Homework 2 - Probability Theory and Logistic Regression

Ran Ju

Netid: *rj133*

1

[1.5 points] For each part (a) through (c), indicate whether we would generally expect the performance of a flexible (more complex) statistical learning method to be better or worse than an inflexible method. Justify your answer.

a. The sample size n is extremely large, and the number of features p is small.

b. The number of features p is extremely large, and the number of observations n is small.

c. The relationship between the features and target variable is highly non-linear.

ANSWER

a. A flexible statistical learning method would preform **better** because the large n will reduces the risk of overfitting, which is the biggest problem with highly flexible methods.

b. A flexible statistical learning method would preform **worse** because there is a great risk of overfitting.

c. A flexible statistical learning method would preform **better** because it is more easily to fit the non-linear function than the inflexible model.

2

[1.5 points] For each of the following, (i) explain if each scenario is a classification or regression problem and (ii) provide the sample size n and number of features p indicated for each scenario.

(a) We collect a set of data on the top 500 firms in the US. For each firm we record profit, number of employees, industry and the CEO salary. We are interested in understanding which factors affect CEO salary.

(b) We are considering launching a new product and wish to know whether it will be a success or a failure. We collect data on 20 similar products that were previously launched. For each product we have recorded whether it was a success or failure, price charged for the product, marketing budget, competition price, and ten other variables.

(c) We are interesting in predicting the % change in the US dollar in relation to the weekly changes in the world stock markets. Hence we collect weekly data for all of 2012. For each week we record the % change in the dollar, the % change in the US market, the % change in the British market, and the % change in the German market.

ANSWER

(a) (i) It is a **regresion** problem because we want to know how the variables affect a continuous variable.

(ii) $n = 500$ because there are 500 fiirms to provide the data. $p = 3$ because profit, number of employees and salary are the factors that have effect on the CEO salary.

(b) (i) It is a **classification** problem because we only care about it is success or failure which is a binary selection.

(ii) $n = 20$ because there are 20 similar products. $p = 13$ because there are price charged for the product, marketing budget, competition price, and ten other variables that have effect on the result.

c) (i) It is a **regresion** problem because we want to know % change in the US dollar which is a continuous number in relation to the weekly changes in the world stock markets.

(ii) $n = 52$ because there are 52 weeks in 2012. $p = 3$ because the world stock market is related with three parts, the % change in the US market, the % change in the British market, and the % change in the German market.

3

[2 point] Suppose that we have three coloured boxes r (red), b (blue), and g (green). Box r contains 3 apples, 4 oranges, and 3 limes, box b contains 1 apple, 1 orange, and 0 limes, and box g contains 3 apples, 3 oranges, and 4 limes.

(a) If a box is chosen at random with probabilities p(r) = 0.2, p(b) = 0.2, p(g) = 0.6, and a piece of fruit is removed from the box (with equal probability of selecting any of the items in the box), then what is the probability of selecting an apple?

(b) If we observe that the selected fruit is in fact an orange, what is the probability that it came from the green box?

Explain your reasoning.

ANSWER:

(a) The probability is 0.34. Because

$$P(apple) = P(apple|red) + P(apple|blue) + P(apple|green)$$
$$= 0.2 \times (\frac{3}{3+4+3}) + 0.2 \times (\frac{1}{1+1+0}) + 0.6 \times (\frac{3}{3+3+4}) = 0.5$$

(b) The probability is 0.5. Because

$$P(green|orage) = \frac{P(green)P(orage|green)}{P(orage)}$$
$$= \frac{0.6 \times (\frac{3}{3+4+3})}{0.2 \times (\frac{4}{3+3+4}) + 0.2 \times (\frac{1}{1+1+0}) + 0.6 \times (\frac{3}{3+3+4})} = 0.5$$

4

[5 points] Participate in the Kaggle Titanic Machine Learning Challenge.

Kaggle (https://en.wikipedia.org/wiki/Kaggle) is a platform that hosts machine learning competitions, notebooks, and datasets. This exercise will get you started with participating in Kaggle competitions, a skill you will need in week four of this course.

(a) [1 point] Create an account on Kaggle and sent your username via email to Chenglong Tang (ct265@duke.edu (mailto:ct265@duke.edu)). Your username has to be either your real name, your netid, or a pseudonym you sent him via email *before* you open the account. If you already have a Kaggle account, you probably will have trouble to open a second one → sent me an email.

Then participate in the Titanic: Machine Learning from Disaster (https://www.kaggle.com/c/titanic/overview) competition. Ressources that help you get started are this video (https://www.youtube.com/watch?v=8yZMXCaFshs) and Alexis Cook tutorial (https://www.kaggle.com/alexisbcook/titanic-tutorial). However, you should not use a Kaggle notebook and instead write your Python code on your own computer.

(b) [2 points] Write a notebook that uses logistic regression, at least three different features, and 3 fold cross-validation. Then make a prediction for the test data set.

(c) [1 point] Add L_1 regularization to your notebook and use GridSearchCV to determine the best regularization strength.

(d) [1 point] make a submission to the Titanic contest and document it with a screenshot of your position at the leaderboard. Your Ranking does not matter.

Besides the PDF containing your code (for questions 4 b and c) and the answers (to questions 1 to 3), you should also submit an additional notebook which allows to run your code. Run ***restart and clear all*** before you submit the notebook. For (d) you need to submit an image, preferably in JPEG or PNG format.

ANSWER

In [1]:

```
# import necessary library and package
import pandas as pd
import numpy as np
import seaborn as sns
%matplotlib inline

import matplotlib.pyplot as plt

train=pd.read_csv('./train.csv')#load training data
train.head()
```

Out[1]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [2]:

```
train.describe()# the situation of the features
```

Out[2]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [3]:

```
import warnings
warnings.filterwarnings('ignore')# do not show the warnings
def dataprocess(data):
    #select the primary features manually
    data=data.drop('PassengerId',axis=1)
    data=data.drop('Ticket',axis=1)
    data=data.drop('Cabin',axis=1)
    data=data.drop('Name',axis=1)
    # imputation for age
    mean=data['Age'].mean()
    where_are_nan=data['Age'].isnull()
    data['Age'][where_are_nan]=mean
    for i in range(len(data['Sex'])):
        if data['Sex'][i]=='male':
            data['Sex'][i]=0# set male=0 for convenience
        else:
            data['Sex'][i]=1# set female=1 for convenience

    for i in range(len(data['Embarked'])):
        if data['Embarked'][i]=='C':
            data['Embarked'][i]=0# set Embarked c=0 for convenience
        if data['Embarked'][i]=='Q':
            data['Embarked'][i]=1# set Embarked q=1 for convenience
        else:
            data['Embarked'][i]=2# set Embarked s=2 for convenience
    return data
datatrain=dataprocess(train)
datatrain.describe()
```

Out[3]:

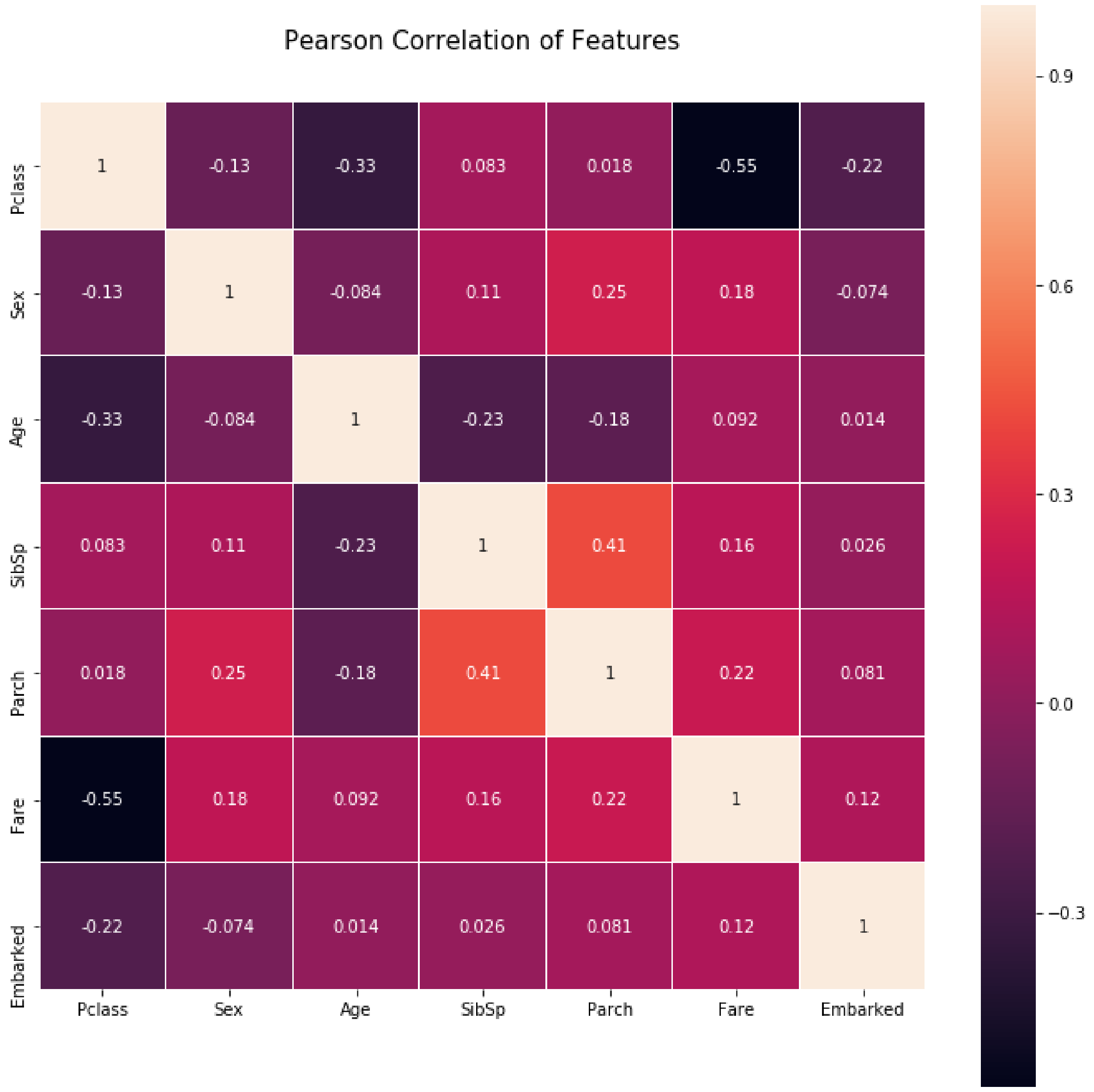
	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	13.002015	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	22.000000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	29.699118	0.000000	0.000000	14.454200
75%	1.000000	3.000000	35.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [4]:

```
trainx=datatrain.drop('Survived',axis=1)
trainy=datatrain['Survived']
```

In [5]:

```
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
colormap = plt.cm.RdBu# use heatmap we can find the relation between each features
plt.figure(figsize=(12,12))
plt.title('Pearson Correlation of Features', y=1.05, size=15)
ax=sns.heatmap(trainx.astype(float).corr(),linewidths=0.05,vmax=1.0, square=True,
            linecolor='white', annot=True)
ax.set_ylim([7, 0])
plt.show()
```



	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
Pclass	1	-0.13	-0.33	0.083	0.018	-0.55	-0.22
Sex	-0.13	1	-0.084	0.11	0.25	0.18	-0.074
Age	-0.33	-0.084	1	-0.23	-0.18	0.092	0.014
SibSp	0.083	0.11	-0.23	1	0.41	0.16	0.026
Parch	0.018	0.25	-0.18	0.41	1	0.22	0.081
Fare	-0.55	0.18	0.092	0.16	0.22	1	0.12
Embarked	-0.22	-0.074	0.014	0.026	0.081	0.12	1

In [6]:

```
from sklearn.linear_model import LogisticRegression
totalnumber=len(trainx)
split=totalnumber//3#3-fold cross validation
trsl=[]
tesl=[]
for i in range(3):
    testxc=trainx[i*split:(i+1)*split]#split the data
    testyc=trainy[i*split:(i+1)*split]#pay attention to set the name else the numbe
                                        #will decrease after each round of validati
    trainxc=np.concatenate((trainx[0:i*split],trainx[(i+1)*split:]),axis=0)
    trainyc=np.concatenate((trainy[0:i*split],trainy[(i+1)*split:]),axis=0)
    log_reg=LogisticRegression(penalty='l1',C=0.101)# use the c according to the re
    log_reg.fit(trainxc,trainyc)
    trainscore=log_reg.score(trainxc, trainyc)
    trsl.append(trainscore)
    testscore=log_reg.score(testxc, testyc)
    tesl.append(testscore)
print('train scores: ')
print(trsl)
print('test scores: ')
print(tesl)
print('mean of train scores: {:.4f}'.format(np.mean(trsl)))
print('mean of test scores: {:.4f}'.format( np.mean(tesl)))
```

```
train scores:
[0.7861952861952862, 0.797979797979798, 0.7861952861952862]
test scores:
[0.7912457912457912, 0.7845117845117845, 0.7912457912457912]
mean of train scores: 0.7901
mean of test scores: 0.7890
```

In [7]:

```
from sklearn.linear_model import LogisticRegression
totalnumber=len(trainx)
split=totalnumber//3#3-fold cross validation
trsl=[]
tesl=[]
for i in range(3):
    testxc=trainx[i*split:(i+1)*split]#split the data
    testyc=trainy[i*split:(i+1)*split]#pay attention to set the name else the numbe
                                       #will decrease after each round of validati
    trainxc=np.concatenate((trainx[0:i*split],trainx[(i+1)*split:]),axis=0)
    trainyc=np.concatenate((trainy[0:i*split],trainy[(i+1)*split:]),axis=0)
    log_regw=LogisticRegression(penalty='l2',C=1e42)# use the c very large so there
    log_regw.fit(trainxc,trainyc)
    trainscore=log_regw.score(trainxc, trainyc)
    trsl.append(trainscore)
    testscore=log_regw.score(testxc, testyc)
    tesl.append(testscore)
print('train scores: ')
print(trsl)
print('test scores: ')
print(tesl)
print('mean of train scores: {:.4f}'.format(np.mean(trsl)))
print('mean of test scores: {:.4f}'.format( np.mean(tesl)))
```

```
train scores:
[0.8097643097643098, 0.8047138047138047, 0.803030303030303]
test scores:
[0.7811447811447811, 0.7811447811447811, 0.8047138047138047]
mean of train scores: 0.8058
mean of test scores: 0.7890
```

It seems that if there are no regularization here the accuracy will be better. However, when I tried to upload the test results to kaggle, I find the model with regularization will get higher scores.

In [8]:

```
test=pd.read_csv('./test.csv')#load testing data
testdata=dataprocess(test)
```

In [9]:

```
sum=testdata.isna().sum()#find is there any null value in test data
print(sum)
```

```
Pclass      0
Sex         0
Age         0
SibSp       0
Parch       0
Fare        1
Embarked    0
dtype: int64
```

In [10]:

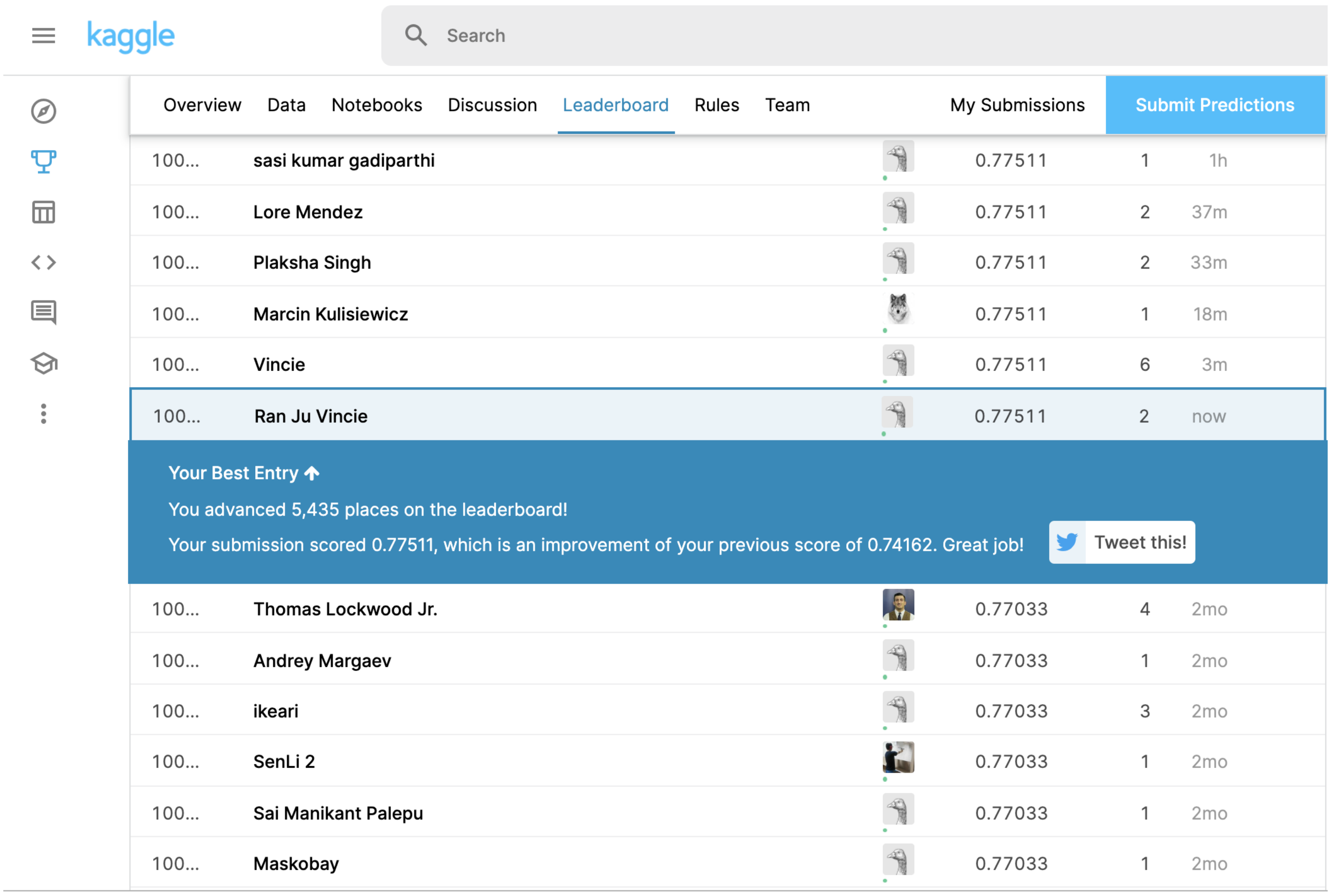
```
mean=testdata['Fare'].mean()# beacuse there is a null value in fare so I use the mea
where_are_nan=testdata['Fare'].isnull()
testdata['Fare'][where_are_nan]=mean
result=log_reg.predict(testdata)
pid=pd.DataFrame(columns=['PassengerId'],data=test['PassengerId'])
testresult=pd.DataFrame(columns=['Survived'],data=result)
testresult=pid.join(testresult)# set the result csv as the form needed include passe
                                #and whether they are survived
testresult.to_csv('./testdataresult.csv',index=False)
```

In [11]:

```
from sklearn.model_selection import GridSearchCV
c=np.linspace(0.001,10,1000)
parameters={'C':c}#select the best regularization strength
lg=LogisticRegression(penalty='l1')
clf=GridSearchCV(lg,parameters,cv=3)
clf.fit(trainx,trainy)
print("Best parameters set found on development set:")#print the result
print(clf.best_params_)
print('The score of best parameters:  ')
print('{:.4f}\n'.format(clf.best_score_))
```

```
Best parameters set found on development set:
{'C': 0.1010900900900901}
The score of best parameters:
0.7935
```

My rank in the leaderboard



100...	sasi kumar gadiparthi	0.77511	1	1h
100...	Lore Mendez	0.77511	2	37m
100...	Plaksha Singh	0.77511	2	33m
100...	Marcin Kulisiewicz	0.77511	1	18m
100...	Vincie	0.77511	6	3m
100...	Ran Ju Vincie	0.77511	2	now

Your Best Entry ↑

You advanced 5,435 places on the leaderboard!

Your submission scored 0.77511, which is an improvement of your previous score of 0.74162. Great job!

Tweet this!

100...	Thomas Lockwood Jr.	0.77033	4	2mo
100...	Andrey Margaev	0.77033	1	2mo
100...	ikeari	0.77033	3	2mo
100...	SenLi 2	0.77033	1	2mo
100...	Sai Manikant Palepu	0.77033	1	2mo
100...	Maskobay	0.77033	1	2mo