Tribuo analisi – secondo assegnamento Ids

Introduzione

Il progetto che abbiamo scelto di analizzare e’ tribuo.

Tribuo e’ una libreria open source scritta in Java dal team di ricercatori di Oracle Labs’ Machine Learning. E’ stato usato internamente ai laboratori Oracle per svariati anni, per poi essere rilasciato in licenza open source da Oracle nell’ agosto 2020 su github sotto una licenza Apache 2.0.

Tribuo nasce con l’obbiettivo di fornire una libreria di Machine Learning in ambiente Java che sia in linea con le funzionalita’ e bisogni richiesti per lo sviluppo di grandi sistemi software.

La prima versione 1.0 di Tribuo viene scritta nel 2016 e rilasciata nell’autunno dello stesso anno.

Le versioni open source, invece, rilasciate a partire da agosto 2020, sono:

* v4.0.0 (13 agosto 2020)
* v4.0.1 (1 settembre 2020)
* v4.0.2 (5 novembre 2020)
* v.4.1.0 (26 maggio 2021)

Analisi produzione

1. Introduzione

L’analisi del processo di produzione del software si e’ basata sulle informazioni ricavate dalla repository oracle/tribuo di GitHub, dal sito, dalla documentazione ufficiale di Tribuo ed da alcuni articoli relativi al progetto.

Abbiamo inoltre contattato Adam Pocock, lead developer del progetto, che ha fornito ulteriori informazioni riguardo l’organizzazione del team e l’evoluzione di Tribuo.

1. Team e organizzazione

Il team di sviluppo e’ formato da un gruppo di ricercatori in Machine Learning che hanno come focus primario research & data science.

Tribuo nasce inizialmente come progetto di due persone e cresce nel tempo coinvolgendo altri membri vista la sua utilita’ in altri progetti del gruppo di ricerca.

L’ultima release di Tribuo ha coinvolto 6 sviluppatori interni e un paio di sviluppatori esterni. Molti sviluppatori di Tribuo lavorano contemporaneamente ad altri progetti interni ad Oracle, quindi, il lavoro degli sviluppatori interni e’diviso sulla base delle aree di maggior competenza (decidere se citare philip //TODO ) e sugli impegni legati ai progetti di ricerca a cui attualmente questi stanno lavorando.

I meeting giornalieri sono previsti solo nei momenti in cui il team sta lavorando all’implementazione e al design di features che riguardano una grossa parte del progetto. Non sono previsti meetings ricorrenti ma solo in caso di necessità. I vari membri della squadra si coordinano utilizzando Slack.

I contributors esterni possono collaborare al progetto tramite feedback su GitHub, segnalando bug, proponendo bug fixes, features e scrivendo documentazione aggiuntiva.

1. Tribuo development

Tribuo nasce dalla necessita’ di avere uno strumento che supporti i lavori di data science all’interno di Oracle, per questo Tribuo non segue necessariamente un processo standard di sviluppo software.

Le priorità e gli obbiettivi di Tribuo sono quindi definite sulla base:

* delle necessità interne dei ricercatori di Oracle
* di ciò che la community vuole
* di ciò che gli sviluppatori pensano sia utile
* degli elementi presenti nel backlog del progetto gestito tramite JIRA e GitHub

Tribuo aderisce al concetto di *semantic versioning.* L’implementazione di alcuni dei punti sovrastanti potrebbe quindi essere rimandata alla prossima major release, mentre altri

Tribuo conforms to semantic versioning, so some of these features may be held back until the next major release, while others will land in minor releases.

Una release principale si concentra sull’implementazione o rework di una funzionalità principale accompagnata da una serie di piccoli bug fixes e features minori presenti nel backlog del progetto.

Tribuo approximates semantic versioning. Major version bumps can break the backwards compatibility of both the code and serialized models (though we hope to fix the latter by moving to a new serialization architecture). Provided that it's an upwards compatible change, minor version bumps can add new features, improve performance (both statistically and in terms of runtime/memory usage), and add new functionality to existing algorithms. Patch releases fix bugs in existing versions and resolve security issues when they are discovered. Patch releases may also add small methods or classes if they are required to fix bugs.

Tramite dei meeting coordinativi iniziali vengono individuate la feature principale su cui basare la prossima release e le features minori da implementare, presenti nel backlog, selezionate secondo una priorità decisa dal team. Tutto ciò viene inserito in un documento accessibile a tutto il gruppo sulla base della quale verrà effettuato lo sviluppo software.

Non seguendo in maniera precisa una metodologia di sviluppo software, né Agile né Plan-based, lo sviluppo non è scandito in fasi ben delineate, ma segue il documento (precedente), il quale viene costantemente aggiornato dagli sviluppatori e gestito tramite Slack.

Come precedentemente detto, il compito di ogni membro del team, a differenza di un modello Plan-based, non è limitato ad un unico aspetto del progetto, ma può includere attività diverse (potenzialmente anche variabili nel tempo) in relazione ai propri impegni legati ai progetti di ricerca.

La coordinazione generale di questi compiti viene supervisionata dal lead developer del progetto.

Come programmano (con software)

Test driven development

Branch separati e poi uniti, pair programming

- Tribuo development is driven by a few main processes: features that we need for a current data science collaboration, a backlog of features from previous collaborations or requests, and occasionally implementing something because someone in our group read a paper that had an interesting technique or algorithm in. The backlog we track in JIRA or in GitHub issues depending on the source of the feature/bug report, and things are worked on based on a rough prioritisation done by our team. We don’t have standup meetings, scrum or xp, nor do we follow a waterfall methodology. Our most recent release was a mixture of small performance improvements or new features and a complete rewrite of our TensorFlow package. The next release will likely have ONNX export as its big feature and several smaller features from our backlog/roadmap. Those features tend to be developed in separate branches and are merged when the feature is done.

- We don’t strictly follow test driven development. It depends on the nature of the feature being developed. In some cases we will follow something like TDD, where we’ll write a new test for the specific functionality (or to exercise a bug) and then write the code to fix that test. In other cases we’ll write the tests after most of the code is written to validate it’s behaviour. Sometimes we’re simply expanding the test coverage for an existing part of the library, though that’s usually in preparation for a refactor that will touch that code.

- We have used JaCoCo to measure code coverage of tests, and we use the coverage tooling built into the IDEs that the development team use (most of us use IntelliJ but not all). We run static analysers to catch bugs as well.

- My day tends to involve a meeting for one of the research projects I’m involved in, some catch up with the rest of our group, and then programming or research. We tend not to have Tribuo specific group meetings unless we’re working on the design of a large cross-cutting feature (e.g. the provenance system), work is usually co-ordinated over Slack. We tend not to do pair programming at the moment due to the pandemic, though we did occasionally do that when puzzling out something particularly tricky before it. As well as leading Tribuo’s development I also maintain the ONNX Runtime Java API and I help run the TensorFlow SIG-JVM group which builds TensorFlow-Java, along with working on some ML research projects.

Thanks,

Adam

Tribuo 4.1 analisi completa (CK, LOC , CC , metriche )

Confronto grafico con versioni precendenti

Conclusione