# BỨC TRANH LỚN VỀ CÁC MÔ HÌNH

Table here.

# ĐẶC TRƯNG CỦA CÁC MÔ HÌNH

## SVR + GA

### Đặc trưng

Ở đây, sử dụng hàm Kernel là Gaussian với hàm tổn thất là . Tác giả có 2 cải tiến như sau:

* *Dùng GA để chọn ra bộ tham số tốt nhất (C, , ) cho mô hình SVR*. Thông thường, để chọn ra bộ tham số cho SVR người ta cho các tham số tăng dần theo số mũ và chọn ra bộ tốt nhất (dựa trên cross-validation). Nhận thấy ngay với cách làm này miền các giá trị được chọn là miền rời rạc. Với GA, miền được chọn liên tục và do đó cho phép cải tiến độ tốt của bộ tham số.
* *Nhận ra được tầm quan trọng của số node đầu vào của mô hình SVR.* Tuy nhiên, ở đây vẫn dùng phương pháp thử và sai (cho số node chạy từ 1-5.) Việc tìm cách chọn ra số node đầu vào một cách bài bản được tác giả định hướng là công việc tương lai.

Công thức tiền xử lý của tác giả để scale về (0, 1) của tác giả cũng khá lạ:

Trong đó:

: giá đóng cửa tại thời điểm t

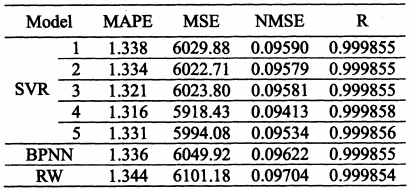
: giá đóng cửa cao nhất

: giá đóng cửa thấp nhất

(Lý lẽ của tác giả là “This scaling for original data points helps to improve the forecasting accuracy [3]” với [3] C. C. Chang and C. J. Lin, LIBSVM:A library for support vector machines, Retrieved May 20, 2004, from National Taiwan University, Department of Computer Science and Information Engineering Website: http://www.csie.edu.tw/~cjlin/papers/libsvm.pdf, 2001.)

### Kết quả

* Ở đây, ta so sánh giữa SVR, BPNN (back propagation NN), RW trên 3 độ đo lỗi: NMSE (Normalized Mean Square Error), MSE (Mean Square Error), MAPE (Mean Absolute Percentage Error.)
* Nhìn chung, cả 3 đều cho kết quả tốt, trong đó SVR tốt hơn 2 cái còn lại. Hệ số tỉ lệ giữa giá dự đoán và giá thực của cả 3 đều gần = 1, trong đó SVR là gần 1 nhất.
* Về khảo sát số node đầu vào: khi ta tăng số node đầu vào thì độ đo lỗi MAPE giảm nhưng khi số node này quá 4 thì MAPE lại tăng. Do đó, số node đầu vào = 4 được đánh giá là tối ưu.



## FUZZY SVR

Bằng cách tiền xử lý bằng 10 days-EMS và sử dụng hàm fuzzy để đặt giá trị liên thuộc cho mỗi điểm dữ liệu trong chuỗi time-series, dụng ý của tác giả là *đặt trọng lên dữ liệu gần*, giảm ảnh hưởng của nhiễu.

Hàm liên thuộc: được xác định đơn giản như sau:

* Chọn L > 0 là cận dưới của hàm liên thuộc.
* Ta có hàm liên thuộc theo thời gian: si = f(ti)
  + Ta có: sn = f(tn) = 1 và s1 = f(t1) = L với n là tổng số điểm dữ liệu training.
  + Tùy vào ta muốn hàm liên thuộc là tuyến tính (si = ati + b) hay bậc 2 (si = a(ti - b)2 + c), ta sẽ dễ dàng tìm được công thức cho hàm liên thuộc.

# KẾT LUẬN