Support Vector Regression Analysis for Price Prediction in a Car Leasing Application

Master Thesis

March 2009

Mariana Listiani
Matriculation Number: 33750
Information and Media Technology
Hamburg University of Technology

Reviewed by:
Prof. Dr. Ralf. Möller
Institute of Software, Technology, and Systems -
Hamburg University of Technology
Prof. Dr. Michael Morlock
Institute of Biomechanics - Hamburg University of Technology

External Supervisor:
Dr. Stefan Lessmann
Institute of Information Systems
Department of Business Administration - University of Hamburg

Abstract

To remain profitable under a tight competition, a leasing company has to offer a good leasing price. In order to determine the right price, it is necessary to predict the future price of a second hand car. By knowing the car's value depreciation, the leasing price could be set to cover it. The approach commonly used for a price prediction task is multiple linear regression analysis. However, there are a large number of factors that drive the price, that make this crucial task difficult. The standard regression approach might not be suitable for high dimensional data. A modern data mining technique which is independent of input dimension, namely Support Vector Regression, will be applied to overcome this potential problem. The forecasting accuracy will then be compared against the statistical regression model. In particular, a fully automatic approach for tuning and applying SVR is developed, borrowing ideas from the field of evolutionary search. The whole experiment with the machine learning approach is based upon a real-world data from a leading German car manufacturer.

Keywords: Support Vector Regression, parameter search optimization, hyperparameter selection, learning curve, grid search, evolution strategy, SVM parameter impact

Declaration

I declare that:
this work has been prepared by myself,
all literal or content based quotations are clearly pointed out,
and no other sources or aids than the declared ones have been used.

Hamburg, Germany

March 9, 2009

Mariana Listiani

Acknowledgement

First of all, I would like to thank Prof.Möller that gives me an introduction to the interesting world of data mining, and for giving me the chance to write this thesis in the Institute of Software, Technology, and Systems. I would also thank Prof.Morlock for giving a very good statistics lecture, that enables me to conduct and analyze the statistical experiment in this thesis.

My biggest gratitude goes to Dr.Stefan Lessmann, who has been inspiring and guiding me through the experiment, criticizing this report, and even giving me a support for writing a joint paper on this thesis. I thank as well Dr.Stefan Gnutzmann for the introduction on the experimental data, and all scientists who build a profound basis for the support vector regression, and wrote the articles or books that I use as the information sources for this thesis.

Moreover, I have to express my gratefulness to Siemens, the company which sponsors my double degree master study at Hamburg University of Technology and at Northern Institute of Technology Management. Without Siemens, I would not have been able to reach my dream this far.

A bunch of appreciation goes to my beloved mom, dad, sister, and my extended family in Indonesia for their support and prayer. A special thank is dedicated to my boyfriend, Bernd for his patience, time and sincere caring. I would like to also mention my clever and funny international friends, Shen, Chris, Gabi and Arturo, who always cheer me up and share knowledge, thought, laughter, as well as delicious meals together.

Last but not least, praise God for the strength that He gives me to finalize this experiment and thesis.

Hamburg, Germany

March 9, 2009

Mariana Listiani

List of Abbreviations

DAT	Deutsche Automobil Treuhand
EA	Evolutionary Algorithm
ES	Evolution Strategy
ERM	Empirical Risk Minimization
GA	Genetic Algorithm
GS	Grid Search
MSE	Mean Squared Error
RBF	Radial Basis Function
RMSE	Root Mean Squared Error
SCC	Squared Correlation Coefficient
SRM	Structural Risk Minimization
SSE	Sum of Squared Error
SVM	Support Vector Machine
SVM	Support Vector Regression
VC	Vapnik-Chervonenkis

Contents

Chapter 1

Introduction

1.1 Car Leasing

Leasing is one of the options for financing a good. The lessor simply pays for and retains ownership of the good, but permits the lessee to use it. In return, the lessee is required to pay a rent, which covers the exclusive-right-of-use-fee, the loss of good's value and the interest on money 'loaned' over the leasing period. Leasing is often favored by those people who like to enjoy the benefits of new goods but would not like to be burdened with the risk of re-selling, or because of tax incentives and some other reasons.

In many parts of the world it is common to lease a vehicle. There are usually two types of calculations, kilometer or residual value leasing. With the former, the customer agrees to drive a certain number of kilometers per year, whereas with the later both parties agree on a final value of the vehicle after the leasing period. When the customer returns the vehicle, irrespective of the deal, the leasing company will do 'wear-and-tear' evaluation. The vehicle will be re-valued and one may be liable for additional body repair or refurbishment costs if the value is actually less than planned.

Figure 1.1 depicts an example of residual value leasing. The initial value of the car is 100,000 euro. According to the prediction, the loss of the car's value for two years is 40,000 euro, or equal to 1,667 euro per month when counted linearly. In order to cover the loan interest and lessor's benefit, the customer should pay 2,000 euro per month. Given that the interest and initial car's price are known, one main factor that determines the leasing price is the unknown future residual value. Therefore the lessor has to predict this value accurately, in order to make a profitable offer. The price should not be too low, otherwise the lessor will suffer from loss when re-selling the car, and not too high, otherwise the customers will buy or lease from other providers.

In Germany, there are well-known lists, namely Schwacke List [AG09] and Deutsche Automobil Treuhand GmbH (DAT) List [Gmb09], that help public domain to approximate the value of a second hand car, based on its attributes like type, year of manufacture, common equipments and kilometers driven. Nevertheless, one has to note that both lists have a limitation on the attributes scope. Only basic car equipments are included for price calculation, while other features, that may also be influential, are failed to be captured.

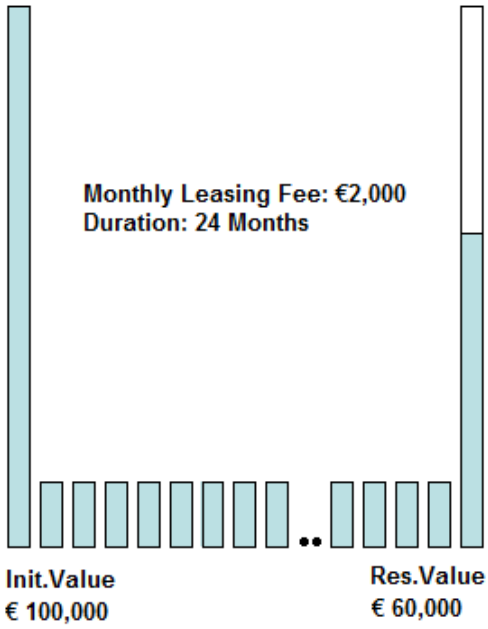


Figure 1.1: Car Leasing with Residual Value Scheme

		Time	
		Current	Future
Factor	Internal	type, equipments	vehicle's condition
	External	interest rate	general economic and vehicle market condition, location, change of policy, e.g. incentive for environmental-friendly cars, tax for old cars

Table 1.1: Examples on Two Dimensional Factors Affecting Used Car Price

Moreover, there are other factors that play a role in determining the residual value, which could be observed from the examples in Table 1.1. An external factor is unlikely recorded if it does not show any direct impact, since it would probably only waste data storage and maintenance resources, but could not provide relevant information, whereas a future factor cannot be captured because it is unknown. They both account for error while predicting the salvage value of a used vehicle, since one can only base the prediction on the new car's attributes. The price during the worldwide economic depression in the second half of 2008, for example, cannot be predicted correctly based on the historical data within normal conditions. Despite the absence of future and external factors, an accurate residual value prediction model based upon available internal factors needs to be optimized here.

To understand the importance of a prediction model in a bigger business-perspective, we will look at some car market data. Germany's yearly vehichle market volume in 2004 was 160 billion euro. Nevertheless, the industry only left 4.1 billion euro or 2.5% profit to be divided between car producers, banks, importers, spare parts- and car dealers (see Figure 1.2). The used car segment which generates 52 billion revenue yields just 0.03 billion or 1%,

altough it has a profit potential of 3% [Con05]. This is due to the lack of professionalism in dealers-networks and over production that leads to a rebate war. An abundance of relatively new second hand vehicles forces the residual value for leasing to decrease, and at the end of the leasing period, car manufacturer and dealers then have to clear this expensive cost. Also in the USA, Germany's car makers had to cover around 1 billion euro lost while reselling 2.2 million cars, whose leasing contracts ended in 2008 [Gmb08]. To stop further lost in the used car segment, Mercer and Oliver Wyman Management Consulting suggested a residual value management that covers the whole product life cycle. One of the steps is the professional price calculation by dealers [MC06]. This thesis tries to answer the challenge in optimizing the residual value prediction.

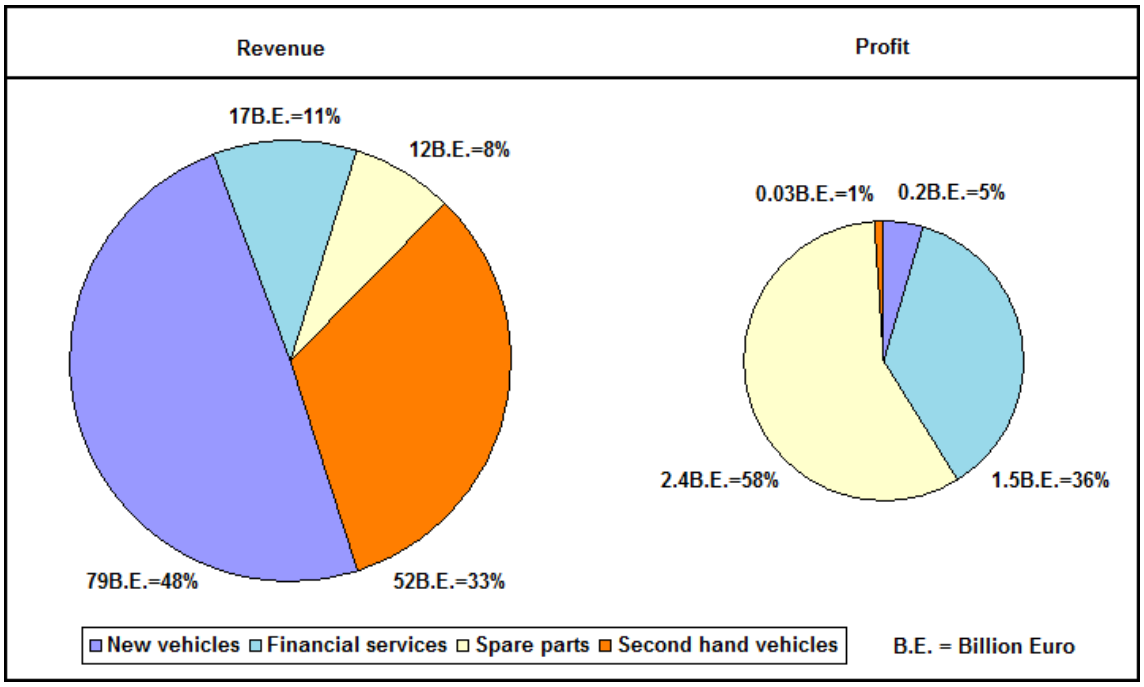


Figure 1.2: Division of Revenue and Profit in Germany's Car Market Segments [Con05]

1.2 Problem Definition

The basic problem in this thesis is to find *a good regression model* that can explain the vehicle's residual price. In order to solve it, a set of historical data of second hand cars with their attributes and their re-sell price is provided. The attributes of the cars being the independent variables or input, while the price being the dependent variable or output. Given some samples of input and output, we would like to find the unknown function that explains the relationship between them. As a simplified illustration, the regression function in Figure 1.3, $y = f(x) = wx + b$, with w as the weight of factor x and b as the constant, maps one independent variable, e.g. the kilometer driven (x), to the price (y).

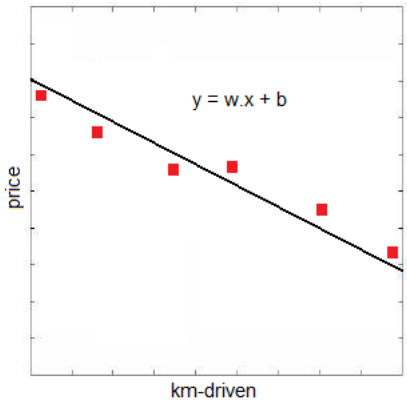


Figure 1.3: Example of a Regression Function with One Independent Variable

The input space of used car data is high dimensional, which means, there are many independent variables that may affect the price. Learning from a huge set of high dimensional data is not an easy task. Hence, one needs a machine that can learn from this data set, deal with such complexity and generate a good model out of it. A good model is required to have a good performance on future or previously unseen data, hereafter regarded as test or validation set. In other words the prediction model should not be [Fre02]

- Over-fitting: where the solution derived from training data is more complex than the real function. The model is too tailored to the training data, and thus it does not fit well for unseen data.
- Under-fitting: where the solution derived from training data is too simple, and therefore incapable to capture the right influence of the independent variables.

Both have the same effect of reducing model's accuracy in predicting future outcome.

It is also important to note, that prediciting the future is an *ill-defined, non-deterministic* task [Fre02]. In the sense that, using only training data, one cannot be sure that the discovered function will have a high predictive accuracy on the test set. The reason is, there could be more than one hypothesis function that confirms with all training data, and how should one choose from among multiple consistent hypotheses? This question will be answered later in chapter two, specifically for the learning machine used.

1.3 Motivation and Research Contribution

As described previously, the learning problem here is difficult, yet it is crucial for improving the marginal profit in the used car market. Therefore, a state-of-the-art learning machine, namely Support Vector Machine (SVM), is going to be applied for the regression problem here, due to its computation that does not depend on the input space dimension.

This thesis will also provide additional empirical study for SVM users. An experiment methodology will be applied to handle a huge data set, in order to maximize the use of the information available, without getting trapped in an excessive learning duration.

Moreover, one can learn about parameter search automization and parameters impact from the experiment results. The input needed in SVM is not only a historical data set, but also some parameters; and the performance of SVM in generating a model is highly dependent on the chosen parameters (see Figure 1.4). To replace the inefficient try-and-error approach, two automatic parameters selection schemes will be tested and reported here.

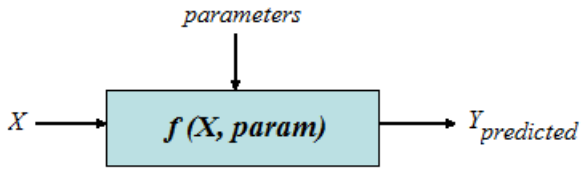


Figure 1.4: Support Vector Machine Prediction Model

1.4 Outline

The reminder of this work is organized as follow. Chapter 2 describes the background theories on two approaches that will be used to solve this problem and their suitability for implementation: linear regression analysis and Support Vector Regression (SVR). Supporting knowledge for using SVR, namely kernel functions and parameters selection algorithms, will also be explained here.

In chapter 3 the methodology will be elaborated in detail, including the reasoning behind each step taken along the course of this experiment. Starting from data preparation, learning from a smaller data set until the final model building and testing. Chapter 4 will present the experiment results and its analysis. While chapter 5 shows some benchmarking results with the statistical linear regression. Finally, conclusions will be drawn in chapter 6, along with some limitation from this study and outlook for the future research.

Chapter 2

Background Theory

In regression, the task is to find a mapping between input or independent variables x, and the dependent output y, where y is a continuous value instead of a discrete one as in classification. The approach used for such a problem is usually linear regression. However, the linear regression is based on some assumptions, that cannot always be matched by the characteristic of a contemporary data set. Thus, these assumptions pose a limitation on statistical regression analysis. Therefore, SVM is used to counter it, and the background theory on how it solves a regression problem will be explained in here. Two general views on the risk of prediction will also be elaborated to clarify the underlying difference between these two approaches.

2.1 Regression Analysis

This subchapter will start with simple linear regression before proceeding with the multiple one.

2.1.1 Linear Regression

Simple Linear Regression

In simple linear regression, there is only one independent variable. Given n samples of observation $(x_i, y_i), i = 1 \cdots n$, the regression model is [KKMN98]

$$y_i = w_0 + w_1 x_i + \varepsilon_i \tag{2.1}$$

where w_0, w_1 are the regression coefficients, and ε_i is the residual, i.e. the difference between the desired and predicted value of the dependent variable $\varepsilon_i = y_i - \overline{y}$.

The most common method to estimate the regression line is by minimizing the sum of squared residuals

$$SSE = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} (y_i - \overline{y})^2 \tag{2.2}$$

Minimization of this function solves a predicted regression slope w_1 and intercept w_0, that gives the *regression line* $y = \hat{w_0} + \hat{w_1}x$

$$\hat{w_1} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2} \quad \text{and} \quad \hat{w_0} = \overline{y} - \hat{w_1}\overline{x} \tag{2.3}$$

Multiple Linear Regression

Multiple Linear Regression deals with problems having more than one (m) independent variables [KKMN98]. Given n samples of observation $(x_{i1}, \cdots, x_{im}, y_i), i = 1 \cdots n$, the regression model is

$$y_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_m x_{im} + \varepsilon_i \tag{2.4}$$

There are $p + 1$ regression coefficients, and the residual ε_i to capture other influences on y_i apart from $w_1 x_{i1}, \cdots, w_m x_{im}$.

To ease the computation one could represent equation 2.4 in matrix notation:

$$Y = wX + \varepsilon \quad \text{with} \quad X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1m} \\ 1 & x_{21} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nm} \end{pmatrix} \quad \text{and} \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \tag{2.5}$$

The coefficients matrix w is then found by solving the equation below [1].

$$\left(X^T X\right)\hat{w} = X^T Y \quad \text{that yields} \quad \hat{w} = \left(X^T X\right)^{-1} X^T Y \tag{2.6}$$

2.1.2 Linear Regression Assumptions

Before doing the aforementioned linear regression analysis, there are some prerequisites to be checked [2] [Abr07]:

- Linearity
  Linear regression assumes that there is a straight line relationship between *continuous independent variables* and the dependent one. It could be seen from a bivariate scatterplot, a graph with the independent variable on one axis and the dependent variable on the other.

- Normality
  The dependent variable, as well as the independent ones should be normally distributed. This could be checked by several ways, e.g. looking at the histogram for each variable. Another way is by calculating the skewness and kurtosis for each variable. Skewness is a measure how symmetrical the data is. When the data is skewed,

[1] The first column of X is used to represent the intercept term w_0.

[2] Failing to satistfy those assumptions does not mean that the resulting regression model will be completely wrong, it will just under-fit the real relationship [Sch03].

then the mean is not in the middle of the distribution, and thus not normally distributed. While kurtosis is a measure how peaked the distribution is, and normality means it is not too peaked and not too flat. Any value greater than +3 or less than -3 needs a pre-transformation before linear regression.

- Homoscedasticity
  Linear regression also assumes that the relationship between the dependent variable and binary independent variables are homoscedastic. This means that the residuals are approximately equal for all predicted dependent variable scores. One can check homoscedasticity by looking at the residuals plot, where the x-axis is the standardized predicted value and the y-axis the standardized residual. Data is homoscedastic if the residuals plot is the same width for all values of the predicted dependent variable. Heteroscedasticity is usually shown by a cluster of points that is wider as the values for the predicted dependent variable get larger.

- Multicollinearity and Singularity
  Multicollinearity is a condition where the independent variables are very highly correlated (.90 or greater), and singularity is when they are perfectly correlated, e.g. one independent variable is a combination of one or more of the other independent variables. High bivariate correlations can be seen by running correlations among the independent variables. If there is high bivariate correlations, one of the two variables has to be deleted.

To wrap up the prerequisites, the statistical inference is based on the following assumptions:

1. Data can be modeled by a set of linear functions.
2. The data follows a normal probability or Gaussian distribution.
3. Because of the first assumption, the induction for parameter estimation is the *maximum likelihood* method, which is then translated as minimization of sum-of-errors-squares cost function.

This assumptions turned out to be inappropriate for many contemporary real-life problems because of the following issues [HKK06]:

1. Modern problems are high dimensional and if the mapping is not very smooth, the linear paradigm needs an exponentially increasing number of samples with an increasing number of independent variables. This is known as *the curse of dimensionality* [Bel61]. For example, to build a model with one dimension, one needs only n samples, but for generating a model with d dimensions, one will need approximately n^d samples to achieve the same accuracy. Table 2.1 shows the curse of dimensionality problem in Silverman's experiment [Sil86], where a density function had to be estimated with a predefined accuracy.

Dimensionality	Required Sample Size
1	4
3	67
5	786
7	10,700
10	842,000

Table 2.1: Sample Size Required to Estimate a Density Function with Accuracy of 0.1

2. The data may be far from the normal distribution.

3. Because of the above mentioned tendencies, maximum likelihood method estimator (and consequently the sum-of-errors-squares cost function) should be replaced by other induction paradigm, in order to model non-Gaussian distribution.

SVM is a machine learning method that has been developed to work with data sets that are typically high dimensional and sparse (data set contains a small number of the training data pairs). More detail of it will be elaborated below.

2.2 Basic Idea of Support Vector Machine

According to [RN03], there are three types of machine learning: unsupervised learning, reinforcement learning and supervised learning. In supervised learning the *desired outcome* is available from the training data set. The supervised learning is divided into two categories, namely regression (if the outcome is a continuous value) and classification (if the outcome is a class label).

A supervised learning consists of basically two phases. The first is the learning phase, where training data is used to build a mathematical model that explains the relationship between some variables. The second is the test phase, where the model is used to predict the outcome of test data set [RN03].

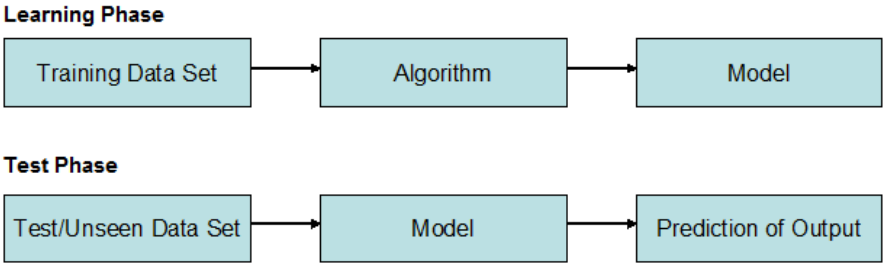


Figure 2.1: Supervised Learning Scheme

Empirical Risk Minimization (ERM)
The problem setting in this supervised learning is: there is some unknown dependency between some high dimensional input vector x and the scalar output vector y. A machine with parameters w is used to learn the mapping $x \rightarrow y$. The machine will tune its

parameters w to get a possible mapping $x \rightarrow f(x, w)$. Since the actual risk measure of a machine is not always possible, one could relax it to empirical risk measure. It is the error associated with a function f over the training data set, and usually defined as:

$$f(x, w) = \frac{1}{n} \sum_{i=1}^{n} E\left(y - f(x, w)\right)^2 \tag{2.7}$$

Most training algorithms for learning machines implement ERM, i.e. minimise the empirical error based on maximum likelihood estimation for the parameters w. Those conventional training algorithms do not consider the capacity of the learning machine and this can result in over-fitting, i.e. using a learning machine with too much capacity while learning from a finite training data set [Vap98, Chi98b].

Structural Risk Minimization (SRM)
A new paradigm of risk, i.e. SRM, is introduced by Vapnik and Chervonenkis to overcome this problem. SRM is an inductive principle for model selection used for learning from finite training data sets, that prescribe a way to balance the learning power of a machine [Vap98, VC74]. The goal of SRM is to find the learning machine that yields a good trade-off between *low empirical risk and small capacity*. There are two major problems in achieving this goal [Chi98b].

- The SRM requires a measure of the capacity of a particular learning machine, or at least an upper bound of this measure.
- An algorithm to select the desired learning machine according to SRM's goal is needed. One can divide the entire class of machines into nested subsets with decreasing capacity. Then one can train a series of machines, one for each subset, using the ERM principle. Finally the machine that gives the best trade-off can be selected. This can be a very difficult task. An alternative is to define a learning machine with variable capacity and a corresponding training algorithm that minimizes both the empirical error and capacity of that machine.

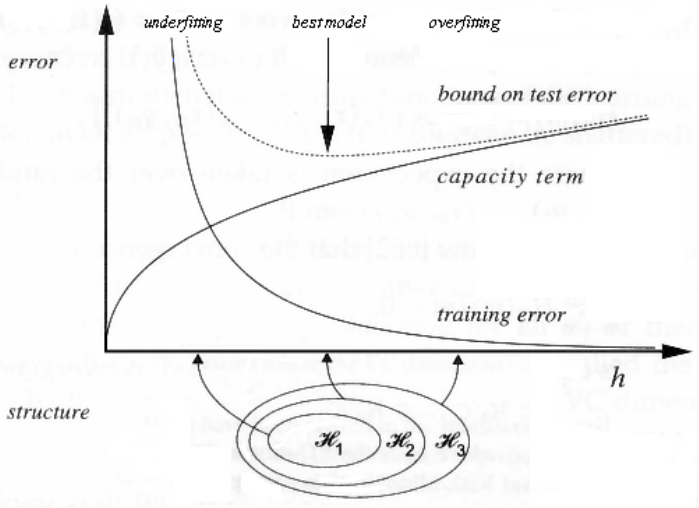


Figure 2.2: Structural Risk Minimization [VC74]

To address the first problems, the concept of Vapnik Chervenenkis (VC) confidence is developed. VC confidence is defined as the right hand part of equation 2.8. The upper bound of the generalization risk with a probability of $1-\eta$ is given by

$$R(w_m) \leq R_{emp}(w_m) + \sqrt{\frac{h\left[ln\left(\frac{2n}{h}\right)+1\right] - ln\left(\frac{\eta}{4}\right)}{n}} \tag{2.8}$$

where:
w_m = parameter vector of model m subject to training
h = Vapnik-Chervonenkis (VC) dimension, a measure of model complexity. A machine with oriented hyperplanes in R^m as its mapping function, has a VC dimension of $m+1$
n = data set size

Figure 2.2 clearly shows that a learning machine with large capacity will give low empirical risk, but the VC confidence interval is also large for that learning machine, i.e the machine does not generalise well. By measuring this bound, one can select a learning machine that give the lowest expected generalization risk, which is the point where the combination of the VC confidence and the empirical error is at the minimum.

Based on SRM, Support Vector Machine (SVM) is proposed to tackle the second problem [Vap98]. The SVM employed the following generalization risk:

$$R = \sum_{i=1}^{n} \underbrace{L_\varepsilon}_{\text{empirical error}} + \underbrace{\Omega(n,h)}_{\text{Capacity of a machine}} \tag{2.9}$$

2.3 Support Vector Regression

The SVM used to solve regression problem is called Support Vector Regression. The learning machine is given a training data set $\{\chi = [x_i, y_i] \in R^m \times R, i = 1 \cdots n\}$, where the inputs x are m-dimensional vectors $x \in R^m$ and the outcomes $y \in R$ are continuous values. The approximation function of this data is a *linear regression hyperplane*

$$f(x,w) = w^T x + b \tag{2.10}$$

In measuring the error of approximation, Support Vector Machine uses a novel equation called Vapnik's ε-insensitivity error function which is defined as [HKK06]

$$|y - f(x,w)|_\varepsilon = \begin{cases} 0 & , \quad if\, |y - f(x,w)| \leq \varepsilon \\ |y - f(x,w)| - \varepsilon & , \quad otherwise \end{cases} \tag{2.11}$$

where ε is a radius of a tube within which the regression function must lie, after the successful learning. The idea is to reduce model's complexity by tolerating errors up to a certain point. Figure 2.3 gives a visual comparison of Vapnik's ε-insensitivity error function to other two classical error functions, namely the quadratic error $(y - f(x,w))^2$, as in linear regression, and the absolute error $|y - f(x,w)|$.

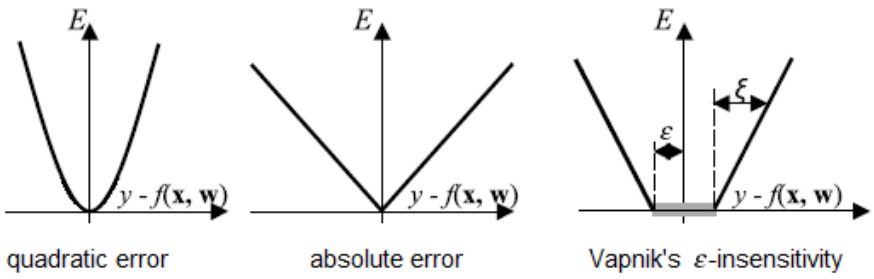


Figure 2.3: Error Functions [HKK06]

As mentioned in the introduction part, SVM has to deal with a non-deterministic problem. This means, there could be more than one hypothesis functions that have the same empirical risk on the training data as the regression function, as illustrated in figure 2.4.

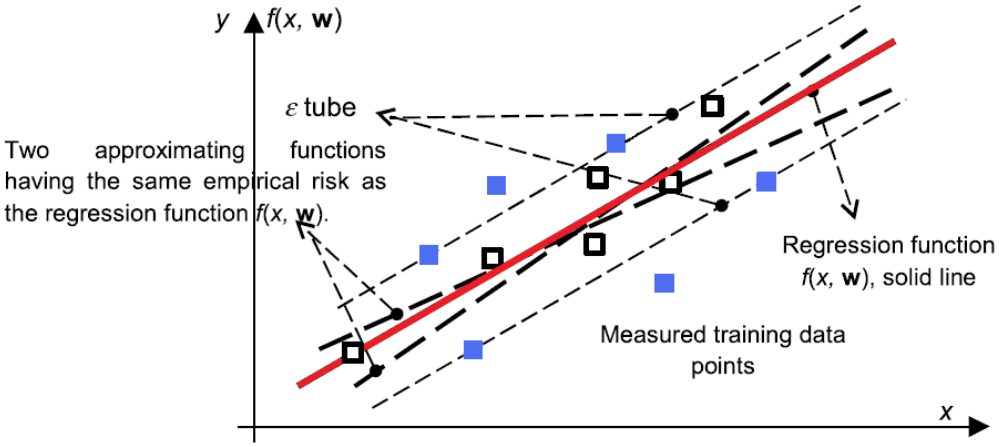


Figure 2.4: Two linear approximations (dashed lines) have the same empirical risk on the training data as the regression function (solid line) [HKK06]

Given some functions with the same empirical/training error, to achieve minimum generalization error, i.e. the optimal solution, SVM should minimize the capacity of machine learning, and consequently the model complexity, which is reflected by $\|w\|^2$ [HKK06]. Thus, this problem could be then viewed as a convex optimization problem as in 2.12.

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|w\|^2 \\ \text{subject to} & \begin{cases} y_i - \langle w, x_i \rangle - b \leq \varepsilon \\ \langle w, x_i \rangle + b - y_i \leq \varepsilon \end{cases} \end{array} \tag{2.12}$$

Due to the use of Vapnik's ε-insensitivity loss function, for all training data outside the ε-tube a new 'error' variable is introduced.

$$\begin{array}{lll} |y - f(x, w)| - \varepsilon = \xi & , & \text{for data above an } \varepsilon\text{-tube, and} \\ |y - f(x, w)| - \varepsilon = \xi^* & , & \text{for data below an } \varepsilon\text{-tube} \end{array} \tag{2.13}$$

The slack variables should be kept at a minimum, which implies that there should be a function that penalizes non-zero ξ_i, ξ_i^*. This can be done by adding the sum of all slack

variables in the objective function and the cost $C > 0$, a parameter to be determined by the user, which controls the trade-off between the model complexity and the amount up to which deviations larger than ε are tolerated. Hence the formula 2.12 could be restated as [SS03]

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ \text{subject to} & \begin{cases} y_i - \langle w, x_i\rangle - b \leq \varepsilon + \xi_i & i = 1..n \\ \langle w, x_i\rangle + b - y_i \leq \varepsilon + \xi_i^* & i = 1..n \\ \xi_i, \xi_i^* \geq 0 & i = 1..n \end{cases} \end{array} \tag{2.14}$$

This is a classic quadratic optimization problem with inequality constraints. Such optimization problem could be solved by the saddle points λ of the Lagrangian Λ.

$$\Lambda(x, \lambda) = f + \sum_k \lambda_k g_k \tag{2.15}$$

where $f(x)$ is the objective function and $g_k(x) = 0$ are the constrains [3].

The primal Lagrangian function is as follow

$$\begin{aligned} L_p(w, b, \xi_i, \xi_i^*, \alpha_i, \alpha_i^*, \beta_i, \beta_i^*) = \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ & - \sum_{i=1}^{n}\alpha_i(\varepsilon + \xi_i - y_i + \langle w, x_i\rangle + b) \\ & - \sum_{i=1}^{n}\alpha_i(\varepsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b) \\ & - \sum_{i=1}^{n}(\beta_i\xi_i + \beta_i^*\xi_i^*) \end{aligned} \tag{2.16}$$

The stationary points are achieved by solving a number of equations resulted from setting Langragian's derivation to zero $\nabla\Lambda = 0$.

$$\partial_b L = \sum_{i=1}^{n}(\alpha_i^* - \alpha_i) = 0 \tag{2.17}$$

$$\partial_w L = w - \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\, x_i = 0 \tag{2.18}$$

$$\partial_{\xi_i} L = C - \alpha_i - \beta_i = 0 \tag{2.19}$$

$$\partial_{\xi_i^*} L = C - \alpha_i^* - \beta_i^* = 0 \tag{2.20}$$

After the substitution of these derivations into 2.16 and eliminating dual variable β_i, β_i^*, it becomes a maximization of a dual variables Lagrangian below[SS03]

$$\begin{array}{ll} L_d(\alpha_i, \alpha_i^*) = & -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)\left(\alpha_j - \alpha_j^*\right)\langle x_i, x_j\rangle \\ & -\varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*) \\ \text{subject to} & \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \text{ and } \alpha_i, \alpha_i^* \in [0, C] \end{array} \tag{2.21}$$

[3] The domain of f should be an open set containing all points satisfying the constraints. Furthermore, f and the g_k must have continuous first partial derivatives and the gradients of the g_k must not be zero on the domain [CST00].

The equation 2.18 could be rewritten as

$$w = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) x_i \text{thus} \tag{2.22}$$

$$f(x) = \langle w, x \rangle + b = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \langle x_i, x \rangle + b \tag{2.23}$$

While b is given by [SS03].

$$max \{-\varepsilon + y_i - \langle w, x_i \rangle | \alpha_i < C \text{ or } \alpha_i^* > 0\} \leq b \leq min \{-\varepsilon + y_i - \langle w, x_i \rangle | \alpha_i^* < C\} \tag{2.24}$$

After the learning process, the number of support vectors is equal to the number of non-zero α_i and α_i^*. Moreover, since w can be described as a linear combination of the training patterns x_i, the complexity of a function's representation is *independent of the dimensionality of the input space* χ, and depends only on the number of support vectors [HKK06]. This independency is a strength of SVM in dealing with high dimensional input, which is also useful for the formulation of a non-linear extension.

2.4 Kernel Functions

In general, complex data needs a more expressive function than a linear one. In this case, SVM first non linearly transforms the original input space into a higher dimensional feature space [CST00]. Afterwards, the SVM will do the same linear calculation to find the optimal regression hyperplane in this feature space, as illustrated in figure 2.5.

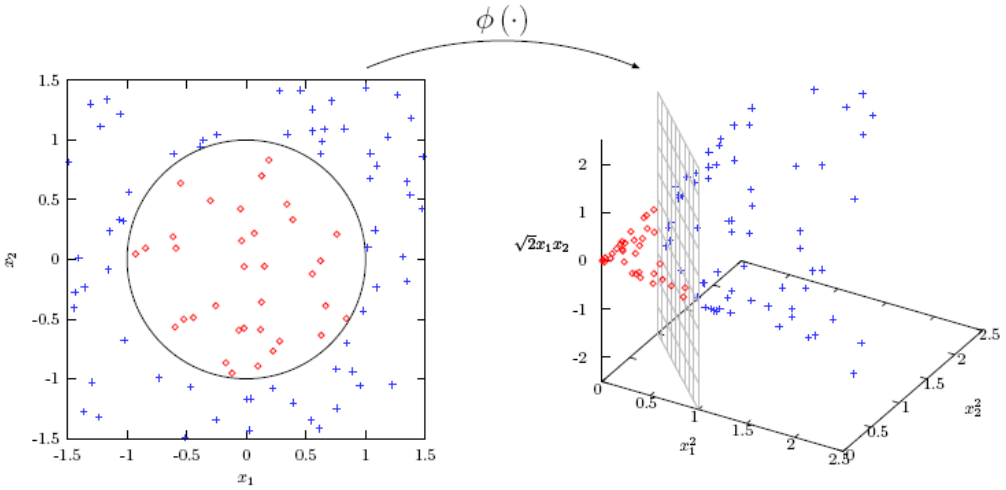


Figure 2.5: After the mapping of a two dimensional classification set into a three dimensional feature space, the data becomes linearly separable. [Gij07]

The SVM algorithm in its dual formulation depends only on the inner products of the training samples. Thus, one first maps data points using Φ, then the formula would

be simply the inner products of the points in the feature space. This mapping into the hypothetical feature space is implicit, therefore it becomes feasible to use feature spaces of infinite dimensionality [CST00].

$$k(x,z) = \langle \Phi(x), \Phi(z) \rangle \tag{2.25}$$

By integrating the kernel function into 2.21 one would get the following formula

$$\begin{aligned} L_d(\alpha_i, \alpha_i^*) = \quad & -\tfrac{1}{2} \sum_{i,j=1}^{n} (\alpha_i - \alpha_i^*) \left(\alpha_j - \alpha_j^*\right) k(x_i, x_j) \\ & -\varepsilon \sum_{i=1}^{n} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i (\alpha_i - \alpha_i^*) \\ \text{subject to} \quad & \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) = 0 \text{ and } \alpha_i, \alpha_i^* \in [0, C] \end{aligned} \tag{2.26}$$

Not all kernels are valid to be applied on the equation 2.26. Only those which satisfy the symmetric, continuous, and positive semi-definite conditions are *admissible* [HSS08]. There are several common kernel functions that are proven to be admissible for classification and regression purposes, such as polynomial and radial basis function (RBF) [HCL08].

$$\begin{aligned} & \textit{Polynomial}: && (x,z) = (\langle x, z \rangle + c)^d && \text{with } d \text{ natural number}, c \geq 0 \\ & \textit{RBF}: && k(x,z) = \exp\left(-\gamma \|x - z\|^2\right) && \text{with } \gamma > 0 \end{aligned} \tag{2.27}$$

2.5 Hyperparameters Selection

These kernel functions in 2.27 are parameterized to allow for adjustments with respect to the training data. The kernel parameters, as well as SVM parameters C and ε, optimize the generalization performance, and they are known as *hyperparameters*.

The selection of hyperparameters determines the performance of SVM significantly, meaning that the model's accuracy depends on those parameters. Therefore, it is important to set good values for them. The optimal hyperparameter setting depends on the actual training data, and commonly must be set manually, because there is currently no theory that defines what a good value is.

The choice of ε is generally easier than the choice of C. It is given as either maximally allowed error or some desired percentage of the output mean value $\overline{y}$, e.g. $\varepsilon = 0.05$ of the mean value of y [HKK06]. Whereas for C, there is a prescription on the approximate value from V. Cherkassky and Y. Ma [CM02]

$$C = max(|\overline{y} + 3\sigma_y|, |\overline{y} - 3\sigma_y|) \tag{2.28}$$

Traditionally people use empirical try-and-error approach or intuition for hyperparameters selection. This is insufficient because try-and-error is time consuming, and an inappropriate manual setting can hamper SVM performance on problem solving. Therefore, an automated approach for hyperparameters selection is needed here.

A popular way is to employ *grid search model selection*. The basic idea is to set several values of each parameter within certain range using intervals, e.g. $C = \{100, 200 \cdots 1000\}$ and $\varepsilon = \{2^{-2}, 2^{0} \cdots 2^{8}\}$. After running SVM training and testing for each parameters combination, choose the good values by looking at the least error generated. This grid search could be done in coarse steps at the beginning to find the approximation area, and then in finer steps to find the optimum parameters.

2.6 Evolution Strategy

Even though grid search is an automatic approach, it is still computation expensive, since machine learning has to be performed over a wide range of parameter values. Therefore, one still needs an automatic approach to choose the good values with less computation. Evolutionary Algorithms (EA) could be applied to improve it.

EAs consist of several search algorithms that are based on Darwin's evolution theory, the general idea is as follow: [Fre02]

- EAs work with a population of individuals.
- To get the final solution, the population experiences several rounds of selection process, each resulting in a new population of selected individuals. The selection is based on the fitness of an individual as a solution. The better it is, the more often the individual is selected, and the more some parts of its 'genetic material' will be passed on to later generations.
- They generate new individuals by applying stochastic operators to the existing individuals of the current generation. The two popular operators are crossover (recombination) and mutation. Crossover swaps some genetic material between two or more individuals, while mutation changes the value of a small part of the genetic material of an individual to a random value, simulating the errorneous self-replication of individuals.

Deriving from this general idea to the problem, the candidate values of SVM hyperparameters are the individuals in EA. Initially certain values have to be set to form the parents population. After undergoing several rounds of regeneration and selection, these values will converge to the optimal ones. The selection criteria here is the estimation error produced by each model.

There are two main subparadigms in EAs for parameter optimization [Gij07, Fre02]:

- Genetic Algorithm (GA)
  GA operates in the *genotype* realm [4], and the individuals are usually represented in binary form. GA emphasizes on crossover, where parents are exchanging part of their chromosome to produce an offspring. A mutation could as well be done by flipping the bits in the chromosome with certain probability.

[4] Genotype distinction is a paradigm in evolution and inheritance of traits study focusing on organism's hereditary information

- Evolution Strategy (ES)
  ES on the other hand operates in *phenotype* realm [5], with real value individual representation. To generate the offsprings, ES emphasizes on mutation according to a probability distribution.

ES will be employed for hyperparameters selection, due to its nature of application for real-value representation, which could be easily applied to the problem [Gij07].

Algorithm 1 Pseudocodes for the Evolution Strategy $(\mu,^{+}\lambda)$

Require: $\mu > 0$ and $\lambda > \mu$ % μ and λ are the sizes of parent and offspring population
  $P \Leftarrow$ initialize(μ) % create initial random population
  $P.evaluate\,()$
  while isNotTerminated **do**
    $O \Leftarrow P.reproduceandmutate\,(\lambda)$ % create offspring population
    $O.evaluate\,()$
    if usePlusStrategy() **then**
      $O \Leftarrow O \bigcup P$ % combine parent and offspring populations
    end if
    $P \Leftarrow O.select\,(\mu)$
  end while

As shown in the pseudocodes of ES Algorithm 1, there are two options in ES, (μ, λ) and $(\mu + \lambda)$. In (μ, λ)-ES selection is done only within the newly generated offsprings population, and this means that an individual's lifetime is limited to one generation. On the other hand, $(\mu + \lambda)$-ES allows a potential individual with high fitness to survive multiple generations, because the selection's domain is the combined parents and offsprings populations.

The mutation for the real-value is typically implemented as the normal distribution around the object individual with the mean of 0 and standard deviation σ, although other distribution may be used as well.

$$x_i' = x_i + N_i\,(0, \sigma_i) \tag{2.29}$$

The σ value determines the step of mutation, and users need to specify σ_i for each parameter in the chromosome. This is an additional parameter selection task for hyperparameters tuning, therefore, it has to be simplified. One needs an automatic approach to update the mutation's step, so that the search range could be narrowed along with the increasing generation. The first round should have a relatively big step size, then it should be gradually reduced during the evolutionary process. The σ-tuning mechanism is called *self adaptation*, because it is done by embedding σ in the chromosome (x, c). This chromosome could be updated into a new one (x', c') by applying 2.29 and

$$\sigma_i' = \sigma_i \exp\left(\tau' N\,(0,1) + \tau N_i\,(0,1)\right) \tag{2.30}$$

[5] Phenotype distinction focuses on organism's observed properties, such as morphology or behavior

The $N(0,1)$ denotes a random value which is identical for each object parameter in the chromosome, while $N_i(0,1)$ is a random value specific for each distinct object parameter. The τ and τ' are the so-called *learning parameters*, and they are constants determining the rate of self adaptation. These learning parameters could be chosen according to [Sch81]

$$\tau \propto \frac{1}{\sqrt{2\sqrt{m}}} \tag{2.31}$$

$$\tau' \propto \frac{1}{\sqrt{2m}} \tag{2.32}$$

where m denotes the number of object parameters.

In general, all the values given by users are called the *strategy parameters*. The one which needs to be updated during the evolution, namely σ, is called *endogenous strategy parameter*, while those which need to be determined only once in the initialization phase, such as $\mu, \lambda, \tau, \tau'$, are called *exogenous strategy parameter*.

Chapter 3

Implementation

3.1 Experiment Methodology

The general scheme for supervised learning in Figure 2.1 is also applicable for SVR. Nonetheless, before using SVR to predict an outcome, one should select the suitable kernel and hyperparameters first. Therefore, an intermediary validation step is needed before the test phase. Test and validation are basically the same assessment, the difference is just the aim and the sequence in the learning scheme. In the hyperparameter selection phase, several rounds of learning and *validation* are needed, in order to indentify the most suitable hyperparameters for each proposed kernel. The fitness of hyperparameters is then compared among each other based on the predicted outcome's deviation, i.e. by their root-mean-squared-error (RMSE) on the validation set. Thus, the *best hyperparameters* for each kernel can be found. Afterwards, they will be used to train the machine, and the resulting model will be *tested* in the final test.

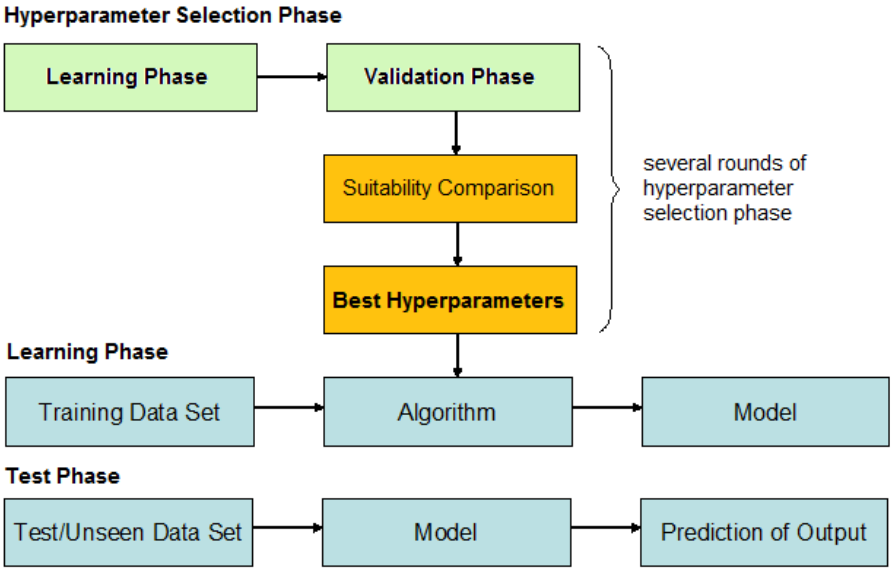


Figure 3.1: SVR Learning Scheme with Hyperparameters Selection

The whole experiment is done twice with the same data set, but different independent variables inclusion. The first part uses only continuous variables (see V3 .. V6 in Table 3.1) to build a regression model for the future price of second hand cars. The second part utilizes the whole continuous and binary variables. The aim of employing different complexity here is, to observe the relevance of the binary factors in improving the fitness of the model. Meaning, how much decrease in error prediction can be achieved by including additional binary variables.

3.1.1 Data Preparation

This experiment has been done using data of a certain model of a high quality cars, whose nature tend to loose value in a slower rate than other economic ones. The data comprises 180 columns and 124,386 samples without missing input. The data structure is given in Table 3.1.

The dependent variable is the re-sell value of second hand cars in the form of percentage from their original manufacture price (V2). The original data contains several independent variables with continuous values, such as the car's lifetime, kilometer driven and number of previous owners, as well as multinomial values (having more than two discrete values), such as model year, paint color and other optional equipments.

Valid samples		**124,386**
Columns		**180**
	V1	ID No.
	V2	Price Percentage
	V3	Lifetime Month
	V4	1/10,000 Km Driven
	V5	Days Sold since 1.1.1999
	V6	No.of Prev. Owner
	V7 .. V9	Customer Group
	V10	Tax
	V11 .. V17	Transaction Type
	V18 .. V27	Type of Car Use
	V28 .. V33	Model Year
	V34 .. V57	Car Series
	V58 .. V86	Paint Color
	V89 .. V103	Cushion
	V104 .. V180	Optional Equipments

Table 3.1: Summary of Second Hand Vehicles Data

Multinomial to Binary Variables Transformation
The use of independent multinomial variables in regression analysis and SVR is not justifiable, since they can cause a non-linear effect. Therefore, the data in this format need to be converted into several *dummy binary variables* before they can be used in linear regression. A binary variable only has two values, thus, only a linear relationship could be build out of it. Variable Paint Color, for example, has to be broken down into 29 variables (V58..V86),

such as variable black, variable silver, etc. with value of true (1) or false (0).

Independent Variables Standardization
Table 3.3 shows the basic descriptive analysis of the dependent and continuous independent variables. One can directly see that the values in those continuous variables are much larger than the rest of 174 binary variables. According to [Sar09], the contribution of an input will depend heavily on its variability relative to other inputs. If one input has a range of 0 to 1, while another input has a range of 0 to 1,000, then the contribution of the first input to the regression function will be swamped by the second input. Thus, it is essential to normalize the inputs so that their variability reflects their importance, or at least is not in inverse relation to their importance. In this experiment normalization to the same range approach is used, as suggested by [HCL08]. *Normalizing* here means scaling the values of those continuous variables by their minimum and maximum, to make all elements lie in a range of [0, 1], similar to those binary variables' range.

	Minimum	Maximum	Mean	Std. Deviation
Price Percentage	5.1819	98.5611	57.8186	16.1792
Lifetime Month	1	1,220	36.80	22.253
1/10,000 Km Driven	0.1	99.9	5.143	4.7403
Days Sold since 1.1.1999	2	2,066	850.68	527.169
No.of Prev. Owner	1	9	1.32	0.564
Valid N (listwise)	124,386			

Table 3.2: Descriptive Analysis from Data's Continuous Variables *before* Standardization

	Minimum	Maximum	Mean	Std. Deviation
Lifetime Month	0	1	0.0294	0.0183
1/10,000 Km Driven	0	1	0.0505	0.0475
Days Sold since 1.1.1999	0	1	0.4112	0.2554
No.of Prev. Owner	0	1	0.0400	0.0705
Valid N (listwise)	124,386			

Table 3.3: Descriptive Analysis from Data's Continuous Variables *after* Standardization

Data division
As there is a huge number of samples at disposal, they have to be arranged to suit the SVR learning scheme. Obviously the experiment needs a bigger portion of data in the learning phase than in the test phase. As a rule of thumb, one could use 70% - 30% division. Therefore the samples are divided as in Table 3.4.

Total samples		124,386
70%	for hyperparameter selection phase and for final training phase	87,070
	70% training data	60,949
	30% validation data	26,121
30%	for final test	37,316

Table 3.4: Data Division

Data shuffling
One important thing to note here is the data shuffling prior to data division. Since this data is documented over a long period, which is reflected in the increasing ID number, there is a possibility of different time-wise-characteristics. If one build the model by taking samples from certain time frame, and then test it against another time frame with quite a large time lag, the model could not perform well. Therefore, one should randomize the order, to assure that the resulting three sub-group have the same behaviour.

3.1.2 Kernel Selection

In addition to standard linear SVR, two other kernels are assessed in this experiments, namely polynomial and RBF kernel. Both are the classical kernels that perform well in many cases [HKK06]. Before choosing the right kernel, it is important to measure the nature of the data, i.e., the distribution of the dataset. According to [AS03], if the data distribution is normal, then the Gaussian or RBF kernel is recommended to be used, otherwise the polynomial kernel. At the final round the results of different kernels will be compared, which one is most suitable for the nature of data set, and at the same time checking the validity of this suggestion.

3.1.3 SVR Learning Curve

The main experimental part in SVR is to find suitable hyperparameter values through several rounds of model building, as depicted in Figure 3.1. For example, the iterations needed for grid search with a set of m cost values and n epsilon values is $m \times n$ times, with an avarage computation effort of t. Thus, the duration for grid search would be approximately $m \times n \times t$. The actual time for each round depends heavily on size of data processed. The more data SVR has to crunch, the more time is needed to build a model and test it. Therefore, the computation effort has to be minimized. Since SVR learning is based on support vectors, it is actually enough to learn from a partial data set, in order to generate a model. Thus, one would want to find out the required minimum training set size that can generate just a *nearly* optimum model, to speed up the search.

The purpose of learning curve is to understand the sensitivity of SVR, with respect to training data set size, as suggested by [PPS03, WT07]. This is done by running SVR several times with increasing sample numbers, and monitoring the learning effect that it gains from additional data, by measuring the RSME. A percentage of error decrease or learning improvement, will be calculated as well.

$$RMSE_{decrease} = \left(RMSE_{i+1} - RMSE_i\right) / RMSE_i \tag{3.1}$$

A threshold for $RMSE_{decrease}$ is to be set. Any learning improvement below the threshold is not regarded as substantial anymore. Thus, the last data set size used for training is considered big enough for further experiments.

In this learning curve experiment, 60,949 samples are crunched by SVR step by step. Each round adds 500 more data, and the threshold value is 0.1%.

3.1.4 Grid Search Analysis

For linear SVR, the hyperparameters are the cost and the epsilon value. The grid search is done in rather fine steps with an increasing exponent to the base of two. Cost values are set to $\{2^{-6}, 2^{-5}, \cdots, 2^{15}\}$, and epsilon values to $\{2^{-9}, 2^{-8}, \cdots, 2^{9}\}$.

For SVR with polynomial and RBF kernel, only a coarse grid search will be conducted, because there is an additional parameter, which increases the number of combinations to be tested. Thus, a larger interval is used to select good values of cost and epsilon. The ranges are the same as for the linear one, but each with 2^3 exponential increase: $C = \{2^{-6}, 2^{-3}, \cdots, 2^{15}\}$ and $\varepsilon = \{2^{-9}, 2^{-6}, \cdots, 2^{9}\}$. An equal step is also used for the gamma values $\{2^{-15}, 2^{-12}, \cdots, 2^{3}\}$ for SVR with RBF kernel. Whereas for SVR with polynomial kernel, degrees $\{2, 3, 4, 5, 6\}$ are tested.

3.1.5 Evolution Strategy

In ES the initialization values for reproduction and selection are as follows:

- plus comma: a selection strategy that determines from which population the selection is to be done. Comma strategy uses only offspring population as domain for the next generations, while plus strategy uses a combination of parents and offsprings population. In this experiment, both strategies are applied and observed.
- parents population size: a fixed exogenous strategy parameter which is set to 3.
- parents offsprings ratio: used to get the offsprings population size, it is set to 4.
  offsprings population size = parents population size × parents offsprings ratio
  The offsprings population size, which is 12 in this experiment, is a fixed exogenous strategy parameter.
- standard deviation: a dynamic endogenous strategy parameter, that determines the step size for updating object parameters while reproducing the offsprings.

While the parents population of the first generation (initial values) are set as follows:

1. SVR with linear kernel
   - C: 106.5544, 106.5544, 106.5544, initially the three parents are set to the same value, as suggested in equation 2.28
   - ε: 0.25, 1, 4
2. SVR with polynomial kernel
   - C: 106.5544, 106.5544, 106.5544
   - ε: 0.25, 1, 4
   - degree: 2, 3, 4
3. SVR with RBF kernel

- C: 106.5544, 106.5544, 106.5544
- ε: 0.25, 1, 4
- γ: 0.125, 1, 8

While the stopping criteria used is number of maximum generation, which is 15.

3.1.6 Final Training and Testing

In this phase, the hyperparameter candidates for the three different kernels suggested by grid search and evolution strategy will be compared against each other. The best one, i.e. the one with the lowest error, will be taken as the hyperparameter setting for the final experiment. The whole samples will be used: 87,070 data for training the machine, and the rest 37,316 for testing the model.

3.2 Experiment Tools

Software

This experiment has been run with the following softwares:

- Matlab® for its ease of use and efficiency in matrix calculation.
- LibSVM, a library for SVM written by Chang and Lin [CL01] in C with an interface to Matlab.
- SPSS® for its comprehensive result on statistical experiments.

Hardware

A same platform accross nine computers has been set up for these experiments with the following attributes: Intel Pentium 4, 2.4 GHz CPU, with a 2 GB main memory and Microsoft Windows Server 2003 Service Pack 2 operating system.

Chapter 4

Analysis

In this chapter, the experimental results will be presented, and then discussed mainly from the RMSE viewpoint, and to some extent the training time. This chapter is organized according to the methodolody described in the previous chapter, starting with the learning curve, then proceeding to the result from hyperparameters selection with grid search and evolution strategy. The summary of the hyperparameter search result can be observed from Table 4.4, and at last, the result of final experiment using the chosen hyperparameter setting will be given.

4.1 Learning Curve

In this section, the result from three different curves are presented, for SVR with linear, polynomial and RBF kernels. All kernels used the same standard setting of $C = 1$ and $\varepsilon = 1$, while degree is set to 2 for polynomial SVR and $\gamma = 0.25$ for RBF SVR.

One can see from Figure 4.1, that there is a steep decrease in RMSE by adding 500 samples in the first 10,000 data. This holds for all three kernels. The decreasing trend is shown with less gradient until the curves end. Nevertheless, a decision about minimal learning data set size cannot be based upon an optical view of the curves. Therefore, a threshold value of 0.1% will determine at which point the learning improvement is not considered important anymore. Table 4.1 suggests that the minimum training set sizes are 27,500 for linear SVR, 22,000 for polynomial SVR degree 2 and 32,500 for RBF SVR.

However, one needs the same training data set for all SVR kernels in order to make a fair evaluation for further experiments. Thus, a maximum number of them is chosen, which is 32,500. From this training set size, a number for validation set size of 14,000 is derived, with respect to 70%-30% proportion.

SVR Kernel	**No.of Iter.**	**Min.Train Data**	**Train Dur.(sec)**	**RMSE**
Linear	55	27,500	91	8.4918
Polynomial deg.2	44	22,000	76	10.5736
RBF	65	32,500	262	8.6585

Table 4.1: Minimum Training Set Size Derived from Learning Curve Experiment

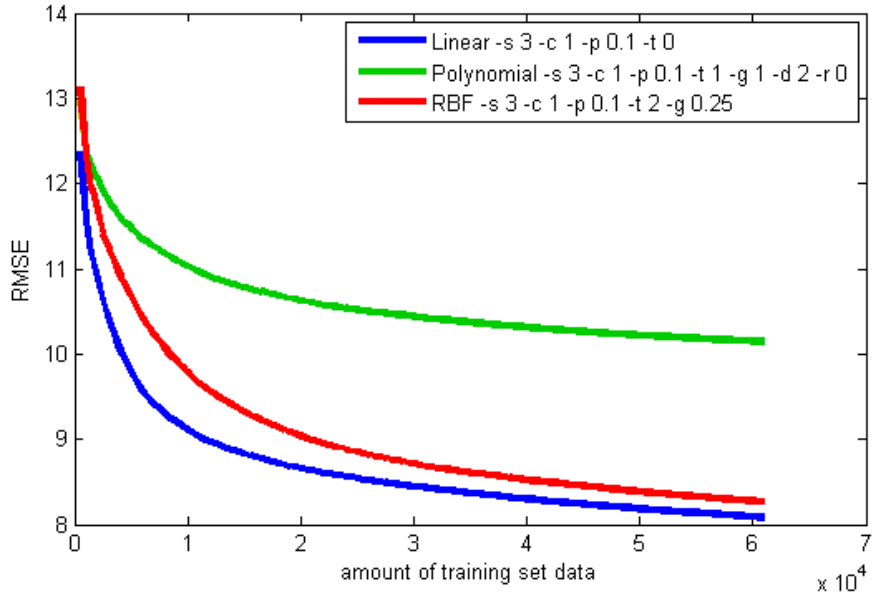


Figure 4.1: Learning Curve of SVR with Linear, Polynomial and RBF Kernels

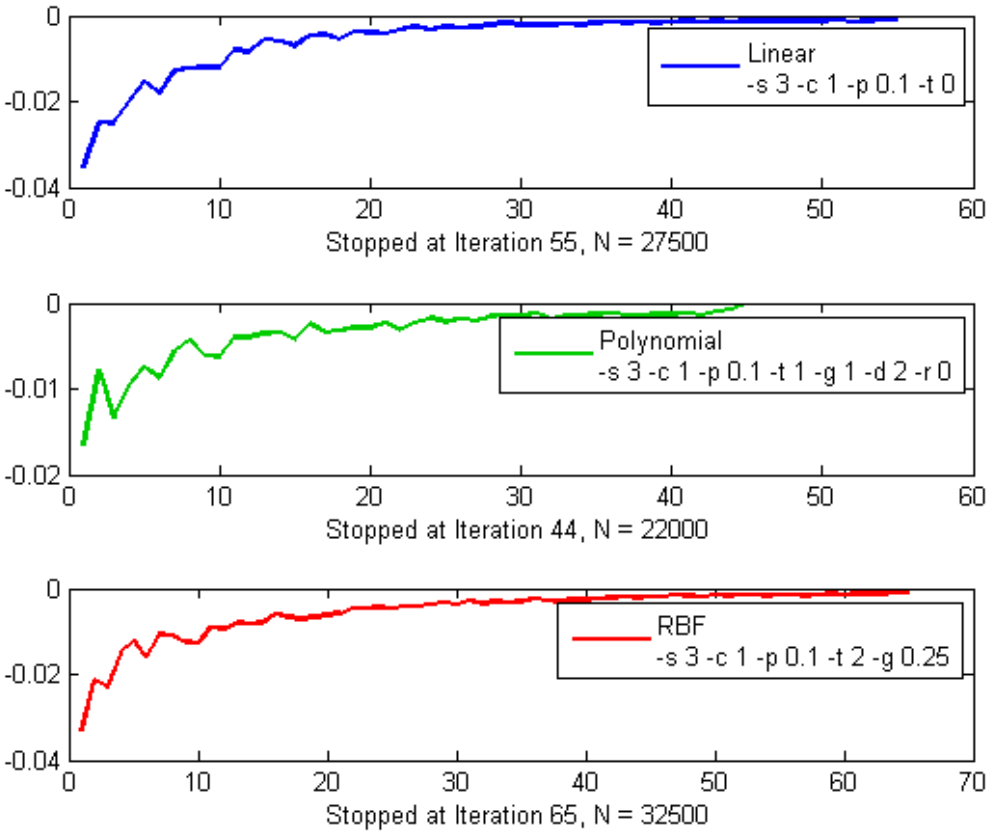


Figure 4.2: Learning Improvement of SVR with Linear, Polynomial and RBF Kernels

Training Duration						
t(5,000)	**t(10,000)**	**t(20,000)**	**t(30,000)**	**t(40,000)**	**t(50,000)**	**t(60,949)**
9.46	18.25	36.85	60.43	79.58	98.18	119.75
16.16	32.25	64.56	100.26	134.49	162.43	191.48
37.24	74.98	147.31	229.79	303.85	372.55	463.22

Table 4.2: Training Duration for Different Data Set Sizes (in Seconds)

Table 4.2 shows the training duration needed for the increasing data set sizes. The duration increases linearly along with the data size, e.g. SVR with RBF kernel takes 37.24 seconds for processing 5,000 data, and approximately four and eight times longer for 20,000 and 40,000 data respectively. Therefore, one can expect, that the time saving by reducing data set size from 60,949 to 32,500 is $1 - (32,500/60,949) \approx 46\%$. Important to note here, the time taken is based on solely four continuous independent variables.

4.2 Grid Search Analysis

In grid search analysis, SVR is trained and validated for each parameters combination to choose good values by looking at the lowest error generated. This section is divided into two subsections. The first subsection focuses only on the continuous variables, while the second subsection elaborates the results from the experiment with the whole continuous and binary variables.

4.2.1 Grid Search with Continuous Variables

SVR with Linear Kernel

For linear SVR, a grid search with 22 cost and 19 epsilon values is executed. The number of iterations needed to do this grid search is 418 times. Figure 4.3 displays the validation RMSE for each combination of cost and epsilon. As one could already predict, RMSE goes down along with increasing cost and decreasing epsilon. The underlying reason could be studied from equation 2.11 and 2.14. A higher cost penalizes the empirical training error (non-zero ξ_i, ξ_i^*) more, so that the model built is less under-fit. While lower ε defines a lower insensitivity loss function, which reflects a higher learning capacity for a better model fitting.

In this experiment, the RMSE minimum value of 7.45 is achieved with $C = 2^{14} = 16,384.00$ and $\varepsilon = 2^3 = 8.00$. Actually, there is an area of good RMSE ≤ 8.00 for cost values of $\{2^2, 2^3, \cdots, 2^{15}\}$ and epsilon values of $\{2^{-9}, 2^{-8}, \cdots, 2^3\}$. Even an RMSE ≤ 7.50 can be achieved with cost values starting from $2^6 = 64$, without changing the epsilon range.

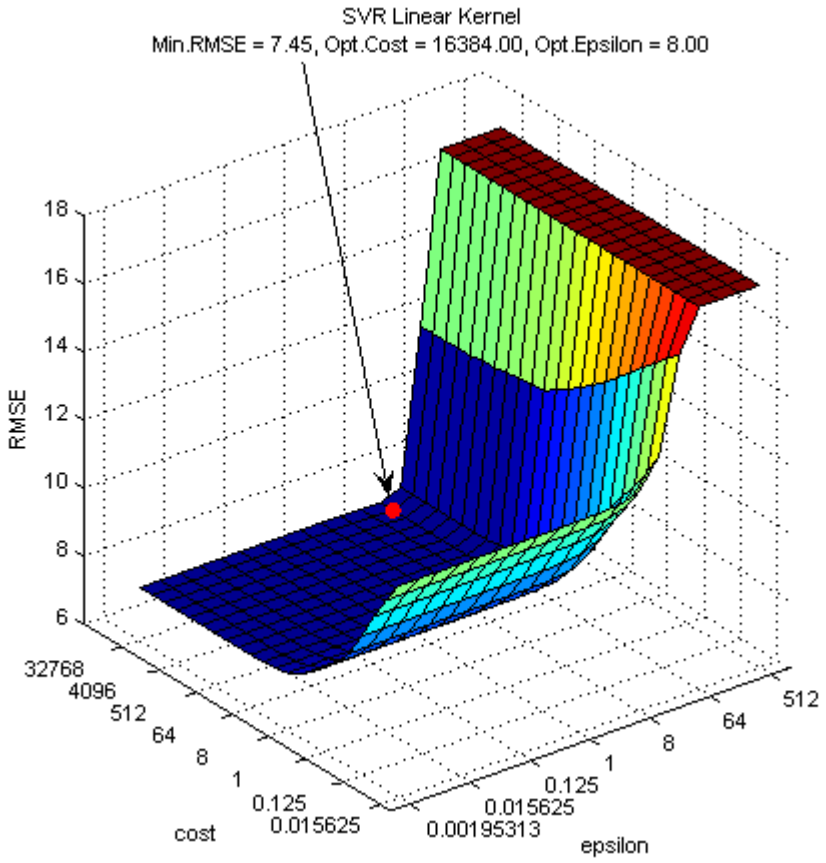


Figure 4.3: Grid Search for SVR with Linear Kernel for Models with Continuous Variables

Figure 4.4 depicts the training time based on different cost and epsilon. It shows that there is a high increase in training duration for a small epsilon and big cost combination. To understand which parameter plays more significant role, a comparison between their impact is made. Mean values of training time with diverse epsilon and same cost are counted to disclose the cost effect, and vice versa with diverse cost and same epsilon for epsilon effect. The result is presented by Figure 4.5. The cost plotting has a linear curve, and the epsilon plotting has a logarithmic one.

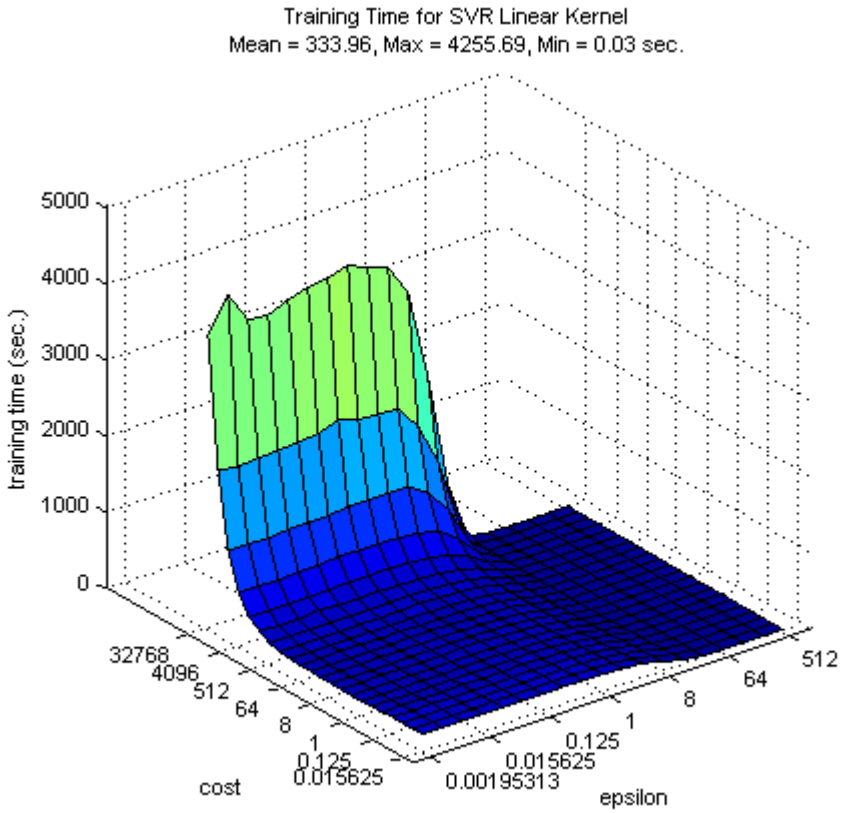


Figure 4.4: Training Time for SVR with Linear Kernel for Models with Continuous Variables

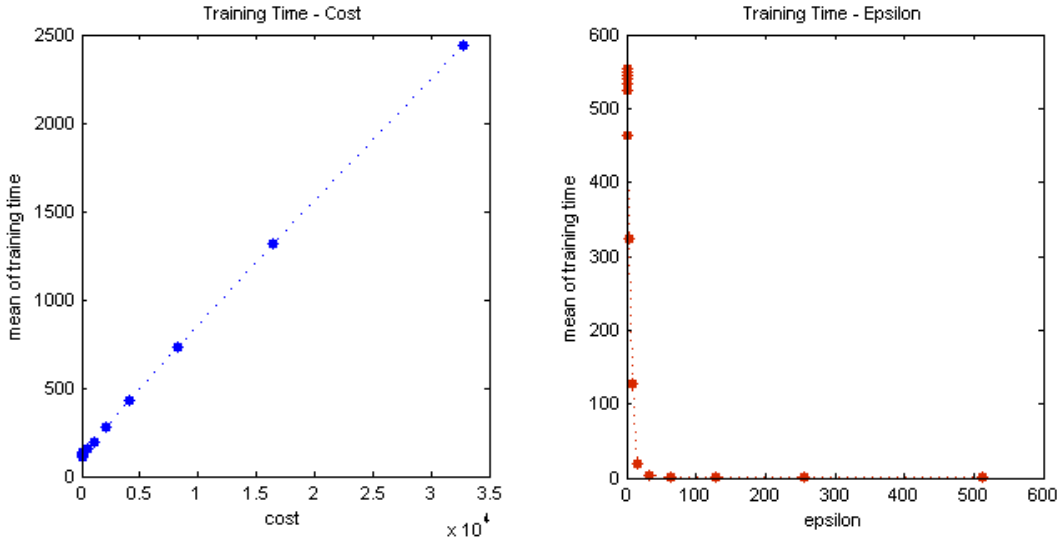


Figure 4.5: Cost and Epsilon Effect on The Training Time of SVR with Linear Kernel

From the RMSE and the training time outcome, there are two findings:

1. The band of *good enough* parameters can be found in quite a large range. Therefore a finer grid search is not necessary for model selection, it is better to do just a coarse search in order to reduce computational effort and time.

2. The training time depends on both epsilon and cost values, besides training data size. Nonetheless, in this particular case, the epsilon shows much higher influence. The lower the epsilon, the longer duration it takes, since more learning effort is needed.

SVR with Polynomial Kernel

The grid search for polynomial SVR is done with five different degrees, eight cost and seven epsilon values in coarse steps of 2^3. The total of iteration is 280 times. Figure 4.6 displays five plots of RMSE with respect to all degrees experimented. The similar minimum RMSE of 6.97 is achieved for three settings. First, $d = 3, C = 512.00, \varepsilon = 8.00$, second, $d = 4, C = 512.00, \varepsilon = 8.00$, and third, $d = 5, C = 64.00, \varepsilon = 8.00$. One thing to mention here, the value of epsilon which generates a minimum RMSE for all degrees is 8.00.

Moreover, a higher degree of polynomial is more sensitive to the change of cost. One can observe this effect in the increase of RMSE for $C = 32,768.00$ and $\varepsilon = \left\{2^{-9}, 2^{-6}, 2^{-3}, 2^{0}\right\}$ which are $\approx \{0.01, 0.1, 1, 2, 20\}$ for degree $\{2, 3, 4, 5, 6\}$. While building a model, SVR with a high value of cost combined with a high learning capacity reduces the empirical training error more than it is needed to achieve the best model. This leads to over-fitting, and therefore, the generalization error (RMSE) for the validation set goes up.

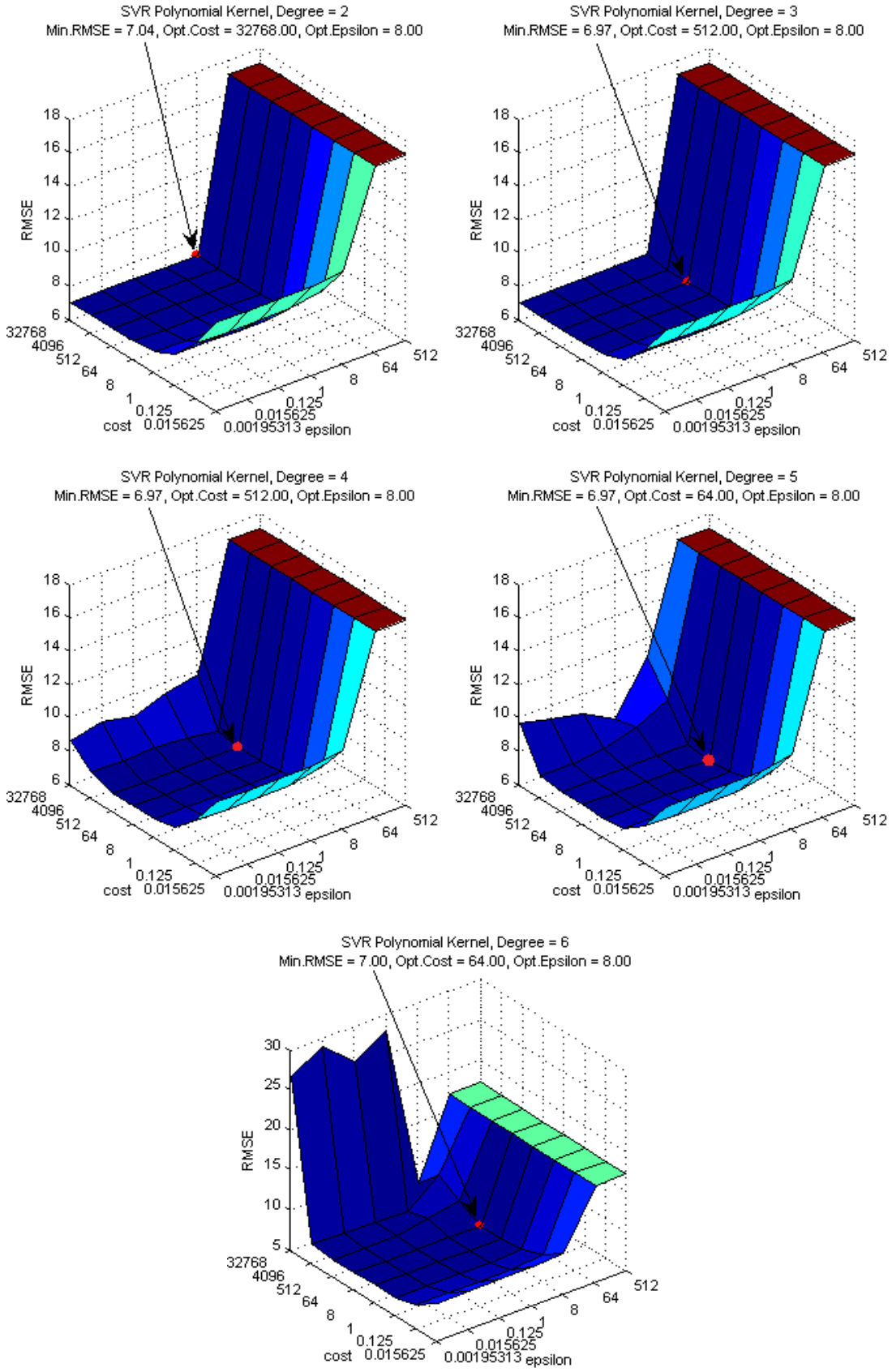


Figure 4.6: Grid Search for SVR with Polynomial Kernel for Models with Continuous Variables, Degree = {2, 3, 4, 5, 6}

SVR with RBF Kernel

SVR with RBF kernel is run with 392 combinations of eight cost, seven epsilon and seven gamma values. The minimum error of 6.88 is accomplished with the following hyperparameters, $C = 512.00, \varepsilon = 1.00, \gamma = 8.00$, as shown in Figure 4.7.

Whereas Figure 4.8 shows, that increasing gamma settings $\{2^{-15}, 2^{-12}, 2^{-9}, 2^{-6}, 2^{-3}, 2^{0}\}$ generate decreasing minimum RMSE $\{8.11, 7.53, 7.33, 7.07, 7.00, 6.93\}$. The area of C and ε combinations that gives RMSE level ≤ 8 also gets larger. This figure exhibits model improvement as γ increases.

An explanation for this has to be searched, because the theory states the opposite. The RBF kernel can also be written as

$$k(x, z) = \exp\left(-\frac{\|x - z\|^2}{2\alpha^2}\right) \tag{4.1}$$

The α determines the area of influence of a support vector over data space. A large α allows a support vector to have a strong influence over a larger area, so that SVR can build a smoother regression model. Therefore, as α increases, the value of generalization error will decrease [Chi98a]. From Equations 4.1 and 2.27, $\gamma = 1/\left(2\alpha^2\right)$. Thus, one should expect that as γ decreases, the value of generalization error will decrease.

One can prove this theory by doing further experiment with larger gamma values, because obviously the trend of decreasing RMSE for increasing gamma will have to *stop* at a certain turning point.

Another interesting finding from Figure 4.8 is the value of cost that minimizes the RMSE. For different $\gamma = \{2^{-15}, 2^{-12}, 2^{-9}, 2^{-6}, 2^{-3}, 2^{0}\}$, and consequently diverse epsilon values, the optimum cost is $2^{15} = 32768$, the largest cost available in this grid search. The best parameters combination, in contrast, uses only a considerably low cost of $2^{9} = 512.00$ (see Figure 4.7). This indicates that, with the right setting of gamma and epsilon to determine the machine learning capacity, a very high cost to penalize the empirical error is *no longer needed* in order to achieve a low generalization error.

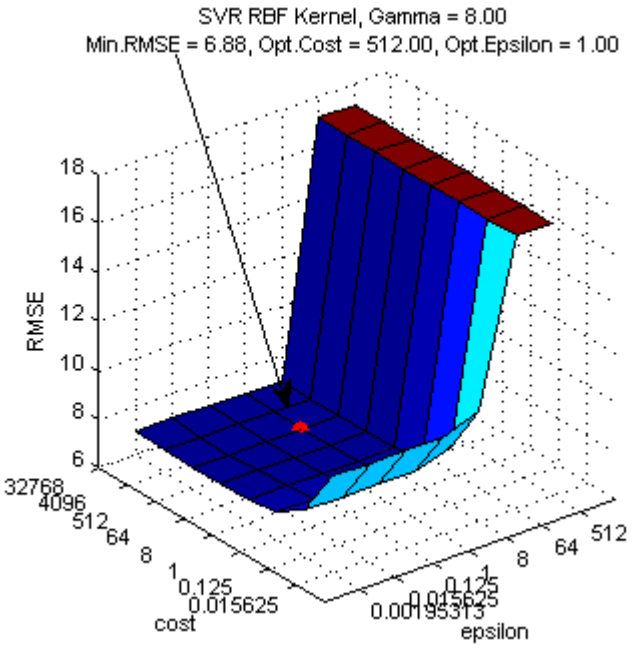


Figure 4.7: Grid Search for SVR with RBF Kernel for Models with Continuous Variables, Gamma = 2^3

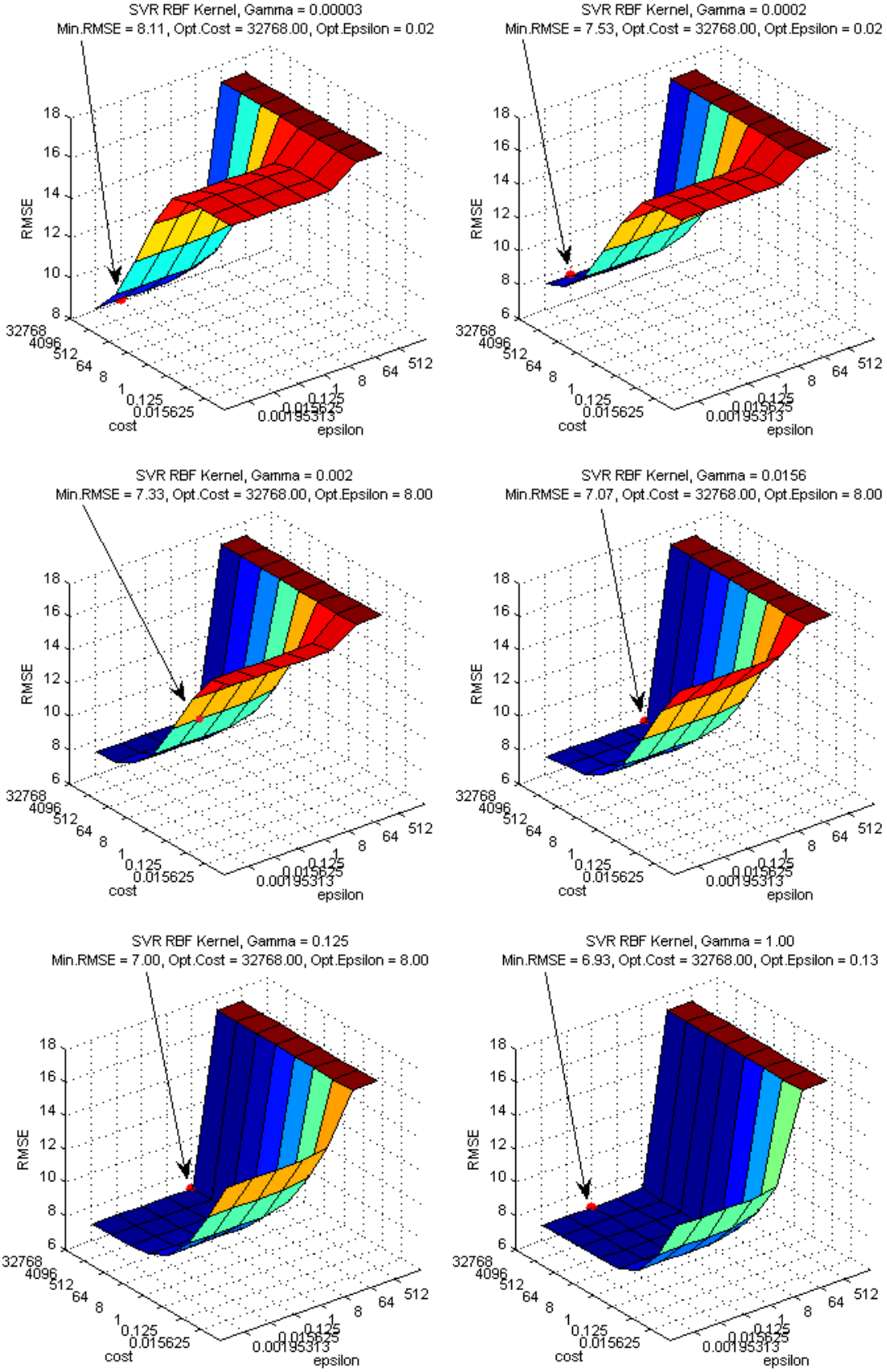


Figure 4.8: Grid Search for SVR with RBF Kernel for Models with Continuous Variables, Gamma = $\{2^{-15}, 2^{-12}, 2^{-9}, 2^{-6}, 2^{-3}, 2^{0}\}$

4.2.2 Grid Search with All Variables

SVR with Linear Kernel

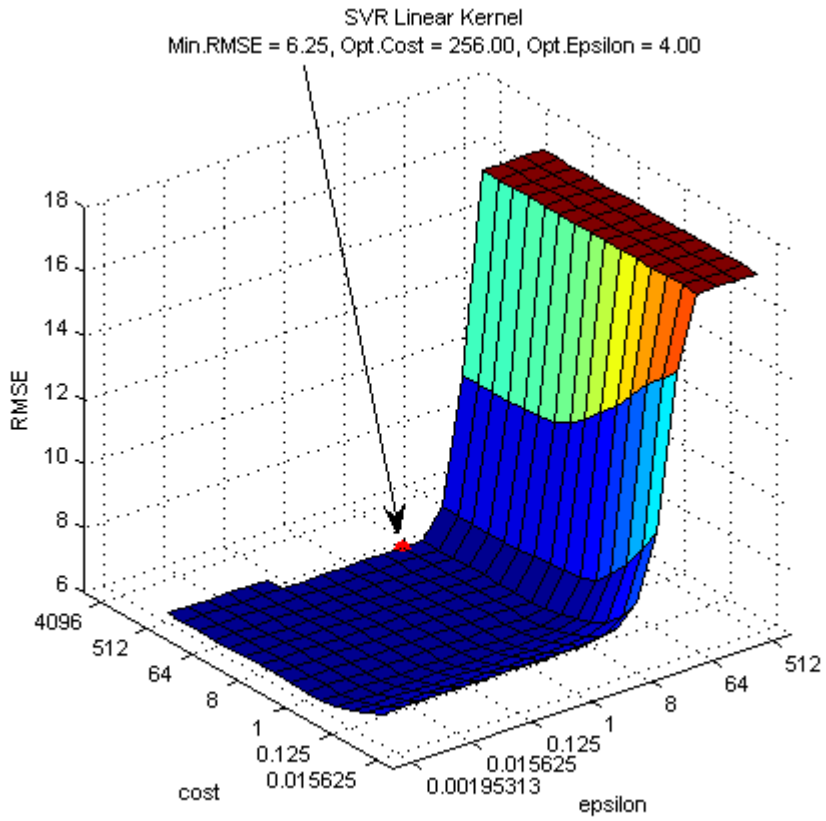


Figure 4.9: Grid Search for SVR with Linear Kernel for Models with All Variables

The grid search for linear SVR with all variables is done for 16 cost and 19 epsilon values. Actually there are 22 cost values to be evaluated, but at the time this thesis is written, only 15 have been executed completely and one partially. Due to time limitation, after 708 hours (30 days), the result has to be fetched before the experiment is finished. Thus, one can only learn from 291, instead of 418, cost and epsilon combinations. Figure 4.9 displays the available RMSE outcome.

There is a general decrease in RMSE in comparison to the result of grid search for linear SVR with only continuous variables. Taking the same range of cost $\{2^{-6}, 2^{-5}, \cdots, 2^{8}\}$ and epsilon $\{2^{-9}, 2^{-8}, \cdots, 2^{9}\}$, the mean of RMSE with all variables is 9.0924, while with continuous variables 10.9042. The minimum RMSE is also smaller, 6.25 in compare to 7.45. Thus, the increase of input dimension by including the binary variables yields a 16% decrease in RMSE.

The optimum values of cost and epsilon are 256.00 and 4.00 respectively. Parallel to the finding in grid search for continuous variables, there is a large area of good RMSE ≤ 6.5 for $C = \{2^{-3}, 2^{-2}, \cdots, 2^{8}\}$ and $\varepsilon = \{2^{-9}, 2^{-8}, \cdots, 2^{2}\}$.

Besides that, the whole plot for linear SVR with continuous and all variables are showing the same tendency. They both experience a leap in RMSE value for the epsilon values above $2^5 = 32.00$. This is due to the radius of the insensitivity tube which is too big. As the result, the model is under-fitting, and the empirical, as well as generalization error, goes up.

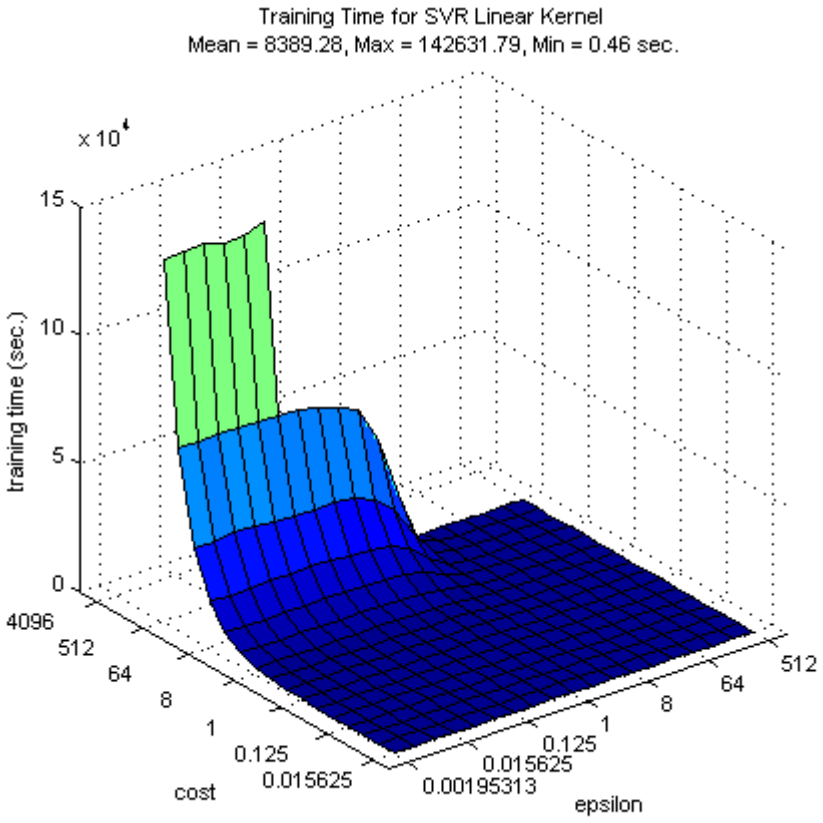


Figure 4.10: Training Time for SVR with Linear Kernel for Models with All Variables

As in any other circumstances, where the complexity drives up the problem solving duration, in this case, adding 174 binary variables leads to a longer training time. Although the experiment with all variables has not covered the grid area with big cost values completely $\{2^9, 2^{10}, \cdots, 2^{15}\}$, the mean of training period has soared to more than 25 times the one with continuous variables only. If the training time for $C = 2^9$ and $\varepsilon = 2^{-4}$ epsilon values is almost 40 hours, then with the linear effect of cost to time increase (see Figure 4.5), one should anticipate, that the single calculation for $C = 2^{15}$ and the same epsilon will be solved in 708 hours (30 days) by the same computer.

Unfortunately, such a long computation does not entail low generalization error due to over-learning. Therefore, it has to be avoided. This is exactly why evolution strategy is prefered over grid search for hyperparameters selection. ES could find reasonably good parameters without being trapped in the over-fit cases, as what the grid search faces. One can always limit the ES search either by maximum generation number, maximum allowed training time or RMSE improvement threshold.

SVR with Polynomial Kernel

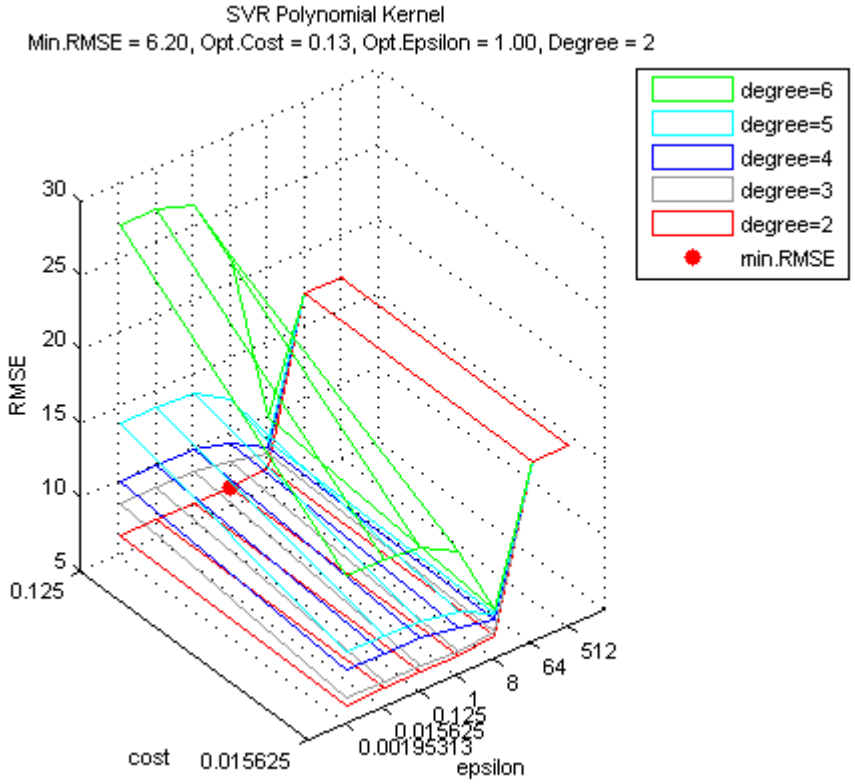


Figure 4.11: Grid Search for SVR with Polynomial Kernel for Models with All Variables

Epsilon	0.0020	0.0156	0.1250	1.00	8.00	64.00	512.00
Cost	**Degree = 2**						
0.015625	6.3307	6.3303	6.3286	6.3215	6.6502	17.1804	17.1804
0.125	6.2451	6.2441	6.2395	6.1955	6.4233	17.1804	17.1804
	Degree = 3						
0.015625	6.7518	6.7498	6.7354	6.6376	6.8701	17.1804	17.1804
0.125	8.4401	8.4340	8.3870	8.0600	7.4283	17.1804	17.1804
	Degree = 4						
0.015625	8.6715	8.6643	8.6085	8.2350	7.6348	17.1804	17.1804
0.125	9.9408	9.9295	9.8408	9.2128	7.8344	17.1804	17.1804
	Degree = 5						
0.015625	9.8817	9.8714	9.7918	9.2751	8.0278	17.1804	17.1804
0.125	13.9080	13.8825	13.6666	12.2153	8.2264	17.1804	17.1804
	Degree = 6						
0.015625	15.0006	14.9681	14.7456	13.3869	8.3505	17.1804	17.1804
0.125	27.2258	27.1393	26.4392	21.8533	9.8126	17.1804	17.1804

Table 4.3: Grid Search Evaluation with Polynomial SVR for Models with All Variables

After 36 days running SVR with polynomial kernel to look for the best hyperparameters, there are only 70 cost - epsilon - degree combinations available for analysis. Fortunately, the minimum RMSE attained from this set is quite low, thus, the parameters setting is good enough to be used in the final experiment.

As one can already notice in Figure 4.11, the RMSE is smaller for the lower polynomial degree. In accordance with the learning from grid search with continuous variables, a higher polynomial degree is wigglier to the change of cost, as suggested by the RMSE of polynomial SVR degree 5 and 6. Moreover, the effect of epsilon setting to under-fitting can be observed in Table 4.3. With epsilon values larger than 64.00, the RMSE is the same, 17.1804, for all cost values and degrees tested.

The lowest RMSE of 6.20 is found for a mix of low cost, epsilon and degree ($C = 0.125, \varepsilon = 1.00, d = 2$). The improvement in term of RMSE, in compare to the one with continuous variables only, is 11%.

SVR with RBF Kernel

In the experiment with RBF kernel, the evaluation of different cost, epsilon and gamma has been more thoroughly done, although it does not manage to build the models with the last cost setting of $C = 2^{15} = 32768$. This is due to the excessively long computation to generate SVR model with a mixture of large cost, low epsilon and approriate gamma setting (appropriate here means that the gamma is likely to yield the minimum RMSE). For example, the training time for the SVR with $C = 2^{15}, \varepsilon = 2^{-9}, \gamma = 2^{-9}$ is approximately 155 hours (6.5 days).

Moreover, it is expected that a very large cost is not necessary to gain a low generalization error, when the right setting of gamma and epsilon is found, as indicated by the finding in the RBF SVR experiment with continuous variables. Thus, the present result with $C = \{2^{-6}, 2^{-3}, \cdots, 2^{12}\}$, and 87.5% coverage of the grid search area, by running 343 out of 392 parameter combinations, can be considered as complete.

The lowest RMSE of 6.07 is achieved for $C = 64.00, \varepsilon = 1.00, \gamma = 0.0156$, as one can see in Figure 4.12. Unlike the result from the experiment with continuous variables, where the chosen gamma (2^3) is at the end of its range, with all variable, the chosen gamma (2^{-6}) is in the middle of its range. Figure 4.13 shows that the increasing gamma values between 2^{-15} to 2^{-9} *decreases* the minimum RMSE, whereas the values between 2^{-3} and 2^3 *increases* it. This is coherent to the previous assumption in the analysis of the experiment with continuous variables, where the decreasing trend of RMSE along the increasing value of gamma cannot continue forever.

Parallel to the previous results, by including the binary variables, the SVR model accuracy can be increased by almost 12% in compare to the one with continuous variables only.

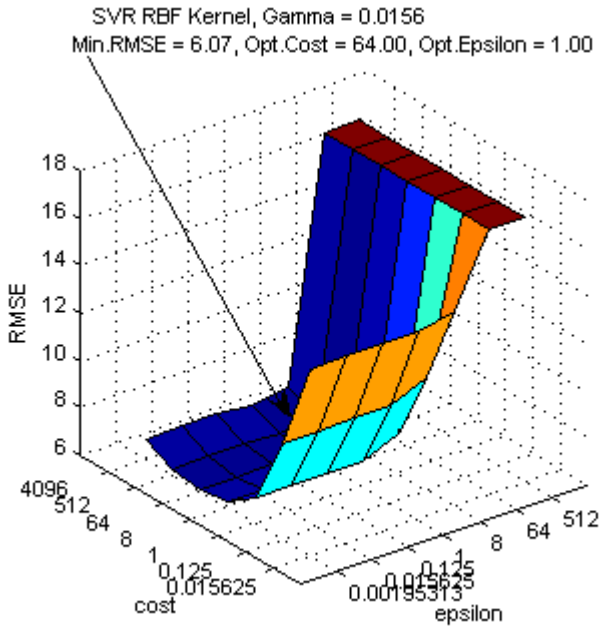


Figure 4.12: Grid Search for SVR with RBF Kernel for Models with All Variables, Gamma $= 2^{-6}$

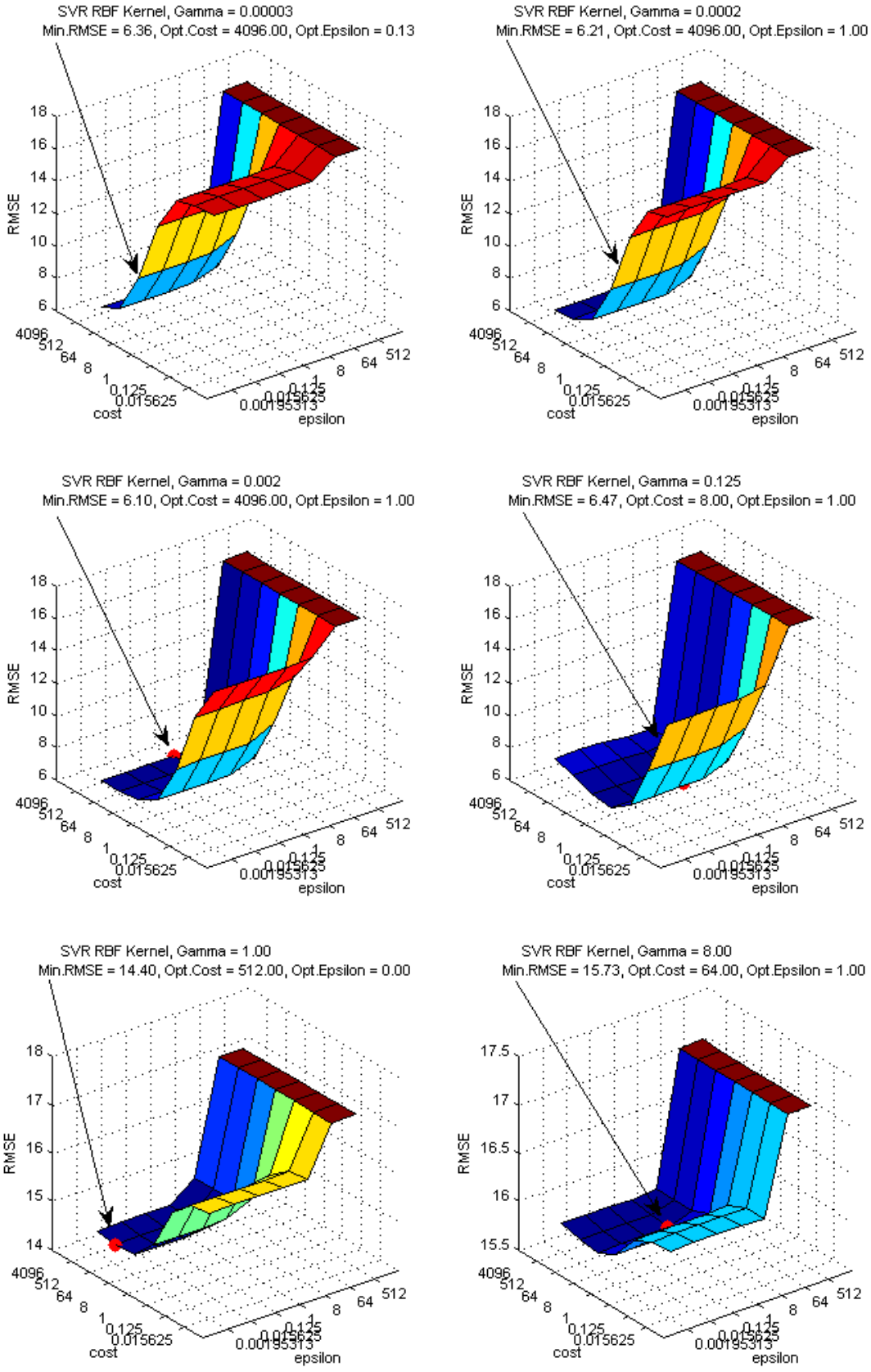


Figure 4.13: Grid Search for SVR with RBF Kernel for Models with All Variables, Gamma $= \{2^{-15}, 2^{-12}, 2^{-9}, 2^{-3}, 2^{0}, 2^{3}\}$

4.3 Evolution Strategy

In the following, the results of hyperparameter selection with ES will be presented. As in the grid search analysis, this section is also divided into two subsections, namely ES with continuous variables and ES with all variables.

There are two kinds of plots that are utilized to describe the results here. The first one is the proliferation figure, which displays the mutation of hyperparameter candidates through the evolution process of ES. The green points depitc all individuals generated through mutation. The blue circles are the selected candidates, which are passed on to the next generation, and the big red dot is the chosen individual with highest fitness.

The second type of figure is the box plot, which is used in order to depict the RMSE spread of individuals in successive generations. Each structure in this plot summarizes the RMSE minimum, lower quartile (Q1), median (Q2), upper quartile (Q3), RMSE maximum and the outliers for each generation.

4.3.1 Evolution Strategy with Continuous Variables

SVR with Linear Kernel

The results achieved by performing plus and comma strategy for the linear SVR with continuous variables are similar. Both RMSEs are converging to 7.44 and epsilon to 10. These values are confirming the result from grid search. However, the optimum costs chosen show quite a difference, 112.69 and 207.26 (grid search result is $16,384$), but with minor consequence on error discrepancy. This is due to the inferior role of cost and dominant role of epsilon in reducing error, as one can see in the grid search before.

As shown from Figure 4.14, ES converges quite quickly to the optimum epsilon value. From totally 15 generations, starting from the fifth generation for (μ, λ)-ES and the fourth for $(\mu + \lambda)$-ES, the *best epsilon values* in each generation steadily moving around 0.1% range from the optimum one. One thing need to be stated here, those values are selected because of their fitness, and not due to an overly decrease in standard deviation for mutation.

From Figure 4.17 one can learn, that generally, regardless of the kernel, the first few generations have higher RMSE median, which is decreasing along the increasing generation, due to the selection procedure. It is also common that early generations have larger interquartile range, which indicates larger spread. Furthermore, this figure exhibits that the self adaptation procedure for the mutation's step σ, does a good job to reduce the spread, and focuses on good parameter values instead.

SVR with Polynomial Kernel

As one could see from Figure 4.15, the minimum RMSE for both $(\mu + \lambda)$-ES and (μ, λ)-ES is 6.97. This is the same result achieved from grid search. The selected degrees in both strategies are 5, and the epsilon values are approximately 8.00. The values of cost chosen in ES-comma (130.06) and ES-plus (111.14) are also similar. Both are approximately twice as much the optimum cost found with grid search. Considering the big interval within grid

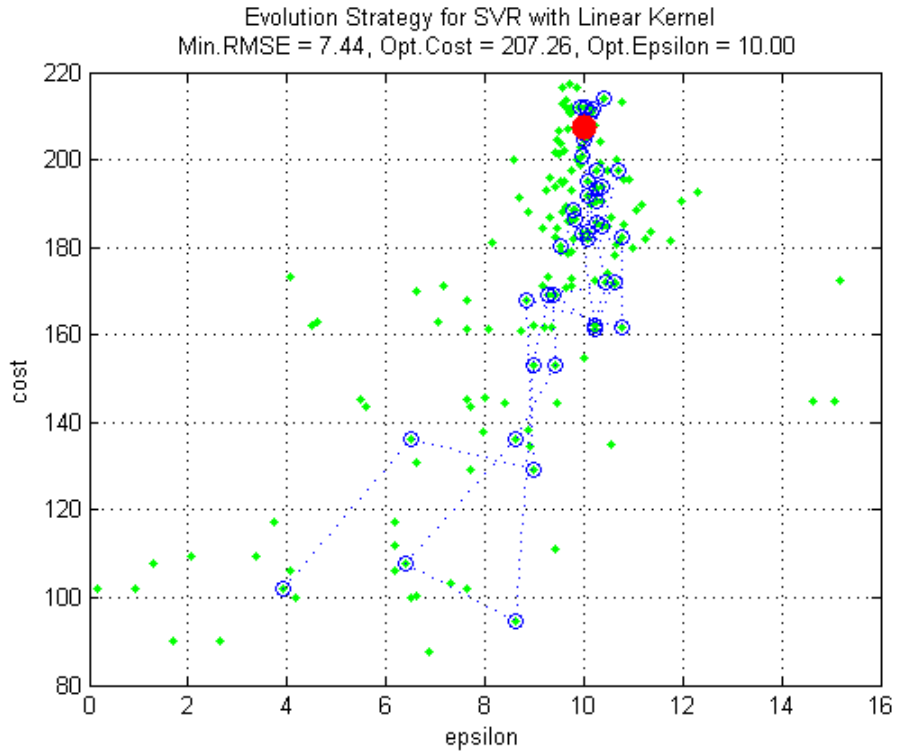
search (2^3), the cost values from ES and grid search have no major difference. Hence, all the best parameter combinations suggested by ES are synchron to the result from grid search analysis.

Another thing to mention here is the error spread during the evolution course. Figure 4.17 reveals that the RMSE spread of polynomial SVR in the first three generations is much larger than the RMSE variance for the other two kernels. This is mainly caused by some individuals with higher polynomial degrees (e.g. 6, 7), which do not fit to the data set nature, and thus, generate high generalization errors.

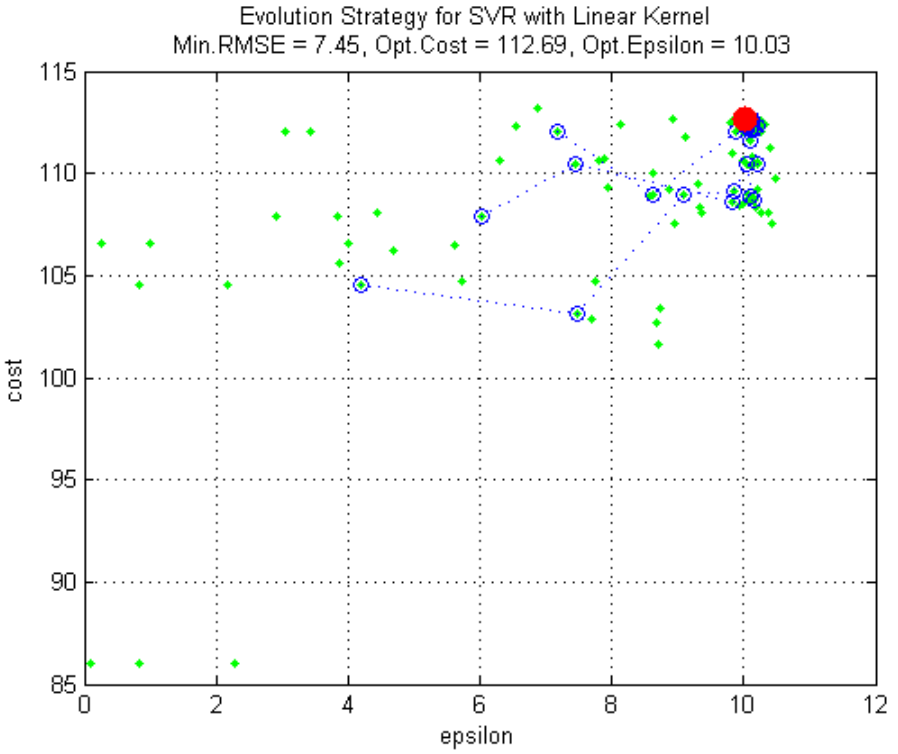
SVR with RBF Kernel

The resulting RMSE of 6.88 can be achieved with ES-comma, and 6.86 with ES-plus. Thus, the best parameters setting for RBF SVR as suggested by ES is $C = 197.72, \varepsilon = 2.93, \gamma = 12.18$ (see Figure 4.16).

Looking at the optimal parameters found in grid search (Figure 4.7), $C = 512.00, \varepsilon = 1.00, \gamma = 8.00$, again, the similarity of the results between the two approaches has to be drawn for the experiment with RBF SVR. A slight decrease of 0.02 in RMSE is achieved by loosening the epsilon tube a bit and reducing the cost. In the grid search analysis for RBF SVR, it is proposed to check higher values of gamma in order to find even a lower RMSE. ES does this search, and proves, that a little higher gamma can yield a lower RMSE.

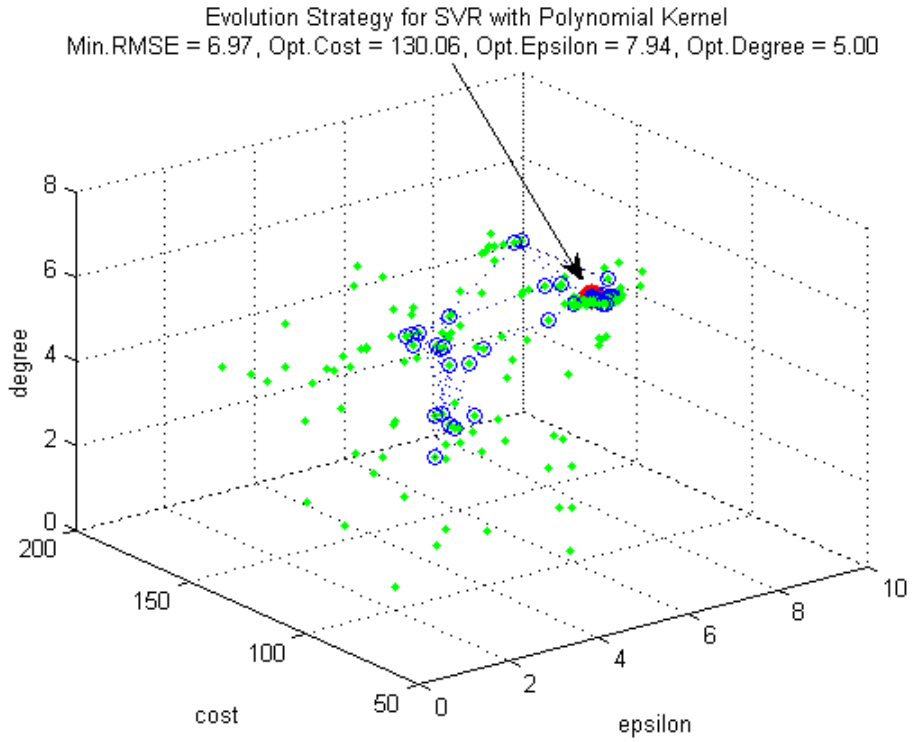


(a) (μ, λ)-ES selection

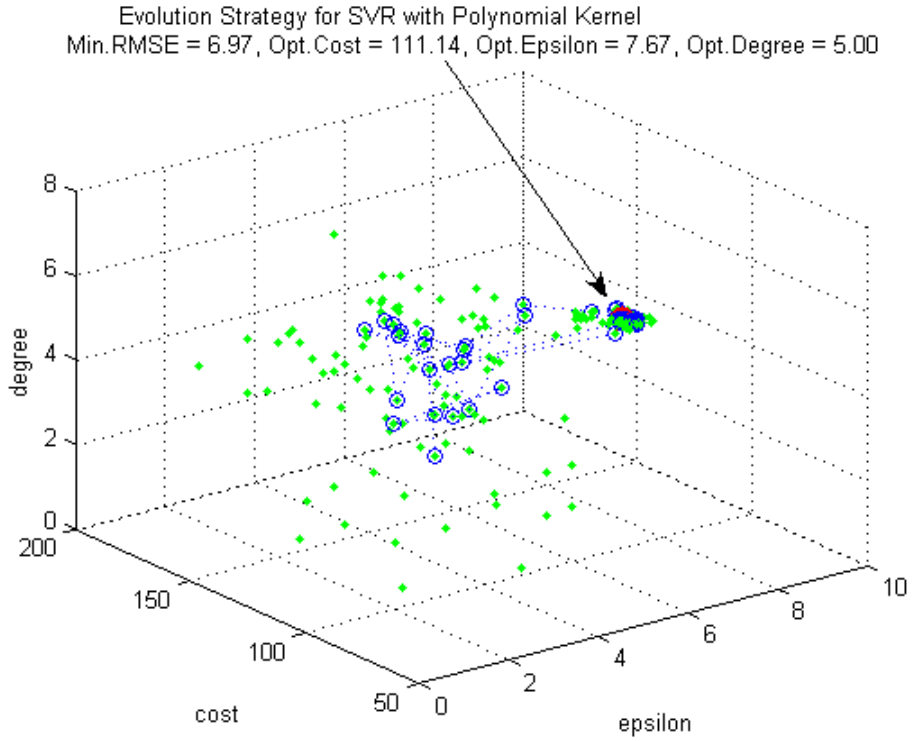


(b) $(\mu + \lambda)$-ES selection

Figure 4.14: Evolution Strategy for SVR with Linear Kernel for Models with Continuous Variables

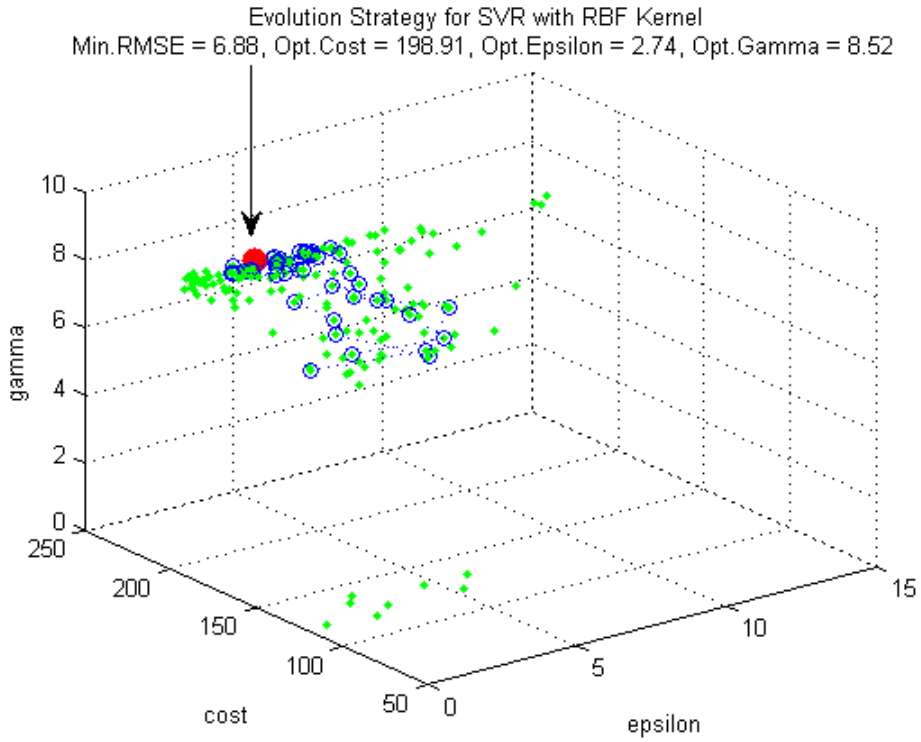


(a) (μ, λ)-ES selection

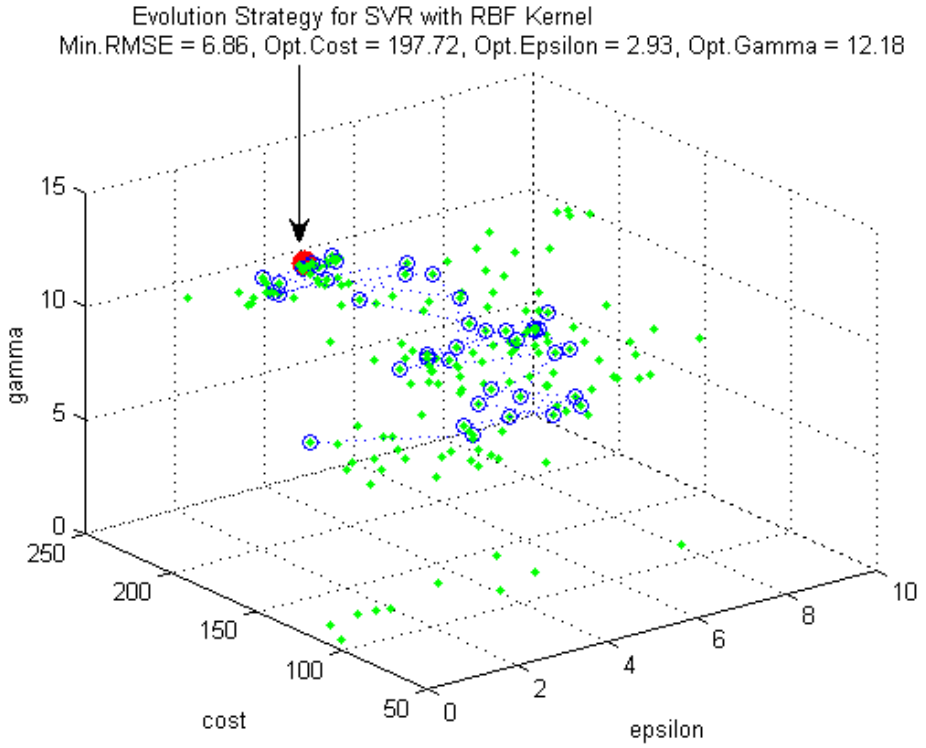


(b) $(\mu + \lambda)$-ES selection

Figure 4.15: Evolution Strategy for SVR with Polynomial Kernel for Models with Continuous Variables



(a) (μ, λ)-ES selection



(b) $(\mu + \lambda)$-ES selection

Figure 4.16: Evolution Strategy for SVR with RBF Kernel for Models with Continuous Variables

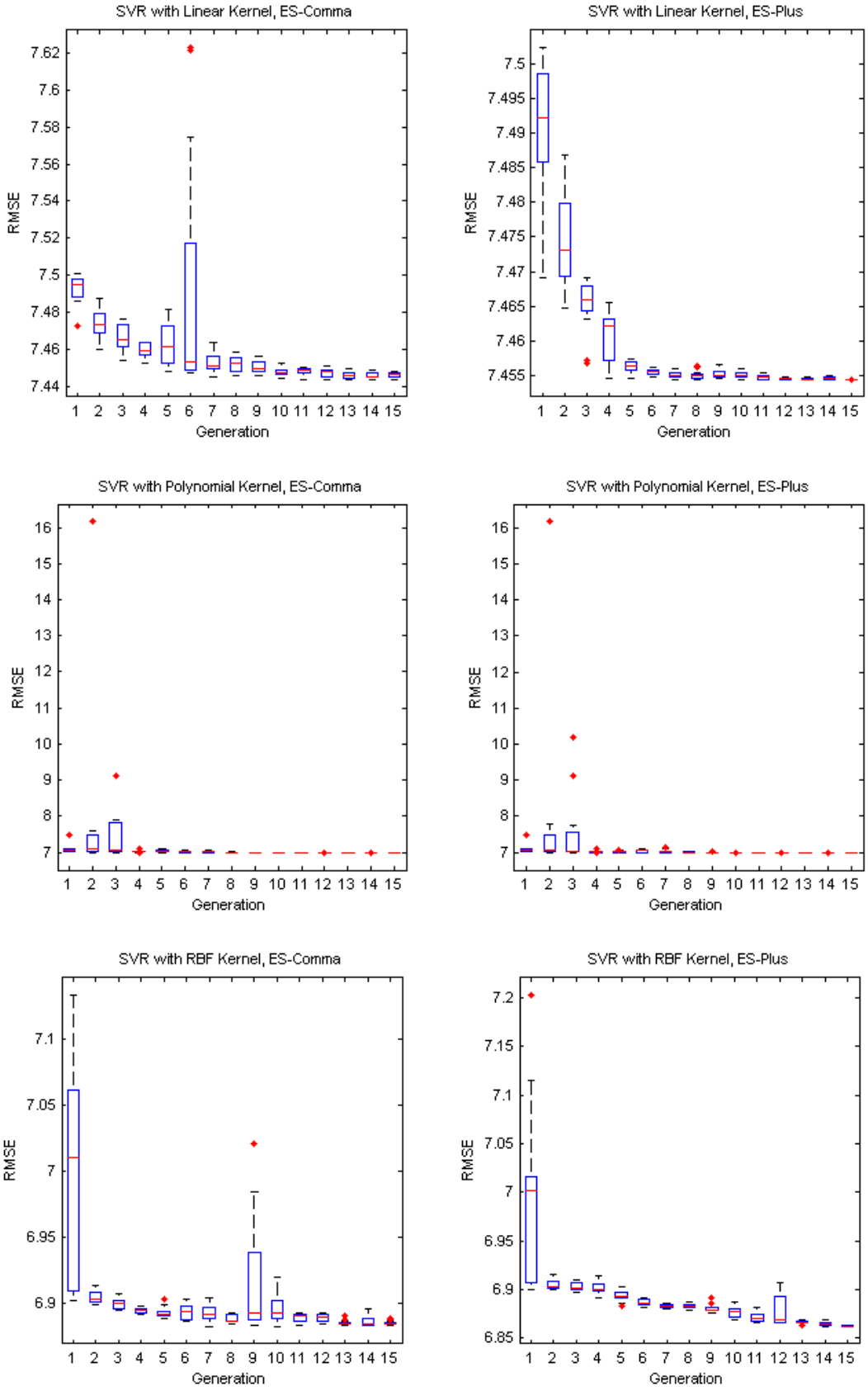


Figure 4.17: RMSE Spread of Individuals within Different Generations for Models with Continuous Variables

4.3.2 Evolution Strategy with All Variables

SVR with Linear Kernel

From the two graphs in Figure 4.18 one can see that both ES-comma and ES-plus find the same minimum RMSE of 6.24, and the same optimum cost (182.71) as well as epsilon value (6.42). This figure also displays a high similarity of both parameters proliferation (but not the same), although ES-comma and ES-plus are run in different platform. The same remark can also be made based on RMSE spread within different generations, as depicted by two graphs for linear SVR in Figure 4.20. This can happen because Matlab coindentally uses similar initial states for its *randn* function, to generate the random numbers that follows the normal distribution according to the ziggurat algorithm [Inc05].

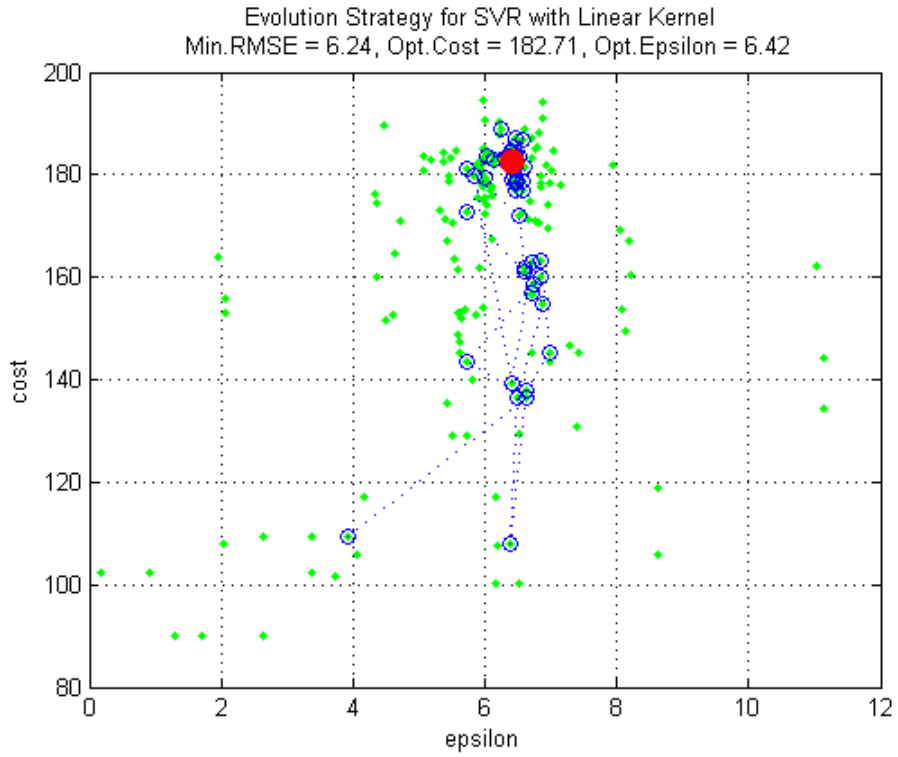
Besides the forementioned observation, there is only a minor decrease in RMSE, that has been achieved by both ES strategies in compare to the grid search result (6.25).

SVR with Polynomial Kernel

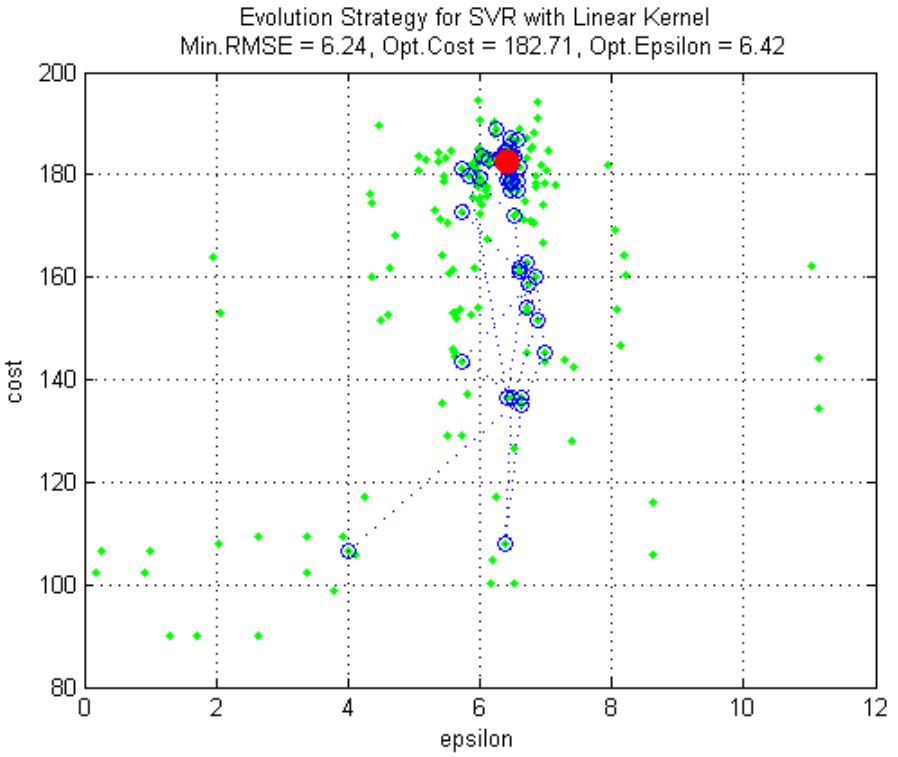
Unfortunately, the experiment with polynomial SVR for models with all variables has to excluded from this report. Due to an excessively long ES-search that takes approximately 3.5 days for a single parameters combination, there are only six models built in the time being. Thus, there is a lack of comprehensive result to be analyzed, and further be used as a basis for a decision of the best parameters setting.

SVR with RBF Kernel

The results from the ES experiment with RBF SVR are displayed in Figure 4.19. A slightly better RMSE is found by employing ES-plus strategy rather than ES-comma, which can only find the the same minimum RMSE as in the grid search. Not surprisingly, the best hyperparameter settings are also alike, especially for the gamma values (ES = 0.01, GS = 0.0156). ES' best radius for the epsilon tube is around three time larger than the best one according to grid searh (ES = 3.42, GS = 1.00), whereas the cost is twice as much (ES = 110.96, GS = 64.00). Considering the grid search steps, this variation is not so important.

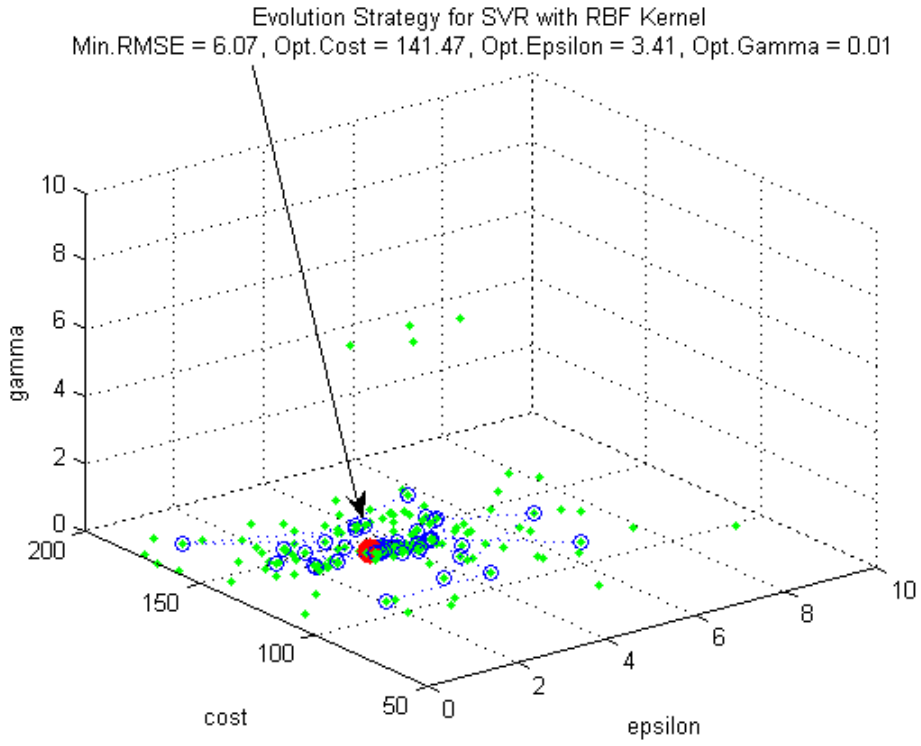


(a) (μ, λ)-ES selection

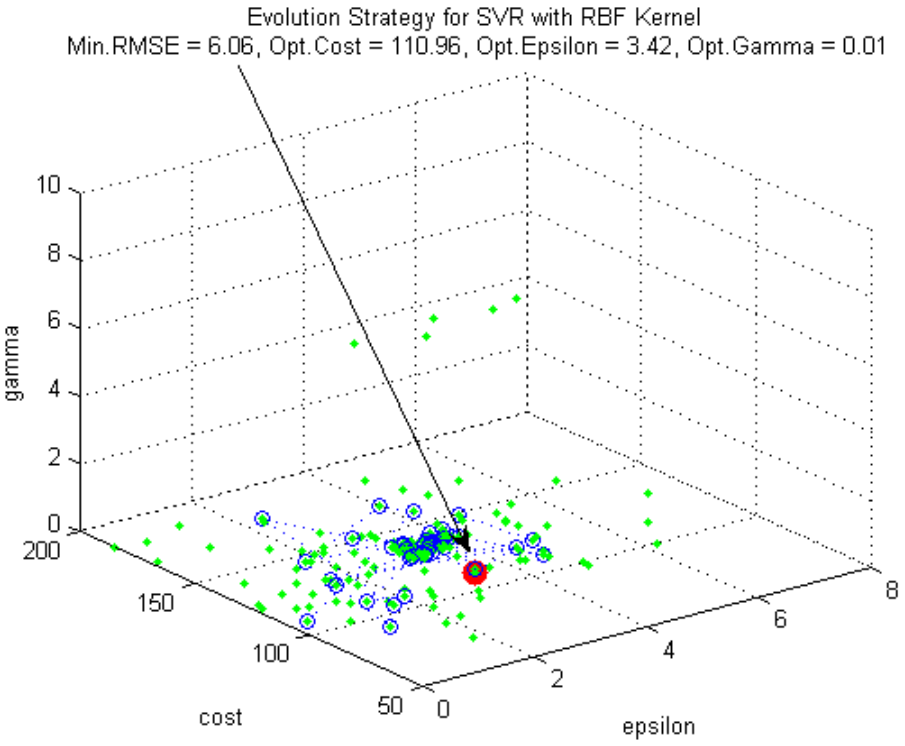


(b) $(\mu + \lambda)$-ES selection

Figure 4.18: Evolution Strategy for SVR with Linear Kernel for Models with All Variables



(a) (μ, λ)-ES selection



(b) $(\mu + \lambda)$-ES selection

Figure 4.19: Evolution Strategy for SVR with RBF Kernel for Models with All Variables

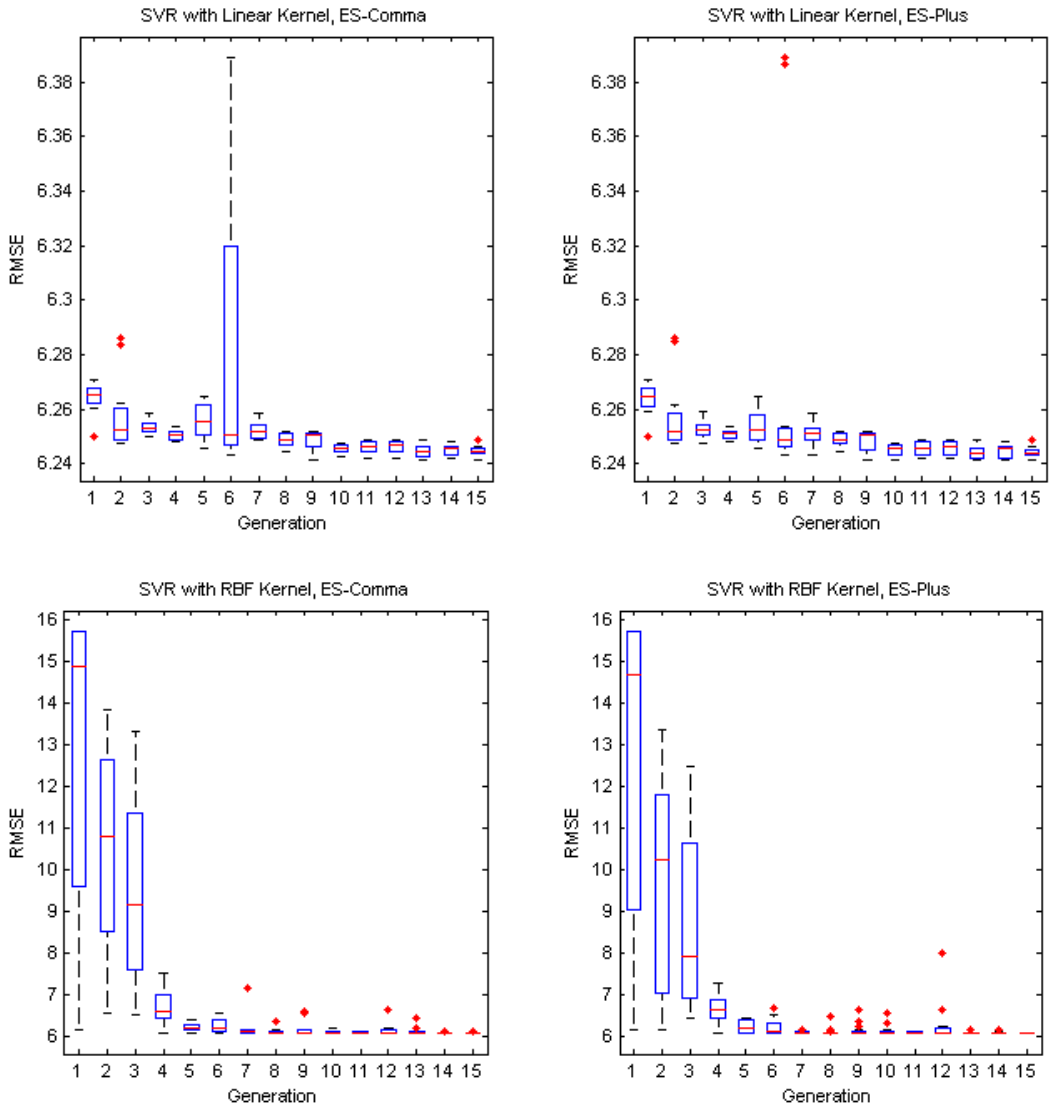


Figure 4.20: RMSE Spread of Individuals within Different Generations for Models with All Variables

4.4 Summary of Hyperparameters Selection

SVR Kernel		Min.RMSE	Hyperparameters Setting
With Continuous Variables			
Grid Search			
Linear Kernel		7.45	$C = 16,384.00, \varepsilon = 8.00$
Polynomial Kernel		6.97	$C = 512.00, \varepsilon = 8.00, d = 3$ $C = 512.00, \varepsilon = 8.00, d = 4$ $C = 64.00, \varepsilon = 8.00, d = 5$
RBF Kernel		6.88	$C = 512.00, \varepsilon = 1.00, \gamma = 8.00$
Evolution Strategy			
Linear Kernel	(μ, λ)	$12^{th}gen., 7.44$	$C = 207.26, \varepsilon = 10.00$
Linear Kernel	$(\mu + \lambda)$	$15^{th}gen., 7.45$	$C = 112.69, \varepsilon = 10.03$
Polynomial Kernel	(μ, λ)	$13^{th}gen., 6.97$	$C = 130.06, \varepsilon = 7.94, d = 5$
Polynomial Kernel	$(\mu + \lambda)$	$14^{th}gen., 6.97$	$C = 111.14, \varepsilon = 7.67, d = 5$
RBF Kernel	(μ, λ)	$10^{th}gen., 6.88$	$C = 198.91, \varepsilon = 2.74, \gamma = 8.52$
RBF Kernel	$(\mu + \lambda)$	$14^{th}gen., 6.86$	$C = 197.72, \varepsilon = 2.93, \gamma = 12.18$
With All Variables			
Grid Search			
Linear Kernel		6.25	$C = 256.00, \varepsilon = 4.00$
Polynomial Kernel		6.20	$C = 0.125, \varepsilon = 1.00, d = 2$
RBF Kernel		6.07	$C = 64.00, \varepsilon = 1.00, \gamma = 0.0156$
Evolution Strategy			
Linear Kernel	(μ, λ)	$15^{th}gen., 6.24$	$C = 182.71, \varepsilon = 6.42$
Linear Kernel	$(\mu + \lambda)$	$15^{th}gen., 6.24$	$C = 182.71, \varepsilon = 6.42$
RBF Kernel	(μ, λ)	$11^{th}gen., 6.07$	$C = 141.47, \varepsilon = 3.41, \gamma = 0.01$
RBF Kernel	$(\mu + \lambda)$	$7^{th}gen., 6.06$	$C = 110.96, \varepsilon = 3.42, \gamma = 0.01$

Table 4.4: Best Hyperparameter Candidates Suggested by Grid Search and Evolution Strategy

Before deciding the parameters for the final experiment, a comparison between hyperparameters achieved from different searching methods needs to be done. Table 4.4 summarizes all the forementioned results to facilitate this objective [1]. The criterion for selection is the lowest RMSE. The best hyperparameters are then listed in Table 4.6 and 4.7, and used for the final experiment.

There are several learning items from ES approach in comparison to grid search for different SVR kernels and variables inclusions from this table:

1. The best parameters combination from grid search learning can be *found* as well in an approximity by ES. This is due to the fact, that there is quite a large plateau in grid search with different parameter settings, which can generate low errors.

[1] The generation numbers accompanying the RMSE values in the ES division are the points of time where the minimum RMSE are detected along the evolution for ES-comma, or the earliest one in the case of ES-plus.

2. The resulting minimum RMSE from ES search is always better, or at least as good as the one found in grid search. This is due to the flexible search area of ES, which is exactly the advantage of ES over grid search. The grid search is limited to the parameter values set initially. Thus, one should do a two phase grid search, a coarse one and then a finer one, to improve the model accuracy. Moreover, one should also consider to check beyond the area of current search, if the minimum RMSE is achieved by a parameter value in the search range border. The ES approach is not curbed by these issues. To find the optimal parameters setting, ES can start the search from any point, and then move closer to the final solution. It also adapts its search level autonomously, so that it begins the search with a relatively coarse step, and then, it tunes itself to a finer search.

3. Although ES can start the search from any point, the initialization point is important to enable it to find the right parameters quicker. Small epsilon values of {0.25, 1, 4} and cost derived from Cherkassky and Ma's Equation 2.28 have been proven to be good starting points for SVR to assess this data set. For SVR with polynomial kernel, one can use low degrees such as {2, 3, 4} for the first parents. Whereas {0.125, 1, 8} can be employed to initialize gamma setting for SVR with RBF kernel.

4. Both (μ, λ)-ES and $(\mu + \lambda)$-ES generate a similar error level, with less then 0.3% RMSE difference for all kernels, independent of variables selection used to build the regression models. Besides, neither comma nor plus strategy can always yield the minimum RMSE for all kernels. These facts lead to a conclusion, that the results achieved by performing plus and comma strategy are the same for this particular huge data set.

Additionally, a summary of the training time from the hyperparameter selection tasks is displayed in Table 4.5, excluding the predicting time and other calculations. From the experiment with continuous variables which is completed according to the plan, one can see, that the time saving by using ES instead of GS is huge. For linear SVR, GS needs around 25 times longer duration than ES, whereas for polynomial and RBF SVR, GS takes 16 and 3 times longer period than ES.

One could probably get such a comparison for the training time in the experiment with all variables, if GS had been completed. Unfortunately, this is not the case. Therefore, Table 4.5 just shows the number of the SVR models successfully built for each kernel, and display the time accordingly. The note 'partially used results' means, that there are 2 or 3 models left out from being taken into the analysis, due to the lack of range comprehensiveness. These excluded results have either big cost setting, higher polynomial degree or appropriate gamma, which makes a single SVR calculation take an excessive time.

Despite the uncomplete GS result, one can easily notice that the number of models to be built for finding the best parameters with GS is higher than with ES. Thus, GS needs a longer search duration, which can be much worsened when SVR is trapped in some over learning areas. This trap will also blockade the evaluations of other parameters combinations, which have more probability to be the best hyperparameters. For example, the two excluded models in polynomial SVR take alone almost 240 hours to build. Therefore, GS is *not time efficient* for hyperparameter search, especially for high dimensional and huge data set.

SVR Kernel	Note	No.of Models	T_{train}(s.)	T_{train}(h.)
With Continuous Variables				
Grid Search				
Linear Kernel		418	139590	39
Polynomial Kernel		280	298550	83
RBF Kernel		392	114840	32
Evolution Strategy				
Linear Kernel	comma	183	6503	2
Linear Kernel	plus	183	4610	1
Polynomial Kernel	comma	183	17764	5
Polynomial Kernel	plus	183	17649	5
RBF Kernel	comma	183	35886	10
RBF Kernel	plus	183	33777	9
With All Variables				
Grid Search				
Linear Kernel		291	2441300	678
Polynomial Kernel	partially used results	70	2175300	604
	all results	72	3028300	841
RBF Kernel	partially used results	343	1611300	448
	all results	346	2180200	606
Evolution Strategy				
Linear Kernel	comma	183	1264200	351
Linear Kernel	plus	183	1260800	350
Polynomial Kernel	plus, unused results	6	1799000	500
RBF Kernel	comma	183	693810	193
RBF Kernel	plus	183	733880	204

Table 4.5: Comparison of Training Time from Grid Search and Evolution Strategy

4.5 Final Training and Testing

Table 4.6 shows the best parameter settings for each different kernel used in the final experiment with continuous variables only, whereas Table 4.7 displays the parameters employed in the experiment with all variables. The resulting RMSE are listed to choose which kernel and parameters setting is the best. The results of another generalization error measure, namely Squared Correlation Coefficient (SCC), are also recorded here, to strengthen the decision made by selecting the lowest RMSE. SCC (a.k.a. R squared) defines the correlation of variance between the predicted values and the given outcomes [KKMN98]. The SCC values vary from 0 to 1. The larger the SCC value, the better the model performance is in predicting a future outcome. In addition to the error measures, these tables exhibit the training duration used to build the respective SVR model.

From Table 4.6, one can learn, that the best kernel for the model with continuous variables in the used car price prediction application is the RBF. The following setting, $C = 197.72, \varepsilon = 2.93, \gamma = 12.18$, gives an RMSE as low as 6.9331, and an SCC of 0.8184. The linear SVR is following in the second rank with RMSE of 8.2701. Whereas SVR with

polynomial kernel degree 3 comes in the last place, with a big disparity of 66.5% to the result from RBF SVR. This is a proof that the polynomial SVR is *not suitable* for the particular data set, considering only the continuous variables.

Table 4.7 shows, that the result of the experiment with all variables reaffirms the above-mentioned kernel ranking. The best SVR kernel for this application is the RBF, with an RMSE less than 6 - the lowest recorded error ever, and an SCC of 0.8665. This is an improvement of 14.5% in compare to the best one with continuous variables only. The second place is given to the linear SVR, with the RMSE of 6.3917, but this is achieved with a much longer training time than the RBF SVR's. Whereas the high RMSE of polynomial SVR restates, that it is unproper to be applied for the second hand car data.

SVR Kernel	**Hyperparameters Setting**	**RMSE**	**SCC**	T_{train}**(sec.)**
Linear Kernel	$C = 207.26, \varepsilon = 10.00$	8.2701	0.7444	245.62
Polynomial Kernel	$C = 512.00, \varepsilon = 8.00, d = 3$	11.5450	0.4960	795.08
RBF Kernel	$C = 197.72, \varepsilon = 2.93, \gamma = 12.18$	6.9331	0.8184	1345.90

Table 4.6: Final Experiment with Continuous Variables

SVR Kernel	**Hyperparameters Setting**	**RMSE**	**SCC**	T_{train}**(sec.)**
Linear Kernel	$C = 182.71, \varepsilon = 6.42$	6.3917	0.8454	159295
Polynomial Kernel	$C = 0.125, \varepsilon = 1.00, d = 2$	15.3626	0.4657	8101
RBF Kernel	$C = 110.96, \varepsilon = 3.42, \gamma = 0.01$	5.9383	0.8665	26279

Table 4.7: Final Experiment with All Variables

Chapter 5

Benchmarking with Statistical Linear Regression

In this chapter the statistical models generated by SPSS will be presented and compared with the SVR approach. The model is built solely on continuous independent variables in the first section, whereas all variables are used in the second. They are built based on the same 87,070 cases, and tested against 37,316 unseen data to observe the generalization error. The error measurement which is used throuhgout the experiment to compare the fitness of regression models is the RMSE.

5.1 Experiment with Continuous Variables

For this experiment, SPSS calculates the regression line based on the four continuous independent variables only. In the following, the assumptions and the model fitness will be discussed according to the result.

The first step of regression analysis is to check the assumptions. The *normality* of data distribution could be inspected from descriptive statistics in Table 5.1. The skewness of V3, as well as kurtosis of V3, V4, V6 are beyond normal distribution range. Thus, the data set actually needs a pre-transformation before linear regression. However, this will *not* be done for the sake of fairness to SVR experiment, which has been performed beforehand. Table 5.2 exhibits the Pearson Correlations, which measure the *linearity* between each variables. These correlation coefficients are significant, as indicated by their significance level (0.000), since the probability of obtaining result as the one observed is very small. The correlation indexes of V3 and V4 to the dependent variable V2 are bigger in comparison to V5 and V6, thus, one could expect a stronger influence from V3 and V4 in the regression equation. The last presumption to be checked is *multicollinearity*. Table 5.6 suggests that the model has no problem with collinearity, because the condition indexes are well below 15 [Inc06].

Table 5.4 displays the result of an analysis of variance on the mean value of the dependent variable, which comes from two sources, the regression and the residual. The sum of squared value of the regression is comparatively larger than the residual's, and it means that the model accounts for most of variation in the dependent variable. This is supported

by the significance value of the F statistic, which is very small (0.000), indicating that the independent variables can explain the variation well.

Table 5.3 exhibits the summary of the model fitness. R shows the correlation between the observed and predicted values of dependent variable, while R squared indicates the proportion of variation in the dependent variable explained by the regression model. Since R squared tends to optimistically estimate it, the adjusted R squared is calculated to more closely reflect the goodness of model's fit in the population [Inc06]. Due to the huge samples size, both values are very close to 0.7689. The standard error of the estimate in this table, 7.7657, is also known as RMSE. It is derived from the residual mean square, i.e. MSE, in Table 5.4.

The coefficients of the regression model can be found in column B (Unstandardized Coefficients) of Table 5.5. Using the model

$$V2_{pred} = 81.116 + (-424.262 * V3) + (-126.711 * V4) + (-10.058 * V5) + (-7.000 * V6)$$

a test is done against 37,316 unseen data, producing the RMSE of 8.0612, which is a little higher than the RMSE obtained from the training data set.

This test set's RMSE is the benchmark of accuracy improvement, that one can yield by employing SVR to build a regression model. The RMSE from the best SVR kernel and parameters setting is 6.9331, which means an enhancement in prediction by 14%. It is the expected decrease of error prediction when building a model merely on the continuous variables.

5.2 Experiment with All Variables

In the experiment with all variables, the linear regression is built using *enter* method and *forward stepwise* method. With the former, all variables are included simultaneously to generate a single model. Whereas in forward stepwise, the independent variables are included one by one based on their statistic score, resulting in multiple models. Independent variables are tested consecutively according to their relevance with the dependent variable. The one with the strongest relationship, i.e. the largest absolute value of correlation coefficient, is chosen first.

The motivation for running regression twice is to show the effect of model complexity to generalization error. With feature selection, generalization error is expected to decrease as less important variables are excluded to avoid over-fitting.

5.2.1 Enter Method

With enter method, SPSS uses 170 out of 178 independent variables available to create the following regression line, excluding 8 variables (V7, V15, V25, V31, V34, V80, V88, V98) due to multicollinearity problem. The model summary as well as the ANOVA are presented in Table 5.7 and 5.8 subsequently [1].

[1] The comprehensive results from SPSS can be found in the CD attached.

From Table 5.7 one could observe, that the training set's RMSE declines from 7.7657 to 6.1408, by including all independent variables. The test set's RMSE corresponds with the same manner. By applying the equation 5.9, a 22.79% RMSE decrease to 6.2240 is achieved, in compare to the model with continuous variables only.

5.2.2 Forward Stepwise Method

The outcomes of forward stepwise method are 109 regression models. The summary of these models could be seen in Table 5.10. Column Var. Entered displays which independent variable is entered for each consecutive model. The criteria used for variable inclusion is the probability of F $\leq$ 0.05 [2].

Since the R squared indicates the proportion of variation in the dependent variable explained by the regression, it could be used to determine the best model. [Inc06] recommends to choose a model with high value of R squared that does not include too many variables, because models with too many variables are often over-fit and hard to interpret. Based on this recommendation and considering the insignificant R Square Change of model 35 onwards (0.000), the author chooses model 34 as the best one. The coefficients of this regression model can be found in Table 5.12, while its ANOVA in Table 5.11.

By applying model 34 to the test set, an RMSE of 6.3519 is obtained. This is a 21.20% RMSE reduction from the model with continuous variables, but surprisingly, the performance turns out to be not as good as the one with enter method (6.2240).

The lowest RMSE achieved in the final SVR experiment including all variables is 5.9383. This means, that SVR can improve the accuracy by 4.6% in compare to the statistical model resulted from the enter method, or even 6.5% to the one with forward stepwise method.

	N	Mean	Std. Deviation	Skewness		Kurtosis	
	Statistic	Statistic	Statistic	Statistic	Std. Error	Statistic	Std. Error
V2	87070	57.8456	16.1548	-0.2636	0.0083	-0.6352	0.0166
V3	87070	0.0294	0.0181	4.4410	0.0083	179.7834	0.0166
V4	87070	0.0505	0.0477	2.6692	0.0083	28.1393	0.0166
V5	87070	0.4112	0.2553	0.3946	0.0083	-0.7581	0.0166
V6	87070	0.0402	0.0705	2.0336	0.0083	7.9582	0.0166

Table 5.1: Descriptive Statistics of Dependent Variable and Independent Continuous Variables

[2] The criteria used for exclusion is the probability of F $\geq$ 0.10.

		V2	V3	V4	V5	V6
Pearson Correlation	V2	1.000	-0.809	-0.725	-0.634	-0.439
	V3	-0.809	1.000	0.579	0.646	0.503
	V4	-0.725	0.579	1.000	0.424	0.310
	V5	-0.634	0.646	0.424	1.000	0.343
	V6	-0.439	0.503	0.310	0.343	1.000
Sig. (1-tailed)	V2	.	0.000	0.000	0.000	0.000
	V3	0.000	.	0.000	0.000	0.000
	V4	0.000	0.000	.	0.000	0.000
	V5	0.000	0.000	0.000	.	0.000
	V6	0.000	0.000	0.000	0.000	.

Table 5.2: Correlations - Continuous Variables

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	0.8770	0.7689	0.7689	7.7657

Table 5.3: Model Summary - Continuous Variables

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	17472421	4	4368105.227	72432.111	0.000
	Residual	5250559	87065	60.306		
	Total	22722980	87069			

Table 5.4: ANOVA - Continuous Variables

Model		Unstandardized Coef.		Standardized Coef.	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	81.116	0.054		1,489.079	0.000
	V3	-424.262	2.271	-0.474	-186.822	0.000
	V4	-126.711	0.680	-0.374	-186.360	0.000
	V5	-10.058	0.136	-0.159	-74.206	0.000
	V6	-7.000	0.432	-0.031	-16.201	0.000

Table 5.5: Regression Coefficients - Continuous Variables

Model	Dimen.	Eigenvalue	Condition	Variance Proportions				
			Index	(Constant)	V3	V4	V5	V6
1	1	3.9120	1.0000	0.0123	0.0069	0.0158	0.0092	0.0191
	2	0.5740	2.6107	0.0467	0.0007	0.0102	0.0093	0.8135
	3	0.2848	3.7065	0.1709	0.0005	0.7845	0.0353	0.0177
	4	0.1457	5.1811	0.7311	0.0435	0.0632	0.4874	0.0637
	5	0.0836	6.8413	0.0390	0.9484	0.1263	0.4588	0.0861

Table 5.6: Collinearity Diagnostics - Continuous Variables

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	0.925	0.856	0.856	6.1408

Table 5.7: Model Summary - Enter Method

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	19446071.866	170	114388.658	3033.426491	.000(a)
	Residual	3276908.152	86899	37.709		
	Total	22722980.018	87069			

Table 5.8: ANOVA - Enter Method

$$
\begin{aligned}
V2_{pred} = \; & 79.9619 + (-86.3978 * V3) + (-94.1792 * V4) + (-30.9193 * V5) + (-5.9418 * \\
& V6) + (-5.4134 * V8) + (-1.5103 * V9) + (1.6185 * V10) + (-0.4702 * V11) + \\
& (-1.0318 * V12) + (-0.6459 * V13) + (0.2661 * V14) + (-0.9407 * V16) + \\
& (-3.2836 * V17) + (-1.5689 * V18) + (2.6305 * V19) + (1.1536 * V20) + \\
& (-2.1629 * V21) + (-1.5373 * V22) + (1.9688 * V23) + (-1.5257 * V24) + \\
& (-2.8816 * V26) + (-0.7412 * V27) + (-7.9317 * V28) + (-5.1911 * V29) + \\
& (-1.8808 * V30) + (1.9583 * V32) + (4.4570 * V33) + (-1.5375 * V35) + \\
& (-3.4558 * V36) + (-2.7253 * V37) + (-4.5433 * V38) + (-2.7596 * V39) + \\
& (-4.7640 * V40) + (-2.4419 * V41) + (2.9402 * V42) + (1.5986 * V43) + (0.2484 * \\
& V44) + (-0.8348 * V45) + (3.4632 * V46) + (0.8415 * V47) + (-0.8680 * V48) + \\
& (0.3012 * V49) + (3.4655 * V50) + (0.9649 * V51) + (6.5376 * V52) + (7.8867 * \\
& V53) + (4.9999 * V54) + (2.8996 * V55) + (8.9795 * V56) + (8.4479 * V57) + \\
& (-1.2725 * V58) + (-0.9764 * V59) + (-3.0832 * V60) + (-0.9567 * V61) + \\
& (-1.1983 * V62) + (-1.0620 * V63) + (-1.2266 * V64) + (-2.1482 * V65) + \\
& (0.7759 * V66) + (-2.2587 * V67) + (-1.0758 * V68) + (0.1718 * V69) + \\
& (-1.0461 * V70) + (-2.0234 * V71) + (-1.8560 * V72) + (-1.1497 * V73) + \\
& (-3.3939 * V74) + (-1.7986 * V75) + (-2.4727 * V76) + (-1.1551 * V77) + \\
& (-1.3934 * V78) + (-1.3169 * V79) + (-1.2590 * V81) + (-2.1558 * V82) + \\
& (-0.3334 * V83) + (-1.5544 * V84) + (-1.1133 * V85) + (-1.0379 * V86) + \\
& (0.1248 * V87) + (-0.3463 * V89) + (-0.2439 * V90) + (-0.4037 * V91) + \\
& (-1.5990 * V92) + (-1.1756 * V93) + (-0.6423 * V94) + (-0.1986 * V95) + \\
& (-0.6582 * V96) + (-0.4294 * V97) + (0.1010 * V99) + (0.7275 * V100) + \\
& (-0.0460 * V101) + (-1.4296 * V102) + (-0.2321 * V103) + (-0.4321 * V104) + \\
& (0.4010 * V105) + (-1.1251 * V106) + (-0.4195 * V107) + (-0.4152 * V108) + \\
& (-0.1240 * V109) + (-0.3681 * V110) + (-0.3635 * V111) + (-0.2279 * V112) + \\
& (0.1593 * V113) + (0.1331 * V114) + (-0.5259 * V115) + (-0.8022 * V116) + \\
& (-0.0658 * V117) + (0.3279 * V118) + (-2.1053 * V119) + (-1.9921 * V120) + \\
& (-0.9213 * V121) + (-0.3503 * V122) + (-0.9337 * V123) + (0.1713 * V124) + \\
& (-0.5788 * V125) + (-0.3980 * V126) + (-0.4773 * V127) + (-0.0442 * V128) + \\
& (-0.4605 * V129) + (-0.7047 * V130) + (-0.7625 * V131) + (-0.1858 * V132) + \\
& (-1.2545 * V133) + (-0.0939 * V134) + (-0.2106 * V135) + (0.0008 * V136) + \\
& (-0.3187 * V137) + (-0.2994 * V138) + (-0.4376 * V139) + (-0.0811 * V140) + \\
& (-0.2381 * V141) + (0.6580 * V142) + (0.3003 * V143) + (-0.2427 * V144) + \\
& (-0.6669 * V145) + (0.0323 * V146) + (0.2126 * V147) + (-0.2496 * V148) + \\
& (-0.8285 * V149) + (-0.6827 * V150) + (-1.1741 * V151) + (0.3630 * V152) + \\
& (-0.3916 * V153) + (0.3280 * V154) + (-0.0443 * V155) + (-1.0064 * V156) + \\
& (-0.4844 * V157) + (-0.2652 * V158) + (-0.1415 * V159) + (-1.8737 * V160) + \\
& (-0.6365 * V161) + (-0.3161 * V162) + (1.6179 * V163) + (1.4242 * V164) + \\
& (-0.6974 * V165) + (-0.1148 * V166) + (0.0518 * V167) + (0.1469 * V168) + \\
& (-0.1489 * V169) + (-1.9362 * V170) + (-0.1507 * V171) + (1.6113 * V172) + \\
& (2.4232 * V173) + (0.1285 * V174) + (0.5679 * V175) + (-0.0930 * V176) + \\
& (1.5410 * V177) + (1.9008 * V178) + (1.8380 * V179) + (2.0977 * V180)
\end{aligned}
$$

Table 5.9: Regression Model - Enter Method

Md.	Var. Entered	R	R Square	Adj.R Square	Std. Err. of Estim.	Change Statistics				
						R Square Chg.	F Change	df1	df2	Sig. F Chg.
1	V3	0.809	0.654	0.654	9.5001	0.654	164704.928	1	87068	0.000
2	V4	0.868	0.753	0.753	8.0217	0.099	35052.707	1	87067	0.000
3	V5	0.876	0.768	0.768	7.7774	0.015	5556.906	1	87066	0.000
4	V8	0.883	0.780	0.780	7.5762	0.012	4686.906	1	87065	0.000
5	V20	0.888	0.789	0.789	7.4138	0.009	3855.283	1	87064	0.000
6	V56	0.893	0.797	0.797	7.2798	0.008	3235.830	1	87063	0.000
7	V28	0.895	0.802	0.802	7.1914	0.005	2155.863	1	87062	0.000
8	V29	0.899	0.808	0.808	7.0700	0.007	3016.371	1	87061	0.000
9	V52	0.902	0.813	0.813	6.9916	0.004	1963.656	1	87060	0.000
10	V57	0.904	0.817	0.817	6.9188	0.004	1842.478	1	87059	0.000
11	V53	0.906	0.821	0.821	6.8364	0.004	2111.930	1	87058	0.000
12	V42	0.908	0.825	0.825	6.7565	0.004	2074.223	1	87057	0.000
13	V21	0.910	0.828	0.828	6.6911	0.003	1709.797	1	87056	0.000
14	V40	0.911	0.831	0.831	6.6461	0.002	1185.227	1	87055	0.000
15	V80	0.912	0.833	0.833	6.6098	0.002	957.586	1	87054	0.000
16	V38	0.913	0.834	0.834	6.5773	0.002	864.011	1	87053	0.000
17	V10	0.914	0.836	0.836	6.5515	0.001	689.308	1	87052	0.000
18	V179	0.915	0.837	0.837	6.5264	0.001	669.862	1	87051	0.000
19	V178	0.916	0.838	0.838	6.4983	0.001	756.293	1	87050	0.000
20	V30	0.916	0.839	0.839	6.4748	0.001	634.275	1	87049	0.000
21	V50	0.917	0.841	0.841	6.4504	0.001	660.640	1	87048	0.000
22	V33	0.917	0.842	0.842	6.4280	0.001	608.523	1	87047	0.000
23	V17	0.918	0.843	0.843	6.4109	0.001	466.797	1	87046	0.000
24	V9	0.918	0.843	0.843	6.3935	0.001	475.774	1	87045	0.000
25	V39	0.919	0.844	0.844	6.3766	0.001	462.713	1	87044	0.000
26	V54	0.919	0.845	0.845	6.3607	0.001	435.702	1	87043	0.000
27	V31	0.920	0.846	0.846	6.3456	0.001	416.305	1	87042	0.000
28	V121	0.920	0.846	0.846	6.3324	0.001	364.403	1	87041	0.000
29	V120	0.920	0.847	0.847	6.3206	0.001	325.177	1	87040	0.000
30	V25	0.921	0.848	0.848	6.3079	0.001	352.728	1	87039	0.000
31	V35	0.921	0.848	0.848	6.2954	0.001	347.823	1	87038	0.000
32	V19	0.921	0.849	0.849	6.2835	0.001	330.210	1	87037	0.000
33	V36	0.922	0.849	0.849	6.2719	0.001	322.635	1	87036	0.000
34	V142	0.922	0.850	0.850	6.2613	0.001	297.166	1	87035	0.000
35	V6	0.922	0.850	0.850	6.2526	0.000	243.566	1	87034	0.000
36	V37	0.922	0.851	0.851	6.2448	0.000	218.755	1	87033	0.000
37	V60	0.922	0.851	0.851	6.2372	0.000	211.644	1	87032	0.000
38	V177	0.923	0.851	0.851	6.2303	0.000	196.069	1	87031	0.000
39	V15	0.923	0.852	0.852	6.2232	0.000	199.024	1	87030	0.000

Continued on next page

Table5.10 – continued from previous page

Md.	Var. Entered	R	R Square	Adj.R Square	Std. Err. of Estim.	Change Statistics				
						R Square Chg.	F Change	df1	df2	Sig. F Chg.
40	V46	0.923	0.852	0.852	6.2169	0.000	175.706	1	87029	0.000
41	V172	0.923	0.852	0.852	6.2110	0.000	168.527	1	87028	0.000
42	V127	0.923	0.853	0.852	6.2055	0.000	154.029	1	87027	0.000
43	V55	0.923	0.853	0.853	6.2005	0.000	142.053	1	87026	0.000
44	V173	0.924	0.853	0.853	6.1955	0.000	140.640	1	87025	0.000
45	V133	0.924	0.853	0.853	6.1911	0.000	125.400	1	87024	0.000
46	V165	0.924	0.853	0.853	6.1866	0.000	127.657	1	87023	0.000
47	V43	0.924	0.854	0.854	6.1833	0.000	94.892	1	87022	0.000
48	V145	0.924	0.854	0.854	6.1800	0.000	93.385	1	87021	0.000
49	V123	0.924	0.854	0.854	6.1772	0.000	80.336	1	87020	0.000
50	V149	0.924	0.854	0.854	6.1747	0.000	70.185	1	87019	0.000
51	V160	0.924	0.854	0.854	6.1724	0.000	67.421	1	87018	0.000
52	V65	0.924	0.854	0.854	6.1705	0.000	53.603	1	87017	0.000
53	V12	0.924	0.854	0.854	6.1688	0.000	48.428	1	87016	0.000
54	V125	0.924	0.854	0.854	6.1672	0.000	47.300	1	87015	0.000
55	V126	0.924	0.854	0.854	6.1646	0.000	72.327	1	87014	0.000
56	V130	0.924	0.855	0.854	6.1633	0.000	39.522	1	87013	0.000
57	V152	0.924	0.855	0.855	6.1619	0.000	38.880	1	87012	0.000
58	V144	0.924	0.855	0.855	6.1607	0.000	36.421	1	87011	0.000
59	V51	0.925	0.855	0.855	6.1596	0.000	32.932	1	87010	0.000
60	V138	0.925	0.855	0.855	6.1585	0.000	29.680	1	87009	0.000
61	V67	0.925	0.855	0.855	6.1575	0.000	29.488	1	87008	0.000
62	V116	0.925	0.855	0.855	6.1567	0.000	24.097	1	87007	0.000
63	V137	0.925	0.855	0.855	6.1559	0.000	23.309	1	87006	0.000
64	V141	0.925	0.855	0.855	6.1552	0.000	22.442	1	87005	0.000
65	V161	0.925	0.855	0.855	6.1544	0.000	21.904	1	87004	0.000
66	V72	0.925	0.855	0.855	6.1538	0.000	19.540	1	87003	0.000
67	V47	0.925	0.855	0.855	6.1531	0.000	18.977	1	87002	0.000
68	V27	0.925	0.855	0.855	6.1526	0.000	17.212	1	87001	0.000
69	V114	0.925	0.855	0.855	6.1520	0.000	16.105	1	87000	0.000
70	V75	0.925	0.855	0.855	6.1515	0.000	15.550	1	86999	0.000
71	V74	0.925	0.855	0.855	6.1510	0.000	15.515	1	86998	0.000
72	V76	0.925	0.855	0.855	6.1505	0.000	15.390	1	86997	0.000
73	V118	0.925	0.855	0.855	6.1501	0.000	12.535	1	86996	0.000
74	V110	0.925	0.855	0.855	6.1496	0.000	15.426	1	86995	0.000
75	V139	0.925	0.855	0.855	6.1492	0.000	12.552	1	86994	0.000
76	V26	0.925	0.855	0.855	6.1488	0.000	12.669	1	86993	0.000
77	V16	0.925	0.855	0.855	6.1482	0.000	15.331	1	86992	0.000

Continued on next page

Table5.10 – continued from previous page

Md.	Var. Entered	R	R Square	Adj.R Square	Std. Err. of Estim.	Change Statistics				
						R Square Chg.	F Change	df1	df2	Sig. F Chg.
78	V154	0.925	0.855	0.855	6.1479	0.000	11.732	1	86991	0.001
79	V84	0.925	0.855	0.855	6.1475	0.000	11.716	1	86990	0.001
80	V170	0.925	0.855	0.855	6.1471	0.000	11.311	1	86989	0.001
81	V34	0.925	0.855	0.855	6.1468	0.000	11.185	1	86988	0.001
82	V100	0.925	0.855	0.855	6.1465	0.000	9.504	1	86987	0.002
83	V87	0.925	0.855	0.855	6.1462	0.000	9.656	1	86986	0.002
84	V23	0.925	0.855	0.855	6.1459	0.000	9.405	1	86985	0.002
85	V82	0.925	0.855	0.855	6.1456	0.000	8.938	1	86984	0.003
86	V111	0.925	0.855	0.855	6.1453	0.000	8.833	1	86983	0.003
87	V150	0.925	0.855	0.855	6.1450	0.000	8.622	1	86982	0.003
88	V88	0.925	0.855	0.855	6.1448	0.000	8.695	1	86981	0.003
89	V99	0.925	0.855	0.855	6.1444	0.000	10.900	1	86980	0.001
90	V156	0.925	0.855	0.855	6.1441	0.000	8.800	1	86979	0.003
91	V69	0.925	0.856	0.855	6.1439	0.000	8.725	1	86978	0.003
92	V112	0.925	0.856	0.855	6.1436	0.000	7.448	1	86977	0.006
93	V151	0.925	0.856	0.855	6.1434	0.000	7.441	1	86976	0.006
94	V66	0.925	0.856	0.855	6.1431	0.000	9.668	1	86975	0.002
95	V44	0.925	0.856	0.855	6.1428	0.000	8.551	1	86974	0.003
96	V61	0.925	0.856	0.855	6.1426	0.000	8.179	1	86973	0.004
97	V131	0.925	0.856	0.855	6.1423	0.000	7.891	1	86972	0.005
98	V158	0.925	0.856	0.855	6.1421	0.000	7.211	1	86971	0.007
99	V49	0.925	0.856	0.855	6.1419	0.000	6.134	1	86970	0.013
100	V102	0.925	0.856	0.855	6.1418	0.000	5.585	1	86969	0.018
101	V143	0.925	0.856	0.855	6.1416	0.000	5.124	1	86968	0.024
102	V103	0.925	0.856	0.855	6.1415	0.000	5.258	1	86967	0.022
103	V122	0.925	0.856	0.855	6.1413	0.000	5.097	1	86966	0.024
104	V153	0.925	0.856	0.855	6.1412	0.000	5.026	1	86965	0.025
105	V168	0.925	0.856	0.855	6.1411	0.000	4.864	1	86964	0.027
106	V162	0.925	0.856	0.856	6.1409	0.000	4.656	1	86963	0.031
107	V175	0.925	0.856	0.856	6.1408	0.000	4.423	1	86962	0.035
108	V148	0.925	0.856	0.856	6.1407	0.000	4.360	1	86961	0.037
109	V92	0.925	0.856	0.856	6.1406	0.000	3.911	1	86960	0.048

Table 5.10: Model Summary - Stepwise Method

Model		Sum of Squares	df	Mean Square	F	Sig.
34	Regression	19310883	34	567967.155	14487.579	.000
	Residual	3412097	87035	39.204		
	Total	22722980	87069			

Table 5.11: ANOVA - Model 34 - Stepwise Method

Model		Unstandardized Coef.		Standardized Coef.	t	Sig.
		B	Std. Error	Beta		
34	(Constant)	79.227	0.119		663.611	0.000
	V3	-100.315	3.441	-0.112	-29.156	0.000
	V4	-93.788	0.645	-0.277	-145.345	0.000
	V5	-31.044	0.212	-0.491	-146.356	0.000
	V8	-5.354	0.064	-0.119	-83.339	0.000
	V20	3.572	0.083	0.087	42.901	0.000
	V56	6.819	0.097	0.108	70.300	0.000
	V28	-12.501	0.167	-0.257	-74.995	0.000
	V29	-9.483	0.143	-0.196	-66.458	0.000
	V52	4.793	0.095	0.072	50.499	0.000
	V57	6.276	0.152	0.059	41.396	0.000
	V53	6.194	0.141	0.060	43.825	0.000
	V42	2.668	0.066	0.058	40.723	0.000
	V21	-0.610	0.098	-0.012	-6.214	0.000
	V40	-6.178	0.178	-0.046	-34.787	0.000
	V80	1.258	0.047	0.036	27.027	0.000
	V38	-5.368	0.181	-0.039	-29.631	0.000
	V10	1.517	0.053	0.047	28.739	0.000
	V179	2.006	0.126	0.025	15.960	0.000
	V178	1.850	0.126	0.022	14.683	0.000
	V30	-4.365	0.112	-0.113	-38.883	0.000
	V50	2.840	0.120	0.032	23.591	0.000
	V33	3.250	0.136	0.036	23.874	0.000
	V17	-2.373	0.135	-0.024	-17.571	0.000
	V9	-1.312	0.050	-0.037	-26.096	0.000
	V39	-2.967	0.135	-0.030	-22.030	0.000
	V54	3.849	0.193	0.027	19.983	0.000
	V31	-1.906	0.085	-0.055	-22.422	0.000
	V121	-1.485	0.069	-0.030	-21.565	0.000
	V120	-2.917	0.164	-0.024	-17.748	0.000
	V25	1.800	0.075	0.055	23.888	0.000
	V35	-1.826	0.091	-0.027	-20.084	0.000
	V19	4.526	0.245	0.025	18.442	0.000
	V36	-4.220	0.229	-0.025	-18.457	0.000
	V142	0.892	0.052	0.024	17.238	0.000

Table 5.12: Regression Coefficients - Model 34 - Stepwise Method

Chapter 6

Conclusion and Outlook

As stated in the introduction chapter, there are three main tasks to be carried out in this thesis:

1. Applying the Support Vector Machine to build a good regression model that can explain the car's residual price.
2. Conducting an SVR parameters search automization experiment to improve SVR accuracy and time efficiency.
3. Presenting the learning on methodology and results from the experiment with a large, high dimensional data set for the SVR communities.

In the following section, the experiment process, results and analysis will be summarized to conclude the forementioned tasks.

6.1 Conclusion

In this experiment, the SVR has successfully proved its capability in generating a good prediction model. With the right kernel and hyperparameters setting, it achieves a better accuracy than the standard solution, multiple linear regression. Figure 6.1 shows the RMSE comparison between different statistical regression and SVR models on the final test set in this experiment. In both cases, with continuous and all variables, the RMSEs for SVR are lower than those achieved by the statistical model, demonstrating the superiority of SVR over its benchmark.

A lower generalization error can be achieved by SVR because it applies the structural risk minimization paradigm while learning from a data set, which means SVR minimizes both the empirical error and the model complexity. Due to this concept, SVR computation does not depend on the input space dimension, therefore it is suitable to solve a high dimensional problem, without having to face the curse of dimensionality. Another factor that decreases the generalization error is the use of kernel, which can capture the non-linear relationship in the data set. However, the kernel suitability depends on the nature of the data. For

this particular data set, the RBF kernel shows to be more appropriate than the linear and polynomial kernel.

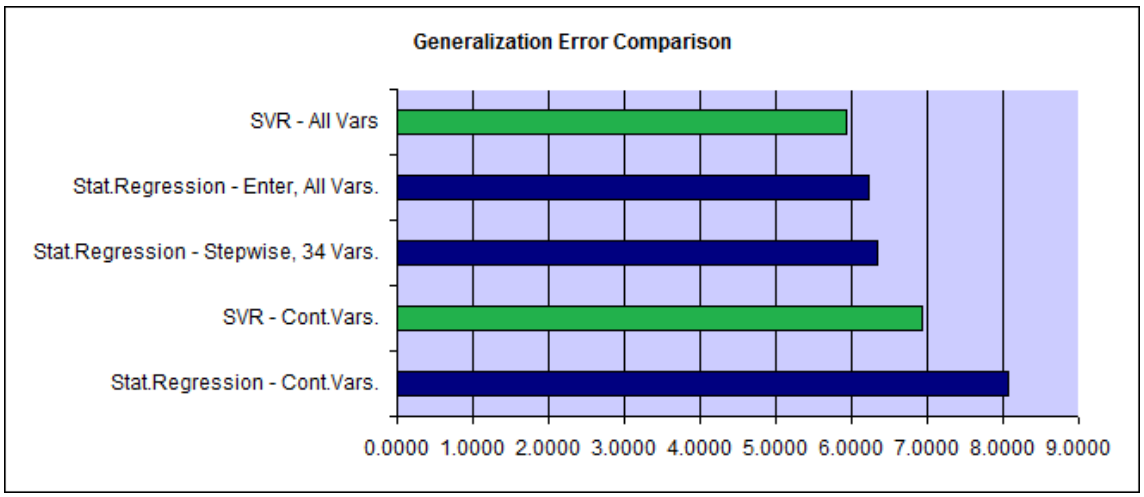


Figure 6.1: RMSE Comparison of Different Statistical Regression and SVR Models from The Final Experiment

Since the model accuracy depends on the hyperparameters, finding the right setting is a very important task in the SVR experiment. As the parameters setting depends on the training data set, and there is no off-the-shelf solution, it has to be searched. Two automated searching algorithm have been tested here, namely grid search and evolution strategy. The results show that evolution strategy has an advantage over grid search in terms of finding better setting which generates lower error, and learning time needed to find the optimal setting.

Regarding the methodology, based on the experience gained in this experiment, the author suggests the following procedure for using SVR in a large data set application:

1. Data preparation: including multinomial to binary variables transformation, independent variables standardization, data shuffling and data division to create three subsets, which serve as the training and validation set in the hyperparameter selection phase, and as the final training and test set.
2. Selection of SVR kernels.
3. Learning curve: to determine the minimal training data set size needed for the hyperparameter search process. This step is important to handle a huge data set, in order to maximize the use of the information available, without getting trapped in an excessive learning duration.
4. Hyperparameters search with the evolution strategy algorithm.
5. Best hyperparameters selection based on the result of previous search.
6. Final training and testing by utilizing the best parameters setting.

The above mentioned steps need to be done consistently with the data subsets prepared in the beginning.

With respect to high dimensionality, the car data set consists of 178 attributes. Therefore, there are two arrangements for the attributes inclusion. The first considers only the continuous independent variables, while the second includes the binary variables as well, when predicting the future re-sell price. The results from SVR with all kernels and multiple linear regression show, that the prediction accuracy can be improved substantially by incorporating all independent variables.

Considering the time, the whole process of building an SVR model from a huge data set may take days, depending on the power of the computer. On the other hand, the computation of multiple linear regression, with the same data set size and platform, can be finished just in several minutes. This is the weakness of SVR. However, as it is mentioned in the introduction chapter, the car makers and dealers need a professional price calculation, in order to avoid the expensive cost of wrong prediction. SVR can provide this, because it yields a higher accuracy than statistical model. Moreover, the lengthy model construction takes place only periodically, when the model needs to be updated, whereas the predicting time is actually only several miliseconds. Another convenient reason to employ SVR is its independence of the input space dimensionality. It frees the analyst from scrutinizing variables effect on the outcome one by one, as what happen when the multiple linear regression is used. Besides, the whole learning process can be automated. With the aforementioned arguments, the users are expected to accept SVR training time, and utilize it due to its high performance.

6.2 Summary of Findings

There are many findings regarding parameters selection that have been achieved through this experiment, and they will be elaborated in two parts: grid search and evolution strategy.

6.2.1 Grid Search and SVR Hyperparameters Impact

In grid search, different combinations of parameters within certain range are tested, to check which one generates the lowest error. Although grid search is computation expensive and time consuming, it gives a good insight on the SVR hyperparameters interplay and their consequences for different kernels.

SVR with Linear Kernel

- There are two hyperparameters for linear SVR: the cost and the epsilon. The generalization error (RMSE) goes down along with the increasing cost and decreasing epsilon. A higher cost penalizes the empirical training error more, so that the model built is less under-fit. While lower epsilon defines a lower insensitivity loss function, which reflects a higher learning capacity for a better model fitting. However, how big or small the values should be are relative to a particular data set.
- The cost has an inferior effect to epsilon in determining the model accuracy. Thus,

when an SVR model generates a very high error, one has to first check the epsilon value. If the radius of the insensitivity tube is too big, then the model is under-fitting, thus, the empirical, as well as generalization error, goes up.

- The plateau of decent parameter combinations is quite large. Therefore, a finer grid search is not necessary for model selection, it is enough to do a coarse search in order to reduce computational effort and time.
- The training time for linear SVR depends on both epsilon and cost values, besides training data size. There is a high increase in training duration for a small epsilon and big cost combination. The lower the epsilon is, the longer duration it takes, since more learning effort is needed. A comparison between their impact reveals, that the epsilon gives much higher influence than the cost to training time. The cost has a linear effect, whereas the epsilon has a logarithmic one.

SVR with Polynomial Kernel

- The important parameters for polynomial SVR, beside the cost and the epsilon, is the polynomial degree. Due to the particular nature of the car data set, the RMSE is smaller for the lower polynomial degree.
- A higher polynomial degree is more sensitive to the change of cost. While building a model, SVR with a high value of cost combined with a high learning capacity reduces the empirical training error more than it is needed to achieve the best model. This leads to over-fitting, and therefore, the generalization error for the validation set goes up.
- Different settings of epsilon show the same impact in accordance with the learning from linear SVR. A big epsilon value leads to under-fitting.

SVR with RBF Kernel

- The hyperparameters for RBF SVR are: the cost, the epsilon and the gamma. The search for an appropriate gamma value in this experiment shows, that increasing gamma settings generate the trend of decreasing RMSE. However, this trend will stop at a certain turning point. Afterwards, as gamma further increases, the value of generalization error will increase as well.
- With the right setting of gamma and epsilon to determine the machine learning capacity, a very high cost to penalize the empirical error is not necessary in order to gain a low generalization error.
- One can expect a long computation duration to generate SVR model with a mixture of large cost, low epsilon and approriate gamma setting (appropriate here means, that the gamma is likely to yield the minimum RMSE).

As in any other circumstances, where the complexity drives up the problem solving time, a high dimensional data leads to a longer training duration. Unfortunately, such a long computation does not always entail low generalization error due to over-learning. Therefore, it has to be avoided. This is exactly why evolution strategy is prefered over grid search for hyperparameters selection.

6.2.2 Evolution Strategy

Evolution strategy is an automated search approach, which is based on Darwin's evolution theory. A population of SVR hyperparameter candidates must undergo several rounds of regeneration and selection process. The selection is based on the fitness of an individual as a solution, in this case the RMSE value.

The regeneration is done through mutation. In this experiment, the mutation for the parameter values is implemented as the normal distribution around the object individual, with the mean of 0 and standard deviation σ. The standard deviation (mutation step) is also self adapted by embedding σ in the candidates' chromosome. There are two options to pass the solution chromosomes on to the next generation. With comma strategy, only partial solution chromosomes can be passed on, because the selection is done within the newly mutated offsprings population. Whereas plus strategy allows the whole solution chromosomes of an individual with high fitness to survive multiple generations, because the selection's domain is the combined parents and offsprings populations.

There are several points to be mentioned here about evolution strategy in compare to grid search:

1. The best parameters combination found in grid search can be matched in approximity by evolution strategy, because grid search shows that there is a large plateau of low errors with different appropriate parameter settings.

2. The resulting minimum RMSE from evolution strategy is always better, or at least as good as the one found in grid search. This is due to the flexible search area of evolution strategy, which is the advantage of this approach over grid search. The grid search is limited to the parameter values set initially. Thus, one should do a two phase grid search, a coarse one and then a finer one, to improve the model accuracy. Moreover, one should also consider to check beyond the area of current grid search, if the minimum RMSE is achieved by a parameter value in the search range border. The evolution strategy is not curbed by these issues.

3. To find the optimal parameters setting, evolution strategy can start the search from any point, and then move closer to the final solution. This is proven by the decreasing trend in RMSE values along the increasing generation. It also adapts its search level autonomously, so that the initial search could be done within a relatively big range, and then, the range is narrowed down along with the increasing generation. The smaller RMSE spread among the individuals in the later generations proves, that Schwefel's self adaptation procedure for the mutation's step does a good job to reduce the search range and focuses on good parameters instead.

4. Evolution strategy can converge faster toward the more important optimum parameter values, which are the epsilon, polynomial degree and gamma, than to cost.

5. Although evolution strategy can start the search from any point, the initialization point is important to enable it to find the right parameters faster. A small epsilon value can be initialized as an allowed error percentage of the output mean value. The cost derived from Cherkassky and Ma's equation has also proved to be a good starting point for SVR. For polynomial SVR, one can use low degrees such as 2 or 3

for the first parents. Whereas gamma for RBF SVR can be initialized with quite a large range of exponential values, e.g. $\{2^{-3}, 2^0, 2^3\}$.

6. In this experiment with a huge data set, there is no considerable difference in minimum RMSE that can be observed by employing comma or plus strategy. Therefore, the author does not make any recommendation on it.

7. A relatively small parents population size, e.g. 3, and parents-offsprings ratio, e.g. 4, provides large enough spread to find the good solutions.

8. The time saving by using evolution strategy instead of grid search is huge. There are two reasons behind it. First, the number of models that have to be built to find the best parameters with grid search is likely to be higher than with evolution strategy, and thus, longer search duration is needed. Second, with grid search, the SVR can be trapped in some over learning areas. This trap will block the evaluations of other parameter combinations, which might have more probability to be the best hyperparameters. Therefore, grid search is not time efficient for hyperparameter search, especially for high dimensional and huge data set. Evolution strategy, on the other hand, can find good parameters without being trapped in those over-fit cases.

9. One can always limit the search in evolution strategy by either maximum generation number, maximum allowed training time, RMSE improvement threshold or other constraint, in order to avoid over learning.

Hopefully, this learning could give a guidance for novice SVR users in understanding the effect of parameters tuning, as well as the importance of an automated and fast search algorithm in selecting SVR hyperparameters.

6.3 Limitations and Outlook

With the given time constraint to write this thesis, some parts of the experiments have to be left out. Those are the grid search experiments with all variables and high cost settings, and evolution strategy with all variables for polynomial SVR. However, as mentioned before, this should not change the final result, because a very high cost is not needed to achieve a low RMSE, if the other learning capacity parameters are appropriate. The training duration for those high cost settings are overly long, but they are expected to give only a slight improvement to the generalization error.

Moreover, the author chooses to conduct the whole experiment without outliers removal to keep the naturality of the data set. Although outliers are the source of distortion, because they pull the regression model towards themselves. This opens a chance for a further study. Using the result of this experiment as a benchmark, one could check how much accuracy improvement can be gained by removing the outliers.

Furthermore, the multiple linear regression performance on the the particular data set has to be recognized. This is most likely due to the fact that those leased cars are high quality cars, which loose value in a slow rate. Thus, the re-sell price tend to have linear relationship with its independent variables. The performance gap between the SVR and statistical solutions may be larger, when an economic car data set is used, since these cars

loose value quite rapidly, and the re-sell price may follow a logarithmic curve, instead of a linear one. If this is the case, then a kernel usage will boost the accuracy of the SVR model, creating a considerable discrepancy to the multiple linear regression result. Hence, one can do further experiment with other types of cars.

One can also do benchmarking work against other data mining tools in the future. Some possible tools to be tested are neural networks, regression trees, or meta-learning that combines the predictions from multiple models [dep08].

Another constraint in this experiment comes from the data set itself. There are no variables dedicated to external information, such as general economic or vehicle market condition indicators, which could increase the prediction accuracy. As SVR is capable in handling the high dimensionality well, adding some variables to the data set will not be a problem from the machine learning viewpoint. Thus, the author suggests, to choose a small number of external indicators to be used as the pilot data enrichment, and then, to see if those variables can really improve the prediction accuracy.

List of Figures

List of Tables

Bibliography

[Abr07] Deborah Abrams. Introduction to regression. http://dss.princeton.edu/online_help/analysis/regression_intro.htm, 2007.

[AG09] EurotaxGlass's International AG. Schwacke liste. http://www.schwacke.de/, 2009.

[AS03] Shawkat Ali and Kate Smith. Automatic parameter selection for polynomial kernel. *Information Reuse and Integration*, 2003.

[Bel61] Richard Bellman. *Adaptive Control Processes.* Princeton University Press, 1961.

[Chi98a] K.K. Chin. Using a radial basis function as kernel. http://svr-www.eng.cam.ac.uk/~kkc21/thesis_main/node31.html, 1998.

[Chi98b] K.K. Chin. Vc dimension and vc confidence. http://svr-www.eng.cam.ac.uk/~kkc21/thesis_main/node11.html, 1998.

[CL01] Chih-Chung Chang and Chih-Jen Lin. *LibSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[CM02] Vladimir Cherkassky and Yunqian Ma. Selection of meta-parameters for support vector regression. *Lecture Notes in Computer Science*, 2002.

[Con05] Mercer Management Consulting. Systemprofit automobilvertrieb 2015 - die agenda für profitables wachstum der markenkanäle, 2005.

[CST00] Nello Cristianini and John Shawe-Taylor. *Support Vector Machines and Other Kernel-Based Learning Methods.* Cambridge University Press, 2000.

[dep08] StatSoft R&D department. Electronic textbook: Elementary concepts in statistics - data mining techniques. http://www.statsoft.com/textbook/stdatmin.html, 2008.

[Fre02] Alex Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms.* Springer-Verlag, 2002.

[Gij07] Arjan Gijsberts. Evolutionary optimization of kernel machines. Master's thesis, Delft University of Technology, 2007.

[Gmb08] Deutsche Presse-Agentur GmbH. Markt schwächelt, deutsche stark. `http://www.auto-motor-und-sport.de/news/us-absatz-markt-schwaechelt-deutsche-stark-701883.html`, 2008.

[Gmb09] Deutsche Automobil Treuhand GmbH. Dat deutschland. `http://www.dat.de/`, 2009.

[HCL08] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. *Manual of LibSVM: A Practical Guide to Support Vector Classification*, 2008.

[HKK06] Te-Ming Huang, Vojislav Kecman, and Ivica Kopriva. *Kernel Based Algorithms for Mining Huge Data Sets.* Springer-Verlag, 2006.

[HSS08] Thomas Hofmann, Bernhard Schölkopf, and Alexander Smola. Kernel methods in machine learning. *The Annals of Statistics*, 2008.

[Inc05] The Mathworks Inc. *Help Documentation of Matlab Version 7*, 2005.

[Inc06] SPSS Inc. *Tutorial of SPSS 15.0 for Windows*, 2006.

[KKMN98] David Kleinbaum, Lawrence Kupper, Keith Muller, and Azhar Nizam. *Applied Regression Analysis and Multivariable Methods.* Duxbury Press, 1998.

[MC06] Mercer and Oliver Wyman Management Consulting. Schwacher neuwagenabsatz erfordert effektives restwertmanagement und vorwärtsstrategien für gebrauchtwagen. *Spektrum*, 2006.

[PPS03] Claudia Perlich, Foster Provost, and Jeffrey Simonoff. Tree induction vs. logistic regression: A learning-curve analysis. *Journal of Machine Learning Research*, 2003.

[RN03] Stuart Russel and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall, 2003.

[Sar09] Warren Sarle. Neural network faq. periodic posting to the usenet newsgroup. `http://www.faqs.org/faqs/ai-faq/neural-nets/`, 2009.

[Sch81] Hans-Paul Schwefel. *Numerical Optimization of Computer Models.* John Wiley & Sons, Inc., 1981.

[Sch03] James Schwab. Solving problem in spss: Impact of assumptions and outliers. `http://www.utexas.edu/courses/schwab/sw388r7/SolvingProblems/MultipleRegression_AssumptionsAndOUtliers.ppt`, 2003.

[Sil86] Bernard Silverman. *Density estimation: for statistics and data analysis.* Chapman and Hall, 1986.

[SS03] Alex Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 2003.

[Vap98] Vladimir Vapnik. *The Nature of Statistical Learning Theory.* John Wiley & Sons, Inc., 1998.

[VC74] Vladimir Vapnik and Alexey Chervonenkis. Structural risk minimization. `http://www.svms.org/srm/`, 1974.

[WT07] Gary Weiss and Ye Tian. Maximizing classifier utility when there are data acquisition and modeling costs. *Data Mining and Knowledge Discovery*, 2007.