Machine Learning

Regresión lineal y descenso de gradiente

Jesús Medina

Índice general

Capítulo 1

Regresión lineal

La regresión lineal es una técnica estadística y matemática empleada para realizar predicciones a partir de un modelo lineal. El objetivo es encontrar una función que describa, de la forma más simple posible, la relación entre una variable dependiente y una o más variables independientes, basándose en un conjunto de datos observados.

Dado que en la práctica las observaciones suelen ser discretas y limitadas, es común que no contemos con registros exactos para cada posible valor de las variables independientes. En esos casos, la regresión lineal nos permite construir una función aproximada cuya trayectoria se asemeje, en la medida de lo posible, al comportamiento de los datos.

Cuando se trabaja con una sola variable independiente, el modelo resultante es una recta que intenta ajustarse a los puntos observados. Este caso se conoce como **regresión lineal simple**, y se expresa mediante una ecuación de la forma:

$$\hat{y} = \theta_0 + \theta_0 x$$

donde θ_1 representa la pendiente de la recta, θ_0 es la intersección con el eje vertical, y $\hat{y}$ es el valor estimado de la variable dependiente. El modelo busca que dicha recta pase lo más cerca posible de todos los puntos del conjunto de datos, minimizando alguna medida del error, como el error cuadrático medio.

Si se dispone de varias variables independientes, el modelo se extiende a lo que se conoce como **regresión lineal múltiple**. En este caso, se ajusta un plano o hiperplano (dependiendo del número de variables) que represente la relación lineal entre la variable dependiente y las independientes. El modelo general toma la forma:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Regresión lineal simple

Para utilizar un modelo de regresión lineal simple, se requiere un conjunto X de n observaciones, donde cada dato está compuesto por una variable independiente $x^{(i)}$ y una variable dependiente $y^{(i)}$, con $i = 1, \ldots, n$. El objetivo es encontrar una función lineal que relacione ambas variables, de modo que los valores de x permitan estimar los correspondientes valores de y.

Ejemplo 1.1.1

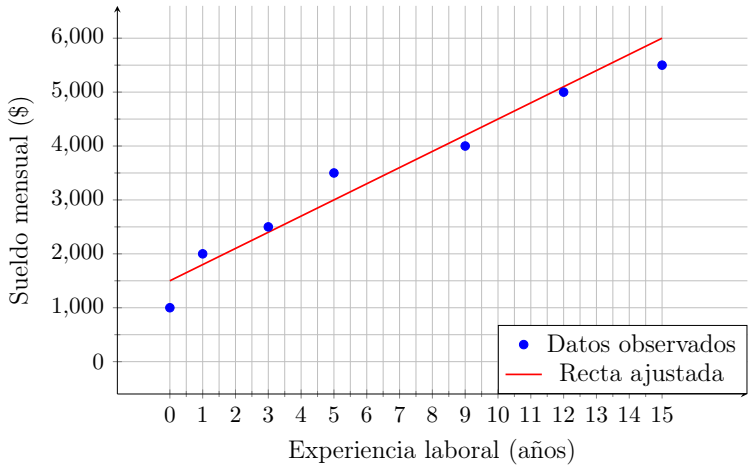
Se tienen los sueldos de 7 empleados junto con los años de experiencia laboral de cada uno. Con estos datos, se desea encontrar un modelo que represente la dependencia del sueldo respecto a la experiencia laboral. La tabla de observaciones es la siguiente:

Sueldo mensual ($)	1000	2000	2500	3500	4000	5000	5500
Experiencia laboral (años)	0	1	3	5	9	12	15

En este caso, $n = 7$, la variable independiente $x^{(i)}$ corresponde a la experiencia laboral en años, y la variable dependiente $y^{(i)}$ al sueldo mensual en pesos.

Por ejemplo, el cuarto empleado tiene $x^{(4)} = 5$ años de experiencia y un sueldo de $y^{(4)} = 3500$ pesos mensuales.

Gráficamente, los datos se verían de la siguiente manera en el plano, junto con un prospecto de recta que modela su comportamiento.



Deducción de una fórmula

Para encontrar la recta que mejor se ajusta a los datos, supondremos que existe una función lineal $h : X \to \mathbb{R}$ que la modela. A esta función se le conoce como la **hipótesis** del modelo.

Definición 1.1.1: Hipótesis

La hipótesis es una función lineal que toma como entrada un valor $x^{(i)}$ y devuelve una predicción para la variable dependiente $y^{(i)}$, dada por:

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x^{(i)}$$

donde el vector de parámetros es $\theta = \langle \theta_0, \theta_1 \rangle$.

Al comparar la salida de la hipótesis con el valor real observado $y^{(i)}$, encontramos una discrepancia. Esta diferencia entre la predicción y el valor real se expresa como:

$$h_\theta(x^{(i)}) - y^{(i)}.$$

Sin embargo, cuando hablamos de *distancia* o error, nos interesa únicamente la magnitud de esta diferencia, no su signo. Para evitar que errores positivos y negativos se cancelen entre sí, se eleva al cuadrado la diferencia:

$$\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2 \geq 0.$$

Observación 1.1.1

Alternativamente, se podría utilizar la función valor absoluto para medir la magnitud del error. No obstante, el valor absoluto no es derivable en el punto cero, lo que complica la aplicación de técnicas de optimización basadas en cálculo diferencial. En cambio, la función cuadrática es suavemente derivable en todo su dominio, lo que la convierte en una opción más conveniente para los fines del modelo.

Si generamos distintas hipótesis $h_\theta(x^{(i)})$ y medimos el error asociado a cada una, nuestro objetivo será encontrar aquella que minimice el error total. No obstante, cada medición individual de error corresponde a un único punto $\langle x^{(i)}, y^{(i)} \rangle$, y lo que realmente buscamos es una hipótesis que se ajuste bien a todos los puntos del conjunto de datos, es decir, para todo $i = 1, \ldots, n$.

Para lograr esto, sumamos los errores al cuadrado asociados a cada observación. Esta suma representa el error global del modelo. Si además dividimos entre $2n$, obtenemos una expresión conveniente que nos servirá como función objetivo para minimizar. A esta se le conoce como la **función de costo**.

Definición 1.1.2: Función de costo

La función de costo mide el error promedio cuadrático de la hipótesis con respecto a los datos observados, y se define como:

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

La función de costo es una función cuadrática estrictamente convexa respecto a los parámetros θ_0 y θ_1. Esto significa que posee un único mínimo global, el cual se alcanza en el

punto donde su derivada (o gradiente) se anula.

Para encontrar dicho mínimo, es necesario resolver el sistema:

$$\nabla_\theta J(\theta) = 0$$

lo cual equivale a resolver las siguientes ecuaciones:

$$\frac{\partial J}{\partial \theta_0} = 0 \quad \text{y} \quad \frac{\partial J}{\partial \theta_1} = 0$$

Para encontrar el mínimo de la función de costo, calculamos sus derivadas parciales con respecto a los parámetros θ_0 y θ_1.

Observación 1.1.2 **P**

ara simplificar la notación, definimos la media de los datos $a^{(i)}$, con $i = 1, 2, \ldots, n$, como:

$$\bar{a} = \frac{1}{n} \sum_{i=1}^{n} a^{(i)}$$

Derivada parcial con respecto a θ_0

$$\begin{aligned}
\frac{\partial J}{\partial \theta_0} &= \frac{\partial}{\partial \theta_0} \left[\frac{1}{2n} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 \right] \\
&= \frac{1}{2n} \sum_{i=1}^{n} \frac{\partial}{\partial \theta_0} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 \\
&= \frac{1}{2n} \sum_{i=1}^{n} 2 \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot \frac{\partial}{\partial \theta_0} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot \frac{\partial}{\partial \theta_0} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right)
\end{aligned}$$

Que desarrollando se tiene:

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)})-y^{(i)}\right) &= \frac{1}{n}\sum_{i=1}^{n}\left(\theta_0+\theta_1 x^{(i)}-y^{(i)}\right)\\
&= \frac{1}{n}\left(\sum_{i=1}^{n}\theta_0+\sum_{i=1}^{n}\theta_1 x^{(i)}-\sum_{i=1}^{n}y^{(i)}\right)\\
&= \frac{1}{n}\left(n\theta_0+\theta_1\sum_{i=1}^{n}\ x^{(i)}-\sum_{i=1}^{n}y^{(i)}\right)\\
&= \frac{n\theta_0}{n}+\theta_1\left(\frac{1}{n}\sum_{i=1}^{n}x^{(i)}\right)-\frac{1}{n}\sum_{i=1}^{n}y^{(i)}\\
&= \theta_0+\theta_1\bar{x}-\bar{y}
\end{aligned}$$

Fórmula para obtener θ_0

Como intentamos anular la derivada parcial con respecto de θ_0 entonces se tiene que:

$$\theta_0+\theta_1\bar{x}-\bar{y}=0$$

Despejando θ_0:

$$\theta_0=\bar{y}-\theta_1\bar{x} \tag{1.1}$$

Derivada parcial con respecto a θ_1

$$\begin{aligned}
\frac{\partial J}{\partial\theta_1} &= \frac{\partial}{\partial\theta_1}\left[\frac{1}{2n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)})-y^{(i)}\right)^2\right]\\
&= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial}{\partial\theta_1}\left(h_\theta(x^{(i)})-y^{(i)}\right)^2\\
&= \frac{1}{2n}\sum_{i=1}^{n}2\left(h_\theta(x^{(i)})-y^{(i)}\right)\cdot\frac{\partial}{\partial\theta_1}\left(h_\theta(x^{(i)})-y^{(i)}\right)\\
&= \frac{1}{n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)})-y^{(i)}\right)\cdot\frac{\partial}{\partial\theta_1}\left(\theta_0+\theta_1 x^{(i)}-y^{(i)}\right)\\
&= \frac{1}{n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)})-y^{(i)}\right)x^{(i)}
\end{aligned}$$

Si realizamos el mismo procedimiento que con θ_0 entonces se tendrá que:

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\left(h_\theta(x^{(i)})-y^{(i)}\right)x^{(i)} &= \frac{1}{n}\sum_{i=1}^{n}\left(\theta_0+\theta_1x^{(i)}-y^{(i)}\right)x^{(i)} \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\bar{y}-\theta_1\bar{x}+\theta_1x^{(i)}-y^{(i)}\right)x^{(i)} \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(x^{(i)}\bar{y}-\theta_1x^{(i)}\bar{x}+\theta_1x^{(i)}x^{(i)}-x^{(i)}y^{(i)}\right) \\
&= \frac{1}{n}\left(\bar{y}\sum_{i=1}^{n}x^{(i)}-\theta_1\bar{x}\sum_{i=1}^{n}x^{(i)}+\theta_1\sum_{i=1}^{n}x^{(i)}x^{(i)}-\sum_{i=1}^{n}x^{(i)}y^{(i)}\right) \\
&= \bar{y}\bar{x}-\theta_1\bar{x}^2+\theta_1\left(\frac{1}{n}\sum_{i=1}^{n}(x^{(i)})^2\right)-\left(\frac{1}{n}\sum_{i=1}^{n}x^{(i)}y^{(i)}\right) \\
&= \bar{y}\bar{x}-\left(\frac{1}{n}\sum_{i=1}^{n}x^{(i)}y^{(i)}\right)+\theta_1\left(\frac{1}{n}\sum_{i=1}^{n}(x^{(i)})^2-\bar{x}^2\right) \\
&= 0
\end{aligned}$$

Fórmula para obtener θ_1

Despejando para θ_1:

$$\theta_1 = \frac{\frac{1}{n}\sum_{i=1}^{n}x^{(i)}y^{(i)}-\bar{y}\bar{x}}{\frac{1}{n}\sum_{i=1}^{n}(x^{(i)})^2-\bar{x}^2}$$

Con ello, hemos obtenido 2 ecuaciones llamadas **ecuaciones normales** que nos sirven para obtener los parámetros de regresión lineal θ_0 y θ_1.

Regresión Lineal Múltiple

Cuando se trabaja con múltiples variables independientes, la notación escalar utilizada en la regresión lineal simple se vuelve engorrosa e ineficiente. En estos casos, es preferible utilizar una **representación matricial**, que permite manipular todos los datos y operaciones de manera compacta y eficiente.

Definición 1.2.1: Hipótesis matricial

La hipótesis del modelo para la i-ésima observación se expresa como:

$$\hat{y}^{(i)} = h_\theta(x^{(i)}) = \theta_0 + \theta_1 x_1^{(i)} + \cdots + \theta_d x_d^{(i)}$$

donde $\hat{y}^{(i)} \in \mathbb{R}$ es un escalar.
Mientras que el **vector de predicciones** completo se escribe en forma matricial como:

$$\hat{y} = \begin{pmatrix} \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \vdots \\ \hat{y}^{(n)} \end{pmatrix} = X\theta$$

donde:

- n es el número de observaciones (filas del conjunto de datos),
- d es el número de variables independientes (también llamadas características),
- $x^{(i)} \in \mathbb{R}^{1\times(d+1)}$ es el vector fila correspondiente a la i-ésima observación (incluyendo un 1 inicial para el término independiente),
- $X \in \mathbb{R}^{n\times(d+1)}$ es la **matriz de diseño**, donde cada fila representa una observación:

$$X = \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_d^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n)} & x_2^{(n)} & \cdots & x_d^{(n)} \end{pmatrix}$$

- $\theta \in \mathbb{R}^{(d+1)\times 1}$ es el **vector de parámetros**:

$$\theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{pmatrix}$$

De la forma escalar a la forma matricial de la función de costo

En notación escalar, la función de costo se define como:

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

Si definimos el vector de predicciones como $\hat{y} = X\theta$, y el vector de salidas reales como $y \in \mathbb{R}^n$, entonces el **vector de errores** es:

$$\hat{y} - y = X\theta - y$$

Este es un vector columna de longitud n. Al multiplicar este vector por su transpuesta, se obtiene la suma de los errores al cuadrado:

$$(\hat{y} - y)^T(\hat{y} - y) = \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

Por lo tanto, la función de costo en forma matricial se expresa como:

$$J(\theta) = \frac{1}{2n}(\hat{y} - y)^T(\hat{y} - y) = \frac{1}{2n}(X\theta - y)^T(X\theta - y)$$

Minimización de la función de costo

Para encontrar el valor óptimo de θ, derivamos la función de costo con respecto a θ y la igualamos a cero:

$$\nabla_\theta J(\theta) = \frac{1}{n} X^T(X\theta - y) = 0$$

Resolviendo esta ecuación obtenemos las **ecuaciones normales**:

$$\boxed{\theta = (X^T X)^{-1} X^T y} \tag{1.2}$$

Esta fórmula nos proporciona el vector de parámetros θ que minimiza el error cuadrático medio.

Dicha formulación matricial generaliza la regresión lineal simple al caso multivariable. Si $d = 1$, el modelo se reduce al caso univariable, pero ahora podemos manejar múltiples características de manera simultánea, lo cual es esencial para conjuntos de datos del mundo real con alta dimensionalidad.

> **Observación 1.2.1**
>
> Para que esta solución exista, la matriz $X^T X$ debe ser invertible. En la práctica, cuando no lo es, se recurre a técnicas de regularización, como la **regresión Ridge**, para evitar problemas de singularidad.

Descenso de Gradiente

Como se ha discutido anteriormente, nuestro objetivo al entrenar modelos lineales es encontrar los valores del vector de parámetros θ que minimicen la función de costo $J(\theta)$, la cual representa la discrepancia entre las predicciones del modelo y los valores reales observados.

Ya vimos que existen fórmulas analíticas para obtener los valores óptimos de θ, como las ecuaciones normales. Sin embargo, cuando el número de variables independientes (d) o de observaciones (n) es muy grande, estas fórmulas se vuelven poco prácticas o computacionalmente costosas. Además, si algunas variables aportan poca o nula información al modelo, el ajuste manual de los parámetros puede ser ineficiente.

Afortunadamente, existe una técnica de optimización muy popular y efectiva llamada **descenso de gradiente**, que permite aproximar iterativamente los valores óptimos de θ a partir de una suposición inicial. Esta técnica es especialmente útil cuando se trabaja con grandes conjuntos de datos o con modelos complejos.

Idea fundamental

Para entender cómo funciona el descenso de gradiente, debemos comprender el significado del vector gradiente de la función de costo:

$$\nabla_\theta J(\theta)$$

Este vector indica la dirección en la que $J(\theta)$ crece más rápidamente. Como queremos minimizar J, buscamos movernos en la dirección opuesta. Por ello, usamos el gradiente negativo:

$$-\nabla_\theta J(\theta),$$

que apunta hacia la dirección en la que $J(\theta)$ disminuye más rápidamente.

Algoritmo del descenso de gradiente

El procedimiento inicia eligiendo un valor inicial para los parámetros, por ejemplo:

$$\theta^{(0)} = \begin{pmatrix} \theta_0^{(0)} \\ \theta_1^{(0)} \\ \vdots \\ \theta_d^{(0)} \end{pmatrix} = \begin{pmatrix} \text{aleatorio} \\ \text{aleatorio} \\ \vdots \\ \text{aleatorio} \end{pmatrix}$$

Dado que estos valores iniciales difícilmente representarán una buena solución, los vamos a ir ajustando paso a paso en dirección al mínimo. Para ello, definimos una tasa de aprendizaje $\alpha > 0$ (teniendo valores usuales de 0.001 a 0.1), un número pequeño que regula el tamaño de los pasos que damos.

La actualización de los parámetros en cada iteración se define como:

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \nabla_\theta J(\theta^{(t)})$$

Esto significa que, en cada paso, nos movemos en la dirección del gradiente negativo, escalado por α para controlar la velocidad de aprendizaje.

Cada componente del vector se actualiza como:

$$\theta_j^{(t+1)} = \theta_j^{(t)} - \alpha \frac{\partial J(\theta^{(t)})}{\partial \theta_j} \quad \text{para } j = 0, 1, \ldots, d$$

donde el denominador no lleva superíndice t porque la derivada es con respecto a θ_j en el punto $\theta^{(t)}$.

Derivando $J(\theta)$ en el caso de regresión lineal, se obtiene:

Definición 1.3.1: Descenso de gradiente

$$\theta_j^{(t+1)} = \theta_j^{(t)} - \alpha \cdot \frac{1}{n} \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)}\right) x_j^{(i)} \quad \text{para } j = 0, 1, \ldots, d$$

Repitiendo este proceso múltiples veces, obtenemos una secuencia de vectores:

$$\theta^{(0)}, \quad \theta^{(1)}, \quad \theta^{(2)}, \quad \ldots$$

que convergen hacia un mínimo de $J(\theta)$.

Para simplificar y acelerar el cálculo, podemos expresar la actualización de todos los parámetros simultáneamente usando notación matricial y vectorizada:

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \cdot \frac{1}{n} X^T (\hat{y} - y)$$

donde X es la matriz de diseño que incluye todas las muestras y características, $\hat{y}$ el vector de predicciones, y y el vector de valores reales.

Esta forma vectorizada no sólo es más compacta y elegante, sino que también es computacionalmente mucho más eficiente que calcular cada componente por separado mediante un ciclo. Esto se debe a que las operaciones matriciales están altamente optimizadas en librerías numéricas y pueden aprovechar paralelismo y memoria caché, reduciendo el tiempo de cómputo especialmente para conjuntos de datos grandes.

¿Cómo sabemos cuándo detenernos?

Existen varias estrategias para decidir cuándo terminar el descenso de gradiente. Algunas de las más comunes son:

- **Número fijo de iteraciones:** se ejecuta el algoritmo un número predeterminado de pasos.
- **Cambio mínimo:** se detiene cuando el cambio entre dos iteraciones consecutivas de $J(\theta)$ es menor a un umbral ϵ, es decir, cuando:

$$|J(\theta^{(t+1)}) - J(\theta^{(t)})| < \epsilon$$

- **Norma del gradiente:** se detiene cuando la magnitud del gradiente es muy pequeña, es decir, cuando:
$$\|\nabla_\theta J(\theta^{(t)})\| < \epsilon$$

El valor de la tasa de aprendizaje α es crucial. Si es demasiado pequeño, el algoritmo puede tardar mucho en converger. Si es muy grande, puede provocar oscilaciones o incluso que el algoritmo diverja. Elegir α adecuadamente es una tarea importante del diseño del modelo.

Observación 1.3.1

El descenso de gradiente es la base de muchos algoritmos de aprendizaje automático, incluyendo el entrenamiento de redes neuronales, y puede extenderse a versiones más sofisticadas como descenso de gradiente estocástico (SGD), descenso mini-batch, y métodos adaptativos como Adam o RMSprop.

Algoritmo del descenso de gradiente

En este apartado jugaremos un poco con la notación. A algunos matemáticos nos da un microinfarto cuando vemos expresiones como:

$$x = x + 1$$

Desde una perspectiva puramente matemática, esto podría llevarnos a absurdos como concluir que $0 = 1$. Sin embargo, en el ámbito de la computación, esta expresión tiene un significado muy distinto. Por ejemplo, en Python podríamos escribir lo siguiente:

```
x = 5
x = x + 1
```

Siempre que se haya definido previamente un valor para `x`, no estamos cometiendo ningún error. Lo que está ocurriendo en este código es una *actualización* del valor de `x`. En la primera línea, `x` vale 5, y en la segunda pasa a valer $5 + 1 = 6$.

Esta aclaración es fundamental, ya que en matemáticas solemos utilizar notaciones como:

$$\theta_i^{(t)}$$

Mientras que en programación es común adoptar notaciones más prácticas y operativas. Por lo tanto, a lo largo de este documento, expresiones como $x = x + 1$ serán perfectamente válidas y se entenderán como actualizaciones del valor de una variable.

El algoritmo de descenso de gradiente parte de un valor inicial para el vector de parámetros, denotado como $\theta^{(0)}$, y lo actualiza iterativamente para minimizar la función de costo $J(\theta)$.

Para simplificar la notación, trabajamos directamente con un único vector $\theta \in \mathbb{R}^{(d+1)\times 1}$, cuyas componentes son θ_j, con $j = 0, 1, \ldots, d$.

La regla de actualización para cada componente θ_j se expresa como:

$$\theta_j = \theta_j - \alpha \cdot \frac{1}{n} \sum_{i=1}^{n} \left(\hat{y}^{(i)} - y^{(i)}\right) x_j^{(i)} \quad \text{para } j = 0, 1, \ldots, d.$$