Forecast of parameter values for series projects and risk assessment in production and economic systems

Master’s Thesis

Submitted to the
Department of Computer Science and Languages at
Anhalt University of Applied Sciences
in fulfillment of the requirements for the degree of
Master of Science

Student: A. Seledkova
(Matr. Nr.: 4065061)
supervisor: Dr. B. Krause
June 2017

Annotation

The master's thesis describes the choice features of methods for predicting the parameter values of production and economic systems. For this, the data features collected and used are considered, the main problem in the projects’ data analysis implemented in production and economic systems will be either a small amount of statistical data or data describing a number of products, product modifications, etc.

The main approaches used for predicting parameters are considered. These are approaches described by special curves, supervised learning, unsupervised learning and semi-supervised learning. Features of the forecasted data and also forecasts based on regression analysis, the method of support vector machine, auto regression and wavelet analysis are constructed. For the predicted values, the method of choosing the forecasting method based on the risk assessment of the introduced forecasts is given. The choice is proposed to be made on the basis of an estimate of the exact prediction for a maximum period of time. Since, starting with a certain forecast value, risk value assessment begins to increase sharply, which allow choosing the method of forecasting.

Also, for these forecast values, the root-mean-square error and the mean error of approximation are calculated. The best methods are compared on the basis of calculating the accuracy of forecasting and assessing the risk of compiled forecasts. The result of the comparison with a small margin of error makes it possible to use the risk assessment in the future to estimate the prediction of the model

Contents

List of abbreviations

SVM	– support vector machine
K-NN	– k-nearest neighbors
PES	– production and economic systems
ARIMA	– autoregressive and integrated moving average model

Introduction

Based on the international rating of production competitiveness for 2016, the greatest impact on the competitiveness of industrial companies has had the application of predictive models in management. This allowed the USA and Chinese industrial companies to take a leading position for the last 5 years. The role of using forecasts in management tasks is significantly increased in connection with the substitution of cognitive methods for statistical data processing. An important role in this is the reliability of forecasts. Therefore, the task of assessing the reliability of the forecast and assessing the risks of the received forecasts is relevant.

Initially, time series analysis was used only for models with a systematic component (in such models, the time course does not affect the random component).

Most of the parameters characterizing production and economic systems are the ability to describe the change in the values of most parameters using S-curves or periodic curves. However, lately there has been an increase in the mobility of the values of the parameters characterizing the production and economic systems. To predict such values, it is necessary to use methods that can self-correct in the light of current trends and respond quickly to changes in the values of production and economic systems.

Thus, when considering individual parameters (without considering methods of complex prediction of a number of parameters, such as PLS regression [1] and cognitive maps [2]), the prediction task will be reduced to the selection and application of the following methods:

- approaches used to predict data described by special curves (usually these are parameters described by innovative and S-shaped curves (see, for example, [3] and [4]);

- unsupervised learning (used for models with a large amount of training sample, when it is necessary to determine the internal relationships, dependencies, patterns existing between objects, is based on the application of the clustering method). One of the methods used for analysis is the k-means, hierarchical clustering, neural networks, autoregression method [5];

- supervised learning (used to determine the data interrelationship functions based on the training sample). The training sample consists of a set of input and output values. The methods used are the support vector method (SVM) [6], k-nearest neighbors algorithm (k-NN) , Gaussian mixture mode), naive Bayesian classifier naive Bayes, decision trees, radial basis function classifiers, etc.);

- semi-supervised learning (machine learning methods that simultaneously use a training sample with unlabeled data and a small number with tagged, popular approaches like: self-training, co-training, graph-based algorithms Based algorithms, generative models, Cluster-and-Label) [7].

The purpose of this work is to build a forecast and assess the risk of the values of the parameters of serial production in production and economic systems (PES).

For this you need to solve the following tasks:

- Prepare data for further analysis, restore missing values, expand data by years using seasonal coefficients;

- Apply methods of regression analysis, support vector method, autoregression, wavelet analysis and fractal analysis to predict values;

- Based on the forecasts received, calculate the value of the error and the risk of the predicted values.

Relevance of the research: effective management of production and economic systems is possible only with the use of predictive models, this is due to the mobility of the system parameters. In this case, there are risks of forecasting, which are also worth considering.

Chapter 1. Basic approaches to working with time series

1.1. Basic methods of forecasting time series

The questions of forecasting data characterizing production and production and economic systems are relevant because such information is used in management tasks, both by experts and in models used to support decision making. At present, approaches are being developed on the basis of predictive models combining approaches based on the principles of optimal control and predicting the values of parameters, introducing a new time factor in traditional tasks. In such problems, the accuracy of the predicted values will acquire special significance. It should be borne in mind that it is through forecasting values in such tasks that the influence of the external environment is taken into account and indirectly through it the influences of other "players" in the subject domain of production and or market. Therefore, work with such systems will be cyclical, due to the need to refine the forecasts when new data appear. This factor becomes especially important in the context of data limitations, as well as in connection with the characteristics of the statistics that are collected, which usually does not allocate each unit of product name and modification, so that even if the nature of the data change is known, this information becomes difficult to apply.

At the moment, the following methods and approaches are used to solve the tasks of forecasting time series:

- The approaches used to predict the data described by special curves (as a rule, these parameters are described by innovative and S-shaped curves, see [8], [9], [10], [11]

- Unsupervised learning. One of the ways of machine learning, in which the test system spontaneously learns to perform the task without intervention from the experimenter. He studies a wide class of data processing tasks, in which only descriptions of a set of objects (learning sample) are known, and it is required to detect internal relationships, dependencies, patterns existing between objects. This approach is based on the application of the clustering method. Methods used for analysis: k-means, hierarchical clustering, neural networks.

- supervised learning. These methods are based on the use of labeled training data. These data have input and output parameters. Based on which a function is constructed, which can be used later to predict the values of new, unlabeled data. This type of training uses classification methods. The main methods used in this method are neural network, support vector method (SVM), neighbor nearest-neighbor algorithm (k-NN), Gaussian mixture model, naive Bayes classifier, decision trees and radial basis functions of classifiers. [12]

- Semi-supervised learning is a method of machine learning that simultaneously uses the elements of supervised and unsupervised learning. Typically, the system uses a small amount of label data and a large amount of unlabeled data. The combination of two techniques of machine learning can significantly improve the accuracy of training. Also, the use of this method simplifies and reduces the cost of preparing the labeled data. Because processing with a teacher a large number of data is either impossible, or very expensive. In such situations, semiautomatic training can be of great practical importance. [13]

As methods for teaching with partial involvement of the teacher, classification algorithms can be distinguished: self-training, co-training, graph based algorithms, generative models, Cluster-and-Label.

1.2. Features of time series in production and economic systems

The first step in any analysis is the collection and data preparation. There is a widespread opinion that up to 80% of the data analysis process is the time spent on their preparation. Data preparation is not only the first step, but they must be repeated many times during the analysis as new problems are identified or new data are collected. Part of the problem is the breadth of the work from the systematization of data, the reduction of data from different sources, the restoration of missing values.

When analyzing data on projects implemented in production and economic systems, one has to work either with a small amount of statistical data, or with data describing a number of products, product modifications, etc. (I.e., unlabeled data). In this case, as a rule, there are periods when several types or modifications of the product were made simultaneously. The data can be characterized by a different frequency of data collection (different time intervals between values). In addition, data may be contained in different sources and may be incomplete or have errors.

Such situation with statistical data can be resolved either by using additional information about the analyzed data, or by using methods to increase the statistical sample.

Additional information is expressed in information about the type of the curve which describes data or data areas. Knowledge of the laws of changing values allows us to determine the form of the function and reduce the solution of the problem to the search for the coefficients of functions for the best description of the data.

The innovation curve is a piecewise-defined function: exponential growth, linear section, parabolic maturity. The system of equations describing the curve (1.1) [14]:

$$\begin{cases} f_1(t) = e^{c_0 t}, & 0 < t < t_1 \\ f_2(t) = c_1 + c_2 t, & t_1 < t < t_2 \\ f_3(t) = c_3 + c_4 t + c_5 t^2, & t_2 < t < t_3 \end{cases} \qquad (1.1)$$

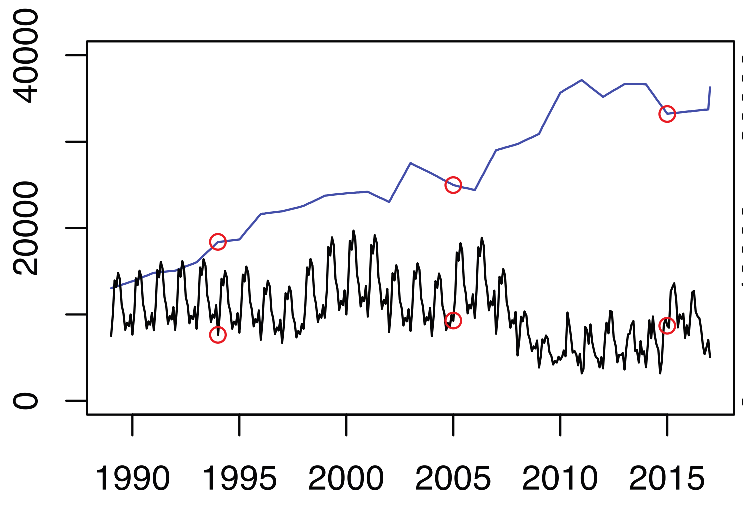
where the coefficients c determine the position, shape and growth (fall) of the function and depend on the features of the innovative project. The t_1, t_2, t_3 – the time corresponding to the transition points from one stage of the innovation project to another.

The task of determining the points of transition from one stage to another stage of model modification development can be solved by an expert method based on the features of the innovation curve. Such as [15]: the area of the figure, limited by a segment of exponential growth (entering the market), is 3% of the total area of the figure; The area of the figure bounded by the linear growth area is 13% of the total area of the figure; The area of the figure bounded by the maturity site is 34% of the total area of the figure; The area of the figure, bounded by the recession area, is 16% of the total area of the figure.

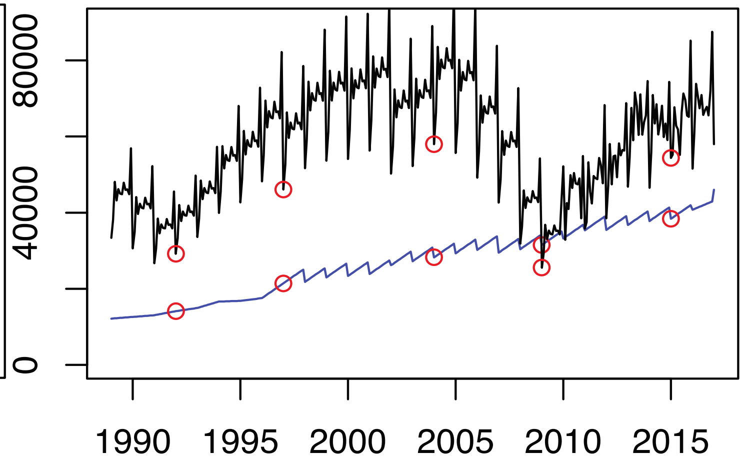
When predicting the values of the parameters described by S-shaped curves, questions arise about the origin of the coordinates of such curves, the location of the bending point, the choice of the functional description (the Pearl curve, Gompertz curve, etc.). These data can be determined in the same way as in the previous case by the expert method.

In this case, there is a problem that the collected data characterize several projects or models. The application of methods based on innovative and S-shaped curves in this case probably introduces features related to the need to determine the points of the generation change (see, for example, Fig. 1.1) by expert methods, special processing at the stages of transition from one generation to another, a short period of time Forecasting limited to one stage of the project [16]. Everything becomes difficult (see, for example, Figure 1.1), and the accuracy of the methods existing on the basis of this approach is not high in connection with the need to determine the points of transition from the release of one model to another and the absence of it [17]

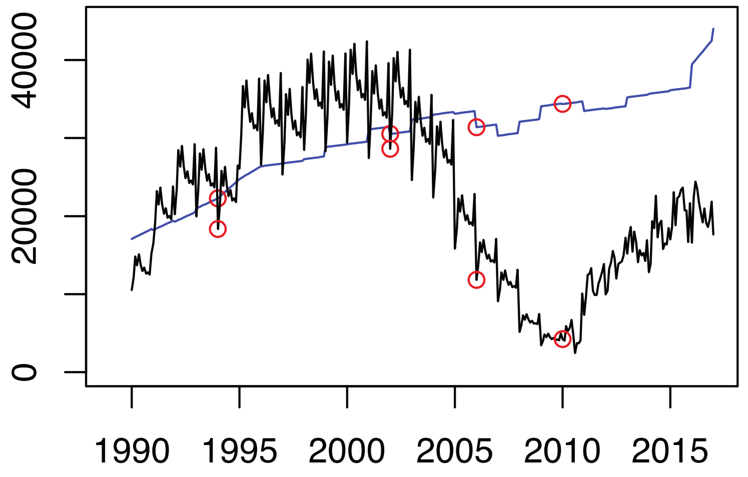
.



a)



b)



c)

Fig. 1.1. Change in price value (blue), sales value (black) and red points - generation change for a) Ford Mustang car (1979-1993 - third generation, 1994 - 2004 fourth generation, 2005-2014 - fifth generation, 2015 sixth generation b) car Ford F-Series, c) car Ford Explorer..

In this case, the information about the change in the model parameters within the model range is known and given in Table 1.1. The generational change points are shown in red, in Figure 1.1.

Table 1.1.

Data about the change of the model range for Ford cars models: Mustang, F-series and Explorer

Mustang	**F-series**	**Explorer**
The third generation (1979-1993) 1992 - 1993, increase the power from 88 to 105 hp. The fourth generation (1994-2004) 1994 - 1995 "SN-95", Serious change 1996 - 1998 8-cylinder 1998 Engine power increased 1999-2004 "New Edge 2001 - Power increased Fifth generation 2005 – 2014 2005 «S-197» 2006-2008, Shelby GT 2010 Updated model 2011 Engine Update 2012 New Mustang Boss 302 introduced The Sixth Generation (2015)	Eighth generation 1987 * 1991 The ninth generation (1992-1997) 993 SVT Lightning The tenth generation (1997-2003) The eleventh generation (2004-2008) The Twelfth Generation (2009-2014) The thirteenth generation (2015) 2017 - Restyling	First generation 1990 – 1994 Second 1994 * 2001 Third 2002-2005 Fourth 2006-2010 Fifth 2010- Now

1.3. Features of data preparation when working with time series

Due to the peculiarity of data in which there is no selection of goods and their modifications and different frequency of compiling reports and collecting data, an important stage is the preparation of data. In addition to this reason, the data may be contained in different sources, be incomplete or have errors. Therefore, before working with data, it is so important to properly prepare them.

Let's consider some peculiarities of working with data characterizing production and economic systems on the example of open data on the volume of sales of Ford cars in the US market.

The aim of the study is to predict and assess the risk of the values of the parameters of serial production in PES.

Data for analysis are taken from open sources: WolframAlpha and a site containing sales statistics for various brands of cars for the markets of the USA and Canada (http://www.goodcarbadcar.net/2011/01/ford-crown-victoria-sales-figures.html) and Another site (http://carsalesbase.com/us-car-sales-data/ford/ford-mustang).

Since the data is taken from different sources and presented in a different format (xls document and web page). The original data was tabulated in the xls format. The data is filled manually, using the Microsoft Excel program (Table 1.1)

The data shown in Table 1 has gaps. To use such data in predictive models, they require a synchronization operation.

The filling of the gaps between 1989 and 1994 can be accomplished by extrapolating the data by the number of sales. Such extrapolation can be performed both by mathematical methods and by an expert method. However, working with the received data, we observe that some of the data is specified with a periodicity in a month and a part with a periodicity in a year. When choosing the size of the period per year, the amount of data can be so insignificant that the use of mathematical methods will be difficult. Therefore, further it is necessary to deploy by months based on seasonality of sales. Data for the period from 1989 to 2009 inclusive are given for the year, and from 01/01/2010 to 01/01/2017, data are presented for each month. We expand the data based on the seasonal coefficients for each model.

Table 1.2.

An example of data used for analysis.

Date	Sales value in USA Mustang	Price average Mustang	Price min Mustang	Price Max Mustang	Sales value in USA F-150	Price average F-150	Price min F-150	Price Max F-150	Sales value in USA Explorer	Price average Exp-r	Price min Exp-r	Price Max Exp-r
01.01.1989		13026	9050	17512								
01.01.1990		13838	9456	18805		12607	10366	14933				
01.01.1991		14813	10157	19864		13051	10398	15857		18216	14792	22125
01.01.1992		15069	10215	20199		14128	10336	17296		19284	15854	23553
01.01.1993		16034	10810	20950		14963	11138	18923		20991	16690	27422
01.01.1994		18389	13695	22195		16646	14256	18971		22284	17970	28910
01.01.1995		18667	14530	22795		16833	14306	19294		24739	18985	33935
01.01.1996		21623	15180	27580		17551	15150	19840		26385	19770	34950
01.01.1997		21928	15355	28135		21461	15525	27115		26754	20085	35005
01.01.1998		22553	16150	28510		21800	15865	27415		27298	19880	33720
01.01.1999		23751	16470	31470		23018	16220	29565		28841	20065	34540
01.01.2000		24024	16710	31605		23366	16455	32305		29209	19970	34565
01.01.2001		24200	17095	32605		23877	17245	31745		31098	25210	34660
01.01.2002	138356	23007	17475	28645	813701	26188	18540	35995	433847	30524	24585	34510
01.01.2003	140350	27520	17720	39275	845586	27237	19125	37035	373118	32353	26285	37000

Seasonal fluctuations are characterized by seasonality indices (I_S). The annual parameter of seasonal fluctuations forms a seasonal wave. The seasonality index refers to the percentage of actual annual levels within an annual average.

To identify seasonal fluctuations, data are usually taken over several years, distributed over the months. These data, usually used in a sample over a period of three years, are used to identify a stable seasonal wave. Sampling of the data for several years allows to avoid the influence of random annual parameters and to represent the average seasonal wave.

When calculating the seasonality indexes, different methods are used. In the absence of pronounced seasonal trends, the values of the indices are calculated on the basis of the initial empirical data, without additional alignment.

Then the calculation algorithm has the following form: the average value of the index is calculated for each month, in this case for seven years ($\overline{y_\iota}$), then the average monthly level of the series for the year ($\bar{y}$) is calculated from the average values of the month and, as a result, Values for the month to the total average monthly value (1.2), i.e. [12]

$$I_S = \frac{\overline{y_\iota}}{\bar{y}} \cdot 100\% \qquad (1.2)$$

The sum of the coefficients should be 1200, with the correct calculation.

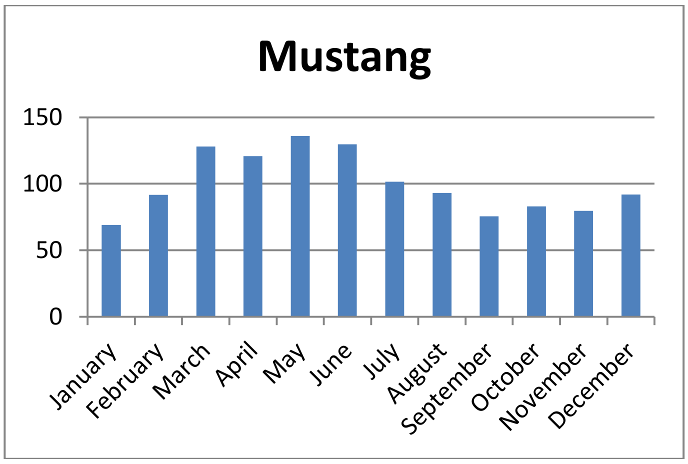
Next, we need to restore the value of the amount of sales of each brand for a month, taking into account the seasonal coefficients. To do this, we divide the value of annual sales by 1200 and multiply by the coefficient of each month. We perform this operation for each model. (Fig.1.2)

As a result, we have 337 rows of data. In final, we have restored the intermediate data for a month; also for the further investigation it is necessary to restore the data for the intervals at the beginning and end of the time periods. To do this, we again apply the expert method. Since the intervals needed for recovery are small, a maximum of 24 points out of 338. To do this, we use the linear extrapolation method. The essence of the method lies in the fact that the forecast values are determined on the basis of the average increase (decrease) in the indicator under study over a certain period of time. [13] Forecasting is carried out on the average absolute increase, and can be performed if the general trend is assumed to be linear.

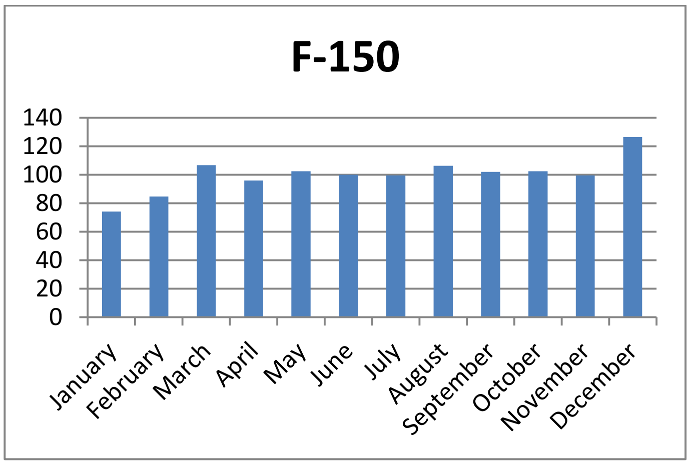
If the main trend of the changes in the sign is close to rectilinear, then the value of the level of the series in the subsequent period (n +1) can be approximately determined by formula (1.3):

$$y_{n+1} = y_n + \overline{\Delta y}, \qquad (1.3)$$

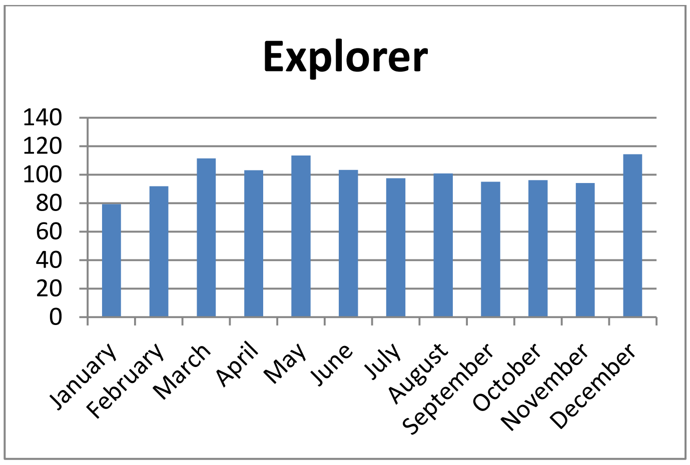
where: $\overline{\Delta y}$ - is the average absolute increase. [18].



a)



b)



c)

Fig.1.2. Seasonal coefficients for 3 selected Ford cars' models

In this case, the use of this method is acceptable, because we need to rebuild a small area, while, judging by the graph of 1, on the interval at the beginning of the time range the graph is close to the linear function. Therefore, the use of linear extrapolation is justified (Table 1.3, Fig.1.3).

Table 1.3.

Values of R^2 linear approximation

	Price min F-150	Price max F-150	Price min Explorer	Price max Explorer
R^2 linear approximation	0.9457	0.9732	0.9397	0.7651

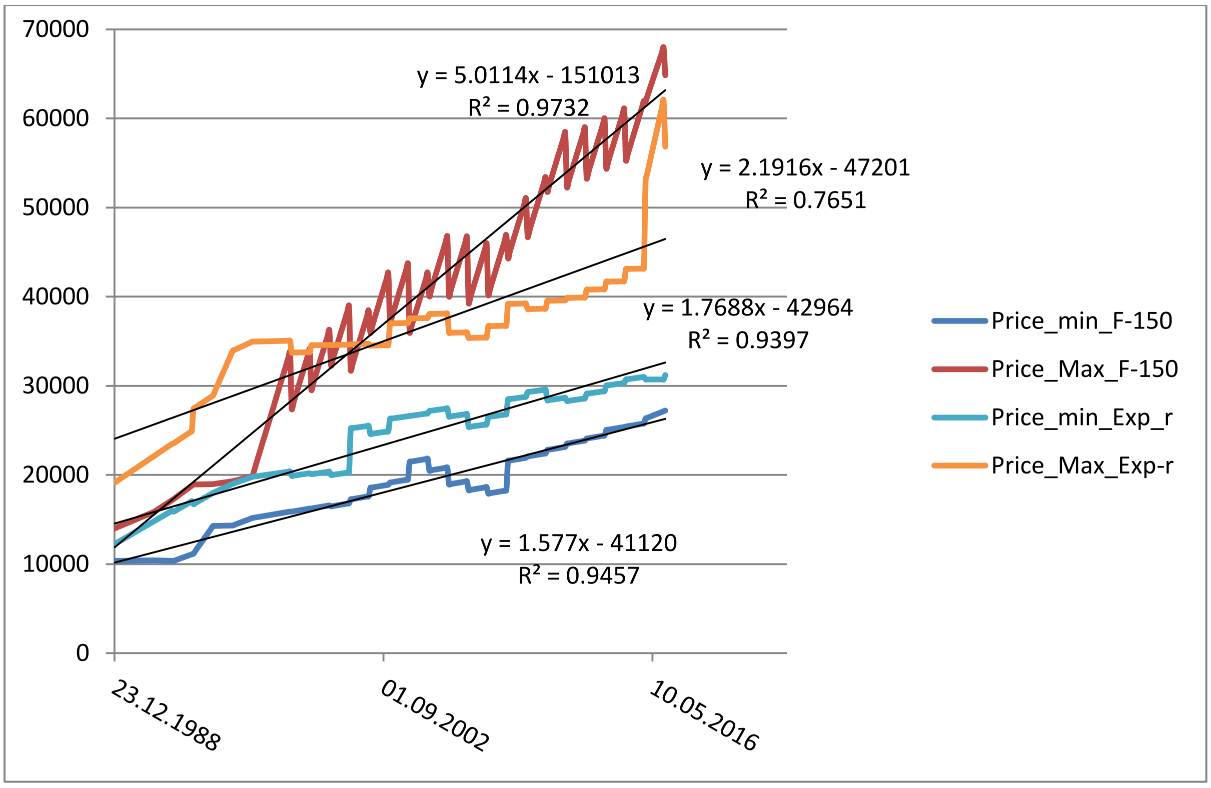


Fig.1.3. Graphs of functions for which approximation is necessary

As a result, data were obtained for further analysis using linear extrapolation, linear interpolation, and seasonal coefficients.

Chapter 2. Data forecasting described by time series in production and economic systems

Each method has features and accuracy of adjustment. Therefore, often the problem arises not only of choosing a method that is optimal for these conditions, but also of selecting parameters for them. To select the optimal model for forecasting and its parameters, it is suggested to divide the retrospective data into two sections: training and test. After the preliminary method, it is necessary to refine the internal parameters already on the combined data.

2.1. Using regression analysis to predict data

Regression is the mathematical expectation of a random variable on one or more variables (other random variables) (2.1), that is,

$$E(y|x) = f(x) \tag{2.1}$$

Regression analysis is the search for a function f that describes this dependence. Regression can be represented as a sum of non-random and random components (2.2).

$$y = f(x) + v \tag{2.2}$$

where f is a regression function, and v is an additive random variable with zero expectation. The assumption of the nature of the distribution of this quantity is called the data generation hypothesis. It is usually assumed that the quantity v has a Gaussian distribution with zero mean and variance σ_v^2.

The problem of finding the regression model of several free variables is set as follows. A sample is defined-the set $\{x_1, \dots, x_N | x \in R^M\}$ of values of free variables and the set $\{y_1, \dots, y_N | y \in R\ \}$of the corresponding values of the dependent variable. These sets are denoted as D, the set of initial data $\{(x, y)\}$. A regression model is defined-a parametric family of functions $f(w, x)$ depending on the parameters $w \in R$ and free variables x. It is required to find the most probable parameters $\overline{w}$ (2.3):

$$\overline{w} = argmax_W p(y|x, w, f) = p(D|w, f) \tag{2.3}$$

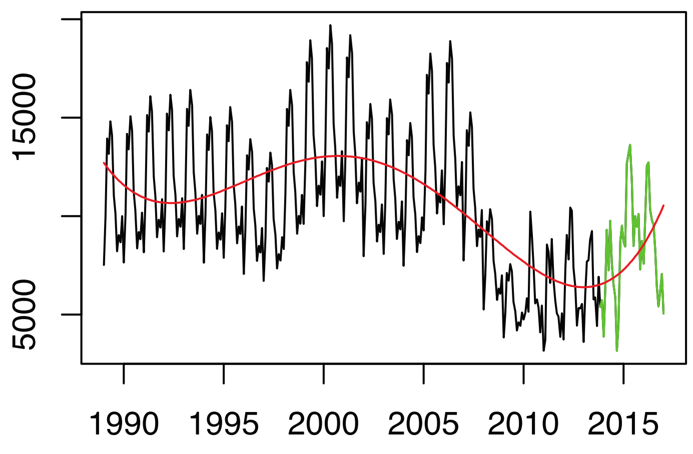
The probability function p depends on the hypothesis of data generation and is given by the Bayesian derivation or the maximum likelihood method. [19] The quality of predicting the regression model depends on the type of function chosen.

Let's construct a regression model for the sales volume of each car model (Listing 2.1, Figure 2.1).

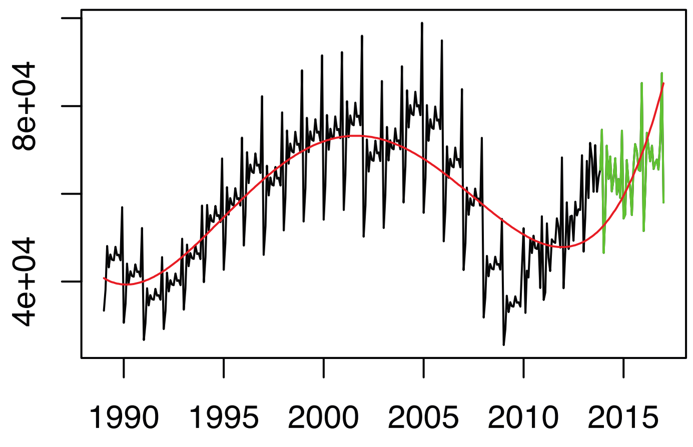
The disadvantages of the method include the need to select the form of the function, and in the event that the data have complex regularities, as in the case of the parameters of production and economic systems, it is necessary to use power series that give low accuracy beyond the range in question with the data, and hence the poor quality of the forecast.

Listing 2.1. An example of regression analysis code in the language R, for model F-Series.

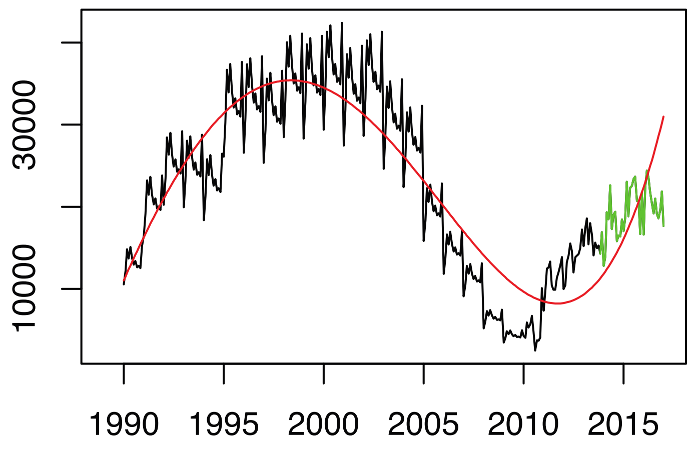
```
plot(pr_av_f,sf)
  points(test[,7],test[,6], col= 'green')
  carlf <-lm(formula = sf ~ pr_av_f)
  lpr_f<- predict(carlf)
  summary(carlf)
  abline(carlf, col='red')
  error <- sf - lpr_f         # error linear predict mustang
  predictionRMSE <- rmse(error)     # 14046.504 root-mean-square
deviation for mustang linear regression
  dost_f_av<- sigm(error)
  print(dost_f_av)
  #for sales only
  x<- 1:337
  e<-data.frame(x=x,y=sf)
  lm_f_s<- nls(y ~ k1+k2*x+k3*x*x+k4*x*x*x+k5*x*x*x*x, data=e,
start=list(k1=0.1, k2=0.1, k3=0.1, k4=0.1, k5=0.1))
  lpr_tr_f_s<- predict(lm_f_s)
  coef_func <- coef(lm_f_s)
   lpr_tr_f<- predict(lm_f_s)
  plot(date,sf, type = "l", ylab = "Sales F-series")
  points(date_t,test[,6], col= 'green', type="l") # test data
points
  lines(date, lpr_tr_f_s, col='red')
```



a)



b)



c)

Fig. 2.1. Forecasting of regression sales volume - initial data (black), data for forecasting (green) and data obtained by regression (red line): a) Ford Mustang car, b) Ford F-Series car, c) Ford Explorer car.

2.2. Using supervised learning method

2.2.1. Support vector machine method

The Support Vector Machine (SVM) is supervised learning models with associated learning algorithms used for classification tasks and regression data analysis. It belongs to the family of linear classifiers and can also be regarded as a special case of Tikhonov regularization. A special property of the support vector method is a continuous decrease in the empirical classification error and an increase in the gap, so the method is also known as the classifier method with the maximum gap.

The main idea of the method is the translation of the initial vectors into a space of higher dimension and the search for a separating hyperplane with the maximum gap in this space. Two parallel hyperplanes are constructed on both sides of the hyperplane that separates the classes. The separating hyperplane is a hyperplane that maximizes the distance to two parallel hyperplanes. The algorithm works under the assumption that the greater the difference or the distance between these parallel hyperplanes, the smaller the average classifier error [20].

Describing the rules of classification in its unconditional form shows that the maximum margin of the hyperplane and, consequently, the classification problem is only a function of support vectors.

Observations for training lie on the edge.

Using the fact (2.4)

$$\|w\| = w * w \tag{2.4}$$

And substituting (2.5),

$$\boldsymbol{w} = \sum_{i=1}^{n} \boldsymbol{a_i y_i x_i} \tag{2.5}$$

It can be shown that the second form of the support vector method allows to solve the optimization problem:

Maximizing with respect to a_i (2.6):

$$L(a) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i,j}^{n} a_i a_j y_i y_j x_i^T x_i = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i,j}^{n} a_i a_j y_i y_j k(x_i, x_i)$$

$$a_i \geq 0 \tag{2.6}$$

The restriction on minimization for b (2.7)

$$\boldsymbol{\sum_{i=1}^{n} a_i y_i = 0} \tag{2.7}$$

The kernel is defined as k $k(x_i, x_j) = x_i, x_j$

W can be computed by conditions (2.8):

$$\boldsymbol{w = \sum_{i=1}^{n} a_i y_i x_i} \tag{2.8}$$

However, in our case, an explicit nonlinearity of the function is observed. For such cases, a non-linear classifier is used. In 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik proposed a method for creating a nonlinear classifier using an arbitrary kernel trick (proposed by Eiserman) to find the planes of maximum difference.

The final algorithm is formally simple, except that each scalar product is replaced by a nonlinear kernel function. This allows us to find the hyperplane of the maximum difference in the changed function space. [20]

The change can be non-linear and can be transformed into a space with high dimensionality.

Although the classifier is a hyperplane in a multidimensional function space, it can be non-linear in the source space of the training sample.

If the kernel used Gaussian radial basis functions, the corresponding function space is a Hilbert space with infinite dimension.

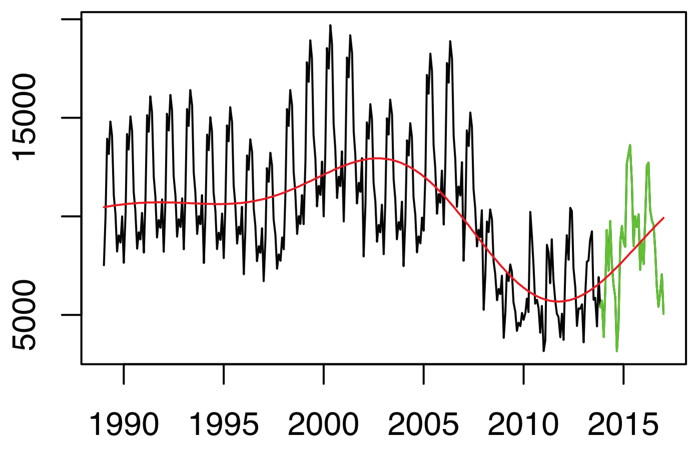
The classifier of the maximum difference is well settled, because Infinite dimension will not spoil the results. It is worth noting that the accuracy of the received forecasts depends on the volume of the training sample.

We obtain a function for constructing the forecast obtained by the support vector method for the Mustang model (Listing 2.2). To build the forecast of the remaining models, we apply the same algorithm as in the listing. 2.2.

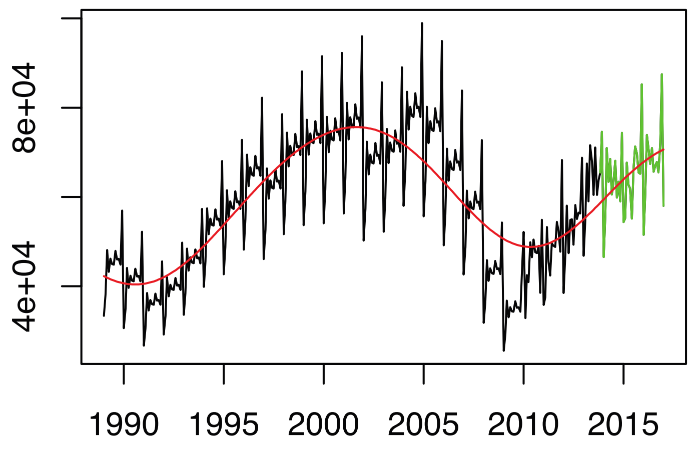
Listing 2.2 the program code for building the support vector method for the Mustang model in RStudio

```
f.svm_m_av<- function(test,d,sm) #svm {
  #from time sales only
  for (i in 1:length(test)){
    test[i,14]<-    as.numeric(as.Date(test[i,1],    format    =
'%d.%m.%Y'))}
  date_t<- as.Date (test[,1], "%d.%m.%Y")
    model1<- svm(sm ~ date, d)
  prY2 <- predict(model1,d)
  plot(date,sm, type = "l")
  points(date_t,test[,2], col= 'green', type = "l") # test data
points
  points(date, prY2, col = "red", type ="l")}
```

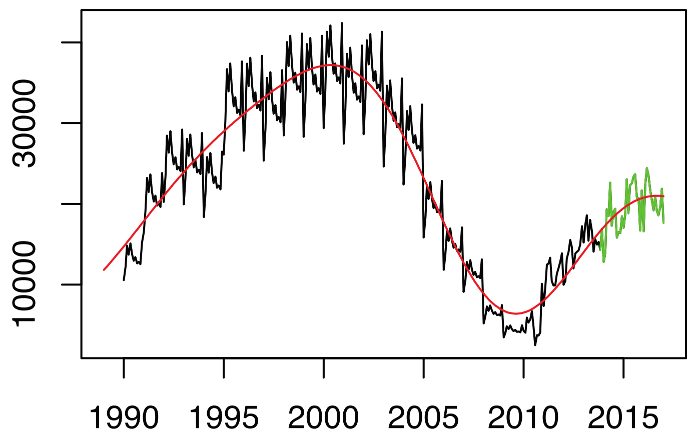
Let us consider the results that can be obtained by the support vector method (Fig. 2.2) using data with which we supplemented our samples.



a)



b)



c)

Fig.2.2. Forecasting the volume of sales by the method of reference vectors - initial data (black), data for checking forecasting (green) and data obtained by the support vector method (red line): a) Ford Mustang car, b) Ford F-Series car, c) Ford Explorer car .

2.3. Using semi-supervise learning methods

2.3.1. Autoregression

The simplest method is the autoregressive method. Forecasting using the autoregression model is based on previous sales values. The word autoregression means the dependence of the subsequent value of sales on previous sales. Dependence in the case of autoregression is assumed to be linear, that is, the forecast is the sum of sales for the previous days with some coefficients that are constant and determine the parameters of the autoregression model. How many days (periods in the general case) of such sales from the past we will take to try to predict future sales is called the order of the autoregression model p. For example, we want to enter the entrance to sell for the previous three days, and predict how much will be sold the next day. Then, the order of the model is p = 3. In brief this is often written as AR (3). The formula will have the form (2.9) [21]

$$Y_t = c + \varepsilon_t + a_1 * Y_{t-1} + a_2 * Y_{t-2} + a_3 * Y_{t-3} \tag{2.9}$$

In this work, were used the autoregressive and integrated moving average model (ARIMA.). ARIMA model is used to model non-stationary time series. This method is well suited for describing periodic data, but it does not allow taking into account fluctuations in the data.

The non-stationary time series is characterized by unstable mathematical expectation, variance, autocovariance and autocorrelation.

At the heart of the ARIMA model are two processes:

• autoregression process;

• moving average process.

The ARIMA process can be determined by the formula (2.10):

$$xt = a + \delta 1xt - 1 + \delta 2xt - 2 + \cdots + \varepsilon t, \tag{2.10}$$

where a is a constant, a free member of the model;

$\delta 1$ $\delta 2 \ldots$- parameters of the autoregression model;

ε - random effect (model error).

Any observation in the autoregression model is the sum of the random component and the linear combination of the previous values

The moving average process can be described by a formula of the form (2.11):

$$xt = \mu + \varepsilon t - \theta 1 \varepsilon t - 1 - \theta 2 \varepsilon t - 2 - \ldots - \varepsilon t \quad (2.11)$$

where μ is the free term of the model, constant;

$\theta 1$ $\theta 2$…... - parameters of the moving average model;

ε - random effect (model error).

The current observation of ARIMA consists of the sum of the values of the initial component at the current time and a combination of random effects at previous time instants.

As a result, the ARIMA model is presented in general form in the formula (2.12):

$$y_t = C + \sum_{i=1}^{R} A\, R_i y_{t-i} + \sum_{j=1}^{M} M\, A_j \varepsilon_{t-j} + \varepsilon_t \quad (2.12)$$

where C is the free term of the model, a constant;

$\varepsilon_t$is a random residue uncompensated by the model.

In the works of Box and Jenkins, the autoregression model and the integrated moving average are presented in the form ARIMA (p,d,q), where

- p - parameters of the autoregression process;
- d is the order of the difference operator;
- q - parameters of the moving average process.

When modeling ARPSS models in non-stationary time series, three stages of modeling.

1. Checking the time series for stationarity;
2. Identification of the order of the model and estimation of unknown parameters;
3. Forecast.

When using the autoregression model of the integrated moving average, it is necessary to verify the stationary of the investigated series. This assumption is verified using autocorrelation and partial autocorrelation functions of a number of residues. Remnants mean the difference between the observed time series and the values calculated using the model.

The nonstationarity of the time series can be eliminated by the method of difference operators.

By a difference operator of the first order we mean the replacement of the initial level of the time series by first-order differences (2.13):

$$\nabla_1^t = y_{t+1} - y_t \qquad (2.13)$$

The application of first-order difference operators makes it possible to eliminate linear trends. And the application of difference operators of the second order makes it possible to exclude parabolic trends.

Seasonal difference operators are used to exclude 12- or 4-period seasonality (2.14):

$$\nabla_1^t = y_{t+12} - y_t, \text{ или } \nabla_1^t = y_{t+4} - y_t \qquad (2.14)$$

If the model contains both trend and seasonal components, then both operators must be used.

At the second stage, the number of autoregression parameters and the moving average are selected.

In the process of estimating the ARIMA model, a quasi-Newton algorithm is used to maximize the likelihood of observing the values of the series in terms of the parameter values. In this case, the (conditional) sum of squares of the remainders of the model is minimized. To assess the significance of the parameters, Student's t-statistics are used. If the values of the obtained t-statistics are not significant, the corresponding parameters, as a rule, can be deleted from the model without affecting the total values of the system. The obtained parameter estimates are used at the last stage to calculate the new values of the series and construct the confidence interval for the prediction.

The accuracy of the obtained forecast based on ARIMA model is determined using the mean-square error (2.15):

$$\nabla_1^t = y_{t+1} - y_t \qquad (2.15)$$

The smaller the error value, the more accurate the forecast.

ARIMA model is considered adequate to the initial data if the remnants of the model are uncorrelated normally distributed random variables. [22]

Let's apply the autoregressive method in the RStudio, build the forecast for 39 points (the amount of data in the test file) and present the result as a graph for the Mustang model (Listing 2.3):

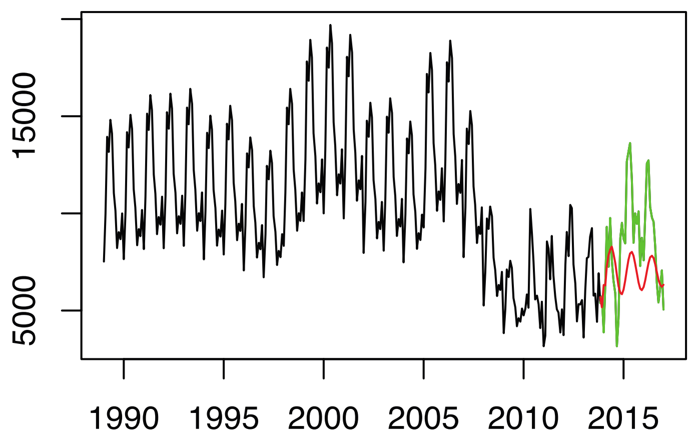
Listing 2.3 the code of the program for constructing the autoregressive method for the Mustang in the RStudio program

```
      model<- auto.arima(smt)
       future<- forecast(model, h=39)
       plot(date, sm, type = "l")
       points(date_t,test[,2], col= 'green', type = "l") # test
data points
       points(date_t, future_ar_m$mean, type="l", col="red")
```

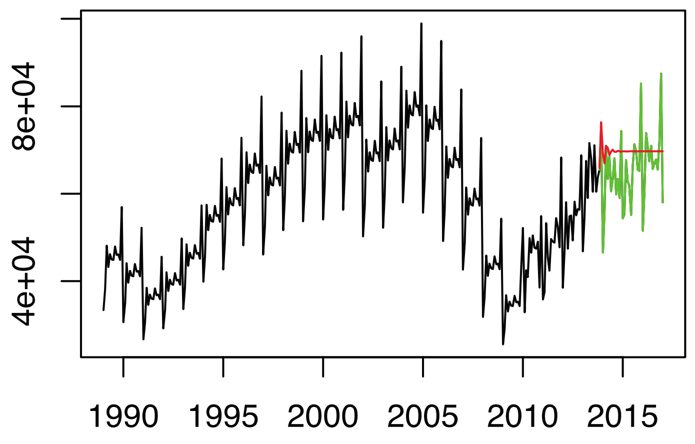
Result of autoregression calculation:

```
Series: smt
ARIMA(4,1,3)
Coefficients:
         ar1     ar2      ar3      ar4      ma1      ma2
ma3
      0.7200  0.3054  -0.2458  -0.4009  -1.1780  -0.1499
0.5396
s.e.  0.1256  0.1856   0.1027   0.0575   0.1309   0.2354
0.1232

sigma^2 estimated as 2698607:  log likelihood=-2619.14
AIC=5254.27   AICc=5254.77   BIC=5283.82
```

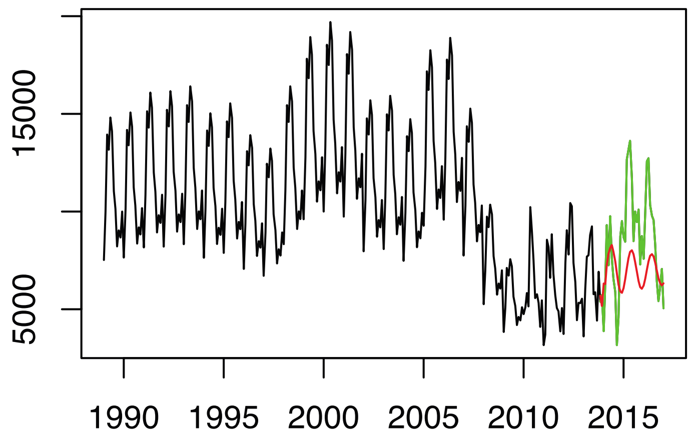
The forecast command predicts values based on the training data, the result of the calculation of the representation in Figure 2.3.



a)



b)



c)

Fig. 2.3. Forecasting sales volume using the autoregressive method - initial data (black), data to check the forecasting (green) and data obtained by the autoregressive method (red line): a) Ford Mustang car; b) Ford F-Series car; c) Ford Explorer car.

The prediction function forecast also builds the value of the prediction interval. The prediction interval should not be confused with a confidence interval.

The prediction interval is the interval associated with a random variable that is still to be observed, with a given probability of a random variable lying within the interval. Prediction intervals can occur in Bayesian or frequency statistics.

The confidence interval is the interval associated with the parameter. The parameter is considered non-random, but unknown, and the confidence interval is calculated from the data. Since the data is random, the interval is random. The 95% confidence interval will contain the true parameter with a probability of 0.95. That is, with a large number of repeated samples, 95% of the intervals will contain the true parameter. [23]

2.3.2. Wavelet analysis

Another method for analyzing complex functions is Fourier analysis. It is based on Fourier series. In accordance with the principle of interference, the series begins with the decomposition of a complex form into simple ones. Fourier showed how these simple solutions can be summed up to obtain a solution to the whole complex problem as a whole. (In the language of mathematics, the Fourier series is a method of representing a function by the sum of harmonics - a sinusoid and a cosine wave, so Fourier analysis was also known as "harmonic analysis.")

The Fourier transform allows us to represent a continuous function f (x) (signal) defined on the interval {0, T} as the sum of an infinite number of (infinite series) of trigonometric functions (sine and / or cosine) with certain amplitudes and phases, also considered on the segment {0, T}. Such a series is called the Fourier series. [24]

Another method of training the model is the application of wavelet analysis. Wavelet is a mathematical function that allows analyzing various frequency components of data. The graph of the function looks like wave-like oscillations with an amplitude decreasing to zero far from the origin. However, this is a particular definition - in the general case, signal analysis is performed in the plane of wavelet coefficients (Scale-Time-Amplitude). The wavelet coefficients are determined by the integral transformation of the signal. The wavelet spectrograms obtained differ fundamentally from the usual Fourier spectra in that they give a clear binding of the spectrum of various signal features to time [10].

One of the most promising methods remains the use of wavelet analysis. The wavelet spectrograms obtained differ fundamentally from the usual Fourier spectra in that they give a clear binding of the spectrum of various signal features to time (Fig. 2.4). The spectrogram allows you to select the periods that will have the greatest impact on the forecast of the values, after which for each curve of the reconstructed curve for the selected periods, forecast one of the methods giving the best result (in our case, using the autoregressive method). The accuracy of the forecast values depends on the selected periods. [25] The obtained data can be averaged to account for all components in the simplest case [26]. As a result, we obtain the data shown in Fig. 7. [27]

The wavelet analysis function for the Mustang model, in the RStudio program, is given in Listing 2.4.

Listing 2.4. The wavelet analysis function code for the Mustang model, in the RStudio program

```
my.data<- data.frame(t1=t1)  #sales

vs time
  plot(my.data, type = "l")
  my.w_m_sail<- analyze.wavelet(my.data, "t1.2",
                         loess.span = 0,
                         dt=1, dj=1/298,
                         lowerPeriod = 1,
                         upperPeriod = 128,
                         make.pval = F, n.sim=10)
  wt.image(my.w, n.levels = 298,
          legend.params = list(lab="wavelet power levels"))
library(WaveletComp)
my.rec3<- reconstruct(my.w_m_sail, sel.period = 3, show.legend =
F)
x.rec3<- my.rec3$series$t1.2.r
model3<-auto.arima(x.rec3)
future3<-forecast(model3, h=39)
plot(future3)
```

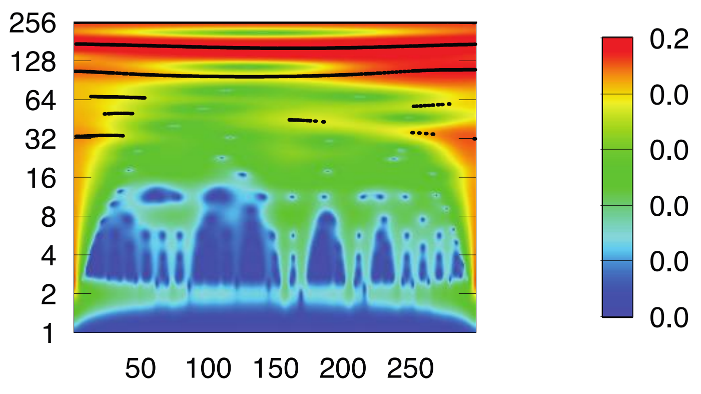
```
my.rec5<- reconstruct(my.w_m_sail, sel.period = 5, show.legend =
F)
x.rec5<- my.rec5$series$t1.2.r
model5<-auto.arima(x.rec5)
future5<-forecast(model5, h=39)
plot(future5)

my.rec8<- reconstruct(my.w_m_sail, sel.period = 8, show.legend =
F)
x.rec8<- my.rec8$series$t1.2.r
model8<-auto.arima(x.rec8)
future8<-forecast(model8, h=39)
plot(future8)

my.rec18<- reconstruct(my.w_m_sail, sel.period = 18, show.legend
= F)
x.rec18<- my.rec18$series$t1.2.r
model18<-auto.arima(x.rec18)
future18<-forecast(model18, h=39)
plot(future18)

my.rec135<- reconstruct(my.w_m_sail, sel.period = 135,
show.legend = F)
x.rec135<- my.rec135$series$t1.2.r
model135<-auto.arima(x.rec135)
future135<-forecast(model135, h=39)

plot(future135)

my.rec150<- reconstruct(my.w_m_sail, sel.period = 150,
show.legend = F)
x.rec150<- my.rec150$series$t1.2.r
model150<-auto.arima(x.rec150)
future150<-forecast(model150, h=39)
plot(future150)

my.rec256<- reconstruct(my.w_m_sail, sel.period = 256,
show.legend = F)
x.rec256<- my.rec256$series$t1.2.r
model256<-auto.arima(x.rec256)
future256<-forecast(model256, h=39)
plot(future256)
fin.rec<-NA
for (i in 1:298)
  fin.rec[i]<- sum (x.rec3[i], x.rec5[i],x.rec8[i],
                x.rec135[i], x.rec18[i],
                x.rec150[i],x.rec256[i]
  )/7
fin_m<-NA
for (i in 1:39)
  fin_m[i]<- sum (future3$mean[i],
future5$mean[i],future8$mean[i],
                future135$mean[i], future18$mean[i],
                future150$mean[i],future256$mean[i]
  )/7
plot(date, sm, type="l")
lines(date_t, test$Sales_value_in_USA_Must.g, col="green")
lines(date_t, fin_m, col="red")
lines(date1, fin.rec, col="red")
asd<- NA
asd[1]<- fin.rec[298]
asd[2]<- fin_m[1]
lines(date[298:299],asd, col="red" )
```

At the first stage we get a spectrogram for training values. On the basis of this spectrogram, the main periods are selected for further prediction of the system (Figure 2.4)

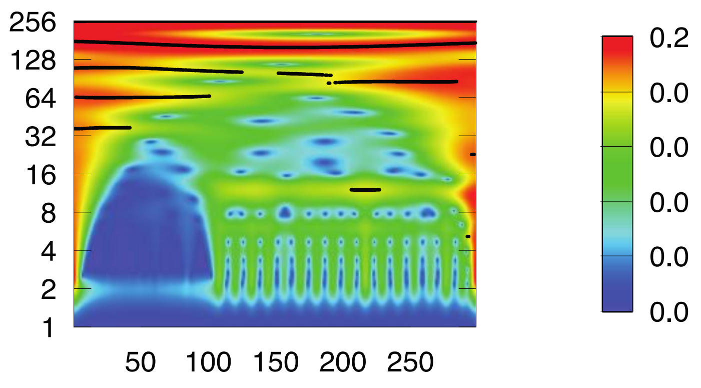
The next stage of data recovery for selected values based on the spectrogram. And learning with the help of any machine learning model, in this case it uses ARIMA autoregressive method, as one of the most effective for this model. And the subsequent forecasting. Below is an example for the 3 period, for the Mustang model (Listing 2.5, Fig.2.5).

Listing 2.5 Auto-regression program code for the Mustang in the RStudio program

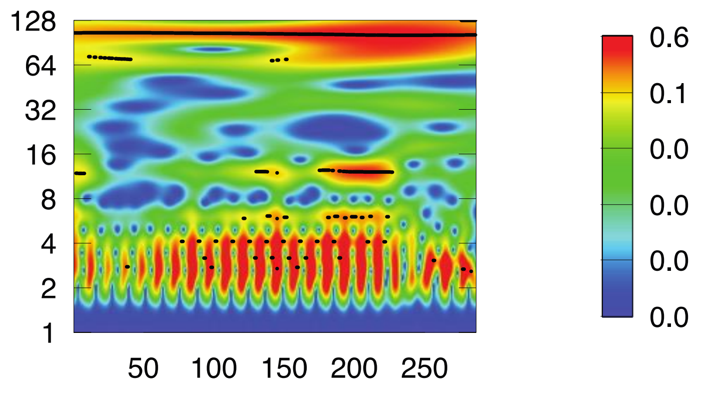
```
my.rec3<- reconstruct(my.w_m_sail, sel.period = 3, show.legend =
F)
x.rec3<- my.rec3$series$t1.2.r
model3<-auto.arima(x.rec3)
future3<-forecast(model3, h=39)
plot(future3)
```



a)



b)



c)

Fig. 2.4. Spectrogram for time (horizontally) versus period (vertical) for sales volume data: a) Ford Mustang car, b) Ford F-Series car, c) Ford Explorer car.

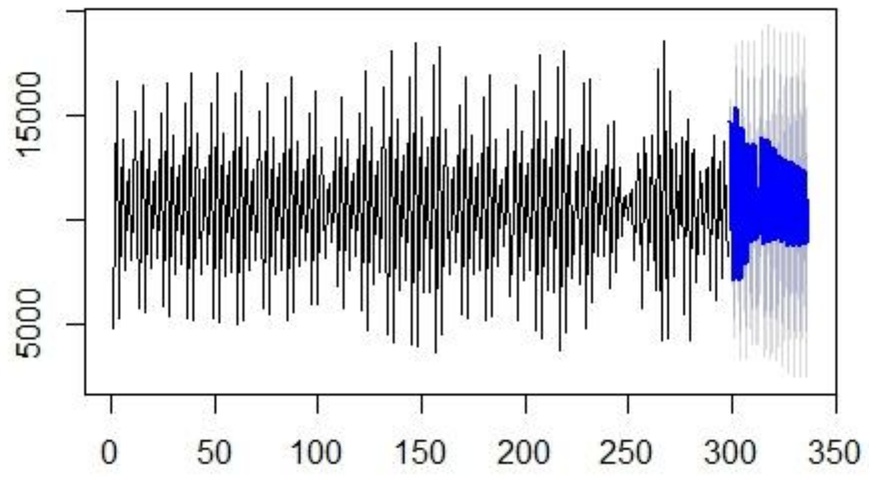


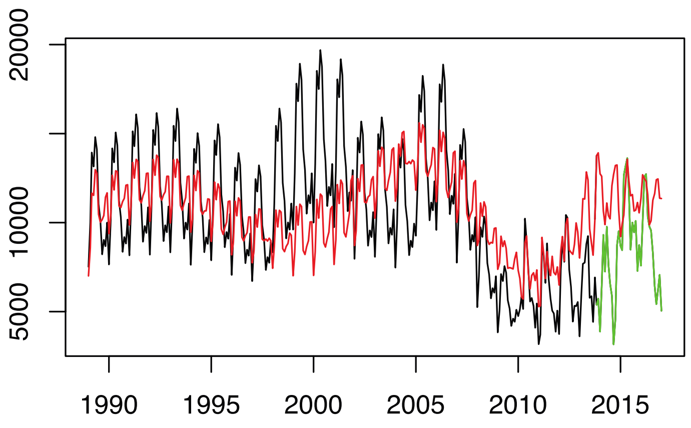
Fig. 2.5. Recover volume data sales data using the wavelet analysis method for a period of 3, the Ford Mustang

Restore the data for periods equal to 3, 5, 8, 18, 135, 150, 256 for the Mustang model. For the periods 3, 5, 8, 128, 150, 290 and for the periods 2, 3, 8, 28, 125, 135, 256 for the F-Series and Explorer models respectively.

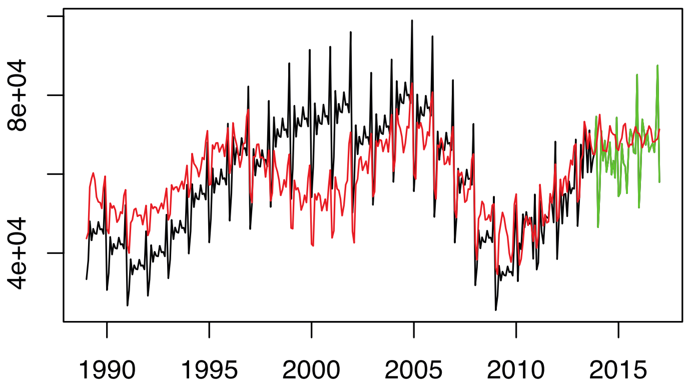
And we will calculate the average value based on the received periods.

As a result of such stages for all models and all periods, we get the predicted values of sales. (Figure 2.6)

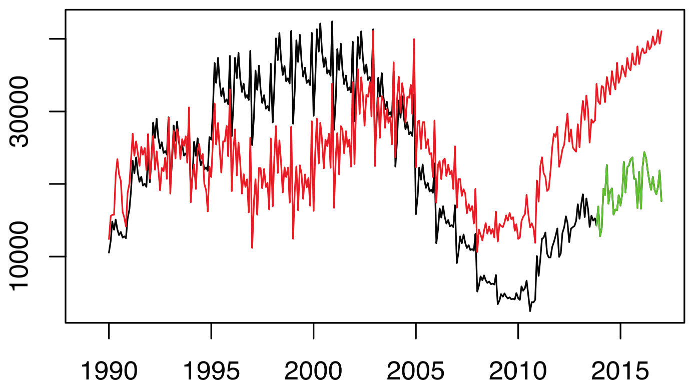
As can be seen from Fig. 2.6 The method works well for given periods but a big role, with the choice of the periods used that are selected manually. On the basis of this, it can be concluded that the adjustment of this method plays a significant influence on the results obtained.



a)



b)



c)

Fig. 2.6. Forecasting of the volume of sales by the wavelet analysis method - initial data (black), data for forecasting verification (green) and data obtained by the wavelet analysis method (red line): a) Ford Mustang car, b) Ford F-Series car, c) Ford Explorer car

2.3.3. Fractal analysis

Fractal (Latin fractus - fractured, broken, broken) is a mathematical set possessing the property of self-similarity (an object that exactly or roughly coincides with a part of itself, that is, the whole has the same form as one or more parts). [3]

Fractals have certain features and properties. The first property is self-similarity. It means that the parts are in some way related to the whole. This property makes the fractal scale-invariant. Fractal dependencies have a direct form on the graphs, where both axes have a logarithmic scale. The models described in this way should use the power law (real number raised to the power). This feature of power law scaling is the second property of fractals, a fractal dimension that can describe either a physical structure, such as a light or a time series.

Fractal properties help us to distinguish and predict the features of the surrounding reality. Today, the fractal analysis method helps physicians in analyzing the fractal dimensions of complex signals, such as encephalograms or heart sounds, to help diagnose severe diseases at the initial stage. This helps to cure the patient before the disease becomes incurable. Also, analysts comparing the behavior of prices, can foresee future development, not allowing a gross error in forecasting.

The main characteristic of fractal objects is the fractal dimension D, which is the main parameter reflecting their properties. This is a coefficient that describes a fractal structure based on its complexity.

The fractal coefficient can take either an integer or an integer value; this is because the fractal set fills the space differently than the usual geometric set does.

For example, a curve with a fractal dimension very close to 1, say 1.10, behaves quite like an ordinary line, but a curve with fractal dimension 1.9 is wound in space, almost like a surface. Similarly, a surface with a fractal dimension of 2.1 behaves. It fills the space almost like an ordinary surface, but the surface with a fractal dimension of 2.9 folds and tends to fill the space almost like a volume. [28]

The concept of fractal dimension is a classic kind of complex structure. In practice, fractal dimensions can be determined using methods using an approximate scale. As a definition of fractal dimension in the book of S.V Bozhokin and Parshina D.A "Fractals and Multifractals" uses the following formula:

$$D = -\lim_{\varepsilon \to 0} \frac{\ln{(N_\varepsilon)}}{\ln{(\varepsilon)}} \tag{2.15}$$

Where N_ε is the minimal number of n-dimensional "balls" of radius ε required to cover the set. [8] The accuracy of the predicted values obtained by this method depends on the dimension of the fractal.

Based on the theory of fractal analysis for time series, we construct a model of fractal analysis in the RStudio environment (Listing 2.5).

The result of the forecast by the method of fractal analysis, as well as the initial data for checking the forecasting are presented in Figure 2.7.

When constructing methods for fractal analysis in the RStudio program, it is possible to apply several methods for analysis. We choose the method giving the best value of the risk assessment.

As a result, the original data was trained on four models: regression, support vector method, autoregression, wavelet analysis and fractal analysis. Based on the trained model, a forecast is made for 39 points. The obtained predicted values are presented on the charts together with the initial training data and data for the prediction check. A visual assessment of the accuracy of the forecast was made.

Listing 2.5 4 The code of the program for constructing the fractal analysis method for the Mustang in the RStudio program

```
endingIndex <- 298
method <- "variogram"
random_sample_count <- 39
Sm2 <- as.data.frame(smt, row.names = NULL)  #train
#do 39 predictions of next values in Sm2
for(i in 1:39){
    delta <- c()
    # calculate delta
  for(j in 2:length(smt)){
      delta <- rbind(delta, (smt[j]-smt[j-1])) }
  # calculate standard deviation of delta
  Std_delta <- apply(delta, 2, sd)
  #update fractal dimension used as reference
  V_Reference <- fd.estimate(smt, method=method, trim=TRUE)$fd
  Sm_guesses        <-        rnorm(random_sample_count        ,
mean=smt[length(smt)], sd =Std_delta)
  minDifference = 1000000
  for(j in 1:length(Sm_guesses)){
    new_Sm <- rbind(Sm2, Sm_guesses[j])
    new_V_Reference  <-  fd.estimate(new_Sm$smt,  method=method,
trim=TRUE)$fd
        if  (abs(new_V_Reference  -  V_Reference)  <  minDifference
){
      Sm_prediction <- Sm_guesses[j]
      minDifference = abs(new_V_Reference - V_Reference)}}
  print(i)
  #add prediction to Sm2
  Sm2 <- rbind(Sm2, Sm_prediction)}
plot(Sm2)
plot(Sm2$smt,   type="l",   xlab="Value   Index",   ylab="Adjusted
Close", main="SPY Rodogram")
lines(as.data.frame(sm[1:(endingIndex+39)],  row.names  =  NULL),
col="blue")
lines(287:325, set1, col="green")
```

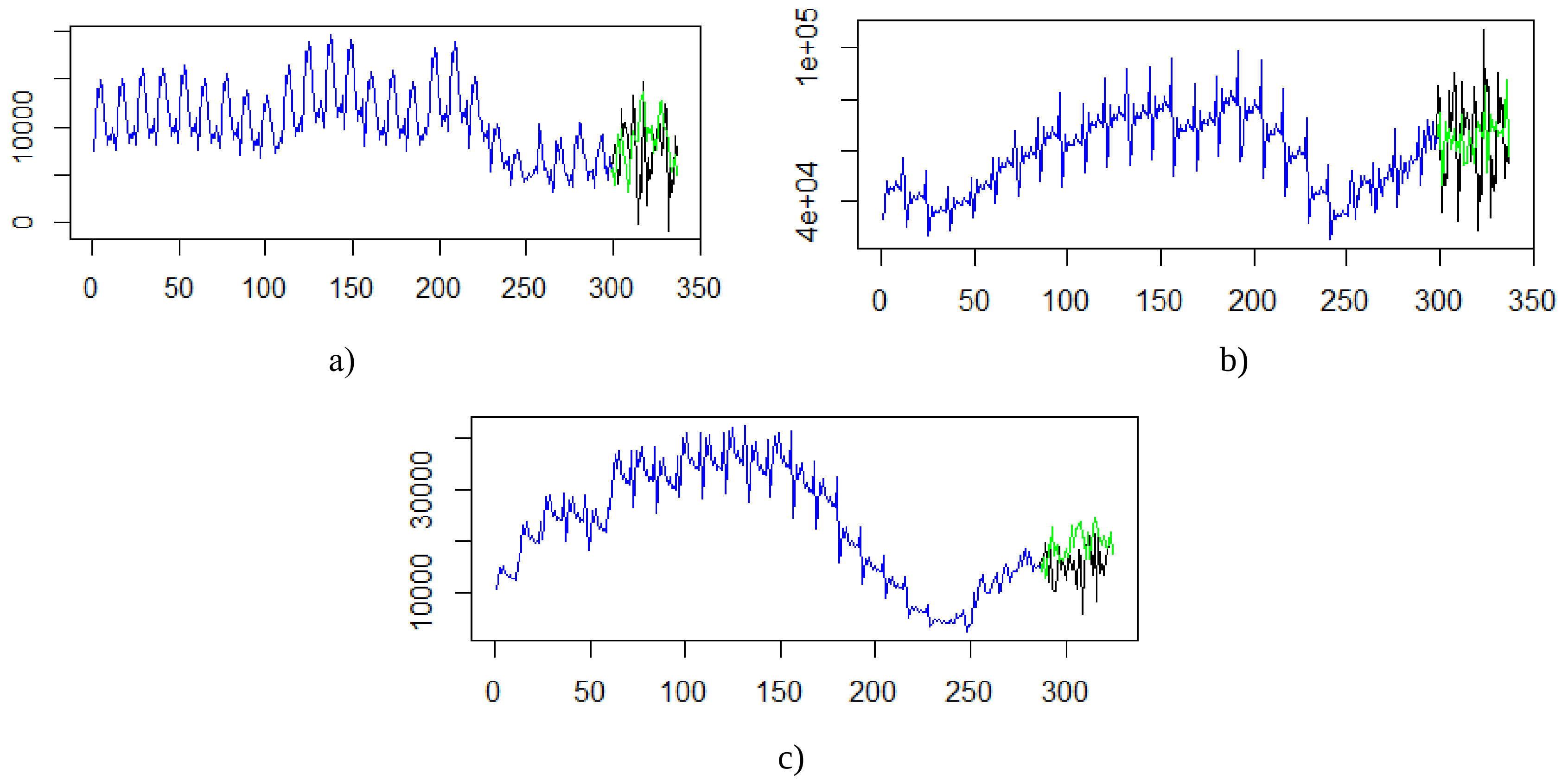


Fig. 2.7. Forecasting the volume of sales by the fractal analysis method - the initial test data (blue), the data for the prediction verification (green) and the data obtained by the fractal analysis (black): A) Ford Mustang car, b) Ford F-Series car, c) Ford Explorer car

Chapter 3. Evaluation of the reliability of the received forecasts

3.1. Assessment of the risks of applying forecasts for making managerial decisions

Risk assessment is the process of determining the probability of occurrence of a risk, i.e. events or situations capable of adversely affecting the development of the system. In a narrower sense, risk assessment is the quantitative or qualitative determination of the magnitude of the risk.

Using forecasting data, the risk's magnitude of the results obtained will increase when the results are deleted in time.

The risk assessment is calculated taking into account a certain number of factors influencing the risk (3.1) [29]:

$$r = 1 - \frac{a}{a^*}, \tag{3.1}$$

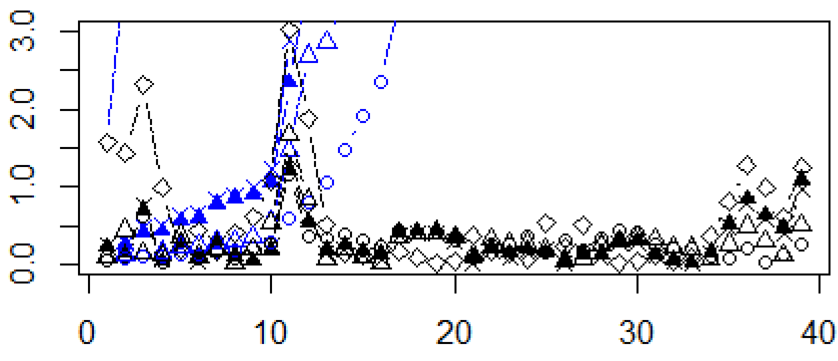
To calculate the risk assessment and determine the planning horizon, the existing set of retrospective data will be divided into two sections. The first data set (training sample) will be used to train the forecast model (determine the parameters of the forecasting method), and the second (test sample) to check the accuracy of the model for risk assessment. Then the value of the risk assessment can be calculated by the cumulative total (Figure 3.1 and Table 3.1). To do this, we will use the expression $P = 1 - r$, where P is the probability and magnitude of the risk assessment given by the table, which for assessing the overall risk should be considered as a chain of interrelated events.

Considering the fact that the probability for the independent events is c_1 and c_2 under the condition that c_2 occurs after c_1. The considered values are independent, because we get them from the predicted part and do not use them in the training data set. $P(c_1{+}c_2) = P(c_1) + P(c_2) - P(c_1)P(c_2)$ we get $r(c_1 + c_2) = 1 - P(c_1 + c_2) = 1 - 1 + r(c_1) - 1 + r(c_2) + 1 - r(c_1) - r(c_2) + r(c_1)r(c_2) = r(c_1)r(c_2)$. To calculate the subsequent values, we use the formula $r(c_{i-2} + c_{i-1} + c_i) = r(c_{i-2} + c_{i-1}) + r(c_{i-1})r(c_i)$. Obtained values show that starting from a certain moment, the value of the risk assessment begins to increase the sharply, which allows us to choose a method for forecasting and the planning horizon. The choice will be based on the choice of the method with the best planning horizon. For the calculated forecast results best results based on the risk assessment , for the Ford Mustang car models is autoregression and SVM, for F-Series is autoregressive and wavelet analysis methods, for Explorer is autoregressive and regression methods.

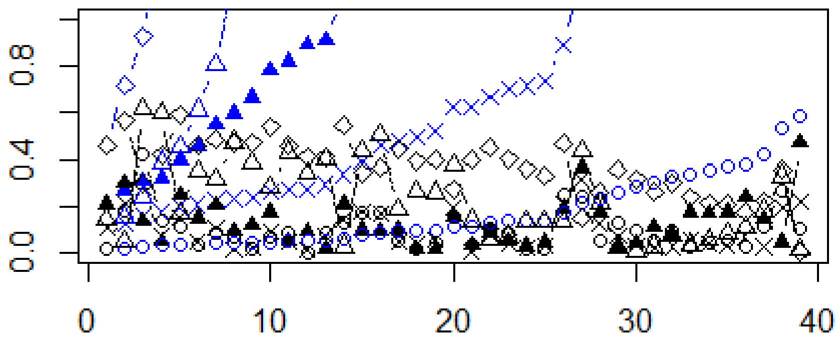
Table 3.1.

An example of the values of the cumulative value of the risk assessment for a Ford Explorer car.

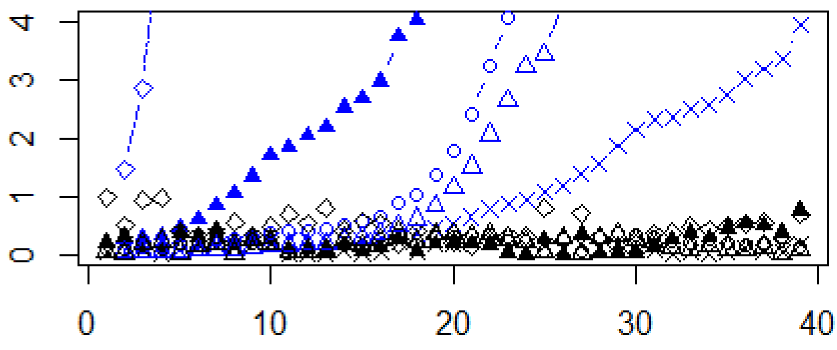
Regression	**Autoregression**	**Wavlet analysis**	**SVM**	**Fractal analysis**
0.226705351	0.073774554	**1.565765958**	0.172078643	0.028302
0.299352744	0.081265481	3.80131923	0.221558541	0.030991
0.319640138	0.096799634	12.60947032	0.231096739	0.041999
0.354823755	0.106097902	25.04742219	0.254775734	0.047554
0.474047645	0.128584038	29.7246083	0.341784982	0.047783
0.607404479	0.151047464	42.53763036	0.443574415	0.051717
0.846435461	0.200271118	49.60492369	0.628044399	0.05631
0.99983937	0.223949516	69.95623009	0.76701035	0.066946
1.23218959	0.268221414	112.8889433	0.980487266	0.07976
1.510034166	0.325520482	233.8868897	**1.254996018**	0.08679
1.543559308	0.336800478	940.6309626	1.380283654	0.107599
1.602461284	0.36227517	2701.765777	1.549989297	0.128541
1.611942475	0.387347823	4063.764445	1.706528997	0.163016
1.765946975	0.45481378	4671.109333	2.027722401	0.163505
1.77373331	0.502543694	5188.521849	2.243151895	0.222566
1.878033431	0.593393124	6298.560108	2.619525362	0.293144
2.296344235	0.794378715	7290.257192	3.450159974	0.318288
2.359977517	0.945963597	7879.571457	3.968584227	0.374516
2.739447366	**1.245218462**	8040.515591	5.042090965	0.428973
3.157301247	1.648392207	8423.482828	6.386118285	0.55742
3.681721398	2.221769445	11487.58146	8.181849074	**0.582459**



a)



b)



c)

Fig. 3.1. The instantaneous value of the risk assessment using forecasts and cumulative values, where the cross is the support vector method, the black triangle regression, the white circle - autoregression, the rhombic wavelet analysis, the white triangle - the fractal analysis: a) Ford Mustang car, b) Ford car F-Series, c) Ford Explorer car.

3.2. Comparison of the results obtained with the help of risk assessment with estimates of errors

Accuracy is the most important characteristic of the forecast. We calculate the accuracy of the obtained prediction by several methods.

1. Absolute forecast error, which is defined as the difference between the actual and theoretical levels of a series of dynamics (3.2):

$$\Delta F = y_t - \widehat{y_t} \tag{3.2}$$

Where y_t - actual values of the level of the series; $\widehat{y_t}$ are the forecast values.

2. The relative error of the forecast is defined as the ratio of the absolute error of the forecast (3.3):

$$d^P_{\text{отн}} = \frac{\Delta^P}{y_t} = \frac{|y_t - \widehat{y_t}|}{y_t} \cdot 100 \tag{3.3}$$

Absolute and relative forecast errors are used to check the accuracy of the forecast of one-dimensional series, which depend only on time. This reduces their importance, since usually a number of factors affect the system. [30]

Comparative accuracy indicators of the forecast include the average index of forecast accuracy and the standard error of the forecast.

The average index of forecast accuracy is calculated as the arithmetic mean of simple absolute errors and shows a generalized estimate of the degree of deviation of the actual and forecast values (3.4):

$$\overline{\Delta}F = \frac{\sum_{t=i}^{n} \Delta p}{n} = \frac{\sum_{t=i}^{n} |y_t - \widehat{y_t}|}{n} \tag{3.4}$$

where - n is the length of the time series.

The mean square error of the forecast is determined by the formula (3.5)

$$\delta_{\text{отн}} = \sqrt{\frac{\sum_{t=i}^{n} (y_t - \widehat{y_t})^2}{n}} \tag{3.5}$$

In practice, to determine the accuracy of the forecast, determine the mean error of approximation by the formula [32]:

$$\overline{\varepsilon}^F = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{y_t - \widehat{y_t}}{y_t} \right| \cdot 100 \tag{3.6}$$

For each method and each value, calculate the mean square error and the mean error of approximation. [31]

Table 3.1.

The results of estimating the accuracy of the predictions received for the Mustang:

	Regression	SVM	Autoregression	Wavelet analysis	Fractal analysis
Mean- root-square error	2792.051	**2586.582**	***2780.096***	4444.728	2902.137
Mean error of approximation, %	30.05322	***28.54587***	**26.8457**	59.44638	28.5975
Risk assessment (number of iterations, up to a sharp increase in the values of the integral risk assessment)	***11***	***11***	**15**	2	10

Table 3.2.

The results of estimating the accuracy of the predictions received for the F-series:

	Regression	SVM	Autoregression	Wavelet analysis	Fractal analysis
Mean- root-square error	10543.86	**7769.33**	***8167.399***	9522.45	12945.55
Mean error of approximation, %	13.04173	**9.30563**	***9.527343***	12.13911	15.67242
Risk assessment (number of iterations, up to a sharp increase in the values of the integral risk assessment)	***27***	26	**36**	5	8

Table 3.3.

The results of estimating the accuracy of the predictions received for the Explorer:

	Regression	SVM	Autoregression	Wavelet analysis	Fractal analysis
Mean- root-square error	5355.316	**2371.569**	4883.227	17161.69	***3131.141***
Mean error of approximation, %	22.35523	**10.77976**	20.56545	90.50585	***13.18923***
Risk assessment (number of iterations, up to a sharp increase in the values of the integral risk assessment)	8	**26**	16	3	***18***

In this chapter, the methods for assessing risks have been described, the values of the accumulated risk assessment have been calculated, and the risk assessment charts for each model have been constructed. Also, the Root-mean-square error and the average error of approximation of each method of all vehicle models were calculated. These methods are usually used to assess the accuracy of forecasts.

As a result of comparing the best methods obtained by estimating the forecast and using the risk assessment, different absolute best values for each method were obtained. However, the accumulated value of the risk assessment coincided with the 1 or 2 best method for the considered model of the car. This result allows, with a small margin of error in the future, to use the risk assessment method for estimating the prediction of the model.

Conclusion

In the course of the study, issues related to the features of predicting the parameters of production and economic systems were considered. To this end, the methods of working with incomplete data and methods of preparing data for analysis, as well as the features of restoring data taking into account seasonal coefficients, as well as extrapolation and interpolation of data were considered.

In the second part of the work, the regression analysis method, support vector method, autoregression, wavelet analysis and fractal analysis was considered. At training values derived from the original reconstructed data, a model is trained and predicted values for each method are constructed.

For the obtained forecast values, the method of selecting a method with the time factor based on the planning horizon using an integrated assessment of the accumulation of the risk assessment value is considered, using the example of using methods such as: regression analysis, autoregression, support vector method, wavelet analysis.[32]

Also, for these forecast values, the mean-root-square error and the mean error of approximation are calculated. The best methods are compared on the basis of calculating the accuracy of forecasting and assessing the risk of compiled forecasts. The result of the comparison with a small margin of error makes it possible to use the risk assessment in the future to estimate the prediction of the model.

List of references

1. Hair Jr J.F. A primer on partial least squares structural equation modeling (PLS-SEM) / J.F. Hair Jr, G.T.M. Hult, C. Ringle, M. Sarstedt, Sage Publications, 2016.

2. Kulinich AA Computer systems for modeling cognitive maps: approaches and methods // Problems of management. 2010. № 3.

3. Mylovnikov L.A. Support of decision-making in the management of innovative projects // Perm: Izd-vo Perm. State. Tech. University. 2011.

4. Fayzrakhmanov RA. Methodology for determining and refining the values of coefficients of cognitive maps on the example of analyzing the relationship between the volume of funding research and patent activity // Economic Analysis: Theory and Practice. 2013. No. 30 (333).

5. Afanasyev VN, Yuzbashev M.M. Time series analysis and forecasting // Moscow: Finance and statistics. 2001. (228). C. 2.

6. Jose C. Local deep kernel learning for efficient non-linear svm prediction 2013. 486–494 c.

7. Turian J., Ratinov L., Bengio Y. Word representations: a simple and general method for semi-supervised learning Association for Computational Linguistics, 2010. 384–394 c.

8. Amberg M, Mylnikov L Innovation project lifecycle prolongation method // Innovation and knowledge management in twin track economies: challenges & solutions. 2009. № 1–3. C. 491–495.

9. Alkirdou R.Kh., Mylnikov L.A. Approach to forecasting the prospects for the development of innovative projects on the basis of the innovation curve // VI All-Russian school-seminar of young scientists "Management of large systems", a collection of works. 2009. № Izhevsk. C. 38-47.

10. Alkirdou R.Kh., Mylnikov LA Forecasting the prospects for the development of the parameters of innovative projects described by the S-shaped

curve // VII All-Russian school-seminar of young scientists "Management of large systems": Proceedings. 2010. (1). C. 118-123.

11. Mylnikov LA, Alkdirou R.H. Approach to forecasting the development and management of the life cycle of investment projects // Management of large systems. 2009. № 27. C. 293-307.

12. Box G.E.P. Time Series Analysis: Forecasting and Control / G.E.P. Box, G.M. Jenkins, G.C. Reinsel, G.M. Ljung, John Wiley & Sons, 2015. 707 c.

13. Sadovnikova NA, Shmoilova RA Time series analysis and forecasting, .2016

14. Mylnikov LA, Alkirdou R.Kh. Approach to forecasting the development and management of the life cycle of investment projects / / Management of large systems. 2009. № 27. C. 293-307.

15. Amberg M., Mylnikov, L. Innovation project lifecycle prolongation method // Innovation and knowledge management in twin track economies: challenges & solutions. 2009. № 1–3. C. 491–495.

16. Sadiakhmatov M., Mylnikov L. Specificity of Undivided Time Series Forecasting Described with Innovation Curves // Proceedings of International Conference on Applied Innovation in IT. 2017. № 5 (1).

17. Mikhail Sadiakhmatov, Leonid Mylnikov Specificity of Undivided Time Series Forecasting Described with Innovation // Curves Proc. of the 5th International Conference on Applied Innovations in IT, (ICAIIT). 2017. C. 43–49.

18. Forecasting methods: interpolation, extrapolation // lektsii.org [Electronic resource]. URL: /3-50385.html (circulation date: April 20, 2017).

19. Xiaojin Zhu Semi-Supervised Learning Literature Survey // Computer Sciences University of Wisconsin, Madison. 2008.

20. Support vector machine // Wikipedia. [2017]. URL: http://en.wikipedia.org/?oldid=84406130 (date of circulation: 21/03/2017).

21. Autoregression (AR, autoregression) // Forecast NOW! [Electronic resource]. URL: https://fnow.ru/articles/avtoregressia (date of application: April 20, 2017).

22. Yakovleva A.V. Econometrics // M .: Eksmo. 2008.

23. The difference between prediction intervals and confidence intervals // Hyndsight. 2013.

24. R., Enoxon L. Applied analysis of time series / R. Otnes, L. Enoxon, Moscow: Mir, 1982. 428 p.

25. Miftakhova M.E., Panasyuk M.V. Wavelet analysis of the dynamics of the regional system // Uchenye zapiski Kazanskogo Universiteta. Series of Natural Sciences. 2009. № 1 (151).

26. Alexeev V.I Analysis and prediction of cyclic time series using wavelets and neural network fuzzy inference rules // Bulletin of Yugorsk State University. 2013. No. 3 (30).

27. Angi Rosch, Harald Schmidbauer. WaveletComp: A guided tour through the R-package // Electronic resource]. URL:http://www.hs-stat.com/projects/WaveletComp/WaveletComp_guided_tour.pdf (date of application: April 20, 2017).

28. Xiaojin Zhu Semi-Supervised Learning Literature Survey // Computer Sciences University of Wisconsin, Madison. 2008.

29. Mylnikov Leonid Risk Evaluation in Manufacturing Organization Tasks for Product Technological Projects and Establishment of Project Portfolio for Production Systems // Beijing, China. 2016. C. 339–402.

30. Taylor D. Introduction to the theory of errors / D. Taylor, Moscow: Mir, 1985. 272 с.

31. Seledkova A, Mylnikov L. Methods of forecasting and evaluation of results using the risk assessment // Information technologies.[In the press]

32. Seledkova AB, Mylnikov LA, Krause B. Peculiarities of time series forecasting for support of management decision-making in production and economic systems Federal State Autonomous Educational Institution of Higher Education Saint-Petersburg State Electrotechnical University LeTI them. VI Ulyanov (Lenin), 2017.[In the press]

Appendix A. The code of all functions used in program Rstudio.

```
d<-read.table("C:\\Users\\User\\YandexDisk\\Project\\data_month_for
_r5.csv", sep=";",na.strings="NA", header=TRUE)
as.matrix(d) #download data from csv format

library(xts)
f<-f.plot_input(d)
f.plot_m_base<- function(d)
{  #mustang
  for (i in 1:length(d[,1])) {
    d[i,14]<- as.numeric(as.Date(d[i,1], format = '%d.%m.%Y'))}
  date<- as.Date (d[,1], "%d.%m.%Y")

  plot(date,d[,3], type = "l", col='blue', ylim=range(0,40000), ylab =
"prise average Mustang")
  lines(d[,14],d[,2] )
  points(d[61,14], d[61,3],  col='red')
  points(d[193,14], d[193,3], col='red')
  points(d[313,14], d[313,3], col='red')
  points(d[61,14], d[61,2],  col='red')
  points(d[193,14], d[193,2], col='red')
  points(d[313,14], d[313,2], col='red')
}
f.plot_m_sales<- function(d)
{
  #sales
  plot(date, d[,2], type="l", ylab = "sales value Mustang")
  points(d[61,14], d[61,2],  col='red')
  points(d[193,14], d[193,2], col='red')
  points(d[313,14], d[313,2], col='red')
}
f.plot_fs_base<- function(d)  #f-series
{
  plot(date,d[,7],  type="l",  col='blue',  ylab  =  "prise  average  and
sales  F-Series",
       ylim = range(0,90000))
  points(d[37,14], d[37,7],  col='red')
  points(d[97,14], d[97,7], col='red')
  points(d[181,14], d[181,7], col='red')
  points(d[241,14], d[241,7], col='red')
  points(d[313,14], d[313,7], col='red')
}
f.plot_fs_sales<- function(d)  #sales f-150
{
  lines(date, d[,6])
  points(d[37,14], d[37,6],  col='red')
  points(d[97,14], d[97,6], col='red')
  points(d[181,14], d[181,6], col='red')
  points(d[241,14], d[241,6], col='red')
  points(d[313,14], d[313,6], col='red')
}
f.plot_e_base<- function(d)  #explorer
{
  plot(date,d[,11],  col='blue',  type="l",  ylab  =  "prise  average  and
sales Explorer", ylim=range(0,45000))
  points(d[61,14], d[61,11],  col='red')
  points(d[157,14], d[157,11],  col='red')
  points(d[205,14], d[205,11],  col='red')
  points(d[253,14], d[253,11],  col='red')
}
f.plot_e_sales<- function(d)
{
  lines(date, d[,10], type="l", ylab = "sales explorer")
  points(d[61,14], d[61,10],  col='red')
  points(d[157,14], d[157,10],  col='red')
  points(d[205,14], d[205,10],  col='red')
  points(d[253,14], d[253,10],  col='red')
}
#--------------linear regression
sm<- d[,2]
pr_av_m<-d[,3]
sf<- d[,6]
pr_av_f<-d[,7]
se<- d[,10]
pr_av_e<-d[,11]

f.lregr_m<-  function(test,train,d,sm,  pr_av_m,  sigm,  error)   #linear
mustang
{
  plot(pr_av_m,sm)
  points(test[,3],test[,2], col= 'green') # test data points
  carlm <-lm(formula = sm ~ pr_av_m)
  lpr_m<- predict(carlm)
  summary(carlm)
  abline(carlm, col='red')

    # #predict
    plot(train[,3],train[,2], col = 'blue')
    lm_m<- lm(formula = train[,2] ~ train[,3])
    #for sales only
    x<- 1:337

    e<-data.frame(x=x,y=sm)
    lm_m_s<-   nls(y   ~   k1+k2*x+k3*x*x+k4*x*x*x+k5*x*x*x*x,   data=e,
start=list(k1=0.1, k2=0.1, k3=0.1, k4=0.1, k5=0.1))
    lpr_tr_m_s<- predict(lm_m_s)
    coef_func <- coef(lm_m_s)

    #lpr_tr_m<- predict(lm_m)
    plot(date,sm, type = "l", ylab = "Sales Mustang")
    points(date_t,test[,2], col= 'green', type="l") # test data points
    lines(date, lpr_tr_m_s, col='red')
    #error
    error <- smt1 - lpr_tr_m_s[299:337]          # error linear predict
mustang
    #  predictionRMSE  <-  rmse(error)        #  3104.2510    Средняя
квадратическая ошибка прогноза root-mean-square deviation for mustang
linear regression
    dost_m_av<- sigm(error)
    print(dost_m_av)
    #средняя ошибка аппроксимации
    croshappr<- sum(abs(error/smt1))/length(smt1)*100
    print(croshappr)
    #risk
    x1<-1:39
    e_2<-(1-lpr_tr_m_s[x1]/smt1[x1])
      e2<-matrix()
      e2[1]<-1.45
    for (i in 2:39) {
      e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])

    }
    plot(x1, abs(e2), col = "red", type = "l", ylim=range(0,10))
    lines(x1,abs(e_2), col = "black", type = "l",lty=4)
  }
f.lregr_f<- function(test,train,d,sf, pr_av_f, sigm, error)    #linear
f-150
{  plot(pr_av_f,sf)
  points(test[,7],test[,6], col= 'green')
  carlf <-lm(formula = sf ~ pr_av_f)
  lpr_f<- predict(carlf)
  summary(carlf)
  abline(carlf, col='red')
  error <- sf - lpr_f         # error linear predict mustang
  predictionRMSE  <-  rmse(error)        #  14046.504  root-mean-square
deviation for mustang linear regression
  dost_f_av<- sigm(error)
  print(dost_f_av)
  #for sales only
  x<- 1:337
  e<-data.frame(x=x,y=sf)
  lm_f_s<-   nls(y   ~   k1+k2*x+k3*x*x+k4*x*x*x+k5*x*x*x*x,   data=e,
start=list(k1=0.1, k2=0.1, k3=0.1, k4=0.1, k5=0.1))
  lpr_tr_f_s<- predict(lm_f_s)
  coef_func <- coef(lm_f_s)

  lpr_tr_f<- predict(lm_f_s)
  plot(date,sf, type = "l", ylab = "Sales F-series")
  points(date_t,test[,6], col= 'green', type="l") # test data points
  lines(date, lpr_tr_f_s, col='red')
  #error
  error <- sft1 - lpr_tr_f_s[299:337]          # error linear predict
mustang
  #  predictionRMSE  <-  rmse(error)          #  3104.2510    Средняя
квадратическая ошибка прогноза root-mean-square deviation for mustang
linear regression
  dost_m_av<- sigm(error)
  print(dost_m_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/sft1))/length(sft1)*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-lpr_tr_f_s[x1]/sft1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
}
f.lregr_e<- function(test,train,d,se, pr_av_e, sigm, error)    #linear
explorer
```

```
{
  plot(pr_av_e,se)
  points(test[,11],test[,10], col= 'green')
  carle <-lm(formula = se ~ pr_av_e)
  lpr_e<- predict(carle)
  summary(carle)
  abline(carle, col='red')
  for (i in 1:325)
    se_n[i]<- d[i+12,10]
  date_n<- matrix()
  for (i in 1:325)
    date_n[i]<- date[i+12]
  error <- se_n - lpr_e         # error linear predict mustang
  predictionRMSE  <-  rmse(error)        #  10082.29  root-mean-square
deviation for mustang linear regression
  dost_e_av<- sigm(error)
  print(dost_e_av)
  #for sales only
  x<- 1:337
  e<-data.frame(x=x,y=se)
  lm_e_s<-   nls(y   ~   k1+k2*x+k3*x*x+k4*x*x*x+k5*x*x*x*x,   data=e,
start=list(k1=0.1, k2=0.1, k3=0.1, k4=0.1, k5=0.1))
  lpr_tr_e_s<- predict(lm_e_s)
  coef_func <- coef(lm_e_s)

  lpr_tr_e<- predict(lm_e_s)
  plot(date,se, type = "l", ylab = "Sales Explorer")
  points(date_t,test[,10], col= 'green', type="l") # test data points
  lines(date_n, lpr_tr_e_s, col='red')
  #risk
  x1<-1:39
  e_2<-(1-lpr_tr_e_s[x1]/set1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
  #error
  error <- set1 - lpr_tr_e_s[287:325]          # error linear predict
mustang
  #  predictionRMSE  <-  rmse(error)         #  3104.2510    Средняя
квадратическая ошибка прогноза root-mean-square deviation for mustang
linear regression
  dost_m_av<- sigm(error)
  print(dost_m_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/set1))/length(set1)*100
  print(croshappr)
}
#support vector machine
  library(e1071)
f.svm_m_av<-  function(test,train,d,sm,  pr_av_m,  sigm,  error  )  #svm
mustang sales ans average prise
{
  model<- svm(sm ~ pr_av_m,d)
  prY1 <- predict(model,d)
  plot(pr_av_m,sm, type = "l")
  points(test[,3],test[,2],  col=  'green',  type  =  "l")  #  test  data
points
  points(pr_av_m, prY1, col = "red", type ="l")
  summary(model)
  #deviation
  error <- smt1 - prY1[299:337]         # error linear predict mustang
  predictionRMSE  <-  rmse(error)        #  2774.115  root-mean-square
deviation for mustang linear regression
  dost_m_av<- sigm(error)
  print(dost_m_av)
  #from time sales only
  for (i in 1:length(test)){
    test[i,14]<- as.numeric(as.Date(test[i,1], format = '%d.%m.%Y'))}
  date_t<- as.Date (test[,1], "%d.%m.%Y")
    model1<- svm(sm ~ date, d)
  prY2_m <- predict(model1,d)
  plot(date,sm, type = "l")
  points(date_t,test[,2], col= 'green', type = "l") # test data points
  points(date, prY2_m, col = "red", type ="l")
  #deviation
  error  <-  smt1  -  prY2_m[299:337]            #  error  linear  predict
mustang
  predictionRMSE  <-  rmse(error)        #  2790.9426  root-mean-square
deviation for mustang linear regression
  dost_m_av<- sigm(error)
  print(dost_m_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/smt1))/length(smt1)*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-prY2_m[x1]/smt1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)

}
f.svm_f_av<-  function(test,train,d,sf,  pr_av_f,  sigm,  error  )  #svm  f-
series sales ans average prise
{
  model<- svm(sf ~ pr_av_f,d)
  prY2 <- predict(model,d)
  plot(pr_av_f,sf)
  points(test[,7],test[,6], col= 'green') # test data points
  points(pr_av_f, prY2, col = "red")
  summary(model)
  #deviation
  error <- sf - prY2         # error linear predict mustang
  predictionRMSE  <-  rmse(error)        #  10422.37  root-mean-square
deviation for mustang linear regression
  dost_f_av<- sigm(error)
  print(dost_f_av)
  #from time sales only
  for (i in 1:length(test)){
    test[i,14]<- as.numeric(as.Date(test[i,1], format = '%d.%m.%Y'))}
  date_t<- as.Date (test[,1], "%d.%m.%Y")
  model1<- svm(sf ~ date, d)
  prY2_f <- predict(model1,d)
  plot(date,sf, type = "l")
  points(date_t,test[,6], col= 'green', type = "l") # test data points
  points(date, prY2_f, col = "red", type ="l")
  #deviation
  error <- sft1 - prY2_f[299:337]             # error linear predict
mustang
  predictionRMSE  <-  rmse(error)        #  8786.4529  root-mean-square
deviation for mustang linear regression
  dost_m_av<- sigm(error)
  print(dost_m_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/sft1))/length(sft1)*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-prY2_f[x1]/sft1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
}
f.svm_e_av<-  function(test,train,d,se,  pr_av_e,  sigm,  error,  svme,
se_n ) #svm f-series sales ans average prise
{
  model<- svm(se ~ pr_av_e,d)
  prY3 <- predict(model,d)
  plot(pr_av_e,se)
  points(test[,11],test[,10], col= 'green') # test data points
  se_n<- matrix()
  for (i in 1:325){
    se_n[i]<- d[i+12,11]}
  points(se_n, prY3, col = "red")
  summary(model)
  #deviation
  error <- se_n - prY3             # error linear predict mustang
  predictionRMSE  <-  rmse(error)        #  12780.74  root-mean-square
deviation for mustang linear regression
  dost_e_av<- sigm(error)
  print(dost_e_av)
  #from time sales only
  for (i in 1:length(test)){
    test[i,14]<- as.numeric(as.Date(test[i,1], format = '%d.%m.%Y'))}
  date_t<- as.Date (test[,1], "%d.%m.%Y")
  model1<- svm(se ~ date, d)
  prY2_e <- predict(model1,d)
  prY2_n<- matrix()
  for (i in 1:325){
    prY2_n[i]<- prY2_e[i+12]    }
  plot(date,se, type = "l")
  points(date_t,test[,10],  col=  'green',  type  =  "l")  #  test  data
points
  points(date_n, prY2_n, col = "red", type ="l")
  #deviation
  error <- se_n[287:325] - prY2_n [287:325]          # error linear
predict mustang
  predictionRMSE  <-  rmse(error)       #  14696.2404  root-mean-square
deviation for mustang linear regression
  dost_m_av<- sigm(error)
  print(dost_m_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/se_n[287:325]))/length(se_n[287:325])*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-prY2_e[x1]/set1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
  #predict
  model<- svm(train[,10] ~ train[,11],train)
  model_e<- as.ts(train[,11], model )
  prY_e <- forecast(model_e,39)

  prY_e <- predict(model,newdata = test[,11])
  plot(train[,11], train[,10])
  points(test[,11],prY_e, col='red')
  plot(prY_e)
}
# root-mean-square deviation
rmse <- function(error)
  {
    sqrt(mean(error^2))
  }                               #function rmse
sigm<- function(error)            # Достоверность
{
  l<- length(error)
  sqrt(sum((error^2))/l)
}
#_______здесь были расчеты------
#-----------autoregration----------------
train<-
read.table("C:\\Users\\User\\YandexDisk\\Project\\data_month_training.
csv", sep=";",na.strings="NA", header=TRUE)
as.matrix(train)
train2<-
read.table("C:\\Users\\User\\YandexDisk\\Project\\data_month_training2
.csv", sep=";",na.strings="NA", header=TRUE)
as.matrix(train2)
train3<-
read.table("C:\\Users\\User\\YandexDisk\\Project\\data_month_training3
.csv", sep=";",na.strings="NA", header=TRUE)
as.matrix(train3)
test<-
read.table("C:\\Users\\User\\YandexDisk\\Project\\data_month_test.csv"
, sep=";",na.strings="NA", header=TRUE)
as.matrix(test)
library('e1071')
library(forecast)
```

```
# trainig data
date1<- as.Date (train[,1], "%d.%m.%Y")
smt<- train[,2]
pmt<- train [,3]
sft<- train [,6]
set<- train [,10]
#test data
date2<- as.Date (test[,1], "%d.%m.%Y")
smt1<- test[,2]
pmt1<-test[,3]
sft1<- test [,6]
set1<- test [,10]
f.autoregr_m_sales<-   function(smt)   #autoregression   function   for
mustang sales
{
  model<- auto.arima(smt)
  future_ar_m<- forecast(model, h=39)
  plot(future_ar_m)
  #plot for article
  plot(date, sm, type = "l")
  points(date_t,test[,2], col= 'green', type = "l") # test data points
  points(date_t, future_ar_m$mean, type="l", col="red")
  #error
  m.accur<- accuracy(future_ar_m, smt1)
  tr<-  residuals.Arima(future_ar_m)       #Returns  time  series  of
residuals from a fitted model.
  #error standart
 #  trainy<-  c(568.508,  5162.420,  6317.372,  6290.167,  7245.803,
7851.728, 8123.545, 8280.318, 7944.192, 7440.366,
#                  6827.474, 6252.108, 5909.257, 5839.302, 6071.335,
6531.942, 7089.062, 7601.857, 7934.976, 8009.847,
#                  7816.118, 7412.066, 6910.054, 6442.823, 6130.083,
6047.584, 6208.754, 6563.763, 7014.226, 7440.425,
#                 7732.986, 7820.756, 7687.967, 7376.410, 6972.692,
6584.330, 6311.230, 6220.119, 6328.408)
  error <- smt1 - future_ar_m$mean       # error svm predict mustang
  predictionRMSE <- rmse(error)    #2884.2807
  p_m_av<- sigm(error)
  print(p_m_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/smt1))/length(smt1)*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-future_ar_m$mean[x1]/smt[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
}
# write risk table
  must<-  data.frame(future_ar_m$mean,  prY2_m[299:337],    e_ar,  e_2<
e_wv )
  write.table(must, file = file.choose())
  fser<- data.frame(future_f_s$mean, fin_f,  e_ar, e_wv )
  write.table(fser, file = file.choose())
  expl<- data.frame(future_e_s$mean,lpr_tr_e_s[287:325] ,  e_ar, e_r )
  write.table(expl, file = file.choose())
# write risk table for all models and elements
  all<-  data.frame(future_ar_m$mean,  prY2_m[299:337],    e_ar,  e_2,
future_f_s$mean,    fin_f,        e_ar,     e_wv,    future_e_s$mean,
lpr_tr_e_s[287:325],  e_ar, e_r )
  write.table(all, file = file.choose())
f.autoregr_f_sales<-   function(sft)   #autoregression   function   for
mustang sales
{
  model_f<- auto.arima(sft)
  future_f_s<- forecast(model_f, h=39)
  plot(future_f_s)
  #plot for article
  plot(date, sf, type = "l")
  points(date_t,test[,6], col= 'green', type = "l") # test data points
  points(date_t, future2$mean, type="l", col="red")
  #error
  m.accur<- accuracy(future2, sft1)
  tr<-  residuals.Arima(future)     #Returns  time  series  of  residuals
from a fitted model.
  #error standart
#   trainy2<-  c(  66260.94,  63160.00,  64722.95,  64551.66,  64165.50,
64518.18, 64375.71, 64368.14, 64421.11, 64383.02,
#                    64394.68,  64398.35,  64391.68,  64395.58,  64394.80,
64394.10, 64394.89, 64394.52, 64394.54, 64394.65,
#                    64394.56,  64394.59,  64394.60,  64394.58,  64394.59,
64394.59, 64394.59, 64394.59, 64394.59, 64394.59,
#                    64394.59,  64394.59,  64394.59,  64394.59,  64394.59,
64394.59, 64394.59, 64394.59, 64394.59)
  error <- sft1 - future_f_s$mean     # error svm predict mustang
  predictionRMSE <- rmse(error)    #8167.399
  p_f_av<- sigm(error)
  print(p_f_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/sft1))/length(sft1)*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-future_f_s$mean[x1]/sft1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
}
f.autoregr_e_sales<-   function(set)   #autoregression   function   for
mustang sales
{
  model_e<- auto.arima(set)
  future_e_s<- forecast(model_e, h=39)
  plot(future_e_s)
  #plot for article
  plot(date, sm, type = "l")
  points(date_t,test[,2], col= 'green', type = "l") # test data points
  points(date_t, future$mean, type="l", col="red")
  #error
  m.accur<- accuracy(future3, set1)
  tr<-  residuals.Arima(future)     #Returns  time  series  of  residuals
from a fitted model.
  #error standart
#   trainy3<-  c(  15320.62,  15190.30,  15258.67,  15261.50,  15236.40,
15250.75, 15250.57, 15245.77, 15248.76, 15248.58,
#                   15247.67, 15248.29, 15248.22, 15248.05, 15248.18,
15248.16, 15248.13, 15248.15, 15248.15, 15248.14,
#                   15248.15, 15248.14, 15248.14, 15248.14, 15248.14,
15248.14, 15248.14, 15248.14, 15248.14, 15248.14,
#                   15248.14, 15248.14, 15248.14, 15248.14, 15248.14,
15248.14, 15248.14, 15248.14, 15248.14)
  trainy3<- future_e_s$mean
  error <- set1 - trainy3       # error svm predict mustang
  predictionRMSE <- rmse(error)    #5043.714
  p_e_av<- sigm(error)
  print(p_e_av)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/set1))/length(set1)*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-future_e_s$mean[x1]/set1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
}
#_________________________WAVELET
#wavelet analyses
library(WaveletComp)
for (i in 1:length(train[,1])) {
  train[i,14]<- as.numeric(as.Date(train[i,1], format = '%d.%m.%Y'))}
attach(train)
test_e1<- matrix()
test_e2<- matrix()
for (i in 1:286){
test_e1[i]<- train[i+12,10]
test_e2[i]<- train[i+12,11]
}
t1 = cbind(1:298, train[,2])
t2 = cbind(1:298, train[,3])
tf1 = cbind(1:298, train[,6])
tf2 = cbind(1:298, train[,7])
te1 = cbind(1:286, test_e1)
te2 = cbind(1:286, test_e2)
t2_1 = cbind(1:337, train3[,2])
t2_2 = cbind(1:337, train3[,3])
f.wavelet_2_m<-function(t1,t2, t2_1, t2_2) #for mustang sales
{
  my.data<- data.frame(t1=t1)  #sales vs time
  plot(my.data, type = "l")
  my.w_m_sail<- analyze.wavelet(my.data, "t1.2",
                         loess.span = 0,
                         dt=1, dj=1/298,
                         lowerPeriod = 1,
                        upperPeriod = 128,
                        make.pval = F, n.sim=10)
  wt.image(my.w, n.levels = 298,
           legend.params = list(lab="wavelet power levels"))
 # my.data<- data.frame(t2=t2)  #prise vs time
#  plot(my.data, type = "l")
#  my.w_m_pr<- analyze.wavelet(my.data, "t2.2",
 #                       loess.span = 0,
  #                      dt=1, dj=1/298,
   #                     lowerPeriod = 1,
    #                    upperPeriod = 256,
     #                  make.pval = F, n.sim=10)
#  wt.image(my.w, n.levels = 298,
 #          legend.params = list(lab="wavelet power levels"))}
library(WaveletComp)
my.rec3<- reconstruct(my.w_m_sail, sel.period = 3, show.legend = F)
x.rec3<- my.rec3$series$t1.2.r
model3<-auto.arima(x.rec3)
future3<-forecast(model3, h=39)
plot(future3)

my.rec5<- reconstruct(my.w_m_sail, sel.period = 5, show.legend = F)
x.rec5<- my.rec5$series$t1.2.r
model5<-auto.arima(x.rec5)
future5<-forecast(model5, h=39)
plot(future5)

my.rec8<- reconstruct(my.w_m_sail, sel.period = 8, show.legend = F)
x.rec8<- my.rec8$series$t1.2.r
model8<-auto.arima(x.rec8)
future8<-forecast(model8, h=39)
plot(future8)

my.rec18<- reconstruct(my.w_m_sail, sel.period = 18, show.legend = F)
x.rec18<- my.rec18$series$t1.2.r
model18<-auto.arima(x.rec18)
future18<-forecast(model18, h=39)
plot(future18)

my.rec135<-  reconstruct(my.w_m_sail,  sel.period  =  135,  show.legend  =
F)
x.rec135<- my.rec135$series$t1.2.r
model135<-auto.arima(x.rec135)
future135<-forecast(model135, h=39)
plot(future135)

my.rec150<-  reconstruct(my.w_m_sail,  sel.period  =  150,  show.legend  =
F)
x.rec150<- my.rec150$series$t1.2.r
model150<-auto.arima(x.rec150)
future150<-forecast(model150, h=39)
plot(future150)

my.rec256<-  reconstruct(my.w_m_sail,  sel.period  =  256,  show.legend  =
F)
x.rec256<- my.rec256$series$t1.2.r
model256<-auto.arima(x.rec256)
future256<-forecast(model256, h=39)
```

```
plot(future256)
fin.rec<-NA
for (i in 1:298)
  fin.rec[i]<- sum (x.rec3[i], x.rec5[i],x.rec8[i],
                x.rec135[i], x.rec18[i],
                x.rec150[i],x.rec256[i]
  )/7
fin_m<-NA
for (i in 1:39)
  fin_m[i]<- sum (future3$mean[i], future5$mean[i],future8$mean[i],
                future135$mean[i], future18$mean[i],
                future150$mean[i],future256$mean[i]
  )/7
plot(date, sm, type="l")
lines(date_t, test$Sales_value_in_USA_Must.g, col="green")
lines(date_t, fin_m, col="red")
lines(date1, fin.rec, col="red")
asd<- NA
asd[1]<- fin.rec[298]
asd[2]<- fin_m[1]
lines(date[298:299],asd, col="red" )
#error
error <- smt1 - fin_m          # error6
predictionRMSE <- rmse(error) #30196.102
fin.pr<- sigm(error)
print(fin.pr)
#средняя ошибка аппроксимации
 croshappr<- sum(abs(error/smt1))/length(smt1)*100
 print(croshappr)
 #risk
 x1<-1:39
 e_2<-(1-fin_m[x1]/smt1[x1])
 e2<-abs(e_2)
 for (i in 2:39) {
   e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
 }
 plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
 lines(x1,abs(e_2), col = "black", type = "l",lty=4)
f.wavelet_2_f<-function(tf1,tf2, t2_1, t2_2) #for f-series
{
  my.data<- data.frame(tf1=tf1)  #sales vs time
  plot(my.data, type = "l")
  my.w_f_s<- analyze.wavelet(my.data, "tf1.2",
                        loess.span = 0,
                        dt=1, dj=1/298,
                        lowerPeriod = 1,
                        upperPeriod = 128,
                        make.pval = F, n.sim=10)
  wt.image(my.w_f_s, n.levels = 298,
           legend.params = list(lab="wavelet power levels"))
 # my.data<- data.frame(tf2=tf2)  #prise vs time
#  plot(my.data, type = "l")
 # my.w<- analyze.wavelet(my.data, "tf2.2",
  #                      loess.span = 0,
   #                    dt=1, dj=1/298,
    #                   lowerPeriod = 1,
     #                 upperPeriod = 256,
      #                make.pval = F, n.sim=10)
#  wt.image(my.w, n.levels = 298,
 #          legend.params = list(lab="wavelet power levels"))
  library(WaveletComp)
  my.rec3<- reconstruct(my.w_f_s, sel.period = 3, show.legend = F)
  x.rec3<- my.rec3$series$tf1.2.r
  model3<-auto.arima(x.rec3)
  future3<-forecast(model3, h=39)
  plot(future3)

  my.rec5<- reconstruct(my.w_f_s, sel.period = 5, show.legend = F)
  x.rec5<- my.rec5$series$tf1.2.r
  model5<-auto.arima(x.rec5)
  future5<-forecast(model5, h=39)
  plot(future5)

  my.rec8<- reconstruct(my.w_f_s, sel.period = 8, show.legend = F)
  x.rec8<- my.rec8$series$tf1.2.r
  model8<-auto.arima(x.rec8)
  future8<-forecast(model8, h=39)
  plot(future8)

  my.rec135<- reconstruct(my.w_f_s, sel.period = 128, show.legend = F)
  x.rec135<- my.rec135$series$tf1.2.r
  model135<-auto.arima(x.rec135)
  future135<-forecast(model135, h=39)
  plot(future135)

  my.rec150<- reconstruct(my.w_f_s, sel.period = 150, show.legend = F)
  x.rec150<- my.rec150$series$tf1.2.r
  model150<-auto.arima(x.rec150)
  future150<-forecast(model150, h=39)
  plot(future150)

  my.rec256<- reconstruct(my.w_f_s, sel.period = 290, show.legend = F)
  x.rec256<- my.rec256$series$tf1.2.r
  model256<-auto.arima(x.rec256)
  future256<-forecast(model256, h=39)
  plot(future256)
  fin.rec<-NA
  for (i in 1:298)
    fin.rec[i]<- sum (x.rec3[i], x.rec5[i],x.rec8[i],
                      x.rec135[i],
                      x.rec150[i],x.rec256[i]
    )/6
  fin_f<-NA
  for (i in 1:39)
    fin_f[i]<- sum (future3$mean[i], future5$mean[i],future8$mean[i],
                  future135$mean[i],
                  future150$mean[i],future256$mean[i]
    )/6

  plot(date, sf, type="l")
  lines(date_t, test$Sales_value_in_USA_F.150, col="green")
  lines(date_t, fin_f, col="red")
  lines(date1, fin.rec, col="red")
  asd<- NA
  asd[1]<- fin.rec[298]
  asd[2]<- fin_f[1]
  lines(date[298:299],asd, col="red" )
  #error
  error <- sft1 - fin_f          # error6
  predictionRMSE <- rmse(error) #30196.102
  fin.pr<- sigm(error)
  print(fin.pr)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/sft1))/length(sft1)*100
  print(croshappr)
  x1<-1:39
  e_2<-(1-fin_f[x1]/sft1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
}
f.wavelet_2_e<-function(te1,te2, t2_1, t2_2) #for explorer
{
  my.data<- data.frame(te1=te1)  #sales vs time
  plot(my.data, type = "l")
  my.w_e_s<- analyze.wavelet(my.data, "te1.test_e1",
                        loess.span = 0,
                        dt=1, dj=1/298,
                        lowerPeriod = 1,
                        upperPeriod = 128,
                        make.pval = F, n.sim=10)
  wt.image(my.w_e_s, n.levels = 298,
           legend.params = list(lab="wavelet power levels"))
 # my.data<- data.frame(te2=te2)  #prise vs time
  #plot(my.data, type = "l")
  #my.w<- analyze.wavelet(my.data, "te1.test_e1",
   #                     loess.span = 0,
    #                    dt=1, dj=1/298,
     #                   lowerPeriod = 1,
      #                  upperPeriod = 256,
       #                 make.pval = F, n.sim=10)
#  wt.image(my.w, n.levels = 298,
 #          legend.params = list(lab="wavelet power levels"))
  library(WaveletComp)
  my.rec3<- reconstruct(my.w_e_s, sel.period = 2, show.legend = F)
  x.rec3<- my.rec3$series$te1.test_e1.r
  model3<-auto.arima(x.rec3)
  future3<-forecast(model3, h=39)
  plot(future3)
  lines(x.rec3, col="red")

  my.rec5<- reconstruct(my.w_e_s, sel.period = 3, show.legend = F)
  x.rec5<- my.rec5$series$te1.test_e1.r
  model5<-auto.arima(x.rec5)
  future5<-forecast(model5, h=39)
  plot(future5)
  lines(x.rec5, col="red")
  my.rec8<- reconstruct(my.w_e_s, sel.period = 8, show.legend = F)
  x.rec8<- my.rec8$series$te1.test_e1.r
  model8<-auto.arima(x.rec8)
  future8<-forecast(model8, h=39)
```

```
plot(future8)
lines(x.rec8, col="red")
my.rec28<- reconstruct(my.w_e_s, sel.period = 28, show.legend = F)
x.rec28<- my.rec28$series$te1.test_e1.r
model28<-auto.arima(x.rec28)
future28<-forecast(model28, h=39)
plot(future28)
lines(x.rec28, col="red")

my.rec135<- reconstruct(my.w_e_s, sel.period = 24, show.legend = F)
x.rec135<- my.rec135$series$te1.test_e1.r
model135<-auto.arima(x.rec135)
future135<-forecast(model135, h=39)
plot(future135)
lines(x.rec135, col="red")

my.rec150<- reconstruct(my.w_e_s, sel.period = 125, show.legend = F)
x.rec150<- my.rec150$series$te1.test_e1.r
model150<-auto.arima(x.rec150)
future150<-forecast(model150, h=39)
plot(future150)
lines(x.rec150, col="red")

my.rec256<- reconstruct(my.w_e_s, sel.period = 256, show.legend = F)
x.rec256<- my.rec256$series$te1.test_e1.r
model256<-auto.arima(x.rec256)
future256<-forecast(model256, h=39)
plot(future256)
lines(x.rec256, col="red")

fin.rec<-NA
for (i in 1:286)
  fin.rec[i]<- sum (x.rec3[i], x.rec5[i],x.rec8[i],
                    x.rec135[i],
                    x.rec28[i],
                    x.rec150[i]
                    #x.rec256[i]
  )/6
fin_e<-NA
for (i in 1:39)
  fin_e[i]<- sum (future3$mean[i], future5$mean[i],future8$mean[i],
                future135$mean[i],
                future28$mean[i],
                future150$mean[i]
               # future256$mean[i]
  )/6

plot(date, se, type="l")
lines(date_t, test$Sales_value_in_USA_Exp.r, col="green")
lines(date_t, fin_e, col="red")
date_t_n<- matrix()
for (i in 1:286)
  date_t_n[i]<- date1[i+12]
lines(date_t_n, fin.rec, col="red")
asd<- NA
asd[1]<- fin.rec[286]
asd[2]<- fin_e[1]
lines(date[298:299],asd, col="red" )
```

```
  #error
  error <- set1 - fin_e          # error6
  predictionRMSE <- rmse(error) #30196.102
  fin.pr<- sigm(error)
  print(fin.pr)
  #средняя ошибка аппроксимации
  croshappr<- sum(abs(error/set1))/length(set1)*100
  print(croshappr)
  #risk
  x1<-1:39
  e_2<-(1-fin_e[x1]/set1[x1])
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  plot(x1, abs(e2),col="red", type = "l", ylim=range(0,10))
  lines(x1,abs(e_2), col = "black", type = "l",lty=4)
}
#_____________Fractal__________________________
library(quantmod)
library(fractaldim)

endingIndex <- 298
#method <- "rodogram" # 10 ,  1.395073e+00
method <- "variogram" #12 1.088473e+00
#method <- "madogram" #8 1.250406e+00
#method <- "variation" #11, 2.087000e+00
#method <- "incr1"   # 5,  1.1668507 ступенчатое приращение
##method <- "boxcount" #2 1.114187e+00
#method <- "dctII"  #3, 1.216222e+00

Sm1 <- as.data.frame(smt1, row.names = NULL) #test
Sm2 <- as.data.frame(smt, row.names = NULL)  #train

#number of samples to draw for each guess
random_sample_count <- 39

#Mustang
for(i in 1:39){

  delta <- c()

  for(j in 2:length(smt)){
    delta <- rbind(delta, (smt[j]-smt[j-1]))
  }
  # calculate standard deviation of delta
  Std_delta <- apply(delta, 2, sd)

  #update fractal dimension used as reference
  V_Reference <- fd.estimate(smt, method=method, trim=TRUE)$fd

  Sm_guesses <- rnorm(random_sample_count , mean=smt[length(smt)], sd
=Std_delta)

  minDifference = 1000000

  # check the fractal dimension of Sm plus each different guess and
  # choose the value with the least difference with the reference
  for(j in 1:length(Sm_guesses)){
    new_Sm <- rbind(Sm2, Sm_guesses[j])
    new_V_Reference    <-    fd.estimate(new_Sm$smt,    method=method,
trim=TRUE)$fd

    if (abs(new_V_Reference - V_Reference) < minDifference ){
      Sm_prediction <- Sm_guesses[j]
      minDifference = abs(new_V_Reference - V_Reference)
    }
  }
  print(i)
  Sm2 <- rbind(Sm2, Sm_prediction)
}
plot(Sm2$smt,    type="l",    xlab="Дата",    ylab="Объем    продаж",
main="Фрактальный анализ для модели Mustang")
lines(as.data.frame(sm[1:(endingIndex+39)],    row.names    =    NULL),
col="blue")
lines(299:337, smt1, col="green")

#calculate risk
x1<-1:39
fm_2<-(1-Sm2$smt[299:337]/smt1[x1])
fm2<-abs(fm_2)
for (i in 2:39) {
  fm2[i]<-abs(fm2[i-1])+abs(fm_2[i]*fm2[i-1])
}
plot(x1, abs(fm2),col="red", type = "l", ylim=range(0,10))
lines(x1,abs(fm_2), col = "black", type = "l",lty=4)
#error
error <- smt1 - Sm_prediction         # error6
predictionRMSE <- rmse(error) #30196.102
fin.pr<- sigm(error)
print(fin.pr)
#средняя ошибка аппроксимации
croshappr<- sum(abs(error/smt1))/length(smt1)*100
print(croshappr)
}

#F-series
function_fractal_f-ser<- function(sft, sft1, sf){

random_sample_count <- 39
Sf1 <- as.data.frame(sft1, row.names = NULL) #test
Sf2 <- as.data.frame(sft, row.names = NULL)  #train

#do 39 predictions of next values in Sm
for(i in 1:39){
  delta <- c()

  for(j in 2:length(sft)){
    delta <- rbind(delta, (sft[j]-sft[j-1]))
  }
  # calculate standard deviation of delta
  Std_delta <- apply(delta, 2, sd)
  V_Reference <- fd.estimate(sft, method=method, trim=TRUE)$fd
  Sf_guesses  <-  rnorm(random_sample_count  , mean=sft[length(sft)],  sd
=Std_delta)
  minDifference = 1000000
  for(j in 1:length(Sf_guesses)){
    new_Sf <- rbind(Sf2, Sf_guesses[j])
    new_V_Reference    <-    fd.estimate(new_Sf$sft,    method=method,
trim=TRUE)$fd

    if (abs(new_V_Reference - V_Reference) < minDifference ){
      Sf_prediction <- Sf_guesses[j]
      minDifference = abs(new_V_Reference - V_Reference)
    }
  }
  print(i)
  Sf2 <- rbind(Sf2, Sf_prediction)
}
#plot(Sf2)
plot(Sf2$sft,    type="l",    xlab="Дата",    ylab="Объем    продаж",
main="Фрактальный анализ для модели F-series")
lines(as.data.frame(sf[1:(endingIndex+39)],    row.names    =    NULL),
col="blue")
lines(299:337, sft1, col="green")
#calculate risk
x1<-1:39
ff_2<-(1-Sf2$sft[299:337]/sft1)
ff2<-abs(ff_2)
for (i in 2:39) {
  ff2[i]<-abs(ff2[i-1])+abs(ff_2[i]*ff2[i-1])
}
plot(x1, abs(ff2),col="red", type = "l", ylim=range(0,10))
lines(x1,abs(ff_2), col = "black", type = "l",lty=4)
#error
error <- sft1 - Sf_prediction         # error6
predictionRMSE <- rmse(error) #30196.102
fin.pr<- sigm(error)
print(fin.pr)
#средняя ошибка аппроксимации
croshappr<- sum(abs(error/sft1))/length(sft1)*100
print(croshappr)
}
#Explorer_____________________
function_fractal_must<- function(smt, smt1, sm){

random_sample_count <- 39
Se1 <- as.data.frame(set1, row.names = NULL) #test
Se2 <- as.data.frame(set, row.names = NULL)  #train

#do 39 predictions of next values in Sm
for(i in 1:39){
  delta <- c()
  for(j in 14:length(set)){
    delta <- rbind(delta, (set[j]-set[j-1]))
  }
  # calculate standard deviation of delta
  Sed_delta <- apply(delta, 2, sd)
  V_Reference <- fd.estimate(set, method=method, trim=TRUE)$fd
  Se_guesses  <-  rnorm(random_sample_count  ,  mean=set[length(set)],  sd
=Sed_delta)
  minDifference = 1000000
  for(j in 1:length(Se_guesses)){
    new_Se <- rbind(Se2, Se_guesses[j])
    new_V_Reference    <-    fd.estimate(new_Se$set,    method=method,
trim=TRUE)$fd
    if (abs(new_V_Reference - V_Reference) < minDifference ){
      Se_prediction <- Se_guesses[j]
      minDifference = abs(new_V_Reference - V_Reference)
    }
  }
  print(i)
  Se2 <- rbind(Se2, Se_prediction)
}
#plot(Se2)
plot(Se2$set[13:337],  type="l",  xlab="Дата",  ylab="Объем  продаж",
main="Фрактальный анализ для модели Explore")
lines(as.data.frame(se[13:(endingIndex+39)],   row.names   =   NULL),
col="blue")
lines(287:325, set1, col="green")
#calculate risk
x1<-1:39
fe_2<-(1-Se2$set[287:325]/set1)
fe2<-abs(fe_2)
for (i in 2:39) {
  fe2[i]<-abs(fe2[i-1])+abs(fe_2[i]*fe2[i-1])
}
plot(x1, abs(fe2),col="red", type = "l", ylim=range(0,10))
lines(x1,abs(fe_2), col = "black", type = "l",lty=4)
#error
error <- set1 - Se_prediction         # error6
predictionRMSE <- rmse(error) #30196.102
fin.pr<- sigm(error)
print(fin.pr)
#средняя ошибка аппроксимации
croshappr<- sum(abs(error/set1))/length(set1)*100
print(croshappr)
}

f.plot_risk_m<- function()
{
  x1<-1:39
  #svm
  e_2<-(1- prY2_m[299:337]/smt1)
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  #regress
  e_r<- (1-lpr_tr_m_s[299:337]/smt1)
  er<- abs(e_r)
  for (i in 2:39) {
    er[i]<-abs(er[i-1])+abs(e_r[i]*er[i-1])
  }
  #avtoregr
  e_ar<- (1-future_ar_m$mean[x1]/smt1)
  ear<- abs(e_ar)
  for (i in 2:39) {
    ear[i]<-abs(ear[i-1])+abs(e_ar[i]*ear[i-1])
  }
  #wavelet
  e_wv<- (1-fin_m[x1]/smt1)
  ewv<- abs(e_wv)
  for (i in 2:39) {
    ewv[i]<-abs(ewv[i-1])+abs(e_wv[i]*ewv[i-1])
  }
  #fractal
  x1<-1:39
  fm_2<-(1-Sm2$smt[299:337]/smt1)
  fm2<-abs(fm_2)
  for (i in 2:39) {
    fm2[i]<-abs(fm2[i-1])+abs(fm_2[i]*fm2[i-1])
  }
  plot(x1,  abs(e2),col="blue",  type  =  "b",  ylim=range(0,3),  pch=4,
main = "Mustang") # x
  lines(x1,abs(e_2), col = "black", type = "b",lty=4, pch =4)
  lines(x1,abs(er),  col  =  "blue",  type  =  "b",lty=5,  pch  =17)  #
triangle black
  lines(x1,abs(e_r), col = "black", type = "b",lty=5, pch =17)
  lines(x1,abs(ear),  col  =  "blue",  type  =  "b",lty=6,  pch  =1)  #  ring
white
  lines(x1,abs(e_ar), col = "black", type = "b",lty=6, pch =1)
  lines(x1,abs(ewv), col = "blue", type = "b",lty=6, pch =23)  # romb
  lines(x1,abs(e_wv), col = "black", type = "b",lty=6, pch =23)
  lines(x1,abs(fm2),  col  =  "blue",  type  =  "b",lty=6,  pch  =2)    #
triangle white
  lines(x1,abs(fm_2), col = "black", type = "b",lty=6, pch =2)
  #print in file
  m_b<- data.frame (future_ar_m$mean, fin_m,  e_ar,  e_wv)
  must<-  data.frame(lpr_tr_m_s[299:337],  e_r,  prY2_m[299:337],  e_2,
future_ar_m$mean,   e_ar, fin_m, e_wv, Sm2$smt[299:337], fm_2 )
  write.table(must, file = file.choose())
  write.table(m_b, file = file.choose())
}
f.plot_risk_f<- function() #plot risk f-series sales
{
  x1<-1:39
  #svm
  e_2<-(1-prY2_f[299:337]/sft1)
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  #regress
  e_r<- (1-lpr_tr_f_s[299:337]/sft1)
  er<- abs(e_r)
  for (i in 2:39) {
    er[i]<-abs(er[i-1])+abs(e_r[i]*er[i-1])
  }
  #avtoregr
  e_ar<- (1-future_f_s$mean[x1]/sft1)
  ear<- abs(e_ar)
  for (i in 2:39) {
    ear[i]<-abs(ear[i-1])+abs(e_ar[i])*ear[i-1]
  }
  #wavelet
  e_wv<- (1-fin_f[x1]/sft1)
  ewv<- abs(e_wv)
  for (i in 2:39) {
    ewv[i]<-abs(ewv[i-1])+abs(e_wv[i]*ewv[i-1])
  }
  #fractal
  ff_2<-(1-Sf2$sft[299:337]/sft1)
  ff2<-abs(ff_2)
  for (i in 2:39) {
    ff2[i]<-abs(ff2[i-1])+abs(ff_2[i]*ff2[i-1])
  }
  plot(x1,  abs(e2),col="blue",  type  =  "b",  ylim=range(0,1),  pch=4,
main = "F-series") # x
  lines(x1,abs(e_2), col = "black", type = "b",lty=4, pch =4)
  lines(x1,abs(er),  col  =  "blue",  type  =  "b",lty=5,  pch  =17)  #
triangle black
  lines(x1,abs(e_r), col = "black", type = "b",lty=5, pch =17)
  lines(x1,abs(ear),  col  =  "blue",  type  =  "b",lty=6,  pch  =1)  #  ring
white
  lines(x1,abs(e_ar), col = "black", type = "b",lty=6, pch =1)
  lines(x1,abs(ewv), col = "blue", type = "b",lty=6, pch =23)  # romb
  lines(x1,abs(e_wv), col = "black", type = "b",lty=6, pch =23)
  lines(x1,abs(ff2),  col  =  "blue",  type  =  "b",lty=6,  pch  =2)    #
triangle white
  lines(x1,abs(ff_2), col = "black", type = "b",lty=6, pch =2)
  #print in file
  #print all in file
  fser<-  data.frame(lpr_tr_f_s[299:337],  e_r,  prY2_f[299:337],  e_2,
future_f_s$mean,   e_ar, fin_f, e_wv, Sf2$sft[299:337], ff2 )
  write.table(fser, file = file.choose())
}
f.plot_risk_e<- function() #plot risk explorer
{
  x1<-1:39
  #svm
  e_2<-(1-prY2_e[287:325]/set1)
  e2<-abs(e_2)
  for (i in 2:39) {
    e2[i]<-abs(e2[i-1])+abs(e_2[i]*e2[i-1])
  }
  #regress
  e_r<- (1-lpr_tr_e_s[287:325]/set1)
  er<- abs(e_r)
  for (i in 2:39) {
    er[i]<-abs(er[i-1])+abs(e_r[i]*er[i-1])
  }
  #avtoregr
  e_ar<- (1-future_e_s$mean[x1]/set1)
  ear<- abs(e_ar)
  for (i in 2:39) {
    ear[i]<-abs(ear[i-1])+abs(e_ar[i]*ear[i-1])
  }
  #wavelet
  e_wv<- (1-fin_e[x1]/set1)
  ewv<- abs(e_wv)
  for (i in 2:39) {
    ewv[i]<-abs(ewv[i-1])+abs(e_wv[i]*ewv[i-1])
  }
  #fractal
  fe_2<-(1-Se2$set[287:325]/set1)
  fe2<-abs(fe_2)
  for (i in 2:39) {
    fe2[i]<-abs(fe2[i-1])+abs(fe_2[i]*fe2[i-1])
   }
  plot(x1,  abs(e2),col="blue",  type  =  "b",  ylim=range(0,4),  pch=4,
main = "Explorer")
  lines(x1,abs(e_2), col = "black", type = "b",lty=4, pch =4)
  lines(x1,abs(er), col = "blue", type = "b",lty=5, pch =17)
  lines(x1,abs(e_r), col = "black", type = "b",lty=5, pch =17)
  lines(x1,abs(ear), col = "blue", type = "b",lty=6, pch =1)
  lines(x1,abs(e_ar), col = "black", type = "b",lty=6, pch =1)
  lines(x1,abs(ewv), col = "blue", type = "b",lty=6, pch =23)
  lines(x1,abs(e_wv), col = "black", type = "b",lty=6, pch =23)
  lines(x1,abs(fe2),  col  =  "blue",  type  =  "b",lty=6,  pch  =2)    #
triangle white
  lines(x1,abs(fe_2), col = "black", type = "b",lty=6, pch =2)
  #print all in file
  fser<-  data.frame(lpr_tr_e_s[287:325],  e_r,  prY2_e[299:337],  e_2,
future_e_s$mean,   e_ar, fin_f, e_wv, Se2$set[287:325], fe_2 )
  write.table(fser, file = file.choose())
  fser2<-  data.frame(lpr_tr_e_s[287:325],  er,  prY2_e[299:337],  e2,
future_e_s$mean,   ear, fin_f, ewv, Se2$set[287:325], fe2 )
  write.table(fser2, file = file.choose())
}
```