AVRO Case for Enel

08.10.2020

Nico Biagioli

Agenda

- Introduction / business problem
- Executive summary
- Exploratory analysis
- Model and results
- Additional features
- Conclusion and next steps

Introduction / business problem

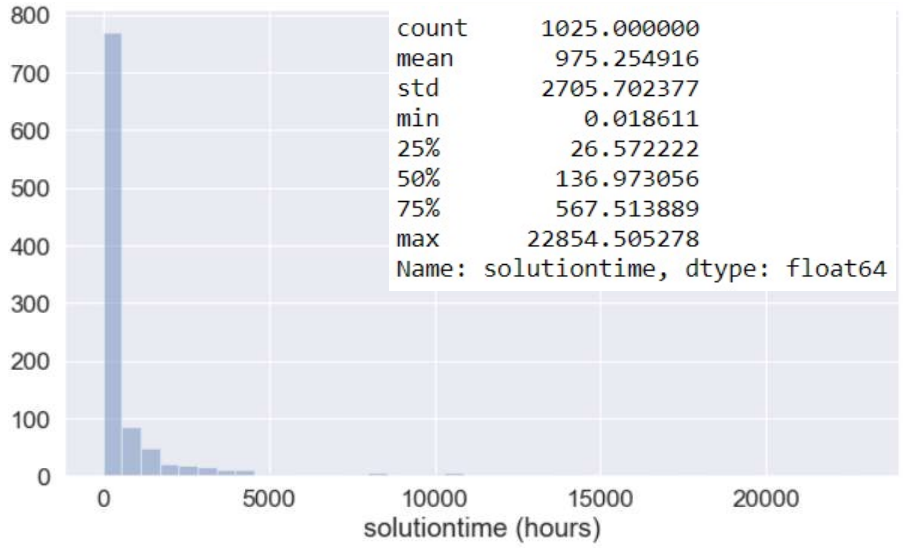
- Apache Avro is an open source data serialization system. The software issues, bugs and improvements are tracked by developers (https://issues.apache.org/jira/projects/AVRO/issues/AVRO).
- A dataset of recorded issues is available and it is required to develop a model able to predict the resolution time based on alert characteristics and tracking information.
- The expected benefit of such prediction capability for the software management business would probably translate in improvement of the incident tracking, delivery of update deployments and overall faster resolutions.

Executive summary

- Preliminary analysis showed highly skewed distributions for both independent (numerical and categorical) and target variables and unbalance.
- After an understanding and selection of the available features, the first approach was to use a Random Forest Regressor because of its ability in dealing with such tricky data characteristics.
- Particular attention was posed on the validation procedure, in order tackle little data, unbalance and model instability.
- First accuracy results were very poor, not improving the simplest target mean benchmark. A trivial resampling to solve unbalance was performed. It improved the coefficient of determination (not the absolute error) and needs further understanding and consolidation.
- Additional features were identified in the Avro website and extracted from the json file, but did not enhance the model.
- Further improvements and mitigations were identified and proposed. Time did not allow their implementation.

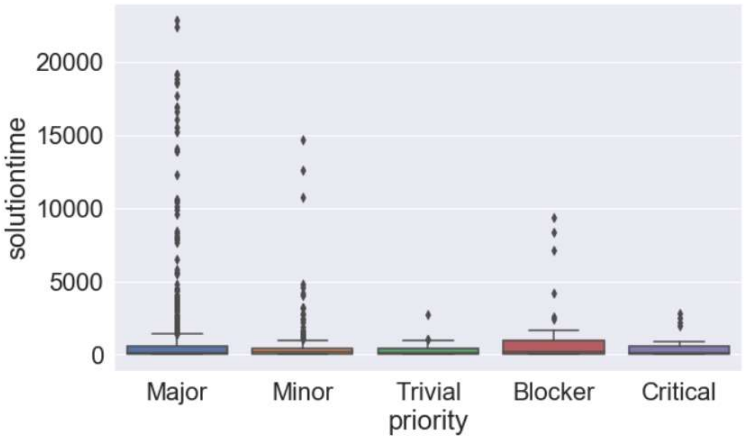
Exploratory analysis – target variable

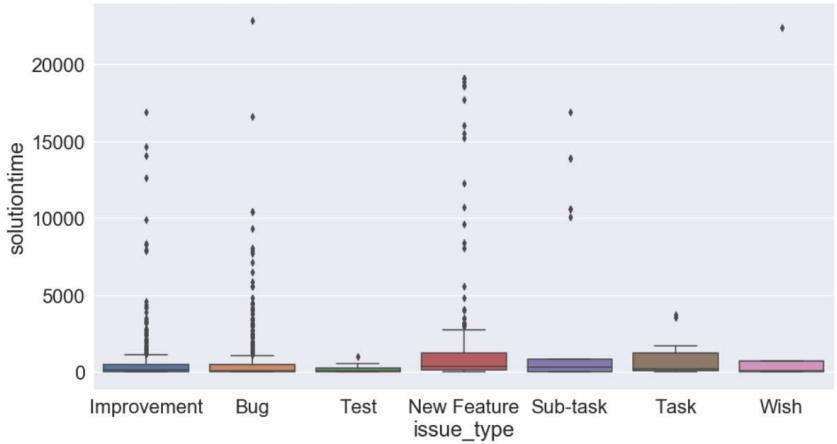
- Only datapoints where resolution time is available are considered, reducing the size from 1458 to 1025.
- The resulting dataset is unbalanced because there are 40 status=Resolved and 985 status=Closed. The status itself is not considered among the features, but the two groups have different characteristics and responses to other variables (details included in the report and notebook).
- Distribution positively skewed.



Exploratory analysis – categorical variables

'status', 'priority', 'issue_type', 'reporter', 'assignee'

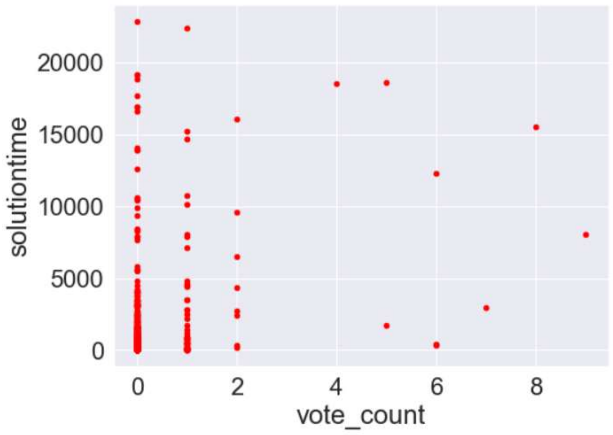


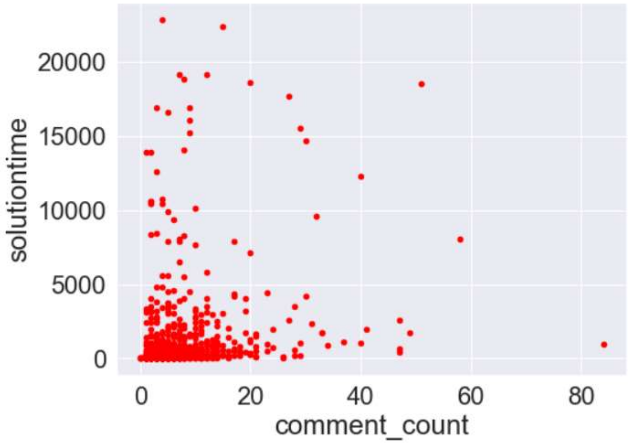


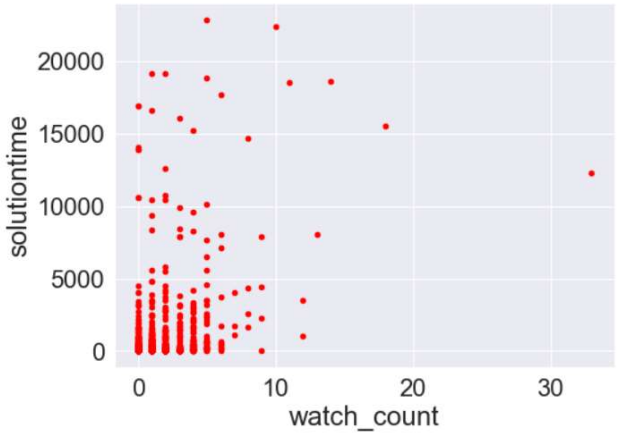
- 'Priority' and 'issue', the two variables that could intuitively impact the target, don't show significant trends with respect to the target and have several outliers.
- 'Reporter' and 'Assignee' mostly have very low value counts. Roughly, only 10% of the reporter/assignee counts is 10 or larger. With the same top 2 persons at the top of both count lists. Although this may suggest not much impact on the target, the two features are kept because model-based selection has shown some importance for the accuracy results.

Exploratory analysis – numerical variables

'vote_count', 'comment_count', 'description_length', 'summary_length', 'days_in_current_status'







- All numerical variables have right skewed distributions with exception of 'summary_length' (almost normal).
- 'vote_count', 'comment_count' and 'watch_count', that intuitively shall impact the target, have non-linear and unclear relationships with solutiontime.
- 'summary_length' and 'descrption_length' are discarded from the dataset because of lack of meaning with respect to the target, and based on iterative model-based feature selection.

Exploratory analysis – time variables

- ‘created’ & ‘resolutiondate’: used for solutiontime calculation.
- ‘updated’: date of the latest update in the issue thread. For Closed cases it corresponds to the latest change in status from Resolved to Closed. For Resolved (but not closed) cases the update date corresponds to the latest change and can included anything (status, comment, patch, attachment etc.). Thus ‘updated’ tells nothing on the resolution time and will be excluded from the features.
- ‘days_in_current_status’: (numerical variable but is listed here for convenience) days passed between the latest status update and the time of observation. It is not of interest because it only depends on when the data are recorded. It could give an idea of the solution time but only for the open cases, not for the fixed ones.

Model

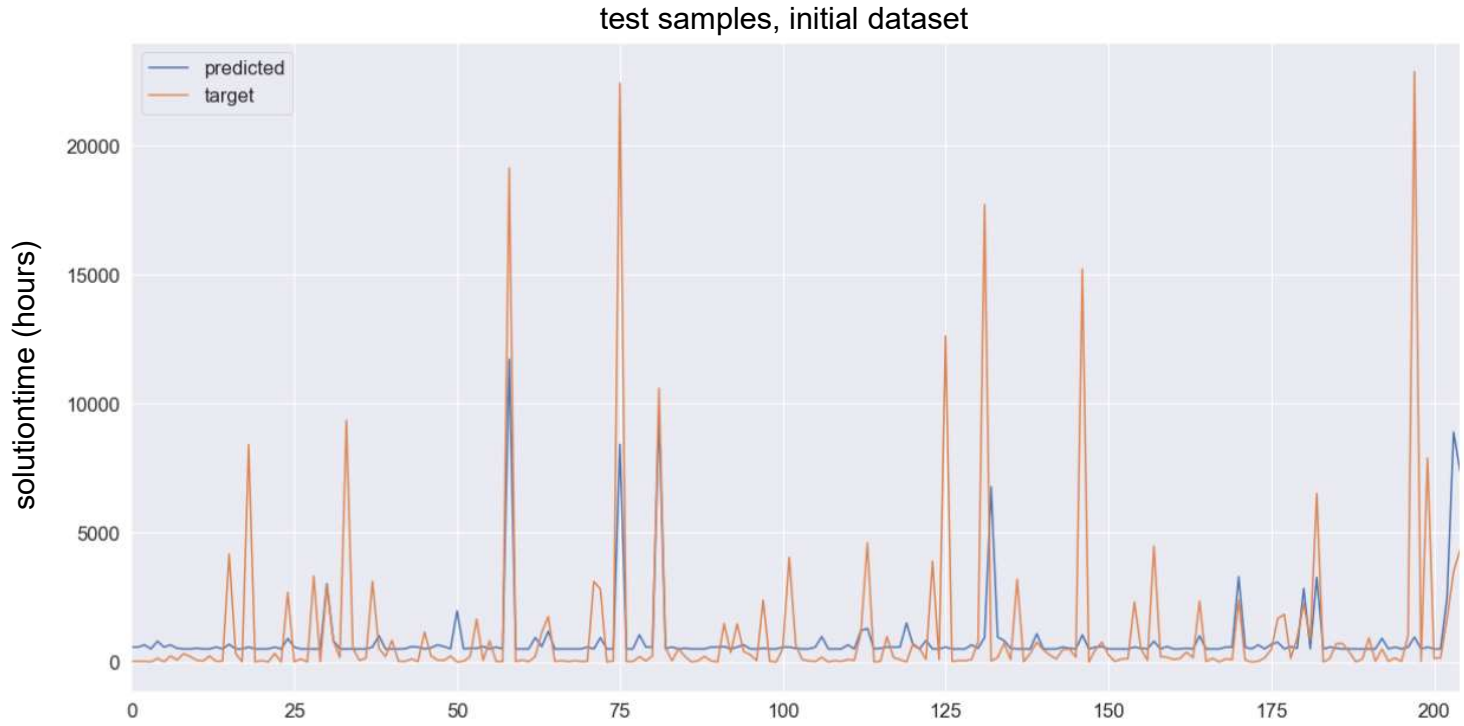
- **Feature selection:** the final list of selected features includes: ‘priority’, ‘issue_type’, ‘reporter’, ‘vote_count’, ‘assignee’, ‘comment_count’, ‘watch_count’.
- **Regressor**: the choice of Random Forest Regressor is motivated by the fact that it can deal with skewed distributions and high number of features (one-hot-encoding). It also outputs feature importance coefficients to refine predictor selection.
- **Validation strategy**: Grid Search is performed on the train set to determine the best model parameters, max depth and n-estimators. In order not to rely on a single random test/train split for the test accuracy, considering the small and unbalanced dataset, 5 different random splits are run and the mean scoring is taken.

Model - results

- Accuracy of the model run on the original dataset shows poor results, with the absolute error being in the same range of the target mean.
- Because unbalance certainly takes a role on that, the dataset was resampled by oversampling the minority class (status=Resolved) simply by multiplying the datapoints until the count reaches the same level of status=Closed.
- r2 score on the resampled dataset is significantly improved, but the mean error is on the same range as the previous one. More understanding and consolidation of the model with resampled dataset is required.

	r2 score		Mean absolute error (hours)		Target mean (hours)
	Original dataset	Resampled	Original dataset	Resampled	
Train set	0.60	0.86			975
Test set	0.25	0.80	988.5	854.2	

Model - results



- The model clearly predicts better for low magnitude targets.
- Prediction results for three interesting cases are included in the report. Not surprisingly the model predicts better for Closed cases than for Resolved ones.

Additional features

Additional potentially useful information on the issue tracking were identified in the Avro website and retrieved from the json file. The following variables were considered:

- **'component'**: the environment affected by the issue. Can be Java (larger portion), C, Python, etc. Intuitively it should have an interdependency with the target.
- **'version':** this is the Avro release version to which the alert refers. A rapid look at the data shows that about 50% of the values are missing and therefore it won't be considered further.
- **'total logs':** this parameter indicates the total number of times an issue thread has been updated. We want to verify whether it has any significant relationship with the target. Intuitively a higher amount of updates should potentially determine longer resolution times. However, the number of changes can be unrelated to their time frequencies.

The model was run on the original dataset (not resampled) with addition of 'component' and 'total logs'. No effects on the scoring. More details in the report.

R2 score	Original dataset	'component' and 'logs' added	Only 'component' added	Only 'logs' added
Train set	0.60	0.53	0.53	0.53
Test set	0.25	0.19	0.20	0.20

Conclusion and next steps

The obtained model can be considered a reasonable starting point, but the dataset characteristics require more analysis and understanding to obtain decent accuracy results. The following improvements and further actions are proposed to tackle the listed issues:

- **Positive skewness**: data transformation (i.e. logarithmic) should be applied to achieve normal – like distributions. This would allow use of linear models for a basic understanding and better benchmark. Use of neural network regressor would then also be possible.
- **Data unbalance**: a quick solution was implemented by oversampling the status Resolved. That improved the coefficient of determination only and requires understanding of the consequences and consolidation. Another methodology to counteract unbalance is weighting the classes. (Scikit learn includes some options for weighting at sampling or scoring levels).
- **Lack of data**: Taking more datapoints from Avro website would improve any model by reducing overfitting, unbalance and instability.
- **Outliers**: Data contains high rates of outliers. A simple approach for modelling would be to gradually start removing outliers values and see the effects. K-Means clustering algorithm could also be used to better investigate possible smaller data clusters.