Universidad de
SanAndrés

I302 - Aprendizaje Automático y Aprendizaje Profundo

1er Semestre 2024

Trabajo Práctico 2

Fecha de entrega: Miércoles 27 de marzo, 23:59 hs.

Formato de entrega: Los archivos desarrollados deben ser entregados en un archivo comprimido .zip a través del Campus Virtual, utilizando el siguiente formato de nombre de archivo: *Apellido_Nombre_TP2.zip*. Se aceptará únicamente 1 archivo por estudiante. En caso de que el nombre del archivo no cumpla con la nomenclatura especificada, el trabajo no será corregido.

La carpeta comprimida deberá constar de N sub-carpetas, una por cada problema del TP (es decir, cada problema tiene su sub-carpeta denominada "Problema N"). Dentro de cada sub-carpeta deberán incluir un Jupyter Notebook (archivo *.ipynb*) en el cual se den las respuestas a los incisos del problema y se muestren los gráficos resultantes. Pueden agregar en el Jupyter Notebook mas resultados o análisis de los expresamente solicitados en los incisos, si lo consideran adecuado. Junto con el Jupyter Notebook podrá haber uno o mas archivos de Python que contengan código que sea parte de la resolución del problema. Dicho código debe seguir las buenas prácticas de programación y modularización que se presentaron en clase.

---

1. **Máxima verosimilitud y ajuste polinómico.** El archivo *toy_dataset.pkl* contiene pares de muestras (X, y) independientes e identicamente distribuidas donde X es el conjunto de variables de entrada (en inglés se le dice "input features", o mas apropiadamente lo que en clase llamamos "raw input features"), e y es el conjunto de valores objetivos (lo que en inglés muchas veces se conoce como "target variable").

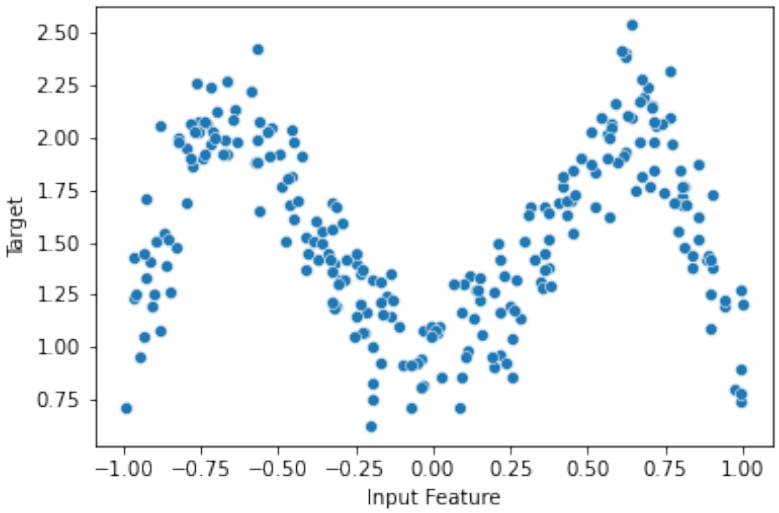

Figure 1: Dispersion de valores objetivos con respecto a las features de entrada

Se busca ajustar un modelo de regresión polinómico $\hat{y}(x, \mathbf{w})$ de grado M:

$$\hat{y}(x, \mathbf{w}) = \sum_{j=0}^{M} w_j \phi_j(x) \tag{1}$$

donde M es el grado máximo del polinomio y $\phi_j(x) = x^j$.

(a) Derivar las ecuaciones para determinar los parámetros óptimos $\mathbf{w}^*$ aplicando el principio de máxima verosimilitud, asumiendo que las muestras son independientes e identicamente distribuidas con un ruido Gaussiano $\mathcal{N}(0, \sigma^2)$. Demostrar que $\mathbf{w}^*$ es el óptimo global, que maximiza globalmente la verosimilitud (probabilidad conjunta) de los datos de entrenamiento.

(b) Seleccionar el valor de M más adecuado, entrenando modelos con distintos valores de M y evaluando el error de validación (o validación cruzada, si quisieran). Justificar su decisión mostrando curvas de error de entrenamiento y validación.

(c) Derivar las ecuaciones para determinar los parámetros óptimos $\mathbf{w}^*$ aplicando el principio de máxima verosimilitud, pero en este caso asumiendo además que la incertidumbre en el vector $\mathbf{w}$ está dado por la distribución de probabilidad:

$$P(\mathbf{w}) = \mathcal{N}(\mathbf{w}_0, s^2 I) \tag{2}$$

Donde $\mathbf{w}_0$ es un vector constante. Es decir, la distribución a-priori (antes de ver los datos) de $\mathbf{w}_0$ es una Gaussiana de media $\mathbf{w}$ y co-varianza $s^2 I$.

(d) Reentranar el modelo de regresión polinómica con $M = 20$ y seleccionar un valor adecuado para el parámetro de regularización λ, donde $\lambda = \frac{\sigma^2}{s^2}$. Mostrar las curvas de error de entrenamiento y validación que justifiquen su decisión.

2. **Regresión lineal.** Los datos presentes en el corpus de datos (conjunto de datos) *Student Performance* se recolectaron para estudiar cuáles son los factores mas influyentes en el desempeño de un grupo de estudiantes en sus exámenes finales, con el objetivo de desarrollar un modelo predictivo que pueda estimar cuál será el desempeño de un estudiante en sus exámenes finales a partir de conocer cuantas horas estudia, cuales son sus calificaciones previas, cuantas horas duerme y cuantos ejercicios de repaso logró resolver. De este conjunto de datos se realizó previamente una división de datos de desarrollo y de testeo, de entre los cuales usted solo tiene acceso al subset de desarrollo en el archivo *Student_Performance_DEV.csv*.

   (a) Implementar un modelo de regresión lineal con regularización L-2, que estime el desempeño de un estudiante en base a los datos existentes.

   (b) Ajustar el hiperparámetro de regularización (la constante λ) utilizando cross-validation. Deberán mostrarse los gráficos de error de entrenamiento y error de cross-validation para distintos valores del hiperparámetro.

   (c) Reportar el RMSE, MAE y R^2 del modelo desarrollado (es decir, con los parámetros e hiperparámetros óptimos) evaluados mediante cross-validation. Además, graficar $\hat{y}$ vs. y para los datos "held out" de cada uno de los folds de cross-validation. Dicho gráfico debe tener una recta de pendiente 1 que pase por el origen, para poder visualizar mejor las desviaciones de las estimaciones del modelo vs. los datos reales.

   NOTA: Su modelo será evaluado por los docentes sobre el test set, que no le ha sido dado.

3. **Redes Multicapa para Regresión.** En este problema abordaremos el mismo set de datos que en el problema anterior, pero en este caso con una red neuronal multicapa densa ("fully connected"). Esta arquitectura de modelo tiene como hiperparámetros la cantidad de capas ocultas L (profundidad de la red) y la cantidad de unidades ocultas en la capa l (ancho de la capa l) $M^{(l)}$.

   (a) Implementar una red neuronal multicapa, que funcione para cualquier valor $L \geq 1$, y $M^{(l)} \geq 1$, así como el algoritmo de optimización de descenso por gradiente y backpropagation para optimizar los pesos de las unidades ocultas, asumiendo una "loss-function" que sea la suma de los errores cuadráticos.

   (b) Implementar descenso por gradiente estocástico y graficar la evolución del error de entrenamiento en función de las épocas.

(c) Implementar descenso por gradiente con mini-baches, y graficar la evolución del error de entrenamiento en función de las épocas, para distintos tamaños de baches. Deberás elegir a tu criterio el tamaño de los mini-batches.

(d) Utilizando el set de desarrollo y la metodología de cross-validation, seleccionar los hiperparámetros de la red. Para esto puede usar el algoritmo de gradiente descendiente que usted desee (normalmente, uno querrá usar el optimizador que mejor resultado le haya dado, en cuanto a performance del modelo entrenado y el tiempo de convergencia).

Sugerencia: si bien uno puede hacer una búsqueda exhaustiva sobre el espacio de hiperparámetros L y $M^{(l)}$, esto puede resultar computacionalmente demasiado costoso, debido a que para cada combinación y cada fold, se de debe re-entrenar el modelo y evaluarlo. Piense si hay alguna manera de recorrer el espacio de búsqueda con menor costo computacional. Además, recuerde que dados dos modelos de performance similar, se privilegia elegir el de menor complejidad (menor cantidad de parámetros), porque tenderá a tener menor varianza en el error predictivo sobre data sets "nuevos" (es decir, que no hayan sido usados para el deasrrollo del modelo).

(e) Analizar la performance de su modelo final en términos de RMSE, MAE y R^2, y graficar $\hat{y}$ vs. y sobre los datos "held out" de los folds de cross-validaion. Compare esto con la performance del modelo de regresión lineal desarrollado en el problema anterior. ¿Cuál de los dos modelos cree usted que generalizará mejor? ¿Por qué?

NOTA: Al igual que en el problema anterior, su modelo será evaluado por los docentes sobre el test set, que no le ha sido dado.

4. **Ejercicios opcionales**: Para quienes quieran profundizar un poco más en el entrenamiento de redes multicapa, se proponen los siguientes experimentos:

(a) Agregar regularización L-2 al proceso de enrenamiento de la red neuronal, y observar el efecto que esto tiene sobre la performance del modelo.

(b) Implementar el algoritmo de optimización ADAM, con gradiente descendiente por mini-batches. Graficar la evolución del error de entrenamiento en función de las épocas para distintos valores de los hiper-parámetros γ_v, γ_s, etc.

(c) Implementar un *learning rate scheduler* (sin ADAM), el cual consiste en un learning rate variable a medida que transcurre el entrenamiento, en lugar de utilizar un valor de learning rate constante. Explorar las siguientes variantes:

- Linear decay
- Power law
- Exponential decay

Sugerencia: Ver el Capítulo 7.3.2. de Bishop (2023), para una definición de cada uno de estos schedules.