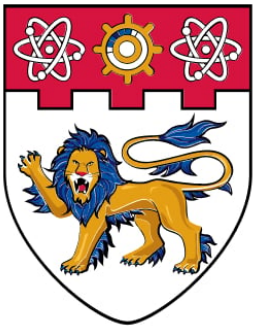


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

PREDICTING LUNG CANCER OUTCOMES: LEVERAGING MACHINE LEARNING TECHNIQUES FOR EARLY DETECTION AND PROGNOSIS

MH4510: Statistical Learning and Data Mining

Author: Ian Oon (N2401547J), Tan Lu Yan (U2240431J), Poon Hiu Ching (N2400987E), Sean Gastinov Soeandi (U2220491L), and Ho Inn Jong Jereme (U1940727B)

Team: Sunset Coral

22 November 2024

Abstract

Lung cancer is a leading cause of cancer-related mortality, emphasizing the need for early detection. This project applies machine learning to predict lung cancer risk using patient survey data, classifying outcomes into binary risk categories while prioritizing the reduction of false negatives. The model provides actionable insights to healthcare professionals, enabling early intervention without formal screening and empowering patients to seek care proactively. We compare several machine learning models: gradient boosting, deep forest, logistic regression, artificial neural networks, and stacking methods to identify the most effective model for deployment.

1.0 Introduction

Lung cancer is a leading cause of cancer-related mortality worldwide, responsible for 1.8 million deaths worldwide in 2020 [1]. Early detection is critical in improving survival rates, as it enables timely interventions and better treatment outcomes [2]. This project aims to develop a machine learning model that predicts lung cancer risk using survey data, helping identify individuals at high risk.

With a model achieving 90% accuracy, it is estimated that early detection could prevent up to 1.2 million deaths annually by enabling earlier intervention. This approach does not require any formal screening, allowing patients to assess their risk through a simple survey before seeing a doctor, empowering proactive care and early detection while providing doctors with initial awareness of potential cases.

Loading Libraries

2.0 Data Preparation

2.1 Dataset Description

This dataset focuses on various health-related factors associated with lung cancer. It comprises of fifteen features that capture lifestyle choices and medical symptoms. It includes attributes such as age, gender, smoking status, as well as indicators of chronic disease such as anxiety, fatigue, and respiratory issues. The dataset was selected for its relevance to lung cancer research and is widely recognized on Kaggle. It is also the most downloaded and utilized for related projects. Its comprehensive nature provides an excellent resource for exploring correlations between lifestyle factors and lung cancer, yielding insights that can inform public health strategies and preventive measures.

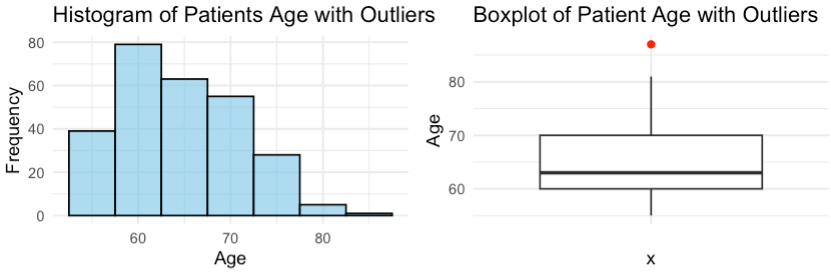
Feature	Description
GENDER	1: Male, 0: Female
AGE	Age of the individual (in years)
SMOKING	1: Smoker, 0: Non-smoker
YELLOW_FINGERS	1: Yes (yellow fingers), 0: No
ANXIETY	1: Yes, 0: No (anxiety symptoms)
PEER_PRESSURE	1: Yes, 0: No (influence from peers)
CHRONIC.DISEASE	1: Yes, 0: No (chronic health issues)
FATIGUE	1: Yes, 0: No (feeling tired)
ALLERGY	1: Yes, 0: No (presence of allergies)
WHEEZING	1: Yes, 0: No (wheezing sound)
ALCOHOL.CONSUMING	1: Yes, 0: No (alcohol consumption)
COUGHING	1: Yes, 0: No (coughing symptoms)
SHORTNESS.OF.BREATH	1: Yes, 0: No (difficulty breathing)
SWALLOWING.DIFFICULTY	1: Yes, 0: No (difficulty swallowing)
CHEST.PAIN	1: Yes, 0: No (chest pain)
LUNG_CANCER	1: Yes, 0: No (lung cancer diagnosis)

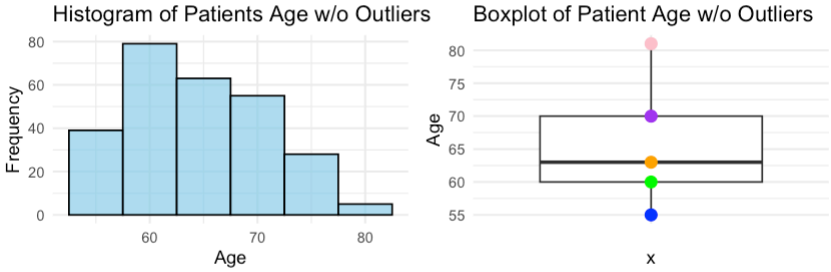
2.2 Data Cleaning

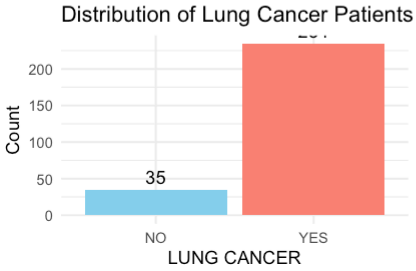
Rows with missing values were removed to prevent skewing of results. The dataset was filtered to include only individuals aged 55 and older, as this demographic is more susceptible to lung cancer. Categorical variables, like gender and smoking status, were converted into binary factors. Duplicate rows were retained, as identical responses from individuals of the same age and gender are valid within the context of a YES/NO survey.

Outliers were identified using boxplots generated by `ggplot`. The removal of the outlier whose age is 87 is justified as it represented only a single data point in that age range. This would make predictions for that age range unreliable compared to other age groups with more substantial data representation. Additionally, the distribution of lung cancer diagnoses was assessed, highlighting a notable imbalance. A higher count of positive cases than negatives cases will be addressed in the following models.

‘Percentage of Records with Lung Cancer: 86.99 %’





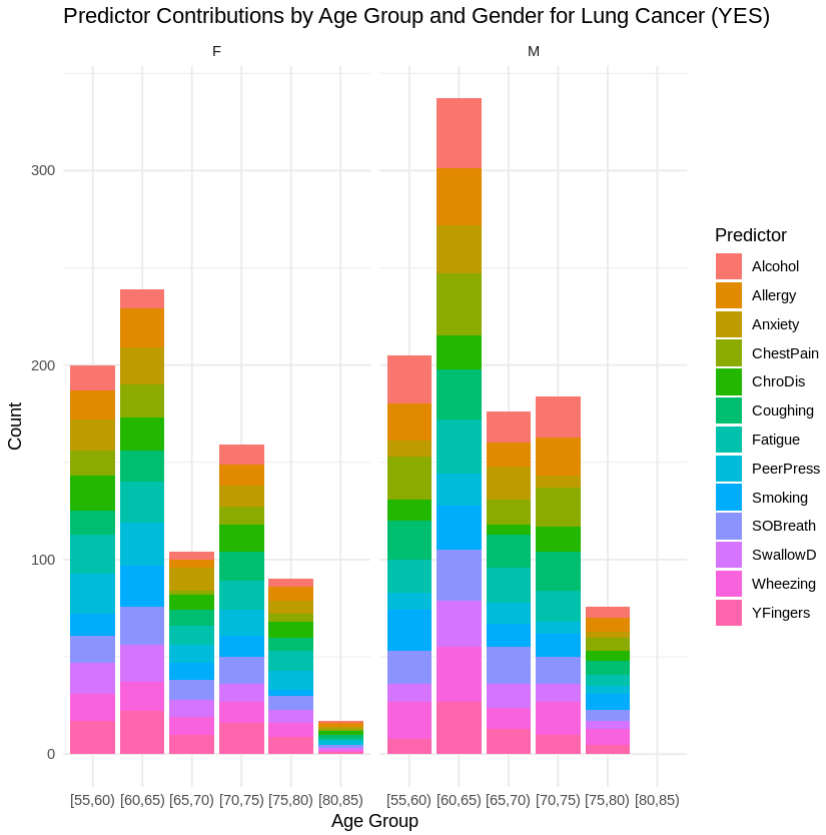


3.0 Exploratory Data Analysis

3.1 Stacked Bar Chart

The stacked bar chart reveals a significant data imbalance, with 86.99% of records indicating positive lung cancer cases (234 "YES" cases) and only 13% negative cases (35 "NO" cases). Due to the imbalanced dataset, there is a tendency to predict 'YES' more frequently. Therefore, models using the balanced accuracy metric are selected to address this imbalance.

Males have a higher incidence of lung cancer than females, especially in the 60-65 age range, which had the highest number of diagnoses, followed by the 55-60 range. The 65-70 and 70-75 age groups showed similar rates, with females slightly outnumbering males in the 70-75 group. The 80-85 age group had no diagnoses among males. Alcohol consumption, allergies, and anxiety are leading factors associated with lung cancer in both genders.



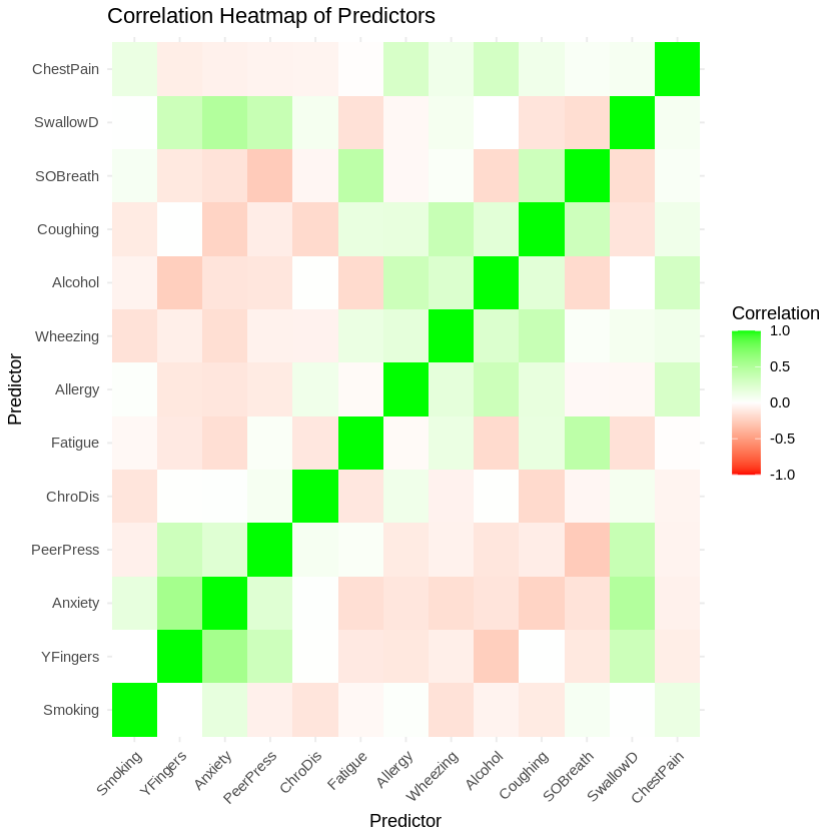
3.2 Contingency Table

The contingency table for lung cancer incidence reveals a slight gender disparity, with 105 female and 129 male cases among the affected population. However, our primary focus is on the binary outcomes of lung cancer diagnosis, which indicates that while gender differences exist, they do not significantly alter the overall prevalence of the disease.

3.3 Pearson's Correlation Heatmap

Based on the Pearson's correlation heatmap, multicollinearity is minimal, with no significant issues for model building. The threshold for moderate and high correlation was set at 0.6 and 0.8, respectively. The highest correlation observed was between Yellow Fingers and Anxiety, with $r = 0.56$.

```
No highly correlated predictor pairs found above the threshold.
```



3.4 Chi Squared Test

We performed a chi-square test of independence for all unique pairs of predictor variables in the dataset to assess their associations. The goal is to identify non-significant pairs (p-value > 0.05) and determine if any predictors can be removed due to having no significant relationships.

```
No predictors can be removed based on non-significant associations.
```

No predictors can be removed, indicating all predictors contribute some meaningful information to the dataset.

3.5 Summary of Statistics

Sections 3.1-3.4 can be summarised as follows.

Statistic	Value
Total Cases	269
Positive Cases (“YES”)	234 (86.99%)
Negative Cases (“NO”)	35 (13.01%)
Male Cases	129
Female Cases	105
Age Group with Most Diagnoses	60-65 years
Top Predictors for Lung Cancer	Alcohol Consumption, Allergies, Anxiety
Highest Pearson Correlation (r)	Yellow Fingers and Anxiety (r = 0.56)
Chi-Square Significant Associations	None are significant enough to be removed
Non-Significant Pairs (to Remove)	Some age-gender combinations, lifestyle factors

Table 1: Summary of sections 3.1-3.4 exploratory data analysis.

4.0 Split Training and Test Dataset

The dataset was partitioned into training and testing subsets in a 70-30 ratio for subsequent model building.

5.0 Gradient Boosting Algorithms

5.1 Extreme Gradient Boosting (XGBoost)

Extreme Gradient Boosting (XGBoost) is a supervised machine learning algorithm for binary classification and regression tasks. It constructs decision trees sequentially to optimize an objective function, even with small datasets. The model's prediction is given by:

$$f(x) = \sum_{k=1}^{K} f_k(x_i)$$

where k is the number of trees. The objective function combines the training loss and regularization term:

$$\text{Obj}(\theta) = L(\theta) + \Omega(\theta)$$

The log-loss function is used for binary classification:

$$L(\theta) = \sum_{i=1}^{N} \left[y_i \log(1 + e^{-\hat{y}_i}) + (1 - y_i) \log(1 + e^{\hat{y}_i}) \right]$$

Regularization prevents overfitting:

$$\Omega(\theta) = \alpha\lambda \sum_{j=1}^{p} |\theta_j| + (1 - \alpha)\lambda \sum_{j=1}^{p} \theta_j^2$$

Key parameters include `max.depth = 2` and `eta = 1`. The `early_stopping_rounds` parameter (set to 10) terminates training if the log-loss does not improve, and `scale_pos_weight = 0.1495726` addresses class imbalance. These settings eliminated the need for ROSE. The model achieved its best log-loss of 0.1635 at 50 trees.

Performance was evaluated using balanced accuracy, which accounts for sensitivity and specificity:

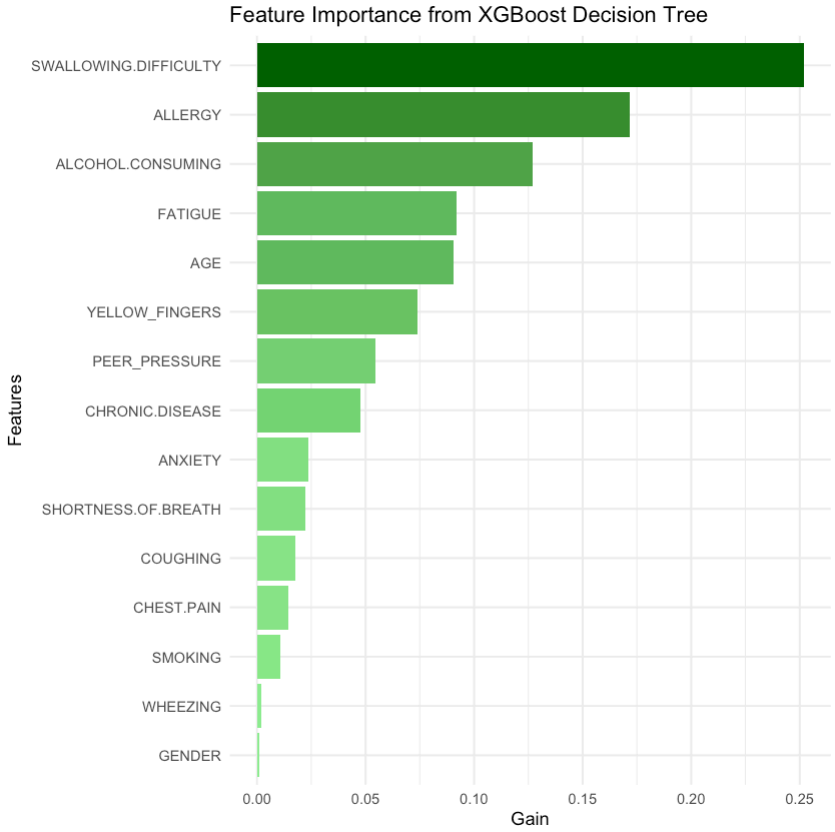
$$\text{Balanced Accuracy} = \frac{1}{2} \left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP} \right)$$

At a threshold of 0.1, the model achieved a training balanced accuracy of 87.82% and a test balanced accuracy of 75.18%, with a 12.64% difference indicating slight overfitting. The training recall was 76.9%, while the test recall was 55.5%, showing a 21.4% gap. Important predictors identified were swallowing difficulty, allergy, and alcohol consumption, aligning with exploratory data analysis.

```
Scale Pos Weight Parameter for XGBoost =  0.1495726
```

'Best Iteration for Log Loss XGBoost Tree: 167 with 0.151997906835292'

```
Training Recall:  0.7692308
Test Recall:  0.5555556
Recall Difference:  0.2136752
```



5.2 Adaptive Boosting (ADABoost)

Adaptive Boosting (AdaBoost) is a supervised ensemble method that combines weak learners to create a robust model. It adjusts sample weights rather than optimizing a loss function, indirectly influencing feature importance.

Initially, all data points have equal weights:

$$w(x_i, y_i) = \frac{1}{N}, \quad \text{for } i = 1, 2, \ldots, N$$

For each feature, decision stumps are created, and the Gini index is calculated:

$$\text{Gini} = 1 - (p_1^2 + p_2^2)$$

where p_1 and p_2 are class probabilities. The stump with the lowest Gini index is selected as the classifier. Its importance is determined by:

$$\alpha_t = \frac{1}{2} \log \left(\frac{1 - \text{total error}}{\text{total error}} \right)$$

Weights are updated iteratively:

$$w_i^{(t+1)} = w_i^{(t)} \cdot e^{\pm \alpha_t}$$

Correctly classified points have reduced weights, while misclassified points are given higher weights, enabling the model to focus on difficult instances.

Parameters used were `tree_depth = 1` and `nrounds = 100`. Deeper trees and more iterations caused significant overfitting. The absence of `early_stopping_rounds` limited control over stopping criteria.

AdaBoost achieved a training balanced accuracy of 89.11% and a test balanced accuracy of 70.27%, with an 18.84% difference, indicating severe overfitting. Training recall was 97.5%, while test recall was 44.4%, showing a 32.5% difference, highlighting high overfitting likely caused by limited hyperparameter tuning.

```
Balanced Accuracy: 0.8911073
Train Recall: 0.9745223
Test Recall:  0.4444444
```

5.3 Categorical Boosting (CatBoost)

Categorical Boosting (CatBoost) uses decision trees and iteratively refines predictions. It converts categorical features to numerical ones through quantization using the **BinarisedTargetMeanValue** function:

$$\text{Ctr} = \frac{\text{countInClass} + \text{prior}}{\text{totalCount} + 1}$$

- countInClass: Sum of label values divided by the max label value k.
- totalCount: Instances with the feature value.
- prior: A constant defined by parameters.

CatBoost splits features by minimizing penalties and uses gradient descent to optimize a loss function:

$$L(f(x), y) = \sum_i w_i \cdot l(f(x_i), y_i) + \lambda \|\theta\|_2$$

- $l(f(x), y)$: Loss function value.
- w_i: Object weights.
- $\lambda\|\theta\|_2$: Regularization.

Subsequent trees approximate gradients of the loss:

$$g_i = -\frac{\partial l(a, y_i)}{\partial a}, \quad a = F_{T-1}(x_i)$$

Gradient approximation quality is measured by the **L2 score function**:

$$L2 = -\sum_i w_i \cdot (a_i - g_i)^2$$

CatBoost uses symmetric trees, splitting features consistently across leaves. Feature importance is calculated using prediction changes:

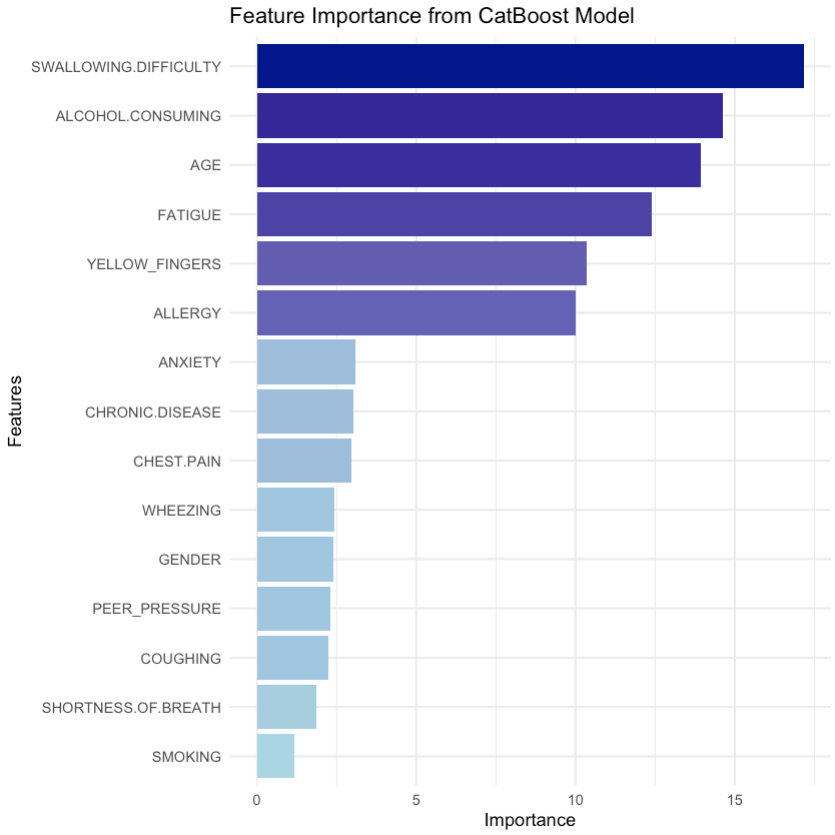
$$F_{\text{importance}} = \sum_{\text{trees, leaves}} ((v_1 - \text{avr})^2 \cdot c_1 + (v_2 - \text{avr})^2 \cdot c_2)$$

where: - avr $= \frac{v_1 \cdot c_1 + v_2 \cdot c_2}{c_1 + c_2}$, - c_1, c_2: Weights of left/right leaves, - v_1, v_2: Predicted values.

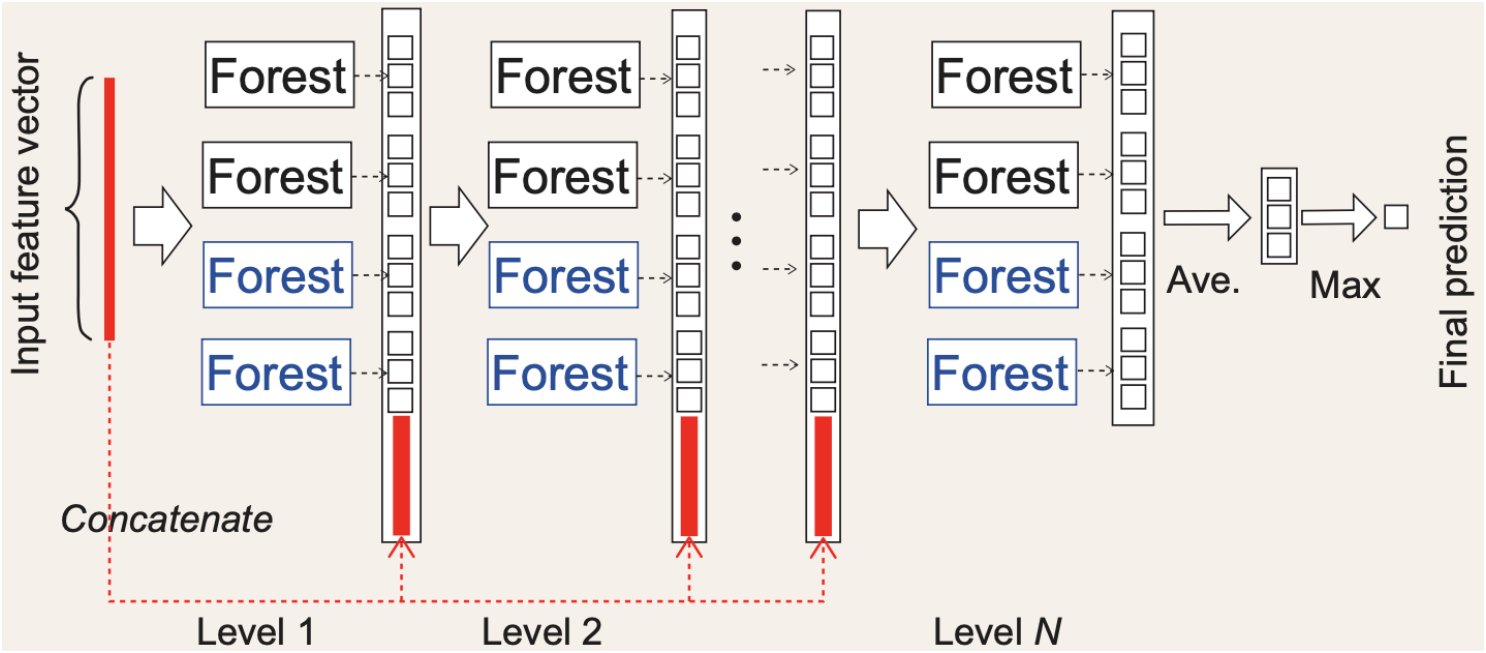
The `class_weights` parameter ((1, COUNT(Negative)/COUNT(Positive))) addressed class imbalance, negating the need for ROSE. **Early stopping rounds** were set to 10 for automatic termination if no improvement occurred.

CatBoost's default of 1,000 trees led to test accuracy of 79% and training accuracy of 99% at a threshold of 0.1, indicating severe overfitting. Reducing to 100 trees improved balanced test accuracy to **82.40%** and training accuracy to **93.95%**, with an 11.55% difference indicating moderate overfitting. Training recall was 100%, while test recall was 77.8% (22.2% difference). The most important variables were swallowing difficulty, alcohol consumption, and age, consistent with exploratory data analysis and XGBoost findings.

```
Scale Pos Weight Parameter for CatBoost =  0.1495726
Train Recall:  1
Test Recall:  0.7777778
Recall Difference:  0.2222222
```



6.0 Deep Forest Model



In Deep Forest, proposed by Zhou and Feng (2017) [11], each cascade layer is represented as an ensemble of decision tree forests. This is not limited to decision tree forests; other methods, such as XGBoost, can also be used. It implements the idea of representation learning through layer-by-layer processing of raw features. Each level of the cascade structure receives feature information processed by its preceding level and outputs the processed result to the next level.

As shown in the figure, each level of the cascade consists of several Random Forests that generate probability class vectors, which are concatenated with each other and the original input. After the final level, each prediction from the decision trees is averaged to obtain the final prediction.

In this implementation, we select the predictor in hyperparameter tuning to be XGBoost and LightGBM, where the equation for XGboost in **SECTION 5.1**

LightGBM

LightGBM [17] is a distributed high-performance framework that uses decision trees for ranking, classification, and regression tasks. In comparison to XGBoost, it provides faster performance in training.

Given a training set $\{(x_1, y_1), (x_2,y_2), \ldots, (x_n, y_n)\}$, where x are the data samples and y are the class labels. $F(x)$, represent the estimated function and the optimization goal of Gradient Boosting Decision Tree is to minimize the loss function $L(y, F(x))$:

$\hat{F} = \arg\min_F E_{x,y}[L(y, F(x))]$

Using a line search to minimize the loss function,

$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$

where $\gamma_m = \arg\min_\gamma \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i))$, m is the iteration number, $h_m(x)$ represents the base decision tree.

LightGBM uses GOSS to determine the split point via calculating variance gain. First, sorting the absolute values of the gradients of the training examples in descending order and the top a × 100% data samples of gradient values are selected called A. Then the subset B whose size is b × |Ac| is randomly selected from the retained samples Ac. Finally, the instances are split through the estimated variance $\widetilde{V}_j(d)$ on A □ B.

$$\widetilde{V}j(d) = \frac{1}{n}\left(\frac{(\sum x_i \in A_l g_i + \frac{1-a}{b}\sum_{x_i \in B_l} g_i)^2}{n_l^j(d)} + \frac{(\sum_{x_i \in A_r} g_i + \frac{1-a}{b}\sum_{x_i \in B_r} g_i)^2}{n_r^j(d)}\right) \quad (11)$$

where $A_l = x_i \in A : x_{ij} \leq d$, $A_r = x_i \in A : x_{ij} > d$, $B_l = x_i \in B : x_{ij} \leq d$, $B_r = x_i \in B : x_{ij} > d$, g_i represents the negative gradient of the loss function, $\frac{1-a}{b}$ is employed to normalize the sum of gradients.

Overall Architecture

The deep forest model can be broken down into the following layers: 1. **First Layer**
- The initial raw feature vector is supplied to the first layer. - Configured estimators (n in total) are executed, producing prediction vectors: - For binary classification: outputs two values. - For multi-class classification: outputs `n_class` values.

2. **Second Layer and Subsequent Layer**
    - Prediction vectors from the first layer are concatenated with the initial raw vector to form the input of the second layer.
    - The dimensionality of the features increases with each subsequent layer.
3. **Penultimate Step**
    - An average of last layer's prediction is calculated.
4. **Final Output**
    - A max function is applied to obtain the final prediction.

6.1 Import and Library and Data using Numpy with ROSE (appendix)

6.2 Hyperparameter Tuning Using Cross Validation where Recall as the Metrics

```
Best result with maximum CV score with XGBoost: 0.9666667
[1] "n_estimators= 4, n_trees=auto, max_layers=2"


Best result with maximum CV score with LightGBM: 0.9666667
[1] "n_estimators= 9, n_trees=auto, max_layers=1"
```

6.3 Best Deep Forest Model

```
[1] "Best Deep Forest Model: n_estimators = 4, n_trees = 'auto', max_layers = 2, predictor = 'xgboost'
Training Balanced Accuracy: 98.939 %
Training Recall: 98.889 %

Testing Balanced Accuracy: 78.195 %
Testing Recall: 92.105 %

Testing F1 Score: 0.795
```

7.0 Logistic Regression

7.1 Logistic Regression without Regularization

Since our study is concerned with the binary classification of the chance of people getting lung cancer, we apply logistic regression on the binary response variable `LUNG_CANCER`, with output ranging from 0 to 1. Let p(X) be the estimated probability P(Y=1|X), we have

$$log\frac{\text{p(X)}}{\text{1 - p(X)}} = \beta_0 + \beta_1 \text{X}_1 + ... + \beta_p \text{X}_p$$

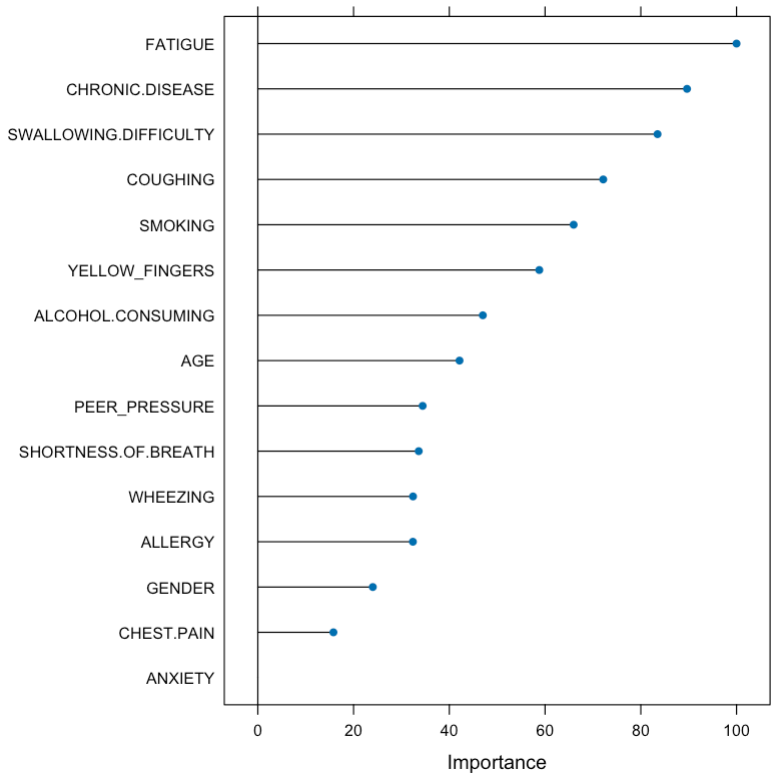
The loss function is

$$J(\beta) = -\sum_i (y^i log p(x^i) + (1 - y^i) log(1 - p(x^i)))$$

The goal is to find the optimal coefficients β that minimize the loss function, i.e., maximize the likelihood of observing the actual labels given the predicted probabilities. The loss is then calculated based on how well the model's predicted probabilities match the actual class labels.

First, we train a logistic regression model without undergoing any regularization. 5-fold cross validation is implemented here to strike a good balance between training and testing data set size in order to overcome overfitting.

```
5-fold CV error is  0.0868263
Training balanced accuracy:  0.8911073
Testing balanced accuracy:  0.7813853
```



7.2 Least Absolute Shrinkage and Selection Operator (LASSO)

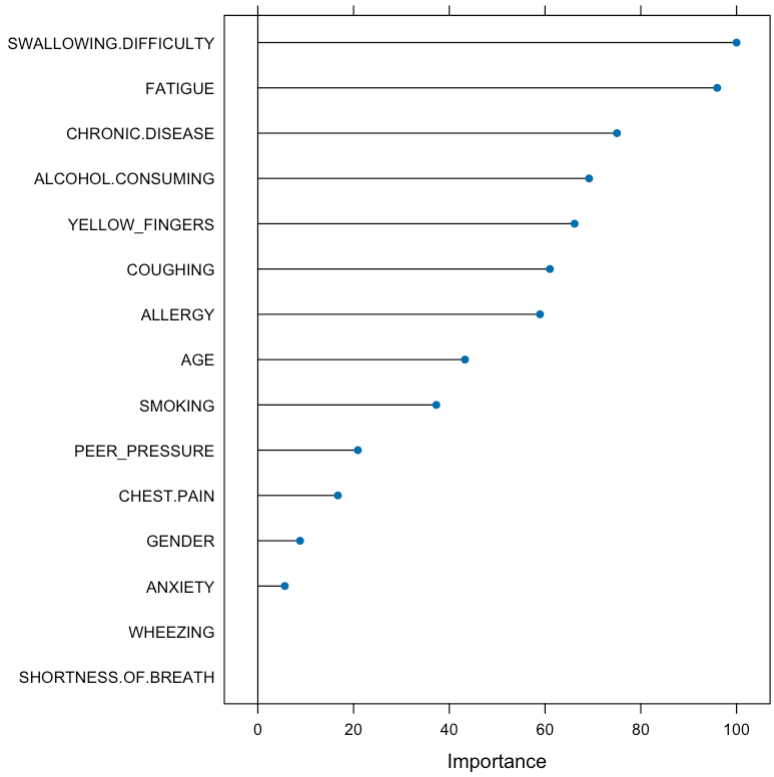
After that, we consider two different regularizations, Elastic Net and Least Absolute Shrinkage and Selection Operator (LASSO). Regarding the choice of the penalty term (λ), 20 evenly distributed numbers from the sequence 0.01 to 1.0 are selected such that it gives enough granularity without taking too much time for computation. The range of λ spans from small values (0.01) to a relatively large value (1.0), which shrinks more coefficient to 0 to perform feature selection.

LASSO regression, also known as L1 regularization, adds a penalty term to penalize the coefficients' absolute values to avoid overfitting and enhance the accuracy of model. The regularization term prevents model from fitting in noises, thus reducing the chance of overfitting. Its loss function is

$$L_L(\beta) = J(\beta) + \lambda \sum_{j=1}^{p} |\beta_j|$$

where $ J(□) $ is the loss function of logistic regression and the term $ λ∑_{j=1}^{p}|□_j| $ is the penalty term. When λ is small, LASSO tends to give a model similar to regular linear regression with smaller coefficients but not necessarily sparse; when λ is large, it produces a simpler model with fewer non-zero coefficients. This effectively performs feature selection that only relevant parameters is suspected to make the final prediction.

```
5-fold CV error is  0.0880117
Training balanced accuracy:  0.8879226
Testing balanced accuracy:  0.8008658
```



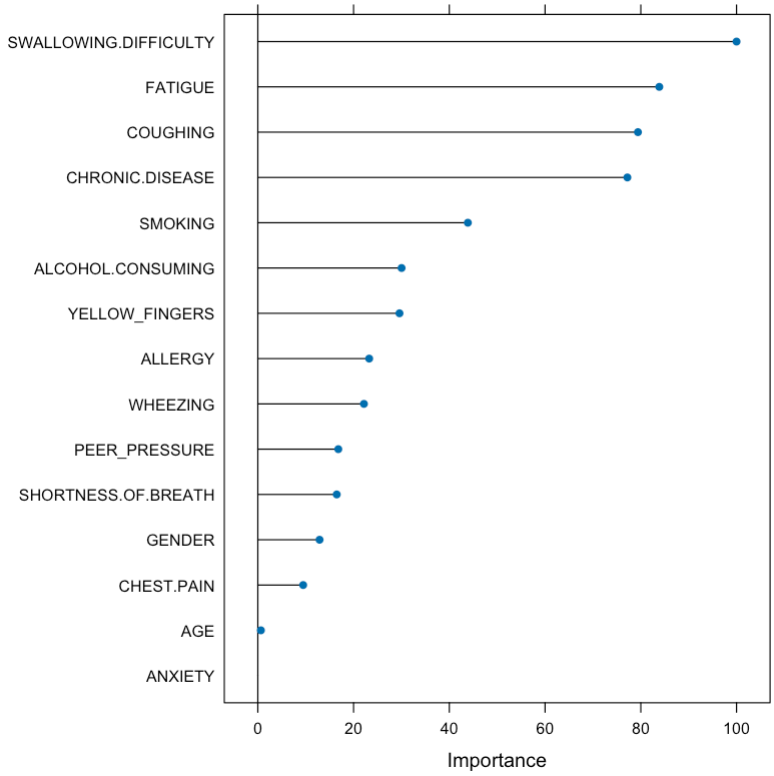
7.3 Elastic Net

Elastic net regularization is a combination of ridge and LASSO regression. In this study, ridge regression is not considered separately as it does not perform feature selection. The loss function of elastic net regularization is

$$L_E(\beta) = J(\beta) + (1-\alpha)\lambda \sum_{j=1}^{p} \beta_j^2 + \alpha\lambda \sum_{j=1}^{p} |\beta_j|$$

The first term is the residual sum of square, which measures how well the model fits the data; the second term is the L1 penalty term, while the last term is the L2 penalty term. The optimal values of the hyperparameters λ and α are selected by cross-validation to construct the best model for prediction.

```
5-fold CV error is  0.1099099
Training balanced accuracy:  0.8911073
Testing balanced accuracy:  0.7813853
```



7.4 Summary of results

All three models selected `SWALLOWING.DIFFICULTY` and `FATIGUE` as the variables that have the largest effect on whether one will get lung cancer. For LASSO, coefficients of `WHEEZING` and `SHORTNESS.OF.BREATH` are shrinked to zero. Logistic regression without regularization gives 0.78 test balanced accuracy, while LASSO and elastic net give 0.80 and 0.78 respectively. By comparing their test balanced accuracy, we can conclude that LASSO performed the best among the three models, even though it is slightly overfitted.

8.0 Artificial Neural Networks

8.1 ROSE Method for Imbalanced Data

We first need to fix the issue of the data being imbalanced. Thus, we will be implementing ROSE (Random Over-Sampling Examples) method, which is a data resampling technique that aims to address the issue of imbalanced datasets.

ROSE generates synthetic data points by drawing random samples from the feature space of existing minority and majority classes. These samples are created by introducing some random noise into the original data, resulting in a more balanced dataset that allows machine learning models to learn better representations of both classes.

8.2 Artificial Neural Network (ANN) Overview

Structure

- **Input Layer**: Accepts features from the dataset. The number of neurons equals the number of features.
- **Hidden Layers**:
  - **Dense Layers**: Fully connected layers.
  - **ReLU Activation**: Introduces non-linearity:
    $$f(x) = \max(0, x)$$
    enabling complex pattern learning.
  - **Batch Normalization**: Speeds up training and improves stability.
  - **Dropout**: Prevents overfitting by randomly deactivating neurons.
- **Output Layer**: A single neuron with a **sigmoid activation**:
  $$f(x) = \frac{1}{1 + e^{-x}}$$
  outputs the probability of lung cancer.

Learning Components

- **Loss Function**: Binary cross-entropy measures the difference between predicted probabilities and actual labels:
  $$L = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]$$
  Where (y_i) is the true label, (p_i) is the predicted probability, and (N) is the sample size.
- **Optimizer**: **Adam** combines adaptive learning rates (**AdaGrad**) and momentum (**RMSprop**) for efficient weight updates.

```
Training Recall: 0.9545455
Test Recall: 0.952381
Training Balanced Accuracy: 0.4761905
Test Balanced Accuracy: 0.475
```

8.4 Summary of Results

The ANN model demonstrates reasonable performance, achieving a training balanced accuracy of 83.9% and a test balanced accuracy of 80.4%, which suggests effective learning and generalization to unseen data. The training and test recall values of 73.9% and 64.5%, respectively, indicate that the model captures a significant proportion of positive cases while maintaining class balance.

9.0 Topological Data Analysis

9.1 Persistent Homology

Persistent Homology in Topological Data Analysis (TDA)

Persistent Homology, a key method in TDA, analyzes the topological structure of data by examining features like connected components (0D), loops (1D), and voids (2D) across scales. These features are robust to continuous distortions (e.g., stretching or bending) but not tearing or gluing.

Vietoris-Rips Filtration

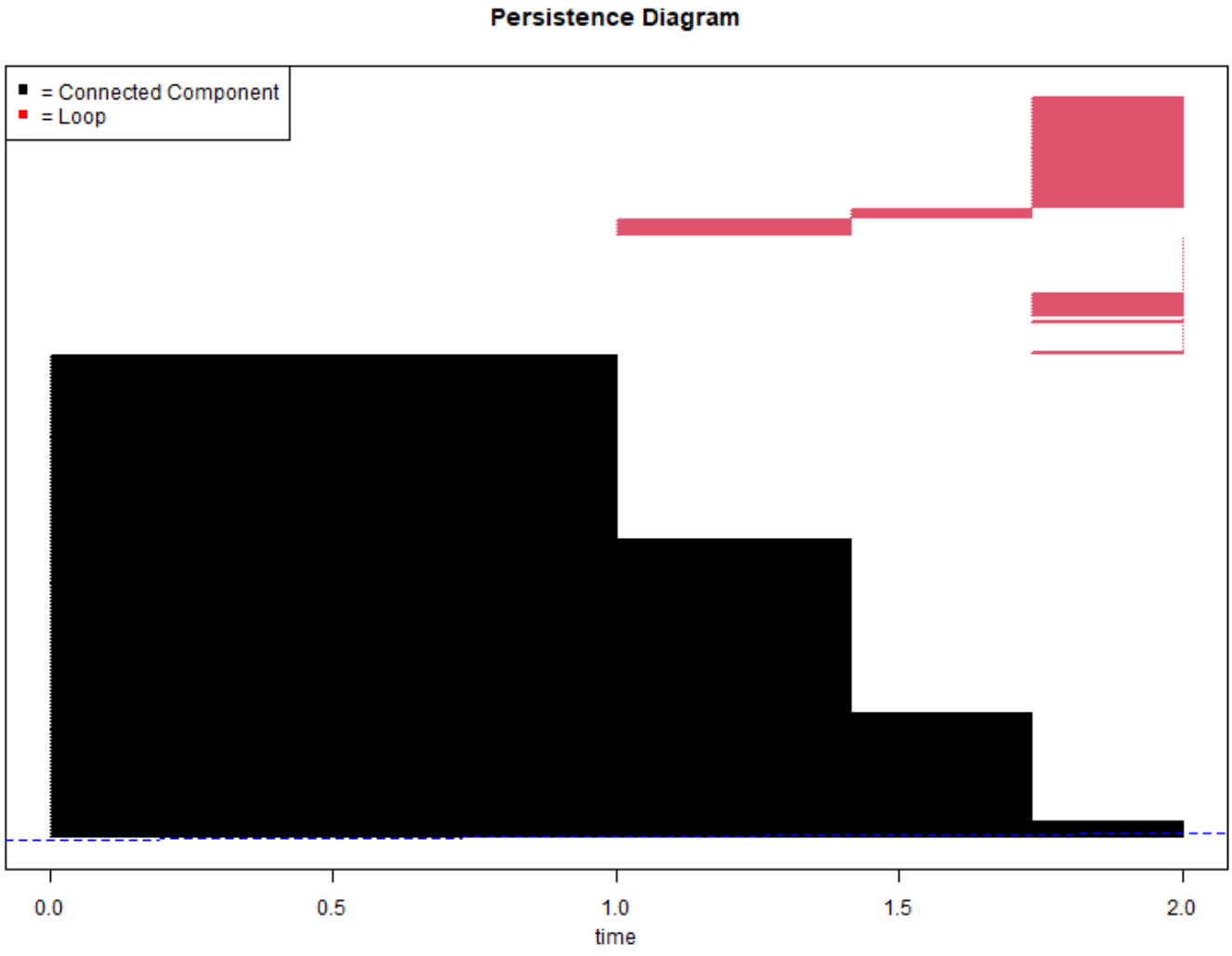
Using the Vietoris-Rips filtration, disks grow around data points based on Euclidean distance. As the radius increases, connections form, creating higher-dimensional features. The process is visualized in a **persistence diagram**, where features are represented by their birth and death scales. Persistent features (above the diagonal) indicate statistically significant structures, while transient ones (below) may represent noise.

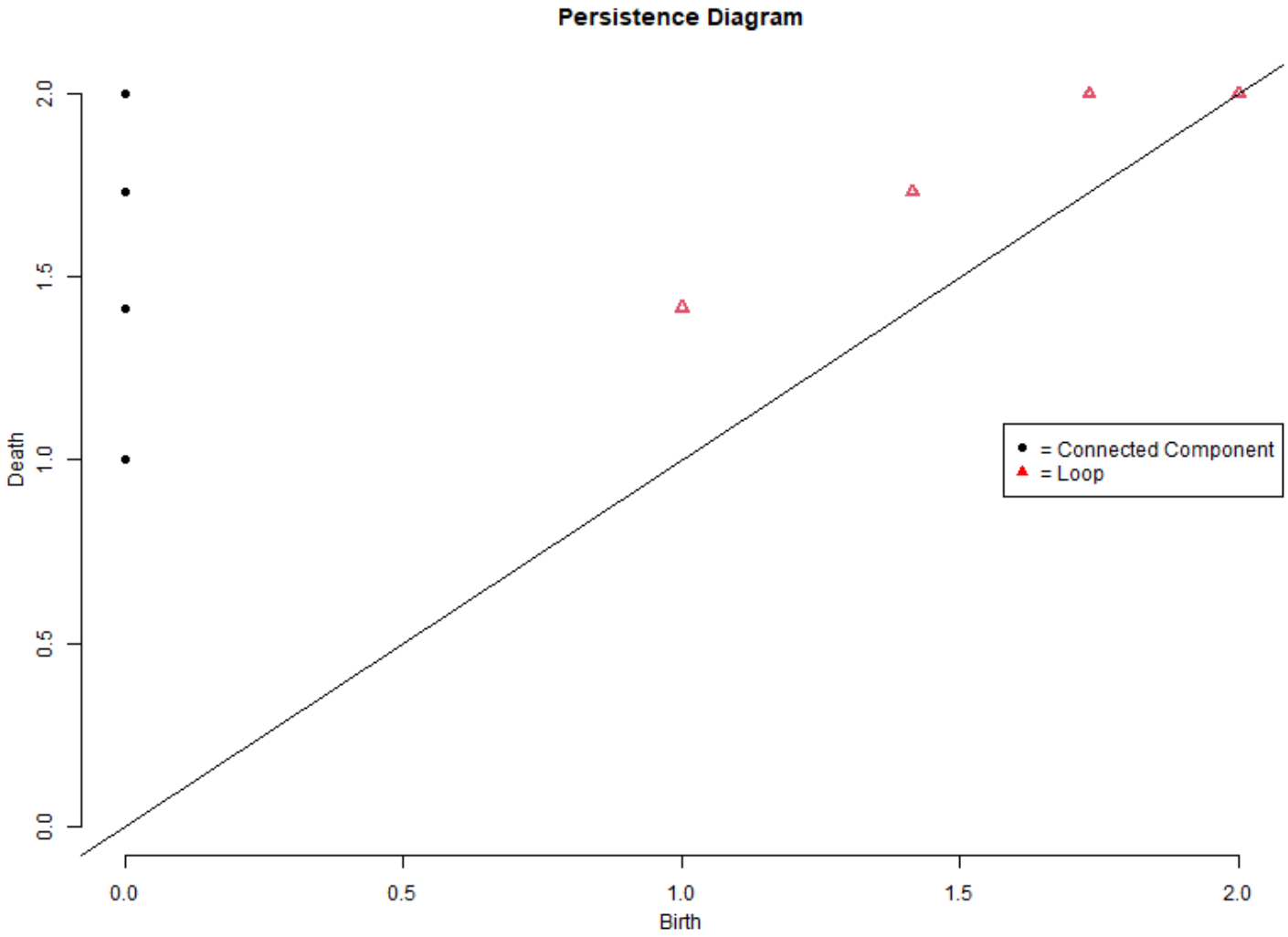
Key Observations

- **Parameter Settings**: The filtration used $maxscale = 2$ and $maxdimension = 1$ to balance computational complexity.
- **Persistence Diagram**: At $r = 0$, each data point starts as its own component. Four loops (red triangles) appear after $r = 1.0$, with one transient loop emerging and disappearing at $r = 2.0$.
- **Binary Data Limitation**: Due to the sparsity of binary combinations, the simplicial complex contained only four unique vertices, limiting the insights from TDA. These findings suggest TDA may not be well-suited for this binary classification task.

Conclusion

While TDA provides a unique perspective on data, its application to binary datasets should be approached with caution. This study highlights its limitations in identifying meaningful topological features for binary classification problems.





10.0 Results & Discussion

Model Performance Evaluation

The models were selected for binary classification given the class imbalance and predominantly categorical variables. Minimal hyperparameter tuning was applied, except for logistic regression and deep forest, due to computational constraints. The table below summarizes the performance of trained models.

To assess model performance, the difference between training and test balanced accuracy was examined. The optimal model minimizes overfitting while achieving high test balanced accuracy.

Model	Training Balanced Accuracy	Test Balanced Accuracy	Difference	Train Recall	Test Recall	Status
Extreme Gradient Boosting	87.82%	75.18%	12.64%	76.9%	55.5%	Mild Overfitting
Adaptive Boosting	89.11%	70.27%	18.84%	97.5%	44.4%	Severe Overfitting
Categorical Boosting	**93.95%**	**82.40%**	**11.55%**	**100%**	**77.8%**	**Mild Overfitting**

Model	Training Balanced Accuracy	Test Balanced Accuracy	Difference	Train Recall	Test Recall	Status
Logistic Regression	89.11%	78.14%	10.97%	-	-	Mild Overfitting
LASSO	**88.79%**	**80.09%**	**8.70%**	**-**	**-**	**Mild Overfitting**
Deep Forest	98.94%	78.20%	20.74%	98.89%	92.11%	Severe Overfitting
Elastic Net	89.11%	78.14%	10.97%	-	-	Mild Overfitting
Artificial Neural Network	83.9%	80.4%	3.5%	73.9%	64.5%	Slight Overfitting

Table 2: Summary of Trained Models and Performance.

Key Insights

- **Best Models**:
  - **Categorical Boosting** achieves the highest test balanced accuracy, excelling with categorical features.
  - **LASSO** minimizes overfitting due to its simplicity and feature selection capabilities, resulting in the smallest accuracy difference.
- **Gradient Boosting**: Well-suited for imbalanced datasets due to ensemble learning and direct handling of categorical features. With hyperparameter tuning, performance could improve further.
- **Deep Forest**: Achieves the best performance in terms of recall metrics; however, it was overfitted when compared to balanced accuracy.
- **ANN and Stacking Ensemble**: Show minimal overfitting, balancing test performance and robustness.

11.0 Conclusion

This project effectively evaluates the performance of various machine learning models in predicting lung cancer outcomes within a dataset of survey questions. Categorical Boosting has proven to be the most effective model in our assessment, achieving the highest test balanced accuracy and demonstrating robust capabilities in handling categorical features. Given its performance, the next steps involve rigorous hyperparameter tuning to optimize its predictive capabilities further. However, this would involve more computational power and time. Additionally, exploring larger and more diverse datasets could enhance the model's effectiveness. Ultimately, this refined model can be integrated into a practical application, allowing users to input responses to a series of yes/no survey questions related to lung cancer risk. This application would then provide a probability estimate for lung cancer, offering valuable support in healthcare decision-making.

12.0 References

1. **Lung Cancer**. World Health Organization. Accessed 18 Nov. 2024.
2. Pastorino, U., et al. "Prolonged Lung Cancer Screening Reduced 10-Year Mortality in the MILD Trial: New Confirmation of Lung Cancer Screening Efficacy." *Annals of Oncology: Official Journal of the European Society for Medical Oncology*, vol. 30, no. 7, July 2019, pp. 1162–69. PubMed, https://doi.org/10.1093/annonc/mdz117.
3. Introduction to Boosted Trees — Xgboost 2.1.1 Documentation. https://xgboost.readthedocs.io/en/stable/tutorials/model.html. Accessed 18 Nov. 2024.

4. **Xgb. Train function**—Rdocumentation. (n.d.). Retrieved 28 October 2024, from https://www.rdocumentation.org/packages/xgboost/versions/1.7.8.1/topics/xgb.train
5. **Xgboost function**—Rdocumentation. (n.d.). Retrieved 28 October 2024, from https://www.rdocumentation.org/packages/xgboost/versions/0.4-4/topics/xgboost
6. Anshul. "Guide on AdaBoost Algorithm." Analytics Vidhya, 15 Sept. 2021, https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/.
7. "ML | Gini Impurity and Entropy in Decision Tree." GeeksforGeeks, 14 July 2020, https://www.geeksforgeeks.org/gini-impurity-and-entropy-in-decision-tree-ml/.
8. **How training is performed**. (n.d.). Retrieved 28 October 2024, from https://catboost.ai/en/docs/concepts/algorithm-main-stages
9. **Score functions**. (n.d.). Retrieved 28 October 2024, from https://catboost.ai/en/docs/concepts/algorithm-score-functions
10. **Quantization**. (n.d.). Retrieved 28 October 2024, from https://catboost.ai/en/docs/concepts/quantization
11. Zhou, Zhi-Hua, and Ji Feng. "**Deep Forest: Towards an Alternative to Deep Neural Networks**." Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Aug. 2017, https://doi.org/10.24963/ijcai.2017/497. Accessed 10 June 2022.
12. Provost, Simon. "**Diversity and a Novel Deep Learning Model Called Deep Forest: Applying an Important Concept to a Promising Framework**." Medium, 10 Dec. 2021, https://simon-provost.medium.com/how-to-use-its-own-base-learner-to-improve-diversity-deep-forest-model-a-novel-deep-learning-9200165e2a2e. Accessed 28 Oct. 2024.
13. Duzhin, Fedor, and Ran Deng. "Topological Data Analysis Helps to Improve Accuracy of Deep Learning Models for Fake News Detection Trained on Very Small Training Sets." 2022.
14. Chazal, F., & Michel, B. (2021). "An Introduction to Topological Data Analysis: Fundamental and Practical Aspects for Data Scientists." *Frontiers in Artificial Intelligence*, 4. https://doi.org/10.3389/frai.2021.667963
15. Munch, E. (2017). "A User's Guide to Topological Data Analysis." *Journal of Learning Analytics*, 4(2), 47–61. https://doi.org/10.18608/jla.2017.42.6
16. Fasy, B. T., Kim, J., Millman, D. L., Lecci, F., Clement, M., & Rouvreau, V. (n.d.). "Introduction to the R Package TDA." CMU TopStat Group. https://cran.r-project.org/web/packages/TDA/vignettes/article.pdf
17. Chen, Cheng, et al. "LightGBM-PPI: Predicting Protein-Protein Interactions through LightGBM with Multi-Information Fusion." Chemometrics and Intelligent Laboratory Systems, vol. 191, Aug. 2019, pp. 54–64, https://doi.org/10.1016/j.chemolab.2

13.0 Appendix

Code on https://www.dropbox.com/scl/fo/oidr3vg4a7q1j23zz5fp6/AKX8XOL1PdIFXjw3IjeeORE?rlkey=eiz2jwirip8ahicapqhifsinh&