



Practical Considerations for Deploying Naive Bayes Models

Let us now look at some practical considerations in applying NB

Assumption of Normality

- Recall that Naïve Bayes model assumes that all features are independent given the class label Y . That is to say

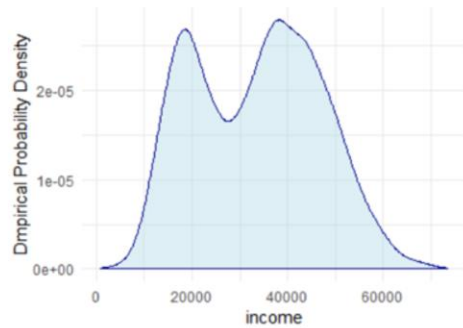
$$P(X_1, \dots, X_n|Y) = \prod_{i=1}^n P(X_i|Y)$$

- For categorical variables, this calculation is very straightforward.
- However, when dealing with numerical variables, it is often assumed that data follows a Gaussian distribution, so that to make it easy to calculate probabilities.
- This assumption, however, may not hold true all the time.

Generally, the NB model is only applied when all predictor variables are categorical. This allows us to calculate the conditional probabilities easily. We could also calculate the conditional probabilities for numeric predictors if we make some assumptions on their distributional forms. One such form is to assume that the distribution is Gaussian (Normal).

Example

- Let us consider again the credit card default example.
- The distribution of income does not follow a bell shape Gaussian distribution.
- Instead, we can see a bi-modal distribution with two peaks.



For this example, it is clear that the distribution of data is not normal. We could transform the data to make it Normal. This is similar to how in regression we transform variables to satisfy the normality assumption.

Box-Cox Transformation

- Box-Cox transformation can be applied to make a non-normal distribution to look more like a normal distribution.
- The Box-Cox transformation of the variable x is also indexed by λ , and is defined as

$$x'_{\lambda} = \frac{x^{\lambda} - 1}{\lambda}.$$

- The `preProcess()` function in 'caret' package can be used to apply the Box-Cox transformation by finding the optimal value of λ

The box-cox approach allows us to determine the transformation that is needed to get non-normal data to look more normal. This transformation is part of the “caret” package.

Example II

```
library(ISLR)
library(caret)
#Create a Box-Cox Transformation Model

Box_Cox_Transform<-preProcess(Default,method = "BoxCox")
Box_Cox_Transform

## Created from 10000 samples and 3 variables
##
## Pre-processing:
##   - Box-Cox transformation (1)
##   - ignored (2)
##
## Lambda estimates for Box-Cox transformation:
## 0.7
#Apply the model

Default_Transformed=predict(Box_Cox_Transform,Default)
```

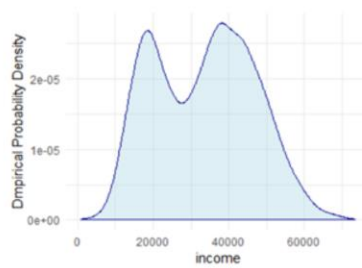
Optimal Lambda

Transformation is applied only to one variable, income

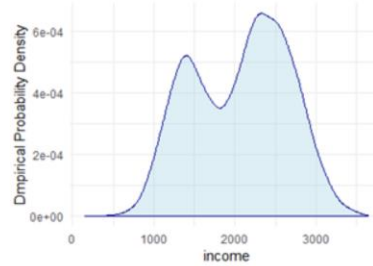
The Box-Cox transformation here suggests a value of 0.7 for lambda, where the transformation is applied to the income variable. In other words, we transform income to $(\text{income}^{0.7} - 1) / 0.7$.

Box-Cox Transformation Distribution

- Box-Cox transformation, however, may not be sufficient to make a variable normally distributed.



Income variable, original



Income variable, box-cox transformed. Slightly improved but still far from a Normal distribution

Notice that while the distribution looks more normal, it is still a bi-modal distribution. In such case, stronger measures are needed to satisfy assumptions.

Kernel Estimators and Laplace Smoother

- We can also use non-parametric kernel density estimators to try get a more accurate representation of continuous variable probabilities.
- Moreover, since naïve Bayes uses the product of feature probabilities conditioned on each class, we run into a serious problem when new data includes a feature value that never occurs for one or more levels of a response class.
- The conditional probability for this feature will be zero and will result the product of all probabilities to be zero.
- A solution to this problem involves using the **Laplace smoother**. The Laplace smoother adds a small number to each of the counts in the frequencies for each feature to avoid this this issue.

There are three main difficulties in applying the NB model.

First, the naive Bayes classifier requires a very large number of records to obtain good results.

Second, where a predictor category is not present in the training data, naive Bayes assumes that a new record with that category of the predictor has zero probability. This can be a problem if this rare predictor value is important. A popular solution in such cases is to replace zero probabilities with non-zero values using a method called smoothing (e.g., Laplace smoothing can be applied by using argument `laplace = 0` in function `naiveBayes()`).

Finally, good performance is obtained when the goal is classification or ranking of records according to their probability of belonging to a certain class. However, when the goal is to estimate the probability of class membership (propensity), this method provides very biased results. For this reason, the naive Bayes method is rarely used in credit scoring (Larsen, 2005).

Tuning a Naive Bayes Model

- We can use caret package to tune a Naive Bayes Model with respect to the discussed considerations. The following hyperparameters are used:
- **usekernel** allows the model to use a kernel density estimate for continuous variables versus a Gaussian density estimate.
- **adjust** allows the model to adjust the bandwidth of the kernel density (larger numbers mean more flexible density estimate),
- **fL** allows us to incorporate the Laplace smoother.

Hypertuning is always an important aspect of improving model performance. The NB model is no different. This slide provides some hypertuning parameters.

Example normalizaiton

```
library(caret)
library(ISLR)

#remove student status, which is the second variable
MyData<-Default[,-2]
set.seed(123)

#Divide data into test and train
Index_Train<-createDataPartition(MyData$default, p=0.8, list=FALSE)
Train<-MyData[Index_Train,]
Test <-MyData[-Index_Train,]

# Build a naïve Bayes classifier
nb_model <-train(default~balance+income,data = Train, preProc = c("BoxCox", "center", "scale"))

# Predict the default status of test dataset
Predicted_Test_labels <-predict(nb_model,Test)

library("gmodels")

# Show the confusion matrix of the classifier
CrossTable(x=Test$default,y=Predicted_Test_labels, prop.chisq = FALSE)
```

Here, we both normalize the data, and also use the Box-Cox transformation.

Example misclassification

Cell Contents	
	N
N / Row Total	
N / Col Total	
N / Table Total	

Total Observations in Table: 1999

Test\$default	Predicted_Test_Labels		Row Total
	No	Yes	
No	1928 0.997 0.977 0.964	5 0.003 0.200 0.003	1933 0.967
Yes	46 0.697 0.023 0.023	20 0.303 0.800 0.010	66 0.033
Column Total	1974 0.987	25 0.013	1999

False Positive

A total of 51
Misclassification Cases

False Negative

This produces a slight improvement in misclassification rates.

This concludes our presentation for the NB model.