Автоматизация процесса назначения разработчика в системах отслеживания ошибок

Черняев А. В., студент 4 курса кафедры системного программирования СПбГУ, chernyaev.arseniy@gmail.com

Аннотация

Определение разработчика, который будет заниматься новым сообщением о неисправности программного обеспечения - важная задача, которую необходимо решать в системе отслеживания ошибок. Для крупного проекта возможна ситуация, в которой в систему за день может приходить большое количество новых сообщений, что делает процесс ручного определения очень трудоемким.

В данной работе рассматривается возможность применения методов машинного обучения для автоматизации данного процесса на примере набора данных об исключительных ситуациях проекта ReSharper.

Введение

При создании современного программного обеспечения зачастую возникает потребность в использовании специальных средств, позволяющих программистам контролировать ход разработки, получать отзывы от пользователей и поддерживать связь с другими членами команды, особенно в больших распределенных проектах.

Одним из таких средств является система отслеживания ошибок. Эта система позволяет команде разработчиков учитывать и контролировать ошибки, найденные в ходе эксплуатации программного обеспечения. Каждый, кто имеет доступ к системе, может создать специальный запрос, затем внутри команды этот запрос будет обработан, будет выделен выделен человек или команда, ответственные за решение поставленной задачи или исправление допущенных исключительных ситуаций. Запрос получит уникальный номер, состояние, символизирующее статус работы, и может служить местом обсуждения проблемы внутри команды или между командой разработчиков и пользователем.

В связи с выше сказанным, в системах отслеживания ошибок можно выделить ряд прикладных задач, позволяющих упростить процесс работы создателей программного обеспечения. Одна из таких задач -

задача определения ответственного за устранения неисправности, описанной в сообщении. В больших проектах возможны ситуации, когда ежедневно к отслеживанию добавляются тысячи новых ошибок и предложений к улучшению, требующих назначения ответственного. В особенности это касается отчетов об исключительных ситуациях - во многих крупных проектах имеется возможность автоматической отправки такого типа сообщений, из-за чего система отслеживания ошибок будет подвергаться огромной нагрузке.

Очевидно, что ручное определение разработчика в таком случае может занимать значительный объем времени, даже в случае, если человек, обрабатывающий запросы, обладает достаточными знаниями о всех деталях разрабатываемого программного обеспечения.

Таким образом, можно сделать заключение, что необходима автоматизация данного процесса, позволяющая переложить большую часть работы с разработчика на машину. Методы машинного обучения, позволяющие решать многие задачи классификации и кластеризации, могут быть эффективными и для проблемы классификации отчетов об исключительнх ситуациях.

Обзор

В качестве набора данных используются отчеты об исключительных ситуациях проекта ReSharper, расположенного в системе отслеживания ошибок YouTrack за все время его существования. Данный набор содержит примерно 5974 размеченных отчета (они помечены как исправленные, и для них известен ответственный за них разработчик), а также примерно 34000 уникальных неразмеченных отчета. Для обучения классификаторов использовалось 80% размеченных данных, выбираемых произвольно, оставшиеся 20% использовались для тестирования.

Методы машинного обучения применялись к области классификации сообщений об ошибках и раньше. Большинство подходов основывались на принципе рассмотрения задачи как более общей задачи классификации текста, применялись различные классификаторы: наивный байесовский классификатор [4] [2], метод опорных векторов [1] [2], решающие деревья [2].

Проблема применимости данных подходов к конкретной задаче заключается в том, что они основываются на идее обработки текста сообщения, который в случае отчетов об исключительных ситуациях яв-

ляется менее информативным: многие элементы выборки имеют один и тот же текст, но при этом относятся к абсолютно разным разработчикам. Основным источником для определения сути проблемы в таком случае является стек вызовов, неизменно прилагающийся к отчету. По этой причине рассмотрение другого подхода к подготовке данных может улучшить результаты.

Также стоит отметить, что существующие исследовательские работы не рассматривают возможности применения алгоритмов ансамблирования классификаторов, которые могут увеличить точность предсказания, а также не рассматривают возможность игнорирования предсказания в случае низкой степени уверенности классификатора, что позволит уменьшить вероятность ошибки за счет небольшого количество данных, которые придется классифицировать вручную.

В качестве вспомогательных библиотек для исследования использовались следующие библиотеки для языка Python: scikit-learn [5], выбранная в силу большого количества оптимизированных алгоритмов обучения и подготовки данных, включая возможность работы с разреженными матрицами, а также библиотека Xgboost [3], содержащая реализацию метода градиентного бустинга.

Ход работ

Для решения поставленной задачи рассматривались различные способы подготовки данных:

- Обработка текста отчета
- Разбиение стека вызовов на слова
- Добавление дополнительных характеристик сообщения к рассмотрению
  - Дата создания
  - Подсистема проекта

Рассматривались различные способы представления текстовых данных в виде числового вектора, а также способы уменьшения размерности полученных числовых векторов.

К подготовленным данным применялись различные алгоритмы классификации:

- Логистическая регрессия

- Гауссовский наивный байесовский классификатор
- Метод ближайших соседей
- Метод опорных векторов с линейным ядром
- Решающие деревья
- Случайный лес
- AdaBoost
- Градиентный бустинг (xgboost)

Дополнительно рассматривалась возможность использования классификатора, способного игнорировать результаты некоторых предсказаний, с целью уменьшения величины ошибки, а также некоторые методы частичного обучения, позволяющие дополнительно использовать большое количество неразмеченных данных.

Заключение

В данной статье была рассмотрена задача автоматизированного назначения ответственного за ошибку для набора отчетов об исключительных ситуациях проекта ReSharper. Были рассмотрены различные способы преобразования данных, была рассмотрена возможность применения различных классификаторов.

На основании полученных данных был выбран классификатор на основе метода градиентного бустинга библиотеки xgboost, как обладающий лучшими показателями усредненной по всем классам F_1-меры, выбранной в качестве оценки точности - при помощи него удалось достить значения в 0.557. При этом данный классификатор также показал лучшие результаты при измерении процента правильно предсказанных отчетов - он верно определял разработчика в 73.9% случаев. На основе данного метода был реализован анализатор, позволяющий определять разработчика, который вероятнее всего ответственен за данный отчет об ошибке. Для уменьшения вероятности выдачи неверного предсказания было добавлено игнорирование результатов предсказания, обладающих недостаточно высокими вероятностными показателями. Это позволило снизить вероятность ошибки до 15.7% от общего размера тестовой выборки при общем размере количества проигнорированных элементов в 17.3%.

Литература

[1] Anvik J. Hiew L. Murphy G.C. Who should fix this bug? // ICSE. — 2006. — P. 361–370.

[2] Bhattacharya P. Neamtiu I. Shelton C.R. Automated, highly-accurate, bug assignment using machine learning and tossing graphs // The Journal of Systems and Sofstware. — 2012. — Vol. 85, no. 10. — P. 2275–2292.

[3] Chen T. Guestrin C. XGBoost: A Scalable Tree Boosting System. — 2016. — URL: `http://arxiv.org/pdf/1603.02754v1.pdf` (online; accessed: 20.01.2016).

[4] Cubranic D. Murphy G.C. Automatic bug triage using text categorization // SEKE. — 2004. — P. 92–97.

[5] Scikit-learn: Machine Learning in Python / F. Pedregosa, G. Varoquaux, A. Gramfort et al. // Journal of Machine Learning Research. — 2011. — Vol. 12. — P. 2825–2830.