Text Mining

Marcelo Mendoza
http://www.inf.utfsm.cl/~mmendoza
mmendoza@inf.utfsm.cl
A 131, Campus San Joaquín - UTFSM

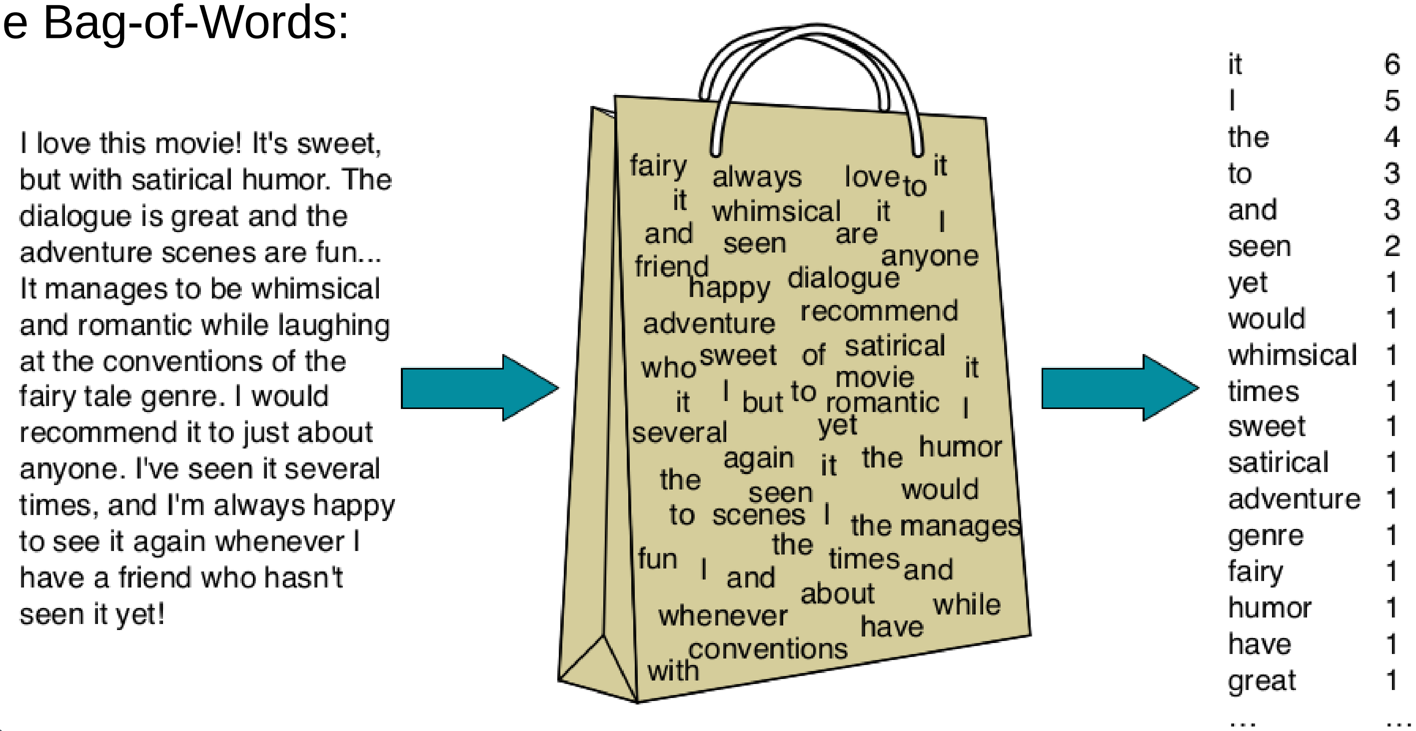
Clasificación de texto

Naive Bayes: es un método clave en clasificación de texto ya que tiene una conexión con los modelos de lenguaje.

Naive Bayes: es un método clave en clasificación de texto ya que tiene una conexión con los modelos de lenguaje.

Objetivo: Modelar documentos para clasificarlos.

El enfoque Bag-of-Words:



Clasificación de texto

Clasificación Bayesiana:

Necesitamos modelar el documento

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d)$$

Clasificación de texto

Clasificación Bayesiana:

Necesitamos modelar el documento

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d)$$

Usando la regla de Bayes:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \longrightarrow \hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c)P(c)}{P(d)}$$

Clasificación de texto

Clasificación Bayesiana:

Necesitamos modelar el documento

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d)$$

Usando la regla de Bayes:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \longrightarrow \hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c)P(c)}{P(d)}$$

Para un documento fijo, podemos descartar el denominador:

Prior de la clase

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} P(d|c)P(c)$$

likelihood

Clasificación de texto

Clasificación Bayesiana:

Necesitamos modelar el documento

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d)$$

Usando la regla de Bayes:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \longrightarrow \hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d|c)P(c)}{P(d)}$$

Para un documento fijo, podemos descartar el denominador:

Prior de la clase

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|d) = \underset{c \in C}{\operatorname{argmax}} P(d|c)P(c)$$

likelihood

Supuesto 'naive': independencia condicional entre palabras dada la clase:

$$P(f_1, f_2, ..., f_n|c) = P(f_1|c) \cdot P(f_2|c) \cdot ... \cdot P(f_n|c)$$

Features que codifican a las palabras

Clasificación Bayesiana:

Necesitamos modelar el documento

$$\hat{c} = \operatorname*{argmax}_{c \in C} P(c|d)$$

Usando la regla de Bayes:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \longrightarrow \hat{c} = \operatorname*{argmax}_{c \in C} P(c|d) = \operatorname*{argmax}_{c \in C} \frac{P(d|c)P(c)}{P(d)}$$

Para un documento fijo, podemos descartar el denominador:

Prior de la clase

$$\hat{c} = \operatorname*{argmax}_{c \in C} P(c|d) = \operatorname*{argmax}_{c \in C} P(d|c)P(c)$$

likelihood

Supuesto 'naive': independencia condicional entre palabras dada la clase:

$$P(f_1, f_2, ..., f_n|c) = P(f_1|c) \cdot P(f_2|c) \cdot ... \cdot P(f_n|c)$$

Features que codifican a las palabras

Finalmente: $$c_{NB} = \operatorname*{argmax}_{c \in C} P(c) \prod_{f \in F} P(f|c)$$

Clasificación Bayesiana: al igual que en modelos de lenguaje, muchas veces es más práctico construir el clasificador en el log space.

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} \log P(c) + \sum_{i \in positions} \log P(w_i|c)$$

Palabra *i*-ésima en el documento

Clasificación de texto

Clasificación Bayesiana: al igual que en modelos de lenguaje, muchas veces es más práctico construir el clasificador en el log space.

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} \log P(c) + \sum_{i \in positions} \log P(w_i|c)$$

Palabra *i*-ésima en el documento

Entrenamiento basado en corpus:

$$\hat{P}(c) = \frac{N_c}{N_{doc}}$$

$$\hat{P}(w_i|c) = \frac{count(w_i, c)}{\sum_{w \in V} count(w, c)}$$

Clasificación de texto

Clasificación Bayesiana: al igual que en modelos de lenguaje, muchas veces es más práctico construir el clasificador en el log space.

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} \log P(c) + \sum_{i \in positions} \log P(w_i|c)$$

Palabra *i*-ésima en el documento

Entrenamiento basado en corpus:

$$\hat{P}(c) = \frac{N_c}{N_{doc}} \qquad \hat{P}(w_i|c) = \frac{count(w_i, c)}{\sum_{w \in V} count(w, c)}$$

Variante con suavizado de Laplace para OOV:

$$\hat{P}(w_i|c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} (count(w, c) + 1)} = \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

Clasificación de texto

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms
  N_{doc} = number of documents in D
  N_c = number of documents from D in class c
  $logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$
  $V \leftarrow$ vocabulary of D
  $bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c
  for each word w in V # Calculate $P(w|c)$ terms
    $count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
    $$loglikelihood[w,c] \leftarrow \log \frac{count(w,c) + 1}{\sum_{w' \text{ in } V} (count(w',c) + 1)}$$
return $logprior$, $loglikelihood$, V

Clasificación de texto

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms

N_{doc} = number of documents in D

N_c = number of documents from D in class c

$logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$

$V \leftarrow$ vocabulary of D

$bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c

for each word w in V # Calculate $P(w|c)$ terms

$count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$

$$loglikelihood[w,c] \leftarrow \log \frac{count(w,c) + 1}{\sum_{w' \text{ in } V} (count(w',c) + 1)}$$

return $logprior$, $loglikelihood$, V

function TEST NAIVE BAYES(*testdoc*, *logprior*, *loglikelihood*, C, V) **returns** best c

for each class $c \in C$

$sum[c] \leftarrow logprior[c]$

for each position i in *testdoc*

$word \leftarrow testdoc[i]$

if $word \in V$

$sum[c] \leftarrow sum[c] + loglikelihood[word,c]$

return $\text{argmax}_c\ sum[c]$

Clasificación de texto

Naive Bayes en Python (sklearn)

	abstract	category	label
0	In this paper I explore the possibility and ra...	Epistemology	0
1	Merleau-Ponty identifies an intertwined affect...	Epistemology	0
2	This is an inquiry into the economic psycholog...	Epistemology	0
3	The paper begins with the account of a focus g...	Epistemology	0
4	The recent accounting scandals have raised con...	Epistemology	0

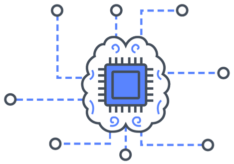
Clasificación de texto

Naive Bayes en Python (sklearn)

	abstract	category	label
0	In this paper I explore the possibility and ra...	Epistemology	0
1	Merleau-Ponty identifies an intertwined affect...	Epistemology	0
2	This is an inquiry into the economic psycholog...	Epistemology	0
3	The paper begins with the account of a focus g...	Epistemology	0
4	The recent accounting scandals have raised con...	Epistemology	0

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df[‘abstract’],
df[‘label’], random_state=1)
```



Clasificación de texto

Naive Bayes en Python (sklearn)

	abstract	category	label
0	In this paper I explore the possibility and ra...	Epistemology	0
1	Merleau-Ponty identifies an intertwined affect...	Epistemology	0
2	This is an inquiry into the economic psycholog...	Epistemology	0
3	The paper begins with the account of a focus g...	Epistemology	0
4	The recent accounting scandals have raised con...	Epistemology	0

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df['abstract'],
df['label'], random_state=1)
```

```
from sklearn.feature_extraction.text import CountVectorizer

cv = CountVectorizer(strip_accents='ascii', token_pattern=u'(?ui)\\b
\\w*[a-z]+\\w*\\b', lowercase=True, stop_words='english')

X_train_cv = cv.fit_transform(X_train)
X_test_cv = cv.transform(X_test)
```

Clasificación de texto

Naive Bayes en Python (sklearn)

	abstract	category	label
0	In this paper I explore the possibility and ra...	Epistemology	0
1	Merleau-Ponty identifies an intertwined affect...	Epistemology	0
2	This is an inquiry into the economic psycholog...	Epistemology	0
3	The paper begins with the account of a focus g...	Epistemology	0
4	The recent accounting scandals have raised con...	Epistemology	0

```
word_freq_df = pd.DataFrame(X_train_cv.toarray(),
columns=cv.get_feature_names())
```

Clasificación de texto

Naive Bayes en Python (sklearn)

	abstract	category	label
0	In this paper I explore the possibility and ra...	Epistemology	0
1	Merleau-Ponty identifies an intertwined affect...	Epistemology	0
2	This is an inquiry into the economic psycholog...	Epistemology	0
3	The paper begins with the account of a focus g...	Epistemology	0
4	The recent accounting scandals have raised con...	Epistemology	0

```
word_freq_df = pd.DataFrame(X_train_cv.toarray(),
columns=cv.get_feature_names())
```

```
from sklearn.naive_bayes import MultinomialNB
naive_bayes = MultinomialNB()
naive_bayes.fit(X_train_cv, y_train)
predictions = naive_bayes.predict(X_test_cv)
```

Clasificación de texto

Naive Bayes en Python (sklearn)

	abstract	category	label
0	In this paper I explore the possibility and ra...	Epistemology	0
1	Merleau-Ponty identifies an intertwined affect...	Epistemology	0
2	This is an inquiry into the economic psycholog...	Epistemology	0
3	The paper begins with the account of a focus g...	Epistemology	0
4	The recent accounting scandals have raised con...	Epistemology	0

```
word_freq_df = pd.DataFrame(X_train_cv.toarray(),
columns=cv.get_feature_names())
```

```
from sklearn.naive_bayes import MultinomialNB
naive_bayes = MultinomialNB()
naive_bayes.fit(X_train_cv, y_train)
predictions = naive_bayes.predict(X_test_cv)
```

```
from sklearn.metrics import accuracy_score, precision_score,
recall_score

print('Accuracy score: ', accuracy_score(y_test, predictions))
print('Precision score: ', precision_score(y_test, predictions))
print('Recall score: ', recall_score(y_test, predictions))
```

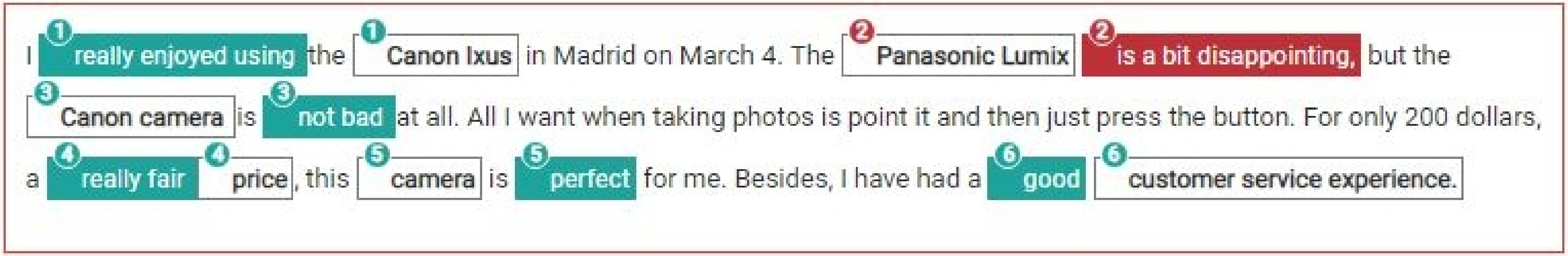
Aplicación de clasificación de texto

Sentiment analysis

I really enjoyed using the Canon Ixus in Madrid on March 4. The Panasonic Lumix is a bit disappointing, but the Canon camera is not bad at all. All I want when taking photos is point it and then just press the button. For only 200 dollars, a really fair price, this camera is perfect for me. Besides, I have had a good customer service experience.

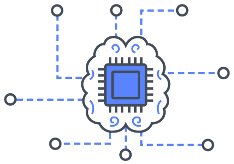
Aplicación de clasificación de texto

Sentiment analysis



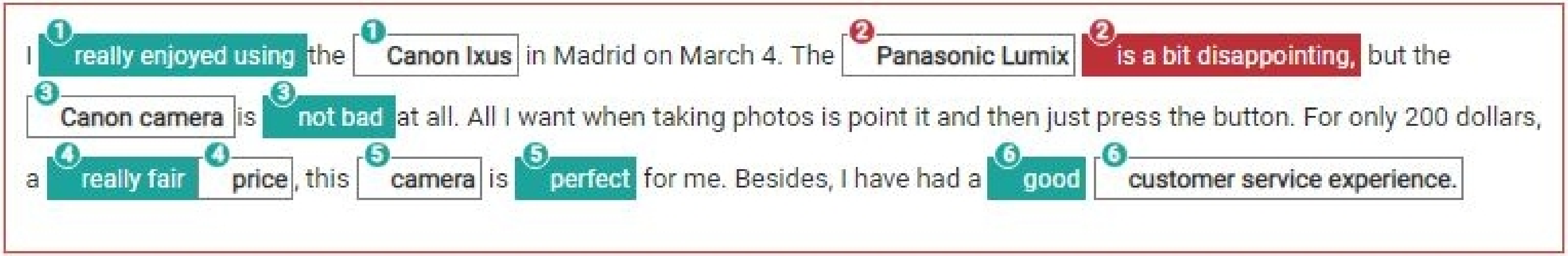
: entidades (1, 2, 3, 5) o atributos de entidades (4, 6)

: opiniones (1, 2, 4, 5, 6)



Aplicación de clasificación de texto

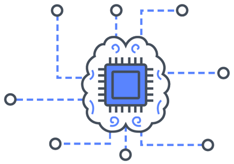
Sentiment analysis



: entidades (1, 2, 3, 5) o atributos de entidades (4, 6)

: opiniones (1, 2, 4, 5, 6)

¿Cuál es la polaridad con respecto a la entidad?



Aplicación de clasificación de texto

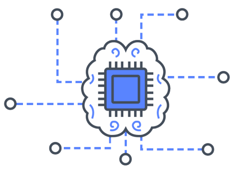
Sentiment analysis

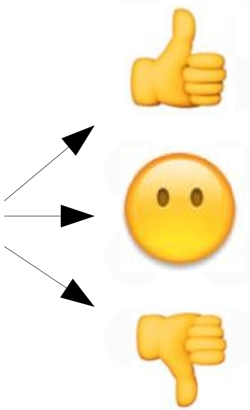
I really enjoyed using the Canon Ixus in Madrid on March 4. The Panasonic Lumix is a bit disappointing, but the Canon camera is not bad at all. All I want when taking photos is point it and then just press the button. For only 200 dollars, a really fair price, this camera is perfect for me. Besides, I have had a good customer service experience.

: entidades (1, 2, 3, 5) o atributos de entidades (4, 6)

: opiniones (1, 2, 4, 5, 6)

¿Cuál es la polaridad con respecto a la entidad?





Aplicación de clasificación de texto

Sentiment analysis

- Percepción de nuevos productos
- Percepción de marcas
- Análisis de reputación
- Análisis temporal de opiniones (dinámicas)
- Word-of-mouth es importante para viral marketing y otros fenómenos de propagación en redes sociales

Sentiment analysis

$$\text{Tripla: } (O_i, h_j, s_{ij})$$

- O_i: la entidad en cuestión (ej.: producto)
- h_j: el opinólogo (ej.: Ud.)
- so_{ij}: la orientación de h_j con respecto a O_i.

Sentiment analysis

- Factual and opinionated. Ej.:

Sentiment analysis

- Factual and opinionated. Ej.:

Aplicación de clasificación de texto

Sentiment analysis

- Idea: a partir de ejemplos etiquetados, y características POS tags, construir clasificadores.
- Algunos resultados (dependientes del algoritmo de entrenamiento (y de los datos!)):

  NRCNaive Bayes, EM, SVM, ...Andan bien (80 % en accuracy, app.).

Clasificación de texto

Regresión logística: muy usado en clasificación de texto (strong baseline).

A diferencia de naive Bayes, que puede ser considerado un método generativo, LR es discriminativo.

Entrada (vector de características)

$$z = \left(\sum_{i=1}^{n} w_i x_i\right) + b$$

bias

Peso (parámetro del modelo)

Regresión logística: muy usado en clasificación de texto (strong baseline).

A diferencia de naive Bayes, que puede ser considerado un método generativo, LR es discriminativo.

Entrada (vector de características)

$$z = \left(\sum_{i=1}^{n} w_i x_i \right) + b$$

bias

Peso (parámetro del modelo)

En notación vectorial: $z = w \cdot x + b$

Producto punto

Clasificación de texto

Regresión logística: muy usado en clasificación de texto (strong baseline).

A diferencia de naive Bayes, que puede ser considerado un método generativo, LR es discriminativo.

Entrada (vector de características)

$$z = \left(\sum_{i=1}^{n} w_i x_i\right) + b$$

bias

Peso (parámetro del modelo)

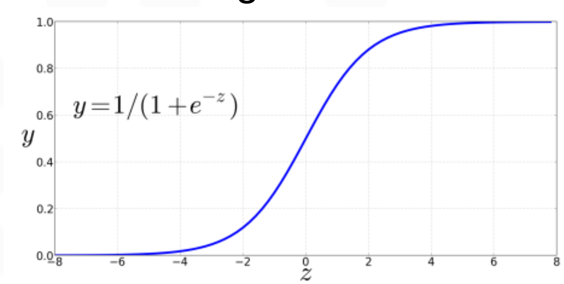
En notación vectorial: $z = w \cdot x + b$

Producto punto

Podemos forzar a z a ser una probabilidad, i.e. z in [0, 1]:

$$y = \sigma(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+\exp(-z)}$$

sigmoid



Clasificación de texto

La sigmoide nos permite obtener un escalar en [0, 1]. Si el problema de clasificación es binario, para transformar z en probabilidad hacemos lo siguiente:

$$\begin{aligned} P(y=1) &= \sigma(w \cdot x + b) \\ &= \frac{1}{1+\exp(-(w \cdot x + b))} \end{aligned}$$

$$\begin{aligned} P(y=0) &= 1-\sigma(w \cdot x + b) \\ &= 1-\frac{1}{1+\exp(-(w \cdot x + b))} \\ &= \frac{\exp(-(w \cdot x + b))}{1+\exp(-(w \cdot x + b))} \end{aligned}$$

Clasificación de texto

La sigmoide nos permite obtener un escalar en [0, 1]. Si el problema de clasificación es binario, para transformar z en probabilidad hacemos lo siguiente:

$$
\begin{aligned}
P(y=1) &= \sigma(w \cdot x + b) \\
&= \frac{1}{1+\exp(-(w \cdot x + b))}
\end{aligned}
$$

$$
\begin{aligned}
P(y=0) &= 1-\sigma(w \cdot x + b) \\
&= 1-\frac{1}{1+\exp(-(w \cdot x + b))} \\
&= \frac{\exp(-(w \cdot x + b))}{1+\exp(-(w \cdot x + b))}
\end{aligned}
$$

Luego, la clasificación se realiza según:

$$
\hat{y} = \begin{cases} 1 & \text{if } P(y=1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}
$$

Aplicación en sentiment analysis (binario: +/-)

Características (6):

Var	Definition
x_1	count(positive lexicon) $\in$ doc)
x_2	count(negative lexicon) $\in$ doc)
x_3	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$
x_4	count(1st and 2nd pronouns $\in$ doc)
x_5	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$
x_6	log(word count of doc)

Aplicación en sentiment analysis (binario: +/-)

Características (6):

Var	Definition
x_1	count(positive lexicon) $\in$ doc)
x_2	count(negative lexicon) $\in$ doc)
x_3	$\begin{cases} 1 & \text{if ``no''} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$
x_4	count(1st and 2nd pronouns $\in$ doc)
x_5	$\begin{cases} 1 & \text{if ``!''} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$
x_6	log(word count of doc)

Doc (test):

$x_2=2$ $x_3=1$

It's hokey . There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ? For one thing , the cast is great . Another nice touch is the music I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you .

$x_1=3$ $x_5=0$ $x_6=4.19$ $x_4=3$

Aplicación en sentiment analysis (binario: +/-)

Parámetros del modelo (aprendidos)

$$
\begin{aligned}
p(+|x) = P(Y=1|x) &= \sigma(w \cdot x + b) \\
&= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\
&= \sigma(.833) \\
&= 0.70 \\
p(-|x) = P(Y=0|x) &= 1 - \sigma(w \cdot x + b) \\
&= 0.30
\end{aligned}
$$

Aplicación en sentiment analysis (binario: +/-)

Parámetros del modelo (aprendidos)

$$
\begin{aligned}
p(+|x) = P(Y=1|x) &= \sigma(w \cdot x + b) \\
&= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\
&= \sigma(.833) \\
&= 0.70 \\
p(-|x) = P(Y=0|x) &= 1 - \sigma(w \cdot x + b) \\
&= 0.30
\end{aligned}
$$

Entrenamiento:

Cross-entropy loss (MLE condicional): escogemos los parámetros que maximizan la probabilidad de las etiquetas verdaderas en el training set.

En clasificación binaria, corresponde a una MLE Bernoulli:

$$p(y|x) = \hat{y}^{y} (1 - \hat{y})^{1-y}$$

Entrenamiento de la LR

likelihood

En log space:

$$\begin{aligned} \log p(y|x) &= \log\left[\hat{y}^{y}\,(1-\hat{y})^{1-y}\right] \\ &= y\log\hat{y} + (1-y)\log(1-\hat{y}) \end{aligned}$$

Entrenamiento de la LR

likelihood

En log space:

$$\begin{aligned} \log p(y|x) &= \log\left[\hat{y}^{y}\,(1-\hat{y})^{1-y}\right] \\ &= y\log\hat{y} + (1-y)\log(1-\hat{y}) \end{aligned}$$

Para transformar la verosimilitud en pérdida de información usamos el signo -:

$$L_{\mathrm{CE}}(\hat{y},y) = -\log p(y|x) \;=\; -\left[y\log\hat{y} + (1-y)\log(1-\hat{y})\right]$$

Entrenamiento de la LR

likelihood

En log space:

$$\log p(y|x) = \log\left[\hat{y}^{y}\,(1-\hat{y})^{1-y}\right]$$
$$= y\log\hat{y} + (1-y)\log(1-\hat{y})$$

Para transformar la verosimilitud en pérdida de información usamos el signo -:

$$L_{\mathrm{CE}}(\hat{y},y) = -\log p(y|x) = -[y\log\hat{y} + (1-y)\log(1-\hat{y})]$$

Finalmente, para la LR, reemplazamos la estimación por la sigmoide:

$$L_{\mathrm{CE}}(\hat{y},y) = -[y\log\sigma(w\cdot x+b) + (1-y)\log(1-\sigma(w\cdot x+b))]$$

Entrenamiento de la LR

likelihood

En log space:

$$\begin{aligned}\log p(y|x) &= \log\left[\hat{y}^{y}\,(1-\hat{y})^{1-y}\right] \\ &= y\log\hat{y} + (1-y)\log(1-\hat{y})\end{aligned}$$

Para transformar la verosimilitud en pérdida de información usamos el signo -:

$$L_{\mathrm{CE}}(\hat{y},y) = -\log p(y|x) \;=\; -[y\log\hat{y} + (1-y)\log(1-\hat{y})]$$

Finalmente, para la LR, reemplazamos la estimación por la sigmoide:

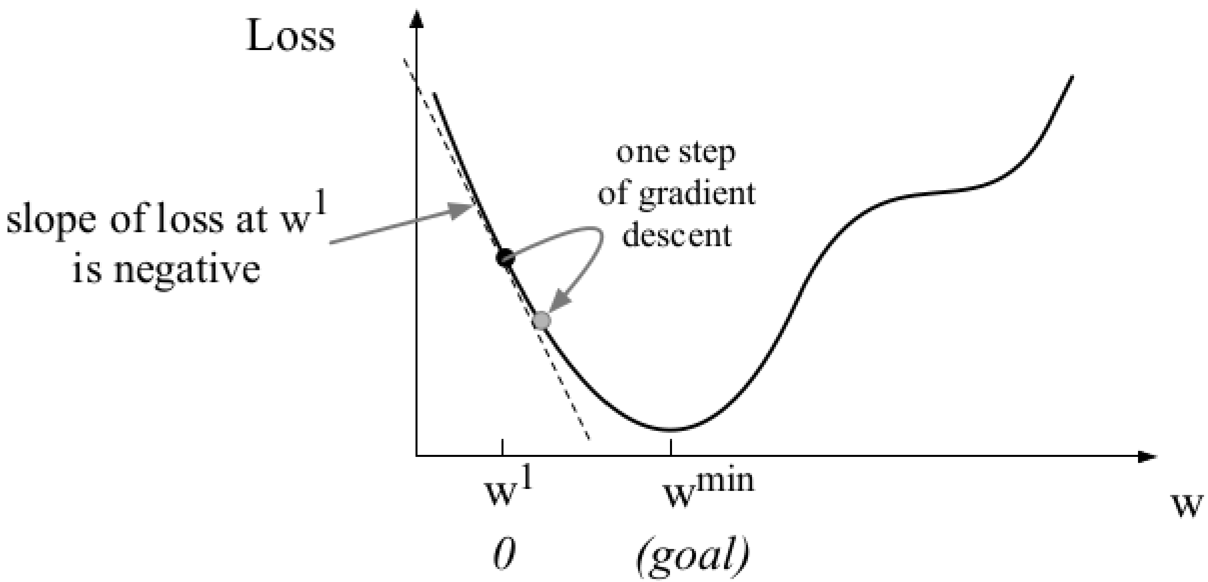
$$L_{\mathrm{CE}}(\hat{y},y) \;=\; -[y\log\sigma(w\cdot x+b) + (1-y)\log(1-\sigma(w\cdot x+b))]$$

Entrenamiento usando
m instancias:

$$\hat{\theta} \;=\; \underset{\theta}{\operatorname{argmin}}\frac{1}{m}\sum_{i=1}^{m} L_{\mathrm{CE}}(f(x^{(i)};\theta),y^{(i)})$$

Entrenamiento de la LR

Como la cross-entropy es convexa, la podemos optimizar usando gradiente descendente:



Idea: mover w en el sentido contrario de la pendiente de la función de pérdida.

Learning rate

Gradiente!

$$w^{t+1} = w^t - \eta \frac{d}{dw} f(x;w)$$

Entrenamiento de la LR

Como trabajamos con un vector de características, necesitamos calcular:

$$\nabla_\theta L(f(x;\theta),y)) = \begin{bmatrix} \frac{\partial}{\partial w_1} L(f(x;\theta),y) \\ \frac{\partial}{\partial w_2} L(f(x;\theta),y) \\ \vdots \\ \frac{\partial}{\partial w_n} L(f(x;\theta),y) \end{bmatrix}$$

Por lo tanto: $\theta_{t+1} = \theta_t - \eta \nabla L(f(x;\theta),y)$

Entrenamiento de la LR

Como trabajamos con un vector de características, necesitamos calcular:

$$\nabla_\theta L(f(x;\theta),y)) = \begin{bmatrix} \frac{\partial}{\partial w_1} L(f(x;\theta),y) \\ \frac{\partial}{\partial w_2} L(f(x;\theta),y) \\ \vdots \\ \frac{\partial}{\partial w_n} L(f(x;\theta),y) \end{bmatrix}$$

Por lo tanto: $\theta_{t+1} = \theta_t - \eta \nabla L(f(x;\theta),y)$

Regularización: se usa para evitar over-fitting, penalizando pesos grandes.

$$\hat{\theta} = \operatorname*{argmax}_{\theta} \sum_{i=1}^{m} \log P(y^{(i)}|x^{(i)}) - \alpha R(\theta)$$

L_2 $$R(\theta) = ||\theta||_2^2 = \sum_{j=1}^{n} \theta_j^2$$

L_1 $$R(\theta) = ||\theta||_1 = \sum_{i=1}^{n} |\theta_i|$$

Extensión de la LR a clasificación multi-clase

LR multinomial: útiles para clasificación multi-clase.

Regresión logística multinomial (softmax):

$$\mathrm{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad 1 \leq i \leq k$$

Extensión de la LR a clasificación multi-clase

LR multinomial: útiles para clasificación multi-clase.

Regresión logística multinomial (softmax):

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad 1 \le i \le k$$

El softmax de un vector de k entradas es un vector de k probabilidades:

$$\text{softmax}(z) = \left[\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)}, \ldots, \frac{\exp(z_k)}{\sum_{i=1}^{k} \exp(z_i)} \right]$$

Extensión de la LR a clasificación multi-clase

LR multinomial: útiles para clasificación multi-clase.

Regresión logística multinomial (softmax):

$$\mathrm{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad 1 \leq i \leq k$$

El softmax de un vector de k entradas es un vector de k probabilidades:

$$\mathrm{softmax}(z) = \left[\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)}, \ldots, \frac{\exp(z_k)}{\sum_{i=1}^{k} \exp(z_i)} \right]$$

En LR multinomial, la probabilidad de la clase condicionada a la instancia x se expresa según:

$$p(y = c|x) = \frac{\exp(w_c \cdot x + b_c)}{\sum_{j=1}^{k} \exp(w_j \cdot x + b_j)}$$

Extensión de la LR a clasificación multi-clase

LR multinomial: la función de pérdida se generaliza a k términos.

$$
\begin{aligned}
L_{\mathrm{CE}}(\hat{y}, y) &= -\sum_{k=1}^{K} y_k \log \hat{y}_k \\
&= -\sum_{k=1}^{K} y_k \log \hat{p}(y = k|x)
\end{aligned}
$$

Extensión de la LR a clasificación multi-clase

LR multinomial: la función de pérdida se generaliza a k términos.

$$\begin{aligned} L_{\mathrm{CE}}(\hat{y},y) &= -\sum_{k=1}^{K} y_k \log \hat{y}_k \\ &= -\sum_{k=1}^{K} y_k \log \hat{p}(y=k|x) \end{aligned}$$

Si la clasificación no es multi-etiqueta, el vector y es un **one-hot** vector.
Luego, la función de pérdida se reescribe según:

$$\begin{aligned} L_{\mathrm{CE}}(\hat{y},y) &= -\sum_{k=1}^{K} \mathbb{1}\{y=k\} \log \hat{p}(y=k|x) \\ &= -\sum_{k=1}^{K} \mathbb{1}\{y=k\} \log \frac{\exp(w_k \cdot x + b_k)}{\sum_{j=1}^{K} \exp(w_j \cdot x + b_j)} \end{aligned}$$

Extensión de la LR a clasificación multi-clase

LR multinomial: la función de pérdida se generaliza a k términos.

$$
\begin{aligned}
L_{\mathrm{CE}}(\hat{y},y) &= -\sum_{k=1}^{K} y_k \log \hat{y}_k \\
&= -\sum_{k=1}^{K} y_k \log \hat{p}(y=k|x)
\end{aligned}
$$

Si la clasificación no es multi-etiqueta, el vector y es un **one-hot** vector. Luego, la función de pérdida se reescribe según:

$$
\begin{aligned}
L_{\mathrm{CE}}(\hat{y},y) &= -\sum_{k=1}^{K} \mathbb{1}\{y=k\} \log \hat{p}(y=k|x) \\
&= -\sum_{k=1}^{K} \mathbb{1}\{y=k\} \log \frac{\exp(w_k \cdot x + b_k)}{\sum_{j=1}^{K} \exp(w_j \cdot x + b_j)}
\end{aligned}
$$

Lo que se reduce a la **negative log likelihood**:

$$
\begin{aligned}
L_{\mathrm{CE}}(\hat{y},y) &= -\log \hat{y}_k, \\
&= -\log \frac{\exp(w_k \cdot x + b_k)}{\sum_{j=1}^{K} \exp(w_j \cdot x + b_j)}
\end{aligned}
$$

Clase correcta

LR en Python (sklearn)

```
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB


X_train, X_test, y_train, y_test =
train_test_split(df['Consumer_complaint_narrative'], df['Product'],
random_state = 0)
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)


clf = MultinomialNB().fit(X_train_tfidf, y_train)
```