CS 534 Machine Learning Homework 5

Zexi Yuan

Problem 1: ("Illustrating the "curse of dimensionality")

(a) The volume with radius a in d dimensions is

$$V = \frac{S \times a^d}{d}$$

so we have

$$V_a = \frac{S \times a^d}{d} \; and \; V_{a-\varepsilon} = \frac{S \times (a-\varepsilon)^d}{d}$$

Then the part of the volume which lies at values of the radius between $a - \varepsilon$ and a, where $0 < \varepsilon < a$, is given by $V_a - V_{a-\varepsilon}$

$$V_a - V_{a-\varepsilon} = \frac{S \times a^d}{d} - \frac{S \times (a-\varepsilon)^d}{d} = \frac{S \times a^d}{d} \times \left[1 - \left(1 - \frac{\varepsilon}{a}\right)^d\right]$$

so the fraction is $f = 1 - \left(1 - \frac{\varepsilon}{a}\right)^d$.

Because $0 < \varepsilon < a$, we know that $0 < 1 - \varepsilon/a < 1$, thus leading to $(1 - \varepsilon/a)^d \to 0$ when $d \to \infty$. Therefore, the fraction f tends to 1 as $d \to \infty$.

(b) see the Figure 1-1 below.

(c) As we can see from the plot on the left, we can conclude that

1. Given the fixed ε/a, the fraction f increases as d increases and tends to be 1 when d is large enough.

2. The amount of the increase of f by d depends on ε/a. In other words, given the fixed d, the fraction f increases as the ε/a increases.

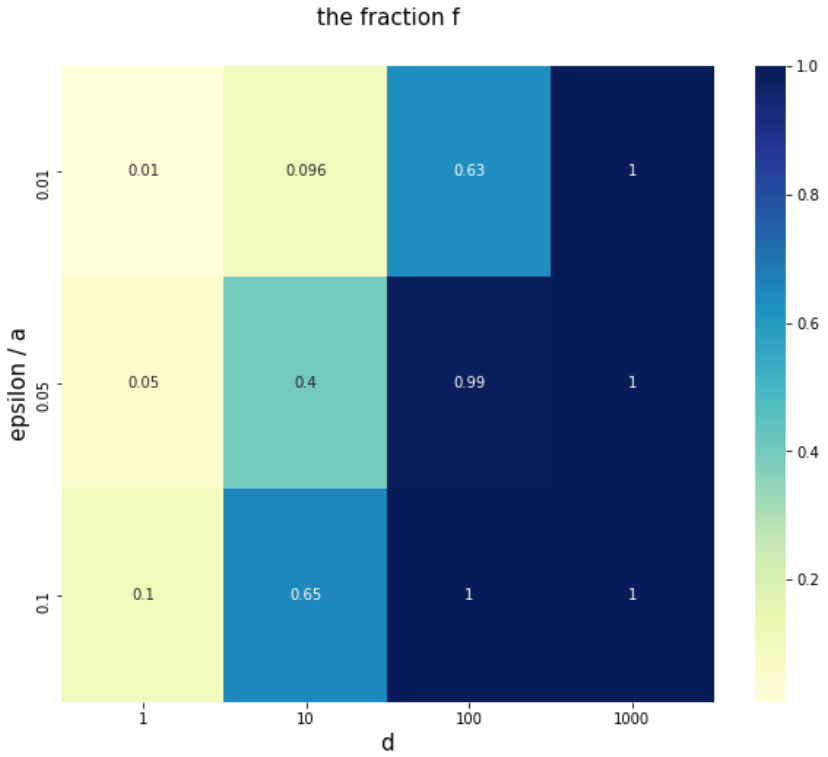

Figure 1-1

Problem 2: (PCA & NMF)

(a) remove missing values and ‘college name’ feature.

(b) As we can see from the Figure 2-1 below, we need 8 components to capture 95% of the variance in the data.

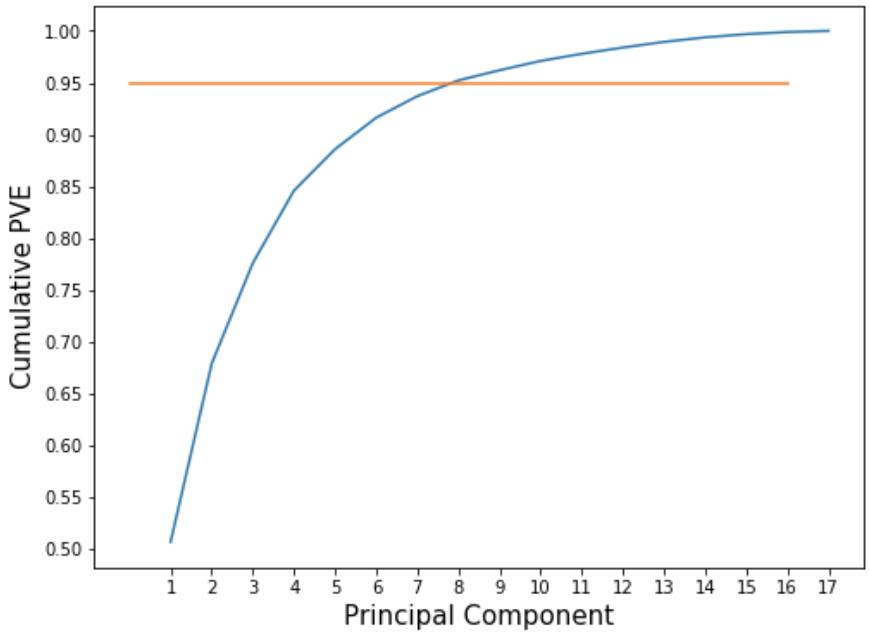

Figure 2-1

The first 3 principal components are characterized as shown in Table 2-1:

1. 'apps received', 'apps accepted', 'new stud enrolled', 'num FT undergrad' and 'num PT undergrad' have a great impact on PC1
2. '% new stud from top 10%', '% new stud from top 25%' , 'num PT undergrad', 'in-state tuition' and 'out-of-state tuition' have a great impact on PC2
3. 'num PT undergrad', 'in-state tuition' and 'add fees' have a great impact on PC3

	PC1	PC2	PC3
apps received	0.436000	-0.263735	-0.083196
apps accepted	0.446503	-0.146855	-0.091613
new stud enrolled	0.451651	-0.025573	-0.042923
% new stud from top 10%	0.039528	-0.483218	0.148270
% new stud from top 25%	0.034860	-0.287361	0.089271
num FT undergrad	0.470225	0.026752	-0.004117
num PT undergrad	0.370320	0.390176	0.589736
in-state tuition	-0.110154	-0.414996	0.241611
out-of-state tuition	-0.033458	-0.332763	0.164392
room	0.014474	-0.173355	0.053042
board	-0.000924	-0.143163	0.093912
add fees	0.141021	-0.036254	-0.704692
est book costs	0.016077	-0.026770	0.037006
est personal costs	0.091634	0.161335	0.031843
% fac with PHD	0.045811	-0.121150	0.036914
stud:fac ratio	0.037640	0.150303	-0.085630
graduation rate	-0.001695	-0.200914	0.019537

Table 2-1

(c) Reasons for normalization before PCA

Because the variables in this data have highly different scales, we need to ensure that these variables have the close or even same scale to reduce the influence to build a proper primary component. Therefore, we need to normalize/standardize the data.

(d) The squared error of NMF is 1.24350389669 and the squared error of PCA is 20.2265921356.

The first 3 principal components of NMF are characterized as shown in Table 2-2:

1. '% new stud from top 10%','% new stud from top 25%', 'in-state tuition', 'out-of-state tuition', 'room', 'board', 'est book costs', '% fac with PHD' and 'graduation rate' have a great impact on PC1
2. 'apps received', 'apps accepted', 'new stud enrolled' and 'num FT undergrad' have a great impact on PC2
3. 'num PT undergrad' has a great impact on PC3

	PC1	PC2	PC3
apps received	0.053208	0.716041	0.000000
apps accepted	0.037936	0.691417	0.076842
new stud enrolled	0.025563	0.599177	0.228694
% new stud from top 10%	0.508607	0.182249	0.000000
% new stud from top 25%	0.536489	0.173748	0.038698
num FT undergrad	0.000000	0.571126	0.283220
num PT undergrad	0.000000	0.000000	0.836956
in-state tuition	0.615202	0.000000	0.000000
out-of-state tuition	0.590044	0.088499	0.009910
room	0.540823	0.108522	0.107720
board	0.555210	0.065595	0.143757
add fees	0.305525	0.326776	0.018175
est book costs	0.519241	0.047773	0.216325
est personal costs	0.397997	0.036473	0.382007
% fac with PHD	0.518612	0.129140	0.156079
stud:fac ratio	0.462399	0.063118	0.277415
graduation rate	0.559634	0.124077	0.057734

Table 2-2

(e) By using the two methods, there are three conclusions.

1. PCA's implementation is easier than NMF's becasue PCA only needs the SVD to obtain the components but NMF need much time to search the optimal solution.

2. PCA's component values may be negative but NMF's are all positive so that NMF's interpretability is better.

3. NMF's squared error is lower than PCA's, so NMF does not change the original data too much.

Problem 3: (Simulated Neural Network)

(a)

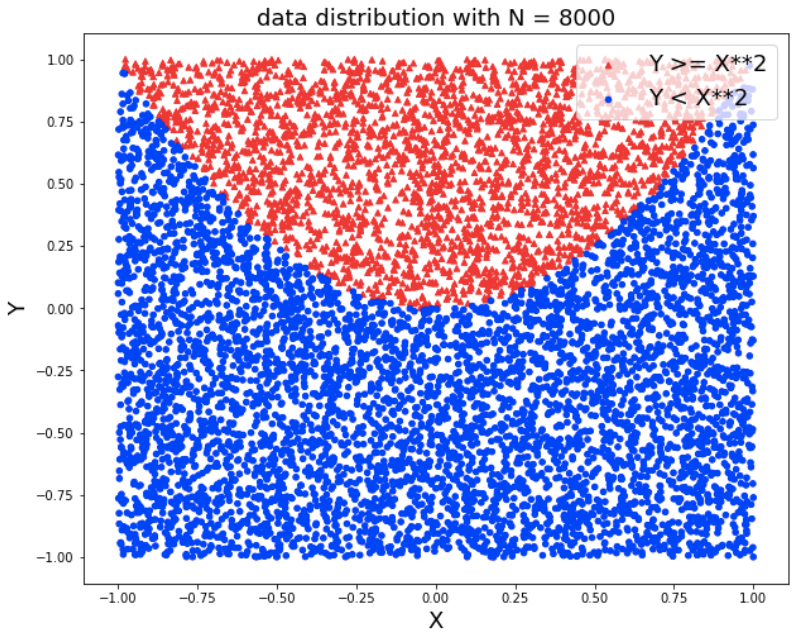

Figure 3-1

1. As is shown on Figure 3-1, because this problem is to learn y = square(x), the problem space is 2 dimensions. That means that we do not need huge amount of data for this problem space. Of course, the more the data is given, the better the performance will be. However, more data leads to more time to train. Therefore, I think if the given samples can support me to solve this problem, or build a decision boundary, that is enough.
2. As we can see from the plot above, 8000 samples are enough to fill this space and the decision boundary is clear.
3. In addition, to improve computation efficiency and avoid the "vanishing gradient" problem, I plan to create samples according to Gaussian distribution and normalize the data set to zero-mean and standard variation.
4. Because I will split the data set, so I prefer to create 10000 (8000 train and 2000 test) samples.

(b) First, I consider a 1-hidden layer neural network and optimize the parameters. Then, I found that the optimal number of neurons in one hidden layer is 19 and the train-validation score matrix is as shown in Table 3-1.

Table 3-1

	Misclassification Error	F1 Score	F2 Score
1 hidden layer (19)	0.007641	0.863581	0.863637

Second, I consider a 2-hidden layers neural network and optimize the parameters. Then, I found that the optimal number of neurons in two hidden layers is (27, 1) and the train-validation score matrix is as shown in Table 3-2.

Table 3-2

	Misclassification Error	F1 Score	F2 Score
2 hidden layer (12,5)	0.006003	0.866596	0.864752

According to the analysis above, I prefer to use the 1-layers neural network with 19 nodes because its accuracy, f1 and f2 score are slightly worse than 2-layer neural network but it is simple. I think the simple model is better and enough if it can reach a good performance.

(c) The test score matrix is shown in Table 3-3.

Table 3-3

	Misclassification Error	F1 Score	F2 Score
1 hidden layer (19)	0.009	0.987161	0.988007

As we can see, the neural network(NN) with only 1 hidden layer perform well on this data set, so I think I do not need more complex model like 2-hidden layer NN to learn. Its decision boundary is shown in Figure 3-1.

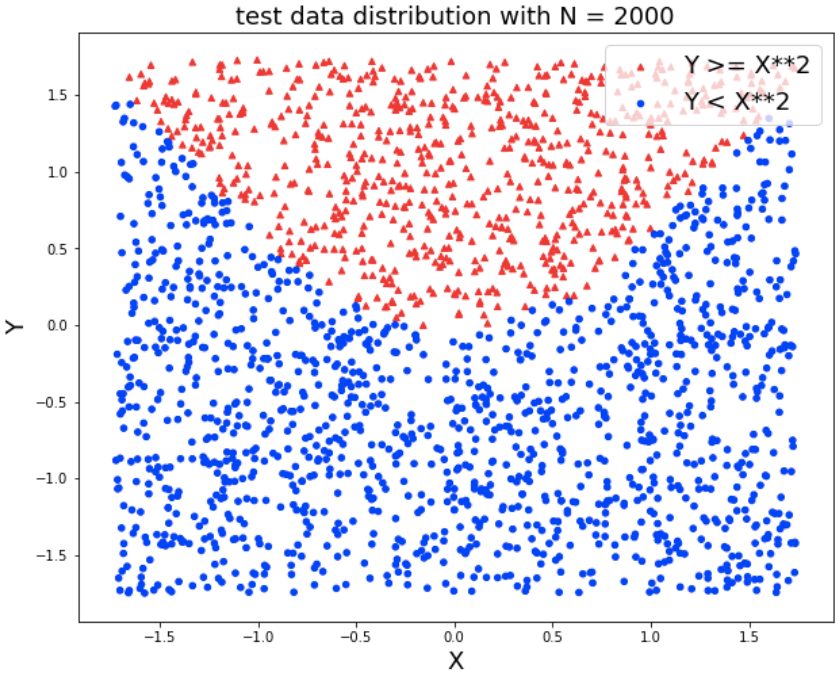

Figure 3-1

Problem 4: (Thyroid Prediction with Neural Networks)

(a) As you can see from the Table 4-1 below,

1. Missing values.

There are missing values in data set. We handle the missing values according to their typye: (1) using the mean to fill 'float' missing values; (2) using the mode to fill the 'object' value. Because original data are object values, we transform the object vaues to binary values or multinomial values for convenient computations. The original class are multinomial class values, but here we only care about the 'hyperthyproid' and 'negative'. So we transform multinomial class values to binary values such that '1' is 'hyperthyproid' and '0' is others classes including 'negative'.

Table 4-1

	percentage of missing values	data type
age	0.035714	float64
sex	3.928571	object
TSH	10.142857	float64
T3	20.892857	float64
TT4	6.571429	float64
T4U	10.607143	float64
FTI	10.535714	float64

2. Features selection.

Most object type values are binary type except for 'refSource'. So 'refSource' is transformed to a multinomial feature and others are transformed to binary features. In addition, the feature 'TBGInd' is full of single value such that it does not have influence on classification, so it can be removed. In addition, I investigate the correlations among the features. As is shown in Figure 4-1, 'T4UInd' and 'FTIInd' are highly correlated, so we have to drop one of them.

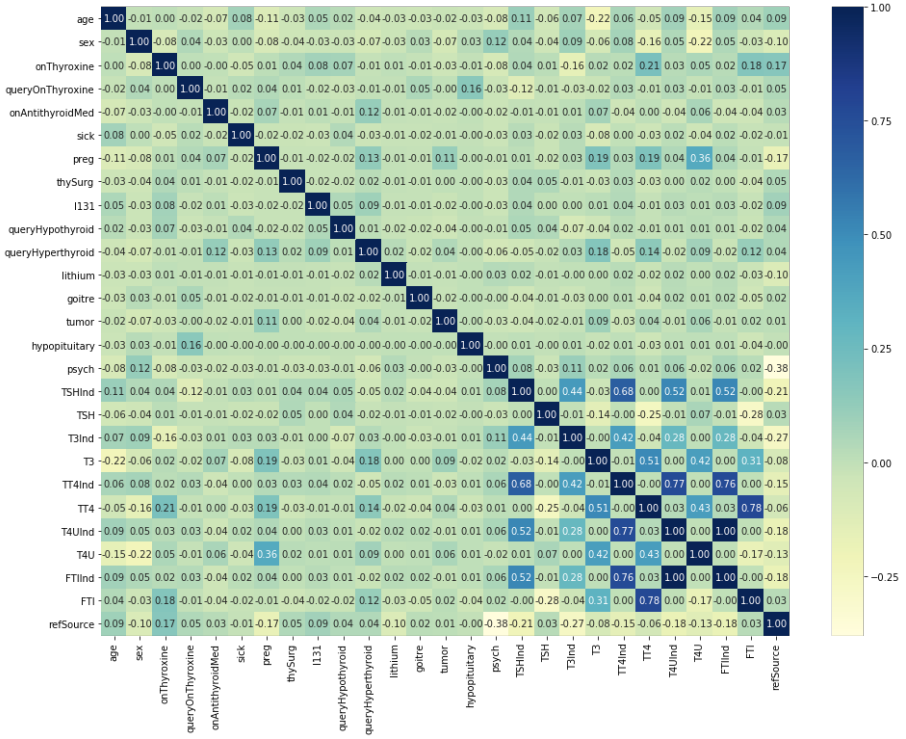

Figure 4-1

3. Standardizing the data. For fast computation and avoiding "vanishing gradient", the data is standardized by zero-mean and unit variance.

(b) The percentage of class = "hyperthyroid" in the data is 2.21428571429 %, which means this data set is highly imbalanced. Because of the imbalanced data set, I prefer to use F2 score as the main assessment criterion. This data set only has 2.21% data of positive class so that the models are easily to be overfitting on negative class such that F score cannot reach higher. Therefore, because I want more true positives, the models are required to focus on F2 score with high weighted recall instead of accuracy and F1 score which in parallel weights the precision and recall.

Neural Network

Because exploring the optimal parameter takes too much time, I run this part code on the 'Colfax' and obtain the optimal parameters. The optimal model I found is a neural network with 2-hidden layers and the nodes is (20, 16). The optimal learning rate is 0.000429193426013.

(c) Evaluation

1. The score matrix of this model is shown in Figure 4-2.

	Misclassification	F1 score	F2 score
train	0.006324	0.823555	0.830495
test	0.015714	0.620690	0.661765

Figure 4-2

2. Comparison with other models.

Train-validation score comparison is shown in Figure 4-3.

	Misclassification	F1 score	F2 score
Neural Network	0.006324	0.823555	0.830495
Random Forest	0.006176	0.814160	0.774567
Linear SVM	0.009821	0.719019	0.728457
Polynomial SVM	0.008185	0.756841	0.703664
RBF SVM	0.010342	0.712142	0.711543

Figure 4-3

Test score comparison is shown in Figure 4-4.

	Misclassification	F1 score	F2 score
Neural Network	0.015714	0.620690	0.661765
Random Forest	0.017143	0.571429	0.625000
Linear SVM	0.017143	0.666667	0.625000
Polynomial SVM	0.018571	0.518519	0.583333
RBF SVM	0.017143	0.600000	0.625000

Figure 4-4

3. As we can see above, for my data set, the neural network with its optimal hyperparameters has the best performance among these models.

(1) First, there is an interesting thing here that some models's best F2 scores and misclassification errors are the same. I think this phenomenon results from the data split, I tried to split the data with

different proportion and then they were different. However, because at the beginning I optimized my NN model with this kind of data split and it will spend much time to optimize it again, I plan to keep this result table even though it looks like weird.

(2) According to score matrics, it is obvious that the performance of Neural Network(NN) is better than others'. Given the subtle difference on misclassification error, the F2 score and F1 score of NN is higher than that of other models.

(3) According to the computational complexity, the computational cost of NN is the highest one. Because of its large amount of parameters like weights to train, NN's performance is better than others', but its computational complexity is much higher than others. NN need more time to train the model or networks.

(4) According to numbers of parameters, the NN has the most numbers of parameters to tweak, such as 'learning rates', 'activate functions' and especially 'hidden layers' that has various combinations to consider. Random Forest also has many parameters to tweak, such as numbers of estimators, decision criterion, max_depth and min_sample_leafs. SVM models have several important parameters to tweak, such as penalty parameter, kernel type and other parameter related to different kernels.