YANG LU 260631276

Discussed with Zhuoran Zhao

Q1

1.1 pseudocode

```
create a list named activityCollect
    For I in range(num_of_days):
        One Activity sample from np.random.uniform between 0 and 1
        activityCollect.append(Activity)
    For I in range(num_of_days):
        If activityCollect[I]<0.2: put it in Movies
        Elif activityCollect[I]<0.6: put it in COMP551
        Elif activityCollect[I]<0.7:put it in Playing
        Else PUT activityCollect[I] in Study.
    Calculate the ratio of  elements in Movies, COMP551, Playing, Study with respect to num_of_day.
```

1.2

Result from simulation:

```
fractions(100-day simulation): {'Movie': 0.18, 'COMP551': 0.41, 'Playing': 0.11, 'Studying': 0.3}
fractions(1000-day simulation): {'Movie': 0.202, 'COMP551': 0.403, 'Playing': 0.102, 'Studying': 0.293}
```

	Movie	COMP551	Playing	Studying
100 day	0.18	0.41	0.11	0.3
1000 day	0.202	0.403	0.102	0.293
Target ratio	0.2	0.4	0.1	0.3
absError 100 day	0.02	0.01	0.01	0

absError 1000 day	0.002	0.003	0.002	0.007

Overall error of 100-day simulation: 0.04

Overall error of 1000-day simulation: 0.014

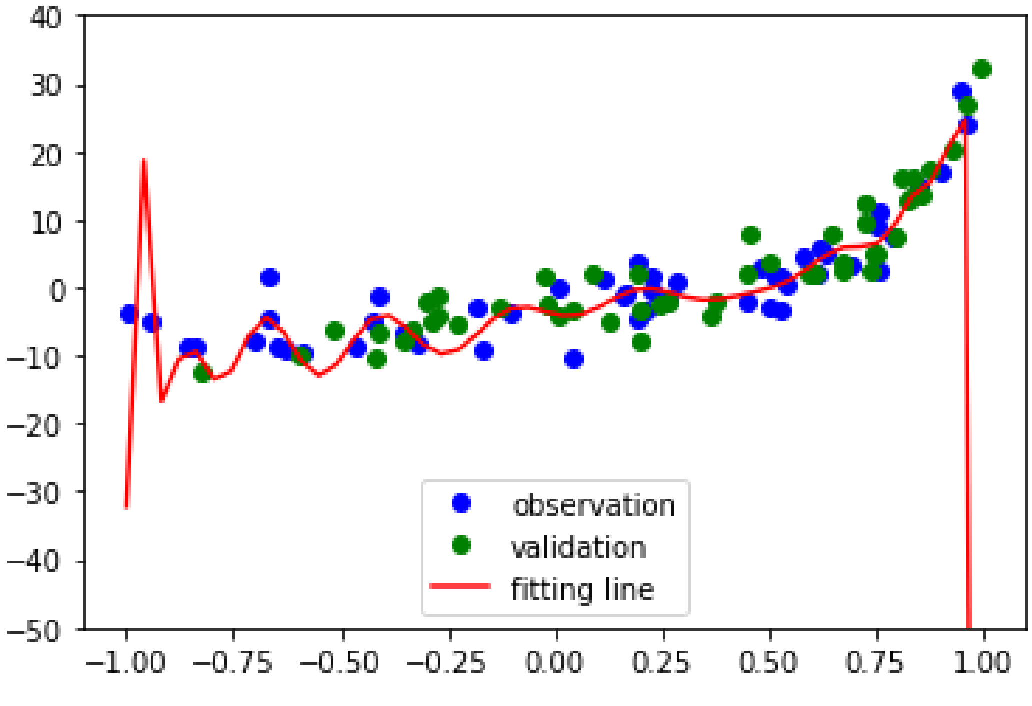
Conclusion: 1000-day simulation has a better performance in terms of simulating the multinomial distribution.

Q2:

2.1.a

```
Training Set MSE: 6.474690849607278
Validation Set MSE: 1420.5567393797228
```

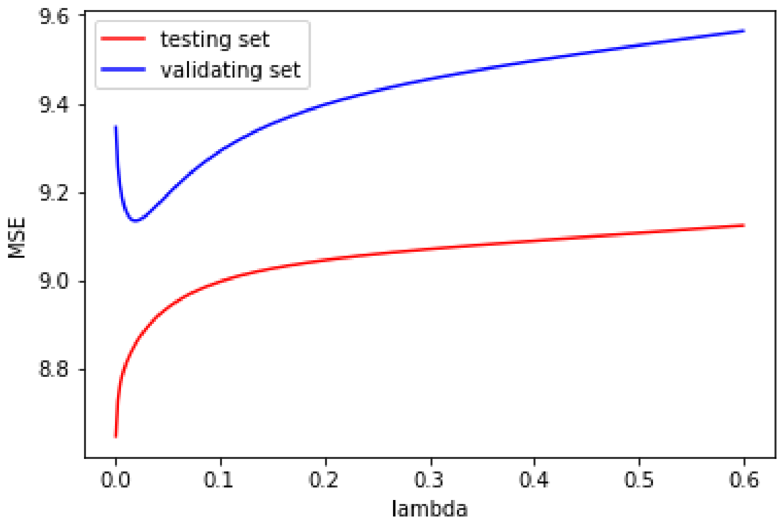
2.1.b



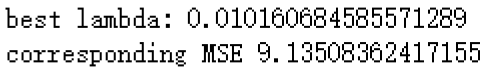
2.1.c

The model is overfitting due to the following reasons: firstlt, although the training set MSE is small, suggesting that the model almost perfectly fit the training data, the validation MSE is pretty high. Secondly, from the plot we can see that at the two ends(-1 and 1), the fitting line tends to fluctuate dramatically while the green dot near 1 does not drop obviously. This also suggests our model is being overfitting.

2.2.a



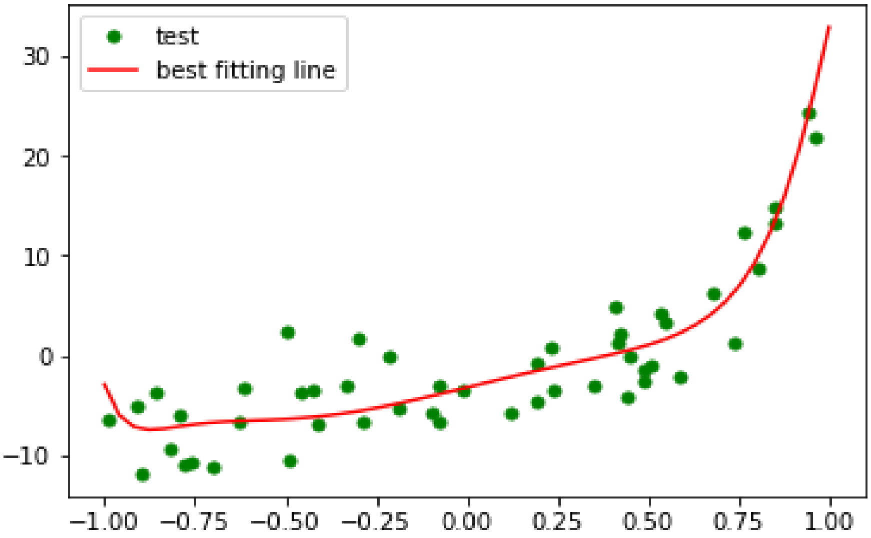
2.2.b



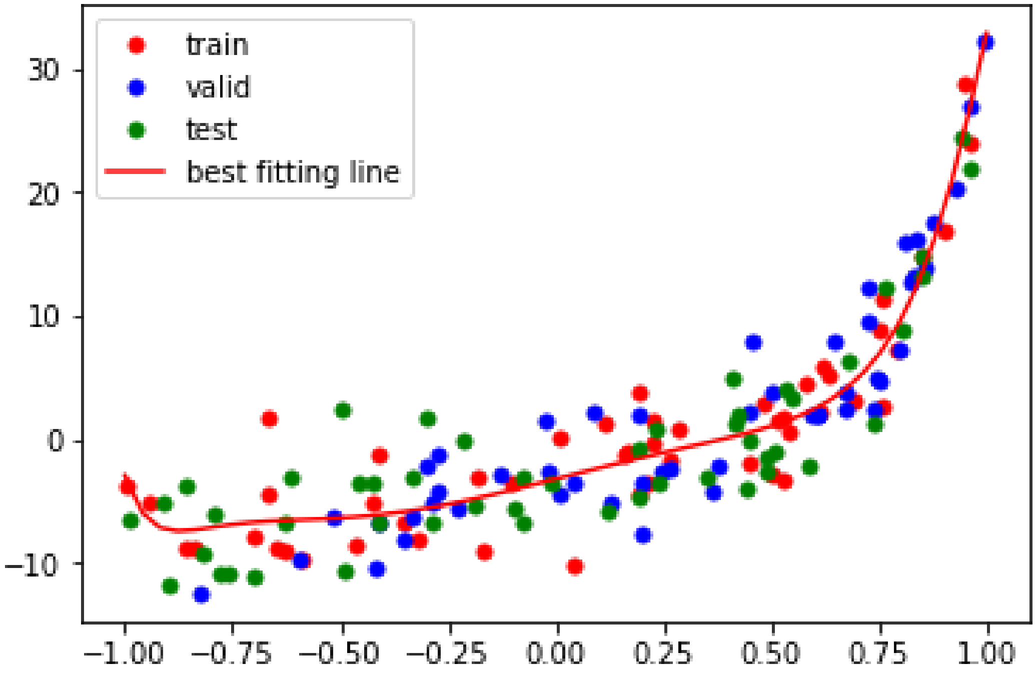
```
best lambda: 0.010160684585571289
corresponding MSE 9.13508362417155
```

The best lambda value produced by minimizing the validation MSE is 0.010160684585571289, and the corresponding minimized validation MSE is 9.13508362417155.

2.2.c



2.2.d



This L2 regularized model (with coefficient best lambda) fits well with training set, validation set and the test set.

2.3

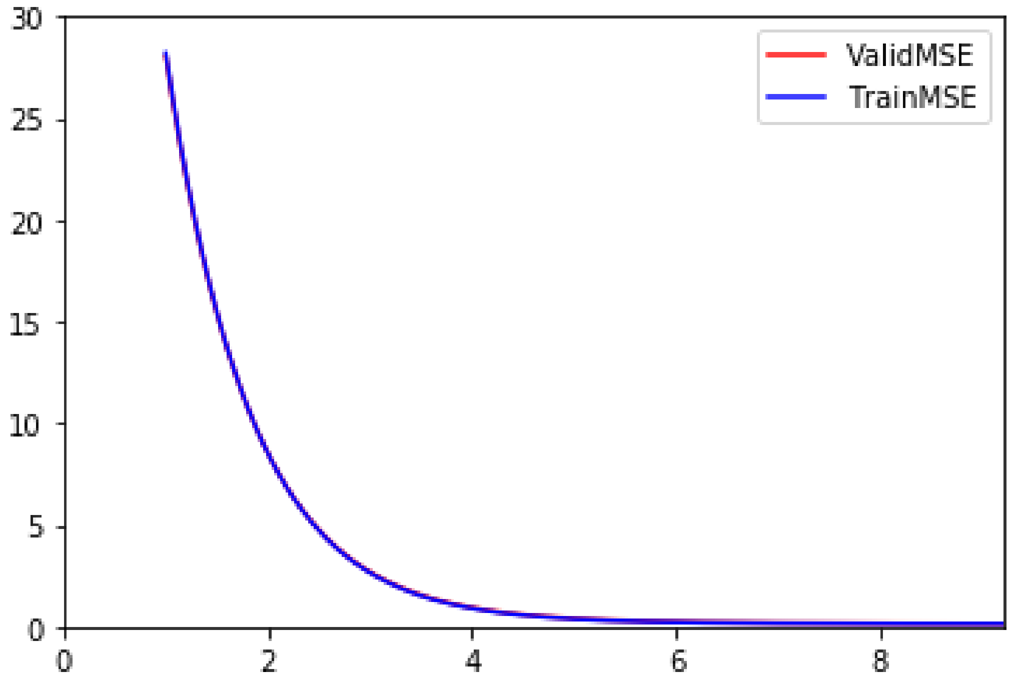
From my point of view, the degree of the source polynomial is 2 or 3. To support this, notice that there exists a nonlinear, most likely quadratic trends at the right end around x=1.00. Also, the data near the other end around x=-1 suggests a locally decreasing behavior.

Q3:

3.1.a

Validation MSE Saved to “Assignment1_260631276_3_1_a.txt”

3.1.b



3.2.a

With same amount of epoch: iteration=10000

	Step size	Validation MSE
1	1e-1	0.09504700958892913
2	1e-2	0.07385177731131386
3	1e-3	0.07408975946396018

	Step size	VMSE
2	1e-2	0.07372359233739122
3	1e-3	0.07401668811945634
4	1e-4	0.0753355722063099
5	1e-5	0.15871000177861416
4	1e-4	0.07407217343572765
5	1e-5	0.07533446302864259
6	1e-6	0.15871064336470503
7	1e-7	18.856945232407316

	Step size	VMSE
2	1e-2	0.07370361157331304
3	1e-3	0.07531194807696287
4	1e-4	0.15866421258537258

Clearly the best step size among these 7 options must be one of 2,3,4,5

Now we reduce the amount of epoch to 1000

0.07372359233739122, '3': 0.07401668811945634, '4': 0.0753355722063099, '5': 0.15871000177861416

Now we reduce the amount of epoch to 100

Now we reduce the amount of epoch to 50

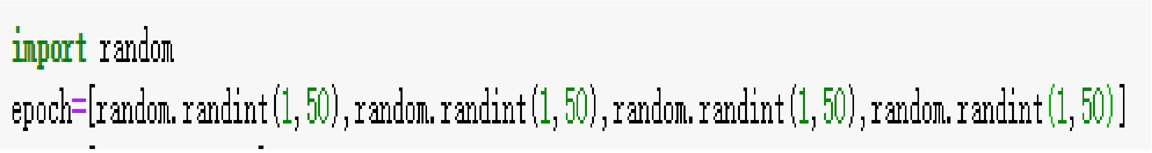
Conclusion: stepsize= 1e-2 is the best among the 7 graphs we tried.

3.2.b

And the corresponding test MSE is 0.06924258191661617

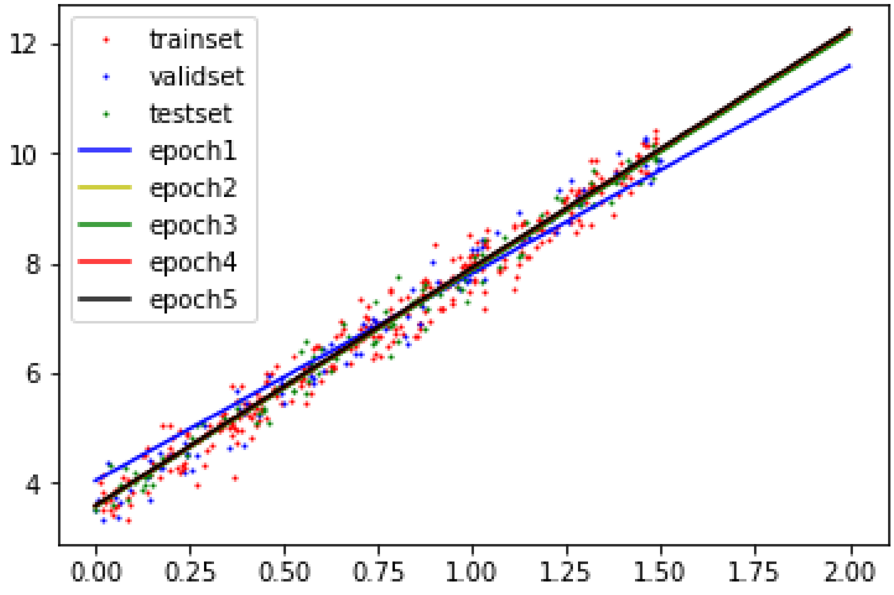
3.3

epoch=[3， 15， 20， 37， 40] generated by



```
import random
epoch=[random.randint(1,50),random.randint(1,50),random.randint(1,50),random.randint(1,50)]
```

	Step size	VMSE
2	1e-2	0.07374367663878252
3	1e-3	0.08426766147739163



The regression fit converges fast initially, and then adjust very little as the epoch ongoing.

Q4：

1.
    a. Sample mean makes no sense for the first 6 columns of dataset since they are attributes obviously irrelevant to the problem we interested in. However, filling the hole with sample mean is generally a good choice if we do so.
    b. We can also use Mode or median to filling the missing attributes.
    c. We can fill in the data with median/mean for some attributes do not fluctuate a lot and drops those features with too many missing attributes. This can improve the overall performance of our model since we don’t know much about how strong such missing attributes are related to the real distribution. If we fill some attributes, which may have a large variance, with mean, the performance might be compromised.
    d. File saved as “CandC-test1.csv” “CandC-test2.csv” “CandC-test3.csv” “CandC-test4.csv” “CandC-test5.csv”,

” CandC-train1.csv” ,” CandC-train2.csv” ,” CandC-train3.csv” ,” CandC-train4.csv” ,” CandC-train5.csv”.

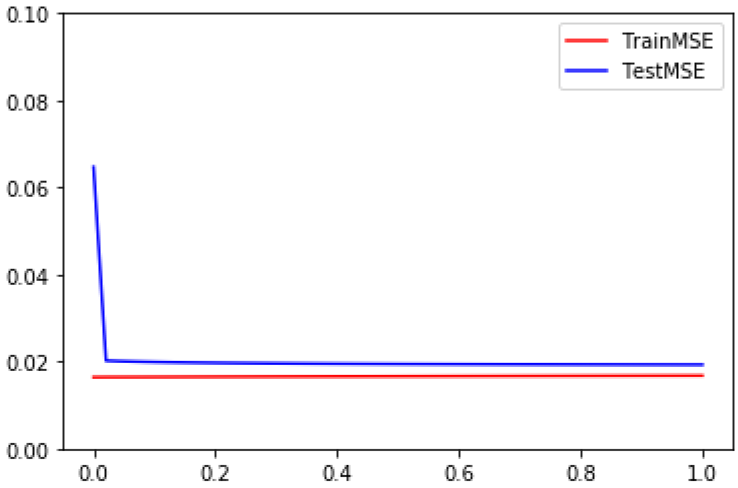
4.2

Result saved in “Assignment1_260631276_4_2.txt”

4.3.a

MSE of the best fit achieved on test data, averaged over 5 different 80-20 splits, along with the parameters learnt for each of the five models is saved in “Assignment1_260631276_4_3.txt”

4.3.b



The best lambda is 0.7142857142857142 with corresponding Validation MSE=0.03601346392210256.