Homework 3

Thomas Zhang (tczhang)

March 28th 2024

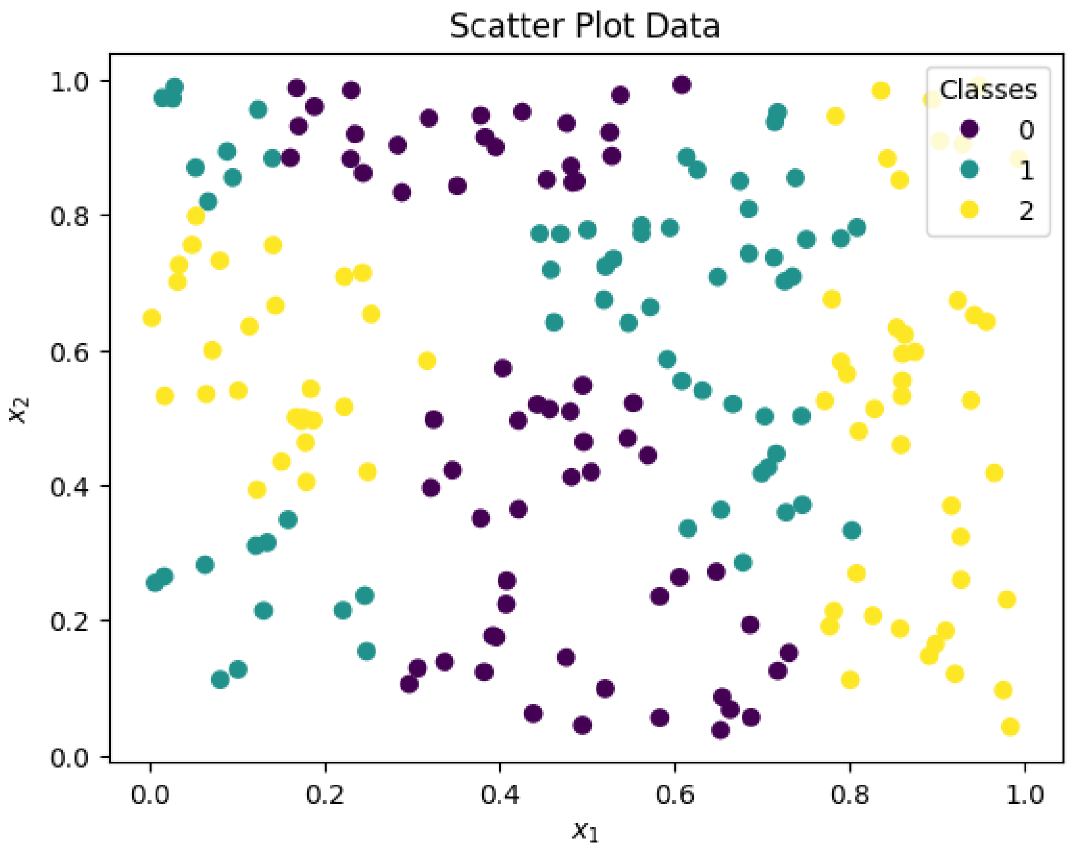

Figure 1: Scatter plot of the dataset for visualization

1.

(a,b,c)

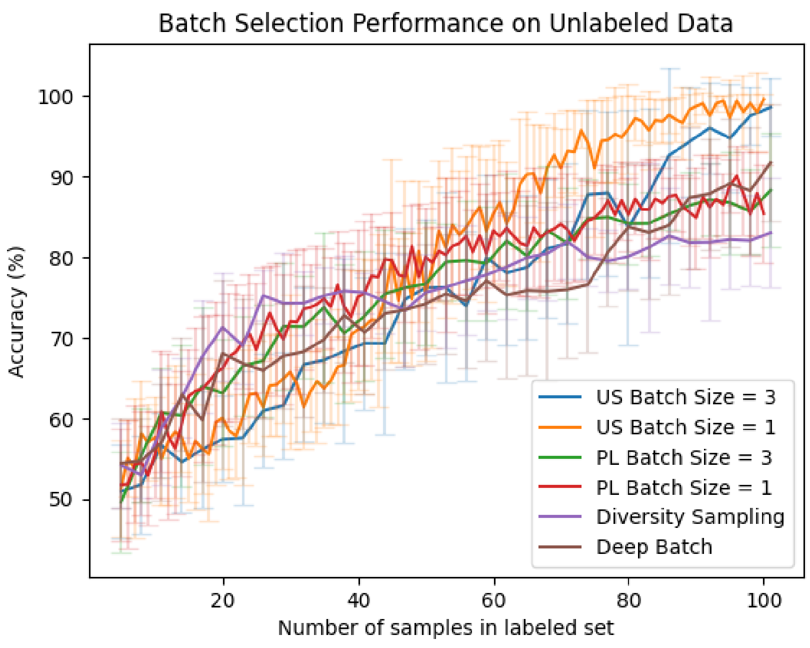

Figure 2: The unobserved accuracy. US: Uncertainty sampling. PL: Passive Learning. Batch sizes for diversity sampling and deep batch active learning was 3.

(d) The data set used is a classification dataset. It has two features (x1 and x2) that are floats. There are three classes (0, 1, 2). This is the same dataset that is used for exercise 2.

The base learner used for all the active learning methods is a multi-layer perceptron (MLP) with three hidden layers with dropout (p = 0.05) between each of the hidden layers. The number of neurons in the hidden layers from input to output is as follows: 16, 64, 16. The output of the multi-layer perceptron is a length 3 vector corresponding to the models prediction on what class the input belongs to.

To perform diversity sampling the unlabeled data was clustered using SKlearn's KMeans clustering algorithm with 10 random initializations. The number of clusters used was equivalent to the size of the batches to be added in each active learning iteration. So for this exercise, the number of clusters was 3. Each iteration one random unlabeled sample is chosen from each of the clusters to move from the unlabeled set to the labeled set. The probability of choosing a sample from each cluster is inversely proportional to the distance of that sample to the center of its cluster.

Deep batch is the Monte Carlo dropout active learning method described in Bailey et. al. (eLife 2024). In this implementation, the Monte carlo dropout method performs passes unlabeled data through the neural network (dropout p= 0.05) 50 times. This is used to approximate y for each sample according to the formula:

$$\hat{y} = \frac{1}{S}\sum_{t=1}^{S} f_{\theta}(x, \mathbf{b}_s)$$

The $\hat{y}$ values are used for calculating the score $\log(\det(\text{cov}(\beta)))$, where $\text{cov}(\beta)$ is the epistemic covariance matrix. Formally, the objective the algorithm solves for to determine the next batch of samples to add to the labeled set is:

$$\underset{\beta\subset U, |\beta|=N}{argmax}\ \log(\det(\text{cov}(\beta)))$$

where U is the unlabeled set, β is the batch of unlabeled samples and N is the size of the batch. This is reflective of finding the batch, β, that has the highest independent uncertainty. Looking for a batch with highest independent uncertainty prevents choosing samples that are highly correlated while still choosing samples with relatively high uncertainty.

To solve this, the algorithm creates a set of batches, each of the appropriate size. Then the algorithm greedily updates each potential batch by replacing one sample in the batch with one sample from the unlabeled set. After the entire set of batches has converged, the samples in the batch with the lowest score is moved from the unlabeled set to the labeled set.

The results matched my expectations for the most part. The uncertainty sampling starts off worse than the other methods. This is expected because the uncertainty sampling method chooses the most difficult samples for the model. Hence at the start of the simulation the accuracy of the model drops as it receives samples that contradicts it's prior knowledge. Then as the simulation continues, the uncertainty sampling methods keep adding samples that are difficult. As a result only samples that are "easy" are left in the test pool, resulting in the unlabeled accuracy of the uncertainty sampling methods to be the highest.

Diversity sampling performs better than uncertainty sampling at first, however at the end of simulation, diversity sampling ends up performing worse than passive learning. This is expected because at the start, diversity sampling chooses samples that are different in the feature space. Hence it gains a lot of information at the start of the simulation. However, the clusters are formed in the feature space and feature space clustering doesn't necessarily reflect the labels of the samples. This can be seen in the visualization of the data in figure 1. So once the model learns the information it can from the clusters it stops improving as much. The data cannot be completely separated by label with just 3 clusters. Thus the the best way for the model to continue learning would be to choose points within a cluster that have differing labels, but this is not what diversity sampling does.

The deep batch active learning method (Bailey et. al. eLife 2024.) performed about as expected. It ends up better than the passive learning strategies but worse than the uncertainty sampling methods. This is expected because it tries to balance the diversity of the samples with the amount of uncertainty from the samples. Thus it performs better than the uncertainty sampling method at the start. However as more samples are added, balancing the diversity of the samples and the uncertainty of the samples ends up causing the model to improve less than uncertainty sampling.

The one thing that is somewhat surprising is that the batch size did not significantly impact the performance of uncertainty sampling. The expected result is that uncertainty sampling with a batch size of 3 performs worse than uncertainty sampling with a batch size of 1 because the smaller batch size allows for more fine grained control of the samples being added to the labeled data set. However this trend only appears to happen for a little bit during the middle of the simulation. At the beginning and the end of the simulation uncertainty sampling of batch size 1 and 3 have very similar performance.

2.

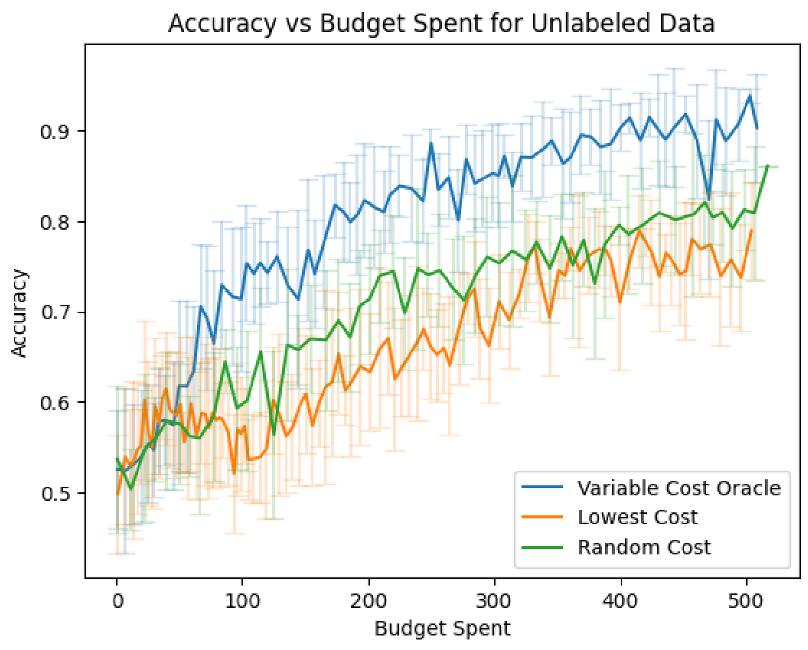

Figure 3: The accuracy of the classifier on the unobserved data. Active learning with costs was performed with batch size = 1. (Each iteration only one sample from the unlabeled set is moved to the labeled set).

The base learner and dataset used for this exercise is the same as in question 1.

The variable cost oracle method was implemented by modifying uncertainty with the costs. In particular the uncertainty and cost vectors for the unlabeled data was normalized to the range $[0, 1]$. Then the a utility score, $U(x, k)$, was calculated for each unlabeled sample using the formula:

$$U(x, k) = V(x) - C_k(x)$$

where $V(x)$ is the uncertainty of sample x and $C_k(x)$ is the cost of using oracle k to label sample x. Then the unlabeled sample with the maximum utility score is chosen to be labeled.

The lowest cost and random cost learner were implemented as their name implies. For the lowest cost learner, the unlabeled sample with the lowest cost was chosen to be added to the labeled set. For the random cost learner, a random unlabeled sample was chosen to be added to the labeled set.

The results were as expected. The variable cost learning method performed the best compared to the random cost and lowest cost methods. This makes sense because the variable cost method takes into account both the uncertainty of the samples and the cost of the samples, thus in general it should choose samples that have high uncertainty for relatively low cost. Hence its cost to information gained ratio should be relatively high compared to the random cost and lowest cost method.

The lowest cost method performed slightly worse than the random cost method. This is somewhat expected because only choosing the lowest cost samples may may only give samples that provide little information to the model, where as with the random cost method there is a chance of choosing samples that contain a good amount of information for relatively low cost. However this is not guaranteed because the lowest cost samples could have a correspondence to higher uncertainty samples. If this were the case then the lowest cost method would have performed better than the random cost method.