

Preguntas de ML

Final de Matemática Discreta II

FaMAF 2024

- ¿En qué tipo de problemas es adecuado utilizar aprendizaje automático?.....2
- ¿Cuál es la diferencia entre la función de costo o pérdida, y las métricas de rendimiento del modelo?..... 2
- ¿Cómo se puede entrenar un modelo utilizando features o características no numéricas?..... 2
- Explique por qué se intenta minimizar la función de costo utilizando un método iterativo (numérico) y no un método analítico minimizando la derivada de la función..... 2
- ¿Por qué el algoritmo SGD se llama “estocástico”?.....2
- ¿Qué impacto tiene el learning rate en la optimización de la función?.....2
- Si una red tiene muy buen rendimiento durante el entrenamiento, pero muy bajo rendimiento en un conjunto de datos de evaluación, ¿qué acciones tomaría para mejorar el modelo?..... 3
- ¿Por qué inicializamos los pesos de un modelo en valores aleatorios? ¿Qué pasaría si los inicializamos todos en cero? ¿Y todos en uno?..... 3
- ¿Por qué el dropout puede ser considerado una técnica de regularización?... 3
- ¿Por qué está recomendada la estandarización o escalado de los valores de entrada de un modelo como la regresión lineal o el mlp?..... 3

- **¿En qué tipo de problemas es adecuado utilizar aprendizaje automático?**

Es adecuado utilizar aprendizaje automático en problemas de **detección/predicción de comportamientos/valores** los cuales trabajan con grandes bases de datos, como la clasificación de imágenes, los algoritmos de recomendación de las grandes redes sociales, etc. También, los problemas en los que queremos **no determinismo** como la generación de texto/imagen/video que con un input nos den más de un resultado.

- **¿Cuál es la diferencia entre la función de costo o pérdida, y las métricas de rendimiento del modelo?**

La función de costo, efectivamente **nos da un valor de “rendimiento”** de cada paso del modelo, **pero** tiene la propiedad de que **es fácil de derivar** pues se deben hacer muchas derivadas de la misma para actualizar los costos en cada iteración, y **no queremos** que sea una **carga computacional extrema**.

- **¿Cómo se puede entrenar un modelo utilizando features o características no numéricas?**

Se deben **convertir a features numéricas** con **métodos de codificación** como **one-hot encoding** o directamente **binary encoding** donde a cada label no numérico se le asocia (biyectivamente) un valor numérico.

- **Explique por qué se intenta minimizar la función de costo utilizando un método iterativo (numérico) y no un método analítico minimizando la derivada de la función**

La función de costo se minimiza de manera iterativa pues en **modelos complejos** como las redes neuronales, **minimizar la derivada sería algo muy costoso computacionalmente** mientras que los **métodos iterativos** nos brindan **suficiente precisión** para mejorar.

- **¿Por qué el algoritmo SGD se llama “estocástico”?**

El algoritmo **SGD** o Descenso Estocástico por el Gradiente, se llama **“estocástico”** por su **no determinismo**, es decir que tiene características que no siempre son las mismas, como **por ejemplo la inicialización de parámetros aleatoria**, la utilización de **batches**, y porque **tampoco garantiza determinísticamente llegar a un mínimo** de cualquier función **para un learning rate dado**.

- **¿Qué impacto tiene el learning rate en la optimización de la función?**

El learning rate es la **velocidad a la que la función de pérdida (idealmente) converge** a un mínimo local. Un learning rate **muy chico**, asegura la convergencia pero **muy lenta**, es decir que se requieren muchas iteraciones para llegar a valores cercanos al mínimo. Un learning rate **muy grande**, puede quedarse **oscilando** alrededor de un mínimo, o peor aún, causar **comportamiento divergente**. Lo **ideal** suele ser un **learning rate adaptativo**, que se vaya achicando a medida que nos movemos en la dirección correcta.

- **Si una red tiene muy buen rendimiento durante el entrenamiento, pero muy bajo rendimiento en un conjunto de datos de evaluación, ¿qué acciones tomaría para mejorar el modelo?**

Esto significa que hay **overfitting** i.e que el modelo se adapta muy precisamente a los datos de entrenamiento y eso causa que no sea extensible a datos por fuera de ese conjunto. Para intentar solucionarlo, se puede **reducir la cantidad de features, eliminando features redundantes o innecesarias** o también se puede hacer algún tipo de **regularización** de los datos para limitar la complejidad del modelo.

- **¿Por qué inicializamos los pesos de un modelo en valores aleatorios? ¿Qué pasaría si los inicializamos todos en cero? ¿Y todos en uno?**

La idea de inicializar los pesos del modelo en valores aleatorios es para **intentar converger a distintos mínimos locales** de la función, **idealmente** obteniendo un conjunto de valores aleatorios para los cuales la función converge a un **mínimo global**. Inicializarlos **todos en un único valor no garantiza nada** y reduce la probabilidad de converger a un mínimo global. Peor aún, en una red neuronal donde **todos los pesos son iguales**, todas las neuronas de cada capa van a recibir los **mismos valores desde una capa anterior**. Por lo tanto **todas las neuronas** de cada capa **contribuirán de la misma forma al valor final** lo que implica que el **algoritmo de backpropagation** también **actualizará todos los valores a lo mismo**.

- **¿Por qué el dropout puede ser considerado una técnica de regularización?**

El dropout “**desactiva**” **neuronas** durante el entrenamiento del modelo. Esto es para evitar que su peso (importancia) crezca demasiado, permitiendo a las otras neuronas adaptarse también al modelo, y evitando que el resultado dependa de conexiones específicas. Como consecuencia, se reduce el **overfitting** y por ello puede ser considerado una **técnica de regularización**. Sería comparable a **eliminar features**, pero **de manera temporal** y aleatoria.

- **¿Por qué está recomendada la estandarización o escalado de los valores de entrada de un modelo como la regresión lineal o el mlp?**

Está recomendado estandarizar o escalar los valores de entrada ya que si todos los valores están en **escalas distintas**, los valores en **escalas más grandes** podrían contribuir de más al modelo, **adueñándose de los resultados** y **dificultando la convergencia** de los modelos. Estandarizando, serán los **pesos** de los features los que efectivamente **determinen su importancia**, y estos **se irán adaptando**, en lugar de los **valores** que son **constantes**. Esto último significa que podemos **interpretar los pesos**. Por ejemplo, si vemos que un peso es el doble que otro, podemos concluir que la primera feature es mucho más significativa para la predicción que la segunda.