DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 1

"Si nos organizamos aprobamos todos..."

Metodos numericos
Primer Cuatrimestre de 2015

Integrante	LU	Correo electrónico
Gastón Zanitti	058/10	gzanitti@gmail.com
Ricardo Colombo	156/08	ricardogcolombo@gmail.com
Dan Zajdband	144/10	Dan.zajdband@gmail.com
Franco Negri	893/13	franconegri200@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (54 11) 4576-3359
http://www.fcen.uba.ar

Índice

1. Introduccion

El objetivo de este trabajo será reconocer imagenes que contienen digitos atravez de la utilizacion de tecnicas simples de machine learning.
La metodologia consistira en la siguiente. Tendremos una base de datos con imagenes ya etiquetadas y una base de datos con imagenes sin etiquetas.
El objetivo se sentrará en que el sistema pueda utilizar la base de datos para poder etiquetar de manera correcta la base de datos sin etiquetar.
Para ello utilizaremos primero el metodo mas intuitivo posible. Esto es para cada imagen a etiquetar, buscamos la que mas se le parezca en la base de datos etiquetada, y decimos que esa es su etiqueta. Por supuesto, todavia queda determiar que es que dos imagenes se 'parezcan'. Eso será descripto en mayor profundidad en el siguiente apartado.
Como segunda idea intuitiva podemos pensar que tal vez tenemos mucha mala suerte y la imagen mas parecida es la de una etiqueta erronea, luego la manera de resolver esto es, en vez de tomar un solo vecino, tomar k vecinos y de esos k vecinos mas cercanos ver que etiqueta es la que mas veces se repite. Estas ideas basicas, es la de KNN que será explicada en mas detalle en el proximo apartado de este trabajo.
Luego,a esta idea, intentaremos aplicarle una mejora sustancial utiliando un metodo probabilistico conocido como PCA que consistira en aplicarle una transformacion a la imagen de tal manera de solo quedarnos con aquello de mayor variabilidad y desechar aquello que pueda estar introduciendo ruido.

2. Desarrollo

2.1. Algortimo de kNN

Como primera aproximación para la resolución del problema de OCR, implementamos el algoritmo de K-vecinos mas cercanos (o kNN por sus siglas en ingles). Este metodo de clasificación consiste basicamente en dado un dato del que no conocemos a que clase pertenece, buscar entre los vecinos los k mas parecidos (habiendo que definir que es ser "parecido"), y luego de estos k vecinos, determinar cual es la moda.
Para nuestro trabajo en particular, tomaremos las imagenes como vectores numericos y definimos que dos imagenes son "parecidas", si la norma dos entre ellas es chica. Luego la idea de el knn será tomar todas las imagenes etiquetadas, compararlas contra la nueva imagen a reconocer, ver cuales son las k imagenes cuya norma es la menor posible y de esos k vecinos ver a que clase pertenecen. La etiqueta para esta imagen vendrá dada por la moda.
//ACA UN PSEUDOCODIGO?
Preliminarmente pensamos en diferentes maneras de mejorar el procesamiento de las imagenes, ya sea pasandolas a blanco y negro para no tener que lidiar con escala de grises o recortar los bordes de las imagenes, ya que en ellos no hay demaciada informacion util (en todas las imagenes vale 0).
Sin embargo, y mas alla de las mejoras que puedan realizarse sobre los datos en crudo, este algoritmo es muy sensible a la variabilidad de los datos. Un conjunto de datos con un poco de dispersion entre las distintas clases de clasificacion, hace empeorar rapidamente los resultados.
En el siguiente apartado pasaremos a describir una metodologia mas sofisticada para resolver este problema que mejore de alguna manera tanto los tiempos de ejecución como la tasa de reconocimiento con respecto a este metodo.

2.2. Optimizacion mediante Análisis de componentes principales

Analisis de Componentes Principales o PCA es un procedimiento probabilistico que utiliza una transformacion ortogonal para convertir un conjunto de variables, posiblemente correlacionadas, en un conjunto de variables linealmente independientes llamadas componentes principales.
Esta transformacion esta definida de tal manera que la primera componente principal tenga la varianza mas grande posible, la segunda componente tenga la segunda varianza mas grande posible y asi sucesivamente.
De esta manera será facil quedarnos con los λ componentes principales que concentren toda la varianza y quitar el resto. En la seccion de experimentacion, uno de los objetivos principales será buscar cual es el λ que concentra la mayor cantidad de varianza de manera tal de optimizar el numero de predicciones.
A fines practicos, lo que haremos, será a partir de nuestra base de datos de elementos etiquetados, construir la matriz de cobarianza M de tal manera que en la coordenada M_{ij} obtenga el valor de la cobarianza del pixel i contra el pixel j.
Luego utilizando el metodo de la potencia, procederemos a calcular λ autovectores de esta matriz. Luego, multiplicamos cada elemento por los λ autovectores y asi obtenemos un nuevo set de datos.
Sobre este set de datos, ahora aplicamos knn nuevamente y lo que esperariamos ver es un mayor numero de aciertos, ya que hemos quitado ruido del set de datos y mejores tiempos de ejecución, ya que hemor reducido la dimencionalidad.
//ACA UN PSEUDOCODIGO?

2.3. Cross-validation

Para medir cuan precisos son nuestros resultados utilizaremos la metodologia de cross-validation. Esto consiste en tomar nuestra base de entrenamiento y dividirla en k bloques. Ahora tomamos el

primer bloque para testear y los demas bloques para entrenar a nuestro modelo y vemos que resultados obtenemos. Luego se toma el segundo bloque para testear y los demas de entrenamiento.
De esta manera evitamos testear contra datos propios del modelo, lo que podria resultar en que el modelo solo reconozca sus propias imagenes de entrenamiento pero no imagenes fuera de el, que es justamente el proposito de este trabajo.

3. Análisis

3.1. KNN

Analisis de KNN

3.1.1. Cantidad de vecinos

Mover el k y presentar un analisis

3.1.2. Mejoras en kNN

Cortar el dataset en x valor y comparar para ver si mejora

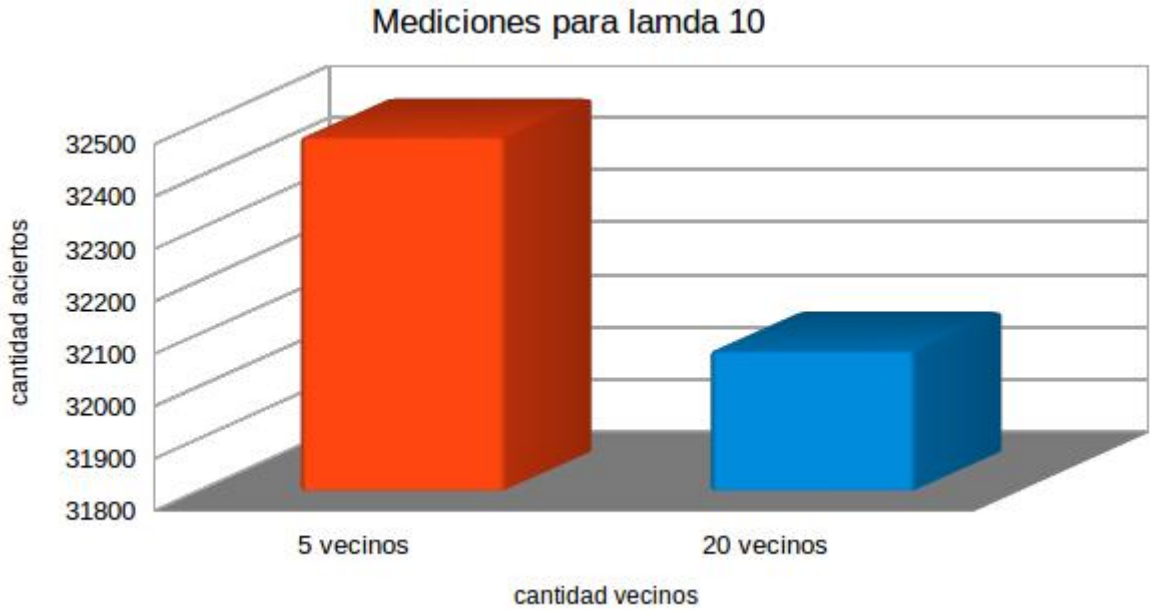
3.2. PCA

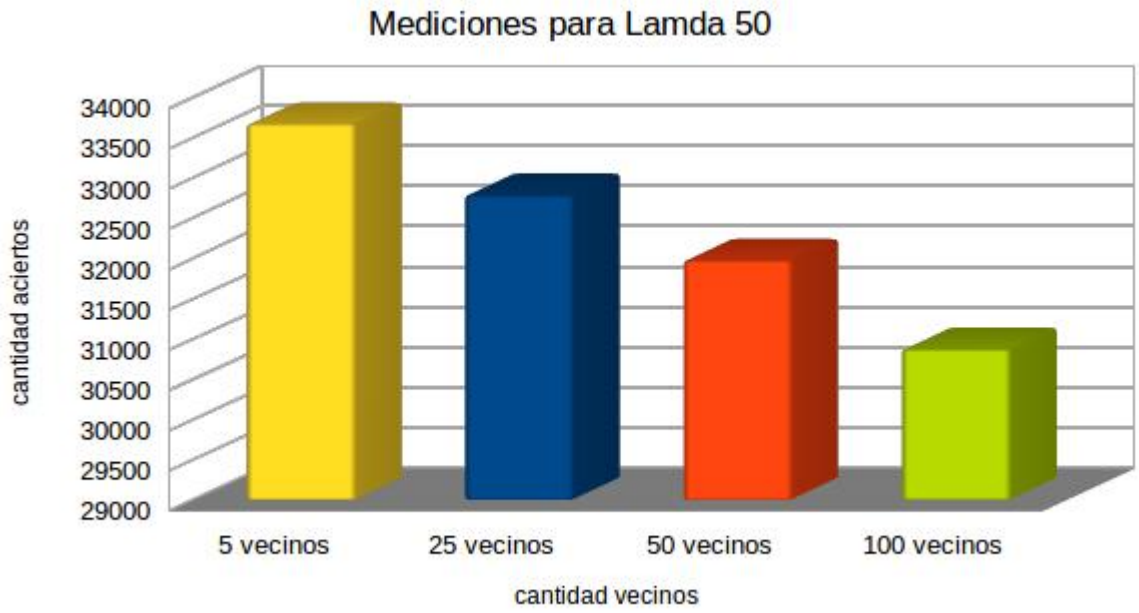
Analisis de PCA Ahora vamos a analizar el algoritmo del PCA. Vamos a probar el algoritmo para distintas medidas de k y α, que van a ser: k: cantidad de vecinos a considerar en el algoritmo kNN. α: a la cantidad de componentes principales a tomar.

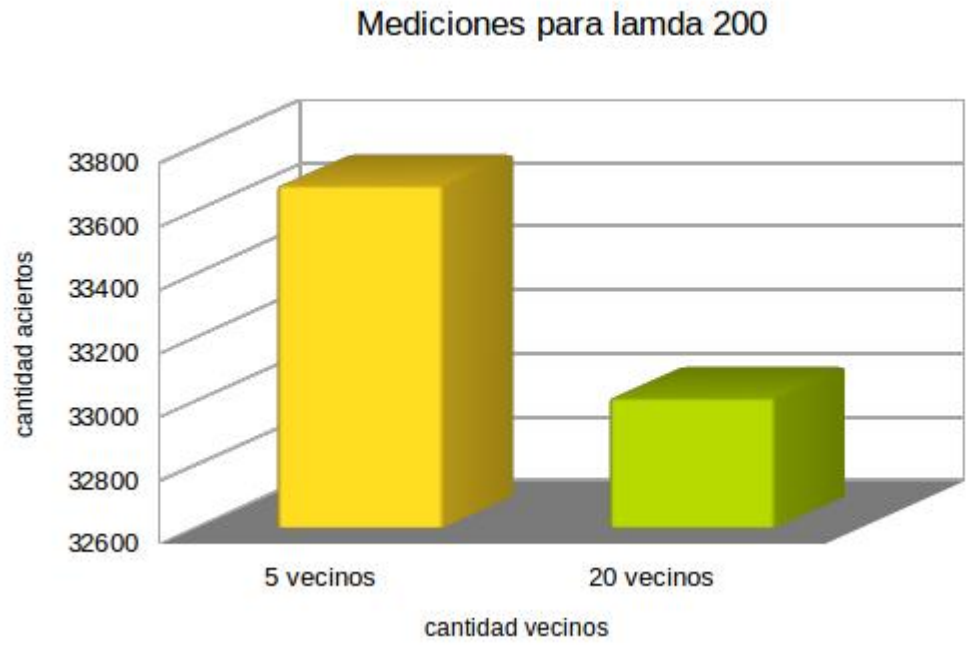
Vamos a probar el algoritmo para los siguientes valores: k: α:

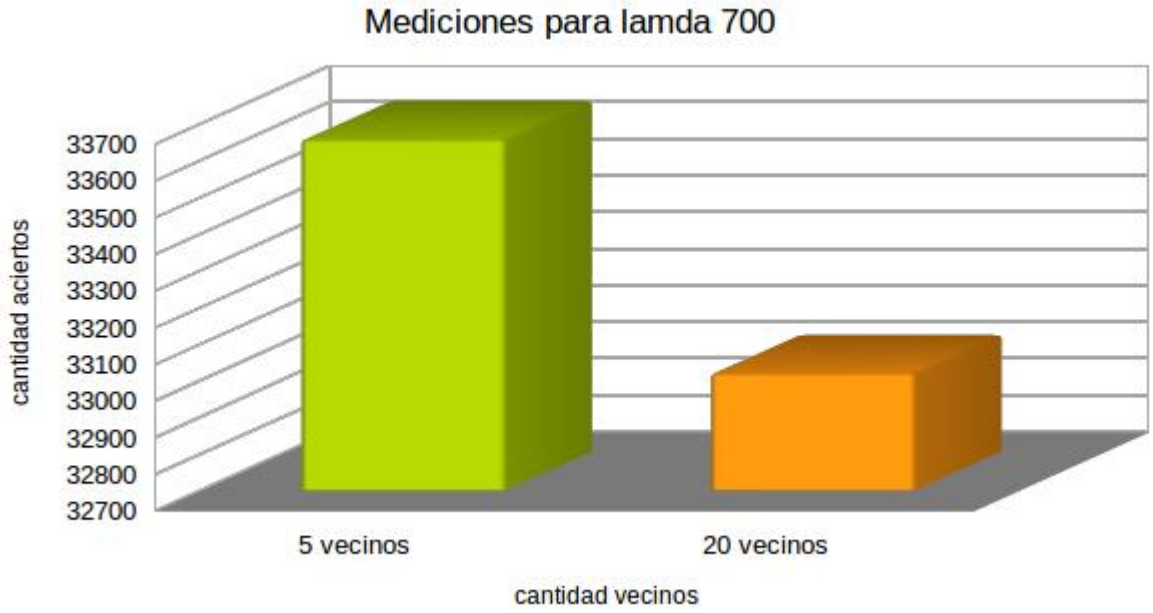
Lo que vamos a probar es fijando un valor de lamda, para que cantidad de vecinos vamos a tener la mayor cantidad de aciertos, y asi, maximizar la cantidad de aciertos. Nosotros despues de aplicar el algoritmo pca, aplicamos el knn y armamos una cola de prioridad para los resultados de aplicar el algoritmo knn. Lo que se hace es agarrar dos imagenes, restarlas y aplicarle la norma2 para saber en cuanto difieren una imagen y la otra. El la cola de prioridad, estan adelante los valores mas chicos, o sea, las imagenes del test que mas cerca de coincidir estan con respecto a la imagen de la base de datos. Por lo tanto, si elegimos mas cantidad de vecinos, lo que nos puede pasar son dos cosas: 1) Que sea beneficioso ya que a mayor cantidad de pruebas vamos a tener mas aciertos 2) Que sea malicioso ya que a mayor cantidad de pruebas vamos a obtener peores datos, o sea, vamos a mirar las imagenes que menos coiciden con la imagen de prueba de la base de datos.

La conclusion que llegamos es que a mayor cantidad de vecinos, el algoritmo empieza a funcionar peor, ya que estamos mirando los vecinos que menos coinciden con la base de datos, porque la cola de prioridad los ordena segun menos diferencias haya entre la foto obtenida y las fotos de la base de datos. Por lo tanto, cuanto mas vecinos exploro, menos cantidad de aciertos obtengo. Eso lo puedo ver a traves de las siguientes mediciones realizadas para distintos k vecinos y fijando un valor de lamda.


Mediciones para lamda 10
cantidad aciertos
32500
32400
32300
32200
32100
32000
31900
31800
5 vecinos
20 vecinos
cantidad vecinos


Mediciones para Lamda 50
cantidad aciertos
34000
33500
33000
32500
32000
31500
31000
30500
30000
29500
29000
5 vecinos
25 vecinos
50 vecinos
100 vecinos
cantidad vecinos





Otra conclusion que sacamos es que para k=5 vecinos es la mejor cantidad de vecinos para obtener la mayor cantidad de aciertos. Ahora, ¿No seria mejor agarrar solo el primero de la cola de prioridad, o sea, k=1? Lo que podria pasar es que el unico vecino que agarremos, sea el mejor pero no alcance para saber cual es el digito de la imagen, ya que si no se acierta con el unico vecino que elegi, me podria fallar el digito del resultado y eso reduciria muchisimo la cantidad de aciertos.

3.2.1. Lambda inicial

3.2.2. Algo mas que no me acuerde

4. Resultados

4.1.

5. Conclusiones

5.1. Aca tiramos tiros

6. Apendice