MVP for project Luther <Li Zhang sf16-ds4>

Goal: predict the **rating** of a movie based on metrics such as gross, runtime, and whether it becomes to a certain genre.

Web-scraping: scraped data from imdb.com. The following is a example pandas data frame used for modeling. There are 3902 movies/rows in total after cleaning.

212	Megaforce	5333658	20000000	3.5	2561	['Sci-Fi', 'Action']	PG	1982-06-25	99	1.85
213	An Officer and a Gentleman	129795554	7500000	7	38444	['Drama', 'Romance']	R	1982-08-13	124	1.85
214	One from the Heart	900000	27000000	6.5	3830	['Drama', 'Musical', 'Romance']	R	1982-08-06	107	1.37
215	Poltergeist	76600000	10700000	7.4	106945	['Fantasy', 'Horror']	PG	1982-06-04	114	2.35
216	Porky's	105500000	25000000	6.2	31502	['Comedy']	R	1982-03-19	94	1.85
217	Rocky III	122823200	17000000	6.7	125223	['Drama', 'Sport']	PG	1982-05-28	99	1.37
218	Star Trek II: The Wrath of Khan	78900000	11000000	7.7	92453	['Action', 'Adventure', 'Sci-Fi']	PG	1982-06-04	113	2.35
219	The Thing	13782838	15000000	8.2	261714	['Horror', 'Mystery', 'Sci-Fi']	R	1982-06-25	109	2.35

Adding categorical dummy variables:

(1) type A dummy variables based on genre ('Drama', 'Comedy', etc)

(2) type B dummy variables based on MPAA rating ('R', 'PG', 'G')

The data frame now has size of (3902, 35) after adding the dummy variables (0 or 1) as features.

Find the most relevant features by calculating correlation between "rating" and all other features.

```
In [17]: df.corr()['rating'].sort_values(ascending=False)

Out[17]: rating        1.0000000000
         numvote       0.4826799998
         runtime       0.4128140099
         Drama         0.3396078128
         gross         0.2176408314
         Biography     0.1745699935
         R             0.1396622896
         History       0.1253871354
         War           0.1043518640
         Animation     0.0810181251
         budget        0.0557852143
         Crime         0.0492459099
         aspect        0.0399777100
         Western       0.0299270088
         Mystery       0.0184563322
         G             0.0147271979
         Sport         0.0121631406
         Musical       0.0082084803
         roi           0.0007497458
         Romance      -0.0053576852
         Music        -0.0091522934
         Adventure    -0.0177603582
         Thriller     -0.0520764048
         Sci-Fi       -0.0625707514
         Fantasy      -0.0678357365
         Family       -0.0780813511
         Action       -0.0946854480
         PG           -0.1439278240
         Comedy       -0.1745911050
         Horror       -0.2170142414
         Film-Noir              NaN
         Name: rating, dtype: float64
```

Keep the features that have a correlation greater than 0.1, we end up with

```
df_rating
```

	rating	numvote	runtime	Drama	gross	Biography	R	History	War	PG	Comedy	Horror
16	6.4	65168	118	0	47095453	0	0	0	0	1	1	0
254	8.3	273982	160	1	51600000	1	1	1	0	0	0	0
256	7.7	68025	99	0	2150000	0	1	0	0	0	0	0
269	6.2	4310	88	0	4000000	0	0	0	0	1	0	1
279	7.4	13088	164	1	26400000	0	0	1	0	1	0	0
283	6.3	4935	122	1	8800000	0	0	0	0	1	0	0
294	7.7	40168	97	1	10600000	0	1	0	0	0	1	0
295	8.5	742192	116	0	210609762	0	0	0	0	1	1	0
296	6.5	22340	80	0	21000000	0	0	0	0	1	0	0
297	8.0	153538	132	1	9929000	0	1	0	0	0	0	0
298	7.9	260380	97	1	38100000	0	1	0	0	0	1	0

After running various linear regression models, the best model so far is the Polynomial feature with degree = 2 along with Lasso option (alpha = 1e-2).

Train = 80%; Test = 20%;
y_predict is the predicted rating based on all other features
y_test is the observed rating
RMSE is the root mean square error of the prediction