Aprendizaje automático

Práctica 6: Análisis de componentes principales (PCA)

GAJAN Antoine (894825)

Escuela de Ingeniería y Arquitectura
Universidad Zaragoza

Índice

Introducción

Como parte de la asignatura “Aprendizaje automático", los estudiantes se familiarizarán con los conceptos de inteligencia artificial. Más concretamente, durante el cuatrimestre se estudiarán las bases del aprendizaje supervisado y no supervisado.

Durante esta sexta práctica, vamos a poner en práctica los conocimientos aportados sobre el PCA en las clases magistrales. En particular, vamos a entender cómo funciona este algoritmo y las diferentes maneras de implementarlo con el lenguaje Matlab. Además, los resultados de esta práctica se compararán a los de las prácticas anteriores para comprender las ventajas y desventajas de la técnica PCA.

Esta memoria pondrá de relieve el proceso emprendido para responder a las preguntas, y producirá una reflexión personal sobre los resultados obtenidos.

1. Estudio previo

En primer lugar, tuvimos que estudiar los algoritmos del curso para comprender cómo funcionaban y luego implementarlos.

Una vez dominadas estas nociones, revisé mis conocimientos matemáticos relativos a los valores propios y vectores propios, que se utilizan en los algoritmos PCA para optimizarlos.

Así que aquí vamos a proponer los algoritmos PCA. El primero consiste en indicar el número de dimensiones k deseado para realizar el estudio, mientras que el segundo buscará el valor ideal de k para alcanzar un determinado umbral de variabilidad.

Algorithm 1 PCA_reduccion(X_norm, k)

{Calculo de la covarianza muestral de los datos normalizados}

$sig \leftarrow covarianza\_muestral(X\_norm)$
{Calculo valores y vectores propios}

$[U, lambda] \leftarrow eig(sig)$

$Uk \leftarrow U(:, 1:k)$
{Reduccion de dimesion de los datos normalizados}

$Z \leftarrow X\_norm * Uk$
return Z

Algorithm 2 PCA_variabilidad(X_norm, variabilidad)

{Calculo de la covarianza muestral de los datos normalizados}

$sig \leftarrow covarianza\_muestral(X\_norm)$
{Calculo valores y vectores propios}

$[U, lambda] \leftarrow eig(sig)$
{Eleccion de k}

$k \leftarrow elegir\_K(lambda, variabilidad)$

$Uk \leftarrow U(:, 1:k)$
{Reduccion de dimesion de los datos normalizados}

$Z \leftarrow X\_norm * Uk$
return Z

Las distintas funciones auxiliares que permiten elegir K o calcular la covarianza de los datos en el curso magistral no presentan particularidad. Por lo tanto, no se detallan aquí, pero están presentes en el código Matlab adjunto a este informe.

Antes de comenzar esta práctica, vamos a estandarizar los datos. En efecto, para que la técnica PCA funcione, es necesario que los datos sean del mismo orden de

magnitud, para evitar que algunas variables tengan una influencia mayor que otras. Por lo tanto, en el desarrollo de la práctica a continuación, nos aseguraremos de trabajar con datos estandarizados.

2. Memoria de la práctica

2.1. Cuestión 1: Reducción de la dimensión en MNIST

En la declaración se nos indica que vamos a utilizar el mismo conjunto de datos que en las prácticas anteriores, a saber, el dataset MNIST. Para verificar la pertinencia de la reducción de dimensión propuesta, utilizaremos el clasificador proporcionado en la práctica 5 y utilizaremos las métricas adecuadas procesadas hasta ahora para evaluar los errores del modelo. En esta primera parte, evaluaremos el uso de PCA para la reconstrucción de dígitos y su clasificación con 2 métodos: la reducción de dimensión y el mantenimiento de la variabilidad.

2.1.1. PCA para reducir las dimensiones de los datos

Con 2 dimensiones Después de haber codificado en Matlab todos los algoritmos necesarios para el buen desarrollo de esta práctica, al principio quise evaluar como evocado en clase magistral los resultados que se obtendrían si se optaba por conservar solo 2 dimensiones.

Comenzamos llamando al algoritmo PCA escrito arriba para obtener el vector Z de dimensiones 4000 * 2. Gracias a este vector Z obtenido, ahora podemos visualizar los datos en 2 dimensiones usando el código proporcionado. Las dos dimensiones corresponden a los çomponentes principales", es decir, los dos componentes que abarcan el máximo de la varianza, permitiendo la pérdida mínima de información.

Así obtenemos:

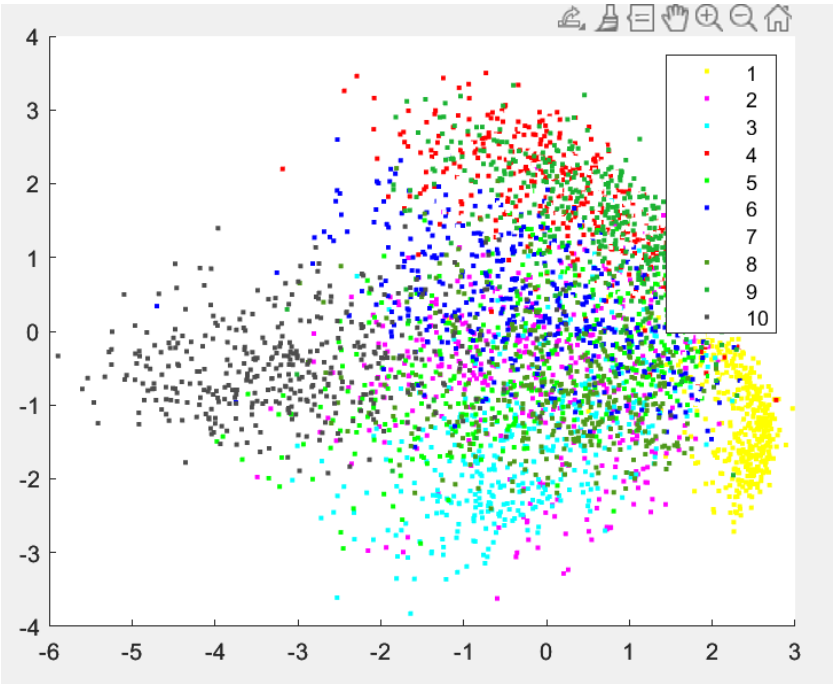

Figura 1: Representación de datos MNIST en 2 dimensiones

Podemos observar por estos datos que algunas clases serán más fáciles de aprender que otras más adelante con la clasificación Bayesiana. Por ejemplo, observamos que las clases 1 y 10 (asociadas a los números 1 y 0, respectivamente) están separadas de las demás clases. Como resultado, el algoritmo de aprendizaje supervisado de Bayes detectará más fácilmente un elemento de clase 10 o clase 1. A la inversa, los puntos morados y verdes se agrupan en el mismo lugar del plano de 2 dimensiones y se superponen. Como resultado, el algoritmo de Bayes tendrá dificultades para diferenciar los números 2 y 8, ya que su distribución es similar. El algoritmo de Bayes tendrá mayores dificultades para determinar a qué clase pertenece el dato.

Esto se confirma si prestamos atención a la matriz de confusión resultante de este aprendizaje con 2 dimensiones.

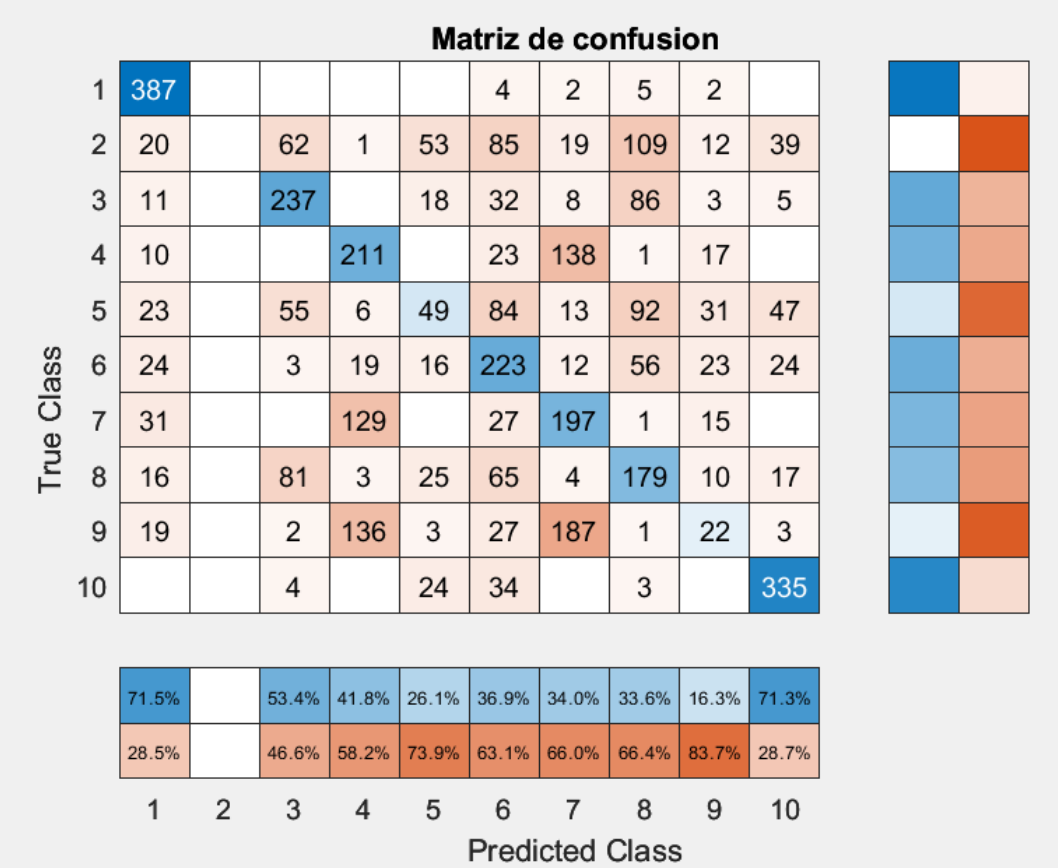

Figura 2: Matriz de confusión con datos de entrenamiento en 2 dimensiones

Las clases 1 y 10 se predijeron bastante bien (datos altos en azul oscuro en la diagonal) como era de esperar. También vemos que hay confusión entre los datos 2 y 8.

Debido al escaso número de dimensiones conservadas y la pérdida de información asociada, el aprendizaje de Bayes no fue un gran éxito. De hecho, la tasa de acierto es de solo 46 %, lo cual es muy baja. Sin embargo, muestro este valor bajo. No esperaba que con dos dimensiones pudiéramos detectar con precisión adecuada las

clases 1 y 10.

	Tasa de acierto
Datos de entrenamiento	0.46
Datos de test	0.466

Cuadro 1: Tasa de acierto con 2 dimensiones

Sin embargo, dado que la tasa de acierto es baja, así como la precisión y el recall, sería conveniente estudiar los datos con un mayor número de dimensiones.

Búsqueda del número de dimensiones ideal La técnica PCA es una técnica potente que permite reducir el número de dimensiones. Como ya se ha mencionado, el número mínimo de dimensiones debería determinarse ahora para no alterar los resultados de la clasificación. Por lo tanto, recordamos que en la práctica anterior medimos una tasa de acierto del orden del 95 % con los datos de entrenamiento. Así, en la continuación consideraremos que "no alterar los resultados de la clasificación"significa obtener una tasa de acierto superior a 95 %.

Así, hemos iterado con un paso de 1 en el número de dimensiones hasta obtener una tasa de acierto considerado suficiente.

Hemos conseguido que el número mínimo de dimensiones a considerar para obtener la tasa de acierto del 95 % sea 16. Este número de dimensiones nos permite obtener una tasa de acierto del 95.275 % con los datos de entrenamiento y del 93.8 % con los datos de test. Aunque los resultados son ligeramente inferiores a los obtenidos en la práctica anterior, podemos alegrarnos de estos resultados. De hecho, hemos dividido por 25 el número de dimensiones estudiadas, lo que resulta en un considerable ahorro de tiempo al ejecutar el algoritmo de clasificación Bayesiana.

	Tasa de acierto
Datos de entrenamiento	0.95275
Datos de test	0.938

Cuadro 2: Tasa de acierto con 16 dimensiones

También continué mis pruebas tratando con números de dimensiones más grandes. Me di cuenta de que no era pertinente considerar más dimensiones. En efecto, el aumento en términos de tasa de acierto aportado por esta mayor complejidad de los datos no vale la pena. La ejecución del algoritmo de clasificación será mucho más larga para no cambiar el orden de magnitud de la tasa de acierto, como se muestra

en la siguiente tabla (columna 1: número de dimensiones, columna 2: tasa de acierto con datos de entrenamiento y columna 3: tasa acierto con datos de test).

11	0.9163	0.9010
12	0.9325	0.9170
13	0.9420	0.9310
14	0.9440	0.9290
15	0.9478	0.9370
16	0.9528	0.9380
17	0.9568	0.9460
18	0.9578	0.9490
19	0.9598	0.9540
20	0.9628	0.9570
21	0.9650	0.9530
22	0.9663	0.9570
23	0.9675	0.9590
24	0.9685	0.9630
25	0.9713	0.9630
26	0.9715	0.9660
27	0.9710	0.9670
28	0.9728	0.9690
29	0.9733	0.9680
30	0.9753	0.9710

Figura 3: Comparación de métricas asociadas con el problema para diferentes valores de k entre 11 y 30 (número de dimensiones)

Con la elección de las dimensiones a k = 16, podemos así darnos cuenta del rendimiento de nuestro modelo gracias al estudio de la matriz de confusión, tanto con los datos de entrenamiento como con los datos de test.

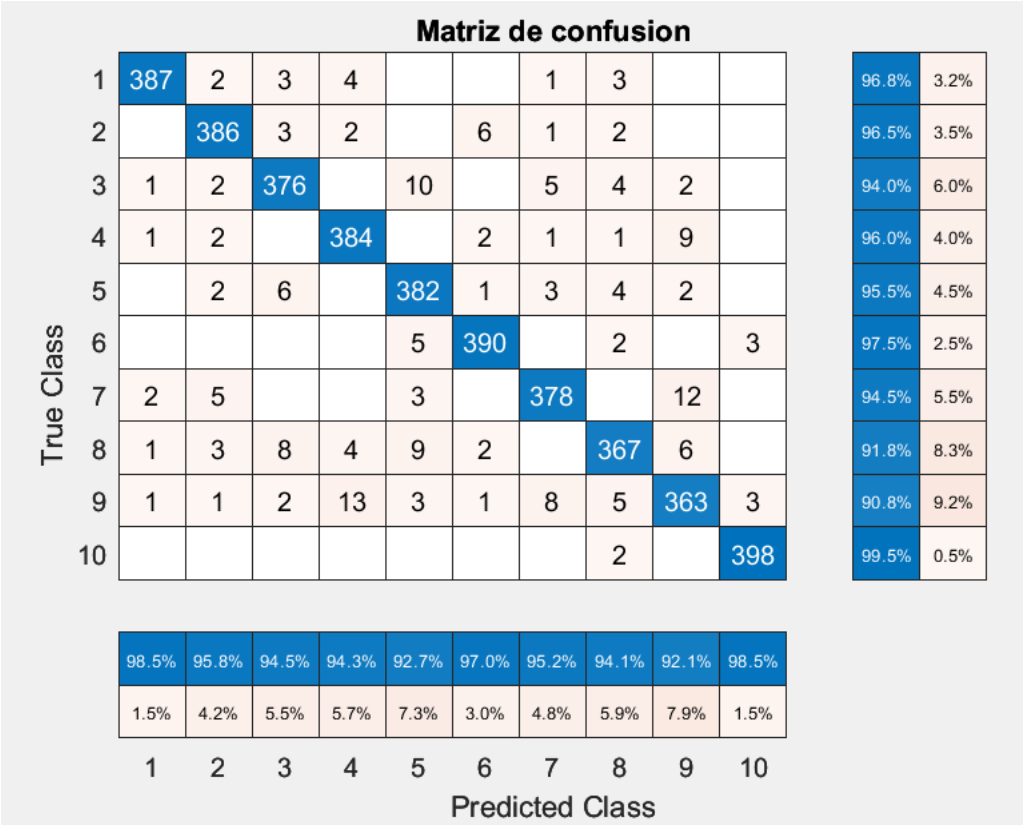

Figura 4: Matriz de confusión con datos de entrenamiento en 16 dimensiones

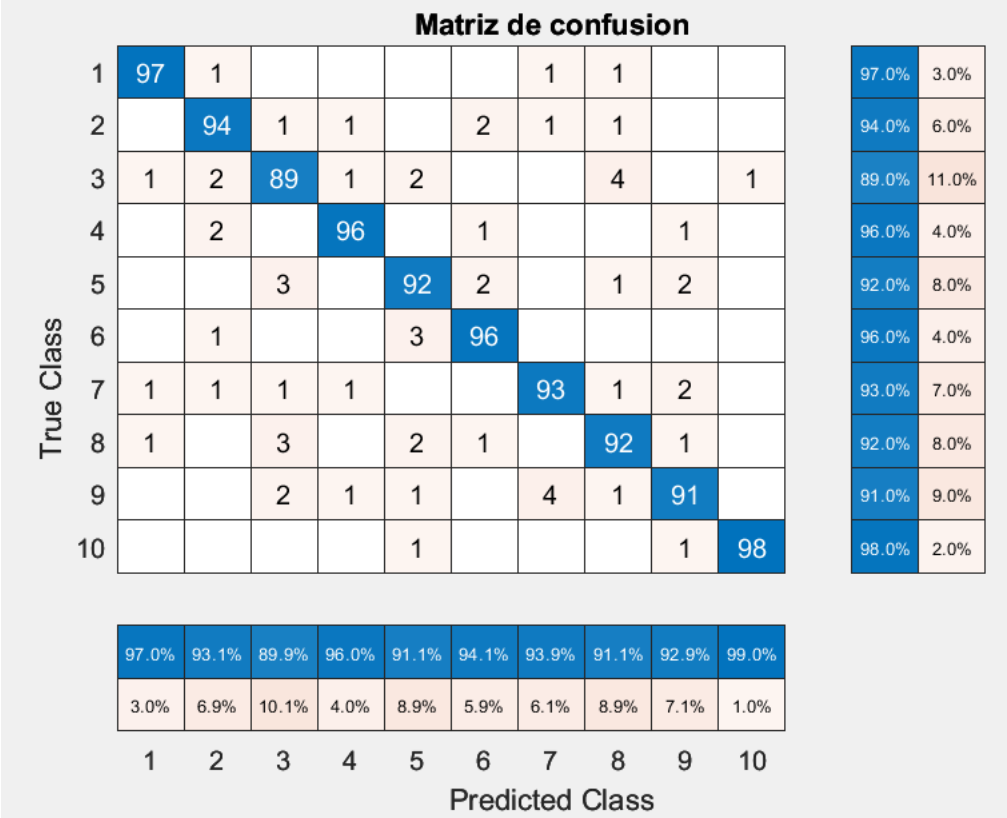

Figura 5: Matriz de confusión con datos de test en 16 dimensiones

Notamos como se esperaba que la precisión y el recall son significativamente más altos, símbolo de una mejor clasificación bayesiana. El número de buenas predicciones del modelo para cada clase es mucho mayor y el número de errores cometidos es mucho menor.

2.1.2. PCA y mantenimiento de la variabilidad

Ahora que hemos visto un primer método de funcionamiento del algoritmo PCA, que consiste en elegir por nosotros mismos el número de dimensiones deseado, vamos a estudiar el dataset MNIST con una versión del algoritmo PCA que permite mantener una cierta variabilidad de los datos. En el caso de esta práctica, trabajaremos con un umbral de 0.99, lo que significa que queremos mantener al menos el 99 % de la varianza de los datos iniciales.

Aplicando el algoritmo proporcionado en el estudio anterior a la realización de la práctica anterior, obtenemos que el número de dimensiones a considerar para conservar el 99 % de la varianza inicial es 153. Este número de dimensiones es claramente superior al mencionado en la parte anterior para obtener una tasa de acierto del 95 %. Así que ahora vamos a estudiar las confusiones, tanto con los datos de entrenamiento como con los datos de test, para ver las ventajas y desventajas de este método. A título indicativo, el número de dimensiones necesarias para conservar el 95 % de la variabilidad de los datos iniciales es 79. Con este número de dimensiones, conseguimos obtener una tasa de acierto del 98 % con los datos de entrenamiento.

	Tasa de acierto
Datos de entrenamiento	0.992
Datos de test	0.966

Cuadro 3: Tasa de acierto con 153 dimensiones (variabilidad = 99 %)

Al mantener una mayor variabilidad, tuvimos que considerar un mayor número de dimensiones. Este número aumenta la complejidad de los datos y, por tanto, el algoritmo de clasificación de Bayes requiere más tiempo de trabajo. Sin embargo, observamos que el rendimiento es mayor, tanto desde el punto de vista de la tasa de acierto como desde las conclusiones de la matriz de confusión a continuación.

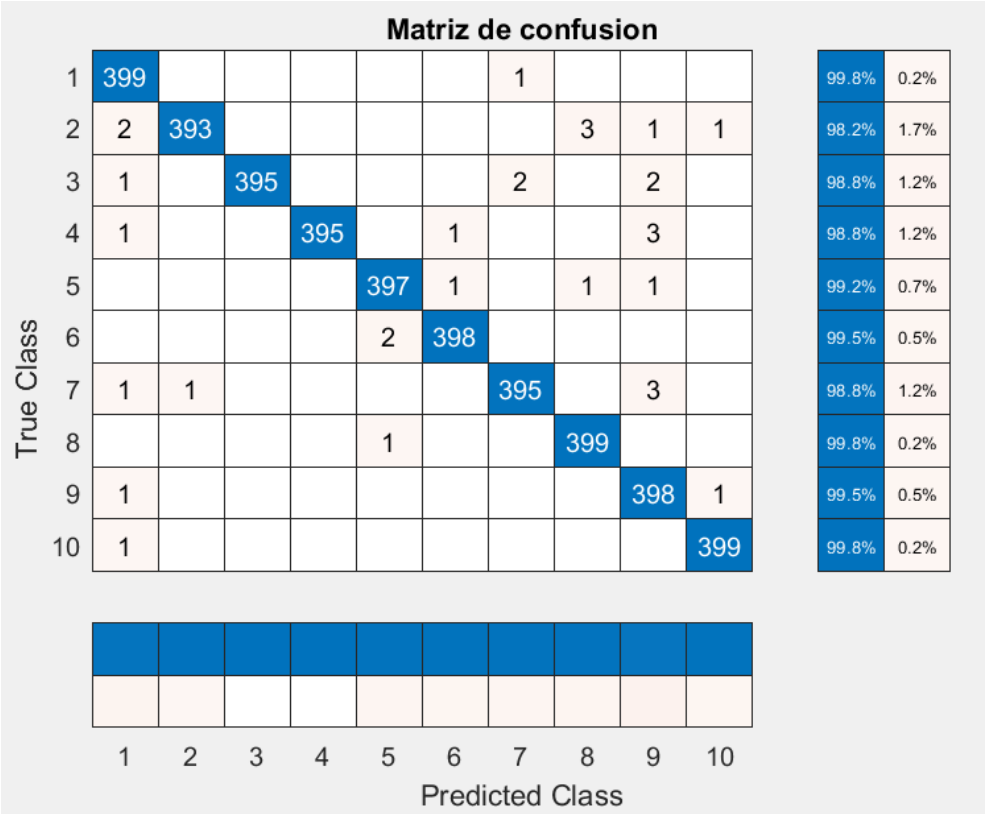

Figura 6: Matriz de confusión con datos de entrenamiento y variabilidad = 0.99

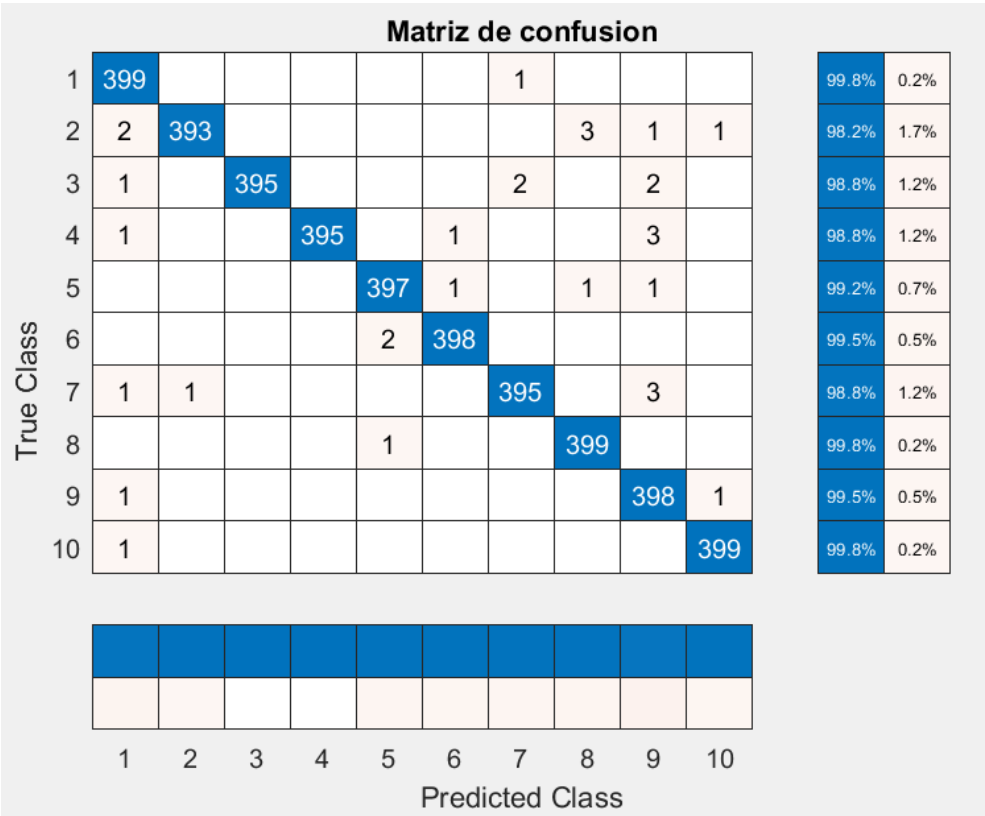

Figura 7: Matriz de confusión con datos de test y variabilidad = 0.99

El número de predicciones correctas establecidas por la clasificación bayesiana es mucho mayor, como lo demuestra la diagonal. Observamos que la precisión y el recall son significativamente más altos que antes, tanto con datos de entrenamiento como de test.

Este mayor número de buenas predicciones está relacionado con el hecho de que hemos sido capaces de elegir el número mínimo de dimensiones para mantener cierta variabilidad de los datos. Los datos se simplifican entonces (menos parámetros gracias a la reducción de dimensión) sin por ello haber perdido información (gran variabilidad conservada).

2.1.3. Comparación de los 2 métodos

Por lo tanto, fijar un número de dimensiones o mantener una cierta variabilidad de los datos son los 2 métodos de aplicación del algoritmo PCA estudiado aquí. Aunque el segundo método ha permitido obtener mejores resultados, tengo una preferencia por el primer método. En efecto, con 10 veces menos dimensiones que en el método 2, hemos conseguido obtener un índice de acierto del mismo orden de magnitud, aunque ligeramente más bajos que con el segundo método.

Durante esta primera parte de la práctica, me interesé especialmente por el número de dimensiones que deben conservarse para poder después tener la mejor clasificación posible con Bayes. Por lo tanto, he obtenido los siguientes tasa de acierto en función del número de componentes conservados.

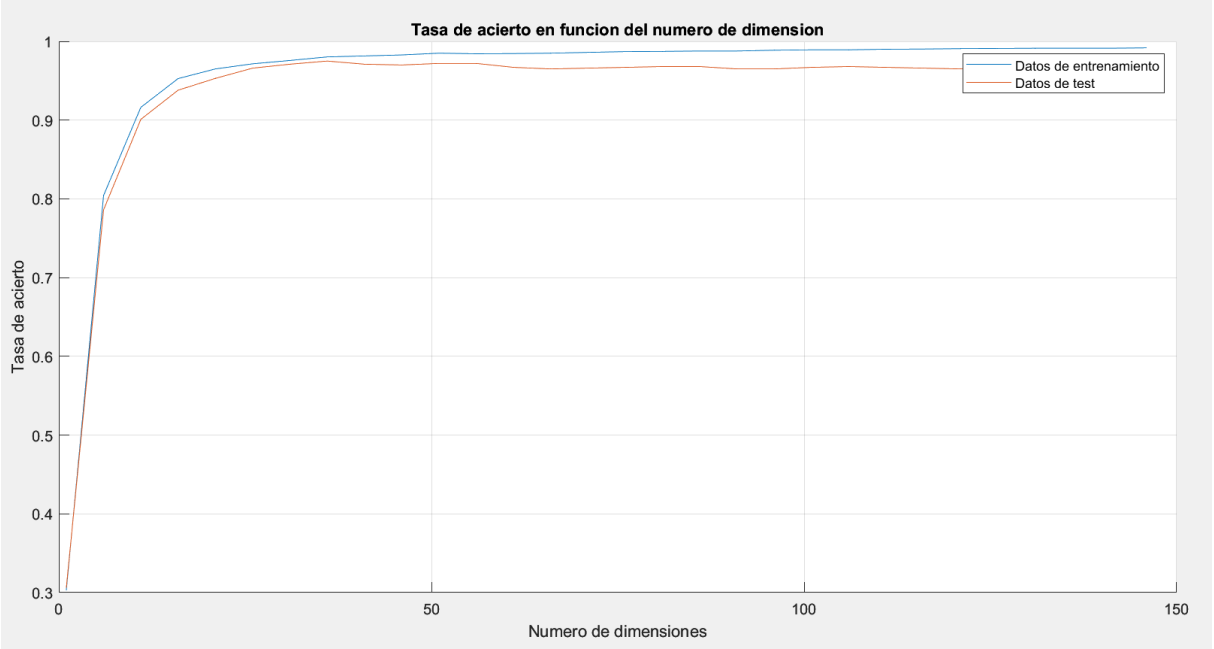

Figura 8: Tasa de acierto en función del número de componentes conservados

Este gráfico coincide con mi razonamiento anterior. El aumento del número de dimensiones por encima de un determinado umbral no permitirá mejorar sensiblemente la clasificación bayesiana aplicada posteriormente.

Sin embargo, la selección de los componentes que deben conservarse es más compleja de lo que parece. En efecto, como se ve en la práctica, la distinción de las letras .ºz "q.en una imagen puede hacerse con muy pocos componentes, en particular los que captan menos varianza.

2.2. Cuestión 2: Filtrado de ruido en imágenes

Al reducir el tamaño y la complejidad de los datos, ahora podemos reducir significativamente el tiempo de ejecución del aprendizaje de clasificación de Bayes y mejorar el rendimiento del modelo. Ahora sería interesante ver si el algoritmo de aprendizaje no supervisado PCA es resistente al ruido.

2.2.1. Comprensión del tipo de ruido añadido

Al principio, tuve que entender qué tipo de ruido se introducía en los datos. Para ello, después de analizar el código y probar con diferentes valores, me di cuenta de que consideramos un ruido que corresponde en realidad a la realización de una ley normal centrada de desviación estándar noise_std. Como parte de esta práctica, consideraremos noise_std = 0.3, cantidad proporcionada en el código asociado.

Sin embargo, en esta parte vamos a probar diferentes valores de ruido. De este modo, podremos entender gráficamente mejor qué es el ruido. Para el 9, obtenemos las siguientes imágenes normales/imágenes ruidosas:

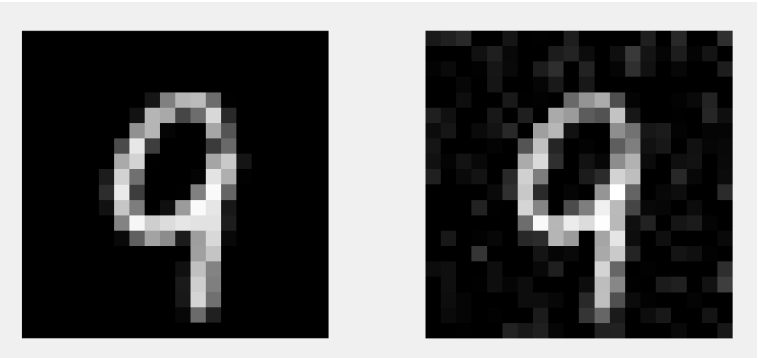

Figura 9: Imagen normal/imagen ruidosa del dígito 9 con noise_std = 0.1

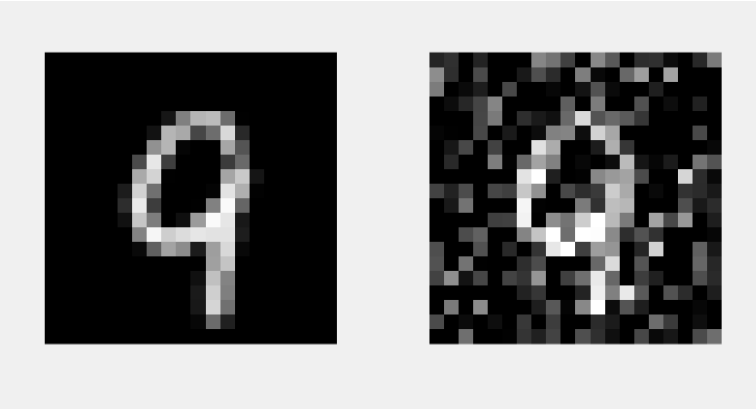

Figura 10: Imagen normal/imagen ruidosa del dígito 9 con noise_std = 0.3

Figura 11: Imagen normal/imagen ruidosa del dígito 9 con noise_std = 0.8

Como era de esperar, cuanto más grande es el ruido, más ruidosa es la imagen, por lo que menos se parece a la imagen original. El ruido complicará el reconocimiento del número.

2.2.2. PCA para eliminar el ruido de las imágenes

PCA para reducir las dimensiones Ahora vamos a intentar eliminar el ruido que se introdujo anteriormente con el PCA. Para ello, vamos a aplicar el mismo algoritmo que antes. Como se ha explicado anteriormente, voy a privilegiar aquí la velocidad de ejecución sobre la pertinencia del reconocimiento. Por lo tanto, el objetivo aquí será obtener de nuevo una tasa de acierto de 0.95 con los datos de entrenamiento.

Obtenemos así que hay que tener en cuenta 26 dimensiones para obtener una tasa de acierto del 95 % con los datos de entrenamiento. Esto da lugar a la siguiente tasa de acierto con datos de entrenamiento y de test:

	Tasa de acierto
Datos de entrenamiento	0.95175
Datos de test	0.926

Cuadro 4: Tasa de acierto con 26 dimensiones

Por lo tanto, el algoritmo PCA fue capaz de reducir el tamaño de los datos, mientras que la eliminación de ruido. De hecho, logramos obtener una tasa de acierto alta como se deseaba (del orden del 95 %) a pesar de la presencia de ruido y una pérdida de información relacionada con la reducción de la dimensión.

Además, obtenemos las siguientes matrices de confusión:

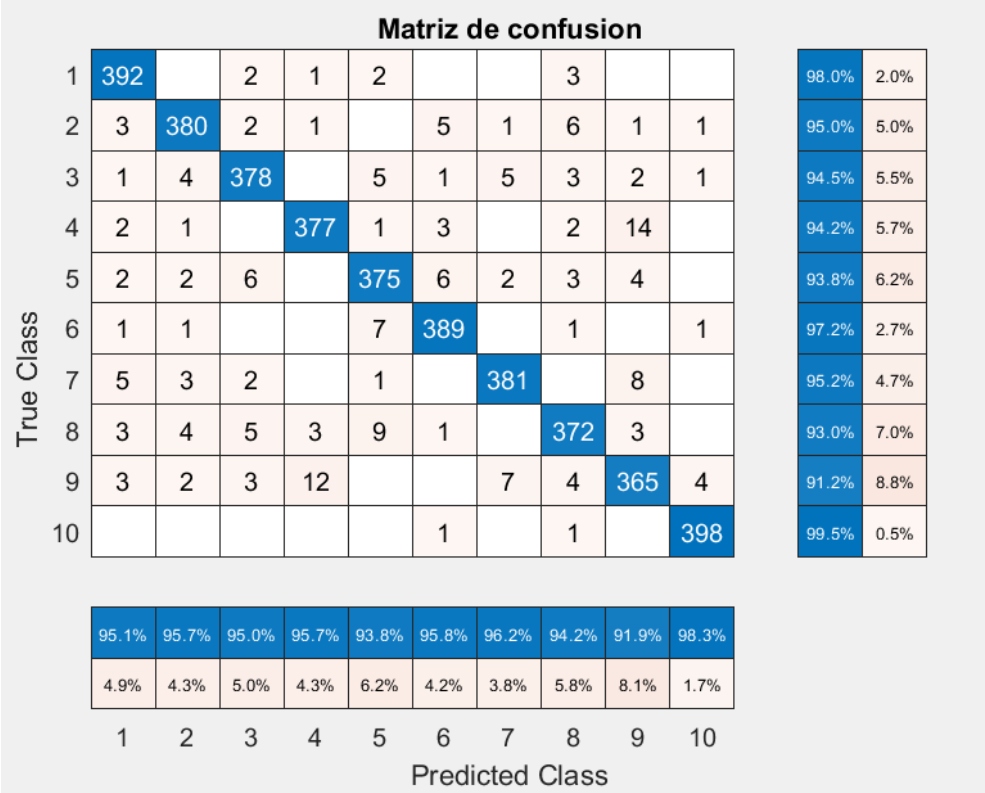

Figura 12: Matriz de confusión con datos de entrenamiento y con 26 dimensiones

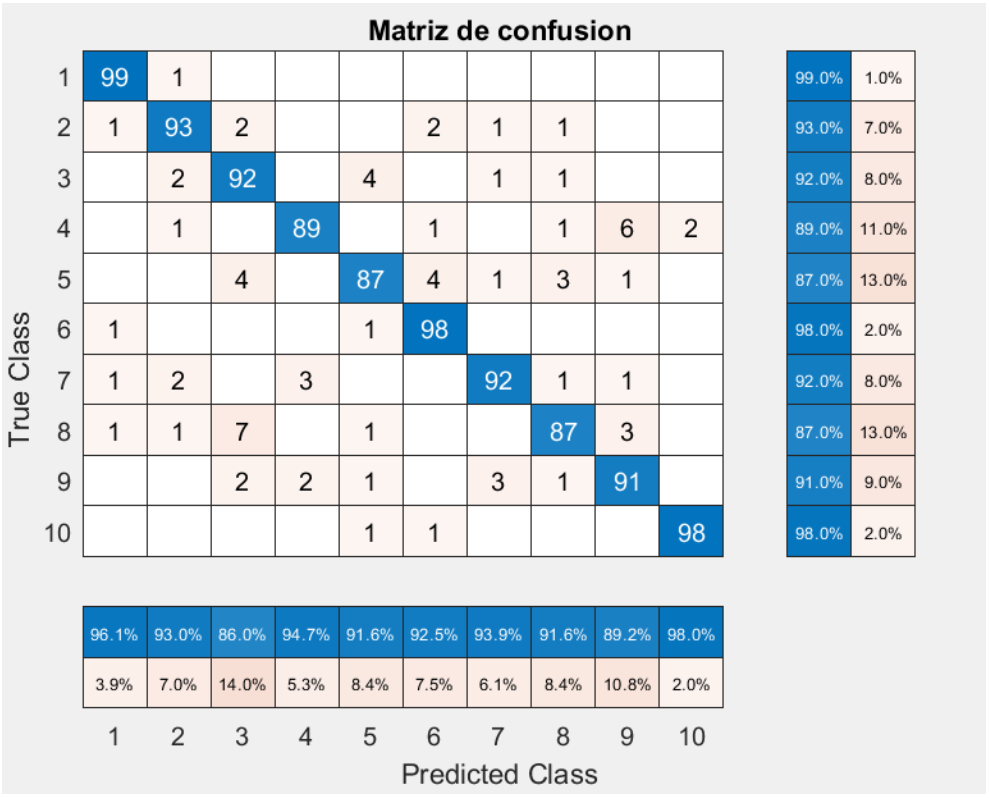

Figura 13: Matriz de confusión con datos de test y con 26 dimensiones

Estas matrices de confusión confirman la información proporcionada por la tasa de acierto. En efecto, se cometen pocos errores de predicción, tanto en el entrenamiento con la clasificación bayesiana como en la fase de test.

PCA para mantener variabilidad >0.99 También he probado para obtener la varianza de los datos con el algoritmo PCA que consiste en elegir el número de dimensiones de tal manera que se alcance un cierto valor de variabilidad.

No hay novedad desde el punto de vista lógico del código. De hecho, es solo una copia con algunas adaptaciones del código realizado en la pregunta 1 con el método para mantener la variabilidad. De este modo, he conseguido que para conservar el 99 % de la variabilidad de los datos iniciales, es necesario conservar 394 dimensiones. Este gran número de dimensiones resulta del ruido con una distribución normal en cada píxel. En efecto, para conservar el 99 % de la varianza, los píxeles inicialmente todos negros que tenían una varianza nula al principio (por ejemplo, en los bordes de la imagen) tienen ahora una varianza que tiene en cuenta el algoritmo PCA.

	Tasa de acierto
Datos de entrenamiento	1
Datos de test	0.878

Cuadro 5: Tasa de acierto con datos ruidosos y variabilidad = 0.99

Obtener una tasa de acierto de 1 con los datos de entrenamiento y solo 87.8 % con los datos de test nos indica que el aprendizaje ha llevado a un problema de sobreajuste. De hecho, la clasificación fue capaz de aprender de memoria los datos de entrenamiento, pero no fue capaz de identificar las propiedades interesantes para poder generalizar con los datos de test.

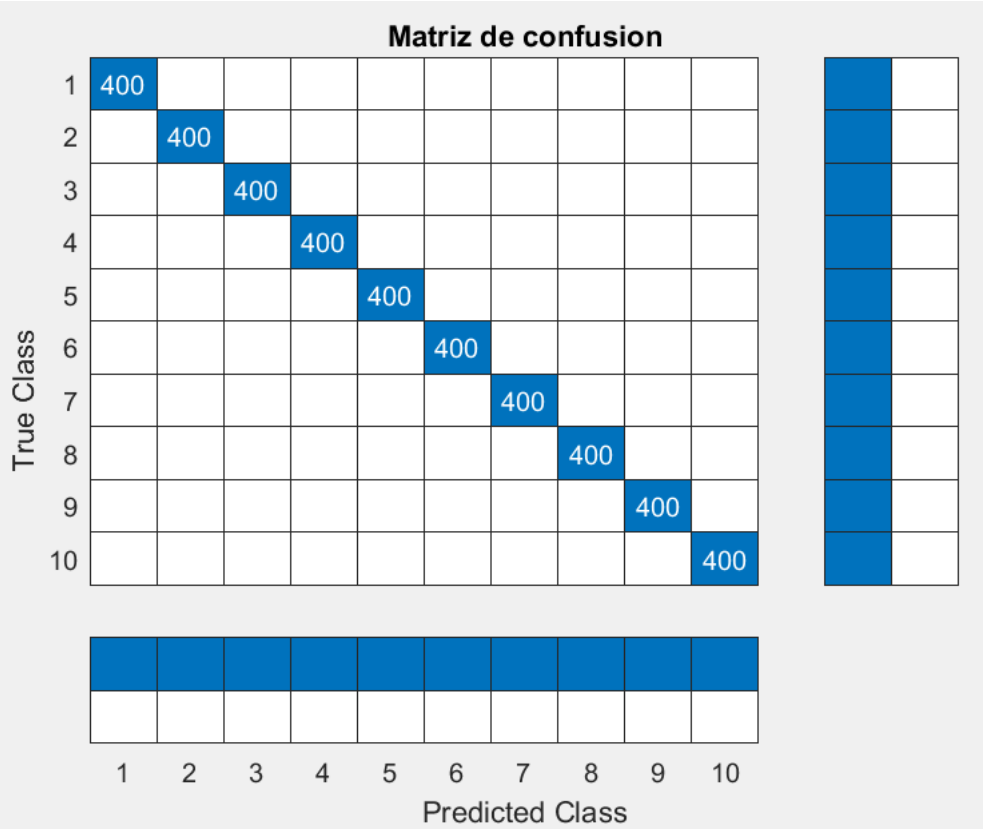

Figura 14: Matriz de confusión con datos de entrenamiento y variabilidad = 0.99

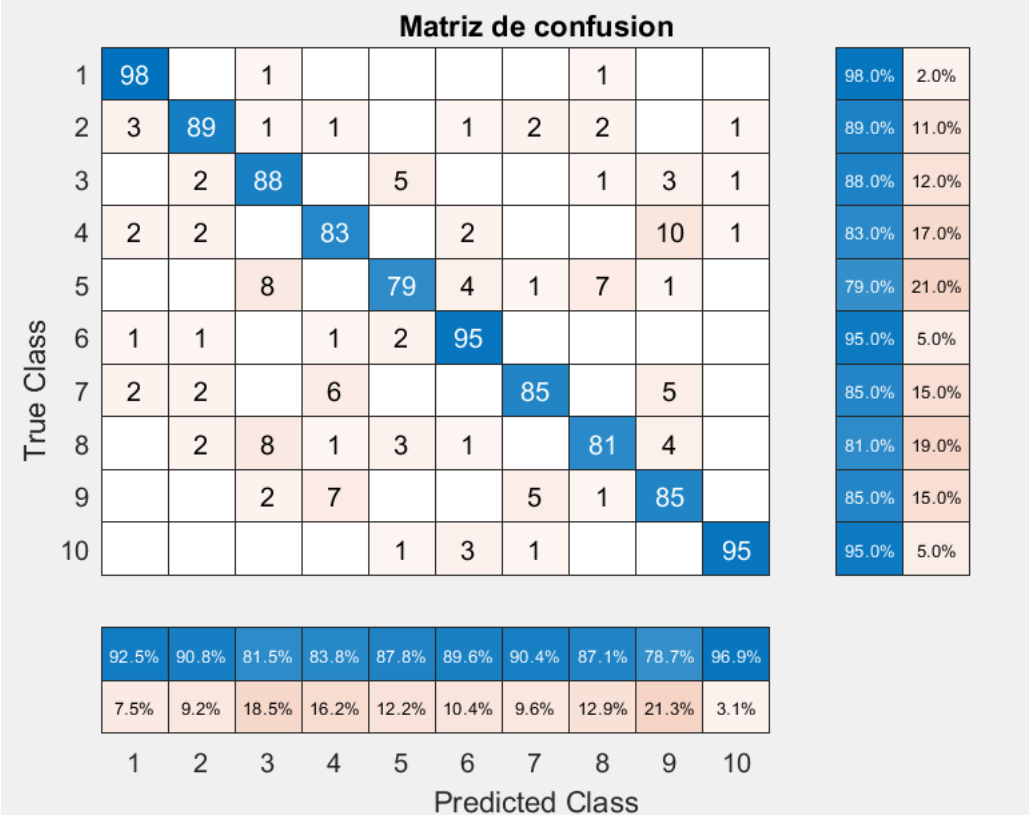

Figura 15: Matriz de confusión con datos de test y variabilidad = 0.99

2.2.3. Comparación del denoising en función del número de dimensiones conservadas

Durante esta parte, me interesé por el impacto del número de dimensiones elegido para la reconstrucción de la imagen. En consecuencia, he obtenido los siguientes resultados:

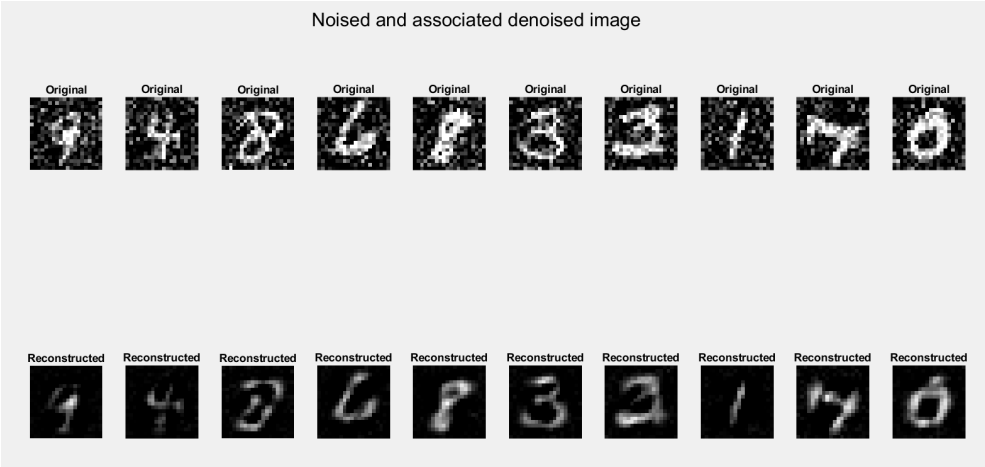

Figura 16: Comparación imágenes normales / imágenes ruidosas con k = 25

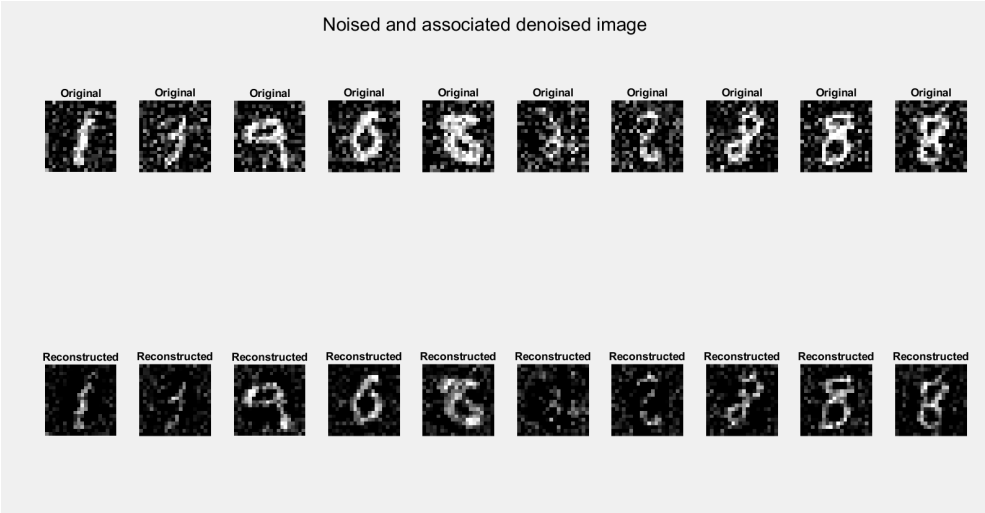

Figura 17: Comparación imágenes normales / imágenes ruidosas con k = 140

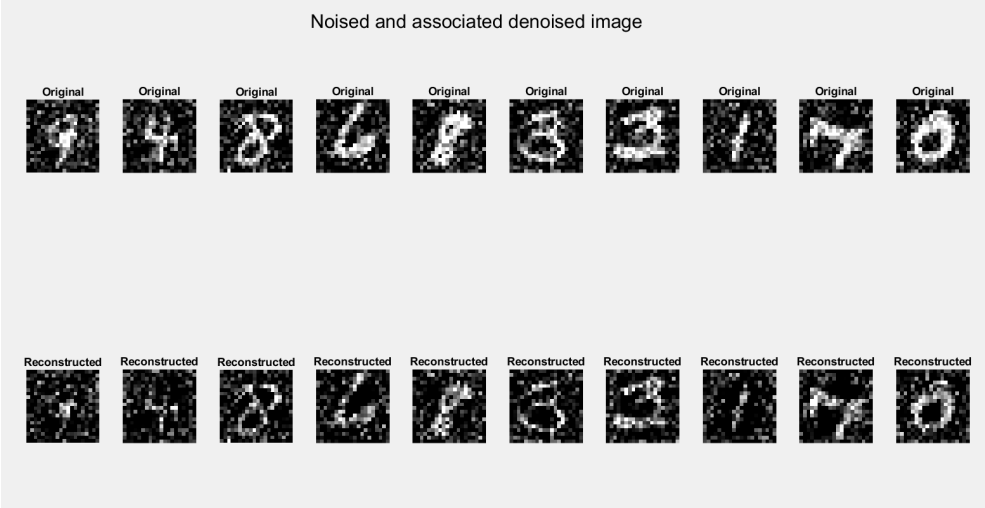

Figura 18: Comparación imágenes normales / imágenes ruidosas con k = 393

Entonces me di cuenta de que cuantos más componentes se seleccionan, más se mantiene la variabilidad del ruido introducido. Por lo tanto, tomamos en consideración píxeles innecesarios y esto se siente en el denoising.

2.2.4. Comparación de la clasificación con y sin ruido

Por lo tanto, pudimos aplicar el algoritmo PCA para los llamados datos normales y ruidosos. Observamos que PCA es bastante eficiente en la eliminación de ruido.

Sin embargo, cuando se analiza el método de mantener cierta variabilidad, podemos deducir que el algoritmo PCA es sensible al ruido. En efecto, al querer mantener un umbral importante de variabilidad, hemos conservado componentes innecesarios cuya variabilidad observada sólo depende del ruido introducido. Esto implicó que

para mantener el 99 % de la variabilidad de los datos de ruido, fue necesario conservar casi toda la información original (394 de 400 dimensiones con datos de ruido frente a 153 dimensiones con datos no ruidosos).

Esto también hizo que la clasificación fuera más difícil de lo que esperábamos. De hecho, el ruido que altera las imágenes hace que sea mucho más difícil diferenciarlas, especialmente para la inteligencia artificial. Deduzco que es por eso que los "Captcha"para comprobar si no somos un robot en los sitios web ofrecen cada vez más datos borrosos (por lo tanto ruidosos). En efecto, dado que el reconocimiento es ya complicado para un ser humano, resulta casi imposible para una inteligencia artificial generalizar las propiedades interesantes.

Así pues, cuanto mayor sea la cantidad de ruido añadido, mayor será el número de dimensiones conservadas por PCA y mayor será la dificultad del algoritmo de clasificación para cumplir su misión.

2.2.5. Opcional: Estrategia para seleccionar el número de componentes a utilizar en el denoising

Para evitar el ruido en la parte anterior, utilizamos el algoritmo PCA. Sin embargo, hemos visto claramente que PCA no siempre puede eliminar el ruido. Por lo tanto, sería relevante encontrar una técnica para eliminar el ruido de los datos.

Aunque no es académico y no presenta nada matemático, podría ser interesante seleccionar manualmente los datos a conservar. En efecto, a la introducción de una cierta cantidad de ruido, las imágenes se vuelven totalmente irreconocibles, y por lo tanto, me parece poco pertinente conservarlos porque incluso un humano no sería capaz de predecir que se trata de la imagen de un número entre 0 y 9. Una vez realizada esta costosa selección a mano, podemos aplicar el PCA para determinar el número de dimensiones ideal.

Otro enfoque podría adoptarse también por computadora. En efecto, hemos visto que con datos normales, los píxeles asociados a una gran variabilidad se sitúan en el centro de la imagen. Los píxeles en los contornos de la imagen son todos negros (varianza nula). Esto puede generalizarse en la vida cotidiana. Hay muy pocas imágenes donde la información crucial está en los bordes. Por lo tanto, podría ser útil recortar la imagen (es decir, eliminar los píxeles de los contornos) y luego llamar a PCA. Por lo tanto, PCA ya no tendrá en cuenta la variabilidad de píxeles en los contornos relacionados con el ruido introducido.

Conclusión

Asi, el Análisis Principal de Componentes (PCA) es una técnica de análisis multivariante que reduce la dimensionalidad de los datos transformando un conjunto de variables en un conjunto de variables linealmente no correlacionadas llamadas “componentes principales". Estos componentes principales capturan la mayor parte de la varianza de los datos originales y permiten visualizar y comprender las relaciones subyacentes entre las variables.

El algoritmo PCA podrá permitir mejorar la calidad de la clasificación bayesiana posteriormente, reduciendo la dimensión del espacio de las variables y evitando el ruido como hemos observado anteriormente.

Sin embargo, esto no siempre garantiza evitar completamente el sobreajuste como se ve al final de esta práctica. Esto depende de la calidad de los datos y de la complejidad del modelo bayesiano utilizado. Por tanto, me parece importante considerar otros métodos de regularización y optimización para evitar el sobreajuste.

Esta práctica me ha permitido comprender mejor los beneficios del algoritmo PCA y sus limitaciones. Trataré de movilizarlo lo mejor posible durante mis proyectos personales y profesionales, en particular para la compresión de datos, el reconocimiento de formas y la visualización de datos.

Índice de figuras

Índice de cuadros