Case study of Decision Tree Classification and Random Forest Classification

Oksana Dura
Student ID: 1316268

April 1, 2023

Abstract

Every day hundreds and thousands of messages are being sent all over the world. Some of them are important messages while the rest is marketing or even scam messages. In our phones or emails, our messages are classified for us. In this case study we will learn two methods on how these classifications are made and most important we will be able to study different classification methods and find out their accuracy for a given dataset. In this report, we will be classifying email messages as spam or ham using Decision Tree Classification and Random Forest Classification and studying their classification accuracies.

1 Introduction

For the case study we were given a dataset containing 57 attributes that encode the total number a word or character occurs, and a total of 4601 instances. The dataset classifies email messages as spam or ham. In the given dataset we had to apply the Random Forest Classification and Decision Tree Classification and report the following findings:

- Compare the accuracies of the Random Forest classifier as a function of the number of base learners (e.g., 10, 50, 100, 500, 1000, and 5000) and the number of features to consider at each split (e.g., auto or sqrt)
- Compare the results of all the classifiers (with the best possible parameter setting for each classifier). Use classification accuracy (number of instances correctly classified/total number of instances presented for classification), per class classification accuracy, and confusion matrix to compare the classifiers.

2 Understanding the data

2.1 Getting started with the data

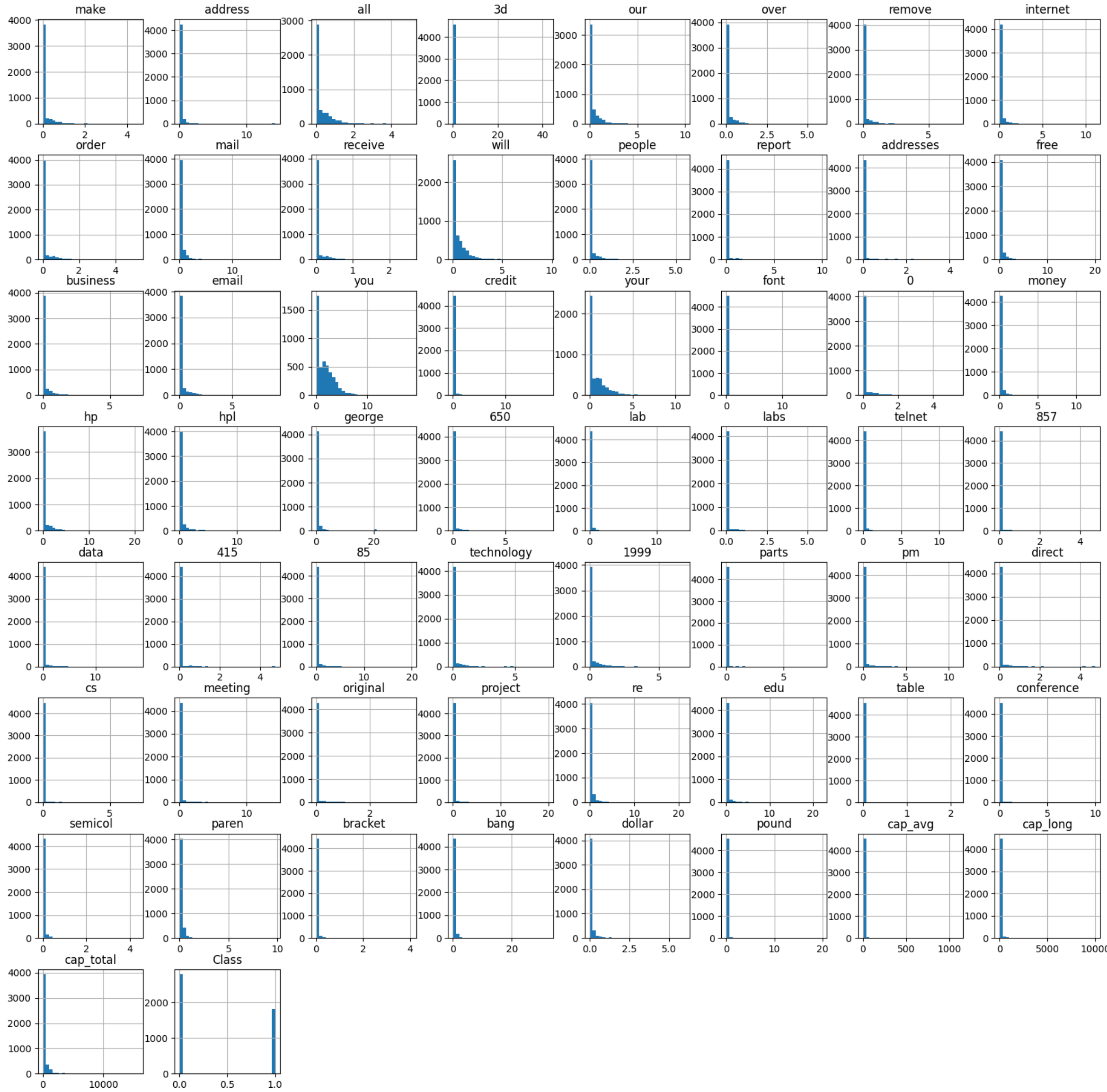
Before implementing the classification methods is a good practice to understand the data you are working with, and make the necessary changes to get the best output. Some of the changes may include cleaning it, checking for missing data, replacing some values, naming the features. All of the preparations of the data depend on the dataset that we were given.

Important elements from the dataset:

- The Datset consists of 57 features 4601 samples.
- The data consists of 55 float type, 2 int type, and 1 object type.
- The object type named "Class" takes to values, either ham or spam.
- There are no missing values in the dataset.

2.2 Visualizing the dataset

The figure below represent per feature histogram



I have assigned 1 to 'spam' and 0 to ham in the dataset, because it makes it easier for me to work with having them as integers. The replacement is seen in the following code:

```
[5] spam_dataset_dataframe['Class']=spam_dataset_dataframe['Class'].apply(lambda x: 1 if x=='spam' else 0)
    spam_dataset_dataframe.head()
```

Here we can see the changes that happened to "Class" column in the dataset. :

	make	address	all	3d	our	over	remove	internet	order	mail	...	semicol	paren	bracket	bang	dollar	pound	cap_avg	cap_long	cap_total	Class
0	0.00	0.00	0.29	0.0	0.00	0.00	0.00	0.00	0.00	0.00	...	0.000	0.178	0.0	0.044	0.000	0.00	1.666	10	180	0
1	0.46	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	...	0.000	0.125	0.0	0.000	0.000	0.00	1.510	10	74	0
2	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.00	0.00	0.00	...	0.000	0.000	0.0	0.000	0.000	0.00	1.718	11	55	0
3	0.33	0.44	0.37	0.0	0.14	0.11	0.00	0.07	0.97	1.16	...	0.006	0.159	0.0	0.069	0.221	0.11	3.426	72	819	1
4	0.00	2.08	0.00	0.0	3.12	0.00	1.04	0.00	0.00	0.00	...	0.000	0.000	0.0	0.263	0.000	0.00	1.428	4	20	1

5 rows × 58 columns

2.3 Splitting the dataset

The dataset was split for training and testing, the respective variables were the following: **spam_training_set, spam_test_set**. The training set contains 1000 instances and the testing set contains 3501 instances. The following figures show the implementation of the slipt method.

```
[8] #Create a training and test set
    spam_training_set, spam_test_set = train_test_split(spam_dataset_dataframe, test_size = 3601
    , random_state = 42)
    spam_dataset_dataframe.keys()
```

```
spam_training_data, spam_training_target = spam_training_set[["make", "address", "all","3d","our", "over", "remove", "internet", "order", "mail", "receive", "will",
spam_test_data, spam_test_target = spam_test_set[["make", "address", "all","3d","our", "over", "remove", "internet", "order", "mail", "receive", "will", "people", "
spam_training_data.head()
```

2.4 Optimizing the parameters for the Random Forest

A random forest has a multitude of parameters that can be changed and adopted to give the best output. For this assignment we are going to concentrate only on two of them, as required from the assignment.

- The first parameter is n_estimators which denotes the number of trees in the forest. For this purpose we will be considering the following list n_estimators = [10, 50, 100, 500, 1000, 5000]. We are going to be iterating through all of the elements of the list n_estimators compare the accuracies and find the optimal value for n_estimators.
- The second parameter that is going to be considered is max_features, that represents he number of features to consider when looking for the best split. There are three possible choices here, max_features = [”sqrt”, ”log2”, None]. We are going to be iterating thorugh each of them, comparing the accuracies and finding the highest accuracy.
  - ”sqrt” represents the square root of n-features.
  - ”log2” represents the logarithm with base 2 of n-features.
  - None means that all features are considered.
- There is a third parameter that is considered for the Random Forest classifier, and that is random_state. Random state is used to reproduce the same result across different run. In both of the classifiers I have assigned random_state = 101.

The following code shows the implementation of the optimization of parameters

```
n_estimators = [10, 50, 100, 500, 1000, 5000]
max_features = ["sqrt", "log2", None]
highest_accuracy = 0
a = 0
b = ""

for x in max_features:
  for n in n_estimators:
    clf1 = RandomForestClassifier(n_estimators = n, max_features = x, random_state = 101)
    clf1.fit(spam_training_data,spam_training_target)
    spam_test_target_predict=clf1.predict(spam_test_data)
    print("For no_estimators = ",n,", and max_fetures = ", x , "the accuracy score is: ", accuracy_score(spam_test_target,spam_test_target_predict))

    if accuracy_score(spam_test_target,spam_test_target_predict) > highest_accuracy:
      highest_accuracy = accuracy_score(spam_test_target,spam_test_target_predict)
      a = n
      b = x

print("The random forest with the highest accuracy ",highest_accuracy, "has the following parameters: n_estimators = ", a, " and max_features = ", b)
```

As

we wrote above the parameter optimization is considering only two of the parameters that Random Forest has. For the first feature n_estimators we had 6 elements, and for the second feature max_features there are 3 elements. Hence to find to best accuracy we have to compare all the possible combinations, in this case we get 18 accuracy values. We need to compare those and get the highest accuracy.

```
For no_estimators =  10 , and max_fetures =  sqrt the accuracy score is:  0.918911413496251
For no_estimators =  50 , and max_fetures =  sqrt the accuracy score is:  0.9266870313801722
For no_estimators =  100 , and max_fetures =  sqrt the accuracy score is:  0.9311302415995557
For no_estimators =  500 , and max_fetures =  sqrt the accuracy score is:  0.9316856428769786
For no_estimators =  1000 , and max_fetures =  sqrt the accuracy score is:  0.9341849486253818
For no_estimators =  5000 , and max_fetures =  sqrt the accuracy score is:  0.9355734518189391
For no_estimators =  10 , and max_fetures =  log2 the accuracy score is:  0.928353235212441
For no_estimators =  50 , and max_fetures =  log2 the accuracy score is:  0.9369619550124966
For no_estimators =  100 , and max_fetures =  log2 the accuracy score is:  0.9411274645931685
For no_estimators =  500 , and max_fetures =  log2 the accuracy score is:  0.9364065537350736
For no_estimators =  1000 , and max_fetures =  log2 the accuracy score is:  0.9358511524576506
For no_estimators =  5000 , and max_fetures =  log2 the accuracy score is:  0.9352957511802277
For no_estimators =  10 , and max_fetures =  None the accuracy score is:  0.9164121077478479
For no_estimators =  50 , and max_fetures =  None the accuracy score is:  0.9236323243543461
For no_estimators =  100 , and max_fetures =  None the accuracy score is:  0.9239100249930575
For no_estimators =  500 , and max_fetures =  None the accuracy score is:  0.9233546237156346
For no_estimators =  1000 , and max_fetures =  None the accuracy score is:  0.9236323243543461
For no_estimators =  5000 , and max_fetures =  None the accuracy score is:  0.9233546237156346
```

After the program is run we see the following parameters produce the highest accuracy:

- random_state = 101 (Is constant)
- n_estimators = 100
- max_features = log2

And the accuracy we get is = 0.9411274645931685

```
The random forest with the highest accuracy  0.9411274645931685 has the following parameters: n_estimators =  100  and max_features =  log2
```

3 Optimizing the parameters for the Decision Tree

The second part of the assignment asks to compare the Decision Tree Classifier with the Random Forest Classifier. Before doing the comparison we need to find the optimal parameters to get the highest accuracy from the Decision Tree Classifier. Same as the random forest, the decision tree has a multitude of parameters that can be changed and adopted to give the best output. For this assignment we are going to concentrate on four of them.

- The first parameter is criterion which measures the quality of a split. For this purpose we will be considering the following criterion_DT = [”gini”, ”entropy”]. We are going to be iterating through all of the elements of the list criterion_DT compare the accuracies and find the optimal value for criterion.
- The second parameter that is going to be considered is max_features, that represents he number of features to consider when looking for the best split. There are three possible choices here, max_features = [”sqrt”, ”log2”, None]. We are going to be iterating thorugh each of them, comparing the accuracies and finding the highest accuracy.
  - ”sqrt” represents the square root of n-features.
  - ”log2” represents the logarithm with base 2 of n-features.
  - None means that all features are considered.
- The third parameter that is considered is splitter which denotes the strategy used to choose the split at each node. The list has two elements as following splitter= [”best”, ”random”].
- the fourth parameter that is considered is max_depth that is the maximum depth of the tree. For this case we are going to be looking at the following list max_depth = [2,4,6,8,10,12].

- There is a fifth parameter that is considered for the Decision Tree classifier, and that is random_state. Random state is used to reproduce the same result across different run. In both of the classifiers I have assigned random_state = 101.

The following code shows the implementation of the optimization of parameters

```
criterion_DT = ["gini", "entropy"]
splitter_DT = ["best", "random"]
max_features_DT = ["sqrt", "log2", None]
max_depth_DT = [2,4,6,8,10,12]
highest_accuracy_DT = 0
f = ""
g = ""
h = ""
y = ""
for y in max_depth_DT:
  for x in max_features_DT:
    for n in criterion_DT:
      for z in splitter_DT:
        clf = DecisionTreeClassifier(criterion = n, max_features = x, splitter= z, max_depth = y,random_state = 101)
        clf.fit(spam_training_data,spam_training_target)
        spam_test_target_predict=clf.predict(spam_test_data)
        print("For criterion_DT = ",n,", and max_features_DT = ", x , "and splitter_DT = ", z, "max_depth_DT = ", y, " the accuracy score is: ", accuracy_score(spam_

        if accuracy_score(spam_test_target,spam_test_target_predict) > highest_accuracy_DT:
          highest_accuracy_DT = accuracy_score(spam_test_target,spam_test_target_predict)
          f = n
          g = x
          h = z
          w = y

print("The decision tree with the highest accuracy ",highest_accuracy_DT, "has the following parameters: criterion_DT = ", f,"max_depth_DT = ",y , " max_features_DT
```

As we wrote above the parameter optimization is considering four of the parameters that Decision Tree has. For the first parameter criterion we had 2 elements, and for the second feature max_features there are 3 elements, for the third parameter we had 6 values and for the fourth parameter we had 2 values. Hence to find to best accuracy we have to compare all the possible combinations, in this case we get 72 accuracy values. We need to compare those and get the highest accuracy.

```
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  2  the accuracy score is:  0.798389336295473
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  2  the accuracy score is:  0.6403776728686476
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  2  the accuracy score is:  0.7925576228825326
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  2  the accuracy score is:  0.6403776728686476
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  2  the accuracy score is:  0.7911691196889753
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  2  the accuracy score is:  0.650097195223549
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  2  the accuracy score is:  0.7942238267148014
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  2  the accuracy score is:  0.650097195223549
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  2  the accuracy score is:  0.8455984448764232
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  2  the accuracy score is:  0.7078589280755345
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  2  the accuracy score is:  0.8455984448764232
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  2  the accuracy score is:  0.7078589280755345
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  4  the accuracy score is:  0.8294918078311581
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  4  the accuracy score is:  0.7073035267981116
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  4  the accuracy score is:  0.8083865592890863
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  4  the accuracy score is:  0.7073035267981116
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  4  the accuracy score is:  0.8136628714246043
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  4  the accuracy score is:  0.6995279089141905
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  4  the accuracy score is:  0.7928353235212441
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  4  the accuracy score is:  0.7064704248819772
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  4  the accuracy score is:  0.8828103304637601
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  4  the accuracy score is:  0.8047764509858373
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  4  the accuracy score is:  0.897528464315468
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  4  the accuracy score is:  0.8044987503471258
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  6  the accuracy score is:  0.8464315467925576
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.7136906414884754
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  6  the accuracy score is:  0.8672590946959178
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.7136906414884754
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  6  the accuracy score is:  0.8411552346570397
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.7289641766176063
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  6  the accuracy score is:  0.8136628714246043
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.7289641766176063
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  6  the accuracy score is:  0.8983615662316023
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.851985559566787
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  6  the accuracy score is:  0.8980838655928909
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.8464315467925576
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  8  the accuracy score is:  0.8633712857539573
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  8  the accuracy score is:  0.8075534573729519
```

```
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.8519855595667
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  6  the accuracy score is:  0.8980838655928
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  6  the accuracy score is:  0.8464315467925576
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  8  the accuracy score is:  0.8633712857539573
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  8  the accuracy score is:  0.8075534573729519
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  8  the accuracy score is:  0.8605942793668425
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  8  the accuracy score is:  0.8044987503471258
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  8  the accuracy score is:  0.8742016106637045
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  8  the accuracy score is:  0.7692307692307693
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  8  the accuracy score is:  0.8600388780894196
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  8  the accuracy score is:  0.7561788392113302
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  8  the accuracy score is:  0.8944737572896417
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  8  the accuracy score is:  0.8675367953346292
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  8  the accuracy score is:  0.9008608719800055
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  8  the accuracy score is:  0.860871980005554
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  10  the accuracy score is:  0.8617050819216884
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  10  the accuracy score is:  0.8078311580116634
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  10  the accuracy score is:  0.8619827825603998
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  10  the accuracy score is:  0.8078311580116634
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  10  the accuracy score is:  0.8625381838378229
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  10  the accuracy score is:  0.7908914190502638
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  10  the accuracy score is:  0.8755901138572618
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  10  the accuracy score is:  0.779783393501805
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  10  the accuracy score is:  0.8925298528186615
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  10  the accuracy score is:  0.8633712857539573
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  10  the accuracy score is:  0.8947514579283532
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  10  the accuracy score is:  0.8480977506248264
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  12  the accuracy score is:  0.8642043876700917
For criterion_DT =  gini , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  12  the accuracy score is:  0.8103304637600667
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  best max_depth_DT =  12  the accuracy score is:  0.8697584004443211
For criterion_DT =  entropy , and max_features_DT =  sqrt and splitter_DT =  random max_depth_DT =  12  the accuracy score is:  0.8156067758955846
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  12  the accuracy score is:  0.8725354068314357
For criterion_DT =  gini , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  12  the accuracy score is:  0.8000555401277423
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  best max_depth_DT =  12  the accuracy score is:  0.8747570119411274
For criterion_DT =  entropy , and max_features_DT =  log2 and splitter_DT =  random max_depth_DT =  12  the accuracy score is:  0.8017217439600111
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  12  the accuracy score is:  0.8916967509025271
For criterion_DT =  gini , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  12  the accuracy score is:  0.8711469036378784
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  best max_depth_DT =  12  the accuracy score is:  0.9016939738961399
For criterion_DT =  entropy , and max_features_DT =  None and splitter_DT =  random max_depth_DT =  12  the accuracy score is:  0.8755901138572618
```

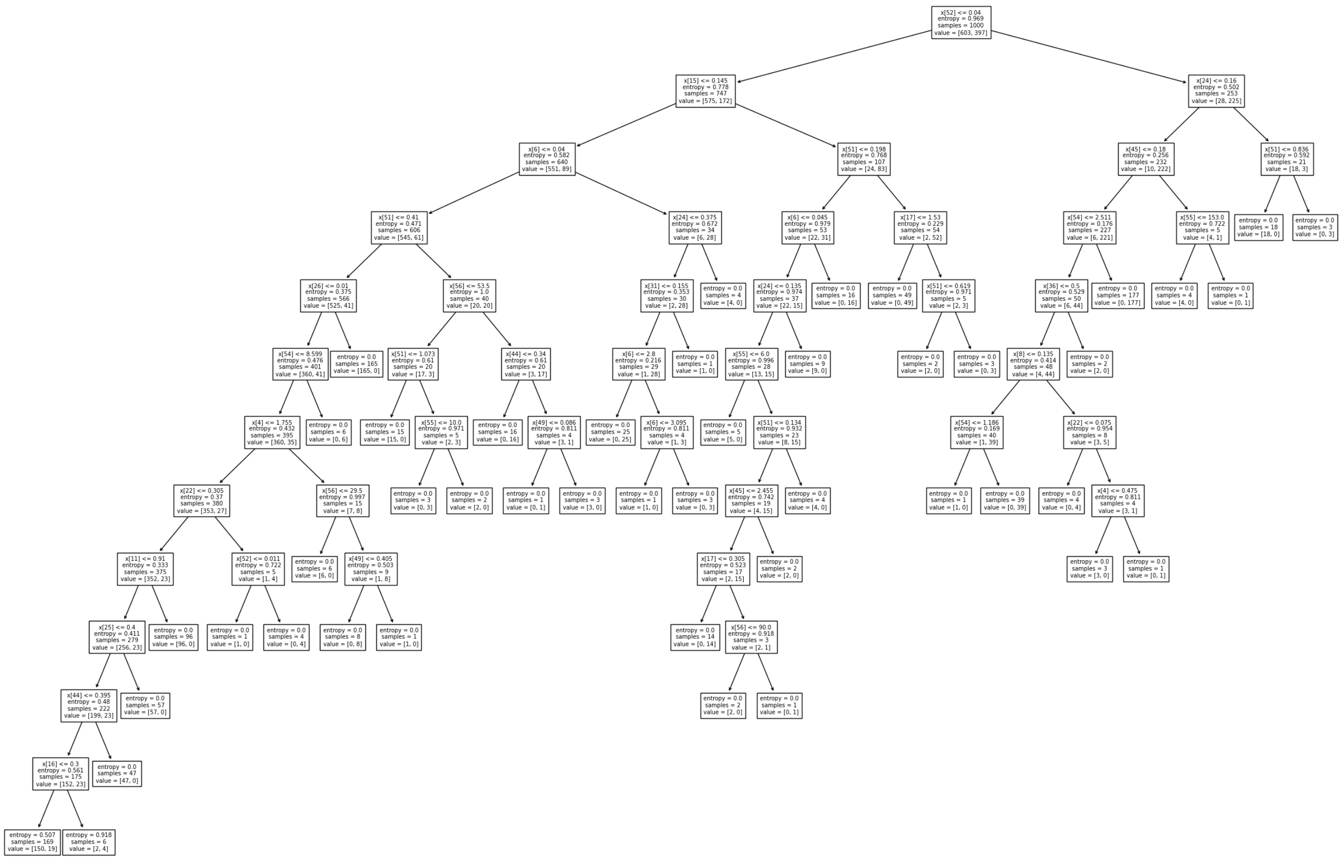
After the program is run we see the following parameters produce the highest accuracy:

- random_state = 101 (Is constant)
- criterion = entropy
- max_depth = 12 max_features = None
- splitter = best
- max_features = log2

And the accuracy we get is = 0.9016939738961399

```
The decision tree with the highest accuracy  0.9016939738961399 has the following parameters:
criterion_DT =  entropy max_depth_DT =  12  max_features_DT =  None splitter_DT =  best
```

"Visualizing the decision tree"



4 Comparing Decision Tree Classifier and Random Forest Classifier

We have previously decided the best parameters to be used for both of the classifiers to get the highest accuracy. Now it is time to compare the results we get from these two classifiers.

Outputting 20 predicted values for Decision Tree

```
Prediction for 20 observation:     [0 0 1 0 1 1 1 0 1 1 0 0 0 0 1 1 0 1 1 1]
Actual values for 20 observation:  [0 0 1 0 0 1 1 0 1 1 0 0 0 0 1 1 0 1 0 1]
```

Outputting 20 predicted values for Random Forest

```
Prediction for 20 observation:     [0 0 1 0 0 1 1 0 1 1 0 0 0 0 1 1 0 1 0 1]
Actual values for 20 observation:  [0 0 1 0 0 1 1 0 1 1 0 0 0 0 1 1 0 1 0 1]
```

The Decision Tree Classifier Outputs the following: Here we need to remember that 0 represents "ham", and 1 represents "spam".

```
              precision    recall  f1-score   support

           0       0.92      0.91      0.92      2185
           1       0.87      0.88      0.88      1416

    accuracy                           0.90      3601
   macro avg       0.90      0.90      0.90      3601
weighted avg       0.90      0.90      0.90      3601

0.9016939738961399
```

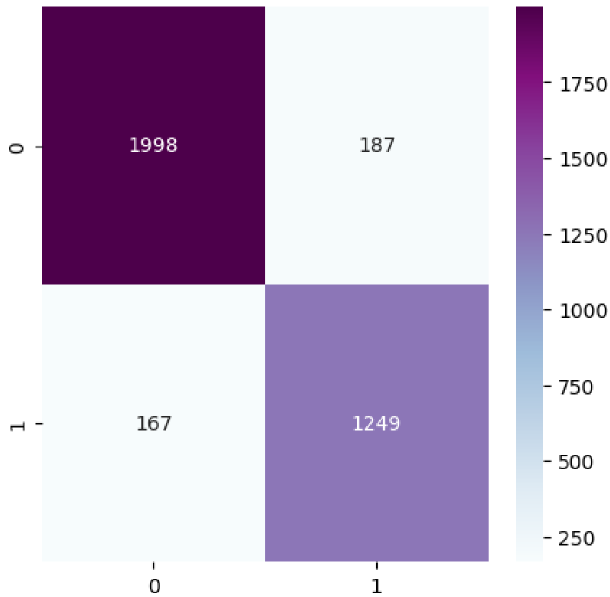
"ham precision"	0.92
"ham precision"	0.87
Accuracy	0.9016939738961399

The Random Forest Classifier Outputs the following: Here we need to remember that 0 represents "ham", and 1 represents "spam".

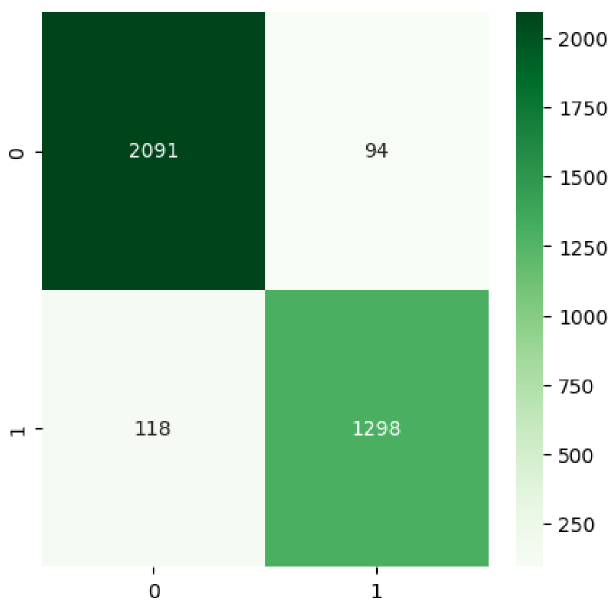
```
[[2091   94]
 [ 118 1298]]
              precision    recall  f1-score   support

           0       0.95      0.96      0.95      2185
           1       0.93      0.92      0.92      1416

    accuracy                           0.94      3601
   macro avg       0.94      0.94      0.94      3601
weighted avg       0.94      0.94      0.94      3601

0.9411274645931685
```

"ham precision"	0.95
"ham precision"	0.93
Accuracy	0.9411274645931685

Confusion Matrix for Decision Tree Classifier



Confusion Matrix for Random Forest Classifier



Conclusion

From the above observation we can reach to the conclusion that:

- Random Forest Classifier has a higher accuracy than Decision Tree.
- Random Forest Classifier has a higher precision than Decision Tree.
- From the Confusion Matrix we can clearly notice that:
  - False Positive of Random Forest $<$ False Positive of Decision Tree
  - False Negative of Random Forest $<$ False Negative of Decision Tree