Machine learning interview

ShuaiW

machine learning

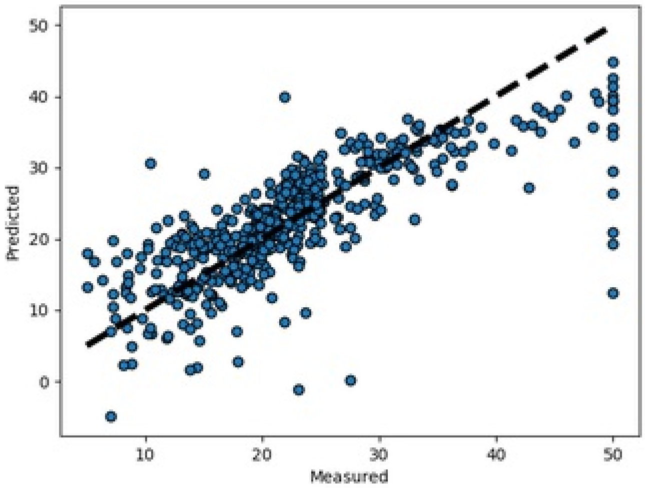
Machine learning

First, it's alw ays a good idea to review Chapter 5
of the deep learning book, w hich covers machine learning basics.

- Linear regression
- Logistic regression
- KNN
- SVM
- [Naive Bayes]
- Decision tree
- Bagging
- Random forest
- Boosting
- Stacking
- [Clustering]
- MLP
- CNN
- RNN and LSTM
- w ord2vec
- Generative vs discriminative
- Paramteric vs Nonparametric

Linear regression

- how to learn the parameter: minimize the cost function
- how to minimize cost function: gradient descent
- regularization:
  - L1 (lasso): can shrink certain coef to zero, thus performing feature selection
  - L2 (ridge): shrink all coef w ith the same proportion; almost alw ays outperforms L1
  - combined (Elastic Net):
- assumes linear relationship betw een features and the label
- can add polynomial and interaction features to add non-linearity



back to top

Logistic regression

- Generalized linear model (GLM) for classification problems
- Apply the sigmoid function to the output of linear models, squeezing the target to range [0, 1]
- Threshold to make prediction: if the output > .5, prediction 1; otherw ise prediction 0
- a special case of softmax function, w hich deals w ith multi-class problem

back to top

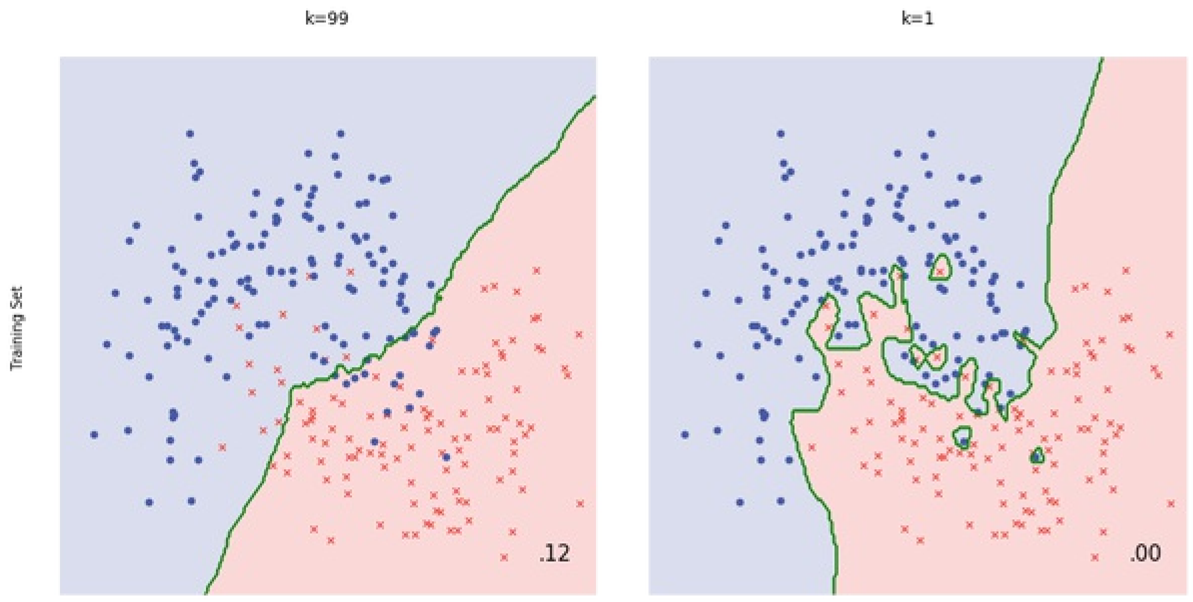
KNN

Given a data point, w e compute the K nearest data points (neighbors) using certain distance metric (e.g., Euclidean metric). For classification, w e take the majority label of neighbors; for regression, w e take the mean of the label values.

Note for KNN technically w e don't need to train a model, w e simply compute during inference time. This can be computationally expensive since each of the test example need to be compared w ith every training example to see how close they are.

There are approximation methods can have faster inference time by partitioning the training data into regions.

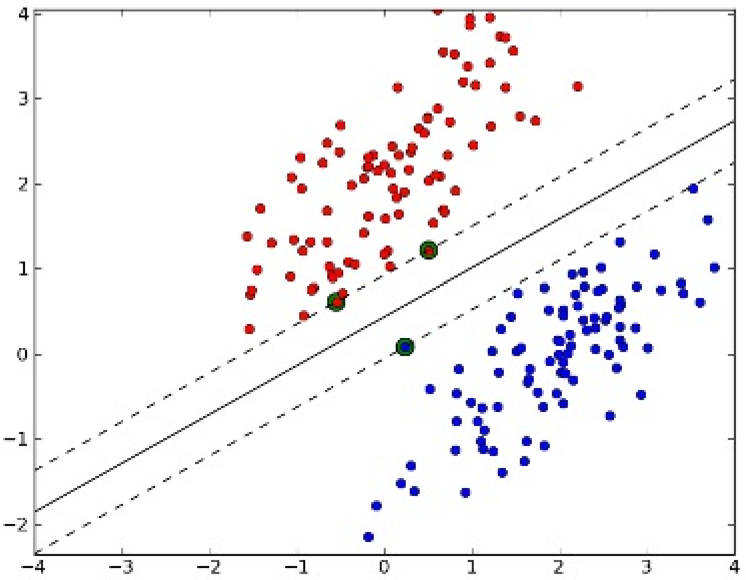
Note w hen K equals 1 or other small number the model is prone to overfitting (high variance), w hile w hen K equals number of data points or other large number the model is prone to underfitting (high bias)



back to top

SVM

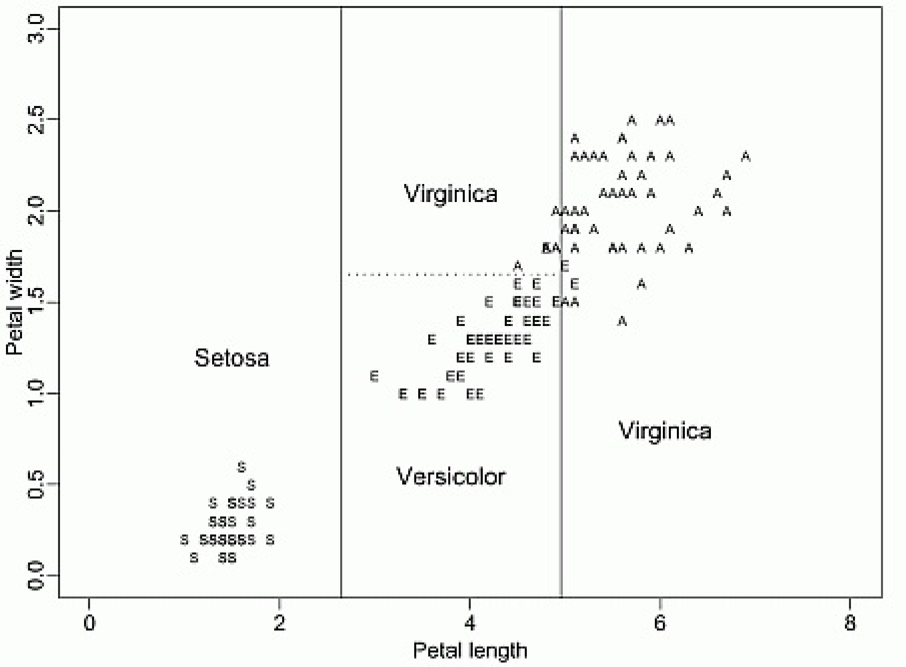
- can perform linear, nonlinear, or outlier detection (unsupervised)
- large margin classifier: not only have a decision boundary, but want the boundary to be as far from the closest training point as possible
- the closest training examples are called the support vectors, since they are the points based on which the decision boundary is drawn
- SVMs are sensitive to feature scaling



back to top

Decision tree

- Non-parametric, supervised learning algorithms
- Given the training data, a decision tree algorithm divides the feature space into regions. For inference, w e first see w hich
  region does the test data point fall in, and take the mean label values (regression) or the majority label value (classification).
- **Construction**: top-dow n, chooses a variable to split the data such that the target variables w ithin each region are as homogeneous as possible. Tw o common metrics: gini impurity or information gain, w on't matter much in practice.
- Advantage: simply to understand & interpret, mirrors human decision making
- Disadvantage:
  - can overfit easily (and generalize poorly)if w e don't limit the depth of the tree
  - can be non-robust: A small change in the training data can lead to a totally different tree
  - instability: sensitive to training set rotation due to its orthogonal decision boundaries

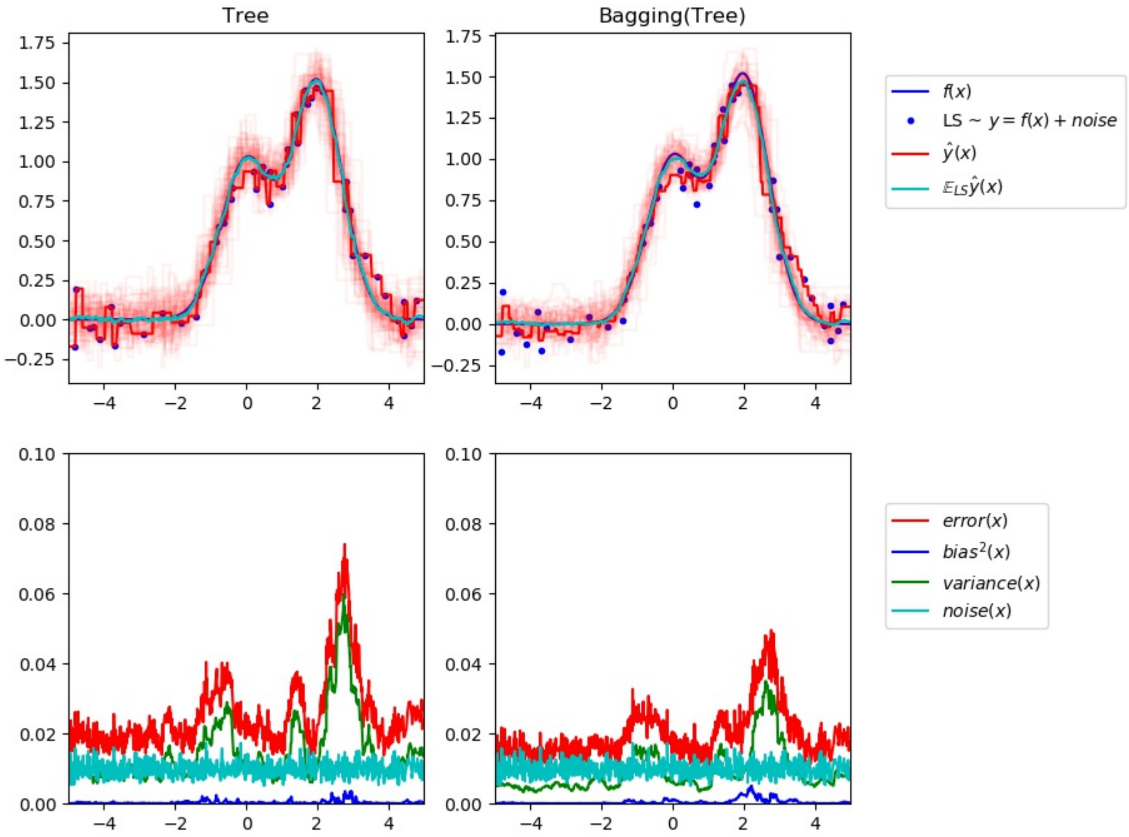


back to top

Bagging

To address overfitting, w e can use an ensemble method called bagging (bootstrap aggregating), w hich reduces the variance of the meta learning algorithm. Bagging can be applied to decision tree or other algorithms.

Here is a great illustration of a single estimator vs. bagging



- Bagging is when samlping is performed *with* replacement. When sampling is performed *without* replacement, it's called pasting.
- Bagging is popular due to its boost for performance, but also due to that individual learners can be trained in parallel and scale well
- Ensemble methods work best when the learners are as independent from one another as possible
- Voting: soft voting (predict probability and average over all individual learners) often works better than hard voting
- out-of-bag instances (37%) can act validation set for bagging

back to top

Random forest

Random forest improves bagging further by adding some randomness. In random forest, only a subset of features are selected at random to construct a tree (while often not subsample instances). The benefit is that random forest **decorrelates** the trees.

For example, suppose we have a dataset. There is one very predicative feature, and a couple of moderately predicative features. In bagging trees, most of the trees will use this very predicative feature in the top split, and therefore making most of the trees look similar, **and highly correlated**. Averaging many highly correlated results won't lead to a large reduction in variance compared with uncorrelated results. In random forest for each split we only consider a subset of the features and therefore reduce the variance even further by introducing more uncorrelated trees.

I wrote a notebook to illustrate this point.

In practice, tuning random forest entails having a large number of trees (the more the better, but always consider computation constraint). Also, `min_samples_leaf` (The minimum number of samples at the leaf node)to control the tree size and overfitting. Always CV the parameters.

Feature importance

In a decision tree, important features are likely to appear closer to the root of the tree. We can get a feature's importance for random forest by computing the averaging depth at which it appears across all trees in the forest.

back to top

Boosting

How it works

Boosting builds on weak learners, and in an iterative fashion. In each iteration, a new learner is added, while all existing learners are kept unchanged. All learners are weighted based on their performance (e.g., accuracy), and after a weak learner is added, the data are re-weighted: examples that are misclassified gain more weights, while examples that are correctly classified lose weights. Thus, future weak learners focus more on examples that previous weak learners misclassified.

Difference from random forest (RF)

- RF grows trees **in parallel**, while Boosting is sequential
- RF reduces variance, while Boosting reduces errors by reducing bias

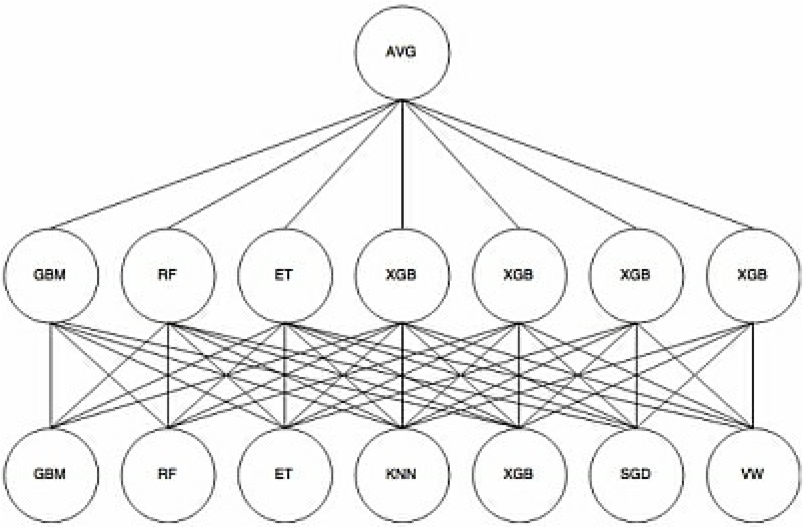
XGBoost (Extreme Gradient Boosting)

> XGBoost uses a more regularized model formalization to control overfitting, which gives it better performance

back to top

Stacking

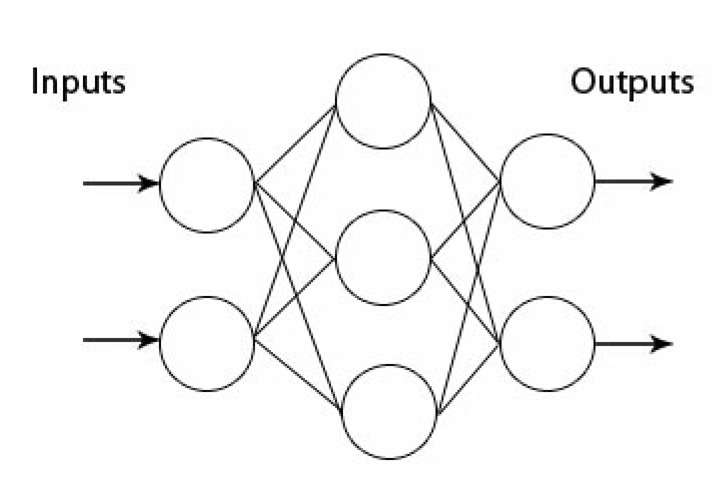
- Instead of using trivial functions (such as hard voting) to aggregate the predictions from individual learners, train a model to perform this aggregation
- First split the training set into two subsets: the first subset is used to train the learners in the first layer
- Next the first layer learners are used to make predictions (meta features) on the second subset, and those predictions are used to train another models (to obtain the weigts of different learners) in the second layer
- We can train multiple models in the second layer, but this entails subsetting the original dataset into 3 parts



back to top

MLP

A feedforw ard neural netw ork w here w e have multiple layers. In each layer w e can have multiple neurons, and each of the neuron in the next layer is a linear/nonlinear combination of the all the neurons in the previous layer. In order to train the netw ork w e back propagate the errors layer by layer. In theory MLP can approximate any functions.

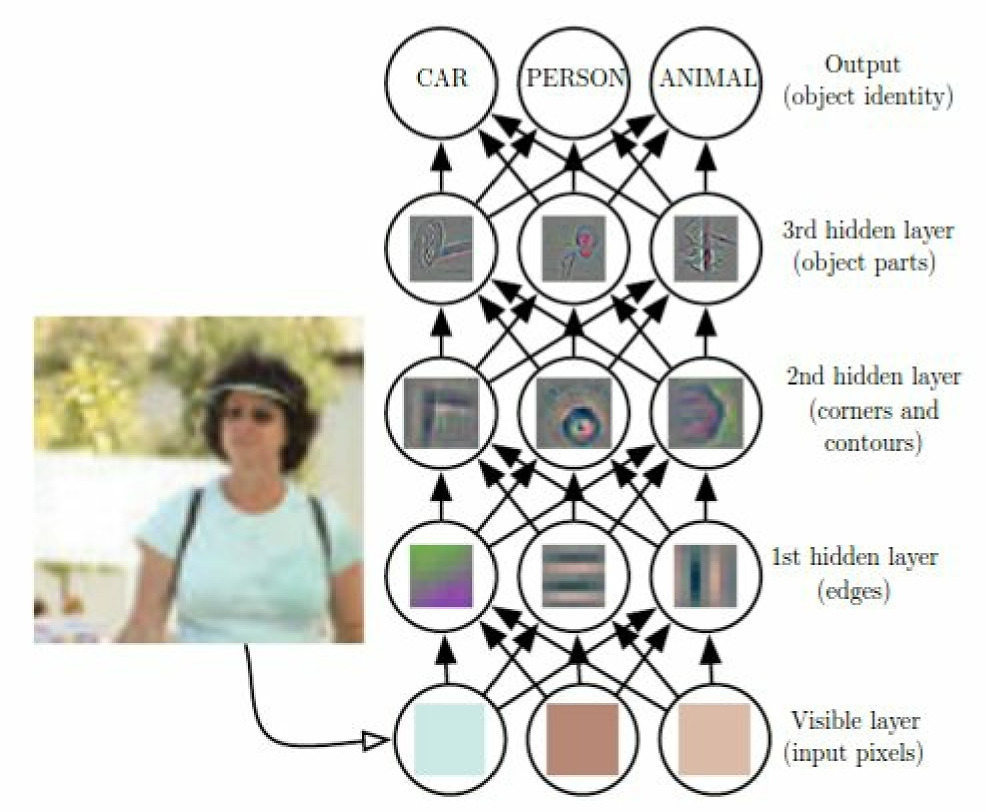


back to top

CNN

The Conv layer is the building block of a Convolutional Netw ork. The Conv layer consists of a set of learnable filters (such as 5 * 5 * 3, w idth * height * depth). During the forw ard pass, w e slide (or more precisely, convolve) the filter across the input and compute the dot product. Learning again happens w hen the netw ork back propagate the error layer by layer.

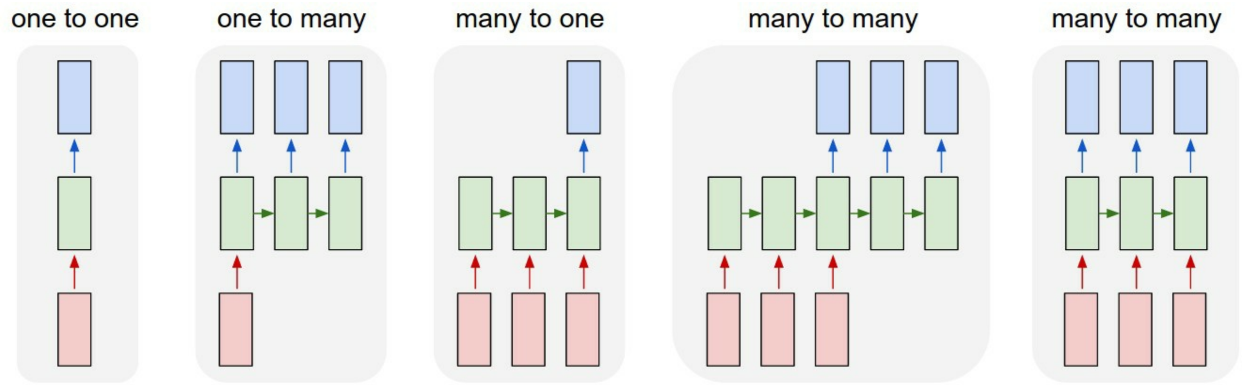
Initial layers capture low -level features such as angle and edges, w hile later layers learn a combination of the low -level features and in the previous layers and can therefore represent higher level feature, such as shape and object parts.



back to top

RNN and LSTM

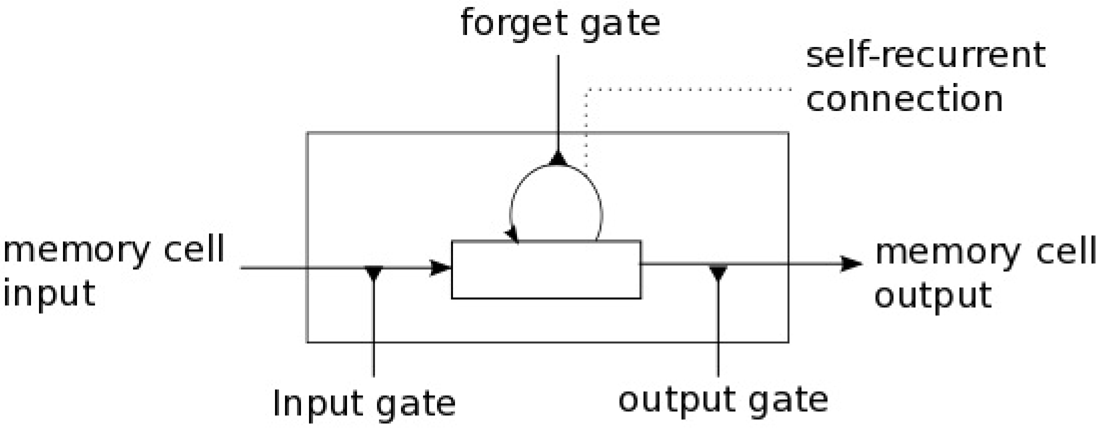
RNN is another paradigm of neural netw ork w here w e have difference layers of cells, and each cell only take as input the cell from the previous layer, but also the previous cell w ithin the same layer. This gives RNN the pow er to model sequence.



This seems great, but in practice RNN barely w orks due to exploding/vanishing gradient, w hich is cause by a series of multiplication of the same matrix. To solve this, w e can use a variation of RNN, called long short-term memory (LSTM), w hich is capable of learning long-term dependencies.

The math behind LSTM can be pretty complicated, but intuitively LSTM introduce
- input gate
- output gate
- forget gate
- memory cell (internal state)

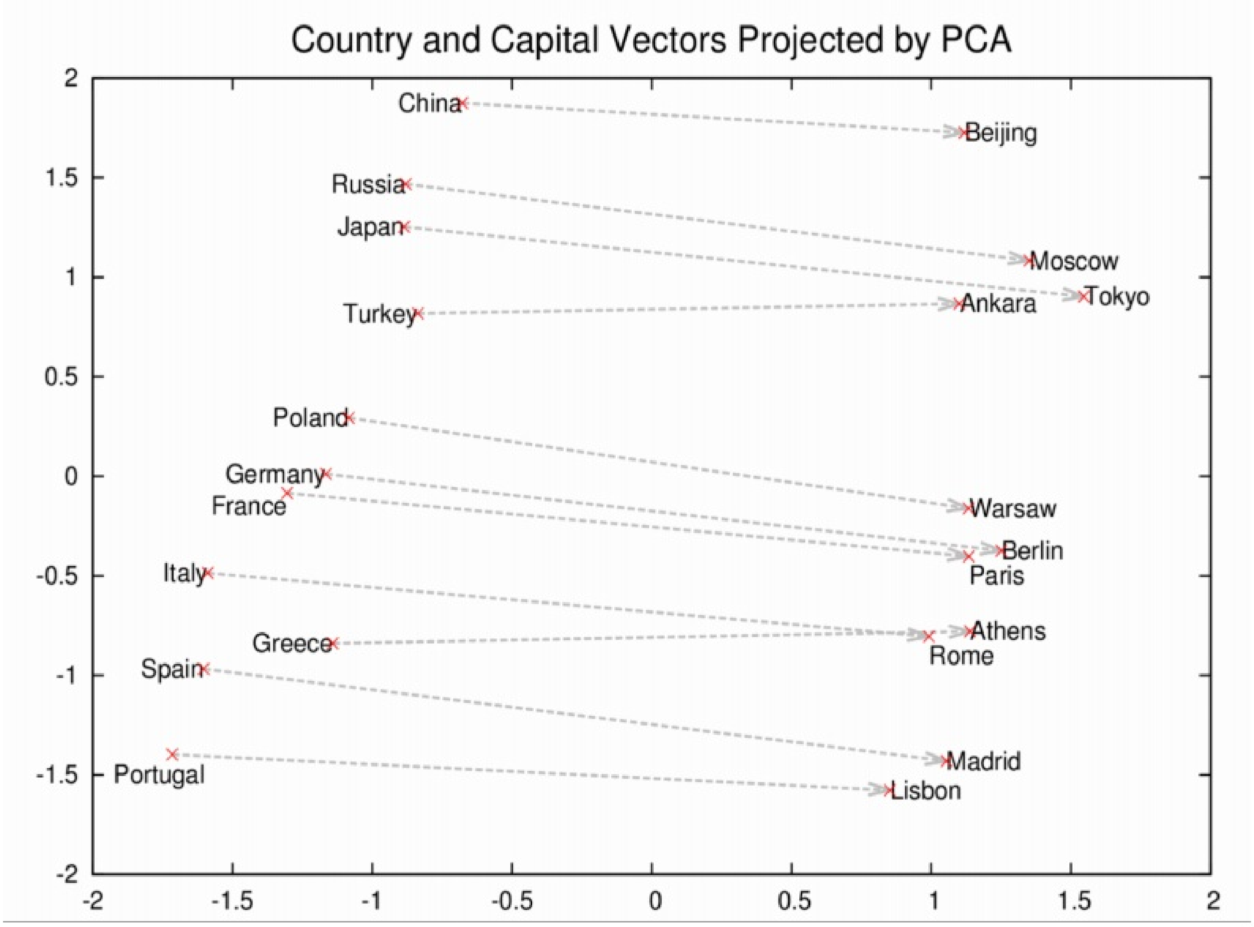
LSTM resembles human memory: it forgets old stuff (old internal state * forget gate) and learns from new input (input node * input gate)



back to top

word2vec

- Shallow, tw o-layer neural netw orks that are trained to construct linguistic context of w ords
- Takes as input a large corpus, and produce a vector space, typically of several hundred dimension, and each w ord in the corpus is assigned a vector in the space
- The key idea is context: w ords that occur often in the same context should have same/opposite meanings.
- Tw o flavors
    - continuous bag of w ords (CBOW): the model predicts the current w ord given a w indow of surrounding context w ords
    - skip gram: predicts the surrounding context w ords using the current w ord



back to top

Generative vs discriminative

- Discriminative algorithms model *p(y|x; w)*, that is, given the dataset and learned parameter, w hat is the probability of y belonging to a specific class. A discriminative algorithm doesn't care about how the data w as generated, it simply categorizes a given example
- Generative algorithms try to model *p(x|y)*, that is, the distribution of features given that it belongs to a certain class. A generative algorithm models how the data w as generated.

> Given a training set, an algorithm like logistic regression or the perceptron algorithm (basically) tries to find a straight line—that is, a decision boundary—that separates the elephants and dogs. Then, to classify a new animal as either an elephant or a dog, it checks on w hich side of the decision boundary it falls, and makes its prediction accordingly.

> Here's a different approach. First, looking at elephants, w e can build a model of w hat elephants look like. Then, looking at dogs, w e can build a separate model of w hat dogs look like. Finally, to classify a new animal, w e can match the new animal against the elephant model, and match it against the dog model, to see w hether the new animal looks more like the elephants or more like the dogs w e had seen in the training set.

back to top

Paramteric vs Nonparametric

- A learning model that summarizes data w ith a set of parameters of fixed size (independent of the number of training examples) is called a parametric model.
- A model w here the number of parameters is not determined prior to training. Nonparametric does not mean that they have NO parameters! On the contrary, nonparametric models (can) become more and more complex w ith an increasing amount of data.

back to top