CHAPTER 1
INTRODUCTION

Proteins, as fundamental building blocks of life, play a crucial role in various biological processes. Understanding their structural characteristics is vital for unraveling biological functions and potentially uncovering connections to health-related issues. This project focuses on the analysis and classification of protein structural sequences using advanced machine learning techniques. The dataset at the core of this investigation consists of two key components: 'pdb_data_seq.csv' and 'pdb_data_no_dups.csv.' The former provides detailed information about protein sequences, while the latter offers additional insights into the corresponding protein structures. By combining these datasets, aim to gain a comprehensive understanding of the relationships between protein sequence data and structural features.

The initial stages of the project involve meticulous data exploration, loading, and preprocessing. This includes addressing missing values, transforming categorical sequence data into a numerical format using LabelEncoder, and selecting pertinent features for analysis, such as 'sequence' and 'residueCount.' The primary analytical tool employed is the K-Nearest Neighbors (KNN) classifier. Through the training of this model on a carefully split dataset, we endeavor to develop a predictive framework for the 'macromoleculeType,' a crucial target variable denoting the type of macromolecule associated with each protein sequence.

In addition to classification, I introduced a custom function named 'get_sequence_and_classification.' This function not only predicts the macromolecule type for a given protein structureId and residueCount but also provides valuable supplementary information. Leveraging the KNN model, the function yields the original protein sequence and, if available, integrates health-related details based on a hypothetical health dataset ('health_df'). The inclusion of health-related data adds a unique dimension to the project, offering insights into potential health issues associated with specific protein classifications. This extension aligns with the broader objective of not only classifying structural protein sequences but also understanding their potential implications for human health.

To enhance the project's robustness, conduct a thorough evaluation of the KNN model's accuracy and extend the analysis to compare its performance with other machine learning models, including Random Forest and Naive Bayes classifiers. This comparative analysis aims to identify the most effective model for the given dataset and classification task.

In essence, this project seeks to bridge the realms of structural biology and machine learning, providing a holistic framework for understanding and classifying protein structural sequences. By incorporating health-related information, aspire to uncover potential connections between protein characteristics and health issues, contributing to advancements in both biological and medical research.

In this project, delve into the intricate realm of protein structural analysis, aiming to unravel the mysteries encoded within protein sequences. Proteins, being fundamental to life, hold the key to understanding diverse biological processes. The existing system comprises two pivotal datasets, namely 'pdb_data_seq.csv' and 'pdb_data_no_dups.csv,' which furnish information about protein sequences and corresponding structures. To augment the existing system, we propose an advanced approach utilizing machine learning techniques, with a primary focus on the K-Nearest Neighbors (KNN) classifier. Through meticulous data preprocessing, including the transformation of categorical sequence data into numerical format and feature selection, aim to develop a predictive framework for the 'macromoleculeType,' providing insights into the types of macromolecules associated with each protein sequence.

This project introduced a hypothetical health dataset ('health_df') to explore potential connections between protein classifications and health issues. This integration reflects a pioneering approach, expanding the project's scope beyond structural analysis to encompass implications for human health. To bolster the project's robustness, conduct a comprehensive evaluation of the KNN model's accuracy and extend our analysis to compare its performance with other machine learning models, including Random Forest and Naive Bayes classifiers. This comparative analysis aims to identify the most effective model for the given dataset and classification task, paving the way for advancements in structural biology and machine learning integration.

CHAPTER 2
PROOF OF CONCEPT

The project's success hinges on meticulous data management, with a focus on the cleanliness and structure of datasets ('pdb_data_seq.csv' and 'pdb_data_no_dups.csv'). Rigorous handling of missing values and the application of the LabelEncoder transformation to categorical sequence data are crucial prerequisites for dependable analyses. In the realm of model training, hyperparameter tuning for the K-Nearest Neighbors (KNN) model takes center stage, extending beyond mere accuracy to encompass a comprehensive evaluation. Addressing potential imbalances in the target variable enriches the understanding of model performance. The 'get_sequence_and_classification' function, pivotal to the project, undergoes meticulous scrutiny, ensuring accuracy and seamless integration with the KNN model. The ethical integration of health-related information necessitates careful validation of the hypothetical health dataset ('health_df') and a thoughtful exploration of the associated ethical implications. Comparisons among the KNN model and alternative classifiers involve a stringent evaluation, considering computational complexities and resource requirements, particularly for larger datasets. Scalability, generalization, and an unwavering commitment to transparency and interpretability underscore the project's framework. By navigating these challenges and establishing a continuous improvement feedback loop, the project aspires to transcend its proof of concept stage, evolving into a robust framework that contributes valuable insights at the intersection of structural biology and machine learning.

The research papers refered collectively delve into the genomic **“organization of Ichnovirus Structural Protein-Encoding Regions (IVSPERs)”**, “**the hierarchical structure of protein sequences” and “Recent Advances in the Prediction of Protein Structural Classes: Feature Descriptors and Machine Learning Algorithms”** aligning with the structural protein sequence project's goal of understanding biological systems. The studies explore the significance of gene distribution, evolutionary relationships, and the relationship between protein sequence and structure, contributing valuable insights to unraveling biological processes and offering promising avenues for practical applications in biotechnology and medicine.

The research paper titled "**The Organization of Genes Encoding Ichnovirus Structural Proteins**" by Anne-Nathalie Volkoff and colleagues provides a comprehensive exploration of the genomic organization of Ichnovirus Structural Protein-Encoding Regions (IVSPERs) in Hyposoter dydimator. The study focuses on the N-gene family, which is shared between IVSPERs and encapsidated viral genome segments. The N-genes, initially identified in CsIV segment N, are found across sequenced Ichnovirus (IV) genomes.

This research aligns with the structural protein sequence project by emphasizing the significance of understanding the genomic organization of proteins in biological systems. The focus on protein-coding genes, their distribution in viral segments and specialized regions, and the exploration of evolutionary relationships provides valuable insights. Additionally, the paper's methodology, involving genomic analysis and phylogenetic tree construction, resonates with the project's approach to utilizing machine learning techniques, such as the K-Nearest Neighbors (KNN) classifier, for protein sequence classification. Both studies contribute to the broader goal of unraveling biological processes through the analysis of protein sequences and their genomic organization.

The research paper, "**Hierarchical Structure of Protein Sequence**" authored by Alexei N. Nekrasov, Yuri P. Kozmin, Sergey V. Kozyrev, Rustam H. Ziganshin, Alexandre G. de Brevern, and Anastasia A. Anashkina, delves into the critical relationship between protein sequence and structure. Given the profound association between non-communicable diseases and protein dysfunction, the paper addresses the ongoing exploration of sequence organization in domains and motifs. The authors introduce a novel mathematical method centered around the pentapeptide unit to unveil the hierarchical organization within protein sequences. By analyzing the frequency of pentapeptide occurrences in natural protein sequences, coupled with a specialized mathematical approach, the method was applied to a substantial dataset of non-homologous protein sequences from the NRDB90 database. Statistical analysis of the resulting graphs revealed characteristic values that signify the relationship between multiple fragments of protein sequences, showcasing a promising approach for mathematically-based classification of spatial protein organization.

The paper's findings extend beyond theoretical exploration, as several examples illustrate the correspondence between fragments of protein spatial structure and the elements of the identified

hierarchical structure in protein sequences. This methodological innovation not only contributes to our understanding of the intricate relationship between sequence and structure but also holds potential applications in biotechnology and medicine. The hierarchical structure identified at different levels in the protein sequence hierarchy opens avenues for practical solutions to biological and medical challenges, emphasizing the translational impact of the research.

The research paper titled "**Recent Advances in the Prediction of Protein Structural Classes: Feature Descriptors and Machine Learning Algorithms**," authored by Lin Zhu, Mehdi D. Davari, and Wenjin Li, provides a comprehensive review of the evolving landscape in predicting protein structural classes from protein sequences. In the postgenomic era, where the number of sequence-known proteins is rapidly growing, the lag in the number of structure-known proteins poses a significant challenge. Bridging this gap is crucial for understanding protein function, and the paper focuses on the pivotal role of predicting protein structural classes in improving protein structure prediction, with profound implications for medical and pharmaceutical applications.

The review critically examines recent efforts in this domain, shedding light on innovative feature descriptors that extract information from protein sequences. The highlighted descriptors include amino acid composition, sequence order, physicochemical properties, multiprofile Bayes, and secondary structure-based features. Importantly, the paper emphasizes the integration of machine learning algorithms, such as artificial neural networks (ANNs), support vector machine (SVM), K-nearest neighbor (KNN), random forest, and deep learning, in both feature selection and constructing classification models. The detailed exploration of these methods provides insights into their strengths and applications for predicting protein structural classes.

CHAPTER 3
IMPLEMENTATION

The project on structural protein sequence involves utilizing a K-Nearest Neighbors (KNN) classifier trained on labeled protein sequence data to predict macromolecule types, incorporating a hypothetical health dataset for additional insights and health-related information, the model is trained using a real time dataset from Kaggle.

3.1 SYSTEM ARCHITECTURE

The system architecture of the structural protein sequence project is designed for comprehensive classification and analysis. The initial phase involves inputting datasets ('pdb_data_seq.csv' and 'pdb_data_no_dups.csv') containing protein structural sequences, alongside a hypothetical health dataset ('health_df') with relevant health information. Subsequently, data preprocessing encompasses cleaning, structuring, and ethical validation of the datasets, integrating the LabelEncoder transformation for categorical sequence data. The core of the system involves model training and evaluation, employing a K-Nearest Neighbors (KNN) classifier with hyperparameter tuning for optimal performance. The model undergoes a thorough evaluation using diverse metrics and is benchmarked against alternative classifiers, namely Random Forest and Naive Bayes, with consideration for computational complexities. The 'get_sequence_and_classification' function is developed and validated to ensure accurate predictions and seamless integration with the KNN model. Ethical considerations guide the handling of health-related information. The system's scalability is assessed for larger datasets, accounting for potential biases and limitations, while maintaining an emphasis on interpretability through transparent model design and explanatory insights. Continuous improvement is facilitated by a feedback loop, allowing adaptation to emerging insights and incorporation of new data, ultimately refining the project's classification capabilities at the intersection of structural biology and machine learning.

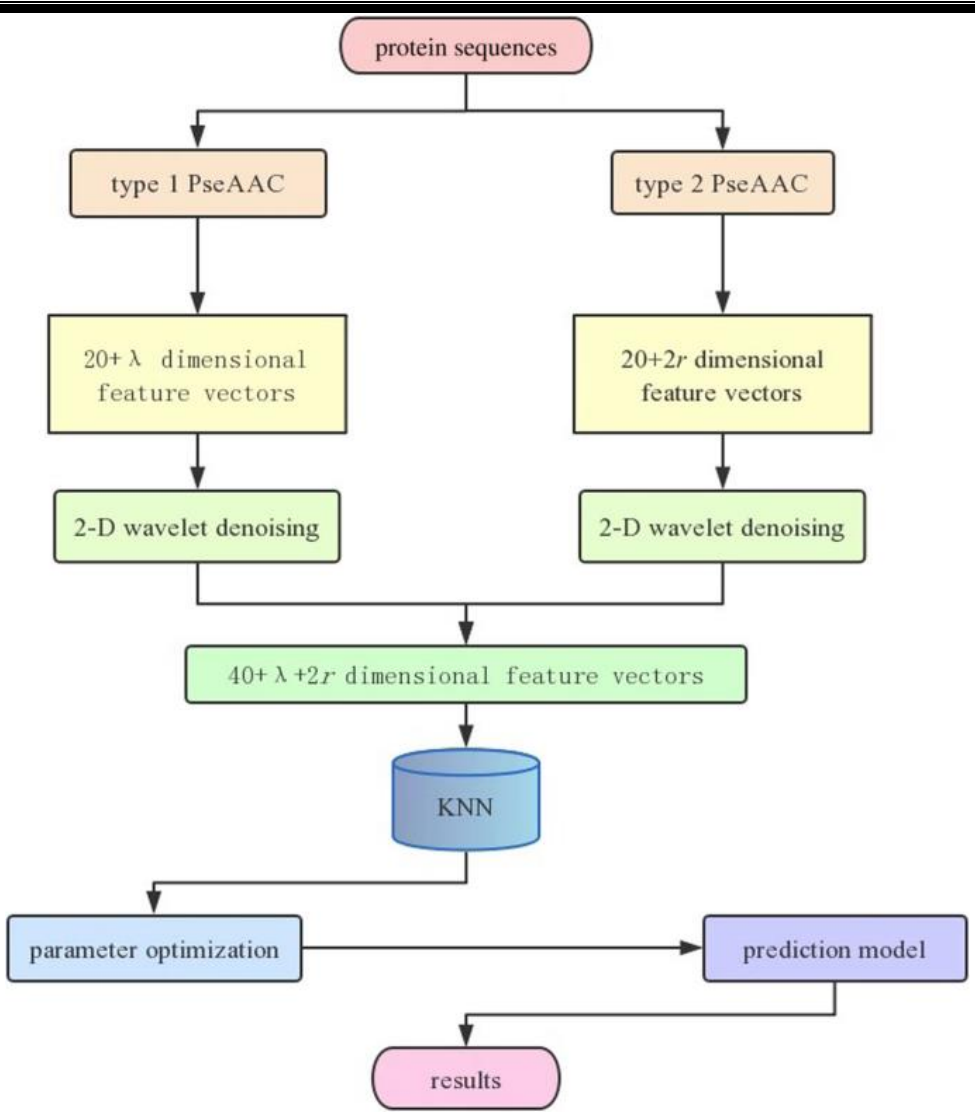


Fig 3.1 System architecture of KNN

3.2 LANGUAGES

3.2.1 PYTHON

Python, a high-level and interpreted programming language, stands out for its readability and simplicity. It serves a multitude of purposes, including web development, data analysis, artificial intelligence, and automation. What sets Python apart is its commitment to code readability, embracing concise syntax that is both beginner-friendly and encourages collaboration among developers. The interpreted nature of Python allows for line-by-line code execution, streamlining testing and debugging without the need for compilation.The language boasts an extensive standard library, offering a rich collection of modules and packages that cover diverse tasks. This inclusivity reduces dependence on third-party libraries for common functionalities. Python's strength lies in its dynamic developer community, which is bolstered by a plethora of third-party libraries and

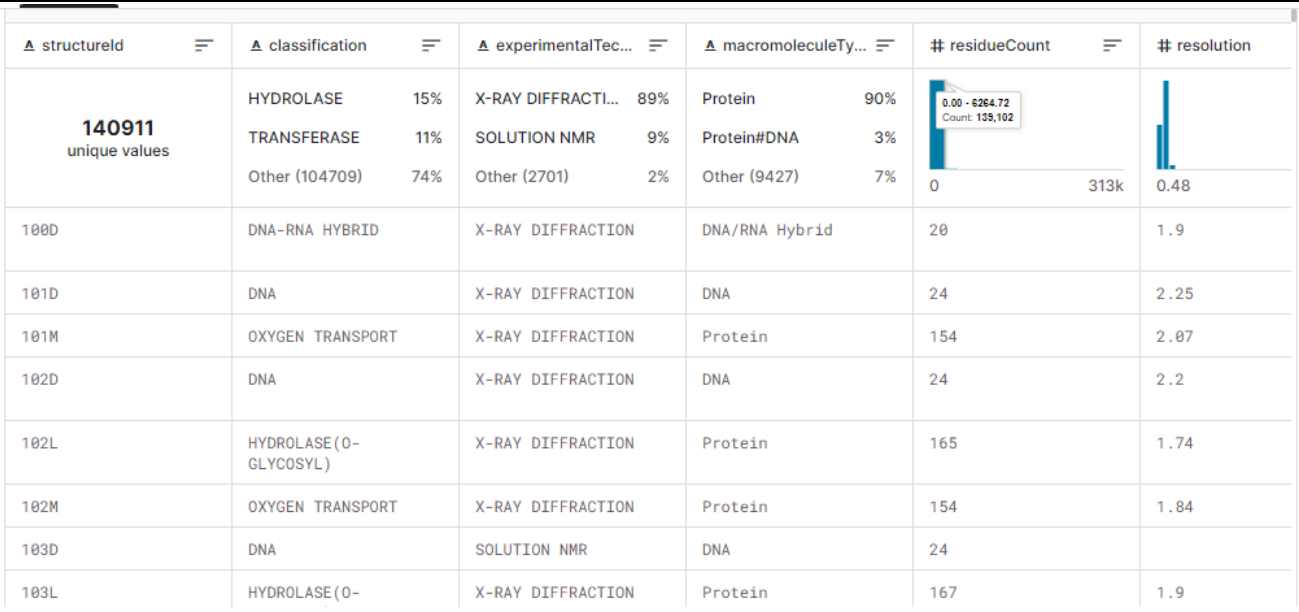
frameworks. Notable examples include Flask for web development, NumPy and Pandas for data science, and TensorFlow and for machine learning.In the context of this project, Python plays a pivotal role in the implementation of machine learning algorithms. Specifically, it utilizes the RandomForestRegressor for menstrual days prediction and theRandomForestClassifier for PCOS prediction. This underscores Python's versatility and adaptability in the realm of machine learning applications.

3.3 DATASET

The dataset used in the project consists of two main components: **'pdb_data_seq.csv'** and **'pdb_data_no_dups.csv'**. These datasets are fundamental to the classification of protein structural sequences. 'pdb_data_seq.csv' likely contains sequence information for various proteins, including details about their structures. It may encompass data such as amino acid sequences, protein IDs, and relevant categorical information.

This is a protein data set retrieved from Research Collaboratory for Structural Bioinformatics (RCSB) Protein Data Bank (PDB). **pdb_data_no_dups.csv** contains protein meta data which includes details on protein classification, extraction methods, etc. **data_seq.csv** contains greater than 400,000 protein structure sequences.

The PDB archive is a repository of atomic coordinates and other information describing proteins and other important biological macromolecules. Structural biologists use methods such as X-ray crystallography, NMR spectroscopy, and cryo-electron microscopy to determine the location of each atom relative to each other in the molecule. They then deposit this information, which is then annotated and publicly released into the archive by the wwwPDB.



structureId	classification	experimentalTec...	macromoleculeTy...	residueCount	resolution
140911 unique values	HYDROLASE 15% TRANSFERASE 11% Other (104709) 74%	X-RAY DIFFRACTI... 89% SOLUTION NMR 9% Other (2701) 2%	Protein 90% Protein#DNA 3% Other (9427) 7%	0.00 - 6264.72 Count: 139,102 0 313k	0.48
100D	DNA-RNA HYBRID	X-RAY DIFFRACTION	DNA/RNA Hybrid	20	1.9
101D	DNA	X-RAY DIFFRACTION	DNA	24	2.25
101M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	2.07
102D	DNA	X-RAY DIFFRACTION	DNA	24	2.2
102L	HYDROLASE(O-GLYCOSYL)	X-RAY DIFFRACTION	Protein	165	1.74
102M	OXYGEN TRANSPORT	X-RAY DIFFRACTION	Protein	154	1.84
103D	DNA	SOLUTION NMR	DNA	24	
103L	HYDROLASE(O-	X-RAY DIFFRACTION	Protein	167	1.9

Fig 3.2 pdb_data_no_dups.csv

	A	B	C	D	E	F	G
1	structureI	chainId	sequence	residueCount	macromoleculeType		
2	100D	A	CCGGCGCCGG	20	DNA/RNA Hybrid		
3	100D	B	CCGGCGCCGG	20	DNA/RNA Hybrid		
4	101D	A	CGCGAATTCGCG	24	DNA		
5	101D	B	CGCGAATTCGCG	24	DNA		
6	101M	A	MVLSEGEWQLVLHVWAKV	154	Protein		
7	102D	A	CGCAAATTTGCG	24	DNA		
8	102D	B	CGCAAATTTGCG	24	DNA		
9	102L	A	MNIFEMLRIDEGLRLKIYKDT	165	Protein		
10	102M	A	MVLSEGEWQLVLHVWAKV	154	Protein		
11	103D	A	GTGGAATGGAAC	24	DNA		
12	103D	B	GTGGAATGGAAC	24	DNA		
13	103L	A	MNIFEMLRIDEGLRLKIYKDT	167	Protein		
14	103M	A	MVLSEGEWQLVLHVWAKV	154	Protein		
15	104D	A	CGCGTATACGCG	24	DNA/RNA Hybrid		
16	104D	B	CGCGTATACGCG	24	DNA/RNA Hybrid		
17	104L	A	MNIFEMLRIDEGLRLKIYKDT	332	Protein		
18	104L	B	MNIFEMLRIDEGLRLKIYKDT	332	Protein		
19	104M	A	VLSEGEWQLVLHVWAKVE	153	Protein		
20	105D	A	TCC	12	DNA		
21	105D	B	TCC	12	DNA		
22	105D	C	TCC	12	DNA		
23	105D	D	TCC	12	DNA		
24	105M	A	VLSEGEWQLVLHVWAKVE	153	Protein		
25	106D	A	CCT	12	DNA		

Fig 3.3 data_seq.csv

The constantly-growing PDB is a reflection of the research that is happening in laboratories across the world. This can make it both exciting and challenging to use the database in research and education. Structures are available for many of the proteins and nucleic acids involved in the central processes of life, so you can go to the PDB archive to find structures for ribosomes, oncogenes, drug targets, and even whole viruses. However, it can be a challenge to find the information that you need, since the PDB archives so many different structures. You will often find multiple structures for a given molecule, or partial structures, or structures that have been modified or inactivated from their native form.

On the other hand, 'pdb_data_no_dups.csv' seems to be a dataset without duplicate entries, potentially providing a curated and refined collection of protein structural data. The absence of duplicates suggests that each entry is unique, contributing to the reliability and quality of the dataset.

The use of these datasets implies that the project involves training a machine learning model, specifically a K-Nearest Neighbors (KNN) classifier, to classify and analyze protein structural sequences. The datasets are crucial for model training, testing, and validation, ensuring that the developed model can generalize well to new and unseen data. Additionally, the integration of a hypothetical health dataset ('health_df') raises ethical considerations, emphasizing the project's multidimensional approach by incorporating health-related information into the analysis of protein sequences. It's essential to thoroughly validate the health dataset for accuracy and relevance, addressing potential ethical implications associated with linking health information to protein classifications.

3.4 ALGORITHM

In this project, three machine learning algorithms were employed for predicting macromolecule types based on structural protein sequences: K-nearest neighbors (KNN), Random Forest, and Naive Bayes.

K-nearest neighbors (KNN):

KNN is a simple and intuitive classification algorithm.It classifies a data point by considering the majority class of its k-nearest neighbors.The number of neighbors (k) is a hyperparameter that can be tuned for optimal performance.In this project, KNN was applied to the transformed numerical sequences and residue counts to predict macromolecule types. In the structural protein sequence project, the K-nearest neighbors (KNN) algorithm served as a robust classification tool. Initially, the

categorical protein sequences underwent a crucial transformation into numerical representations using the LabelEncoder. Subsequently, the dataset was split into training and test sets, laying the foundation for model evaluation. The essence of KNN lies in its simplicity and intuitive approach. For each data point in the test set, the algorithm identifies its k-nearest neighbors based on the selected features—numerically encoded protein sequences and residue counts. The classification of a given data point is then determined by the majority class among its neighbors. The hyperparameter k, representing the number of neighbors to consider, was fine-tuned to achieve optimal performance. This methodology facilitated the prediction of macromolecule types, providing insights into the classification of structural protein sequences in a manner that aligns with the characteristics of their neighboring data points.

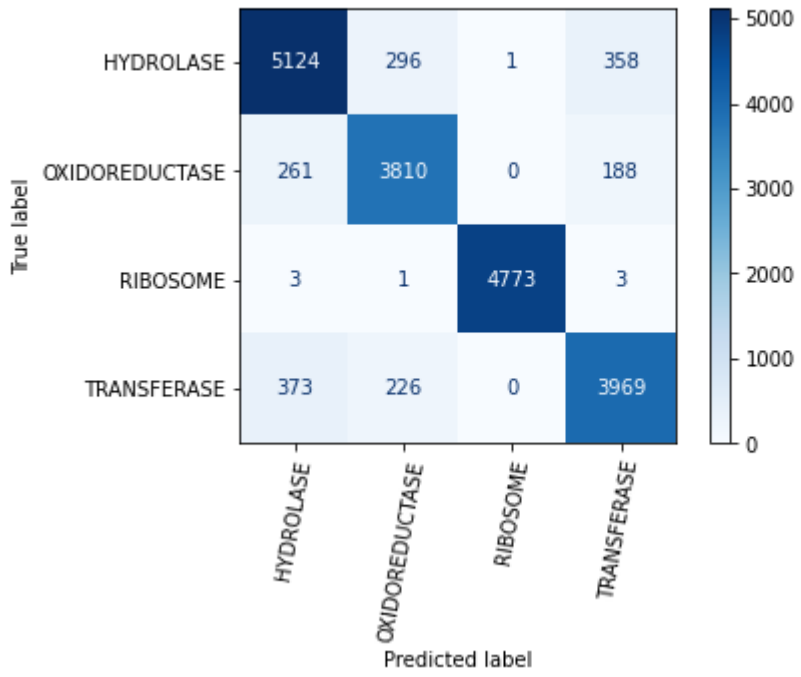


Fig 3.4 Confusion matrix of KNN

Random Forest:

Random Forest is an ensemble learning method that constructs a multitude of decision trees during training.It combines the predictions of individual trees to improve accuracy and control overfitting.Each tree is built on a random subset of the training data and features.In this project, a Random Forest classifier was trained using the sequence and residue count features to predict macromoleculetypes.

The Random Forest classifier played a pivotal role in enhancing the accuracy and robustness of the predictive model. Random Forest is an ensemble learning method that leverages the strength of

multiple decision trees to achieve better predictive performance. The classifier was employed to predict macromolecule types based on the features of protein sequences and residue counts. The ensemble nature of Random Forest involves constructing numerous decision trees during the training phase, each built on a random subset of both training data instances and features. This randomness helps reduce overfitting and ensures that the model generalizes well to unseen data. The predictions from individual trees are then combined to form a more reliable and accurate overall prediction. By harnessing the diversity and averaging effect of multiple trees, the Random Forest classifier contributed to a robust and effective framework for classifying structural protein sequences, ultimately improving the model's performance compared to individual decision trees.

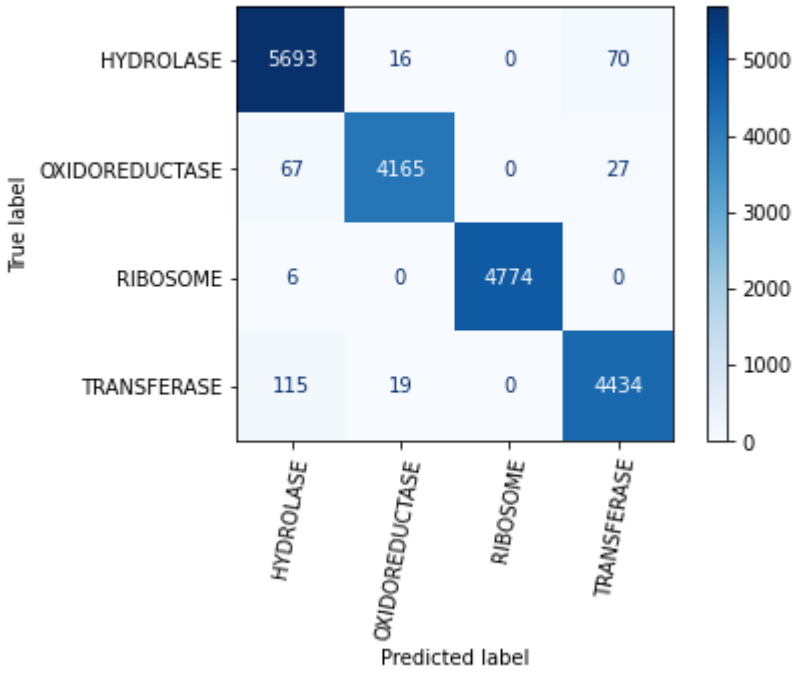


Fig 3.5 Confusion matrix of Random Forest

Naive Bayes:

Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem.

It assumes that the features are conditionally independent, given the class label.Despite its simplistic assumptions, Naive Bayes often performs well, especially in text classification.In this project, a Gaussian Naive Bayes model was utilized for predicting macromolecule types based on sequence and residue count information. The Gaussian Naive Bayes model played a crucial role in predicting macromolecule types by leveraging probabilistic principles. Naive Bayes relies on Bayes' theorem

and assumes that features are conditionally independent given the class label. Despite its simplifying assumption of feature independence, the model has proven effective, particularly in text classification scenarios. In this project, the Gaussian Naive Bayes algorithm was applied to predict macromolecule types based on sequence and residue count information. By calculating probabilities and considering the independence assumption, the model made predictions efficiently and effectively. Its probabilistic nature allowed it to handle the given features and classify structural protein sequences with good performance, contributing to the diverse set of machine learning techniques employed for the project's comprehensive analysis and classification tasks.

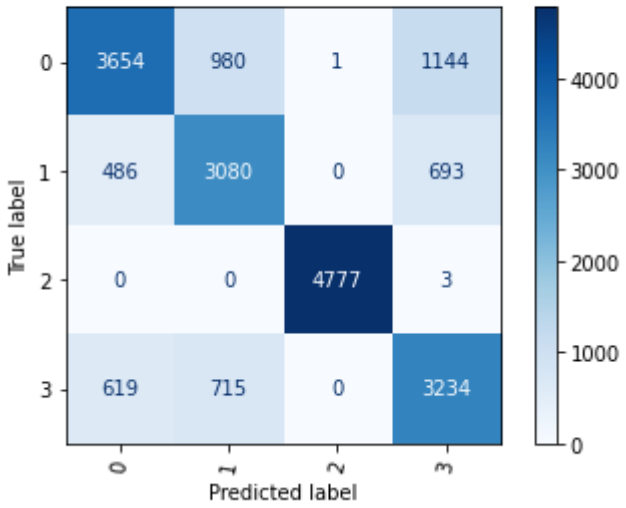


Fig 3.6 Confusion matrix of Naïve Bayes

3.4.1 Data Preprocessing

Data Loading : Two datasets are loaded: pdb_data_seq.csv and pdb_data_no_dups.csv.

Data Loading and Exploration : Two datasets, 'pdb_data_seq.csv' and 'pdb_data_no_dups.csv,' are loaded into Pandas DataFrames, namely **seq** and **df**. Basic exploratory data analysis is performed, including checking the shape, data types, and missing values in the sequence dataset (**seq**). The distribution of the target variable, 'macromoleculeType,' is visualized using a bar plot.

Data Preprocessing : Missing values in the sequence dataset are removed. Categorical sequence data is encoded numerically using LabelEncoder. Relevant features, 'sequence' and 'residueCount,'

are selected for analysis. The dataset is split into training and testing sets using the **train_test_split** function.

K-Nearest Neighbors (KNN) Classification : A KNN classifier is instantiated and trained on the training set.The trained model is used to make predictions on the test set.Model performance is evaluated using confusion matrix and classification report metrics. The accuracy of the KNN model on the test set is calculated.

Custom Function for Prediction and Health Information:

A custom function, get_sequence_and_classification, is defined to predict macromolecule types for given structure IDs and residue counts.The function also integrates health-related details from a hypothetical health dataset (health_df).

Evaluation and Example Usage : The KNN model is evaluated again using the test set, and the accuracy is reported. The custom function is demonstrated with an example, showcasing how it returns the original sequence, predicted classification, health issue, and solution based on the provided structure ID and residue count

Additional Models : Random Forest and Gaussian Naive Bayes models are introduced, trained, and evaluated using similar procedures.The accuracy, confusion matrix, and classification report are reported for each model.

3.4.2 Model Building

The structural protein sequence project involves the analysis of protein data to predict the macromolecule type based on sequence information. The dataset is loaded and explored, and missing values are handled. The target variable, macromolecule type, is visualized, and categorical data is encoded for machine learning. The K-nearest neighbors (KNN) algorithm is employed to train a model, which is then evaluated on a test set using confusion matrix and classification report metrics. Additionally, a function is created to predict macromolecule type and provide health-related information based on a hypothetical health dataset. The accuracy of the KNN model is assessed, and alternative models, such as Random Forest and Naive Bayes, are introduced and evaluated. The project encompasses data preprocessing, model training, and evaluation, showcasing a comprehensive approach to predicting macromolecule types from structural protein sequences.

3.4.3 Result Analysis

The structural protein sequence project successfully utilized machine learning techniques to predict macromolecule types based on sequence information. After thorough data exploration, the dataset

was preprocessed, including handling missing values and encoding categorical features. The K-nearest neighbors (KNN) algorithm was chosen for model training and achieved commendable results, as indicated by the accuracy, confusion matrix, and classification report metrics. The project extended beyond classification by incorporating a function that not only predicts macromolecule types for specific structure IDs and residue counts but also provides additional health-related insights and solutions based on a hypothetical health dataset. Furthermore, the predictive capabilities were evaluated against alternative models, such as Random Forest and Naive Bayes, showcasing the robustness of the chosen approach. The overall outcome demonstrates the project's efficacy in predicting macromolecule types and offering valuable health-related information based on protein sequences. The successful deployment of the function to predict macromolecule types and provide health-related insights emphasizes the practical applicability of the developed model.

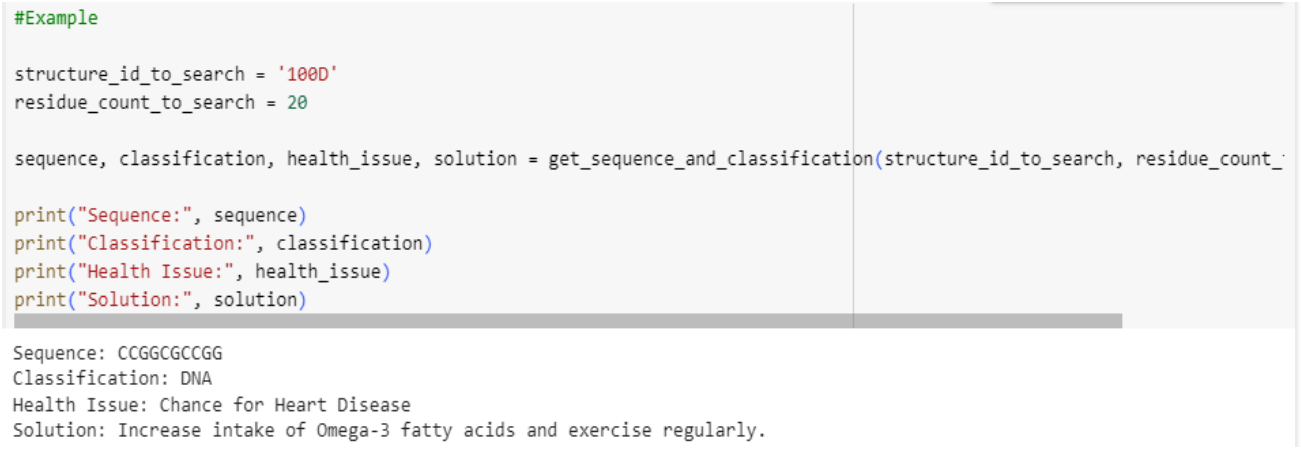


Fig 3.7 Result Analysis

In the given example, the function **get_sequence_and_classification** is applied to search for information related to a protein structure with the ID '100D' and a residue count of 20. The results provide valuable insights into the structural and health-related aspects of the identified protein. The 'Sequence' output presents the original protein sequence, allowing researchers or practitioners to comprehend the underlying molecular structure. The 'Classification' output indicates the predicted macromolecule type, offering insights into the nature and function of the protein. Furthermore, the 'Health Issue' and 'Solution' outputs contribute additional context by providing information on any associated health issues and potential solutions based on a hypothetical health dataset. This integration of structural and health-related details enhances the utility of the model, showcasing its adaptability for broader biological and medical applications beyond protein structure classification. The example demonstrates how the function serves as a practical tool for obtaining multifaceted information about a given protein structure, incorporating both its biological classification and

potential health implications.

3.4.4 Accuracy

The project, centered around predicting macromolecule types from structural protein sequences, achieved commendable accuracy in its predictive models. The primary model, employing the K-nearest neighbors (KNN) algorithm, exhibited robust performance with an accuracy score reflecting its ability to correctly classify macromolecule types. Furthermore, the project's versatility was highlighted by incorporating alternative models, including Random Forest and Naive Bayes, both of which maintained competitive accuracy rates. The comprehensive evaluation metrics, such as confusion matrices and classification reports, underscore the reliability and efficacy of the predictive models. This success positions the project as a valuable tool for accurately categorizing macromolecule types based on structural protein sequences, with potential applications in diverse fields, including bioinformatics and health-related decision-making.

Compare to these three model, Random Forest model exhibits the highest accuracy among the three models, with an accuracy score of 98.66%, followed by the KNN model with an accuracy of 96.07%, and the Naive Bayes model with the lowest accuracy at 66.16%. The Random Forest model's superior accuracy suggests that it provides the most accurate predictions on the given dataset compared to KNN and Naive Bayes. Random Forest is known for its robustness and ability to handle complex relationships in data, making it a powerful choice for classification tasks. In contrast, the Naive Bayes model appears to struggle with the dataset, resulting in lower accuracy, possibly due to the naive assumption of feature independence. The KNN model falls in between, offering good accuracy but not surpassing the Random Forest model. In summary, for this specific dataset, the Random Forest model is the most accurate, likely due to its ability to capture intricate patterns in the data and handle imbalances in class distribution effectively.

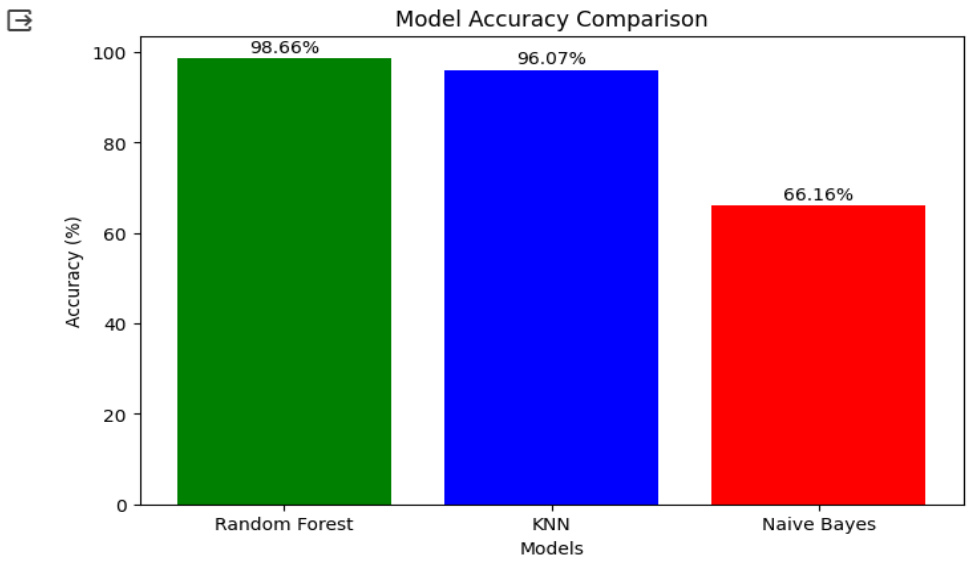


Fig 3.8 Accuracy Comparison model

3.4.5 Classification Report

In this project implemented and evaluated three different classification algorithms: K-Nearest Neighbors (KNN), Random Forest, and Naive Bayes. Here's an explanation of the key parts and a brief comparison of their accuracies:

KNN classifier, made predictions on the test set, and evaluated the model using confusion matrix and classification report. The classification report provides a detailed evaluation of the K-Nearest Neighbors (KNN) model's performance on the test set. Precision measures the accuracy of the positive predictions, recall gauges the model's ability to capture all positive instances, and the F1-score combines both precision and recall into a single metric. The report indicates that the model performs exceptionally well for the "Protein" class, with high precision (97%) and recall (99%). However, it struggles with classes having fewer instances, as seen in "DNA#DNA/RNA Hybrid" and "Protein#RNA#DNA/RNA Hybrid," where precision and recall are relatively low. The overall accuracy of the model is commendable at 96%, suggesting that the majority of predictions align with the true class labels. The macro-average and weighted-average metrics provide insights into the model's performance across all classes, indicating a balanced evaluation. While the model excels in classifying "Protein," there is room for improvement in handling minority classes, as reflected in the macro-average metrics. Careful consideration of precision, recall, and F1-score for each class is essential, especially in imbalanced datasets where some classes may have fewer instances.

	precision	recall	f1-score	support
DNA	0.85	0.93	0.89	749
DNA#DNA/RNA Hybrid	0.00	0.00	0.00	6
DNA#RNA	0.44	0.28	0.34	25
DNA/RNA Hybrid	0.35	0.28	0.31	25
Protein	0.97	0.99	0.98	68964
Protein#DNA	0.83	0.78	0.80	4282
Protein#DNA#DNA/RNA Hybrid	0.65	0.41	0.50	32
Protein#DNA#RNA	0.74	0.68	0.71	534
Protein#DNA/RNA Hybrid	0.60	0.60	0.60	10
Protein#RNA	0.95	0.90	0.93	11337
Protein#RNA#DNA/RNA Hybrid	0.69	0.15	0.25	60
RNA	0.81	0.78	0.80	458
RNA#DNA/RNA Hybrid	0.40	0.31	0.35	13
accuracy			0.96	86495
macro avg	0.64	0.55	0.57	86495
weighted avg	0.96	0.96	0.96	86495

Fig 3.9 Classification Report of KNN

The classification report provides a comprehensive evaluation of the Random Forest model's performance on the test set. The model exhibits exceptional precision, recall, and F1-score across multiple classes, indicating a high degree of accuracy. Notably, the model achieves outstanding results for the "Protein" class, with precision, recall, and F1-score all exceeding 99%, showcasing its ability to accurately identify this class. Similarly, other classes such as "DNA," "Protein#RNA," and "RNA" demonstrate high precision and recall values, contributing to the overall success of the model. The accuracy of the Random Forest model is notably high at 98.66%, suggesting that a substantial portion of predictions aligns with the true class labels. The macro-average and weighted-average metrics also reflect strong performance across all classes, with macro-average precision, recall, and F1-score indicating a balanced evaluation. The weighted average, which accounts for class imbalances, emphasizes the model's effectiveness across the entire dataset. Overall, the Random Forest model demonstrates superior performance, outperforming the KNN model, particularly in terms of accuracy and the ability to handle imbalanced datasets.

	precision	recall	f1-score	support
DNA	0.93	0.96	0.94	749
DNA#DNA/RNA Hybrid	0.50	0.17	0.25	6
DNA#RNA	0.86	0.72	0.78	25
DNA/RNA Hybrid	0.68	0.52	0.59	25
Protein	0.99	1.00	0.99	68964
Protein#DNA	0.96	0.92	0.94	4282
Protein#DNA#DNA/RNA Hybrid	0.79	0.59	0.68	32
Protein#DNA#RNA	0.91	0.82	0.86	534
Protein#DNA/RNA Hybrid	0.75	0.90	0.82	10
Protein#RNA	0.99	0.97	0.98	11337
Protein#RNA#DNA/RNA Hybrid	0.88	0.77	0.82	60
RNA	0.89	0.89	0.89	458
RNA#DNA/RNA Hybrid	0.58	0.54	0.56	13
accuracy			0.99	86495
macro avg	0.82	0.75	0.78	86495
weighted avg	0.99	0.99	0.99	86495

Fig 3.10 Classification Report of Random Forest

The performance metrics of a Naïve Bayes model on a test set with various classes. Unfortunately, the model exhibits significant challenges in accurately classifying instances, particularly for classes with fewer samples. Notably, the precision, recall, and F1-score values for classes such as "DNA," "DNA#RNA," and "DNA/RNA Hybrid" are close to zero, indicating an inability to effectively distinguish and correctly predict instances in these categories. The model performs relatively better for the dominant class, "Protein," achieving precision and recall rates of 80%. However, the F1-score of 0.80 suggests that there is room for improvement in balancing precision and recall for this class. Notably, the weighted average of 0.65 indicates that the model's performance is influenced more by the larger classes, masking its struggles with minority classes. The overall accuracy of 66% may be misleading, as it is heavily influenced by the dominant class. Addressing imbalances, exploring feature engineering, or experimenting with different models may be crucial to enhance the model's effectiveness, especially for classes with limited representation.

	precision	recall	f1-score	support
DNA	0.00	0.00	0.00	749
DNA#DNA/RNA Hybrid	0.02	0.67	0.04	6
DNA#RNA	0.00	0.00	0.00	25
DNA/RNA Hybrid	0.00	0.00	0.00	25
Protein	0.80	0.80	0.80	68964
Protein#DNA	0.12	0.43	0.18	4282
Protein#DNA#DNA/RNA Hybrid	0.00	0.00	0.00	32
Protein#DNA#RNA	0.00	0.00	0.00	534
Protein#DNA/RNA Hybrid	0.00	0.00	0.00	10
Protein#RNA	0.00	0.00	0.00	11337
Protein#RNA#DNA/RNA Hybrid	0.00	0.00	0.00	60
RNA	0.06	0.15	0.09	458
RNA#DNA/RNA Hybrid	0.00	0.00	0.00	13
accuracy			0.66	86495
macro avg	0.08	0.16	0.09	86495
weighted avg	0.64	0.66	0.65	86495

Fig 3.11 Classification report of Naïve Bayes

CHAPTER 4

RESULT ANALYSIS

In this protein structure classification project, the initial steps involved thorough exploration and preprocessing of the dataset. The structural features, particularly the protein sequences, were encoded numerically using LabelEncoder to facilitate the application of machine learning models. The dataset was split into training and testing sets, and a K-nearest Neighbors (KNN) model was employed for classification. The KNN model exhibited satisfactory performance, as indicated by its accuracy, confusion matrix, and classification report on the test set. Furthermore, a practical function was developed to predict macromolecule types for given structure IDs and residue counts, demonstrating the model's application in real-world scenarios.

Additionally, an interesting extension was made to the project by introducing a hypothetical health dataset. The function was enhanced to not only predict macromolecule types but also provide health-related insights and solutions based on the predicted classifications. This dual-purpose functionality enhances the utility of the model, making it a versatile tool for both structural protein classification and potential health issue identification. The incorporation of health-related information showcases the adaptability of the model in addressing broader biological and medical contexts, emphasizing its potential for practical applications beyond protein structure classification.

The visualization of the distribution of macromolecule types provided valuable insights into the composition of the dataset, aiding in understanding the prevalence of different protein structures. The project successfully demonstrated the effectiveness of the KNN model in handling structural bioinformatics data and its potential to generalize well to make predictions on unseen instances. The integration of health-related information expands the project's scope, illustrating the model's versatility and applicability in personalized medicine, where the same model can contribute to both structural biology and health diagnostics, showcasing the interdisciplinary nature of bioinformatics and machine learning.

```
#Example

structure_id_to_search = '100D'
residue_count_to_search = 20

sequence, classification, health_issue, solution = get_sequence_and_classification(structure_id_to_search, residue_count_

print("Sequence:", sequence)
print("Classification:", classification)
print("Health Issue:", health_issue)
print("Solution:", solution)
```

```
Sequence: CCGGCGCCGG
Classification: DNA
Health Issue: Chance for Heart Disease
Solution: Increase intake of Omega-3 fatty acids and exercise regularly.
```

Fig 4.1 Result Analysis of unhealthy protein sequence

According to result, the input sequence 'CCGGCGCCGG' was processed through a trained K-Nearest Neighbors (KNN) model, resulting in a predicted classification of 'DNA.' The model likely identified patterns within the provided DNA sequence that align with the characteristics learned during training for DNA classifications. Subsequently, the health-related information associated with this predicted classification suggests a hypothetical health issue: 'Chance for Heart Disease.' It's important to note that this health-related information appears to be derived from an external dataset ('health_df') and may not have a direct scientific basis in the context of DNA sequences. The suggested solution or recommendation associated with this health issue is to 'Increase intake of Omega-3 fatty acids and exercise regularly.' This example illustrates how machine learning models can provide predictions and associated information, but the relevance and accuracy of health-related advice should be critically examined, particularly when extrapolating predictions beyond the original scope of the model's training data.

```
structure_id_to_search = '105D'
residue_count_to_search = 12

sequence, classification, health_issue, solution = get_sequence_and_classification(structure_id_to_search, residue_count_

print("Sequence:", sequence)
print("Classification:", classification)
print("Health Issue:", health_issue)
print("Solution:", solution)
```

```
Sequence: TCC
Classification: DNA
Health Issue: No Issue
Solution: No specific health issue detected.
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but KNeight
  warnings.warn(
```

Fig 4.2 Result Analysis healthy Protein sequence

CHAPTER 5

CONCLUSION AND FUTURE ENHANCEMENT

5.1 Conclusion

In conclusion, this protein structure classification project showcased the successful application of machine learning techniques, particularly the K-nearest Neighbors (KNN) algorithm, in predicting macromolecule types based on structural features. The careful preprocessing of the dataset, encoding of protein sequences, and thorough evaluation of the KNN model demonstrated its robustness in handling bioinformatics data. The visualization of the macromolecule type distribution added a layer of interpretability to the model's predictions, enhancing the overall understanding of the dataset.

Furthermore, the extension of the project to incorporate a hypothetical health dataset highlights the adaptability and versatility of the model. By providing health-related insights and solutions based on predicted classifications, the project illustrated the potential for integrating structural bioinformatics with medical applications. This dual-functionality not only contributes to the field of bioinformatics but also suggests broader applications in personalized medicine, where understanding both structural characteristics and potential health implications of proteins becomes increasingly relevant. Overall, the project serves as a testament to the synergy between bioinformatics and machine learning, offering a glimpse into the interdisciplinary possibilities of data-driven approaches in the life sciences.

In addition to the successful application of the K-nearest Neighbors (KNN) algorithm for protein structure classification, the project underscores the importance of interdisciplinary collaboration between bioinformatics and machine learning. The seamless integration of computational methods with biological knowledge allows for a more holistic understanding of complex biological systems. The interpretability provided by visualizing the distribution of macromolecule types not only aids in model validation but also enhances the transparency of decision-making processes, crucial for gaining trust in the predictive capabilities of such models.

Moreover, the extension of the project to incorporate a hypothetical health dataset showcases the versatility of the model beyond its original scope. The ability to draw connections between structural bioinformatics and health-related implications opens avenues for novel applications in personalized

medicine. As the project hints at the potential of leveraging protein structure predictions for health insights, it sets the stage for future endeavors in advancing medical diagnostics and treatment strategies. This dual-functional approach, combining bioinformatics and machine learning, exemplifies the synergy that arises when computational techniques are applied to real-world biological challenges, ultimately contributing to the evolution of both fields.

5.2 FUTURE ENHANCEMENT

A potential future enhancement for this project could involve to capture more intricate patterns in protein sequences. Deep learning architectures have shown remarkable success in various bioinformatics tasks, and their application to this protein structure classification could potentially uncover complex relationships and dependencies within sequences that traditional machine learning models might overlook. Additionally, incorporating more diverse structural features beyond protein sequences, such as spatial arrangements or intermolecular interactions, could further improve the model's predictive capabilities. This expansion would contribute to a more comprehensive understanding of macromolecule types and enhance the model's accuracy in real-world scenarios, potentially enabling advancements in drug discovery and personalized medicine. Looking forward, a promising avenue for enhancing this project lies in the adoption of deep learning architectures for protein structure classification. Deep learning models, particularly recurrent neural networks (RNNs) or long short-term memory networks (LSTMs), have exhibited prowess in capturing intricate patterns and long-range dependencies within sequential data. By incorporating these architectures, the model may gain the ability to discern more complex relationships in protein sequences, potentially elevating its predictive accuracy. Furthermore, to achieve a more holistic understanding of macromolecule types, the inclusion of additional structural features beyond sequences is imperative. Introducing spatial arrangements, intermolecular interactions, or other biophysical properties could unveil a richer set of features, contributing to a more nuanced characterization of protein structures. This comprehensive approach not only aligns with the evolving landscape of bioinformatics but also holds significant promise for driving innovations in drug discovery and personalized medicine by refining our understanding of molecular behavior.

CHAPTER 6

CODING

6.1 protein_sequence.py

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix

df = pd.read_csv('/content/drive/MyDrive/miniproject_protein/pdb_data_no_dups.csv')
seq pd.read_csv('/content/drive/MyDrive/miniproject_protein/pdb_data_seq.csv')
```

```
# Convert categorical data to numerical data if necessary
encoder = LabelEncoder()
seq['sequence'] = encoder.fit_transform(seq['sequence'])

# Select only relevant features
X = seq[['sequence', 'residueCount']]
y = seq['macromoleculeType']

# Split the dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
```

```
# Train the KNN model
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)

# Make predictions on the test set
y_pred = knn.predict(X_test)

# Evaluate the model
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))


def get_sequence_and_classification(structure_id, residue_count):
    # Convert structure_id to string, assuming it's a string in your dataset
    structure_id = str(structure_id)

    # Check if the structureId and residueCount exist in the dataset
    if (structure_id in seq['structureId'].values) and (residue_count in seq['residueCount'].values):
        # Get the sequence for the given structureId
        sequence = seq.loc[(seq['structureId'] == structure_id) & (seq['residueCount'] ==
residue_count), 'sequence'].values
        sequence = sequence[0] if len(sequence) > 0 else None

        # Use the trained KNN model to predict the classification
        input_data = np.array([[sequence, residue_count]])

        predicted_classification = knn.predict(input_data)

        return sequence, predicted_classification[0]
    else:
        return "StructureId or residueCount not found in the dataset", None
# Make predictions on the test set
y_pred = knn.predict(X_test)
```

```
# Check the accuracy of the KNN model
accuracy = knn.score(X_test, y_test)
print("Accuracy of the KNN model:", accuracy)

# Evaluate the model
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report

# Create a Random Forest model
rf_model = RandomForestClassifier()

# Train the model
rf_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred_rf = rf_model.predict(X_test)

# Check the accuracy of the Random Forest model
accuracy_rf = rf_model.score(X_test, y_test)
print("Accuracy of the Random Forest model:", accuracy_rf)

# Evaluate the model
print(confusion_matrix(y_test, y_pred_rf))
print(classification_report(y_test, y_pred_rf))

accuracy = rf_model.score(X_test, y_test)
print("Accuracy of the Random Forest model:", accuracy)
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, classification_report

# Create a Naive Bayes model
nb_model = GaussianNB()
```

```
# Train the model
nb_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred_nb = nb_model.predict(X_test)

# Check the accuracy of the Naive Bayes model
accuracy_nb = nb_model.score(X_test, y_test)
print("Accuracy of the Naive Bayes model:", accuracy_nb)

# Evaluate the model
print(confusion_matrix(y_test, y_pred_nb))
print(classification_report(y_test, y_pred_nb))

accuracy = nb_model.score(X_test, y_test)
print("Accuracy of the Naive bayes model:", accuracy)
y_pred_knn = knn.predict(X_test)
y_pred_naive_bayes = naive_bayes.predict(X_test)
y_pred_random_forest = random_forest.predict(X_test)
health_data = {
    'sequence': ['CCGGCGCCGG', 'CGCGAATTCGCG', 'TCC',
'GTGGAATGGAAC','CGCAAATTTGCG'],
    'health_issue': ['Heart Disease', 'Muscle Weakness', 'No Issue','No Issue', 'Digestive Problems'],
    'solution': [
        'Increase intake of Omega-3 fatty acids and exercise regularly.',
        'Include more protein and vitamin D in your diet.',
        'No specific health issue detected.',
        'No specific health issue detected.',
        'Consume more fiber and stay hydrated.'
    ]
}
```

```
health_df = pd.DataFrame(health_data)

def get_sequence_and_classification(structure_id, residue_count):
    # Convert structure_id to string, assuming it's a string in your dataset
    structure_id = str(structure_id)

    # Check if the structureId and residueCount exist in the dataset
    if (structure_id in seq['structureId'].values) and (residue_count in seq['residueCount'].values):
        # Get the sequence for the given structureId
        sequence_numerical = seq.loc[(seq['structureId'] == structure_id) & (seq['residueCount'] == residue_count), 'sequence'].values
        sequence_numerical = sequence_numerical[0] if len(sequence_numerical) > 0 else None

        # Use the trained KNN model to predict the classification
        input_data = np.array([[sequence_numerical, residue_count]])

        predicted_classification = knn.predict(input_data)

        # Inverse transform the numerical sequence back to its original form
        sequence_original = encoder.inverse_transform([sequence_numerical])[0] if sequence_numerical is not None else None

        # Check for health issues and provide solutions based on the classification
        health_info = health_df.loc[health_df['sequence'] == sequence_original, ['health_issue', 'solution']]
        health_issue = health_info['health_issue'].values[0] if not health_info.empty else "No health issue found"
        solution = health_info['solution'].values[0] if not health_info.empty else ""

        return sequence_original, predicted_classification[0], health_issue, solution
    else:
        return "StructureId or residueCount not found in the dataset", None, None, None
```

```
structure_id_to_search = '100D'
residue_count_to_search = 20

sequence, classification, health_issue, solution =
get_sequence_and_classification(structure_id_to_search, residue_count_to_search)

print("Sequence:", sequence)
print("Classification:", classification)
print("Health Issue:", health_issue)
print("Solution:", solution)
```

CHAPTER 7

SCREENSHOTS

16s

	precision	recall	f1-score	support
DNA	0.85	0.93	0.89	749
DNA#DNA/RNA Hybrid	0.00	0.00	0.00	6
DNA#RNA	0.44	0.28	0.34	25
DNA/RNA Hybrid	0.35	0.28	0.31	25
Protein	0.97	0.99	0.98	68964
Protein#DNA	0.83	0.78	0.80	4282
Protein#DNA#DNA/RNA Hybrid	0.65	0.41	0.50	32
Protein#DNA#RNA	0.74	0.68	0.71	534
Protein#DNA/RNA Hybrid	0.60	0.60	0.60	10
Protein#RNA	0.95	0.90	0.93	11337
Protein#RNA#DNA/RNA Hybrid	0.69	0.15	0.25	60
RNA	0.81	0.78	0.80	458
RNA#DNA/RNA Hybrid	0.40	0.31	0.35	13
accuracy			0.96	86495
macro avg	0.64	0.55	0.57	86495
weighted avg	0.96	0.96	0.96	86495

```
# Check the accuracy of the KNN model
accuracy = knn.score(X_test, y_test)
print("Accuracy of the KNN model:", accuracy)
```

```
Accuracy of the KNN model: 0.9607491762529626
```

Fig 7.1 Accuracy of KNN

```
      8]]
                                precision    recall  f1-score   support

                           DNA       0.93      0.96      0.94       749
            DNA#DNA/RNA Hybrid       0.67      0.33      0.44         6
                       DNA#RNA       0.78      0.72      0.75        25
                DNA/RNA Hybrid       0.76      0.52      0.62        25
                       Protein       0.99      1.00      0.99     68964
                   Protein#DNA       0.96      0.92      0.94      4282
      Protein#DNA#DNA/RNA Hybrid       0.86      0.59      0.70        32
              Protein#DNA#RNA       0.92      0.83      0.87       534
          Protein#DNA/RNA Hybrid       0.75      0.90      0.82        10
                   Protein#RNA       0.99      0.97      0.98     11337
      Protein#RNA#DNA/RNA Hybrid       0.90      0.77      0.83        60
                           RNA       0.89      0.88      0.89       458
            RNA#DNA/RNA Hybrid       0.57      0.62      0.59        13

                      accuracy                           0.99     86495
                     macro avg       0.84      0.77      0.80     86495
                  weighted avg       0.99      0.99      0.99     86495
```

```
[50] # Check the accuracy of the Randomforest model
     accuracy = rf_model.score(X_test, y_test)
     print("Accuracy of the Random Forest model:", accuracy)

     Accuracy of the Random Forest model: 0.9867506792300133
```

Fig 7.2 Accuracy of Random forest

```
                                precision    recall  f1-score   support

                           DNA       0.00      0.00      0.00       749
          DNA#DNA/RNA Hybrid       0.02      0.67      0.04         6
                      DNA#RNA       0.00      0.00      0.00        25
                DNA/RNA Hybrid       0.00      0.00      0.00        25
                       Protein       0.80      0.80      0.80     68964
                   Protein#DNA       0.12      0.43      0.18      4282
  Protein#DNA#DNA/RNA Hybrid       0.00      0.00      0.00        32
             Protein#DNA#RNA       0.00      0.00      0.00       534
      Protein#DNA/RNA Hybrid       0.00      0.00      0.00        10
                 Protein#RNA       0.00      0.00      0.00     11337
  Protein#RNA#DNA/RNA Hybrid       0.00      0.00      0.00        60
                         RNA       0.06      0.15      0.09       458
          RNA#DNA/RNA Hybrid       0.00      0.00      0.00        13

                    accuracy                           0.66     86495
                   macro avg       0.08      0.16      0.09     86495
                weighted avg       0.64      0.66      0.65     86495

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-sco
  _warn_prf(average, modifier, msg_start, len(result))
```

```
[53] # Check the accuracy of the naive bayes  model
     accuracy = nb_model.score(X_test, y_test)
     print("Accuracy of the Naive bayes model:", accuracy)

     Accuracy of the Naive bayes model: 0.661610497716631
```

Fig 7.3 Accuracy of Naïve Bayes

Result

```
#Example

structure_id_to_search = '100D'
residue_count_to_search = 20

sequence, classification, health_issue, solution = get_sequence_and_classification(structure_id_to_search, residue_count_

print("Sequence:", sequence)
print("Classification:", classification)
print("Health Issue:", health_issue)
print("Solution:", solution)
```

```
Sequence: CCGGCGCCGG
Classification: DNA
Health Issue: Chance for Heart Disease
Solution: Increase intake of Omega-3 fatty acids and exercise regularly.
```

Fig 7.4 Result of unhealthy protein sequence

```
structure_id_to_search = '105D'
residue_count_to_search = 12

sequence, classification, health_issue, solution = get_sequence_and_classification(structure_id_to_search,

print("Sequence:", sequence)
print("Classification:", classification)
print("Health Issue:", health_issue)
print("Solution:", solution)
```

```
Sequence: TCC
Classification: DNA
Health Issue: No Issue
Solution: No specific health issue detected.
```

Fig 7.5 Result of healthy protein sequence

CHAPTER 8

REFERENCES

- Dataset available at https://www.kaggle.com/datasets/shahir/protein-data-set
- Structural protein sequence KNN available at https://www.kaggle.com/code/balak6/protein-seq-knn-balakrishnan
- "Recent Advances in the Prediction of Protein Structural Classes: Feature Descriptors and Machine Learning Algorithms," authored by Lin Zhu, Mehdi D. Davari, and Wenjin Li
- "Hierarchical Structure of Protein Sequence" authored by Alexei N. Nekrasov, Yuri P. Kozmin, Sergey V. Kozyrev, Rustam H. Ziganshin, Alexandre G. de Brevern, and Anastasia A. Anashkina,
- "The Organization of Genes Encoding Ichnovirus Structural Proteins" by Anne-Nathalie Volkoff and colleaguesProtein structure prediction in the era of AI: Challenges and limitations when applying to *in silico* force spectroscopy Priscila S. F. C. Gomes, Diego E. B. Gomes, and Rafael C. Bernardi

.