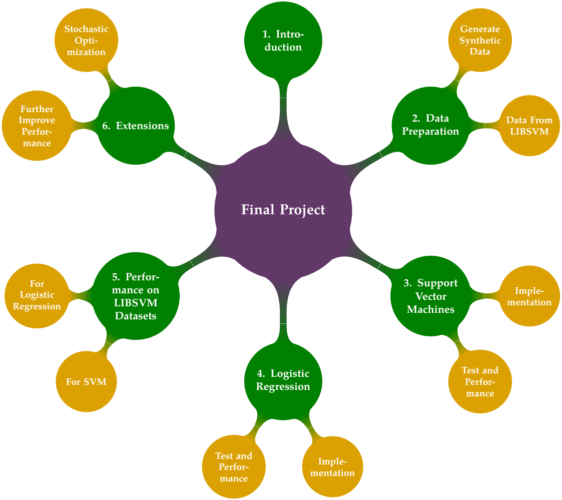
MDS 6106: Introduction to Optimization

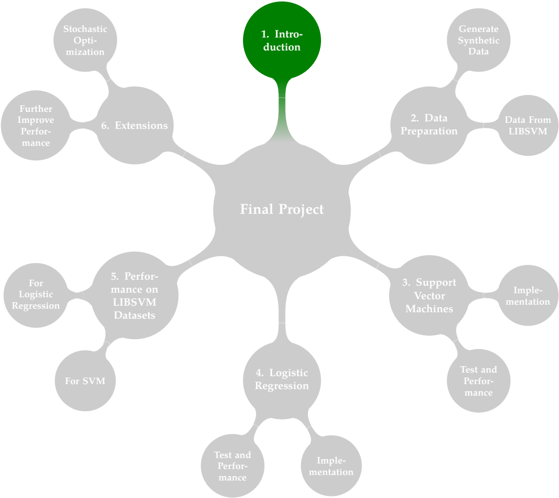
Final Project

Presentation *December 30th*

Content

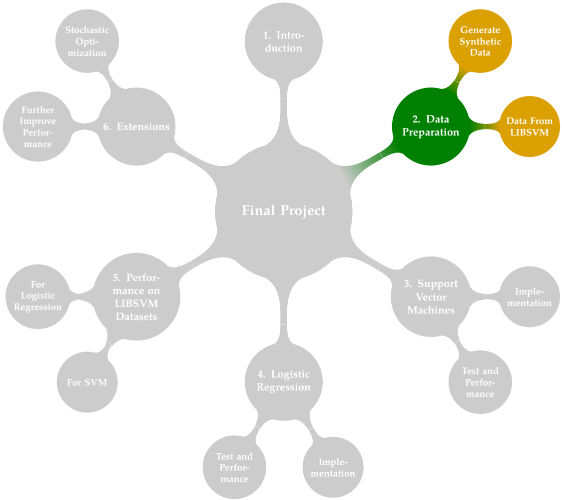


Final Project
1. Intro-duction
2. Data Preparation
Generate Synthetic Data
Data From LIBSVM
3. Support Vector Machines
Imple-mentation
Test and Perfor-mance
4. Logistic Regression
Test and Perfor-mance
Imple-mentation
5. Perfor-mance on LIBSVM Datasets
For Logistic Regression
For SVM
6. Extensions
Stochastic Opti-mization
Further Improve Perfor-mance


1. Introduction
Final Project
2. Data Preparation
Generate Synthetic Data
Data From LIBSVM
3. Support Vector Machines
Implementation
Test and Performance
4. Logistic Regression
Test and Performance
Implementation
5. Performance on LIBSVM Datasets
For Logistic Regression
For SVM
6. Extensions
Stochastic Optimization
Further Improve Performance

Background

- Binary Classification Problem: Assume that our training data consists of feature vectors $a_i \in \mathbb{R}^n, i \in \{1, 2, \ldots, m\}$ and corresponding class labels $b_i \in \{-1, 1\}$.
- Goal: Utilize minimization methodologies for solving support vector machine and logistic regression binary classification models


Final Project
1. Intro-duction
2. Data Preparation
Generate Synthetic Data
Data From LIBSVM
3. Support Vector Machines
Imple-mentation
Test and Perfor-mance
4. Logistic Regression
Test and Perfor-mance
Imple-mentation
5. Perfor-mance on LIBSVM Datasets
For Logistic Regression
For SVM
6. Extensions
Stochastic Opti-mization
Further Improve Perfor-mance

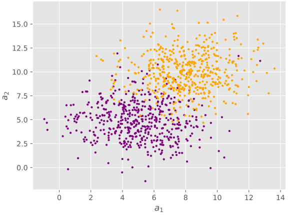
Generate Synthetic Data:

- Formula:

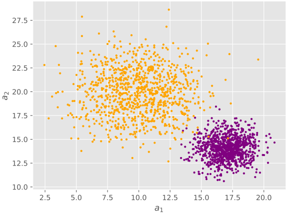
$$a_i = c_1 + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \end{pmatrix}, \quad a_j = c_2 + \begin{pmatrix} \delta_1 \\ \delta_2 \end{pmatrix}$$

  where $i = 1, 2, 3, ..., m_1, j = 1, 2, 3, ..., m_2$, $\varepsilon_1, \varepsilon_2 \sim N\left(0, \sigma_1^2\right)$, $\delta_1, \delta_2 \sim N\left(0, \sigma_2^2\right)$

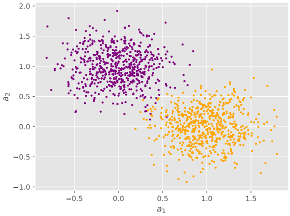
- Parameters:
  - Dataset1: $c_1 = (5, 5)$, $c_2 = (8, 10)$, $\sigma_1 = 2$ $\sigma_2 = 2$ and $m_1 = m_2 = 500$
  - Dataset2: $c_1 = (17, 14)$, $c_2 = (10, 20)$, $\sigma_1 = 1.2$ $\sigma_2 = 2.5$ and $m_1 = m_2 = 1000$
  - Dataset3: $c_1 = (0, 1)$, $c_2 = (1, 0)$, $\sigma_1 = 0.3$ $\sigma_2 = 0.3$ and $m_1 = m_2 = 600$
  - Dataset4: $c_1 = (10, 15)$, $c_2 = (15, 10)$, $\sigma_1 = 3$ $\sigma_2 = 2$ and $m_1 = m_2 = 1200$
- Result: Four datasets generated by different parameters.



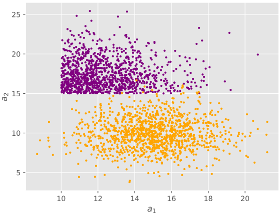
(a) Dataset1



(b) Dataset2



(c) Dataset3



(d) Dataset4

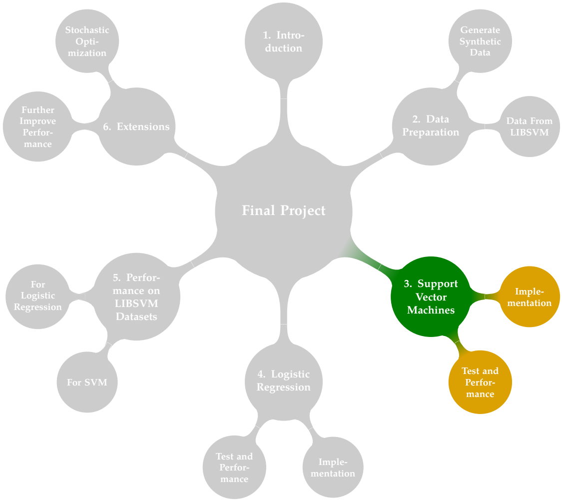
Data From LIBSVM

Table below are the description of the datasets from LIBSVM we used in later numerical comparison.

data set	m	n	data set	m	n
a9a	32561	122	mushrooms	8124	112
breast-cancer	683	10	news20	19996	1355191
covtype	581012	54	phishing	11055	68
gisette	6000	5000	rcv1	20242	47236

Data Processing

- Train Test Split: from sklearn.model_selection import train_test_split
- Normalization: from sklearn.processing import MaxAbsScalar


Final Project
1. Intro-duction
2. Data Preparation
Generate Synthetic Data
Data From LIBSVM
3. Support Vector Machines
Imple-mentation
Test and Perfor-mance
4. Logistic Regression
Test and Perfor-mance
Imple-mentation
5. Perfor-mance on LIBSVM Datasets
For Logistic Regression
For SVM
6. Extensions
Stochastic Opti-mization
Further Improve Perfor-mance

Methodology: Consider a smooth variant of the support vector machine:

$$\min_{x,y} f_{\mathrm{svm}}(x,y) := \frac{\lambda}{2}\|x\|^2 + \sum_{i=1}^{m} \varphi_+\left(1 - b_i\left(a_i^\top x + y\right)\right)$$

Here, $\varphi_+(t)$ denotes a Huber-type version of the max-function $\max\{0, t\}$:

$$\varphi_+(t) = \begin{cases} \frac{1}{2\delta}(\max\{0,t\})^2 & \text{if } t \le \delta \\ t - \frac{\delta}{2} & \text{if } t > \delta \end{cases}$$

Implementation

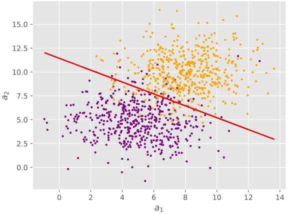
- GM: With backtracking ($\gamma = 0.1, \sigma = 0.5$, and $s = 1$)
- AGM: Follow the basic extrapolation strategy

$$\alpha_k = \frac{1}{L}, \quad \beta_k = \frac{t_{k-1} - 1}{t_k},$$
$$t_k = \frac{1}{2}\left(1 + \sqrt{1 + 4t_{k-1}^2}\right), \quad t_{-1} = t_0 = 1$$

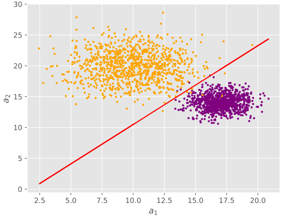
  and use the adaptive version of Lipschitz constant.
- BFGS:
  - With backtracking($\gamma = 0.1, \sigma = 0.5$, and $s = 1$) and $H_0 = I$ as initial matrix.
  - Use $\left(s^k\right)^\top y^k > 10^{-14}$ to guarantee positive definiteness.

Performance on Synthetic Datasets

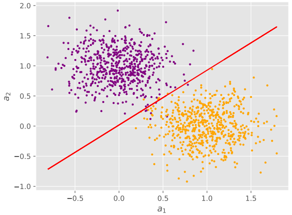
Seperate Line



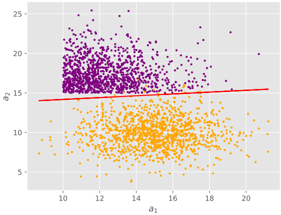
(a) Dataset1



(b) Dataset2

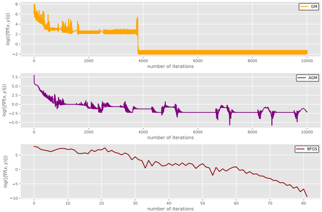


(c) Dataset3

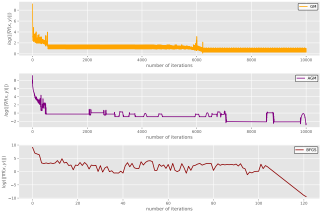


(d) Dataset4

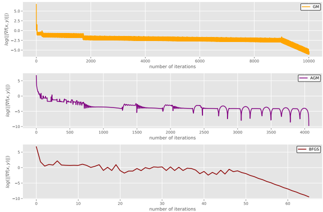
Convergence of $\log\left(\|\nabla f(x,y)\|\right)$



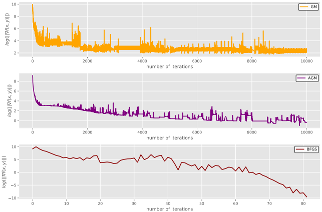
(a) Performance for Dataset1



(b) Performance for Dataset2



(c) Performance for Dataset3



(d) Performance for Dataset4

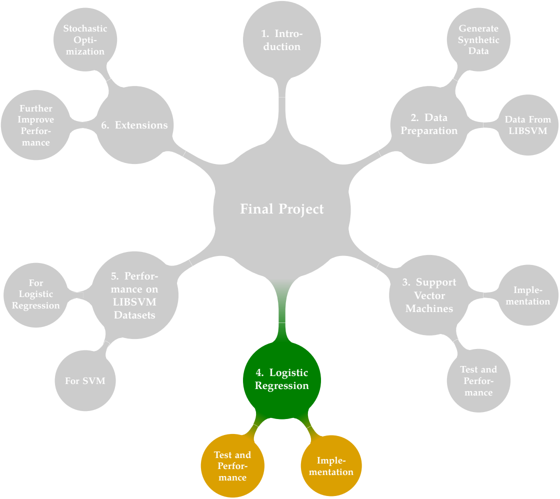
Performance on Synthetic Datasets

More performance index

datasets	methods	accuracy	iter	cpu-time	datasets	methods	accuracy	iter	cpu-time
dataset 1	GM	0.922	10000*[1]	329.54s	dataset 3	GM	0.992	10000*	271.28s
	AGM	0.923	10000*	20.11s		AGM	0.992	4061	10.23s
	BFGS	0.923	74	3.08s		BFGS	0.992	65	2.22s
dataset 2	GM	0.996	10000*	749.91s	dataset 4	GM	0.99	10000*	778.71s
	AGM	0.997	10000*	39.09s		AGM	0.995	10000*	46.21s
	BFGS	0.997	95	9.15s		BFGS	0.995	80	6.55s

[1] * means the max iteration times reached

Logistic Regression



Logistic Regression

Methodology: Consider logistic regression model, the corresponding optimization problem is given by:

$$\min_{x,y} f_{\log}(x, y) = \frac{1}{m} \sum_{i=1}^{m} \log\left(1 + \exp\left(-b_i \cdot \left(a_i^\top x + y\right)\right)\right) + \frac{\lambda}{2}\|x\|^2$$

- The sigmoid function: $\sigma : \mathbb{R} \to \mathbb{R}, \sigma(a) = \frac{1}{1+\exp(-a)}$
- The linear model: $\ell_{(x,y)}(a) = a^\top x + y$

$$\sigma\left(\ell_{(x,y)}\left(a_i\right)\right) \approx \begin{cases} 1 & \text{if } a_i \text{ belongs to class } \mathcal{C}_1, \text{ i.e., } b_i = +1 \\ 0 & \text{if } a_i \text{ belongs to class } \mathcal{C}_2, \text{ i.e., } b_i = -1 \end{cases}$$

A new data point $a \in \mathbb{R}^n$ can then be classified via

$$\begin{cases} +1 & \text{if } \sigma\left(\ell_{(x,y)}(a)\right) > \frac{1}{2} \\ -1 & \text{if } \sigma\left(\ell_{(x,y)}(a)\right) \leq \frac{1}{2} \end{cases} \quad \text{or} \quad \begin{cases} \mathcal{C}_1 & \text{if } \sigma\left(\ell_{(x,y)}(a)\right) > \frac{1}{2} \\ \mathcal{C}_2 & \text{if } \sigma\left(\ell_{(x,y)}(a)\right) \leq \frac{1}{2} \end{cases}$$

Implementation

- GM: With backtracking $(\gamma = 0.1, \sigma = 0.5, s = 1)$
- AGM: Follow the basic extrapolation strategy as before. But take $L = \frac{1}{4m}\sum_{i=1}^{m}\|a_i\|^2$ as the Lipschitz constant of $\nabla f_{\log}$.
- L-BFGS:
  - With backtracking$(\gamma = 0.1, \sigma = 0.5$, and $s = 1)$ and $H_0 = I$ as initial matrix.
  - $s^{k-1} = x^k - x^{k-1}, \quad y^{k-1} = \nabla f\left(x^k\right) - \nabla f\left(x^{k-1}\right)$, $\rho_k = \left(\left(s^k\right)^\top y^k\right)^{-1}$, $\gamma^k = \frac{\left(s^{k-1}\right)^\top y^{k-1}}{\|y^{k-1}\|^2}$, $H_k^0 = \gamma^k I$.Here we set the memory parameter $m_{L-BFGS} = 5$. And in the two loop recursion, if $k < m_{L-BFGS}$, we choose to iterate k times instead of m_{L-BFGS} times.

L-BFGS

Algorithm 2: The L-BFGS Method

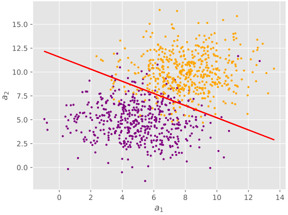
Initialization: set $\mathbf{x}^0 = 0$, $\mathbf{x}^0 \in \mathbb{R}^{n+1}$, $d^0 = -\nabla f(\mathbf{x}^0)$. And use backtracking $(\gamma = 0.1, \sigma = 0.5, s = 1)$ to decide α_0.

for $k = 1, 2, ..., max_{iter}$ **do**

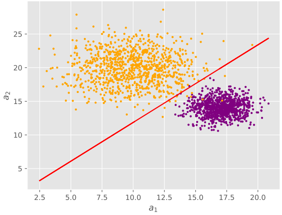
- set $\mathbf{x}^k = \mathbf{x}^{k-1} + \alpha_{k-1} d^{k-1}, s^{k-1} = \mathbf{x}^k - \mathbf{x}^{k-1}, y^{k-1} = \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k-1})$
- Break when $\|\nabla f(\mathbf{x}^k)\| <= tol$
- **if** $(s^k)^\top y^k > 10^{-14}$ **then**
  - $\gamma^k = \frac{(s^{k-1})^\top y^{k-1}}{\|y^{k-1}\|^2}$;
- **else**
  - $\gamma^k = 1$;
- set $q = \nabla f\left(\mathbf{x}^k\right)$
- **for** $i = k-1, k-2, \dots, k - m_{L-BFGS}$ **do**
  - set $\alpha_i = \rho_i \cdot \left(s^i\right)^\top q$ and
  - **if** $(s^i)^\top y^i > 10^{-14}$ **then**
    - $q = q - \alpha_i y^i$;
  - **else**
    - $q = q$;
- Set $r = H_k^0 q$
- **for** $i = k - m_{L-BFGS}, k - m_{L-BFGS} + 1, \dots, k-1$ **do**
  - **if** $(s^i)^\top y^i > 10^{-14}$ **then**
    - $\beta = \rho_i \cdot \left(y^i\right)^\top r$ and $r = r + (\alpha_i - \beta) s^i$;
  - **else**
    - $r = r$;
- set $r = H_k \nabla f\left(\mathbf{x}^k\right), d^k = -r$, use backtracking $(\gamma = 0.1, \sigma = 0.5, s = 1)$ to decide α_k.

Performance on Synthetic Datasets

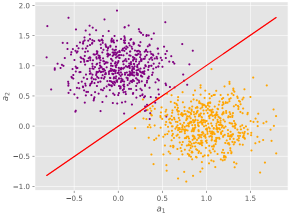
Seperate Line



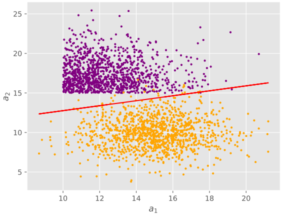
(a) Dataset1



(b) Dataset2



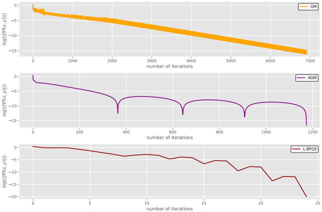
(c) Dataset3



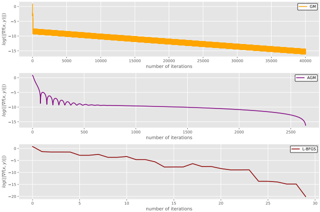
(d) Dataset4

Performance on Synthetic Datasets

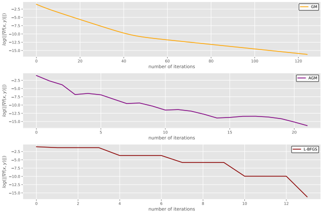
Convergence of $\log\left(\|\nabla f(x, y)\|\right)$



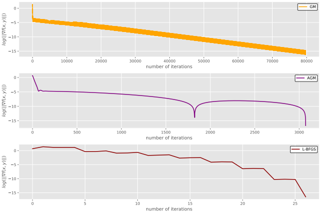
(a) Performance for Dataset1



(b) Performance for Dataset2



(c) Performance for Dataset3

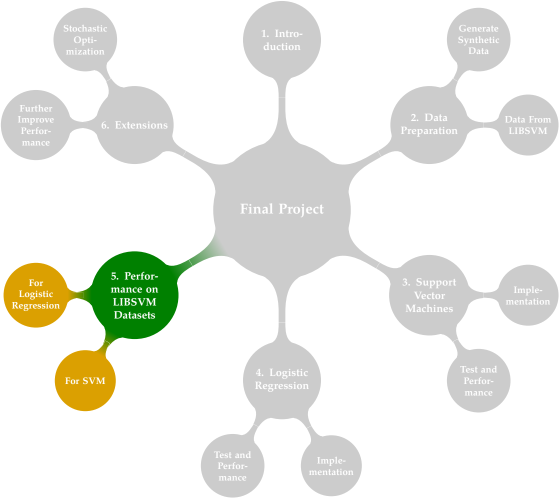


(d) Performance for Dataset4

Performance on Synthetic Datasets

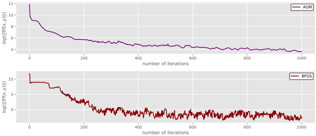
More performance index

datasets	methods	accuracy	iter	cpu-time	datasets	methods	accuracy	iter	cpu-time
dataset 1	GM	0.922	6903	4.31s	dataset 3	GM	0.991	124	0.64s
	AGM	0.922	1173	0.63s		AGM	0.991	21	0.69S
	L-BFGS	0.922	24	0.52s		L-BFGS	0.991	13	0.52s
dataset 2	GM	0.994	40068	27.01s	dataset 4	GM	0.989	79722	62.91s
	AGM	0.994	2644	1.16s		AGM	0.989	3073	1.44s
	L-BFGS	0.944	29	0.86s		L-BFGS	0.989	26	0.72S

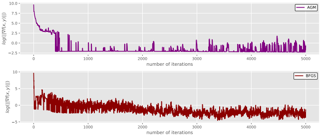

Final Project
1. Introduction
2. Data Preparation
Generate Synthetic Data
Data From LIBSVM
3. Support Vector Machines
Implementation
Test and Performance
4. Logistic Regression
Test and Performance
Implementation
5. Performance on LIBSVM Datasets
For Logistic Regression
For SVM
6. Extensions
Stochastic Optimization
Further Improve Performance

Performance on LIBSVM Datasets

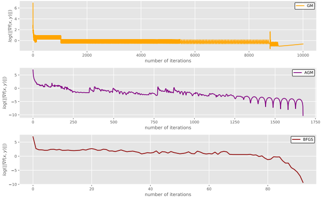
For SVM



(a) Performance for a9a Dataset



(b) Performance for mushroom Dataset



(c) Performance for breast-cancer Dataset

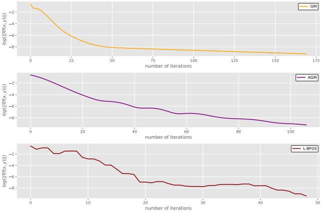
Performance on LIBSVM Datasets

For SVM

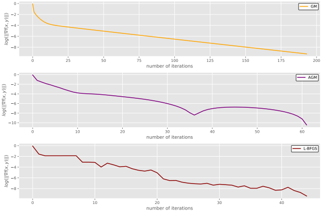
datasets	methods	accuracy	iter	cpu-time	datasets	methods	accuracy	iter	cpu-time
breast cancer	GM AGM BFGS	0.956 0.956 0.956	10000 1673 92	154.97s 2.18s 0.59 S	a9a	GM AGM BFGS	# 0.850 0.850	# 1000* 1000*	# 51.91s 176.33s
mushroom	GM AGM BFGS	# [2] 1.000 1.000	# 5000 5000	# 53.31s 124.17s	#	#	#	#	#

[2] Due to limitation of time and computing resources, we suspend these parts.

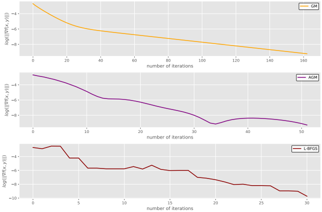
For Logistic Regression



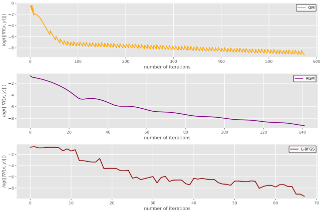
(a) mashroom Dataset



(b) breast cancer Dataset



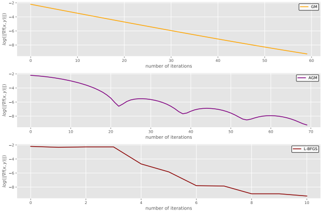
(c) covtype Dataset



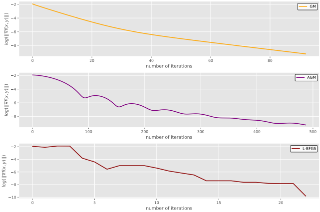
(d) phishing Dataset

Performance on LIBSVM Datasets

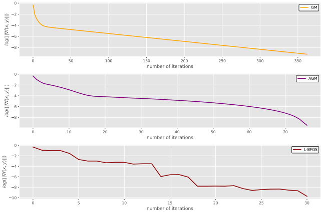
For Logistic Regression



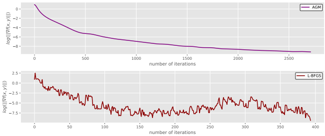
(a) rcv1 Dataset



(b) news20 Dataset



(c) a9a Dataset

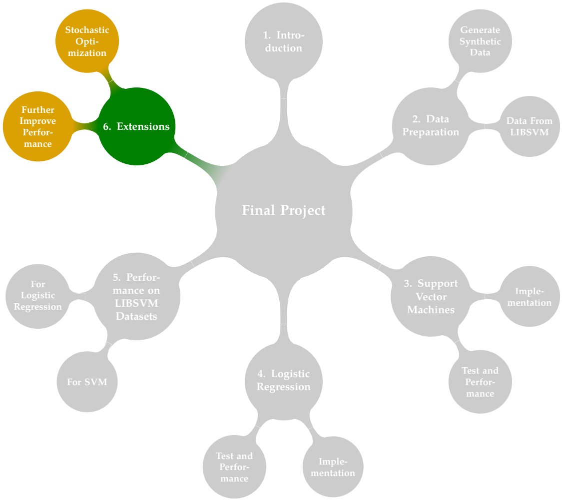


(d) gisette Dataset

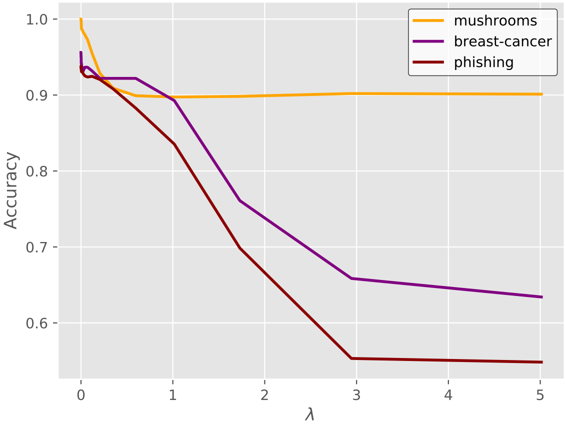
For Logistic Regression

datasets	methods	accuracy	iter	cpu-time	datasets	methods	accuracy	iter	cpu-time
breast-cancer	GM	0.93	193	0.47s	mushroom	GM	0.96	169	0.60s
	AGM	0.93	61	0.27s		AGM	0.96	106	0.37s
	L-BFGS	0.93	44	0.41s		L-BFGS	0.96	48	0.93s
a9a	GM	0.80	362	4.34s	covtype	GM	0.63	162	16.79s
	AGM	0.80	76	0.76s		AGM	0.63	51	3.54s
	L-BFGS	0.80	30	1.36s		L-BFGS	0.63	30	12.52s
Phishing	GM	0.92	574	3.08s	news20	GM	0.89	92	38.54s
	AGM	0.92	141	0.55s		AGM	0.89	487	190.98s
	L-BFGS	0.92	67	1.06s		L-BFGS	0.89	22	107.54s
Rcv1	GM	0.84	59	8.18s	gisette	GM	# [3]	#	#
	AGM	0.84	69	6.64s		AGM	0.97	2712	953.09s
	L-BFGS	0.84	10	5.53s		L-BFGS	0.97	393	605.16s

[3] Due to limitation of time and computing resources, we suspend these parts.


Final Project
1. Introduction
2. Data Preparation
Generate Synthetic Data
Data From LIBSVM
3. Support Vector Machines
Implementation
Test and Performance
4. Logistic Regression
Test and Performance
Implementation
5. Performance on LIBSVM Datasets
For Logistic Regression
For SVM
6. Extensions
Stochastic Optimization
Further Improve Performance

Further Improve Performance: Take Logistic Regression model with L-BFGS method as the example to adjust the regularization parameter λ.



Stochastic Optimization

- Methodology:
  - Empirical risk minimization problem:

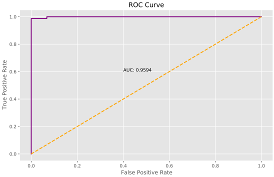
$$\min_{x,y} f(x,y) = \frac{1}{m}\sum_{i=1}^{m} f_i(x,y)$$

  - Update Rule:

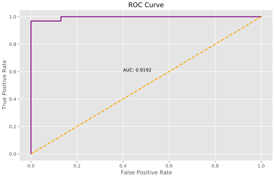
$$\begin{pmatrix} x^{k+1} \\ y^{k+1} \end{pmatrix} = \begin{pmatrix} x^{k} \\ y^{k} \end{pmatrix} - \frac{\alpha_k}{|\mathcal{S}_k|}\sum_{i\in\mathcal{S}_k} \nabla f_i\left(x^k, y^k\right)$$

Stochastic Optimization

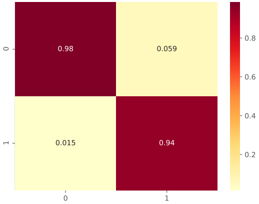
- Implementation and Performance:



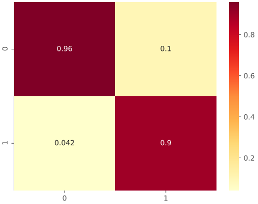
(a) mushroom Dataset



(b) phishing Dataset



(c) mushroom Dataset



(d) phishing Dataset

Main Observations

- Logistic Regression Model performs better than SVM model in terms of convergence and speed
- L-BFGS performs quite well on large-scale datasets.
- SGD performs quite well in spite of high randomness and bad convergence.

Future Works

- There are many other strategies can be applied to improve the performance of the models and algorithms.
- More principles should be investigated in order to have better interpretation of the results.
- All of our works are done via github, we will keep updating and optimizing this project in the future via our repository (https://github.com/Yihang-Li/MDS6106Project).

Happy New Year!
Frohes neues Jahr!