

**Incident and Known Error Recommender**  
**Suman Das - 324917**

**BFSI UK 2.3**  
**February 2020**  
**Version 1.00**

#### Notice

© 2020 Tata Consultancy Services Limited

This is a controlled document. Unauthorized access, copying, replication or usage for a purpose other than for which it is intended, are prohibited.

All trademarks that appear in the document have been used for identification purposes only and belong to their respective companies.

## References

***<https://towardsdatascience.com/introduction-to-natural-language-processing-for-text-df845750fb63>***

## Abstract

---

This paper implements the theme of the Digital World relying heavily on one of the pillars of Business 4.0 (Intelligence).

Remedy is a very popular tool in the domain of IT Production Support. Every organization which uses this application has hundreds of old remedies which describes one issue or the other. Consider the amount of information stored in the remedy system for an organization which has been using it for years. Does it not make sense to have all these info in the fingertips so that they can be readily accessed and any new similar issue can be solved with the least amount of time by implementing a similar old fix?

This white paper describes how to use new digital technologies to improve this search functionality. The framework used is **Content Based Recommendation** to create an **Incident Recommender**. An user who wants to search an old but similar Incident can use the recommendations to find the fix that was done earlier on a similar remedy and use the knowledge to fix his current issue right away.

### General Terms

Remedy Search, Recommendation, Incidents, Known Error

### Keywords

Content Based Recommendation, Flask, Python, Natural Language Processing (NLP)

## Contents

---

1. INTRODUCTION.....	3
2. CHALLENGES IN CURRENT REMEDY SYSTEM.....	4
3. NLP POWERED HOLISTIC SEARCH.....	5
4. IMPLEMENTATION... ..	6
5. GENERALIZING THE CONCEPT .....	7
6. CONCLUSION.....	8

## 1. Introduction

---

An organization, which doesn't have a KEDB for its IT problems will end up spending on repetitive tasks and a huge chunk of the money will go to its suppliers, thereby impacting profitability. However, to build a KEDB needs a lot of manual effort on a daily basis and is prone to human errors. What if we can build a real time KEDB using all the information in the existing Incidents that were raised in the past? That will not only save time for any new issues that come up, but also save a lot of money.

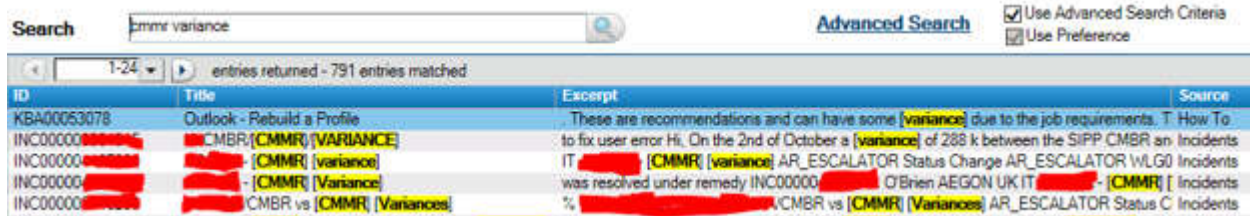
BMC Remedy from Remedy Corporation is one of the prominent tool that is being used by a lot of organization to support their Production line. However, Remedy has a search functionality which is outdated and uses simple keyword or phrase search. That is nowhere close to the modern NLP powered search in today's digital era. Some of the new age tools like ServiceNow are building up their capabilities to include AI intelligent search within themselves, but again these come with a higher cost.

In this paper we will leverage Digital Technologies to build a real time Recommendation Engine using open source material and thereby incurring zero cost. The application will recommend a set of old remedies or known errors to a user who is facing a new live production issue. The idea is to use the current content in the system to recommend something similar to the text being searched and hence the process is known as Content Based Recommendation. These recommendations will very closely match to the text of the current issue. The user can then refer to the fix that was done on the old Incident and implement a similar strategy to fix the recent problem.

The paper is organized into eight sections—Section 2 discusses the challenges in the current remedy search. Section 3 is devoted to the design of the NLP powered recommender. Sections 4 shows how the implemented web application looks like and what are the output of the recommender. Section 5 generalizes the concept and shows how the same functionality can be used in any kind of text search. Finally section 6 concludes the paper.

## 2. Challenges in the current Remedy system

The two figures below shows the search results in the current Remedy System. You can see from fig 1 that the keyword search is spot on however fig 2 shows when your search gets a bit complex the search criterion fails and the incidents predicted do not match the criterion. In fact the predicted incidents are nearly the same and, in the second search there is no link to account 64.



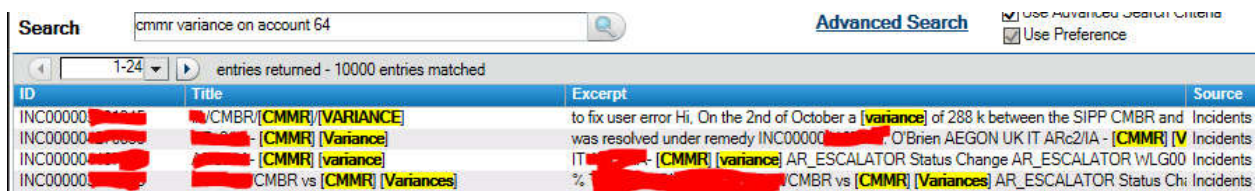
Search: cmmr variance

Advanced Search ☒ Use Advanced Search Criteria ☒ Use Preference

entries returned - 791 entries matched

ID	Title	Excerpt	Source
KBA00053078	Outlook - Rebuild a Profile	These are recommendations and can have some [variance] due to the job requirements. T. How To	
INC00000 [REDACTED]	[REDACTED] CMBR [CMMR] [VARIANCE]	to fix user error Hi. On the 2nd of October a [variance] of 288 k between the SIPP CMBR an Incidents	
INC00000 [REDACTED]	[REDACTED] [CMMR] [variance]	IT [REDACTED] [CMMR] [variance] AR_ESCALATOR Status Change AR_ESCALATOR WLG00 Incidents	
INC00000 [REDACTED]	[REDACTED] [CMMR] [Variance]	was resolved under remedy INC00000 [REDACTED] O'Brien AEGON UK IT [REDACTED] [CMMR] [ Incidents	
INC00000 [REDACTED]	[REDACTED] CMBR vs [CMMR] [Variances]	% [REDACTED] CMBR vs [CMMR] [Variances] AR_ESCALATOR Status C Incidents	

Fig 1: Keyword search



Search: cmmr variance on account 64

Advanced Search ☒ Use Advanced Search Criteria ☒ Use Preference

entries returned - 10000 entries matched

ID	Title	Excerpt	Source
INC00000 [REDACTED]	[REDACTED] CMBR [CMMR] [VARIANCE]	to fix user error Hi. On the 2nd of October a [variance] of 288 k between the SIPP CMBR and Incidents	
INC00000 [REDACTED]	[REDACTED] [CMMR] [Variance]	was resolved under remedy INC00000 [REDACTED] O'Brien AEGON UK IT ARc2/IA - [CMMR] [V Incidents	
INC00000 [REDACTED]	[REDACTED] [CMMR] [variance]	IT [REDACTED] [CMMR] [variance] AR_ESCALATOR Status Change AR_ESCALATOR WLG00 Incidents	
INC00000 [REDACTED]	[REDACTED] CMBR vs [CMMR] [Variances]	% [REDACTED] CMBR vs [CMMR] [Variances] AR_ESCALATOR Status Chi Incidents	

Fig 2: Complex search

Even though Remedy allows you to do an Advanced Search, you can at best search for a phrase. However, if the words you are looking for is not the exact phrase the search is futile.

### 3. NLP powered holistic search

We design a new search functionality which has the below components.

1. A dump of the existing Incidents and Known Errors with their notes and some other relevant details. Each Incident acts as record and is saved in a csv file.
2. A browser based GUI which uses a text (details of the new remedy) input by the user.
3. Python source code which does NLP processing on both the dump and the input text and then compares the input text to all the individual notes of the dump. The Incident/Known Error which has the maximum number of matching words has the closest similarity to the new Incident and has the highest score. This is then wrapped into a Flask application.
4. An AWS EC2 private cloud instance to deploy the above Flask application and make it available for all user.

The third component is the backbone of the entire application and we will discuss this in further details.

Say we have the below dump, saved in a csv file and we need to output the one which matches closely to the input text '**CMMR variance on a running account 64**'. In this case INC000002 is closest match and INC000003 is the next match.

Incident No	Notes	Vendor Ticket No	Team
INC000001	There is an issue on rebalancing of funds.	1234	Java IT Support
INC000002	Found CMMR variance on many account. One account is 64.	9999	Oracle Admin Support
INC000003	CMMR Variance SIPP account	2568	Java IT Support

**Step 1. Remove the stop words, convert to lower case, remove punctuation and stem the other words for input text and the dump.**

*Input text: 'CMMR variance on a running account 64'*

*Converted to: 'cmmr varianc run account 64'*

*Dump Converted to:*

Incident No	Notes	Vendor Ticket No	Team
INC000001	issu rebalanc fund	1234	Java IT Support
INC000002	found cmmr varianc mani account one account 64	9999	Oracle Admin Support
INC000003	cmmr varianc sip account	2568	Java IT Support

**Step 2. Use Cosine Similarity to find the score of individual match per Incident.**

This is a python api which converts each word into a feature(column) and calculates the fraction of matching features to the total number of input features (in this case it is 5 as there are 5 words in the input text after step 1).

So, for INC000001 score is:  $0/5=0$ , for INC000002 score is:  $4/5=0.80$  and for INC000003 score is:  $3/5=0.60$ .

Hence the closest match is INC000002 followed by INC000003. And this is returned in the GUI.



## 4. Implementation

Fig 3 shows how the search functionality would look like in bare bones. Fig 4 shows the output of the predicted remedies. This is hosted in AWS EC2 private cloud instance.

### Search Similar Incidents

[Home](#)

Enter the text you want to search(Mandatory):

cmmr variance account 64

Input Assignee Group where you want to search(Mandatory):

- ☐ Support Queue 1  
☐ Support Queue 2  
☐ Support Queue 3

Search Similar Incidents

Fig 3: Web Implementation

### Similar Incidents Found

[Home](#)

[Search Another Incident](#)

Incident	Vendor Ticket No	Notes
Your Incident's Text	NA	cmmr variance account 64
INC000004	VEN0087	CMMR SIPP Variance has been reported in account 64
INC000008	NO_TICKET	CMMR Variance on all accounts
INC000009	VEN0055	many variance reported on a lot of accounts. This involves account 64
INC000007	VEN0017	CMMR variance causing issue
INC000006	VEN0080	CMMR SIPP Variance has been reported in account 79
INC000001	VEN0087	CMMR SIPP Variance has been reported in account 76
INC000010	VEN0066	Rebalancing issue causing variances in multiple accounts
INC000003	NO_Ticket	CMMR for the 02/01/19 has not been produced for SIPP 2 accounts

Fig 4: Recommended Incidents

In Fig 4 the Incidents at the top are closer match to the input text rather than the incidents below them. The first column has the Incident number and the 2nd column has the vendor tickets number. These details can then be used by the user for finding out how a fix was implemented in the past.

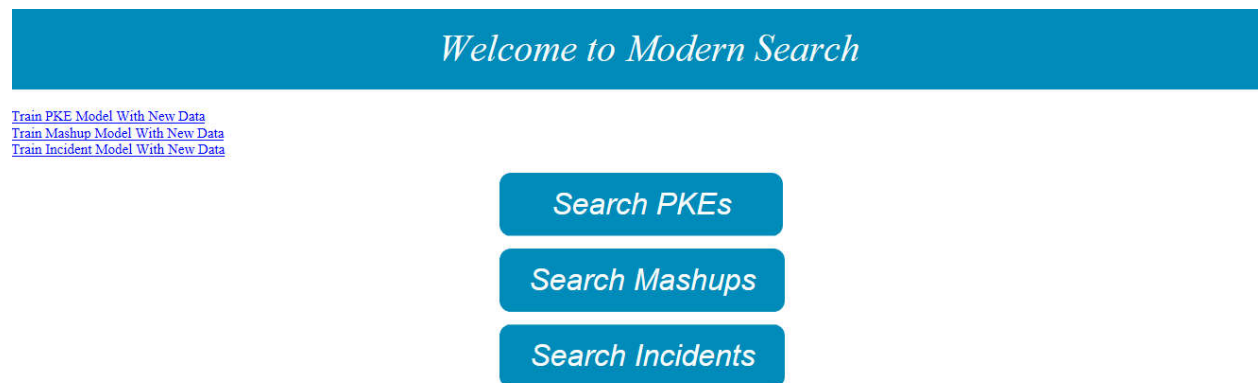
## 5. Generalizing the concept

The functionality that has been designed here is purely NLP powered. The other technological bits like Python, Flask, AWS EC2 etc are all IT enablers. So, it makes sense if we can generalize the functionality for any kind of text search.

This design can be used to search any kind of text be it a Incident, Known Error or any other vendor tickets for which you have a dumps of all the different issues.

The search has been extended to a third party vendor's incident tracking system (called Mashups in this instance). Fig 5 shows 2 different functionalities added to search Mashups and PKE (Problem Known Errors).

The hyperlink on the top left is used to preprocess the new data (train) by following step 1 and 2 in section 3. This needs to be done at a regular interval whenever there are more new incident/PKEs/Mashup added to the system and you want your search to run through them.



**Fig 5. Generalizing the Search Functionality**

## 6. Conclusion

---

Implementing this functionality helps the client in searching over a wide range of platforms and coming to a quick conclusion on an Incident which has newly occurred in the production line. This search functionality caters like a virtual KEDB and puts all different issues at the fingertips of the user. This design when implemented has prevented the customer from going to the 3<sup>rd</sup> party suppliers with the same issue that has been raised, say around a couple of years back.

The idea of Cosine Similarity involved here can be replaced by more language intensive Deep Learning technologies like LSTM and RNN which remembers the reference of a text and uses that to compare with other relevant information. This can then recommend similar Incidents/PKEs not only on the matching words but also on matching reference to any issue.

