# Data Driven Project Management: Predicting the Development Time

Marko Prelevikj

Faculty of Computer and Information Science
University of Ljubljana
Ljubljana, Slovenia
Email: mp2638@stuent.uni-lj.si

*Abstract—*

## I. Introduction

The project manager's (PM) main task is to break down the project they manage into tasks which are manageable, not very complex and make a round unit which can be executed with the knowledge of a single person. Once the project is broken down into pieces the PM needs to answer the following questions for each task:

1) how much time the task is going to take to develop; and
2) which project member is the best fit for the task

In this paper we are focusing on the task of estimating the time required to develop a given task. We use data provided from a company's JIRA portal [1], which keeps record of the project's tasks. We made 4 different models of the time required to develop a task, where we changed the time span of development time, ranging from all time down to a maximum of 10 days. We evaluated the variations of the model using 4 distinct methods: Naive Bayes, Random Forests, SVM, all provided by Scikit-Learn [2], and additionally XGBoost [3]. In the end we uncover which are the most important features of the task which should be considered when estimating the development time for which we used the SHAP [4] method.

## II. Model data

The available information used to build the model was extracted from a company's JIRA portal [1]. We are taking into account all the tasks which have been resolved and have at some point been in development. The total number of such tasks amounts to 2935. All of the tasks are described by their categorical features: *type*, *priority*, *components* of the project they affect, and *labels* which are specific for the project. Due to their high cardinality, the values of the *components* and *labels* features have been filtered such that there are only left values which have at least 50 entries in the dataset. The filtered values are used in their one-hot-encoded form to reduce the complexity of the model. Additionally, we take into account the *number of linked issues* which is a discrete feature denoting the number of tasks which are in a relation with the observed task.

Due to improper usage of JIRA, there is some noise in the data which causes a strong bias towards low values of the predicted time to develop. To reduce this effect we have filtered out all tasks which have development time lower than $2h$. To further reduce the variance in our dataset we have decided to limit the upper bound of the development time. We have done so in 4 different stages to measure the effect of the variance on our model: 1) there is no upper bound, 2) the upper bound is 1 quarter ($\approx 90$ days), 3) the upper bound is 1 month ($\approx 30$ days), and 4) the upper bound is 10 days. The summary of the datasets is shown in Table I, where the first column associates to the general statics of all the data at our disposal. From Table I we can observe that the majority of the tasks ($\approx 83\%$ of them) had their development done in less than 10 days.

## III. Quality evaluation

We evaluated the variations of the model data using 4 distinct methods: Naive Bayes, Random Forests, SVM, and XGBoost. For each of these methods we did not tune any of the available parameters in order to improve the performance, but rather used recommended parameters available in the online documentation of the method implementations [2], [3]. The observed statistics of model quality are Mean Squared Error (MAE), Root Mean Squared Error (RMSE) and $R^2$. The evaluations of the listed methods are shown in Table V.

Our goal is to reduce the MAE and RMSE measures which express the fitness of the predicted values to the target values. We can observe this as the measures decrease as the variance in the data decreases. For each of the dataset variants we had at least one method which has a lower MAE value than the mean value of the data. And both of the measures are monotonously decreasing as the variance in the data decreases, which implies that they are correlated.

The $R^2$ measure expresses how much we have improved over the mean value of the model. The built model has failed to show that there are any improvement over the mean model. This can be seen from Table V where the $R^2$ model is either very close to $0$ or even negative in some of the cases. This suggests that we cannot get a very accurate prediction of the target variable, *hours of development*. However, the low $R^2$ value does not prevent us from exploring which features are the most important.

Figure 1. Figure



(a)



(b)
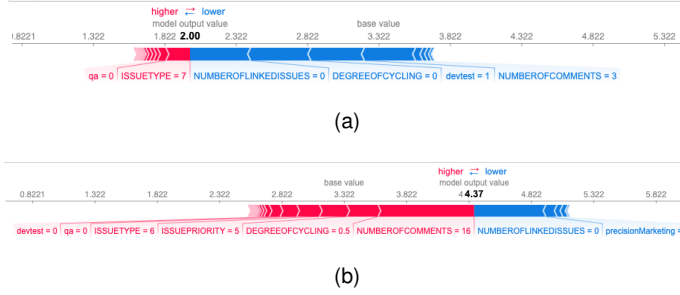
Table I
DATASET CHARACTERISTICS.

| Statistic | All | 1Q | 1M | 10D |
|---|---|---|---|---|
| count | 2935.000 | 2902.000 | 2775.000 | 2451.000 |
| mean | 205.252 | 149.210 | 96.165 | 55.762 |
| std | 678.103 | 293.573 | 132.257 | 57.019 |
| min | 2.000 | 2.000 | 2.000 | 2.000 |
| 25% | 15.000 | 15.000 | 13.000 | 11.000 |
| 50% | 49.000 | 48.000 | 44.000 | 33.000 |
| 75% | 148.000 | 143.000 | 121.000 | 83.000 |
| max | 15003.000 | 2260.000 | 754.000 | 228.000 |

Table II
PERFORMANCE OF THE LISTED METHODS ON THE VARIATIONS OF THE
DATASET. THE RESULTS ARE ORDERED BY DECREASING AMOUNT OF
VARIANCE IN THE DATA.

| Dataset | Method | RMSE | MAE | $R^2$ |
|---|---|---|---|---|
| All | XGBoost | **791.428** | 254.770 | -0.042 |
| | Naive Bayes | 948.649 | 420.036 | -0.498 |
| | Random Forest | 840.027 | 256.399 | -0.174 |
| | SVM | 792.344 | **191.267** | -0.045 |
| 1Q | XGBoost | **286.835** | 160.587 | 0.018 |
| | Naive Bayes | 577.520 | 389.602 | -2.982 |
| | Random Forest | 361.223 | 179.258 | -0.558 |
| | SVM | 302.772 | **124.272** | -0.095 |
| 1M | XGBoost | **138.034** | 92.616 | -0.048 |
| | Naive Bayes | 255.083 | 213.834 | -2.579 |
| | Random Forest | 178.280 | 109.402 | -0.748 |
| | SVM | 143.368 | **78.249** | -0.131 |
| 10D | XGBoost | **56.406** | 43.847 | 0.041 |
| | Naive Bayes | 120.824 | 106.642 | -3.398 |
| | Random Forest | 77.243 | 56.629 | -0.797 |
| | SVM | 60.460 | **41.976** | -0.101 |

## IV. MODEL EXPLAINABILITY

Write about feature importance and how to explain the made decisions.

## V. CONCLUSION

Quick recap of the problem and how we solved it. XG-Boost [3], SHAP [4].

REFERENCES

[1] "Jira - issue tracking software," https://www.atlassian.com/software/jira, accessed: 2020-05-28.
[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
[3] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.
[4] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee, "From local explanations to global understanding with explainable ai for trees," *Nature Machine Intelligence*, vol. 2, no. 1, pp. 2522–5839, 2020.