CS5785 Assignment1

Yue Wang yw986 Jingxuan SUN js3422

September 23, 2018

1 Digit Recognizer

1.1 Observe Data

There are 42000 training sample, each with a label and a feature dimension of 784, which is 28*28, the size of the digit image. There are in total 10 classes, corresponding to the digits 0 to 9. Each class contains approximately 4000 samples. The prior probability of the classes are not uniform. The numbers of samples from each class is plotted into a normalized histogram, which implies the probability distribution. Please refer to Part c in jupyter notebook for the distribution.
Training data is loaded into a data framework with pandas. I also transformed it into numpy array. Class zero and one are saved into separate arrays for later use. Please refer to Part a in jupyter notebook.

1.2 Display MNIST Digit

Each data point is obtained from one row of training set and then reshaped into a square matrix corresponding to the physical representation of the digit image. Each entry of the feature corresponding to the color of a pixel in the image. The images are plotted with pyplot. Please refer to Part b in jupyter notebook.

1.3 Best Match

For each data point in training set, the L2 distance between each of the rest training samples and itself are calculated, and the minimum distance is compared with and updated. The best match is the second nearest data point in the training set, because the nearest should be the examined point itself. Finally, the best match is returned with its index and distance to the examined data point. Please refer to Part d in jupyter notebook.

1.4 Genuines and Imposters

Genuine pairs refer to 0-0 and 1-1 pairs while imposters refer to 0-1 pairs. Thus, pairwise L2 distances are calculated from zero and one arrays. (It was very slow

using dataframe obtained from the first step, so I circumvented it with arrays.)
Distance distribution of genuines and imposters are plotted into a normalized histogram. Please refer to Part e in jupyter notebook.
The ROC curve is directly produced from sklearn library. For each distance threshold, FPs refer to imposter pairs whose distance is smaller than the threshold; TPs refer to genuine pairs whose distance is smaller than the threshold; FNs refer to genuine pairs whose distance is larger than the threshold; TNs refer to imposter pairs whose distance is larger than the threshold. FPR and TPR are then calculated for a continuous choices of distance threshold.
Equal error rate refers to the point where FPR + TPR = 1. The EER of a random classifier is 0.5. Please refer to Part f in jupyter notebook.

1.5 KNN Classifier

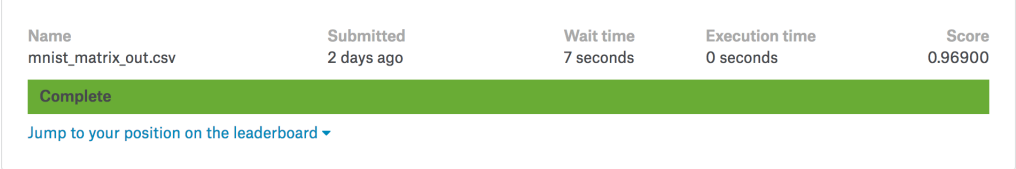
To implement a KNN classifier actually does not need to train a model (lazy learning). The testing data is fed to the classifier as a matrix and pairwise distances between each of the testing sample and all the training sample are calculated. The output distance matrix is of M*N dimension, where M is the size of training set and N is the size of testing set. Each row refers to one test data point, in which each entry is the distance between the test point and each training point. (External library is used for the sake of speed. Original implementation is commented out.)
For each test data point, the distance (one row) is sorted and the smallest k distances are chosen and the label of corresponding training data point are voted upon. The label with the most votes is the classification result. Please refer to Part g in jupyter notebook.

1.6 Cross Validation

As the training set is already randomized with respect to the labels, I directly sliced it into 3 subsets. Each cross validation uses one subset as the validation set and the other two as the training set. The validation set without labels is fed to the KNN classifier defined in the former steps. We chose neighbourhood size k as 5. Predicted labels are obtained and then compared with the validation set ground truth labels. Accuracy is calculated in each cross validation step, as well as the average accuracy for 3 cross validation steps. We achieved an average accuracy of 0.966, which is considered to be satisfiable. Please refer to Part h in jupyter notebook.
Normalized confusion matrix is generated through sklearn. For each class, predicted labels are put into buckets and the proportion is calculated. The correct predictions are on the diagonal. Digit "8" is the most tricky one to classify according to the confusion matrix. Please refer to Part i in jupyter notebook.

1.7 Prediction on Test Set

Test set is loaded and without labels, and it is fed to KNN classifier. The predictions are saved into csv file and submitted to Kaggle. The final accuracy is 0.969. Please refer to Part j in jupyter notebook.



2 The Titanic Disaster

2.1 Data Description

The train dataset has 891 rows and 12 columns, including features: PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch(Parent and children), Ticket, Fare, Cabin, Embarked.
We could describe the features in the following categories:
- Categorical: Survived, Sex, Embarked
- Ordinal: Pclass
- Continuous Numeric: Age, Fare
- Discrete Numeric: SibSp, Parch

2.2 Data Pre-processing

By roughly understanding the features, we can drop the "PassengerId" first. Secondly, we need to take a look at if we have any missing values in these features, but how to fill those features depends. There are two strategy we took to fill up the missing values:
(1) filling with average value in train / test dataset.
(2) getting random values based on original dataset's mean and standard deviation. For the latter one, it is used in continuous features, for example 'Age'.

2.3 Feature and Label Relation

2.3.1 Pclass

There is 3 different outputs of ticket class, which is 1 – first class, 2 – second class, 3 – third class. From the matrix below, we can easily see the this feature have a lot of impact on "Survival". So we will keep it.

	Pclass	Survived
0	1	0.629630
1	2	0.472826
2	3	0.242363

2.3.2 Sex

Also as presented in the following matrix, we could see the influence of sex on survival rate. Further, we need to transform the categorical feature into numeric, by mapping female to 1, male to 0.

	Sex	Survived
0	female	0.742038
1	male	0.188908

2.3.3 Parch and Sibsp

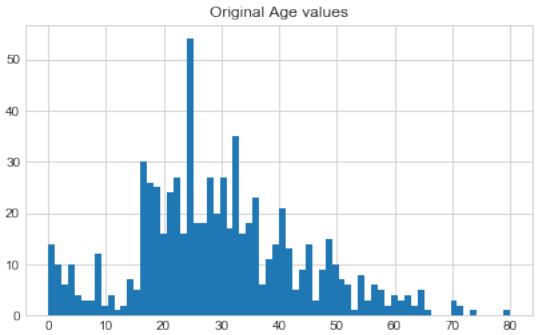
For these two features, they are amost telling the same. However, the reason we do not merge them together is that the rate of survival is not linear with the number of family members on board.

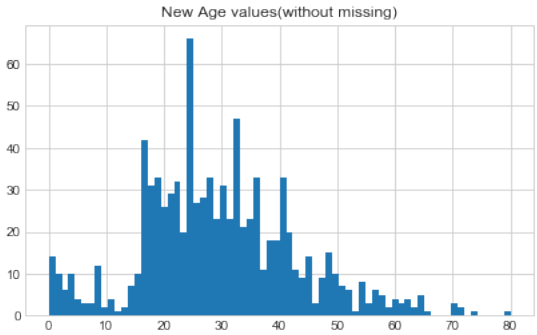
	SibSp	Survived
1	1	0.535885
2	2	0.464286
0	0	0.345395
3	3	0.250000
4	4	0.166667
5	5	0.000000
6	8	0.000000

	Parch	Survived
3	3	0.600000
1	1	0.550847
2	2	0.500000
0	0	0.343658
5	5	0.200000
4	4	0.000000
6	6	0.000000

2.3.4 Age

Since 'Age' is almost continuous, so we cannot group them directly with probability of survival. So we plot the distribution of age, also filling the missing values inside train and test dataset by strategy 2, using random values based on mean and standard deviation.





2.3.5 Name

The 'Name' attribute is tricky, it seems they are random and no logic, but after research deeper into it, the "title" inside this attribute also carry a lot information.
So we merge all the titles into 5 category, which could present their marriage, celebrity status, education background, etc. As presented in the right figure, this categories have great influence on probability of survival rather than sex itself. Thus, we drop the 'Name' attribute.

Sex	female	male
Title		
Capt	0	1
Col	0	2
Countess	1	0
Don	0	1
Dr	1	6
Jonkheer	0	1
Lady	1	0
Major	0	2
Master	0	40
Miss	182	0
Mlle	2	0
Mme	1	0
Mr	0	517
Mrs	125	0
Ms	1	0
Rev	0	6
Sir	0	1

	Title	Survived
0	Master	0.575000
1	Miss	0.702703
2	Mr	0.156673
3	Mrs	0.793651
4	Rare	0.347826

2.3.6 Fare

For 'Fare', it has missing value in test dataset, since it only have one, so we use strategy (1) mean of test dataset to fill in.
In order to find its relationship with 'Survival', we used range of fare to visualize.

	FareBand	Survived
0	(-0.001, 8.662]	0.198052
1	(8.662, 26.0]	0.402778
2	(26.0, 512.329]	0.559322

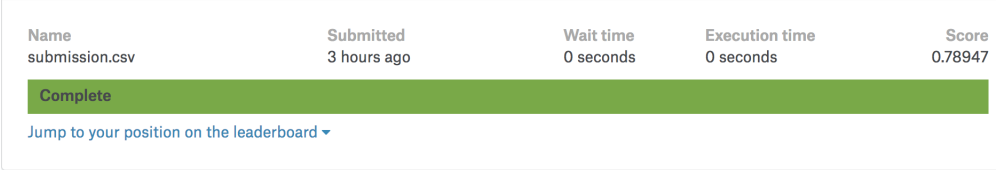
According to matrix above, we could almost found out that higher fare leads to higher probability of survival. Instead of using 'Fare', we transform the range of fare into discrete number. The logic for this is that, for ticket price from 0 to 8 dollar, there is no big difference. So the distance between fares would produce noises for classifying.

2.3.7 Others

For the attributes, we choose to drop 'Cabin' and 'Ticket'. For the first one is because it has too many missing values, and for latter one, it varies too much in different rows.

2.4 Final Result

For the final version of train and test dataset, we have 8 features. And applying logistic regression on it. For our train dataset, the accuracy is 0.8125701459034792. After uploading on Kaggle, the final result is 0.78947.



3 Variance of a sum

According to :

$$var[W] = E[W^2] - E[W]^2$$

Thus left part of equation:

$$\text{var}[X - Y] = \text{E}[(X - Y)^2] - \text{E}[X - Y]^2$$

$$= \text{E}[X^2 - 2XY + Y^2] - \text{E}[X - Y]^2$$

Also, according to :

$$cov[X, Y] = E[(X - E[X])(Y - E[Y])]$$

which we could get:

$$cov[X, Y] = \mathrm{E}[XY - X\mathrm{E}[Y] - Y\mathrm{E}[X] + \mathrm{E}[X]\mathrm{E}[Y]]$$

$$= \mathrm{E}[XY - \mathrm{E}[X]\mathrm{E}[Y] - \mathrm{E}[Y]\mathrm{E}[X] + \mathrm{E}[X]\mathrm{E}[Y]$$

$$= E[XY] - E[X]E[Y]$$

So, for right part of equation:

$$\mathrm{var}[X] + \mathrm{var}[Y] - 2\mathrm{cov}[X, Y]$$

$$= \mathrm{E}[X^2] - \mathrm{E}[X]^2 + \mathrm{E}[Y^2] - \mathrm{E}[Y]^2 - 2[E[XY] - E[X]E[Y]]$$

$$= E[X^2] + E[Y^2] - 2E[XY] - [E[X]^2 + E[Y]^2 - 2E[X]E[Y]]$$

Since:

$$E[X]^2 + E[Y]^2 - 2E[X]E[Y] = [E[X] - E[Y]]^2 = E[X - Y]^2$$

So, for right part:

$$\mathrm{var}[X] + \mathrm{var}[Y] - 2\mathrm{cov}[X, Y]$$

$$= E[X^2] + E[Y^2] - 2E[XY] - E[X - Y]^2$$

Finally, left = right,

$$\mathrm{var}[X - Y] = \mathrm{var}[X] + \mathrm{var}[Y] - 2\mathrm{cov}[X, Y]$$

4 Bayes rule for quality control

Given:
$\mathrm{P}(+ \mid defective) = 0.95(1)$
$P(- \mid non - defective) = 0.95(2)$
$total = 10000000(3)$
$P(defective) = 1/100000(5)$

4.1 a

Find: P(defective | +)
From given conditions (4):
$P(non - defective) = 99999/100000$

According to Bayes Theory:
$P(defective \mid +)$

$= P(+ \mid defective) * P(defective)/P(+)$
$= 0.95 * 1/100000/P(+)(5)$

$P(non - defective \mid +)$
$= P(+ \mid non - defective) * P(non - defective)/P(+)$
$= 0.05 * 99999/100000/P(+)(6)$

$P(defective) + P(non - defective) = 1(7)$

From (5)(6)(7):$P(+) = 0.95 * 1/100000 + 99999/100000 * 0.05 = 0.05$
$P(defective \mid +) = 0.95 * 1/100000/0.05 = 0.000019$

4.2 b

Find: P(non-defective $\mid +$), $P(defective \mid -)$
$P(non - defective \mid +)$
$= 1 - P(defective \mid +)$
$= 1 - 0.000019 = 0.999981$

$P(defective \mid -)$
$= P(- \mid defective) * P(defective)/P(-)$
$= 0.05 * 1/100000/(1 - 0.05) = 0.00000053$

number of non-defective given positive = 10000000*0.05*0.999981 = 499991
number of defective given negative = 10000000*0.95*0.00000053 = 5.

5 KNN

5.1 a

Choose of k:
When k = n, which means the test point will compare to all the data points in the data-set, including itself. So the prediction error would be 50%, since prior probability of each class is the same.
Also, for another corner case, when k = 1, the test point would only be compared to itself, so the accuracy is 100%, which means prediction error is 0.
Except for the two corner cases above, based on the majority votes, the prediction error should be lower when the k is approximately and slightly larger than the number of overlapping data points. This is because under this condition, when test point is in the area of overlapping, the majority of k-nearest neighbors of it would not be mislead by negative class.

5.2 b

Prediction error on held-on validation set:
Since the held-on set is randomly chose, when k is appropriate, the prediction

error wouldn't change too much on different test points.
However, when k is too small, the test point becomes extremely sensitive, thus the prediction error would change dramatically.

5.3 c

Computational requirement:
For f-fold cross-validation, f times of validation is carried out on n/f test data points. In each validation, each test point is compared with (f-1)*n/f training points. Therefore, time complexity is $O(n^2)$.

Validation accuracy:
It's a variance-bias trade-off.
For large f, each test set can see a great proportion of the whole data set. One extreme case is leave-one-out cross-validation, in which there is only one test sample and it is compared to all other data points. In such case, the classification result has higher bias and lower variance.
For small f, each test set can only see a limited proportion of the whole data set. In 2-fold cross-validation, each test sample is compared to only half of the training points, which will result in higher variance and lower bias.
The choice of f should be varied according to specific characters of different data sets.

5.4 d

A modification to KNN:
There are two conditions we need to consider to improve, first is overlapping, another is outlier.
First thought of improvement is to give every distance a proper weight, which is not equal, and also, for those data points far from its class, they should have less weight thus not effect other points. Also, for overlapping area, when test point belong to class A, the surrounding neighbors which are also from class A should have more weight so that we could have higher accuracy.
So, our final solution is finding the mass of different class. And the weight of each distance from train point to test point is 1/(dis from train to mass).

5.5 e

For pairwise comparison, increasing feature dimension will cause an exponential increase in distance computation.

For feature space, as feature dimension increases, data become sparser near the origin. It would be easier to fit a hyperplane to separate different classes of data points more easily and "exactly", which would cause overfitting. Then, it is necessary to increase the number of training samples exponentially too.