Machine Learning Group 1

**Support Vector Machine(Regression)**

# Group Members

## William Clark

Sumati Kulkarni

Babak Maleki Shoja

Venkatesh Reddy Pala

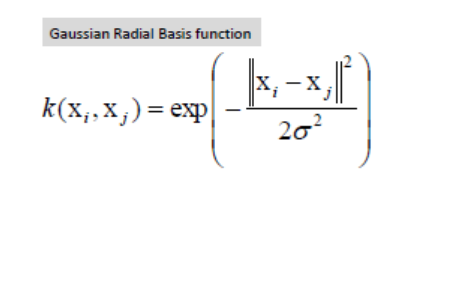
Vishwa Patel

**Support Vector Regression:**

Support Vector Machine can also be used as a regression method, maintaining all the main features that characterize the algorithm (maximal margin). The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem. But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken in consideration. However, the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.

The kernel we have selected is Radial Basis Function and Polynomial function.

The formula for Radial Basis Function is mentioned below:

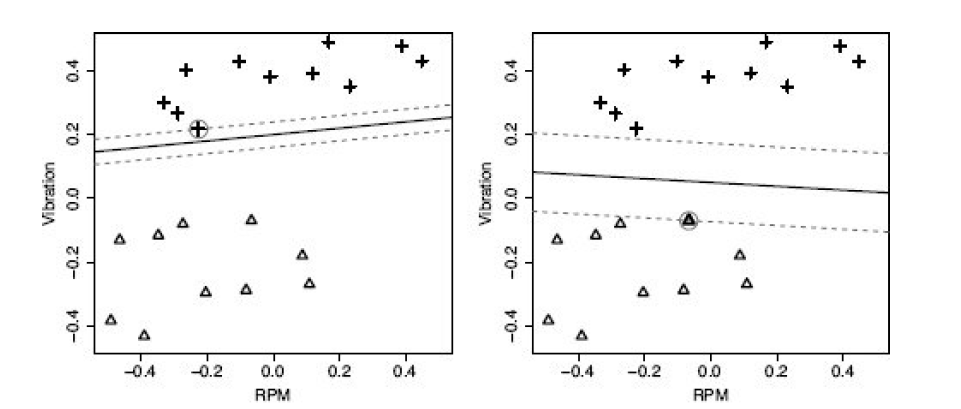


**NOTE:** Here we have used cropped image of formula of Gaussian Radial Basis Function because the formula mentioned above is difficult to write because of the mode between the both variables on right side.

The formula for polynomial function is:

**k(xi, xj) = (xi.xj)d**

For example, here an example from book is mentioned below. This diagram shows the clear boundary for 2 variables good and bad.



The above-mentioned figure shows a diagram with different decision boundary,

which has a much larger margin. The intuition behind support vector machines is that this

second decision boundary should distinguish between the two target levels much more

reliably than the first. Training a support vector machine involves searching for the

decision boundary, or **separating hyperplane**,20 that leads to the maximum margin as this

will best separate the levels of the target feature. Although the goal of finding the best

decision boundary is the same for algorithms that build support vector machines as it is for

logistic regression models, the inductive bias encoded in the algorithms to select this

boundary is different, which leads to different decision boundaries being found.

The instances in a training dataset that fall along the margin extents, and so define the

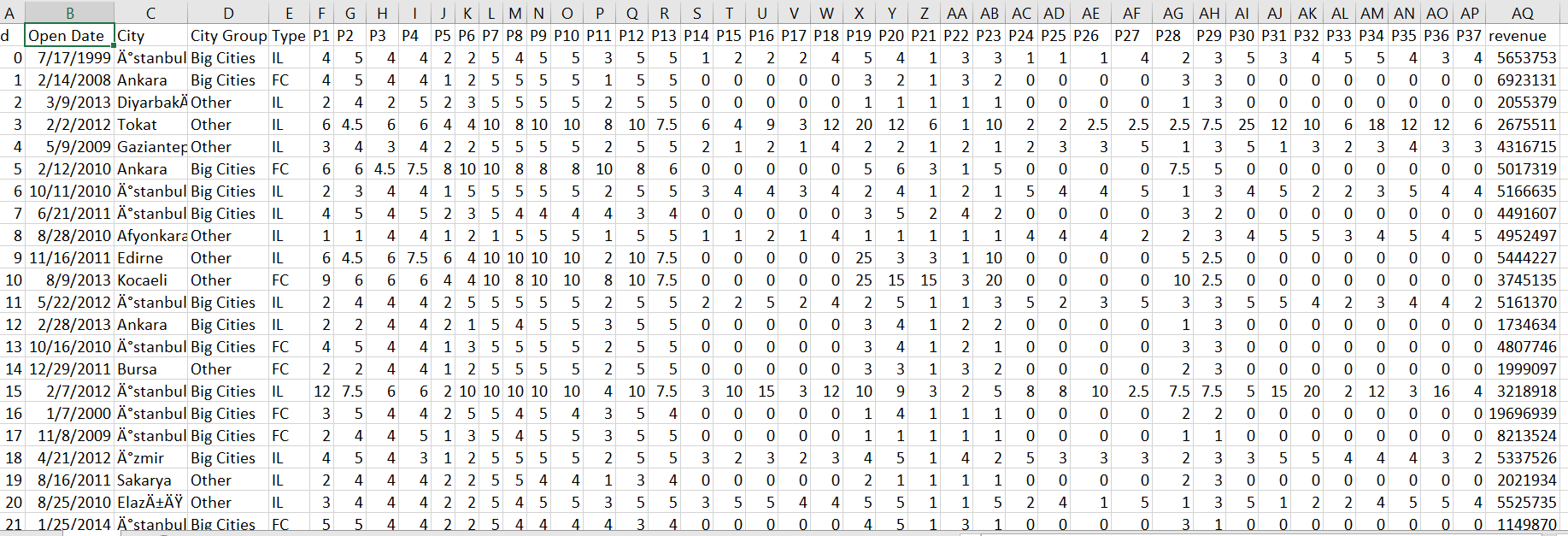
margins, are known as the **support vectors**. These are the most important instances in the

dataset because they define the decision boundary. There will always be at least one

support vector for each level of the target feature, but there is no limit to how many

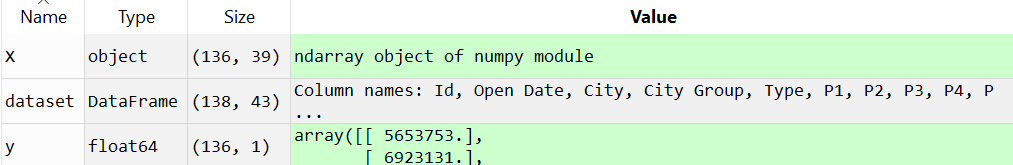
support vectors there can be in total.

**Screenshots with Explanation:**

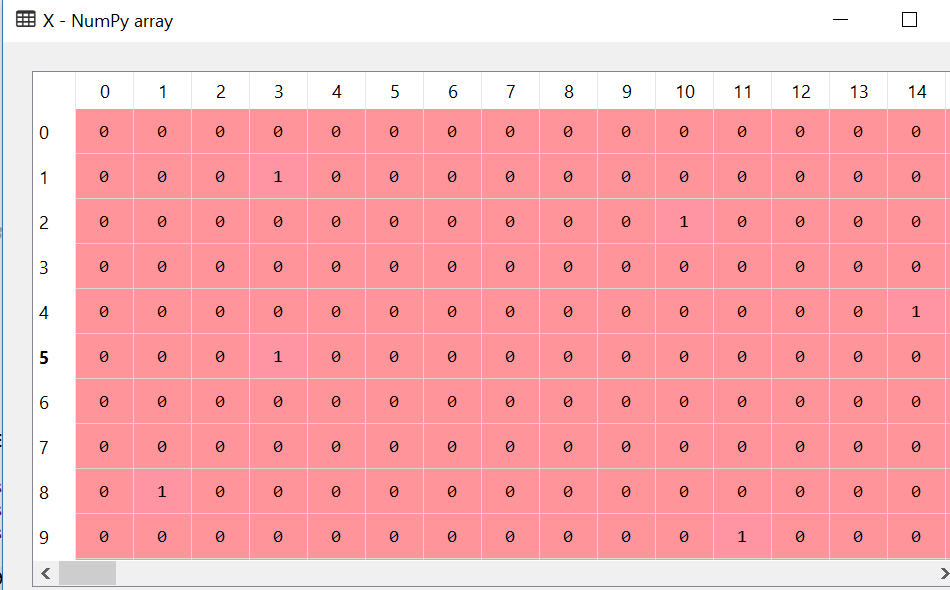


Our dataset is contained originally in single file, which is ‘train.csv’. Train file is used to provide training and testing dataset to our SVR model.

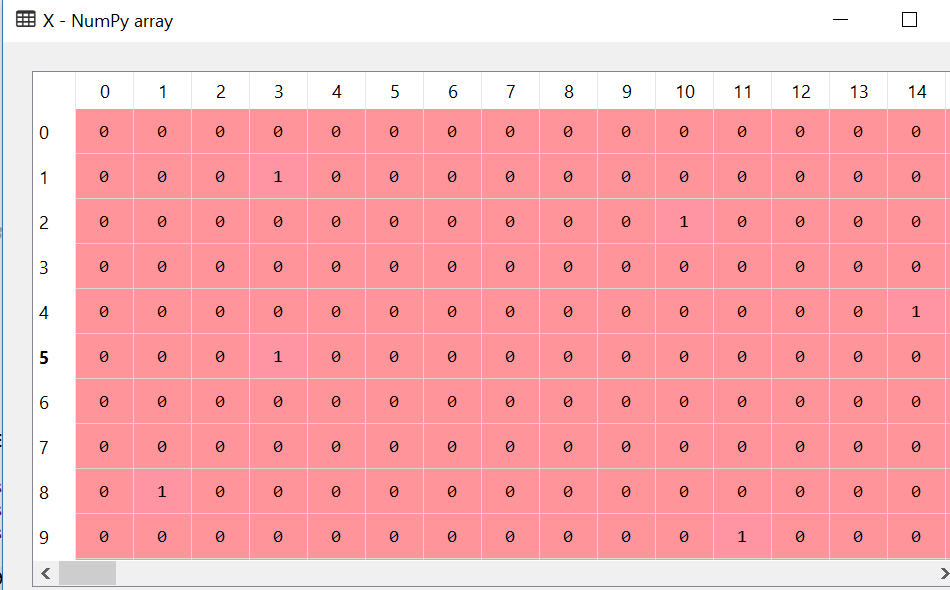
Our dataset is basically “Restaurant Revenue”, it has several field combinations of both categorical and continuous features, but the target features is continuous. We have downloaded this dataset from ‘Kaggle’ website, which is free for people who want to learn Machine Learning. Target features is how much revenue that Restaurant in that city collected in a certain amount of time.



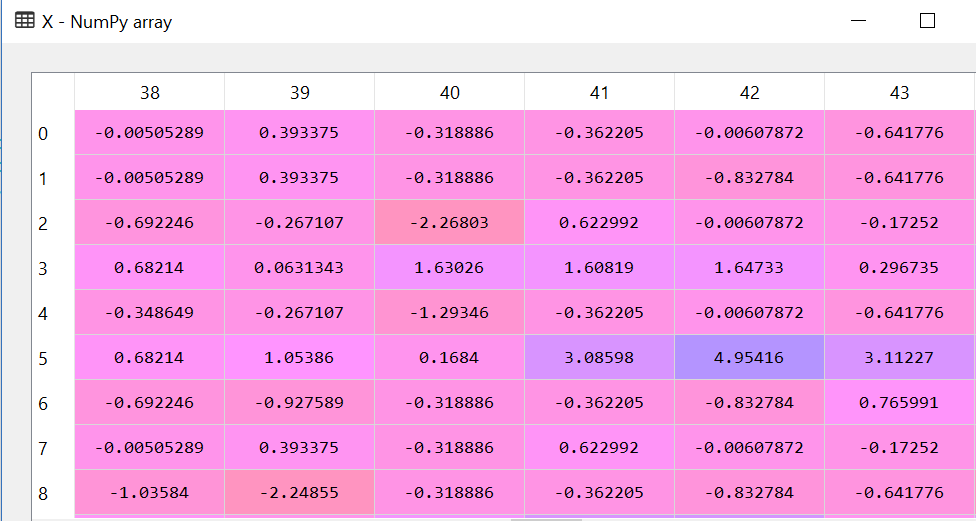
Now, we have divided the dataset into 2 matrices, one contains descriptive features(X) and other contain target feature(y). These both are selected from the ‘train.csv’ file.



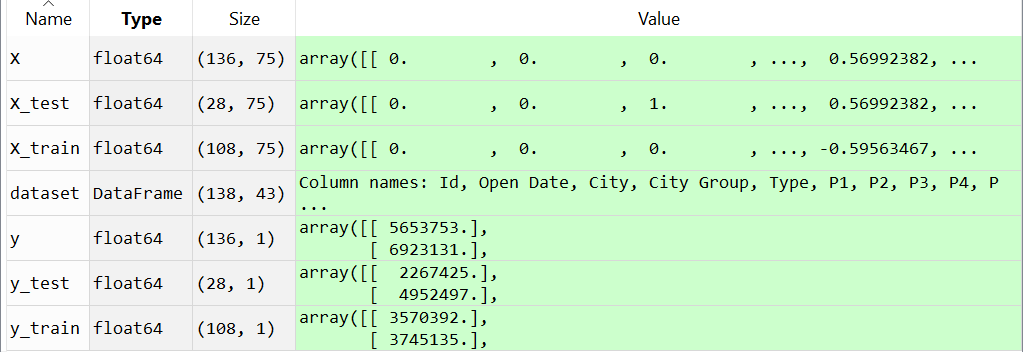
Now, the next step is to transform categorical values into continuous values by using LabelEncoder and OneHotEncoder these both libraries are used to transform the categorical data into continuous data, once the data is converted now with the help of OneHotEncoder we can create dummy variables for those categorical features. I have applied to first 3 columns for X matrix, since these 3 columns contains categorical values which are City, CityGroup and Type. After creating dummy variables, the number of columns have increased from 39 to 75, because there are more categorical variables



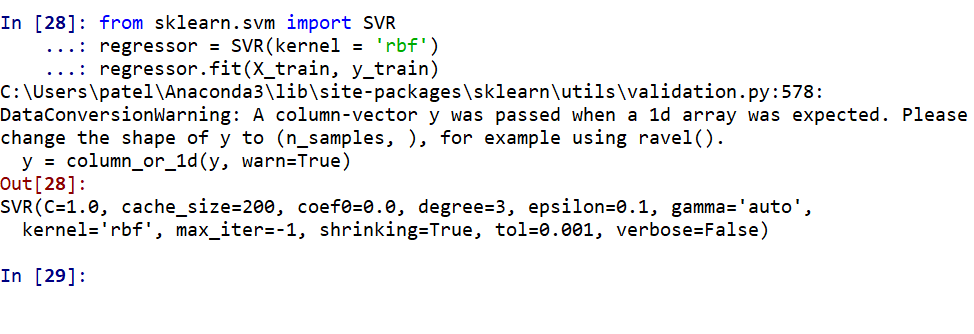
Here, in the next step we need to remove one dummy variable because if we don’t do that then it will be redundant for our model and to avoid that we remove one dummy variable, from X.



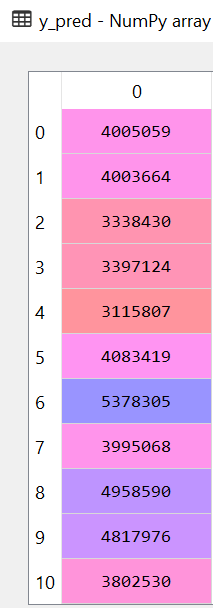
Now, the next step is to feature scale the dataset. We have scaled our dataset by using “Standardization Feature Scaling” which will transform the matrices X and y. But the trick here is that we don’t have to Standardized the dummy variables, just the rest of descriptive features. For each matrix we need to create independent objects which will be used for that matrix respectively.



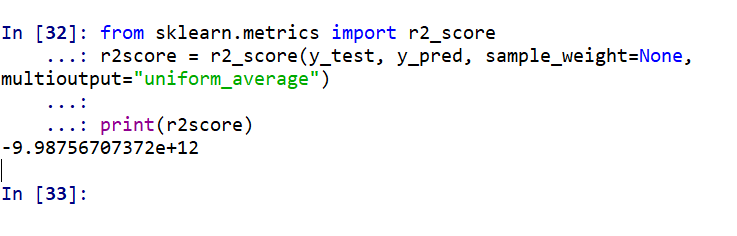
In above image, we can see that 4 new matrices have been created namely: 1) X\_train 2) X\_test 3) y\_train 4) y\_test. Test matrices contain test data while Training variables contain training data. We have given 20% of data into testing part while 80% of data is given into training part. In above image we can see the number of rows and columns in each matrices and glimpse of values.



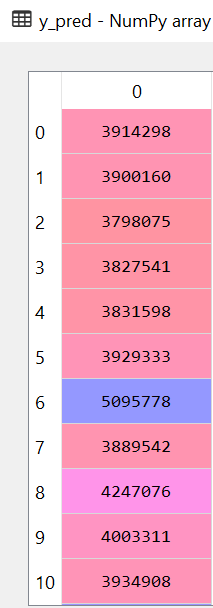
We have applied here the Support Vector Regression algorithm on our dataset and used ‘Radial Basis Function’ as our kernel. Once we fit the kernel, then we called ‘fit’ method which will take inputs X\_train and y\_train which are descriptive features and target features respectively. On right side of the screenshot, we can see that all features like size, coefficient, degree, epsilon etc.



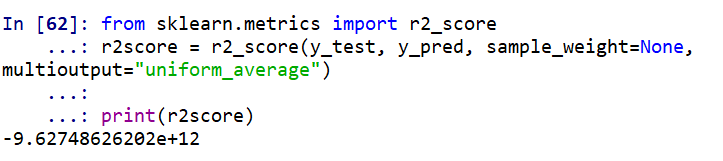
Here, we have given the X\_test matrix as input of regressor object which is an object of SVR. It is noticeable that we have used an ‘inverse\_transform’ method of StandardScalar library. The job of this function is to transform the standardized values into its original values. That is how we can have the values in money format (i.e. with standardized the data can look like 2.484939 but when we apply this function we can get the original revenue price i.e. 3204849). Here, the predicted values for test data is described, which is not in standardized form but in it’s real form. These are the total number of revenue for our test dataset.



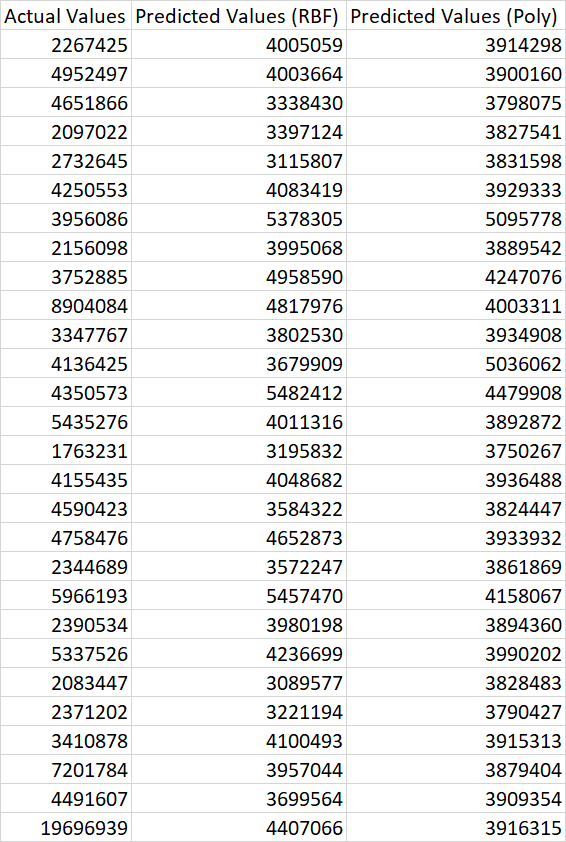
Above mentioned value is R^2 measure score for SVM for regression with Radial Basis Function as kernel. Usually in all cases, the value of R^2 measure must be between 0-1. But as we can see here that the value obtained is negative and very less (in exponential of 12). This means that the model which we have used for this dataset is wrongly chosen. Hence, we can conclude that for this dataset, SVM for regression is not a good model.



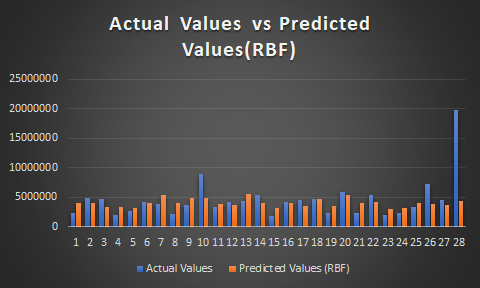
The above-mentioned screenshot is screenshot of the SVR algorithm, but the only change is kernel. In above photo, SVR algorithm is applied on test dataset by using kernel = ‘poly’.R^2 measure for SVM with regression with “Polynomial” kernel is mentioned in the next image.

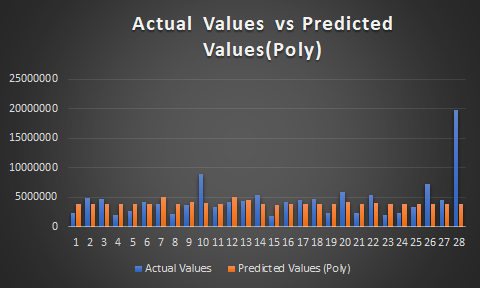


Here also the conclusion is the same, that SVM for regression is chosen by mistake. This model is not suitable for given dataset. Hence, it will be wise to use any other Regression model whoes R^2 measure comes between 0-1. The difference between results of both kernels is defined below briefly.



The relation between these columns is described below. Firstly, we will compare the Actual Value column with predicted Values with RBF kernel. Then lastly we will compare Actual Values column with predicted values with Poly kernel.





As, we can visualize from above mention graphs, that there is a major difference between the actual values and predicted values for both Radial Basis Function and Polynomial kernels of SVM for regression. And we can also see that both predicted values (RBF) and predicted values (POLY) are almost same.

**References:**

**1)** [**http://www.saedsayad.com/support\_vector\_machine\_reg.htm**](http://www.saedsayad.com/support_vector_machine_reg.htm)

**2) Machine Learning for Predictive Data Analytics by: John D. Kelleher**