Universidad de Granada
Escuela Técnica Superior de Ingeniería Informática y de Telecomunicación
Grado en Ingeniería Informática
Departamento de Ciencias de la Computación e Inteligencia Artificial
Aprendizaje Automático
Curso 2019/2020
Grupo A2
Javier Rodríguez Rodríguez

Trabajo 3: Ajuste de Modelos Lineales

Optical Recognition of Handwritten Digits:

1. **Comprensión del problema**

   Nos encontramos ante un problema de clasificación. El conjunto de datos X está compuesto por una serie de tuplas de 64 valores enteros en el rango 0...16, obtenidos y preprocesados a partir de mapas de bits de dígitos escritos a mano por un conjunto de 43 personas, donde cada uno de estos 64 valores corresponde a la medición de determinada propiedad del dígito escrito, como lo son simetría, intensidad, entre otros.

   A cada tupla de X se le asigna un valor y en el rango 0...9 que representa el valor del dígito escrito originalmente por la persona.

   Ante estos datos, tenemos un problema de aprendizaje supervisado de clasificación multiclase, donde ante una tupla de datos que describan a un dígito escrito a mano, sea posible determinar qué valor representa, es decir, qué valor entero entre el 0 y el 9 corresponde con las propiedades del mapa de bits del trazo especificado, mediante la función ajustada según los datos de entrenamiento etiquetados provistos al algoritmo.

2. **Clases de funciones**

   Utilizaremos funciones lineales para este problema: son las más sencillas de utilizar e interpretar, existen múltiples implementaciones de métodos que ajustan esta clase de funciones y se adecúan correctamente al problema.

3. **Training, Test y Validación**

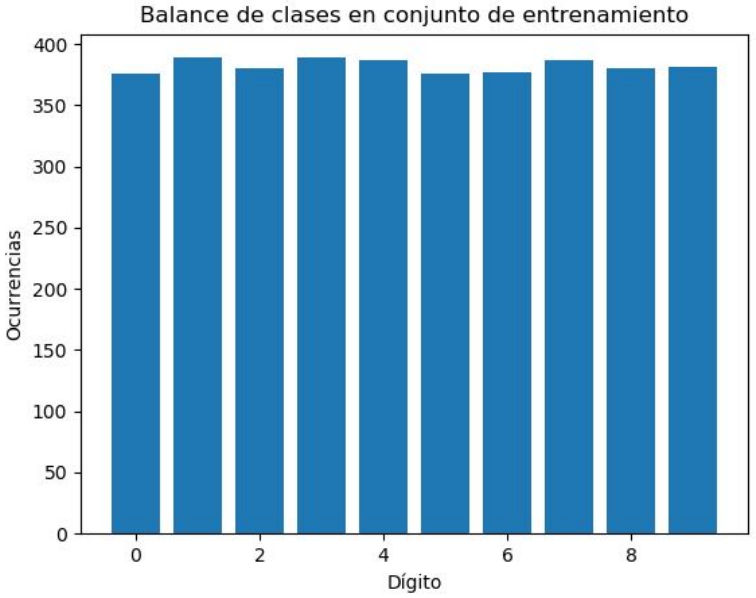
   Los conjuntos Training y Test vienen prefabricados de la fuente de los datos. En este caso, el conjunto Training contiene 3823 instancias mientras que el conjunto Test contiene 1797 instancias, resultando que los datos se reparten aproximadamente en 68% Training y 32% Test.

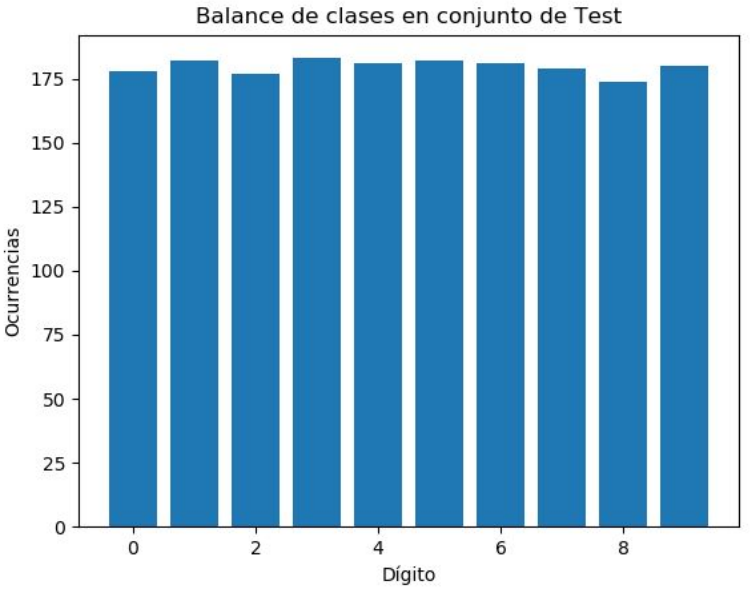
   Para la validación utilizaremos el método 5-Folds implementado en Scikit-Learn, por lo que no es necesario separar de antemano un conjunto de Validación.

4. **Preprocesamiento**

Se indica en la fuente que los datos están previamente preprocesados, de forma que no hay valores faltantes y que la asignación de valores enteros entre 0 y 16 a los atributos ya reduce la dimensionalidad y la variabilidad ante distorsiones.

Para asegurar que las clases estén balanceadas, graficamos la ocurrencia de cada clase dentro del conjunto de training y de test.





Como vemos, en ambos conjuntos las clases están bastante balanceadas, por lo que no hace falta ningún tipo de preprocesamiento adicional.

5. **Métrica**

Como las clases están balanceadas, podemos utilizar la precisión del modelo como métrica para su elección, es decir, la tasa de aciertos del conjunto de predicciones realizadas.

6. **Técnica de ajuste**

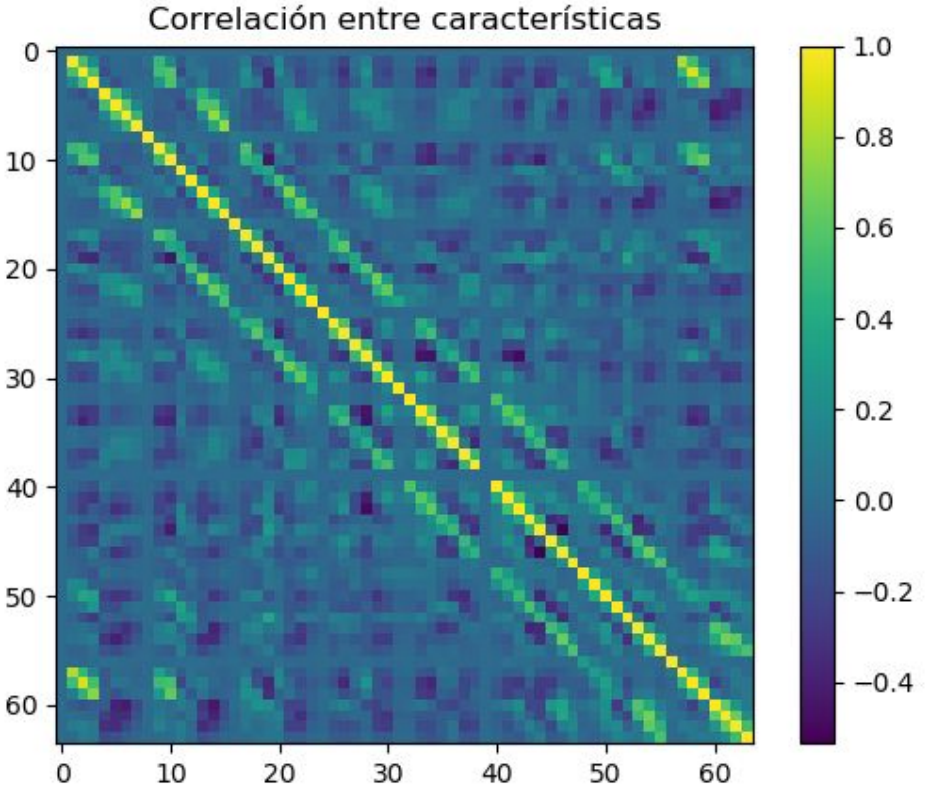
Utilizaremos dos técnicas, el algoritmo SAGA y SGD. El algoritmo SAGA lo utilizamos porque viene implementado dentro del módulo de Regresión Logística (uno de los modelos considerados) y, según la documentación de este, proporciona muy buenos resultados. Además, soporta regularización l1 y l2, a diferencia de la mayoría de técnicas implementadas en el módulo, y además soporta clasificación multiclase nativamente.

SGD lo utilizamos por ser un algoritmo conocido, con rendimiento reconocido como adecuado, que también viene implementado en Scikit-Learn y ajusta varios modelos, como los son Regresión Logística o

Perceptron. Sin embargo, no soporta clasificación multiclase nativamente, por lo que utiliza una estrategia One vs Rest para separarlo en múltiples casos binarios.

7. **Regularización**

Calculamos la correlación entre las características de los datos, para determinar así qué tipo de regularización usar. Graficamos la matriz de correlaciones en forma del siguiente mapa de calor:



Como vemos, la mayor parte de los datos tienen muy baja correlación, siendo los valores con alta correlación la diagonal, que no tiene ninguna relevancia (un dato siempre estará altamente correlado consigo mismo). Por esto, optamos por regularización Lasso (L1) para este problema.

Ciertos valores en el cálculo devuelven NaN. Esto se debe a que el coeficiente de correlación para una característica constante (varianza 0) no está definida. He decidido reemplazarlos por 0s por legibilidad, pues de estar correlada con otra característica dejarían de ser constantes.

8. **Modelos**

Dada la naturaleza del problema, el modelo adecuado parece ser Regresión Logística, pues determina la probabilidad de, dados valores para una serie de características, predecir una clase, que es exactamente el problema en cuestión. Además, tenemos dos técnicas a disposición que ajustan este modelo (SAGA y SGD)

Sin embargo, dado que la implementación de SGD soporta el modelo

Perceptron, también lo contemplamos como modelo para así comparar resultados y llegar a conclusiones de interés.

9. **Hiperparámetros y selección de modelo**

En cuanto a los hiperparámetros, probamos varios valores de alfa para SGD y C para SAGA, en busca de determinar aquellos que den mejores resultados.

Para realizar los experimentos y determinar la calidad de cada uno utilizaremos validación cruzada a través de 5-fold estratificado. Esta técnica subdivide los datos de entrenamiento en 5 conjuntos balanceados aleatorios, donde 4 se utilizan para el entrenamiento del modelo con los hiperparámetros probados y el 5to se utiliza como conjunto test para valorar su rendimiento.

Finalmente, probamos las distintas técnicas de ajuste mediante la validación cruzada descrita anteriormente sobre el conjunto Test, obteniendo así los siguientes resultados:

Algoritmo	**Precisión$_{Test}$**
SAGA(Regresión Logística)	0.9571508069003896
SGD(Regresión Logística)	0.9443516972732332
SGD(Perceptrón)	0.9343350027824151

Seleccionaremos aquel que de la máxima precisión, Regresión Logística a través de la técnica SAGA, como nuestro mejor modelo.

10. **Estimación del error**

El error que se produce al ajustar el modelo no es más que 1-Precisión de dicho modelo, dado que fue la métrica elegida para este problema. Recordemos que la Precisión es la tasa de aciertos entre el total de predicciones, mientras que el Error es la tasa de fallos entre el total de predicciones.

Por tanto, E_{Out} = 0.04284919309

11. **Justificación**

El modelo representa adecuadamente los datos. No se preprocesaron las características más del preprocesamiento inicial de la fuente, y la regularización respeta su correlación.

La calidad del modelo sin duda es buena, con un error por debajo del 5% fuera de la muestra. Además, aún utilizando SGD, su error para el caso de Regresión Logística es igualmente cercano al 5%, apoyando así la validez del modelo.

Dentro de los experimentos realizados, el error obtenido es el mejor. La Regresión Logística, con ambas técnicas, da mejor error que el modelo Perceptrón. Sin embargo, esto está limitado a los modelos lineales considerados, y es posible que ajustando otro modelo de un error mejor.

La principal razón de elegir estos modelos es que ofrecen un compromiso ideal entre facilidad de uso y parametrización, pues sus implementaciones en Scikit-Learn no solo son efectivas, sino que permiten configurar distintos hiperparámetros con sencillez, son fáciles de interpretar y toman muy poco tiempo en ajustar. Además, están diseñadas con la posibilidad de utilizar los módulos de validación cruzada sobre ellas, ofreciendo así una capa adicional de configuración.

Precisamente estas razones fueron las que invitaron a utilizar el modelo Perceptrón, aún ya decididos a utilizar Regresión Logística. El costo a nivel computacional en el experimento adicional es mínimo, pero proporciona certeza sobre nuestra suposición inicial.

Communities and Crime:

1. **Comprensión del problema**

   Tenemos una base de datos que, para un conjunto de ciudades de Estados Unidos, se tienen 128 métricas sociales, numéricas y no numéricas, y la cantidad de crímenes violentos per cápita, la variable que se desea predecir. Como tenemos este valor para las 1994 ciudades del conjunto, nos encontramos ante un problema de aprendizaje supervisado. Además, como lo que se pretende predecir es un valor numérico, y no una clase, el problema en cuestión es un problema de regresión. La idea es encontrar una función tal que, dada una lista de métricas sociales, poder predecir con precisión cuántos crímenes violentos per cápita se comiten.

   Importante destacar el hecho de que todas las variables numéricas están normalizadas de forma tal que sean un valor real en el intervalo 0...1 a excepción de las primeras cinco categorías, que a pesar de ser numéricas, su valor es simplemente identificativo a nivel geográfico o es residuo del proceso de extracción de datos. Estas cinco categorías no son predictoras.

2. **Clases de funciones**

   Nuevamente, tilizaremos funciones lineales para este problema: son las más sencillas de utilizar e interpretar, sus resultados son adecuados, si no ideales, como quedó probado en el dataset anterior, y se adecúan correctamente al problema.

3. **Training, Test y Validación**

   En este caso, los datos vienen dados en un único conjunto de datos, por lo que es nuestra responsabilidad construir un conjunto de Training y uno de Test. La cantidad de instancias es notablemente menor al dataset anterior (1994 vs 5620), por lo que existe el temor que, de utilizar porcentajes similares de separación, resulten muy pocas instancias para el entrenamiento. Por esta razón consideramos 1594 instancias en Training y 400 en Test, para una repartición de aproximadamente 80% Training y 20% Test.

   Para la validación utilizaremos nuevamente el método 5-Folds implementado en Scikit-Learn, ahorrando la necesidad de generar a este punto un conjunto de validación

4. **Preprocesamiento**

   Primero que nada, no todas nuestras variables son numéricas. De hecho, una sola no lo es. Por suerte, esta característica pertenece al conjunto de cinco características no predictoras del dataset mencionadas cuando lo estudiamos, y por tanto, lo primero que haremos será eliminarlas.

   Por tanto, las únicas variables que quedan son aquellas características numéricas ya normalizadas dentro del intervalo 0...1. Sin embargo, varias presentan valores faltantes. Reemplazamos los valores faltantes por el promedio de los valores existentes en cada categoría, con el fin de permitir el proceso de aprendizaje.

5. **Métrica**

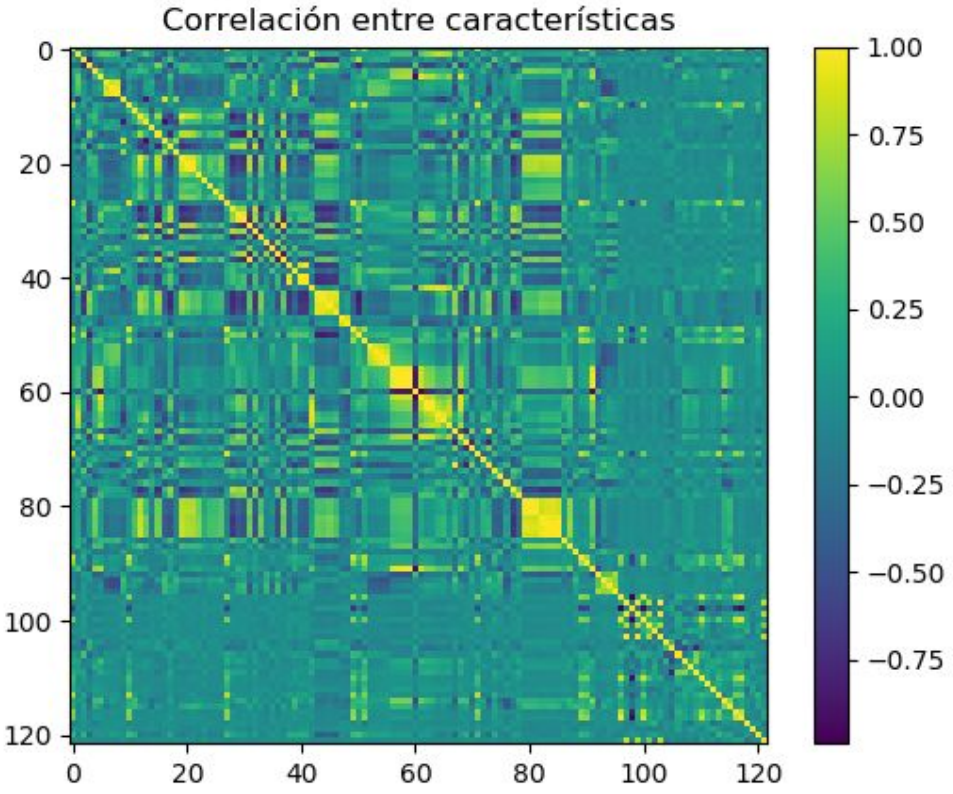
   Dado que estamos ante un problema de regresión, la métrica a utilizar será el Error Cuadrado Medio (MSE), donde se mide el promedio de las distancias cuadradas entre cada dato y el valor de la función para ese punto.

6. **Técnica de ajuste**

   Utilizaremos dos técnicas, la técnica de Mínimos Cuadrados Ordinarios y el algoritmo SGD. Nuevamente, la razón de su elección es que son técnicas probadas y conocidas, implementadas dentro del módulo Scikit-Learn y que permiten probar distintos parámetros y contrastar resultados.

7. **Regularización**

   Calculamos la correlación entre las características de los datos, para determinar así qué tipo de regularización usar. Graficamos la matriz de correlaciones en forma del siguiente mapa de calor:



A diferencia del mapa de calor del dataset anterior, en este notamos que existen múltiples zonas amarillas separadas de la diagonal, espacios donde la correlación entre esas características es alta. Por esto, utilizar regularización Ridge (L2) resulta la opción correcta.

8. Modelos

Considerando que se trata de un modelo lineal para resolver un problema de regresión, el modelo inmediato es Regresión Lineal, y por tal razón, el que utilizaremos. Precisamente es la Regresión Lineal el modelo que aproxima una variable según el valor de múltiples variables adicionales, que es el problema en cuestión.

9. Hiperparámetros y selección de modelo

Para el problema de regresión no surgen nuevos hiperparámetros de interés, por lo que nuevamente nos limitamos a evaluar el rendimiento de los algoritmos para distintos valores de alfa. Notable destacar que, dado que estamos ante un tipo de regularización distinto, los valores a evaluar de alfa van de 0,1 a 10, a diferencia los valores entre 0,0001 a 0,1 del dataset anterior. Al igual que antes, estos experimentos los realizamos mediante validación cruzada con 5-fold estratificado. Recordemos que esta técnica subdivide los datos de entrenamiento en 5 conjuntos balanceados aleatorios, donde 4 se utilizan para el entrenamiento del modelo con los hiperparámetros probados y el 5to se utiliza como conjunto test para valorar

su rendimiento.

Finalmente, probamos las distintas técnicas de ajuste mediante la validación cruzada descrita anteriormente sobre el conjunto Test, obteniendo así los siguientes resultados:

Algoritmo	**Error$_{Test}$**
OLS (Regresión Lineal)	0.01771997867567909
SGD (Regresión Lineal)	0.018363994032323135

Seleccionaremos aquel que dé el menor error, Regresión Lineal a través de SGD con los mejores hiperparámetros.

10. Estimación del error

Nuestra métrica usada es directamente el error fuera de la muestra (conjunto Test). Así que la estimación del error es directa.

E_{Out} = 0.019583054399181742

11. Justificación

El modelo parece representar sumamente bien los datos. El preprocesamiento inicial de la fuente permite un ajuste adecuado por parte del algoritmo, y limitarnos a los datos predictores evita contemplar datos categóricos de poco a nulo interés. La regularización es acorde a la correlación vista en los datos, y sustituir los valores faltantes por la media de los existentes es un compromiso balanceado que da lugar a buenos resultados.

En cuanto a esto, los resultados son más que suficientes, teniendo un error fuera de la muestra por debajo de 0.02. Esta vez utilizamos un solo modelo, por lo que tiene sentido que ambos algoritmos que lo ajustan den resultados similares. Por esta razón, queda probado que el modelo es válido para representar este problema.

Difícilmente podemos determinar si nuestro modelo es el que proporciona el mejor error, siendo el único que consideramos, pero dado lo bueno que son los resultados, es una conclusión lógica pensar que este modelo, por lo menos dentro de los lineales, es de los que proporciona el mejor error.

La razón por la cual escoger este modelo es que es el único modelo lineal estudiado para problemas de regresión, y por una buena razón. Es intuitivo y eficiente, y Scikit-Learn nos provee de la implementación de varias técnicas que ajustan adecuadamente este modelo, con posibilidad de regularizar y aplicar distintos hiperparámetros para comprobar resultados de forma sencilla.