Princess Sumaya University for Technology

Software Developers Performance Evaluation

Name	Student ID
Mohammad Abdellatif	20208032
Waed Alsawarieh	20208020
Yaser Toutah	20188058

Agenda

- Introduction.
- Problem Statement and Challenges.
- Work Stream and Data Gathering.
- Demo.
- Findings.
- Limitation and Future Enhancements.
- Lessons Learned.

Introduction

Evaluating and measuring the software developers' performance is one of the most difficult tasks for the engineering managers. It's also a source of anxiety for developers at all experience levels: how do you know if you're doing well enough when most of your work is intangible, how it should be measured? can it even be accurately measured? So, it's really a hard job to do, and many managers fail in doing it the right way and without unfairness.

In this project, we'll develop an engine that will act as a recommendation engine (or a decision support tool) to help the engineering managers conduct a well, fair and clear evaluation process for the software developers. It will also reduce the injustice and increase the accuracy of the results when conducting the assessment.

Problem Statement and Challenges

As we have mentioned earlier; the evaluation process is a hard job and it's not straightforward. Many factors affect and influence the evaluation process; even the mood of the reviewer can have an impact!

We're human beings; which means we forget sometimes, we get influenced by external factors, we might personalize things at work, we exaggerate sometimes too, etc. due to all of that, and given that the evaluation process is not like 1+1=2, it would be wonderful if we can develop a model/engine that can help in the evaluation process and can provide indicators that will help the reviewers.

Problem Statement and Challenges

Challenges that affect the evaluation process:

- Biased review.
- Intangible achievements.
- Subjectivity.
- Personalize things.
- Soft skills ignorance.
- Team variation.

Work Stream and Data Gathering /Preparation

Started with two datasets: emails, issues

(Real data from a company in Jordan)

↓

Masked emails, for privacy purposes

↓

Analyzed datasets (Transformation): summaries, drop NA, drop duplicates, etc.

↓

Found inaccuracy between the assignees of tasks and actual work log

↓

Requested more data for work log only

↓

Produced relationships

Work Stream and Data Gathering/Preparation

Analyzed datasets (Transformation): summaries, drop NA, drop duplicates, etc.

↓

Joined the issues data with emails data

↓

Apply Kmeans

↓

Enhanced and developed the relationships using Neo4J

↓

Nifi work

↓

README file

Demo

- Our work is on github: https://github.com/waedbara/SoftwareDevelopersEvaluationProject

- The README file has all technical details and how to run the application.

Findings

- Too many external (out of organization) reporters. (why!)
- Too many developers create tickets and assign them to themselves. (process!)
- Too many developers don’t send emails. (~85%)
- 80% work of a data engineer is data preparation.

Limitations and Future Enhancements

- We were having 2 data sources:
  - GitLab (data about developers).
  - Emails (data about developers + non-developers).

  The second data source (Emails data) has inaccuracy.
- We need more data (Gitlab data is for 3 months, while emails data is for only one month).
- We had to do masking for privacy purposes, but at the same time we couldn't link the entities with each other directly. (we need to do mapping whenever we need to know who we are referring to).
- NiFi does not support neo4j v4+, so we had to use an older version (we used 3.4).
- After using NiFi, we found that some simple tasks can be done in Airflow in a simpler fashion, like splitting a single record into multiple records.
- NiFi from tooling perspective is more powerful than Airflow.
- Integrate with BI tool (Kibana).

Lessons Learned

- Data is the new oil!
- The more data you have, the better.
- You can’t imagine how many things you can produce/elicit until you try and see by yourself!
- Start with simple things first, don’t jump to the edge cases first.
- Don’t assume, always ask and get things clarified.
- Verify the accuracy of the data, and make sure that this is what really needed.
- Ask for more data in the early stages (as soon as you feel you need more data).

Thank You


Princess Sumaya University for Technology
Since 1991 تأسست
PSUT
جامعة الأميرة سمية للتكنولوجيا