Faculty of Mathematics and Information Science

WARSAW UNIVERSITY OF TECHNOLOGY

Bioinformatics

Project 2: Enhancer Classification Problem

Wysocka Patrycja
306022

December 5, 2024

1 Objective

This project aims to develop a classifier capable of predicting enhancer sequences using the frequency of k-mers in DNA sequences as input features. Enhancers are non-coding DNA sequences that play a critical role in regulating gene expression by facilitating interactions between transcription factors and gene promoters, often bridging vast genomic distances. Understanding enhancer sequences is vital for uncovering the mechanisms behind gene regulation, the physiological basis of diseases, and genetic variation among individuals and species. By identifying patterns in enhacer sequences such as Transcription Factor Binding Sites (TFBS), the classifier will provide insights into the functional motifs that enable transcriptional activation.

2 Data

The data for this project includes enhancer sequences and genomic DNA sequences, obtained from publicly available databases and processed into positive and negative samples.

- **Positive Data**: Extracted from file (`experiments.tsv.gz`), containing a total of **4,482 records**, of which **2,267** are marked as positive (`curation status = 'positive'`).
- **Negative Data (two variants)**:
    1. **From experiments**: Extracted from file (`experiments.tsv.gz`), containing a total of **1,913 sequences** (`curation status = 'negative'`).
    2. **Random Negatives**: Extracted from the *GRCh38 (hg38)* genome FASTA file downloaded from GENCODE. After preprocessing to exclude overlaps, ambiguous bases (e.g., 'N'), and to match sequence lengths, **2,132 valid random negatives** were included.

This processed dataset includes **2,267 positive** and **2,132 or 1,1913 negative sequences** (depending on variation), ensuring balance and relevance for classifier training.

3 Methods

3.1 Preprocessing Steps

The data preprocessing involved several key steps to prepare positive and negative enhancer sequences for classifier training:

1. **Translation of Genomic Records**: Genomic sequences in FASTA format were parsed and normalized to uppercase letters (A, T, C, G). Invalid sequences (e.g., empty or with translation errors) were filtered out.

2. **Extraction of Positive and Negative Enhancer Records**: Data from the VISTA Enhancer Browser (`experiments.tsv.gz`) was loaded, selecting the relevant columns: curation status, genomic coordinates, and sequences. Positive sequences were filtered based on `curation_status = "positive"` and negative ones on `curation_status = "negative"`.

3. **Generation of Random Negative Samples**: Random genomic regions of the same length as the positive enhancers were generated from the GRCh38 genome. These regions were checked for overlap with positive sequences and filtered for ambiguous bases (e.g., 'N').

4. **Canonicalization and K-mer Counting**: For each sequence, k-mers were computed, considering both the sequence and its reverse complement to select the canonical form. K-mer frequencies were normalized by sequence length.

5. **Preparation of Final Dataset**: Positive and negative sequences were combined, and k-mer frequency vectors were calculated for each sequence. Missing k-mers were added as columns with zero frequency to ensure a consistent feature space.

This preprocessing ensured the data was clean, biologically relevant, and ready for model training. Selected K for for K-mer counting was: 3, 4, 5.

3.2 Classifiers

For this project, I used three classifiers: **Random Forest**, **Naive Bayes**, and **XGBoost** to predict enhancer sequences based on k-mer frequencies.

Random Forest was chosen for its robustness and ability to handle high-dimensional data. As an ensemble method, it builds multiple decision trees and combines their predictions to improve accuracy. Random Forest is resistant to overfitting and provides feature importance, helping identify key DNA motifs in enhancer prediction.

Naive Bayes, a probabilistic classifier based on Bayes' theorem, was selected as a simple baseline model. Despite its simplicity, it is efficient with high-dimensional data and works well for tasks involving count-based features like k-mers.

XGBoost is a gradient boosting model known for its high performance in classification tasks. It builds decision trees sequentially, with each tree attempting

to correct the errors of the previous one. XGBoost is particularly effective with imbalanced datasets and offers regularization to prevent overfitting, making it well-suited for predicting enhancer sequences from complex genomic data.

All three models were used with their default configurations. Random Forest used 100 trees without a maximum depth, Naive Bayes was implemented using the Multinomial variant, and XGBoost utilized its default hyperparameters.

In summary, Random Forest was used for its robust performance and feature evaluation, Naive Bayes served as a simple baseline, and XGBoost was included for its high accuracy and ability to handle complex datasets.

4 Results

4.1 Tables

4.1.1 Results for models trained on `experiments`

Classifier	k	**Negative Data Type**	**Accuracy**	**AUC-ROC**
Naïve Bayes	3	Experiment Negatives	0.51375	0.5649
Naïve Bayes	3	Random Negatives	0.725	0.848
Naïve Bayes	4	Experiment Negatives	0.52625	0.5883
Naïve Bayes	4	Random Negatives	0.6625	0.7836
Naïve Bayes	5	Experiment Negatives	0.53375	0.5736
Naïve Bayes	5	Random Negatives	0.75375	0.8130
Random Forest	3	Experiment Negatives	0.5375	0.5647
Random Forest	3	Random Negatives	0.7275	0.7928
Random Forest	4	Experiment Negatives	0.50625	0.5546
Random Forest	4	Random Negatives	0.74875	0.8124
Random Forest	5	Experiment Negatives	0.525	0.5506
Random Forest	5	Random Negatives	0.75	0.8204
XGBoost	3	Experiment Negatives	0.52	0.5139
XGBoost	3	Random Negatives	0.6675	0.7148
XGBoost	4	Experiment Negatives	0.55625	0.5790
XGBoost	4	Random Negatives	0.62625	0.6825
XGBoost	5	Experiment Negatives	0.555	0.5801
XGBoost	5	Random Negatives	0.7175	0.7791

Table 1: Classifier performance metrics (Accuracy and AUC-ROC) for different configurations using Experiment Negatives for training.

4.1.2 Results for models trained on `ranom negatives`

Classifier	k	Negative Data Type	Accuracy	AUC-ROC
Naïve Bayes	3	Experiment Negatives	0.52875	0.5735
Naïve Bayes	3	Random Negatives	0.80875	0.8953
Naïve Bayes	4	Experiment Negatives	0.535	0.5740
Naïve Bayes	4	Random Negatives	0.82625	0.9121
Naïve Bayes	5	Experiment Negatives	0.53375	0.5782
Naïve Bayes	5	Random Negatives	0.84375	0.9247
Random Forest	3	Experiment Negatives	0.53875	0.5821
Random Forest	3	Random Negatives	0.80875	0.8984
Random Forest	4	Experiment Negatives	0.52875	0.5910
Random Forest	4	Random Negatives	0.8325	0.9155
Random Forest	5	Experiment Negatives	0.50875	0.5833
Random Forest	5	Random Negatives	0.815	0.9060
XGBoost	3	Experiment Negatives	0.5225	0.5614
XGBoost	3	Random Negatives	0.775	0.8829
XGBoost	4	Experiment Negatives	0.5475	0.5774
XGBoost	4	Random Negatives	0.82875	0.9198
XGBoost	5	Experiment Negatives	0.53	0.5935
XGBoost	5	Random Negatives	0.835	0.9207

Table 2: Classifier performance metrics (Accuracy and AUC-ROC) for different configurations using Random Negatives for training.

4.2 Comparison of performance on Negative Datasets

The evaluation highlights a significant disparity in classifier performance when comparing the two types of negative datasets:

1. **Tested records where `curation_status == 'negative'` (*Experiment Negatives*)**:
    - Models trained with these curated negatives consistently underperform compared to those trained with *Random Negatives*.
    - For **Naïve Bayes**, the **Accuracy** remains low (ranging between 51.3% and 55.6%) and **AUC-ROC** does not exceed 0.5883, regardless of k. This underscores the challenge of distinguishing curated negatives from positives due to their inherent similarity and less distinct features.

- Similar trends are observed for **Random Forest** and **XGBoost**. Despite being more advanced models, they achieve maximum **AUC-ROC** values of 0.5647 and 0.5801 respectively, indicating limited ability to separate these negatives from positives.

2. **Tested random Sequences Extracted from the Genome (*Random Negatives*)**:

   - Models trained with these negatives achieve significantly higher performance. For example:
     - **Naïve Bayes** reaches an **Accuracy** of up to 84.38% and **AUC-ROC** of 0.9247 for $k = 5$.
     - **Random Forest** and **XGBoost** show even better performance, with the latter achieving an **AUC-ROC** of 0.9207.
   - These results suggest that the preprocessing constraints for *Random Negatives* (e.g., non-overlap with positives, proper lengths, and removal of ambiguous bases) lead to patterns that are more distinct and separable, enhancing classification.

4.3 Effect of k

The parameter k, representing the number of neighbors or other context-based parameters, exerts a clear influence:

- **For *Random Negatives***:
  - As k increases, all classifiers generally exhibit improved performance, reflecting the stabilization of predictions with more contextual information.
  - The improvement is especially noticeable for **Naïve Bayes** and **Random Forest**, while **XGBoost** demonstrates the most substantial gains, with **AUC-ROC** peaking at 0.9207 for $k = 5$.
- **For *Experiment Negatives***:
  - The effect of k is less consistent, with marginal or no gains observed. This suggests that increasing k provides little value when the negatives are inherently difficult to distinguish from positives.

4.4 Classifier Comparisons

1. **Naïve Bayes**:
    - Performs moderately well with *Random Negatives*, achieving a maximum **Accuracy** of 84.38% and **AUC-ROC** of 0.9247 at $k = 5$.
    - Struggles with *Experiment Negatives*, with **Accuracy** peaking at 53.38% and **AUC-ROC** at 0.5883 for $k = 4$.
    - Its reliance on the independence assumption makes it less suited for datasets with subtle or complex correlations, as seen with *Experiment Negatives*.
2. **Random Forest**:
    - Exhibits improved performance compared to Naïve Bayes, especially with *Random Negatives*, achieving a maximum **AUC-ROC** of 0.9060 at $k = 5$.
    - For *Experiment Negatives*, it fares slightly better than Naïve Bayes, but its **AUC-ROC** remains low, peaking at 0.5821 for $k = 3$.
    - Its ensemble approach captures non-linear patterns and interactions, explaining its stronger performance with *Random Negatives*.
3. **XGBoost**:
    - Achieves the highest overall performance, especially with *Random Negatives*, reaching a maximum **Accuracy** of 83.50% and **AUC-ROC** of 0.9207 at $k = 5$.
    - With *Experiment Negatives*, it shows modest improvements over other classifiers but still achieves limited success, with maximum **Accuracy** and **AUC-ROC** of 55.63% and 0.5801, respectively.
    - The boosting mechanism and ability to model complex relationships make it the most robust choice when data provides clear separability.

4.5 Superior Performance with Random Negatives

Models trained on *Random Negatives* consistently outperformed those trained on *Experiment Negatives*. For example, **Naïve Bayes** achieved an **AUC-ROC** of 0.9247 and an **Accuracy** of 84.38% with *Random Negatives*, compared to 0.5883 and 53.38% with *Experiment Negatives*. Similar trends were observed for **Random Forest** and **XGBoost**.

This performance gap highlights the distinctiveness of *Random Negatives*, which are designed to avoid overlap with positives, making them easier for classifiers to separate. The results underscore the importance of carefully constructing negative datasets for effective classification.

5 Summary

The analysis reveals several key findings. Firstly, the use of **Random Negatives** significantly enhances classifier performance compared to **Experiment Negatives**. Among the classifiers evaluated, **XGBoost** demonstrates the highest effectiveness, leveraging its advanced boosting mechanisms to achieve superior results, particularly when paired with **Random Negatives**. Additionally, increasing k values consistently improves performance, especially with **Random Negatives**, although the extent of improvement varies across classifiers. In contrast, the poor performance observed with **Experiment Negatives** highlights the need for additional preprocessing or feature engineering to enhance model discrimination in such cases.