UNIVERSITÀ
DEGLI STUDI
FIRENZE

Computational Replication of Essential Gene Identification Methods in Targeted Genome Sequencing

Niccolò Caselli

30/12/2024

Project submitted for the final exam of the Artificial Intelligence course - B003725

Professor:
Prof. Paolo Frasconi

Department:
Department of Information Engineering

Program:
Bachelor's Degree in Computer Engineering

1 Introduction

The purpose of this work is to replicate the experiments described in the article *"Towards the identification of essential genes using targeted genome sequencing and comparative analysis"* by Gustafson et al. (2006) to evaluate the performance of classifiers in predicting gene essentiality. A gene is considered essential if its absence prevents the survival of an organism. Identifying essential genes is a critical area where machine learning can provide significant value, as a reliable computational approach could drastically reduce experimental work which requires extensive skills and time. This resume compares the performance of Naive Bayes and Random Forest Classifiers on the provided datasets for *S. cerevisiae* and *E. coli*. This replication is conducted to validate the findings of Gustafson et al., focusing exclusively on the computational methods and results without exploring the biological aspects.

2 Methods

2.1 Tools and Libraries

The implementation adopted in this project relies on Python's `scikit-learn` library for machine learning, as opposed to the original paper, which utilized the `Orange` package. Other key libraries used in this implementation include `pandas` for data manipulation, `numpy` for numerical operations, and `matplotlib` for data visualization.

2.2 Material

The data sets are provided in the **Electronic Supplementary Material** section of the article. **Additional File 3** and **Additional File 4** contain the feature matrices for *S. cerevisiae* and *E. coli*, respectively. These data sets are provided in both raw and discretized formats and consist of gene-specific features along with binary labels that indicate gene essentiality. **Additional File 1** is also crucial, as it contains the CMIM feature ranking and the divisions of features into the three overlapping groups, which will be discussed later. The remaining material primarily includes the classification results reported by the authors.

2.3 Features selection

In order to replicate the experiment, it is essential to fully understand and reproduce the feature selection process carried out in the original study. As detailed in the subparagraph *performance of integrated features in yeast* in the **Results and discussion** section (Gustafson et al. 2006, p. 10), all the features for S. cerevisiae were assigned into three different overlapping sets:

- `SC_GenProt`: composed of all features which can be obtained directly from sequenced data.
- `SC_GenProt_No`: as *SC_ GenProt* but without the phyletic retention measure.
- `SC_All`: composed of features that require extensive experimentation in addition to the other easily obtainable features.

Similarly, the following paragraph states that two groups of features were analyzed for E. coli:

- `EC_GenProt`: composed of all features which can be obtained directly from sequenced data.
- `EC_GenProt_No`: as *SC_ GenProt* but without the phyletic retention measure.

In addition to features selection, mutual information maximization (CMIM) was used to rank the features and extract the most informative ones. In the subparagraph *Integration of Features* in the **Methods** section (Gustafson et al. 2006, p. 14, last paragraph) the authors reveal that the optimal number of features was determined empirically: after they were ranked they were iteratively removed one at a time. In this reproduction the CMIM is not calculated but it is used the pre-calculated one provided in the materials of the publication.

2.4 Data preprocessing

The process of discretization doesn't need to be implemented, as the paper's authors already supplied the discretized data in the given datasets as a second sheet named *entropy discretized*. For reference, as detailed in the subparagraph *Integration of Features* in the **Methods** section (Gustafson et al. 2006, p. 14, last paragraph), all features were discretized except phyletic retention and binary localization features. However, additional data preprocessing is required: a label encoder is applied to the feature matrices. The use of a label encoder, not mentioned in the original article, is justified as it allows the entries in the provided discretized data, which are

Organism	Group	Total number of features
S. cerevisiae	All	42
S. cerevisiae	GenProt	16
S. cerevisiae	GenProt_No	15
E. coli	GenProt	28
E. coli	GenProt_No	27

Table 1: Number of features for each group in *S. cerevisiae* and *E. coli*.

non-ordinal and represented as discrete categories (e.g., $\leq X$, $> X$, $(X, Y]$), to be transformed into unique numerical values. The used implementaion of the Label encoder is the one of *sklearn.preprocessing*.

2.5 Train-test split

In order to perform the training of the models in accordance with the methods used in the original paper, the datasets were divided into training and test sets. Specifically, *"half of all essential and half of all non-essential genes were randomly chosen to be included in the training set. The classifier was then tested on the remaining genes, which assigns a probability of essentiality"* (Gustafson et al. 2006, Integration of features). The task is accomplished with the *train_test_split* function of *sklearn* with the option *stratify* to avoid problems due to the unbalanced examples for essentiality.

Dataset	Total Rows	Train (50%)	Test (50%)
S. cerevisiae	4729	2364	2365
E. coli	3570	1785	1785

Table 2: Dataset sizes with train-test split (50%-50%)

2.6 Implementation

All the classifiers are coded using an OOP style by extending an abstract base class *EssentiallyClassifier*, which helps to avoid repetition in the code and encapsulates the methods for loading and processing the data, as well as empirically finding the optimal number of features. Two classifiers were implemented:

1. **Naive Bayes:**
   Naive Bayes is used to replicate the procedures of the article with fidelity as it is the classifier used in the original work. Although the paper does not specify the version of the classifier, the Categorical version best fits the problem. The number of categories is calculated using the *nunique* method in Pandas and is passed as a parameter to the model. Bootstrapping is also implemented as the article mentions that *"training/testing was bootstrapped 100 times"*.

2. **Random Forest:**
   Random Forest is used for comparison with Naive Bayes, although it is not employed in the publication. The classifier is initialized with the following hyperparameters:

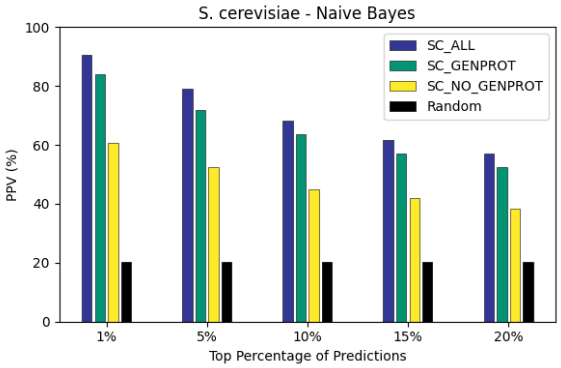
   - `n_estimators=100`: Number of trees in the forest.
   - `max_depth=15`: Maximum depth of each tree.
   - `min_samples_leaf=5`: Minimum number of samples per leaf.
   - `max_features='sqrt'`: Number of features considered for the best split.
   - `class_weight='balanced'`: Adjusts class weights for imbalanced classes.

   In contrast to Naive Bayes, K-fold Cross-Validation is used instead of bootstrapping, and the dataset was used without discretization.

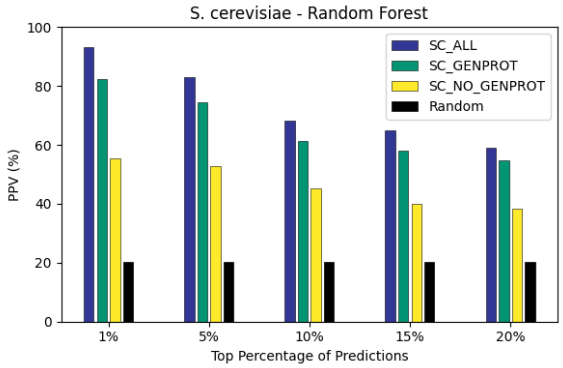
2.7 Evaluation Metrics

The performance of the classifiers is evaluated using two metrics: the Positive Predictive Value (PPV) at 1%, 5%, 10%, 15%, and 20% of the top predictions, which is the metric used in the original paper, and the Area Under the Receiver Operating Characteristic Curve (AUC). These metrics provide a comprehensive understanding of both the recision at various levels of prediction and the overall discriminative power of the classifiers.

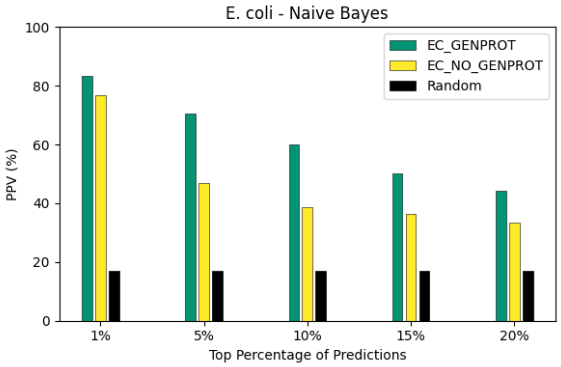
3 Results



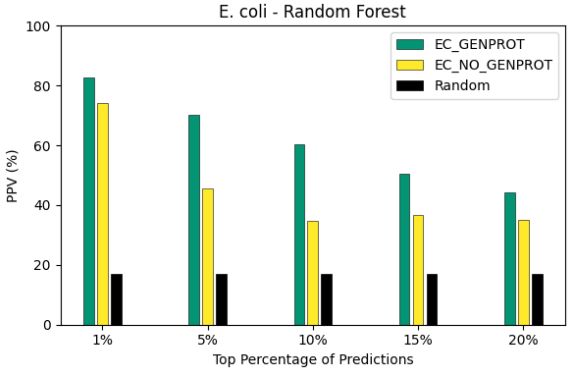
(a) Results for *S. cerevisiae* using Naive Bayes



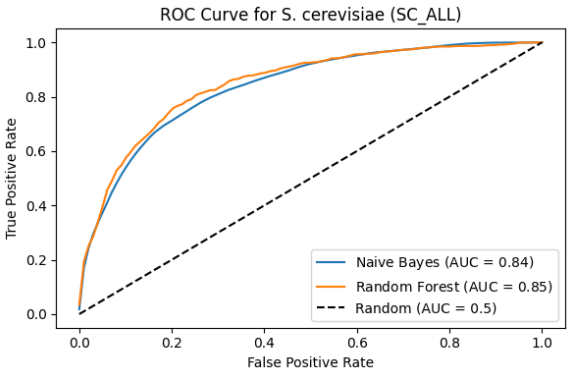
(b) Results for *S. cerevisiae* using Random Forest



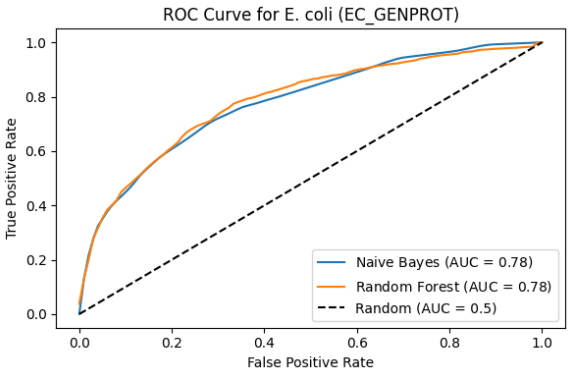
(c) Results for *E. coli* using Naive Bayes



(d) Results for *E. coli* using Random Forest



(e) ROC curve for *S. cerevisiae* classifiers



(f) ROC curve for *E. coli* classifiers

Figure 1: Comparison of classification results for *S. cerevisiae* and *E. coli* using different classifiers.

As shown in Figure 1a, the general trend of PPV results for S. cerevisiae aligns with those reported in the article (Gustafson et al. 2006, Figure 6a), with slightly better performance for the PPV at 1% of top predictions for SC_All (0.92 compared to 0.89 in the article) when using the 23 most informative features in SC_All, 11 in SC_GenProt, and 13 in SC_GenProt_No. The AUC (Figure 1e) also outperforms that of the original research, which, although not included in the materials, can be derived from the complete set of predictions. As expected, the Random Forest classifier achieves better results, as shown in both the ROC curves and PPV plots, but used more features for SC_All (36 instead of 21). Regarding E. coli, the performance is certainly worse due to the limited number of features and the more unbalanced dataset. However, the overall results align with the original experiment: despite a slightly worse PPV at the 1% percentile for EC_GenProt, EC_GenProt_No yielded better predictions, and the AUC (Figure 1f) is subtly higher than the original one. However, the number of features determined to be optimal, is lower than in the paper (3 in EC_GenProt and 4 in EC_GenProt_No). With this dataset, Random Forest (possibly due to incorrect optimization of hyperparameters, the same as those

used for S. cerevisiae) does not show a significant improvement.

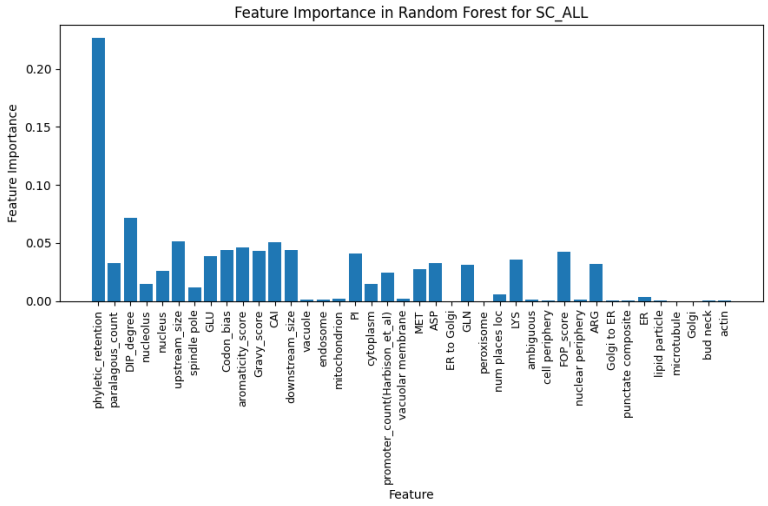


Figure 2: Importance of features in Random Forest for SC_All

4 Discussion

Although the general trends align with the original study, slight discrepancies in numerical results were noted; the reasons may vary and could stem from differences in hyper-parameter optimization and the implementations of the libraries used. Overall, good results were obtained, with the models trained showing slightly improved ROC curves compared to the original paper. However, these improvements remain modest, suggesting that the performance gains are not significant, except for demonstrating the success of a faithful replication. An important achievement is the confirmation that, despite SC_All outperforms SC_GenProt, the difference is small (approximately 5% at the top 15% and 20% of predictions), as also stated in the article (Gustafson et al. 2006, *performance of integrated features in yeast*). This is remarkable because it proves that we are losing little by ignoring features derived from extensive experimental data. Additionally, thanks to Random Forest, even without computing the CMIM, we are able to identify the most relevant features for classification, validating the CMIM ranking and the findings stated in the publication. Figure 2 clearly demonstrates, in the experiment for *S. cerevisiae*, that *phyletic retention* is the most predictive characteristic of essentiality, followed by *DIP Degree* and *Paralogous Count*; while *DIP Degree* requires a large amount of experimental work to obtain, the other two are genomic characteristics that are easily obtainable.

5 Conclusion

This replication study successfully validates the findings of Gustafson et al., showing that by using features directly available from sequence data, a simple classifier is capable of predicting essential genes with high accuracy. This method reduces both time and cost in essential gene identification by prioritizing genes predicted to be essential with a high confidence for experimental investigation. That said, there are areas in the article that could benefit from further refinement: incorporating a validation set for hyper-parameter optimization could significantly improve the model's robustness, while resampling techniques could address the problem of high class imbalance.

Additional Information

The source code and all the plots and tables of the results can be accessed on GitHub at `https://github.com/NiccoloCase/essential-genes-identification`.

References

Gustafson, Adam M. et al. (2006). "Towards the identification of essential genes using targeted genome sequencing and comparative analysis". In: *BMC Genomics* 7, p. 16. DOI: 10.1186/1471-2164-7-265. URL: https://doi.org/10.1186/1471-2164-7-265.