OLS (Ordinary Least Squares Method – Regression)

Intro and Formula Derivation:

Ordinary Least Squares Estimators - derivation in matrix form - part 1,
Ordinary Least Squares Estimators - derivation in matrix form - part 2
Ordinary Least Squares Estimators - derivation in matrix form - part 3

Assumptions:

Watch Video Instead of Blogs - ols linear regression assumptions
Videos - Gauss MArkov Assumptions 1 , Gauss Markov Assumptions 2

I) The regression model is linear in the coefficients and the error term
II) The error term has a population mean of zero
III) All independent variables are uncorrelated with the error term
IV) Observations of the error term are uncorrelated with each other
V) The error term has a constant variance (no heteroscedasticity)
VI) No independent variable is a perfect linear function of other explanatory variables
VII) The error term is normally distributed (optional)

Checking Predicted Results are Trust Worthy

Resisual Plot Analysis, Goodness of Fit and R2

Decision Trees

Videos : Let’s Write a Decision Tree Classifier from Scratch

Blogs : Classification and Regression Analysis with Decision Trees , Alogorithm Explained

Code:

```
From sklear.model_selection import test_train_split
Xt,yt,xv,yv = test_train_split(X,Y,test_size = 0.2, random_seed = 5, stratify = y)

From sklearn.preprocessing
From sklearn.tree import DecisionTreeClassifier
Tree = DecisionTreeClassifier(Criterion = 'Entrophy',max_depth=4)
Tree.fit(x,y)

From sklear.tree import DecisionTreeRegressor
tree = DecisionTreeRegressor(criterion='mse',max_depth=3)
tree.fit(X, y)
```

How to avoid/counter Overfitting in Decision Trees?

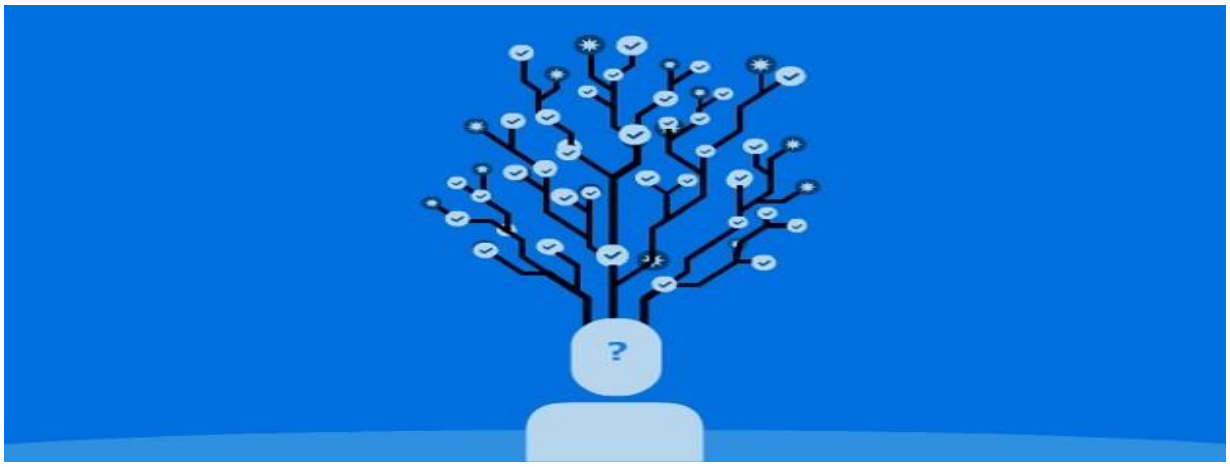
The common problem with Decision trees, especially having a table full of columns, they fit a lot. Sometimes it looks like the tree memorized the training data set. If there is no limit set on a decision tree, it will give you 100% accuracy on the training data set because in the worst case it will end up making 1 leaf for each observation. Thus, this affects the accuracy when predicting samples that are not part of the training set.

Here are two ways to remove overfitting:

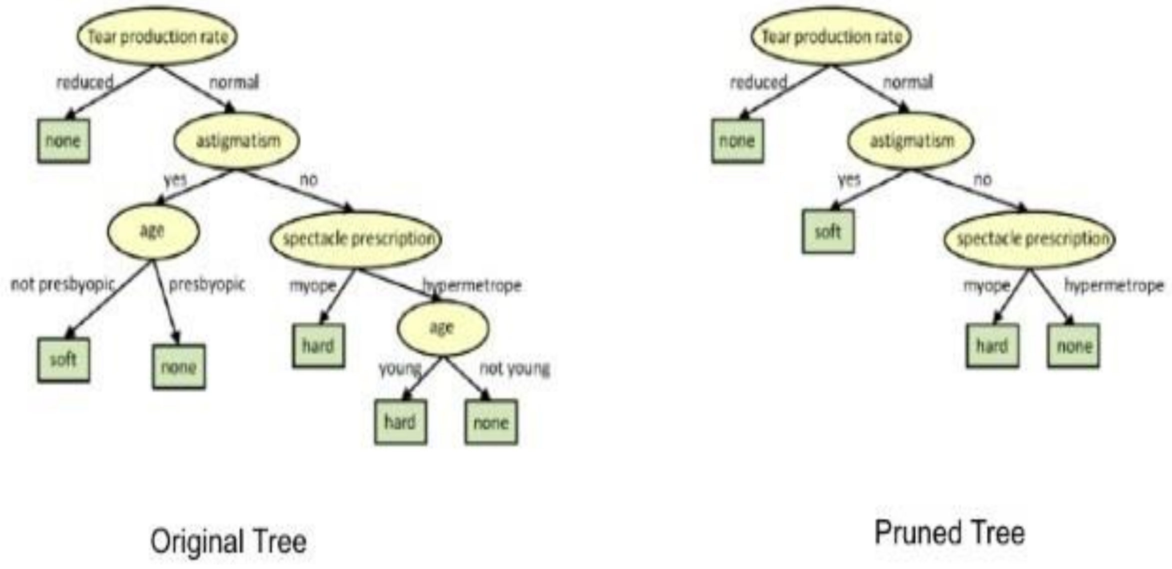
1. Pruning Decision Trees.
2. Random Forest

Pruning Decision Trees

The splitting process results in fully grown trees until the stopping criteria are reached. But the fully grown tree is likely to overfit the data, leading to poor accuracy on unseen data.



In **pruning**, you trim off the branches of the tree, i.e., remove the decision nodes starting from the leaf node such that the overall accuracy is not disturbed. This is done by segregating the actual training set into two sets: training data set, D and validation data set, V. Prepare the decision tree using the segregated training data set, D. Then continue trimming the tree accordingly to optimize the accuracy of the validation data set, V.



In the above diagram, the 'Age' attribute in the left−hand side of the tree has been pruned as it has more importance on the right−hand side of the tree, hence removing overfitting.

Which is better Linear or tree-based models?

Well, it depends on the kind of problem you are solving.

3. If the relationship between dependent & independent variables is well approximated by a linear model, linear regression will outperform the tree−based

model.

4. If there is a high non−linearity & complex relationship between dependent & independent variables, a tree model will outperform a classical regression method.
5. If you need to build a model that is easy to explain to people, a decision tree model will always do better than a linear model. Decision tree models are even simpler to interpret than linear regression!

SVM

22 February 2021 22:20

Intro and Intuitive Understanding

Intro

Mathematics behind

Math

Advantages and Disadvantages

https://towardsdatascience.com/solving-svm-stochastic-gradient-descent-and-hinge-loss-8e8b4dd91f5b

Advantages:

- SVM works relatively well when there is a clear margin of separation between classes.
- SVM is more effective in high dimensional spaces.
- SVM is effective in cases where the number of dimensions is greater than the number of samples.
- SVM is relatively memory efficient.

Disadvantages:

Choosing a “good” kernel function is not easy.

- Long training time for large datasets.
- Difficult to understand and interpret the final model, variable weights and individual impact
- SVM algorithm is not suitable for large data sets (Computational Problems)
- SVM does not perform very well when the data set has more noise i.e., target classes are overlapping.
- In cases where the number of features for each data point exceeds the number of training data samples (Over Fit).
- As the support vector classifier works by putting data points, above and below the classifying hyperplane there is no probabilistic explanation for the classification.

Tuning Hyperparameters

- **Kernel**: The main function of the kernel is to transform the given dataset input data into the required form. There are various types of functions such as linear, polynomial, and radial basis function (RBF). Polynomial and RBF are useful for non−linear hyperplane. Polynomial and RBF kernels compute the separation line in the higher dimension. In some of the applications, it is suggested to use a more complex kernel to separate the classes that are curved or nonlinear. This transformation can lead to more accurate classifiers.
- **Regularization**: Regularization parameter in python's Scikit−learn C parameter used to maintain regularization. Here C is the penalty parameter, which represents misclassification or error term. The misclassification or error term tells the SVM optimization how much error is bearable. This is how you can control the trade−off between decision boundary and misclassification term. A smaller value of C creates a small−margin hyperplane and a larger value of C creates a larger−margin hyperplane.

- **Gamma**: A lower value of Gamma will loosely fit the training dataset, whereas a higher value of gamma will exactly fit the training dataset, which causes over−fitting. In other words, you can say a low value of gamma considers only nearby points in calculating the separation line, while the a value of gamma considers all the data points in the calculation of the separation line.

Naive Bayes

Intro

Theory

Code:

```
From sklearn.naive_bayes import GaussianNB
From sklearn.preprocessing import LabelEncoder

Dataframe['column'] = LabelEncoder().fit_transform(data['some_column'])
```

For one hot encoding

```
Df_with_one_hoy_encode = pd.get_dummies(df,columns=['columns'],prefix=['columns'])
```

Assumptions:
The naive Bayes algorithm does that by making an assumption of conditional independence over the training dataset.

Hyper Parameters:
smoothing parameter

Types of Hyper Parameters:

There are three types of Naive Bayes model under the scikit−learn library:

- **Gaussian:** It is used in classification and it assumes that features follow a normal distribution.
- **Multinomial:** It is used for discrete counts. For example, let's say, we have a text classification problem. Here we can consider Bernoulli trials which is one step further and instead of "word occurring in the document", we have "count how often word occurs in the document", you can think of it as "number of times outcome number x_i is observed over the n trials".
- **Bernoulli:** The binomial model is useful if your feature vectors are binary (i.e. zeros and ones). One application would be text classification with 'bag of words' model where the 1s & 0s are "word occurs in the document" and "word does not occur in the document" respectively.

What are the Pros and Cons of Naive Bayes?
Pros:

y and fast to predict class of test data set. It also performs well in multi class prediction When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data. It performs well in case of categorical input variables compared to numerical variable(s) For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons:

- I
  t

  i
  s

  e
  a
  s

- If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.
- On the other side naive Bayes is also known as a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.
- Another limitation of Naive Bayes is the assumption of independent predictors. In real life, it is almost impossible that we get a set of predictors which are completely independent.

Tips to improve the power of Naive Bayes Model

Here are some tips for improving power of Naive Bayes Model:

- If continuous features do not have normal distribution, we should use transformation or different methods to convert it in normal distribution.
- If test data set has zero frequency issue, apply smoothing techniques “Laplace Correction” to predict the class of test data set.
- Remove correlated features, as the highly correlated features are voted twice in the model and it can lead to over inflating importance.
- Naive Bayes classifiers has limited options for parameter tuning like alpha=1 for smoothing, fit_prior=[True|False] to learn class prior probabilities or not and some other options . I would recommend to focus on your pre−processing of data and the feature selection.
- You might think to apply some *classifier combination technique like* ensembling, bagging and boosting but these methods would not help. Actually, “ensembling, boosting, bagging” won’t help since their purpose is to reduce variance. Naive Bayes has no variance to minimize.

Linear Regression

Linear Regression

I believe that everyone should have heard or even have learned about the **Linear model** in Mathethmics class at high school. **Linear regression** is the simplest and most extensively used statistical technique for predictive modelling analysis. It is a way to explain the relationship between a dependent variable (target) and one or more explanatory variables(predictors) using a straight line. There are two types of linear regression − **Simple** and **Multiple**.

Quick reminder: **4 Assumptions of Simple Linear Regression**

1. ***Linearity****: The relationship between X and the mean of Y is linear.*
2. ***Homoscedasticity****: The variance of residual is the same for any value of X (Constant variance of errors).*
3. ***Independence****: Observations are independent of each other.*
4. ***Normality****: The error(residuals) follows a normal distribution.>*

HyperParameters
Learning Rate, Regularization coefficient

Advantages and Disadvantages
https://www.geeksforgeeks.org/ml-advantages-and-disadvantages-of-linear-regression/

Logistic Regression

INTRO

Assumptions:

However, some other assumptions still apply.

First, binary logistic regression requires the dependent variable to be binary and ordinal logistic regression requires the dependent variable to be ordinal.

Second, logistic regression requires the observations to be independent of each other. In other words, the observations should not come from repeated measurements or matched data.

Third, logistic regression requires there to be little or no multicollinearity among the independent variables. This means that the independent variables should not be too highly correlated with each other.

Fourth, logistic regression assumes linearity of independent variables and log odds. although this analysis does not require the dependent and independent variables to be related linearly, it requires that the independent variables are linearly related to the log odds.

HyperParameters

Regularization Coefficient, learning rate in case algorithm uses SGD classifier

Advantages and dis advantages:

advs&disadvs

K means

This is how the algorithm works:

1. K centroids are created randomly (based on the predefined value of K)
2. K−means allocates every data point in the dataset to the nearest centroid (minimizing Euclidean distances between them), meaning that a data point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other centroid
3. Then K−means recalculates the centroids by taking the mean of all data points assigned to that centroid's cluster, hence reducing the total intra−cluster variance in relation to the previous step. The "means" in the K−means refers to averaging the data and finding the new centroid
4. The algorithm iterates between steps 2 and 3 until some criteria is met (e.g. the sum of distances between the data points and their corresponding centroid is minimized, a maximum number of iterations is reached, no changes in centroids value or no data points change clusters)

Intro and explanation

INTRO

K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks

Advantages

K−means is a must−have in your data science toolkit, and there are several reasons for this. First of all it's **easy** to implement and brings an **efficient performance**. After all, you need to define just one parameter (the value of K) to see the results. It is also **fast** and works really well with **large datasets**, making it capable of dealing with the current huge volumes of data. It's so **flexible** that it can be used with pretty much any datatype and its **results are easy to interpret** and more explainable than other algorithms. Furthermore, the algorithm is so **popular** that you may find use cases and implementations in almost any discipline.

Disadvantages

Nevertheless, K−means presents some *disadvantages*. The first one is that you need to define the **number of clusters,** and this decision can seriously affect the results. Also, as the location of the initial centroids is random, results may not be comparable and show lack of consistency. K−means produces clusters with **uniform sizes** (each cluster with roughly an equal quantity of observations), even though the data might behave in a different way, and it's very sensitive to outliers and noisy data. Additionally, it assumes that data points in each cluster are modeled as located within a sphere around that cluster centroid (**spherical limitation**), but when this condition (or any of the previous ones) is violated, the algorithm can behave in non−intuitive ways.

Hyper Parameters: No of Clusters

How to Find valid k:

The Elbow Method

This is probably the most well−known method for determining the optimal number of clusters. It is also a bit naive in its approach.

Calculate the Within−Cluster−Sum of Squared Errors (WSS) for different values of k, and choose the k for which WSS becomes first starts to diminish. In the plot of WSS−versus−k, this is visible as an elbow.

Within−Cluster−Sum of Squared Errors sounds a bit complex. Let's break it down:

The Squared Error for each point is the square of the distance of the point from its representation i.e., its predicted cluster centers.

The WSS score is the sum of these Squared Errors for all the points.

Any distance metric like the Euclidean Distance or the Manhattan Distance can be used.

The Silhouette Method

The silhouette value measures how similar a point is to its own cluster (cohesion) compared to other clusters (separation).

Source: Wikipedia

The range of the Silhouette value is between +1 and −1. A high value is desirable and indicates that the point is placed in the correct cluster. If many points have a negative Silhouette value, it may indicate that we have created too many or too few clusters.

From wikipedia:

a(i)=a(i)= average distance between point ii and other points in the same cluster

b(i)=b(i)= average distance between point ii and all points in the closest cluster

s(i)=(b(i)−a(i)) ƒ max{a(i),b(i)}

Hierarchical Clustering

Intro

https://www.kdnuggets.com/2019/09/hierarchical−clustering.html

How we measure goodness?
Dunn Index −
Dunn's index is the ratio between the minimum inter−cluster distances to the maximum intra−cluster diameter. The diameter of a cluster is the distance between its two furthermost points. In order to have well separated and compact clusters you should aim for a higher Dunn's index.

<u>Advantages</u>

1) No apriori information about the number of clusters required.

2) Easy to implement and gives best result in some cases.

<u>Disadvantages</u>

1) Algorithm can never undo what was done previously.

2) Time complexity of at least O($n^2 \log n$) is required, where *'n'* is the number of data points.

3) Based on the type of distance matrix chosen for merging different algorithms can suffer with one or more of the following:

i) Sensitivity to noise and outliers

ii) Breaking large clusters

iii) Difficulty handling different sized clusters and convex shapes

4) No objective function is directly minimized

5) Sometimes it is difficult to identify the correct number of clusters by the dendogram.

dbscan

https://www.kdnuggets.com/2020/04/dbscan−clustering−algorithm−machine−learning.html

HYPER PARAMETERS: min pts, epsilon

Advantage

- DBSCAN can find arbitrary shaped clusters using MinPts parameters
- The order of the point in the database is insensitive
- Handle noise and outliers

Disadvantage

- Cannot perform well with large differences in densities
- Not suitable when various density involves

PCA

https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
pca.fit()
```

Correlation between all kinds of variables

https://medium.com/@outside2SDs/an-overview-of-correlation-measures-between-categorical-and-continuous-variables-4c7f85610365

Imp Links

This repository contains a curated list of awesome open-source libraries that will help you deploy, monitor, version, scale, and secure your production machine learning.

https://github.com/EthicalML/awesome-production-machine-learning

Associative Rule Mining

(Google it)

Boosting and Bagging Decision Trees

(Google it)