In [ ]:
```
%pip install -U pip
%pip install -U setuptools wheel
%pip install pandas
%pip install numpy
%pip install sklearn
%pip install matplotlib
%pip install seaborn
%pip install autogluon
%pip install autogluon.eda
```

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

In [65]:
```
df_all = pd.read_csv('./DataFiles/modelTestDataMthFormatted.csv')
df = df_all[:50000]
```

In [36]:
```
from sklearn.model_selection import train_test_split
```

In [52]:
```
df.head()
```

Out[52]:

	userId	movieId	userLikedTheMovie	title	yearRatingMade	mthRatingMade	aver
0	1	296	1	Pulp Fiction (1994)	2006	4	
1	1	306	0	Three Colors: Red (Trois couleurs: Rouge) (1994)	2006	4	
2	1	307	1	Three Colors: Blue (Trois couleurs: Bleu) (1993)	2006	4	
3	1	665	1	Underground (1995)	2006	4	
4	1	899	0	Singin' in the Rain (1952)	2006	4	

5 rows × 29 columns

In [66]:
```
train_data, test_data = train_test_split(df, test_size=0.2, shuffle=True)
```

In [58]:
```
print(df.columns)
```

```
Index(['userId', 'movieId', 'userLikedTheMovie', 'title', 'yearRatingMade',
       'mthRatingMade', 'averageRating', 'numVotes', 'movieReleaseYear',
       'Genre_(no genres listed)', 'Genre_Action', 'Genre_Adventure',
       'Genre_Animation', 'Genre_Children', 'Genre_Comedy', 'Genre_Crime',
       'Genre_Documentary', 'Genre_Drama', 'Genre_Fantasy', 'Genre_Film-Noi
r',
       'Genre_Horror', 'Genre_IMAX', 'Genre_Musical', 'Genre_Mystery',
       'Genre_Romance', 'Genre_Sci-Fi', 'Genre_Thriller', 'Genre_War',
       'Genre_Western'],
      dtype='object')
```

In [59]:
```
model_target = 'userLikedTheMovie'
```

In [13]:
```
import autogluon.eda.analysis as eda
import autogluon.eda.visualization as viz
import autogluon.eda.auto as auto
```

In [67]:
```
auto.dataset_overview(train_data=train_data, test_data=test_data, label=mode
```

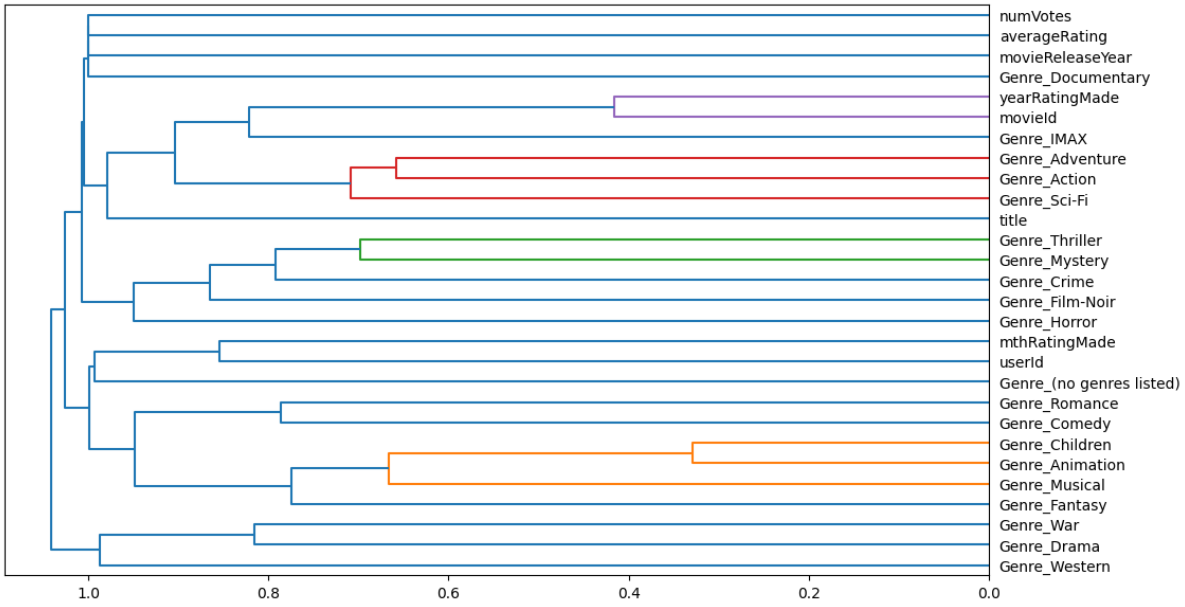
`train_data` dataset summary

	count	unique	top	freq	mean	std	min
Genre_(no genres listed)	40000	2			0.000425	0.020611	0.0
Genre_Action	40000	2			0.29645	0.456698	0.0
Genre_Adventure	40000	2			0.228725	0.420017	0.0
Genre_Animation	40000	2			0.063375	0.243639	0.0
Genre_Children	40000	2			0.08005	0.271374	0.0
Genre_Comedy	40000	2			0.3606	0.480181	0.0
Genre_Crime	40000	2			0.16845	0.37427	0.0
Genre_Documentary	40000	2			0.010975	0.104186	0.0
Genre_Drama	40000	2			0.45215	0.497711	0.0
Genre_Fantasy	40000	2			0.105775	0.307553	0.0
Genre_Film-Noir	40000	2			0.007775	0.087834	0.0
Genre_Horror	40000	2			0.07135	0.257412	0.0
Genre_IMAX	40000	2			0.04175	0.20002	0.0
Genre_Musical	40000	2			0.03825	0.191802	0.0
Genre_Mystery	40000	2			0.07885	0.269508	0.0
Genre_Romance	40000	2			0.196775	0.397566	0.0
Genre_Sci-Fi	40000	2			0.1683	0.374137	0.0
Genre_Thriller	40000	2			0.272375	0.445187	0.0
Genre_War	40000	2			0.0544	0.226808	0.0
Genre_Western	40000	2			0.0186	0.135109	0.0
averageRating	39978	77			7.276705	0.93146	1.5
movieId	40000	5990			18285.45425	35155.993118	1.0
movieReleaseYear	39989	103			1993.775788	38.384134	500.0
mthRatingMade	40000	10			5.635275	3.232424	1.0
numVotes	39978	5854			395346.353795	484806.505902	44.0
title	40000	5990	Forrest Gump (1994)	172			
userId	40000	406			195.1558	116.293682	1.0
userLikedTheMovie	40000	2			0.529525	0.499134	0.0
yearRatingMade	40000	24			2007.1517	7.363884	1996.0

`test_data` dataset summary

	count	unique	top	freq	mean	std	min	
Genre_(no genres listed)	10000	2			0.0003	0.017319	0.0	
Genre_Action	10000	2			0.2936	0.455434	0.0	
Genre_Adventure	10000	2			0.227	0.418914	0.0	
Genre_Animation	10000	2			0.0631	0.243155	0.0	
Genre_Children	10000	2			0.0809	0.272695	0.0	
Genre_Comedy	10000	2			0.3577	0.479347	0.0	
Genre_Crime	10000	2			0.1706	0.376178	0.0	
Genre_Documentary	10000	2			0.0131	0.113709	0.0	
Genre_Drama	10000	2			0.4565	0.498129	0.0	
Genre_Fantasy	10000	2			0.1101	0.31303	0.0	
Genre_Film-Noir	10000	2			0.0085	0.091807	0.0	
Genre_Horror	10000	2			0.0708	0.256503	0.0	
Genre_IMAX	10000	2			0.0443	0.205771	0.0	
Genre_Musical	10000	2			0.0363	0.187045	0.0	
Genre_Mystery	10000	2			0.08	0.271307	0.0	
Genre_Romance	10000	2			0.1918	0.393736	0.0	
Genre_Sci-Fi	10000	2			0.1643	0.370566	0.0	
Genre_Thriller	10000	2			0.2694	0.44367	0.0	
Genre_War	10000	2			0.0504	0.21878	0.0	
Genre_Western	10000	2			0.0181	0.13332	0.0	
averageRating	9994	65			7.292055	0.918853	1.9	
movieId	10000	3278			18996.5822	35913.861931	1.0	10
movieReleaseYear	9998	102			1993.835367	39.150087	500.0	19
mthRatingMade	10000	10			5.6454	3.231393	1.0	
numVotes	9994	3260			393259.717931	472471.5575	37.0	757
title	10000	3278	Matrix, The (1999)	40				
userId	10000	405			196.0072	116.63063	1.0	
userLikedTheMovie	10000	2			0.5333	0.498915	0.0	
yearRatingMade	10000	24			2007.3197	7.395401	1996.0	20

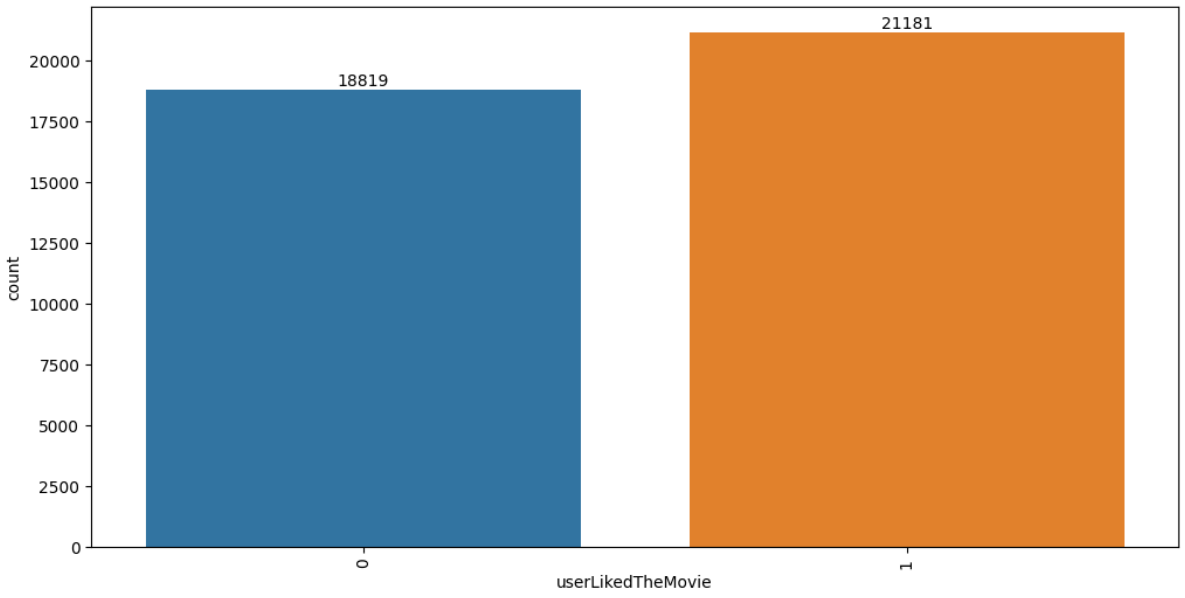
Feature Distance



In [68]:
```
auto.target_analysis(train_data=train_data, label=model_target)
```

Target variable analysis

	count	mean	std	min	25%	50%	75%	max	dtypes	uniqu
userLikedTheMovie	40000	0.529525	0.499134	0.0	0.0	1.0	1.0	1.0	int64	



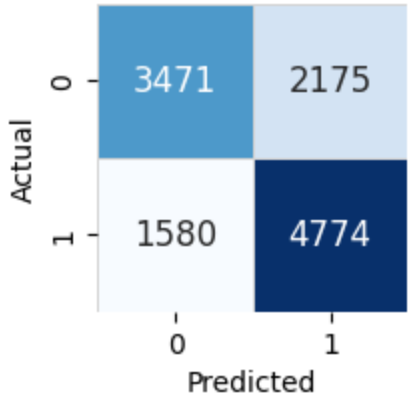
Target variable correlations

- ⚠ no fields with absolute correlation greater than `0.5` found for target variable `userLikedTheMovie`.

In [69]:
```
auto.quick_fit(train_data, model_target, show_feature_importance_barplots=Tr
```

```
No path specified. Models will be saved in: "AutogluonModels/ag-20230531_05
5600/"
```

Model Prediction for userLikedTheMovie

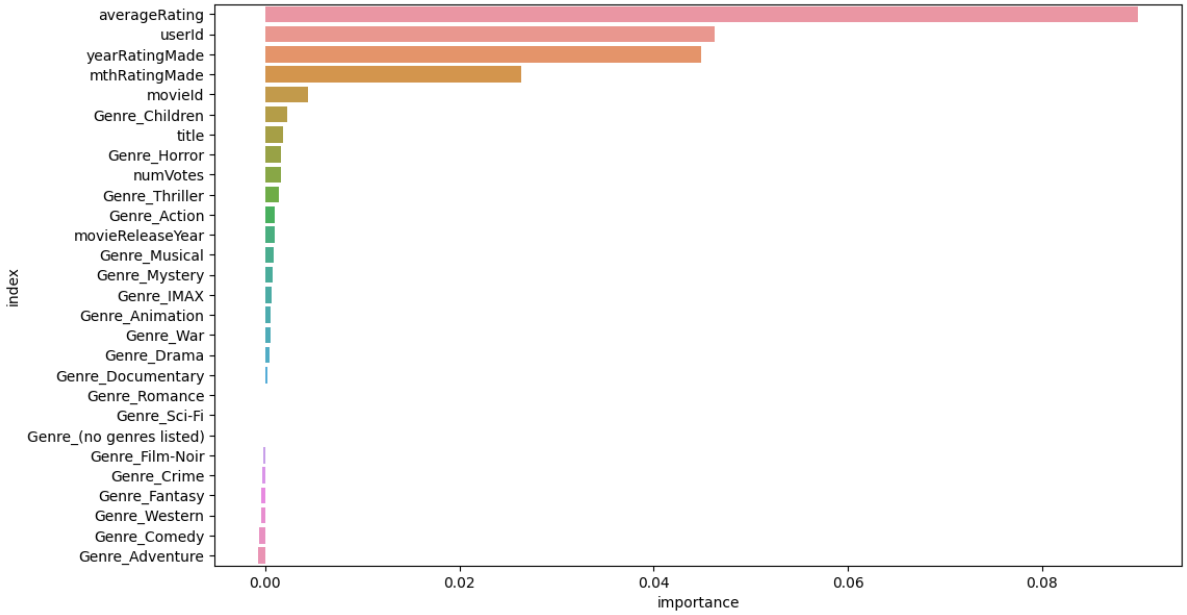


Model Leaderboard

	model	score_test	score_val	pred_time_test	pred_time_val	fit_time	pred_time_
0	LightGBMXT	0.687083	0.708	0.768155	0.216482	5.695906	

Feature Importance for Trained Model

	importance	stddev	p_value	n	p99_high	p99_low
averageRating	8.984000e-02	0.006290	0.000003	5	0.102792	0.076888
userId	4.632000e-02	0.003926	0.000006	5	0.054403	0.038237
yearRatingMade	4.488000e-02	0.003526	0.000005	5	0.052140	0.037620
mthRatingMade	2.640000e-02	0.002963	0.000019	5	0.032501	0.020299
movieId	4.440000e-03	0.002794	0.011867	5	0.010193	-0.001313
Genre_Children	2.240000e-03	0.001479	0.013814	5	0.005286	-0.000806
title	1.800000e-03	0.002877	0.117222	5	0.007725	-0.004125
Genre_Horror	1.680000e-03	0.001616	0.040374	5	0.005008	-0.001648
numVotes	1.680000e-03	0.002766	0.123004	5	0.007376	-0.004016
Genre_Thriller	1.440000e-03	0.001670	0.063022	5	0.004878	-0.001998
Genre_Action	1.040000e-03	0.003064	0.245064	5	0.007349	-0.005269
movieReleaseYear	1.000000e-03	0.001954	0.158203	5	0.005024	-0.003024
Genre_Musical	9.200000e-04	0.000576	0.011685	5	0.002106	-0.000266
Genre_Mystery	7.200000e-04	0.000716	0.043823	5	0.002193	-0.000753
Genre_IMAX	6.800000e-04	0.000965	0.095186	5	0.002668	-0.001308
Genre_Animation	5.600000e-04	0.000921	0.122746	5	0.002456	-0.001336
Genre_War	5.200000e-04	0.000965	0.147408	5	0.002508	-0.001468
Genre_Drama	4.400000e-04	0.001424	0.263821	5	0.003372	-0.002492
Genre_Documentary	2.800000e-04	0.000729	0.219534	5	0.001782	-0.001222
Genre_Romance	4.000000e-05	0.002114	0.484138	5	0.004392	-0.004312
Genre_Sci-Fi	2.220446e-17	0.001871	0.500000	5	0.003852	-0.003852
Genre_(no genres listed)	0.000000e+00	0.000000	0.500000	5	0.000000	0.000000
Genre_Film-Noir	-2.000000e-04	0.000283	0.905498	5	0.000382	-0.000782
Genre_Crime	-3.200000e-04	0.001016	0.739973	5	0.001772	-0.002412
Genre_Fantasy	-3.600000e-04	0.000639	0.861956	5	0.000955	-0.001675
Genre_Western	-4.400000e-04	0.000555	0.924528	5	0.000703	-0.001583
Genre_Comedy	-6.400000e-04	0.002066	0.736697	5	0.003614	-0.004894
Genre_Adventure	-6.800000e-04	0.001346	0.839097	5	0.002092	-0.003452



Rows with the highest prediction error

Rows in this category worth inspecting for the causes of the error

	userId	movieId	title	yearRatingMade	mthRatingMade	averageRating	numV
42754	343	1924	Plan 9 from Outer Space (1959)	2005	2	3.9	394
9213	72	611	Hellraiser: Bloodline (1996)	2001	1	5.0	23
5022	34	374	Richie Rich (1994)	2011	8	5.4	71
42913	345	891	Halloween: The Curse of Michael Myers (Halloween 6: The Curse of Michael Myers) (1995)	2001	2	4.7	39
20251	166	2643	Superman IV: The Quest for Peace (1987)	2000	1	3.7	505
4996	33	50	Usual Suspects, The (1995)	2019	2	8.5	1103
2310	12	5796	Casino Royale (1967)	2007	12	5.0	31
22568	177	858	Godfather, The (1972)	1999	6	9.2	1908
9926	74	318	Shawshank Redemption, The (1994)	1996	8	9.3	2743
6637	49	256	Junior (1994)	1996	6	4.7	70

10 rows × 32 columns

Rows with the least distance vs other class

Rows in this category are the closest to the decision boundary vs the other class and are good candidates for additional labeling

	userId	movieId	title	yearRatingMade	mthRatingMade	averageRating	numVot
23406	181	104879	Prisoners (2013)	2019	1	8.1	745158
42669	342	1584	Contact (1997)	1997	9	7.5	28259
7115	57	6373	Bruce Almighty (2003)	2006	2	6.8	415515
8897	70	1247	Graduate, The (1967)	2009	8	8.0	27988
15067	120	155	Beyond Rangoon (1995)	2000	5	6.5	533
9114	72	224	Don Juan DeMarco (1995)	2001	1	6.7	52136
35174	276	3450	Grumpy Old Men (1993)	2000	8	7.0	5147
49249	401	30812	Aviator, The (2004)	2005	9	7.5	371058
37203	290	107883	The Lunchbox (2013)	2017	5	7.8	5811
12483	92	5010	Black Hawk Down (2001)	2006	5	7.7	406974

10 rows × 32 columns

In [62]:
```
from autogluon.tabular import TabularDataset, TabularPredictor
```

In [70]:
```
metric = 'accuracy'
predictor = TabularPredictor(label=model_target,eval_metric=metric).fit(trai
predictions = predictor.predict(test_data)
```

```
No path specified. Models will be saved in: "AutogluonModels/ag-20230531_05
5738/"
Beginning AutoGluon training ...
AutoGluon will save models to "AutogluonModels/ag-20230531_055738/"
AutoGluon Version:  0.7.0
Python Version:     3.9.13
Operating System:   Linux
Platform Machine:   x86_64
Platform Version:   #1 SMP Thu May 4 09:55:30 UTC 2023
Train Data Rows:    40000
Train Data Columns: 28
Label Column: userLikedTheMovie
Preprocessing data ...
AutoGluon infers your prediction problem is: 'binary' (because only two uni
que label-values observed).
        2 unique label values:  [1, 0]
        If 'binary' is not the correct problem_type, please manually specif
y the problem_type parameter during predictor init (You may specify problem
_type as one of: ['binary', 'multiclass', 'regression'])
Selected class <--> label mapping:  class 1 = 1, class 0 = 0
Using Feature Generators to preprocess the data ...
Fitting AutoMLPipelineFeatureGenerator...
        Available Memory:                    6569.52 MB
        Train Data (Original)  Memory Usage: 11.96 MB (0.2% of available me
mory)
        Inferring data type of each feature based on column values. Set fea
ture_metadata_in to manually specify special dtypes of the features.
        Stage 1 Generators:
                Fitting AsTypeFeatureGenerator...
                        Note: Converting 20 features to boolean dtype as th
ey only contain 2 unique values.
        Stage 2 Generators:
                Fitting FillNaFeatureGenerator...
        Stage 3 Generators:
                Fitting IdentityFeatureGenerator...
                Fitting CategoryFeatureGenerator...
                        Fitting CategoryMemoryMinimizeFeatureGenerator...
                Fitting TextSpecialFeatureGenerator...
                        Fitting BinnedFeatureGenerator...
                        Fitting DropDuplicatesFeatureGenerator...
                Fitting TextNgramFeatureGenerator...
                        Fitting CountVectorizer for text features: ['titl
e']
                        CountVectorizer fit with vocabulary size = 95
                Warning: Due to memory constraints, ngram feature count is
being reduced. Allocate more memory to maximize model quality.
                Reducing Vectorizer vocab size from 95 to 24 to avoid OOM e
rror
        Stage 4 Generators:
                Fitting DropUniqueFeatureGenerator...
        Types of features in original data (raw dtype, special dtypes):
                ('float', [])       :  3 | ['averageRating', 'numVotes',
'movieReleaseYear']
                ('int', [])         : 24 | ['userId', 'movieId', 'yearRati
ngMade', 'mthRatingMade', 'Genre_(no genres listed)', ...]
                ('object', ['text']) :  1 | ['title']
```

```
        Types of features in processed data (raw dtype, special dtypes):
                ('category', ['text_as_category'])  :  1 | ['title']
                ('float', [])                       :  3 | ['averageRatin
g', 'numVotes', 'movieReleaseYear']
                ('int', [])                         :  4 | ['userId', 'movi
eId', 'yearRatingMade', 'mthRatingMade']
                ('int', ['binned', 'text_special']) : 16 | ['title.char_cou
nt', 'title.word_count', 'title.capital_ratio', 'title.lower_ratio', 'titl
e.digit_ratio', ...]
                ('int', ['bool'])                   : 20 | ['Genre_(no genr
es listed)', 'Genre_Action', 'Genre_Adventure', 'Genre_Animation', 'Genre_C
hildren', ...]
                ('int', ['text_ngram'])             : 25 | ['__nlp__.1990',
'__nlp__.1992', '__nlp__.1993', '__nlp__.1994', '__nlp__.1995', ...]
        3.9s = Fit runtime
        28 features in original data used to generate 69 features in proces
sed data.
        Train Data (Processed) Memory Usage: 5.76 MB (0.1% of available mem
ory)
Data preprocessing and feature engineering runtime = 4.05s ...
AutoGluon will gauge predictive performance using evaluation metric: 'accur
acy'
        To change this, specify the eval_metric parameter of Predictor()
Automatically generating train/validation split with holdout_frac=0.0625, T
rain Rows: 37500, Val Rows: 2500
Fitting 13 L1 models ...
Fitting model: KNeighborsUnif ...
        0.5776   = Validation score   (accuracy)
        0.06s    = Training   runtime
        0.72s    = Validation runtime
Fitting model: KNeighborsDist ...
        0.5768   = Validation score   (accuracy)
        0.06s    = Training   runtime
        0.42s    = Validation runtime
Fitting model: LightGBMXT ...
        0.6952   = Validation score   (accuracy)
        8.23s    = Training   runtime
        0.14s    = Validation runtime
Fitting model: LightGBM ...
        0.7112   = Validation score   (accuracy)
        7.73s    = Training   runtime
        0.18s    = Validation runtime
Fitting model: RandomForestGini ...
        0.626    = Validation score   (accuracy)
        18.72s   = Training   runtime
        0.38s    = Validation runtime
Fitting model: RandomForestEntr ...
        0.636    = Validation score   (accuracy)
        19.83s   = Training   runtime
        0.26s    = Validation runtime
Fitting model: CatBoost ...
        0.7248   = Validation score   (accuracy)
        69.44s   = Training   runtime
        0.03s    = Validation runtime
Fitting model: ExtraTreesGini ...
        0.6108   = Validation score   (accuracy)
```

```
        16.89s   = Training   runtime
        0.25s    = Validation runtime
Fitting model: ExtraTreesEntr ...
        0.6112   = Validation score   (accuracy)
        17.1s    = Training   runtime
        0.4s     = Validation runtime
Fitting model: NeuralNetFastAI ...
        0.6816   = Validation score   (accuracy)
        114.15s  = Training   runtime
        0.14s    = Validation runtime
Fitting model: XGBoost ...
        0.7236   = Validation score   (accuracy)
        11.22s   = Training   runtime
        0.06s    = Validation runtime
Fitting model: NeuralNetTorch ...
        0.6688   = Validation score   (accuracy)
        189.66s  = Training   runtime
        0.06s    = Validation runtime
Fitting model: LightGBMLarge ...
        0.7216   = Validation score   (accuracy)
        10.82s   = Training   runtime
        0.14s    = Validation runtime
Fitting model: WeightedEnsemble_L2 ...
        0.728    = Validation score   (accuracy)
        3.11s    = Training   runtime
        0.0s     = Validation runtime
AutoGluon training complete, total runtime = 498.04s ... Best model: "Weigh
tedEnsemble_L2"
TabularPredictor saved. To load, use: predictor = TabularPredictor.load("Au
togluonModels/ag-20230531_055738/")
```

In [72]:
```
predictor.leaderboard(test_data, silent=True)
```

Out[72]:

	model	score_test	score_val	pred_time_test	pred_time_val	fit_time
0	CatBoost	0.7246	0.7248	0.035197	0.031973	69.439812
1	WeightedEnsemble_L2	0.7213	0.7280	1.605099	0.764912	414.350275
2	XGBoost	0.7199	0.7236	0.153429	0.061632	11.222906
3	LightGBMLarge	0.7184	0.7216	0.388713	0.139887	10.815943
4	LightGBM	0.7056	0.7112	0.252381	0.181767	7.727931
5	LightGBMXT	0.7052	0.6952	0.332557	0.143820	8.226042
6	NeuralNetFastAI	0.6783	0.6816	0.333562	0.139806	114.154014
7	NeuralNetTorch	0.6631	0.6688	0.096062	0.061192	189.655741
8	RandomForestEntr	0.6421	0.6360	1.669120	0.264285	19.832674
9	RandomForestGini	0.6385	0.6260	1.798824	0.381200	18.724065
10	ExtraTreesGini	0.6175	0.6108	1.385740	0.252083	16.889799
11	ExtraTreesEntr	0.6137	0.6112	2.099012	0.404819	17.104045
12	KNeighborsUnif	0.5851	0.5776	1.773412	0.721792	0.062398
13	KNeighborsDist	0.5725	0.5768	1.245847	0.416108	0.060322

In [75]:
```
predictor.evaluate(train_data)
```

```
Evaluation: accuracy on test data: 0.77665
Evaluations on test data:
{
    "accuracy": 0.77665,
    "balanced_accuracy": 0.7726952831144299,
    "mcc": 0.5521129735078575,
    "roc_auc": 0.8620285633024508,
    "f1": 0.7992539996404817,
    "precision": 0.7625519873086652,
    "recall": 0.8396676266465228
}
```

Out[75]:
```
{'accuracy': 0.77665,
 'balanced_accuracy': 0.7726952831144299,
 'mcc': 0.5521129735078575,
 'roc_auc': 0.8620285633024508,
 'f1': 0.7992539996404817,
 'precision': 0.7625519873086652,
 'recall': 0.8396676266465228}
```

In [73]:
```
predictor.evaluate(test_data)
```

```
Evaluation: accuracy on test data: 0.7213
Evaluations on test data:
{
    "accuracy": 0.7213,
    "balanced_accuracy": 0.7162518781807836,
    "mcc": 0.4387417835651197,
    "roc_auc": 0.7930215948653209,
    "f1": 0.751935914552737,
    "precision": 0.7156895967468655,
    "recall": 0.7920495030939434
}
```

Out[73]:
```
{'accuracy': 0.7213,
 'balanced_accuracy': 0.7162518781807836,
 'mcc': 0.4387417835651197,
 'roc_auc': 0.7930215948653209,
 'f1': 0.751935914552737,
 'precision': 0.7156895967468655,
 'recall': 0.7920495030939434}
```

In [74]:
```
# Confusion matrix on testing data
auto.analyze(model=predictor, val_data=test_data, anlz_facets=[
    eda.model.AutoGluonModelEvaluator(),
], viz_facets=[
    viz.layouts.MarkdownSectionComponent(markdown=f'### Model Prediction for
    viz.model.ConfusionMatrix(fig_args=dict(figsize=(3,3)), annot_kws={"size
])
```

Model Prediction for userLikedTheMovie