Bayes Classifier

Author: George Barrinuevo

November 2025

1 Introduction

1.1 Purpose

1.2 Classes

The number of classes and their corresponding labels must be selected. An example of this is the 5-classes. These are the list of regimes to detect. See Table 1.

Table 1: Placeholder Caption

Class Label
Strong Uptrend
Weak Uptrend
Sideways
Weak Downtrend
Strong Downtrend
————
mean-reverting
high-vol

These are the various regimes the algorithm needs to identify.

1.3 Features

Features are the inputs to the model. They can be numerical or categorical and they encode information about the data. The data can be the daily closing prices of a stock symbol. Here is a partial list of possible features. See Table 2.

Table 2: Placeholder Caption

Feature
mean
variance
return
lognormal of return
slope
MA, RSI, MACD
volatility
distance from MA
cumulative log-return
regression slope
MA slope
volatility-adjusted returns
log-return momentum
10-day moving average slope

For this implementation, we will use these features.

- Slope

  positive, near 0, negative
- Lognormal of returns

  negative, flat, positive
- Volatility

  ???
- Distance from MA

  $> 0, near_0, < 0$

Calculate Log-Returns

From historical data, compute log-returns:

$$r_t = \ln \frac{P_t}{P_{t-1}}$$

Over a bunch of bars $t = 1, \ldots, N$, compute:

$$\mu = \frac{1}{N} \sum_{t=1}^{N} r_t$$

$$\sigma = \sqrt{\frac{1}{N-1} \sum_{t=1}^{N} (r_t - \mu)^2}$$

- $\mu \approx$ average log-return per bar (usually small, near 0).
- $\sigma =$ volatility (std dev) of log-return per bar.

Pick a multiplier k that controls how "strict" you want to be:

- $k \approx 0.5$: more bars labeled positive/negative, fewer as "near 0"
- $k \approx 1$: only bigger moves are positive/negative, more "near 0"

Set:

$$\theta = k\sigma$$

Then your decision rule is just:

negative if $r < \mu - \theta$
near 0 if $\mu - \theta \leq r \leq \mu + \theta$
positive if $r > \mu + \theta$

Calculate Slope

Compute log-returns

Given prices P_t:

$$r_t = \ln \left(\frac{P_t}{P_{t-1}} \right)$$

Now you have a series of log-returns:

$$r_1, r_2, \ldots, r_T$$

Compute the slope of log-price (or cumulative log-returns)

The easiest and most meaningful slope is the linear regression slope of log-price vs. time.

Since:

$$\ln(P_t) = \ln(P_0) + \sum_{i=1}^{t} r_i$$

define:

$$y_t = \ln(P_t)$$

Then compute the regression slope:

$$\text{slope} = \frac{\sum_{t=1}^{T}(t-\bar{t})(y_t-\bar{y})}{\sum_{t=1}^{T}(t-\bar{t})^2}$$

Where:

$$\bar{t} = \frac{1}{T}\sum_{t=1}^{T} t, \quad \bar{y} = \frac{1}{T}\sum_{t=1}^{T} y_t$$

This slope represents average log-return per bar over the window. There is a simpler equivalent formula, but this regression version will reduce unwanted noise; use the simple difference if you want speed.

Estimate typical slope volatility

To classify slopes, you need the typical "noise size."

Collect slopes from many historical windows:

$$s_1, s_2, \ldots, s_N$$

Compute:

$$\mu_s = \text{mean of slopes}$$

$$\sigma_s = \text{std dev of slopes}$$

Usually $\mu_s \approx 0$, so classification becomes symmetric.

Choose a threshold using $k\sigma$

Pick a sensitivity constant:

- $k = 0.5$: more sensitive
- $k = 1.0$: noise-filtered, most common
- $k = 1.5$: very strict

Set:

$$\theta = k\sigma_s$$

Classify the slope

If μ_s is small (almost always true), use:

negative if $s < -\theta$

near 0 if $|s| \le \theta$

positive if $s > \theta$

If μ_s is not negligible (rare):

negative if $s < \mu_s - \theta$

near 0 if $\mu_s - \theta \le s \le \mu_s + \theta$

positive if $s > \mu_s + \theta$

1.4 Training - Find Class Labels

The data must be trained using training data before using it on test data. This involves using another method to label each data point with the categorical class labels. One of the most quant-finance friendly methods to do this is the Log-Return Momentum Method. This method works better for volatility as compared with raw returns and works cleanly with Gaussian likelihoods.

The rolling cumulative log-return is computed:

$$R_t = \sum_{i=t-n+1}^{t} \ln\left(\frac{P_i}{P_{i-1}}\right)$$

Formula 1

Let:

- Weak threshold = α_1
- Strong threshold = α_2

Typical thresholds:

- $\alpha_1 = 0.3\%$
- $\alpha_2 = 1.0\%$

Here is the class definitions:

- Strong Uptrend
  $R_t > \alpha_2$
- Weak Uptrend
  $\alpha_1 < R_t \leq \alpha_2$
- Sideways
  $|R_t| \leq \alpha_1$
- Weak Downtrend
  $-\alpha_2 \leq R_t < -\alpha_1$
- Strong Downtrend
  $R_t < -\alpha_2$

1.5 Training - Find Gaussian Likelihood

Calculate the Prior Gaussian Parameters for mean μ and standard deviation σ. This is done on a per feature i and per class k. So, if you have 2 features and 5 classes, then you have 2 * 5 = 10 sets of μ and σ.

For each class, compute mean and variance

Use only feature i from x_i data and with in class k.

- A mean for each feature
  $$\mu_{j,c} = \text{average value of feature } j \text{ inside class } c$$
- A variance for each feature
  $$\sigma^2_{j,c} = \text{variance of feature } j \text{ inside class } c$$

Select one feature j and one class c. Using all training data in class c, calculate the Gaussian Prior:

$$\mu_{jc} = \frac{1}{N_c} \sum_{i=0}^{N_c} x_{ij}$$

$$\sigma_{jc}^2 = \frac{1}{N_c - 1}\sum_{i=0}^{N_c}(x_{ij} - \mu_{jc})^2$$

Where:

i = i-th data
j = j-th features
c = c-th class
where N_c is the number of data in class c
x_{ij} is the j-th feature of the i-th sample.

The Gaussian density is:

$$\mathcal{N}(x_j \mid \mu_{jc}, \sigma_{jc}^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Calculate the Gaussian Posterior for all features j, given one class c:

$$P(\mathbf{x} \mid y = c) = \prod_{j=1}^{d} P(x_j \mid y = c) = \prod_{j=1}^{d} \mathcal{N}(x_j \mid \mu_{jc}, \sigma_{jc}^2)$$

1.6 Testing - Find Gaussian Prior

The prior is simply the probability of each class before seeing any features. This method is called the 'Empirical Prior' and is the most common. It is estimated directly from the training data. Most implementations (e.g., scikit-learn GaussianNB with prior=None) use this by default.

$$P(y = c) = \frac{\text{number of training samples with class } c}{\text{total number of samples}}$$

Calculate this for each class.

1.7 Testing - Find Gaussian Score

The Score is calculated by multiplying the Likelihood and the Prior:

Likelihood:

$$P(\mathbf{x} \mid y = c)$$

Prior:

$$P(y = c)$$

Calculate Score:

$$P(y = c \mid \mathbf{x}) \propto P(y = c)P(\mathbf{x} \mid y = c)$$

$$P(y = c \mid \mathbf{x}) \propto P(y = c)\prod_{j=1}^{d} \mathcal{N}(x_j \mid \mu_{jc}, \sigma_{jc}^2)$$

In practice you use log-probabilities to avoid underflow:

$$\text{LogScore}(c) = \log P(y = c) + \sum_{j=1}^{d} \log \mathcal{N}(x_j \mid \mu_{jc}, \sigma_{jc}^2)$$

$$\log \mathcal{N}(x_j \mid \mu_{jc}, \sigma_{jc}^2) = -\frac{1}{2}\log(2\pi\sigma_{jc}^2) - \frac{(x_j - \mu_{jc})^2}{2\sigma_{jc}^2}$$

$$\text{LogScore}(c) = \log P(y = c) + \sum_{j=1}^{d}\left[-\frac{1}{2}\log(2\pi\sigma_{jc}^2) - \frac{(x_j - \mu_{jc})^2}{2\sigma_{jc}^2}\right]$$

Then:

1. Compute LogScore(Strong-Uptrend), LogScore(Weak-Uptrend), etc.

2. Predict the class with the higher log score:

$$\hat{y} = \arg \max_{c \in \{\text{Strong-Uptrend}, \text{Weak-Uptrend}, \ldots\}} \text{LogScore}(c).$$

If you want the actual posterior probabilities:

$$P(y = c \mid \mathbf{x}) = \frac{\exp(\text{LogScore}(c))}{\sum_{c'} \exp(\text{LogScore}(c'))}.$$

Where:
c': All classes

Then, select the class with the largest posterior probabilities. Next, appply a threshold.

1.8 Testing - Evalute using Threshold

Take the highest LogScore() and compare it with threshold values. Thresholds prevent whipsaws, false regime flips, micro noise triggering regime changes. They ensure classification is stable. Here are the recommended thresholds for Trading Regimes, see Table 3.

Purpose	**Threshold**
Fast switching / reactive	0.50 – 0.55
Balanced trading	0.60 – 0.70
Stable regime detection	0.70 – 0.80
Very slow regime switching	> 0.80
Uncertain	else

Table 3: Threshold Levels

Most quants settle around 0.65 – 0.75.