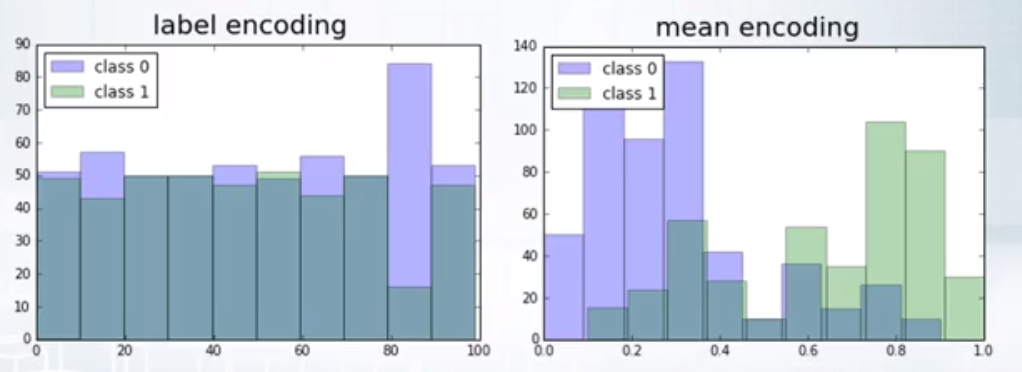
# Concept of mean encoding

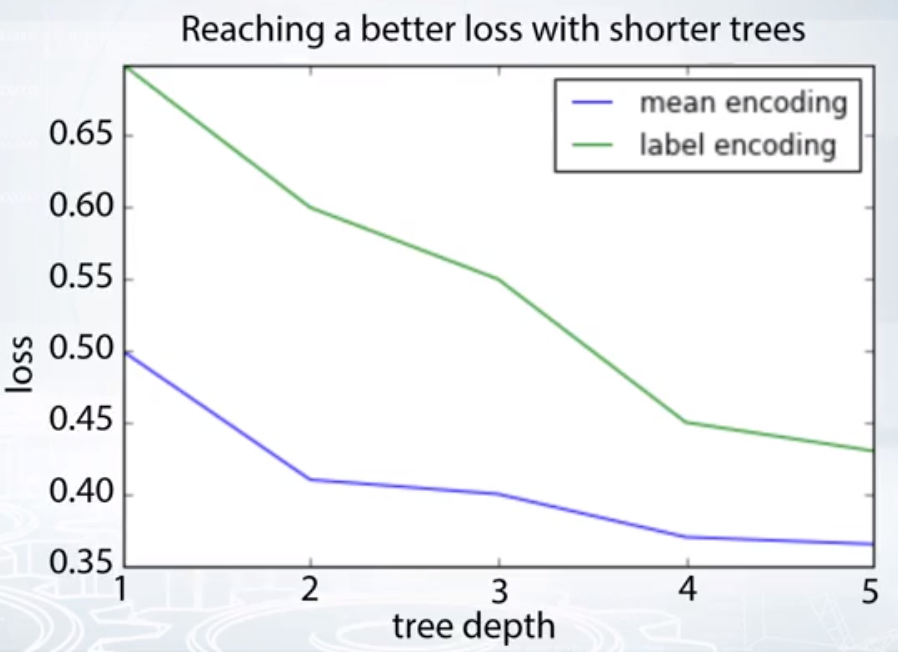
Привет всем. В этом разделе мы рассмотрим очень мощную технику - mean encoding. На самом деле у него есть несколько имен. Некоторые называют это likelihood encoding, некоторые target encoding, но в этом курсе мы будем называть просто mean encoding. Общая идея этого метода состоит в том, чтобы добавить новые переменные, основанные на переменной, которая была изначально. В простейшем случае мы кодируем каждый уровень категориальной переменной с соответствующим целевым средним. Давайте посмотрим на следующий пример. Здесь у нас есть некоторая задача бинарной классификации, в которой мы имеем категориальную переменную - город. И, конечно же, мы хотим численно его кодировать. Наиболее очевидный способ, которым обычно пользуются люди - это label encoding.



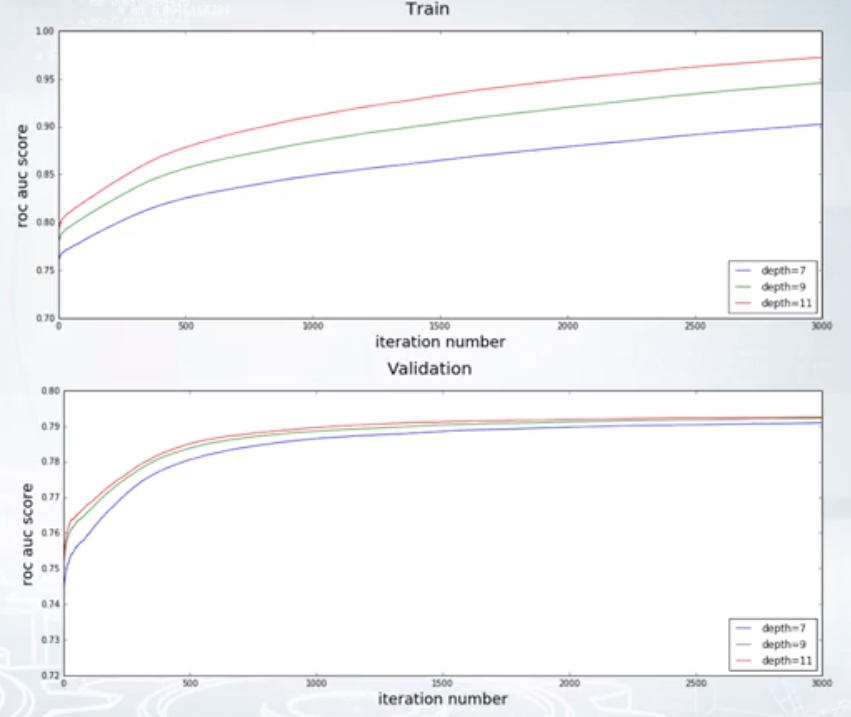
Это то, что мы имеем во втором столбце. Mean encoding выполняется по-другому, посредством кодирования каждого города с соответствующей средней целью. Например, для Москвы у нас есть пять наблюдений с 3 нулями и 2 единицами. Таким образом, мы кодируем ее числом 0,4(2/5). Точно так же мы делаем для остальных городов, это довольно просто. То, что здесь описано является идеей очень высокого уровня. Существует огромное количество подводных камней, которые нужно преодолеть в настоящем соревновании. Пока мы углубились в детали, просто имейте это в виду. Почему это работает? Представьте, что наш набор данных намного больше и содержит сотни разных городов. Что ж, давайте попробуем, конечно, очень абстрактно сравнить mean encoding с label encoding. Мы строим гистограммы для класса 0 и класса 1. В случае label encoding мы всегда получим полное и случайное изображение, потому что нет логического порядка, но когда мы используем mean encoding, классы выглядят более отделимыми. График выглядит отчасти сортированным.



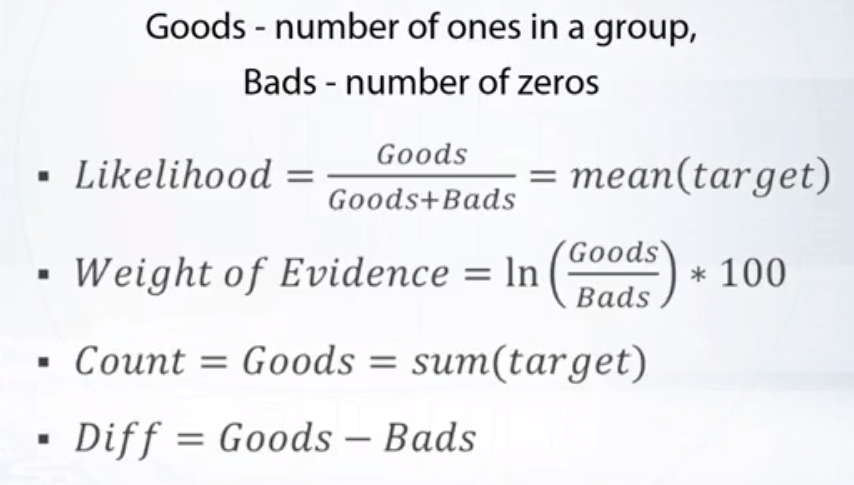
Оказывается, это качество сортировки mean encoding весьма полезно. Помните, что является самым популярным и эффективным способом решения проблемы машинного обучения? Это градиентный бустинг на деревьях(XGBoost или LightGBM). Одним из немногих недостатков является неспособность обрабатывать категориальные переменные высокой мощности. Деревья имеют ограниченную глубину. С mean encoding, мы можем компенсировать это, мы можем достичь лучших потерь с более короткими деревьями. Потери на кросс-валидации может выглядеть следующим образом.



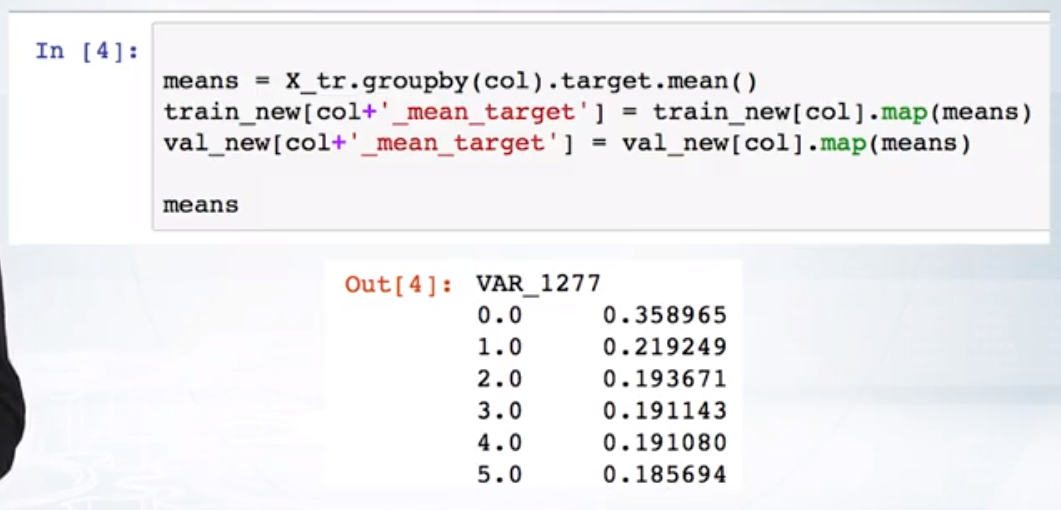
В общем, чем более сложнее и нелинейна зависимость между переменной и таргетом, тем эффективнее mean encoding. Далее в этом разделе вы узнаете, как создавать mean encodings. На самом деле есть много способов. Также имейте в виду, что мы использовали только задачу классификации в качестве примера. Мы можем использовать и в других задачах. Основная идея остается прежней. Несмотря на простоту идеи, вы должны быть очень осторожны с проверкой. Она должна быть безупречной. Это, наверное, самая важная часть. Понимание правильной проверки без ссылок(linkless) также является основой для размещения(basis for stacking). Последнее, но не менее важное, это расширения. Есть бесчисленные возможности для получения новых переменных из целевой переменной. Иногда они дают значительные улучшения для ваших моделей. Давайте начнем с некоторых характеристик наборов данных, которые указывают на полезность mean encoding. Наличие категориальных переменных с большим количеством уровней уже является хорошим индикатором, но нам нужно пойти немного глубже. Давайте посмотрим на результаты обучения XGBoost для соревнования Springleaf. Я запустил три модели с различной глубиной, 7, 9 и 11. Результаты обучения находятся на верхнем графике. Результаты теста находятся внизу.



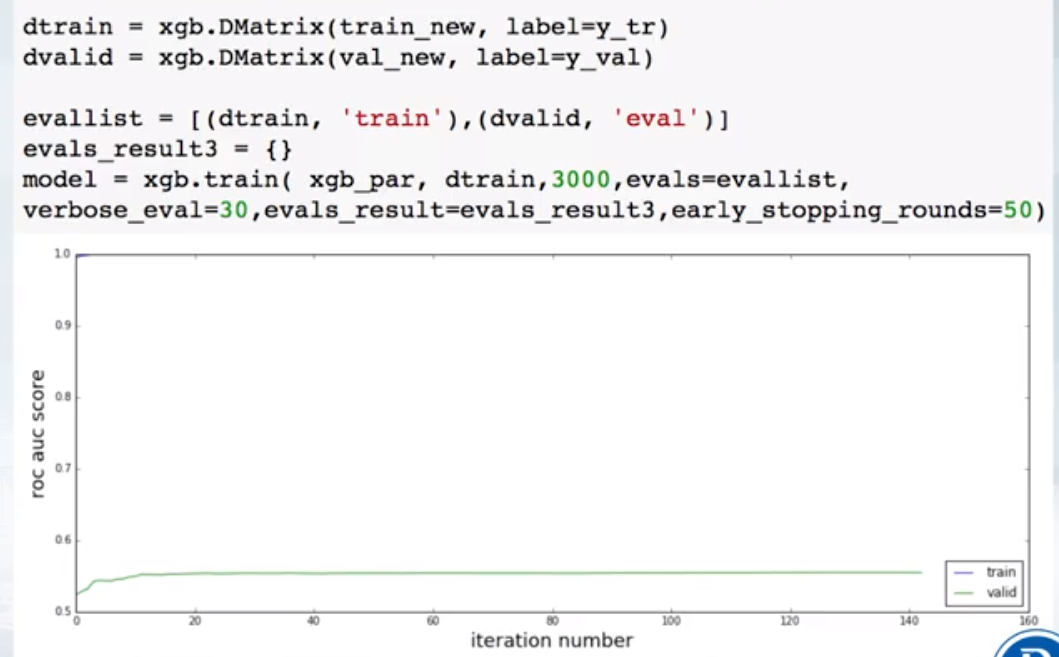
Как вы можете видеть, с увеличением глубины деревьев наш score на train становится все лучше и лучше, почти идеальным, и это нормальная часть. Но на самом деле мы не переобучаемся, и это странно. Наша оценка валидации также увеличивается, это признак того, что деревьям требуется огромное количество разбиений для извлечения информации из некоторых переменных. And we can check it for mortal dump. Оказывается, что некоторые переменные имеют огромное количество точек разделения, например 1200 или 1600, и это много. Наша модель пытается по-разному относиться ко всем этим категориям, и они также очень важны для прогнозирования таргета. Мы можем помочь нашей модели с помощью mean encodings. Существует несколько способов вычисления кодировок. Первый - тот, который мы обсуждали до сих пор. Просто принимая среднее значение целевой переменной. Другой популярный вариант - взять натуральный логарифм этой величины, это называется weight of evidence. Или вы можете рассчитать все числа равные единице. Или взять разницу между количеством единиц и количеством нулей. Все это переменные параметры. Все это доступные опции.



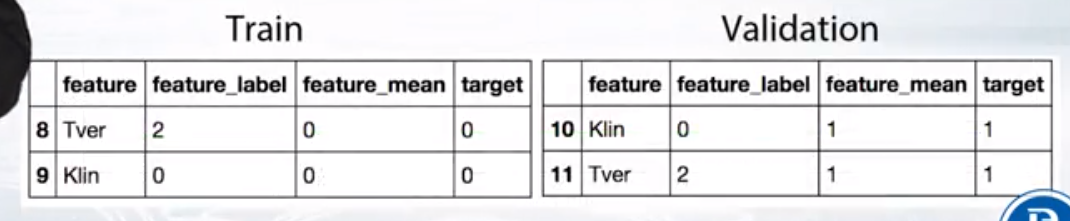
Теперь давайте на самом деле построим переменные. Мы сделаем это на датасете соревнования Springleaf. Предположим, что мы уже разделили данные для обучения и проверки, датафреймы X\_tr и X\_val. Этот фрагмент кода показывает, как построить mean encoding для произвольного столбца и отобразить его в новый датафрейм, train\_new и val\_new. Мы просто группируем по этому столбцу и используем target как для map. Результатом является переменная means, формат которой Pandas Series.



It is then mapped to tree and validation data sets by a map operator. После того, как мы повторили этот процесс для каждого столбца, мы можем обучить модель XGBoost на новых данных. Но что-то определенно не правильно, после нескольких эпох обучения AUC составляет почти 1, в то время как при проверке, оценка устанавливает около 0,55, что практически является шумом.



Это явный признак ужасного переобучения Я объясню, что случилось через несколько минут. Прямо сейчас я хочу указать, что по крайней мере мы валидировали правильно. Мы разделили обучение и тест и использовали все данные обучения для оценки mean encoding. Если, например, мы бы оценили mean encoding до разделения проверки поезда, то мы не заметили бы такого переобучения. Теперь давайте выясним причину переобучения.



Довольно часто можно получить результаты, как в примере, таргет 0 на обучении и таргет 1 на проверке. Mean encoding превращается в идеальную переменную для таких категорий. Вот почему мы сразу получаем очень хорошие результаты на тренировке и проваливаемся на проверке. До сих пор мы поняли концепцию mean encoding и рассмотрели некоторые тривиальные примеры, которые, очевидно, не могут использовать mean encoding на практике. Сначала нам нужно разобраться с переобучением, нам нужна какая-то регуляризация. И я расскажу вам о разных методы в следующем видео.