COMS W4771 : Machine Learning - Problem Set #3

Dhananjay Shrouty - ds3521@columbia.edu

October 13, 2016

Problem 1

(a) Note : Origin is taken as the point where all the features are zero
Centering - No. Standardization - No. Centering will not affect the learning algorithm as we are just changing the origin point from where we locate our data point from. Changing the origin will not have any effect on where the data point lies from the origin. Standardization will not have an affect because the covariance matrix is identity matrix, so all the σ's are 1. So, if we divide by σ's or not it doesn't matter because we are just dividing and scaling it by 1.

(b) Centering - No. Standardization - Yes. Centering will not affect the 1-NN classifier because we are just changing our origin from one point to another point. The Euclidean distance is based on the distance between the data points, so it doesn't matter from where a data point lies with respect to the origin. Standardization will affect the 1-NN algorithm because 1-NN considers Euclidean distance to be meaningful. If a feature has a big scale compared to another, but the first feature truly represents greater diversity, then features in that dimension should be penalized (ones in which the scale is big).

(c) Centering - No. Standardization - No. The decision trees make decisions based on the facts that a feature is present or not (i.e. it's value is 0 or 1 or a simple yes/no operation). After splitting the tree on a single feature, it then goes on to split the tree based on the second feature and so on. Here the Centering will not affect the learning algorithm as we are just making decisions based on a yes/no answer (whether a value is greater than a particular value) and going deep for further splits on all features. The standardization will also not affect the learning algorithm based on the previous argument. We are just answering a yes/no question to split the tree and thus make decisions. It won't go any faster or slower if the we do centering or standardization

(d) Centering - No. Standardization - Yes The centering will have not have effect because the linear classifier will change the decision boundary and the data points by the same amount. It will still have to calculate the bias and thresholds and

changing where the points lie with respect to origin will not affect how the algorithm works as it is not reducing/increasing any calculations so it won't make the algorithm go any faster or slower. Plus for ERM, it will have to do all the computations for the whole bunch of linear classifiers from which it will choose the best, so centering doesn't affect it. The linear classifiers will be affected by the scale, because if we standardize the scale, the data points may move close to or away from the decision boundary(which is to be determined). Now if the positive and negative points are very close because of scaling there is not much room for wiggling of the decision boundary and it might take time to get an accurate decision boundary if the data points (negative and positive) are close to each other. Their closeness will be affected by scaling and therefore scaling (standardization) affects the ERM.

Problem 2

Note: I have used **one-fifth** of the training examples from the training set to train my classifiers i.e. I have used **200000** examples out of the 1000000 examples which were in the entire training set. I have used **scikit-learn** library of python to vectorize my data. In particular, from the scikit-learn library I have used the functions **CountVectorizer**, **TfidfTransformer** and **BernoulliNB**.

(1) In the fourth training technique I am trying, I am using the sublinear transformation of the TFIDF. The sublinear transformation is a modification which uses the logarithm of the term frequency (instead of the term frequency) in the calculation of TFIDF. It replaces tf(w;d) by wf(w;d) where wf(w;d) is

$$\text{wf}(w;d) = \left\{ \begin{array}{cc} 1 + \log(\text{tf}(w;d)) & \text{if tf}(w,d) > 0 \\ 0 & \text{otherwise} \end{array} \right\}$$

Therefore, the TFIDF looks like

$$\text{TFIDF} = (1 + \log_e(\text{tf}(w;d))) \times (1 + \log_e(\text{idf}(w;d)))$$

(2) The cross validation error rates **(in percentages)** are as below.
Key : {AP = Average Perceptron, NB = Naive Bayes}

Method	Unigram,AP	TFIDF,AP	Bigram,AP	Sublinear,AP	Unigram,NB
Kfold 1	11.4	12.6975	10.0275	12.4775	26.0875
Kfold 2	11.5125	13.13	10.1625	12.6225	25.84
Kfold 3	11.4725	13.0425	9.93	12.4975	25.895
Kfold 4	11.545	12.725	10.02	12.5225	26.085
Kfold 5	11.535	12.875	9.8175	12.42	25.795
Average	11.493	12.894	9.9915	12.508	25.9405

(3) The method ultimately selected using the cross validation procedure is the **Bigram representation** along with the **Averaged Perceptron**

(4) Training Error Rate = 6.6755% ; Test Error Rate = 9.7684%

Problem 3

(a)

$$\mathcal{P} = \left\{ P_{\mu,\sigma^2} : \mu \in \mathbb{R}^d, \sigma^2 > 0 \right\}$$

The probability distribution is given by

$$P = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(\frac{(x_i - \mu)^2}{2\sigma^2} \right)$$

Taking the log of both the sides to get the log likelihood

$$logP = \sum_{i=1}^{n} -\frac{1}{2} log(2\pi\sigma^2) - \sum_{i=1}^{n} \frac{(x_i - \mu)^2}{2\sigma^2}$$

$$= -\frac{n}{2} log(\sigma^2) - \sum_{i=1}^{n} \frac{(x_i - \mu)^2}{2\sigma^2}$$

Taking partial derivative by σ^2 both sides and equating it to zero

$$\frac{\partial logP}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \sum_{i=1}^{n} \frac{(x_i - \mu)^2}{2\sigma^4}$$

Now, equating this to zero yields

$$\sigma^2 = \sum_{i=1}^{n} \frac{(x_i - \mu)^2}{n}$$

This is the MLE for the σ^2