**[Vector Borne Disease Prediction](https://www.kaggle.com/datasets/richardbernat/vector-borne-disease-prediction)**

**Team 11**

**1. Introduction**

Usually, a doctor couldn’t tell a certain vector borne disease a patient has got just based on their symptoms. Before the result of relatively time-consuming examinations comes out, a doctor may need to rely on experience to guess which disease the patient has got as the disease can be fatal with time passing by. This data analysis aims to predict the disease based on the symptoms to help better treat the patient and reduce the fatality rate accordingly.

**2. Profile of the dataset**

2.1 Source

The dataset for this project is provided by Kaggle (https://www.kaggle.com/competitions/playground-series-s3e13/data), generated from a deep learning model trained on the Vector Borne Disease Prediction dataset. And the feature distributions are highly similar to the original.

2.2 Structure

• The whole project is mainly based on a training dataset "train.csv" and a test dataset "test.csv".

• The training data set has 706 rows, with 66 characteristics, including 64 boolean features, and each of them represents a single symptom. For every feature, the value can only be 0.0 or 1.0, in other words, negative or positive.

2.3 Limitation

• Small data size may lead to a bad similarity.

• Relative high data complexity: large amount of features may lead to overfitting.

**3. Preprocessing and Mining**

3.1 Preprocessing

Data cleaning: Including handling missing values or outliers, removing duplicates, and correcting any errors in the data. Since the datasets we use are directly provided by Kaggle, the spaces in the data sets have already been replaced with underscores. We will first remove any rows with underscores as well as any duplicate entries in the dataset. Then we will identify or correct the inaccuracies in the dataset, such as invalid value under symptom features. We will also deal with any outliers which can skew the analysis, by either filtering them out or adjusting them approximately.

Data reduction: Reducing the size of the data while preserving the essential characteristics using Principal Component Analysis or feature selection. As the training dataset has many features but at the same time, a relatively small size, to improve the performance of the model, we will remove irrelevant or redundant features to improve the quality of data. Alternatively, we might also select the most relevant features by Recursive Feature Elimination(RFE) or Tree-based feature selection.

Data transformation: Scaling numerical features and encoding categorical features. Normally data transformation is used when the features' values range wildly, in this project, the values of 64 features are all binary, either 0.0 or 1.0, so we do not need to standardise the range of input here. But, additionally, apart from the above feature-reducing methods, to ensure the accuracy of prediction, we might also use feature engineering instead.

Feature interaction: Identifying the correlation between features and the target variable. For example, there are two attributes "back pain" and "weakness"; individually, both of them can be important factors in predicting disease like Lyme disease. However, they may also interact with each other. In this case, the disease prediction model that considers these two symptoms as independent features may not accurately predict the final disease. Thus, it is important to take into account the potential interactions between such features to develop a more accurate model.

Data visualisation: Using graphs or charts to visualise the data, helping to identify the patterns and outliers. And we can also identify and select the most important features from the graphs.

3.2 Mining Techniques

In this project, we mainly depend on the Random Forest algorithm to conduct the data mining process. It is crucial that the hyperparameters we use are the most optimal ones. Thus, having hyperparameter tuning is necessary before the actual mining process begins. Generally, we will use grid search or random search to find the proper number of trees in the forest, the maximum depth of each tree, as well as the number of features at each split.

After tuning the specific hyperparameter, we now can train a random forest on the training dataset. A random forest is an ensemble of decision trees, where each tree is trained on a randomly selected subset of the training set and a random subset of the specific input features. Since the training dataset is not big enough, here we must use a relatively shallow tree to prevent overfitting.

Once the random forest training is completed, we would then evaluate it on a separate validation dataset to assess its performance.

**4. Evaluation and Results**

Since we are dealing with a recommendation system, the metric MAP@k is one the most relevant for this type of problem. In order to use the MAP, we start with calculating the precision and recall. First of all we need to understand the meaning of MAP. M denotes Mean and AP denotes the Average Precision of each item. For this reason, for example, if we have 2500 (k) users, then the sum of the Average Precision for each user divided by 2500 is equal to MAP. Since Kaggel provides a test set for this evaluation method, we will use this.

Thus it turned out to be very efficient since the information retrieval is done and ranking of examples (recommendations) in the dataset is equally important. In addition, this evaluation method provides us a good tool to compare our final results with those of the other competitors, as this is the specified evaluation method of Kaggel. Thus, the evaluation results of the other participants of the Kaggle Competition can be used to compare our results.

In addition, we want to use a second evaluation method to answer the question of how good our elaborated model is on previously unseen data. We would like to determine the values of accuracy etc. through the Method of Cross-Validation. Thus we split the given training-dataset into 10 subsets of similar size. the training set will therefore not be used by us. Each of these subsets can then be used for testing and the remainder for training.

Because the dataset has 709 examples we can use this method and do not have to use the holdout method. By dividing the data into different subsets, we can test our results on different training sets and therefore through the Cross-Validation Method the dataset can be used for both: Training and Testing.

**5. Expectation**

We are pleased to introduce the random forest model, an ensemble learning technique used for classification, regression, and other tasks. This method constructs multiple decision trees during the training phase. Our approach involves conducting a Principle Component Analysis (PCA) to reduce the number of features from 64 to less than 20. We then create a correlation graph of the selected features to assess their interdependence and choose the most representative among them. This process results in approximately 15 features.

Next, we apply these selected features to the random forest model, which incorporates randomness during the tree construction process, making it less susceptible to outliers. To prevent overfitting, we adjust the hyperparameters of the model to limit the depth of the trees to below 10. A visualisation of the error and example node distribution would aid in analysing the distribution of error points. Our ultimate goal is to obtain a random forest model with a depth below 10 that can account for more than 90% of the points in the training set and deliver better detection results.

In conclusion, predicting vector borne diseases such as Dengue, Zika, and Japanese encephalitis based on patients' symptoms has significant potential to improve disease diagnosis and management. By using such techniques, we will identify patterns and relationships between different symptom features and disease outcomes that are difficult for human experts to detect. This can be especially useful in areas where these diseases are prevalent and can help healthcare providers start treatment early, potentially improving patient outcomes, also can help healthcare providers make informed decisions about patient care, including the appropriate level of medical intervention and monitoring. With an accurately trained model, healthcare providers can make more informed decisions about patient care and potentially prevent outbreaks by identifying high-risk areas and populations.

I had a problem: we have 64 features, and I thought I could downscale them to keep about half of the features, then Simon and Stefano would only have to consider the remaining half while modeling. So I calculated the correlation of every two features, but the problem here is, because the results only show two features strongly correlated, if I want to choose one or the other, I do not know which one to delete (what is the criterion). To resolve this problem, my idea is to use ANOVA, Chi-Square, and multi-info in “sklearn.feature\_selection”. And if there are features not included in the above three selected feature-set, then I consider them unimportant.

And about PCA and other deduction methods, seems no practical significance as they turn high-dimensional data into a low-dimensional space, which means they all have new mixed features, and I don’t know how to use them in the next step. Of course, I could use the random forest to deduce/cluster features, but it seems not different from the modeling step.