STRATEGY FOR ACCURATE CNV DETECTION AND ML ALGORITHM FOR CLASSIFYING NGS VARIANTS

Alba Malagón Márquez

Scientific director: Jairo Rodríguez Lumbiarres[1]

[1]Research and Development Department, qGenomics Laboratory, Barcelona, Spain.

Abstract

Motivation: Next-generation sequencing (NGS) has become a revolutionary technology in clinical genetics and diagnostics. Nevertheless, the interpretation of the vast amount of data generated by NGS remains a challenging task since it requires expert assistance to distinguish the small number of clinically relevant variants among a large number of non-clinically significant ones. This project focuses on two closely related strategies: the improvement in the detection specificity of copy number variants (CNVs) and the development and implementation of a machine learning (ML) tool for the classification of single-nucleotide variants (SNVs) and short insertions/deletions (INDELs).

Results: A substantial impact on the efficiency of genomic analysis procedures of clinical samples has been shown. By recognising incompatible scenarios of CNVs, the number of dubious deletion calls has been reduced by a 5.86%, and thus the overall analysis time per sample has been slightly shortened. To aid in variant classification, we have designed a machine learning tool using the Random Forest algorithm since it has been the most beneficial predictive model. Our evaluation on a large set has confirmed that our approach successfully predicts variant classification by a percentage above 96%. Furthermore, the *benign* and *risk factor* variants could be filtered out, accounting for approximately 10.17% of total variants, due to their excellent performance measures. Consequently, this project significantly impacts variant interpretation efficiency, which is a hugely relevant achievement as many human resources are devoted to interpretation. Both implementations provide countless advantages, from automation to efficiency and continuous improvement, among other benefits.

1. Introduction

Notwithstanding nearly all children are born healthy, about 5% of couples with no family history or any unhealthy suspicion possible have a high risk of sharing a mutation in the same gene (Abulí *et al.*, 2016). Therefore, it is highly recommended that couples planning to have children consider undergoing a genetic test to determine the carrier status of the significant recessive genetic diseases. Although each of these diseases is rare and uncommon, the at-risk couples can be identified and provided with appropriate genetic counselling.

The *qCarrier Plus* (Abulí *et al.*, 2016) is a laboratory test developed by *qGenomics*, in collaboration with *Dexeus*, which uses next-generation sequencing (NGS) technology to detect +9,000 mutations reported in the databases being the cause of +300 recessive diseases. In a single analysis, the NGS allows the study of several types of mutations: single nucleotide variants (SNVs), minor insertions/deletions, copy number variants, and rearrangements at selected genes.

Classification and interpretation of those variants is a crucial step in reporting clinical results. Nevertheless, the numerous variants detected, many of which have never been seen before, make the interpretation of clinically relevant variants challenging (Cutting, 2014). As a result, clinical laboratories worldwide have widely adopted recommendations by the *American College of Medical Genetics and Genomics* (ACMG) and the *Association for Molecular Pathology* (AMP) (Richards *et al.*, 2015) to guide clinical interpretation of sequence variants (Zhang *et al.*, 2020). Despite this, manual curation is time-consuming on large amounts of variants, being among the many challenges that expert interpreters face, thus requiring the need for automation (Pandey *et al.*, 2012).

CNVs, by definition, consist of a gain or loss of genomic DNA, giving rise to an abnormal number of copies in a region ranging from tens to millions of bases long (Eichler, 2008). With the advent of NGS, many bioinformatics and statistical tools have been developed for CNV detection from the sequencing data (Zare *et al.*, 2017). While most of them struggle with small CNVs, which are frequently involved in several genetic diseases, these tools perform well for large CNVs (Moreno-Cabrera *et al.*, 2020). The software tested in this work belongs to the read depth methodology, which identifies the presence or absence of CNVs and determines their number of copies. A standard read depth approach uses a combination of samples (or a single one) as a reference to control for the variability at the capture and sequencing steps (Plagnol *et al.*, 2012). Despite this, technical variability between samples makes the analysis complicated and can give rise to spurious CNV calls. *ExomeDepth* (Plagnol, 2019), the CNV calling algorithm we have focused on during this project, is meant to handle this technical variability (Plagnol *et al.*, 2012).

ExomeDepth is effective across many exome datasets, even for small (e.g. one to two exons) and heterozygous deletions. It utilises a robust model for the read count data to build an optimised reference set to maximise the CNV detection capability (Plagnol *et al.*, 2012). Importantly, *ExomeDepth* assumes that the CNV of interest is absent from the aggregate reference set, which combines exomes from the same batch and is optimised for each exome. Thus, this tool is noteworthily suited to detect rare CNV calls, commonly for rare Mendelian disorder analysis (Plagnol, 2019).

Typically, *qGenomics* experts start the variant interpretation journey with an annotated list of identified high-quality variants that must be interpreted to determine whether they are causative for disease. The interpretation procedure includes looking if the variant has direct entries in several databases, including ClinVar, LOVD, HGMD and the Genome Aggregation

Database (*GnomAD*), to annotate the quantity and frequency of homozygous and hemizygous the variant has. Interpretation also requires extensive literature searches, which at present cannot be easily automated. With all of this information uncovered, the interpreters make a judgment call on reporting each variant based on the evidence and their deep clinical expertise. The time required to review and validate each variant depends on certain factors, including the number of papers available. About 3 minutes need to be spent for each novel variant that has no entry in the University of California, Santa Cruz (UCSC); about 15 minutes for those having one or a few papers; and variants with several papers, especially if they have conflicting information between them, can take up to 30 minutes. Sometimes, if the variant is complicated, that time may increase up to an hour. Thus, it takes much time to analyse all variants of each test.

Therefore, this project has been motivated by the need for robust clinical decision support solutions at *qGenomics*. With this background in mind, this project aims to improve the efficiency of NGS variants interpretation by reducing the time spent by experts analysing SNPs, INDELs and CNVs variants.

1.1 Objectives

The motivation behind this project is to reduce the time spent by the experts analysing the variants through two related strategies. The first, the improvement in the detection of copy number variants by assessing their credibility. The second, improved efficiency in the tertiary analysis of NGS data by implementing machine learning tools to predict their clinical importance. In essence, both parts are aimed at improving the efficiency of NGS processes in the interpretation part.

1.1.1 CNVs part

The proposal is to develop a strategy that predicts whether a copy number variant has been correctly called or not by taking advantage of the potential presence of small variants (SNV and INDEL) in a CNV. This strategy aims at providing a more accurate CNVs filtering strategy, so looking for incompatible cases will be the key.

While the work presented in this project focuses on the establishment and validation of a method able to assign a score to each CNV prediction in the dataset, the main goal will be to use this evaluation method to systematically reduce the number of dubious calls that need to be interpreted and thus should shorten the overall analysis time per sample. In other words, the challenge is to identify the reliable calls of Exome Depth with sufficient sensitivity and specificity to be used as a screening step in a diagnostic setting.

The research plan objectives consist of: i) developing and implementing a method that readily identifies *ExomeDepth*'s false positive CNVs calls; ii) validation of the results against array data to examine how well the strategy performs.

1.1.2 ML part

The objective is to develop and implement a machine learning algorithm for predicting SNPs and INDELs variant classification. This predicted classification aims to reduce the time spent by the experts in the interpretation process. The purpose is to make predictions with a probability score and use successful predictions as a hard filter in the interpretation pipeline.

The aim is to reduce the number of variants that need to be interpreted since a large number of SNPs and INDEL variants exist, and only a few of them end up being reported concerning a particular phenotype due to their known or potential clinical relevance. The primary purpose is to predict their classification based on thousands of variants already labelled by experts at *qGenomics*. Several approaches for data preprocessing, data architecture, and algorithms will be assessed, including Decision Trees, Random Forest and Artificial Neural Networks. The machine-learned approach is designed to have access to underlying data used in manual variant classification, including functional prediction, splice prediction, quality filters, allelic balance, read depth, among many more. The resulting software will predict specific categories — "artefact", "benign", "pathogenic", "riskfactor", "polymorphism", "vous", and "others"— to describe the variants identified.

Since the objective is to design a machine learning algorithm, the research plan objectives consist of: i) searching and selecting the algorithms to be analysed; ii) collecting and preprocessing the data; iii) training and evaluating model performances; iv) hyperparameter tuning; v) making predictions.

2. Methods

2.1 NGS data generation

Throughout this project, we have taken advantage of data already generated by *qGenomics*, which uses NGS technology to detect variants. The sequencing is performed on Illumina NextSeq500 and NovaSeq6000 instruments, and the reads are aligned against the human reference genome (b37) using the Burrows-Wheeler Aligner (BWA-MEM) algorithm (Li, 2013). Alignment post-processing is performed according to Genome Analysis Toolkit (GATK) (McKenna *et al.*, 2010) best practice guidelines. Variant calling of SNPs and INDELs is performed using both GATK HaplotypeCaller v3.7 and UnifiedGenotyper. In contrast, CNVs calls are performed using *ExomeDepth* (Plagnol, 2019), which identifies CNVs based on variations in depth of coverage (DOC) of aligned sequence reads against the reference genome (Kadalayil *et al.*, 2014).

The dataset comprising SNPs and INDELs data of a single sample is made up of approximately 10,000 variants and a total of 35 features for the enrichment panel *qCarrier*. Variant interpretation specialists examine an annotated dataset and majorly classify each variant as one of the six following categories: "pathogenic", "benign", "vous", "polymorphism", "artefact", and "riskfactor". This dataset is distinguished for carrying all sample variants reported: single nucleotide variants, minor insertions/deletions, copy number variants and specific rearrangements coming from distinct procedures. It likewise includes a number of annotations that help interpreters determine which sort of variant they are in front of, and this is the knowledge we will use to train a supervised machine learning algorithm.

2.2 CNVs detection

The scripts developed, programmed in python, take advantage of a broadly used package: *pandas* (v0.23.1) (McKinney, 2010), which is extremely useful for storing and efficiently dealing with all datasets. We also import *NumPy* (v1.19.5) (Harris *et al.*, 2020) since it contains an extensive collection of high-level mathematical functions to operate with, the *Matplotlib* package (v2.1.2) (Hunter, 2007) for making visualisations, and the *binom* function from *Scipy* package (v1.5.4) (Virtanen *et al.*, 2020), which allows us to apply a binomial discrete random variable.

2.2.1 ExomeDepth

ExomeDepth (Plagnol *et al.*, 2012) is a CNV calling algorithm that uses a robust beta-binomial model with GC correction and a hidden Markov model (HMM) to combine likelihood across samples. This tool is suitable for pooled data, where changes in the number of copies are detected as deviations from reading counts concerning the average depth of coverage profile of a region. In other words, the model develops an aggregated reference dataset, which is optimised from a set of samples iteratively. At the exon level, it is provided with the read count ratio between a test sample and reference dataset and the likelihood of the read count data being a duplication or deletion in the test sample. These probability values are combined using HMM to call CNVs. Thus, *ExomeDepth* is best suited for datasets where the normalised read counts of samples are highly correlated, and for small and rare CNVs. However, its high sensitivity comes at the expense of reduced specificity.

qGenomics' custom CNV detection pipeline is based on the R package *ExomeDepth* and is already programmed so that it receives as input the Binary Alignment Map (BAM) and Browser Extensible Data (BED) files of all samples, and it outputs a text file with the CNVs identified. The programming language used is *R*, and some libraries required to execute it are *GenomicRanges* (Lawrence *et al.*, 2013), *IRanges* (Lawrence *et al.*, 2013) and, obviously, *ExomeDepth* (Plagnol, 2019).

2.2.2 CNV detection workflow

The proposed strategy, represented as a tree diagram in *Figure 1*, is based on a number of assumptions given the biology of the different situations that can take place. For example, since deletions involve loss of alleles, when considering heterozygous deletions of a diploid region, we expect all variants to become homozygous within the deleted regions. Thus, the presence of heterozygous variants is incompatible with any deletion, be it a homozygous or a heterozygous deletion. Following the same logic, if we have a hemizygous region or a homozygous deletion, we do not expect variants at all.

For each CNV, we first consider whether we face a deletion or a duplication, which is assessed by taking into account the observed versus expected reads, as computed by *ExomeDepth*. When deletion is being considered, we first need to assess whether the deletion is homozygous or heterozygous since, as we introduced above, our expectations on the type (homozygous, heterozygous or absent) of variants must be modulated based on this observation.

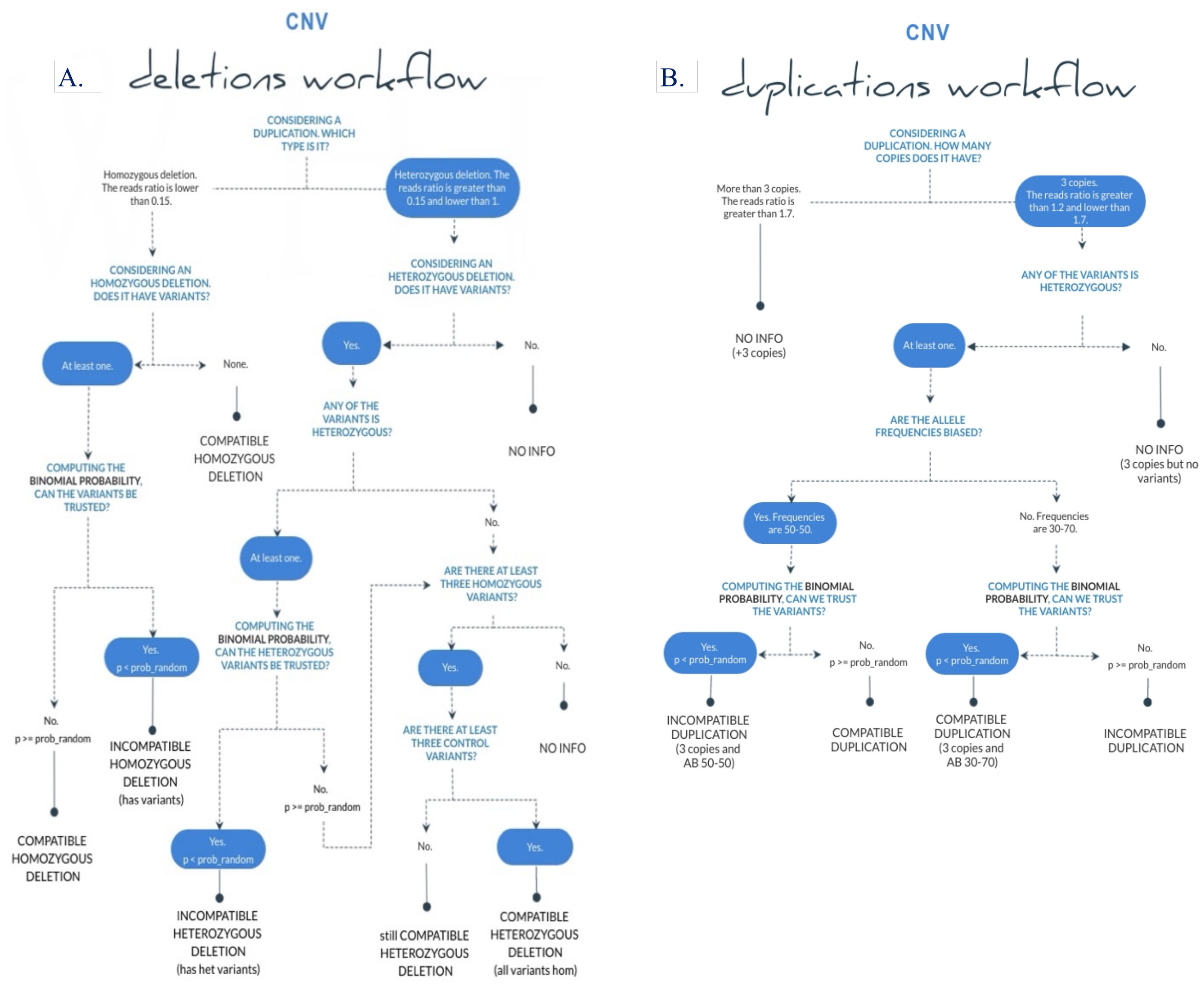

A.
CNV
deletions workflow
CONSIDERING A DUPLICATION. WHICH TYPE IS IT?
Homozygous deletion. The reads ratio is lower than 0.15.
Heterozygous deletion. The reads ratio is greater than 0.15 and lower than 1.
CONSIDERING AN HOMOZYGOUS DELETION. DOES IT HAVE VARIANTS?
CONSIDERING AN HETEROZYGOUS DELETION. DOES IT HAVE VARIANTS?
At least one.
None.
COMPATIBLE HOMOZYGOUS DELETION
Yes.
No.
NO INFO
ANY OF THE VARIANTS IS HETEROZYGOUS?
COMPUTING THE BINOMIAL PROBABILITY, CAN THE VARIANTS BE TRUSTED?
At least one.
No.
ARE THERE AT LEAST THREE HOMOZYGOUS VARIANTS?
COMPUTING THE BINOMIAL PROBABILITY, CAN THE HETEROZYGOUS VARIANTS BE TRUSTED?
Yes. p < prob_random
No. p >= prob_random
INCOMPATIBLE HOMOZYGOUS DELETION (has variants)
COMPATIBLE HOMOZYGOUS DELETION
Yes.
No.
ARE THERE AT LEAST THREE CONTROL VARIANTS?
NO INFO
No. p >= prob_random
Yes. p < prob_random
INCOMPATIBLE HETEROZYGOUS DELETION (has het variants)
No.
Yes.
still COMPATIBLE HETEROZYGOUS DELETION
COMPATIBLE HETEROZYGOUS DELETION (all variants hom)
B.
CNV
duplications workflow
CONSIDERING A DUPLICATION. HOW MANY COPIES DOES IT HAVE?
More than 3 copies. The reads ratio is greater than 1.7.
3 copies. The reads ratio is greater than 1.2 and lower than 1.7.
ANY OF THE VARIANTS IS HETEROZYGOUS?
NO INFO (+3 copies)
At least one.
No.
ARE THE ALLELE FREQUENCIES BIASED?
NO INFO (3 copies but no variants)
Yes. Frequencies are 50-50.
No. Frequencies are 30-70.
COMPUTING THE BINOMIAL PROBABILITY, CAN WE TRUST THE VARIANTS?
COMPUTING THE BINOMIAL PROBABILITY, CAN WE TRUST THE VARIANTS?
Yes. p < prob_random
No. p >= prob_random
Yes. p < prob_random
No. p >= prob_random
INCOMPATIBLE DUPLICATION (3 copies and AB 50-50)
COMPATIBLE DUPLICATION
COMPATIBLE DUPLICATION (3 copies and AB 30-70)
INCOMPATIBLE DUPLICATION

Figure 1. Copy number variants workflow*. Showing the CNV strategy for detecting accurate deletions (**a**) and duplications (**b**) predictions.*

When considering a *homozygous deletion*, we do expect the CNV not to have variants; if this is satisfied, we would be in front of a compatible homozygous deletion. Otherwise, we would compute the probability of those variants being bona fide variants or otherwise artefactual. This computation is performed using the error rate of the methodology (around 1%), the allele balance, and each variant's depth. If the detected variants are considered to be artefactual, the CNV would still be compatible; alternatively, the presence of bona fide variants would be incompatible with a homozygous deletion.

Having a *heterozygous deletion* implies that all the variants, if present, must be homozygous within the deleted region. We have considered that a minimum of three homozygous variants within the deletion is needed to have enough statistical support, as well as three control variants that we know have no CNV. Thanks to this, we observe how much our observation deviates from the distribution of the controls, and we can state if our CNV keeps being compatible with a heterozygous deletion. If we detect no variants, or not enough, we cannot report information about those CNVs. In case it has heterozygous variants, we compute the probability of those variants being heterozygous by chance. The theory declares that there cannot be heterozygous variants within a deletion, but the reading depth comes into play. If the depth is low enough, there may be sequencing read errors detected as heterozygous variants with a low allele balance (well below the expected 50%). Using the error probability, the number of alternatives and the depth, we can compute the probability of those heterozygous variants appearing at random, and with that, we can compute how much we can rely on the CNV prediction. It is crucial to notice that the relationship between the presence of heterozygous variants and the absence of deletion is quite powerful.

We need to look for incompatibility points for duplication cases since there is no biological presupposition related to variants at first glance. The proposed strategy is represented as a tree diagram in *Figure 1b*. If we have a duplication, we can only inform those having three copies and heterozygous variants within the region; otherwise, it would be compatible with a duplication. Having three copies and a heterozygous variant implies an increment of the allele balance of 30% approximately. Therefore, given this situation, we have to compute the probability of two distinct null hypotheses: having two copies and observing the altered allele balance by chance, or having three copies and observing at random an allele balance of 0.5. This will let us decide if we are in front of a compatible duplication or not. However, notice that most duplications lack significant clinical relevance, and we do not have enough information to make more specific detections. Therefore, although the strategy has been proposed and implemented, it will not be validated.

2.2.3 Prior considerations

As stated above, the overall strategy consists of taking advantage of biological knowledge to state how much we can rely on the CNVs predictions made by *ExomeDepth* using the information of small variants (SNV and INDELs) contained within the CNV. However, since NGS methodologies are not error-free, we must assess the possibility that any given variant is not a true variant (sequencing error) or does not belong to the region of interest (mapping error). In the following paragraphs, different situations that lead to the exclusion of SNV and INDEL variants are presented.

The first consideration taken has been that the variants coming from segmental duplication regions need to be excluded from the analysis. Segmental duplications are hard to map because the reads matching in that region can have recurring matches in other genome regions (Zeng *et al.*, 2015). Thus, we cannot provide information on any CNV falling entirely into a segmental duplication region. Therefore, only variants falling outside these regions should be considered when deciding whether the CNV call is true or not. Similarly, pseudogenes result in a limitation in detection and validation of the variants present in the associated genes (Torrents *et al.*, 2003); thus, they should also be filtered out.

We must also consider if an INDEL variant falls within a low complexity or a simple repeat region or is otherwise located in a unique genomic region. This is assessed by comparing the INDEL variants with the *RepeatMasker* database. It is crucial since INDELs mapping to tandem repeats or low complexity regions have a very high probability of being sequencing artefacts (Treangen and Salzberg, 2012) and therefore must be excluded. In addition, we have also filtered all those variants that not have not passed filters initially set during variant calling.

Another factor that must be taken into account is related to the pseudoautosomal regions (PARs) in sexual chromosomes since they are the only male sexual chromosome regions that can be heterozygous (Helena Mangs and Morris, 2007). The Y chromosome in the GRCh37 assembly contains two PARs that map to regions in the X chromosome: 10001-2649520 and 59034050-59363566 for chromosome Y, and 60001-2699520 and 154931044-155260560 for chromosome X. This denotes that if the CNV we are examining belongs to a chromosome X of a male and it does not fall within the PARs, we can only expect to find homozygous variants or no variants at all. If it falls within the PARs, we will consider it to behave the same way as having two copies.

All these data have been extracted from the UCSC genome browser (Karolchik *et al.*, 2004) database using the GRCh37 assembly.

2.2.4 Validation

A validation step is required to test the reliability of our approach. By comparing *ExomeDepth* calls to CNV calls from the same samples using an alternative methodology (DNA microarrays), we check if the predictions based on the presence/absence of variants can help filter out dubious *ExomeDepth* calls.

To perform the validation, we need to identify samples with a complete analysis of CNVs by arrays and the NGS test. Once identified, we can only validate those *ExomeDepth* CNV calls that are sufficiently covered in the arrays. This is, Exome Depth CNV call regions that are covered by at least three probes on the arrays, independent of the array resolution. At this point, the strategy proposed is applied to *ExomeDepth* CNVs, and the result is then compared to CNVs coming from arrays.

2.3 SNPs and INDELs classification

We take advantage of the most used libraries for Deep and Machine Learning throughout this project: *Keras* (v2.3.1) (Chollet, 2015) and *sklearn* (v0.24.2) (Pedregosa *et al.*, 2011), respectively. Furthermore, we likewise use *Tensorflow* (v1.14.0) (Abadi *et al.*, 2016) to make basic machine learning tasks with *Keras*. *Pandas* (v0.23.1) (McKinney, 2010), *NumPy* (v1.19.5) (Harris *et al.*, 2020), *Matplotlib* (v2.1.2) (Hunter, 2007) and *Scipy* (v1.5.4) (Virtanen *et al.*, 2020) packages are also used.

2.3.1 Algorithms

For the selection of the algorithms, three characteristics were taken into account: i) the need for a supervised learning algorithm since we have the data already labelled by specialists; ii) to be a multiclass classification algorithm; and iii) usage in the scientific community and methodological relevance. All algorithms selected —Decision Trees, Random Forest and Artificial Neural Networks— are supervised learning methods and can be used for multiclass data. Once the algorithms have been trained, they determine which label corresponds to new data by associating patterns with unlabelled new data. Next, the three methods used are explained in more detail.

2.3.1.1 Decision Trees

A Decision Tree (Quinlan, 1986) is characterised for building classification models in a tree structure. The dataset is broken into smaller subsets while simultaneously, an associated decision tree is incrementally developed. Entropy, which is the degree of uncertainty in the randomness of elements, and information gain, which quantifies the relative variation in entropy regarding the independent attribute, are used to construct a decision tree. Its weakness is overfitting, as it tries to fit the model by going deeper in the training set and thereby reducing test accuracy; nevertheless, overfitting can be minimised by applying pruning nodes.

2.3.1.2 Random Forest

Random Forest (Biau, 2012) is a combination of learning models based on bagging. In this algorithm, overfitting is prevented by creating trees on random subsets since it takes the average of all the predictions and thus cancels out the biases. Besides, while growing the tree, it searches for the best feature among a random subset of features, and this adds randomness to the model, resulting in a wide diversity that generally benefits the model.

2.3.1.3 Artificial Neural Networks

An Artificial Neural Network (Amato *et al.*, 2013) is a simulation of the neurons network that make up the human brain. Neurons can be modelled as mathematical functions represented as nodes, and their connections have weights associated with them. The weighted sum of each neuron input is calculated, and it is passed through the activation function, which is a nonlinear function, to produce an output. Hence, neurons translate inputs into a single output, which can then be picked up as input for another layer of neurons later on. Thus, there can be several hidden layers, and there are also called biased units, which are neurons that provide a value of 1.

2.3.2 Data summary

The performance of a model is directly proportional to the amount of data the model learns from. For this reason, we have gathered all data generated by *qGenomics* from October 2020 to May 2021, resulting in 121,425 interpreted variants from 2944 samples in 40 sequencing runs (average 41 interpreted variants per sample). Considering the raw dataset, the variant types we are interested in are classified as follows: 21.47% refer to *artefact*, 10.08% to *benign*, 4,74% to *pathogenic*, 25.00% to *polymorphism*, 16.67% to *vous*, 0.1% to *riskfactor,* and the 1,2% to *other*; the rest of variants will not be used either for training or testing. The cleaned and formatted dataset, after data preprocessing, is composed of 67506 variants and 86 features. Most of the features used to learn are frequencies from databases or directly related to the run: *FreqGnomAD, NHomHemGnomAD, RunCount, X1000G_ALL, ESP6500siv2_ALL, qExFreqInd, qGenFreq_536, qCarFreqInd, ExAcFreqObs, pLi* and *RunFreq*. Some others inform about remarkable prediction algorithms: *SIFT_pred,* *Polyphen2_HDIV_pred, Polyphen2_HVAR_pred, LRT_pred, MutationTaster_pred, FATHMM_pred, MutationAssessor_pred, RadialSVM_pred* and *LR_pred*. Many others are variant's characteristics like the *AB* (allele balance), *DP* (read depth), *Ref, Alt, State, Fun_refGene, ExonicFun_refGene, Clinvar_20170905, genomicSuperSups* and *Indel_length*. Finally, some others are more specific: *Procedencia, syn_z, mis_z, Num_qMales, Num_qFemales* and *Filter*.

2.3.3 Data preprocessing

Machine Learning models need the data cleaned and formatted, so the raw data cannot be directly used (Kotsiantis and Kanellopoulos, 2006). This preprocessing step includes handling missing values, feature encoding and feature selection (García, Luengo and Herrera, 2016).

To deal with missing values, we have performed a distinct procedure depending on each feature since a given general methodology can introduce biases in the dataset. In particular, for the read depth feature, using a specific value can adversely influence the rest, so we have better used the mean. Notice that this replacement has been performed after splitting the train and test to avoid leaking information. For allelic balances, since the 0 value corresponds to the homozygous alternative and the 1 to the homozygous reference, we have assigned a value of -1 for missing values. In categorical features, missing values are interpreted as a specific category. All these parameters have been selected for producing the optimum global accuracy after performing many combinations in a trial-and-error manner.

Dimension reduction has been performed by simplifying categorical features. Instead of using all specific INDELs annotated on *Alt* and *Ref*, we have assigned them the category *indel*, and a new feature containing its length has been created. In the case of *Clinvar*, the assignment has been done depending on the combination of categories they had and giving priority to the less relevant clinical ones, so the priority order is *uncertain significance* > *likely benign* > *benign* > *likely pathogenic* > *pathogenic* > *drug response*. For the segmental duplication feature, its genome position has not been considered, but the fact of overlapping one or not. The other features have been simplified with almost no effect on dimensionality reduction. Extra features informing about specific prediction algorithms, such as *SIFT* (Sort intolerant from tolerated), *Polyphen v2* (Polymorphism phenotyping v2), *LRT* (Likelihood ratio test), *MutationTaster* and *MutationAssessor*, have been added to make better predictions. In total, from almost 40,000 features we had initially, only 104 remain thanks to these considerations. A table showing the final categorical features is attached in *Supplementary Material Table S2.*

Identifying all features data types present in the dataset is needed to make the data proper for the algorithms concerning feature encoding. Machine learning algorithms cannot handle the text data directly; this is why the categorial features must be converted to numeric values. The technique used for encoding our nominal data is *Dummy coding*, which converts the categorical variable into a series of dichotomous variables. This technique maps the absence or presence of the diverse categories present in the feature using a vector consisting of 0 or 1. Notice that the algorithm learning rate can be reduced if the number of categories of data existing in a particular feature is too large; for this reason, features like the *Gene_refGene* have been avoided.

Feature selection is fundamental in machine learning since unnecessary features decrease generalisation performance on the test set and decrease training speed and model interpretability (Kotsiantis, 2014). Uninformative features regarding variant interpretation have been removed manually, like the *sample*, *chromosome*, *start*, *end,* and *gender*. We have also used feature importances for feature selection by removing low importance features according to the best learning model: Random Forest with specific hyperparameters. The features are averaged over three training runs in order to reduce variance. Features with zero importance - *ExonicFunc_refGene_ncRNA_exonic;splicing*, *Func_refGene_ncRNA_exonic;splicing*- have been directly removed since in a tree-based model, they are not used to split any nodes, and thus they are not affecting the model performance. In addition, the least important features not contributing to specified total importance are also removed. Only 87 out of the 132 features are required for 0.99 of cumulative importance in our training model. *LRT_pred_U*, *Filter_hard2validate* and *Ref_TT* are some of the low importance features removed. Normalised and cumulative importance of each feature using Random Forest is attached in *Supplementary Material Table S3*.

2.3.4 Model training

As it is common practice, our training dataset corresponds to 80% of the total information. Once the algorithm has been trained, it should determine which label corresponds to new data by associating patterns with unlabelled new data. An internal validation test will be performed to evaluate the performance of each of the models using the 20% leftover. Thus, we end up having 54004 variants in the training dataset and 13502 in the test dataset. In addition, an external validation test will be performed with the optimum performing model by using more sequencing runs never seen before by the algorithm.

2.3.5 Tuning hyperparameters and models

As a first approach, models have been trained with the *Scikit*-learn (Pedregosa *et al.*, 2011) sensible default hyperparameters. Then we have moved on to model hyperparameter tuning to guarantee that the used parameters are optimal for our application. Parameters optimisation has been performed by trial and error, so we need to evaluate different model performances to determine the optimal settings. Overfitting is prevented by performing many iterations of the entire K-Fold Cross Validation process, each time using different model settings (Berrar, 2018).

Secondly, we have evaluated model performances using *Scikit-Learn's RandomizedSearchCV*method (Bergstra and Bengio, 2012) with a wide range of values for each hyperparameter. The K-Fold CVs are performed with a combination of values taken randomly from the grid defined. We have adjusted the following hyperparameters regarding *Decision Trees*: *max_features*, *max_depth*, *criterion, and min_samples_leaf*; these regarding *Random Forest*: *n_estimators*, *max_features*, *max_depth*, *min_samples_split*, *min_samples_leaf*, *bootstrap,* and *criterion*; and these regarding *Artificial Neural Networks*: *batch_size*, *init*, *epochs*, and *optimisers*. The algorithm chooses between 8640, 432 and 90 combinations of settings for *DecisionTrees*, *RandomForest* and *Artificial Neural Networks at each iteration,* respectively. However, since the search is random, not every combination is tested. We have performed 100 iterations with three folds for cross-validation. Although this method can already improve accuracy, we can further improve the results by focusing on the most promising hyperparameters ranges found at this point (see *Table 1*).

From these results, the range of values for each hyperparameter is narrowed down. At this point, we use the *GridSearchCV* method (Siji George and Sumathi, 2020), which evaluates all combinations defined in a new grid based on an explicit specification of the settings provided by random search. So now, 18, 12 and 15 combinations of settings are evaluated.

To determine which method yielded a better model, we have compared the base model with the best random search model, which at the same time is compared with the grid search. The model with the optimum results will be selected to perform the training and the external validation test. Furthermore, an additional model using the parameter *sample_weight* assigned to *balanced* has been created for the Random Forest algorithm; it automatically adjusts the weights of the labels to class frequencies in the input data in an inversely proportional manner.

The model chosen will be tested using an additional external dataset with data never seen by the model.

All code developed for this project is available at GitHub link https://github.com/albamalagon/FinalDegreeProject.

3. Results and Discussion

As mentioned above, this project intends to have a substantial impact on the efficiency of genomic analysis procedures of clinical samples, focusing on two closely related strategies: the improvement in the detection specificity of copy number variants and the development and implementation of a machine learning tool for classifying single-nucleotide variants and short insertions/deletions.

Table 1: Model parameters*. Showing the settings used for the default, random and grid models.*

	DTD	*DTR*	*DTG*	*RFD*	*RFR*	*RFG*	*RFGW*	*ANND*	*ANNR*	*ANNG*
max_features	None	auto	auto	auto	auto	auto	auto			
max_depth	None	40	40	None	None	None	None			
min_samples_leaf	1	1	1	1	1	1	1			
criterion	gini	entropy	entropy	gini	gini	gini	gini			
min_samples_split	2	2	2	2	2	3	3			
n_estimators				100	1000	1200	1200			
bootstrap				True	False	False	False			
sample_weight				unlabeled	unlabeled	unlabeled	labeled			
optimiser								adam	adam	adam
batch_size								55	40	35
init								he_uniform	he_uniform	he_uniform

DTD: Decision Tree Default. DTR: Decision Tree Random. DTG: Decision Tree Grid. RFD: Random Forest Default. RFR: Random Forest Random. RFG: Random Forest Grid. RFGW: Random Forest Grid Weighted. ANND: Artificial Neural Network Default. ANNR: Artificial Neural Network Random. ANNG: Artificial Neural Network Grid.

Table 2: Validation results. *Showing the predictions our method makes for the concordant and discordant CNVs calls from ExomeDepth and arrays, as well as the segmental duplications and pseudogenes cases.*

	method outputs				
	compatible	incompatible	no info	superdup	TOTAL
Concordant call array & ED	4	0	7	0	11
Discordant call array & ED	4	17	74	0	95
Genomic superdups + pseudo	9	0	222	214	445

3.1 Copy number variants detection

This project explored the possibility of providing a more accurate CNVs filtering strategy by making predictions for each CNV. Each prediction considers the presence or absence of small variants and provides the probability that the detected variants are true variants and not sequencing artefacts. This calculation is made based on several variant's features, such as depth and allele balance. In the end, variant information is used to accept or reject CNVs given some biological expectations (see methods section). In addition, a validation has been performed, where CNVs detected by ExomeDepth have been compared to a reference dataset generated by microarray analysis of the same samples (*Table 2*).

The most remarkable result is that we cannot provide information about most CNVs by only using variants' information. This is because a large number of called CNVs are too small and do not contain small variants, or the variants have been filtered out (see methods). With this in mind, results are differentiated into two parts. On the one hand, concordant calls between *ExomeDepth* and arrays are expected to be predicted as compatible by our method. This happens for all CNVs from which we can extract information (n=4), which is a favourable result. However, compatibility cases should be taken with caution since the absence of evidence is not evidence of absence.

On the other hand, discordant calls between *ExomeDepth* and arrays (the call is made by Exome Depth but not using the arrays, and most likely are false positives) are expected to be predicted as incompatible by the method. In this case, for CNVs for which we have small variant information (n=21), we observe that the incompatible prediction is enriched with an 80%, but there are still some CNVs that are predicted as compatible. These compatible cases are homozygous deletions that do not contain variants and a heterozygous deletion containing more than three homozygous variants. As previously said, compatible cases are not that significant since they may lack conflicting evidence. Although the results are promising, note the small validation set performed.

Regarding the CNVs that *ExomeDepth* and arrays detected to be overlapping with segmental duplications or containing a pseudogene (n=445), our method has directly filtered half of them (n=214). However, the other half (n=231) has been analysed because they were overlapping partially with a segmental duplication, meaning that the variants falling outside could still be interpreted. Among these, only 9 CNVs turned out to be compatible with a deletion, and the rest do not contain enough small variant information to make predictions.

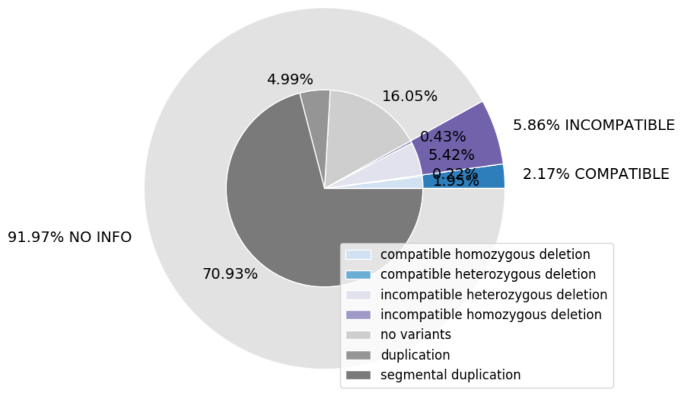


Figure 2: CNVs pie chart. *Showing the percentages of each type of CNV predictions, grouping them as compatible, incompatible and no info.*

After doing the project, we realised that the impact of using variant information for predicting the credibility of the CNVs is limited when considering all copy number variants generated by *ExomeDepth*. Most of them cannot be assessed, 91.97% to be precise, since they overlap with segmental duplications or contain a pseudogene (70.93%), do not contain variants (16.05%), or are duplication cases (4.99%). Since the idea is to reduce the time experts devote to interpreting the variants, getting rid of false-positive calls made by *ExomeDepth* is the key. Thus, we could get rid of 5.86% of the total variants, which at the end of the year turns out to impact the variant interpretation efficiency significantly.

3.2 SNPs and INDELs classification

3.2.1 Features information

Feature importances provide us insight into which features are helpful for the model when predicting the target variables. All scores obtained in this project can be found in *Supplementary Material Table S3*.

By inspecting *Figure 3*, we observe that the most important feature is *NHomHemGnomAD,* a rather intuitive finding. This tells us that the best predictor in classifying variants is the number of homozygous or hemizygous in the Genome Aggregation Database (gnomAD), which makes total sense because having homozygous or hemizygous in a general population is incompatible with severe disease. The following four features are also related to frequencies, also not that surprising. The second most important weighted feature is the *X1000G_ALL*, which uses the latest 1000 Genomes Project dataset with allele frequencies in six populations. The *RunFreq* indicates how many times the variant has been seen in the same run; since we are working with rare diseases, a variant that is found many times

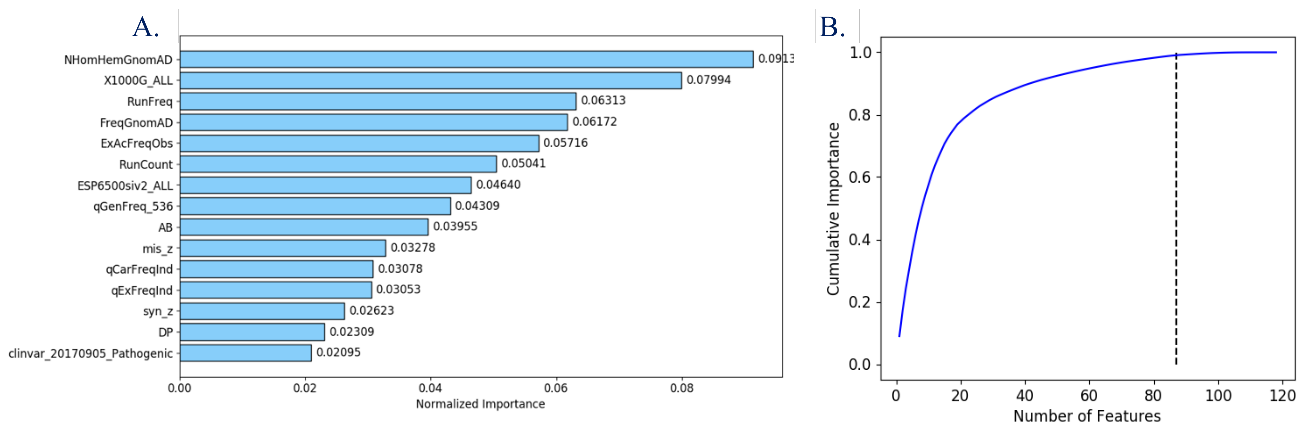


Figure 1: a) Top 15 feature importances. *Showing the 15 most essential features plotted in terms of normalised importance where the total sums to 1.* ***b) Cumulative feature importance****. Showing the cumulative importance versus the number of features. The vertical line is drawn at the threshold of the cumulative importance, in this case 99%.*

among the samples of the same run is most likely not pathogenic. The *RunCount* is similar to the *RunFreq*, one is a linear combination of the other, but the RunFreq is more important because it considers the run size. Other population frequencies appearing between the top 15 features are the *FreqGnomAD*, *ExAcFreqObs*, *ESP6500siv2_ALL*, *qGenFreq_536*, *qCarFreqInd* and *qExFreqInd*. The allele balance (*AB*) and the read depth (*DP*) are among the most important features. The features *mis_z* and *syn_z* come from GnomAD and give information about how much a gene tolerates missense and synonyms variants. The last feature *clinvar_20170905_pathogenic* relates to disease-specific variants.

As mentioned previously, only 87 out of the 132 features are required for 0.99 of cumulative importance in our training model (*Figure 2b*). The resulting optimal dataset contains all of the essential features that might bear our variant classification goal, leading to the best possible model outcomes.

3.2.2 Comparing model performances

To evaluate the model performances, we used the following measures: true positive, true negative, false positive and false negative, to compute measures like the precision score, the sensitivity, the F1-score, and finally, the accuracies and times of the training and test dataset. Accuracy and times' performances are shown in *Table 3*, whereas precision, recall and f1-scores of the benign class are detailed in *Figure 4*.

Training time comes as a factor during the implementation phase, and only Artificial Neural Networks are an outlier due to their complex computation, lasting ~15 times longer than Random Forest. It is an important metric since the training of the model would constantly be required with new data. The test time comes from the prediction phase, and Random Forest is investing the highest time, although it is not that significant.

Table 3: Accuracy and times' performances. *Showing accuracies and times for each of the models tested.*

	accuracy train	*accuracy test*	*training time*	*test time*
training decision tree default	99.91	94.56	0.9199	0.2813
training decision tree random	99.91	94.04	0.3642	0.3006
training decision tree grid	99.91	93.76	0.3626	0.2799
training random forest default	99.91	96.35	6.4315	1.6247
training random forest random	99.91	96.32	87.2785	18.0562
training random forest grid	99.91	96.40	101.6447	22.9277
training random forest grid weighted	99.87	96.44	110.5766	23.1817
training artificial neural network default	92.35	89.76	1194.4949	2.0067
training artificial neural network random	90.75	88.74	1680.1718	2.1313
training artificial neural network grid	91.08	88.07	1734.5624	2.2373
EXTERNAL RandomForest grid weighted 3runs		96		0.9835
EXTERNAL RandomForest grid weighted 1run		97		0.4372

Accuracy measures are shown in percentage, and times in seconds.

Since we want to reduce the time spent classifying variants, we have focused on the *benign* class as it shows promising performances (see *Figure 4*); *riskfactor* is not taken as a reference because of its small support (only representing 0.1% of the total variants). By looking at the precision score of the benign class, Random Forest is clearly performing the best (~0.98). Ensemble methods are often powerful due to their averaging method from multiple trees. Nevertheless, precision only talks about the positive class, and a high precision score can still perform poorly when we consider false negatives. The recall score is an essential metric since it reports the ability of the classifier to find all the positive samples. Random Forest also has the higher recall scores for the benign class; 0.99 to be precise. The higher the f1 score, the better the model is, and Random Forest keeps being the one with the higher scores (0.99), except for the default and weighted models (0.98).

Based on the analysis, we observe that Artificial Neural Networks are the worst models (92.35% and 89.76% accuracy for the training and test data, respectively). Although Decision Tree models are good, their weakness is overfitting, which reduces test accuracy. Finally, we have chosen a Random Forest model with the specific grid and weighted samples because it generates the highest recall, precision and f1-scores: 0.98,

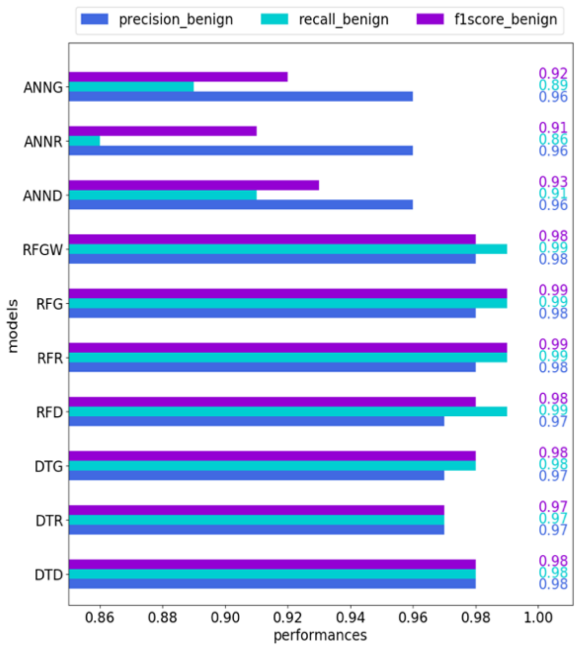


Figure 3: Bar plot with performances metrics of each model. *Notice the x limit starting at 0.85 to make visualisation differences among the outputs. DTD: Decision Tree Default. DTR: Decision Tree Random. DTG: Decision Tree Grid. RFD: Random Forest Default. RFR: Random Forest Random. RFG: Random Forest Grid. RFGW: Random Forest Grid Weighted. ANN: Artificial Neural Network Default. ANNR: Artificial Neural Network Random. ANNG: Artificial Neural Network Grid.*

0.99 and 0.98, respectively. In addition, it also has an excellent test accuracy (96.44%). External validation tests, with new data, have been performed with this model, showing satisfactory results. Accuracies of 96% and 97% have been obtained for the external validations using three and one runs, respectively. Thus, we feel confident that our model can predict variant's classification with 96% accuracy from six months of historical data.

Table 4: Classification report and predicted probabilities*. Showing the precision, recall and f1-score of each of the predictions, as well as their probabilities of being predicted as another class.*

predictions	precision	recall	f1score	artefact	benign	pathogenic	polymorphism	riskfactor	vous	other
artefact	0.94	0.95	0.94	93.62	0.0	2.15	3.23	0.0	0.58	0.41
benign	0.98	0.99	0.98	0.0	97.7	0.0	2.14	0.0	0.16	0.0
pathogenic	0.94	0.84	0.89	3.33	0.0	94.12	0.78	0.0	2.55	0.33
polymorphism	0.98	0.97	0.97	0.51	0.23	0.19	97.91	0.0	1.10	0.06
riskfactor	1.00	1.00	1.00	0.0	0.0	0.0	0.0	100.0	0.0	0.0
vous	0.95	0.98	0.97	0.34	0.15	3.17	0.98	0.0	95.34	0.02
other	0.96	0.91	0.93	0.81	0.0	0.81	2.44	0.0	0.0	95.93

Precision, recall and f1-scores values are between 0 and 1, whereas the rest are up to 100.

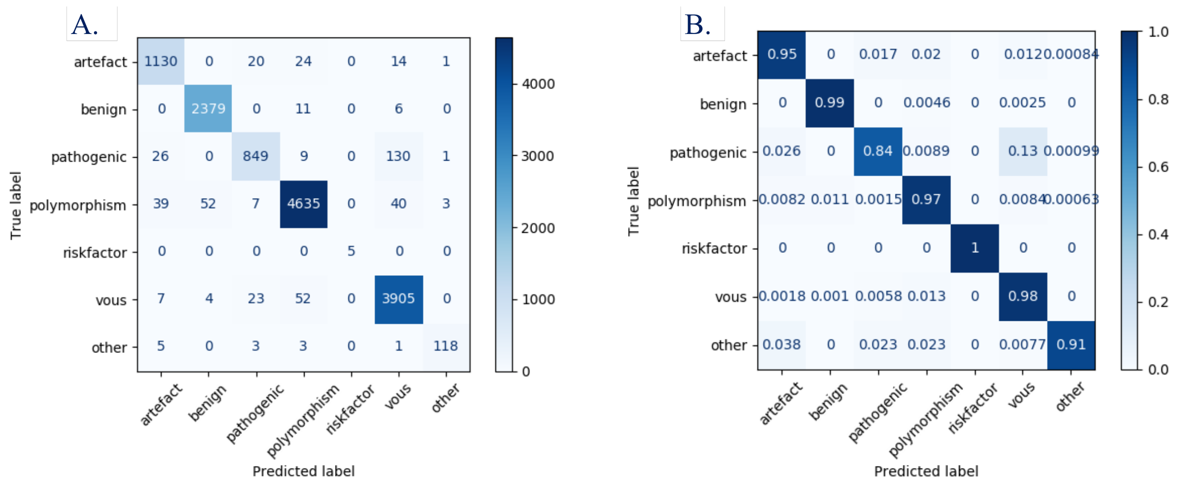


Figure 5: Confusion matrices*. Showing the confusion matrices of the Random Forest with a specific grid weighted.* ***a****) without normalisation.* ***b****) with normalisation.*

3.2.3 Outperforming model

The computational strategy chosen predicts the potential pathogenicity of SNP and INDEL variants with the optimum performing metrics possible. Of course, some labels may be not predicted in an outperforming way, but looking at the metrics, we will state which set of classes have enough probability to be trusted and which ones can be confused.

These confusion matrices (*Figure 5*) are used to evaluate the output of the best performing model: Random Forest with a specific grid. The diagonal values represent the number of times each class has been correctly predicted (Abhishek Sharma, 2018). The fewer numbers outside the diagonal indicate that the instances have been better classified. We have also normalised the data since our data is unbalanced, and it allows us to know what class is being wrongly classified.

We can observe that it ranks very well for riskfactor and benign classes, with scores of 0.99 and 1, but it has problems for the pathogenic class (0.84). Pathogenic and artefact are sometimes confused, which is expected since experts prefer not to consider quality filters when a variant shows pathogenic traits, and here the algorithm is taking the artefact signal. Vous, pathogenic and other classes are the most confused ones.

About 2% of the times we predict a variant to be benign, it is a real polymorphism, which is not a problem. In a small proportion, benign is also confused with the vous class, but it is never confused with the pathogenic one, which is an outstanding result. We can undoubtedly say that the *benign* and the *riskfactor* classes are well predicted or not confused with a variant class that risks lowering sensitivity. This means that they could be filtered directly, and interpreters would not need to analyse them. Both types of variants account for approximately 10.17% of the total variants interpreted, so the potential impact on the overall interpretation time may be significant. For the rest of the classes, we can simply report their probabilities being well predicted (see *Table 4*), which is unquestionably helpful for expert variants.

The Receiver Operating Characteristic (ROC) curve allows us to plot the true positive rate against the false-positive rate, which is the same as saying the sensitivity against the specificity, at various threshold values (Majnik and Bosnić, 2013). The Area Under the Curve (AUC) quantifies the ability of a classifier to distinguish between classes, and the higher the value, the better the ability to distinguish between positive and negative classes. In our model, which is plotted in *Figure 6*, almost all labels have AUC=1, except pathogenic and other labels, as expected.

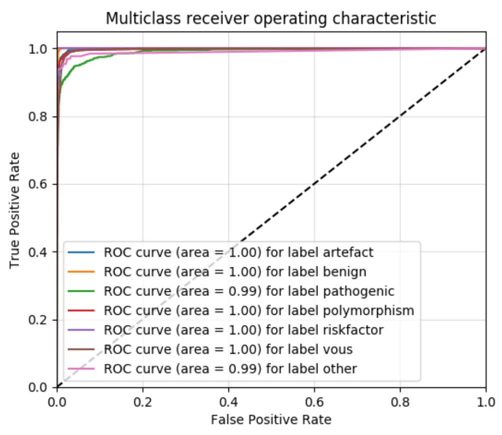


Figure 6: ROC curve*. Showing the multiclass ROC curve for each of the labels.*

4. Conclusions

Notwithstanding the widespread usage of NGS in genetic disease studies and diagnostics, the interpretation part for the identification of causal or disease-associated variants is still a time-consuming and non-scalable manual process. Furthermore, due to the vast quantity of variants that require human resources, it keeps being a huge challenge for clinical interpretation. Consequently, the process of interpretation requires to be optimised as much as possible. In essence, the two parts of the project are closely related as both aim to improve the efficiency of NGS processes in the interpretation part.

The first purpose of this project is to improve the detection of copy number variants since high false-positive rates strongly limit the existing CNV detection methods. The intention is to reduce the number of dubious calls that need to be interpreted and thus shorten the overall analysis time per sample. Noticing that we cannot provide information about most CNVs by only using variants' information and that compatibility cases should be taken with caution because of lacking evidence, we assume that the impact of this strategy is limited. However, we can still eliminate false-positive calls, representing 5.86% of the total variants, and thus shorten the overall analysis per sample. Consequently, provided with the Exome Depth predictions datasets, our method allows improving the detection specificity without coming at the expense of reduced sensitivity.

The second goal has to do with improving the efficiency in the tertiary analysis of NGS data through the implementation of machine learning tools. Noticing that our model can predict variant's classification with 96% accuracy from six months of historical data and that 10.17% of the total variants can be filtered out, predicting SNPs and INDELs variants considerably reduces the time spent by the experts interpreting them.

In general, all aims described above are directed to maximise the incredible value of efficiency to aid in the clinical interpretation of variants, leading to reduced diagnostic times and costs. The implementation of both methodologies, which have reduced the burden of downstream analysis and validation, is hugely relevant as many human resources need to be devoted to interpretation. Therefore, any help in this regard can substantially impact their efficiency, and they would be able to deploy their expertise where it is needed most.

5. Acknowledgements

I would like to express my deepest appreciation to my supervisor Jairo Rodriguez for the patience and excellent guidance throughout the project. I would also like to give my sincere thanks to Nacho Coca for the helpful discussions during the last month.

I am grateful to all members of the *qGenomics* company, especially to Neus Fornés and Olaya Villa, for providing me with information related to variant interpretation. In addition, I extend my thanks to Lluís Armengol and Xavi Armengol for helping me whenever I needed it.

6. Supplementary Material

Supplementary information accompanies this project at https://drive.google.com/file/d/1hGe_fesnoM03GpP98GZkktjgYncG3fZp/view?usp=sharing.

Figure S1. Confusion matrix of each model tested. **Figure S2**. Reduced Random Forest tree example. **Table S1**. Classification report of each model tested. **Table S2**. Specific categories of categorical features. **Table S3**. Normalised and cumulative importance of each feature using Random Forest.

Availability of data and materials

The datasets analysed during the current study are not publicly available as they are patient samples tested in a clinical diagnostic lab and sharing them could compromise research participant privacy.

7. References

Abadi, M. *et al.* (2016) 'TensorFlow: A system for large-scale machine learning', in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016*.

Abhishek Sharma (2018) 'Confusion Matrix in Machine Learning', *Www.Geeksforgeeks.Org*.

Abulí, A. *et al.* (2016) 'NGS-Based Assay for the Identification of Individuals Carrying Recessive Genetic Mutations in Reproductive Medicine', *Human Mutation*, 37(6). doi: 10.1002/humu.22989.

Amato, F. *et al.* (2013) 'Artificial neural networks in medical diagnosis', *Journal of Applied Biomedicine*. doi: 10.2478/v10136-012-0031-x.

Bergstra, J. and Bengio, Y. (2012) 'Random search for hyper-parameter optimization', *Journal of Machine Learning Research*, 13.

Berrar, D. (2018) 'Cross-validation', in *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*. doi: 10.1016/B978-0-12-809633-8.20349-X.

Biau, G. (2012) 'Analysis of a random forests model', *Journal of Machine Learning Research*.

Chollet, F. (2015) keras, GitHub. https://github.com/fchollet/keras

Cutting, G. R. (2014) 'Annotating DNA variants is the next major goal for human genetics', *American Journal of Human Genetics*. doi: 10.1016/j.ajhg.2013.12.008.

Eichler, E. E. (2008) 'Copy Number Variation and Human Disease', *Nature Education*, 1.

García, S., Luengo, J. and Herrera, F. (2016) 'Tutorial on practical tips of the most influential data preprocessing algorithms in data mining', *Knowledge-Based Systems*, 98. doi: 10.1016/j.knosys.2015.12.006.

Harris, C. R. *et al.* (2020) 'Array programming with NumPy', *Nature*. doi: 10.1038/s41586-020-2649-2.

Hunter, J. D. (2007) 'Matplotlib: A 2D graphics environment', *Computing in Science and Engineering*, 9(3). doi: 10.1109/MCSE.2007.55.

Kadalayil, L. *et al.* (2014) 'Exome sequence read depth methods for identifying copy number changes', *Briefings in Bioinformatics*, 16(3). doi: 10.1093/bib/bbu027.

Karolchik, D. *et al.* (2004) 'The UCSC table browser data retrieval tool', *Nucleic Acids Research*, 32(DATABASE ISS.). doi: 10.1093/nar/gkh103.

Kotsiantis, S. B. (2014) 'RETRACTED ARTICLE: Feature selection for machine learning classification problems: a recent overview', *Artificial Intelligence Review*, 42(1). doi: 10.1007/s10462-011-9230-1.

Kotsiantis, S. B. and Kanellopoulos, D. (2006) 'Data preprocessing for supervised leaning', *International Journal of …*, 1(2). doi: 10.1080/02331931003692557.

Lawrence, M. *et al.* (2013) 'Software for Computing and Annotating Genomic Ranges', *PLoS Computational Biology*, 9(8). doi: 10.1371/journal.pcbi.1003118.

Li, H. (2013) '[Heng Li - Compares BWA to other long read aligners like CUSHAW2] Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM', *arXiv preprint arXiv*.

Majnik, M. and Bosnić, Z. (2013) 'ROC analysis of classifiers in machine learning: A survey', *Intelligent Data Analysis*. doi: 10.3233/IDA-130592.

McKenna, A. *et al.* (2010) 'The genome analysis toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data', *Genome Research*, 20(9). doi: 10.1101/gr.107524.110.

McKinney, W. (2010) 'Data Structures for Statistical Computing in Python', in *Proceedings of the 9th Python in Science Conference*. doi: 10.25080/majora-92bf1922-00a.

Moreno-Cabrera, J. M. *et al.* (2020) 'Evaluation of CNV detection tools for NGS panel data in genetic diagnostics', *European Journal of Human Genetics*, 28(12). doi: 10.1038/s41431-020-0675-z.

Pandey, K. R. *et al.* (2012) 'The Curation of Genetic Variants: Difficulties and Possible Solutions', *Genomics, Proteomics and Bioinformatics*. doi: 10.1016/j.gpb.2012.06.006.

Pedregosa, F. *et al.* (2011) 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research*, 12.

Plagnol, V. *et al.* (2012) 'A robust model for read count data in exome sequencing experiments and implications for copy number variant calling', *Bioinformatics*, 28(21). doi: 10.1093/bioinformatics/bts526.

Plagnol V. ExomeDepth. (2019, December 31). https://cran.r-project.org/web/ packages/ExomeDepth/vignettes/ExomeDepth-vignette.pdf

Quinlan, J. R. (1986) 'Induction of Decision Trees', *Machine Learning*, 1(1). doi: 10.1023/A:1022643204877.

Richards, S. *et al.* (2015) 'Standards and guidelines for the interpretation of sequence variants: A joint consensus recommendation of the American College of Medical Genetics and Genomics and the Association for Molecular Pathology', *Genetics in Medicine*, 17(5). doi: 10.1038/gim.2015.30.

Siji George, C. G. and Sumathi, B. (2020) 'Grid search tuning of hyperparameters in random forest classifier for customer feedback sentiment prediction', *International Journal of Advanced Computer Science and Applications*, 11(9). doi: 10.14569/IJACSA.2020.0110920.

Torrents, D. *et al.* (2003) 'A genome-wide survey of human pseudogenes', *Genome Research*, 13(12). doi: 10.1101/gr.1455503.

Treangen, T. J. and Salzberg, S. L. (2012) 'Repetitive DNA and next-generation sequencing: Computational challenges and solutions', *Nature Reviews Genetics*. doi: 10.1038/nrg3117.

Virtanen, P. *et al.* (2020) 'SciPy 1.0: fundamental algorithms for scientific computing in Python', *Nature Methods*, 17(3). doi: 10.1038/s41592-019-0686-2.

Zare, F. *et al.* (2017) 'An evaluation of copy number variation detection tools for cancer using whole exome sequencing data', *BMC Bioinformatics*, 18(1). doi: 10.1186/s12859-017-1705-x.

Zeng, X. *et al.* (2015) 'Perm-seq: Mapping Protein-DNA Interactions in Segmental Duplication and Highly Repetitive Regions of Genomes with Prior-Enhanced Read Mapping', *PLoS Computational Biology*, 11(10). doi: 10.1371/journal.pcbi.1004491.

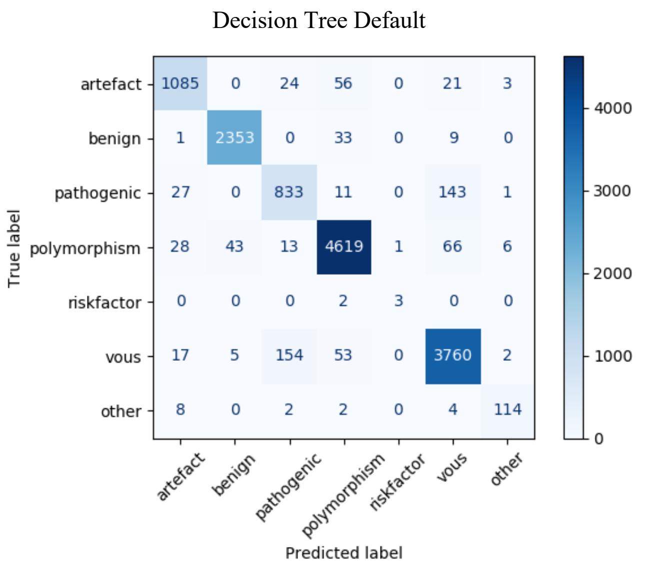
Zhang, J. *et al.* (2020) 'Clinical Interpretation of Sequence Variants', *Current Protocols in Human Genetics*, 106(1). doi: 10.1002/cphg.98.

STRATEGY FOR ACCURATE CNV DETECTION AND ML ALGORITHM FOR CLASSIFYING NGS VARIANTS

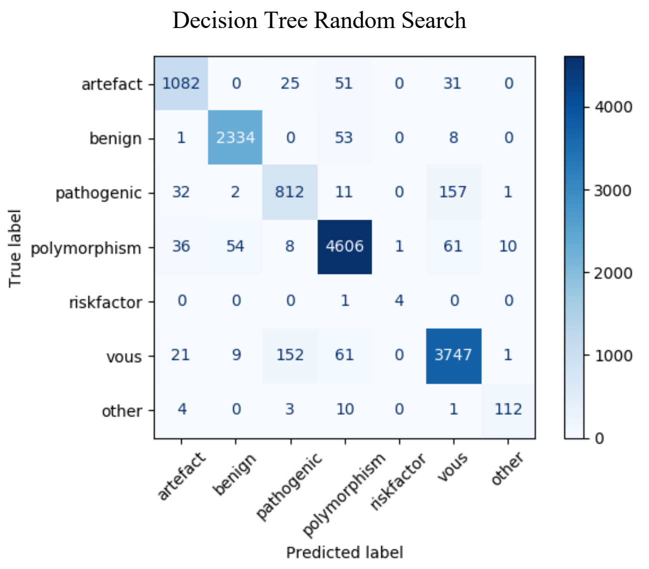
SUPPLEMENTARY MATERIAL

Alba Malagón Márquez

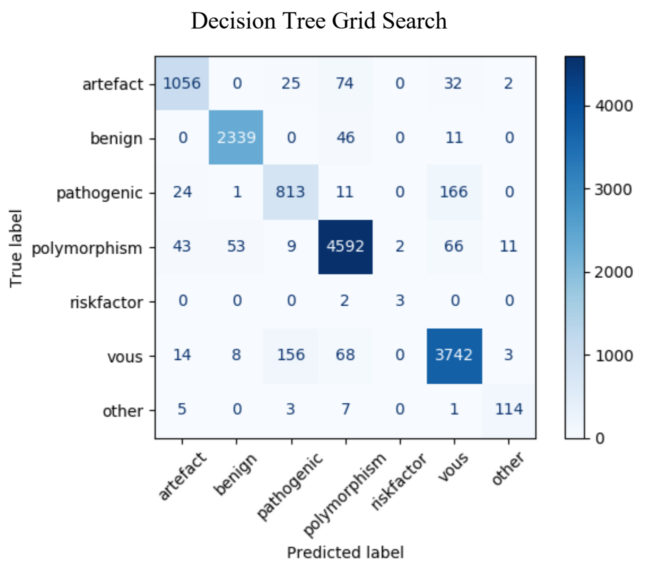
***Figure S1: Confusion matrices**. Showing the confusion matrix for each of the models tested.*



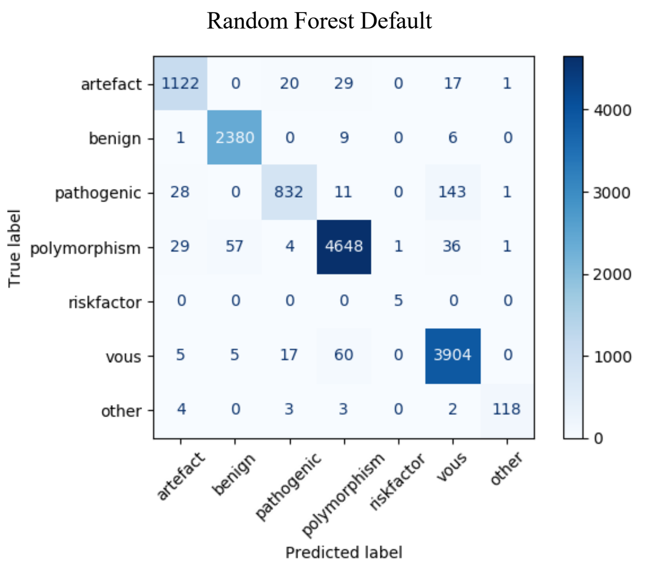
	artefact	benign	pathogenic	polymorphism	riskfactor	vous	other
artefact	1085	0	24	56	0	21	3
benign	1	2353	0	33	0	9	0
pathogenic	27	0	833	11	0	143	1
polymorphism	28	43	13	4619	1	66	6
riskfactor	0	0	0	2	3	0	0
vous	17	5	154	53	0	3760	2
other	8	0	2	2	0	4	114



	artefact	benign	pathogenic	polymorphism	riskfactor	vous	other
artefact	1082	0	25	51	0	31	0
benign	1	2334	0	53	0	8	0
pathogenic	32	2	812	11	0	157	1
polymorphism	36	54	8	4606	1	61	10
riskfactor	0	0	0	1	4	0	0
vous	21	9	152	61	0	3747	1
other	4	0	3	10	0	1	112

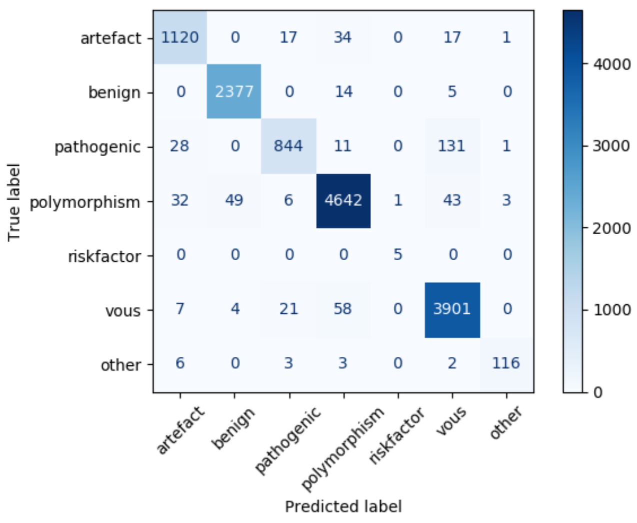


	artefact	benign	pathogenic	polymorphism	riskfactor	vous	other
artefact	1056	0	25	74	0	32	2
benign	0	2339	0	46	0	11	0
pathogenic	24	1	813	11	0	166	0
polymorphism	43	53	9	4592	2	66	11
riskfactor	0	0	0	2	3	0	0
vous	14	8	156	68	0	3742	3
other	5	0	3	7	0	1	114

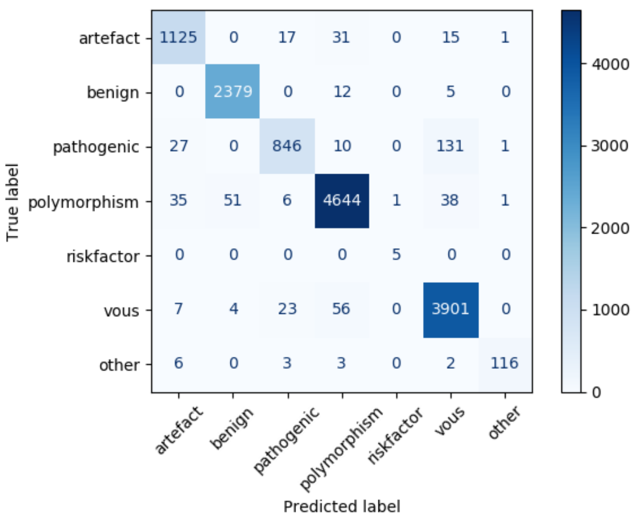


	artefact	benign	pathogenic	polymorphism	riskfactor	vous	other
artefact	1122	0	20	29	0	17	1
benign	1	2380	0	9	0	6	0
pathogenic	28	0	832	11	0	143	1
polymorphism	29	57	4	4648	1	36	1
riskfactor	0	0	0	0	5	0	0
vous	5	5	17	60	0	3904	0
other	4	0	3	3	0	2	118

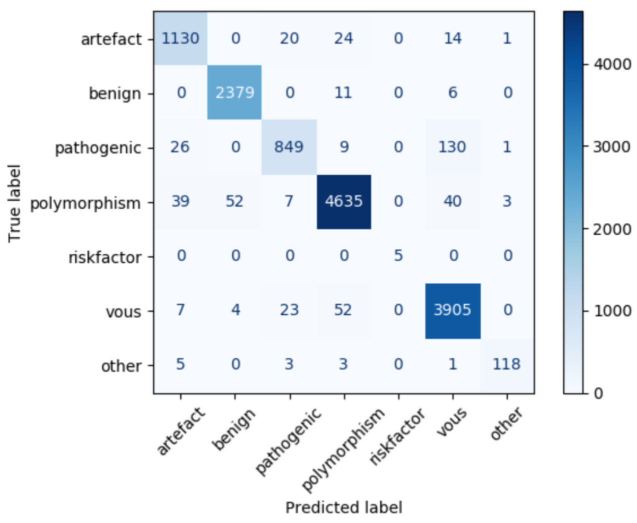
Random Forest Random Search



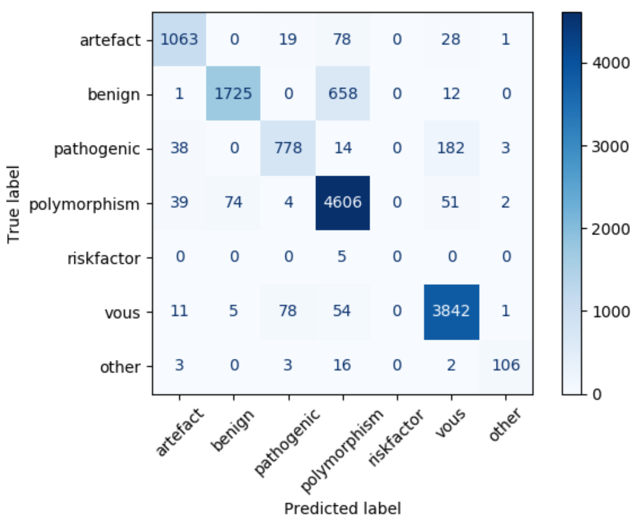
Random Forest Grid Search



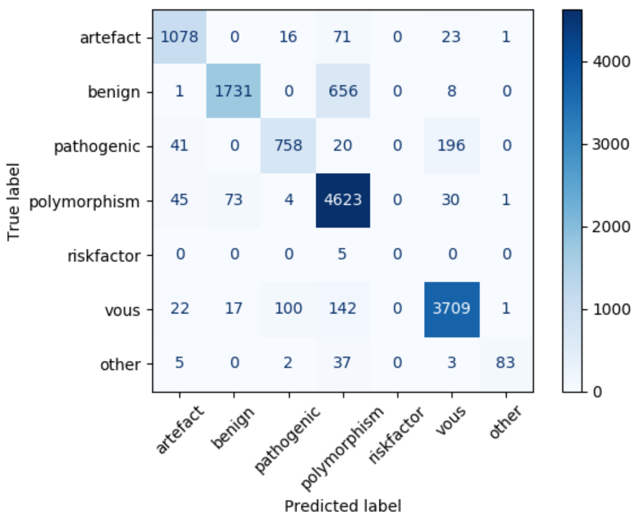
Random Forest Grid Search Weighted



Artificial Neural Network Default



Artificial Neural Network Random Search



Artificial Neural Network Grid Search

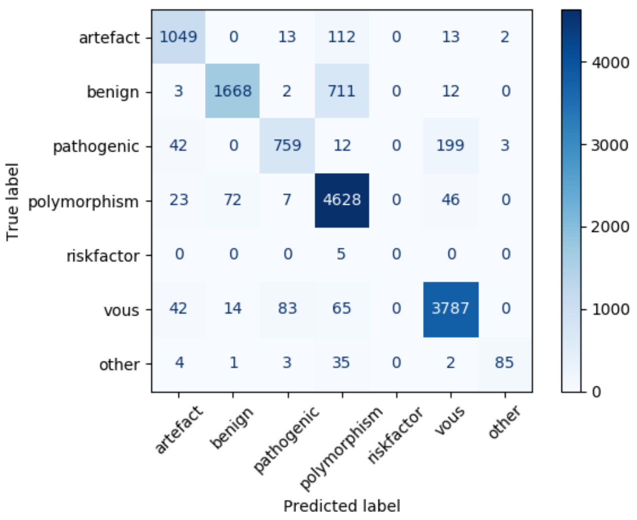


***Table S1: Classification report**. Showing the precision, recall and f1-score for each of the models tested.*

Decision Tree Default

	precision	recall	f1-score
artefact	0.93	0.91	0.92
benign	0.98	0.98	0.98
pathogenic	0.81	0.82	0.82
polymorphism	0.97	0.97	0.97
riskfactor	0.75	0.60	0.67
vous	0.94	0.94	0.94
other	0.90	0.88	0.89

Decision Tree Random Search

	precision	recall	f1-score
artefact	0.92	0.91	0.92
benign	0.97	0.97	0.97
pathogenic	0.81	0.80	0.81
polymorphism	0.96	0.96	0.96
riskfactor	0.80	0.80	0.80
vous	0.94	0.94	0.94
other	0.90	0.86	0.88

Decision Tree Grid Search

	precision	recall	f1-score
artefact	0.92	0.89	0.91
benign	0.97	0.98	0.98
pathogenic	0.81	0.80	0.80
polymorphism	0.96	0.96	0.96
riskfactor	0.60	0.60	0.60
vous	0.93	0.94	0.93
other	0.88	0.88	0.88

Random Forest Default

	precision	recall	f1-score
artefact	0.94	0.94	0.94
benign	0.97	0.99	0.98
pathogenic	0.95	0.82	0.88
polymorphism	0.98	0.97	0.97
riskfactor	0.83	1.00	0.91
vous	0.95	0.98	0.96
other	0.98	0.91	0.94

Random Forest Random Search

	precision	recall	f1-score
artefact	0.94	0.94	0.94
benign	0.98	0.99	0.99
pathogenic	0.95	0.83	0.89
polymorphism	0.97	0.97	0.97
riskfactor	0.83	1.00	0.91
vous	0.95	0.98	0.96
other	0.96	0.89	0.92

Random Forest Grid Search

	precision	recall	f1-score
artefact	0.94	0.95	0.94
benign	0.98	0.99	0.99
pathogenic	0.95	0.83	0.89
polymorphism	0.98	0.97	0.97
riskfactor	0.83	1.00	0.91
vous	0.95	0.98	0.97
other	0.97	0.89	0.93

Random Forest Grid Search Weighted

	precision	recall	f1-score
artefact	0.94	0.95	0.94
benign	0.98	0.99	0.98
pathogenic	0.94	0.84	0.89
polymorphism	0.98	0.97	0.97
riskfactor	1.00	1.00	1.00
vous	0.95	0.98	0.97
other	0.96	0.91	0.93

Artificial Neural Network Default

	precision	recall	f1-score
artefact	0.92	0.89	0.91
benign	0.96	0.72	0.82
pathogenic	0.88	0.77	0.82
polymorphism	0.85	0.96	0.90
riskfactor	0.00	0.00	0.00
vous	0.93	0.96	0.95
other	0.94	0.82	0.87

Artificial Neural Network Random Search

	precision	recall	f1-score
artefact	0.90	0.91	0.91
benign	0.95	0.72	0.82
pathogenic	0.86	0.75	0.80
polymorphism	0.83	0.97	0.90
riskfactor	0.00	0.00	0.00
vous	0.93	0.93	0.93
other	0.97	0.64	0.77

Artificial Neural Network Grid Search

	precision	recall	f1-score
artefact	0.90	0.88	0.89
benign	0.95	0.70	0.80
pathogenic	0.88	0.75	0.81
polymorphism	0.83	0.97	0.89
riskfactor	0.00	0.00	0.00
vous	0.93	0.95	0.94
other	0.94	0.65	0.77

EXTERNAL DATASET RANDOM FOREST GRID WEIGHTED 3 runs

	precision	recall	f1-score
artefact	0.94	0.90	0.92
benign	0.99	1.00	0.99
pathogenic	0.90	0.81	0.86
polymorphism	0.96	0.98	0.97
riskfactor	1.00	1.00	1.00
vous	0.96	0.97	0.96
other	0.97	0.94	0.95

EXTERNAL DATASET RANDOM FOREST GRID WEIGHTED 1 runs

	precision	recall	f1-score
artefact	0.96	0.97	0.96
benign	0.99	1.00	1.00
pathogenic	0.86	0.83	0.84
polymorphism	0.99	0.99	0.99
riskfactor	1.00	1.00	1.00
vous	0.97	0.96	0.97
other	1.00	1.00	1.00

Table S2: Categories of categorical features. *When coding the dataset, all these categories will be converted to features.*

Features	**Categories**
Ref	A, C, G, T, indel, other, missing, TT, HBA2
Alt	A, C, G, T, indel, other, TG10, TG11, TG13, TG12, missing, -0.0
State	hom, het, CHECK, unknown, missing
Filter	PASS, AB_0.2, LowQual, SnpCluster, End, hard2validate, missing
Fun_refGene	exonic, splicing, intronic, ncRNA_exonic, ncRNA_intronic, ncRNA_splicing, UTR3, missing, upstream, exonic;splicing, ncRNA_exonic;splicing
Exonic_refGene	nonsynonymous_SNV, splicing, frameshift_insertion, stoploss, frameshift_deletion, nonframeshift_insertion, duplication, deletion, nonframeshift_deletion, stopgain, PseudoGen_deletion, PseudoGen_duplication, c.121034TG(10)T(5), c.121034TG(11)T(5), c.121034TG(13)T(5), c.121034TG(12)T(5), missing, unknown, synonymous_SNV, nonframeshift_substitution, frameshift_substitution, exonic;splicing, ncRNA_splicing, ncRNA_exonic;splicing
Procedencia	No1en25, ClinVar_NoClinVar, NoCVar20160104, missing, SMN1file, TGfile, HBAbedFile
clinvar_20170905	Uncertain significance, missing, Pathogenic, Likely pathogenic, Benign, Likely benign, drug response
genomicSuperDups	yes, no, missing

Figure S2: Reduced Random Forest tree example. *Showing a reduced size tree annotated using the Random Forest algorithm with a maximum depth of three.*

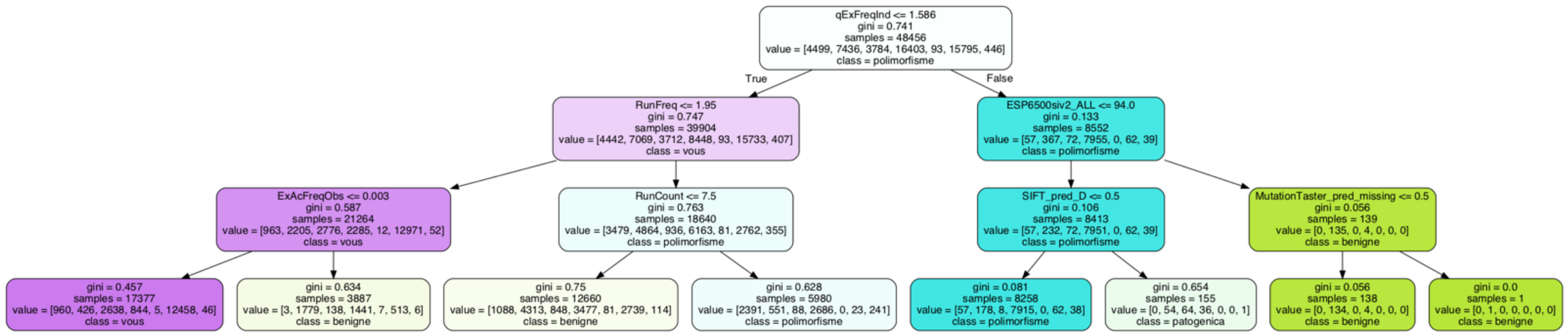


Table S3: Feature importances. *Showing all features sorted by their normalised importance in the Random Forest algorithm. Cumulative importances are also detailed.*

	feature	**normalized_importance**	**cumulative_importance**
0	NHomHemGnomAD	0.09134113797384272	0.09134113797384272
1	X1000G_ALL	0.07994372492869516	0.17128486290253786
2	RunFreq	0.06313407410908753	0.2344189370116254
3	FreqGnomAD	0.06172115834103226	0.2961400953526577
4	ExAcFreqObs	0.05716195301619706	0.3533020483688547
5	RunCount	0.05041107024534763	0.40371311861420234
6	ESP6500siv2_ALL	0.046396924681169516	0.45011004329537185
7	qGenFreq_536	0.04309091864641361	0.4932009619417855
8	AB	0.039547715090346665	0.5327486770321321
9	mis_z	0.03278368471804564	0.5655323617501777
10	qCarFreqInd	0.030780326695381912	0.5963126884455596
11	qExFreqInd	0.030525599019250248	0.6268382874648099
12	syn_z	0.026230704174174894	0.6530689916389848
13	DP	0.02308949967721985	0.6761584913162046
14	clinvar_20170905_Pathogenic	0.020946112396244245	0.6971046037124489
15	ExonicFunc_refGene_nonsynonymous_SNV	0.018532328752631663	0.7156369324650805
16	pLI	0.015239106441735943	0.7308760389068165
17	clinvar_20170905_Benign	0.013963883488371085	0.7448399223951876
18	clinvar_20170905_missing	0.011976180114793214	0.7568161025099808
19	indel_length	0.010895328657218447	0.7677114311671992
20	Num_qFemales	0.010112382674382343	0.7778238138415815
21	clinvar_20170905_Uncertain significance	0.009172413517310468	0.786996227358892
22	Alt_indel	0.009151728517626228	0.7961479558765182
23	Ref_indel	0.008739523398197457	0.8048874792747157
24	Filter_PASS	0.008399088393750273	0.813286567668466
25	Num_qMales	0.007599359599946924	0.8208859272684129
26	FATHMM_pred_T	0.007149926054200106	0.828035853322613
27	Procedencia_No1en25	0.006410394062546915	0.8344462473851599
28	clinvar_20170905_Likely pathogenic	0.005701269576511723	0.8401475169616717
29	ExonicFunc_refGene_stopgain	0.00539123184288078	0.8455387488045525
30	MutationTaster_pred_A	0.005092164831313157	0.8506309136358656
31	LRT_pred_missing	0.004865709412335823	0.8554966230482014
32	Alt_C	0.004860388153491554	0.860357011201693
33	Alt_A	0.004252758418116585	0.8646097696198095
34	Ref_G	0.0041465096263073875	0.8687562792461169
35	clinvar_20170905_Likely benign	0.003965019449003856	0.8727212986951207
36	Ref_T	0.0038687655272684664	0.8765900642223892
37	Alt_T	0.003868241812630797	0.88045830603502
38	LRT_pred_N	0.0037368941121854434	0.8841952001472054
39	Ref_C	0.0037199255537147128	0.8879151257009201
40	Func_refGene_exonic	0.0036875258828382718	0.8916026515837584
41	ExonicFunc_refGene_missing	0.003644540939991928	0.8952471925237503
42	State_het	0.0034079740463306106	0.8986551665700809
43	ExonicFunc_refGene_nonframeshift_insertion	0.0033737231785353704	0.9020288897486163
44	MutationTaster_pred_missing	0.0033093715130536492	0.9053382612616699
45	Ref_A	0.003066812263702331	0.9084050735253723
46	MutationTaster_pred_D	0.0030198374945313473	0.9114249110199036
47	Func_refGene_splicing	0.002985840548803356	0.914410751568707

48	Alt_G	0.0028612036455296667	0.9172719552142367
49	ExonicFunc_refGene_splicing	0.0027730504241293883	0.920045005638366
50	RadialSVM_pred_T	0.002685684664706631	0.9227306903030726
51	SIFT_pred_D	0.0026471213788558925	0.9253778116819286
52	LR_pred_T	0.002599371522612513	0.927977183204541
53	FATHMM_pred_D	0.0025367508393247364	0.9305139340438657
54	State_hom	0.0025100464996253733	0.9330239805434911
55	RadialSVM_pred_missing	0.002413338281580907	0.935437318825072
56	Procedencia_NoCVar20160104	0.0023229824263948807	0.9377603012514668
57	SIFT_pred_T	0.0023058310330382964	0.9400661322845051
58	SIFT_pred_missing	0.0022668567218998223	0.9423329890064049
59	Func_refGene_ncRNA_exonic	0.002263845563126047	0.944596834569531
60	LRT_pred_D	0.0022363999100112834	0.9468332344795423
61	MutationAssessor_pred_N	0.0021855557037003303	0.9490187901832426
62	MutationTaster_pred_P	0.0021707405354387116	0.9511895307186814
63	Polyphen2_HDIV_pred_missing	0.002121312604941732	0.9533108433236231
64	Polyphen2_HVAR_pred_missing	0.0020639845260214183	0.9553748278496446
65	LR_pred_missing	0.002047036380843021	0.9574218642304876
66	MutationTaster_pred_N	0.002026866721003519	0.9594487309514911
67	Func_refGene_intronic	0.0020085887589443084	0.9614573197104354
68	Polyphen2_HDIV_pred_B	0.0019947801667300753	0.9634520998771654
69	RadialSVM_pred_D	0.001991373550262656	0.9654434734274281
70	MutationAssessor_pred_missing	0.0019778164089047155	0.9674212898363328
71	Procedencia_ClinVar_NoClinVar	0.0019432557639044619	0.9693645456002372
72	MutationAssessor_pred_M	0.0018958042937399049	0.9712603498939771
73	Polyphen2_HVAR_pred_B	0.0018690626009057762	0.9731294124948829
74	LR_pred_D	0.0017591014731607986	0.9748885139680438
75	MutationAssessor_pred_L	0.001729343224547485	0.9766178571925912
76	ExonicFunc_refGene_frameshift_insertion	0.001627567730666048	0.9782454249232573
77	FATHMM_pred_missing	0.0015053417026634542	0.9797507666259208
78	Polyphen2_HVAR_pred_P	0.0014681737357211103	0.9812189403616418
79	Polyphen2_HDIV_pred_D	0.0013332803180695608	0.9825522206797114
80	Polyphen2_HVAR_pred_D	0.0012831050932293672	0.9838353257729407
81	Polyphen2_HDIV_pred_P	0.001227862688819194	0.98506318846176
82	Filter_AB_0.2	0.0012204987680547092	0.9862836872298147
83	Filter_LowQual	0.0012204265086319204	0.9875041137384467
84	Filter_missing	0.0011999279675084205	0.9887040417059552
85	genomicSuperDups_missing	0.0011786589922162692	0.9898827006981714
86	genomicSuperDups_yes	0.0009888672715610406	0.9908715679697324
87	State_unknown	0.0007782169663488785	0.9916497849360812
88	Alt_other	0.0006737184361911426	0.9923235033722724
89	Ref_other	0.0006714623572617238	0.992994965729534
90	LRT_pred_U	0.0006514857620189548	0.993646451491553
91	MutationAssessor_pred_H	0.0006134852452488762	0.9942599367368018
92	Comment2	0.0005853050381212396	0.9948452417749231
93	Func_refGene_UTR3	0.0005484655670428321	0.9953937073419659
94	State_CHECK	0.0005354415417221508	0.995929148883688
95	Ref_TT	0.0005047029795305648	0.9964338518632185
96	ExonicFunc_refGene_deletion	0.0005027036950650866	0.9969365555582836
97	Procedencia_TGfile	0.0004716501426100263	0.9974082057008937
98	Procedencia_HBAbedFile	0.0003543955578943688	0.9977626012587881
99	ExonicFunc_refGene_synonymous_SNV	0.0003517564415571008	0.9981143577003452

100	ExonicFunc_refGene_c.1210-34TG(10)T(5)	0.0003009854185376352	0.9984153431188828
101	ExonicFunc_refGene_unknown	0.00030000962780334155	0.9987153527466861
102	Alt_TG10	0.00028260939909427284	0.9989979621457804
103	Filter_SnpCluster	0.00022252989431957752	0.9992204920401
104	Alt_TG11	0.0001937056404030421	0.999414197680503
105	ExonicFunc_refGene_c.1210-34TG(11)T(5)	0.00016644029234887708	0.9995806379728519
106	Filter_End	0.00014757037774991641	0.9997282083506018
107	ExonicFunc_refGene_c.1210-34TG(12)T(5)	6.54592657005025e-05	0.9997936676163023
108	Alt_TG12	6.212782441856694e-05	0.9998557954407209
109	Filter_hard2validate	4.860291523981956e-05	0.9999043983559607
110	ExonicFunc_refGene_stoploss	2.708764055068281e-05	0.9999314859965114
111	ExonicFunc_refGene_nonframeshift_substitution	1.896783718583627e-05	0.9999504538336972
112	Func_refGene_exonic;splicing	1.6658266408535476e-05	0.9999671121001057
113	ExonicFunc_refGene_c.1210-34TG(13)T(5)	1.5679268395217496e-05	0.9999827913685009
114	Alt_TG13	1.294503376134259e-05	0.9999957364022622
115	Func_refGene_upstream	2.6854501334163275e-06	0.9999984218523956
116	ExonicFunc_refGene_frameshift_substitution	1.5781476039597342e-06	0.9999999999999996
117	Func_refGene_ncRNA_exonic;splicing	0.0	0.9999999999999996
118	ExonicFunc_refGene_ncRNA_exonic;splicing	0.0	0.9999999999999996