MIDS W205

Lab #	10	Lab Title	OpenRefine -- Introduction
Related Module(s)	10	**Goal**	Get you started on OpenRefine and Edit Distance
Last Updated	9/27/15	**Expected duration**	60 minutes

Introduction

OpenRefine is an open source tool for working with bad data. In this Lab we will give you a quick tour of how you can use it to clean data. If you want a more comprehensive tutorial you can follow any of the tutorials listed in the resources section.

We will be using two data sets one from with earthquake data and one with customer complaint data. The first data set is the eq2015 data set which data about earthquakes of magnitude 3 or more during the first 6 months of 2015. You can download the data set here . You can fine a data attribute glossary here The second data set contains customer complaints, you can download that data set here . Please answer the following questions by using OpenRefine.

OpenRefine is to a large extent menu driven. But it also allows you to use a language for doing certain types of transformations.

The basic idea in OpenRefine is that you think of exploring your data in terms of patterns, called facets. OpenRefine also has functions for doing transformations of data. These transformations can be expressed in the language GREL although there are a few other options as well. As an example, you can decide to create a new column based on an existing column but with a transformation applied to the data. GREL is a

Instructions, resources and prerequisites

This Lab has 3 parts. The first two are involves using OpenRefine to clean up some data files. The third one involves calculating the Levenshtein distance between two strings.

Resource	What
http://openrefine.org/	This is where you download OpenRefine.
http://arcadiafalcone.net/GoogleRefineCheatSheets.pdf	A short description of OpenRefine commands.
http://enipedia.tudelft.nl/wiki/OpenRefine_Tutorial	Another tutorial on OpenRefine .
http://davidhuynh.net/spaces/nicar2011/tutorial.pdf	Another tutorial on OpenRefine .

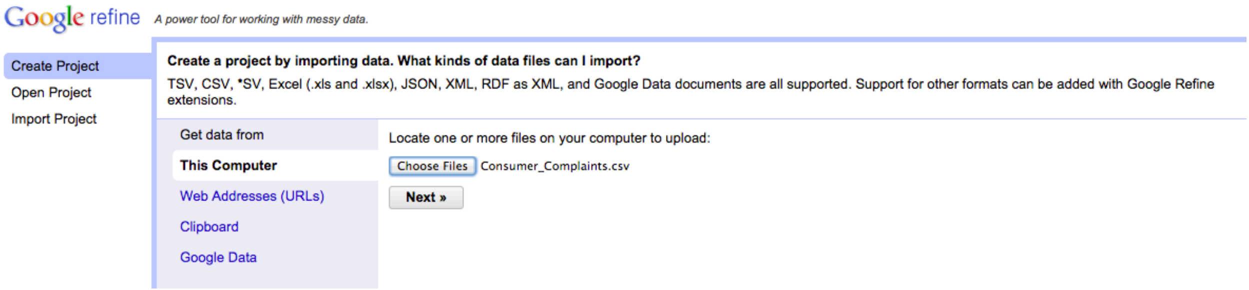
http://schoolofdata.org/handbook/recipes/cleaningdatawithrefine/	Programming guide for the Spark Context object. Here you can find actions available on the Spark Contexts.
https://github.com/OpenRefine/OpenRefine/wiki/General-Refine-Expression-Language	GREL is the language used in OpenRefine for data refinements. This is a reference guide for the GREL language.
http://earthquake.usgs.gov/eart	Explanation of the Earthquake data.
https://pypi.python.org/pypi/python-Levenshtein/0.12.0	A Levenshtein module you can use to check your results in a Python shell.

Cleaning Data with OpenRefine.

Step-1. Wrangling the Customer Complaints Data

Uploading data

After you started OpenRefine you can pick a data set. For this first step choose the Customer Complaints Data set.



A power tool for working with messy data.

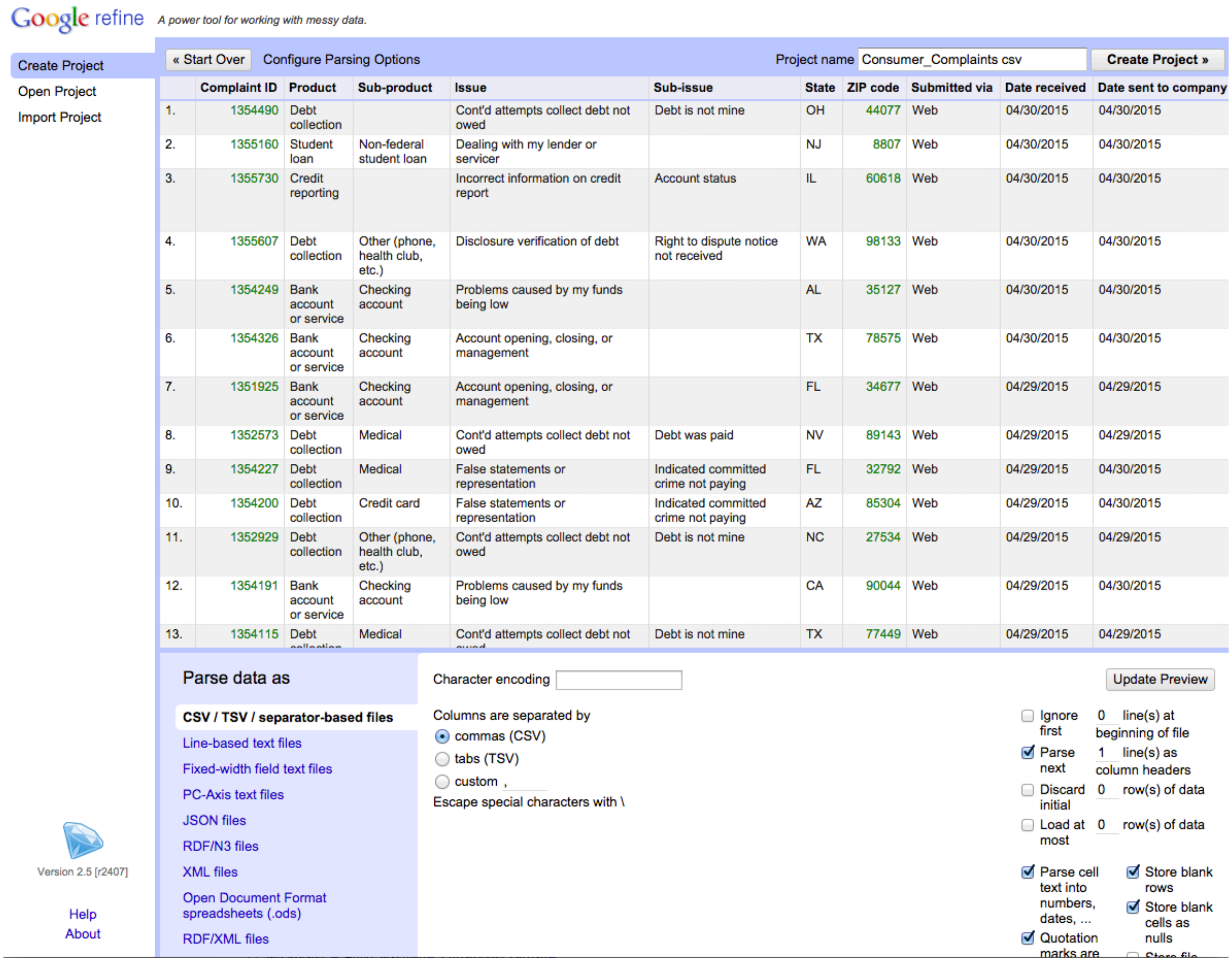
Create a project by importing data. What kinds of data files can I import?

TSV, CSV, *SV, Excel (.xls and .xlsx), JSON, XML, RDF as XML, and Google Data documents are all supported. Support for other formats can be added with Google Refine extensions.

Locate one or more files on your computer to upload:

Consumer_Complaints.csv

Once the data is read you can inspect it. In this case it looks ok. But lets say that it would have been tab separated, then OpenRefine would not have read it correctly. You have the opportunity to look at the data here and confirm it is ok. Ion this case we think it looks good and we click the “Create Project” button.



Configure Parsing Options

Project name Consumer_Complaints csv

	Complaint ID	Product	Sub-product	Issue	Sub-issue	State	ZIP code	Submitted via	Date received	Date sent to company
1.	1354490	Debt collection		Cont'd attempts collect debt not owed	Debt is not mine	OH	44077	Web	04/30/2015	04/30/2015
2.	1355160	Student loan	Non-federal student loan	Dealing with my lender or servicer		NJ	8807	Web	04/30/2015	04/30/2015
3.	1355730	Credit reporting		Incorrect information on credit report	Account status	IL	60618	Web	04/30/2015	04/30/2015
4.	1355607	Debt collection	Other (phone, health club, etc.)	Disclosure verification of debt	Right to dispute notice not received	WA	98133	Web	04/30/2015	04/30/2015
5.	1354249	Bank account or service	Checking account	Problems caused by my funds being low		AL	35127	Web	04/30/2015	04/30/2015
6.	1354326	Bank account or service	Checking account	Account opening, closing, or management		TX	78575	Web	04/30/2015	04/30/2015
7.	1351925	Bank account or service	Checking account	Account opening, closing, or management		FL	34677	Web	04/29/2015	04/29/2015
8.	1352573	Debt collection	Medical	Cont'd attempts collect debt not owed	Debt was paid	NV	89143	Web	04/29/2015	04/29/2015
9.	1354227	Debt collection	Medical	False statements or representation	Indicated committed crime not paying	FL	32792	Web	04/29/2015	04/30/2015
10.	1354200	Debt collection	Credit card	False statements or representation	Indicated committed crime not paying	AZ	85304	Web	04/29/2015	04/30/2015
11.	1352929	Debt collection	Other (phone, health club, etc.)	Cont'd attempts collect debt not owed	Debt is not mine	NC	27534	Web	04/29/2015	04/29/2015
12.	1354191	Bank account or service	Checking account	Problems caused by my funds being low		CA	90044	Web	04/29/2015	04/30/2015
13.	1354115	Debt	Medical	Cont'd attempts collect debt not	Debt is not mine	TX	77449	Web	04/29/2015	04/29/2015

Parse data as

- **CSV / TSV / separator-based files**
- Line-based text files
- Fixed-width field text files
- PC-Axis text files
- JSON files
- RDF/N3 files
- XML files
- Open Document Format spreadsheets (.ods)
- RDF/XML files

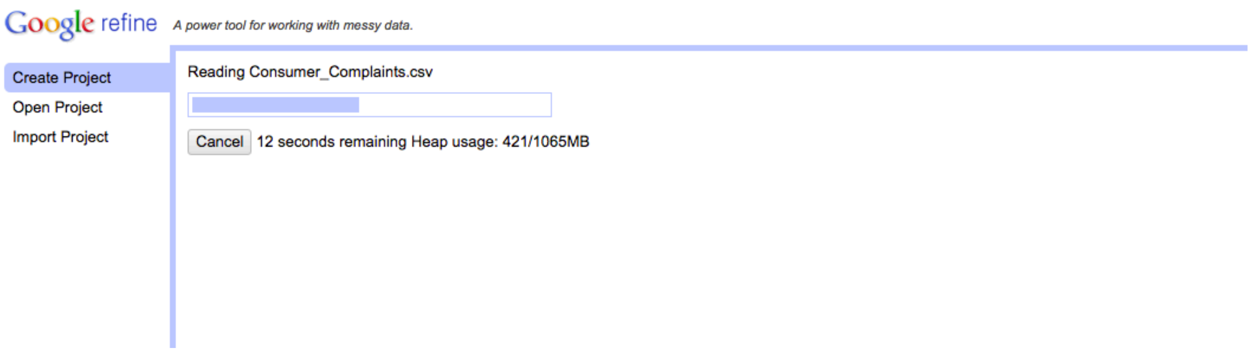
Character encoding

Columns are separated by
● commas (CSV)
○ tabs (TSV)
○ custom ,
Escape special characters with \

□ Ignore first 0 line(s) at beginning of file
☑ Parse next 1 line(s) as column headers
□ Discard initial 0 row(s) of data
□ Load at most 0 row(s) of data
☑ Parse cell text into numbers, dates, ...
☑ Quotation marks are
☑ Store blank rows
☑ Store blank cells as nulls
□ Store file

Creating a project

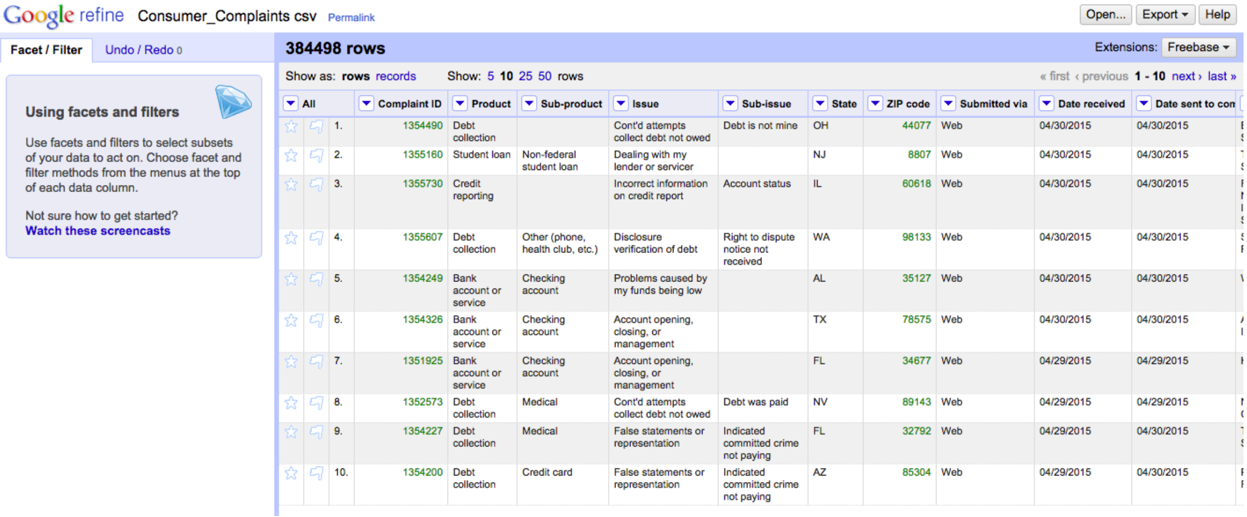
Creating the project can take a little time.



Reading Consumer_Complaints.csv

12 seconds remaining Heap usage: 421/1065MB

Once the project is created you can see that it has 384498 rows.



Consumer_Complaints csv

Using facets and filters

Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.

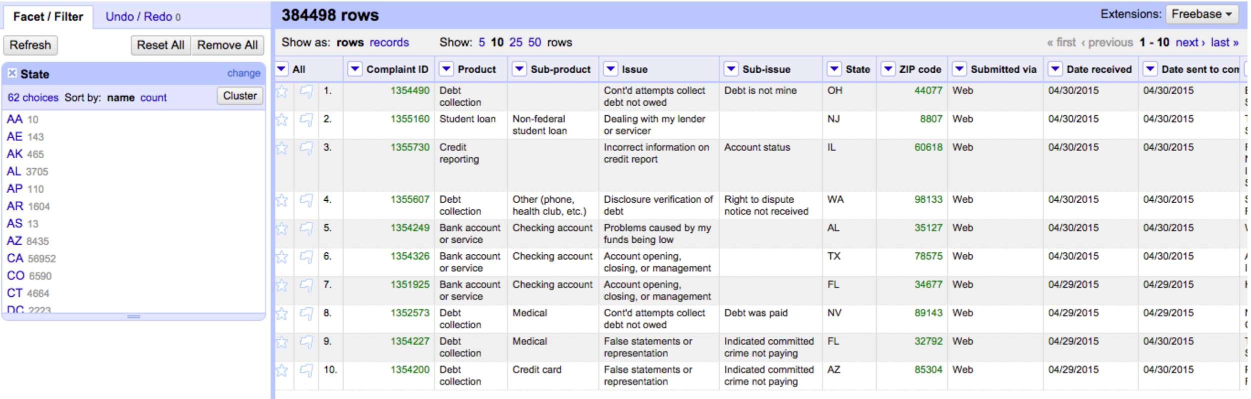
Not sure how to get started?
Watch these screencasts

384498 rows

All	Complaint ID	Product	Sub-product	Issue	Sub-issue	State	ZIP code	Submitted via	Date received	Date sent to com
1.	1354490	Debt collection		Cont'd attempts collect debt not owed	Debt is not mine	OH	44077	Web	04/30/2015	04/30/2015
2.	1355160	Student loan	Non-federal student loan	Dealing with my lender or servicer		NJ	8807	Web	04/30/2015	04/30/2015
3.	1355730	Credit reporting		Incorrect information on credit report	Account status	IL	60618	Web	04/30/2015	04/30/2015
4.	1355607	Debt collection	Other (phone, health club, etc.)	Disclosure verification of debt	Right to dispute notice not received	WA	98133	Web	04/30/2015	04/30/2015
5.	1354249	Bank account or service	Checking account	Problems caused by my funds being low		AL	35127	Web	04/30/2015	04/30/2015
6.	1354326	Bank account or service	Checking account	Account opening, closing, or management		TX	78575	Web	04/30/2015	04/30/2015
7.	1351925	Bank account or service	Checking account	Account opening, closing, or management		FL	34677	Web	04/29/2015	04/29/2015
8.	1352573	Debt collection	Medical	Cont'd attempts collect debt not owed	Debt was paid	NV	89143	Web	04/29/2015	04/29/2015
9.	1354227	Debt collection	Medical	False statements or representation	Indicated committed crime not paying	FL	32792	Web	04/29/2015	04/30/2015
10.	1354200	Debt collection	Credit card	False statements or representation	Indicated committed crime not paying	AZ	85304	Web	04/29/2015	04/30/2015

Check states with text facet

If you select text facet for state you will see a summary in the left column pane. It indicates you we have 62 different state value (?). Try to figure out why.



State

62 choices Sort by: name count

AA	10
AE	143
AK	465
AL	3705
AP	110
AR	1604
AS	13
AZ	8435
CA	56952
CO	6590
CT	4664
DC	2223

384498 rows

All	Complaint ID	Product	Sub-product	Issue	Sub-issue	State	ZIP code	Submitted via	Date received	Date sent to con
1.	1354490	Debt collection		Cont'd attempts collect debt not owed	Debt is not mine	OH	44077	Web	04/30/2015	04/30/2015
2.	1355160	Student loan	Non-federal student loan	Dealing with my lender or servicer		NJ	8807	Web	04/30/2015	04/30/2015
3.	1355730	Credit reporting		Incorrect information on credit report	Account status	IL	60618	Web	04/30/2015	04/30/2015
4.	1355607	Debt collection	Other (phone, health club, etc.)	Disclosure verification of debt	Right to dispute notice not received	WA	98133	Web	04/30/2015	04/30/2015
5.	1354249	Bank account or service	Checking account	Problems caused by my funds being low		AL	35127	Web	04/30/2015	04/30/2015
6.	1354326	Bank account or service	Checking account	Account opening, closing, or management		TX	78575	Web	04/30/2015	04/30/2015
7.	1351925	Bank account or service	Checking account	Account opening, closing, or management		FL	34677	Web	04/29/2015	04/29/2015
8.	1352573	Debt collection	Medical	Cont'd attempts collect debt not owed	Debt was paid	NV	89143	Web	04/29/2015	04/29/2015
9.	1354227	Debt collection	Medical	False statements or representation	Indicated committed crime not paying	FL	32792	Web	04/29/2015	04/30/2015
10.	1354200	Debt collection	Credit card	False statements or representation	Indicated committed crime not paying	AZ	85304	Web	04/29/2015	04/30/2015

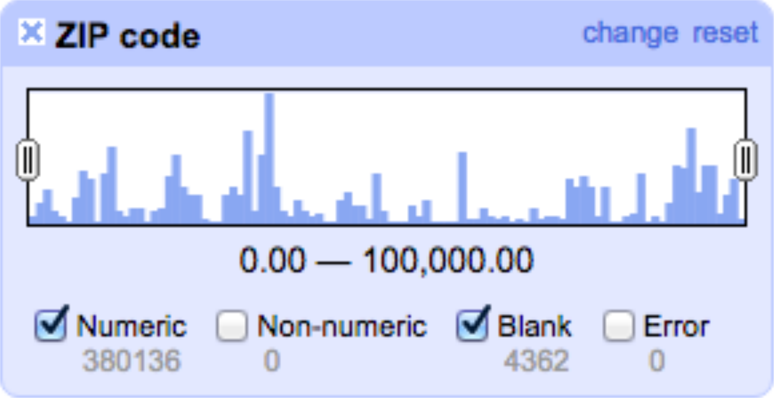
- A1: How many rows are missing value in the state column? Explain how you came up with the number?

Checking zip codes

Try the text facet on zip codes, what happens? You can see that there Is 24748 different zip codes in this data set. Is that reasonable? Eye ball the data, does all zip codes look valid?

Now try the numeric facet. With the numeric facet the zip code attribute is treated is a numeric value. What would you say the scalar type is for zip codes, can be treated as a numeric attribute? Histogram below shows the distribution when the attribute is treated as numeric. By un checking numeric you can get a list of row that are missing.

- A2: How many rows with missing zip codes do you have?


ZIP code change reset

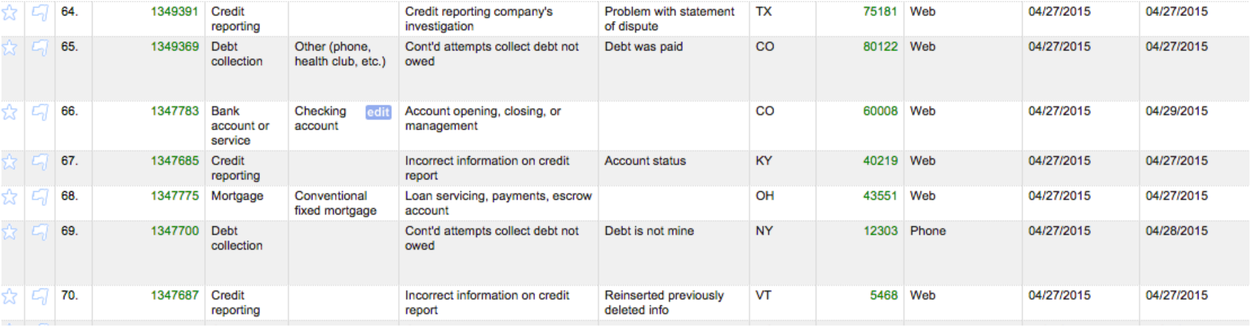
0.00 — 100,000.00

☑ Numeric 380136 □ Non-numeric 0 ☑ Blank 4362 □ Error 0

One way of filling in missing values if to take the previous value and use that. In OpenRefine it is called fill-down. Find a row that is blank. Apply fill down to the fill down by:

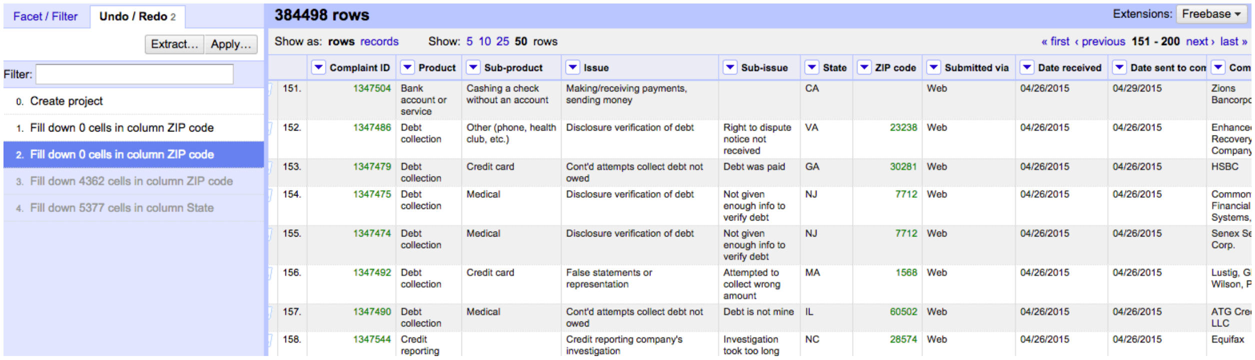
Edit Cell->Fill Down

What happened to the empty cell.



64.	1349391	Credit reporting		Credit reporting company's investigation	Problem with statement of dispute	TX	75181	Web	04/27/2015	04/27/2015
65.	1349369	Debt collection	Other (phone, health club, etc.)	Cont'd attempts collect debt not owed	Debt was paid	CO	80122	Web	04/27/2015	04/27/2015
66.	1347783	Bank account or service	Checking account edit	Account opening, closing, or management		CO	60008	Web	04/27/2015	04/29/2015
67.	1347685	Credit reporting		Incorrect information on credit report	Account status	KY	40219	Web	04/27/2015	04/27/2015
68.	1347775	Mortgage	Conventional fixed mortgage	Loan servicing, payments, escrow account		OH	43551	Web	04/27/2015	04/27/2015
69.	1347700	Debt collection		Cont'd attempts collect debt not owed	Debt is not mine	NY	12303	Phone	04/27/2015	04/28/2015
70.	1347687	Credit reporting		Incorrect information on credit report	Reinserted previously deleted info	VT	5468	Web	04/27/2015	04/27/2015

If you need to undo the operation, switch to the Undo/Redo tab. Select the a previous state for the data. In this example I went back to state 2. As you can see in this screen shot row 151 has a missing zip code, so presumely the fill downs for Zip code and State has been un done.



0. Create project
1. Fill down 0 cells in column ZIP code
2. Fill down 0 cells in column ZIP code
3. Fill down 4362 cells in column ZIP code
4. Fill down 5377 cells in column State

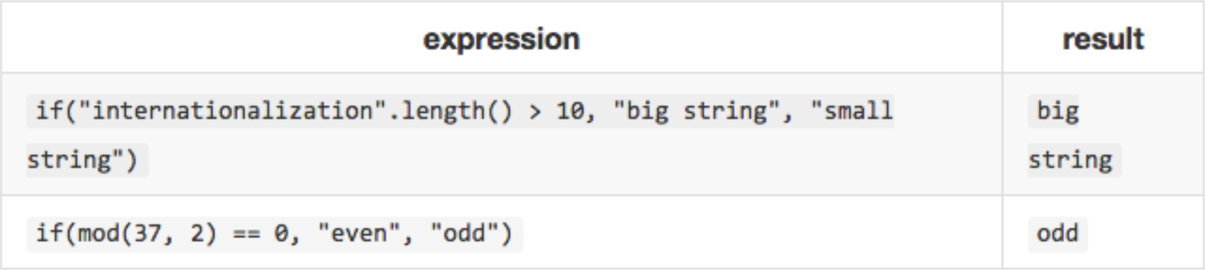
384498 rows

Show as: rows records   Show: 5 10 25 50 rows

	Complaint ID	Product	Sub-product	Issue	Sub-issue	State	ZIP code	Submitted via	Date received	Date sent to con	Com
151.	1347504	Bank account or service	Cashing a check without an account	Making/receiving payments, sending money		CA		Web	04/26/2015	04/29/2015	Zions Bancorpc
152.	1347486	Debt collection	Other (phone, health club, etc.)	Disclosure verification of debt	Right to dispute notice not received	VA	23238	Web	04/26/2015	04/26/2015	Enhance Recovery Company
153.	1347479	Debt collection	Credit card	Cont'd attempts collect debt not owed	Debt was paid	GA	30281	Web	04/26/2015	04/26/2015	HSBC
154.	1347475	Debt collection	Medical	Disclosure verification of debt	Not given enough info to verify debt	NJ	7712	Web	04/26/2015	04/26/2015	Common Financial Systems,
155.	1347474	Debt collection	Medical	Disclosure verification of debt	Not given enough info to verify debt	NJ	7712	Web	04/26/2015	04/26/2015	Senex Se Corp.
156.	1347492	Debt collection	Credit card	False statements or representation	Attempted to collect wrong amount	MA	1568	Web	04/26/2015	04/26/2015	Lustig, G Wilson, P
157.	1347490	Debt collection	Medical	Cont'd attempts collect debt not owed	Debt is not mine	IL	60502	Web	04/26/2015	04/26/2015	ATG Cre LLC
158.	1347544	Credit reporting		Credit reporting company's investigation	Investigation took too long	NC	28574	Web	04/26/2015	04/26/2015	Equifax

● A3: Transforming zip code column. Lets create a new column called “ZipCode5” with all zip codes that contains 5 digits preserved. All other rows should have the zip code 99999. (Technically speaking the 4-digit zip codes may be valid zip codes, we do this more to illustrate transformations).

Transformations are expressed in some language. OpenRefine supports a few alternatives, we will be using GREL. You can find a link to a language reference in the resources section. For this simple transnformation we will be using an if statement.



expression	result
`if("internationalization".length() > 10, "big string", "small string")`	`big string`
`if(mod(37, 2) == 0, "even", "odd")`	`odd`

For the Zip code column select:

Edit Column -> Add column based on this column.

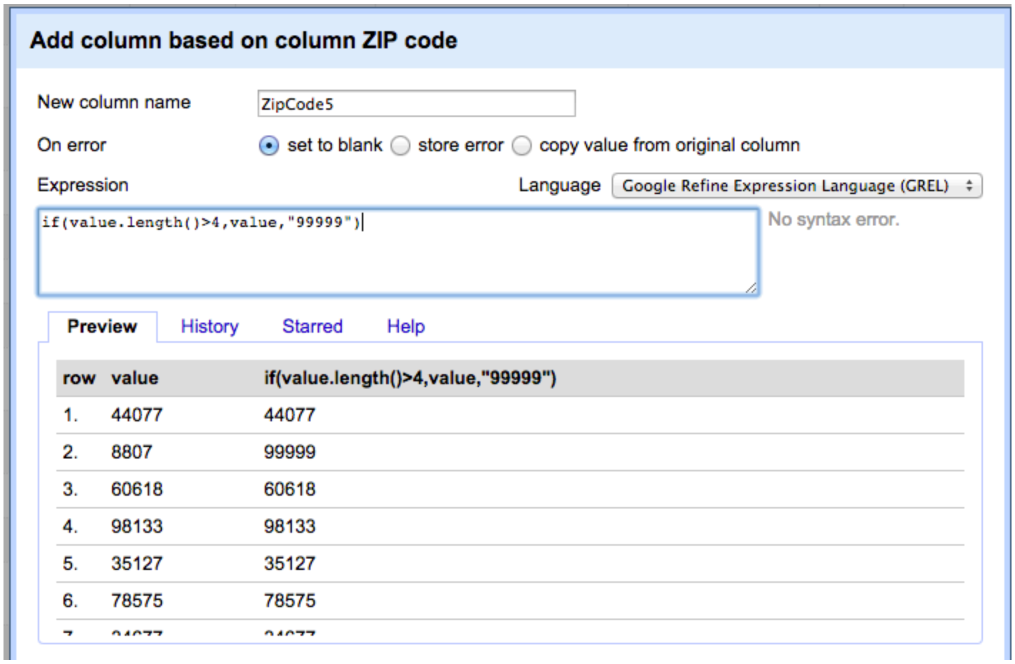
You will get the dialogue below. Insert the name of the new column and the expression:

If(value.length() > 4, value, “99999”)

This expressen states that if the length of value is more than 4 insert value, otherwise insert the string “99999”.

Look at the result, this it do what you wanted? That seems to be wrong with that? What happens if you instead insert a numeric value using the expression:

If(value.length() > 4, value, 99999)



Add column based on column ZIP code

New column name: ZipCode5

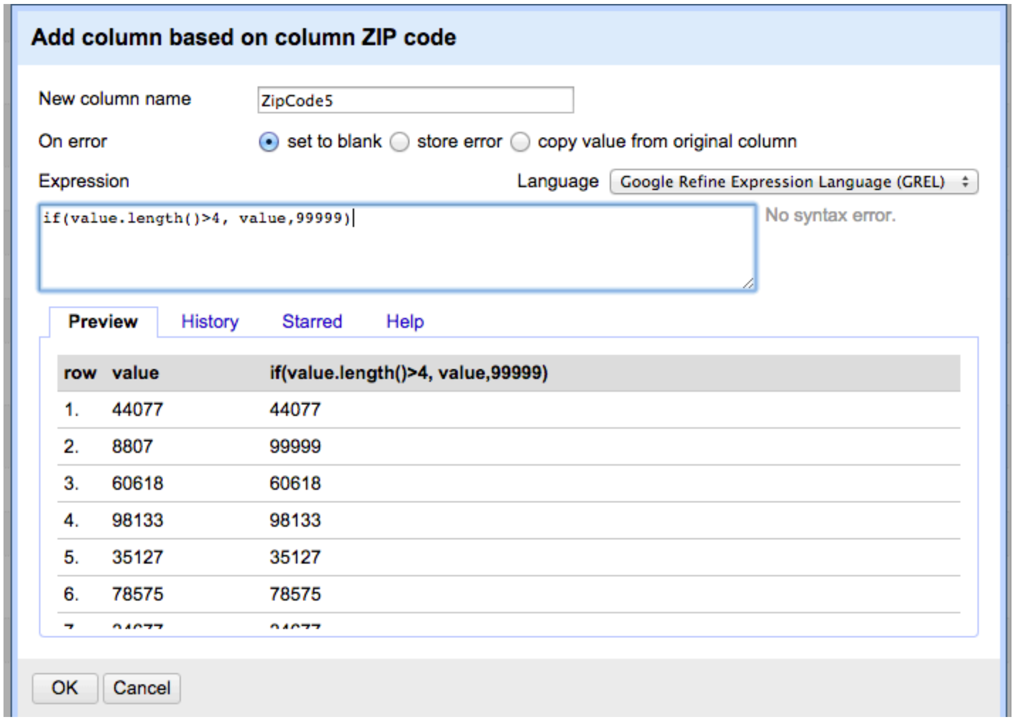
On error: ● set to blank ○ store error ○ copy value from original column

Expression — Language: Google Refine Expression Language (GREL)

```
if(value.length()>4,value,"99999")
```

No syntax error.

row	value	if(value.length()>4,value,"99999")
1.	44077	44077
2.	8807	99999
3.	60618	60618
4.	98133	98133
5.	35127	35127
6.	78575	78575



Add column based on column ZIP code

New column name: ZipCode5

On error: ● set to blank ○ store error ○ copy value from original column

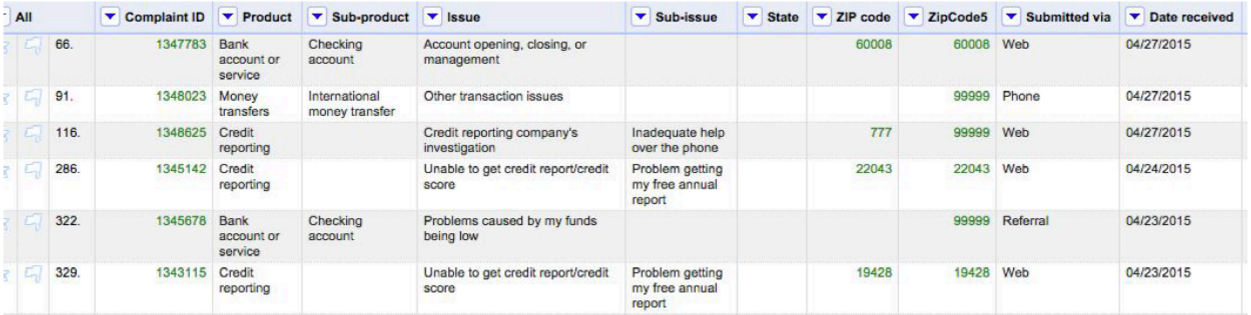
Expression — Language: Google Refine Expression Language (GREL)

```
if(value.length()>4, value,99999)
```

No syntax error.

row	value	if(value.length()>4, value,99999)
1.	44077	44077
2.	8807	99999
3.	60618	60618
4.	98133	98133
5.	35127	35127
6.	78575	78575

You should have the same type for all cells in the created column.
Example of result:



All	Complaint ID	Product	Sub-product	Issue	Sub-issue	State	ZIP code	ZipCode5	Submitted via	Date received
66.	1347783	Bank account or service	Checking account	Account opening, closing, or management			60008	60008	Web	04/27/2015
91.	1348023	Money transfers	International money transfer	Other transaction issues				99999	Phone	04/27/2015
116.	1348625	Credit reporting		Credit reporting company's investigation	Inadequate help over the phone		777	99999	Web	04/27/2015
286.	1345142	Credit reporting		Unable to get credit report/credit score	Problem getting my free annual report		22043	22043	Web	04/24/2015
322.	1345678	Bank account or service	Checking account	Problems caused by my funds being low				99999	Referral	04/23/2015
329.	1343115	Credit reporting		Unable to get credit report/credit score	Problem getting my free annual report		19428	19428	Web	04/23/2015

● A4: If you consider all zip codes less than 99999 valid zip codes. How many valid and invalid zip codes do you have respectively.

Step-2. Cleaning up eq2015 Data.

Upload the data

● A5: For column “nst” fill in missing values.

● A6: Clean up the place column so that it has state or country name depending on what is in the text.

● A7: From the column “updated” extract the Date without time into a new column called “eventdate”

● A8: Run cluster en edit on “location” column. Run nearest neighbor and levenshtein distance. Answer the following questions:

  ○ Does it make sense to merge detected values?

  ○ Why or why not?

● A9: Try to do nearest neighbor clustering on “place’ column.

  ○ What happens?

  ○ Explain why it is happening.

Step-3 Levenshtein Distance

In this lab we will go over a simple example of Levenshtein distance calculation. We will then ask you to calculate the distance for two strings gumbarrel" and "gunbarell". We will point you to a simple implementation of the Levenshtein distance that you can use to check your result.

Installing Levenshtein python module

The following steps will just clone and build a Python Levenshtein module in a directory. It does not fully install the module. But you can use it to run a distance function from your shell to check your results.

```
$ git clone https://github.com/ztane/python-Levenshtein/
$ cd python-Levenshtein/
$ python setup.py build
$ cd Levenshtein/
$ python
>>> from  Levenshtein import *
>>> distance("hej","hei")
1
>>> distance("monthgomery st","montgomery street")
5
```

Example: Levenshtein Calculation

Lets step through the calculation of distance between the words `LOYOLA` and `LAJOLLA`. We will denote a cell with the `d[i,j]`, where `i` is the row and `j` is the column. The first column and row indicates the indices we will be using.

As a reminder the algorithms is as follows:

Denote the column by c and row by r. We have n rows and m columns. d[i,j] denotes the value on row i and columns j.
$cost\ [i,j] = 1\ if\ c[i]\ != r[j]$
$cost\ [i,j] = 0\ if\ c[i] == r[j]$
d[i,j] is to be set to the minimum of: d[i-1,j]+1 or d[i,j-1]+1 or d[i-1, j-1]+cost[I,j]
Distance is found in the resulting value d[n,m]

We first set up the matrix. The blue row and column just contains the i and j values. We then insert values `0-m` in first row `i==1` and `0-n` in the column `j==1`.

		1	2	3	4	5	6	7
			L	O	Y	O	L	A
1		0	1	2	3	4	5	6
2	L	1						
3	A	2						
4	J	3						
5	O	4						
6	L	5						
7	L	6						
8	A	7						

Lets calculate the d[i,2]. Meaning the value for each row in the column 2.

d[2,2], cost is 0, minimum is d[1,1]+0=>0
d[3,2], cost is 1, minimum is d[2,2]+1=>1
d[4,2], cost is 1, minimum is d[3,2]+1=>2
d[5,2], cost is 1, minimum is d[4,2]+1=>3
d[6,2], cost is 0, minimum is d[5,1]+0=>4, or d[5,2]+1
d[7,2], cost is 0, minimum is d[6,2]+0=>5, or d[6,2]+1
d[8,2], cost is 1, minimum is d[7,2]+1=>6

		1	2	3	4	5	6	7
			L	O	Y	O	L	A
1		0	1	2	3	4	5	6
2	L	1	0					
3	A	2	1					
4	J	3	2					
5	O	4	3					
6	L	5	4					
7	L	6	5					
8	A	7	6					

Lets calculate the d[i,3]. Meaning the value for each row in the column 3.

d[2,3], cost is 1, minimum is d[2,2]+1=>1
d[3,3], cost is 1, minimum is d[2,2]+1=>1
d[4,3], cost is 1, minimum is d[3,2]+1=>2, or d[3,3]+1
d[5,3], cost is 0, minimum is d[4,2]+0=>2
d[6,3], cost is 1, minimum is d[5,3]+1=>3
d[7,3], cost is 1, minimum is d[6,2]+1=>4, or d[6,3]+1
d[8,3], cost is 1, minimum is d[7,2]+1=>4

		1	2	3	4	5	6	7
			L	O	Y	O	L	A

1		0	1	2	3	4	5	6
2	L	1	0	1				
3	A	2	1	1				
4	J	3	2	2				
5	O	4	3	2				
6	L	5	3	3				
7	L	6	3	4				
8	A	7	4	4				

Now if the you do the same thing for the rest of the columns we will get the following matrix. You see the calculated edit distance in the cell d[8,7].

		1	2	3	4	5	6	7
			L	O	Y	O	L	A
1		0	1	2	3	4	5	6
2	L	1	0	1	2	3	4	5
3	A	2	1	1	2	3	4	4
4	J	3	2	2	2	3	4	5
5	O	4	3	2	3	2	3	4
6	L	5	3	3	3	3	2	3
7	L	6	3	4	4	4	3	3
8	A	7	4	4	5	5	4	**3**

If you use the Levenshtein function to check the result you will see

```
>>> distance("loyola","lajolla")
3
```

So we are assuming we got the calculation right.

Calculation: gumbarrel v.s gunbarell

Now calculate the edit distance of the words: “gumbarrel” and “gunbarell". Use the python levenshtein function to check you result.

		1	2	3	4	5	6	7	8	9	10
			G	U	M	B	A	R	R	E	L
1		0	1	2	3	4	5	6	7	8	9
2	G	1									
3	U	2									
4	N	3									
5	B	4									
6	A	5									
7	R	6									
8	E	7									
9	L	8									

10	L	9									

SUBMISSION 3: submit a representation of the resulting matrix from the leveshtein edit distance calculation. The resulting value should be correct.