What is machine learning?

**What are the applications of Machine Learning?**

Companies like Google, Facebook, Baidu, IBM, Microsoft uses ML extensively to push their respective ads to the relevant users.

Here are a few applications that you should know:

1. Banking & Financial services: ML can be used to predict the customers who are likely to default from paying loans or credit card bills. This is of paramount importance, as machine learning would help the banks to identify the customers who can be granted loans and credit cards.
2. Healthcare: It is used to diagnose deadly diseases (e.g. cancer) based on the symptoms of patients and tallying them with the past data of similar kind of patients.
3. Retail: It is used to identify products which sell more frequently (fast moving) and the slow moving products which help the retailers to decide what kind of products to introduce or remove from the shelf. In addition, machine-learning algorithms can be used to find which two / three or more products sell together. This is done to design customer loyalty initiatives, which in turn helps the retailers to develop and maintain loyal customers.

**Types of Machine learning**

**Support vector Machines**

**Hidden Markov model**

**RandomForest**

**GradientBoostingTrees (**random set of features or all used, fraction of samples for overfitting**)**

**Hyper-parameter(important ones) tuning in XGBoost**

1. Gamma: - it is a pseudo-regularization parameter in gradient boosting. It is dependent on training dataset and other parameters. Higher the Gamma, higher will be the regularization and at value 0, there is no regularization. Gamma values more than 20 are very high and should be used in conjunction with high max depth. Higher value of gamma means more positive loss reduction required to further split the trees. It becomes more conservative to adapt to complexity in the model. Gamma thus in a way help avoid overfitting by pruning the tree.
2. Max depth: - It is how many levels deep a tree is allowed to go. The larger the value, more is the overfitting of the training set. Good values range from 6 to 20.
3. Min child weight: - This is how each group in the tree has to be. Larger values are more robust than smaller values. However the higher max\_depth is, the higher this value should also be in order to avoid overfitting. In simple words if the minimum sum of instance weight is less than the min child weight, tree would stop further partitioning

**Deep Learning**

Deep learning is a family of advanced machine learning algorithms which used the concept of human brain neurons to model arbitrary functions to capture the real world non-linear complexities which traditional machine learning algorithms are not capable of. These algorithms require lot of data to perform.

Libraries like Theano and Tensorflow help in specific machine learning number-crunching operations like derivatives on huge matricies with large efficiency

Tree based Modeling

#TODO - https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/

**Deep Learning, NLP and intersection**:-

Deep Learning – is a subfield of machine learning. it is a family of advanced machine learning algorithms which used the concept of human brain neurons to model arbitrary functions to capture the real world non-linear complexities which traditional machine learning algorithms are not capable of. These algorithms require lot of data to perform.

It attempts to learn representations and output. It can learn unsupervised(from raw text) or supervised(with specific labels)

Deep learning in speech

Deep learning in vision

NLP – Natural Language Processing is a field at the intersection of computer science, artificial intelligence and linguistics.

Reason for Intersection of Deep Learning and NLP is that it will help computers to understand natural language like humans to perform useful tasks e.g. Question-Answering

Following steps can be performed in general:-

1. Speech (phonetic or phonological analysis) or text(OCR/tokenization)
2. Morphological Analysis
3. Syntactic Analysis
4. Semantic Interpretation
5. Discourse Processing

Applications of NLP – spell checking, keyword search, finding synonyms, extracting info from websites such as product prices, names, date, location etc., classification on reading level of text , sentiment analysis and identification on positive or negative. More complex tasks like machine translation, complex question answering or spoken dialogue systems.

NLP in industry –

1. Search (written and spoken)
2. Online Advertisements
3. Automated translation
4. Sentiment Analysis for marketing/trading/elections
5. Speech recognition
6. Automating customer support

Why NLP is complex?

NLP disambiguos examples

Deep Learning + NLP = Deep NLP

Earlier semantics were represented by Lambda Calculus but with deep learning its just a vector and calculations

Also traditional approaches involve using bag of words representation or hand designated negation features.

Question Answering –

A lot of feature engineering to capture real world e.g. regular expressions

**Word Vectors**

**Count based(traditional) vs Prediction based(unconventional)**

1. Document word concurrence matrix - lead to general topics
2. Window based concurrence matrix – captures both syntactic and semantic information

Dimensionality reduction on concurrence matrix X with the help of singular value decomposition

A word can be represented by a dense vector. How many dimensions to use is something an important decision based on the nature of the problem.

SVD doesn’t scale and computation time increases quadrtically

Word2vec is something as state of the art. Predict surrounding words in a window of length m of every word

import numpy as np

import random

*def softmax(x):*

*"""Compute the softmax function for each row of the input x. It is crucial that this function is optimized for speed because it will be used frequently in later code.You might find numpy functions np.exp, np.sum, np.reshape,np.max, and numpy broadcasting useful for this task. (numpy broadcasting documentation:*

*http://docs.scipy.org/doc/numpy/user/basics.broadcasting.html*

*"""tmp\_x = np.max(x.T, axis=0)*

*tmp\_exp = np.exp(x.T-tmp\_x)*

*x = tmp\_exp / np.sum(tmp\_exp, axis=0)*

*x = x.T*

*return x*



**Motivation: Why Learn Word Embeddings?**

NLP systems traditionally treat words as discrete atomic symbols, and therefore 'cat' may be represented as Id537 and 'dog' as Id143. These encodings are arbitrary, and provide no useful information to the system regarding the relationships that may exist between the individual symbols. This means that the model can leverage very little of what it has learned about 'cats' when it is processing data about 'dogs' (such that they are both animals, four-legged, pets, etc.). Representing words as unique, discrete ids furthermore leads to data sparsity, and usually means that we may need more data in order to successfully train statistical models.

Vector space models (VSMs) represent (embed) words in a continuous vector space where semantically similar words are mapped to nearby points ('are embedded nearby each other'). VSMs have a long, rich history in NLP, but all methods depend in some way or another on the Distributional Hypothesis, which states that words that appear in the same contexts share semantic meaning. The different approaches that leverage this principle can be divided into two categories: count-based methods (e.g. Latent Semantic Analysis), and predictive methods (e.g. neural probabilistic language models).

In a nutshell: Count-based methods compute the statistics of how often some word co-occurs with its neighbor words in a large text corpus, and then map these count-statistics down to a small, dense vector for each word. Predictive models directly try to predict a word from its neighbors in terms of learned small, dense embedding vectors (considered parameters of the model).

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 model (CBOW) and the Skip-Gram model. Algorithmically, these models are similar, except that CBOW predicts target words (e.g. 'mat') from source context words ('the cat sits on the'), while the skip-gram does the inverse and predicts source context-words from the target words. This inversion might seem like an arbitrary choice, but statistically it has the effect that CBOW smoothes over a lot of the distributional information (by treating an entire context as one observation). For the most part, this turns out to be a useful thing for smaller datasets. However, skip-gram treats each context-target pair as a new observation, and this tends to do better when we have larger datasets.

**Skip- gram model**

E.g. **the quick brown fox jumped over the lazy dog**

We first form a dataset of words and the contexts in which they appear. We could define 'context' in any way that makes sense, and in fact people have looked at syntactic contexts (i.e. the syntactic dependents of the current target word, see e.g. Levy et al.), **words-to-the-left of the target, words-to-the-right of the target**, etc. Using a window size of 1, we then have the dataset

**([the, brown], quick), ([quick, fox], brown), ([brown, jumped], fox), ...**

of (context, target) pairs. Recall that skip-gram inverts contexts and targets, and tries to predict each context word from its target word, so the task becomes to predict 'the' and 'brown' from 'quick', 'quick' and 'fox' from 'brown', etc. Therefore, our dataset becomes

**(quick, the), (quick, brown), (brown, quick), (brown, fox), ...**

of (input, output) pairs. The objective function is defined over the entire dataset, but we typically optimize this with stochastic gradient descent (SGD) using one example at a time (or a 'minibatch' of batch\_size examples, where typically 16 <= **batch\_size** <= 512)

**CBOW model**

Algorithmically, these models are similar, except that CBOW predicts target words (e.g. 'mat') from source context words ('the cat sits on the'), while the skip-gram does the inverse and predicts source context-words from the target words

**Sampling**

It is a statistical procedure concerned with the selection of the individual observation to make statistical inferences about the population.

Types of sampling –

Random sampling – simple random sampling, equal probability systematic sampling, stratified simple random sampling, multistage stratified random sampling, cluster sampling, Multistage cluster sampling,

Non-random sampling – availability sampling, quota sampling, expert sampling,

Probability or non-probability sampling

**Important Concepts:-**

**26-01-2017**

1. Mahalanobis distance – It is best suited to detect outliers in n-dimensions where n >1. Mahalanobis’ distance is a statistical measure of the extent to which cases are multivariate outliers, based on a chi-squared distribution. R's mahalanobis() function provides a simple means of detecting outliers in multidimensional data.

#TODO - <https://www.r-bloggers.com/outlier-detection-with-mahalanobis-distance/>

1. Goodness-of-fit - A goodness-of-fit test, in general, refers to measuring how well do the observed data correspond to the fitted (assumed) model

**Chi Square test, Kolmogorov–Smirnov test, Cramér–von Mises criterion**

#TODO - <https://www.r-bloggers.com/goodness-of-fit-test-in-r/>

1. Bootstrap sampling - If we don’t have enough data to train our algorithm then we can increase the size of our training set by randomly selecting items and duplicating them (with replacement).

#TODO - <http://www.statisticssolutions.com/sample-size-calculation-and-sample-size-justification/sampling/>

1. Adjusted Rand Score – Given the knowledge of the ground truth class assignments labels\_true and our clustering algorithm assignments of the same samples labels\_pred, the adjusted Rand index is a function that measures the similarity of the two assignments, ignoring permutations and with chance normalization. ARI requires knowledge of the ground truth classes while is almost never available in practice or requires manual assignment by human annotators

ARI = (RI - Expected\_RI) / (max(RI) - Expected\_RI)

#TODO - <http://scikit-learn.org/stable/modules/clustering.html#adjusted-rand-score>

1. For K-fold cross validation, what k should be selected?

#TODO - <http://stats.stackexchange.com/questions/61783/variance-and-bias-in-cross-validation-why-does-leave-one-out-cv-have-higher-var>

<https://www.quora.com/For-K-fold-cross-validation-what-k-should-be-selected>

1. How to deal with multi-collinearity situation in linear regression model? - To check multicollinearity, we can create a correlation matrix to identify & remove variables having correlation above 75% (deciding a threshold is subjective). In addition, we can use calculate VIF (variance inflation factor) to check the presence of multicollinearity. VIF value <= 4 suggests no multicollinearity whereas a value of >= 10 implies serious multicollinearity. Also, we can use tolerance as an indicator of multicollinearity.But, removing correlated variables might lead to loss of information. In order to retain those variables, we can use penalized regression models like ridge or lasso regression. Also, we can add some random noise in correlated variable so that the variables become different from each other. But, adding noise might affect the prediction accuracy, hence this approach should be carefully used.
2. Threshold in classification problems(Confidence Splitting criteria) - <http://nerds.airbnb.com/confidence-splitting-criterions/>
3. Accuracy paradox in an imbalanced dataset – Accuracy should not be used as the measure of model performance because accuracy might be only predicting major class correctly but we are interested only in the minor class. Therefore, we should use sensitivity i.e. true positive rate (also called as recall), specificity i.e. true negative rate, and precision i.e. positive predicted value). F measure to determine class wise performance of the classifier.

#TODO - <https://en.wikipedia.org/wiki/Sensitivity_and_specificity>

#TODO - <https://www.analyticsvidhya.com/blog/2016/03/practical-guide-deal-imbalanced-classification-problems/>

1. Ensemble Learning – use multiple algorithms for prediction and they combine output of multiple machine learning algorithms for getting more robust or generalized output which outperform all the individual models

#TODO - <https://www.analyticsvidhya.com/blog/2015/09/questions-ensemble-modeling/>

#TODO - <https://www.analyticsvidhya.com/blog/2015/08/introduction-ensemble-learning/>

#TODO - <http://mlwave.com/kaggle-ensembling-guide/>

1. R-squared and adj. R-squared – R-squared cannot determine whether the coefficient estimates and predictions are biased, which is why we must assess the residual plots. However, R-squared has additional problems that the adjusted R-squared and predicted R-squared are designed to address.Every time you add a predictor to a model, the R-squared increases or remains same.

#TODO - <https://discuss.analyticsvidhya.com/t/difference-between-r-square-and-adjusted-r-square/264/3>

**27-01-2017**

1. Entity disambiguation –

#TODO - <http://www.theatlantic.com/technology/archive/2011/03/does-anne-hathaway-news-drive-berkshire-hathaways-stock/72661/>