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*—**

1. 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 unit we are forecasting in: days or hours, and 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, XGBoost [2] and SVM. 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 [3] method.

2. Model data

The available information used to build the model has been extracted from a company's JIRA portal [1]. All 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 used the following discrete features: the *number of comments* each task has, the *number of linked issues*, and their *degree of cycling*.

Statistic	1	2	3	4
unit	days	hours	hours	hours
count	2935.000	2935.000	2771.000	2472.000
mean	8.536	205.252	95.229	57.273
std	28.265	678.103	130.036	59.080
min	0.000	2.000	2.000	2.000
25%	1.000	15.000	13.000	12.000
50%	2.000	49.000	44.000	33.000
75%	6.000	148.000	121.000	86.000
max	625.000	15003.000	719.000	240.000

Table 1. DATASET CHARACTERISTICS.

Due to improper usage of JIRA, there is some noise in the data, which causes a strong bias toward 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 3 different stages to measure the effect of the variance on our model: 1) there is no upper bound, 2) the upper bound is 30 days, and 3) the upper bound is 10 days. The summary of the datasets is shown in Table 1.

3. Testing model quality

Present different the results (MAE, RMSE, R2) obtained by different regressors. As the variance lowers, so do the metrics of quality of the models.

4. Model Explainability

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

5. Conclusion

Quick recap of the problem and how we solved it. XGBoost [2], SHAP [3].

References

[1] "Jira - issue tracking software," https://www.atlassian.com/software/jira, accessed: 2020-05-28.

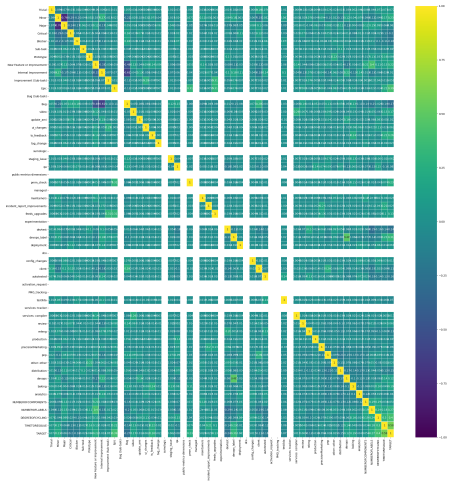


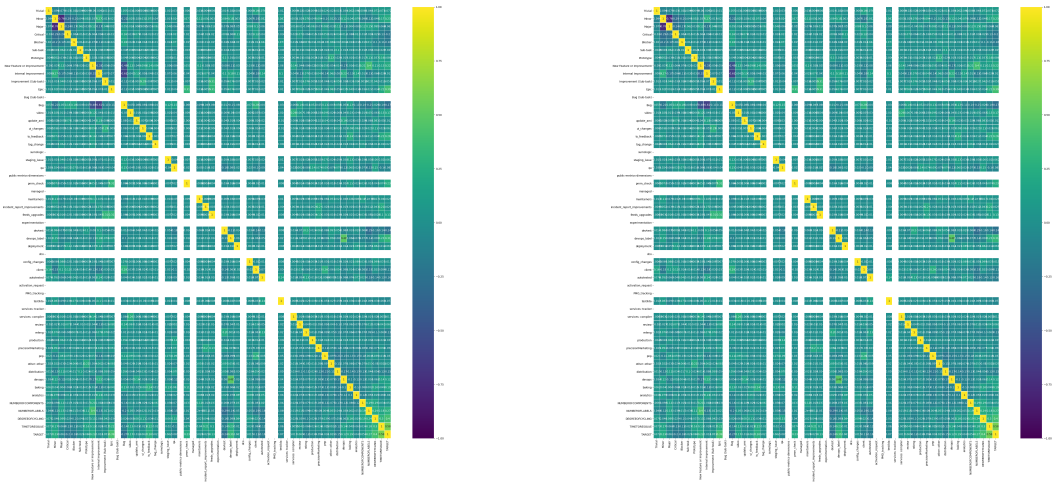
Figure 1. Simulation results for the network.

DataSet	Method	RMSE	MAE	R^2
1	boost	33.476	11.307	-0.073
	naive	53.129	40.126	-1.702
	forest	41.136	10.392	-0.620
	SVM	**33.079**	**7.988**	-0.047
1*	boost	**32.974**	10.581	-0.041
	naive	69.530	48.106	-3.628
	forest	34.364	9.683	-0.130
	SVM	33.020	**7.961**	-0.044
2	XGBoost	809.712	281.738	-0.091
	GaussianNB	890.796	381.685	-0.321
	RandomForest	1021.232	280.129	-0.736
	SVM	**792.962**	**193.070**	-0.046
2*	XGBoost	**792.284**	255.451	-0.045
	GaussianNB	941.595	411.579	-0.475
	RandomForest	897.283	267.935	-0.340
	SVM	792.589	**191.877**	-0.045
3	boost	**135.966**	89.841	-0.026
	naive	263.943	209.445	-2.865
	forest	170.992	103.386	-0.622
	SVM	144.008	**79.159**	-0.150
3*	boost	**135.020**	90.459	-0.011
	naive	279.474	232.818	-3.333
	forest	180.686	113.533	-0.811
	SVM	143.295	**79.605**	-0.139
4	XGBoost	**54.720**	41.941	0.050
	GaussianNB	122.290	105.834	-3.744
	RandomForest	75.167	54.669	-0.792
	SVM	59.238	**40.390**	-0.113
4*	XGBoost	**53.706**	42.230	0.085
	GaussianNB	124.868	109.788	-3.946
	RandomForest	78.889	56.857	-0.974
	SVM	58.114	**39.974**	-0.071

Table 2. PERFORMANCE OF DIFFERENT METHODS ON THE VARIATIONS OF THE DATASET. THE $*$ SYMBOL INDICATES THAT THE DATASET DOES NOT CONTAIN ALL THE INITIAL ATTRIBUTES, THUS IT IS MORE REALISTIC.

[2] 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.

[3] 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.



(a) Case I (b) Case II

Figure 2. Simulation results for the network.