JUSIKOM PRIMA (Jurnal Sistem Informasi dan Ilmu Komputer Prima)
Vol. 6 No. 1, Agustus 2022 E-ISSN : 2580-2879

THYROID DISEASE CLASSIFICATION ANALYSIS USING XGBOOST MULTICLASS

[1]Haris Samuel Pranada Panjaitan, [2]Agustinus Gulo, [3]Ahmad Haikal Alfi, [4]Okta Jaya Harmaja, [5]Evta Indra*
Program Studi Sistem Informasi, Fakultas Teknologi dan Ilmu Komputer, Universitas Prima Indonesia
Jalan Sampul
E-mail : * oktajaya.h@unprimdn.ac.id

ABSTRAK- Sickness is an unusual condition of the body or mind that causes discomfort, malfunction, or suffering to the sick person. One disorder that occurs due to a lack of health concerns is thyroid disease. The thyroid is a butterfly-shaped endocrine gland near the neck's bottom. The diagnosis of thyroid disease is complicated because the symptoms of thyroid disease can fluctuate based on the rise and fall of thyroid hormones, which increase the utilization of oxygen by the body's cells. In this case, a thyroid examination by a doctor and proper interpretation of clinical data is required to identify thyroid disease. However, the limitations of a doctor due to age and time constraints lead to a lack of interpretation of patient clinical data. Therefore, a study was conducted on the analysis of thyroid disease classification to simplify and speed up the process of diagnosing thyroid disease using the Xgboost Multiclass method, which is expected to get an accuracy value above 90%.
Keywords: Classification, Thyroid, Xgboost Multiclass, Machine Learning

1. INTODUCTION

In 2015 in the United States, there were 62,450 cases of thyroid cancer, with 3 out of 4 cases occurring in women [1]. Based on data from the Cancer Registration Agency of the Indonesian Cancer Foundation in 2005, thyroid cancer ranks 9th out of 10 malignant tumors and is the most common type of endocrine gland malignancy in Indonesia [2].

Sickness is an unusual condition of the body or mind that causes discomfort, malfunction, or suffering to the sick person [3]. A disorder that occurs due to a lack of health concerns is thyroid disease [4]. The thyroid is a butterfly-shaped endocrine gland near the neck's bottom [5]. The thyroid gland's job is to make thyroid hormones that help regulate the body's metabolism [6][7].

The diagnosis of thyroid disease is complicated because the symptoms of thyroid disease can fluctuate based on the fluctuation of thyroid hormone, which increases the utilization of oxygen by the body's cells [8]. In this case, a thyroid examination by a doctor and proper interpretation of clinical data is required to identify thyroid disease. However, the limitations of a doctor due to age and time constraints lead to a lack of interpretation of patient clinical data. Therefore, an analysis of thyroid disease classification is needed to simplify and speed up the process of diagnosing thyroid disease.

In this research, the analysis of thyroid disease has been carried out previously by [9] with the title "Comparison of the Naive Bayes Data Mining Algorithm and Bayes Network To Identify Thyroid Disease" in identifying thyroid disease is to assess the performance of the two algorithms, apply the Cross-Validation testing technique and Split Percentage, and with Confusion matrix to measure the value of both methods accuracy. The Bayes Network method has greater accuracy with a value of 98.49%, than the Naive Bayes method with a value of 91.80%.

While in the second study studied by [10] with the title "Implementation of the J48 Algorithm for Thyroid Disease Detection" conducted a study using the J48 algorithm in predicting thyroid disease with 7,200 data records yielding an accuracy value of 87 percent. It is necessary to develop a method for classifying thyroid disease. Therefore researchers are interested in conducting a study entitled "Analysis of Thyroid Disease Classification Using Xgboost Multiclass."

2. METHOD

2.1 Method

This study classifies thyroid disease using the Extreme Gradient Boosting Multiclass method [11]. XGBoost is an enhanced technique based on gradient enhancement decision trees that can create enhanced trees quickly and work in parallel [12].

Using the previous "weaker" classifier as a foundation, the ensemble technique aims to produce a robust classifier. The following predictors correct previous model errors as the models are added on top of each other iteratively, and this process continues until the model predicts or replicates the training data correctly. In short, the researcher used gradient descent to update the model [13].

The XGBoost implementation provides several advanced features for model customization, processing environment, and algorithm enhancement. It is capable of performing three main types of gradient enhancement (Gradient Enhancement (GB), GB Stochastic, and GB Regular) and is robust enough to allow fine-tuning and addition of regularization parameters [14].

In the regression tree, the inner node indicates the value for the attribute test, while the leaf node with the score reflects the rating [15]. The prediction

result is the number of scores anticipated by the K tree, as illustrated in the equation:

$$obj(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k}^{K} \Omega(f_k)$$

Where $\sum_{i=1}^{n} l(y_i, y$ is the differentiated loss function to measure whether the model is suitable for the training data set and is the item that determines the complexity of the model. As the complexity of the model increases, the corresponding score decreases

2.2 Research Flow

In conducting research on the classification of thyroid disease, the researcher made a research flow chart that aims to make the research run well, while the research flow chart can be seen in Figure 1:

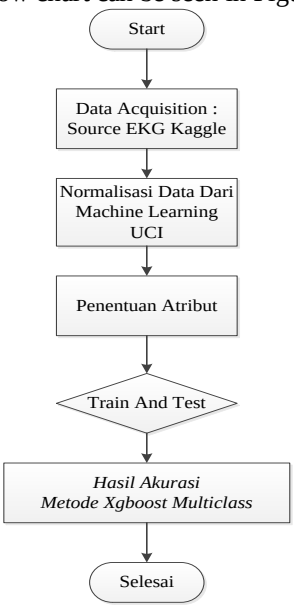

Figure 1. Flowchart Research

1. Data Acquisition

Data Acquisition is retrieving data from the UCL machine learning repository source and entering the data into Google Colab for processing thyroid disease classification data.

2. UCI Machine Learning Data Normalization

In this process, normalization is performed on the dataset taken from machine learning UCI with a total of 7200 instances.

3. Attribute Determination

At this stage, the attributes that will be used in the study of thyroid disease classification are determined.

4. Train And Test

At this stage, the processed data is tested to obtain accurate results in classifying thyroid disease.

5. Accuracy Results

After the classification process is complete, the last step is to get an accuracy value which will see how accurate the Xgboost Multiclass method is in classifying.

3. RESULTS AND DISCUSSION

3.1 Problem Analysis

The method analysis in this study used the XGBoost multigrade approach to identify patients with different thyroid-related disorders based on their age, sex, and medical information – including findings of thyroid hormone levels in the blood.

3.2 Data Analysis

Data were obtained from this UCI machine learning repository [17]. The repository includes various text files with different subsets of data. One of them has information for 9000 different individuals along with medical diagnoses from 20 potential classifications. Classification makes 7 different types of diagnosis such as negative diagnosis, a hyperthyroid condition, hypothyroid condition, protein binding, non-thyroid, undergoing replacement therapy, and discordant results:

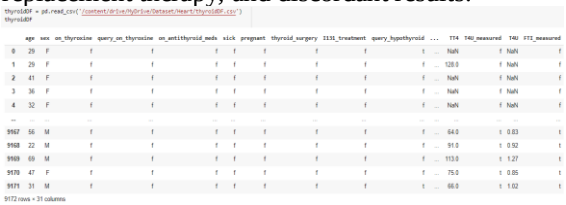

Figure 2. Dataset

3.3 Pengolahan Data

In data processing, the first step is to import libraries and datasets into Google Colab, the second stage is to clean up data, such as replacing null values with median values, the third stage is data visualization, and the last stage is the distribution of testing data and test data to obtain data. Value accuracy classification of thyroid disease. The flow of data processing can be seen in Figure 3 below:

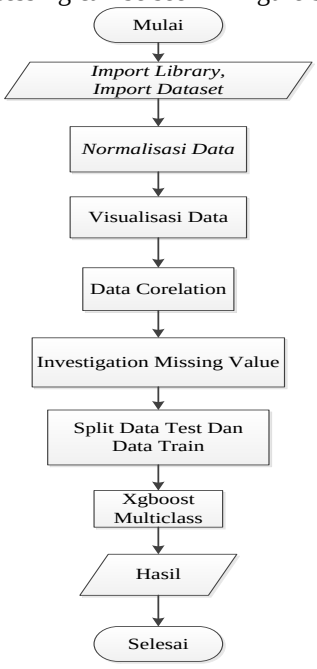

Figure 3. Data Processing Flowchart

3.3.1 Import Library

All Google Colab libraries are called at the import library stage, which will be used to process thyroid disease classification data. The library to be imported can be seen in Figure 4 below:

```
import numpy as np
import pandas as pd
import seaborn as sns #
import matplotlib.pyplot as plt
%matplotlib inline
!pip install dython
import dython
from dython.nominal import associations
from dython.nominal import identify_nominal_columns
import xgboost as xgb
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import confusion_matrix, plot_confusion_matrix, classification_report
from sklearn.metrics import balanced_accuracy_score, accuracy_score, precision_score, recall_score, f1_score
from sklearn.utils.class_weight import compute_sample_weight
```

Figure 4. Import Library

3.3.2 Import Dataset

At this stage, the process of importing the dataset that has been downloaded from the UCL machine learning repository into the Google Collaboratory is carried out. The dataset import process can be seen in Figure 5 below:

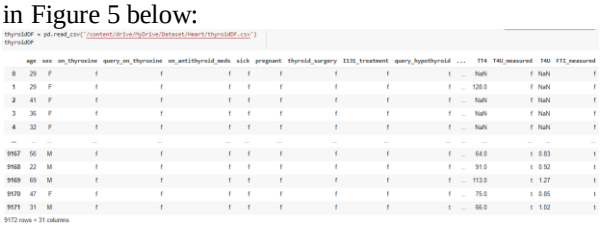

Figure 5. *Import* Dataset

The dataset contains several text files with different data types. One contains information for 9000 medical patient diagnoses consisting of 20 possible classes. These classes have 7 different types of diagnosis, such as Negative Diagnosis, Hyperthyroid Conditions, Binding Protein, Non-Thyroidal, Undergoing Replacement Therapy, and Discordant Result.

3.3.3 Data Normalization

The data normalization process or data cleaning is carried out at this stage. The data cleaning stages were removing 4 diagnostic targets, deleting redundant columns or repeatedly, and observing 100-year-old patients. The stages of normalization will be described in the following explanation:

1. Drop Diagnosis

Several inconclusive diagnoses were dropped because they accounted for less than 3% of the total data set. So the investigators have decided to maintain the observation for patients with a negative diagnosis, hyperthyroidism, or hypothyroidism.

```
# re-mapping target vaues to diagnostic groups
diagnoses = {'-': 'negative',
             'A': 'hyperthyroid',
             'B': 'hyperthyroid',
             'C': 'hyperthyroid',
             'D': 'hyperthyroid',
             'E': 'hypothyroid',
             'F': 'hypothyroid',
             'G': 'hypothyroid',
             'H': 'hypothyroid'}

thyroidDF['target'] = thyroidDF['target'].map(diagnoses) # re-mapping
# dropping observations with 'target' null after re-mapping
thyroidDF.dropna(subset=['target'], inplace=True)
```

Figure 6. *Drop* Diagnosis

2. Patient Age Observation

The first step is to observe the patient's age, and after the observation, it is continued by changing the data of patients with ages above 100 to be blank (null) because they have a negative diagnostic value. The patient's age to be processed is less than 60 years old.

```
# inspecting observations with age > 100
thyroidDF[thyroidDF.age > 100]
```

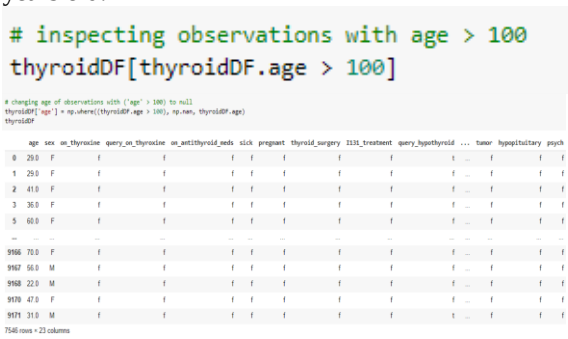

Figure 7. Age Observation And Change Age

3. Drop Column Redundancy

In the dataset, there is a lot of redundant data, or it repeatedly appears, which makes the data terrible, so data redundancy is removed in several columns, such as the TSH_Measured, T3_Measured, TT4_Measured, T4U_Measured, FTI_Measured, TBG_Measured, Patient_id columns.

```
# dropping redundant attributes from thyroidDF dataset
thyroidDF.drop(['TSH_measured', 'T3_measured', 'TT4_measured', 'T4U_measured',
                'FTI_measured', 'TBG_measured', 'patient_id', 'referral_source'],
               axis=1, inplace=True)
```

Figure 8. Drop Column Redundancy

3.3.4 Data Visualization

Researchers made Exploratory Data Analysis see the distribution of hormone levels in the blood for each target class of patients. This process is carried out to see how well the predictors of each of these attributes are.

```
#setting up grid for multiple seaborn plots
fig, axes = plt.subplots(3,2,figsize=(15,8))
fig.suptitle('Numerical Attributes vs. Target')
sns.set_style("whitegrid");

# TSH vs. 'target'
sns.stripplot(x=thyroidDF.target, y=thyroidDF.TSH, linewidth=0.6, jitter= 0.3, ax=axes[0, 0])
# T3 vs. 'target'
sns.stripplot(x=thyroidDF.target, y=thyroidDF.T3, linewidth=0.6, jitter= 0.3, ax=axes[0, 1])
# TT4 vs. 'target'
sns.stripplot(x=thyroidDF.target, y=thyroidDF.TT4, linewidth=0.6, jitter= 0.3, ax=axes[1, 0])
# T4U vs. 'target'
sns.stripplot(x=thyroidDF.target, y=thyroidDF.T4U, linewidth=0.6, jitter= 0.3, ax=axes[1, 1])
# FTI vs. 'target'
sns.stripplot(x=thyroidDF.target, y=thyroidDF.FTI, linewidth=0.6, jitter= 0.3, ax=axes[2, 0])
# TBG vs. 'target'
sns.stripplot(x=thyroidDF.target, y=thyroidDF.TBG, linewidth=0.6, jitter= 0.3, ax=axes[2, 1])
```

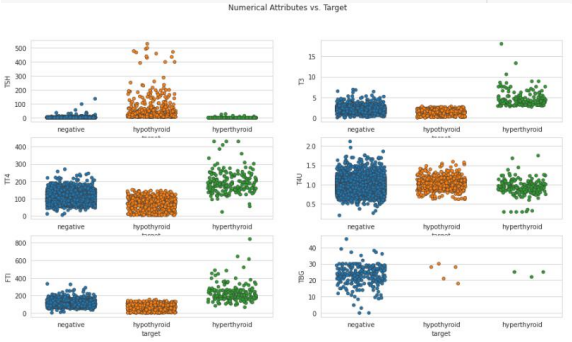

Figure 9. Plots Numerical Atrributes Vs Target

Figure 9 shows that FTI, T3, and TT4 will be excellent addition to the research model. TSH is also good but needs to handle outliers for the 'target' hypo and further analyze attribute distributions

before generating results. This is all in line with the knowledge found about Hormone level tests.

The next step is to make pair plots of numeric variables and see if they can find clusters that form between variables. PairPlots can be seen in Figure 10 below.:

Figure 10 Pair Plots Numerical Vs Target

On the diagonal of the pair plot, we can see the distribution of each numeric variable concerning each other. It is apparent how unbalanced the dataset is, with so many hostile 'targets' compared to hypothyroid or hyperthyroid. To handle the imbalance of the target class, it is necessary to resample the data protocol using another model by applying the Xgboost Multiclass method.

3.3.5 Data Correlation

In looking at the correlation data, the researcher uses the Dython Library as a tool for exploring multi-variate correlations. This is a great way to explore the relationships between variables of different types in a data set. The correlation data will help researchers get an idea of the correlation of variables with Exploratory Data Analysis (EDA).

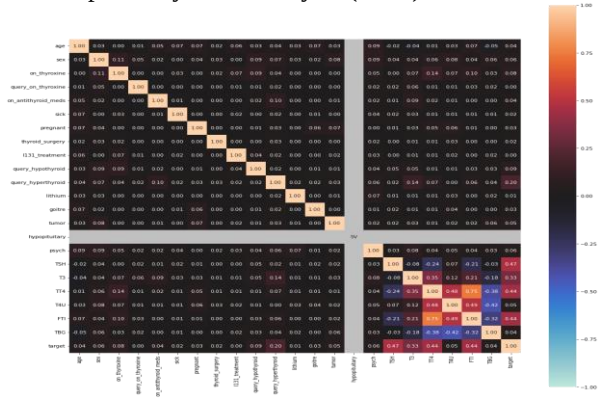

Figure 11 Data Correlation

From the EDA data correlation in Figure 11, it is concluded that the Hormone test is the most helpful in predicting the target diagnosis.

3.3.6 Investigation Missing Value

First, the researcher did some calculations to determine the severity of the missing value data problem. The Source code function in Figure 12 below takes a data frame as input and stores a calculation of the missing values per column, then calculates the percentage of missing values in that column and summarizes the information in an easy-to-view output data frame.

```
# calculating missingess of entire DF
missingness = thyroidDF.isnull().sum().sum() / thyroidDF.count().sum()
print('Overall Missingness of thyroidDF is: {:.2f}%'.format(missingness * 100))

# Create table for missing data analysis
def missing_table(df):
    total = df.isnull().sum().sort_values(ascending=False)
    percent = (df.isnull().sum()/df.isnull().count()).sort_values(ascending=False)
    missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
    return missing_data

# Analyze missing data
missing_table(thyroidDF).head(10)
```

```
Overall Missingness of thyroidDF is: 7.54%
```

	Total	Percent
TBG	7287	0.965677
T3	2209	0.292738
TSH	722	0.095680
T4U	676	0.089584
FTI	669	0.088656
TT4	354	0.046912
sex	250	0.033130
age	4	0.000530
goitre	0	0.000000
psych	0	0.000000

Figure 12. Investigation Missing Value

3.3.7 Split Data Train and Test

At this stage, the distribution of the dataset of 80:20 is carried out where 80% for test data and 20 for training data with a random state standard google colab of 42. For the data sharing process, see Figure 13 below:

```
import xgboost as xgb
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import confusion_matrix, plot_confusion_matrix, classification_report
from sklearn.metrics import balanced_accuracy_score, accuracy_score, precision_score, recall_score, f1_score
from sklearn.utils.class_weight import compute_sample_weight

# replacing boolean strings with binary 0 and 1
thyroidDF.replace('f', 0, inplace=True)
thyroidDF.replace('t', 1, inplace=True)

# replacing sex with binary 0 and 1
thyroidDF.replace('M', 0, inplace=True) # male mapped to 0
thyroidDF.replace('F', 1, inplace=True) # female mapped to 1

xgbDF = thyroidDF.replace(np.nan, 0)

# re-mapping target vaues to diagnostic groups
# *** I get different final results by ordering these classes differently ***
diagnoses = {'negative': 0,
             'hypothyroid': 1,
             'hyperthyroid': 2}

xgbDF['target'] = xgbDF['target'].map(diagnoses) # re-mapping

# train and test split --> stratified
X = xgbDF.drop('target', axis=1).copy()
y = xgbDF['target'].copy()

X_train, X_test, y_train, y_test = train_test_split(X, y , random_state=42, stratify=y)
```

Figure 13 Spilting Data

3.3.8 Xgboost Multiclass

1. Xgboost Class 1

For Xgboost Class 1, using default hyperparameters settings such as num_class = 3, missing= 1, erly_stopping_rounds = 10, eval_metric, seed = 42, we get 99% accuracy and 94% balanced accuracy.

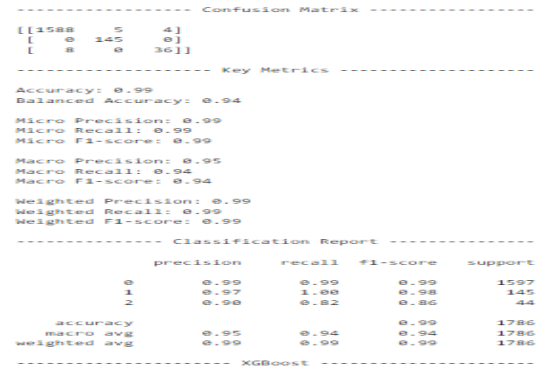

Figure 14 Xgboost Class 1

2. Xgboost Class 2

For Xgboost Class 2 use optimized hyperparameters settings like num_class = 3, missing= 1, gamma = 0, learning_rate = 0.1, max_depth = 3, reg_lambda = 1, subsample = 1, colsample_bytree = 1, erly_stopping_rounds = 10, eval_metric, seed = 42 get the same accuracy as multiclass 1 of 99% and balanced accuracy of 94%.

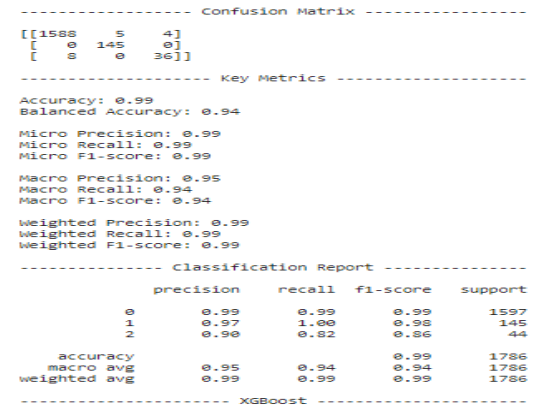

Figure 15 Xgboost Class 2

3. Xgboost Class 3

For Xgboost Class 3, use optimized hyperparameters settings such as num_class = 3, missing= 1, gamma = 0, learning_rate = 0.1, max_depth = 5, reg_lambda = 1, erly_stopping_rounds = 10, eval_metric, seed = 42 to get the same accuracy as multiclass 1 of 99% and balanced accuracy of 97%.

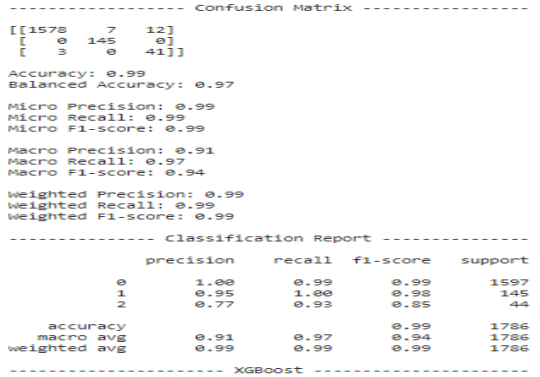

Figure 16 Xgboost Class 3

4. CONCLUSION

In analyzing the classification of thyroid disease predictions using the multiclass algorithm, it was found that the xgboost multiclass version 3 algorithm was the best in classifying with an accuracy score of 99% and balanced accuracy of 97%, followed by the xgboost method versions 1 and 2 with 99% accuracy and balanced accuracy of 94%.

Researchers hope that in the future, the development of methods to produce more accurate results of thyroid disease calcification will be carried out. Moreover, it is also hoped to add thyroid disease predictive analysis using more other attributes.

BIBLIOGRAPHY

[1] "Klasifikasi Nodul Tiroid Pada Citra Ultrasonografi Berdasarkan Analisis Karakteristik Tekstur Menggunakan Multilayer Perceptron Classifier." http://etd.repository.ugm.ac.id/penelitian/detail/156791 (accessed Jun. 24, 2022).

[2] M. T. Nugraha, Y. W. Finansah, N. Primadina, and N. Yuliyanasari, "FNAB and Anatomic Pathology Biopsy Accuracy in Thyroid Nodule Diagnosis," MAGNA MEDICA Berk. Ilm. Kedokt. dan Kesehat., vol. 9, no. 1, p. 10, 2022, doi: 10.26714/magnamed.9.1.2022.10-16.

[3] "Hubungan Kebiasaan Merokok Dan Lama Paparan Radiasi Terhadap Volume Tiroid Dan Morfologi Nodul Tiroid Berdasarkan Ultrasonografi Pada Radiografer Malang Raya - Brawijaya Knowledge Garden." http://repository.ub.ac.id/id/eprint/184455/ (accessed Jun. 24, 2022).

[4] "Analisis Regresi Logistik Ordinal Pada Faktor-Faktor Yang Mempengaruhi Stadium Kanker Tiroid - ITS Repository." https://repository.its.ac.id/81028/ (accessed Jun. 24, 2022).

[5] C. Nas, "Sistem Pakar Diagnosa Penyakt Tiroid Menggunakan Metode Dempster Shafer," J. Teknol. Dan Open Source, vol. 2, no. 1, pp. 1–14, 2019, doi: 10.36378/jtos.v2i1.114.

[6] I. D. Antika, R. Hanriko, and T. . Larasati, "Studi Diagnostik Ultrasonografi dalam Mendiagnosis Nodul Tiroid di RSUD Dr. H. Abdul Moeloek Bandar Lampung," Medula, vol. 8, no. 2, pp. 40–46, 2019.

[7] M. Hidayat, D. Milvita, and F. Nazir, "Analisis Uptake Tiroid Menggunakan Teknik Roi (Region Of Interest) Pada

Pasien Nodul Tiroid," J. Fis. Unand, vol. 3, no. 1, pp. 59–64, 2014.

[8] A. Putra ZM, Ernawati, and A. Erlansari, "Sistem Pakar Diagnosa Penyakit Tiroid Menggunakan Metode Naive Bayes Berbasis Android," J. Rekursif, vol. 5, no. 3, pp. 270–284, 2017.

[9] B. Wijonarko, "Perbandingan Algoritma Data Mining Naive Bayes Dan Bayes Network Untuk Mengidentifikasi Penyakit Tiroid," J. PILAR Nusa Mandiri, vol. 14, no. 1, pp. 21–26, 2018, [Online]. Available: http://www.bsi.ac.id.

[10] S. Agustiani, A. Mustopa, A. Saryoko, W. Gata, and S. K. Wildah, "Penerapan Algoritma J48 Untuk Deteksi Penyakit Tiroid," Paradig. - J. Komput. dan Inform., vol. 22, no. 2, pp. 153–160, 2020, doi: 10.31294/p.v22i2.8174.

[11] A. Danandeh Mehr, "Drought classification using gradient boosting decision tree," Acta Geophys., vol. 69, no. 3, pp. 909–918, 2021, doi: 10.1007/s11600-021-00584-8.

[12] W. F. Mustika, H. Murfi, and Y. Widyaningsih, "Analysis Accuracy of XGBoost Model for Multiclass Classification - A Case Study of Applicant Level Risk Prediction for Life Insurance," Proceeding - 2019 5th Int. Conf. Sci. Inf. Technol. Embrac. Ind. 4.0 Towar. Innov. Cyber Phys. Syst. ICSITech 2019, pp. 71–77, Oct. 2019, doi: 10.1109/ICSITECH46713.2019.8987474.

[13] T. Chen and C. Guestrin, "XGBoost : A Scalable Tree Boosting System," pp. 785–794, 2016.

[14] "XGBoost for Multi-class Classification | by Ernest Ng | Towards Data Science." https://towardsdatascience.com/xgboost-for-multi-class-classification-799d96bcd368 (accessed Jun. 25, 2022).

[15] I. Muslim and K. Karo, "Implementasi Metode XGBoost dan Feature Important untuk Klasifikasi pada Kebakaran Hutan dan Lahan," J. Softw. Eng. Inf. Commun. Technol., vol. 1, no. 1, pp. 10–16, 2020, [Online]. Available: https://ejournal.upi.edu/index.php/SEICT/article/view/29347.

[16] J. Infokum, "Classification Of Electrocardiogram (Ecg) Waves Of Heart Disease Using The Xgboost Metode Method," vol. 10, no. 2, pp. 891–904, 2022.

[17] "UCI Machine Learning Repository: Thyroid Disease Data Set." https://archive.ics.uci.edu/ml/datasets/thyroid+disease (accessed Jun. 24, 2022).