Hochschule Neubrandenburg
University of Applied Sciences

Department of Landscape Sciences and Geomatics
Master's Program Geodesy and Geo-informatics
Summer-Semester 2019

Module - Data Analysis and Knowledge Processing
Subject Code - GGI.022

Classification of Selected Baked Recipes with Support Vector Machines (SVM)

Submitted by

Gavvala Rajesh
Matriculation Number: 280818
&
Omkaram Rangashesa Uday Kumar Raju
Matriculation Number:280518

Supervised by
Dr. Jochen Wauer

Table of Contents

1 Introduction....4
1.1 Why Support Vector Machines....4
1.2 How Support Vector Machines Work....5
2 Linear Classifiers....5
3 Non-linear Classifiers....9
4 Project data....11
5 Project methodology....11
6 Results....12
7 Conclusions....17
References....18

List of Figures

Figure-1: Support Vectors, classification among classes 4

Figure-2: Support Vectors with function classification 5

Figure-3: Data classification with possible hyper planes 6

Figure-4 : Non-linear data – Feature space transformation 9

Figure-5 : Non-linear data classification hyper plane with linear, polynomial, RBF 10

Figure-6: Project methodology 11

Figure-7: Selected data recipes with their ingredients 12

Figure-8: Best model accuracy and test data for linear and polynomial kernels 13

Figure-9: Muffin vs Cupcake classification and prediction 14

Figure-10: Muffin vs Scone classification and prediction 15

Figure-11: Cupcake vs Scone classification and prediction 16

1 Introduction

1.1 Why Support Vector Machines

Let's take an example that one has seen two pet's on the Park namely Dog and Cat. Both were looking similar to each other. But eventually he/she could figure it out, it's a Cat groomed like a Dog. Imagine it was challenging for a person to classify among two pets, however, to precisely classify among many pets for a computer it's really how difficult is the task. To correctly classify an object to a given set of data points, need to be written complex algorithms. In extreme cases like aforesaid example, an algorithm is suited to call Support Vector Machine (SVM) algorithm.

The objective of the SVM algorithm is to find a hyper plane in multidimensional space that distinctly classifies the given data points. Figure-1 depicts Support Vectors, hyper plane and margin for a linear plane. Red square points belong to one class of data points and blue circle points belongs to another class of data points. Support Vectors are the closest points in the hyper plane. These points are more relevant for the constructions of the classifier and separating line better by calculating margins. Hyper plane is a decision plane it separates given set of object points among two different classes. Margin is the perpendicular distance between the lines of support vectors. If the distance between support vectors of the margin is more, then it is considered as best margin, or margin distance is less considered as bad margin.

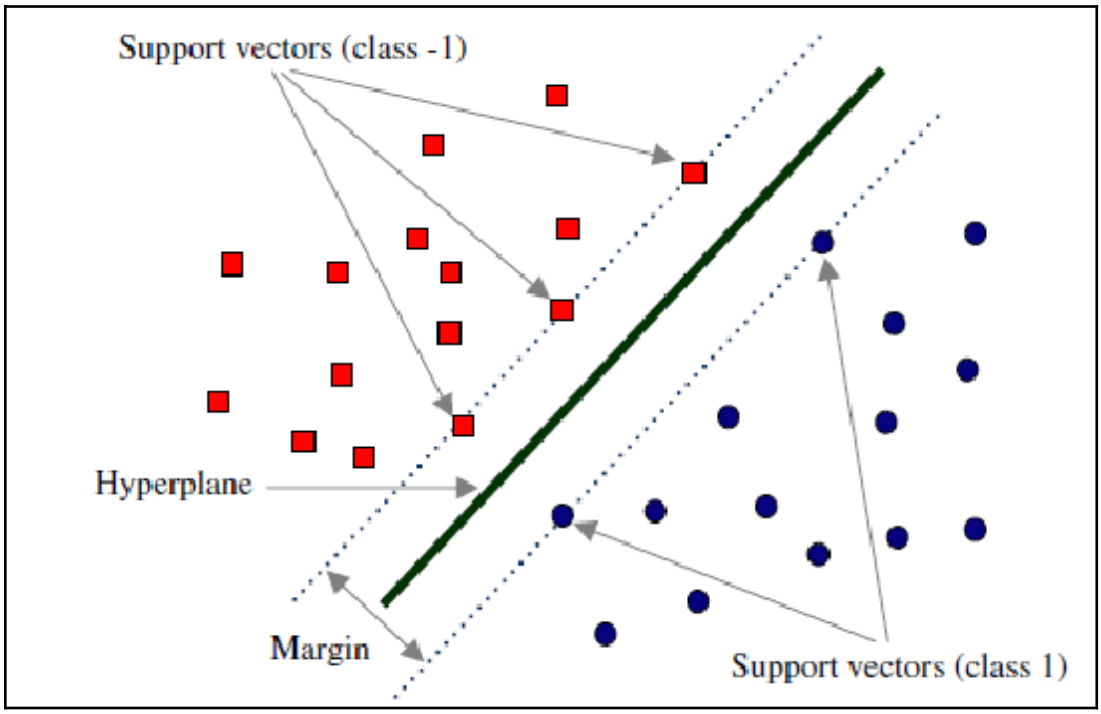

Figure-1: Support Vectors, classification among classes [1]

1.2 How Support Vector Machines Work

As discussed above, the main objective of SVM is to segregate given data points of different classes in the best possible way and to select a hyper plane with the maximum possible margin between support vectors. The SVM algorithm searches maximum marginal hyper plane in the following steps.

2 Linear Classifiers

Figure-2 illustrates the hyper plane with the maximum width of margin that separates the two classes of data. Two data classes were scale invariance, one class of data in the +1 region and another class of data in the -1 region.

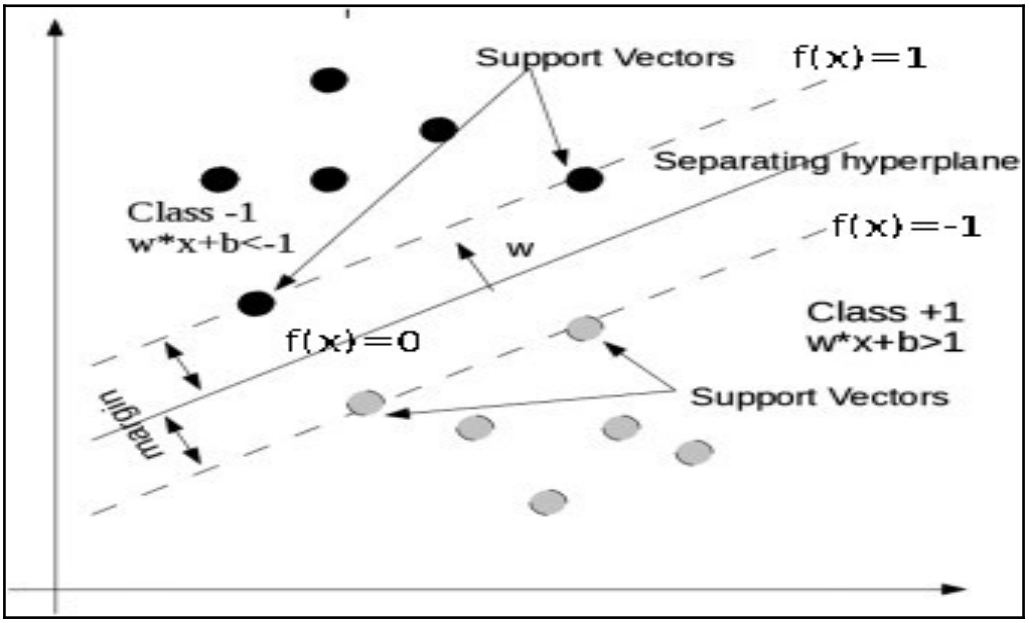

Figure-2: Support Vectors with function classification[2]

Figure-3 on the left side illustrates, the two class data set recipes of Muffin and Cupcake and their ingredients are sugar and flour are drawn in x and y and on the right side there are some possible hyper planes which separates two classes. The question arises among possible hyper planes which decision boundary (hyper plane) is better and how can quantify better and learn best parameter settings. For this the possible answers are;

1. Scale invariance; define one class in +1 region and other class in -1 region and make those regions as part as possible
2. Maximize margin width
3. Margin based classifier

from figure-2;

Hyper plane function $f(x) = w*x + b$ …….(1)
where vectors w = (w0, w1), x = (x,y) and b and equation (2) equivalent to;

$w0x + w1\ y + b = 0$
$w1y = -w0x - b$
from above we get y: $y = -(w0/w1)\ x - (b/w1)$
define $m = -(w0/w1)$ and $c = -(b/w1)$

$y = m\ x+c$……..(2)
m = slope of the hyper plane
c = y intercept

From the above equation (1), hyper plane is a set of points that satisfies the function f(x) =0
i.e., $w*x + b = 0$ ……(3)

if $f(x^+) > 1$, $w*x^+ + b > 1$ class +1 region
where as $f(x^-)<1$, $w*x^- + b < 1$ class -1 region
closest two points on the hyper plane margin satisfies;
$x^+ = x^- + r\ w$
$w*x^+ + b = 1$ & $w*x^- + b = -1$
$w * (x^- + rw) + b = 1$
$r\ w^2 + w\ x^- + b = 1$
$r\ w^2 - 1 = 1$
$r = 2 / \ ||w^2||$

Margin width $M = ||x^+ - x^-|| = ||r\ w||$
$= (2 / (||w^2||)) * ||w||$
$= 2 / (sqrt(w^T w)$ ……...(4)

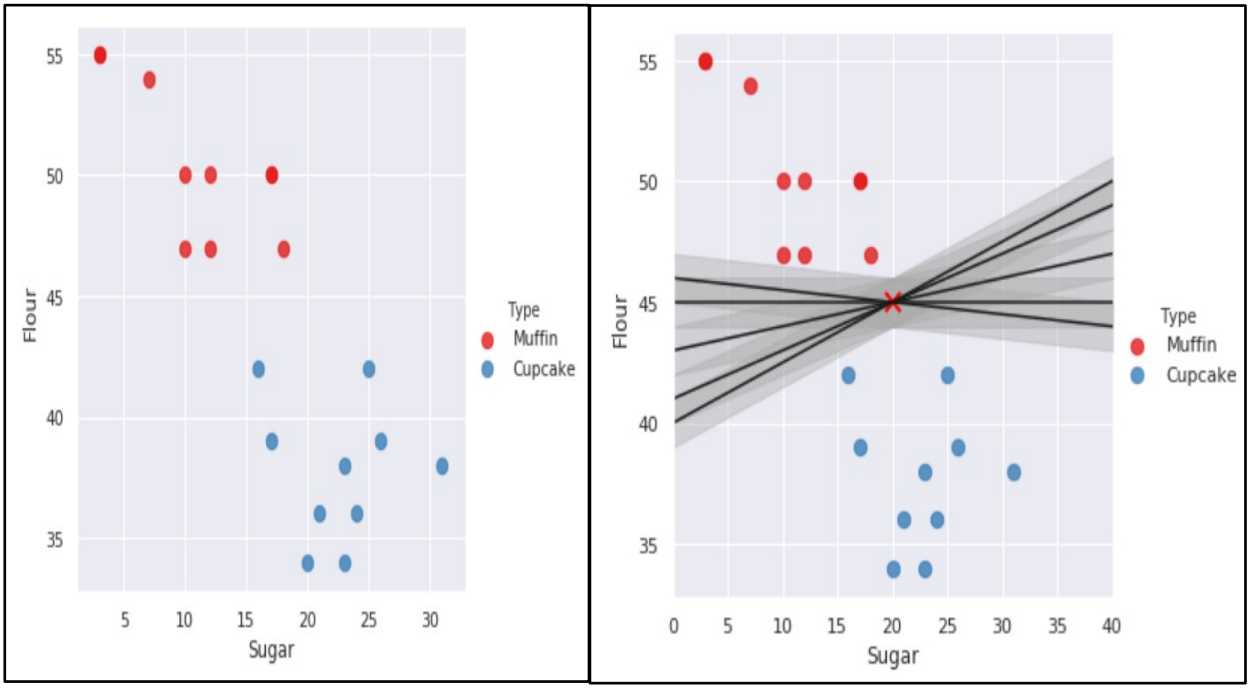

Figure-3: Data classification with possible hyper planes

To optimize the parameters, constrained optimization problem comes into the picture. maximize the margin equation subject to the constraints, all data should be on the correct side of the margin. Finding the value of w it maximizes the 1/w value and equivalently find the value of w that minimizes that the squared length.

$w = \max_w 2 / (\text{sqrt}(w^T * w)$

$w = \min_w \Sigma_j w_j^2$

to minimize w want to optimize constrained system

$w = \min_w \Sigma_j w_j^2$ subjected to

$y^{(i)} (w x^{(i)} + b) >= 1$……..(5)

$f(\theta) = \min_{w,b} \Sigma_j w_j^2$ and $g_i(\theta) = 1- y^{(i)} (w x^{(i)} + b) <= 0$

$\theta = (w,b)$

introduce Lagrange multiplier α for each constraints

$\theta = \min_\theta \max_{(\alpha>=0)} f(\theta) + \Sigma_i \alpha_i g_i(\theta)$

θ and α together can optimize with a simple constraint set through gradient steps;

$g_i(\theta) <= 0 : \alpha_i = 0$ and $g_i(\theta) > 0 : \alpha_i$ tends to ∞

Any optimum of the original problem is a saddle point of Lagrangian and vice versa. This converts a set of constraints that were difficult to satisfy into a nearly unconstrained easily satisfied constraint problem over more variables. To deal with inequality constraints, additional requirement also be satisfied called KKT complimentary slackness.

$w = \min_\theta \max_{(\alpha>=0)} ½ \Sigma_j w_j^2 + \Sigma_i \alpha_i (1- y^{(i)} (w x^{(i)} + b))$

Solving the minimization problem involves taking the partial derivatives of Lagrangian with respect to w and b.

$w = \Sigma^m_{i=1} \alpha_i y^{(i)} x^{(i)}$

and since any support vector has y = wx+b

by using equation (5);

$y^{(i)} (w x^{(i)} + b) -1 >=0$

What it says is that the closest points to the hyper plane will have a functional margin of 1

$y^{(i)} (w x^{(i)} + b) = 1$……..(6)

multiply $y^{(i)}$ on the both sides, as $(y^{(i)})^2 = 1$;

$(w x^{(i)} + b) = y^{(i)}$

$b = y^{(i)} - w x^{(i)}$

with the random support vector $x^{(i)}$ the solution is does not stable, so in order to get the stable solution taking the average provides us with a numerically more stable solution;
$b = 1/N_{sv} \Sigma_{(i \in sv)} (y^{(i)} - w x^{(i)})$

N_{sv} is the total number of support vectors

This formulation of SVM is called hard margin SVM. Sometimes it might nor work when the data is not linearly separable, in this case the other type formulation called soft margin SVM is used.

In the soft margin SVM, the goal is to allow some classification errors, but restrict the errors as possible as low. To do so, the slack variable ζ (zeta) is introduced in the constraints of the optimization problem.

The equation (5) becomes y(i) (w x(i) + b) >= 1 – ζi
minimize (w,b,ζ) = ½ Σj wj2 + Σi ζi
to control over the soft margin introduce C; minimize (w,b,ζ) = ½ Σj wj2 + C Σi ζi

If the data is non separable ,the Soft margin optimization:

$$w^* = \arg \min_{w,c} \Sigma j\ wj2 + C\ \Sigma i\ \zeta i$$

$$y(i)\ (w\ x(i) + b) >+ 1 - \zeta i \text{ (violate margin by } \zeta \text{)}$$
$$\zeta i >= 0$$

When we need to maximize soft margin Wolfe dual problem:

$$\max L_D = \Sigma_i\ \alpha_i - ½\ \Sigma_i \Sigma_j\ \alpha_i\ \alpha_j\ y_i y_j\ X_i . X_j$$

where $Xi . X_j$ is the dot product

$$0 <= \alpha_i <= C \text{ for any } i= 1,2,3,\ldots..m$$

$$\Sigma_i\ \alpha_i\ y_i = 0$$

here the constraint $\alpha_i >=0$ change to $0 <= \alpha_i <= C$. This constraint is also called as box constraint because the vector α_i is constrain to lie within the box with side length C.

3 Non-linear Classifiers

The provided data is linear, divide the data by using a separating hyper plane. If the data are far from linear and the data sets are inseparable. Figure-4 illustrates non linear data with the input features (x). Feature points can map from X to Φ (X), the original feature space can transform into new feature space. In some cases, it is possible that the training points are linearly separable in the transformed feature space. So usually can map the mapping data to a higher dimensional feature space. Imagine the case of use of big data set feature points, then computational costs also would affect.

For example, have m training examples that need to do m2 computations like this and each time we need to find a dot product if Xi. Xj. If these vectors have dimensionality D then finding the dot product takes computation time O (D2). Now, transform it to a higher dimensional space this Xi Xj become Φ (Xi). Φ (Xj), then this time taken will be O (D2) which will be quite large. So, normally if we transform this feature to a higher dimensional feature space. It leads to higher dimensional spaces, then higher computational cost, but we will see that by using kernel function. By using the kernel function can achieve this transformation without any major implication on computational cost.

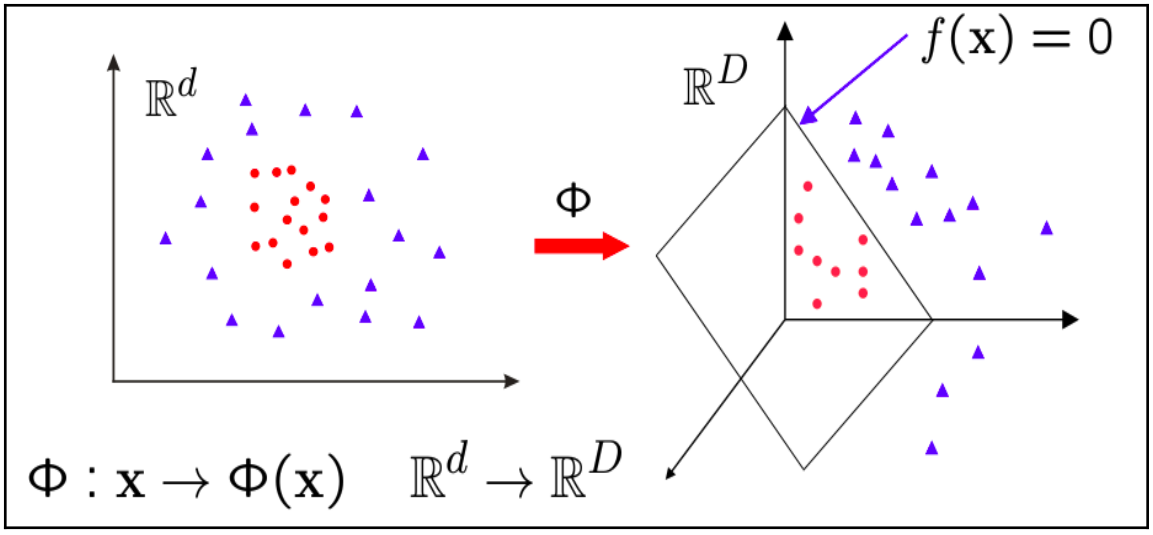

Figure-4 : Non-linear data – Feature space transformation[3]

Input data feature space is mapped to a new set of input features through feature mapping Φ. The Kernel function can be written in terms of the scalar product , canreplace X_i . X_j with $\Phi(X_i)$. $\Phi(X_j)$. For certain Φ's there is a simple operation on two vectors in the low-dimensional space that can be used to compute the scalar product in high-dimensional space.

$K(X_i,X_j) = \Phi(X_i)$. $\Phi(X_j)$ this is called kernel trick.

Where $K(X_i,X_j)$ represents kernel function, by using kernel trick, our discriminant function

$$f(X) = W\,\Phi(X_i)^T.\Phi(X) + b = \Sigma_{i\varepsilon SV}\,\alpha_i\,\Phi(X_i)^T.\Phi(X) + b$$

where α_i non zero for support vectors (SV)

The dot product of feature vectors can be used in both training and testing data sets.,The commonly used kernel functions are: Linear kernel: $K(X_i,X_j) = X_i^T . X_j$

Linear kernels are best to apply on linearly separable data. The kernel does the dot product of in the original space of the data. This kernel mostly used when there are a large number of features in a particular data set.

Radial Basis Function : $K(X_i,X_j) = \exp((-\gamma\|X_i - X_j^T\|^2) / 2\sigma^2)$

Radial Basis Function (RBF) is also known as Gaussian kernel function, that is used for infinite dimensional feature space . The σ determines the width of Gaussian kernel, standard deviation and t and σ^2 is variance. The Gaussian kernel is essentially zero if the squared distance $\|X_i - X_j^T\|^2$ is much larger than σ.

Polynomial Kernel: $K(X_i,X_j) = (X_i^T . X_j + b)^d$

polynomial contains the terms up to degree d (d>0), Polynomial Kernel looks the given input features samples and determine their similarity. From training or testing the samples can computes the vector of features and b >=0 is a free parameter, which influence the higher order terms in polynomial as well as in the low order terms in polynomial. This kernel is mostly used in image processing.

Above all functions satisfy the Mercer's condition, K is said to be non-negative definite (positive semi definite).

$$\Sigma_i\,\Sigma_i\,k(x_i, x_j)\,c_i, c_j >= 0$$

Figure-5 illustrates non-linear data classification, data separation of hyper plane with linear, polynomial and radial basis functions.

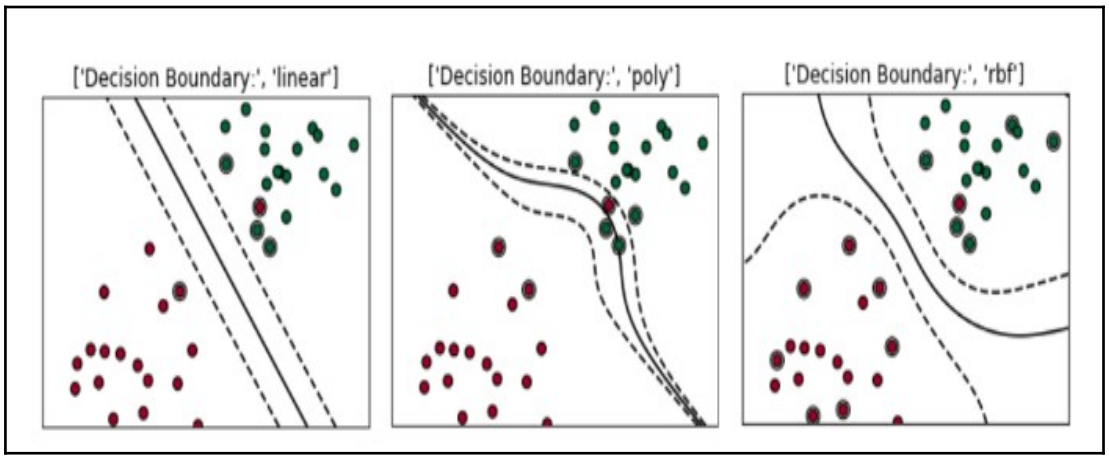

Figure-5 : Non-linear data classification hyper plane with linear, polynomial, RBF

4 Project data

Data prepared based on the selected recipes of muffin, cupcake and scone. Total 46 samples were prepared for the selected recipes, out of which 18 samples are muffin, 16 samples are cupcake and 13 samples are scones. For the three recipes common ingredients were selected, which are Flour, Milk, Sugar, Butter, Egg and Baking powder, and saved into a comma separated values (CSV).

5 Project methodology

With the prepared data for the selected recipes, python programming has written by using Jupiter Notebook to test and train the data to predict best model by using in built python libraries of SVM algorithm. Two models were considered, since its the linear classification data, SVM linear kernel and polynomial with degree 2. SVM classification has done for the data, however for the three data recipes, classification is performed as a pair of data; muffin vs cupcake, muffin vs scone and cupcake vs scone. For the classification only Sugar and Flour ingredients were considered. And also prediction check performed on this two ingredients for the pairs of data. Figure-6 below illustrates the project methodology.

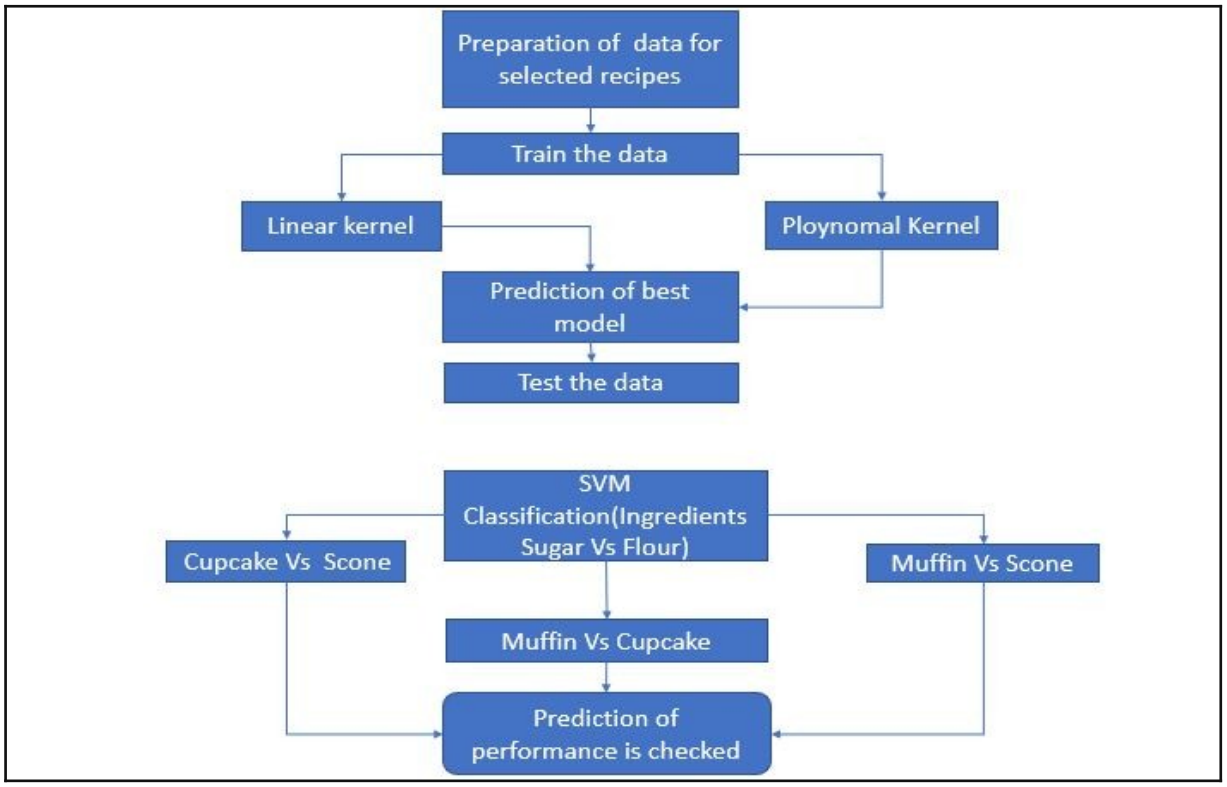

Figure-6: Project methodology

6 Results

SVM test and train data, classification & prediction results shown in below.

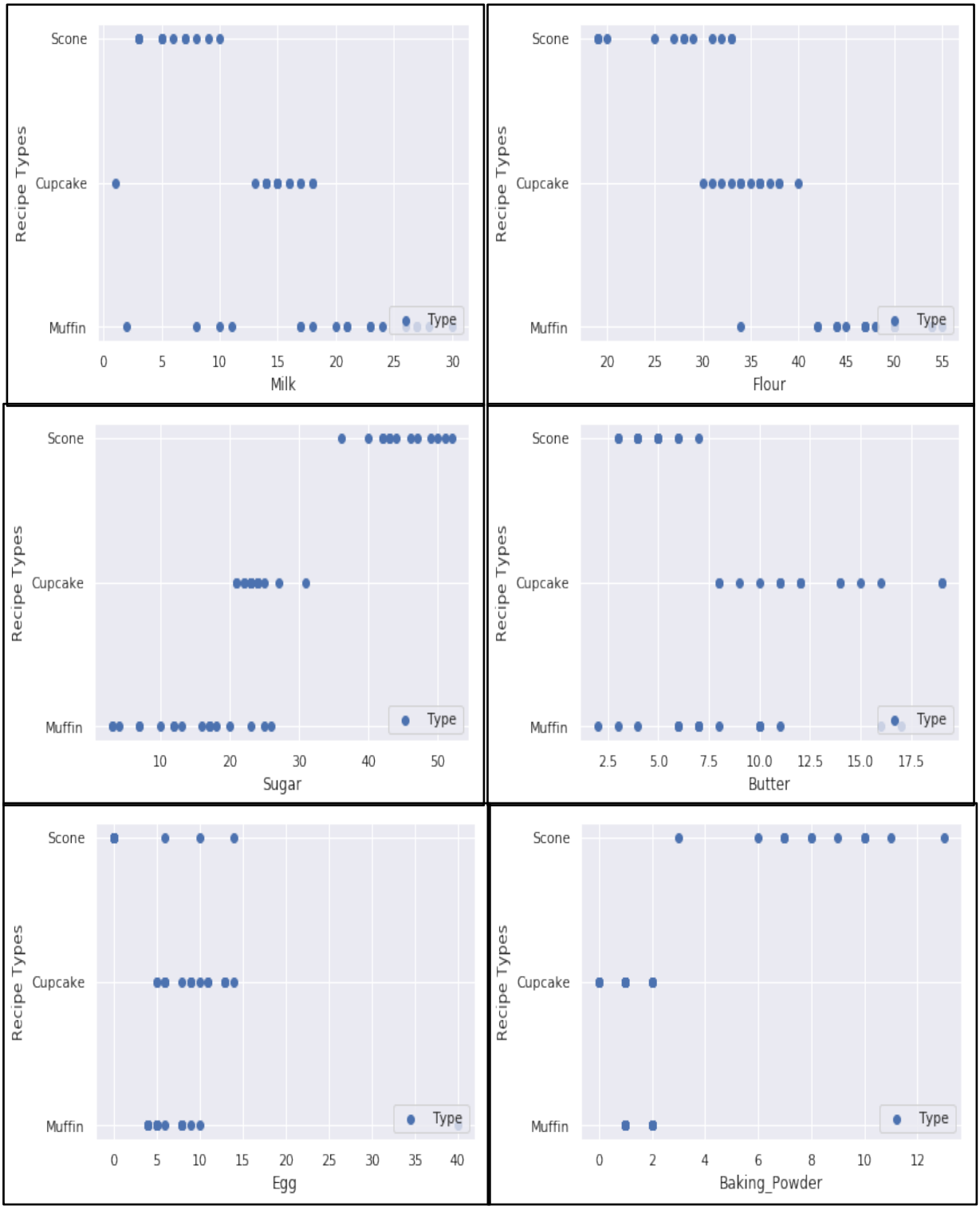

Figure-7: Selected data recipes with their ingredients

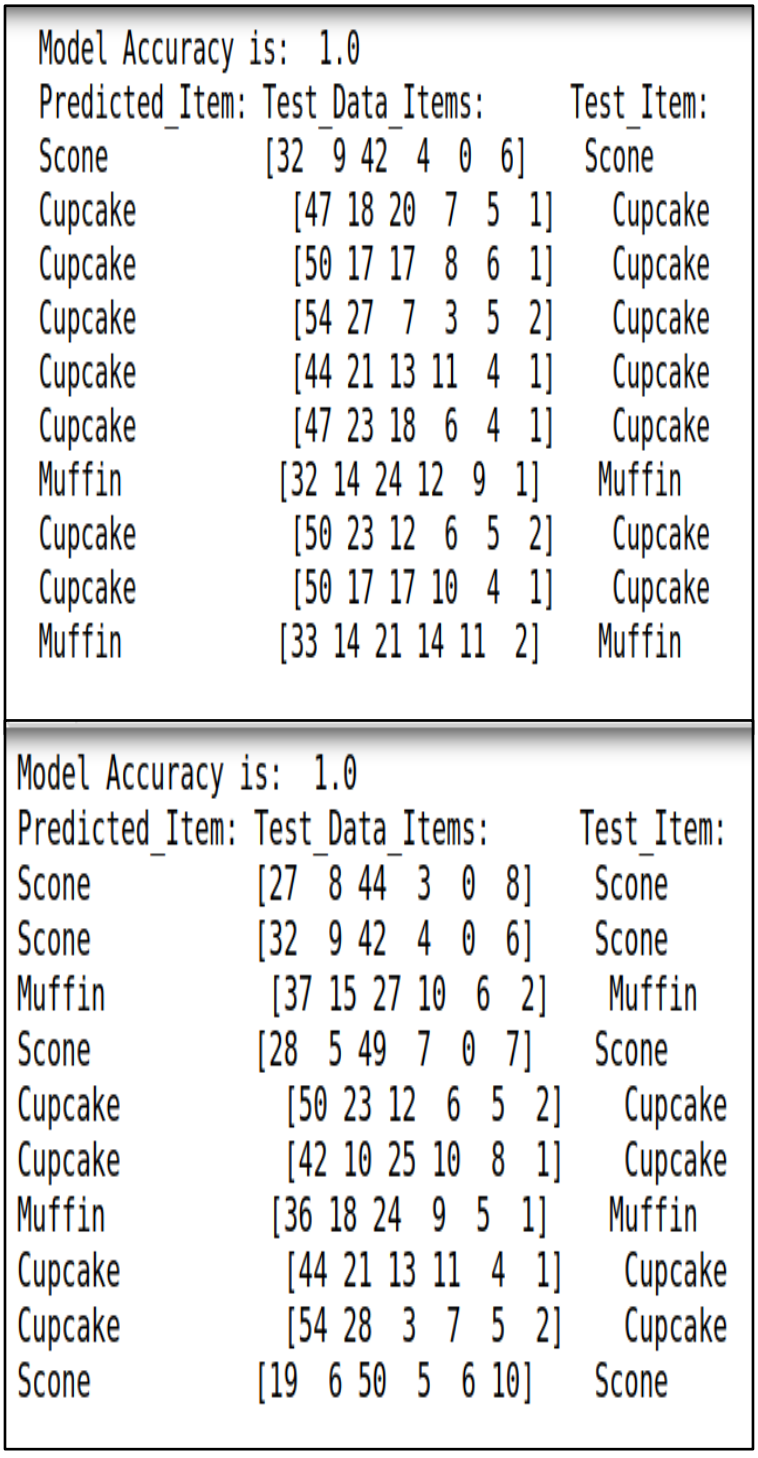

Figure-8: Best model accuracy and test data for linear and polynomial kernels

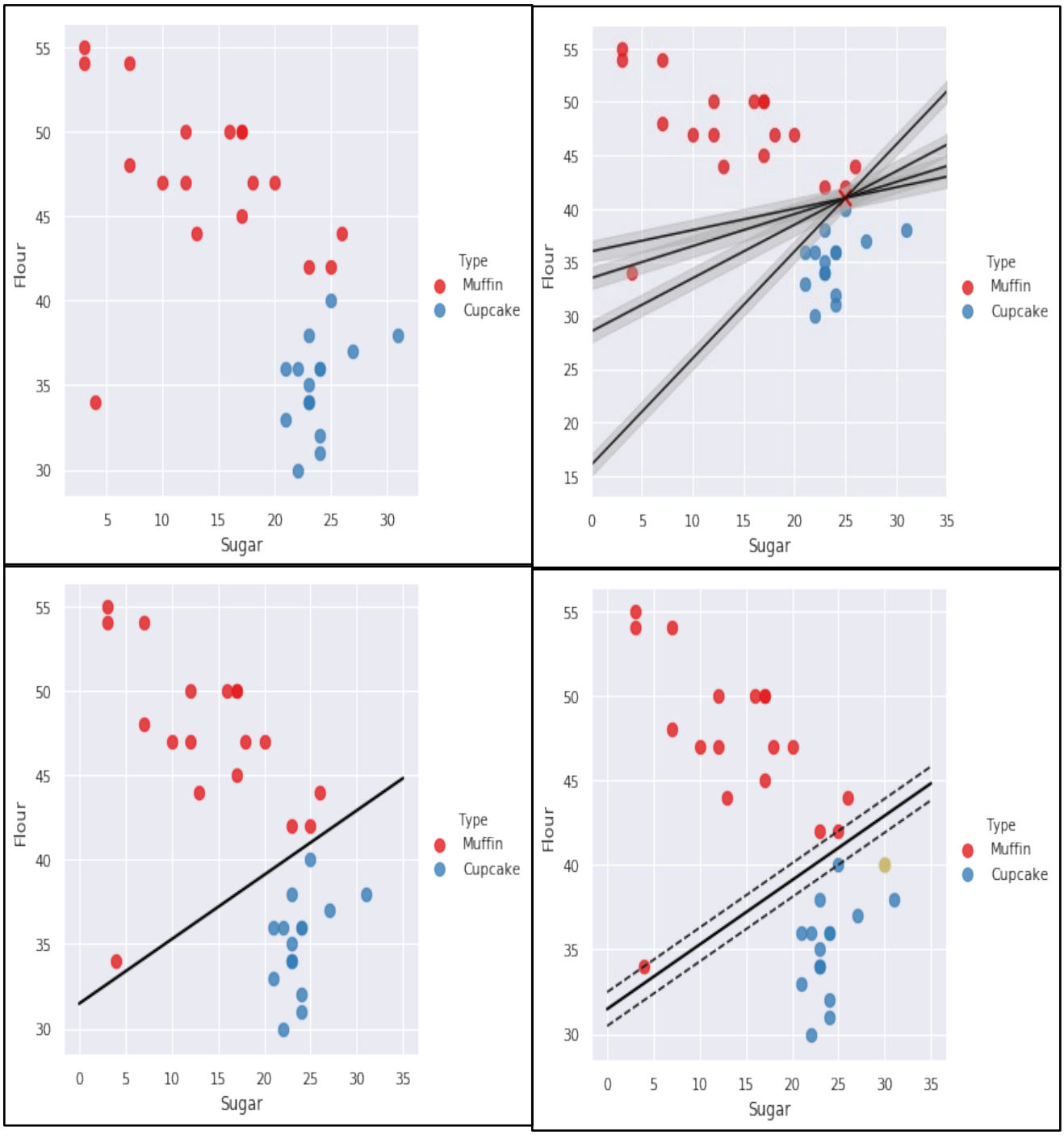

Figure-9: Muffin vs Cupcake classification and prediction

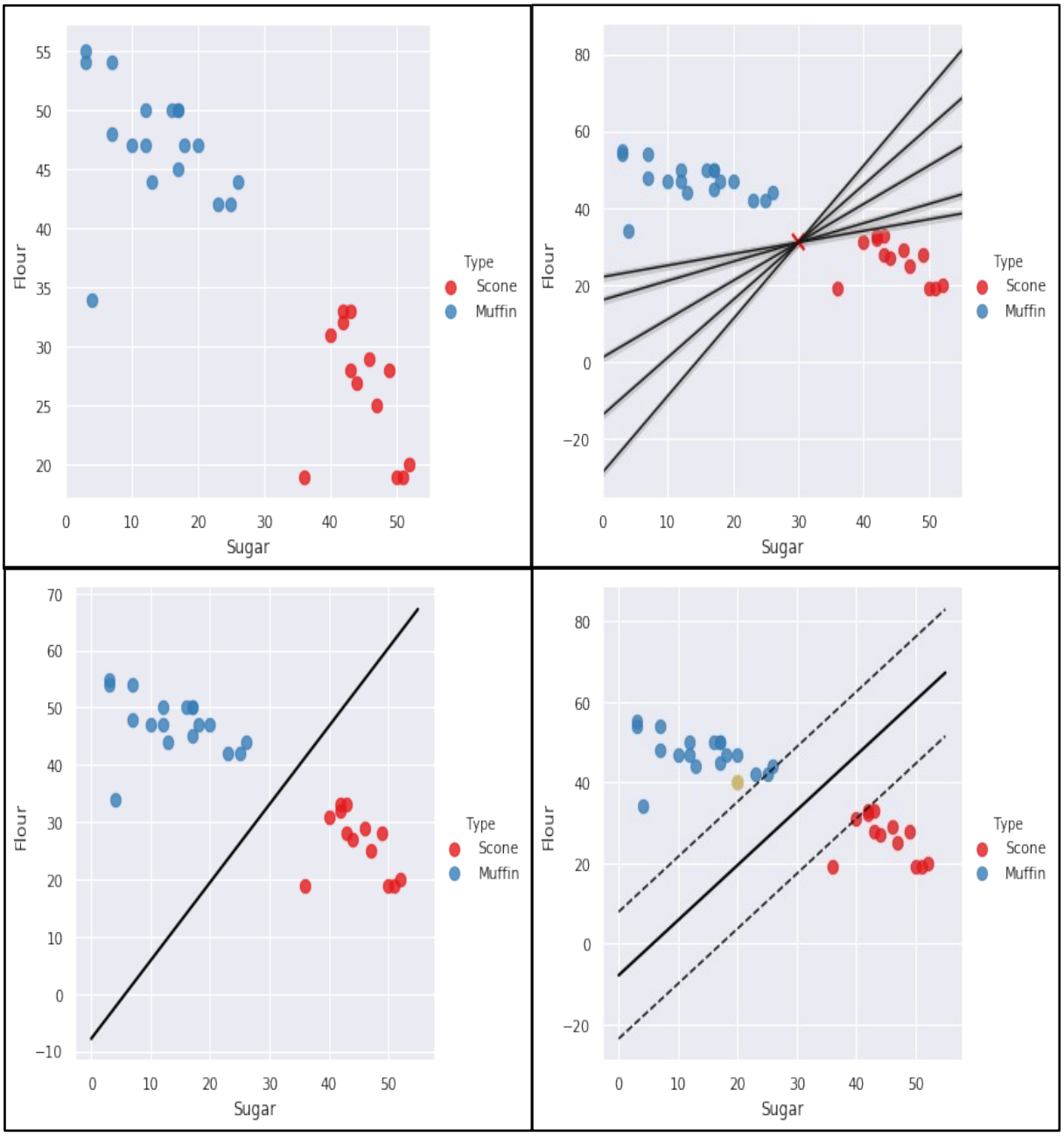

Figure-10: Muffin vs Scone classification and prediction

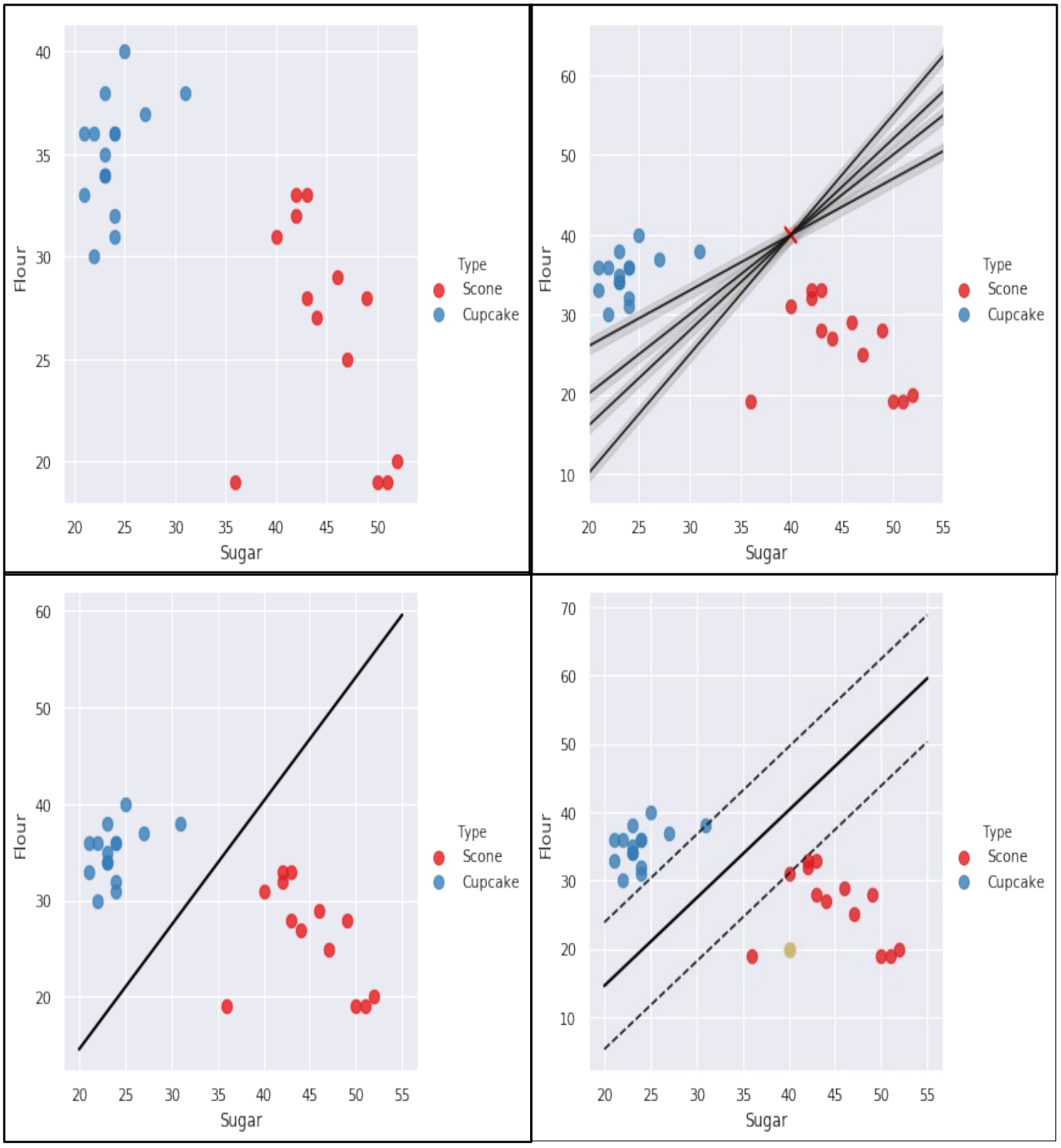

Figure-11: Cupcake vs Scone classification and prediction

7 Conclusions

with the advantages of python libraries data visualization can be done very effectively. Support Vector Machines (SVM) and Kernel Methods are supervised learning models that are trained for classification and regression. SVM are effective in high-dimensional space, even when the number of dimensions is greater than the number of sample data sets. SVM aims to find the separating hyper planes that maximize the margin between data sets. SVM are applied to solve the complex problems of high dimensionality of data.

From the results can conclude that, with model accuracy 100 %, then the test data can achieve 100% accuracy.

References

1. https://www.researchgate.net/figure/Optimal-hyperplane-for-support-vector-machine-with-two-classes-Taken-from-53_fig5_264420380

2.https://www.researchgate.net/figure/Concept-of-Support-Vector-Machines-The-optimal hyperplane-separates-two-classes-of_fig1_315784930

3. https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496

4. Ng, Andrew: Lecture Notes, CS229, Stanford University, 2009, http://cs229.stanford.edu/notes/cs229-notes3.pdf

5. Alexandre Kowalczyk, Support Vector Machines http://jermmy.xyz/images/2017-12-23/support_vector_machines_succinctly.pdf

6. Christopher Bishop (2006). Pattern Recognition and Machine Learning http://users.isr.ist.utl.pt/~wurmd/Livros/school/Bishop%20-%20Pattern%20Recognition%20And%20Machine%20Learning%20-%20Springer%20%202006.pdf

7. Tom Mitchell (1997). Machine Learning, New York, United States: McGraw Hill.

8. A. Zisserman: Lecture Notes, C19 Machine Learning, Oxford University,2015. http://www.robots.ox.ac.uk/~az/lectures/ml/lect3.pdf