General Linear Model

GLM in general refers to conventional Liner regression models for a continuous response variable given continuous and categorical predictors. It includes multiple liner regression as well as ANOVA and ANCOVA.

$$y_i \sim N(x_i^T \beta, \sigma^2)$$

x_i → Contains known Covariates.

β → coefficient to be estimated

These models are fit by least Squares and weighted least Squares using.

for Ex: SAS's GLM procedure or R's lm() function

Regression.

Liner regression

Prediction is done by apply complex a weighted sum of input features plus a constant called the bias term. / Intercept

=> Gradient Decent is used to tune parameters iteratively in order to minimize a cost fuction. => size of step is important

↓

Learning rate.

Logistic Regression (can be used for classification as well)

binary classifier. It is used to estimate not the probability that an instance belongs to a particular class

if the estimated probability > 50% then the instance belongs to class

→ way to reduce overfitting.

Regularized linear models. Regularization

Regularization → typically achieved by constraining the weights of the model.

a) Ridge regression → regularized version of linear Regression.
term $\lambda(\text{slope})^2$ → it not only fits the data but also keep the model weights as small as possible.
(regularized term)

a) Lasso regression
↳ least absolute shrinkage & selection operator.
it- tends to completely eliminate the weights of the least important feat.
it- performs feature selection.

c) Elastic net → middle btw Ridge & Lasso

Ensemble Techniques

Two most popular ensemble methods

Bagging & Boosting (here a single base learning algorithm is used)

Bagging is a technique for reducing prediction variance by producing additional data for training from dataset by combining repetitions with combinations to create multi-sets of the original data.

Boosting is an iterative strategy for adjusting an observation's weight based on the previous classification.

Ensemble method is used in ML to train multiple models or weak learners to resolve the same problem & integrated to gain desired results.

is weak models combined rightly give accurate models.

The Ensemble model made by Bagging & Boosting is known as homogeneous model.

There are some methods in which different types of base learning algorithms are also implied with heterogeneous weak learners to make a 'heterogeneous ensemble models'

Ensemble Techniques

Bagging → A homogeneous weak learners' model that learns from each other independently in parallel & combines them for determining the model average.

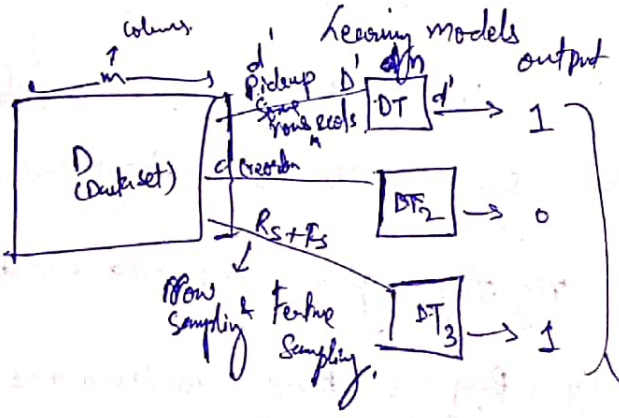
Boosting :- A homogeneous weak learners' model but works differently from Bagging. Here learners learn sequentially & adaptively to improve model predictions of a learning algorithm.

Ensemble Techniques
↳ combining multiple models.

① Bagging (Bootstrap aggregation)
② Random Forest
↳ Row sampling with replacement.

② Boosting
③ Ada Boost
④ Gradient Boosting
⑤ xg Boost

Random forest



D' < D
n < m
d' < d

DT1 & DT2 etc are different sets shuffling was done.

Decision Tree properties when we let DT to go all the depth.

① Low Bias
② High Variance

Cure is Random forest as it get trained for specific records & gives better test accuracy.

In Regression problem in RF we take either average or median of all the results, classifier takes majority of votes.

Ensemble methods :- Bagging, boosting, Stacking etc.

Hard voting classifier:- The class which gets the most votes

Bagging:- Uses the same training algorithm for every predictor but to train them on different random subsets of the training set.

(Bootstrap aggregating) → Here sampling is performed with replacement.

Pasting → when sampling is performed without replacement.

Feature Importance:- RF makes it easy to measure the relative importance of each feature.

Scikit-learn measures/determines how much the tree nodes that use that feature reduce impurity on avg.

Boosting:- Ensemble method that combines several weak learners into a strong learner.

→ general idea:- to train predictors sequentially → each trying to correct its predecessor

One way to do it is Ada Boost. → Pays more attention to the training instances that the predecessor underfitted.

↳ results in new predictors focussing more and more on the hard cases.

Ada Boost :- ① Base Classifier (DT classifier) is trained & used to make predictions on the training set.
② Relative weight of misclassified training instances is then increased.
③ A second classifier is trained using the updated weights & gain it to make predictions on the training set.
④ weights are updated and so on
⑤ Once all the predictors are trained, the ensemble makes predictions very much like bagging or pasting, except that predictors have different weights depending on their overall accuracy on the weighted training set.

Drawback :- It cannot be parallelized since each predictor can only be trained after the previous predictor has been trained.
→ result :- It does not scale as well as bagging or pasting.

Gradient Boosting :- Works by sequentially adding predictors to an ensemble each one corrected its predecessor at every iteration
→ Instead of tweaking weights like AdaBoost it tries to fit the new predictor to the residual errors made by previous predictors.

XG Boost :- An optimize implementation of Gradient Boosting available in python library XGBoost.
↓
Extreme Gradient Boosting.

Aims :- Being extremely fast, scalable & portable.

Stacking → Here instead of using trivial functions to aggregate the predictions of all predictors in an ensemble it trains a model to perform this aggregation.
→ the final predictor (Blender or meta learner) takes these predictions as inputs & makes the final predictions.

It is actually possible to train several blenders (one using Linear regression, other using RF etc - -)

Trick is to split the training set into three subsets.

→ The first one is used to train first layer.

→ Second one is used to create the training set. used to train the second layer. (using predictions made by predictors of the first layer)

→ third one is used to create the training set to train 3rd layer. (using predictions made by the predictors of the 2nd layer)

→ Once it is Done, we can make prediction for a new instance by going through each layer sequentially.

SVM

SVM → capable of performing linear or non linear classification, regression and even outliers detection.

→ well suited for classification of complex but small or medium sized datasets.

Linear SVM classification

→ two classes can ~~be~~ ~~easily~~ clearly be separated easily with a straight line.

→ The st. line not only separates the two classes but also stays as far away from the closest training instances as possible.

Hard margin classification

↳ only work for data which is linearly separable & it is quite sensitive to outliers.

Soft margin classification

↳ more flexible., to keep the street as large as possible & limiting the margin violation.

Non linear SVM classification

↳ to add more features. (poly, rbf) ← slow for high complex features.

Decision Tree (Classification & regression task → versatile) & multioutput tasks

→ capable of fitting complex data.

Start from root node (depth = 0, at the top)

root's child nodes (Left & Right) → depth 1 and so on.

↓ if condition satisfies.

Leaf node → doesn't have any child node.

Decision trees require very little data preparation & do not require feature scaling at all in particular.

node's value attribute tells how many training instances of each class this node applies to.

When a node is pure (gini = 0) if all the training instances belong to same class.

$$G_i = 1 - \sum (P_i^2)$$

A DT can also estimate the probability that an instance belongs to a particular class.

reduction of entropy is often called an information gain

$$Hi = -\sum P_{i,k} \log_2 (P_{i,k})$$

Random forest is an ensemble of Decision Trees.
here it is easy to measure the relative importance of each feature.

Boosting : To train predictors sequentially.

Ada Boost : Adaptive. firstly base classifier is trained & used to make predictions on training set.
Second classifier is trained using the updated weights. & so on. it does it by the instance weights at every iteration

Boost & Random forest come under ensemble

Gradient Boosting
Is works sequentially like AdaBoost
Adding predictors each one correcting its predecessor.
This method tries to fit the new predictor to the residual errors made by previous Predictor

Learning rate : steps, low learning rate (0.1) needs more trees in the ensemble to fit the training set.
To find optimal no of trees early stopping are used.