Prediction of Movie type base of given dataset

# Abstract:

## Sujeet Maurya (RKM030A11) Computer Science Department Lovely Professional University

Jalandhar , Punjab.

In this project, we apply KNNClassifier technique and machine learning algorithms using Python software to predict the type of movie based on several attributes. In order of doing this, we will develop a methodology on the basis of historical data to reduce certain level of uncertainty concerned to movie's type outcome. Some of the criteria in calculating movie success included budget, actors, director, producer, story writer, movie release day, competing movie releases at the same time, music, release location and target audience. Since, movie making involves huge investment thus movie prediction plays a vital role in the movie industry. This model helps movie makers to modify the criteria of blockbusters. It also helps movie watchers to determine a blockbuster before purchasing a ticket. Each attribute has some criteria and on the basis of that weightage has been given and then prediction is made based on that. Here, we also analyse key factors for movie profitability. This project also show the power of predictive and prescriptive data analytics for information systems to aid movie business decisions. This model also helps to find out the review of the new movie.

Keywords: Machine Learning, Movie, Python Software

# Introduction:

Movies are the most convenient way to entertain peoples. However only few movies get higher success and are ranked high. Many movies are produces by the movie industry in a year. A movie revenue depends on various components such as cast acting in a movie, budget for the making of the movie, film critics review, rating for the movie, release year of the movie, etc. Because of these multiple components there is no formula that helps us to provide analysis for predicting how much revenue a particular movie will be generating. However by analyzing the revenues generated by previous movies, a model can be built which can help us predict the expected revenue for a particular movie. As we know in today’s world the movie is one of the biggest source of entertainment and also for business purposes. . To expend this business further we need the technology through which we can predict the success rate of the movie. If we were able to predict the movie success rate in the correct manner then it will be easy for the businessman to get higher profit from it and also if the prediction shows the success rate is low of certain movie then it helps those businessmen to improve the content of the movie so that they can get higher revenue from it. Success rate of movies, models and mechanisms can be used to predict the success of a movie. It will help the business significantly. Stakeholders such as actors, producers, directors etc. can use these predictions to make more informed decisions. They can make the decision before the movie release. This proposed work aims to develop a model based upon the data mining techniques that may help in predicting the success of a movie in advance thereby reducing certain level of uncertainty. The excellent way to find detailed information about almost every film ever made is through IMDb. Vast amount of data, which contains much valuable information about general trends in films. Data mining techniques enable us to uncover information which will both confirm or disprove common assumptions about movies, and also allow us to predict the success of a future film given select information about the film before its release. So here we are developing the software for data analytics through which we can predict the success rate of the movie which high accuracy. Here we are using the Python-software to predicting the movie type into which first we have downloaded the data set from kaggle.com and after that we are generating the training and test data set. In a dataset, a training set is implemented to build up a model, while a test (or validation) set is to validate the model built. The main attributes selected for building model are critics score, imdb\_rating, imdb\_num\_votes, audience\_score. Data points in the training set are excluded from the test (validation) set. Usually, a dataset is divided into a training set, a validation set (some people use 'test set' instead) in each iteration, or divided into a training set, a validation set and a test set in each iteration. These training and test dataset is used to build model for selected set of attributes. On the basis of the generated model prediction has been done and result has been generated. Through the acquired result we can easily conclude that the movie type weather action, or romance. The outcome of this research is movie type it provides tools and techniques to transform the database data into a format suitable for data mining, and provides a selection of information mined from this refined data.

# Dataset Description:-

The dataset requirement for our project is fulfilled through kaggle repository. From here we downloaded the dataset and used as an input. Kaggle.com is a website that provides dataset for free for its users. Thus we got dataset for free of cost. This dataset consists of 8 rows and 4 columns. The dataset we get is preprocessed so we need not to pre-process it. Our first task for this assignment is to choose which variables to include in our model. It would be easier to start with eliminating variables that will obviously not be of use for our model. Uniform Resource Locators or commonly known as URLs provide an easy way to find more information for each movie but will not provide information whether a movie is Romance or Action. Kisses and kicks in the movie in a key ingredient for type for a movie. Most movies are of Romance or Action. The title of a movie is usually what a moviegoer remembers when a movie is popular but it is not what makes a movie popular. However, this not the case when it comes to actors, actresses or directors. Movie goers turn into devoted fans when a certain actor, actress or director captures their imagination and becomes a key determine whether subsequent movies from the same person is a must see. Let us now focus our attention to choosing our response variable. Since, we are working on development of model and prediction, the requirement of our data is numerical. Thus, the attribute selected here must be an attribute containing numerical values.

Table 1:- Below is a list of the variables that measures a movie’s type.

Movie name, kicks, kisses, movie type

California Man, 3, 104, Romance

He's not really into Dudes,2, 100, Romance

Beautiful Woman, 1, 81, Romance

Kevin Long blade, 101, 10, Action

Robo Slayer, 99, 5, Action

Amped II, 98, 2, Action

?, 18, 90, Unknown

# Output Description:-

self.training features : [[ 3. 104.]

[ 2. 100.]

[ 1. 81.]

[101. 10.]

[ 99. 5.]

[ 98. 2.]]

self.training labels : ['Romance', 'Romance', 'Romance', 'Action', 'Action', 'Action']

self.test features : [18. 90.]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

classifytestdata: test\_data = None

classify test data: self.test\_features = [18. 90.]

classifytestdata says self.testfeatures : [18. 90.]

self.training\_features : [[ 3. 104.]

[ 2. 100.]

[ 1. 81.]

[101. 10.]

[ 99. 5.]

[ 98. 2.]]

self.training\_labels : ['Romance', 'Romance', 'Romance', 'Action', 'Action', 'Action']

feature Vector Size : 6

after Tile of test data :

[[18. 90.]

[18. 90.]

[18. 90.]

[18. 90.]

[18. 90.]

[18. 90.]]

diffMat : [[-15. 14.]

[-16. 10.]

[-17. -9.]

[ 83. -80.]

[ 81. -85.]

[ 80. -88.]]

sqDifMat : [[ 225. 196.]

[ 256. 100.]

[ 289. 81.]

[6889. 6400.]

[6561. 7225.]

[6400. 7744.]]

RowwiseSum SqDistances : [ 421. 356. 370. 13289. 13786. 14144.]

distances(Sq.Root of SqDistances :: [ 20.51828453 18.86796226 19.23538406 115.27792503 117.41379817

118.92854998]

sortedDistanceIndices ::: [1 2 0 3 4 5]

self.training\_labels : ['Romance', 'Romance', 'Romance', 'Action', 'Action', 'Action']

sortedDistanceIndices[i] : 1

voteILabel : Romance

sortedDistanceIndices[i] : 2

voteILabel : Romance

sortedDistanceIndices[i] : 0

voteILabel : Romance

sortedDistanceIndices[i] : 3

voteILabel : Action

sortedDistanceIndices[i] : 4

voteILabel : Action

classCount = {'voteILabel': 1}

sortedclassCount : [('voteILabel', 1)]

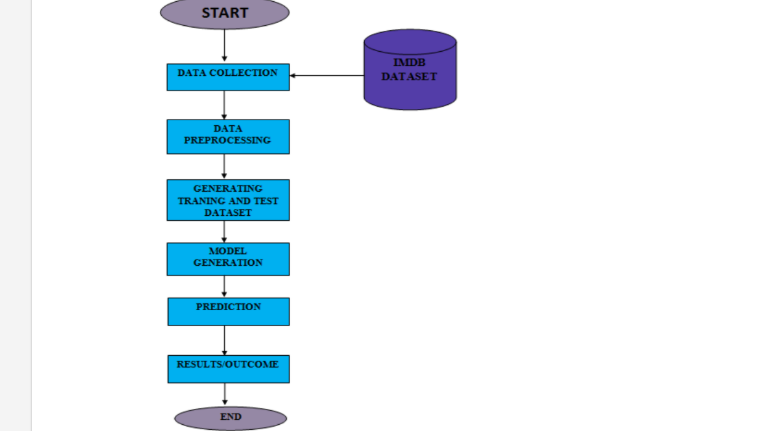
sortedclassCount[0] : ('voteILabel', 1)

sortedclassCount[0][0] : voteILabel

predictMovieType ClassofTestdata voteILabel

Most Likely movie,[18. 90.] is of type

# Flow Diagram of implementation:-



# Proposed Methodology:-

The proposed methodology deals with different stages of the project which consists of data collection, data preprocessing, generating training and testing, model generation, prediction and outcomes. These all methods prevent us from getting any irrelevant data which further keeps our outcomes more relevant and accurate for the prediction. Here we collected dataset which consist of 8 attributes and 4 tuples. Further steps are explained below:

# Data collection:-

The crude IMDb dataset is organized so that the greater part of its properties and information is sorted out and put away independently in compacted plain content documents. For example, the majority of the approximately 8 movies picture appraisals from the database are put away in the compacted content document evaluations. Rundown which incorporates literary information about the information just as a table of film rank, the number of votes and film titles. In this way, some kind of cleaning, mix and preprocessing is probably going to be required so as to utilize the information with the end goal of information mining through supervised machine learning strategies. The information was gathered utilizing IMDB which contains the IMDB motion picture dataset of in excess of 8 films in the dataset.

# Data Processing:-

In this stage dataset is prepared for applying data mining technique. Before applying data mining technique, pre-processing methods like cleaning, variable transformation and data partitioning and other techniques for attribute selection must be applied. After pre-processing we have attributes or variables for each movie. Each test file will contain best attributes and rebalanced. As the data is taken in the raw format from IMDb it is first required to be pre-processed. To overcome missing value scenario central tendency method is used both mean and median and later the duplicate items are removed. Pre-processing is the crucial phase for the project as it mainly focuses on the working of the algorithm. As the data is now pre-processed next comes data integration and transformation in which the alpha numerical data need to be converted to the numerical data as it is required for regression model. The correlation between the features is identified using the greedy backwards method.

# Conclusion:-

In this project we are just trying to determine if there is any association between different attributes present in our dataset. Here, our main aim is to find association between numeric type attributes that is used s a scoring systems and how we can use this association for prediction. As a result we found that critics score is strongly positive relationship between critics score and audience score. And we can also conclude that critics score are best predictor of audience scores. Thus, we can predicted our movies success on the basis of critics score. In future, we can add many attributes as our predictors and build model for that attributes to perform prediction. Here, we can assume that if we have movie gross score and movie net profit along with movie manufacturing cost, then we can build a more strong model for movie success prediction. In future, we can apply other machine learning algorithms for movie success prediction.

# PROJECT PYTHON CODE:-

# import numpy as np

# from operator import itemgetter

# import csv,os

# class KNNClassifier(object):

# def \_\_init\_\_(self):#constructor

# self.training\_features=None #movie kicks kisses {[3,4,5,]}

# self.training\_labels=None #movie gener[Romance,Comedy],label=output of trining,acutual

# self.test\_features=None #kicks=18 kisses =90 [18,90]

# #Build Meaningful result

# self.elegantResult="Most Likely {0},{1} is of type"#elegantresult is the unknowdata

# def loadTrainingFromFile(self,file\_path):

# if file\_path is not None and os.path.exists(file\_path):#the file name is there in os

# tr\_features=[]

# self.training\_labels=[]

# with open(file\_path,'r') as training\_data\_file:

# reader=csv.DictReader(training\_data\_file)#read all line other the 1st line

# for row in reader:#divide all the data excpect 1st row

# if row['moviename']!='?':

# tr\_features.append([float(row['kicks']),float(row['kisses'])])

# #input feacture of 1st input

# self.training\_labels.append(row['movietype'])

# else:

# self.test\_features=np.array([float(row['kicks']),float(row['kisses'])])

# if len(tr\_features)>0:

# self.training\_features=np.array(tr\_features)

# print("self.training features : ",self.training\_features)

# print("self.training labels :",self.training\_labels)

# print("self.test features :",self.test\_features)

# #[1,1,1] #[10,90] #k=5

# def classifyTestData(self,test\_data=None,k=0):#here the test\_data is local varible,none is used her for internal

# print("classifytestdata: test\_data =",test\_data)#

# if test\_data is not None:

# self.test\_features=np.array(test\_data,dtype=float)

# print("classify test data: self.test\_features =",self.test\_features)

# #ensure we have training data, training labels and testdata and ....

# if self.test\_features is not None and self.training\_features is not None and self.training\_labels is not None:

# print("classifytestdata says self.testfeatures :",self.test\_features)#[18,90]

# print("self.training\_features :",self.training\_features)#[[3,104],...

# print("self.training\_labels :",self.training\_labels)#['romance'....]

# featureVectorSize=self.training\_features.shape[0]# answer will be 6,number of rows

# print("feature Vector Size :",featureVectorSize)#array of column(features)

# tileofTestData=np.tile(self.test\_features,(featureVectorSize,1))#make 6 row in 1 value ,repeact 6 times

# print("after Tile of test data :\n",tileofTestData)

# diffMat=self.training\_features-tileofTestData

# print("diffMat :",diffMat)#diffrence matrix

# sqDifMat=diffMat\*\*2

# print("sqDifMat :",sqDifMat) #6x2

# sqDistances=sqDifMat.sum(axis=1) #6x1 pruduce the row

# print("RowwiseSum SqDistances :",sqDistances)

# distances=sqDistances\*\*0.5

# print("distances(Sq.Root of SqDistances :: ",distances)

# sortedDistanceIndices=distances.argsort()

# print("sortedDistanceIndices :::",sortedDistanceIndices)

# print("self.training\_labels : ",self.training\_labels)

# classCount={}

# for i in range(k): #k=5 == 0,1,2,3,4

# print("sortedDistanceIndices[i] :",sortedDistanceIndices[i])

# voteILabel=self.training\_labels[sortedDistanceIndices[i]]

# print("voteILabel :",voteILabel)

# classCount['voteILabel']=classCount.get(voteILabel,0)+1

# #classCount={Action:2 , Romance:3}

# print("classCount = ",classCount)

# sortedclassCount=sorted(classCount.items(),key=itemgetter,reverse=True)

# # sortedClassCount = {"Roamnce",3),(" Action" ,2}

# print("sortedclassCount :",sortedclassCount)

# print("sortedclassCount[0] :",sortedclassCount[0])

# print("sortedclassCount[0][0] :",sortedclassCount[0][0])

# return sortedclassCount[0][0]

# else:

# return "Can't Determine result for applying test\_data"

# def predictMovieType():

# instance=KNNClassifier()

# instance.loadTrainingFromFile("LgR\_Movies\_kNN\_classifier.csv")

# print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

# #my\_test\_data=[50,50] #can be supplied to instance classifyTestData()

# #classOfTest\_data=instance.classifyTestData(test\_data=my\_test\_data,k=5)

# classOfTest\_data=instance.classifyTestData(test\_data=None,k=5)

# print("predictMovieType ClassofTestdata",classOfTest\_data)

# return instance.elegantResult.format(('movie'),str(instance.test\_features),classOfTest\_data)

# if \_\_name\_\_=="\_\_main\_\_" :

# print(predictMovieType())

REFERENCES

1] Krushikanth R. Apala ; Merin Jose ; Supreme Motnam ; C.-C. Chan ; Kathy J. Liszka ; Federico de Gregorio” Prediction of Movies Box Office Performance Using Social Media”, 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining(ASONAM 2013), PP 1209- 1214.

[2] Javaria Ahmad ; Prakash Duraisamy ; Amr Yousef ; Bill Buckles” Movie Success Prediction Using Data Mining”, 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), PP 1- 4.

[3] Susmita S. Magdum ; J. V. Megha” Mining Online Reviews and Tweets for Predicting Sales Performance and Success of Movies”, 2017 International Conference on Intelligent Computing and Control Systems (ICICCS)PP 334-339.

[4] Md Shamsur Rahim ; A Z M Ehtesham Chowdhury ; Md. Asiful Islam ; Mir Riyanul Islam” Mining Trailers Data from YouTube for Predicting Gross Income of Movies”,2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC),PP 551-554.