# CA4015 AutoML Framework vs Ensemble Methods in Bio-Chemistry

Seán Cummins
18404744
sean.cummins26@mail.dcu.ie

October 2021

**Abstract**

Feature selection, feature preprocessing, feature construction, model selection, and hyperparameter optimisation are amongst the most tedious and time-consuming tasks for data scientists developing Machine Learning (ML) frameworks. Researchers often resort to inefficient 'brute-force search' for pipeline optimisation. Automated Machine Learning (AutoML) frameworks aim to automate these processes, reducing time to completion drastically while achieving greater results. In this research, we will contrast and compare the results of two traditional ML ensemble frameworks with an AutoML framework on a common classification task. We will be using bacterial Volatilome data to predict species and strain-level specificity. Our findings are two fold; we show stability and predictability in strain-level differences across varying growth media. We also show the improved performance and efficiency of AutoML frameworks over traditional ML pipelines.

## 1 Introduction

In the following section we will discuss the current problems in ML that warrant the development of AutoML frameworks as well as highlight some of the key benefits they bring. We will also introduce the dataset provided by Fitzgerald et al. (2021), that will be used in the comparison of machine learning frameworks. We will supply some background information about the dataset also.

### 1.1 AutoML Problem Motivation & Benefits

It is a challenging and time-consuming task to build well-performing machine learning applications, often requiring highly specialized data scientists. The widespread attention that ML has received in today's literature necessitates the development of an easy to use automated framework known as AutoML. AutoML aims to simplify the process of developing a ML framework, allowing non-experts to take advantage of ML (Kirikkayis, 2021; Nagarajah and Poravi, 2019).

The lack of experts in the field makes Artificial Intelligence (AI) development a long and strenuous procedure. The infinite possibilities for hyperparameter tuning consumes a considerable amount of time for data scientists. Off-the-shelf solutions that can be used by even novice developers to any given use case is an crucial step in the growth of AI (Nagarajah and Poravi, 2019).

### 1.2 Microbial Volatilomic Data

The study of microbial volatilomics is growing rapidly and has shown great potential for application across a range of disciplines such as agriculture, food, health, etc. The strains of micro-organisms investigated by Fitzgerald et al. (2021) are responsible for severe infections in diabetic foot ulcers (DFUs.) The severity and duration of the DFU is closely associated with species and strain-level microbial diversity in the infection (Abdulrazak et al., 2005). Rapid, non-invasive techniques for pathogenic bacterial volatiles in wounds could increase patient turnover.

Microbial volatilomics is the study of emissions produced by various strains of bacteria across different growth medias. Fitzgerald et al. (2021)'s particular study focuses on multiple strains of *Staphylococcus aureus*, *Pseudomonas aeruginosa*, and *Escherichia coli* across three common growth medias. Gas chromatography mass spectrometry (GC-MS) is used to analyze the volatilome of each strain. We would like to invetigate these volatilomes in various medias regarding the stability of their presence. Essentially, we are trying to predict the strain and growth media of bacteria from compounds present in the volatilomic data.

## 2 Materials and Methods

This sections provides background into the two Ensemble ML pipelines used to try and predict strain and growth media as described previously. We will also further discuss TPOT (Le et al., 2020). Finally, we will discuss the scoring system used to compare all 3 ML models.

### 2.1 Random Forest

Both Amit and Geman (1994) and Ho (1995) presented the idea for Random Forest ensemble learners around roughly the same time, before it was popularised by Breiman (2001), a publication which has received over 80,000 citations. Random Forest remains one of the most popular ensemble methods today because of its predictive power and simplicity. As well as this, it has few hyperparamaters, all of which are intuitive and easily interpreted (Sagi and Rokach, 2018). Random Forest differ from traditional decision trees because they use a 'random subspace method'. This means that rather than choosing the best split at each node, a random subset of features are selected and the best split is chosen from them only. Random Forest also uses Bootstrapping[1] when training on a sample of data. This results in higher diversity in trees produced, thus leading to better generalizations (Ho, 1995; Breiman, 2001; Sagi and Rokach, 2018).

### 2.2 Extremely Randomised Trees

Extremely Randomised Trees (Extra Trees), coined by Geurts et al. (2006), is an algorithm that shows improvement over Random Forests in terms of efficiency, as well as generalisation performance. It injects some randomness in the training process by not only adopting the 'random subspace method' mentioned above, but also randomising the splitting features cut-points (Sagi and Rokach, 2018). Unlike Random Forest, Extra Trees learns from the full original learning sample rather than using a Bootstrap method. The motivation behind this is in order of minimising bias (Geurts et al., 2006; Sagi and Rokach, 2018). Geurts et al. (2006) show Extra Trees does have

[1] Bootstrapping is sampling training data with replacement. This encourages diversity in the trees, which reduces variance.

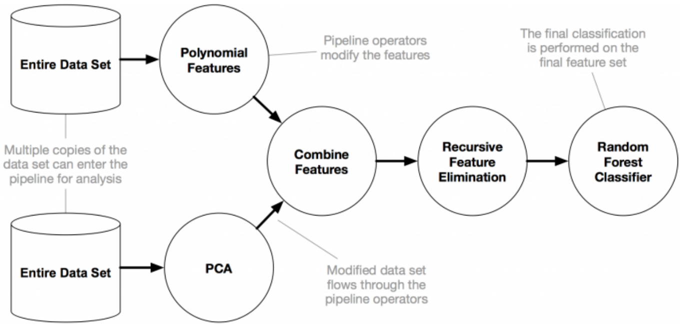

Entire Data Set
Polynomial Features
Pipeline operators modify the features
The final classification is performed on the final feature set
Multiple copies of the data set can enter the pipeline for analysis
Combine Features
Recursive Feature Elimination
Random Forest Classifier
Entire Data Set
PCA
Modified data set flows through the pipeline operators

Figure 1: An example tree-based pipeline from TPOT. (Olson and Moore, 2016)

components of high bias and variance, but this can be counteracted by sufficiently large collections of trees, resulting in better performance over Random Forest.

## 2.3 TPOT: Tree-based Pipeline Optimisation Tool

TPOT is a tool designed to automate the creation of ML pipelines such that there is no need for human intervention (Olson and Moore, 2016). TPOT uses a version of genetic programming (GP), a well-known technique for the automatic construction of programs. TPOT focuses on classification and regression problems and contains three main operators, *Feature Preprocessing Operators*, *Feature Selection Operators*, and *Supervised Classification Operators* (Olson and Moore, 2016; Nagarajah and Poravi, 2019). An example TPOT pipeline can be seen in 1.

## 2.4 K-Fold Cross Validation

In order to assess and compare our pipelines, we will be using 10-fold cross validation. Cross validation is a statistical method used to evaluate learning algorithms. It divides the data into two segments: *learning*, and *training*, with the former used to train the model and the latter used to validate the model. The $k$ represents the number of times the data is equally partitioned into $k$-folds. In an example using 10-fold Cross Validation, 9 folds are used for training and 1 fold is used for validation. This is repeated 10 times with each partition acting as the validation set once (Refaeilzadeh et al., 2009). In our study, we will be using SciKit-learn's KFold[2] implementation to assess our pipelines. The metric we will be using for comparison is 'accuracy', a commonly used measure in classification tasks (Hossin and Sulaiman, 2015).

# 3 Analysis

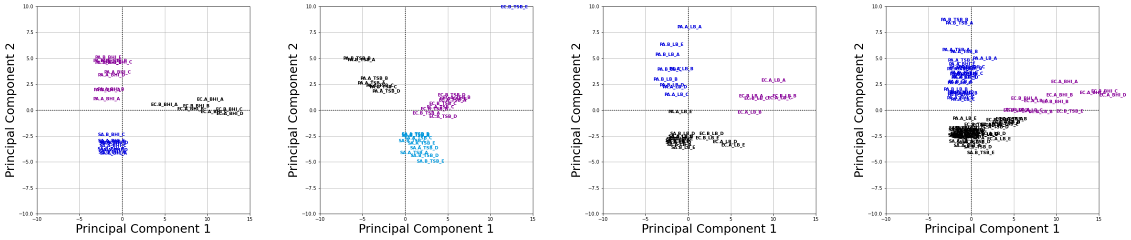
In the following section, we will begin with preliminary data analysis proving that unique strains can be successfully clustered across various growth media as shown in (Fitzgerald et al., 2021). We will then discuss the configuration of each machine learning pipeline individually. In Section 4, we will compare and discuss the results achieved. It should be noted that the implementation of Sci-kit

[2] Find more information about SciKit-learn's KFold API at https://scikit-learn.org/stable/modules/generated/sklearn.model$_{s}election.KFold.html$

learn used throughout this research does not support the use of ’missing’ data. (Fitzgerald et al., 2021) describes that missing values and 0 values carry different meanings in GCMS readings. In this case, we must unfortunately ignore these separate meanings and fill missing values with 0.

## 3.1 Preliminary Data Analysis

To prove that our data is indeed clusterable, we will be repeating some analysis previously performed by Fitzgerald et al. (2021). We will separate our data by growth media and use Principal Component Analysis (PCA) for dimensionality reduction, allowing us to plot on a 2-dimensional plane using hierarchical clustering. The results, which are visualised in Figure 2 show that the presence of core volatile organic compounds remains stable across different nutrient-rich media. The clear separation and successful clustering of strains confirms that the presence of strain-dependent volatilomes is not static noise and is indeed predictable.



(a)

(b)

(c)

(d)

Figure 2: Bacteria in different growth media: (a) BHI (b) TSB (c) LB (d) All Media

## 3.2 Random Forest

Through trial and error or ’brute-force search’, we are able to tune the hyperparameters of the Random Forest ensemble to produce the highest accuracy. The results can be seen in Table 2. For the portion of sample size to be included, usually denoted `max_samples`, 100% inclusion achieved the highest average mean. This is standard practice. For the number of trees produced, usually denoted `n_estimators`, we see that the accuracy achieved plateaus beyond 100 trees, while the elapse time increases heavily for each increment. General practice for the selection of the random subspace, denoted `max_features`, is: $\sqrt{N}$ where $N$ is the feature space, in our case 66. Therefore we expect to achieve the best results for the mean around a subspace of 8. This behaviour is exhibited by the pipeline and can be visualised in Figure 3. The gain in accuracy plateaus beyond this point and only elapse time increases.

## 3.3 Extremely Randomised Trees

We adopt a similar approach to hyperparameter tuning as in Random Forest, Section 3.2. The results can be seen in Table 3. For the number of trees, once again we see accuracy plateau beyond 100. In this instance, 100 trees achieved the highest mean accuracy and lowest elapse time. The

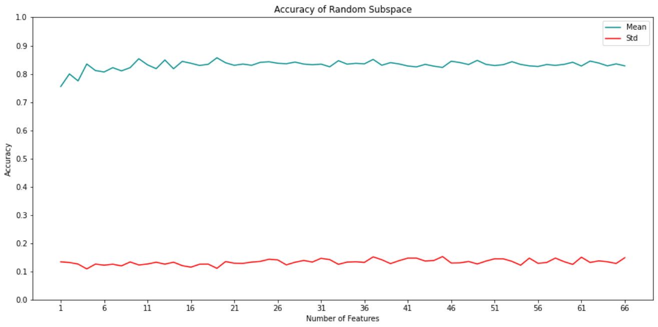


Figure 3: Accuracy of Varying Random Subspace in Random Forest

random subspace sample is as before, with accuracy achieved stagnating beyond roughly 10. In the Table 3, it seems that including 11 features achieves considerably lower accuracy than some of the other measures such as 44 features or all features. However, it should be noted that this is due to the stochastic nature of the algorithm. We can better visualise the mean accuracy in Figure 4. Varying the number of minimum splits, usually denoted `min_samples_split`, saw slight increase in mean accuracy from values of 2 until 5, after which a decline in accuracy and increase in standard deviation is observed.

## 3.4 TPOT

When instantiating our TPOT Classifier, we use `config_dict=’TPOT sparse’` which allows TPOT to work with missing values, a feature that is currently not supported by Scikit-learn. We set our scoring metric to ’accuracy’ and use 10-fold cross validation. Our final configuration is:

```
tpot = TPOTClassifier(scoring = ’accuracy’,
                      max_time_mins = 300,
                      config_dict=’TPOT sparse’,
                      n_jobs=-1,
                      verbosity = 2,
                      cv=10)
```

TPOT ran for 5 hours, after which it produced its best scoring pipeline (out of 33 generated) in terms of accuracy:

```
XGBClassifier(LinearSVC(BernoulliNB(KNeighborsClassifier(input_matrix, n_neighbors=8,
p=2, weights=uniform),
alpha=0.001, fit_prior=False),
C=15.0, dual=False, loss=squared_hinge, penalty=l2, tol=0.001),
learning_rate=1.0, max_depth=6, min_child_weight=1, n_estimators=100, n_jobs=1,
subsample=0.75, verbosity=0)
```

# 4 Results

Here we will present the final results for our three pipelines. Both the Random Forest and Extra Trees were trained on our data with optimal hyperparameters and evaluated using cross validation as before.Each algorithm's score is the average score achieved after 50 iterations. The scores are presented in Table 1. The results are discussed further in Section 5.

| Pipeline | Mean Accuracy | Mean Std |
|---|---|---|
| Random Forest | 0.792 | 0.116 |
| Extra Trees | 0.793 | 0.138 |
| TPOT | **0.932** | Null |

Table 1: Results for each pipeline.

# 5 Discussion

It is not possible to compute the standard deviation in TPOT. This means in Table 1 there is a missing value for TPOT. The results show that TPOT achieves considerably greater accuracy than both Random Forest and Extra Trees. Of the two latter, Extra Trees shows marginal improvement over Random Forest in terms of mean accuracy, but also exhibits an increase in standard deviation. Although Extra Trees shows marginal improvement over Random Forest, the computational speed is considerably greater, completing 50 iterations in a fraction of the time taken by Random Forest.

TPOT was able to create an advanced ML pipeline in the space of 5 hours which, in comparison, could take a team of specialists weeks. The preparation steps for configuring and running TPOT are trivial as the user is required to only identify the training labels and features. On the other hand, setting up an environment for hyperparameter tuning in Random Forest and Extra Trees is more difficult and time consuming.

It should also be noted that in this particular instance TPOT was stopped early and only completed 33 out of 100 generations. It is likely that TPOT would have found further optimisation

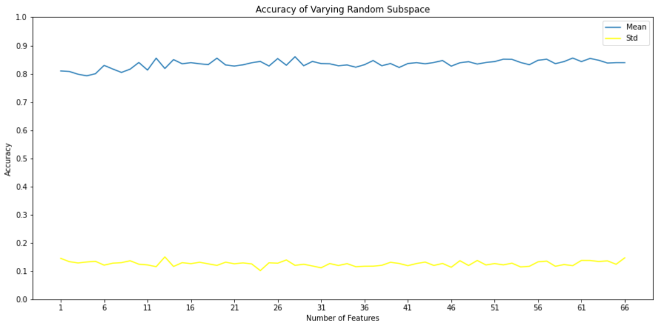


Figure 4: Accuracy of Varying Random Subspace in Extra Trees

if allowed to finish.

In the future, with the availability of more VOC data, it would be interesting to test the performance of TPOT generated pipelines with respect to classification of strains and media.

# 6 Conclusion

The findings of this research are two fold, we conclude that bacterial volatilomes are indeed influenced by nutritional environments and strain-level differences. The success of both traditional ensemble and AutoML methods in predicting strain and media dependencies proves the predictable nature of VOC's within this dataset.

As well as this, our preliminary data analysis influenced by  Fitzgerald et al. (2021), show clear separation of different strains with respect to varying growth media. Variation between *P.aeruginosa* and *S.aureus* can be summarised by the second principal component (y-axis), while the variation between these two strains and *E.coli* is represented by the first principal component (x-axis). In all three media, *S.aureus* shows a high degree of stability, remaining tightly clustered near the origin in the third quadrant. The 2 remaining strains exhibit higher variability across the 3 growth medias, with *E.coli* appearing the least stable.

Secondly, we were able to demonstrate the increased utility that AutoML frameworks provide over traditional pipeline development. Automating the development of a ML pipeline negates the need for tedious and difficult 'brute-force-search' when selecting and constructing features, selecting a model, and tuning hyperparameters. AutoML frameworks are able to construct, optimise, and test a considerably greater amount of frameworks than traditional pipeline development in a fraction of the time. In practice, the use of AutoML should allow researchers to spend more time focusing on tasks such as transparency and explainability in AI, both of which are becoming increasingly important topics. AutoML frameworks are an important and useful tool for machine learning researches and practitioners and show clear benefits. The availability of these frameworks is a crucial step forward for the progress of AI and machine learning.

| Hyperparameter | Measure | Mean Accuracy | Std | Elapse Time (seconds) |
|---|---|---|---|---|
| Sample Size | 20% | 0.710 | 0.142 | 0.706 |
| | 40% | 0.767 | 0.141 | 0.600 |
| | 60% | 0.801 | 0.136 | 0.569 |
| | 80% | 0.811 | 0.132 | 0.560 |
| | 100% | **0.823** | **0.120** | **0.553** |
| Random Subspace | 1 | 0.755 | 0.134 | **0.515** |
| | 11 | 0.831 | **0.126** | 0.535 |
| | 22 | 0.835 | 0.129 | 0.561 |
| | 33 | **0.847** | 0.142 | 0.575 |
| | 44 | 0.827 | 0.139 | 0.594 |
| | 55 | 0.828 | 0.147 | 0.615 |
| | 66 (All) | 0.828 | 0.149 | 0.640 |
| Number of Trees | 10 | 0.743 | 0.129 | **0.249** |
| | 100 | **0.801** | 0.114 | 0.867 |
| | 200 | 0.800 | **0.110** | 1.628 |
| | 350 | 0.796 | 0.112 | 2.800 |
| | 500 | **0.801** | 0.115 | 4.050 |

Table 2: Analysis of different values for hyperparameters in Random Forest.

| Hyperparameter | Measure | Mean Accuracy | Std | Elapse Time (seconds) |
|---|---|---|---|---|
| Number of Trees | 10 | 0.777 | 0.149 | 0.514 |
| | 100 | **0.846** | 0.129 | **0.380** |
| | 200 | 0.819 | 0.126 | 0.507 |
| | 350 | 0.814 | **0.124** | 0.757 |
| | 500 | 0.830 | 0.125 | 0.892 |
| Random Subspace | 1 | 0.810 | 0.145 | **0.407** |
| | 11 | 0.813 | 0.122 | 0.420 |
| | 22 | 0.831 | 0.129 | 0.428 |
| | 33 | 0.828 | 0.120 | 0.439 |
| | 44 | **0.840** | 0.120 | 0.451 |
| | 55 | 0.832 | **0.117** | 0.463 |
| | 66 (All) | **0.840** | 0.147 | 0.474 |
| Minimum Number of Splits | 2 | 0.825 | 0.126 | 0.968 |
| | 5 | **0.836** | 0.131 | 0.579 |
| | 8 | 0.831 | **0.115** | 0.519 |
| | 11 | 0.821 | 0.123 | 0.494 |
| | 14 | 0.794 | 0.141 | **0.479** |

Table 3: Analysis of different values for hyperparameters in Extra Trees.

# References

[1] Shane Fitzgerald, Aoife Morrin, and Linda Holland. Wound-associated bacterial pathogens volatilomic data, Sep 2021. URL https://figshare.com/articles/dataset/Wound-associated_bacterial_pathogens_volatilomic_data/16692217/2.

[2] Yusuf Kirikkayis. Proceedings of the 2020 omi seminars (promis 2020). *Universität Ulm*, 2021. doi: 10.18725/OPARU-38460. URL https://oparu.uni-ulm.de/xmlui/handle/123456789/38536.

[3] Thiloshon Nagarajah and Guhanathan Poravi. A review on automated machine learning (automl) systems. In *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, pages 1–6, 2019. doi: 10.1109/I2CT45611.2019.9033810.

[4] Shane Fitzgerald, Linda Holland, and Aoife Morrin. An investigation of stability and species and strain-level specificity in bacterial volatilomes. *Frontiers in Microbiology*, 12:3102, 2021. ISSN 1664-302X. doi: 10.3389/fmicb.2021.693075. URL https://www.frontiersin.org/article/10.3389/fmicb.2021.693075.

[5] Adel Abdulrazak, Zouheir Ibrahim Bitar, Abdullah Ayesh Al-Shamali, and Lubna Ahmed Mobasher. Bacteriological study of diabetic foot infections. *Journal of Diabetes and its Complications*, 19(3):138–141, 2005. ISSN 1056-8727. doi: https://doi.org/10.1016/j.jdiacomp.2004.06.001. URL https://www.sciencedirect.com/science/article/pii/S105687270400073X.

[6] Trang T Le, Weixuan Fu, and Jason H Moore. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. *Bioinformatics*, 36(1):250–256, 2020.

[7] Yali Amit and Donald Geman. Randomized inquiries about shape: An application to handwritten digit recognition. Technical report, CHICAGO UNIV IL DEPT OF STATISTICS, 1994.

[8] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[9] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[10] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *WIREs Data Mining and Knowledge Discovery*, 8(4):e1249, 2018. doi: https://doi.org/10.1002/widm.1249. URL https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1249.

[11] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[12] Randal S. Olson and Jason H. Moore. Tpot: A tree-based pipeline optimization tool for automating machine learning. In Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors, *Proceedings of the Workshop on Automatic Machine Learning*, volume 64 of *Proceedings of Machine Learning Research*, pages 66–74, New York, New York, USA, 24 Jun 2016. PMLR. URL https://proceedings.mlr.press/v64/olson_tpot_2016.html.

[13] Payam Refaeilzadeh, Lei Tang, and Huan Liu. *Cross-Validation*, pages 532–538. Springer US, Boston, MA, 2009. ISBN 978-0-387-39940-9. doi: 10.1007/978-0-387-39940-9_565. URL https://doi.org/10.1007/978-0-387-39940-9_565.

[14] Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5 (2):1, 2015.