**Conceptos de red neuronal profunda**

Antes de explorar cómo entrenar un modelo de Machine Learning de red neuronal profunda(DNN), consideremos lo que se intenta lograr. El aprendizaje automático está relacionado con la predicción de una *etiqueta* en función de algunas *características* de una observación determinada. En términos simples, un modelo de Machine Learning es una función que calcula ***y*** (la etiqueta) a partir de ***x*** (las características): ***f(x)=y***.

**Un ejemplo de clasificación simple**

Por ejemplo, supongamos que la observación consta de algunas medidas de un pingüino.



Específicamente, las medidas son las siguientes:

* El largo del pico del pingüino.
* La profundidad del pico del pingüino.
* El largo de la aleta del pingüino.
* El peso del pingüino.

En este caso, las características (***x***) son un vector de cuatro valores, o bien, matemáticamente expresado, ***x***=[x1,x2,x3,x4].

Imagine que la etiqueta que intenta predecir (***y***) es la especie del pingüino y que hay tres posibilidades de especie:

1. *Adelia*
2. *Papúa*
3. *Barbijo*

Este es un ejemplo de un problema de *clasificación*, en el que el modelo de Machine Learning debe predecir la clase más probable a la que pertenece la observación. Un modelo de clasificación logra esto mediante la predicción de una etiqueta que consta de la probabilidad de cada clase. Es decir, ***y*** es un vector de valores de tres probabilidades, una para cada una de las posibles clases: ***y***=[P(0),P(1),P(2)].

Puede entrenar el modelo de Machine Learning con las observaciones para las que ya conoce la etiqueta verdadera. Por ejemplo, puede tener las medidas de características siguientes para un pingüino de *Adelia*:

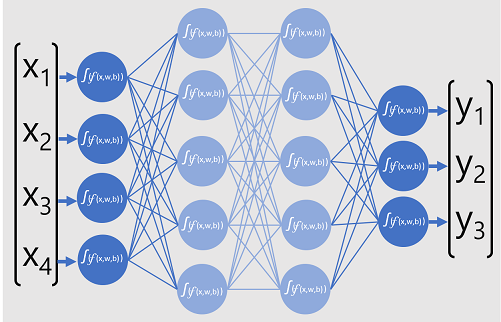
***x***=[37.3, 16.8, 19.2, 30.0]

Ya sabe que se trata de un ejemplo de un pingüino de *Adelia* (clase 0), por lo que una función de clasificación perfecta debe dar como resultado una etiqueta que indique una probabilidad del 100 % para la clase 0 y una probabilidad del 0 % para las clases 1 y 2:

***y***=[1, 0, 0]

**Un modelo de red neuronal profunda**

Por tanto, ¿cómo podríamos usar el aprendizaje profundo para crear un modelo de clasificación para el modelo de clasificación de los pingüinos? Veamos un ejemplo:



El modelo de red neuronal profunda para el clasificador consta de varias capas de neuronas artificiales. En este caso, hay cuatro capas:

* Una capa de *entrada* con una neurona para cada valor de entrada esperado (***x***).
* Dos capas denominadas *ocultas*, cada una con cinco neuronas.
* Una capa de *salida* con tres neuronas, una para cada valor de probabilidad de clase (***y***) que el modelo va a predecir.

Debido a la arquitectura en capas de la red, este tipo de modelo se conoce a veces como *perceptrón de varias capas*. Además, tenga en cuenta que todas las neuronas de las capas de entrada y ocultas están conectadas a todas los neuronas de las capas subsiguientes. Este es un ejemplo de una *red totalmente conectada*.

Al crear un modelo como este, debe definir un capa de entrada que admita la cantidad de características que va a procesar el modelo y una capa de salida que refleje la cantidad de salidas que espera que genere. Puede decidir la cantidad de capas ocultas que quiere incluir y cuántas neuronas en cada una de ellas, pero no tiene ningún control sobre los valores de entrada y salida de estas capas, ya que se determinan mediante el proceso de entrenamiento del modelo.

**Entrenamiento de una red neuronal profunda**

El proceso de entrenamiento de una red neuronal profunda se compone de varias iteraciones, llamadas *épocas*. Para la primera época, empiece por asignar valores de inicialización aleatorios para los valores de ponderación (***w***) y sesgo ***b***. Luego, el proceso es tal como se muestra a continuación:

1. Las características de las observaciones de datos con valores de etiqueta conocidos se envían a la capa de entrada. Por lo general, estas observaciones se agrupan en *lotes* (que se suelen denominar como *mini lotes*).
2. Luego, las neuronas aplican su función y, si está activada, pasan el resultado a la capa siguiente hasta que la capa de salida genera una predicción.
3. La predicción se compara con el valor conocido real y se calcula el monto de la varianza entre los valores previstos y los verdaderos (lo que se conoce como la *pérdida*).
4. En función de los resultados, se calculan los valores revisados de os valores de ponderación y sesgo a fin de reducir la pérdida y estos ajustes se *retropropagan* a la neuronas de las capas de la red.
5. En la época siguiente, el entrenamiento por lotes se repite con propagación hacia adelante con los valores de ponderación y sesgo corregidos, a fin de mejorar la precisión del modelo (mediante la reducción de la pérdida).

**Nota**

El procesamiento de las características de entrenamiento como un lote mejora la eficacia del proceso de entrenamiento mediante el procesamiento de varias observaciones de manera simultánea como una matriz de características con vectores de ponderaciones y sesgos. Las funciones algebraicas lineales que trabajan con matrices y vectores también se usan en el procesamiento de gráficos 3D. Como resultado, los equipos con unidades de procesamiento gráfico (GPU) ofrecen un rendimiento significativamente mejor para entrenar modelos de aprendizaje profundo que los equipos que solo tienen CPU (unidad de procesamiento central).

**Una visión más detallada de las funciones de pérdida y la retropropagación**

En la descripción anterior del proceso de entrenamiento de aprendizaje profundo se mencionó que la pérdida del modelo se calcula y utiliza para ajustar los valores de ponderación y sesgo. ¿Cómo funciona esto exactamente?

**Cálculo de la pérdida**

Supongamos que uno de los ejemplos transmitidos a través del proceso de entrenamiento contiene las características de un pingüino de *Adelia* (clase 0). La salida correcta de la red sería [1, 0, 0]. Ahora, supongamos que la red genera la salida [0.4, 0.3, 0.3]. Si las comparamos, podemos calcular una varianza absoluta para cada elemento (es decir, la distancia entre cada valor previsto y el valor que debería ser) como [0.6, 0.3, 0.3].

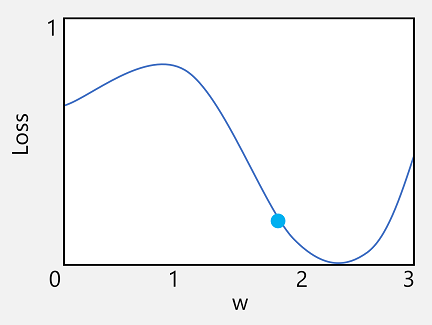
Dado que en la práctica se procesan varias observaciones, generalmente se agrega la varianza, por ejemplo, elevando cada valor de varianza a la potencia de 2 y luego calculando la media para terminar con un solo valor de pérdida promedio, como 0,18.

**Optimizadores**

Ahora viene la parte inteligente. La pérdida se calcula con una función que opera en los resultados de la capa final de la red, que también es una función. La capa de red final funciona en las salidas de las capas anteriores, que también son funciones. Por tanto, todo el modelo, desde la capa de entrada hasta el cálculo de la pérdida, no es más que una gran función anidada. Las funciones tienen algunas características muy útiles, entre las que se incluyen las siguientes:

* Puede conceptualizar una función como una línea trazada si compara su salida con cada una de sus variables.
* Puede usar un cálculo diferencial para calcular la *derivada* de la función en cualquier punto con respecto a sus variables.

Tomemos la primera de estas funcionalidades. Podemos trazar la línea de la función para mostrar cómo se compara un valor de ponderación individual con la pérdida y marcar en esa línea el punto en el que el valor de ponderación actual coincide con el valor de pérdida actual.



Ahora aplicaremos la segunda característica de una función. La derivada de una función para un punto determinado indica si la pendiente (o *gradiente*) de la salida de la función (en este caso, la pérdida) aumenta o disminuye con respecto a una variable de función (en este caso, el valor de ponderación). Una derivada positiva indica que la función está aumentando, mientras que una negativo indica que disminuye. En este caso, en el punto trazado del valor de ponderación actual, la función tiene una gradiente descendente. En otras palabras, el aumento de la ponderación disminuirá la pérdida.

Usamos un *optimizador* para aplicar este mismo truco a todas las variables de ponderación y sesgo del modelo y determinar en qué dirección se deben ajustar (hacia arriba o hacia abajo) para disminuir la cantidad total de pérdidas en el modelo. Hay varios algoritmos de optimización de uso común, que incluyen el *descenso por gradiente estocástico (SGD)*, la *tasa de aprendizaje adaptativo (ADADELTA)* y la *estimación de impulso adaptativo (Adam)*, entre otros; todos diseñados para determinar cómo ajustar las ponderaciones y los sesgos para reducir las pérdidas.

**Velocidad de aprendizaje**

Ahora, la siguiente pregunta obvia es: ¿cuánto debería ajustar el optimizador los valores de ponderaciones y sesgos? Si observa el trazado del valor de ponderación, puede ver que con un ligero aumento de la ponderación, la línea funcional cae (disminución de la pérdida). Sin embargo, si el valor aumenta demasiado, la línea de función comenzará a aumentar nuevamente y la pérdida puede incluso aumentar. Puede ser que después de la época siguiente resulte necesario disminuir la ponderación.

El tamaño del ajuste se controla mediante un parámetro que se establece para el entrenamiento, denominado *velocidad de aprendizaje*. Una velocidad de aprendizaje baja genera ajustes pequeños (por lo que puede tardar más épocas en minimizar la pérdida), mientras que una velocidad de aprendizaje alta genera ajustes grandes (de modo que es posible que no se alcance el mínimo).

**Creación de recursos de proceso**

Para entrenar e implementar modelos mediante el diseñador de Azure Machine Learning, necesita destinos de proceso a fin de ejecutar el proceso de entrenamiento. También usará estos destinos de proceso para probar el modelo entrenado después de su implementación.

**Creación de destinos de proceso**

Los destinos de proceso son recursos basados en la nube en los que se pueden ejecutar procesos de entrenamiento de modelos y exploración de datos.

En [Estudio de Azure Machine Learning](https://ml.azure.com/), expanda el panel izquierdo seleccionando las tres líneas de la parte superior izquierda de la pantalla. Vea la página **Proceso** (en **Administrar**). Debe administrar los destinos de proceso para las actividades de ciencia de datos en Studio. Puede crear cuatro tipos de recursos de proceso:

* **Instancias de proceso**: estaciones de trabajo de desarrollo que los científicos de datos pueden usar para trabajar con datos y modelos.
* **Clústeres de proceso**: clústeres escalables de máquinas virtuales para el procesamiento a petición de código de experimento.
* **Clústeres de inferencia**: destinos de implementación para servicios predictivos que usan los modelos entrenados.
* **Proceso asociado**: vínculos a recursos de proceso de Azure existentes, como clústeres de Azure Virtual Machines o Azure Databricks.

**Nota**

Las instancias de proceso y los clústeres se basan en imágenes de máquina virtual de Azure estándar. Para este módulo, se recomienda la imagen *Standard\_DS11\_v2* para lograr el equilibrio óptimo entre el costo y el rendimiento. Si la suscripción tiene una cuota que no incluye esta imagen, elija una imagen alternativa, pero tenga en cuenta que una imagen más grande puede incurrir en un costo mayor y una imagen más pequeña puede no ser suficiente para completar las tareas. Como alternativa, pida al administrador de Azure que amplíe la cuota.

1. En la pestaña **Instancias de proceso**, agregue una nueva instancia de proceso con los valores siguientes:
   * **Nombre del proceso**: *escriba un nombre único*
   * **Ubicación**: *tenga en cuenta el valor predeterminado. La ubicación de la instancia de proceso siempre se crea en la misma ubicación que el área de trabajo.*
   * **Tipo de máquina virtual**: CPU
   * **Tamaño de la máquina virtual**:
     + Elija **Seleccionar de entre todas las opciones**
     + Busque y seleccione **Standard\_DS11\_v2**
   * Seleccione **Crear**
2. Mientras se crea la instancia de proceso, cambie a la pestaña **Clústeres de proceso** y agregue un nuevo clúster de proceso con la configuración siguiente:
   * **Ubicación**: *seleccione la misma que el área de trabajo. Si esa ubicación no aparece, elija la más cercana.*
   * **Nivel de máquina virtual**: dedicado
   * **Tipo de máquina virtual**: CPU
   * **Tamaño de la máquina virtual**:
     + Elija **Seleccionar de entre todas las opciones**
     + Busque y seleccione **Standard\_DS11\_v2**
   * Seleccione **Siguiente**.
   * **Nombre del proceso**: *escriba un nombre único*
   * **Número mínimo de nodos**: 0
   * **Número máximo de nodos**: 2
   * **Segundos de inactividad antes de la reducción vertical**: 120
   * **Habilitar acceso SSH**: no seleccionado
   * Seleccione **Crear**

**Exploración de datos**

Para entrenar un modelo de regresión, necesita un conjunto de datos que incluya *características* históricas, características de la entidad para la que quiere realizar una predicción. También necesita valores de *etiqueta* conocidos, el valor numérico que quiere para entrenar un modelo que va a predecir.

**Crear una canalización**

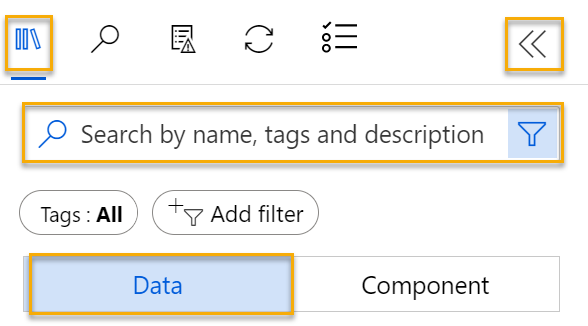
Para usar el diseñador de Azure Machine Learning, cree una *canalización* que usará para entrenar un modelo de Machine Learning. Esta canalización se inicia con el conjunto de datos desde el que quiere entrenar el modelo.

1. En [Estudio de Azure Machine Learning](https://ml.azure.com/), expanda el panel izquierdo seleccionando las tres líneas de la parte superior izquierda de la pantalla. Vea la página **Diseñador** (en **Autor**) y seleccione **+** para crear una canalización.
2. En la parte superior derecha de la pantalla, seleccione **Configuración**. Si el panel de **Configuración** no está visible, seleccione el icono **⚙** situado junto al nombre de la canalización en la parte superior.
3. En **Configuración**, tendrá que especificar un destino de proceso en el que ejecutar la canalización. En **Select compute type (Seleccionar tipo de proceso)**, seleccione **Clúster de proceso**. Después, en **Select Azure ML *compute-type* (Seleccionar tipo de proceso de Azure ML)**, seleccione el clúster de proceso que creó anteriormente.
4. En **Configuración**, en **Draft Details (Detalles del borrador)**, cambie el nombre del borrador (**Pipeline-Created-on-*date***) a **Auto Price Training**.
5. Seleccione el *icono de cierre* en la esquina superior derecha del panel de **Configuración** para cerrarlo.

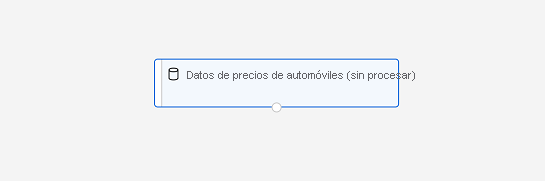
**Adición y exploración de un conjunto de datos**

En este módulo, entrenará un modelo de regresión que predice el precio de un automóvil en función de sus características. Azure Machine Learning incluye un conjunto de datos de ejemplo que puede usar para este modelo.

1. Junto al nombre de la canalización en el lado izquierdo, seleccione el botón **>>** para expandir el panel si está contraído. El panel debería abrirse de forma predeterminada en el **Panel biblioteca**, indicado por el icono de libros situado en la parte superior del panel. Tenga en cuenta que hay una barra de búsqueda para buscar activos. Debajo de los botones **Etiquetas** y **Filtro**, hay dos iconos juntos. Mantenga el mouse sobre el primer icono de cilindro (a la izquierda) para ver que representa los **Data Assets (activos de datos)**. Mantenga el mouse sobre el segundo icono del gráfico (a la derecha) para ver que representa los **Componentes**.



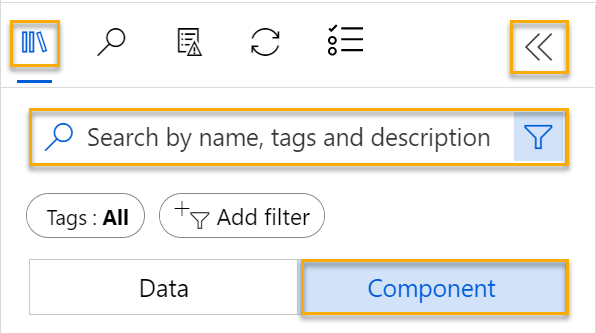
1. Haga clic en el icono de cilindro para **Data Assets (activos de datos)**. Arrastre el conjunto de datos **Datos de precio de automóviles (Sin formato)** al lienzo.
2. Haga clic con el botón derecho (Ctrl+clic en Mac) en el conjunto de datos **Automobile price data (Raw)** en el lienzo y, en el menú **Salidas**, haga clic en **Salida del conjunto de datos** en el icono gráfico *Vista previa de datos*.
3. Revise el esquema de los datos. Tenga en cuenta que puede ver las distribuciones de las distintas columnas como histogramas.
4. Desplácese a la derecha del conjunto de datos hasta que vea la columna **Price**, que es la etiqueta que predice el modelo.
5. Seleccione el encabezado de la columna **price** y vea los detalles que se muestran en el panel de la derecha. Incluyen varias estadísticas para los valores de columna y un histograma con su distribución.
6. Desplácese de nuevo a la izquierda y seleccione el encabezado de la columna **normalized-losses** (Pérdidas normalizadas). Después, revise las estadísticas de esta columna. Tenga en cuenta que faltan bastantes valores en esta columna. Los valores que faltan limitan la utilidad de la columna para predecir la etiqueta **price**, por lo que es posible que quiera excluirla del entrenamiento.
7. Vea las estadísticas de las columnas **bore**, **stroke** y **horsepower**. Observe el número de valores que faltan. Estas columnas tienen menos valores que faltan que **normalized-losses**, por lo que pueden seguir siendo útiles para predecir el valor **price** cuando se excluyan del entrenamiento las filas en las que faltan valores.
8. Compare los valores de la columnas **stroke**, **peak-rpm** y **city-mpg**. Estas columnas se miden en diferentes escalas y es posible que los valores más grandes para **peak-rpm** puedan sesgar el algoritmo de entrenamiento y crear una dependencia excesiva en esta columna, en comparación con las de valores inferiores, como **stroke**. Normalmente, los científicos de datos mitigan este posible sesgo mediante la *normalización* de las columnas numéricas para que estén en escalas similares.
9. Cierre la ventana **Visualización de los resultados de Automobile price data (Raw)** para poder ver el conjunto de datos en el lienzo de esta forma:



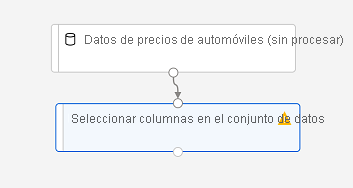
**Adición de transformaciones de datos**

Normalmente se aplican transformaciones de datos para preparar los datos para el modelado. En el caso de los datos de precios de automóviles, debe agregar transformaciones para solucionar los problemas que ha identificado al explorar los datos.

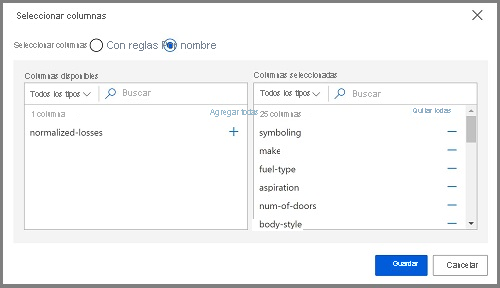
1. En el **Panel biblioteca** de la izquierda, haga clic en el icono de cuadrados para acceder a **Componentes**, que contiene una amplia gama de módulos que puede usar para la transformación de datos y el entrenamiento del modelo. También puede utilizar la barra de búsqueda para localizar los módulos con rapidez.



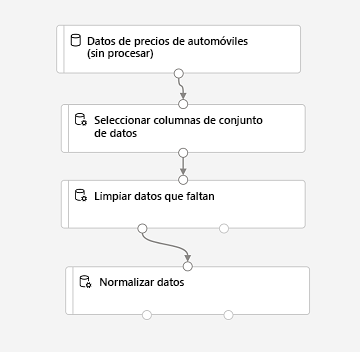
1. Localice un módulo **Seleccionar columnas del conjunto de datos** y arrástrelo al lienzo, debajo del módulo **Datos de precio de automóviles (Sin formato)**. Después, conecte la salida de la parte inferior del módulo **Automobile price data (Raw)** a la entrada de la parte superior del módulo **Seleccionar columnas del conjunto de datos**, de esta forma:



1. Seleccione el módulo **Seleccionar columnas del conjunto de datos** y, en el panel **Configuración** de la derecha, seleccione **Editar columna**. Después, en la ventana **Seleccionar columnas**, seleccione **Por nombre** y use los vínculos **+** para agregar todas las columnas menos **normalized-losses**, de esta forma:

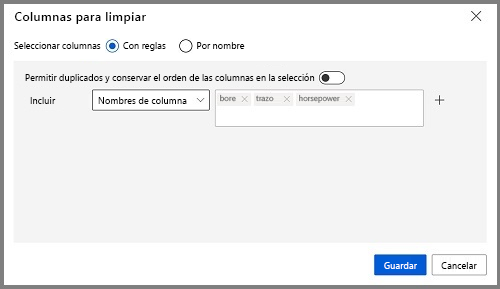


En el resto de este ejercicio, aprenderá paso a paso cómo crear una canalización similar a la siguiente:

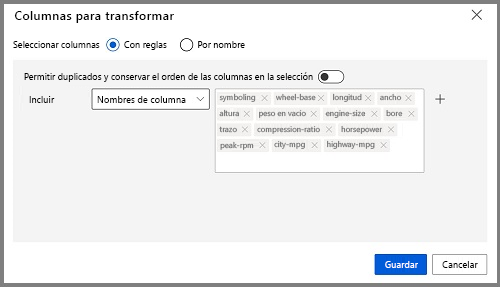


Siga los pasos restantes, y use la imagen como referencia a medida que agrega y configura los módulos necesarios.

1. Busque un módulo **Limpiar datos que faltan** y colóquelo en el módulo **Seleccionar columnas del conjunto de datos** en el lienzo. Después, conecte la salida del módulo **Seleccionar columnas del conjunto de datos** a la entrada del módulo **Limpiar datos que faltan**.
2. Seleccione el módulo **Limpiar datos que faltan** y, en el panel Configuración de la derecha, haga clic en **Editar columna**. Después, en la ventana **Seleccionar columnas**, seleccione **Con reglas**, en la lista **Incluir** seleccione **Nombres de columna** y, en el cuadro de nombres de columna, escriba **bore**, **stroke** y **horsepower** de esta forma:



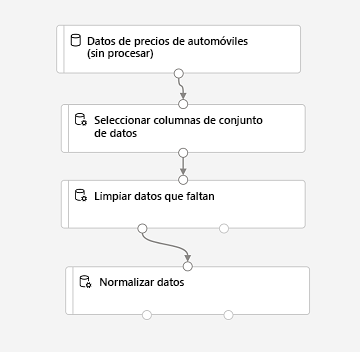
1. Con el módulo **Limpiar datos que faltan** aún seleccionado, en el panel Configuración, establezca las siguientes opciones de configuración:
   * **Relación mínima de valores que faltan**: 0,0
   * **Relación máxima de valores que faltan**: 1,0
   * **Modo de limpieza**: quitar toda la fila
2. Localice un módulo **Normalizar datos** y arrástrelo al lienzo, debajo del módulo **Limpiar datos que faltan**. Después, conecte la salida del módulo **Limpiar datos que faltan** a la entrada del módulo **Normalizar datos**.
3. Seleccione el módulo **Normalize Data** (Normalizar datos) y vea su configuración. Tenga en cuenta que es necesario especificar el método de transformación y las columnas que se transformarán. Después, establezca la transformación en **MinMax**. Aplique una regla para editar las columnas a fin de incluir los **Nombres de columna** siguientes:
   * **symboling**
   * **wheel-base**
   * **length**
   * **width**
   * **height**
   * **curb-weight**
   * **engine-size**
   * **bore**
   * **stroke**
   * **compression-ratio**
   * **horsepower**
   * **peak-rpm**
   * **city-mpg**
   * **highway-mpg**



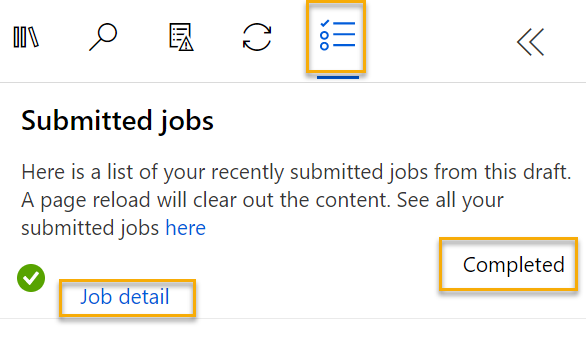
**Ejecución de la canalización**

Para aplicar las transformaciones de datos, debe ejecutar la canalización como un experimento.

1. Asegúrese de que la canalización sea similar a esta imagen:

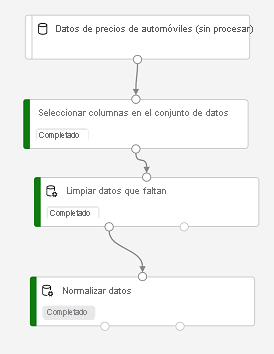


1. Seleccione **Enviar** y ejecute la canalización como un nuevo experimento denominado **mslearn-auto-training** en el clúster de proceso.
2. Espere a que finalice la ejecución, lo que puede tardar 5 minutos o más.



Observe que el panel izquierdo está ahora en el panel **Trabajos enviados**. Sabrá cuándo se completa la ejecución porque el estado del trabajo cambiará a **Completado**.

1. Una vez completada la ejecución, haga clic en **Detalles del trabajo**. Se le dirigirá a otra ventana que mostrará los módulos de la siguiente manera:



**Visualización de los datos transformados**

Ahora el conjunto de datos está preparado para el entrenamiento del modelo.

1. Seleccione el módulo **Normalize Data** (Normalizar datos) completado y, en el panel **Configuración** de la derecha, en la pestaña **Salidas y registros**, seleccione el icono **Vista previa de los datos** del **Conjunto de datos transformado**.
2. Visualice los datos. Observe que se ha quitado la columna **normalized-losses**, que todas las filas contienen datos para **bore**, **stroke** y **horsepower**, y que las columnas numéricas que ha seleccionado se han normalizado a una escala común.
3. Cierre la visualización del resultado normalizado de los datos.

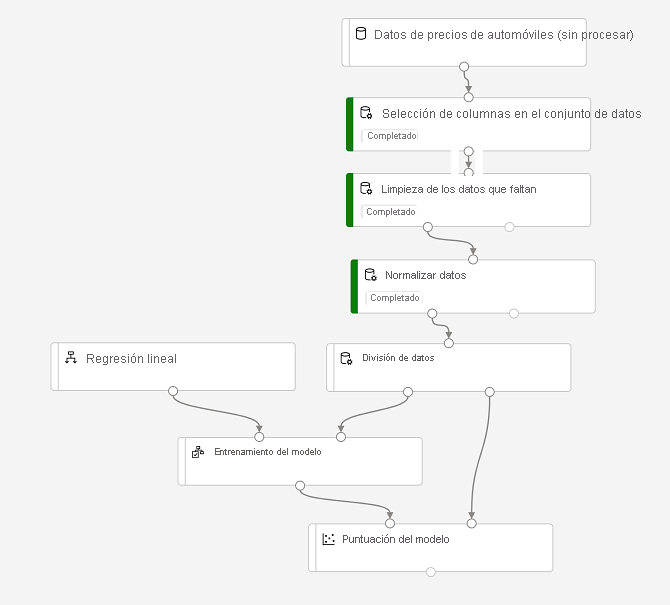
**Creación y ejecución de una canalización de entrenamiento**

Después de haber usado transformaciones de datos para preparar los datos, puede usarlos para entrenar un modelo de Machine Learning.

**Adición de módulos de entrenamiento**

Es habitual entrenar el modelo con un subconjunto de los datos, a la vez que se retienen algunos con los que probar el modelo entrenado. Esto le permite comparar las etiquetas que predice el modelo con las etiquetas conocidas reales del conjunto de datos original.

En este ejercicio, aprenderá paso a paso cómo ampliar la canalización **Entrenamiento de precios automático** como se muestra aquí:



Siga estos pasos y use la imagen anterior como referencia a medida que agrega y configura los módulos necesarios.

1. Abra la canalización **Entrenamiento de precios automático** que ha creado en la unidad anterior si todavía no está abierta.
2. En el **Panel biblioteca** de la izquierda, en la sección **Componentes**, localice y arrastre un módulo **Dividir datos** en el lienzo debajo del módulo **Normalizar Datos**. Después, conecte la salida *Conjunto de datos transformado* (izquierda) del módulo **Normalizar los datos** a la entrada del módulo **Dividir datos**.

**Sugerencia**

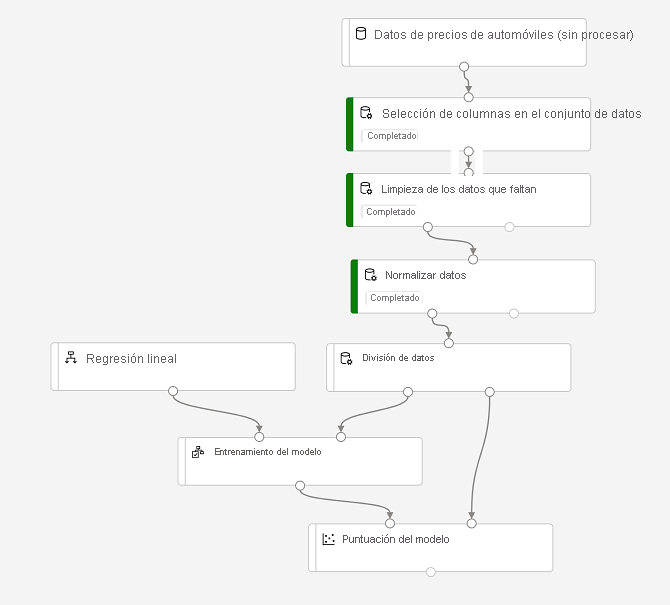
Use la barra de búsqueda para localizar los módulos con rapidez.

1. Seleccione el módulo **Dividir datos** y configure sus valores como se indica a continuación:
   * **Modo de división**: dividir filas
   * **Fracción de filas del primer conjunto de datos de salida**: 0,7
   * **Valor de inicialización aleatorio**: 123
   * **División estratificada**: Falso
2. Expanda la sección **Entrenamiento del modelo** del panel de la izquierda y arrastre el módulo **Entrenar modelo** al lienzo, bajo el módulo **Dividir datos**. Después, conecte la salida de *Conjunto de datos de resultados 1* (izquierda) del módulo **Dividir datos** a la entrada *Conjunto de datos* (derecha) del módulo **Entrenar modelo**.
3. El modelo que se va a entrenar predecirá el valor **price**, por lo que debe seleccionar el módulo **Entrenar modelo** y modificar su configuración para establecer la **columna Label** (Etiqueta) en **price** (con la misma ortografía y las mismas mayúsculas y minúsculas).
4. La etiqueta **price** que predecirá el modelo es un valor numérico, por lo que es necesario entrenar el modelo mediante un algoritmo de *regresión*. Expanda la sección **Algoritmos de aprendizaje automático** y, en **Regresión**, arrastre un módulo **Regresión lineal** al lienzo, a la izquierda del módulo **Dividir datos** y por encima del módulo **Entrenar modelo**. Después, conecte su salida a la entrada **Modelo no entrenado** (izquierda) del módulo **Entrenar modelo**.

**Nota**

Puede usar varios algoritmos para entrenar un modelo de regresión. Para ayudarle a elegir uno, eche un vistazo a la [**Hoja de referencia rápida de algoritmos de aprendizaje automático del diseñador de Azure Machine Learning**](https://aka.ms/mlcheatsheet).

1. Para probar el modelo entrenado, es necesario usarlo para *puntuar* el conjunto de datos de validación que se ha conservado al dividir los datos originales, es decir, para predecir las etiquetas de las características del conjunto de datos de validación. Expanda la sección **Puntuación y evaluación del modelo** y arrastre un módulo **Puntuar modelo** al lienzo, debajo del módulo **Entrenar modelo**. Después, conecte la salida del módulo **Entrenar modelo** a la entrada **Modelo entrenado** (izquierda) del módulo **Puntuar modelo** y arrastre la salida **Conjunto de datos de resultados 2** (derecha) del módulo **Dividir datos** a la entrada **Conjunto de datos** (derecha) del módulo **Puntuar modelo**.
2. Asegúrese de que la canalización es similar a la imagen siguiente:



**Ejecución de la canalización de entrenamiento**

Ya está a punto para ejecutar la canalización de entrenamiento y entrenar el modelo.

1. Seleccione **Enviar** y ejecute la canalización mediante el experimento existente denominado **mslearn-auto-training**.
2. Espere a que se complete la ejecución del experimento. Esto puede tardar cinco minutos o más.
3. Cuando se haya completado la ejecución del experimento, seleccione el módulo **Score Model** (Puntuar modelo) y, en el panel Configuración, en la pestaña **Resultados y registros**, en la sección **Scored dataset** (Conjunto de datos puntuado) de **Data outputs** (Salidas de datos), use el icono **Vista previa de datos** para ver los resultados.
4. Desplácese a la derecha y observe que junto a la columna **price** (que contiene los valores reales conocidos de la etiqueta) hay una nueva columna denominada **Etiquetas puntuadas**, que contiene los valores de etiqueta pronosticados.
5. Cierre la ventana **Visualización del resultado de Puntuar modelo**.

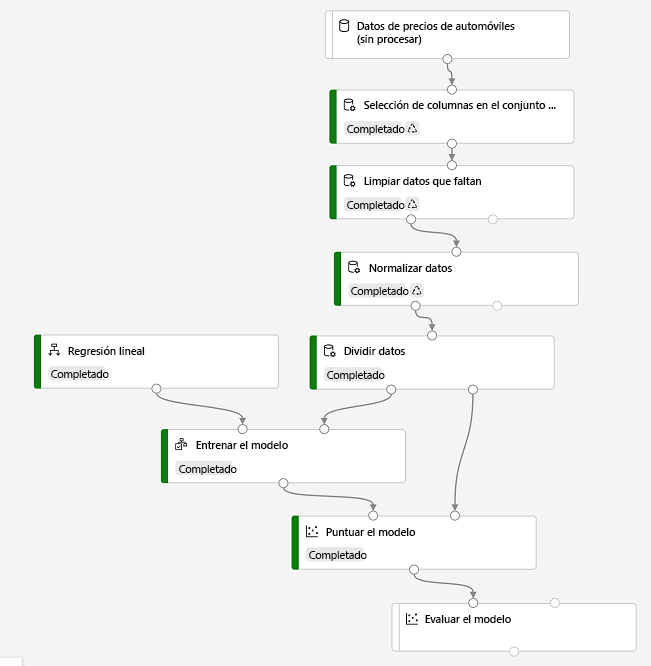
El modelo predice valores para la etiqueta **price**, ¿pero qué fiabilidad tienen sus predicciones? Para valorarlo, tendrá que evaluar el modelo.

**Evaluación de un modelo de regresión**

Para evaluar un modelo de regresión, puede comparar simplemente las etiquetas pronosticadas con las reales del conjunto de datos de validación como retención durante el entrenamiento, pero sería un proceso impreciso y no proporciona una métrica sencilla que pueda usar para comparar el rendimiento de varios modelos.

**Adición de un módulo Evaluar modelo**

1. Abra la canalización **Entrenamiento de precios automático** que ha creado en la unidad anterior si todavía no está abierta.
2. En el panel de la izquierda, en la sección **Puntuación y evaluación del modelo**, arrastre un módulo **Evaluar modelo** al lienzo, bajo el módulo **Puntuar modelo**, y conecte la salida del módulo **Puntuar modelo** a la entrada **Conjunto de datos puntuado** (izquierda) del módulo **Evaluar modelo**.
3. Asegúrese de que la canalización es similar a la siguiente:



1. Seleccione **Enviar** y ejecute la canalización mediante el experimento existente denominado **mslearn-auto-training**.
2. Espere a que se complete la ejecución del experimento.
3. Cuando se haya completado la ejecución del experimento, seleccione el módulo **Evaluate Model** (Evaluar modelo) y, en el panel Configuración, en la pestaña **Resultados y registros**, en la sección **Resultados de la evaluación** de **Data outputs** (Salidas de datos), use el icono **Vista previa de datos** para ver los resultados. Se incluyen las métricas de rendimiento de regresión siguientes:
   * **Error medio absoluto (EAM)**: la diferencia media entre los valores pronosticados y los reales. Este valor se basa en las mismas unidades que la etiqueta, en este caso, dólares. Cuanto menor sea este valor, mejor será la predicción del modelo.
   * **Raíz del error cuadrático medio (RECM)**: raíz cuadrada de la diferencia cuadrática media entre valores predichos y verdaderos. El resultado es una métrica basada en la misma unidad que la etiqueta (dólares). En comparación con el valor EAM (anterior), una diferencia mayor indica una mayor desviación en los errores individuales (por ejemplo, algunos errores son muy pequeños, mientras que otros son grandes).
   * **Error cuadrático relativo (ESR)**: una métrica relativa entre 0 y 1 en función del cuadrado de las diferencias entre los valores pronosticados y los reales. Cuanto más cercano a 0 sea el valor de esta métrica, mejor funciona el modelo. Como esta métrica es relativa, se puede usar para comparar modelos en los que las etiquetas se encuentran en unidades distintas.
   * **Error absoluto relativo (EAR)**: una métrica relativa entre 0 y 1 en función de las diferencias absolutas entre los valores pronosticados y los reales. Cuanto más cercano a 0 sea el valor de esta métrica, mejor funciona el modelo. Al igual que ESR, esta métrica se puede usar para comparar modelos en los que las etiquetas se encuentran en unidades distintas.
   * **Coeficiente de determinación (R2)**: esta métrica se suele denominar *R cuadrado* y resume la cantidad de la varianza entre los valores reales y los previstos que se explica en el modelo. Cuanto más cercano a 1 sea esta valor, mejor funciona el modelo.
4. Cierre la ventana **Visualización del resultado de Evaluar modelo**.

Puede probar otro algoritmo de regresión y comparar los resultados si conecta las mismas salidas del módulo **Dividir datos** a un segundo módulo **Entrenar modelo** (con otro algoritmo) y un segundo módulo **Puntuar modelo**; y, después, conecta las salidas de los dos módulos **Puntuar modelo** al mismo módulo **Evaluar modelo** para realizar una comparación en paralelo.

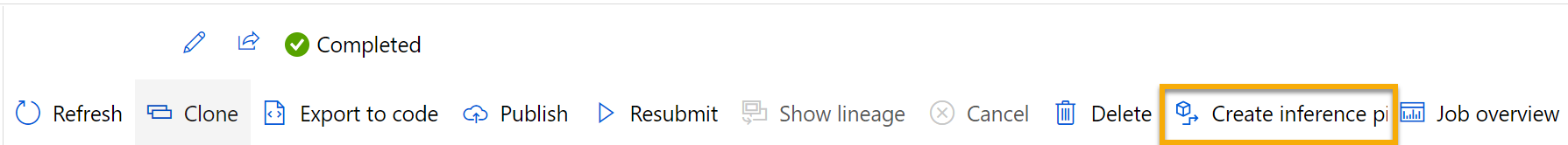
Cuando haya identificado un modelo con métricas de evaluación que se ajusten a las necesidades, puede prepararse para usar ese modelo con nuevos datos.

**Creación de una canalización de inferencia**

Después de crear y ejecutar una canalización para entrenar el modelo, necesita una segunda canalización conocida como canalización de inferencia. La canalización de inferencia realiza las mismas transformaciones de datos que la primera canalización para los datos *nuevos*. Después, usa el modelo entrenado para *deducir* o predecir valores de etiqueta en función de sus características. Este modelo formará la base de un servicio predictivo que puede publicar para que lo usen las aplicaciones.

**Creación y ejecución de una canalización de inferencia**

1. En Estudio de Azure Machine Learning, expanda el panel izquierdo seleccionando las tres líneas de la parte superior izquierda de la pantalla. Haga clic en **Trabajos** (en **Activos**) para ver todos los trabajos que ha ejecutado. Seleccione el experimento **mslearn-auto-training** y, después, la canalización **mslearn-auto-training**.
2. Busque el menú situado encima del lienzo y haga clic en **Create inference pipeline (Crear canalización de inferencia)**. Es posible que tenga que acceder a la pantalla completa y hacer clic en el icono de tres puntos **...** en la esquina superior derecha de la pantalla para buscar **Create inference pipeline (Crear canalización de inferencia)** en el menú.

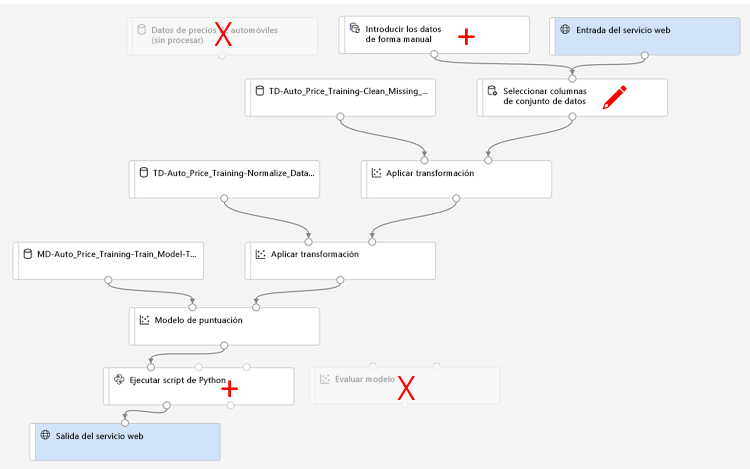


1. En la lista desplegable **Crear canalización de inferencia**, haga clic en **Canalización de inferencia en tiempo real**. Después de unos segundos, se abrirá una versión nueva de la canalización denominada **Entrenamiento de precios automático-inferencia en tiempo real**.

*Si la canalización no incluye los módulos****Entrada de servicio web****y****Salida de servicio web****, vuelva a la página****Diseñador****y, después, abra de nuevo la canalización****Entrenamiento de precios automático-inferencia en tiempo real****.*

1. Cambie el nombre de la nueva canalización a **Predicción de precios automática** y después revísela. Contiene una entrada de servicio web para los nuevos datos que se van a enviar y una salida de servicio web para devolver los resultados. Algunas de las transformaciones y los pasos de entrenamiento forman parte de esta canalización. El modelo entrenado se usará para puntuar los nuevos datos.

Va a realizar los cambios siguientes en la canalización de inferencia en los pasos 5 a 9 siguientes:



Use la imagen como referencia a medida que modifique la canalización en los pasos siguientes.

1. La canalización de inferencia supone que los datos nuevos coincidirán con el esquema de los datos de entrenamiento originales, por lo que se incluye el conjunto de datos **Automobile price data (Raw)** de la canalización de entrenamiento. Pero estos datos de entrada incluyen la etiqueta **price** que predice el modelo, lo que no resulta intuitivo incluir en los nuevos datos de automóvil para los que todavía no se ha realizado una predicción del precio. Elimine este módulo y reemplácelo por un módulo **Escribir los datos manualmente** de la sección **Entrada y salida de datos**, que contiene los siguientes datos CSV, que incluyen valores de características sin etiquetas para tres automóviles (copie y pegue todo el bloque de texto):

CSVCopiar

symboling,normalized-losses,make,fuel-type,aspiration,num-of-doors,body-style,drive-wheels,engine-location,wheel-base,length,width,height,curb-weight,engine-type,num-of-cylinders,engine-size,fuel-system,bore,stroke,compression-ratio,horsepower,peak-rpm,city-mpg,highway-mpg

3,NaN,alfa-romero,gas,std,two,convertible,rwd,front,88.6,168.8,64.1,48.8,2548,dohc,four,130,mpfi,3.47,2.68,9,111,5000,21,27

3,NaN,alfa-romero,gas,std,two,convertible,rwd,front,88.6,168.8,64.1,48.8,2548,dohc,four,130,mpfi,3.47,2.68,9,111,5000,21,27

1,NaN,alfa-romero,gas,std,two,hatchback,rwd,front,94.5,171.2,65.5,52.4,2823,ohcv,six,152,mpfi,2.68,3.47,9,154,5000,19,26

1. Conecte el nuevo módulo **Escribir los datos manualmente** a la misma entrada **conjunto de datos** del módulo **Seleccionar columnas del conjunto de datos** como **Entrada de servicio web**.
2. Ahora que ha cambiado el esquema de los datos entrantes para excluir el campo**price**, tendrá que quitar todos los usos explícitos de este campo en los módulos restantes. Seleccione el módulo **Seleccionar columnas del conjunto de datos** y, después, en el panel Configuración, edite las columnas para quitar el campo **price**.
3. La canalización de inferencia incluye el módulo **Evaluar modelo**, que no resulta útil al realizar predicciones a partir de los datos nuevos, por lo que puede eliminarlo.
4. En la salida del módulo **Puntuar modelo** se incluyen todas las características de entrada y la etiqueta predicha. Para modificar la salida de forma que solo incluya la predicción:
   * Elimine la conexión entre el módulo **Puntuar modelo** y **Salida de servicio web**.
   * Agregue un módulo **Ejecutar script de Python** desde la sección **Python Language (Lenguaje Python)** y reemplace todo el script de Python predeterminado por el código siguiente (que solo selecciona la columna **Scored Labels** y le cambia el nombre por **predicted\_price**):

PythonCopiar

import pandas as pd

def azureml\_main(dataframe1 = None, dataframe2 = None):

scored\_results = dataframe1[['Scored Labels']]

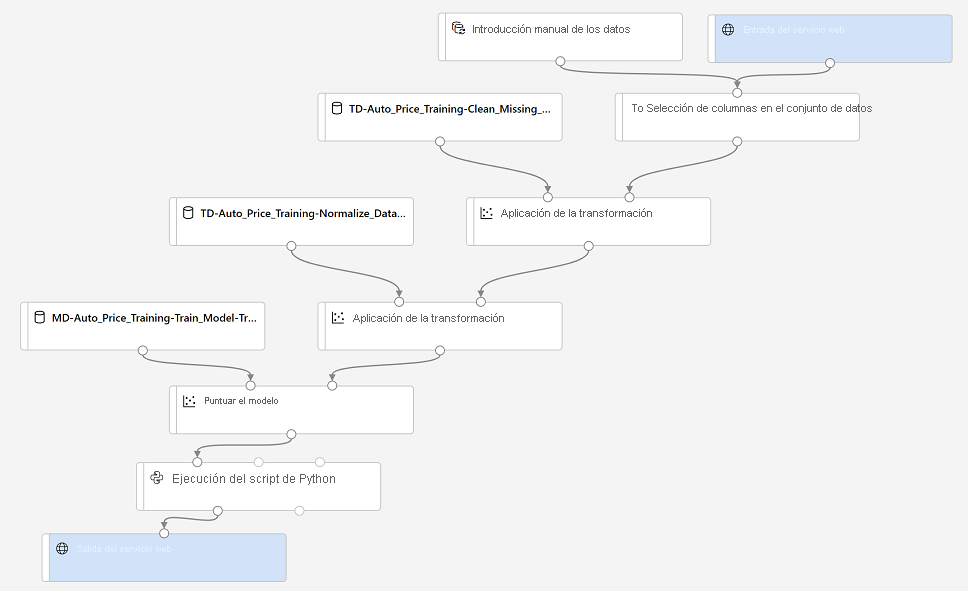
scored\_results.rename(columns={'Scored Labels':'predicted\_price'},

inplace=True)

return scored\_results

* + Conecte la salida del módulo **Puntuar modelo** a la entrada **Conjunto de datos 1** (en el extremo izquierdo) del módulo **Ejecutar script de Python**, y conecte la salida del módulo **Ejecutar script de Python** a **Salida del servicio web**.

1. Compruebe que la canalización tiene un aspecto similar a la imagen siguiente:



1. Envíe la canalización como un nuevo experimento denominado **mslearn-auto-inference** en el clúster de proceso. El experimento puede tardar un tiempo en ejecutarse.
2. Una vez que se haya completado la canalización, seleccione el módulo **Ejecutar script de Python** y, en el panel Configuración, en la pestaña **Salida y registros**, visualice el **Conjunto de datos de resultados** para ver los precios pronosticados de los tres automóviles de los datos de entrada.
3. Cierre la ventana de visualización.

La canalización de inferencia predice los precios de los automóviles en función de sus características. Ya está a punto para publicar la canalización a fin de que las aplicaciones cliente la puedan usar.

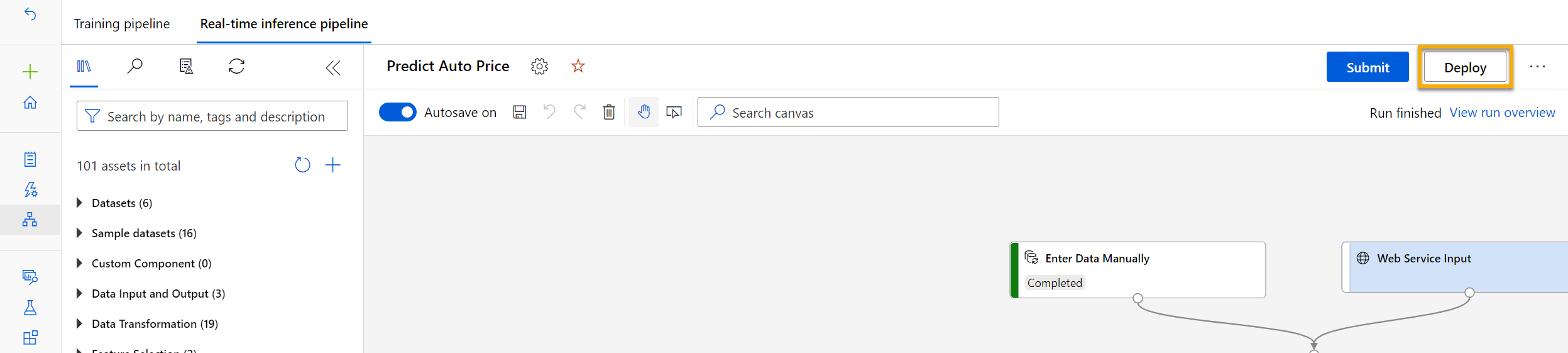
**Implementación de un servicio predictivo**

Después de crear y probar una canalización de inferencia para la inferencia en tiempo real, puede publicarla como un servicio para que lo usen las aplicaciones cliente.

**Nota**

En este ejercicio, implementará el servicio web en una instancia de Azure Container (ACI). Este tipo de proceso se crea dinámicamente y resulta útil para el desarrollo y las pruebas. Para producción, debe crear un *clúster de inferencia*, que proporciona un clúster de Azure Kubernetes Service (AKS) que ofrece mejor escalabilidad y seguridad.

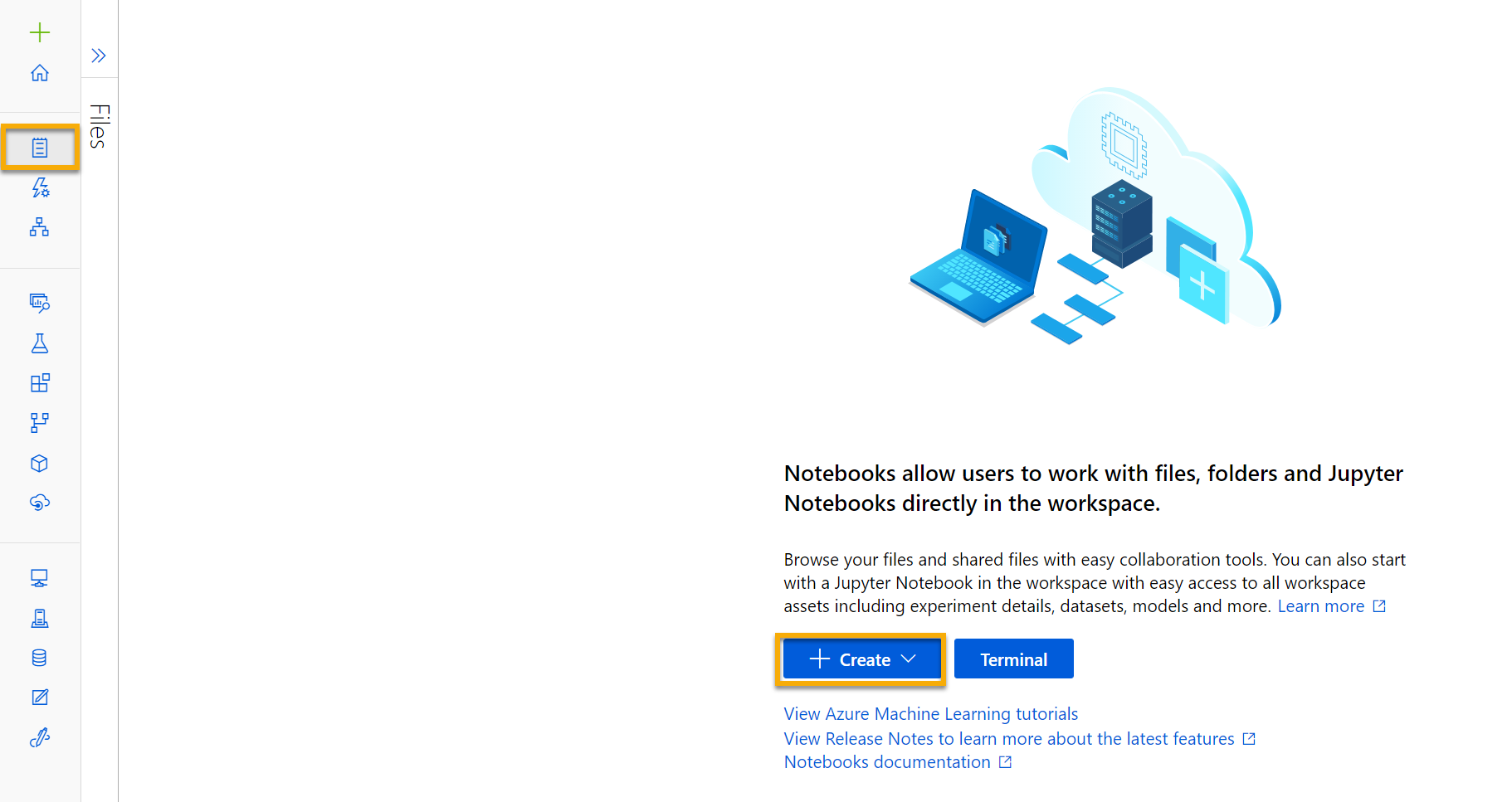
**Implementación de un servicio**



1. Vea la canalización de inferencia **Predicción de precios automática** que ha creado en la unidad anterior.
2. En la parte superior derecha, seleccione **Implementar** e implemente un nuevo punto de conexión en tiempo real con la siguiente configuración:
   * **Nombre**: predict-auto-price
   * **Descripción**: regresión de precios automática.
   * **Tipo de proceso**: instancia de Azure Container.
3. Espere a que se implemente el servicio web; esto puede tardar varios minutos. El estado de implementación se muestra en la parte superior izquierda de la interfaz del diseñador.

**Probar el servicio**

1. En la página **Puntos de conexión**, abra el punto de conexión en tiempo real **predict-auto-price**.
2. Cuando se abra el punto de conexión **predict-auto-price**, examine la pestaña **Consumir** y observe la información siguiente. La necesitará para conectarse al servicio implementado desde una aplicación cliente.
   * El punto de conexión REST para el servicio
   * La clave principal para el servicio
3. Observe que puede usar el vínculo ⧉ situado junto a estos valores para copiarlos en el Portapapeles.
4. Con la página **Consumir** de la página del servicio **predict-auto-price** abierta en el explorador, abra una nueva pestaña y una segunda instancia de [Azure Machine Learning Studio](https://ml.azure.com/). Después, en la pestaña nueva, vea la página **Cuadernos** (en **Autor**).
5. Vaya al panel izquierdo y haga clic en **Cuadernos**. Después, use el botón **🗋** para crear un archivo con la siguiente configuración:
   * **Ubicación del archivo**: Users/*su nombre de usuario*
   * **Nombre de archivo**: Test-Autos.ipynb
   * **Tipo de archivo**: cuaderno
   * **Sobrescribir si ya existe**: seleccionado



1. Cuando se haya creado el cuaderno, asegúrese de que la instancia de proceso que ha creado antes esté seleccionada en el cuadro **Proceso** y que tiene el estado **En ejecución**.
2. Use el botón **≪** para contraer el panel del explorador de archivos y así disponer de más espacio para centrarse en la pestaña del cuaderno **Test-Autos.ipynb**.
3. En la celda rectangular que se ha creado en el cuaderno, pegue el código siguiente:

PythonCopiar

endpoint = 'YOUR\_ENDPOINT' #Replace with your endpoint

key = 'YOUR\_KEY' #Replace with your key

import urllib.request

import json

import os

# Prepare the input data

data = {

"Inputs": {

"WebServiceInput0":

[

{

'symboling': 3,

'normalized-losses': None,

'make': "alfa-romero",

'fuel-type': "gas",

'aspiration': "std",

'num-of-doors': "two",

'body-style': "convertible",

'drive-wheels': "rwd",

'engine-location': "front",

'wheel-base': 88.6,

'length': 168.8,

'width': 64.1,

'height': 48.8,

'curb-weight': 2548,

'engine-type': "dohc",

'num-of-cylinders': "four",

'engine-size': 130,

'fuel-system': "mpfi",

'bore': 3.47,

'stroke': 2.68,

'compression-ratio': 9,

'horsepower': 111,

'peak-rpm': 5000,

'city-mpg': 21,

'highway-mpg': 27

},

],

},

"GlobalParameters": {

}

}

body = str.encode(json.dumps(data))

headers = {'Content-Type':'application/json', 'Authorization':('Bearer '+ key)}

req = urllib.request.Request(endpoint, body, headers)

try:

response = urllib.request.urlopen(req)

result = response.read()

json\_result = json.loads(result)

y = json\_result["Results"]["WebServiceOutput0"][0]

print(y)

except urllib.error.HTTPError as error:

print("The request failed with status code: " + str(error.code))

# Print the headers to help debug the error

print(error.info())

print(json.loads(error.read().decode("utf8", 'ignore')))

**Nota**

No se preocupe demasiado por los detalles del código. Simplemente envía los detalles de un automóvil y usa el servicio **predict-auto-price** que ha creado para obtener un precio pronosticado.

1. Cambie a la pestaña del explorador que contiene la página **Consumir** para el servicio **predict-auto-price** y copie el punto de conexión REST del servicio. Después, vuelva a la pestaña que contiene el cuaderno y pegue la clave en el código, para reemplazar YOUR\_ENDPOINT.
2. Cambie a la pestaña del explorador que contiene la página **Consumir** para el servicio **predict-auto-price** y copie la clave principal del servicio. Luego vuelva a la pestaña que contiene el cuaderno y pegue la clave en el código, para reemplazar YOUR\_KEY.
3. Guarde el cuaderno. Después, use el botón **▷** situado junto a la celda para ejecutar el código.
4. Después de ejecutar el código, desplácese hacia abajo hasta la parte inferior de la pantalla. Debería ver la salida **"predicted\_price"**. La salida es el precio previsto para un vehículo con las características de entrada concretas especificadas en el código.