Machine Learning Assignment # 2

June 10, 2020

Submitted by

Rishav 2016A7TS0108P
Rishabh Jain 2016B4A70729P
Vibhav Oswal 2016B4A70594P

Submitted to
Prof. Navneet Goyal
For partial fulfillment of the course
BITSF464- Machine Learning

Contents

1 Introduction

1.1 Part 1: Active Learning

1.1.1 Active Learning

Active Learning is a semi-supervised machine learning strategy which can be used for several algorithms. Labeling data is an expensive task, hence we want to learn as much as possible with as little data as possible, this is where active learning comes into picture. Suppose we have a huge pool of unlabelled data and a very small amount of labelled data, we can achieve good performance for a machine learning algorithm with that small amount of data and that unlabelled data. We firstly learn from whatever labelled data we have, then depending upon the 'query' strategy we ask the Human oracle only to label the data which is most 'informative' to us.

There are several query strategies available, but before going deep into any of them, let us first discuss the two most popular active learning methodologies, namely, stream based and pool based active learning methods. A small introduction about various query strategies would also be explained later.

1.1.2 Stream Based Learning

In this setting for active learning, we make an assumption that getting an unlabelled instance is free. Based on this assumption, we then select each unlabelled instance one at a time and allow the learner to determine whether it wants to query the label of the instance or reject it based on its 'informativeness'. To determine informativeness of the the instance, we use a query strategy (explained in the following section). Following with the example above, we would select one data point from the set of unlabelled data points, determine whether it needs to be labelled or discarded, and then repeat with the next data point.

1.1.3 Pool Based Learning

In this setting, we assume that there is a large pool of unlabelled data, as with the stream-based selective sampling. We draw instances from the pool according to some 'informativeness' measure. This measure is applied to all instances in the pool (or some subset if the pool is very large) and then the most informative instance(s) are selected. This is the most common scenario in the active learning. All the unlabelled data points in the pool will be ranked and then the best (most informative) instance(s) will be selected and their labels requested.

1.1.4 Query Strategies

Query strategies can be broadly divided into two main categories, namely Query by committee & uncertainty sampling, these two categories can further be divided into several other categories. Some of them which have been used in the experiments are discussed here:

- **Query By Committee** It is a selection algorithm that is based on forming a committee of models trained on same labelled data that have competing hypotheses.The best query is decided based on degree of disagreement between models. The following are the methods to measure this disagreement:

  - KL-Divergence: This method selects that point which has the highest difference between the label distributions of any one model in the consensus and the committee.

$$X^*_{KL} = argmax_x \frac{1}{c} \sum_{c=1}^{c} D(P_\theta || P_C) \tag{1}$$

    where

$$D(P_\theta || P_C) = \sum_i P_\theta(y_i|x) \log \frac{P_\theta(y_i|x)}{P_c(y_i|x)} \tag{2}$$

    given

    $\theta : model$, C: committee size and $P_c(y_i|x)$ is the consensus probability that y_i is the correct label.

  - Vote-By-Entropy: Each instance has a associated probability distribution to it. The distribution for which the calculated value of entropy(shown below) is largest is selected.

$$x^*_{VE} = argmax_x - \sum_i \frac{V(y_i)}{C} \log \frac{V(y_i)}{C} \tag{3}$$

    where y_i: label, $V(y_i)$: Vote and C: committee size

- **Uncertainty Sampling**

  - Entropy Sampling: The sample with the highest entropy value is selected for querying. Entropy for samples are calculated as:

$$H(x) = -\sum_k p_k \log(p_k) \tag{4}$$

  - Least Confidence Sampling: The sample to be used for querying is selected based on least value of uncertainty which is calculated as:

$$U(x) = 1 - P(\hat{x}|x) \tag{5}$$

    x:instance to be predicted,$\hat{x}$:most likely prediction

  - Margin Sampling: Sample is selected based on the least difference between the probabilities of first and second most likely prediction.

$$M(x) = P(\hat{x1}|x) - P(\hat{x2}|x) \tag{6}$$

1.1.5 Labeling Strategy

Cluster based labeling can also save us a lot of cost and time. This is specially true when we don't care about the accuracy of our model way too much. An experiment is done in Section 4 for the same.

1.2 Part 2: Unsupervised Learning

Unsupervised learning refers to training machine learning algorithms without the use of labelled data. We need to figure out the underlying structure inside of the data to train a model and come up with an inference when a new data point comes in. A very famous unsupervised learning algorithm is discussed below.

1.2.1 Self Organising Maps

A self-organizing map (SOM) is a type of artificial neural network that is trained using unsupervised learning to produce a low-dimensional, discretized representation of the input space of the training samples, called a map, and is therefore also a method to do dimensionality reduction. Self-organizing maps differ from other ANNs as they apply competitive learning as opposed to error-correction learning (such as backpropagation with gradient descent), and in the sense that they use a neighborhood function to preserve the topological properties of the input space. The exact algorithm is presented in the SOM section.

The next sections have solutions to problems asked in the assignment. Q1 (i) to (iv) is answered via Section 2 & 3, Q1 (iv) is answered in Section 4, while Section 5 answers Q2 of the assignment.

2 Stream Based Active Learning Experiments

Stream based active learning algorithm can be described as the best case scenario for active learning. Getting an unlabeled instance is definitely not free all the time which is the case with stream based active learning approach. In this section, we present results of stream based active learning when **SVM classifier** was **actively** trained on **Digits** dataset which is described as follows:

- Dataset Name: Digits
- Training Size: 7493 samples
- Test size: 3497 samples
- Classes: 10 (0-9 digits)
- Attributes: 16

10% of data is initially sampled from the training set to for the learner. An SVM algorithm is fitted on the whole training dataset (7493 samples) inorder to determine a rough threshold value for our stream based learning approach. The accuracy of test set for the SVM trained on whole dataset was approx **98%**, hence we keep our **Threshold** value at **97%** for the active learning process, i.e., our algorithm learns until it obtains the required accuracy. We only use those points for labelling whose classifier uncertainty is above a certain value. Following sections have the results with various query strategies. We also use random sampling strategy for training our algorithm

2.1 Results with Uncertainty Sampling

Table 1 presents the results with various uncertainty query strategies. Graph 1 also represents the process pictorially. From this analysis margin sampling strategy is able to surpass the required threshold with the minimum number of extra points. Hence the order of new points needed is

Margin Sampling < Least Confident < Entropy Sampling << Random Sampling

Concludingly we were able to obtain similar test accuracy as KNN trained on the whole dataset (7493 points) with just 1006 points in the case of Margin Sampling & with 1066 points, 1018 points in case of entropy sampling & least confident respectively. Random sampling performed much worse at 2588 points. Finally, this hold inline with the premise of active learning being able to work with limited data.

2.2 Results with Query By Committee

Table 2 presents the results with various QBC query strategies. Graph 2 also represents the process pictorially. From this analysis vote entropy strategy is

Table 1: Additional points (Apart from 10% or 749 initial points) that were required to be labelled for obtaining performance close to the threshold, margin sampling is the clear winner.

Uncertainty Strategy	# of Additional Points required (Apart from initial 10%)
Margin Sampling	**257**
Least Confident	269
Entropy Sampling	317
Random Sampling	1839

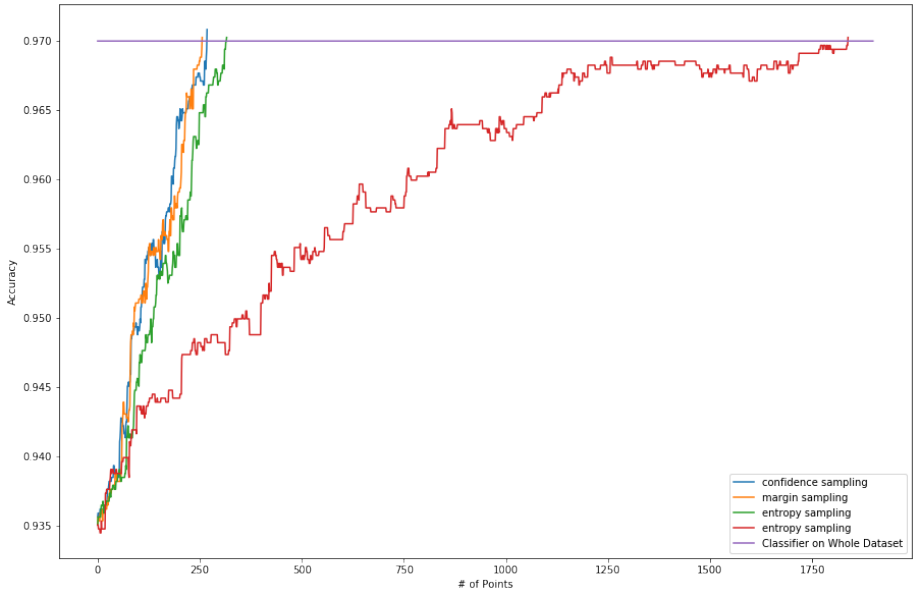

Figure 1: Pictorial Depiction of the Training of the Active learning algorithm for SVM when using various uncertainty strategies, as evident, classifier margin requires the least amount of new data points for performance similar to the threshold.

Table 2: Additional points (Apart from 10% or 749 initial points) that were required to be labelled for obtaining performance close to the threshold, vote entropy is the clear winner.

QBC Strategy	# of Additional Points required (Apart from initial 10%)
KL-Divergence	60
Vote-Entropy	**48**
Random Sampling	404

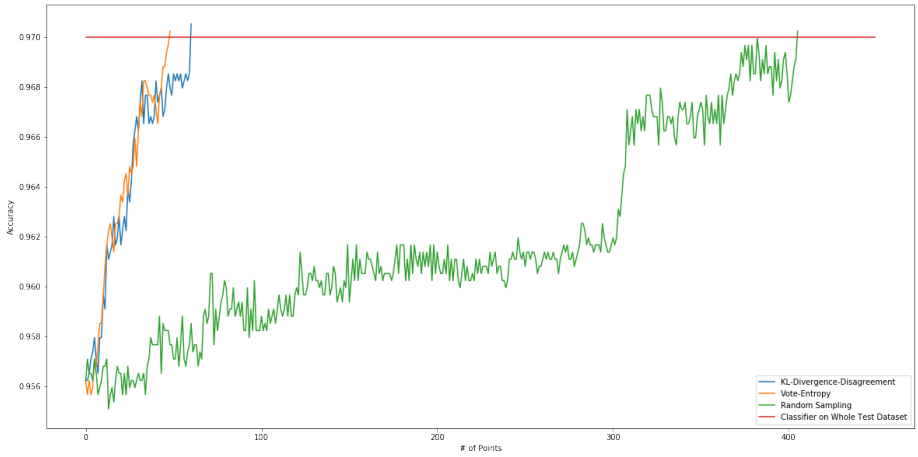

Figure 2: Pictorial Depiction of the Training of the Active learning algorithm for SVM when using various QBC strategies, as evident, vote entropy requires the least amount of new data points for performance similar to the threshold

able to surpass the required threshold with the minimum number of extra points. Hence the order of new points needed is

Vote-Entropy < KL-Divergence << Random Sampling

Table 3: Results for pool based active learning for different uncertainity sampling strategies with every 10% additional points being labelled at each iteration. *this table partially answers (ii) and (iv) from Q1.*

Strategy	Initial Accuracy	10% add	20% add	30% add	40% add
Least Confident Sampling	93.22	98.11	98.14	98.14	98.14
Margin Sampling	95.36	**98.39**	98.16	98.16	98.16
Entropy Sampling	93.36	98.31	98.16	98.16	98.16
Random Sampling	94.11	97.11	97.88	97.94	98.0

3 Pool Based Active Learning Experiments

This active learning approach is a more realistic approach where we request the label for most informative instances from a pool and we don't have an assumption that all the unlabelled data comes for free. The data-set & algorithm used for experiments in this section is the same as that was used in the previous section. Hence the details are available in the previous section itself. In experiments involved with pool based active learning, on every query we request for the most informative 10% data from out pool based on various strategies and teach our algorithm on that dataset. The following sections describe the results with various querying strategies.

3.1 Results with Uncertainty Sampling

Table 3 presents the results with various uncertainty query strategies. Graph **??** also represents the process pictorially. We also use a random sampling strategy for labeling in order to show the efficacy of using mathematically chosen strategies over naively chosen random points. Overall, after sequentially adding 10% points after each query till we labelled additional 40% of points, we were able to get the following result in terms of accurracy.

Entropy Sampling $\approx$ **Margin Sampling** $>$ **Least Confident Sampling** $>>$ **Random Sampling**

obviously there were stages were one or the other algorithm was better. The Detailed result is presented in Table 3.

3.2 Results with Query By Committee

The results for query by committee methods compared with random sampling is presented in Table 4 and Figure 4, the query by committee methods were not able to perform as good as uncertainty based methods but overall the performance was satisfactory. The query process remains the same as previous section with 10% additional data at each iteration. After incorporation of 40% additional points the following was the order of accuracy:

KL-Divergence $\approx$ **Vote Entropy** $>>$ **Random Sampling**

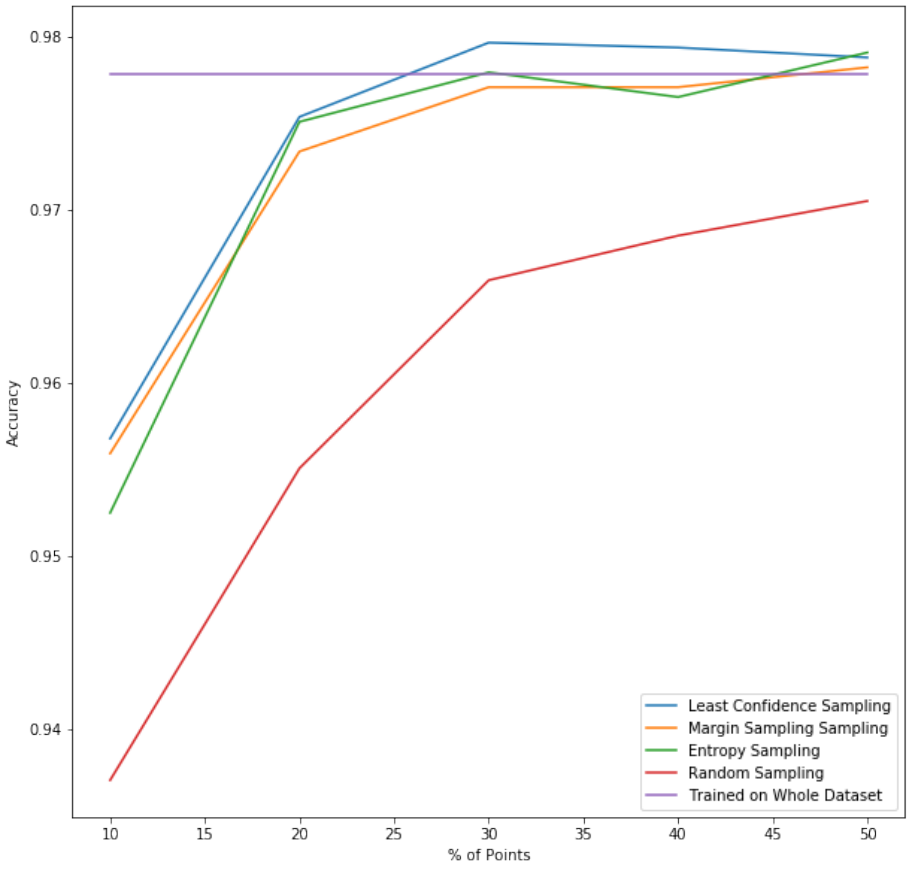

Figure 3: The accuracy trend of various uncertainty sampling strategies, with every query, 10% additional data is labelled, as is evident, sometime Least confident strategy works better, sometimes the some other, but one thing is clear the performance got even better than the original classifier trained on the whole dataset. Random sampling failed and performed consistently worse than all methods.

Table 4: Results for pool based active learning for different QBC sampling strategies with every 10% additional points at each iteration *this table partially answers (ii) and (iv) from Q1.*

Strategy	Initial Accuracy	10% add	20% add	30% add	40% add
KL-Divergence	95.71	97.19	97.99	**98.19**	98.14
Vote Entropy	96.16	97.48	97.91	98.05	98.08
Random Sampling	95.65	97.6	97.90	97.99	97.99

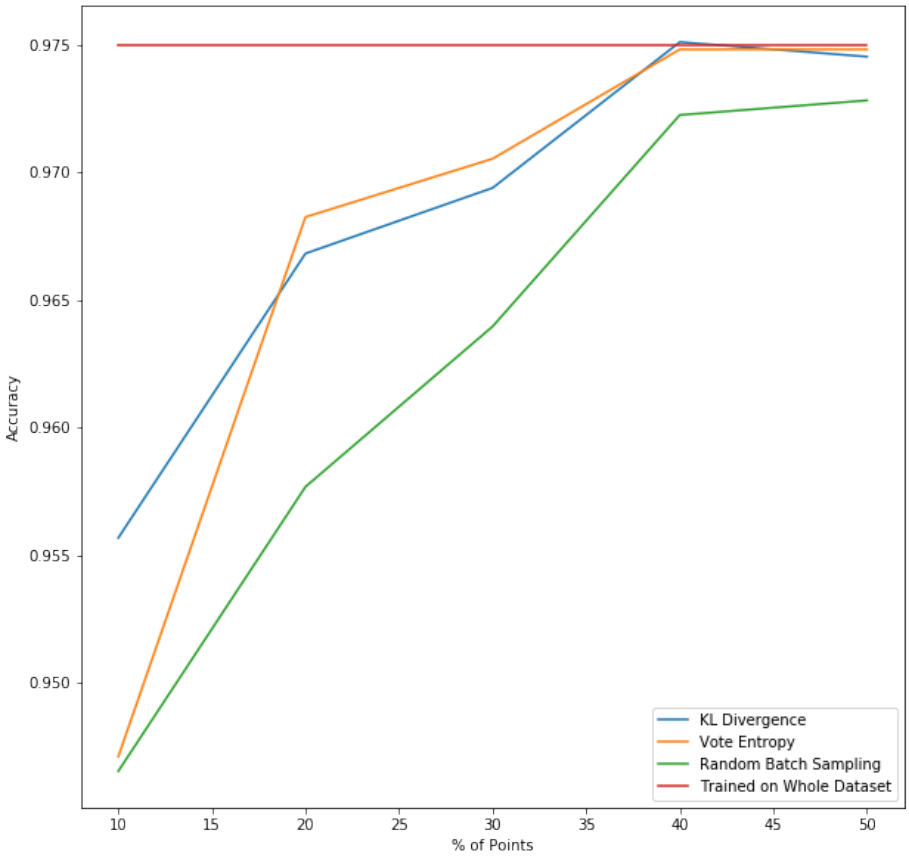

Figure 4: The accuracy trend of various QBC sampling strategies, with every query, 10% additional data is labelled, as is evident, sometime KL-divergence strategy works better, sometimes the other, but one thing is clear the performance got even better than the original classifier trained on the whole dataset. Random sampling failed and performed consistently worse than all methods.

3.3 Version Space

Version space is defined as the region which is still unknown to the model. In QBC, we can see that a certain set of 10% points is queried at each iteration which reduces the version space. From the curve it is evident that the at the 3rd iteration (30% additional points) we have the maximum accuracy, hence the least number of regions which are unknown. Hence when we start with the QBC process, the version space is approx. 30% of the remaining points,i.e., approx. 2200 points.

```
print("Accuracy of cluster based labeling: %.2f"%np.mean(correct_preds))
print("The cost saved, saved time due to the labeling strategy respectively: Rs %.2f %.2f Hours"%(find_cost(100,1))
```

```
Accuracy of cluster based labeling: 0.81
The cost saved, saved time due to the labeling strategy respectively: Rs 216200.00 2162.00 Hours
```

```
#Now we find test accuracy when model was fitted on cluster based labelled data & original_data
original_knn = KNeighborsClassifier(n_neighbors=3).fit(x_90_40,y_90_40)
cluster_fitted_knn = KNeighborsClassifier(n_neighbors=3).fit(x_90_40_cluster_labelled,y_90_40_cluster_labelled)
print("Test Accuracy of KNN when fitted with orginally labelled data: %.2f"%(original_knn.score(x_tes,y_tes)))
print("Test Accuracy of KNN when fitted with data after cluster labeling: %.2f"%(cluster_fitted_knn.score(x_tes,y_t
```

```
Test Accuracy of KNN when fitted with orginally labelled data: 0.97
Test Accuracy of KNN when fitted with data after cluster labeling: 0.72
```

Figure 5: Kmeans Cluster Labeling Results on Digits

4 Cluster Labeling

4.1 KMeans Clustering

KMeans aims at identification of k centroids and then allocation of data points to one of these clusters while keeping the centroids small. The algorithm is(ref: *towarsdatascience.com*):

- Specify number of clusters K.
- Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement.
- Compute the sum of the squared distance between data points and all centroids.
- Assign each data point to the closest cluster (centroid).
- Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.
- Keep iterating until there is no change to the centroids. i.e assignment of data points to clusters isn't changing.

4.2 Results

We were able to obtain **81%** accurate cluster labeling on Digits dataset with the given strategy. We fitted a KNN classifier on the original 40% of the data and the data that we obtained after cluster labeling. KNN after fitting on our cluster labelled data was able to obtain an accuracy of **72%** on test data while the original one was able to obtain an accuracy of **97%** on the original data. The code for this has been provided in the form of a jupyter notebook along with this assignment. A snap shot of the result is also shown below.

5 Self Organising Maps

The algorithm for SOM is (ref: *towardsdatascience.com*):

- Each node's weights are initialized.
- A vector is chosen at random from the set of training data. Every node is examined to calculate which one's weights are most like the input vector. The winning node is commonly known as the Best Matching Unit (BMU).
- Then the neighbourhood of the BMU is calculated. The amount of neighbors decreases over time.
- The winning weight is rewarded with becoming more like the sample vector. The nighbors also become more like the sample vector. The closer a node is to the BMU, the more its weights get altered and the farther away the neighbor is from the BMU, the less it learns.
- Repeat step 2 for N iterations.

5.1 Results & Visualizations

We fit pendigits dataset (the dataset has been described in Section 2) on our 30x30 SOM. The various results that came out of it are:

The Hexagonal Grid for clusters is shown in Figure 6 and the node differences for the same in Figure 7.

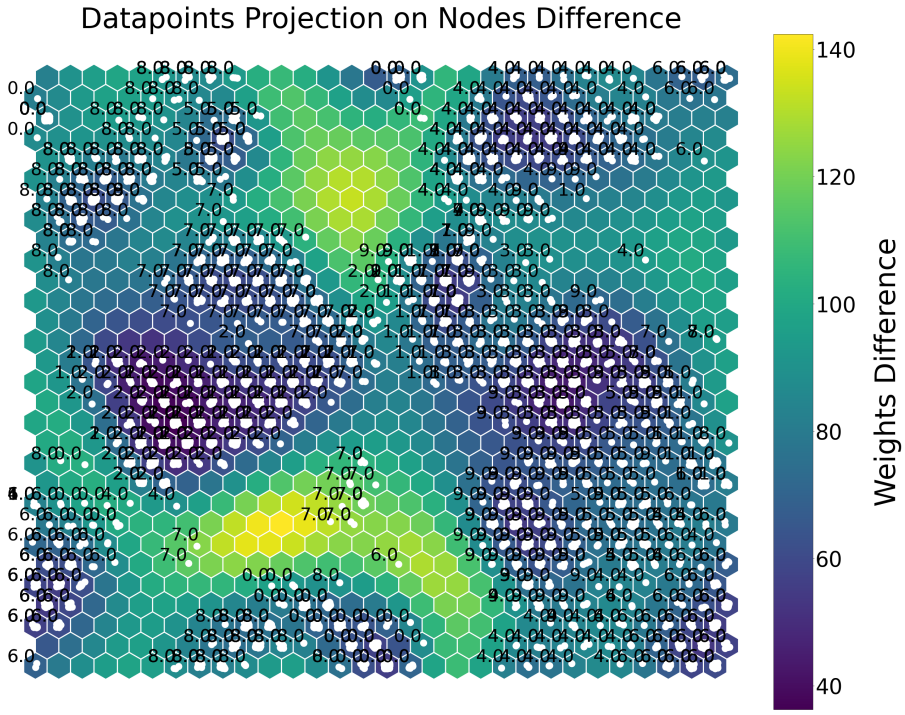

Figure 6: The grid after matching with real labels, except a few most of the points are clustered accurately.

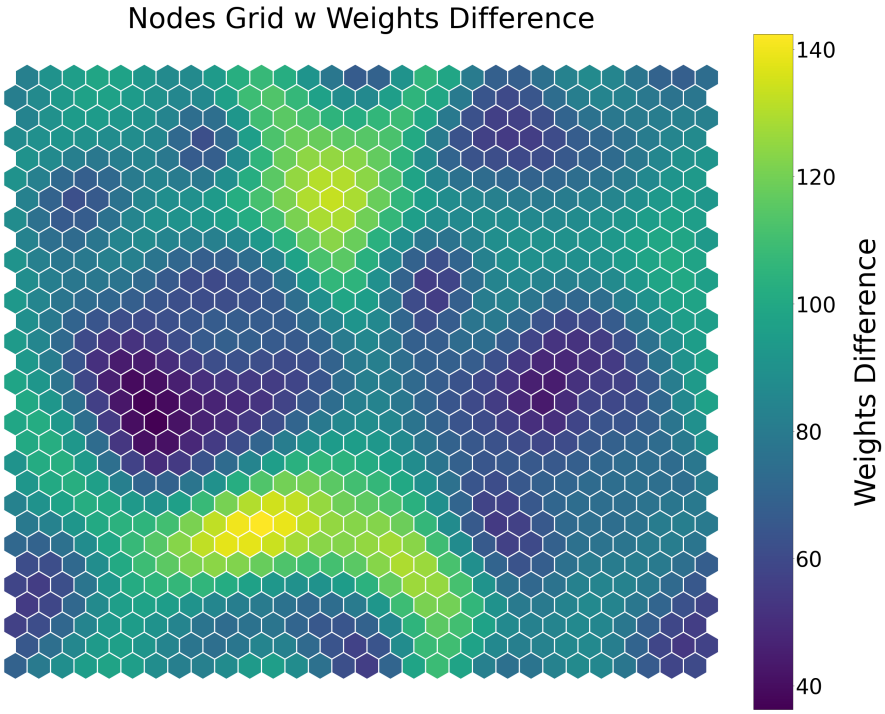

Figure 7: The weight difference grid, as can be seen, nodes in a cluster have very less weight difference in comparison to nodes not belonging to a specific cluster.