Movie Score Prediction

Xi Chen

Kang Yang

Haoyang Li

Why this topic?

Watching movies has been a popular and common thing in our normal life. However, the strange thing is that we always lost in the plenty of movie resources. Whether this movie is good or suck? Hard to decide.
What if we could predict the quality of a movie? Is it good enough for me to watch?

Solution

We could rate each movie with a certain score based on widely accepted algorithms.
We collect a large amount of known movies as the training data and analyze all the related detail information by applying three algorithms learned from class--KNN, ANN, C5.0.
Finally, we abstract a module system where we can predicate a relatively accurate score for every movie.

Data Collection

The movie data derived from IMDB, which is the most popular and authoritative source for movie, TV and celebrity content.

After deleting all the records with NA values. This is the final dataset with 3801 records

	movie_imdb_link	num_user_for_reviews	language	country	content_rating	budget	title_year	actor_2_facebook_likes	aspect_ratio	movie_facebook_likes	imdb_score
an|christian film|christianity|...	http://www.imdb.com/title/tt4824308/?ref_=fn_tt_tt_1	102	English	USA	PG	5000000	2016	420	2.35	0	3.4
ence to avenged sevenfold|re...	http://www.imdb.com/title/tt4786282/?ref_=fn_tt_tt_1	95	English	USA	PG-13	4900000	2016	509	2.35	0	6.9
y best friend|gay man|gay m...	http://www.imdb.com/title/tt4438848/?ref_=fn_tt_tt_1	111	English	USA	R	35000000	2016	329	2.35	0	6.0
pronounced dead|recovery	http://www.imdb.com/title/tt4337690/?ref_=fn_tt_tt_1	29	English	USA	PG-13	5000000	2015	849	2.35	0	4.6
n|naked woman|witch	http://www.imdb.com/title/tt4263482/?ref_=fn_tt_tt_1	452	English	USA	R	3500000	2015	191	1.66	43000	6.8
n|christianity|falling into a ho...	http://www.imdb.com/title/tt4257926/?ref_=fn_tt_tt_1	55	English	USA	PG	13000000	2016	3000	1.85	16000	6.8
gent|repeated scene	http://www.imdb.com/title/tt4196776/?ref_=fn_tt_tt_1	297	English	UK	PG-13	120000000	2016	365	2.35	31000	7.1
n a job|gift|rape|substance ab...	http://www.imdb.com/title/tt4178092/?ref_=fn_tt_tt_1	279	English	USA	R	5000000	2015	562	2.35	15000	7.1
n a job|gift|rape|substance ab...	http://www.imdb.com/title/tt4178092/?ref_=fn_tt_tt_1	279	English	USA	R	5000000	2015	562	2.35	15000	7.1
gent|libya|mercenary|u.s. am...	http://www.imdb.com/title/tt4172430/?ref_=fn_tt_tt_1	219	English	USA	R	50000000	2016	726	2.35	44000	7.4
ght|hitman|kitten	http://www.imdb.com/title/tt4139124/?ref_=fn_tt_tt_1	84	English	USA	R	15000000	2016	622	2.35	0	6.4
ential election|reference to g...	http://www.imdb.com/title/tt4094724/?ref_=fn_tt_tt_1	94	English	France	R	10000000	2016	465	2.35	0	6.1
nd|skinhead|suspense	http://www.imdb.com/title/tt4062536/?ref_=fn_tt_tt_1	125	English	USA	R	5000000	2015	442	2.35	10000	7.1
|trapped	http://www.imdb.com/title/tt4052882/?ref_=fn_tt_tt_1	139	English	USA	PG-13	17000000	2016	350	2.35	0	6.8
sistance|virus	http://www.imdb.com/title/tt4046784/?ref_=fn_tt_tt_1	360	English	USA	PG-13	61000000	2015	960	2.35	24000	6.4
|nanny|surprise ending	http://www.imdb.com/title/tt3882082/?ref_=fn_tt_tt_1	155	English	USA	PG-13	10000000	2016	334	2.35	20000	6.0
y|krampus|santa claus	http://www.imdb.com/title/tt3850590/?ref_=fn_tt_tt_1	181	English	USA	PG-13	15000000	2015	658	2.35	27000	6.2
hop|party|tomboy	http://www.imdb.com/title/tt3850214/?ref_=fn_tt_tt_1	89	English	USA	R	7000000	2015	256	2.35	23000	7.3
ashed|male objectification|n...	http://www.imdb.com/title/tt3787590/?ref_=fn_tt_tt_1	60	English	UK	R	2000000	2015	625	1.85	0	6.1
emarriage|suburb|wedding	http://www.imdb.com/title/tt3760922/?ref_=fn_tt_tt_1	103	English	USA	PG-13	18000000	2016	312	2.35	19000	6.1
|northeast region of brazil|p...	http://www.imdb.com/title/tt3742378/?ref_=fn_tt_tt_1	26	Portuguese	Brazil	R	4000000	2015	9	2.35	0	7.9
ovie|football team|high scho...	http://www.imdb.com/title/tt3719896/?ref_=fn_tt_tt_1	20	English	USA	PG	20000000	2015	769	2.35	0	7.0
|secret|skype|webcam	http://www.imdb.com/title/tt3713166/?ref_=fn_tt_tt_1	309	English	USA	R	1000000	2014	305	1.85	13000	5.7
noking|male in shower|male ...	http://www.imdb.com/title/tt3707106/?ref_=fn_tt_tt_1	61	English	USA	R	10000000	2015	11000	2.35	0	5.3
t|london england|queen	http://www.imdb.com/title/tt3691740/?ref_=fn_tt_tt_1	106	English	UK	PG	140000000	2016	400	2.35	27000	6.8
iation|spy	http://www.imdb.com/title/tt3682448/?ref_=fn_tt_tt_1	355	English	USA	PG-13	40000000	2015	535	2.35	55000	7.6
martial art|silk road|sword	http://www.imdb.com/title/tt3672840/?ref_=fn_tt_tt_1	86	Mandarin	China	R	65000000	2015	18	2.35	0	6.1
ooperation|left for dead|nasa...	http://www.imdb.com/title/tt3659388/?ref_=fn_tt_tt_1	1023	English	USA	PG-13	108000000	2015	801	2.35	153000	8.1
ighbor|friendship|travel|trip	http://www.imdb.com/title/tt3622592/?ref_=fn_tt_tt_1	160	English	USA	PG-13	12000000	2015	558	2.35	0	6.4
ory|gangster|identical twins|...	http://www.imdb.com/title/tt3569230/?ref_=fn_tt_tt_1	174	English	UK	R	30000000	2015	154	2.35	43000	7.0
fusal to kill|wuxia	http://www.imdb.com/title/tt3508840/?ref_=fn_tt_tt_1	87	Mandarin	Taiwan	Not Rated	15000000	2015	103	1.37	0	6.4
fe|marvel cinematic universe|	http://www.imdb.com/title/tt3498820/?ref_=fn_tt_tt_1	1022	English	USA	PG-13	250000000	2016	19000	2.35	72000	8.2

Showing 1 to 32 of 3,801 entries

KNN

Step 1: Handle the missing data
Step 2: Nomalize the data
Step 3: Generate training dataset and test dataset
Step 4: Use knn algorithm to build the model and calculate wrong rate

Step 1: Handle the missing data

```
table1 <- na.omit(table)
```

Step 2: Normalize the data

```
mmnorm <-function(x,minx,maxx){
        z<-((x-minx)/(maxx-minx))
        return(z) }
```

Step 3: Traing and test dataset

- Store every fifth record in a “test” dataset starting with the first record

```
idx=seq(from=1,to=nrow(table_norm),by=5)
```

- Store the rest in the “training” dataset

```
training<-table_norm[-idx,]
```

Training

	critic	duration	directorFacebookLikes	actorFacebookLikes	gross	votedUsers	castFacebookLikes	posterFaces	reviews	country	budget	aspectRatio	score
2	0.37068966	0.4505119	0.0244782609	0.0625000000	4.068398e-01	0.278865211	0.0736223410	0.00000000	0.24451473	0.0245589446	0.07894737	0.0000000000	0.714
3	0.74014778	0.3788396	0.0000000000	0.0171875000	2.630802e-01	0.163255825	0.0178155406	0.02325581	0.19628385	0.0200564681	0.07894737	0.2435530086	0.675
4	1.00000000	0.4334471	0.9565217391	0.0421875000	5.892533e-01	0.677216100	0.1625614788	0.00000000	0.53370231	0.0204657842	0.07894737	0.4699140401	0.896
5	0.56773399	0.3242321	0.0206521739	0.0010000000	9.606571e-02	0.125579447	0.0028520092	0.02325581	0.14568096	0.0215873101	0.07894737	0.0687679083	0.649
7	0.39778325	0.2150171	0.0006521739	0.0012484375	2.640442e-01	0.174465708	0.0031002086	0.02325581	0.07629966	0.0212844162	0.04520918	0.0830945559	0.805
8	0.78078818	0.3549488	0.0000000000	0.0406250000	6.035345e-01	0.273804726	0.1400880118	0.09302326	0.22059696	0.0204657842	0.07894737	0.3381088825	0.766
9	0.46059113	0.3959044	0.0122608696	0.0390625000	3.970474e-01	0.190435441	0.0894629452	0.06976744	0.19213283	0.0204657842	0.07894737	0.0286532951	0.766
10	0.82758621	0.4982935	0.0000000000	0.0234375000	4.342491e-01	0.219933138	0.0372299118	0.00000000	0.59636292	0.0204657842	0.07894737	0.5644699140	0.688
12	0.49507389	0.2354949	0.0171739130	0.0007046875	2.213899e-01	0.195755134	0.0030804136	0.02325581	0.24550306	0.0163726238	0.07894737	0.0000000000	0.662
13	0.38423645	0.3890785	0.0244782609	0.0625000000	5.562515e-01	0.308940506	0.0738294276	0.04651163	0.36192924	0.0184192040	0.07894737	0.0143266476	0.740
14	0.55295567	0.3856655	0.0244782609	0.0625000000	1.174084e-01	0.107581614	0.0696739908	0.02325581	0.14034394	0.0176005719	0.07894737	0.1375358166	0.636
15	0.90147783	0.3617747	0.0000000000	0.0234375000	3.826683e-01	0.324642745	0.0312076500	0.00000000	0.50108717	0.0184192040	0.07894737	0.3381088825	0.727
17	0.86453202	0.4641638	0.0000000000	0.0406250000	8.195591e-01	0.589084005	0.1335358519	0.06976744	0.34018581	0.0180098879	0.04520918	0.3524355301	0.844
18	0.55049261	0.3378840	0.0109565217	0.0625000000	3.169782e-01	0.219379805	0.0823519559	0.09302326	0.09547341	0.0204657842	0.07894737	0.1661891117	0.662
19	0.55418719	0.2354949	0.0081739130	0.0156250000	2.353969e-01	0.158690677	0.0191433314	0.02325581	0.06720696	0.0184192040	0.04520918	0.1146131805	0.675
20	0.51847291	0.4334471	0.0000000000	0.0078125000	3.354455e-01	0.209629302	0.0139357118	0.00000000	0.15833169	0.0204657842	0.07894737	0.1862464183	0.766
22	0.42118227	0.4061433	0.0000000000	0.0013921875	1.383547e-01	0.125319646	0.0049396251	0.00000000	0.10772880	0.0163726238	0.07894737	0.0487106017	0.662
23	0.62561576	0.5085324	0.0000000000	0.0078125000	3.397150e-01	0.286156192	0.0139357118	0.13953488	0.18778415	0.0184192040	0.07894737	0.2378223496	0.818
24	0.30788177	0.2593857	0.0056086957	0.0250000000	9.215363e-02	0.088186540	0.0367061045	0.04651163	0.13144890	0.0147353596	0.07894737	0.0000000000	0.584
25	0.54802956	0.5597270	0.0000000000	0.0093750000	2.867186e-01	0.187016610	0.0108461620	0.00000000	0.51729591	0.0169456662	0.07894737	0.0000000000	0.727
27	0.63423645	0.3754266	0.0040869565	0.0328125000	5.354294e-01	0.161363248	0.0986676412	0.00000000	0.20181854	0.0204657842	0.07894737	0.2063037249	0.857
28	0.46305419	0.3208191	0.0231304348	0.0218750000	8.569692e-02	0.119766783	0.0406240007	0.00000000	0.14825064	0.0171093926	0.07894737	0.1260744986	0.558
29	0.79187192	0.2969283	0.0158695652	0.0046875000	8.575572e-01	0.247496241	0.0128789609	0.00000000	0.25479344	0.0122794634	0.05533063	0.4297994269	0.701
30	0.92241379	0.3617747	0.0000000000	0.0013796875	4.002075e-01	0.308934588	0.0031047767	0.00000000	0.29590828	0.0163726238	0.07894737	0.2292263610	0.805
32	0.74753695	0.5392491	0.0434782609	0.0328125000	5.377897e-01	0.329919237	0.0463295418	0.06976744	0.23443368	0.0163726238	0.07894737	0.2722063037	0.727

Test

	critic	duration	directorFacebookLikes	actorFacebookLikes	gross	votedUsers	castFacebookLikes	posterFaces	reviews	country	budget	aspectRatio	score
1	0.88916256	0.4812287	0.000000e+00	0.0015625000	1.000000000	0.524452895	0.0073607114	0.00000000	0.603478948	0.019401562	0.04048583	0.0945558739	0.81818
6	0.48152709	0.4061433	0.000000e+00	0.0375000000	0.442508383	0.226689723	0.0701277542	0.00000000	0.375765962	0.021120690	0.07894737	0.0000000000	0.59740
11	0.53325123	0.4505119	0.000000e+00	0.0281250000	0.263073965	0.142263483	0.0456671692	0.00000000	0.467681360	0.017109393	0.07894737	0.0000000000	0.58441
16	0.31650246	0.3856655	3.478261e-03	0.0343750000	0.186210128	0.088720936	0.0345606261	0.09302326	0.086380708	0.018419204	0.07894737	0.0000000000	0.64935
21	0.73645320	0.3959044	2.017391e-02	0.0234375000	0.344547722	0.267374223	0.0433800801	0.00000000	0.241945048	0.018828520	0.07894737	0.1604584527	0.70129
26	0.38669951	0.5358362	0.000000e+00	0.0453125000	0.866097589	0.469329650	0.0688608713	0.00000000	0.499505831	0.016372624	0.07894737	0.0744985673	0.79220
31	0.36822660	0.3344710	0.000000e+00	0.0375000000	0.490959816	0.243324048	0.0660667245	0.02325581	0.257363115	0.016372624	0.07894737	0.0000000000	0.74025
36	0.44950739	0.3856655	0.000000e+00	0.0013968750	0.528696281	0.191271063	0.0049000350	0.00000000	0.284245898	0.016372624	0.07894737	0.0000000000	0.57142
41	0.37315271	0.2354949	2.117391e-02	0.0015625000	0.251741331	0.059874219	0.0068247225	0.00000000	0.055742242	0.016372624	0.07894737	0.0286532951	0.61038
46	0.80418719	0.2935154	1.717391e-02	0.0265625000	0.266074867	0.275195457	0.0493231617	0.00000000	0.196481518	0.015553992	0.07894737	0.3696275072	0.70129
51	0.37561576	0.2696246	7.782609e-03	0.0234375000	0.119335704	0.131615219	0.0245900142	0.04651163	0.089345720	0.016372624	0.07894737	0.0659025788	0.64935
56	0.39532020	0.2901024	2.960870e-02	0.0015593750	0.171554883	0.031721683	0.0020206173	0.09302326	0.085194703	0.015144676	0.07894737	0.0859598854	0.76623
61	0.47167488	0.3071672	0.000000e+00	0.0265625000	0.062294294	0.082608230	0.0720752821	0.02325581	0.142122949	0.014407907	0.07894737	0.1260744986	0.49350
66	0.50123153	0.2013652	0.000000e+00	0.0015625000	0.385242872	0.393884572	0.0040123034	0.02325581	0.138960269	0.014326044	0.04520918	0.0773638968	0.87012
71	0.32389163	0.2559727	1.552174e-02	0.0078125000	0.134352714	0.069786283	0.0139037352	0.04651163	0.098833762	0.011870147	0.07894737	0.0000000000	0.46753
76	0.30665025	0.2764505	9.043478e-03	0.0359375000	0.197457390	0.103312366	0.0406300915	0.09302326	0.105356790	0.014326044	0.07894737	0.0000000000	0.54545
81	0.49261084	0.2047782	7.434783e-03	0.0171875000	0.317429798	0.149282827	0.0236261477	0.04651163	0.125123542	0.014735360	0.07894737	0.2550143266	0.70129
86	0.54433498	0.3174061	2.882609e-02	0.0031250000	0.122836298	0.075928579	0.0076835229	0.04651163	0.098043092	0.015553992	0.06882591	0.1060171920	0.63636
91	0.35344828	0.2081911	1.108696e-02	0.0281250000	0.285846430	0.287274694	0.0311436968	0.00000000	0.097054754	0.013507411	0.07894737	0.0945558739	0.85714
96	0.22906404	0.2354949	1.552174e-02	0.0359375000	0.190021128	0.161098713	0.0690192316	0.04651163	0.195097845	0.003110784	0.07894737	0.0401146132	0.66233
101	0.28325123	0.2081911	1.082609e-02	0.1359375000	0.079756591	0.048749555	0.1407823611	0.00000000	0.124135205	0.013098095	0.07894737	0.0000000000	0.51948
106	0.27955665	0.2593857	6.521739e-03	0.0008296875	0.137255334	0.062991823	0.0011633396	0.09302326	0.044672860	0.012688779	0.04048583	0.0315186246	0.61038
111	0.36206897	0.4095563	7.782609e-03	0.0328125000	0.381317643	0.228236689	0.0813317497	0.02325581	0.374579957	0.012279463	0.07894737	0.0000000000	0.77922
116	0.58743842	0.3105802	9.565217e-01	0.0359375000	0.270009306	0.580521246	0.0906887153	0.00000000	0.530539632	0.012279463	0.07894737	0.0429799427	0.87012
121	0.49876847	0.2218430	3.000000e-03	0.0015625000	0.526934178	0.249534401	0.0039316005	0.00000000	0.178493773	0.012279463	0.07152497	0.1661891117	0.77922
126	0.63423645	0.2662116	0.000000e+00	0.0406250000	0.238019232	0.317387864	0.0914515859	0.02325581	0.145680965	0.012279463	0.07894737	0.1805157593	0.70129

Step 4: KNN and wrong rate

```r
# use knn algorithm to build the model
predict<-knn(training[,-13],test[,-13],training[,13],k=1)

# combine the prediction with the test data and calculate the wrong rate
results<-cbind(test, predict)
table(results[,13],results[,14])
wrong<-results[,13]!=results[,14]
rate<-sum(wrong)/length(wrong)
rate
```

k	1	5	10
wrong rate	0.9250986	0.9421813	0.9461235

Why?

We could consider the formular

$$d_{\text{Euclidean}}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2}$$

where $\mathbf{x} = x_1, x_2, \ldots, x_m$, and $\mathbf{y} = y_1, y_2, \ldots, y_m$

represent the $\boxed{m \text{ attributes}}$

C 5.0

(1) Data preprocessing
(2) Generate training data and testing data
(3) Use C5.0 to analyze the data
(4) Use test data to calculate the wrong data

Data Preprocessing

1. Data Cleaning → Deal with the missing values

Filter

	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes	actor_2_name	actor_1_facebook_likes	gross
1	Color	James Cameron	723	178	0	855	Joel David Moore	1000	760505847
2	Color	Gore Verbinski	302	169	563	1000	Orlando Bloom	40000	309404152
3	Color	Sam Mendes	602	148	0	161	Rory Kinnear	11000	200074175
4	Color	Christopher Nolan	813	164	22000	23000	Christian Bale	27000	448130642
5		Doug Walker	NA	NA	131	NA	Rob Walker	131	NA
6	Color	Andrew Stanton	462	132	475	530	Samantha Morton	640	73058679
7	Color	Sam Raimi	392	156	0	4000	James Franco	24000	336530303
8	Color	Nathan Greno	324	100	15	284	Donna Murphy	799	200807262
9	Color	Joss Whedon	635	141	0	19000	Robert Downey Jr.	26000	458991599
10	Color	David Yates	375	153	282	10000	Daniel Radcliffe	25000	301956980
11	Color	Zack Snyder	673	183	0	2000	Lauren Cohan	15000	330249062
12	Color	Bryan Singer	434	169	0	903	Marlon Brando	18000	200069408
13	Color	Marc Forster	403	106	395	393	Mathieu Amalric	451	168368427
14	Color	Gore Verbinski	313	151	563	1000	Orlando Bloom	40000	423032628
15	Color	Gore Verbinski	450	150	563	1000	Ruth Wilson	40000	89289910
16	Color	Zack Snyder	733	143	0	748	Christopher Meloni	15000	291021565
17	Color	Andrew Adamson	258	150	80	201	Pierfrancesco Favino	22000	141614023
18	Color	Joss Whedon	703	173	0	19000	Robert Downey Jr.	26000	623279547
19	Color	Rob Marshall	448	136	252	1000	Sam Claflin	40000	241063875

Data Preprocessing

2. Data Transformation → Normalization: Scaling attribute values to fall within a specified range

Before

```
> head(table_new)
  critic duration directorFacebookLikes actorFacebookLikes     gross votedUsers castFacebookLikes posterFaces reviews
1    723      178                     0               1000 760505847     886204              4834           0    3054
2    302      169                   563              40000 309404152     471220             48350           0    1238
3    602      148                     0              11000 200074175     275868             11700           1     994
4    813      164                 22000              27000 448130642    1144337            106759           0    2701
5    462      132                   475                640  73058679     212204              1873           1     738
6    392      156                     0              24000 336530303     383056             46055           0    1902
     budget aspectRatio movieFacebookLikes score
1 237000000        1.78              33000   7.9
2 300000000        2.35                  0   7.1
3 245000000        2.35              85000   6.8
4 250000000        2.35             164000   8.5
5 263700000        2.35              24000   6.6
6 258000000        2.35                  0   6.2
>
```

Data Preprocessing

2. Data Transformation → Normalization: Scaling attribute values to fall within a specified range [0, 1]

```
mmnorm <-function(x,minx,maxx) {z<-((x-minx)/(maxx-minx))
return(z)
}
```

Console ~/

```
> head(table_norm)
     critic  duration directorFacebookLikes actorFacebookLikes      gross votedUsers castFacebookLikes posterFaces   reviews    country
1 0.8891626 0.4812287            0.00000000          0.0015625 1.00000000  0.5244529       0.007360711  0.00000000 0.6034789 0.01940156
2 0.3706897 0.4505119            0.02447826          0.0625000 0.40683981  0.2788652       0.073622341  0.00000000 0.2445147 0.02455894
3 0.7401478 0.3788396            0.00000000          0.0171875 0.26308023  0.1632558       0.017815541  0.02325581 0.1962839 0.02005647
4 1.0000000 0.4334471            0.95652174          0.0421875 0.58925329  0.6772161       0.162561479  0.00000000 0.5337023 0.02046578
5 0.5677340 0.3242321            0.02065217          0.0010000 0.09606571  0.1255794       0.002852009  0.02325581 0.1456810 0.02158731
6 0.4815271 0.4061433            0.00000000          0.0375000 0.44250838  0.2266897       0.070127754  0.00000000 0.3757660 0.02112069
      budget aspectRatio     score
1 0.04048583  0.09455587 0.8181818
2 0.07894737  0.00000000 0.7142857
3 0.07894737  0.24355301 0.6753247
4 0.07894737  0.46991404 0.8961039
5 0.07894737  0.06876791 0.6493506
6 0.07894737  0.00000000 0.5974026
> 
```

Training Data and Testing Data

Get every five rows as test data

test	761 obs. of 13 variables
training	3040 obs. of 13 variables

Decision Tree

- A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represents classification rules.
- A decision tree consists of 3 types of nodes:
  - ➢ Decision nodes
  - ➢ Chance nodes
  - ➢ End nodes

Decision Tree

For some applications, especially those with many attributes, it may be useful to know how the individual attributes contribute to the classifier.

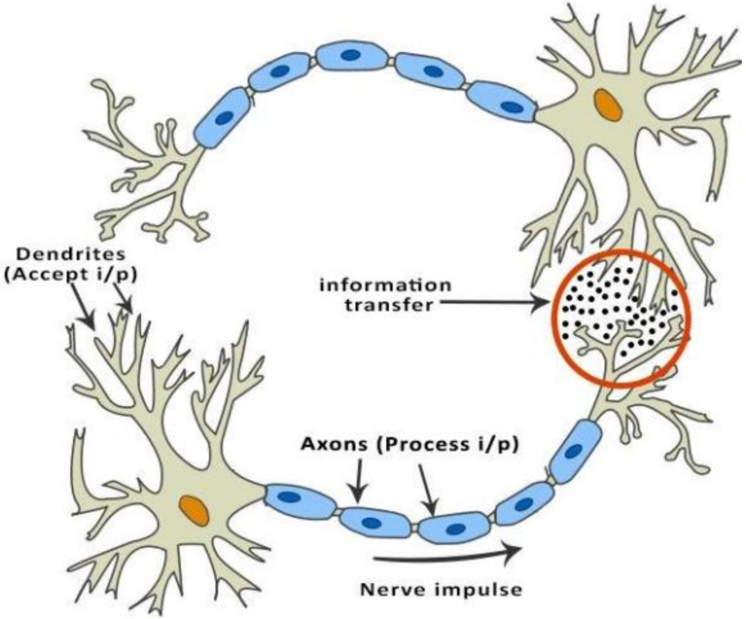
The figure before each attribute is the percentage of training cases in movie_metadata for which the value of that attribute is known and is used in predicting a class.

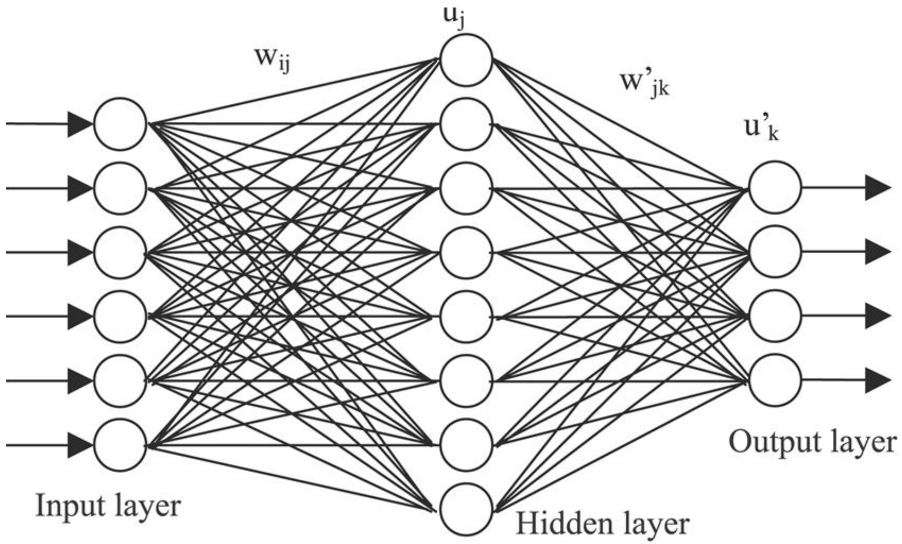
```
Attribute usage:

100.00% votedUsers
 91.45% duration
 81.02% budget
 64.24% castFacebookLikes
 59.77% reviews
 59.57% gross
 50.76% posterFaces
 48.85% critic
 48.16% aspectRatio
 47.80% directorFacebookLikes
 37.99% movieFacebookLikes
 34.44% actorFacebookLikes
```

Wrong Rate

```
> wrong<-results[,13]!=results[,14]
> rate<-sum(wrong)/length(wrong)
> rate
[1] 0.9198423
```

ANN Artificial Neural Network





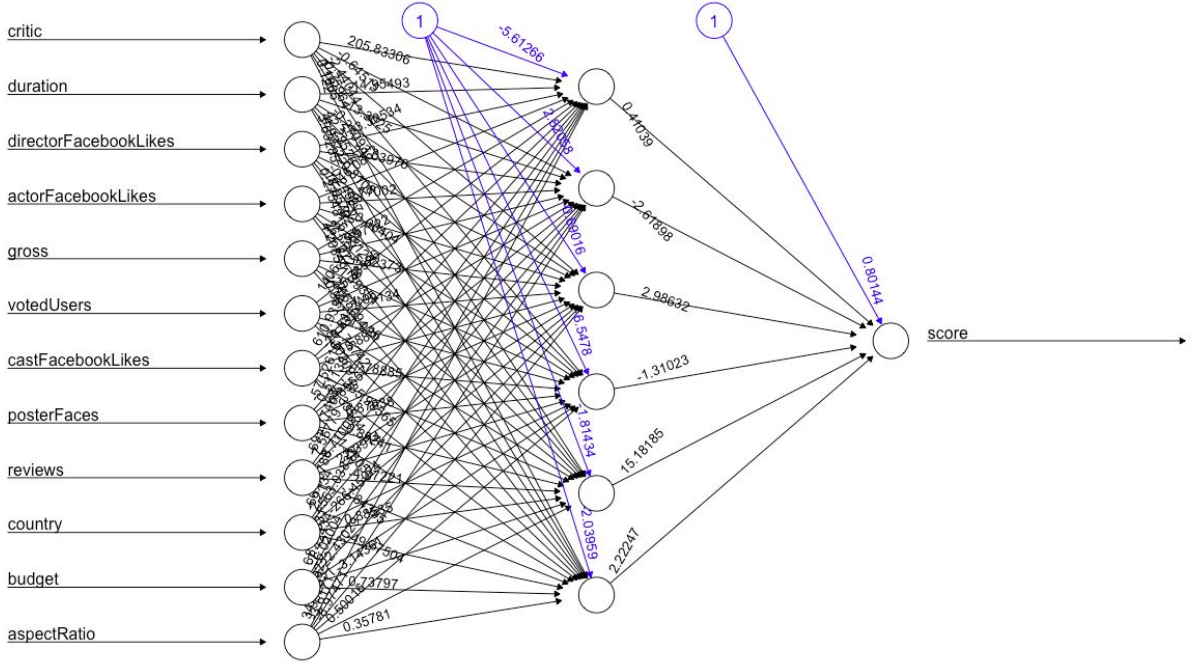
ANN

- Normalization → Comparable range of 0 to 1
- Training → Supervised Training
- Build & Test → Iterative Learning Process
- Wrong Rate → Calculate Error Rate

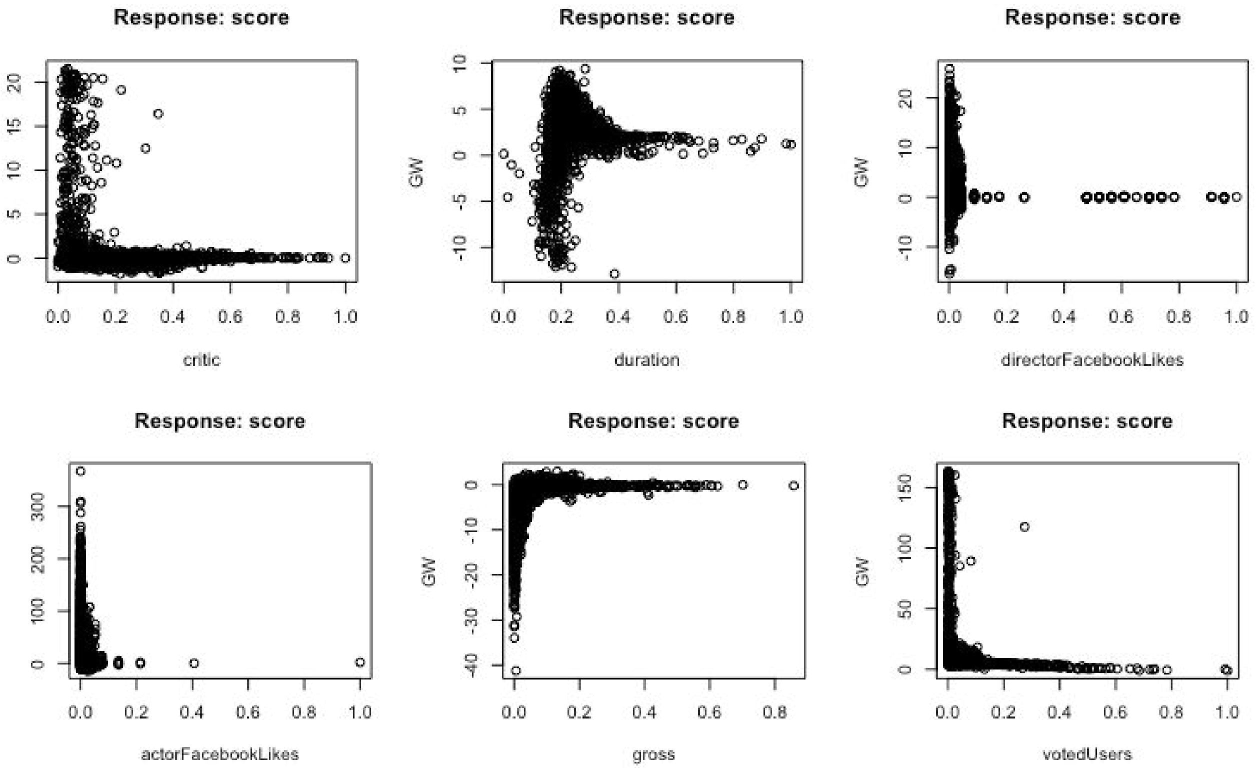
How many nodes in the hidden layer ?

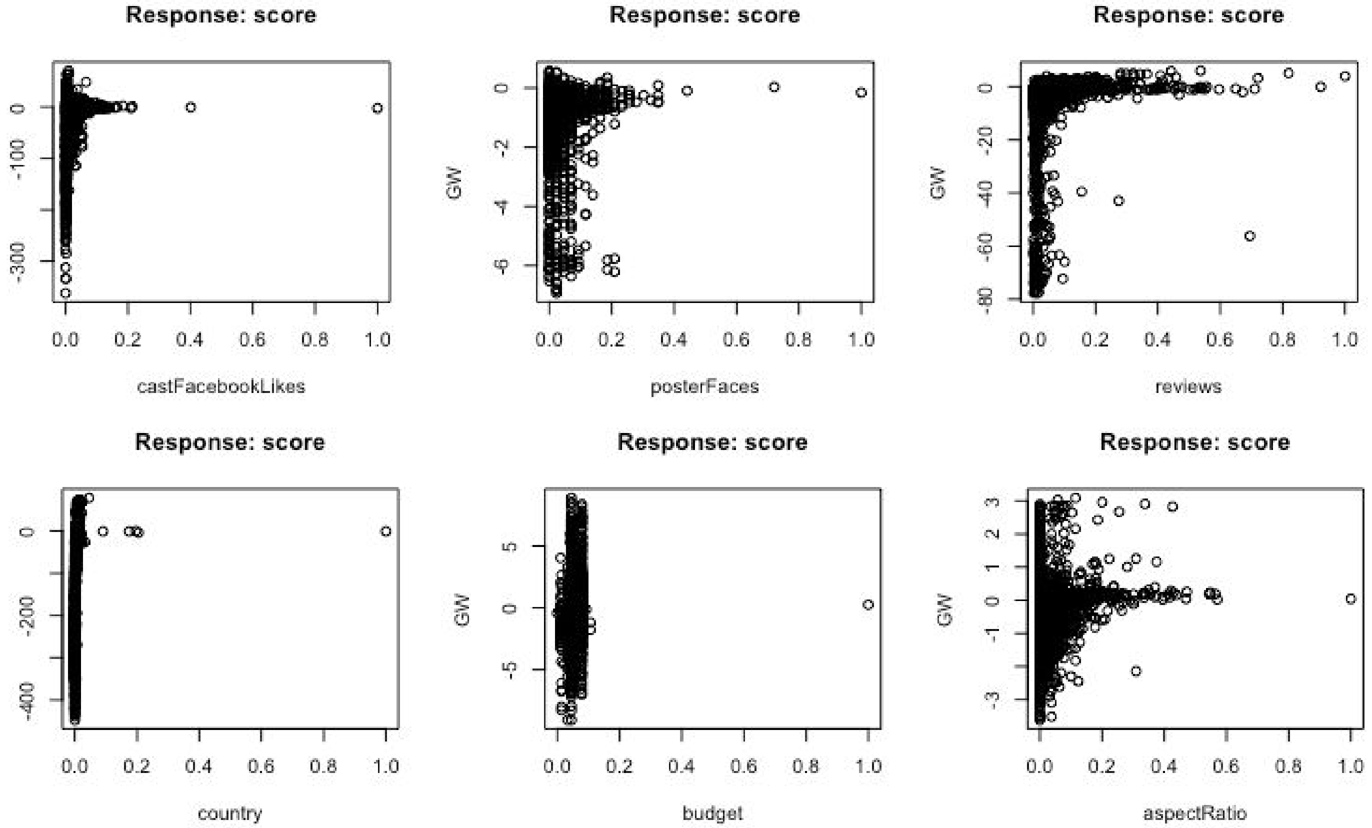
```
nn = neuralnet(score ~ critic + duration + directorFacebookLikes + actorFacebookLikes + gross +
               votedUsers + castFacebookLikes + posterFaces + reviews + country + budget +
               aspectRatio, data = training, hidden = 6, err.fct = "sse", linear.output = FALSE)
```

Train_WR → 0.22171052
Test_WR → 0.53552631



Generalized Weights for Different Covariates




Response: score
castFacebookLikes
Response: score
GW
posterFaces
Response: score
GW
reviews
Response: score
country
Response: score
GW
budget
Response: score
GW
aspectRatio

Summary

ANN
0.5355263

KNN
0.9250986

C5.0
0.9198423

The performance of ANN is better than KNN and C5.0 Decision Tree. It built the module with lowest wrong rate to predict the movie rate.

Thank You