Using Machine Learning for predicting the effect of mutations in the initiation codon

J. Castell-Díaz, F. Abad-Navarro, M. E. de la Morena-Barrio, J. Corral, and J. T. Fernández-Breis

***Abstract*— The effect of mutations has been traditionally predicted by studying what may happen due to the substitution of one amino acid for another one. This approach may be effective for mutations with impact in the function of the protein, but ineffective for mutations in the translation initiation codon. Such mutation might avoid the generation of the protein. Consequently, specific methods for predicting the effect of mutations in the translation initiation codon are needed. We propose a method for predicting the effect of mutations in the canonical translation initiation codon based on a biological model that considers specific features of such mutations, like the distance to a potential alternative initiation codon. Our predictor has been developed using tree-based machine learning algorithms and data extracted from *Ensembl*. Our final model is able to detect whether a mutation in the canonical initiation codon is deleterious or benign with a precision of 44.28% and an accuracy of 98.32%, which improves the results of state of the art tools such as *PolyPhen*, *SIFT*, or *CADD* for this type of mutation.**

***Index Terms*— Bioinformatics, initiation codon, machine learning, mutation, prediction**

I. Introduction

DNA is in the nucleus of all cells and contains the sequence of information that is needed to create proteins. Those proteins are made outside the nucleus, in the ribosomes, so the information needs to be transported there by the RNA. This is done through a process called transcription, where the information needed is copied from the DNA to the RNA in the form of transcripts. The ribosome scans the sequence of the transcript until detecting the translation initiation codon AUG (ATG in the DNA sequence) in a favorable context. In this line, Kozak discovered a consensus sequence (also known as Kozak sequence) around AUG codons that empowers the recognition as translation initiation site by the ribosome [1].

This research is part of the grant PID2020-113723RB-C22 funded by MCIN/AEI/10.13039/501100011033/, and has also been possible possible thanks to the funding of the Instituto de Salud Carlos III and the European Regional Development Fund through grant PI18/00598 and of the Fundación Séneca, Agencia Regional de Ciencia y Tecnología through grant 19873/GERM/15.

J. Castell-Díaz, F. Abad-Navarro, and J. T. Fernández Breis are with the Departamento de Informática y Sistemas, Universidad de Murcia, CEIR Campus Mare Nostrum, IMIB-Arrixaca, Spain (e-mail: javier.castelld@um.es; francisco.abad@um.es; jfernand@um.es)

M. E. de la Morena-Barrio is with Servicio de Hematología y Oncología Médica, Hospital Universitario Morales Meseguer, Centro Regional de Hemodonación, Universidad de Murcia, CEIR Campus Mare Nostrum, IMIB-Arrixaca, Murcia, Spain (e-mail: mml13317@um.es)

J. Corral is with Centro de Investigación Biomédica en Red de Enfermedades Raras (CIBERER), Instituto de Salud Carlos III (ISCIII), Spain (e-mail: javier.corral@carm.es)

The translation initiation codon is the signal in the RNA sequence which triggers the translation of RNA into protein. Then, translation can start from that point to generate the protein.

The DNA may have errors which can be manifested through changes in the content of the nucleotide sequence. Such changes are called mutations, which can be due to a number of factors, and may affect interactions among proteins [2], [3] or the proteins' functions itself [4], [5].

In the last decades, different methods for predicting the effects of mutations have been proposed. *SIFT* [6] and *PolyPhen* [7] are likely to be the most popular tools, and their predictions mainly use the amino acid conservation levels and the effect that a substitution of one amino acid could have. Other popular techniques, such as *CADD*, use a score to determine the deleteriousness of a variant by combining different genomic features extracted from multiple annotations [8]. However, those methods are not optimal for mutations such as in the translation initiation codon. Mutations in this codon may cause the generation of a protein different from the intended one or the non-generation of any protein. Therefore, this kind of mutation is not necessarily related to the functions of the new protein that might be generated. Such kind of mutation may avoid the translation to protein, because there could be no starting site [9].

To the best of our knowledge, there are no specific predictors of the effect of mutations in the translation initiation codon, which are of particular relevance for the generation of proteins. A number of 26037 mutations affecting canonical start codons were identified in the *GRCh38* version of the human genome. This is why the main objective of this paper is the development of a predictor of the effect of mutations that affect the canonical translation initiation codon to classify them as either benign (not harmful) or deleterious (damaging). To this end, we will follow a machine learning approach, whose application to biomedical data has increased in the last years with different purpose, such as predicting protein-protein interactions [10], predicting S-nitrosylation sites [11], lethal interactions in cancers [12] or real-time heart diseases [13].

The main difficulty of this dataset comes from the imbalance found, where 98.5% of the mutations are classified as deleterious. This imbalance derives from the importance of the initiation codon in the creation of proteins, as there are few cases in which a mutation in this codon will be benign, since it is greatly affecting the resulting protein. This supports the need for such a tool, since by identifying the benign mutations in the initiation codon, we might be able to

understand the mechanisms that palliate the harmful effects of these mutations.

Our hypothesis is that if the canonical initiation codon is lost, the translation may begin in the next initiation codon found in an adequate context [9]. Therefore, we believe that aspects such as the distance to the alternative initiation codon from the canonical one or whether the original reading frame is kept should be taken into account when predicting the effect of mutations in the translation initiation codon. Larger distances or not keeping the reading frame increase the chances of having a deleterious mutation. From the technical perspective, our predictor will make use of machine learning algorithms, whose usage in bioinformatics has increased in the last years because of the availability of larger datasets [14], [15]. Each script used in this work can be found in the following *GitHub* repository: `https://github.com/JavierCastellD/InitiationCodonMutationPredictor`.

The remainder of this article is structured as follows. Section II explain the methods applied in this work to develop the predictor and the datasets used. We show the results of the predictor in Section III. The results are discussed in Section IV. Finally, some conclusions are put forward in Section V.

II. Materials and Methods

We propose a method based on machine learning techniques to obtain a model that is capable of classifying mutations in the initiation codon as benign or deleterious. Our method has three main steps (Figure 1):

- Configuration of the process, which mainly includes the selection of the variables and of the machine learning algorithm.
- Hyperparameter fine-tuning of the selected machine learning algorithms to prevent overfitting.
- Obtaining the final performance for our model through a held out test set.

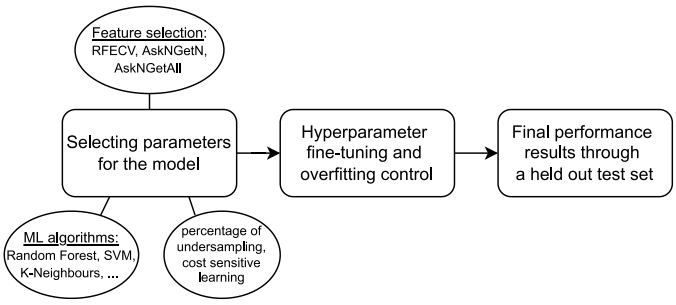

Fig. 1. The process followed to develop the predictor of the effect of mutations in the transcription initiation codon.

A. The dataset

The dataset used in this work [16] contains the 26037 mutations observed in the initiation codon in the human genome (*GRCh38* version), and 199 mutations from the goat genome (*ARS1* version). We used the goat genome to increase the number of instances of benign mutations. The goat data can be used in combination with the human one since the transcription and translation genetic mechanism are the same in both organisms. This necessity came from the extreme imbalance present in our dataset, as 98.5% of the instances (25815) belonged to the *DELETERIOUS* class, while the remaining 1.5% (421) corresponded to the *BENIGN* class. 36% of those 421 *BENIGN* instances came from the goat genome.

This dataset was obtained from *Ensembl* release 100 [17] and it contains the target feature CLASS and 13 features, six of which are categorical and, the rest, numerical. The distribution of each variable and its meaning can be seen in Tables I and II. There were no missing or null values for any feature of the dataset. Each mutation is classified as benign or deleterious by considering the minor allele frequency (MAF) obtained from *Ensembl*, assuming those with low frequency (MAF <0.01) to be deleterious [18]. There might be benign mutations that present low frequency due to being new, which would be wrongly classified as deleterious, but this approach has already been previously used as noted by Niroula and Vihinen [19] "*Variants with allele frequencies (AFs)* $\geq 1\%$ *are generally assumed to be benign, assumption widely used by e.g. predictor developers*", and Dong et al. [20] "*Higher prediction score from deleteriousness prediction tools indicates higher risk of deleteriousness, while higher MMAF means that the mutation is common and therefore is oftentimes less likely to be deleterious*", among other related work [21]–[24].

We used the results provided by Noderer et al. [25] for selecting the alternative initiation codon. In this work, the efficiency of each possible translation initiation site sequences spanning the -6 to +5 positions was provided, obtaining the Kozak context [1] an efficiency of 87. Here, we used this value as a threshold so that we select the first AUG codon in a context with an efficiency higher than 87 as alternative initiation codon.

At the beginning of our work, we split our dataset in two subsets: training and test. The training dataset contained 90% of the original data (23232 deleterious and 380 benign) and the test one had the remaining 10% (2583 deleterious and 41 benign). The split was done in a stratified manner, so each subset contains approximately the same percentage of samples of each target class as the complete set. The training subset was used to choose the parameters and fine-tune the hyperparameters of the model, while the test subset was only used at the end of the experiment to estimate the performance of the final model.

B. Selection of algorithms, variables and undersampling

The objective of this phase is to select some machine learning algorithms, as well as the combination of variables and percentage of undersampling to use.

When performing feature selection, the number of variables was determined by using a combination of five statistical tests: chi-square, ANOVA, mutual information and LASSO logistic regression with liblinear and saga. This was done in order to reduce bias, as selecting one over the other depends on the training set [26]. Our feature selection is based on the one used

TABLE I
CATEGORICAL FEATURES OF THE DATASET USED IN THIS WORK.

Feature name	Meaning
CLASS	Type of mutation.
CDS_COORDS	Coordinates of the mutation in relation to the coding region.
AMINOACID_CHANGE	Amino acid change due to the mutation.
CODON_CHANGE	Codon change due to the mutation.
READING_FRAME_STATUS	If the original reading frame was kept.
NO_STOP_CODON	Existence of a stop codon after the alternative initiation codon.
PREMATURE_STOP_CODON	Existence of a premature stop codon after the alternative initiation codon.

TABLE II
NUMERICAL FEATURES OF THE DATASET USED IN THIS WORK.

Feature name	Meaning
NMETS_5_UTR	Number of ATG in the 5' UTR of the mRNA.
CONSERVED_METS_IN_5_UTR	Number of ATG in the 5' UTR of the mRNA in phase with the original codon.
LOST_METS_IN_5_UTR	Number of ATG in the 5' UTR of the mRNA which are not in phase with the original codon.
CONSERVED_METS_NO_STOP_IN_5_UTR	Number of ATG in the 5' UTR of the mRNA in phase with the original codon without a stop codon.
MET_POSITION	Distance to the alternative initiation codon.
STOP_CODON_POSITION	Distance to the stop codon from the original initiation codon.
MUTATED_SEQUENCE_LENGTH	Length percentage of the resulting protein after the mutation in relation to the original.

in [27]. We developed two very similar techniques (Figure 2) that both required to ask for a number of features to each statistical test. In the first technique (*AskNGetN*), we executed the five tests, each test returned n features and we selected the n features with the highest frequency in the results offered by the five tests. For the second technique (*AskNGetAll*), we asked each test for n features and we selected all the features returned by at least one test. Additionally, we also used the implementation of *Scikit-learn* [28] of recursive feature elimination with cross-validation (RFECV) as a multivariate feature selection technique. To account for variance in model initialization and training, we applied repeated stratified cross-validation to the training set.

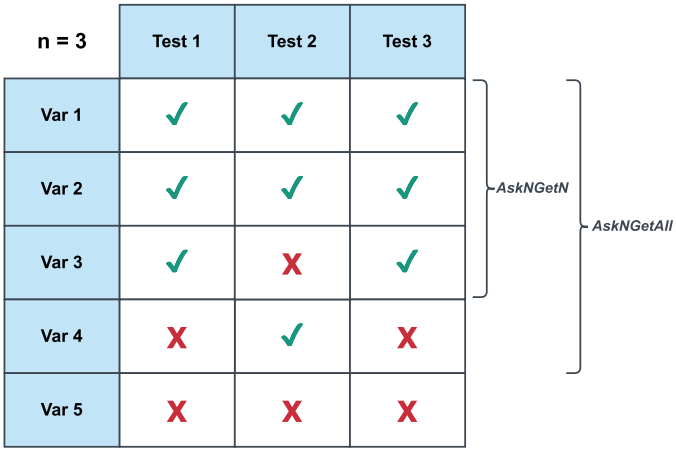

Fig. 2. Two feature selection techniques created for this work. For *AskNGetN*, each statistical test returns *n* features, and we keep the *n* most frequent. For *AskNGetAll*, each test also returns *n* features, but we keep each feature selected by at least one test.

Besides choosing the number of features, in this step we also selected the percentage of undersampling to apply, which affects to how much of the training set corresponds to the minority class *BENIGN*. We also studied the use of cost sensitive learning, in which the weight of the data is influenced by the class frequency. The following supervised machine learning algorithms were tested: *SVM* [29], *KNeighbours* [30], *AdaBoost* [31], *Gradient Boosting* [32], *GaussianNB* [33], *Gradient Descent* [34], *Decision Tree* [35], *Random Forest* [36], *Extremely Randomized Trees* [37] and *Bagging Classifier* [38]. These algorithms were used with the default hyperparameters according to the *Scikit-learn* library.

The output of this step was a group of different models selected by training a combination of machine learning algorithms, number of features, feature selection techniques, percentage of undersampling, and cost sensitive learning, and then choosing the most promising ones. In order to determine which models were the most promising, we focused on the F1-score using the BENIGN class as the positive class, as well as in the precision metric to reduce false positives, that is deleterious mutations misclassified as benign. The selection procedure consisted in ordering the possible models by their F1-score, removing those with a precision score of less than 0.45 and choosing the top three of each batch. We did this in four batches, each one being a combination of cost sensitive usage and feature selection technique.

C. Hyperparameter fine-tuning

The objective of this step is to fine-tune the hyperparameters of the models chosen in the previous section. For this, we selected different values for its hyperparameters and performed an exhaustive search. Every combination for the values we selected was tried with the objective of maximizing the F1 value.

For each combination, we performed stratified cross-validation to preserve the percentage of each class and to avoid training only with elements of the majority class. To prevent overfitting, which was likely to happen for the *BENIGN* class, we only chose those having a difference between training and test in the F1-score lower than 0.35. We also established a minimum of 0.4 for the precision score.

D. Comparison with SIFT, PolyPhen and CADD

To compare the performance of our method with state of the art methods such as *PolyPhen* [7], *SIFT* [6], or *CADD* [8], we used a different version of the whole dataset that only contained mutations in the translation initiation codon in the human genome, as *PolyPhen* and *CADD* are focused on human genome, and the list of genomes for *SIFT* does not contain the goat (*capra hircus*). The predictions of these models for the dataset were obtained from *Ensembl*.

To keep the variance under 1%, we repeated 1000 times the train/test split, so that our model was each time trained on the train set, and then every tool was evaluated on the test set. As we could only obtain *CADD* scores for around 50% of the dataset, we made sure that each instance of the test set had an available *CADD* score when performing the train and test split for the comparison.

III. Results

During the first phase of our experiment almost all machine learning algorithms were discarded but some of the tree-based ones: *Random Forest (RF)*, and *Bagging Classifier with Decision Trees (BCDT)*. Tree-based methods obtained the best results regardless of the use of cost sensitive learning, but only two of the feature selection techniques, *AskNGetAll* and *RFECV*, obtained a precision high enough to overcome the threshold of 45%. The results of this phase are shown in Tables III and IV, which include the top three performing models of each batch. An ID was assigned to each combination of parameters and machine learning algorithm, composed of the initials of that algorithm and a number.

TABLE III
PHASE 1 RESULTS FOR FEATURE SELECTION TECHNIQUE *AskNGetAll*. ML STANDS FOR THE MACHINE LEARNING ALGORITHM, *N_var* FOR THE NUMBER OF FEATURES ASKED TO EACH STATISTICAL TEST, *US* FOR PERCENTAGE OF UNDERSAMPLING, *CS* FOR USE OF COST SENSITIVE LEARNING, AND *Acc* AND *Prec* FOR ACCURACY AND PRECISION RESPECTIVELY.

ML	N_var	US	CS	Acc	F1	Prec
RF1	3	0.05	NO	0.984	0.343	0.539
RF2	4	0.05	NO	0.985	0.312	0.593
RF3	5	0.05	NO	0.985	0.290	0.611
RF4	3	0.05	YES	0.984	0.354	0.553
RF5	4	0.05	YES	0.985	0.325	0.635
BCDT1	4	0.05	YES	0.984	0.298	0.508

TABLE IV
PHASE 1 RESULTS OF COMBINATIONS OF UNDERSAMPLING AND COST SENSITIVE USAGE BY USING RECURSIVE FEATURE ELIMINATION.

ID	US	CS	Acc	F1	Prec
RF6	0.05	NO	0.983	0.372	0.451
RF7	0.05	YES	0.982	0.387	0.465

We applied random undersampling with the following values: [0.05, 0.1, 0.2, 0.3, 0.4, 0.5]. For the feature selection techniques *AskNGetN* and *AskNGetAll*, we set n to values between 2 to 5. As for the third feature selection technique, *RFECV*, the minimum number of features to be selected was set to 3.

The optimization of the hyperparameters was studied in the second phase. For the models chosen in the previous step, we fine-tuned the following hyperparameters: *max_depth*, which refers to the maximum depth of the tree; *min_samples_split*, which is the minimum number of samples to split a node; *min_samples_leaf*, which describes the minimum number of samples to be a leaf node; *n_estimators*, which indicates the number of trees in the forest; *bootstrap*, which refers to whether we are applying bootstrapping samples or not; and *max_features*, which is the number of features that are considered when looking for the best split.

For each of those hyperparameters we tested certain values: *max_depth* with values in [1, 2, 4, 8, 16, 32, 64, 128], *min_samples_split* with values in [2, 5, 10], *min_samples_leaf* with values in [1, 2, 4], *n_estimators* with values in [1, 2, 5, 10, 20, 30, 40, 50, 100, 200, 300, 400, 500, 750, 1000], *bootstrap* with values in [*True*, *False*], and *max_features* with values in [*None*, *sqrt*, *log2*]. We can see the results of the top performing models of this phase in Table V, being the model RF1 highlighted as it was chosen as the final model. Those results were obtained after performing a hundred training and test splits with cross-validation to reduce variance in order to make the best decision.

To obtain the definitive performance results for our final model, we used the whole training set for training and then the trained model was evaluated with the test set we had set apart at the beginning of the experiment. This was repeated a hundred times to reduce the variance of the results under 1%, but the content of the training and test sets were not changed. The results were 98.32 ± 0.02 for accuracy, 25.38 ± 0.84 for F1-score, and 44.28 ± 0.84 for precision, with a confidence interval of 95%.

Finally, we also compared the performance of our final model with that of *PolyPhen*, *SIFT*, and *CADD*. As we previously mentioned, we used a different version of our dataset for this comparison. That version only included mutations of the transcription initiation codon for the human genome, as our data for goat genome did not have the needed features to perform the prediction for *PolyPhen*, *SIFT*, and *CADD*. The results can be seen in Table VI with a confidence interval of 95%, after repeating the training and testing a thousand times.

We have made available a web form[1] so that it can be used by researchers to test our predictor. This web form has three different ways of performing the prediction: by features, by sequences, or by *Ensembl* transcripts.

For the prediction by features, the user has to input the value of each of the features of the final predictor: PREMATURE_STOP_CODON (if there is a premature stop codon after the alternative initiation codon), READING_FRAME_STATUS (whether the original reading frame was kept), MUTATED_SEQUENCE_LENGTH (length percentage of the mutated protein), LOST_METS_IN_5_UTR (number of ATG in the 5' UTR which are not in phase with the original codon), STOP_CODON_POSITION (distance to the stop codon) and MET_POSITION (distance to the alternative initiation codon). For the sequence one, the input has to be

[1] http://sele.inf.um.es/initiationMutationPredictor/

TABLE V
PHASE 2 RESULTS OF VALUES FOR THE HYPERPARAMETERS OF THE CHOSEN MODELS.

ID	max_depth	min_samples_split	min_samples_leaf	n_estimators	bootstrap	max_features	Acc	F1	Prec
RF1	**16**	**5**	**1**	**1000**	**False**	**sqrt**	**98.39 ± 0.03**	**26.62 ± 1.26**	**52.60 ± 2.66**
RF6	16	5	2	1000	False	log2	98.22 ± 0.03	24.25 ± 1.06	43.02 ± 2.26
RF2	16	5	2	5	False	log2	98.29 ± 0.03	17.77 ± 1.23	40.39 ± 2.56
RF3	16	2	2	50	False	sqrt	98.47 ± 0.02	16.93 ± 1.19	66.26 ± 2.98

TABLE VI
COMPARISON WITH *PolyPhen*, *SIFT*, AND *CADD* BY USING MUTATIONS IN THE INITIATION CODON FOUND IN HUMAN GENOME.

Method	Accuracy	F1	Precision	Recall
SIFT	81.99 ± 0.03	4.49 ± 0.03	2.36 ± 0.02	44.94 ± 0.35
PolyPhen	54.01 ± 0.03	3.11 ± 0.01	1.59 ± 0.01	78.40 ± 0.28
CADD	84.70 ± 0.03	7.48 ± 0.04	3.97 ± 0.02	65.66 ± 0.33
Ours	99.04 ± 0.01	21.28 ± 0.36	47.85 ± 0.86	13.91 ± 0.25

the complementary DNA sequence, the coding region and the mutated complementary DNA sequence. For the transcript, it is needed both the *Ensembl ID* and the codon change, which is the resulting codon of the mutation. An example of the web form and the first input option can be seen in figure 3. That figure shows the data input for the Features tab and the result of the prediction, which is calculated after clicking on the evaluate button. In this example, the mutation is predicted to be deleterious and the confidence of the prediction is 97.52%.

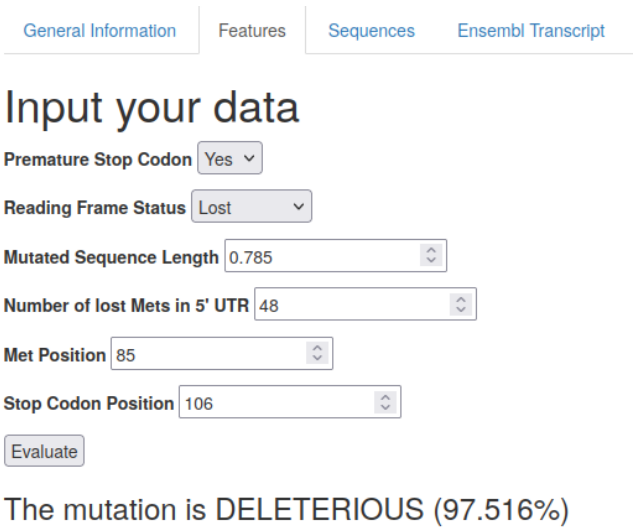

Fig. 3. Web form to perform predictions of mutations in the initiation codon with different input options: features, sequences, or *Ensembl* transcripts.

IV. DISCUSSION

In this paper, we have presented a predictor specific for mutations that affect the translation initiation codon. Our work applies the biological model described in [9], which showed the need of a specific treatment of this kind of mutation. Other general approaches for predicting the effect of mutations in the effect of the functions of the protein are not effective in this case. We have applied and tested different machine learning algorithms for the development of our predictor, which served to identify which ones were more appropriate for our study.

The main challenge of our work came from the extreme imbalance of our data (98.5% of the instances belonging to the *DELETERIOUS* class). This means that our predictor could have a high accuracy by classifying all mutations as deleterious. We applied different techniques to deal with such an imbalance. Aside from using undersampling, cost sensitive learning, and metrics such as precision or F1 to identify how well our model deals with the minority class, we also tested the use of oversampling (alone and merged with undersampling), and the use of data augmentation, with techniques such as SMOTE [39]. However, we considered that synthetic data was not adequate for this situation, as we could not guarantee that the generated data will be correct from a biological point of view. As for the use of oversampling, during preliminary testing, the results we obtained were not better than those of undersampling and we also considered problematic that, due to the low amount of instances of the minority class, our classifier might end up learning the same repeated instances, instead of identifying the differences between them. The usage of goat data was also intended to help reduce the imbalance of the dataset, as the number of benign instances increased after adding the available goat mutations for the initiation codon. Likewise, we used goat data instead of that of rats or more genetically known organisms because we could only find in *Ensembl* a large enough dataset of this kind of mutation for the goat.

For this experiment we had to decide the percentage of undersampling and the values for the number of features for the feature selection technique, as well as for the different hyperparameters of the machine learning algorithms. The undersampling percentage applied ranged from 5% to 50% in order to reduce imbalance. It is worth mentioning that the larger the percentage of undersampling, the more data is lost. Having a perfectly balanced dataset would require to lose more than 25000 instances of the *DELETERIOUS* class (about 97% of all the data). In the end, we found that an undersampling percentage of greater than 20% led to worse results for most of the machine learning algorithms we tried.

In order to decide what number of features to test, we previously performed an statistical analysis using chi-square and *ANOVA*, where we found that at least eight features had a significant relation to the target variable (p *value* <0.05), five of which had a p *value* lower than 0.01. After preliminary testing we established the range from 2 to 5. For *RFECV*, we chose a minimum of 3 features to be selected because we had to first apply several transformations in our dataset (such as using one hot encoding to categorical features and *min-max* scaling for numerical features) in order to perform *RFECV*, which meant that there would be more candidate features.

The results of the first phase revealed that only two of

the three feature selection techniques allowed for a model that could fulfill the threshold of precision: *AskNGetAll* and *RFECV*. Using *AskNGetN* led to a greater overall recall, at the expense of very low precision and lower than the others F1-score. The objective of using a threshold was to reduce the number of possible combinations of values for the parameters. Apart from acting as a filter mechanism, it also served as a way of trying to prevent the predictor of only focusing on the majority class by establishing a minimum accuracy for the minority class. We focused on precision to reduce the number of false positives, which are instances of the DELETERIOUS class predicted as BENIGN. The values of the threshold were chosen by analyzing our preliminary results. In this first phase, the training and testing with different subsets was only repeated fifty times. In the other two phases the experiment was repeated a hundred times to ensure low variance, as in this case there were a lot of possible combinations and the first step had a preliminary nature.

During the second phase, we noticed that there was overfitting only for the prediction of the minority class, with some instances where the difference between training and testing for recall or F1 was greater than 50%. We considered that this was happening due to the low number of instances of the BENIGN class and because of the way we performed the exhaustive search of the hyperparameters, as the implementation used (*GridSearchCV* from *scikit-learn* library [28]) applies cross-validation and created small samples sizes. We reduced the value for the thresholds established in the previous step looking at the results of each execution. We also found that models using cost-sensitive learning could not fulfill the threshold for precision after filtering out those suffering from overfitting.

Lastly, we compared the performance of our model with that of *PolyPhen*, *SIFT* and *CADD* for a certain dataset that only contained mutations in the transcription initiation codon of human genome. As mentioned in section II, we only had a *CADD* score for 50% of all the human instances, with only 117 of those being of the BENIGN class. As the *CADD* score only mattered when performing the evaluation comparison, and not when training the model, we opted for choosing the test set from among those with an available *CADD* score. The rest of the instances that were not part of the test set were considered as the training set. This particular split, and its corresponding training and evaluation, was repeated several times to account for variance in the undersampling and model training.

As shown in table VI, our model obtained better overall accuracy than the rest, although this metric is not the most important due to the skewed nature of the dataset towards deleterious mutations. As for recall, all other tools are able to obtain better results than ours, but that is at the cost of precision, and F1-score. That is why we consider our model to be preferable that theirs, since *SIFT*, *PolyPhen* and *CADD* obtain scores of 2.36%, 1.59% and 3.97% for precision respectively, while we achieve a precision of 47.85%. This means that they are suffering from false positives, where DELETERIOUS mutations are predicted as BENIGN, which is far more serious than the opposite (false negatives). That is because misclassifying BENIGN mutations as DELETERIOUS only leads to further tests that could detect the benign nature of the mutation. A false positive would mean ignoring a deleterious mutation. We tested our tool in a use case of a mutation that affects the initiation codon of *SERPINC1* [40], which is the gene encoding antithrombin. By classifying this mutation with our method, we would be able to correctly identify it as deleterious, which contrasts with the prediction given by other tools.

In the comparison with CADD results, we noticed that, for the human genome dataset, there were differences in the performance of our model between using stratified cross-validation or performing the special train/test split (see table VI). In the case of cross-validation, the recall of our model increased 20% (30.96% vs 13.91%) at the cost of a 10% decrease in precision (38.09% vs 47.85%), with an overall decrease in F1-score of 10% (33.65% vs 21.28%). The results for *SIFT* and *PolyPhen* were not affected by this change in the test set selection. This might mean that some of those instances with *CADD* score affected the model generalization capabilities, and not having them available for training, hindered its performance. It is also worth mentioning that the results obtained by performing stratified cross-validation on the dataset with no goat mutations showed an overall improvement compared to the final performance results shown in section III. This might mean that by adding goat genome data, we are actually adding unnecessary noise, or that the benign mutations added by goat genome made it harder for the model to generalise them. It is also possible that, due to the lower number of samples of the BENIGN class, our model was able to learn them better.

The features used in the final model were: MUTATED_SEQUENCE_LENGTH, PREMATURE_STOP_CODON, READING_FRAME_STATUS, LOST_METS_IN_5_UTR, STOP_CODON_POSITION, and MET_POSITION. During preliminary testing, we applied the statistical tests chi-square and ANOVA for the categorical and numerical features respectively to study the correlation with the target feature CLASS. Of those features chosen for the final model, only MET_POSITION had a *p-value* of less than 0.05, and the rest presented *p-values* of less than 0.001, except for STOP_CODON_POSITION (*p-value* of 0.0217). Other features that were not chosen and had an apparent statistical significance (*p-value* <0.05) were NMETS_5_UTR (number of ATGs in the 5' UTR), CODON_CHANGE (the change of codon due to the mutation), and CDS_COORDS (the coordinates of the mutation in relation to the coding region).

We also observed that, for our dataset, benign mutations usually have a closer alternative initiation codon than their deleterious counterparts, with a mean value of 146 vs 159 respectively for MET_POSITION. Additionally, benign mutations usually show greater MUTATED_SEQUENCE_LENGTH and with no values above 100%. Thus, benign mutations led to a resulting protein with a more similar length to the original than deleterious mutations. Finally, deleterious mutations are usually those that lose the original reading frame, since only 40% of them keep it.

As future work, having more data available, or at least a less

imbalanced dataset with more instances of the BENIGN class, may improve the quality of the predictor, as this has been one of the main challenges of this work. It might be interesting to try different techniques to deal with the imbalance such as a one class classifier [41] or other alternatives based on the use of neural networks and deep learning [42]–[44].

V. Conclusion

We were able to develop a predictor capable of detecting whether a mutation in the translation initiation codon is deleterious or benign by using a tree-based machine learning algorithm such as Random Forest. This model achieved during the testing process a final performance of 25.38% *F1-score*, an accuracy of 98.32%, and a precision of 44.28%. Due to the low number of instances of BENIGN mutations, the model classified some of them as DELETERIOUS. However, when comparing it with other state of the art techniques such as *SIFT*, *PolyPhen*, or *CADD*, our method outperformed them for this type of mutations. Looking at the results, we consider that our working hypothesis holds, since features such as the distance to the alternative initiation codon were chosen by the feature selection technique and they helped obtain better results than previous models.

References

[1] M. Kozak, "Point mutations define a sequence flanking the aug initiator codon that modulates translation by eukaryotic ribosomes." *Cell*, vol. 44, pp. 283–292, 1986, 10.1016/0092-8674(86)90762-2.

[2] R. Jones, M. Ruas, F. Gregory, S. Moulin, D. Delia, S. Manoukian, J. Rowe, S. Brookes, and G. Peters, "A cdkn2a mutation in familial melanoma that abrogates binding of p16ink4a to cdk4 but not cdk6." *Cancer Res.*, vol. 67, pp. 9134–9141, 2007, 10.1158/0008-5472.CAN-07-1528.

[3] M. U. Ung, B. Lu, and J. A. McCammon, "E230q mutation of the catalytic subunit of camp-dependent protein kinase affects local structure and the binding of peptide inhibitor." *Biopolymers.*, vol. 81, pp. 428–439, 2006, 10.1002/bip.20434.

[4] Y. Yamada, Y. Banno, H. Yoshida, R. Kikuchi, Y. Akao, T. Murate, and Y. Nozawa, "Catalytic inactivation of human phospholipase d2 by a naturally occurring gly901asp mutation." *Arch. Med. Res.*, vol. 37, pp. 696–699, 2006, 10.1016/j.arcmed.2006.01.006.

[5] O. Takamiya, M. Seta, K. Tanaka, and F. Ishida, "Human factor vii deficiency caused by s339c mutation located adjacent to the specificity pocket of the catalytic domain." *Clin. Lab. Haematol.*, vol. 24, pp. 233–238, 2002, 10.1046/j.1365-2257.2002.00449.x.

[6] P. Ng and S. Henikoff, "Sift: Predicting amino acid changes that affect protein function." *Nucleic Acids Res.*, vol. 31, p. 2003, 3812-4, 10.1093/nar/gkg509.

[7] I. A. Adzhubei, S. Schmidt, L. Peshkin, V. E. Ramensky, A. Gerasimova, P. Bork, A. S. Kondrashov, and S. R. Sunyaev, "A method and server for predicting damaging missense mutations." *Nat. Methods.*, vol. 7, pp. 248–249, 2010, 10.1038/nmeth0410-248.

[8] M. Kircher, D. M. Witten, P. Jain, B. J. O'Roak, G. M. Cooper, and J. Shendure, "A general framework for estimating the relative pathogenicity of human genetic variants," *Nature Genetics*, vol. 46, no. 3, pp. 310–315, Mar 2014. [Online]. Available: https://doi.org/10.1038/ng.2892

[9] F. Abad-Navarro, M. E. de la Morena-Barrio, J. T. Fernández-Breis, and J. Corral, "Lost in translation: Bioinformatic analysis of variations affecting the translation initiation codon in the human genome." *Bioinformatics*, vol. 34, pp. 3788–3794, 2018, 10.1093/bioinformatics/bty453.

[10] H. Lei, Y. Wen, Z. You, A. Elazab, E.-L. Tan, Y. Zhao, and B. Lei, "Protein–protein interactions prediction via multimodal deep polynomial network and regularized extreme learning machine," *IEEE journal of biomedical and health informatics*, vol. 23, no. 3, pp. 1290–1303, 2018.

[11] Q. Zhao, J. Ma, Y. Wang, F. Xie, Z. Lv, Y. Xu, H. Shi, and K. Han, "Mul-sno: A novel prediction tool for s-nitrosylation sites based on deep learning methods," *IEEE Journal of Biomedical and Health Informatics*, 2021.

[12] Z. Hao, D. Wu, Y. Fang, M. Wu, R. Cai, and X. Li, "Prediction of synthetic lethal interactions in human cancers using multi-view graph auto-encoder," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 10, pp. 4041–4051, 2021.

[13] D. Bertsimas, L. Mingardi, and B. Stellato, "Machine learning for real-time heart disease prediction," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 9, pp. 3627–3637, 2021.

[14] A. Pandurangan and T. Blundell, "Prediction of impacts of mutations on protein structure and interactions: Sdm, a statistical approach, and mcsm, using machine learning." *Protein Sci.*, vol. 29, pp. 247–257, 2019, 10.1002/pro.3774.

[15] Z. Shamsi, M. Chan, and D. Shukla, "Tlmutation: Predicting the effects of mutations using transfer learning." *J. Phys. Chem. B*, vol. 124, pp. 3845–3854, 2020, 10.1021/acs.jpcb.0c00197.

[16] F. Abad-Navarro and J. Castell-Díaz, "Mutations in the initiation codon of homo sapiens and capra hirucs genome." 2020, 10.5281/zenodo.3884050.

[17] K. L. Howe, P. Achuthan, J. Allen, J. Allen, J. Alvarez-Jarreta, M. R. Amode, I. M. Armean, A. G. Azov, R. Bennett, J. Bhai, K. Billis, S. Boddu, M. Charkhchi, C. Cummins, L. D. R. Fioretto, C. Davidson, K. Dodiya, B. E. Houdaigui, R. Fatima, A. Gall, C. G. Giron, T. Grego, C. Guijarro-Clarke, L. Haggerty, A. Hemrom, T. Hourlier, O. G. Izuogu, T. Juettemann, V. Kaikala, M. Kay, I. Lavidas, T. Le, D. Lemos, J. G. Martinez, J. C. Marugán, T. Maurel, A. C. McMahon, S. Mohanan, B. Moore, M. Muffato, D. N. Oheh, D. Paraschas, A. Parker, A. Parton, I. Prosovetskaia, M. P. Sakthivel, A. I. A. Salam, B. M. Schmitt, H. Schuilenburg, D. Sheppard, E. Steed, M. Szpak, M. Szuba, K. Taylor, A. Thormann, G. Threadgold, B. Walts, A. Winterbottom, M. Chakiachvili, A. Chaubal, N. D. Silva, B. Flint, A. Frankish, S. E. Hunt, G. R. IIsley, N. Langridge, J. E. Loveland, F. J. Martin, J. M. Mudge, J. Morales, E. Perry, M. Ruffier, J. Tate, D. Thybert, S. J. Trevanion, F. Cunningham, A. D. Yates, D. R. Zerbino, and P. Flicek, "Ensembl 2020," *Nucleic Acids Res.*, vol. 48, pp. 682–688, 2019, 10.1093/nar/gkaa942.

[18] S. Richards, N. Aziz, S. Bale, D. Bick, S. Das, J. Gastier-Foster, W. W. Grody, M. Hegde, E. Lyon, E. Spector, K. Voelkerding, H. L. Rehm, and ACMG Laboratory Quality Assurance Committee, "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," *Genet Med*, vol. 17, no. 5, pp. 405–424, Mar. 2015.

[19] A. Niroula and M. Vihinen, "How good are pathogenicity predictors in detecting benign variants?" *PLoS Comput Biol*, vol. 15, no. 2, p. e1006481, Feb. 2019.

[20] C. Dong, P. Wei, X. Jian, R. Gibbs, E. Boerwinkle, K. Wang, and X. Liu, "Comparison and integration of deleteriousness prediction methods for nonsynonymous SNVs in whole exome sequencing studies," *Human Molecular Genetics*, vol. 24, no. 8, pp. 2125–2137, 12 2014. [Online]. Available: https://doi.org/10.1093/hmg/ddu733

[21] K. Chennen, T. Weber, X. Lornage, A. Kress, J. Böhm, J. Thompson, J. Laporte, and O. Poch, "Mistic: A prediction tool to reveal disease-relevant deleterious missense variants." *PloS one*, vol. 15, 2020, 10.1371/journal.pone.0236962.

[22] I. Korvigo, A. Afanasyev, N. Romashchenko, and M. Skoblov, "Generalising better: Applying deep learning to integrate deleteriousness prediction scores for whole-exome SNV studies," *PLoS One*, vol. 13, no. 3, p. e0192829, Mar. 2018.

[23] N. M. Ioannidis, J. H. Rothstein, V. Pejaver, S. Middha, S. K. McDonnell, S. Baheti, A. Musolf, Q. Li, E. Holzinger, D. Karyadi, L. A. Cannon-Albright, C. C. Teerlink, J. L. Stanford, W. B. Isaacs, J. Xu, K. A. Cooney, E. M. Lange, J. Schleutker, J. D. Carpten, I. J. Powell, O. Cussenot, G. Cancel-Tassin, G. G. Giles, R. J. MacInnis, C. Maier, C.-L. Hsieh, F. Wiklund, W. J. Catalona, W. D. Foulkes, D. Mandal, R. A. Eeles, Z. Kote-Jarai, C. D. Bustamante, D. J. Schaid, T. Hastie, E. A. Ostrander, J. E. Bailey-Wilson, P. Radivojac, S. N. Thibodeau, A. S. Whittemore, and W. Sieh, "REVEL: An ensemble method for predicting the pathogenicity of rare missense variants," *Am J Hum Genet*, vol. 99, no. 4, pp. 877–885, Sep. 2016.

[24] M. F. Rogers, H. A. Shihab, M. Mort, D. N. Cooper, T. R. Gaunt, and C. Campbell, "FATHMM-XF: accurate prediction of pathogenic point mutations via extended features," *Bioinformatics*, vol. 34, no. 3, pp. 511–513, Feb. 2018.

[25] W. L. Noderer, R. J. Flockhart, A. Bhaduri, A. J. Diaz de Arce, J. Zhang, P. A. Khavari, and C. L. Wang, "Quantitative analysis of mammalian translation initiation sites by facs-seq." *Mol. Syst. Biol.*, vol. 10, 2014, 10.15252/msb.20145136.

[26] G. Aldehim and W. Wang, "Determining appropriate approaches for using data in feature selection." *Int. J. Mach. Learn. Cyb.*, vol. 8, pp. 915–928, 2017, 10.1007/s13042-015-0469-8.

[27] R. Dubey, J. Zhou, Y. Wang, P. M. Thompson, J. Ye, and Alzheimer's Disease Neuroimaging Initiative, "Analysis of sampling techniques for imbalanced data: An n = 648 adni study." *NeuroImage.*, vol. 87, pp. 220–241, 2014, 10.1016/j.neuroimage.2013.10.005.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[29] C. Cortes and V. Vapnik, "Support-vector networks." *Mach. Learn.*, vol. 20, pp. 273–297, 1995, 10.1007/BF00994018.

[30] E. Fix and J. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties." *Int. Stat. Rev.*, vol. 57, pp. 238–247, 1989.

[31] Y. Freund and R. Schapire, "A decision-theoretic generalization of online learning and an application to boosting." *J. Comput. Syst. Sci.*, vol. 55, pp. 119–139, 1997, 10.1006/jcss.1997.1504.

[32] J. Friedman, "Stochastic gradient boosting." *Comput. Stat. Data An.*, vol. 38, pp. 367–378, 2002, 10.1016/S0167-9473(01)00065-2.

[33] T. Chan, G. Golub, and R. LeVeque, "Updating formulae and a pairwise algorithm for computing sample variances." 1979, pp. 30–41, 10.1007/978-3-642-51461-6_3.

[34] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function." *Ann. Math. Statis.*, vol. 23, pp. 462–466, 1952.

[35] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees.* Wadsworth and Brooks, 1983.

[36] L. Breiman, "Random forests." *Mach. Learn.*, vol. 45, pp. 5–32, 2001, 10.1023/A:1010933404324.

[37] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees." *Mach. Learn.*, vol. 63, pp. 3–42, 2006, 10.1007/s10994-006-6226-1.

[38] L. Breiman, "Bagging predictors." *Mach. Learn.*, vol. 24, pp. 123–140, 1996, 10.1007/BF00058655.

[39] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique." *J. Artif. Int. Res.*, vol. 16, pp. 321–357, 2002, 10.5555/1622407.1622416.

[40] J. Navarro-Fernández, M. Eugenia de la Morena-Barrio, E. Martínez-Alonso, I. Dybedal, M. Toderici, N. Bohdan, A. Miñano, K. Heimdal, U. Abildgaard, J. Ángel Martínez-Menárguez, J. Corral, and V. Vicente, "Biochemical and cellular consequences of the antithrombin p.met1? mutation identified in a severe thrombophilic family," *Oncotarget*, vol. 9, no. 69, pp. 33 202–33 214, 2018. [Online]. Available: https://www.oncotarget.com/article/26059/

[41] X. Guo, Y. Yin, C. Dong, G. Yang, and G. Zhou, "On the class imbalance problem." *4th Int. Conf. Nat. Comput.*, vol. 4, pp. 192–201, 2008, 10.1109/ICNC.2008.871.

[42] N. Japkowicz, "Supervised versus unsupervised binary-learning by feedforward neural networks." *Mach. Learn.*, vol. 42, pp. 97–122, 2001, 10.1023/A:1007660820062.

[43] S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy, "Training deep neural networks on imbalanced data sets." in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 4368–4374, 10.1109/IJCNN.2016.7727770.

[44] D. Ravì, C. Wong, F. Deligianni, M. Berthelot, J. Andreu-Perez, B. Lo, and G.-Z. Yang, "Deep learning for health informatics," *IEEE journal of biomedical and health informatics*, vol. 21, no. 1, pp. 4–21, 2016.