Word2Vec is a shallow, two-layer neural networks which is trained to reconstruct linguistic contexts of words.   
It takes as its input a large corpus of words and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space.  
Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space.  
Word2Vec is a particularly computationally-efficient predictive model for learning word embeddings from raw text.  
It comes in two flavors, the Continuous Bag-of-Words (CBOW) model and the Skip-Gram model.

**Continuous Bag-of-Words (CBOW)**

CBOW predicts target words (e.g. ‘mat’) from the surrounding context words (‘the cat sits on the’).

#### Skip-Gram

Skip-gram predicts surrounding context words from the target words (inverse of CBOW).  
Statistically, skip-gram treats each context-target pair as a new observation, and this tends to do better when we have larger datasets.

Word2Vec uses a trick you may have seen elsewhere in machine learning.  
Word2Vec is a simple neural network with a single hidden layer, and like all neural networks, it has weights, and during training, its goal is to adjust those weights to reduce a loss function. However, Word2Vec is not going to be used for the task it was trained on, instead, we will just take its hidden weights, use them as our word embeddings, and toss the rest of the model.

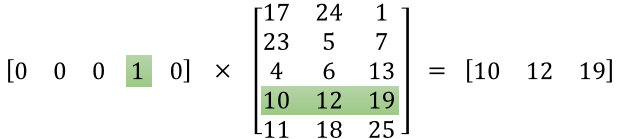
*Another place you may have seen this trick is in unsupervised feature learning, where an auto-encoder is trained to compress an input vector in the hidden layer and decompress it back to the original in the output layer. After it’s done training, the output layer (the decompression step) is stripped off and only the hidden layer is used since it has learned good features, it’s a trick for learning good image features without having labelled training data..*

First of all, we cannot feed a word as string into a neural network.  
Instead, we feed words as one-hot vectors, which is basically a vector of the same length as the vocabulary, filled with zeros except at the index that represents the word we want to represent, which is assigned “1”.

The hidden layer is a standard fully-connected (Dense) layer whose weights are the word embeddings.

The output layer outputs probabilities for the target words from the vocabulary.

The hidden layer operates as a lookup table. The output of the hidden layer is just the “word vector” for the input word.  
More concretely, if you multiply a 1 x 10,000 one-hot vector by a 10,000 x 300 matrix, it will effectively just select the matrix row corresponding to the ‘1’.



The end goal of all of this is to learn this hidden layer weight matrix and then toss the output layer when we’re done! Word2Vec is able to capture multiple different degrees of similarity between words. Patterns such as “Man is to Woman as Brother is to Sister” can be generated and also for a range of semantic relations (such as Country—Capital) as well as syntactic relations (e.g. present tense—past tense).