Predicting Hit Songs

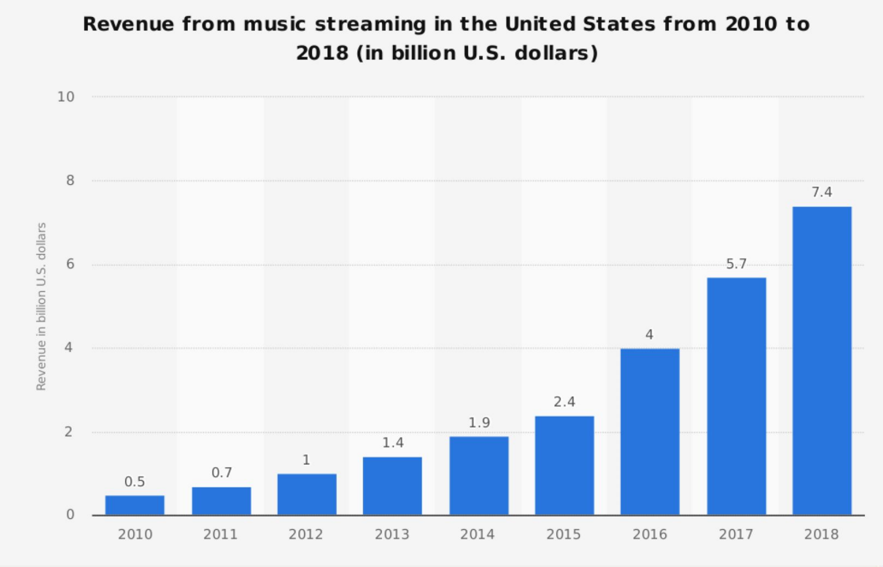
Prof. Alvarez CSCI3345

Henry Chen
Cecilia Wu
Yiyuan Xu
Timothy Yang
Zheng Zhou

Background

Growing presence of music streaming services

- Apple Music achieved 60 million users worldwide
- US music streaming revenue reached 7.4 billion in 2018



Objectives

<u>Interests</u>

Spotify reports that their artists only get paid from $0.006 to $0.0084 per play.

- To help artists produce hit songs and get bigger paychecks

<u>Academics</u>

- To build a classifier that can predict whether or not a song will be a “hit song” based on the songs musical features and artist features

** “hit song” defined as a song that has made it to Billboard's top 100 ranking

Dataset - Source

Million Song Dataset

- The original dataset has 1 million songs (300GB)
- Due to resource constraints, decided to use a random sample of 10,000 songs

data.world

- Weekly rankings for Billboard’s Hot 100 in csv format

Final Dataset

- Matched song names from the MSD data to add labels
- +1 if the song ever made to the Billboard
- -1 if it has not

Challenge

<u>Imbalanced Classification Data</u>

- The 10,000 sample has a split of 88%/12%

  1224 cases of +1 labels

  8776 cases -1 labels

- The resulting classifier has a tendency to predict -1 better than +1

```
Number of songs predicted to be +1: 18
Number of songs predicted to be -1: 1982
```

- Solution:

  => Rebalance the classes (67%/33%)

  1224 +1, 612: -1

Techniques

Data split

- 80% training & 20% testing

To avoid overfit

- 10-fold cross validation

Music features included

- Artist hotness, artist familiarity, danceability, duration, end of fade in, energy, key, key confidence, loudness, mode, mode confidence, start of fade out, tempo, time signature, time signature confidence
- **SelectKBest:** select k highest scoring features

Classifiers used

1. KNN 2. SVM 3. Naive Bayes 4. Logistic Regression

Results

In-sample error, Out-of-sample error, Precision, Recall, F1

Measurements

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$= \frac{True\ Positive}{Total\ Actual\ Positive}$$

$$\text{F1} = 2 \text{ x } \frac{Precision * Recall}{Precision + Recall}$$

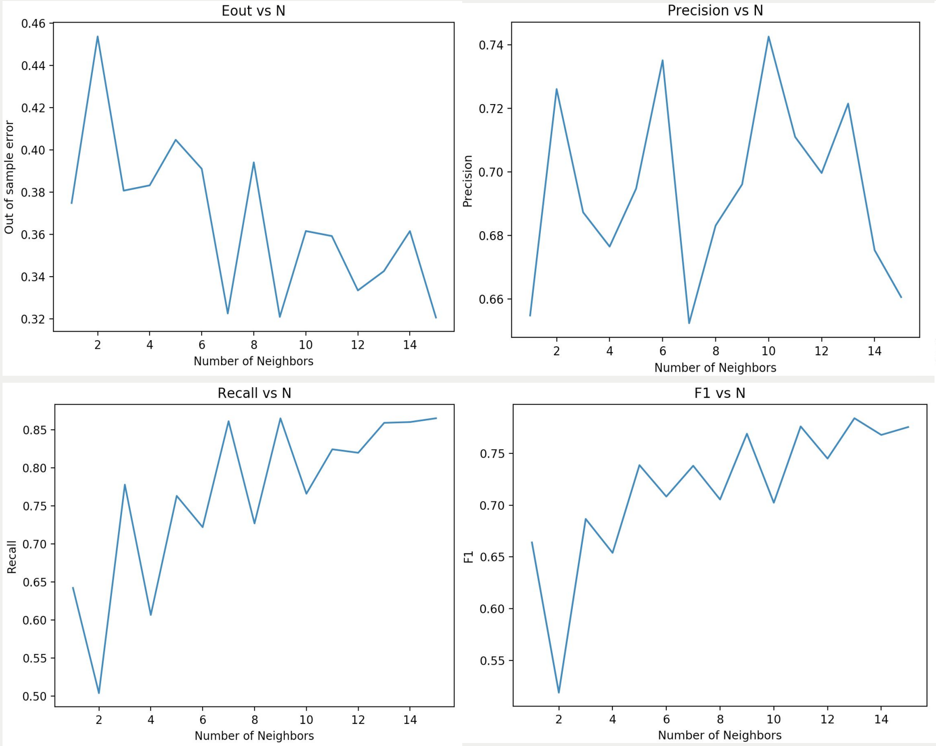
$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$= \frac{True\ Positive}{Total\ Predicted\ Positive}$$

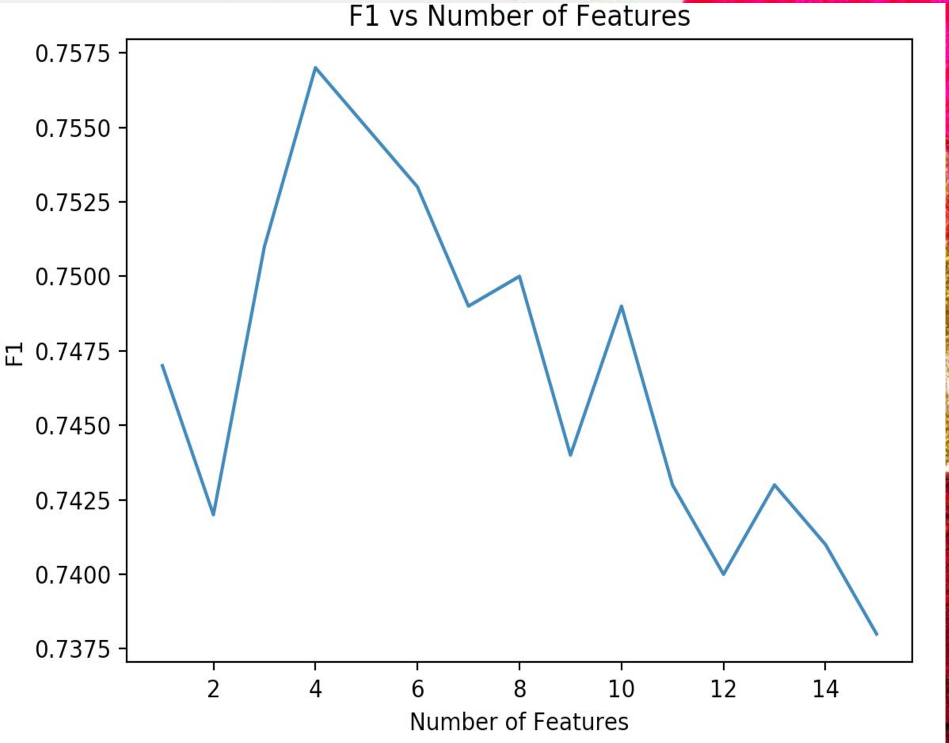
		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

KNN

- Choosing N



Choosing features



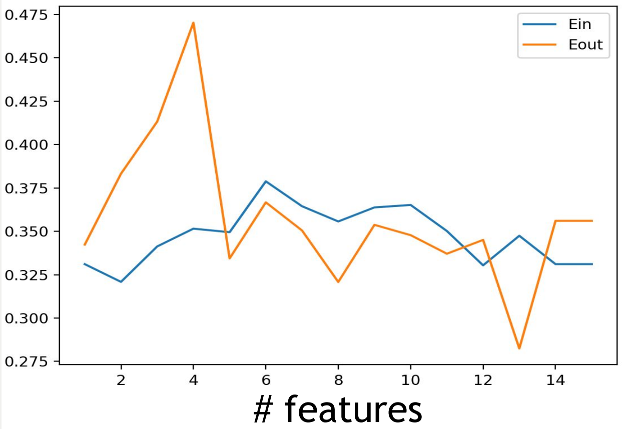
KNN (cont.)

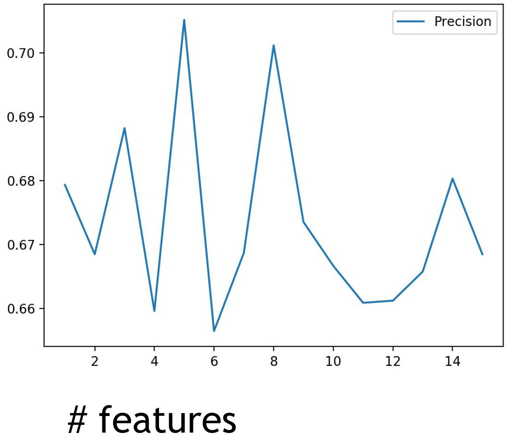
- Rebalance the classes. 1224: +1, 612: -1 (67%, 33%)

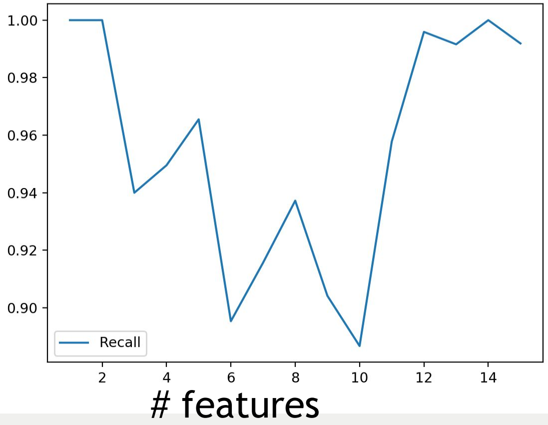
```
Number of songs predicted to be +1: 308
Number of songs predicted to be -1: 60
ein: 0.36512856683865197
eout: 0.36858608345187305
precision: 0.6802569393003819
recall: 0.8478102821104926
f1: 0.7543238500460879
```

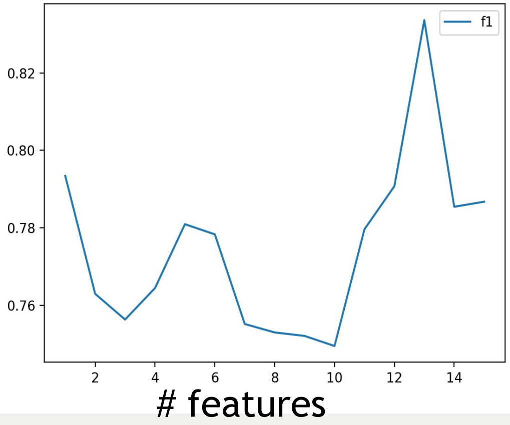
- While the classifier stills predicts the majority to be the dominant class, the precision and recall are both significantly higher.
    - 68.0% of what we classified as as hit songs were actually hit songs
    - 84.8% of actual hit songs were correctly predicted as such.

SVM









Optimal Result:
13 features
Ein 0.3489
Eout 0.2868
F1 0.8322
Recall 0.99
Precision 0.6487

Hyperparameters Selection (Kernel degree p & penalty constant c)

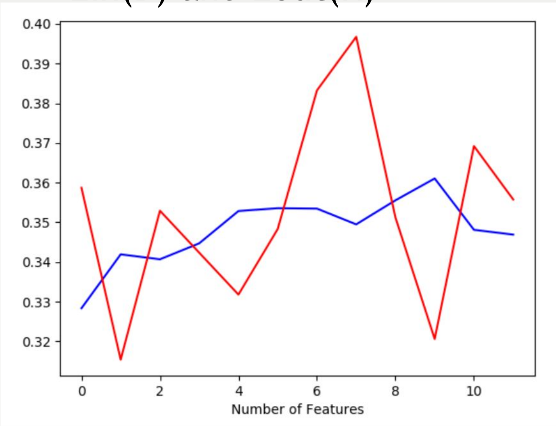
Degree :[1,2,4], C : [0.5,1,2]

best performance for: {'C': 0.5, 'degree': 1}

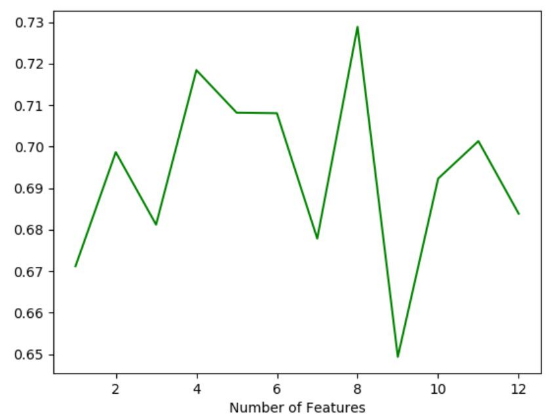
cross-validated accuracy : 0.6267

Naive Bayes Plots

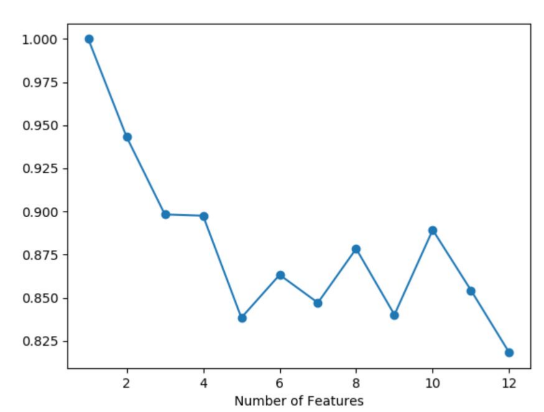
Ein(B) and Eout(R)



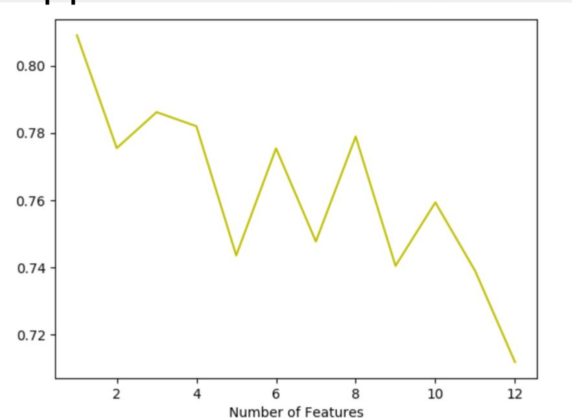
Precision



Recall



F1

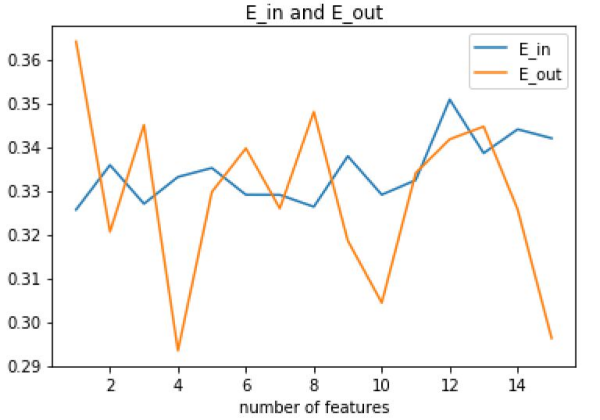


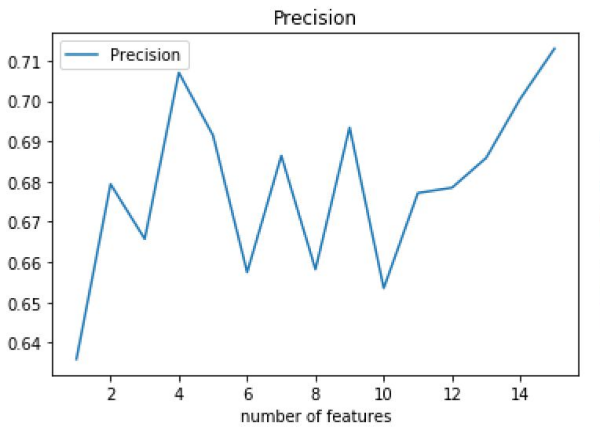
Naive Bayes

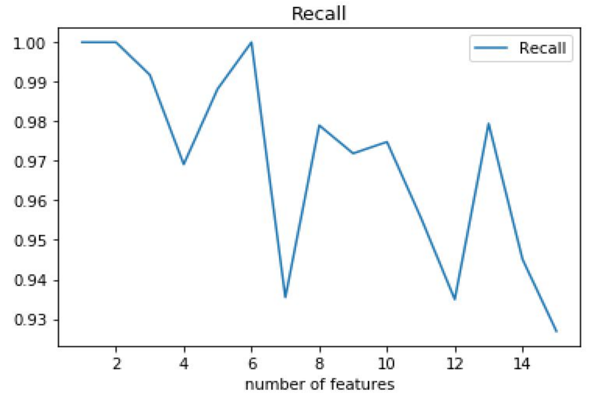
```
Number of song predicted to be +1: 312
Number of song predicted to be -1: 56
ein: 0.374618307152536
eout: 0.353153153153153l
precision: 0.7083333333333334
recall: 0.8565891472868217
F1: 0.7754385964912281
```

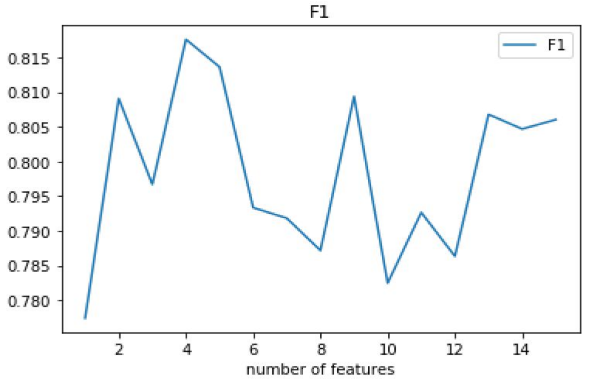
- Of the 1836 data samples(1:1224 and -1:612), 20% or 368 are used for tests
- Of the number of test songs Naive bayes has around a 85% accuracy predicting the songs that were not hit songs and around 70% accuracy predicting songs that are hit songs.
- The increase in number of features K has a negative effect on total recall and f1

Logistic Regression







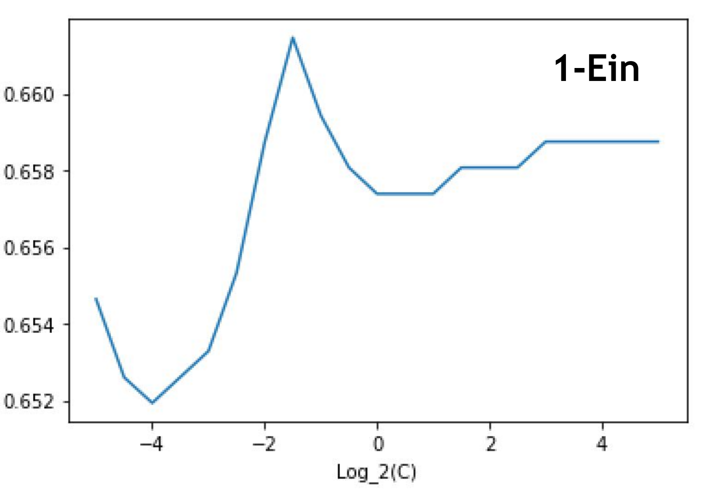


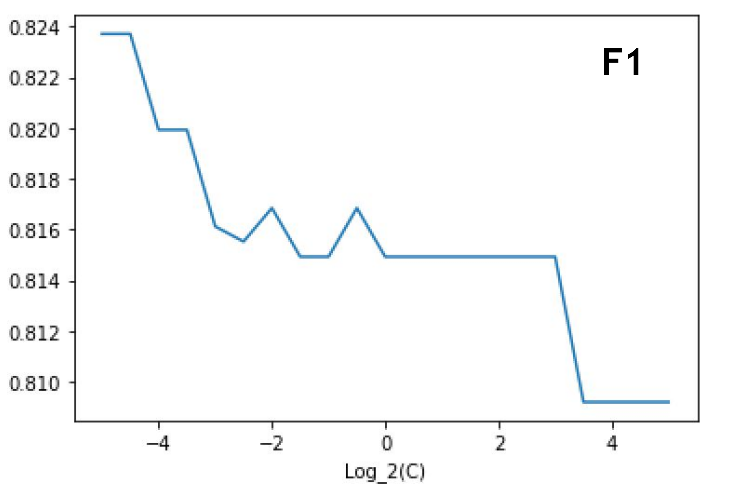
Optimal 4 features with max F1:

```
E_in: 0.3310036343304444
E_out: 0.29343629343629357
Precision: 0.7070422535211267
Recall: 0.9691119691119688
F1: 0.8175895765472312
```

70% of songs we predict as hit songs are true hit songs. 97% of true hit songs are predicted as hit songs by the model.

Attempts to Regularize with 4 Features





Optimal result with maximum F1:

```
Log_2(C): -5.0
E_out: 0.30162399241346605
Precision: 0.7002724795640327
Recall: 1.0
F1: 0.8237179487179488
```

Conclusion

Classifier	Ein	Eout	Precision	Recall	F1
KNN (n=9)	.3762	.3782	.6802	.8318	.7480
SVM (13 f)	.3489	.2868	.6678	.9993	.8322
NB	.3447	.3676	.6615	.9149	.7679
LR	.3452	.3016	.7003	1.0000	.8237

SVM and Logistic Regression are among the best models

Thank you!

Questions?