Aprendizaje de Máquina

ITAM

Semestre agosto-diciembre 2017

Menú

- Métodos Lineales
  - Regresión
    - Mínimos Cuadrados Estático
    - Mínimos Cuadrados Iterativo (SGD)

Regresión Lineal

- Tipo de Método
  - Supervisado
  - Regresión: valor numérico
- Qué suponemos de los datos?
  - Atributos numéricos
  - Muestras de datos tomadas, preferiblemente, i.i.d (independientes e idénticamente distribuidas)
- Aplicaciones
  - Predicción de tendencias, precios,…

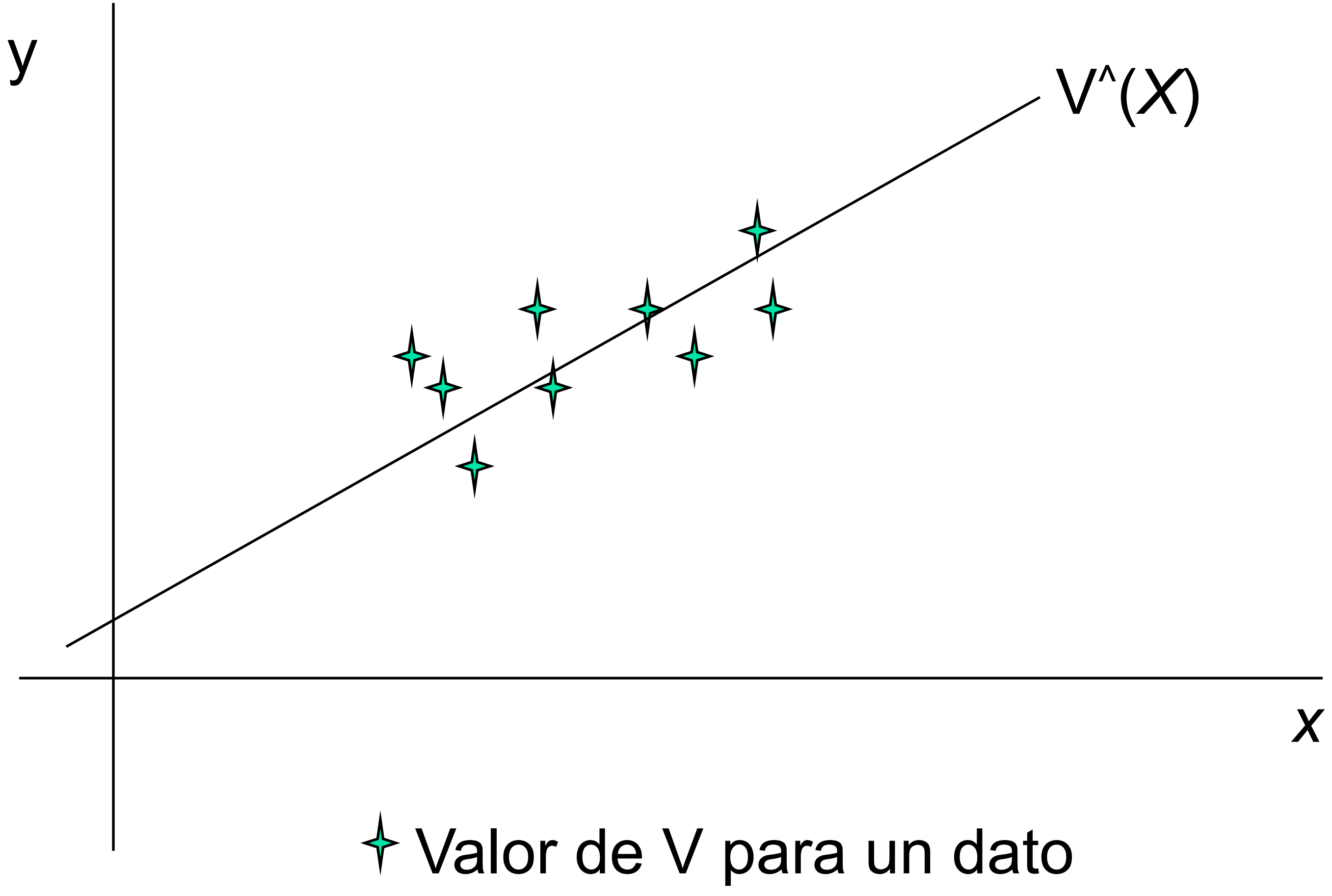
Regresión Lineal

- Como es un método de aprendizaje supervisado, requerimos que cada dato de entrenamiento tenga un valor numérico asociado
  - $(X_1,y_1), (X_2,y_2),\ldots(X_n,y_n)$
  - Donde cada X_i es un registro completo
    - $Xi=<atrib_1,atrib_2,\ldots,atrib_p>$
    - Cada atributo tiene que ser numérico (o haber sido transformado en numérico)
  - Donde las y_i son valores numéricos y representan el valor de entrenamiento de la función objetivo

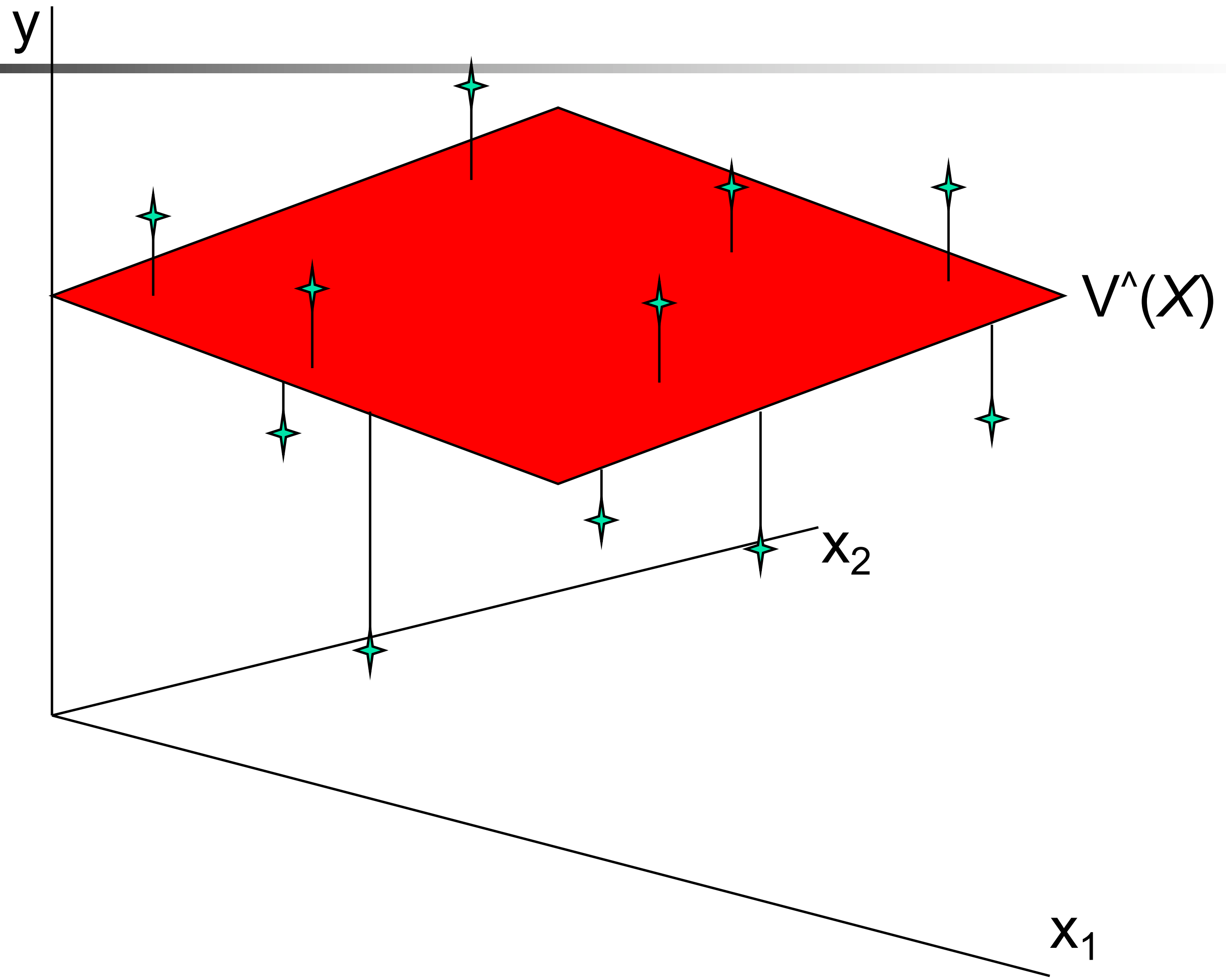
Regresión Lineal

- El objetivo es poder determinar el valor de *y* para datos nuevos, por ejemplo:
  - Datos:
    - <Línea de crédito, saldo> y tenemos la deuda asociada a esos datos (<Línea de crédito, saldo>,deuda)
  - Generar un modelo para:
    - “predecir” la deuda de un individuo del cual sólo conocemos su linea de credito y saldo
- Estos métodos asumen que la función que se intenta estimar V es adecuadamente aproximada por una recta, un plano o hiper-plano
  - V es una función desconocida (que suponemos existe) que determina la relación entre las X y las *y*

Aproximación Lineal de V



Aproximación Lineal de V

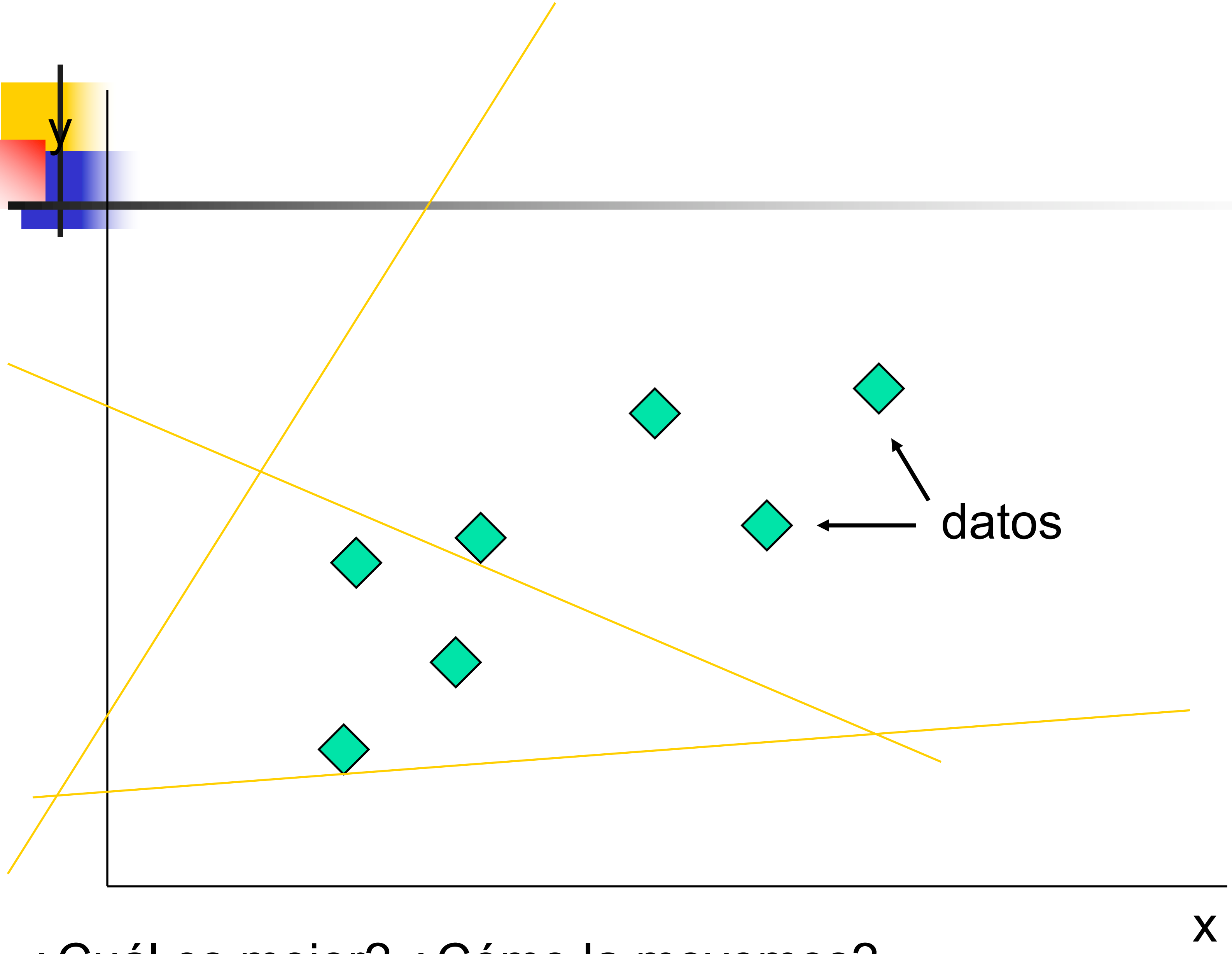


Regresión Lineal

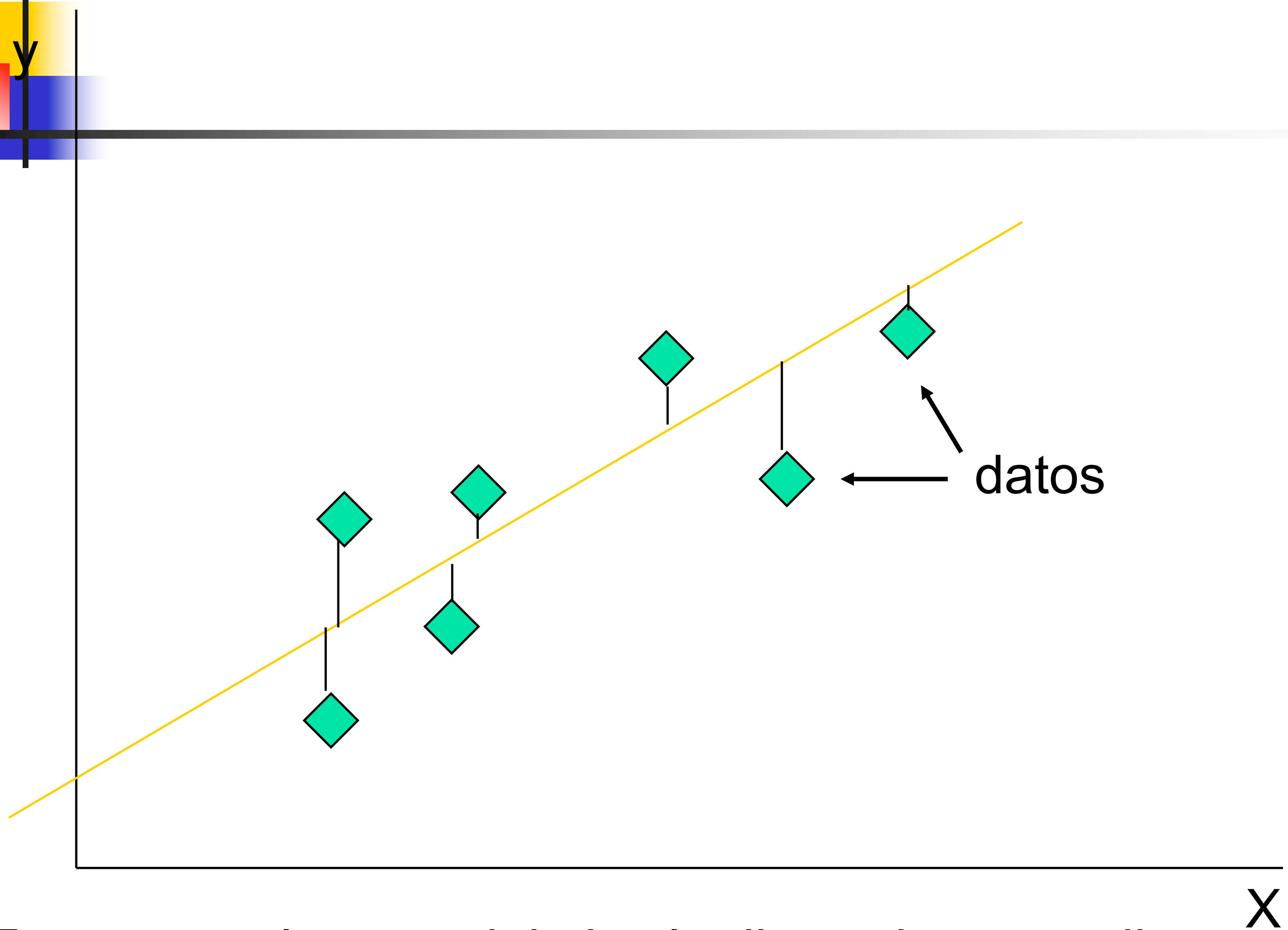
- El modelo de regresión lineal tiene la siguiente forma:
  - $V\hat{}(X)=w_o+\sum x_j w_j$
  - V^ es la aproximación lineal a V, la función que verdaderamente describe los datos
  - La suma es sobre todos los atributos x_j del dato X
  - Las w_j's son los coeficientes de la función
- Los métodos de regresión lineal buscan encontrar valores para los parámetros w_j
  - Encontrar los valores de las w es aprender

Regresión Lineal

- Para esto necesitamos definir una medida de “ajuste” de nuestro modelo a los datos
  - Para valores dados de las w, qué tan bien se ajusta a los datos de entrenamiento
- Requerimos, pues, una medida de error
  - Una vez definida dicha medida, el algoritmo de aprendizaje la utiliza para ajustar el modelo y minimizar el error



¿Cuál es mejor? ¿Cómo la movemos?



Escogemos la que minimice la distancia promedio

Regresión Lineal
Mínimos Cuadrados

- La medida más común para definir el grado de ajuste se conoce como mínimos cuadrados
  - Tomamos las diferencias al cuadrado de cada valor y_i con lo calculado por V^ para los atributos correspondientes al dato x_i
  - Intentamos minimizar la suma de esta medida para todos los datos de entrenamiento

$$MinCuad(B) = \sum_{i=1}^{N}\left(y_i - W_0 - \sum_{j=1}^{p} x_{i,j} W_j\right)^2$$

  - Donde N es el número de datos y p el número de atributos en cada dato
  - ¿Cómo lo minimizamos?

Regresión Lineal
Mínimos Cuadrados

- Supongamos por un momento que sólo tenemos un atributo x y su correspondiente valor *y* para cada uno de los N datos:
  - (X_1,y_1), (X_2,y_2),…, (X_N,y_N)
  - $(<x_{1,1}>,y_1)$, $(<x_{2,1}>,y_2)$,…, $(<x_{N,1}>,y_N)$
- El modelo es una recta
  - $V\hat{}(X_i)=w_o+x_{i,1}w_1$
  - w_o es la ordenada al origen y w_1 es la pendiente

Regresión Lineal
Mínimos Cuadrados

- Y el error:
  - $Mincuad(W)=\sum^N(y_i-V\hat{}(X_i))^2$
  - $Mincuad(W)=\sum^N(y_i-w_o-x_{i,1}w_1)^2$
- Para minimizarlo, primero derivamos
  - $d/w_o=-\sum^N 2(y_i-w_o-x_{i,1}w_1)$
  - $d/w_1=-\sum^N 2(y_i-w_o-x_{i,1}w_1)(x_{i,1})$
- Segundo, igualamos la primer derivada a cero
  - $-\sum^N 2(y_i-w_o-x_{i,1}w_1)=0$
  - $-\sum^N 2(y_i-w_o-x_{i,1}w_1)(x_{i,1})=0$

Regresión Lineal
Mínimos Cuadrados

- Resolvemos para w_o y w_1
  - $w_o N \quad + w_1 \sum^N x_{i,1} \quad = \sum^N y_i$
  - $w_o \sum^N x_{i,1} \quad + w_1 \sum^N x_{i,1}^2 = \sum^N x_{i,1} \, y_i$

Regresión Lineal
Mínimos Cuadrados

- Generalización a p atributos
  - $MinCuad(\mathbf{W})=(\mathbf{y}-\mathbf{XW})^T(\mathbf{y}-\mathbf{XW})$
  - Donde **X** es una matriz con N renglones y p+1 columnas. La primer columna tiene 1's y se utilizan para multiplicar a W_0
  - Donde **W** es el vector de p+1 w_i's
  - Donde **y** es el vector de valores para la función objetivo de las **X**

Regresión Lineal
Mínimos Cuadrados

- La derivada es:
  - $d/d\mathbf{W}=-2\mathbf{X}^T(\mathbf{y}-\mathbf{X}\mathbf{W})$
- Igualando a cero y resolviendo para **W**
  - $\mathbf{W}=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$
- Una vez que se tienen los valores para las w_i' s, encontrar el valor de *V^* para un nuevo dato se lo logra sustituyendo los valores de las w_i' s y los valores de las X del dato en:
  - $V\hat{}(X_i)=w_o+\sum x_{i,j}w_{i,j}$

Ejemplo

- Haga una regresión lineal utilizando los datos provistos por el profesor (reglin)
  - Utilize sklearn (LinearRegression) y el ipython Notebook
  - Sigua el procedimiento para generar modelos delineado por el profesor en clase
  - Grafique los datos y el resultado del modelo
  - Grafique como cambia el error con el valor de los pesos W
    - Para diferentes valores de W calcule el error del modelo y grafique. Dónde quedan las W encontradas por sklearn
- Repita para los archivos reglin2 y 3
  - Anote sus comentarios
  - Qué podemos hacer?

Regresión lineal
Transformaciones de los atributos

- Una regresión lineal encuentra una combinación lineal de los atributos (variables independientes)
- Sin embargo, podemos aplicar transformaciones no-lineales a los atributos para obtener una relación no lineal entre la variable dependiente y las variables independientes originales
  - $y=w_0+\sum f(x_i)$, donde f es la transformación

Regresión lineal
Transformaciones de los atributos

- Por ejemplo: dados mis parejas de datos (x,y) puedo hacer una regresión lineal sobre x^2

  $y=w_o+ \sum x^2$, $f(x)=x^2$
- Qué pasa si hago esto con regLin2?
  - Pruébelo
- Qué tal regLin3? Qué transformación aplicaría?
- Por supuesto la gran pregunta es como escoger la transformación

Regresión Lineal
Método Iterativo (a.k.a Stochastic Gradient Descent)

- Supongamos que los datos no se encuentran todos disponibles al mismo tiempo
  - Que nuevos datos se hacen disponibles en cualquier momento.
- Lo que deseamos es un método incremental que sea capaz de integrar la información contenida en una nueva observación, sin necesidad de reentrenar con todos los datos
  - En ocasiones no solamente queremos incorporar información nueva, sino que también deseamos que información vieja pierda influencia

Regresión Lineal
Método Iterativo(SGD)

- O bien, suponga que no es viable ejecutar la inversión de la matriz
  $\mathbf{W}=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$
- Ya sea porque no esta bien condicionada o porque es demasiado grande….porque hay demasiados datos

Regresión Lineal
Método Iterativo

- Este es el algoritmo que va a aproximar incrementalmente la función objetivo V, i.e., aprende V^
- Requiere de ejemplos con un valor asociado para entrenar. Cada ejemplo es una tupla:
  - (X, y)
  - Donde:
    - X es un datos con p atributos
    - y es el valor que se le asigna a un dato durante el entrenamiento

Regresión Lineal
Método Iterativo

- Nuestra función es la siguiente combinación lineal
  - $V\hat{}(X_i) = w_0 + \Sigma w_{i,j} x_{i,j}$
  - $V\hat{}(X_i) = w_0 x_{i,0} + w_1 x_{i,1} + w_2 x_{i,2} + w_3 x_{i,3} + w_4 x_{i,4} + w_5 x_{i,5}$
    - Truco, insertar en todos los datos el atributo $X_0=1$
- Donde
  - las w's son coeficientes numéricos que determinan la importancia relativa de cada término
  - Las $x_{i,j}$ son los atributos del dato X_i

Regresión Lineal
Método Iterativo

- Al igual que en el método estático necesitamos ajustar los pesos w_i de nuestra función
  - $V\hat{}(X_i) = w_0 x_{i,0} + w_1 x_{i,1} + w_2 x_{i,2} + w_3 x_{i,3} + w_4 x_{i,4} + w_5 x_{i,5}$
  - Ajustar los pesos correctamente es, en este caso, aprender
- Nuestro objetivo es encontrar el valor de los pesos w_i que hagan que nuestra función se ajuste a los ejemplos de entrenamiento

Método Iterativo
Mínimos Cuadrados

- Vamos a buscar los valores para las w_i' s que minimicen el error cuadrático
  - $(y-V^{\wedge}(X))^2$
  - Es la diferencia entre el valor de entrenamiento y lo que estima nuestra función (al cuadrado)
- El algoritmo que vamos se conoce como la regla LMS "least mean squares" (El menor error cuadrático medio) y es también conocida como la regla Widrow-Hoff

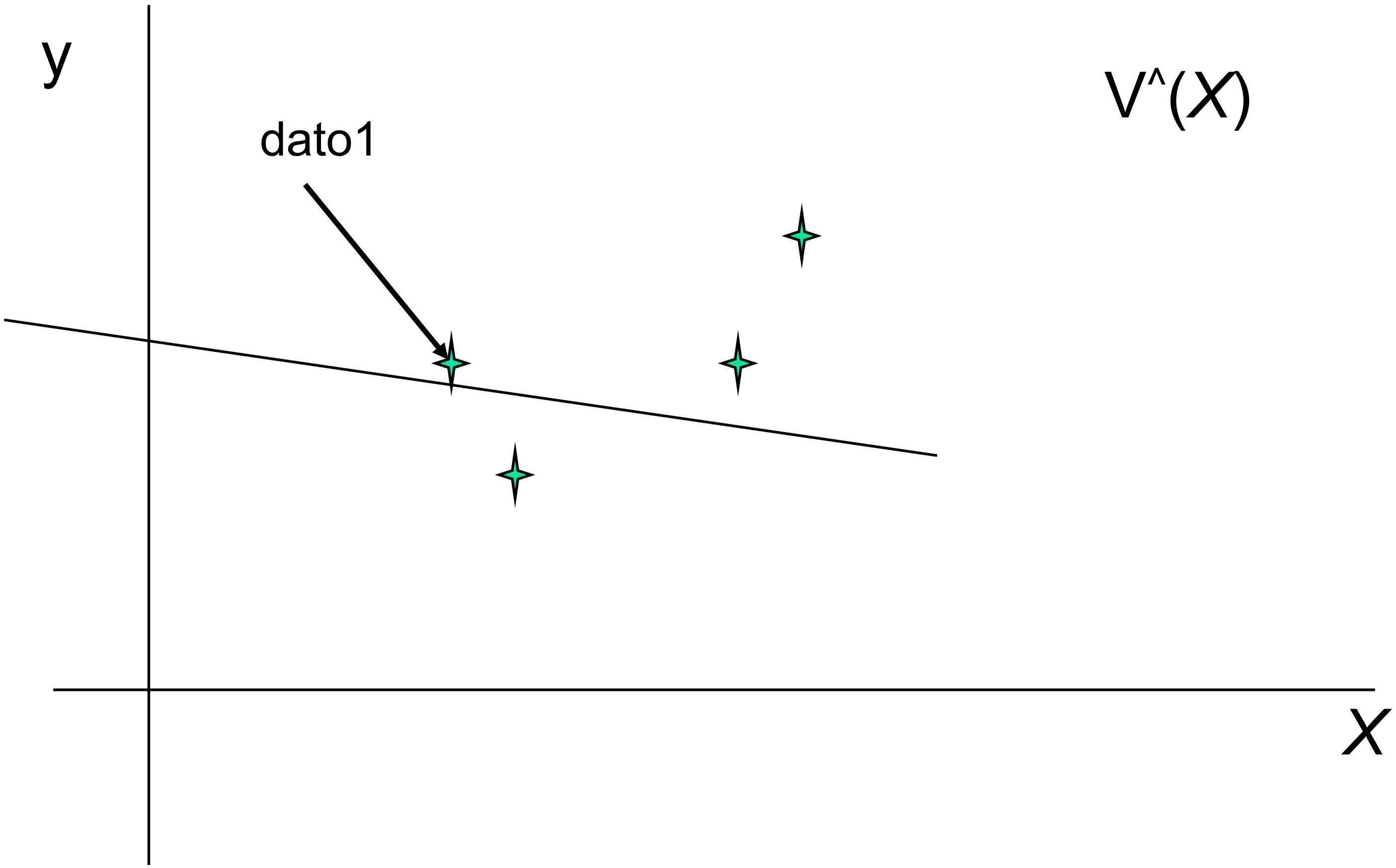
Algoritmo de Aprendizaje LMS

- Para cada ejemplo de entrenamiento (*X*, y)
  - Calcule V^ con las w's actuales
  - Para cada w_i ,
    - w_i <---w_i + η(y-V^(X)) x_i
- Donde η es una constante pequeña menor a 1, e.g. 0.1
- La regla se aplica iterativamente un número fijo de veces ó hasta que se logran los errores deseados ó si no se detecta progreso o en cada ocasión que se presenta un dato

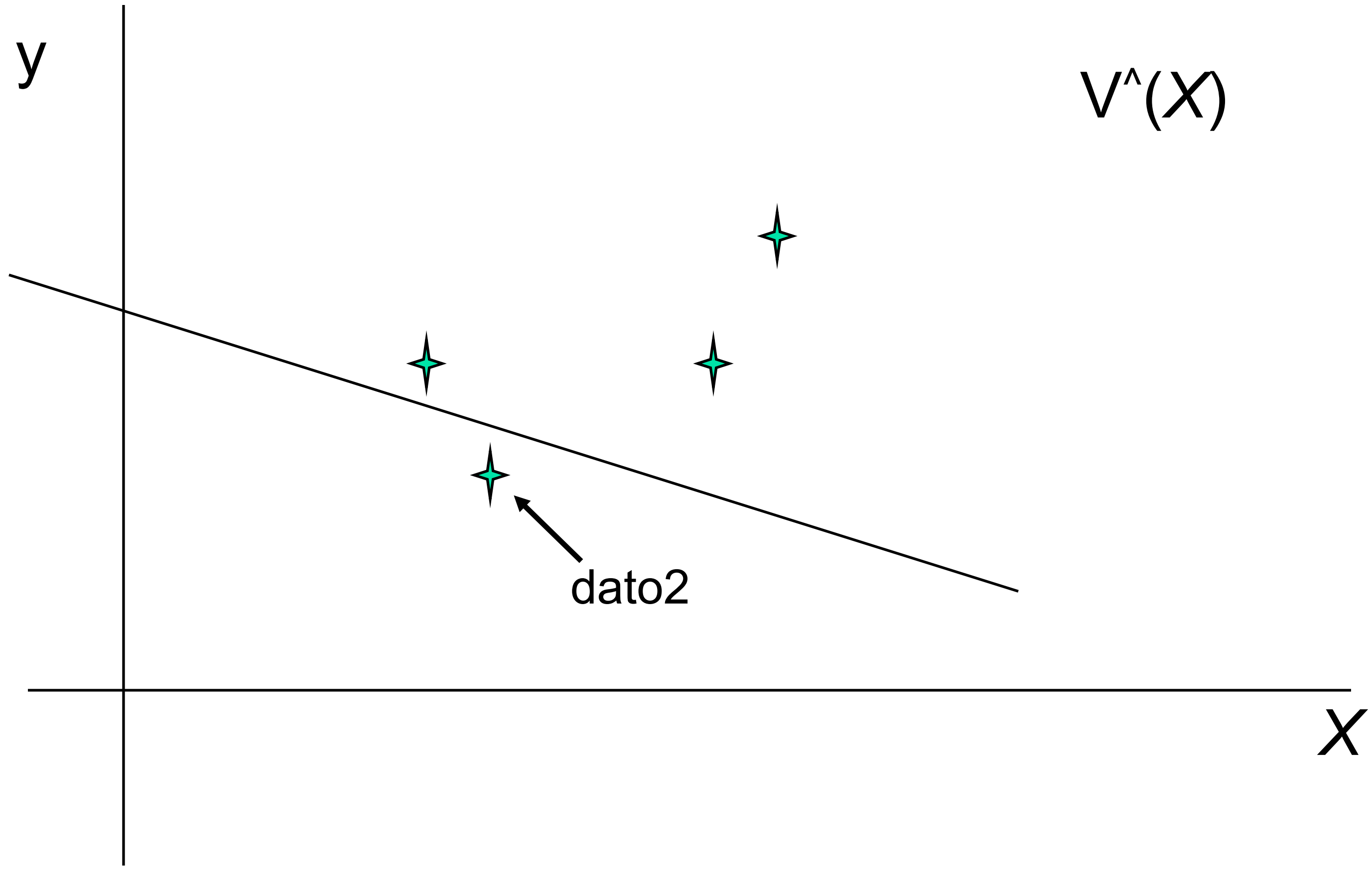
Algoritmo de Aprendizaje LMS

- ¿Porqué sirve? **w_i <---w_i + η(y-V^(X)) x_i**
  - Cuando el error (y-V^(X)) es cero nada cambia
  - Cuando es positivo (V^(X) es muy bajo) cada peso se modifica en proporción al valor de la x_i correspondiente. Esto aumenta V^(X) y reduce el error
  - Análogamente cuando es negativo
  - Nótese que si alguna x_i vale cero el peso correspondiente no cambia. Sólo se ajustan los pesos de las variables que contribuyen en la instancia

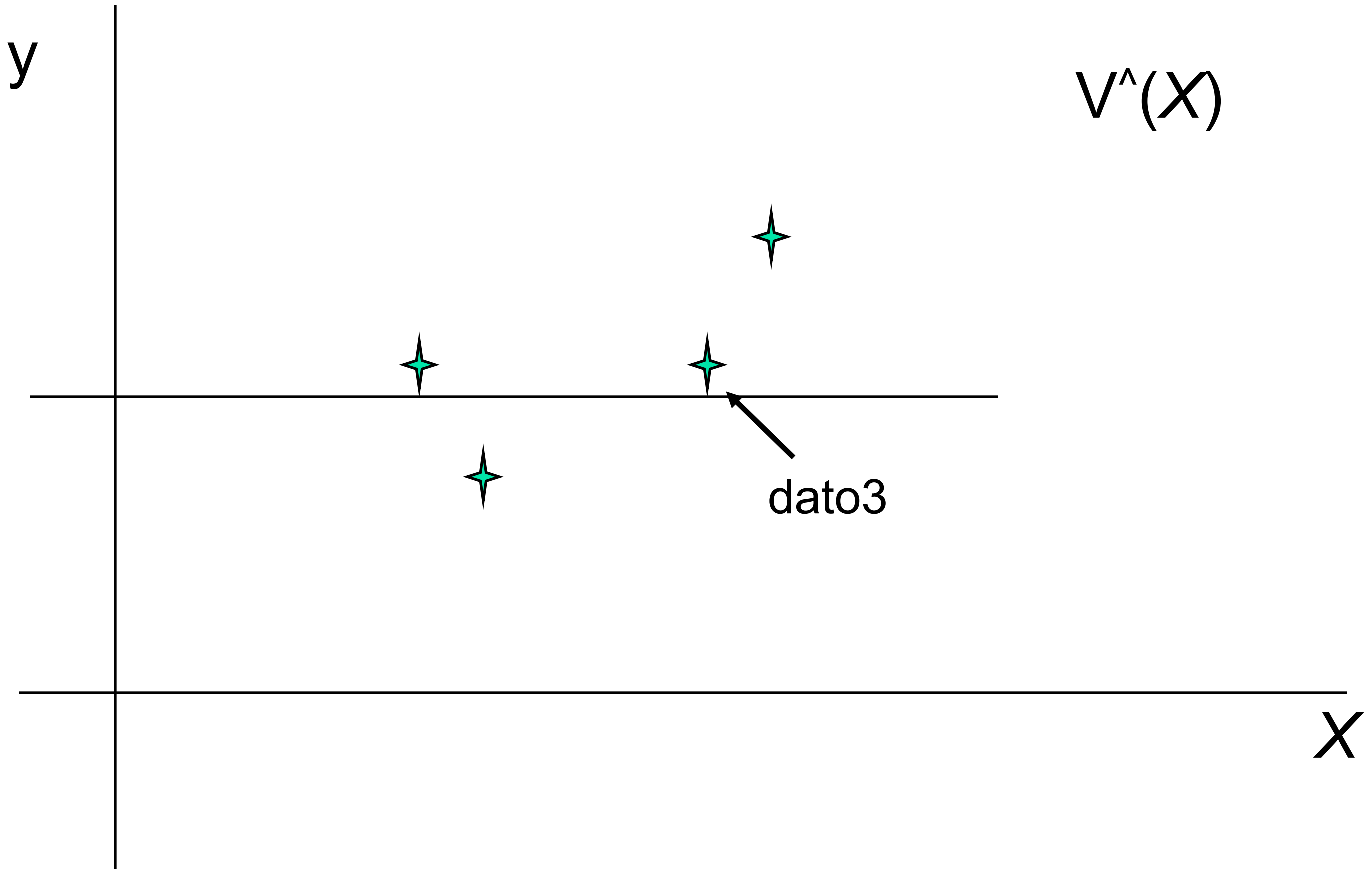
Aproximación Lineal de V



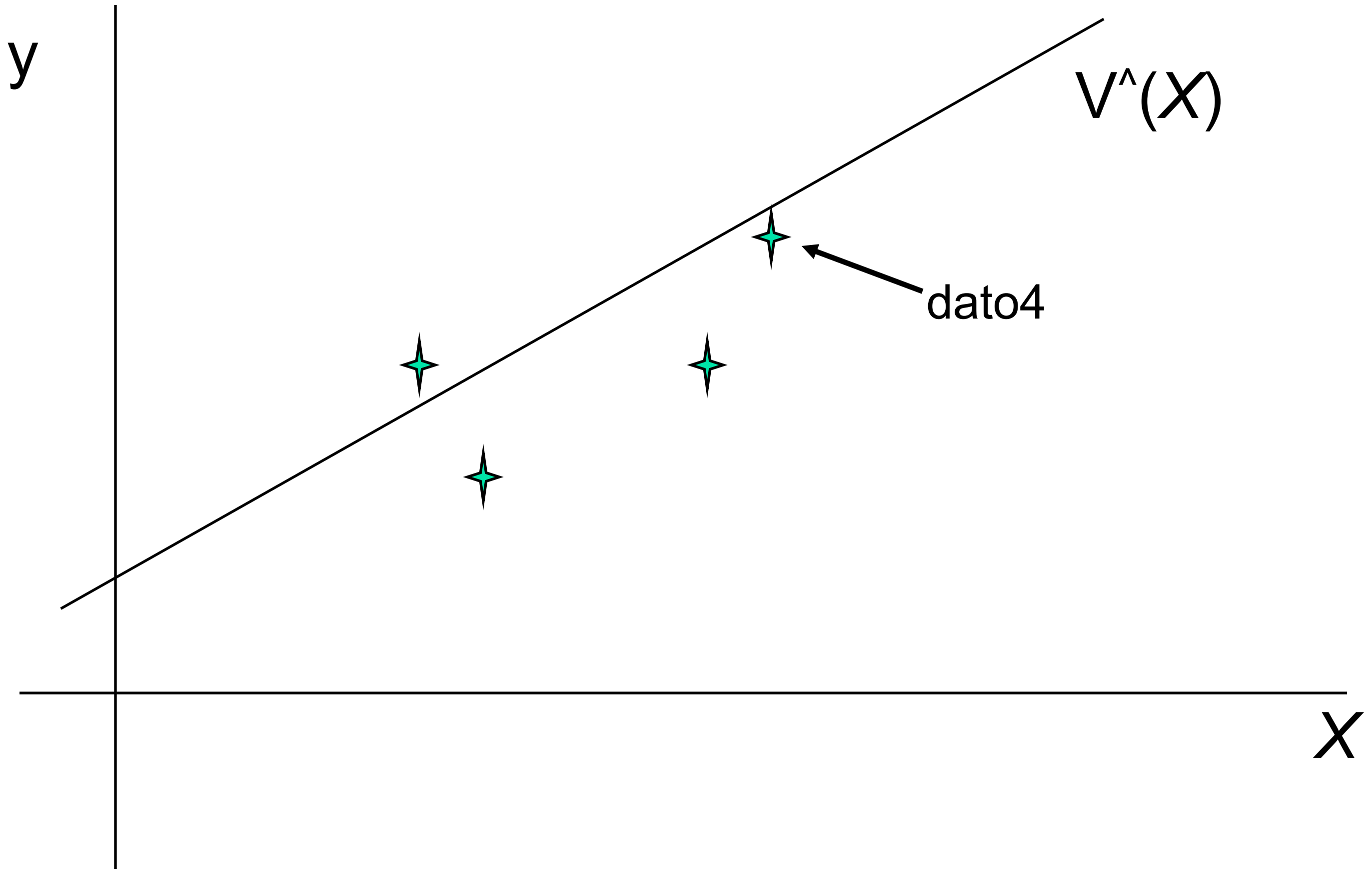
Aproximación Lineal de V



Aproximación Lineal de V



Aproximación Lineal de V



Algoritmo Dinámico Ejemplo

- Datos:
  - (<1,0,1,2,0,0>,1)
  - (<1,1,0,2,0,1>,-1)

Algoritmo Dinámico Ejemplo

	x0	x1	x2	x3	x4	x5
x´s	1	0	1	2	0	0
w´s	1	1	1	1	1	1
$x_i w_i$	1	0	1	2	0	0

y=1, V^(X)=4, Error=1-4=-3, η =0.1

w0 = 1 + 0.1(-3)1= 0.7

w2 = 1 + 0.1(-3)1 = 0.7

w3 = 1 + 0.1(-3)2 = 0.4

Para verificar calculamos el nuevo error para este ejemplo: 1-2.2= -1.2

Algoritmo Dinámico Ejemplo

	x0	x1	x2	x3	x4	x5
x´s	1	1	0	2	0	1
w´s	0.7	1	0.7	0.4	1	1
$x_i w_i$	0.7	1	0	0.8	0	1

y=-1, V^(X)=3.5, Error=-1-3.5=-4.5, η =0.1

w0 = 0.7 + 0.1(-4.5)1=0.25

w1 = 1 + 0.1(-4.5)1 = 0.55

w3 = 0.4 + 0.1(-4.5)2 = -0.5

w5 = 1 + 0.1(-4.5)1 = 0.55

Verificando: el nuevo error con este ejemplo es -1-0.35=-1.35 (¿Cuál es el error con el anterior?)

Ejercicio

- Implementar LMS en Python
- Probar con reglin.csv con η= 0.05
- Graficar como va cambiando el error para cada w
- Pruebe enstandarizar los datos
- Repita el experimento para η= 0.1 y 1
- Escriba sus observaciones

Mini batch

- Hacer el ajuste ejemplo a ejemplo causa que el aprendizaje se vuelva lento
  - Adicionalmente es más difícil identificar variables superfluas (ya que para un ejemplo en particular el valor de la variable superflua puede ayudar a minimizar el error
- Podemos entonces elegir un tamaño de ventana para los datos y particionar los datos en ventanas
  - Calcular el error promedio de esa ventana
  - Ajustar los pesos usando el error calculado
  - Repetir para las demás ventanas
  - Repetir hasta criterio de terminación
- Nota: si se trata de datos estáticos es muy importante aleatorizar el orden de los datos para minimizar el que las ventanas contengan artefactos del orden
  - Porqué?

Notas acerca de regresión lineal

- Estos métodos asumen que la función objetivo puede aproximarse linealmente
  - En muchas ocasiones esta simplificación da muy buenos resultados
- Como siempre, es importante tener una buena muestra de datos y de los correspondientes valores de la función de evaluación para entrenar
- Esta idea es la base de muchos otros metodos y generalizaciones

Notas acerca de regresión lineal

- Vimos que podemos utilizar la regresión lineal para ajustar funciones no lineales
  - Esto se logra haciendo no lineales los regresores, las variables de entrada, y agregando múltiples de estas
- El problema es que no siempre sabemos la forma adecuada de la función ni cuantas variables extras incluir. Si incluimos demasíado no generalizaremos bien
  - Algunos de los temas que veremos en las siguientes clases giran alrededor de solventar esto

Tarea

- Leer el artículo de Searle, objeción a Turing