APJ Abdul Kalam Technological

Machine Learning

Assignment 1

S7, CSE
Government Engineering
College, Thrissur

November 24, 2020

1. **Distinguish between classification and regression with an example. (Dec 2018) 4 marks**

   Both regression and classification are supervised learning problems in machine learning where there is an input X, output Y and the task is to learn mapping from input to output.
   Output Y is
   1. a number in regression
   2. a class code in the case of classification.

   Classification: Identification of class to which a data belongs It is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y).

   The output variables are often called labels or categories. The mapping function predicts the class or category for a given observation.

   For example, an email of text can be classified as belonging to one of two classes: "spam" and "not spam".

   - A classification problem requires that examples be classified into one of two or more classes.
   - Classification can have real-valued or discrete input variables.
   - Problem with two classes is often called a two-class or binary classification problem.
   - Problem with more than two classes is often called a multi-class classification problem.
   - Problem where an example is assigned multiple classes is called a multi-label classification problem.

   It is common for classification models to predict a continuous value as the probability of a given example belonging to each output class. The probabilities can be interpreted as the likelihood or confidence of a given example belonging to each class. A predicted probability can be converted into a class value by selecting the class label that has the highest probability.

   A common example of classification comes with detecting spam emails. To write a program to filter out spam emails, a computer programmer can train a machine learning algorithm with a set of spam-like emails labelled as spam and regular emails labelled as not-spam. The idea is to make an algorithm that can learn characteristics of spam emails from this training set so that it can filter out spam emails when it encounters new emails. .

A specific email of text may be assigned the probabilities of 0.1 as being “spam” and 0.9 as being “not spam”. We can convert these probabilities to a class label by selecting the “not spam” label as it has the highest predicted likelihood.

Regression is the task of approximating a mapping function (f) from input variables (X) to a continuous output variable (y).

A continuous output variable is a real-value, such as an integer or floating point value. These are often quantities, such as amounts and sizes.

For example, a house may be predicted to sell for a specific price , perhaps in the range of Rs 100,000 to Rs 200,000 by looking into a whole bunch of exogenous variables such as neighbourhood, location, bathrooms in the house, bedrooms, how far is it from the city .Thus here the input variables are neighbourhood, location, bathrooms in the house, bedrooms, how far is it from the city and the output variable is price of the house. Another example include predicting the number of death cases world wide due to COVID-19 for next 10 days using the help of previous COVID-19 death cases' time series table as training dataset which is used to train the regression model

- A regression problem requires the prediction of a quantity.
- A regression can have real valued or discrete input variables.
- A problem with multiple input variables is often called a multivariate regression problem.
- A regression problem where input variables are ordered by time is called a time series forecasting problem.

A classification algorithm may predict a continuous value, but the continuous value is in the form of a probability for a class label. A regression algorithm may predict a discrete value, but the discrete value in the form of an integer quantity.

2. **Briefly describe the concept of Expectation Maximization algorithm. (Dec 2018) 4 marks**

The expectation-maximization algorithm (EM algorithm) is an approach for performing maximum likelihood estimation in the presence of latent variables. It does this by first estimating the values for the latent variables, then optimizing the model, then repeating these two steps until convergence. It is an effective and general approach and is most commonly used for density estimation with missing data, such as clustering algorithms like the Gaussian Mixture Model.

The maximum likelihood estimation method (MLE) is a method for estimating the parameters of a statistical model, given observations. The method attempts to find the parameter values that maximize the likelihood function, or equivalently the log-likelihood function, given the observations. The expectation-maximisation algorithm is used to find maximum likelihood estimates of the parameters of a statistical model in cases where the equations cannot be solved directly. These models generally involve latent or unobserved variables in addition to unknown parameters and known data observations. For example, a Gaussian mixture model can be described by assuming that each observed data point has a corresponding unobserved data point, or latent variable, specifying the mixture component to which each data point belongs.

Algorithm:

- Given a set of incomplete data, consider a set of starting parameters.
- Expectation step (E – step): Using the observed available data of the dataset, estimate (guess) the values of the missing data.
- Maximization step (M – step): Complete data generated after the expectation (E) step is used in order to update the parameters.
- Repeat step 2 and step 3 until convergence.

In the case of Gaussian mixture problems, because of the nature of the function, finding a maximum likelihood estimate by taking the derivatives of the log-likelihood function with respect to all the parameters and simultaneously solving the resulting equations is nearly impossible. So we apply the EM algorithm to solve the problem.

3. **A patient takes a lab test and the result comes back positive. It is known that the test returns a correct positive result in only 98% of the cases and a correct negative result in only 97% of the cases. Furthermore, only 0.008 of the entire population has this disease.**

   (a) What is the probability that this patient has cancer?

   (b) What is the probability that he does not have cancer?

   (c) What is the diagnosis? (May 2019) 4 marks

From the given data, we can determine the following probabilities

P(cancer) = 0.008
P(¬ cancer) = 1 - 0.008 = 0.992
P(+ve | cancer) = 0.98
P(-ve | cancer) = 1 - 0.98 = 0.02
P(-ve | ¬ cancer) = 0.97
P(+ve | ¬ cancer) = 1 - 0.97 = 0.03

According to Bayes theorem:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

(a) Using Bayes formula:

$$P(cancer| + ve) = \frac{P(+ve|cancer) * P(cancer)}{P(+ve)}$$

$$= \frac{P(+ve|cancer) * P(cancer)}{P(+ve|cancer) * P(cancer) + P(+ve|\neg cancer) * P(\neg cancer)}$$

$$= \frac{0.98 * 0.008}{0.98 * 0.008 + 0.03 * 0.0992}$$

$$= 0.21$$

(b) Using Bayes formula:

$$\begin{aligned} P(\neg cancer \mid +ve) &= \frac{P(+ve|\neg cancer) * P(\neg cancer)}{P(+ve)} \\ &= \frac{P(+ve|\neg cancer) * P(\neg cancer)}{P(+ve|\neg cancer) * P(\neg cancer) + P(+ve|cancer) * P(cancer)} \\ &= \frac{0.03 * 0.992}{0.98 * 0.008 + 0.992 * 0.03} \\ &= 0.79 \end{aligned}$$

(c) From the given data, we can see that every 8 out of 1000 people has cancer. It is also clear that the number of false positive and false negative cases in cancer detection is very low. From (a) it is clear that the probability of having cancer in a positive test result is 21%. From (b) it is clear that the probability of not having cancer in a positive test result is 79%.

4. . **Explain any two model combination scheme to improve the accuracy of a classifier. (Sept 2020) 4 marks**

Ensemble is a machine learning concept in which multiple models are trained using the same learning algorithm. The principle behind the ensemble model is that a group of weak learners come together to form a strong learner, thus increasing the accuracy of the model.

Bagging and Boosting are the ensemble learning methods i.e., the methods for combining the predictions from different models.

Bagging: The term bagging is also known as bootstrap aggregation. In bagging methods, ensemble model tries to improve prediction accuracy and decrease model variance by combining predictions of individual models trained over randomly generated training samples. The final prediction of ensemble model will be given by calculating the average of all predictions from the individual estimators.

One of the best examples of bagging methods are random forests.

Boosting: In boosting method, the main principle of building ensemble model is to build it incrementally by training each base model estimator sequentially. As the name suggests, it basically combine several week base learners, trained sequentially over multiple iterations of training data, to build powerful ensemble. During the training of week base learners, higher weights are assigned to those learners which were misclassified earlier.

An example of boosting method is AdaBoost.

5. **Explain the procedure for the computation of the principal components of the data. (Sept 2020) 4 marks**

Firstly, lets see why we would need to compute the principal components of the data. We use certain procedures to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process. We call the process as Principal Component Analysis or PCA. Thus using PCA, we reduce the number of variables of a data set, while preserving as much information as possible.

Lets look at the steps involved in PCA:

Step 1: Standardization

The aim of this step is to standardize the range of the continuous initial variables so that each one of them contributes equally to the analysis. More specifically, the reason why it is critical to perform standardization prior to PCA, is that the latter is quite sensitive regarding the variances of the initial variables. That is, if there are large differences between the ranges of initial variables, those variables with larger ranges will dominate over those with small ranges (For example, a variable that ranges between 0 and 100 will dominate over a variable that ranges between 0 and 1), which will lead to biased results. So, transforming the data to comparable scales can prevent this problem.

Mathematically, this can be done by subtracting the mean and dividing by the standard deviation for each value of each variable.

$$z = \frac{value - mean}{standard\ deviation}$$

Once the standardization is done, all the variables will be transformed to the same scale.

Step 2: Covariance Matrix computation

The aim of this step is to understand how the variables of the input data set are varying from the mean with respect to each other, or in other words, to see if there is any relationship between them. Because sometimes, variables are highly correlated in such a way that they contain redundant information. So, in order to identify these correlations, we compute the covariance matrix.

The covariance matrix is a p × p symmetric matrix (where p is the number of dimensions) that has as entries the covariances associated with all possible pairs of the initial variables. For example, for a 3-dimensional data set with 3 variables x, y, and z, the covariance matrix is a 3×3 matrix of this form:

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix}$$

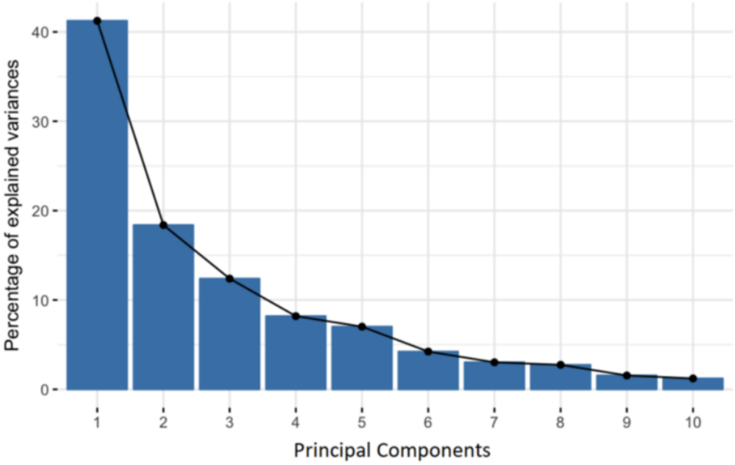
Since the covariance of a variable with itself is its variance, i.e., (Cov(a,a)=Var(a)), in the main diagonal (Top left to bottom right) we actually have the variances of each initial variable. And since the covariance is commutative, i.e., (Cov(a,b)=Cov(b,a)), the entries of the covariance matrix are symmetric with respect to the main diagonal, which means that the upper and the lower triangular portions are equal. The covarience table summarises the correlations between all the possible pairs of variables. If the sign of the covarience is:

- negative, the two variables increase or decrease together(correlated)
- positive, one increases when the other one decreases(inversely correlated)

Step 3: Compute the eigenvectors and eigenvalues of the covariance matrix to identify the principal components

Eigenvectors and eigenvalues are the linear algebra concepts that we need to compute from the covariance matrix in order to determine the principal components of the data. Before getting to the explanation of these concepts, let's first understand what do we mean by principal components.

Principal components are new variables that are constructed as linear combinations or mixtures of the initial variables. These combinations are done in such a way that the new variables (i.e., principal components) are uncorrelated and most of the information within the initial variables is squeezed or compressed into the first components. So, the idea is 10-dimensional data gives you 10 principal components, but PCA tries to put maximum possible information in the first component, then maximum remaining information in the second and so on, until having something like shown in the screen plot below.



Organizing information in principal components this way, will allow you to reduce dimensionality without losing much information, and this by discarding the components with low information and considering the remaining components as your new variables.

An important thing to realize here is that, the principal components are less interpretable and don't have any real meaning since they are constructed as linear combinations of the initial variables.

Geometrically speaking, principal components represent the directions of the data that explain a maximal amount of variance, that is to say, the lines that capture most information of the data. The relationship between variance and information here, is that, the larger the variance carried by a line, the larger the dispersion of the data points along it, and the larger the dispersion along a line, the more the information it has. To put all this simply, just think of principal components as new axes that provide the best angle to see and evaluate the data, so that the differences between the observations are better visible.

Step 4: Feature Vector

Computing the eigenvectors and ordering them by their eigenvalues in descending order allow us to find the principal components in order of significance. In this step, what we do is, to choose whether to keep all these components or discard those of lesser significance (of low eigenvalues), and form with the remaining ones a matrix of vectors that we call Feature vector.

So, the feature vector is simply a matrix that has as columns the eigenvectors of the components that we decide to keep. This makes it the first step towards dimensionality reduction, because if we choose to keep only p eigenvectors (components) out of n, the final data set will have only p dimensions. Thus, it's up to you to choose whether to keep all the components or discard the ones of lesser significance, depending on what you are looking for. Because if you just want to describe your data in terms of new variables (principal components) that are uncorrelated without seeking to reduce dimensionality, leaving out lesser significant components is not needed.

Step 5: Recast the Data Along the Principal Components Axes

In the previous steps, apart from standardization, you do not make any changes on the data, you just select the principal components and form the feature vector, but the input data set remains always in terms of the original axes (i.e, in terms of the initial variables).

In this step, which is the last one, the aim is to use the feature vector formed using the eigenvectors of the covariance matrix, to reorient the data from the original axes to the ones represented by the principal components (hence the name Principal Components Analysis). This can be done by multiplying the transpose of the original data set by the transpose of the feature vector.

$$FinalDataSet = FeatureVector^T * StandardizedOriginalDataset^T$$