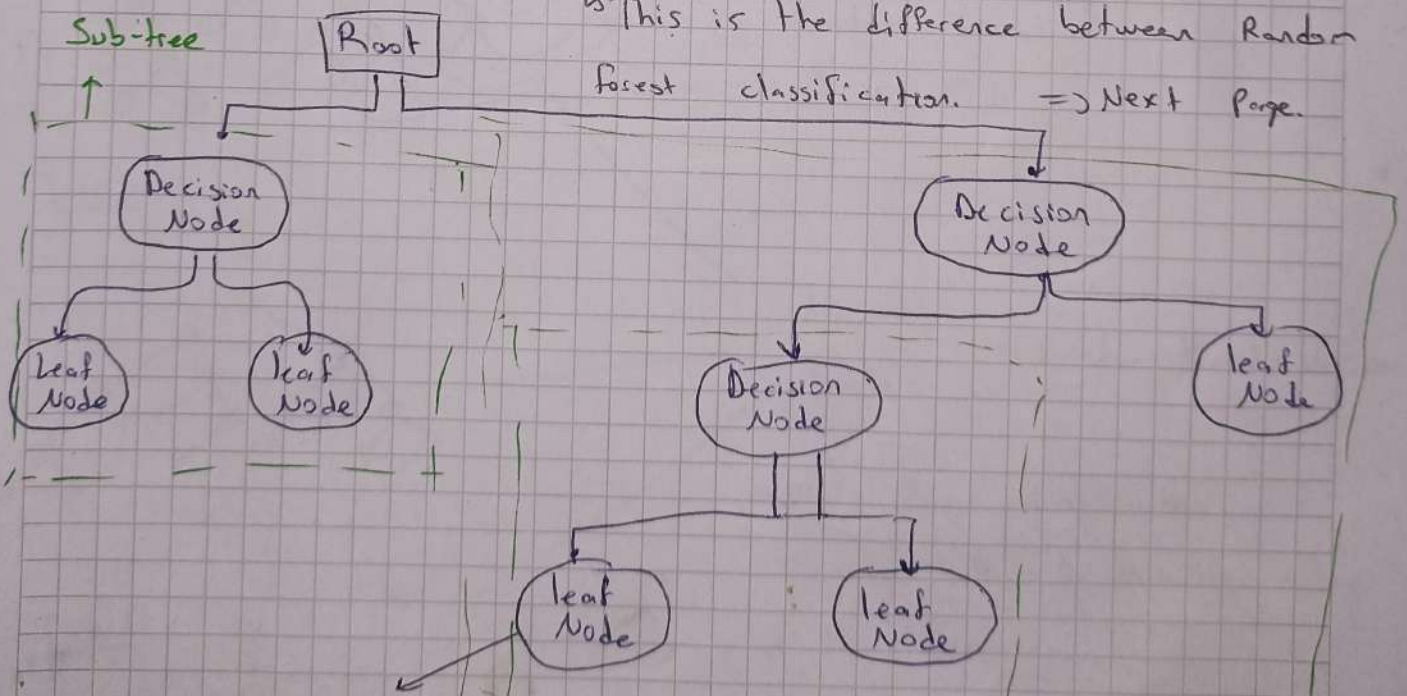
Abdurrahman BULUT
1901042258

CSE 454 - Data Mining
Assignment-1

Q1) Random Forest Classification.

A supervised machine learning technique called a random forest is build from decision tree algorithms. Random forest is a machine learning method for tackling classification and regression issues. It makes use of ensemble learning, a method for solving complicated issues by combining a number of classifiers. The random forest algorithm creates a "forest" that is arrived via bagging or bootstrap aggregation. The accuracy of machine learning algorithms is increased by bagging, an ensemble meta-algorithm. Based on the predictions of the decision trees, the random forest algorithm determines the result. It makes predictions by averaging or averaging out the results from different values of trees. The accuracy of the results grows as the number of trees increases.

This is the difference between Random forest classification. => Next Page.



if it is classification issue: R.F.C

if it is making some calculation: RF

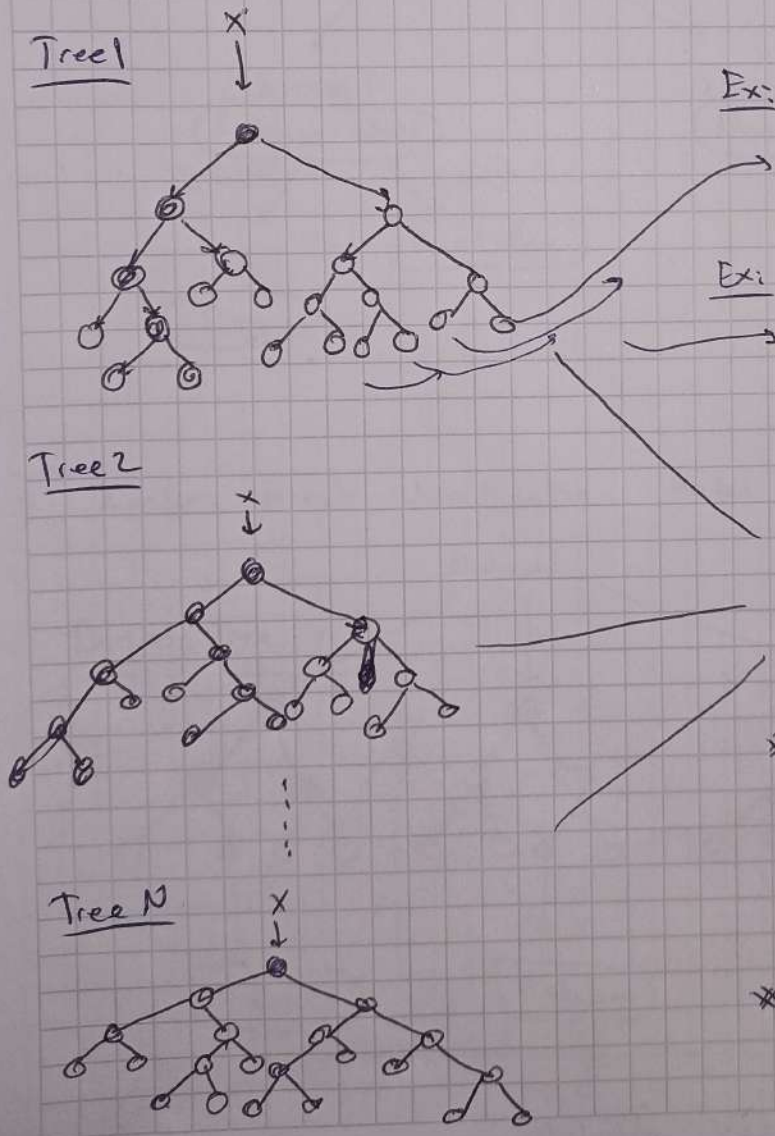
=>

Subject :

Date : ...../...../.....

The only difference between random forest classification and random forest is the result values in the leaves and the type of result obtained as a result. As I said in previous page, the outcome is established based on the prediction of the decision trees. It predicts by taking the average or median or something else. It will be defined by some aggregation methods.

But, for random forest classification method, calculations are not necessary for leaves node values. It sorts into classes based on the majority of values in the leaf.



Ex: Random forest classification

→ is Male or Female

Ex: Random Forest

→ it is 5.4

Majority Vote → Random forest's decision

* Random forest classification:
Majority vote is majority value in the leaves.

* Random forest:
Majority vote is calculated with some aggregation methods like, Mean.

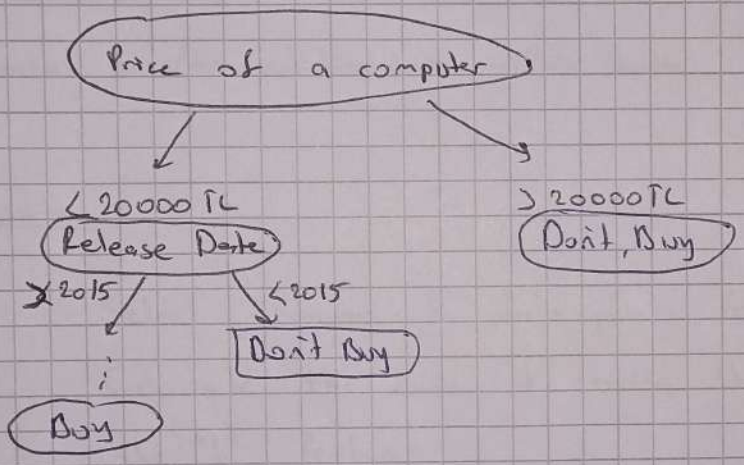
Subject :

Date :...../...../.....

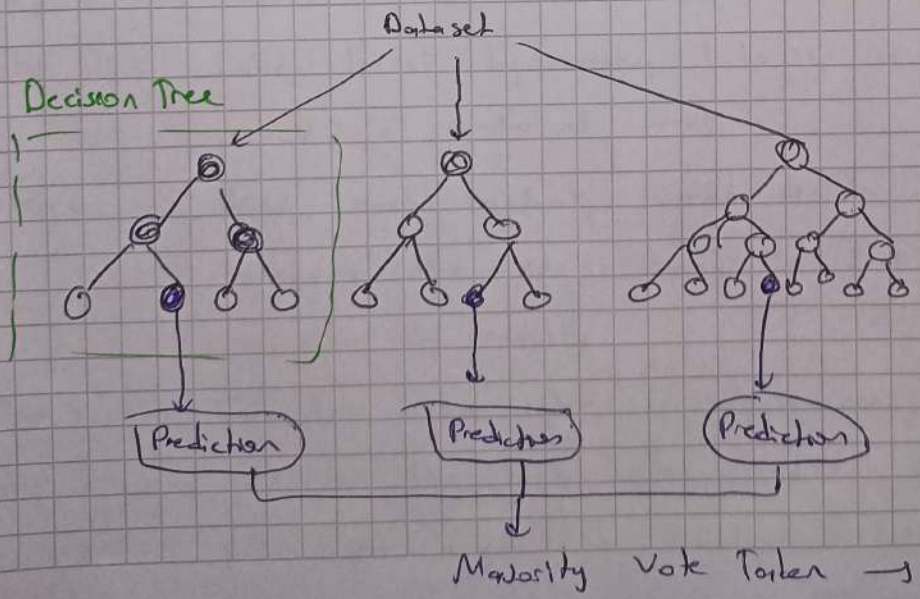
Features:

- It's more accurate than the decision tree algorithm
- It provides an effective way of handling missing data
- It can produce a reasonable prediction without hyper-parameter tuning
- It solves the issue of overfitting in decision trees.
- In every random forest tree, a subset of features is selected randomly at the node's splitting point.

Ex



Random forest classification Model example



Majority Vote Taken → Final Prediction

Subject :

Date :...../...../.....

Advantages

- It will give good results and it is easy to understand
- It is efficient on large datasets
- It gives better accuracy than decision tree
- It can be used for both regression and classification problems.
- Classifiers does not overfit with enough trees.

Disadvantages

- More resources are required for computation
- It consumes more time than decision tree
- It cannot describe relationships within data

Pseudo code

Step 1: Select random "k" data points from the training set (m)

Step 2: Calculate the node "d" using the best split point among the "k" features.

Step 3: Split the node into daughter nodes using the best split

Step 4: Repeat 1 & 3 steps until number of nodes has been reached

Step 5: Build forest by repeating 1 to 4 for "n" times to create "n" number of trees.

Step 6: Uses each randomly formed decision tree's rules to forecast the result using the test attributes, then records the results.

Step 7: Calculate the votes for each predicted target

Step 8: Consider the high voted predicted value as the final prediction from the random forest algorithm.

Subject :

Date :...../...../.....

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

Algorithm from another source

To generate c classifiers :

for i=1 to c do

Randomly sample the training data D with replacement to produce D_i

Create a root node, N_i containing D_i

call BuildTree(N_i)

endfor

BuildTree(N):

if N contains instances of only one class then

return

else

Randomly select x% of the possible splitting feature in N

Select the feature F with the highest information gain to split on

Create f child nodes of $N, N_1, \ldots N_f$, where F has f possible values($F_1, \ldots F_f$)

for i=1 to f do ;

Set the contents of N_i to D_i, where D_i is all instances in N that match F_i

Call BuildTree(N_i)

endfor

endif

Subject :

Date :...../...../.....

(Q2) Transfer learning

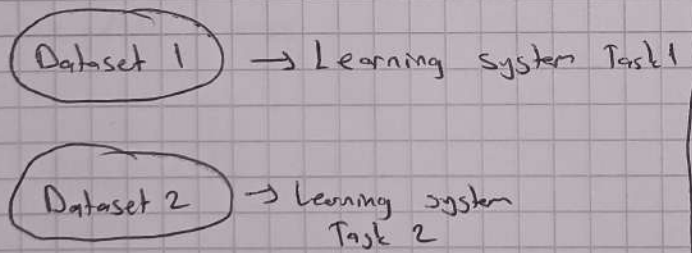
Transfer learning is a machine learning technique in which an AI that has been trained to perform a specific task is being reused (repurposed) as a starting point for another similar task. Transfer learning is widely used since starting from a pre-trained AI model can dramatically reduce the computational time required if training is performed from scratch.

"Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned"

Chapter 11: Transfer learning, Handbook of Research on ML applications, 2009

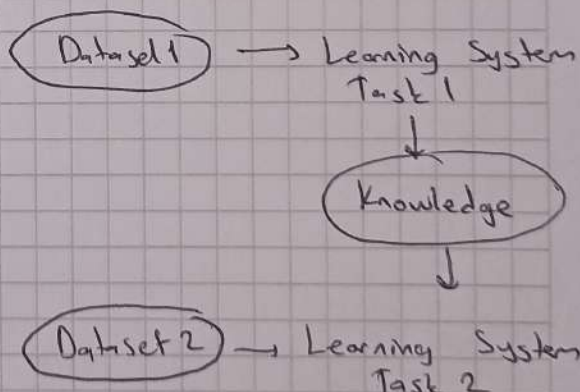
Traditional ML

- Isolated, single task learning

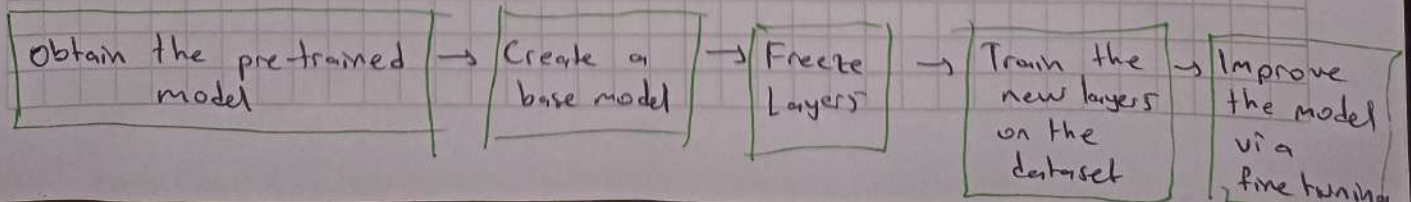


Transfer learning

- Learning of a new tasks relies on the previously learned tasks:



Transfer learning steps



Subject :

Date :...../...../.....

Example Model for transfer learning: "Xception"

Depthwise seperable convolutions are used in the deep convolutional neural network architecture known as Xception. Researchers from Google created it. Architecture Xception, which stands for "Extreme Inception". The Xception architecture has 36 convolutional layers forming the feature extraction base of the network. The data first goes through the entry flow, then through the middle flow which is repeated eight times, and finally through the exit flow.

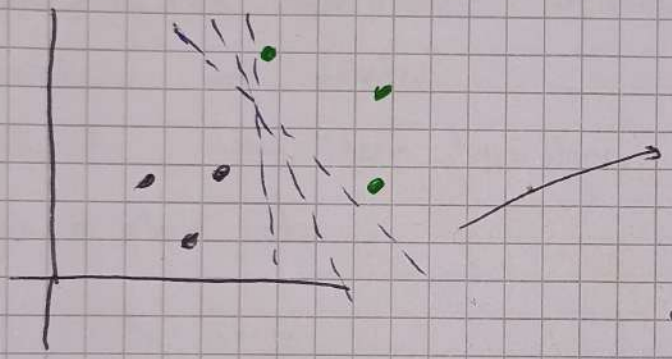
An open-source implementation of Xception using Keras and Tensorflow is provided as part of the Keras Applications module under MIT license

Subject :

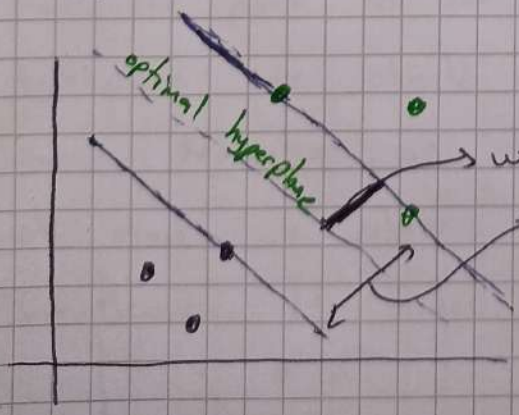
Date : ...../...../.....

Q3. SVM (Support vector Machine)

- Support vector machines, which examine data for classification and regression analysis, are supervised models with corresponding learning algorithms. Finding a hyperplane in an N-dimensional space (N = number of features) that categorizes the data points clearly. This is the goal of SVM algorithm. There are a variety of different hyperplanes that might be used to split the two classes of data points.

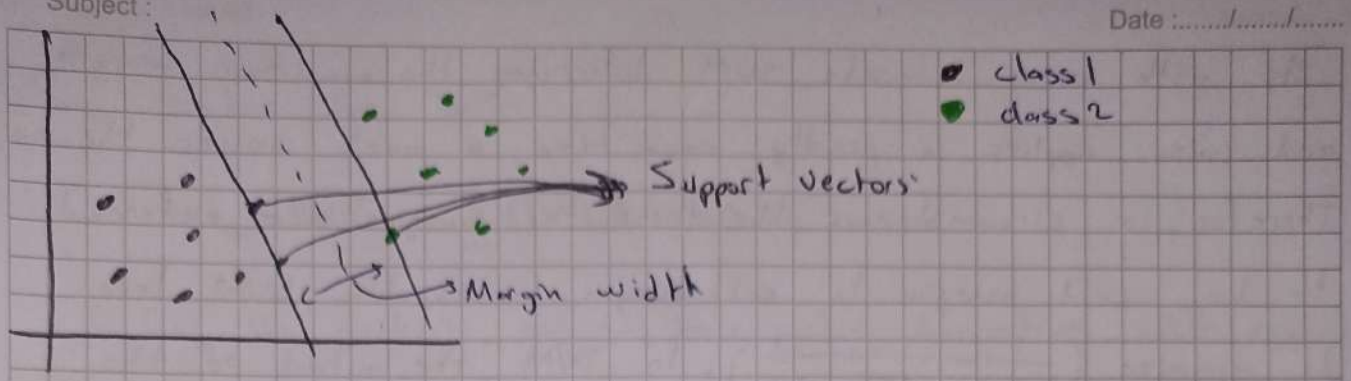


Finding a plane with the greatest margin, that is the greatest seperation between data points from both classes is our goal. Maximizing the margin distance adds some support, increasing the confidence with which future data points can be categorised.



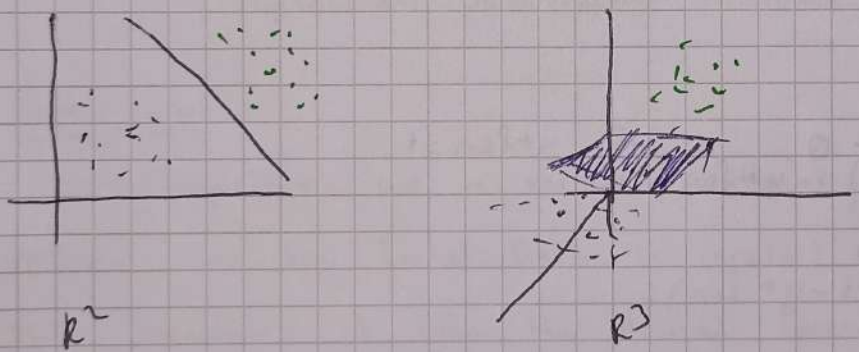
Subject :

Date :...../...../.....

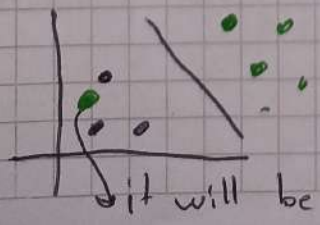


closer data points to the hyperplane are called support vectors. They have an impact on the hyperplane's position and orientation. By utilizing these support vectors, we increase the classifier's margin. The hyperplane's location will vary if the support vectors are deleted.

By the way, These hyperplanes can be in R^2 as a line, R^3 as a plane etc.



For the part so far, we can say that to select the best hyperplane, we should describe the one that represents the largest seperation or margin between two classes. If any data will place on other class, The SVM algorithm will ignore that. Because, It makes if an outlier. SVM is robust to outliers



Subject : Date :...../...../.....

As with prior datasets, SVM determines the maximum margin and also applies a penalty each time a point crosses the margin. Therefor, in circumstances like these, the margins are referred to be soft margins. In soft margin cases, the SVM tries to minimize $\left(\frac{1}{\text{margin}+\lambda(\Sigma\text{penalty})}\right)$. In SVM, the output of the linear function is taken into consideration. If the output is more than 1, it is associated with one class, and if it is less than 1, it is associated with a different class. We get this reinforcement range values ([-1, 1]) that serves as a margin because the treshold values in SVM are altered to 1 and -1. The goal of the SVM method is to increase the distance between the data points and the hyperplane. Hinged loss is the loss function that aids in maximizing the margin

Hinge loss function

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

$$c(x, y, f(x)) = (1 - y * f(A))_+$$

If the projected values and the actual value have the same sign, there is no cost. If not, we next determine the loss value. The cost function additionally receives a regularization parameter from us. The regularization parameter's goal is to strike a compromise between margin maximization and loss. The cost functions appears as follows when the regularization parameter has been added.

Subject :

Date :...../...../.....

Loss function for SVM

$$\min_w \lambda \|w\|^2 + \sum_{i=1}^{n} (1 - y_i \langle x_i, w \rangle)_+$$

To find the gradients, we can take partial derrivatives with regard to the weights.

Gradients

$$\frac{\delta}{\delta w_L} \lambda \|w\|^2 = 2\lambda w_L$$

$$\frac{\delta}{\delta w_L} (1 - y_i \langle x_i, w \rangle)_+ = \begin{cases} 0, & \text{if } y_i \langle x_i, w \rangle \geq 1 \\ -y_i x_{iL}, & \text{else} \end{cases}$$

$$w = w - a \cdot (2\lambda w) \quad \text{(No misclassification)}$$

$$w = w + a \cdot (y_i \cdot x_i - 2\lambda w) \quad \text{(misclassification)}$$

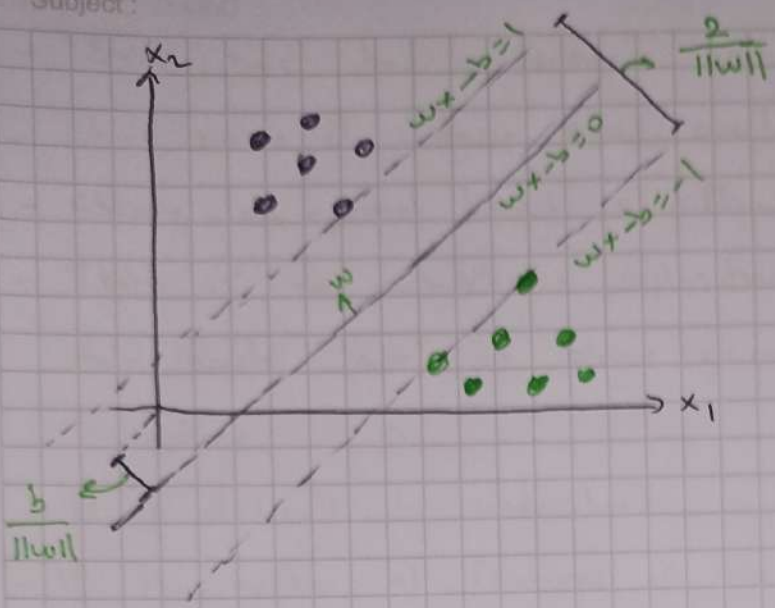
SVM kernel

SVM algorithms use a set of mathematical functions that are defined as the kernel. Data is inputted into the kernel, which then transforms it into the desired form. Different kernel functions are used by various SVM algorithms. There are various forms of these functions. For instance linear, nonlinear, polynomial, sigmoid and radial basis functions. It is mostly useful in non-linear seperation problems. The kernel functions return the inner product between two points in a suitable feature space.

$$K(\bar{x}) = \begin{cases} 1, & \text{if } \|\bar{x}\| \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

Subject :

Date :



Advantages of SVM :

- SVM is very effective even with high dimensional data
- when number of features is more than the number of rows of data, SVM can perform in that case as well.
- when there is a clear margin of seperation between classes. SVM works relatively well.
- SVM has a nature of Convex Optimization
- Outliers have less influence
- SVM can be used for both regression and classification problem
- SVM can work well with image data as well
- SVM is relatively memory efficient

Disadvantages of SVM -

- SVM takes more time to train for large datasets. For large datasets this can still give us rich feature space representations, but with many fewer dimensions than data points. It will not support large datasets and many dimensions at the same time.

=> Continued

Subject :

Date :...../...../.....

- SVM has bad performance on high noise. When the data has noise, it contains many overlapping points.
- Chosing an optimal kernel is difficult
- There is no probabilistic explanation for the classification if data points are on above or/and below the classifying hyperplane.
- It is difficult to understand
- If the number of features is much greater than the number of samples, avoid over-fitting in chosing Kernel functions and regularization term is crucial.

Subject :

Date :...../...../.....

Q(4) Fasttext Classification Model

Its main goal is the rapid and accurate processing of enormous datasets while finding scalable solutions for the text classification and representation problems. This model enables the development of supervised and unsupervised learning algorithms for the acquisition of word vector representations. By using two methods to handle classification and train word representations of text, it outperforms previously published state-of-the-art models in terms of computing efficiency and accuracy. These two methods are Hierarchical Softmax and word n-grams. Hierarchical softmax takes advantage of the unbalanced distribution of the classes to speed up computation. These different concepts are being used for two different tasks: efficient text classification and learning word vector representations.

To be efficient on datasets with very large number of categories, it uses a hierarchical classifier instead of a flat structure. This reduces the time complexities of training and testing text classifier from linear to logarithmic with respect to the number of classes. By using the Huffman algorithm to construct the tree used to represent categories, Fasttext additionally takes use of the fact that classes' are unbalanced. Because of this, the depth of the tree for highly frequent categories is shallower than for infrequent ones, improving computational efficiency.

Subject :

Date :...../...../.....

Additionally, FastText uses a low-dimensional vector to represent a text. This vector is created by adding vectors for each word that appears in the text. Each word in the lexicon of fastText has a low-dimensional vector associated with it. All classifiers for various categories share this hidden representation, making it possible for information about words learned for one category to be used by another category. These representations, often known as "bags of words," disregard the sequence of the words. To account for local word order, which is crucial for many text classification issues, fasttext also uses vectors to encode word n-grams.

A binary tree's label are represented by a hierarchical classifier. The binary tree's nodes each stand for a probability. The likelihood along the road to a given label serves as a representation of that label. This implies that the labels are represented by the binary tree's leaf nodes. When there are several categories and there is a class imbalance in the data, hierarchical Softmax appears to be quite effective. Instead of being organized in a flat, list-like layout, the classes are distributed in a tree structure. The Huffman coding tree, which employs shorter trees to represent more commonly occuring classes and longer trees for rarer, more rarely classes, is the foundation for the building of the hierarchical softmax layer. A depth-first search along the nodes across the many branches is used to investigate the likelihood that a specific text belongs to a class. Therefor, low probability branches can be ignored.

Subject :

Date : ..../..../.....

Fasttext employs the N-Gram technique to train the model when we have an unlabeled dataset. Important sequential informations are lost if the text is just represented as a bag of words. For huge datasets, word order consideration will turn out to be computationally expensive. For example, word is "[hecnic]"

[technic]

"technic" = ['t', 'ec', 'tec', 'tech', 'techn', 'techni', 'technic' ---]

There are some of the provided words' n-gram components. Only a handful of the numerous parts that make up this term are listed here in order to give an idea. When the model updates fasttext learns the weight for every n-gram along with the entire word token.

advantages:

- fasttext performs much better. It is much faster than training neural networks on multidimensional word vectors, and also achives good performance on the test set.
- Model builds semantic similarity between two words. 'tech' - 'technic'
- It allows for capturing the meaning of suffixes / prefixes for the given words in the corpus.
- It allows for generating better word embedding for different or rare words as well.
- It takes into account the internal structure of words while learning word representations.

Subject :

Date :...../...../.....

→ While using fasttext even if you don't remove the stop words still the accuracy is not compromised.

- It can also be used on morphological rich languages.

Disadvantages:

- fasttext uses more memory as it generates a lot of sub-words for each word.

Subject :

Date :...../...../.....

(Q5) when the class distributions are highly imbalanced, then class imbalanced problem will occur. There are different methods for overcoming the class imbalance problem. In literature on imbalanced classification, the most common methods employed are undersampling of the majority class, oversampling of the minority class, ensemble methods, cost-sensitive learning, asymetric classification, dimension reduction, treshold moving and feature selection

Undersampling:

The simplest approach to undersampling is to randomly select a fraction of records from the majority class. It works by decreasing the number of negative tuples. The random under-sampling method tries to balance the distribution of class by randomly removing majority class sample. The loss of important information is the issue with this approach.

Oversampling:

In terms of oversampling, the fundamental strategy is to enchance the cardinality of the classes by randomly duplicating the data in the minority classes. SMOTE is a very-well-liked strategy that improves diversity by producing phony minority class data. It resamples the positive tuples. In this method learning process consume more time because original dataset contain very small number of minority samples.

Subject :

Date :...../...../.....

Boosting and other ensemble methods such as Bagging:

They have demonstrated to be especially resillient when managing unbalanced data. A recent analysis of ensemble techniques used to address the issue of class imbalance is AdaBoost, for example, uses misclassified training methods/patterns to lessen the bias toward the majority class, while Bagging adds the idea bootstrap aggregation, which entails training multiple classifiers using bootstrapped copies of the initial training set. Due to their accuracy-focused nature, ensemble classifiers by themselves do not resolve the imbalance problem when applied directly to data. However, when combined with other methods, they produce effective outcomes.

Cost-Sensitive learning:

This method assigns a different cost to false negatif and false positive patterns:

Asymetric classifiers:

The main difference with cost-sensitive classification is that asymetric classifiers are not exclusively focused on assigning a different weight to false negative and false positive.

Dimension reduction:

By getting a set of principle variables, dimensionality reduction is the process of reducing the number of random variables being considered. It can be split into two categories:

Subject :

Date :...../...../.....

feature extraction and feature selection.

Feature selection

The process of choosing a subset of pertinent features to be used in model creation is known as feature selection. For high dimensional data sets, it uses filters that score each feature independently based on a rule. Because the class imbalance problem is commonly accompanied by the issue of high dimensionality of the data set, hence applying feature selection techniques is essential. Sampling techniques and algorithmic methods may not be enough to solve high dimensional class imbalance problems.

Treshold moving:

Perhaps the simple approach to handle a severe class imbalance is to change the decision treshold. Tuning or shifting the decision treshold in order to accommodate the broader requirements of the classification problem is generally referred to as "threshold-moving". Making predictions on a test dataset comes after the model has been fitted to a training dataset. The forecast take the form of scores that have been translated into normalized probability or normalized probabilities themselves. The crisp labels that arise are then assessed using a selected evaluation metric. after various threshold values have been attempted, when the model makes predictions based on new data in the future, it uses the threshold that produces the best evaluation metric.