

Recomendación de Información basada en Análisis de Redes Sociales y Procesamiento de Lenguaje Natural

Trabajo Final de la Licenciatura en Ciencias de la Computación

Pablo Gabriel Celayes

Martes 30 de mayo de 2017

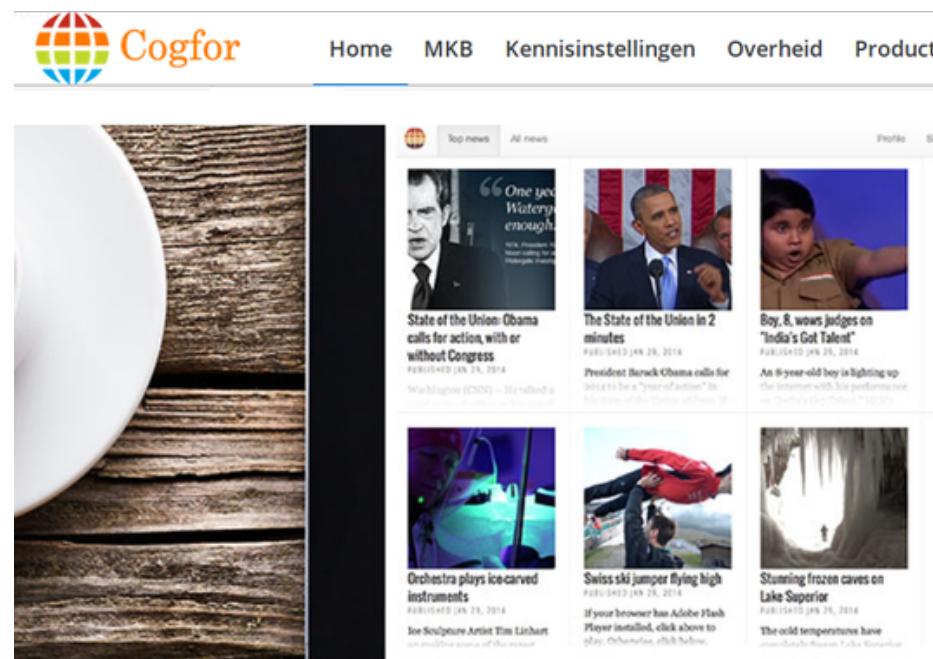
Resumen

- Introducción
- Herramientas
- Datos
- Predicción social
- Agregando PLN
- Conclusiones

Introducción

Idea original

- Recomendador *personalizado* de artículos basado en contenido (NLP)
- Mejorarlo con información social (de fuentes externas o relaciones *inferidas*)
- Preferencias de usuarios de Cogfor



Mutación

- Set the datos propio (usuarios, preferencias, conexiones)
- Predecir preferencias usando información social
- Combinar con recomendación basada en contenido

Herramientas

Datos



SQLAlchemy

Grafos

NetworkX



Análisis



pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



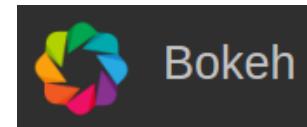
ML + PLN



NLTK

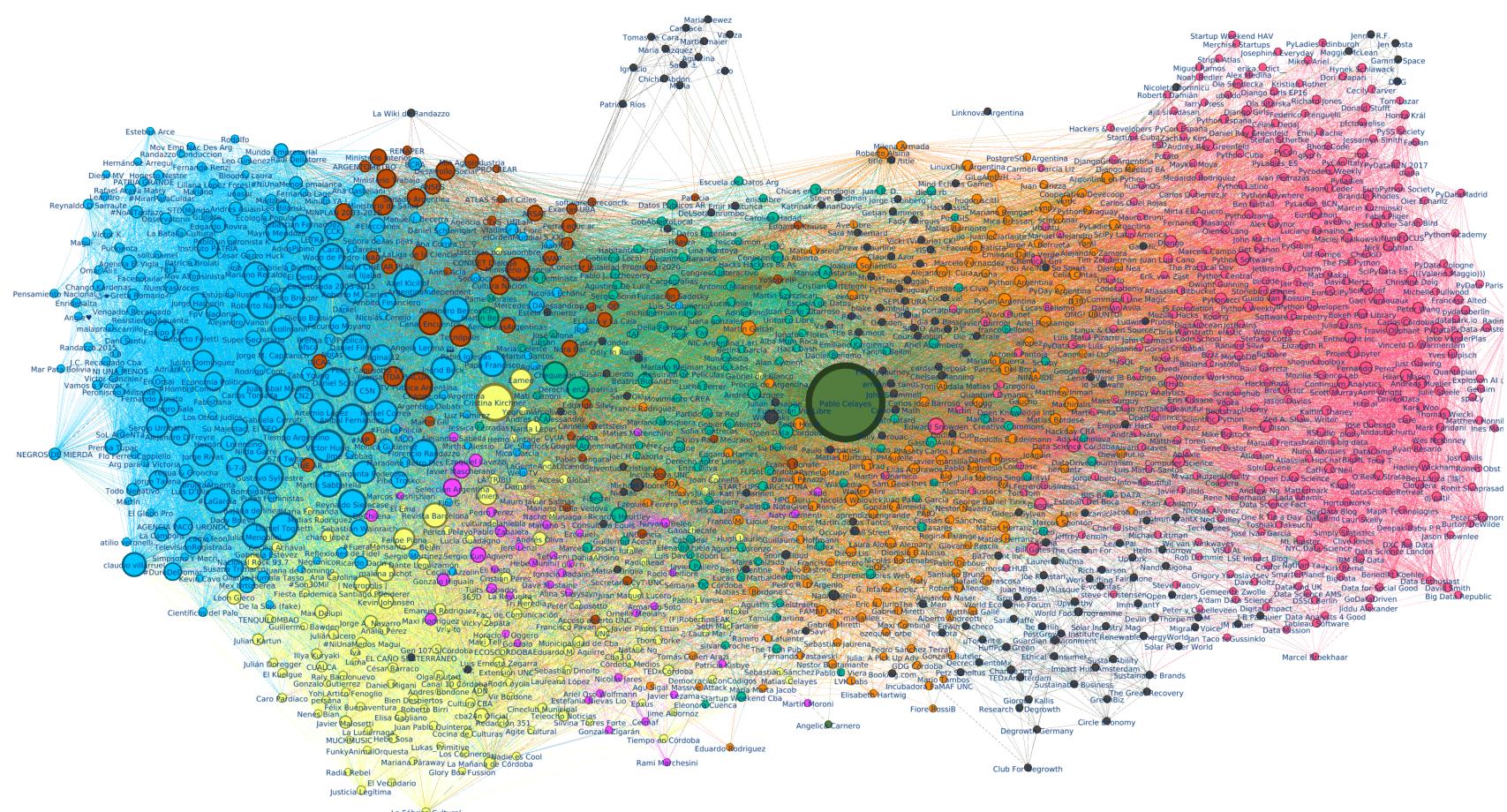


Visualización



Datos: Grafo social

- Todos los usuarios que sigo en Twitter
- relación seguir entre ellos
- 1213 nodos-usuarios
- 40489 aristas-relaciones



Datos: Contenido

- *timelines* entre 18/3/2017 y 17/4/2017 (+ *retweets* y *favs*)
- 163173 tweets totales.
- 109040 en español

Predicción social

¿ Dado un usuario, cuanto puedo saber de sus preferencias sabiendo las de su entorno ?

- **Contenido** = tweets en español
- **Preferencias** = retweets
- **Entorno** = seguidos + seguidos-por-seguidos

Selección de usuarios

- Al menos 100 (re)tweets
- Al menos 100 usuarios en su entorno
- 98 usuarios

Tweets visibles

- $T_u := \bigcup_{x \in \{u\} \cup \text{seguidos}(u)} \text{timeline}(x)$
- máximo 10000 (submuestra de negativos si es necesario)

Extracción de características

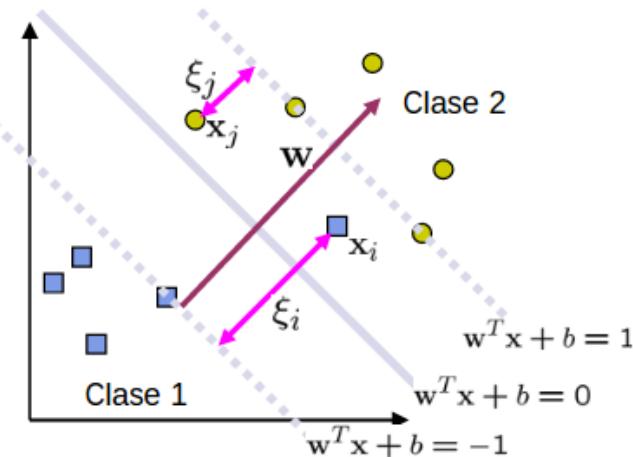
- Entorno $E_u = (\bigcup_{x \in \{u\} \cup \text{seguidos}(u)} \text{seguidos}(x)) - \{u\}$
- $E_u = \{u_1, u_2, \dots, u_n\}$ y $T_u = \{t_1, \dots, t_m\}$
$$M_u := [\text{tweet_en_tl}(t_i, u_j)]_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}}$$
$$y_u := [\text{tweet_en_tl}(t_i, u)]_{1 \leq i \leq m}$$

Problema de clasificación

- Predecir y_u en base a filas de M_u
- Particionado:
 - 70% entrenamiento (M_u^{en}, y_u^{en})
 - 10% ajuste (M_u^{aj}, y_u^{aj})
 - 20% evaluación (M_u^{ev}, y_u^{ev})

Support Vector Machines

- Objetivo: **maximizar** margen y **minimizar** errores
- Funciones *kernel* permiten encontrar fronteras **no lineales**:
 - *Radial Basis Function*
 - Polinómico



Calidad de clasificación

$$\text{precision} := \frac{|\{x_i | f(x_i) = 1 \text{ y } y_i = 1\}|}{|\{x_i | f(x_i) = 1\}|}$$

$$\text{recall} := \frac{|\{x_i | f(x_i) = 1 \text{ y } y_i = 1\}|}{|\{x_i | y_i = 1\}|}$$

$$F1 := \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

Ajuste de parámetros

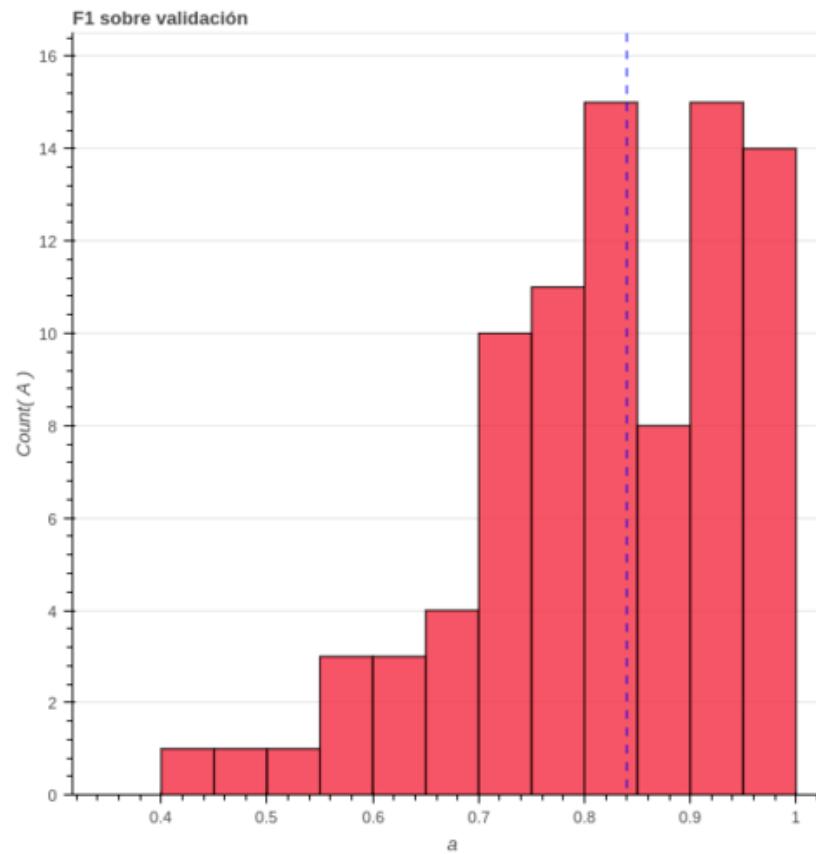
- Búsqueda exhaustiva GridSearchCV
- Validación cruzada en 3 partes
- Objetivo: maximizar $F1$
- Grilla:

```
{  
    "C": [ 0.01, 0.1, 1 ],  
    "class_weight": [ "balanced", None ],  
    "gamma": [ 0.1, 1, 10 ],  
    "kernel": [ "rbf", "poly" ]  
}
```

- C : controla balance entre margen y errores
- $class_weight$: ¿dar más importancia a clase minoritaria?
- $gamma$: forma de la frontera de decisión

Resultados

- $F1$ sobre M_u^{aj}
- Promedio 0,84



Agregando PLN

Selección de usuarios

- $F1 < 0,75$ en M_u^{aj} (23 usuarios)
- 10 usuarios más (al azar)

Pre-procesamiento

- Normalización
- Tokenizado
- Extracción de frases
- Diccionario de términos
- *Bag of terms*
- LDA

Normalización

```
print tweet
```

```
#HaceInstantes Abrazo al Congreso organizado por gremios docentes después de la represión policial de ayer. https://t.co/ToyertpI70
```

```
tweet = normalize(tweet)  
print tweet
```

```
#haceinstantes abrazo al congreso organizado por gremios docentes despues de la represion policial de ayer
```

Tokenizado

```
tweet = tokenize(tweet)  
print tweet
```

```
[u'haceinst', u'abraz', u'al', u'congres', u'organiz', u'por', u'gremi', u'docent', u'despu', u'de', u'la', u'repre  
sion', u'policial', u'de', u'ayer']
```

Extracción de frases

- Phraser de gensim
- Bigramas $a b$ relevantes:
 - $min_count = 5$
 - $cnt(a, b) \geq min_count$
 - $\frac{(cnt(a,b)-min_count)*N}{cnt(a)*cnt(b)} > 10$
- 2 pasadas sobre el corpus tokenizado (bigramas y trigramas)

Diccionario de términos

- T retokenizado con frases
- término = palabra o frase
- significativo (al menos 3 veces).
- informativo (en menos del 30% de los tweets).
- Diccionario D de 26201 términos.

Bag of terms

- Texto t : → multiconjunto (*bag*) de los términos en t .
- No importa el orden, pero sí repeticiones.
- En tweets (≤ 140 caracteres), en general son *conjuntos* (0 o 1 ocurrencia).
- Diccionario fijo $D = \{t_1, \dots, t_{26201}\}$: → vector de características enteras (booleanas):

$$v_{BOT}(\text{tweet}) = [count(t_i, tokens(tweet))]_{i=1}^{26201}$$

- v_{BOT} se representa ralo (*sparse*):

```
ind = 90172
print all_tweets_text_es[ind]
#HaceInstantes Abrazo al Congreso organizado por gremios docentes después de la represión policial de ayer. http
s://t.co/ToyertpI70

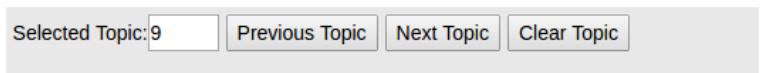
print [dictionary[i] for (i,c) in bow[ind]]
[u'represion_policial', u'gremi_docent', u'ayer', u'organiz', u'abraz_congres', u'despu']

print bow[ind]
[(221, 1), (1176, 1), (3055, 1), (6257, 1), (14203, 1), (25141, 1)]
```

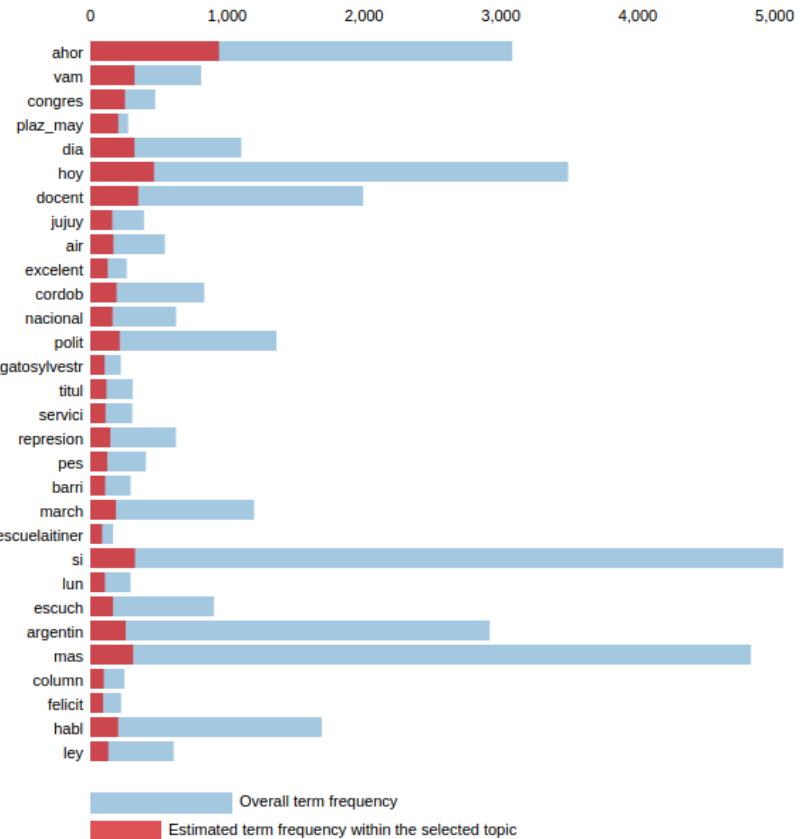
LDA

- Descubre temas subyacentes en textos
- Reducción de dimensionalidad (espacio de *términos* a espacio de *temas*)
- Probamos modelos de 10 y 20 temas

Ejemplo: LDA 10 temas



Top-30 Most Relevant Terms for Topic 9 (9.1% of tokens)



1. saliency(term, w) = frequency(w) * [sum_t p(t | w) * log(p(t | w) / p(t))] for topics t; see Chuang et. al (2012)
 2. relevance(term, w | topic, t) = $\lambda * p(w | t) + (1 - \lambda) * p(w | t) / p(w)$; see Sievert & Shirley (2014)

Extendiendo características

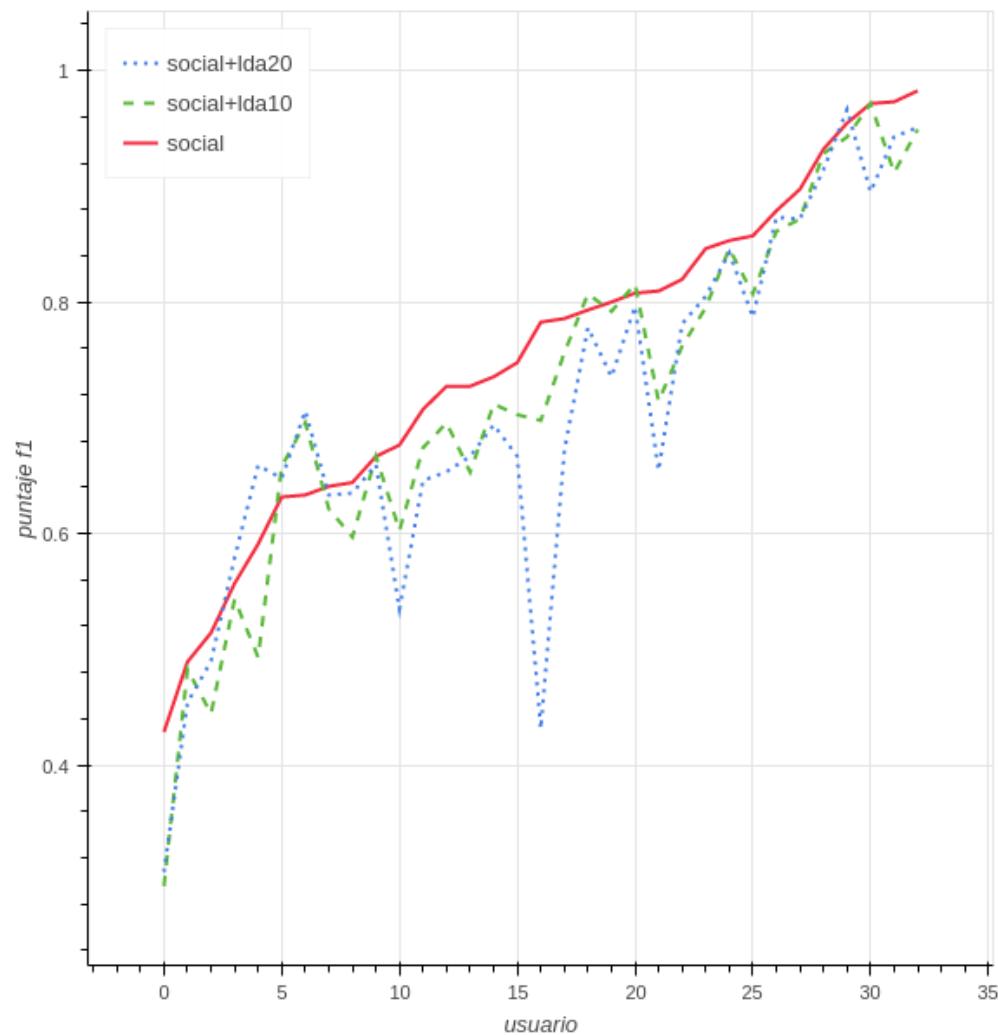
- Características sociales + LDA:

```
[1, 0, 0, 1, ..., 0, 1, 0.15, 0.25, ..., 0, 0.05]
```

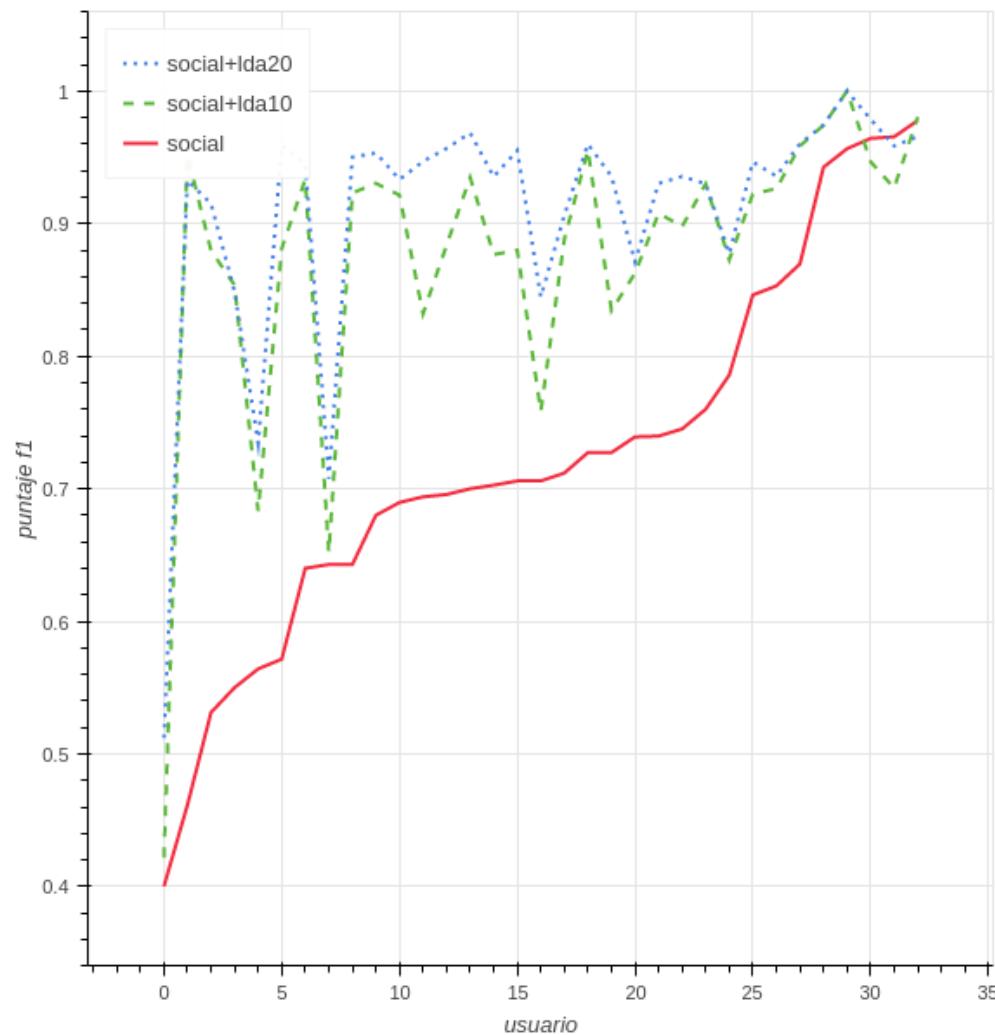
- Agregamos escalado (cada columna con media 0 y varianza 1)

Evaluando sobre M_u^{ev}

- *LDA10*
 - mejora para 4 usuarios
 - mejora media de 2, 8%
- *LDA20*
 - mejora para 5 usuarios
 - mejora media de 3, 8%.



Sobreajuste



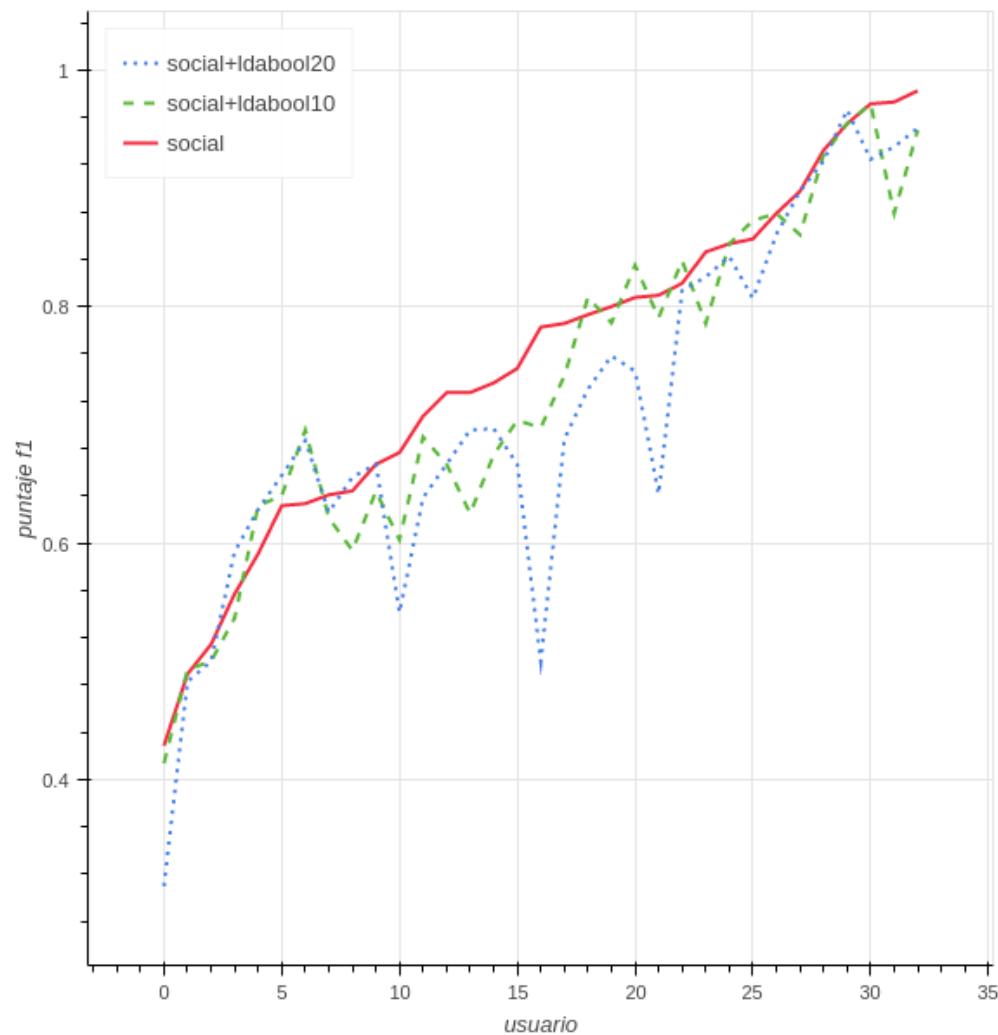
Discretizando...

- Sobreajuste puede venir de características *demasiado* descriptivas
- Asignamos 1 para cada tema con puntaje $\geq 0,25$ y 0 en caso contrario

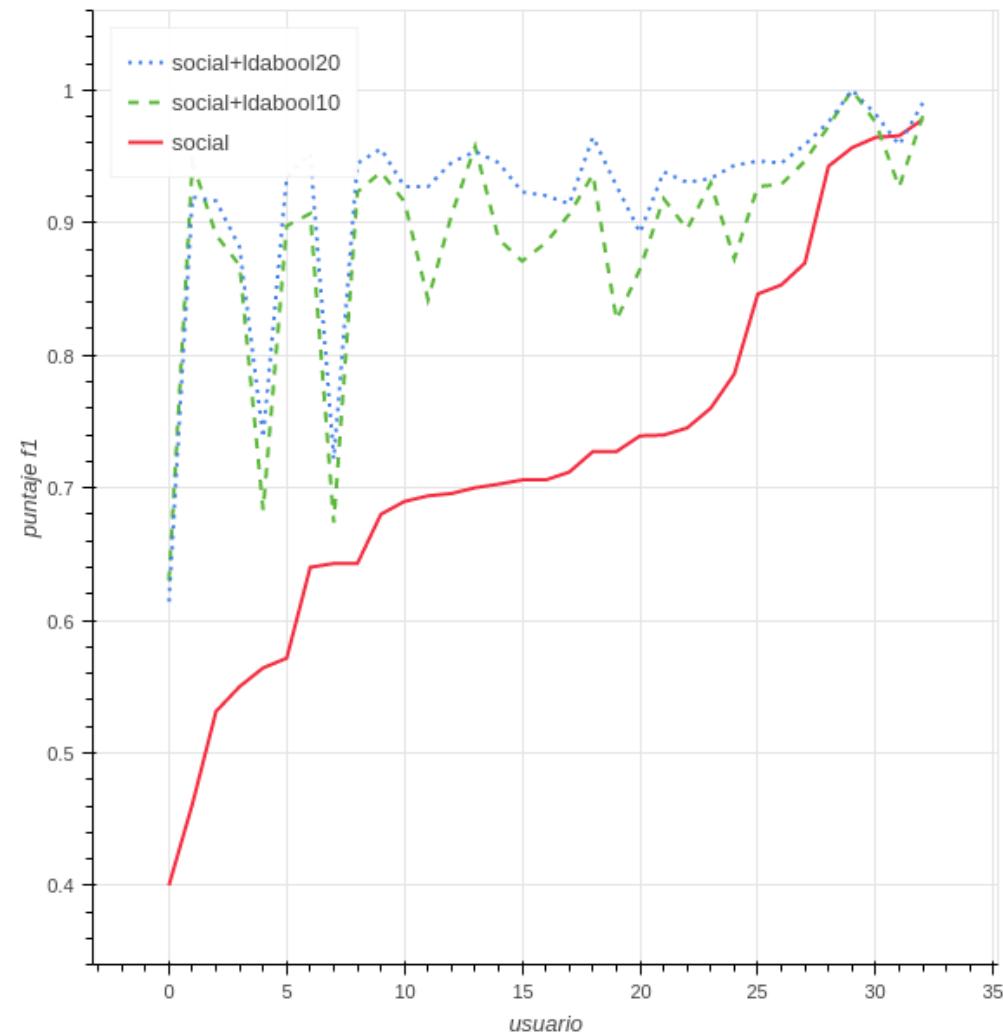
Mejoras?

- *LDAbool10*
 - mejora para 8 usuarios
 - mejora media de 2,4%
- *LDAbool20*
 - mejora para 7 usuarios
 - mejora media de 2,5%.

Impacta a más usuarios, pero mejora menos



...pero sigue el sobreajuste



Conclusiones

- Predicción social pura mejor de lo esperado
- LDA mejora casos flojos, pero hay que resolver sobreajuste
- Twitter es muy buena fuente de datos

Próximos pasos

- Reducir sobreajuste
- Características adicionales
- Considerar temporalidad
- Generalizar (modelo que no dependa del usuario)

¿Preguntas?



@PCelayers



https://github.com/pablocelayers/sna_classifier (https://github.com/pablocelayers/sna_classifier)