Final Project

Feature Selection Challenge

Zhiqian Zhou

CMP 4720

May 11, 2018

1. Introduction

The NIPS 2003 challenge in feature selection consisted of five two-class classification problems. Each problem provided with testing and validation datasets. The datasets were chosen to span a variety of domains and were transformed in the form of Integer with many probes, which is addition of 'random' features distributed similarly to the real features.

I picked three of them to do my project. The first one is ARCENE, whose task is to distinguish cancer versus normal patterns from mass-spectrometric data. This is a two-class classification problem with continuous input variables and 10,000 features. The second one is DEXTER, the filter of 'corporate acquisition' texts. This one contains sparse continuous input variables with 20,000 features. And the last one is MADELON, which is artificial data with continuous input variables and 500 features. Over 90% of the data is probe.

Table 1: NIPS 2003 challenge datasets.

Dataset	**Domain**	**Type**	**#Feature**
ARCENE	Mass Spectrometry	Dense	10,000
DEXTER	Text classification	Sparse	20,000
MADELON	Artificial	Dense	500

The goal of this project is to classify the datasets using a minimal number of features, and to find out how important feature selection can be in the process of classification.

2. Method

Since all the datasets have very high dimensions, it's hard to visualize the data in our homework and feature selection seems to be very important in this case. Here are the methods I choose for the classification process to improve the accuracy.

Preprocessing

Standardization $x_{norm} = \frac{x-\mu}{\sigma}$

Feature Selection

Mutual Information $I(X;Y) = \sum_{x \epsilon X} \sum_{y \epsilon Y} p(x,y) log \frac{p(x,y)}{p(x)p(y)}$

PCA (remove features according to given percentage)

Classifier

KNN (non-parameter)

Naïve Bayes (Gaussian) $$P(A|B) = \frac{P(A) \times P(B|A)}{P(B)}$$

Perceptron + Gradient Descent (Linear)

Different Classifiers were used after using the same methods to do the feature selection.

3. Process

MADELON

I started the classification with the dataset MADELON, which has least number of features. First, I plot out its mean (see Figure 1) and variance (see Figure 2).

In figure 1, the Y Label is mean, the X label is the number of features. The Y Label range from 0 to 500. In the figure 2, the Y Label change to variance and it range from 0 to 17500. Then I zoom in the area from 0 to 2000 in the figure 2 (see Figure 3), which helps me to see the variance of those features whose variance is relatively lower.

Therefore, I decided to remove those features whose variance is lower than 100 according to figure 3.

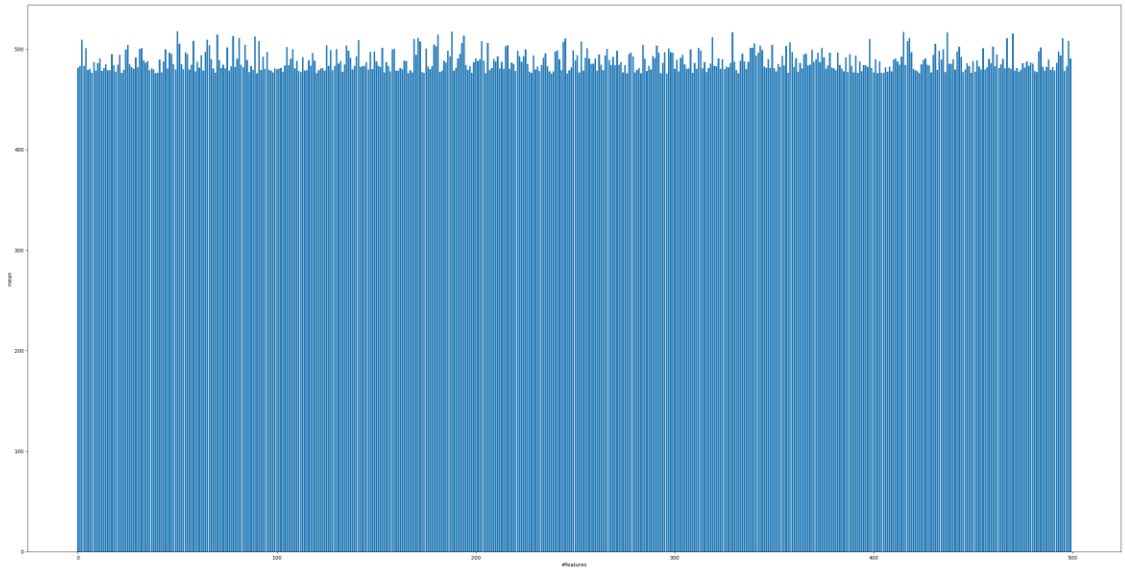

Figure 1: Mean of each feature in MADELON

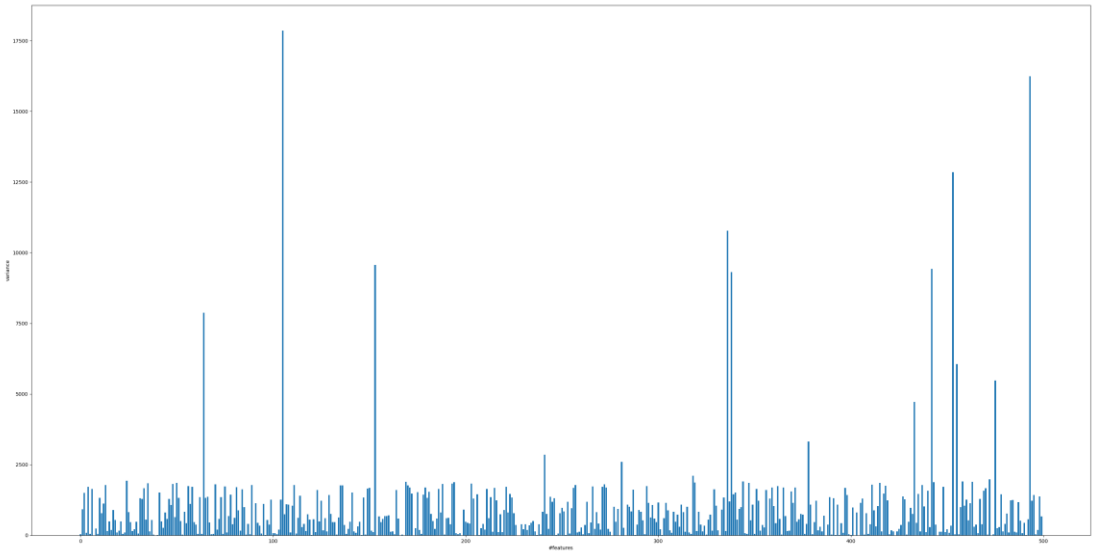

Figure 2: Variance of each feature in MADELON

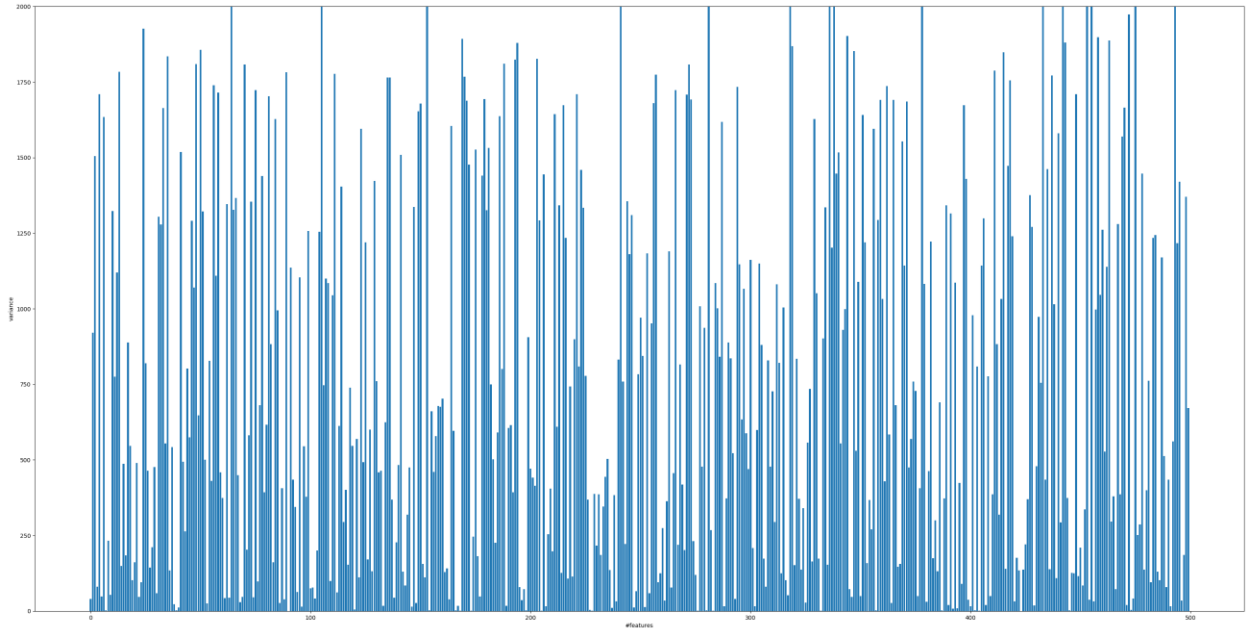

Figure 3: Variance from 0 to 2000 in MADLEON

After deciding the variance threshold, I standardize the data, apply PCA to reduce dimensions and then use three different classifiers to compare the result.

However, the accuracy was lower than 0.6 for all the classifiers (see Table 2).

Table 2: Result of MADELON

	KNN	Perceptron	Bayes
Accuracy	0.58-0.61	0.50-0.57	0.56-0.60

Then, I tried the original 500 features without any preprocessing and feature selection by accident. The result was better by using the KNN method (see Table 3).

Table 3: Result of MADELON using original data

	KNN	Perceptron	Bayes
Accuracy	0.76	0.50	0.59

It turns out that standardization of data will lose information in this case. After remove standardization from the process, I tried different parameters for feature selection and get the result (see Table 4).

Table 4: Result of MADELON without standardization

	KNN	Perceptron	Bayes
Accuracy	0.75-0.78	0.52-0.57	0.55-0.60

Table 4 shows that KNN has the best classification result in this case. To find out the relationship between the number of the features and the classification accuracy. I tried different value of parameters to record its accuracy and the feature numbers (see Table 5).

Table 5: Result of MADELON using KNN

Feature#	500	258	247	237	228	220	212	204	191
Accuracy	0.761	0.75	0.768	0,765	0.773	0.77	0.778	0.765	0.751

The result shows that reducing features can help increase the accuracy. However, after eliminating noise features, accuracy will not increase as features decreasing. Also, keep eliminating features might lose information. Also, in some cases, we can achieve good classification performance using all the features.

ARCENE

This dataset has 10,000 features. Same as the former one, I plot out its mean (see Figure 4) and variance (see Figure 5).

In this case, the mean and variance are kind of mess in the figure, so I apply mutual information instead of removing lower variance this time. I first standardization the data, then applied mutual information and PCA to reduce dimension, and use the three classifiers to compare the result.

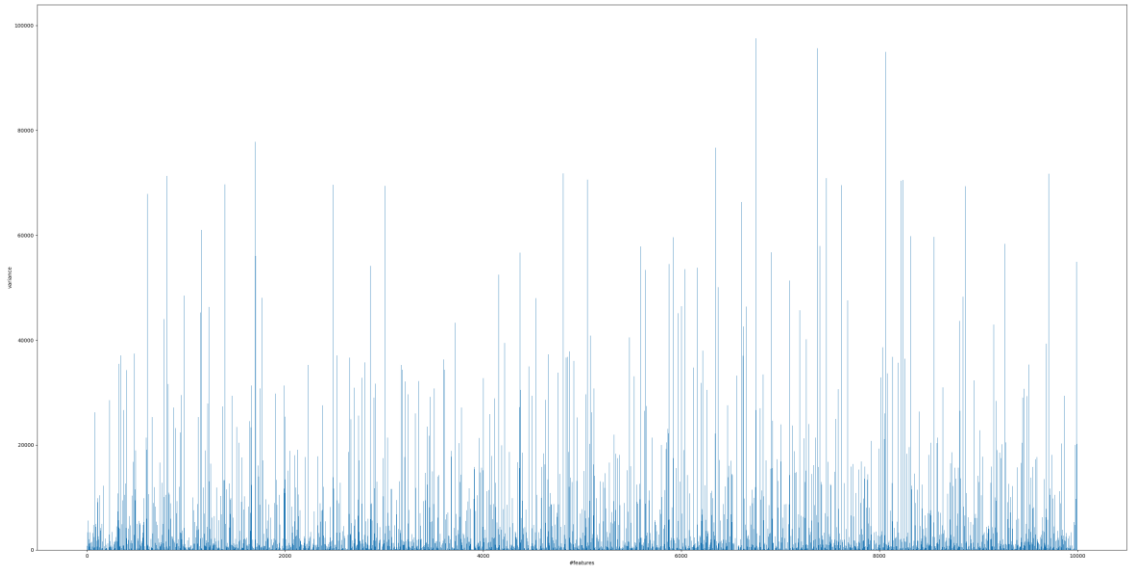

Figure 4: Mean of each feature in ARCENE

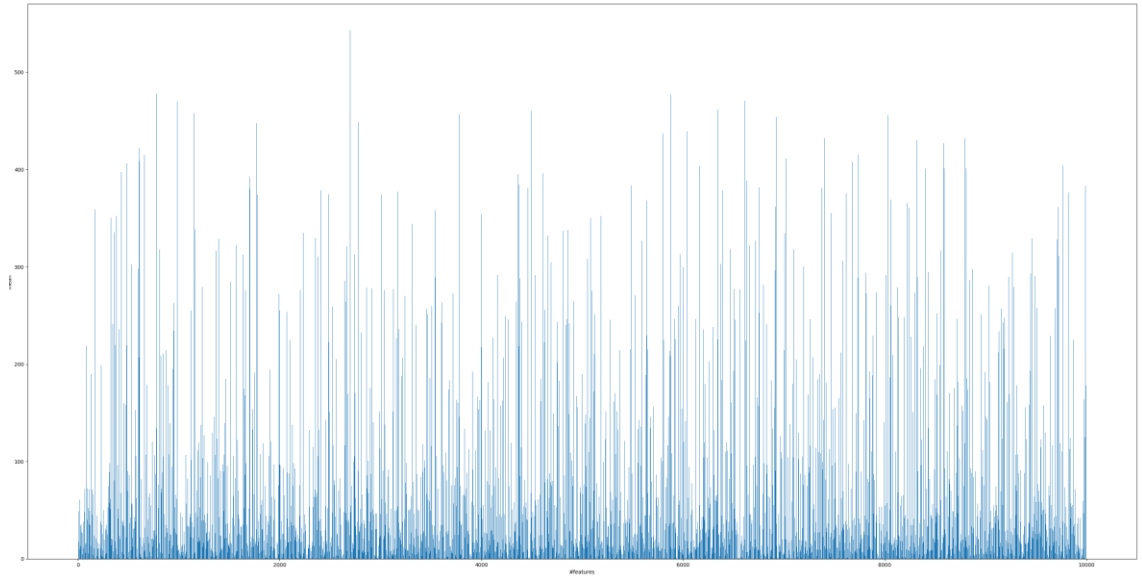

Figure 5: Variance of each feature in ARCENE

Different values of parameters were used to find the rough range of the accuracy (see Table 6).

Table 6: Result of ARCENE

	KNN	Perceptron	Bayes
Accuracy	0.61-0.64	0.77-0.87	0.60-0.64

I also tried the original 10,000 features in case that standardization might lose information (see Table 7). We can see that Perceptron has the best accuracy in this case.

Table 7: Result of ARCENE using original data

	KNN	Perceptron	Bayes
Accuracy	0.61	0.81	0.54

I picked the Perceptron to show the relationship between number of features and classification accuracy (see Table 8).

Table 8: Result of ARCENE using Perceptron

Feature#	10000	9500	81	78	75	72	69	66
Accuracy	0.81	0.81	0.81	0.81	0.82	0.84	0.83	0.77

The result also shows that reducing features can help increase the accuracy. However, after eliminating noise features, accuracy will not increase as features decreasing.

DEXTER

This dataset has 20,000 features, which contains the largest number features in the three datasets. This one is a little special as its data type is sparse. To put this clearly, I plot out its number of valid data of each feature (see Figure 6) and then zoom in the area from 0 to 50 (see Figure 7).

The datasets provided 300 training data. Thus, those features whose number of valid data is less than 10 should be removed.

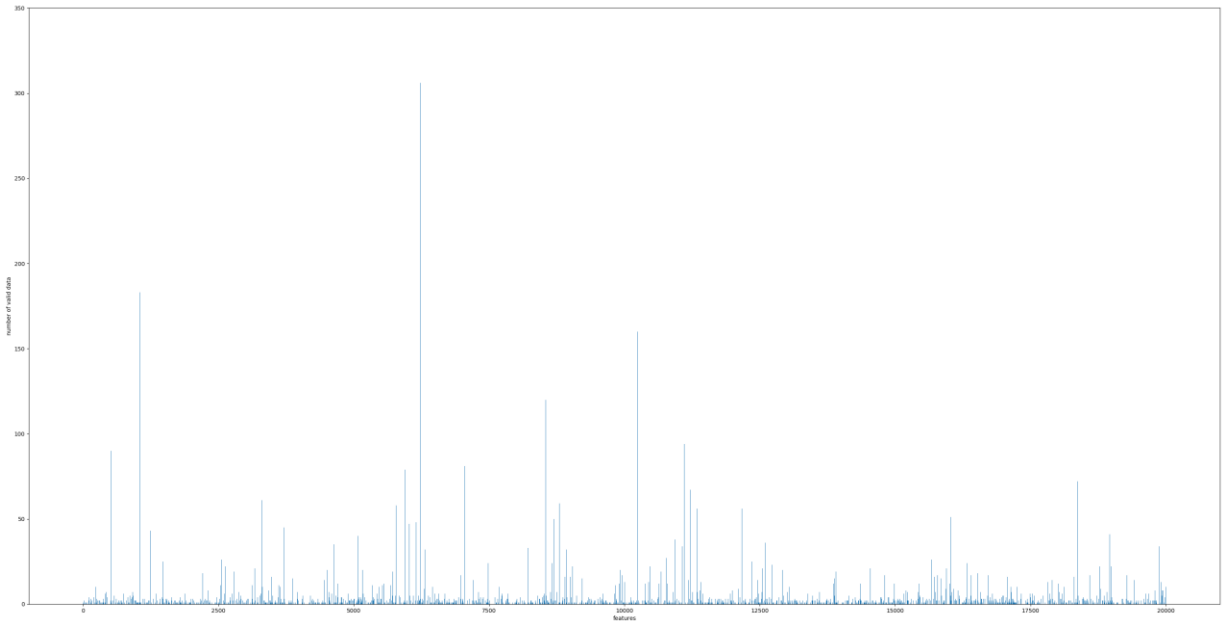

Figure 6: Number of valid data of each feature in DEXTER

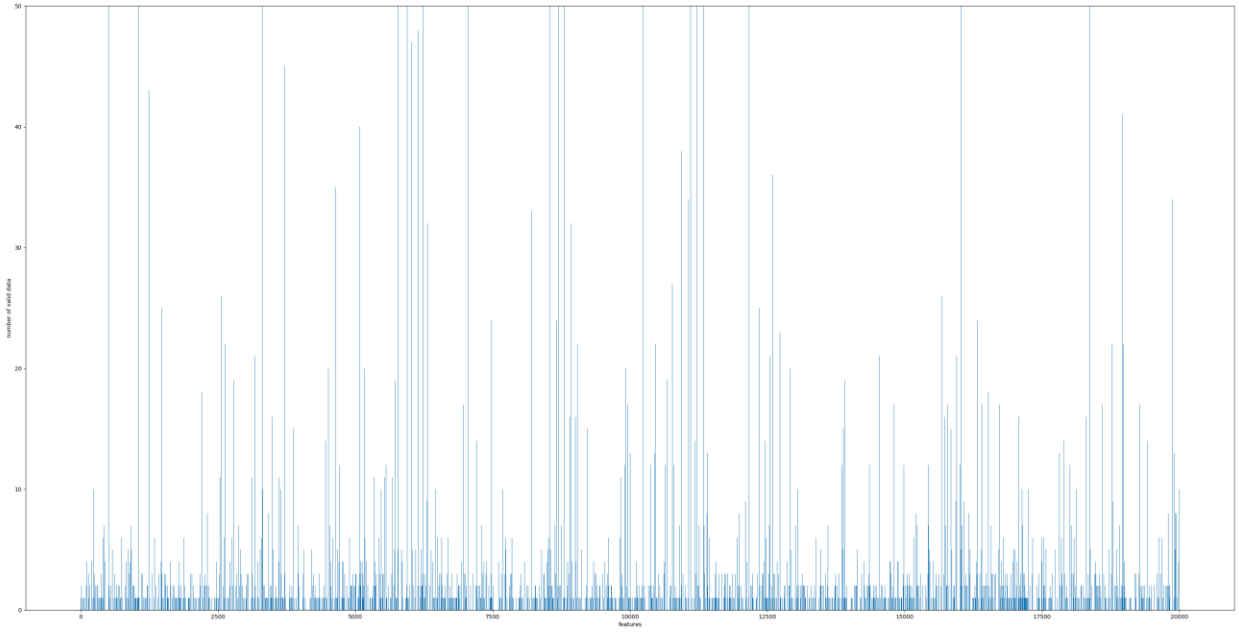

Figure 7: Number of valid data from 0 to 50 in DEXTER

I changed the value of threshold for the number of valid data to apply the feature selection. And after removing those features with less data, I replaced 0 with nan in the sparse matrix. The result is as follow (see Table 9).

Table 9: Result of DEXTER

	KNN	Perceptron	Bayes
Accuracy	0.59-0.72	0.84-0.91	0.86-0.89

It seems that Perceptron and Bayes both worked very well, so I compared the accuracy both classifiers with the number of features (see Table 10).

Table 10: Result of DEXTER using Perceptron and Bayes

Feature#	1045	851	722	625	539	485	427
Perceptron	0.867	0.907	0.9	0.893	0.88	0.85	0.85
Bayes	0.86	0.873	0.877	0.88	0.873	0.867	0.867

From the result, we can see that perceptron perform better than Bayes, but if keep reducing the dimension, Bayes seems to work better than Perceptron. I don't know if it happens by accident or it is related to their property.

Another thing is that, after eliminating the noise feature, feature selection cannot improve the accuracy much.

4. Conclusions and future work

- Standardization of data might lose information.
- In some cases, we can achieve good classification performance using all the features.

- Feature Selection will help increase the accuracy. However, after eliminating noise features, accuracy will not increase as features decreasing. Also, keep eliminating features might lose information.

In this project, some of the classification has a good result while some are not and all of them still contained probes after feature selection. I should consider of how to remove the probes which is very similar to the original features.

I also tried to apply feature selection into different types of data, i.e. the sparse one, ARCENE. It's very interesting to meet this kind of data and use different ways to do the feature selection. In the future, when encountering the practical problem, the situation can be more complicated. Therefore, training different kinds of data in ideal situation is a good way to start.