A PROJECT REPORT

ON

Building Polynomial Regression Models

BY

Kumar Pranjal (2018A7PS0163H)

Sneh Lohia (2018A7PS0171H)

Abhishek Mishra (2018A7PS0019H)

Under The Guidance of

Dr NL Bhanu Murthy

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

HYDERABAD CAMPUS

(NOVEMBER, 2020)

Assignment 3: Building Polynomial Regression Models

Introduction: In Statistics, Regression is an approach to modelling the relationship between a scalar response and one or more explanatory variables. When more than one explanatory variables are involved, the process is called Multiple Linear Regression.

We were given a Dataset called "Insurance.txt." The file consisted of 2 independent variables, namely age and BMI. One dependent variable is "Charges." We were asked to build a regression model using the dataset by applying polynomial features and predict the charges. Regression models are made using different methods, and the error associated with each algorithm is calculated.

Data Pre-Processing

Mentioned below are the steps involved before building the Model. They are:

- **Data Pre-Processing:** Data Preprocessing is a technique that transforms the given raw data into a clean dataset. This step improves the accuracy of the model. Data normalization has been done for this assignment.
- **Normalization:** This step is to make all the elements lie between 0 and 1, thus bringing all the values of numeric columns in the dataset to a standard scale.

We have used the normalize_dataset function, which normalizes the dataset by using the formula-

$$X_{normalized} = \frac{X - min(X_i)}{max(X_i) - min(X_i)}$$

- **Standardization:** Standardization is a useful technique to transform attributes with a Gaussian distribution and differing means and standard deviations to a standard Gaussian distribution with a mean of 0 and a standard deviation of 1.

  The formula to standardize the data is: $Z = \frac{(xi - \mu)}{\sigma}$

- **Adding polynomial features:** Linear data is converted to higher degree data by processing it with sklearn.preprocessing.PolynomialFeatures class.

Functions to calculate Error and Accuracy

We have used the Mean Square Error Method to calculate the Error -

$$MSE = \frac{\sum_{i=1}^{m} (y_i - \hat{y}_i)^2}{2m}$$

To find Accuracy, we have used the R^2 metric -

$$R^2 = 1 - \frac{2 \times MSE}{Var(Y)}$$

Here the value of R^2 can vary from negative infinity to +1 based upon the performance of the model. 0 R^2 means the model is predicting precisely like the mean line. Positive or negative values signify better or worse predictions, respectively.

Cost Function Formulas

Two Different Cost Functions are used in the Program :

$$\textbf{Lasso Cost} = MSE + \frac{\lambda_1}{2m} \sum_{i=1}^{M} |W_i|$$

$$\textbf{Ridge Cost} = MSE + \frac{\lambda_2}{2m} \sum_{i=1}^{M} W_i^2$$

Steps Used In Building The Model

The data was read from the “Insurance.txt” file. The random shuffling of the data is done each time a regression model is made using the random shuffle function. After random shuffling, the entire dataset is divided into three parts, training set, validation set and testing set. The training set consists of 70% of the dataset, and the testing set consists of 10% of the dataset, and Validation Set consist of 20% of Dataset. Two different algorithms, The gradient descent method and Stochastic gradient descent method are run for training the models, and errors are calculated.

Gradient Descent model

Function for finding W using Gradient Descent :

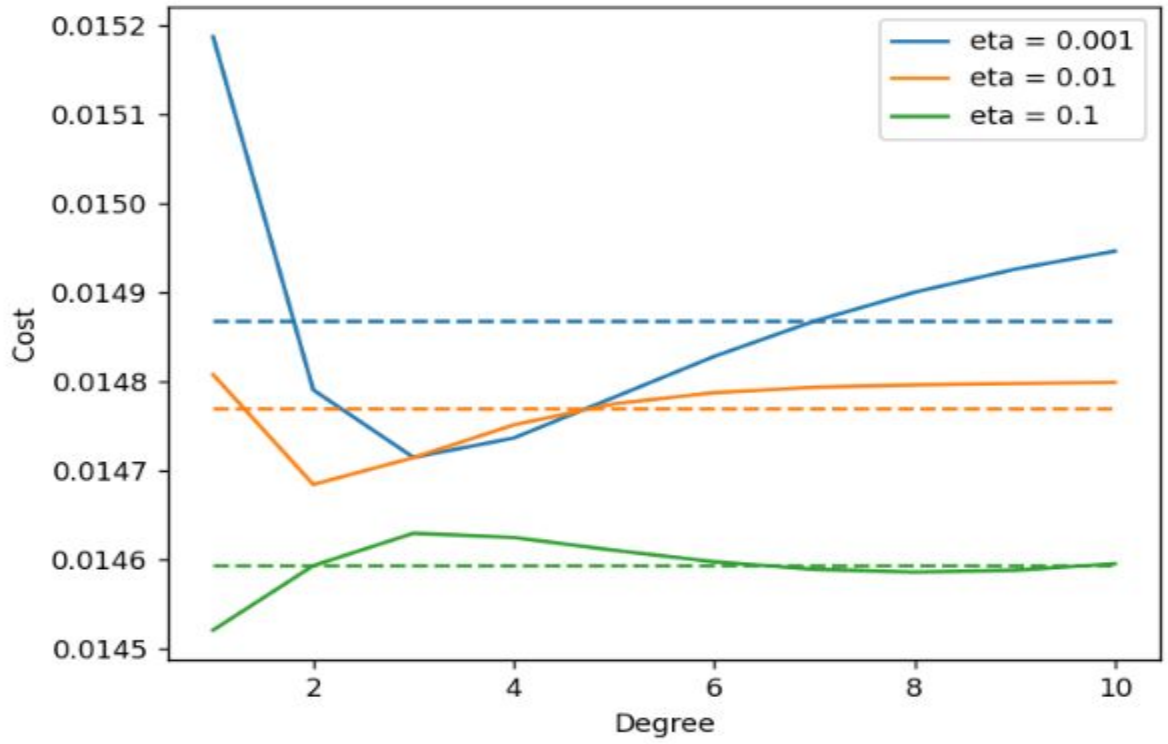
$$dW = (1 / m) * (X^T (XW - y))$$
$$W = W - (eta * dW)$$

The choice of learning rate is crucial for gradient and stochastic gradient descent methods. With a very high learning rate, the function may fail to converge and overshoot the minimum. If the learning rate is very low, the process takes a very long time to converge, which is computationally expensive.

Running without regularization for various learning rates

The Gradient Descent method was run for three different learning rates (Eta = 0.001, 0.01 and 0.1) without Regularization. For each learning rate, the model was trained for 10 degrees ranging from degree 1 to degree 10, and the results were analyzed.

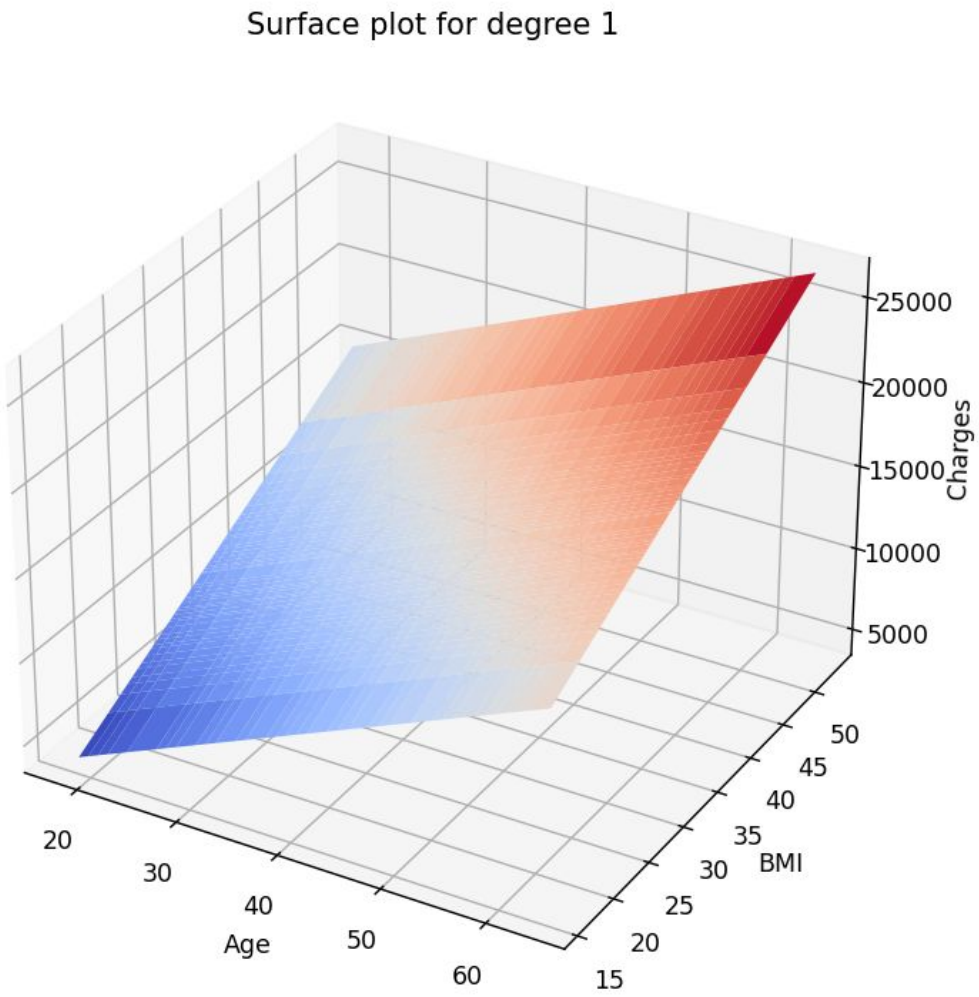
Visualizing the effect of learning rates

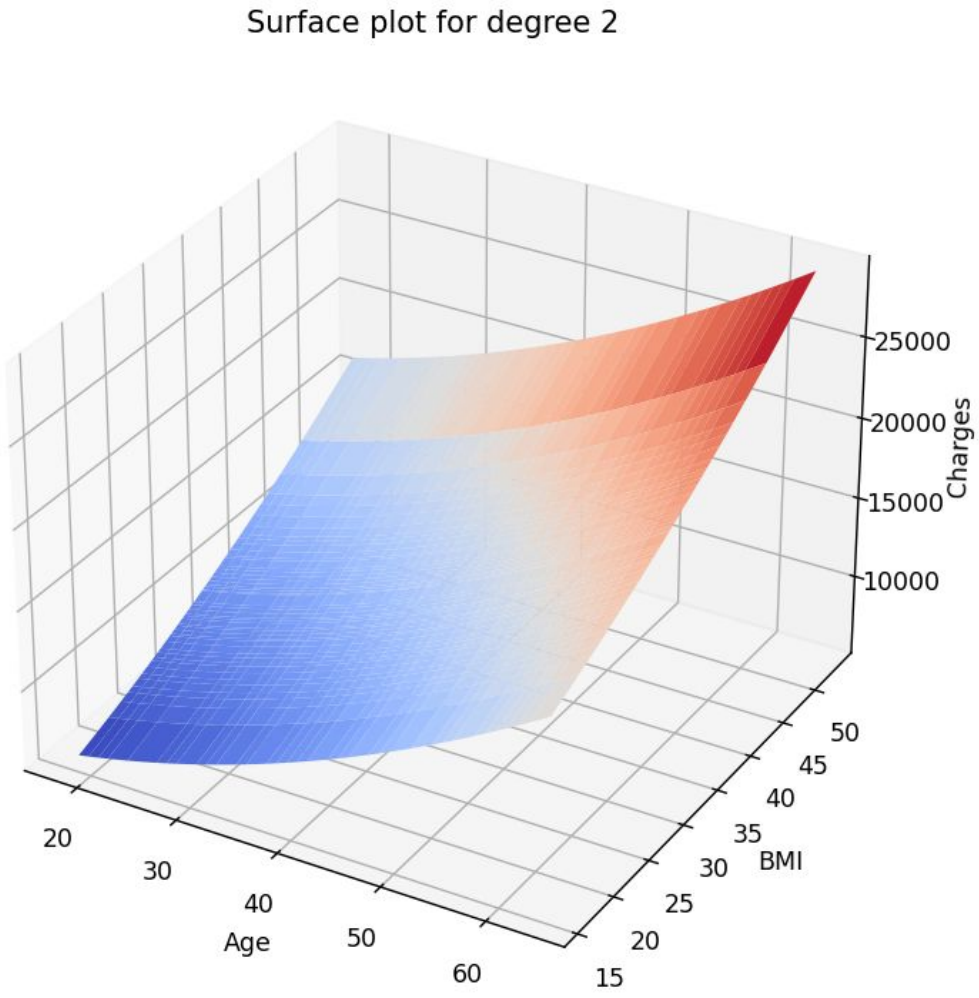


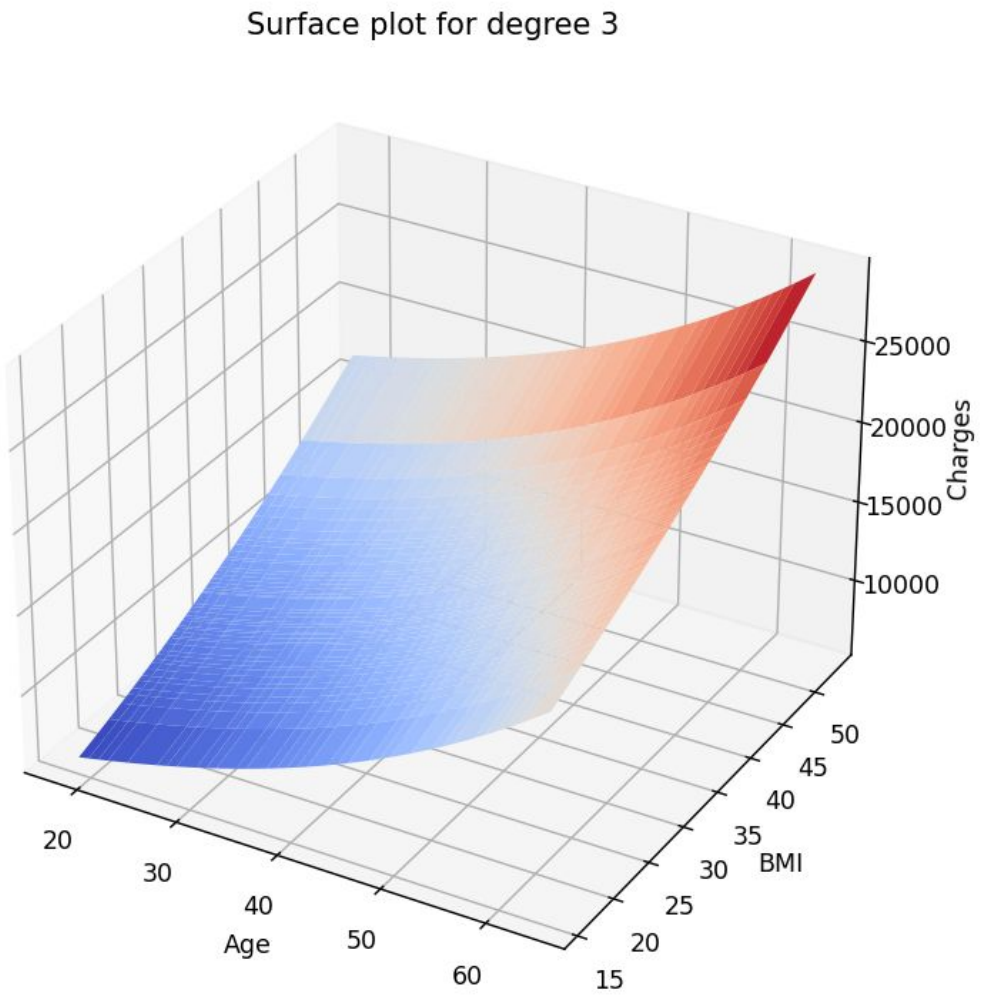
Finding the best learning rate

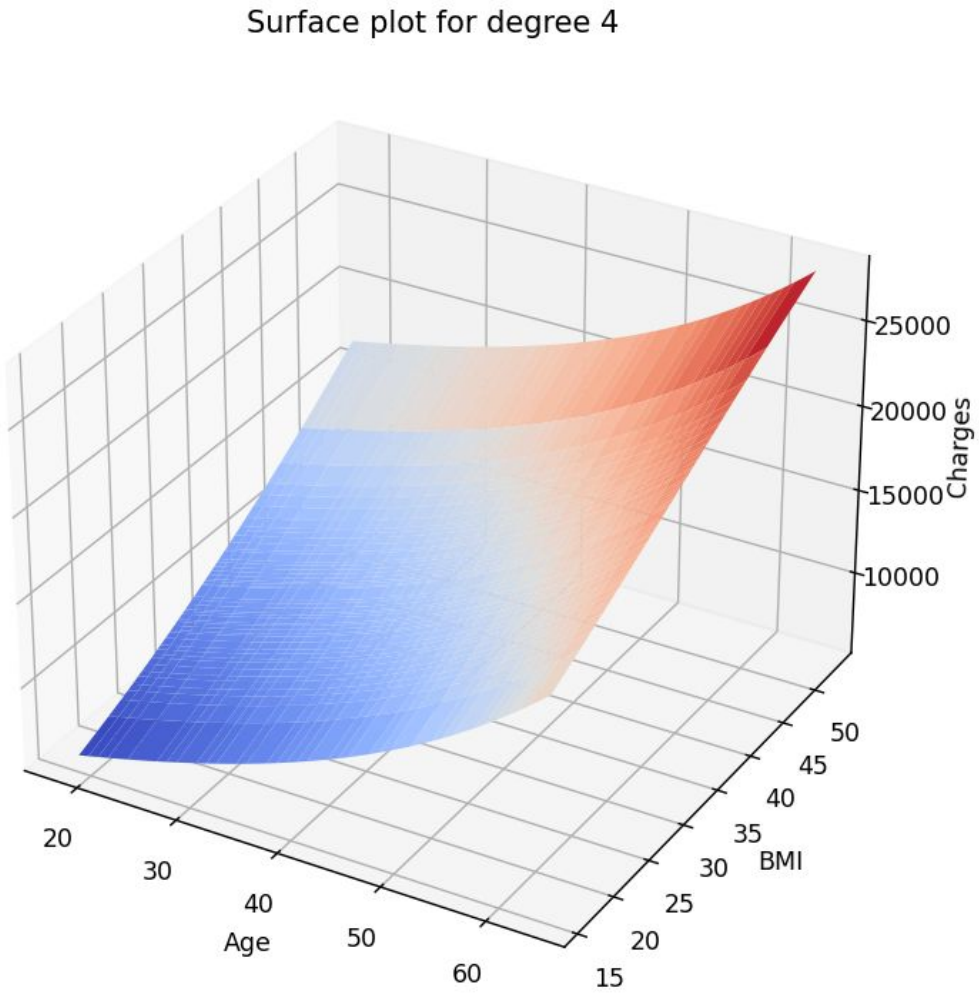
Best learning rate will be the one which has the lowest average validation cost.
Best learning rate with lowest average validation cost was found out to be 0.1

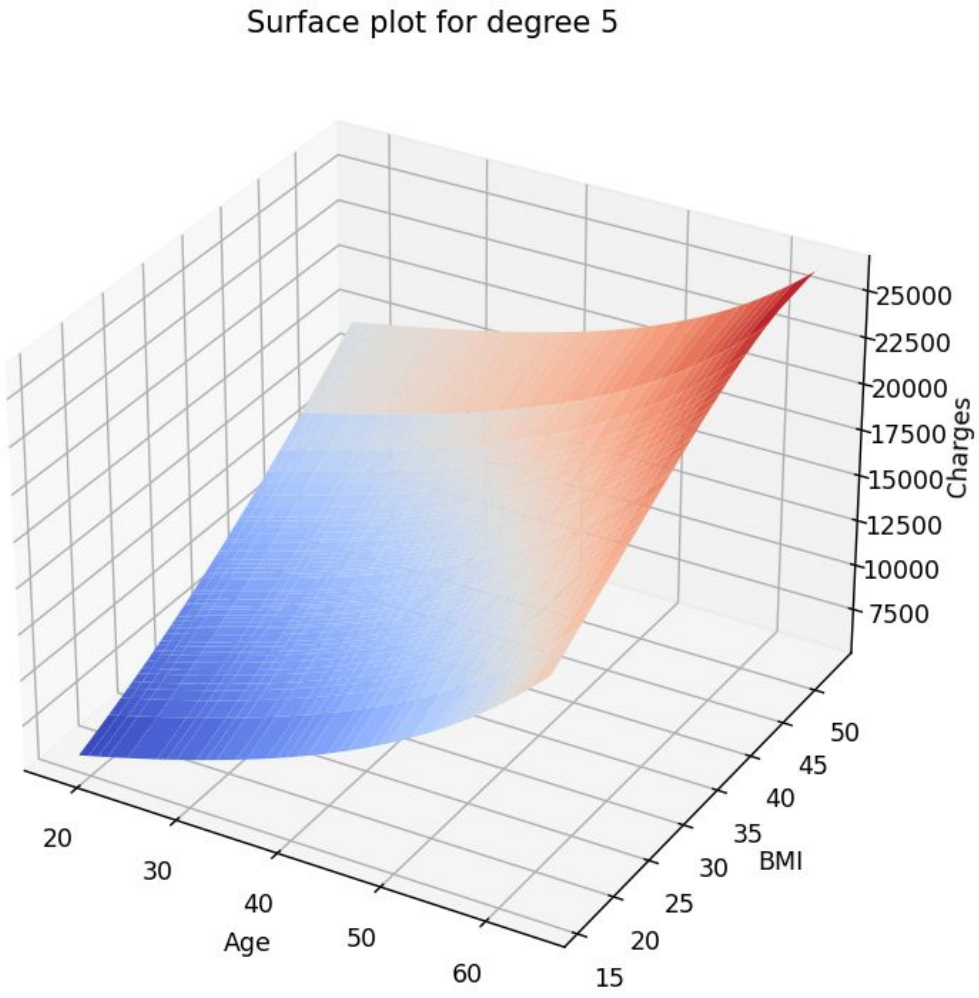
Generating Surface Plots for different degrees



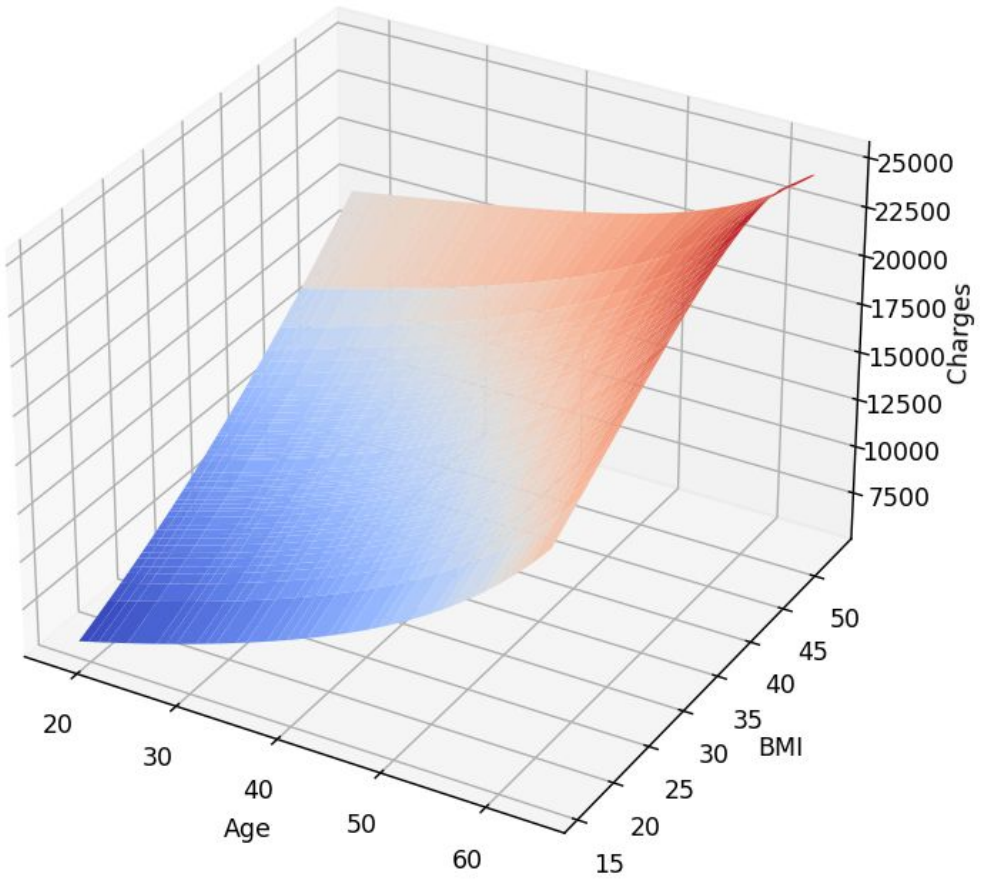

Surface plot for degree 2
25000
20000
15000
10000
Charges
20
30
40
50
60
Age
15
20
25
30
35
40
45
50
BMI


Surface plot for degree 3
25000
20000
15000
10000
Charges
20
30
40
50
60
Age
15
20
25
30
35
40
45
50
BMI

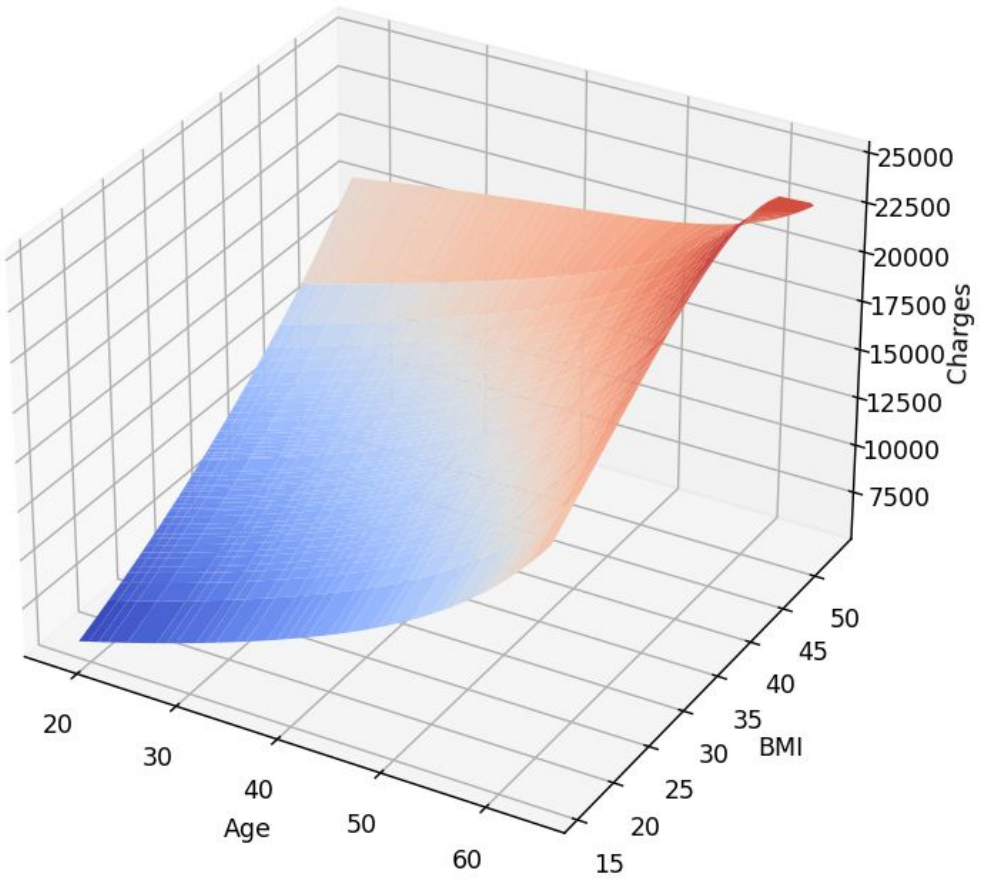

Surface plot for degree 4
25000
20000
15000
10000
Charges
20
30
40
50
60
Age
15
20
25
30
35
40
45
50
BMI


Surface plot for degree 5
25000
22500
20000
17500
15000
12500
10000
7500
Charges
20
30
40
50
60
Age
15
20
25
30
35
40
45
50
BMI

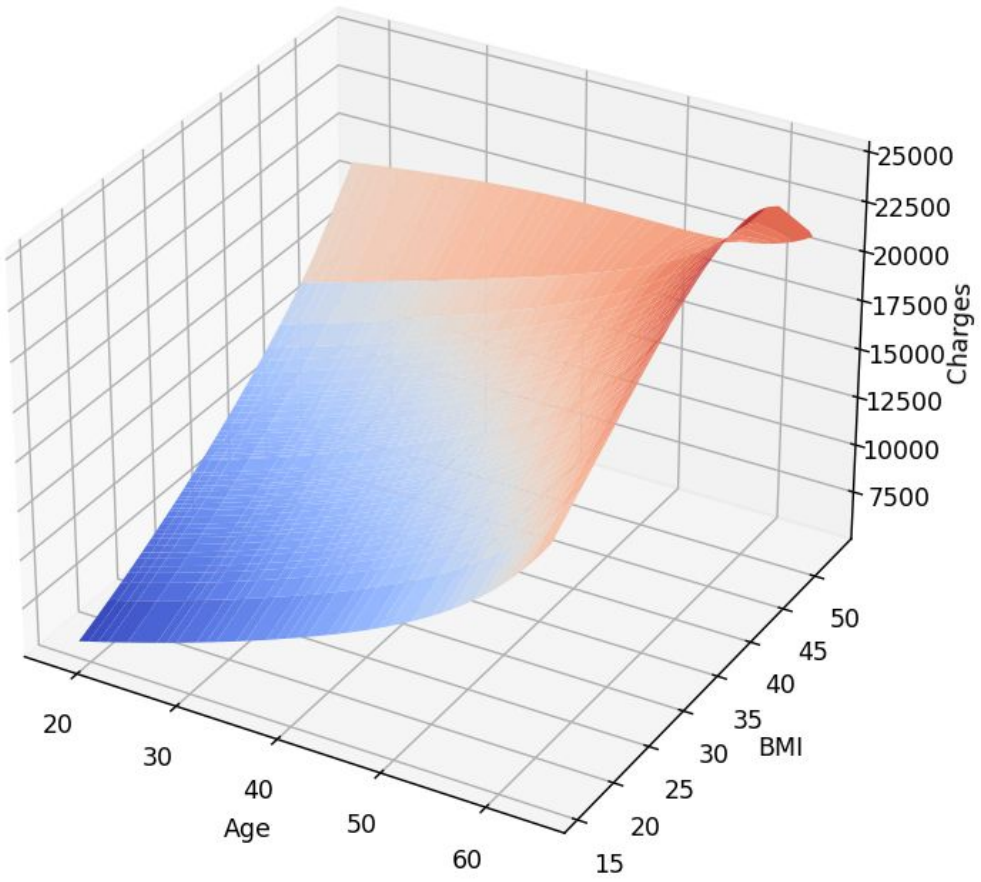
Surface plot for degree 6

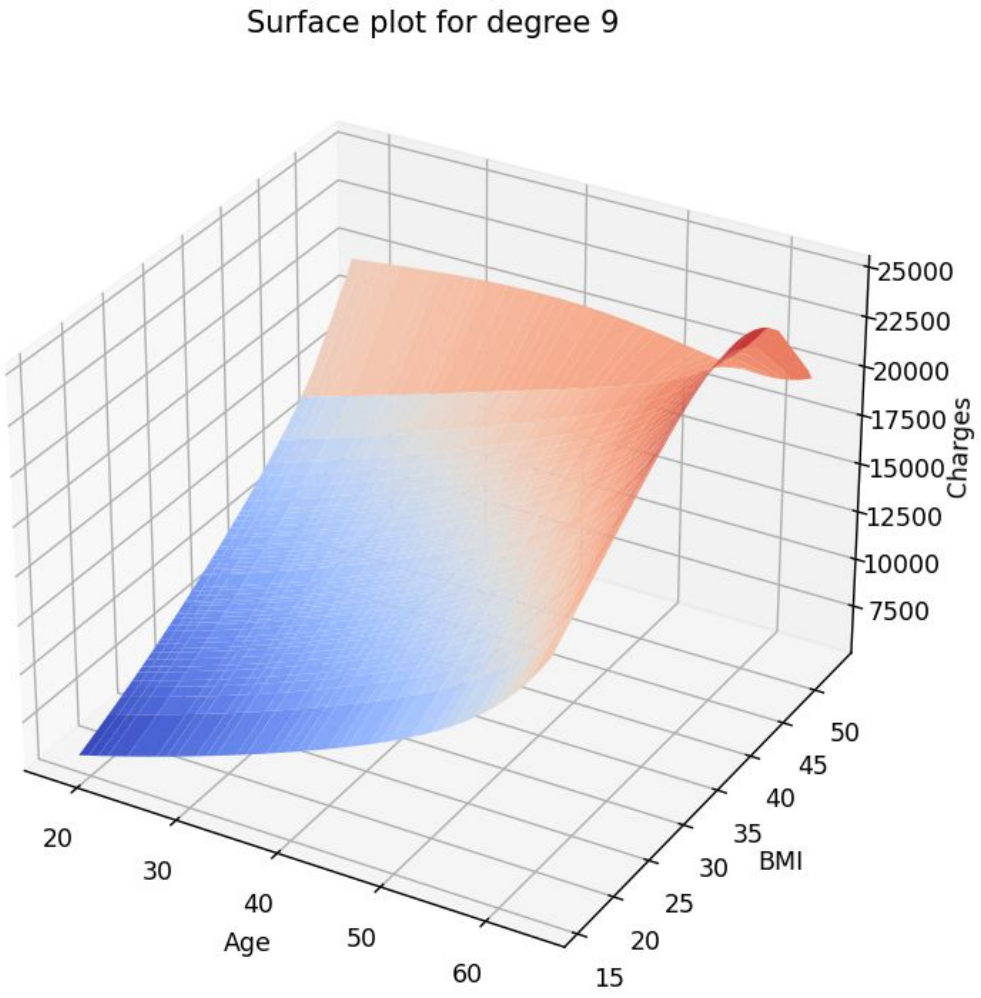


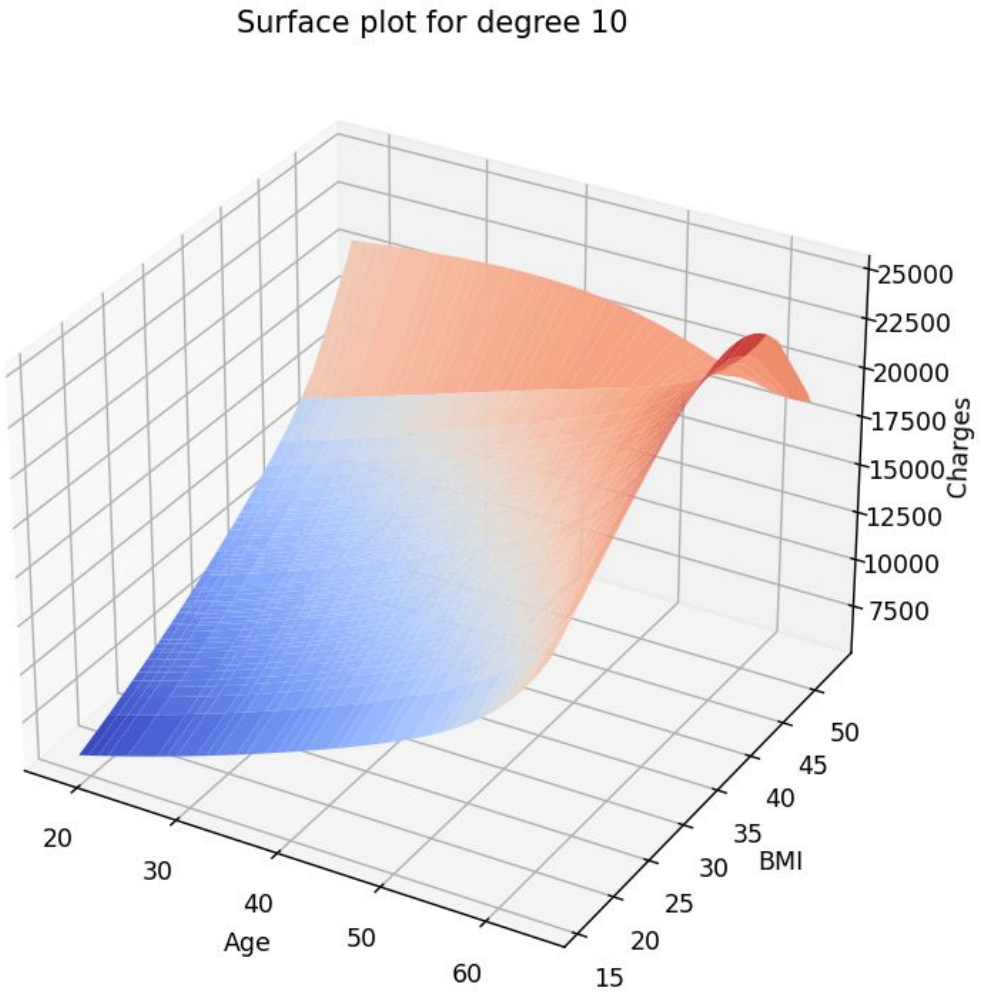
Surface plot for degree 7



Surface plot for degree 8




Surface plot for degree 9
25000
22500
20000
17500
15000
12500
10000
7500
Charges
20
30
40
50
60
Age
15
20
25
30
35
40
45
50
BMI


Surface plot for degree 10
25000
22500
20000
17500
15000
12500
10000
7500
Charges
20
30
40
50
60
Age
15
20
25
30
35
40
45
50
BMI

Running with Lasso Regularization for various λ

The Model was Run with Lasso Regualrization for Various Lamdas and the results were analyzed.

Finding best λ for different degrees

Best lamda for any particular degree will be the one which will have the lowest validation cost. Here are the best lamdas for different degrees :

1: 0.02910639252002367
2: 0.11059057123348093
3: 0.13491497736281066
4: 0.008702433279120236
5: 0.10949665721202306
6: 0.32902587678842
7: 0.07776776288159049
8: 0.06417858910380636
9: 0.029246577763218484
10: 0.0322162571903154

Running with Ridge Regularization for various λ

The Model was Run with Ridge Regualrization for Various Lamdas and the results were analyzed.

Finding best λ for different degrees

Best lamda for any particular degree will be the one which will have the lowest validation cost. Here are the best lamdas for different degrees :

1: 0.16141094315921056
2: 0.06018094083935677
3: 0.013413745151899081
4: 0.07946854352355492
5: 0.06809418221905927
6: 0.02494350625279751
7: 0.15598243092866415
8: 0.03499111333523519
9: 0.025000515794799116
10: 0.10661152587779177

Stochastic Gradient Descent model

SGD picks up a "random" instance of training data at each step and then computes the gradient making it much faster as there are much less data to manipulate at a single time, unlike GD, which picks up the whole batch at once.

Function for finding W using Stochastic Gradient Descent :

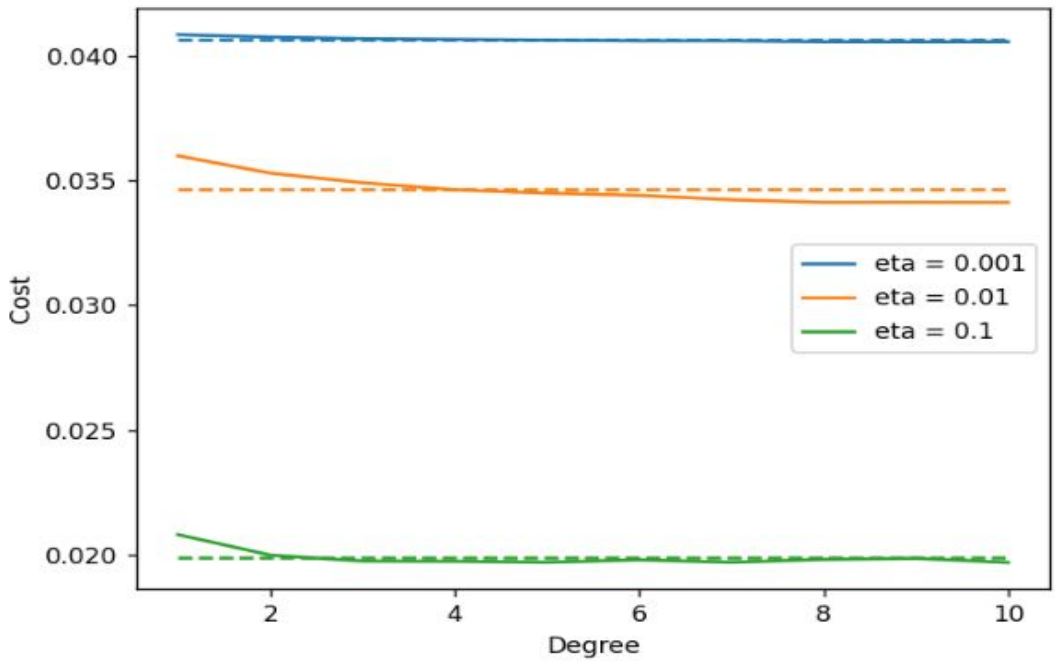
$$dW = (1 / m) * (X^T (XW - y))$$
$$W = W - (eta * dW)$$

Stochastic Gradient Descent method was also run for three different learning rates. For each learning rate, 10 models with varying degrees were trained, and the results were analyzed

Running without regularization for various learning rates

The Stochastic Gradient Descent method was run for three different learning rates (Eta = 0.001, 0.01 and 0.1) without Regularization. For each learning rate, the model was trained for 10 degrees ranging from degree 1 to degree 10, and the results were analyzed.

Visualizing the effect of learning rates



Finding the best learning rate

Best learning rate will be the one which has the lowest average validation cost.
Best learning rate with lowest average validation cost was found out to be 0.1

Running with Lasso Regularization for various λ

The Model was then Run with Lasso Regualrization for Various Lamdas and the results were analyzed.

Finding best λ for different degrees

Best lamda for any particular degree will be the one which will have the lowest validation cost.
Here are the best lamdas for different degrees :

1: 0.15987456751605156
2: 0.0011941393451226912
3: 0.05408743612052913
4: 0.10431502768434397
5: 0.02571585119901809
6: 0.18131582476542651
7: 0.10080480283740034
8: 0.18044924826777475
9: 0.03151045213541681
10: 0.01246642218524263

Running with Ridge Regularization for various λ

The Model was Run with Ridge Regualrization for Various Lamdas and the results were analyzed.

Finding best λ for different degrees

Best lamda for any particular degree will be the one which will have the lowest validation cost.
Here are the best lamdas for different degrees :

1: 0.06764920434741795
2: 0.005198914147222622
3: 0.11083165361142489
4: 0.0768573695730771

5: 0.0698843411566864
6: 0.03658109114909791
7: 0.026669855662970532
8: 0.011156119608608961
9: 0.027137978476624358
10: 0.05551366767898058

Cost Table for Validation

Degree	GD No Reg	GD R1 Reg	GD R2 Reg	SGD No Reg	SGD R1 Reg	SGD R2 Reg
1	0.014520540127792825	0.016949989275169672	0.016884917076748152	0.020818713635471085	0.022863796430565958	0.020754092305432334
2	0.014593049414749287	0.017044971833293932	0.016797105103000011	0.019999567204450265	0.015102843613906571	0.019286092088012238
3	0.014629479675423093	0.017091261416131327	0.016796555351761375	0.019759668347331494	0.016839503522426225	0.019395465947255264
4	0.014624686746326832	0.016978513888863393	0.016804453650381633	0.019752238033091934	0.019509191108578447	0.018964098201749637
5	0.014610321685892647	0.017073002422490657	0.016800157426803585	0.019710064902545119	0.015398911969343367	0.019013718970998 8
6	0.014597413688788831	0.017252126022341392	0.016790744485922694	0.019808157915860004	0.023634631501499338	0.019011235659479656
7	0.014588952486884252	0.017034280839620669	0.016799631596366288	0.019710476320727473	0.019632356177454024	0.018919848495421825
8	0.014585729110393 3	0.017015463437831 3	0.016781324244463647	0.019825716963519403	0.023849766049178 6	0.019069789090542844

9	0.0145878702381658 5	0.0169615563028202 5	0.0167747007784453 05	0.0198689538400877 4	0.0157663362557580 52	0.0189918369093523 66
10	0.0145951992169542 72	0.0169676511165831 6	0.0167782620896917 6	0.0197024351921085 5	0.0150865765833788 04	0.0191874670593391 03

NR = Non Regularized , R1 = Lasso, R2 = Ridge

QUESTIONS TO PONDER:

1. **What happens to the training and testing error as polynomials of higher degree are used for prediction?**

   Ans. The training error continually decreases due to increased flexibility. But the error in the validation set decreases only upto a certain limit. On further increasing the flexibility, the error in the validation set increases. This happens because the model learns noise along with testing data.

2. **Does a single global minimum exist for Polynomial Regression as well? If yes, justify.**

   Yes, a single global minima exists for polynomial regression. For polynomial regression, the training error is high before a certain degree of polynomial, while after that degree, the testing error is higher. Therefore, there is a single degree of polynomial where the error is least.

3. **Which form of regularization curbs overfitting better in your case? Can you think of a case when Lasso regularization works better than Ridge?**

   In our case, Ridge regularization works better.

   The disadvantage of ridge regression is model interpretability**.** It will shrink the coefficients for least important predictors, very close to zero but it will never make them exactly zero. In other words, the final model will include

all predictors. However, in the case of lasso, the penalty has the effect of forcing some of the coefficient estimates to become exactly equal to zero when the tuning parameter λ is sufficiently large. Therefore, in this scenario, Lasso performs better than the ridge regression model.

4. **How does the regularization parameter affect the regularization process and weights? What would happen if a higher value for λ (> 2) was chosen?**

   Increasing the value of regularization parameter reduces the value of coefficients and thus reduces the variance but does not lose the important properties in the data. We need to find an optimal value of λ so that the generalization error is minimum.

   λ is a hyper parameter whose value is decided by us. If **λ(>2)** is high it adds a high penalty to the error term making the learned hyper plain almost linear. Hence, for a higher value of **λ ,** the algorithm fails even to fit the training data and result in underfitting. Gradient descent will also fail to converge.

5. **Regularization is necessary when you have a large number of features but limited training instances. Do you agree with this statement?**

   Yes, we agree with the statement.Regularization significantly reduces the variance of the model without substantial increase in its bias. If we add many new features, the model becomes more expressive and it fits better to our training set. If too many new features are added, this can lead to overfitting of the training set. Hence, regularization is necessary to zero-out some features in order to avoid overfitting and to reduce the computational complexity.

6. **If you are provided with D original features and are asked to generate new matured features of degree N, how many such new matured features will you be able to generate? Answer in terms of N and D.**

   The number of matured features will be :

$$^{N+D-1}C_{D-1} + {}^{N+D-2}C_{D-2} + {}^{N+D-3}C_{D-3} + \ldots + {}^{N}C_{0}$$

7. **What is bias-variance trade off and how does it relate to overfitting and regularization?**

Bias-variance trade off refers to the trade off we encounter while choosing the number of parameters while building a good model. For example, if the model is too simple and has very few parameters, it will have high bias and low variance. If we start increasing the number of parameters, the variance starts increasing and bias decreases accordingly. To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.

Overfitting happens when our model captures the noise along with the underlying pattern in data. It happens when we train our model a lot over noisy dataset. These models have low bias and high variance.

The regularized model has higher bias than the polynomial fit but the testing error is greatly improved. The goal that is to avoid fitting the random noise and thus eliminating the high variance issue can be achieved by the bias-variance tradeoff. Thus, the bias-variance tradeoff can be leveraged to improve model predictive capability.