Name: Vikrant Singh Jamwal
Student ID: 23104534 **Course:** 1MAI1

ML Assignment 2 – Evaluating Classifiers

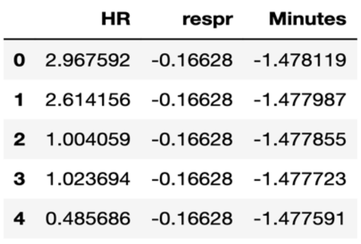
1. Problem Statement and Pre-Processing

This dataset contains participant number, heart rate, respiratory rate, timestamps and labels collected through Empatica E4 (wearable) watch. The objective is to predict whether the participant of the study is stressed (label=1) or not (label=0) using their readings. Python and Jupyter Notebooks are used.

Missing values: There were 44 rows where 'HR' value were NaN, these rows are dropped than imputing a mean or median value of 'HR' as the mean will include heart rates of different participants, which might make these rows outliers.

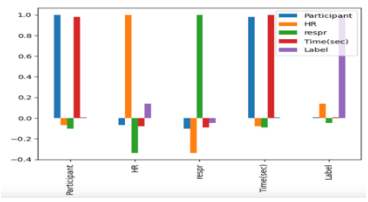
Time formatting: In the dataset, Time format was UNIX, it was converted to Date and Time format and hence date was dropped. Then the Time is converted to minutes of that date.

Feature Scaling: As Random forest and Decision trees are used, which works on if-else conditions making them independent of scales of features. Without scaling the features, distance based algorithms like KNN will give Bias to larger values in the dataset. We have used Standard Scaler feature of sklearn's pre-processing library.



	HR	respr	Minutes
0	2.967592	-0.16628	-1.478119
1	2.614156	-0.16628	-1.477987
2	1.004059	-0.16628	-1.477855
3	1.023694	-0.16628	-1.477723
4	0.485686	-0.16628	-1.477591

Feature Selection: Plotting correlation bar graph of each feature. We found Participant as a redundant feature as it was strongly correlated with Time feature, hence dropped participant from Independent/Input variables. Time was an interacting feature as alone it was weakly correlated with dependent variable but with other features gives good contribution.



2. Splitting the dataset

Used the train_test_split() to split the dataset. Also used **stratify** argument for dependent variable so that data will split with same proportion of categories in test and train as in the complete dataset to avoid data imbalance after splitting.

2.1. Splitting dataset into Training, validation and Test set

Using train_test_split() with test size as 0.1 or 10%, dataset is divided into train and test sets. We get train_X, train_y and test_X, test_y.

```
train_y.value_counts()

0    68091
1    33133
Name: Label, dtype: int64
```

```
test_y.value_counts()

0    7566
1    3682
Name: Label, dtype: int64
```

Divided the above created Train data into two parts using train_test_split(), Training set (90%) and Validation set (10%). We get training_X, training_y and val_X, val_y.

```
training_y.value_counts()

0    61281
1    29820
Name: Label, dtype: int64
```

```
val_y.value_counts()

0    6810
1    3313
Name: Label, dtype: int64
```

2.2. Evaluating classifiers on training and validation set

2.2.1. Accuracy and Classification report

Both classifiers (Decision Tree and Random Forest) are fitted on training set and predicted on validation set. Accuracy of Random forest (99.04) came out to be better than accuracy of Decision Tree (97.12).

```
Decision Tree
Training score: 98.23%
Accuracy: 97.12%
Classification Report:
               precision    recall  f1-score   support

           0       0.99      0.97      0.98      6898
           1       0.94      0.97      0.96      3225

    accuracy                           0.97     10123
   macro avg       0.96      0.97      0.97     10123
weighted avg       0.97      0.97      0.97     10123
```

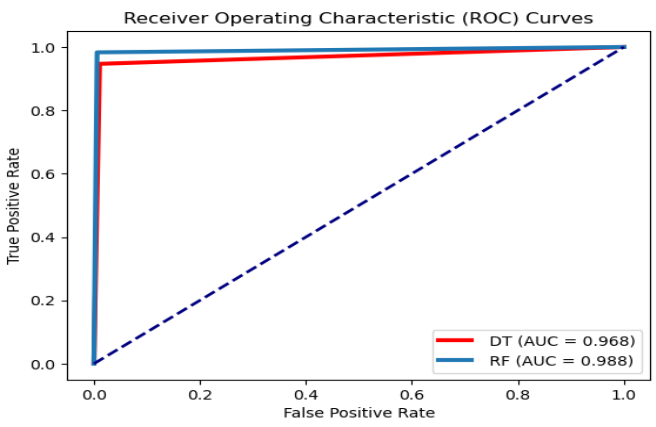
```
Random Forest
Training score: 99.88
Accuracy: 99.04
Classification Report:
               precision    recall  f1-score   support

           0       0.99      0.99      0.99      6819
           1       0.98      0.99      0.99      3304

    accuracy                           0.99     10123
   macro avg       0.99      0.99      0.99     10123
weighted avg       0.99      0.99      0.99     10123
```

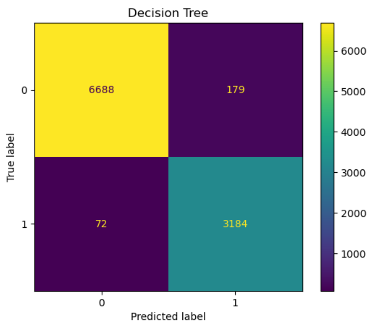
2.2.2. ROC curve and Area Under ROC curve (AUROC)

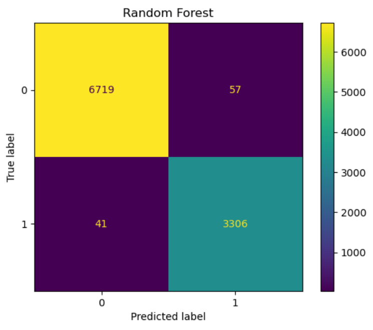
ROC curve, which visualises the relation between True positive rate with False Positive rate with decreasing threshold, is drawn for both the classifiers. Area under the curve (AUROC score) for Random forest (0.988) came out to be better than Decision Tree (0.968).



2.2.3. Confusion Matrix

Confusion matrix is also plotted for both the classifiers and we can see False Positive and False negative of Random forest is less than Decision tree. Hence, Random Forest predicted more correctly than Decision Tree





3. Paired T-Test

The Paired T-test is used to determine whether there is a significant difference in the performance of two classifiers when evaluated on the same data.

3.1. Creating a list of Accuracies

Iterated splitting of train data into training and validation data, calculating accuracy score each time and storing it in a list.

Lists of accuracies

```
print(f" Decision Tree Accuracies:\n {decision_tree_accuracy}") # Viewing the list of accuracies of Decision tree

print(f" Random Forest Accuracies:\n {random_forest_accuracy}") # Viewing the list of accuracies of Random forest
```

```
 Decision Tree Accuracies:
 [97.56001185419342, 98.2218709868616, 98.3305344265534, 97.64891830485034, 97.68843228292008]
 Random Forest Accuracies:
 [99.03190753729132, 98.86397313049491, 99.06154302084363, 98.93312259211696, 99.07142151536105]
```

Mean of accuracies

```
print(f"Mean Accuracy of Decision Tree: {np.mean(decision_tree_accuracy) :.3f}%")

print(f"Mean Accuracy of Random Forest: {np.mean(random_forest_accuracy) :.3f}%")
```

```
Mean Accuracy of Decision Tree: 97.890%
Mean Accuracy of Random Forest: 98.992%
```

3.2. Applying Paired T-test

The Paired T-test is used to determine whether there is a significant difference in the performance of two classifiers when evaluated on the same data. We are using the same training data and the same validation data to evaluate. Hence, Paired T-test will be suitable for comparing the classifiers' performance.

Threshold value (**alpha**) is taken as **0.05 / 5%**

T-stat value

```
print(f"T-statistics value: {t_stat}")
```

```
T-statistics value: -6.374957531980566
```

P-Value (Significance statistics)

```
print(f"P-value: {p_value}")
```

```
P-value: 0.0031056889249266796
```

3.3. Comparing P-value with alpha value

As p-value is less than the threshold, we can say that there is a significant difference between them as we can reject the null hypothesis. As there is a significant difference, we can compare their mean accuracies to determine which is better.

```
As p_value < alpha, we reject null hypothesis.
Hence, there is a SIGNIFICANT DIFFERENCE
Random Forest is better than Decision Tree
```

4. K-Fold (K=10) Cross Validation

- Cross validation is a technique to estimate how well the model will perform on unseen data. K-fold cross validation divides the dataset into K equal folds. In each iteration, each fold is taken as a testing set and remaining folds are taken as training set.

- Stratified K-Fold is used as stratify will make the distribution of data in each fold similar to the distribution in the complete dataset. This will avoid bias over unbalanced folds that may arise due to uneven distribution. Average metrics of these iterations are used as a single representative measure to compare. Both mean accuracy and standard deviation of all folds are calculated for each classifier.

Mean and Standard deviation for 10 folds of each classifier

```
print(f"Cross-validation mean_accuracy of Decision Tree: { cross_val_score_dt.mean()*100 :.2f}%")
print(f"Cross-validation mean_accuracy of Random Forest: { cross_val_score_rf.mean()*100 :.2f}%")
```

```
Cross-validation mean_accuracy of Decision Tree: 97.69%
Cross-validation mean_accuracy of Random Forest: 99.07%
```

```
print(f"Cross-validation standard deviation of Decision Tree: { cross_val_score_dt.std()}")
print(f"Cross-validation standard deviation of Random Forest: { cross_val_score_rf.std()}")
```

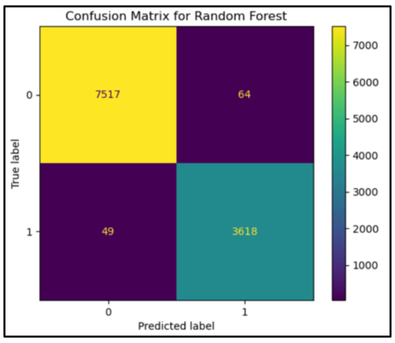
```
Cross-validation standard deviation of Decision Tree: 0.007658166588030069
Cross-validation standard deviation of Random Forest: 0.00131318926962472
```

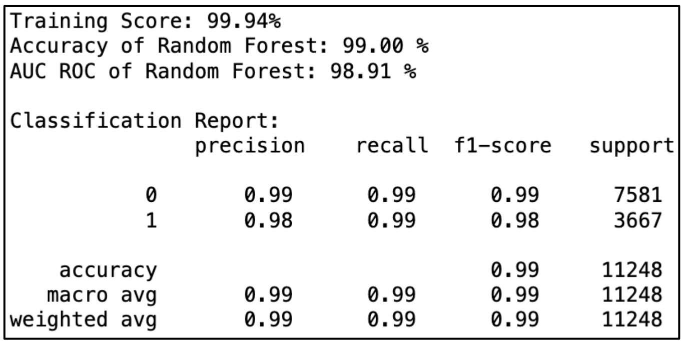
Conclusion

- As **Random forest (99.07%)** has greater mean accuracy over **Decision Tree (97.69%)**, we can conclude that random forest performs better than Decision Tree for this particular Dataset.
- As Standard Deviation of **Random Forest (~0.0013)** is lower than that of **Decision Tree(~0.0076)**, we can say that **Random Forest is more stable and less likely to change with random division of data, making it more robust.**

5. Random Forest evaluation

- Created Random forest classifier with controlled depth and trees. (Maximum Depth=25, Maximum Trees in the forest =25).
- Fitted the model on Train data and predicted the "Label values" on unseen Test dataset.
- Calculated Accuracy, Classification report, ROC curve, AUROC, Confusion matrix and Training score.



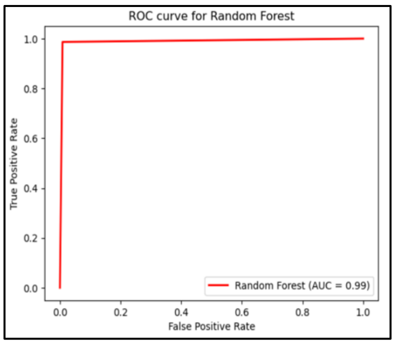


```
Training Score: 99.94%
Accuracy of Random Forest: 99.00 %
AUC ROC of Random Forest: 98.91 %

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      7581
           1       0.98      0.99      0.98      3667

    accuracy                           0.99     11248
   macro avg       0.99      0.99      0.99     11248
weighted avg       0.99      0.99      0.99     11248
```


ROC curve for Random Forest
True Positive Rate
False Positive Rate
Random Forest (AUC = 0.99)