BITS F464 - Machine Learning
I Semester 2022-2023
Assignment : 2
Active Learning Implementation

Professor : Dr. Navneet Goyal
Group Members :

- Ujjwal Jain (2019B4A70647P)
- Shobhit Jain (2019B3A70385P)

ACKNOWLEDGEMENT

We express gratitude to Dr. Navneet Goyal for his immense effort in guiding us and providing us the required resources and knowledge.

My sincerest gratitude to Birla Institute of Technology and Science, Pilani for providing me with such a great learning opportunity.

ABSTRACT

The objective of the assignment is to implement Active Learning with various Uncertainty Sampling methods like Least Confidence, Smallest Margin Sampling, Largest Margin Sampling and Entropy.

The assignment also implements Query by Committee with Vote Entropy and KL-Divergence as the measure of disagreement.

The assignment also compares the performance of various methods through learning curves.

The assignment concludes with explanation for the Stream Based Active Learning.

CONTENTS

Introduction to Active Learning

Active learning is a machine learning technique in which a model is trained to make predictions on data that is initially unlabeled. In active learning, the goal is to minimize the amount of data that needs to be labeled by a human in order to train an accurate model.

The key difference between active learning and traditional supervised learning is that in active learning, the model is able to select the data points that it wants to be labeled by a human. This is done using a variety of techniques, such as uncertainty sampling, in which the model selects the data points that it is least confident about, or query by committee, in which a committee of models makes predictions on the data and selects points that are predicted differently by the models in the committee.

After the selected data points have been labeled, they are used to train the model, allowing it to improve its performance on future predictions. This process is repeated iteratively, with the model selecting data points for labeling and using the labeled data to improve its performance, until the model reaches a satisfactory level of accuracy.

Overall, the goal of active learning is to use the collective knowledge of the model and the human labeler to train the model

effectively, using only the minimum amount of labeled data necessary. This can be especially useful when there is a limited amount of labeled data available, or when it is expensive or time-consuming to label large amounts of data.

Dataset : Fashion MNIST

Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples.
Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image.

Each training and test example is assigned to one of the following labels:

- T-shirt/top
- Trouser
- Pullover
- Dress
- Coat
- Sandal
- Shirt
- Sneaker
- Bag

- Ankle boot

Flow of Execution

The **languages used** in the **task are** C++ and **Python,** the base language **is C++.** All sampling, **QBC,** etc. are implemented in that. Python is only used for creating of different models for the task which requires it.

RUN "ActiveLearning.cpp" file to start the execution.

Query By Committee

In Python, 10% of the labeled points were randomly selected from a pool of labeled data. A panel of five deep neural networks (DNNs) were trained on this data, and the labels for the remaining 90% of the data were removed. Each DNN was configured differently. At this point, the accuracy of the panel was calculated.

In this process, we used a DNN to find the softmax output of all unlabeled data points, which was then sent in CSV format to a C++ program. This program performed the required QBC (query by committee) on the data and sent the indices of the most informative points, ranked in order, to a CSV file.
In Python, we then selected the first 10% of these data points from the CSV file and passed them to the DNN as additional training points. After this, 10% of the data points used for additional training were unlabeled, and the accuracy was calculated.
This process was repeated four times in total.

Voting Entropy and KL Divergence

Both QBC methods followed the same procedure. The only provision we took was that the first 10% of marked points is same in all cases for better comparison.

Vote Entropy

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

where y_i again ranges over all possible labelings, and $V(y_i)$ is the number of "votes" that a label receives from among the committee members' predictions,

KL Divergence

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

Here P and Q are two probability distributions, and the summation is over all possible values of i that is all the classes.

Here, for each data point we calculate the divergence and the higher KL divergence means that more informative/confusing that point is.

Uncertainty Sampling:

First, in Python, 10% of the labeled points are randomly selected from a pool of labeled data. A deep neural network (DNN) is trained on this data, and the labels for the remaining 90% of the data are removed. The DNN has the following configuration: 784 input neurons, 4 hidden layers containing 64, 32, 32, 16 neurons respectively, and an output layer containing 10 neurons.
We then used the DNN to find the softmax output for all unlabeled data points and sent them in CSV format to a C++ program. This program performed uncertainty sampling on the data and sent the indices of the most informative points, ranked in order, to a CSV file.

In Python, we picked the first 10% of these data points from the CSV file and sent them to the DNN as additional training points. After this, 10% of the data points used for additional training were unlabeled.

This process was repeated four times in total, using the same 10% of data for all three types of uncertainty sampling methods for better comparison between them.

Smallest Margin Sampling

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

Entropy Sampling

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

Least Confident Sampling

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

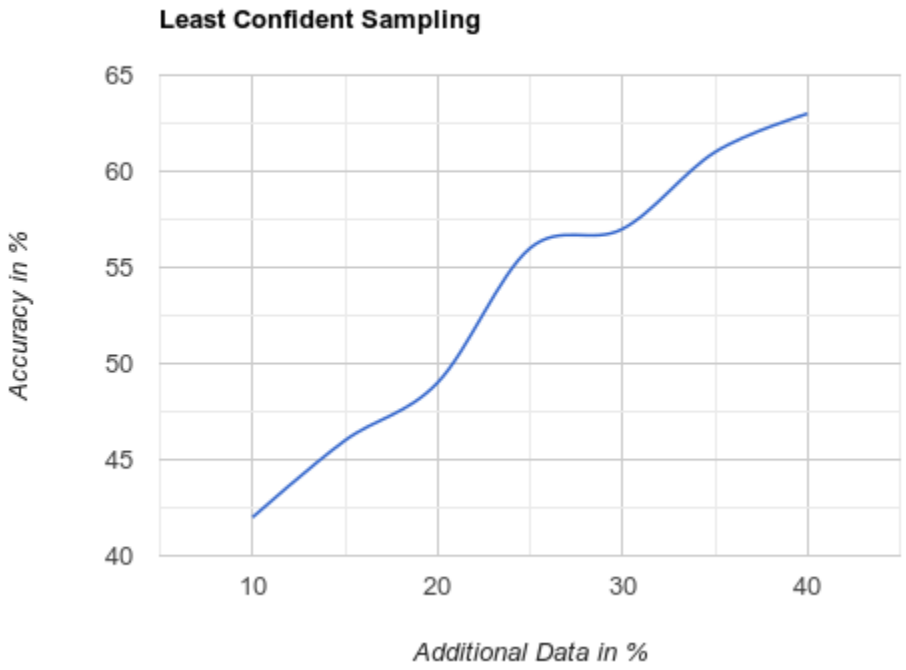
$$\text{where } \hat{y} = \text{argmax}_y P_\theta(y|x)$$

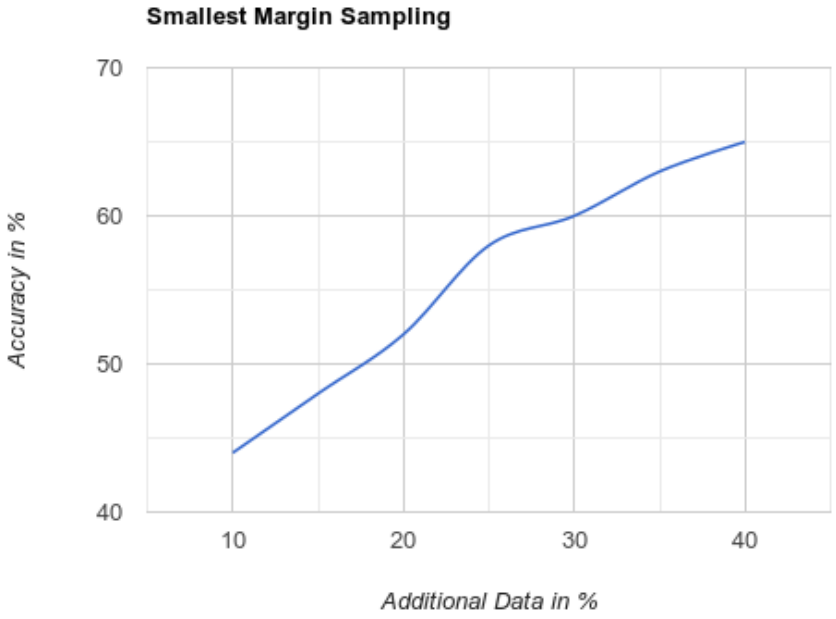
Largest Margin Sampling

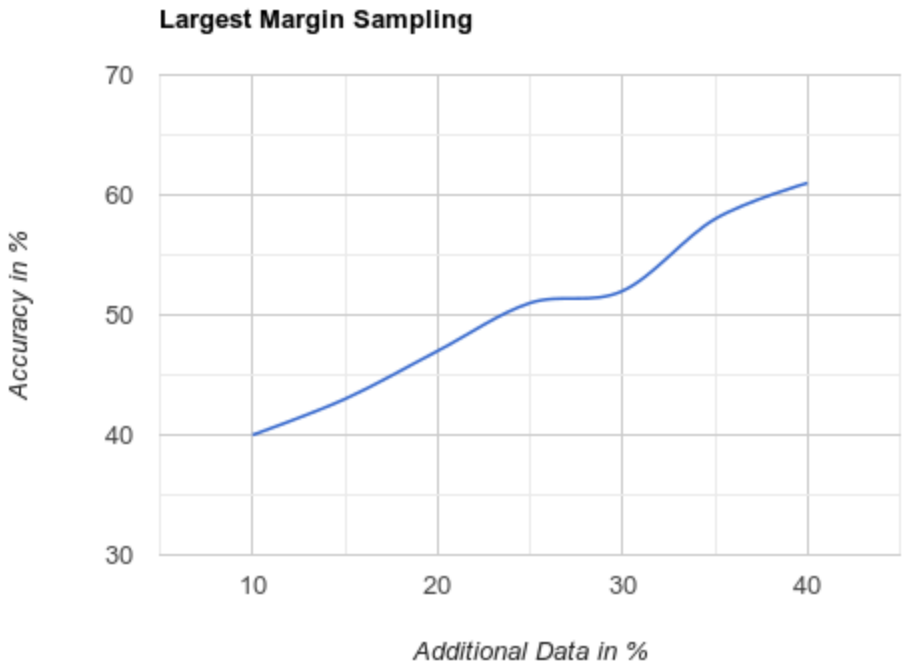
Here the difference between the maximum and minimum probability class for each data point is first obtained and the one with the minimum probability difference is given a higher rank.

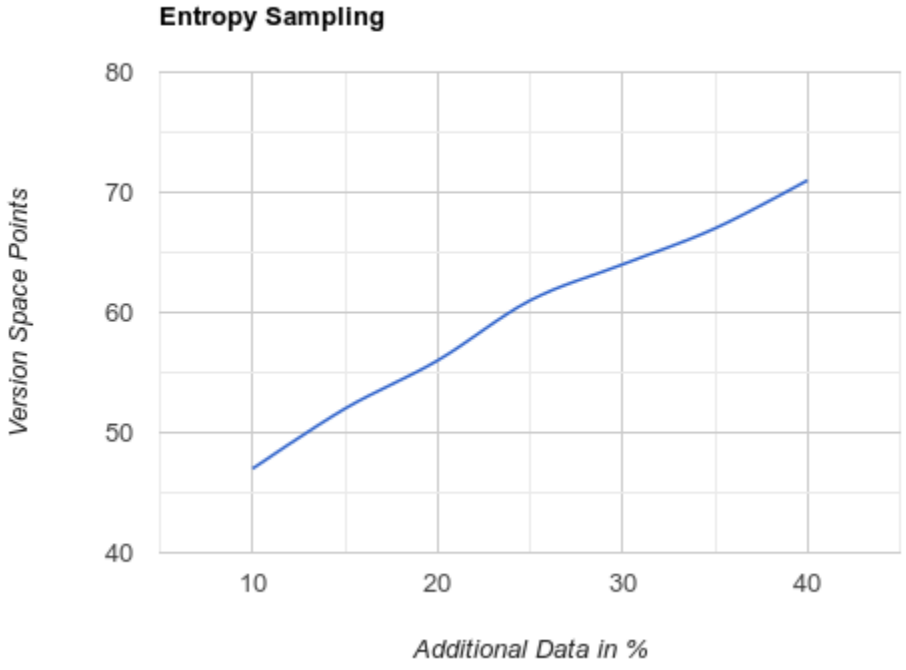
Results

The Learning Curves for the different methods of sampling :

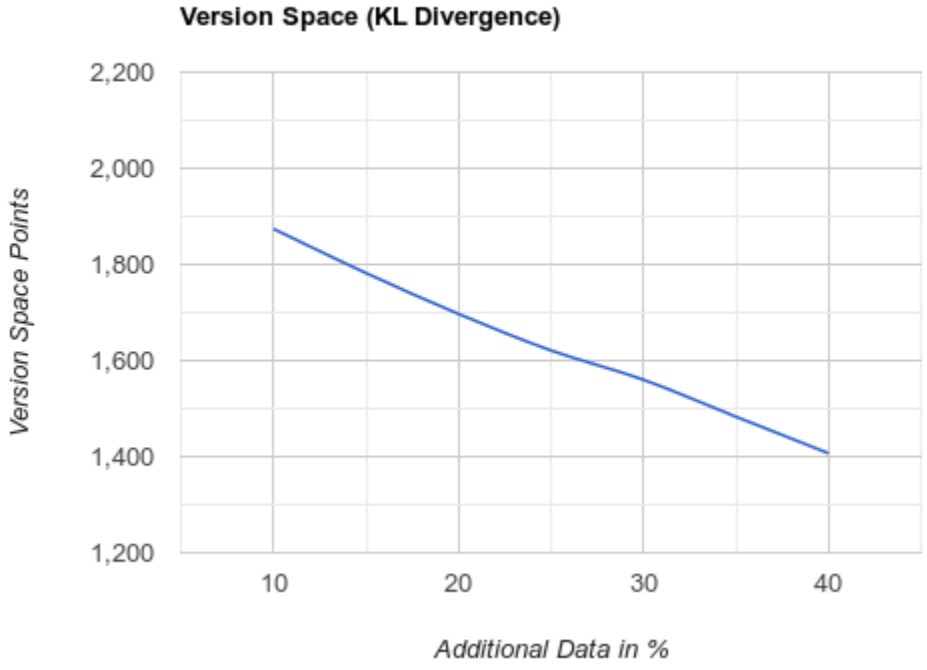



Smallest Margin Sampling
70
60
50
40
Accuracy in %
10
20
30
40
Additional Data in %


Largest Margin Sampling
70
60
50
40
30
Accuracy in %
10
20
30
40
Additional Data in %


Entropy Sampling
80
70
60
50
40
Version Space Points
10
20
30
40
Additional Data in %


Version Space (Vote Entropy)
2,200
2,000
1,800
1,600
1,400
1,200
Version Space Points
10
20
30
40
Additional Data in %


Version Space (KL Divergence)
2,200
2,000
1,800
1,600
1,400
1,200
Version Space Points
10
20
30
40
Additional Data in %

Comparisons

Least Confidence final accuracy = 62.73%
Smallest Margin final accuracy = 65.09%
Largest Margin final accuracy = 60.94%
Entropy Sampling final accuracy = 71.55%
Vote Entropy final accuracy = 74.32%
KL divergence final accuracy = 71.64%

The above accuracy is based on the fact that is has been trained in total of 50% of data in each case by successively

Accuracy when random choosing 50% of points = 34.17%

K Nearest Neighbours

In active learning, the goal is to train a model with a limited amount of labeled data. This is because labeling data can be time-consuming and expensive, so it is often useful to be able to train a model with only a small amount of labeled data. In the case of KNN, active learning can be used to identify the most useful data to label, which can then be used to train the model.

To use active learning with KNN, the algorithm first selects a small number of data points that it is unsure about (these are called "unlabeled" data points). It then calculates the distances between these points and all other data points in the dataset. The points that are closest to the unlabeled points are then labeled, and this labeled data is used to train the model. This process can be repeated until the model has been trained to a satisfactory level of accuracy.

Overall, active learning can be a useful technique for improving the performance of a KNN model when only a small amount of labeled data is available. By actively selecting the most relevant data to learn from, the algorithm is able to make better use of the limited amount of labeled data and can achieve better performance than it would with a larger amount of randomly selected data.

Python was used to create the KNN model, while C++ was used to determine the accuracy by submitting the results through a CSV file.

With 24000 total points in our sample, 90% of the data (21600) was obtained without regard to labels. A baseline model was trained using the 10% of cases. 40% of the data from this unlabeled pool were collected in order to cluster them into 10 clusters. In each cluster, 20% of the data was selected for Oracle's labelling (in this example, the labels were displayed to the model), and the model then assigned labels to each cluster as a whole. Cluster 1 in our model had 2109 data points, Cluster 2 had 2174, cluster 3 had 2156, cluster 4 with 2137, cluster 5 with 2184, cluster 6 contained 2190, cluster 7 contained 2127 , cluster 8 contained 2238 , cluster 9 contained 2122 , cluster 10 contained 2163. .

The result is that 20% of each cluster equals 422 for cluster 1, 434 for cluster 2, 431 for cluster 3, 427 for cluster 4, 437 for cluster 5, 438 for cluster 6,425 for cluster 7, 448 for cluster 8, 424 for cluster 9, 432 for cluster 10. Thus, in total, we had to label 4318 data points, as opposed to 784 if KNN had not been utilised. As a result, KNN lowered our effort by 17282 data points. We may estimate that KNN saved Rs. 1728200 and 17282 hours by assuming that each label costs Rs. 100 and that each labelling process takes one hour. 11,888 out of the 21600 points that were used for clustering were classified properly. Therefore, the accuracy is 55.03%.

Stream-Based Active Learning

Stream-based active learning is a method of training a machine learning model that involves continuously providing the model with new data points from a stream, and asking it to predict the label for each point. As the model makes these predictions, it is given feedback in the form of the correct label for each data point, which it uses to improve its accuracy.

The key advantage of stream-based active learning is that it allows the model to continuously learn and adapt to new data as it becomes available, without the need to retrain the entire model from scratch. This makes it particularly useful for applications where data is constantly changing, such as in real-time decision making or online learning.

Another advantage of stream-based active learning is that it allows the model to focus on the most informative data points, rather than trying to learn from all of the data points in a dataset. This can make the learning process more efficient and improve the model's performance.

In the above case, rather than calculating the power for all unlabeled data, by first setting a threshold and then calculating the power for the unlabeled data points individually and linearly, you can implement stream-based active learning.It is included in the pool of data if the threshold is exceeded. Do this until you reach 10% of the data, that is, until you have reached the required number of data points or have exhausted the data points to

search. This must be followed for both uncertainty sampling and query by committee. Other steps remain the same.