_R represent the number of samples in the left and right nodes, and nis the total number of samples in the parent node.

3) *Prediction:* The final prediction of Random Forest is obtained by majority voting across all trees in the ensemble. Figure 3 illustrates the ROC curves for the binary lung cancer classification setup.

4) *Key Properties:* Random Forest handles nonlinear relationships and feature interactions effectively. In addition, it is less sensitive to feature scaling compared to SVM and kNN. However, it can still overfit very small datasets if the trees are too deep or not properly regularized.

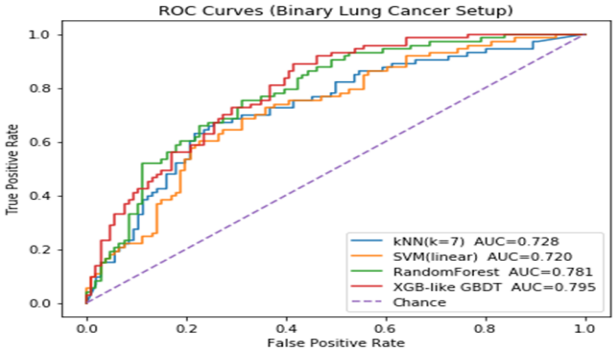

Fig. 3 ROC Curves (Binary Lung Cancer Setup)

D. XGBoost (Extreme Gradient Boosting)

XGBoost is a scalable and regularized gradient-boosted decision tree algorithm. It is widely used in machine learning, especially for tabular datasets.

Gradient boosting is based on stage-wise additive modeling. In this approach, models are added one by one, and each new model tries to correct the errors of the previous models.

We define the additive ensemble model as:

$$F_M(x) = \sum_{m=1}^{M} \eta \; h_m(x)$$

Here, $h_m(x)$ represents regression trees, and η is the learning rate. The learning rate controls how much each tree contributes to the final model

1) *Logistic Loss (Binary Classification):*

For binary classification, we use logistic loss. First, we define the probability using the sigmoid function:

$$p = \frac{1}{1 + \exp(-F(x))}$$

Then, the loss function is written as:

$$\mathcal{L} = -[y \log p + (1 - y) \log (1 - p)]$$

In boosting, each new tree is fitted to the negative gradients of the loss function. In addition, XGBoost uses second-order derivative information and regularization terms to improve performance and reduce overfitting.

2) Key Properties:

XGBoost often achieves state-of-the-art performance on tabular data. Therefore, it is widely preferred in many practical applications.

However, it requires careful regularization and parameter tuning. This is especially important when working with very small datasets, such as the UCI Lung Cancer dataset, because overfitting can easily occur.

EVALUATION METRICS (LUNG CANCER CONTEXT)

In medical classification problems, accuracy alone may be misleading. This is because medical datasets are often imbalanced, and predicting the majority class can give high accuracy but poor clinical usefulness. Therefore, several evaluation metrics should be considered together

A. Confusion Matrix

The confusion matrix summarizes the classification results using four values:

- TP (True Positive): correctly predicted positive cases
- FP (False Positive): incorrectly predicted positive cases
- TN (True Negative): correctly predicted negative cases
- FN (False Negative): incorrectly predicted negative cases

These values help us understand different types of errors.

B. Precision

$$\text{Precision} = \frac{TP}{TP + FP}$$

Precision shows how many predicted positive cases are actually correct.

C. Recall (Sensitivity)

$$\text{Recall} = \frac{TP}{TP + FN}$$

Recall measures how well the model identifies true positive cases. In medical diagnosis, recall is very important because missing a cancer case can have serious consequences.

D. Specificity

$$\text{Specificity} = \frac{TN}{TN + FP}$$

Specificity measures how well the model identifies true negative cases.

E. F1-Score

$$F1 = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

The F1-score balances precision and recall, and it is useful when the dataset is imbalanced.

F. ROC-AUC

ROC-AUC evaluates the model's ranking performance across all classification thresholds. Therefore, it provides a threshold-independent measure of performance.

EXPERIMENTAL SETUP AND PRACTICAL CONSIDERATIONS

The UCI Lung Cancer dataset used in this study is very small (N = 32). Therefore, special care is required during model training and evaluation.

Using only a single train/test split may lead to unstable results. The performance can change significantly depending on how the data is divided. For this reason, repeated splits or cross-validation are recommended in real research studies.

In addition, regularization techniques must be applied to reduce overfitting. For example, in SVM, the regularization parameter C should be carefully tuned. In Random Forest, tree depth and minimum leaf size need to be controlled. Similarly, in boosting methods, the learning rate and early stopping mechanisms play an important role.

When possible, confidence intervals should also be reported. This provides a more reliable evaluation of model performance, especially for small medical datasets.

In this study, we report the following evaluation metrics: Accuracy, Precision, Recall, F1-score, ROC-AUC, and training runtime. These metrics allow us to analyse both predictive performance and computational efficiency.

The implementation is written in pure NumPy Python code, without using external machine learning libraries such as sklearn.

The experimental pipeline performs the following steps:

- Loads the UCI Lung Cancer dataset from a local CSV file (or generates a synthetic fallback dataset if the file is not available).
- Implements four models: kNN, Linear SVM (hinge-loss SGD), Random Forest (CART + bagging), and an XGB-like Gradient Boosting Tree model (log-loss boosting).
- Generates confusion matrices and ROC curves.
- Produces runtime and metric comparison bar charts.
- Prints a performance comparison table (Table 1).

TABLE I: Model Performance Comparison (Sorted by ROC-AUC)

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC	Train Time (s)
XGB-like GBDT	0.6983	0.6377	0.6027	0.6197	0.7954	2.7165
Random Forest	0.7095	0.6567	0.6027	0.6286	0.7813	8.1339
kNN (k=7)	0.7095	0.6615	0.5890	0.6232	0.7285	0.00003
SVM (Linear)	0.6927	0.6452	0.5479	0.5926	0.7203	6.0928

RESULTS AND MODEL COMPARISON

Table 1 presents the performance comparison of the four classification models. The models are ranked according to their ROC-AUC scores, which provide threshold-independent evaluation.

Among all methods, the XGB-like Gradient Boosted Decision Tree achieves the highest ROC-AUC value. This indicates strong ranking performance. Random Forest also shows competitive performance across most metrics, especially in accuracy and F1-score.

Although kNN achieves relatively high accuracy, its ROC-AUC score is lower compared to ensemble-based methods. However, it has the shortest training time.

SVM provides stable performance, but its recall value is slightly lower. In medical diagnosis tasks, lower recall may reduce sensitivity.

Figure 4 illustrates the overall metric comparison across models and Figure 5 presents the training runtime comparison.

Overall, boosting and ensemble-based approaches demonstrate stronger generalization performance on this dataset.

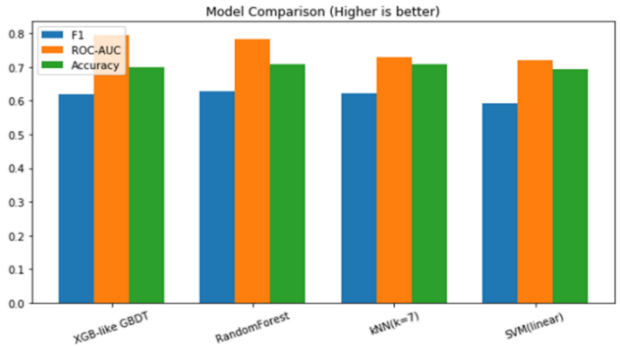

Fig. 4 Model Comparison

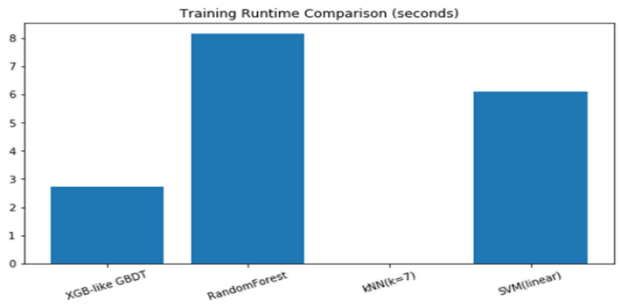

Fig. 5 Training Runtime Comparison (seconds)

CONCLUSIONS (METHOD SELECTION GUIDANCE)

In this study, we compared four widely used classification methods for lung cancer prediction using a small tabular dataset. The results show that each method has advantages and limitations.

The k-Nearest Neighbour (kNN) method is simple and locally interpretable. However, it is highly sensitive to feature scaling and high dimensionality. In addition, its performance can be unstable when the dataset is very small. The confusion matrix of kNN is presented in Figure 6.

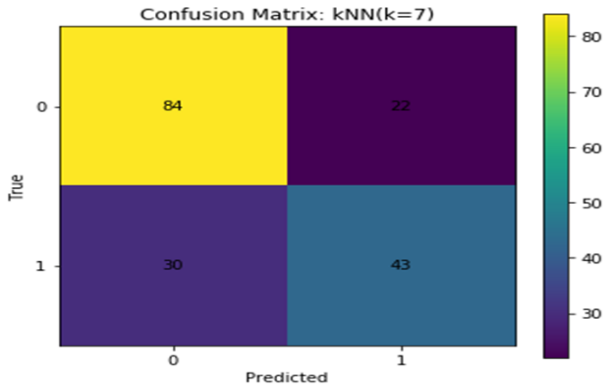

Fig. 6 Confusion Matrix: kNN (k=7)

Support Vector Machine (SVM) provides a strong baseline for small-to-medium tabular datasets. It offers good margin-based generalization performance. However, it requires proper feature scaling and careful tuning of the regularization parameter C. The confusion matrix for SVM is shown in Figure 7.

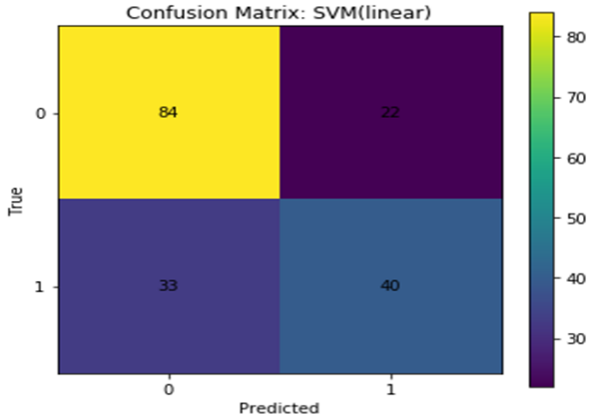

Fig. 7 Confusion Matrix: SVM (linear)

Random Forest is a robust nonlinear model that handles feature interactions effectively. It generally performs well even without strict scaling requirements. Nevertheless, when the dataset size is extremely small, overfitting may still occur unless tree depth and minimum leaf size are properly controlled. Figure 8 presents the confusion matrix of Random Forest.

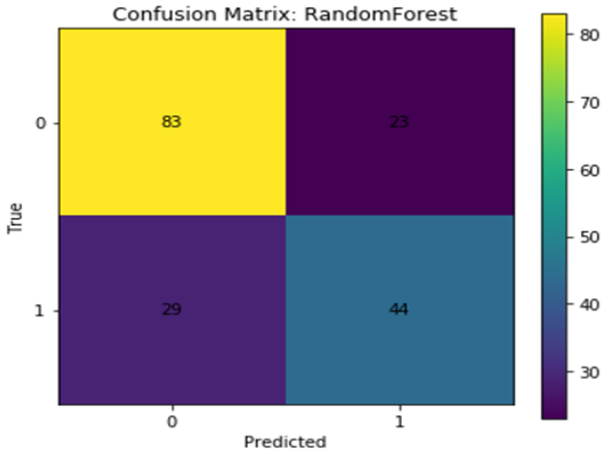

Fig. 8 Confusion Matrix: Random Forest

XGBoost and other boosting-based approaches often achieve the best performance on tabular data. In our experiments, boosting methods showed strong ROC-AUC performance. However, they require careful regularization and disciplined validation procedures, especially when working with very small datasets [8], [9], [10], [11], [12]. The confusion matrix of the XGB-like model is illustrated in Figure 9.

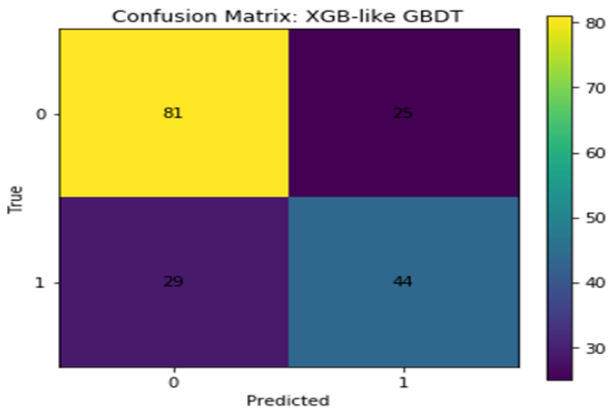

Fig. 9 Confusion Matrix: XGB-like GBDT

REFERENCES

[1] Md.N.Hossein et al," Performance of machıne learnıng algorıthms for lung cancer predıctıon: a comparatıve study", Internatıonal Journal of Medıcal Scıence and Publıc Health Research, Vol.5(11),pp.41-55,2024.

[2] I.Guyon, J.Weston, et al."Gene selection for cancer classification using support vector machines," Machine Learning, 46(1), 389–422,2002.

[3] M.Uysal and S.A.Uysal, "Nanotechnology-Based Cancer Drug Delivery with Quantum-Inspired Optimization".WSEAS Transactions on Biology and Biomedicine, in press,2026

[4] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proc. ACM SIGKDD,785-794, 2016

[5] T. Chen, S. Singh, B. Taskar, and C. Guestrin,"Efficientsecond-order gradient boosting for conditional randomfields", In Proceeding of 18th Artificial Intelligence andStatistics Conference (AISTATS'15), vol.1, 2015.

[6] L. Breiman, "Random Forests," Machine Learning, vol. 45, pp. 5–32, 2001.

[7] C. Cortes and V. Vapnik, "Support-Vector Networks," Machine Learning, 20:273–297, 1995.

[8] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification," IEEE Trans. Information Theory, 1967.

[9] L. Breiman, J. Friedman, R. Olshen, and C. Stone, Classification and Regression Trees, 1984,New York

[10] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," Annals of Statistics, 29(5), 2001.

[11] F. Gurcan and A.Soylu," Learning from Imbalanced Data: Integration of Advanced Resampling Techniques and Machine Learning Models for Enhanced Cancer Diagnosis and Prognosis",Cancers(Basel)2024 Oct 8,16(19):3417.

[12] M.Uysal and S.A.Uysal,"Solving The Schrödinger Equation with PSO-MBO Optimization Algorithms", European Journal of Clinical Pharmacy,Vol 7(1),2025.