Annotating data in active learning more efficiently

Diana Maria Conceição Mortágua, Rama Shriki
2020242508@student.uc.pt, 2020242366@student.uc.pt

University of Coimbra

1 Introduction

A lot of *machine learning* techniques require a large amount of training example and, most of the times, the size of the training set will have a positive correlation with the accuracy of the model. However, as machine learning supervised models are used for new objectives (usually more and more challenging), we need more training examples and, if we need labelled examples, someone needs to label them, which is time consuming and costly for the company that pays humans who do it. In some cases, it is even unrealistic having dozens of people just labeling examples for machine learning algorithms. Still, unlabeled examples are usually easy to find and don't take a lot of time to get ready to use. For these reasons, researchers started searching about how they could use unlabeled examples in an efficient way, where you could get similar results with less labeled data. This is how the proposal of active learning came to existence.

Active learning is a subfield of ML that has gained popularity thanks to its great benefits to humans. The *artificial intelligence* (AI) algorithm is given only a few annotated examples of data, which he gets trained with. After an iteration, it uses a function to decide on which small number of unlabeled data it wants to have annotations on. This new subfield has been receiving attention since it shortens time needed to train algorithms, increases accuracy, and reduces the number of samples that will be labelled, which results in lower financial costs, thanks to less working hours.

There are many ways to select which data samples the algorithm will request annotations for. These are called *sampling strategies* and the most known are *random*, *uncertainty* and *diversity sampling*. Those samples are sent to the annotators to be labeled and then sent back to the algorithm. The label selected can come from a singular person or a committee/oracle. In this work we will explore the use of *query-by-committees (QBC)*, where an oracle gives its view about the labels.

The purpose of this work is to explore this topic and its applications to be able to explain how committees work better or worse in active learning. We intend to understand this by using a lot of tests. First, we will try an oracle where all members have the same knowledge/training on the data, where will give the same value/weight to each opinion. After that we will also try with members who have different levels of training and, so, their opinions have different wights. Then, we will also try with just one annotator to compare the results. Different experiments will also be done with the numbers of members in the oracle, so that we can try to understand what works better for higher accuracy/f1-score and why, and also with different estimators, to be able to compare our findings to previous work and also to share our own results.

This work is divided into 6 sections, starting with the introduction and moving on to related work where we will explore results of previous related work. In section 3 we will go into further detail about our data and approach and then, in section 4, share our experimentation. Finally, we will state our conclusion and then present our bibliography.

2 Related work

Liere and Tadepalli concluded on their paper [3] "Active Learning with Committees for Text Categorization" that "active learning with committees is the best approach when one as a limited supply of labeled examples" since it used a lot less training example as the supervised learners, it has the best average accuracy, and it is the one that has less execution time. This conclusion is consistent with the state of art on the topic, as well as with Stefanowski and Pachocki's paper [2] called "Comparing Performance of Committee Based Approaches to Active Learning" where they concluded that "using a relatively small number of examples – well selected for labeling – it is possible to generate a final classifier characterized by an accuracy comparable to passive approaches using much larger set of examples".

In this last paper the team decided to experiment with different models to see how they interfered with the results. To continue their research, we decided to do the same thing to try to understand if different models have any correlation with the number of members when using QBC.

By trial and error, Liere and Tadepalli concluded that the 7 was the best number of members on the oracle. We will also compare if our research shares that conclusion.

3 Data and approach

In this work we used the famous *load_digits* dataset from *scikit-learn* with 8x8 images of digits and the true classes of those digits (the true number the images show). To train the

models, we used the dataset's annotations but, for the 5 images that the models were most uncertain about, we used the annotation that comes from the oracle. The algorithm will be trained to answer correctly saying the number that is shown in the picture, so, we will do this work as if we don't have already a supervised dataset, but rather as if we only have the training data. The test data will be used as if we don't know the true label. We adapted the code available in [4] to the experiments we will run about QBC based on the website [5] recommended by our professor. We used python to run our experiments.

Our approach was to run the tests with the different categories explained in the last section and save the f1-score and accuracy of all. In the end we created graphs to visually show the results of our experiments, so that we can easily take conclusions. We decided to use accuracy for this comparison since it is the metric the related works used, and we decided to use f1-score to see if the results on these two metrics would be similar.

To define our training and test we used sklearn.model_selection's train_test_split with test_size being 0.8, and random_state being 42. We did 5 iterations through 20% of the test data to use our oracle 25 times (5 times labelling the 5 most uncertain images in that iteration).

4 Experimentation

As mentioned earlier we did tests where our oracle has 3, 5 and 7 members, so we could conclude which number is best. We decided to stop at 7 since having 9 members for a dataset this small would be, in our opinion, against our objective of having a balance and effectiveness between the effort in labeling and its costs, and the results. We also had a case with only one member labeling the 5 images that the model was more uncertain about.

The models we used were SVC, K-nearest Neighbors (KNN, k=5), Gaussian Naïve Bayes and Random Forest with 50 estimators (since it was used in related work [2] with the best results). We chose very well-known models because, if they also had very good results on AC (like Random Forest has shown to have), it would be an advantage for beginners learning about the field.

We have 4 types of QBC. One where all the members have the same training (knowledge), another one where ¼ of the members (rounded up) had twice the training as the other ¾ and their weight was 1.5 (let's call this situation 2 moving forward), another one where ¼ of the members (rounded up) had twice the training as the other 1/2 and their weight was 1.5 and the other one where ¼ of the members (rounded up) had three times more the training as the other 1/2 and their weight was 2 (lets call this situation 3 moving forward).

Our code is a combination of the original code in on scikit learn website [4] and the code on committees suggested by our professor [5]. We experiment with all the combinations possible, and results about f1-score are in Fig 1, 2 and 3, and results on accuracy on Fig 4, 5 and 6.

Furthermore, the results for the solo annotator where 0.96, 0.957, 0.794, and 0.926 for the f1-score and 0.96, 0.957, 0.7955 and 0.926 for accuracy; regarding svc, k-nearest neighbors, gaussian naïve bayes and random forest respectively.

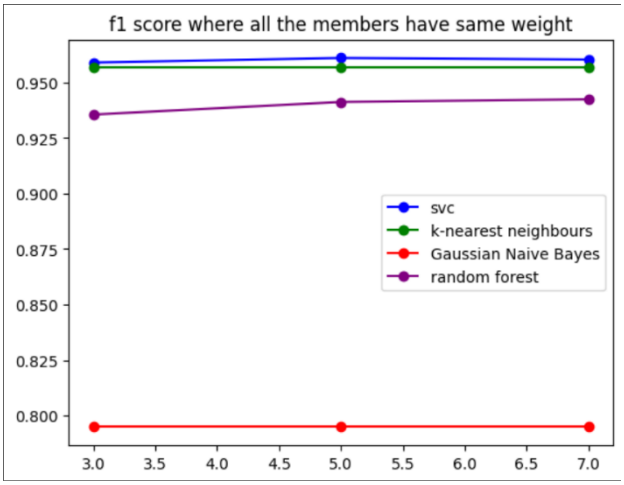

Fig 1 – F1-score when all members have the same weight

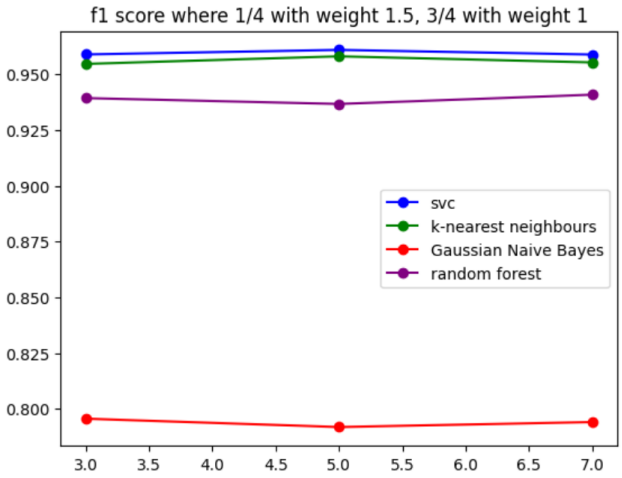

Fig 2 – F1-score in situation 2

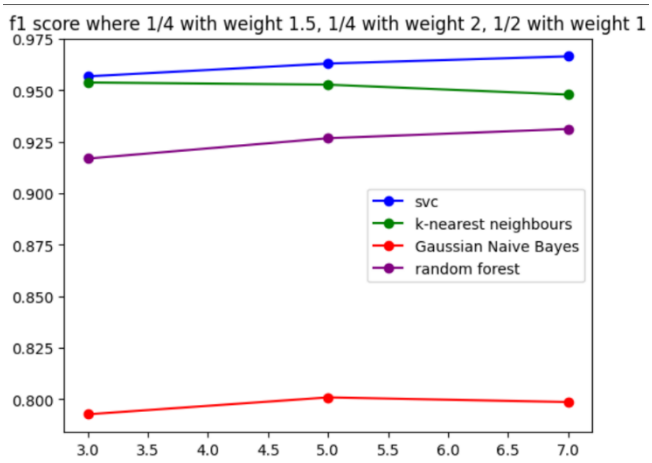

Fig 3 – F1-score in situation 3

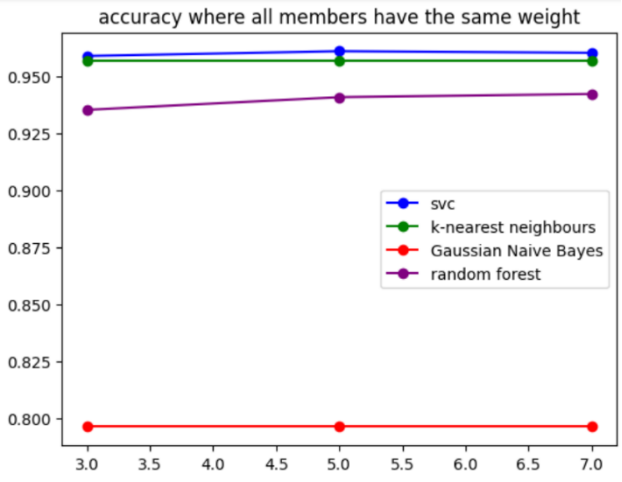

Fig 4 – Accuracy when all members have the same weight

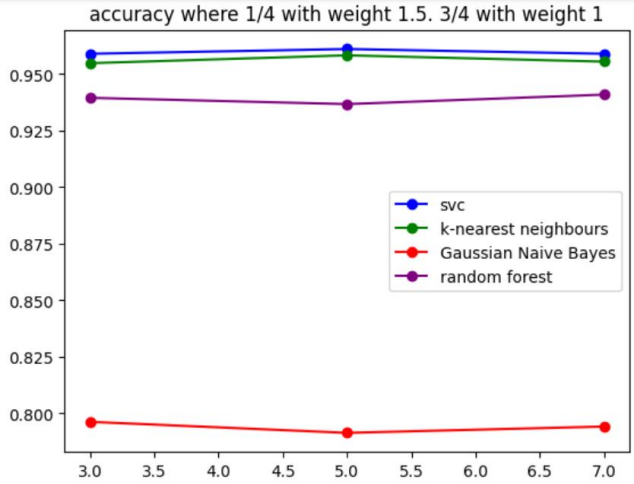

Fig 5 – Accuracy in situation 2

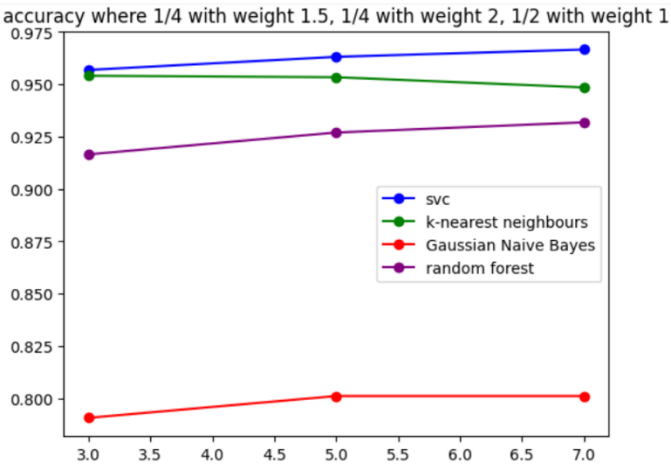

Fig 6 – Accuracy in situation 3

5 Conclusion

From the results we can clearly see which model was the best for our dataset. Gaussian Naïve Bayes (NB) was much worse than the other and we assume the reason for that is the distribution of the data. After Gaussian NB we can see Random Forest with more than 0.1 accuracy value difference, which is followed by K-nearest neighbors (KNN) and then SVC (which have similar results in all the experiments).

Random Forest performed well as we expected from previous works on the field. It can handle complex patterns and interactions between features and, so, it works well on our dataset. It was able to capture the underlying structure of the data effectively and, so, we are happy with its results.

We also used an SVM to check of the features were linearly separable and we got score 1 which means they are. We believe that's why SVC had such a great performance. We can conclude that, for this dataset, SVC and K-nearest neighbors are the best options when it comes to active learning, with random forest closely following.

When it comes to the number of members in the committee's oracle, we can see that the performances in our options were all very similar, with a tendency to have some more decimals score in performance when we increase by 2 the number of learners. Because of that, even though the difference is not too big, we can conclude that having 7 opinions on the oracle is better than having 3 or 5 (regarding this dataset).

Similarly, the different types of committees we have (when it comes to weights) doesn't make a huge difference. We can see that the performance of Random Forest decreases as we have more diversity in weight in our members of the oracle, while Gaussian NB's performance increases. Other than that, we can see that when we have more diversity on amounts of training and weight, SVC performs a little better while K-nearest neighbors performs a little bit worse. It's also interesting that when all the members of the oracle have the same training and weight, and in situation 2, SVC and KNN perform very similarly and are always very close to each other but, in situation 3, they begin with similar scores when we have only 3 members but as we increase that number, KNN's score decreases while SVC's increases. That might have to do with the characteristics of the models and so, we believe maybe KNN could be overfitting while SVC could be able to better handle the increased variability and complexity in the data.

The values for our solo annotator where like the ones we see with 3 members on the oracle. It's makes sense that it is similar to our experiment with less members on the oracle since we already expected from previous work that 7 members would be better than 3 but it's surprising how actually close the values were.

Even though our results on the number of oracle's members weren't very evident, we can still see the trend of the scores and conclude that 7 members was the best option in this context, as it is using SVC. The reason that made our research not have so evident results might be for the size of our dataset (that doesn't have a big amount of data – it only has

1797 samples) combined with its nature (being used to recognize pictures). Nevertheless, we are happy that our results on the number of members of the oracle is supported by previous research and we found interesting how our models performed when we changed the weights of different members. It helped us realize that doing some experiments might be a very good thing to do before doing an AC work to know which type of committees would be better for the task we have in hands.

6 Bibliography

[1] Settles, J. (2010). Active Learning Literature: A Basic Guide. Burr Settles. https://burrsettles.com/pub/settles.activelearning.pdf

[2] Stefanowski, J., & Pachocki, M. (2009). Comparing Performance of Committee Based Approaches to Active Learning.

[3] Liere, R., Tadepalli, P. Active Learning with Committees for Text Categorization

[4] Scikit learn. Label Propagation digits active learning https://scikit-learn.org/stable/auto_examples/semi_supervised/plot_label_propagation_digits_active_learning.html#sphx-glr-auto-examples-semi-supervised-plot-label-propagation-digits-active-learning-py

[5] Tivadar Danka Revision. ModAL. Committee https://modal-python.readthedocs.io/en/latest/content/models/Committee.html