Мультимоделирование SVM

Сергей Иванычев
sergeyivanychev@gmail.com

Александр Адуенко
aduenko1@gmail.com

Московский физико-технический институт
Факультет управления и прикладной математики
Кафедра «Интеллектуальные системы»

Предпосылки. Комбинирование алгоритмов.

Понятие

Сильный классификатор — $|\mathrm{AUC} - 0.5| \gg 0$ во многих реальных задачах.

Примеры: SVM, логистическая регрессия, Lasso...

Понятие

Слабый классификатор — $|\mathrm{AUC} - 0.5| > 0$ в большинстве задач

Примеры: решающий пень...

Предпосылки. Комбинирование алгоритмов.

Комбинирование сильных классификаторов

- Простое голосование
- Линейная комбинация
- ...

Комбинирование слабых классификаторов

- Bagging
- Boosting

Предпосылки. Задача

Вопросы

Может ли нелинейная комбинация классификаторов решать задачу лучше, чем каждый классификатор по отдельности?

Сужение задачи: модели — SVM с разными ядрами, комбинирующий алгоритм — логистическая регрессия.

Цели исследования

Цели

- Построить лучшую по сравнению с отдельными моделями комбинацию (супермодель)
- Установить связь между множествами опорных объектов и схожестью классификаторов
- Использовать знание множеств опорных объектов для улучшения комбинации

Постановка задачи

Пусть $X^l = (x_i, y_i)_{i=1}^l, x \in R^n, y \in \{\pm 1\}$.

Определения

S-я модель — SVM с ядром K_s из множества ядер:

$$\mathcal{K} = \{K_j\}_{j=1}^m$$

Отступ — значение дискриминантной функции на объекте

$$M_s = \sum_{i=1}^{l} \lambda_i y_i K_s(x_i, x) - w_0$$

Постановка задачи

$$M = \begin{pmatrix} M_1 & \dots & M_s \end{pmatrix}$$

— матрица отступов, новая матрица «объект-признак».
Пусть $\mathcal{A}$ – множество алгоритмов классификации.

$$\mathcal{A} = \{a(x) = g(x, \theta) | \theta \in \Theta\} \quad g : R^m \to Y$$

Пару $(g, \mathcal{K})$ будем называть **супермоделью**.

Задача выбора алгоритма комбинирования

$$L(y, g(M(X^l), \theta)) \to \min_{\Theta}$$

Где L — функционал качества (в нашем случае AUC)

Литература

1. Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. 1995
2. Alex J Smola et al. A Tutorial on Support Vector Regression. 2004
3. Rauf Izmailov, Vladimir Vapnik and Akshay Vashist. Multidimensional Splines with Infinite Number of Knots as SVM Kernels. 2013
4. D. Gorgevik и D. Cakmakov. Handwritten Digit Recognition by Combining SVM Classiers. 2005
5. Salah Althloothi и др. Human activity recognition using multi-features and multiple kernel learning. 2014
6. S.S. Bucak, R. Jin и Ak. Jain. Multiple Kernel Learning for Visual Object Recognition: A Review. 2014

Связь между разными расстояниями.

Необходимо найти способ определять схожие модели, то есть дающие схожие результаты, чтобы не включать таковые в супермодель.

Расстояния

Опорным расстоянием $\rho_S(K_i, K_j)$ на выборке X^l будем называть функцию:

$$\rho_M(K_i, K_j, X^l) = \frac{\#\left[\mathrm{SV}_i \,\Delta\, \mathrm{SV}_j\right]}{\#\left[\mathrm{SV}_i \cup \mathrm{SV}_j\right]}$$

Отступным расстоянием $\rho_M(K_i, K_j)$ будем называть следующую функцию

$$\rho_M(K_i, K_j, X^l) = 1 - \mathrm{corr}(M_i, M_j)$$

Связаны ли эти расстояния? Проанализируем эволюцию распределения пар расстояний в зависимости от параметра регуляризации.

Эксперимент. Ядра и данные.

В качестве исходных данных взяты датасеты German Credits, Wine и Heart disease из UCI.
Ядра:

- Линейное
- Полиномиальное (степени 3, 4, 5)
- RBF-ядро ($\gamma \in \{0.0001, 0.001, 0.01, 0.1, 1\}$)
- INK-spline ядро

Эксперимент. German credit

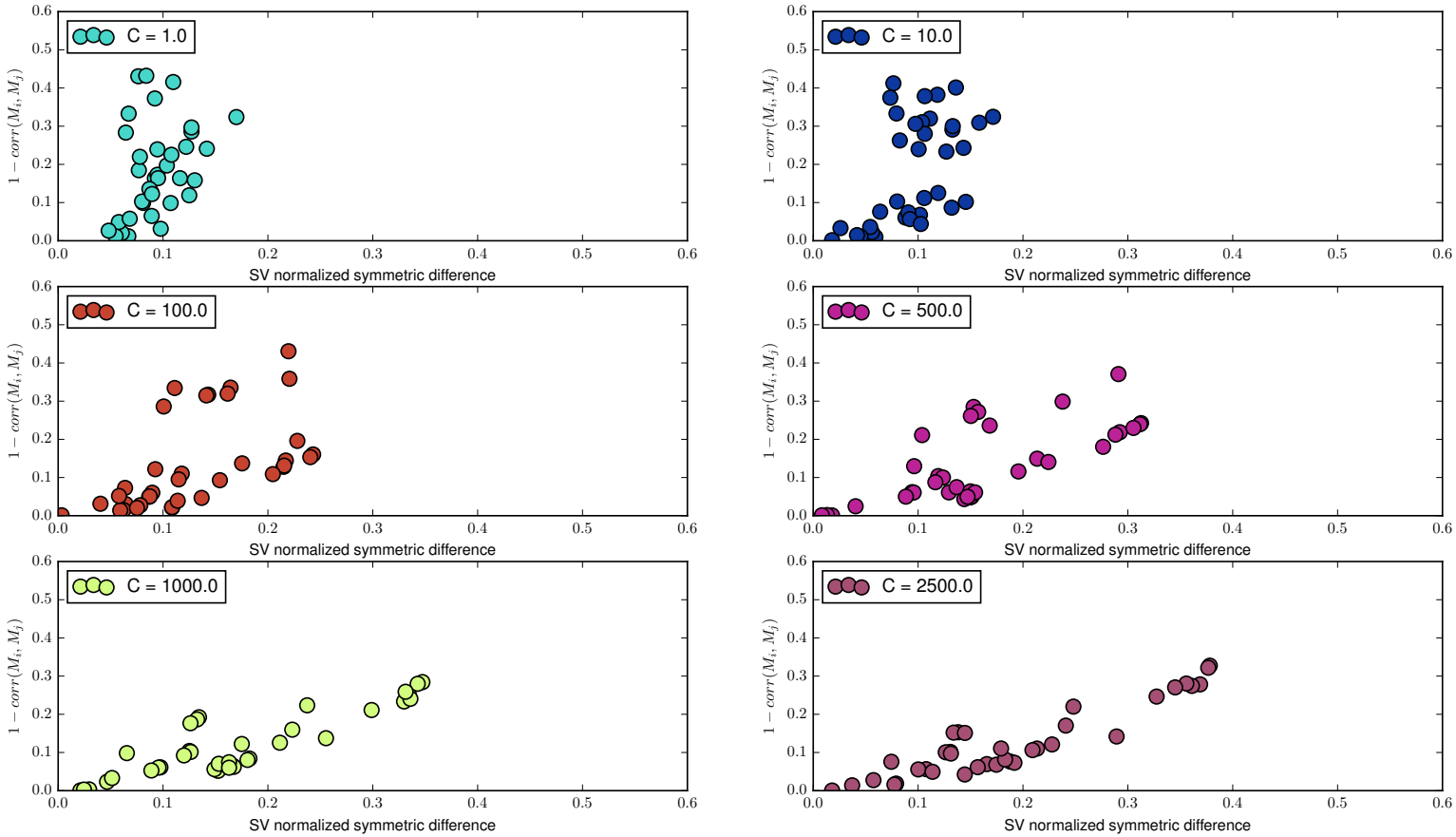

Рис.: German credit

Эксперимент. German credit

Таблица: German info

	$\mathrm{mean}(\#\mathrm{SV})$	$\mathrm{mean}(\rho_M)$	$\mathrm{mean}(\rho_S)$	Correlation
$C = 1.0$	603.4	0.184	0.094	0.376
$C = 10.0$	603.6	0.187	0.097	0.537
$C = 100.0$	594.7	0.134	0.131	0.556
$C = 500.0$	584.3	0.133	0.161	0.717
$C = 1000.0$	581.6	0.120	0.172	0.870
$C = 2500.0$	577.9	0.126	0.189	0.918

Эксперимент. Wine

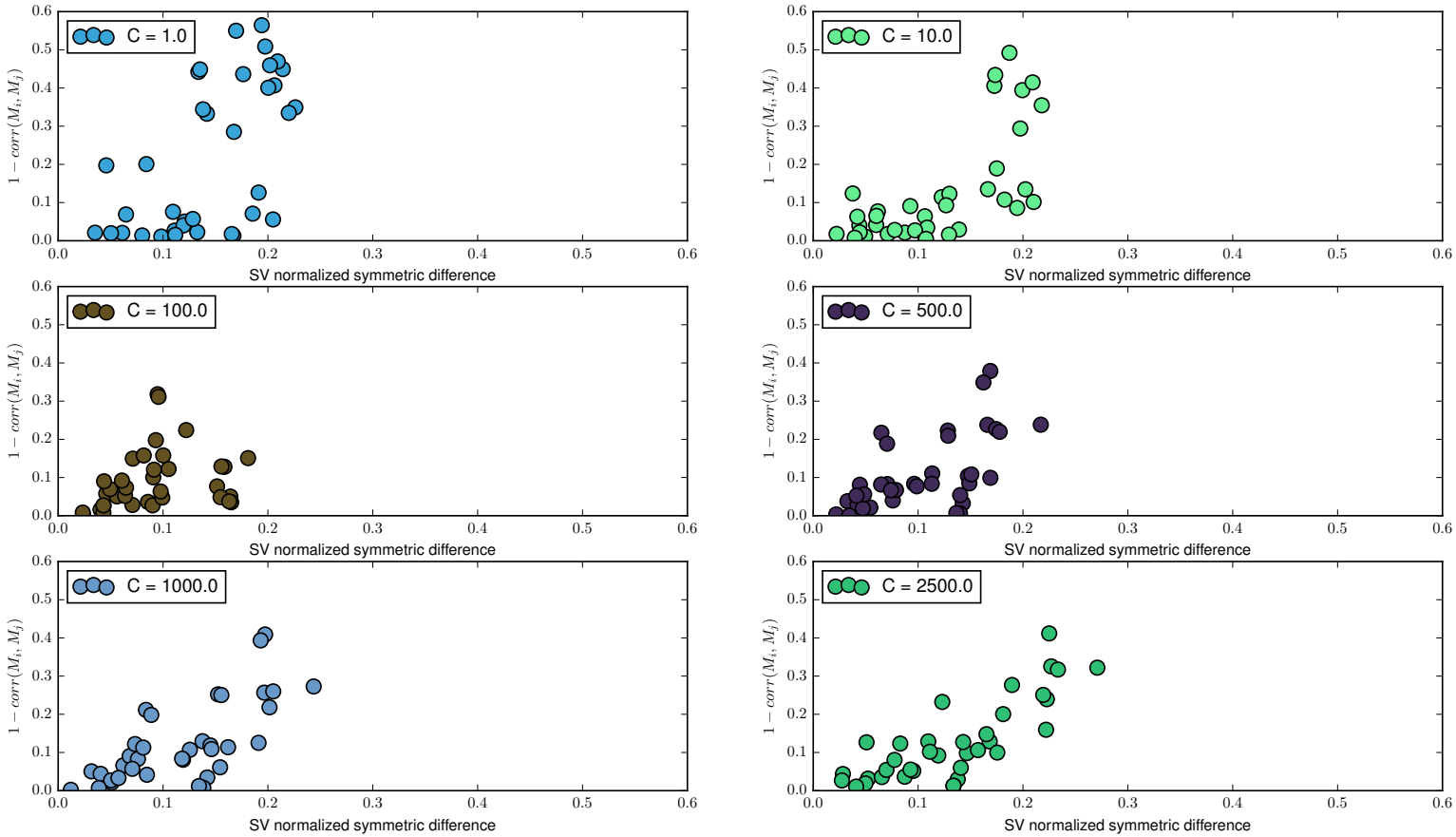

Рис.: Wine

Эксперимент. Wine

Таблица: Wine info

	$\mathrm{mean}(\#\mathrm{SV})$	$\mathrm{mean}(\rho_M)$	$\mathrm{mean}(\rho_S)$	Correlation
$C = 1.0$	3284.1	0.220	0.144	0.600
$C = 10.0$	3284.9	0.130	0.121	0.687
$C = 100.0$	3275.0	0.091	0.091	0.270
$C = 500.0$	3252.6	0.110	0.105	0.591
$C = 1000.0$	3235.2	0.124	0.118	0.694
$C = 2500.0$	3208.6	0.127	0.133	0.795

Эксперимент. Heart disease

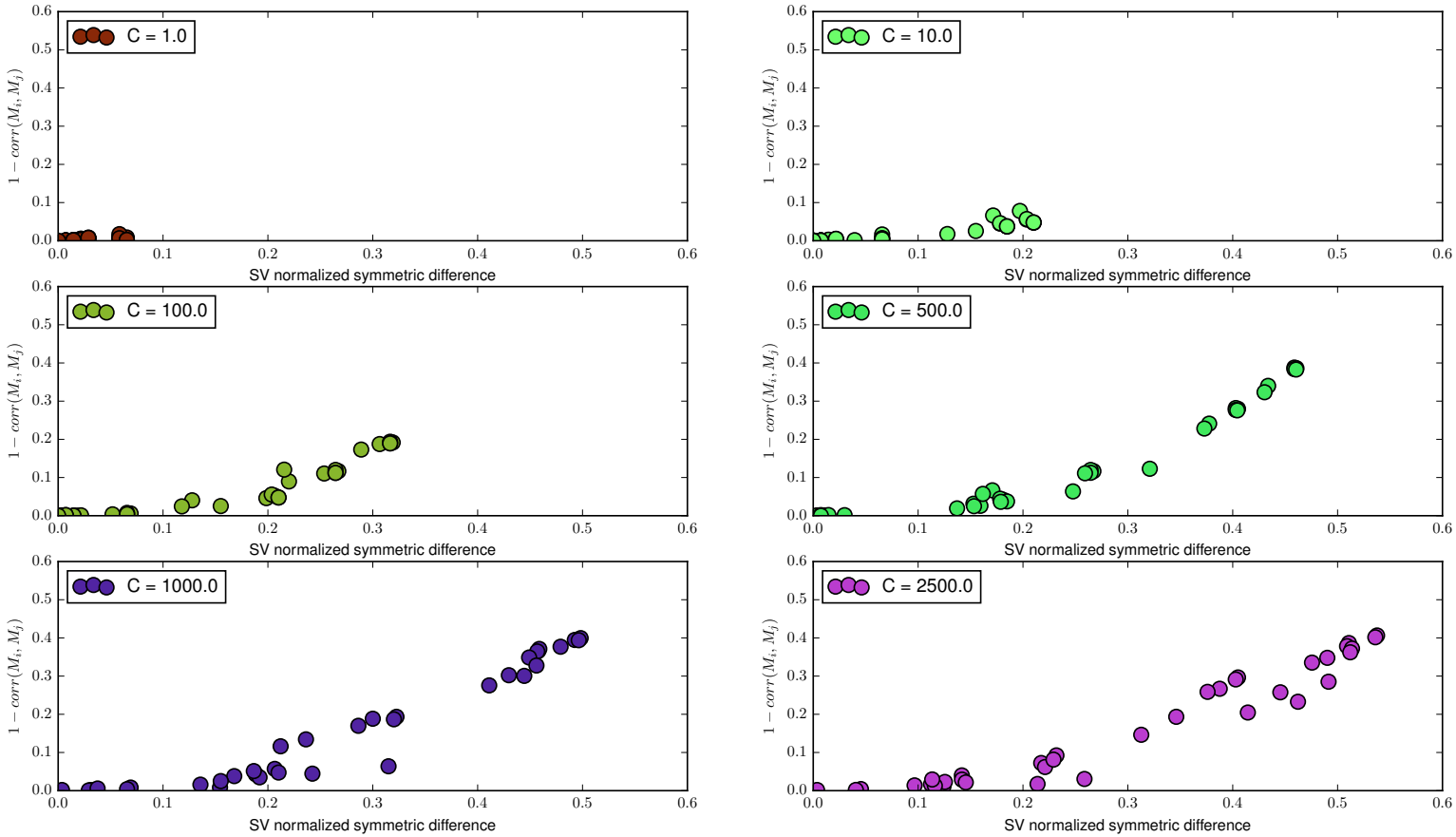

Рис.: Heart disease

Эксперимент. Heart disease

Таблица: Heart info

	$\mathrm{mean}(\#\mathrm{SV})$	$\mathrm{mean}(\rho_M)$	$\mathrm{mean}(\rho_S)$	Correlation
$C = 1.0$	272.0	0.003	0.027	0.608
$C = 10.0$	260.8	0.020	0.088	0.929
$C = 100.0$	249.1	0.063	0.152	0.927
$C = 500.0$	231.9	0.135	0.238	0.940
$C = 1000.0$	223.1	0.157	0.268	0.953
$C = 2500.0$	211.4	0.166	0.297	0.962

Результаты эксперимента

- С ростом константы регуляризации расстояние между ядрами и расстояние между их отступами лучше коррелируют между собой.
- При высоких параметре регуляризации коэффициент корреляции Пирсона достигает более 0.8, то есть расстояния практически линейно зависят друг от друга.

Вывод

Если множества опорных объектов пары классификаторов похожи , то и векторы отступов похожи

Построение супермодели

Обозначения: $X_{\text{train}}, X_{\text{test}}, y_{\text{train}}, y_{\text{test}}$

```
/* Обучение                                                    */
SVM.fit(X_train, y_train);
M_train ← SVM.margin(X_train);
Logregr.fit(M_train, y_train);
/* Прогнозирование                                             */
M ← SVM.margin(X_test);
Logregr.predict_probability(M);
```

Algorithm 1: Наивная логистическая регрессия

Плохо! Если во множестве моделей есть переобученный классификатор, то супермодель будет также переобучена

Построение супермодели

/* Обучение */
$X_{\text{train}}, X_{\text{val}} \leftarrow \text{split}(X)$;
$\text{SVM.fit}(X_{\text{train}}, y_{\text{train}})$;
$M_{\text{val}} \leftarrow \text{SVM.margin}(X_{\text{val}})$;
$\text{Logregr.fit}(M_{\text{val}}, y_{\text{val}})$;
/* Прогнозирование */
$M \leftarrow \text{SVM.margin}(X_{\text{test}})$;
$\text{Logregr.predict\_probability}(M)$;

Algorithm 2: Стэкинг

Уже лучше, однако не понятно, как интерпретировать отрицательные веса в логистической регрессии.

Построение супермодели

Используем **робастную логистическую регрессию**. Преобразуем задачу оптимизации:

$$R(w) + C\sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1) \to \min_{w \geqslant 0, c}$$

Получаем вектор неотрицательных весов

Выводы

Эксперимент был проведен на датасетах Housing, Heart, German и синтетически сгенерированном. С учетом примененных эвристик, таких как:

- Использование классификаторов только из интервала $[\mathrm{AUC}_{\mathsf{best}}, \mathrm{AUC}_{\mathsf{best}} - \delta]$
- Повторное обучение SVM на X, после обучения регрессии
- Использование l_1-регуляризации для логистической регрессии

пока не удалось получить супермодель с устойчивым выигрышем.

$$\mathrm{AUC}_{\mathsf{best}} - \mathrm{AUC}_{\mathsf{super}} \approx 0.002$$

Направления дальнейшей работы

- Усовершенствование или исправление комбинирующего алгоритма.
- Использование расстояний между моделями для улучшения качества предсказаний.

Code sharing

Исходный код проекта написан на языке Python 3.5. Код и данные доступны по ссылке:

https://sourceforge.net/p/mlalgorithms/code/HEAD/tree/Group374/Ivanychev2016SVM_Multimodelling/

Супермодель представлена в виде объекта `SVMSupermodel` и обладает стандартным интерфейсом классификаторов библиотеки `scikit-learn`.

Основная статья, а также подробная документация доступна в папке с проектом.