**STANFORD UNIVERSITY**

**STATS LEARN COURSE – REGRESSION PROBELM**

(Response variable,target Variable , Dependent variable) - The variable we want to predict.= Y

X= Set of predictors,inputs,features,attribute) variables we want to use to predict Y values

X = a vector = ( X1 X2 X3 )

**Our Model generates a inferring function which is able to map X values to Y and relate them .**

**Y = F(X) + E(errors) ,**  y as a function of X , and their relationship.

How one influences other.

Given X = (X1 , X2 , X3,…Xp) a input space with p components of a training data , we want to see which inputs Affects the Y values (target ) values the most and which are irrelevant , and how they are related to each other.

And their distributions.

Eg- we have a plot with a regression line which fits some data

We want to predict the value of Y at some X=4 , but there can be many values of Y at X=4 for some large *training* data set, then how do we predict them ?

**TAKING THE AVERAGE OF ALL THOSE output(y) VALUES(MEAN) for those X points = EXPECTED(mean) VALUE**

**F(4) = E (Y | X = 4) , E = expected value**

**SO WHAT IS THE F(x) MAPPING FUNCTION in REGRESSION – *F(X) = E( Y | X = x ) = REGRESSION Function***

How Do we know about the accuracy of the model and how correct are the calculated Y values ?

**Error (LOSS) function= E( (f(x)actual - F(x)calcuated )^2 | X = x) = | (Yactual - Ycalculated )^2| + var(e)**

F(x)calculated = output(Y) which the model calculated using the mapping function which it generated and learned.

f(x)actual = actual Y values for the training data

*We calculated the sum of squared errors , and our main aim is to minimize this Error function or optimize the Model in some way such that we obtain minimum error . Also called OLS(ordinary Least squared errors) – Regression is a OLS model.*

**2)K- NEAREST NEIGHBORS PROBLEM -**

But what if the test point that we give the model to predict the Y value doesn’t actually exist in the Data Set , then how do we generate the Mapper Function F(X) ?

Ans – Using Nearest Neighbors

In this case we calculate the Nearest local points to that particular test Point ‘q‘ using a Distance Metric d() which calculates all the nearest points to the test instance .

And we calculate the Y value of the Test(Query point) using the **average of outputs(Y) of those nearest neighboring local points** .

**F(X) = AVG( Y | X = N(x) ) , N(x) = set of all the local (nearest) data points to that Query Point q.**

**CURSE OF DIMENTIONALITY**

**BUT, Nearest Neighbor averaging can be pretty good for a small ‘p’ (dimensions) , (*p <= 4)***

**p = no of predictor(INPUT=X) variables & large K = no of nearest neighbors used to average Y values and give us our estimate**

**This is because it is very hard to find New near neighborhoods in high dimensions, as the points will no longer be LOCAL and nearer , as the neighborhood boundary is going to increase as we increase the dimensions (parameters p ) *and the points no longer be closer or Local*.**

**How do we deal with this , We introduce structural to our Model .**

**LINEAR MODEL – SIMPLEST STRUCTURAL MODEL**

**P =features(inputs X = ( x1,x2,x3 …. Xp )**

**Y = F(x) = ( a + b1\*X1 + b2\*X2 + b3\*X3 + b4\*X4 … . Bp\*Xp)**

**a = y - Intercepts**

**b= slopes-coefficients – It is nothing but the multiplicative change in Y(target) w.r.t 1 unit change in X(inputs) , as slope = rate of change of Y wrt x**

**Linear Models are not always correct and accurate but they serve as a good Functional approximation to the unknown True Function f(x) - the actual relation b/w training data x and y .**

**Eg y = income , x1 = seniority , x2 = education**

**F(X) = Income = f (seniority-x1 , education –x2) = a + b1\*seniority + b2\*education**

The above example captures the actual relationship b/w Income and the 2 features and how x1 and x2 affects the Incomes. It has 3 parameters.

**Thin Plate Spline** –A Regression Model **Optimizer** **\smoother** which adds more flexibility to the Model and fits the training data better with **less Training Error.** – **A BLACK BOX PREDICTOR**

But sometimes more Flexibility and More complexity such as using splines as smoothening techniques and Using polynomial(higher degree –X^n) regression cause OVERFITTING .

**SOME TRADE OFFS**

**1)Prediction Accuracy vs Interpretability**

**-Linear Models are easy to interpret , thin plate splines are not.**

**2) BIAS(Underfitting) – VARIANCE(overfitting) Tradeoffs**

**-Good fit vs Under or Over Fit.**

**PARSIMONY vs BLACK BOX :**

**We prefer simpler and consistent(not too complex) Models(hypothesis,Rules) involing fewer variables over a Too complex, Flexible , Black Box predictor(Model) involving them all.**

1. **Linear Regression Models are Simpler Models and not much Complex.**
2. **Thin plates Spline and K-NN are more complex.**

**Now we Need a way to access a Model’s accuracy and performance ?**

**For a model function F(x) fitting it to some training data Tr = {Xi,Yi}**

**To calculate error in Tr = MSETr = AVG [{ Yi - F(x) }^2]**

**Y =actual Target values in the training data, F(x) = outputs calculated by the Model’s Function**

**But it is sometimes biased towards OVERFITTING , i.e Training Error is gets lower when a model is highly flexible and Complex .**

**Test Error = MSETe = Test Data = {xi , yi } = AVG [ (Yi – F(x)^2 ) ]**

**What actually happens now ?**

1. **For More complex models , which were very flexible and fitted the training Data very well , They had very less** **MSETr (Training Error) , but when we Tested the accuracy of the Model on an unseen Test Data Set Te, The Test Error (MSETe)**

**Was very high and kept on increasing as the model became more and more complex and Flexible .**

**EXCEPTION- For quiet wiggly true functions having low noise , then The more Complex Models Do quiet a good job.**

**BIAS (F(x) ) = E[ F(x) – f(x)) ] , F(x) = Function that the Learner(learning algorithm) Generated,**

**f(x) = true function of our data set which reated Xs and Y.**

1)Hence now as the Flexibility,Complexity of the Model increases the variance increases and the Bias Decreases, And the Model becomes more prone to Overfit on Training Data.

2) Now when the simplicity of the Model increases the Bias increases and the Variance decreases and the Model becomes more prone to Underfitting.

We actually want to select a Model which neither under-fits or Over-fits ,hence that is not much simple and not much complex. This is called BIAS-VARIANCE TRADEOFF .

Basically Model Selection will depend on the type of Problem and Type of Data we are dealing with. Sometimes Flexible and Complex Models such as K-NN and Thin Plates SPLINES will do better , or Vice versa. All perfomance depends on the MSE on TEST DATA , or we can use various other OPTIMIZATION techniques to improve accuracy of the Model.

Eg - Introducing the quadratic term or increasing the polynomial degree of the predictors will make your Linear model more complicated and Complex .More complicated & Complex models typically have lower bias at the cost of higher variance. This has an unclear effect on Reducible Error-MSEtr and MSEte (could go up or down) and no effect on Irreducible Error.

CLASSIFICATION PROBLEM

Classification is Supervised learning Technique in which the response (target) variable is **Discrete--Qualitative**(Nominal,categorical) ,eg( True/False, Yes/No , 0-10 etc).

It aims at understanding the relationship b/w the input (Xi) and Target(Y) variables in the Training Data and based on that try to Generalize to unseen Test Data which it didn’t saw during training using the Training Data.

The curve Generated b/w the Y and X of the training data is actually the Probability values of observing a Y value of 1 or 0 in a Binary Classification Problem.

What is an Ideal Classsifier function C(x) ?

Suppose there are K labels in C(Class) = Y

So the function generated by a classifier is Conditional Probability of Observing a Y = K , given X =x

Pk(x) = P ( Y = k | X = x ) , k = 1,2,3,4 …K

Hence this is called the **Bayes Optimal Classifier which uses the Conditional Probabilities to Classify a Test point.**

**K-NN can also be used in Classification in the same way like regression but instead of computing the AVERAGES of the Y values , we calculate the VOTES, i.e Proportions of the Majority of class labels (y) in the K – nearest neighborhood set of training examples .**

**And we classify the Test point ‘q’ on the basis of the ‘y’ value which majority of the nearest neighbors had(The Y value which had the Maximum VOTES).**

**And same things happen here i.e As the Dimention ‘d’ grows , the size of the nearest neighborhood keeps on growing and eventually the data points don’t remain nearer and Local anymore. ->>CURSE OF DIMENTIONALITY**

We measure the Perfomance of a Classifier based on the Misclassification

Error Rate

ErrTest = Avgi in Test Data [ Yi != C(Xi) ], Yi = actual Target Class corresponding to the Xi values in Test set , C(Xi) = Classifier function which the Learner Generated from the Training Data which calculated some Class(Y) for test Data set. It is simply termed as Misclassifications or the number of Mismatches

*K-NN is a very power full Classifier- it is said that for 1/3 of the classification Problems, Best option would be using K-NN algorithm as a Classifier.*

*NOTE:*

*ALSO AS THE VALUE OF K INCREASES THE BIAS INCREASES AND IT TENDS TO HAVE LESS VARIANCE AND LESS OVERFITTING, because we will have enough nearest,local data points to look over and calculate the class label of test data using the proportions(probabilities) of class labels in the K-Nearest neighbor set.*

*BUT FOR SMALL VALUES OF K IN K-NN ALGORITHM THERE ARE MOSTLY CHANCES THAT WE OVERFIT TO THE TRAINING DATA.*

*As a rule of thumb we usually select the value of K as the no obtained form 10% of the data points in Training Data.*