ML BD

Katya Koshchenko

Spring 2020

1 Лекция 5. Categorical Features.

Фичи бывают: бинарные, вещественные, категориальные (айди, категория, город и тд), порядковые. Кросс-фича $X_{ij} \in C_i \times C_j$ построена из двух категориальных фичей. Она может иметь и sparse (one-hot encoding), и dense (target mean encoding) представление.

Даже на искусственном примере с полом юзера и доменом объявления кросс-фичи вроде могут быть полезными, так как несут обычно для модели очень мощный сигнал. Но с ними есть проблемы:

- Ручной поиск таких правил очень трудоемкий. Надо разбираться в предметной области, чтобы искать такие сочетания. В примере были кросс-фичи второго порядка (две категориальные фичи в представлении). А если порядок увеличивать, то сложность еще сильнее возрастает.
- При их использовании размерность признакового пространства быстро растет. Тк даже если у нас есть 3 варианта пола, 20000 объявлений, это уже 60000. А если размерность категориальных фичей миллионы, то безумно много. С такими большими пространствами сложно работать

Сегодня говорим про то ,как люди на практике с ними борятся.

1.1 Field-aware Factorization Machines

1.1.1 Poly2

Самая простая — Poly2. Как перейти от ручного поиска кроссфичей — добавить сразу все фичи. То есть рассмотреть все попарные умножения с некоторым весом:

$$Poly2(W, X) = LM(W, X) + \sum_{j1=1}^{n} \sum_{j2=j1+1}^{n} W_{h(j1,j2)} \cdot X_{j1} \cdot X_{j2}$$

где h кодирует j1, j2 в натуральное число. Проблемы подхода:

- Вычислительная сложность. Если ненулевых слагаемых n', то будет $O(n' * *2)$ действий.
- Получаем только кроссфичи второго порядка.
- С этой моделью легко переобучиться. Если пара X_j1, X_j2 в обучающем множестве встречается редко, то вес в слагаемом не сможем достаточно хорошо обучить.

1.1.2 FM

Factorization machines — следуюий шаг в развиитии poly2. Отличие тв том, что в каждой паре был отдельный вес, а теперь у каждой фичи будем обучать вектор веса размерностью k:

$$FM(W, X) = LM(W, X) + \sum_{j1=1}^{n} \sum_{j2=j1+1}^{n} (W_{j1} \cdot W_{j2}) \cdot X_{j1} \cdot X_{j2}$$

Обычно это k небольшое. Почему это вообще должно работать. Потому что это эмбеддинги, ура. Можно любую матрицу положительноопределенную разложить на WWT, а это оно и есть. Плюс подхода: переобучиться сложнее, тк для каждой фичи обучаем отдельный вектор. Сложность почти такая же: $O(n' * *2 * k)$. Но можно немного модицифровать подсчет, и тогда будет O(n' * k):

$$FM(W, X) = \frac{1}{2} \cdot \sum_{j=1}^{n} [(\sum_{k=1}^{n} W_k \cdot X_k) - W_j \cdot X_j] \cdot W_j \cdot X_j$$

(На слайде график как меняется RMSE на датасете нетфликса. Даже если на каждом признаке обучать маленький вектор, то по сравнению с SVM уже растет качество.)

1.1.3 FFM

Field Aware Factorization Machines. Есть запись, что юзер кликнул по объявлению на такой площадке, с таким рекламодателем, такой-то пол. В обычных FM у все попарные произвдеения признаков — три слагаемых. Хотим дать больше степеней свободы модельке. Все фичи можно разбить на категории (fields), и для каждой фичи обучать не один, а F векторов. И теперь при разбиении на пары брать именно тот вектор, который соответствует напарнику фичи:

$$FFM(W, X) = LM(W, X) + \sum_{j1=1}^{n} \sum_{j2=j1+1}^{n} (W_{j1,f2} \cdot W_{j2,f1}) \cdot X_{j1} \cdot X_{j2}$$

На практике размерность категорий обычно сильно меньше размерности фичей. Время работы O(n'**2 * k).

1.1.4 Оптимизация

Как обучать. Можно, конечно, SGD, но обычно гораздо лучше работает Adagrad. На слайде сравнения всякие. Тк ушли от того, что считали по отдельному весу для каждой пары, а вме-

сто этого обучали вектора, то сходиться стало быстрее, и время обучения уменьшилось. FFM обучается помедленнее, тк для каждой фичи больше векторов, но зато размерность векторов сильно меньше можно сделать. Ну и loglos сильно меньше у него.

Оставшиеся минусы. Все еще только фичи второго порядка. И все еще долго обучается.

1.2 Deep Crossing

Теперь про глубокое обучение. Умеем обучать кроссфичи второго порядка. Хотим находить более сложные, но чтобы при этом не переобучались, были устойчивы и тд. Для этого есть Deep Crossing. Что это за модель. Взяли one-hot, заэмбеддили. Затем их все вектора сложили в один большой вектор. В него же суем вектор с некатегориальными (веществennозначными) фичами, можно даже не эмбеддить. Суем это все дело в пять слоев с relu и residual connections (архитектура на слайде). В явном виде веса для фичей мы не получим, то есть не проинтерпретируешь. Но вроде эта вся радость обучается.

1.3 Deep 'n Cross

Сделали опять вектор вещественных с вектором эмбеддингов категориальных. Засовываем эту штуку в Deep network и Cross Network. Deep — обычный DNN. Cross network — пытаются в явном виде обучать взаимодействия определенного порядка. Сколько уровней, вплоть до такого порядка и обучим.

На слайдах таблички с evaluation. Если посмотреть на logloss, то разница с FFM в четвертом знаке, но утверждается, что это уже приносит много денег. На слайде таблица со сравнением числа параметров DCN и DNN, нужных, чтобы добиться какого-то logloss. У DCN это число сильно меньше из-за Cross network подсетки, тк она дает высшие фичи.

Есть ли смысл в итерациях больше второго порядка? На слайде опять график. Как только в cross network появляется хоть 2й слой, то logloss сильно падает, то есть фичи второго порядка сильно помогают. 3й слой уже не так сильно помогает, так что 2го порядка кажется, что уже хватает (но авторы это все привели только на одном датасете).

1.4 Deep FM

Просто взяли FM сложили в сетку. FM компонента: считаем попарные произведение между эмбеддингами. Deep компонента: все эмбеддинги отправляются в полносвязный слой, и с ними что-то происходит.