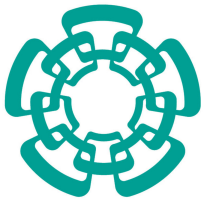
LTI Cinvestav



Comparación de clasificadores probabilísticos

Reconocimiento de patrones

Profesor: Dr. Wilfrido Gómez Flores

Estudiante: Rafael Pérez Torres

1. Introducción

Los conceptos probabilísticos proveen de elementos suficientes para realizar el reconocimiento y clasificación de entidades. Distintos enfoques y variantes que dependen de las características de los datos pueden ser utilizados para la tarea de clasificación.

El presente documento describe de forma breve la implementación y análisis de resultados obtenidos por distintas técnicas probabilísticas al intentar clasificar un dataset del mundo real.

2. Marco teórico

Los clasificadores basados en la teoría de Bayes funcionan utilizando una estimación de los datos obtenidos del mundo real. Como ha sido analizado, dicha estimación comúnmente es realizada mediante la *PDF*[1] gaussiana.

La *PDF* gaussiana puede ser definida a grandes rasgos si se conoce la media de los datos involucrados en cada una de las clases, así como las matrices de covarianza de los atributos de aquellas instancias pertenecientes a cada una de las clases.

[1] *PDF*, función de densidad de probabilidad, por sus siglas en inglés.

En ocasiones es posible tener, de antemano o a través de algún proceso de análisis, cierto conocimiento acerca de la naturaleza de los datos. Gracias a algunas características de esta información, los valores de los parámetros de la *PDF* gaussiana pueden ser utilizados para realizar una simplificación o especialización en los cálculos. En particular, las combinaciones de los valores de las matrices de covarianza de cada una de las clases permite identificar algunos casos específicos en los clasificadores bayesianos.

2.1. Caso 1: $\Sigma_i = \sigma^2 I$

Si las características son estadísticamente independientes y además se observa que la varianza de los atributos es la misma para todas las clases, entonces es posible realizar una simplificación.

Si además de lo anterior, las clases involucradas son equiprobables y el valor de la varianza es 1 ($\sigma^2 = 1$) entonces se obtiene el clasificador de distancia Euclidiana:

$$g_i(x) = -\frac{1}{2}(x - \mu_i)^T(x - \mu_i)$$

2.2. Caso 2: $\Sigma_i = \Sigma$ (diagonal)

Para este caso, las clases presentan la misma covarianza aunque sus atributos definen distintas varianzas. Si se asume que las clases son equiprobables, entonces es posible definir el clasificador de distancia Mahalanobis:

$$g_i(x) = -\frac{1}{2}(x - \mu_i)^T\Sigma_i^{-1}(x - \mu_i)$$

2.3. Caso 3: $\Sigma_i = \Sigma$ (no diagonal)

Este caso define las mismas características que el anterior, sin embargo la matriz de covarianza no es diagonal. Por esta razón, Σ^{-1} será un factor de rotación y alargamiento en el espacio de características.

2.4. Caso 4: $\Sigma_i = \sigma_i^2 I$

En este caso, cada una de las clases define una matriz de covarianza diferente, pero que es proporcional a la matriz identidad. El valor del discriminante puede ser calculado a través de:

$$g_i(x) = -\frac{1}{2}(x-\mu_i)^T\sigma_i^{-2}(x-\mu_i) - \frac{d}{2}ln \mid \sigma_i^2 \mid + lnP(w_i)$$

2.5. Caso 5: $\Sigma_i \neq \Sigma_j$

Para este caso no existe coincidencia alguna entre los valores de las matrices de covarianza. Este representa el caso más general del clasificador bayesiano, cuyo valor de discriminante es definido como:

$$g_i(x) = -\frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i) - \frac{1}{2}ln \mid \Sigma_i \mid + lnP(w_i)$$

2.6. Clasificador Naïve Bayes

El cálculo de las matrices de covarianza para conjuntos de datos de alta dimensionalidad resulta temporal y computacionalmente costoso. Una consideración un tanto ingenua podría ser el afirmar que los valores de los datos no guardan correlación alguna entre sí, evitando realizar el cálculo de esta matriz.

Así, la pertenencia hacia una clase puede obtenida a través de:

$$\prod_{j=1}^{l} P(x_j \mid w_i) = \prod_{j=1}^{l} \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} \exp(-\frac{(x_j-\mu_{ij})}{2\pi\sigma})$$

Finalmente, la regla *MAP* queda definida como:

$$\hat{\omega}_{ML}(x) = \arg \overset{\text{máx}}{w_i \epsilon \Omega} P(w_i) \prod_{j=1}^{l} P(x_j \mid w_i)$$

2.7. Clasificador *k-nn* (k - nearest neighbours)

Existe otra vertiente de clasificadores que también intentan asignar la clase más probable a un patrón pero sin utilizar elementos de la teoría de Bayes.

El clasificador *k-nn* (k - nearest neighbours, k - vecinos más cercanos) asigna la clase a un patrón, de acuerdo a la clase más común descrita por los k elementos más cercanos a éste.

Básicamente, el algoritmo *k-nn* dado un patrón, realiza las siguientes etapas:

1. Obtener la distancia hacia todos los puntos existentes (dichos puntos ya cuentan con las etiquetas de clase).

2. Elegir los k elementos más cercanos.

3. Asignar al patrón la clase que más frecuencia tenga entre los k elementos más cercanos.

Naturalmente, un factor importante en este algoritmo es la métrica para estimar la distancia entre los puntos, la cual afectará la calidad de los resultados obtenidos.

3. Metodología

Cada uno de los algoritmos fue implementado de forma individual en Matlab con la intención de clasificar los elementos de un conjunto de 7 datasets, pertenecientes a características de números escritos de forma manual.

Se realizó la división de cada dataset en 70 % para entrenamiento y 30 % para prueba, ejecutando 31 veces cada clasificador por cada uno de los datasets.

Para la familia de algoritmos bayesianos –caso general, clasificador Euclidiano y clasificador Mahalanobis (caso 3)– se utilizó el mismo procedimiento de entrenamiento, mientras que para el clasificador Naïve Bayes se optó por realizar un entrenamiento distinto a nivel de código, ya que no requiere calcular las matrices de covarianza por cada una de las clases. Para el algoritmo *k-nn* que no requiere entrenamiento, se decidió que el 70 % de datos de entrenamiento utilizado por los otros algoritmos fuera empleado para definir los *vecinos* que existen inicialmente, tomando el 30 % restante para realizar su clasificación.

Los clasificadores Euclidiano y Mahalanobis fueron implementados generalizando los cálculos a través de las fórmulas equivalentes a las distancias Euclidiana $\left(\sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}\right)$ y Mahalanobis $\left(\sqrt{\sum_{i=1}^{n}(p_i - q_i)^T \Sigma^{-1}(p_i - q_i)}\right)$, auxiliándose de los datos obtenidos durante la etapa de entrenamiento.

Adicionalmente, es importante señalar que se realizó una corrección especial (denominada corrección de Laplace) para el entrenamiento del clasificador Naïve Bayes, la cual consiste básicamente en sumar una cantidad pequeña a las varianzas y medias para evitar valores de 0 que nulifiquen a las probabilidades calculadas durante la etapa de clasificación.

	Bayesiano	Euclidiano	Mahalanobis	Naïve Bayes	k-nn (k=1)	k-nn (k=5)	k-nn (k=9)
morphological features	0.093	0.174	0.209	0.249	0.867	1.350	11.683
Zernike moments	0.019	0.037	0.040	0.039	0.084	0.097	0.268
Karhunen-Love coefficients	0.036	0.108	0.138	0.173	0.788	1.225	11.059
Fourier coefficients	0.039	0.098	0.123	0.134	0.347	0.380	0.974
profile correlations	0.129	0.327	0.411	0.481	3.608	4.206	12.720
pixel averages	0.132	0.332	0.417	0.488	3.757	4.297	13.123
todas	0.132	0.332	0.416	0.489	3.910	4.302	13.262

Cuadro 1: Resumen de los tiempos de ejecución obtenidos

	Bayesiano	Euclidiano	Mahalanobis	Naïve Bayes	k-nn (k=1)	k-nn (k=5)	k-nn (k=9)
morphological features	86.613	19.661	4.027	19.833	15.242	78.962	15.306
Zernike moments	54.253	28.468	8.538	22.204	17.651	8.231	26.887
Karhunen-Love coefficients	27.371	19.935	4.253	22.349	5.258	9.387	2.672
Fourier coefficients	30.048	25.688	6.608	23.968	90	7.677	90
profile correlations	23.756	8.449	1.247	7.412	2.505	1.118	2.230
pixel averages	23.074	7.585	1.212	6.885	2.357	1.060	2.468
todas	22.684	7.399	1.336	6.922	2.691	1.138	3

Cuadro 2: Resumen de los porcentajes de error obtenidos

4. Resultados

Como ha sido descrito, se realizó la ejecución de cada algoritmo sobre cada uno de los datasets 31 veces. Se midió el tiempo de procesamiento así como el porcentaje de errores obtenido durante cada ejecución. El equipo de cómputo utilizado fue una laptop con procesador Intel Core i7 a 2.0 GHz, con 6 GB de memoria *RAM.*

La Tabla 1 muestra las mediciones de tiempo obtenidas durante la ejecución de los algoritmos, mientras que la Tabla 2 muestra lo propio para el porcentaje de errores.

Para la métrica de tiempo de ejecución, mostrada gráficamente en la Figura 1, es evidente que los clasificadores que realizan operaciones complejas con la matriz de covarianza, como el clasificador Bayesiano y el clasificador Mahalanobis (caso 3) obtienen valores muy altos, lo que hasta cierto punto es una situación predecible. Por otro lado, las ejecuciones del algoritmo *k-nn* con valores de $k = \{1,5,9\}$ también observan altos valores, debido al cálculo que se realiza para obtener la distancia entre cada punto de prueba y todos los puntos de entrenamiento. Resulta evidente observar que el clasificador Euclideano y el clasificador Naïve Bayes obtienen los menores tiempos de procesamiento.

Por otro lado, la métrica de porcentaje de errores mostrada gráficamente en la Figura 2, revela que para la mayoría de los casos los clasificadores bayesianos obtienen promedios de error superiores a los de la familia *k-nn*.

Esto puede ser atribuible a las características del dataset y al hecho de que, por natu-

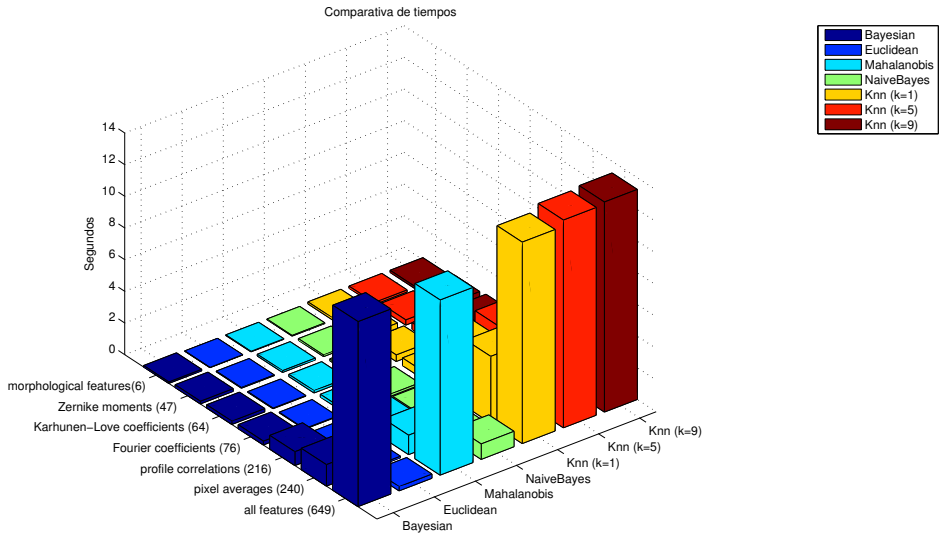


Figura 1: Gráfica de los tiempos de ejecución promedio, por dataset y clasificador

raleza, la clasificación de un dígito manuscrito obedece más a la idea de asignarle la clase a la que se parezca más y no a razones probabilísticas.

Para un dataset con características como el analizado, y en la ausencia de restricciones temporales, los clasificadores de la familia *k-nn* se posicionan como la mejor opción.

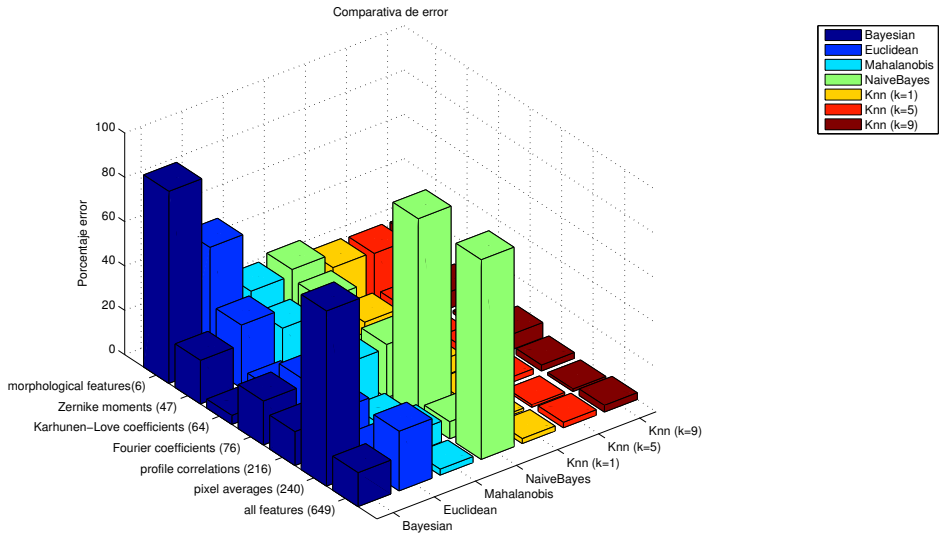


Figura 2: Gráfica de los porcentajes de error obtenidos, por dataset y clasificador

5. Conclusiones

Se ha presentado un análisis de los resultados de tiempo de ejecución y de porcentaje de errores obtenidos por los clasificadores Bayesiano, Euclidiano, Mahalanobis, Naïve Bayes y *k-nn* al procesar un dataset referente a características de dígitos manuscritos.

La familia de los clasificadores *k-nn* obtienen los mejores resultados a costa de un tiempo de ejecución alto, aunque posiblemente mejorable tras refactorización u optimización del código Matlab creado.

Adicionalmente, esta actividad ha permitido conocer las diferencias en los resultados obtenidos por las distintas variantes de los clasificadores bayesianos, retornando a la premisa inicial de que la determinación de un *buen* clasificador depende de la naturaleza en las características de los datos así como de la información que se desea obtener.