Prediction of the Effect of Naturally Occurring Missense Mutations on Cellular N-Acetyl-Glucosaminidase Enzymatic Activity

Ford, C., Nodzak, C., Uppal, A., Zhao, L.

PARTICIPANTS

Colby Ford, M.S. – Doctor of Philosophy student in Bioinformatics and Computational Biology at the University of North Carolina at Charlotte. cford38@uncc.edu

Conor Nodzak, B.S. – Master of Science student in Bioinformatics and Computational Biology at the University of North Carolina at Charlotte. cnodzak@uncc.edu

Aneeta Uppal, B.S. – Master of Science student in Bioinformatics and Computational Biology at the University of North Carolina at Charlotte. auppal@uncc.edu

Lei Zhao, M.S. – Doctor of Philosophy student in Bioinformatics and Computational Biology at the University of North Carolina at Charlotte. lzhao13@uncc.edu

ABSTRACT

The Critical Assessment of Genome Interpretation (CAGI) has proposed a challenge to devise a computational method for predicting the phenotypic consequences of genetic variants of a lysosomal hydrolase enzyme known as α-N-acetyl-glucosaminidase (NAGLU). In 2014, the Human Gene Mutation Database released that 153 NAGLU mutations associated with MPS IIIB and 90 of them are missense mutations. The ExAC dataset indicates 189 missense mutations NAGLU based on about 60,000 individual sequenced exomes and 24 of them are known to be disease associated. Biotechnology company BioMarin has quantified the relative functionality of NAGLU for the remaining subset of 165 missense mutations. For this particular challenge, we examined the subset missense mutations within the ExAC dataset and predicted the probability of a given mutation affecting the level of enzymatic activity. In doing so, we hoped to learn which changes in amino acid physicochemical properties can be tolerated and which cannot be.

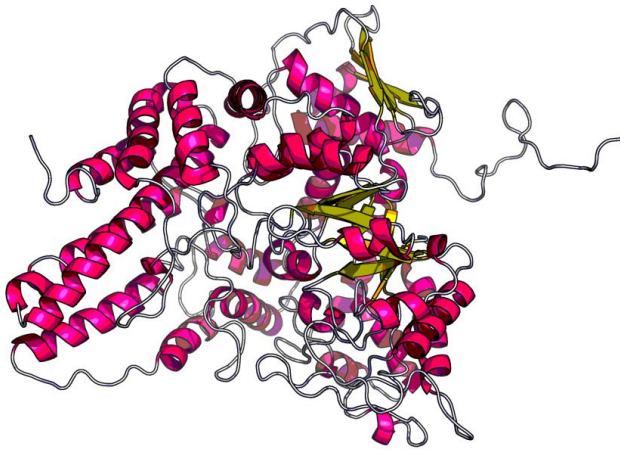

Figure 1: NAGLU Protein Structure
Generated by RaptorX

Amino acid substitution (AAS) prediction methods are mainly based on the sequence and structure information. Simple comparisons between different AAS methods are not only difficult, but also irrational because each method was tested on various datasets and based on varied versions of databases. Currently, several AAS prediction methods have been introduced. PolyPhen-2, an updated version of PolyPhen, is a tool used to predict possible impacts of an amino acid substitution on the structure and function. Users are required to provide protein or SNP identifier, protein sequences, substitution positions, etc. A score is provided, ranging from 0 to 1, corresponding to the probability of a mutation resulting in no functionality for the enzyme.

Once the probability score was generated, the dataset was then run through a large machine learning decision forest algorithm. This attempted to predict the PolyPhen-2 probability score using the other information about the mutation (amino acid type, location, allele frequency, etc.) as independent variables. This generated a predicted aggregate score for each mutation, which was then reported back to CAGI. The results of the analysis are siginifcant enough to hold confidence that the scores are decent predictors of enzymatic activity given a mutation in the NAGLU amino acid sequence.

Background

Sanfilippo Syndrome type B disease (Mucopolysaccharidosis IIIB, MPS IIIB) is an autosomal recessive, neurodegenerative disease of children, with the clinical symptoms including intellectual disability, behavior disturbance and death in second or third decade. MPS IIIB is the inborn error of glycosaminoglycan metabolism and is characterized by the systematic accumulation of heparan sulfate. The lysosomal hydrolase α-N-acetyl-glucosaminidase (NAGLU) performs the function of removing terminal α-N-acetyl-glucosaminidase residues from heparin sulfate and lacking NAGLU is one of the four systematic defects that cause MPS IIIB.

Contest

We have elected to submit our results predicting the effects of the NAGLU enzymatic activity to the Critical Assessment of Genome Interpretation contest. The submitted prediction is numerically valued in the following manner:

Minimum Score: 0	→	Maximum Score: 1
No enzymatic activity		Wild-type enzymatic activity

Table 1: Prediction Scoring

Along with the above values, we reported the amino acid mutation being scored and the standard deviation of the prediction. CAGI also allows for a comment to be added, in which we have included information about whether the mutation is nonsynonymous or not, deleterious or neutral, and benign or damaging.

Impact and Significance

According to the work done by Meikel, et. al., in *Prevalence of Lysosomal Storage Disorders*, Sanfilippo syndrome type IIIB has a prevalence of about 1 in 200,000 people (Meikel, 1999). This disease generally manifests in young children, carried by an autosomal recessive gene. For those affected, the symptoms are debilitating and nonetheless horrible.

The National MPS Society reports that most of the symptoms are highly neurological. "Overactive and difficult" behavior is generally seen in children. As the disease progresses with age, many developmental problems occur such as chewing on the hands, never being able to be toilet trained, learning disabilities, etc. Other symptoms may persist including nausea, digestive problems, and diarrhea as well as the apparent facial formation and distinctive look of MPS-III patients (MPS III, 2011).

Better understanding of this disease will hopefully lead to better diagnosis and better treatment research. Currently, there is no cure for the disease and many people with the disease die at a very early age. Research in gene therapy, enzymatic therapy, and other treatments may benefit from this genetic analysis.

Analysis Process

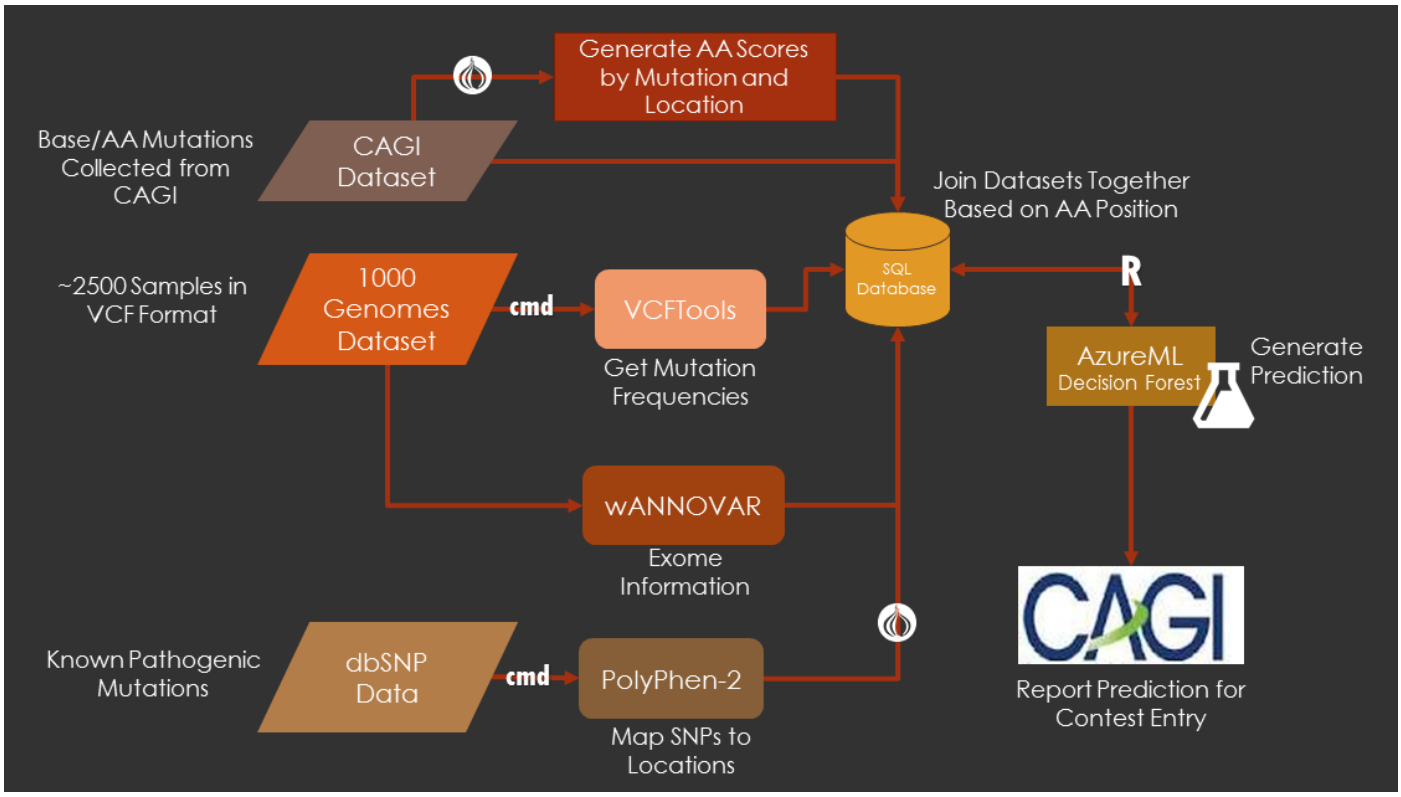

Figure 2: Workflow of the Analysis Process

Struggles with VCF Format

Data from 1000 Genomes is reported in the Variant Call Format. For MPS-111B, the interest is only in a particular region on the 17th chromosome. However, only data for the entire 17th chromosome is available for direct download. This means that once the data was downloaded, the file needed to be trimmed such that we are only working with the region of interest. Upon downloading the entire chromosome, the file was noticeably large: ~23GB for the .vcf file. At this size, it's far too large to manage simply by using a spreadsheet program. This format was also initially difficult to use in its current tabular layout as it seemed to have to needed information, but presented in the wrong way for what we want to do with it. *See table 2 below.*

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	NA00001	NA00002	NA00003
17	14370	rs6054257	G	A	29	PASS	NS=3;DP=14;AF=0.5;DB;H2	GT:GQ:DP:HQ	0|0:48:1:51,51	1|0:48:8:51,51	1/1:43:5:
17	17330	rs6040361	T	A	3	q10	NS=3;DP=11;AF=0.017	GT:GQ:DP:HQ	0|0:49:3:58,50	0|1:3:5:65,3	0/0:41:3
17	1110696	rs6040355	A	G,T	67	PASS	NS=2;DP=10;AF=0.333,0.667;AA=T;DB	GT:GQ:DP:HQ	1|2:21:6:23,27	2|1:2:0:18,2	2/2:35:4

Table 2: Sample of the .vcf file for the human chromosome 17

VCFTools

The discovery of a Perl add-in called VCFTools proved to simplify the process of distilling the ~23GB file into a more manageable and useful format. We ran the tool on the entire .vcf chromosome file and summarized the data to a format that reports the frequency of different alleles at each position on the chromosome. This is summarized from the 2,504 different genomes in the original file. By doing this summarization, the file was much smaller and we then removed the regions of the chromosome not in question. *See table 3 below.*

Chr	Position	NumAlleles	Allele1	Allele2
17	40688016	2	T:0.9998	C:0.000199681
17	40688079	2	G:0.999601	C:0.000399361
17	40688091	2	G:0.986022	C:0.0139776
17	40688124	2	G:0.997604	T:0.00239617
17	40688147	2	G:0.9998	A:0.000199681

Table 3: Summarized information after the use of VCFTools

The CAGI data

The dataset given by CAGI includes information for nucleotide changes at various positions and the resulting amino acid substitution that results from that nucleotide change (Critical Assessment of Genome Interpretation, 2015) *See table 4 below.*

chromosome	variant_position	cDNA_nucleotide_change	AA_substitution
17	40688337	47C>T	A16V
17	40688640	350T>C	V117A
17	40688642	352C>T	P118S
17	40689424	392A>C	Y131S
17	40689436	404T>G	V135G
17	40689442	410C>T	T137M
17	40689453	421T>A	S141T
17	40689487	455G>A	R152Q
17	40689493	461T>C	I154T
17	40689504	472G>T	A158S
17	40689538	506G>A	S169N

Table 4: Sample of the CAGI NAGLU dataset for prediction

PolyPhen-2 Software: Installation and Execution

The first step required when installing this standalone software is to ensure that the computer has three specific perl modules present. Those modules are XML::Simple, DBD::SQLite, and LWP::SImple. In order to check if the computer had these modules, within the terminal window the command "cpan -l" was used. This command then prints a verbose list of the Perl modules already installed. Once it was determined that these modules were not present, the "cpan" command was used again, this time without options. Executing this command then opens the cpan shell, which provides an access point to download the necessary Perl modules from the "Comprehensive Perl Archive Network." *See figure 3 below.*

```
mncomputer:~ mitchnodzak$ cpan
Terminal does not support AddHistory.

cpan shell -- CPAN exploration and modules installation (v2.00)
Enter 'h' for help.

cpan[1]> h

Display Information                                                    (ver 2.00)
 command  argument          description
 a,b,d,m  WORD or /REGEXP/  about authors, bundles, distributions, modules
 i        WORD or /REGEXP/  about any of the above
 ls       AUTHOR or GLOB    about files in the author's directory
    (with WORD being a module, bundle or author name or a distribution
    name of the form AUTHOR/DISTRIBUTION)

Download, Test, Make, Install...
 get      download                  clean    make clean
 make     make (implies get)        look     open subshell in dist directory
 test     make test (implies make)  readme   display these README files
 install  make install (implies test)  perldoc  display POD documentation
```

Figure 3: CPAN functionality

As shown in the screenshot, within the cpan shell one may easily search for the necessary module if the name is known. For example, in order to check on the availability of the XML::Simple Perl module, typing "i /XML::Simple/" will show the package. The command "get" and "install" where then used to acquire the modules and have them ready to use in the local library. Once completed, the task of downloading the source code from the main PolyPhen-2 website was done and the tarballs were then moved to the home directory. The same website also provided tarballs for the precomputed MLC and MultiZ alignment files, which are necessary in order to decrease the overall computational time of the program, as well as bundled databases. From the home directory the tarballs were all then extracted using the recommended command from the documentation which may be generalized as "tar vxjf large_files.tar.bz2." Once extraction was complete, the next task was using ftp to complete the massive bulk downloads of ncbi-blast-2.2.31+ tools, UniRef100 nonredundant protein sequence databases, the PDB database and DSSP database. Next, the configuration of the software prompts the user to provide yes/no answers as to the validity of the path to specific areas in the PolyPhen directory structure, and make any necessary changes, *as shown below in figure 4.*

```
#===============================================================================

#-------------------------------------------------------------------------------
# PolyPhen-2    Change this to point to the directory where PolyPhen-2 is
# home          installed
#-------------------------------------------------------------------------------
PPH             = /Users/mitchnodzak/polyphen-2.2.2
Path seems to be ok
Keep it (Y/n)? Y

#-------------------------------------------------------------------------------
# Scratch       General repository for all files calculated by PolyPhen-2:
#               BLAST search results, multiple alignments, PSIC profiles, etc.
#               The directory should exist and should have the following
#               five subdirectories created inside it:
#                   alignments
#                   blastfiles
#                   profiles
#                   structures
#                   lock
#-------------------------------------------------------------------------------
SCRATCH         = /Users/mitchnodzak/polyphen-2.2.2/scratch
Path seems to be ok
Keep it (Y/n)? Y
```

Figure 4: Configuration prompt of PolyPhen-2

With the installation procedure complete, the program itself is broken up into three main components. The tools within PolyPhen-2 run in a sequential order, beginning with the optional first tool, mapsnps.pl. For the CAGI dataset, which provided the reference amino acids and their mutated variant given the position, the PolyPhen-2 input variant format was chosen as the method for running the data through the pipeline. Mapsnps.pl is an annotation tool that uses the genomic coordinates for each SNP and produces an output file of those variants which produce missense mutations, which is formatted to be piped into run_pph.pl. There are multiple formats for input files, some of which contain only the original and mutated amino acid for a given protein, making this step optional as a result. For the 1000 genomes VCFs, the only information present is the genomic position and the variants, making the initial mapsnps.pl tool necessary in order to extract the nsSNPs. This leads into the next tools for protein variant annotation and probabilistic classification, run_pph.pl and run_weka.pl. The PolyPhen-2 output file for SNPs belonging to our CAGI dataset was subsampled and reformatted by a piece of Perl code written by us in the table below, in order to highlight columns of interest. *Seen below in table 5*, a high probability close to 1.0 indicates amino acid change likely to damage the functionality of the protein.

```
#!/usr/bin/perl
use strict;
use warnings;

my $filename = 'prediction.txt';

open(FILE,$filename) || die "Cannot Open This File!\n";

open my $out_file, '>','result_prediction.txt';

while (<FILE>) {

my @data = split(/\t/,$_);

my $output = join "\t",$data[0],$data[6],$data[7],$data[8],$data[11],$data[14],$data[15],$data[16],$data[17],$data[18],"\n";

print $out_file $output;

}

close FILE;

exit;
```

Figure 5: Sample code for column selection

#o_acc	pos	aa1	aa2	prediction	pph2_class	pph2_prob	pph2_FPR	pph2_TPR	pph2_FDR
P54802	16	A	V	unknown	none				
P54802	117	V	A	possibly damaging	deleterious	0.919	0.0594	0.811	0.0903
P54802	118	P	S	possibly damaging	deleterious	0.553	0.0945	0.878	0.127
P54802	131	Y	S	probably damagin	deleterious	1	0.00026	0.00018	0.0109
P54802	135	V	G	benign	neutral	0.342	0.11	0.9	0.142
P54802	137	T	M	probably damagin	deleterious	1	0.00026	0.00018	0.0109
P54802	141	S	T	probably damagin	deleterious	0.974	0.0438	0.763	0.0722
P54802	152	R	Q	benign	neutral	0.195	0.126	0.916	0.157
P54802	154	I	T	probably damagin	deleterious	0.993	0.0301	0.696	0.0553
P54802	158	A	S	probably damagin	deleterious	0.973	0.044	0.765	0.0723
P54802	169	S	N	benign	neutral	0.004	0.408	0.975	0.328
P54802	181	A	D	possibly damaging	deleterious	0.673	0.085	0.862	0.118
P54802	183	G	A	probably damagin	deleterious	0.982	0.0393	0.748	0.0664
P54802	190	N	S	benign	neutral	0.088	0.15	0.932	0.179
P54802	192	F	L	benign	neutral	0.015	0.209	0.956	0.229
P54802	198	F	Y	possibly damaging	deleterious	0.661	0.0864	0.864	0.119
P54802	202	G	R	possibly damaging	deleterious	0.878	0.0649	0.825	0.0963
P54802	204	M	I	benign	neutral	0.031	0.181	0.947	0.205
P54802	211	D	N	benign	neutral	0.159	0.132	0.92	0.163
P54802	216	P	R	benign	neutral	0	1	1	0.575

Table 5: Sample PolyPhen-2 output of CAGI dataset parsed by the code described above.

Shown above, the output of PolyPhen-2 generated a 38 column table, so the columns relevant for our prediction purposes were extracted using the PERL script above.

As previously stated, the data from the 1000 genomes is provided as a vcf file, which provides the genomic coordinates of each of the mutations, but lacks any clear information to ascertain which result in nonsynonymous mutations within the exonic regions

for NAGLU. Polyphen-2 has a perl script "mapsnps.pl" to handle this task, however the output from this tool showed that there were no non-synonymous mutations of the 253 variants called within the region of the NAGLU gene. This result seemed unlikely, and a test was done by joining the table of 1000 genomes SNPs on the table of CAGI nsSNPs, which showed some matches. To determine which variants from the 1000 genomes data were nsSNPs, and thus variants that could serve as input for run_pph.pl and the run_weka.pl scripts, wANNOVAR was used. These variants need to be processed through this software in order to make the necessary comparisons across datasets.

wANNOVAR

To accomplish the task of accurately describing the 1000 genomes dataset, wANNOVAR was employed. This tool can be found at http://wannovar.usc.edu/, and allows for the selection the hg19 genome build used by 1000 genomes, and several different file formats. The header from the original VCF was not maintained, so the input of the variants was converted to "annovar" format, which is extensively described in the documentation and available tutorials. In brief, the input format describes the variant genomic position start and end, reference nucleotide, and the variant nucleotide in separate columns. This web-based tool, created by Dr. Kai Wang at the University of Southern California, annotates the set of variants, allowing for the extraction of those SNPs designated as being exonic and resultant in nonsynonymous mutations.

Database Creation

As you can see, there are various steps in our analysis process that have yielded separate datasets. In order to harness all of the information that has been collected, a database was created with architecture to handle and combine the various data. *See table 6.*

Table	Utilized Contents	Source
1000GenomesSummary	Alleles and their respective frequencies for each position on the chromosome	1000 Genomes (Summarized by VCFTools)
AARef	Reference amino acids by position for NAGLU. Also contains domain information for the sequence.	NCBI
CAGIData	Wild Type/Mutant codon base and amino acid changes by position. (Additional information: amino acid types, domain, and an indicator if the amino acid type changed.	CAGI (Additional information added manually)
PolyPhen_CAGIPredictions	Predictions for functionality changes in the amino acid mutations from the CAGI dataset input.	PolyPhen2
wANNOVAR1000Gexome_summary	Exonic function for each position. (Nonsynonymous mutations) References 1000 Genomes.	wANNOVAR

Table 6: Database tables used for in analysis

These data were housed in a Microsoft SQL Server 2014 server instance. Importing the datasets was performed using SQL Server Integration Services, an enterprise-grade Extract, Transform, Load (ETL) software. Once the data was successfully put into the database in their respective tables, the tables were joined together by amino acid sequence position. This combines, for example the .vcf summary with the CAGI data with the PolyPhen probabilities, resulting in one final SQL view that was used as the input dataset for our analysis.

MACHINE LEARNING

Using the conglomerate dataset, we utilized a large Decision Forest algorithm to make our prediction. At first, we ran the data through R using the randomForest package (RColorBrewer, 2012). *See figure 6.*

The R package was called using the following parameters:

Parameter	Selection
Dependent Variable	pph2_prob
Independent Variables	Wt_nucleotide Mut_nucleotide AA_position Wt_AA_Name Mut_AA_Name Wt_AA Mut_AA Mut_Codon_Pos Wt_AA_type Mut_AA_type AATypeChange Domain allele1_freq
Number of Trees	5, 25, … ,10,000

Table 7: Parameters for Decision Forest Algorithm.
(The same parameters apply for Microsoft Azure Machine Learning as well as the R randomForest package.)

```
##Must have randomForest package installed
#install.packages("randomForest")

##Microsoft Azure Machine Learning Initialization
#Define Datasets by Port
train <- maml.mapInputPort(1) #class: data.frame
test <- maml.mapInputPort(2) #class: data.frame

#Call the Library
library(randomForest)

#Get Probabilistic Complements
train$pph2_prob <- 1-train$pph2_prob
test$pph2_prob <- 1-test$pph2_prob

##Generate Prediction
forest <- randomForest(pph2_prob ~ Wt_nucleotide + Mut_nucleotide + AA_position +
Wt_AA_name + Mut_AA_name + Wt_AA + Mut_AA + Mut_Codon_Pos + Wt_AA_type + Mut_AA_type +
AATypeChange + Domain + allele1_freq,data = train,ntree = 25)
pred <- as.data.frame(predict(forest,test,type="response",predict.all=TRUE)

#Add Aggregate Prediction Score to Test Data
test$prediction <- pred$aggregate

#Calculate Standard Deviation for randomForest Trees and Add to Test Data
indiv <- pred
indiv$aggregate <- NULL
test$pred.sd <- apply(indiv,1,sd)

#Map Output to Azure ML Output Port
maml.mapOutputPort("test");

###################################
```

Figure 6: Sample R-Script for randomForest package
(For use in Azure Machine Learning environment)

Although the randomForest package works well for a small number of trees, there was evidence to support that using a larger number of random forest trees to make the prediction would increase the predictive power of the algorithm and therefore yield a better result. To handle this, we harnessed the power of the

cloud by then running the dataset through the Microsoft Azure Machine Learning system (Microsoft Corporation, 2015). This allowed for much larger random forests to be created. *See figure 7.*

For our final prediction, we created a forest that contained 10,000 trees. Running this through R on a local desktop would have taken a very long time. The cloud, however, completed the experiment in 15.25 minutes. The experiment adds two columns to the data: Score Label Mean (the arithmetic mean of the predictions of each of the 10,000 trees) and Scored Label Standard Deviation (the standard deviation of the prediction over all 10,000 individual trees), which is required by CAGI.

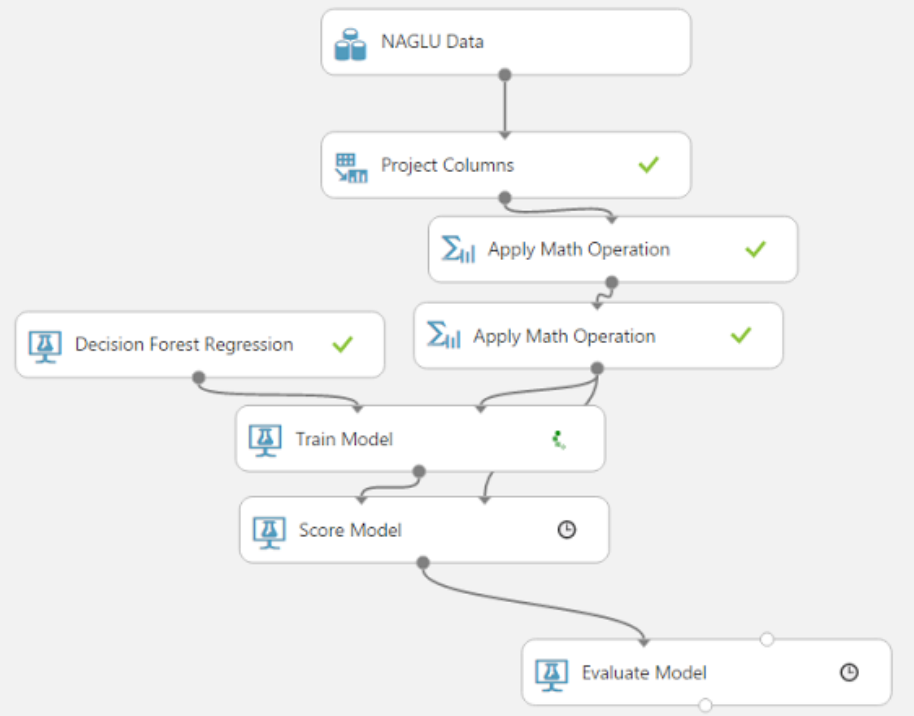

Figure 7: Azure Machine Learning Workflow

RESULTS

After attaining the predictions from the Azure Machine Learning system, we joined the data back up with the rest of the information housed in our database. CAGI allows for only four columns to be submitted to the contest: AA_subsitution (from their original dataset), relative activity (predicted using the PolyPhen2 probabilities in the Azure Machine Learning Decision Forest experiment), standard_deviation (computed from the 10,000 different predictions from the large Decision Forest), and comment. For the comment column, we included the PolyPhen2 predictions along with the wANNOVAR functional information. *See table 8 below.*

AA_substitution	relative activity	standard_deviation	comment
A16V	0.404481	0.569491	PolyPhen2 Prediction: none|PolyPhen2 Class: unknown| wANNOVAR Exonic Function: nonsynonymous SNV
V117A	0.376404	0.41849	PolyPhen2 Prediction: deleterious|PolyPhen2 Class: possibly damaging|
P118S	0.630965	0.421451	PolyPhen2 Prediction: deleterious|PolyPhen2 Class: possibly damaging|
Y131S	0.041277	0.198787	PolyPhen2 Prediction: deleterious|PolyPhen2 Class: probably damaging|
V135G	0.47299	0.328624	PolyPhen2 Prediction: neutral|PolyPhen2 Class: benign|
T137M	0.022574	0.079438	PolyPhen2 Prediction: deleterious|PolyPhen2 Class: probably damaging| wANNOVAR Exonic Function: nonsynonymous SNV

Table 8: Sample of the submission dataset for the CAGI competition

The Evaluate Model node in the Azure Machine Learning studio conveniently reports model metrics for the Decision Forest experiment. *See table 9.*

Final Decision Forest Output Statistics:	
Mean Absolute Error	0.203338
Root Mean Squared Error	0.267673
Relative Absolute Error	0.509781
Relative Squared Error	0.394408
Coefficient of Determination	0.605592
Average Standard Deviation	0.331365452
Minimum Standard Deviation	0.002228193
Maximum Standard Deviation	0.645632309

Table 9: Statistics of the Decision Forest prediction

SUPPLEMENTARY ANALYSIS

BIOPERL & PERL

To first execute the alignment, most online documentation made it seem apparent that BioPerl was a better tool to use for sequence alignments. BioPerl runs through modules designed by Christopher Fields for a wide array of applications such as DNA statistical analysis, sequence alignments and many more. When trying to use BioPerl we ran into numerous issues. As Macintosh continues to update their Mac-OSX software, it seems that BioPerl tends to lag behind. All of the documentation supported alignment features that were out of date. The build in Perl modules use specific programs that had to be installed on the user's machine, such as NCBI Blast. However, NCBI Blast has updated their program to no longer contain the Bl2seq feature. This was a big issue because all of the documentation for aligning sequences used the Bl2seq program. Next, we figured we could use a different program and still run it through BioPerl such as Tcoffee or Muscle. These two programs are also used as alignment tools for protein sequences. Tcoffee and Muscle installed on the machine perfectly. However, the BioPerl modules for Tcoffee and Muscle had to be separately installed through BioPerl "BioPerl-Run" add on package. When

trying to install the "BioPerl-Run" package through the terminal, following the documentation, the program would fail to install due to a certain number of failed tests during compilation. There was also a "BioPerl-Ext" package that contained other modules that could be useful. However, that package could only be installed once "BioPerl-Run" had successfully installed. After reading many online sources, it was stated that many core features have been removed from the "BioPerl-Run" package, causing the tests to fail upon compilation. Some people were able to correct the issue by installing parts of the package piece by piece and testing each piece before continuing.

We also were not sure whether the variation between our scores would even be reliable to draw any conclusions. So to spend this much time still in BioPerl was no longer worth it. Instead, a Perl script was written in order to generate a score based on the reference sequence for NAGLU. The Perl script was able to count all of the amino acids in the reference sequence and generate an overall score of the protein sequence based on the Blosum62 scoring matrix. *See figures 8 and 9 below.* When generating a new score for the alignment based on the single amino acid change, there was not a large variation between the scores. We did see the general trend that higher scores were generated when the amino acid substitution did not change the charge. However, those scores were still not consistent throughout. Amino acids that changed from basic charge groups to polar charged groups seemed to have the lowest scores in the data set, but it was not true for every basic to polar amino acid change. This does not allow us to conclude that every basic charge change to a polar charge generates the lowest score. If we had more than one mutation to base our conclusion off of, this could have given us more vital information. Even when executed on BLAST to check scores, BLAST did not generate more than a 1% variation difference with these sequences. *See table 10 and figures 10 and 11 below.* If we had several mutations to study the scores would have varied more and we may have been able to draw a better conclusion.

```
my $input =
    'MEAVAVAAAVGVLLLAGAGGAAGDEAREAAAVRALVARLLGPGPAADFSVSVERALAAKPGLDTYSLGGGGAARVRVRGSTGVAAAAGLHRYLRDFCGCHVAWSGSQLRL
    PRPLPAVPGELTEATPNRYRYYQNVCTQSYSFVWWDWARWEREIDWMALNGINLALAWSGQEAIWQRVYLALGLTQAEINEFFTGPAFLAWGRMGNLHTWDGPLPPSWHIK
    QLYLQHRVLDQMRSFGMTPVLPAFAGHVPEAVTRVFPQVNVTKMGSWGHFNCSYSCSFLLAPEDPIFPIIGSLFLRELIKEFGTDHIYGADTFNEMQPPSSEPSYLAAATT
    AVYEAMTAVDTEAVWLLQGWLFQHQPQFWGPAQIRAVLGAVPRGRLLVLDLFAESQPVYTRTASFQGQPFIWCMLHNFGGNHGLFGALEAVNGGPEAARLFPNSTMVGTGM
    APEGISQNEVVYSLMAELGWRKDPVPDLAAWVTSFAARRYGVSHPDAGAAWRLLLRSVYNCSGEACRGHNRSPLVRRPSLQMNTSIWYNRSDVFEAWRLLLTSAPSLATSP
    AFRYDLLDLTRQAVQELVSLYYEEARSAYLSKELASLLRAGGVLAYELLPALDEVLASDSRFLLGSWLEQARAAAVSEAEADFYEQNSRYQLTLWGPEGNILDYANKQLAG
    LVANYYTPRWRLFLEALVDSVAQGIPFQQHQFDKNVFQLEQAFVLSKQRYPSQPRGDTVDLAKKIFLKYYPGWVAGSW';

$input =~ s/\s//g;
my @seq = split ('', $input);

my $count_of_A = 0;
my $count_of_C = 0;
my $count_of_S = 0;
my $count_of_T = 0;
my $count_of_P = 0;
my $count_of_G = 0;
my $count_of_N = 0;
my $count_of_D = 0;
my $count_of_E = 0;
my $count_of_Q = 0;
my $count_of_H = 0;
my $count_of_R = 0;
my $count_of_K = 0;
my $count_of_M = 0;
my $count_of_I = 0;
my $count_of_L = 0;
my $count_of_V = 0;
my $count_of_F = 0;
my $count_of_Y = 0;
my $count_of_W = 0;

    for my $AA (@seq) {
    if ($AA eq 'A' ) {
        ++$count_of_A;
    } elsif ($AA eq 'C' ) {
        ++$count_of_C;
    } elsif ($AA eq 'S' ){
            ++$count_of_S;
    } elsif ($AA eq 'T' ){
        ++$count_of_T;
    } elsif ($AA eq 'P' ){
            ++$count_of_P;
    } elsif ($AA eq 'G' ){

my $count_of_Anew = $count_of_A * 4;
my $count_of_Cnew = $count_of_C * 9;
my $count_of_Snew = $count_of_S * 4;
my $count_of_Tnew = $count_of_T * 4;
my $count_of_Pnew = $count_of_P * 7;
my $count_of_Gnew = $count_of_G * 6;
my $count_of_Nnew = $count_of_N * 6;
my $count_of_Dnew = $count_of_D * 6;
my $count_of_Enew = $count_of_E * 5;
my $count_of_Qnew = $count_of_Q * 5;
my $count_of_Hnew = $count_of_H * 8;
my $count_of_Rnew = $count_of_R * 5;
my $count_of_Knew = $count_of_K * 5;
my $count_of_Mnew = $count_of_M * 5;
my $count_of_Inew = $count_of_I * 4;
my $count_of_Lnew = $count_of_L * 4;
my $count_of_Vnew = $count_of_V * 4;
my $count_of_Fnew = $count_of_F * 6;
my $count_of_Ynew = $count_of_Y * 7;
my $count_of_Wnew = $count_of_W * 11;

my $totalscore = ($count_of_Anew + $count_of_Cnew + $count_of_Snew + $count_of_Tnew + $count_of_Pnew +
    $count_of_Gnew + $count_of_Nnew + $count_of_Dnew + $count_of_Enew + $count_of_Qnew + $count_of_Hnew +
    $count_of_Rnew + $count_of_Knew + $count_of_Mnew + $count_of_Inew + $count_of_Lnew + $count_of_Vnew +
    $count_of_Fnew + $count_of_Ynew + $count_of_Wnew);

print "The total max score for this sequence: $totalscore\n";
```

Figure 8: Samples of Perl script for Reference Amino Acid Scoring

```
Aneeta-Uppals-MacBook-Pro:Downloads aneetauppal$ perl project.pl
There are 743 total amino acids in this sequence
The total max score for this sequence: 3910
```

Figure 9: Output of Perl script for Reference Amino Acid Scoring

AA_substitution	Score for 1 AA	Score for Total
A16V	0	3906
A16V	0	3906
T137M	-1	3905
T137M	-1	3905
S141T	1	3907
S141T	1	3907
A158S	1	3907
A158S	1	3907
G183A	0	3904
G183A	0	3904
M204I	1	3906
M204I	1	3906
D211N	-1	3903
D211N	-1	3903
R228Q	1	3906
R228Q	1	3906
R228W	-3	3902
R228W	-3	3902
R297Q	-1	3904
R297Q	-1	3904
R375C	-3	3902
R375C	-3	3902
Q398K	1	3906
Q398K	1	3906
S449N	1	3907
S449N	1	3907

Table 10: Sample Output of Amino Acid Scoring

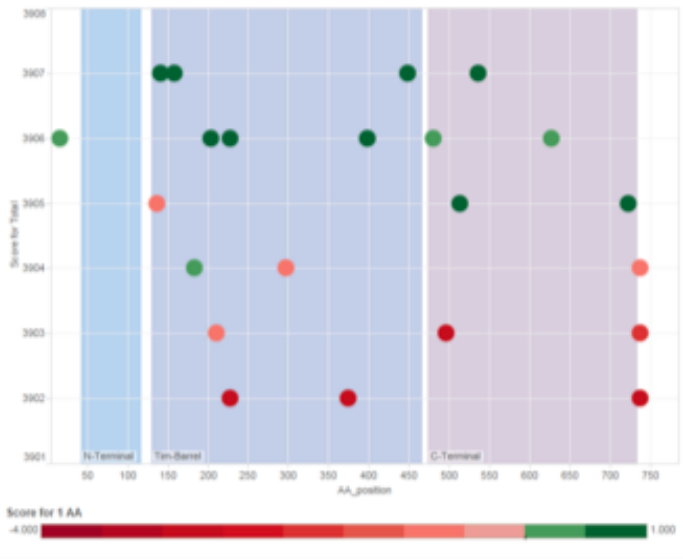

Figure 10: Scatter plot of generated Amino Acid substitution scores

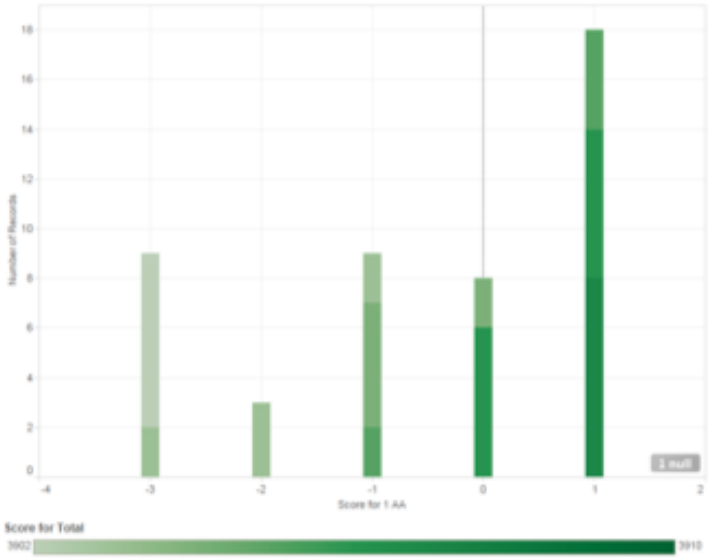

Figure 11: Bar chart of generated Amino Acid substitution scores

Visual Analysis

Further analysis was also performed to attempt to visually identify any patterns between amino acid type changes (polar to non-polar, for example) and location in the sequence. This showed no visible pattern. However, it did show that the Decision Forest performed best (had the lowest standard deviation) for the extrema cases in the amino acid changes (those closest to 0 or closest to 1). *See figure 12 below.*

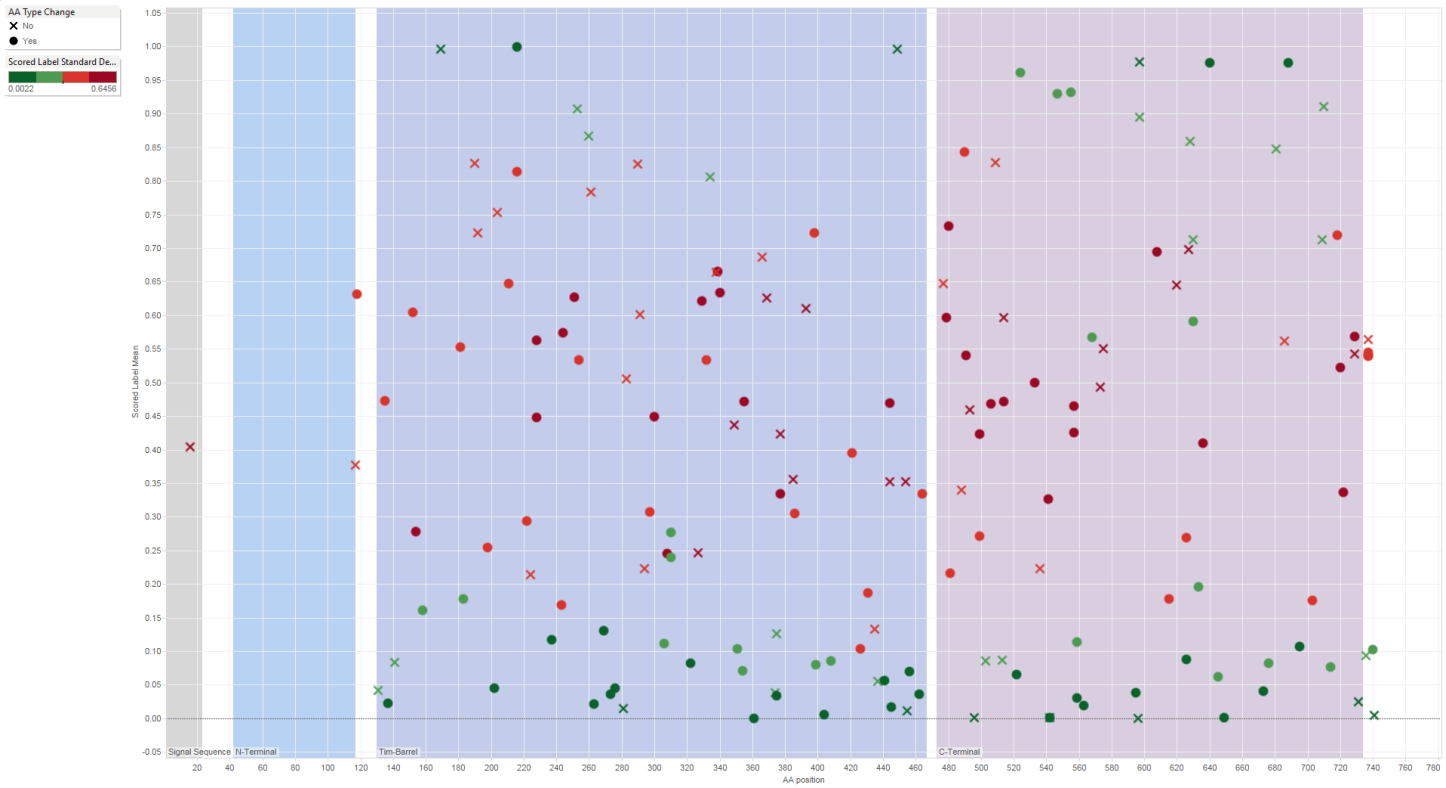

Figure 12: Visualization of variant position versus amino acid type change.
(Also note that standard deviation and domain are also shown.)

Furthermore, plotting exonic function versus position and type changes also showed very little in regards to patterns. *See figure 13 below.*

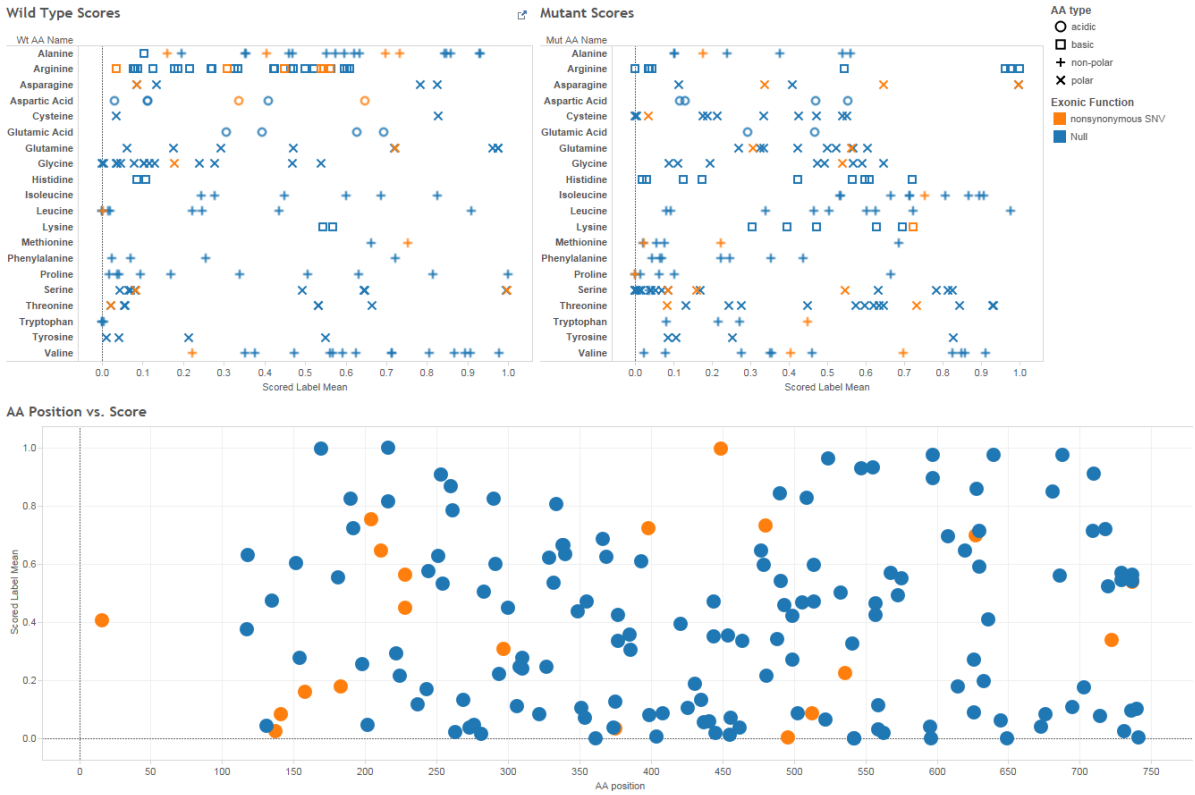

Figure 13: Visualization of exonic function versus position and amino acid type change.

Future Work

Future work is still needed to test the validity of these predictions. The predictions are made solely on the likelihood that a given mutation in an amino acid sequence will affect the functionality of the resulting enzyme. Just because our analysis gave a low score for a mutation (meaning that the mutation is likely lower the enzymatic activity in reference to wild-type activity), the change may not cause anything at all or may enhance the functionality of the enzyme.

One other note is that the independent variables that we have taken into account definitely do not explain all the variation in the enzymatic activity for a mutation. There may be other factors to take into account, which would then need to be incorporated into the data to again predict the enzymatic activity.

Related Work and References

Adzhubei IA, Schmidt S, Peshkin L, Ramensky VE, Gerasimova A, Bork P, Kondrashov AS, Sunyaev SR. Nat Methods 7(4):248-249 (2010). http://genetics.bwh.harvard.edu/pph2/dokuwiki/downloads

Adzhubei I, Jordan DM, Sunyaev SR. Predicting Functional Effect of Human Missense Mutations Using PolyPhen-2. *Current protocols in human genetics / editorial board, Jonathan L Haines . [et al]*. 2013;0 7:Unit7.20. doi:10.1002/0471142905.hg0720s76.

Altschul, S.F., Gish, W., Miller, W., Myers, E.W. & Lipman, D.J. (1990) "Basic local alignment search tool." J. Mol. Biol. 215:403-410. PubMed

Altschul, S.F. & Gish, W. (1996) "Local alignment statistics." Meth. Enzymol. 266:460-480. PubMed

Critical Assessment of Genome Interpretation. (2015) NAGLU dataset. https://genomeinterpretation.org/content/4-NAGLU

Dayhoff, M.O., Schwartz, R.M. & Orcutt, B.C. (1978) "A model of evolutionary change in proteins." In "Atlas of Protein Sequence and Structure, vol. 5, suppl. 3." M.O. Dayhoff (ed.), pp. 345-352, Natl. Biomed. Res. Found., Washington, DC.

Exome Aggregation Consortium (ExAC), Cambridge, MA (URL: http://exac.broadinstitute.org) [25 (September, 2015) accessed].

Ho, T. K. (1995). Random decision forests. *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, 278-282.

Meikle PJ, Hopwood JJ, Clague AE, Carey WF. Prevalence of Lysosomal Storage Disorders. *JAMA*. 1999;281(3):249-254. doi:10.1001/jama.281.3.249.

MPS III. (2011, May 4). Retrieved September 30, 2015. http://mpssociety.org/mps/mps-iii/

Microsoft Corporation. (2015, July 8). Decision Forest Regression. Retrieved from https://msdn.microsoft.com/library/azure/562988b2-e740-4e3a-8131-358391bad755

Online Mendelian Inheritance in Man (OMIM)- N-acetyl-glucosaminidase-alpha, NAGLU http://www.omim.org/entry/609701

Jonathan H. LeBowitz, Wyatt T. Clark, G. Karen Yu BioMarin Pharmaceutical, Inc. 105 Digital Drive, Novato CA 94949

Ng, P. C., & Henikoff, S. (2006). Predicting the effects of amino acid substitutions on protein function. *Annu. Rev. Genomics Hum. Genet.*, *7*, 61-80.

Peng Yue, Eugene Melamud and John Moult. SNPs3D: Candidate Gene and SNP selection for Association Studies. BMC Bioinformatics. 2006 Mar 22;7(1):166.

RColorBrewer, S., Liaw, A., Wiener, M., & Liaw, M. A. (2012). Package 'randomForest'.

Stajich JE, Block D, Boulez K, Brenner SE, Chervitz SA, Dagdigian C, Fuellen G, Gilbert JG, Korf I, Lapp H, Lehväslaiho H, Matsalla C, Mungall CJ, Osborne BI, Pocock MR, Schattner P, Senger M, Stein LD, Stupka E, Wilkinson MD, and Birney E. *The Bioperl toolkit: Perl modules for the life sciences.* Genome Res. 2002 Oct;12(10):1611-8. DOI:10.1101/gr.361602 | PUBMED ID:12368254 | HUBMED [bioperl2002]

Stajich JE and Hahn MW. *Disentangling the effects of demography and selection in human history.* Mol Biol Evol. 2005 Jan;22(1):63-73.DOI:10.1093/molbev/msh252 | PUBMED ID:15356276 | HUBMED [perlmorphism2005]

Stein LD, Mungall C, Shu S, Caudy M, Mangone M, Day A, Nickerson E, Stajich JE, Harris TW, Arva A, and Lewis S. *The generic genome browser: a building block for a model organism system database.* Genome Res. 2002 Oct;12(10):1599-610. DOI:10.1101/gr.403602 | PUBMED ID:12368253 |HUBMED [gbrowse2002]

Stenson, P. D., Mort, M., Ball, E. V., Shaw, K., Phillips, A., and Cooper, D. N. (2014) The Human Gene Mutation Database: building a comprehensive mutation repository for clinical and molecular genetics, diagnostic testing and personalized genomic medicine Hum Genet 133, 1-9.

Thomas, P. D., Campbell, M. J., Kejariwal, A., Mi, H., Karlak, B., Daverman, R., ... & Narechania, A. (2003). PANTHER: a library of protein families and subfamilies indexed by function. Genome research, 13(9), 2129-2141.

Tisdall, J. (2001). Beginning Perl for bioinformatics (1st ed., p. 400). Beijing: O'Reilly & Associates, Sebastopol, CA.