Machine Learning:
Assignment 2

Active Learning and SOM Implementation

Prepared by

J LAKSHMI TEJA	**2017A7PS0068P**
ANISHKUMAR SS	**2017A7PS0069P**
ANIRUDDHA JAYANT KARAJGI	**2017A7PS0084P**

IN PARTIAL FULFILMENT OF THE COURSE
MACHINE LEARNING
(BITS F464)

Active Learning:

Active learning is a subfield of machine learning and, more generally, artificial intelligence. The key hypothesis is that if the learning algorithm is allowed to choose the data from which it learns—to be “curious,” if you will—it will perform better with less training.

Why is this a desirable property for learning algorithms to have? Consider that, for any supervised learning system to perform well, it must often be trained on hundreds of labeled instances. Sometimes these labels come at little or no cost. But for many other more sophisticated supervised learning tasks, labeled instances are very difficult, time-consuming, or expensive to obtain.

Active learning systems attempt to overcome the labeling bottleneck by asking queries in the form of unlabeled instances to be labeled by an oracle In this way, the active learner aims to achieve high accuracy using as few labeled instances as possible, thereby minimizing the cost of obtaining labeled data.

The three main settings that have been considered in the literature are:-

(i) membership query synthesis

(ii) stream-based selective sampling

(iii) pool-based sampling.

MEMBERSHIP QUERY SYNTHESIS:

One of the first active learning scenarios to be investigated is learning with membership queries In this setting, the learner may request labels for any unlabeled instance in the input space, including queries that the learner generates on the spot, rather than those sampled from some underlying natural distribution.

Query synthesis is reasonable for many problems, but labeling such arbitrary instances can be awkward if the oracle is a human annotator.

STREAM BASED SELECTIVE SAMPLING:

The key assumption is that obtaining an unlabeled instance is free, so it can first be sampled from the actual distribution, and then the learner can decide whether or not to request its label. This approach is sometimes called stream-based or sequential active learning, as each unlabeled instance is typically drawn one at a time from the data source, and the learner must decide whether to query or discard it.

The decision whether or not to query an instance can be framed several ways. One approach is to evaluate samples using some “informativeness measure” or “query strategy” and make a biased random decision, such that more informative instances are more likely to be queried.

POOL BASED SAMPLING:

For many real-world learning problems, large collections of unlabeled data can be gathered at once. This motivates pool-based sampling. Queries are selectively drawn from the pool, which is usually assumed to be closed although this is not strictly necessary. Typically, instances are queried in a greedy fashion, according to an informativeness measure used to evaluate all instances in the pool .

The main difference between stream-based and pool-based active learning is that the former scans through the data sequentially and makes query decisions individually, whereas the latter evaluates and ranks the entire collection before selecting the best query.

QUERY STRATEGIES:

From this point on, the notation x∗A to refer to the most informative instance(i.e., the best query) according to some query selection algo A.

- **UNCERTAIN SAMPLING:** The simplest and most commonly used query framework is uncertainty sampling. In this framework, an active learner queries the instances about which it is least certain how to label. This approach is often straightforward for probabilistic learning models.

$$x^*_{LC} = \underset{x}{\operatorname{argmax}}\ 1 - P_\theta(\hat{y}|x),$$

where $\hat{y} = \text{argmax}_y P_\theta(y|x)$

- **MARGIN SAMPLING:** The criterion for the least confident strategy only considers information about the most probable label. Thus, it effectively “throws away” information about the remaining label distribution. To correct for this, some researchers use a different multi-class uncertainty sampling variant called margin sampling.

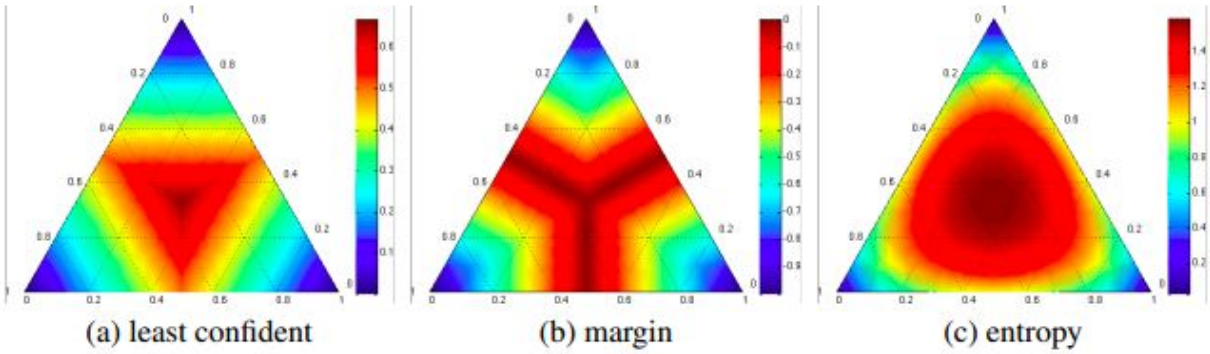
$$x_M^* = \underset{x}{\text{argmin}}\, P_\theta(\hat{y}_1|x) - P_\theta(\hat{y}_2|x)$$

  where yˆ1 and yˆ2 are the first and second most probable class labels under the model, respectively.

- **ENTROPY:** A more general uncertainty sampling strategy (and possibly the most popular) uses entropy (Shannon, 1948) as an uncertainty measure.

$$x_H^* = \underset{x}{\text{argmax}} - \sum_i P_\theta(y_i|x) \log P_\theta(y_i|x)$$

Empirical comparisons of these measures have yielded mixed results, suggesting that the best strategy may be application-dependent. Intuitively, though, entropy seems appropriate if the objective function is to minimize log-loss, while the other two (particularly margin) are more appropriate if we aim to reduce classification error.

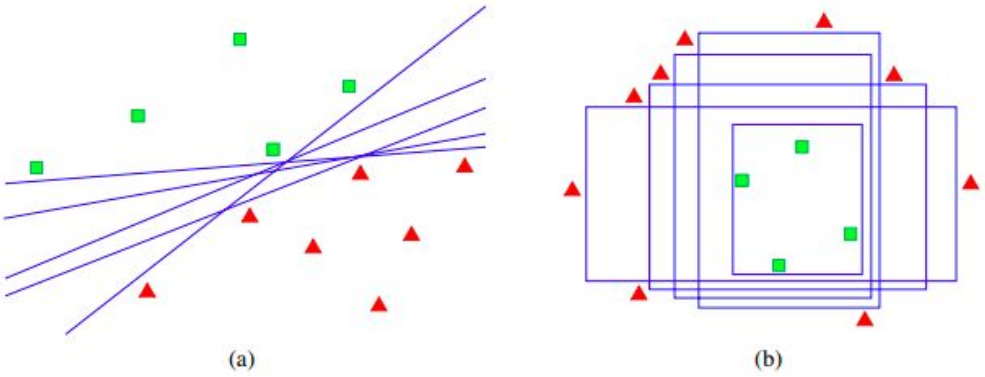


The heat maps from 3-class probability distribution. From
http://burrsettles.com/pub/settles.activelearning.pdf

QUERY-BY-COMMITTEE:

The QBC approach involves maintaining a committee C = {θ (1), . . . , θ(C)} of models which are all trained on the current labeled set L, but represent competing hypotheses. Each committee member is then allowed to vote on the labelings of query candidates. The most informative query is considered to be the instance about which they most disagree.

The fundamental premise behind the QBC framework is minimizing the version space, which is the set of hypotheses that are consistent with the current labeled training data L. Figure below illustrates the concept of version spaces . All hypotheses are consistent with the labeled training data in L but each represents a different model in the version space.



version spaces. From http://burrsettles.com/pub/settles.activelearning.pdf

This is exactly what QBC aims to do, by querying in controversial regions of the input space. In order to implement a QBC selection algorithm, one must:

- be able to construct a committee of models that represent different regions of the version space, and
- have some measure of disagreement among committee members.

For measuring the level of disagreement, two main approaches have been proposed:

- **VOTE ENTROPY**:

$$x^*_{VE} = \underset{x}{\operatorname{argmax}} - \sum_i \frac{V(y_i)}{C} \log \frac{V(y_i)}{C}$$

where yi again ranges over all possible labelings, and V(yi) is the number of “votes” that a label receives from among the committee members’ predictions, and C is the committee size.

- **Kullback-Leibler (KL) divergence** :

$$x^*_{KL} = \operatorname*{argmax}_x \frac{1}{C} \sum_{c=1}^{C} D(P_{\theta^{(c)}} \| P_{\mathcal{C}})$$

where

$$D(P_{\theta^{(c)}} \| P_{\mathcal{C}}) = \sum_i P_{\theta^{(c)}}(y_i|x) \log \frac{P_{\theta^{(c)}}(y_i|x)}{P_{\mathcal{C}}(y_i|x)}.$$

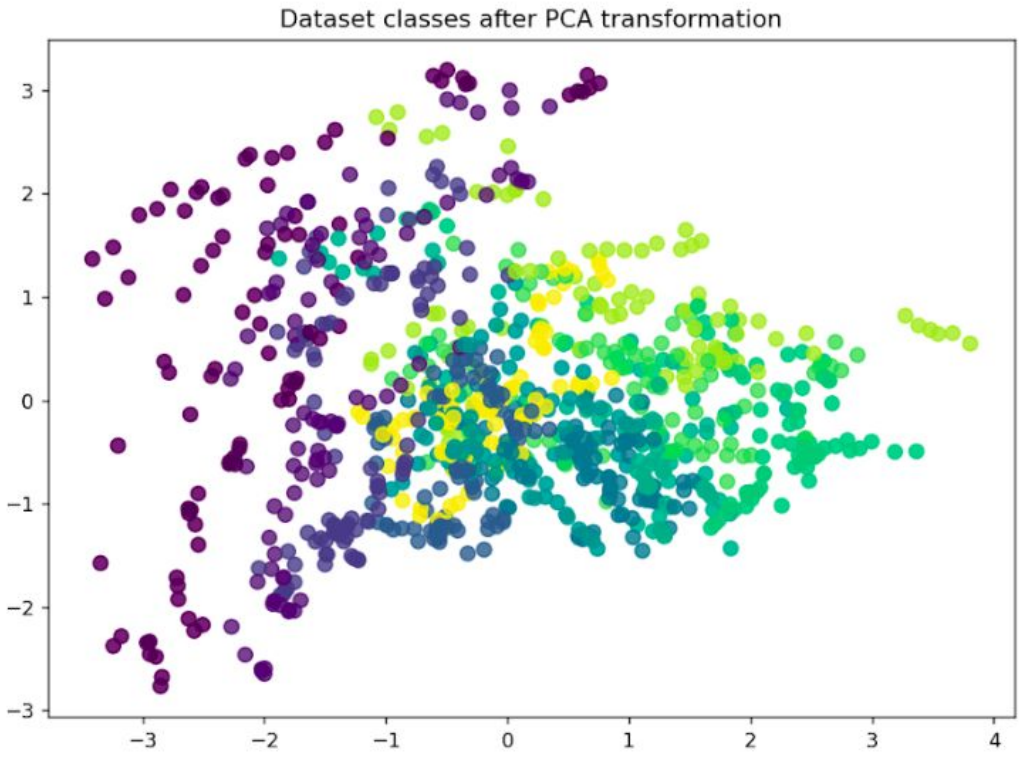
Here θ (c) represents a particular model in the committee, and C represents the committee as a whole, thus

$$P_{\mathcal{C}}(y_i|x) = \frac{1}{C} \sum_{c=1}^{C} P_{\theta^{(c)}}(y_i|x)$$

is the “consensus” probability that y_i is the correct label. K-L divergence is an information-theoretic measure of the difference between two probability distributions. So this disagreement measure considers the most informative query to be the one with the largest average difference between the label distributions of any one committee member and the consensus.

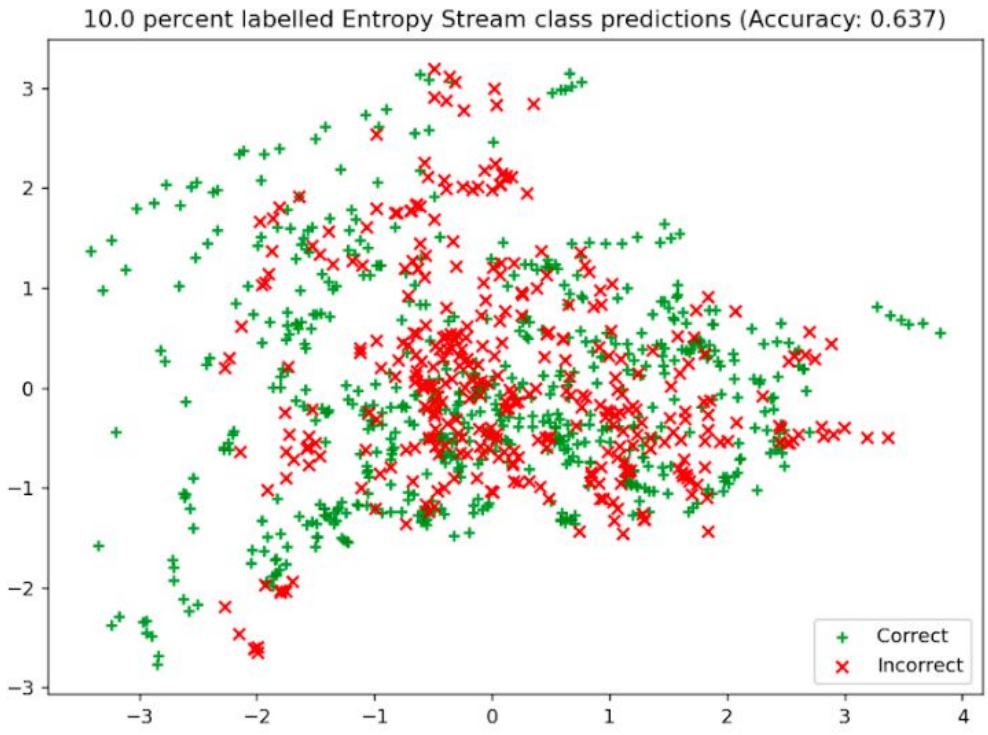
Dataset:

The dataset has 10 features and a target variable, which has 11 classes. There are 990 data points. The dataset was reduced to 2 dimensions using PCA just for visualization purposes. The dataset looks as shown below. The dataset is the file “phpd8EoD9.csv” present in the same folder as this report.



The dataset after applying PCA (PCA applied only for visualization purpose)

Here is how the predictions look like for the 10 percent labelled dataset before applying active learning. The accuracy is **0.637**. We can see that most of the points the model mis-classifies are the ones present close to the ideal classification boundary.



Results:

All the plot grids below have 4 horizontal lines, representing the points of additional 10%, 20%, 30% and 40% labelling.

Q1)b)i)

Final Accuracy of Pool based Least Confident: 1.0

Final Accuracy of Stream based Least Confident: 0.9848484848484849

Final Accuracy of Pool based Margin: 0.9939393939393939

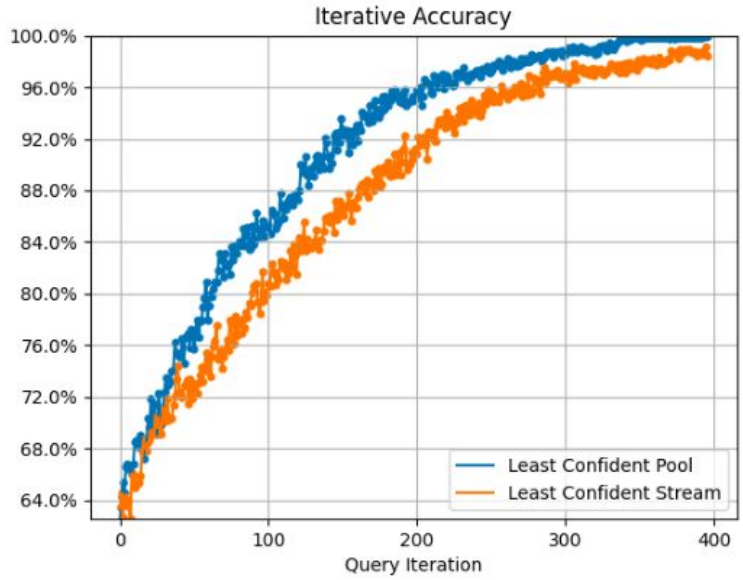
Final Accuracy of Stream based Margin: 0.990909090909091

Final Accuracy of Pool based Entropy: 0.9818181818181818

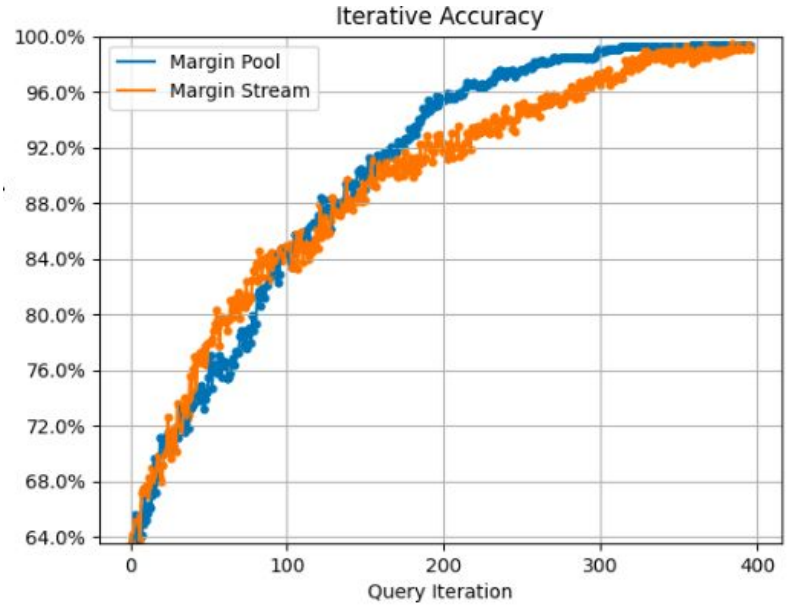
Final Accuracy of Stream based Entropy: 0.9515151515151515

It can be observed that the stream based approaches are less accurate than their pool based counterparts, which is not surprising considering that the stream based doesn't have the information of the whole dataset before deciding which point to query.

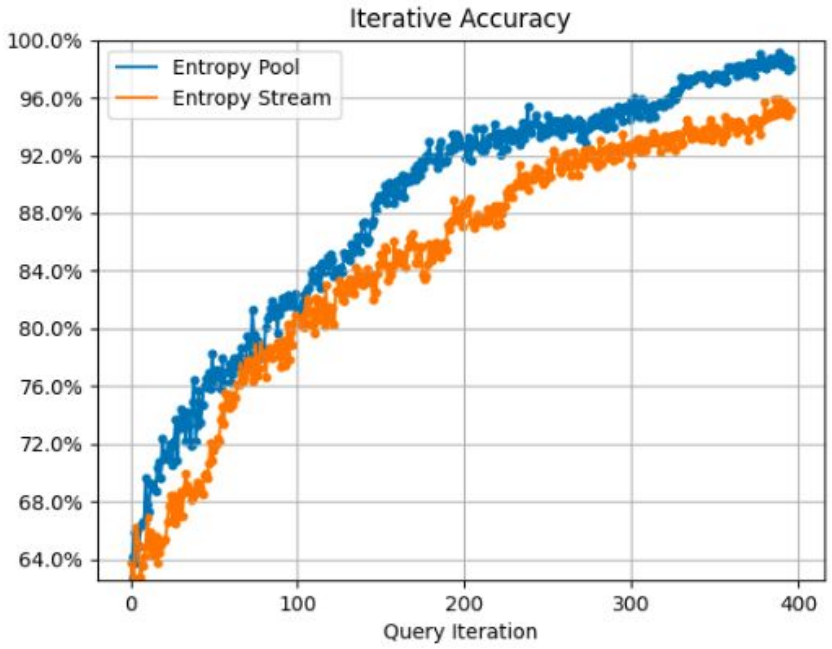
Also, least confident uncertainty is the best query measure for pool based learning, while margin uncertainty is the best for stream based learning. Entropy is the worst query measure for both stream and pool based learning. This is also not surprising, given that entropy focuses on probability distribution more than classification error.



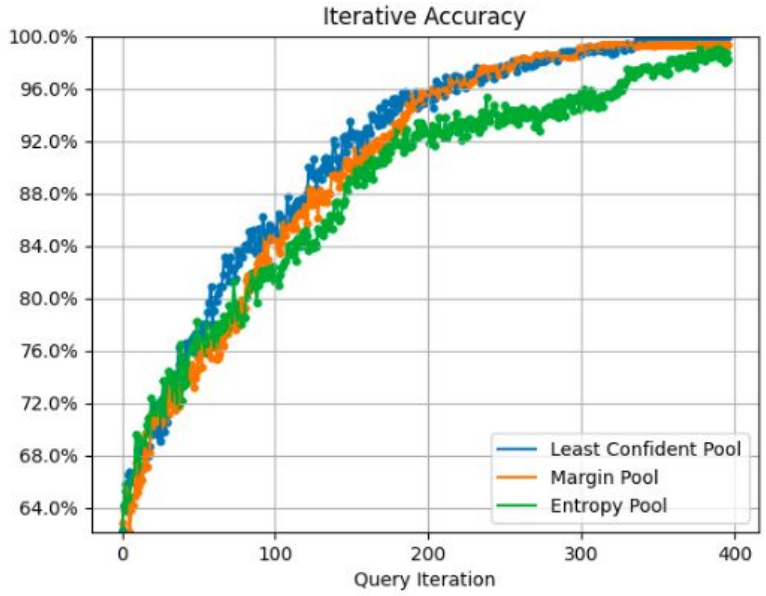
Comparing the accuracies of pool and stream based least confident uncertainty.



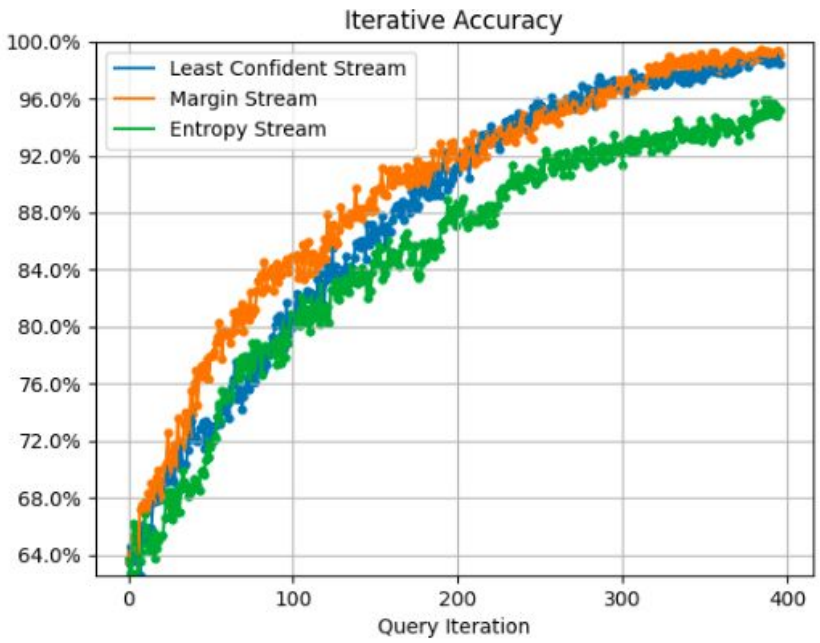
Comparing the accuracies of pool and stream based margin uncertainty.



Comparing the accuracies of pool and stream based entropy uncertainty



Comparing the accuracies of all the 3 pool based uncertainty measures



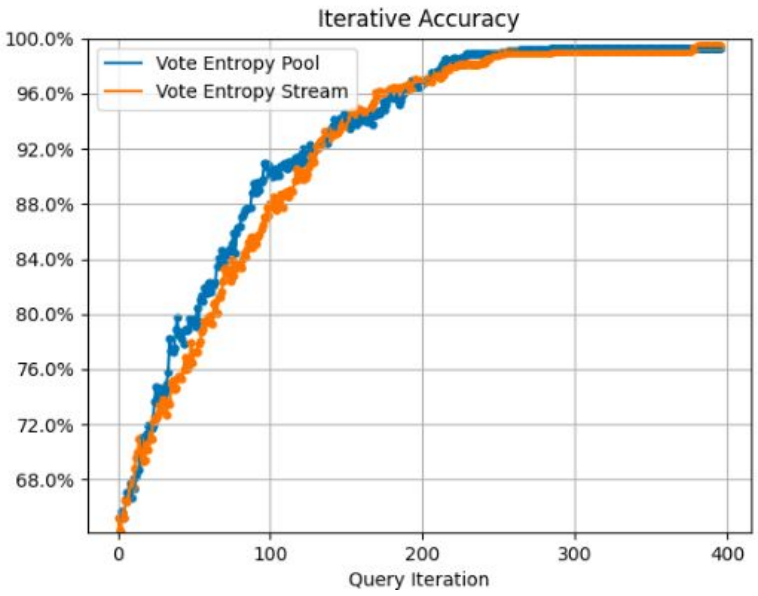
Comparing the accuracies of all the 3 stream based uncertainty measures

Q1)b)ii)

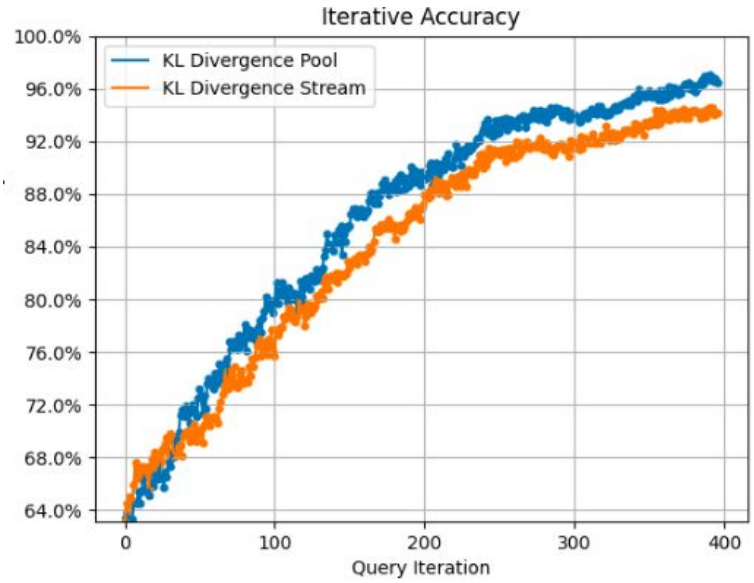
Final Accuracy of Pool based Vote Entropy: 0.9929292929292929
Final Accuracy of Stream based Vote Entropy: 0.9949494949494949
Final Accuracy of Pool based KL Divergence: 0.9646464646464646
Final Accuracy of Stream based KL Divergence: 0.9414141414141414

The pool and stream based vote entropy strategy are almost equally good, while in KL Divergence, the pool based active learning is better. The reasons why pool based works better are the same as explained in Q)1)b)i).

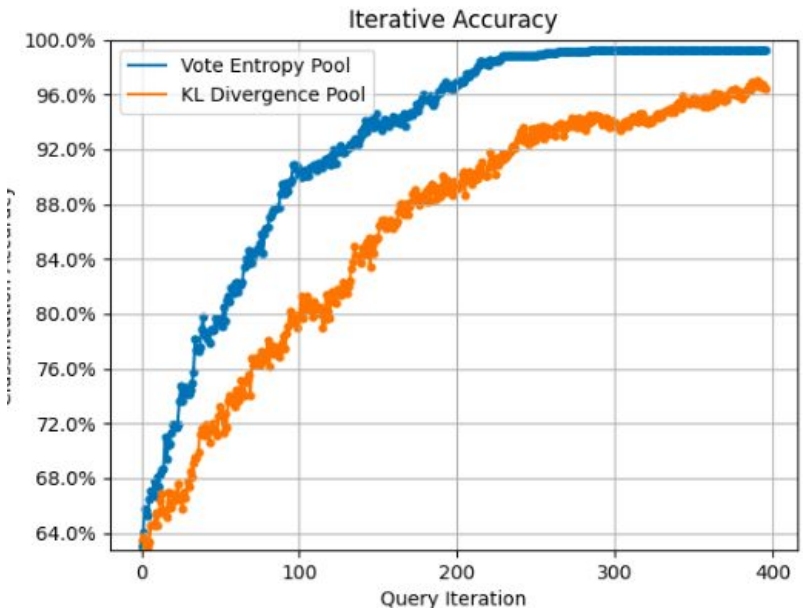
Also, it can be seen that vote entropy works better than KL Divergence in both pool based and stream based active learning.



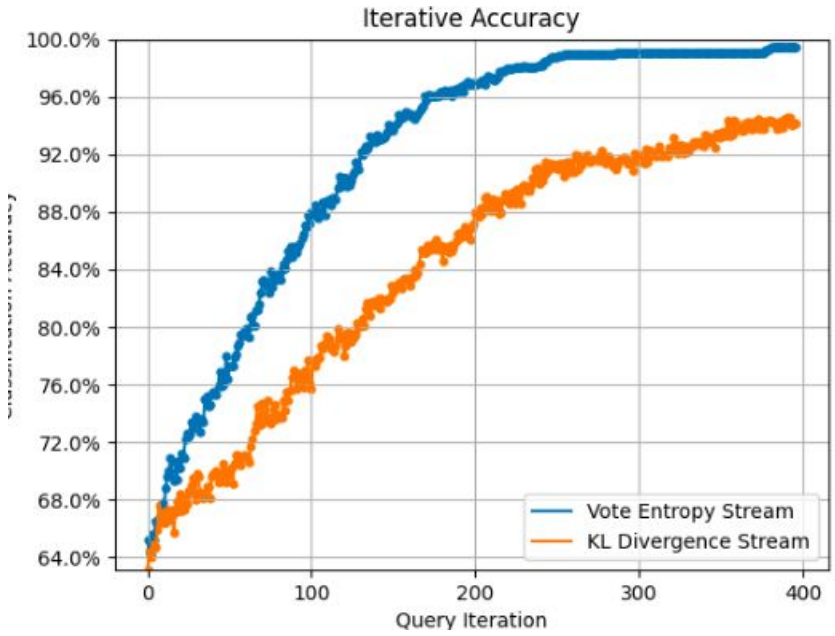
Comparing the accuracies of pool and stream based vote entropy



Comparing the accuracies of pool and stream based KL Divergence



Comparing the accuracies of pool based vote entropy and KL Divergence

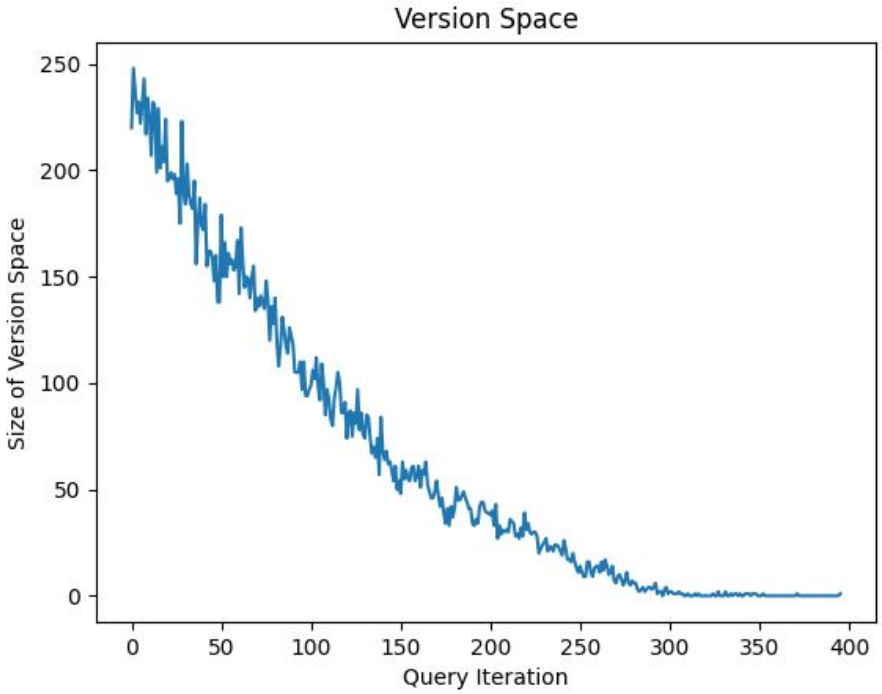


Comparing the accuracies of stream based vote entropy and KL Divergence

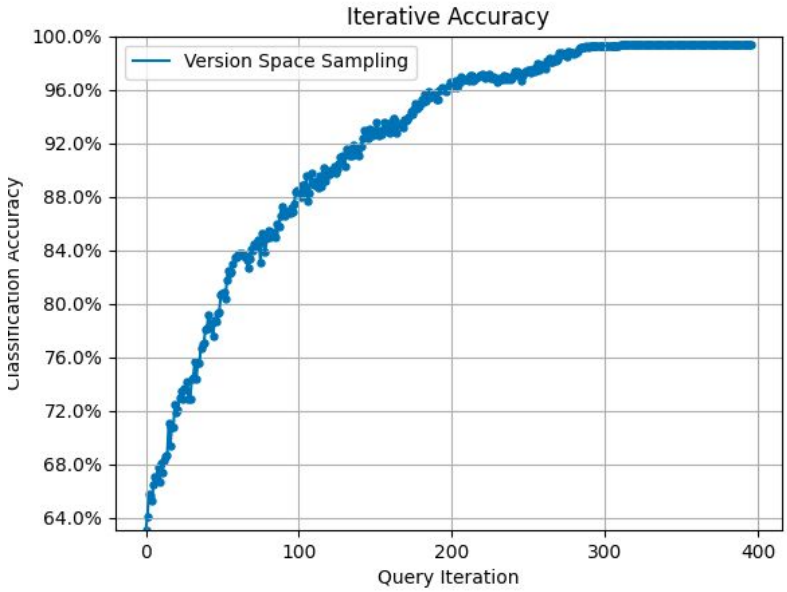
Q1)b)iii) As it is impossible to decide the point which would lead to the greatest reduction in the size of version space, a heuristic was used: that point was chosen to query, which had the greatest no. of committee members disagreeing. The argument is that, by labelling this point, the most number of committee members would adjust themselves and converge to the right position, hence leading to more reduction in the version space.

Final Accuracy of Version Space Sampling: 0.9939

We can see that the accuracy is comparable to the techniques of Q)1)b)i) and Q)1)b)ii)



The decreasing trend in version space size for version space sampling

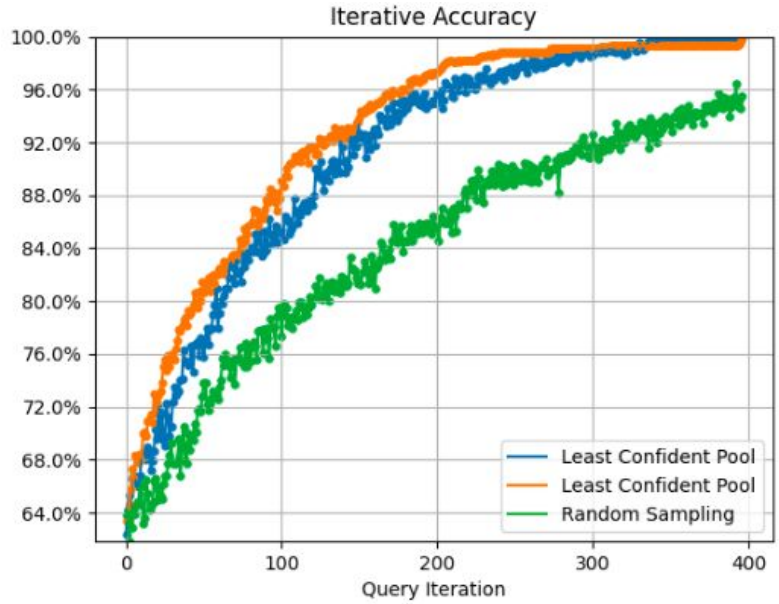


The accuracy plot for version space based sampling

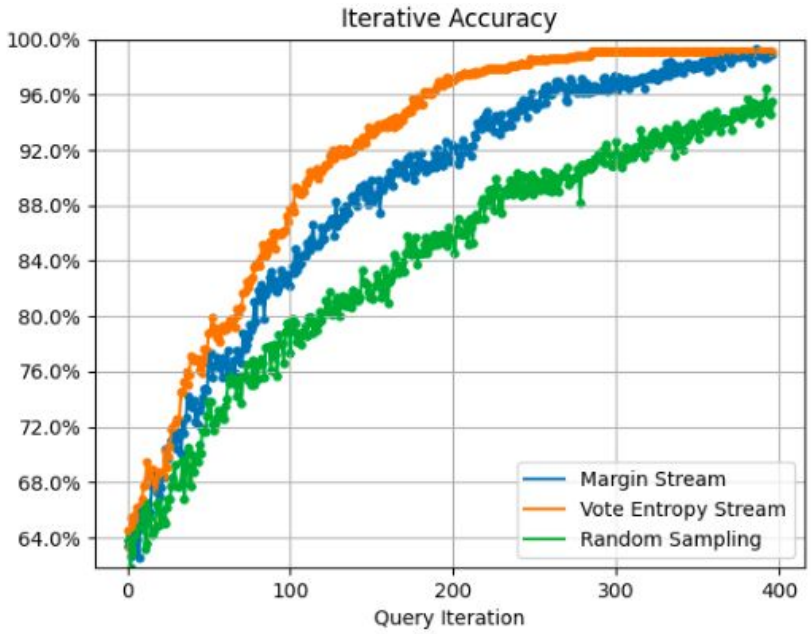
Q1)b)iv)

Final Accuracy of Random Sampling: 0.9373737373737374

We can see that the accuracy is lesser when compared to the strategies in Q1)b)i) and Q1)b)ii, as expected.



The accuracy plot for random sampling and the best of pool based uncertainty sampling (least confident) and QBC (vote entropy). (The orange line represents vote entropy pool: the legend is wrong)



The accuracy plot for random sampling and the best of stream based uncertainty sampling (margin) and QBC (vote entropy).

Q1)b)v)

```
Cluster Sampling Accuracy is: 0.5949494949494949
Cost saved = Rs.81400.0
Hours saved:  814.0
```

It can be observed that the final accuracy is worse than that of the normal 10% labelled dataset. This is a very surprising result, as accuracy is supposed to increase as the labelling increases. A closer look at the results of the k-means clustering explains this poor behaviour. The class labels of the 11 clusters are as follows (They range from 0 to 10):

```
[array([0, 1, 0, 1, 2, 2, 2, 0, 1, 2, 1, 1, 1, 2, 2,
1, 1, 2, 2, 2, 0, 2,
       1, 1, 0])

 array([ 9,  8,  4,  7,  8, 10,  6,  7,  8,  9,  9,
4,  6,  7,  8, 10,  6,
        8,  9,  8,  8,  8,  9,  6,  7,  8,  6,  9,  6,
7,  9,  4,  4,  9,
        6,  7,  8,  8,  8,  8])

 array([ 4,  5,  4,  4,  5,  6,  4,  6,  6,  5,  3,
6, 10,  4,  5, 10, 10,
        5,  5,  4,  4,  5, 10,  6, 10,  6,  3, 10, 10,
5,  5,  6,  5,  5,
        4, 10,  3,  6,  3, 10])

 array([0, 0, 1, 2, 1, 1, 2, 0, 2, 1, 2, 0, 1, 0, 2,
9, 0, 2, 0, 2, 2, 0,
       1, 1, 2, 2, 1, 2, 2, 0, 2, 0, 0, 2, 2, 2, 0,
1])
```

```
 array([ 4,  3,  4,  5,  5,  5,  4,  3,  3,  4,  3,
3,  3,  3,  5,  3,  5,
        3,  3,  5,  5,  3,  5,  3,  3, 10,  3, 10,  3,
4,  4, 10,  5,  3,
        2,  3])

 array([ 2, 10,  8,  2,  3,  4,  4,  5,  4,  7, 10,
8,  6,  2,  4, 10,  3,
        3,  8,  2, 10,  2, 10, 10,  5,  3,  3,  8,  2,
3, 10, 10, 10,  3,
       10,  5,  2,  4, 10, 10,  2,  2, 10, 10, 10,
2])

 array([6, 8, 8, 7, 4, 5, 9, 7, 6, 6, 8, 7, 6, 7, 8,
8, 8, 8, 7, 7, 7, 8,
       6, 8, 7, 8, 6, 6, 6, 8])

 array([ 9,  0,  1,  9,  1,  9,  2,  1,  0,  0,  0,
10,  1,  1,  9,  9,  1,
        9,  1,  0,  1,  0,  0,  1,  9,  9,  0,  1,  1,
0,  9,  0])

 array([9, 8, 9, 9, 7, 8, 9, 9, 9, 8, 8, 9, 9, 9, 9,
9, 9, 9, 7, 8, 7, 9,
       7, 9, 9, 7, 9, 9, 9, 8, 7, 7, 8, 9, 9, 7, 9,
9])

 array([ 8, 10, 10,  8,  5,  5,  1,  8, 10, 10,  2,
1, 10,  9, 10,  9, 10,
       10, 10,  8,  1,  1,  1,  9,  1, 10,  9,  5,  1,
5, 10, 10,  5,  8,
        8,  8])

 array([6, 6, 6, 4, 8, 7, 8, 6, 6, 5, 9, 6, 5, 4, 6,
6, 4, 7, 5, 8, 6, 8,
       6, 4, 4, 9, 4, 4, 7, 6, 4, 4, 5, 6, 6])]
```

It can clearly be seen that the clustering results are not great. There is a lot of intermixing of class labels in the clusters. So, if 20 percent of each cluster is sampled to label the whole cluster, the resulting labelling would be very inaccurate, thus leading to the very poor accuracy. Much better results may be expected for datasets, which can be clustered properly using k-means. To test this, we use the cluster based sampling on iris dataset, which can be clustered well using k-means. The results are as follows.

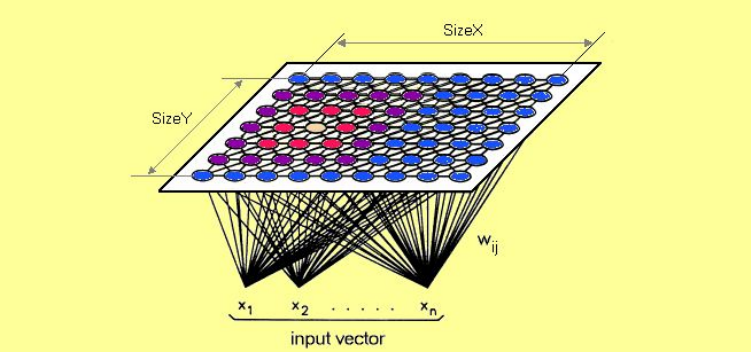
```
Cluster Sampling Accuracy is: 0.94
Cost saved = Rs. 12400.0
Hours saved: 124.0
```

As can be seen, the results are much better, thus validating the usefulness of active learning with cluster based sampling.

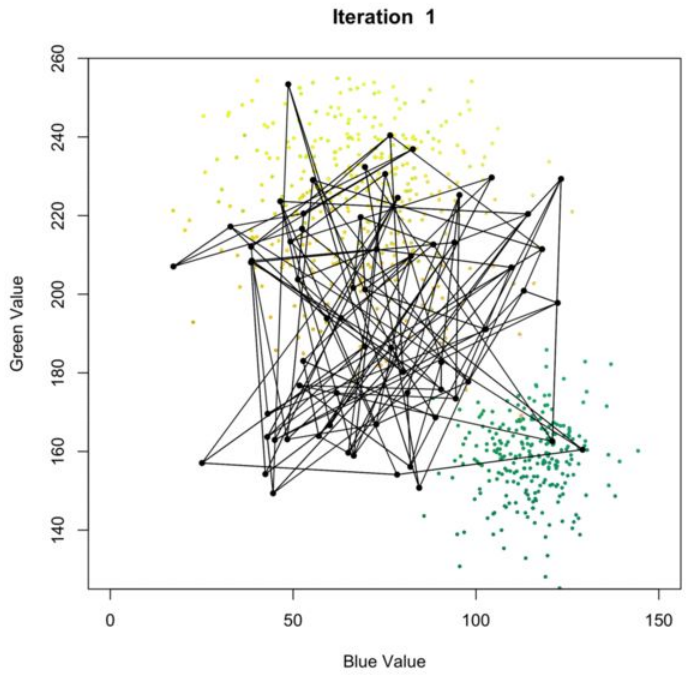
Self Organizing Map (SOM):

A *self-organizing map* (SOM) is a clustering technique that helps you uncover categories in large datasets, such as to find customer profiles based on a list of past purchases. It is a special breed of unsupervised neural networks, where neurons (also called *nodes* or *reference vectors*) are

arranged in a single, 2-dimensional grid, which can take the shape of either rectangles or hexagons.



Through multiple iterations, neurons on the grid will gradually coalesce around areas with high density of data points. Hence, areas with many neurons might reflect underlying clusters in the data. As the neurons move, they inadvertently bend and twist the grid to more closely reflect the overall topological shape of our data. Look at the below figure for an example.



SOM ALGORITHM:

- Step 0: Randomly position the grid's neurons in the data space.

- Step 1: Select one data point, either randomly or systematically cycling through the dataset in order

- Step 2: Find the neuron that is closest to the chosen data point. (position of the neuron is represented by it's input weight vector)

- Step 3: Move this neuron closer to that data point (i.e update the input weight vector). The distance moved is determined by a learning rate, which decreases after each iteration.

- Step 4: Move its neighbors closer to that data point as well, with farther away neighbors moving less (i.e update the input weight vector). Neighbors are identified using a radius around it, and the value for this radius decreases after each iteration.

- Step 5: Update the learning rate and the neuron's radius. These steps can be iterated over as required.

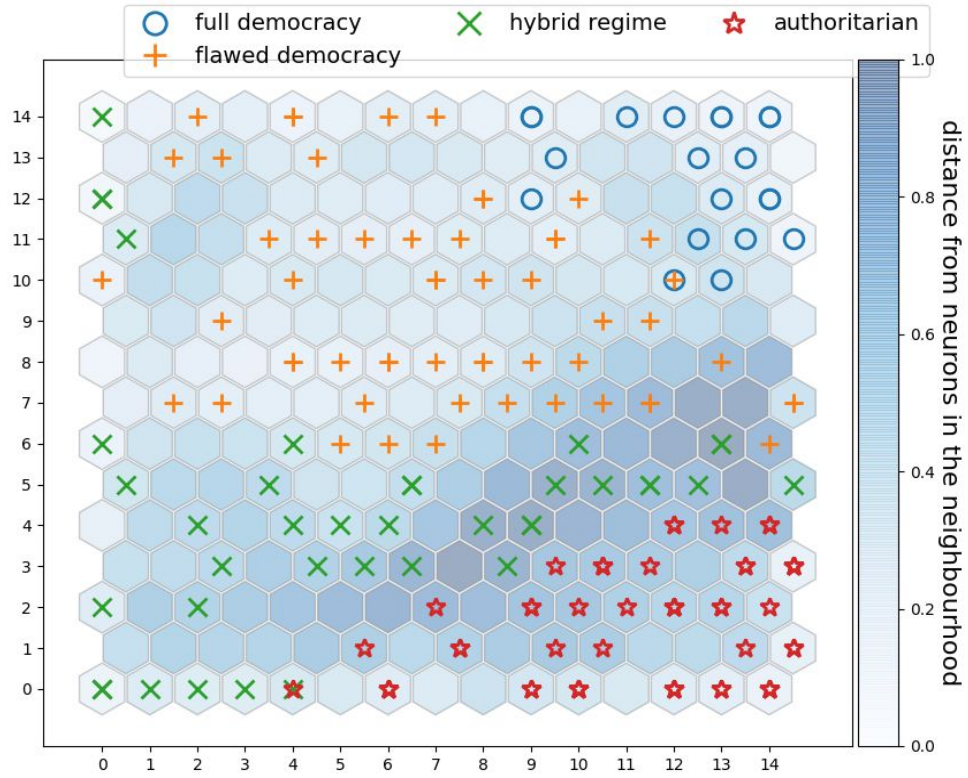
Results:

ARCHITECTURE USED: An SOM architecture of 15 x 15 output nodes is used. Each output node is connected to all the 6 input nodes (each input node represents one feature). Learning rate is 0.7, distance function is euclidean and the neighborhood function is Guassian.

The hexagonal grid plot is plotted using the “Democracy Index” dataset which uses 4 classes, 6 attributes and 168 data points. It is the file “data.csv” present in the same folder as this report.

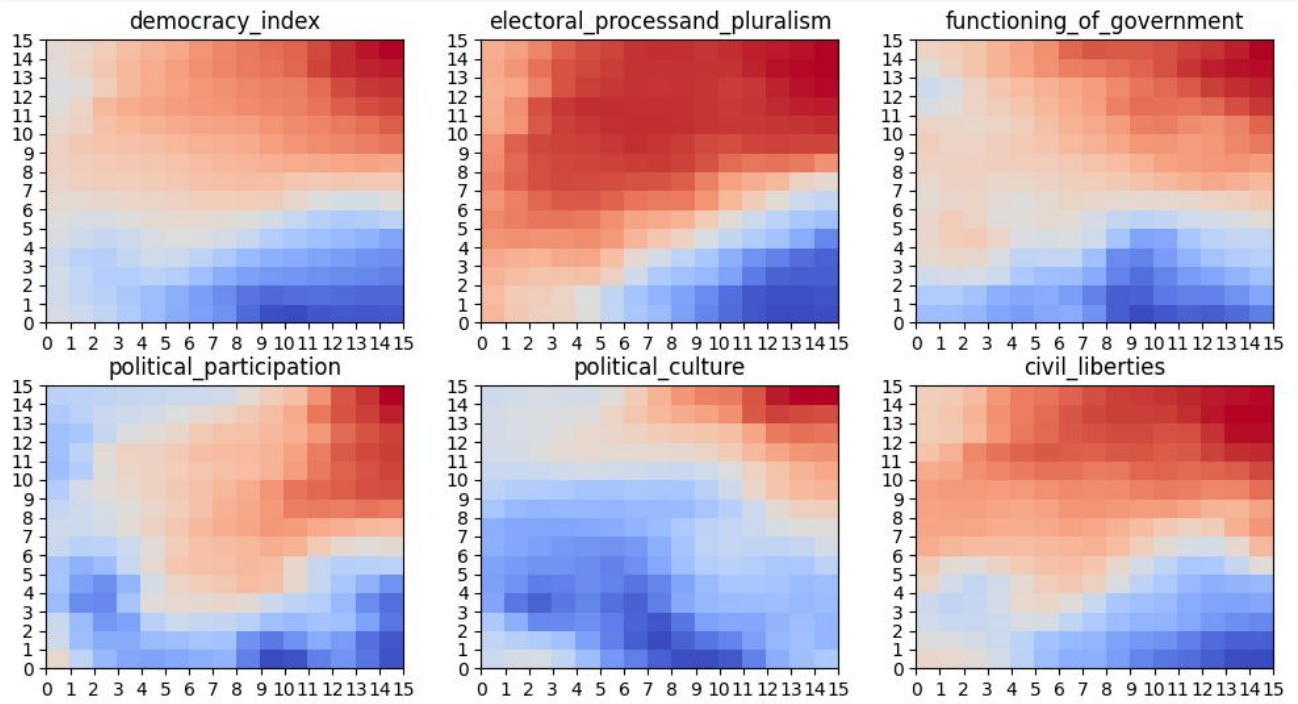
quantization error: 0.9831757122260334

It can be seen how all the 4 classes are suitably separated out into different regions. It can also be seen that in the inter-class regions, the coloring is darker, compared to intra-class regions. This indicates that inter-cluster distances are greater than intra-cluster distances, which is a very suitable property for clustering results.

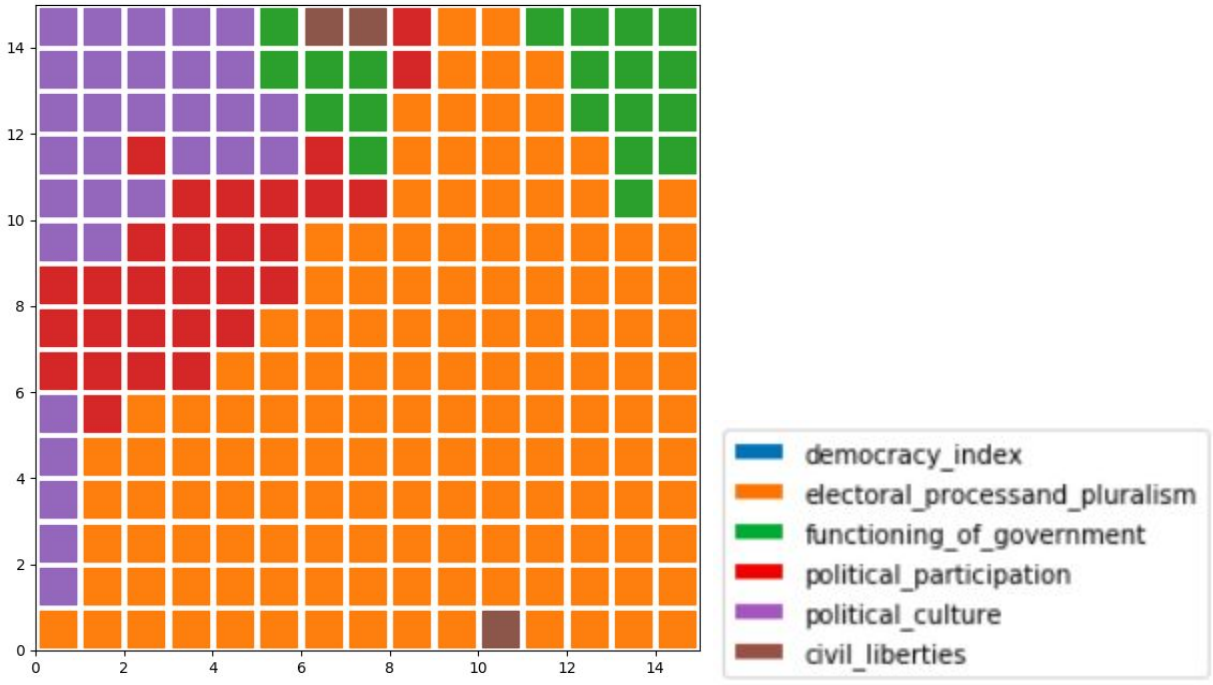


The feature planes of all the 6 features are as shown below. The red color indicates a high value, while blue indicates a low value. It can be seen that all have higher values at the upper right corner. The corresponding class in the upper right corner of the hexagonal grid above is that of a “full democracy”. So, it can be concluded that a full democracy will need all the features to be as high as possible.

The characteristics of the rest of the classes can also be interpreted similarly, from the feature maps below.



The most relevant feature map (the features which contribute the maximum to each output node) is given below.



References

- Active Learning Tutorial
- Introduction to Active Learning
- Active Learning on MNIST — Saving on Labeling
- https://www.cs.cmu.edu/~tom/10701_sp11/recitations/Recitation_13.pdf
- https://cran.r-project.org/web/packages/som/som.pdf
  https://clarkdatalabs.github.io/soms/SOM_NBA
  https://wonikjang.github.io/deeplearning_unsupervised_som/2017/06/30/som.html
- https://www.kaggle.com/lmzentner/som-mnist-digit-recognizer
  http://burrsettles.com/pub/settles.activelearning.pdf
- https://algobeans.com/2017/11/02/self-organizing-map/