Assignment #1

Zahra Parham 2122841

1- Sampling

1.1-

random ←—-random sampling from uniform distribution over (0,1)

if random < 0.2 ***then***:

    Movie ++;

else if 0.2 ≤ random < 0.6 ***then***:

    INF8245E ++;

else if 0.6 ≤ random < 0.7 ***then***:

    Playing ++;

else:

    Studying ++;

END

1.2-

```
in 100 days - movie: 0.18, INF8245E:0.37, playing:0.15, studying:0.3
in 1000 days - movie: 0.216, INF8245E:0.404, playing:0.09, studying:0.29
```

Based on comparing the fraction in 100 days and 1000 days, we can assume that by increasing the number of sampling, each activity will get closer to its probability

2- Model Selection

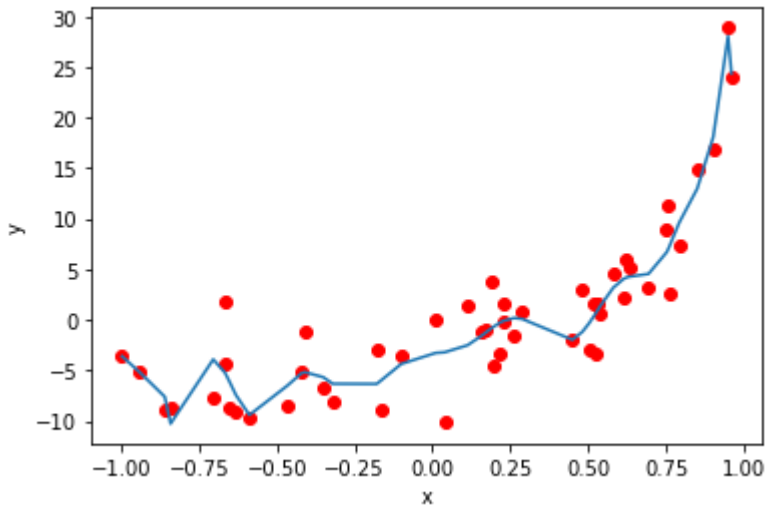
1.

1. **a-**

```
Train RMSE:2.6774364810380065, Validation RMSE18.352463341124924
```

---

1. **b-**

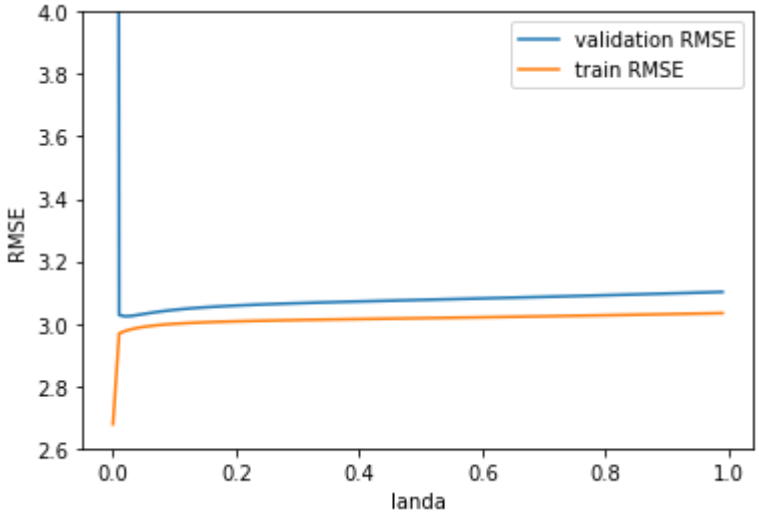


1. **c-**

The model is overfitting since, based on the plot, it fits the train data too closely, and also based on comparing the RMSE for train data and test data, we can assume that the model is overfitting.

2.

2. **a-**

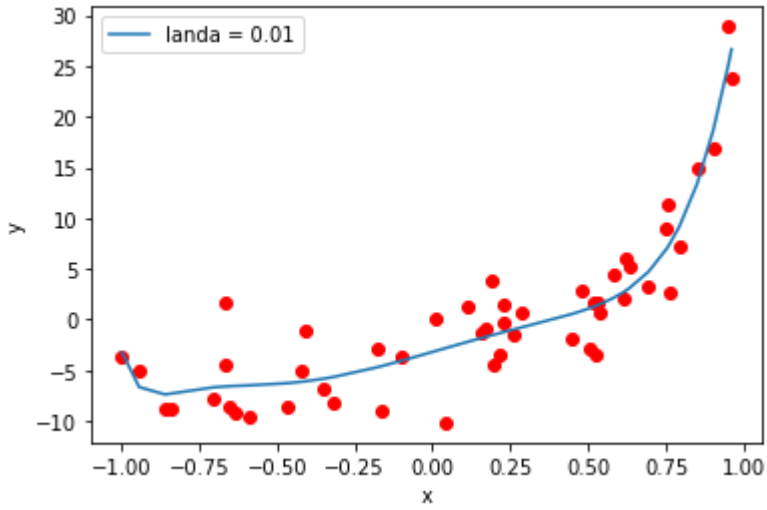


2. **b-**

based on the above plot, the best value for the landa is 0.01:

```
RMSE for test data with the best landa = 0.01 is: 3.2925595397745555
```

2. **c-**



2. **d-**

This is a good fit for the data because:

1- based on the plot, it does not fit too closely or does not have enough flexibility in terms of line fitting(learning too much or too small)

2- based on the RMSE on test and train data, we can assume that the model is not overfitting or underfitting since it has good RMSE(result) on both the train and test dataset

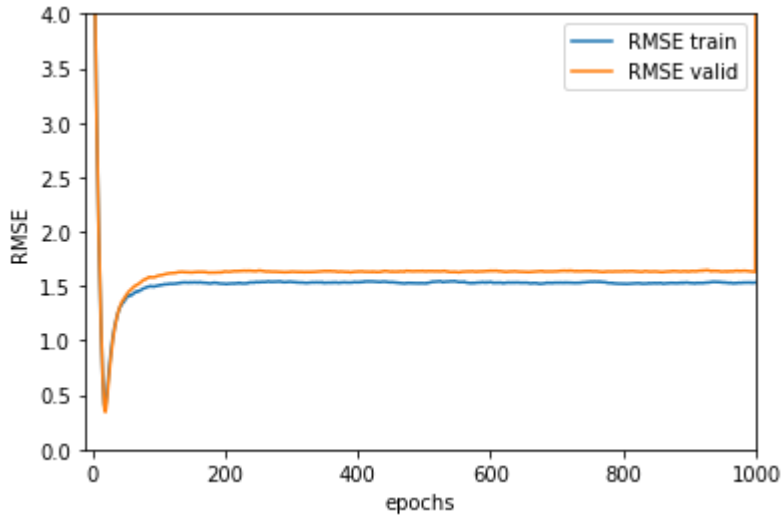
3.

it's hard to infer the exact degree from the last plot but from comparing the last plot with the other degree polynomial it can be a third-degree polynomial

3- Gradient Descent for Regression

1.

1. **a**



2.

2. **a-**

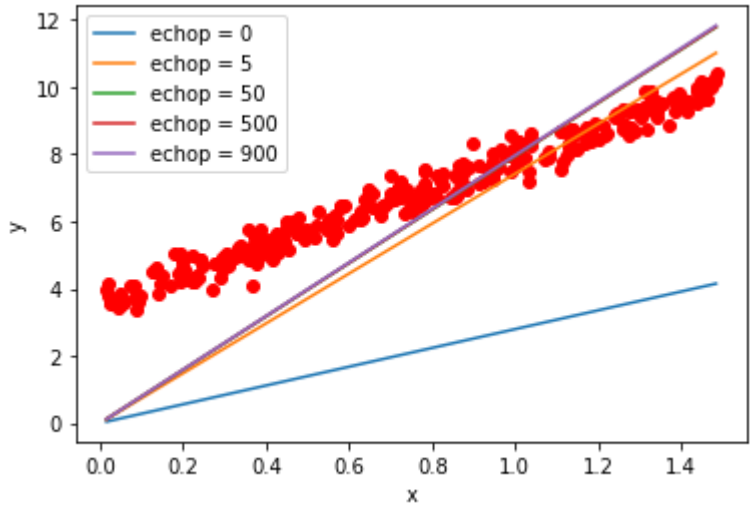
	0	1
0	step size	rmse
1	0.0001	1.63535537775918
2	0.001	1.62534367284661
3	0.01	1.685956558252589
4	0.1	1.7243225040018213
5	0.5	1.932250067612219

The best step size is 0.001

2. **b-**

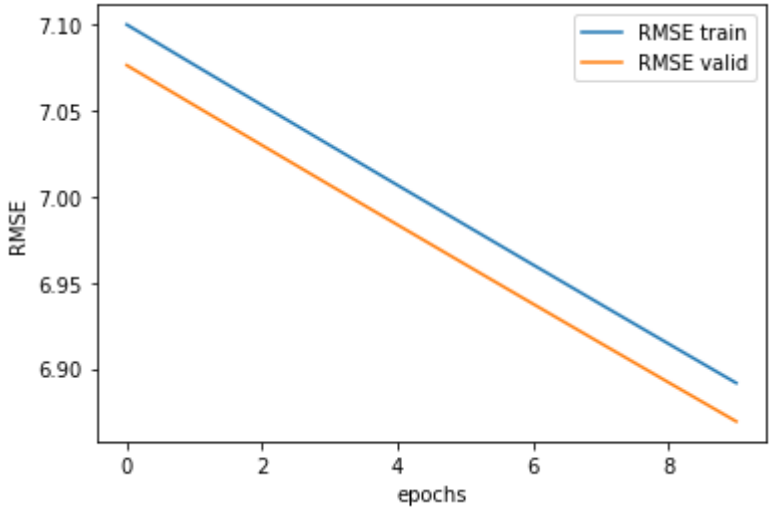
RMSE test with step size= 0.001 is 1.6147289304723258.

3.

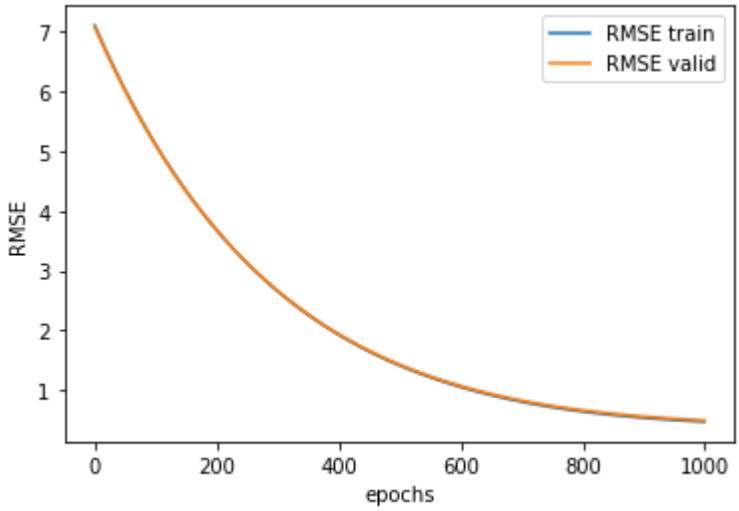


4.

with iteration = 10



with iteration = 1000



5.

1- on Full Gradient Descent we run the algorithm through all the data in your training set and do a single update for the parameters(w, b), on the other hand, in SGD we run the algorithm on just one sample or a subset of data in training data.

2- gradient descent (with a large number of data) can be too slow to converge but SGD is faster since we update the parameter based on a small number of samples

3- The error function in SGD is not as minimum as GD

4- Real life dataset

1.

1. **a-**

filled the missing data with the mean of each column:

	0	1	2	3	4	5	6	7	8	9	...	118	119	120	121	122	123	124	125	126	127
0	8	58	46188	Lakewoodcity	1	0.19	0.33	0.02	0.9	0.12	...	0.12	0.26	0.2	0.06	0.04	0.9	0.5	0.32	0.14	0.2
1	53	58	46188	Tukwilacity	1	0	0.16	0.12	0.74	0.45	...	0.02	0.12	0.45	0	0	0	0	0	0	0.67
2	24	58	46188	Aberdeentown	1	0	0.42	0.49	0.56	0.17	...	0.01	0.21	0.02	0	0	0	0	0	0	0.43
3	34	5	81440	Willingborotownship	1	0.04	0.77	1	0.08	0.12	...	0.02	0.39	0.28	0	0	0	0	0	0	0.12
4	42	95	6096	Bethlehemtownship	1	0.01	0.55	0.02	0.95	0.09	...	0.04	0.09	0.02	0	0	0	0	0	0	0.03

filling the missing data with the mean of each column maybe not be a good choice especially in skewed data and it can also reduce the variance of the data which can produce bias in our model. it is also doesn't factor in the correlation between features. it can be used for numerical datasets and also it only works on the column level

1. **b-**

1- ignore the data that is missing which is not a good way since you might lose some valuable information(Dropping rows with null values, Dropping features with high nullity)
2-imputation using mean/median
3- imputation using the most frequent item
4-imputation using zero or constant
5-imputing using k-nn algorithm
6- linear/stochastic regression imputation

1. **c-**

in regression imputation, we fill the missing data by predicting it by using the regression model. we will predict the missing data with the information of other variables.

In the first step, we fill the missing data with some trivial method like filling with mean of each column, and then the regression model is estimated in the information of other data and using the regression weights to predict the missing data.

1. **d-**

	0	1	2	4	5	6	7	8	9	10	...	118	119	120	121	122	123	124	125	
0	8	-1.3294e+76	-1.3294e+76	1	0.19	0.33	0.02	0.9	0.12	0.17	...	0.12	0.26	0.2	0.06	0.04	0.9	0.5	0.32	
1	53	-1.3294e+76	-1.3294e+76	1	0	0.16	0.12	0.74	0.45	0.07	...	0.02	0.12	0.45	-1.3294e+76	-1.3294e+76	-1.3294e+76	-1.3294e+76	0	-1.329
2	24	-1.3294e+76	-1.3294e+76	1	0	0.42	0.49	0.56	0.17	0.04	...	0.01	0.21	0.02	-1.3294e+76	-1.3294e+76	-1.3294e+76	-1.3294e+76	0	-1.329
3	34	5	81440	1	0.04	0.77	1	0.08	0.12	0.1	...	0.02	0.39	0.28	-2.34403e+76	-2.34403e+76	-2.34403e+76	-2.34403e+76	0	-2.3440
4	42	95	6096	1	0.01	0.55	0.02	0.95	0.09	0.05	...	0.04	0.09	0.02	-1.75461e+75	-1.75461e+75	-1.75461e+75	-1.75461e+75	0	-1.7546

5 rows × 127 columns

2.

2. **a-**

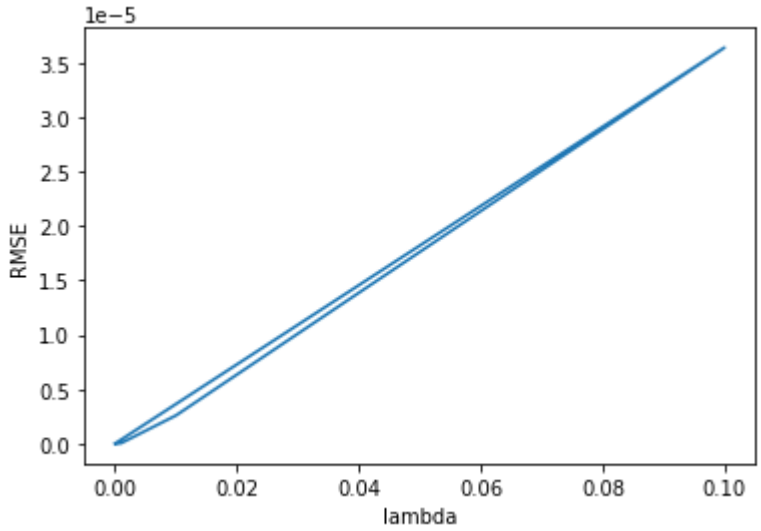
```
5-fold cross-validation average RMSE is :0.0014401135767270292
```

2. **b-**

test RMSE is : 0.008347174346829198

3.

3. **a-**



3. **b-**

lambda = 0.1 is the best fit

3. **c-**

test RMSE is :0.022755918237488515

3. **d-**

yes, we can use the information for feature selecting. we can omit(delete) the features where w_i=0. if the w=0 in regression with regularization it means that that feature_i related to w is not important and does not have a huge impact on our prediction

3. **f-**

by reducing the feature we will reduce the computational complexity of the model and just consider the feature that is most important for our prediction. so it will decrease the RMSE error as well.