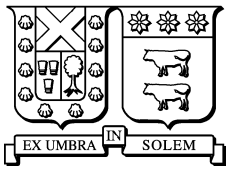
Análisis Inteligente de Datos

Tarea 2

Diego Salazar - Anibal Pérez

24 de Junio de 2016



1. Reducción de dimensionalidad para Clasificación

Se analizaron los datos de un dataset de sonidos fonéticos con 10 características, y una columna de respuesta con las 11 vocales del inglés británico.

a. Se comienza por descargar los datos desde su dirección original, para luego traspasarlos a archivos csv. Con esto es posible utilizarlos como dataframes y hacer el análisis respectivo. Los datos de entrenamiento de este set son 528, y los de pruebas 462. Las palabras son dichas la misma cantidad de veces en los dos datasets, 48 en el de entrenamiento y 42 en el de pruebas.

b. Se normalizan los datos con esperanza nula y varianza unitaria, para eliminar la influencia de outliers en las técnicas de reducción para la clasificación a utilizar.

c. El primer ejercicio de reducción de dimensionalidad se realiza con PCA, sobre las dos componentes principales (de 10) del dataset. Se elige el color de paleta **arcoiris** pues tiene una alta gama de colores, la cual permitirá diferenciar los registros de clases distintas claramente. Como se puede apreciar en la Figura 1, la reducción por componentes principales no parece funcionar para nada bien en la clasificación, dado que los elementos se encuentran dispersos en todo el espacio, esto es causado porque las direcciones principales que este tipo de reducción calcula, es basado en la covarianza de los datos, es decir en sus diferencias en vez de centrarse en su parecido.

Figura 1: Dispersión de elementos en reducción PCA.

d. El segundo ejercicio se realiza con Análisis de discriminante lineal (LDA) también conocido como Fisher Discriminant. Para esta reducción en dos dimensiones, se puede apreciar en la Figura 2 como claramente aporta mejor que PCA a la clasificación de los elementos, dado que están bastante más agrupados que en la imagen anterior. Esto es a causa de que se realiza una regresión lineal sobre la columna de respuesta, entregando mejores resultados para el objetivo esperado que era clasificar.

Figura 2: Dispersión de elementos en reducción LDA.

e. La elección de un método de reducción (ya que la comparación cualitativa de los métodos se realizó en los dos ítems anteriores), se tendría que basar en cuán alejados quedan los

elementos de otros de su misma clase, calculando esta distancia con métodos como distancias de Manhattan, K-means, etc. El que tenga menor distancia interclase, es el que claramente sirve mejor para la clasificación.

f. Se construye un clasificador que cuenta las ocurrencias de cada clase, y las divide por el total. Dado que las 11 vocales se repitieron la misma cantidad de veces, se presenta una probabilidad A PRIORI para cualquier input X, de $1/11 obien 0{,}091$ para cualquier clase.

g. Al utilizar LDA, QDA y modelo de Vecinos más Cercanos (KNN) con K=10 para la clasificación, se puede observar en el Cuadro 1 que el clasificador que mejor se comporta es el de vecinos más cercanos, dado que este entrega la mayor precisión sobre el conjunto de pruebas. El que se comporta mejor sobre el de entranmiento es QDA, con una precisión de casi 99 por ciento, lo que es esperado dado que aprende del mismo set que entrena.

Cuadro 1: Tabla de comparación de precisión para distintos métodos para clasificar.

Técnica utilizada / Tipo Set	**Set de Entrenamiento**	**Set de Pruebas**
LDA	0.6837	0.4524
QDA	0.9886	0.4156
KNN (k=10)	0.9318	0.4918

Al observar la Figura 3, se puede ver el efecto de cambiar los k vecinos para KNN, en el la precisión de este método. Se puede observar como la mayor precisión se presenta en K=8 (el gráfico está corrido en 1 K) para el set de pruebas, solo perdiendo precisión al aumentar los vecinos luego de esta cantidad. El efecto es distinto en el de entrenamiento pues al la mayor precisión se alcanza con 1 vecino, dado que aprende de si mismo como ya se mencionó.

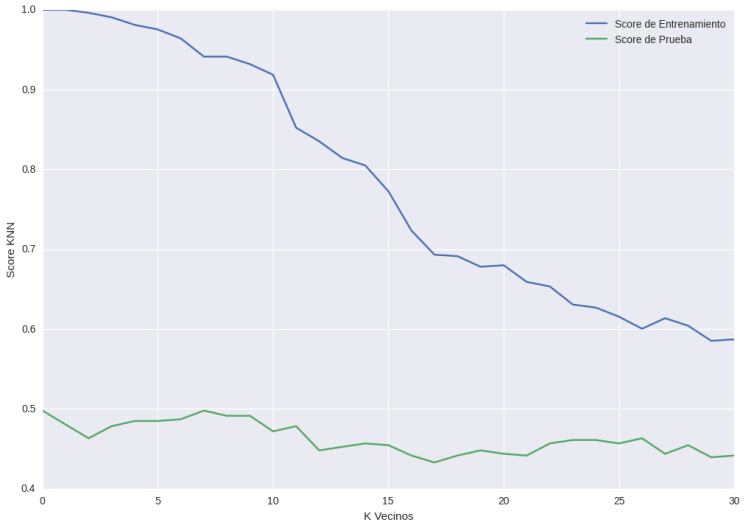


Figura 3: Score para distintos K en reducción de vecinos más cercanos.

h. Luego de reducir con PCA las dimensiones del problema, se compara el entrenamiento a medida que suben estas dimensiones para los métodos LDA, QDA y KNN. En la Figura 4 se puede apreciar la afirmación de lo obtenido en ítems anteriores, pues KNN es quien obtiene

el menor error para el conjunto de pruebas, cuando K está entre 8 y 10, el que corresponde a 1 menos el score ya mostrado, que era 0.4918, lo que se asevera en el gráfico.

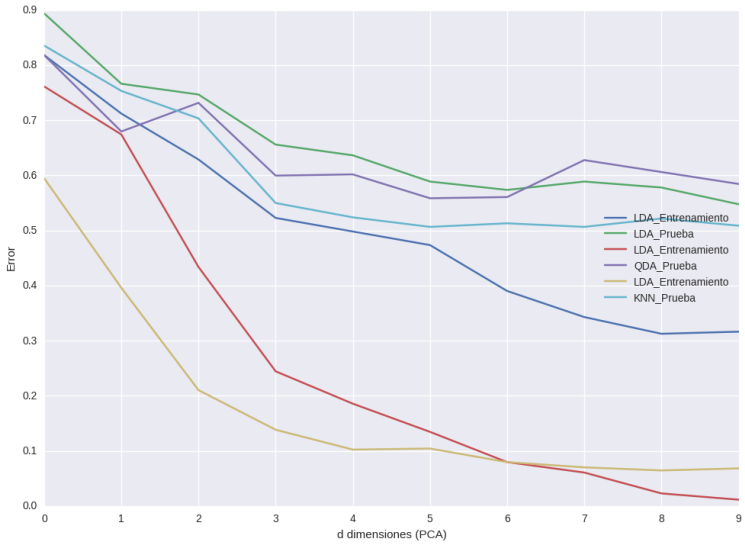


Figura 4: Error de precisión para distintas dimensiones aplicando PCA, ajustado a los métodos LDA, QDA y KNN para los dos conjuntos de datos.

i. Análogo al ítem anterior, pero reduciendo dimensiones con LDA, se puede observar en la Figura 5, que quien se comporta de mejor manera para el set de pruebas sigue siendo KNN, con un score parecido al del ítem anterior, lo que hace concluir que al aumentar las dimensiones, para cualquier reducción (PCA o LDA), el error de prueba para KNN va a converger al valor que se observó en el Cuadro 1.

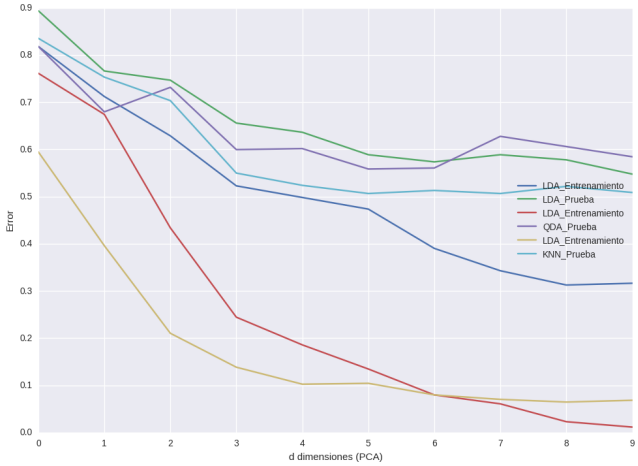


Figura 5: Error de precisión para distintas dimensiones aplicando LDA, ajustado a los métodos LDA, QDA y KNN para los dos conjuntos de datos.

2. Análisis de opiniones sobre películas

a. Se determina la forma de los datasets -ambos contienen dos atributos por entrada correspondientes a una frase (texto) y su respectivo sentimiento (entero 1 o -1). Los datos de entrenamiento poseeen 3554 entradas mientras que los de test, 3554.

Cuadro 2: Número de registros por dataset

Dataset	**Comentarios (+)**	**Comentarios (-)**
Training	1770	1784
Test	1751	1803

b. Se utlizaron las siguientes frases para hacer las pruebas solicitadas:

- *I love to eat cake*
- *I love eating cake*
- *I loved eating the cake*
- *I do not love eating cake*
- *I don't love eating cake*
- *Those are stupid dogs*
- *It wasn't really a baaaad movie*

A estas frases se les hizo pasar por 3 funciones distintas de transformación:

- **Preprocesamiento**: Una combinación de eliminación de *stopwords*, *lowercasing*, y la sustitución de palabras repetidas.
- ***Stemming***: Agrupación de palabras hacia su raíz. Por ejemplo, todas las conjugaciones de un verbo y su gerundio pasan a formar la misma palabra (el verbo raíz).
- ***Lematizado***: Agrupa palabras de acuerdo a su raíz y contexto. Dadas las características de los textos estudiados (breves e informales), agrupan en palabras diferentes el gerundio y la conjugación de un verbo (ej: distingue *eat* de *eating*.

Al analizar las frases transformadas algunas cosas saltan a la vista: el preprocesado elimina el adverbio *not* (por ser *stopword*), lo que puede alterar por completo el sentido de una frase; stemming tiende a hacer agrupaciones fonéticas, así *movie*, *movies* y *movy* pasan a ser *movi*; y -como fue dicho antes- al lematizar la agrupación es menos brusca, distinguiendo *eat* de *eating*.

c. *Respondido en b.*

d. Se realiza el análisis de los dos datasets para 4 tipos de filtro: Stemming, stemming pre-procesado, lematizado y lematizado pre-procesado. Los resultados se resumen en la siguiente tabla.

Cuadro 3: Ranking palabras más utilizadas en training

Rank	**Stemming (train)**	**Stemming proproc (Train)**	**Lemmatizing (Train)**	**Lemmatizing preproc (train)**
1	the	film	the	film
2	and	movi	and	movie
3	of	like	of	one
4	it	one	it	like
5	to	make	to	make
6	is	stori	is	story
7	in	charact	in	character
8	that	time	that	comedy
9	film	even	film	even
10	but	good	but	time

Cuadro 4: Ranking palabras más utilizadas en testing

Rank	**Stemming (test)**	**Stemming proproc (test)**	**Lemmatizing (test)**	**Lemmatizing preproc (test)**
1	the	film	the	film
2	and	movi	and	movie
3	of	one	of	one
4	it	like	it	like
5	to	stori	to	story
6	is	make	is	character
7	that	charact	that	time
8	in	time	in	make
9	film	good	film	comedy
10	but	even	movie	good

Al observar ambas tablas es fácil detectar el impacto que puede llegar a tener la eliminación de *stopwords*. Particularmente, es posible ver como en los datos de test y training aparece la palabra "but.$^{\text{en}}$tre las 10 más repetidas antes de pre-procesar. Esto es relevante ya que a diferencia de las otras *stopwords*, "butçontiene un significado que, en efecto, puede alterar por completo el sentido de una crítica. Así, sin la palabra 'but', una crítica desfavorable puede se tomada como más condescendiente al hacer esta omisión.

e. *No se hacen preguntas, sólo se implementa.*

f. En esta sección se omitirá el detalle de los datos procesados en el resto del ejercicio, dado que la data se somete a 48 modelos predictivos distintos para los que se toman 15 frases aleatorias por cada uno. Para el análisis más en detalle de estos datos, se recomienda ver el *dump* en el archivo `outfile` que se encuentra en el repositorio (o correr personalmente el script `parte2.py`).

Al observar los resultados, salta a la vista que los valores de *recall* y *precision* tienen una diferencia de a lo más $\pm 1\,\%$ para todos los modelos. Esto podría ser producto de que tanto los datos de entrenamiento como los de test están balanceados. Dado esto, se decide utilizar *test accuracy* como indicador de qué tan bueno es un modelo u otro, sin tener que hacer un doble análisis (uno para *recall* y otro para *precision*).

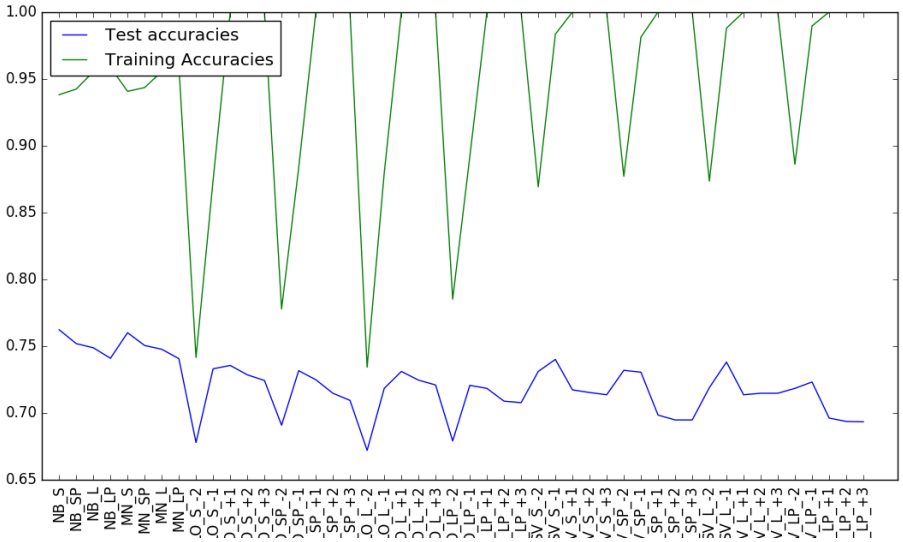


Figura 6: Comparación de *accuracies*

En el gráfico es posible ver que el modelo más efectivo es *Naive Bayes* sin pre-procesado con un *test accuracy* de 76,21 % . Suponemos que esto ocurre dada las ventajas que tiene este modelo por sobre el lematizado para textos breves sin mayor contexto, tal y como fue observado en b. . Referente a los otros modelos (regresión logística y SVM lineal), es posible notar como presentan enormes *overfitting* o *underfitting* para los datos de entrenamiento. Se deduce que al ser modelos sobreajustados o demasiado simples, sencillamente no representan la realidad de la data poblacional, ya que son propensos a sobreestimar errores (en el caso de *overfitting*) o a tener un *accuracy* absurdamente bajo (en el caso de *underfitting*).