COMP 4670/8600: Introduction to Statistical Machine Learning

Assignment II

Manab Chetia, u5492350

1. **Parametric Model**

Given error function ,

It is also given,

where lies in the subspace spanned by .

Let denote the subspace spanned by .

… (a)

If is in span of then for some coefficient .

Let denote the subspace, which is orthogonal to S, spanned by and lies in . Therefore for some coefficient .

… (b)

Thus from (a) and (b), is a linear combination of basis functions and a linear combination of basis of the nullspace (orthogonal subspace of S) i.e.

… (c)

As , therefore

Now is a monotonically increasing function and

Let’s consider a vector that does not have a perpendicular component. A vector that does not have a perpendicular component is always smaller than a vector that has perpendicular component i.e.

.

This leads to .

This leads to only.

… (d)

Therefore

Thus from (d) and above , we can say which minimizes will be a linear combination of basis functions .

1. **Conditional Probability and Variance via Parzen Estimator**
2. Given,

data points and training set .

The joint probability distribution for the a new input and target is given as,

Also given,

Conditional density function is given as,

Total area under the normal curve is 1, i.e.

Therefore,

Translating along the axis, we get

where kernel is given as

Conditional mean given as,

Conditional variance is given as,

We know if has mean and covariance , then , then

1. is given as,

Denominator is the total probability mass in the sample. is the PDF at divided by the total mass, therefore sums to 1 i.e.

.

# *References:*

*Pattern Recognition and Machine Learning by Christopher M Bishop*

1. **Maximum Margin Hyperplane**
2. We are given 2 data points from classes and respectively. Let denote the class of positive samples and denote the class of negative samples.

Therefore and .

Let the equation of the decision boundary be

Assuming and are located at a distance of 1 on the either side of the decision boundary. Therefore,

Now multiplying the above equations by and respectively, we get,

Or we can write

Now,

X1-X2

X2

w

X1

The width of the strip is

Now from

For a positive sample () and a negative sample (), we can get

, ,

Therefore width of the strip is

We want to maximize the width of the strip or minimize subject to constraints .

Therefore making use of Lagrange multipliers we get,

Setting the derivatives of with respect to and and maximizing with respect to , we get

Eliminating and using these conditions we get the Dual Representation of the maximum margin Hyperplane in which we maximize,

subject to constraints and , we get

Let’s define the kernel function

Therefore,

To maximize we get the partial derivatives and set them to 0.

From the above equations and considering constraint we get,

Substituting in , we get

and then

KKT conditions say,

Thus for every point or

Also,

As for points for which , it won’t contribute to the above sum.

Therefore in our case as , therefore i.e. are support vectors.

From , we get

Now, assigning to and ,

Our Hyperplane is

Therefore we have determined the maximum-margin hyperplane with only 2 data points and it is irrespective of the dimensionality of data space.

1. A plane is defined by dependent and independent variables. At most one dependent variable is necessary for the definition of a plan. The other variables can be independently defined using the dependent variable. Therefore dimension of the plane is always .
2. In order to classify new points, we evaluate the sign of

If sign is +ve, we classify as *Class1*  and if sign is –ve, we classify it as *Class2*.

1. The Lagrange parameters are:

From proof in (a), when , .

As in our as case, as , therefore are support vectors.

1. **Constructing New Kernels**

Given are valid kernels and *A* is a symmetric Positive Semi Definite matrix.

Necessary and Sufficient condition for a function to be a kernel is Gram Matrix *K*, whose elements are given by is positive semi definite for all possible choices of set .

Since *k3* is a valid kernel, applying it to any set of vectors yields a positive semi definite matrix. Therefore is a valid kernel.

We know,

is a valid kernel function since it is an inner product in .

We take the Eigendecomposition of A i.e . *V* is an orthogonal matrix and is a diagonal matrix containing +ve eigen values. Let be the diagonal matrix containing the square roots of eigen values. Let Therefore

is an inner product using linear feature mapping *B*  and therefore is a valid kernel function.

Let and be kernel matrices of and applied to any set of vectors.

Let be any vector and

From (b) we get and , therefore write,

Let and be kernel matrices of and applied to any set of vectors and be the tensor product of and . The tensor product of 2 positive semi definite matrices is itself a positive semi definite matrix since the eigenvalues of the product are pairs of products of eigenvalues of the components ( and ). We know Hadmard product in kernel space becomes tensor product in feature space. Therefore matrix corresponding is the Hadamard product *H* of and . *H* is a principal submatrix of *K*. Also for any , there exists a corresponding such that . Therefore *H* is positive semi definite and is a valid kernel.

# *References: Kernel Methods for Pattern Analysis by John Shawe Taylor, Nello Cristiani*

1. **Kernel for XOR**
2. We are given 4 points in (d = 2 here) space and Radon’s theorem states that any set of points can be partitioned into 2 disjoint sets whose convex null intersect. Therefore 4 points can only be separated into 2 subsets only with intersecting convex hulls only. Therefore 4 points are linearly not separable.

Also let’s assume, there exists a plane , such that it can linearly separate the points . Points belongs to and belongs to .

Therefore,

Now,

This leads to,

Adding the first 2 expressions we get

Adding the last 2 expressions we get

Therefore we get no solution for and . Therefore our assumption is wrong and no plane exists which can linearly separate and .

1. Given

For ,

This kernel corresponds to mapping,

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Class** |  |  |  |  |  |
| **Class 1** | 1 | 1 | 1 | 1 |  |
| -1 | -1 | 1 | 1 |  |
| **Class 2** | 1 | -1 | 1 | 1 | - |
| -1 | 1 | 1 | 1 | - |

It can be seen the 2 classes are linearly separable on axis.

Now considering

The above kernel is valid only for and . A function is a kernel only if i.e. should return the inner product between the images of 2 arguments.

Now, if ,

The above expression cannot be expressed in terms of inner products between and . This can be generalized to any

Now if ,

The above expression cannot be expressed in terms of inner products between and .

Now if ,

The above expression cannot be expressed in terms of inner products between and . This can be generalized to for any

Therefore .

Now coming to case, , we have only proved in part , that the points are linearly not separable as points to the linearly separable case.

Therefore my choice of in is .

# *References:* [*http://en.wikipedia.org/wiki/VC\_dimension*](http://en.wikipedia.org/wiki/VC_dimension)

1. **(Semi/Un) Supervised Learning and EM**

a) Class “Iris-Setosa”:

Mean: [ 5.006 3.428 1.462 0.246]

Covariance: [[ 0.12424898 0.09921633 0.0163551 0.01033061]

[ 0.09921633 0.1436898 0.01169796 0.00929796]

[ 0.0163551 0.01169796 0.03015918 0.00606939]

[ 0.01033061 0.00929796 0.00606939 0.01110612]]

Class “Iris-Versicolor”:

Mean: [ 5.936 2.77 4.26 1.326]

Covariance: [[ 0.26643265 0.08518367 0.18289796 0.05577959]

[ 0.08518367 0.09846939 0.08265306 0.04120408]

[ 0.18289796 0.08265306 0.22081633 0.07310204]

[ 0.05577959 0.04120408 0.07310204 0.03910612]]

Class “Iris-Virginica”:

Mean: [ 6.588 2.974 5.552 2.026]

Covariance: [[ 0.40434286 0.09376327 0.3032898 0.04909388]

[ 0.09376327 0.10400408 0.07137959 0.04762857]

[ 0.3032898 0.07137959 0.30458776 0.04882449]

[ 0.04909388 0.04762857 0.04882449 0.07543265]]

For cross-validation errors please refer to: *u5492350\_ManabChetia\_Supervised.py*

Classification Error is varying from

b)

|  |  |  |
| --- | --- | --- |
| Start | Log likelihood | Expected Log likelihood |
| 1 | -189.34 | -211.52 |
| 2 | -180.35 | -184.98 |
| 3 | -193.45 | -205.64 |
| 4 | -189.68 | -218.97 |
| 5 | -194.69 | -201.97 |

For the table of log likelihood and expected log likelihood, please refer to the Code: *u5492350\_ManabChetia\_Unsupervised.py*

*(I am checking for convergence by observing the difference between the New Log Likelihood and the Old Likelihood is within a threshold).*

We do not find the same solution each time as the convergence depends on the choice of starting points, as EM tries to increase the likelihood on each iteration, therefore we might get stuck on different local maxima or stationary points.

1. Now we are given two sets of data:

* Labeled Data
* Unlabeled Data

Algorithm for Semi Supervised Classifier:

1. Train an Initial Model using the Labeled Data Set and get the initial parameters using MLE or MAP.
2. E Step: Using the above model to compute the expected *(guessed)* labels for all *U* i.e. .
3. M Step: Re-train the model using now + data with the expected *(guessed)* labels.
4. Use the newly trained model to refine the guesses of the unlabeled data set *U*.
5. Repeat until converged.

Semi supervised classifier will perform better as the number of iterations to reach convergence will drastically decrease as we have better initializations of parameters from the labeled data set each time.

But the model we learn from unlabeled data has to be a good model to infer . If the model is not good enough, the unlabeled data will degrade the prediction accuracy by misguiding the inference.

Therefore this approach will be effective only if models are good.

# *References:* [*http://en.wikipedia.org/wiki/Semi-supervised\_learning*](http://en.wikipedia.org/wiki/Semi-supervised_learning)

1. **Rejection Sampling**
2. For my proposal Distribution, I chose a Normal Distribution of and .

I chose as most of the data of the wallaby distribution is located around the mean 5-9 of the Wallaby distribution and peak of the distribution is at . I chose so that it’s wide enough to capture all the data from the 3 distributions. Moreover when it is scaled by *k*, holds. After trying out different values for mean and covariance, this pair gave me the least rejected samples.

Number of Rejected Samples: 8951994

Refer to Code: *u5492350\_ManabChetia\_RejectionSampling.py*

2. Refer to Code: *u5492350\_ManabChetia\_RejectionSampling.py*

3. Sum of Squared Errors:

Refer to Code: *u5492350\_ManabChetia\_RejectionSampling.py*

4. Before even implementing, I was wondering the need for rejection sampling for a distribution like Wallaby when mixing coefficients, means and covariances all are defined.

Instead we can directly take samples 30% of time from , another 30% of time from and finally 40% of time from . In a way, we will get uniformly distributed samples from each of the 3 distributions. Here we don’t lose time rejecting samples. In rejection sampling, we spend 90% of the time rejecting samples.

Times: Rejection Sampling: 2.903 secs, Improved Sampling: 0.0697 secs

Refer to Code: *u5492350\_ManabChetia\_RejectionSampling.py*

5. Refer to Code: *u5492350\_ManabChetia\_RejectionSampling.py*

