**Máster de Visión Artificial**

**Asignatura: Reconocimiento de Patrones**

**Práctica 2: Reducción de dimensionalidad**

**Orión García Gallardo**

**DNI: 48330747L**

INTRODUCCIÓN

Esta práctica tiene como objetivos principales:

1. Introducir al alumno en el problema de la clasificación de señales de tráfico en imágenes.
2. Aprender a utilizar la técnica de reducción de dimensionalidad supervisada: Análisis Discriminante Lineal (LDA).

Para alcanzar estos objetivos en práctica se ha realizado una serie de operaciones sobre un conjunto de imágenes de señales aportadas por la asignatura. Dichas operaciones han consistido en:

1. Programación del Análisis Discriminante Lineal (LDA) y cálculo de la matriz de proyección LDA, A3x400.
2. Comprobación de la calidad del LDA mediante validación cruzada de 10 grupos (10 fold).
3. Construcción de un clasificador basado en niveles de gris con 4 clases en el espacio de características de 3 dimensiones encontrado mediante LDA en el paso anterior.
4. Exploración de la solución basándonos en PCA+LDA.

MÉTODO DESARROLLADO

Para la realización de todos los puntos explicados en el apartado anterior se implementa previamente una lectura de las imágenes en blanco y negro, y se reducen las imágenes a dimensión 25x25 para tener menos datos a computar.

En el primero de los puntos, que consiste en el cálculo del LDA, se ha hecho uso de las trasparencias usadas en clase, donde se describía el algoritmo en pseudocódigo (LDA.m).

En el segundo punto se realiza una validación cruzada seleccionando, de todos los datos proporcionados, 9 grupos de entrenamiento y 1 para test (checkLDAQuality.m). Con los datos de entrenamiento se calcula la matriz de proyección. Por otro lado los datos de test se proyectan con dicha matriz dándonos como resultado un conjunto de datos con sólo 3 carácterísticas. Para comprobar la calidad del LDA se muestran en el espacio 3D estos datos proyectados y se calcula como de singular es la matriz St (checkSW.m).

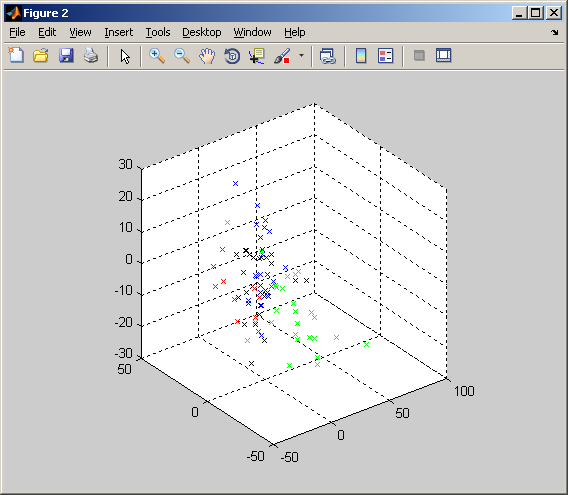
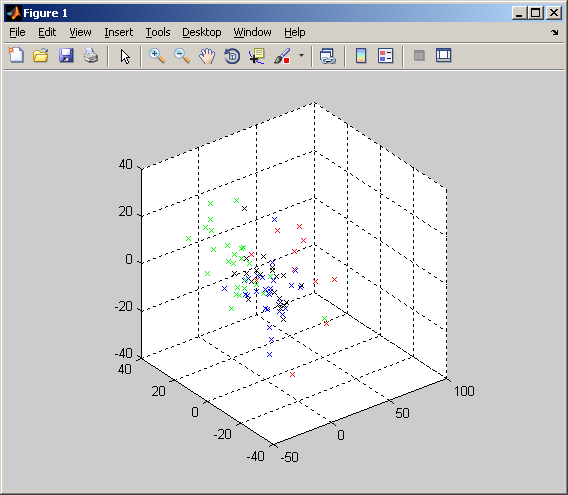
La realización del tercer paso es relativamente sencilla habiendo realizado los dos pasos anteriores (TestLDA.m). Para la realización de la clasificación se ha elegido el algoritmo K-NN. La validación cruzada, además de darnos el valor de eficiencia medio, ha tenido el objetivo de buscar ese K más óptimo. Para ello por cada k seleccionado se ha realizado una división de los datos en 10 grupos, seleccionando uno de ellos como conjunto de test en cada iteración. Como se explicará en el apartado de evaluación, se ha visto que los valores obtenidos dependen de la elección aleatoria de los datos a la hora de crear los grupos. Es por ello, por lo que, con el objetivo de obtener unos valores medios no dependientes de la ejecución se ha realizado estas validaciones cruzadas un número de veces determinado.

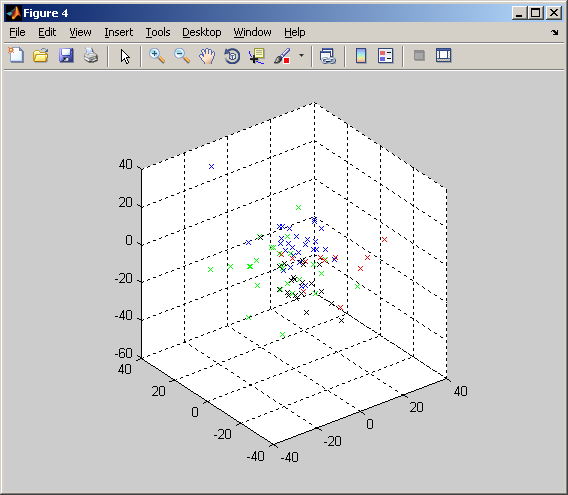
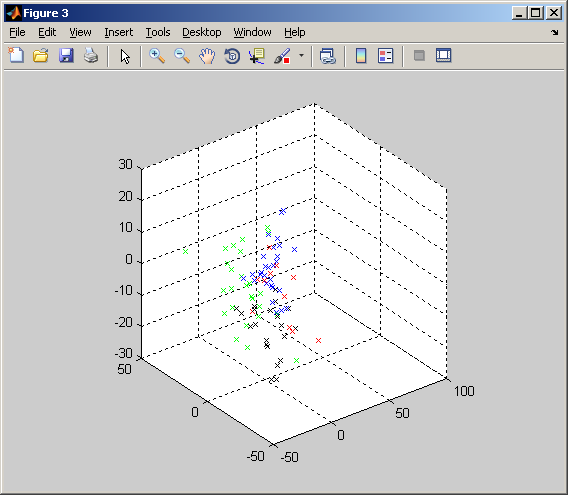
Por último el desarrollo del cuarto paso ha consistido en realizar una clasificación de los datos usando PCA y posteriormente, una vez reducido el espacio de características, aplicando LDA (TestPCA.m). La elección de conjunto de autovectores del PCA se realiza mediante validación cruzada de 10 grupos. Por otro lado el parámetro del clasificador (en este caso k por haber seleccionado el algoritmo K-NN) también habrá de ser evaluado en esta validación cruzada. Por lo tanto en este paso en la validación cruzada se buscará la tupla conjunto de autovectores, valor k que mejor clasifique los datos. Para la implementación de PDA se han seguidos los pasos del pseudocódigo explicado en clase (PCA.m).

EVALUACIÓN

Para realizar la evaluación de los algoritmos se han hecho pruebas con las imágenes aportadas en la práctica. Dichas imágenes se dividen en 4 clases: ceda al paso, peligro, prohibición y stop. Cada clase cuenta con 111, 259, 277 y 205 imágenes respectivamente.

La primera prueba que se realiza es la de comprobar la calidad del LDA mediante validación cruzada de 10 grupos (10 fold). Se dividen las imágenes de cada tipo de señal de tráfico en 10 grupos (con igual número de imágenes) y se usan 9 grupos para calcular el LDA. El grupo restante se usa para proyectarlo y visualizar el resultado. Por tanto haremos y visualizaremos las proyecciones LDA de cada uno de los 10 grupos de prueba (usando los restantes 9 para calcular el LDA). A continuación se muestran algunas de los resultados de las pruebas.



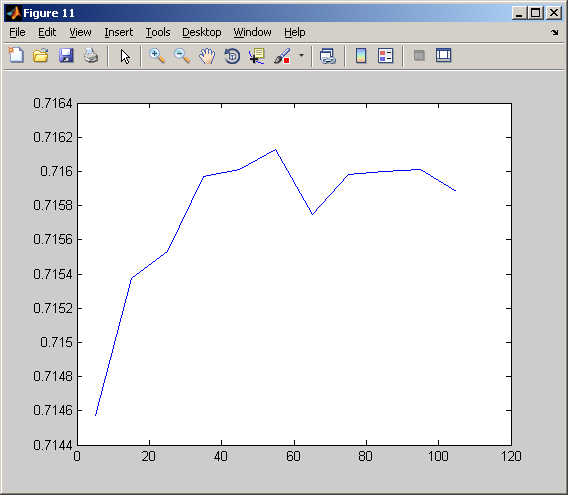


En principio podría parecer que la clases si se encuentran lo suficientemente separadas. Sin embargo, como se verá a continuación en la siguiente prueba y evaluando este resultado de forma numérica el LDA entrelaza bastante los datos. Para evaluar la calidad del LDA de forma numérica calculamos la dispersión total de los datos como St = Sw + Sb. Si St es singular (deficiente en rango), no se puede encontrar la solución. Para averiguar la singularidad de St se usa la funcion rcond de matlab. Esta función nos devolverá un valor entre 0 y 1. Si el resultado es un número cercano a 0 se podría decir que St es singular.

|  |  |
| --- | --- |
| NFOLD | rcond(St) |
| 1 | 0,262806 |
| 2 | 0,734058 |
| 3 | 0,310616 |
| 4 | 0,509016 |
| 5 | 0,46502 |
| 6 | 0,531942 |
| 7 | 0,769916 |
| 8 | 0,522123 |
| 9 | 0,44472 |
| 10 | 0,519314 |

El valor medio de rcond(St) es 0.5070. Es decir que St no es singular, pero tampoco tiene una dispersión total muy buena. Esto puede verse en la prueba siguiente donde se evalúa la clasificación de los datos tras realizar un LDA.

Para comprobar la eficiencia del LDA a continuación se muestran los resultados al evaluar la clasificación de las imágenes realizando validación cruzada con 10 grupos (10 folds). Como se ha visto que los valores obtenidos dependen de la elección aleatoria de los datos y con el objetivo de obtener unos valores medios no dependientes de la ejecución se ha realizado esta clasificación un número de veces determinado, que se ha establecido en 10. Como algoritmo de clasificación se ha elegido el K-NN. Los valores de K seleccionados para hacer la evolución han sido kvalues = 5:10:105. Como puede verse en la siguiente gráfica como valor medio de eficiencia de clasificación se obtiene cerca del 0.7157 sobre 1. También se puede observar que el mejor resultado se encuentra con K = 55 con porcentaje de acierto entorno al 0.7161. Dicha eficiencia se encuentra un poco alejada de lo que consideraríamos unos valores óptimos de eficiencia.

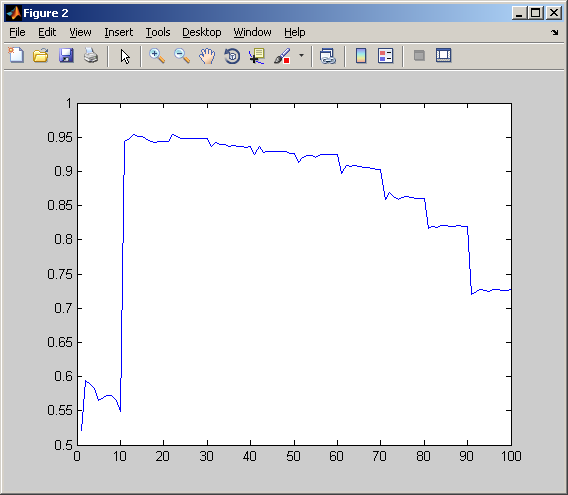


El tiempo empleado en ejecutar esta prueba es de 1775.939046 segundos, es decir, entorno a los 30 minutos (unos 3 minutos por iteración).

A continuación se explora la solución aportada en clase basada en PCA+LDA. El objetivo de esta evaluación en este caso es encontrar la tupla número de características, número de vecinos más cercanos (d, k) donde el rendimiento de clasificación sea más alto.

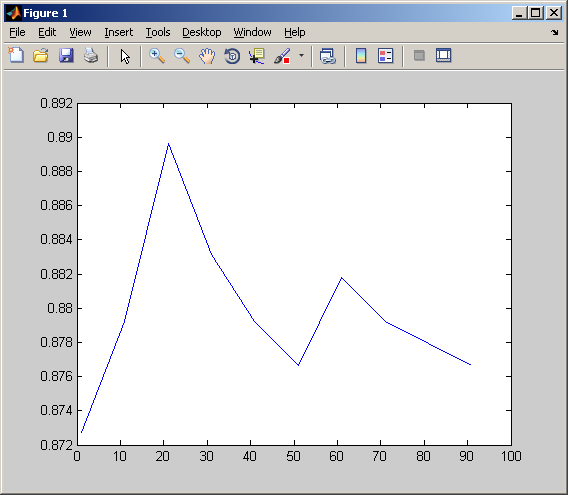
Como primera prueba se cogen valores posibles de d en todo el espacio de características, es decir, entre 4 y 624. Se escoge 4 (el número de clases) porque el LDA no puede hacer mayor reducción de características. Para acelerar la prueba se coge un incremento de estos valores de 68 unidades. Esto significa que los valores de d serán {4, 72, 140, 208, 276, 344, 412, 480, 548, 616}. Por otro lado, los valores iniciales escogidos para k serán 1:10:100.

El tiempo empleado en ejecutar esta prueba es de 368.28 segundos.

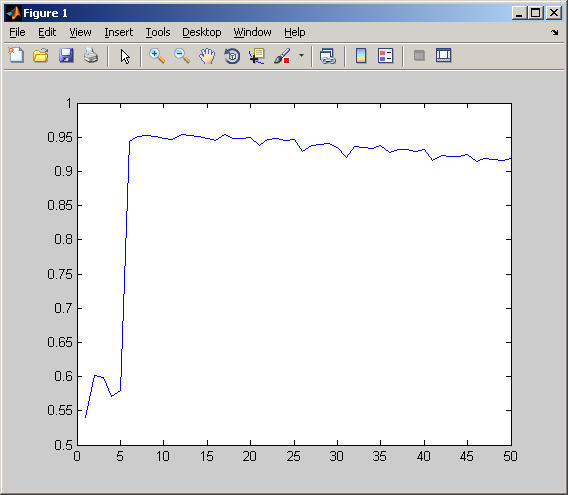


|  |  |  |  |
| --- | --- | --- | --- |
| Nº Features | K | % Test | x axis |
| 72 | 21 | 0,9545 | 13 |
| 140 | 11 | 0,9530 | 22 |
| 140 | 21 | 0,9510 | 23 |
| 72 | 31 | 0,9509 | 14 |
| 72 | 41 | 0,9509 | 15 |
| 140 | 31 | 0,9480 | 24 |
| 140 | 71 | 0,9480 | 28 |
| 140 | 81 | 0,9480 | 29 |
| 140 | 91 | 0,9480 | 30 |
| 72 | 11 | 0,9484 | 12 |

Eficiencia media de los valores de k en el algoritmo k-nn.



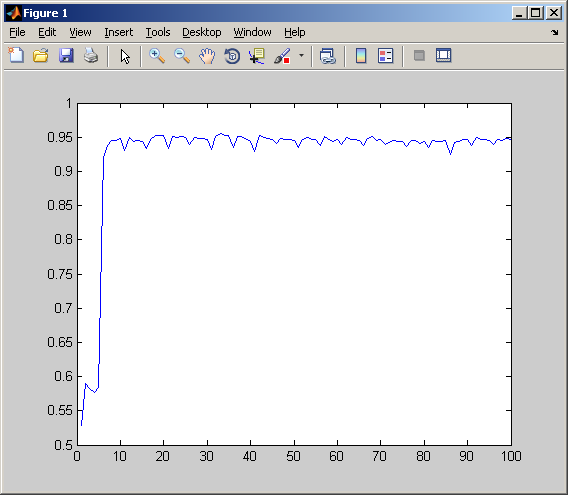
Se puede observar que el mejor rendimiento se obtiene de la mitad de la gráfica hacia la izquierda, es por ello por lo que se realiza una nueva evaluación de resultados pero cogiendo como máximo la mitad de las características, esto es hasta 312 (en matlab size(datos, 2)-1)/2). Por otro lado también se va a acotar los valores de k posibles a {1, 11, 21, 31, 41} ya que como puede verse en la gráfica con valores mayores el rendimiento decae.



|  |  |  |  |
| --- | --- | --- | --- |
| Nº Features | K | % Test | x axis |
| 106 | 11 | 0,9542 | 17 |
| 72 | 11 | 0,9541 | 12 |
| 72 | 21 | 0,9529 | 13 |
| 38 | 21 | 0,9518 | 8 |
| 38 | 11 | 0,9508 | 7 |
| 38 | 31 | 0,9506 | 9 |
| 72 | 31 | 0,9506 | 14 |
| 106 | 41 | 0,9496 | 20 |
| 106 | 21 | 0,9484 | 18 |
| 106 | 31 | 0,9484 | 19 |

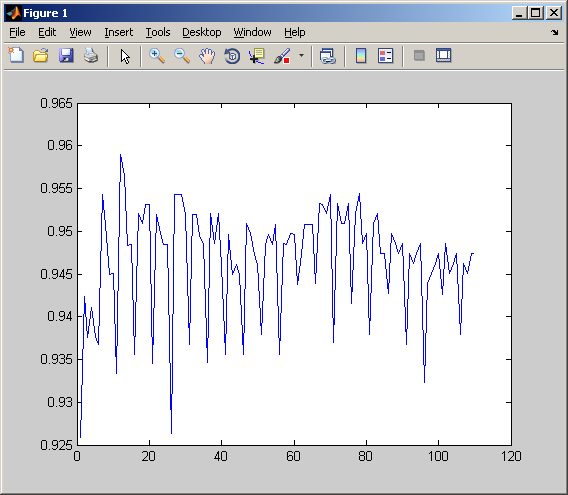
El tiempo empleado en ejecutar esta prueba es de 57.777067 segundos. Se puede observar que este tiempo es bastante menos de la mitad del anterior. A su vez el mejor rendimiento se vuelve a obtener en la mitad izquierda de la gráfica.

La siguiente prueba se realiza con hasta 156 características (en matlab size(datos, 2)-1)/4). A su vez se aprovecha para disminuir el intervalo a la hora de escoger las características. Se elige iterar cada 8 características (en matlab floor((featuresEnd-CLASSNUMBER)/19))

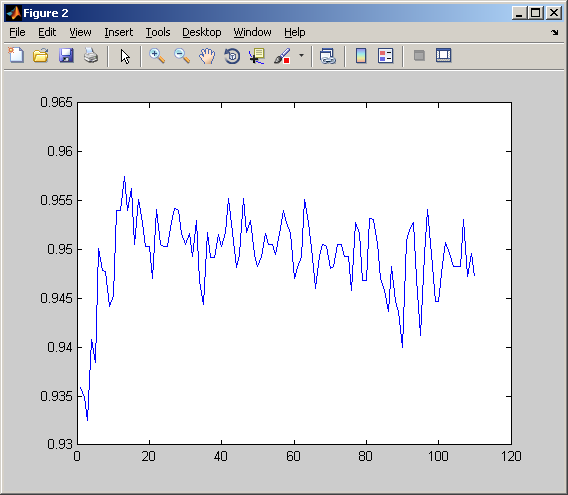


|  |  |  |  |
| --- | --- | --- | --- |
| Nº Features | K | % Test | x axis |
| 52 | 21 | 0,9555 | 33 |
| 28 | 31 | 0,9529 | 19 |
| 28 | 41 | 0,9529 | 20 |
| 28 | 21 | 0,9528 | 18 |
| 52 | 31 | 0,9519 | 34 |
| 68 | 11 | 0,9519 | 42 |
| 52 | 41 | 0,9519 | 35 |
| 60 | 21 | 0,9509 | 38 |
| 92 | 11 | 0,9509 | 57 |
| 36 | 31 | 0,9508 | 24 |
| 52 | 11 | 0,9508 | 32 |
| 60 | 11 | 0,9508 | 37 |
| 108 | 21 | 0,9508 | 68 |
| 36 | 11 | 0,9506 | 22 |
| 68 | 21 | 0,9496 | 43 |

El tiempo empleado en ejecutar esta prueba es de 45.428378 segundos. Se vuelve a observar que este tiempo es menor que en el caso anterior pese a estar probando con más características. El factor determinante es que se prueba con dimensiones de características menores aunque se hagan más iteraciones. Sin embargo, seguimos observando que entre los 15 mejores resultados sigue sin haber ningún valor que se encuentre en el 25% de valores más altos de número características. Esto hace que se opte por elegir menos características, llegando al 20% de las características iniciales (o en matlab sería featuresEnd = round((size(datos, 2)-1)/5)). A su vez se puede observar en las gráficas que los valores de la izquierda aportan resultados por debajo del 60% de acierto. Con el objetivo de acelerar la ejecución se aprovecha para subir el número de características iniciales a 16 (en matlab featuresStart = CLASSNUMBER + round(((size(X, 2)-1)/5)/10))

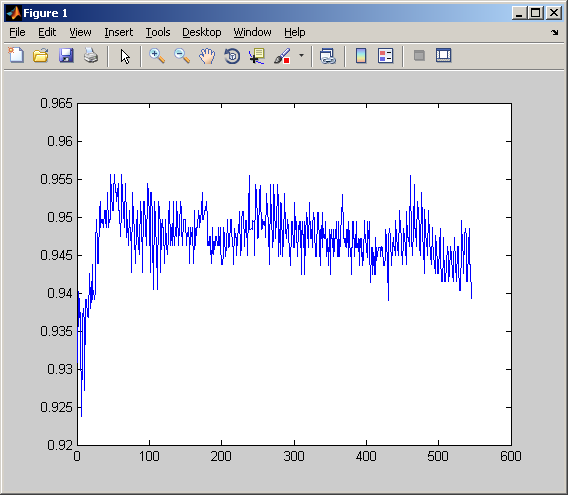
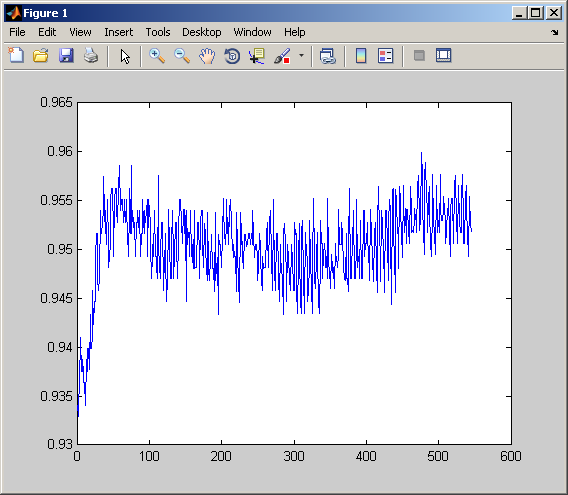
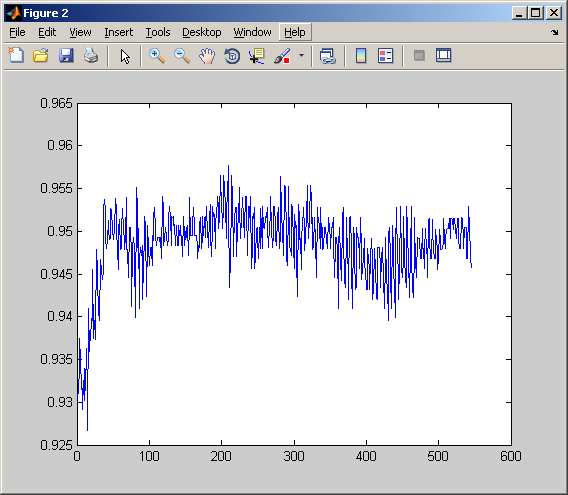
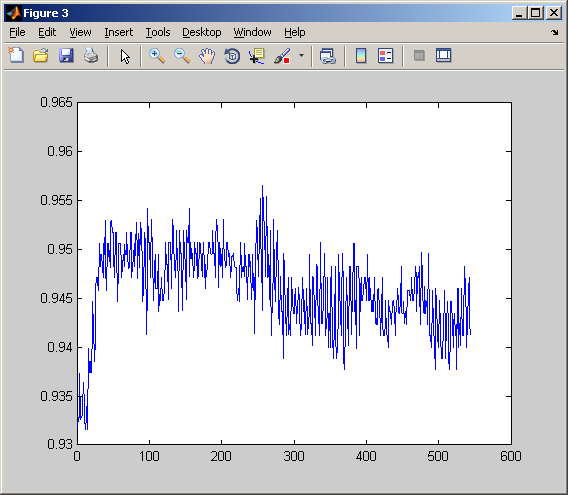


El tiempo empleado en ejecutar esta prueba es de 42.580749 segundos. Se puede observar en la gráfica que hay mucho mínimos relativos y estos mínimos, mirando toda la tabla de resultados que no se muestra en este documento, se corresponden con una clasificación del vecino más próximo, es decir, cuando k=1. Es por ello por lo que se opta por elegir otros valores de k. Para posteriores pruebas se eligen como posibles valores de k {5, 15, 25, 35, 45}.



El tiempo empleado en ejecutar esta prueba es de 43.320901 segundos, un tiempo parecido al de la prueba previa. La gráfica nos muestra valores de porcentaje de acierto más estables que anteriormente.

Con las pruebas anteriores se puede decir que se dispone de un espacio de características donde se puede centrar más exhaustivamente la búsqueda de la mejor tupla (d,k). A continuación para buscar esa tupla se realizan 4 experimentos donde los posibles valores de d irían de 16 a 124 de uno en uno y los valores de k de 5 a 45 de diez en diez.

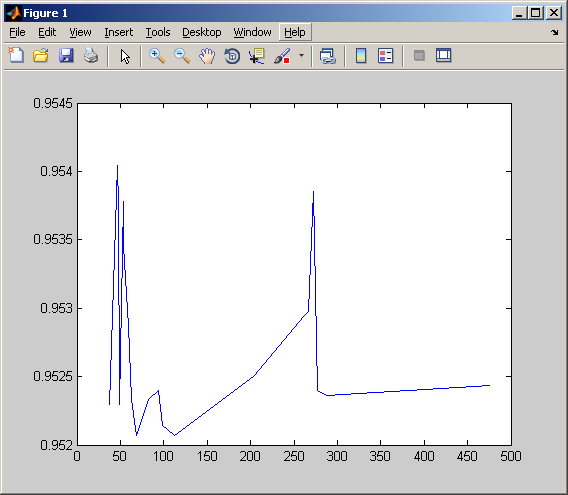
   

A continuación se muestra el resumen tras 4 ejecuciones:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nº Features | K | 1º Test | 2º Test | 3º Test | 4º Test | Mean | Var | x axis |
| 25 | 15 | 0,955542 | 0,955134 | 0,952673 | 0,952809 | 0,95404 | 2,2796E-06 | 47 |
| 70 | 15 | 0,954243 | 0,955134 | 0,952945 | 0,953081 | 0,953851 | 1,0719E-06 | 272 |
| 26 | 25 | 0,953217 | 0,956297 | 0,953836 | 0,951782 | 0,953783 | 3,5496E-06 | 53 |
| 25 | 25 | 0,953217 | 0,956297 | 0,952537 | 0,952945 | 0,953749 | 2,9641E-06 | 48 |
| 26 | 35 | 0,953217 | 0,956297 | 0,952537 | 0,951782 | 0,953458 | 3,9255E-06 | 54 |
| 69 | 15 | 0,954243 | 0,953972 | 0,951782 | 0,951918 | 0,952979 | 1,7143E-06 | 267 |
| 68 | 15 | 0,953081 | 0,951646 | 0,951646 | 0,955406 | 0,952945 | 3,1502E-06 | 262 |
| 27 | 35 | 0,950891 | 0,958623 | 0,951374 | 0,950619 | 0,952877 | 1,4771E-05 | 59 |
| 56 | 25 | 0,948565 | 0,954108 | 0,956569 | 0,950755 | 0,952499 | 1,2555E-05 | 203 |
| 111 | 5 | 0,953217 | 0,959921 | 0,946859 | 0,949728 | 0,952431 | 3,1692E-05 | 476 |

El tiempo medio empleado en ejecutar cada una de estas pruebas es de 179,6676 segundos. Como puede observarse tanto en las gráficas como en la tabla de resultados hay tres zonas diferenciadas con resultados bastante eficientes. La primera, la de la izquierda en las gráficas (valores del eje de las x entre 45 y 60), con resultados más estables y que se correspondería con seleccionar en el PCA entre 25 y 30 número de características. La segunda, más central en las gráficas (valores del eje de las x entre 200 y 275), con resultados algo más inestables que en la zona anterior, pero obteniendo buenas medias. Y la última zona, la de más a la derecha, (valores del eje de las x entorno al 500), con resultados muy inestables y que, por lo general, obtienen peores medias. Ejemplo de ello se puede observar en la última línea de la tabla de resultados, donde en la segunda prueba llega a obtenerse el mejor rendimiento de todas las ejecuciones (0,959921). Sin embargo, debido a los malos resultado obtenidos en las otras pruebas, puede observarse que tiene la mayor varianza de los diez, su media desciende hasta la décima posición. Por otro lado es importante destacar que entre los posibles valores de k los que dan mejor rendimiento a la clasificación son aquellos que van de 15 a 35 vecinos.

A continuación se muestra una gráfica resumen de las 4 ejecuciones, con la medias de los 20 mejores resultados:



CONCLUSIONES

Como puede observarse de la evaluación para este caso una clasificación después de realizar una reducción de dimensionalidad usando sólo LDA no es aceptable ya que se óptenla como mucho un 0,7161 de tasa de acierto. Es por ello, por lo que es recomendable hacer una reducción de dimensionalidad previa con el algoritmo PCA. Cuando esto se realiza podemos ver como los porcentajes de tasa de acierto suben de 0,9 sobre 1 en más de la mitad de los casos. En cuanto al PCA, se puede observar que la mayoría de la información discriminante la poseen entorno a un 20% de las características, en nuestro caso de los pixeles. Esto puede verse en la última evaluación donde seleccionando tan sólo con 25 de las 625 características y con los 15 vecinos más cercanos se obtiene el mejor rendimiento de clasificación.