Homework 10

Argiro Chatzis (Team: Good Toads)

Our team made multiple submissions for ensemble learning because as you will see in the code below any number of submissions can be considered for ensemble learning so we tried to ensemble multiple different combinations of submissions until we got a better score. In the code below we used 9 different submissions, some of them are from the previous HW (HW 9) which were results from using different machine learning algorithms and some are from our efforts in HW 5.

We first started out by importing the necessary packages and reading all the submission files we wanted to use. In this case we used 9 different submissions.

```
In [1]:  import pandas as pd
         import numpy as np
         from numpy import *
```

```
In [2]:  p_1 = pd.read_csv('factorization_machine_1.csv')
         p_2 = pd.read_csv('ensemble_3.csv')
         p_3 = pd.read_csv('gradient_boosted_tree-2.csv')
         p_4 = pd.read_csv('ml_alg_4.txt')
         p_5 = pd.read_csv('decision_tree.csv')
         p_6 = pd.read_csv('sum.csv')
         p_7 = pd.read_csv('ensemble_2.csv')
         p_8 = pd.read_csv('SVM.csv')
         p_9 = pd.read_csv('logistic_regression.csv')
```

We then replaced all the 0's in the data to -1's to match the predictions to the truth's for each file.

```
In [3]:  p_1.Predictor.replace(0,-1,inplace= True)
         p_2.Predictor.replace(0,-1,inplace= True)
         p_3.Predictor.replace(0,-1,inplace= True)
         p_4.Predictor.replace(0,-1,inplace= True)
         p_5.Predictor.replace(0,-1,inplace= True)
         p_6.Predictor.replace(0,-1,inplace= True)
         p_7.Predictor.replace(0,-1,inplace= True)
```

We're using the least squares solution in order to find the weights a_1, a_2, a_3,… and we use the equation $a_{LS}=(S^TS)^{-1}S^Tx$. We first want to find the matrix S and to do this we read all the data as a list and converted all the predictions into a single matrix.

```
In [4]: s1 = p_1.Predictor.tolist( )
        s2 = p_2.Predictor.tolist( )
        s3 = p_3.Predictor.tolist( )
        s4 = p_4.Predictor.tolist( )
        s5 = p_5.Predictor.tolist( )
        s6 = p_6.Predictor.tolist( )
        s7 = p_7.Predictor.tolist( )
        s8 = p_8.Predictor.tolist( )
        s9 = p_9.Predictor.tolist( )

        S=np.transpose(mat([s1,s2,s3,s4,s5,s6,s7,s8,s9]))
        S
```

```
Out[4]:

matrix([[-1., -1., -1., ..., -1., -1., -1.],
        [-1.,  1., -1., ..., -1., -1., -1.],
        [-1., -1., -1., ..., -1., -1., -1.],
        ...,
        [ 1.,  1.,  1., ...,  1.,  1.,  1.],
        [-1., -1., -1., ...,  1., -1.,  1.],
        [ 1.,  1.,  1., ...,  1.,  1.,  1.]])
```

We then listed the scores for each model and created a transpose matrix out of those scores. After that we used the equation $S^Tx= N(2P_i-1)$ and plugged in the scores matrix for P_i and N would be the number of scores in our submission file which are 120000.

```
In [5]: score1 = 0.85745
        score2 = 0.87654
        score3 = 0.85753
        score4 = 0.87152
        score5 = 0.85756
        score6 = 0.84544
        score7 = 0.83141
        score8 = 0.85578
        score9 = 0.85771

        score_mat=np.transpose(mat([score1,score2,score3,score4,score5,score6,score7,score8,score9]))
        N=120000
        STx=N*(2*score_mat-1)

        STx
```

Out[5]:

```
matrix([[85788. ],
        [90369.6],
        [85807.2],
        [89164.8],
        [85814.4],
        [82905.6],
        [79538.4],
        [85387.2],
        [85850.4]])
```

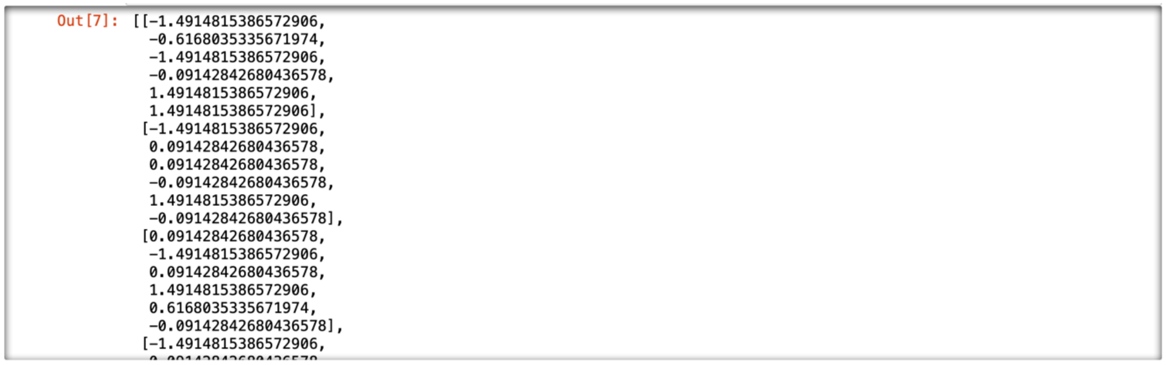
Finally, we combined the S matrix we found and the S^Tx matrix we found and plug them into the least squares equation, $a_{LS}=(S^TS)^{-1}S^Tx$. We created a new matrix from this, new_S and then we got the data back in the format we need for submitting so we had to make the ratings 0's and 1's again.

In [7]:

```
STS=np.transpose(S)*S
STS_inverse=np.linalg.pinv(STS)
STS_inverse

a=STS_inverse*STx
S_new=S*a
S_new=S_new.T
new_S=np.array(S_new).reshape((20000,6)).tolist( )
new_S
```

Out[7]:



```
Out[7]: [[-1.4914815386572906,
          -0.6168035335671974,
          -1.4914815386572906,
          -0.09142842680436578,
          1.4914815386572906,
          1.4914815386572906],
         [-1.4914815386572906,
          0.09142842680436578,
          0.09142842680436578,
          -0.09142842680436578,
          1.4914815386572906,
          -0.09142842680436578],
         [0.09142842680436578,
          -1.4914815386572906,
          0.09142842680436578,
          1.4914815386572906,
          0.6168035335671974,
          -0.09142842680436578],
         [-1.4914815386572906,
```

In [8]:

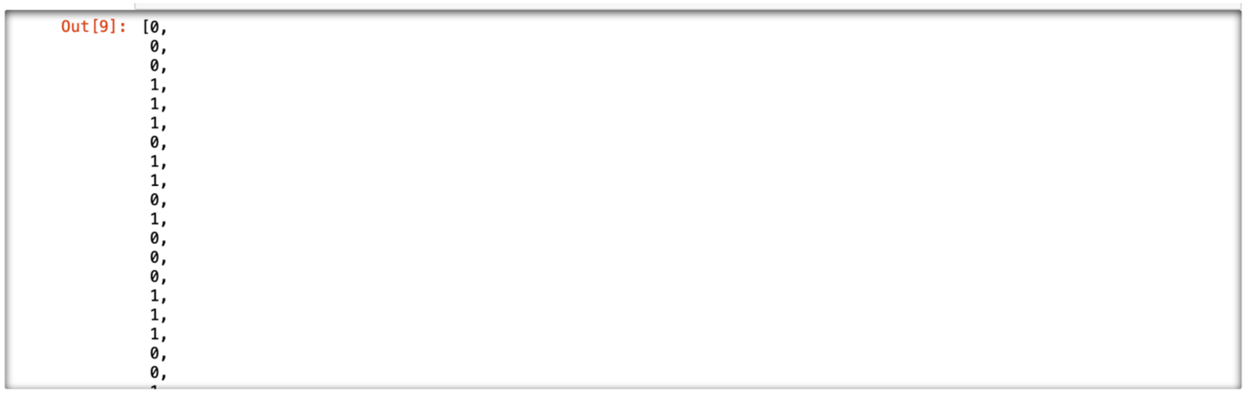
```
def prediction_table(grades_table):
  prediction_table = [0 for i in range(len(grades_table)*6)]
  for i in range(len(grades_table)):
        x = np.array(grades_table[i])
        x = x.argsort( )

        for k in range(6):
            if k<3:
                prediction_table[i * 6 + x[k]] = 0
            else:
                prediction_table[i * 6 + x[k]] = 1

    return prediction_table
```

```
In [9]:  new_ratings_table = prediction_table(new_S)
         new_ratings_table
```

Out[9]:



```
Out[9]: [0,
         0,
         0,
         1,
         1,
         1,
         0,
         1,
         1,
         0,
         1,
         0,
         0,
         0,
         1,
         1,
         1,
         0,
         0,
```

```
In [10]: p_2.Predictor = new_ratings_table
In [11]: p_2
```

Out[11]:

	TrackID	Predictor
0	199810_208019	0
1	199810_74139	0
2	199810_9903	0

	TrackID	Predictor
3	199810_242681	1
4	199810_18515	1
...	...	...
119995	249010_72192	0
119996	249010_86104	0
119997	249010_186634	1
119998	249010_293818	0
119999	249010_262811	1

120000 rows × 2 columns

```
In [12]: p_2.to_csv('ensemble_5.csv',index = False)
```

We saved the new file as ensemble_5.csv and although our previous ensemble submissions (i.e. ensemble_1.csv, ensemble_2.csv, ensemble_3.csv, ensemble_4.csv) were a bit lower, when we submitted our file we were finally able to improve our performance to get a score of 0.87654 which is our highest submission.