ML Design Pattern #5: Repeatable sampling

Use a well-distributed column to split your data into train/valid/test

Lak Lakshmanan Nov 8, 2019 · 3 min read

An occasional series of design patterns for ML engineers. Full list here.

Many machine learning tutorials will suggest that you split your data randomly into training, validation, and test datasets:

```
df = pd.DataFrame(...)

rnd = np.random.rand(len(df))
train = df[ rnd < 0.8  ]
valid = df[ rnd >= 0.8 & rnd < 0.9 ]
test  = df[ rnd >= 0.9 ]
```

The problem is that this fails in many real-world situations. The reason is that it is rare that the rows are independent. For example, if you are training a model to predict flight delays, the arrival delays of flights on the same day will be highly correlated with each other. This is called *leakage*, and it is an important problem to avoid when doing machine learning.

Use the Farm Fingerprint hashing algorithm on a well-distributed column to split your data into train/valid/test

The solution is to split the dataset based on the date column:

```
SELECT
  airline,
  departure_airport,
  departure_schedule,
  arrival_airport,
  arrival_delay
FROM
  `bigquery-samples`.airline_ontime_data.flights
WHERE
  ABS(MOD(FARM_FINGERPRINT(date), 10)) < 8 -- 80% for TRAIN
```

Besides solving the original problem (data leakage), this also gives you repeatability:

1. FARM_FINGERPRINT is an open-source hashing algorithm that is implemented consistently in C++ (and hence: Java or Python) and in BigQuery SQL.
2. All the flights on any given date will belong to the same split — train, valid, or test. This is repeatable regardless of things like the random seed.

Choosing split column

How do you choose the column to split on? The date column has to have several characteristics for us to be able to use it as the splitting column:

1. Rows at the same date tend to be correlated — again, this is the key reason why we want to ensure that all rows on the same date are in the same split.

2. Date is not an input to your model (features extracted from date such as dayofweek or hourofday can be inputs, but you can't use an actual input to split because the trained model will then not have seen 20% of the possible input values).

3. There have to be enough date values. Since you are computing the hash and finding the modulo with respect to 10, you need at least 10 unique hash values. The more unique values you have, the better, of course. To be safe, shoot for 3–5x the denominator for the modulo, so in this case, you want 50 or more unique dates.

4. The label has to be well-distributed among the dates. If it turns out that all the delays happened on Jan. 1 and the rest of the year, there were no delays, this wouldn't work since the split datasets will be skewed. To be safe, look at a graph and make sure that all three splits have a similar distribution of labels by departure delay or some other input value. You can automate this using the Kolomogorov-Smirnov test.

Variation 1: Single query

You don't need three separate queries to generate training, validation, and test splits. You can do it in a single query as follows:

```
CREATE OR REPLACE TABLE mydataset.mytable AS

SELECT
  airline,
  departure_airport,
  departure_schedule,
  arrival_airport,
  arrival_delay,
  CASE(ABS(MOD(FARM_FINGERPRINT(date), 10)))
      WHEN 9 THEN 'test'
      WHEN 8 THEN 'validation'
      ELSE 'training' END AS split_col
FROM
  `bigquery-samples`.airline_ontime_data.flights
```

Variation 2: Random split

What if you want a random split, but just need repeatability? In that case, you can simply hash the row data itself. Here's an easy way to do that:

```
SELECT
  airline,
  departure_airport,
  departure_schedule,
  arrival_airport,
  arrival_delay
FROM
  `bigquery-samples`.airline_ontime_data.flights f
WHERE
  ABS(MOD(FARM_FINGERPRINT(TO_JSON_STRING(f), 10)) < 8
```

Note that if you have duplicate rows, then they will always end up in the same split. If this is a concern, add a unique id column to your SELECT query.