**ProteinLocRF: Protein Localization Prediction Pipeline**

**Project Summary**

ProteinLocRF is a machine learning pipeline designed to predict the cellular localization of proteins based on their physicochemical properties. The pipeline processes protein sequences from a FASTA file, extracts features, assigns synthetic labels, and trains a Random Forest Classifier to predict localization. It consists of three Python scripts:

* **extract\_features.py**: Extracts features and assigns synthetic labels.
* **model.py**: Trains and evaluates the Random Forest Classifier.
* **main.py**: Orchestrates the pipeline execution.

The pipeline includes:

* **Feature Extraction**: Computes molecular weight, isoelectric point, aromaticity, instability index, and GRAVY.
* **Label Assignment**: Assigns synthetic labels (cytoplasm, nucleus, mitochondria) for simulation.
* **Model Training**: Uses a Random Forest Classifier with standardized features.
* **Evaluation**: Measures performance with accuracy and a classification report.
* **Output**: Saves features to proteinfeatures.csv and logs steps to outputlogging.log.

**Overall Logic and Workflow**

The pipeline has a proper workflow to process protein sequences and predict their cellular localization. It has 3 scripts. Script summaries are:

**1. extract\_features.py**

This script processes a FASTA file to extract physicochemical features for each protein sequence using ProteinAnalysis module from Biopython. Functions:

* **Feature Extraction**:
  + Extracting molecular weight, isoelectric point, aromaticity, instability index, and GRAVY.
  + Storing features in a pandas DataFrame. Saving the output to proteinfeatures.csv
* **Label Assignment**:
  + Assigns random synthetic labels (cytoplasm, nucleus, mitochondria) using a function.
* **Error Handling**:
  + Try-except blocks are used to handle file not found and other errors.

**2. model.py**

This script defines a function to train and evaluate a Random Forest Classifier. Components are:

* **Data Preparation**:
  + Takes a DataFrame with features and labels, splitting it into X features and y labels.
  + Splits data into training (80%) and testing (20%) sets.
* **Preprocessing**:
  + Uses StandardScalar to Standardize features.
* **Model Training**:
  + Trains Random Forest Classifier with 100 trees
* **Evaluation**:
  + Accuracy and classification report are generated.

**3. main.py**

This script serves as the pipeline’s entry point, coordinating the execution of feature extraction, label assignment, and model training. Tasks include:

* **Input**:
  + Specifies the input FASTA file.
* **Function Calling**:
  + Extract features and assign values functions are called.
  + Calls model to train and evaluate the Random Forest Classifier.
* **Logging**:
  + Logs execution steps to output.log for traceability.

**Classification Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Precision** | **Recall** | **F1-Score** | **Support** |
| Cytoplasm | 0.00 | 0.00 | 0.00 | 3 |
| Mitochondria | 0.50 | 0.67 | 0.57 | 3 |
| Nucleus | 0.60 | 0.75 | 0.67 | 4 |

* **Overall Accuracy**: 0.50 (50% of test samples correctly classified).
* **Macro Average**:
  + Precision: 0.37
  + Recall: 0.47
  + F1-Score: 0.41
* **Weighted Average**:
  + Precision: 0.39
  + Recall: 0.50
  + F1-Score: 0.44

**Key Observations**

* **Poor Performance for Cytoplasm**:
  + Precision, recall, and F1-score are 0.00, indicating no correct predictions for cytoplasm.
  + Due to synthetic labels and feature overlap, making it hard to distinguish cytoplasm.
* **Better Performance for Nucleus and Mitochondria**:
  + Nucleus: F1-score of 0.67 (60% precision, 75% recall).
  + Mitochondria: F1-score of 0.57 (50% precision, 67% recall).
  + Model has ability to learn patterns for these classes, despite random labels.
* **Small Test Set**:
  + Only 10 samples in the test set.
  + Small size contributes to low accuracy.
* **Impact of Synthetic Labels**:
  + Random labels limit the model’s ability to learn meaningful patterns.

**Recommendations for Improvement**

To enhance the pipeline’s performance and usability:

* **Use Real Labels**: Replace synthetic labels with true localization data. This will enable meaningful predictions.
* **Increase Dataset Size**: Use a larger FASTA file to provide more training and test samples.
* **Feature Engineering**: Include additional features (e.g., amino acid composition) to improv