Machine Learning Assignment 1

GANSHENG TAN

October 18, 2018

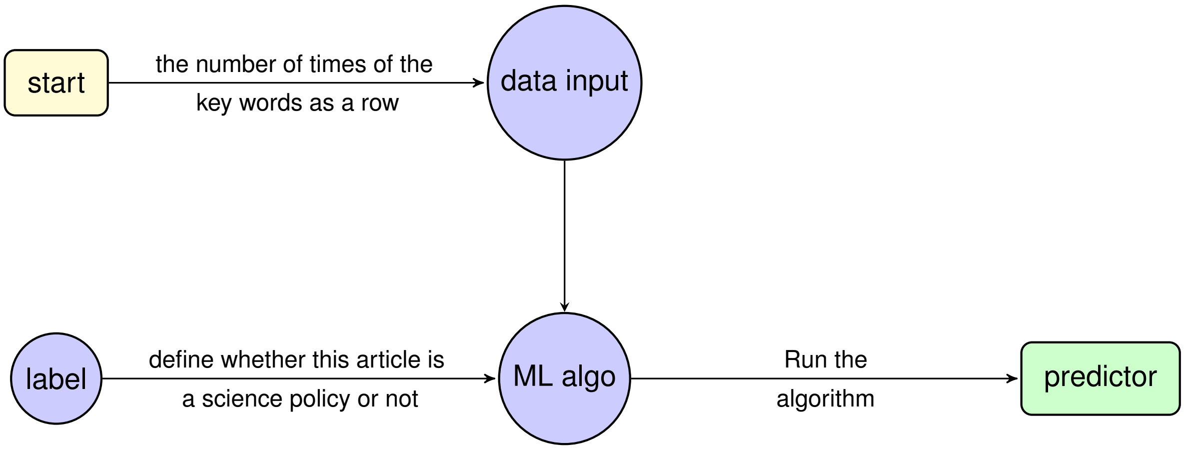
I The Learning Problem

Questions 1. (b)
Mainly there are three types of machine learning strategies. The supervised, unsupervised and reinforcement strategies. The unsupervised strategy is used to data clustering, the supervised strategy is used for classification, while the reinforcement learning is used to game playing and robot controlling. One should know that learning is used when human expertise does not exist or humans are unable to explain their expertise or complex solutions change in time

There exists some mathematical rules to determine a number is prime or not, so to speak, human expertise, so (a) can be solved by classical program. Newtons laws of motions show us the pattern exist, so (c) is as same as (a), the classical can give us an exact solution, while learning from data will lead us to an approximate solution. Humans are less able to identify the pattern of fraud, however, machine learning systems can analyze vast amounts of historical data to identify patterns associated with fraud, and certainly we have lots of hostorical data, so (b) can be a classification problem in machine learning.

As far as I am concerned, (d) can be done by machine learning, we can assume that if there are lots of car in one direction, we should increase the time of green light on this direction. So, it's more like a supervised problem. However, in the aspect of labeled data, we don't have much unless we do some experiment. Since the change of the green light time can cause a big problem, we shouldn't do it frequently.

Questions 2.
It's a supervised classification machine learning problem which is a problem where we are using data to predict which category something falls into. The information of one article will be stored in a row and the predictor is $\in \{0, 1\}$



Questions 3.

(a) No, it's not a ML problem because we can know the exact result by multiplying the volume of gas from the manual and the cost of gas.

(b) Yes, it's a supervised classification ML problem.The label Y, data matrix X, predictor vector θ is presented below

$$X = \begin{bmatrix} & feature \quad from \quad building \quad plan & parameter \quad from \quad engineer's \quad note \\ house1 & . & . \\ house2 & . & . \\ house3 & . & . \\ \vdots & \vdots & \vdots \end{bmatrix}$$

$$Y = \begin{bmatrix} heating \quad load \quad of \quad house1 \\ heating \quad load \quad of \quad house2 \\ heating \quad load \quad of \quad house3 \\ \vdots \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \vdots \end{bmatrix}$$

θ is used in $y^\star = \theta^T x^\star$ while x is a unknow house, ${x^\star}_i$ is its feature both from building plan and engineer's note

II Dimention Reduction

Questions 4.

(a) They are symmetric and real
Prove:

* symmetric: $(MM^T)^T = (M^T)^T M^T = MM^T$, same reason for $M^T M$
* square: one has the difinition of matrix product, hence the dimension of (MM^T) and $(M^T M)$ are respectively $m \times m$ and $n \times n$.
* real: Not necessary if M is a complex matrix, for example

$$\begin{bmatrix} 1\text{+i} & i \\ 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 1\text{+i} & 1 \\ i & 2 \end{bmatrix} = \begin{bmatrix} \text{2i-1} & \text{1-i} \\ 1+3i & 5 \end{bmatrix}$$

but it is true when M is real. That's what we will admit for the following questions

(b) We use SVD for matrix M

$$M_{m\times n} = U_{m\times r} S_{r\times r} V_{n\times r}^T$$

because$UU^T = I$, we have

$$M^T M = (USV^T)^T USV^T = VS^2V^T$$

we do the same to MM^T,We obtain

$$MM^T = USV^T(USV^T)^T = US^2U^T$$

Since $M^T M$ is a real symmetric matrix,, the eigenvalues are real and the eigenvectors can be chosen such that they are orthogonal to each other. Thus a real symmetric matrix $M^T M$ and MM^T can be decomposed as

$$\mathbf{M^T M} = \mathbf{Q\Lambda Q}^T$$
$$\mathbf{MM^T} = \mathbf{Q'\Lambda'(Q')}^T$$

where $\mathbf{Q}$ is an orthogonal matrix whose columns are the eigenvectors of M, and $\mathbf{\Lambda}$ is a diagonal matrix whose entries are the eigenvalues of $\mathbf{Q}$.

Then we notice that $\mathbf{\Lambda} = S^2 = \mathbf{\Lambda'}$

The eigenvectors are not necessary the same because the dimension of two eigenvectors one of which is of $M^T M$ and the other is of MM^T have different dimension.

(c) These expressions are written in answer (b)

(d) we can see in (b), the singular of a matrix $M_{m\times n}$ are the square roots of the eigenvalue of MM^T

Remark 1.
If M is a square invertible matrix, then we have a easier approach, yje cjaracteristic polynomial of MM^T: $p_{MM^T}(t) = \det\left(MM^T - tI\right) = \det(M)\det\left(MM^T - tI\right)\det\left(M^{-1}\right) = p_{M^T M}(t)$
Since they have same characteristic polynomial, they have same spectrum. In addition, we can extend this result to square mareix since invertible matrices are dense in square matrices, and the characteristic polynomial depends continuously on matrix.

Questions 5.

(a) the first principal axis is $(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})^T$ by calulating the eigenvector of MM^T, where M is

$$\begin{Bmatrix} -1 & -1 \\ 0 & 0 \\ 1 & 1 \end{Bmatrix}$$

(b) The mathematical way of projection is to the scalar product of sample vector $x_1{}^T = (-1, -1)^T$,$x_2{}^T = (0, 0)^T$,$x_3{}^T = (1, 1)^T$ and the first pricipal $z = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})^T$. so $x_1{}' = x_1{}^T z = -\sqrt{2}$, $x_2{}' = 0$, $x_3{}' = \sqrt{2}$.

(c)

$$Var = \frac{1}{3}\sum_{i=1}^{3}(x_i' - 0) = \frac{3}{4}$$

III Model Evaluation and Selection

Questions 6.
Relating the concept of confusion matrix that we see in course, we can denote

$$\begin{array}{cccc} (labelededible, edible) & is & TrueNegative & \textbf{TN} \\ (labelededible, poisonous) & is & FalsePositive & \textbf{FP} \\ (labeledpoisonous, poisonous) & is & TruePositive & \textbf{TP} \\ (labeledpoisonous, edible) & is & FalseNegative & \textbf{FN} \end{array}$$

(a) Accuracy(Algo1)=$\dfrac{TP+TN}{TP+FN+FP+TN} = \dfrac{97+100}{200} = 0.985$

Accuracy(Algo2)=$\dfrac{TP+TN}{TP+FN+FP+TN} = \dfrac{96+100}{200} = 0.98$

(b) Algorithme 1, because Algorithme 1 classifies all the lebaled poisonous mushrooms as poisonous, contrarily Algorithme 2 has type II error which is the missing error. So in our case, we want to assure that every poisonous mushroom will be classified as poisonous, so we want to minimize the false negative rate(**FN**). That's why we choose Algo 2.

Questions 7.
False Positive Rate. We want to minimise unditected(predicted condition is negative, which is false in reality) fraud (True condition is positive),that is to say FP, given a total number of transaction, we want to minimize FPR.

IV Linear Regression, Logistic Regression and Feature Selection

Questions 8.

(a) Using the notation in class, we have $x^{(i}$ standing for i^{th}sample, and $Y_{n\times 1}$ the label. After tedious computation in paper(it's not so readable, so...), we have

$$\nabla_\theta J_{RR}(\theta) = X^T(X\theta - Y) + \lambda I\theta$$

So the closed form solution is when $\nabla_\theta J_{RR}(\theta) = 0$

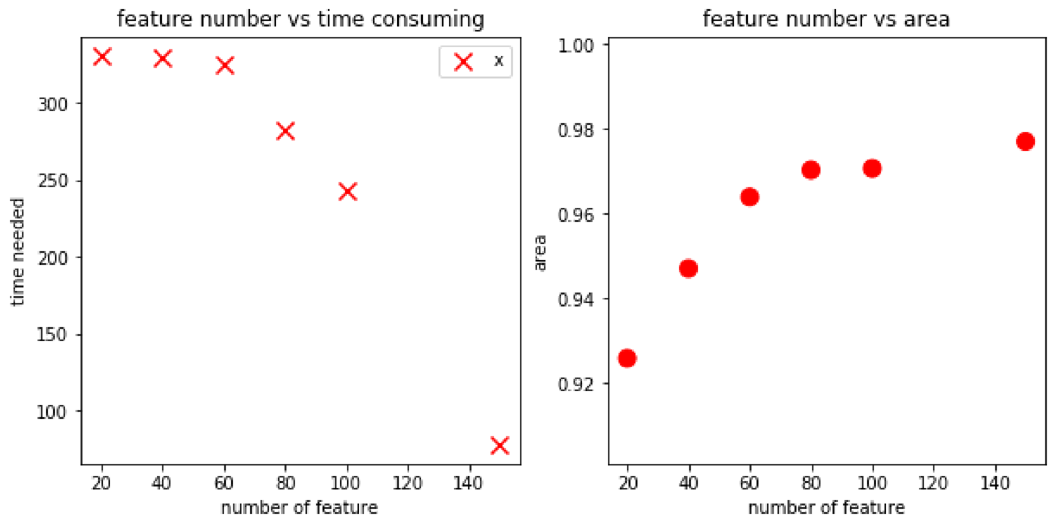
$$\hat{\theta} = (X^TX + \lambda I)^{-1}X^TY$$

(b) L2 regularization term prefers parameters close to zero, so it is more robust to overfitting.

Questions 9.

1 the area under curve is 0.9775851267688679, the required running time is 5.332571918898696. The implementation in Python is shown below.

2 We can see from the curve: feature selection improve the accuracy,but the accuracy is tending to saturation, we observe that more feature result to less running time?



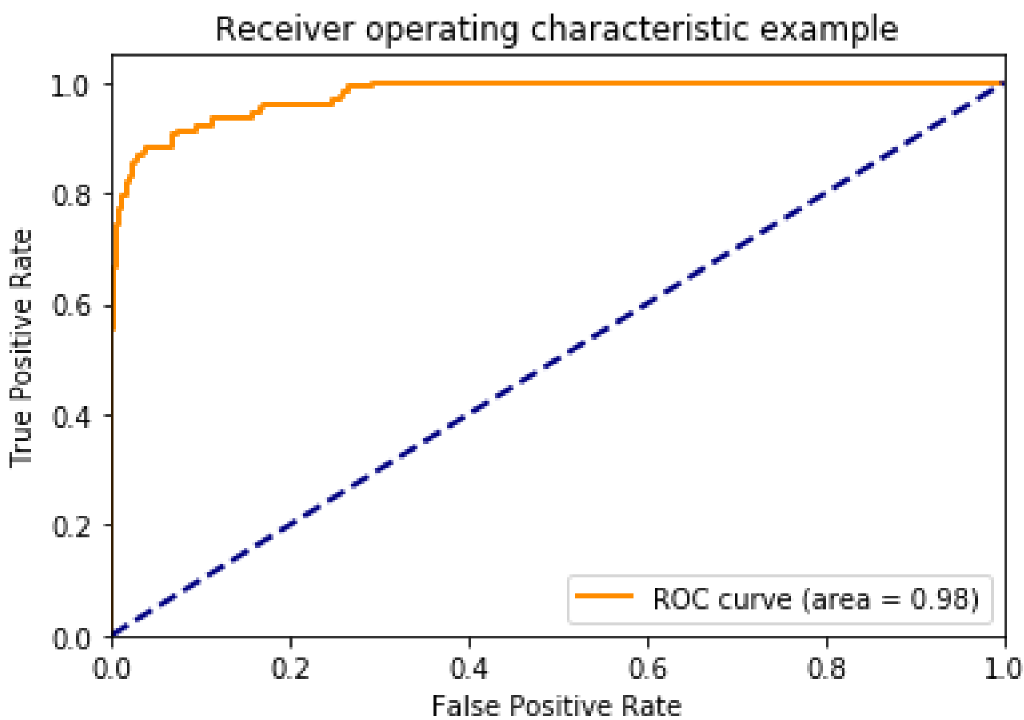
MuskClassify

October 18, 2018

In [2]:
```
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt

#load data
from sklearn.linear_model import LogisticRegression
import numpy as np
data = np.loadtxt('data.csv',delimiter = ',')
Y = data[:,0:1]
#Y = column_or_1d(Y,warn=True)
X = data[:,1:data.shape[1]]

import timeit
start = timeit.default_timer()
clf =LogisticRegression(random_state=0, solver="liblinear").fit(X, Y)
elapsed = (timeit.default_timer() - start)

#load test
test_data = np.loadtxt('test.csv',delimiter = ',')
test = test_data[:,1:test_data.shape[1]]
test_Y = test_data[:,0:1]
test_Y = test_Y.transpose()[0]
clf.predict(test)
fpr = dict()
tpr = dict()
roc_auc = dict()
y_score = clf.fit(X,Y).decision_function(test)
fpr, tpr,threshold = roc_curve(test_Y, y_score)
roc_auc = auc(fpr, tpr)


# Plot ROC
plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
```

```python
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic example')
plt.legend(loc="lower right")
plt.show()
#calculate score
from sklearn.metrics import roc_auc_score
score = roc_auc_score(test_Y, y_score)
print("examing the test set, score is")
print(score)

#regression and record training time
print("time needed for training is")
print(elapsed)
```

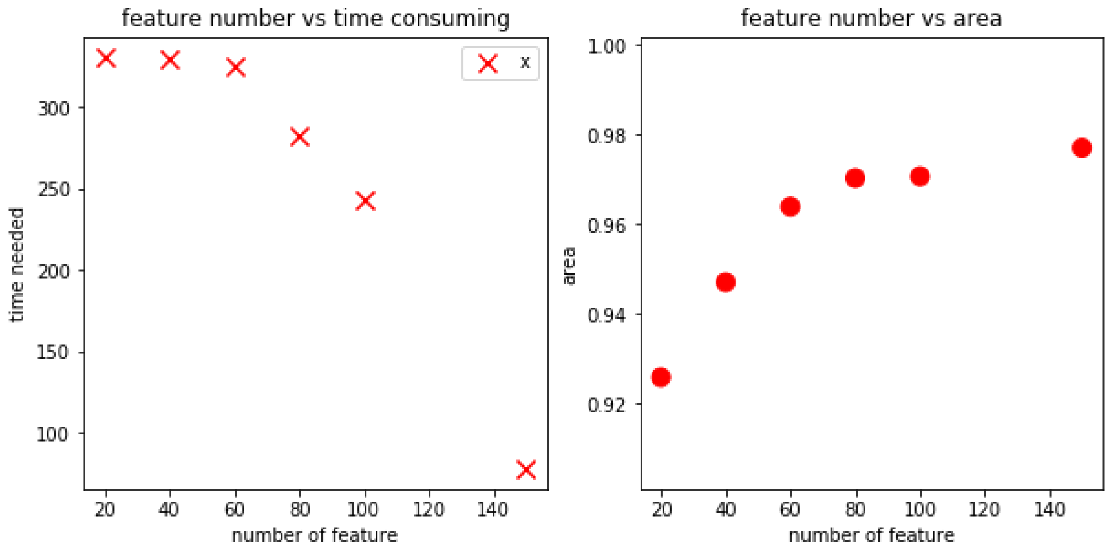
```
F:\software\1PYTHON\anaconda\lib\site-packages\sklearn\utils\validation.py:578: DataConversionWa
  y = column_or_1d(y, warn=True)
```




```
examing the test set, score is
0.9775851267688679
time needed for training is
5.332571918898696
```

In [22]:
```
from sklearn.feature_selection import RFE
estimator = LogisticRegression(random_state=0, solver="liblinear")
FeatureRange=[20,40,60,80,100,150]
#for test FeatureRange=[20]
Areas=[]
UsingTime=[]
for i in FeatureRange:
    selector = RFE(estimator, i, step=1)
    import timeit
    start = timeit.default_timer()
    selector = selector.fit(X, Y)
    UsingTime.append(timeit.default_timer() - start)
    selector.predict(test)
    y_score = selector.fit(X,Y).decision_function(test)
    from sklearn.metrics import roc_auc_score
    Areas.append(roc_auc_score(test_Y, y_score))

import matplotlib.pyplot as plt
```

In [25]:
```
fig = plt.figure(figsize=(10, 10))
ax1 = fig.add_subplot(222)
ax1.set_title('feature number vs area')
plt.xlabel('number of feature')
plt.ylabel('area')
lValue = Areas*10
ax1.scatter(FeatureRange,Areas,c='r',s= 100,linewidths=lValue,marker='o')


ax2 = fig.add_subplot(221)
ax2.set_title('feature number vs time consuming')
plt.xlabel('number of feature')
plt.ylabel('time needed')
lValue = Areas*10
ax2.scatter(FeatureRange,UsingTime,c='r',s= 100,linewidths=lValue,marker='x')

plt.legend('x1')

plt.show()
```


feature number vs time consuming
time needed
300
250
200
150
100
20
40
60
80
100
120
140
number of feature
x
feature number vs area
area
1.00
0.98
0.96
0.94
0.92
20
40
60
80
100
120
140
number of feature