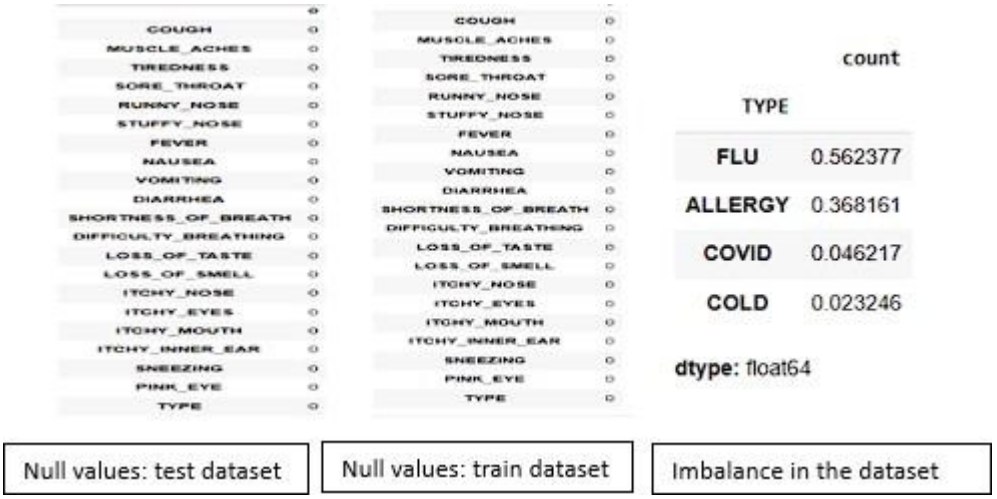
Assignment-3

Principles of Machine Learning

Data Exploration

- Load the dataset and analyse class distribution and other key characteristics.
  The dataset is of the shape: (40007, 20) excluding the 'TYPE' column. We looked at the dataset.describe() in order to understand the hyperparameters. We also checked if there are any null values and confirmed the imbalance in the dataset.

- Summarized findings in a brief data exploration report.



	0
COUGH	0
MUSCLE_ACHES	0
TIREDNESS	0
SORE_THROAT	0
RUNNY_NOSE	0
STUFFY_NOSE	0
FEVER	0
NAUSEA	0
VOMITING	0
DIARRHEA	0
SHORTNESS_OF_BREATH	0
DIFFICULTY_BREATHING	0
LOSS_OF_TASTE	0
LOSS_OF_SMELL	0
ITCHY_NOSE	0
ITCHY_EYES	0
ITCHY_MOUTH	0
ITCHY_INNER_EAR	0
SNEEZING	0
PINK_EYE	0
TYPE	0

Null values: test dataset

	0
COUGH	0
MUSCLE_ACHES	0
TIREDNESS	0
SORE_THROAT	0
RUNNY_NOSE	0
STUFFY_NOSE	0
FEVER	0
NAUSEA	0
VOMITING	0
DIARRHEA	0
SHORTNESS_OF_BREATH	0
DIFFICULTY_BREATHING	0
LOSS_OF_TASTE	0
LOSS_OF_SMELL	0
ITCHY_NOSE	0
ITCHY_EYES	0
ITCHY_MOUTH	0
ITCHY_INNER_EAR	0
SNEEZING	0
PINK_EYE	0
TYPE	0

Null values: train dataset

TYPE	count
FLU	0.562377
ALLERGY	0.368161
COVID	0.046217
COLD	0.023246

dtype: float64

Imbalance in the dataset

- To address the issue of class imbalance in the training dataset, a combined approach of undersampling and SMOTE (Synthetic Minority Oversampling Technique) was implemented. The majority classes, "FLU" and "ALLERGY," were undersampled to 10,000 samples each to

reduce their dominance, ensuring that the model does not become biased towards them. For the minority classes, "COVID" and "COLD," SMOTE was used to generate synthetic samples, increasing their representation to 5,000 samples each. This hybrid approach maintains a balanced dataset while avoiding potential discrepancies caused by solely relying on synthetic data for smaller classes. By combining actual data from under sampling with synthetic data generated for minority classes, the final dataset achieves a more equitable distribution across all classes, allowing the model to generalize better during predictions.

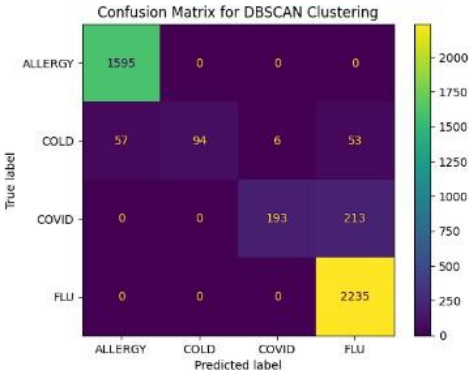
2. Classification Task:

Random Forest Classifier:

Random Forest builds multiple decision trees during training. Each tree is constructed using a random subset of the training data and features. The algorithm combines the predictions from all the trees to make the final decision: Uses majority voting among the trees. The confusion matrix was plotted in order to evaluate the trained model using the test dataset.

Accuracy: 0.9260008996851102
Precision: 0.9572470393131759
Recall: 0.7307471264367815
F1 Score: 0.7956793437928156

Random Forest Confusion Matrix



We performed hyper-parameter tuning by using randomised search CV in order to achieve optimum parametric values for training the model.

The following displays the best training score for Random Forest classifier:

Best Cross-Validation accuracy: 0.9536949313573001

Best Parameters: RandomForestClassifier(max_depth=10, min_samples_leaf=4, n_samples_split=5, n_estimators=500)

KNN Classification:

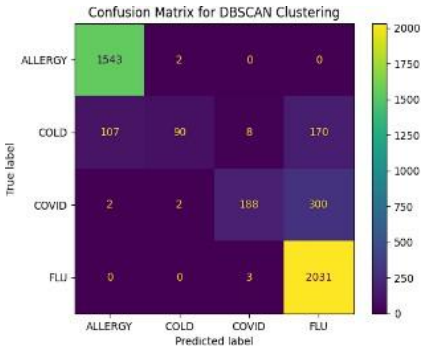
For a given input data point, the algorithm identifies the k closest points (neighbors) from the training dataset based on a chosen distance metric. The class of the input point is determined by the majority class among its k-nearest neighbors (majority voting).
We performed Hyper-parameter tuning using randomised search CV. Following displays the best parameters and performance metrics:

```
Best Parameters: {'weights': 'uniform', 'p': 2, 'n_neighbors': 11, 'metric': 'euclidean'}
Best Cross-Validation Accuracy: 0.9087668290025317
```

We evaluated the model using the confusion matrix as well.

```
K nearest metrics
 Accuracy: 0.8663967611336032
 Precision: 0.6548360991256618
 Recall: 0.9120662417482915
 F1 SCore: 0.6972285746158464
```

KNN Confusion Matrix



We chose not to apply scaling methods like normalization, standard scaling, or min-max scaling to the dataset because all the feature values in both the training and test datasets are binary (0 or 1). These binary values already exist in a consistent range, and scaling would neither enhance the relationships between features nor improve the performance of classification algorithms like KNN or Random Forest.

In fact, scaling binary data can be counterproductive or meaningless, as transforming a binary value (e.g., 0 to a scaled 0.0 or 1 to a scaled 1.0) does not provide any additional information or improve model interpretability. For algorithms like KNN, which rely on distances, or Random Forest, which uses decision trees to split data based on feature thresholds, the binary nature of the features is already optimal and requires no further adjustment. Consequently, scaling is unnecessary and might even introduce inconsistencies when dealing with such inherently categorical data. This decision ensures the dataset retains its natural structure, allowing algorithms to effectively learn from the binary patterns.

3. Clustering Task:

We chose DBSCAN and Agglomerative Clustering with Hamming distance because the dataset consists entirely of binary values (0s and 1s), making Hamming distance an appropriate metric for measuring dissimilarity by counting the number of differing bits.

KMeans and Gaussian Mixture Models (GMM) were not used because they rely on distance metrics like Euclidean distance, which are not meaningful for binary data; such metrics would fail to capture the true similarity between rows and could lead to incorrect cluster assignments. Additionally, scaling methods like normalization, Standard Scaling or Min Max Scaling were not applied because scaling binary values (0 and 1) would have no effect or introduce unnecessary transformations, as the values are already in a consistent range. Thus, DBSCAN and Agglomerative Clustering with Hamming distance provide a more accurate and meaningful approach for clustering binary datasets.

DBSCAN: When we applied DBSCAN, we got a single cluster label as a result of there being a high similarity in feature representation of 4 classes-FLU, ALLERGY, COVID, and COLD.

```
set(dbscan_labels)
```

```
{0}
```

We can also calculate cosine similarity between these classes randomly and show a high similarity score, indicating that these classes overlap in feature space.
Cosine similarity between Covid and Allergy is 0.99371801

```
cosine_similarity(t1_ar, t2_ar) # Covid vs Allergy
```

```
array([[0.99371801]])
```

Cosine similarity between Covid and Flu is 0.98338409

```
cosine_similarity(t1_ar, t2_ar) # Covid vs Flu
```

```
array([[0.98338409]])
```

Cosine similarity between Covid and Cold is 0.99314586

```
cosine_similarity(t1_ar, t2_ar) # Covid vs Cold
```

```
array([[0.99314586]])
```

Calculating cosine similarity between two random rows from different classes (e.g., COVID vs. COLD) is a meaningful way to quantify their similarity in the binary feature space. Since all values are 0 or 1 with no outliers or missing data, cosine similarity effectively captures the overlap in feature patterns. A high similarity score indicates that the feature vectors for these classes are highly aligned, reinforcing the observation that the categories are inherently similar and challenging to differentiate using clustering.

4. Comparative Analysis and Reporting :

The performance metrics of KNN and Random Forest classifiers highlight the advantages of supervised learning over clustering methods. For KNN, the model achieved an accuracy of

86.6%, with a high recall of 91.2%, indicating its strength in correctly identifying most true positives. However, its precision was lower at 65.5%, suggesting some false positives. The F1 score of 69.7% reflects a balance between precision and recall, making KNN an effective choice for datasets where recall is more critical. Random Forest, on the other hand, outperformed KNN with an accuracy of 92.6% and a higher precision of 95.7%, showcasing its ability to correctly classify positive cases with fewer false positives. However, its recall of 73.1% was lower than KNN, indicating that while Random Forest makes fewer mistakes, it may miss some true positives. The F1 score of 79.6% demonstrates an overall better balance of precision and recall compared to KNN.

In contrast, clustering methods like DBSCAN and Agglomerative Clustering, which aim to find natural groupings without utilizing class labels, struggled with this dataset. The low silhouette score of 0.03 indicates poorly defined clusters, and the Adjusted Rand Index (ARI) of 0.58 shows moderate agreement with the true labels. These results stem from the inherent similarity between the classes (FLU, ALLERGY, COVID, and COLD), as evidenced by high cosine similarity scores between random rows of different classes. Clustering methods are not equipped to handle such overlapping classes effectively, as they lack the benefit of labelled data to guide the grouping process. In comparison, supervised methods like KNN and Random Forest leverage the labelled data to learn class-specific patterns, making them more accurate and reliable for classification tasks in such scenarios.

References:

https://scikit-learn.org/dev/modules/generated/sklearn.ensemble.RandomForestClassifier.html
https://scikit-learn.org/dev/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
https://scikit-learn.org/1.5/modules/generated/sklearn.cluster.DBSCAN.html
https://scikit-learn.org/dev/modules/generated/sklearn.cluster.AgglomerativeClustering.html
https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.RandomOverSampler.html
https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html
https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html
https://scikit-learn.org/dev/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html
https://scikit-learn.org/1.5/modules/mixture.html