**Random Forest Feature Selection:**

<https://chrisalbon.com/machine_learning/trees_and_forests/feature_selection_using_random_forest/>

Must be tested. In our case, the accuracy drop might not even be significant.

**Hyperparameter optimization:**

<https://machinelearningmastery.com/tune-machine-learning-algorithms-in-r/>

Two main methods identified. 1) The use of random parameters within given ranges, 2) grid search. I wonder if we could apply genetic algorithms to accelerate the hyperparameter search process.

**Feature Selection Methods:**

<https://towardsdatascience.com/explaining-feature-importance-by-example-of-a-random-forest-d9166011959e>

Feature selection methods shown:

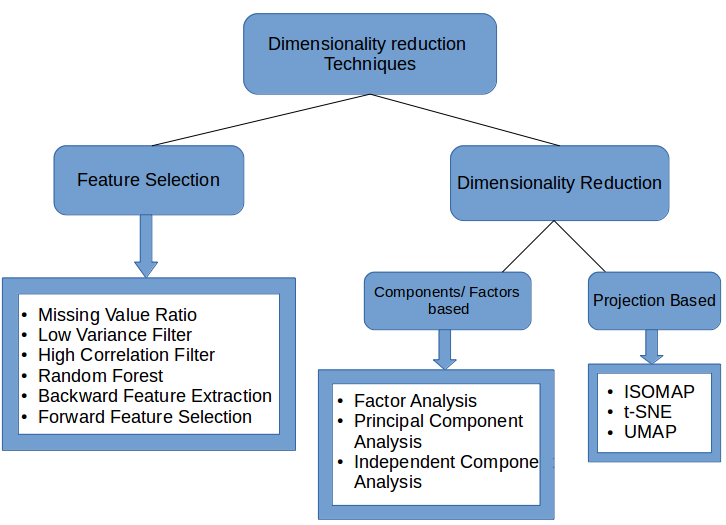
* RF feature importance
* Permutation Feature Importance (shuffling each feature and check accuracy drops)
* Drop Column Feature Importance (dropping one feature and checking differences in accuracy)
* References
  + [Beware Default Random Forest Importances](https://explained.ai/rf-importance/index.html)
  + [Conditional variable importance for random forests](https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-9-307)
  + [Interpreting random forests](http://blog.datadive.net/interpreting-random-forests/)
  + [Random forest interpretation — conditional feature contributions](http://blog.datadive.net/random-forest-interpretation-conditional-feature-contributions/)

**Additional feature selection method:**

<https://blog.datadive.net/selecting-good-features-part-iii-random-forests/>

Method shown:

* Author shuffles each feature and checks accuracy drops (Permutation Feature Importance)



**Dimensionality reduction techniques:**

<https://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/>

**UMAP:**

<https://arxiv.org/pdf/1802.03426.pdf>