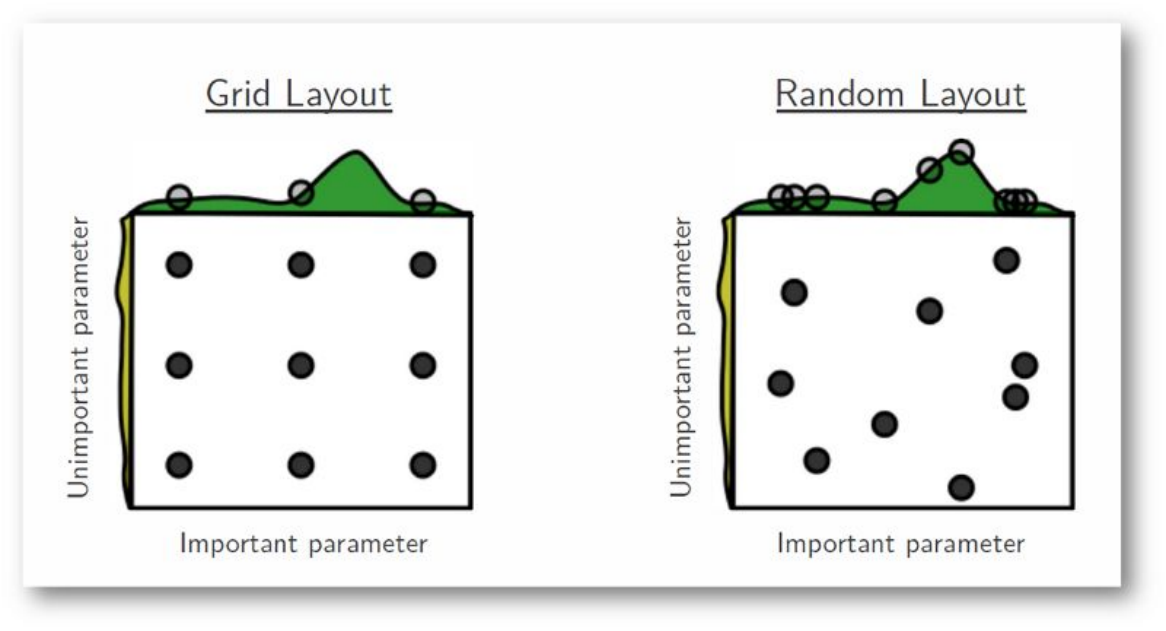

DigitalHouse > Coding School

DATA SCIENCE

UNIDAD 2
MÓDULO 4

Grid Seach - Tuneo de Hiperparámetros

Octubre 2017

Evaluación Avanzada de Modelos

1. Conocer a GridSearch: un método para experimentar sobre hiper-parámetros

2. Comparar GridSearch y RandomSearch, dos aproximaciones a la búsqueda de hiperparámetros óptimos

3. Implementar GridSearch de sklearn para autoajustar un modelo

Regresión lineal

Calcula los parámetros ideales para minimizar la suma de los errores al cuadrado.

- **Estandarización**: Los coeficientes dan cuenta perfectamente de cualquier unidad en la que estén expresados los datos. Por lo tanto, estandarizar o reescalar no cambia los resultados

- **Variables categóricas**: Admite. Hay que crear k-1 variables dummy por cada categórica para evitar la multicolinealidad.

- **Hiperparámetros**: No tiene

Regresiones Ridge, Lasso, Elastic Net

Calcula los parámetros ideales para minimizar la suma de los errores al cuadrado al mismo tiempo penalizando el valor de los Betas, usando la norma 1 del vector (Lasso) o la norma 2 (Ridge) o ambas (Elastic Net).

- **Estandarización**: El método requiere estandarizar. Si las variables no están todas en la misma escala, arbitrariamente vamos a penalizar los coeficientes de unas más que los de otras. Esto puede funcionar bien pero habitualmente no es lo que queremos.

- **Variables categóricas**: Admite. Hay que crear k-1 variables dummy por cada categórica para evitar la multicolinealidad. Si queremos ponder cada dummy en el término de penalización igual que las demás variables entonces también hay que estandarizarlas.

- **Hiperparámetros**: Alpha (a veces llamado Lambda), que regula cuánto se penaliza el valor de los coeficientes. A mayor alpha mayor sesgo y menor varianza.

K Nearest Neighbors (KNN)

Clasifica cada puntos nuevos buscando los "k" datos del conjunto de entrenamiento más cercanos y promediando la clase de éstos.

- **Estandarización**: El método se basa en una matriz de distancias. Si los features no se encuentran todos en la misma escala una dimensión podría tener arbitrariamente más peso que otras a la hora de determinar las distancias.

- **Variables categóricas**: La medida de distancia por default (euclídea) no tiene sentido matemático con variables categóricas. Se puede usar otras medidas de distancia como base y el algoritmo funciona.

- **Hiperparámetros**:
  - K: La cantidad de vecinos que se usa en la clasificación. A mayor K, mayor sesgo y menor varianza.
  - Weight: Si dentro de los K vecinos más cercanos, se quiere ponderar más a algunos para hacer la clasificación.

Regresión Logística

Calcula la probabilidad P(Y=1|X) aplicando una función "sigmoidea" a una regresión lineal para obtener valores entre 0 y 1.

- **Estandarización**: Igual que en regresión lineal, los parámetros pueden dar cuenta de las distintas unidades. Estandarizar no cambia en nada los resultados de la predicción.

- **Variables categóricas**: Admite. Hay que crear k-1 variables dummy por cada categórica para evitar la multicolinealidad.

- **Hiperparámetros**:
  - No tiene

Regresión Logística Regularizada

Calcula la probabilidad P(Y=1|X) aplicando una función "sigmoidea" a una regresión lineal para obtener valores entre 0 y 1 y aplica un coeficiente de penalización sobre el valor absoluto de los coeficientes.

- **Estandarización**: Hay que estandarizar por la misma razón que en Ridge y Lasso.

- **Variables categóricas**: Admite. Hay que crear k-1 variables dummy por cada categórica para evitar la multicolinealidad.

- **Hiperparámetros**:
  - C. Se comporta al revés que Lambda, un C más alto hace que el modelo elija coeficientes también más altos, es decir, regulariza menos el modelo.
    A mayor C menor sesgo y mayor varianza.

Naive Bayes

Calcula la probabilidad de Y en base a cada uno de los features de forma independiente y se calcula la productoria de todas las dimensiones de X.

- **Estandarización**: No necesita estandarización. En ningún momento se hacen comparaciones entre distintos features, con lo cual todas pueden tener unidades distintas y no es un problema para el modelo.

- **Variables categóricas**: Admite. Hay que crear k-1 variables dummy por cada categórica para evitar la multicolinealidad.

- **Hiperparámetros**:
  - No tiene

Support Vector Machines

Se basa en trazar las mejores líneas discriminantes posibles entre cada par de clases.

- **Estandarización**: El algoritmo se basa en las distancias de los vectores de soporte a la línea discriminante. Si no queremos sobreponderar arbitrariamente una de las dimensiones, todas las variables tienen que estar en la misma escala.

- **Variables categóricas**: Pierden sentido matemático en SVM. Puede soportar algunas variables con pocas categorías pero el modelo funciona mejor para features numéricos.

- **Hiperparámetros**:
  - C: regula el grosor de los márgenes. Es el costo de clasificar mal un punto. A mayor C, menor margen, menor sesgo y mayor varianza.
  - Gamma: define el radio de influencia de los datos de entrenamiento. A valores altos de gamma, el radio de influencia es más pequeño y hay mayor sesgo pero menor varianza.

Hiperparámetro: aquella/s característica/s del modelo que no se “aprenden” de forma directa en los estimadores.

Son valores que tiene que definir quien implementa el modelo.

Algunos modelos de scikit learn transforman los hiperparámetros en parámetros. Por ejemplo el modelo RidgeCV(), elige el mejor valor de alpha, por lo cual en este modelo alpha es un parámetro y no un hiperparámetro.

En el modelo Ridge() de la misma biblioteca, alpha ocupa el lugar de parámetro.

- Dos grandes métodos (no los únicos) de búsqueda de hiperparámetros:

  - **Grid Search:** se busca encontrar el mejor hiperparámetro dentro de una "grilla" (grid) especificada de forma manual. Se realiza una búsqueda exhaustiva para cada valor de la grilla y se elige el parámetro que minimiza una determinada métrica de error (generalmente calculada mediante cross-validation)

  - **Random Search:** dado que grid search es exhaustiva, puede ser computacionalmente intensiva si el espacio de búsqueda es muy grande. Por eso, random search, realiza la búsqueda en un subset (seleccionado aleatoriamente) de parámetros, achicando el espacio de búsqueda.

- GridSearch es la estrategia mediante la cual buscamos los hiperparametros óptimos para un modelo predictivo

- ¿Cómo lo hace? Experimenta con diferentes tipos de combinaciones hiperparametros

- Se trata de buscar las características óptimas, para el problema que queremos abordar, y validando el efecto de ésta selección sobre la performance del modelo con ***cross-validation***

- Se denomina grid, porque la idea es hacer un "retículo" con todos los hiperparametros ensayados y sus resultados en el modelo

A la hora de implementar en sklearn una búsqueda sobre los hiperparámetros tenemos que tener en cuenta las siguientes cuestiones:

- Elegir un estimador, es decir, un modelo sobre el cual queremos trabajar
- Elegir un espacio de parámetros donde vamos a hacer la búsqueda.
- Elegir un método de búsqueda sobre los modelos candidatos (Random Search o Gridsearch)
- Un esquema de validación cruzada, donde se deben elegir la cantidad de particiones.
- La métrica de evaluación para elegir el mejor modelo

Práctica Guiada : usando GridSearch en K- Nearests Neighbours

Práctica Independiente : usando GridSearch en SVM

- Los **hiperparámetros** son características de un modelo que no se “aprenden” directamente de los datos
- Se puede realizar una estimación de la combinación óptima usando como método una búsqueda exhaustiva a lo largo de un “grid” de valores
- Se elige la combinación que minimiza alguna métrica de error o de costo