

CHAPTER 4 – RESULTS AND DISCUSSIONS

4.1 Training Result (Multi-Class Classification)

At first, the classification method that was used is multi-class classification. Multiclass classification is a machine learning task which aims to assign examples to one of several predetermined classes or categories—where each instance belongs to exactly one class. This means that each of the class is mutually exclusive to each other.

In multi-class classification, the output is restricted to only one class. It is later realized that this classification method is not the most suitable method for this research because there is a chance that an animal can have more than one disease at the same time. Therefore, another classification method was deemed more suitable which is multi-label classification which will be explained further in *Chapter 4.2*. Nevertheless, the experiment results using multi-class classification will be explained in the subchapters below.

4.1.1 First Training

The first dataset used for the training consists of 68 usable samples with 3 different labels. The dataset comprises 8 attributes (7 predictors, 1 class) as shown in *Table 4.1*. The results of training using this dataset are shown in *Table 4.2*, *Table 4.3*, and *Table 4.4*.

Table 4.1: Attributes of the First Dataset from Medical Record

No.	Attribute	Options
1	Body Temperature	Number value
2	General Condition	Normal; Weak
3	Diarrhea	Yes; No; Bloody
4	Feces Consistency	Solid; Squishy; Liquid
5	Feces Condition	Normal; Bloody

6	Vomit	Yes; No
7	Appetite Lost	Yes; No
8	Diagnose	Clean; Parvo; Corona

There are only samples for Parvo and Corona because other diseases have fewer samples with worse quality. Even in this case, the samples are also relatively low compared with other related works with only 5 samples of Corona, 27 samples of Parvo, and 36 samples of clean diagnoses.

Table 4.2:Result of First Training with 70/30 Train/Test Split

Model	F1 (weighted)	Training Time
Random Forest	0.774190	1.83 sec
Logistic Regression	0.726857	0.38 sec
Naïve Bayes	0.710286	0.14 sec
KNN	0.703524	0.22 sec
SVM	0.431429	0.16 sec

Table 4.3:Result of First Training with 75/25 Train/Test Split

Model	F1 (weighted)	Training Time
Random Forest	0.689214	1.82 sec
Naïve Bayes	0.680690	0.13 sec
Logistic Regression	0.646143	0.40 sec
KNN	0.608048	0.21 sec
SVM	0.387143	0.16 sec

Table 4.4:Result of First Training with 80/20 Train/Test Split

Model	F1 (weighted)	Training Time
Random Forest	0.695635	2.40 sec
Logistic Regression	0.639937	0.37 sec
KNN	0.622984	0.29 sec
Naïve Bayes	0.540683	0.19 sec
SVM	0.331190	0.23 sec

From these results, the highest score achieved is using Random Forest with 70/30 split ratio. The model with the fastest training time overall is Naïve Bayes with an average of 0.16 seconds, but it has worse scores overall compared to random forest. Random Forest has the longest training time among other models. Even then, 1.83 seconds achieved with 70/20 split is still considered fast.

Even though the result from the first training looks decent, the dataset is actually imbalanced. As have been mentioned previously, there were only 5 samples of Corona, 27 samples of Parvo, and 36 samples of no disease diagnosed. Imbalanced dataset has a high chance to produce a biased result. The result is usually biased towards the class with the highest sample size and almost ignoring the class with lowest sample size.

4.1.2 Second Training

The dataset used for the second training consists of 3000 samples of 3 different labels (1000 each) with 15 various attributes (14 predictors, 1 class) as shown in *Table 4.5*. It is randomly generated based on a set of possible symptoms for each disease which was made by a senior veterinarian from XYZ Clinic. This is done because the dataset gathered from the medical record for the first training was lacking in complexity and had a fairly low accuracy overall. The results of training using this dataset are shown in *Table 4.6*, *Table 4.7*, and *Table 4.8*.

Table 4.5: Attributes of the Second Dataset Made with the Guidance from Senior Veterinarians

No.	Attribute	Options
1	Food Type	Food Type A; Food Type B; Food Type C; Food Type D; Food Type E
2	Eating Frequency	Eat 1x/day; Eat 2x/day; Eat 3x / day; Eat >5x/day
3	Eating Appetite	Does not want to eat; Picky eater; Normal appetite; Eat everything; Eat poop
4	Drinking Appetite	Normal drinking; Less drinking; Does not like to drink; Drink a lot

5	Body Temperature	Normal (38.6 – 39.2 °C); Hot (>39.5 °C); Cold (<38.0 °C)
6	General Condition	Agile; Sluggish; Lethargic; Pale; Unconscious; Dehidration
7	Vomit Frequency	Vomit 1x/day; Vomit 2x/day; Vomit 3x/day; Vomit >5x/day
8	Vomit Time	Vomit Time A; Vomit Time B; Vomit Time C; Vomit Time D; Vomit Time E
9	Vomit Type	Vomit Type A; Vomit Type B; Vomit Type C; Vomit Type D
10	Vomit Color	Vomit Color A; Vomit Color B; Vomit Color C; Vomit Color D; Vomit Color E; Vomit Color F
11	Diarrhea Frequency	Diarrhea 1x/day; Diarrhea 2x/day; Diarrhea 3x/day; Diarrhea >5x/day
12	Diarrhea Type	Diarrhea Type A; Diarrhea Type B; Diarrhea Type C; Diarrhea Type D; Diarrhea Type E; Diarrhea Type F; Diarrhea Type G; Diarrhea Type H; Diarrhea Type I; Diarrhea Type J; Diarrhea Type K; Diarrhea Type L
13	Diarrhea Color	Diarrhea Color A; Diarrhea Color B; Diarrhea Color C; Diarrhea Color D; Diarrhea Color E; Diarrhea Color F; Diarrhea Color G; Diarrhea Color H
14	Diarrhea Smell	Diarrhea Smell A; Diarrhea Smell B
15	Diagnose	Parvo, Distemper, Corona

Table 4.6:Result of Second Training with 70/30 Train/Test Split

Model	F1 (weighted)	Training Time
Naïve Bayes	0.944221	0.18 sec
Logistic Regression	0.942774	1.56 sec
SVM	0.938448	20.20 sec
Random Forest	0.935178	3.28 sec
KNN	0.876471	3.73 sec

Table 4.7:Result of Second Training with 75/25 Train/Test Split

Model	F1 (weighted)	Training Time
Naïve Bayes	0.944291	0.18 sec
SVM	0.942431	21.89 sec
Random Forest	0.935480	3.66 sec
Logistic Regression	0.934446	2.63 sec
KNN	0.870690	3.93 sec

Table 4.8:Result of Second Training with 80/20 Train/Test Split

Model	F1 (weighted)	Training Time
Naïve Bayes	0.944474	0.28 sec
SVM	0.939469	24.45 sec
Random Forest	0.935786	3.40 sec
Logistic Regression	0.933623	1.60 sec
KNN	0.870894	5.53 sec

From these results, the highest score is achieved using Naïve Bayes with 80/20 split ratio. The model with the fastest training time overall is also Naïve Bayes with only 0.28 seconds on 80/20 split ratio.

The second training shows a better result overall, but considering this is a synthetic dataset without labelling from experts makes this dataset not reliable enough to be used. The good result produced by the second dataset is considered because of there is a unique trait for each disease in this dataset. It is assumed that the model can differentiate the diseases better because of these unique characteristics

4.1.3 Third Training

The third training also uses medical records from XYZ Clinic but this time the filtering is changed after having discussions with a senior veterinarian. This resulted in a dataset consisting of 176 rows with 5 different labels. The dataset comprises 9 attributes (8 predictors, 1 class) as shown in *Table 4.9*.

Table 4.9:Attributes of the Third Dataset from Medical Record

No.	Attribute	Options
1	General Condition	Normal, Lethargic, Dehydration
2	Appetite Lost	Yes; No
3	Change of Food	Yes; No
4	Diarrhea	Yes; No
5	Feces Consistency	Solid; Watery; Frothy; Squishy
6	Additional Secretion	No; Blood; Mucose; Both
7	Vomit	Yes; No
8	Vaccinated	Yes; No
9	Diagnose	Amoeba; Bacteria; Giardia; Parvo; Clean

The third dataset was expected to have a better result compared to the first dataset but the results state otherwise. This is suspected because the predictors for each label have many similarities. The results for the third training are shown in *Table 4.10*, *Table 4.11*, and *Table 4.12*.

Table 4.10:Result of Third Training with 70/30 Train/Test Split

Model	F1 (weighted)	Training Time
Logistic Regression	0.395995	0.42 sec
SVM	0.393209	0.25 sec
KNN	0.392785	0.35 sec
Random Forest	0.373284	1.86 sec
Naïve Bayes	0.239202	0.14 sec

Table 4.11:Result of Third Training with 75/25 Train/Test Split

Model	F1 (weighted)	Training Time
KNN	0.505951	0.37 sec
SVM	0.424803	0.27 sec
Random Forest	0.418774	1.86 sec
Logistic Regression	0.413990	0.43 sec
Naïve Bayes	0.249979	0.15 sec

Table 4.12:Result of Third Training with 80/20 Train/Test Split

Model	F1 (weighted)	Training Time
Logistic Regression	0.435402	0.41 sec
SVM	0.426677	0.28 sec
KNN	0.401921	0.37 sec
Random Forest	0.382952	1.88 sec
Naïve Bayes	0.282290	0.15 sec

From these results, the highest score is achieved using K-Nearest Neighbor with 75/25 split ratio. The model with the fastest training time overall is Naïve Bayes with an average of 0.15 seconds, but it has the worst score overall too. K-Nearest Neighbor has the third fastest training time among other models. With training time still under 1 second, it is still considerably fast.

4.2 Training Result (Multi-Label Classification)

Multi-label classification is a machine learning task where multiple labels can be assigned to an instance simultaneously, making it possible to assign multiple binary labels to each instance. After doing experiments using multi-class classification, it is later realized that multi-label classification is more suitable because instances may be assigned with several labels at the same time.

For this training, the dataset that is used is the third dataset because it is more superior compared to the first dataset. For the multi-class classification using the third dataset, instances with more than one labels assigned were split into two different instances. For this training, the dataset is adjusted to fit into the standard required for multi-label

classification. The training results for each of the class are shown in *Table 4.13* until *Table 4.17*.

Table 4.13: Training Result for Amoeba

Model	Avg ROC-AUC	Training Time
Naive Bayes Amoeba	0.781136	0.12 sec
SVM Amoeba	0.747727	0.15 sec
Logistic Regression Amoeba	0.725909	0.31 sec
Random Forest Amoeba	0.722273	1.91 sec
KNN Amoeba	0.633409	0.12 sec

Table 4.14: Training Result for Bacteria

Model	Avg ROC-AUC	Training Time
Naive Bayes Bacteria	0.608491	0.12 sec
SVM Bacteria	0.588857	0.16 sec
Random Forest Bacteria	0.580303	1.89 sec
Logistic Regression Bacteria	0.573074	0.20 sec
KNN Bacteria	0.540878	0.12 sec

Table 4.15: Training Result for Giardia

Model	Avg ROC-AUC	Training Time
Random Forest Giardia	0.638168	2.51 sec
KNN Giardia	0.623584	0.16 sec
SVM Giardia	0.550027	0.25 sec
Naive Bayes Giardia	0.475828	0.14 sec
Logistic Regression Giardia	0.408413	0.18 sec

Table 4.16: Training Result for Parvo

Model	Avg ROC-AUC	Training Time
Logistic Regression Parvo	0.945726	0.15 sec
SVM Parvo	0.925214	0.14 sec
Random Forest Parvo	0.914957	1.85 sec
KNN Parvo	0.880235	0.14 sec
Naive Bayes Parvo	0.827885	0.12 sec

Table 4.17: Training result for Clean

Model	Avg ROC-AUC	Training Time
Random Forest Clean	0.797382	1.92 sec
Logistic Regression Clean	0.777618	0.17 sec
KNN Clean	0.762927	0.14 sec
SVM Clean	0.702778	0.17 sec
Naive Bayes Clean	0.559135	0.11 sec

4.3 Hyperparameter Tuning

The hyperparameter tuning is done in order to achieve a higher accuracy result. It is done using the model from the multi-label classification training as a base. The hyperparameter tuning is done using `RandomizedSearchCV` to find the best combination of parameters. `RandomizedSearchCV` will run for as many iterations as stated at the beginning and it will return the best score achieved by a certain combination of parameters

`RandomizedSearchCV` tries out a fixed number of parameter settings sampled from the specified distributions. This is different from `GridSearchCV` which tries out all possible combinations of given parameters. Both methods are tried out to determine whether `RandomizedSearchCV` is enough compared to the training time it takes or `GridSearchCV` is better.

4.3.1 Tuned Models for Amoeba (`RandomizedSearchCV`)

The time taken for running the `RandomizedSearchCV` and the best parameters for each of the models are shown in *Figure 4.1* until *Figure 4.5*.

```
*****
Logistic Regression Amoeba random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 4.31 seconds
Logistic Regression Amoeba best score: 0.6943
Logistic Regression Amoeba best params: {'solver': 'liblinear', 'penalty': 'l1', 'C': 29.763514416313132}
*****
```

Figure 4.1: `RandomizedSearchCV` Duration and Best Parameters for Logistic Regression on Amoeba

```
*****
Random Forest Amoeba random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 41.02 seconds
Random Forest Amoeba best score: 0.6181
Random Forest Amoeba best params: {'n_estimators': 450, 'min_samples_split': 4, 'min_samples_leaf': 3, 'max_depth': 7}
*****
```

Figure 4.2:RandomizedSearchCV Duration and Best Parameters for Random Forest on Amoeba

```
*****
Naïve Bayes Amoeba random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.31 seconds
Naïve Bayes Amoeba best score: 0.7420
Naïve Bayes Amoeba best params: {'var_smoothing': 0.009221978823334321}
*****
```

Figure 4.3:RandomizedSearchCV Duration and Best Parameters for Naïve Bayes on Amoeba

```
*****
KNN Amoeba random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.33 seconds
KNN Amoeba best score: 0.6519
KNN Amoeba best params: {'weights': 'distance', 'n_neighbors': 7, 'metric': 'manhattan'}
*****
```

Figure 4.4:RandomizedSearchCV Duration and Best Parameters for K-Nearest Neighbor on Amoeba

```
*****
SVM Amoeba random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.55 seconds
SVM Amoeba best score: 0.7274
SVM Amoeba best params: {'kernel': 'rbf', 'gamma': 0.1, 'C': 0.1}
*****
```

Figure 4.5:RandomizedSearchCV Duration and Best Parameters for Support Vector Machine on Amoeba

Table 4.18:ROC-AUC Scores for Amoeba after Hyperparameter Tuning using
RandomizedSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
Naive Bayes Amoeba	0.781136	0.781136	0.38 sec	0.09 sec
Logistic Regression Amoeba	0.725909	0.740000	2.51 sec	0.16 sec
Random Forest Amoeba	0.722273	0.730000	42.72 sec	9.45 sec
SVM Amoeba	0.747727	0.709091	0.52 sec	0.15 sec
KNN Amoeba	0.633409	0.682045	0.36 sec	0.12 sec

As shown in *Table 4.18*, Naïve Bayes have the best score before and after tuning even though there is no improvement from the tuning. This means that we can use either the tuned or non-tuned Naïve Bayes for predicting amoeba label during the deployment. The ROC-AUC curve for tuned Naïve Bayes on amoeba is shown on *Figure 4.6*.

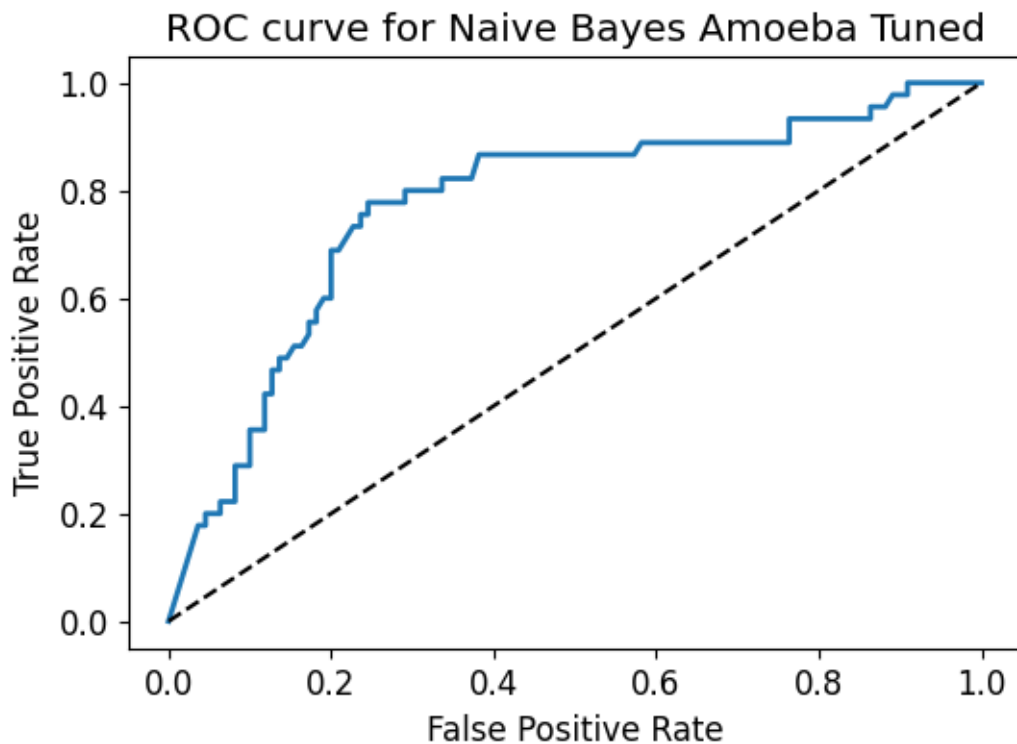


Figure 4.6:ROC Curve for RandomizedSearchCV Tuned Naive Bayes on Amoeba

4.3.2 Tuned Models for Amoeba (GridSearchCV)

The time taken for running the GridSearchCV and the best parameters for each of the models are shown in Figure 4.7 until Figure 4.11.

```
*****
Logistic Regression Amoeba grid search running...

Fitting 5 folds for each of 40 candidates, totalling 200 fits

Time taken: 3.10 seconds
Logistic Regression Amoeba best score: 0.7758
Logistic Regression Amoeba best params: {'C': 11.288378916846883, 'penalty': 'l1', 'solver': 'liblinear'}
*****
```

Figure 4.7:GridSearchCV Duration and Best Parameters for Logistic Regression on Amoeba

```
*****
Random Forest Amoeba grid search running...

Fitting 5 folds for each of 1216 candidates, totalling 6080 fits

Time taken: 4880.42 seconds
Random Forest Amoeba best score: 0.7414
Random Forest Amoeba best params: {'max_depth': 5, 'min_samples_leaf': 3, 'min_samples_split': 8, 'n_estimators': 250}
*****
```

Figure 4.8:GridSearchCV Duration and Best Parameters for Random Forest on Amoeba

```
*****
Naive Bayes Amoeba grid search running...

Fitting 5 folds for each of 200 candidates, totalling 1000 fits

Time taken: 7.16 seconds
Naive Bayes Amoeba best score: 0.8025
Naive Bayes Amoeba best params: {'var_smoothing': 0.0029331662783900424}
*****
```

Figure 4.9:GridSearchCV Duration and Best Parameters for Naïve Bayes on Amoeba

```
*****
KNN Amoeba grid search running...

Fitting 5 folds for each of 48 candidates, totalling 240 fits

Time taken: 1.31 seconds
KNN Amoeba best score: 0.6869
KNN Amoeba best params: {'metric': 'minkowski', 'n_neighbors': 11, 'weights': 'uniform'}
*****
```

Figure 4.10:GridSearchCV Duration and Best Parameters for K-Nearest Neighbor on Amoeba

```

*****
SVM Amoeba grid search running...

Fitting 5 folds for each of 100 candidates, totalling 500 fits

Time taken: 4.86 seconds
SVM Amoeba best score: 0.7556
SVM Amoeba best params: {'C': 10000.0, 'gamma': 0.0001, 'kernel': 'rbf'}
*****

```

Figure 4.11: GridSearchCV Duration and Best Parameters for Support Vector Machine on Amoeba

Table 4.19: ROC-AUC Scores for Amoeba after Hyperparameter Tuning using GridSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
Naïve Bayes Amoeba	0.781136	0.781136	7.16 sec	0.12 sec
Logistic Regression Amoeba	0.725909	0.742500	3.10 sec	0.10 sec
Random Forest Amoeba	0.722273	0.726818	4880 sec	3.88 sec
SVM Amoeba	0.747727	0.721818	4.86 sec	0.27 sec
KNN Amoeba	0.633409	0.682045	1.31 sec	0.12 sec

As shown in *Table 4.19*, Naïve Bayes have the best score before and after tuning even though there is no improvement from the tuning. The score obtained from GridSearchCV tuning is the same as RandomizedSearchCV for Naïve Bayes on amoeba. The ROC-AUC curve for tuned Naïve Bayes on amoeba is shown on *Figure 4.12*.

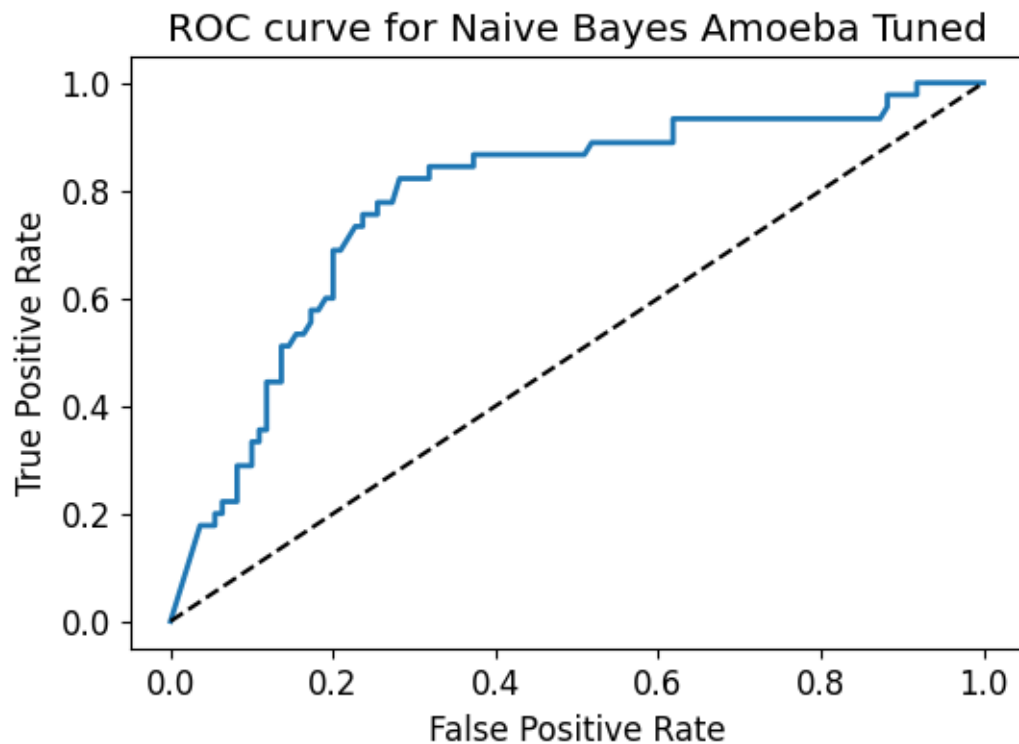


Figure 4.12:ROC Curve for GridSearchCV Tuned Naive Bayes on Amoeba

4.3.3 Tuned Models for Bacteria (RandomizedSearchCV)

The time taken for running the RandomizedSearchCV and the best parameters for each of the models are shown in *Figure 4.13* until *Figure 4.17*.

```
*****
Logistic Regression Bacteria random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.35 seconds
Logistic Regression Bacteria best score: 0.5178
Logistic Regression Bacteria best params: {'solver': 'liblinear', 'penalty': 'l2', 'C': 0.08858667904100823}
*****
```

Figure 4.13:RandomizedSearchCV Duration and Best Parameters for Logistic Regression on Bacteria

```
*****
Random Forest Bacteria random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 45.10 seconds
Random Forest Bacteria best score: 0.5395
Random Forest Bacteria best params: {'n_estimators': 450, 'min_samples_split': 4, 'min_samples_leaf': 3, 'max_depth': 7}
*****
```

Figure 4.14:RandomizedSearchCV Duration and Best Parameters for Random Forest on Bacteria

```
*****
Naïve Bayes Bacteria random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.55 seconds
Naive Bayes Bacteria best score: 0.4897
Naive Bayes Bacteria best params: {'var_smoothing': 0.20970464013232323}
*****
```

Figure 4.15:RandomizedSearchCV Duration and Best Parameters for Naïve Bayes on Bacteria

```
*****
KNN Bacteria random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.64 seconds
KNN Bacteria best score: 0.5814
KNN Bacteria best params: {'weights': 'uniform', 'n_neighbors': 11, 'metric': 'manhattan'}
*****
```

Figure 4.16:RandomizedSearchCV Duration and Best Parameters for K-Nearest Neighbor on Bacteria

```
*****
SVM Bacteria random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 1.02 seconds
SVM Bacteria best score: 0.5984
SVM Bacteria best params: {'kernel': 'rbf', 'gamma': 0.0001, 'C': 1}
*****
```

Figure 4.17:RandomizedSearchCV Duration and Best Parameters for Support Vector Machine on Bacteria

Table 4.20:ROC-AUC Scores for Bacteria after Hyperparameter Tuning

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
SVM Bacteria	0.595107	0.644223	0.59 sec	0.16 sec
Random Forest Bacteria	0.580303	0.641635	41.87 sec	7.75 sec
Naive Bayes Bacteria	0.608491	0.605587	0.34 sec	0.11 sec
Logistic Regression	0.573074	0.575979	0.40 sec	0.10 sec

Bacteria				
KNN Bacteria	0.540878	0.575884	0.30 sec	0.10 sec

As shown in *Table 4.20*, Naïve Bayes has the best score before tuning but Support Vector Machine has higher score overall after tuning. This means that we can use the tuned Support Vector Machine for predicting bacteria label during the deployment. The ROC-AUC curve for tuned Support Vector Machine on bacteria is shown on *Figure 4.18*.

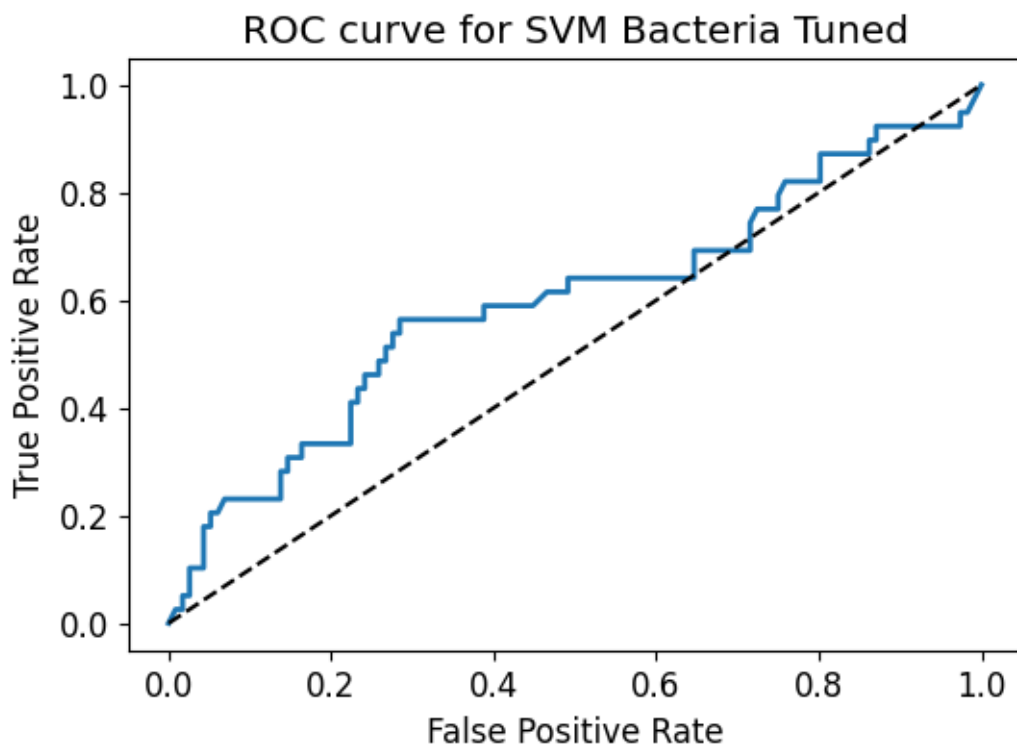


Figure 4.18:ROC Curve for RandomizedSearchCV Tuned Support Vector Machine on Bacteria

4.3.4 Tuned Models for Bacteria (GridSearchCV)

The time taken for running the GridSearchCV and the best parameters for each of the models are shown in Figure 4.19 until Figure 4.23.

```
*****
Logistic Regression Bacteria grid search running...

Fitting 5 folds for each of 40 candidates, totalling 200 fits

Time taken: 1.26 seconds
Logistic Regression Bacteria best score: 0.6033
Logistic Regression Bacteria best params: {'C': 0.08858667904100823, 'penalty': 'l2', 'solver': 'liblinear'}
*****
```

Figure 4.19:GridSearchCV Duration and Best Parameters for Logistic Regression on Bacteria

```
*****
Random Forest Bacteria grid search running...

Fitting 5 folds for each of 1216 candidates, totalling 6080 fits

Time taken: 4885.60 seconds
Random Forest Bacteria best score: 0.6697
Random Forest Bacteria best params: {'max_depth': 5, 'min_samples_leaf': 3, 'min_samples_split': 2, 'n_estimators': 100}
*****
```

Figure 4.20:GridSearchCV Duration and Best Parameters for Random Forest on Bacteria

```
*****
Naïve Bayes Bacteria grid search running...

Fitting 5 folds for each of 200 candidates, totalling 1000 fits

Time taken: 5.99 seconds
Naïve Bayes Bacteria best score: 0.6016
Naïve Bayes Bacteria best params: {'var_smoothing': 0.23272024789604084}
*****
```

Figure 4.21:GridSearchCV Duration and Best Parameters for Naïve Bayes on Bacteria

```
*****
KNN Bacteria grid search running...

Fitting 5 folds for each of 48 candidates, totalling 240 fits

Time taken: 2.11 seconds
KNN Bacteria best score: 0.6108
KNN Bacteria best params: {'metric': 'minkowski', 'n_neighbors': 17, 'weights': 'uniform'}
*****
```

Figure 4.22:GridSearchCV Duration and Best Parameters for K-Nearest Neighbor on Bacteria

```
*****
SVM Bacteria grid search running...

Fitting 5 folds for each of 100 candidates, totalling 500 fits

Time taken: 7.18 seconds
SVM Bacteria best score: 0.6613
SVM Bacteria best params: {'C': 0.23357214690901212, 'gamma': 1, 'kernel': 'rbf'}
*****
```

Figure 4.23:GridSearchCV Duration and Best Parameters for Support Vector Machine on Bacteria

Table 4.21: ROC-AUC Scores for Bacteria after Hyperparameter Tuning using GridSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
SVM Bacteria	0.561774	0.678883	7.18 sec	0.14 sec
Random Forest Bacteria	0.580303	0.645802	4885 sec	1.66 sec
Naïve Bayes Bacteria	0.608491	0.601420	5.99 sec	0.13 sec
Logistic Regression Bacteria	0.573074	0.575979	1.26 sec	0.12 sec
KNN Bacteria	0.540878	0.573169	2.11 sec	0.15 sec

As shown in *Table 4.21*, Naïve Bayes has the best score before tuning but Support Vector Machine has higher score overall after tuning. The score obtained from GridSearchCV tuning is higher than RandomizedSearchCV for Support Vector Machine on bacteria. The ROC-AUC curve for tuned Support Vector Machine on bacteria is shown on *Figure 4.24*.

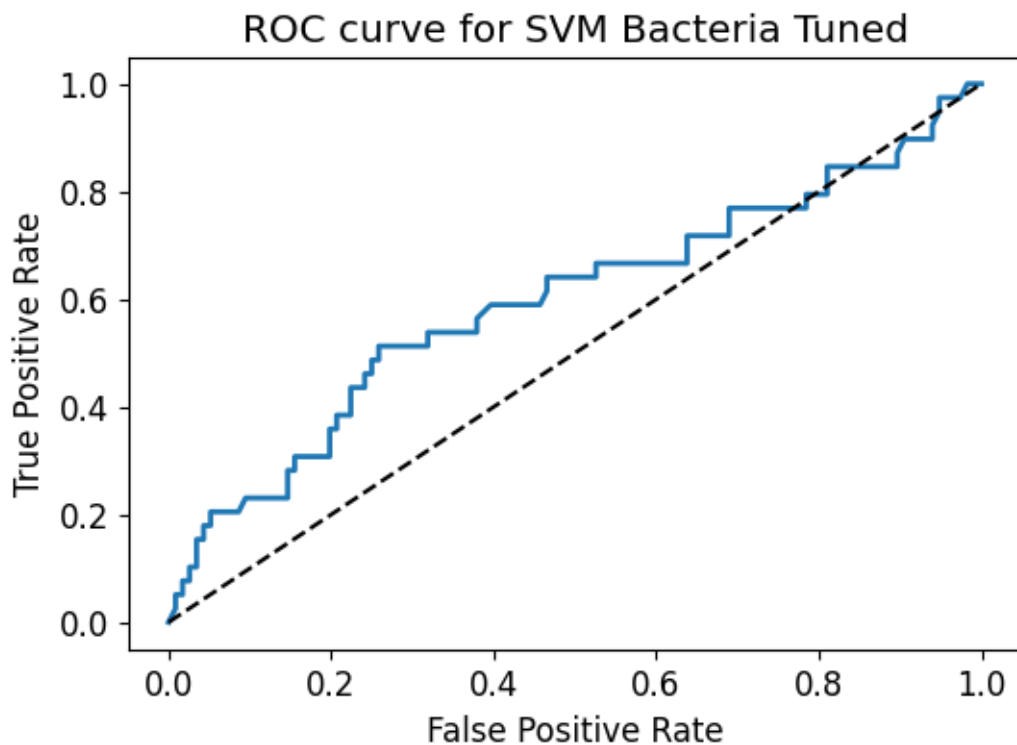


Figure 4.24: ROC Curve for GridSearchCV Tuned Support Vector Machine on Bacteria

4.3.5 Tuned Models for Giardia (RandomizedSearchCV)

The time taken for running the RandomizedSearchCV and the best parameters for each of the models are shown in *Figure 4.25* until *Figure 4.29*.

```
*****
Logistic Regression Giardia random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.64 seconds
Logistic Regression Giardia best score: 0.5000
Logistic Regression Giardia best params: {'solver': 'liblinear', 'penalty': 'l1', 'C': 0.0006951927961775605}
*****
```

Figure 4.25:RandomizedSearchCV Duration and Best Parameters for Logistic Regression on Giardia

```
*****
Random Forest Giardia random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 40.88 seconds
Random Forest Giardia best score: 0.5203
Random Forest Giardia best params: {'n_estimators': 450, 'min_samples_split': 4, 'min_samples_leaf': 3, 'max_depth': 7}
*****
```

Figure 4.26:RandomizedSearchCV Duration and Best Parameters for Random Forest on Giardia

```
*****
Naive Bayes Giardia random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.34 seconds
Naive Bayes Giardia best score: 0.3974
Naive Bayes Giardia best params: {'var_smoothing': 0.20970464013232323}
*****
```

Figure 4.27:RandomizedSearchCV Duration and Best Parameters for Naïve Bayes on Giardia

```
*****
KNN Giardia random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.30 seconds
KNN Giardia best score: 0.6806
KNN Giardia best params: {'weights': 'distance', 'n_neighbors': 5, 'metric': 'euclidean'}
*****
```

Figure 4.28:RandomizedSearchCV Duration and Best Parameters for K-Nearest Neighbor on Giardia

```
*****
SVM Giardia random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.56 seconds
SVM Giardia best score: 0.5542
SVM Giardia best params: {'kernel': 'rbf', 'gamma': 1, 'C': 1}
*****
```

Figure 4.29:RandomizedSearchCV Duration and Best Parameters for Support Vector Machine on Giardia

Table 4.22:ROC-AUC Scores for Giardia after Hyperparameter Tuning using RandomizedSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
Random Forest Giardia	0.638168	0.649279	43.15 sec	9.27 sec
KNN Giardia	0.623584	0.621554	0.35 sec	0.11 sec
SVM Giardia	0.550027	0.577324	0.55 sec	0.15 sec
Logistic Regression Giardia	0.408413	0.500000	0.45 sec	0.14 sec
Naive Bayes Giardia	0.475828	0.440785	0.36 sec	0.13 sec

As shown in *Table 4.22*, Random Forest has the best score before and after tuning. This means that we can use the tuned Random Forest for predicting giardia label during the deployment. The ROC-AUC curve for tuned Random Forest on giardia is shown on *Figure 4.30*.

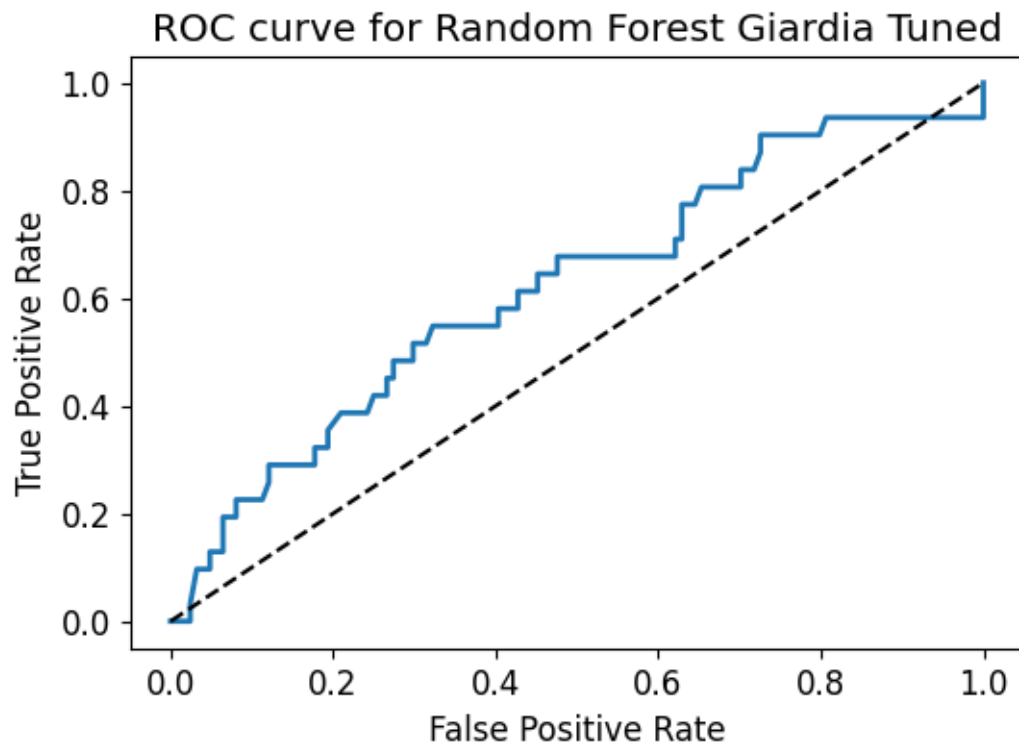


Figure 4.30:ROC Curve for RandomizedSearchCV Tuned Random Forest on Giardia

4.3.6 Tuned Models for Giardia (GridSearchCV)

The time taken for running the GridSearchCV and the best parameters for each of the models are shown in Figure 4.31 until Figure 4.35.

```
*****
Logistic Regression Giardia grid search running...

Fitting 5 folds for each of 40 candidates, totalling 200 fits

Time taken: 1.29 seconds
Logistic Regression Giardia best score: 0.5000
Logistic Regression Giardia best params: {'C': 0.0001, 'penalty': 'l1', 'solver': 'liblinear'}
*****
```

Figure 4.31:GridSearchCV Duration and Best Parameters for Logistic Regression on Giardia

```
*****
Random Forest Giardia grid search running...

Fitting 5 folds for each of 1216 candidates, totalling 6080 fits

Time taken: 4931.17 seconds
Random Forest Giardia best score: 0.6403
Random Forest Giardia best params: {'max_depth': 9, 'min_samples_leaf': 3, 'min_samples_split': 8, 'n_estimators': 900}
*****
```

Figure 4.32:GridSearchCV Duration and Best Parameters for Random Forest on Giardia

```
*****
Naive Bayes Giardia grid search running...

Fitting 5 folds for each of 200 candidates, totalling 1000 fits

Time taken: 7.38 seconds
Naive Bayes Giardia best score: 0.4564
Naive Bayes Giardia best params: {'var_smoothing': 1.6831803533309549e-09}
*****
```

Figure 4.33:GridSearchCV Duration and Best Parameters for Naïve Bayes on Giardia

```
*****
KNN Giardia grid search running...

Fitting 5 folds for each of 48 candidates, totalling 240 fits

Time taken: 1.38 seconds
KNN Giardia best score: 0.6746
KNN Giardia best params: {'metric': 'minkowski', 'n_neighbors': 3, 'weights': 'uniform'}
*****
```

Figure 4.34:GridSearchCV Duration and Best Parameters for K-Nearest Neighbor on
Giardia

```
*****
SVM Giardia grid search running...

Fitting 5 folds for each of 100 candidates, totalling 500 fits

Time taken: 4.81 seconds
SVM Giardia best score: 0.6823
SVM Giardia best params: {'C': 1.623776739188721, 'gamma': 1, 'kernel': 'rbf'}
*****
```

Figure 4.35:GridSearchCV Duration and Best Parameters for Support Vector Machine
on Giardia

Table 4.23:ROC-AUC Scores for Giardia after Hyperparameter Tuning using
GridSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
KNN Giardia	0.623584	0.704274	1.38 sec	0.12 sec
Random Forest Giardia	0.638168	0.641854	4931 sec	15.27 sec
SVM Giardia	0.477804	0.553018	4.81 sec	0.14 sec
Logistic Regression Giardia	0.408413	0.500000	1.29 sec	0.13 sec
Naive Bayes Giardia	0.475828	0.475828	7.38 sec	0.09 sec

As shown in *Table 4.23*, Random Forest has the best score before tuning but K-Nearest Neighbor has higher score overall after tuning. The score obtained from GridSearchCV tuning for K-Nearest Neighbor is higher than RandomizedSearchCV for Random Forest on giardia. The ROC-AUC curve for tuned K-Nearest Neighbor on giardia is shown on *Figure 4.36*.

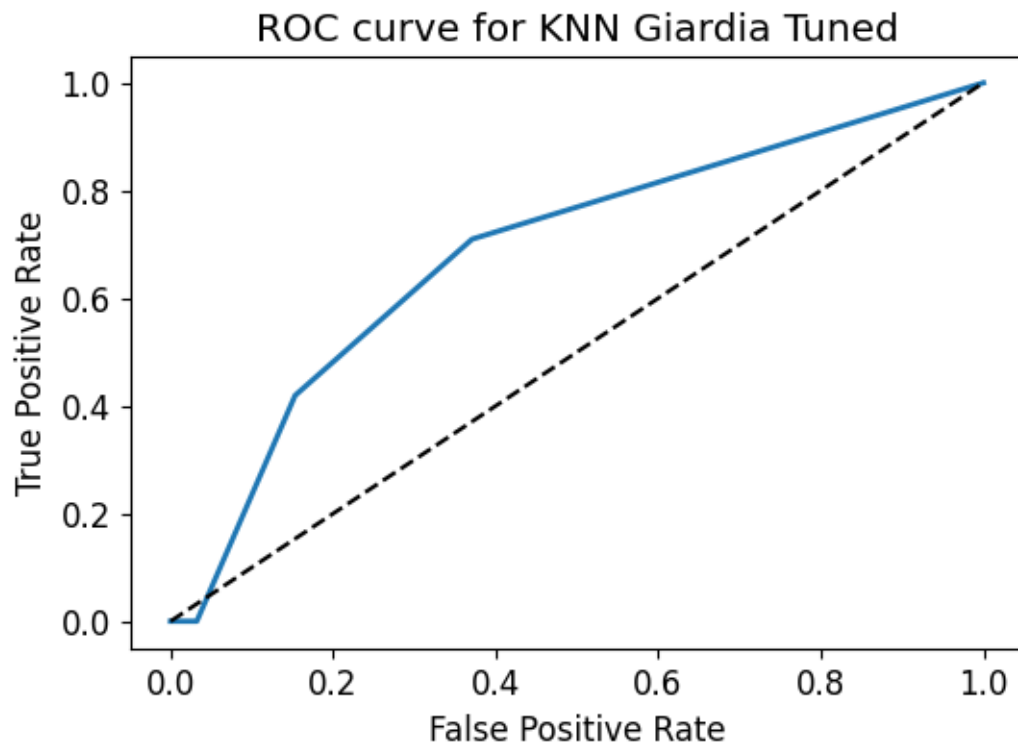


Figure 4.36:ROC Curve for GridSearchCV Tuned K-Nearest Neighbor on Giardia

4.3.7 Tuned Models for Parvo (RandomizedSearchCV)

The time taken for running the RandomizedSearchCV and the best parameters for each of the models are shown in *Figure 4.37* until *Figure 4.41*.

```
*****
Logistic Regression Parvo random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.32 seconds
Logistic Regression Parvo best score: 0.9405
Logistic Regression Parvo best params: {'solver': 'liblinear', 'penalty': 'l2', 'C': 29.763514416313132}
*****
```

Figure 4.37:RandomizedSearchCV Duration and Best Parameters for Logistic Regression on Parvo

```
*****
Random Forest Parvo random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 42.45 seconds
Random Forest Parvo best score: 0.9150
Random Forest Parvo best params: {'n_estimators': 350, 'min_samples_split': 8, 'min_samples_leaf': 7, 'max_depth': 11}
*****
```

Figure 4.38:RandomizedSearchCV Duration and Best Parameters for Random Forest on Parvo

```
*****
Naïve Bayes Parvo random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.35 seconds
Naïve Bayes Parvo best score: 0.9123
Naïve Bayes Parvo best params: {'var_smoothing': 0.20970464013232323}
*****
```

Figure 4.39:RandomizedSearchCV Duration and Best Parameters for Naïve Bayes on Parvo

```
*****
KNN Parvo random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.38 seconds
KNN Parvo best score: 0.8887
KNN Parvo best params: {'weights': 'uniform', 'n_neighbors': 13, 'metric': 'euclidean'}
*****
```

Figure 4.40:RandomizedSearchCV Duration and Best Parameters for K-Nearest Neighbor on Parvo

```
*****
SVM Parvo random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.48 seconds
SVM Parvo best score: 0.9433
SVM Parvo best params: {'kernel': 'rbf', 'gamma': 0.01, 'C': 1000}
*****
```

Figure 4.41:RandomizedSearchCV Duration and Best Parameters for Support Vector Machine on Parvo

Table 4.24:ROC-AUC Scores for Parvo after Hyperparameter Tuning using
RandomizedSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
Logistic Regression	0.939957	0.957265	0.34 sec	0.15 sec

Parvo				
Naive Bayes Parvo	0.827885	0.954701	0.38 sec	0.10 sec
SVM Parvo	0.925214	0.941880	0.76 sec	0.21 sec
Random Forest Parvo	0.914957	0.933974	45.98 sec	7.34 sec
KNN Parvo	0.880235	0.900641	0.64 sec	0.16 sec

As shown in *Table 4.24*, Logistic Regression has the best score before and after tuning. This means that we can use the tuned Logistic Regression for predicting parvo label during the deployment. The ROC-AUC curve for tuned Logistic Regression on parvo is shown on *Figure 4.42*.

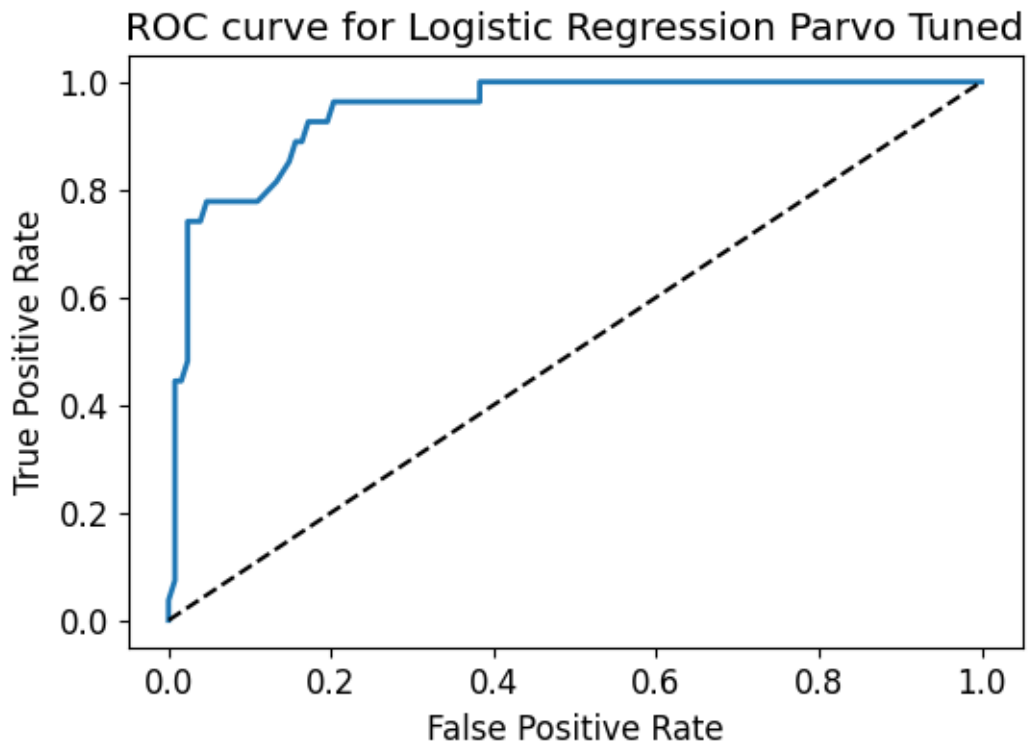


Figure 4.42: ROC Curve for RandomizedSearchCV Tuned Logistic Regression on Parvo

4.3.8 Tuned Models for Parvo (GridSearchCV)

The time taken for running the GridSearchCV and the best parameters for each of the models are shown in *Figure 4.43* until *Figure 4.47*.

```
*****
Logistic Regression Parvo grid search running...

Fitting 5 folds for each of 40 candidates, totalling 200 fits

Time taken: 1.45 seconds
Logistic Regression Parvo best score: 0.9513
Logistic Regression Parvo best params: {'C': 78.47599703514607, 'penalty': 'l2', 'solver': 'liblinear'}
*****
```

Figure 4.43:GridSearchCV Duration and Best Parameters for Logistic Regression on Parvo

```
*****
Random Forest Parvo grid search running...

Fitting 5 folds for each of 1216 candidates, totalling 6080 fits

Time taken: 4871.70 seconds
Random Forest Parvo best score: 0.9269
Random Forest Parvo best params: {'max_depth': 7, 'min_samples_leaf': 3, 'min_samples_split': 8, 'n_estimators': 300}
*****
```

Figure 4.44:GridSearchCV Duration and Best Parameters for Random Forest on Parvo

```
*****
Naïve Bayes Parvo grid search running...

Fitting 5 folds for each of 200 candidates, totalling 1000 fits

Time taken: 7.45 seconds
Naïve Bayes Parvo best score: 0.9589
Naïve Bayes Parvo best params: {'var_smoothing': 0.23272024789604084}
*****
```

Figure 4.45:GridSearchCV Duration and Best Parameters for Naïve Bayes on Parvo

```
*****
KNN Parvo random grid search running...

Fitting 5 folds for each of 48 candidates, totalling 240 fits

Time taken: 1.43 seconds
KNN Parvo best score: 0.9319
KNN Parvo best params: {'metric': 'minkowski', 'n_neighbors': 13, 'weights': 'uniform'}
*****
```

Figure 4.46:GridSearchCV Duration and Best Parameters for K-Nearest Neighbor on Parvo

```
*****
SVM Parvo grid search running...

Fitting 5 folds for each of 100 candidates, totalling 500 fits

Time taken: 4.01 seconds
SVM Parvo best score: 0.9492
SVM Parvo best params: {'C': 3792.690190732246, 'gamma': 0.0001, 'kernel': 'rbf'}
*****
```

Figure 4.47:GridSearchCV Duration and Best Parameters for Support Vector Machine on Parvo

Table 4.25: ROC-AUC Scores for Parvo after Hyperparameter Tuning using
GridSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
Logistic Regression Parvo	0.939957	0.958547	1.45 sec	0.10 sec
Naive Bayes Parvo	0.827885	0.954701	7.45 sec	0.14 sec
SVM Parvo	0.925214	0.943162	4.01 sec	0.12 sec
Random Forest Parvo	0.914957	0.936752	4871 sec	4.67 sec
KNN Parvo	0.880235	0.900641	1.43 sec	0.10 sec

As shown in *Table 4.25*, Logistic Regression has the best score before and after tuning. The score obtained from GridSearchCV tuning is higher than RandomizedSearchCV for Logistic Regression on parvo by a slight margin. The ROC-AUC curve for tuned Logistic Regression on parvo is shown on *Figure 4.48*.

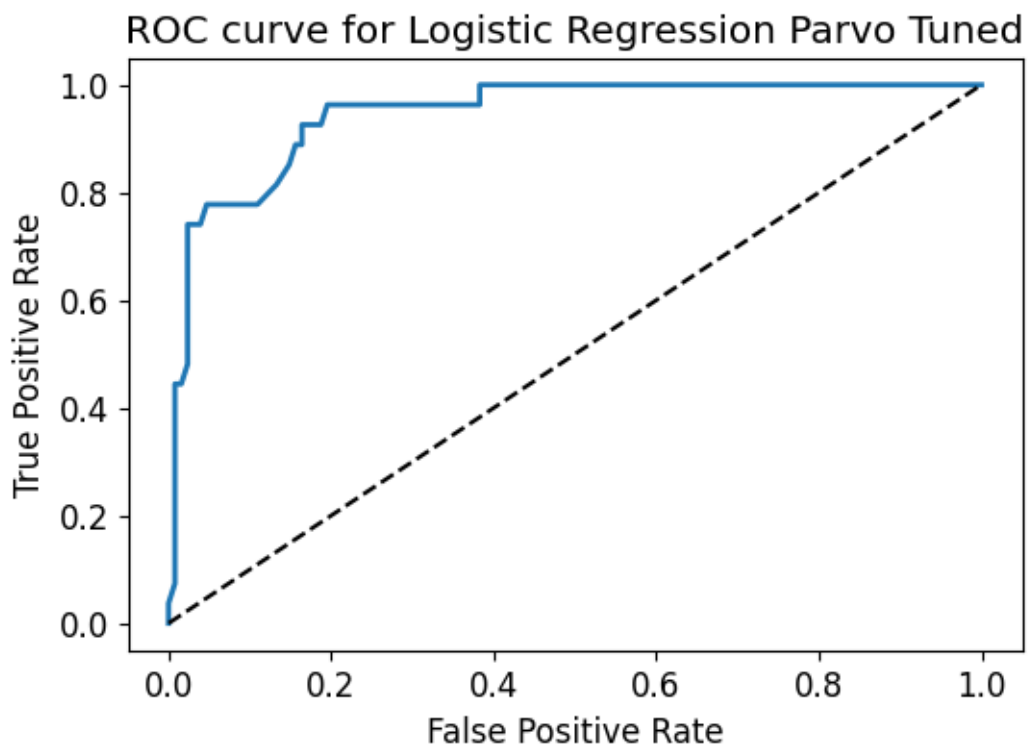


Figure 4.48: ROC Curve for GridSearchCV Tuned Logistic Regression on Parvo

4.3.9 Tuned Models for Clean (RandomizedSearchCV)

The time taken for running the RandomizedSearchCV and the best parameters for each of the models are shown in *Figure 4.49* until *Figure 4.53*.

```
*****
Logistic Regression Clean random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.37 seconds
Logistic Regression Clean best score: 0.7301
Logistic Regression Clean best params: {'solver': 'liblinear', 'penalty': 'l2', 'C': 3792.690190732246}
*****
```

Figure 4.49:RandomizedSearchCV Duration and Best Parameters for Logistic Regression on Clean

```
*****
Random Forest Clean random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 46.81 seconds
Random Forest Clean best score: 0.6908
Random Forest Clean best params: {'n_estimators': 1000, 'min_samples_split': 6, 'min_samples_leaf': 5, 'max_depth': 11}
*****
```

Figure 4.50:RandomizedSearchCV Duration and Best Parameters for Random Forest on Clean

```
*****
Naive Bayes Clean random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.29 seconds
Naive Bayes Clean best score: 0.6548
Naive Bayes Clean best params: {'var_smoothing': 0.20970464013232323}
*****
```

Figure 4.51:RandomizedSearchCV Duration and Best Parameters for Naïve Bayes on Clean

```
*****
KNN Clean random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.29 seconds
KNN Clean best score: 0.7706
KNN Clean best params: {'weights': 'distance', 'n_neighbors': 5, 'metric': 'euclidean'}
*****
```

Figure 4.52:RandomizedSearchCV Duration and Best Parameters for K-Nearest Neighbor on Clean

```

*****
SVM Clean random search running...

Fitting 5 folds for each of 10 candidates, totalling 50 fits

Time taken: 0.82 seconds
SVM Clean best score: 0.7457
SVM Clean best params: {'kernel': 'rbf', 'gamma': 0.001, 'C': 1000}
*****

```

Figure 4.53:RandomizedSearchCV Duration and Best Parameters for Support Vector
Machine on Clean

Table 4.26:ROC-AUC Scores for Clean after Hyperparameter Tuning using
RandomizedSearchCV

Model	Avg ROC- AUC	Avg ROC- AUC Tuned	Tuning Time	Training Time
SVM Clean	0.702778	0.821154	0.59 sec	0.23 sec
KNN Clean	0.762927	0.814530	0.35 sec	0.10 sec
Logistic Regression Clean	0.777618	0.803098	0.60 sec	0.14 sec
Naive Bayes Clean	0.559135	0.753365	0.30 sec	0.10 sec
Random Forest Clean	0.797382	0.744071	40.48 sec	9.21 sec

As we can see from *Table 4.26*, Random Forest has the best score before tuning with Support Vector Machine has the best score after tuning. This means that we can use the tuned Support Vector Machine for predicting clean label during the deployment. The ROC-AUC curve for tuned Support Vector Machine on clean is shown on *Figure 4.54*.

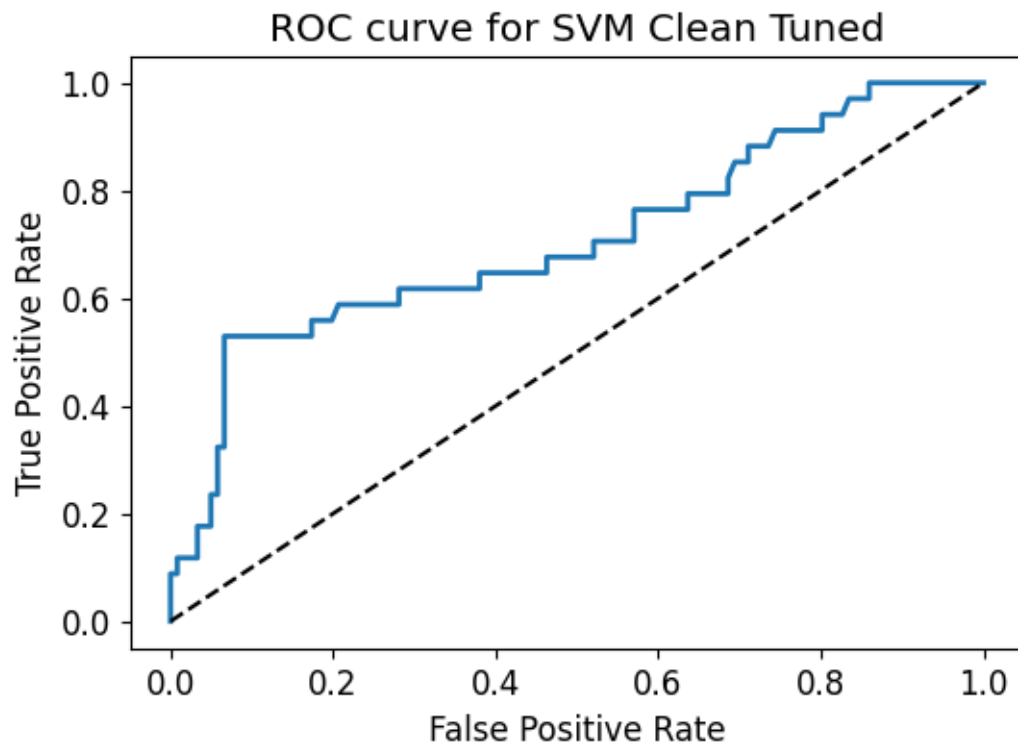


Figure 4.54:ROC Curve for RandomizedSearchCV Tuned Support Vector Machine on Clean

4.3.10 Tuned Models for Clean (GridSearchCV)

The time taken for running the GridSearchCV and the best parameters for each of the models are shown in Figure 4.55 until Figure 4.59.

```
*****
Logistic Regression Clean grid search running...

Fitting 5 folds for each of 40 candidates, totalling 200 fits

Time taken: 1.35 seconds
Logistic Regression Clean best score: 0.8255
Logistic Regression Clean best params: {'C': 11.288378916846883, 'penalty': 'l1', 'solver': 'liblinear'}
*****
```

Figure 4.55:GridSearchCV Duration and Best Parameters for Logistic Regression on Clean

```
*****
Random Forest Clean grid search running...

Fitting 5 folds for each of 1216 candidates, totalling 6080 fits

Time taken: 4922.21 seconds
Random Forest Clean best score: 0.7941
Random Forest Clean best params: {'max_depth': 7, 'min_samples_leaf': 3, 'min_samples_split': 2, 'n_estimators': 100}
*****
```

Figure 4.56:GridSearchCV Duration and Best Parameters for Random Forest on Clean

```
*****
Naive Bayes Clean grid search running...

Fitting 5 folds for each of 200 candidates, totalling 1000 fits

Time taken: 5.49 seconds
Naive Bayes Clean best score: 0.7567
Naive Bayes Clean best params: {'var_smoothing': 0.23272024789604084}
*****
```

Figure 4.57:GridSearchCV Duration and Best Parameters for Naïve Bayes on Clean

```
*****
KNN Clean grid search running...

Fitting 5 folds for each of 48 candidates, totalling 240 fits

Time taken: 1.49 seconds
KNN Clean best score: 0.8238
KNN Clean best params: {'metric': 'manhattan', 'n_neighbors': 11, 'weights': 'distance'}
*****
```

Figure 4.58:GridSearchCV Duration and Best Parameters for K-Nearest Neighbor on
Clean

```
*****
SVM Clean grid search running...

Fitting 5 folds for each of 100 candidates, totalling 500 fits

Time taken: 6.93 seconds
SVM Clean best score: 0.7814
SVM Clean best params: {'C': 29.763514416313132, 'gamma': 0.01, 'kernel': 'rbf'}
*****
```

Figure 4.59:GridSearchCV Duration and Best Parameters for Support Vector Machine
on Clean

Table 4.27:ROC-AUC Scores for Clean after Hyperparameter Tuning using
GridSearchCV

Model	Avg ROC-AUC	Avg ROC-AUC Tuned	Tuning Time	Training Time
Logistic Regression Clean	0.777618	0.814209	1.35 sec	0.12 sec
KNN Clean	0.762927	0.810363	1.49 sec	0.12 sec
SVM Clean	0.702778	0.768590	6.93 sec	0.14 sec
Naive Bayes Clean	0.559135	0.758226	5.49 sec	0.13 sec
Random Forest Clean	0.797382	0.752671	4922 sec	2.09 sec

As shown in *Table 4.27*, Random Forest has the best score before tuning but Logistic Regression has higher score overall after tuning. The score obtained from GridSearchCV tuning for Logistic Regression is lower than RandomizedSearchCV for Support Vector Machine on parvo by a slight margin. The ROC-AUC curve for tuned Logistic Regression on giardia is shown on *Figure 4.60*.

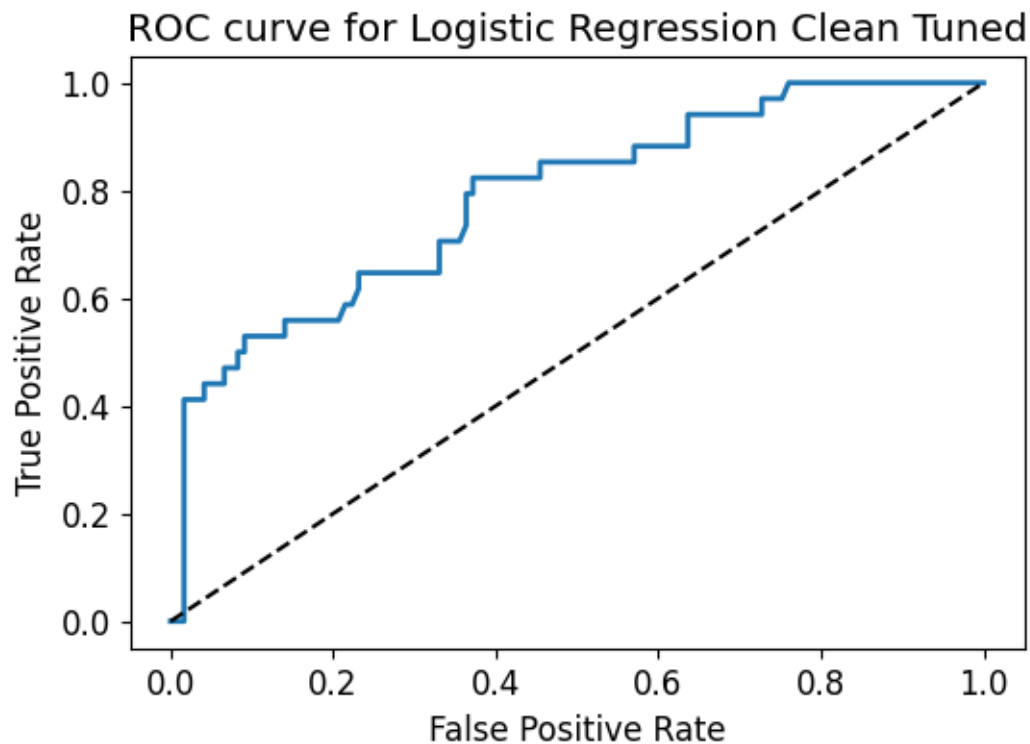


Figure 4.60:ROC Curve for GridSearchCV Tuned Logistic Regression on Clean

4.4 Web Application

The newly added features into the web application are Diagnose, Disease, and Symptom. For Diagnose, it consists of several pages of forms to describe the patient's condition. At the end of the form, a result will be displayed based on the input from the form.

← Back to home

Symptom Assessment

General Information

Please fill in this form with the patient's data.

Animal Type

☒ Dog

☐ Cat

☐ Hamster

Gender

☒ Male

☐ Female

Age

☐ <1

☐ 1-3

☐ 4-7

☒ 8-11

☐ >11

←

→

Figure 4.61:General Information Form of the Diagnose Flow

← Back to home

Symptom Assessment

Pre-diagnose form

Fill the pre-diagnose form below.

Drinking

☐ less

☐ enough

☐ more

Food Type

☐ solid

☐ squishy

Eating

☐ 0x

☐ 1x

☐ 2x

☐ 3x

☐ >3x

General Condition

☐ Paleness

☐ Dehydration

☐ Weak

Body Temperature

☐ Cold

☐ Normal

☐ Hot

←

→

Figure 4.62:Pre-diagnose Form of the Diagnose Flow

Figure 4.63: Symptom Form Page of the Diagnose Flow

The general information form shown in *Figure 4.61* contains general information related with the pet dog. Pre-diagnose form shown in *Figure 4.62* is made from deducting the medical records. Symptom form page shown in *Figure 4.63* contains every symptom stored on the database and each animal can show more than one symptom at the same time.

Figure 4.64: Result Page of the Diagnose Flow

The models are called after finishing the symptom form page. The models are run using python and hosted in a flask server. The backend Express.js server will call the

models when the diagnose function is called by the frontend. The result page of the diagnose flow is shown in *Figure 4.64*.

Disease List

NEW DISEASE

DISEASE	ANIMAL	ACTIONS
Canine Parvovirus	Dog	
Canine Distemper	Dog	
Canine Coronavirus	Dog	
Canine Adenovirus (Infectious Canine Hepatitis)	Dog	
Canine Kennel Cough (Bordetella)	Dog	
Canine Influenza (Dog Flu)	Dog	
Leptospirosis	Dog	
Lyme Disease	Dog	
Heartworm Disease	Dog	
Rabies	Dog	

10 ▾

1

2

3

4

5

>

Showing rows 1 to 10 of 48

Figure 4.65:Disease List Page

Disease Details

Animal

Type
 | ▾

Description

Action

General Information

Body Temperature
 | ▾

Drinking
Frequency (in a day)
 | ▾

Eating
Frequency (in a day)
 | ▾

Food type
 | ▾

General Condition
 | ▾

Symptoms

▾

Figure 4.66:Create New Disease Page

Disease list page is shown in *Figure 4.65*. There is a button to add new disease shown on the top right. If we click on that button, we will be redirected to create new disease page shown on *Figure 4.66*. If we click on the action button next to the disease name on disease list page, we will be redirected to disease update page shown in *Figure 4.67*. There is a delete button shown on the top right.

Disease Details

DELETE

Disease Name *

Canine Parvovirus

Animal

Type

Dog

Description

Canine parvovirus (CPV) is a highly contagious viral disease that affects dogs, especially puppies and unvaccinated adult dogs. The virus primarily attacks the gastrointestinal tract and immune system, causing severe symptoms such as vomiting, diarrhea (often bloody), lethargy, fever, and loss of appetite. In puppies, the disease can progress rapidly and lead to dehydration, septic shock, and death if not promptly treated.

Action

Intravenous fluids to address dehydration and electrolyte imbalances.
Antibiotics to prevent or treat secondary bacterial infections.
Anti-emetic medications to control vomiting.
Nutritional support, sometimes through feeding tubes, to maintain strength and aid recovery.
Isolation to prevent the spread of the virus to other dogs.
Parvo Test Kit to confirm whether it is parvovirus or not.

General Information

Body Temperature

Normal X Hot X

Drinking

Frequency (in a day)

enough X less X more X

Eating

Frequency (in a day)

1x X 2x X 3x X

Food type

solid X squishy X

General Condition

Weak X

Symptoms

Search symptoms, e.g. vomiting

Vomit

- Color: Yellowish, Greenish
- Consistency: Watery, Frothy

Fever

- Temperature: 39-40c, 40-41c

SAVE

Figure 4.67:Update Disease Page

Symptom List

NEW SYMPTOM

SYMPTOM	BODY PART	ACTIONS
Mucus	Nose	
Itch	Fur	
Poop	Stomach	
Fever	Fur	
Diare	Stomach	
Vomit	Mouth	
Fever	Others	
Diarrhea	Stomach	
Saliva	Mouth	

10 ▾

1

Showing rows 1 to 9 of 9

Figure 4.68: Symptom List Page

Symptom Details

Symptom Name *

Body Part associated with the disease

Pick one ▾

Details

Detail (frequency, time, etc)

Values (press enter to create option)

Input type ▾

REMOVE

ADD DETAILS

SAVE

Figure 4.69: Create New Symptom Page

Symptom list page is shown in *Figure 4.68*. There is a button to add new symptom shown on the top right. If we click on that button, we will be redirected to create new symptom page shown on *Figure 4.69*. If we click on the action button next to the symptom name on symptom list page, we will be redirected to symptom update page shown in *Figure 4.70*. There is a delete button shown on the top right.

Symptom Details DELETE

Symptom Name *
Vomit

Body Part associated with the disease
Mouth

Details

Detail (frequency, time, etc)
Color
Clear X Yellowish X Greenish X X
Checkbox REMOVE

Detail (frequency, time, etc)
Consistency
Watery X Frothy X X
Checkbox REMOVE

ADD DETAILS SAVE

Figure 4.70:Update Symptom Page

4.5 Feedback and Evaluation

After discussing the result of the multi-label training and showing the web-application demo to a senior veterinarian from XYZ Clinic, there are some key points gathered:

1. Accuracy – Even though there are 3 out of 5 labels with above 75% accuracy, the overall result has not fulfilled the senior veterinarian expectation yet. Nevertheless, the senior veterinarian understands that the low accuracy level is caused by a low-quality dataset for the training.
2. Diagnose Form Format – One of the main issues for the diagnose flow is the form format. For the demo, the diagnose form was made from deducting the data format of past medical records. Past medical records did not have a specific form format and this issue has been passed on to the senior veterinarian. According to the veterinarian, the clinic will provide a more complete form to be implemented in the next prototype.
3. Synthetic Dataset – Regarding the result from the synthetic dataset, they stated that the good performance from the model is probably because there is a unique trait in each of the disease in the dataset used. They argued that if the diseases are more similar with each other, the result would not be as good as it is.

Based on this feedback, XYZ Clinic tried to cooperate in making a new and proper format for storing the data. A new pre-diagnose form has been made since then as shown in *Figure 4.71*, *Figure 4.72*, and *Figure 4.73*.

Pre-diagnose form
Fill the pre-diagnose form below.

General Condition
General Condition

☐ Sehat dan tidak ada keluhan ☐ Ada keluhan tapi masih lincah dan normal ☐ Tidak mau jalan ☐ Lemas ☐ Tidak sadar ☐ Kutu kucing

☐ Kutu anjing ☐ Sakit kulit

Eating
Jenis Makanan

☐ Kibble ☐ Raw food ☐ Cooked food

Nafsu Makan

☐ Normal ☐ Rakus ☐ Picky eater ☐ Makan sedikit ☐ Tidak mau makan ☐ Dikasih makan enak mau makan

Food Poisoning
Makan Racun

☐ Pestisida ☐ Karbol ☐ Pemutih ☐ Pupuk ☐ Racun tikus ☐ Pembersih kolam

☐ Kosmetik ☐ Obat overdosis ☐ Bahan kimia ☐ Coklat ☐ Anggur ☐ Biji apel

☐ Kacang ☐ Makanan yang rusak

Waktu Memakan Racun

☐ <15 jam ☐ >2 jam

Figure 4.71:New Pre-diagnose Form (1)

Vomit
Bentuk Muntahan

☐ Makanan utuh ☐ Makanan sudah tercerna ☐ Air ☐ Lendir

Warna Muntahan

☐ Kuning ☐ Merah

Frekuensi Muntah

☐ 1x ☐ 2x ☐ >4x ☐ >10x

Poop
Score

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7

Frekuensi Poop

☐ 1x ☐ 2x ☐ >4x ☐ >6x ☐ >10x

Drinking
Nafsu Minum

☐ Normal ☐ Tidak mau minum air putih ☐ Tidak mau minum apapun ☐ Mau minum susu ☐ Mau minum kuah kaldu ☐ Mau minum yang dingin

☐ Banyak minum

Figure 4.72:New Pre-diagnose Form (2)

So far, only the pre-diagnose form got updated, the other forms in the diagnose flow remain the same. Unfortunately, the new form has not been deployed yet and new data has not been able to be gathered.

Urine

Frekuensi Pipis

☐ Susah dan tersendat ☐ Semua daerah dipipis ☐ 1x sehari ☐ 2x sehari ☐ 3x sehari

Volume Urine

☐ Normal ☐ Sedikit sekali ☐ Menetes ☐ Banyak sekali

Warna Urine

☐ Normal kuning bening ☐ Kuning pekat ☐ Kemerahan ☐ Merah darah

Figure 4.73:New Pre-diagnose Form (3)