In [1]:
```
# import random search, random forest, iris data, and distributions
from sklearn.model_selection import cross_validate
from sklearn import datasets
from sklearn.ensemble import RandomForestClassifier
```

In [2]:
```
import pandas as pd
data = pd.read_csv('HaitiPixels_good.csv')
data.head()
```

Out[2]:

	Type	Red	Green	Blue
0	nonblue	104	89	63
1	nonblue	101	80	60
2	nonblue	103	87	69
3	nonblue	107	93	72
4	nonblue	109	99	68

In [3]:
```
from sklearn import datasets
X=data[['Red', 'Green', 'Blue']]  # Features
y=data['Type']  # Labels
X.columns = ['Red','Green','Blue']
y.columns = ['Target']
```

In [4]:
```
# Split dataset into training set and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% t
```

In [5]:
```
#Import Random Forest Model
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)

y_pred=clf.predict(X_test)
```

In [6]:
```
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9998887380671577
```

https://rstudio-pubs-static.s3.amazonaws.com/71575_4068e2e6dc3d46a785ad7886426c37db.html (https://rstudio-pubs-static.s3.amazonaws.com/71575_4068e2e6dc3d46a785ad7886426c37db.html)

In [7]:

```
# Import train_test_split function
from sklearn.model_selection import train_test_split

# Split dataset into features and labels
X=data[['Red', 'Green', 'Blue']]  # Removed feature "sepal length"
y=data['Type']
# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% t
```

In [8]:

```
from sklearn.ensemble import RandomForestClassifier

#Create a Gaussian Classifier
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)

# prediction on test set
y_pred=clf.predict(X_test)

#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.9998887380671577
```

https://medium.com/@hjhuney/implementing-a-random-forest-classification-model-in-python-583891c99652 (https://medium.com/@hjhuney/implementing-a-random-forest-classification-model-in-python-583891c99652)

In [9]:

```
from sklearn import model_selection
# random forest model creation
rfc = RandomForestClassifier()
rfc.fit(X_train,y_train)
# predictions
rfc_predict = rfc.predict(X_test)
```

```
C:\Users\gladi\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: Futu
reWarning: The default value of n_estimators will change from 10 in version 0.2
0 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

In [10]:

```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report, confusion_matrix
```

In [12]:

```
rfc_cv_score = cross_val_score(rfc, X, y, cv=10, scoring='roc_auc')
```

In [13]:

```
print("=== Confusion Matrix ===")
print(confusion_matrix(y_test, rfc_predict))
print('\n')
print("=== Classification Report ===")
print(classification_report(y_test, rfc_predict))
print('\n')
print("=== All AUC Scores ===")
print(rfc_cv_score)
print('\n')
print("=== Mean AUC Score ===")
print("Mean AUC Score - Random Forest: ", rfc_cv_score.mean())
```

```
=== Confusion Matrix ===
[[  2668     13]
 [    30 311862]]


=== Classification Report ===
              precision    recall  f1-score   support

        blue       0.99      1.00      0.99      2681
     nonblue       1.00      1.00      1.00    311892

    accuracy                           1.00    314573
   macro avg       0.99      1.00      1.00    314573
weighted avg       1.00      1.00      1.00    314573



=== All AUC Scores ===
[1.         1.         1.         1.         1.         1.
 0.99999999 0.99999998 0.99999516 0.99866024]


=== Mean AUC Score ===
Mean AUC Score - Random Forest:  0.999865536847714
```

In [ ]:

```
from sklearn.model_selection import RandomizedSearchCV
import numpy as np
# number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 200, stop = 2000, num = 10)]
# number of features at every split
max_features = ['auto','sqrt']

# max depth
max_depth = [int(x) for x in np.linspace(100, 500, num = 11)]
max_depth.append(None)
# create random grid
random_grid = {
 'n_estimators': n_estimators,
 'max_features': max_features,
 'max_depth': max_depth
 }
# Random search of parameters
rfc_random = RandomizedSearchCV(estimator = rfc, param_distributions = random_gr
# Fit the model
rfc_random.fit(X_train, y_train)
# print results
print(rfc_random.best_params_)
```

```
Fitting 3 folds for each of 100 candidates, totalling 300 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
```

In [ ]:

```
rfc = RandomForestClassifier(n_estimators=600, max_depth=300, max_features='sqrt
rfc.fit(X_train,y_train)
rfc_predict = rfc.predict(X_test)
rfc_cv_score = cross_val_score(rfc, X, y, cv=10, scoring='roc_auc')
print("=== Confusion Matrix ===")
print(confusion_matrix(y_test, rfc_predict))
print('\n')
print("=== Classification Report ===")
print(classification_report(y_test, rfc_predict))
print('\n')
print("=== All AUC Scores ===")
print(rfc_cv_score)
print('\n')
print("=== Mean AUC Score ===")
print("Mean AUC Score - Random Forest: ", rfc_cv_score.mean())
```

In [ ]: