1. Illustrating the "Curse of Dimensionality"

(a) Let $V_1 = \frac{S\times a^d}{d}$ and $V_2 = \frac{S\times(a-\epsilon)^d}{d} = \frac{S\times a^d\left(1-\frac{\epsilon}{a}\right)^d}{d}$, with $0 < \epsilon < a$.

Then, $V_1 - V_2 = \frac{S\times a^d}{d} \times \left(1 - \left(1 - \frac{\epsilon}{a}\right)^d\right)$.

The fraction f is: $f = \left(1 - \left(1 - \frac{\epsilon}{a}\right)^d\right)$.

Since $0 < \epsilon < a$, we can have $0 < \epsilon / a < 1$.

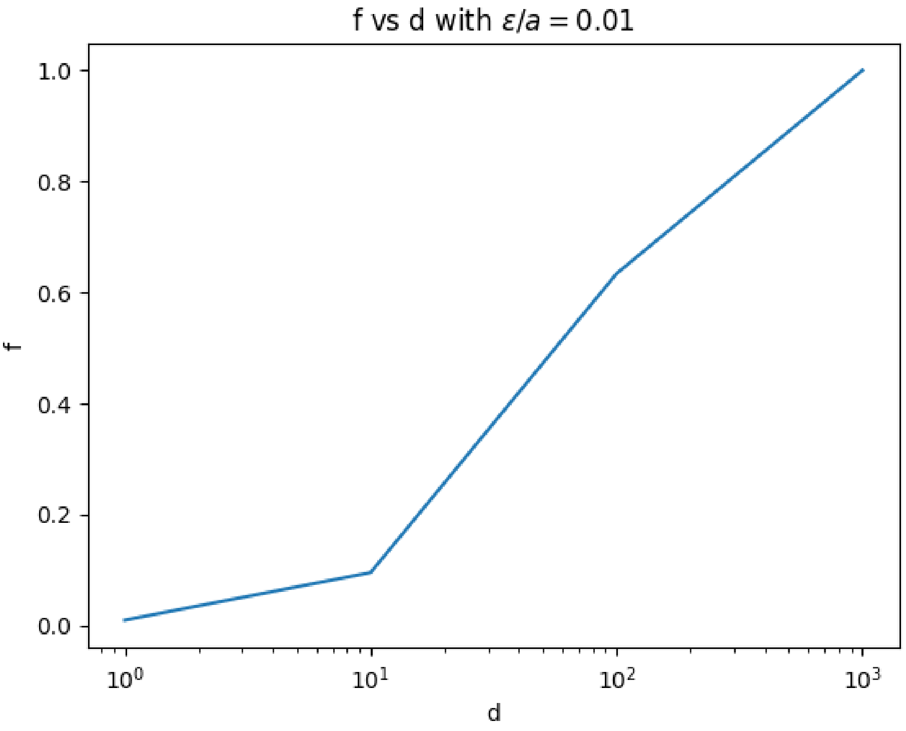
Therefore, for any fixed ϵ (no matter how small), $0 < 1 - \frac{\epsilon}{a} < 1$.

As a result, $\lim_{d\to\infty} \left(1 - \frac{\epsilon}{a}\right)^d = 0$.

Thus, $\lim_{d\to\infty} f = \lim_{d\to\infty} \left(1 - \left(1 - \frac{\epsilon}{a}\right)^d\right) = 1 - 0 = 1$.

(b) Figure 1 shows the values of ratio f with $\epsilon / a = 0.01$ and d = [1, 10, 100, 1000].

Figure 1



(c) From Figure 1, we can see that with a fixed ϵ/a, as d increases exponentially, the value of f quickly gets close to 1. This fit with statement that we just proved in 1(a) that for any fixed ϵ, $\lim_{d\to\infty} f = 1$.

2. Preprocessing Loan Defaults Using PCA & NMF

   (a) Similar to Homework 3 and Homework 4, I split the data into a train set and a test set with a train: test ratio of 80%: 20%. I also chose to use the grid search to find the optimal hyperparameters and use the default 5-fold cross validation in the grid search.

   (b) For "term", I removed "months" from every sample and only kept the numbers (i.e. "36 months" → "36").

   For "emp_length", similar to Homework 3 and Homework 4, I also removed "years" from every sample and only kept the numbers (i.e. "6 years" → "6"). There were some exceptions. I used "-1" to replace "n/a", "0" to replace "< 1 year", and "10" to replace "10+ years". However, since NMF requires non-negative values, I add 1 to every samples.

   For "earliest_cr_line", I kept the year only (i.e. "3/1/2000" → "2000").

   For "grade", "home_ownership", "verification_status", and "purpose", I simply used LabelEncoder from sklearn.preprocessing to transfer them to numerical values.

   (d) I used Spearman correlation to compute the correlation matrix of the features and discarded one feature from any pair of features with correlation score higher than 0.7. By removing highly correlated data, I can reduce the redundancy of the data, and improve the generalizability of the model. I also computed the correlation criteria of each feature with the label and selected the top 10 features. Removing the features that has lower correlation with the label helps ensure that the model would capture the key relation between the features and the label and reduce model's confusion on irrelevant features. Overall, feature selection can help reduce computational complexity and prevent overfitting.

   (e) PCA is sensitive to the scale of the features. Features with different scales are not comparable with each other, resulting in more weights on the features with larger scales. Normalization allows features to be on the same scale and thus contribute equally to the analysis.

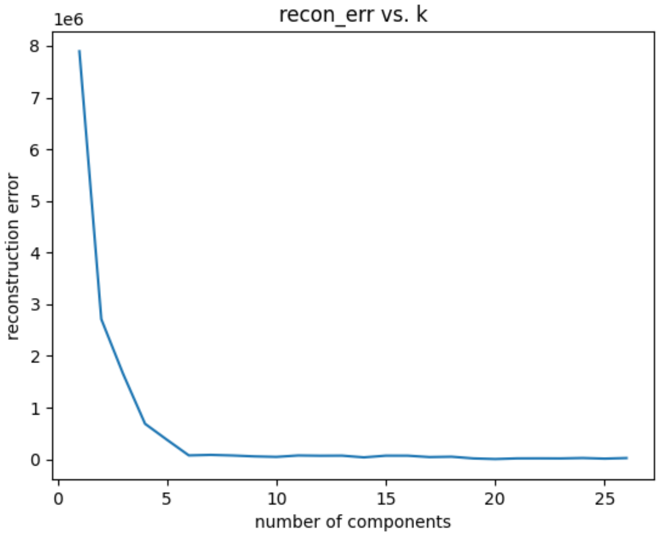
   (g) 20 components are needed to capture 95% of the variance in the normalized data. For the first 3 principal components, the top 3 important original features are:

   1st principal component: [17, 4, 1], ['revol_bal', 'installment', 'loan_amnt'];

   2nd principal component: [19, 24, 25], ['total_acc', 'tot_cur_bal', 'total_rev_hi_lim'];

   3rd principal component: [11, 15, 19], ['dti', 'open_acc', 'total_acc']

Figure 2



(j) The optimal number of components with the minimum reconstruction error is 20 (as shown in Figure 2).

For the first 3 factors of the NMF with optimal number of components, the top 3 important original features are:

1st factor: [8, 24, 0], ['annual_inc', 'tot_cur_bal', 'id']
2nd factor: [17, 8, 25], ['revol_bal', 'annual_inc', 'total_rev_hi_lim']
3rd factor: [13, 24, 8], ['earliest_cr_line', 'tot_cur_bal', 'annual_inc']

3. Model Bake-off for Predicting Loan Defaults

(f) Here is the table (Table 1) that shows the performances of each model on datasets with different preprocessing methods:

Table 1

model	preprocess	train-auc	train-f1	train-f2	val-auc	val-f1	val-f2	test-auc	test-f1	test-f2
logr	baseline	0.848	0.633	0.531	0.824	0.622	0.522	0.813	0.624	0.517
logr	fs	0.781	0.621	0.507	0.780	0.620	0.507	0.758	0.620	0.505
logr	pca	0.841	0.624	0.524	0.823	0.613	0.518	0.804	0.617	0.516
logr	nmf	0.776	0.574	0.465	0.766	0.573	0.463	0.718	0.571	0.502

model	preprocess	train-auc	train-f1	train-f2	val-auc	val-f1	val-f2	test-auc	test-f1	test-f2
dt	baseline	0.839	0.622	0.507	0.819	0.621	0.506	0.807	0.620	0.505
dt	fs	0.872	0.639	0.536	0.823	0.612	0.513	0.804	0.638	0.540
dt	pca	0.826	0.561	0.469	0.733	0.446	0.377	0.677	0.388	0.302
dt	nmf	0.769	0.599	0.482	0.749	0.595	0.480	0.720	0.606	0.490

model	preprocess	train-auc	train-f1	train-f2	val-auc	val-f1	val-f2	test-auc	test-f1	test-f2
rf	baseline	0.848	0.549	0.437	0.827	0.539	0.430	0.793	0.555	0.438
rf	fs	0.867	0.613	0.501	0.837	0.598	0.486	0.810	0.620	0.505
rf	pca	0.832	0.369	0.271	0.766	0.364	0.270	0.721	0.339	0.245
rf	nmf	0.798	0.554	0.438	0.756	0.541	0.425	0.732	0.611	0.495

model	preprocess	train-auc	train-f1	train-f2	val-auc	val-f1	val-f2	test-auc	test-f1	test-f2
svm	baseline	0.765	0.509	0.393	0.761	0.507	0.392	0.797	0.500	0.385
svm	fs	0.861	0.668	0.566	0.792	0.588	0.499	0.722	0.550	0.452
svm	pca	0.812	0.504	0.389	0.805	0.504	0.389	0.782	0.494	0.379
svm	nmf	0.790	0.298	0.210	0.762	0.291	0.205	0.715	0.200	0.135

model	preprocess	train-auc	train-f1	train-f2	val-auc	val-f1	val-f2	test-auc	test-f1	test-f2
nn	baseline	0.849	0.633	0.534	0.827	0.610	0.515	0.804	0.632	0.538
nn	fs	0.703	0.487	0.414	0.680	0.471	0.403	0.649	0.539	0.563
nn	pca	0.883	0.641	0.535	0.829	0.585	0.484	0.782	0.610	0.506
nn	nmf	0.804	0.496	0.393	0.726	0.448	0.356	0.551	0.029	0.019

(g) From Table 1, when only looking at the test performances, we can see that the logistic regression model has the best performance when trained on the dataset without dimension reduction. The decision tree model has the best AUC on the dataset without dimension reduction but has the best F1 and F2 when feature selection is used. The random forest model has the best performance on the dataset with feature selection. The Support Vector Machine (SVM) model has best AUC when no dimension reduction is applied and has best F1 and F2 on the dataset with feature selection. The neural network model has the best AUC and F1 on the dataset without dimension reduction and has the best F2 when feature selection is used.

If we compare all the models with their best test performances achieved, the logistic regression model has the highest AUC, the decision tree has the highest F1 score, and the neural network has the highest F2 score. One thing worth notice here is that I limit the max_iter for both SVM and neural networks, so their training processes might terminate before they reach convergence. Thus, the performances of the SVM and neural networks can be potentially improved if given enough time.

In terms of interpretability and computation time, the logistic regression (linear model) is generally highly interpretable and fast to train. The decision tree also has a high interpretability since we can visualize its decision path It is also fast to train, but the training time would increase if we increase the maximum depth. The random forest is more complex than the decision tree, so we would expect it to have less interpretability and longer training time than the decision tree. SVM has moderate interpretability when using linear kernel, but the interpretability would decrease if non-linear kernels are used. SVM also takes much longer time to train, especially on larger datasets. Neural networks generally are the least interpretable among the models we test, especially when we have more layers. It also takes longer time to train, and the training time also increases as the hidden layers increase.