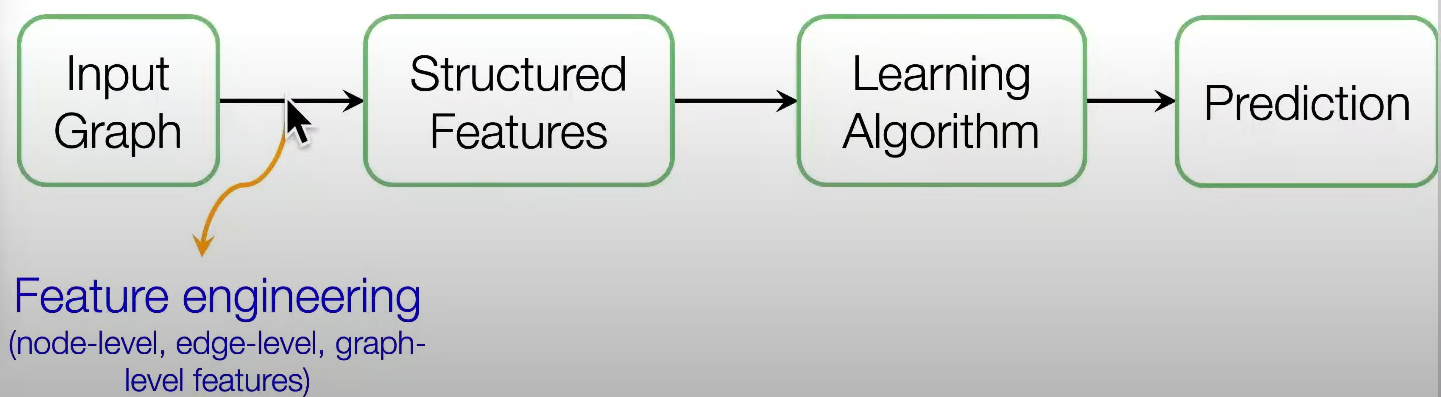
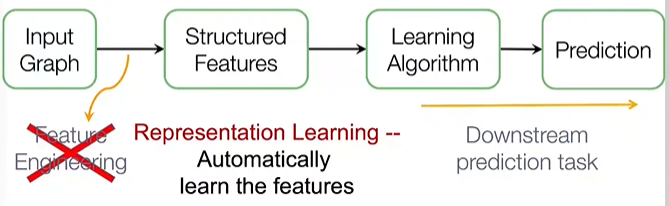
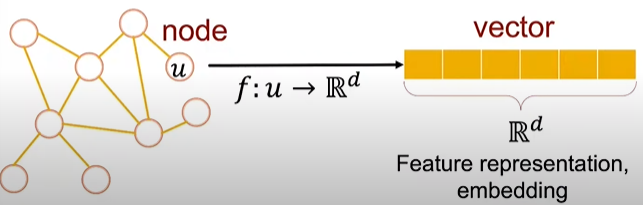
Node Embedding

# Recap --- Traditional ML

Given an input graph, extract node, link and graph-level features, learn a model (SVM, neural network, etc.) that maps features to labels.

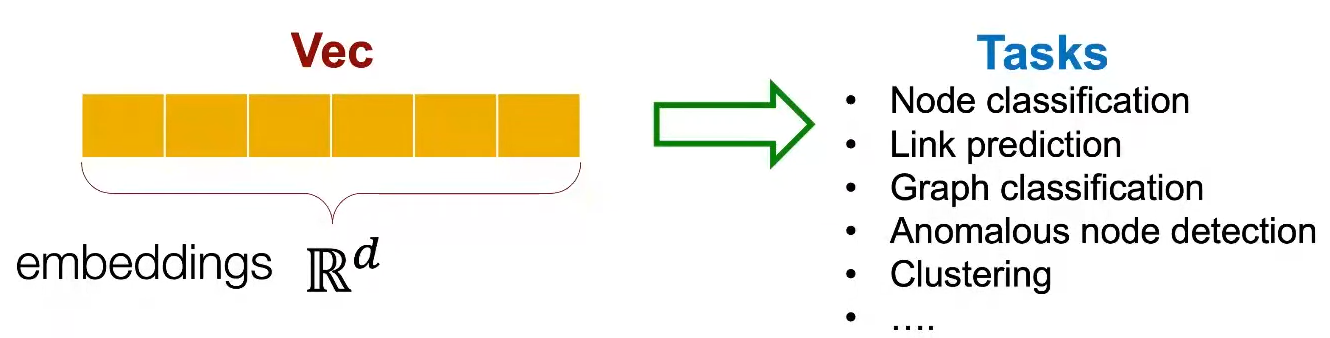
# Graph Representation Learning

****Graph Representation learning alleviates the need to do feature engineering **every single time.**

**Goal:** Efficient task-independent feature learning for machine learning with graphs!

# Why Embedding?

Task: map nodes into an embedding space

* Similarity of embeddings between nodes indicates their similarity in the network.
* Both nodes are close to each other (connected by an edge)
* Encode network information.
* ****Potentially used for many downstream predictions.

**A graph of a diagram

Description automatically generated with medium confidenceExample:**

# Node Embedding --- Encoder and Decoder

A diagram of a diagram of a connection

Description automatically generated**Goal**: is to encode nodes so that similarity in the embedding space (e.g., dot product) approximates similarity in the graph

A black text on a white background

Description automatically generated

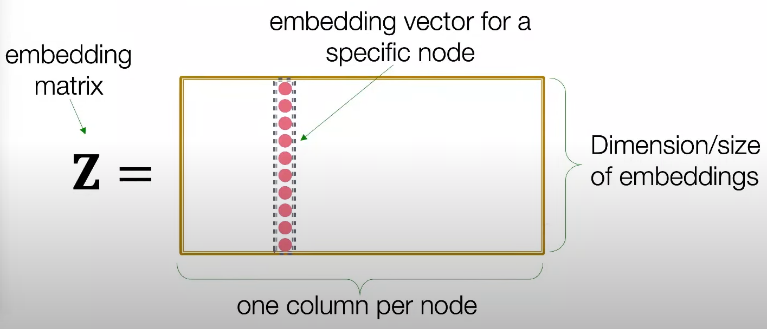
Note: For similarity matrix in an embedding space, people usually use the dot product. Because dot product is the angle between the 2 vectors.

**Learning Node Embeddings:**

1. Encoder maps from nodes to embeddings.
2. Define a node similarity function (i.e., a measure of similarity in the original network).
3. Decode DEC maps from embeddings to the similarity score.
4. Optimize the parameters of the encoder so that:

A black text on a white background

Description automatically generated

Simple encoding approach: encoder is just an embedding-lookup

Each node is assigned a unique embedding vector (i.e., we directly optimize the embedding of each node)

Many methods: DeepWalk, node2vec

**Framework Summary:**

1. **Encoder**

* Shallow encoder --- embedding lookup
* Parameters to optimize: Z which contains node embedding zu for all nodes u from V

1. **Decoder --- based on node similarity**

* **Objective ---** maximize zvTzu for node paris (u, v) that are similar

**How to define node similarity?**

* Key choice of methods is how they define node similarity.
* Should two nodes have a similar embedding if they
* Are linked?
* Share neighbors?
* Have similar “structural roles”?
* We will now learn node similarity definition that uses random walks, how to optimize embeddings for such a similarity measure.

**Note on Node Embeddings:**

Random Walks is an unsupervised way of learning node embeddings.

* Not utilizing node labels
* Not utilizing node features
* The goal is to directly estimate a set of coordinates (i.e., the embedding) of anode so that some aspect of the network structure (capture by DEC) is preserved.

These embeddings are task independent:

* They are not trained for a specific task but can be used for any task.