



# Western Engineering

---

***University of Western Ontario***

ECE 9039:

Machine Learning

---

***Final Project:***

***Lung Cancer Prediction Using a Dataset***

---

***Names: Andres Felipe Lopez Zapata, Justin Chuang, Arash Tash***

***Deadline: April 8<sup>th</sup>, 2024***

***Professor: Ahmed Ibrahim***

## ***Table of Contents***

<b>Introduction</b>	2
<b>Dataset</b>	3
<b>Preprocessing</b>	3
<b>Linear Regression</b>	3
<b>K-Nearest Neighbour</b>	3
<b>Decision Tree</b>	4
<b>Random Forest</b>	4
<b>Stack-Classfier Model</b>	5
<b>Comparison</b>	5
Web Application	6
<b>Conclusion</b>	6
References	7

## **Introduction**

According to the World Health Organization (WHO), lung cancer is the second

most common type of cancer worldwide and is responsible for the highest number of cancer-related deaths each year. The causes of cancer are linked to various factors including genetics, exposure to carcinogens, age, and more. While

there is currently no cure for cancer, early detection plays a critical role in improving survival rates [1] Therefore, it is essential to develop a model that can analyze multiple patient-related factors and determine whether a patient is at high or low risk of developing lung cancer.

In this study different machine learning algorithms have been implemented using the sci-kit learn library to find the optimal method of creating a predictive model for determining if a person has lung cancer [2] The several different models include a k-nearest neighbour, linear regression, and decision trees. Furthermore, using hyperparameter optimization (HPO) to reduce the model loss and improve their accuracy is done before consideration of the model. Using the HPO for the model provides the most optimal initialization to improve the predictive capability [3]

## Dataset

The dataset being utilized comprises sixteen attributes that will be employed to train the model for accurately predicting the likelihood of patients developing lung cancer. The factors under analysis encompass gender, age, smoking status, presence of yellow fingers, anxiety levels, peer pressure, chronic diseases, fatigue, allergies, alcohol consumption, coughing, shortness of breath, difficulty swallowing, chest pain, and the presence of lung cancer.

## Preprocessing

Before training any of the models preprocessing of the data was done to improve the model's ability to converge during training properly. First, the data was adjusted since the independent variables were recorded as a simple 1 for false and 2 for true, This was adjusted to 1 for true and 0 for false, to allow for the data to be one-hot encoded using the get dummy library for python [4] Thus, increasing the total amount used features to train the

models from 15 to 29. Figure 1, shows some of the features used to train the models.

Figure 1. One-Hot Encoded Data Table

## Linear Regression

Linear regression, more especially logistic regression from the sci-kit learn module, was essential to the research of lung cancer prediction utilizing a dataset. With a hyperparameter choice of 1000 maximum iterations, this technique used gradient descent as an optimizer to provide the model with plenty of training opportunities. With a fantastic test accuracy of 0.9797 and a cross-validation accuracy of 0.8181, the linear regression model proved to be effective. These findings demonstrate the utility of logistic regression as a simple yet effective approach for early detection research in lung cancer predictive modelling.

## K-Nearest Neighbour

Since the data is labeled K-nearest neighbor can also be used as a method of prediction. This library allows for the use of its auto feature which automatically determines the best method of computing the nearest neighbours [6] This method provides a relatively different approach from linear regression by determining if the point is true or false depending on the specified number of neighbouring features. Therefore, the number of neighbours is adjusted to improve the accuracy of the model. As shown in Figure 2 the best number of neighbors is 12 since it provides the best cross-validation accuracy of 0.862.

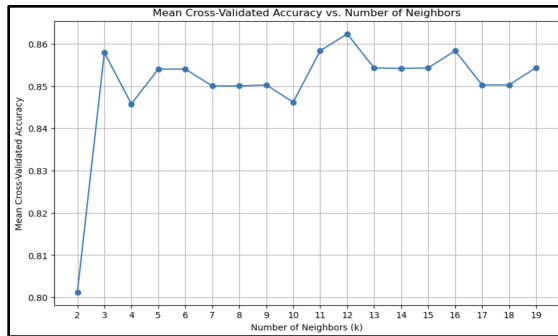


Figure 2. Mean Cross-Validation Accuracy for K-Nearest-Neighbour

## Decision Tree

The basic decision tree was tested to determine the initial potential of a decision tree classifier. Using the Decision Tree Classifier in the sci-kit learn library allows for classification by separating each feature based on the criterion function used. In this case, include Gini indexing, entropy, or logarithmic loss. In addition, the maximum depth of the decision tree must be decided before implementing the model [7]. Therefore, optimization of the hyperparameters for the Decision Tree Classifier model was done using the grid search function from the sci-kit learn library [9]. The best hyperparameters resulted in a decision tree with a Gini indexing criterion and a max depth of 4. A cross-validation of 5 K-folds was done to determine the accuracy of the parameters. This resulted in a cross-validation accuracy of 0.866. In addition, using a training set and test set to split from the original data the test and training data accuracy was also determined to be 0.903 and 0.935 respectively. The final decision tree is shown in Figure 3.

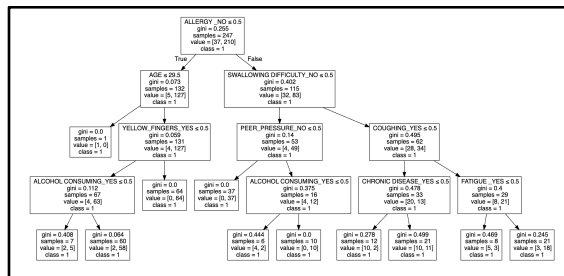


Figure 3. Decision Tree Graph

## Random Forest

A Random Forest Classifier Model was built to test the performance compared to the other models in question. Using the sci-kit learn library [8] The classifier produced 100 decision trees, using the graph viz library [9] the decision trees visualized; the first three trees are shown in Figure 4, Figure 5, and Figure 6.

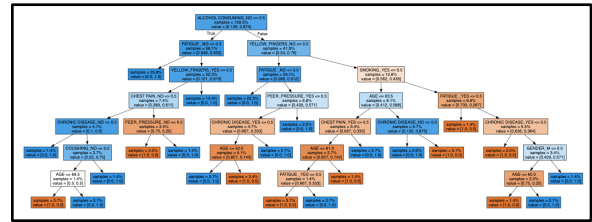


Figure 4. Random Forest Classification Decision Tree

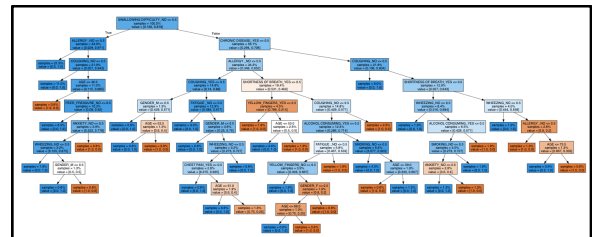


Figure 5. Random Forest Classification Decision Tree #2

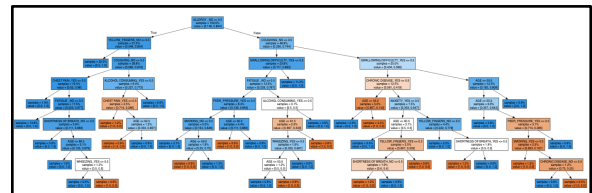


Figure 6. Random Forest Classification Decision Tree #3

These three trees show just how complex the trained model is. Using the sklearn metrics library [11] an accuracy score was calculated based on the testing data split which produced a score of 96.77%. Although the accuracy score is very high, hyperparameter optimization (HPO) was attempted to explore if this model could be further fine-tuned to give a better-performing model [10]. Hyperparameters were optimized to include: 125 estimators, 0.8 max samples, 0.6 max features, and a max depth of 5. A new model was created with the hyperparameters set to the optimal values which provided a cross-validation of accuracy of 0.903.

## XGBoosting

XGBoost was implemented using the XGBoost Classifier from the XGBoost library [12]. This method of constructing decision trees enhances the algorithm's robustness by introducing new decision trees based on the binary classification error of the leaf [13]. Furthermore, multiple decision trees are executed simultaneously to determine the most accurate decision tree for the dataset. Hyperparameters such as the maximum depth of each tree, the learning rate, and the number of decision trees were fine-tuned using the grid search function from sci-kit learn library [9]. The optimal hyperparameters for the model were determined to be a maximum depth of two, a learning rate of 0.1, and 100 decision trees. This xgboost model with these hyperparameters gave a cross-validation accuracy of 0.891. The model also had an accuracy of 0.943 for the training set used and an accuracy of 0.977 for the training set. The final decision tree produced from xgboosting is shown in Figure 7

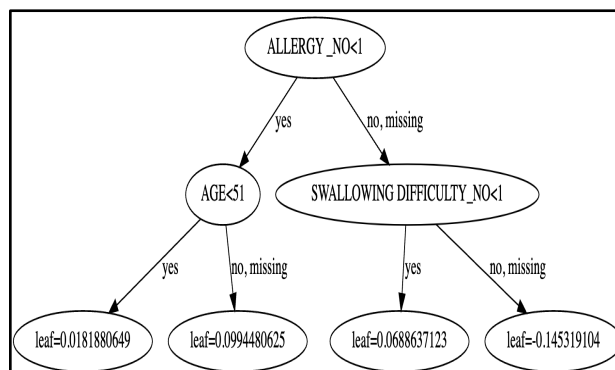


Figure 7. XGBoosting Classifier Descion Tree

## Stack-Classifer Model

Lastly, a compilation of all the previous models studied was put together to attempt to produce a more accurate model. The Stack-Classifer model from sci-kit Learn was used to allow for the stacking of several different models in an XGBoosting algorithm to determine if a combination of the previous models improves the predictive abilities. The cross-validation score shown in Figure 8 provides a mean cross-validation accuracy of 0.891.

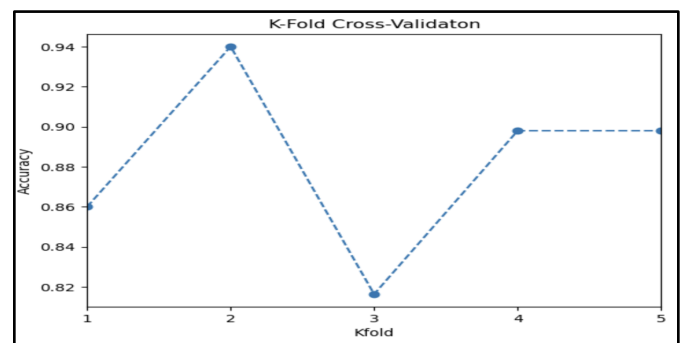


Figure 8. Stack-Classifer K-Fold Validation Accuracy

A visualization of how the final estimator is provided in Figure 9. As can be shown the final decision of the stacking classifier is now based on the accuracy of each model instead of the actual features which provides a more diverse method of classification.

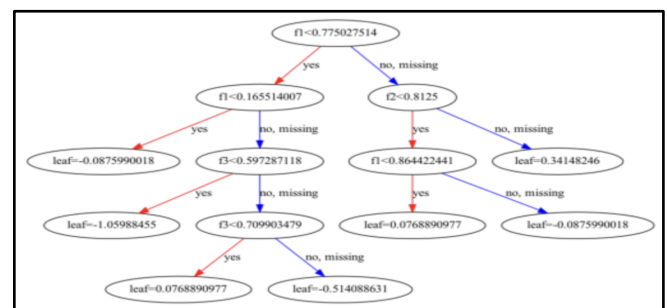


Figure 9. Stack-Classifer Final Estimator Decision Tree (XGBoosting)

## Comparison

To compare all the models' abilities to predict if a person has lung cancer, the confusion matrix of each model was used.

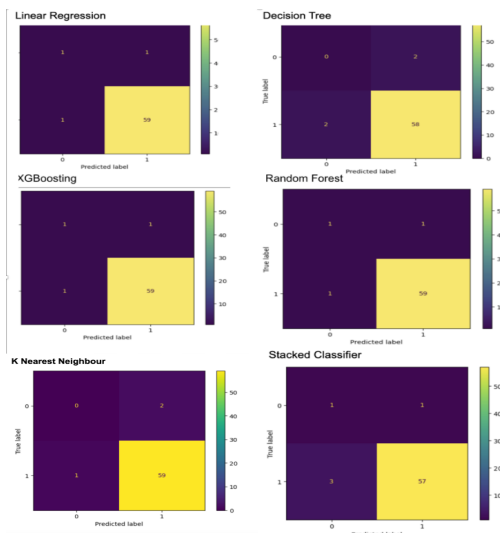


Figure 10. Confusion Matrix of Models Developed

From the confusion matrixes produced it's easy to see that Linear Regression, Random Forest, and XGBoosting have the same confusion matrix that provides a precision of 0.983, recall of 0.983, F1 score of 0.983. The decision tree model confusion matrix provided a precision of 0.967, a recall of 0.967, and an F1 score of 0.967. The K Nearest Neighbour provided a precision of 0.967, a recall of 0.983, and an F1 score of 0.975. Lastly, the Stacked Classifier Model had a precision of 0.983, a recall of 0.95, and an F1 score of 0.966.

$$\frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \quad \text{Equation 1}$$

Precision is used to determine the accuracy of the true positives in comparison with false positives as shown in Equation 1. Therefore, Linear Regression, Random Forest, and XGBoosting have the best overall predictive ability. Between these three models, XGBoosting and the Random Forest provided a higher cross-validation accuracy of around 0.90.

## Web Application

To implement the model and apply it to a practical application, a simple website was built using JavaScript and the React framework. The website acts as a simple form for the user to input their comorbidities. Once the user completes the form and clicks submit, the

website sends all the user data to a Flask server which hosts a platform for the model to run and compute predictions. The application used the Random Forest model to make the prediction. Once the model predicts the risk of lung cancer, the result is displayed so the user can take appropriate actions. The user interface of the website can be seen in Figure 11.

Figure 11. Web Application Using Random Forest Model as Backend

## Conclusion

The best three models for this studies dataset were determined to be Random Forest and XGBoosting. Random Forest provided the most complex model with a larger depth of 5 for its decision tree, as well as the highest cross-validation accuracy of 0.903.

## References

- [1] "Cancer," World Health Organization, <https://www.who.int/news-room/fact-sheets/detail/cancer#:~:text=The%20most%20common%20cancers%20are,and%20lack%20of%20physical%20activity>.
- [2] "Learn," scikit, <https://scikit-learn.org/stable/>
- [3] S. Pandian, "A comprehensive guide on hyperparameter tuning and its techniques," Analytics Vidhya, <https://www.analyticsvidhya.com/blog/2022/02/a-comprehensive-guide-on-hyperparameter-tuning-and-its-techniques/>
- [4] "Pandas.get\_dummies#," pandas.get\_dummies - pandas 2.2.1 documentation, [https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get\\_dummies.html](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get_dummies.html)
- [5] A. Menon, "Linear regression using gradient descent," Medium, <https://towardsdatascience.com/linear-regression-using-gradient-descent-97a6c8700931>
- [6] "Sklearn.neighbors.kneighborsclassifier," scikit, <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>
- [7] "Sklearn.tree.decisiontreeclassifier," scikit, <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>
- [8] "Sklearn.ensemble.randomforestclassifier," scikit, <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>
- [9] "Sklearn.grid\_search.GRIDSEARCHCV¶," scikit, [https://scikit-learn.org/0.17/modules/generated/sklearn.grid\\_search.GridSearchCV.html](https://scikit-learn.org/0.17/modules/generated/sklearn.grid_search.GridSearchCV.html)
- [10] S. Ram, "Mastering random forests: A comprehensive guide," Medium, [https://towardsdatascience.com/mastering-random-forests-a-comprehensive-guide-51307c129cb1#:~:text=n\\_estimators%3A%20Th](https://towardsdatascience.com/mastering-random-forests-a-comprehensive-guide-51307c129cb1#:~:text=n_estimators%3A%20Th)
- [11] "3.3. metrics and scoring: Quantifying the quality of predictions," scikit, [https://scikit-learn.org/stable/modules/model\\_evaluation.html](https://scikit-learn.org/stable/modules/model_evaluation.html)
- [12] "Python package introduction," Python Package Introduction - xgboost 2.0.3 documentation, [https://xgboost.readthedocs.io/en/stable/python/python\\_intro.html](https://xgboost.readthedocs.io/en/stable/python/python_intro.html)
- [13] "What is XGBoost?," NVIDIA Data Science Glossary, <https://www.nvidia.com/en-us/glossary/xgboost#:~:text=XGBoost%2C%20which%20stands%20for%20Extreme,%2C%20classification%2C%20and%20ranking%20problems>.
- [14] "Sklearn.ensemble.StackingClassifier," scikit, <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.StackingClassifier.html>