DEMONSTRATION OF BALLET: A FRAMEWORK FOR OPEN-SOURCE COLLABORATIVE FEATURE ENGINEERING

Micah J. Smith [1] **Kelvin Lu** [1] **Kalyan Veeramachaneni** [1]

ABSTRACT

Feature engineering is a critical part of end-to-end learning pipelines in many practical supervised learning settings. While the most predictive features often build off of diverse domain expertise and human intuition, rarely are more than a small handful of data scientists and researchers involved in this process. Ballet addresses this problem by providing a framework for scaling feature engineering collaborations in an open-source setting. In our approach, collaborators incrementally submit patches containing standalone feature definitions to a central source code repository. Our framework provides functionality for composing the separate features into an executable end-to-end pipeline, evaluating feature submissions in a streaming fashion, and automating project management tasks for maintainers. In this demonstration, audience participants will collaborate in real-time in a feature engineering task on a complex, real-world dataset.

1 INTRODUCTION

The open-source model for software development has led to successful, large-scale collaborations in building software frameworks, software systems, chess engines, scientific analyses, and more. However, data science, and in particular, predictive machine learning (ML) modeling, has not benefited from this development paradigm. Predictive modeling projects — where the output of the project is not a software library but rather a trained model capable of serving predictions for new data instances — are rarely developed in open-source, and when they are, they rarely have more than a handful of collaborators.

Drawing inspiration from successful collaboration paradigms in software engineering, we proposed a new framework for large-scale data science collaborations (Smith et al., 2020). Our approach is based on decomposing the data science process into modular patches — standalone units of contribution — that can then be intelligently combined. Prospective contributors work in parallel to write patches and submit them to an open-source repository. Our software framework provides the underlying functionality to merge high-quality contributions and compose the accepted contributions into a single product.

We instantiated these ideas in *Ballet*[1], a software framework for collaborative feature engineering on tabular data (Lu, 2019; Smith et al., 2018; 2020). Feature engineering is the process of writing code to transform raw variables into feature values that can be used as input to an ML model. We start from the insight that feature engineering can be represented as a dataflow graph over individual features. We structure code that extracts a group of feature values as an individual patch, calling these *logical features*. An example logical feature written using our framework is shown in Listing 1.

A potential project collaborator begins developing a new feature (defining a `Feature` object) in their development environment of choice. When they are satisfied with its performance, they propose it for inclusion in the pipeline using one of several provided interfaces. The most direct approach is using the `ballet` CLI. The submitted features are evaluated by a CI service in a specialized procedure that checks the integrity of the project structure, conformance of the proposed feature to the required feature interface, and finally evaluates the ML performance of the feature using a streaming feature acceptance algorithm. According to this result, the submitted features are marked as accepted or rejected. Accepted features are automatically merged to the repository by `ballet-bot`, a GitHub app, which may further automatically prune newly redundant features from the source tree. The user can monitor the output of this process. If their feature is accepted, they can move on to their next idea; if rejected, they can review the diagnostic information and try to improve their idea, or abandon it entirely. This process is illustrated in Figure 1.

[1]Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA. Correspondence to: Micah J. Smith <micahs@mit.edu>.

Proceedings of the 3^{rd} MLSys Conference, Austin, TX, USA, 2020.

[1]https://github.com/HDI-Project/ballet

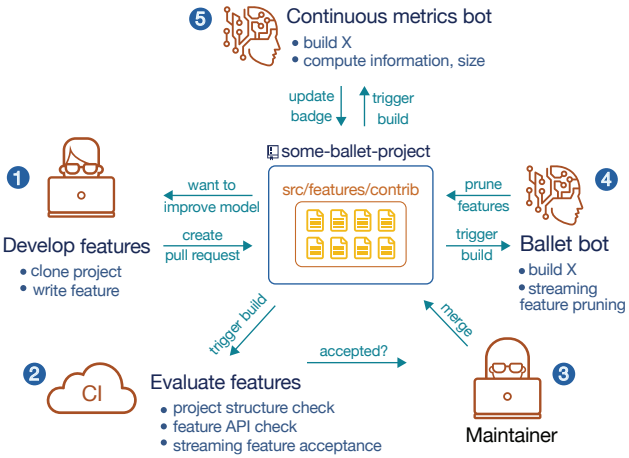

Figure 1. The development lifecycle of a new feature, from proposal to impact on the model.

2 DEMONSTRATION

In this demonstration, we will invite the audience to collaborate in real-time on a feature engineering task. We will present a raw dataset that requires significant feature engineering before a usable feature matrix can be input to a learning algorithm. We will supply two separate problems: the first is a real-world house price prediction problem that is currently a live collaboration[2] and the second is the Fragile Families Challenge (Fragile Families Challenge) dataset, predicting eviction incidence for disadvantaged children given a complex survey dataset. If possible, we may substitute an ongoing Kaggle challenge at the time of the conference.

On individual provided laptops, audience members will be invited to contribute a new feature to a live collaboration. They will click a link to spin up a Binder notebook and write a new feature for the problem (or copy a pre-written, example feature for expediency), i.e. write Python code in a notebook cell using our framework libraries to define a single `Feature` object, as in Listings 1 and 2. They will be able to submit their feature to the project using an in-notebook interface that will extract the feature source code, authenticate with GitHub, and submit a pull request to the collaboration project under their account.

Separately, we will visualize the progress of the ongoing collaboration in a simple dashboard, showing metrics such as the number of features submitted/accepted/rejected, the number of unique collaborators, the mutual information of the extracted feature matrix with the target, and the performance of a simple AutoML model trained on the extracted feature matrix.

[2] https://github.com/HDI-Project/ballet-predict-house-prices

```python
from ballet import Feature
from ballet.eng import ConditionalTransformer
import numpy as np
from sklearn.impute import SimpleImputer

input = 'Lot Area'
transformer = [
    ConditionalTransformer(
        lambda ser: ser.skew() > 0.75,
        lambda ser: np.log1p(ser)),
    SimpleImputer(strategy='mean'),
]
name = 'Lot area unskewed'
feature = Feature(input=input,
    ↪  transformer=transformer, name=name)
```

Listing 1: An example of a user-submitted logical feature in Ballet that conditionally unskews the "lot area" variable by applying a log transformation only if skew is present in the training data and then mean-imputing missing values. This feature leverages the rich set of feature engineering primitives provided in the `ballet.eng` library.

```python
from ballet import Feature
from ballet.eng import NullFiller,
    ↪  SimpleFunctionTransformer
import numpy as np

def calc_garage_per_car(df):
    return df["Garage Area"] / df["Garage Cars"]

input = ["Garage Area", "Garage Cars"]
transformer = [
    SimpleFunctionTransformer(calc_garage_per_car),
    NullFiller(isnull=np.isinf, replacement=0.0),
    NullFiller(replacement=0.0),
]
name = "Garage area per car"
feature = Feature(input=input,
    ↪  transformer=transformer, name=name)
```

Listing 2: A user-submitted logical feature that calculates the garage area per car and cleans infinite and missing values.

REFERENCES

Fragile Families Challenge. Fragile families challenge, 2017. URL http://www.fragilefamieschallenge.org/.

Lu, K. Feature engineering and evaluation in lightweight systems. M.eng. thesis, Massachusetts Institute of Technology, 2019.

Smith, M. J., Lu, K., and Veeramachaneni, K. Ballet: A lightweight framework for open-source, collaborative feature engineering. In *Workshop on Systems for ML at NeuRIPS 2018*, 2018.

Smith, M. J., Lu, K., and Veeramachaneni, K. Enabling open-source collaborative data science development with the ballet framework. Preprint, 2020.