Federated Forest

方沛

Backgrounds: The Joint Model

- The data has been the largest bottleneck for the implementation of AI methods
- In real world, data are scattered across different companies or organizations and stored in the form of data islands, data across different domains cannot be shared with each other
- The GDPR provides individuals with more control over their personal data and states strict principles and absolute transparencies on how businesses should handle these data

How to train joint model: Federated Learning

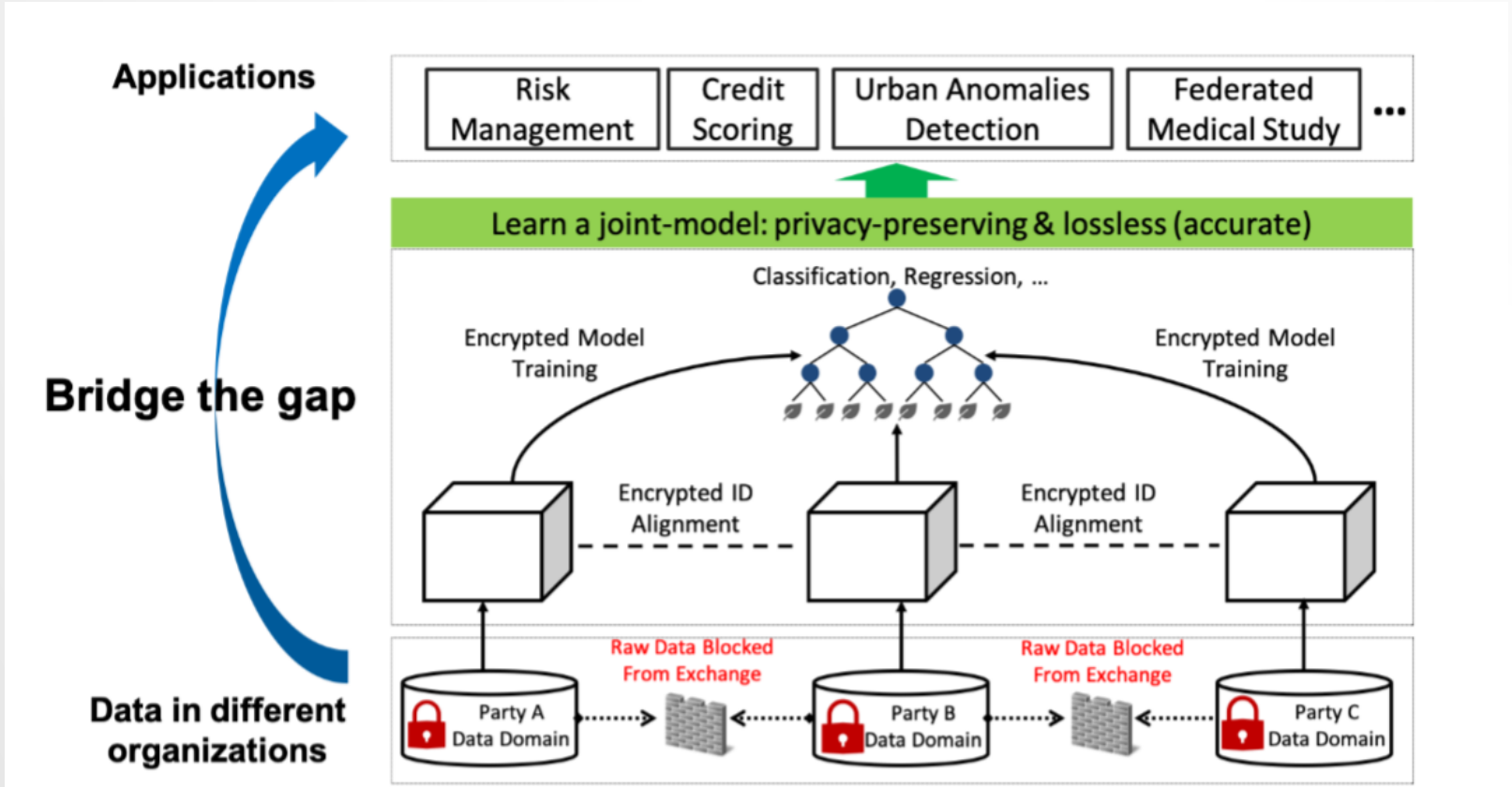


Figure 1: New Era of Machine Learning

Related Works:

- Federated Meta-Learning
- Multitask and Federated AdaBoost
- Vertical and Horizontal Federated Learning
- Federated Logisitic Regression
- Modular Benchmarking Framework
- Transfer Learning and Reinforcement Learning
- Differential Privacy and Homomorphic Encryption

Four Contributions of Federated Forest

1. **Secured Privacy**
   Limit the information exchange to the largest extent.
2. **Loseless**
   Its accuracy is same as Classical FF, using CART and bagging
3. **Efficient**
   MPI for communication, no restriction on depth, sample size.
4. **Practicability and scalability**
   Suitable for both classification and regression.
   Strong in reality.

Problem Formulation

$$The\ model\ is\ assumed\ a\ vertical\ federated\ learning.$$

$$There\ are\ M\ nodes$$

$$To\ each\ node:$$

$$The\ data\ dormain\ is\ D_i$$

$$To\ all\ models:$$

$$Tha\ data\ dormain\ is\ D = D_1 \bigcup D_2 \bigcup ... \bigcup D_m$$

$$where\ 1 \leq i \leq M$$

Problem Formulation

$$Denote\ the\ feature\ space\ of\ D_i\ as\ F_i:$$

$$As\ above: F = F_1 \bigcup F_2 \bigcup ... \bigcup F_M$$

$$where\ 1 \leq i \leq M$$

All features' true name are encoded in order to protect privacy

$$As\ assmuption,\ for\ 1 \leq i, j \leq M:$$

$$if\ i \neq j\ then\ F_i \bigcap F_j = \emptyset$$

In the work, sample nums are same and IDs are in accord

Notation

1. In reality, node num M is always quite small(<4)
2. Forget about the alignment of IDs

Problem Statement

Given: Regional domain D_i and encrypted label y on each client $i, 1 \leq i \leq M$.

Learn: A Federated Forest, such that for each tree in the forest:

1. a complete tree model T is held on master;
2. a partial tree model T_i is stored on each client $i, 1 \leq i \leq M$.

Constraint: The performance (accuracy, f1-score, MSE, e.t.c.) of the Federated Forest must be comparable to the non-federated random forest.

Alogorithm 1 : Client

```
while 还有树要建立:
    if (node收到 Fi,Di):
    Function TreeBuild(){
        建立一个空的树节点
        if 满足剪枝条件 :
            设置为叶节点,bagging
        '''设置要记录的初始量'''
        purify,f_plus = -∞ , None
        '''如果master随机取到的fetures在该节点内还有没被作为分割点'''
        if Fi != None:
            '''计算得到最好的划分点'''
            best_purity = MAX(impurity_improvements)
            f_plus=getNodeOf(best_purity)
        把 best_purity发给master
        '''master告诉该节点这是全局最优'''
        if 从master接收到的split_message == 'Success':
            '''节点正式划分'''
            is_selected=True
            划分样本
            把 left_tree, right_tree发给master
        else:
        '''说明最优划分节点不是这课树'''
            收到left_tree,right_tree各有哪些节点
        '''递归向下建树'''
        left_tree=TreeBuild(left_tree)
        right_tree=TreeBuild(right_tree)
        return node
    Forests.append(Tree)
}
```

Alogorithm 2 : Master

```
while 还需要建树 :
    随机取D,F
    对每个节点,把该节点的Fi发给该节点
    Fuction TreeBuild(D,F,y):
        if 满足剪枝条件:
            设置为叶节点,voting得出label

        接受来自各个节点的增益值,node_best_purities
        global_best_purity=MAX(node_best_purities)
        '''确定最优划分特征的来源节点'''
        selected_node=getNodeOf(global_best_purity)

        for node in nodes:
            if node is selected_node:
                发split_message表示它被选中
            else:
                把划分结果left_tree,right_tree告知node

    返回node根节点
    Forest.append(Tree)
```

Alogorithm 3 : Client(Predict)

```
while 需要预测:
    TreePrediction(Ti,Di_test,Fi):
        '''Si是要预测的一个样本,in leaf表示预测完成'''
        if Si is in leaf:
            return (Si,Labeli)
        else:
            if 节点保存了划分信息(最优节点) :
                划分
                left_tree=TreePrediction(Ti_left,Di_test_left,Fi)
                right_tree=TreePrediction(Ti_right,Di_test_right,Fi)
            else:
                left_tree=TreePrediction(Ti_left,Di_test_left,Fi)
                right_tree=TreePrediction(Ti_right,Di_test_right,Fi)
            return (left_tree,right_tree)

        把 (S1,S2,...,Sm) 送到master
```

根据论文,如果一个样本遇到有划分标准的节点,则划分,否则同时全部进入左右两个节点

对于决策树 Ti 的每个叶节点,都会有一批样本,对第l个叶节点里的样本,记作 S_i^l, $(l \in L, L是T_i的叶节点集合)$

对 $\{S_i^l\}_{i=1}^M$ 取交集运算

每个节点的森林的结构是一样的,因此所有叶节点的位置和数目一一对应,可以做交集运算

虽然论文 appendix 的证明正规冗长,理解起来是不难的:
可以理解为:

$$S \bigcap A \bigcap B \bigcap C = (S \bigcap A \bigcap B) \bigcap (S \bigcap C)$$

S是样本全集. A,B,C是分别满足条件a,b,c的样本集合

$Algorithm\ 4 : Master(Predict)$

```
while 需要预测:
    Gather{S1,S2,...Sm}
    Obtain(S^1,S^2,...S^m)
Return all the label
```

Privacy Protection

1. Identities
   - Hash & MD5
2. Label
   - Encode or Homomorphic Encryption Feature
3. Communication
   - Encode on feature name
4. Model Storage
   - In the procedure

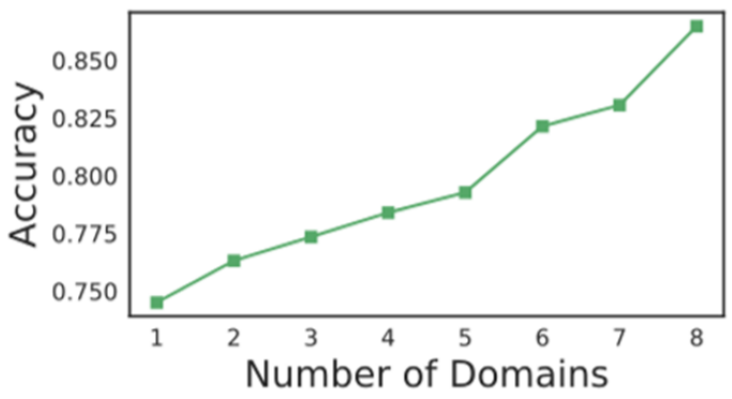
Experiment(2 Parties)

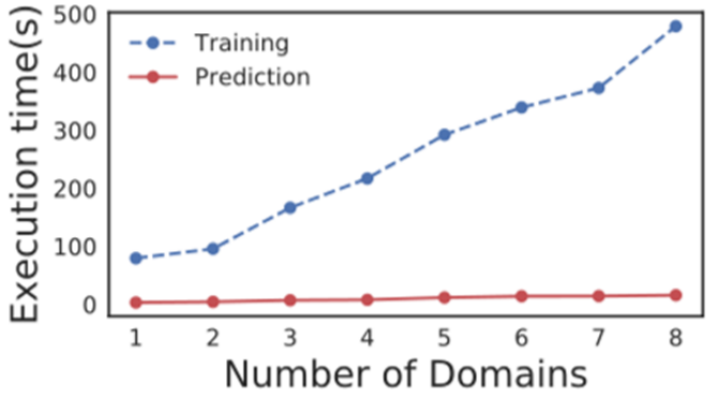
Table 1: Classification and regression experiments

Classification	RF1	RF2	F-LR	NonFF	FF	p-value
target marketing	0.870	0.848	0.862	-	0.890 ± 0.014	-
ionosphere	0.864	0.828	0.873	0.908 ± 0.019	0.896 ± 0.030	**0.211**
spambase	0.844	0.831	0.873	0.943 ± 0.005	0.928 ± 0.020	**0.065**
parkinson [27]	0.849	0.849	0.829	0.859 ± 0.018	0.857 ± 0.013	**0.744**
kdd cup 99	0.974	0.965	-	0.995 ± 0.001	0.995 ± 0.009	0.012
waveform	0.745	0.743	-	0.826 ± 0.008	0.822 ± 0.012	0.029
gene	0.975	0.975	-	0.988 ± 0.005	0.982 ± 0.006	**0.229**
Regression	RF1	RF2	F-LR	NonFF	FF	p-value
year prediction	10.47	10.72	9.56	9.537 ± 0.003	9.555 ± 0.061	**0.058**
Superconduct [13]	19.74	17.49	17.52	15.369 ± 0.118	15.411 ± 0.163	**0.186**

Experiment(multi Parties)

- The training execution time was almost linearly with respect to to the number of domains, which is to be expected because all features are be examined in tree building.





Experiment(efficiency)

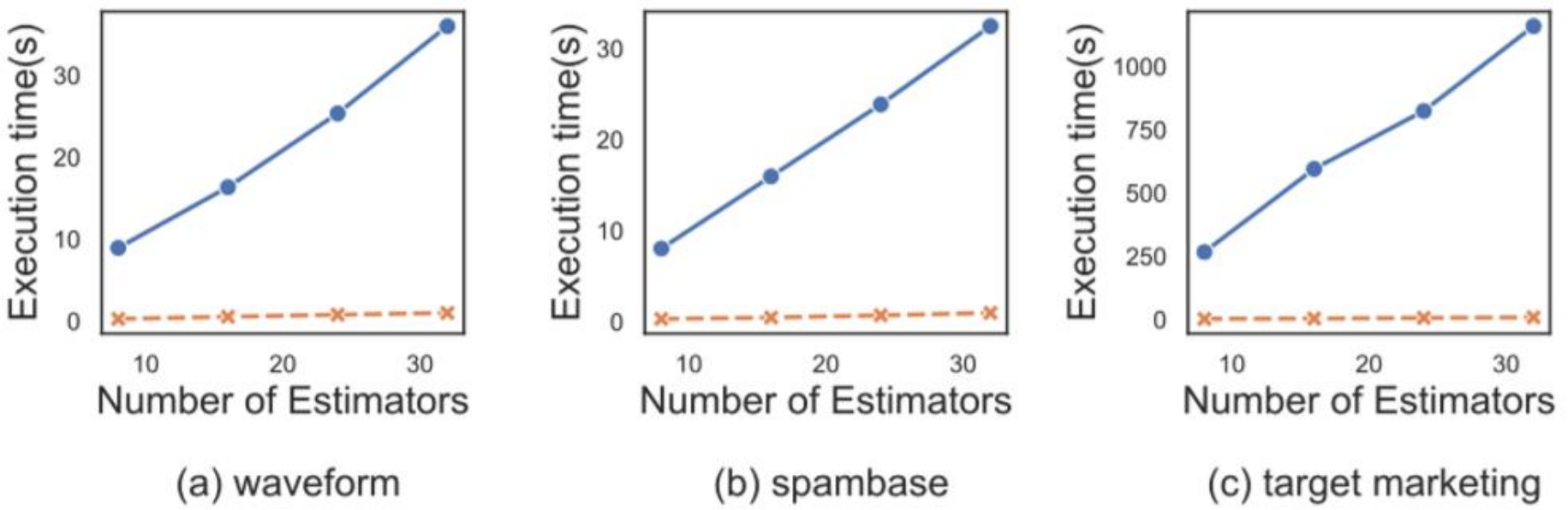


Figure 4: Prediction Time vs. Number of Estimators

Experiment(efficiency)

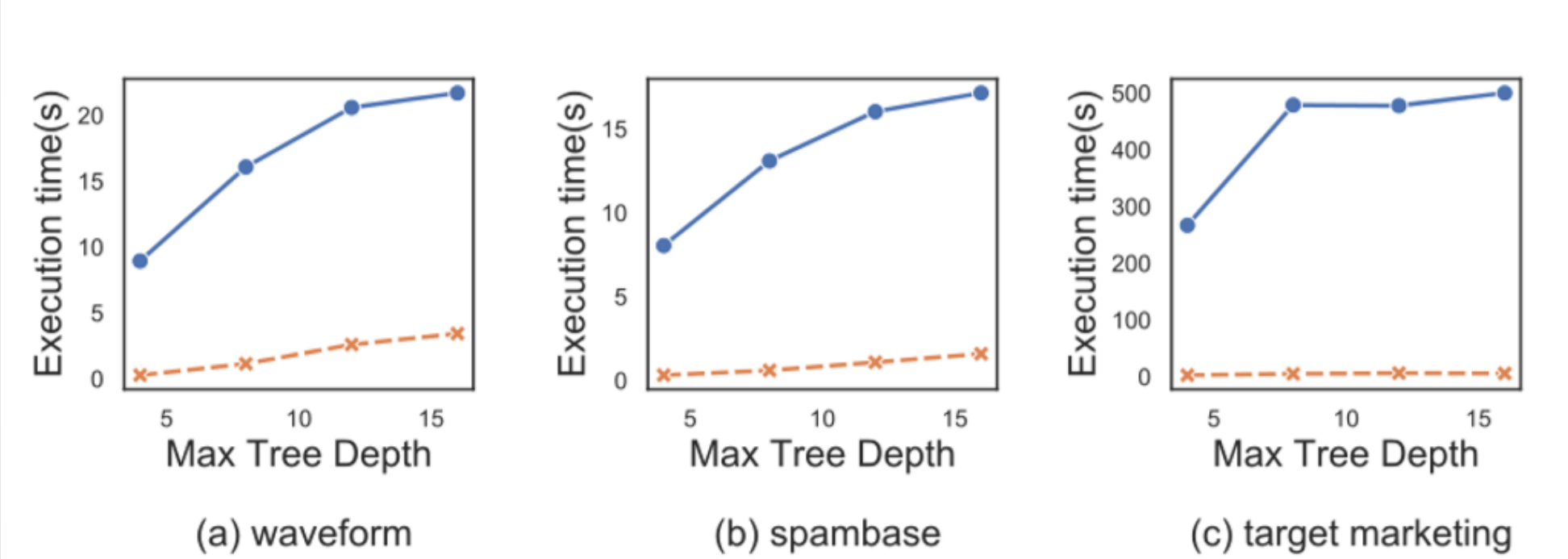


Figure 5: Prediction Time vs. Max Depth

Experiment(efficiency)

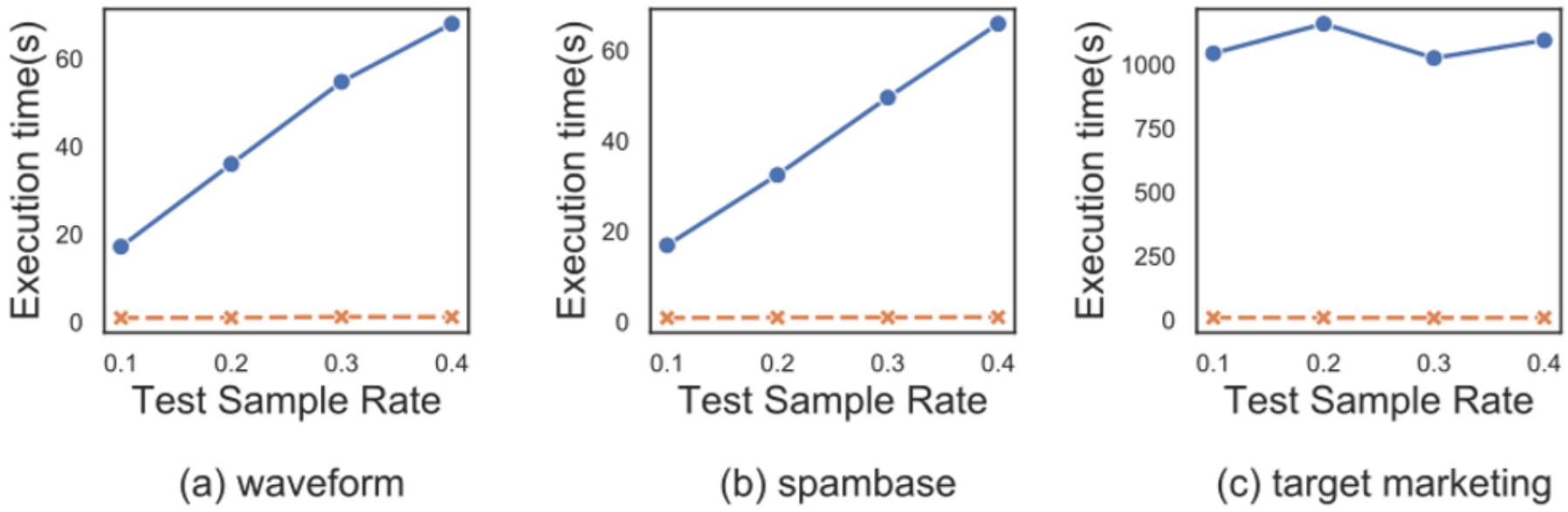


Figure 6: Prediction Time vs. Test Sample Size

My Experiments

Accuracy vs Integrations

	1000	2000	4000	8000	16000	32000
1	0.796	0.603	0.574	/	/	/
5	0.75	0.706	0.700	0.713	/	/
10	0.708	0.741	0.722	0.705	/	/
20	0.83	0.822	0.795	0.816	0.74	/
40	0.904	0.895	0.828	0.818	0.802	0.764

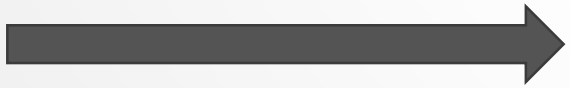
Time cost vs Integrations

	1000	2000	4000	8000	16000	32000
1	3.749	16.783	60.840	/	/	/
5	0.978	3.313	14.135	55.071	/	/
10	0.611	2.010	6.300	28.605	/	/
20	0.301	1.239	3.913	14.256	62.238	/
40	0.318	0.832	2.457	8.124	28.665	118.418

POTENTIAL IMPROVEMENT

- More powerful decision tree like that in sk-learn
- Back & Pre pruning
- Features with more thresholds
- Balanced Data
- Find the best parameters for a single tree

Present
Good:80% 85% 90%
Bad: 65% 50% 35%



Ideal
Good: 80% 85%
Bad: 75% 60%

If we can frequently reach the ideal accuracy with fewer trees, we can reach the accuracy both high in two classes.