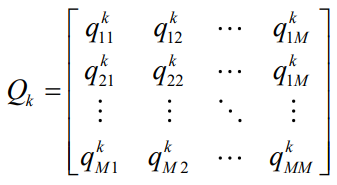
1. **Classifier integration model (CIM)**

* The classifier integration model utilizes the confusion matrix values as a weight value for constituent classifier models.
* For each data x applied to CIM, each local classifier produces the probability that the data x belongs to the class j, Cj , as follows



* where represents the probability that the classifier k, classifies the data as Class j when the data is from of Class i and M denotes the number of classes.
*  is the probability that the classifier k classifies the data as Class j
* (the probability is calculated for each class, vertically)

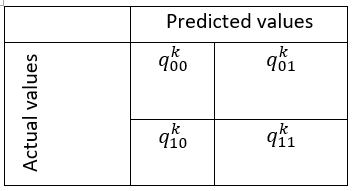
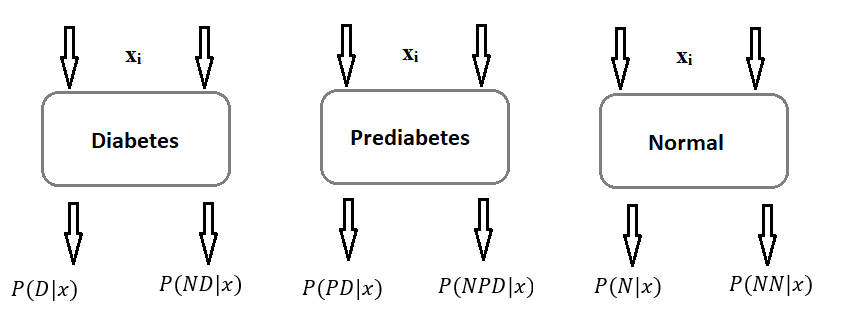


* the classifier yields a probability that the data x belongs to the class j, Cj , as follows



1. **Classifier integration model (CIM) for Diabetes dataset**

* we generated three distinct (ensembled) classifiers for each class (Diabetes, Prediabetes, Normal)




* +
* + [N: normal, NN: not normal, D: diabetes, ND: note diabetes, PD: prediabetes, NPD: not prediabetes]
  1. **Experimental results**
     1. **Diabetes dataset**
        + Baseline classifier: [RF], accuracy= 0.70
        + Voting classifier: soft voting using
          - RF=0.70, SVM=0.72, XGBoost=0.72
          - Overall accuracy=0.715, all classifier is assigned equal weight=1,
        + CIM (using all classes): accuracy=0.68
        + CIM (using single classes): accuracy=0.71
        + Modified CIM: in this section the probability of a given data point belonging to a certain class is modified as follows:
          - Overall accuracy: 0.73
     2. **Wine dataset**
        + Baseline classifier: accuracy = 0.94
        + CIM: accuracy = 0.98
     3. **Wine dataset**
        + Baseline classifier: accuracy = 0. 936
        + CIM: accuracy = 0.92
  2. Genetic algorithm-based weight optimization for classifier models

In this section we used genetic algorithm to find appropriate weight values for each constituent models of the ensemble classifier, with the assumption that the most accurate model will have higher weight value than the low performing models.

* + 1. Diabetes dataset experiments
       - 3 class ensemble approach
         * Baseline classifier: [RF], accuracy 0.68 [0.67 ± 0.015]
         * Using GA: accuracy = 0.735 [ cv score 0.73 ± 0.003] [0.78 ± 0.005]
         * Using Voting classifier = 0.735 [ cv score 0.76 ± 0.03]
       - Single class ensemble approach
         * Baseline classifier: [RF], accuracy 0.68 [cv 0.70 ± 0.005]
         * Using GA:

Experiments

Trial 1:

Trial 2:

Trial 3:

Trial 5:

Trial 6:

**Results**

trial1\_scores f1\_score 0.72 [cv 0.70 ± 0.005] [cv 0.79 ± 0.006]

trial2\_scores f1\_score 0.73 [cv 0.72 ± 0.004] [cv 0.79 ± 0.005]

trial3\_scores f1\_score 0.73 [cv 0.72 ± 0.004] [cv 0.79 ± 0.005]

trial4\_scores f1\_score 0.39 [cv 0.74 ± 0.003] [cv 0.80 ± 0.004]

trial5\_scores f1\_score 0.72 [cv 0.47 ± 0.056] [cv 0.44 ± 0.138]

trial6\_scores f1\_score 0.73 [cv 0.72 ± 0.003] [cv 0.79 ± 0.006]

* + 1. seed dataset experiments
       - 3 class ensemble approach
         * Baseline classifier: [RF], accuracy 0.95 [0.91± 0.084]
         * Using GA: accuracy = 0.95 [ cv score 0.93 ± 0.064]
         * Using Voting classifier = 0.96 [ cv score 0.94 ± 0.025]
       - Single class ensemble approach
         * Baseline classifier: [RF], accuracy 0.95 [0.92 ± 0.064]
         * Using GA:

trial1\_scores f1\_score 0.95 [cv 0.93 ± 0.005]

trial2\_scores f1\_score 0.95 [cv 0. 93 ± 0.006]

trial3\_scores f1\_score 0.95 [cv 0. 93 ± 0.006]

trial4\_scores f1\_score 0.84 [cv 0.83 ± 0.005]

trial5\_scores f1\_score 0.95 [cv 0.94 ± 0.098]

trial6\_scores f1\_score 0.95 [cv 0.94 ± 0.069]

* + 1. wine dataset experiments
       - 3 class ensemble approach
         * Baseline classifier: [RF], accuracy 0.94 [0.97± 0.037]
         * Using GA: accuracy = 0.98 [ cv score 0.97 ± 0.037]
         * Using Voting classifier = 0.96 [ cv score 0.96 ± 0.0368]
       - Single class ensemble approach
         * Baseline classifier: [RF], accuracy 0.94 [0.98 ± 0. 021]
         * Using GA:

trial1\_scores f1\_score 0.96 [cv 0.98 ± 0. 021]

trial2\_scores f1\_score 0.96 [cv 0. 98 ± 0. 021]

trial3\_scores f1\_score 0.96 [cv 0. 98 ± 0.021]

trial4\_scores f1\_score 0.69 [cv 0.90 ± 0.074]

trial5\_scores f1\_score 0.98 [cv 0.99 ± 0.016]

trial6\_scores f1\_score 0.96 [cv 0.99 ± 0.016]

* 1. Stacking single class identifying models using metaclassifiers

In this section, we tried to utilize the performance of a ensemble approach to identify a single class using several models. 100 models for each class [normal, prediabetes, diabetes] are generated using different randomly selected features and using XGBoost and RF algorithms. Each model generates the probability of the data point x belongs to a certain class or not [].

We combined all the probabilities [] for each class including the confusion matrix to generate a new dataset. We trained the meta classifier to get the final result of the ensemble method using the newly generated dataset. The following are the experimental results in terms of the classification accuracy.

* RF: 0.714 (0.726 0.02)
* XGBoost: 0.714 (0.726 0.02)
* SVM: 0.764 (0.71 0.02)
* K-means clustering: 0.73
  1. Hardness measures - k-Disagreeing Neighbors (kDN)

[Smith, Michael R., Tony Martinez, and Christophe Giraud-Carrier. "An instance level analysis of data complexity." Machine learning 95.2 (2014): 225-256.]

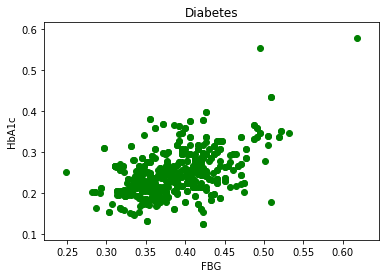
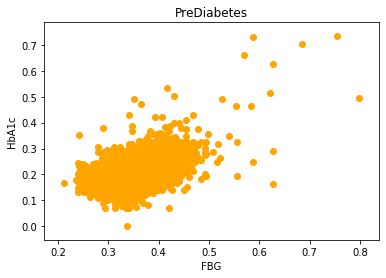
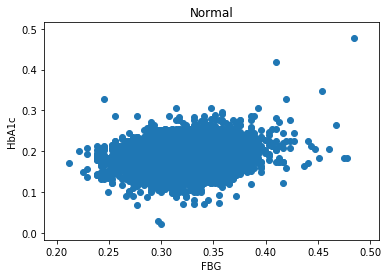
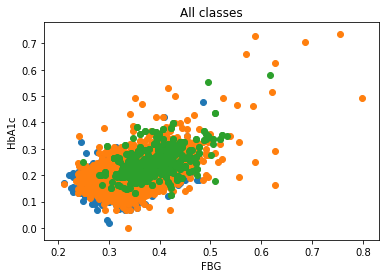
[Walmsley, Felipe N., et al. "An ensemble generation method based on instance hardness." 2018 International Joint Conference on Neural Networks (IJCNN). IEEE, 2018.]

kDN measures the local overlap of an instance in the original task space in relation to its nearest neighbors. The kDN of an instance is the percentage of the k nearest neighbors (using Euclidean distance) for an instance that do not share its target class value.

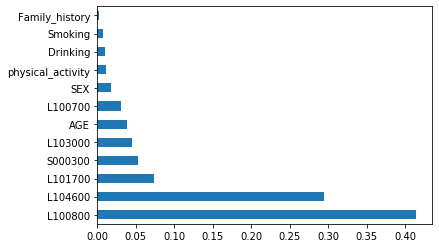
In this experiment, we grouped the dataset into two groups, easy to classy and hard to classify. Based on the kDN measure, all data points less than mean value of the kDN measure are considered as easy to classy and hard to classify for the rest.

* Accuracy
* Easy to classify dataset: 0.898 0.015
* Hard to classify dataset: 0.49 0.06

The following plot shows the class overlap between the three classes using the fasting blood glucose level and HbAlc features.



1. Feature importance from Random forest



L100700 – Uric Acid, L103000 – Triglycerides, S000300 - BMI, L101700 - r-GTP gamma, L104600 - HbA1C, L1000800 - FBG

1. Histogram plot of HbA1C, AGE

