Targetowanie reklam z machine learning

Czyli słowa kluczowe i trochę Adtechu

Spis treści

Uwaga - dużo nerdbełkotu!

Ten nudny slajd ze mną

1. Jestem programistą Pythona
2. Należę do Hackerspace Silesia
3. W pracy zajmuję się Adtech-em
4. Zajmuję się zliczaniem wejść i zakupów na stronę oraz raportowaniem

Geneza problemu

- Klient zajmuje się reklamą wielu sklepów
- Klient bardzo chciał machine learning
- Jedynie jakie mamy dane to z logi tracker-ów
- Dane muszą zostać przerobione do postaci linia - słowa kluczowe

Analiza danych

- Jakie to są dane?
- Jak one wyglądają?
- (dokąd zmierzamy?)
- Czy wszystkie słowa są nam potrzebne?

Parsowanie danych

1. Mamy tylko logi z NGINX
2. Interesuje nas tylko adres (bez host-a)
3. Pozbywamy się stron typu admin / login
4. Pozbywamy się liczb (nic nie mówiące ID)
5. Pozbywamy się niepotrzebnych słów oraz znaków specjalnych
6. Słowa kluczowe grupujemy po IP/cookieID
7. Dokument = słowa kluczowe per IP/cookieID

Uczenie Maszynowe

- Dane musimy przerobić tak, by stały się “przeliczalne”
- Jeżeli uda nam się zamienić na macierz liczb - możemy stworzyć model dla uczenia maszynowego

Zdanie jako wektor

- Dokument = 1 linijka zgrupowanych słów (per IP/cookie)
- Dokument zawiera “Term-y”
- Term-em może być znak, słowo, para słów, link etc.
- Dokument możemy przedstawić jako wektor częstotliwości słów
- Wiele dokumentów = wiele wektorów = macierz ^_^

1. Ala ma kota
2. Kot ma Alę
3. Basia ma kota

Wiersz	Ala	ma	kota	kot	Alę	Basia
1	**1**	**1**	**1**	0	0	0
2	0	**1**	0	**1**	**1**	0
3	0	**1**	**1**	0	0	**1**

1. Ala ma kota
2. Basia ma kota

Wiersz	Ala	Ala ma	ma kota	kota	Basia	Basia ma
1	**1**	**1**	**1**	1	0	0
2	0	0	**1**	**1**	**1**	**1**

TFIDF

- TF = **T**erm **F**requency
- (częstotliwość term-u w dokumencie)
- IDF = **I**nverse **D**ocument **F**requency
- (im rzadziej się term pojawia w dokumentach tym bardziej ma wyższy ranking)

$$\mathrm{TFIDF}(\mathrm{term}, \mathrm{doc}) = \mathrm{TF}(\mathrm{term}, \mathrm{doc}) \times \mathrm{IDF}(\mathrm{term})$$

$$\mathrm{TF} = \mathrm{count}(\mathrm{term}) \;/\; \mathrm{count}(\mathrm{terms\ in\ doc})$$

$$\mathrm{IDF} = \log_{10}\left(\mathrm{count}(\mathrm{docs}) \;/\; \mathrm{count}(\mathrm{docs\ with\ term}) \right)$$

term	TF	IFD	TFIFD
Ala	0.33	log(3/1) ≈ 1.1	0.363
ma	0.33	log(3/3) = 0	0
kota	0.33	log(3/2) ≈ 0.4	0.132

1. Ala ma kota
2. Kot ma Alę
3. Basia ma kota

term	TF	IFD	TFIFD
Kot	0.33	log(3/1) ≈ 1.1	0.363
ma	0.33	log(3/3) = 0	0
Alę	0.33	log(3/1) ≈ 1.1	0.363

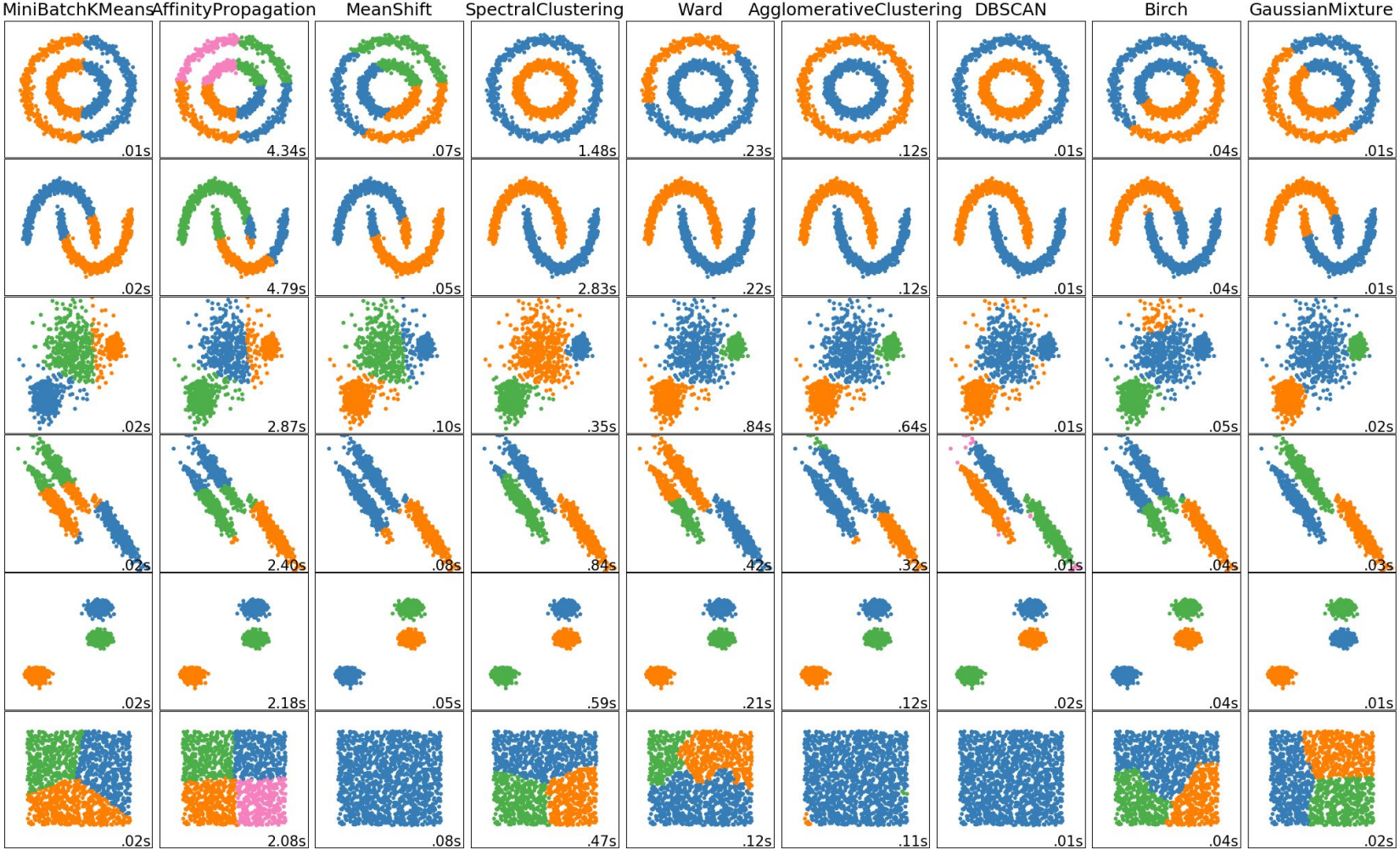
term	TF	IFD	TFIFD
Basia	0.33	log(3/1) ≈ 1.1	0.363
ma	0.33	log(3/3) = 0	0
kota	0.33	log(3/2) ≈ 0.4	0.132

LSA

- LSA - Latent semantic analysis
- Grupuje wektory na podstawie wagi (np. TFIDF) term-ów
- Grupuje term-y na podstawie podobnych wag
- Na podstawie grup jesteśmy w stanie określić relację termów
- Animacja bo ja tego nie wytłumaczę :)

Clustering

- Nie wiemy nic co możemy uzyskać z danych
- Musimy użyć z uczenia nienadzorowanego
- Clustering; Klasteryzacja - na podstawie wartości wektorów jesteśmy w stanie sklasyfikować grupy term-ów

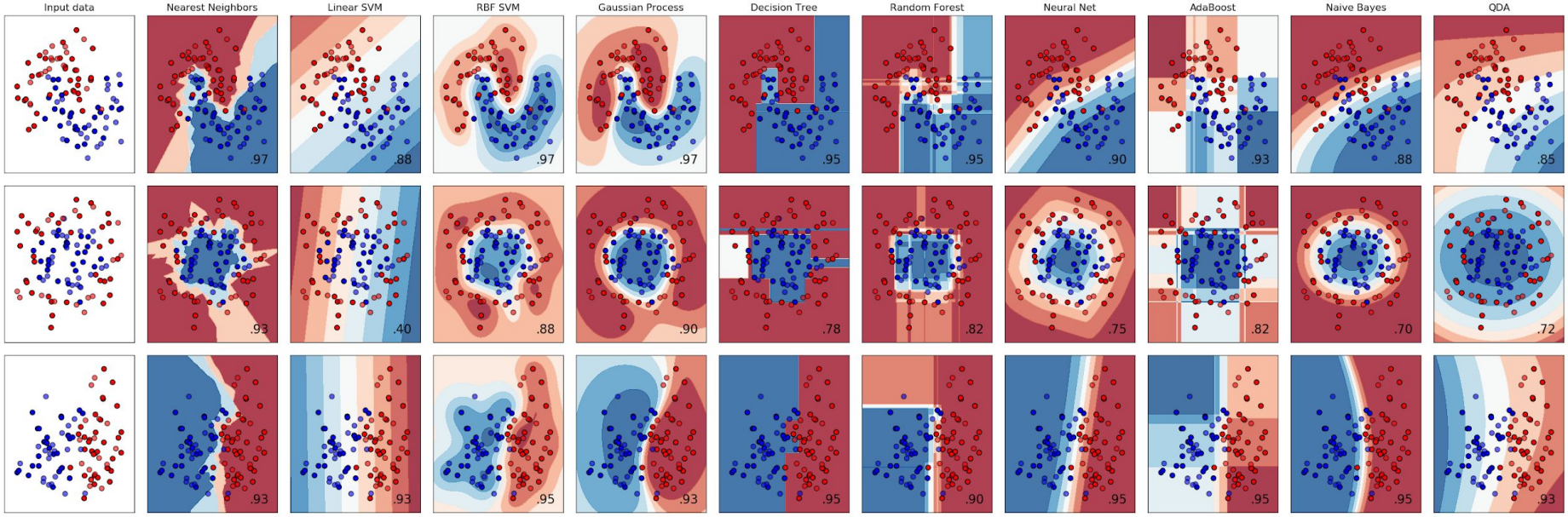

MiniBatchKMeans
AffinityPropagation
MeanShift
SpectralClustering
Ward
AgglomerativeClustering
DBSCAN
Birch
GaussianMixture
.01s
4.34s
.07s
1.48s
.23s
.12s
.01s
.04s
.01s
.02s
4.79s
.05s
2.83s
.22s
.12s
.01s
.04s
.01s
.02s
2.87s
.10s
.35s
.84s
.64s
.01s
.05s
.02s
.02s
2.40s
.08s
.84s
.42s
.32s
.01s
.04s
.03s
.02s
2.18s
.05s
.59s
.21s
.12s
.02s
.04s
.01s
.02s
2.08s
.08s
.47s
.12s
.11s
.01s
.04s
.02s

Clustering

- Każdy jeden wiersz w macierzy możemy potraktować jako wektor w przestrzeni N wymiarowej
- Algorytm k-means: szukanie N najbliższych wektorów tworząc grupę

Klasyfikacja

- Na podstawie stworzonych grup możemy zbudować klasyfikator
- Zbudowany klasyfikator będzie w stanie ocenić ‘grupę’ na podstawie wejścia (w naszym przypadku, zawartości stron po których się porusza)


Input data
Nearest Neighbors
Linear SVM
RBF SVM
Gaussian Process
Decision Tree
Random Forest
Neural Net
AdaBoost
Naive Bayes
QDA
.97
.88
.97
.97
.95
.95
.90
.93
.88
.85
.93
.40
.88
.90
.78
.82
.75
.82
.70
.72
.93
.93
.95
.93
.95
.90
.95
.95
.95
.93

Wynik + co dalej

- Dzięki klasteryzacji słów jesteśmy w stanie stworzyć odpowiednie kampanie reklamowe czy zmienić kategorię produktów na stronie
- Na podstawie cookieID oraz słów kluczowych których używa gość na stronie mogę wyświetlić potencjalne reklamy lub zaproponować odpowiedni produkt
- **WIĘCEJ DANYCH** - z wielu dni + inne dane o klientach

Źródło

1. https://en.wikipedia.org/wiki/Latent_semantic_analysis
2. http://scikit-learn.org/stable/auto_examples/text/document_clustering.html
3. http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html
4. http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html
5. https://pandas.pydata.org/
6. https://github.com/firemark/word-analyser