Яндекс

MatrixNet

Андрей Гулин

Москва, 16.12.2010

Машинное обучение

- Supervised setting
  - Примеры с правильными ответами
  - Подбираем модель + параметры
- Модели
  - Линейная
  - Полином
  - Нелинейная (f.e. “Нейронная” сеть)
  - Деревья (decision tree)
  - …

Применение ML

- Можно предсказать
  - Будет ли завтра дождь?
  - Дадут ли грант?
  - Получился ли бозон Хиггса?
  - Кликнет ли пользователь на первый ответ?
  - Пользователь – человек? Какого пола?
  - …

[скачать ответ бесплатно]

- WEKA
- SVMlight/SVMlib
- R-пакет gbm
- OpenCV
- Apache Mahout
- …

Методы решения

- Boosted trees
- Random forest
- SVM
- Neural networks
- …

Поисковая система

- Скачиваем Интернет
- Запрос -> N ответов
- Хорошие & Неподходящие ответы
- Спрашиваем людей оценить
- Тренируем машинку отвечать как люди

Я

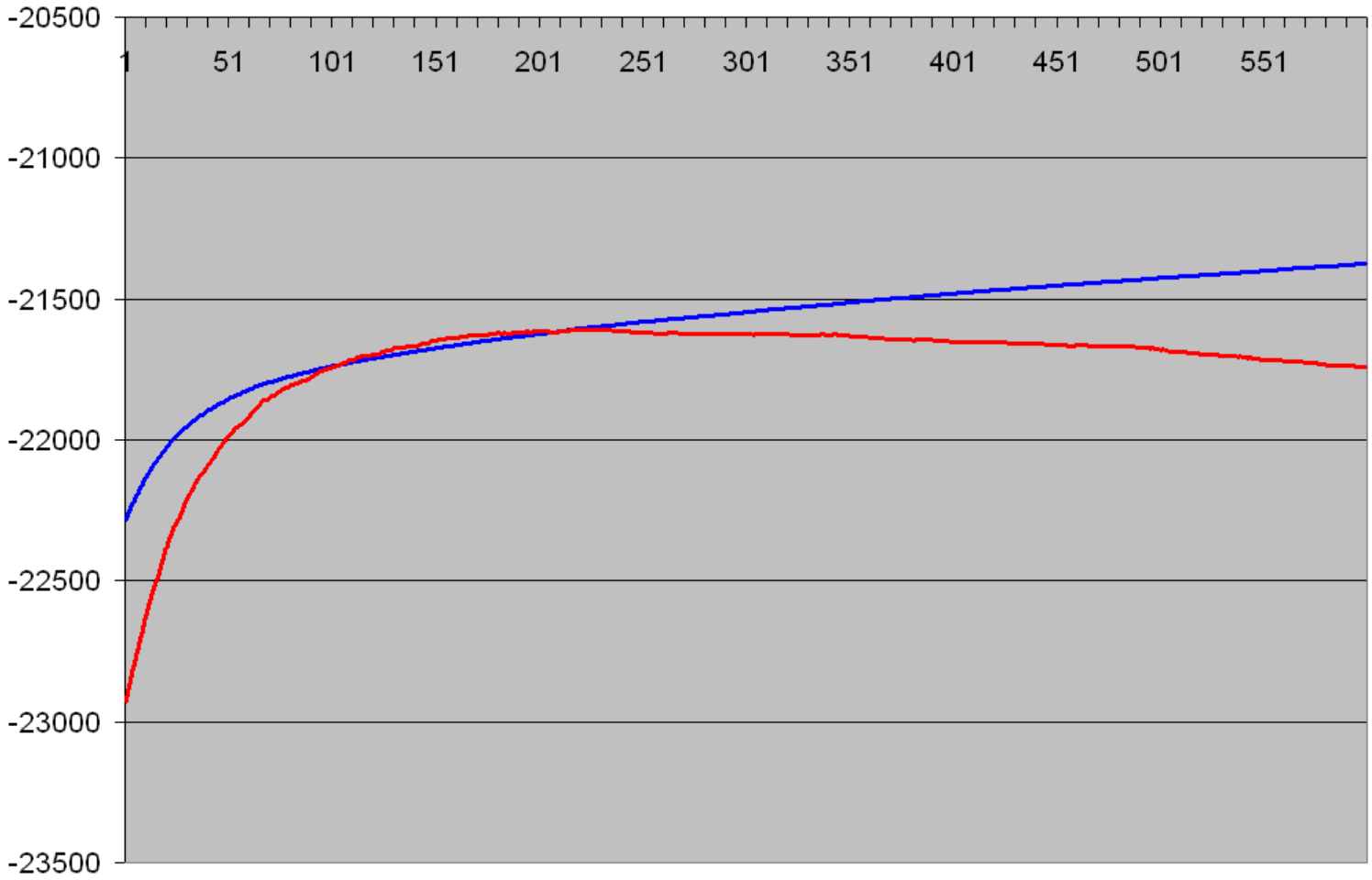
Линейная регрессия

- Дано: K N-мерных самплов $\{x_i\}$ для каждого известно значение функции $\{f_i\}$
- Найти: вектор a, такой что $a^T x_i = f_i$
- Решение: $a=(X^T X)^{-1} f$

Overfitting

- Можем подобрать параметры “слишком хорошо”
- Проблема переобучения
- Тестовая выборка

Overfitting



Регуляризация

- Когда данных мало простое решение не работает
- Нужна какая-то дополнительная информация, например, мы можем сказать, что мы хотим “маленький” или “простой” вектор a
- $(Xa^T-f)^T (Xa^T-f) + \lambda |a|_k \to \min$
- Меры простоты:
  - L0 = feature selection
  - L1 = lasso
  - L2 = ridge = по Тихонову $[a=(X^TX+\lambda I)^{-1}f]$

Я

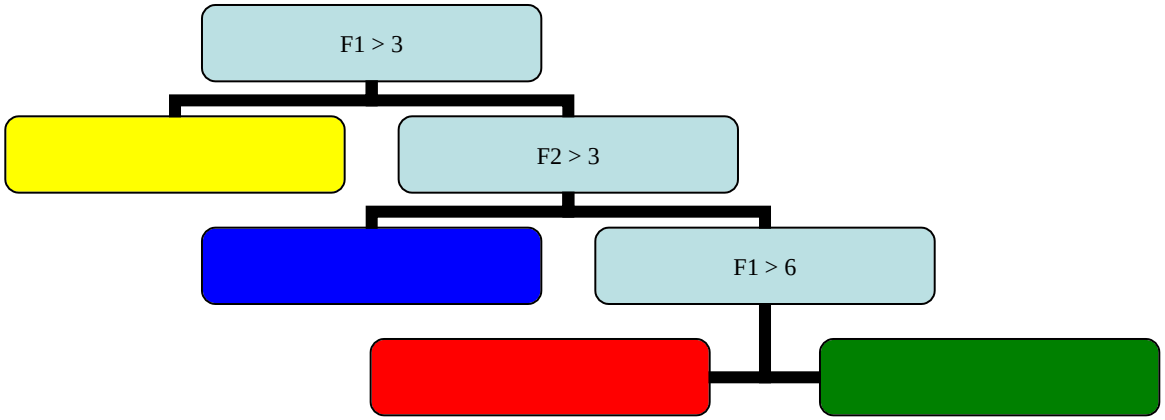
L1 регуляризация

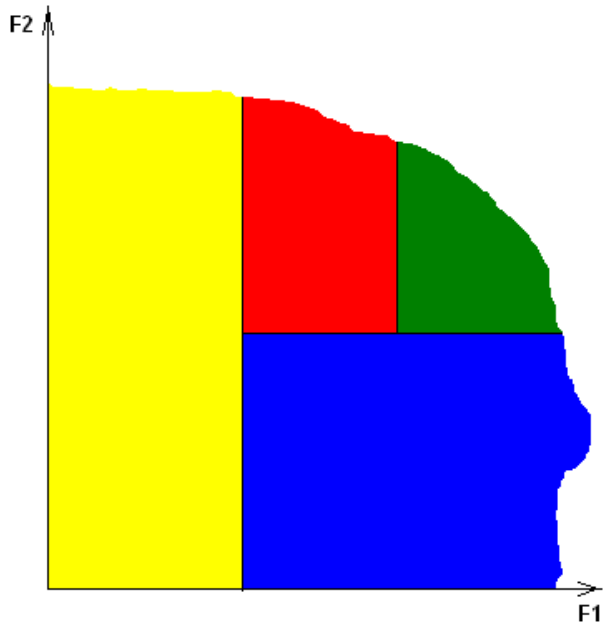
- Покоординатный спуск = итеративный алгоритм L1 регуляризации
- У нас есть текущий “остаток” r_i, который в начале равен f_i
- На каждой итерации мы
  - Выбираем самый похожий на r_i фактор и считаем с каким множителем α нам нужно его брать
  - Добавляем $\lambda\alpha$ к коэффициенту при этом факторе ($\lambda < 1$)
  - Считаем новый остаток r_i
- http://www-stat.stanford.edu/~tibs/lasso.html

Decision Tree

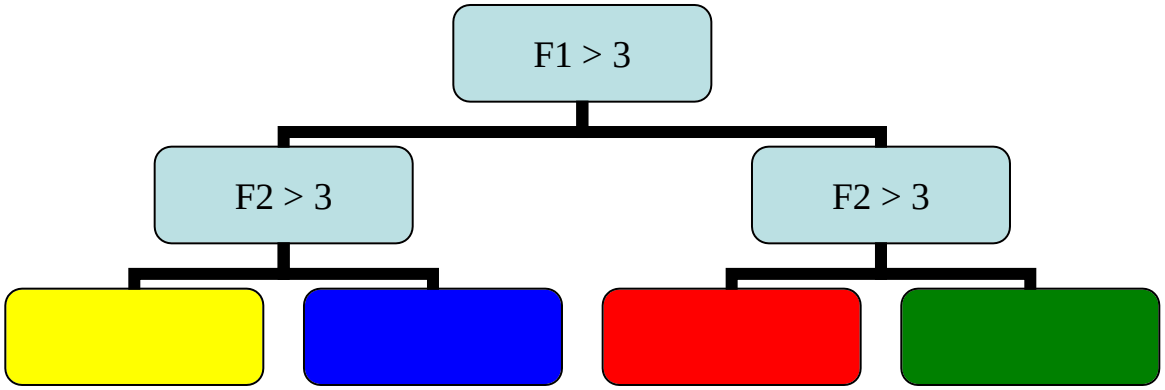
- На каждом шаге будем строить дерево, максимально подходящее под “остаток” r_i
- Преимущества деревьев
  - Не зависит от монотонных преобразований факторов
  - Можно аппроксимировать любую “разумную” функцию
  - Ошибка не тесте лучше для многих реальных данных

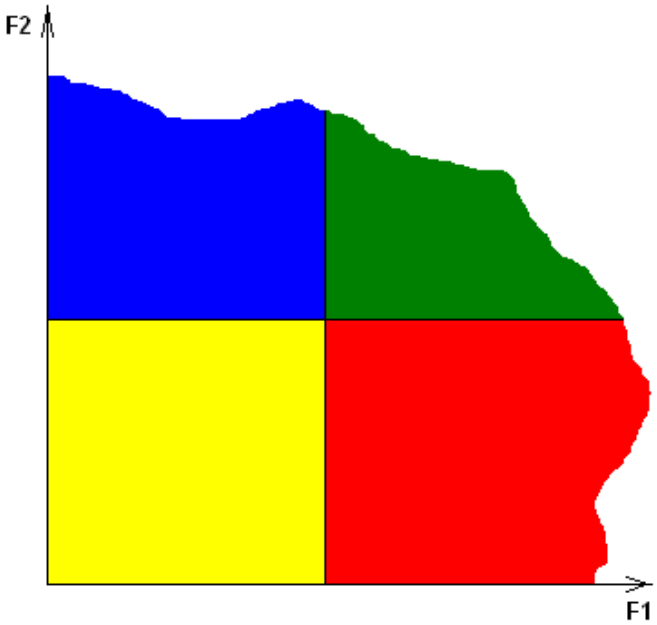
Decision Tree





Oblivious Trees





More sparsity

- L1 даёт разреженный (sparse) результат и это хорошо
- Можно регуляризовать каждое полученное дерево, чтобы сделать их более разреженными
- F.e. обнулим те листья дерева куда попало мало примеров

Можно ли лучше L1?

- L0 теоретически самая хорошая, но NP-complete
- L1 выпуклая и легко считается
- L0.5 регуляризация – компромисс.
- Для L0.5 существует сравнительно быстрый алгоритм
- “Data Modeling: Visual Psychology Approach and L1/2 Regularization Theory” by Zongben Xu

Bootstrapping

- Если выборка независимая, то мы можем сделать из неё n^n выборок
- Для получения распределения статистики можно брать разные выборки и считать для них статистику
- Использование разных выборок на каждой итерации вместо исходных данных улучшает результат

Gradient Boosting

- Наш алгоритм это почти Gradient Boosting
- Вместо r_i будем приближать производную целевой функции
- Profit

Целевые функции

- Не всегда MSE – лучший выбор
  - Классификация: log(p)
  - Ранжирование: log(p)
- Вместо p (0;1) оптимизируем по x (-inf;inf)
  - p = 1/(1+exp(-x))

Запускаем

- Хотим, чтобы работало быстро
- Multicore
- Много маленьких задач
- Существенные потери на лагах синхронизации

Сериализация

- Сериализация = параллельное выполнение даёт тот же результат, что и какое-то из последовательных
- Способы сериализации
  - Locks (mutex/critical section)
  - Fine grain Locks
  - Single instruction Locks (atomic read/modify/write)
  - *Transactional memory (возможное будущее)*

Я

Non-blocking algorithms

- Non-blocking = подвисание одного или нескольких тредов не делает невозможным продолжение вычислений в остальных тредах
- Lock-free = гарантируется system-wide progress
- Wait-free = для каждой операции есть верхняя граница количества шагов алгоритма

Lockfree algorithms

- Atomic counters
- Что угодно можно сделать Lockfree с помощью атомарного CAS (CompareAndSwap)
- С помощью CAS меняем указатель на структуру со старым состоянием на структуру с новым состоянием

ABA problem

- Указатель – неуникальный идентификатор, memory allocator выделяет освобождённые блоки повторно
- Можно к указателю добавить счётчик для большей уникальности (1996, Michael & Scott)
  - Требует DCAS (double word CAS)
- Можно и без DCAS, но или медленнее или ненадёжно

Lockfree в MatrixNet

- Использует lockfree очередь задач (+3-5%)
- Выигрыш растёт с увеличнием количества ядер
- Гораздо больший прирост в случае распределённых вычислений (лаг на любой из коробок тормозит весь процесс)

Распределённый MatrixNet

- Много multicore > одной multicore?
- Не всегда
- Network delays & “lost” packets – используя свой протокол поверх UDP можно заметно снизить задержки

Вопросы?


Akasaka Mitsuki
Everyone possesses many different faces.
For living...
For working...
And for falling in love...
I wonder how many faces I have?
And which one is the real me?
ANIMEWALLPAPERS.COM

Я