

# Data Mining and Machine learning

## Naïve Bayes classifier

Edgar Acuna

Department of Mathematical Science

[academic.uprm.edu/eacuna](http://academic.uprm.edu/eacuna)

# Building a Bayesian Classifier

- Assume you want to predict output  $Y$  which assumes  $n_Y$  different values  $v_1, v_2, \dots, v_{n_Y}$ .
- Assume there are  $m$  input attributes called  $X_1, X_2, \dots, X_m$
- Break dataset into  $n_Y$  smaller datasets called  $DS_1, DS_2, \dots, DS_{n_Y}$ .
- Define  $DS_i$  = Records in which  $Y=v_i$
- For each  $DS_i$ , learn Density Estimator  $M_i$  to model the input distribution among the  $Y=v_i$  records. There are several approaches but the easier is the Naïve Density estimator.
- $M_i$  estimates  $P(X_1, X_2, \dots, X_m \mid Y=v_i)$
- Estimate  $P(Y=v_i)$ . as fraction of records with  $Y=v_i$ .
- For a new prediction:

$$\begin{aligned} Y^{\text{predict}} &= \underset{v}{\operatorname{argmax}} P(Y = v \mid X_1 = u_1 \cdots X_m = u_m) \\ &= \underset{v}{\operatorname{argmax}} P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v) \end{aligned}$$

# Naïve Bayes Classifier

In the case of the naive Bayes Classifier, the random variables  $X_1, \dots, X_m$  are considered independent, hence the previous equation can be simplified as:

$$y^{\text{predict}} = \underset{v}{\operatorname{argmax}} P(Y = v) \prod_{j=1}^m P(X_j = u_j | Y = v)$$

# Naïve Bayes Classifier

In the case of the naive Bayes Classifier, the random variables  $X_1, \dots, X_m$  are considered independent, hence the previous equation can be simplified as:

$$y^{\text{predict}} = \underset{v}{\operatorname{argmax}} P(Y = v) \prod_{j=1}^m P(X_j = u_j | Y = v)$$



If you have a lot of input attributes **that** product will underflow in floating point math. You should use logs:

$$y^{\text{predict}} = \underset{v}{\operatorname{argmax}} \left( \log P(Y = v) + \sum_{j=1}^m \log P(X_j = u_j | Y = v) \right)$$

# Naïve Bayes classifier

Naïve Bayes classifiers can be applied when there are continuous attributes in the dataset. There are two options:

- a) Discretize the continuous attributes using methods such as, Equal width discretization, Equal frequency discretization and, Entropy with MDLP discretization. After that, apply NB on the discretized data. Scikit learn has two function BernoulliNB ( it assumes that all the predictors are binary) and MultinomialNB (it assumes that the predictors can assume more than two different values).
- b) Assume a Gaussian distribution for the continuous predictors Scikit learn has a GaussianNB function.

The option a) is preferred by many.

# Example

| X1 | X2 | X3 | Y |
|----|----|----|---|
| 0  | 0  | 1  | 0 |
| 0  | 1  | 0  | 0 |
| 1  | 1  | 0  | 0 |
| 0  | 0  | 1  | 1 |
| 1  | 1  | 1  | 1 |
| 0  | 0  | 1  | 1 |
| 1  | 1  | 0  | 1 |

# Example: Cont

To which class will be assigned the record (0,0,1)?

$$P(Y = 0) = 3/7$$

$$P(Y = 1) = 4/7$$

$$\begin{aligned} P(X_1 = 0, X_2 = 0, X_3 = 1 / Y = 0) &= P(X_1 = 0 / Y = 0) P(X_2 = 0 / Y = 0) P(X_3 = 1 / Y = 0) = \\ &= (2/3)(1/3)(1/3) = 2/27 \end{aligned}$$

$$\begin{aligned} P(X_1 = 0, X_2 = 0, X_3 = 1 / Y = 1) &= P(X_1 = 0 / Y = 1) P(X_2 = 0 / Y = 1) P(X_3 = 1 / Y = 1) = \\ &= (2/4)(2/4)(3/4) = 3/16 \end{aligned}$$

X1=0, X2=0, x3=1 will be assigned to class 1

## Example 2. (continuous and discrete attributes)

| X1 | X2 | X3 | X4    | Y |
|----|----|----|-------|---|
| 0  | 0  | 1  | 3.15  | 0 |
| 0  | 1  | 0  | 8.17  | 0 |
| 1  | 1  | 0  | 5.72  | 0 |
| 0  | 0  | 1  | 7.16  | 1 |
| 1  | 1  | 1  | 9.32  | 1 |
| 0  | 0  | 1  | 12.81 | 1 |
| 1  | 1  | 0  | 15.48 | 1 |



# Naïve Bayes for Diabetes

Without Discretization

```
# Calculo de las probabilidades posteriores y de las clases predichas
clf = GaussianNB()
clf.fit(X1,y1)
proba=pd.DataFrame(clf.predict_proba(X1))
pred=clf.predict(X1)
print 'Accuracy by Resubstitution',(pred==y1).sum()/float(768)
Accuracy by Resubstitution 0.76171875
# Calculo de las probabilidades posteriores y de las clases predichas
```

With Discretization; Equal width

```
from sklearn.preprocessing import KBinsDiscretizer
est = KBinsDiscretizer(n_bins=10, encode='ordinal', strategy='uniform')
est.fit(X)
Xt = est.transform(X)
clf=MultinomialNB()
clf.fit(Xt,y1)
scores = cross_val_score(clf, X1, y1, cv=10)
print("Accuracy by cross validation: %0.3f (+/- %0.3f)" % (scores.mean(), scores.std()))
Accuracy by cross validation: 0.654 (+/- 0.048)
```

# The Auto-mpg dataset

Donor: Quinlan, R. (1993)

Number of Instances: 398 minus 6 missing=392 (training: 196 test: 196):

Number of Attributes: 9 including the class attribute 7. Attribute Information:

1. mpg: continuous (discretizado  $\text{bad} \leq 25, \text{good} > 25$ )
2. cylinders: multi-valued discrete
3. displacement: continuous (discretizado  $\text{low} \leq 200, \text{high} > 200$ )
4. horsepower: continuous ((discretizado  $\text{low} \leq 90, \text{high} > 90$ )
5. weight: continuous (discretizado  $\text{low} \leq 3000, \text{high} > 3000$ )
6. acceleration: continuous (discretizado  $\text{low} \leq 15, \text{high} > 15$ )
7. model year: multi-valued discrete (discretizado 70-74, 75-77, 78-82)
8. origin: multi-valued discrete
9. car name: string (unique for each instance)

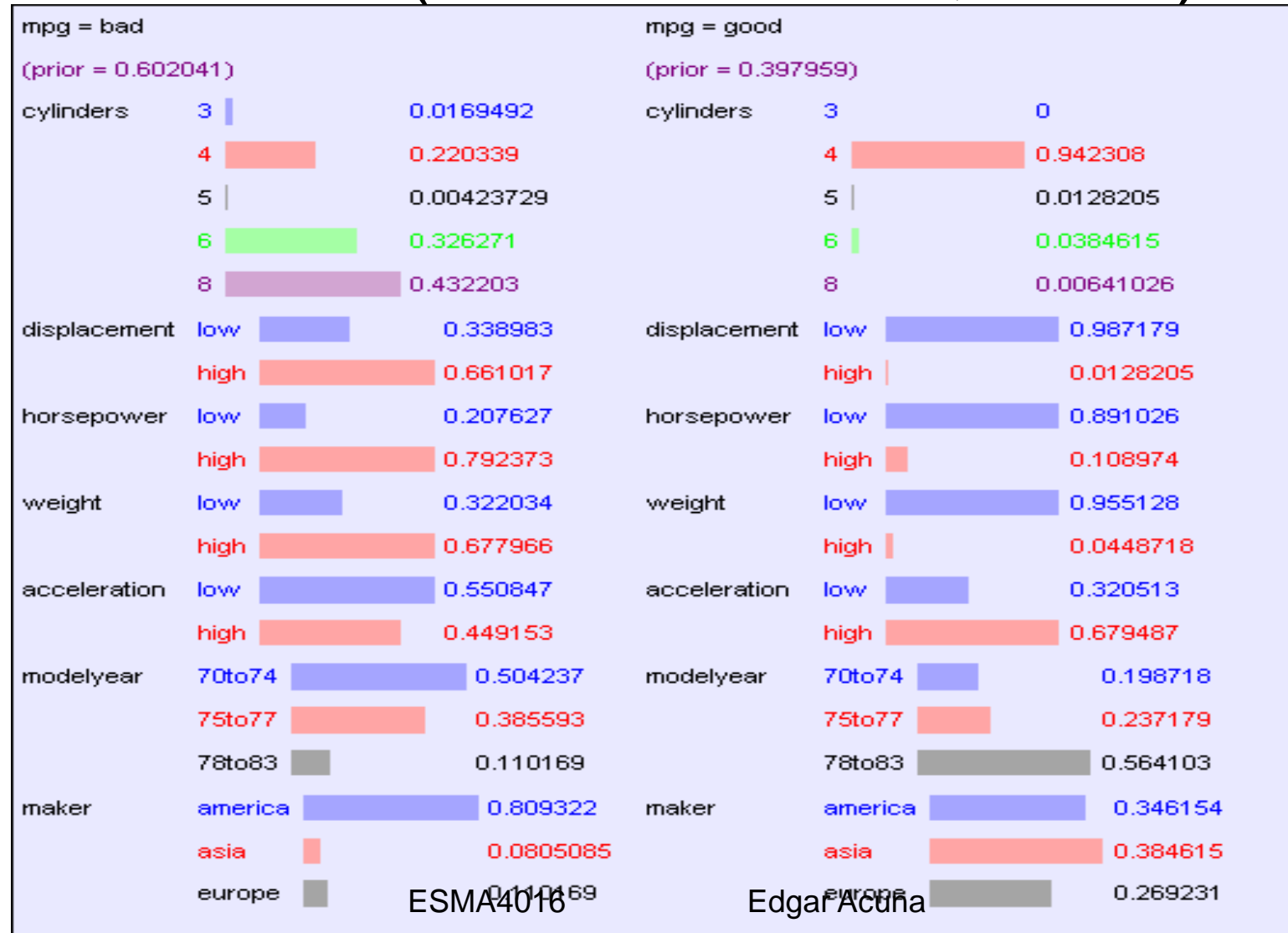
Note: horsepower has 6 missing values

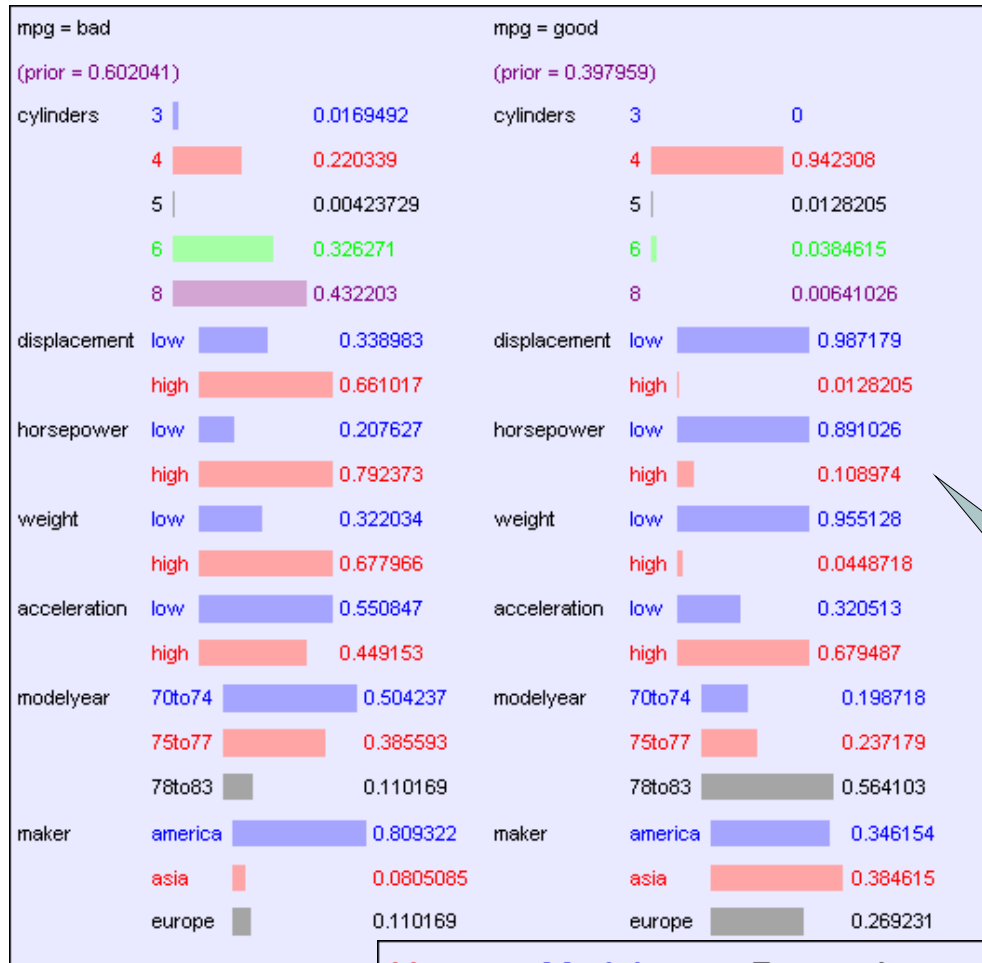
Available at: <https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data>

# The auto-mpg dataset

```
18.0 8 307.0 130.0 3504. 12.0 70 1 "chevrolet chevelle malibu"  
15.0 8 350.0 165.0 3693. 11.5 70 1 "buick skylark 320"  
18.0 8 318.0 150.0 3436. 11.0 70 1 "plymouth satellite"  
16.0 8 304.0 150.0 3433. 12.0 70 1 "amc rebel sst"  
17.0 8 302.0 140.0 3449. 10.5 70 1 "ford torino"  
  
.....  
27.0 4 140.0 86.00 2790. 15.6 82 1 "ford mustang gl"  
44.0 4 97.00 52.00 2130. 24.6 82 2 "vw pickup"  
32.0 4 135.0 84.00 2295. 11.6 82 1 "dodge rampage"  
28.0 4 120.0 79.00 2625. 18.6 82 1 "ford ranger"  
31.0 4 119.0 82.00 2720. 19.4 82 1 "chevy s-10"
```

# Resultados del clasificador NB para “MPG”: 392 records (Prof. A. Moore, CMU)





# BC Results:

## “MPG”: 392 records

The Classifier learned by “Naive BC”

| Name   | Model      | Parameters   | FracRight              |  |
|--------|------------|--|------------------------|--|
| Model1 | bayesclass | density=joint<br>submodel=gauss<br>gausstype=general | 0.885256 +/- 0.0247796 |  |
| Model2 | bayesclass | density=naive<br>submodel=gauss<br>gausstype=general | 0.852372 +/- 0.0400495 |  |

# Advantages/Disadvantages of NB

Probabilities zeros can affect the NB classifier. A correction factor is applied to the computation of probabilities to solve this problem.

- The discretization process seems to affect the performance of the classifier.
- Naïve Bayes is very cheap. It does not have problem working with 10,000 attributes.
- Naïve Bayes is a particular case of Bayesian networks