Introduction to Machine Learning

ELL784 | Assignment 3

Shivangi Bithel, Yukti Kaura, Vishnu Gorty

Index

- Polynomial Curve Fitting
- Genomic Sequence Analysis

Polynomial Curve Fitting

Introduction

Polynomial Regression is a form of linear regression in which the relationship between the independent variable x and dependent variable y is modeled as an nth degree polynomial. Polynomial regression fits a nonlinear relationship for e.g. progression of disease epidemics

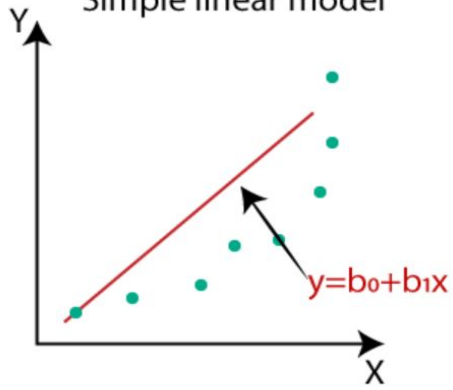
Here y is dependent variable on x, a is y intercept and e is the error rate.

In general, we can model it for nth value.

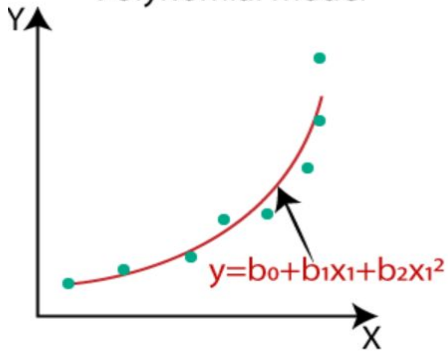
y = a + b1x + b2x^2 +....+ bnx^n + □

Since regression function is linear in terms of unknown variables, hence these models are linear from the point of estimation.

Simple linear model



Polynomial model



Problem Statement

1. To begin with, use only the first 20 data points in your file.
2. Solve the polynomial curve fitting regression problem using error function minimisation.
3. Define your own error function other than the sum-of-squares error.
4. Try different error formulations and report the results.
5. Use a goodness-of-fit measure for polynomials of different order.
6. Can you distinguish overfitting, underfitting, and the best fit?
7. Obtain an estimate for the noise variance.
8. Introduce regularisation and observe the changes. For quadratic regularisation, can you obtain an estimate of the optimal value for the regularisation parameter lambda? What is your corresponding best guess for the underlying polynomial? And the noise variance?
9. Now repeat all of the above using the full data set of 100 data points.
10. How are your results affected by adding more data? Comment on the differences.
11. What is your final estimate of the underlying polynomial? Why?

Data Description

Dataset - https://www.cse.iitd.ac.in/~sumantra/courses/ml/a3/group01.txt

Data of the form (x_i, y_i) for i = 1, ... 100. The relationship between x and y is of the form: y = w_0 + w_1 x + ... + w_M x^M + e. The error e is drawn from a Gaussian distribution with zero mean and unknown (but fixed, for a given input file) variance. M is also unknown.

Number of data points: 100

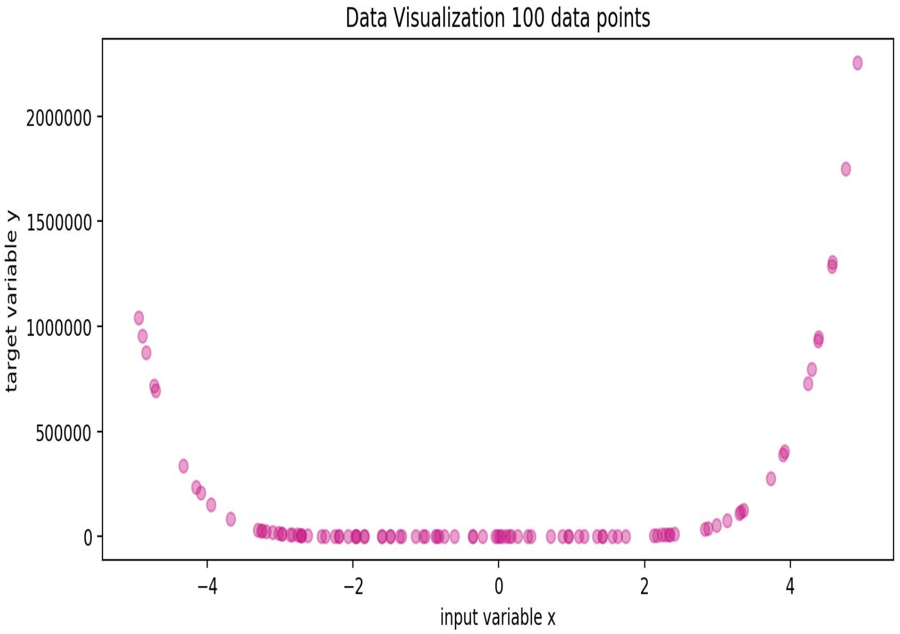
X ranges from : -4.94 to 4.92

Y ranges from : 1.21 to 2254025.74

Sample data(10 points):

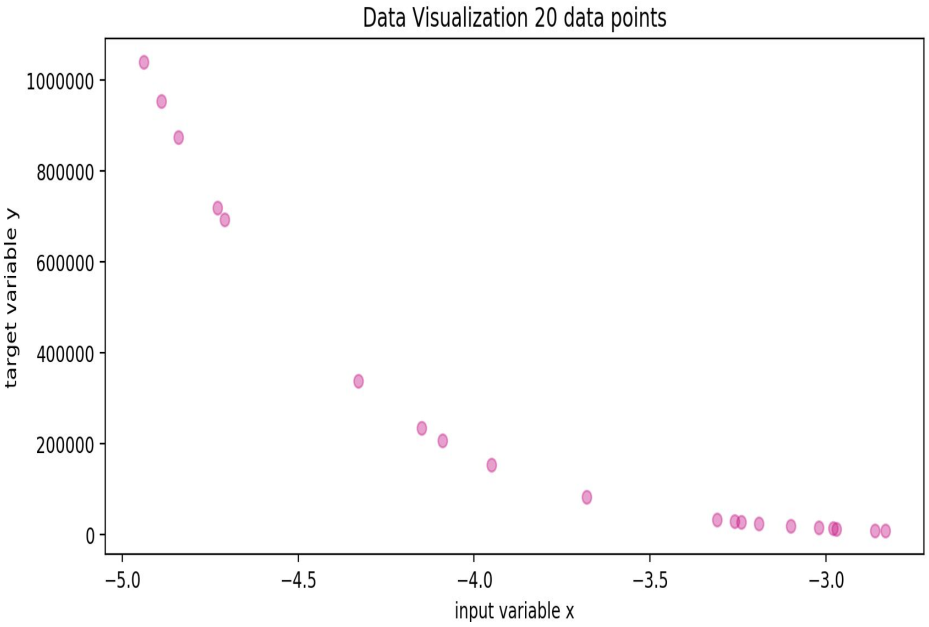
x	y
-4.94	1039755.18
-4.89	953776.95
-4.84	874074.10
-4.73	718895.29
-4.71	693432.09
-4.33	338012.90
-4.15	234701.23
-4.09	207027.09
-3.95	153242.63
-3.68	82835.16

Data Visualization



100 data points

X range: -4.94 to 4.92
Y range: 1.21 to 2254025.74



20 data points

X range: -4.94 to -2.83
Y range: 7990.54 to 1039755.18

Process Overview

Polynomial Degree Range : 0 to 10

Lambda Range ƛ: 1e-3 to 1e2

Cross Validation constant (k) : 5

1. For each degree in degree range do:
   a. Fit model
   b. Calculate performance metrics over k-folds (and alpha, if regularized)
2. Select the degree(and alpha, if regularized) which yields best goodness of fit

PERFORMANCE METRICS

1. Mean Absolute Error

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

MAE	= mean absolute error
y_i	= prediction
x_i	= true value
n	= total number of data points

2. Mean Square Error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE	= mean squared error
n	= number of data points
Y_i	= observed values
$\hat{Y}_i$	= predicted values

Large differences between actual and predicted are punished more in MSE than in MAE.

3. Goodness of Fit (R^2)

$$R^2 = 1 - \frac{RSS}{TSS}$$

R^2	= coefficient of determination
RSS	= sum of squares of residuals
TSS	= total sum of squares

$0 \leq R^2 \leq 1$[Generally]

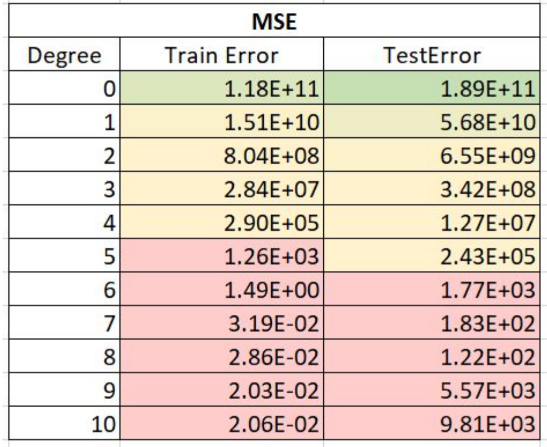
R^2 is used to quantify the amount of variability in the data that is explained by the model.

Polynomial Regression

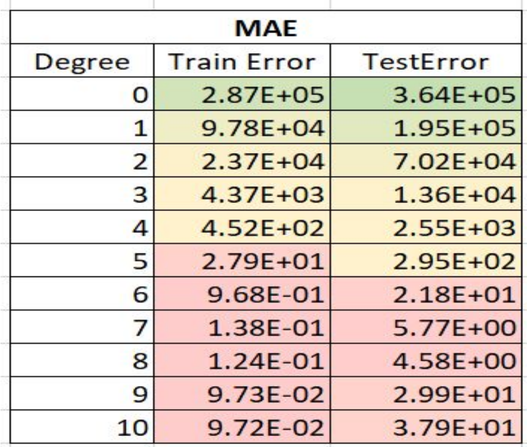
20 Data Points

Data with top 20 data points

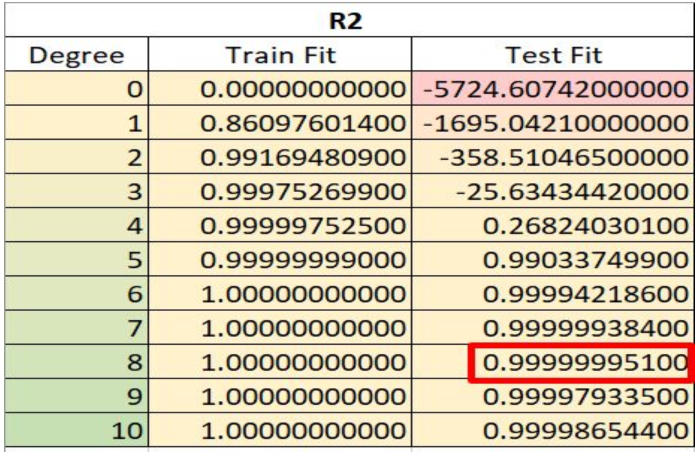
Errors and Goodness of Fit



MSE		
Degree	Train Error	TestError
0	1.18E+11	1.89E+11
1	1.51E+10	5.68E+10
2	8.04E+08	6.55E+09
3	2.84E+07	3.42E+08
4	2.90E+05	1.27E+07
5	1.26E+03	2.43E+05
6	1.49E+00	1.77E+03
7	3.19E-02	1.83E+02
8	2.86E-02	1.22E+02
9	2.03E-02	5.57E+03
10	2.06E-02	9.81E+03



MAE		
Degree	Train Error	TestError
0	2.87E+05	3.64E+05
1	9.78E+04	1.95E+05
2	2.37E+04	7.02E+04
3	4.37E+03	1.36E+04
4	4.52E+02	2.55E+03
5	2.79E+01	2.95E+02
6	9.68E-01	2.18E+01
7	1.38E-01	5.77E+00
8	1.24E-01	4.58E+00
9	9.73E-02	2.99E+01
10	9.72E-02	3.79E+01

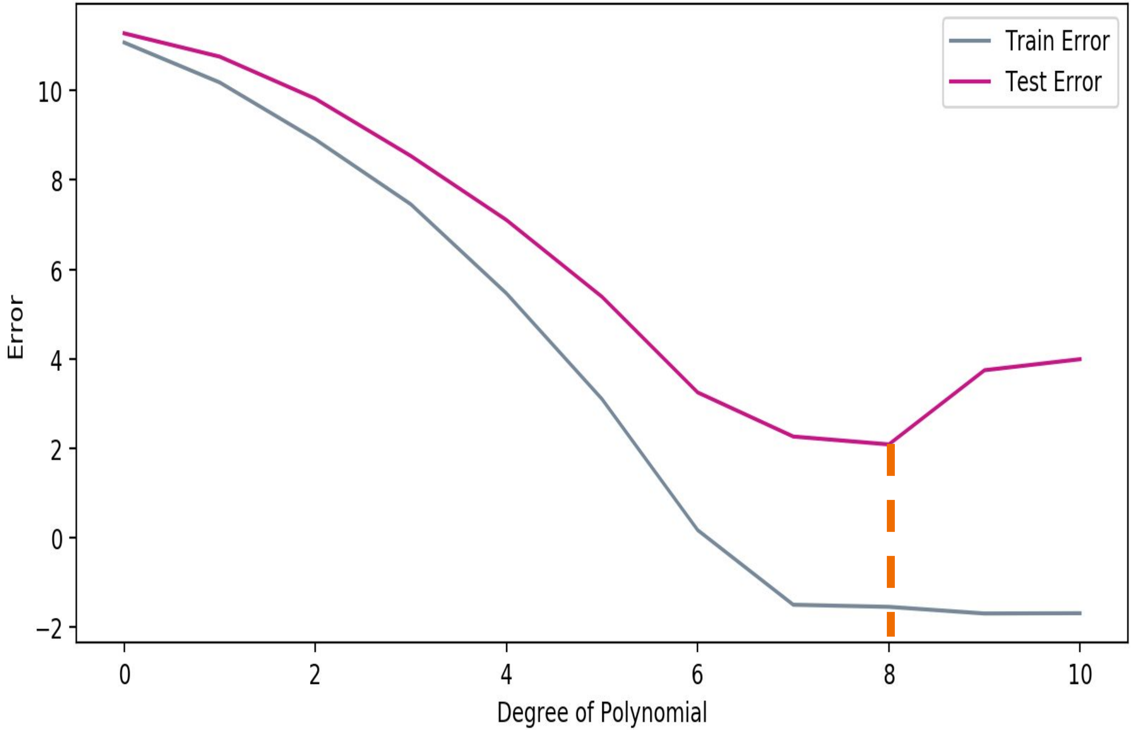


R2		
Degree	Train Fit	Test Fit
0	0.00000000000	-5724.60742000000
1	0.86097601400	-1695.04210000000
2	0.99169480900	-358.51046500000
3	0.99975269900	-25.63434420000
4	0.99999752500	0.26824030100
5	0.99999999000	0.99033749900
6	1.00000000000	0.99994218600
7	1.00000000000	0.99999938400
8	1.00000000000	0.99999995100
9	1.00000000000	0.99997933500
10	1.00000000000	0.99998654400

As is seen for degree 8, R2 value is maximum we choose the degree 8 as the degree of the model

Optimal degree = 8

FIT OF THE MODEL



- **Degree = 0 :** Train & Test error both are very high : UNDERFIT
- **Degree > 8 :** Train error is low & Test error is high : OVERFIT
- **Degree = 8 :** both Train & Test are stable and minimum : BEST-FIT

This correlates with the goodness fit measure .

FINAL POLYNOMIAL REGRESSION MODEL

Polynomial Coefficients : 0.00000000e+00 -6.25701553e+05 -6.02581129e+05 -3.29180875e+05 -1.11568325e+05 -2.40209506e+04 -3.20479316e+03 -2.34696399e+02 -3.51654246e+00

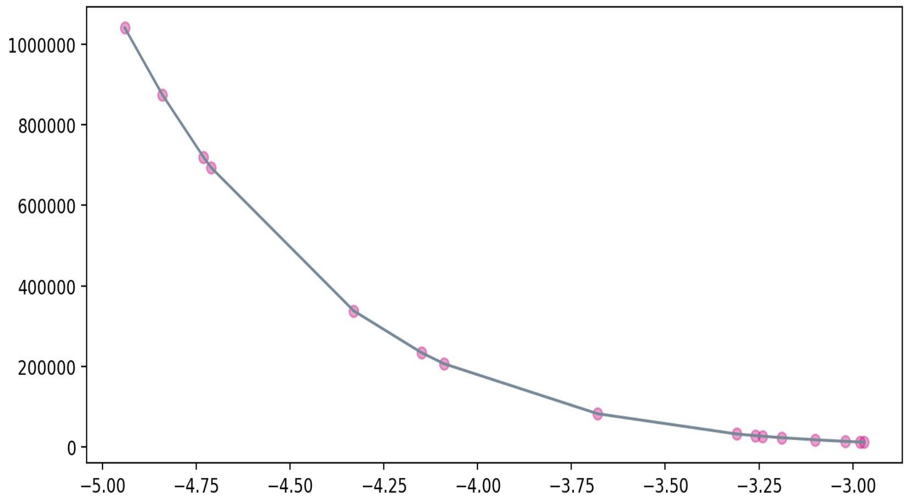
Intercept - -282167.77394048

Variance : .04421229720170674

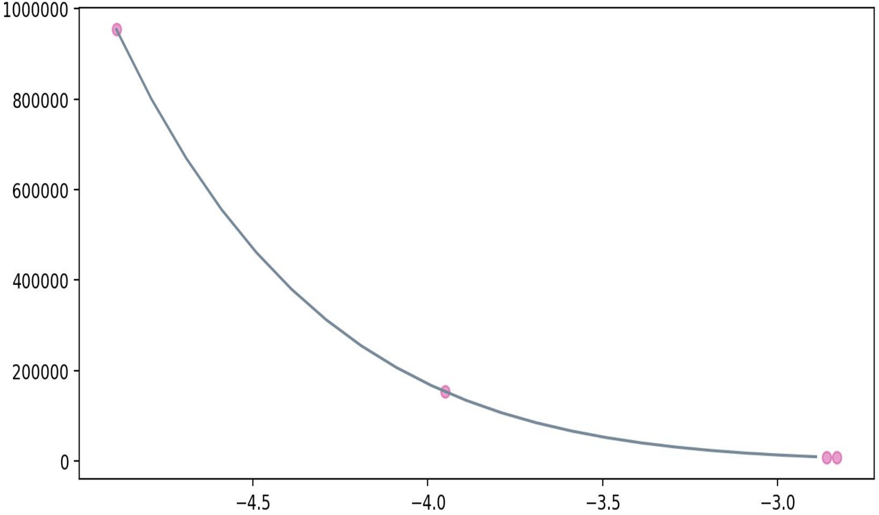
Optimal degree: 8

FITTING THE MODEL TO DATA

TRAINING SET



TEST SET

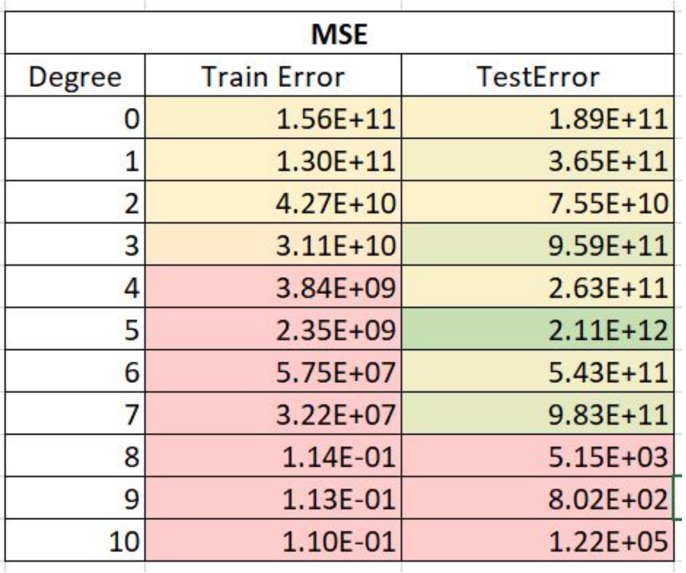


TRAIN TEST SPLIT = 80:20 (Non-Random, No Shuffle)

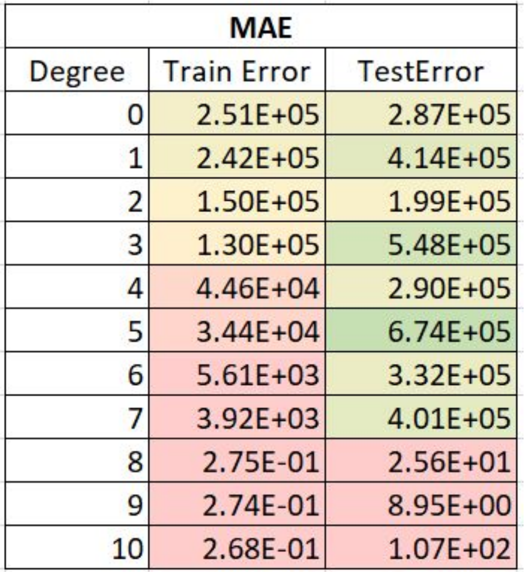
100 Data Points

Complete data

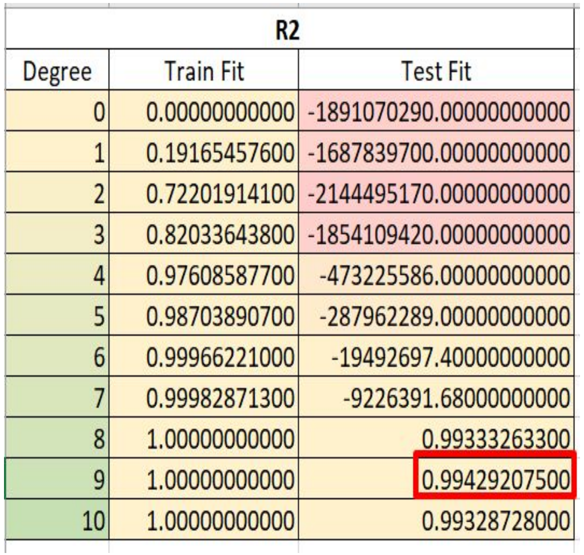
Errors and Goodness of Fit



MSE		
Degree	Train Error	TestError
0	1.56E+11	1.89E+11
1	1.30E+11	3.65E+11
2	4.27E+10	7.55E+10
3	3.11E+10	9.59E+11
4	3.84E+09	2.63E+11
5	2.35E+09	2.11E+12
6	5.75E+07	5.43E+11
7	3.22E+07	9.83E+11
8	1.14E-01	5.15E+03
9	1.13E-01	8.02E+02
10	1.10E-01	1.22E+05



MAE		
Degree	Train Error	TestError
0	2.51E+05	2.87E+05
1	2.42E+05	4.14E+05
2	1.50E+05	1.99E+05
3	1.30E+05	5.48E+05
4	4.46E+04	2.90E+05
5	3.44E+04	6.74E+05
6	5.61E+03	3.32E+05
7	3.92E+03	4.01E+05
8	2.75E-01	2.56E+01
9	2.74E-01	8.95E+00
10	2.68E-01	1.07E+02

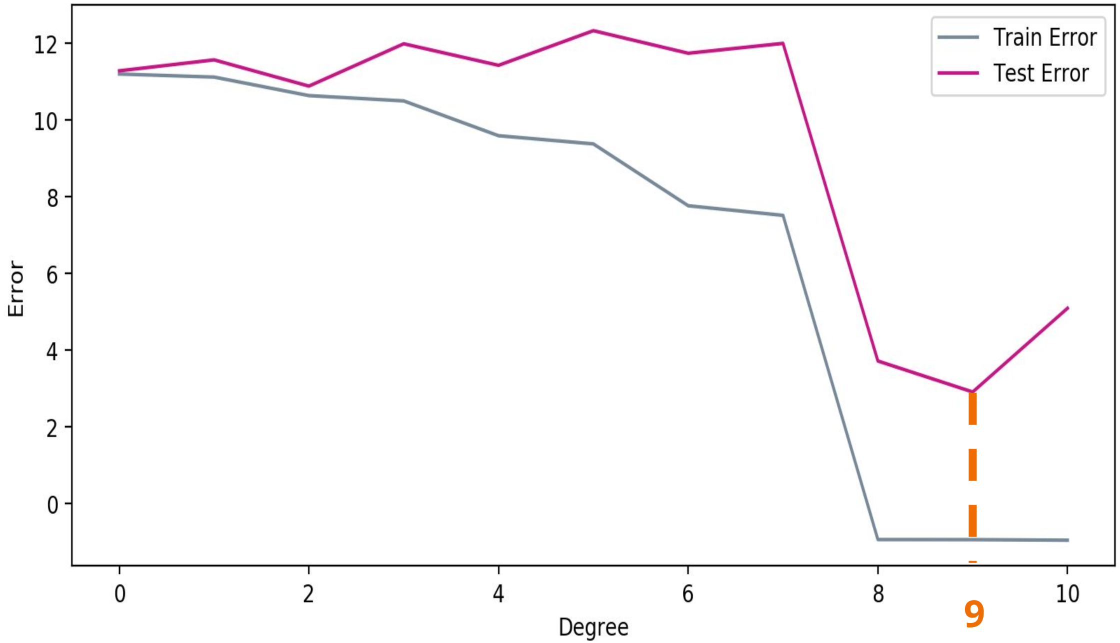


R2		
Degree	Train Fit	Test Fit
0	0.00000000000	-1891070290.00000000000
1	0.19165457600	-1687839700.00000000000
2	0.72201914100	-2144495170.00000000000
3	0.82033643800	-1854109420.00000000000
4	0.97608587700	-473225586.00000000000
5	0.98703890700	-287962289.00000000000
6	0.99966221000	-19492697.40000000000
7	0.99982871300	-9226391.68000000000
8	1.00000000000	0.99333263300
9	1.00000000000	0.99429207500
10	1.00000000000	0.99328728000

As is seen for degree 9, R2 value is maximum we choose the degree 9 as the degree of the model

Optimal degree = 9

FIT OF THE MODEL



- **Degree = 0 :** Train & Test error both are very high : UNDERFIT
- **Degree > 9 :** Train error is low & Test error is high : OVERFIT
- **Degree = 9 :** both Train & Test are stable and minimum : BEST-FIT

This correlates with the goodness fit measure .

FINAL POLYNOMIAL REGRESSION MODEL

Polynomial Coefficients : 0.6.93942057 8.63781465 7.07980821 5.50764703 5.80178037 5.69965776 8.69995532 4.50000523

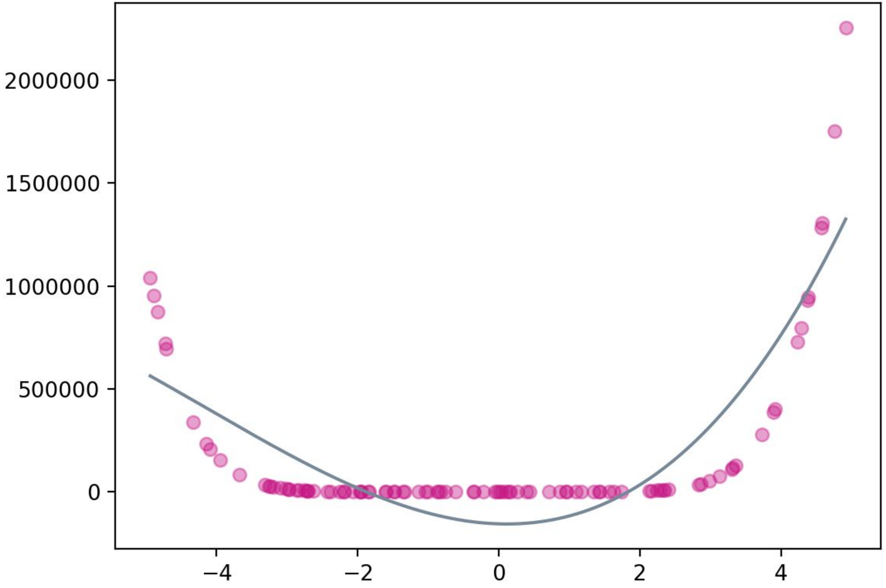
Intercept - 7.23321344

Variance : .1152058975243086

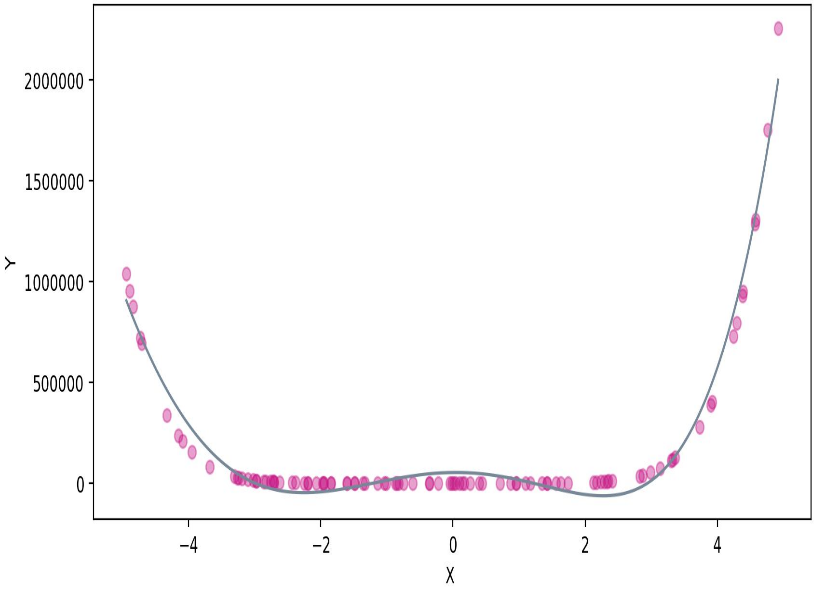
Optimal degree: 9

Model with less than optimal degree

DEGREE = 3

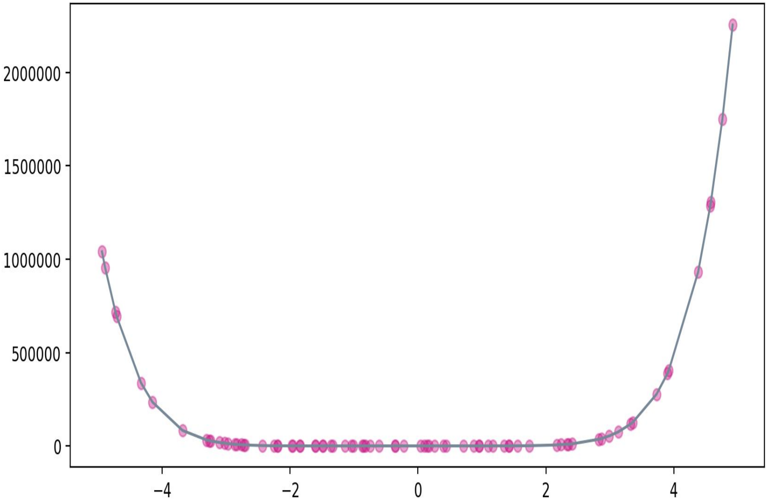


DEGREE = 5

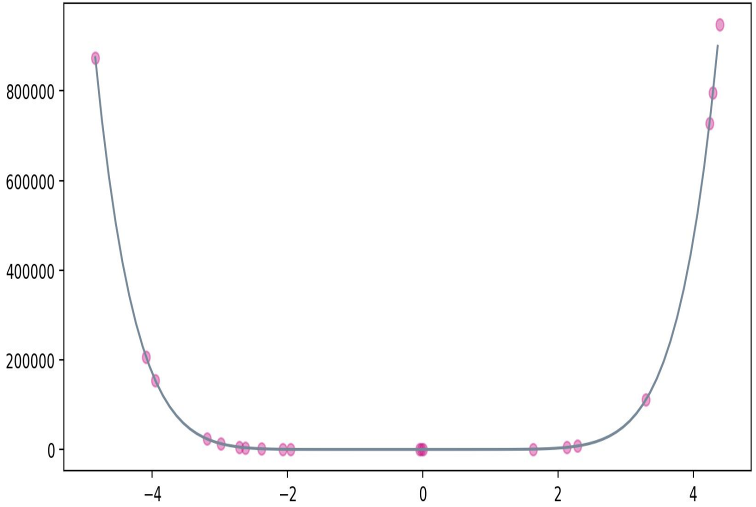


FITTING THE MODEL TO DATA

TRAINING SET



TEST SET



TRAIN TEST SPLIT = 80:20 (Non-Random)

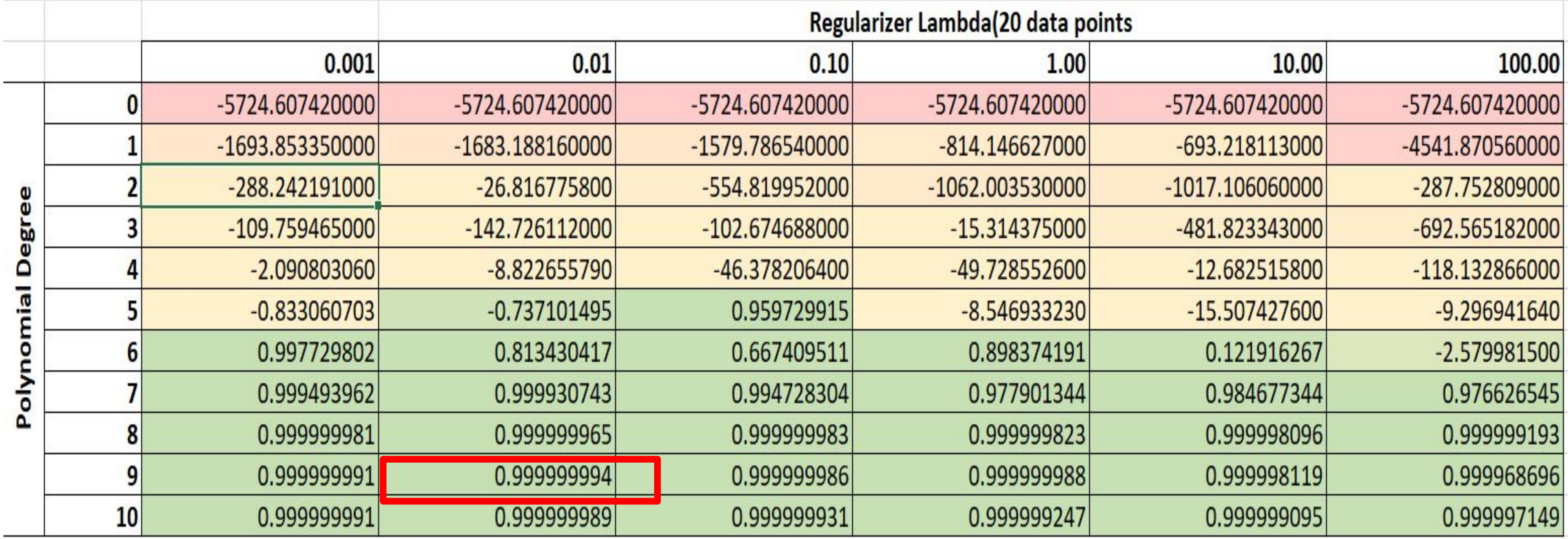
Regularized Regression

(With a regularization parameter lambda)

20 Data Points

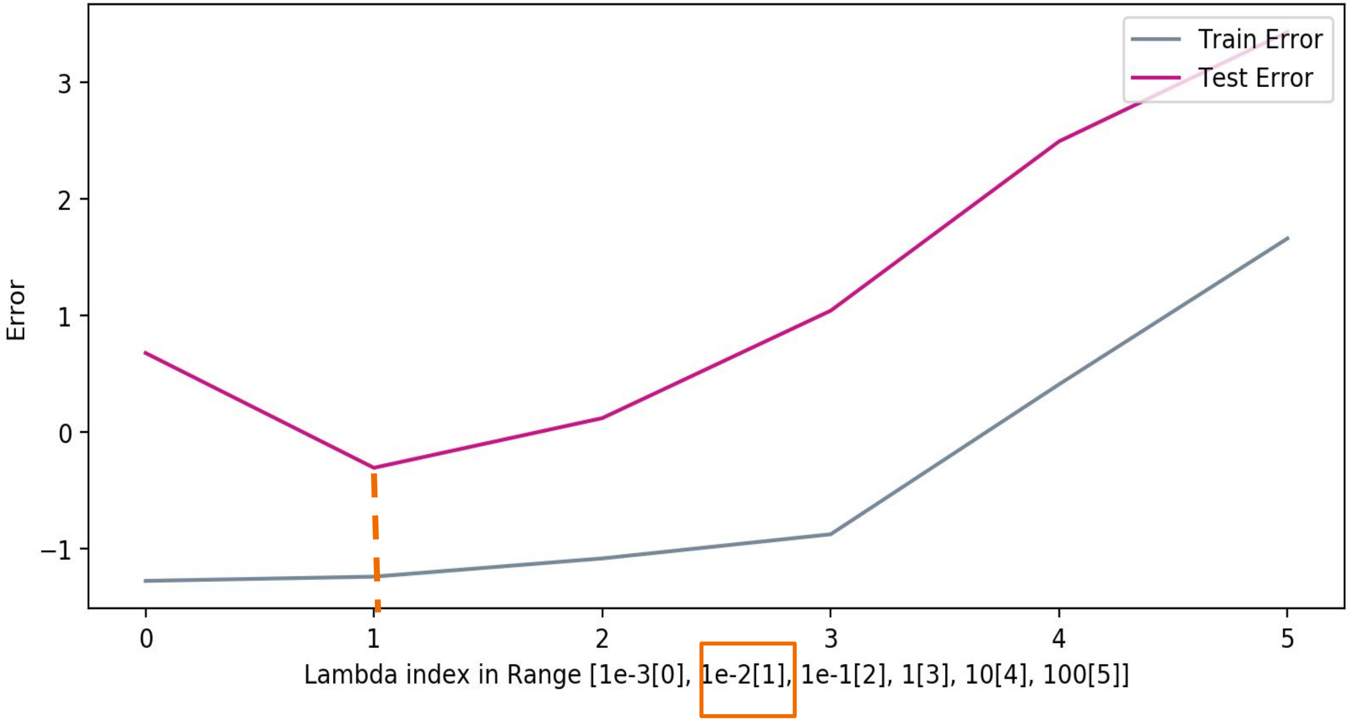
Data with top 20 data points

GOODNESS OF FIT(R^2)



				Regularizer Lambda(20 data points			
		0.001	0.01	0.10	1.00	10.00	100.00
Polynomial Degree	0	-5724.607420000	-5724.607420000	-5724.607420000	-5724.607420000	-5724.607420000	-5724.607420000
	1	-1693.853350000	-1683.188160000	-1579.786540000	-814.146627000	-693.218113000	-4541.870560000
	2	-288.242191000	-26.816775800	-554.819952000	-1062.003530000	-1017.106060000	-287.752809000
	3	-109.759465000	-142.726112000	-102.674688000	-15.314375000	-481.823343000	-692.565182000
	4	-2.090803060	-8.822655790	-46.378206400	-49.728552600	-12.682515800	-118.132866000
	5	-0.833060703	-0.737101495	0.959729915	-8.546933230	-15.507427600	-9.296941640
	6	0.997729802	0.813430417	0.667409511	0.898374191	0.121916267	-2.579981500
	7	0.999493962	0.999930743	0.994728304	0.977901344	0.984677344	0.976626545
	8	0.999999981	0.999999965	0.999999983	0.999999823	0.999998096	0.999999193
	9	0.999999991	0.999999994	0.999999986	0.999999988	0.999998119	0.999968696
	10	0.999999991	0.999999989	0.999999931	0.999999247	0.999999095	0.999997149

FIT OF THE MODEL



ƛ = 100 : Train and Test error both are very high : UNDERFIT

ƛ = 0.01 : Train and Test error are low: BEST-FIT.

ƛ = 0.001 : Train error is low and Test error is high BEST-FIT.

This correlates with the goodness fit measure

FINAL POLYNOMIAL REGRESSION MODEL

Polynomial Coefficients : 0.00000000e+00 1.49778155e-01 -4.74272849e-01 8.62626551e-01 -6.74076261e-01 -1.03036688e+00 1.91346890e+00 7.65801496e+00 4.35992945e+00 -7.38110718e-03

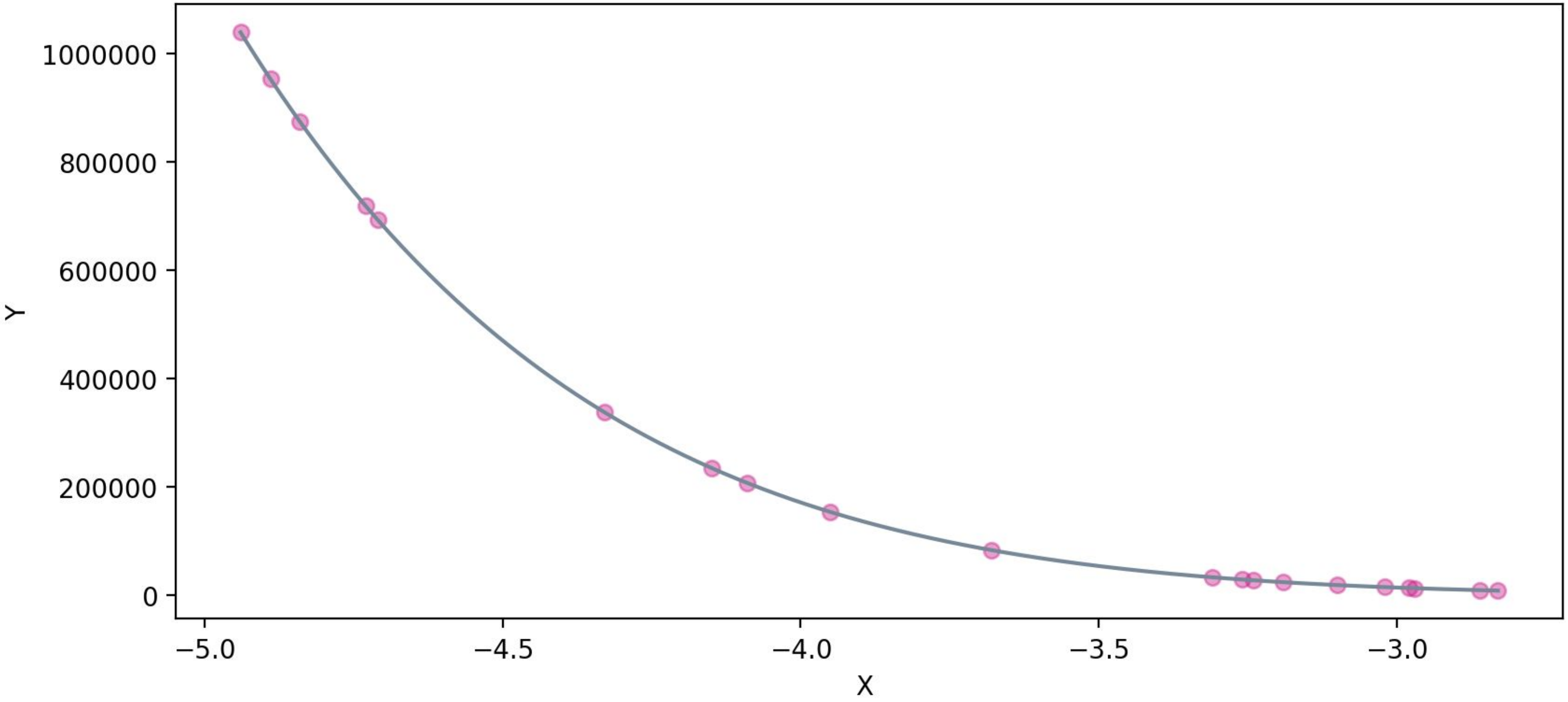
Intercept : -3.38420919

Variance : 0.061805574040417345

Optimal degree: 9

Optimal lambda: 0.01

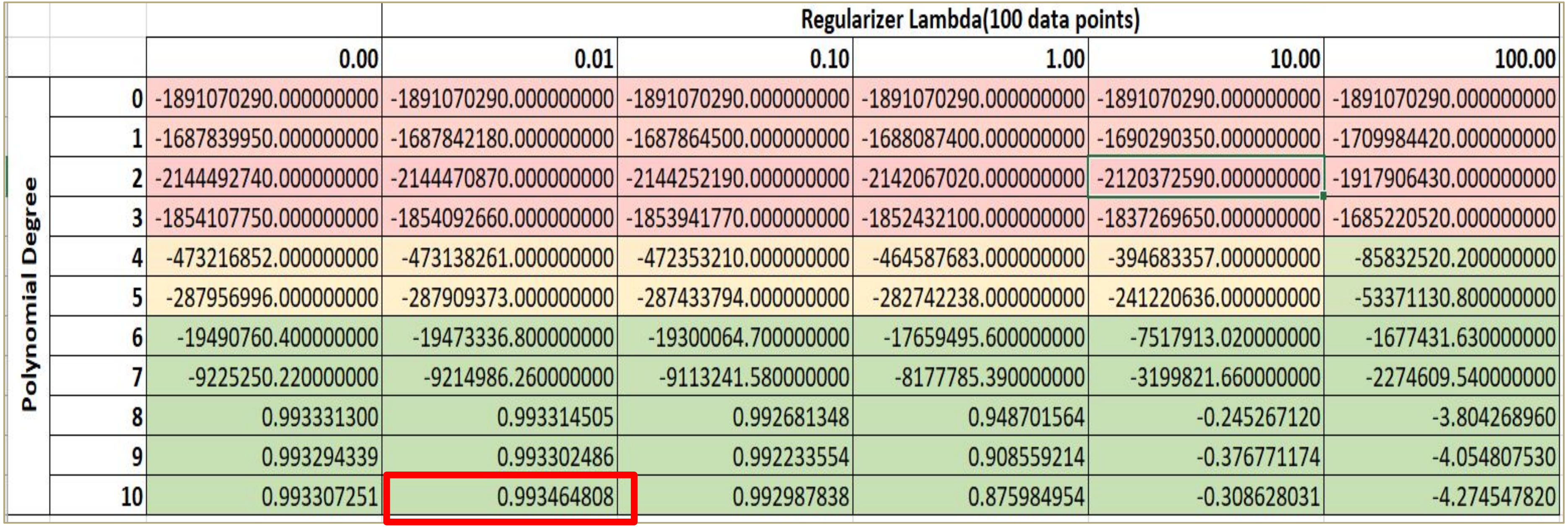
FITTING DATA TO MODEL



100 Data Points

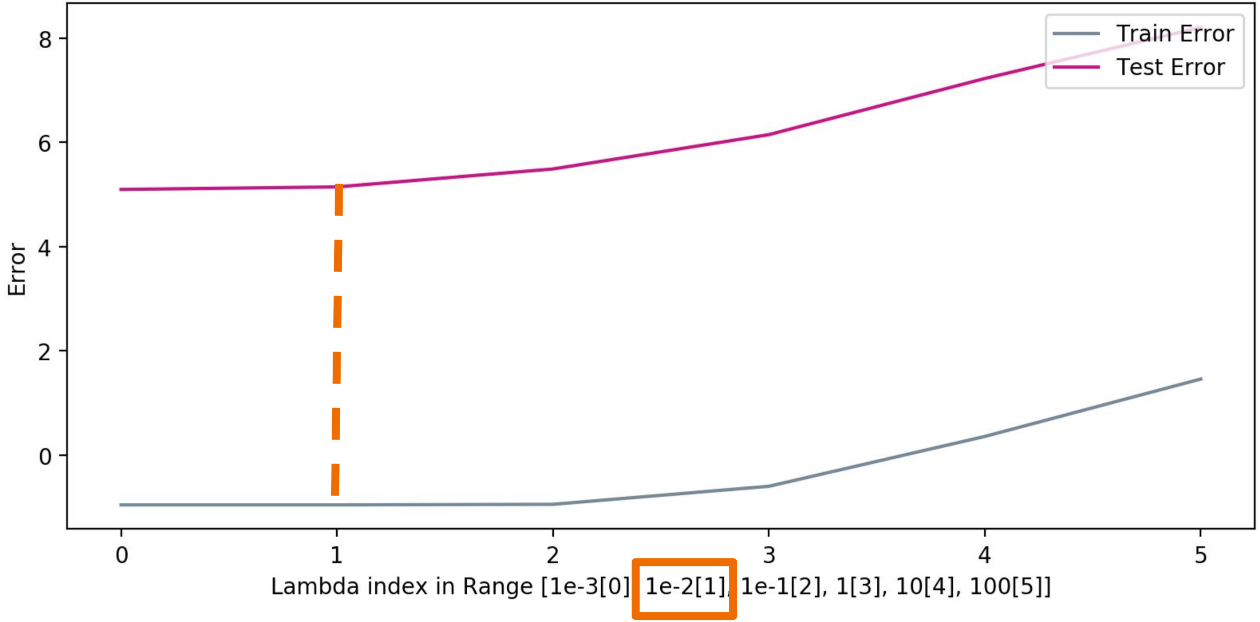
Complete data

GOODNESS OF FIT(R^2)



		Regularizer Lambda(100 data points)					
		0.00	0.01	0.10	1.00	10.00	100.00
Polynomial Degree	0	-1891070290.000000000	-1891070290.000000000	-1891070290.000000000	-1891070290.000000000	-1891070290.000000000	-1891070290.000000000
	1	-1687839950.000000000	-1687842180.000000000	-1687864500.000000000	-1688087400.000000000	-1690290350.000000000	-1709984420.000000000
	2	-2144492740.000000000	-2144470870.000000000	-2144252190.000000000	-2142067020.000000000	-2120372590.000000000	-1917906430.000000000
	3	-1854107750.000000000	-1854092660.000000000	-1853941770.000000000	-1852432100.000000000	-1837269650.000000000	-1685220520.000000000
	4	-473216852.000000000	-473138261.000000000	-472353210.000000000	-464587683.000000000	-394683357.000000000	-85832520.200000000
	5	-287956996.000000000	-287909373.000000000	-287433794.000000000	-282742238.000000000	-241220636.000000000	-53371130.800000000
	6	-19490760.400000000	-19473336.800000000	-19300064.700000000	-17659495.600000000	-7517913.020000000	-1677431.630000000
	7	-9225250.220000000	-9214986.260000000	-9113241.580000000	-8177785.390000000	-3199821.660000000	-2274609.540000000
	8	0.993331300	0.993314505	0.992681348	0.948701564	-0.245267120	-3.804268960
	9	0.993294339	0.993302486	0.992233554	0.908559214	-0.376771174	-4.054807530
	10	0.993307251	0.993464808	0.992987838	0.875984954	-0.308628031	-4.274547820

FIT OF THE MODEL



ƛ = 100 : Train and Test error both are very high : UNDERFIT

ƛ = 0.01 : both are stable and minimum : BEST-FIT after this test and train error starts increasing.

This correlates with the goodness fit measure

FINAL POLYNOMIAL REGRESSION MODEL

Polynomial Coefficients : 0.00000000e+00 6.93550528e+00
8.78616548e+00 7.08446796e+00 5.45249467e+00 5.80063154e+00
5.70665822e+00 8.70003833e+004.49965071e+00 -1.78521148e-06
6.17856419e-06

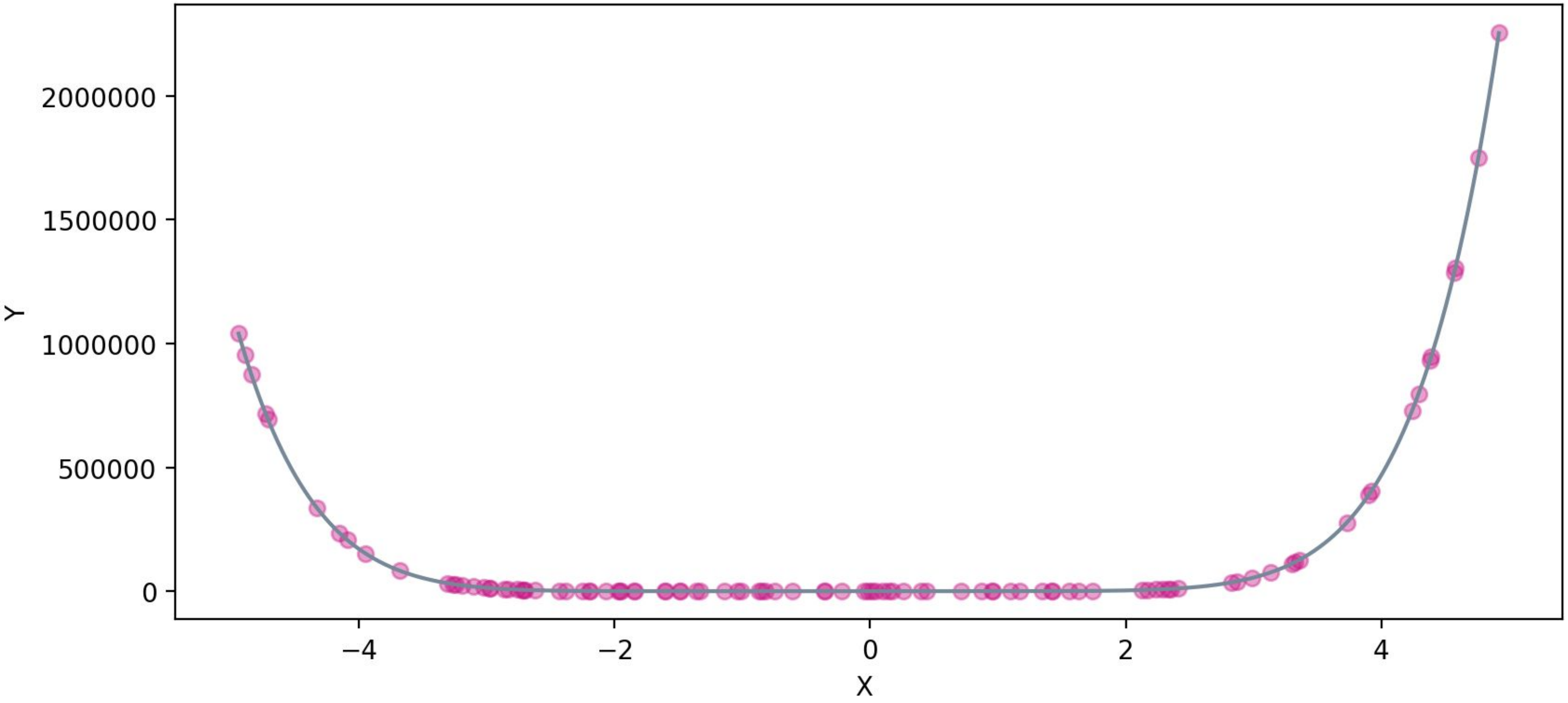
Intercept : 7.17396127

Variance : 0.1108132442535524

Optimal degree: 10

Optimal lambda: 0.01

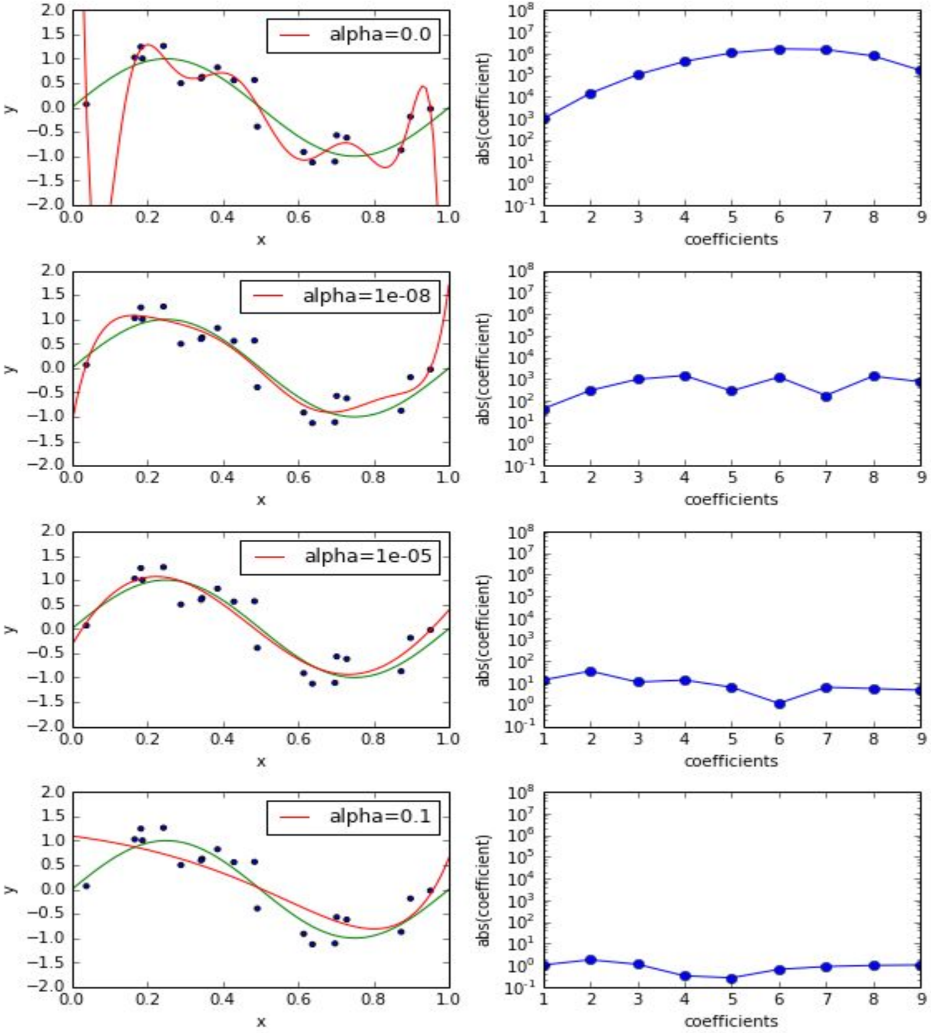
FITTING DATA TO MODEL



SIGNIFICANCE OF Lambda

1. The higher the value of lambda,
   the lower would be the magnitude of the polynomial coefficients. (under-fitting)

2. The lower the value of lambda (approximately zero), the higher the value of coefficients.(over-fitting)
   In this case, the analysis approaches the linear regression model.



Observation

For regression without Regularization and 20 data points the degree was 8 and for 100 data points optimal degree was 9 and variance increased and thus the fit.

For regression with Regularization and 20 data points the degree was 9 and for 100 data points optimal degree was 10 and variance increased and thus the fit.

Therefore, as we increase the data points the model learns better.

Genomic Sequence Analysis

Dataset Description

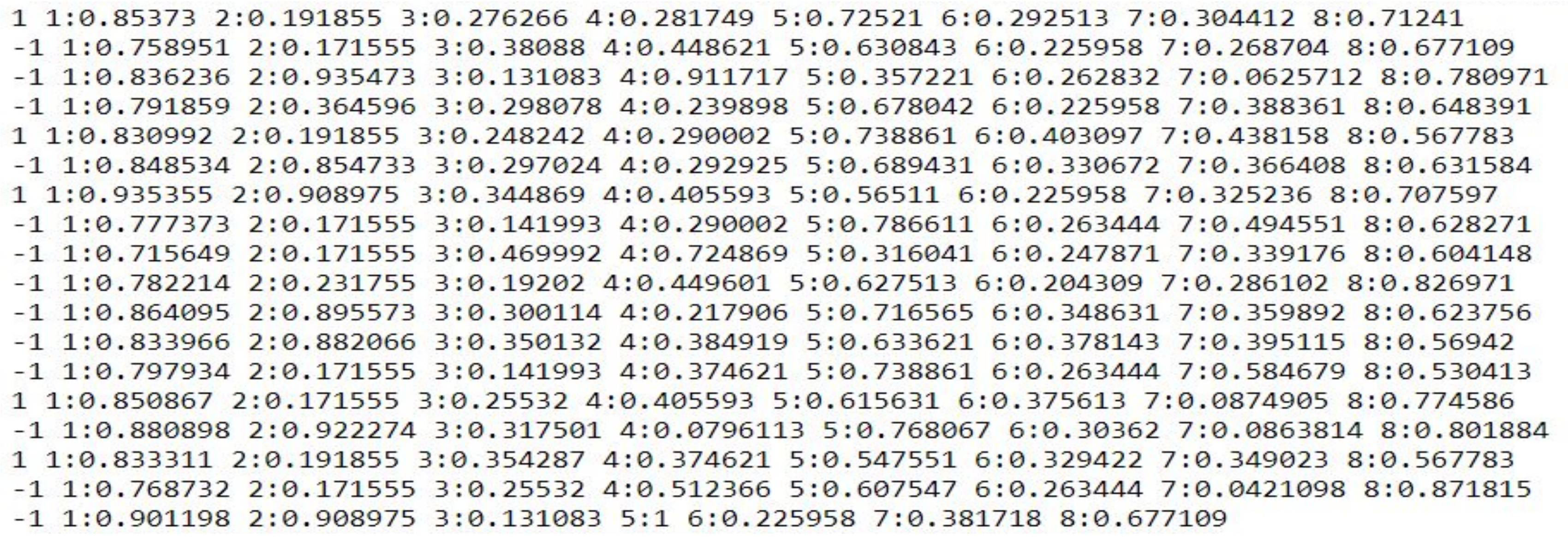
Non-coding ribonucleic acids (ncRNA) are believed to have many roles in a cell, many of which remain to be discovered. However, it is difficult to detect ncRNAs using biochemical screening methods. Recent studies have shown that computational methods can accurately detect ncRNAs, which can be treated as supervised classification. To perform the classification, an 8-dimensional feature vector is used as input to a classifier, including the length of genomic sequence and nucleotide frequencies:

- A feature value computed by the Dynalign algorithm http://www.ncbi.nlm.nih.gov/pubmed/11902836
- Length of shorter sequence
- `A' frequencies of sequence 1
- `U' frequencies of sequence 1
- `C' frequencies of sequence 1
- `A' frequencies of sequence 2
- `U' frequencies of sequence 2
- `C' frequencies of sequence 2

Number of training points = 2000

Number of testing points = 1001

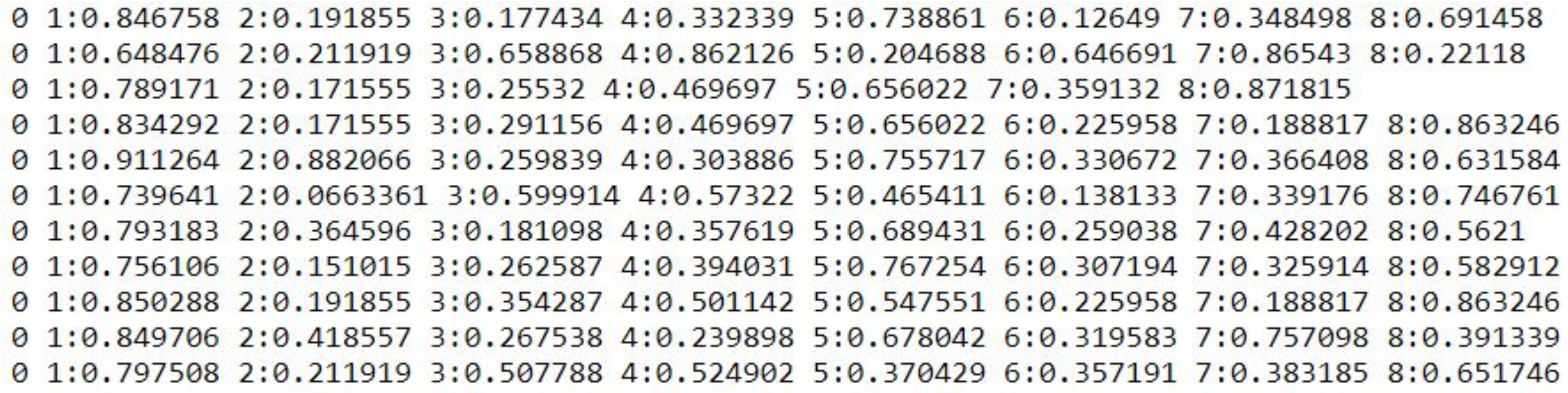
Training Data (2000)



```
1 1:0.85373 2:0.191855 3:0.276266 4:0.281749 5:0.72521 6:0.292513 7:0.304412 8:0.71241
-1 1:0.758951 2:0.171555 3:0.38088 4:0.448621 5:0.630843 6:0.225958 7:0.268704 8:0.677109
-1 1:0.836236 2:0.935473 3:0.131083 4:0.911717 5:0.357221 6:0.262832 7:0.0625712 8:0.780971
-1 1:0.791859 2:0.364596 3:0.298078 4:0.239898 5:0.678042 6:0.225958 7:0.388361 8:0.648391
1 1:0.830992 2:0.191855 3:0.248242 4:0.290002 5:0.738861 6:0.403097 7:0.438158 8:0.567783
-1 1:0.848534 2:0.854733 3:0.297024 4:0.292925 5:0.689431 6:0.330672 7:0.366408 8:0.631584
1 1:0.935355 2:0.908975 3:0.344869 4:0.405593 5:0.56511 6:0.225958 7:0.325236 8:0.707597
-1 1:0.777373 2:0.171555 3:0.141993 4:0.290002 5:0.786611 6:0.263444 7:0.494551 8:0.628271
-1 1:0.715649 2:0.171555 3:0.469992 4:0.724869 5:0.316041 6:0.247871 7:0.339176 8:0.604148
-1 1:0.782214 2:0.231755 3:0.19202 4:0.449601 5:0.627513 6:0.204309 7:0.286102 8:0.826971
-1 1:0.864095 2:0.895573 3:0.300114 4:0.217906 5:0.716565 6:0.348631 7:0.359892 8:0.623756
-1 1:0.833966 2:0.882066 3:0.350132 4:0.384919 5:0.633621 6:0.378143 7:0.395115 8:0.56942
-1 1:0.797934 2:0.171555 3:0.141993 4:0.374621 5:0.738861 6:0.263444 7:0.584679 8:0.530413
1 1:0.850867 2:0.171555 3:0.25532 4:0.405593 5:0.615631 6:0.375613 7:0.0874905 8:0.774586
-1 1:0.880898 2:0.922274 3:0.317501 4:0.0796113 5:0.768067 6:0.30362 7:0.0863814 8:0.801884
1 1:0.833311 2:0.191855 3:0.354287 4:0.374621 5:0.547551 6:0.329422 7:0.349023 8:0.567783
-1 1:0.768732 2:0.171555 3:0.25532 4:0.512366 5:0.607547 6:0.263444 7:0.0421098 8:0.871815
-1 1:0.901198 2:0.908975 3:0.131083 5:1 6:0.225958 7:0.381718 8:0.677109
```

Class labels: 1 and -1 Features: 1,2,3,4,5,6,7 and 8

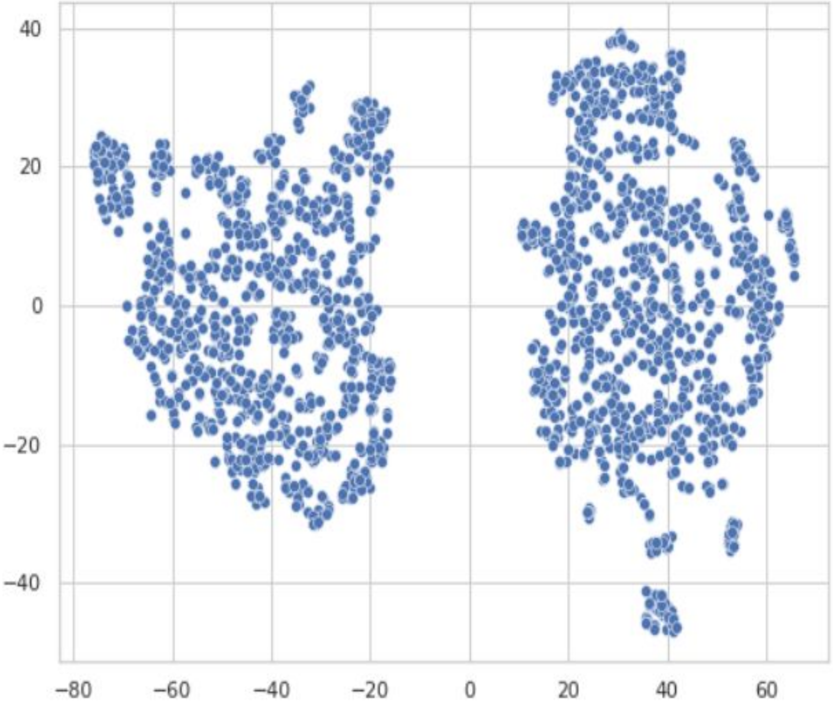
Test Data (1001)

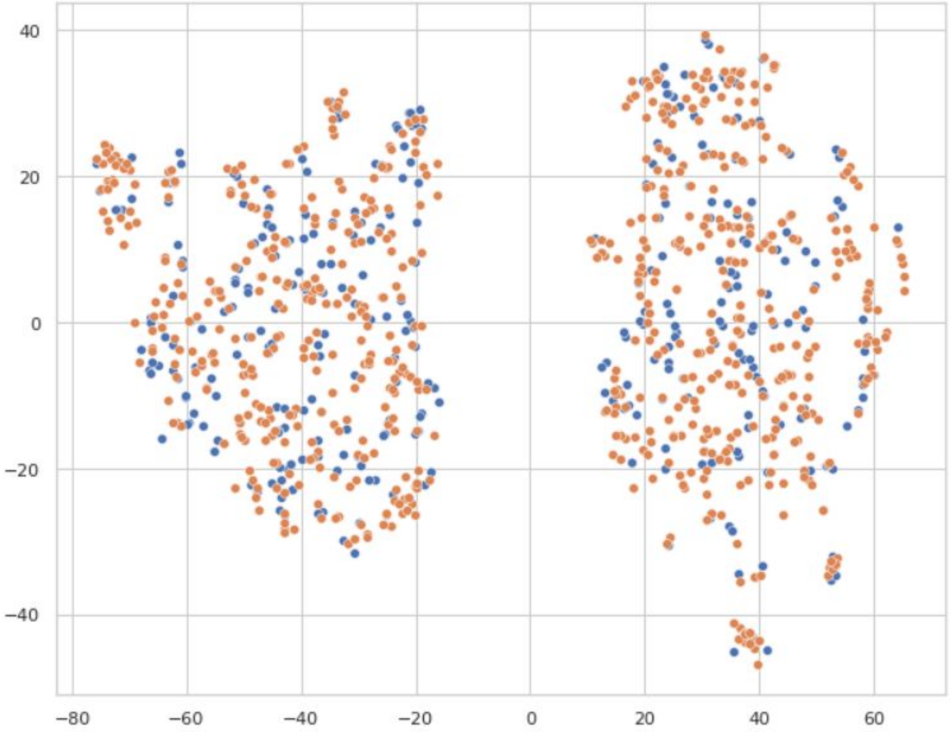


```
0 1:0.846758 2:0.191855 3:0.177434 4:0.332339 5:0.738861 6:0.12649 7:0.348498 8:0.691458
0 1:0.648476 2:0.211919 3:0.658868 4:0.862126 5:0.204688 6:0.646691 7:0.86543 8:0.22118
0 1:0.789171 2:0.171555 3:0.25532 4:0.469697 5:0.656022 7:0.359132 8:0.871815
0 1:0.834292 2:0.171555 3:0.291156 4:0.469697 5:0.656022 6:0.225958 7:0.188817 8:0.863246
0 1:0.911264 2:0.882066 3:0.259839 4:0.303886 5:0.755717 6:0.330672 7:0.366408 8:0.631584
0 1:0.739641 2:0.0663361 3:0.599914 4:0.57322 5:0.465411 6:0.138133 7:0.339176 8:0.746761
0 1:0.793183 2:0.364596 3:0.181098 4:0.357619 5:0.689431 6:0.259038 7:0.428202 8:0.5621
0 1:0.756106 2:0.151015 3:0.262587 4:0.394031 5:0.767254 6:0.307194 7:0.325914 8:0.582912
0 1:0.850288 2:0.191855 3:0.354287 4:0.501142 5:0.547551 6:0.225958 7:0.188817 8:0.863246
0 1:0.849706 2:0.418557 3:0.267538 4:0.239898 5:0.678042 6:0.319583 7:0.757098 8:0.391339
0 1:0.797508 2:0.211919 3:0.507788 4:0.524902 5:0.370429 6:0.357191 7:0.383185 8:0.651746
```

Class labels: 0 (To be predicted)   Features: 1,2,3,4,5,6,7 and 8

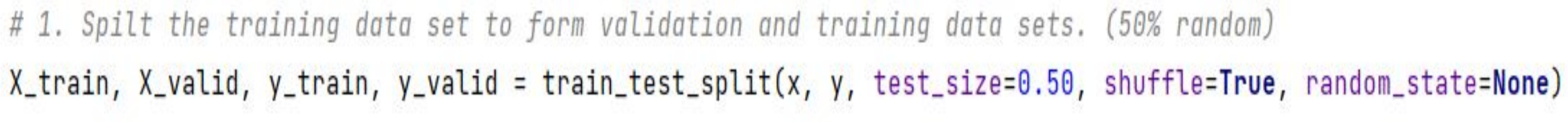
Data Visualization





Problem Statement 1: Classification using linear SVM

1. Spilt the training data set to form validation and training data sets.



```
# 1. Spilt the training data set to form validation and training data sets. (50% random)
X_train, X_valid, y_train, y_valid = train_test_split(x, y, test_size=0.50, shuffle=True, random_state=None)
```

2. Train a set of linear SVMs with different values of the regularisation parameter *C* using the training data set. For each value of *C*, train an SVM and use each trained SVM model to classify the validation data set.

   C_values = `[0.001, 0.01, 0.1, 1, 10, 100, 150, 200]`

   Maximum Accuracy : 95.5

   Optimal Value of C is : 0.1

Results and Accuracy Plot as parameter of C

Coefficient of the optimal Classifier

[[3.21353503 -1.36975459 0.47045147

0.39591541 -0.00936292 0.58258025

0.42093817 0.16810366]]

Intercept of the optimal Classifier

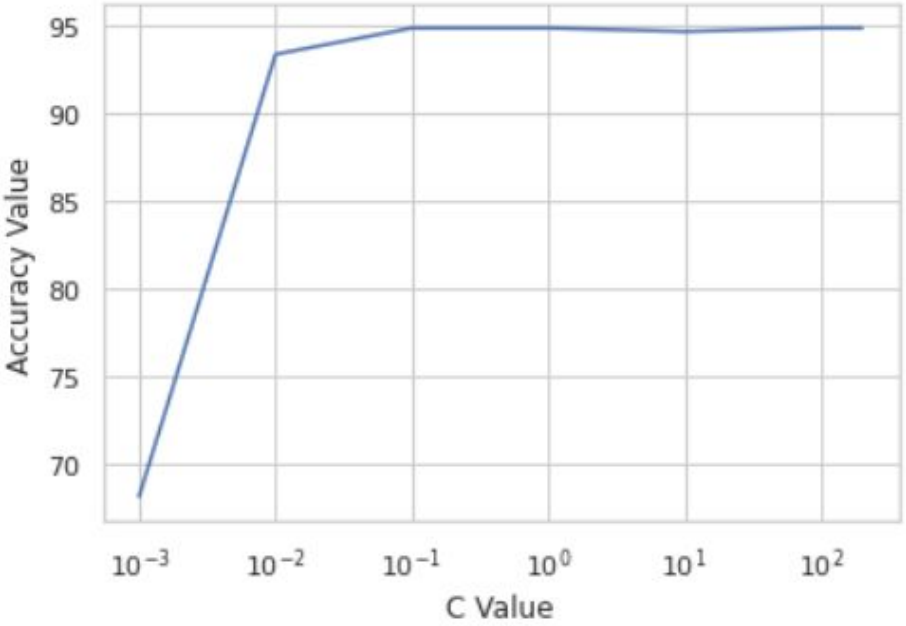
[-0.9601014]

confusion_matrix:

 [[662 23]

 [22 293]]

Accuracy Array: [68.1 93.3 94.8 94.8 94.6 94.8 94.8 94.8]



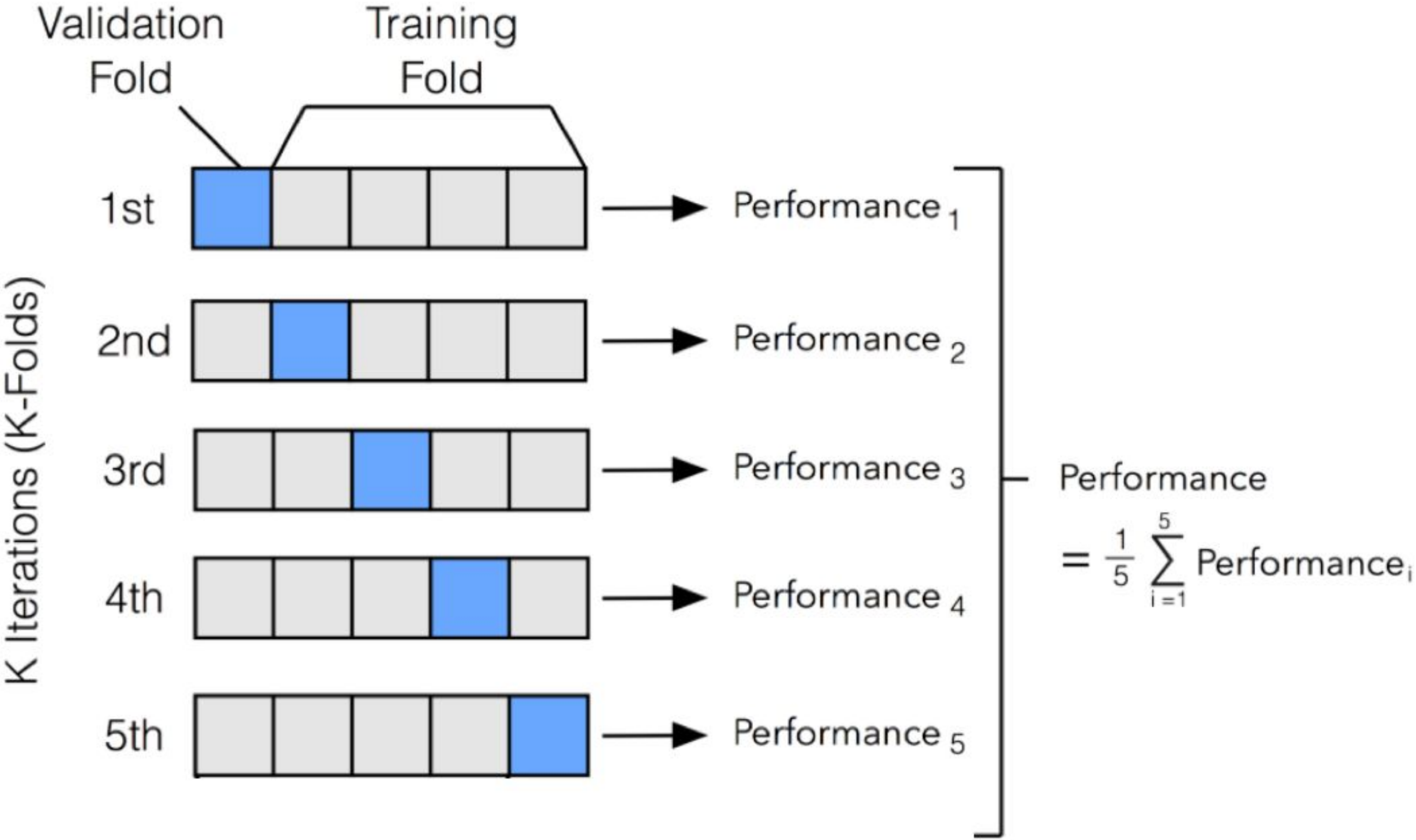
Problem Statement 2: Classification using Gaussian(RBF) SVM

Use 5-fold cross validation to choose the best *C* and *sigma* To do so, first randomly choose 50% of the training set as the cross validation set. Next, divide the cross validation set into 5 subsets of equal size. Each subset is in turn used to validate the classifier trained on the remaining 4 subsets. So you will have 5 trained SVMs and 5 validation subsets. The cross validation accuracy is average accuracy over the 5 validation subsets.

|----> Train (1000)

Data ------>|-----> Validation (1000) ----> Cross validation (5 - fold)

|----> Test (1001)


Validation Fold
Training Fold
K Iterations (K-Folds)
1st
2nd
3rd
4th
5th
Performance $_1$
Performance $_2$
Performance $_3$
Performance $_4$
Performance $_5$
Performance
$= \frac{1}{5}\sum_{i=1}^{5} \text{Performance}_i$

For both C and *sigma* try a number of different values and be sure to try all possible of pairs of values for *C* and *sigma*. Show a matrix of your cross validation results, where the entry *(i, j)* of the matrix corresponds to the classification accuracy on the cross validation set with *i*th value of *C* and *j* th value of *sigma*.

C_range = [`0.001, 0.01, 0.1, 1, 10, 100, 150, 200`**]**

gamma_range = [`0.001,0.01, 0.1, 1, 10, 100, 150, 200`**]**

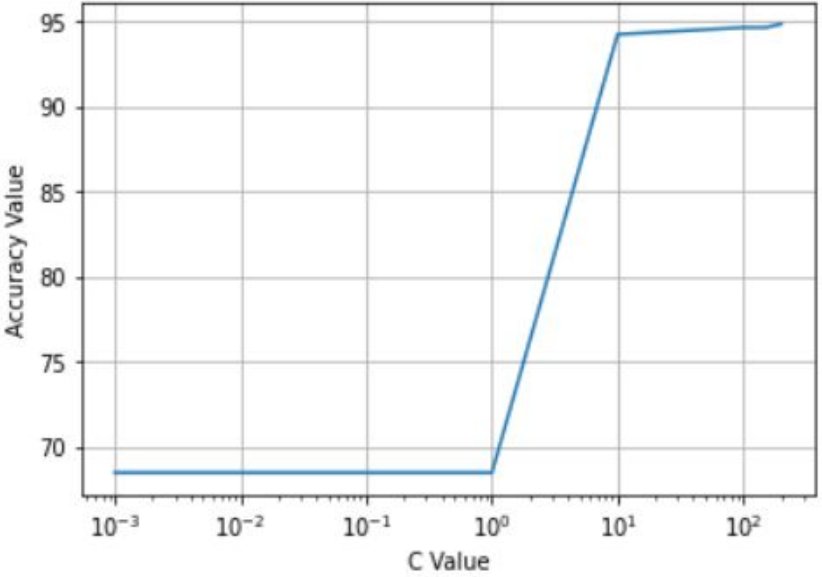
Maximum Accuracy : 94.90

Optimal Value of C : 100

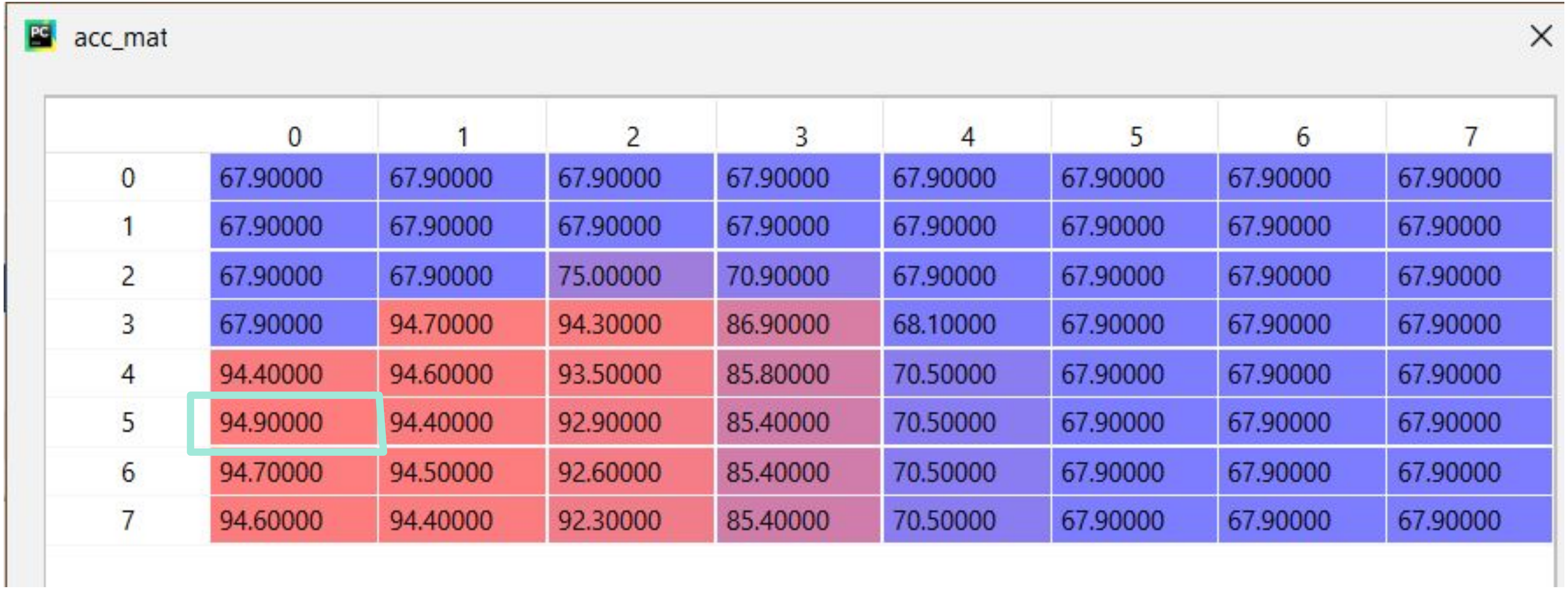
Optimal Value of Gamma : 0.001

Optimal Value of Sigma : 7.0710678118654755

Accuracy Matrix →



Accuracy Matrix



	0	1	2	3	4	5	6	7
0	67.90000	67.90000	67.90000	67.90000	67.90000	67.90000	67.90000	67.90000
1	67.90000	67.90000	67.90000	67.90000	67.90000	67.90000	67.90000	67.90000
2	67.90000	67.90000	75.00000	70.90000	67.90000	67.90000	67.90000	67.90000
3	67.90000	94.70000	94.30000	86.90000	68.10000	67.90000	67.90000	67.90000
4	94.40000	94.60000	93.50000	85.80000	70.50000	67.90000	67.90000	67.90000
5	94.90000	94.40000	92.90000	85.40000	70.50000	67.90000	67.90000	67.90000
6	94.70000	94.50000	92.60000	85.40000	70.50000	67.90000	67.90000	67.90000
7	94.60000	94.40000	92.30000	85.40000	70.50000	67.90000	67.90000	67.90000

Next, use the entire training set to train an SVM classifier with the best *C* and *\sigma* values determined via the cross validation procedure outlined above. Finally, use the trained SVM model to classify the test data set and write the results to a file using the same format as the training data set.

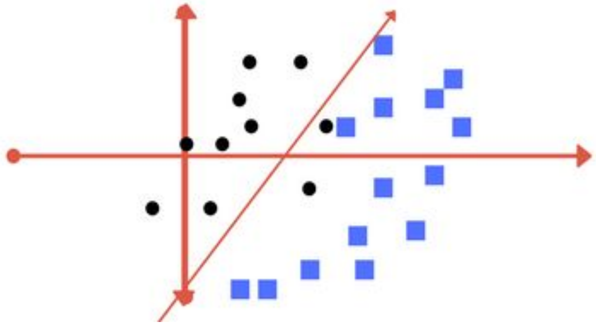
Submitting with the code!!

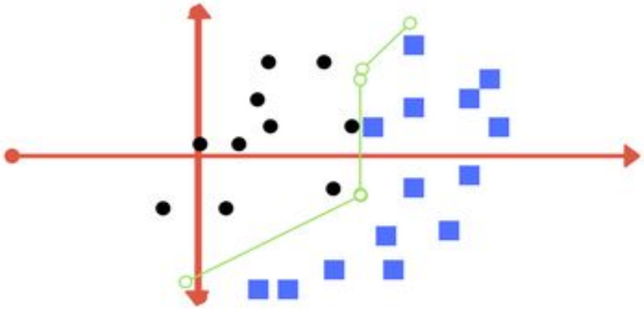
SIGNIFICANCE OF “C”

1. “C” is the regularisation parameter in python sklearn library.

2. Larger “C” leads overfitting and smaller “C” leads to underfitting.

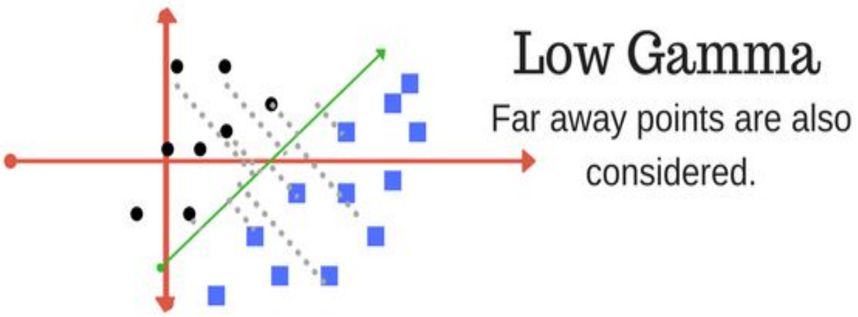
3. The images below are examples of two different regularization parameter. Left one has some misclassification due to lower regularization value. Higher value leads to results like right one.





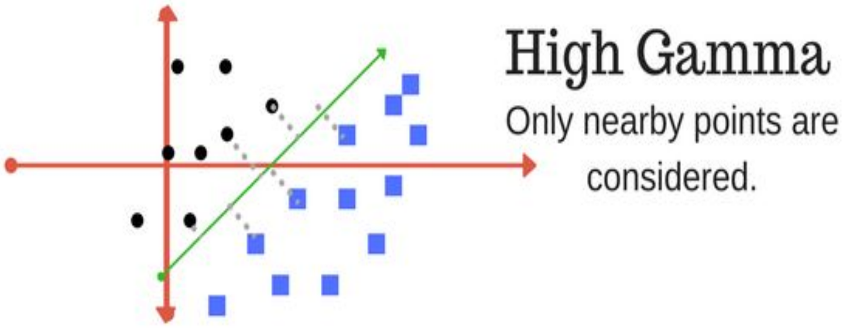
SIGNIFICANCE OF “Gamma/Sigma”

1. Lower “gamma” leads to consideration of the points that are far away from the possible decision boundary in the calculation of the actual decision boundary.
2. Higher “gamma” leads to consideration of the points that are nearer to the possible decision boundary in the calculation of the actual decision boundary.



Low Gamma

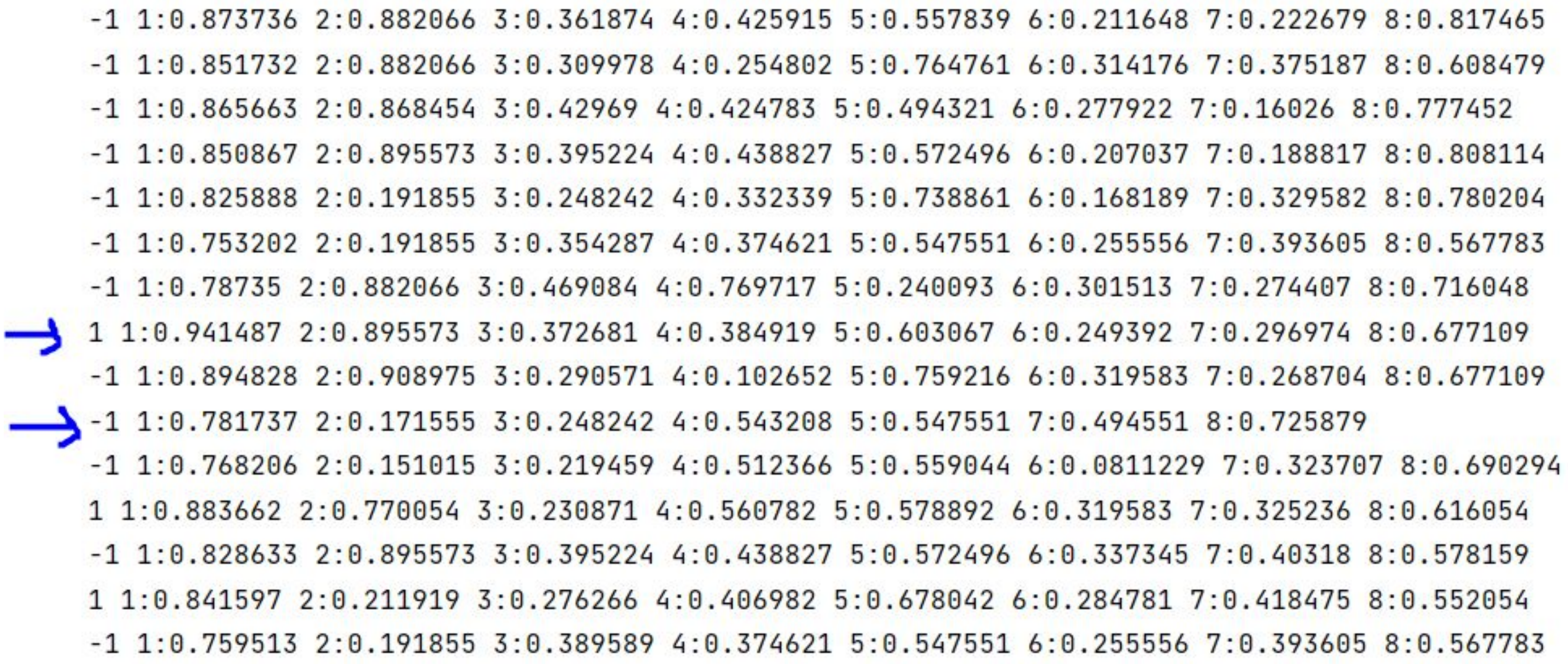
Far away points are also considered.



High Gamma

Only nearby points are considered.

SVM Output



```
  -1 1:0.873736 2:0.882066 3:0.361874 4:0.425915 5:0.557839 6:0.211648 7:0.222679 8:0.817465
  -1 1:0.851732 2:0.882066 3:0.309978 4:0.254802 5:0.764761 6:0.314176 7:0.375187 8:0.608479
  -1 1:0.865663 2:0.868454 3:0.42969 4:0.424783 5:0.494321 6:0.277922 7:0.16026 8:0.777452
  -1 1:0.850867 2:0.895573 3:0.395224 4:0.438827 5:0.572496 6:0.207037 7:0.188817 8:0.808114
  -1 1:0.825888 2:0.191855 3:0.248242 4:0.332339 5:0.738861 6:0.168189 7:0.329582 8:0.780204
  -1 1:0.753202 2:0.191855 3:0.354287 4:0.374621 5:0.547551 6:0.255556 7:0.393605 8:0.567783
  -1 1:0.78735 2:0.882066 3:0.469084 4:0.769717 5:0.240093 6:0.301513 7:0.274407 8:0.716048
→ 1 1:0.941487 2:0.895573 3:0.372681 4:0.384919 5:0.603067 6:0.249392 7:0.296974 8:0.677109
  -1 1:0.894828 2:0.908975 3:0.290571 4:0.102652 5:0.759216 6:0.319583 7:0.268704 8:0.677109
→ -1 1:0.781737 2:0.171555 3:0.248242 4:0.543208 5:0.547551 7:0.494551 8:0.725879
  -1 1:0.768206 2:0.151015 3:0.219459 4:0.512366 5:0.559044 6:0.0811229 7:0.323707 8:0.690294
  1 1:0.883662 2:0.770054 3:0.230871 4:0.560782 5:0.578892 6:0.319583 7:0.325236 8:0.616054
  -1 1:0.828633 2:0.895573 3:0.395224 4:0.438827 5:0.572496 6:0.337345 7:0.40318 8:0.578159
  1 1:0.841597 2:0.211919 3:0.276266 4:0.406982 5:0.678042 6:0.284781 7:0.418475 8:0.552054
  -1 1:0.759513 2:0.191855 3:0.389589 4:0.374621 5:0.547551 6:0.255556 7:0.393605 8:0.567783
```

Thank You