

Alice Wu Lim, PhD
Data Science
Fellow
City of Hope

Genomic cluster project

A large yellow triangle is positioned in the bottom right corner of the slide, pointing towards the top right.

What is Patient-similarity?

- Oftentimes in healthcare, predictions are made about the “average patient”.
 - Patient-similarity models make predictions about individual patients by identifying patients which are like an index patient and tailoring the predictions based on these sets of similar patients.
 - At City of Hope, satellite hospitals can use patient-similarity to match their patients to patient data from the main hospital
-
- See <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5357318/>

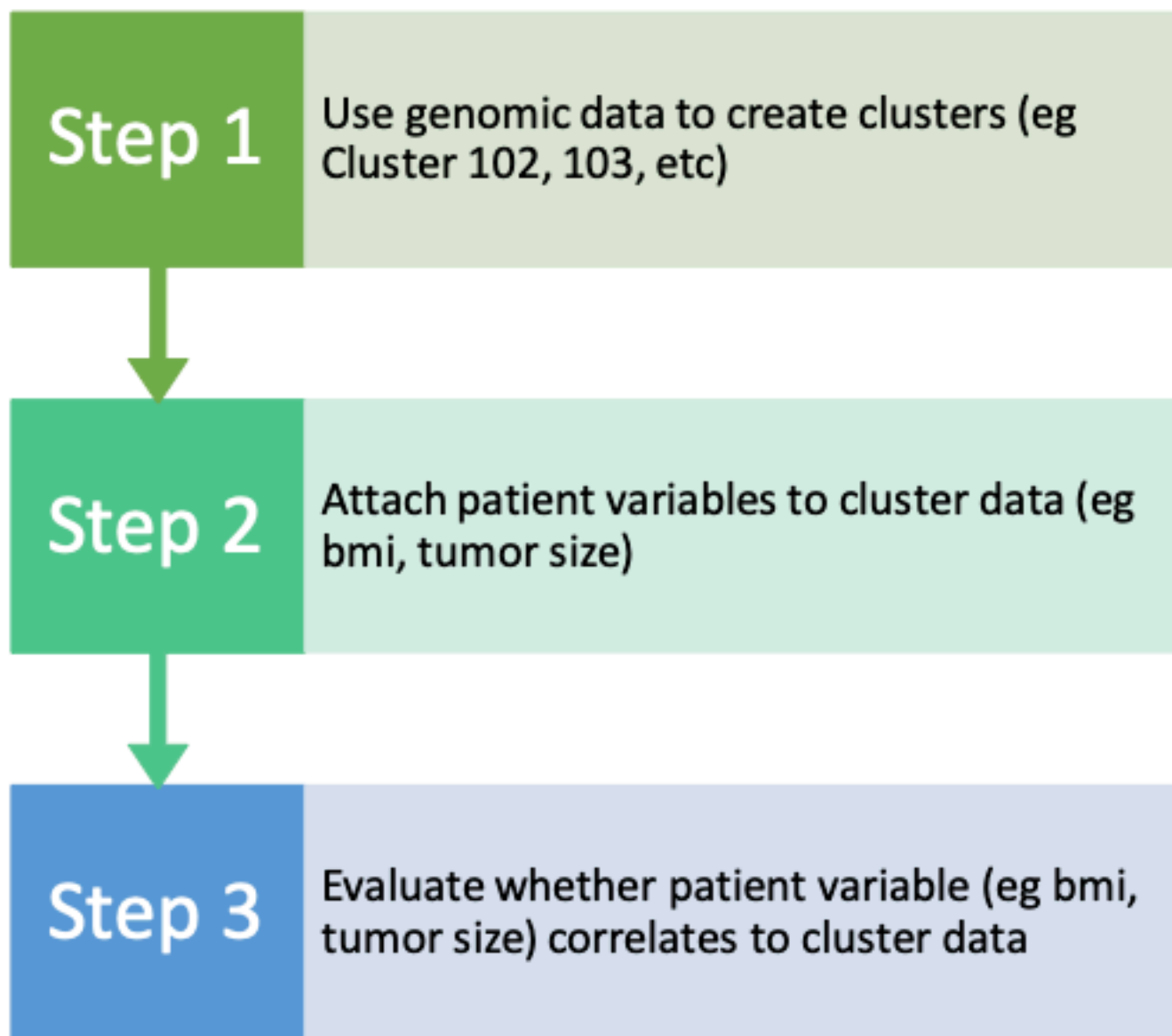
Goal:

Understand what genomic clusters mean with respect to patient variables. Specifically, find patient variables which correlate to genomic data.

ERBB2	GRB7	PNMT	PGAP3	STARD3	IKZF3	...	HS6ST1	CUBN	PRKDC	CSMD3	CRIPAK	CARNS1	DHX40	DNAH9	cluster	uuid
0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

Here, the first 14 columns shown are the genomic data. The 'cluster' column was created by k-means clustering the data. There are 12 possible 'clusters'. The 'uuid' is the individual patient's unique id.

Initial Steps:



Types of patient data

cancerstaging

consent

disease

registrydiagnosis

encounter

encountercode

familyhistory

genomicsample

imaging

insurance

labresults

medicalhistory

medication

nlpbiomarkers

nlpdiseasestatus

nlp specimenreport

order

pathology

protocolparticipation

lvsample

socialhistory

surgery

Example of getting bmi data from City of Hope's Poseidon

- Poseidon: Precision Oncology Software Environment Interoperable Data Ontologies Network

```
[9]: # collect cohort of breast cancer patients using cancer staging body site data
```

```
#get cancerstaging body site codes for each patient
df_uuid_bmi = ks.sql("""
SELECT patient.uuid, MEAN(vitals.bmi)
FROM patient
LEFT JOIN encounter
ON patient.uuid=encounter.uuid
LEFT JOIN vitals
ON encounter.encounterid=vitals.encounterid
GROUP BY patient.uuid
LIMIT 100
""")

# convert to pandas
df_uuid_bmi=df_uuid_bmi.to_pandas()

# de-duplicate
df_uuid_bmi = df_uuid_bmi.drop_duplicates()

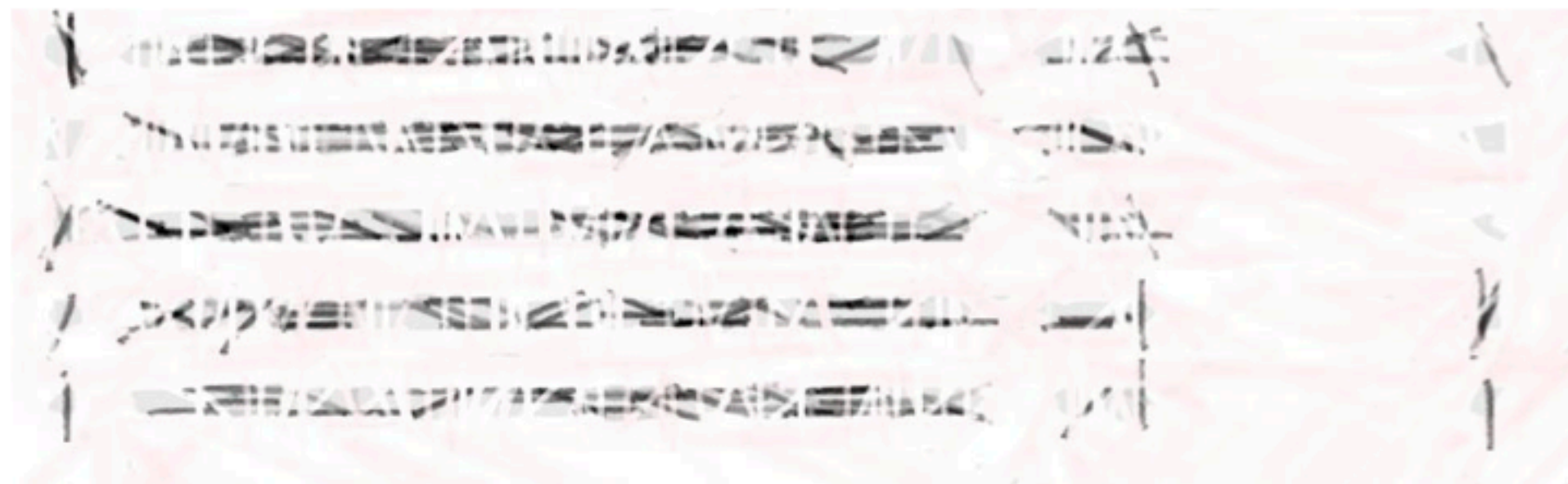
# Check
df_uuid_bmi.head()
```

```
[9]:
```

	uuid	avg(bmi)
0	74010710-00000000-00000000-00000000	17.44
1	74010710-00000000-00000000-00000000	NaN
2	74010710-00000000-00000000-00000000	NaN
3	74010710-00000000-00000000-00000000	NaN
4	74010710-00000000-00000000-00000000	NaN

Step 2 example: Attach alcoholusecode to cluster

uuid cluster alcoholusecode



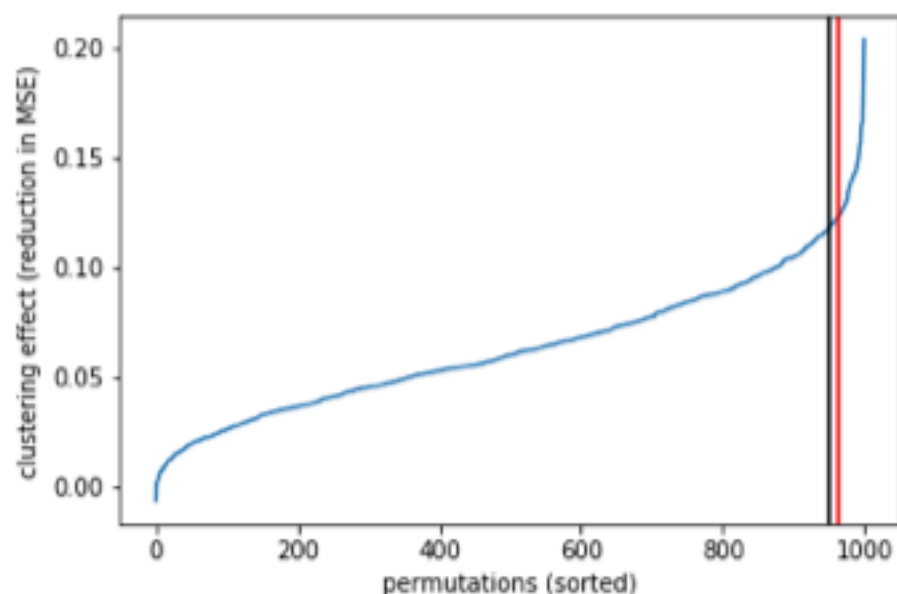
11111111-1111-1111-1111-111111111111	1	1
22222222-2222-2222-2222-222222222222	2	2
33333333-3333-3333-3333-333333333333	3	3
44444444-4444-4444-4444-444444444444	4	4
55555555-5555-5555-5555-555555555555	5	5

The 'uuid' refers to the patient's unique id. The 'cluster' is the genomic cluster. The alcoholusecode is a code which tells us how much alcohol the patient consumes per week. We would like to see if there is correlation between the genomic cluster and the alcoholusecode.

Step 3 example: Evaluate whether ageofonset correlates to cluster data

permutation p-value of the effect of clustering on improving prediction is: 0.036

[14]: 0.036



Some Initial results

$p > 0.05$

- Alcoholousecode (in socialhistory)
- Alcohol_oz_week (in socialhistory)
- Alcoholhowoftencode (in socialhistory)
- Tobaccopackperday (in socialhistory)
- Ethniccode (in patientethnic)
- Relationshipcode (in familyhistory)
- Tumorsizemm (in cancerstaging)
- Classificationcode (in cancerstaging)
- Histologycode (in registrydiagnosis)

$p < 0.05$

- Ageofonset (in familyhistory)

Promising patient variable: T-stage

- From the NIH website:
 - [Stage](#) refers to the extent of your cancer, such as how large the tumor is, and if it has spread.
 - The TNM system is the most widely used cancer [staging system](#).
 - The T refers to the size and extent of the main tumor. The main tumor is usually called the [primary tumor](#).
 - TX: Main tumor cannot be measured.
 - T0: Main tumor cannot be found.
 - T1, T2, T3, T4: Refers to the size and/or extent of the main tumor. The higher the number after the T, the larger the tumor or the more it has grown into nearby tissues. T's may be further divided to provide more detail, such as T3a and T3b.

T-stage + cluster

uuid tstage_q cluster

8	0x50f27b4c3e79334c79544c9f1078392	2.2	107.4
3	0x7294e892715a7443f10a4720701207	1.4	103.2
2	0x430779325570a4497702c4e3732929	2.2	93.0
4	0x7800545a3180534c7830c0e0795c4	2.4	70.2
4	0x07e234949981c907307394f4e07095	0.6	4.0

This is our most promising variable. We attached the tstage_q variable with the genomic cluster variable. Next, we will work with the tstage_q variable and the cluster variable to see if certain tstage-q's and clusters have low p-value.

Initial binning technique for t-stage:

```
: #Create bins zero_two,two_three,three_five,five_fivepointnine
```

```
bins = [0, 2, 3, 5, 5.9]
```

```
uuid_tstage_q_cluster['binned'] = pd.cut(uuid_tstage_q_cluster['tstage_q'], bins, labels=["zero_two", "two_three", "three_five", "five_fivepointnine"], right=False)
tstage_for_plot=uuid_tstage_q_cluster.copy()
tstage_for_plot=tstage_for_plot.sort_values('tstage_q')
```

```
uid_tstage_q_cluster=uid_tstage_q_cluster[['uid','cluster','binned']]
uid_tstage_q_cluster.head()
```

uid	cluster	bin
1	1	1
2	1	1
3	1	1
4	1	1
5	1	1
6	1	1
7	1	1
8	1	1
9	1	1
10	1	1
11	1	1
12	1	1
13	1	1
14	1	1
15	1	1
16	1	1
17	1	1
18	1	1
19	1	1
20	1	1
21	1	1
22	1	1
23	1	1
24	1	1
25	1	1
26	1	1
27	1	1
28	1	1
29	1	1
30	1	1
31	1	1
32	1	1
33	1	1
34	1	1
35	1	1
36	1	1
37	1	1
38	1	1
39	1	1
40	1	1
41	1	1
42	1	1
43	1	1
44	1	1
45	1	1
46	1	1
47	1	1
48	1	1
49	1	1
50	1	1
51	1	1
52	1	1
53	1	1
54	1	1
55	1	1
56	1	1
57	1	1
58	1	1
59	1	1
60	1	1
61	1	1
62	1	1
63	1	1
64	1	1
65	1	1
66	1	1
67	1	1
68	1	1
69	1	1
70	1	1
71	1	1
72	1	1
73	1	1
74	1	1
75	1	1
76	1	1
77	1	1
78	1	1
79	1	1
80	1	1
81	1	1
82	1	1
83	1	1
84	1	1
85	1	1
86	1	1
87	1	1
88	1	1
89	1	1
90	1	1
91	1	1
92	1	1
93	1	1
94	1	1
95	1	1
96	1	1
97	1	1
98	1	1
99	1	1
100	1	1
101	1	1
102	1	1
103	1	1
104	1	1
105	1	1
106	1	1
107	1	1
108	1	1
109	1	1
110	1	1
111	1	1
112	1	1
113	1	1
114	1	1
115	1	1
116	1	1
117	1	1
118	1	1
119	1	1
120	1	1
121	1	1
122	1	1
123	1	1
124	1	1
125	1	1
126	1	1
127	1	1
128	1	1
129	1	1
130	1	1
131	1	1
132	1	1
133	1	1
134	1	1
135	1	1
136	1	1
137	1	1
138	1	1
139	1	1
140	1	1
141	1	1
142	1	1
143	1	1
144	1	1
145	1	1
146	1	1
147	1	1
148	1	1
149	1	1
150	1	1
151	1	1
152	1	1
153	1	1
154	1	1
155	1	1
156	1	1
157	1	1
158	1	1
159	1	1
160	1	1
161	1	1
162	1	1
163	1	1
164	1	1
165	1	1
166	1	1
167	1	1
168	1	1
169	1	1
170	1	1
171	1	1
172	1	1
173	1	1
174	1	1

1. The first step in the process is to identify the problem or issue that needs to be addressed. This involves gathering information and understanding the context of the problem.

2. Once the problem is identified, the next step is to define the objectives and goals of the project. This helps to clarify what needs to be achieved and provides a clear direction for the team.

3. The third step is to develop a plan or strategy to address the problem. This involves breaking down the problem into smaller, manageable tasks and determining the resources needed to complete each task.

4. The fourth step is to implement the plan. This involves putting the strategy into action and monitoring progress regularly to ensure that the project is on track.

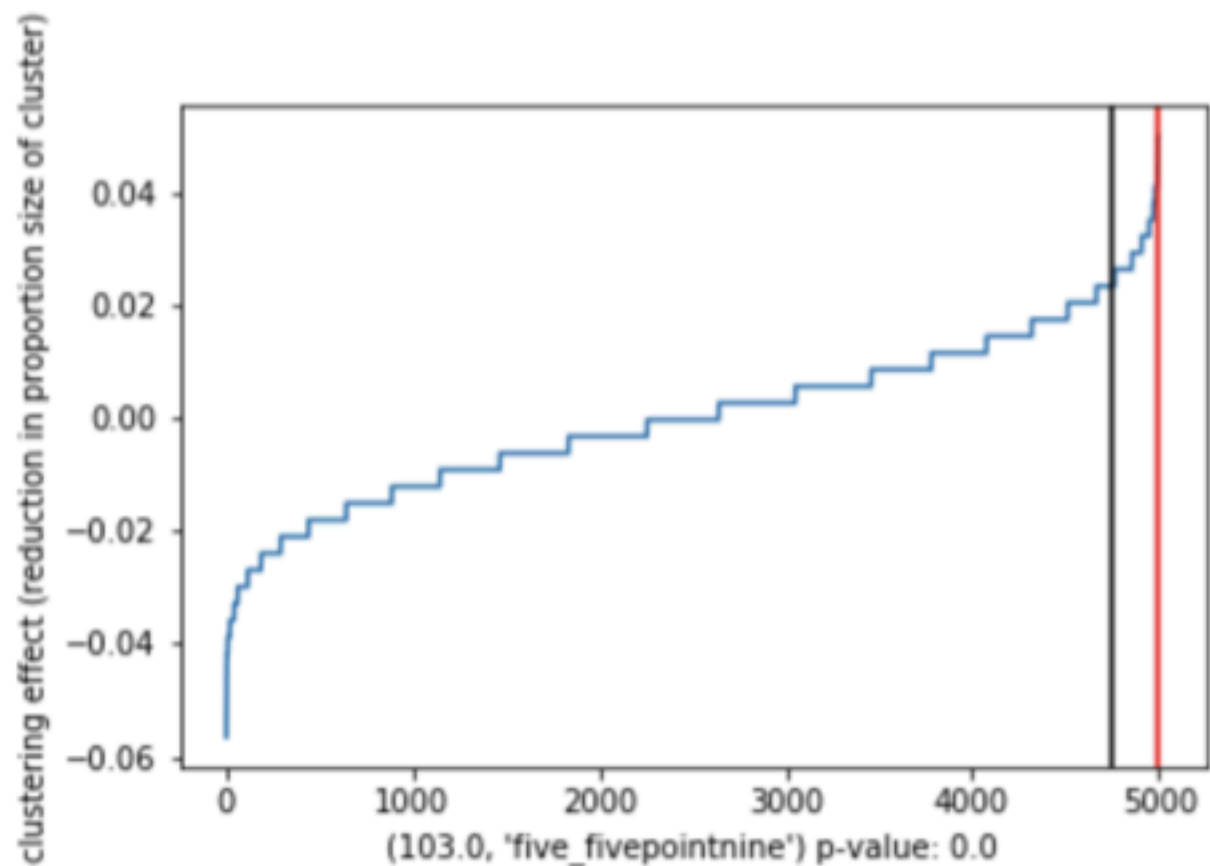
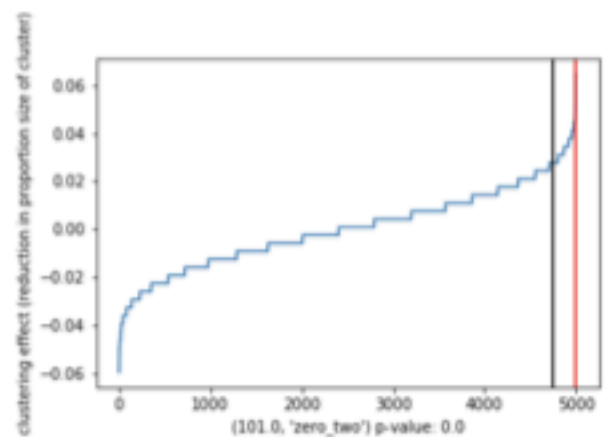
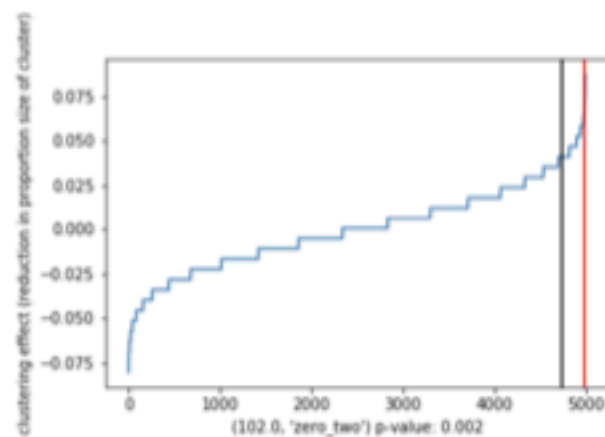
5. The final step is to evaluate the results of the project. This involves assessing the outcomes against the objectives and goals and identifying any areas for improvement or further action.

Value counts for some of the cluster- bin combos:

cluster	binned	
1.0	two_three	28
	zero_two	9
	three_five	8
	five_fivepointnine	7
3.0	two_three	9
	three_five	3
	five_fivepointnine	2
	zero_two	1
5.0	two_three	12
	three_five	6
	zero_two	4
	five_fivepointnine	4
6.0	two_three	13
	five_fivepointnine	3
	three_five	1
100.0	two_three	17
	three_five	7
	zero_two	2
	five_fivepointnine	1
101.0	two_three	191
	zero_two	43
	three_five	35
	five_fivepointnine	29
102.0	two_three	55
	zero_two	11
	three_five	7
	five_fivepointnine	5
103.0	two_three	192
	three_five	59
	zero_two	49
	five_fivepointnine	38
104.0	two_three	24
	five_fivepointnine	15
	three_five	5
	zero_two	1
105.0	two_three	188
	three_five	32
	zero_two	29
	five_fivepointnine	21
106.0	two_three	19
	three_five	6
	five_fivepointnine	3
	zero_two	2
107.0	two_three	8
	zero_two	2

P-values of
combinations of
cluster-bin:

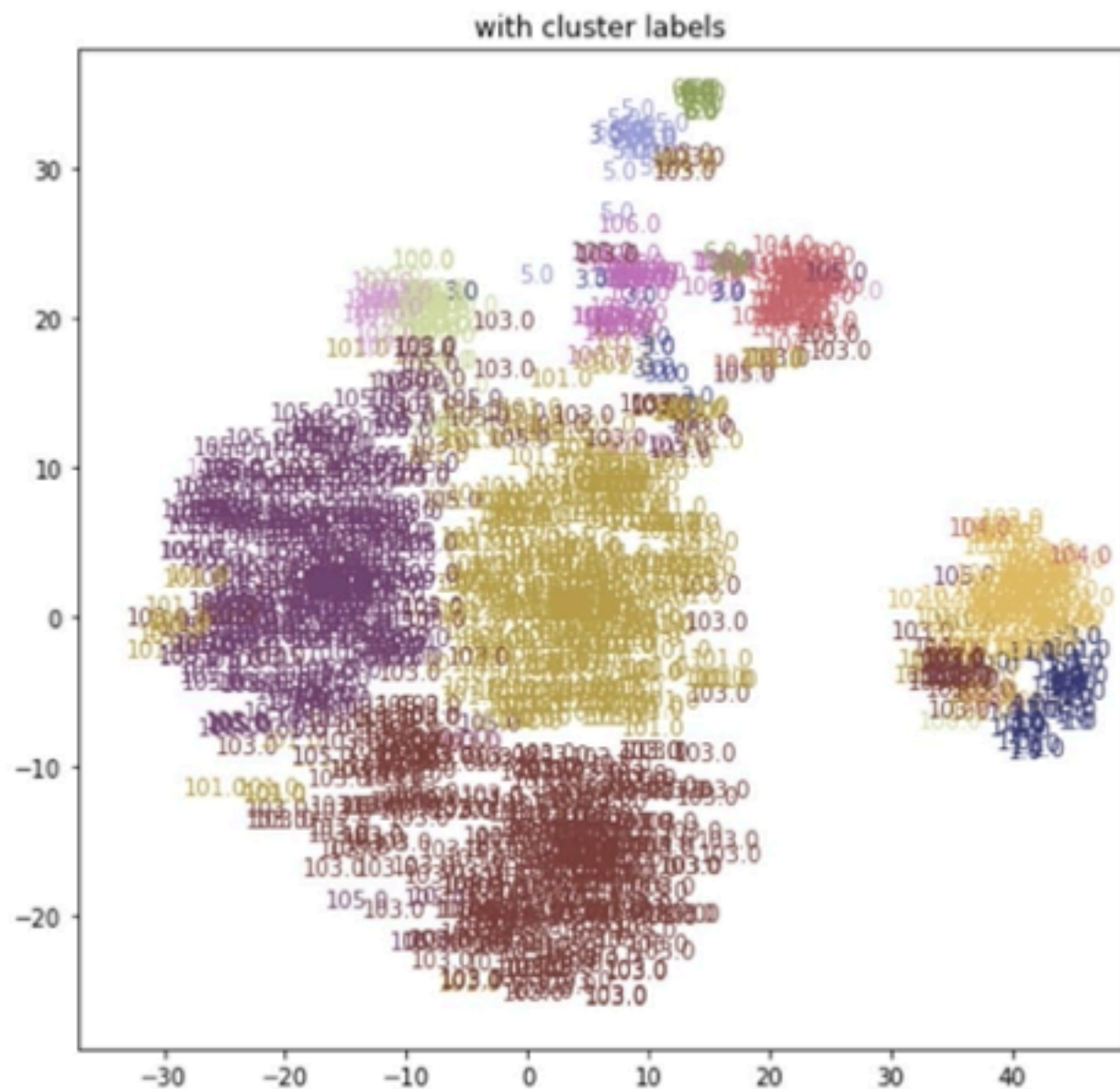
0	0.982
1	0.002
2	0.4052
3	0.016
4	0.2868
5	0.4792
6	0.2412
7	0.3812
8	0.4988
9	0.0
10	0.8772
11	0.0996
12	0.68
13	0.734
14	0.0
15	0.3568
16	0.5344
17	0.42
18	0.1608



P-values:

as using t-SNE with perplexity: 30 and seed: 2

t-sne
visualizations
of the
clusters



We duplicated each of the rows 10x in order to get less sparse value counts in our cluster-bin combos.

New p-values:

0 0.0
1 0.0
2 0.3323
3 0.0
4 0.0
5 0.0
6 0.0
7 0.1645
8 0.0
9 0.9246
10 0.2161
11 0.0
12 0.0463
13 0.0002
14 0.0051
15 0.195
16 0.1984
17 0.0135
18 0.4113
19 0.0496
20 0.0078
21 0.0504
22 0.0009

23 0.0
24 0.0241
25 0.0296
26 0.0
27 0.1848
28 0.0441
29 0.0017
30 0.1905
31 0.0
32 0.166
33 0.0242
34 0.6528