Malaria Cell Classification using Machine Learning Algorithms

Abstract

The main aim of our project is to classify sample blood cells as infected or non-infected. In this report, we have covered every step involved in the development of a model to achieve our goal. The dataset has been described to give a better understanding of our project.

All the models have been developed in Python and to compare the performance of various models, we have developed the following models:

- Naïve Bayes
- Support Vector Machine/Classifier
- Logistic Regression
- Random Forest
- Ensemble

We have made an attempt to briefly describe each model and the underlying concept.

Lastly, we have compared the performance of the above mentioned models by using two sets of data – the train set and the test set. The report contains the accuracy obtained of each model on both the sets.

Table of Contents

Lists of Figures

Introduction

In this project, our aim is to create a predictive model using various machine learning techniques. The dataset has been cleaned, pre-processed and converted into the right format for the selected model. All these processes have been performed manually. Our aim is to develop a model with the highest possible accuracy.

Motivation

Malaria is a serious and sometimes fatal disease caused by a parasite that commonly infects a certain type of mosquito which feeds on humans. Because malaria causes so much illness and death, the disease is a great drain on many national economies.

Usually, people get malaria by being bitten by an infective female Anopheles mosquito. Only Anopheles mosquitoes can transmit malaria and they must have been infected through a previous blood meal taken from an infected person. When a mosquito bites an infected person, a small amount of blood is taken in which contains microscopic malaria parasites. About 1 week later, when the mosquito takes its next blood meal, these parasites mix with the mosquito's saliva and are injected into the person being bitten.

Because the malaria parasite is found in red blood cells of an infected person, malaria can also be transmitted through blood transfusion, organ transplant, or the shared use of needles or syringes contaminated with blood. Malaria may also be transmitted from a mother to her unborn infant before or during delivery ("congenital" malaria).

Through our project, we hope to classify the red blood cell samples into infected and uninfected using machine learning algorithms. Our model would to an extent be able to predict whether a person is infected with malaria or not depending on the blood sample. Considering the impact of the disease, the model should be as accurate as possible.

Tools used

Python for the development, training and testing of the model.

Dataset Description

This Dataset has been taken from the official NIH Website **[1]** : https://ceb.nlm.nih.gov/repositories/malaria-datasets/

The dataset contains 2 folders

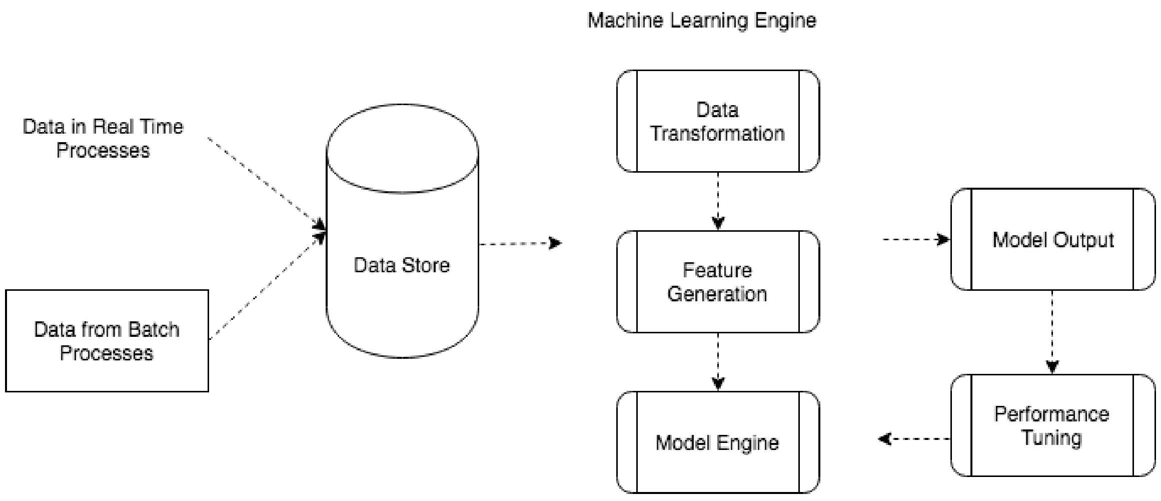
- Infected
- Uninfected

And a total of 27,558 images.

Data Pre-processing in Python

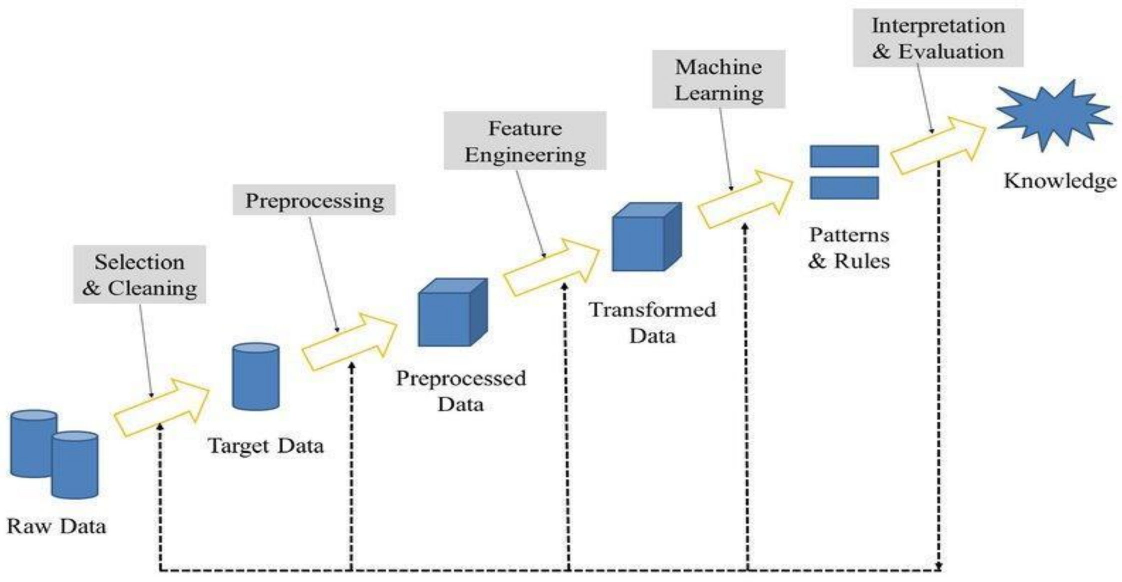
- Standard Scaler
- Image Resizing
- Normalization

System Architecture [2]



Data mining tasks performed

- **Selection and cleaning**
- **Pre processing**
- **Feature Engineering**
- **Machine Learning**
- **Interpretation and Evaluation**



Algorithm

- **Support Vector Classifier [3]**
  Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outlier detection.
  Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
  Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

- **Logistic Regression [4]**
  Logistic regression models the probabilities for classification problems with two possible outcomes. It's an extension of the linear regression model for classification problems.
  A solution for classification is logistic regression. Instead of fitting a straight line or hyperplane, the logistic regression model uses the logistic function to squeeze the output of a linear equation between 0 and 1. The logistic function is defined as:
  $\text{Logistic}(\eta) = 1/(1+\exp(-\eta))$

- **Naïve Bayes [5]**
  It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.
  Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:

$$\underbrace{P(c \mid x)}_{\text{Posterior Probability}} = \frac{\overbrace{P(x \mid c)}^{\text{Likelihood}}\,\overbrace{P(c)}^{\text{Class Prior Probability}}}{\underbrace{P(x)}_{\text{Predictor Prior Probability}}}$$

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

P(*c*|*x*) is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
P(*c*) is the prior probability of *class*.
P(*x*|*c*) is the likelihood which is the probability of the predictor given *class*.
P(*x*) is the prior probability of *predictor*

- **Random Forest [6]**

  Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,** which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

- **Ensemble Model [7]**

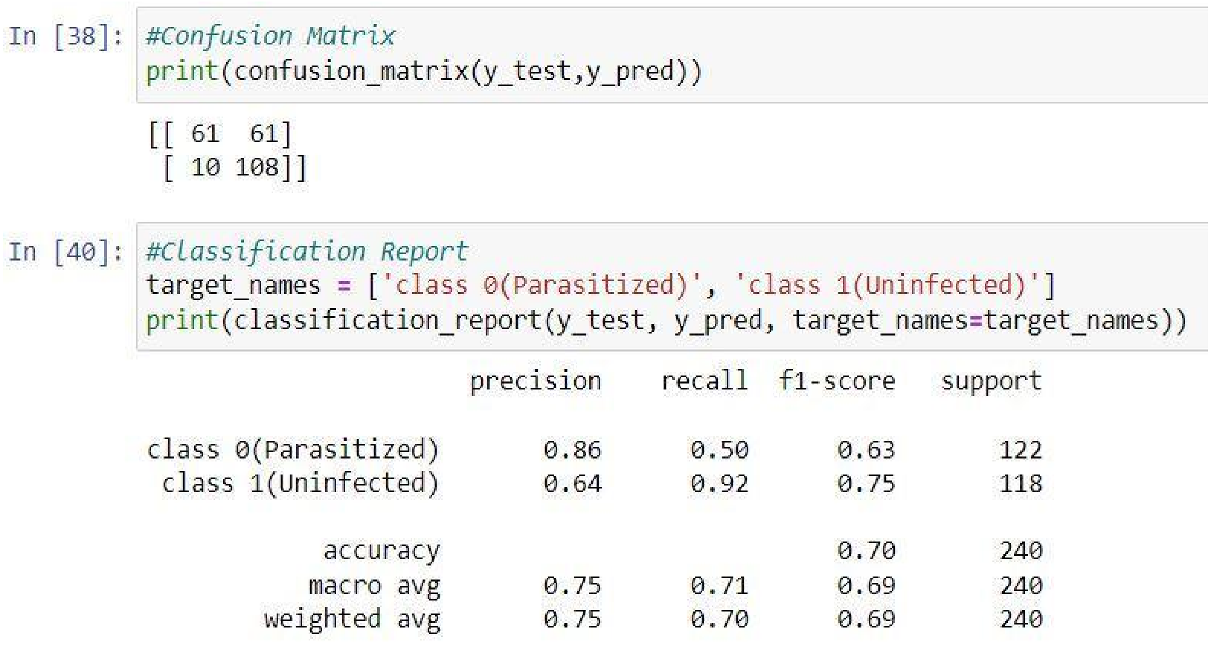
  Ensemble learning is the process by which multiple models, such as classifiers or experts, are strategically generated and combined to solve a particular computational intelligence problem. Ensemble learning is primarily used to improve the (classification, prediction, function approximation, etc.) performance of a model, or reduce the likelihood of an unfortunate selection of a poor one. Other applications of ensemble learning include assigning a confidence to the decision made by the model, selecting optimal (or near optimal) features, data fusion, incremental learning, nonstationary learning and error-correcting.

OUTPUT

Algorithm	**Accuracy on Train set**	**Accuracy on Test set**
Naïve Bayes	65.1%	70.4%
SVM	92.7%	85.8%
Logistic Regression	100%	75.8%
Random Forest	100%	87.9%
Hyper Parameter Random Forest	88.5%	81.7%
Ensemble	94%	87.9%

Visualization Screenshots

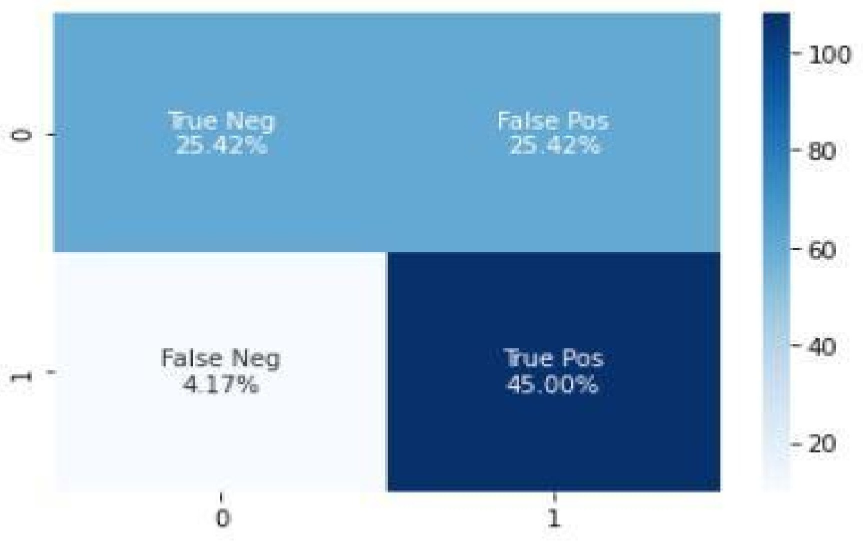
- **Naïve Bayes**



```
In [38]: #Confusion Matrix
         print(confusion_matrix(y_test,y_pred))

         [[ 61  61]
          [ 10 108]]

In [40]: #Classification Report
         target_names = ['class 0(Parasitized)', 'class 1(Uninfected)']
         print(classification_report(y_test, y_pred, target_names=target_names))

                               precision    recall  f1-score   support

         class 0(Parasitized)       0.86      0.50      0.63       122
          class 1(Uninfected)       0.64      0.92      0.75       118

                     accuracy                           0.70       240
                    macro avg       0.75      0.71      0.69       240
                 weighted avg       0.75      0.70      0.69       240
```

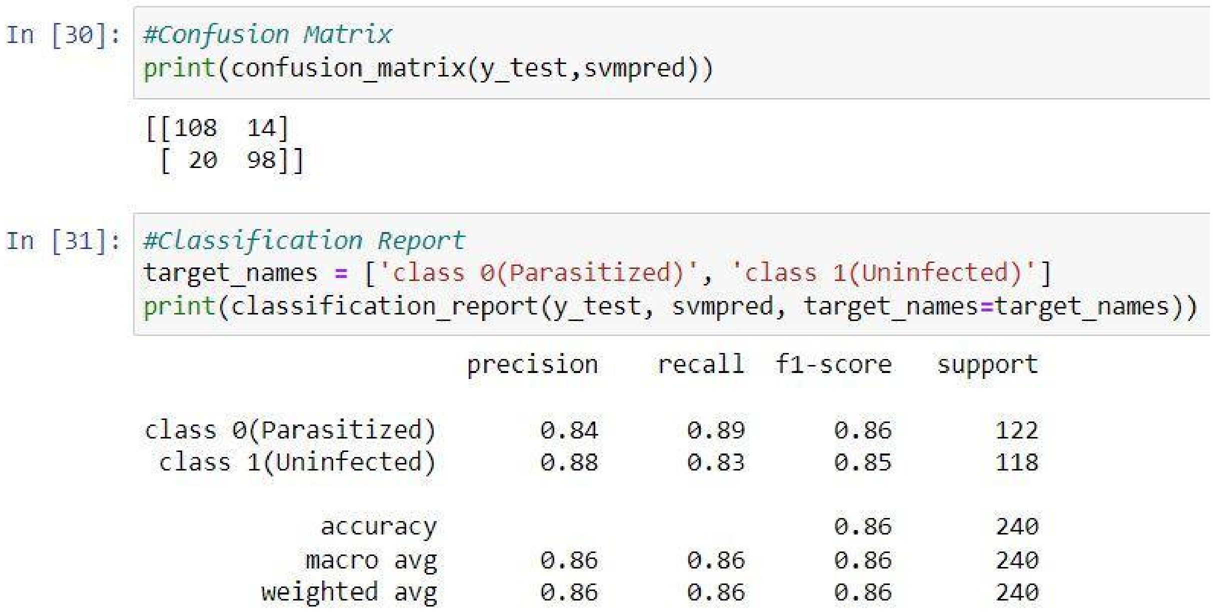
Naive Bayes Classification Report

```
Naive Bayes

<AxesSubplot:>
```



Naive Bayes Confusion Matrix

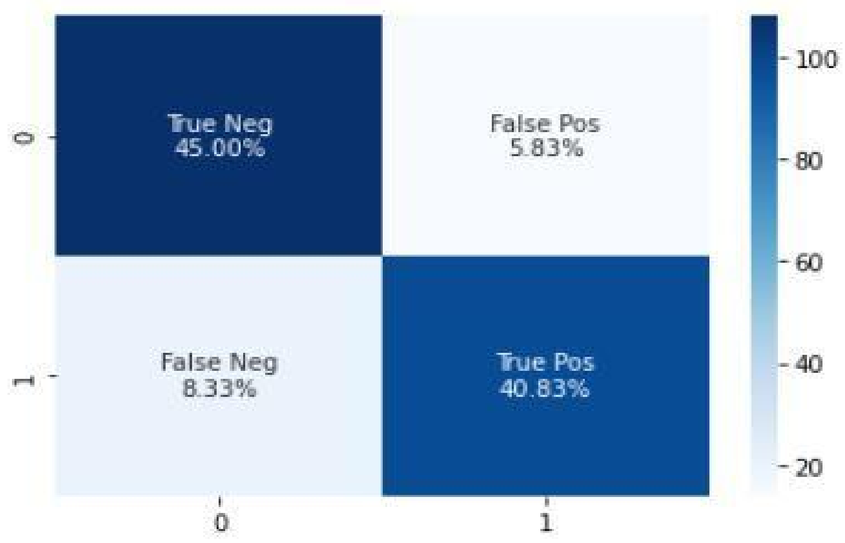
- **Support Vector Classifier**



```
In [30]: #Confusion Matrix
         print(confusion_matrix(y_test,svmpred))

         [[108  14]
          [ 20  98]]

In [31]: #Classification Report
         target_names = ['class 0(Parasitized)', 'class 1(Uninfected)']
         print(classification_report(y_test, svmpred, target_names=target_names))

                               precision    recall  f1-score   support

         class 0(Parasitized)       0.84      0.89      0.86       122
          class 1(Uninfected)       0.88      0.83      0.85       118

                     accuracy                           0.86       240
                    macro avg       0.86      0.86      0.86       240
                 weighted avg       0.86      0.86      0.86       240
```

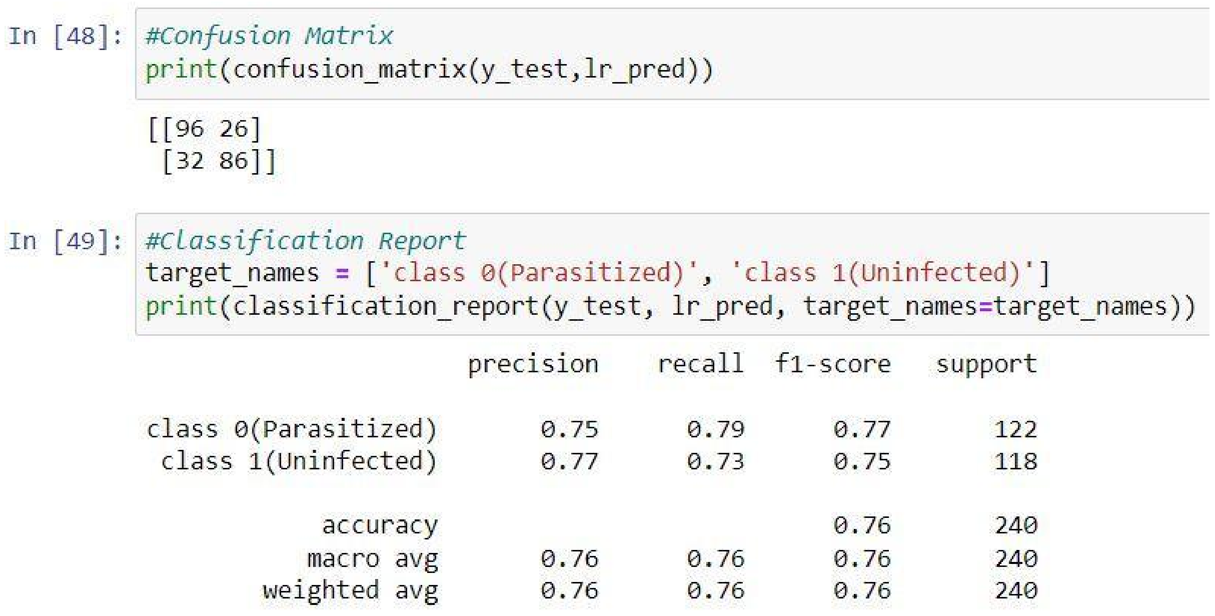
Support Vector Classification Report

```
SVM

: <AxesSubplot:>
```



SVM Confusion Matrix

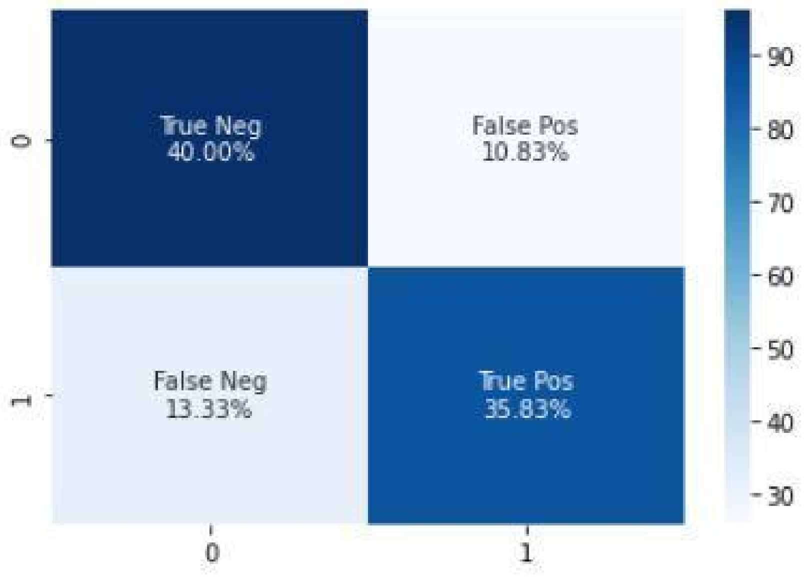
- **Logistic Regression**



```
In [48]: #Confusion Matrix
         print(confusion_matrix(y_test,lr_pred))

         [[96 26]
          [32 86]]

In [49]: #Classification Report
         target_names = ['class 0(Parasitized)', 'class 1(Uninfected)']
         print(classification_report(y_test, lr_pred, target_names=target_names))

                               precision    recall  f1-score   support

         class 0(Parasitized)       0.75      0.79      0.77       122
          class 1(Uninfected)       0.77      0.73      0.75       118

                     accuracy                           0.76       240
                    macro avg       0.76      0.76      0.76       240
                 weighted avg       0.76      0.76      0.76       240
```

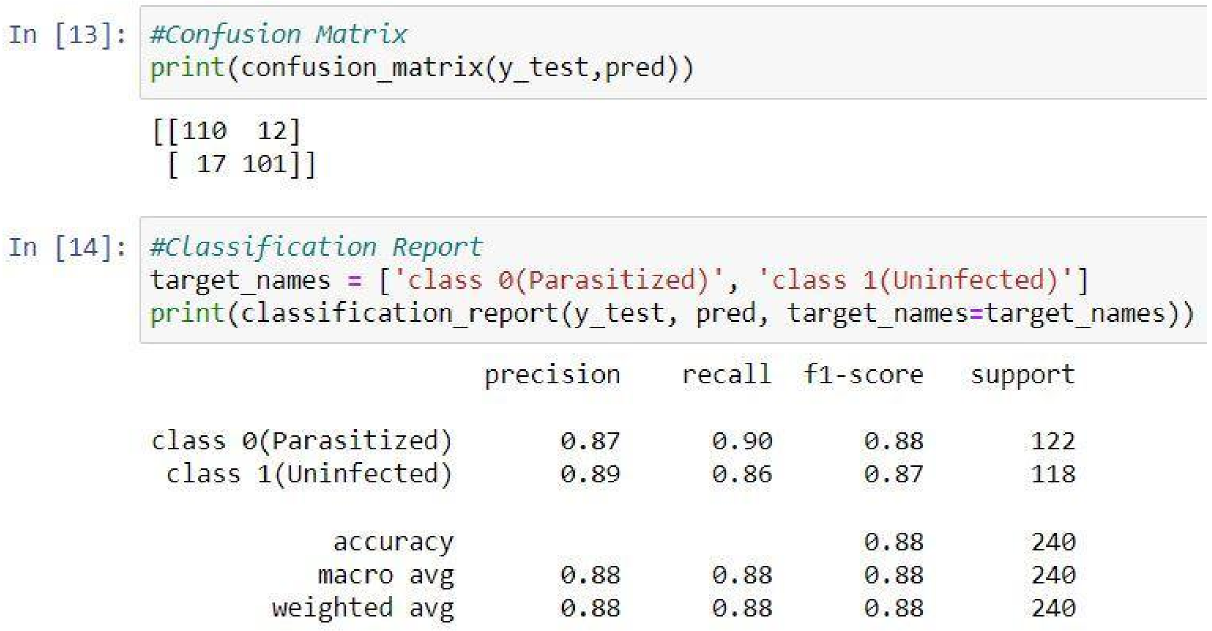
Logistic Regression Classification Report

```
Logistic Regression

<AxesSubplot:>
```



Logistic Regression Confusion Matrix

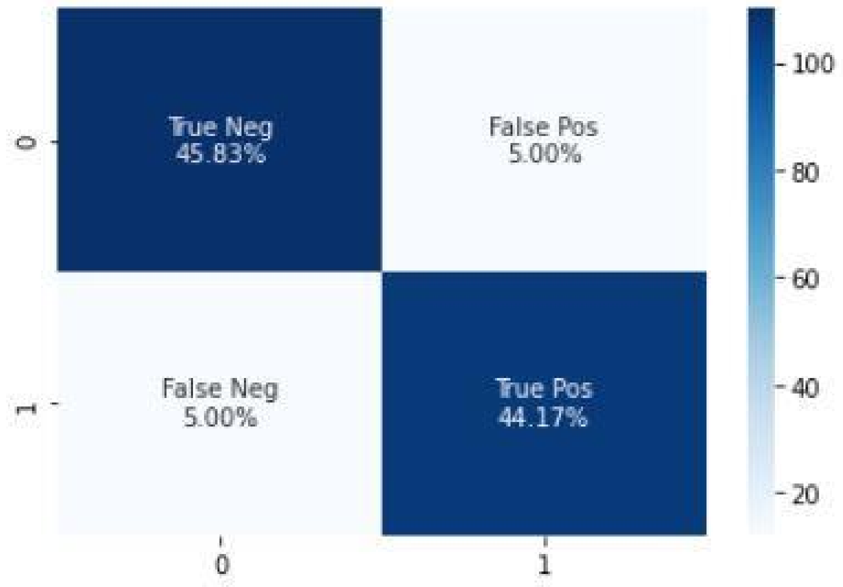
● **Random Forest**



```
In [13]: #Confusion Matrix
         print(confusion_matrix(y_test,pred))

         [[110  12]
          [ 17 101]]

In [14]: #Classification Report
         target_names = ['class 0(Parasitized)', 'class 1(Uninfected)']
         print(classification_report(y_test, pred, target_names=target_names))

                              precision    recall  f1-score   support

         class 0(Parasitized)       0.87      0.90      0.88       122
          class 1(Uninfected)       0.89      0.86      0.87       118

                     accuracy                           0.88       240
                    macro avg       0.88      0.88      0.88       240
                 weighted avg       0.88      0.88      0.88       240
```

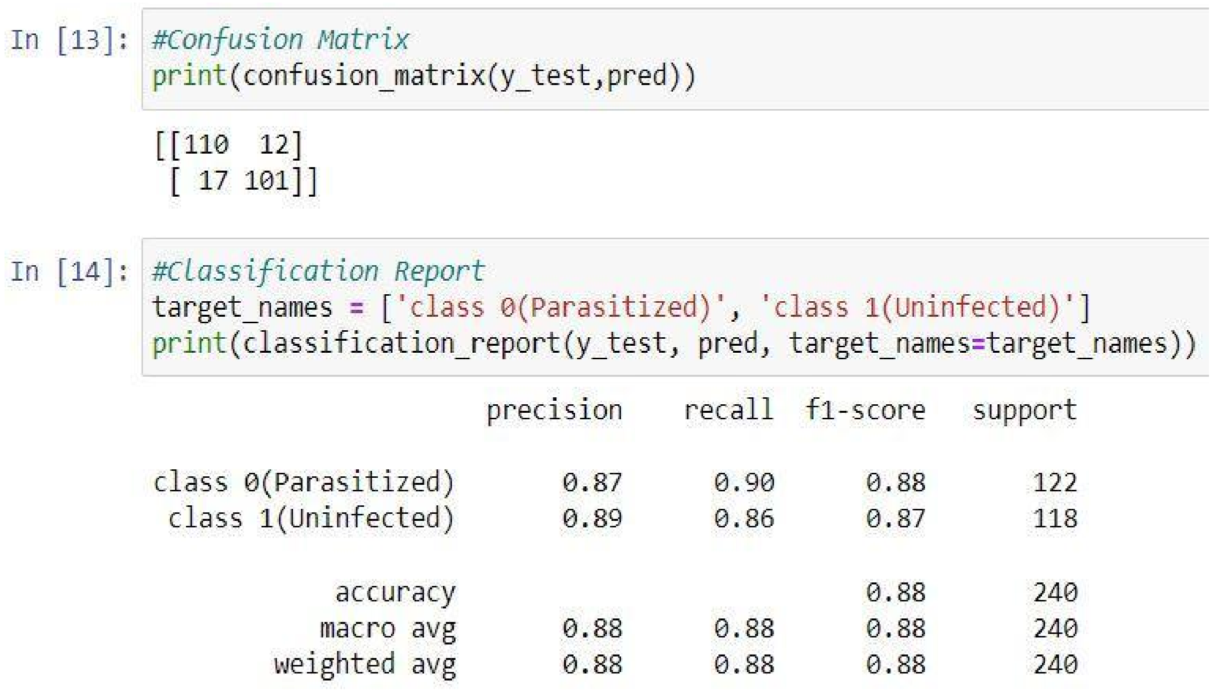
Random Forest Classification Report

```
Vanilla Random Forest

<AxesSubplot:>
```



Random Forest Confusion Matrix

- **Random Forest Hyper Parameter**

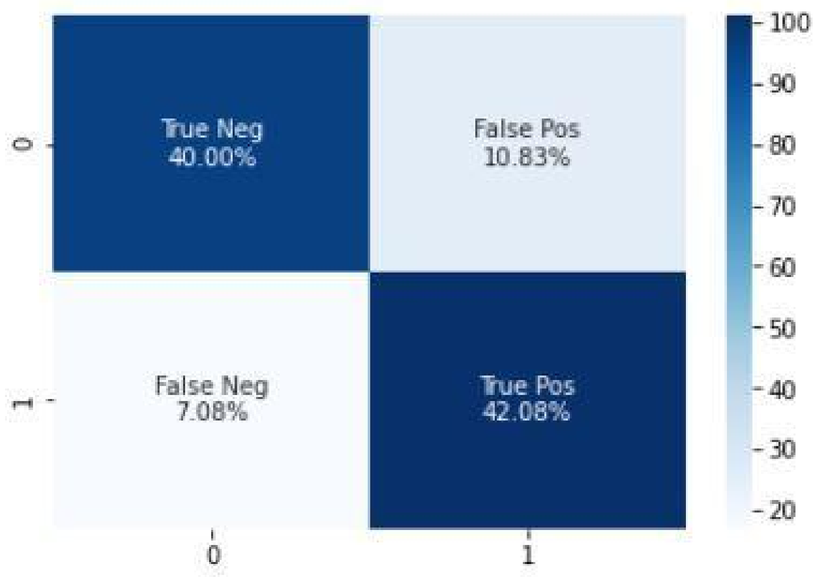


```
In [13]: #Confusion Matrix
         print(confusion_matrix(y_test,pred))

         [[110  12]
          [ 17 101]]
```

```
In [14]: #Classification Report
         target_names = ['class 0(Parasitized)', 'class 1(Uninfected)']
         print(classification_report(y_test, pred, target_names=target_names))

                               precision    recall  f1-score   support

         class 0(Parasitized)       0.87      0.90      0.88       122
          class 1(Uninfected)       0.89      0.86      0.87       118

                     accuracy                           0.88       240
                    macro avg       0.88      0.88      0.88       240
                 weighted avg       0.88      0.88      0.88       240
```

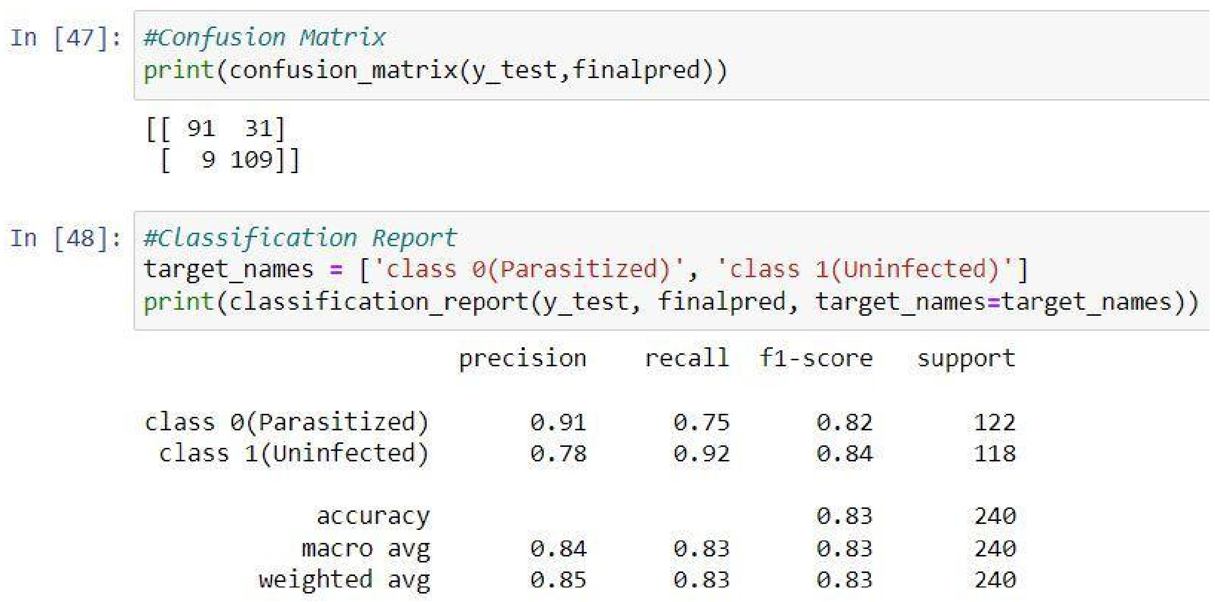
Hyper Parameter Random Forest Classification Report

```
Hyper Parameter Random Forest

<AxesSubplot:>
```



Hyper Parameter Random Forest Confusion Matrix

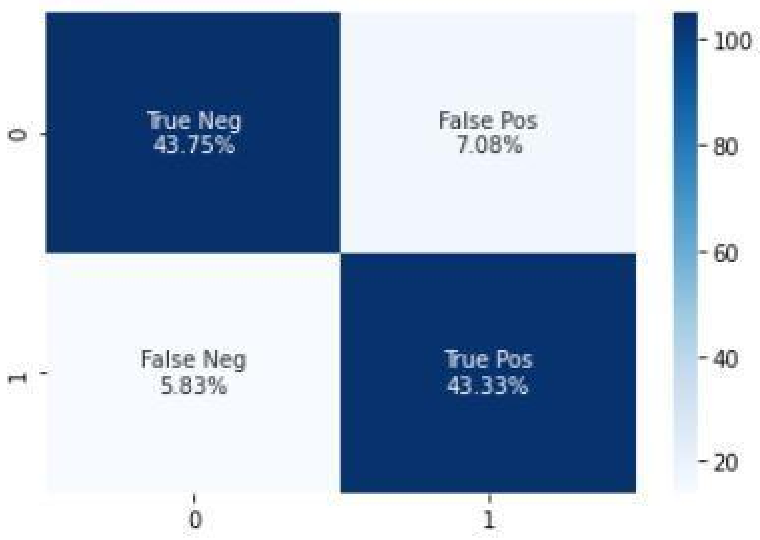
- **Ensemble model**



```
In [47]: #Confusion Matrix
         print(confusion_matrix(y_test,finalpred))

         [[ 91  31]
          [  9 109]]

In [48]: #Classification Report
         target_names = ['class 0(Parasitized)', 'class 1(Uninfected)']
         print(classification_report(y_test, finalpred, target_names=target_names))

                                precision    recall  f1-score   support

         class 0(Parasitized)       0.91      0.75      0.82       122
          class 1(Uninfected)       0.78      0.92      0.84       118

                     accuracy                           0.83       240
                    macro avg       0.84      0.83      0.83       240
                 weighted avg       0.85      0.83      0.83       240
```

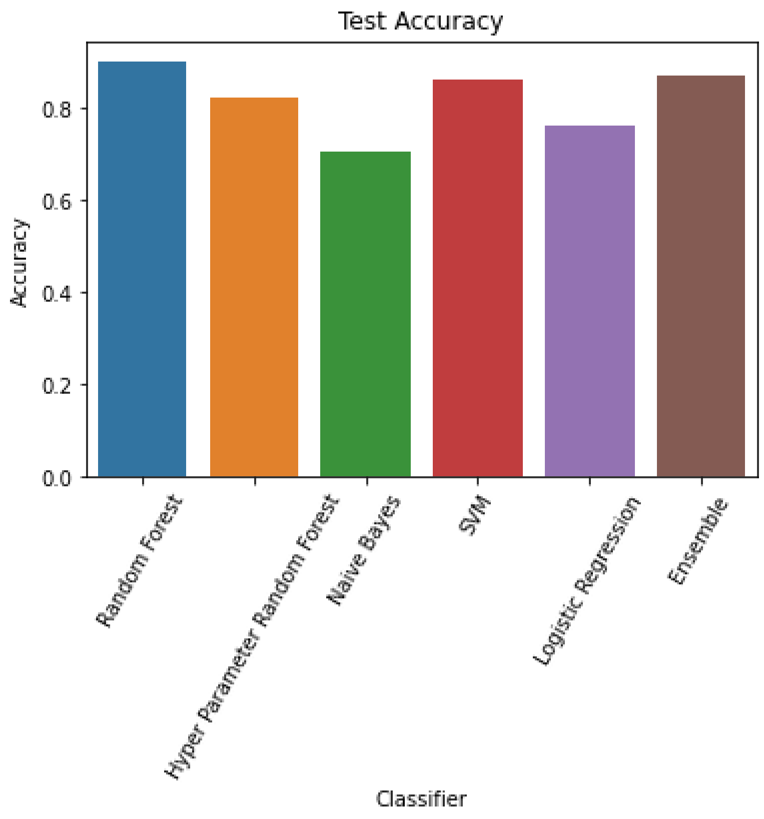
Ensemble Classification Report

```
MaxVote Ensemble

<AxesSubplot:>
```

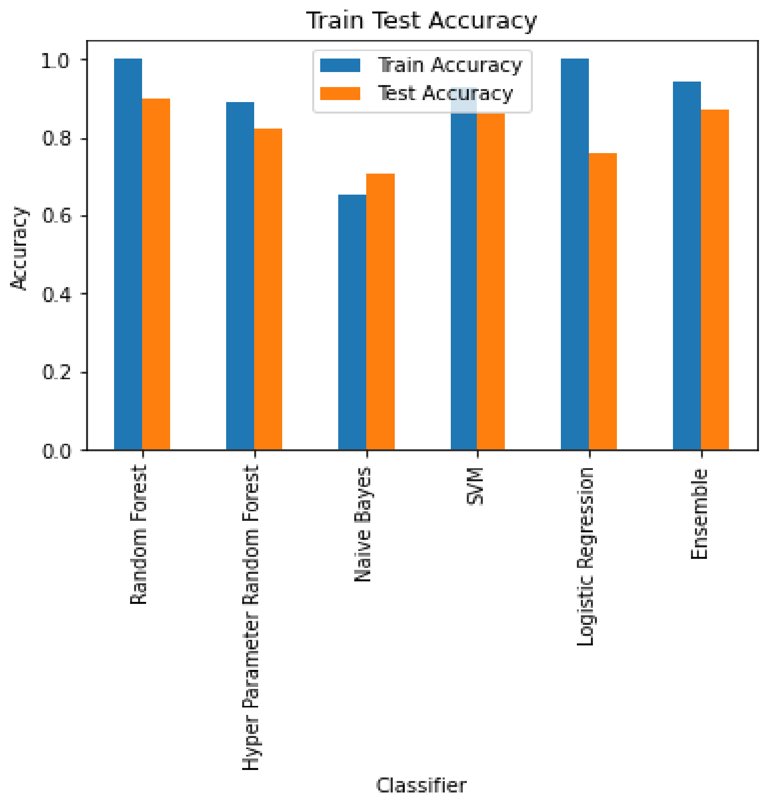


Ensemble Confusion Matrix

Model Accuracy



Test Accuracy



Conclusion

We have implemented 5 different machine learning algorithms Naïve Bayes, SVM, Logistic Regression, Random Forest, Ensemble model to classify malaria cells. Out of all those algorithms, the Ensemble model has achieved best accuracy on the train set and test set as 94% and 87.9% respectively.

References

[1] https://ceb.nlm.nih.gov/repositories/malaria-datasets/

[2] https://www.oreilly.com/library/view/hands-on-machinelearning/9781788992282/d9d74493-e4b0-4592-9515-12188ca60ee5.xhtml

[3] https://scikit-learn.org/stable/modules/svm.html

[4] https://machinelearningmastery.com/logistic-regression-for-machine-learning/

[5] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[6] https://scikit-learn.org/stable/modules/sgd.html

[7]https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/