Class Imbalance in Active Learning

Brent Matthys & Mats Van Belle

December 22, 2023

Abstract

Collecting substantial amounts of annotated data is often one of the more challenging parts of training a machine learning model. Not every domain has large data sets readily accessible. One strategy to address this problem is through active learning. In active learning we start from an initially unlabeled data set. We iterate and label more samples until the model finally achieves a desired result.

In this paper we look at the influence of class imbalance on the efficiency of active learning. Concretely, we research multiple different techniques and how they affect the accuracy of our active learning model given a data set with class imbalance. After researching using synthetic data we look at real-world data sets to verify our findings.

1 Introduction

Say we wanted to create a simple model to classify birds in our local area. This would be an easy enough task since we could simply grab a model from a library and ask our friends and neighbours to take pictures of any birds they see. However, a problem arises with the latter. Our data is not annotated, therefore our model will have a hard time classifying the birds. We could label all the images ourselves, but this is tedious work and it would result in us losing interest in the project. Hiring a bird expert to do it for us would be way too costly.

We are certainly not the first to have run into this issue where we need vast amounts of annotated data. One of the most common strategies to address this problem is active learning. The idea is to start training the model with unlabeled data. We then use the prediction of the model to query the most informative sample from our unlabeled data set. This sample is then labeled by an oracle. Our model now has more information to do a better prediction. This process of training, querying and labeling is then repeated until we reach a desired accuracy. By doing so we reduce the amount of labeled samples needed to a minimum.

When looking at the local bird count of Natuurpunt [12] we notice that the distribution between birds differs. Consider the following comparison.

Bird species	Count
House sparrow	78,426
Red robin	20,821

Table 1: Bird count

Given this, we would probably receive 80% House sparrow images and only 20% Red robin images from our friends and neighbours. This could prove to be an issue for our model. It might always predict the majority class and think it is doing a good job with 80% accuracy.

Imbalanced data comes with its fair share of challenges and issues. Luckily there are various ways of addressing these problems. In this paper we will look into these methods, and how they behave when using active learning. In general, we ask ourselves what the influence is of class imbalance on active learning. What are the things to do and not to do when using active learning with imbalanced data sets.

Previous research done in this field mostly ignores any imbalance in the data sets [3, 6, 17]. Even so, many blog posts and technical papers initially helped us in getting started [6, 11, 14, 17]. Others have shown us many interesting concepts to look into about class imbalance [2, 13]. Our code is shown on our public github repo [10].

The rest of this paper is structured as follows:

- In section 2 we present the methods used throughout this paper.
- In section 3 we showcase our findings and discuss which methods are a valuable investment of your time and which ones are better to avoid.
- We look into some real world data sets in section 4 and conclude our paper in section 5.

2 Methods

In this section, we offer an explanation for the methodology we use throughout the experiment. First we will illustrate how we will implement a basic active learning loop using a library of our choice. After this, we mainly take a look at the query strategies we used as well as different classifiers for the underlying model. We also explain how we create our classification and the different kinds of accuracy we used to grade our performance. We will explain our process to lay a foundation for our results and conclusions that will follow in the next section.

2.1 Active learning

To achieve this good foundation, we prefer a sturdy but flexible library to run our experiments. Our active learning process needs to be consistent and our results easily interpretable. For this purpose we opted for the scikit-activeml [8] library. This library is an easy choice because the library has a very good integration with the other sklearn [15] libraries which are widely used and facilitate some handy testing tools.

Algorithm 1 The active learning cycle

```
1:  procedure GET_ACCURACY
2:      qs ← new query_sampler
3:      X, y ← new classification
4:      clf ← new classifier
5:      y_known ← empty
6:      for _ in 0..N do
7:          y_new ← query from qs
8:          add y_new to y_known
9:          fit clf on X, y_known
10:         score ← score from clf
11:         output score
```

Our main active learning loop is illustrated in algorithm 1. The algorithm itself doesn't need extensive explanation, it shows how a basic model (*classifier*) is gradually trained on more and more known labels (*y_known*) until we reach the amount of desired labels (N). The algorithm perfectly shows how active learning works only a few samples at a time in each loop (*y_new*). In the real world the labeling would be done by humans but here, for testing purposes, we can use a known classification. This way we can more accurately test our results. To further improve the accuracy we will run this algorithm a good number of times and average the results to eliminate any outliers. From the algorithm we acquire a good number of parameters we can experiment with. In the rest of this section, we will discuss different query samplers, different classifiers, how we achieve our classification and a few score functions.

2.2 Query strategies

From our literature study, we already know that different query strategies can offer widely different results [6]. This is because the query strategies can also differ a lot. We have selected the, in our opinion, most commonly used and diverse strategies. We hope this can provide a wide array of results to choose from.

2.2.1 Random- and Uncertainty sampling

In our discussion we will mainly mention two sampling methods: random sampling and uncertainty sampling. This is because they provide a solid baseline.

It is important to note that random sampling is **not** an active learning method. It chooses its samples/features completely randomly. This provides a baseline to compare our active learning methods against. Random sampling is used because it fits perfectly, without any modification, in the active learning loop (algorithm 1).

On the other hand, uncertainty sampling is by far the most well-known active learning query strategy. It chooses its features by asking the classifier which samples it is the least certain about. This means that our model needs to provide an accurate probability score function[1]. This is an easy but strong way to compare different features, it provides a very solid baseline.

2.2.2 Other query strategies

To compare, we used two other query strategies: query by committee and fourD's. Query by committee extends the uncertainty principle by training an ensemble of estimators. It then identifies which instances the estimators agree the least upon. This has the benefit that the model does not need to score the instances itself. It only needs to provide a prediction of the data. It however takes a lot longer to train multiple models at once, but this is mainly overlooked in real-world cases by the time it takes for humans to label the data.

FourD's is a lesser known query strategy we used to correspond with the mixture model classifier (see section 3.4). It considers four criteria in its selection strategy: distance, density, diversity, distribution [16].

[1] predict_proba() method of sklearn

2.2.3 Batch sampling

In real world cases however it is unrealistic for the oracle to request instances to be labeled one at a time. It is often preferred to label a group of samples all at once. This is called batch sampling. There has already been a lot of research done on optimizing batch algorithms [4, 7] and we will discuss results when introducing imbalanced data in section 3.8. Batch sampling is however not the main focus of this paper but we do briefly use some well-known algorithms such as batchBALD [7]. Skactiveml also has a default option to request a bigger batch sizes at the query sampler.

In section 3.9 we will use a new metric called batch fairness for our purposes with imbalanced data.

$$Fairness = \frac{amount\_of\_samples\_from\_class}{amount\_of\_total\_samples} \quad (1)$$

This fairness is explained in formula 1. It shows how fair the batch algorithm is in choosing new instances to be labelled. For a data set with 2 classes we would want the fairness for each class to be as close to 0.50 as possible. For 3 classes it would be 0.33 and so forth. This would show if the batch algorithm is also imbalanced when choosing samples. Our hypothesis is that a more fair batch algorithm will provide better accuracies.

2.3 Classifiers

For the underlying models we tested some well-known classifiers (logistic regression, stochastic gradient descent and multilayer perceptron). We also used a random forest estimator because in the literature we saw, this classifier kept performing exceedingly well. In scikit-activeml [8] a parzen window and a mixture model classifier were also provided for active learning purposes so we decided to test those as well.

It is important to note that we still believe that some models can be more or less suited for your purpose but these should provide a solid starting point when you are using active learning with imbalanced data.

2.4 Data set generation

When testing the accuracies of a model it is practical to generate synthetic data sets ourselves. Luckily, scikit-learn[15] has a flexible method[2] with a lot of features we can use for our purposes. After this, it can be handy to introduce a train-test split. This way we can test and train our model on different data sets which originate from the same classification problem. For this, sklearn also provides an excellent method[3].

[2] make_classification
[3] train_test_split

We can now test on objective, easy to produce data sets which can be more accurate then immediately testing on real-world data. In section 4 we will however provide evidence for our conclusions with some real world case studies.

2.4.1 Resampling methods

A common way to combat the issues of class imbalance is through the use of resampling. In figure 1 the three main resampling methods are illustrated.

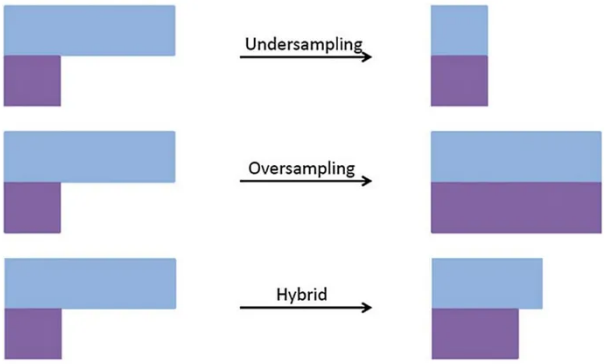


Figure 1: Resampling [5]

There are various models for each of those resampling methods. We will use the most common ones that are present in the imbalanced-learn [9] library. We will use tomek links for undersampling, SMOTE for oversampling and various other models when studying the hybrid method.

2.5 Accuracy calculations

In the final step of our active learning cycle (algorithm 1), we output an accuracy score of the algorithm. However, how do we score how good a model is performing? A simple way to do this would be just to ask the model to score itself because most scikit-learn [15] models exhibit this functionality. But, in section 3.1 we will observe this is not always the best option. We can also just ask the model to predict the labels of our testing data set and implement a new accuracy function ourselves. We implemented 2 different other ways to score the model. These are the balanced_accuracy and the gmean method. They take a weighted average instead of an absolute one. Their implementation is illustrated by algorithm 2 and algorithm 3.

Algorithm 2 Arithmetic mean

Input: *predicted_y*, *actual_y*

procedure GET_BALANCED_ACCURACY
  actual ← count **of** unique *y* **in** *actual_y*
  correct ← count **of** *y* **in** *actual_y* = *predicted_y*
  accuracy ← 0
  for unique *y* **in** *actual_y* **do**
    $weighted\_accuracy \leftarrow correct_y \ / \ actual_y$
    $accuracy \leftarrow accuracy + weighted\_accuracy$
  $N \leftarrow \#$ unique *y* **in** *actual_y*
  $accuracy \leftarrow accuracy \ / \ N$
return *accuracy*

Algorithm 3 Geometric mean

Input: *predicted_y*, *actual_y*

procedure GET_GMEAN
  actual ← count **of** unique *y* **in** *actual_y*
  correct ← count **of** *y* **in** *actual_y* = *predicted_y*
  accuracy ← 1
  for unique *y* **in** *actual_y* **do**
    $weighted\_accuracy \leftarrow correct_y \ / \ actual_y$
    $accuracy \leftarrow accuracy \times weighted\_accuracy$
  $N \leftarrow \#$ unique *y* **in** *actual_y*
  $accuracy \leftarrow \sqrt[N]{accuracy}$
return *accuracy*

3 Results and discussion

In this section we will look at the results of all our different experiments and discuss our findings.

3.1 Accuracy

First and foremost we researched the difference in accuracy between balanced and imbalanced data on active learning. Not only is this the core of this paper, but it also provides a baseline for the following experiments. We can then reason if they will improve or worsen the model.

In figure 2a we see the confusing result that the imbalanced data performs better. This is very counter intuitive, and simply not true. When working with imbalanced data it is always a good idea to use balanced accuracy or gmean (algorithms 2 & 3).

3.1.1 Balanced accuracy and Gmean

Since the accuracy is a bad indicator for how good or bad our model performs, we redid the experiment, but now we measured the balanced accuracy and the gmean.

The results we get in figure 2b and 2c are to be expected. Active learning gives a great improvement over random sampling while an imbalance in the data makes the model perform worse. Both balanced accuracy and gmean are great ways to measure our models. Since balanced accuracy is somewhat simpler and more used, we will be using it for all future results.

3.2 Classifiers

In a next experiment we compare different estimators to see which ones perform better with active learning. In the previous section we only used logistic regression which produced good and accurate results, but maybe other classifiers are preferred or should be completely avoided. In online literature we learned that neural networks should perform really well with active learning [3, 18]. The same holds for a random forest classifier [14]. We however also used some other estimators to cover a lot of bases. The complete result of the test with all the classifiers can be found in figure 3. We observe that most estimators perform quite well. There are however some outliers like the mixture model classifier. We find this result quite strange so we will discuss this further in section 3.4 where we use another query strategy to support this model. We can also observe that the neural network (MLP) and the random forest classifier do indeed perform really well when using imbalanced data sets with active learning. These models could provide a solid starting point for other imbalanced data sets. However, in general the results can change depending on the data set.

3.3 Query by committee

With enough different estimators to bundle into a single ensemble we can take a look at a different query sampler. In theory the query by committee strategy provides a lot of benefits at the cost of some computing power. We hope this also happens when using imbalanced data sets. The results of the experiment can be observed in figure 4. We can clearly see that the query be committee strategy does provide a good understanding of the problem. The results we see are the averaged result of the whole ensemble, but the averaged result of just using uncertainty sampling still outperforms it. It would seem that, although this query strategy provides a solid and very consistent model, query by committee shouldn't be preferred over more simple query samplers. Even more so when we factor in the amount of time it takes to train all the different models. Just selecting a single good model like a neural network would be preferred.

3.4 FourD's

In section 3.2, we saw that the mixture model classifier does not perform well on our generated data sets. We find these results strange so we have tried a diverse number of things to improve its accuracy. One of these things is the usage of another new query strategy called 4D's. This query sampler is build explicitly for use with mixture models so in theory it should improve our results.

However, in figure 5a we can observe that the model does not improve. In another figure (figure 5b) we can clearly see that the model does provide decent results when working on balanced data sets. This means that mixture models should be avoided when training on imbalanced data sets. We think this is something worth looking even further into but we can already conclude imbalance plays a huge role here.

3.5 Skew

When generating a synthetic data set so far we've only used an imbalance of 0.8 to 0.2. But, it could be worth looking into other variations of the skew in our imbalance to know its effect. We've tested this exact thing in an experiment where we run the same model on the same kind of data sets but with a skew ranging from 0.5/0.5 to 0.9/0.1. The results of this experiment are shown in figure 6. The results are intuitively obvious. The balanced accuracy gets worse the more imbalanced the data set becomes. This happens regarding of the underlying classifier we use. Even more interesting to note is that the model gets exponentially worse with its imbalance. This implies that there should be a point where the model's accuracy will be unacceptable in all cases. In future research papers, it could be interesting to study whether active learning raises this floor or not.

It might also be interesting to see the per class accuracy, and look how it behaves when becoming more or less imbalanced. In figure 7 we can see the accuracy per class. *Class 1 true* means that the model correctly predicted a sample from class 1, whereas *class 1 false* means it predicted class 1 when it should have been class 2. We notice the following in the results: the minority class gets the worse accuracy, and thus the lowest false rate. It is the opposite for the majority class. Depending on the use case of a model it could be important to keep this in mind. We will discuss this further with an example in section 4.1.

3.6 Classes

So far we have only looked at classification problems with two classes, but what happens when we introduce more classes? We will look at the balanced accuracy of random sampling and uncertainty sampling with linear regression and with a neural network (MLP). The results can be found in figure 8.

To put everything into perspective we also tested the influence of more classes on balanced data. We can clearly observe that the balanced accuracy drops when classifying for more classes. It is interesting to note that the balanced accuracy of the imbalanced data set drops linearly together with that of the balanced data set.

A good way to counter the balanced accuracy's reduction is to use a model that is better suited for handling multiple classes. Just as you would when using a balanced data set. Here we opted for the use of a neural network to improve over linear regression.

3.7 Resampling

When working with imbalanced data, one of the most common solutions is the data-level solution resampling. We test this using two different hybrid resampling methods in figure 9a. This yields great results. We get a 5% balanced accuracy improvement for free. The problem here however is that we used the labels in order to resample the data. This is something we cannot do since the labels will only become available during the active learning cycle.

With this in mind we can try to perform resampling during the active learning cycle. We do this by slightly altering the core of the cycle, as show in algorithm 4.

Algorithm 4 Active learning cycle with resampling

```
procedure GET_ACCURACY
    for _ in 0..N do
        y_new ← query from qs
        add y_new to y_known
        X_r, y_r ← resample X, y_known
        fit clf on X_r, y_r
        score ← score from clf
        output score
```

Since a lot of the resampling algorithms need k-neighbours to work, we only started resampling as soon as each class had k+1 known samples. In figure 9b we can see this yields almost no improvements or drawbacks over the normal active learning cycle. Except when we are using under sampling like tomek, then the performance drops.

3.8 Batches

For a human annotator it is often far more convenient to label a group of samples all at once in comparison to labeling a single sample once in a while. This however comes with its own share of drawbacks. These problems are already addressed in various other papers and we won't focus on them here [4, 7]. We simply want to research if the introduction of class imbalance adds additional downsides or not.

In figure 10a we can see the comparison of balanced versus imbalanced data when using batches. The gap in balanced accuracy is in line with the balanced accuracy gap of normal active learning.

Next we looked at the influence of the batch size on our model. This is shown in figure 10b. The balanced accuracy drops when the batch size gets larger.

It looks like generally most of the problems with batches are kept when using imbalanced data. To combat them, we can use the same tweaks for batches as shown in other literature. We should however stay especially vigilant with our imbalanced data sets because many problems are exaggerated and amplified.

3.9 Batch fairness

An interesting thing to study about batches is how fair the query sampler is in selecting new instances of the data. We've already defined this fairness in formula 1. We will now take a look at a simple test case with two different batch algorithms. You can see the results of this test in figure 11a. From these results we can assume that the normal batch algorithm is better than the batch bald model, but what about the batch fairness? The fairness for each feature of the test data set can be seen in table 2.

Model	Class 0	Class 1
Normal Batch	50.64%	49.36%
Batch Bald	73.81%	26.19%

Table 2: Batch fairness of the models in figure 11a.

The normal batch model is way more fair when it is choosing its samples compared to the batch bald algorithm which leans more towards the imbalance of the classes itself (class 0 has weight 0.8 to class 1 with weight 0.2). When testing on a balanced data set, this fairness is corrected and we achieve way better results as shown in figure 11b. This is in line with other research [7]. Batch bald now has a way better batch fairness as shown in table 3.

Model	Class 0	Class 1
Normal Batch	49.75%	50.25%
Batch Bald	55.61%	44.39%

Table 3: Batch fairness of the models in figure 11b.

There is clearly a correlation between our defined batch fairness and the performance of the model when using an imbalanced data set. It could be important to keep this in mind when working with heavily balanced data sets. In the future, we could more extensively research which query strategies inherently provide a better batch fairness or what the impact of the testing data sets itself is on this batch fairness.

4 Case-study

Now that we have researched various ways that influence the performance of active learning when using imbalanced data, we will look at some real world examples to verify our results.

4.1 Breast cancer

The first data set [19] we will look into contains information about breast cells and whether or not they are malignant. Each cell has 10 features and is labeled either M (malignant) or B (benign). Malignant means the cell is a cancer cell, whereas benign means the opposite. The data set is imbalanced with the following ratio:

Class	percentage
Benign	62.7%
Malignant	37.3%

Table 4: Class imbalance in the breast cancer dataset.

The creators of the data set specified that random forest classification works well on this data set, so we will be using this as our classifier. Like before we will once again use uncertainty sampling as our query strategy. In figure 12 we can see that after its initial struggle the model performs really well. It is interesting to note that the minority class gets a worse accuracy. This is confirmed in figure 13 when we alter the data set to experiment with the skew. The results align with what we found in section 3.5

When getting medical results it is desired to get *true benign* or *true malignant* results. This means that the result of the test is correct.

If the test is *false malignant* it indicates we have breast cancer when we don't. This might be shocking, but is still fine, since further inquiry will show that we don't have cancer. However if the test says *false benign* we get the feeling that we are safe, and are less inclined to do further testing. This is dangerous and should be avoided.

In our results the red line indicates *false benign*. The skew of the data set results in a *false benign* line that is still relatively high. We would prefer a skew in the opposite direction instead. For any doctor using this model trained on the skewed data it is very important to keep the influence of the imbalanced data in mind when forming a diagnose.

4.2 Car evaluation

Another interesting data set is a data set about car evaluation [1]. The data set is different from the breast cancer data set because it has more instances but a lot less features. It also labelled the data with strings which were harder to work with but we converted them to more useful integers.

We trained 3 models on this data set in its entirety, the results of which can be seen in figure 14. We can clearly see that all models perform quite well on this data set but there are some things to note which we have not yet previously mentioned.

The first and fairly obvious thing is the fact that both models, with the logistic regression classifiers, converge to the same accuracy in the end. This happens whether we use active learning or not. This is important to note for real world cases where we have already labelled all the data. In this case active learning does not provide an obvious advantage when training on the complete set.

Secondly we can now observe the impact of the underlying estimator in the problem. We see that our neural network has a huge advantage over the simpler logistic regression model. We can conclude that choosing a good underlying model, specific to your data set, is as if not more important than our other conclusions.

A final observation is that active learning has a massive head-start in the accuracy. This is even more pronounced when we put all the models on the same graph showing only the first 150 samples (figure 15). In most realistic cases this is when you would stop labeling the data as a human annotator and here it is shown that active learning performs a lot better. You can first see the difference that active learning makes and then observe the improvement of the underlying classifier.

5 Conclusion

In conclusion, class imbalance poses a significant challenge when using active learning models, potentially compromising the precision of the model. However, in our research, we found that paying attention to and managing certain factors can help reduce the negative impact of class imbalance, leading to more dependable and accurate results.

Factors such as skew, fairness in the batch selection, our choice of query sampler, and the underlying models all play important roles in addressing class imbalance. Integrating these factors into the active learning cycle allows us to navigate the challenges associated with imbalanced data sets.

By maintaining a keen awareness of these considerations, researchers and practitioners alike can optimize their active learning processes, ensuring a more accurate and reliable outcome, even when working with imbalanced class distributions.

References

[1] Marko Bohanec. *Car Evaluation*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5JP48. 1997.

[2] Jason Brownlee. *Develop an Intuition for Severely Skewed Class Distributions*. Last accessed 20 december 2023. URL: https://machinelearningmastery.com/how-to-develop-an-intuition-skewed-class-distributions/.

[3] D. Cohn, L. Atlas, and R. Ladner. *Improving Generalization with Active Learning*. Last accessed 20 december 2023. URL: https://link.springer.com/article/10.1007/bf00993277.

[4] Ahmet E. Bayraktar. *Batch Mode Active Learning*. Last accessed 20 december 2023. URL: https://medium.com/wbaa/batch-mode-active-learning-8e1e8323a9e1.

[5] Eqibuana. *How to Deal with Imbalanced Data in Classification Tasks?* Last accessed 19 december 2023. 2021. URL: https://miro.medium.com/v2/resize:fit:1100/format:webp/1*EdNwlqR2gxXHRKeJcvq8_A.jpeg.

[6] Daniel Gissin. *Discriminative Active Learning: Review*. Last accessed 20 december 2023. URL: https://dsgissin.github.io/DiscriminativeActiveLearning/.

[7] Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. *BatchBALD: Efficient and Diverse Batch Acquisition for Deep Bayesian Active Learning*. Last accessed 20 december 2023. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/95323660ed2124450caaac2c46b5ed90-Paper.pdf.

[8] Daniel Kottke et al. "scikit-activeml: A Library and Toolbox for Active Learning Algorithms". In: *Preprints* (2021). DOI: 10.20944/preprints202103.0194.v1. URL: https://github.com/scikit-activeml/scikit-activeml.

[9] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning". In: *Journal of Machine Learning Research* 18.17 (2017), pp. 1–5. URL: http://jmlr.org/papers/v18/16-365.

[10] Brent Matthys and Mats Van Belle. *Class imbalance in active learning*. 2023. URL: https://github.com/brmatthy/Class-Imbalance-in-Active-Learning.

[11] modAl. *Query by committee*. Last accessed 20 december 2023. URL: https://modal-python.readthedocs.io/en/latest/content/examples/query_by_committee.html.

[12] Natuurpunt. *Rapport het grote vogel weekend 2023*. Last accessed 20 december 2023. 2023. URL: https://www.natuurpunt.be/sites/default/files/images/inline/rapport_hgvw_4-2023_finaleversie_16052023.pdf.

[13] Barack Or. *Solving The Class Imbalance Problem*. Last accessed 20 december 2023. URL: https://medium.com/metaor-artificial-intelligence/solving-the-class-imbalance-problem-58cb926b5a0f#:~:text=Class%20imbalance%20is%20a%20common%20problem%20in%20machine%20learning%20that,can%20negatively%20impact%20its%20performance..

[14] Petra Ormel. *The Basic Concept of Active Learning*. Last accessed 20 december 2023. URL: https://amsterdamintelligence.com/posts/active-learning.

[15] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

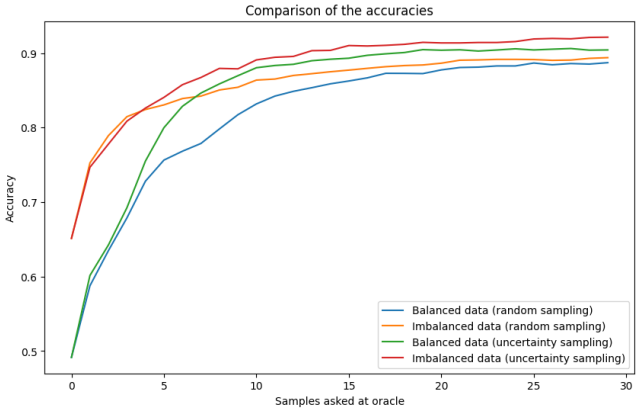
[16] Tobias Reitmaier and Bernhard Sick. *Let us know your decision: Pool-based active training of a generative classifier with the selection strategy 4DS*. Last accessed 20 december 2023. 2013. URL: https://www.sciencedirect.com/science/article/abs/pii/S0020025512007682?via%3Dihub.

[17] Bur Settles. *Active Learning Literature Survey*. Last accessed 20 december 2023. URL: https://burrsettles.com/pub/settles.activelearning.pdf.

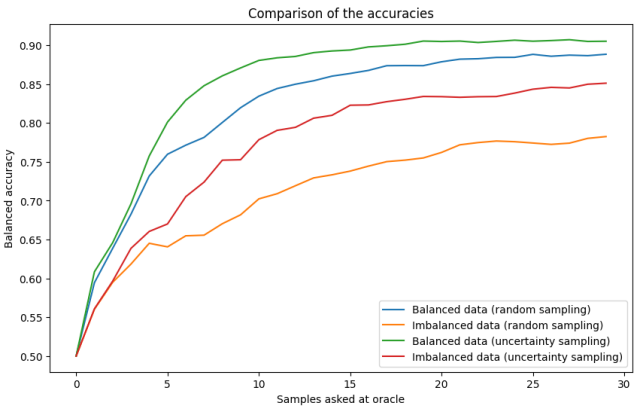
[18] Shoujin Wang et al. *Training deep neural networks on imbalanced data sets*. 2016. DOI: 10.1109/IJCNN.2016.7727770. URL: https://ieeexplore.ieee.org/abstract/document/7727770/metrics#metrics.

[19] William Wolberg et al. *Breast Cancer Wisconsin (Diagnostic)*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5DW2B. 1995.

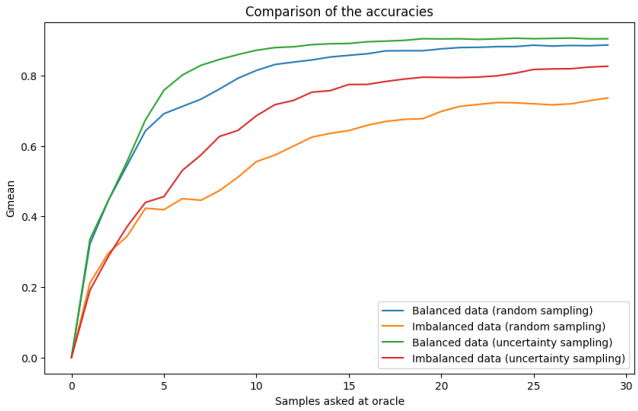
A Figures



(a) Accuracy of balanced vs imbalanced [0.8,0.2] data on active learning and random sampling.



(b) Balanced accuracy of the same problem.



(c) Gmean of the same problem.

Figure 2: Importance of using a weighted accuracy when using imbalanced data sets.

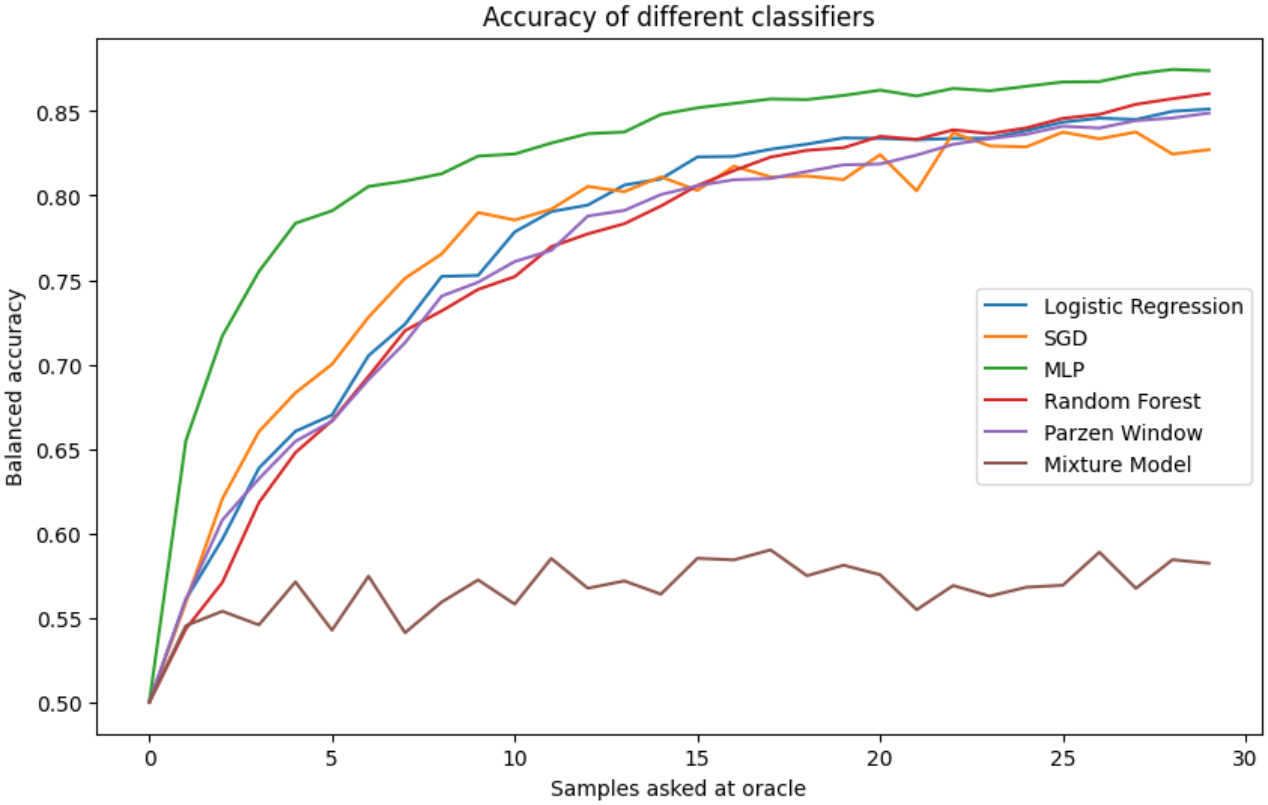


Figure 3: Averaged results of comparing different estimators against the same 100 imbalanced data sets.

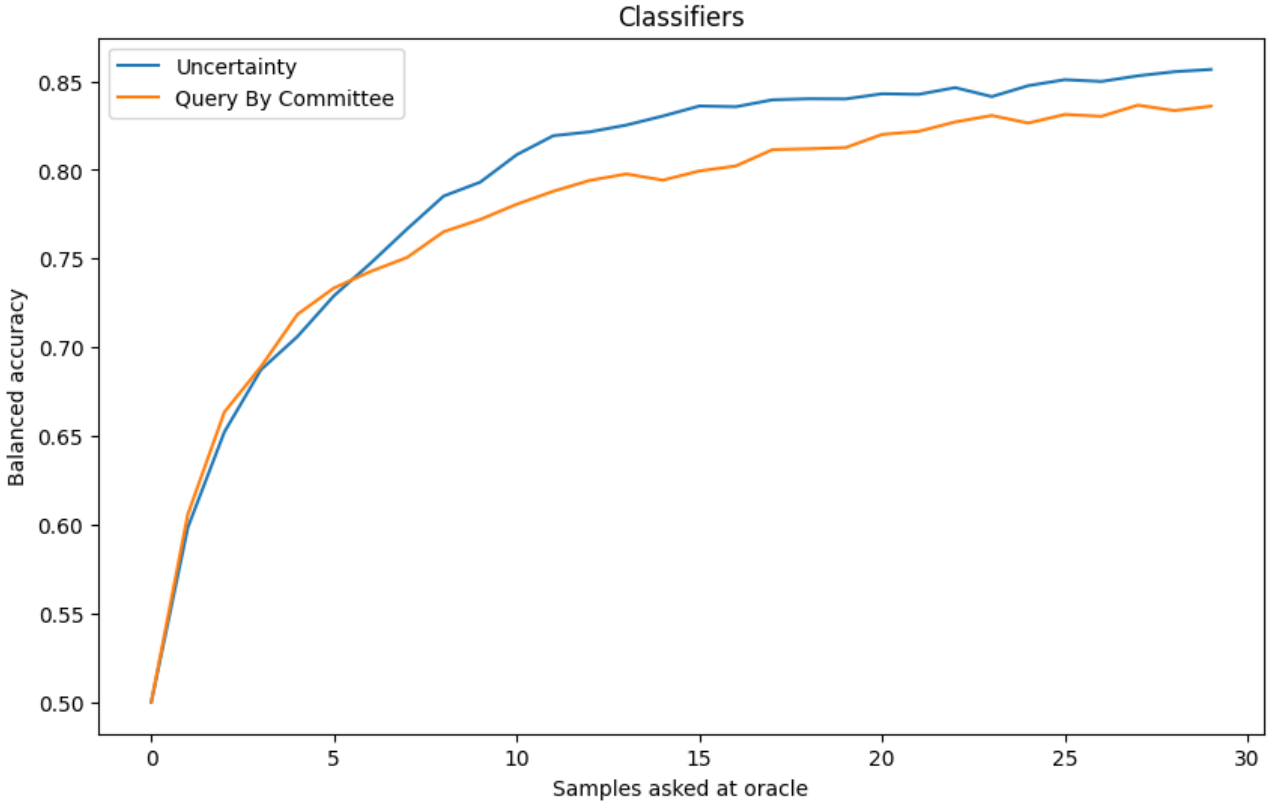
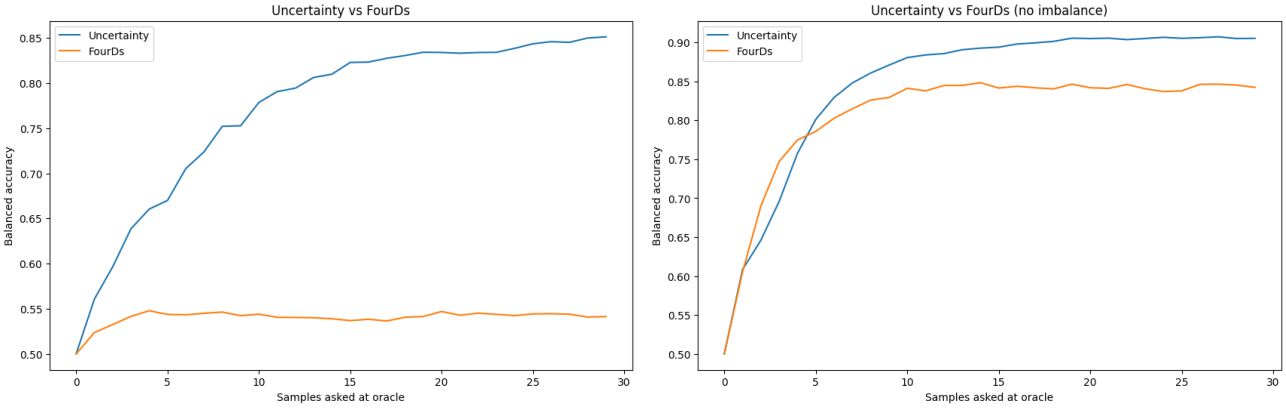


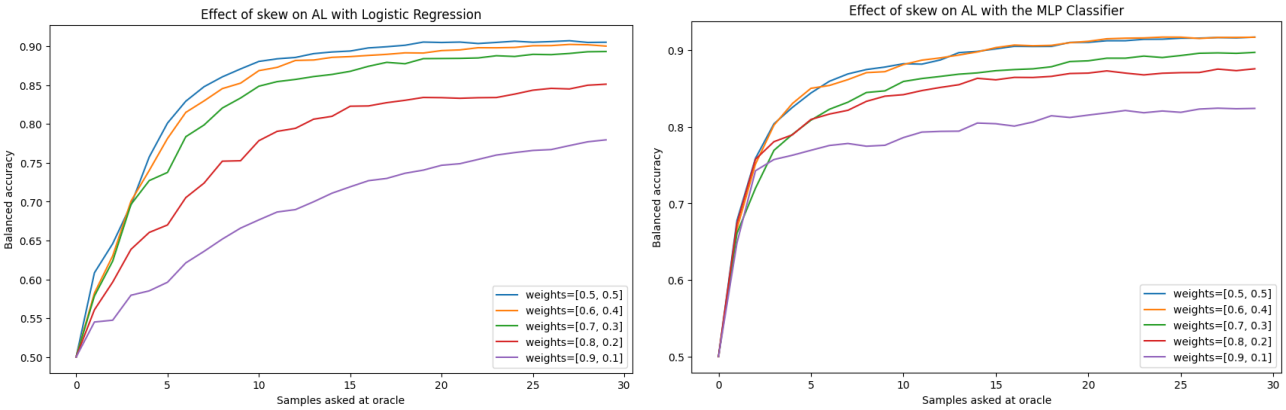
Figure 4: Averaged results of a query by committee ensemble against the same ensemble averaged after using uncertainty sampling.



(a) Results on imbalanced data sets.

(b) Results on balanced data sets.

Figure 5: Results of a model using a Mixture Model classifier and the FourD's query strategy against a baseline with uncertainty sampling.



(a) Results with logistic regression.

(b) Results with the mlp classifier.

Figure 6: Comparison of the accuracies with different skews in the imbalance when using active learning.

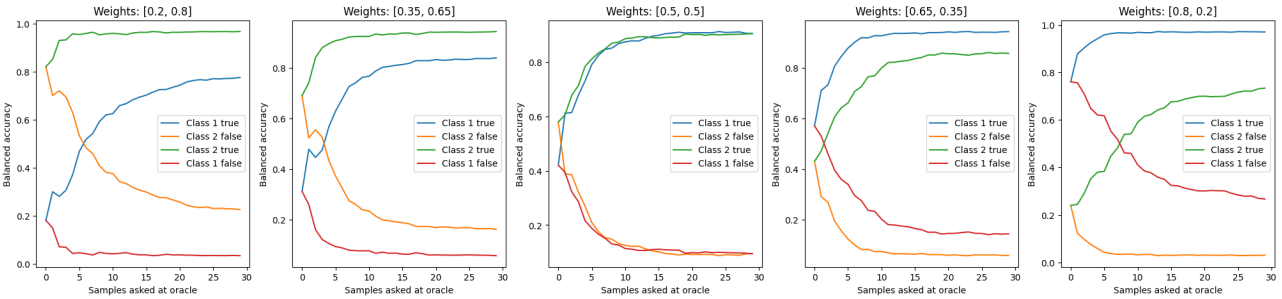


Figure 7: Skew per class

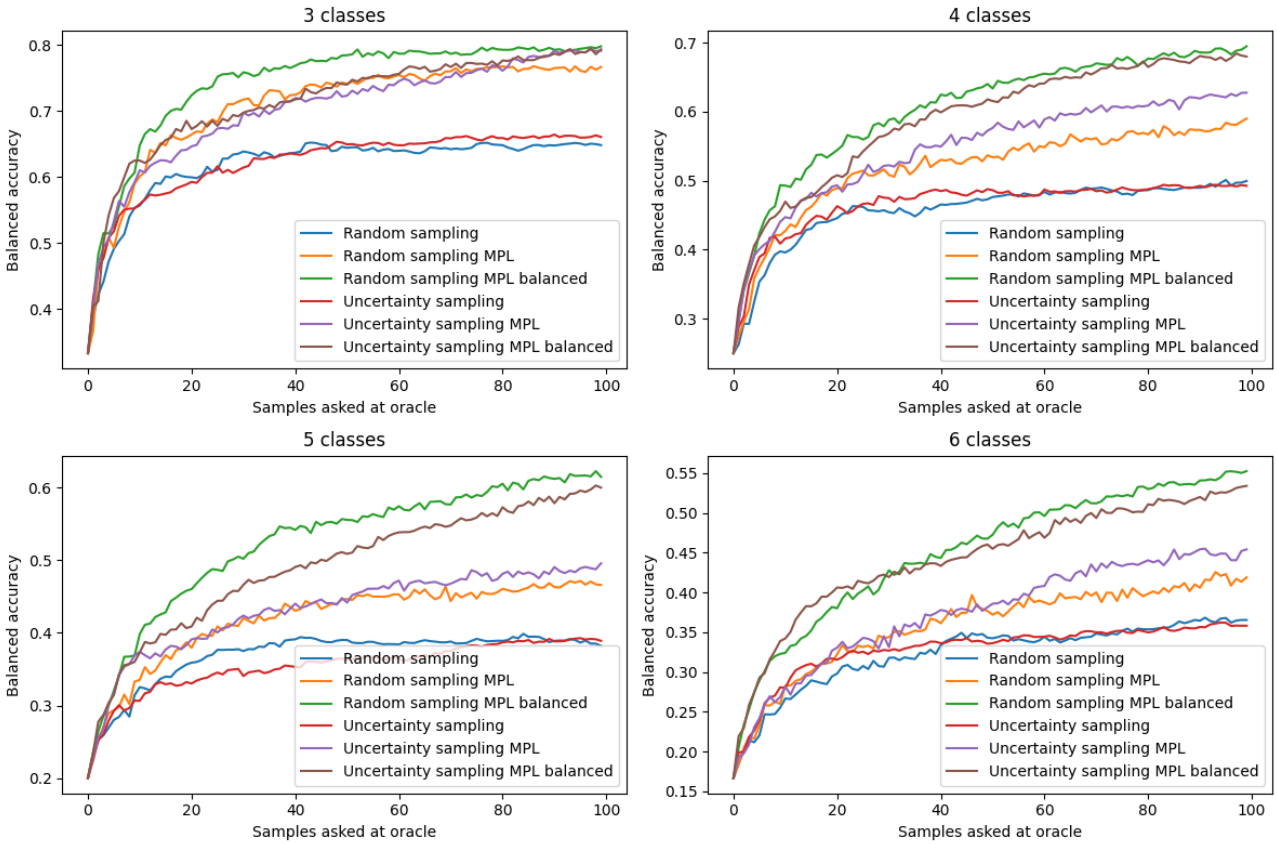
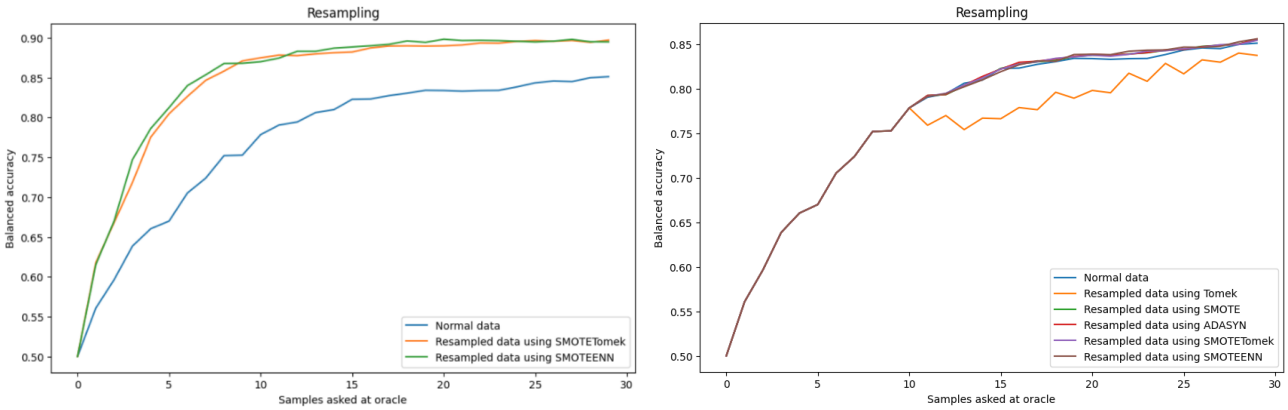


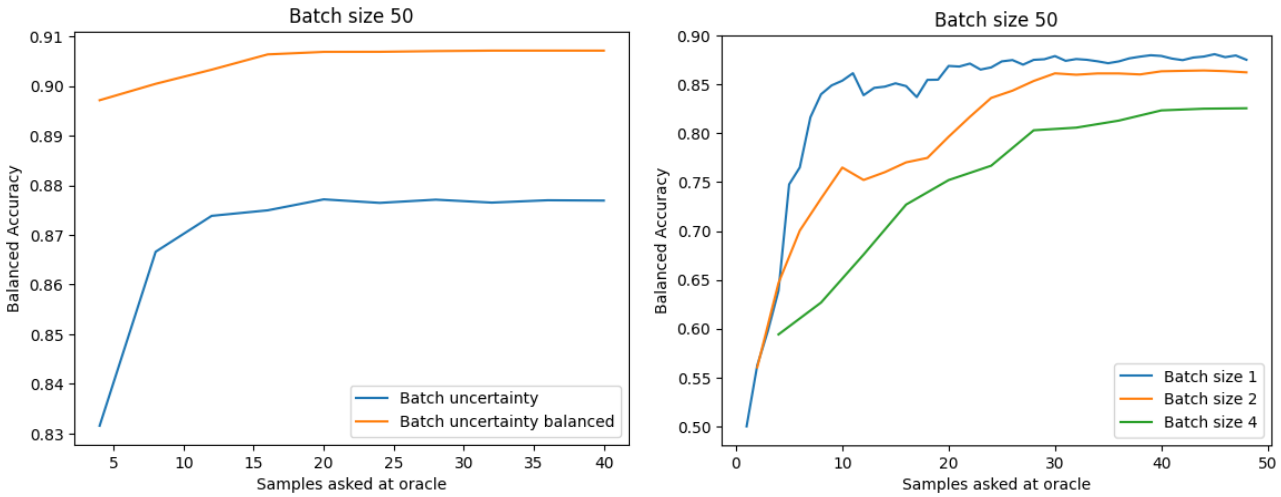
Figure 8: Multiple classes



(a) Resampling preprocessing

(b) Resampling during AL cycle

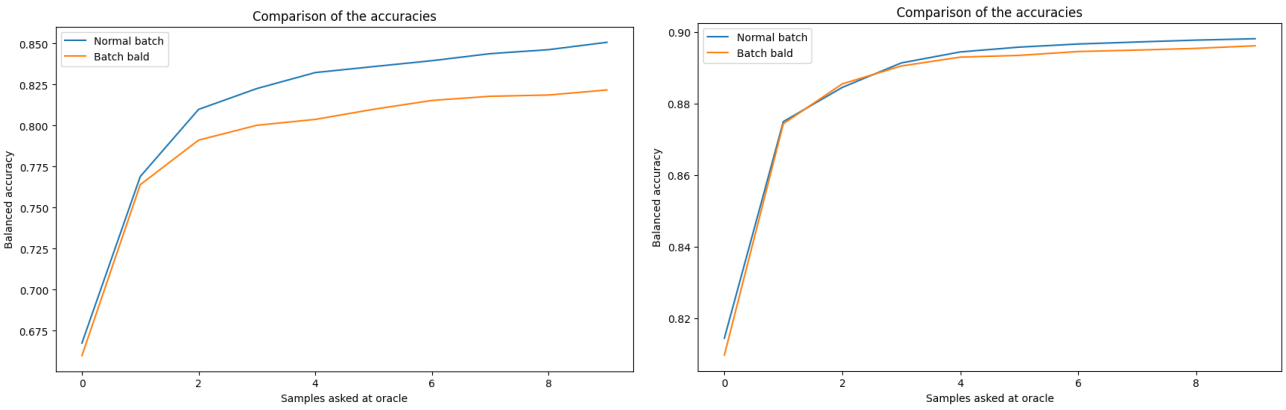
Figure 9: Resampling in our active learning model.



(a) Balanced accuracy of the (im)balanced data set.

(b) Balanced accuracy for different batch sizes

Figure 10: Results when using batches in the active learning cycle



(a) Using imbalanced data.

(b) Using balanced data.

Figure 11: Accuracy of batch function build into skactiveml with uncertainty sampling against the batch bald algorithm.

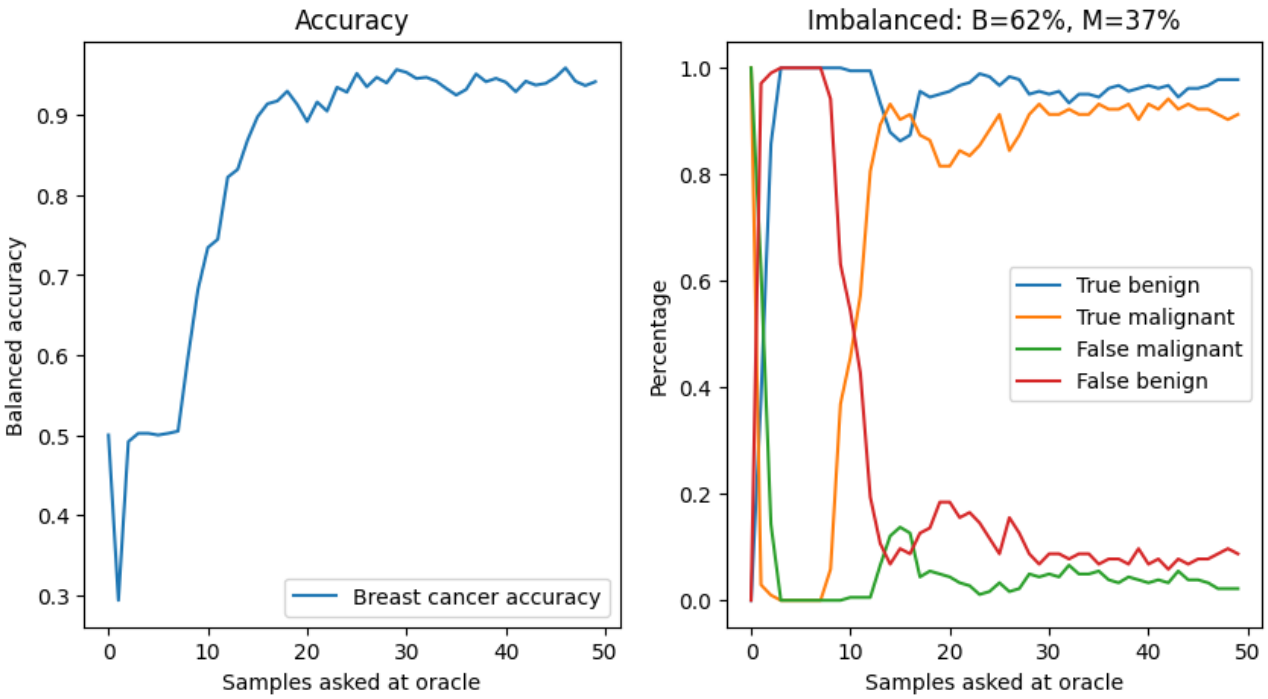


Figure 12: Breast cancer

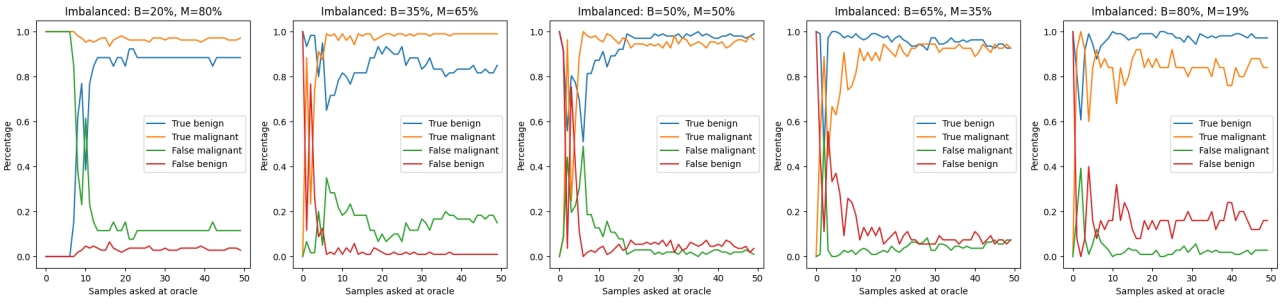
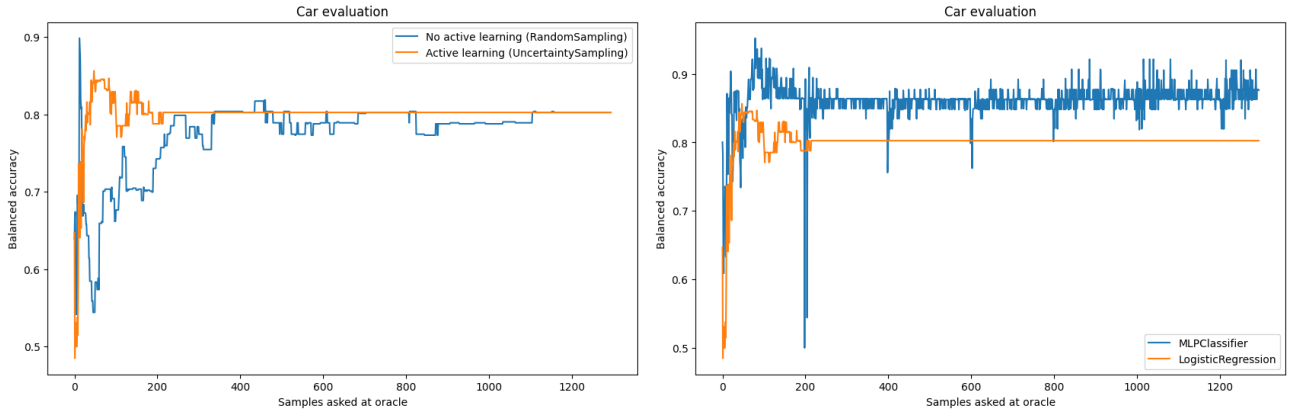


Figure 13: Breast cancer skew



(a) Results of (no) active learning with Logistic Regression. (b) Results of MLP/Logistic Regression with active learning.

Figure 14: Accuracy of certain models on the car evaluation data set trained in its entirety.

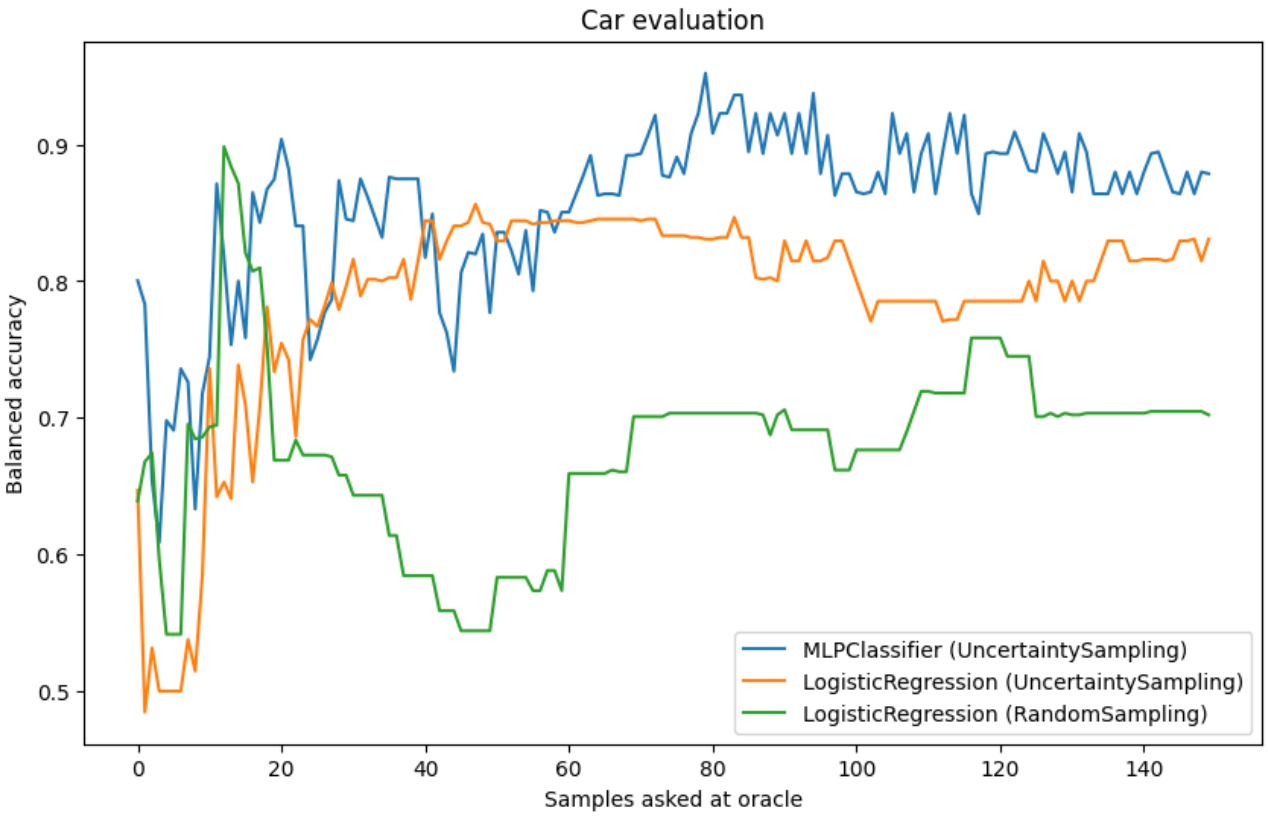


Figure 15: Accuracy of three models on the car evaluation data set.