Notes and Bits

A univ.ai compendium

ROHIT GOSWAMI, AMICHEME

June 21, 2019

Contents

These are to be used in conjunction with the materials which have all been forked to my own github repo. As such I will link to them and these are to be used mainly to capture important aspects of the discussion. The topics listed below have videos and slides in the repositories listed.

Topic	Repo
Regression (KNN and Linear)	Regression
Selection and Regularization	ValidationRegularization
Logistic Regression	Classification
Classification Trees and Bagging	Trees
Enhanced Sampling (Boosting)	Boosting
Perceptrons	ANN
Neural Networks (Regularization and Optimization)	ANN2

Furthermore there are several topics covered as labs with papers.

Topic	Repo
Convolutional Neural Networks	CNN {One, Three, Four}
Language Models	Lang {One, Two, Three}
RNNs (Ensembles)	CaptionHack

Predictors

The measurement erorr is irreducile. Inspite of the way the measurements are independent, we are able to sample over time. The basic concept is that the confidence intervals for the predictors can be estimated. This now allows us to treat it as a cumulative distribution, and therefore we can calculate the standard errors. Now confidence intervals may also be obtained.

Bootstrap

This basically is a quasi-equilibrium assumpion. This allows for the fomulation of multiple populations, and then as a consequencuntly

> Bootstrap aggregating, also called bagging, is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression.

The quoted text is from wikipedia.

Standard Errors

Hypothesis Testing

This is the formal process through wich we evaluate the validyty of a statistical hypothesis by considering evidence for or against the hypothesis gathered by random sampling of the data.

Validation

Essentially the take away is that though we can and will fit perfectly to the training data as we increase the polynomial degree, we will actually see a **decrease** in the accuracy metric (R^2) as the degree is increased due to overfitting.

The large number standard (around >50) for large data-sets without multimodes is a good idea. Basically, don't always use the general $80:20$ for train:test and 25% for validation.

The errors increase with the degree, not because of error propogation, but because **the sensitivity has increased** due to the fact that at higher degrees we have overfitting.

Regularization

Interestingly, using gradient techniques will not give us any better coeffients in this form. That is, the regularization is not achieved by optimization. So in effect, we need more than the loss function, we need the penalty function.

To wit, when using regularization for linear cases, we are essentially using it to reduce the dimensionality of our space. That is it is somewhat like feature selection or dimensional reduction.

LASSO and Ridge

So from the geometry and the fact that is more probable that from the LASSO formulation we will obtain many zeros (it forms a square and the points are most likely). The ridge is faster though. The LASSO is slower because it depends on the L_1 norm which is not exactly differentiable (due to it being a modulus).

Advanced Regression

Closed form regression

So from the euclidean norm, we can use the inner product aspect to get the derivative. Ridge

Norms

Euclidean Distance (L2)

This is the L2 and is basically the distance.

L1 Norm

This is just the modulus, or the absolute sum of the values.

General Norms

So given a p-norm, we have the sum for every element, to the p^{th} power, and then raise the entirity of it to $\frac{1}{p}$

> Can't we apply a transform to ensure the the isocontour of our Ridge regression is actually on the axes as well? Just like the corners of the L1 regularization fall on the axes.

achuta@ucla.edu

Logistic Regression

Logistic regression depends on predicting the $Y = 1$ class, or the positive class. Now the softmax is the generalized logistic regression mapping function, but is basically the linear one in the equation below.

$$\frac{1}{1 - e^{-(\beta_0 + \beta_1 X)}}$$

So one unit change is an e^{β_1} in the odds that $Y = 1$

- This is also called a discriminative classifier, as they directly model the probability of some class being present on the basis of the features

Sparsity

JPEG is by convention 8 by 8 bits. Consider the DCT (Discrete Cosine Transform).

Low freqency variations are more common in images. The SPIE handbook on compression this is really very good.

So it turns out that when we have colors to be encoded, then we deal with them as linear combinations, like 3 matrices for an RGB image. The mosaic class of methods are used to extrapolate other colors from existing ones to reduce complexity.

Sparsity helps work with scale invariace of image sets, which can then be used for image processing. Compressive sensing, or compressed sensing. This is used for taking tiny datasets and forming a lot of information from it. This is apparently lossless (must ask for the proof). The proof is by restricted isometry. An approximate isometry property. A matrix A is said to satisfy the RIP of order K with isometry constant. (ask for the rest)

Decision Trees

So this is different from KNN because KNN does not look at variables, basically it is a clustering algorithm. Clustering algorithms may also look at density and but they do not consider labels. In a decision tree we simply draw the line where the probability of being in both classes is equal. The decision tree approach does not look at the density per-se. KNN is predictive as opposed to inferential. So the decision boundary we put is sort of not just where the probability is equal, it is technically wherever we need to put the positive prediction threshold.

So the confidence of the model can only be visualized (via noting how peaked the histogram is). Basically given the same point on the decision boundary, there may be multiple curves through it but they will have different ”sharpness-factors”.

> Basically, a flow chart whose graph is a tree (connected and no cycles) represents a model called a decision tree.

It turns out that though we can actually use **m-nary** trees instead of binary trees, but it is more relevant (computationally) to use the binary one instead.

Errors

The Gini index is used to determine purity post split. Another method is to use the maxiumum misclassification rate, weigh it according to the number of points, and move towards the minimum of that.

Classification Trees

Regression Trees

When we discuss these, then it is important to note that local effects are better captured by the tree regression techniques compared to the polynomial and linear regression methods. This means that the linear/polynomial version is global in nature, while the tree model is local. Naturally, the linear/polynomial has a guarantee to a minimum, but the regression tree, typically being piecewise linear, the regression tree will be better for situations where there is less smooth. So we can test the Hessian and see if the data is smooth, if it is the polynomial or linear regression is better, while if it is a sort of jagged system, then the polynomial will be of a high degree and therefore we don't want to try that.

There is no need for standardization typically because this is one feature at a time.

- In both cases we report only accuracies from the leaf nodes, that is the R^2
- In both cases we report only accuracies from the leaf nodes, that is the R^2 for the regression case, and the accuracy for the classification.

It has been proven that it is computationally more tractable to use decision trees when compared to piecewise linear regression.

Weighted Samples

The most important thing about this is that though we do use weights which effectively change the distribution, this is ONLY done to the **training-set**. The test and validation sets must be as biased as the original training set.

Impudation

This seems to be the way you fix missing data. Conditional impudation is when you look at it. KNN is also used for the imputation of points. KNN imputation is expensive. Typically we can throw away columns if there is a lot of points missin.

Machine Learning and Denoising

So a dictionary learning thing is something which can be used to learn a compressible basis, sort of like the DCT or the DFT. Well it is better to consider it as a **sparse basis**.

The basic concept is that an image, can be broken down into a n-dimensional representatin with relative energies of the frequencies.

Now with the n-dimensions, we can create a matrix A (which is fat as it contains m-dimensions, and $n < m$). Now we have a measurement of n-dimensions, so we can have the

L_0 Norm This is simply the number of nonzero numbers.

Learning a dictionary is the main problem, this is done by K-SVD. Basically we need the dictionary to enforce the constraint that the sparse solution we obtain from the LASSO formalization, should be such that it corresponds to a true image, i.e. it ensures that only the first n-columns of the matrix have values and the rest are 0s.

- The sparsest solution is proven to be unique.

Check and read this book.

Artificial Neural Networks

Overfitting and Regularization

- Dropout is a bit slower due to the weighted probabilities implementiation
- Random noise is also a valid way to prevent overfitting (Adversarial)
- Early stopping is typically the most popular

Convolutional Neural Networks

The basic concept of working with these is to generate kernels such that each one extracts information from the image.

- There are no guarantees that the kernels learned by the covolutional networks correspond to meaningful image transforms
- Imposing constraints on the intermediate layers has not yet shown any promise
- Convolutions in space are not are the same as the signal impulse domain transforms, since these smooth them out

Dilated CNNs

So here we basically skip some connections during the inner layers. Essentially we have a wider receptive field. We do **not** lose information simply by using dilated CNNs since they are **lossless** (just a bunch of multiplications). Technically this is not true when we use max pooling in between the subsequent layers, but even so the trade off is probably worth it. A stride will skip a few pixels and therefore lose information.

Saliency Map

This is done to determine the most relevant section of the image and it essentially consists of taking the partial derivative of the loss function w.r.t. every single input pixel, however in practice this is simply part of back-propogation.

Max Pooling and Back Propagation

Max pooling averages (smears out) the information, so it is problematic for backpropagation. The fix for this is a rather novel way of book-keeping. It basically stores the derivative and the corresponding value of the previous layer, then we simply increment the value in the previous layer (this is best seen in the slides).

Language Embeddings

Personality Traits and Sentiment Analysis

- A simple logistic regression on a bag-of-words model works quite well.
- Typically, an older style used a bunch of grammar rules corresponding to various sentiments. Nowadays it is more expedient to embed the language into a representative space and use a neural network.
- The cosine similarity (the dot product) of the vector of traits (the embedding) is the metric for determining, say, personality traits.
- Since this is an angle oriented measure, it is kind of scale independent in the sense that it can be used to compare systems of different vector lengths.

Words

For words, or categorical systems, the one-hot-encoding causes the cosine similarity to mess up as the construction causes the values to zero out.

- To circumvent this scenario, a **lower dimensional** embedding is learnt, such that the similarity is maintained. In practice this corresponds to a linear regression over other metrics (for different **tasks**) which are then used to obtain ranks or scores with numeric content so the cosine similarity is maintained.
- Latent spaces are used along with non negative matrix methods to encode these into a space.
- The best way to get better embeddings is to use multiple tasks for originally obtaining the space. This is used in the newer neural natural language processing systems like BERT, ELMO etc.