Aprendizaje Profundo
Facultad de Ingeniería
Universidad de Buenos Aires

Ing. Ezequiel Esposito
(eesposito@optiwe.com)

Análisis matemático para I.A. 1. Espacios vectoriales. 2. Operaciones matriciales. 3. Autovalores y autovectores. 4. Descomposición en valores singulares. 5. Cálculo multivariable. 6. Gradiente. 7. Optimización convexa y no convexa.	**Probabilidad y estadística para I.A.** 1. Espacios de probabilidad. 2. Variables aleatorias. 3. Modelos multivariados. 4. Esperanza condicional. 5. Estimadores puntuales. 6. Estimadores por intervalo. 7. Reglas de decisión. 8. Enfoque Bayesiano.	**Introducción a inteligencia artificial** 1. Teoría de juegos. 2. Búsqueda. 3. Regresión, clasificación y clusterización. 4. Redes Bayesianas. 5. Clasificador Bayesiano. 6. Naive Bayes. 7. Máxima verosimilitud 8. Esperanza-maximización
Visión por computadora I 1. Images. 2. Filtros. 3. Bordes y esquinas. 4. Transformada de Hough y Fourier. 5. Extracción de características. 6. Movimiento. 7. Seguimiento (Kalman y partículas). 8. Clasificación. 9. Detección. 10. Segmentación.	**Aprendizaje de máquina I** 1. Datos. 2. Entrenamiento, validación y testeo. 3. Validación cruzada. 4. Métricas. 5. Evaluación. 6. Regresión y clasificación. 7. Aprendizaje supervisado. 8. Árboles de decisión. 9. kNN. 10. Redes neuronales.	**Aprendizaje profundo** 1. Clasificación binaria. 2. Regresión. 3. Gradiente descendente. 4. Gradiente descendente estocástico. 5. Vectorización. 6. Funciones de activación. 7. Propagación de error. 8. Niveles. 9. Bloques básicos.
Visión por computadora II 1. Redes neuronales convolucionales. 2. Arquitecturas: ResNets, R-CNN, YOLO y UNet. 3. Redes neuronales recurrentes. 4. Descripción de imágenes. 5. Aplicaciones en la industria.	**Aprendizaje de máquina II** 1. Espacios en dimensión reducida. 2. Aprendizaje no supervisado. 3. Clusterización. 4. k-Means. 6. Reducción de dimensión. 7. Análisis de componentes principales.	**Procesamiento de lenguaje natural** 1. Bolsa de palabras. 2. N-grama. 3. TF-IDF. 4. Word2Vec. 5. Vectores de palabras (Glove, FastText). 6. Representación de oraciones. 7. Similaridad entre textos. 8. Seq2Seq. 9. BERT y ElMo.

Clase 1: Introduction to Deep Learning

Clase 2: Likelihood, loss functions, hidden units and output units.

Clase 3: Optimization, regularization and hyperparameter optimization

Clase 4: Feedforward Neural Networks

Clase 5: Convolutional Neural Networks

Clase 6: Recurrent Neural Networks

Clase 7: Representation learning, embeddings and transfer learning.

Clase 8: Exam.

El régimen de aprobación de la materia es simple:

- Trabajos prácticos a implementarse y entregar durante las clases.
- Un examen final online (teórico y práctico) en la clase 8.

Dinámica esperada para las clases:

- 50 minutos de teoría
- 10 minutos de descanso
- 50 minutos de teoría
- 10 minutos de descanso
- 60 minutos de ejercicios

Durante la cursa vamos a utilizar las siguientes herramientas:

- Lenguaje de programación:
  - Python 3.8
  - Herramienta pip para instalar librerias de codigo y dependencias
- Librerias de codigo:
  - Numpy 1.18 y SciPy 1.5
  - PyTorch
- Consola interactiva de Python:
  - iPython y Google Colab
- Herramientas:
  - PyTest para tests, GitHub para repositorios y uWSGI para servidor web
- IDE recomendado:
  - VSCode: https://code.visualstudio.com/

La bibliografía es solo a modo de sugerencia y no será obligatorio el uso de dicho material. El curso está diseñado para ser completamente autocontenido.

- Deep Learning | Ian Goodfellow | https://www.deeplearningbook.org/

Índice

Introduction to Deep Learning

- Loss function
  - Error cuadrático medio
  - Binary Cross Entropy
  - Cross Entropy
  - KL-Divergence
- Model
  - Non-linear neural network
    - Layers: Linear layer | Convolution layer | Recurrent layer
    - Activation Functions: Sigmoid, Softmax, ReLu, Tanh
- Optimization
  - Algorithms
    - Gradient Descent
  - Stochastic Gradient Descent
  - Mini-Batch Gradient Descent

Qué estamos “Aprendiendo”?

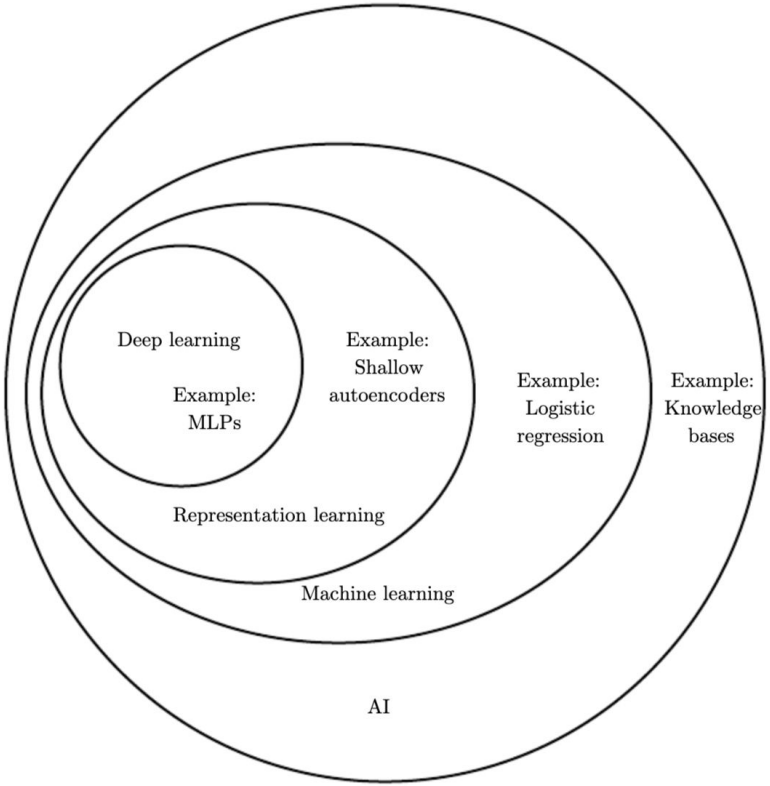
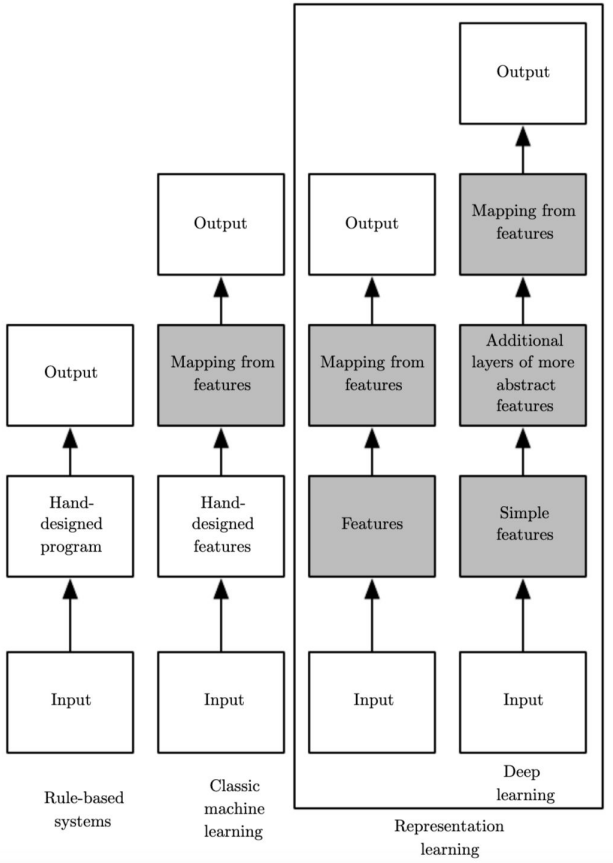
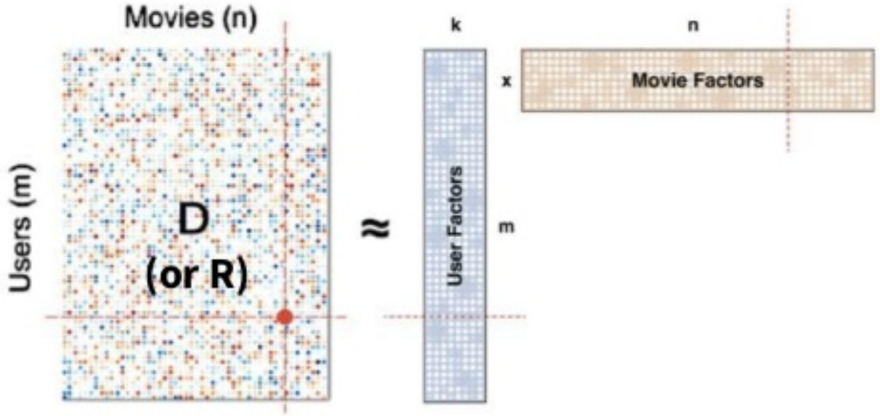


Diagrama de Venn de algoritmos



Bloques que se aprenden en Deep Learning

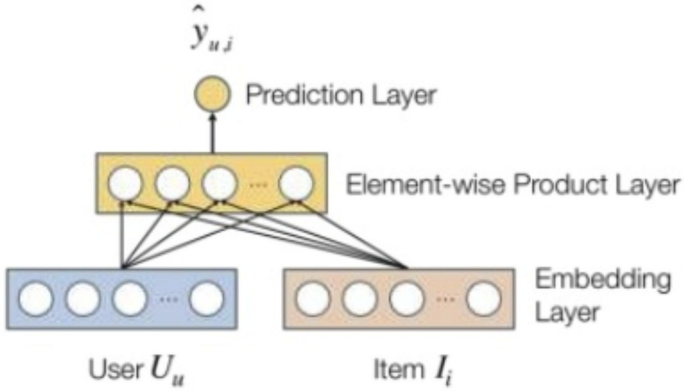
Qué estamos “Aprendiendo”? Ejemplo en Sistemas de Recomendación



$$\nabla \left(\sum_{u,i:R_{u,i}\neq 0} (R_{u,i} - U_u \cdot I_i)^2 \right) = \vec{0}$$

- El objetivo es aprender todos los Uu y Ii que optimizan el error cuadrático medio.
- Uu e Ii son representaciones densas de usuarios e ítems.
- Uu e Ii son conocidos con el nombre de embeddings, representaciones densas.
- Los embeddings tambien los podemos aprender con una red neuronal.

Qué estamos “Aprendiendo”? Ejemplo en Sistemas de Recomendación

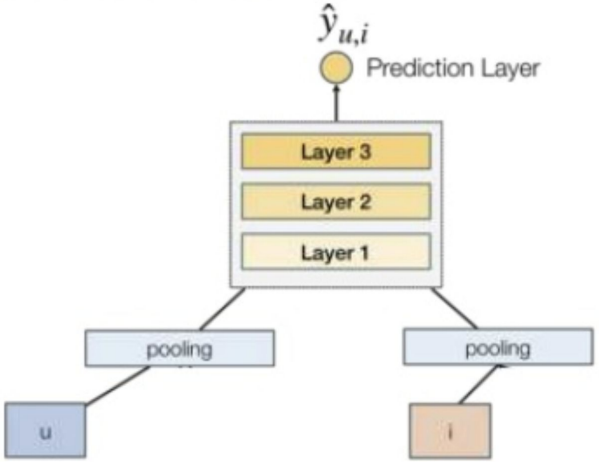


$$\nabla \left(\sum_{u,i:R_{u,i}\neq 0} (R_{u,i} - U_u \cdot I_i)^2 \right) = \vec{0}$$

- Una red neuronal muy simple para resolver el problema es la que vemos en la imagen.
- Lo que aprendemos son los embeddings de los usuarios y los embeddings de los ítems.
- La operación es el producto interno entre Uu e Ii.
- El output es simplemente una neurona mas su funcion de activacion.
- Podríamos usar Binary Cross Entropy como función de costo.

Qué estamos “Aprendiendo”? Ejemplo en Sistemas de Recomendación

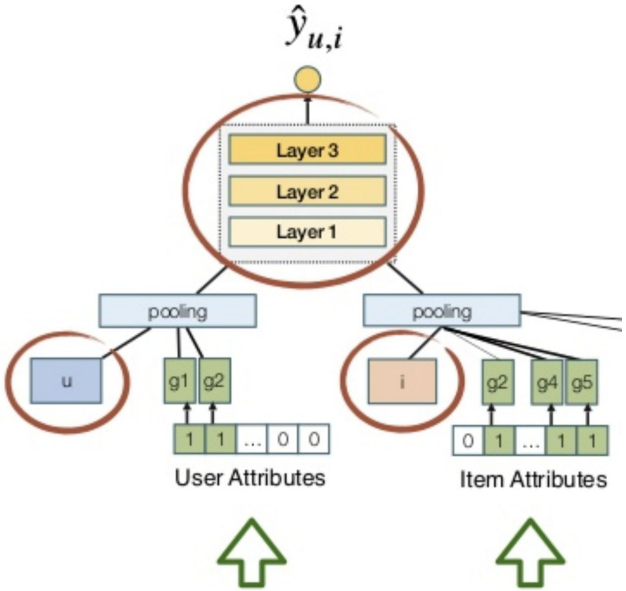
$$\nabla \left(\sum_{u,i:R_{u,i} \neq 0} (R_{u,i} - U_u \cdot I_i)^2 \right) = \vec{0}$$



- Una red neuronal mas compleja es la que vemos en la imagen
- Lo que aprendemos son los embeddings de los usuarios y los embeddings de los ítems.
- La operación es el producto interno entre Uu e Ii.
- El output es simplemente una neurona mas su funcion de activacion.
- Podríamos usar Binary Cross Entropy como función de costo.

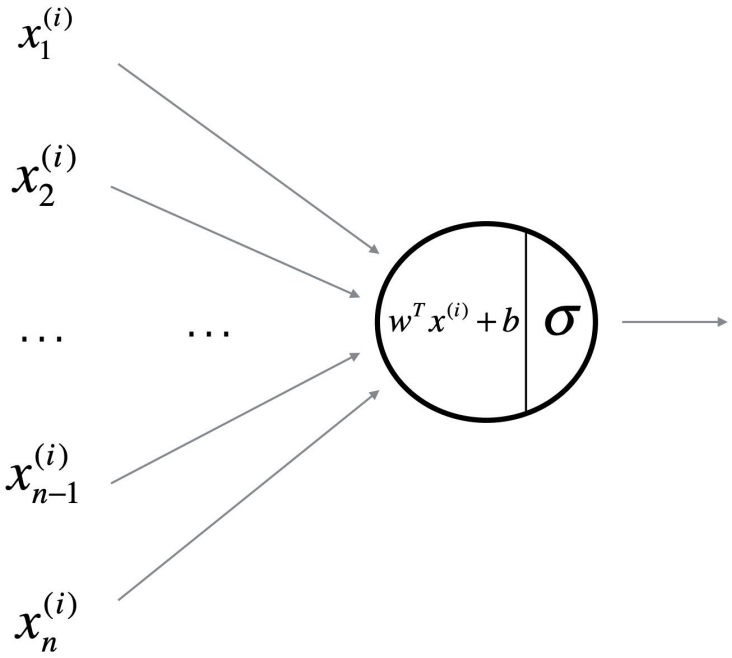
Qué estamos “Aprendiendo”? Ejemplo en Sistemas de Recomendación

$$\nabla \left(\sum_{u,i:R_{u,i} \neq 0} (R_{u,i} - U_u \cdot I_i)^2 \right) = \vec{0}$$

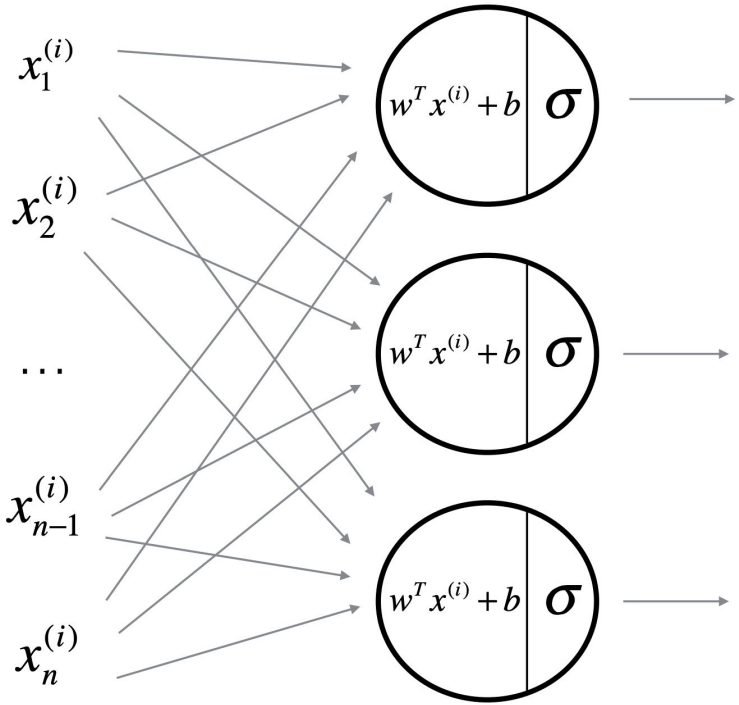


- Círculos rojos -> lo que aprendemos con Gradient Descent
- Flechas verdes -> dataset de entrada (input)

Unidad básica - Neurona



Unidad básica - Layer



Aprendiendo una XOR con un modelo lineal

Aprendiendo una XOR con un modelo no lineal