Word2vec &
Generative adversarial networks
on sales data

NUS CBE
2018.07.18

Word2vec - Recap

- Input: only take a sequence of data, no labels

	basketId	productId
1	11	['38057', '37134', '42403', '38800']
2	12	['37677', '54220', '54233', '46086', '54226', '54239', '35624', '46140']
3	16	['36520', '36529']
4	17	['46957', '47863']
5	20	['54804', '54686']
6	31	['43420', '43256']
7	40	['37783', '42722']
8	44	['22448', '4177', '22436', '8552']

Word2vec

- Output: a vector of real numbers for each productId

	productId	productName	1	2	3	4	5	6	7	8	9	10
1	2544	NEW ALOE ANTI-PERSPIRANT DEODORANT 50ML	-1.46	1.941	4.57	0.33	-2.37	1.91	3.702	-0.15	-5.01	-2.66
2	2617	NEW VIT.E MOIST CREAM 50ML	1.92	1.497	-0.15	1.70	4.46	-2.30	-5.891	0.55	-1.55	-0.82
3	4887	ALL IN ONE FACE BASE 03	1.49	0.932	0.89	-2.56	1.98	-1.45	-2.905	1.65	2.21	-1.74
4	4896	ALL IN ONE FACE BASE 04	0.38	0.594	0.24	-2.08	1.92	-0.73	-2.275	1.07	1.71	-2.38
5	6040	MMF LIP & CHEEK STAIN 01 PINK ROSE	0.65	-2.355	-1.48	-1.98	-0.97	0.68	-0.096	0.99	1.32	-2.24
6	7996	WHITE MUSK SMOOTH SATIN BODY LOTION 250ML	-3.33	2.493	2.70	-1.17	-2.54	4.93	0.065	2.10	1.72	3.51
7	32768	NEW VIT C F/ CLEAN POLISH 100ML	-1.91	-1.988	5.80	1.08	6.34	1.75	-1.452	-1.88	-0.32	-1.87
8	39357	MORINGA BODY MIST 100ML	-3.03	0.643	-1.34	-0.50	-2.99	1.58	-1.866	1.79	-0.21	0.53
9	39457	VANILLA BODY MIST 100ML	0.68	0.077	-3.13	-2.06	-3.86	1.27	-1.995	0.18	0.38	1.08
10	65160	STAMPCARD 2015	-1.99	-1.793	-2.89	-1.96	-0.98	0.92	1.360	2.08	-0.42	0.30

Count-based

- GloVe: Global Vectors for Word Representation (Pennington et al. 2014)
- Firstly, get the co-occurrence matrix (numbers are for demonstration)

	11788	11814	11823	11849	11860	11870	11907	11921
11788	0	6	0	1	1	0	4	2
11814	6	0	1	0	1	0	3	4
11823	0	1	0	0	0	0	0	0
11849	1	0	0	0	0	0	0	0
11860	1	1	0	0	0	0	0	0
11870	0	0	0	0	0	0	0	1
11907	4	3	0	0	0	0	0	4
11921	2	4	0	0	0	1	4	0

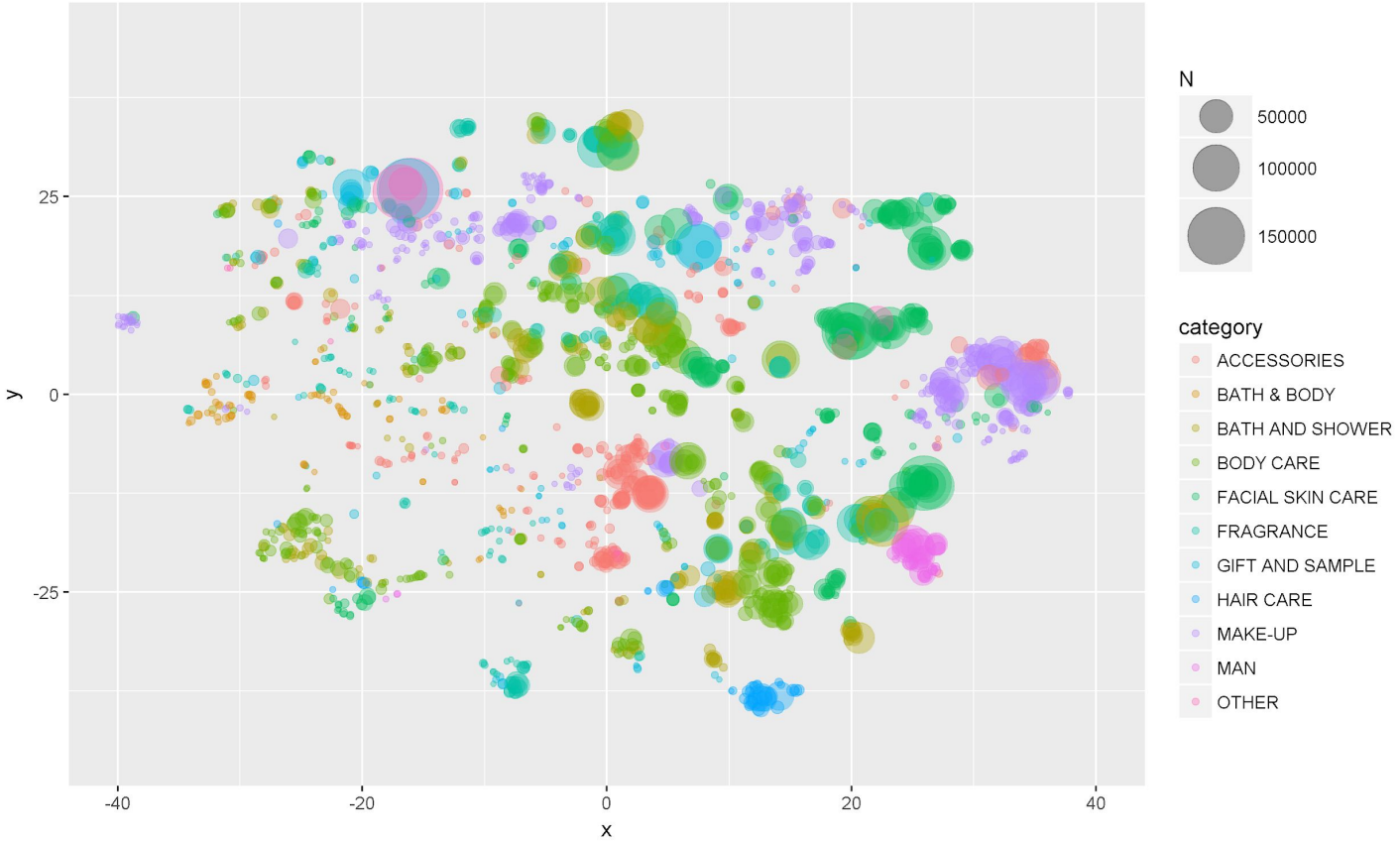
Weighted Least Squares

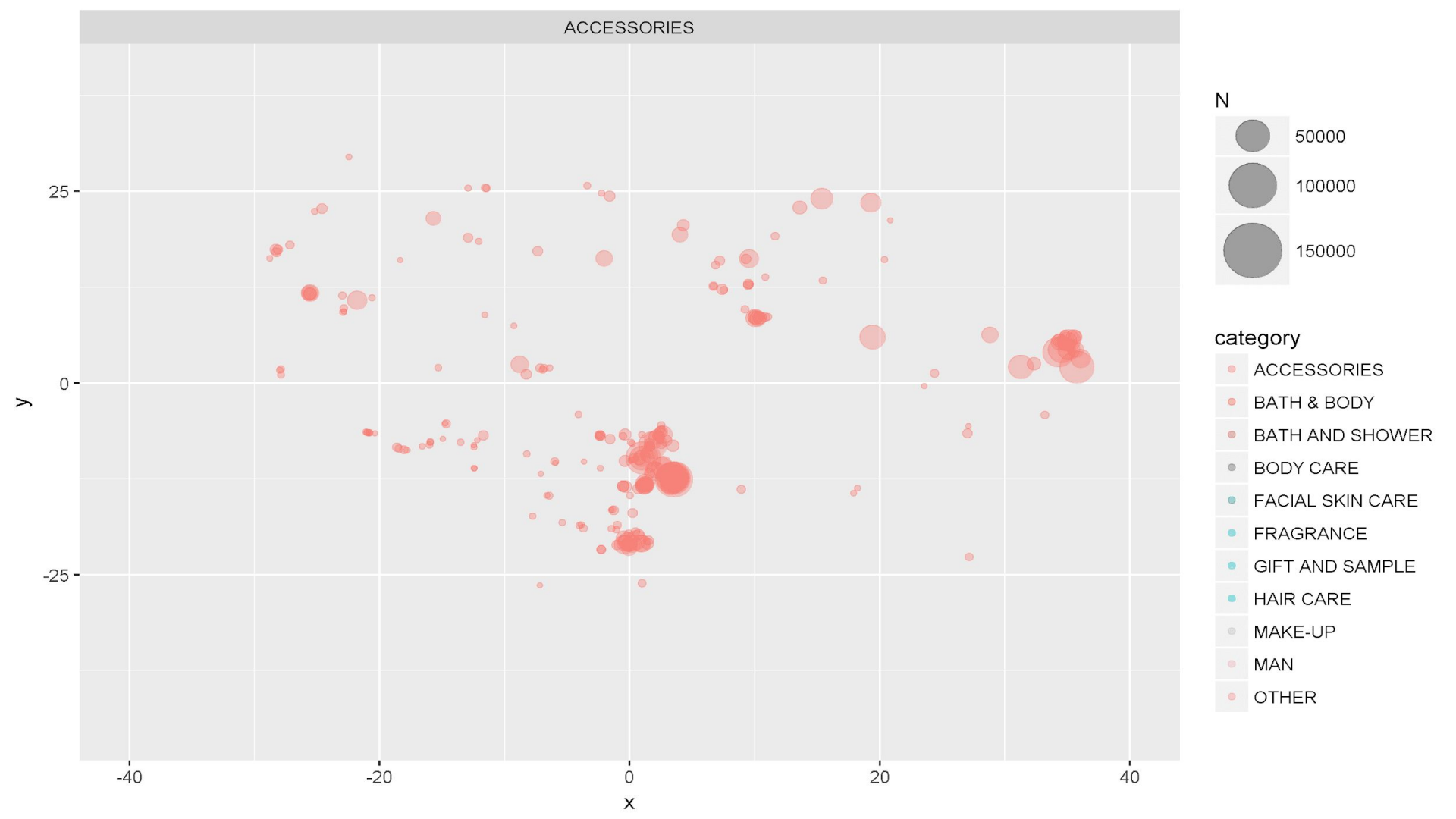
$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f(X_{ij}) \left(\vec{w}_i^T \vec{w}_j + b_i + b_j - \log X_{ij} \right)^2$$

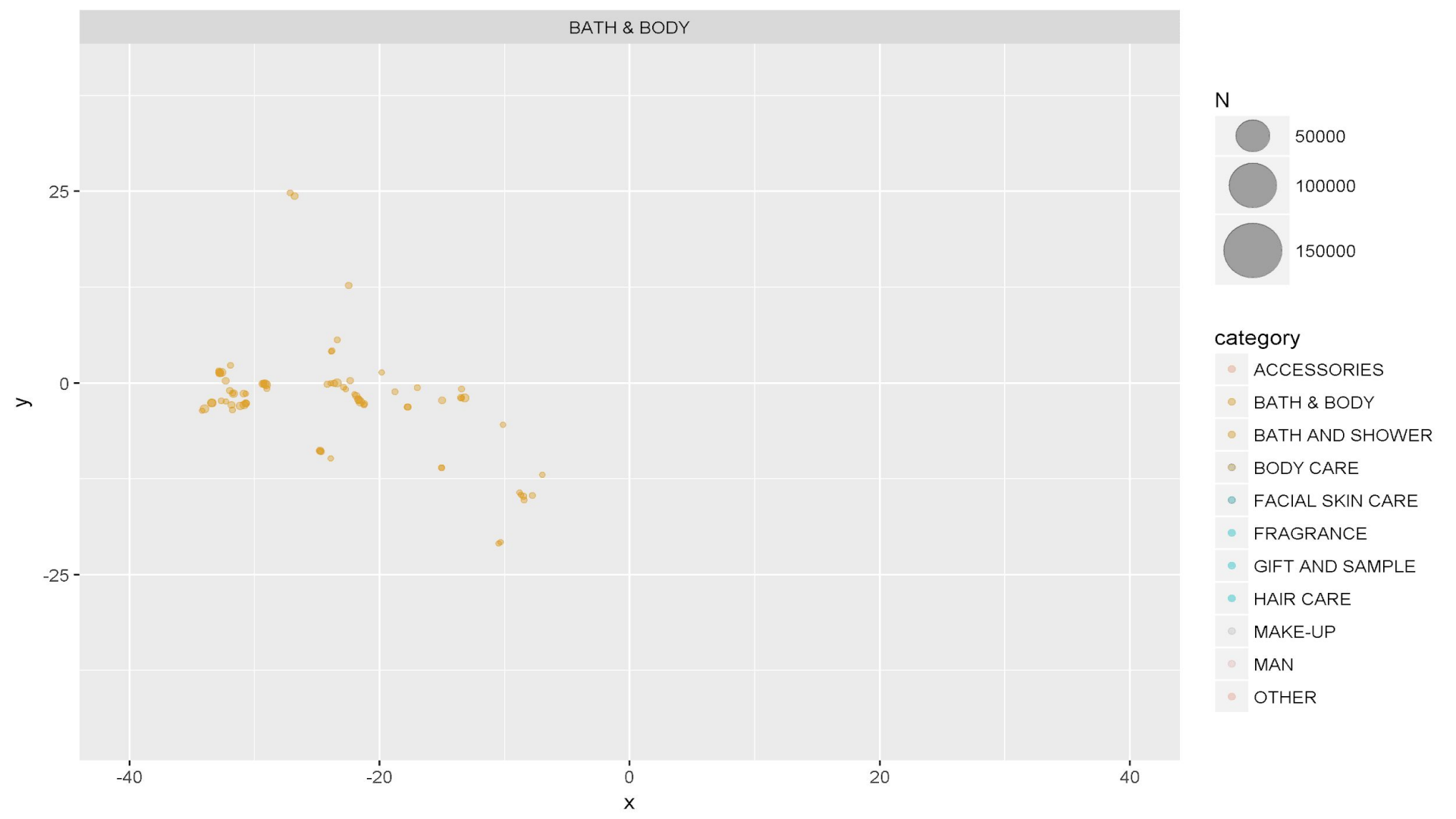
$$f(X_{ij}) = \begin{cases} \left(\frac{X_{ij}}{x_{\max}} \right)^{\alpha} & \text{if } X_{ij} < x_{\max} \\ 1 & \text{otherwise.} \end{cases}$$

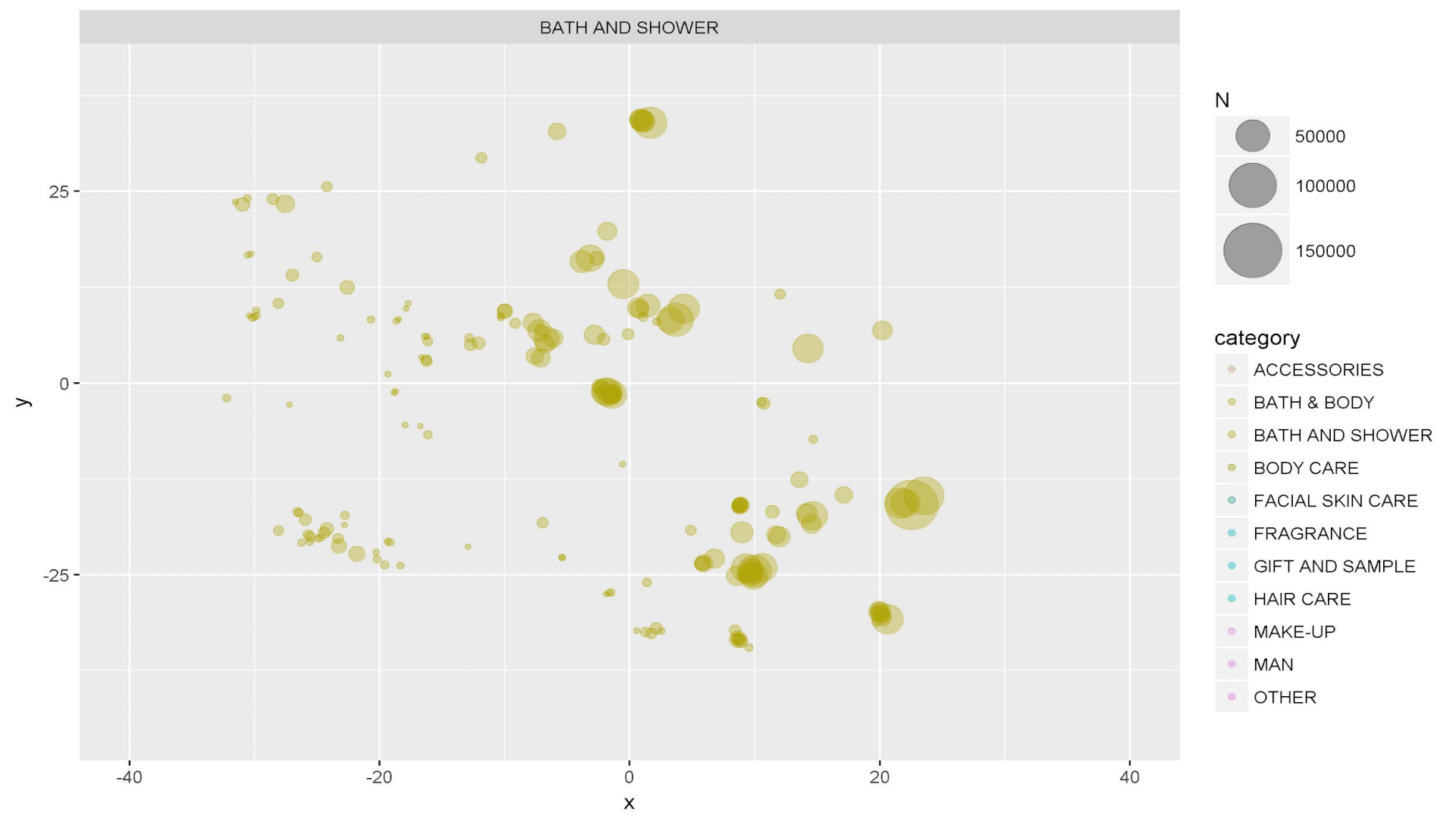
- f(X) helps prevents common word pairs from skewing our objective. For extremely common word pairs (where Xij> xmax) this function simply returns 1. For all other word pairs, we return some weight in the range (0,1)
- 2k words * 150 = 300k parameters to optimize. Often by gradient descent methods.

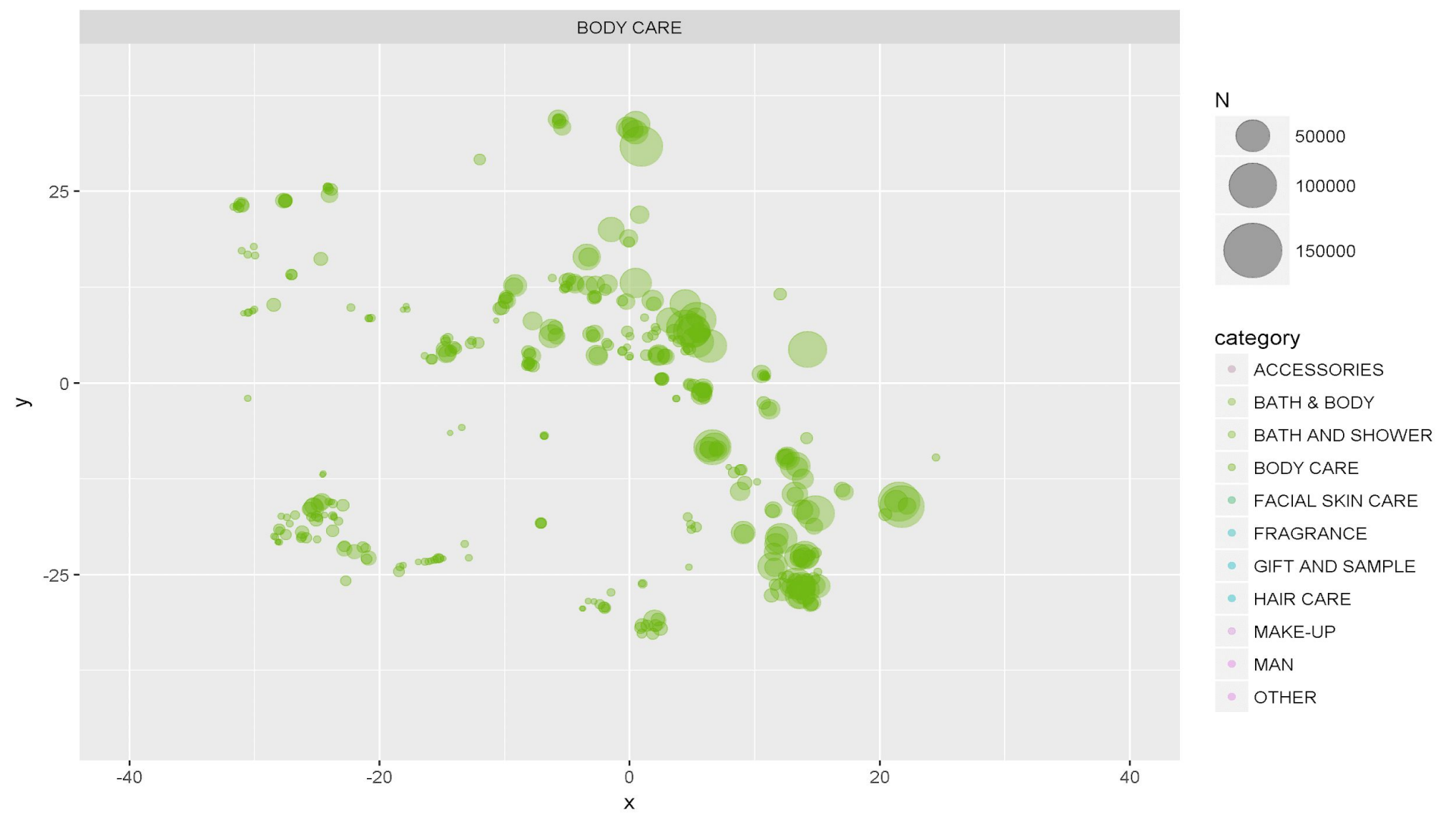
Per = 25, Dim = 150, Min = 1000, Max = 30000

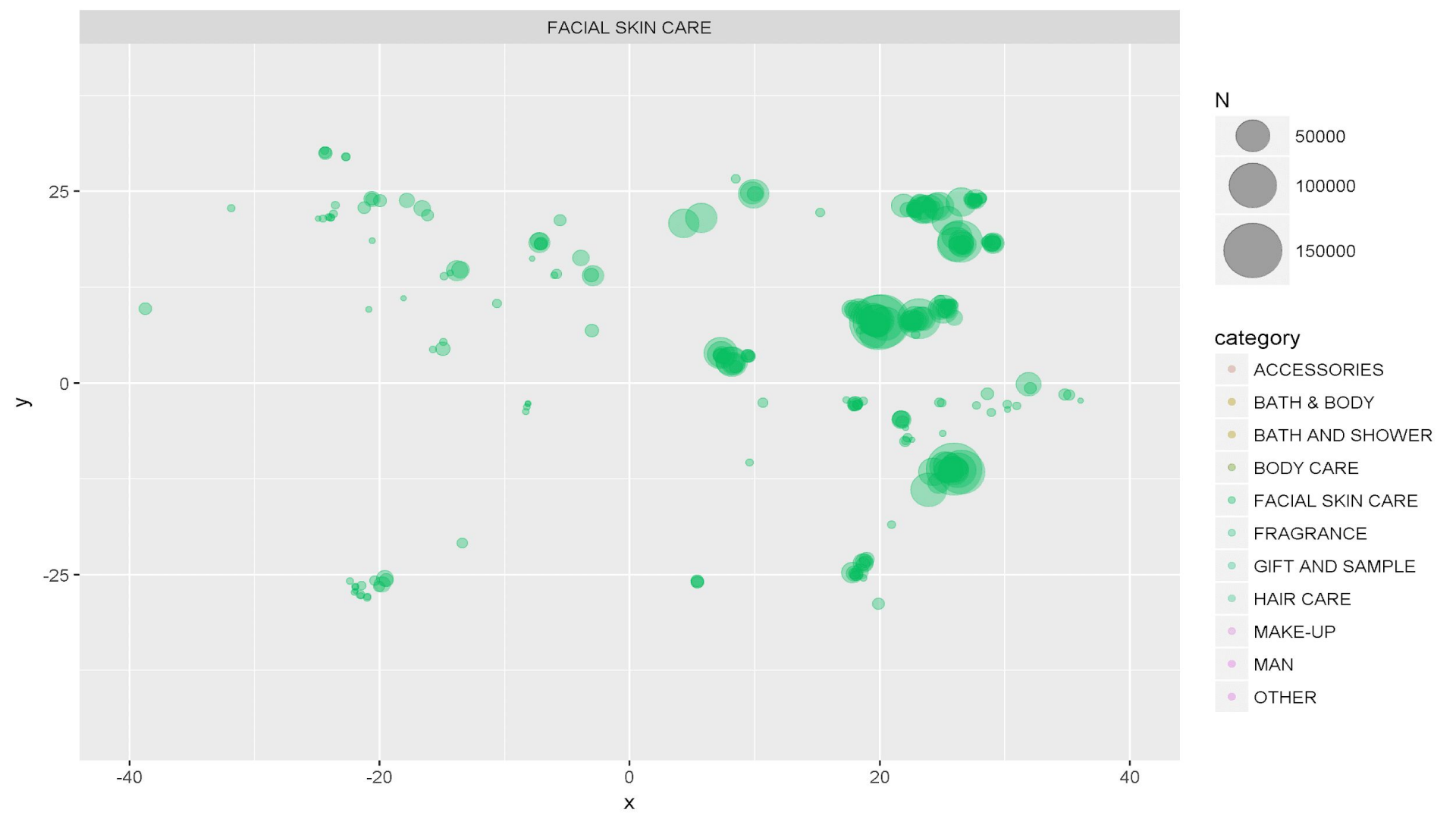


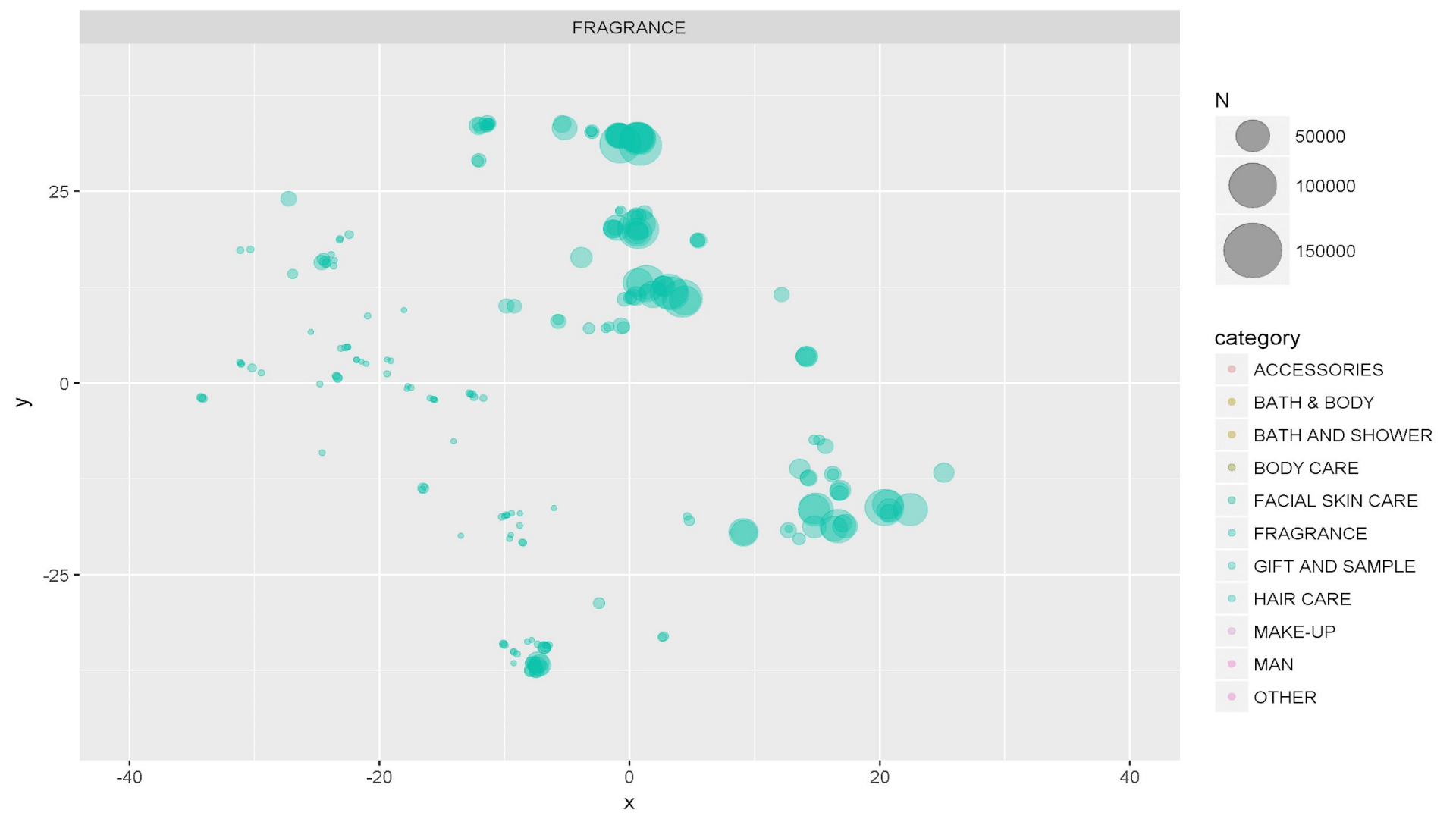

ACCESSORIES
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER

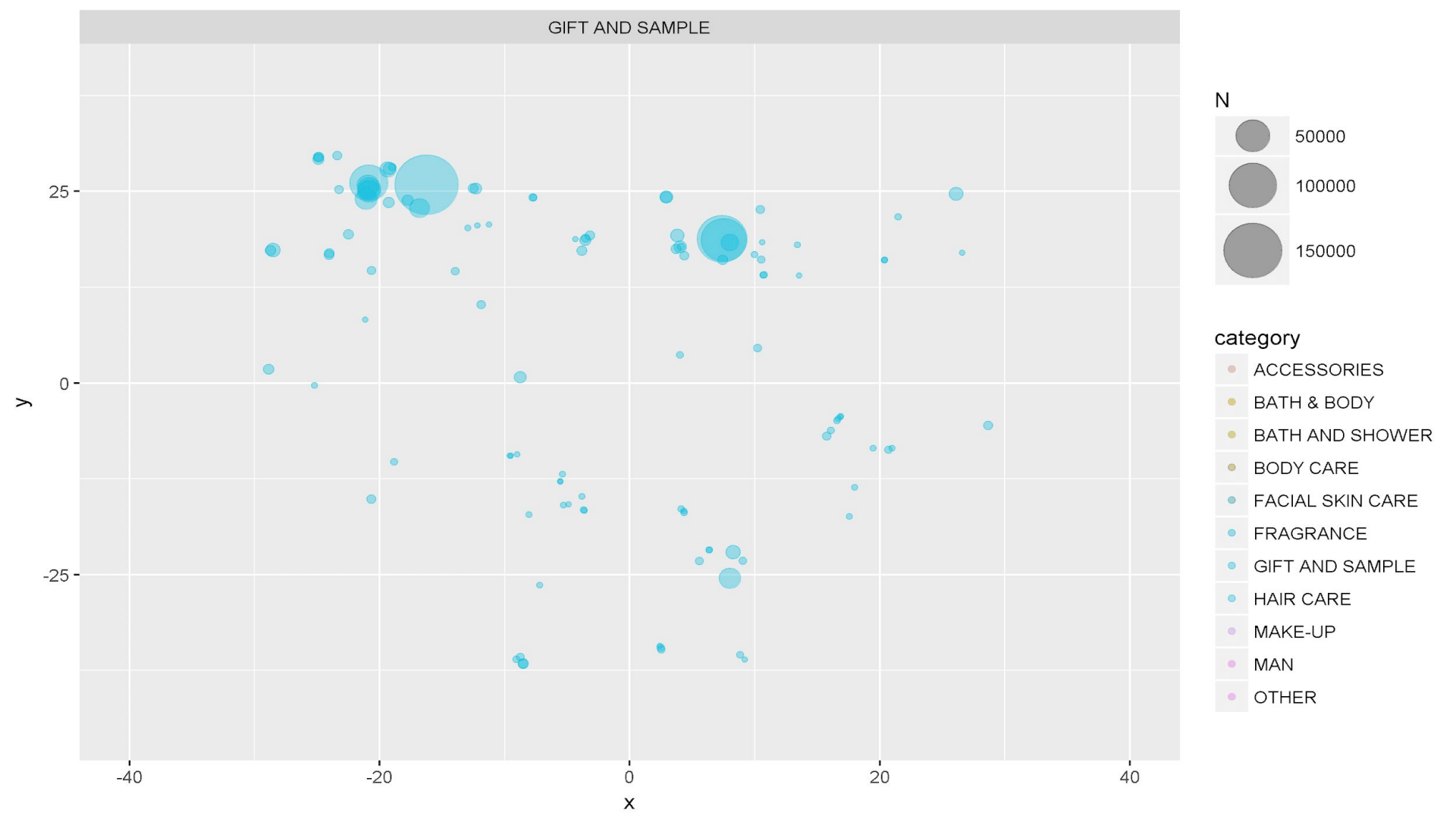

BATH & BODY
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER

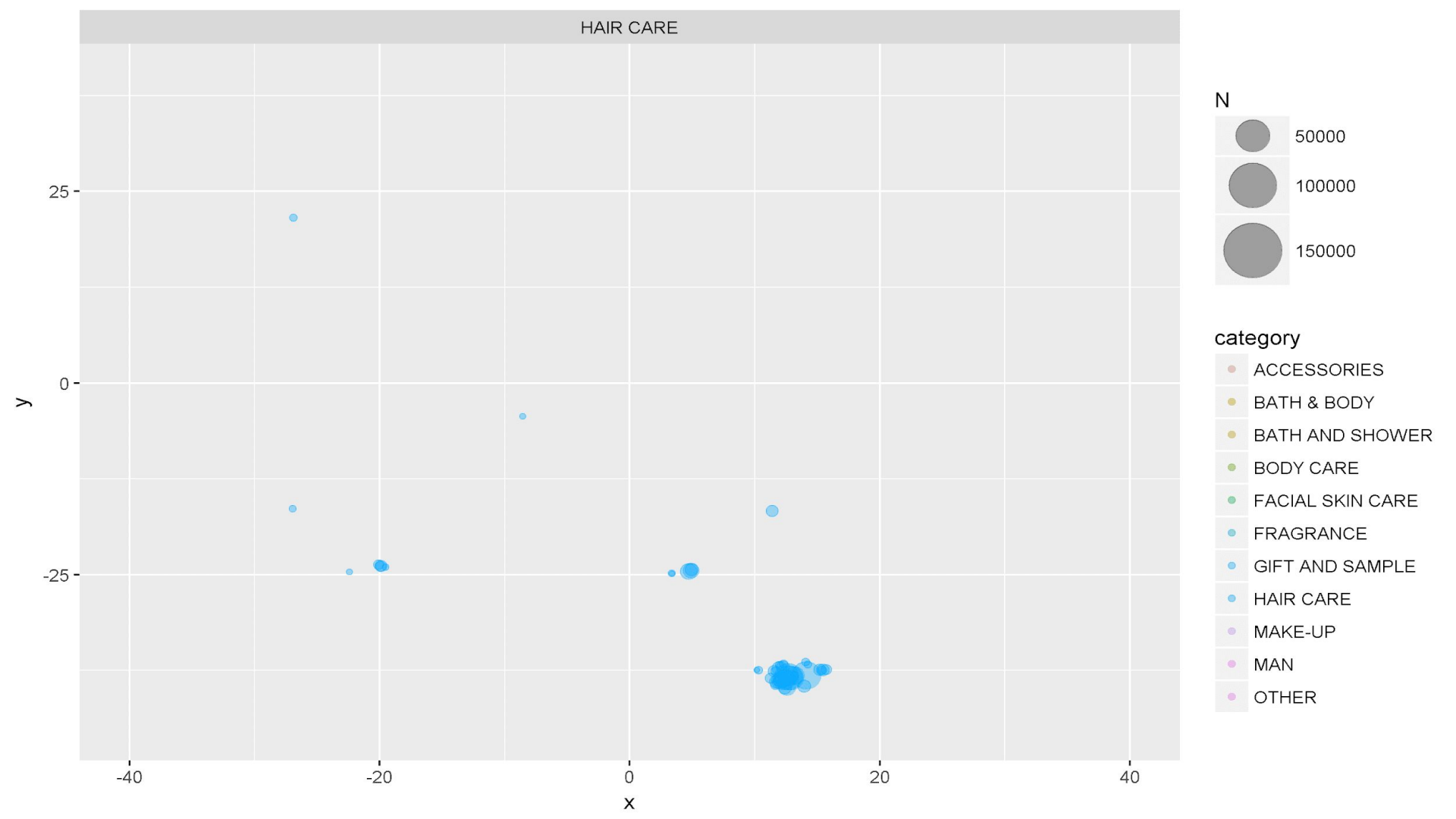

BATH AND SHOWER
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER

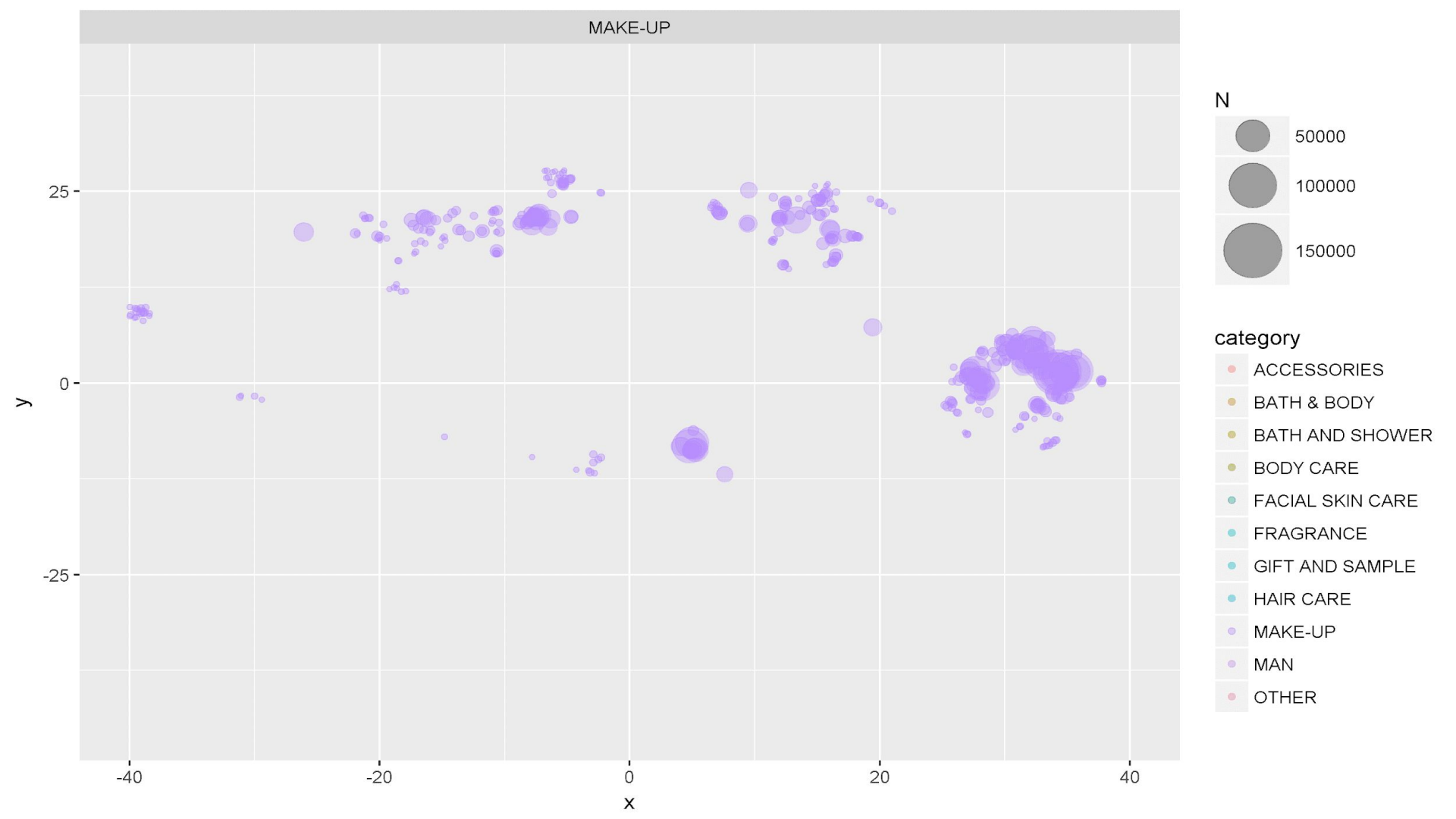

BODY CARE
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER

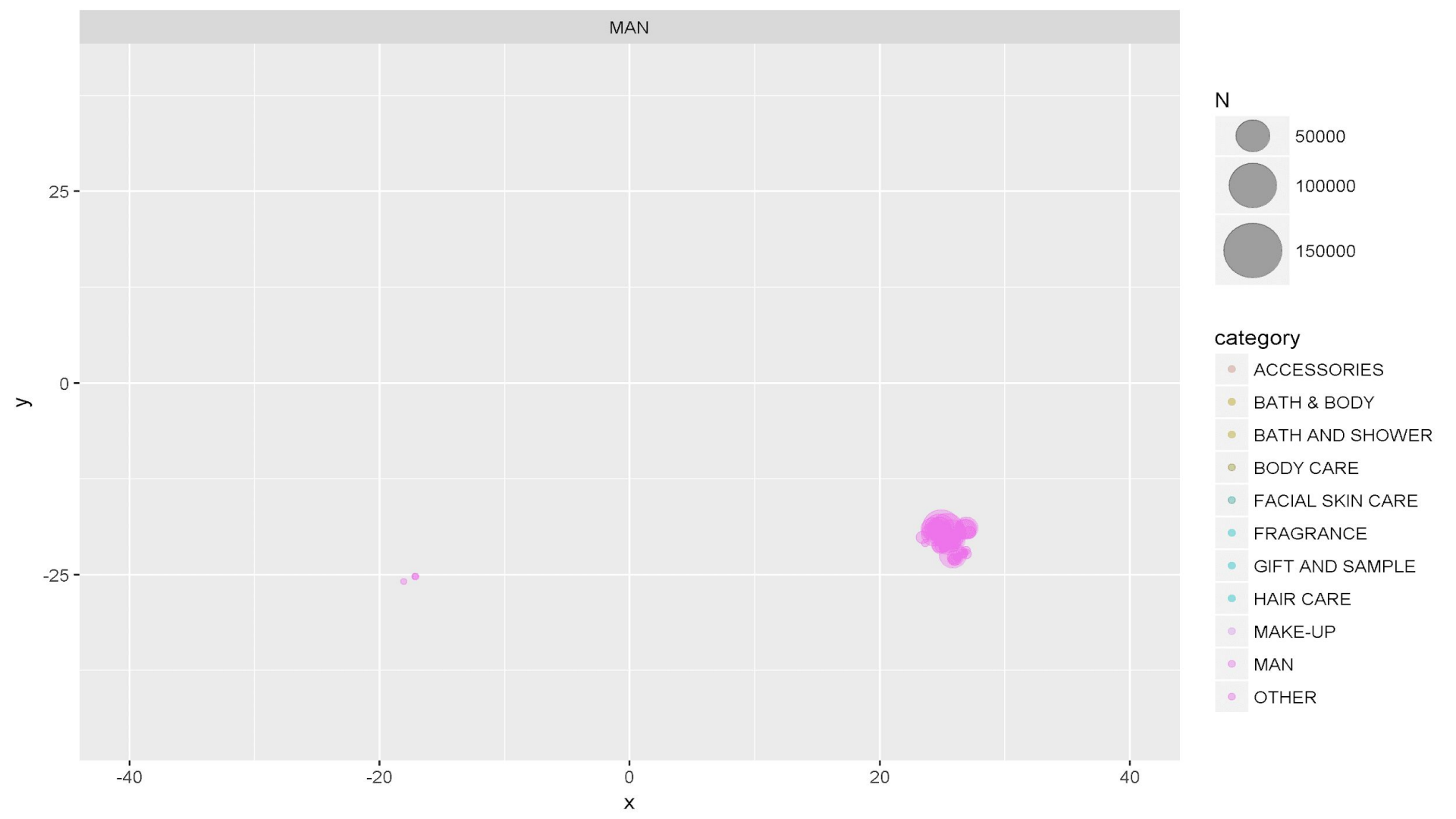

FACIAL SKIN CARE
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER

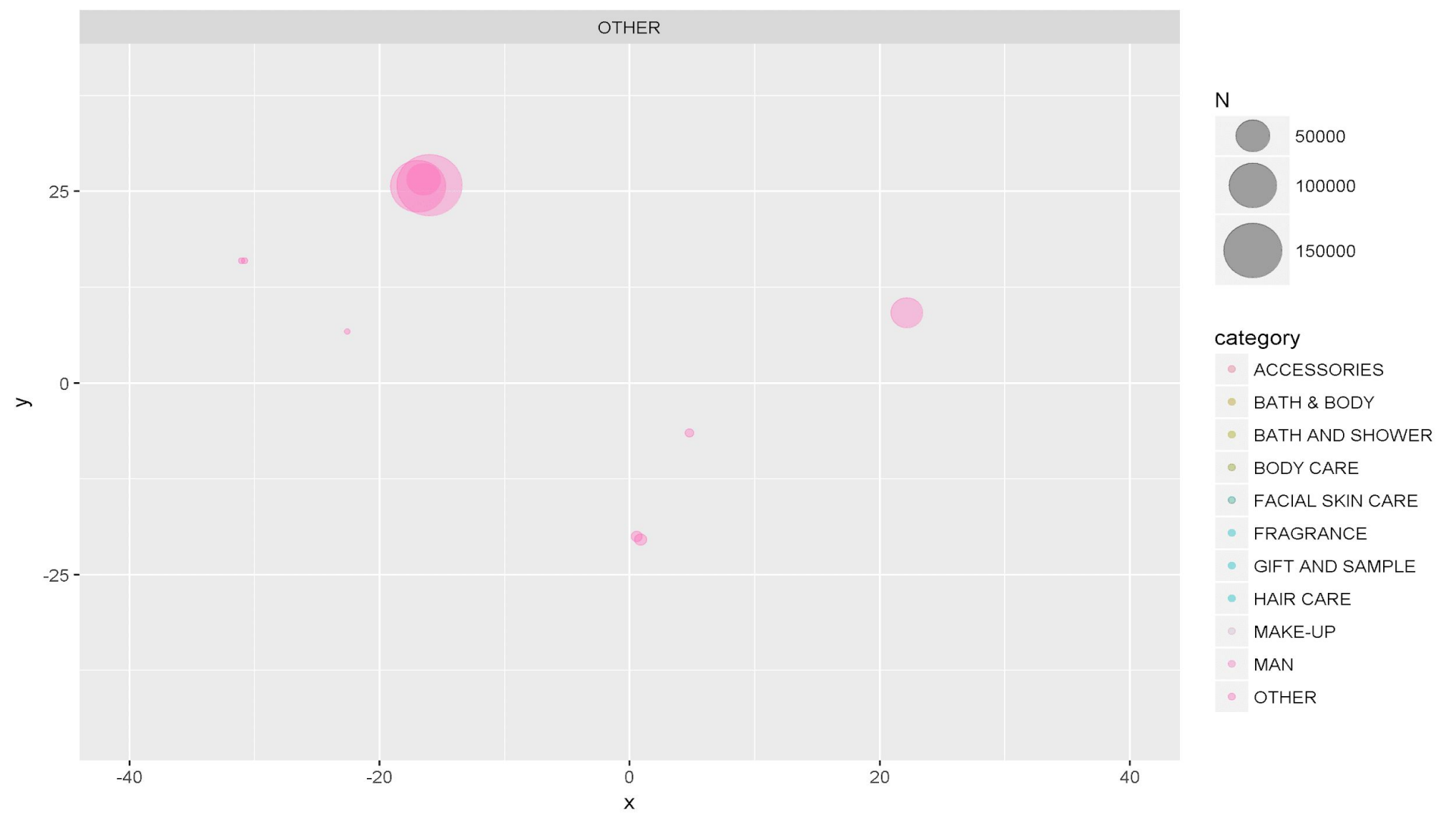

FRAGRANCE
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER


GIFT AND SAMPLE
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER


HAIR CARE
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER


MAKE-UP
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER

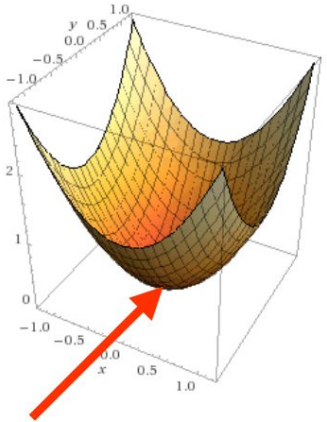

MAN
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER


OTHER
25
0
-25
y
-40
-20
0
20
40
x
N
50000
100000
150000
category
ACCESSORIES
BATH & BODY
BATH AND SHOWER
BODY CARE
FACIAL SKIN CARE
FRAGRANCE
GIFT AND SAMPLE
HAIR CARE
MAKE-UP
MAN
OTHER

Generative adversarial networks

- Goodfellow et al. 2014
- A generative model G that learn to estimate the data distribution
- A discriminative model D that learns to determine whether a sample came from the training data or G.
- Analogy:
  - G: a team of counterfeiters, trying to produce fake currency
  - D: the police, trying to detect the counterfeit currency
  - Competition drives both teams to improve their methods until the counterfeits are indistinguishable from the genuine articles.
- This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D gives 0.5 probability everywhere.

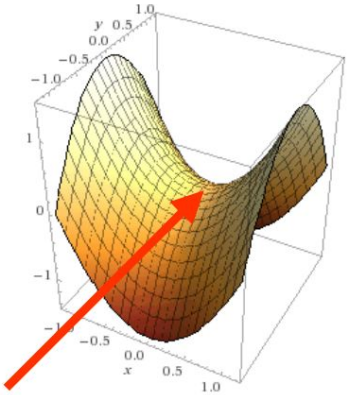
Traditional ML:
optimization



Minimum

One player,
one cost

Adversarial ML:
game theory



Equilibrium

More than one player,
more than one cost

(Goodfellow 2018)

Generative Modeling: Sample Generation

Training Data
(CelebA)

Sample Generator
(Karras et al, 2017)

Unsupervised image-to-image translation

Day to night

(Liu et al., 2017)

CycleGAN

(Zhu et al., 2017)

Experiment on sales data

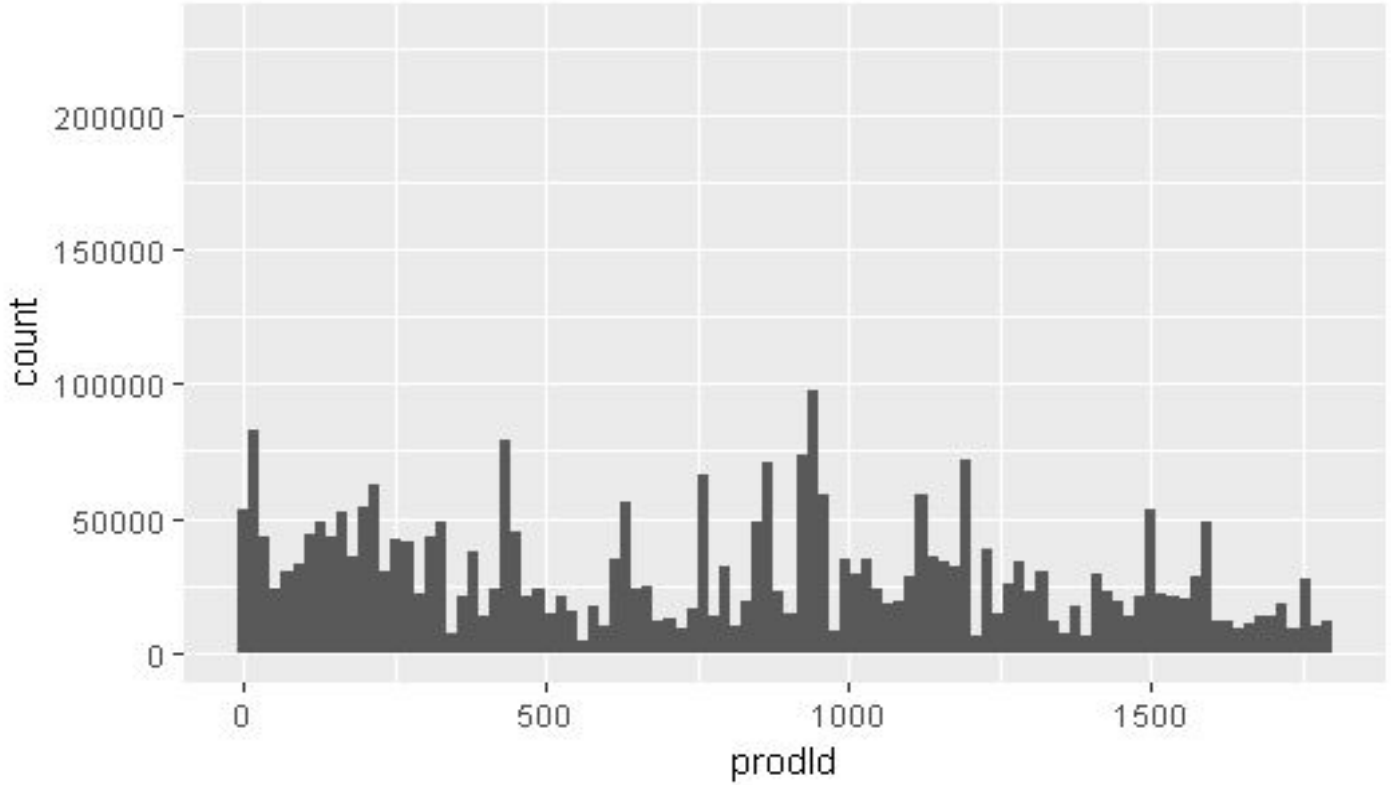
- Data: 3m samples, each sample is a product and its date & time of purchase. Each product is represented by its product vector (from word2vec) instead of by a dummy vector which does not reflect relations between products.
- Goal: use GAN to generate synthetic data similar to these samples.

	V1	V2	V3	V4	V5	V150	date	time
1	0.0172419250011444	-0.0821054577827454	0.28655868768692	-0.0180888921022415	0.211732909083366	-0.0341860800981522	950	12
2	-0.0844327211380005	-0.0163798620924354	-0.220107309520245	0.0743319839239121	-0.42529807984829	-0.559539020061493	1643	17.7
3	-0.32538054138422	0.444591775536537	0.600041389465332	0.288769759237766	-0.447193935513496	-0.75873613357544	1609	19.1
4	-0.0527503900229931	-0.21038556843996	0.289005145430565	0.0928692519664764	0.112089006230235	0.0402777940034866	1040	15.7
5	0.134681582450867	0.503233224153519	0.176852717995644	0.373667806386948	-0.0321113988757134	0.177543625235558	1148	10.5
6	0.0342223942279816	0.708288043737412	0.168218985199928	-0.977737128734589	-0.100249543786049	0.453723162412643	1758	15.2
7	-0.253069676458836	0.313125997781754	0.714814990758896	0.341831728816032	0.283551543951035	0.125047728419304	1043	11.2
8	-0.375039651989937	0.0767158195376396	-0.384532943367958	0.206331215798855	0.00166600197553635	-0.181033924221993	1009	18.9
9	0.308141499757767	-0.107846033759415	0.153598342090845	-0.136195577681065	-0.426411390304565	-1.07701203227043	1318	15.1
10	-0.609996274113655	0.473380725830793	0.521254189312458	0.112563403788954	0.441251337528229	0.0440352521836758	807	13

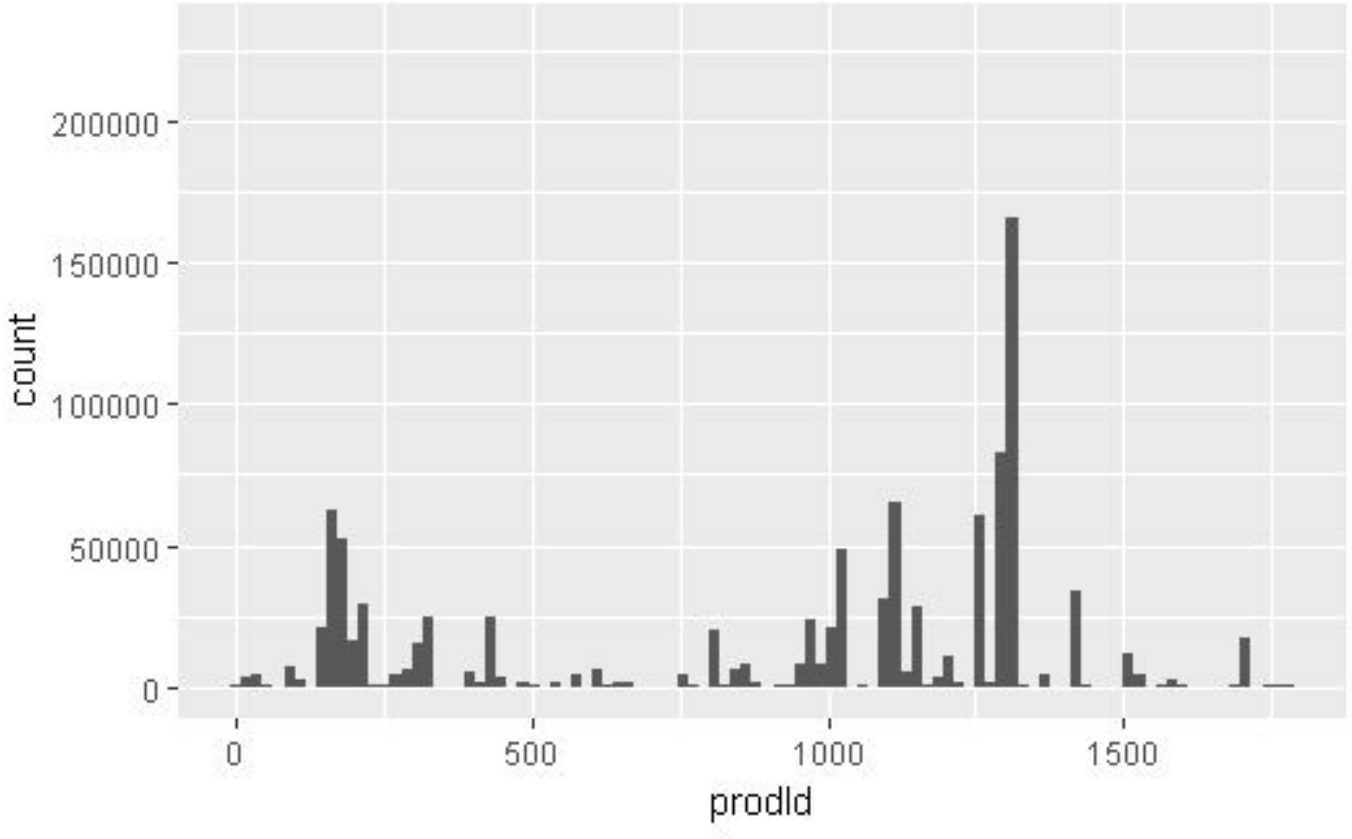
Experiment on sales data

- Alternate between 10 steps of optimizing D and 1 step of optimizing G.
- Traditional optimization: loss(MSE …) goes down over time
- But for GAN - a zero sum game: fluctuating, difficult to know when to stop the training process
- For images, people manually check the generated samples quality.
- For this task, I check the distribution of samples at different stages of the training process.
- Convert those product vectors to nearest products to get prodId

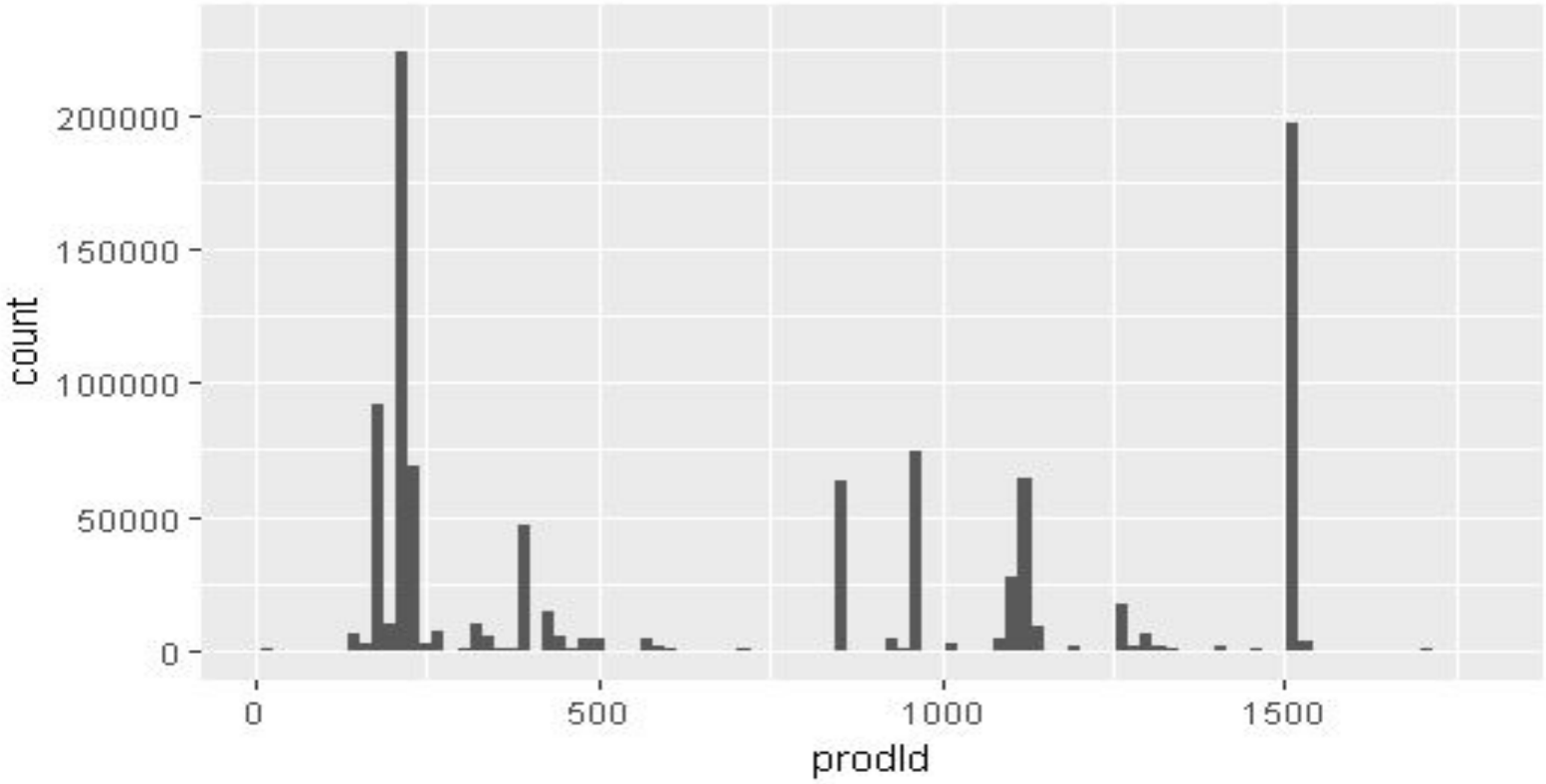
Distribution of 3m real samples



Distribution of 1m synthetic samples - epoch 230



Distribution of 1m synthetic samples - epoch 290



Notes on implementation

- GANs are known to be hard to train
- Adapted the code from Generating Multi-Categorical Samples with Generative Adversarial Networks (Camino et al. 2018)
- https://github.com/rcamino/multi-categorical-gans
- A variant of GAN: WGAN with gradient penalty
- Python
- Pytorch package (by Facebook)
- Ubuntu (Linux)

Summary

- Word2vec for detecting clusters in data, producing product vectors for other tasks
- GAN for generating synthetic data.
- Moving forward: GAN for counterfactual, causality?

References

http://www.iangoodfellow.com/slides/2018-05-24.pdf

https://arxiv.org/abs/1701.00160 - NIPS 2016 Tutorial by Goodfellow