Selected topics in AI-2

Assignment 1

Name	ID
Marwan Mohamed abdelmonem	20190513
Youssef Hesham Mohamed	20190648

In this report we have constructed a classification model using active learning strategies applied on three datasets, two datasets are normally distributed while the third is unbalanced.

First, we'll give a brief regarding the datasets and the strategies used then we advance to the analysis and the information extracted from these scenarios.

Dataset brief

1. Iris dataset: It consists of 150 samples of iris flowers, each containing measurements of the sepal length, sepal width, petal length, and petal width. The goal is to classify each sample into one of three species: setosa, versicolor, or virginica, based on the four measurements.

2. Mushroom datasets: This dataset includes descriptions of hypothetical 8124 samples corresponding to 23 species of gilled mushrooms in the Agaricus and Lepiota Family Mushroom drawn from The Audubon Society Field Guide to North American Mushrooms (1981). Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended. This latter class was combined with the poisonous one. The Guide clearly states that there is no simple rule for determining the edibility of a mushroom; no rule like "leaflets three, let it be'' for Poisonous Oak and Ivy.

3. Stroke Prediction Dataset: This dataset is used to predict whether a patient is likely to get stroke based on the input parameters like gender, age, various diseases, and smoking status. Each row in the data provides relevant information about the patient. Contains 5110 observations with 12 attributes.

   Attribute Information
   1) id: unique identifier
   2) gender: "Male", "Female" or "Other"
   3) age: age of the patient
   4) hypertension: 0 if the patient doesn't have hypertension, 1 if the patient has hypertension
   5) heart_disease: 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease
   6) ever_married: "No" or "Yes"
   7) work_type: "children", "Govt_jov", "Never_worked", "Private" or "Self-employed"
   8) Residence_type: "Rural" or "Urban"
   9) avg_glucose_level: average glucose level in blood
   10) bmi: body mass index
   11) smoking_status: "formerly smoked", "never smoked", "smokes" or "Unknown"*
   12) stroke: 1 if the patient had a stroke or 0 if not

   Next, we discuss the four query strategies that were used.

Query strategies

1. Uncertainty Sampling: This strategy selects the instances where the model is most uncertain about the predicted class label. The query strategy queries the instances for which the classifier is least confident. In the code, it is implemented by selecting the samples with the least confidence score as measured by the maximum probability of the predicted class. This strategy helps to maximize the model's accuracy by focusing on the most informative samples.

2. Entropy Sampling: This strategy measures the uncertainty of the predicted probability distribution of the classes. In other words, the query strategy chooses samples for which the model is most uncertain about the predicted probability distribution of the classes. In the code, it is implemented by selecting the samples with the highest entropy as measured by the predicted probability distribution. This strategy is useful when the model has low confidence or exhibits poor performance, but the samples it identifies are diverse and can help to better understand the data.

3. Random Sampling: This strategy selects the samples randomly from the unlabeled dataset, without any bias towards the informative samples. In the code, it is implemented by randomly selecting an instance from the pool of unlabeled samples. This strategy is useful when the dataset is uniformly distributed and there is no specific structure or pattern in the data.

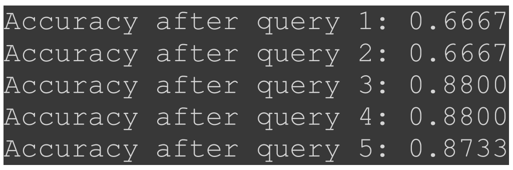
4. Margin Sampling: This strategy selects the samples based on the difference in the probability of the predicted classes. The query strategy chooses samples for which the difference in the probability of the predicted classes is the smallest. In the code, it is implemented by selecting the samples with the smallest margin as measured by the predicted probability difference. This strategy is useful when the model has high accuracy and exhibits a high degree of confidence, but the samples it identifies are biased towards the most informative samples.

Now onto the analysis of the experiment,

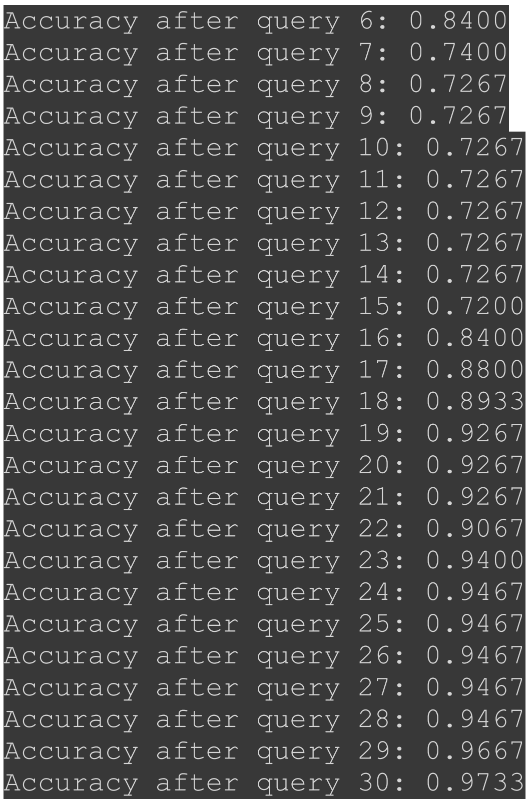
Iris Analysis

We picked three random samples from dataset as initialized dataset (trained with 4,5 and few more and found that 3 samples are the threshold for achieving low but improvable accuracy) then conducted a pool-based sampling approach for training and KNN model for prediction. The model achieved 0.333 without active learning after applying it here are the results.

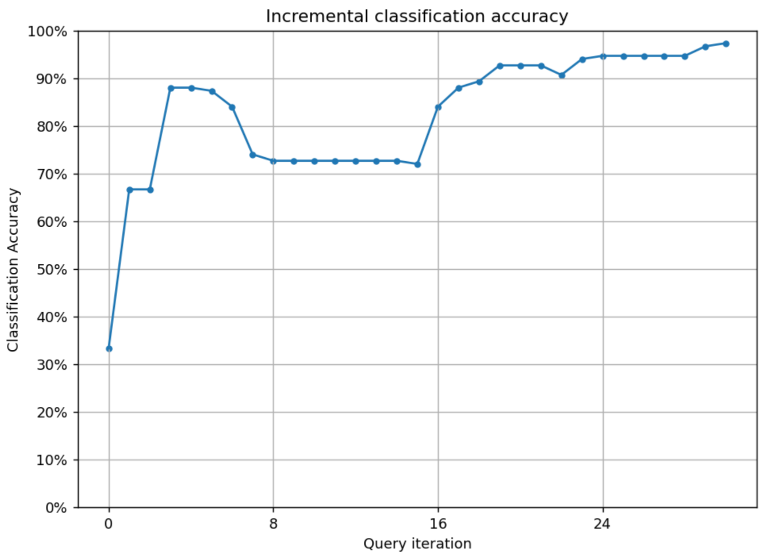
1. Uncertainty sampling



```
Accuracy after query 1: 0.6667
Accuracy after query 2: 0.6667
Accuracy after query 3: 0.8800
Accuracy after query 4: 0.8800
Accuracy after query 5: 0.8733
```

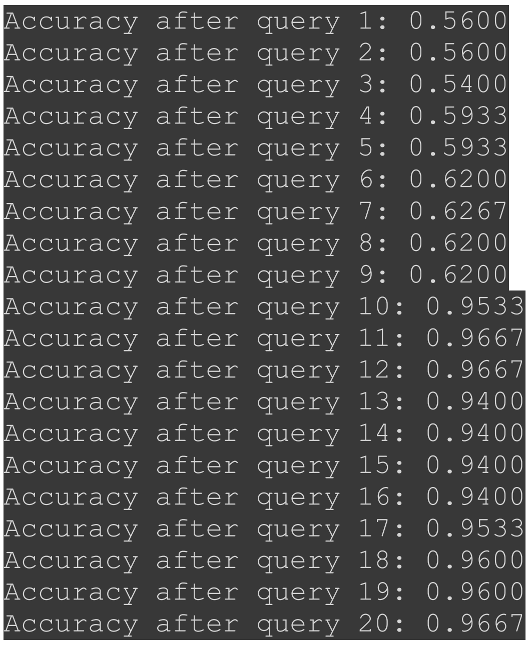


```
Accuracy after query 6: 0.8400
Accuracy after query 7: 0.7400
Accuracy after query 8: 0.7267
Accuracy after query 9: 0.7267
Accuracy after query 10: 0.7267
Accuracy after query 11: 0.7267
Accuracy after query 12: 0.7267
Accuracy after query 13: 0.7267
Accuracy after query 14: 0.7267
Accuracy after query 15: 0.7200
Accuracy after query 16: 0.8400
Accuracy after query 17: 0.8800
Accuracy after query 18: 0.8933
Accuracy after query 19: 0.9267
Accuracy after query 20: 0.9267
Accuracy after query 21: 0.9267
Accuracy after query 22: 0.9067
Accuracy after query 23: 0.9400
Accuracy after query 24: 0.9467
Accuracy after query 25: 0.9467
Accuracy after query 26: 0.9467
Accuracy after query 27: 0.9467
Accuracy after query 28: 0.9467
Accuracy after query 29: 0.9667
Accuracy after query 30: 0.9733
```

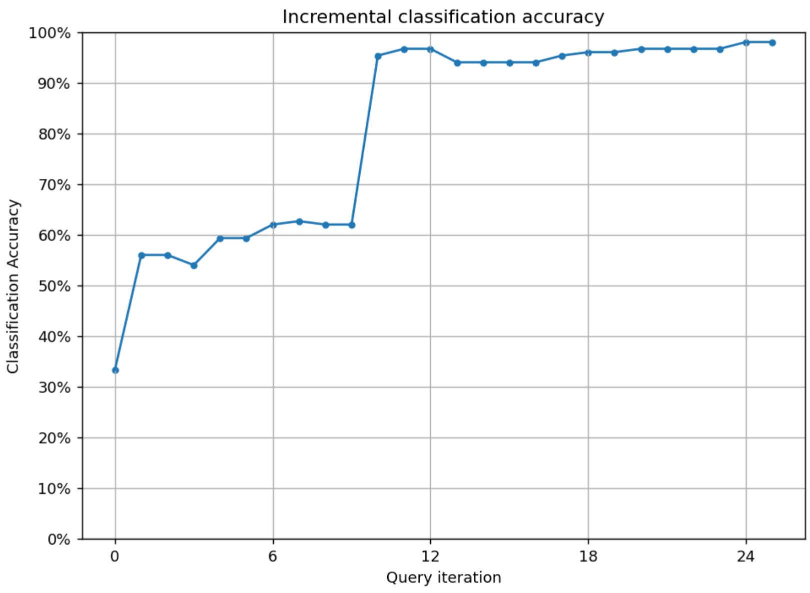


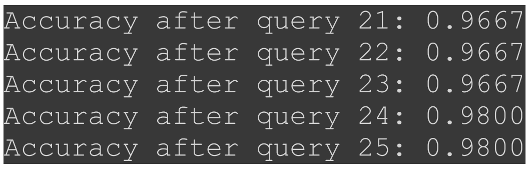
2. random sampling

we used 25 queries instead of 30 as the first because it worsens after this number and this is the best strategy in terms of final acc



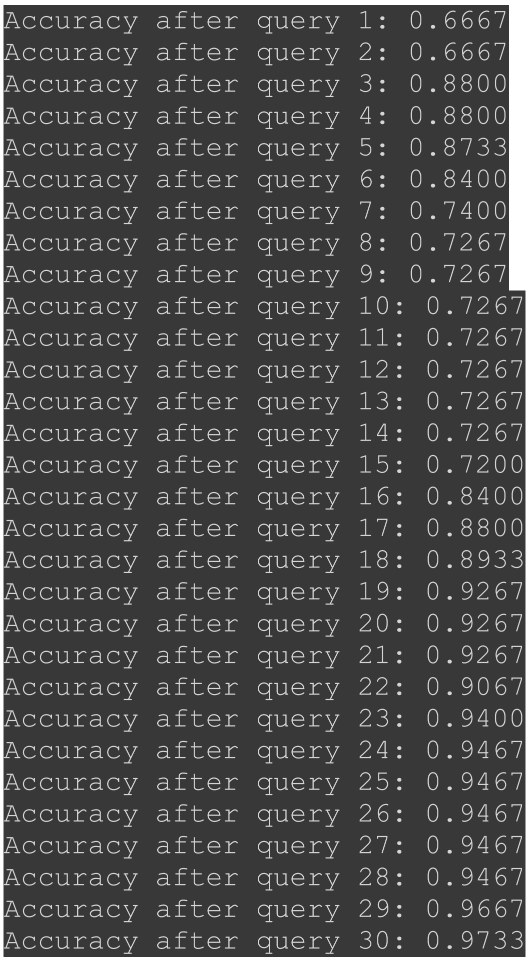
```
Accuracy after query 1: 0.5600
Accuracy after query 2: 0.5600
Accuracy after query 3: 0.5400
Accuracy after query 4: 0.5933
Accuracy after query 5: 0.5933
Accuracy after query 6: 0.6200
Accuracy after query 7: 0.6267
Accuracy after query 8: 0.6200
Accuracy after query 9: 0.6200
Accuracy after query 10: 0.9533
Accuracy after query 11: 0.9667
Accuracy after query 12: 0.9667
Accuracy after query 13: 0.9400
Accuracy after query 14: 0.9400
Accuracy after query 15: 0.9400
Accuracy after query 16: 0.9400
Accuracy after query 17: 0.9533
Accuracy after query 18: 0.9600
Accuracy after query 19: 0.9600
Accuracy after query 20: 0.9667
```



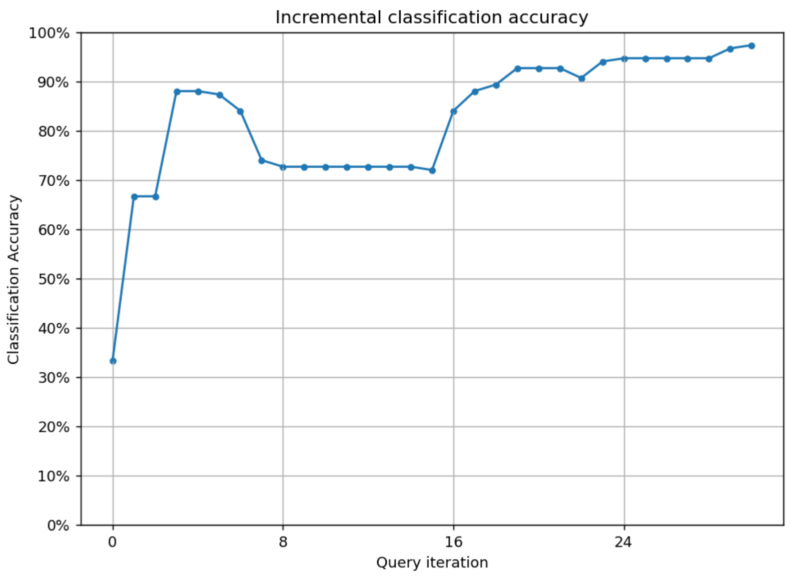


```
Accuracy after query 21: 0.9667
Accuracy after query 22: 0.9667
Accuracy after query 23: 0.9667
Accuracy after query 24: 0.9800
Accuracy after query 25: 0.9800
```

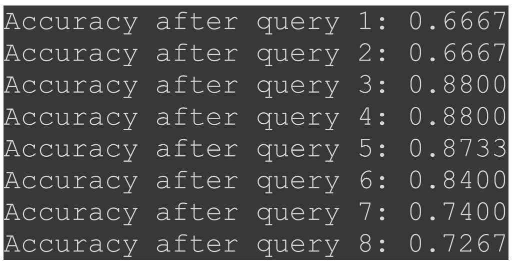
3. entropy sampling



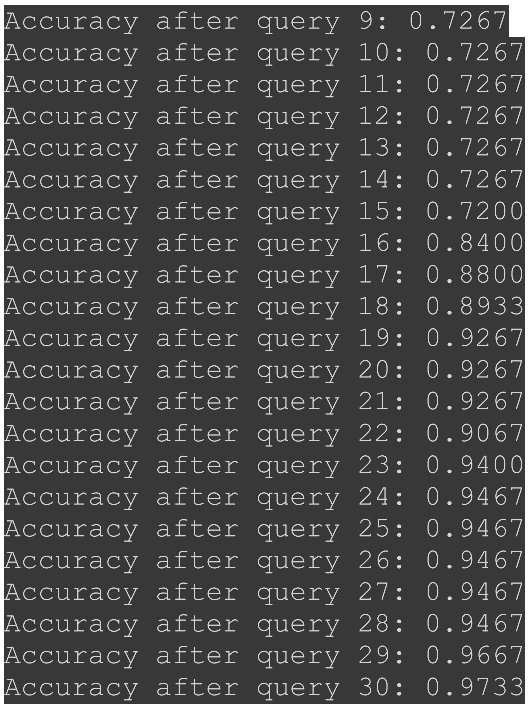
```
Accuracy after query 1: 0.6667
Accuracy after query 2: 0.6667
Accuracy after query 3: 0.8800
Accuracy after query 4: 0.8800
Accuracy after query 5: 0.8733
Accuracy after query 6: 0.8400
Accuracy after query 7: 0.7400
Accuracy after query 8: 0.7267
Accuracy after query 9: 0.7267
Accuracy after query 10: 0.7267
Accuracy after query 11: 0.7267
Accuracy after query 12: 0.7267
Accuracy after query 13: 0.7267
Accuracy after query 14: 0.7267
Accuracy after query 15: 0.7200
Accuracy after query 16: 0.8400
Accuracy after query 17: 0.8800
Accuracy after query 18: 0.8933
Accuracy after query 19: 0.9267
Accuracy after query 20: 0.9267
Accuracy after query 21: 0.9267
Accuracy after query 22: 0.9067
Accuracy after query 23: 0.9400
Accuracy after query 24: 0.9467
Accuracy after query 25: 0.9467
Accuracy after query 26: 0.9467
Accuracy after query 27: 0.9467
Accuracy after query 28: 0.9467
Accuracy after query 29: 0.9667
Accuracy after query 30: 0.9733
```



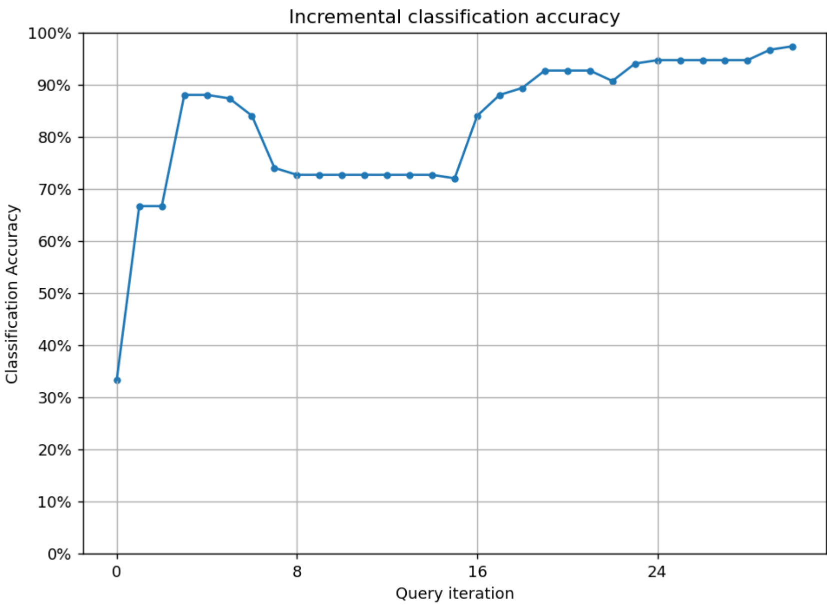
4. margin sampling



```
Accuracy after query 1: 0.6667
Accuracy after query 2: 0.6667
Accuracy after query 3: 0.8800
Accuracy after query 4: 0.8800
Accuracy after query 5: 0.8733
Accuracy after query 6: 0.8400
Accuracy after query 7: 0.7400
Accuracy after query 8: 0.7267
```



```
Accuracy after query 9: 0.7267
Accuracy after query 10: 0.7267
Accuracy after query 11: 0.7267
Accuracy after query 12: 0.7267
Accuracy after query 13: 0.7267
Accuracy after query 14: 0.7267
Accuracy after query 15: 0.7200
Accuracy after query 16: 0.8400
Accuracy after query 17: 0.8800
Accuracy after query 18: 0.8933
Accuracy after query 19: 0.9267
Accuracy after query 20: 0.9267
Accuracy after query 21: 0.9267
Accuracy after query 22: 0.9067
Accuracy after query 23: 0.9400
Accuracy after query 24: 0.9467
Accuracy after query 25: 0.9467
Accuracy after query 26: 0.9467
Accuracy after query 27: 0.9467
Accuracy after query 28: 0.9467
Accuracy after query 29: 0.9667
Accuracy after query 30: 0.9733
```

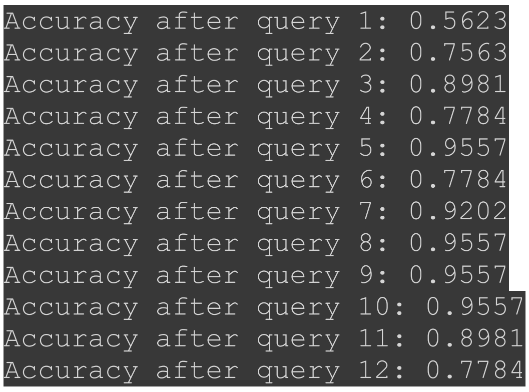


From these results and based on the initial setup we had to ensure that all strategies are applied within the same environment we found out that random sampling achieved the best accuracies while the other 3 strategies resulted in almost identical results and this conclude that small datasets will not have high impact in the aspect of selecting which strategy to compute but overall active learning will reduce the cost of samples and the computation power with higher accuracy than the normal model

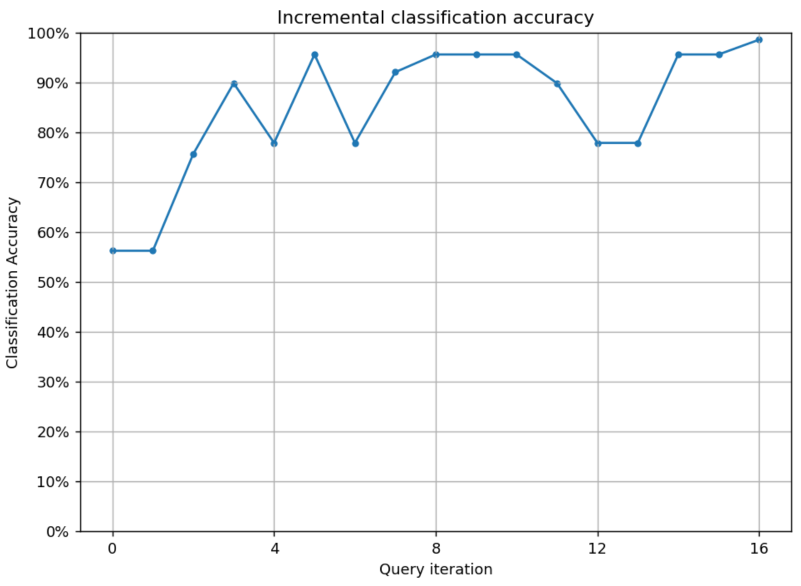
Mushrooms dataset

We picked 5 random samples from dataset as initialized dataset (trained with few more and found that 5 samples are the threshold for achieving low but improvable accuracy) then conducted a pool-based sampling approach for training and Decision Tree Classifier for prediction. The model achieved 0.562 without active learning after applying it here are the results.

1. Uncertainty sampling
   We used 16 queries as more will result in a plateau.



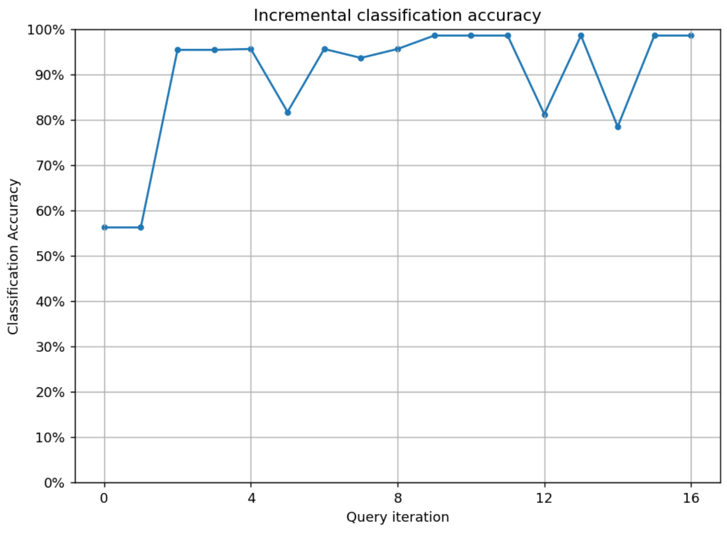
```
Accuracy after query 1: 0.5623
Accuracy after query 2: 0.7563
Accuracy after query 3: 0.8981
Accuracy after query 4: 0.7784
Accuracy after query 5: 0.9557
Accuracy after query 6: 0.7784
Accuracy after query 7: 0.9202
Accuracy after query 8: 0.9557
Accuracy after query 9: 0.9557
Accuracy after query 10: 0.9557
Accuracy after query 11: 0.8981
Accuracy after query 12: 0.7784
```



```
Accuracy after query 13: 0.7784
Accuracy after query 14: 0.9557
Accuracy after query 15: 0.9557
Accuracy after query 16: 0.9852
```

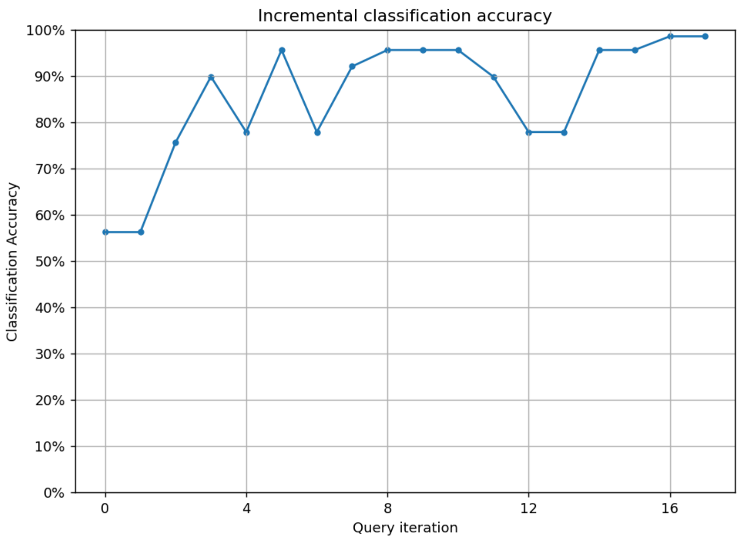
2. random sampling

```
Accuracy after query 1: 0.5623
Accuracy after query 2: 0.9537
Accuracy after query 3: 0.9537
Accuracy after query 4: 0.9557
Accuracy after query 5: 0.8168
Accuracy after query 6: 0.9557
Accuracy after query 7: 0.9360
Accuracy after query 8: 0.9557
Accuracy after query 9: 0.9852
Accuracy after query 10: 0.9852
Accuracy after query 11: 0.9852
Accuracy after query 12: 0.8119
Accuracy after query 13: 0.9852
Accuracy after query 14: 0.7843
Accuracy after query 15: 0.9852
Accuracy after query 16: 0.9852
```

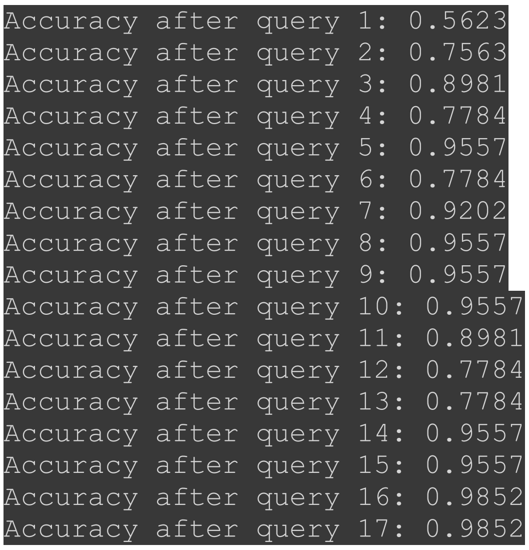


3. entropy sampling

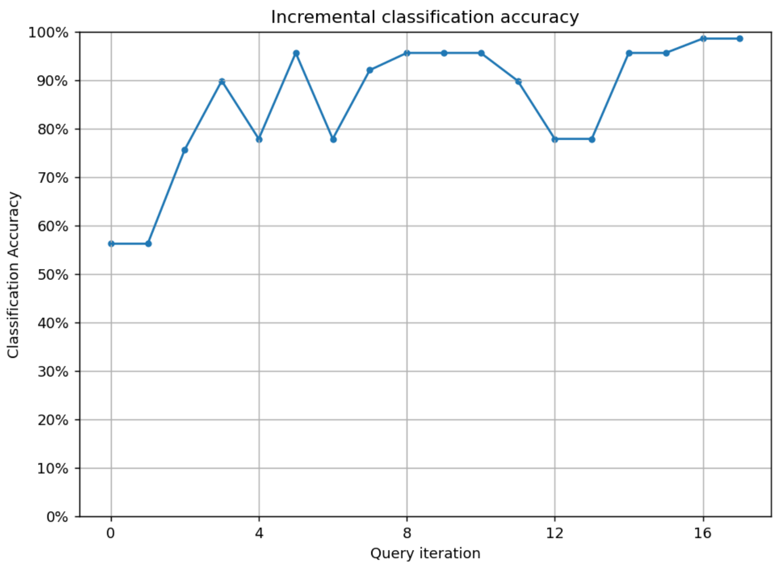
```
Accuracy after query 1: 0.5623
Accuracy after query 2: 0.7563
Accuracy after query 3: 0.8981
Accuracy after query 4: 0.7784
Accuracy after query 5: 0.9557
Accuracy after query 6: 0.7784
Accuracy after query 7: 0.9202
Accuracy after query 8: 0.9557
Accuracy after query 9: 0.9557
Accuracy after query 10: 0.9557
Accuracy after query 11: 0.8981
Accuracy after query 12: 0.7784
Accuracy after query 13: 0.7784
Accuracy after query 14: 0.9557
Accuracy after query 15: 0.9557
Accuracy after query 16: 0.9852
Accuracy after query 17: 0.9852
```



4. margin sampling

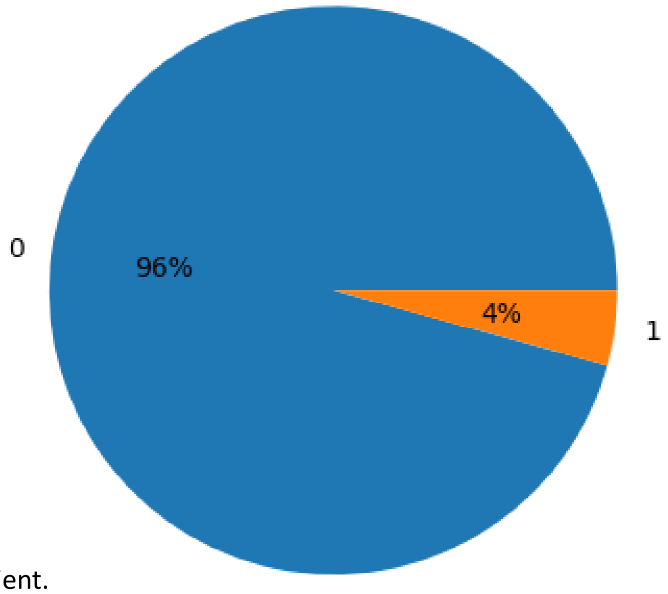


```
Accuracy after query 1: 0.5623
Accuracy after query 2: 0.7563
Accuracy after query 3: 0.8981
Accuracy after query 4: 0.7784
Accuracy after query 5: 0.9557
Accuracy after query 6: 0.7784
Accuracy after query 7: 0.9202
Accuracy after query 8: 0.9557
Accuracy after query 9: 0.9557
Accuracy after query 10: 0.9557
Accuracy after query 11: 0.8981
Accuracy after query 12: 0.7784
Accuracy after query 13: 0.7784
Accuracy after query 14: 0.9557
Accuracy after query 15: 0.9557
Accuracy after query 16: 0.9852
Accuracy after query 17: 0.9852
```



Stroke Prediction dataset

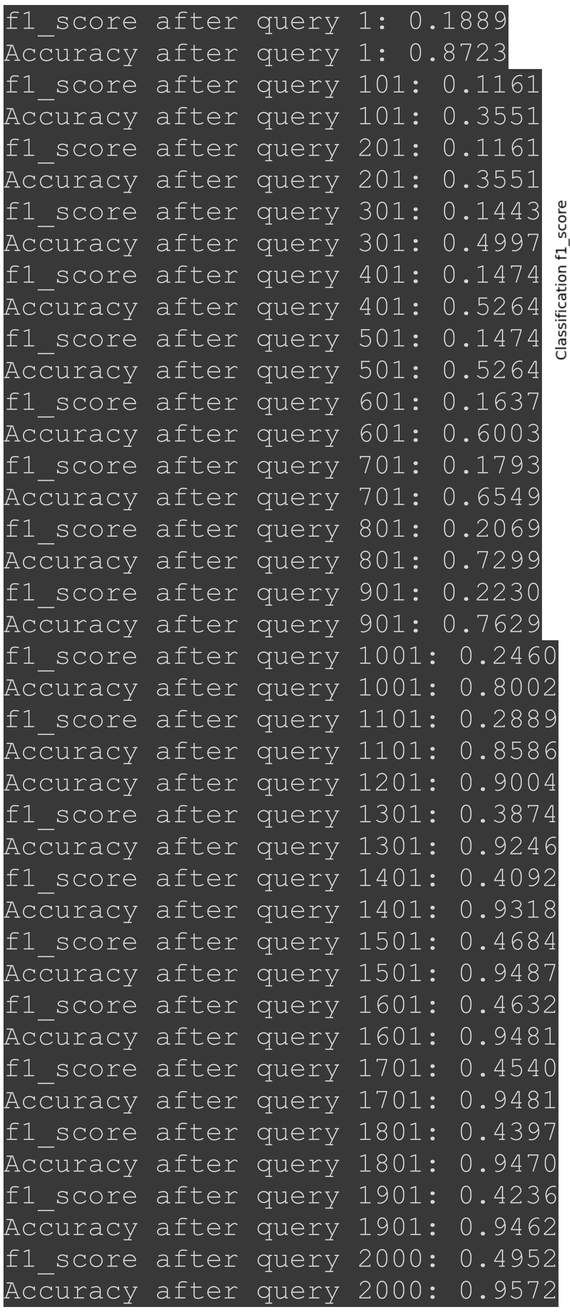
The distribution of data:



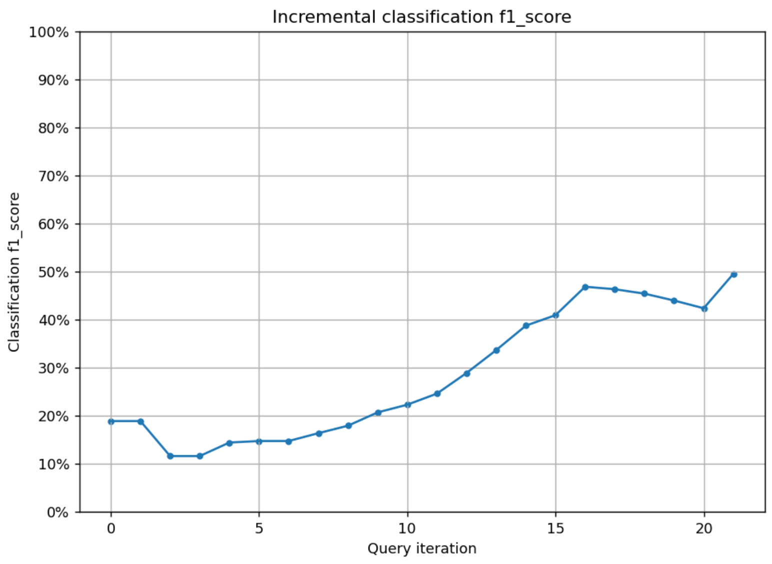
Where '0' class is non-patient while '1' class is patient.

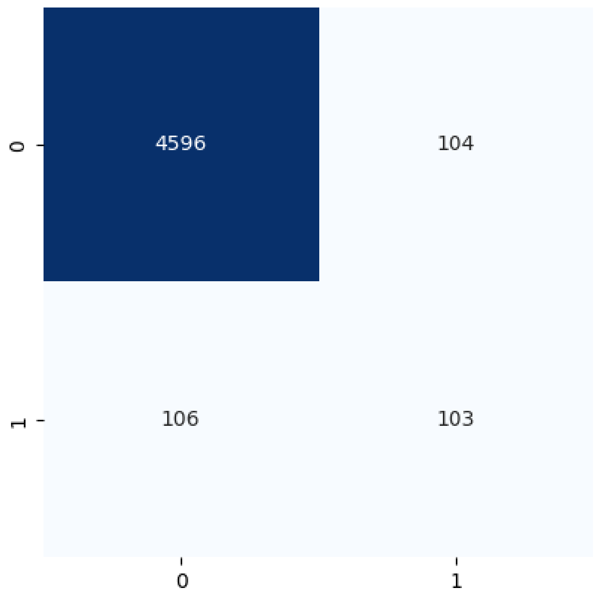
We picked 10 random samples from dataset as initialized dataset (trained with few more and found that 10 samples are the threshold for achieving low but improvable accuracy) then conducted a pool-based sampling approach for training and XGB Classifier for prediction. The model achieved f1 score with 0.19 without active learning after applying it here are the results.

1. Uncertainty sampling:

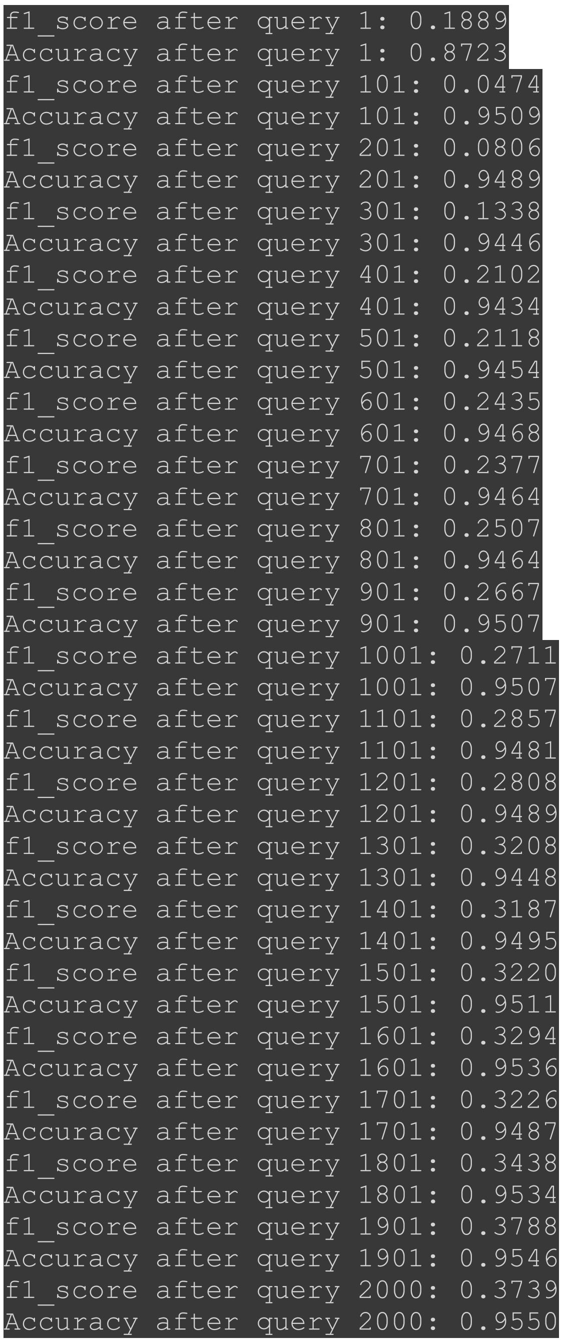


```
f1_score after query 1: 0.1889
Accuracy after query 1: 0.8723
f1_score after query 101: 0.1161
Accuracy after query 101: 0.3551
f1_score after query 201: 0.1161
Accuracy after query 201: 0.3551
f1_score after query 301: 0.1443
Accuracy after query 301: 0.4997
f1_score after query 401: 0.1474
Accuracy after query 401: 0.5264
f1_score after query 501: 0.1474
Accuracy after query 501: 0.5264
f1_score after query 601: 0.1637
Accuracy after query 601: 0.6003
f1_score after query 701: 0.1793
Accuracy after query 701: 0.6549
f1_score after query 801: 0.2069
Accuracy after query 801: 0.7299
f1_score after query 901: 0.2230
Accuracy after query 901: 0.7629
f1_score after query 1001: 0.2460
Accuracy after query 1001: 0.8002
f1_score after query 1101: 0.2889
Accuracy after query 1101: 0.8586
Accuracy after query 1201: 0.9004
f1_score after query 1301: 0.3874
Accuracy after query 1301: 0.9246
f1_score after query 1401: 0.4092
Accuracy after query 1401: 0.9318
f1_score after query 1501: 0.4684
Accuracy after query 1501: 0.9487
f1_score after query 1601: 0.4632
Accuracy after query 1601: 0.9481
f1_score after query 1701: 0.4540
Accuracy after query 1701: 0.9481
f1_score after query 1801: 0.4397
Accuracy after query 1801: 0.9470
f1_score after query 1901: 0.4236
Accuracy after query 1901: 0.9462
f1_score after query 2000: 0.4952
Accuracy after query 2000: 0.9572
```

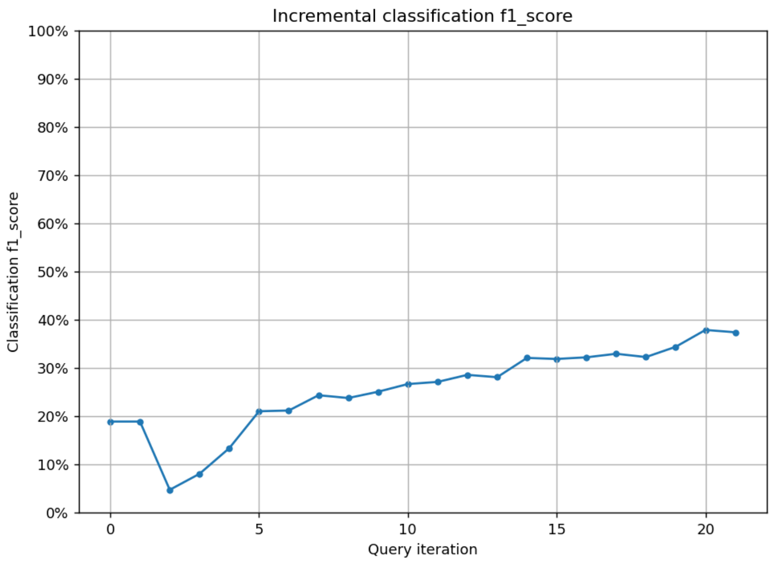


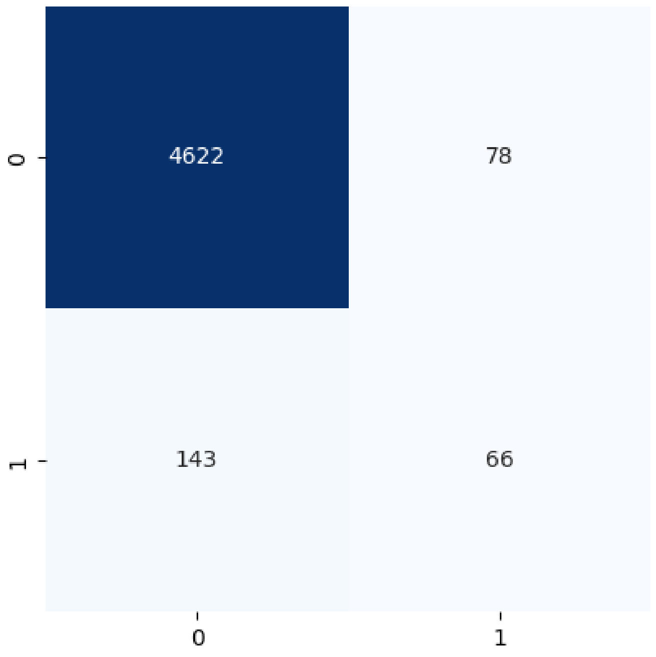


2. random sampling

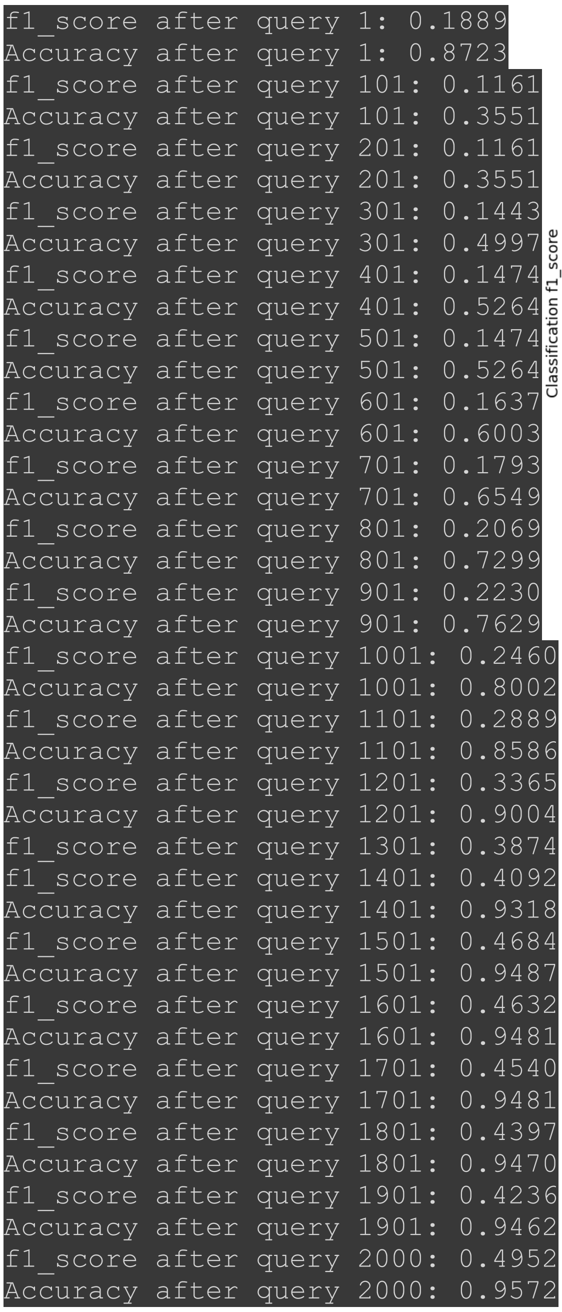


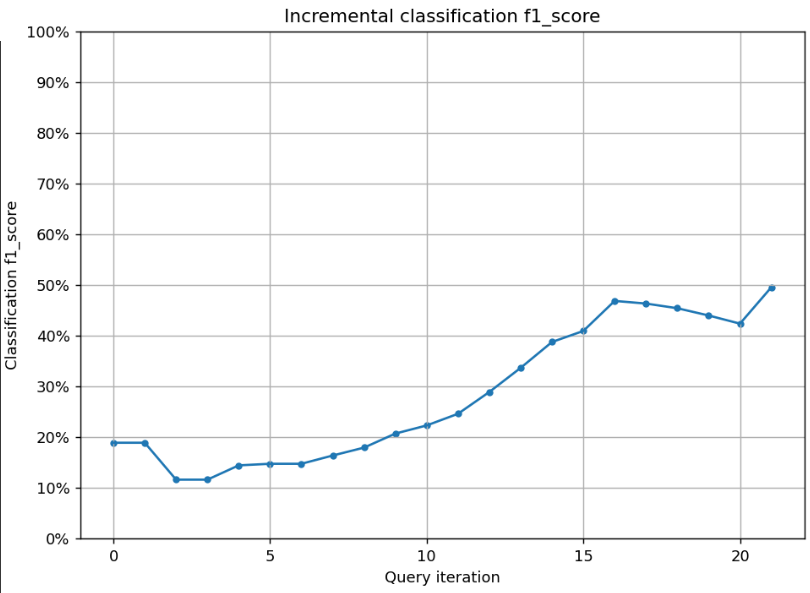
```
f1_score after query 1: 0.1889
Accuracy after query 1: 0.8723
f1_score after query 101: 0.0474
Accuracy after query 101: 0.9509
f1_score after query 201: 0.0806
Accuracy after query 201: 0.9489
f1_score after query 301: 0.1338
Accuracy after query 301: 0.9446
f1_score after query 401: 0.2102
Accuracy after query 401: 0.9434
f1_score after query 501: 0.2118
Accuracy after query 501: 0.9454
f1_score after query 601: 0.2435
Accuracy after query 601: 0.9468
f1_score after query 701: 0.2377
Accuracy after query 701: 0.9464
f1_score after query 801: 0.2507
Accuracy after query 801: 0.9464
f1_score after query 901: 0.2667
Accuracy after query 901: 0.9507
f1_score after query 1001: 0.2711
Accuracy after query 1001: 0.9507
f1_score after query 1101: 0.2857
Accuracy after query 1101: 0.9481
f1_score after query 1201: 0.2808
Accuracy after query 1201: 0.9489
f1_score after query 1301: 0.3208
Accuracy after query 1301: 0.9448
f1_score after query 1401: 0.3187
Accuracy after query 1401: 0.9495
f1_score after query 1501: 0.3220
Accuracy after query 1501: 0.9511
f1_score after query 1601: 0.3294
Accuracy after query 1601: 0.9536
f1_score after query 1701: 0.3226
Accuracy after query 1701: 0.9487
f1_score after query 1801: 0.3438
Accuracy after query 1801: 0.9534
f1_score after query 1901: 0.3788
Accuracy after query 1901: 0.9546
f1_score after query 2000: 0.3739
Accuracy after query 2000: 0.9550
```

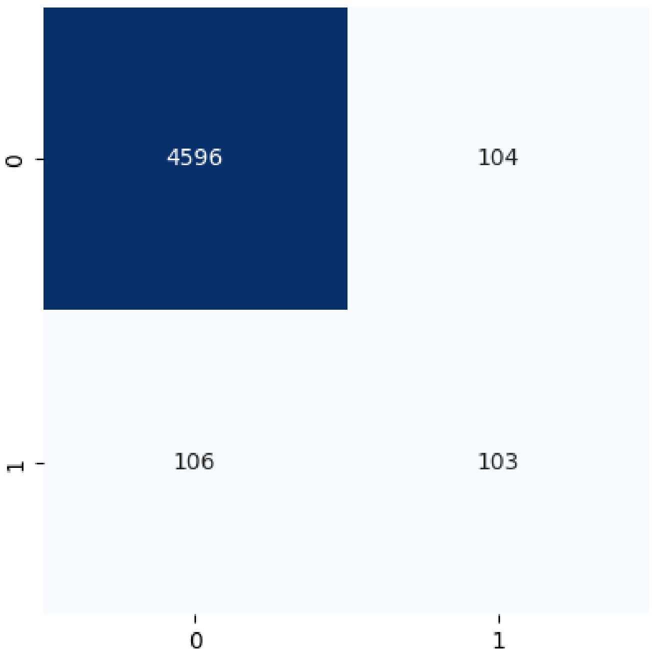




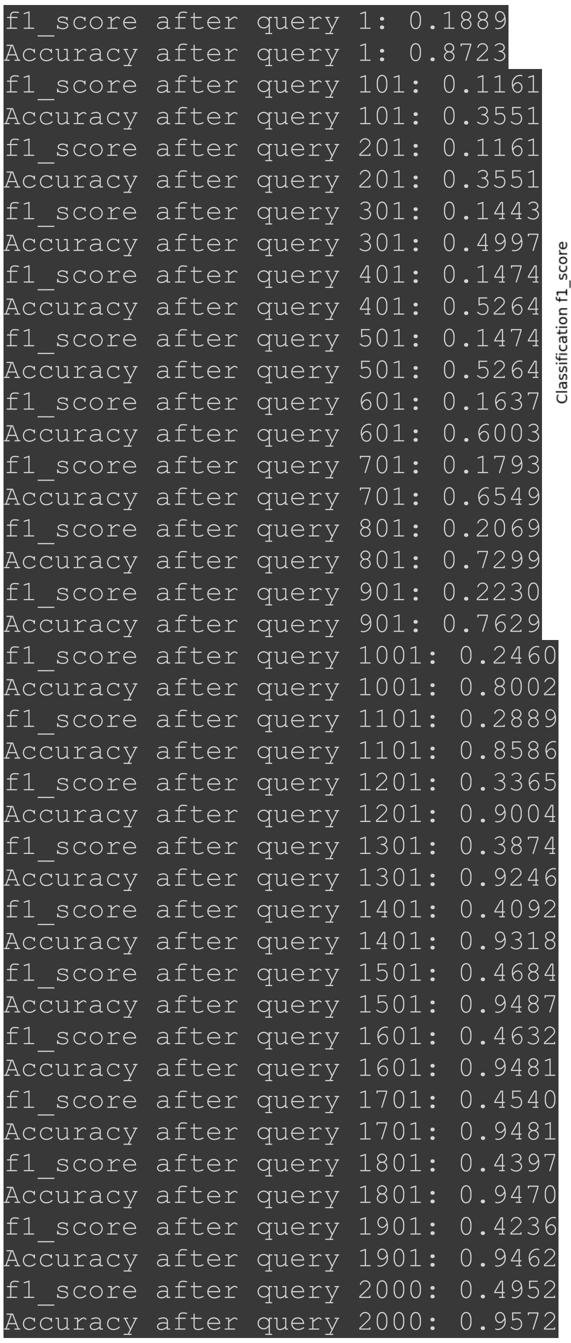
3. entropy sampling


```
f1_score after query 1: 0.1889
Accuracy after query 1: 0.8723
f1_score after query 101: 0.1161
Accuracy after query 101: 0.3551
f1_score after query 201: 0.1161
Accuracy after query 201: 0.3551
f1_score after query 301: 0.1443
Accuracy after query 301: 0.4997
f1_score after query 401: 0.1474
Accuracy after query 401: 0.5264
f1_score after query 501: 0.1474
Accuracy after query 501: 0.5264
f1_score after query 601: 0.1637
Accuracy after query 601: 0.6003
f1_score after query 701: 0.1793
Accuracy after query 701: 0.6549
f1_score after query 801: 0.2069
Accuracy after query 801: 0.7299
f1_score after query 901: 0.2230
Accuracy after query 901: 0.7629
f1_score after query 1001: 0.2460
Accuracy after query 1001: 0.8002
f1_score after query 1101: 0.2889
Accuracy after query 1101: 0.8586
f1_score after query 1201: 0.3365
Accuracy after query 1201: 0.9004
f1_score after query 1301: 0.3874
f1_score after query 1401: 0.4092
Accuracy after query 1401: 0.9318
f1_score after query 1501: 0.4684
Accuracy after query 1501: 0.9487
f1_score after query 1601: 0.4632
Accuracy after query 1601: 0.9481
f1_score after query 1701: 0.4540
Accuracy after query 1701: 0.9481
f1_score after query 1801: 0.4397
Accuracy after query 1801: 0.9470
f1_score after query 1901: 0.4236
Accuracy after query 1901: 0.9462
f1_score after query 2000: 0.4952
Accuracy after query 2000: 0.9572
```

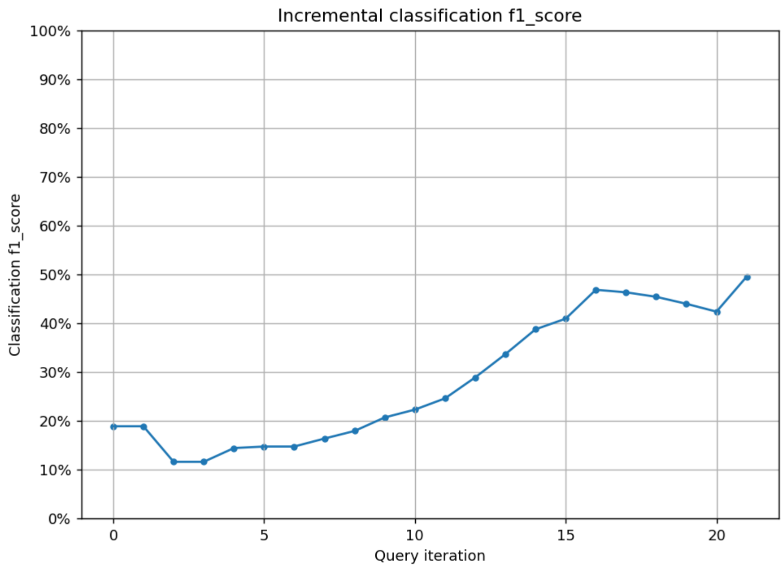


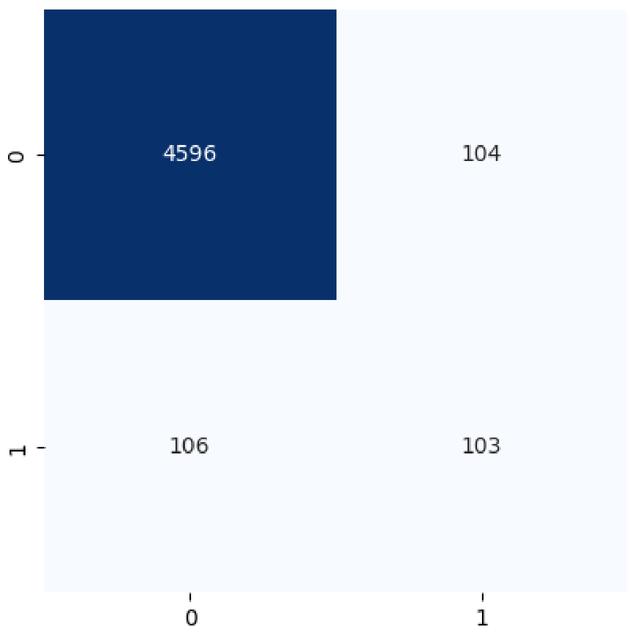


4. margin sampling



```
f1_score after query 1: 0.1889
Accuracy after query 1: 0.8723
f1_score after query 101: 0.1161
Accuracy after query 101: 0.3551
f1_score after query 201: 0.1161
Accuracy after query 201: 0.3551
f1_score after query 301: 0.1443
Accuracy after query 301: 0.4997
f1_score after query 401: 0.1474
Accuracy after query 401: 0.5264
f1_score after query 501: 0.1474
Accuracy after query 501: 0.5264
f1_score after query 601: 0.1637
Accuracy after query 601: 0.6003
f1_score after query 701: 0.1793
Accuracy after query 701: 0.6549
f1_score after query 801: 0.2069
Accuracy after query 801: 0.7299
f1_score after query 901: 0.2230
Accuracy after query 901: 0.7629
f1_score after query 1001: 0.2460
Accuracy after query 1001: 0.8002
f1_score after query 1101: 0.2889
Accuracy after query 1101: 0.8586
f1_score after query 1201: 0.3365
Accuracy after query 1201: 0.9004
f1_score after query 1301: 0.3874
Accuracy after query 1301: 0.9246
f1_score after query 1401: 0.4092
Accuracy after query 1401: 0.9318
f1_score after query 1501: 0.4684
Accuracy after query 1501: 0.9487
f1_score after query 1601: 0.4632
Accuracy after query 1601: 0.9481
f1_score after query 1701: 0.4540
Accuracy after query 1701: 0.9481
f1_score after query 1801: 0.4397
Accuracy after query 1801: 0.9470
f1_score after query 1901: 0.4236
Accuracy after query 1901: 0.9462
f1_score after query 2000: 0.4952
Accuracy after query 2000: 0.9572
```





Conclusion

Each strategy in the uncertainty family doesn’t vary much as we can't for sure say that one strategy is better than other overall however the active learning approaches as a whole is better in the factors of maintaining high score in evaluation metrics (accuracy in most cases)

While reducing cost of computation, annotation and budget as we can run many models with many varieties in environment in parallel.