INTRODUCCIÓN A LA **CIENCIA DE DATOS**

VALIDATION

Preguntas

UNIVERSIDAD DE MONTEVIDEO

Queremos hacer "buenas" predicciones. Ajustar un modelo a nuestros datos de entrenamiento es una cosa, pero:

1. ¿Cómo sabemos que se generaliza bien a los datos no vistos?

2. ¿Cómo sabemos que no se limita a memorizar los datos que le proporcionamos y no hace buenas predicciones sobre muestras futuras, muestras que no ha visto antes?

3. ¿Y cómo seleccionamos un buen modelo en primer lugar?

4. ¿Quizás un algoritmo de aprendizaje diferente podría ser más adecuado para el problema en cuestión?

Objetivos

UNIVERSIDAD DE MONTEVIDEO

Resumamos los puntos principales por los que evaluamos el rendimiento predictivo de un modelo:

1. Queremos **estimar el rendimiento de generalización**, el rendimiento predictivo de nuestro modelo sobre datos futuros (no vistos).

2. Queremos **aumentar el rendimiento predictivo** ajustando el algoritmo de aprendizaje y seleccionando el modelo de mejor rendimiento de un espacio de hipótesis dado.

3. Queremos **identificar el algoritmo** de aprendizaje automático que mejor se adapte al problema en cuestión; por lo tanto, queremos comparar diferentes algoritmos, seleccionando el de mejor desempeño así como el modelo de mejor desempeño del espacio de hipótesis del algoritmo.

Conceptos previos

I.I.D: Se asume que las instancias de entrenamiento son **independientes e identicamente distribuidas,** esto quiere decir que todas las instancias fueron extraídas de la misma distribución de probabilidad y son estadísticamente independientes unas de otras. Un escenario donde las instancias no son independientes es el caso de series de tiempo.

Precisión predictiva: El número de todas las predicciones correctas dividido por el número de ejemplos en el conjunto de datos.

$$\mathrm{ACC} = 1 - \mathrm{ERR},$$

$$\mathrm{ERR}_S = \frac{1}{n} \sum_{i=1}^{n} L(\hat{y}_i, y_i).$$

Función de pérdida 0-1:

$$L(\hat{y}_i, y_i) = \begin{cases} 0 & \text{if } \hat{y}_i = y_i \\ \\ 1 & \text{if } \hat{y}_i \neq y_i, \end{cases}$$

Conceptos previos

Sesgo: el sesgo de un estimador $\hat{\beta}$ es la diferencia entre su valor esperado E[$\hat{\beta}$] y el valor real de un parámetro β que se estima:

$$\text{Bias} = E[\hat{\beta}] - \beta.$$

Varianza: La varianza es simplemente la varianza estadística del estimador $\hat{\beta}$ y su valor esperado E [$\hat{\beta}$], por ejemplo, la diferencia al cuadrado de:

$$\text{Variance} = E\left[\left(\hat{\beta} - E[\hat{\beta}]\right)^2\right].$$

La varianza es una medida de la variabilidad de las predicciones de un modelo si repetimos el proceso de aprendizaje varias veces con pequeñas fluctuaciones en el conjunto de entrenamiento. Cuanto más sensible sea el proceso de construcción de modelos a estas fluctuaciones, mayor será la varianza

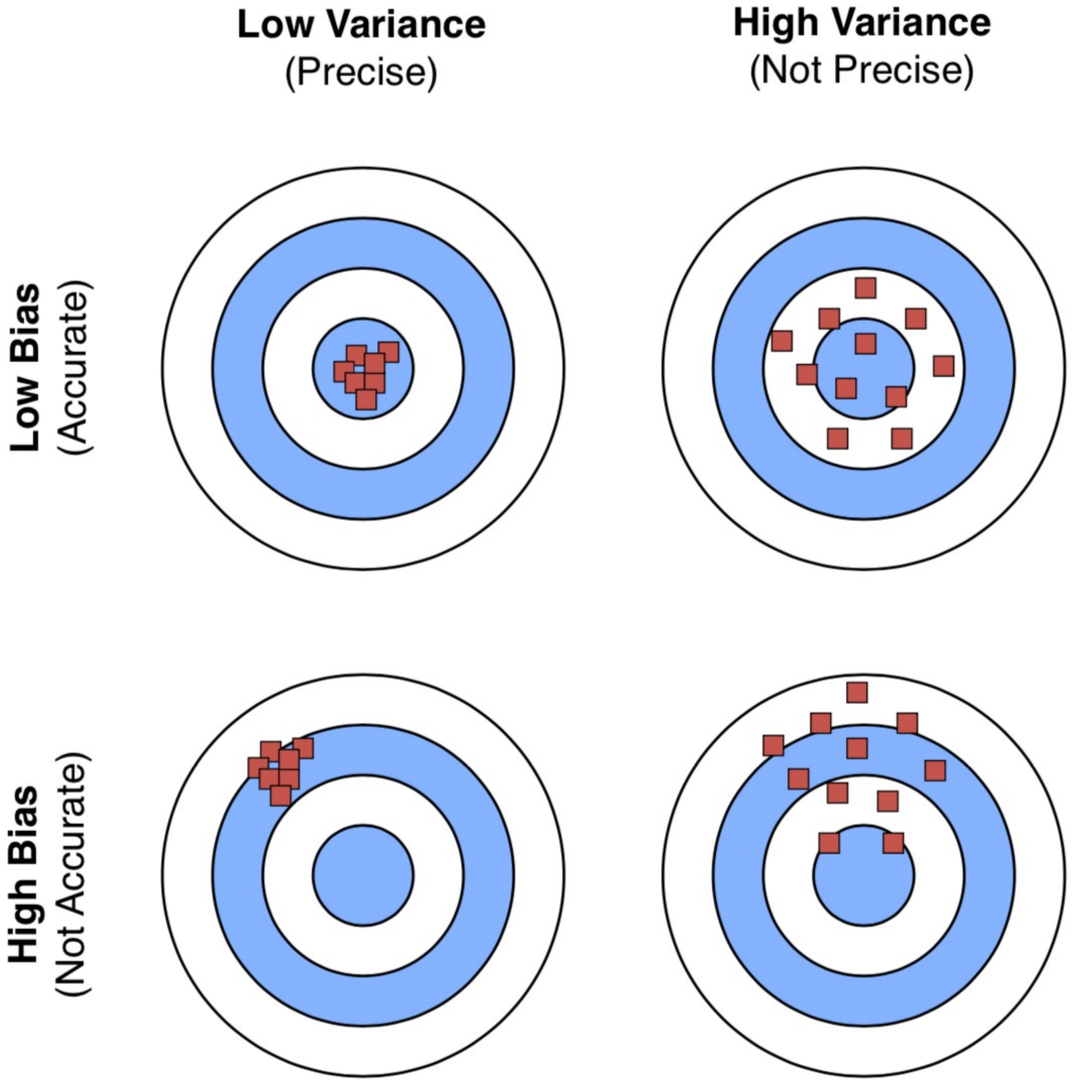
Conceptos previos

En español usamos la misma palabra (precisión) para **accuracy** y para **precision,** por lo que usaremos las palabras en inglés.

En un conjunto de mediciones, **accuracy** es la cercanía de las mediciones a un valor específico, mientras que **precision** es la cercanía de las mediciones entre sí.

Accuracy, es una descripción de errores sistemáticos, una medida de sesgo estadístico; la baja precisión provoca una diferencia entre un resultado y un valor "verdadero".

Precision es una descripción de errores aleatorios, una medida de variabilidad estadística.



Conceptos previos

UNIVERSIDAD DE MONTEVIDEO

Función objetivo: En el modelado predictivo, normalmente estamos interesados en modelar un proceso en particular; queremos aprender o aproximarnos a una función desconocida específica. La función objetivo $f(x) = y$ es la verdadera función f (·) que queremos modelar.

Hipótesis: Una hipótesis es una determinada función que creemos (o esperamos) que sea similar a la función verdadera, la función objetivo f (·) que queremos modelar.

Modelo: En el campo del aprendizaje automático, los términos hipótesis y modelo a menudo se usan indistintamente. En otras ciencias, estos términos pueden tener diferentes significados: una hipótesis podría ser la "suposición fundamentada" del científico, y el modelo sería la manifestación de esta suposición para probar esta hipótesis.

Conceptos previos

UNIVERSIDAD DE MONTEVIDEO

Algoritmo de aprendizaje: Nuevamente, nuestro objetivo es encontrar o aproximar la función objetivo, y el algoritmo de aprendizaje es un conjunto de instrucciones que intentaron modelar la función objetivo utilizando un conjunto de datos de entrenamiento. Un algoritmo de aprendizaje viene con un espacio de hipótesis, el conjunto de posibles hipótesis que puede explorar para modelar la función objetivo desconocida mediante la formulación de la hipótesis final.

Hiperparámetros: Los hiperparámetros son los parámetros de ajuste de un algoritmo de aprendizaje automático; por ejemplo, la profundidad máxima de un clasificador de árbol de decisión. Por el contrario, los parámetros del modelo son los parámetros que un algoritmo de aprendizaje ajusta a los datos de entrenamiento, los parámetros del modelo en sí. Por ejemplo, los coeficientes de peso (o pendiente) de una línea de regresión lineal y su término de sesgo (aquí: intersección del eje y) son parámetros del modelo.

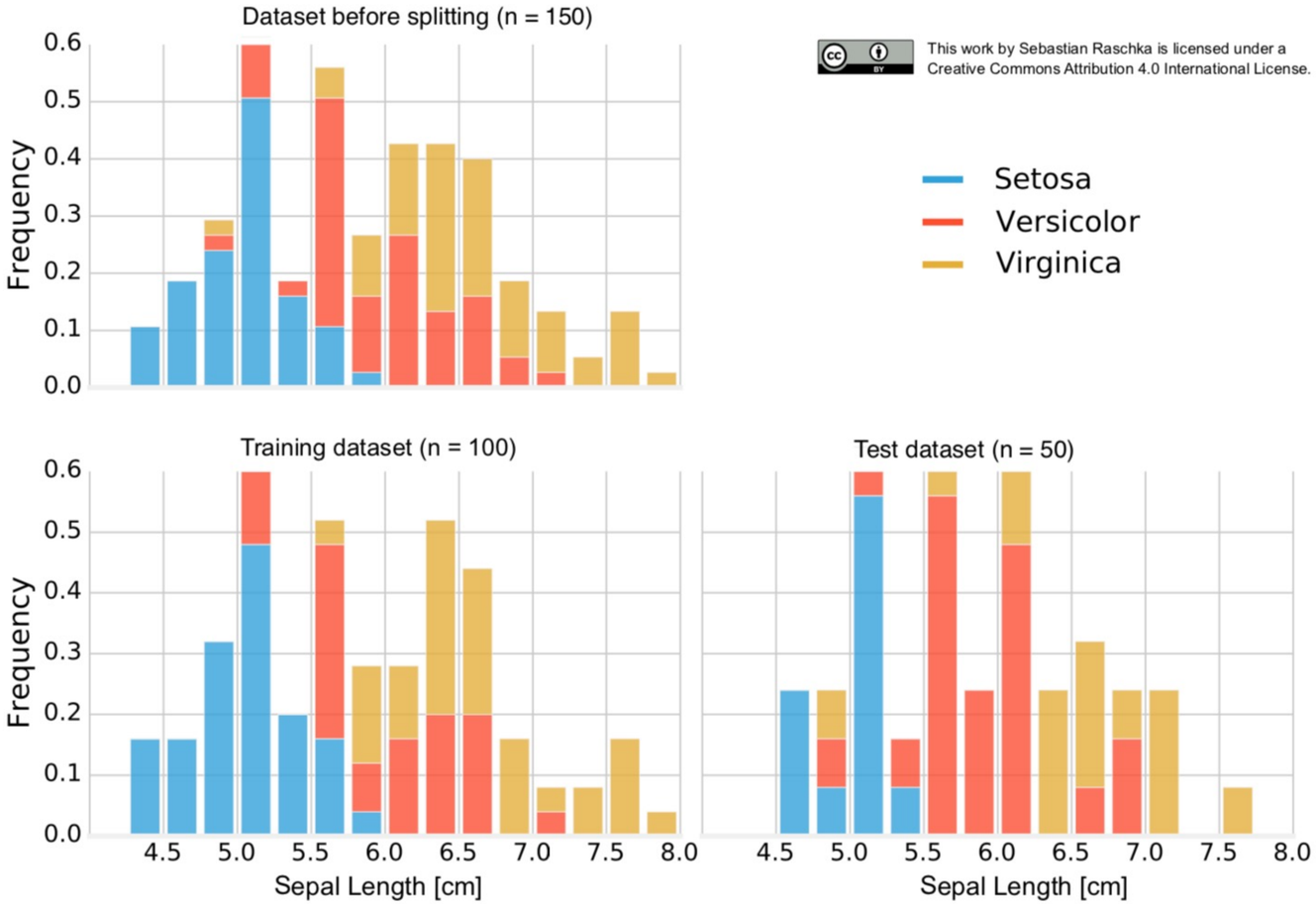
Validación de resustitución y método de exclusión

UNIVERSIDAD DE MONTEVIDEO

Validación de resustitución (resubstitution validation): se entrena y evalúa un modelo en el mismo conjunto de datos de entrenamiento (esto se llama validación de resustitución o evaluación de resustitución), ya que normalmente introduciría un sesgo muy optimista debido al sobreajuste (overfitting). En otras palabras, no podemos decir si el modelo simplemente memorizó los datos de entrenamiento o si se generaliza bien a datos nuevos que no se ven.

Método de exclusión (Holdout method): Primero, tomamos un conjunto de datos etiquetados y lo dividimos en dos partes: un conjunto de training y un conjunto de testing. Luego, ajustamos un modelo a los datos de entrenamiento y predecimos las etiquetas del conjunto de test. La fracción de predicciones correctas, que se puede calcular comparando las etiquetas predichas con las etiquetas verdaderas del conjunto de prueba, constituye nuestra estimación de la precisión de predicción del modelo.

Stratification and class imbalance



If a random function assigns 2/3 of the flowers (100) to the training set and 1/3 of the flowers (50) to the test set, it may yield the following (also shown in Figure 1):

- training set $\rightarrow$ 38 $\times$ Setosa, 28 $\times$ Versicolor, 34 $\times$ Virginica
- test set $\rightarrow$ 12 $\times$ Setosa, 22 $\times$ Versicolor, 16 $\times$ Virginica

Stratification and class imbalance

UNIVERSIDAD DE MONTEVIDEO

Estratificación: La estratificación simplemente significa que dividimos aleatoriamente un conjunto de datos de manera que cada clase esté correctamente representada en los subconjuntos resultantes (el entrenamiento y el conjunto de prueba); en otras palabras, la estratificación es un enfoque para mantener la proporción de clase original en los subconjuntos resultantes.

Desequilibrio de clases: Un problema de modelado predictivo de clasificación donde la distribución de ejemplos entre las clases no es igual. Por ejemplo, podemos recopilar medidas de flores y tener 80 ejemplos de una especie de flor y 20 ejemplos de una segunda especie de flor, y solo estos ejemplos comprenden nuestro conjunto de datos de entrenamiento. Esto representa un ejemplo de un problema de clasificación desequilibrado.

Holdout Validation

UNIVERSIDAD DE MONTEVIDEO

Paso 1: Dividimos aleatoriamente nuestros datos disponibles en dos subconjuntos: un entrenamiento y un conjunto de prueba. Dejar de lado los datos de prueba es una solución alternativa para lidiar con las imperfecciones de un mundo no ideal, como los datos y recursos limitados, y la incapacidad de recopilar más datos de la distribución. Aquí, el conjunto de prueba representará datos nuevos no vistos para el modelo. Es importante que el conjunto de prueba se use solo una vez para evitar introducir sesgos cuando estimamos el desempeño de la generalización. Normalmente, asignamos 2/3 al conjunto de entrenamiento y 1/3 de los datos al conjunto de prueba. Otras divisiones comunes de entrenamiento/prueba son 60/40, 70/30 u **80/20**, o incluso 90/10 si el conjunto de datos es relativamente grande.

Paso 2: Después de dejar de lado los ejemplos de prueba, elegimos un algoritmo de aprendizaje que creemos que podría ser apropiado para el problema dado. Tenemos que especificar los valores de los hiperparámetros manualmente; el algoritmo de aprendizaje no los aprende de los datos de entrenamiento en contraste con los parámetros reales del modelo. Dado que los hiperparámetros no son aprendidos durante el ajuste del modelo, necesitamos algún tipo de "procedimiento adicional" o ”loop externo" para optimizarlos por separado. Entonces, por ahora, tenemos que ir con algunos valores de hiperparámetros fijos; podríamos usar nuestra intuición o los parámetros predeterminados de un algoritmo estándar.

Holdout Validation

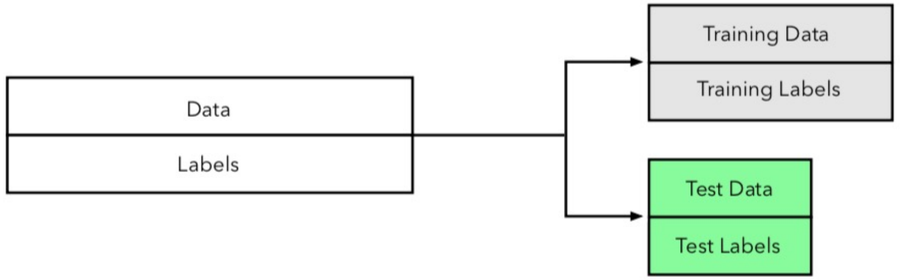
UNIVERSIDAD DE MONTEVIDEO

Paso 3: Después de que el algoritmo de aprendizaje se ajusta a un modelo en el paso anterior, la siguiente pregunta es: ¿Qué tan "bueno" es el rendimiento del modelo resultante? Aquí es donde entra en juego el conjunto de prueba independiente. Dado que el algoritmo de aprendizaje no ha "visto" este conjunto de pruebas antes, debería proporcionar una estimación relativamente imparcial de su rendimiento en datos nuevos no vistos. Ahora, tomamos este conjunto de prueba y usamos el modelo para predecir las etiquetas de clase. Luego, tomamos las etiquetas de clase predichas y las comparamos con la "verdad fundamental", las etiquetas de clase correctas, para estimar la precisión o el error de generalización de los modelos.

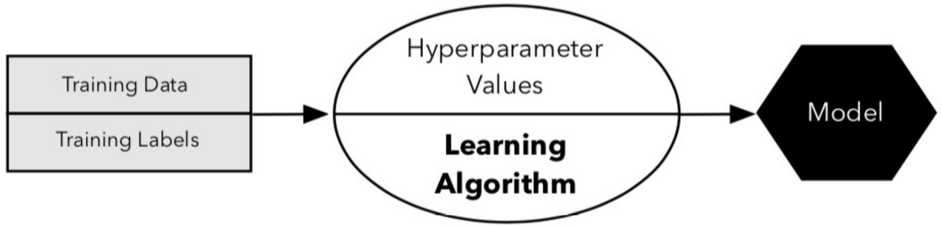
Paso 4: Finalmente, obtuvimos una estimación de qué tan bien funciona nuestro modelo con datos no vistos. Por lo tanto, ya no hay razón para retener el conjunto de prueba del algoritmo. Dado que asumimos que nuestras muestras son i.i.d., no hay razón para suponer que el modelo funcionaría peor después de alimentarlo con todos los datos disponibles. Como regla general, el modelo tendrá un mejor rendimiento de generalización si los algoritmos utilizan datos más informativos, asumiendo que aún no ha alcanzado su capacidad.

UNIVERSIDAD DE MONTEVIDEO

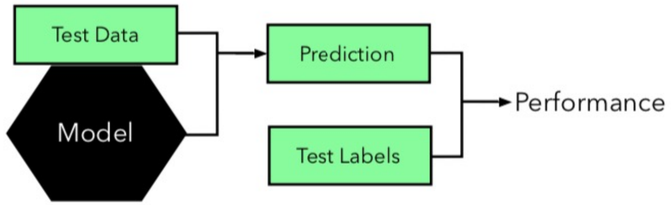
1



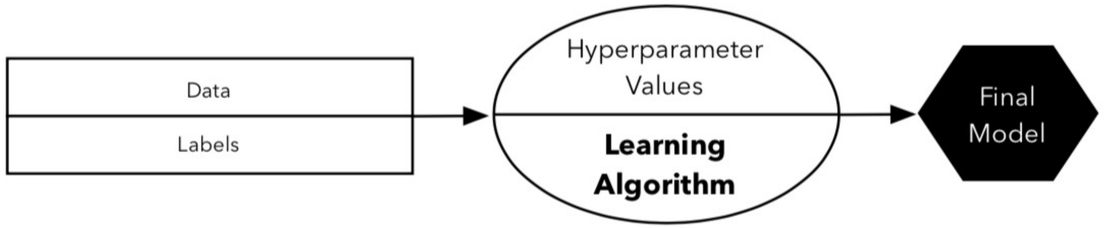
2



3



4



Resampling

UNIVERSIDAD DE MONTEVIDEO

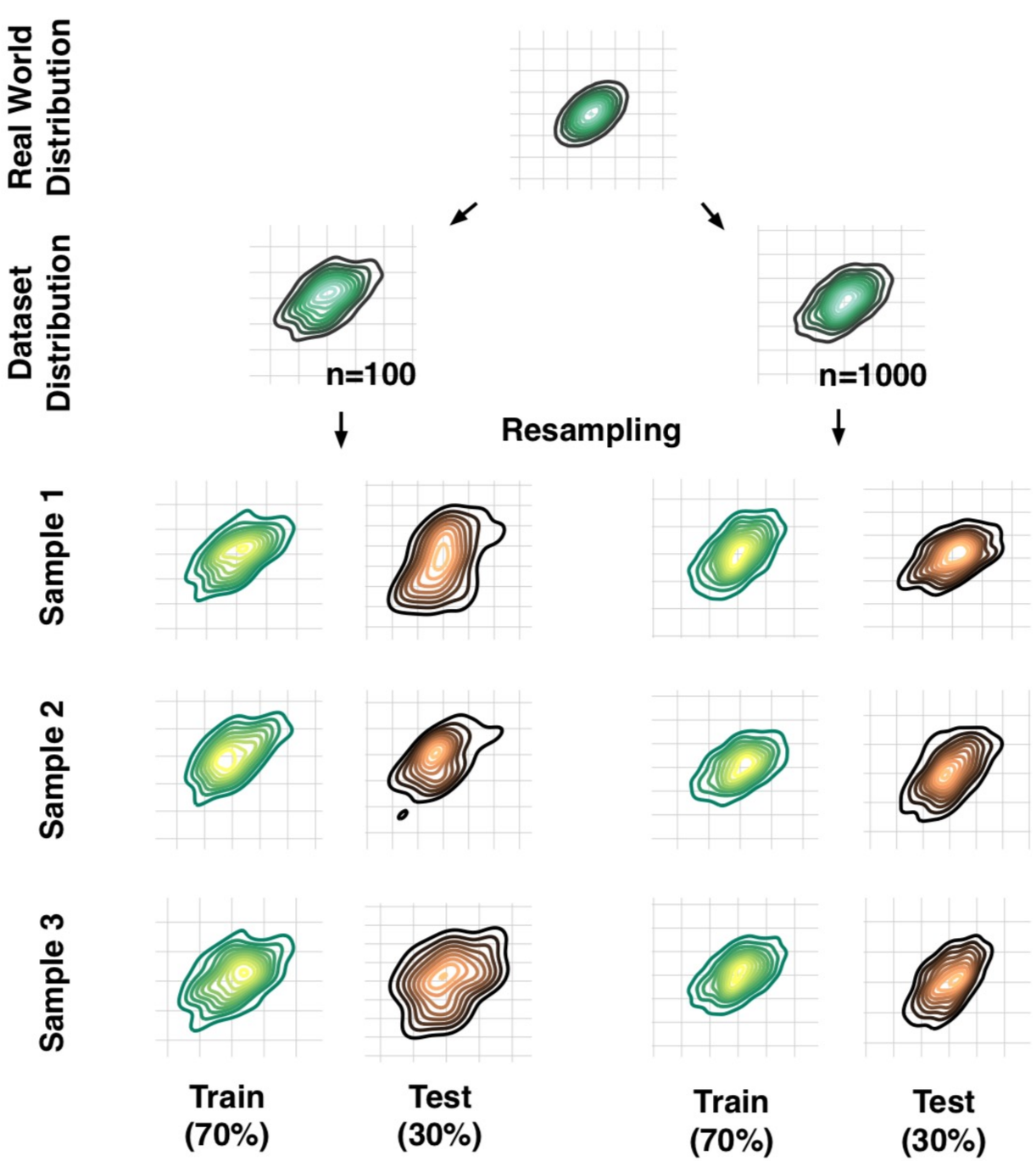
La mayoría de los algoritmos de aprendizaje supervisado para clasificación y regresión, así como las estimaciones de rendimiento, operan bajo el supuesto de que un **conjunto de datos es representativo de la población de la que se extrajo**.

Por ejemplo, la evaluación por resustitución (ajustar un modelo a un conjunto de entrenamiento y usar el mismo conjunto de entrenamiento para la evaluación del modelo) tiene un sesgo muy optimista (no hay instancias de testing, solo training).

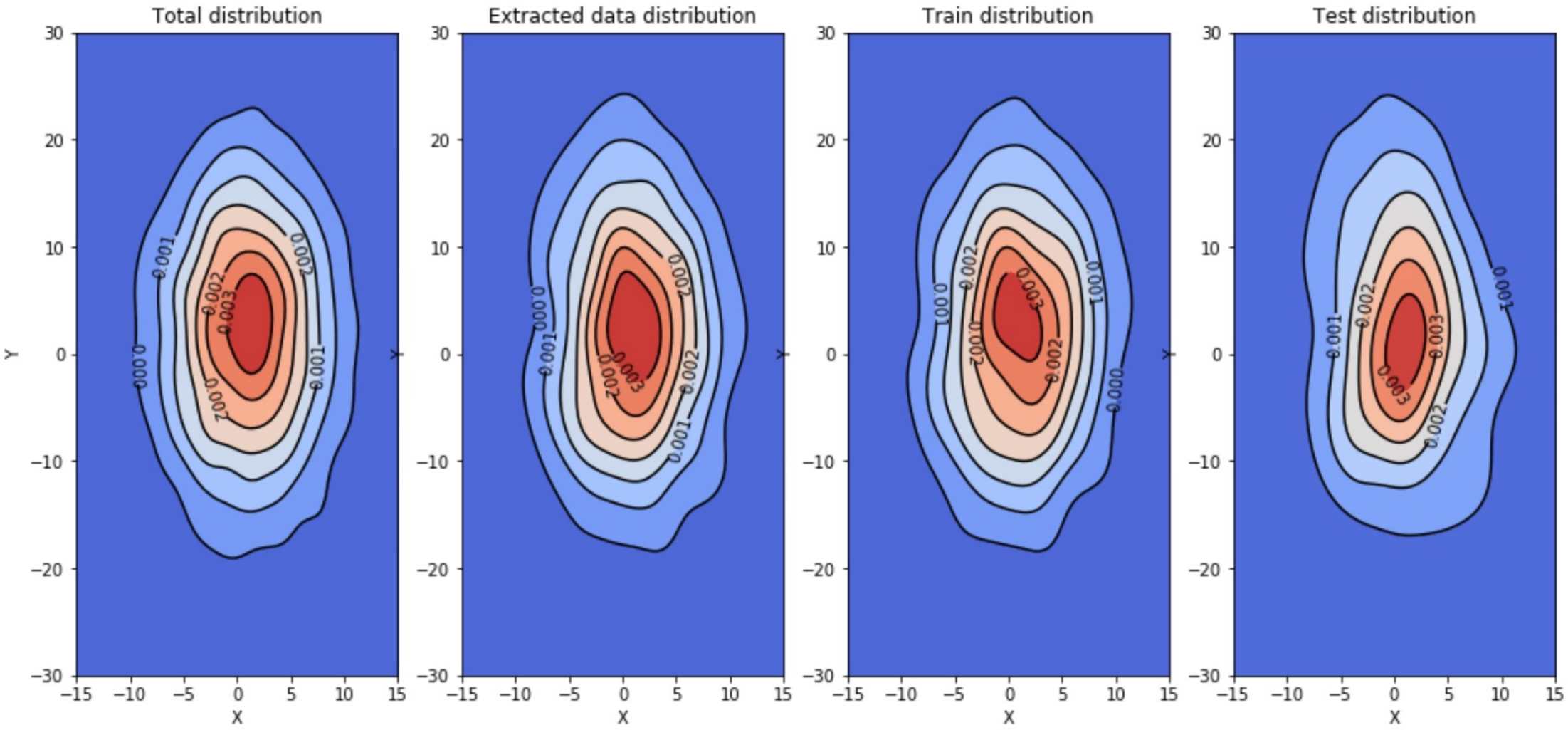
En definitiva, el algoritmo tendrá una performance sobre-optimista porque todas las instancias donde será evaluado son instancias ya vistas (training = testing).

Viceversa, retener una gran parte del conjunto de datos como conjunto de testing puede llevar a estimaciones con sesgo pesimista (muy poco training, mucho testing). Si bien la reducción del tamaño del conjunto de pruebas puede disminuir este sesgo pesimista, lo más probable es que aumente la varianza de las estimaciones de rendimiento.

Es decir, si el conjunto de datos de testing es desproporcionalmente más largo al conjunto de training, es más probable que existan en testing casos no antes vistos en training, y por tanto, el algoritmo fallará a la hora de generalizar a estos casos no vistos.



UNIVERSIDAD DE MONTEVIDEO


Total distribution
Extracted data distribution
Train distribution
Test distribution
X
Y
30
20
10
0
−10
−20
−30
−15
−10
−5
0
5
10
15
0.000
0.001
0.002
0.003

Resampling

La razón por la que un conjunto de testing proporcionalmente grande aumenta el sesgo pesimista es que es posible que el modelo aún no haya alcanzado su capacidad total. En otras palabras, el algoritmo de aprendizaje podría haber formulado una hipótesis de clasificación más poderosa y generalizable si hubiera visto más datos.

Un algoritmo de entrenamiento puede beneficiarse de más datos de entrenamiento, datos que se retuvieron para realizar testing. Por lo tanto, después de evaluar un modelo, es posible que deseemos ejecutar el algoritmo de aprendizaje una vez más en el conjunto de datos completo antes de usarlo en una aplicación del mundo real.

Disminuir el tamaño del conjunto de prueba plantea otro problema: puede resultar en una variación sustancial de la estimación de rendimiento de un modelo.

Resampling

Training chico y testing grande:

- Es más probable que training no sea representativo de la realidad que quiero modelar.
- El modelo aprenderá con menos casos teniendo un sesgo pesimista.
- Habrá más varianza en testing.
- Underfitting.
- El modelo no alcanzará su máxima capacidad.
- Puede aprender ruido.

Training grande y testing chico:

- Incrementa la variación de nuestra estimación de la performance futura del modelo.
- El modelo se evaluará sobre casos ya vistos por lo que tendrá un sesgo optimista.
- Overfitting.

Repeated Holdout Validation (Monte Carlo)

UNIVERSIDAD DE MONTEVIDEO

Una forma de obtener una estimación de rendimiento más robusta que sea menos variante de cómo dividimos los datos en conjuntos de training y testing es repetir holdout validation k veces con diferentes semillas aleatorias (random seeds) y calcular el rendimiento promedio sobre estas k repeticiones.

Este procedimiento de holdout validation, a veces también llamado **Validación cruzada de Monte Carlo o Repeated Holdout Validation**, proporciona una mejor estimación de qué tan bien puede funcionar nuestro modelo en un conjunto de pruebas aleatorio, en comparación con el método de holdout validation estándar.

Además, proporciona información sobre la estabilidad del modelo: cómo el modelo, producido por un algoritmo de aprendizaje, cambia con las diferentes divisiones del conjunto de entrenamiento.

Repeated Holdout Validation (Monte Carlo)

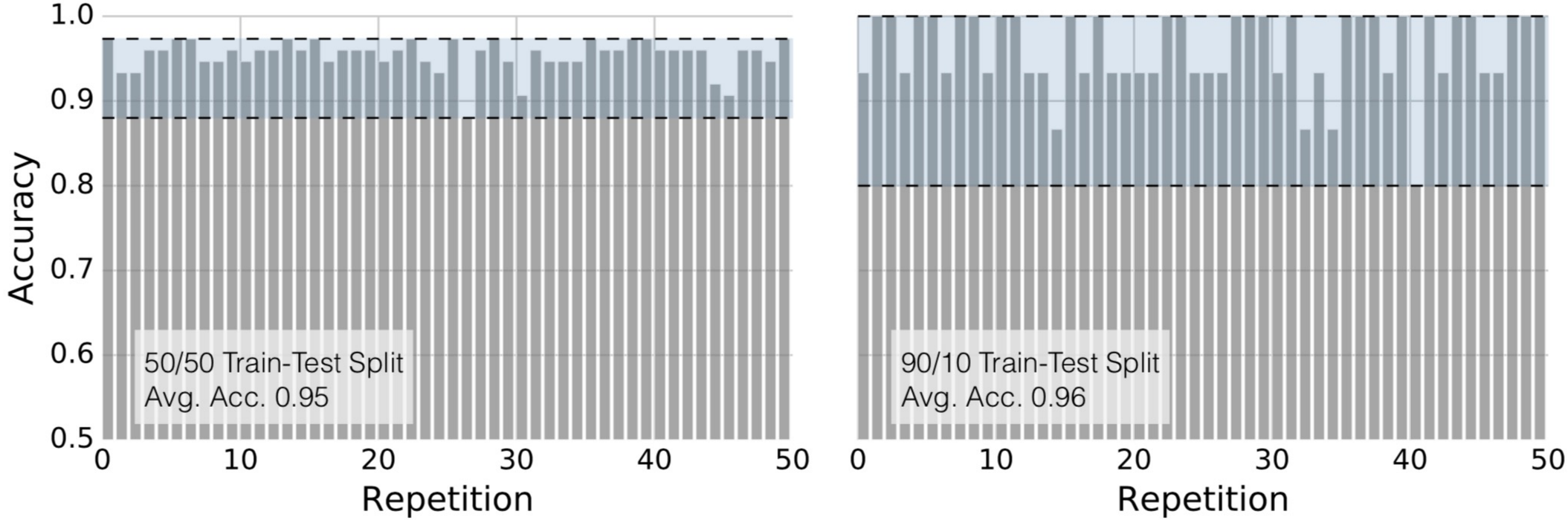


Imagen: 50 repeticiones del mismo experimento pero con splits de distintas proporciones (50/50 y 90/10)

Primero, vemos que la varianza de nuestra estimación aumenta a medida que disminuye el tamaño del conjunto de prueba. En segundo lugar, vemos un pequeño aumento en el sesgo pesimista cuando disminuimos el tamaño del conjunto de entrenamiento; retenemos más datos de entrenamiento en la división 50/50, lo que puede ser la razón por la cual el rendimiento promedio sobre las 50 divisiones es ligeramente menor en comparación a las divisiones 90/10.

Validation and Hyperparameter Optimization

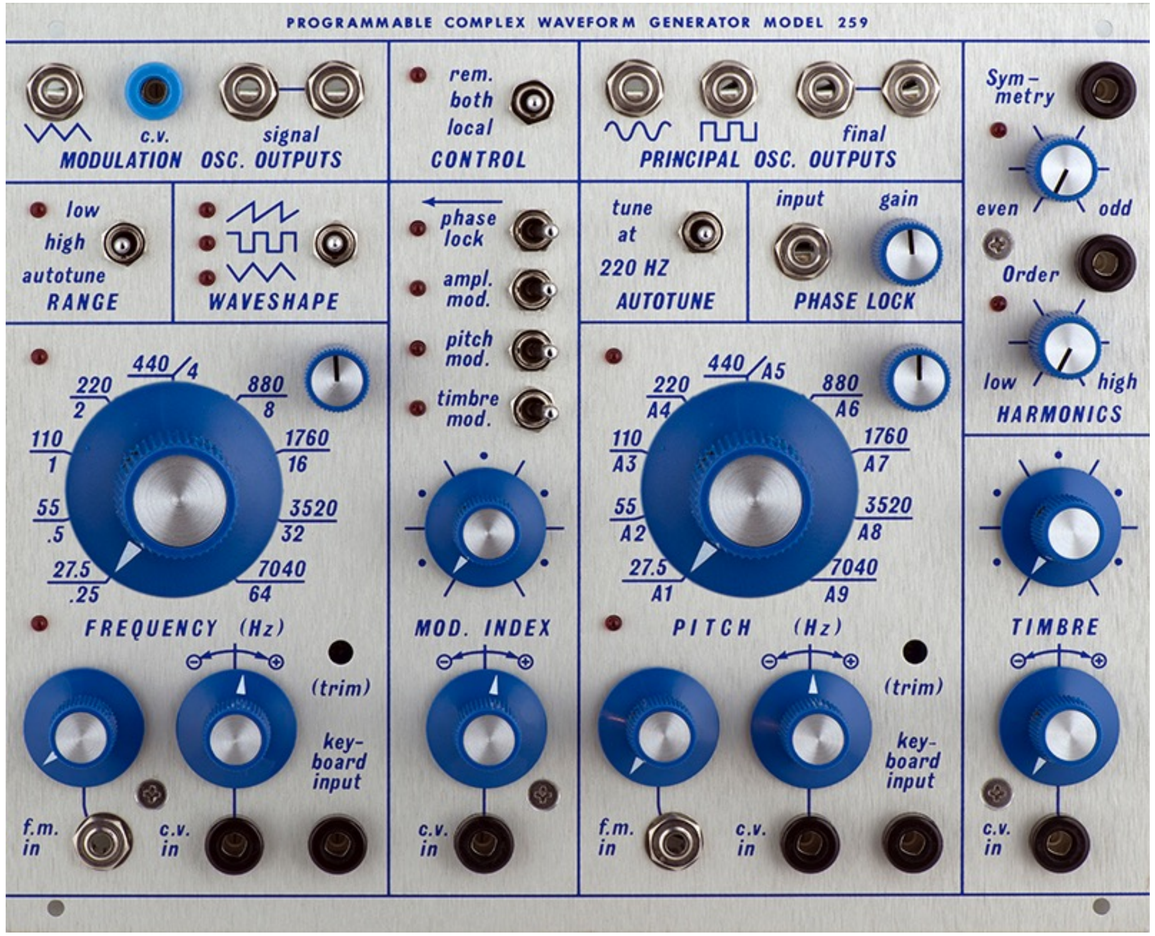
UNIVERSIDAD DE MONTEVIDEO

Casi todos los algoritmos de aprendizaje automático vienen con una gran cantidad de hiperparámetros que nosotros, científicos de datos, debemos especificar. Estos botones de ajuste, los llamados **hiperparámetros**, nos ayudan a **controlar el comportamiento** de los algoritmos de aprendizaje automático cuando **optimizamos el rendimiento**, encontrando el equilibrio adecuado entre sesgo y varianza.

El ajuste de hiperparámetros para la optimización del rendimiento es un arte en sí mismo y no existen reglas estrictas que garanticen el mejor rendimiento en un conjunto de datos determinado.

Cambiar los valores de los hiperparámetros cuando se ejecuta un algoritmo de aprendizaje sobre un conjunto de entrenamiento puede dar como resultado modelos diferentes.

El proceso de encontrar el modelo de mejor rendimiento a partir de un conjunto de modelos que fueron producidos por diferentes configuraciones de hiperparámetros se llama **selección de modelos**.



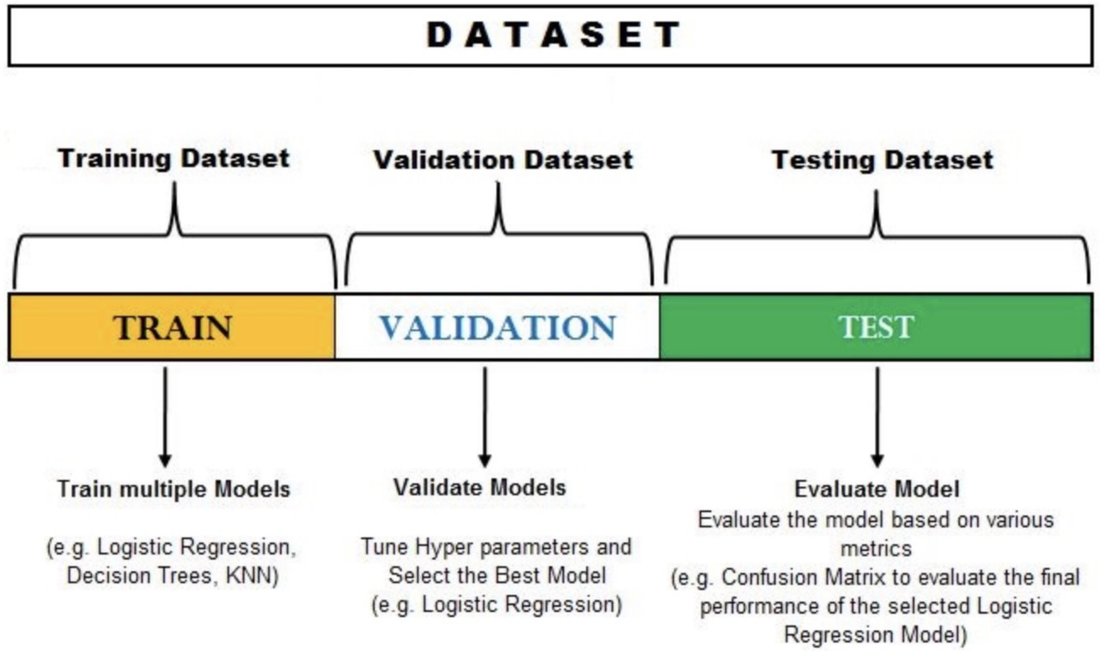
Validation and Hyperparameter Optimization

Como queremos saber qué tan bien se generaliza nuestro modelo a nuevos datos, usamos holdout validation para dividir el conjunto de datos en dos partes, un conjunto de entrenamiento y un conjunto de prueba independiente.

¿Podemos usar holdout validation para el ajuste de hiperparámetros?

La respuesta es sí. Sin embargo, tenemos que hacer una pequeña modificación a nuestro enfoque inicial, y dividir el conjunto de datos en tres partes:

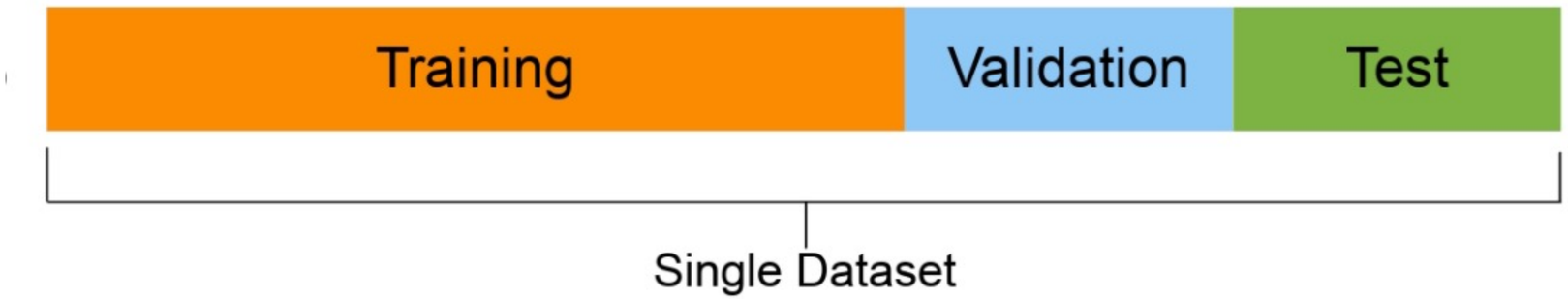
1. Un conjunto de **entrenamiento** (train).
2. Un conjunto de **validación** (validation).
3. Un conjunto de **prueba** (test).



Validation and Hyperparameter Optimization

Reutilizar el conjunto de prueba varias veces introduciría un sesgo, la estimación de rendimiento final probablemente resultaría en estimaciones demasiado optimistas del rendimiento de generalización; se podría decir que "el conjunto de prueba filtra información".

Tener un par de entrenamiento-validación para el ajuste de hiperparámetros y las selecciones de modelos nos permite mantener el conjunto de prueba "independiente" para la evaluación del modelo.



Three-way Holdout Method

UNIVERSIDAD DE MONTEVIDEO

Paso 1. Comenzamos dividiendo nuestro conjunto de datos en tres partes, un conjunto de entrenamiento para el ajuste del modelo, un conjunto de validación para la selección del modelo y un conjunto de prueba para la evaluación final del modelo seleccionado.

Paso 2. Este paso ilustra la etapa de ajuste de hiperparámetros. Usamos el algoritmo de aprendizaje con diferentes configuraciones de hiperparámetros (aquí: tres) para ajustar los modelos a los datos de entrenamiento.

Paso 3. A continuación, evaluamos el rendimiento de nuestros modelos en el conjunto de validación. Este paso ilustra la etapa de selección del modelo; después de comparar las estimaciones de rendimiento, elegimos la configuración de hiperparámetros asociados con el mejor rendimiento. Tenga en cuenta que a menudo fusionamos los pasos dos y tres en la práctica: ajustamos un modelo y calculamos su rendimiento antes de pasar al siguiente modelo para evitar mantener todos los modelos ajustados en la memoria.

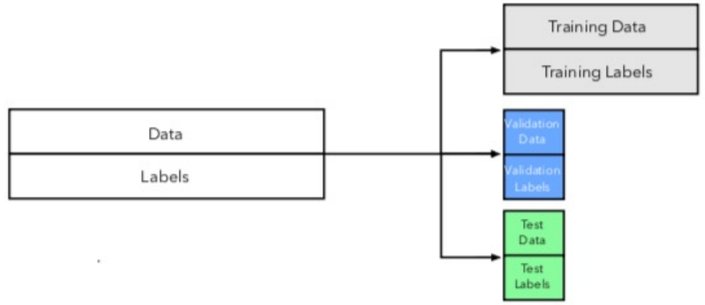
Paso 4. Como se discutió anteriormente, las estimaciones de rendimiento pueden sufrir un sesgo pesimista si el conjunto de entrenamiento es demasiado pequeño. Por lo tanto, podemos fusionar el conjunto de entrenamiento y validación después de la selección del modelo y usar la mejor configuración de hiperparámetros del paso anterior para ajustar un modelo a este conjunto de datos más grande.

Paso 5. Ahora, podemos usar el conjunto de pruebas independientes para estimar el desempeño de generalización de nuestro modelo. Recuerde que el propósito del conjunto de prueba es simular nuevos datos que el modelo no ha visto antes. La reutilización de este conjunto de pruebas puede resultar en un sesgo demasiado optimista en nuestra estimación del rendimiento de generalización del modelo.

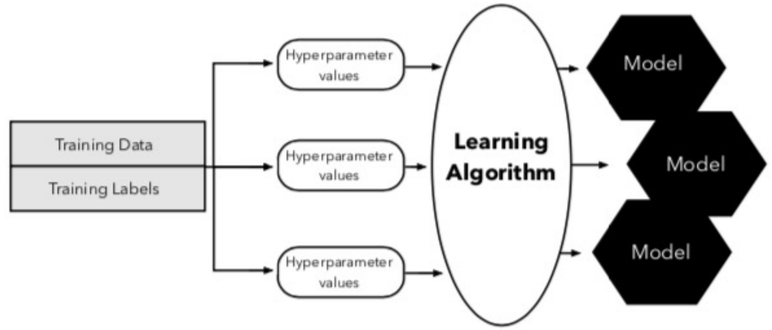
Paso 6. Finalmente, podemos hacer uso de todos nuestros datos, fusionando el entrenamiento y el conjunto de pruebas, y ajustar un modelo a todos los puntos de datos para uso en el mundo real.

UNIVERSIDAD DE MONTEVIDEO

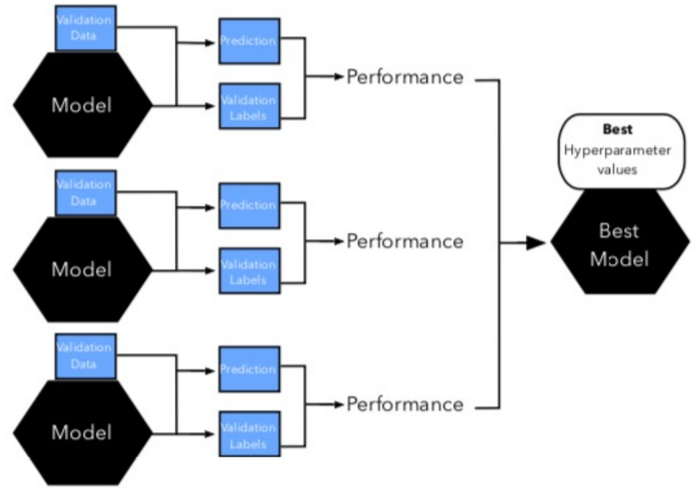
1



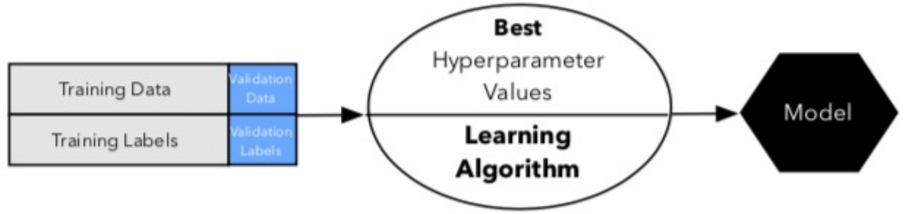
2



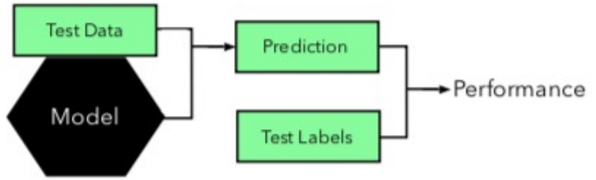
3



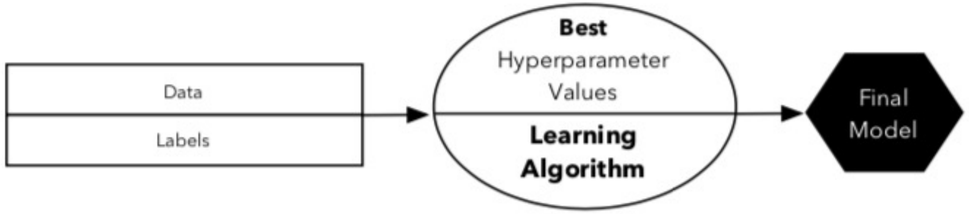
4



5



6



K-fold Cross-validation

UNIVERSIDAD DE MONTEVIDEO

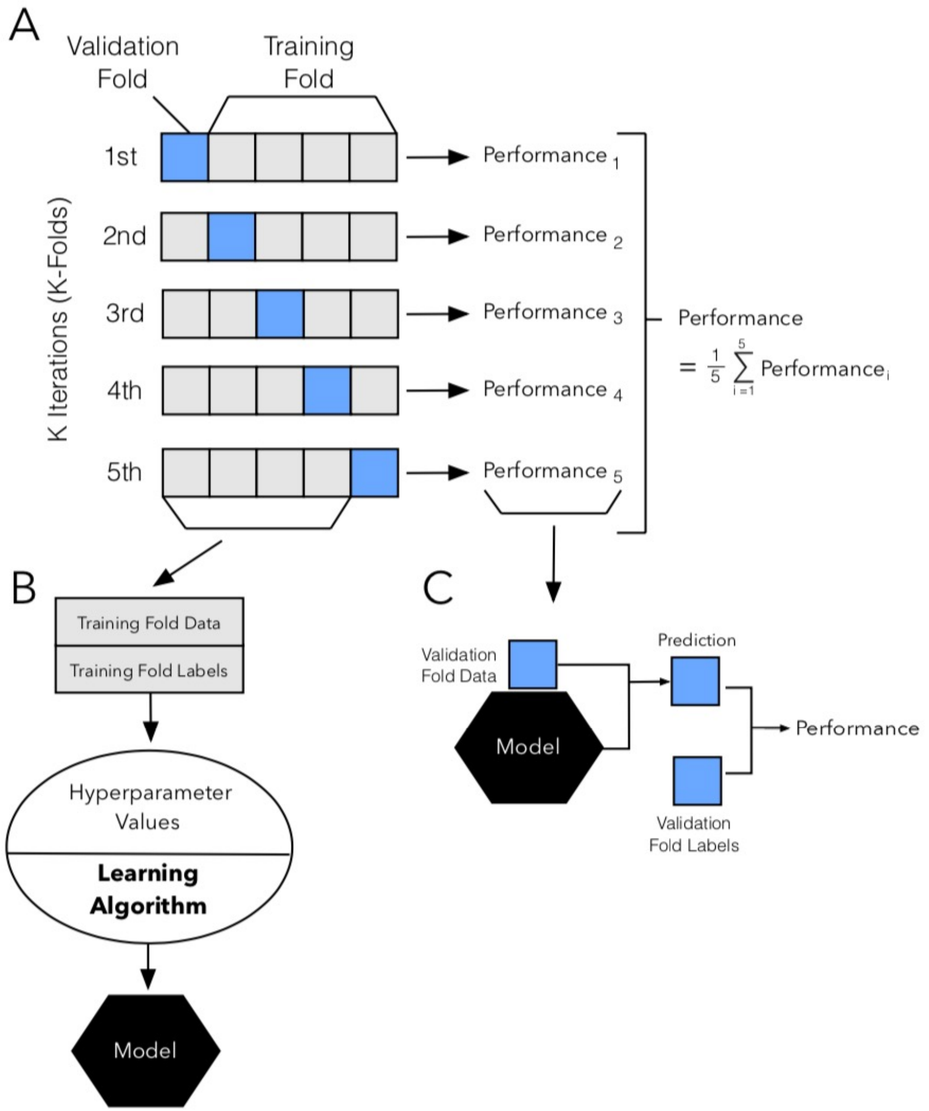
Es hora de introducir la técnica probablemente **más común** para la evaluación y selección de modelos en la práctica del aprendizaje automático: la **validación cruzada de k-particiones.**

El término validación cruzada se usa de manera vaga en la literatura, donde los profesionales e investigadores a veces se refieren al método de retención (holdout method) de entrenamiento/prueba como una técnica de validación cruzada.

Sin embargo, podría tener más sentido pensar en validación como cruce de etapas de formación y validación en rondas sucesivas. Aquí, la idea principal detrás de la validación cruzada es que cada muestra en nuestro conjunto de datos tiene la oportunidad de ser probada.

La validación cruzada de k-particiones es un caso especial de validación cruzada donde **iteramos sobre un conjunto de datos k veces.** En cada ronda, dividimos el conjunto de datos en k partes: una parte se usa para la validación, y las k - 1 partes restantes se fusionan en un subconjunto de entrenamiento para la evaluación del modelo, como se muestra en la figura, que ilustra el proceso de cruce de 5 partes -validación.

K-fold Cross-validation

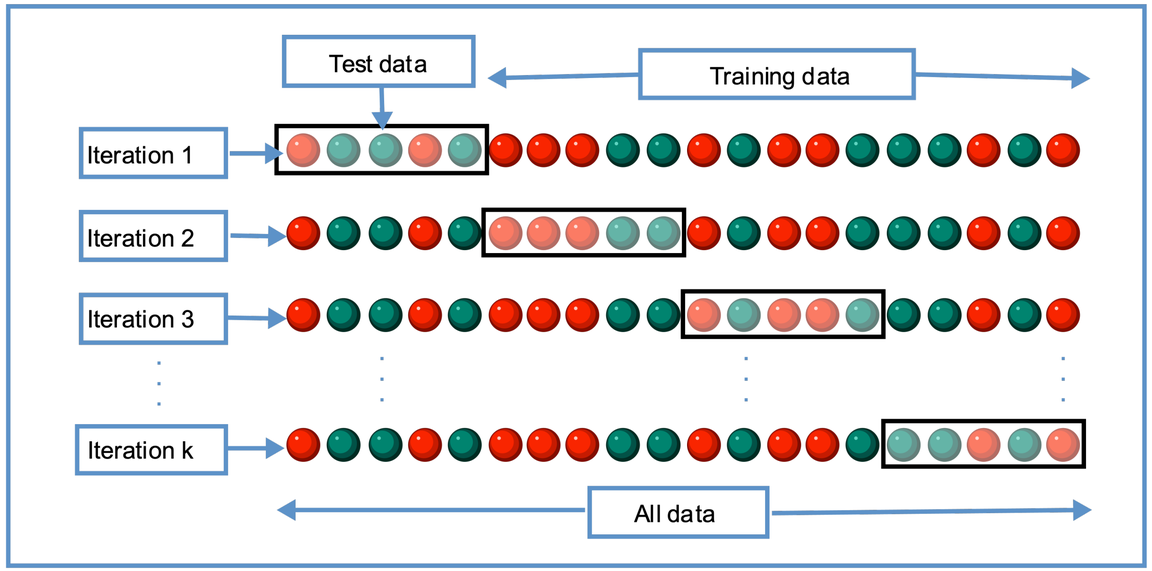


K-fold Cross-validation

Usamos un algoritmo de aprendizaje con **configuraciones de hiperparámetros fijos** para ajustar los modelos a las particiones de entrenamiento en cada iteración cuando usamos el método de validación cruzada de k-particiones para la evaluación del modelo.

En la validación cruzada de 5 particiones (5-fold), este procedimiento dará como resultado cinco modelos diferentes ajustados/entrenados. Estos modelos se ajustaron a conjuntos de entrenamiento distintos pero parcialmente superpuestos y se evaluaron en conjuntos de validación no superpuestos.

Finalmente, calculamos el rendimiento de la validación cruzada como la media aritmética sobre las k estimaciones de rendimiento de los conjuntos de validación.



K-fold Cross-validation

UNIVERSIDAD DE MONTEVIDEO

Paso 1. De manera similar al holdout method, dividimos el conjunto de datos en dos partes, un conjunto de entrenamiento y un conjunto de prueba independiente; guardamos el conjunto de prueba para el paso de evaluación del modelo final al final (Paso 4).

Paso 2. En este segundo paso, ahora podemos experimentar con varios ajustes de hiperparámetros; podríamos utilizar la optimización bayesiana, la búsqueda aleatoria o la búsqueda en cuadrícula, por ejemplo. Para cada configuración de hiperparámetro, aplicamos el método de validación cruzada de k veces en el conjunto de entrenamiento, lo que da como resultado múltiples modelos y estimaciones de rendimiento.

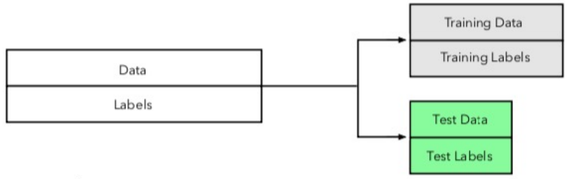
Paso 3. Tomando los ajustes de hiperparámetros que produjeron los mejores resultados en el procedimiento de validación cruzada de k veces. Podemos usar el conjunto de entrenamiento completo para el ajuste del modelo con estos ajustes.

Paso 4. Ahora, usamos el conjunto de prueba independiente que retuvimos anteriormente (Paso 1); utilizamos este conjunto de pruebas para evaluar el modelo que obtuvimos del Paso 3.

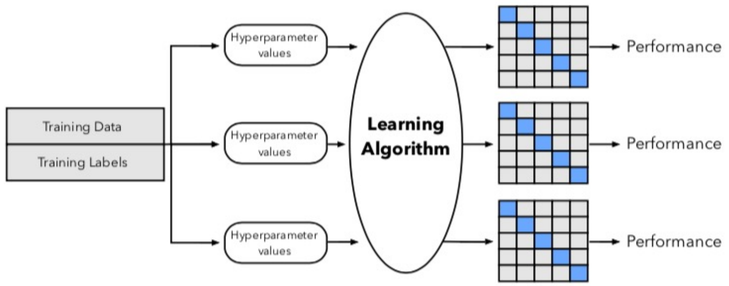
Paso 5. Finalmente, después de que completamos la etapa de evaluación, opcionalmente podemos ajustar un modelo a todos los datos (conjuntos de datos de entrenamiento y prueba combinados), que podría ser el modelo para producción (deployment).

UNIVERSIDAD DE MONTEVIDEO

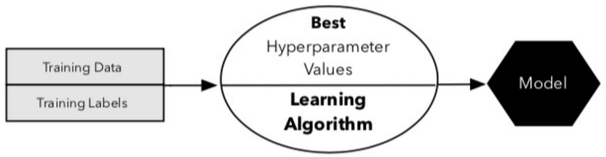
1



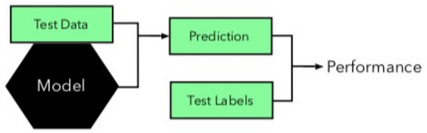
2



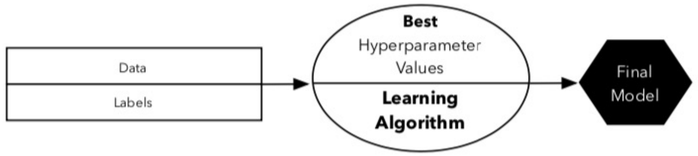
3



4



5



Comentario sobre la selección de modelos y grandes conjuntos de datos

UNIVERSIDAD DE MONTEVIDEO

Cuando examinamos la literatura sobre deep learning, a menudo encontramos que el 3-way holdout method es el método de elección cuando se trata de evaluación de modelos; también es común en la literatura más antigua (que no es de deep learning).

Como se mencionó anteriormente, el 3-way holdout method puede ser preferible a la validación cruzada de k-veces ya que la primera es computacionalmente barata en comparación. Aparte de las preocupaciones de eficiencia computacional, solo usamos algoritmos de aprendizaje profundo cuando tenemos tamaños de muestra relativamente grandes de todos modos, escenarios en los que no tenemos que preocuparnos tanto por la alta varianza (la sensibilidad de nuestras estimaciones hacia cómo dividimos el conjunto de datos para entrenamiento, validación y pruebas).

Por lo tanto, está bien usar el holdout method con una división de entrenamiento, validación y prueba en vez de la validación cruzada de k-veces para la selección del modelo si el conjunto de datos es relativamente grande.

Comentario sobre la selección de features durante la selección del modelo

Tenga en cuenta que si normalizamos los datos o seleccionamos features, normalmente realizamos estas operaciones dentro del ciclo de validación cruzada de k-fold, en contraste con la aplicación de estos pasos a todo el conjunto de datos por adelantado antes de dividir los datos en partes.

La selección de features dentro del ciclo de validación cruzada **reduce el sesgo** a través desobreajuste (overfitting), ya que evita la fuga de información (data leakage) de los datos de prueba durante la etapa de entrenamiento.

Sin embargo, la selección de features dentro del ciclo de validación cruzada **puede llevar a una estimación demasiado pesimista**, ya que hay menos datos disponibles para entrenamiento.