CSE 6363
Machine Learning

Project Report

HIERARCHICAL CLUSTERING FOR UCI SEED DATASET

Dhairya Parekh

(1001868341)

Purpose of the Project

- Implementing a supervised K Nearest Neighbors algorithm to identify the species that are obtained as a result of Hierarchical Agglomerative clustering too create labels for the species and to determine the performance of both the learning algorithms

Steps implemented to fulfil the purpose

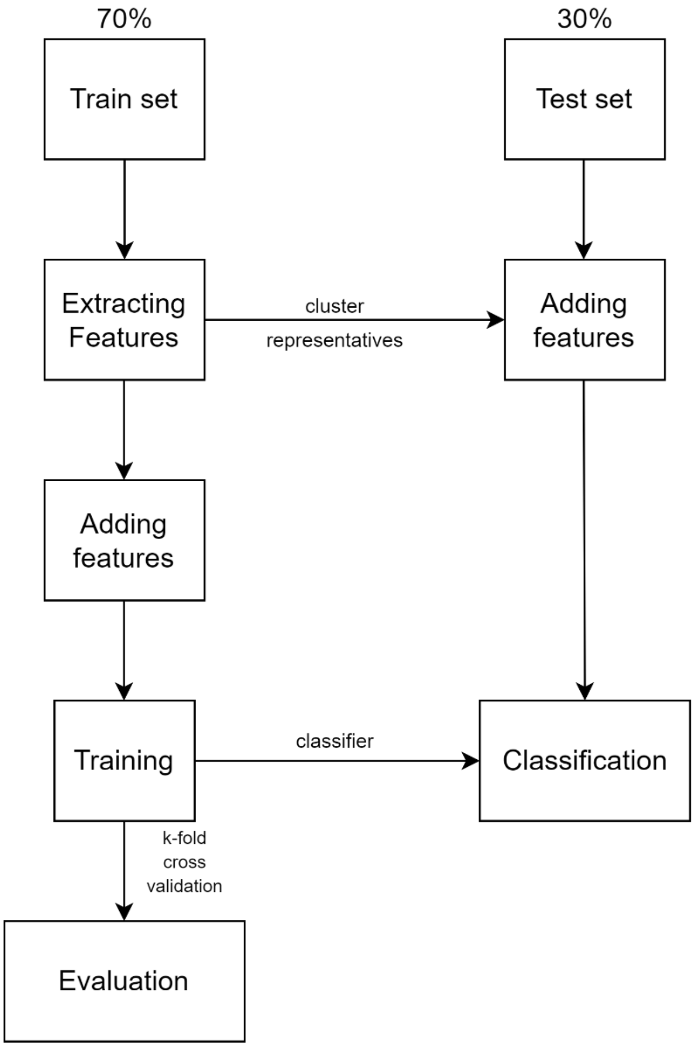
- Grouping the given dataset into clusters to assign species based on similarity through cluster grouping
- Generating new features for the datapoints in the dataset based on some association with other clusters and adding them to the existing features of the dataset
- Training a K Nearest Neighbor Classifier on the modified dataset to identify the species
- Evaluating the performance of the Classifier
- Determining the association of the datapoints with the clusters through K Nearest Neighbors
- Using K-fold Cross Validation to evaluate the expected fit of the KNN classifier to estimate the performance of the model on unknown data
- Determining the suitable number of clusters for respective linkages
- Identifying the species of the datapoints in the new data

Explanation of the code

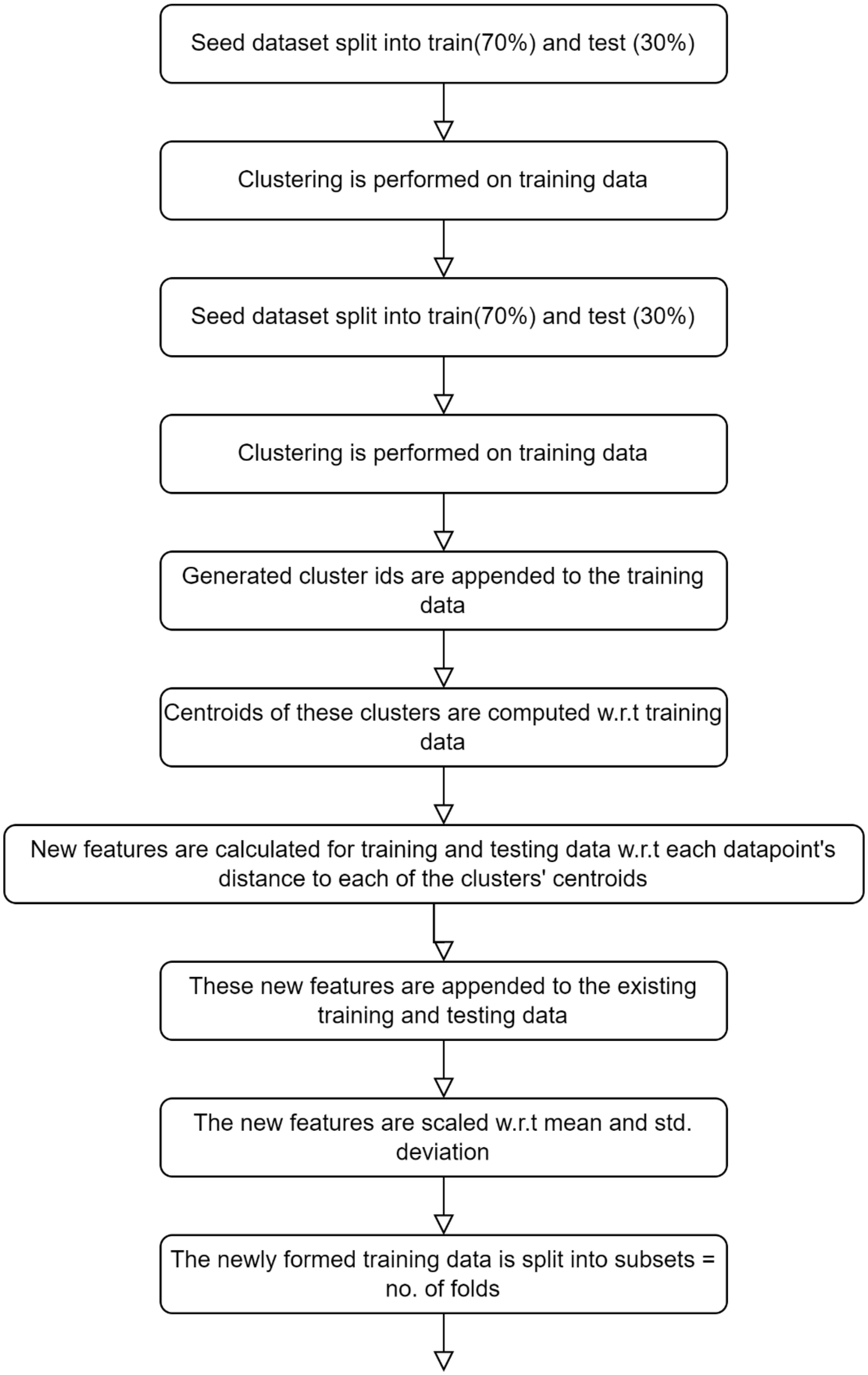
- Functions for Agglomerative Clustering algorithm
- Functions for K Nearest Neighbor Classification
- Functions to generate new features based on cluster representatives and scaling them
- Creating K-fold cross validation functions and determining the performance of the classifier pertaining to different types of clustering linkages
- Creating function for returning the accuracy
- Creating function to return the result in the form of accuracy from cross validation

- Consequently, code segment to determine the optimal number of clusters based on the accuracy for respective linkages
- Code to predict the species of unlabeled data

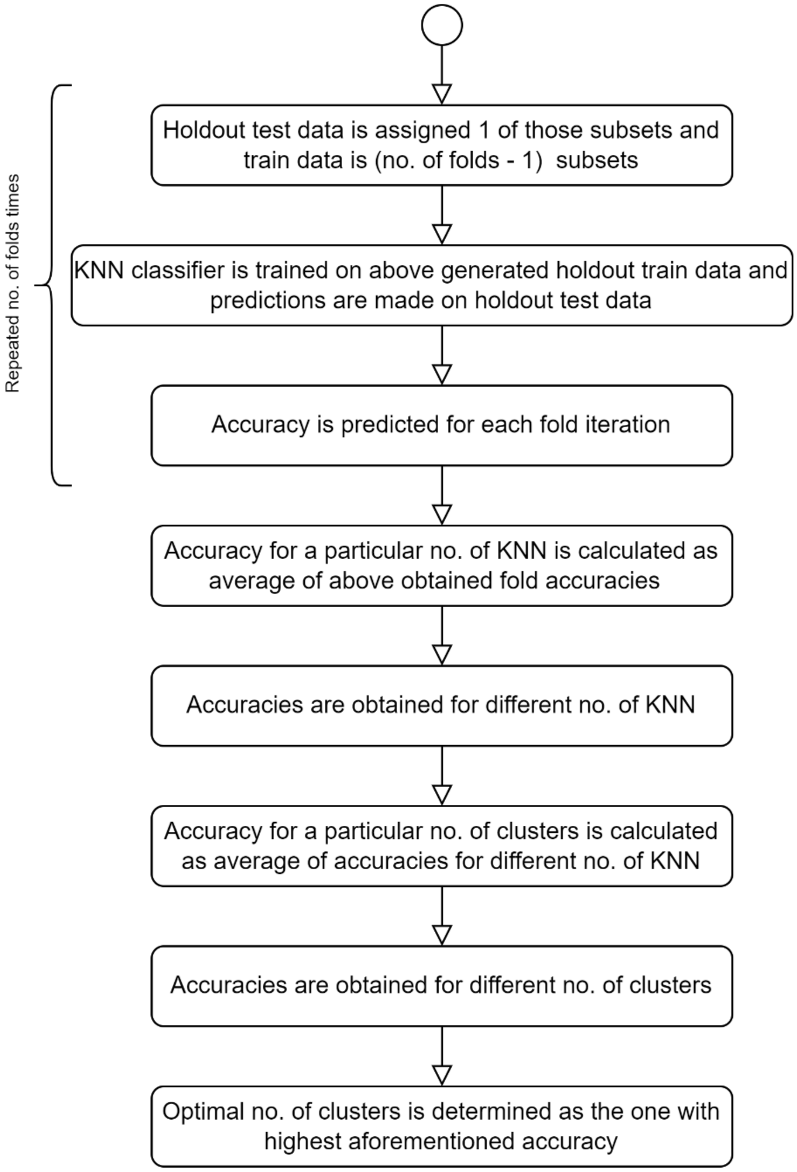
Idea



Walkthrough of the code


Seed dataset split into train(70%) and test (30%)
Clustering is performed on training data
Seed dataset split into train(70%) and test (30%)
Clustering is performed on training data
Generated cluster ids are appended to the training data
Centroids of these clusters are computed w.r.t training data
New features are calculated for training and testing data w.r.t each datapoint's distance to each of the clusters' centroids
These new features are appended to the existing training and testing data
The new features are scaled w.r.t mean and std. deviation
The newly formed training data is split into subsets = no. of folds

Walkthrough (contd.)



Holdout test data is assigned 1 of those subsets and train data is (no. of folds - 1) subsets

KNN classifier is trained on above generated holdout train data and predictions are made on holdout test data

Accuracy is predicted for each fold iteration

Accuracy for a particular no. of KNN is calculated as average of above obtained fold accuracies

Accuracies are obtained for different no. of KNN

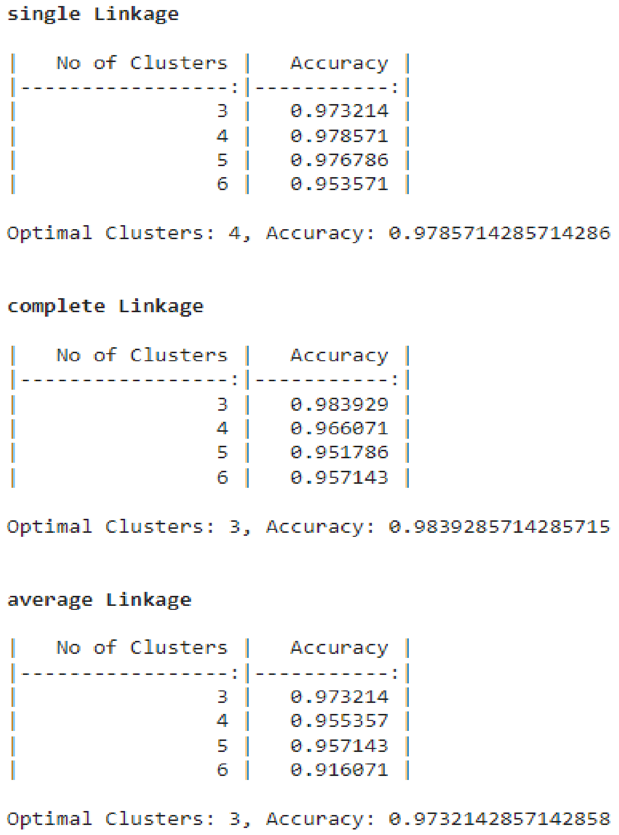
Accuracy for a particular no. of clusters is calculated as average of accuracies for different no. of KNN

Accuracies are obtained for different no. of clusters

Optimal no. of clusters is determined as the one with highest aforementioned accuracy

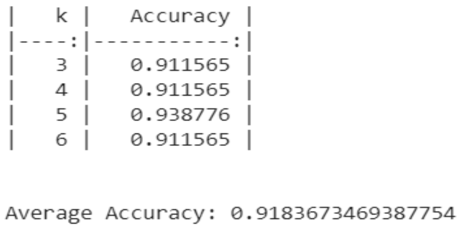
Results

Output after evaluating the model on modified dataset with added features



```
single Linkage

|   No of Clusters |   Accuracy |
|-----------------:|-----------:|
|                3 |   0.973214 |
|                4 |   0.978571 |
|                5 |   0.976786 |
|                6 |   0.953571 |

Optimal Clusters: 4, Accuracy: 0.9785714285714286


complete Linkage

|   No of Clusters |   Accuracy |
|-----------------:|-----------:|
|                3 |   0.983929 |
|                4 |   0.966071 |
|                5 |   0.951786 |
|                6 |   0.957143 |

Optimal Clusters: 3, Accuracy: 0.9839285714285715


average Linkage

|   No of Clusters |   Accuracy |
|-----------------:|-----------:|
|                3 |   0.973214 |
|                4 |   0.955357 |
|                5 |   0.957143 |
|                6 |   0.916071 |

Optimal Clusters: 3, Accuracy: 0.9732142857142858
```

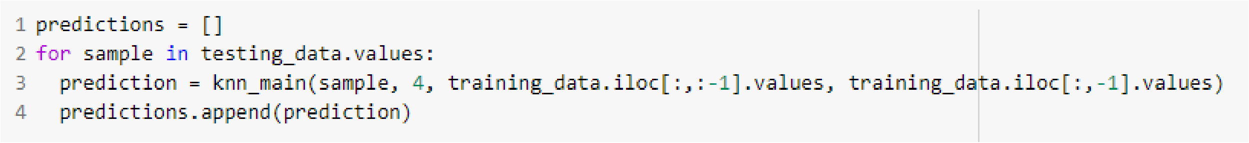
Output of classifier on original dataset without adding anything new to feature space



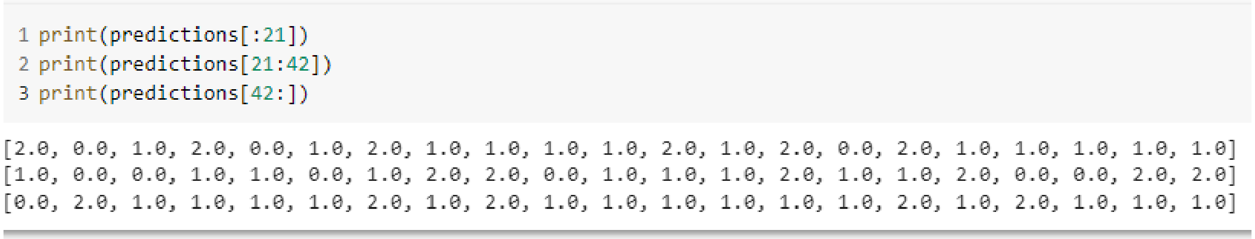
```
|   k |   Accuracy |
|----:|-----------:|
|   3 |   0.911565 |
|   4 |   0.911565 |
|   5 |   0.938776 |
|   6 |   0.911565 |

Average Accuracy: 0.9183673469387754
```

- From the above results, it can be inferred that adding new feature vectors to the existing dataset as a result of clustering and consequently implementing KNN classification on the modified dataset gives a better performance in evaluation as compared to classifying on the original Seed Dataset. Hence, the generated feature vectors are improving the performance of the classification model.

- Predictions for modified testing data with added features using
    - ***training_data, testing_data = datasets_for_KNN(df, 3, 'average')***



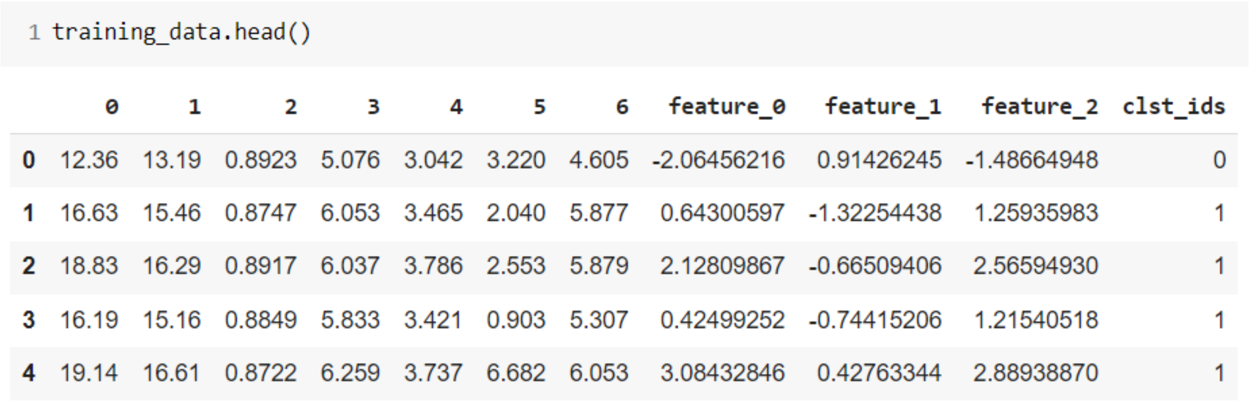
```
predictions = []
for sample in testing_data.values:
  prediction = knn_main(sample, 4, training_data.iloc[:,:-1].values, training_data.iloc[:,-1].values)
  predictions.append(prediction)
```



```
print(predictions[:21])
print(predictions[21:42])
print(predictions[42:])
```

```
[2.0, 0.0, 1.0, 2.0, 0.0, 1.0, 2.0, 1.0, 1.0, 1.0, 1.0, 2.0, 1.0, 2.0, 0.0, 2.0, 1.0, 1.0, 1.0, 1.0, 1.0]
[1.0, 0.0, 0.0, 1.0, 1.0, 0.0, 1.0, 2.0, 2.0, 0.0, 1.0, 1.0, 1.0, 2.0, 1.0, 1.0, 2.0, 0.0, 0.0, 2.0, 2.0]
[0.0, 2.0, 1.0, 1.0, 1.0, 1.0, 2.0, 1.0, 2.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 2.0, 1.0, 2.0, 1.0, 1.0, 1.0]
```

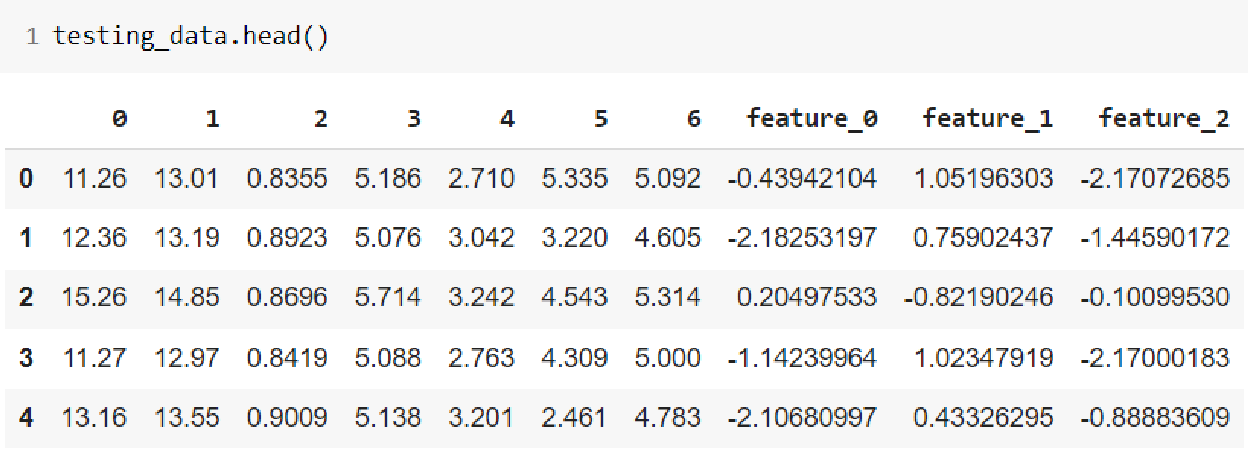
- Training Data sample with added features



```
training_data.head()
```

	0	1	2	3	4	5	6	feature_0	feature_1	feature_2	clst_ids
0	12.36	13.19	0.8923	5.076	3.042	3.220	4.605	-2.06456216	0.91426245	-1.48664948	0
1	16.63	15.46	0.8747	6.053	3.465	2.040	5.877	0.64300597	-1.32254438	1.25935983	1
2	18.83	16.29	0.8917	6.037	3.786	2.553	5.879	2.12809867	-0.66509406	2.56594930	1
3	16.19	15.16	0.8849	5.833	3.421	0.903	5.307	0.42499252	-0.74415206	1.21540518	1
4	19.14	16.61	0.8722	6.259	3.737	6.682	6.053	3.08432846	0.42763344	2.88938870	1

- Testing Data sample with added features



```
testing_data.head()
```

	0	1	2	3	4	5	6	feature_0	feature_1	feature_2
0	11.26	13.01	0.8355	5.186	2.710	5.335	5.092	-0.43942104	1.05196303	-2.17072685
1	12.36	13.19	0.8923	5.076	3.042	3.220	4.605	-2.18253197	0.75902437	-1.44590172
2	15.26	14.85	0.8696	5.714	3.242	4.543	5.314	0.20497533	-0.82190246	-0.10099530
3	11.27	12.97	0.8419	5.088	2.763	4.309	5.000	-1.14239964	1.02347919	-2.17000183
4	13.16	13.55	0.9009	5.138	3.201	2.461	4.783	-2.10680997	0.43326295	-0.88883609