**Random Forest:**

**How it Works**

The Random Forest Classifier is an optimized version of decision trees that addresses their limitations, such as overfitting and sensitivity to training data. Rather than relying on a single decision tree, a Random Forest constructs a collection (or "forest") of decision trees, each trained on a different subset of the data.

Its steps are:

1. **Bootstrap Sampling**: The algorithm generates multiple training datasets by randomly sampling from the original dataset withreplacement (bootstrapping). Each of these datasets may contain duplicate samples and will be used to train a separate decision tree.
2. **Feature Randomization**: When constructing each tree, the algorithm selects a random subset of features (typically log2 or sqrt of the total number of features) at each split, rather than using all available features. This introduces additional diversity among the trees and reduces correlation between them.
3. **Tree Building**: Each decision tree is independently trained on its respective bootstrapped dataset using the selected random features.
4. **Prediction and Aggregation**: When making a prediction, the input data is passed through all trees in the forest. Each tree produces a classification result, and the final prediction is determined by the majority outcome, the most common class among all trees.

**Justification for Choosing Random Forest**

I used the Random Forest classifier because it works well with structured data and is especially effective when the relationship between features and the target variable is non-linear or complex. Unlike single decision trees, which are prone to overfitting, Random Forest builds multiple decision trees and combines their outputs, improving accuracy and reducing variance.

The dataset contains a mix of numerical features with varying levels of correlation. This makes Random Forest a suitable choice because it can naturally handle feature interactions and is non-parametric meaning it does not assume feature independence or any specific distribution.

**Model Performance**

**1.** Accuracy

* **Training Accuracy:** 100%
* **Testing Accuracy:** 97.7%

The model demonstrates excellent performance on both training and unseen data. While the perfect training accuracy may hint at slight overfitting, the high testing accuracy suggests that the model still generalizes well.

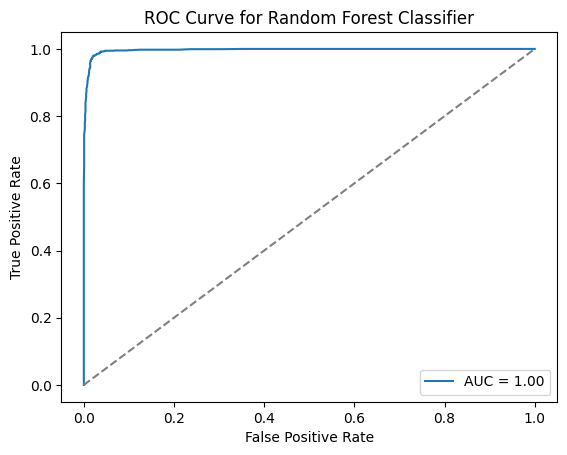
**2.** F1-Score

* **Not Addicted (0):** 0.98
* **Addicted (1):** 0.98

The F1-scores are high and balanced across both classes, indicating that the model effectively manages potential class imbalance and maintains strong precision and recall.

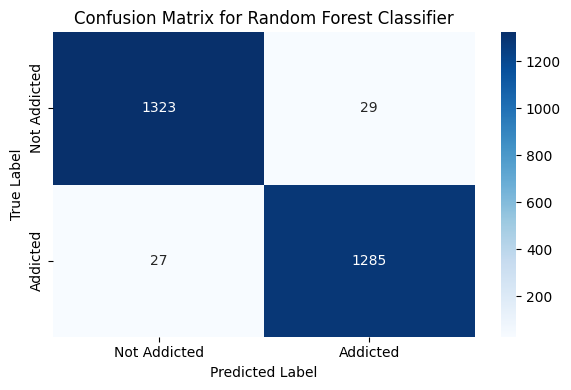
3. AUC-ROC

* **AUC Score:** 1.0

An AUC of 1.0 indicates perfect class separation on the test set (fig 1.1). This may reflect highly distinguishable features in the dataset, though it could also suggest the presence of clean or ideal data that may not fully represent real-world variability.

**Fig 1.1** This plot shows how well the model distinguishes between the two classes. A curve close to the diagonal indicates weak performance.

**Confusion Matrix and ROC Analysis**

The confusion matrix (Figure 1.2) shows that the model made very few classification errors, with 1,320 true negatives and 1,284 true positives. Only 29 "Not Addicted" instances were incorrectly classified as "Addicted," and 27 "Addicted" instances were incorrectly predicted as "Not Addicted." These low misclassification numbers are in line with the high F1-scores for both classes.

**Fig 1.2** Confusion Matrix

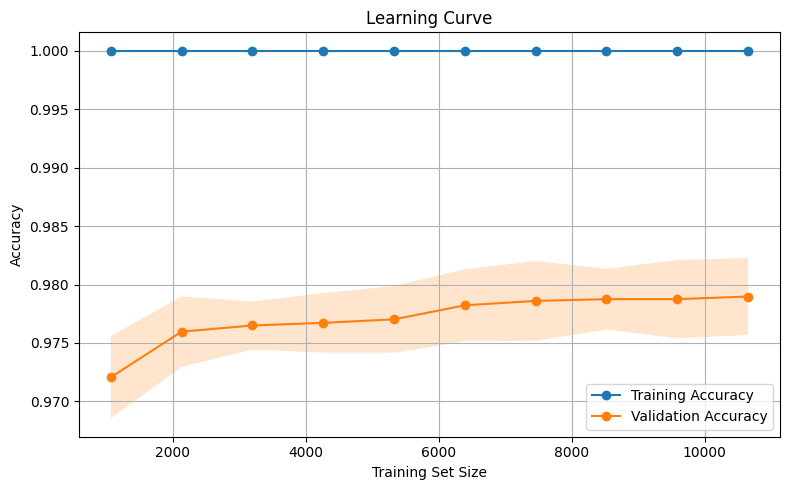
The ROC curve (fig 1.1) further supports this performance, with an AUC score of 1.00. This indicates a perfect balance between sensitivity and specificity, suggesting that the model is extremely effective at distinguishing between addicted and not addicted users.

**Correlation Matrix Analysis**

Random Forest does not rely on feature independence or linear relationships, so it can capture patterns even in the presence of moderate feature correlations. The strong correlations between some features in the dataset and the target likely helped the model learn accurate splits in decision trees.

**Is the Model Overfitting or Underfitting?**

The learning curve (fig 1.3) shows that the training accuracy stays consistently high at 100% across all training set sizes, while the validation accuracy remains stable between 97.8% and 98.0%. The small and steady gap between the two curves suggests only slight overfitting. Despite the perfect training performance, the consistently strong validation accuracy indicates that the model generalizes well to unseen data. This balanced behavior reflects Random Forest’s robustness, as its ensemble structure and use of randomization help prevent severe overfitting and maintain good predictive performance across different data sizes.



**Fig 1.3** Learning Curve

**Why the Accuracy is High**

The high accuracy and strong performance metrics can be attributed to the fact that the features provide a clear separation between the classes, as reflected in the high AUC and F1 scores. Variables such as *notifications* and *app\_sessions* show strong negative correlations with addiction, which helps the model make reliable distinctions. In addition, Random Forest is well-suited for capturing complex, non-linear relationships between features, which likely enhances its predictive power. Finally, the dataset appears to be clean, well-balanced, and structured, all of which contribute to the model’s strong generalization performance.

**Conclusion**

The Random Forest classifier performed exceptionally well on this dataset, with very high accuracy, balanced F1 scores, and perfect AUC. While the training accuracy of 1.00 suggests slight overfitting, the strong and stable validation performance confirms that the model generalizes well. The dataset's informative features, moderate correlations, and numerical nature made Random Forest a strong choice for this problem. Overall, the model achieves a good balance between complexity and generalization, making it well-suited for predicting mobile phone addiction.