Adaptive Random Forest with Resampling for Imbalanced data Streams

Data Mining Lab SS2020
22.07.2020

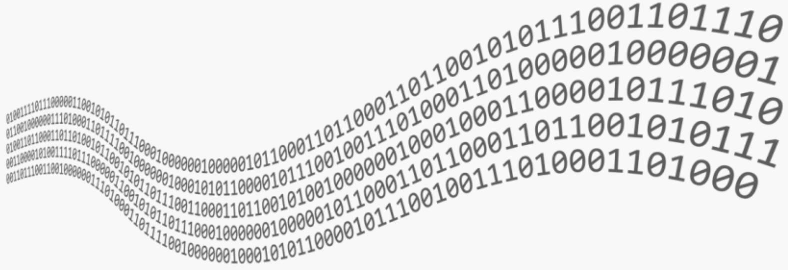
Kush Varma (kush.varma@stud.uni-hannover.de)

Outline

- Introduction
- Problem Statement
- Class Imbalance
- Solution: Adaptive Random Forest with Resampling
- ARF_{RE} Algorithm
- Moa & scikit-multiflow
- Dataset
- Performance Metrics
- Overall Results
- Conclusion

Introduction

- Large volume of continuous Real time *Streams*
- Traditional ML algorithm works in batch with fixed data set
- Traditional algorithm can not be applied for streams
- Algorithms which solves these issue
  - Online Bagging
  - Adaptive Random Forest
- Imbalance datasets
- Imbalance datasets in streams
- ARF with Resampling



Problem Statement

- Classification problem involving imbalanced data, in context to data streams
- Traditional classification algorithms focus on Representative instances, hence neglecting minority instances
- In streams, problem become more evident, reducing observation of minority instances.
- Delay in discovery of existing patterns
- Also faces concept drifts

Class Imbalance

- Imbalanced data is characterized by having more instances belonging to one class than others.
- Minority class instances rarely occurs.
- Rare, undiscovered or ignored classification.
- How to deal with class imbalance?
  - Traditionally it can be solved using:
    - Sampling
    - Ensembles
    - Cost-sensitive methods
- For streaming: Hybrid solution: ARF with Resampling: $\mathbf{ARF_{RE}}$

Solution: Adaptive Random Forest with Resampling (**ARF$_{RE}$**)

- Understanding ARF:
  - ARF simulate sampling with reposition, instead of growing each tree sequentially on different subsets of data.
  - In Online Bagging, instead of sampling with replacement, it gives weight according to Poisson ($\lambda = 1$). ARF increased λ to 6, so same instance can be used.
- In ARF$_{RE}$ combines weights to the output of Poisson Distribution, changing the chances of an instance being used for training based on current class distribution.
- $weight(S_c, Sn, \lambda) = \left(\left(100 - \frac{S_c * 100}{S_n}\right) / 100\right) * \mathrm{Poisson}(\lambda)$ [Eq. 1] where S_c is total instances from class c, S_n is total number of instances observed in stream.

ARF_{RE} Algorithm

function $\text{ARF}_{RE}(m, n, \delta_w, \delta_d, \lambda)$
  $T \leftarrow CreateTrees(n)$
  $W \leftarrow InitWeights(n)$
  $S_c \leftarrow S_n \leftarrow 0$
  while $HasNext(S)$ **do**
    $(x, y) \leftarrow next(S)$
    $S_n \leftarrow S_n + 1$
    $S_{c=y} \leftarrow S_{c=y} + 1$
    for all $t \in T$ **do**
      $\hat{y} \leftarrow predict(t, x)$
      $W(t) \leftarrow P(W(t), \hat{y}, y)$
      $k = weight(S_c, S_n, \lambda)$ ▷ Equation 1
      $TreeTrain(m, t, k, x, y)$
      if $C(\delta_w, t, x, y)$ **then**
        $b \leftarrow CreateTree()$
        $B(t) \leftarrow b$
      end if
      if $C(\delta_d, t, x, y)$ **then**
        $t \leftarrow B(t)$
      end if
    end for
    for all $b \in B$ **do**
      $k = weight(S_c, S_n, \lambda)$ ▷ Equation 1
      $TreeTrain(m, b, k, x, y)$
    end for
  end while
end function

ARF with resampling.

Symbols: *m*: maximum features evaluated per split

n: total number of trees

δ_w: warning threshold

δ_d: drift threshold

$c(\cdot)$: change detection method

S: Data stream

B: Set of background trees

W(*t*): Tree *t* weight

$P(\cdot)$: Learning performance estimation function

S_n: Current instance counter

S_c: Number of occurrences of class label c

λ: Expected value and variation of a Poisson distribution

MOA & scikit-multiflow

- **Moa: Massive Online Analysis**
  - Java based ML tool for stream
  - ML Algo: classification, regression, clustering, outlier detection, concept drift detection and recommender systems and tools for evaluation.
  - Experiments in the paper were performed on MOA with ARF_{RE} extension.
- **Scikit-multiflow**
  - Inspired by MOA
  - Inspired by scikit-learn
  - multi-output/multi-label and stream data.

Datasets

- Considering 6 data sets, 5 real and 1 synthetic.
- **AIR:** Airline dataset, task is to predict which flights are going to be delayed based on information on the scheduled departure.
- **ELEC:** Electricity dataset, task is to predict whether the electricity prices are going up or down relative to moving avg of last 24h.
- **GMSC:** Goal is to predict whether a loan should be allowed.
- **PIMA:** It is a by product of a longitudinal study of health in the Pima Indian population
- **WEATHER:** Weather dataset, which is not from the paper.
- **SEA Generator:** Data streams with three continuous features.

Performance Metrics

- Prequential evaluation(PE) with the recall
  - test-then-train approach.
  - best for estimating error on data streams.
  - Faster response for detecting drifts
- Recall is best metric to access performance when dealing with imbalanced dataset.
- Macro Avg of Recall(balanced Average), CPU time to train and update the model and g-mean.
- Testing ARF_{RE} with ARF also with state-of-the-art ensembles classifier like: Leveraging bagging, online bagging, learnNSE and Online accuracy update ensemble.

Overall Result

GMSC, ELEC, AIR

- Adaptive Random Forest with Resampling (ARF_RE)
- Adaptive Random Forest(ARF)
- Learn NSE (L NSE)
- Leveraging bagging(LB)
- Online bagging(OZA)
- Online accuracy update ensemble(OAUE)

	My findings					Paper findings				
GMSC	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-Accuracy**	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-accuracy**
ARF_RE	89.13	65	8.43	76.11471605	77.065	90.05	56.37	14.51	71.24688414	73.21
ARF	99.13	18.75	178	43.11249819	58.94	99.27	12.66	49	35.4507856	55.965
L NSE	77.92	53.731	23.05	64.70486473	65.8255	76.57	51.83	70.04	62.99700866	64.2
LB	99.23	18.75	8.61	43.13423814	58.99	99.47	10.38	16.86	32.13251624	54.925
OZA	99.67	6.25	2.77	24.95871591	52.96	99.66	7.58	5.59	27.48495588	53.62
OAUE	99.02	12.5	6.19	35.18167136	55.76	99.64	7.5	11.76	27.3367884	53.57
ELEC	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-Accuracy**	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-accuracy**
ARF_RE	94.86	89.86	4.55	92.32615881	92.36	86.81	91.44	9.54	89.09492915	89.125
ARF	95.93	90.43	64.12	93.1394111	93.18	85.41	92.13	11.08	88.70638816	88.77
L NSE	68.09	74.48	2.17	71.21336391	71.285	70.15	71.74	6.37	70.94054553	70.945
LB	93.36	90.99	3.22	92.16738252	92.175	85.67	92.74	7.08	89.1349303	89.205
OZA	87.36	88.74	0.92	88.04729638	88.05	76.7	86.64	1.91	81.51863591	81.67
OAUE	93.14	90.24	1.89	91.67853402	91.69	83.91	90.56	3.72	87.17161006	87.235
AIR	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-Accuracy**	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-accuracy**
ARF_RE	55.28	65.48	271	60.16422857	60.38	67.93	56.05	396.76	61.70475265	61.99
ARF	65.33	61.93	2495	63.60728653	63.63	71.34	51.88	544.46	60.83682437	61.61
L NSE	56.49	59.1	746	57.7802648	57.795	67.94	55.4	985.42	61.35043602	61.67
LB	59.44	58.39	311	58.91266078	58.915	74.73	48.72	502.04	60.33941995	61.725
OZA	74	48.69	47.19	60.02549458	61.345	84.09	41.07	67.95	58.76713622	62.58
OAUE	67.41	62.17	192.3	64.7370041	64.79	81.37	50.27	295.78	63.9567815	65.82

Overall Result

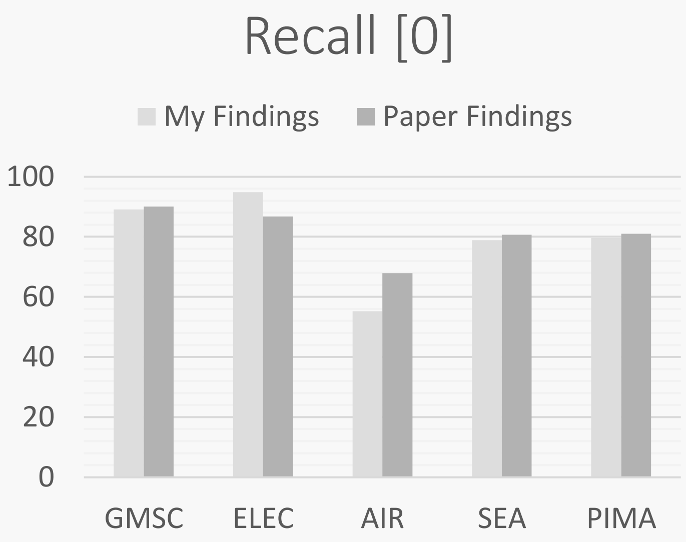
SEA, PIMA

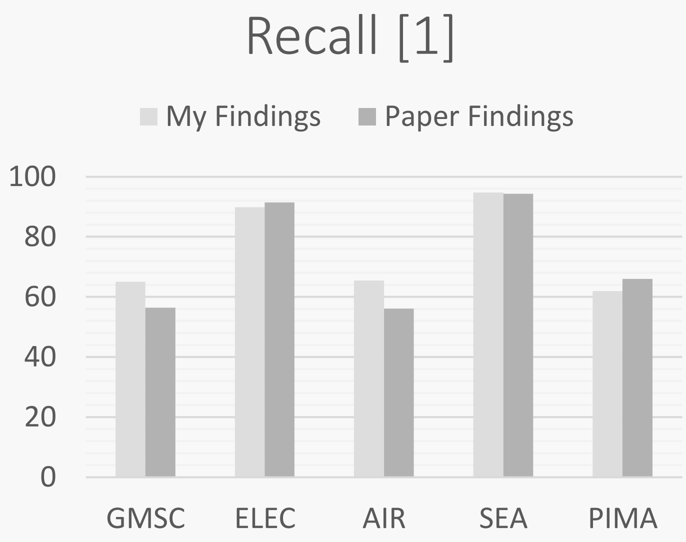
- Adaptive Random Forest with Resampling (ARF_RE)
- Adaptive Random Forest(ARF)
- Learn NSE (L NSE)
- Leveraging bagging(LB)
- Online bagging(OZA)
- Online accuracy update ensemble(OAUE)

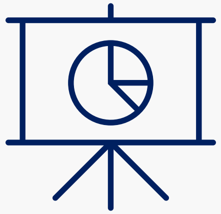
	My findings					Paper findings				
SEA	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-Accuracy**	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-accuracy**
ARF_RE	78.9	94.8	108.23	86.48537449	86.85	80.68	94.34	173.55	87.24305818	87.51
ARF	OOM	OOM		-		80.28	94.63	246.47	87.16017669	87.455
L NSE	71.38	93.27	588	81.59419465	82.325	72.91	93.22	1260.47	82.44192016	83.065
LB	80.34	95.56	39.41	87.62014837	87.95	80.61	94.93	74.84	87.47746738	87.77
OZA	80.63	95.56	13.25	87.77814534	88.095	80.36	94.8	21.8	87.28188816	87.58
OAUE	80.05	95.25	26.7	87.31988605	87.65	80.57	94.85	46.65	87.41890242	87.71
PIMA	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-Accuracy**	**recall [0]**	**recall[1]**	**CPU seconds**	**g-mean**	**Balanced-accuracy**
ARF_RE	79.6	61.94	0.078	70.216978	70.77	81	66.04	0.34	73.13849875	73.52
ARF	87.2	49.25	1.09	65.53319769	68.225	87.6	54.48	1.18	69.08290671	71.04
L NSE	87	36.56	0	56.3978723	61.78	87	36.57	0.08	56.40558483	61.785
LB	83.6	53.73	0.03	67.02110115	68.665	83.6	53.73	0.2	67.02110115	68.665
OZA	78.2	59.32	0.015	68.10891278	68.76	78.2	59.33	0.14	68.11465334	68.765
OAUE	94.8	19.4	0	42.8849624	57.1	94.8	19.4	0.05	42.8849624	57.1

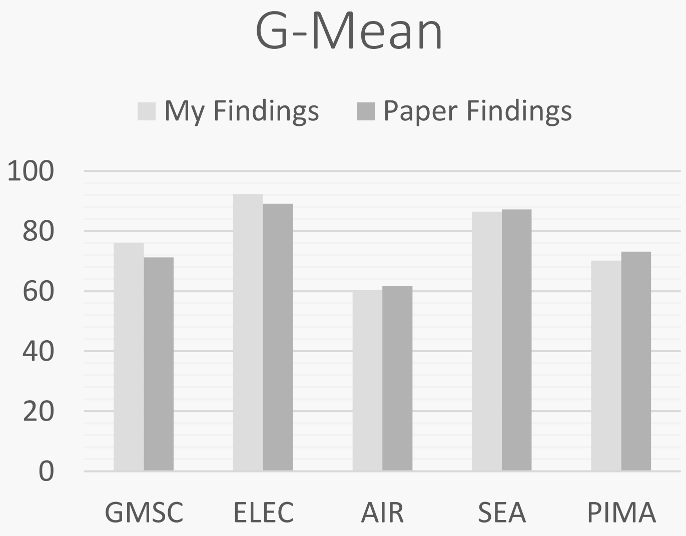
Overall Result

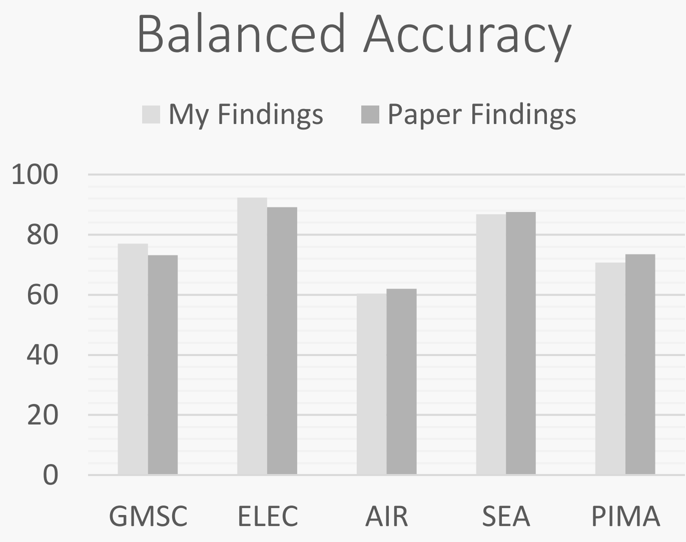
comparing ARF_{RE} FOR GMSC, ELEC, AIR, SEA, PIMA











Overall Result
WEATHER

- Adaptive Random Forest with Resampling (ARF_RE)
- Adaptive Random Forest(ARF)
- Learn NSE (L NSE)
- Leveraging bagging(LB)
- Online bagging(OZA)
- Online accuracy update ensemble(OAUE)

My findings

WEATHER	recall [0]	recall[1]	CPU seconds	g-mean	Balanced-Accuracy
ARF_RE	56.46	85.46	1.65	69.46273533	70.96
ARF	81.54	8.3	39.98	26.01503411	44.92
L NSE	65.93	68.57	0.5	67.23704411	67.25
LB	73.34	72.95	1.29	73.14474007	73.145
OZA	70.5	65.84	0.37	68.13016953	68.17
OAUE	74.13	71.584	0.718	72.84587785	72.857

Overall Result

Different λ

λ=7

ARF_RE	Recall 0	1	CPU Seconds	g-mean	balanced-accuracy
GMSC	86.3	63.74	11.84	74.167122	75.02
ELEC	95.5	89.11	6.28	92.249688	92.305
AIR	57.01	61.7	281.56	59.308659	59.355
SEA	80.63	94.64	121.89	87.354583	87.635
PIMA	78.2	64.18	0.12	70.844026	71.19
WEATHER	54.73	84.97	2.39	68.193901	69.85

λ=8

ARF_RE	Recall 0	1	CPU Seconds	g-mean	balanced-accuracy
GMSC	84.78	71.25	18.09	77.721136	78.015
ELEC	96.35	88.55	6.78	92.367703	92.45
AIR	55.11	60.75	301.12	57.861321	57.93
SEA	80.63	94.64	134.59	87.354583	87.635
PIMA	81.39	58.2	0.14	68.825126	69.795
WEATHER	58.99	81.96	2.61	69.532873	70.475

λ=9

ARF_RE	Recall 0	1	CPU Seconds	g-mean	balanced-accuracy
GMSC	86.08	65	21.19	74.80107	75.54
ELEC	95.93	89.11	7.59	92.457138	92.52
AIR	56.15	60.99	318.12	58.519984	58.57
SEA	80.05	94.49	140.56	86.970826	87.27
PIMA	80.8	60.07	0.14	69.668185	70.435
WEATHER	57.57	83.87	2.97	69.48666	70.72

λ=10

ARF_RE	Recall 0	1	CPU Seconds	g-mean	balanced-accuracy
GMSC	86.63	62.5	21.92	73.582437	74.565
ELEC	95.71	90.43	7.97	93.03255	93.07
AIR	54.41	61.22	327.59	57.714645	57.815
SEA	79.76	95.1	153.05	87.092916	87.43
PIMA	81.8	59.32	0.19	69.658998	70.56
WEATHER	59.77	81.69	3.17	69.875685	70.73

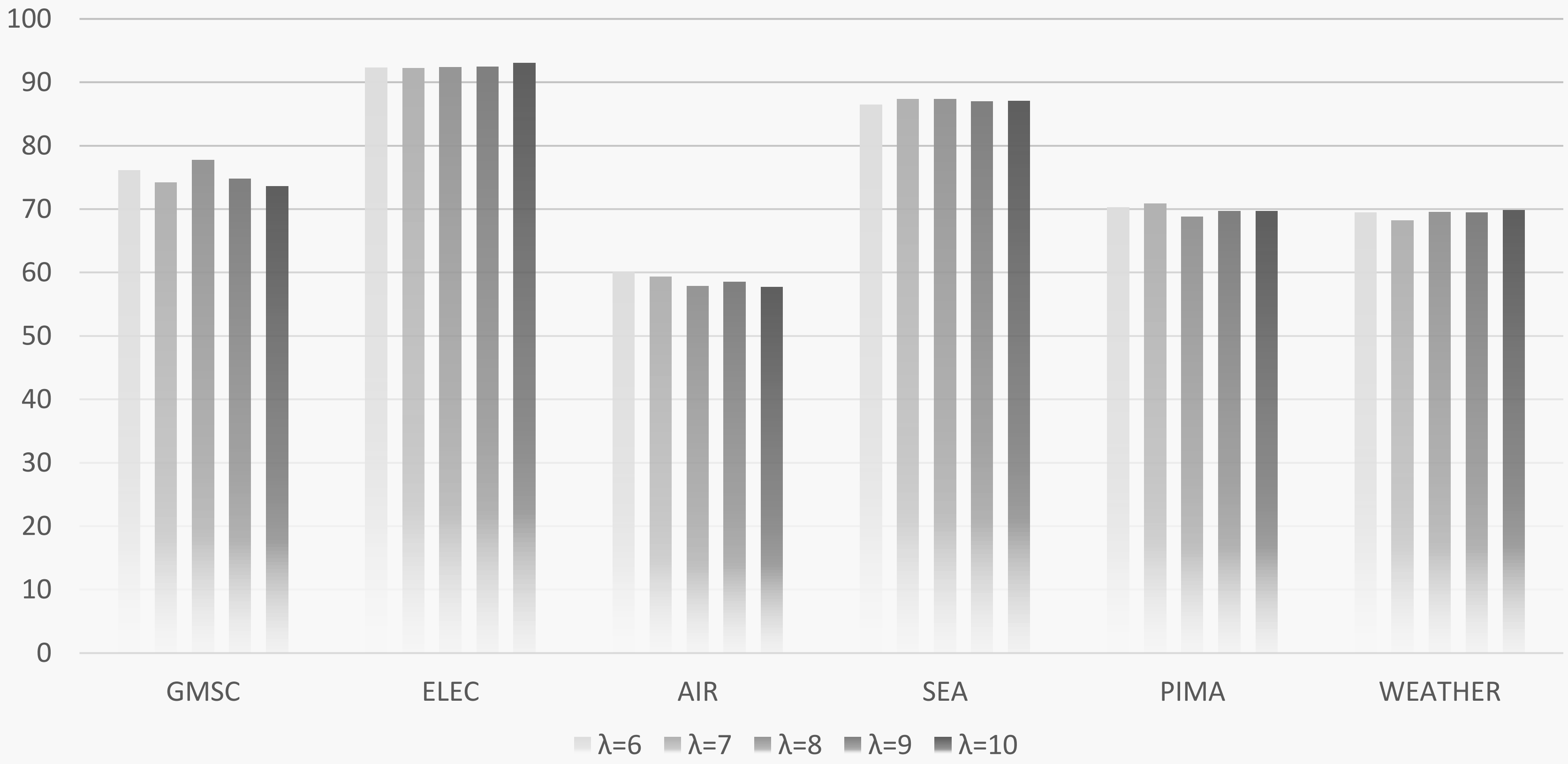
Overall Result

comparing g-mean and balanced-accuracy among different λ

	GMSC		ELEC		AIR		SEA		PIMA		WEATHER	
	G-MEAN	Balanced-Accuracy	G-MEAN	Balanced-Accuracy	G-MEAN	Balanced-Accuracy	G-MEAN	Balanced-Accuracy	G-MEAN	Balanced-Accuracy	G-MEAN	Balanced-Accuracy
λ=6	76.11471605	77.065	92.32616	92.36	60.16423	60.38	86.48537	86.85	70.21698	70.77	69.46274	70.96
λ=7	74.1671221	75.02	92.24969	92.305	59.30866	59.355	87.35458	87.635	70.84403	71.19	68.1939	69.85
λ=8	77.72113612	78.015	92.3677	92.45	57.86132	57.93	87.35458	87.635	68.82513	69.795	69.53287	70.475
λ=9	74.80106951	75.54	92.45714	92.52	58.51998	58.57	86.97083	87.27	69.66818	70.435	69.48666	70.72
λ=10	73.58243676	74.565	93.03255	93.07	57.71464	57.815	87.09292	87.43	69.659	70.56	69.87568	70.73

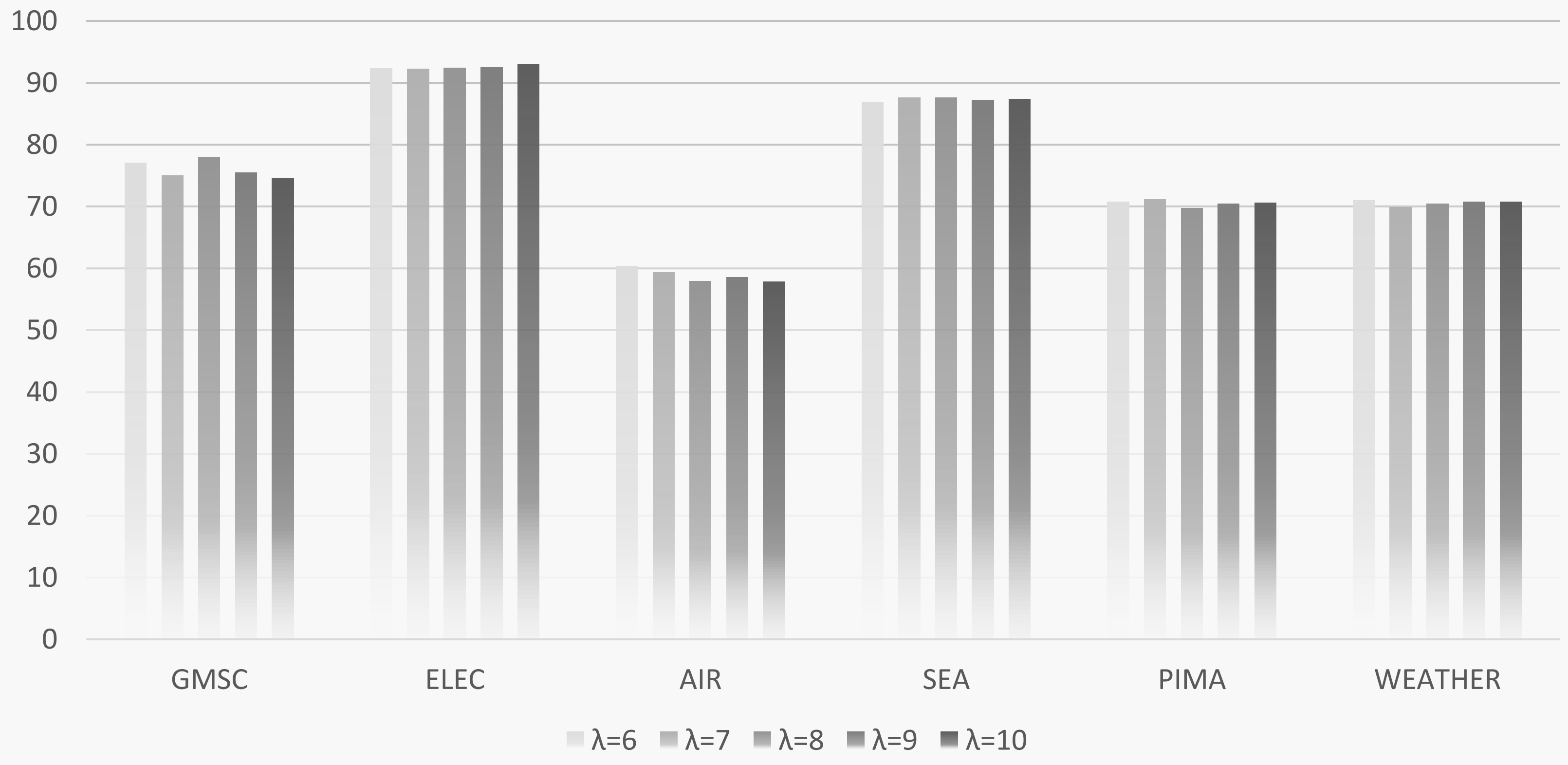
Overall Result

comparing g-mean among different λ



Overall Result

comparing balanced accuracy among different λ



Conclusion

- Ensemble classifier with resampling
- ARF_{RE} inserts weights, changes the λ.
- Less probability for Majority class and high for minority of being presented to ARF tree.
- It not only improved the overall performance as compared to ARF but also with better computational cost.
- With different λ, performance improves from 6 to 8 but then decreases. On the average, there is no major difference.
- Able to reproduce the results from the paper, also further extending to see the role of λ in overall result.
- I also worked on scikit-multiflow python porting, but it is still in pending state. Need further work.

Thank You