The Algorithm of 10,000 Variables

A Case Study On Machine Learning in Software Assessment

Cobec Consulting Inc.

May 3rd, 2021

Agenda

- Introduction
  - Who We Are
  - Project Background
- Methodology
- Data Source
  - Examples Data Pipeline
- Models
- Experimental Results
- Conclusion

Introduction - Morgan Heimbrock

Morgan Heimbrock is a Consultant with Cobec Consulting supporting the acquisition oversight group at the Federal Aviation Administration's Office of Investment Planning and Analysis. Currently, she is helping with financial data aggregation, software development, and product management. Ms. Heimbrock has a Bachelor of Science in Finance and Accounting from the University of Kentucky.

Introduction - Bryan Anderson

Bryan Anderson is a Consultant with Cobec Consulting focusing on OR and software engineering in the Federal Aviation Administration's AFI and AJW-2000 offices. Mr. Anderson's academic background includes a BA in Economics & Mathematics from Augsburg College and MS in Industrial and Systems Engineering from the University of Minnesota - Twin Cities. Mr. Anderson has presented at various conferences and engaged in the ICEAA community. Prior to being working in the federal government, he was an Industrial Engineer at a national grocer optimizing supply chain operations.

Figure 2: Bryan Anderson

Introduction - Project Background

- FAA owns and operates equipment across the whole National Airspace System (NAS). The FAA Logistics Center (FAALC) manages the NAS inventory and distribution of needed parts. FAALC provides:
  - support for more than 48,000 systems
  - management of 62,000 National Stock Numbers
  - inventory value of more than $900 million
  - customer support of over 8,000 FAA, DoD, State, and foreign country facilities
- There is a need to enhance the FAA supply chain operations by replacing legacy logistics support system with a COTS ERP. Due to the complexity of supply chain operations, changing needs of NAS needs, and consolidation of information incurred impediments in program implementation.

Introduction - Project Background (cont. 1)

- With delayed requirements, program functional and non-functional needs continued to evolve causing software rework. Assessment of the cost and schedule of this software work was critical for planning but difficult in practice with traditional methods.
- Function point analysis was initially performed, but thought not to be able to capture all the work required due to non-functionality rework and data quality. SLOC was also not readily available and not a practical approach.
- This project attempts to use a novel approach to assess resource needs for software programs using actual data collected by the development's Issue Tracking System (i.e. Jira).
- Extracting this historical data, several statistical models where trained and used to predict developer level work activities.

Methodology - Overview

- General approach has two major components:
  - Data Engineering
    - Data Pipeline - Data is extracted from the source and goes through several steps to transform, validate, and formatted for analytics.
    - Feature Extraction - Feature extraction and engineering is done at each step. Each step provides more rigorous methods to generate features.
  - Statistical Modeling - Several models where generated using a advanced analytical tool. Model results where analyzed for quality and enhancement options.

Methodology - Data Pipeline

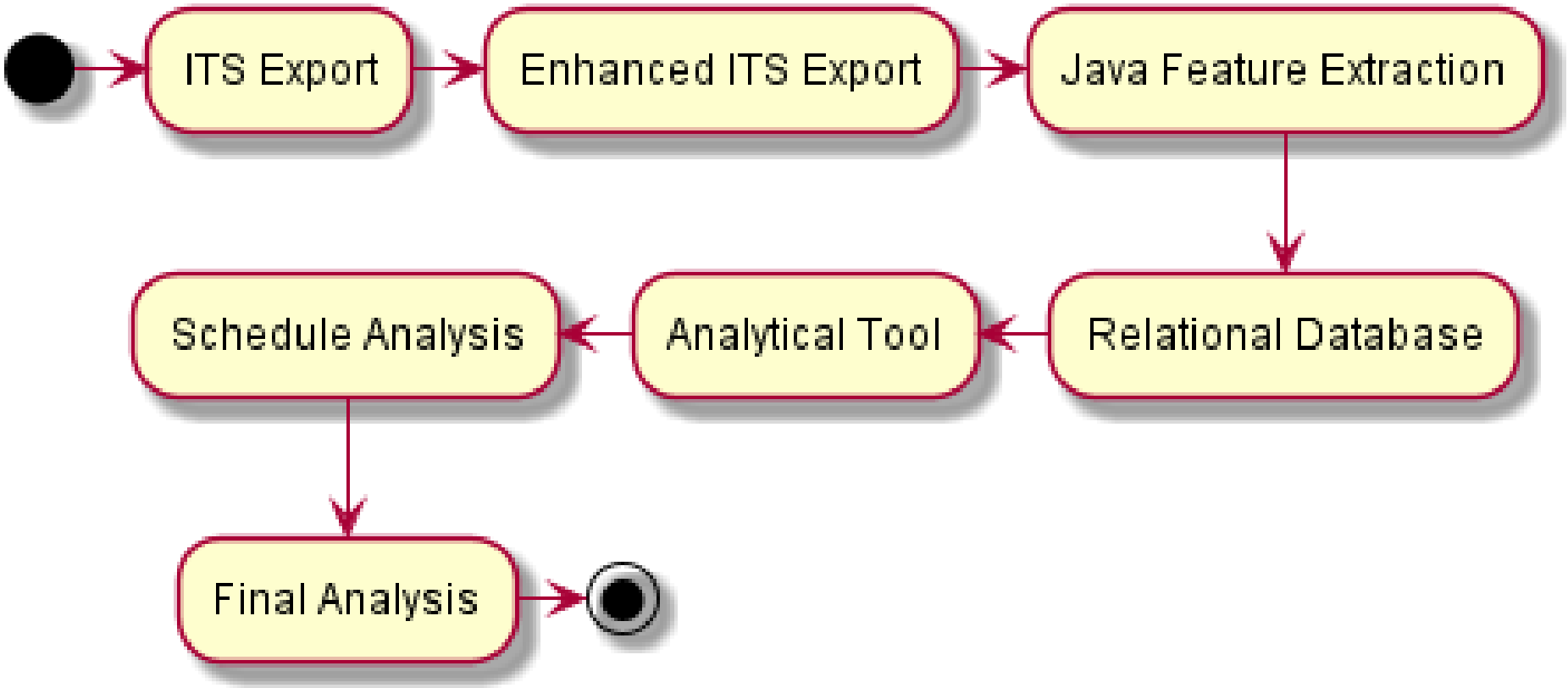


Figure 3: Data Pipeline

Methodology - Data Pipeline Descriptions

1. ITS Export

- Initial Jira data exported as CSV file. Contains raw data such as issue type, issue comments, individuals assigned to roles, status, creation date, resolution date etc.

2. Enhanced ITS Export (EITSE)

- Even though it was standard report from Jira, report structure wasn't standard among the several exports. E.g. comments on an issue would generate a new column. Data was transformed in Excel using VBA and formulas to standardize data structure to combine the several reports.

Methodology - Data Pipeline Descriptions (cont. 1)

3. Java Feature Extraction

- A Java program parses the EITSE and performed another round of feature extraction. Using Snowball evaluated basic text token extraction (token identification and stemming) and other data transformations (e.g. parse strings to timestamps).

4. Relational Database

- The parsed data from the previous step was then persisted in a relational database. More feature engineering was done in this layer such as encoding of categorical data. This layer provided a way to create a standard way to create initial feature sets with stored procedures and data lineage.

Methodology - Data Pipeline Descriptions (cont. 2)

5. Analytical Tool

- Feature sets created in the RDBS is then brought into a analytical tool for advanced statistical analysis. Here several models where trained, validated, analyzed, and forecast remaining unresolved issues.

6. Schedule Analysis

- The net outcome from the previous step is prediction of unresolved issues. To account for resource constraints (e.g. can't work all unresolved simultaneously), results where simplified into 12 week durations and limits imposed on number of issues that can be resolved per month.

Methodology - Data Pipeline Descriptions (cont. 3)

7. Final Analysis

- Findings from the Schedule Analysis is then combined with SEER-SIM using a heuristic to formulate the schedule estimate.

Data Source

- ITS provides project members to plan and organize work activities. In this use case, the popular Atlassian Jira system was used.
- Developers 'work against' issues. Issues contain the work that needs to be done.
- ITS captures historical activity on the developer's work in these issues.
- Issues are atomic units of work. Issues contain data elements such as:
  - Types with standardized workflows
  - Textual description of the work (summaries, description, comments).
  - Attachments
  - Specific individuals assigned to roles (assignee, creator, commentator, watcher).
  - Specific timestamps (creation, resolution, comments)
- Development frameworks (e.g. SAFe) are desired for mature workflows. This makes data more predictable and reliable.

Data Source Example

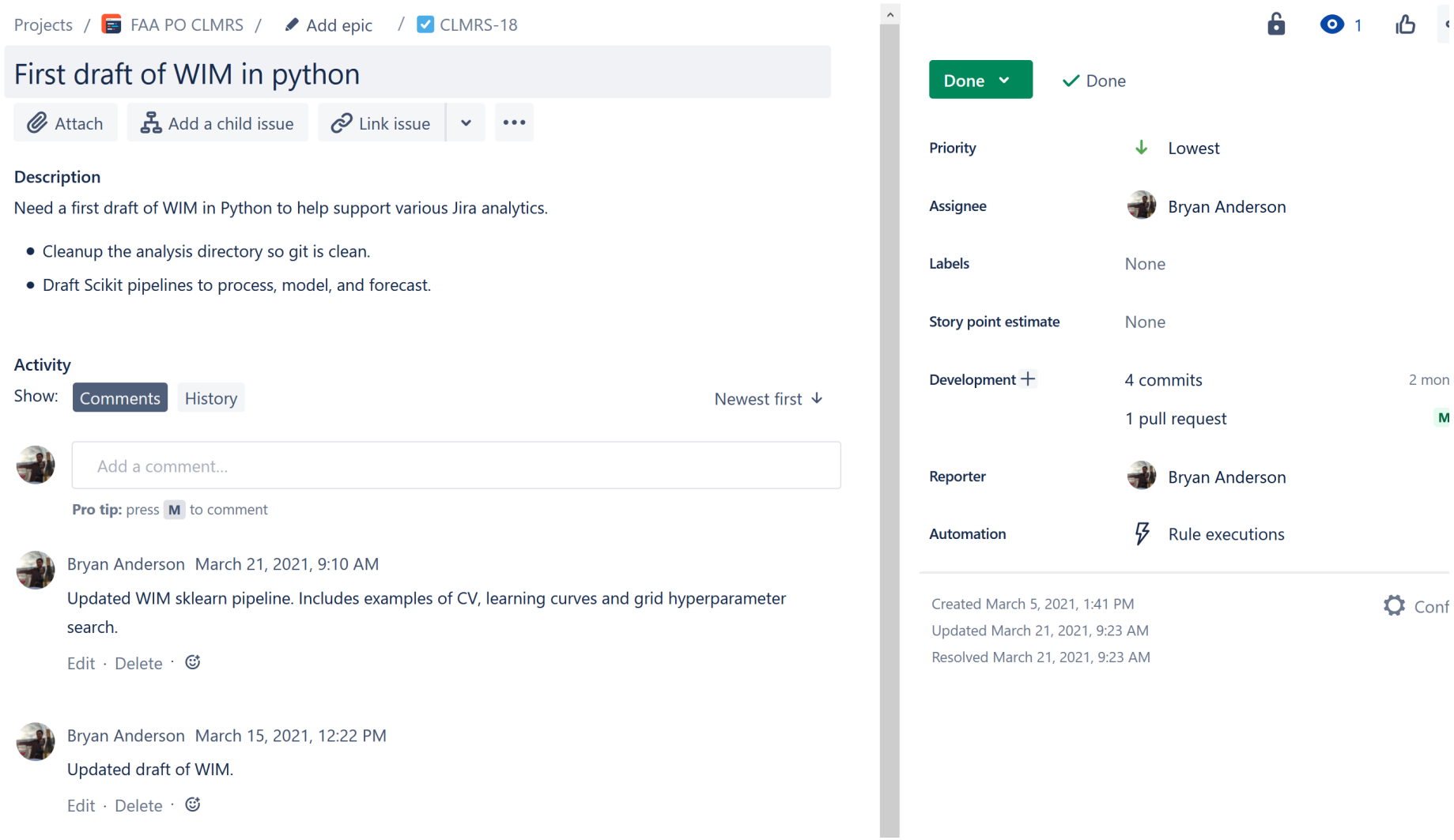
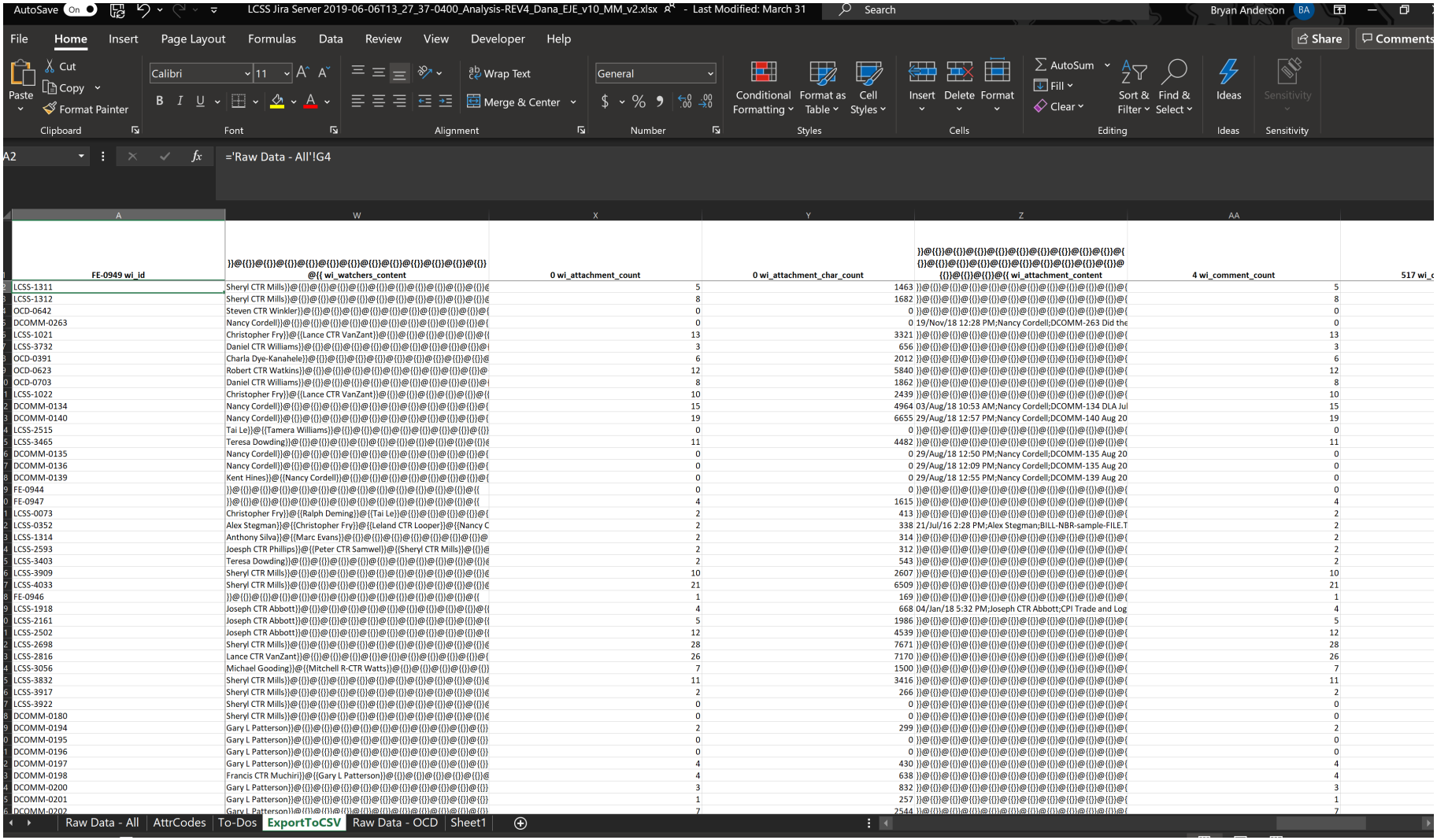


Figure 4: Example Jira Ticket

Data Source Example - EITS 0



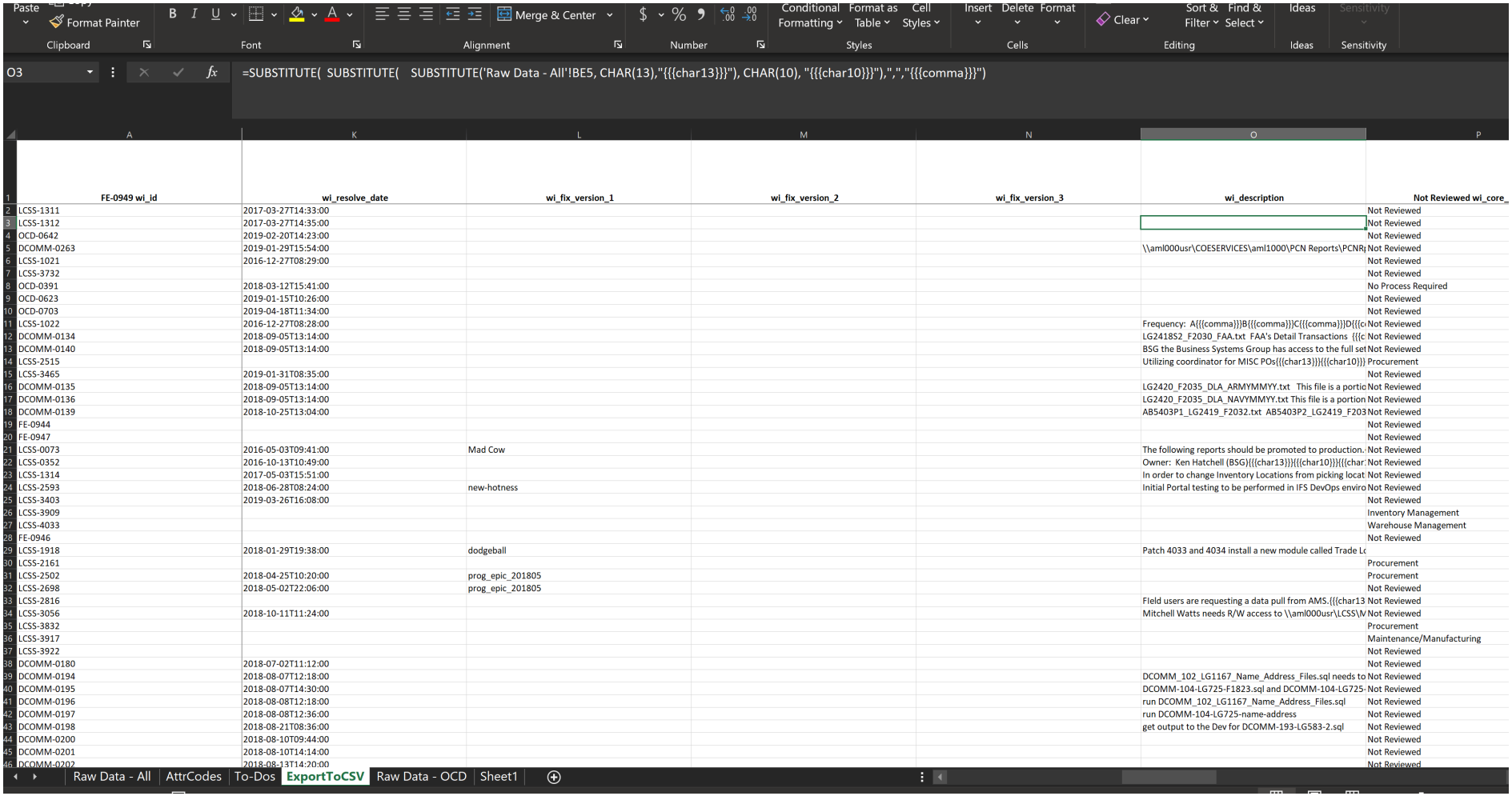
LCSS Jira Server 2019-06-06T13_27_37-0400_Analysis-REV4_Dana_EJE_v10_MM_v2.xlsx - Last Modified: March 31

A2 ='Raw Data - All'!G4

FE-0949 wi_id	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}} @{{ wi_watchers_content	0 wi_attachment_count	0 wi_attachment_char_count	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{ {}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@ {{}}@{{}}@{{}}@{{ wi_attachment_content	4 wi_comment_count	517 wi_
LCSS-1311	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	5	1463	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	5	
LCSS-1312	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	8	1682	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	8	
OCD-0642	Steven CTR Winkler}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	0	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	
DCOMM-0263	Nancy Cordell}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	0	19/Nov/18 12:28 PM;Nancy Cordell;DCOMM-263 Did the	0	
LCSS-1021	Christopher Fry}}@{{Lance CTR VanZant}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	13	3321	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	13	
LCSS-3732	Daniel CTR Williams}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	3	656	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	3	
OCD-0391	Charla Dye-Kanahele}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	6	2012	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	6	
OCD-0623	Robert CTR Watkins}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	12	5840	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	12	
OCD-0703	Daniel CTR Williams}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	8	1862	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	8	
LCSS-1022	Christopher Fry}}@{{Lance CTR VanZant}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	10	2439	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	10	
DCOMM-0134	Nancy Cordell}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	15	4964	03/Aug/18 10:53 AM;Nancy Cordell;DCOMM-134 DLA Jul	15	
DCOMM-0140	Nancy Cordell}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	19	6655	29/Aug/18 12:57 PM;Nancy Cordell;DCOMM-140 Aug 20	19	
LCSS-2515	Tai Le}}@{{Tamera Williams}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	0	0	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	
LCSS-3465	Teresa Dowding}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	11	4482	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	11	
DCOMM-0135	Nancy Cordell}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	0	29/Aug/18 12:50 PM;Nancy Cordell;DCOMM-135 Aug 20	0	
DCOMM-0136	Nancy Cordell}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	0	29/Aug/18 12:09 PM;Nancy Cordell;DCOMM-135 Aug 20	0	
DCOMM-0139	Kent Hines}}@{{Nancy Cordell}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	0	29/Aug/18 12:55 PM;Nancy Cordell;DCOMM-139 Aug 20	0	
FE-0944	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	0	0	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	
FE-0947	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	4	1615	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	4	
LCSS-0073	Christopher Fry}}@{{Ralph Deming}}@{{Tai Le}}@{{}}@{{}}@{{}}@{{}}@{	2	413	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	2	
LCSS-0352	Alex Stegman}}@{{Christopher Fry}}@{{Leland CTR Looper}}@{{Nancy C	2	338	21/Jul/16 2:28 PM;Alex Stegman;BILL-NBR-sample-FILE.T	2	
LCSS-1314	Anthony Silva}}@{{Marc Evans}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	2	314	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	2	
LCSS-2593	Joesph CTR Phillips}}@{{Peter CTR Samwel}}@{{Sheryl CTR Mills}}@{{}}@	2	312	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	2	
LCSS-3403	Teresa Dowding}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	2	543	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	2	
LCSS-3909	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	10	2607	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	10	
LCSS-4033	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	21	6509	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	21	
FE-0946	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	1	169	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	1	
LCSS-1918	Joseph CTR Abbott}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	4	668	04/Jan/18 5:32 PM;Joseph CTR Abbott;CPI Trade and Log	4	
LCSS-2161	Joseph CTR Abbott}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	5	1986	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	5	
LCSS-2502	Joseph CTR Abbott}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{	12	4539	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	12	
LCSS-2698	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	28	7671	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	28	
LCSS-2816	Lance CTR VanZant}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	26	7170	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	26	
LCSS-3056	Michael Gooding}}@{{Mitchell R-CTR Watts}}@{{}}@{{}}@{{}}@{{}}@{{}}	7	1500	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	7	
LCSS-3832	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	11	3416	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	11	
LCSS-3917	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	2	266	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	2	
LCSS-3922	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	0	0	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	
DCOMM-0180	Sheryl CTR Mills}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@	0	0	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	
DCOMM-0194	Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	2	299	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	2	
DCOMM-0195	Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	0	0	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	
DCOMM-0196	Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	0	0	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	0	
DCOMM-0197	Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	4	430	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	4	
DCOMM-0198	Francis CTR Muchiri}}@{{Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@	4	638	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	4	
DCOMM-0200	Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	3	832	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	3	
DCOMM-0201	Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	1	257	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	1	
DCOMM-0202	Gary L Patterson}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}	7	2544	}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{{}}@{	7	

Figure 5: Example Enhanced ITS Export 0

Data Source Example - EITS 1



O3 =SUBSTITUTE(SUBSTITUTE(SUBSTITUTE('Raw Data - All'!BE5, CHAR(13),"{{{char13}}}"), CHAR(10), "{{{char10}}}"),",","{{{comma}}}")

FE-0949 wi_id	wi_resolve_date	wi_fix_version_1	wi_fix_version_2	wi_fix_version_3	wi_description	Not Reviewed wi_core_
LCSS-1311	2017-03-27T14:33:00					Not Reviewed
LCSS-1312	2017-03-27T14:35:00					Not Reviewed
OCD-0642	2019-02-20T14:23:00					Not Reviewed
DCOMM-0263	2019-01-29T15:54:00				\\aml000usr\COESERVICES\aml1000\PCN Reports\PCNR	Not Reviewed
LCSS-1021	2016-12-27T08:29:00					Not Reviewed
LCSS-3732						Not Reviewed
OCD-0391	2018-03-12T15:41:00					No Process Required
OCD-0623	2019-01-15T10:26:00					Not Reviewed
OCD-0703	2019-04-18T11:34:00					Not Reviewed
LCSS-1022	2016-12-27T08:28:00				Frequency: A{{{comma}}}B{{{comma}}}C{{{comma}}}D{{{c	Not Reviewed
DCOMM-0134	2018-09-05T13:14:00				LG2418S2_F2030_FAA.txt FAA's Detail Transactions {{{c	Not Reviewed
DCOMM-0140	2018-09-05T13:14:00				BSG the Business Systems Group has access to the full set	Not Reviewed
LCSS-2515					Utilizing coordinator for MISC POs{{{char13}}}{{{char10}}}	Procurement
LCSS-3465	2019-01-31T08:35:00					Not Reviewed
DCOMM-0135	2018-09-05T13:14:00				LG2420_F2035_DLA_ARMYMMYY.txt This file is a portio	Not Reviewed
DCOMM-0136	2018-09-05T13:14:00				LG2420_F2035_DLA_NAVYMMYY.txt This file is a portion	Not Reviewed
DCOMM-0139	2018-10-25T13:04:00				AB5403P1_LG2419_F2032.txt AB5403P2_LG2419_F203	Not Reviewed
FE-0944						Not Reviewed
FE-0947						Not Reviewed
LCSS-0073	2016-05-03T09:41:00	Mad Cow			The following reports should be promoted to production.	Not Reviewed
LCSS-0352	2016-10-13T10:49:00				Owner: Ken Hatchell (BSG){{{char13}}}{{{char10}}}{{{char:	Not Reviewed
LCSS-1314	2017-05-03T15:51:00				In order to change Inventory Locations from picking locat	Not Reviewed
LCSS-2593	2018-06-28T08:24:00	new-hotness			Initial Portal testing to be performed in IFS DevOps enviro	Not Reviewed
LCSS-3403	2019-03-26T16:08:00					Not Reviewed
LCSS-3909						Inventory Management
LCSS-4033						Warehouse Management
FE-0946						Not Reviewed
LCSS-1918	2018-01-29T19:38:00	dodgeball			Patch 4033 and 4034 install a new module called Trade Lc	
LCSS-2161						Procurement
LCSS-2502	2018-04-25T10:20:00	prog_epic_201805				Procurement
LCSS-2698	2018-05-02T22:06:00	prog_epic_201805				Not Reviewed
LCSS-2816					FIeld users are requesting a data pull from AMS.{{{char13	Not Reviewed
LCSS-3056	2018-10-11T11:24:00				Mitchell Watts needs R/W access to \\aml000usr\LCSS\W	Not Reviewed
LCSS-3832						Procurement
LCSS-3917						Maintenance/Manufacturing
LCSS-3922						Not Reviewed
DCOMM-0180	2018-07-02T11:12:00					Not Reviewed
DCOMM-0194	2018-08-07T12:18:00				DCOMM_102_LG1167_Name_Address_Files.sql needs to	Not Reviewed
DCOMM-0195	2018-08-07T14:30:00				DCOMM-104-LG725-F1823.sql and DCOMM-104-LG725-	Not Reviewed
DCOMM-0196	2018-08-08T12:18:00				run DCOMM_102_LG1167_Name_Address_Files.sql	Not Reviewed
DCOMM-0197	2018-08-08T12:36:00				run DCOMM-104-LG725-name-address	Not Reviewed
DCOMM-0198	2018-08-21T08:36:00				get output to the Dev for DCOMM-193-LG583-2.sql	Not Reviewed
DCOMM-0200	2018-08-10T09:44:00					Not Reviewed
DCOMM-0201	2018-08-10T14:14:00					Not Reviewed
DCOMM-0202	2018-08-13T14:20:00					Not Reviewed

Figure 6: Example Enhanced ITS Export 1

Data Source Example - RDBS Example

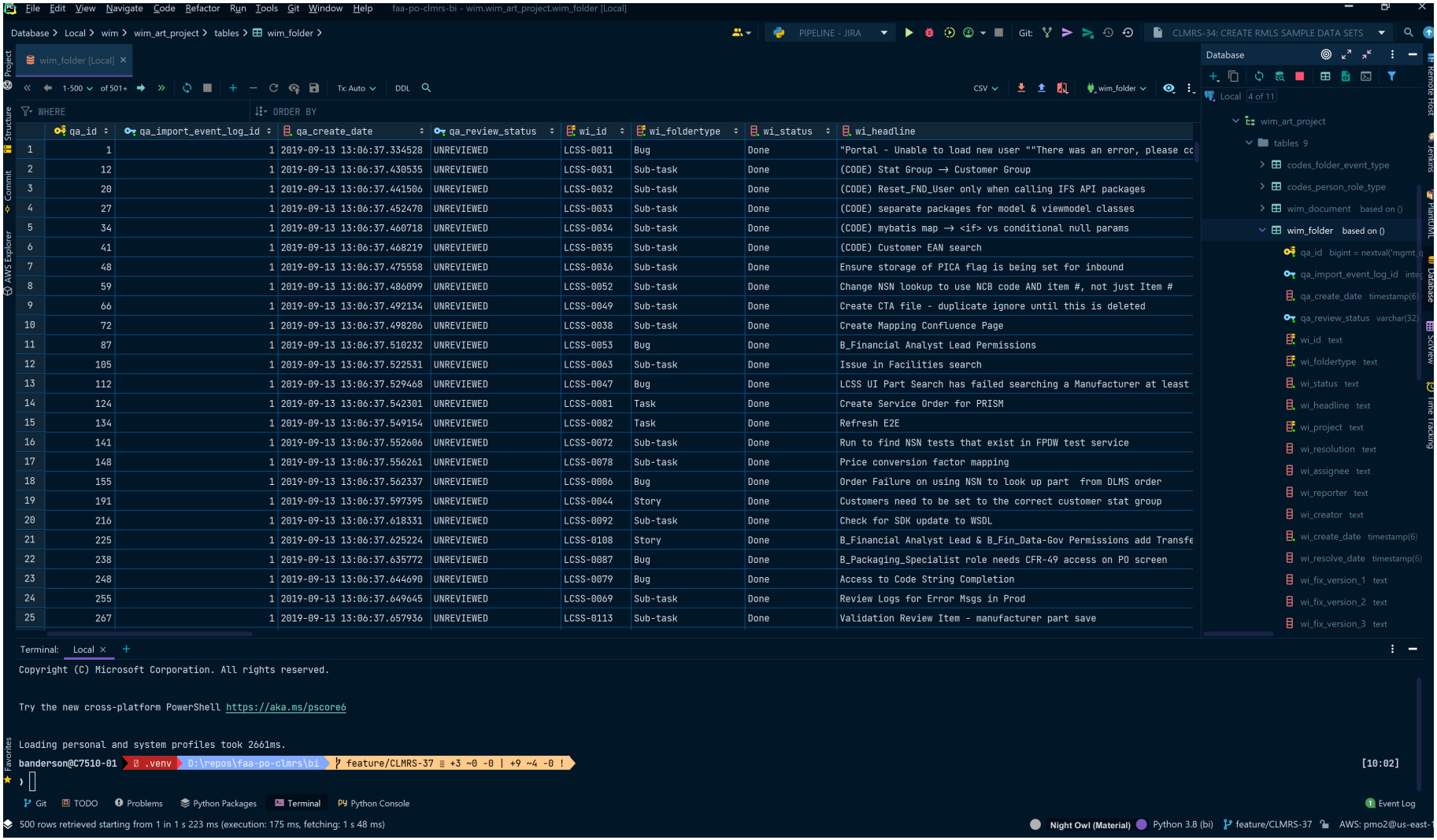


Figure 7: RDBS Example

Data Source Example - RDBS Schema

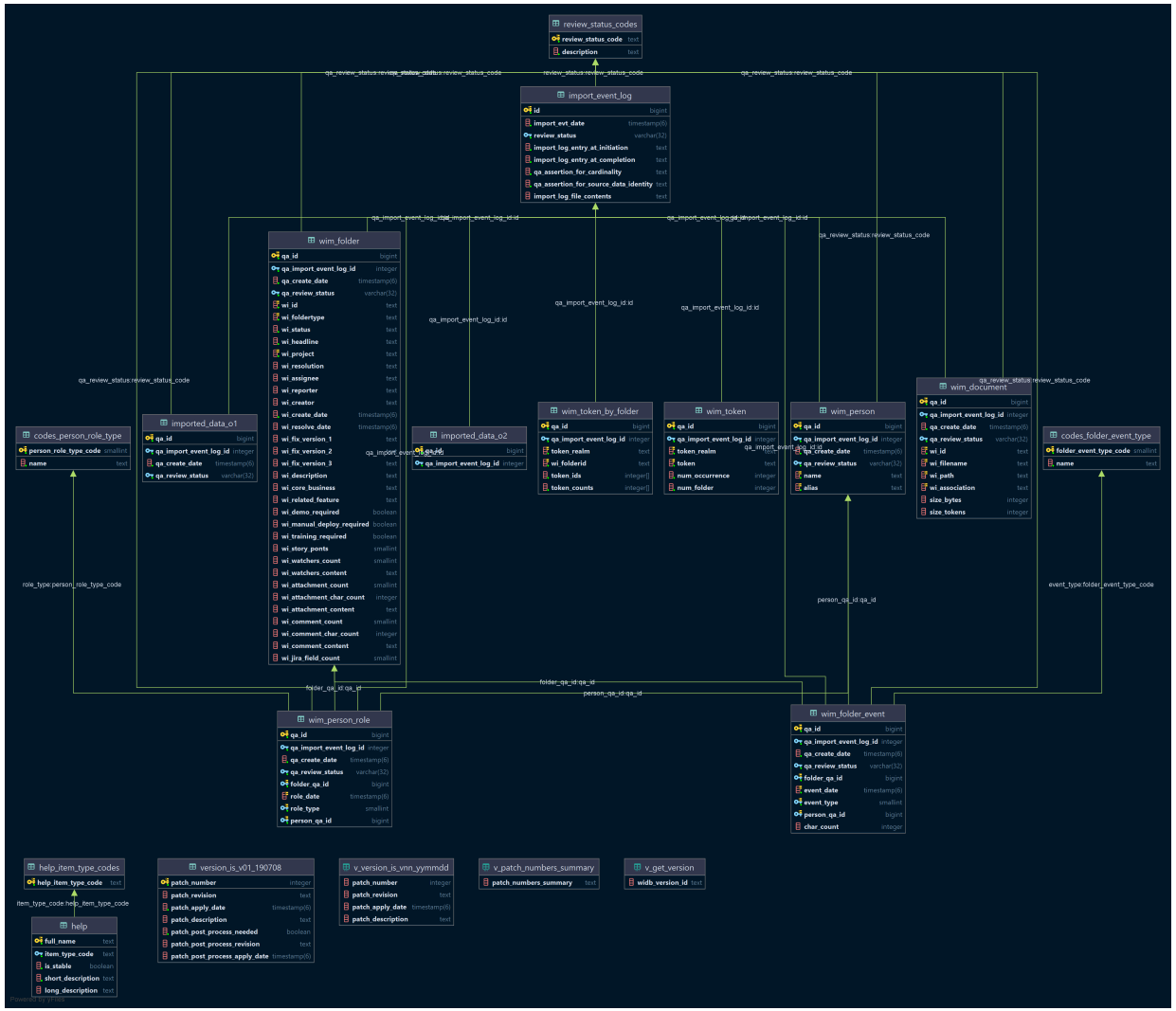


Figure 8: RDBS Schema

Models

The objective of the analysis is to predict how long it will take a developer level work item to be resolved. Thus, the label of the models was the time from the creation to the resolution date. Data is then binned by quantiles[1] of the labels to create classes for prediction.

RapidMiner (RM) was the analytical tool[2] used to evaluated the statistical models. RM can create prediction models from among seven prominent algorithms: Naive Bayes, Generalized Linear Models, Fast Large Margin, Decision, Tree, Random Forrest, Gradient Boosted Trees, and Support Vector Machine). RM also provides further sophisticated-automatic feature engineering, model validation, and tools to analyze model results.

[1] Bins have equal number of issues.

[2] Using an advanced analytical tool vs development using coding languages saved many labor hours.

Models - Groups

The age of an unresolved ticket influences prediction of it's total life-time. This is because - all else being equal - a younger ticket has less developed attributes[3]. To account for this, three age-appropriate modeling groups were created for each of the algorithms. These models where trained with data that has been "alive" for a minimum amount of time.

[3] Fewer comments, attachments, individuals connected, etc. Analogy used was predicting life expectancy of a person based on a resume. Less the resume has, more likely the individual is younger.

Models - Groups (cont. 2)

Models - Groups (cont. 2)

- Model group 1 - Issues that are considered young.

Models - Groups (cont. 2)

- Model group 1 - Issues that are considered young.
- Model group 2 - Issues that where considered middle aged.

Models - Groups (cont. 2)

- Model group 1 - Issues that are considered young.
- Model group 2 - Issues that where considered middle aged.
- Model group 3 - Issues that where considered old.

Models - Train Process

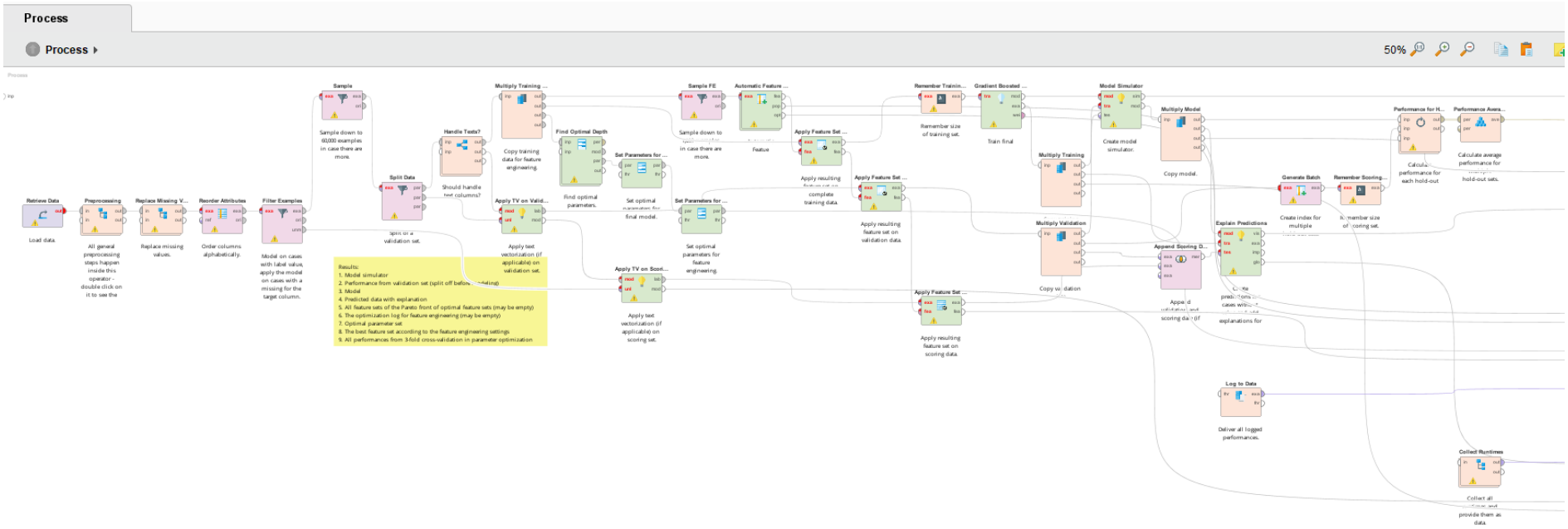


Figure 9: RapidMiner Train Process

Models - Apply Process

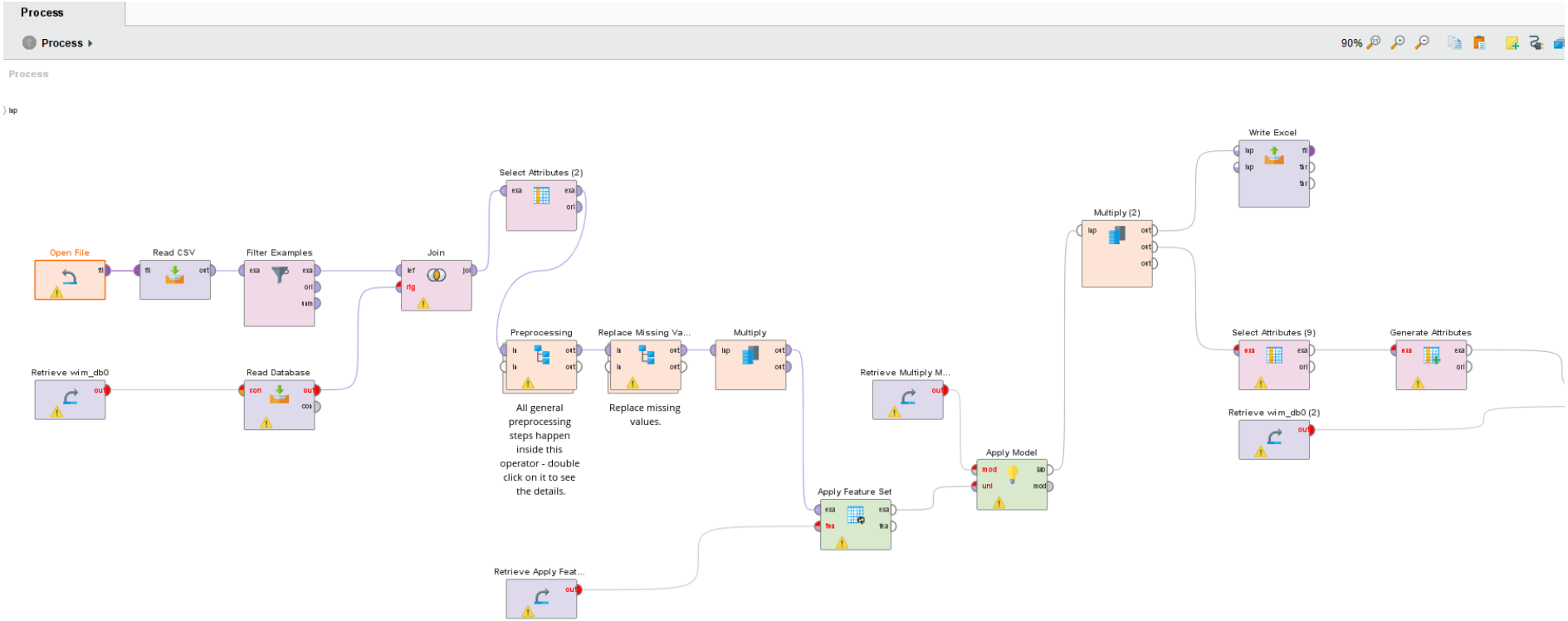


Figure 10: RapidMiner Apply Process

Models - GBT Confusion Matrix

accuracy: 78.49% +/- 2.88% (micro average: 78.49%)

	true range1	true range2	true range3	true range4	class precision
pred. range1	214	23	13	6	83.59%
pred. range2	37	122	17	5	67.40%
pred. range3	13	17	160	30	72.73%
pred. range4	6	3	9	157	89.71%
class recall	79.26%	73.94%	80.40%	79.29%	

Figure 11: GBT Example Confusion Matrix

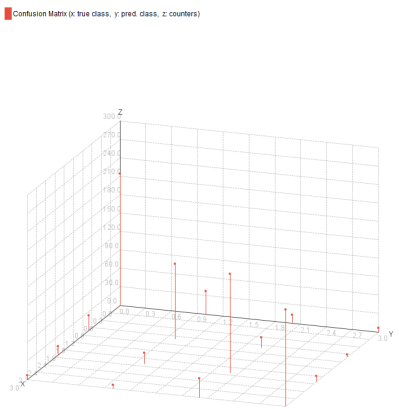


Figure 12: GBT Example Confusion Matrix Plot

Models - GBT Prediction Example

Result History | ExplainPredictionsIOObject (//LCSS/resul [...] et_94/Gradient Boosted Trees/Predictions)

Explanations

Row No.	ttd	prediction(ttd)	confidence(range1)	confidence(range2)	confidence(range3)	confidence(range4)	prj	deploy_me	evc
1	range4	range4	0.006	0.149	0.160	0.685	LCSS	MISSING	10
2	range4	range4	0.005	0.040	0.138	0.817	LCSS	f	10
3	range4	range4	0.005	0.042	0.217	0.736	LCSS	f	1
4	range4	range4	0.010	0.042	0.129	0.819	LCSS	f	10
5	range4	range4	0.006	0.056	0.240	0.698	OCD	MISSING	6
6	range4	range4	0.006	0.052	0.143	0.800	OCD	MISSING	3
7	range4	range4	0.005	0.041	0.132	0.822	MISSING	f	2
8	range4	range4	0.005	0.044	0.253	0.698	LCSS	f	11
9	range4	range3	0.007	0.172	0.543	0.277	OCD	f	6
10	range4	range4	0.006	0.070	0.241	0.683	LCSS	MISSING	65
11	range4	range4	0.288	0.048	0.202	0.462	LCSS	f	9
12	range4	range1	0.630	0.048	0.131	0.191	LCSS	t	3
13	range4	range4	0.005	0.041	0.142	0.812	LCSS	f	7
14	range4	range1	0.436	0.051	0.172	0.341	OCD	f	5
15	range4	range4	0.006	0.063	0.270	0.661	LCSS	f	5
16	range4	range4	0.006	0.056	0.201	0.738	LCSS	f	2
17	range4	range1	0.616	0.045	0.142	0.197	LCSS	f	2
18	range4	range4	0.005	0.040	0.119	0.836	LCSS	MISSING	27
19	range4	range4	0.005	0.041	0.187	0.767	LCSS	f	24
20	range4	range3	0.008	0.242	0.498	0.252	LCSS	MISSING	18
21	range4	range4	0.005	0.113	0.168	0.714	LCSS	f	6
22	range4	range4	0.005	0.040	0.126	0.830	LCSS	f	14
23	range4	range4	0.141	0.044	0.127	0.688	LCSS	f	7
24	range4	range1	0.661	0.039	0.121	0.179	LCSS	f	6
25	range4	range4	0.006	0.058	0.356	0.580	OCD	MISSING	6
26	range4	range3	0.015	0.256	0.506	0.223	LCSS	f	5
27	range4	range4	0.009	0.056	0.279	0.657	LCSS	f	18

Figure 13: GBT Prediction Example

Models - GBT Tree Example

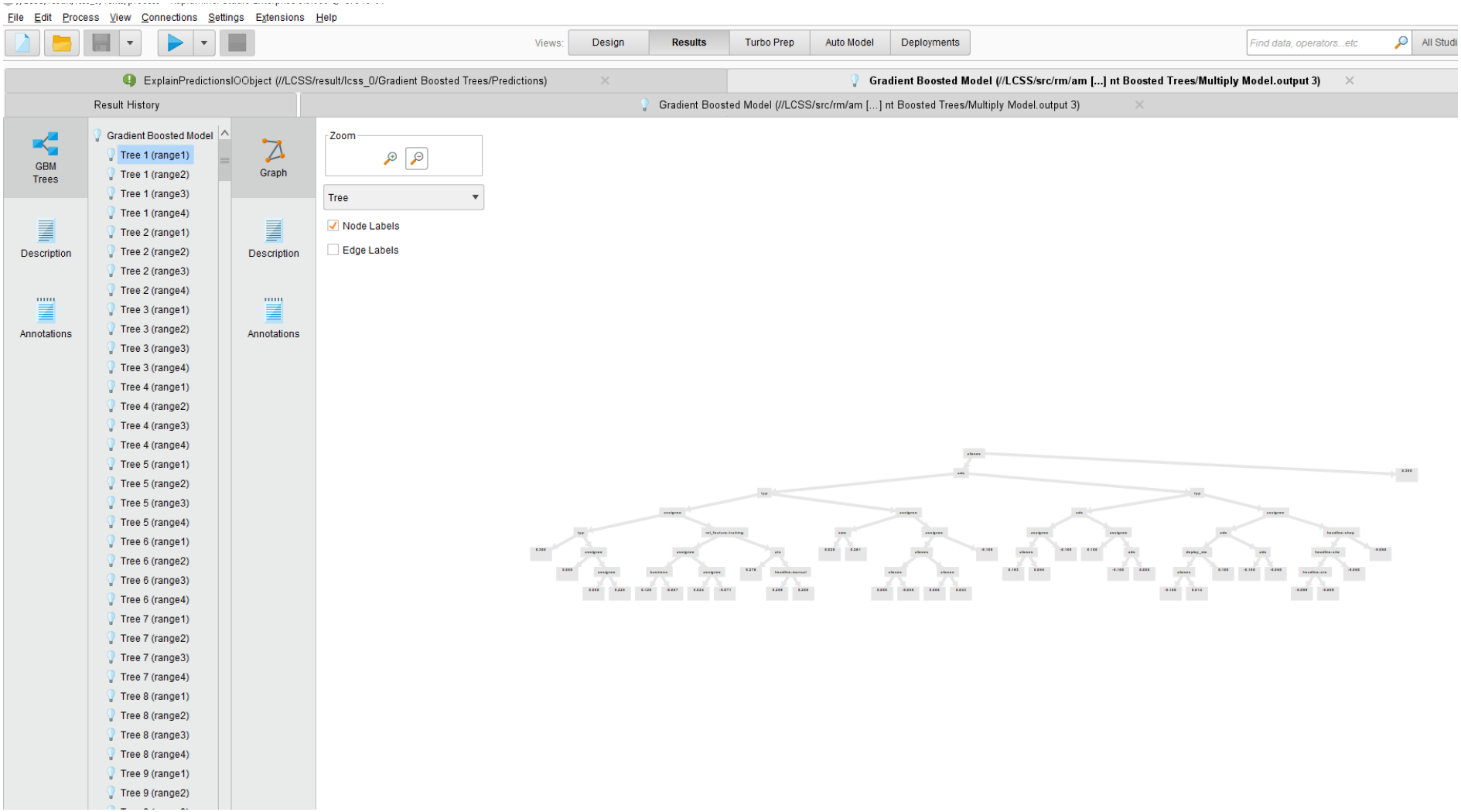


Figure 14: GBT Tree Example

Models - GBT Tree Example Zoom

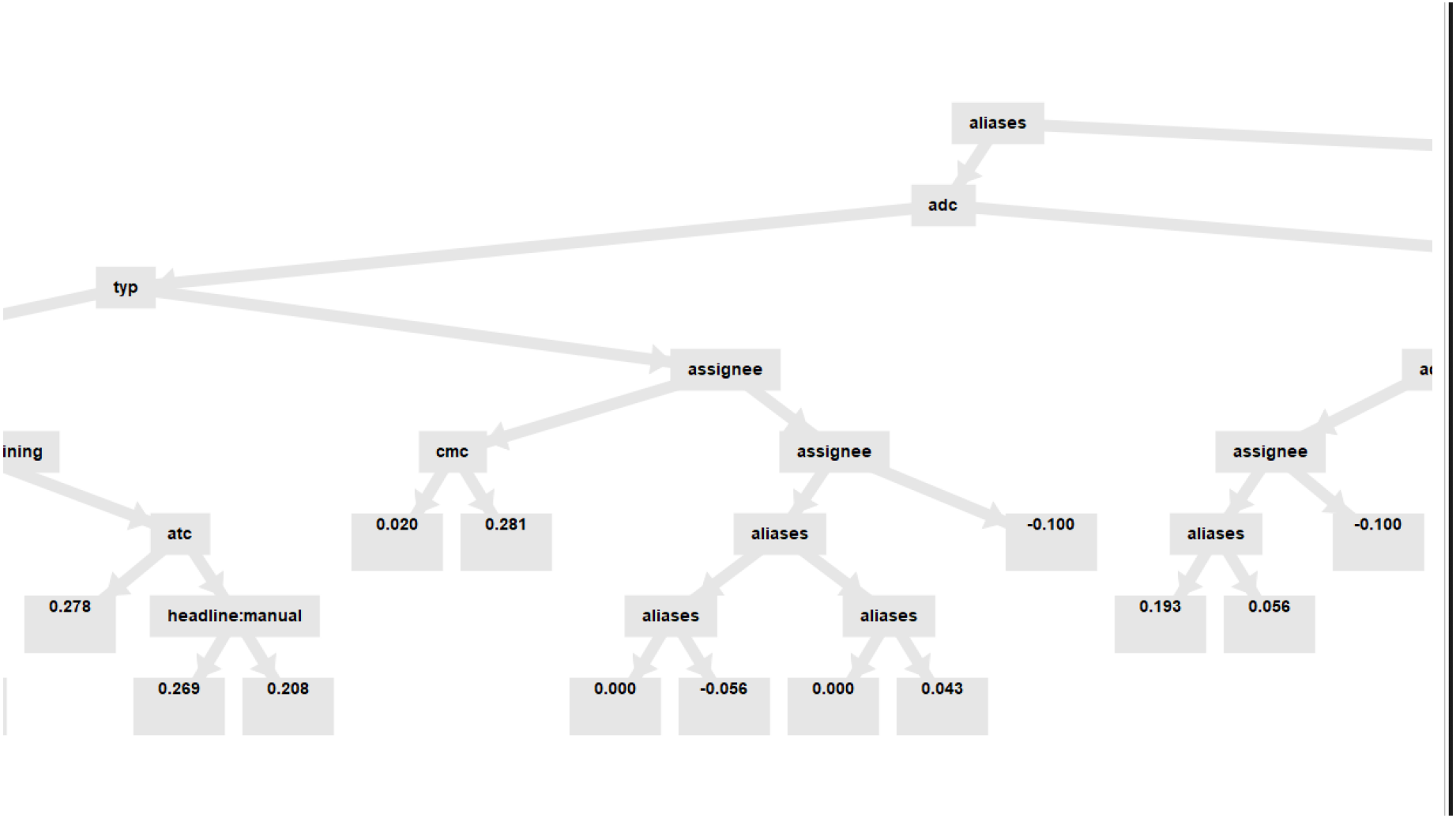


Figure 15: GBT Tree Example Zoom

Models - Schedule Analysis & Final Analysis

So far, the pipeline has predicted creation to resolution duration for each unresolved issue (in weeks). This was not yet ready for the schedule and cost models. The schedule in particularly needed to account for limited (and competing) resources to work on issues. This final analysis determine a single forecast number and guidance of limit on number of issues that can be completed in a time period.

Since each modeling group has somewhat different class ranges, the results where simplified into duration categories that are multiple of 12-weeks (nominally a calendar quarter).

Models - Schedule Analysis & Final Analysis (cont. 1)

The duration forecast is for the entire issue life-time, including pre-work (e.g. process engineering, requirements refinement, waiting that occurs before ticket is allocated to a sprint, testing).

To translate forecast to quarters:

1. Determine the min and max of the model groups class range.
2. Evaluate the PERT formula.
   1. $$\frac{min\_range + 4 * \frac{min\_range + max\_range}{2} + max\_range}{6}$$
3. Evaluate the ceiling of the prior results divided by 12.

Models - Schedule Analysis & Final Analysis (cont. 2)

To provide guidance on a constraint on issue resolutions in a time period, analysis on the historical Jira database provide statistics on resolutions per month:

- Mean = 119 resolutions/month
- Standard deviations = 42 issues/month
- Median = 120 resolutions/month

Working with the developers, an assumption that for every known unresolved issue, another one will be created to satisfy the original issue's intent. So, in practice this number of reduced by half. This information was then passed onto the schedule and cost estimators. The final analysis consisted of a custom heuristic that combined the results with SEER-SEM.

Experimental Results - Data Summary & Key Findings

- Models where trained over 4,000 issues and predicted over 1,000 issues.
- Over 1,000 features where extracted and derived from historical data.
- Examples of model features included issue summary, description, comments, number of times issue was commented, issue type, specific roles fulfilled by a individual, all individuals associated with the issue (commented, mentioned in text), and others.
- Gradient Boosted Trees turned out to be the most accurate trained model.
- Individuals roles and affiliated with the issue seem to be the most important feature.

Conclusion - Future Considerations

- Usage of the more detail historical changes retrieved via the API.
- Model enhancements to:
  - accommodate issue current age
  - refine feature sets
  - simplify pipeline and manual steps
  - port analytical tool's work to tool that doesn't require licensing
- Develop program specific productivity model driven by Function Point size and by non-Function Point Analysis attributes.
- Determine if we can measure the quality of data.
- Refine and formalize model outputs for tracking results and facilitate the hand-off to next estimators.

Conclusion - Summary

- Goal was to create greater fidelity in a cost and schedule estimate.
  - A unique approach was took to help estimate non-Function Point elements of the project.
  - Data was processed through a standard and rigorous pipeline.
  - Several models where generated and evaluated for use.
  - Model forecasts had a final analysis to make results more useable for schedule and cost estimates.
  - These durations aided in the development of a schedule and cost models needed in the acquisition process.

Backup

GBT Example - Introduction

Gradient Boosted Trees is a popular algorithm for modeling various problems. There is much to learn about this approach and how to use it properly. This quick example is just for helping understanding the ideas in this method. This example is from Cory Maklin from Medium [4]. The example looks at predicting housing prices based on a few features.

References:

- https://towardsdatascience.com/machine-learning-part-18-boosting-algorithms-gradient-boosting-in-python-ef5ae6965be4

[4] This example problem seems to be a popular way to explain GBT. Several blogs had the exact or similar datasets and explanation.

GBT Example - Artifact Data

Age	Sqr Footage	Location	Price
5	1500	5	480
11	2030	12	1090
14	1442	6	350
8	2501	4	1310
12	1300	9	400
10	1789	11	500

GBT Example - Calculate Average of Target

2. $$\frac{480 + 1090 + 350 + 1310 + 400 + 500}{6} = 688$$

GBT Example - Calculate Residuals

From the previous step, calculate the residuals by subtracting the actual price from the average.

3

$$residual = actual\_price - avg\_price$$

Age	Sqr Footage	Location	Price	Residuals
5	1500	5	480	-208
11	2030	12	1090	402
14	1442	6	350	-338
8	2501	4	1310	622
12	1300	9	400	-288
10	1789	11	500	-188

GBT Example - Construct Decision Tree

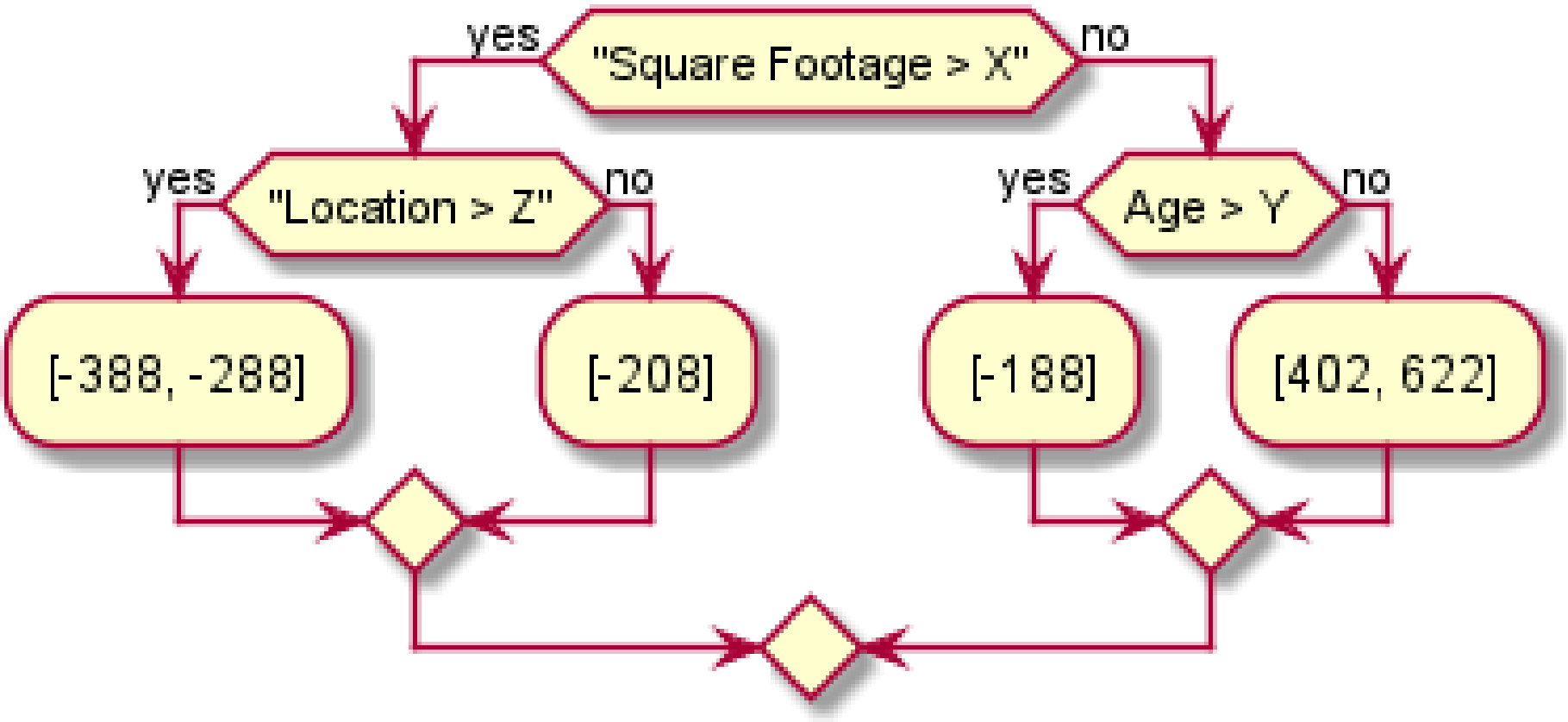
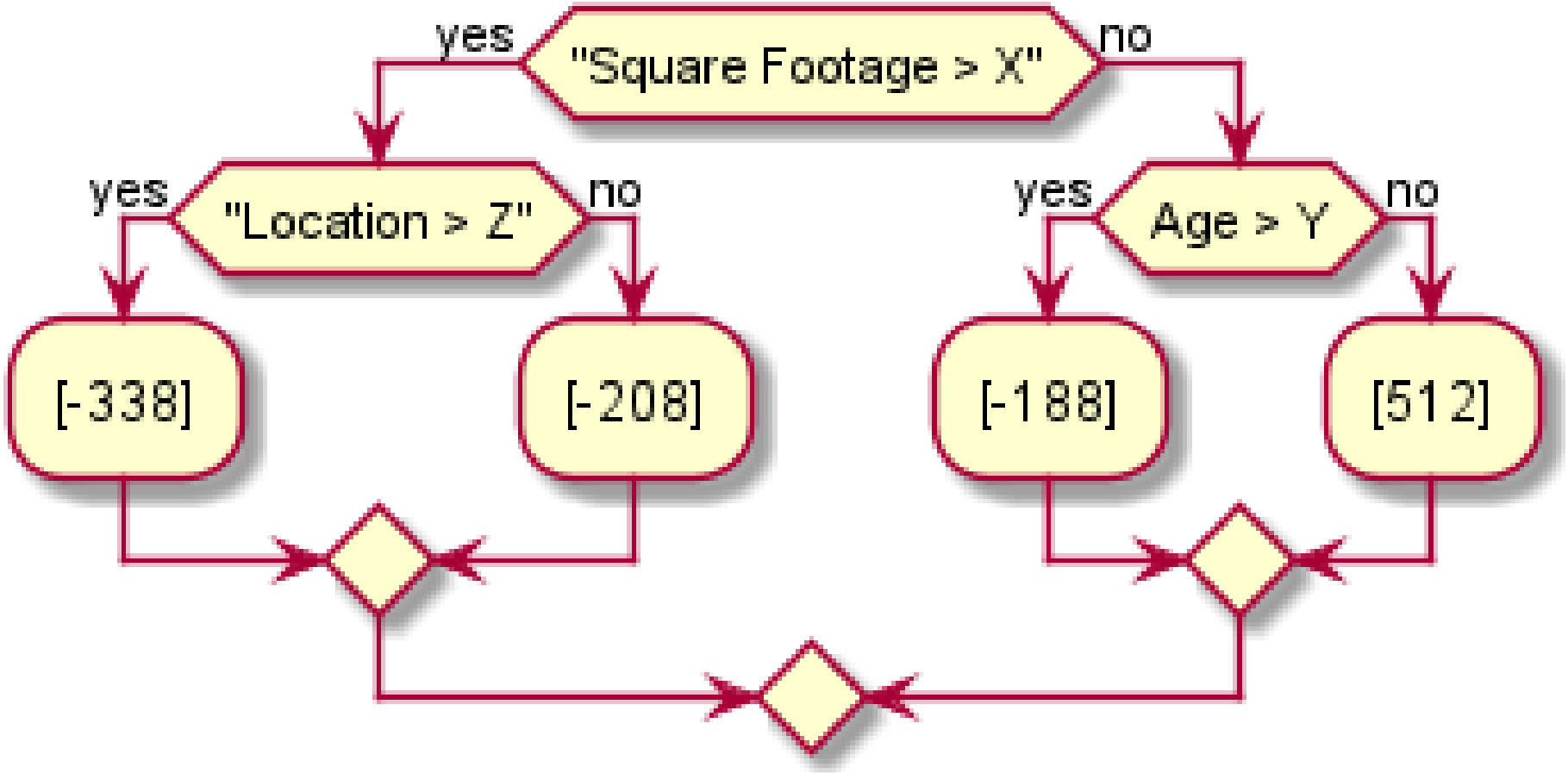


Figure 16: First Decision Tree

GBT Example - Construct Decision Tree Caveat

In the event of multiple residuals in a leaf, compute averages. Thus the prior decision tree is:



GBT Example - Predict Target

4. $$pred\_price = avg\_price + learning\_ratexresidual$$

5. $$654.2 = 688 + 0.1x - 388$$

GBT Example - Compute New Residuals

Calculate new set of residuals by subtracting actual from predictions made in the previous step.

6.

$$residual = actual\_price - pred\_price$$

7.

$$-304.2 = 350 - 654.2$$

Repeat these steps for configurable amount (either iterations or stopping criteria in Hyperparameters).

GBT Example - Apply Model to Data

Final prediction will equal the first step's mean plus the residuals (and learning rate).

$$pred\_price = avg\_price + learning\_rate * \sum_{i}^{n} r_i$$

GBT Example - Regression vs Classification

This GBT example is a regression problem. For Classification, the general approach is the same, but instead of calculating residuals of the label, the label is probabilities of belonging in the class (i.e. binary classification). For multi-class, N-models are generated for each class and the model with the highest probability is chosen.

References:

- https://medium.com/swlh/gradient-boosting-trees-for-classification-a-beginners-guide-596b594a14ea

GBT Example - Hyperparameters

Scikit-learn defines hyper-parameters as parameters that are not directly learnt within the models. Generally hyper-parameters for models are the same among various implementations (e.g. R, Python, analytical tools) but can differ. Hyper-parameter configuration is a full analysis in itself.

Note, RapidMiner uses the H20 GBM implementation.

References:

- https://machinelearningmastery.com/configure-gradient-boosting-algorithm/
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingCl
- docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/gbm.html