Seasonal Time Series Forecasting :A Comparative Study of ARIMA, RNN and SVR Models.

Time Series Forecasting using SVR in R

Hassan OUKHOUYA

Framing: M^r Khalid EL HIMDI

Mohammed V University
Faculty of Science - Rabat
Laboratory of Mathematics, Statistics and Applications
Master of Statistics and Econometrics

June 28, 2020

Plan Of Attack

Time Series generally

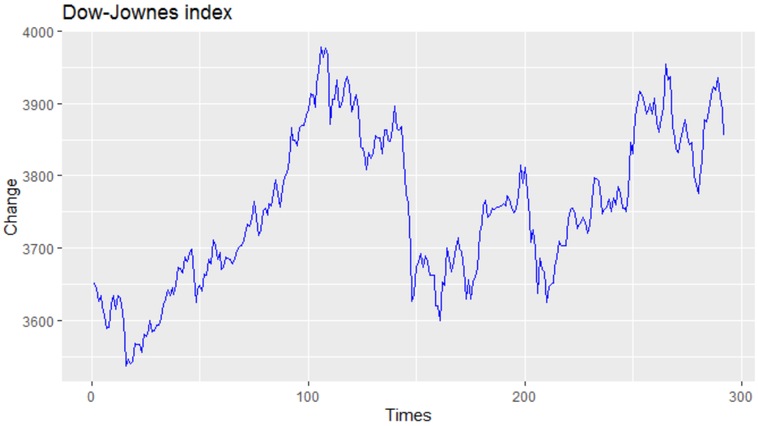
Time Series consist of sequences of observations collected over time. The sequence of random variables $\{Y_t : t = 0, \pm 1, \pm 2, \ldots\}$ is called a **stochastic process** and serves as a model for an observed time series.

For now, we will focus essentially on the **Univariate time series** as they are often needed in:

- Business.
- Meteorology.
- Agriculture.
- Biological sciences.
- Ecology.

Financial Time Series

Dow-Jones index on 251 trading days ending 26 Aug 1994.



Source: Data sets from "Forecasting: methods and applications" by Makridakis, Wheelwright and Hyndman (Wiley, 3rd ed., 1998).

What Is Machine Learning?

In Psychology

Acquisition of a new behavior after training: habituation, conditioning ...

In Neurobiology

Synaptic modifications in neural circuits: rule of Hebb, rule of Rescorla and Wagner ...

Machine Learning

It is the process of building a general model from particular data (observations) in the real world.
So the goal is twofold:

- **Predict** behavior with new data.
- **Approximating** a function or a probability density.

Applications of ML

Applications of ML

Applications of ML

Applications of ML

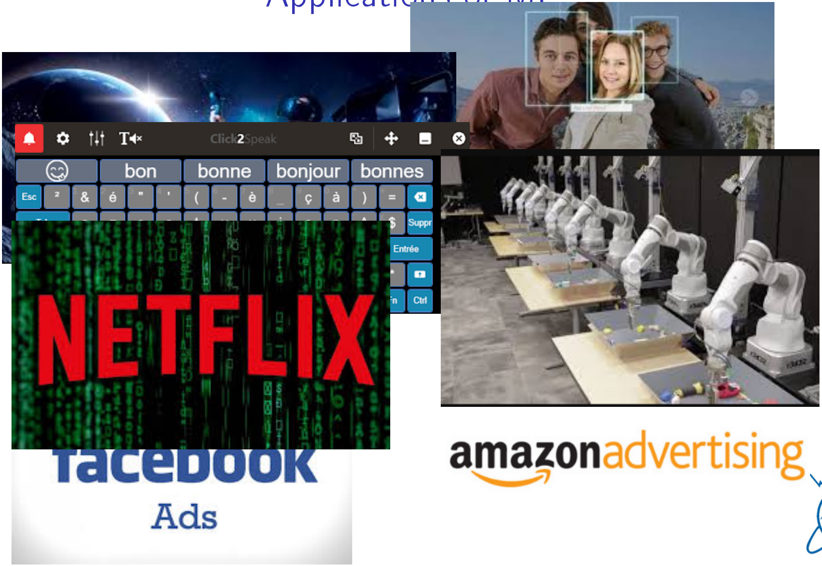
Applications of ML

Applications of ML

facebook Ads

amazonadvertising

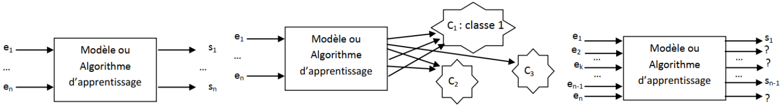
Applications of ML



Applications of ML

Types of Machine Learning

- Supervised learning.
- Unsupervised learning.
- Semi supervised learning.
- Reinforcement Learning.



$\mathcal{N}$**ote:** In our case, we will focus on the type of supervised learning.

History of SVMs

Hype or Hallelujah? Is the provocative title used by Bennett & Campbell (2000) in an overview of Support Vector Machines (SVM).

The initial of SVMs developed by Vapnik and colleagues from AT & T Bell laboratories in 1995.

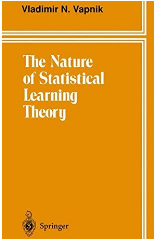


Figure: Vladimir Naumovitch Vapnik, The Nature of Stat. Lea. The.

Basic concept

- Class separation.
- Overlapping classes.
- Nonlinearity.
- Problem solution.

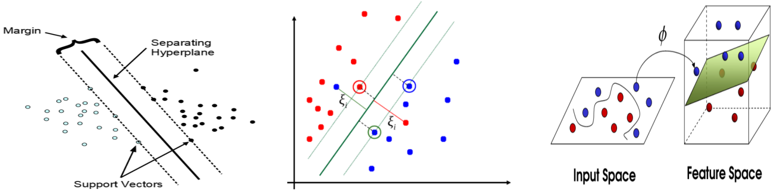


Figure: Linear separable and overlapping classes and nonlinearity.

The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences.

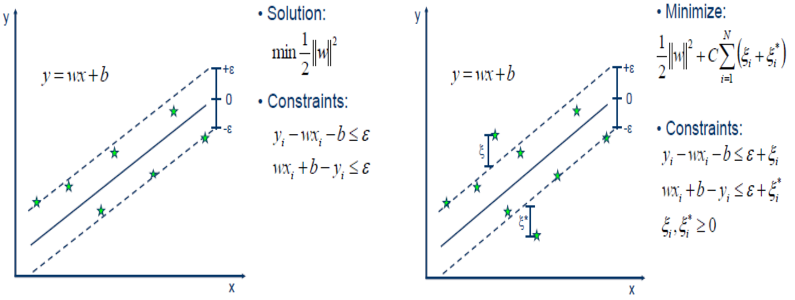


Figure: linear SVR separable and non separable

Linear SVR:

$$y = f(x) = \sum_{i=1}^{N_{SV}} (\alpha_i^* - \alpha_i) x_i . x + b = \sum_{i=1}^{N_{SV}} (\alpha_i^* - \alpha_i) < x_i, x > + b, \ \forall \alpha_i, \alpha_i^* \in [0, C].$$

Non-Linear SVR: The kernel functions transform the data into a higher dimensional feature space to make it possible to perform the linear separation.

$$y = f(x) = \sum_{i=1}^{N_{SV}} (\alpha_i^* - \alpha_i) k(x_i, x) + b, \quad With\ k(x_i, x) = \phi(x_i) \cdot \phi(x)$$

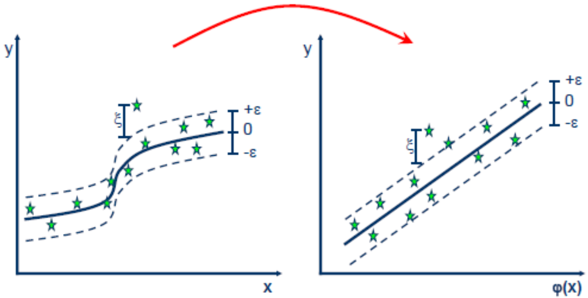


Figure: Non linear SVR

Training and test sets

Training set (e.g.,80%)	Test set (e.g.,20%)

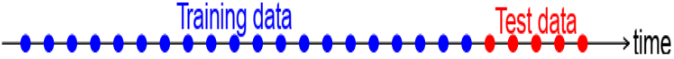


Figure: Traditional evaluation

- The goal of split time series to minimize errors based on training.
- The test set must not be used for any aspect of model development or calculation of forecasts.
- Forecast accuracy is based only on the test set.

The Interface to `libsvm` in package `e1071`

The package `e1071` offers an interface to the award-winning C++ implementation by *Chih-Chung Chang* and *Chih-Jen Lin*, `libsvm`, featuring:

- C- and $\nu-$classification.
- One-class-classification (novelty detection).
- $\epsilon-$ and $\nu-$regression.

and includes:

- linear, polynomial, radial basis function (rbf), and sigmoidal kernels.
- formula interface.
- $k-$fold cross validation.

For further implementation details on `libsvm`, see Chang & Lin (2001).

Let's practice!

Usage in R

Input:

```
# Hello.R
# Importing the dataset
library(fma)
dataset <- dj
# Splitting the dataset
dataset_train = dataset[1:250]
dataset_test = dataset[251:292]
h = 1:250
# install.packages("e1071")
library(e1071)
regressor =svm(formula = dataset_train ~ h,
    y=dataset_train, type = 'eps-regression',kernel =
    'radial',cost = 1,sigma=0.1)
summary(regressor)
```

Output:

```
Call:
svm(formula = dataset_train ~ h, y = dataset_train, type
    = "eps-regression",
    kernel = "radial", cost = 1, sigma = 0.1)

Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  1
    epsilon:  0.1

Number of Support Vectors: 215
```

Predicting a new result.

In:

```
# test with train data
```

```
y_pred =predict(regressor, dataset_test)
table = data.frame(y_pred,dataset_train)
table
```

Out:

y_pred	dataset_train
<dbl>	<dbl>
3621.520	3651.0
3620.027	3645.0
3618.568	3626.0
3617.146	3634.0
3615.764	3620.5
3614.425	3607.0
3613.131	3589.0
3611.885	3590.0
3610.690	3622.0

1-10 of 250 rows

Visualizing the SVR results.

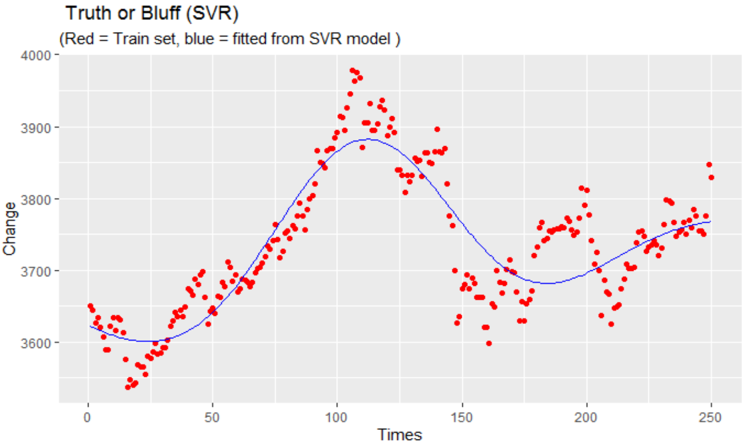


Figure: Dow-Jones Prediction (SVR)

Measures of forecast accuracy

- The Mean Error:

$$ME = \frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_{t|t-1})$$

- The Root Mean Squared Error:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_{t|t-1})^2}$$

- The Mean Absolute Error:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|y_t - \hat{y}_{t|t-1}|$$

- The Mean Percentage Error:

$$MPE = \frac{1}{n}\sum_{t=1}^{n}\left(\frac{y_t - \hat{y}_{t|t-1}}{y_t}\right) \times 100$$

Accuracy measures for a SVR model

The obtained performance measures for the Dow-Jones index series (Training set and Test set) are:

The measures	Training set	Test set
ME (%)	2.60989	2.414896
RMSE (%)	46.5255	31.26024
MAE (%)	36.6118	22.65272
MPE (%)	0.05352	0.063268
MAPE (%)	0.98035	0.584654

Time Series cross-validation.

Assume k is the minimum number of observations for a training set.

1) Select observation $k+i$ for test set, and use observations at times $1, 2, \ldots, k+i-1$ to estimate model.
2) Compute error on forecast for time $k+i$.
3) Repeat for $i = 0, 1, \ldots, T-k$ where T is total number of observations.
4) Compute accuracy measure over all errors.

Applying k fold cross validation

In:

```
# make data frame named ‘Data_Frame_train‘
Data_Frame_train<-data.frame(cbind(h,dj[1:250]))
head(Data_Frame_train)
```

Out:

	h	V2
	<dbl>	<dbl>
1	1	3651.0
2	2	3645.0
3	3	3626.0
4	4	3634.0
5	5	3620.5
6	6	3607.0

6 rows

Train with method svmRadialCost

In:

```
library(caret)
library(lattice)
library(kernlab)

# Define train control for k(10) fold cross validation
control = trainControl(method = "repeatedcv" ,number =10,
    repeats = 3)
set.seed(123)
model = train(V2 ~ h ,data=Data_Frame_train, method
    ='svmRadialCost',trControl=control)
print(model)
plot(model)
```

Out:

```
Support Vector Machines with Radial Basis Function Kernel

250 samples
  1 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 224, 226, 225, 225, 226, 225, ...
Resampling results across tuning parameters:

  C     RMSE      Rsquared   MAE
  0.25  30.48596  0.9116347  22.46143
  0.50  29.30860  0.9165802  21.91550
  1.00  28.76822  0.9183504  21.39149

RMSE was used to select the optimal model using the
    smallest value.
The final value used for the model was C = 1.
```

Out:

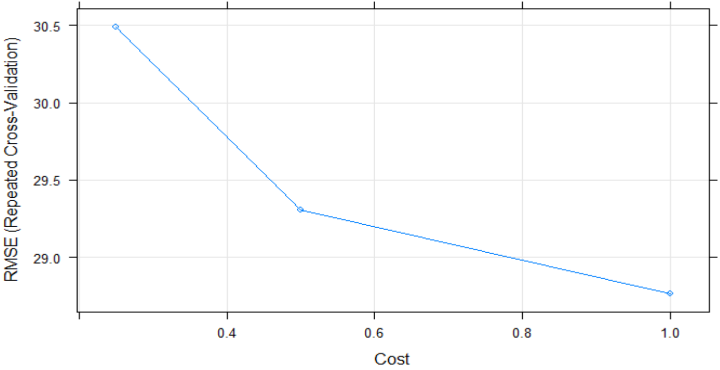


Figure: The variation of RMSE with Cost

We show the optimal model was C=1 and RMSE= 28.77.

Train with method svmRadialSigma

In:

```
control = trainControl(method = "repeatedcv" ,number =10,
    repeats = 3)
set.seed(123)
model = train(V2 ~ h, data=Data_Frame_train, method
    ='svmRadialSigma',trControl=control)
print(model)
plot(model)
```

Out:

```
Support Vector Machines with Radial Basis Function Kernel

250 samples
  1 predictor

No pre-processing
```

```
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 224, 226, 225, 225, 226, 225, ...
Resampling results across tuning parameters:

   sigma       C     RMSE      Rsquared   MAE
   0.1869723 0.25 77.66535 0.4071930 60.32516
   0.1869723 0.50 74.62255 0.4470560 58.08402
   0.1869723 1.00 71.47531 0.4905328 55.40626
  37.0226293 0.25 27.84222 0.9297582 20.70427
  37.0226293 0.50 26.10137 0.9343619 19.56342
  37.0226293 1.00 25.55180 0.9361920 19.25606
  73.8582863 0.25 26.61406 0.9411009 19.46885
  73.8582863 0.50 23.67235 0.9474570 17.62684
  73.8582863 1.00 21.73147 0.9533725 16.40676

RMSE was used to select the optimal model using the
    smallest value.
The final values used for the model were sigma = 73.85829
    and C = 1.
```

Out:

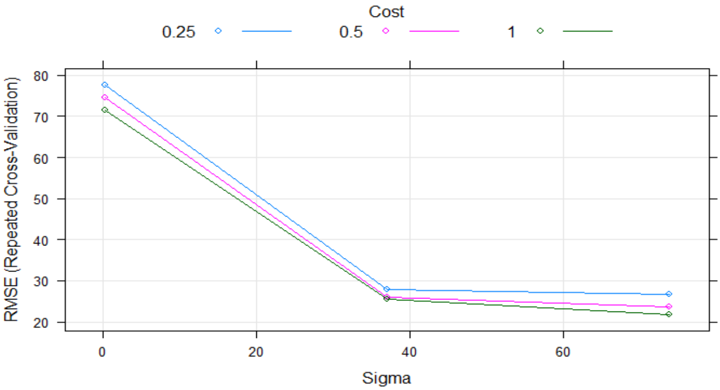


Figure: The variation of RMSE with Sigma

We show the optimal model wers C=1, Sigma=73.86, RMSE= 21.73.

Train with method svmRadial

In:

```
control = trainControl(method = "repeatedcv" ,number =10,
    repeats = 3)
set.seed(123)
model = train(V2 ~ h, data=Data_Frame_train, method
    ='svmRadial',trControl=control)
print(model)
plot(model)
```

Out:

```
Support Vector Machines with Radial Basis Function Kernel

250 samples
  1 predictor
```

```
No pre-processing
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 224, 226, 225, 225, 226, 225, ...
Resampling results across tuning parameters:

  C     RMSE      Rsquared   MAE
  0.25  27.84222  0.9297582  20.70427
  0.50  26.10137  0.9343619  19.56342
  1.00  25.55180  0.9361920  19.25606

Tuning parameter 'sigma' was held constant at a value of
    37.02263
RMSE was used to select the optimal model using the
    smallest value.
The final values used for the model were sigma = 37.02263
    and C = 1.
```

Comparing with SVMPoly

In:

```
control = trainControl(method = "repeatedcv" ,number =10,
    repeats = 3)
set.seed(123)
model = train(V2 ~ h, data=Data_Frame_train, method
    ='svmPoly',trControl=control)
print(model)
plot(model)
```

Out:

```
Support Vector Machines with Polynomial Kernel

250 samples
  1 predictor

No pre-processing
```

```
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 224, 226, 225, 225, 226, 225, ...
Resampling results across tuning parameters:

  degree scale C     RMSE     Rsquared  MAE
  1      0.001 0.25  98.15661 0.1087096 79.76221
  1      0.001 0.50  97.62623 0.1087096 78.98749
  .        .     .     .        .         .
  .        .     .     .        .         .
  .        .     .     .        .         .
  3      0.100 1.00  79.52454 0.3709354 62.29495
RMSE was used to select the optimal model using the
    smallest value.
The final values used for the model were degree = 3,
    scale = 0.1 and C = 1.
```

Out:

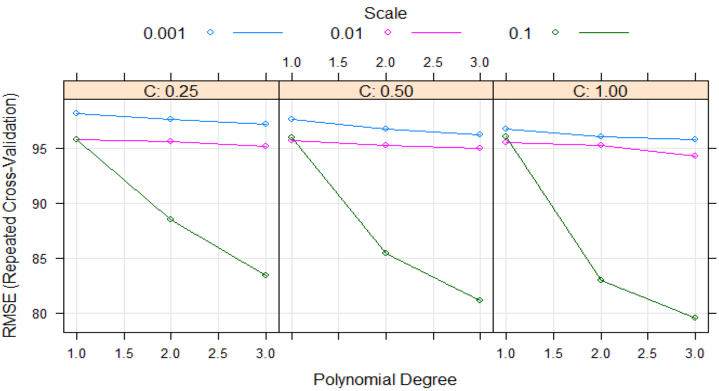


Figure: The variation of RMSE with Polynomial Degree

We show the optimal model wers C=1, scale=0.1, RMSE= 79.52, degree =3.

Bibliography

- Jonathan D. Cryer, Kung-Sik Chan *Time Series Analysis With Applications in R*, Second Edition, 2008, 1-11.
- Mokhtar TAFFAR, *INITIATION A L'APPRENTISSAGE AUTOMATIQUE*,1-42.
- Rob J Hyndman, Yeasmin Khandakar, *Automatic Time Series Forecasting: the forecast Package for R*, July 2008, Volume 27, Issue 3.
- David Meyer, *Support Vector Machines The Interface to libsvm in package e1071*, November 25, 2019.
- External links:
  - https://robjhyndman.com/hyndsight/tscv/
  - https://youtu.be/jq9r3xr4vIM
  - https://github.com/oukhouya62/Pr-sentation_Support_Vector_Machines-

Happy forecasting

Thanks a lot

for your patience in listening

A good forecaster is not smarter than everyone else, he merely has his ignorance better organised.

- *Anonymous*