Bayesian Network for Mutation Impact Prediction

Qiwen Xu
University of California, San Diego
La Jolla, CA 92093
qix007@ucsd.edu

Michael Kroyan
University of California, San Diego
La Jolla, CA 92093
mkroyan@ucsd.edu

Janice Rincon
University of California, San Diego
La Jolla, CA 92093
jrinconrodriguez@ucsd.edu

Vibusha Vadivel
University of California, San Diego
La Jolla, CA 92093
vvadivel@ucsd.edu

Jiya Makhija
University of California, San Diego
La Jolla, CA 92093
jmakhija@ucsd.edu

Abstract

Small, single-base variations in human DNA can have a range of consequences, ranging from completely benign to highly pathogenic. In this project, we use data from the ClinVar dataset to build a probabilistic model which predicts whether any given variant is classified as benign or pathogenic. Our engineered features include variant type (SNV, insertion, deletion, indel), frameshift (handling insertions and deletions that change the length by an amount that is not a multiple of three), stop-gain (whether a variant's protein annotation introduces a premature stop codon), length change category (whether a variant is a small or large insertion or deletion), chromosome type (X, Y, MT), chromosome region (low, mid, high), gene group (tumor suppressor, metabolism, and channel protein, structural, dna repair, other). After investigation, we discovered that frameshift, stop-gain, length change category, and chromosome region were deterministic features, and removed them for our final model. We discretize these features as random variable nodes in a Directed Acyclic Graph, learn the graph using HillClimbSearch from the pgmpy library, estimate its CPTs using MaximumLikelihoodEstimator from the pgmpy library, and then perform inference on these CPTs using the Variable Elimination algorithm. With a 80% training and 20% test split, the tuned Bayesian Network achieved 91.7% accuracy, 73.7% ROC AUC, and 37.6% precision on the held-out test set. This project demonstrates that it is possible to, with a limited set of features, predict whether or not a given variation in human DNA is more likely to be benign or pathogenic.

1 Problem description

Small single-base changes in DNA, known as *point mutations*, can have varying biological effects. While most mutations are harmless, others are capable of disrupting protein function or gene regulation which may lead to disease. Distinguishing benign mutations from pathogenic ones is a central challenge in genetics.

In this project, our goal is to build a **Bayesian Network (BN)** that predicts whether a mutation is pathogenic or not based on observable genomic features such as codon positioning, evolutionary conservation, GC content, and functional region.

This problem is important because whole-genome sequencing can reveal millions of variants in a single individual. Since only a small fraction can be experimentally tested, computational models must help prioritize which mutations are most likely to be pathogenic.

A probabilistic model, such as a Bayesian Network, is well-suited for this task because it can combine different kinds of evidence, handle uncertainty in each feature, and model how the features influence one another.

By predicting which mutations are more likely to be harmful, this model can help guide clinical decision making, focus research priorities, identify variants worth additional experimentation, and narrow down candidates for genetic research, ultimately accelerating the understanding of disease mechanisms.

2 Data sourcing and processing

2.1 Data Source

Our data comes from the ClinVar `variant_summary.txt.gz` file, which is publicly available from the NCBI FTP repository. ClinVar collects genetic variant information submitted by research groups and clinical laboratories. The dataset can be accessed at:

`https://ftp.ncbi.nlm.nih.gov/pub/clinvar/tab_delimited/`

Each row in this file describes a single genetic variant for either the GRCh37 or GRCh38 genome assembly.

2.2 Preprocessing

Before building the Bayesian Network, we applied several preprocessing steps to clean the dataset and create discrete, biologically meaningful features.

Selected relevant columns. We kept only the fields needed to construct our features (`Type`, `ReferenceAllele`, `AlternateAllele`, `Start`, `Stop`, and `ClinSigSimple`). Many other ClinVar fields (e.g., submitter details, IDs, review metadata) were not used because they do not directly describe biological effects or were too high-cardinality for a discrete BN.

Handled missing values. We removed rows missing key fields required for feature engineering (`Type`, `Chromosome`, `Start`, `Stop`, `GeneSymbol`). This removed 492,195 rows (5.89%), ensuring that all engineered features were well-defined.

Feature engineering. We created features that capture major biological mechanisms known to influence pathogenicity:

- **Variant type:** We used ClinVar's `Type` field (SNV, deletion, insertion, indel, etc.). Different variant types often have different functional impacts, so we kept this as a categorical feature.
- **Frameshift:** Insertions or deletions that change length by a number not divisible by three were marked as frameshift. This follows the standard coding rule (supported by Lin et al. 2017) and approximates frameshift effects since the dataset does not include full transcript context.
- **Stop-gain:** We identified variants introducing premature stop codons by detecting HGVS protein patterns such as `Ter` or `*`. This follows pLoF definitions used in Karczewski et al. (2020), where stop-gained variants are treated as strong loss-of-function signals. Because ClinVar does not directly label stop-gain events, this is an approximation.
- **Length change category:** We grouped variants by the size of inserted or deleted DNA: *SNV* (1 bp), *small_indel* (2–20 bp), and *large_indel* (>20 bp). These cutoffs reflect established observations that most natural indels fall within 1–20 bp (Lin et al. 2017), while larger

events behave more like structural variants. These bins let the BN capture strong size-related effects without relying on exact lengths.

- **Chromosome type:** We standardized chromosome labels into four categories: *autosome* (1–22), *sex_chr* (X/Y), *mitochondrial* (MT), and *unknown*. This reduces noise from inconsistent chromosome labels.
- **Chromosome region:** We divided the genomic start coordinates on each chromosome into three regions (*low*, *mid*, *high*) using `pd.qcut`, which creates equal-sized quantile bins. This provides coarse positional context without using raw genomic coordinates, which would produce large CPTs.
- **Gene group:** We mapped commonly studied genes into broad functional categories (e.g., *tumor_suppressor*, *structural*, *metabolism*, *channel_protein*, *dna_repair*) and labeled all others as *other*. This reduces thousands of unique gene symbols into a small, interpretable set usable by a BN.

Target label. We used `ClinSigSimple` as the target variable, where 0 represents benign-like variants and 1 represents pathogenic-like variants.

3 Modeling and inference

3.1 Probabilistic Model

Our probabilistic model is a **Bayesian Network**. We can define random variables in the context elaborated in **Data sourcing and preprocessing**.

3.2 Assumptions, Parameters, and Dependencies

- $Y := \text{ClinSigSimple\_num} \in \{0\ (\text{benign}),\ 1\ (\text{pathogenic})\}$
- $X_{\text{VariantType}} \in \{\text{SNV},\ \text{Deletion},\ \text{Indel},\ \text{CNV},\ \text{Other}\}$
- $X_{\text{LengthChange}} \in \{\text{SNV},\ \text{small\_indel},\ \text{large\_indel}\}$
- $X_{\text{Frameshift}} \in \{0\ (\text{no frameshift}),\ 1\ (\text{frameshift})\}$
- $X_{\text{StopGain}} \in \{0\ (\text{no stop-gain}),\ 1\ (\text{stop-gain})\}$
- $X_{\text{GeneGroup}} \in \{\text{tumor\_suppressor},\ \text{metabolism},\ \text{channel\_protein},\ \text{structural},\ \text{dna\_repair},\ \text{other}\}$
- $X_{\text{Chromosome}} \in \{\text{autosome},\ \text{sex\_chr},\ \text{mitochondrial},\ \text{unknown}\}$
- $X_{\text{PositionBin}} \in \{\text{low},\ \text{mid},\ \text{high}\}$

Each random variable is treated as a node in a DAG (Directed Acyclic Graph) with certain rules enumerated below.

- The label, $\mathbf{Y}$, can only have incoming edges.
- There are a maximum of **4** parents per node.

As well, we have to make one assumption.

- All data samples of each random variable are **i.i.d** distributed.

3.3 Learning and Inference

We split our processed dataset into two disjoint subsets, stratified across our categories: the training set used to learn the Bayesian Network and calculate the CPTs, and the test set used for final performance evaluation. This separation ensures that training does not use any information from the test set, providing an unbiased estimate of general performance.

For structure learning, we used the `HillClimbSearch` class from the `pgmpy` library, together with the Bayesian Information Criterion (BIC) score and `ExpertKnowledge` constraints. We chose Hill Climb Search because the search space of DAGs grows exponentially with the number of variables. Hill Climb Search is scalable because it performs a local greedy optimization over DAGs, starting from an initial graph and then repeatedly proposing local modifications (adding, deleting, or reversing a single edge). This makes it computationally feasible for our number of variables while still exploring a rich space of candidate structures. We wanted the learned structure to be driven directly by data while penalizing over-complex models. We used the BIC constraint so that over-complex models were penalized to encourage sparser and more interpretable networks.

We treated the maximum in-degree of each node (`max_indegree`) as a hyperparameter controlling model complexity. Higher in-degree allows more complex local dependencies but requires more data to estimate CPTs reliably.

Given the selected DAG structure, we estimated all CPTs using Maximum Likelihood Estimation via `pgmpy`'s `MaximumLikelihoodEstimator`. We did *not* implement MLE by hand. Instead, the library computes, for each discrete variable X_i, the conditional probability table directly from frequency counts in the training data `train_df`. These CPTs are then attached to the learned network structure.

For prediction, we used probabilistic inference via the `VariableElimination` class in `pgmpy`. Again, we did not code the Variable Elimination (VE) algorithm ourselves, we relied on the tested external library implementation and focused on applying it correctly to our learned BN.Given an instance (row) from the test set, we treat all non-label columns as observed evidence and query the posterior distribution over the target.

In summary, structure learning and parameter estimation are performed once on the training data and `pgmpy`'s Variable Elimination is then used repeatedly to perform exact inference for test variants.

4 Results and discussion

4.1 Model Performance

We identified that several downstream features (e.g. `StopGain`, `FrameShift`, `LengthChange_bin`, `PositionBin`) introduced deterministic relationships that leaked additional information to into the target label `ClinSigSimple_num`, we decided to remove these features as parents of ClinSig and retrained the Bayesian Network. This corrected an artificially inflated accuracy of ($\approx 93.2\%$) we observed previously in an earlier iteration.

With this leakage removed, the final Bayesian Network achieved a test accuracy of $\approx 91.70\%$ on a held-out test set composed of 7.87 million variants, representing 20% of our data.

However, because `ClinSigSimple_num` is extremely imbalanced, having 9.4% of data points labeled as *pathogenic* and 90.6% labeled *benign*, relying on accuracy alone is not meaningful. This is because a naive classifier that always predict *benign* would achieve a high accuracy score of $\approx 90.6\%$. Therefore, we will be evaluating our model on additional, but complimentary, metrics:

Metric	Value
Accuracy	0.9170
ROC AUC	0.7372
Average Precision (PR-AUC)	0.376
Precision (Class 1)	0.6240
Recall (Class 1)	0.3027
F1 (Class 1)	0.4076

Pathogenic variants make up only 9.4% of the dataset, therefore the baseline for PR-AUC is equal to this prevalence 0.094. Our Bayesian Network achieves a PR-AUC of 0.376, which is approximately 4 times higher than the baseline. This demonstrates the model's ability to capture meaningful biological signal despite the dataset imbalance and limited feature inputs. The model achieves 6.4% precision when predicting pathogenicity but recall is significantly lower at 30.3% for pathogenic variants, showing that many pathogenic variants remain unidentified.

4.2 Network Structure and Dependencies

We learned the network structure using the Hill Climb Search algorithm with BIC scoring and enforced constraints in order to prevent `ClinSigSimple_num` from receiving deterministic information from downstream effect features. Under these constraints and a maximum of four parents per node, the structure learning algorithm produced a compact, interpretable DAG consisting of six edges:

- VariantType $\rightarrow$ ClinSigSimple_num
- GeneGroup $\rightarrow$ ClinSigSimple_num
- Chromosome_clean $\rightarrow$ ClinSigSimple_num
- VariantType $\rightarrow$ GeneGroup
- VariantType $\rightarrow$ Chromosome_clean
- Chromosome_clean $\rightarrow$ GeneGroup

Several meaningful dependency patterns emerged:

- **ClinSigSimple_num has three direct parents**: These parents include `VariantType`, `GeneGroup`, and `Chromosome_clean`. These features capture broad genomic characteristics rather than post-mutation consequences, which suggests the model's ability to predict pathogenicity relies on mutation type, chromosomal context, and gene-level attributes .
- **VariantType is a primary explanatory variable:** While variant type is is not a causal determinant for a gene's category or its chromosomal location, the learned edges of the DAG indicate some mutation types occur at disproportional rates within certain gene groups.
- **Chromosome_clean also influences GeneGroup**: This dependency captures patterns in the dataset, such as gene families or functional groups being more common on particular chromosomes.

4.3 Learned Conditional Probability Tables

Examining selected CPTs reveals biologically meaningful patterns:

- **VariantType distribution**: The marginal distribution of VariantType is highly skewed, given that SNVs account for approximately 91.7% of all variants, with deletions and small indels making up the majority of the remainder. This reflects both biological reality and ascertainment bias in variant databases.
- **GeneGroup given VariantType and Chromosome_clean**: The model assigns 99% probability to VariantType=SNV and GeneGroup="Other". For tumor suppressor genes, they appear at a very low probability of 0.9%. This distribution is primarily caused by the pronounced class imbalance in the dataset, where most genes are categorized as "other" and SNVs are overrepresented in such genes.
- **Chromosome_clean given VariantType**: When conditioned on VariantType = SNV, the CPT shows that the vast majority of all SNVs occur on autosomes ($\approx 96\%$) , with only a small fraction occurring on sex chromosomes ($\approx 3.5\%$) and very few occurring on mitochondrial DNA. This is expected as it reflects the underlying dataset where most genes, and therefore most variants, are located on autosomes.

4.4 Model Limitations and Scalability

Convergence and Stability. After removing downstream functional-impact features (StopGain, Frameshift, PositionBin, and LengthChange_bin) to prevent target leakage, both structure learning and parameter estimation behaved reliably.The Hill Climb Search converged successfully using BIC scoring. MLE-based CPT learning is deterministic and stable given the large sample size (6.3 million training instances). No convergence issues were encountered.

Computational Cost. Structure learning took approximately 118 seconds on the training set. Inference via Variable Elimination on the test set was efficient despite the dataset size. The model scales linearly with the number of instances for parameter learning, though structure learning complexity depends on the search space and scoring metric.

Approximation Limitations. Several biologically informative annotations are absent from the ClinVar summary dataset we used, including evolutionary conservation scores such as PhyloP and GERP, commonly sued in variant effect prediction. Protein structural features, such as domains or active sites, along with splice site and regulatory predictions are also not present.

These missing features may explain cases where the model misclassifies variants, particularly for complex indels or variants in poorly annotated genes.

4.5 Qualitative Insights

The model successfully captures several biological principles:

- **Certain variant types carry a higher pathogenicity risk**: Small indels and structural variants have higher pathogenicity rates compared to SNVs.
- **Gene-level functional categories provide significant predictive value**: Variants which occur in gene-groups associated with disease, such as tumor-suppressors, tend to have a higher predicted pathogenicity.
- **Chromosomal context provides useful predictive information**: Patterns in variant distribution influences the model's predictions, reflecting that certain variant types occure more often in specific genomic regions.

However, the model misses effects like:

- **Specific amino acid substitution properties** (e.g., charge changes, hydrophobicity)
- **Structural context** (e.g., active sites, binding domains)
- **Splice site effects and regulatory region impacts**

These would require richer features or integration with protein structure databases.

4.6 Comparison to Alternative Approaches

Our discrete Bayesian Network offers interpretability advantages over black-box methods like deep learning, as we can inspect CPTs and DAG structure to understand predictions. However, continuous features (e.g., conservation scores, allele frequencies) were discretized, potentially losing information. A hybrid approach using Gaussian Bayesian Networks or conditional linear Gaussian models could leverage continuous data more effectively.

Compared to frequentist methods like logistic regression, our probabilistic model naturally handles missing features during inference and provides full posterior distributions rather than point predictions. This uncertainty quantification is valuable in clinical decision-making.

5 Conclusion

With reliable Accuracy, ROC AUC, and PR-AUC (precision) as discussed in the Results section, we have found that it is possible to create a model that predicts whether a mutation is pathogenic or not based on observed genomic features. This holds even with the removal of observable features deemed to be deterministic, or surrogates for the label. Our key insight is that in genetics, there are many deterministic qualities to a genome. Prior information can be used to come to a conclusion with relatively high confidence.

The main limitation of our model is its binary nature. Pathogenicity is an arbitrary measure, and our model is not able to make judgments on what qualifies a genome for pathogenic status, or on the relative degree of pathogenicity between different genome variants, especially those that affect similar functions. A key extension would involve creating more categories than just pathogenic or not pathogenic, and create lookup tables that would permit the model to advise on health outcomes after predicting the given variant's category, not just declare its category. As well, while our model's precision exceeds our baseline, there is always room to attempt to make it more precise by searching for more features to either record or engineer.

6 Reflections and contributions

For future students trying such a project, we would encourage testing on minimal datasets before expanding to the entire dataset. Loading the data and training a model on it take far longer the larger the dataset is, meaning that the workflow is less agile and it is more difficult to test the efficacy of individual features. As well, we would encourage doing thorough research on the subject material in case you are not familiar with it beforehand, since field knowledge is necessary to pick out good features that are both useful and do not cause data leakage, as well as to appropriately interpret the results of your model's inference. Predicting mutation pathogenicity is actually hard. Reading the DNA sequence and predicting if there is a mutation and determining if it is pathogenic is hard to achieve. A good approach is to find a dataset of human genetic variants processed by clinicians. These datasets annotated functional information about the mutation. It allows us to build an interpretable model without solving the difficult problem of the raw DNA sequence.

Contributions:

- **Michael Kroyan:** I tested and reviewed the code, discovered the data leakage, and wrote the Abstract, Modeling and inference, Conclusion, and Reflections and contributions (bullet 1) sections of this writeup. I found that I did not have much in the way of specific field knowledge, which meant that with each step, I had to do additional research into genetics field subject matters in order to verify whether or not our implementations intrinsically make sense. I learned quite a bit about genetics, but I also learned quite a bit about thorough analysis and testing for data leakage in various configurations; how to make sure that features have providence. These skills will carry forward, as I will inevitably have to test more models in the future.
- **Jiya Makhija:** I worked on cleaning the ClinVar dataset, engineering the biological features for our model, and writing the preprocessing section of the report. Through this, I learned how feature design directly affects a Bayesian Network—especially how binning and simplifying large categories makes CPTs easier to learn. I also learned how to pick reasonable cutoffs by looking at the dataset's real distribution and checking how past studies treat indels and loss-of-function variants. Coming from a non-biology background, I also picked up the basics of why mutations like frameshifts and stop-gains are important and how they're usually approximated when full coding context isn't available. Overall, this helped me understand how data choices, biological assumptions, and modeling decisions interact in a probabilistic system.
- **Janice Rincon:** I worked on implementing the HillClimbSearch algorithm and the Discrete Bayesian Network model to be used for inference. I also contributed the problem description and result sections of the report. This project was challenging because of my limited knowledge of genetics but thankfully the libraries were easy to implement and the results were not too difficult to interpret. With this project, I learned that it is important to print out model values (i.e. CPTs) at each step to verify that the model is doing what you expect. It makes troubleshooting much easier. Having no previous bioinformatics experience, working on this project has made me more interested in the field and the ways that machine learning can be applied to real life problems.
- **Vibusha Vadivel:** I implemented the validations set for tuning the model and also tested the code. On the lab report, I explained the learning and inference process of the model and wrote the results and discussion section. Through this project, I learned how important it is to design a rigorous evaluation pipeline (especially separating validation from test) and to question surprisingly high performance as a possible sign of leakage or modeling issues. One challenge was balancing minimal code changes with adding proper tuning and constraints, which forced me to think carefully about how each step in the pipeline affects the efficiency of the model. Overall, this project gave me a taste of how these abstract theories can make a difference in real-world applications, and this project has sparked my curiosity to explore systematic hyperparameter tuning in learning algorithms in the future.
- **Qiwen Xu:** I contributed to brainstorming multiple project ideas, defining the project scope, overall project planning, and feature selection. I also helped write the reflections and contributions section. Initially, I proposed using a HMM to detect genomic mutations. However, after further consideration and background research, I realized that building a

model to directly analyze raw DNA sequences and predict mutations would be difficult for a five-person team with limited bioinformatics experience and a constrained timeline. As a result, I proposed focusing on analyzing a curated dataset of human genetic variants processed by clinicians and building a model to predict whether a mutation is pathogenic based on selected genomic features.

For this project, we relied on Gen AI assistance for general project planning, code debugging, and LaTex formatting.

7 References

[1] Lin M, Whitmire S, Chen J, Farrel A, Shi X, Guo JT. Effects of short indels on protein structure and function in human genomes. Sci Rep. 2017 Aug 24;7(1):9313. doi: 10.1038/s41598-017-09287-x. PMID: 28839204; PMCID: PMC5570956.

[2] Landrum MJ, Chitipiralla S, Kaur K, Brown G, Chen C, Hart J, Hoffman D, Jang W, Liu C, Maddipatla Z, Maiti R, Mitchell J, Rezaie T, Riley G, Song G, Yang J, Ziyabari L, Russette A, Kattman BL. ClinVar: updates to support classifications of both germline and somatic variants. Nucleic Acids Res. 2024 Nov 23:gkae1090. doi: 10.1093/nar/gkae1090. [PMID:39578691]

[3] Karczewski, K.J., Francioli, L.C., Tiao, G. et al. The mutational constraint spectrum quantified from variation in 141,456 humans. Nature 581, 434–443 (2020). https://doi.org/10.1038/s41586-020-2308-7