Random Forest Classifier is chosen to be the model for this project.

Here is few reasons why Random forest is chosen:

1. This is a classification problem, a classifier is needed.
2. Random Forest is known for providing high accuracy in many prediction tasks. It builds multiple decision trees and merges them together to get a more accurate and stable prediction. The ensemble of trees helps to correct the mistakes of individual trees, leading to better performance on complex datasets.
3. While individual decision trees often overfit to the training data, Random Forest mitigates this by averaging multiple trees to reduce overfitting without significantly increasing error due to bias.
4. It can model complex relationships between features and the target variable, including non-linear interactions, without the need for transformation of the features.

The number of trees in a Random Forest is a key hyperparameter that significantly influences the performance of the model. Generally, increasing the number of trees in a Random Forest improves model performance up to a certain point. More trees reduce the variance of the model without significantly increasing bias, which means the model becomes more stable and less sensitive to the specifics of any single tree. Although adding more trees can improve the model, there is a point of diminishing returns where the performance improvement becomes negligible. Beyond this point, adding more trees only contributes to increasing computational cost and training time without substantial gains in accuracy.

So a test of trees was generated to see different number of trees will have what impact on the model results:

```
Number of trees: 200
Train accuracy: 0.9998541139856059
Test accuracy: 0.9952540480178671
Validation accuracy: 0.9939883645765999
-------------------------------------
Number of trees: 500
Train accuracy: 0.9998541139856059
Test accuracy: 0.9955332216638749
Validation accuracy: 0.9940530058177117
-------------------------------------
Number of trees: 1000
Train accuracy: 0.9998541139856059
Test accuracy: 0.9952540480178671
Validation accuracy: 0.9940530058177117
-------------------------------------
Number of trees: 1500
Train accuracy: 0.9998541139856059
Test accuracy: 0.9955332216638749
Validation accuracy: 0.9940530058177117
-------------------------------------
Number of trees: 2000
Train accuracy: 0.9998541139856059
Test accuracy: 0.9955332216638749
Validation accuracy: 0.9940530058177117
-------------------------------------
```

The diminishing return occur after 500 trees, after that point, adding more trees will not have a huge impact on the model itself.

After all consideration, the parameter was set to be:

```
# use 500 as the number of trees and depth of 10
rf = RandomForestClassifier(n_estimators=500, max_depth=10, random_state=42)
rf.fit(X_train, y_train)
✓ 57.7s
```

The result is really good on this model:

```
# print accuracy
y_pred = rf.predict(X_test)
accuracy_score(y_test, y_pred)
print('Accuracy: ', accuracy_score(y_test, y_pred))
[24] ✓ 0.1s
Accuracy:  0.9952540480178671
```

```
# cross validation
cross_val_score(rf, X_train, y_train, cv=5, scoring='accuracy').mean()
print('Cross validation: ', cross_val_score(rf, X_train, y_train, cv=5, scoring='accuracy').mean())

# confusion matrix
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_pred)
print('Confusion matrix: ', confusion_matrix(y_test, y_pred))
[25] ✓ 1m 43.5s
Cross validation:  0.9940510888363372
Confusion matrix:  [[3367    0]
 [  17  198]]
```