# Penerapan Algoritma *Gradient Boosting* dalam Mendiagnosa Penyakit Kucing dan Anjing

**Vincent[*1], Nur Rachmat[2]**
Program Studi Informatika, Fakultas Ilmu Komputer dan Rekayasa, Universitas Multi Data Palembang
Jl. Rajawali No.14, 9 Ilir, Kec. Ilir Tim. II, Kota Palembang, Sumatera Selatan 30113
Email: [1]vincent.cent@mhs.mdp.ac.id, [2]nur.rachmat@mdp.ac.id

***Abstract. Application of the Gradient Boosting Algorithm in Diagnosing Diseases in Cats and Dogs.*** *Royal Canin, as a domestic animal research institute, revealed that pets in Indonesia rarely undergo routine check-ups at veterinary clinics, with only 29.5% receiving such care. With this percentage, there is growing concern that animals may transmit diseases to humans, known as zoonoses, if they do not receive proper care and early diagnosis of any illnesses they may be experiencing. Therefore, to address the issue of zoonoses. In this study, the gradient boosting method was used as the primary focus for predicting diseases based on the symptoms experienced by pets. Through the hyperparameter tuning process using GridSearch, the best model was obtained with the following parameter combination: learning_rate = 0.05, max_depth = 7, min_samples_leaf = 1, min_samples_split = 2, n_estimators = 200, and subsample = 0.9. The model demonstrated the best performance, with an accuracy of 88%, precision of 97%, recall of 96%, F1-score of 96%, and Hamming loss of 0.29%.*
***Keywords:*** *diagnosis diseases, gradient boosting, pet, zoonotic*

***Abstrak.*** *Royal Canin selaku lembaga riset hewan domestik mengungkapkan bahwa hewan peliharaan di Indonesia jarang sekali melakukan pemeriksaan rutin ke klinik hewan, jika dipersentasekan hanya berada di angka 29,5%. Dengan persentase tersebut, semakin khawatir hewan dapat menularkan penyakit ke manusia atau disebut sebagai zoonosis, jika hewan sama sekali tidak mendapatkan perawatan dan identifikasi dini penyakit yang dialami. Pada penelitian ini menggunakan metode gradient boosting sebagai fokus utama untuk memprediksi penyakit berdasarkan gejala-gejala yang dialami hewan peliharaan. Melalui proses hyperparameter tuning menggunakan gridsearch, diperoleh model terbaik dengan kombinasi parameter: learning_rate 0,05, max_depth 7, min_samples_leaf 1, min_samples_split 2, n_estimators 200, dan subsample 0,9. Dari hasil hyperparameter tuning, model tersebut menunjukkan performa terbaik dengan accuracy 88%, precision 97%, recall 96%, f1-score 96%, dan hamming loss 0,29%. Hasil tersebut menunjukkan bahwa model memiliki kemampuan memprediksi multi-label yang akurat.*
***Kata Kunci:*** *diagnosa penyakit, gradient boosting, hewan, zoonosis*

## 1. Pendahuluan

Kucing dan anjing domestik merupakan hewan peliharaan yang sering kali dijumpai di manapun. Banyak masyarakat mulai mengadopsi kucing dan anjing baik ras liar maupun domestik. Masyarakat sekarang lebih sering memanggil mereka dengan sebutan anabul yang merupakan singkatan dari anak bulu, singkatan ini mewakili mereka yang sangat lucu dan berbulu lebat.

Freedman dan Wayne [1] mendefinisikan anjing domestik atau sebutan ilmiah *Canis lupus familiaris* sebagai hewan karnivora besar yang merupakan hewan pertama dalam spesiesnya yang telah didomestikkan, sedangkan kucing dengan nama ilmiah *Felis silvestris catus* sejenis hewan mamalia karnivora dengan ukuran badan kecil dengan kemampuan komunikasi dengan manusia yang sangat baik melalui vokal, bahasa tubuh, serta dari tatapan mata mereka. Data

survei dari *Rakuten Insight Center* berdasarkan GoodStats [2] menyatakan bahwa masyarakat Indonesia cenderung memilih kucing sebagai hewan peliharaan dengan persentase 65,2%, sedangkan anjing hanya berada di angka 5,3%.

Dengan tingkat kepemilikan hewan peliharaan yang tinggi, jumlah dokter hewan di Indonesia terbilang masih sangat minim atau kurang. Mengutip dari pernyataan [3] mengungkapkan bahwa dokter hewan di Indonesia hanya tercatat sekitar 15 ribu sementara Indonesia sendiri memerlukan sekitar 70 ribu dokter hewan. Akibatnya, pemilik hewan seringkali mengalami kesulitan dalam membawa anabul mereka ke dokter hewan, terutama ketika terjadi situasi darurat atau ketika membutuhkan pemeriksaan rutin.

Hal ini didukung oleh survei yang dilakukan oleh lembaga riset Royal Canin [4] yang mengungkapkan bahwa hewan peliharaan di Indonesia jarang sekali melakukan pemeriksaan rutin ke klinik hewan, jika dipersentasekan hanya berada di angka 29,5%. Dengan persentase tersebut, semakin khawatir hewan dapat menularkan penyakit ke manusia atau disebut sebagai zoonosis, jika hewan tidak mendapatkan perawatan serta identifikasi dini penyakit yang dialami hewan. Zoonosis merupakan sebuah istilah untuk penyakit yang menular serta ditularkan oleh hewan kepada manusia [4], sedikitnya ada 28 jenis virus dan bakteri hingga parasit yang dapat menyerang manusia kapan saja.

Pada penelitian ini menggunakan metode *machine learning* yakni algoritma *gradient boosting* sebagai fokus utama untuk memprediksi penyakit berdasarkan gejala-gejala yang dialami hewan peliharaan. Dengan menggunakan *library MultiLabelBinarizer* serta pendekatan strategi *multi-class/multi-label* menggunakan *OneVsRestClassifier* (OvR) untuk memperoleh *output* berupa *multi-label.*

## 2. Tinjauan Pustaka

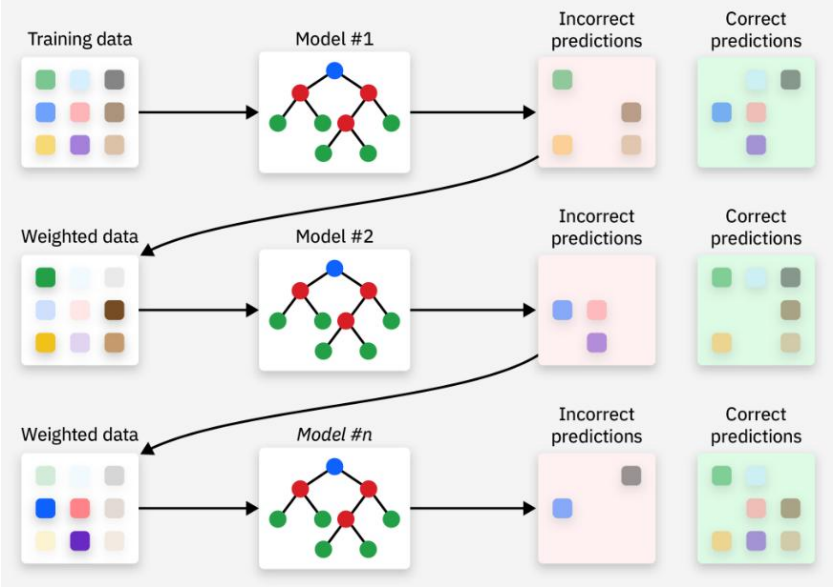
*Gradient boosting* merupakan teknik *machine learning* yang membangun model prediktif secara bertahap (*iterative*) dari kumpulan model yang lemah seperti *decision tree*. Secara konseptual, *gradient boosting* akan mempelajari model baru dengan pembobotan data serta dari kesalahan residual atau disebut *loss function* dari model lemah sebelumnya yang telah dilatih [5]. Penelitian ini menggunakan *gradient boosting* dengan *base learning decision tree* yang merupakan algoritma *machine learning* cukup populer karena efisiensi, akurasi, dan kemampuan interpretasinya. [6].

*Gradient boosting* akan membangun model secara regresi aditif dengan secara berurutan menyesuaikan pengklasifikasian yang lemah dengan residual saat ini. Oleh karena itu, pengklasifikasi lemah yang baru dilatih akan mengoreksi kesalahan penilaian komponen lemah sebelumnya dan meningkatkan kinerja serta efisiensi prediksi. Model terakhir yang disebut sebagai *strong learner* merupakan penggabungan setiap hasil dari *weak learner* sehingga akhirnya disebut sebagai *ensemble* [7]. Gambar 1 menvisualisasikan arsitektur dari *gradient boosting decision tree*.

Untuk mengoptimalkan hasil prediktif pada *gradient boosting decision tree* perlu adanya *hyperparameters tuning* untuk mengatur parameter pada model *gradient boosting decision tree* [8],[9],[10] antara lain *learning_rate* merupakan parameter yang paling penting dan berpengaruh dalam model *gradient boosting* biasanya nilai *learning_rate* semakin menuju nol akan mengurangi jumlah kontribusi yang diberikan oleh setiap *weak learner*. *Max_depth* mengatur seberapa dalam pohon yang akan digunakan (biasanya *max_depth* tiga untuk menghindari *overfitting*). Semakin dalam pohon semakin membebani dalam segi komputasi. *Subsample* mengontrol proporsi data untuk pelatihan setiap n pohon*, n_estimators* adalah jumlah tahapan *boosting* yang harus dilakukan, *min_samples_leaf* diperuntukkan jumlah minimum sampel yang diperlukan untuk sebuah simpul daun*, max_features* menentukan jumlah fitur yang akan dipertimbangkan untuk mencari pemisahan (*split*) terbaik di setiap *node* dan *min_samples_split* mengatur jumlah minimum dari sampel yang diperlukan untuk membagi simpul pada *internal node*.

Membandingkan dengan algoritma *ensemble learning* yakni *random forest* terkait penelitian [11] berfokus pada menganalisis perilaku pro lingkungan mahasiswa secara prediktif

menggunakan model decision tree C.50 yang telah dimodifikasi sesuai dengan data penelitian. Data yang digunakan berasal dari 334 mahasiswa yang berada di Provinsi Guangdong, Tiongkok. Faktor utama yang dianalisis adalah *willingness to behave in an environmentally responsible manner*, *innovative behaviour*, dan *perceived behavioural control*. Hasil penelitian menunjukkan bahwa *willingness to act responsibly* merupakan prediktor terkuat di antara tiga variabel yang disebutkan, diikuti oleh environmental *activism* dan *subjective norms*. Model *decision tree* menghasilkan akurasi prediksi sebesar 72,89% untuk mengidentifikasi perilaku pro lingkungan. Studi ini membantu memahami hubungan antara *machine learning* dan perilaku lingkungan serta menekankan pentingnya menggalang kegiatan yang berfokus pada keberlanjutan di kalangan mahasiswa.




**Gambar 1. Arsitektur *Gradient Boosting Decision Tree* [12]**

Penelitian [13] meramal kecepatan angin dengan data cuaca numerik dengan model *Gradient Boosting Decision Tree* (GBDT). Model ini diterapkan untuk memperbaiki hasil dari model ramalan cuaca numerik *Weather Research and Forecasting* (WRF). Dengan menggunakan sekitar 300 fitur dari berbagai lapisan ketinggian dan tekanan. Model mampu mengurangi *Root Mean Square Error* (RMSE) kecepatan angin dari 2,7—3,5 m/s pada hasil WRF menjadi 1-1,5 m/s. Selain itu, GBDT juga menunjukkan peningkatan signifikan dalam *Index of Agreement* (IA) sebesar 0,10-0,20, *Corelation Coefficient* (R) sebesar 0.10-0,18, dan *Nash-Sutcliffe Efficiency Coefficient* (NSE) yang bervariasi antara -0,06 hingga 0,6.

Penelitian [14] berfokus pada klasifikasi *multi-class gradient boosting* untuk mengklasifikasi jenis hewan pada kebun binatang. Penelitian ini menggunakan data *public* dari platform Kaggle yang berisi kumpulan hewan yang ada di kebun binatang sebanyak 130 data hewan ditambah dengan data hewan yang dimiliki oleh penulis. Hasil dari penelitian ini menggunakan kombinasi parameter *n_estimators* sebanyak 50, *max_depth* tiga, *sub_sample* sebanyak 1,0, *learning_rate* sebesar 0,1 menghasilkan nilai *accuracy* sebesar 93,75%, *recall* 94%, *precision* sebesar 96%, dan MSE sebesar 12,5%.

Penelitian [15] berfokus pada studi regresi obat-obatan kadaluarsa sebagai material antikorosi dengan menggunakan data sebanyak 250 senyawa obat kadaluarsa yang dipublikasikan dengan 10 deskriptor molekuler sebagai fitur *input*. Deskriptor tersebut meliputi variabel seperti *molecular weight* (MW), *constant dissociation* asam (pKa), *log* P (koefisien partisi oktanol-air), *log* S (kelarutan air), *polar surface area* (PSA), *polarizability* (α), energi orbital molekuler tertinggi yang terisi (E-HOMO), energi orbital terisi terendah yang tidak terisi (E-LUMO), *electrophilicity* (ω), dan *fraction electron shared* (ΔN). Data ini digunakan untuk memprediksi

efisiensi inhibisi korosi (CIE) dari senyawa obat kadaluarsa. Penelitian ini menggabungkan metode *gradient boosting* dengan fungsi *polynomial* untuk meningkatkan akurasi dari *gradient boosting* sehingga hasil dari penelitian ini memberikan nilai $R^2$ pada data *training* meningkat dari 0,9988 menjadi 0,9998 dan di data *testing* dari 0,9998 menjadi 0,9999, serta nilai RMSE menurun dari 0,008 menjadi 0,002 di data *training,* dan dari 0,002 menjadi 0,0003 di data *testing*.

## 3. Metodologi

### 3.1 Data

Data yang digunakan untuk penelitian ini berasal dari rekam medis setiap pasien pada klinik Zero Animal Clinic yang telah berkerjasama dalam menyusun penelitian ini. Data yang diberikan memuat berbagai atribut, yakni spesies, umur, *indoor/outdoor*, gejala, dan diagnosa awal/Ddx. Namun spesies, umur, dan *indoor/outdoor* tidak akan digunakan dalam pengembangan model dikarenakan atribut ini merupakan variabel lepas yang tidak mempengaruhi hasil prediktif sehingga menyisakan atribut gejala dan diagnosa awal/Ddx.

Untuk mempermudah dalam pengembangan model, atribut diagnosa awal/Ddx akan dilakukan perubahan nama menjadi diagnosa agar tidak terjadi konflik dengan *space*. Pada Tabel 1 menyajikan contoh data dari *dataset* Zero Animal Clinic yang digunakan dalam proses pengembangan model.

**Tabel 1. Sampel *Dataset* Riwayat Pasien**

| Spesies | Umur | *Indoor/ Outdoor* | Gejala | Diagnosa Awal / Ddx |
|---|---|---|---|---|
| Kucing | 3 tahun | *Outdoor* | Flu, bersin | *Rhinotracheitis*, FIV, FeLV, FIP, *bronchitis* |
| Kucing | 17 tahun | *Indoor* | Nafsu makan menurun, enggan minum | AKI, PKD, hepatitis, CKD, malnutrisi |
| Kucing | 3 tahun | *Outdoor* | Luka, bersin, batuk | Vulnus, bronchitis, *rhinotracheitis* |
| Kucing | 2 tahun | *Outdoor* | Sakit mata, flu | *Rhinotracheitis*, uvetis, FIV, FeLV, FIP |
| Kucing | 3,5 tahun | *Indoor* | Buang air besar membek, iritasi anus, berat badan menurun | IBD, IVDD, bloat, malnutrisi |
| Anjing | 7 bulan | *Indoor* | Kecelakaan | Radius ulna fexter |

Pertama-tama, dilakukan *pre-processing dataset* untuk mengubah data menjadi *multi label* dalam bentuk biner dengan *library MultiLabelBinarizer* pada setiap gejala menjadi kolom individual dengan nilai satu jika benar, nilai nol jika salah, dan diagnosa awal/Ddx dilakukan hal yang sama yakni diubah dalam bentuk *MultiLabelBinarizer* dilampirkan pada Kode 1.

**Kode 1. *Pre-processing Dataset***

```
df = pd.read_csv('data/DatasetZAC1.csv', delimiter = ';')
df['Gejala'] = df['Gejala'].apply(lambda x: ','.join(x) if isinstance(x,
list) else x)
df['Gejala'] = df['Gejala'].apply(lambda x: [gejala.strip() for gejala in
x.split(',') if gejala.strip()])
mlb = MultiLabelBinarizer()
gejala_encoded = pd.DataFrame(mlb.fit_transform(df['Gejala']),
columns=mlb.classes_)
if '' in gejala_encoded.columns:
    gejala_encoded = gejala_encoded.drop(columns=[''])
data_features = pd.concat([gejala_encoded], axis=1)
data_target = df['Diagnosa Awal / Ddx']
data_final = data_features.copy()
data_final['Diagnosa'] = data_target
data_final.info()
```

Kemudian *dataset* dibagi menjadi tiga bagian, yakni 80% data latih, 10% data validasi, dan 10% data uji seperti pada Kode 2. Dari hasil pembagian, data latih memiliki data sebanyak 921 data, 115 data untuk data validasi, dan 116 data uji seperti pada Gambar 2.

**Kode 2. Pembagian *Dataset***

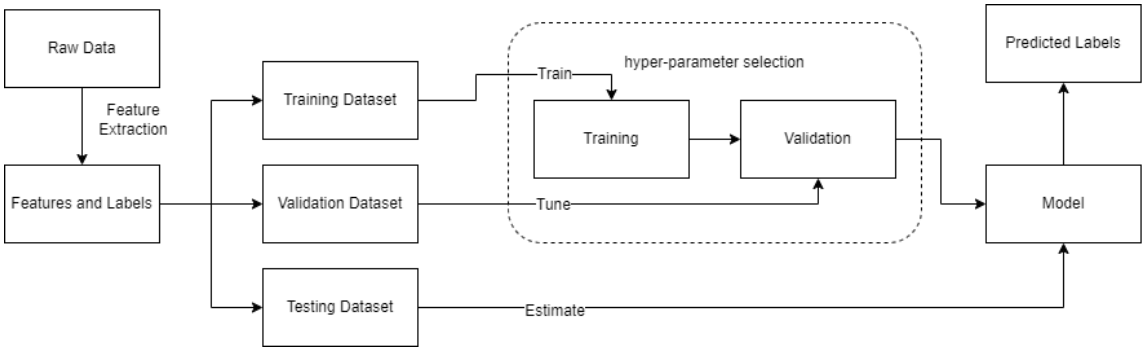
```
X = data_features
y = data_final['Diagnosa'].apply(lambda x: x.split(', '))
mlb = MultiLabelBinarizer()
y_encoded = mlb.fit_transform(y)
X_train, X_temp, y_train, y_temp = train_test_split(X, y_encoded,
test_size=0.2, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp,
test_size=0.5, random_state=42)
print("Shape of data          :", X.shape)
print("Shape of train data    :", X_train.shape)
print("Shape of validation data:", X_val.shape)
print("Shape of test data     :", X_test.shape)
```

```
Shape of data          : (1152, 94)
Shape of train data    : (921, 94)
Shape of validation data: (115, 94)
Shape of test data     : (116, 94)
```

**Gambar 2. Distribusi Data Latih, Validasi, dan Uji**

## 3.2 Pengembangan Model

*Gradient boosting* akan dipilih sebagai metode untuk menyelesaikan persoalan dari penelitian ini. Model *gradient boosting* menggunakan teknik *boosting* yakni menghilangkan bias yang sangat efektif untuk model pembelajaran lemah yang memiliki varian rendah namun bias yang besar [14]. Gambar 3 menjelaskan tahapan dari pengembangan model.




**Gambar 3. Tahapan Pengembangan Model *Gradient Boosting***

Setelah pembagian *dataset*, model akan dilatih menggunakan 921 data. Pelatihan model *gradient boosting* akan dilatih tanpa adanya proses *hyperparameter tuning*, dan model akan dibungkus dengan pendekatan *OneVsRestClassifier* (OvR) sebagai strategi untuk menangani skenario *multi-label/multi-class* dikarena *gradient boosting* hanya dapat menerima data yang bersifat *single-label*, dan *library Pipeline*. Kode 3 untuk pelatihan model *gradient boosting*.

**Kode 3. Pelatihan Model *Gradient Boosting* dengan OvR dan *Pipeline***

```
from sklearn.pipeline import Pipeline
from sklearn.multiclass import OneVsRestClassifier
from sklearn.metrics import classification_report
pipeline = Pipeline(
    [("model", OneVsRestClassifier( GradientBoostingClassifier(
random_state=42,)),)])
pipeline.fit(X_train, y_train)
```

Selama proses pelatihan model akan dicari parameter terbaik dengan menggunakan *hyperparameter tuning* dikarenakan *gradient boosting* sangat diutamakan metode ini untuk meningkatkan performa model itu sendiri. Proses *hyperparameter tuning* ini akan mencari

kombinasi terbaik dari parameter-parameter seperti *learning_rate*, *max_depth, subsample, n_estimators*, *min_samples_leaf, max_features*, dan *min_samples_split*. Setelah menemukan parameter terbaik, model akan dievaluasi menggunakan data validasi, dan membandingkan kedua model tanpa *hyperparameter tuning* dan *hyperparameter tuning* dengan menggunakan metrik *accuracy, precision, recall, F1-score,* dan *hamming loss* untuk mengukur seberapa baik model melakukan prediksi. Tabel 2 menunjukkan parameter yang akan digunakan dalam pencarian kombinasi pada proses *hyperparameter tuning,* dan Kode 4 merupakan kode untuk proses *hyperparameter tuning*.

**Tabel 2. Parameter Pengujian *Hyperparameter Tuning Gradient Boosting***

| Parameter | *Parameter Value* |
|---|---|
| *Learning_rate* | 0,01; 0,1; 0,05 |
| *Max_depth* | 3, 5, 7 |
| *Min_samples_leaf* | 1, 2, 4 |
| *Min_samples_split* | 2, 5, 10 |
| *n_estimators* | 50, 100, 200 |
| *subsample* | 0,8; 0,9, 1,0 |

**Kode 4. Proses *Hyperparameter Tuning***

```
pipeline = Pipeline([("model", OneVsRestClassifier(
GradientBoostingClassifier(random_state=42)),)])
grid_search = GridSearchCV(estimator=pipeline, param_grid=param_grid,
scoring='accuracy', cv=5, n_jobs=-1, verbose=2)
grid_search.fit(X_train, y_train)
results_df = pd.DataFrame(grid_search.cv_results_)
results_df.to_csv("grid_search_results.csv", index=False)
```

Setelah pelatihan telah selesai, model akan dievaluasi dengan metrik-metrik validasi untuk mengukur seberapa jauh model menghasil performa yang terbaik dengan data luar atau *unseen data*. Metrik yang akan digunakan ialah *accuracy, precision, recall, F1-score,* dan *hamming loss*. Terkhususnya *hamming loss*, jika nilai yang dihasilkan menuju angka nol, maka model tersebut memiliki performa yang terbaik. Setiap metrik akan dinilai dengan metode penilaian rata-rata *weighted averaged. Weighted average* akan memberikan bobot yang lebih besar kepada label yang lebih sering muncul, sehingga metode ini dapat mengatasi ketidakseimbangan label (*label imbalance*) pada *dataset* model dievaluasi dengan Persamaan 1-5, dan Persamaan 6 merupakan metode rata-rata *weighted average*.

$$Accuracy, A = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \tag{1}$$

$$Precision, P = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Z_i|} \tag{2}$$

$$Recall, R = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i|} \tag{3}$$

$$F_1 = \frac{1}{n}\sum_{i=1}^{n}\frac{2|Y_i \cap Z_i|}{|Y_i| + |Z_i|} \tag{4}$$

$$HammingLoss, HL = \frac{1}{kn}\sum_{i=1}^{n}\sum_{l=1}^{n}[I(l \in Z_i \wedge l \notin Y_i) + I(l \notin Z_i \wedge l \in Y_i)] \tag{5}$$

$$Weighted\ Average = \frac{\sum_{l=1}^{L} Support_l \times metric_l}{\sum_{l=1}^{L} Support_l} \tag{6}$$

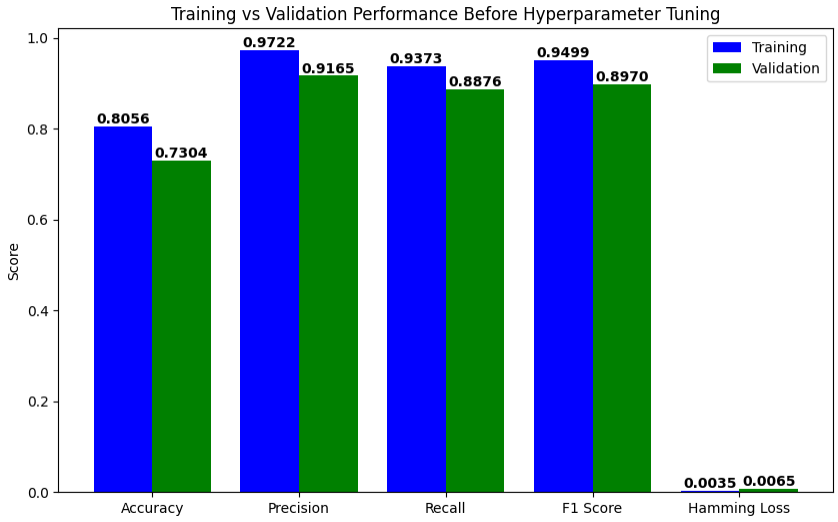
## 4. Hasil dan Diskusi

Gambar 4 menunjukkan hasil evaluasi berdasarkan data latih dan data validasi. Histogram biru adalah hasil dari data latih dan hijau merupakan data validasi. *Accuracy* dihasilkan oleh data latih menyentuh diangka 0,8056 namun setelah dibandingkan dengan data validasi hanya memperoleh angka sebesar 0,7304, *precision* pada data latih sebesar 0,9722, sedangkan 0,9165 untuk data validasi, *recall* pada data latih 0,9373 dan 0,8876, *f1 score* pada data latih sebesar 0,9499 dan 0,8970 untuk data validasi, dan terakhir *hamming loss* untuk data latih hampir mendekati nol dengan angka 0,0035 dan data validasi sedikit menjauh dengan angka 0,0065. Berdasarkan hasil evaluasi, model sudah baik dalam segi performa namun untuk akurasi masih rendah dibandingkan dengan metrik lainnya.

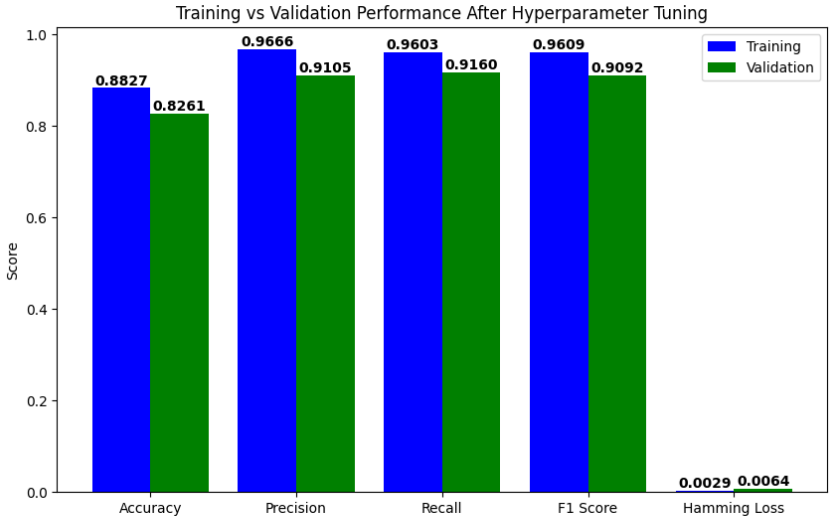
*Gradient boosting* sangat bergantung dengan *tuning* untuk meningkatkan performa dari model. Tabel 3 menunjukkan hasil dari *hyperparameter tuning* dari parameter-parameter yang telah dirinci pada Tabel 2. Setelah memperoleh parameter terbaik, Gambar 5 menunjukkan model yang sudah dilakukan proses *hyperparameter tuning* dengan nilai parameter-parameter *learning rate* 0,05, *max_depth* 7, *min_samples_leaf* 1, *min_samples_split* 2, *n_estimators 200*, dan *subsample 0,9* menghasilkan performa yang cukup lebih baik dibandingkan model tanpa *hyperparameter tuning*. Dari metrik *accuracy* terbilang mengalami peningkatkan yang cukup signifikan pada data latih dari 0,8056 menjadi 0,8827, dan data validasi dari 0,7304 ke 0,8261. Sementara itu untuk metrik lainnya tidak menunjukkan perubahan yang signifikan untuk *precision*, *recall* meningkat sedikit dengan *f1 score* serta *hamming loss.*

**Tabel 3. Hasil *Hyperparameter Tuning***

| *Learning Rate* | *Max Depth* | *Min Samples Leaf* | *Min Samples Split* | *N Estimator* | *Sub Sample* | *Rank* | *Accuracy* |
|---|---|---|---|---|---|---|---|
| 0,05 | 7 | 1 | 2 | 200 | 0.9 | 1 | 0,8077967097532316 |
| 0,05 | 7 | 1 | 2 | 200 | 0.8 | 2 | 0,8077967097532314 |
| 0,1 | 7 | 1 | 2 | 100 | 0.8 | 3 | 0,8056286721504111 |
| 0,05 | 7 | 1 | 10 | 200 | 0.9 | 4 | 0,8056227967097532 |
| 0,05 | 7 | 1 | 5 | 200 | 0.8 | 4 | 0,8056227967097532 |
| 0,05 | 5 | 1 | 2 | 200 | 0.9 | 4 | 0,8056227967097532 |
| 0,05 | 5 | 1 | 2 | 200 | 0.8 | 4 | 0,8056227967097532 |
| 0,1 | 5 | 1 | 2 | 100 | 0.8 | 8 | 0,8045417156286723 |
| 0,05 | 7 | 1 | 5 | 200 | 0.9 | 9 | 0,8045358401880142 |




**Gambar 4. Hasil Metrik Evaluasi Model Tanpa *Hyperparameter Tuning***




**Gambar 5. Hasil Metrik Evaluasi Model Setelah Proses *Hyperparameter Tuning***

Setelah perbandingan data latih dan data validasi untuk model tanpa *hyperparameter tuning* dan dengan *hyperparameter tuning*, tahapan berikutnya akan diuji menggunakan data uji dan membandingkan kedua model tersebut. Hasil dari pengujian dengan data latih model dengan *hyperparameter tuning* menunjukkan performa yang sangat tinggi dimana *accuracy* menyentuh di angka 0,8442, *precision* 0,9236, *recall* 0,9277, *F1-score* 0,9210, dan *hamming loss* sebesar 0,0058 sementara hasil model tanpa *hyperparameter tuning* hanya mampu memberikan *accuracy* 0,7672, *precision* 0,9388, *recall* 0,9059, *F1-score* 0,9133, dan *hamming loss* 0,0064. Secara perbandingan hasil metrik, model dengan *hyperparameter tuning* memberikan performa yang terbaik dibandingkan dengan model tanpa *hyperparameter tuning*.

Untuk menguatkan fakta bahwa model sudah dibatas optimalnya maka dilakukan interferensi dengan data *dummy* dari *y_test* mencari top 10 prediksi dengan *threshold* di atas 70% dilihat pada Kode 5. Gambar 6 merupakan hasil *output* dari Kode 5, namun mengenai penyakit tentunya perlu divalidasi oleh seorang pakar berupa dokter hewan untuk menvalidasi apakah gejala-gejala yang di*input* dan *output* prediksi dari model telah sesuai dengan gejala yang telah diberikan. Tabel 4 merupakan hasil validasi pakar terkait *output* yang diberikan oleh model.

**Kode 5. Interferensi dengan Data *Dummy***

```
top_n = 10
threshold = 0.7
y_test_array = np.array(y_test)
y_pred_prob = pipeline.predict_proba(X_test)
top_n_indices = np.argsort(y_pred_prob, axis=1)[:, -top_n:]
class_labels = mlb.classes_
for i, indices in enumerate(top_n_indices[:3]):
    top_n_labels = class_labels[indices]
    top_n_probs = y_pred_prob[i, indices]
    filtered_labels = [f"{label}: {prob * 100:.2f}%" for label, prob in
zip(top_n_labels, top_n_probs) if prob >= threshold]
    true_label = mlb.inverse_transform(y_test_array[[i]])[0]
    correct_predictions = sum(1 for label in top_n_labels if label in
true_label)
    accuracy_percentage = (correct_predictions / len(top_n_labels)) * 100 if
len(top_n_labels) > 0 else 0
    if len(filtered_labels) > 0:
        print(f"Sampel {i+1}: Top-{top_n} Prediksi (Threshold {threshold}) =
{filtered_labels}, Akurasi = {accuracy_percentage:.2f}%")
    else:
        print(f"Sampel {i+1}: Tidak ada prediksi yang melebihi threshold
{threshold}, Akurasi = 0.00%")
```

```
Sampel 1: Top-10 Prediksi (Threshold 0.7) =
['Hepatitis: 99.82%', 'Anemia: 99.97%', 'Keracunan Leptospira: 100.00%', 'Jaundice:
100.00%', 'Cholangiohepatitis: 100.00%', 'Icterus: 100.00%', 'Parasit Darah: 100.00%'],
Akurasi = 70.00%

Sampel 2: Top-10 Prediksi (Threshold 0.7) = ['Parvo Virus: 77.60%', 'Nefritis: 99.90%',
'Hepatitis: 99.97%', 'Calici Virus: 99.99%', 'Gastritis: 99.99%', 'Keracunan: 99.99%',
'Helminthiasis: 99.99%'],
Akurasi = 70.00%

Sampel 3: Top-10 Prediksi (Threshold 0.7) = ['Hepatitis: 99.82%', 'Anemia: 99.97%',
'Keracunan Leptospira: 100.00%', 'Jaundice: 100.00%', 'Cholangiohepatitis: 100.00%',
'Icterus: 100.00%', 'Parasit Darah: 100.00%'],
Akurasi = 70.00%
```

**Gambar 6. Hasil Interferensi dengan Data *Dummy***

**Tabel 4. Hasil Validasi Pakar**

| No | Gejala | Prediksi Penyakit* | Status |
|---|---|---|---|
| 1 | "Lesu": Berlangsung selama 1 hari, | "*Hepatoenecephalopathy*: 100.00%", | Valid |
| | "Nafsu makan menurun": Berlangsung selama 1 hari, | "Tetraparalisis: 100,00%", | Valid |
| | "Demam": Berlangsung selama 1 hari, | **"Abses: 100.00%",** | Valid |
| | "Kaki luka": Berlangsung selama 1 hari, | "*Dyspnea*: 99.99%", | Valid |
| | "Kuku terlepas": Berlangsung selama 1 hari, | "Trauma otot: 99,99%", | Valid |
| | "Kaku": Berlangsung selama 1 hari, | "Serangan jantung: 99,99%", | Valid |
| | "Kaki kaku": Berlangsung selama 1 hari, | "*Syncope*: 99.99%", | Valid |
| | "Sakit mata": Berlangsung selama 1 hari, | "*TrachealCollapse*:99.99%", | Valid |
| | "Mata abnormal": Berlangsung selama 1 hari | "*Vulnus*:99.98%", | Valid |
| | | **"*Chlamydia*: 99.94%"** | Valid |
| 2 | "Enggan minum": Berlangsung selama 1 hari, | "*Caries*: 100.00%", | Valid |
| | "Lesu": Berlangsung selama 1 hari, | "*Sneezing*: 100.00%", | Valid |
| | "Berat badan menurun": Berlangsung selama 1 hari, | **"*Sinusitis*: 100.00%",** | Valid |
| | "Batuk": Berlangsung selama 1 hari, | "FIP: 100.00%", | Valid |
| | "Bersin": Berlangsung selama 1 hari, | "*Rhinotracheitis*: 100.00%", | Valid |
| | "Kesulitan bernafas": Berlangsung selama 1 hari, | **"FIV: 100.00%",** | Valid |
| | "Mulut berbusa": Berlangsung selama 1 hari, | **"*Stomatitis*: 100.00%",** | Valid |
| | "Bau mulut": Berlangsung selama 1 hari, | "*Gingivitis*: 99.99%", | Valid |
| | "Karang gigi": Berlangsung selama 1 hari | "*Herpes*: 99.98%", | Valid |
| | | "FeLV: 99.97%" | Valid |

***Prediksi penyakit yang dicetak tebal merupakan indikasi *zoonosis***

## 5. Kesimpulan dan Saran

### 5.1 Kesimpulan

Berdasarkan hasil dari pengembangan model, dapat disimpulkan bahwa model *gradient boosting* dengan strategi *OneVsRestClassifier* (Ovr) sangat baik memprediksi penyakit dengan gejala-gejala yang telah diberikan. Berdasarkan hasil *hyperparameter tuning* dengan menggunakan *gridsearch*, model dengan kombinasi parameter *learning rate* 0,05, *max_depth* 7, *min_samples_leaf* 1, *min_samples_split* 2, *n_estimators 200*, dan *subsample 0,9* dengan nilai *accuracy* sebesar 81%. Sehingga dari parameter yang telah ditetapkan, model *gradient boosting* yang diuji menggunakan data uji menghasilkan *accuracy* 88%, *precision* 97%, *recall* 96%, *F1-score* 96%, dan *hamming loss* 0,29%. Dengan interferensi serta validasi dokter hewan, model dapat dikatakan mampu memprediksi penyakit dengan baik melalui gejala-gejala yang telah diberikan.

### 5.2 Saran

Untuk penelitian selanjutnya dapat menggunakan metode *gradient boosting* lainnya agar mempermudah proses *tuning* dikarenakan metode-metode tersebut bisa dibantu dengan kartu grafis agar mempercepat proses pelatihan dengan *hyperparameter tuning*, sebisa mungkin

menggunakan *dataset* yang lebih besar agar prediksi sangat akurat, dan untuk kedepannya dapat diimplementasikan dalam bentuk *mobile* ataupun dalam bentuk *website*.

**Referensi**

[1] A. H. Freedman and R. K. Wayne, "Deciphering the Origin of Dogs: From Fossils to Genomes," *Annual Review of Animal Biosciences*, vol. 5, pp. 281–307, Feb. 2017.

[2] C. R. Wulandari, A. Burhanuddin, P. L. Faradina, P. A. Wibawati, A. Abdramanov, "Understanding the level of animal welfare and associated factors among cat owners in Banyuwangi, Indonesia" *Veterinary World*, vol. 17, pp. 1210-1215, June 2024.

[3] S. A. Azaliarahma, E. L. Martyan, A. Rahmadani, R. T. Dirgahayu, "Pengembangan Aplikasi Konsultasi Online Dan Janji Temu Dokter Hewan Berbasis Android", *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, vol. 2, no.1, pp. 33-41, July 2022

[4] Tempo.co, "Hanya 29,5 Persen Hewan Peliharaan di Indonesia yang Pernah Kunjungi Dokter Hewan," 2024. [Online]. Available: https://gaya.tempo.co/read/1708700/hanya-295-persen-hewan-peliharaan-di-indonesia-yang-pernah-kunjungi-dokter-hewan (accessed Sep. 17, 2024).

[5] A. A. Ahmed, S. Sayed, A. Abdoulhalik, S. Moutari, L. Oyedele, "Applications of machine learning to water resources management: A review of present status and future opportunities" *Journal of Cleaner Production*, vol. 441, 2024

[6] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, Q. Liu, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, vol. 30, 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[7] H. Deng, Y. Zhou, L. Wang, and C. Zhang, "Ensemble learning for the early prediction of neonatal jaundice with genetic features," *BMC Medical Informatics and Decision Making*, vol. 21, no. 1, Dec. 2021.

[8] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," *Artificial Intelligence Review*, vol. 54, no. 3, pp. 1937–1967, Mar. 2021.

[9] B. Tuychlev, "A Guide to The Gradient Boosting Algorithm," 2023. [Online]. Available: https://www.datacamp.com/tutorial/guide-to-the-gradient-boosting-algorithm (accessed Sep. 27, 2024)

[10] W. Xu, L. Ning, and Y. Luo, "Wind speed forecast based on post-processing of numerical weather predictions using a gradient boosting decision tree algorithm," *Atmosphere (Basel)*, vol. 11, no. 7, Jul. 2020.

[11] A. Basavaraju, J. Du, F. Zhou, J. Ji, "Machine Learning Approach to Road Surface Anomaly Assessment Using Smartphone Sensors," *IEEE Sensors Journal*, vol. 20, no. 5, pp. 2635–2647, 2020.

[12] B. Clark, F. Lee, "What is gradient boosting?" 2025. [Online]. Available: https://www.ibm.com/think/topics/gradient-boosting (accessed Sep. 30, 2024)

[13] S. Diantika, H. Nalatissifa, R. Supriyadi, N. Maulidah, and A. Fauzi, "Implementasi Multi-Class Gradient Boosting Untuk Mengklasifikasikan Jenis Hewan Pada Kebun Binatang," *Antivirus : Jurnal Ilmiah Teknik Informatika*, vol. 17, no. 1, pp. 33–40, Jun. 2023.

[14] N. V. Putranto, M. Akrom, G. A. Trinapradika, and A. History, "Implementasi Fungsi Polinomial pada Algoritma Gradient Boosting Regressor: Studi Regresi pada Dataset Obat-Obatan Kadaluarsa sebagai Material Antikorosi," *Jurnal Teknologi dan Manajemen Informatika*, vol. 9, no. 2, pp. 172–182, 2023.

[15] I. D. Mienye and Y. Sun, "A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects," in *IEEE Access*, vol. 10, pp. 99129-99149, 2022.