GE 461 - INTRODUCTION TO DATA SCIENCE

PROJECT – Data Stream Mining

Necati Furkan Çolak / 21803512

Introduction:

This project was designed to explore the application of various online learning classifiers to streaming data. Synthetic data generated from the AGRAWAL and SEA generators, along with the real-world "spam" and "elec" datasets, were used to simulate different environments for our models.

The primary learning model used in this project was the Hoeffding Tree classifier. To ensure our model could adapt to concept drift, we also introduced a set of alternative classifiers, which included the Adaptive Random Forest, Self-Adjusting Memory k-Nearest Neighbors (SAMKNN), Streaming Random Patches, and Dynamic Weighted Majority classifiers. These models served as potential replacements for the primary model in case of a detected concept drift.

Concept drift is detected using the Adaptive Windowing (ADWIN) algorithm, a popular change detector that decides the statistically significant change in the data stream. When ADWIN detects a change, the primary classifier is replaced with the alternative classifier that has the highest correct prediction rate.

This project aimed to provide a comprehensive introduction to online learning and concept drift detection in streaming data. By the end of this project, we hoped to demonstrate the effectiveness of online learning models in dealing with concept drift and the importance of choosing the right learning model based on the characteristics of the data stream.

C.3):

The purpose of this script is to generate synthetic data for completing tasks using the AGRAWAL and SEA generators [2], save the generated data to CSV files, and then load these datasets along with other pre-existing datasets for further use.

The steps to complete this part can be found below:

1. Initializing generators: The script first initializes AGRAWAL and SEA generators with a specific random seed (1 in this case) to ensure the reproducibility of the generated data.

2. Generating data: The script then enters a loop that runs 100,000 times, generating a single data sample (a set of features and a corresponding target value) from each generator on each iteration.

3. Storing data: It stores the generated data and targets in lists, separately for each dataset.

4. Saving data to CSV: Once all data samples are generated, they are merged with their corresponding target values and saved to CSV files, 'AGRAWALDataset.csv' and 'SEADataset.csv', using the NumPy function `np.savetxt()`. This allows the datasets to be stored persistently and reused later.

5. Loading data: The final part of the script loads several datasets from CSV files into pandas DataFrames. These include the newly generated and saved AGRAWAL and SEA datasets, as well as two other datasets, 'spam.csv' and 'elec.csv'. These pandas DataFrames can then be used for further data processing, analysis, or for training machine learning models.

The script thus combines several steps of the data handling process in machine learning: generating synthetic data, saving it to a file, and loading it for use, as well as loading pre-existing data.

C.4):

The purpose of this script is to simulate a common scenario in online learning or streaming data, where a model learns incrementally from a continuous flow of data. The script applies multiple classifiers to the "spam" dataset and monitors their performance over time, potentially switching the primary classifier when it detects concept drift.

The steps to complete this part can be found below:

1. Preparing the Stream: It first assumes the last column of the dataset as the target variable and prepares a data stream for processing. Then, it separates an initial batch from the stream to initialize the classifiers.

2. Classifier Setup: It establishes a Hoeffding Tree classifier as the primary classifier. It also sets up a list of alternative classifiers, which includes an Adaptive Random Forest classifier, a Self-Adjusting Memory k-Nearest Neighbors (SAMKNN) classifier, a Streaming Random Patches classifier, and a Dynamic Weighted Majority classifier. These classifiers are then initialized with the initial batch.

3. Drift Detection Setup: The script initializes the Adaptive Windowing (ADWIN) change detector for each classifier. ADWIN is a popular method for detecting concept drift in data streams.

4. Prediction Counting Setup: It also sets up counters to track the number of correct predictions each classifier makes.

5. Processing the Stream: It enters a loop to process the data stream. On each iteration, it gets a sample from the stream and makes predictions using all classifiers. If a classifier's prediction is correct, it updates the classifier with the new sample and increments its correct predictions counter. It also updates the ADWIN instance of the primary classifier with each sample and checks if it has detected a change.

6. Concept Drift Handling: If the ADWIN instance of the primary classifier detects a change, the script calculates the correct prediction rate for each classifier and switches the primary classifier to the one with the highest rate.

7. Tracking Accuracy: The script keeps track of the accuracy of the primary classifier over time and periodically calculates the prequential accuracy (an accuracy calculated over a sliding window of recent samples). The sliding window size is set to one-twentieth of the number of samples in the stream, or 1 if the stream has fewer than 20 samples. The accuracies are stored in lists for later analysis.

By doing this, the script enables online learning from a stream of data, with the ability to handle concept drift by switching to a classifier that is currently performing best. This is an important strategy in scenarios where the underlying data distribution can change over time, such as in spam detection, financial forecasting, and many other real-time prediction tasks.

5.1): How does your ensemble model perform compared to the state-of-the-art approaches in 4.1? What could be possible improvements for a more robust ensemble.

Answer:

In the evaluation, I compared the performance of ensemble models with single algorithm training approaches. When applied to real datasets such as electricity and spam datasets, the ensemble approach consistently outperformed the single algorithm training approach in terms of accuracy. This indicates that our ensemble model is more effective in handling real-world datasets. Additionally, the performance of single models in real datasets showed frequent fluctuations, as certain classifiers performed well on specific parts of the dataset while others did not capture the variance as effectively. The Agrawal dataset provided consistent predictions across all classifiers in both the single model approach and ensemble approach. It can be seen in Appendix Table A. This may indicate a lack of drift, as all classifiers performed equally well. On the other hand, in the sea dataset, it appears that the SAMKNN algorithm outperformed other algorithms in the single model approach. The synthetic datasets do not capture any. Overall, these observations highlight the advantages of employing an ensemble approach on real datasets while also shedding light on the dominant algorithms in synthetic datasets.

Recommendations for make ensemble more robust:

- In the synthetic datasets, I couldn't observe any drift. In the Agrawal dataset, it was reasonable since all classifiers gave the same performance in the single algorithm training approach. However, in the Sea dataset, the SAMKNN algorithm outperformed other algorithms in the single algorithm training approach. Nevertheless, I didn't observe any drift change, so I can use a more sensitive drift changer if the code works as intended. I used ADWIN, and its sensitivity belonged to the delta parameter. A smaller delta can make the drift changer more sensitive [1].

- Diversifying the number of classifiers can be useful to capture different structures in the data.

5.2): Discuss your findings on the accuracy plots. What is inferred from the drops in the prequential accuracy plot?

Answer:

In data stream models, it is common to observe high accuracy in the initial batches of data as the model adapts quickly to the initial data patterns and concepts. However, as more samples arrive, the accuracy often sharply drops. This is primarily because new samples may introduce different data characteristics or concept drift, which the model hasn't yet learned or adapted to. This situation leads to a drop in the model's performance. The phenomenon of concept drift, where the underlying statistical properties of the data change over time, poses significant challenges in data stream models. When the model detects a drift, it adapts by changing its structure or parameters to better fit the new data. This adaptation can lead to fluctuations in accuracy, with noticeable increases and decreases as the model aligns itself with the new data patterns.

Moreover, when a single model is used for a prolonged period, an overall decrease in accuracy is typically observed. This decrease may occur due to cumulative minor drifts or the evolution of data patterns that the model fails to capture effectively. This decrease in accuracy over time often exhibits an oscillating pattern, which reflects the model's ongoing struggle and adaptation to cope with the changing data. The oscillation can also be a sign of model instability, especially if the fluctuations in accuracy become more severe over time. Thus, managing concept drift is a vital aspect of maintaining the performance of data stream models.
The accuracy plots of each dataset, in ensemble approach, can be found at Appendix part.

5.3): Please also include a paragraph that summarizes your findings in this assignment. What did you learn from this assignment?

Answer:

During this assignment, I gained valuable insights into the mechanism of concept drift and its implications on data stream models. I delved into the practical understanding of different algorithms designed to detect and adapt to concept drift, enhancing the performance of streaming models. In my exploration, I realized that not all datasets are equally susceptible to concept drift. Synthetic datasets, while advantageous in their simplicity and control, often lack the complexity and realism of real-world data. This makes them less likely to exhibit significant or meaningful concept drift. On the contrary, real-world datasets are usually complex, dynamic, and more prone to concept drift due to evolving circumstances and variables over time.Concept drift is an important factor to keep in mind when working with data stream models because it can significantly impact the accuracy and dependability of these models. Effectively handling concept drift ensures that the model remains relevant and accurate as new data streams in, thereby enabling reliable predictions and decisions. Furthermore, I have deepened my understanding of generating synthetic datasets using various algorithms. Generating these datasets helps in testing and benchmarking the performance of different models and algorithms, although it's crucial to complement this with real-world data testing due to the reasons previously mentioned.In the pursuit of handling concept drift, I explored new classifier algorithms such as AdaptiveRandomForestClassifier, SAMKNNClassifier, StreamingRandomPatchesClassifier, DynamicWeightedMajorityClassifier, and the HoeffdingTreeClassifier. Each of these classifiers brings a unique approach to learning from data streams and adapting to concept drift, further enriching my toolkit for handling the challenges posed by data stream modeling.In this assignment, I also learned about the sliding window accuracy evaluation method. This technique is pivotal in data stream models, as it provides a dynamic way to measure model accuracy over a recent set of samples, thereby reflecting the model's performance in the face of changing data distributions or concept drift.

References:

1. Bifet, A., Holmes, G., Pfahringer, B., Kranen, P., Kremer, H., Jansen, T., & Seidl, T. (2017). MOA: Massive Online Analysis, a Framework for Stream Classification and Clustering. Journal of Machine Learning Research, 11, 1601-1604.
Retrieved from: https://scikit-multiflow.readthedocs.io/en/stable/api/generated/skmultiflow.drift_detection.ADWIN.html

2. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., & Gavaldà, R. (2018). New ensemble methods for evolving data streams. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 139-148). Paris, France.
Retrieved from: https://scikit-multiflow.readthedocs.io/en/stable/api/generated/skmultiflow.data.SEAGenerator.html#skmultiflow.data.SEAGenerator

Appendix:

Table A):

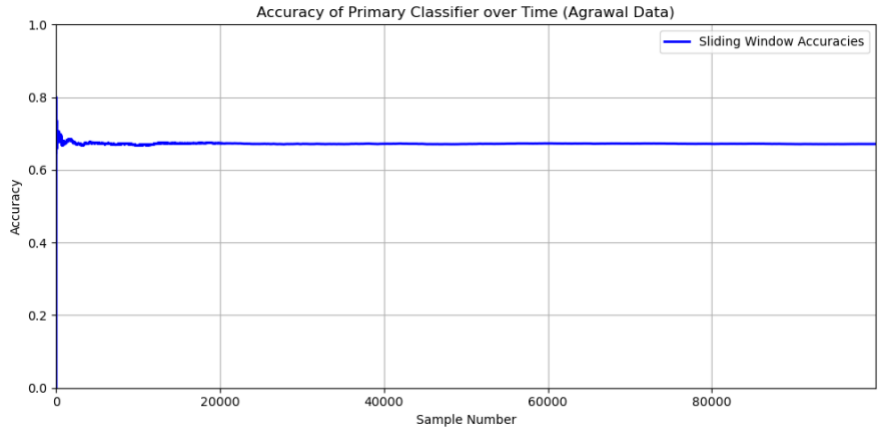

Figure 1: Accuracy plot for Agrawal Data (Ensemble approach).

TABLE B):

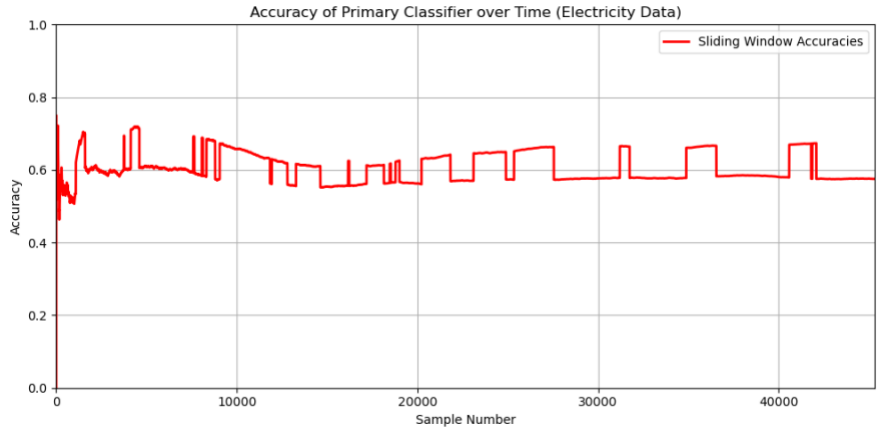

Figure 2: Accuracy plot for Electricity Data (Ensemble approach).

Table C):

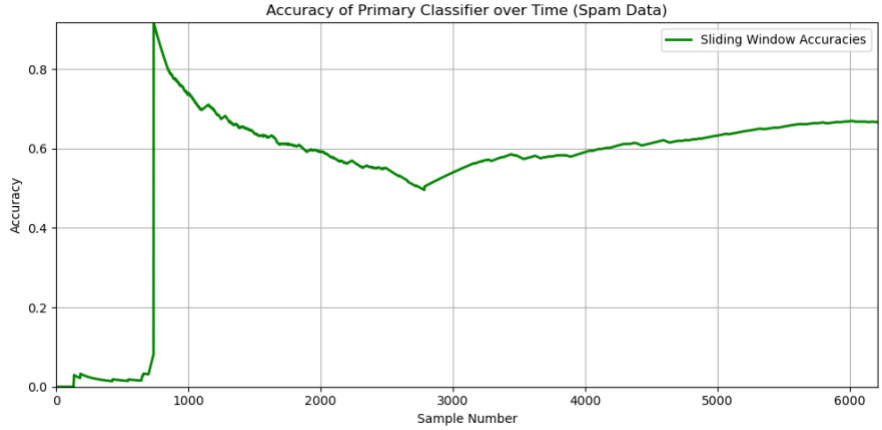

Figure 3: Accuracy plot for Spam Data (Ensemble approach).

Table D):

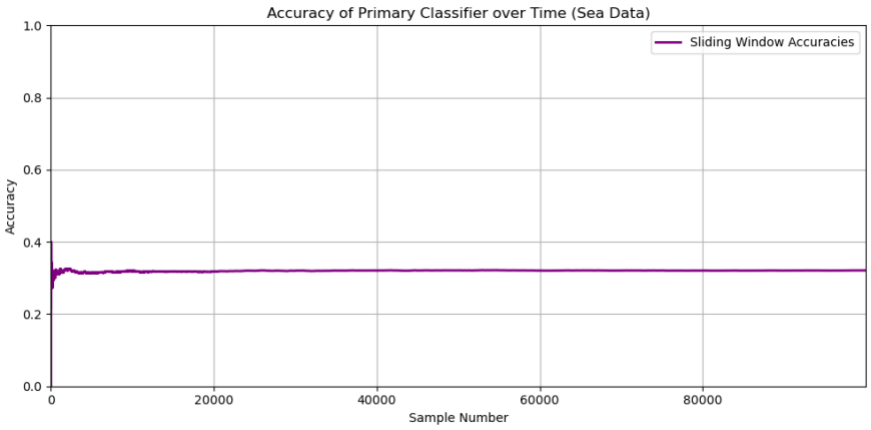

Figure 4: Accuracy Plot for Sea Data (Ensemble approach).