Aprendizaje Automático

Práctica 3:

Ajuste de modelos lineales

Curso 2019-2020

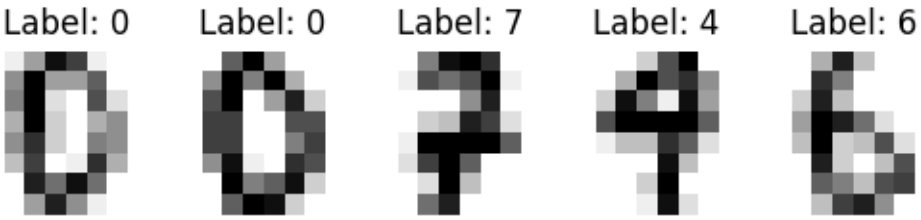
Javier Gálvez Obispo
javiergalvez@correo.ugr.es
Grupo 2 Martes 17:30 – 19:30

1. Optical Recognition of Handwritten Digits Data Set

1.1. Compresión del problema a resolver

Nos encontramos ante un problema de aprendizaje supervisado en el que nos dan un conjunto de imágenes de dígitos escritos a mano y debemos determinar que número es exactamente, un problema de clasificación en el que tenemos que determinar la clase a la que pertenece un dato.

Las imágenes que tenemos son 8x8, por tanto, tendremos 64 atributos por dato, uno por pixel. Cada atributo toma valores enteros entre entre 0 y 16. Las etiquetas son enteros entre 0 y 9, es decir, el dígito que representa la imagen. Por ejemplo:



1.2. Selección de la clase de funciones a usar

Se ha decidido no realizar ninguna transformación de las variables originales ya que el problema es suficientemente sencillo y se obtienen buenos resultados con la clase de funciones lineales.

1.3. Definición de los conjuntos de training y test

Los datos ya vienen separados en dos conjuntos, uno de training, 70% de los datos, y otro de test, 30% restante.

1.4. Preprocesado de datos

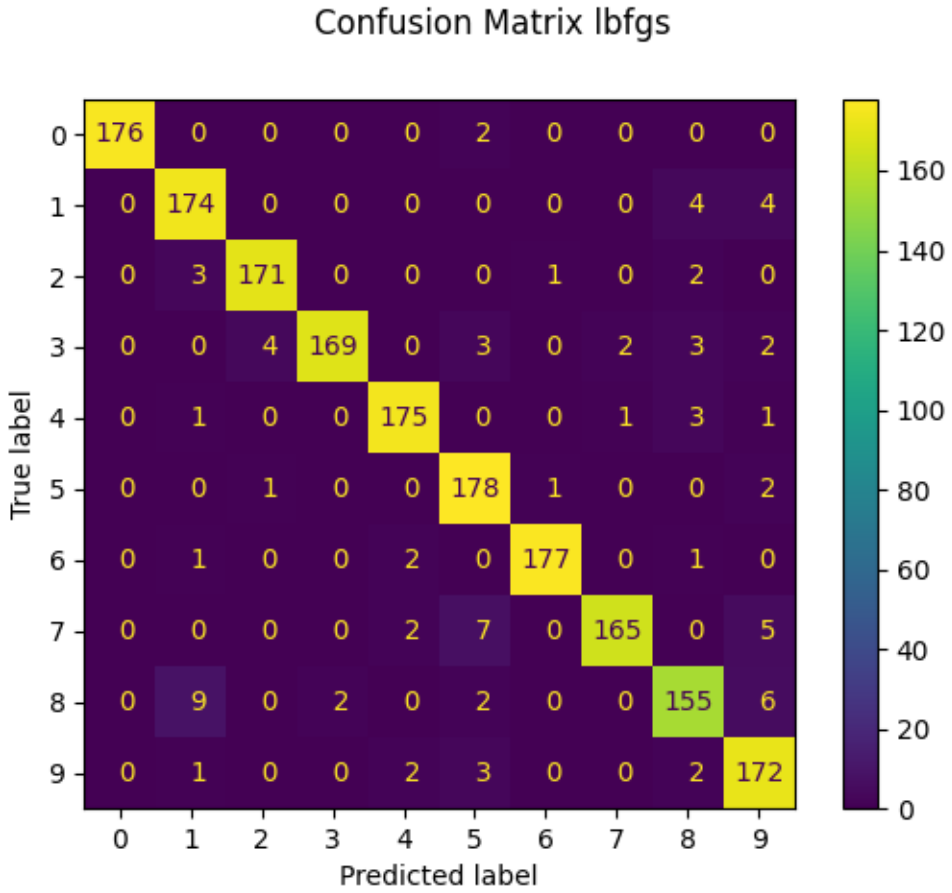
Los datos han sido estandarizados para que los estimadores de sklearn funcionen correctamente.

Las transformaciones que realizamos se le aplican primero al conjunto de training, se guardan y se replican en el conjunto de test.

1.5. Elección de la métrica a usar y discusión de su idoneidad

La métrica que vamos a utilizar es la “accuracy”, es decir, el porcentaje de aciertos. Podríamos utilizar otras métricas como “recall” o “f1-score” pero realmente los falsos positivos o negativos no nos interesan, solo queremos saber si una imagen ha sido bien clasificada o no, el hecho de que un dos se haya clasificado como un tres no es de gran utilidad, lo que nos interesa es conocer que ese dos ha sido mal clasificado. Aun así, puede ser interesante consultar otras métricas y observar la matriz de confusión para ver si hay muchas instancias de una clase que se confundan con otra clase, por ejemplo,

en la siguiente matriz podemos ver que se han clasificado siete imágenes del dígito 7 como el dígito 5 o nueve imágenes del dígito 8 como 1.



1.6. Discusión de la técnica de ajuste elegida

Sklearn nos permite utilizar cinco técnicas distintas con regresión logística: lbfgs, newton-cs, liblinear, sag y saga. De éstas cinco vamos a probar todas menos saga y nos quedaremos con el mejor de los cuatro, es decir, el que obtenga mejor “accuracy”.

1.7. Discusión de la necesidad de regularización y en su caso la función usada par ello.

Siempre es bueno usar regularización para reducir el overfitting y mejorar la generalización. En este caso lo esperable es que la regularización de Ridge, L2, sea mejor ya que los atributos están altamente correlados entre si al ser los píxeles de una imagen. Aun así, se van a realizar pruebas para ver que tipo de regularización obtiene mejores resultados.

1.8. Definición de los modelos a usar

De los modelos lineales para clasificación que hemos visto, regresión logística y el algoritmo perceptrón, nos quedamos con el primero. Nos encontramos ante un problema multiclase por lo que es

preferible utilizar regresión logística que, gracias a *softmax*, puede ser aplicado a éste tipo de problemas de una forma más sencilla. Peceptrón, aunque puede ser adaptado para casos multiclase mediante una perpectiva uno contra todos o uno contra uno, está diseñado para problemas de clasificación binaria y estaríamos complicado el problema más de lo necesario en el caso de usarlo.

1.9. Estimación de los hiperparámetros del modelo y selección del mejor modelo

Vamos a utilizar validación cruzada para encontrar los mejores hiperparámetros. Para cada técnica mencionada antes, lbfgs, newton-cs, liblinear y sag, vamos a probar a usar regularización de Lasso, de Ridge y a no utilizar ninguna de las dos. No todas las técnicas permiten todos los tipos de regularización, por ejemplo, lbfgs sólo permite L2. Además, vamos a probar distintos valores para el parámetro de regularización C, cuanto más pequeño es C mayor es la regularización. Los valores que tomará son: 0.01, 0.1, 0.2, 0.3, 0.4, 0.5 y 1. Todas las pruebas se realizan con un máximo de 100 iteraciones.

Los resultados con los mejores hiperparámetros de cada técnica son los siguientes:

Técnica	Accuracy Test
lbfgs	0.9510
newton-cs	0.9510
liblinear	0.9482
sag	0.9510

Lbfgs, newton-cs y sag obtienen los mismos resultados, mientras que liblinear es el peor de los cuatro. Nos quedamos con lbfgs ya que es el que menos tiempo tarda en ejecutarse. Los mejores hiperparámetros para éste son regularización L2, como era de esperar, y C = 0.2.

1.10. Estimación de error E_{out} utilizando validación cruzada y comparación con E_{test}

Cuando realizamos validación cruzada para ajustar los hiperparámetros obtenemos un error de validación $E_{val} = 0.03348$, podemos calcular una cota de E_{out} utilizando éste resultado:

$$E_{out}(h) \leq E_{in}(h) + \sqrt{\frac{1}{2N}\log\frac{2}{\delta}}$$

Si totamos $\delta = 0.05$ entonces tenemos

$$E_{out}(h) \leq 0.055445 \quad \text{con } 1 - \delta \text{ de probabilidad}$$

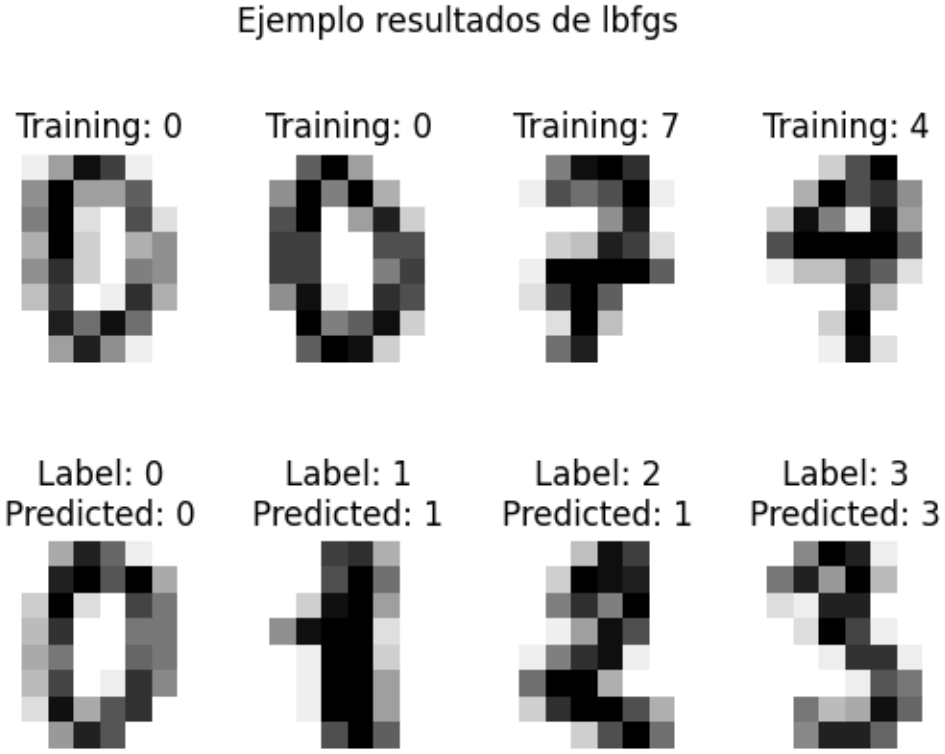
Como podemos ver en la tabla anterior la “accuracy” en el conjunto de test es 0.9510, es decir, $E_{out} = 0.048971$, que es menor que la cota que hemos calculado.

1.11. Proposición y justificación del mejor modelo

Como hemos visto, obtenemos los mismos resultados para tres de las distintas técnicas que tenemos disponibles al usar regresión logística. De esas tres hemos decidido quedarnos con la técnica lbfgs porque es la que tarda menos tiempo en ejecutarse.

Técnica	Tiempo (segundos)
lbfgs	7.8770
newton-cs	17.1148
liblinear	20.6881
sag	48.5939

Tenemos un modelo que obtiene buenos resultados, como ya hemos podido ver, la cota que hemos calculado de E_{out} es pequeña, el modelo representa de manera adecuada los datos.



2.Communities and Crime Data Set

1.1. Compresión del problema a resolver

Nos encontramos ante un problema de aprendizaje supervisado en el que nos dan datos socio-económicos de distintas comunidades de Estados Unidos y a partir de éstos se pretende estimar el número de crímines por cada cien mil habitantes.

1.2. Selección de la clase de funciones a usar

Se ha decidido no realizar ninguna transformación de las variables originales ya que el problema es suficientemente sencillo y se obtienen buenos resultados con la clase de funciones lineales.

1.3. Definición de los conjuntos de training y test

Separamos los datos en un conjunto de training, con 80% de los datos, y en un conjunto de test, con el 20% restante de los datos.

1.4. Preprocesado de datos

En este problema tenemos valores perdidos que debemos tratar. En nuestro conjunto de training ocurre que 1341 de las 1595 muestras, es decir, un 84% de los datos, tiene valores perdidos. Al ser un conjunto tan grande de las muestras no podemos eliminarlo por completo, tenemos que tomar otras medidas. De media, a éstas muestras les falta información de 22 de sus atributos, un 18% del total de las características.

Vamos a sustituir los valores perdidos por la media de cada variable. Éstas medias las guardamos para sustituir los valores perdidos del conjunto de test cuando vayamos a calcular el E_{out} una vez ajustados los hiperparámetros.

Además, los datos vienen normalizados y han sido estandarizados para que las funciones de sklearn funcionen correctamente.

Todas las transformaciones que realizamos se le aplican primero al conjunto de training, se guardan y se replican en el conjunto de test.

1.5. Elección de la métrica a usar y discusión de su idoneidad

Como función de pérdida vamos a utilizar el error cuadrático medio. En general es una buena métrica para cualquier problema de regresión ya que penaliza cualquier error dando mayor importancia a errores grandes.

1.6. Discusión de la técnica de ajuste elegida

Vamos a utilizar SGD como técnica de ajuste ya que ésta nos permite ajustar más hiperparámetros que el algoritmo de la pseudoinversa y así poder conseguir un ajuste mejor. SGD nos da muchas más flexibilidad para encontrar un estimador que se ajuste mejor a los datos.

1.7. Discusión de la necesidad de regularización y en su caso la función usada par ello.

Siempre es bueno usar regularización para reducir el overfitting y mejorar la generalización. En este caso lo esperable es que la regularización de Ridge, L2, sea mejor ya que según la página de donde se han obtenido los datos:

"unrelated attributes were not included"

así que, en un principio, los datos deberían estar correlados entre si. Aun así, vamos a hacer pruebas con los dos tipos de regularización para ver con cuál se obtienen mejores resultados.

1.8. Definición de los modelos a usar

Vamos a utilizar regresión lineal para resolver éste problema. Nos encontramos ante un problema de regresión y se nos ha pedido que ajustemos un modelo lineal, no tenemos más opciones para resolverlo.

1.9. Estimación de los hiperparámetros del modelo y selección del mejor modelo

Utilizamos validación cruzada para encontrar los mejores hiperparámetros. Vamos a probar los dos tipos de regularización, L1 y L2. La constante de regularización, α, tomará los valores: 0.01, 0.1, 0.5, 1 y 10. Cuanto mayor es α mayor es la regularización.

Queremos ver también como afecta el método de actualización del learning rate en la convergencia del SGD por lo que vamos a probar cuatro formas distintas: constante, "optimal", "invscaling" y "adaptive". También, comprobamos como afecta el valor inicial del learning rate, que será: 0.0001, 0.001, 0.01 y 0.1.

Los hiperparámetros del modelo que obtiene el menor E_{val} son los siguientes:

- $\alpha = 0.1$
- lr = 0.1
- Método de actualización del learning rate, "adaptive"
- Regularización, L2 como era de esperar.

1.10. Estimación de error E_{out} utilizando validación cruzada y comparación con E_{test}

Cuando realizamos validación cruzada para ajustar los hiperparámetros obtenemos un error de validación $E_{val} = 0.018185$. Podemos utilizar E_{val} como una aproximación de E_{out}. Si lo comparamos con el error que obtenemos al predecir el resultado del conjunto de test, $E_{test} = 0.020808$, vemos que son muy similares. Tenemos una buena aproximación del error fuera de la muestra.

1.11. Proposición y justificación del mejor modelo

Hemos conseguido un modelo que representa de manera adecuada los datos como podemos observar en los resultados ya mencionados, tenemos una buena aproximación del E_{out} que además es muy bajo.

Gracias a la validación cruzada hemos obtenido el mejor modelo, entre los que hemos probado. Los resultados son suficientemente buenos como para no necesitar probar más modelos.