UNIVERSITY OF SUSSEX

Improved GOF/LOF Mutation Predictions by LoGoFunc: A Comparative Analysis of Classifiers and Feature Selection Techniques

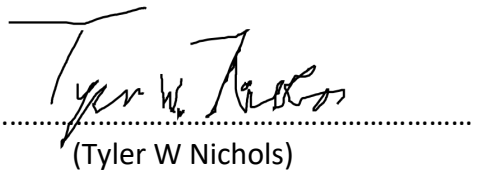
(Candidate Number: 276324)

Word count: 11,879 *(excluding tables and captions)*

MSc Computer Science (conversion)
Supervised by Dr. Benjamin Evans & Dr. Frances Pearl
Department of Informatics
University of Sussex
September 2024

DECLARATION

This report is submitted as part requirement for the degree of Master of Science in Computer Science (conversion) at the University of Sussex. It is the product of my own labor except where indicated in the text. The report may be freely copied and distributed provided the source is acknowledged.

Signed 
(Tyler W Nichols)

Acknowledgements

I want to thank my supervisors, Dr. Benjamin Evans and Dr. Frances Pearl, for agreeing to help me with my project over the past few months. It was important to me that I found an interdisciplinary project that combined computer science with my past fields of biology and chemistry, so I'm grateful for the opportunity. I'd also like to thank all of my friends and family, both in the UK and the US, who have helped me through some challenging times and have brought joy into my life.

ABSTRACT

The development of cancer occurs over several stages, generally requiring the accumulation of many mutations. The vast majority of these mutations are neutral “passenger mutations” that do not contribute to cancerogenesis. The few mutations that do fuel cancer development are called “driver mutations”. These mutations offer selective advantages to cancers by affecting immune system monitoring, growth inhibitor responses, and cell cycle regulation. They are classified into two groups: tumor-suppressor genes and oncogenes. Oncogenes are commonly hosts to pathogenic Gain of Function (GOF) mutations, which generate proteins that foster disordered cell growth and proliferation. Compared to Loss of Function (LOF) mutations, GOF are very difficult to study because they create new interactions between proteins, modify cellular signally networks, affect gene expression, and tend reside in unstructured regions of the genome. Pathogenic GOF/LOF variants have also been implicated in causing disorders such as epilepsy or neurodevelopmental delays and have even been correlated with schizophrenia.

In recent years, advancements in machine learning (ML) techniques, the accessibility of whole-genome sequencing, and the proliferation of richly annotated bioinformatic databases have made it possible for researchers to begin studying pathogenic GOF variants. One such advancement that has recently been developed is called LoGoFunc, a LightGBM-based tool that classifies variants as GOF, LOF, or Neutral. It uses an ensemble of 27 LightGBM classifiers, which have been trained on nearly 500 features taken from genomic databases, to make its predictions. However, LoGoFunc was trained on germline mutations and does not deliver accurate predictions on somatic mutations that may be acquired over a person’s lifetime. To begin working toward the goal of revising LoGoFunc to classify somatic mutations, we first wanted to see if the existing framework can be optimized with alternative classifiers and a reduced feature count.

For this project, I first assessed a series of 8 classifiers and compared their performances to LightGBM. This identified XGBoost as the only candidate to demonstrate superior performance. Next, I began working toward reducing feature count by assessing two different feature importance ranking methods. The first used an ensemble (ENS) of feature ranking algorithms, yielding an overall mean ranking for each feature. I compared this to a Leave-One-Feature-Out (LOFO) method that trained 499 models, each missing one feature, to identify the most important features. I then performed 4 sets of feature subset evaluations, where each model had one more feature than its predecessor, in order of their importance rankings. I identified the optimal feature count for both XGBoost and LightGBM, with the LOFO rankings being more useful. I found that XGBoost was able to deliver comparable or better performance than LightGBM, while requiring fewer features. This reduces the burden of ‘the curse of dimensionality’ and provides a valuable starting point for refining LoGoFunc toward somatic GOF mutation classification.

Table of Contents

Introduction

The clinical importance of GOF mutations

Cancers thrive on the imbalance between cellular repair mechanisms and genetic alterations[1]. A consequence of this imbalance is the tendency of cancer cells to amass a host of genetic and epigenetic modifications at a much higher rate than that observed in healthy cells. Cancer initiation and development occurs over several stages, usually requiring the accumulation of many mutations. However, the majority of those mutations are neutral "passenger mutations" that are not involved in cancer aetiology. The small subset of mutations that *do* encourage cancerogenesis are called "driver mutations". The average number of driver mutations differs by cancer type, with four per patient observed in colon cancer and only one per patient observed in thyroid cancer.

Driver mutations offer selective advantages to cancers by compromising cell cycle regulation, causing growth inhibitor desensitization, and by disrupting immune system monitoring[1]. Driver genes are broadly classified into tumor-suppressor genes and oncogenes. In cancers, the latter are often hosts to pathogenic Gain-of Function (GOF) mutations, yielding proteins that promote dysregulated cellular proliferation and growth. However, within the human genome, the distribution of driver mutations is irregular. This makes them very difficult to study. Attempts at elucidating the evolutionary history of several tumors indicated that at least half of early clonal driver mutations can be traced back to just 9 genes. Meanwhile, subclonal mutations were traced to 35 driver genes, indicating diversification of driver loci in later evolution. Driver mutations are inconsistent between cancers and individual patients, often lying dormant until certain stages of cancer are activated and may only become oncogenic in tandem with other mutations.

The detection of oncogenic driver events is crucial for comprehending the molecular aetiology of cancers[1]. GOF-bearing driver mutations have been demonstrated to be instrumental in cancerogenesis[2]. Unfortunately, the fact that GOF mutations alter cellular signaling networks, change interactions between proteins, and influence gene expression frustrates efforts to study them. This contrasts with Loss-of-Function (LOF) variants, where an assay only needs to determine if functioning has disappeared. Additionally, while most LOF are found in structured protein domains, GOF tend to aggregate in unstructured regions. This makes GOF more difficult to detect and characterize than LOF.

In the clinic, GOF/LOF mutations are not only relevant to cancer. *SCN8A* is a gene that encodes for the voltage-gated sodium channel $Na_V1.6$, a central nervous system (CNS) ion channel that is mainly expressed in the brain's excitatory neurons[3]. Pathogenic variants of *SCN8A* and other members of the *SCNxA* family have been implicated in the aetiologies of ion channelopathies that manifest as diseases of the CNS, kidney, heart, and muscle tissues. For *SCN8A* specifically, it is known that pathogenic LOF variants cause neurodevelopmental delays and adult-onset epilepsy, while GOF variants cause epilepsy in infants. Because GOF variants often cause increased action potential and/or propagation in excitable neurons, patients with these variants respond to

sodium channel blockers (SCBs), but patients with LOF variants will experience worsening symptoms when given the same drugs. This illustrates the clinical importance of not only identifying pathogenic variants, but also differentiating between GOF/LOF variants.

Another CNS ion channel that may be afflicted by pathogenic GOF/LOF variants is the *N*-methyl-*D*-aspartate receptor (NMDAR)[4]. The NMDAR has an elaborate structure and function, requiring glutamate and either glycine or *D*-serine to activate, while extracellular Mg^{+} is needed to relieve the voltage-dependent channel block. Variants of genes encoding for subunits of NMDAR called *GRIN1*, *GRIN2A*, *GRIN2B*, and *GRIN2D* are involved in the aetiologies of a group of neurological conditions called *GRIN*-related disorders. *GRIN2A* variants are correlated with schizophrenia, and other variants in this family are among the most common diagnoses in patients with epileptic and developmental encephalopathy. There is again a clinical necessity to distinguish between GOF/LOF variants because treating patients who are experiencing GOF-related NMDAR dysfunction with drugs designed to address LOF dysfunction causes symptomatic deterioration.

It is evident that elucidating the impact of GOF variants on biological processes is an important, yet poorly understood, area of cancer research[2]. It has also been established that GOF variants play a role in pathologies beyond cancer. GOF mutations present a unique challenge to researchers and, due to the complexity of their effects, advanced computational tools are required to understand the role that GOF variants play in human pathologies. Thankfully, the rise of improved sequencing techniques and richly annotated databases like cBio Cancer Genomics Portal (CBioPortal), Catalogue of Somatic Mutations in Cancer (COSMIC), OncoKB, and Genomics of Drug Sensitivity in Cancer (GDSC) have made genome-wide studies more practical than in previous decades[5–8]. When coupled with advances in machine learning (ML) algorithms, this vast pool of data opens the door for gaining bioinformatic insights into pathogenic GOF variants.

Computational approaches toward GOF/LOF classification

A growing body of evidence affirms the role that ML techniques can play in providing models of cancer progression and treatment outcomes[9]. ML techniques such as Artificial Neural Networks (ANNs), Support Vector Machines (SVMs), and Bayesian Networks have all been used to predict cancer susceptibility, recurrence, and prognosis. This is achieved by taking robust datasets and training models with relevant features. This technology can also be applied to the study of GOF variants and their relationships with various cancers. The Pearl group has previously developed an SVM classifier-based algorithm called MOKCaSVM, which was able to distinguish between LOF and GOF driver missense mutations with high accuracy[10].

The Hammer group (2023) have used a random forest (RF) classifier to differentiate between GOF/LOF variants of *SCN8A*[3]. They first trawled clinical and *in vitro* studies to gather data for 69 pathogenic LOF variants. For GOF variant acquisition, 45 out of 136 patients from a study by Johannesen *et al.* (2022) were randomly selected[3,11]. Next, they compiled a subset of these patients ("truth set") that only had one pathogenic *SCN8A* variant, possessed electrophysiologic data to confirm ion channel alteration, and were not hosts to pathogenic variants other than *SCN8A.* This resulted in a total of 45 GOF and 45 LOF variants, which were used as labels to train

an RF classifier with clinical features including seizure type, age of onset, and degree of intellectual disability. They developed two models: a full model using 34 clinical features, while a simplified clinical model used only 5 core features. The clinical model had slightly lower accuracy (95.3%) compared to the full model (99.9%), but it was more grounded in data that would be available to a clinician. Although much work is still required before this tool could enter clinical use, their clinical model provides rapid GOF/LOF classification for *SCN8A* variants.

Meanwhile, the Shen group (2023) developed a Graphical Missense Variant Pathogenicity Predictor (gMVP) for classifying variants as benign or pathogenic[12]. Their gMVP tool leveraged graph attention neural networks to make predictions. At the core of this architecture is a graph that has amino acid-derived features as nodes, while the edges are endowed with co-evolution strengths as weights. After developing the model, pathogenic and neutral GOF/LOF variants of voltage-gated calcium and sodium channels, including *SCNxA,* were collected. The researchers then employed a 'transfer learning' protocol, where training on the ion channel dataset was guided by the results of the generalized gMVP model, which shrank the parameter space that needed to be explored. This led to the development of a new model (gMVP-TL2) specifically designed for distinguishing between GOF/LOF variants, which consistently outperformed a funNCion model trained on the same curated dataset.

Another tool that has been developed to distinguish between GOF/LOF variants is called Triple-modalities Variant Interpretation and Analysis (TriVIAl)[13]. This tool combined a tabular model enriched with protein structural properties, a graph neural network (GNN), and a protein language model (pLM) to make predictions about the pathogenicity of missense variants. One advantage of this approach over similar tools is that this ensemble offers explainable results, indicating which areas of the protein are contributing to the prediction. This could make TriVIAl useful for clinical diagnostics or for developing targeted therapies through rational drug design.

LoGoFunc

In another example, ML techniques have been used to analyze a large database and distinguish between LOF and GOF variants[14]. In this case an accompanying web interface was developed to make it easier for users to run analyses. This eventually led to the development of LoGoFunc, a tool that leverages ML to anticipate the effects of GOF variants on cellular biology[15]. It uses a suite of LightGBM classifiers that have been trained on a vast pool of gene-, protein-, and variant-level features. Generalizable for every missense variant in canonical human transcripts, LoGoFunc makes genome-wide predictions on their functional impacts.

To assemble the dataset, 11,370 pre-labeled pathogenic gain-of-function (GOF) and loss-of-function (LOF) variants were retrieved from previous work by Bayrak and colleagues[14,15]. This initial dataset was then supplemented by 65,075 novel variants derived from the Human Gene Mutation Database (HGMD), which were labeled using Bayrak's methodology[14,16]. A subset of 32,911 of these samples were deemed to be disease-causing variants, and Natural Language Processing (NLP) techniques were used to query their corresponding publications to label them as either GOF or LOF. This same approach was used to query the ClinVar database and retrieve

additional variants[17]. The Genome Aggregation Database (gnomAD) was used to source neutral variants corresponding to the same genes as the GOF and LOF variants[18]. The final dataset consisted of 1,492 GOF variants, 13,524 LOF variants, and 13,361 neutral variants, with 90% allocated to training and 10% allocated to testing. The train-test split was engineered so that the proteins in the test set did not exceed 40% sequence similarity with those in the training set.

Next, a total of 474 features were engineered to optimize LoGoFunc's performance, each correlating with variant functional effects[15]. The features were extracted from datasets with extensive coverage of the genome or proteome. Redundancy was permitted between features to compensate for discrepancies in coverage and quality between the surveyed datasets. Based on their GRCh38 genomic coordinates, every variant was annotated with affected proteins, genes, and transcripts by the Ensembl database's Variant Effect Predictor (VEP)[19]. If known, the variants were also annotated with their positions within these elements. VEP plugins were used to retrieve variant pathogenicity predictions from PolyPhen2, CADD, CONDEL, SIFT, GERP++, SiPhy, PhastCons, and PhyloP. Additional metrics including GERP scores, BLOSUM 62 scores, and MaxEntScan predictions were also retrieved. Some of the remaining features were engineered from the analysis of protein models generated by AlphaFold2 (AF2), such as residue interactions or ligand-binding metrics[20].

When developing models for LoGoFunc, a 5x5 nested cross validation (CV) approach was used for monitoring variance and tuning hyperparameters[15]. Four types of classifiers were assessed for performance and generalizability: Neural Networks (NN), LightGBM (LGBM), RandomForest (RF), and XGBoost (XGB). The Optuna library was used to optimize the preprocessing pipeline and hyperparameter tuning for each of the classifiers, with the macro-averaged F1 score used as the guiding metric. The combination of preprocessing pipeline and hyperparameters that had the highest F1 score on the inner CV loop were used to evaluate each model on the outer CV loop. Following CV, the F1 score and median Matthew's Correlation Coefficient (MCC) were used to identify the optimal parameters. LGBM was determined to be the classifier with the highest performance metrics. The procedure was repeated using an ensemble of LGBM models ranging in size from 5 to 31. Hyperparameters were tuned for each individual model in the ensemble. LGBM performance was gauged using average precision (AP), F1 score, and MCC, with the optimal ensemble size being determined to be 27.

After determining the optimal parameters, LoGoFunc was compared 10 existing predictors of variant pathogenicity: CADD, SIFT, PolyPhen2, DANN, BayesDel, ClinPred, GenoCanyon, MetaSVM, PrimateAI, and REVEL[15]. Average Precision (AP) was used as the guiding metric for this comparison. LoGoFunc was determined to be the only method to offer significant improvement (AP 0.63) over baseline separation of GOF and LOF variants. While REVEL was able to distinguish between GOF and neutral variants (AP 0.55), LoGoFunc was superior (AP 0.82). REVEL and LoGoFunc offered identical (AP 0.87) performance when distinguishing LOF and neutral variants. LoGoFunc slightly (AP 0.91) outperformed REVEL (AP 0.88) when separating both GOF and LOF from neutral variants. Additionally, LoGoFunc was compared to two preexisting tools for GOF/LOF classification called funNCion and VPatho. LoGoFunc outperformed funNCion on GOF prediction but not LOF prediction, while it outperformed VPatho across all metrics. These results

established that LoGoFunc offers comparable or superior performance for GOF/LOF classification as existing tools, with a particular affinity for discriminating between GOF and LOF variants.

Project Aims and Objectives

The long-term goal of this work is to help develop an updated version of LoGoFunc that can make reliable predictions on a dataset of somatic mutations instead of germline mutations. Previous attempts to yield accurate predictions on somatic mutations were unfruitful. To pave the way toward this goal, it is desirable to further optimize LoGoFunc in two ways: identifying any classifiers that may offer better performance than LightGBM and reducing feature count, because the current dataset is burdened by high dimensionality.

The project described herein will start off by tuning hyperparameters for a selection of classifiers, training an ensemble of models using the provided training data, and evaluating them against the LoGoFunc test set. Overall performance metrics such as accuracy, precision, recall, and F1-score, will be used to identify the most promising classifier. This classifier will move on to a feature ranking and feature selection pipeline in tandem with a LightGBM model (using the original LoGoFunc parameters) to determine if it can outperform the original classifier.

Two methods will be used to rank the relative importance of each feature. First, an ensemble (ENS) of 12 feature-ranking algorithms will rank feature importances in series and then the mean will be computed to yield a ranked list of feature importances. Next, a leave-one-feature-out (LOFO) method will be used to rank the features using the best classifier identified in the previous step. 499 model ensembles will be trained, each missing one feature, and then they will be evaluated to determine feature importance rankings.

After obtaining ENS and LOFO feature importance rankings, they will be assessed using both the best classifier identified during the previous assessment and LightGBM. 499 model ensembles will be trained for all four ranker-classifier combinations, each with the top x number of features. The model ensembles will then be evaluated according to class-wise metrics to determine the optimal feature count for each ranker-classifier combination. This will then be used for further development of LoGoFunc toward somatic mutation classification.

In conclusion, this work aims to facilitate a deeper understanding of GOF variants by engineering a rigorously validated and adaptable ML tool that will contribute to the broader field of clinical genomics. By the end of the project, I will have made the first step toward the development of a more robust version of LoGoFunc that can be effectively deployed across diverse academic and clinical contexts, providing researchers with essential bioinformatic insights into GOF/LOF variants.

Materials and Methods

Sourcing Datasets

All datasets were sourced from the Itan group, as published on the LoGoFunc GitLab repository (available at: https://gitlab.com/itan-lab/logofunc)[15,21]. The training and test datasets were already split in a 90:10 ratio by the Itan group prior to use, containing a total of 28,377 variant samples. The datasets initially contained 500 features, which are described in **Appendix A**. While the authors reported only 474 features, the additional features likely result from their application of ordinal encoding to the training and test datasets prior to use.

Computational Platform

All experiments were performed on a 2023 MacBook Pro computer with an Apple M2 Max processor (12 cores) and 32 GB of RAM, which was running macOS (Sonoma 14.5). Experiments were designed and executed in Jupyter Notebooks using the Python programming language (ver. 3.9.19) inside Microsoft Visual Studio Code (ver. 1.91.1). Two different Anaconda 'conda' environments were used, one for training/testing and the other for data analysis. The YAML configuration files for replicating these environments are included in the supporting information.

Assessing Classifiers

Classifiers Used

A total of 9 classifiers (**Table 1**) were evaluated and compared to identify the most promising candidates for further development. These included representatives from several different families of classifiers including gradient-boosting machines (GBMs), adaptive boosting, support vector machines (SVMs), decision trees, and neural networks. A script was developed that handled preprocessing, hyperparameter tuning, training, validation, and testing. In the interests of simplifying direct comparisons, the script was kept as similar as possible between classifiers aside from hyperparameter tuning. Two LightGBM trials were run. The first trial (LGBM) used the same 499-feature dataset as the other classifiers. The second one (LGBM-P) used the original 500-feature dataset with the 'Protein_dom' feature included. This was done so that the impact of removing this feature could be assessed, since feature importance rankings were not provided by the authors of LoGoFunc, and this could potentially be a critically important feature. It should be noted that due to the feature names containing invalid characters, they needed to be encoded prior to use with XGBoost. This was accomplished using a custom script that only encoded the feature names; the datapoints were pre-processed according to the General Procedure.

Table 1: Classifiers that were assessed for GOF/LOF classification.

Entry:	Classifier:	Description:
1	LightGBM	Gradient-boosting framework with tree-based learning algorithms
2	CatBoost	Gradient-boosting framework with a focus on categorical features
3	XGBoost	Gradient-boosting framework with tree pruning, regularization, and high scalability
4	AdaBoost	Adaptive-boosting framework with a focus on misclassified instances
5	LinearSVC	Linear Support Vector Machine for classification with no kernel
6	SGD	Stochastic gradient descent-optimized linear classifier
7	ExtraTrees	Extremely random decision tree ensemble with node splitting
8	Stacking	Ensemble of stacked classifiers with Random Forest, Extra Trees, and Decision Trees as base models
9	MLP	Feedforward neural network with hidden layers and ReLU activation

LightGBM (LGBM)

LightGBM is a classifier based on gradient-boosting decision trees (GBDTs), a frequently-encountered ML technique that offers high interpretability, accuracy, and efficiency[22]. GBDT is an ensemble method using sequentially trained decision trees that excels in areas such as click prediction, multi-class classification, and ranking tasks. However, in the age of Big Data, GBDT must grapple with the inevitable compromise between efficiency and accuracy. During decision tree learning, GBDT iteratively fits the residual errors (negative gradients). This tree learning cycle is the costliest component of GBDT, with split point determination being the most time-consuming process. By default, GBDT algorithms need to scan every data instance for every feature to evaluate candidate split points. This resulting computational complexity is proportional to feature counts and data instances, rendering it time-consuming and computationally expensive for large datasets. **Figure 1** shows the formula describing the information gain described by the variance after splitting in GBDT.

$$V_{j|O}(d) = \frac{1}{n_O}\left(\frac{\left(\sum_{x_i \in O: x_{ij} \le d} g_i\right)^2}{n_{j_l|O}(d)} + \frac{\left(\sum_{x_i \in O: x_{ij} > d} g_i\right)^2}{n_{j_r|O}(d)}\right)$$

Figure 1: Variance gain after splitting in GBDT.

Considering this, it is pragmatic to minimize both features and data instances[22]. A popular strategy filters out weaker features and down-samples data instances. Although weighted approaches such as Stochastic Gradient Boosting (SGB) are sometimes applied to other boosted training frameworks (e.g., AdaBoost), unfortunately, when applied to GBDT it reduces accuracy. This stems GBDT's lack of native weights for data instances. Similarly, when reducing feature

count through filtering, approaches like projection pursuit and principal component analysis (PCA) depend upon significant feature redundancy, which may not exist.

Large datasets often exhibit high sparsity, and GBDT can reduce training cost by using a pre-sorted algorithm to ignore features containing zero values[22]. An alternative is a histogram-based algorithm that is more efficient in training speed and memory consumption, however it is unable to effectively leverage data sparsity to its advantage. Therefore, it was desirable to develop a method that harnessed the efficiency gains of the histogram-based algorithm while also exploiting data sparsity. Two techniques that have been developed to address this problem are Gradient-Based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). These techniques are what differentiates LightGBM from other GBDT methods.

Development of GOSS originated with the observation that instances with larger gradients ('undertrained instances') contribute more to information gain than other instances[22]. Therefore, during down-sampling, it is ideal to randomly drop instances with smaller gradients and keep the instances with larger gradients. Compared to uniformly random sampling, this achieves a more accurate gain estimation. It has also been noted that even when many features are present, feature spaces tend to be sparse. Because the features in sparse feature spaces are often mutually exclusive (meaning they do not simultaneously take nonzero values), they can be bundled together. The EFB algorithm takes the optimal bundling problem and turns it into a graph coloring problem by treating the features like vertices. It takes the sum of edges for each pair of features and solves the problem using a greedy algorithm. This affords LightGBM a reduction in GBDT training time without diminishing accuracy.

To assess the performance of LightGBM, it was compared to two versions of XGBoost as well as a variant of LightGBM without GOSS or EFB ('baseline') using 5 publicly-available datasets[22]. One version of XGBoost used the histogram-based algorithm and the other used the pre-sorted algorithm. For training time and test accuracy, LightGBM was the fastest with similar accuracy to the other classifiers. This established that GOSS and EFB offer speed gains without negatively impacting accuracy. When trialing GOSS alone in comparison to GBM, the results showed that GOSS is the superior sampling method. When comparing EFB alone to the baseline LightGBM, a significant improvement in training speed was observed. Collectively, these results illustrate the efficiency gains offered by GOSS and EFB that make LightGBM a powerful GBDT-based classifier and demonstrate its ability to perform well in real-world applications.

XGBoost (XGB)

Another popular implementation of GBDT techniques is an open-source package called XGBoost ("e**X**treme **G**radient **Boost**ing")[22–24]. In ML competitions such as Kaggle and the KDDCup, XGBoost has excelled in scenarios as diverse as text classification, sales predictions, customer behavior predictions, and product categorization[23]. The success of this classifier can largely be attributed to its high scalability, often running more quickly than other classifiers and able to support sample sizes in the billions. One factor that makes this possible is that XGBoost uses a unique tree learning algorithm that is optimized to address sparse data. Next, a weighted quantile sketch is

applied to instance weights during tree learning. XGBoost is also optimized to exploit parallel and distributed computing, as well as 'out-of-core' computation, which accelerates learning and exploration rates. The net result of this combination is that it's feasible to process massive datasets from a desktop computer.

The second-order gradient boosting method at the heart of XGBoost is derived from the work of Friedman and colleagues[25]. The learning objective of XGBoost consists of two primary parts: a loss function that compares predictions to outcomes, and a regularization component that penalizes excessive model complexity to discourage overfitting[23]. The tree ensemble model inside XGBoost uses entire decision trees as parameters and is trained in an additive fashion, with second-order approximation used to find the optimal objective. In other words, while training the model, every tree is engineered to rectify the mistakes of its predecessor using the first and second derivatives of the loss function. Furthermore, XGBoost leverages shrinkage (minimizing the influence of individual trees to provide space for additional trees) and column subsampling to further inhibit overfitting. **Figure 2** shows the equation for optimizing the weights of leaves within the decision tree.

$$w^{*}j = -\frac{\sum i \in I_j g_i}{\sum_{i \in I_j} h_i + \lambda}$$

Figure 2: Formula for decision tree leaf weight optimization.

A cornerstone of tree learning is determining the best split, which involves an 'exact greedy algorithm' enumerating over every split that is possible for every feature[23]. This is a computationally expensive operation, and datasets are frequently too large to fit inside the computer's available memory, so an approximation algorithm is needed. First, the data must be prioritized by feature values and then visited in order of priority to collect gradient statistics. Splitting points are determined based on feature distribution percentiles and then the features are mapped into buckets split along these breakpoints. Statistics are then collected, and the algorithm isolates the optimal solution accordingly. This last step is implemented as one of two variants: global or local. The global variant identifies split candidates at the start of tree-building and uses them to find splits at each level, while the local variant identifies candidates following every split. Both approaches are supported by XGBoost and can be selected by users on demand.

Another constituent of XGBoost's approximation algorithm is called weighted quantile sketch[23]. The purpose of this component is to identify optimal feature splitting points when datasets would otherwise be too large to fit into available memory. The split-finding that is used by XGBoost is also 'sparsity aware', meaning that if a value is missing then the classifier classifies in a chosen direction. Each branch has two choices of directions, and the default direction is learned by the classifier from studying the data. This algorithm lends linear computational complexity to the number of non-missing datapoints. Because much of the available data from real-world problems is sparse, this has non-trivial predictive implications. In contrast, most other tree-learning algorithms are streamlined for dense data applications. Other factors that contribute to

XGBoost's speed and scalability include column blocks for parallel learning, cache-aware access, and blocks for out-of-core computation.

When these factors are taken into consideration, XGBoost represents a powerful advancement in gradient tree boosting techniques. The sparsity-aware split finding algorithm, out-of-core computing capabilities, weighted quantile sketch, and block structure synergize to offer higher speed, accuracy, and scalability compared to other classifiers.

CatBoost (CTB)

CatBoost (CTB) is yet another member of the GBDT family, and like XGBoost it is open-source[26,27]. This classifier debuted in late-2018 and has found utility in many Big Data applications. CatBoost constructs its decision trees using an architecture called Oblivious Decision Trees (ODTs)[26]. An ensemble of binary trees is created, with all non-leaf nodes exhibiting identical splitting criteria. The developers of CatBoost assert that ODTs allow for higher speeds and reduced overfitting[27]. The downside is that users need to exercise caution when choosing maximum tree depth because memory consumption increases by a factor of 2 times the total quantity of trees for each maximum tree depth unit[26]. The two unique innovations that CatBoost introduced are Ordered Target Statistics (OTS) and Ordered Boosting (OB), while the overall implementation is a further refinement of the work by Friedman[26–28].

For categorical variables, CatBoost uses one-hot encoding with low cardinality variables and OTS for high cardinality variables[26]. A target statistic (TS) in this context is a method of summarizing the corresponding ground truth output values for each unique value of a categorical feature. The formula that CatBoost uses for computing the TS for categorical features is **shown below.** Substituting a feature's categorical values with a target statistic is an effective way of handling categorical variables within ML pipelines. CatBoost's OTS method (Figure 3) is specifically optimized to inhibit 'target leakage' (which causes overfitting) by not including the target value of the current example in the calculation while encoding it. A random permutation of the dataset is used and, exclusively, preceding examples are considered in the permutation during encoding. The OB technique was developed to address the prediction shift that is caused by target leakage. It guarantees that the subset used during OTS calculation is identical to the subset used in the boosting process's gradient estimation. It should also be noted that, while the automatic encoding provided by CatBoost is a useful feature, it is not unique; LightGBM also provides this functionality. Overall, CatBoost's focus on processing categorical variables while inhibiting overfitting, alongside its efficient ODT architecture, make it one of the most appealing options among GBDT classifiers, especially for datasets that are rich with categorical features.

$$x_{ik} = \frac{\sum_{x_j \in D_k} 1x_j^i = x_k^i \cdot y_j + a \cdot p}{\sum_{x_j \in D_k} 1x_j^i = x_k^i + a}$$

Figure 3: OTS equation for CatBoost.

AdaBoost (ADB)

"Boosting" is a technique that isolates an accurate hypothesis through the combination of several moderately accurate ('weak') hypotheses[29,30]. The AdaBoost (ADB) algorithm was first published in 1995 and addressed several of the shortcomings of preceding boosting algorithms by repeatedly applying a basic learning model to a given dataset to improve its accuracy. AdaBoost assigns equal weights (importances) to every datapoint at the beginning of the algorithm. As each round of learning progresses, the importance of incorrectly classified datapoints is increased, which encourages the model to home in on these challenging examples. AdaBoost eventually yields a final model that combines the previous 'weak' models, with the most accurate models being assigned the highest weight when making decisions. The defining feature of AdaBoost is its efficiency at minimizing error during training. The main disadvantage of boosting methods like this is that they are sensitive to noise. AdaBoost often fails to yield satisfactory results if insufficient data is provided, weak hypotheses are overly complex, or when it is presented with weak hypotheses that are not significantly better than random guessing. Nonetheless, AdaBoost has been in widespread use for nearly 30 years, with researchers finding applications for it in contexts such as ranking problems, natural language processing, and text filtering. **Figure 4** shows the formula that AdaBoost uses to compute the weight update rule at the center of its algorithm.

$$w_{t+1}(i) = w_t(i) \cdot \beta_t^{1-\ell_t(i)}$$

Figure 4: AdaBoost's weight update rule.

LinearSVC (LSVC)

LinearSVC (LSVC) is a linear Support Vector Classifier (SVC) implementation included in the Scikit-learn library[31]. This classifier is based on LIBLINEAR, an open-source library designed for linear classification on large datasets[32]. Linear classification methods often excel at learning from large, sparse datasets that are rich in features and instances. Compared to Scikit-learn's SVC (which itself can operate with a linear kernel), LSVC offers a more versatile selection of loss functions and penalties, which leads to higher scalability[31]. A one-vs-the-rest implementation provides multiclass support for both sparse and dense data inputs. It's worth noting that, while the underlying architecture is similar, LSVC is not identical to LIBLINEAR because when the 'decision function' parameter is set to zero for a given sample, LIBLINEAR predicts the positive class while LinearSVC predicts the negative class.

SGDClassifier (SDGC)

SGDClassifier (SDGC) is a Scikit-learn module that pairs a linear classifier (it uses SVM by default but also supports logistic regression) with Stochastic Gradient Descent (SGD) learning[33]. Because SGD offers a platform for rapidly minimizing a variety of loss functions, and supports logistic optimizations and SVM, it has emerged as a popular technique for solving supervised ML problems on large datasets[34]. The loss gradient is estimated in a stepwise fashion while the model's learning rate decreases over time[33]. SGDC is best suited for features that are rich in

floating-point values, whether the dataset is sparse or dense. SGDC's other notable traits include three options for regularization (L1, L2, or Elastic Net) and support for out-of-core calculations.

ExtraTreesClassifier (ETC)

Scikit-learn's ExtraTreesClassifier (ETC) uses Extremely Randomized Trees (ERT), a decision tree-based ensemble, to solve supervised regression or classification problems[35,36]. In this type of ML model, data is organized into a tree structure and each internal node applies a binary test to a single input feature[37]. Trees are constructed by recursively selecting tests at every node, dividing the data into two halves, and making each grouping as uniform as possible. Because individual regression/classification trees generally experience high variance, ensemble methods offer dramatic improvements in accuracy. By introducing random disorder into the learning protocol, randomization methods can yield different models from one learning sample. Although ERT shares characteristics with the Random Forests (RF) algorithm, such as selecting a random subset of features at each node, ERTs differ by using the entire subset instead of a sample of the subset. Additionally, while RF uses a local sample to choose an optimal cut-point for each feature, ERT selects a cut-point at random. The benefits of ERT methods include parameter-free support of non-linear models and competitive computation times, although they may not always be as accurate as other, more state-of-the art techniques.

Multilayer Perceptron (MLP)

Artificial Neural Networks (ANNs) are a type of artificial intelligence composed of a myriad of simplistic, interconnected processing units called neurons, each one yielding a sequence of activations with real values[38]. Sensors perceiving the environment can activate input neurons, while additional neurons are activated in a cascade according to weighted connections to other neurons in the system. Multilayer perceptrons (MLPs) are a type of ANN that contain at least three layers of neurons (nodes): an input layer, a hidden layer, and an output layer[39]. These feed-forward ANNs serve as powerful global approximators and have an excellent tolerance for data sparsity, high dimensionality, and nonlinear data structures. The MLP topology has found utility in applications as diverse as pattern recognition, classification, approximation, and prediction problems. In a study that compared MLP to SVM, classification-and-regression tree (CART), and K-nearest neighbors (KNN) in a high-dimensionality classification problem, MLP exhibited a superior classification rate[40]. MLP remains a popular choice of classifier in a broad range of fields, finding uses in medical diagnostics, labor data analysis, road accident prediction, and anticipating academic performance[39].

StackingClassifier (STACK)

Scitkit-learn's StackingClassifier (STACK) enlists an ensemble ('stack') of different estimators to generate predictions[41]. Due to the favorable preliminary results observed with ETC, a trio of decision tree-based classifiers were selected as the base estimators: ETC, RandomForestClassifier

(RFC), and DecisionTreeClassifier (DT). The predictions generated by the base estimators are then passed to a LogisticRegression classifier, which generates the final prediction.

General Procedure for Preprocessing

After loading the datasets into Pandas DataFrames, the feature called Protein_dom was dropped because it was derived from annotations from the Pfam database. The long-term goal of this project is to adapt LoGoFunc, which was trained on germline mutations, to classification of somatic mutations, which are not well-represented in this database[15,42].

The preprocessing pipeline was derived from the utility script published on the Logofunc GitLab repository[21]. Categorical features were encoded using Scikit-learn's OrdinalEncoder class, which encodes each of the features into an ordinal integer array, with one column of integers per feature[43]. For features listed in the NEGONE list, Scikit-learn's SimpleImputer class performed imputation with a constant value of -1[44]. For features contained in the MEDIAN list, SimpleImputer performed imputation using the median. Scikit-learn's MinMaxScaler class was used for scaling in both cases[45]. This scales each feature in the dataset so that it is within a given range between zero and one, where the largest datapoint corresponds with the maximum value. For additional data cleaning, any columns that contained only NaN values were dropped using a custom 'drop_allnan' function. Custom encoding was applied to the IMPACT feature, with LOW, MODIFIER, MODERATE, and HIGH mapped to 0, 1, 1.5, and 2, respectively. Similarly, custom encoding was applied to the y_train labels, with GOF, LOF, and Neutral mapped to 0, 1, and 2, respectively.

Tuning Hyperparameters

Hyperparameters were tuned for each classifier using the full training dataset (25,546 samples), which included 499 features. Although manual search and grid search are widely used for optimizing hyperparameters, Bergstra and Bengio demonstrated that random search often yields models of comparable or higher quality using fewer computational resources[46]. Therefore, Scikit-learn's RandomizedSearchCV class was used to randomly search 10% of the selected search space for each classifier, with 3-fold cross-validation for each hyperparameter combination. The hyperparameters that were tuned for each classifier are listed in **Table 2**. The search spaces that were queried during tuning for each classifier are listed in **Appendix B**. 8-core parallel processing was used to increase computational efficiency. Top-performing parameter sets were identified by their mean cross-validation scores. After tuning, the 27 best hyperparameter combinations were exported to JSON files. The full list of parameters for each ensemble can be viewed in **Appendix B**.

Table 2: Hyperparameters that were tuned for each classifier.

Model	Param 1	Param 2	Param 3	Param 4	Param 5	Param 6	Param 7
CTB	Iterations	Learning rate	Depth	L2 leaf reg.	Border count	Bagging temp.	Random strength
XGB	Num. estimators	Learning rate	Max. depth	Subsample	Column sample by tree	Min. child weight	Gamma
ADB	Num. estimators	Learning rate	Algorithm	Max. depth	Min. samples split	Base estimator criterion	
LSVC	Reg. strength	Class weight	Max. iterations	Tolerance			
SGDC	Alpha	Learning rate	L1 ratio	Class weight	Penalty	Eta0	Power t
ETC	Num. estimators	Max. depth	Min. samples split	Min. samples leaf	Max. features	Bootstrap	Criterion
MLP	Hidden sizes	Learning rate	Batch size	Epochs	Dropout rate		
STACK	RFC Num. estimators	RFC Max. depth	ETC Num. estimators	ETC Max. depth	DT Max. depth	Final estimator	

Train-Validation Split

Although performing a train-test split is a critical step toward preventing overfitting on the training set, the risk of overfitting on the test set persists[47]. This is because during tuning, test information can leak into the model, causing generalizability to plummet. Withholding a validation subset of the training set is intended to address this issue. In this procedure, training on the training set is performed as usual, with an additional evaluation step in between the training and testing phases. Only when the training and evaluation steps provide satisfactory metrics does the model proceed to being used in a final evaluation using the unseen test set. The downside is this additional partition between training and validation sets reduces the sample size that the model is trained.

However, due to the large number of classifiers that needed to be assessed, it was decided that an evaluation step after training would be worthwhile for ensuring that the tests remained unbiased and ascertaining the success of hyperparameter tuning before potentially wasting computational resources on testing if the model was not properly tuned. A train-test split was performed using Scikit-learn's 'train_test_split' function (yielding four data subsets: X_train, X_val, y_train, and y_val) before hyperparameter tuning and training[48].

K-Fold Cross-Validation (CV)

K-fold cross-validation (CV) is another method that can help prevent overfitting during hyperparameter tuning[47]. This operation does not require a dedicated validation set. Instead, a *k* number of splits are applied to the training set, dividing it into *k* subsets. During training, $k - 1$ number of folds are used as training data, then the model is evaluated against the withheld subset of the data. The average values obtained from the loop are reported, which may confer additional computational overhead but sacrifices less of the training data. In this project, Scikit-learn's RandomizedSearchCV class was used to apply 3-fold CV to each of the hyperparameter and tuning scripts during classifier assessment. An additional 5-fold CV was performed by Scikit-learn's KFold class prior to hyperparameter tuning to provide a baseline "sanity check" using the default classifier parameters. **Figure 5** illustrates this process.

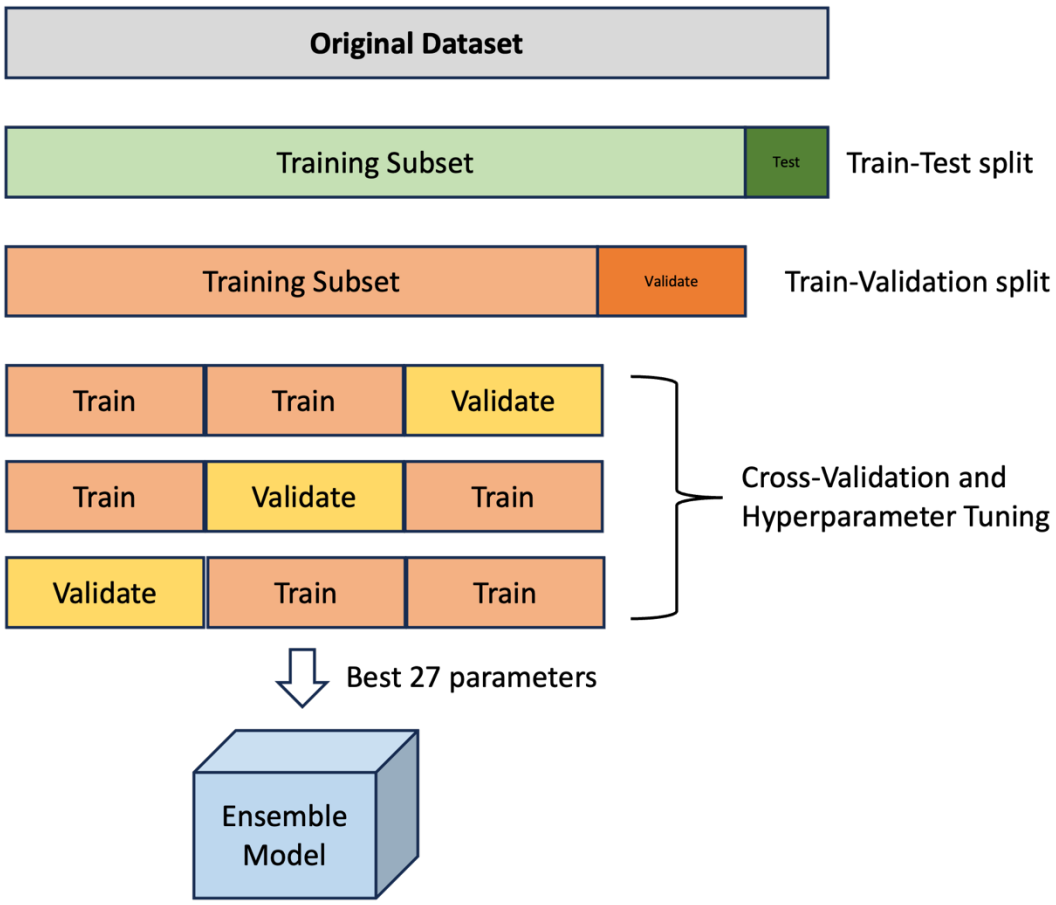


Figure 5: Illustration of the hyperparameter tuning and model ensemble training process.

General Procedure for Model Training

For each classifier, the top-performing parameters were used to train an ensemble of 27 models. This approach mirrors that of the original LoGoFunc implementation using LightGBM, and it allows the ML tool to harness the strengths of multiple hyperparameter settings and improves

its overall robustness[15]. Once trained, the models were evaluated using the validation subset that had been split off the training dataset before hyperparameter tuning and training. A 'soft voting' algorithm was used to evaluate the ensemble by collecting the predicted probabilities for each model and averaging them to arrive at a consensus for the final predictions. Validation metrics including accuracy, precision, recall, F1 score, and ROC AUC were collected to ensure satisfactory performance.

General Procedure for Model Testing

During the testing phase, each of the 27 models for each classifier were iteratively accessed to make predictions on the test set. The same preprocessing pipeline was used for testing as for training. A soft voting algorithm was used to aggregate predictions across the 27 models. The final prediction was made based on the class with the greatest mean probability. The performance of each ensemble was evaluated according to 4 metrics: accuracy, precision, recall, and F1-score. These metrics were then used to quantitatively compare performance between classifiers.

Since overall performance metrics were not reported by the authors of Logofunc, LightGBM was used with the published hyperparameters to serve as a standard by which to judge the performance of the other classifiers. Additionally, a second LightGBM trial was run with Protein_dom included in the feature set to evaluate the impact of excluding this feature. It was observed that all performance metrics slightly improved when this feature was excluded. For all recorded metrics, XGBoost was the only classifier that consistently out-performed LightGBM. The only other classifier to offer comparable performance to LightGBM was CatBoost, although it performed slightly worse for each metric. Given these results, it was decided to use XGBoost for further development and optimization.

Ranking Feature Importances

When developing a predictive ML model, the curse of dimensionality is one of the largest obstacles for researchers to overcome[49]. This is especially true when working with high dimensionality datasets such as the one included with LoGoFunc, which was originally trained using 500 features. The field of bioinformatics is rich with examples that illustrate this problem, such as datasets used in genome-wide association studies (GWAS)[49]. A typical GWAS dataset may include a million single-nucleotide polymorphisms (SNPs) but only a few thousand samples. When this type of dataset is used for ML classification without alteration, it will probably create an overfitted model. Overfitting is caused by the model paying undue attention to random variations and noise in the training data, resulting in favorable performance on the training dataset but not on the unseen test dataset. Additionally, excessive dimensionality significantly dilates computational and learning time due to cluttering of the learning algorithm by redundant and unnecessary features. Therefore, it is prescient to prune unnecessary features from the dataset before use.

One frequently employed method for reducing dimensionality is feature selection[49]. Identifying and extracting a subset of highly relevant features through carefully engineered feature selection increases predictive accuracy and learning efficiency, while minimizing the complexity of the learning results. Feature selection studies are a popular subdiscipline of ML research, so a myriad of algorithms have been developed to aid with this task, each with their own advantages and disadvantages[50]. The ideal feature set contains full discriminatory information regarding the output variable. However, because feature spaces grow exponentially with the number of features, evaluating every possible feature subset from a set of features is an NP-hard problem. This means that feature selection is often a computationally expensive operation and choosing the correct technique is critical when engineering an ML pipeline.

When working with labeled datasets, feature ranking algorithms employing supervised learning techniques associate features with the outcome variable to decide which feature is the most relevant[50]. These feature selection methods can be divided into three main classifications: filter, wrapper, and embedded. Filter-based techniques sort features according to relationships with the dependent variable as well as to other features. Although less accurate than the other methods, filter-based techniques are computationally inexpensive and simple to implement. Wrapper-based techniques use classifiers to find feedback on feature importance based on performance estimates from predefined learning algorithms. These methods are more accurate, but their complexity confers a high computational cost. Embedded techniques leverage the strengths of both filter-based and wrapper-based feature selection methods. They first use the search strategy of filter techniques, and then they use a classifier to evaluate the feature subset as in wrapper-based techniques. The main drawback of embedded techniques is that they are more prone to overfitting than wrapper-based techniques.

Ensemble Feature Importance Ranking (ENS)

Ensemble feature importance ranking (ENS) constructs an ensemble of feature ranking algorithms with the objective of optimizing feature selection robustness[50]. ENS strategies can be broadly split into two groups: homogeneous and heterogenous ensembles. Homogeneous ensembles use rankers chosen based on data perturbation, while the rankers that constitute heterogeneous ensembles are chosen based on functional diversity. Hybrid approaches combine elements of both strategies. For this project, a heterogenous feature selection ensemble was chosen, with a total of 12 diverse feature selection algorithms (**Table 3**) used. Rankings from each algorithm were pooled in a Python dictionary of dictionaries, and then the mean ranking was calculated for each feature and used as the final ranking.

Table 3: Feature-ranking algorithms used during ENS.

Entry:	**Algorithm Name:**	**Description:**
1	F-Regression	Univariate linear correlation of feature to target
2	Linear Regression	Importance based on absolute coefficient magnitude
3	Ridge Regression	Linear regression with L2 regularization
4	Lasso Regression	Linear regression with L1 regularization
5	Mutual Information	Non-linear dependency between feature and target
6	Random Forest Regressor	Ensemble tree-based feature importance scores
7	ExtraTrees Classifier	Importance from extremely randomized tree ensemble
8	RFECV: ElasticNet	L1/L2-regularized iterative feature elimination
9	RFECV: Linear Regression	Linear model with iterative feature elimination
10	RFECV: Logistic Regression	Logistic model with iterative feature elimination
11	RFECV: SVC	Support vector-based iterative feature elimination
12	RFECV: Random Forest	Ensemble tree-based iterative feature elimination

Ensemble Feature Ranking Algorithms

F-regression

In Scikit-learn, the 'f_regression' function is a statistical test that yields p-values and F-statistics via univariate linear regression[51]. It sequentially assesses the impact of individual regressors using a linear model. The function first calculates the cross-correlation between each feature and each target variable (**Figure 6**), then it converts them to F-scores and p-values. This function is derived from 'r_regression' (which instead computes Pearson's correlation coefficient), and if every feature correlates positively with the target then it will rank them in the same order[51,52]. Unlike that ranking method, 'f_regression' cannot return negative values, which makes it more suitable for finding predictive features for downstream classifiers. This is an example of an filter-based feature selection technique[50].

$$\frac{E\left[\left(X[:,i] - \text{mean}(X[:,i])\right) \cdot \left(y - \text{mean}(y)\right)\right]}{\text{std}(X[:,i]) \cdot \text{std}(y)}$$

Figure 6: Formula for F-regression cross-correlation.

Linear Regression

The LinearRegression class in Scikit-learn provides a simple tool for performing linear regression, which models the relationship between a dependent variable and one or more independent variables[53]. This class is fitted to a linear model so that it minimizes the residual sum of squares between predicted and observed outcomes. This tool can handle either single or multiple

regression tasks and uses well-establish methods from the SciPy library in its underlying architecture. **Figure 7** illustrates the formula for linear regression with 499 features[54].

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_{499} x_{499} + \varepsilon$$

Figure 7: Example of linear regression with 499 features.

Ridge Regression

The Ridge class in Scikit-learn solves regression problems by applying L2-norm regularization to control for model complexity while minimizing the linear least squares loss function (**Figure 8**)[55]. This estimator, which is sometimes referred to as Ridge Regression or Tikhonov regularization, natively supports multi-variate regression. This is an example of an embedded feature selection technique[50].

$$\left|\left|y - Xw\right|\right|_2^2 + \alpha \left|\left|w\right|\right|_2^2$$

Figure 8: Objective function that is minimized by the Ridge class.

Lasso Regression

The Lasso class in Scikit-learn uses a linear model that is trained using L1 regularization[56]. The Lasso performs the same objective optimization (**Figure 9**) as Elastic Net but uses an L1 ratio of 1.0 without any accompanying L2 regularization. This is an example of an embedded feature selection technique[50]

$$\frac{1}{2 \cdot n_{\text{samples}}} \cdot |y - Xw|_2^2 + \alpha \cdot |w|_1$$

Figure 9: Lasso optimization objective.

Mutual Information

This Scikit-learn class ('mutual_info_classif') computes mutual information (MI) between a feature set and a discrete target variable[57]. This algorithm quantifies the amount of dependency between a pair of random variables, which invariably yields a non-negative value. MI returns a value of zero when variables are entirely independent, and values increase alongside strengthening relationships between variables. This is achieved via nonparametric techniques that estimate entropy according to distances between K-nearest neighbors. These ideas originate from the Kozachenko-Leonenko entropy estimator, first proposed in 1987[58]. This is an example of a filter-based feature selection technique[50]. The equation for computing mutual information between two discreet random variables X and Y is shown in **Figure 10**[59].

$$I(X;Y) = H(X) - H(X|Y)$$

Figure 10: Formula for mutual information.

Random Forest Regression

The RandomForestRegressor takes sub-samples of a dataset, fits a series of decision tree regressors to them, then improves predictive accuracy using averaging[60]. This also helps inhibit overfitting. This class is built on top of Scikit-learn's DecisionTreeRegressor, where individual trees within the random forest apply the 'best split strategy'[60,61]. Parameters control whether sub-samples or the entire dataset are used for constructing decision trees. The equation for calculating the impurity decrease for decision trees shown in **Figure 11**.

$$\frac{N_t}{N}\left(\text{impurity} - \frac{N_{tr}}{N_t} \cdot \text{right\_impurity} - \frac{N_{tl}}{N_t} \cdot \text{left\_impurity}\right)$$

Figure 11: Formula for calculating impurities upon decision tree splitting.

Recursive Feature Elimination with Cross-Validation (RFECV)

This technique fits a Recursive Feature Elimination (RFE) selector to a user-defined number of cross-validation (CV) splits[62]. This automatically tunes the quantity of selected features. The 'scorer' class evaluates RFE performance for varying feature counts before pooling them together. The average scores are collected and the feature count with highest CV score is selected. This is an example of a wrapper-based feature selection technique[50].

Elastic Net

ElasticNet is a class within Scikit-learn that uses a variant of linear regression that incorporates both L1 and L2 regularization into its ranking pipeline[63]. When ElasticNet was used for feature importance ranking for this project, it was paired with the RFECV algorithm. ElasticNet minimizes the function shown in **Figure 12**.

$$\frac{1}{2 \cdot \text{n\_samples}}(|y - Xw|_2^2 + \alpha \cdot \text{l1\_ratio} \cdot |w|_1 + 0.5 \cdot \alpha \cdot (1 - \text{l1\_ratio}) \cdot |w|_2^2)$$

Figure 12: Objective function minimized by ElasticNet.

Logistic Regression

The LogisticRegression class in Scikit-learn uses the LIBLINEAR library (by default; a choice of four other solvers are also available) to perform regularized logistic regression.[32,64]. This algorithm excels with both dense and sparse datasets and converts any input into CSR matrices or C-ordered arrays. When dealing with multiple classes, the training algorithm can use either cross-entropy loss or a one-vs-rest (OvR) scheme. **Figure 13** shows formula for computing logistic regression for an independent variable X to the rolling mean of the dependent variable P.[65]

$$P = \frac{1}{1 + e^{-(a+bX)}}$$

Figure 13: Formula for probability in logistic regression.

Support Vector Classification (SVC)

Scikit-learn's *C*-Support Vector Classification (SVC) class is built on top of the open-source LIBSVM library[66,67]. LIBSVM has been used in diverse fields including bioinformatics, neuroimaging, natural language processing (NLP), and computer vision[67]. SVC supports multiple classes by using a one-vs-one scheme[66]. Because fitting time scales quadratically alongside sample count, this technique is sometimes infeasible for especially large datasets. SVC solves the primal problem shown in **Figure 14**[68]:

$$\text{minimize } \frac{1}{2} w^T w + C \sum_{i=1}^{n} \zeta_i$$
$$\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0, \; i = 1, \ldots, n$$

Figure 14: Primal problem solved by SVC.

And then the dual problem for the primal, where e is a vector of all ones and Q is an n-by-n positive semidefinite matrix, is[68]:

$$\text{minimize } \frac{1}{2} \alpha^T Q \alpha - e^T$$
$$\text{subject to } y^T \alpha = 0$$
$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n$$

Figure 15: Dual problem for the SVM primal.

After a solution is found for the optimization problem is found, the decision function's output for the sample x becomes **Figure 16**[68]:

$$\sum_{i \in SV} y_i \alpha_i K(x_i, x) + b$$

Figure 16: SVM decision function.

Random Forest Classification

Scikit-learn's RandomForestClassifier class uses DecisionTreeRegressor to construct individual trees using the 'best split strategy'[60,61,69]. While RandomForestRegressor is optimized for regression, this algorithm is optimized for classification tasks. Again, this class can either fit the decision tree classifiers to the entire dataset or to sub-samples. This is an example of an embedded feature selection technique[50].

Procedure for Ensemble Feature Importance Ranking

A simple preprocessing pipeline was constructed to prepare for the ensemble feature importance ranking process. First, features were extracted from the DataFrame, the 'ID' column was dropped because it is an identifier and not relevant for feature ranking. Features were sorted and those that were deemed to be categorical or high cardinality (greater than 10 unique values) were pooled. High-cardinality features were encoded using BinaryEncoder, while the remaining categorical features underwent one-hot encoding using the pandas 'get_dummies' function[70,71]. Next, the mean strategy was applied to impute missing values. Finally, Scikit-learn's StandardScaler class was used to standardize the dataset by removing the mean of the training samples and scaling to unit variance[72]. After the initial preprocessing, a Python dictionary was initialized to store feature rankings, feature names were extracted from the DataFrame, and the labels were encoded using Scikit-learn's LabelEncoder[73]. After the ranking algorithms finished running, the mean rankings were calculated for each entry in the dictionary and the output was exported to a CSV file.

Because binary encoding was applied to high-cardinality features and one-hot encoding was applied to the remaining categorical features before running the feature ranking algorithms, the feature count climbed from 499 to 539. After obtaining the rankings, when proceeding to the training stage there were two options: using the new 539-feature dataset or finding a way to decode the features while preserving the rankings, returning to the original 499 features. It was decided that due to the already high dimensionality it would not be ideal to further increase the feature count. Additionally, since the LOFO pipeline would be using 499 features, it would make it easier to draw comparisons between the two methods if they used the same feature set.

A decoding algorithm was developed that imported the ENS feature ranks CSV file alongside a decoding template containing the original unencoded features, storing them both as dictionaries. Feature names that differed between the dictionaries were identified, with encoded feature names grouped together according to similarity with names in the decoding template. Because the several encoded versions of a given feature may have been assigned very different rankings from one another, the algorithm assigned the decoded feature the lowest ranking that any of its encoded counterparts had. The final decoded feature rankings were exported as a JSON file and then used to generate the feature subsets used in training.

Leave-One-Feature-Out (LOFO) Feature Importance Ranking

Leave-One-Feature-Out (LOFO) feature selection techniques fall under the category of wrapper-based methods[74]. To achieve this, 499 model ensembles were trained using both LightGBM and XGBoost, each missing a single feature. For the LightGBM models, the original LoGoFunc parameters were used as reported by Stein and colleagues[15]. With XGBoost, the previously discussed optimal parameters discovered by hyperparameter tuning were used. Once all models were trained, the testing phase followed, and performance metrics were collected for each model ensemble. Overall accuracy was used as the guiding metric for ascertaining model

performance. The models were ranked by accuracy, where the model with the lowest accuracy was deemed to have been missing the most important feature, and vice versa.

Feature Subset Evaluations

After obtaining the two sets of feature rankings from the Ensemble and LOFO procedures, the final stage of the feature selection process involved training a total of 1,996 model ensembles. 499 models were trained for both LightGBM and XGBoost using both sets of feature rankings. Each model is trained using a feature subset containing the Top x ranked features, ranging from 1 (only containing the single-most important feature) to 499 (the entire feature-set). These four test groups will be referred to as ENS-LGBM, ENS-XGB, LOFO-LGBM, and LOFO-XGB. For clarity, the complete feature selection pipeline is illustrated in **Figure 17**. The training and testing of the models proceeded according to the General Procedure. After testing, performance metrics were collected and plotted to determine the optimal feature count for both classifiers.

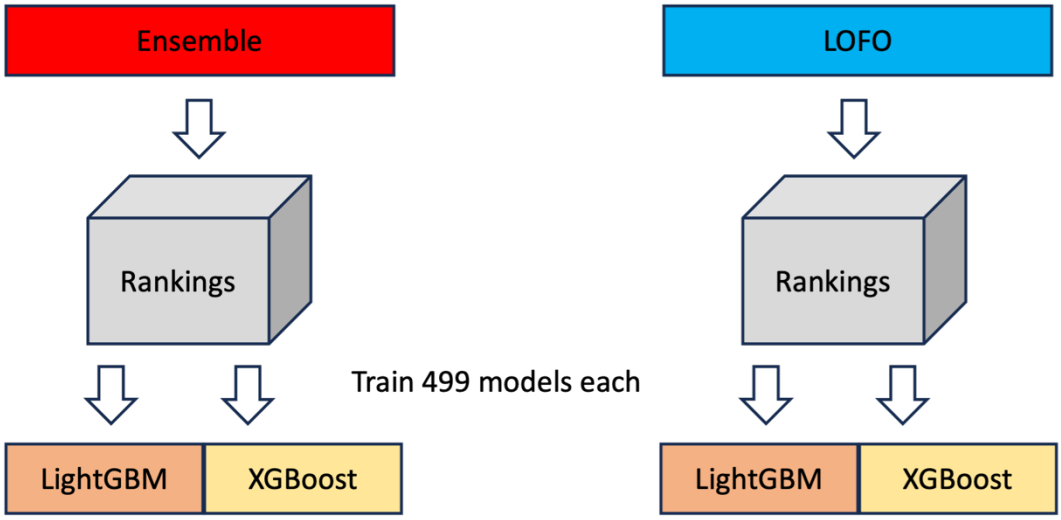


Figure 17: Illustration of the feature selection pipeline for LGBM and XGB.

Model Evaluation

After training and testing each model, a 3-class confusion matrix (**Figure 18**) was populated with the True Positive (TP), False Negative (FN), and False Positive (FP) values[75]. For any class, the True Negative (TN) is the sum of every cell outside of that class's row and column. These values were used to compute overall and class-wise performance metrics (**Table 4**). The overall metrics consisted of Accuracy (ACC), Precision (PREC), micro-averaged Recall (micro-REC), macro-averaged Recall (macro-REC), F1 Score, ROC AUC Score, and the Matthew's Correlation Coefficient (MCC). There are two choices for calculating the overall REC in a 3-class problem. Micro-REC takes the sum of the 3 *TP* values and divides by the sum of the *TP* values plus the sum of all 3 *FN* values. This weights each class equally and computes the average score for their aggregated contributions[75]. Macro-REC first calculates REC for each class individually, sums them, and then divides by the number of classes.

For the class-wise metrics, the following were collected for each class: Average Precision (AP), REC, F1 Score, and MCC. These were computed in a one-versus-rest fashion, where the target class is binarized as 1 and all other classes become 0[15]. Once the models with optimal feature count were selected for each of the four testing groups (ENS-LGBM, ENS-XGB, LOFO-LGBM, and LOFO-XGB), Receiver Operating Characteristic (ROC) curves were plotted and the Area Under the Curve (AUC) was calculated for each ROC.

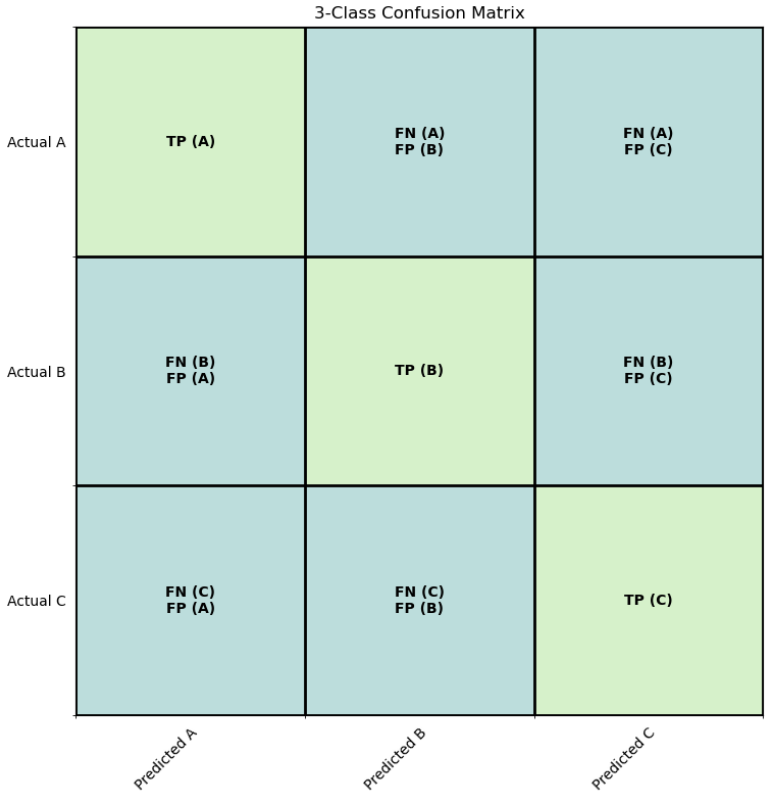


Figure 18: 3-class confusion matrix.

Table 4: Formulae for Performance Evaluation Metrics

Metric	Formula
Accuracy (ACC)[76]	$\frac{TP + TN}{TP + FN + TN + FP}$
Precision (PREC)[76]	$\frac{TP}{TP + FP}$
Recall (REC)	$\frac{TP}{TP + FN}$
Micro-averaged REC	$\frac{TP_A + TP_B + TP_C}{TP_A + TP_B + TP_C + FN_A + FN_B + FN_C}$
Macro-averaged REC	$\frac{Recall_A + Recall_B + Recall_C}{3}$
F1 Score[76]	$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$
MCC[76]	$MCC = \frac{(TP \cdot TN) - (FP \cdot FN)}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$
Avg. Precision (AP)[15]	$AP = \sum_n (R_n - R_{n-1}) P_n$
True Positive Rate (TPR)[76]	$\frac{TP}{TP + FN}$
False Positive Rate (FPR)[76]	$\frac{FP}{TP + FN}$
Area Under Curve (AUC)[77]	$\int TPR \; dFPR$

Multi-Criteria Decision-Making (MCDM)

After collecting the class-wise metrics for each of the 1,996 model ensembles, the challenge was making sense of the resulting metrics to decide which feature count was truly 'optimal' for each of the 4 ranker-classifier combinations. Since there were multiple ways of assigning relative importance to each metric, this could be conceptualized as a variant of Multi-Criteria Decision Making (MCDM) and sensitivity analysis[78]. A simple weighted sum algorithm was developed where 6 different weighting schemes (**Table 5**) were trialed, each with different relative importances assigned to the 9 class-wise performance metrics. **Scheme 1** assigns equal weights to all metrics, **Scheme 2** prioritizes the three GOF metrics, **Scheme 3** prioritizes F1 scores, **Scheme 4** emphasizes MCC, **Scheme 5** concentrates on the AP score, and **Scheme 6** prioritizes the GOF and LOF classes.

Once the score was calculated for each of the 6 weighting schemes, the models were sorted in descending order based on their scores, with the top-performing model ranked as number 1. The next step was to identify models that consistently performed well across the different ranking schemes. This helped to detect the models that gave excellent performance irrespective of which importances were placed on the various metrics. The number of times high-performing models appeared in the top 5 rankings across the various ranking schemes were tallied using Python's 'Counter' class, and the model appearing in the top 5 the most frequently was selected as optimal. An average ranking was also calculated for each model across all 6 schemes, giving additional context about the overall consistency. These two approaches provide a compromise between identifying top-performing models versus identifying models that are stable across weighting schemes. Therefore, the final feature count that was selected as optimal for each ranker-classifier combination in a manner that ensured both high performance and high stability.

Table 5: Weighting schemes that were used to make the final feature selection.

Metric	Scheme 1	Scheme 2	Scheme 3	Scheme 4	Scheme 5	Scheme 6
GOF F1	1/9	0.2	0.2	0.1	0.1	0.2
LOF F1	1/9	0.1	0.2	0.1	0.1	0.2
Neutral F1	1/9	0.1	0.2	0.1	0.1	0.1
GOF MCC	1/9	0.2	0.1	0.2	0.1	0.2
LOF MCC	1/9	0.1	0.1	0.2	0.1	0.2
Neutral MCC	1/9	0.1	0.1	0.2	0.1	0.1
GOF AP	1/9	0.2	0.1	0.1	0.2	0.2
LOF AP	1/9	0.1	0.1	0.1	0.2	0.2
Neutral AP	1/9	0.1	0.1	0.1	0.2	0.1

Results and Discussion

Classifier Assessment

Hyperparameter Tuning

The complete set of the 27 most optimal hyperparameter sets discovered by RandomizedSearchCV for each classifier are tabulated in **Appendix B**. These hyperparameters were used to train an ensemble of models. The complete search spaces explored during hyperparameter tuning are also detailed there.

Evaluation

The performance metrics for each of the classifiers are reported in the plot (**Figure 19**) and table (**Table 6**) below. When comparing LGBM and LGBM-P, all performance metrics slightly improved when the 'Protein_dom' feature was excluded. This suggests that the protein domain (obtained from Pfam or InterPro) is not an essential feature for this classification task. Across classifiers, macro-REC was universally lower than micro-REC. Since the dataset is highly imbalanced, this indicates that the majority classes (LOF and Neutral) performed better than GOF[75]. XGB consistently performed better than any other classifier for all metrics except macro-REC (0.756), where LGBM (0.816), LGBM-P (0.813), and SGD (0.768) had higher scores. This suggests that although XGB generally offered superior performance, it was not as adept at classifying the GOF class. The only other classifier to consistently offer comparable performance to LGBM was CTB, although it yielded slightly lower values for each metric. Given these results, it was decided to use XGB for further development and optimization.

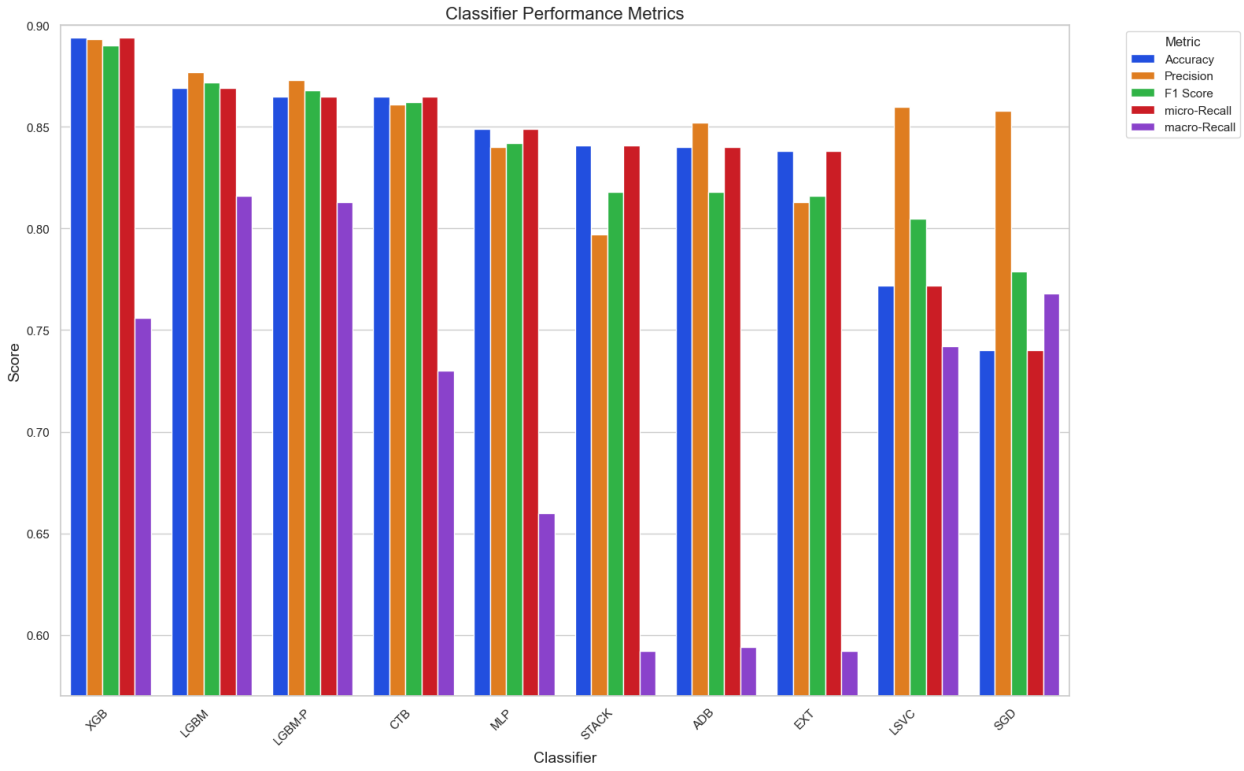


Figure 19: Overall performance metrics for each Classifier.

Table 6: Overall performance metrics for each classifier.

Classifier	Accuracy	Precision	F1-score	Micro-REC	Macro-REC
LGBM	0.869	0.877	0.872	0.869	0.816
LGBM-P	0.865	0.873	0.868	0.865	0.813
CTB	0.865	0.861	0.862	0.865	0.730
XGB	0.894	0.893	0.890	0.894	0.756
ADB	0.840	0.852	0.818	0.840	0.594
LSVC	0.772	0.860	0.805	0.772	0.742
SGD	0.740	0.858	0.779	0.740	0.768
ECT	0.838	0.813	0.816	0.838	0.592
STACK	0.841	0.797	0.818	0.841	0.592
MLP	0.849	0.840	0.842	0.849	0.660

Results for Ensemble Feature Importance Ranking

After calculating the mean feature importance rankings across all 12 ensemble algorithms, the 25 most highly ranked features were plotted (**Figure 20**) against their mean ranking scores. The feature descriptions are listed in **Table 7**. It is apparent from the plot that the 'Consequence_first_0' feature was strongly favored by the ensemble. This feature is derived from Ensembl's Variant Effect Predictor (VEP), a tool that annotates and analyzes variant consequences[19]. The 'IMPACT' feature also originates from VEP, where it describe the modifier impact variant consequence[79]. The 'ConsScore' feature is the score that was assigned to the VEP consequence. It can be reasoned that these features occupied the top 3 rankings because they are intrinsically connected, having all been sourced from the same database entries. The next feature, 'VEST4', is the score from Variant Effect Scoring Tool (VEST4), a supervised ML classifier that assess missense variants[80]. The 'verPhyloP' feature is the non-human vertebrate phyloP score, which comes from the PHAST suite of phylogenetic analysis tools[81]. 'gnomAD_exomes_AF' describes the sample's allele frequency within the exomic dataset of the Genome Aggregation Database (gnomAD)[18]. The full ENS rankings are listed in **Appendix C**.

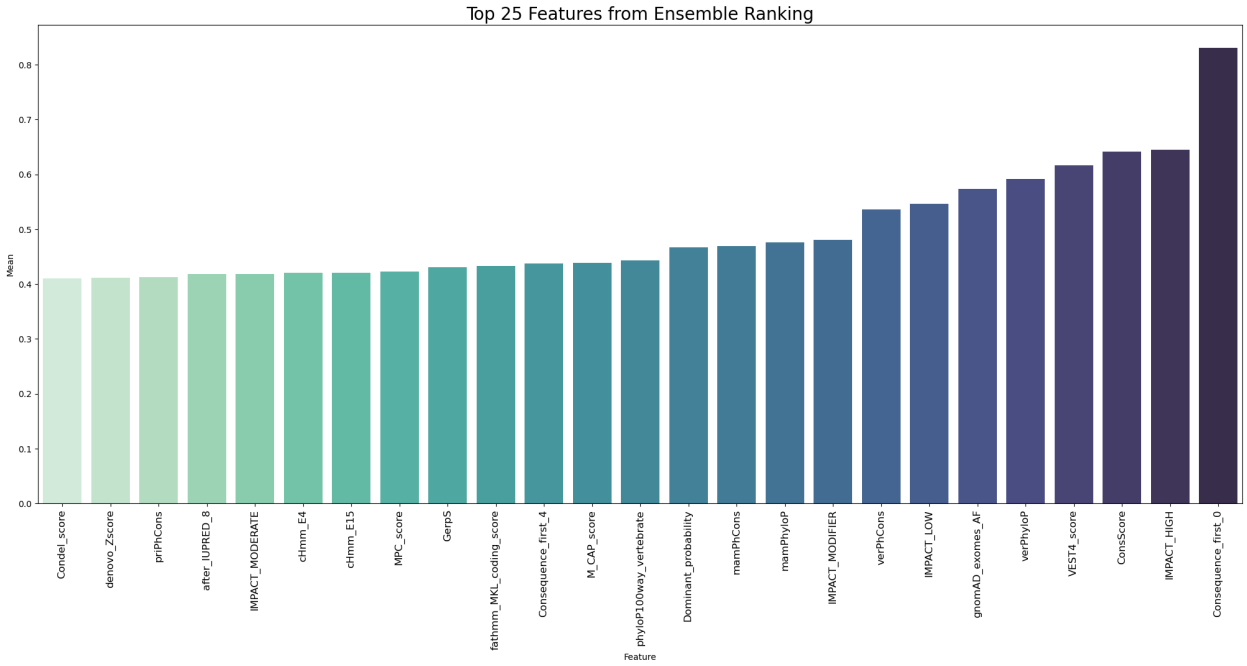


Figure 20: Top 25 features from ENS importance ranking.

Table 7: Descriptions of the most important ENS features.

Rank	Feature Name	Description
1	Consequence_first	First reported consequence from VEP
2	IMPACT	The VEP impact modifier for the consequence type
3	ConsScore	Score assigned to VEP consequence
4	VEST4_score	VEST4 score
5	verPhyloP	Vertebrate PhyloP score (excluding human)
6	gnomAD_exomes_AF	Genome Aggregation Database allele frequency (AF) of Exome dataset
7	verPhCons	Vertebrate PhastCons conservation score (excl. human)
8	mamPhyloP	Mammalian PhyloP score (excl. human).
9	mamPhCons	Mammalian PhastCons conservation score (excl. human)
10	Dominant_probability	Prediction of variants pathogenic for autosomal dominant disease
11	phyloP100way_vertebrate	Conservation score based on multiple alignments of 100 vertebrate genomes including human
12	M_CAP_score	M-CAP score
13	fathmm_MKL_coding_score	FATHMM-MKL score
14	GerpS	GERP++ S score
15	MPC_score	MPC score
16	cHmm_E15	Number of 48 cell types in chromHMM state E15
17	cHmm_E4	Number of 48 cell types in chromHMM state E4
18	after_IUPRED_8	Prediction of intrinsically unstructured proteins downstream of region 8
19	priPhCons	Primate PhastCons conservation score (excl. human)
20	denovo_Zscore	Estimate of the rate of de novo mutation per gene and whether a given gene contains more de novo mutations than expected by chance alone
21	Condel_score	Condel score
22	MVP_score	MVP score
23	after_IUPRED_15	Prediction of intrinsically unstructured proteins downstream of region 15
24	phastCons100way_vertebrate	Conservation score based on multiple alignments of 100 vertebrate genomes including human
25	Recessive_probability	Prediction of variants pathogenic for autosomal recessive disease

To gain further insight into the stability and relative importances of the mean ranking scores (MRS) across the 12-algorithms, they were scatter-plotted alongside their variances (**Figure 21**)[82]. Lower variances indicate greater consistency across algorithms. The plot indicates a positive correlation between mean ranking scores and variances, a near-linear relationship with some of the most highly ranked features existing as outliers. 'Consequence_first_0' (MRS: 0.831) had a variance of only 0.10, which further illustrates that this feature was highly favored across algorithms. In general, it can be concluded that there was greater disagreement among the ranking algorithms for highly important features than for less important features. This provides justification for using diverse ensembles of ranking algorithms.

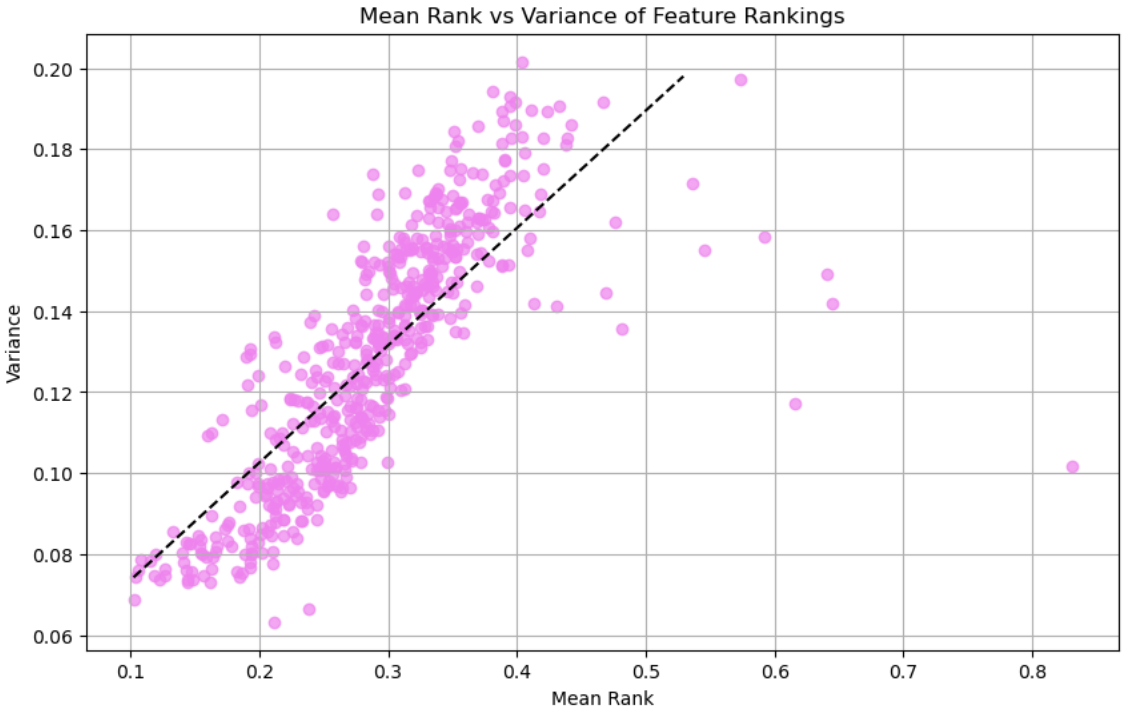


Figure 21: ENS mean rankings plotted against variances.

A violin plot (**Figure 22**) was used to further investigate importance ranking trends for each algorithm[83]. It is clear from the plot that there is a high amount of disagreement about the rankings of individual features across the 12 algorithms. This is likely an artifact of the high degree of dimensionality intrinsic to a dataset with 499 features. The plot clearly illustrates that only 4 of the RFECV algorithms achieved a uniform distribution of ranking scores: ElasticNet, Linear Regression, SVC, and RandomForestClassifier. The rest of the algorithms have distributions that are strongly skewed toward the low end of the ranking scale, except for RFECV Logarithmic Regression, which is strongly skewed toward the upper end of the scale. The distribution of rankings within the mean rankings dictionary are skewed toward the low end of the scale, likely due to the influence of the 7 algorithms that also had their distributions skewed in this manner. This suggests that for this dataset, it may be more lucrative to construct a 4-algorithm ensemble using only the 4 RFECV methods that yielded uniform ranking distributions.

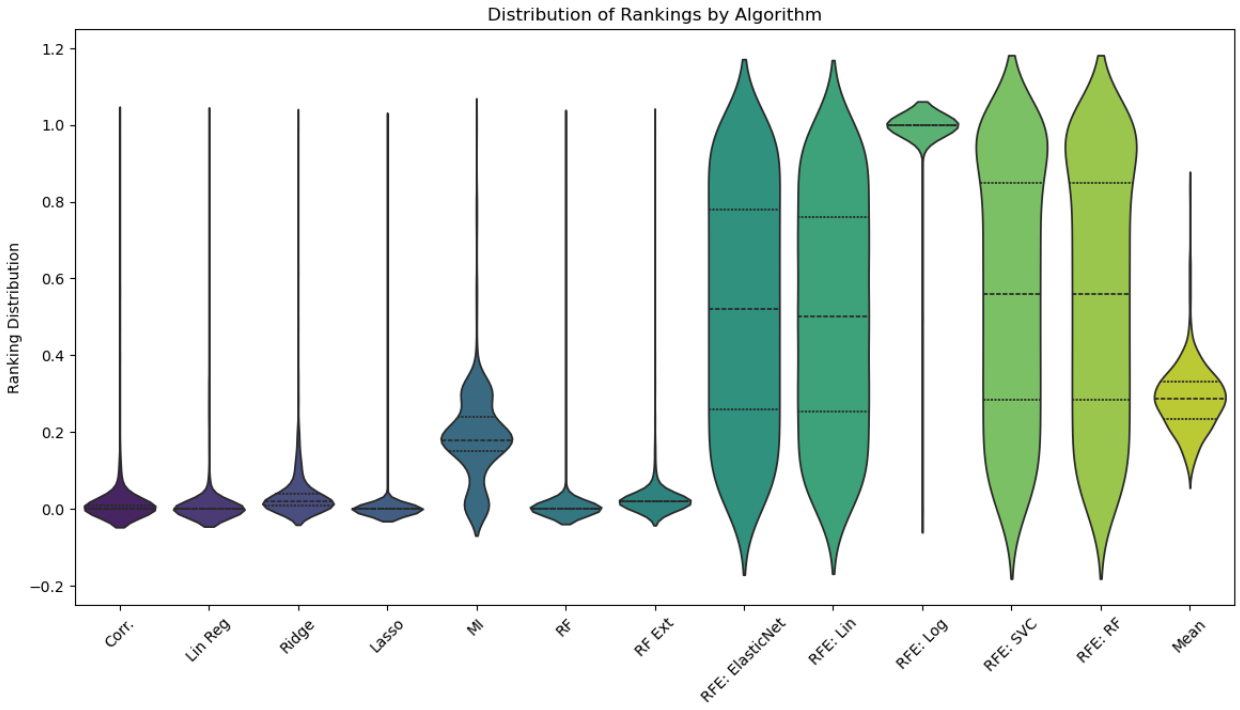


Figure 22: Violin plot of ENS importance ranking distributions across algorithms and mean.

Results for LOFO Feature Importance Ranking

After collecting metrics for all 499 model ensembles, they were sorted in reverse order according to their overall accuracy metrics. The model ensemble with the lowest ACC was deemed to have been missing the most important feature, and this process continued until the least important feature was reached. The 25 most important features are listed in **Table 8**. As can be seen in **Figure 23**, the feature 'gnomAD_exomes_AF' was significantly more important than any other feature. This was also identified as the sixth most important feature by ENS, illustrating its importance across ranking methods. This feature is derived from the Exome dataset of the Genome Aggregation Database (genomAD)[18]. The 'before_ANCHOR_15' and 'before_ANCHOR_3' features are derived from the ANCHOR tool, which makes predictions about protein binding regions in disordered proteins[84]. 'MMSp_acceptorintron' is an acceptor intron score provided by the MMSplice framework, which is based on neural networks[85]. 'Dist2Mutation' is a measure of the gnomAD SNV distances[86]. 'after_IUPRED_15' is one of only two features to appear in the top 25 of both LOFO and ENS, although in different positions. This is a metric provided by IUPred2A, a tool for identifying disordered protein regions[87]. Interestingly, there were no features with identical rankings between LOFO and ENS. The full LOFO rankings are listed in **Appendix D**.

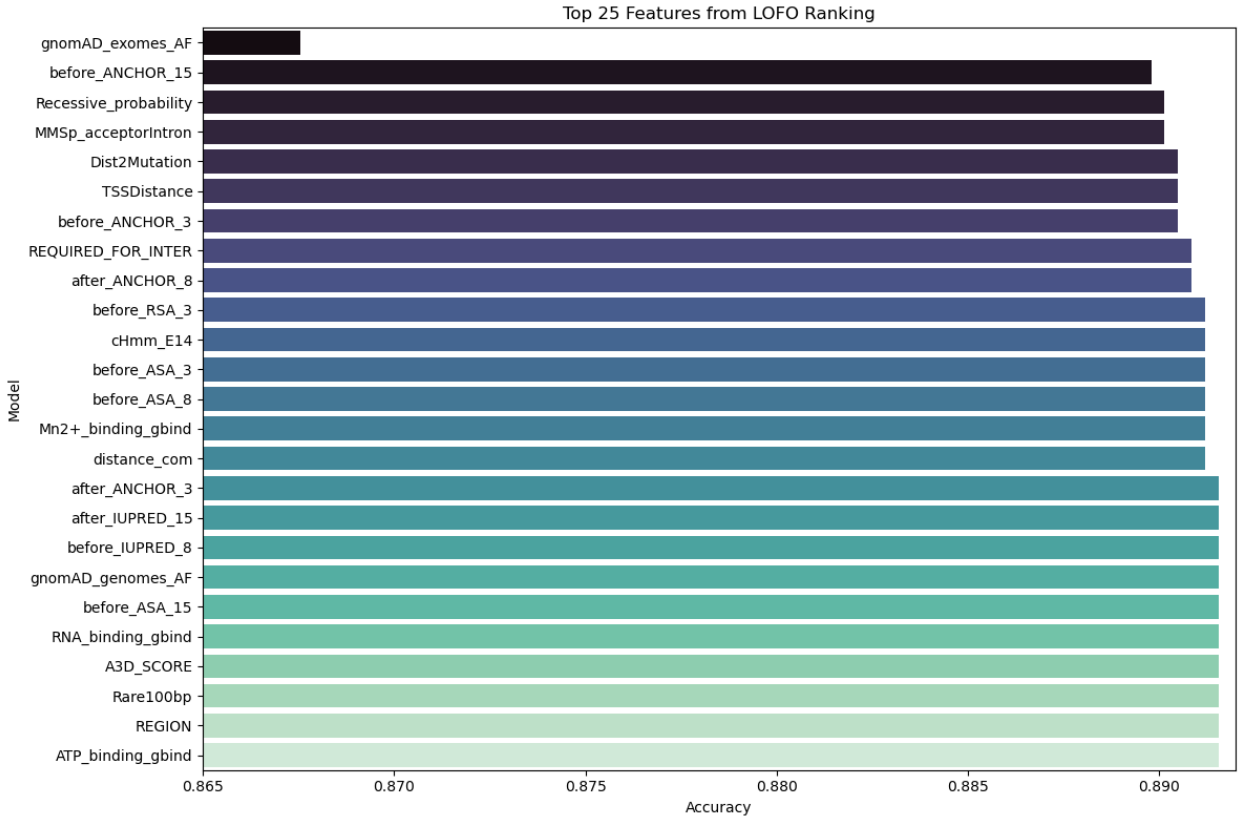


Figure 23: Top 25 features from LOFO feature importance ranking.

Table 8: Descriptions for most important LOFO features.

Rank	Feature Name	Description
1	gnomAD_exomes_AF	Genome Aggregation Database allele frequency (AF) of Exome dataset
2	before_ANCHOR_15	Disordered binding region prediction upstream of region 15
3	Recessive_probability	Prediction of variants pathogenic for autosomal recessive disease
4	MMSp_acceptorIntron	MMSplice acceptor intron (intron 3) score
5	Dist2Mutation	Distance between the closest BRAVO SNV up and downstream (position itself excluded)
6	TSSDistance	The distance from the transcription start site for upstream variants
7	before_ANCHOR_3	Disordered binding region prediction upstream of region 3
8	REQUIRED_FOR_INTER	UniProt region required for interaction
9	after_ANCHOR_8	Disordered binding region prediction downstream of region 8

10	before_RSA_3	Relative solvent accessibility upstream of region 3
11	cHmm_E14	Number of 48 cell types in chromHMM state E14
12	before_ASA_3	Absolute solvent accessibility upstream of region 3
13	before_ASA_8	Absolute solvent accessibility upstream of region 8
14	Mn2+_binding_gbind	GraphBind ligand score
15	distance_com	The residue distance from the protein center of mass calculated from AlphaFold models
16	after_ANCHOR_3	Disordered binding region prediction downstream of region 3
17	after_IUPRED_15	Prediction of intrinsically unstructured proteins downstream of region 15
18	before_IUPRED_8	Prediction of intrinsically unstructured proteins upstream of region 8
19	gnomAD_genomes_AF	Genome Aggregation Database allele frequency (AF) of Genome dataset
20	before_ASA_15	Absolute solvent accessibility upstream of region 15
21	RNA_binding_gbind	GraphBind ligand score
22	A3D_SCORE	Aggrescan3D score calculated on AlphaFold models
23	Rare100bp	Number of rare (MAF < 0.05) BRAVO SNV in 100 bp window nearby
24	REGION	Uniprot region
25	ATP_binding	GraphBind ligand score

Results from feature subset evaluations

ENS-XGB Results

The overall performance metrics for the ENS-XGB models are shown in **Figure 24**, where feature count represents the Top x most important features according to the ENS ranker. The highest score and corresponding feature count are marked for each metric, and they are also displayed in **Table 9**. While most metrics required more than 400 features to achieve peak performance, macro-REC peaked early (0.815) with only 33 features and then sharply declined. The metrics generally increased with feature count until reaching the Top 50 features, where performance dropped. They then gradually increased in performance before the best ROC AUC score was observed (0.945) with 402 features. ACC, PREC, F1, and MCC all had the highest scores with 496 features. This suggests that this ranker-classifier combination may not be useful for significantly reducing the starting feature count of 499 without sacrificing performance.

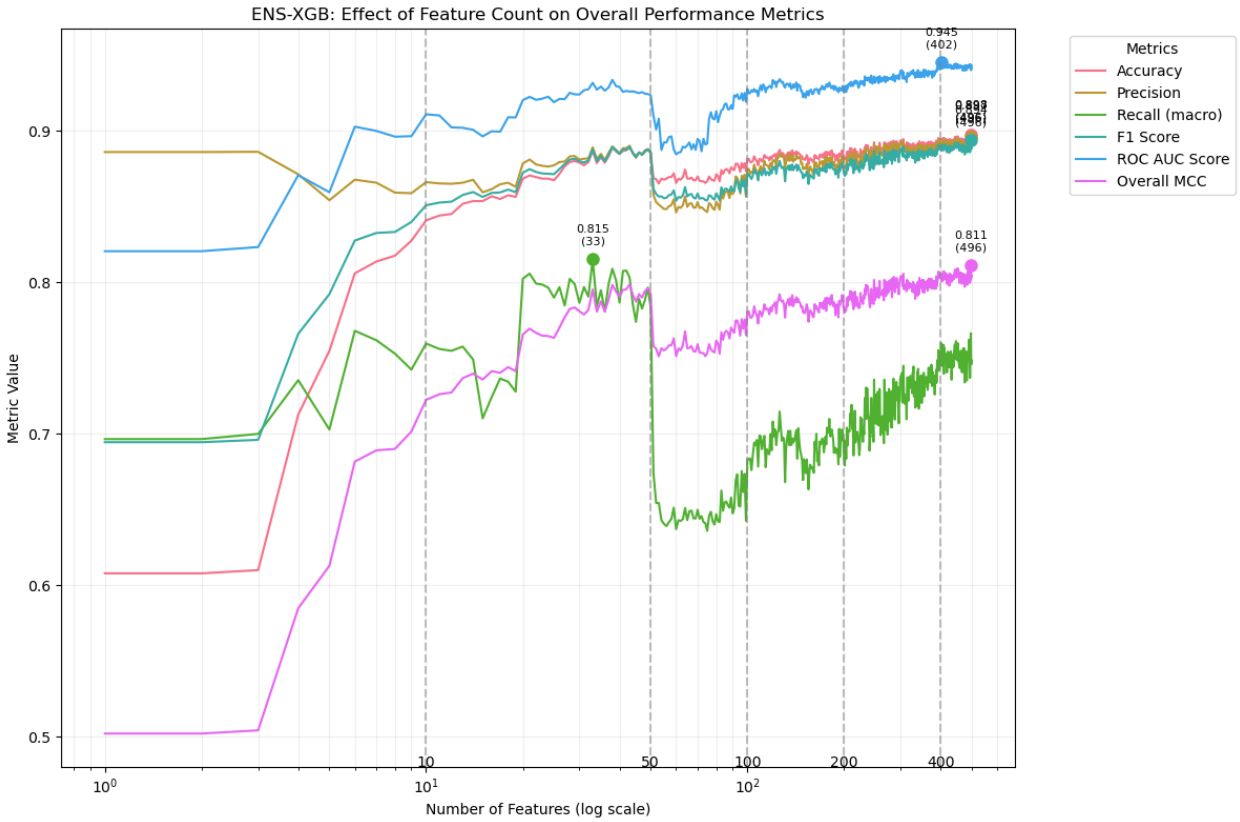


Figure 24: Plot of ENS-XGB overall performance metrics for each feature count.

Table 9: Feature counts with highest scores for each overall ENS-XGB metric.

Metric	Best feature count	Score
ACC	496	0.898
PREC	496	0.897
Macro-REC	33	0.815
F1 score	496	0.894
ROC AUC score	402	0.945
MCC	496	0.811

The class-wise metrics for ENS-XGB at each feature count are shown in **Figure 25**, with the highest-performing feature counts identified in **Table 10**. GOF consistently underperformed compared to LOF or Neutral across all metrics. GOF recall had the highest score (0.910) using only the most highly ranked feature, before sharply declining. GOF also had the steepest decline in performance at the 50-feature mark, suggesting that poor GOF performance was primarily responsible for this same pattern in the overall metrics. For all classes, more than 400 features

were required to achieve maximum AP score. GOF (0.596) and LOF (0.801) needed 496 features to achieve maximum MCC score, while the Neutral class (0.863) only needed 64 features.

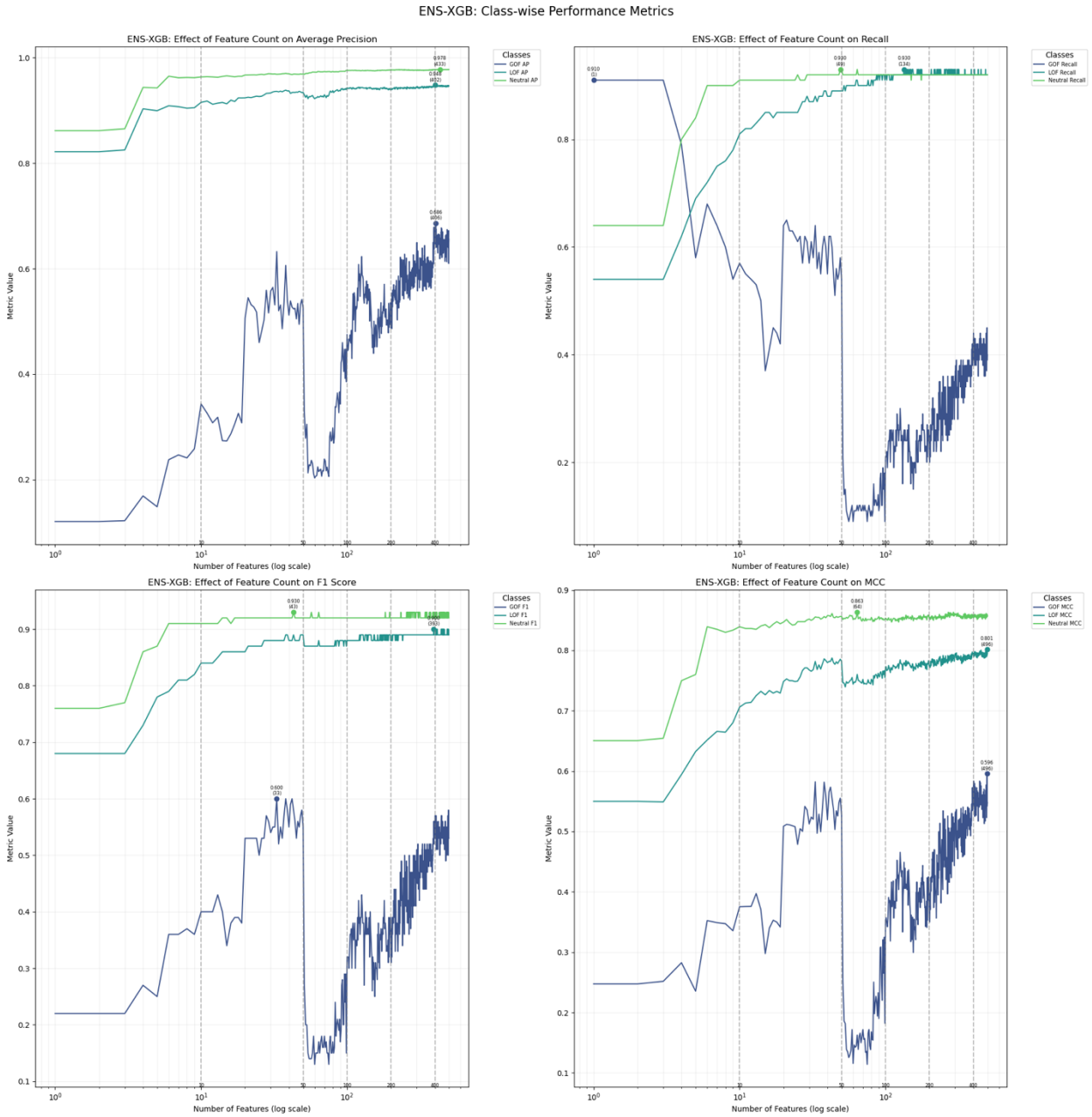


Figure 25: Class-wise ENS-XGB performance metrics for each feature count.

Table 10: Feature counts with highest class-wise metric scores for ENS-XGB.

Metric	GOF	LOF	Neutral
AP	406 (0.686)	402 (0.948)	433 (0.978)
REC	1 (0.910)	134 (0.930)	49 (0.930)
F1 Score	33 (0.600)	393 (0.900)	43 (0.930)
MCC	496 (0.596)	496 (0.801)	64 (0.863)

ENS-LGBM Results

The overall performance metrics for the ENS-LGBM models are shown in **Figure 26**, with the best-performing models for each metric listed in **Table 11**. While the top scores were lower than those for ENS-XGB, fewer features were generally required to achieve maximum scores. PREC was highest (0.883) in the 3-feature model, before gradually trending downward. ACC, F1-score, and MCC peaked with 414 features, while macro-REC and AUC peaked at 430 features.

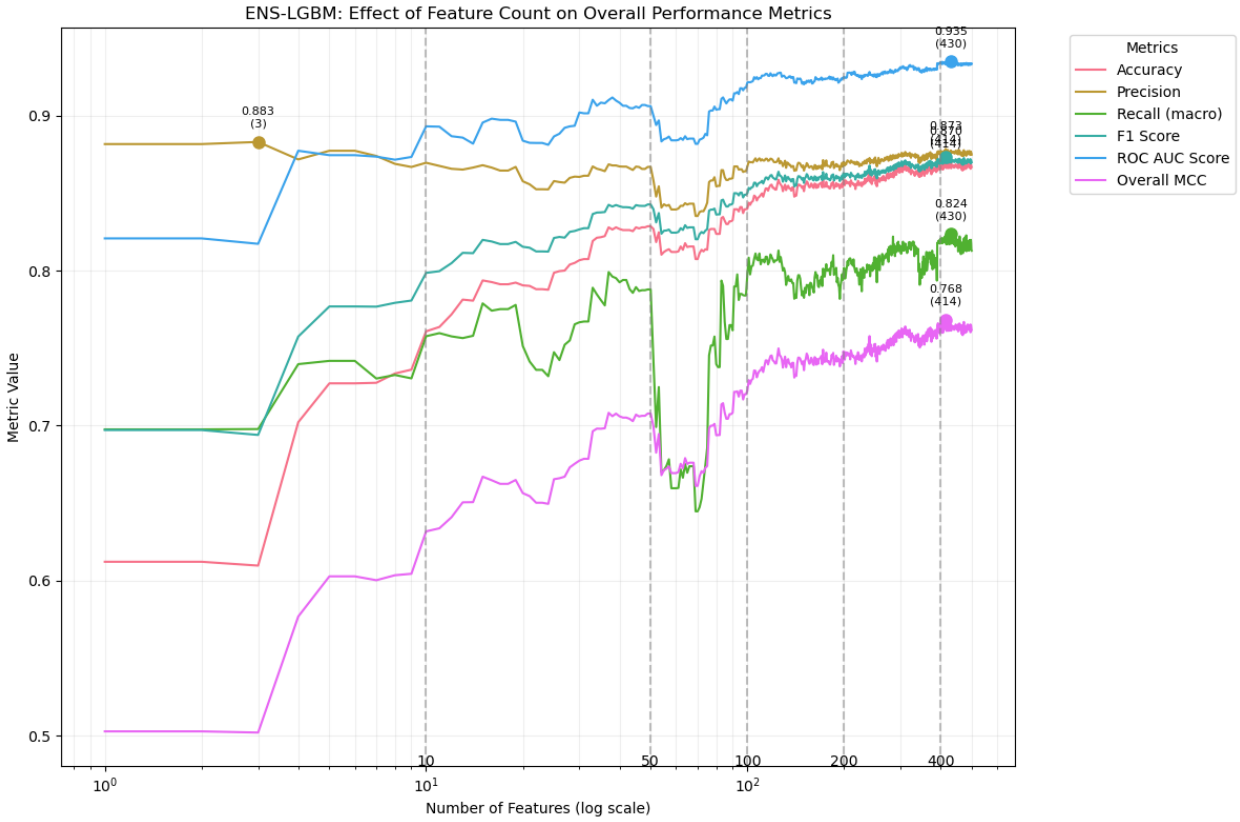


Figure 26: Plot of ENS-LGBM feature count for each overall performance metric.

Table 11: Feature counts with highest scores for each overall ENS-LGBM metric.

Metric	Best feature count	Score
ACC	414	0.870
PREC	3	0.883
Macro-REC	430	0.824
F1 score	414	0.873
ROC AUC score	430	0.935
MCC	414	0.768

The class-wise metrics for ENS-LGBM are shown in **Figure 27**, with highest-scoring models shown in **Table 12**. As with ENS-XGB, GOF generally underperformed compared to LOF and Neutral, except for REC where GOF had the highest performance from 1 to 10 features, peaking at 3 (0.910). As with the overall metrics, ENS-LGBM required fewer features to achieve maximum performance for most class-wise metrics.

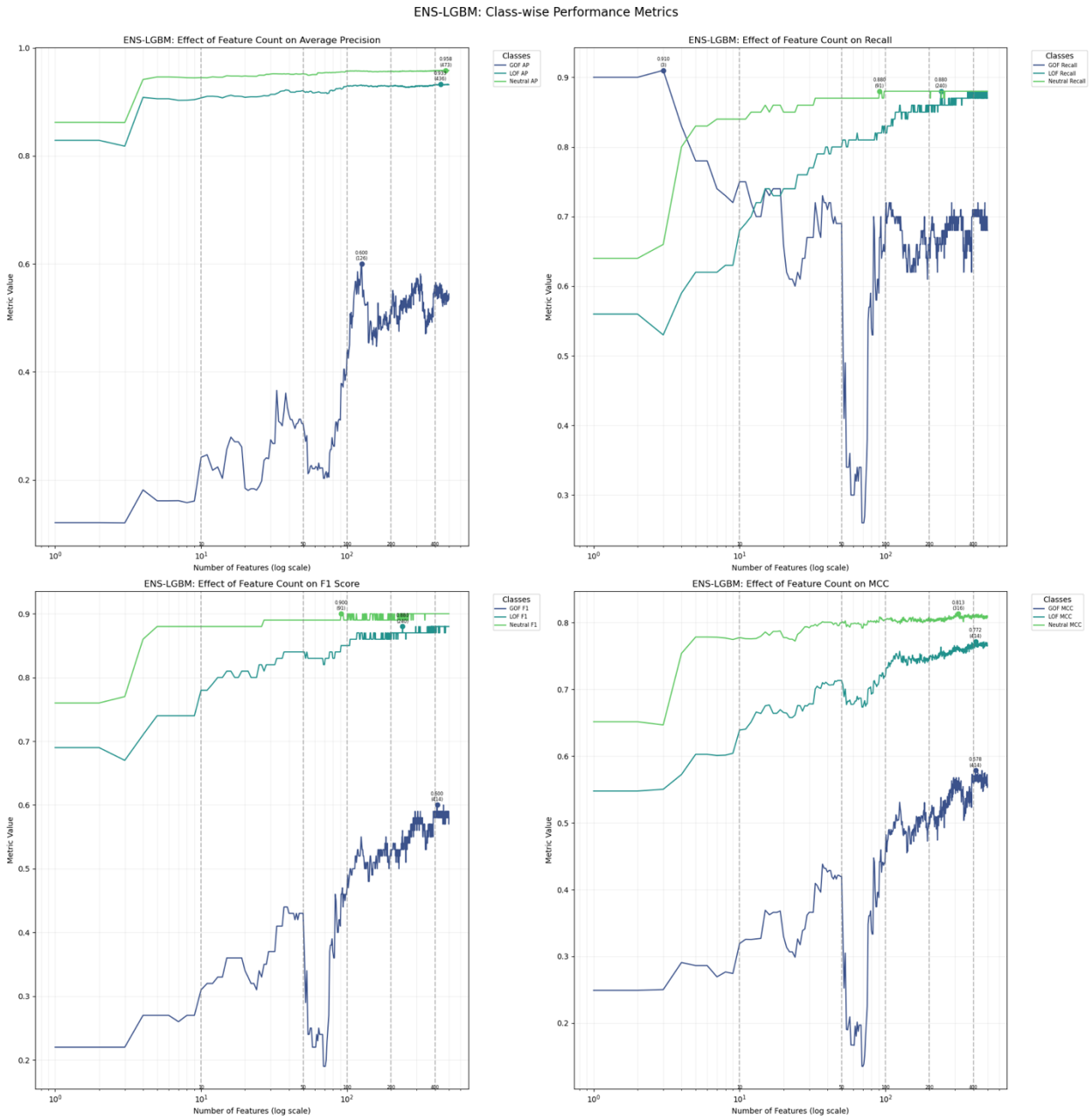


Figure 27: Class-wise performance metrics for ENS-LGBM.

Table 12: Feature counts with highest class-wise metric scores for ENS-LGBM.

Metric	GOF	LOF	Neutral
AP	126 (0.600)	436 (0.933)	473 (0.958)
REC	3 (0.910)	240 (0.880)	91 (0.880)
F1 Score	414 (0.600)	240 (0.880)	91 (0.900)
MCC	414 (0.578)	414 (0.772)	316 (0.813)

LOFO-XGB Results

The overall performance metrics for the LOFO-XGB models are shown in **Figure 28**, with the best-performing models for each metric listed in **Table 13**. It is clear that this ranker-classifier combination yielded models that required fewer features to achieve maximum performance than either ENS method. Additionally, the highest scores for each metric were universally higher than those observed with the ENS methods. ACC (0.906), PREC (0.906), and F1 score (0.905) were maximized at 269 features, while AUC score (0.948) required 389.

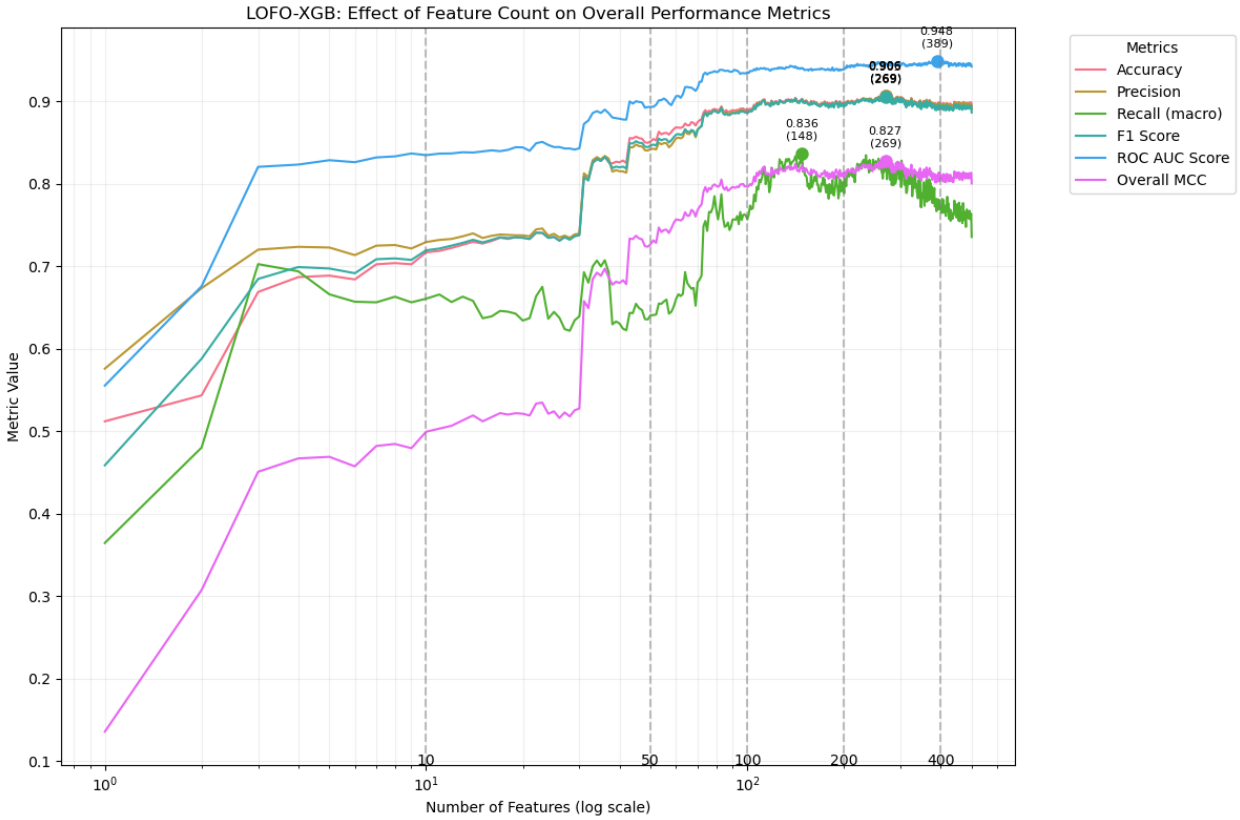


Figure 28: Plot of LOFO-XGB feature count for each overall performance metric.

Table 13: Feature counts with highest scores for each overall LOFO-XGB metric.

Metric	Best feature count	Score
ACC	269	0.906
PREC	269	0.906
Macro-REC	148	0.836
F1 score	269	0.905
ROC AUC score	389	0.948
MCC	269	0.827

The class-wise metrics for LOFO-XGB are shown in **Figure 29**, with highest-scoring models shown in **Table 14**. Although GOF still underperformed compared to LOF and Neutral, it is clear from the graph that the gap in performance between GOF and LOF/Neutral is narrower. As with the overall metrics, LOFO-XGB scores were generally higher than ENS-XGB or ENS-LGBM while requiring fewer features. The same pattern of REC (0.780) peaking with a low feature count (3) was observed, while the majority of performance metrics reached their peaks with fewer than 400 features. One outlier was Neutral AP (0.978), which was observed with 498 features.

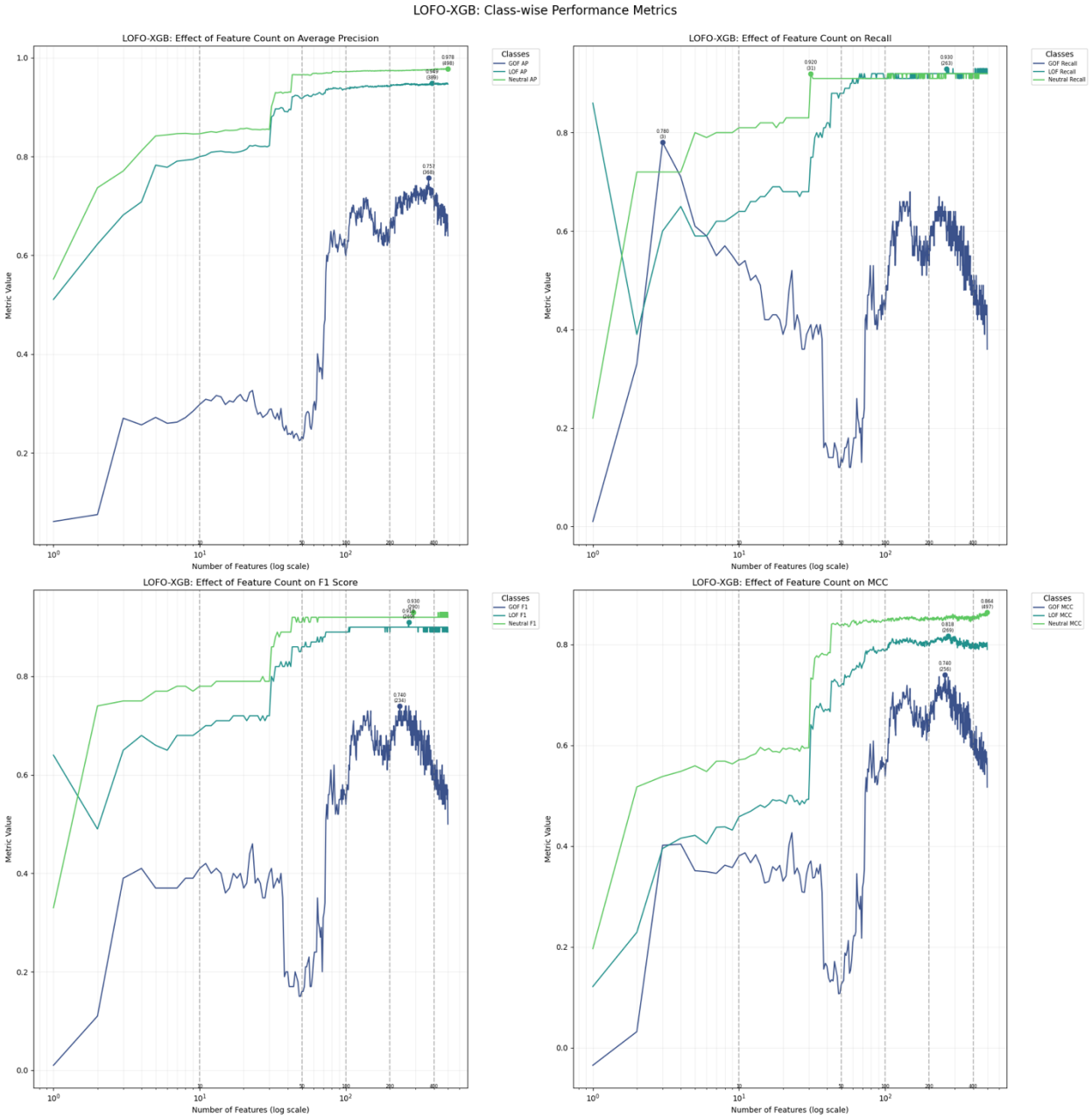


Figure 29: Class-wise performance metrics for LOFO-XGB.

Table 14: Feature counts with highest class-wise metric scores for LOFO-XGB.

Metric	GOF	LOF	Neutral
AP	368 (0.757)	389 (0.949)	498 (0.978)
REC	3 (0.780)	263 (0.930)	31 (0.920)
F1 Score	234 (0.740)	269 (0.910)	290 (0.930)
MCC	456 (0.740)	269 (0.818)	497 (0.864)

LOFO-LGBM Results

The overall performance metrics for the LOFO-LGBM models are shown in **Figure 30**, with the best-performing models for each metric listed in **Table 15**. This series generally performed more poorly than LOFO-XGB but were superior to the ENS series. Every performance metric required more features to reach its maximum value than LOFO-XGB, and all maximum scores were lower except for marginal increases in macro-REC (0.855) and AUC score (0.952).

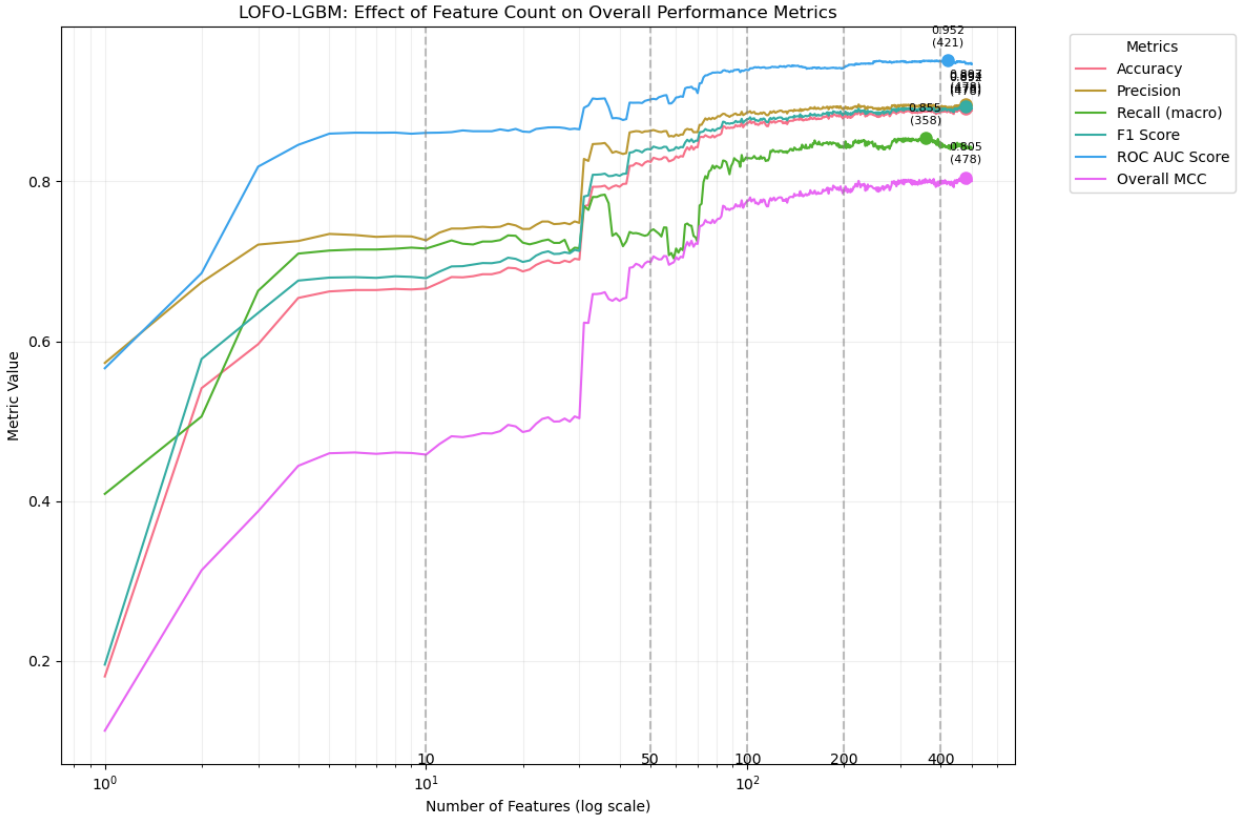


Figure 30: Plot of LOFO-LGBM feature count for each overall performance metric.

Table 15: Feature counts with highest scores for each overall LOFO-LGBM metric.

Metric	Best feature count	Score
ACC	478	0.892
PREC	478	0.897
Macro-REC	358	0.855
F1 score	478	0.894
ROC AUC score	421	0.952
MCC	478	0.805

The class-wise metrics for LOFO-LGBM are shown in **Figure 31** with highest-scoring models shown in **Table 16**. This series was inferior to LOFO-XGB for every metric, aside from the outlier where GOF REC once again peaks using only the single best feature, outperforming LOF/Neutral for a brief time, and then finally underperforming again. For GOF classification, LOFO-LGBM outperformed both ENS-XGB and ENS-LGBM, but ENS-XGB was slightly better at Neutral classification.

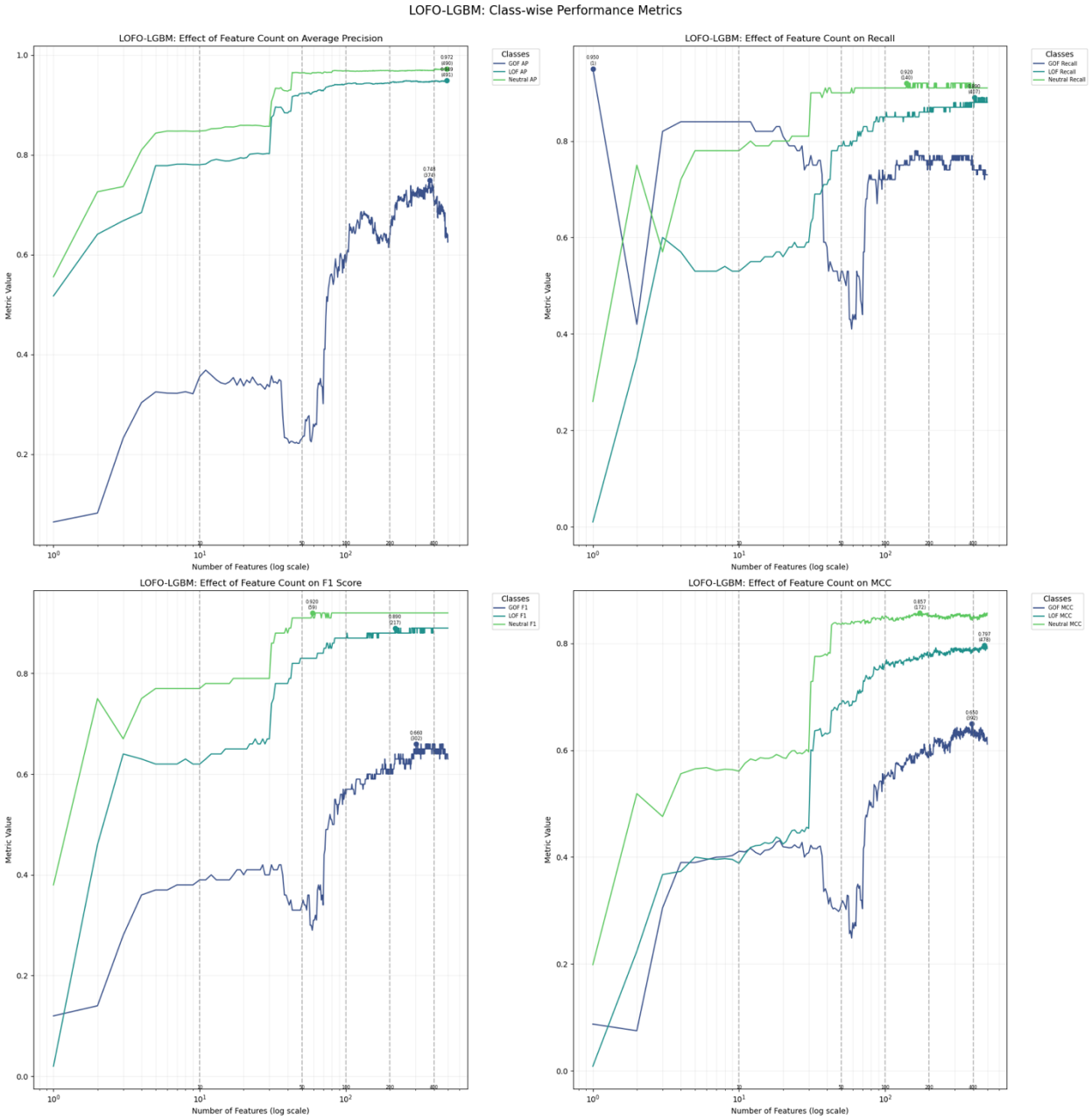


Figure 31: Class-wise performance metrics for LOFO-LGBM.

Table 16: Feature counts with highest class-wise metric scores for LOFO-LGBM.

Metric	GOF	LOF	Neutral
AP	374 (0.748)	491 (0.949)	490 (0.972)
REC	1 (0.950)	407 (0.890)	140 (0.920)
F1 Score	302 (0.660)	217 (0.890)	59 (0.920)
MCC	392 (0.650)	478 (0.797)	172 (0.857)

Final Feature Selection

Using the previously described weighting scheme, the models were ranked within each scheme, sensitivity analysis was performed to identify models that were successful across schemes, and the top-performing model ensemble was identified for all four ranker-classifier combinations (**Table 17**). The performance metrics reported by the developers of LoGoFunc are also included[15]. The full results listing the 5 best models for each weighting scheme are listed in **Appendix E**. The ROC curves for ENS-XGB and ENS-LGBM are in **Figure 32** and class-wise AUC values are in **Table 18**. The ROC curves for LOFO-XGB and LOFO-LGBM are in **Figure 33** and class-wise AUC values are in **Table 19**.

ENS-XGB consistently outperformed ENS-LGBM, although ENS-XGB required more features (496) than ENS-LGBM (414) to achieve optimal performance. LOFO-LGBM required more features (358) than LOFO-XGB (256) to reach optimal performance. For the GOF class, LOFO-XGB demonstrated higher MCC (0.74) and F1 score (0.74) while LOFO-LGBM achieved higher REC (0.77) and AP (0.74). For the LOF class, both models performed similarly, with LOFO-XGB exhibiting higher REC (0.92), F1 (0.90), and MCC (0.82). For the Neutral class, both models performed equally well across all metrics. Both LOFO methods offered significantly higher GOF AP, MCC, and F1 scores than either ENS method or the literature values.

Table 17: Class-wise metrics for the best model from each ranker-classifier combination.

Metric	LGBM (Lit.[15])	LOFO-XGB	LOFO-LGBM	ENS-XGB	ENS-LGBM
Feature Count	500	256	392	496	414
GOF REC	—	0.65	0.77	0.45	0.71
LOF REC	—	0.92	0.88	0.92	0.88
Neutral REC	—	0.92	0.91	0.92	0.88
GOF F1	0.56	0.74	0.66	0.58	0.60
LOF F1	0.87	0.90	0.89	0.90	0.88
Neutral F1	0.89	0.92	0.92	0.93	0.90
GOF MCC	0.54	0.74	0.65	0.60	0.58
LOF MCC	0.75	0.82	0.79	0.80	0.77
Neutral MCC	0.80	0.86	0.85	0.86	0.81
GOF AP	0.52	0.72	0.74	0.67	0.54
LOF AP	0.93	0.95	0.95	0.95	0.93
Neutral AP	0.96	0.97	0.97	0.98	0.96

Table 18: ROC AUC scores for ENS-XGB-496 and ENS-LGBM-414.

Class	ENS-XGB-496	ENS-LGBM-414
Neutral	0.976	0.955
GOF	0.943	0.932
LOF	0.961	0.945
Average	0.960	0.944

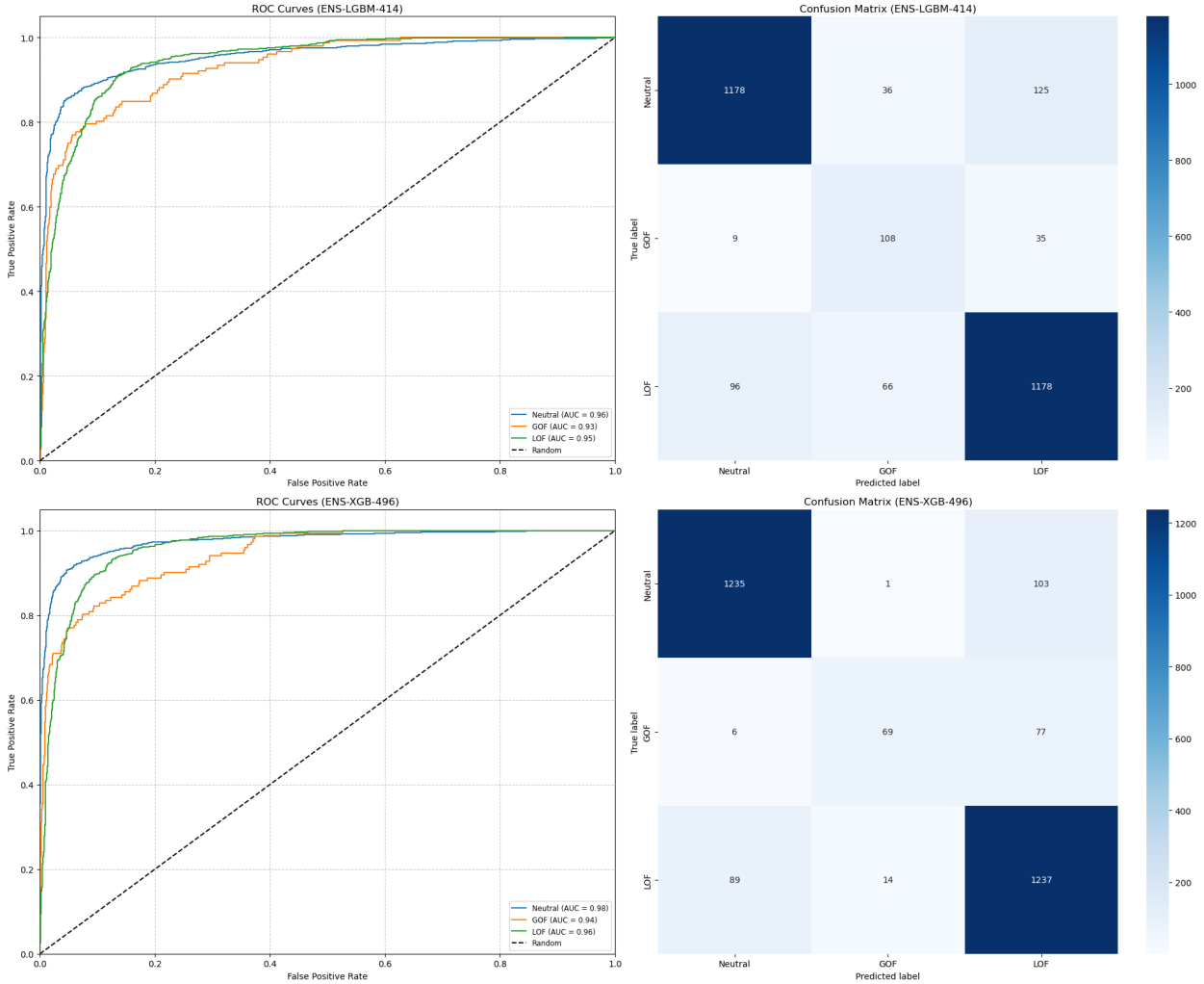


Figure 32: ROC curves and confusion matrices for ENS-XGB-496 and ENS-LGBM-414.

Table 19: ROC AUC scores for LOFO-XGB-256 and LOFO-LGBM-392.

Class	LOFO-XGB-256	LOFO-LGBM-392
Neutral	0.973	0.972
GOF	0.947	0.949
LOF	0.959	0.956
Average	0.960	0.959

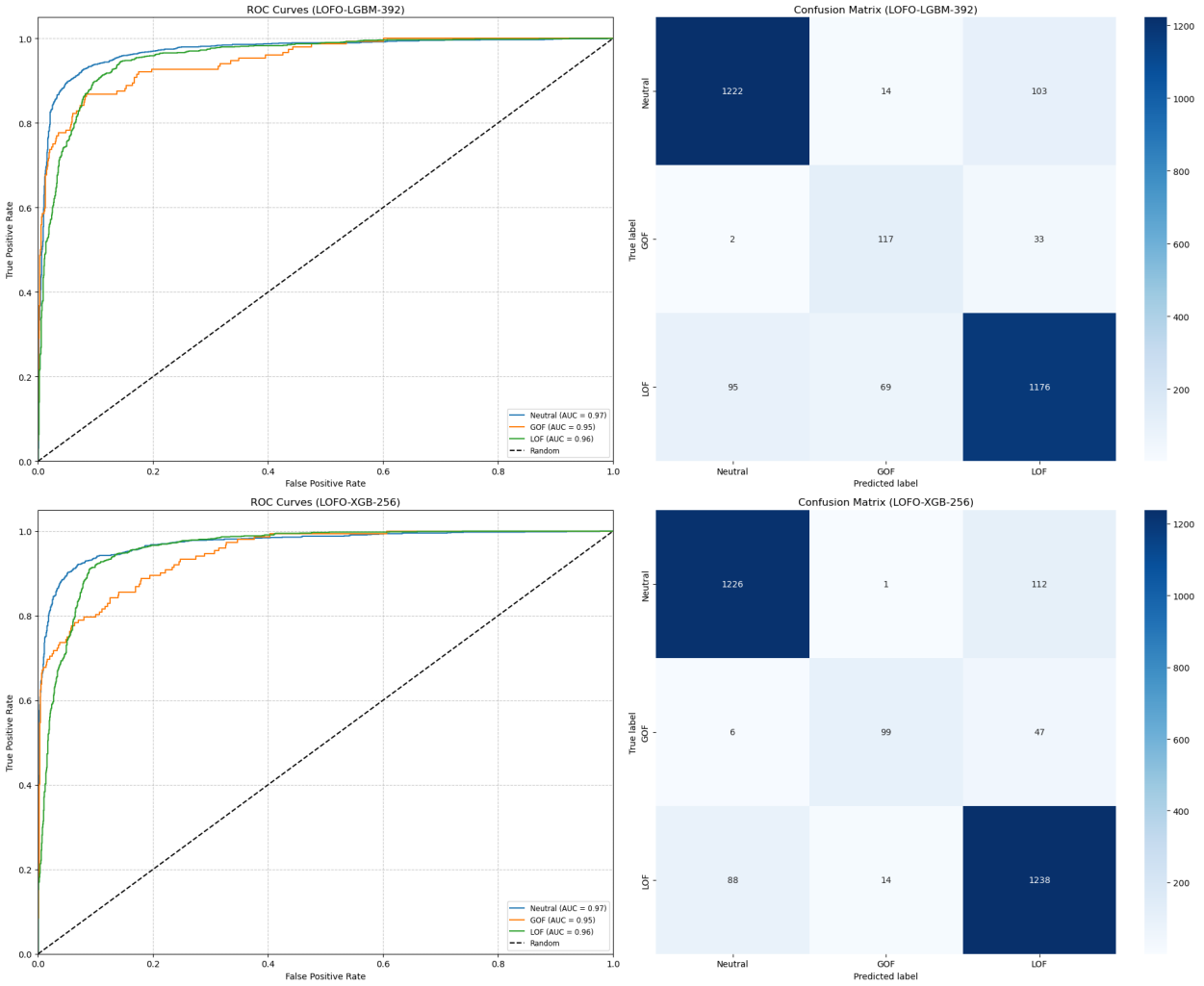


Figure 33: ROC curves and confusion matrices for LOFO-XGB-256 and LOFO-LGBM-392.

Conclusions

In summary, a series of 8 classifiers were trained to make predictions about GOF/LOF mutations, and their performances were compared to LGBM. XGB emerged as the only candidate to offer performance superior to LGBM. Next, feature importances were ranked using ENS and LOFO. Analyses revealed that ENS ranking scores were unevenly distributed across algorithms, although plotting mean scores against variance showed that the top ranked features were consistently favored. The LOFO rankings were completely different, with no rankings shared between LOFO and ENS, although two features appeared in the Top 25 most important features for both methods: 'gnomAD_exomes_AF' and 'after_IUPRED_15'. There is no obvious connection between these two features. With the ENS ranker, Consequence_first' was the most highly ranked feature, and it was closely flanked by two other features derived from VEP: 'IMPACT' and 'ConScore'. It is unusual that LOFO did not also prioritize these other VEP-related features, which instead were assigned rankings 116 and 405, respectively.

Next, the two ranked feature-sets were used to generate subsets, each containing the Top "X" most important features. 499 models were trained for both XGB and LGBM using these two feature subsets. They were evaluated, and their overall and class-wise metrics were compared. Although it was possible to see general trends from charts and tables, selecting the overall optimal feature count for each ranker-classifier combination was challenging. A weighted sensitivity analysis scheme was devised that ranked models according to 6 criteria, compared performance across criteria, and made a final selection.

LOFO-XGB emerged as the favored method, performing better or comparably to LOFO-LGBM while requiring fewer features (256 vs. 392), suggesting greater generalizability and efficiency. LOFO-XGB additionally exhibited more consistent performance among its top models and across varied weighting schemes. LOFO-LGBM showed slightly superior performance with GOF AP and recall, but LOFO-XGB excelled in all other areas. LOFO-XGB also outperformed ENS-XGB and ENS-LGBM. Although ENS-XGB yielded comparable performance to LOFO-XGB, it required nearly twice as many features (496) to achieve this. Similarly, LOFO-LGBM outperformed ENS-LGBM using fewer features (392 and 414, respectively).

These results demonstrate the overall superiority of the LOFO feature importance ranking algorithm over the ENS method. They also show that the XGB classifier offers superior performance over LGBM regardless of ranking scheme, which was also evident when first assessing classifiers before tuning hyperparameters. Therefore, it can be concluded that LOFO-XGB is the superior feature importance ranking and classifier combination for use with LoGoFunc.

It is noteworthy that XGB was one of the classifiers assessed during the development of LoGoFunc, before the Itan group identified LGBM as the optimal classifier[15]. It is not clear why XGB consistently outperformed LGBM during this project, even before feature ranking or feature selection. One possible explanation could be differences in hyperparameter tuning, since, the developers used the Optuna hyperparameter tuning framework instead of

RandomizedSearchCV[88]. The differences in performance between XGB and LGBM became more apparent after feature ranking, and the LOFO ranking algorithm used XGB as its classifier, so it is possible that this introduced bias in favor of XGB performance.

While more work is required to optimize this ranking and feature selection pipeline, these results illustrate the critical role that feature importance rankings and feature selection algorithms play in developing models with favorable characteristics. While this LOFO ranking and feature subset evaluation process is time-consuming and computationally expensive, this report demonstrates its effectiveness. The computational resources used to run through this process could potentially be offset by the increased efficiency of ML models that require fewer features to deliver similar or superior performance outcomes.

Future Work

When assessing classifiers, XGB was identified as superior based on its overall performance metrics. It may be worthwhile to revisit this comparison and use class-wise metrics instead, possibly focusing on GOF since it tends to underperform. This may be a more granular and fruitful approach for assessing classifier performance in the context of GOF/LOF classification.

Although LOFO appears to offer better ranking scores, it is much more computationally expensive than ENS, especially for high-dimensionality datasets such as those used by LoGoFunc. Based on the results observed in the ENS violin plot, four of the RCECV algorithms offered the most uniform distribution of mean ranking scores. It would be beneficial to run the ENS ranker again, using an ensemble of these four algorithms as this may yield more accurate rankings while consuming fewer computational resources. Additionally, the LOFO algorithm could be run again using LGBM, with its rankings compared with the XGB rankings. Then it would be possible to observe how the subsequent ranked subsets affect the results of the final feature selection metrics. Additionally, instead of constructing a weighting scheme to identify the best model across all metrics, it would be optimal to focus solely on GOF, since it consistently underperforms compared to LOF or Neutral even in the best models.

Due to time constraints, it was not possible to begin modifying LoGoFunc so that it can make predictions for somatic instead of germline mutations. However, the identification of a promising alternative classifier, and pathway for reducing feature count via LOFO ranking, will confer reduced reduce dimensionality for the much smaller somatic mutation dataset. This can potentially improve model performance, reduce noise, increase computational efficiency, and decrease the risk of overfitting.

Given that GOF variants are poorly understood, and somatic pathogenic GOF/LOF variants are known to promote cancerogenesis, further refinement of LoGoFunc in this area could contribute to developing diagnostic tools and target therapies. Furthermore, GOF/LOF variants have been shown to be involved in the aetiologies of other diseases, such as CNS ion channel dysfunction, so refinement of this tool could have research and clinical applications beyond cancer.

References

(1) Ostroverkhova, D.; Przytycka, T. M.; Panchenko, A. R. Cancer Driver Mutations: Predictions and Reality. *Trends Mol. Med.* **2023**, *29* (7), 554–566. https://doi.org/10.1016/j.molmed.2023.03.007.

(2) Li, Y.; Zhang, Y.; Li, X.; Yi, S.; Xu, J. Gain-of-Function Mutations: An Emerging Advantage for Cancer Biology. *Trends Biochem. Sci.* **2019**, *44* (8), 659–674. https://doi.org/10.1016/j.tibs.2019.03.009.

(3) Hack, J. B.; Horning, K.; Juroske Short, D. M.; Schreiber, J. M.; Watkins, J. C.; Hammer, M. F. Distinguishing Loss-of-Function and Gain-of-Function *SCN8A* Variants Using a Random Forest Classification Model Trained on Clinical Features. *Neurol. Genet.* **2023**, *9* (3), e200060. https://doi.org/10.1212/NXG.0000000000200060.

(4) Myers, S. J.; Yuan, H.; Perszyk, R. E.; Zhang, J.; Kim, S.; Nocilla, K. A.; Allen, J. P.; Bain, J. M.; Lemke, J. R.; Lal, D.; Benke, T. A.; Traynelis, S. F. Classification of Missense Variants in the N-Methyl-d-Aspartate Receptor GRIN Gene Family as Gain- or Loss-of-Function. *Hum. Mol. Genet.* **2023**, *32* (19), 2857–2871. https://doi.org/10.1093/hmg/ddad104.

(5) Cerami, E.; Gao, J.; Dogrusoz, U.; Gross, B. E.; Sumer, S. O.; Aksoy, B. A.; Jacobsen, A.; Byrne, C. J.; Heuer, M. L.; Larsson, E.; Antipin, Y.; Reva, B.; Goldberg, A. P.; Sander, C.; Schultz, N. The cBio Cancer Genomics Portal: An Open Platform for Exploring Multidimensional Cancer Genomics Data. *Cancer Discov.* **2012**, *2* (5), 401–404. https://doi.org/10.1158/2159-8290.CD-12-0095.

(6) Tate, J. G.; Bamford, S.; Jubb, H. C.; Sondka, Z.; Beare, D. M.; Bindal, N.; Boutselakis, H.; Cole, C. G.; Creatore, C.; Dawson, E.; Fish, P.; Harsha, B.; Hathaway, C.; Jupe, S. C.; Kok, C. Y.; Noble, K.; Ponting, L.; Ramshaw, C. C.; Rye, C. E.; Speedy, H. E.; Stefancsik, R.; Thompson, S. L.; Wang, S.; Ward, S.; Campbell, P. J.; Forbes, S. A. COSMIC: The Catalogue Of Somatic Mutations In Cancer. *Nucleic Acids Res.* **2019**, *47* (D1), D941–D947. https://doi.org/10.1093/nar/gky1015.

(7) Chakravarty, D.; Gao, J.; Phillips, S.; Kundra, R.; Zhang, H.; Wang, J.; Rudolph, J. E.; Yaeger, R.; Soumerai, T.; Nissan, M. H.; Chang, M. T.; Chandarlapaty, S.; Traina, T. A.; Paik, P. K.; Ho, A. L.; Hantash, F. M.; Grupe, A.; Baxi, S. S.; Callahan, M. K.; Snyder, A.; Chi, P.; Danila, D. C.; Gounder, M.; Harding, J. J.; Hellmann, M. D.; Iyer, G.; Janjigian, Y. Y.; Kaley, T.; Levine, D. A.; Lowery, M.; Omuro, A.; Postow, M. A.; Rathkopf, D.; Shoushtari, A. N.; Shukla, N.; Voss, M. H.; Paraiso, E.; Zehir, A.; Berger, M. F.; Taylor, B. S.; Saltz, L. B.; Riely, G. J.; Ladanyi, M.; Hyman, D. M.; Baselga, J.; Sabbatini, P.; Solit, D. B.; Schultz, N. OncoKB: A Precision Oncology Knowledge Base. *JCO Precis. Oncol.* **2017**, *1*, PO.17.00011. https://doi.org/10.1200/PO.17.00011.

(8) Yang, W.; Soares, J.; Greninger, P.; Edelman, E. J.; Lightfoot, H.; Forbes, S.; Bindal, N.; Beare, D.; Smith, J. A.; Thompson, I. R.; Ramaswamy, S.; Futreal, P. A.; Haber, D. A.; Stratton, M. R.; Benes, C.; McDermott, U.; Garnett, M. J. Genomics of Drug Sensitivity in Cancer (GDSC): A Resource for Therapeutic Biomarker Discovery in Cancer Cells. *Nucleic Acids Res.* **2013**, *41* (D1), D955–D961. https://doi.org/10.1093/nar/gks1111.

(9) Kourou, K.; Exarchos, T. P.; Exarchos, K. P.; Karamouzis, M. V.; Fotiadis, D. I. Machine Learning Applications in Cancer Prognosis and Prediction. *Comput. Struct. Biotechnol. J.* **2015**, *13*, 8–17. https://doi.org/10.1016/j.csbj.2014.11.005.

(10) Richardson, C. J.; Gao, Q.; Mitsopoulous, C.; Zvelebil, M.; Pearl, L. H.; Pearl, F. M. G. MoKCa Database—Mutations of Kinases in Cancer. *Nucleic Acids Res.* **2009**, *37* (suppl_1), D824–D831. https://doi.org/10.1093/nar/gkn832.

(11) Johannesen, K. M.; Liu, Y.; Koko, M.; Gjerulfsen, C. E.; Sonnenberg, L.; Schubert, J.; Fenger, C. D.; Eltokhi, A.; Rannap, M.; Koch, N. A.; Lauxmann, S.; Krüger, J.; Kegele, J.; Canafoglia, L.; Franceschetti, S.; Mayer, T.; Rebstock, J.; Zacher, P.; Ruf, S.; Alber, M.; Sterbova, K.; Lassuthová, P.; Vlckova, M.; Lemke, J. R.; Platzer, K.; Krey, I.; Heine, C.; Wieczorek, D.; Kroell-Seger, J.; Lund, C.; Klein, K. M.; Au, P. Y. B.; Rho, J. M.; Ho, A. W.; Masnada, S.; Veggiotti, P.; Giordano, L.; Accorsi, P.; Hoei-Hansen, C. E.; Striano, P.; Zara, F.; Verhelst, H.; Verhoeven, J. S.; Braakman, H. M. H.; van der Zwaag, B.; Harder, A. V. E.; Brilstra, E.; Pendziwiat, M.; Lebon, S.; Vaccarezza, M.; Le, N. M.; Christensen, J.; Grønborg, S.; Scherer, S. W.; Howe, J.; Fazeli, W.; Howell, K. B.; Leventer, R.; Stutterd, C.; Walsh, S.; Gerard, M.; Gerard, B.; Matricardi, S.; Bonardi, C. M.; Sartori, S.; Berger, A.; Hoffman-Zacharska, D.; Mastrangelo, M.; Darra, F.; Vøllo, A.; Motazacker, M. M.; Lakeman, P.; Nizon, M.; Betzler, C.; Altuzarra, C.; Caume, R.; Roubertie, A.; Gélisse, P.; Marini, C.; Guerrini, R.; Bilan, F.; Tibussek, D.; Koch-Hogrebe, M.; Perry, M. S.; Ichikawa, S.; Dadali, E.; Sharkov, A.; Mishina, I.; Abramov, M.; Kanivets, I.; Korostelev, S.; Kutsev, S.; Wain, K. E.; Eisenhauer, N.; Wagner, M.; Savatt, J. M.; Müller-Schlüter, K.; Bassan, H.; Borovikov, A.; Nassogne, M. C.; Destrée, A.; Schoonjans, A. S.; Meuwissen, M.; Buzatu, M.; Jansen, A.; Scalais, E.; Srivastava, S.; Tan, W. H.; Olson, H. E.; Loddenkemper, T.; Poduri, A.; Helbig, K. L.; Helbig, I.; Fitzgerald, M. P.; Goldberg, E. M.; Roser, T.; Borggraefe, I.; Brünger, T.; May, P.; Lal, D.; Lederer, D.; Rubboli, G.; Heyne, H. O.; Lesca, G.; Hedrich, U. B. S.; Benda, J.; Gardella, E.; Lerche, H.; Møller, R. S. Genotype-Phenotype Correlations in SCN8A-Related Disorders Reveal Prognostic and Therapeutic Implications. *Brain J. Neurol.* **2022**, *145* (9), 2991–3009. https://doi.org/10.1093/brain/awab321.

(12) Zhang, H.; Xu, M. S.; Fan, X.; Chung, W. K.; Shen, Y. Predicting Functional Effect of Missense Variants Using Graph Attention Neural Networks. *Nat. Mach. Intell.* **2022**, *4* (11), 1017–1028. https://doi.org/10.1038/s42256-022-00561-w.

(13) Kellerman, R.; Nayshool, O.; Barel, O.; Paz, S.; Amariglio, N.; Klang, E.; Rechavi, G. Mutation Pathogenicity Prediction by a Biology Based Explainable AI Multi-Modal Algorithm. June 5, 2024. https://doi.org/10.1101/2024.06.05.24308476.

(14) Sevim Bayrak, C.; Stein, D.; Jain, A.; Chaudhary, K.; Nadkarni, G. N.; Van Vleck, T. T.; Puel, A.; Boisson-Dupuis, S.; Okada, S.; Stenson, P. D.; Cooper, D. N.; Schlessinger, A.; Itan, Y. Identification of Discriminative Gene-Level and Protein-Level Features Associated with Pathogenic Gain-of-Function and Loss-of-Function Variants. *Am. J. Hum. Genet.* **2021**, *108* (12), 2301–2318. https://doi.org/10.1016/j.ajhg.2021.10.007.

(15) Stein, D.; Kars, M. E.; Wu, Y.; Bayrak, Ç. S.; Stenson, P. D.; Cooper, D. N.; Schlessinger, A.; Itan, Y. Genome-Wide Prediction of Pathogenic Gain- and Loss-of-Function Variants from Ensemble Learning of a Diverse Feature Set. *Genome Med.* **2023**, *15* (1), 103. https://doi.org/10.1186/s13073-023-01261-9.

(16) Stenson, P. D.; Mort, M.; Ball, E. V.; Evans, K.; Hayden, M.; Heywood, S.; Hussain, M.; Phillips, A. D.; Cooper, D. N. The Human Gene Mutation Database: Towards a Comprehensive Repository of Inherited Mutation Data for Medical Research, Genetic Diagnosis and next-Generation Sequencing Studies. *Hum. Genet.* **2017**, *136* (6), 665–677. https://doi.org/10.1007/s00439-017-1779-6.

(17) Landrum, M. J.; Lee, J. M.; Benson, M.; Brown, G. R.; Chao, C.; Chitipiralla, S.; Gu, B.; Hart, J.; Hoffman, D.; Jang, W.; Karapetyan, K.; Katz, K.; Liu, C.; Maddipatla, Z.; Malheiro, A.; McDaniel, K.; Ovetsky, M.; Riley, G.; Zhou, G.; Holmes, J. B.; Kattman, B. L.; Maglott, D. R. ClinVar: Improving Access to Variant Interpretations and Supporting Evidence. *Nucleic Acids Res.* **2018**, *46* (D1), D1062–D1067. https://doi.org/10.1093/nar/gkx1153.

(18) Karczewski, K. J.; Francioli, L. C.; Tiao, G.; Cummings, B. B.; Alföldi, J.; Wang, Q.; Collins, R. L.; Laricchia, K. M.; Ganna, A.; Birnbaum, D. P.; Gauthier, L. D.; Brand, H.; Solomonson, M.; Watts, N. A.; Rhodes, D.; Singer-Berk, M.; England, E. M.; Seaby, E. G.; Kosmicki, J. A.; Walters, R. K.; Tashman, K.; Farjoun, Y.; Banks, E.; Poterba, T.; Wang, A.; Seed, C.; Whiffin, N.; Chong, J. X.; Samocha, K. E.; Pierce-Hoffman, E.; Zappala, Z.; O'Donnell-Luria, A. H.; Minikel, E. V.; Weisburd, B.; Lek, M.; Ware, J. S.; Vittal, C.; Armean, I. M.; Bergelson, L.; Cibulskis, K.; Connolly, K. M.; Covarrubias, M.; Donnelly, S.; Ferriera, S.; Gabriel, S.; Gentry, J.; Gupta, N.; Jeandet, T.; Kaplan, D.; Llanwarne, C.; Munshi, R.; Novod, S.; Petrillo, N.; Roazen, D.; Ruano-Rubio, V.; Saltzman, A.; Schleicher, M.; Soto, J.; Tibbetts, K.; Tolonen, C.; Wade, G.; Talkowski, M. E.; Neale, B. M.; Daly, M. J.; MacArthur, D. G. The Mutational Constraint Spectrum Quantified from Variation in 141,456 Humans. *Nature* **2020**, *581* (7809), 434–443. https://doi.org/10.1038/s41586-020-2308-7.

(19) McLaren, W.; Gil, L.; Hunt, S. E.; Riat, H. S.; Ritchie, G. R. S.; Thormann, A.; Flicek, P.; Cunningham, F. The Ensembl Variant Effect Predictor. *Genome Biol.* **2016**, *17* (1), 122. https://doi.org/10.1186/s13059-016-0974-4.

(20) Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ronneberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapenko, A.; Bridgland, A.; Meyer, C.; Kohl, S. A. A.; Ballard, A. J.; Cowie, A.; Romera-Paredes, B.; Nikolov, S.; Jain, R.; Adler, J.; Back, T.; Petersen, S.; Reiman, D.; Clancy, E.; Zielinski, M.; Steinegger, M.; Pacholska, M.; Berghammer, T.; Bodenstein, S.; Silver, D.; Vinyals, O.; Senior, A. W.; Kavukcuoglu, K.; Kohli, P.; Hassabis, D. Highly Accurate Protein Structure Prediction with AlphaFold. *Nature* **2021**, *596* (7873), 583–589. https://doi.org/10.1038/s41586-021-03819-2.

(21) *itan-lab / LoGoFunc · GitLab*. GitLab. https://gitlab.com/itan-lab/logofunc (accessed 2024-08-16).

(22) Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.-Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc., 2017; Vol. 30.

(23) Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; ACM: San Francisco California USA, 2016; pp 785–794. https://doi.org/10.1145/2939672.2939785.

(24) *Introduction to Boosted Trees — xgboost 2.1.1 documentation*. https://xgboost.readthedocs.io/en/stable/tutorials/model.html (accessed 2024-08-12).

(25) Friedman, J.; Hastie, T.; Tibshirani, R. Additive Logistic Regression: A Statistical View of Boosting (With Discussion and a Rejoinder by the Authors). *Ann. Stat.* **2000**, *28* (2), 337–407. https://doi.org/10.1214/aos/1016218223.
(26) Hancock, J. T.; Khoshgoftaar, T. M. CatBoost for Big Data: An Interdisciplinary Review. *J. Big Data* **2020**, *7* (1), 94. https://doi.org/10.1186/s40537-020-00369-8.
(27) Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A. V.; Gulin, A. CatBoost: Unbiased Boosting with Categorical Features. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc., 2018; Vol. 31.
(28) Friedman, J. H. Greedy Function Approximation: A Gradient Boosting Machine. *Ann. Stat.* **2001**, *29* (5), 1189–1232.
(29) Freund, Y.; Schapire, R. E. A Short Introduction to Boosting.
(30) Schapire, R. E.; Singer, Y. Improved Boosting Algorithms Using Confidence-Rated Predictions. *Mach. Learn.* **1999**, *37* (3), 297–336. https://doi.org/10.1023/A:1007614523901.
(31) *LinearSVC*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.svm.LinearSVC.html (accessed 2024-08-14).
(32) Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; Lin, C.-J. LIBLINEAR: A Library for Large Linear Classification. *J Mach Learn Res* **2008**, *9*, 1871–1874.
(33) *SGDClassifier*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.linear_model.SGDClassifier.html (accessed 2024-08-14).
(34) Wilbur, W. J.; Kim, W. Stochastic Gradient Descent and the Prediction of MeSH for PubMed Records. *AMIA. Annu. Symp. Proc.* **2014**, *2014*, 1198–1207.
(35) Geurts, P.; Ernst, D.; Wehenkel, L. Extremely Randomized Trees. *Mach. Learn.* **2006**, *63* (1), 3–42. https://doi.org/10.1007/s10994-006-6226-1.
(36) *ExtraTreesClassifier*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html (accessed 2024-08-14).
(37) Geurts, P.; Louppe, G. Learning to Rank with Extremely Randomized Trees. In *Proceedings of the Learning to Rank Challenge*; PMLR, 2011; pp 49–61.
(38) Schmidhuber, J. Deep Learning in Neural Networks: An Overview. *Neural Netw.* **2015**, *61*, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003.
(39) Lumacad, G.; Namoco, R. Multilayer Perceptron Neural Network Approach to Classifying Learning Modalities Under the New Normal.
(40) Zekić-Sušac, M.; Pfeifer, S.; Šarlija, N. A Comparison of Machine Learning Methods in a High-Dimensional Classification Problem. *Bus. Syst. Res. J.* **2014**, *5* (3), 82–96. https://doi.org/10.2478/bsrj-2014-0021.
(41) *StackingClassifier*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.ensemble.StackingClassifier.html (accessed 2024-09-02).
(42) Hashemi, S.; Nowzari Dalini, A.; Jalali, A.; Banaei-Moghaddam, A. M.; Razaghi-Moghadam, Z. Cancerouspdomains: Comprehensive Analysis of Cancer Type-Specific Recurrent Somatic Mutations in Proteins and Domains. *BMC Bioinformatics* **2017**, *18* (1), 370. https://doi.org/10.1186/s12859-017-1779-5.

(43) *OrdinalEncoder*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.preprocessing.OrdinalEncoder.html (accessed 2024-08-20).
(44) *SimpleImputer*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.impute.SimpleImputer.html (accessed 2024-08-20).
(45) *MinMaxScaler*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html (accessed 2024-08-20).
(46) Bergstra, J.; Bengio, Y. Random Search for Hyper-Parameter Optimization.
(47) *3.1. Cross-validation: evaluating estimator performance*. scikit-learn. https://scikit-learn/stable/modules/cross_validation.html (accessed 2024-08-19).
(48) *train_test_split*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.model_selection.train_test_split.html (accessed 2024-08-19).
(49) Pudjihartono, N.; Fadason, T.; Kempa-Liehr, A. W.; O'Sullivan, J. M. A Review of Feature Selection Methods for Machine Learning-Based Disease Risk Prediction. *Front. Bioinforma.* **2022**, *2*. https://doi.org/10.3389/fbinf.2022.927312.
(50) Theng, D.; Bhoyar, K. K. Feature Selection Techniques for Machine Learning: A Survey of More than Two Decades of Research. *Knowl. Inf. Syst.* **2024**, *66* (3), 1575–1637. https://doi.org/10.1007/s10115-023-02010-5.
(51) *f_regression*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.feature_selection.f_regression.html (accessed 2024-08-14).
(52) *r_regression*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.feature_selection.r_regression.html (accessed 2024-08-14).
(53) *LinearRegression*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.linear_model.LinearRegression.html (accessed 2024-08-14).
(54) James, G.; Witten, D.; Hastie, T.; Tibshirani, R.; Taylor, J. E. *An Introduction to Statistical Learning: With Applications in Python*; Springer texts in statistics; Springer: Cham, Switzerland, 2023.
(55) *Ridge*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.linear_model.Ridge.html (accessed 2024-08-14).
(56) *Lasso*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.linear_model.Lasso.html (accessed 2024-08-14).
(57) *mutual_info_classif*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html (accessed 2024-08-15).
(58) Kozachenko, L.; Leonenko, N. On Statistical Estimation of Entropy of a Random Vector. Problems In-Formation Transmission. *Probl. Inf. Transm.* **1987**, *23* (2), 9–16.

(59) Brownlee, J. *Information Gain and Mutual Information for Machine Learning*. MachineLearningMastery.com. https://machinelearningmastery.com/information-gain-and-mutual-information/ (accessed 2024-08-24).
(60) *RandomForestRegressor*. scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html (accessed 2024-05-24).
(61) *DecisionTreeRegressor*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html (accessed 2024-08-15).
(62) *RFECV*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.feature_selection.RFECV.html (accessed 2024-08-15).
(63) *ElasticNet*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.linear_model.ElasticNet.html (accessed 2024-08-15).
(64) *LogisticRegression*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (accessed 2024-08-15).
(65) *Logistic Regression*. http://faculty.cas.usf.edu/mbrannick/regression/Logistic.html (accessed 2024-08-24).
(66) *SVC*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.svm.SVC.html (accessed 2024-08-15).
(67) Chang, C.-C.; Lin, C.-J. LIBSVM: A Library for Support Vector Machines. *ACM Trans Intell Syst Technol* **2011**, *2* (3), 27:1-27:27. https://doi.org/10.1145/1961189.1961199.
(68) *1.4. Support Vector Machines*. scikit-learn. https://scikit-learn/stable/modules/svm.html (accessed 2024-08-24).
(69) *RandomForestClassifier*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (accessed 2024-08-15).
(70) *Binary — Category Encoders 2.6.3 documentation*. https://contrib.scikit-learn.org/category_encoders/binary.html (accessed 2024-08-20).
(71) *pandas.get_dummies — pandas 2.2.2 documentation*. https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html (accessed 2024-08-20).
(72) *StandardScaler*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.preprocessing.StandardScaler.html (accessed 2024-08-20).
(73) *LabelEncoder*. scikit-learn. https://scikit-learn/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html (accessed 2024-08-20).
(74) Liu, J.; Danait, N.; Hu, S.; Sengupta, S. A Leave-One-Feature-out Wrapper Method for Feature Selection in Data Classification. In *2013 6th International Conference on Biomedical Engineering and Informatics*; 2013; pp 656–660. https://doi.org/10.1109/BMEI.2013.6747021.

(75) *Micro and Macro Averaging — Python documentation*. https://sklearn-evaluation.ploomber.io/en/latest/classification/micro_macro.html (accessed 2024-09-01).
(76) Chicco, D.; Jurman, G. The Matthews Correlation Coefficient (MCC) Should Replace the ROC AUC as the Standard Metric for Assessing Binary Classification. *BioData Min.* **2023**, *16*, 4. https://doi.org/10.1186/s13040-023-00322-4.
(77) Fawcett, T. An Introduction to ROC Analysis. *Pattern Recognit. Lett.* **2006**, *27* (8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010.
(78) Wolters, W. T. M.; Mareschal, B. Novel Types of Sensitivity Analysis for Additive MCDM Methods. *Eur. J. Oper. Res.* **1995**, *81* (2), 281–290. https://doi.org/10.1016/0377-2217(93)E0343-V.
(79) *Calculated consequences*. https://www.ensembl.org/info/genome/variation/prediction/predicted_data.html (accessed 2024-09-01).
(80) Carter, H.; Douville, C.; Stenson, P. D.; Cooper, D. N.; Karchin, R. Identifying Mendelian Disease Genes with the Variant Effect Scoring Tool. *BMC Genomics* **2013**, *14* (Suppl 3), S3. https://doi.org/10.1186/1471-2164-14-S3-S3.
(81) Ramani, R.; Krumholz, K.; Huang, Y.-F.; Siepel, A. PhastWeb: A Web Interface for Evolutionary Conservation Scoring of Multiple Sequence Alignments Using phastCons and phyloP. *Bioinformatics* **2019**, *35* (13), 2320–2322. https://doi.org/10.1093/bioinformatics/bty966.
(82) Kao, L. S.; Green, C. E. Analysis of Variance: Is There a Difference in Means and What Does It Mean? *J. Surg. Res.* **2008**, *144* (1), 158–170. https://doi.org/10.1016/j.jss.2007.02.053.
(83) *seaborn.violinplot — seaborn 0.13.2 documentation*. https://seaborn.pydata.org/generated/seaborn.violinplot.html (accessed 2024-09-01).
(84) Dosztányi, Z.; Mészáros, B.; Simon, I. ANCHOR: Web Server for Predicting Protein Binding Regions in Disordered Proteins. *Bioinformatics* **2009**, *25* (20), 2745–2746. https://doi.org/10.1093/bioinformatics/btp518.
(85) Cheng, J.; Nguyen, T. Y. D.; Cygan, K. J.; Çelik, M. H.; Fairbrother, W. G.; Avsec, žiga; Gagneur, J. MMSplice: Modular Modeling Improves the Predictions of Genetic Variant Effects on Splicing. *Genome Biol.* **2019**, *20* (1), 48. https://doi.org/10.1186/s13059-019-1653-z.
(86) Zhang, Z.; van Dijk, F.; de Klein, N.; van Gijn, M. E.; Franke, L. H.; Sinke, R. J.; Swertz, M. A.; van der Velde, K. J. Feasibility of Predicting Allele Specific Expression from DNA Sequencing Using Machine Learning. *Sci. Rep.* **2021**, *11* (1), 10606. https://doi.org/10.1038/s41598-021-89904-y.
(87) *IUPred2A: context-dependent prediction of protein disorder as a function of redox state and protein binding | Nucleic Acids Research | Oxford Academic*. https://academic.oup.com/nar/article/46/W1/W329/5026265 (accessed 2024-09-01).
(88) Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A Next-Generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*; KDD '19; Association for Computing Machinery: New York, NY, USA, 2019; pp 2623–2631. https://doi.org/10.1145/3292500.3330701.

Appendices

Contents:

Appendix:	**Description**
A	Full list of the 500 original LoGoFunc features, including descriptions.
B	Hyperparameter tuning --- search spaces and the chosen parameter sets.
C	ENS Feature importance rankings
D	LOFO Feature importance rankings
E	Results from the weighting schemes used for final feature selections.
F	Directory structures and notes about reproducing my results in Jupyter.

APPENDIX A — Feature Descriptions

ID	Feature Name	Description
1	IMPACT	The VEP impact modifier for the consequence type.
2	BIOTYPE	Biotype of transcript or regulatory feature.
3	CADD_raw	The raw CADD score.
4	BLOSUM62	Amino acid substitution score from the BLOSUM 62 matrix.
5	Conservation	GERP scores from the EPO 35 way mammalian alignment.
6	ProteinLengthChange	Change in protein length as a result of variant.
7	MaxEntScan_alt	Splice affect score for the alternate AA.
8	MaxEntScan_diff	Difference of alt and ref splicing scores.
9	MaxEntScan_ref	Splice affect score for the reference AA.
10	TSSDistance	The distance from the transcription start site for upstream variants
11	ada_score	dbscSNV adaboost meta-predictor for spice-altering variants.
12	rf_score	dbscSNV randomforest meta-predictor for spice-altering variants.
13	1000Gp3_AF	1000 genomes project allele frequency – all.
14	FATHMM_score	FATHMM score.
15	GERPplus_plus_NR	GERP++ neutral rate.
16	GERPplus_plus_RS	GERP++ RS value.
17	GM12878_fitCons_score	fitCons score based on cell type GM12878.
18	GenoCanyon_score	Prediction of the functional potential at each nucleotide
19	H1_hESC_fitCons_score	fitCons score based on cell type H1_hESC.
20	HUVEC_fitCons_score	fitCons score based on cell type HUVEC.
21	LINSIGHT	LINSIGHT score.
22	LIST_S2_score	LIST-S2 score.
23	LRT_score	LRT score.
24	M_CAP_score	M-CAP score.
25	MPC_score	MPC score.
26	MVP_score	MVP score.
27	MutationAssessor_score	Mutation Assessor score.
28	MutationTaster_score	Mutation Taster score.
29	PROVEAN_score	PROVEAN score.
30	SiPhy_29way_logOdds	SiPhy score based on 29 mammals genomes.
31	UK10K_AF	UK10K allele frequency.
32	VEST4_score	VEST4 score.
33	fathmm_MKL_coding_score	FATHMM-MKL score.
34	fathmm_XF_coding_score	FATHMM-XF score.
35	gnomAD_exomes_AF	Genome Aggregation Database exomes allele frequency.
36	gnomAD_genomes_AF	Genome Aggregation Database genomes allele frequency.
37	integrated_fitCons_score	fitCons score based on three cell types.
38	phastCons100way_vertebrate	Conservation score based on multiple alignments of 100 vertebrate genomes including human.
39	phastCons17way_primate	Conservation score based on multiple alignments of 17 primate genomes including human.
40	phastCons30way_mammalian	Conservation score based on multiple alignments of 30 mammalian genomes including human.
41	phyloP100way_vertebrate	Conservation score based on multiple alignments of 100 vertebrate genomes including human.
42	phyloP17way_primate	Conservation score based on multiple alignments of 17 primate genomes including human.
43	phyloP30way_mammalian	Conservation score based on multiple alignments of 30 mammalian genomes including human.
44	NMD	Prediction indicating stop_gained variant allowing for transcript escape of nonsense-mediated mRNA decay
45	GDI	The accumulated mutational damage of each human gene in healthy human population.
46	MSC_95CI	The lower boundary of the 95% confidence interval generated by CADD for a gene.
47	rel_cDNA_pos	cDNA position / cDNA length * 10.
48	rel_CDS_pos	CDS position / CDS length * 10.
49	rel_prot_pos	Protein position / Protein length * 10.
50	Selective_pressure	Selective evolutionary pressure acting on a gene estimated by the McDonald–Kreitman neutrality index at the population level.
51	Clarks_distance	Estimate of the amino acid composition complexity of each protein.
52	CDS_len	Coding sequence length.
53	Number_of_paralogs	Number of paralogs for a gene.
54	denovo_Zscore	Estimate of the rate of de novo mutation per gene and whether a given gene contains more de novo mutations than expected by chance alone.
55	RVIS	Ranking of genes by whether they have more or less common functional genetic variation relative to the genome wide expectation.
56	Indispensability_score	Estimates gene essentiality on the basis of a gene’s network and evolutionary properties.
57	Protein_dom	Pfam or interpro domain.
58	RSA	Relative solvent accessibility.

59	ASA	Absolute solvent accessibility.
60	RSA_Zfit	Reliability of the RSA prediction.
61	before_RSA_3	Relative Solvent Accessibility of the third amino acid residue upstream from the mutation site.
62	before_RSA_8	Relative Solvent Accessibility of the eighth amino acid residue upstream from the mutation site.
63	before_RSA_15	Relative Solvent Accessibility of the fifteenth amino acid residue upstream from the mutation site.
64	after_RSA_3	Relative Solvent Accessibility of the third amino acid residue downstream from the mutation site.
65	after_RSA_8	Relative Solvent Accessibility of the eighth amino acid residue downstream from the mutation site.
66	after_RSA_15	Relative Solvent Accessibility of the fifteenth amino acid residue downstream from the mutation site.
67	before_ASA_3	Absolute Solvent Accessibility of the third amino acid residue upstream from the mutation site.
68	before_ASA_8	Absolute Solvent Accessibility of the eighth amino acid residue upstream from the mutation site.
69	before_ASA_15	Absolute Solvent Accessibility of the fifteenth amino acid residue upstream from the mutation site.
70	after_ASA_3	Absolute Solvent Accessibility of the third amino acid residue downstream from the mutation site.
71	after_ASA_8	Absolute Solvent Accessibility of the eighth amino acid residue downstream from the mutation site.
72	after_ASA_15	Absolute Solvent Accessibility of the fifteenth amino acid residue downstream from the mutation site.
73	Phosphorylation	PTM.
74	Acetylation	PTM.
75	Methylation	PTM.
76	Ubiquitination	PTM.
77	Glycosylation	PTM.
78	PTM	1 if residue has any PTM, else 0.
79	Consequence_first	First reported consequence from VEP.
80	Condel_score	Condel score.
81	SIFT_score	SIFT score.
82	NearestExonJB_distance	Distance to nearest exon junction boundary within 10,000 bps.
83	NearestExonJB_len	Length of the nearest exon within 10,000 bps.
84	AF_dssp_secondary_structure	Secondary structure characterization of AlphaFold models from DSSP.
85	AF_Relative_ASA	Relative solvent accessibility calculated with DSSP on AlphaFold models.
86	IUPRED2	Disordered protein region prediction.
87	ANCHOR2	Disordered binding region prediction.
88	before_IUPRED_3	Disordered protein region prediction at the third amino acid residue upstream from the mutation site.
89	before_IUPRED_8	Disordered protein region prediction at the eighth amino acid residue upstream from the mutation site.
90	before_IUPRED_15	Disordered protein region prediction at the fifteenth amino acid residue upstream from the mutation site.
91	after_IUPRED_3	Disordered protein region prediction at the third amino acid residue downstream from the mutation site.
92	after_IUPRED_8	Disordered protein region prediction at the eighth amino acid residue downstream from the mutation site.
93	after_IUPRED_15	Disordered protein region prediction at the fifteenth amino acid residue downstream from the mutation site.
94	before_ANCHOR_3	Disordered binding region prediction at the third amino acid residue upstream from the mutation site.
95	before_ANCHOR_8	Disordered binding region prediction at the eighth amino acid residue upstream from the mutation site.
96	before_ANCHOR_15	Disordered binding region prediction at the fifteenth amino acid residue upstream from the mutation site.
97	after_ANCHOR_3	Disordered binding region prediction at the third amino acid residue downstream from the mutation site.
98	after_ANCHOR_8	Disordered binding region prediction at the eighth amino acid residue downstream from the mutation site.
99	after_ANCHOR_15	Disordered binding region prediction at the fifteenth amino acid residue downstream from the mutation site.
100	A3D_SCORE	Aggrescan3D score calculated on AlphaFold models.
101	n_contacts	The number of residue contacts calculated on AlphaFold models.
102	distance_com	The residue distance from the protein center of mass calculated from AlphaFold models.
103	concavity_score	Predicted ligand-binding sites calculated on AlphaFold models.
104	S_DDG[SEQ]	DDGun delta-delta G predictions from sequence.
105	STABILITY[SEQ]	Categorical DDGun delta-delta g predictions from sequence.
106	S_DDG[3D]	DDGun delta-delta G predictions from AlphaFold models.
107	STABILITY[3D]	Categorical DDGun delta-delta g predictions from AlphaFold models.
108	hgmd_mutcount	The number of HGMD variants proximal to a given residue in 3D space.
109	gnomsingle_mutcount	The number of single occurrence gnomAD variants proximal to a given residue in 3D space.
110	gnom_mutcount	The number of gnomAD variants proximal to a given residue in 3D space.
111	AF_confidence	The AlphaFold pLDDT value for the residue.
112	isHomomultimer	Indicated proteins which form homomultimers.
113	num_interactions	The number of unique protein interactors for a protein.
114	ppi_combined_0	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
115	ppi_combined_1	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
116	ppi_combined_2	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
117	ppi_combined_3	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
118	ppi_combined_4	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
119	ppi_combined_5	One of 64 dimensions summarizing the STRING PPI processed with node2vec.

120	ppi_combined_6	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
121	ppi_combined_7	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
122	ppi_combined_8	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
123	ppi_combined_9	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
124	ppi_combined_10	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
125	ppi_combined_11	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
126	ppi_combined_12	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
127	ppi_combined_13	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
128	ppi_combined_14	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
129	ppi_combined_15	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
130	ppi_combined_16	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
131	ppi_combined_17	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
132	ppi_combined_18	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
133	ppi_combined_19	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
134	ppi_combined_20	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
135	ppi_combined_21	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
136	ppi_combined_22	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
137	ppi_combined_23	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
138	ppi_combined_24	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
139	ppi_combined_25	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
140	ppi_combined_26	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
141	ppi_combined_27	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
142	ppi_combined_28	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
143	ppi_combined_29	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
144	ppi_combined_30	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
145	ppi_combined_31	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
146	ppi_combined_32	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
147	ppi_combined_33	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
148	ppi_combined_34	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
149	ppi_combined_35	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
150	ppi_combined_36	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
151	ppi_combined_37	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
152	ppi_combined_38	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
153	ppi_combined_39	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
154	ppi_combined_40	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
155	ppi_combined_41	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
156	ppi_combined_42	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
157	ppi_combined_43	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
158	ppi_combined_44	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
159	ppi_combined_45	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
160	ppi_combined_46	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
161	ppi_combined_47	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
162	ppi_combined_48	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
163	ppi_combined_49	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
164	ppi_combined_50	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
165	ppi_combined_51	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
166	ppi_combined_52	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
167	ppi_combined_53	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
168	ppi_combined_54	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
169	ppi_combined_55	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
170	ppi_combined_56	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
171	ppi_combined_57	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
172	ppi_combined_58	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
173	ppi_combined_59	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
174	ppi_combined_60	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
175	ppi_combined_61	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
176	ppi_combined_62	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
177	ppi_combined_63	One of 64 dimensions summarizing the STRING PPI processed with node2vec.
178	s_het	Genome-wide distribution of selective effects for heterozygous PTVs.
179	DRNApredDNAscore_aa_window_3_prev	Mean prediction of DNA binding residues for 3 residue window before variant.
180	DRNApredDNAscore_aa_window_8_prev	Mean prediction of DNA binding residues for 8 residue window before variant.

181	DRNApredDNAscore_aa_window_15_prev	Mean prediction of DNA binding residues for 15 residue window before variant.
182	DRNApredDNAscore_aa_window_3_next	Mean prediction of DNA binding residues for 3 residue window after variant.
183	DRNApredDNAscore_aa_window_8_next	Mean prediction of DNA binding residues for 8 residue window after variant.
184	DRNApredDNAscore_aa_window_15_next	Mean prediction of DNA binding residues for 15 residue window after variant.
185	DRNApredDNAscore_aa	Prediction of DNA binding residues for the AA.
186	ASAquick_normscore_aa_window_3_prev	Mean prediction of normalized protein accessible surface area for 3 residue window before variant.
187	ASAquick_normscore_aa_window_8_prev	Mean prediction of normalized protein accessible surface area for 8 residue window before variant.
188	ASAquick_normscore_aa_window_15_prev	Mean prediction of normalized protein accessible surface area for 15 residue window before variant.
189	ASAquick_normscore_aa_window_3_next	Mean prediction of normalized protein accessible surface area for 3 residue window after variant.
190	ASAquick_normscore_aa_window_8_next	Mean prediction of normalized protein accessible surface area for 8 residue window after variant.
191	ASAquick_normscore_aa_window_15_next	Mean prediction of normalized protein accessible surface area for 15 residue window after variant.
192	ASAquick_normscore_aa	Prediction of normalized protein accessible surface area for the AA.
193	ASAquick_rawscore_aa_window_3_prev	Mean prediction of raw protein accessible surface area for 3 residue window before variant.
194	ASAquick_rawscore_aa_window_8_prev	Mean prediction of raw protein accessible surface area for 8 residue window before variant.
195	ASAquick_rawscore_aa_window_15_prev	Mean prediction of raw protein accessible surface area for 15 residue window before variant.
196	ASAquick_rawscore_aa_window_3_next	Mean prediction of raw protein accessible surface area for 3 residue window after variant.
197	ASAquick_rawscore_aa_window_8_next	Mean prediction of raw protein accessible surface area for 8 residue window after variant.
198	ASAquick_rawscore_aa_window_15_next	Mean prediction of raw protein accessible surface area for 15 residue window after variant.
199	ASAquick_rawscore_aa	Prediction of protein accessible surface area for the AA.
200	DFLpredScore_aa_window_3_prev	Mean prediction of disordered flexible linker for 3 residue window before variant.
201	DFLpredScore_aa_window_8_prev	Mean prediction of disordered flexible linker for 8 residue window before variant.
202	DFLpredScore_aa_window_15_prev	Mean prediction of disordered flexible linker for 15 residue window before variant.
203	DFLpredScore_aa_window_3_next	Mean prediction of disordered flexible linker for 3 residue window after variant.
204	DFLpredScore_aa_window_8_next	Mean prediction of disordered flexible linker for 8 residue window after variant.
205	DFLpredScore_aa_window_15_next	Mean prediction of disordered flexible linker for 15 residue window after variant.
206	DFLpredScore_aa	Prediction of disordered flexible linker residues for the AA.
207	DRNApredRNAscore_aa_window_3_prev	Mean prediction of RNA binding residues for 3 residue window before variant.
208	DRNApredRNAscore_aa_window_8_prev	Mean prediction of RNA binding residues for 8 residue window before variant.
209	DRNApredRNAscore_aa_window_15_prev	Mean prediction of RNA binding residues for 15 residue window before variant.
210	DRNApredRNAscore_aa_window_3_next	Mean prediction of RNA binding residues for 3 residue window after variant.
211	DRNApredRNAscore_aa_window_8_next	Mean prediction of RNA binding residues for 8 residue window after variant.
212	DRNApredRNAscore_aa_window_15_next	Mean prediction of RNA binding residues for 15 residue window after variant.
213	DRNApredRNAscore_aa	Prediction of RNA binding residues for the AA.
214	DisoDNAscore_aa_window_3_prev	Mean prediction of disordered DNA binding residues for 3 residue window before variant.
215	DisoDNAscore_aa_window_8_prev	Mean prediction of disordered DNA binding residues for 8 residue window before variant.
216	DisoDNAscore_aa_window_15_prev	Mean prediction of disordered DNA binding residues for 15 residue window before variant.
217	DisoDNAscore_aa_window_3_next	Mean prediction of disordered DNA binding residues for 3 residue window after variant.
218	DisoDNAscore_aa_window_8_next	Mean prediction of disordered DNA binding residues for 8 residue window after variant.
219	DisoDNAscore_aa_window_15_next	Mean prediction of disordered DNA binding residues for 15 residue window after variant.
220	DisoDNAscore_aa	Prediction of disordered DNA binding residues for the AA.
221	DisoPROscore_aa_window_3_prev	Mean prediction of disordered protein binding residues for 3 residue window before variant.
222	DisoPROscore_aa_window_8_prev	Mean prediction of disordered protein binding residues for 8 residue window before variant.
223	DisoPROscore_aa_window_15_prev	Mean prediction of disordered protein binding residues for 15 residue window before variant.
224	DisoPROscore_aa_window_3_next	Mean prediction of disordered protein binding residues for 3 residue window after variant.
225	DisoPROscore_aa_window_8_next	Mean prediction of disordered protein binding residues for 8 residue window after variant.
226	DisoPROscore_aa_window_15_next	Mean prediction of disordered protein binding residues for 15 residue window after variant.
227	DisoPROscore_aa	Prediction of disordered protein binding residues for the AA.
228	DisoRNAscore_aa_window_3_prev	Mean prediction of disordered RNA binding residues for 3 residue window before variant.
229	DisoRNAscore_aa_window_8_prev	Mean prediction of disordered RNA binding residues for 8 residue window before variant.
230	DisoRNAscore_aa_window_15_prev	Mean prediction of disordered RNA binding residues for 15 residue window before variant.
231	DisoRNAscore_aa_window_3_next	Mean prediction of disordered RNA binding residues for 3 residue window after variant.
232	DisoRNAscore_aa_window_8_next	Mean prediction of disordered RNA binding residues for 8 residue window after variant.
233	DisoRNAscore_aa_window_15_next	Mean prediction of disordered RNA binding residues for 15 residue window after variant.
234	DisoRNAscore_aa	Prediction of disordered RNA binding residues for the AA.
235	MMseq2_conservation_level_aa_window_3_prev	MMseq2 based conservation score for 3 residue window before variant.
236	MMseq2_conservation_level_aa_window_8_prev	MMseq2 based conservation score for 8 residue window before variant.
237	MMseq2_conservation_level_aa_window_15_prev	MMseq2 based conservation score for 15 residue window before variant.
238	MMseq2_conservation_level_aa_window_3_next	MMseq2 based conservation score for 3 residue window after variant.
239	MMseq2_conservation_level_aa_window_8_next	MMseq2 based conservation score for 8 residue window after variant.
240	MMseq2_conservation_level_aa_window_15_next	MMseq2 based conservation score for 15 residue window after variant.
241	MMseq2_conservation_level_aa	MMseq2 based conservation score binned to deciles for aa.

242	MMseq2_conservation_score_aa_window_3_prev	MMseq2 based conservation score for 3 residue window before variant.
243	MMseq2_conservation_score_aa_window_8_prev	MMseq2 based conservation score for 8 residue window before variant.
244	MMseq2_conservation_score_aa_window_15_prev	MMseq2 based conservation score for 15 residue window before variant.
245	MMseq2_conservation_score_aa_window_3_next	MMseq2 based conservation score for 3 residue window after variant.
246	MMseq2_conservation_score_aa_window_8_next	MMseq2 based conservation score for 8 residue window after variant.
247	MMseq2_conservation_score_aa_window_15_next	MMseq2 based conservation score for 15 residue window after variant.
248	MMseq2_conservation_score_aa	MMseq2 based conservation score for aa.
249	MoRFchibiScore_aa_window_3_prev	Mean prediction of MoRF regions for 3 residue window before variant.
250	MoRFchibiScore_aa_window_8_prev	Mean prediction of MoRF regions for 8 residue window before variant.
251	MoRFchibiScore_aa_window_15_prev	Mean prediction of MoRF regions for 15 residue window before variant.
252	MoRFchibiScore_aa_window_3_next	Mean prediction of MoRF regions for 3 residue window after variant.
253	MoRFchibiScore_aa_window_8_next	Mean prediction of MoRF regions for 8 residue window after variant.
254	MoRFchibiScore_aa_window_15_next	Mean prediction of MoRF regions for 15 residue window after variant.
255	MoRFchibiScore_aa	Prediction of MoRF regions for the AA.
256	PSIPRED_helix_aa_window_3_prev	Mean prediction of protein helices for 3 residue window before variant.
257	PSIPRED_helix_aa_window_8_prev	Mean prediction of protein helices for 8 residue window before variant.
258	PSIPRED_helix_aa_window_15_prev	Mean prediction of protein helices for 15 residue window before variant.
259	PSIPRED_helix_aa_window_3_next	Mean prediction of protein helices for 3 residue window after variant.
260	PSIPRED_helix_aa_window_8_next	Mean prediction of protein helices for 8 residue window after variant.
261	PSIPRED_helix_aa_window_15_next	Mean prediction of protein helices for 15 residue window after variant.
262	PSIPRED_helix_aa	Prediction of protein helices for the AA.
263	PSIPRED_strand_aa_window_3_prev	Mean prediction of protein strands for 3 residue window before variant.
264	PSIPRED_strand_aa_window_8_prev	Mean prediction of protein strands for 8 residue window before variant.
265	PSIPRED_strand_aa_window_15_prev	Mean prediction of protein strands for 15 residue window before variant.
266	PSIPRED_strand_aa_window_3_next	Mean prediction of protein strands for 3 residue window after variant.
267	PSIPRED_strand_aa_window_8_next	Mean prediction of protein strands for 8 residue window after variant.
268	PSIPRED_strand_aa_window_15_next	Mean prediction of protein strands for 15 residue window after variant.
269	PSIPRED_strand_aa	Prediction of protein strands for the AA.
270	SCRIBERscore_aa_window_3_prev	Mean prediction of protein binding residues for 3 residue window before variant.
271	SCRIBERscore_aa_window_8_prev	Mean prediction of protein binding residues for 8 residue window before variant.
272	SCRIBERscore_aa_window_15_prev	Mean prediction of protein binding residues for 15 residue window before variant.
273	SCRIBERscore_aa_window_3_next	Mean prediction of protein binding residues for 3 residue window after variant.
274	SCRIBERscore_aa_window_8_next	Mean prediction of protein binding residues for 8 residue window after variant.
275	SCRIBERscore_aa_window_15_next	Mean prediction of protein binding residues for 15 residue window after variant.
276	SCRIBERscore_aa	Prediction of protein binding residues for the AA.
277	SignalP_score_aa_window_3_prev	Mean prediction of signal peptides for 3 residue window before variant.
278	SignalP_score_aa_window_8_prev	Mean prediction of signal peptides for 8 residue window before variant.
279	SignalP_score_aa_window_15_prev	Mean prediction of signal peptides for 15 residue window before variant.
280	SignalP_score_aa_window_3_next	Mean prediction of signal peptides for 3 residue window after variant.
281	SignalP_score_aa_window_8_next	Mean prediction of signal peptides for 8 residue window after variant.
282	SignalP_score_aa_window_15_next	Mean prediction of signal peptides for 15 residue window after variant.
283	SignalP_score_aa	Prediction of signal peptides for the AA.
284	gtex_Adipose_-_Subcutaneous	GTEx tissue transcripts per million.
285	gtex_Adipose_-_Visceral_(Omentum)	GTEx tissue transcripts per million.
286	gtex_Adrenal_Gland	GTEx tissue transcripts per million.
287	gtex_Artery_-_Aorta	GTEx tissue transcripts per million.
288	gtex_Artery_-_Coronary	GTEx tissue transcripts per million.
289	gtex_Artery_-_Tibial	GTEx tissue transcripts per million.
290	gtex_Bladder	GTEx tissue transcripts per million.
291	gtex_Brain_-_Amygdala	GTEx tissue transcripts per million.
292	gtex_Brain_-_Anterior_cingulate_cortex_(BA24)	GTEx tissue transcripts per million.
293	gtex_Brain_-_Caudate_(basal_ganglia)	GTEx tissue transcripts per million.
294	gtex_Brain_-_Cerebellar_Hemisphere	GTEx tissue transcripts per million.
295	gtex_Brain_-_Cerebellum	GTEx tissue transcripts per million.
296	gtex_Brain_-_Cortex	GTEx tissue transcripts per million.
297	gtex_Brain_-_Frontal_Cortex_(BA9)	GTEx tissue transcripts per million.
298	gtex_Brain_-_Hippocampus	GTEx tissue transcripts per million.
299	gtex_Brain_-_Hypothalamus	GTEx tissue transcripts per million.
300	gtex_Brain_-_Nucleus_accumbens_(basal_ganglia)	GTEx tissue transcripts per million.
301	gtex_Brain_-_Putamen_(basal_ganglia)	GTEx tissue transcripts per million.
302	gtex_Brain_-_Spinal_cord_(cervical_c-1)	GTEx tissue transcripts per million.

303	gtex_Brain_-_Substantia_nigra	GTEx tissue transcripts per million.
304	gtex_Breast_-_Mammary_Tissue	GTEx tissue transcripts per million.
305	gtex_Cells_-_Cultured_fibroblasts	GTEx tissue transcripts per million.
306	gtex_Cells_-_EBV-transformed_lymphocytes	GTEx tissue transcripts per million.
307	gtex_Cervix_-_Ectocervix	GTEx tissue transcripts per million.
308	gtex_Cervix_-_Endocervix	GTEx tissue transcripts per million.
309	gtex_Colon_-_Sigmoid	GTEx tissue transcripts per million.
310	gtex_Colon_-_Transverse	GTEx tissue transcripts per million.
311	gtex_Esophagus_-_Gastroesophageal_Junction	GTEx tissue transcripts per million.
312	gtex_Esophagus_-_Mucosa	GTEx tissue transcripts per million.
313	gtex_Esophagus_-_Muscularis	GTEx tissue transcripts per million.
314	gtex_Fallopian_Tube	GTEx tissue transcripts per million.
315	gtex_Heart_-_Atrial_Appendage	GTEx tissue transcripts per million.
316	gtex_Heart_-_Left_Ventricle	GTEx tissue transcripts per million.
317	gtex_Kidney_-_Cortex	GTEx tissue transcripts per million.
318	gtex_Kidney_-_Medulla	GTEx tissue transcripts per million.
319	gtex_Liver	GTEx tissue transcripts per million.
320	gtex_Lung	GTEx tissue transcripts per million.
321	gtex_Minor_Salivary_Gland	GTEx tissue transcripts per million.
322	gtex_Muscle_-_Skeletal	GTEx tissue transcripts per million.
323	gtex_Nerve_-_Tibial	GTEx tissue transcripts per million.
324	gtex_Ovary	GTEx tissue transcripts per million.
325	gtex_Pancreas	GTEx tissue transcripts per million.
326	gtex_Pituitary	GTEx tissue transcripts per million.
327	gtex_Prostate	GTEx tissue transcripts per million.
328	gtex_Skin_-_Not_Sun_Exposed_(Suprapubic)	GTEx tissue transcripts per million.
329	gtex_Skin_-_Sun_Exposed_(Lower_leg)	GTEx tissue transcripts per million.
330	gtex_Small_Intestine_-_Terminal_Ileum	GTEx tissue transcripts per million.
331	gtex_Spleen	GTEx tissue transcripts per million.
332	gtex_Stomach	GTEx tissue transcripts per million.
333	gtex_Testis	GTEx tissue transcripts per million.
334	gtex_Thyroid	GTEx tissue transcripts per million.
335	gtex_Uterus	GTEx tissue transcripts per million.
336	gtex_Vagina	GTEx tissue transcripts per million.
337	gtex_Whole_Blood	GTEx tissue transcripts per million.
338	haplo	Predicted probability of gene haploinsufficiency.
339	haplo_imputed	Predicted probability of gene haploinsufficiency with multiple imputation for missing features.
340	PHOSPHORYLATION	PTM.
341	ACETYLATION	PTM.
342	UBIQUITINATION	PTM.
343	S-NITROSYLATION	PTM.
344	N-GLYCOSYLATION	PTM.
345	METHYLATION	PTM.
346	O-GLYCOSYLATION	PTM.
347	MYRISTOYLATION	PTM.
348	C-GLYCOSYLATION	PTM.
349	SUMOYLATION	PTM.
350	S-GLYCOSYLATION	PTM.
351	Dominant_probability	Prediction of variants pathogenic for autosomal dominant disease.
352	Recessive_probability	Prediction of variants pathogenic for autosomal recessive disease.
353	polyphen_dscore	Difference of PSIC scores for two amino acid residue variants (score1 − score2).
354	polyphen_score1	PSIC score for wild type AA residue.
355	polyphen_score2	PSIC score for mutant AA residue.
356	polyphen_nobs	Number of residues observed at the substitution position in multiple alignment (without gaps).
357	polyphen_normasa	Normalized accessible surface area.
358	polyphen_dvol	Change in residue side chain volume.
359	polyphen_dprop	Change in solvent accessible surface propensity resulting from the substitution.
360	polyphen_bfact	Normalized B-factor (temperature factor) for the residue.
361	polyphen_hbonds	The number of hydrogen sidechain-sidechain and sidechain-mainchain bonds formed by the residue .
362	polyphen_avenhet	The number of residue contacts with heteroatoms, average per homologous PDB chain
363	polyphen_mindhet	The closest residue contact with a heteroatom, Å.

364	polyphen_avenint	Number of residue contacts with other chains, average per homologous PDB chain.
365	polyphen_mindint	Closest residue contact with other chain, Å.
366	polyphen_avensit	The number of residue contacts with critical sites, average per homologous PDB chain.
367	polyphen_mindsit	The closest residue contact with a critical site, Å.
368	polyphen_idpmax	Maximum congruency of the mutant amino acid residue to all sequences in multiple alignment.
369	polyphen_idpsnp	Maximum congruency of the mutant amino acid residue to the sequences in multiple alignment with the mutant residue.
370	polyphen_idqmin	Query sequence identity with the closest homologue deviating from the wild type amino acid residue.
371	ConsScore	Score assigned to VEP consequence.
372	GC	Percent GC in a window of +/- 75bp.
373	CpG	Percent CpG in a window of +/-75bp.
374	motifECount	Total number of overlapping motifs
375	motifEHIPos	Is the position considered highly informative for an overlapping motif by VEP?
376	motifEScoreChng	VEP score change for the overlapping motif site.
377	Dst2Splice	Distance to splice site in 20bp; positive: exonic, negative: intronic.
378	Dst2SplType	Closest splice site is ACCEPTOR or DONOR.
379	minDistTSS	Distance to closest Transcribed Sequence Start (TSS).
380	minDistTSE	Distance to closest Transcribed Sequence End (TSE).
381	priPhCons	Primate PhastCons conservation score (excl. human).
382	mamPhCons	Mammalian PhastCons conservation score (excl. human).
383	verPhCons	Vertebrate PhastCons conservation score (excl. human).
384	priPhyloP	Primate PhyloP score (excl. human).
385	mamPhyloP	Mammalian PhyloP score (excl. human).
386	verPhyloP	Vertebrate PhyloP score (excl. human).
387	bStatistic_y	Background selection score.
388	targetScan	targetscan.
389	mirSVR-Score	mirSVR-Score.
390	mirSVR-E	mirSVR-E.
391	mirSVR-Aln	mirSVR-Aln.
392	cHmm_E1	Number of 48 cell types in chromHMM state E1.
393	cHmm_E2	Number of 48 cell types in chromHMM state E2.
394	cHmm_E3	Number of 48 cell types in chromHMM state E3.
395	cHmm_E4	Number of 48 cell types in chromHMM state E4.
396	cHmm_E5	Number of 48 cell types in chromHMM state E5.
397	cHmm_E6	Number of 48 cell types in chromHMM state E6.
398	cHmm_E7	Number of 48 cell types in chromHMM state E7.
399	cHmm_E8	Number of 48 cell types in chromHMM state E8.
400	cHmm_E9	Number of 48 cell types in chromHMM state E9.
401	cHmm_E10	Number of 48 cell types in chromHMM state E10.
402	cHmm_E11	Number of 48 cell types in chromHMM state E11.
403	cHmm_E12	Number of 48 cell types in chromHMM state E12.
404	cHmm_E13	Number of 48 cell types in chromHMM state E13.
405	cHmm_E14	Number of 48 cell types in chromHMM state E14.
406	cHmm_E15	Number of 48 cell types in chromHMM state E15.
407	cHmm_E16	Number of 48 cell types in chromHMM state E16.
408	cHmm_E17	Number of 48 cell types in chromHMM state E17.
409	cHmm_E18	Number of 48 cell types in chromHMM state E18.
410	cHmm_E19	Number of 48 cell types in chromHMM state E19.
411	cHmm_E20	Number of 48 cell types in chromHMM state E20.
412	cHmm_E21	Number of 48 cell types in chromHMM state E21.
413	cHmm_E22	Number of 48 cell types in chromHMM state E22.
414	cHmm_E23	Number of 48 cell types in chromHMM state E23.
415	cHmm_E24	Number of 48 cell types in chromHMM state E24.
416	cHmm_E25	Number of 48 cell types in chromHMM state E25.
417	GerpRS	GERP++ RS score.
418	GerpRSpval	GERP++ RS score p-value.
419	GerpN	GERP++ N score.
420	GerpS	GERP++ S score.
421	tOverlapMotifs	Number of overlapping predicted TF motifs.
422	motifDist	Reference minus alternate allele difference in nucleotide frequency
		within an predicted overlapping motif.
423	EncodeH3K4me1-sum	Sum of Encode H3K4me1 levels (from 13 cell lines).

424	EncodeH3K4me1-max	Maximum of Encode H3K4me1 levels (from 13 cell lines).
425	EncodeH3K4me2-sum	Sum of Encode H3K4me2 levels (from 14 cell lines).
426	EncodeH3K4me2-max	Maximum of Encode H3K4me2 levels (from 14 cell lines).
427	EncodeH3K4me3-sum	Sum of Encode H3K4me3 levels (from 14 cell lines).
428	EncodeH3K4me3-max	Maximum of Encode H3K4me3 levels (from 14 cell lines).
429	EncodeH3K9ac-sum	Sum of Encode H3K9ac levels (from 13 cell lines).
430	EncodeH3K9ac-max	Maximum of Encode H3K9ac levels (from 13 cell lines).
431	EncodeH3K9me3-sum	Sum of Encode H3K9me3 levels (from 14 cell lines).
432	EncodeH3K9me3-max	Maximum of Encode H3K9me3 levels (from 14 cell lines).
433	EncodeH3K27ac-sum	Sum of Encode H3K27ac levels (from 14 cell lines).
434	EncodeH3K27ac-max	Maximum of Encode H3K27ac levels (from 14 cell lines).
435	EncodeH3K27me3-sum	Sum of Encode H3K27me3 levels (from 14 cell lines).
436	EncodeH3K27me3-max	Maximum of Encode H3K27me3 levels (from 14 cell lines).
437	EncodeH3K36me3-sum	Sum of Encode H3K36me3 levels (from 10 cell lines).
438	EncodeH3K36me3-max	Maximum of Encode H3K36me3 levels (from 10 cell lines).
439	EncodeH3K79me2-sum	Sum of Encode H3K79me2 levels (from 13 cell lines).
440	EncodeH3K79me2-max	Maximum of Encode H3K79me2 levels (from 13 cell lines).
441	EncodeH4K20me1-sum	Sum of Encode H4K20me1 levels (from 11 cell lines).
442	EncodeH4K20me1-max	Maximum of Encode H4K20me1 levels (from 11 cell lines).
443	EncodeH2AFZ-sum	Sum of Encode H2AFZ levels (from 13 cell lines).
444	EncodeH2AFZ-max	Maximum of Encode H2AFZ levels (from 13 cell lines).
445	EncodeDNase-sum	Sum of Encode Dnase-seq levels (from 12 cell lines).
446	EncodeDNase-max	Maximum of Encode Dnase-seq levels (from 12 cell lines).
447	EncodetotalRNA-sum	Sum of Encode totalRNA-seq levels (from 10 cell lines always minus and plus strand).
448	EncodetotalRNA-max	Maximum of Encode totalRNA-seq levels (from 10 cell lines always minus and plus strand).
449	Grantham_x	The Grantham distance from reference to mutation amino acid residue.
450	SpliceAI-acc-gain	Masked SpliceAI acceptor gain score.
451	SpliceAI-acc-loss	Masked SpliceAI acceptor loss score.
452	SpliceAI-don-gain	Masked SpliceAI donor gain score.
453	SpliceAI-don-loss	Masked SpliceAI donor loss score.
454	MMSp_acceptorIntron	MMSplice acceptor intron (intron 3') score.
455	MMSp_acceptor	MMSplice acceptor score.
456	MMSp_exon	MMSplice exon score.
457	MMSp_donor	MMSplice donor score.
458	MMSp_donorIntron	MMSplice donor intron.
459	Dist2Mutation	Distance between the closest BRAVO SNV up and downstream (position itself excluded).
460	Freq100bp	Number of frequent (MAF > 0.05) BRAVO SNV in 100 bp window nearby.
461	Rare100bp	Number of rare (MAF < 0.05) BRAVO SNV in 100 bp window nearby.
462	Sngl100bp	Number of single occurrence BRAVO SNV in 100 bp window nearby.
463	Freq1000bp	Number of frequent (MAF > 0.05) BRAVO SNV in 1000 bp window nearby.
464	Rare1000bp	Number of rare (MAF < 0.05) BRAVO SNV in 1000 bp window nearby.
465	Sngl1000bp	Number of single occurence BRAVO SNV in 1000 bp window nearby.
466	Freq10000bp	Number of frequent (MAF > 0.05) BRAVO SNV in 10000 bp window nearby.
467	Rare10000bp	Number of rare (MAF < 0.05) BRAVO SNV in 10000 bp window nearby.
468	Sngl10000bp	Number of single occurence BRAVO SNV in 10000 bp window nearby.
469	EnsembleRegulatoryFeature	Matches in the Ensemble Regulatory Built (similar to annotype).
470	dbscSNV-ada_score	dbscSNV adaboost meta-predictor for spice-altering variants.
471	dbscSNV-rf_score	dbscSNV randomforest meta-predictor for spice-altering variants.
472	RemapOverlapTF	Remap number of different transcription factors binding.
473	RemapOverlapCL	Remap number of different transcription factor - cell line combinations binding.
474	Charge	The change in formal charge resulting from replacing the reference amino acid residue with the mutation.
475	Volume	The change in residue volume resulting from the replacement (in units of cubic Angstroms).
476	Hydrophobicity	The change in hydrophobicity resulting from the substitution.
477	Polarity	Polarity change from reference to mutation amino acid residue.
478	Ex	Amino acid substitution score from the EX matrix.
479	PAM250	Amino acid substitution score from the PAM250 matrix.
480	JM	Amino acid substitution score from the Miyazawa-Jernigan contact energy matrix.
481	HGMD2003	Number of times that the reference to mutation substitution occurs in the Human Gene Mutation Database, 2003 version.
482	VB	Amino acid substitution score from the VB matrix.
483	Transition	Frequency of transition between two neighboring amino acids based on all human proteins in SwissProt/TrEMBL.
484	COSMIC	Ln(frequency) of missense change type in COSMIC (release 38).

485	COSMICvsSWISSPROT	Ln(frequency) of missense change in COSMIC (release 38) normalized by the frequency of reference amino acid residue in human proteins in SwissProt/TrEMBL.
486	HAPMAP	Ln(frequency) of missense change type in HapMap validated SNPs in dbSNP Build 129.
487	COSMICvsHAPMAP	Ln(frequency) of missense change in COSMIC (release 38) normalized by the number of times the change type was observed in HapMap validated SNPs in dbSNP Build 129.
488	MOD_RES	Uniprot modified residue.
489	MOD_RES_DESCRIPTION	Uniprot modified residue description.
490	REGION	Uniprot region.
491	REGION_DESCRIPTION	Uniprot region description.
492	INTERACTION_REGION	Uniprot interaction region.
493	REQUIRED_FOR_INTER	Uniprot region required for interaction.
494	ATP_binding_gbind	GraphBind ligand score.
495	Ca2+_binding_gbind	GraphBind ligand score.
496	DNA_binding_gbind	GraphBind ligand score.
497	HEME_binding_gbind	GraphBind ligand score.
498	Mg2+_binding_gbind	GraphBind ligand score.
499	Mn2+_binding_gbind	GraphBind ligand score.
500	RNA_binding_gbind	GraphBind ligand score.

APPENDIX B -- Hyperparameter Search Spaces

XGB Search Space:

Parameter	Values
n_estimators	100
	200
	300
	400
max_depth	3
	4
	5
	6
	7
	8
learning_rate	np.logspace(-3, 0, 4)
subsample	0.5
	0.75
	1.0
colsample_bytree	0.5
	0.75
	1.0
min_child_weight	1
	2
	3
	4
	5
gamma	0
	0.1
	0.2
	0.3
	0.4

CTB Search Space:

Parameter	Values
iterations	50
	100
	200
	300
	400
learning_rate	np.logspace(-3, 0, 7)
depth	4
	5
	6
	7
	8
l2_leaf_reg	1
	3
	5
	7
	9
border_count	32
	64
	128
bagging_temperature	0
	0.5
	1
random_strength	1
	10
	100

ADB Search Space:

Parameter	Values
n_estimators	50
	100
	200
	300
	400
learning_rate	np.logspace(-3, 1, 7)
algorithm	SAMME
	SAMME.R
base_estimator__max_depth	1
	2
	3
	4
	5
base_estimator__min_samples_sp	2
lit	5
	10
base_estimator__min_samples_le	1
af	2
	4
base_estimator__criterion	gini
	entropy

MLP Search Space:

Parameter	Values
hidden_sizes	[512, 256, 128, 64]
	[256, 128, 64]
	[512, 256, 128]
dropout_rate	0.3
	0.4
	0.5
learning_rate	0.001
	0.0001
	1e-05
batch_size	32
	64
	128
epochs	500
	1000
	1500

SGD Search Space:

Parameter	Values

base_estimator__alpha	np.logspace(-4, 0, 5)
base_estimator__penalty	l2
	l1
	elasticnet
base_estimator__l1_ratio	np.linspace(0, 1, 5)
base_estimator__class_weight	None
	balanced
base_estimator__learning_rate	optimal
	invscaling
	adaptive
base_estimator__eta0	0.001
	0.01
	0.1
base_estimator__power_t	0.25
	0.5
	0.75

ETC Search Space:

Parameter	Values
n_estimators	50
	100
	200
	300
	400
max_depth	None
	10
	20
	30
	40
	50
min_samples_split	2
	5
	10
min_samples_leaf	1
	2
	4
max_features	sqrt
	log2
bootstrap	True
	False
criterion	gini
	entropy

LSVC Search Space:

Parameter	Values
C	uniform(0.1, 1000)
class_weight	None
	balanced
	{'0': 1, '1': 2}
	{'0': 2, '1': 1}
max_iter	1000
	2000
	5000
	10000
tol	loguniform(1e-6, 1e-2)

STACK Search Space:

Parameter	Values
rf__n_estimators	50
	100
	200
rf__max_depth	None
	10
	20
et__n_estimators	50
	100
	200
et__max_depth	None
	10
	20
dt__max_depth	None
	10
	20
final_estimator__C	np.logspace(-2, 2, 5)

APPENDIX B -- Hyperparameter sets used for training model ensembles

ADB Model Parameters:

Set	n_estimators	learning_rate	base_estimator__min_samples_split	base_estimator__min_samples_leaf	base_estimator__max_depth	base_estimator__criterion	algorithm
1	200	0.1	2	4	4	entropy	SAMME.R
2	200	0.46415888336127775	10	1	5	gini	SAMME.R
3	400	0.46415888336127775	2	4	4	entropy	SAMME.R
4	300	0.46415888336127775	2	2	5	entropy	SAMME
5	300	0.46415888336127775	5	1	5	entropy	SAMME
6	200	0.46415888336127775	10	4	5	gini	SAMME
7	400	0.46415888336127775	10	4	5	gini	SAMME.R
8	100	0.1	2	2	5	entropy	SAMME.R
9	400	0.46415888336127775	2	1	4	entropy	SAMME.R
10	100	0.1	10	1	5	entropy	SAMME.R
11	400	0.021544346900318832	10	4	5	gini	SAMME.R
12	100	0.1	5	4	5	entropy	SAMME.R
13	300	0.1	10	1	4	gini	SAMME.R
14	400	0.46415888336127775	10	1	4	entropy	SAMME.R
15	300	0.021544346900318832	5	2	5	entropy	SAMME.R
16	400	0.1	5	4	4	entropy	SAMME.R
17	400	0.46415888336127775	5	2	5	gini	SAMME.R
18	400	0.46415888336127775	2	2	5	entropy	SAMME.R
19	400	0.46415888336127775	10	4	5	entropy	SAMME
20	400	0.021544346900318832	10	2	5	entropy	SAMME.R
21	200	0.1	2	2	5	gini	SAMME.R
22	300	0.46415888336127775	2	4	5	entropy	SAMME.R
23	400	0.46415888336127775	5	4	5	entropy	SAMME.R
24	300	0.1	2	1	5	entropy	SAMME.R
25	200	0.1	10	4	5	entropy	SAMME.R
26	300	0.1	2	2	5	entropy	SAMME.R
27	300	0.1	5	1	5	gini	SAMME.R

CTB Model Parameters:

Set	random_strength	learning_rate	l2_leaf_reg	iterations	depth	border_count	bagging_temperature
1	100	1.0	3	400	8	32	0.5
2	1	0.31622776601683794	5	400	6	64	0.5
3	1	0.31622776601683794	9	400	6	64	0
4	1	1.0	3	400	7	32	0
5	1	0.31622776601683794	1	200	7	128	0.5
6	10	1.0	3	400	6	32	0
7	1	0.31622776601683794	7	300	7	32	0
8	1	0.31622776601683794	1	400	5	64	0
9	100	1.0	3	400	8	32	0
10	1	0.31622776601683794	5	400	6	32	0
11	10	0.31622776601683794	3	200	8	64	0
12	10	0.31622776601683794	7	400	8	64	1
13	1	0.31622776601683794	5	300	7	32	0.5
14	100	0.31622776601683794	1	400	6	128	0.5
15	10	0.31622776601683794	5	400	6	128	0
16	100	0.31622776601683794	1	200	8	64	0.5
17	10	0.1	1	400	8	128	0.5
18	10	0.31622776601683794	1	300	7	128	0.5
19	100	0.31622776601683794	5	300	8	64	0
20	1	0.31622776601683794	1	300	7	128	0.5
21	1	0.31622776601683794	5	400	7	128	0
22	100	0.31622776601683794	1	300	8	32	1
23	1	0.31622776601683794	1	300	8	128	0.5
24	10	0.1	1	400	8	32	0
25	100	0.31622776601683794	1	300	8	128	0
26	100	0.31622776601683794	1	300	8	128	0.5
27	100	0.31622776601683794	3	400	8	128	0.5

ETC Model Parameters:

Set	n_estimators	min_samples_split	min_samples_leaf	max_features	max_depth	criterion	bootstrap
1	200	5	1	log2	None	entropy	False
2	50	2	2	sqrt	40	gini	False
3	400	2	1	sqrt	20	gini	False
4	200	5	1	log2	40	entropy	False
5	50	2	1	sqrt	30	entropy	False
6	300	10	2	sqrt	30	entropy	False
7	300	5	1	sqrt	50	entropy	True
8	100	2	1	sqrt	20	entropy	False
9	300	5	1	log2	40	entropy	False
10	400	2	1	sqrt	None	entropy	True
11	100	5	2	sqrt	None	entropy	False

12	400	5	1	sqrt	30	entropy	True
13	100	2	1	sqrt	30	gini	False
14	200	10	1	sqrt	30	gini	False
15	200	5	2	sqrt	40	entropy	False
16	200	2	2	sqrt	50	entropy	False
17	400	2	2	sqrt	40	gini	False
18	400	10	1	sqrt	None	gini	False
19	300	5	2	sqrt	50	entropy	False
20	200	10	1	sqrt	50	entropy	False
21	400	10	1	sqrt	30	entropy	False
22	300	2	1	sqrt	None	entropy	False
23	300	2	1	sqrt	50	entropy	False
24	100	5	1	sqrt	40	entropy	False
25	200	5	1	sqrt	50	gini	False
26	200	2	1	sqrt	30	entropy	False
27	400	5	1	sqrt	50	entropy	False

LSVC Model Parameters:

Set	C	class_weight	dual	loss	max_iter	penalty	tol
1	356.85332669	{'0': 2, '1': 1}	False	squared_hinge	10000	l2	1.2705645329
	35893						288717e-05
2	772.34476929	{'0': 2, '1': 1}	False	squared_hinge	10000	l2	1.1384639705
	66574						304647e-06
3	139.59386065	{'0': 2, '1': 1}	False	squared_hinge	10000	l2	8.5326780956
	204183						58724e-06
4	176.02525267	{'0': 2, '1': 1}	False	squared_hinge	1000	l2	2.0026211229
	734538						345322e-05
5	25.519126744	None	False	squared_hinge	5000	l1	0.0038287268
	09519						393157052
6	942.95357055	{'0': 1, '1': 2}	False	squared_hinge	5000	l1	4.4145368764
	7981						94481e-06
7	962.54729494	balanced	False	squared_hinge	5000	l2	0.0007064515
	21113						131755324
8	711.24953243	{'0': 1, '1': 2}	False	squared_hinge	5000	l2	0.0003807949
	80178						336768562
9	596.95015794	balanced	False	squared_hinge	5000	l1	1.7073967431
	6487						528114e-06
10	818.11476592	None	False	squared_hinge	5000	l1	4.5564164652
	24931						23591e-06
11	374.64011884	None	False	squared_hinge	5000	l1	0.0013145103
	73625						232150123
12	181.92496720	None	False	squared_hinge	1000	l2	0.0002796485
	710062						9516062485
13	249.39222914	None	False	squared_hinge	5000	l1	6.4900030207
	887495						16533e-06
14	502.77902322	balanced	False	squared_hinge	5000	l1	2.4004075943
	88615						52162e-05
15	837.81010590	{'0': 1, '1': 2}	False	squared_hinge	10000	l1	2.8437674894
	7328						44368e-06
16	755.46141031	balanced	False	squared_hinge	2000	l2	0.0001537592
	76525						0235481777
17	90.706434532	balanced	False	squared_hinge	5000	l2	0.0001140086
	8208						3701127338
18	798.44512498	None	False	squared_hinge	10000	l1	5.5728075000
	45512						299646e-06
19	174.46642900	None	False	squared_hinge	10000	l2	0.0001177620
	499143						6216380241
20	337.71517140	{'0': 1, '1': 2}	False	squared_hinge	5000	l2	7.4999242723
	3628						9817e-06
21	184.95445552	{'0': 1, '1': 2}	False	squared_hinge	2000	l2	6.2953014845
	552704						16133e-05
22	726.05567887	{'0': 1, '1': 2}	False	squared_hinge	10000	l2	4.1498510458
	02394						48951e-05
23	34.488521115	balanced	False	squared_hinge	1000	l2	5.3572800696
	2184						01829e-06
24	940.23344245	None	False	squared_hinge	1000	l2	4.3977668944
	77784						839625e-06
25	713.34478722	None	False	squared_hinge	1000	l2	8.7890043414
	2995						0548e-06
26	369.75445606	None	False	squared_hinge	5000	l2	2.9589152820
	14045						953114e-05
27	539.44224191	balanced	False	squared_hinge	5000	l2	2.6618976467
	56507						93291e-05

MLP Model Parameters:

Set	learning_rate	hidden_sizes	epochs	dropout_rate	batch_size
1	1e-05	[512, 256, 128]	1500	0.3	32
2	1e-05	[256, 128, 64]	1000	0.3	64
3	1e-05	[256, 128, 64]	1000	0.5	32
4	1e-05	[512, 256, 128, 64]	1000	0.5	128
5	1e-05	[256, 128, 64]	1500	0.4	128
6	1e-05	[512, 256, 128, 64]	1000	0.4	32
7	1e-05	[512, 256, 128]	1000	0.4	64
8	0.0001	[512, 256, 128, 64]	1500	0.5	32
9	1e-05	[512, 256, 128]	1000	0.5	128
10	0.0001	[256, 128, 64]	1000	0.5	64
11	0.0001	[512, 256, 128, 64]	1500	0.5	64
12	1e-05	[512, 256, 128]	1000	0.4	128
13	0.0001	[512, 256, 128]	1500	0.5	128
14	1e-05	[256, 128, 64]	1000	0.4	32
15	0.0001	[512, 256, 128, 64]	1000	0.5	128
16	0.0001	[512, 256, 128]	1000	0.5	128
17	0.0001	[512, 256, 128]	1500	0.5	32
18	0.0001	[256, 128, 64]	1500	0.5	32
19	0.0001	[256, 128, 64]	1500	0.5	128
20	1e-05	[512, 256, 128, 64]	1500	0.5	128
21	1e-05	[512, 256, 128, 64]	1500	0.5	64
22	1e-05	[512, 256, 128, 64]	1000	0.5	32
23	1e-05	[512, 256, 128, 64]	1000	0.5	64
24	1e-05	[512, 256, 128]	1000	0.5	64
25	1e-05	[256, 128, 64]	1500	0.5	64
26	1e-05	[256, 128, 64]	1500	0.5	32
27	1e-05	[512, 256, 128]	1500	0.5	64

SGD Model Parameters:

Set	base_estimator__power_t	base_estimator__penalty	base_estimator__learning_rate	base_estimator__l1_ratio	base_estimator__eta0	base_estimator__class_weight	base_estimator__alpha
1	0.75	l2	optimal	0.5	0.1	balanced	0.1
2	0.5	elasticnet	adaptive	1.0	0.1	None	0.01
3	0.5	l1	optimal	0.75	0.01	None	0.01
4	0.25	l1	optimal	0.0	0.001	balanced	0.01

5	0.5	l1	invscaling	0.5	0.01	balanced	0.001
6	0.5	elasticnet	adaptive	0.75	0.1	balanced	0.01
7	0.75	elasticnet	optimal	0.5	0.1	balanced	0.01
8	0.25	l2	invscaling	1.0	0.01	None	0.01
9	0.5	elasticnet	optimal	0.25	0.1	None	0.01
10	0.25	l2	invscaling	0.25	0.1	None	0.001
11	0.5	l2	invscaling	0.5	0.1	None	0.01
12	0.75	l2	optimal	0.25	0.01	None	0.01
13	0.5	l2	optimal	0.0	0.1	None	0.01
14	0.5	l2	optimal	0.75	0.1	balanced	0.01
15	0.75	elasticnet	adaptive	0.0	0.1	balanced	0.01
16	0.25	l2	optimal	0.5	0.1	balanced	0.001
17	0.75	l2	optimal	0.5	0.01	None	0.001
18	0.75	l2	optimal	0.0	0.001	None	0.001
19	0.75	l1	optimal	0.25	0.001	None	0.001
20	0.25	elasticnet	invscaling	0.5	0.1	None	0.001
21	0.25	elasticnet	invscaling	0.5	0.01	balanced	0.001
22	0.5	elasticnet	optimal	0.5	0.01	None	0.001
23	0.25	elasticnet	optimal	0.25	0.001	None	0.0001
24	0.5	l2	adaptive	0.5	0.1	balanced	0.001
25	0.75	elasticnet	adaptive	0.0	0.1	balanced	0.001
26	0.5	elasticnet	adaptive	0.0	0.1	None	0.001
27	0.25	l2	adaptive	0.75	0.01	None	0.001

STACK Model Parameters:

Set	rf__n_estimators	rf__max_depth	final_estimator__C	et__n_estimators	et__max_depth	dt__max_depth
1	50	10	10.0	50	None	10
2	200	10	100.0	100	None	None
3	100	None	1.0	200	None	None
4	200	None	1.0	100	20	20
5	50	None	100.0	100	None	None
6	50	10	10.0	200	None	20
7	100	20	100.0	100	None	10
8	100	None	1.0	200	None	10
9	200	20	0.1	200	None	None
10	200	20	100.0	50	None	None
11	50	20	10.0	50	None	10
12	100	20	1.0	100	None	None
13	200	20	100.0	100	None	10
14	200	None	0.1	200	None	20
15	50	20	10.0	100	None	None
16	100	None	100.0	200	None	10
17	100	20	10.0	50	None	20
18	200	10	1.0	200	None	20
19	200	None	100.0	200	None	None
20	100	20	10.0	50	None	10
21	100	20	1.0	200	None	20
22	200	None	10.0	100	None	10
23	100	20	100.0	50	None	None
24	200	None	1.0	200	None	20
25	100	20	100.0	200	None	10
26	200	20	10.0	200	None	20
27	50	20	10.0	200	None	None

XGB Model Parameters:

Set	subsample	n_estimators	min_child_weight	max_depth	learning_rate	gamma	colsample_bytree
1	1.0	400	1	6	0.1	0.1	0.5
2	0.75	200	3	8	0.1	0.1	1.0
3	1.0	300	2	7	0.1	0	0.75
4	0.75	400	5	7	0.1	0.1	0.75
5	0.75	200	5	8	0.1	0.4	0.5
6	0.75	200	3	8	0.1	0.4	0.5
7	0.75	400	1	6	0.1	0.1	0.5
8	1.0	400	1	6	0.1	0	0.5
9	0.75	300	3	7	0.1	0.2	0.75
10	0.75	400	3	8	0.1	0.4	0.5
11	1.0	300	5	8	0.1	0.1	0.5
12	0.75	400	5	8	0.1	0.1	0.5
13	1.0	200	4	8	0.1	0.1	0.5
14	0.75	300	4	8	0.1	0	0.5
15	0.75	300	3	7	0.1	0	0.75
16	0.75	400	4	8	0.1	0.4	0.5
17	0.75	300	2	8	0.1	0.3	0.5
18	0.75	400	5	8	0.1	0.2	0.5
19	1.0	400	4	7	0.1	0.1	1.0
20	0.75	300	3	8	0.1	0.2	0.5
21	0.75	300	1	8	0.1	0.1	0.5
22	1.0	300	3	8	0.1	0	0.5
23	1.0	400	2	7	0.1	0.1	0.75
24	0.75	400	2	7	0.1	0.1	0.75
25	0.75	400	2	8	0.1	0.1	1.0
26	1.0	400	5	8	0.1	0	0.75
27	1.0	400	1	7	0.1	0	0.5

APPENDIX C -- ENS Feature Importance Rankings

_Rank	_Feature	_Rank_	_Feature_	Rank_	Feature_
1	Consequence_first	168	LRT_score	335	ppi_combined_52
2	IMPACT	169	gtex_Adipose_-_Visceral_(Omentum)	336	ppi_combined_15
3	ConsScore	170	Sngl1000bp	337	MMseq2_conservation_level_aa_window_15_next
4	VEST4_score	171	EncodeH3K4me2-sum	338	DisoDNAscore_aa_window_3_prev
5	verPhyloP	172	GERPplus_plus_RS	339	DRNApredDNAscore_aa_window_8_next
6	gnomAD_exomes_AF	173	ppi_combined_27	340	ppi_combined_25
7	verPhCons	174	gtex_Cervix_-_Ectocervix	341	gtex_Muscle_-_Skeletal
8	mamPhyloP	175	DFLpredScore_aa_window_3_prev	342	Ex
9	mamPhCons	176	DisoRNAscore_aa_window_15_next	343	haplo
10	Dominant_probability	177	EncodeH3K27ac-max	344	Rare1000bp
11	phyloP100way_vertebrate	178	NearestExonJB_distance	345	tOverlapMotifs
12	M_CAP_score	179	cHmm_E18	346	MMSp_acceptorIntron
13	fathmm_MKL_coding_score	180	after_ANCHOR_15	347	ppi_combined_4
14	GerpS	181	ppi_combined_34	348	phyloP17way_primate
15	MPC_score	182	DisoDNAscore_aa_window_8_next	349	DFLpredScore_aa_window_8_prev
16	cHmm_E4	183	DisoPROscore_aa_window_15_prev	350	ppi_combined_11
17	cHmm_E15	184	EncodeH3K9ac-max	351	gtex_Heart_-_Atrial_Appendage
18	after_IUPRED_8	185	MMSp_acceptor	352	JM
19	priPhCons	186	MMSp_exon	353	DRNApredDNAscore_aa_window_15_next
20	denovo_Zscore	187	MMseq2_conservation_score_aa_window_8_prev	354	PTM
21	Condel_score	188	gtex_Brain_-_Hypothalamus	355	MMseq2_conservation_level_aa
22	MVP_score	189	gtex_Stomach	356	mirSVR-Score
23	after_IUPRED_15	190	gtex_Adipose_-_Subcutaneous	357	ASAquick_normscore_aa_window_8_next
24	phastCons100way_vertebrate	191	ppi_combined_5	358	DisoRNAscore_aa
25	Recessive_probability	192	GDI	359	ppi_combined_42
26	Number_of_paralogs	193	CDS_len	360	ASAquick_normscore_aa_window_3_next
27	MutationTaster_score	194	DRNApredRNAscore_aa_window_15_prev	361	ppi_combined_21
28	dbscSNV-rf_score	195	MoRFchibiScore_aa	362	ppi_combined_3
29	BIOTYPE	196	DRNApredRNAscore_aa_window_3_next	363	SCRIBERscore_aa_window_15_prev
30	cHmm_E7	197	cHmm_E12	364	ppi_combined_1
31	UK10K_AF	198	Selective_pressure	365	ppi_combined_35
32	before_ANCHOR_15	199	cHmm_E14	366	DRNApredRNAscore_aa_window_15_next
33	Rare10000bp	200	DRNApredRNAscore_aa_window_3_prev	367	ProteinLengthChange
34	AF_confidence	201	PSIPRED_helix_aa_window_15_prev	368	MMseq2_conservation_score_aa_window_15_prev
35	ANCHOR2	202	GM12878_fitCons_score	369	Hydrophobicity
36	EncodeH3K36me3-sum	203	MMseq2_conservation_score_aa_window_8_next	370	MOD_RES
37	SpliceAI-don-loss	204	ppi_combined_62	371	MMseq2_conservation_level_aa_window_3_next
38	FATHMM_score	205	gtex_Kidney_-_Medulla	372	Phosphorylation
39	EncodeDNase-sum	206	ppi_combined_49	373	DNA_binding_gbind
40	before_IUPRED_3	207	cHmm_E10	374	DisoDNAscore_aa_window_15_prev
41	GerpN	208	cHmm_E25	375	S_DDG[3D]
42	CADD_raw	209	ppi_combined_32	376	DRNApredDNAscore_aa_window_8_prev
43	phastCons17way_primate	210	PSIPRED_helix_aa_window_15_next	377	DFLpredScore_aa_window_3_next
44	cHmm_E8	211	cHmm_E19	378	MoRFchibiScore_aa_window_15_prev
45	fathmm_XF_coding_score	212	DisoPROscore_aa_window_3_prev	379	MaxEntScan_alt
46	LIST_S2_score	213	SCRIBERscore_aa_window_3_next	380	ppi_combined_46
47	before_ASA_15	214	DRNApredRNAscore_aa_window_8_next	381	Ubiquitination
48	1000Gp3_AF	215	MoRFchibiScore_aa_window_3_prev	382	PSIPRED_strand_aa_window_15_prev
49	MutationAssessor_score	216	haplo_imputed	383	SCRIBERscore_aa_window_8_prev
50	before_RSA_8	217	MoRFchibiScore_aa_window_8_prev	384	ASAquick_rawscore_aa_window_8_prev
51	gtex_Small_Intestine_-_Terminal_Ileum	218	cHmm_E6	385	DRNApredDNAscore_aa
52	MSC_95CI	219	cHmm_E23	386	MMSp_donorIntron
53	EncodeH4K20me1-max	220	HUVEC_fitCons_score	387	DisoDNAscore_aa_window_8_prev
54	s_het	221	gtex_Fallopian_Tube	388	gtex_Heart_-_Left_Ventricle
55	hgmd_mutcount	222	gtex_Testis	389	ppi_combined_57
56	gnom_mutcount	223	ppi_combined_19	390	ppi_combined_61
57	gtex_Colon_-_Transverse	224	polyphen_score2	391	SCRIBERscore_aa_window_15_next
58	after_RSA_8	225	priPhyloP	392	DisoDNAscore_aa_window_3_next
59	EncodeH3K9me3-max	226	ppi_combined_14	393	polyphen_nobs
60	PROVEAN_score	227	gtex_Esophagus_-_Muscularis	394	MMseq2_conservation_level_aa_window_3_prev
61	EncodeH3K4me3-max	228	gtex_Esophagus_-_Gastroesophageal_Junction	395	Mg2+_binding_gbind
62	before_ASA_8	229	ppi_combined_2	396	NMD
63	SpliceAI-acc-loss	230	ASAquick_rawscore_aa_window_15_next	397	ppi_combined_41
64	EncodeH2AFZ-sum	231	gtex_Breast_-_Mammary_Tissue	398	PSIPRED_strand_aa_window_8_prev
65	GC	232	NearestExonJB_len	399	motifEHIPos
66	EncodeH3K36me3-max	233	COSMICvsHAPMAP	400	polyphen_idpsnp
67	before_RSA_15	234	EncodeH2AFZ-max	401	A3D_SCORE
68	after_ANCHOR_8	235	gtex_Whole_Blood	402	gtex_Artery_-_Coronary
69	cHmm_E20	236	ppi_combined_30	403	ppi_combined_8
70	gtex_Spleen	237	ppi_combined_33	404	ppi_combined_9
71	after_ANCHOR_3	238	ppi_combined_31	405	PSIPRED_helix_aa_window_8_prev
72	gtex_Esophagus_-_Mucosa	239	DisoRNAscore_aa_window_3_prev	406	DisoPROscore_aa
73	dbscSNV-ada_score	240	polyphen_idqmin	407	Freq1000bp
74	ASA	241	RNA_binding_gbind	408	DRNApredDNAscore_aa_window_3_prev
75	gtex_Brain_-_Putamen_(basal_ganglia)	242	gtex_Pituitary	409	Transition
76	after_ASA_8	243	ppi_combined_29	410	Dst2SplType
77	ppi_combined_47	244	minDistTSS	411	motifDist
78	num_interactions	245	ASAquick_normscore_aa_window_8_prev	412	DRNApredDNAscore_aa_window_15_prev
79	MoRFchibiScore_aa_window_8_next	246	GenoCanyon_score	413	MMseq2_conservation_score_aa
80	EncodeH3K4me2-max	247	cHmm_E1	414	Ca2+_binding_gbind
81	GerpRS	248	DisoRNAscore_aa_window_8_next	415	EnsembleRegulatoryFeature
82	after_IUPRED_3	249	EncodeH3K79me2-max	416	MMseq2_conservation_level_aa_window_15_prev
83	gtex_Cells_-_EBV-transformed_lymphocytes	250	before_IUPRED_15	417	gtex_Adrenal_Gland
84	GERPplus_plus_NR	251	MMSp_donor	418	COSMICvsSWISSPROT
85	Conservation	252	DFLpredScore_aa_window_15_next	419	AF_dssp_secondary_structure
86	ASAquick_rawscore_aa_window_15_prev	253	ppi_combined_24	420	REQUIRED_FOR_INTER

87	cHmm_E3	254	RSA_Zfit	421	Polarity
88	ppi_combined_20	255	ada_score	422	gnomsingle_mutcount
89	gnomAD_genomes_AF	256	Sngl100bp	423	Volume
90	RSA	257	VB	424	DRNApredRNAscore_aa
91	after_RSA_15	258	Dist2Mutation	425	gtex_Thyroid
92	Clarks_distance	259	EncodeH4K20me1-sum	426	RemapOverlapCL
93	ASAquick_normscore_aa_window_1_prev	260	gtex_Brain_-_Spinal_cord_(cervical_c-1)	427	TSSDistance
94	S_DDG[SEQ]	261	gtex_Brain_-_Cerebellum	428	DisoDNAscore_aa
95	before_IUPRED_8	262	MMseq2_conservation_score_aa_window_15_next	429	ASAquick_rawscore_aa_window_8_next
96	gtex_Brain_-_Hippocampus	263	STABILITY[SEQ]	430	ppi_combined_50
97	Indispensability_score	264	REGION	431	ppi_combined_40
98	cHmm_E2	265	ASAquick_normscore_aa_window_1_next	432	MOD_RES_DESCRIPTION
99	EncodetotalRNA-sum	266	COSMIC	433	PSIPRED_helix_aa_window_8_next
100	before_ANCHOR_8	267	RemapOverlapTF	434	HEME_binding_gbind
101	SpliceAI-don-gain	268	ppi_combined_13	435	ppi_combined_0
102	minDistTSE	269	ppi_combined_16	436	Dst2Splice
103	gtex_Brain_-_Cortex	270	ASAquick_normscore_aa	437	ppi_combined_37
104	EncodeDNase-max	271	SCRIBERscore_aa_window_3_prev	438	PSIPRED_strand_aa
105	gtex_Colon_-_Sigmoid	272	cHmm_E24	439	Methylation
106	cHmm_E21	273	HAPMAP	440	gtex_Liver
107	gtex_Brain_-_Anterior_cingulate_cortex_(BA24)	274	HGMD2003	441	ppi_combined_39
108	MoRFchibiScore_aa_window_15_next	275	EncodeH3K27me3-max	442	Rare100bp
109	EncodeH3K79me2-sum	276	REGION_DESCRIPTION	443	ppi_combined_44
110	ppi_combined_12	277	ppi_combined_58	444	SCRIBERscore_aa
111	after_ASA_15	278	cHmm_E11	445	Mn2+_binding_gbind
112	gtex_Bladder	279	EncodeH3K9me3-sum	446	concavity_score
113	RVIS	280	DisoRNAscore_aa_window_8_prev	447	polyphen_avenhet
114	gtex_Brain_-_Frontal_Cortex_(BA9)	281	EncodeH3K27ac-sum	448	DisoRNAscore_aa_window_15_prev
115	gtex_Brain_-_Nucleus_accumbens_(basal_ganglia)	282	DRNApredDNAscore_aa_window_3_next	449	ppi_combined_48
116	Sngl10000bp	283	ASAquick_normscore_aa_window_3_prev	450	GerpRSpval
117	polyphen_score1	284	DisoPROscore_aa_window_8_next	451	BLOSUM62
118	gtex_Artery_-_Aorta	285	DisoPROscore_aa_window_3_next	452	PSIPRED_helix_aa_window_3_prev
119	EncodeH3K27me3-sum	286	integrated_fitCons_score	453	targetScan
120	rf_score	287	PSIPRED_strand_aa_window_15_next	454	METHYLATION
121	EncodeH3K4me1-sum	288	rel_CDS_pos	455	mirSVR-E
122	before_ANCHOR_3	289	gtex_Prostate	456	PSIPRED_strand_aa_window_3_next
123	cHmm_E22	290	PAM250	457	polyphen_bfact
124	gtex_Uterus	291	MMseq2_conservation_level_aa_window_8_prev	458	MaxEntScan_ref
125	DisoPROscore_aa_window_8_prev	292	ppi_combined_43	459	Freq100bp
126	gtex_Kidney_-_Cortex	293	distance_com	460	mirSVR-Aln
127	ppi_combined_63	294	ppi_combined_18	461	PSIPRED_strand_aa_window_8_next
128	cHmm_E16	295	ppi_combined_60	462	ATP_binding_gbind
129	gtex_Skin_-_Sun_Exposed_(Lower_leg)	296	gtex_Lung	463	PSIPRED_helix_aa_window_3_next
130	SiPhy_29way_logOdds	297	ppi_combined_23	464	DFLpredScore_aa
131	cHmm_E17	298	before_RSA_3	465	PSIPRED_helix_aa
132	bStatistic_y	299	ppi_combined_7	466	motifEScoreChng
133	IUPRED2	300	STABILITY[3D]	467	LINSIGHT
134	gtex_Vagina	301	DFLpredScore_aa_window_8_next	468	rel_cDNA_pos
135	after_RSA_3	302	ASAquick_rawscore_aa_window_3_next	469	SignalP_score_aa_window_15_next
136	AF_Relative_ASA	303	gtex_Nerve_-_Tibial	470	polyphen_dprop
137	cHmm_E9	304	ppi_combined_59	471	O-GLYCOSYLATION
138	MoRFchibiScore_aa_window_3_next	305	isHomomultimer	472	Acetylation
139	ppi_combined_22	306	ppi_combined_26	473	N-GLYCOSYLATION
140	n_contacts	307	gtex_Ovary	474	INTERACTION_REGION
141	before_ASA_3	308	CpG	475	Glycosylation
142	gtex_Cells_-_Cultured_fibroblasts	309	DisoRNAscore_aa_window_3_next	476	polyphen_normasa
143	gtex_Brain_-_Amygdala	310	ppi_combined_54	477	UBIQUITINATION
144	SIFT_score	311	DisoDNAscore_aa_window_15_next	478	SignalP_score_aa_window_3_next
145	after_ASA_3	312	Grantham_x	479	polyphen_avensit
146	gtex_Minor_Salivary_Gland	313	MMseq2_conservation_score_aa_window_3_prev	480	polyphen_dvol
147	gtex_Brain_-_Caudate_(basal_ganglia)	314	rel_prot_pos	481	motifECount
148	EncodeH3K9ac-sum	315	ppi_combined_6	482	polyphen_avenint
149	gtex_Skin_-_Not_Sun_Exposed_(Suprapubic)	316	ppi_combined_56	483	PHOSPHORYLATION
150	ppi_combined_51	317	ppi_combined_45	484	PSIPRED_strand_aa_window_3_prev
151	Freq10000bp	318	DisoPROscore_aa_window_15_next	485	polyphen_hbonds
152	EncodeH3K4me1-max	319	gtex_Pancreas	486	polyphen_mindhet
153	EncodeH3K4me3-sum	320	cHmm_E5	487	S-NITROSYLATION
154	ASAquick_rawscore_aa	321	ppi_combined_55	488	ACETYLATION
155	ppi_combined_28	322	DFLpredScore_aa_window_15_prev	489	SignalP_score_aa
156	MaxEntScan_diff	323	polyphen_idpmax	490	polyphen_mindint
157	ppi_combined_38	324	MMseq2_conservation_score_aa_window_3_next	491	polyphen_mindsit
158	gtex_Brain_-_Cerebellar_Hemisphere	325	H1_hESC_fitCons_score	492	SignalP_score_aa_window_15_prev
159	SCRIBERscore_aa_window_8_next	326	MMseq2_conservation_level_aa_window_8_next	493	SignalP_score_aa_window_8_next
160	polyphen_dscore	327	ppi_combined_17	494	SignalP_score_aa_window_8_prev
161	gtex_Brain_-_Substantia_nigra	328	ppi_combined_53	495	S-GLYCOSYLATION
162	SpliceAI-acc-gain	329	phastCons30way_mammalian	496	C-GLYCOSYLATION
163	gtex_Cervix_-_Endocervix	330	ppi_combined_36	497	SUMOYLATION
164	DRNApredRNAscore_aa_window_8_prev	331	Charge	498	SignalP_score_aa_window_3_prev
165	EncodetotalRNA-max	332	phyloP30way_mammalian	499	MYRISTOYLATION
166	cHmm_E13	333	ASAquick_rawscore_aa_window_3_prev		
167	gtex_Artery_-_Tibial	334	ppi_combined_10		

APPENDIX D -- LOFO Feature Importance Rankings

_Rank	_Feature	_Rank_	_Feature_	Rank_	Feature_
1	gnomAD_exomes_AF	168	gtex_Brain_-_Cerebellum	335	rel_cDNA_pos
2	before_ANCHOR_15	169	ASAquick_rawscore_aa_window_3_next	336	LRT_score
3	Recessive_probability	170	MutationTaster_score	337	GERPplus_plus_NR
4	MMSp_acceptorIntron	171	cHmm_E8	338	Dst2Splice
5	Dist2Mutation	172	SCRIBER[illegible]aa_window_3_prev	339	rel_CDS_pos
6	TSSDistance	173	PSIPR[illegible]d_aa_window_15_next	340	polyphen_mindsit
7	before_ANCHOR_3	174	PSIPRED_strand_aa_window_8_next	341	RemapOverlapCL
8	REQUIRED_FOR_INTER	175	PSIPRED_strand_aa_window_8_prev	342	O-GLYCOSYLATION
9	after_ANCHOR_8	176	PSIPRED_helix_aa	343	gtex_Colon_-_Sigmoid
10	before_RSA_3	177	PSIPRED_helix_aa_window_15_next	344	DRNApredRNAscore_aa_window_3_next
11	cHmm_E14	178	PSIPRED_helix_aa_window_8_prev	345	ppi_combined_54
12	before_ASA_3	179	cHmm_E12	346	ppi_combined_62
13	before_ASA_8	180	MMseq2_conservation_score_aa_window_15_next	347	DRNApredDNAscore_aa_window_8_next
14	Mn2+_binding_gbind	181	MMseq2_conservation_score_aa_window_3_next	348	gtex_Kidney_-_Cortex
15	distance_com	182	cHmm_E3	349	gtex_Heart_-_Left_Ventricle
16	after_ANCHOR_3	183	RSA	350	SIFT_score
17	after_IUPRED_15	184	MMseq2_conservation_level_aa	351	ASAquick_normscore_aa_window_15_next
18	before_IUPRED_8	185	gtex_Adipose_-_Subcutaneous	352	gtex_Esophagus_-_Gastroesophageal_Junction
19	gnomAD_genomes_AF	186	EncodeH3K79me2-sum	353	NearestExonJB_len
20	before_ASA_15	187	MMseq2_conservation_level_aa_window_3_prev	354	DFLpredScore_aa_window_8_next
21	RNA_binding_gbind	188	after_RSA_3	355	DRNApredRNAscore_aa_window_8_prev
22	A3D_SCORE	189	DisoPROscore_aa	356	gtex_Artery_-_Tibial
23	Rare100bp	190	after_ASA_3	357	gtex_Adrenal_Gland
24	REGION	191	EncodeH4K20me1-sum	358	SignalP_score_aa
25	ATP_binding_gbind	192	SCRIBERscore_aa_window_15_next	359	SCRIBERscore_aa_window_8_next
26	PTM	193	ppi_combined_26	360	DisoDNAscore_aa_window_8_next
27	DNA_binding_gbind	194	ANCHOR2	361	SCRIBERscore_aa_window_15_prev
28	after_RSA_15	195	ASAquick_normscore_aa_window_8_next	362	DisoPROscore_aa_window_3_next
29	after_RSA_8	196	ppi_combined_55	363	gtex_Brain_-_Cerebellar_Hemisphere
30	UK10K_AF	197	priPhCons	364	gtex_Minor_Salivary_Gland
31	VEST4_score	198	ASAquick_normscore_aa_window_3_prev	365	DRNApredDNAscore_aa_window_3_prev
32	MMSp_exon	199	ppi_combined_10	366	MoRFchibiScore_aa_window_8_prev
33	MMSp_donor	200	MaxEntScan_alt	367	ppi_combined_21
34	IUPRED2	201	ppi_combined_15	368	ppi_combined_19
35	before_IUPRED_3	202	gtex_Heart_-_Atrial_Appendage	369	gtex_Stomach
36	SpliceAI-don-gain	203	ppi_combined_39	370	S_DDG[3D]
37	ppi_combined_22	204	ppi_combined_56	371	ppi_combined_32
38	RVIS	205	DRNApredDNAscore_aa_window_15_next	372	AF_dssp_secondary_structure
39	RSA_Zfit	206	ppi_combined_63	373	ppi_combined_37
40	DisoRNAscore_aa_window_8_next	207	gtex_Cells_-_EBV-transformed_lymphocytes	374	ppi_combined_24
41	BLOSUM62	208	ppi_combined_33	375	ppi_combined_12
42	after_ASA_15	209	gtex_Ovary	376	ppi_combined_41
43	Consequence_first	210	MMseq2_conservation_score_aa_window_15_prev	377	ppi_combined_43
44	SpliceAI-don-loss	211	gtex_Pancreas	378	ppi_combined_38
45	before_ANCHOR_8	212	polyphen_avensit	379	STABILITY[3D]
46	Glycosylation	213	gtex_Liver	380	DFLpredScore_aa
47	Ubiquitination	214	FATHMM_score	381	DisoPROscore_aa_window_3_prev
48	ppi_combined_13	215	MMseq2_conservation_score_aa_window_8_next	382	DRNApredRNAscore_aa
49	cHmm_E11	216	MoRFchibiScore_aa_window_3_prev	383	MaxEntScan_ref
50	cHmm_E24	217	gtex_Brain_-_Putamen_(basal_ganglia)	384	DisoPROscore_aa_window_15_prev
51	cHmm_E23	218	MoRFchibiScore_aa_window_3_next	385	HAPMAP
52	concavity_score	219	SignalP_score_aa_window_8_next	386	hgmd_mutcount
53	cHmm_E15	220	MMseq2_conservation_level_aa_window_8_prev	387	MutationAssessor_score
54	after_ANCHOR_15	221	COSMIC	388	ppi_combined_46
55	cHmm_E22	222	COSMICvsSWISSPROT	389	EncodeH3K9me3-sum
56	cHmm_E25	223	SignalP_score_aa_window_15_prev	390	EncodeH3K9me3-max
57	ppi_combined_34	224	PSIPRED_helix_aa_window_3_next	391	Freq10000bp
58	1000Gp3_AF	225	polyphen_nobs	392	EncodeH3K27me3-sum
59	verPhyloP	226	gtex_Testis	393	EncodeH3K27me3-max
60	cHmm_E20	227	gtex_Thyroid	394	ASAquick_rawscore_aa_window_8_prev
61	STABILITY[SEQ]	228	PSIPRED_strand_aa_window_15_prev	395	EncodeH3K79me2-max
62	tOverlapMotifs	229	PSIPRED_strand_aa	396	ppi_combined_35
63	Methylation	230	SCRIBERscore_aa_window_8_prev	397	fathmm_XF_coding_score
64	ppi_combined_9	231	gnom_mutcount	398	s_het
65	n_contacts	232	cHmm_E13	399	ppi_combined_57
66	Mg2+_binding_gbind	233	CpG	400	Sngl10000bp
67	DRNApredDNAscore_aa_window_15_prev	234	DisoRNAscore_aa	401	GERPplus_plus_RS
68	ppi_combined_20	235	DisoRNAscore_aa_window_15_next	402	ppi_combined_28
69	ppi_combined_31	236	ASAquick_rawscore_aa_window_15_prev	403	EncodeH3K27ac-sum
70	num_interactions	237	ASAquick_rawscore_aa_window_8_next	404	ppi_combined_4
71	ppi_combined_7	238	mamPhCons	405	ConsScore
72	before_IUPRED_15	239	EncodeH3K36me3-sum	406	gtex_Breast_-_Mammary_Tissue
73	ppi_combined_23	240	gtex_Cells_-_Cultured_fibroblasts	407	gtex_Brain_-_Substantia_nigra
74	ppi_combined_18	241	gtex_Brain_-_Hippocampus	408	gtex_Brain_-_Cortex
75	ppi_combined_50	242	DFLpredScore_aa_window_3_next	409	polyphen_score1
76	AF_confidence	243	EncodeH3K9ac-max	410	gtex_Uterus
77	MMseq2_conservation_level_aa_window_3_next	244	DRNApredDNAscore_aa_window_3_next	411	gtex_Brain_-_Anterior_cingulate_cortex_(BA24)
78	cHmm_E6	245	Grantham_x	412	Dst2SplType
79	cHmm_E9	246	gtex_Cervix_-_Ectocervix	413	motifECount
80	Acetylation	247	ppi_combined_61	414	cHmm_E7
81	MMSp_acceptor	248	gtex_Brain_-_Nucleus_accumbens_(basal_ganglia)	415	SCRIBERscore_aa
82	MMSp_donorIntron	249	GerpRSpval	416	MoRFchibiScore_aa_window_15_next
83	dbscSNV-ada_score	250	ppi_combined_59	417	polyphen_avenhet
84	gtex_Bladder	251	MoRFchibiScore_aa_window_15_prev	418	gtex_Kidney_-_Medulla

85	dbscSNV-rf_score	252	GerpRS	419	Rare10000bp		
86	MMseq2_conservation_level_aa_window_15_prev	253	LIST_S2_score	420	minDistTSE		
87	DisoRNAscore_aa_window_8_prev	254	before_RSA_15	421	polyphen_avenint		
88	INTERACTION_REGION	255	PROVEAN_score	422	ppi_combined_36		
89	after_ASA_8	256	DisoRNAscore_aa_window_3_next	423	EnsembleRegulatoryFeature		
90	Ca2+_binding_gbind	257	gtex_Adipose_-_Visceral_(Omentum)	424	ASAquick_normscore_aa		
91	ppi_combined_1	258	gtex_Artery_-_Aorta	425	polyphen_idpsnp		
92	cHmm_E19	259	gtex_Esophagus_-_Muscularis	426	GC		
93	MOD_RES	260	gtex_Esophagus_-_Mucosa	427	EncodeH2AFZ-sum		
94	ppi_combined_8	261	ppi_combined_51	428	polyphen_bfact		
95	GerpN	262	GDI	429	minDistTSS		
96	ppi_combined_11	263	PHOSPHORYLATION	430	ASAquick_rawscore_aa_window_15_next		
97	SpliceAI-acc-gain	264	gtex_Muscle_-_Skeletal	431	EncodetotalRNA-sum		
98	EncodeH3K4me1-sum	265	gtex_Vagina	432	NearestExonJB_distance		
99	ppi_combined_6	266	SignalP_score_aa_window_8_prev	433	MMseq2_conservation_score_aa		
100	DRNApredRNAscore_aa_window_8_next	267	gtex_Brain_-_Frontal_Cortex_(BA9)	434	gtex_Pituitary		
101	cHmm_E18	268	polyphen_mindhet	435	DFLpredScore_aa_window_3_prev		
102	cHmm_E4	269	gtex_Artery_-_Coronary	436	mamPhyloP		
103	mirSVR-E	270	gtex_Brain_-_Caudate_(basal_ganglia)	437	priPhyloP		
104	motifEScoreChng	271	gtex_Skin_-_Not_Sun_Exposed_(Suprapubic)	438	DisoRNAscore_aa_window_3_prev		
105	gtex_Prostate	272	gtex_Cervix_-_Endocervix	439	DisoPROscore_aa_window_8_next		
106	gtex_Brain_-_Hypothalamus	273	polyphen_dvol	440	CADD_raw		
107	SignalP_score_aa_window_15_next	274	gtex_Brain_-_Amygdala	441	Conservation		
108	PSIPRED_strand_aa_window_3_prev	275	DisoRNAscore_aa_window_15_prev	442	fathmm_MKL_coding_score		
109	MMseq2_conservation_score_aa_window_3_prev	276	ppi_combined_53	443	PSIPRED_strand_aa_window_3_next		
110	ppi_combined_27	277	ASAquick_normscore_aa_window_15_prev	444	DisoDNAscore_aa_window_15_prev		
111	DisoDNAscore_aa	278	S_DDG[SEQ]	445	SCRIBERscore_aa_window_3_next		
112	DRNApredDNAscore_aa	279	ASAquick_normscore_aa_window_8_prev	446	DRNApredRNAscore_aa_window_15_prev		
113	ASAquick_rawscore_aa_window_3_prev	280	EncodeDNase-max	447	ada_score		
114	cHmm_E21	281	M_CAP_score	448	gtex_Whole_Blood		
115	ppi_combined_5	282	ppi_combined_60	449	ppi_combined_25		
116	IMPACT	283	ppi_combined_58	450	Sngl1000bp		
117	CDS_len	284	NMD	451	MOD_RES_DESCRIPTION		
118	Number_of_paralogs	285	HUVEC_fitCons_score	452	ppi_combined_45		
119	isHomomultimer	286	H1_hESC_fitCons_score	453	haplo_imputed		
120	after_IUPRED_3	287	ppi_combined_49	454	polyphen_hbonds		
121	after_IUPRED_8	288	Freq100bp	455	Hydrophobicity		
122	before_RSA_8	289	HGMD2003	456	ACETYLATION		
123	ASA	290	ppi_combined_30	457	UBIQUITINATION		
124	AF_Relative_ASA	291	JM	458	polyphen_score2		
125	mirSVR-Score	292	ppi_combined_44	459	S-NITROSYLATION		
126	Selective_pressure	293	ppi_combined_42	460	gtex_Skin_-_Sun_Exposed_(Lower_leg)		
127	DRNApredDNAscore_aa_window_8_prev	294	rf_score	461	MMseq2_conservation_level_aa_window_15_next		
128	motifDist	295	ppi_combined_40	462	METHYLATION		
129	SpliceAI-acc-loss	296	Volume	463	N-GLYCOSYLATION		
130	Phosphorylation	297	ASAquick_rawscore_aa	464	verPhCons		
131	Indispensability_score	298	EncodeH4K20me1-max	465	Condel_score		
132	gtex_Fallopian_Tube	299	DFLpredScore_aa_window_15_next	466	phyloP30way_mammalian		
133	ppi_combined_47	300	MoRFchibiScore_aa_window_8_next	467	phyloP17way_primate		
134	ppi_combined_48	301	DisoPROscore_aa_window_15_next	468	ppi_combined_52		
135	cHmm_E16	302	DFLpredScore_aa_window_15_prev	469	Sngl100bp		
136	ppi_combined_3	303	SiPhy_29way_logOdds	470	phastCons100way_vertebrate		
137	cHmm_E2	304	DisoPROscore_aa_window_8_prev	471	polyphen_idpmax		
138	cHmm_E1	305	DisoDNAscore_aa_window_15_next	472	gtex_Lung		
139	gtex_Colon_-_Transverse	306	gnomsingle_mutcount	473	GenoCanyon_score		
140	gtex_Nerve_-_Tibial	307	DisoDNAscore_aa_window_3_prev	474	EncodeH3K9ac-sum		
141	LINSIGHT	308	MVP_score	475	phastCons17way_primate		
142	MMseq2_conservation_level_aa_window_8_next	309	ppi_combined_17	476	haplo		
143	SignalP_score_aa_window_3_prev	310	PSIPRED_helix_aa_window_15_prev	477	BIOTYPE		
144	ppi_combined_2	311	MaxEntScan_diff	478	Dominant_probability		
145	ppi_combined_0	312	denovo_Zscore	479	integrated_fitCons_score		
146	cHmm_E5	313	PSIPRED_helix_aa_window_3_prev	480	Clarks_distance		
147	ASAquick_normscore_aa_window_3_next	314	MMseq2_conservation_score_aa_window_8_prev	481	EncodeH3K27ac-max		
148	gtex_Brain_-_Spinal_cord_(cervical_c-1)	315	PAM250	482	Polarity		
149	HEME_binding_gbind	316	S-GLYCOSYLATION	483	polyphen_dprop		
150	ProteinLengthChange	317	SUMOYLATION	484	polyphen_normasa		
151	ppi_combined_14	318	Transition	485	motifEHIPos		
152	gtex_Small_Intestine_-_Terminal_Ileum	319	rel_prot_pos	486	Freq1000bp		
153	ppi_combined_29	320	polyphen_dscore	487	Rare1000bp		
154	ppi_combined_16	321	COSMICvsHAPMAP	488	DRNApredRNAscore_aa_window_3_prev		
155	bStatistic_y	322	C-GLYCOSYLATION	489	polyphen_mindint		
156	targetScan	323	Charge	490	Ex		
157	PSIPRED_helix_aa_window_8_next	324	cHmm_E10	491	DisoDNAscore_aa_window_8_prev		
158	EncodetotalRNA-max	325	RemapOverlapTF	492	polyphen_idqmin		
159	SignalP_score_aa_window_3_next	326	phyloP100way_vertebrate	493	GM12878_fitCons_score		
160	gtex_Spleen	327	phastCons30way_mammalian	494	REGION_DESCRIPTION		
161	MoRFchibiScore_aa	328	MYRISTOYLATION	495	EncodeH2AFZ-max		
162	mirSVR-Aln	329	cHmm_E17	496	EncodeH3K4me1-max		
163	GerpS	330	MPC_score	497	EncodeH3K4me2-sum		
164	DisoDNAscore_aa_window_3_next	331	EncodeH3K4me2-max	498	MSC_95CI		
165	EncodeH3K36me3-max	332	EncodeH3K4me3-sum	499	VB		
166	DFLpredScore_aa_window_8_prev	333	EncodeH3K4me3-max				
167	DRNApredRNAscore_aa_window_15_next	334	EncodeDNase-sum				

APPENDIX E - Weighting Scheme Results


```
ENS-XGB Results:

Weight set 1: [0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111]
Top 5 models:
Model 496.0: 0.8070
Model 446.0: 0.8030
Model 443.0: 0.8029
Model 490.0: 0.8018
Model 452.0: 0.8018

Weight set 2: [0.2, 0.1, 0.1, 0.2, 0.1, 0.1, 0.2, 0.1, 0.1]
Top 5 models:
Model 496.0: 0.9111
Model 446.0: 0.9053
Model 402.0: 0.9044
Model 443.0: 0.9042
Model 442.0: 0.9037

Weight set 3: [0.2, 0.2, 0.2, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
Top 5 models:
Model 496.0: 0.9673
Model 446.0: 0.9617
Model 443.0: 0.9616
Model 490.0: 0.9616
Model 452.0: 0.9606

Weight set 4: [0.1, 0.1, 0.1, 0.2, 0.2, 0.2, 0.1, 0.1, 0.1]
Top 5 models:
Model 496.0: 0.9519
Model 446.0: 0.9463
Model 443.0: 0.9461
Model 490.0: 0.9455
Model 442.0: 0.9451

Weight set 5: [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.2, 0.2, 0.2]
Top 5 models:
Model 496.0: 0.9860
Model 443.0: 0.9828
Model 446.0: 0.9827
Model 402.0: 0.9816
Model 452.0: 0.9811

Weight set 6: [0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1]
Top 5 models:
Model 496.0: 1.1759
Model 446.0: 1.1699
Model 443.0: 1.1687
Model 442.0: 1.1679
Model 402.0: 1.1677
```

```
LOFO-XGB Results:

Weight set 1: [0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111,
0.1111111111111111]
Top 5 models:
Model 256.0: 0.8459
Model 269.0: 0.8450
Model 270.0: 0.8443
Model 258.0: 0.8442
Model 302.0: 0.8437

Weight set 2: [0.2, 0.1, 0.1, 0.2, 0.1, 0.1, 0.2, 0.1, 0.1]
Top 5 models:
Model 256.0: 0.9815
Model 258.0: 0.9788
Model 270.0: 0.9786
Model 269.0: 0.9785
Model 234.0: 0.9782

Weight set 3: [0.2, 0.2, 0.2, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
Top 5 models:
Model 256.0: 1.0173
Model 269.0: 1.0165
Model 270.0: 1.0158
Model 258.0: 1.0158
Model 234.0: 1.0151

Weight set 4: [0.1, 0.1, 0.1, 0.2, 0.2, 0.2, 0.1, 0.1, 0.1]
Top 5 models:
Model 256.0: 1.0023
Model 269.0: 1.0008
Model 270.0: 1.0004
Model 258.0: 0.9995
Model 234.0: 0.9988

Weight set 5: [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.2, 0.2, 0.2]
Top 5 models:
Model 256.0: 1.0256
Model 269.0: 1.0247
Model 302.0: 1.0247
Model 299.0: 1.0245
Model 294.0: 1.0242

Weight set 6: [0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1]
Top 5 models:
Model 256.0: 1.2479
Model 269.0: 1.2459
Model 258.0: 1.2450
Model 270.0: 1.2449
Model 234.0: 1.2438
```

```
ENS-LGBM Results:

Weight set 1: [0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111,
0.1111111111111111]
Top 5 models:
Model 414.0: 0.7749
Model 430.0: 0.7747
Model 400.0: 0.7745
Model 433.0: 0.7745
Model 413.0: 0.7743

Weight set 2: [0.2, 0.1, 0.1, 0.2, 0.1, 0.1, 0.2, 0.1, 0.1]
Top 5 models:
Model 318.0: 0.8705
Model 430.0: 0.8698
Model 433.0: 0.8696
Model 414.0: 0.8695
Model 400.0: 0.8692

Weight set 3: [0.2, 0.2, 0.2, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
Top 5 models:
Model 414.0: 0.9354
Model 458.0: 0.9349
Model 430.0: 0.9342
Model 400.0: 0.9341
Model 433.0: 0.9341

Weight set 4: [0.1, 0.1, 0.1, 0.2, 0.2, 0.2, 0.1, 0.1, 0.1]
Top 5 models:
Model 414.0: 0.9136
Model 430.0: 0.9125
Model 458.0: 0.9125
Model 400.0: 0.9122
Model 413.0: 0.9118

Weight set 5: [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.2, 0.2, 0.2]
Top 5 models:
Model 318.0: 0.9434
Model 433.0: 0.9425
Model 430.0: 0.9422
Model 400.0: 0.9421
Model 305.0: 0.9421

Weight set 6: [0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1]
Top 5 models:
Model 414.0: 1.1279
Model 430.0: 1.1278
Model 433.0: 1.1276
Model 400.0: 1.1272
Model 458.0: 1.1272
```

```
LOFO-LGBM Results:

Weight set 1: [0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111, 0.1111111111111111,
0.1111111111111111]
Top 5 models:
Model 392.0: 0.8234
Model 397.0: 0.8232
Model 358.0: 0.8229
Model 369.0: 0.8227
Model 352.0: 0.8227

Weight set 2: [0.2, 0.1, 0.1, 0.2, 0.1, 0.1, 0.2, 0.1, 0.1]
Top 5 models:
Model 392.0: 0.9460
Model 397.0: 0.9453
Model 365.0: 0.9444
Model 358.0: 0.9442
Model 352.0: 0.9441

Weight set 3: [0.2, 0.2, 0.2, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]
Top 5 models:
Model 392.0: 0.9880
Model 397.0: 0.9878
Model 358.0: 0.9876
Model 369.0: 0.9875
Model 352.0: 0.9874

Weight set 4: [0.1, 0.1, 0.1, 0.2, 0.2, 0.2, 0.1, 0.1, 0.1]
Top 5 models:
Model 392.0: 0.9696
Model 358.0: 0.9695
Model 397.0: 0.9694
Model 369.0: 0.9692
Model 352.0: 0.9687

Weight set 5: [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.2, 0.2, 0.2]
Top 5 models:
Model 392.0: 1.0065
Model 397.0: 1.0061
Model 374.0: 1.0059
Model 365.0: 1.0057
Model 375.0: 1.0056

Weight set 6: [0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1]
Top 5 models:
Model 392.0: 1.2084
Model 397.0: 1.2079
Model 358.0: 1.2069
Model 365.0: 1.2068
Model 352.0: 1.2066
```

Appendix F -- Directory Structure & Usage Notes

Conda environments

1. I used two conda environments for this project. `logofunc3` has the versions of Scikit-learn and other packages that are specified in the LoGoFunc documentation. This environment should be used for all of the training and testing procedures.
2. The other environment is `tabulate` which I used for all of the data analysis and graphing operations.
3. The table that describes all of the notebooks contains a column that specifies which environment should be used to run those blocks of code.
4. I exported the packages I explicitly installed for each environment using the `conda env export --from-history > *.yml` command, so it should install the required packages regardless if platform when you run `conda env create -f {env name}.yml`. The YAML files are in the `Conda env` directory.
5. If for some reason this does not work these are the packages required by LoGoFunc:

```
1. conda create -n logofunc python=3 pandas=1.5.0 joblib lightgbm=3.2.1
scikit-learn=1.1.2 imbalanced-learn=0.8.0
2. conda activate logofunc
```

Directory structure

WORKSPACE	FOLDER	NOTEBOOK	DESCRIPTION	ENVIRONMENT
Classifiers	scripts	graphing	Graphs the classifier metrics.	tabulate
Classifiers	scripts	train	Train the LGBM and LGBM-P classifiers.	logofunc3
Classifiers	scripts	encode	Encodes feature names for XGBoost.	tabulate
Classifiers	classifiers	LinearSVM	Assess the LinearSVM classifier.	logofunc3
Classifiers	classifiers	SGDClassifier	Assess the SGD classifier.	logofunc3
Classifiers	classifiers	AdaBoost	Assess the AdaBoost classifier.	logofunc3

WORKSPACE	FOLDER	NOTEBOOK	DESCRIPTION	ENVIRONMENT
Classifiers	classifiers	CatBoost	Assess the CatBoost classifier.	logofunc3
Classifiers	classifiers	ExtraTrees	Assess the ExtraTrees classifier.	logofunc3
Classifiers	classifiers	XGBoost	Assess the XGBoost classifier.	logofunc3
Classifiers	classifiers	PyTorch	Assess the Pytorch MLP classifier.	logofunc3
Classifiers	classifiers	Stacking	Assess the Stacking classifier.	logofunc3
Classifiers	classifiers	LGBM-P test	Assess the LGBM-P classifier.	logofunc3
Classifiers	classifiers	LGBM test	Assess the LGBM classifier.	logofunc3
LOFO Ranker	scripts	graphing	Graphs the Top 25 feature accuracies.	tabulate
LOFO Ranker	scripts	numbering	Creates a numbered JSON of features.	tabulate
LOFO Ranker	scripts	train	Train all 499 LOFO models.	logofunc3
LOFO Ranker	scripts	test	Test all 499 LOFO models.	logofunc3
LOFO Ranker	scripts	encode_all	Encodes feature names for XGBoost.	tabulate
LOFO Ranker	scripts	generate_subsets	Generates the LOFO subsets.	tabulate
LOFO Ranker	scripts	decode	Decodes the ranked feature names.	tabulate
LOFO Ranker	scripts	ranked_subsets	Creates subsets based on LOFO rankings.	tabulate
ENS Ranker	scripts	generate_subsets	Creates subsets based on ENS rankings.	tabulate
ENS Ranker	scripts	ensemble_ranking	Runs the ensemble ranking algorithms.	tabulate
ENS Ranker	scripts	decoding	Decodes the features	tabulate

WORKSPACE	FOLDER	NOTEBOOK	DESCRIPTION	ENVIRONMENT
			after ranking.	
ENS Ranker	scripts	variance	Plots the mean rankings against the variances.	tabulate
LOFO-LGBM	LOFO-LGBM	roc curves	Compute AP, find best model, plot ROC curves.	tabulate
LOFO-LGBM	LOFO-LGBM	train	Train LGBM ensemble with LOFO ranked subsets.	logofunc3
LOFO-LGBM	LOFO-LGBM	test	Test all 499 LGBM models.	logofunc3
LOFO-LGBM	LOFO-LGBM	graphs	Plots both overall and class-wise metrics.	tabulate
LOFO-LGBM	LOFO-LGBM	sort-metrics	Merges metrics collected by test script.	tabulate
LOFO-XGB	LOFO-XGB	roc curves	Compute AP, find best model, plot ROC curves.	tabulate
LOFO-XGB	LOFO-XGB	train	Train XGB ensemble with LOFO ranked subsets.	logofunc3
LOFO-XGB	LOFO-XGB	test	Test all 499 XGB models.	logofunc3
LOFO-XGB	LOFO-XGB	graphs	Plots both overall and class-wise metrics.	tabulate
LOFO-XGB	LOFO-XGB	sort-metrics	Merges metrics collected by test script.	tabulate
ENS-LGBM	ENS-LGBM	roc curves	Compute AP, find best model, plot ROC curves.	tabulate
ENS-LGBM	ENS-LGBM	train	Train LGBM ensemble with ENS ranked subsets.	logofunc3
ENS-LGBM	ENS-LGBM	test	Test all 499 LGBM models.	logofunc3
ENS-LGBM	ENS-LGBM	graphs	Plots both overall and class-wise metrics.	tabulate

WORKSPACE	FOLDER	NOTEBOOK	DESCRIPTION	ENVIRONMENT
ENS-LGBM	ENS-LGBM	sort-metrics	Merges metrics collected by test script.	tabulate
ENS-LGBM	ENS-LGBM	generate_subsets	Generates the ENS subsets.	tabulate
ENS-XGB	ENS-XGB	roc curves	Compute AP, find best model, plot ROC curves.	tabulate
ENS-XGB	ENS-XGB	train	Train XGB ensemble with ENS ranked subsets.	logofunc3
ENS-XGB	ENS-XGB	test	Test all 499 XGB models.	logofunc3
ENS-XGB	ENS-XGB	graphs	Plots both overall and class-wise metrics.	tabulate
ENS-XGB	ENS-XGB	sort-metrics	Merges metrics collected by test script.	tabulate
ENS-XGB	ENS-XGB	generate_subsets	Generates the ENS subsets.	tabulate

Classifiers directory

1. The `LGBM` and `LGBM-P` models were trained using the original LoGoFunc `train.py` script. The only difference is `LGBM` was trained using the dataset without the `Protein_dom` feature, while `LGBM-P` used the full dataset. `LGBM` was intended to serve as the benchmark to compare other classifiers to, and `LGBM-P` was included to show the impact of removing this feature. The final results showed that `LGBM` performed slightly better during testing than `LGBM-P` did, indicating that removing this feature should have negative consequences.
2. `utils.py` comes straight from the LoGoFunc GitLab repository and contains pre-processing functions that are accessed by the classifier notebooks. `utilsencoded.py` is identical aside from having feature names encoded so that they will be valid for use with XGBoost when `XGBoost.ipynb` is run.
3. The `classifiers` folder contains one notebook for each of the classifiers I assessed. These handle hyperparameter tuning, training, and testing in one notebook. The model ensembles are stored in the `models` directory, test results go to `results`, and the various performance metrics end up in the `metrics` directory.
4. The `scripts` directory contains `encode.ipynb`, which is only used to encode feature names for use with XGBoost (because it considers many of the original names invalid!). It

also contains `graphing.ipynb`, which was used to generate the classifier comparison plot for the report. The `train.ipynb` in this folder is just the original LoGoFunc training script ported to a notebook for convenience.

Ensemble ranking directory

1. `scripts/ensemble_ranking.ipynb` is the core set of code that first loads the LoGoFunc datasets with 500 features, then drops the `Protein_dom` column and exports datasets with 499 features. The new reduced datasets are used for the rest of the process.
2. Next, the pre-processing snippet encodes, imputes, and scales the dataset. This increases feature count from 499 to 539.
3. Now the `ranks` dictionary is initialized and the 12 ranking algorithms rank feature importances in series. This will take several hours (or longer).
4. Next, the mean rankings are computed and exported as `feature_ranks.csv`.
5. `decode_rankings.ipynb` runs through a series of steps to decode the ranked features using `decoding_template.json` as a guide. The final rankings are exported as `decoded_ranks.json`.
6. The `generate_subsets.ipynb` notebook uses the decoded rankings to generate train and test subsets, each containing the top "x" most important features, ranging from 1 to 499. These will be used by ENS-XGB and ENS-LGBM for training/testing.
7. The `variance.ipynb` just plotted mean ranking scores against variances for the report.

LOFO ranking directory

1. Because XGBoost considers some of the characters in the feature names invalid, they must first be encoded by running `scripts/encode_all.ipynb`. This will generate `data/X_train_encoded.csv` and `data/X_test_encoded.csv`, which will be used as the base datasets for the remainder of the ranking process. This only encodes the feature names and leaves the actual data intact. This script also encodes two JSON files needed for the preprocessing pipeline called `data/negone_median.json` and `data/patterns.json`. These preprocessing artifacts were inherited from the original LoGoFunc repository.
2. Next, running `scripts/generate_subsets.ipynb` will create the train and test subsets, each missing a single feature. These subsets are stored in `data/train_subsets` and `data/test_subsets`, respectively.
3. Running `scripts/train.ipynb` will iterate over every subset in `data/train_subsets` and create a model for each one, forming the basis for the ranking analysis.

4. After training all of the models, `scripts/test.ipynb` will iterate over every model stored in the `models` directory. This populates both the `results` and `metrics` directories and ranks the models in reverse order according to their overall accuracy, which is stored in `metrics/ranked_by_accuracy.csv`.
5. Running the `scripts/decode.ipynb` notebook will take `metrics/ranked_by_accuracy.csv` and decode the feature names before storing them as `metrics/accuracy_rankings.json`. This makes the ranked features human readable.
6. Finally, `scripts/ranked_subsets.ipynb` reads `metrics/accuracy_rankings.json`, `data/X_train_encoded.csv` and `data/X_test_encode.csv` to generate ranked subsets with the Top "X" features for both training and test datasets. These are stored in `data/ranked_subsets/train_features_csv` and `data/ranked_subsets/test_features_csv`, respectively. These are the subsets that will be used for training and testing LOFO-LGBM and LOFO-XGB.
7. `scripts/graphing.ipynb` is used for generating the graph of the Top 25 features based on the LOFO ranking. The table in my report was generated manually using the data found in `metrics/ranked_by_accuracy.csv`.

ENS-XGB and ENS-LGBM directory

1. The `generate_subsets.ipynb` notebook uses the decoded rankings to generate train and test subsets, each containing the top "x" most important features, ranging from 1 to 499. These will be used by ENS-XGB and ENS-LGBM for training/testing. For ENS-XGB, this notebook subsequently encodes the feature names because otherwise XGBoost will terminate with an error.
2. `train.ipynb` iterates over all 499 of the subsets and trains each model ensemble. This takes over a day to run, so I used the `tqdm` package to provide a progress bar.
3. `test.ipynb` iterates over all 499 of the trained model ensembles and exports results to the `results` folder and the various performance metrics to the `metrics` folder.
4. `sort-metrics.ipynb` is just used for combining the individual metrics and reports into a new CSV.
5. `graphs.ipynb` first computes macro-recall from the confusion matrices (I had originally used weighted recall, which gave the same values as ACC for every model). Then it plots overall performance metrics followed by class-wise performance metrics. I used PrettyTable to display data in a tabular format for ease of reading.
6. `roc curves.ipynb` calculates AP score (I had originally just used PREC), performs the weighting/sensitivity analysis, arrives at optimal model (feature count), and then allows you to plot the ROC curves for that model. I. used PrettyTable again to display data in a tabular format.

LOFO-XGB and LOFO-LGBM directory

1. This workspace is pre-populated with the ranked subsets generated by the `LOFO Ranker` workspace. These are found in `data/ranked_subsets/train_features_csv` and `data/ranked_subsets/test_features_csv`.
2. `LOFO-LGBM/train.ipynb` will iterate over all of the training subsets contained in the `data/ranked_subsets/train_features_csv` directory and generate an ensemble of 27 models, which are stored in the `LOFO-LGBM/models` directory.
3. Running `LOFO-LGBM/test.ipynb` will evaluate all of the models and collect metrics.
4. Next, using `LOFO-LGBM/sort_metrics.ipynb` will run analysis on the collected metrics and collate them into merged and sorted files stored in `LOFO-LGBM/metrics`.
5. `scripts/ranking_script.ipynb` performs sensitivity analysis on the class-wise metrics to aid in selecting the overall highest-performing model (feature count).
6. `scripts/roc curve.ipynb` computes AP and plots the ROC curve for the best model. It also displays the data for the best model in a table using the `prettytable` package.
7. `LOFO-LGBM/graphs.ipynb` creates graphs both for overall and class-wise metrics.