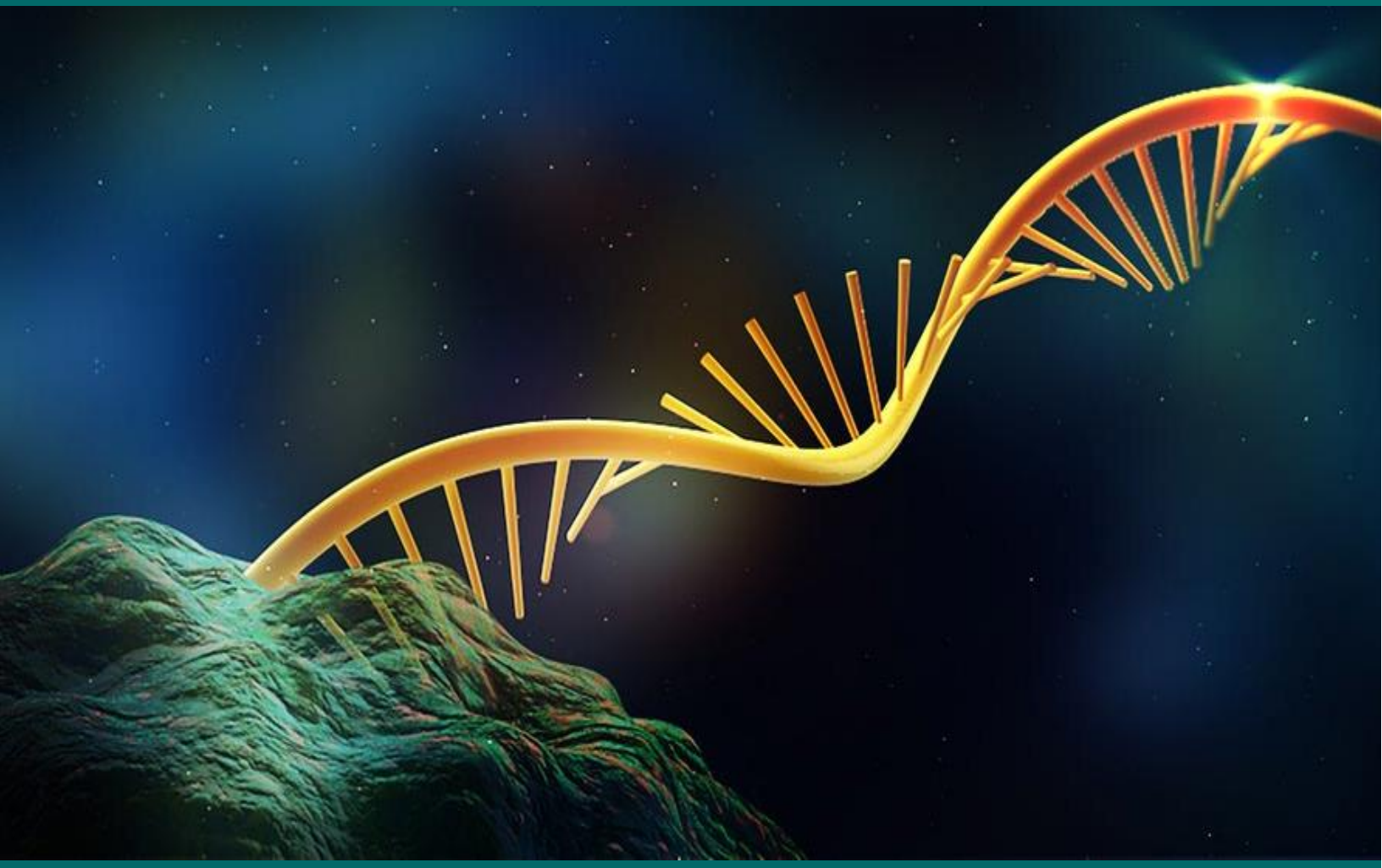
Predicting RNA Editing Sites



Presented By

Ashiqur Rahman (#011142014)

Alif Choyon (#011142002)

Md. Hasanuzzman (#011142017)

Rana Md Shahariar Parbez (#011142071)

United International **University**

Report on

Predicting RNA editing sites

Under the Course

Machine Learning (CSE 489)

Presented By

ASHIQUR RAHMAN (#011142014)

ALIF CHOYON (#011142002)

MD. HASANUZZMAN (#011142017)

RANA MD SHAHARIAR PARBEZ (#011142071)

Advisor

Dr. Swakkhar Shatabda

Associate Professor

Department of Computer Science & Engineering

United International University

TABLE OF CONTENTS

1. Summary

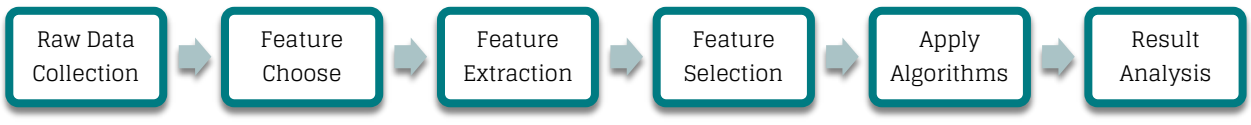
RNA editing sites can alter the transcriptional targets in most eukaryotic cells from fungi to human and therefore it is important. Adenosine to Inosine (A to I) conversion is one of the many RNA editing processes. Which RNA codons are impactful for this modification are still unknown. Some general RNA string properties may give us some clues about this process. Playing with some RNA strings gave us pretty good results.

2. Introduction

The primary directive was to build a system that, can split RNA sequence and extract features that might help to classify if the existence of Adenosine is positive or negative along with if a certain portion of RNA is editable or not. We ended up generating 4000+ features and started training different models using different learning scheme and the results were meaningful. As we learned the nature of the dataset we were able to select important features and found out hidden relations among them.

3. Methodology

The following figure describes our working procedure. We divided our method into six major segments.



3.1. Raw Data Collection

Sequencing RNA and collecting some good strings is the primary step of our experiment. We have used the dataset constructed by Laurent and his colleague for further progress[1]. This dataset is a collection of positive and negative RNA strings.

For testing the strength of our model we've also collected a test dataset which is a work of Yu and his colleagues[2].

3.2. Feature Choose

The Adenosine to Inosine (A-to-I) editing is dependent on various known and unknown patterns of RNA segments. As most of the factors are still unknown to us, we have considered some general string properties of the RNA strings. These following types of features are taken into consideration in our experiment.

- **Frequencies of monograms:** Number of occurrence of each of the bases in the RNA string. That gave us 4 features as RNAs are consists of 4 unique bases which are A, G, U and C.

  E.g. for a RNA string "AACCTGTAAAT" the count of A is 5. Now normalize it by the length of the string, feature monogram (A) = 5 / length ("AACCTGTAAAT") =5/11. Similarly, we calculated frequencies for C, G, and T.
- **Bi grams:** These are also known as di-nucleotides. Here we count all possible occurrences of di-nucleotides: AA, AC, AG, CA, CC, AT and so on. Here total number of features will be 16.
- **Tri grams:** Normalized count of AAA, AAC, and so on. Total number of features are 64.
- **Quad grams:** Count 4-mers and normalize. Total features 256.
- **Gapped n-peptides:** Let's assume we are counting occurrences of A-A. Which means two A's with don't cares in between. These are called gapped di-peptides. If there is one don't care, it means gap=1. 16 features for gap=1. We have considered extraction up to quad (**----**) gapped peptides with up to 10 gaps in between. That gave us 4000 features.

  No of monograms + no of all possible bigrams = 4 + 16 = 20

  Total features = 20 * 10 * 20 = 4000

- **Ratio of Start and End Codons:** Special sequence AUG is known as start codon and UGA, UAA, UAG are known as end codons. Ratio of start and end codon frequencies gave us one feature.
- **Distribution of the Bases:** This feature represents where a base exists the most amount of time throughout the RNA string. For this we divided the RNA string into three portions, calculated where the base counts in each of the portions and find out the denser one. Extracted 4 features from this property.

3.3. Feature Extraction

After choosing the features that we had to extract the features from the RNA strings. For this purpose, we wrote a computer program which can extract the features and save them in a comma separated value file format (CSV file). The pseudo code is given below.

```
Feature Extractor:
      Input: A txt file containing RNA stings in fasta format, input.txt
            Output: A CSV file, output.csv
      Make a list f to store feature names
      Store all feature names in f
      For each RNA string r_i
            For each feature f_j
                  Match the f_j in r_i
                  If matched
                        Write the normalized value in the output file
                  End if
            End For
      End For
```

3.4. Feature Selection

Every model is a combination of pattern and noise in a distribution dependent on the dataset/training set. A pattern is the true nature of a dataset which can be constructed with a learning scheme and Noise is the error of that.

Features are the unique properties of an instance that identifies itself. Some of the features might lead us to pattern or noise but what we need is a noise-free model.

Now, we are describing how we gradually approached our prime model:

At first, we tried out the common approaches to feature selection by building different trees to find out which attributes hold more significance always resulted from an over fitted model that led to poor performance in independent set.

After that, to measure the 4000+ features, we tried BestFirst method and CfsSubsetEval evaluator and it worked like a charm revealing that only 278 features are enough to make an (almost) noise-free model with a better accuracy of 84.33% with a precision of 90% in the trained set.

From the very first Naïve Bayes and random forest were showing promising accuracy and sensitivity so we started trying everything related to those algorithms and as a result, Hoeffding tree was quite capable of classifying as we were using the leaf prediction strategy as Naïve Bayes adaptive.

We further extracted more features from the extracted set (278 features) to (179 features) by using a Decision tree, Logistic and Simple logistic classifiers where precision is mostly dependent which led to increased accuracy of 86% in training set and 95% sensitivity and in the test-set or independent-set 90.667% sensitivity.

3.5. Apply Machine Learning Algorithms

Some machine learning algorithms:

Naïve Bayes: Naïve Bayes is a popular classifier combining the power of probability and algorithm can find out posterior probability based on the prior and conditional probability.
SMO: Support Vector Machine (Used as Classifier) is another popular algorithm to separate data points that with the largest possible space with the help of assigning support vectors.

Random Forest: Random Forest creates random decision trees and uses the majority voting like an ensemble learner.

Bagging: Bagging is another ensemble learner which combines multiple classifiers and uses majority voting.

Adaptive Boosting (AdaBoost): AdaBoost is an ensemble learner which learns thought weight, classifies with multiple classifiers using (weighted) voting.

Hoeffding Tree: Hoeffding tree is an enchanting learning algorithm which can learn from small dataset (supported mathematically by Hoeffding bound) by choosing optimal splitting attribute.

We have applied the mentioned algorithms using 10-fold cross validation using WEKA[3]:

Algorithm	Sensitivity (%)	Specificity (%)	Accuracy (%)	Sensitivity* (%) (Independent)
Naïve Bayes	92.00	78.20	85.25	86.67
SMO	93.60	89.90	91.80	64.67
Random Forest	92.80	73.90	83.61	83.67
Bagging	88.00	70.60	79.51	82.67
AdaBoostM1	96.00	81.50	88.93	85.33
Hoeffding Tree	95.20	77.30	86.48	90.67

3.6. Result Analysis

In comparison, we can see that Naïve Bayes, Ada Boost (Homogeneous Hoeffding Tree) and Hoeffding tree are doing impressive performance where Support vector machine (SMO) has over fitted the training set with 91.8033% accuracy because of the poor performance in independent set.

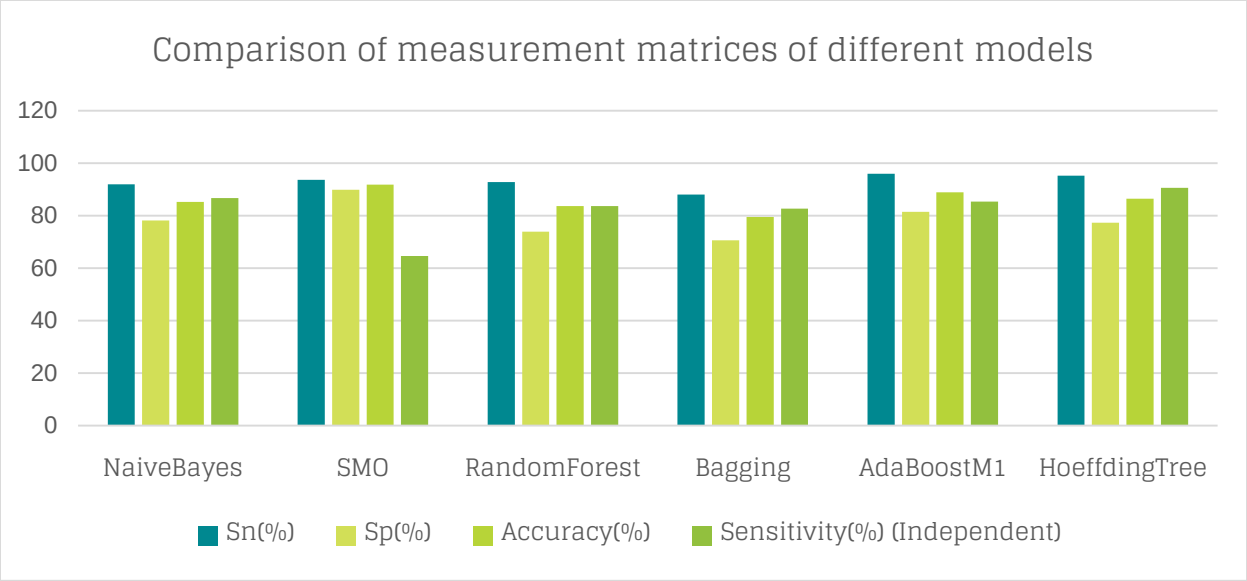


Figure: Comparison of measurement matrices of different models

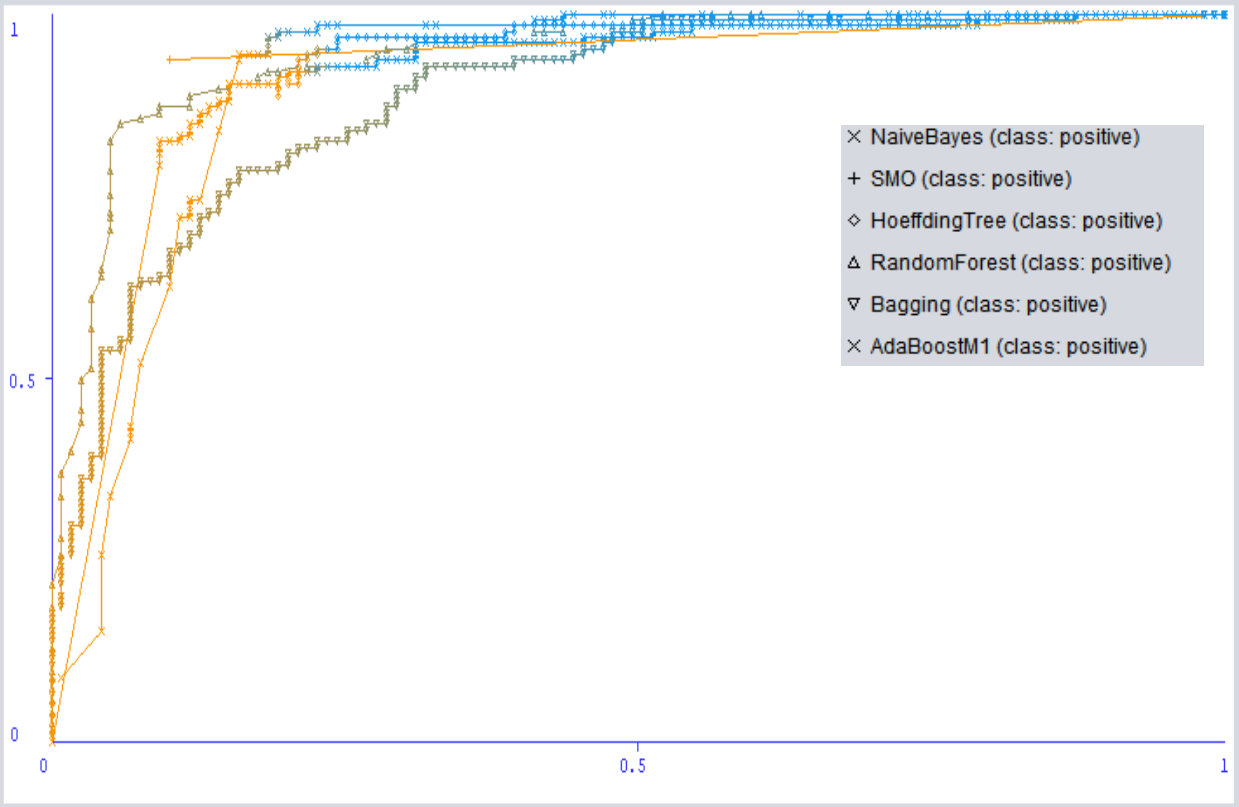


Figure: Combined ROC curve (x-axis: false positive rate, y-axis: true positive rate)

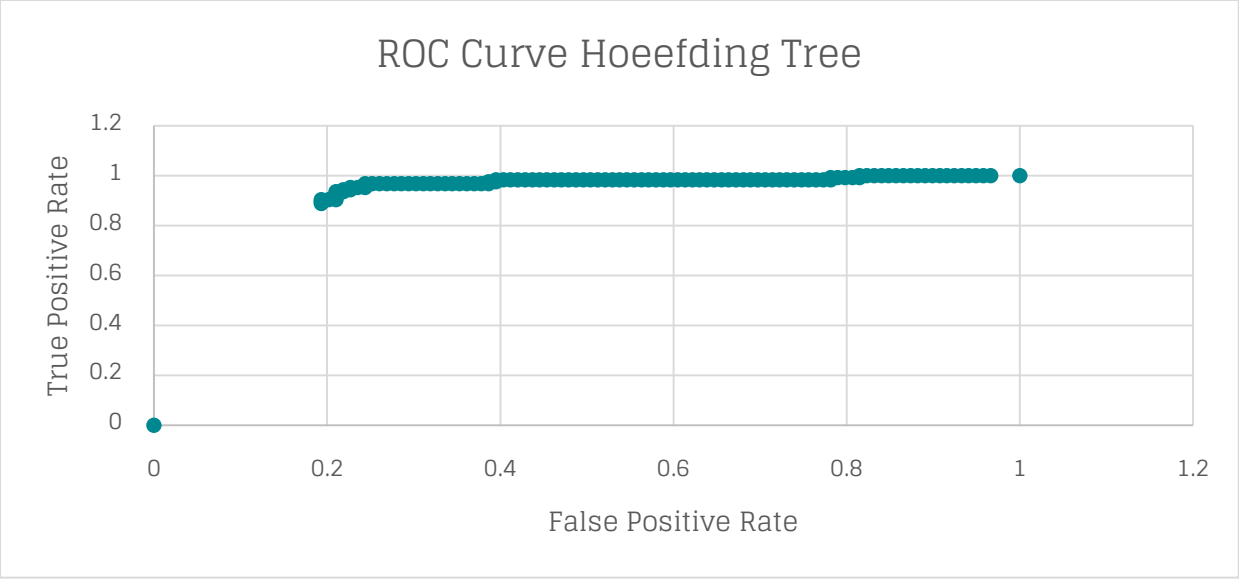


Figure: ROC curve of Hoeffding tree

In the figure above, the curve tends to touch y=1 line which is an out-standing performance.

In PAI: Predicting Adenosine to Inosine editing sites by using pseudo nucleotide compositions, the paper[4] we were following has an accuracy table given below:

Method	Sensitivity (%)	Specificity (%)	Accuracy (%)	MCC
Naïve Bayes	81.60	71.40	76.60	0.53
Bayes Net	81.60	69.70	75.80	0.52
J48	67.50	63.10	65.40	0.31
PAI	85.60	73.11	79.51	0.60

PAI obtained an accuracy of 79.51% with the sensitivity of 85.60%, specificity of 73.11% and in independent set, sensitivity of 82.33.

On the other hand, our procedure have reached the accuracy of 86.47% with a sensitivity of 95.20%, specificity of 77.30% and in independent set, sensitivity of 90.67%.

4. Conclusion

Although our experiment is like acting in anticipation, we have achieved a mesmerizing performance. We know very little about RNA codons and their impacts and correlations with RNA bases and their surroundings. This experiment may lead us to a new way how we study and experiment on genetic sequences.

5. References

1. St Laurent, G. et al. Genome-wide analysis of A-to-I RNA editing by single-molecule sequencing in Drosophila. Nature structural & molecular biology 20, 1333–1339, doi: 10.1038/nsmb.2675 (2013).
2. Yu, Y. et al. The Landscape of A-to-I RNA Editome Is Shaped by Both Positive and Purifying Selection. PLoS genetics 12, e1006191, doi: 10.1371/journal.pgen.1006191 (2016).
3. Eibe Frank, Mark A. Hall, and Ian H. Witten (2016). The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufmann, Fourth Edition, 2016.
4. PAI: Predicting adenosine to inosine editing sites by using pseudo nucleotide compositions Wei Chen1, Pengmian Feng, Hui Ding & Hao Lin.