BOX OFFICE PREDICTION

USING MACHINE LEARNING ALGORITHM

A Course Project report submitted

in partial fulfillment of requirement for the award of degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

BANDI REETHIKA (2103A51447)

SANDI SRIJA (2103A51536)

Under the guidance of

Mr. D. Ramesh

Assistant Professor, Department of CSE.

Department of Computer Science and Artificial Intelligence

CERTIFICATE

This is to certify that project entitled **"BOX OFFICE PREDICTION USING MACHINE LEARNING ALGORITHMS"** is the Bonafede work carried out by **BANDI REETHIKA AND SANDI SRIJA.**

As a course project of the partial fulfillment to award the degree **BACHELOROF TECHNOLOGY in ARTIFICIAL INTELLIGENCE AND MACHINELEARNING** during the academic year **2022-2023** under the guidance and supervision.

Mr. D.Ramesh

Asst.Professor,

SR University,

Ananthasagar, Warangal.Ananthasagar, Warangal

Dr. M.Sheshikala

Asst.Prof .&HOD(CSE) ,

SR University,

ACKNOWLEDGEMENT

We express our thanks to Course co-coordinator **Mr. D. Ramesh, Asst. Prof** for guiding us from the beginning through the end of the course Project . We express our gratitude to Head of the department CS&AI, **Dr. M. Sheshikala** , **AssociateProfessor** for encouragement , support and insightful suggestions,. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Dean , School and Artificial Intelligence , **DrC.V.Guru Rao** , for his continuous support and guidance to complete this project in the institute.

ABSTRACT

Box office in movies refers to the total amount of revenue generated by a film during its theatrical release. The box office is the important measure of a film's commercial success and is often used a benchmark for evaluation of a movie's popularity and profitability. A Movie box office collections dataset is mainly focuses on each and every single movie to predict the financial success of a movie.

The dataset contains various features such as the movie name, year, score, IMDB rating, votes etc., which can be used to train machine learning models to predict the box office revenue for a new movie. By analyzing the historical box office data, we can able to identify movie performance, which can then be used to make predictions about the future financial success of a movie. Movie box office collections dataset can be useful in various fields, including the movie industry, market research, and investment analysis. For example, movie studios can use machine learning models to optimize their production strategies and make data-driven decisions about which movies to produce and how much to spend on them. Market researchers can also use the dataset to analyze trends and consumer behavior related to movie-watching, which can help businesses make informed decisions about marketing and promotion.

TABLE OF CONTENTS

CHAPTER-1

INTRODUCTION

Box office prediction is the process of forecasting the potential earnings of a movie before it is released in theaters. The accuracy of these predictions can have a significant impact on the movie industry, as they inform investment decisions, marketing strategies, and release schedules.

Box office prediction has become increasingly important in recent years as the movie industry has become more competitive and costly. With millions of dollars at stake, accurate predictions can help studios optimize their investments and maximize their returns. On the other hand, inaccurate predictions can lead to financial losses, reputational damage, and even production shutdowns.

To predict box office earnings, analysts use a variety of methods and data sources, including historical box office data. Social media trends, audience demographics and critical reviews, machine learning algorithms and artificial intelligence are also being used to improve prediction accuracy by analyzing vast amounts of data and identifying patterns and correlations.

Despite advances in technology and data analysis, predicting box office earnings remains an inherently challenging task due to the unpredictable nature of audience preferences and behaviors. Nevertheless, accurate predictions can provide valuable insights into audience preferences and trends, and help studios make more informed decisions.

1.1-PROBLEM STATEMENT:

Problem Statement:

The goal of this project is to develop a machine learning model that can predict the success rate of the film whether it reached the aim or not. And also to develop accurate and reliable methods to forecast the potential earnings of the movie before releasing in theatres.

The main motive is to prove the prediction accuracy using the different classification models and compare which model performs better regarding the problem.

1.2-EXISTING SYSTEM:

We have collected the data set from a website called Kaggle.com. Our data set contains imbd ratings and different movie names along with the actors and technicians.It also consists of years when movies were released and money collected. We removed all the strings involved in the data set for further progression.

1.3-PROPOSED SYSTEM:

With the assist of dataset obtained we create 5 different machine learning algorithms and they are **KNN, SVM, LOGISTIC REGRESSION, DECISION TREE, RANDOM FOREST.** We examined the outcomes of accuracy and found that which models performs the best**.**

1.4-OBJECTIVES:

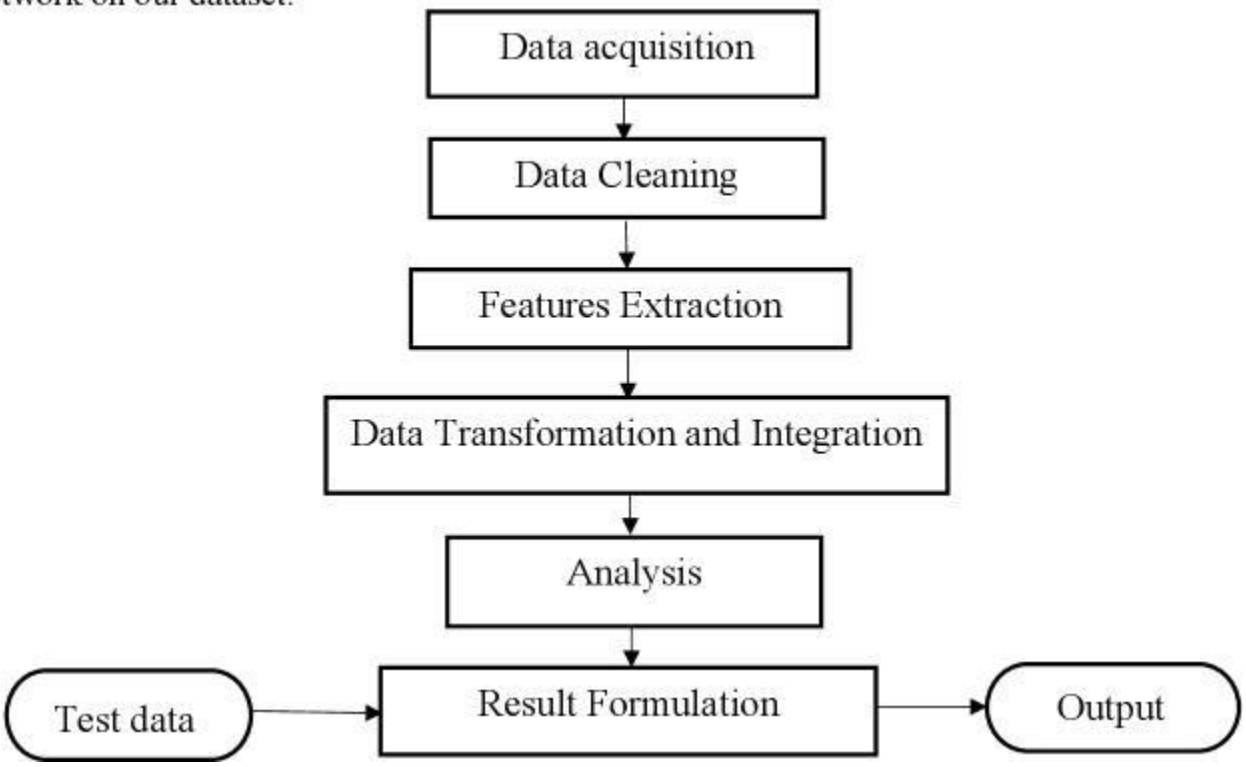
The purpose of the box office prediction is to find a machine learning model to identify the box office success of a movie using the given dataset with accuracy. The model should be able to determine the correctly as the movie's box office success. The dataset is divided into two training and testing datasets.

1.5-ARCHITECTURE:

This is a Supervised learning approach. As the data have categorical values we used classification based machine learning algorithms to predict the condition of movie, which might be a hit or flop.

We collected the data set firstly and after collecting the data pre-processing is done.Then the data set is get divided into 2 sets(i.e training and testing). Using classification based machine learning models we trained the model after finding the accuracy on the training data set we found the accuracy on the testing model. Based on those conclusions we evaluated the model. And this is the architecture followed by us.

Network on our dataset.



CHAPTER-2

LITERATURE SURVEY

Makers and film producers need to predict the profit they are gonna get so that they can invest their money and time accordingly.

Previous studies that have been conducted show that it is possible to predict the success of movie using attributes such as budget, rating, actors and director. Thus the goal of this thesis is to further examine the possibility of using a greater data set with features previously not used with machine learning.

Movies generate income from many revenue streams, movie rentals and purchases, theatre ticket/box office sales, merchandising and home video are a few. To be able to compare a movies, one revenue stream was selected, which is the income generated by theatre ticket sales, or box office sales. The figures used in this study are used from box-office mojo, the leading online box-office reporting service, owned and operated by IMDb.

2.1-Analysis of the Survey:

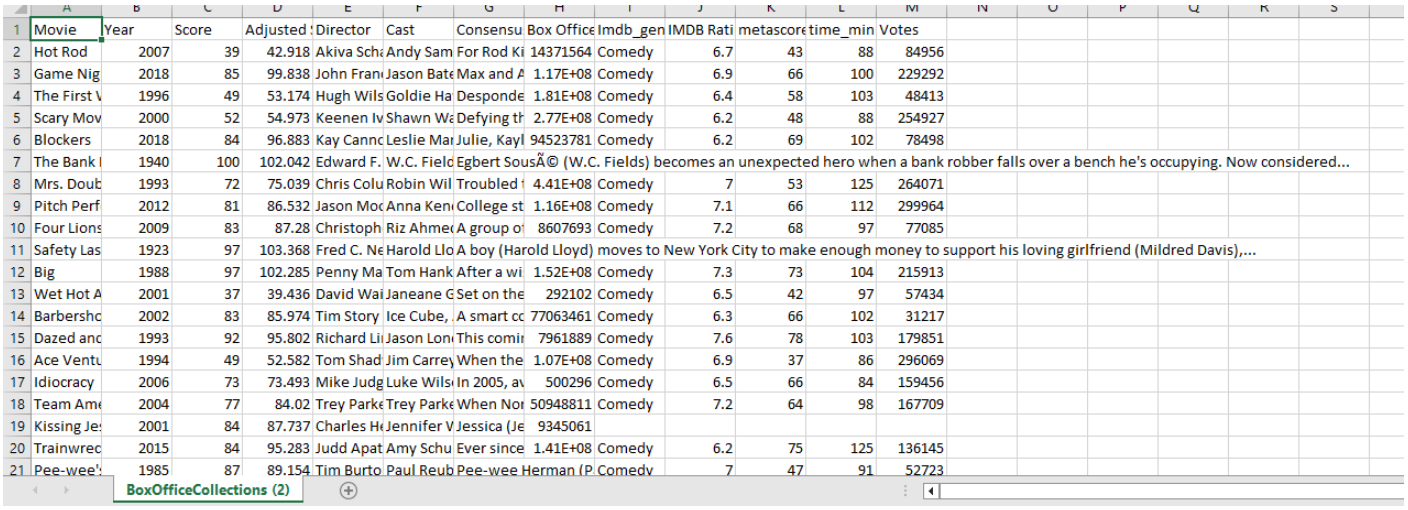
The study can be used as a proof of concept for applications in other areas, and should highlight some of the challenges one needs to overcome to successfully create a prediction model. This idea could in theory be extended to predict credit ratings, the stock market or housing market. The only requirement being a vast and reliable data source. When combining the questions mentioned above to form a problem statement, formulating good as a measurement of a movies rating and sales, the following problem statement was produced.

CHAPTER-3

DATA PRE-PROCESSING

Box office prediction dataset consists of nearly 2000 rows and 13 columns.
The names of columns are listed below:
MOVIE, YEAR,SCORE,ADJUSTED YEAR,DIRECTOR,CAST,CONSENSUS, BOX OFFICE COLLECTION, IMDB_GENRE, RATING,METASCORE,TIME_MINUTE,VOTES .

3.1 DATA DESCRIPTION



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	Movie	Year	Score	Adjusted '	Director	Cast	Consensu	Box Office	Imdb_gen	IMDB Rati	metascore	time_min	Votes						
2	Hot Rod	2007	39	42.918	Akiva Scha	Andy Sam	For Rod Ki	14371564	Comedy	6.7	43	88	84956						
3	Game Nig	2018	85	99.838	John Franı	Jason Bate	Max and A	1.17E+08	Comedy	6.9	66	100	229292						
4	The First V	1996	49	53.174	Hugh Wils	Goldie Ha	Desponde	1.81E+08	Comedy	6.4	58	103	48413						
5	Scary Mov	2000	52	54.973	Keenen Iv	Shawn Wa	Defying th	2.77E+08	Comedy	6.2	48	88	254927						
6	Blockers	2018	84	96.883	Kay Cannc	Leslie Mar	Julie, Kayl	94523781	Comedy	6.2	69	102	78498						
7	The Bank I	1940	100	102.042	Edward F.	W.C. Field	Egbert SousÃ© (W.C. Fields) becomes an unexpected hero when a bank robber falls over a bench he's occupying. Now considered...												
8	Mrs. Doub	1993	72	75.039	Chris Colu	Robin Wil	Troubled '	4.41E+08	Comedy	7	53	125	264071						
9	Pitch Perf	2012	81	86.532	Jason Moc	Anna Ken	College st	1.16E+08	Comedy	7.1	66	112	299964						
10	Four Lions	2009	83	87.28	Christoph	Riz Ahmec	A group of	8607693	Comedy	7.2	68	97	77085						
11	Safety Las	1923	97	103.368	Fred C. Ne	Harold Llo	A boy (Harold Lloyd) moves to New York City to make enough money to support his loving girlfriend (Mildred Davis),...												
12	Big	1988	97	102.285	Penny Ma	Tom Hank	After a wi	1.52E+08	Comedy	7.3	73	104	215913						
13	Wet Hot A	2001	37	39.436	David Wai	Janeane G	Set on the	292102	Comedy	6.5	42	97	57434						
14	Barbersho	2002	83	85.974	Tim Story	Ice Cube,	A smart co	77063461	Comedy	6.3	66	102	31217						
15	Dazed and	1993	92	95.802	Richard Li	Jason Lon	This comir	7961889	Comedy	7.6	78	103	179851						
16	Ace Ventu	1994	49	52.582	Tom Shad	Jim Carre	When the	1.07E+08	Comedy	6.9	37	86	296069						
17	Idiocracy	2006	73	73.493	Mike Judg	Luke Wils	In 2005, av	500296	Comedy	6.5	66	84	159456						
18	Team Ame	2004	77	84.02	Trey Parke	Trey Parke	When Nor	50948811	Comedy	7.2	64	98	167709						
19	Kissing Je	2001	84	87.737	Charles He	Jennifer V	Jessica (Je	9345061											
20	Trainwrec	2015	84	95.283	Judd Apat	Amy Schu	Ever since	1.41E+08	Comedy	6.2	75	125	136145						
21	Pee-wee'	1985	87	89.154	Tim Burto	Paul Reub	Pee-wee Herman (P		Comedy	7	47	91	52723						

BoxOfficeCollections (2)

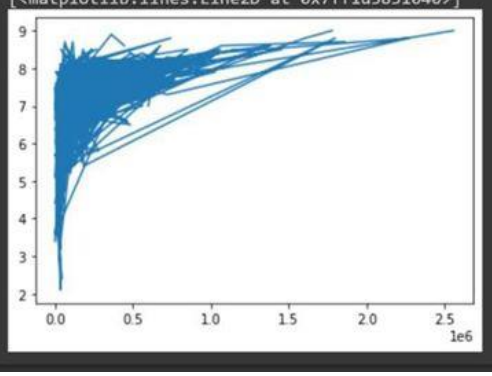
The goal of data preprocessing is to ensure that the data is accurate, complete, and consistent before any analysis is performed.

In our dataset we removed a null value and we added zeroes in to it. This is the preprocessing we used in our data set.

3.2 DATA VISUALIZATION

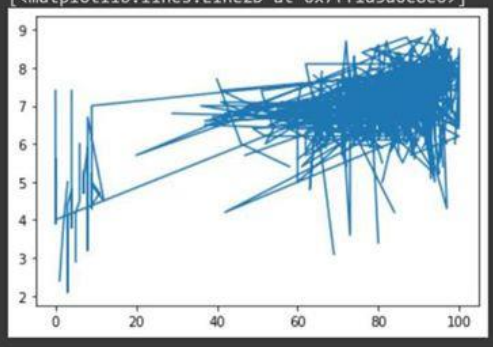
```
x=p['Votes']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d3831040>]
```



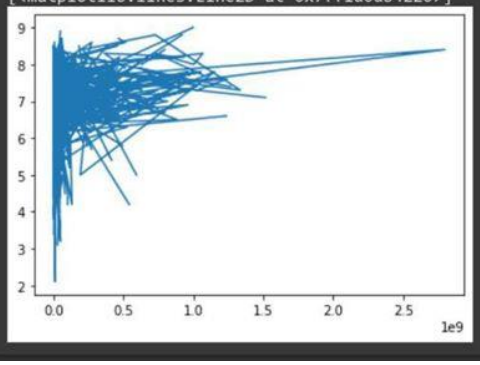

```
x=p['Score']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d3d0c8e0>]
```



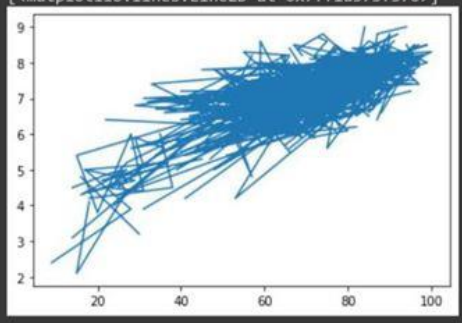

```
x=p['Box Office Collection']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d0db4220>]
```



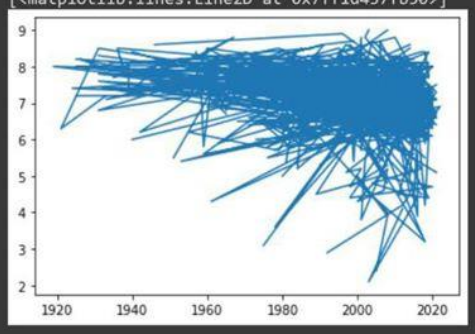

```
x=p['metascore']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d373f370>]
```



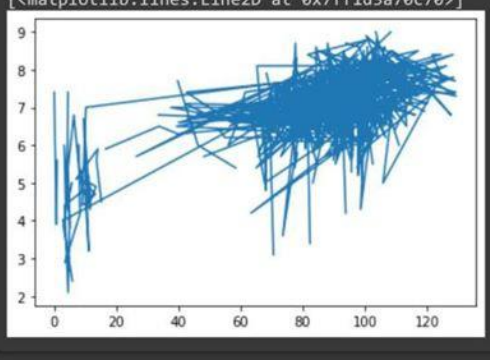
```
x=p['Year']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d457fb50>]
```



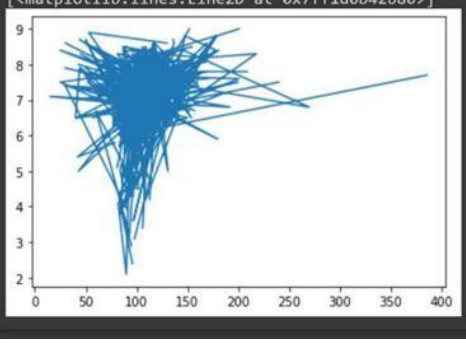

```
x=p['Adjusted Score']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d3a70c70>]
```



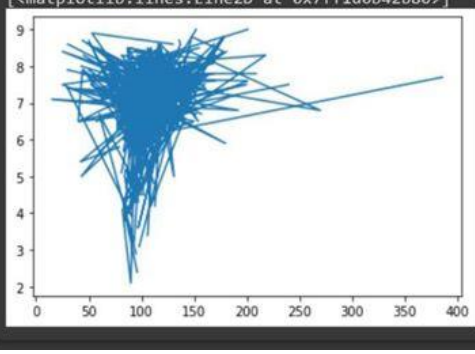

```
x=p['time_minute']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d0b42b80>]
```




```
x=p['time_minute']
y=p['IMDB Rating']
plt.plot(x,y)
```

```
[<matplotlib.lines.Line2D at 0x7ff1d0b42b80>]
```



CHAPTER 4

METHODOLOGY

4.1. PROCEDURE TO SLOVE THE GIVEN PROBLEM:

4.1.1 K-Nearest Neighbors:

The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of K number of neighbors

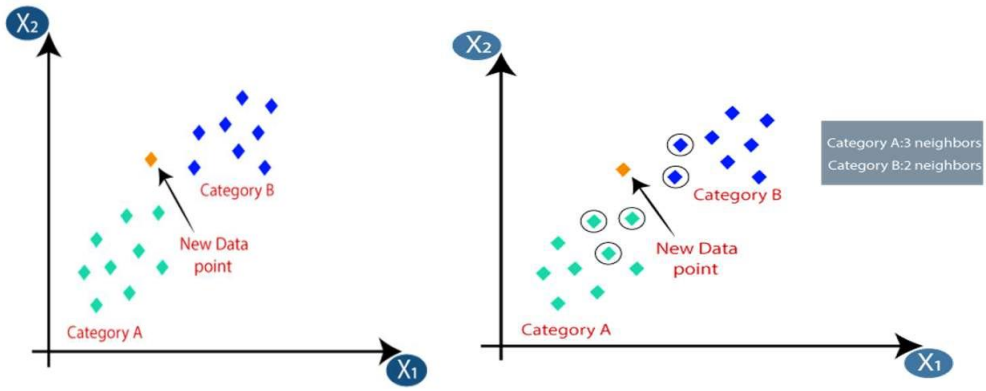
Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:



Firstly, we will choose the number of neighbors ,so we will choose the k=5.Next, we will calculate the **Euclidean distance**between the data points. The Euclidean distance is the distance between points, which we have already studied in geometry. By calculating the Euclidean distance, we get the nearest.Neighbors,as three nearest neighbors in category A and two Nearest neighbors in category B.

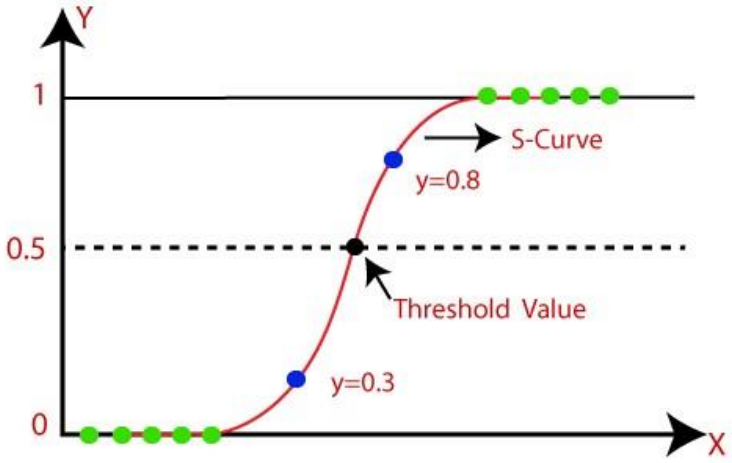
As we can see the 3 nearest neighbors are from category A,hence this new data point must belong to category A.

4.1.2. Logistic Regression:

Logistic regression uses the concept of predictive modeling as regression; therefore, it is called logistic regression, but is used to classify samples; Therefore, it falls under the classification algorithm.

Logistic regression is used when the dependent variable is binary such as click on a given advertisement link or not, spam detection, Diabetes prediction, the customer will purchase or not, an employee will leave the company or not.

Logistic regression uses Maximum Likelihood Estimation (MLE) approach i.e., it determines the parameters (mean and variance) that are maximizing the likelihood to produce the desired output.



Logistic Regression uses a sigmoid or logit function which will squash the best fit straight line that will map any values including the exceeding values from 0 to 1 range. So, it forms an "S" shaped curve.

Sigmoid function removes the effect of outlier and makes the output between 0 and 1.

The logistic function is of the form:

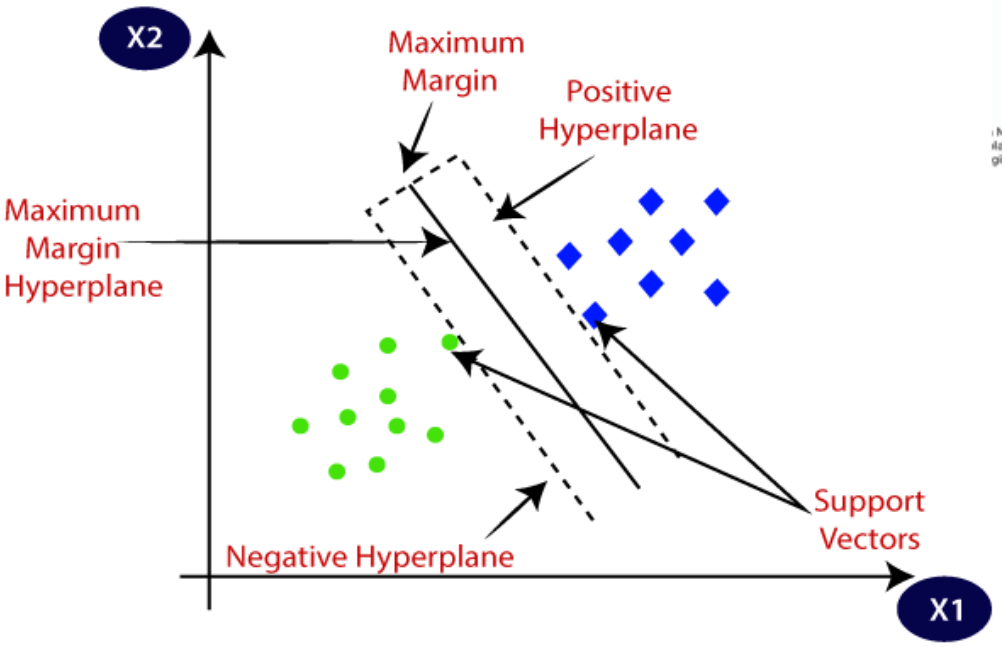
P(X)=1/1+e^-z

where μ is a location parameter

s is a scale parameter.

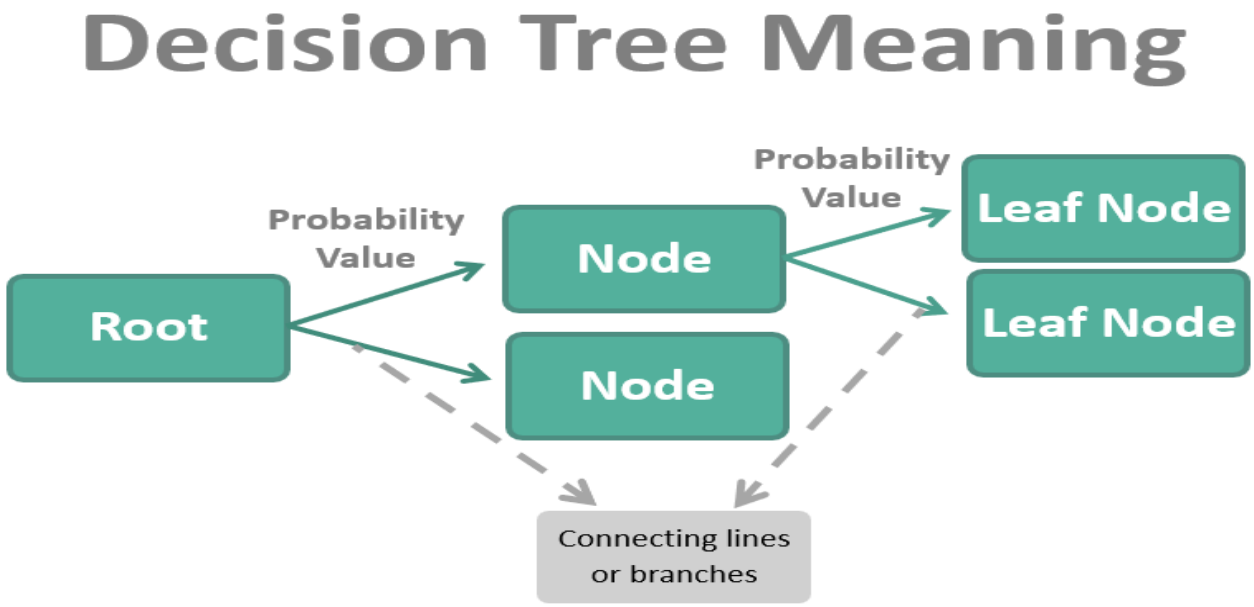
4.1.3 SVM(Support Vector Machine):

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. In the SVM method,we plot each data item as a point in n-dimensional space(where n is no.of features you have).
- We perform classification by finding the hyperplane that differentiates the two classes very well.
- We have three hyper planes(A,B and C). Now,identify the right hyper plane to classify star and circle.
- We want our data points to be as far away from the hyperplane as possible,while still being on the correct side of it.
- The distance between the hyperplane and the nearest data point from either set is known as margin.
- The goal is to choose a hyperplane with the greatest possible margin.
- There will never be any data point inside the margin.



4.1.4Decision Tree:

- It is a non parametric method used for supervised learning method used for both classification and regression.
- It uses tree representation to solve the problem.As deeper the tree goes, the more complex the decision rules and the fitter the model.
- Entropy and Information Gini are used to calculate root node among all the nodes.
- Hence,an optimal tree can be formed.
- From given data a tree can be formed and using entropy & information gini we can calculate accuracy.
- Tree is a hierarchical representation(pictorial representation).
- ENTROPY:Entropy is the measure of uncertainity of a random variable.The higher entropy results in more information.
- **Entropy**=summation(-p_i*log(p_i))
- INFORMATION GINI:Information gini is the measure of changes in the entropy.
- **Information Gini**=E(T)-E(T,X)

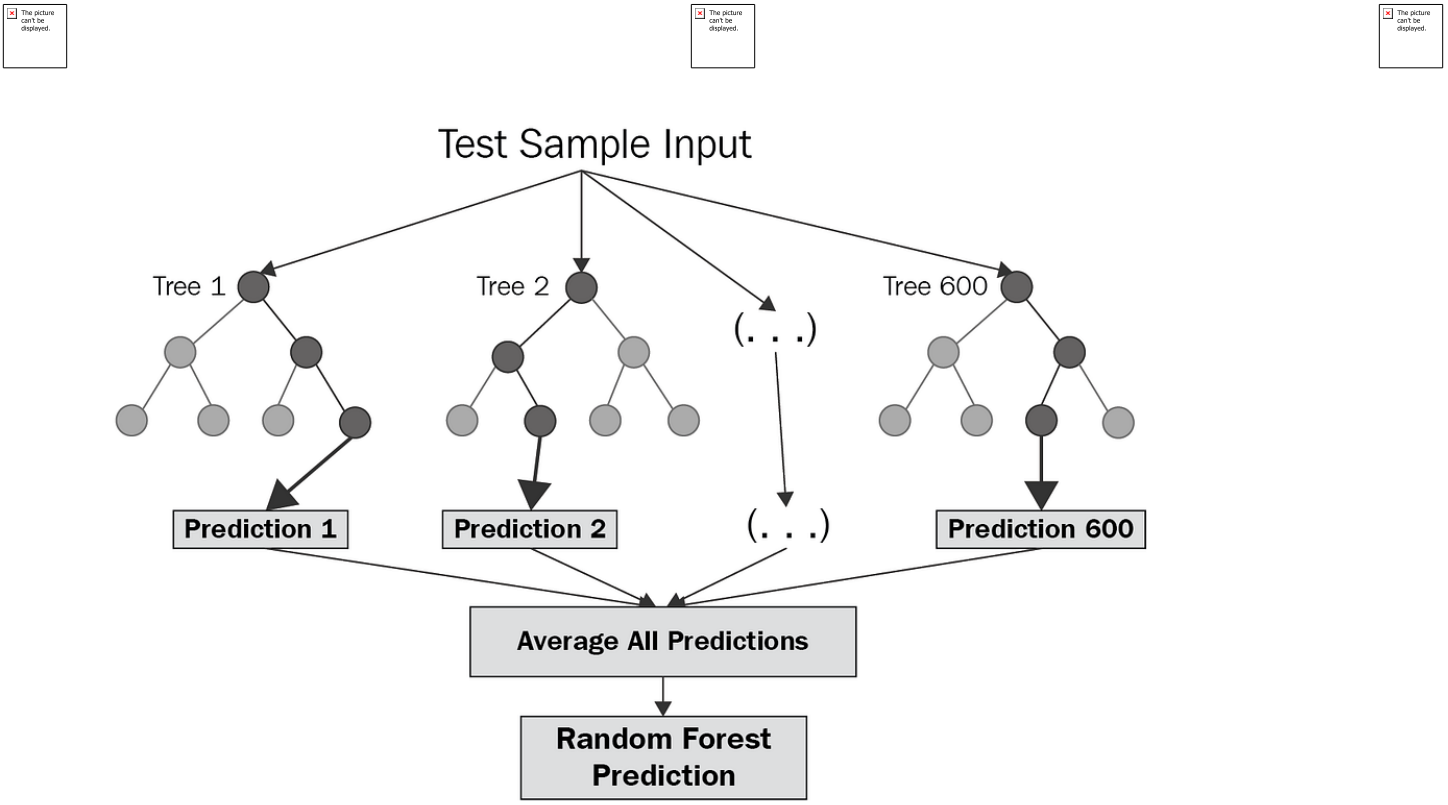


4.1.4Random Forest:

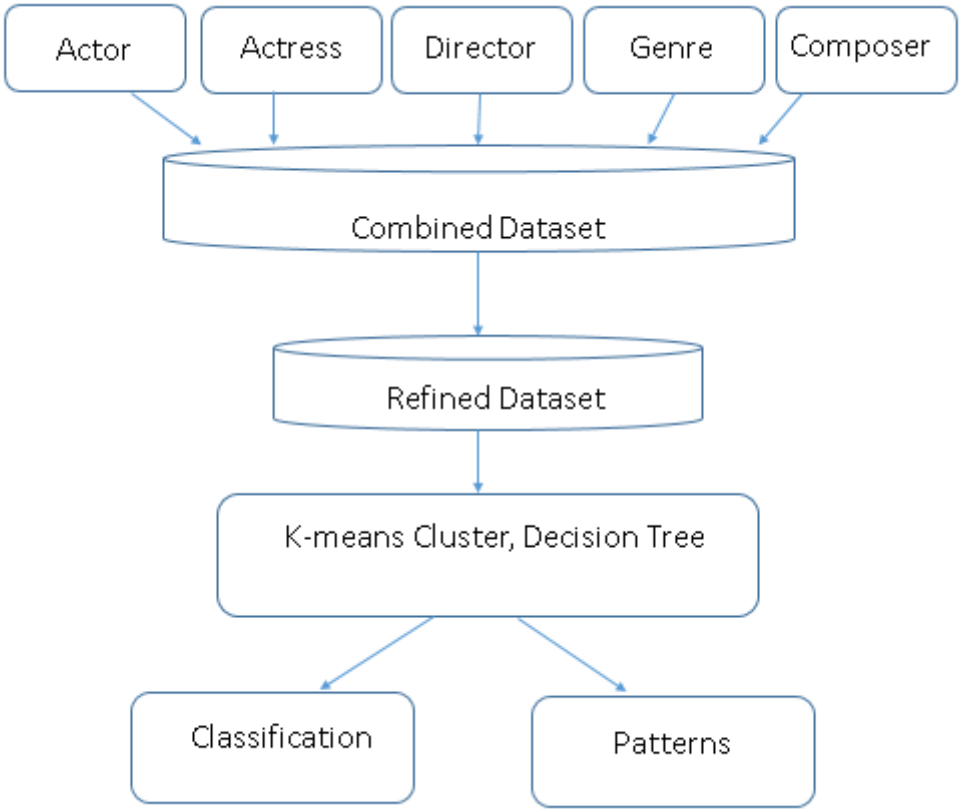
Random forest is a commonly-used machine learning algorithm trademarked by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems.

Random forest algorithms have three main hyperparameters, which need to be set before training. These include node size, the number of trees, and the number of features sampled. From there, the random forest classifier can be used to solve for regression or classification problems.

The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble is comprised of a data sample drawn from a training set with replacement, called the bootstrap sample. Of that training sample, one-third of it is set aside as test data, known as the out-of-bag (oob) sample, which we'll come back to later. Another instance of randomness is then injected through feature bagging, adding more diversity to the dataset and reducing the correlation among decision trees. Depending on the type of problem, the determination of the prediction will vary. For a regression task, the individual decision trees will be averaged.



MODEL ARCHITECTURE:



4.2-SOFTWARE DESCRIPTION:

Software requirements:

Operating systems: windows

Platform: google collab

Programming languages: python.

5.RESULTS

CODE

Dataset:

```
import pandas as pd
d=pd.read_csv('/content/box office prediction 777.csv')
print(d)
```

Output:

```
2     1996   49     53.174     181489203.0    0
3     2000   52     54.973     277200000.0    0
4     2018   84     96.883      94523781.0    0
...    ...  ...        ...             ...  ...
1007  1979  100    103.012        296334.0    3
1008  1985  100    105.082        985454.0    0
1009  1994  100    101.270        655454.0    3
1010  1935  100    107.903       3202000.0    0
1011  1953  100    104.525        678899.0    5

      IMDB Rating  metascore  time_minute   Votes
0             6.7         43           88   84956
1             6.9         66          100  229292
2             6.4         58          103   48413
3             6.2         48           88  254927
4             6.2         69          102   78498
...           ...        ...          ...     ...
1007          8.1         89          162  131448
1008          7.9         87          114   18557
1009          8.1        100           99  101519
1010          7.7         92          101   19036
1011          8.2         85          131   60070

[1012 rows x 9 columns]
```

Linear Regression:

```
from sklearn.linear_model import LinearRegression
```

```
lr=LinearRegression()
```

```
mm=lr.fit(x_train,y_train)
```

```
yp=mm.predict(x_test)
```

```
print(yp)
```

Output:

```
[[ 269887.58816803]
 [ 147890.49998941]
 [ 526237.34620081]
 [ 582742.54846975]
 [ 276758.69921503]
```

```
[ 256513.79355117]
[ 192535.40690198]
[ 196932.64287048]
[ 220477.73557114]
[ 164697.88854685]
[ 140431.25050915]
[ 340959.37109032]
[  55421.73554129]
[  -4854.28174242]
[ 382328.06013519]
[ 297488.08413763]
[ 222596.3476985 ]
[ 201780.11627052]
[ 579414.20365496]
[  33366.44343845]
[ 151617.01498717]
[ 115573.64968497]
[ -43510.41545084]
[ 135072.91590046]
[ 134250.28858732]
[ 352325.79516749]
[ 354852.24657836]
[ 326812.44810206]
[-177285.32839413]
[ 282821.50260669]
[ 216326.61208628]
[ 309061.91158992]
[ 208024.63258324]
[ 310754.18705578]
[ 316657.65215869]
[ 283598.96495413]
[ 214358.09601964]
[ 125897.13856262]
[ 272819.45896438]
[ 297437.56821313]
[ 189739.60364046]
[ 108546.78310002]
[ 321632.11249973]
[ 137529.87861601]
[ 182161.47216378]
[ 475225.18628346]
[ 273227.55110071]
[ 115516.26730323]
[ 158910.10679005]
[ 153007.66575858]
[ -33131.71483312]
[ 106646.0403798 ]
[  37646.6303767 ]
[ 249914.44064287]
[ 503037.20375668]
[ 254510.25526511]
[ 509789.1570332 ]
[ 208780.67917261]
[ 261382.22810003]
[ 171243.54797408]
[ 452959.4726033 ]
[ 614242.1525722 ]
[  88638.76615977]
[ 354521.38257662]
[ 283316.32227608]
[ 160160.93182208]
[ 211962.82043739]
[  44652.71941815]
[ -30047.53604709]
[ 222999.60116169]
[ 171892.91455051]
[1576762.90221834]
[-117336.86688053]
[ 189821.45653523]
[ 143285.34888008]
[ 222753.58325676]
[ 276278.9150022 ]
[  58600.21286757]
[ 325145.47715266]
[ 281444.86457372]
[ 472209.690806  ]
[-816961.40997074]
[ 247852.36180858]
[ -70837.22531562]
[ 151194.93096515]
[ 284018.23283725]
[ 111985.69315093]
[ 312724.55998696]
[  55987.84656954]
[ 177957.84573776]
[ 445745.12546704]
[ 350118.41251209]
[  90698.18259992]
[ 221345.6344704 ]
[ 204831.20371871]
[ 238280.88053168]
[   9931.10237753]
[ 186721.62643621]
[  39614.31525852]
[ 285314.58403968]
```

K nearest neighbors:

```
 from sklearn.neighbors import KNeighborsRegressor
model=KNeighborsRegressor()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
print(y_pred)
from sklearn.metrics import mean_squared_error
print(mean_squared_error(y_test,y_pred))
```

```
[[  207101.2]
 [  161583.  ]
 [  485558.6]
 [1069208.6]
 [   57623.2]
 [  201465.4]
 [   61688.2]
 [  113551.8]
 [  131884.2]
 [  170839.8]
 [   67763.2]
 [  321707.8]
 [  344792.2]
 [   87138.2]
 [  209691.2]
 [   62815.4]
 [  177020.4]
 [  138047.4]
 [  917494.2]
 [  125446.4]
 [  160657.  ]
 [  195731.2]
 [   86814.2]
 [  100239.2]
 [   87549.8]
 [  184494.8]
 [   83605.6]
 [   82822.4]
 [  116381.2]
 [   74101.2]
 [   57442.  ]
 [  380975.8]
 [   87193.4]
 [  117481.6]
 [  323559.8]
 [  113551.8]
 [  288291.2]
 [  148697.  ]
 [  145812.6]
 [  214492.4]
 [  213920.4]
 [  242730.8]
 [  232339.4]
 [   79922.4]
 [   69655.2]
 [  207014.2]
 [  155369.2]
 [   87193.4]
 [  206744.4]
 [  313661.  ]
 [   14794.6]
 [  148697.  ]
 [   82031.  ]
 [  151625.8]
```

```
[  32543. ]
[ 380975.8]
[ 152150.2]
[ 256905.4]
[ 189486.4]
[ 106389.6]
[ 102110. ]
[ 694812. ]
[  44212.6]
[  62815.4]
[ 207101.2]
[  48415. ]
[ 181940. ]
[ 104581.8]
[  66464.4]
[ 145812.6]
[  74960.6]
[ 149707.4]
[ 325285.6]
[ 235236.6]
[ 255190.6]
[  85793.6]
[ 437478.6]
[ 595344.4]
[  47817.2]
[  77973. ]
[ 560641. ]
[ 128703.8]
[ 104774.8]
[  73404.4]
[ 231107.4]
[  90265.8]
[  94409. ]
[ 314851.8]
[  42315.6]
[ 206757.2]
[ 170839.8]
[  55040.8]
[ 125056.4]
[ 113551.8]
[  54918. ]
[ 219611.6]
[ 124834.6]
[ 125056.4]
[ 128303.2]
[ 254382. ]
[  75098.6]
[ 256284.6]
[ 224598.2]
[ 102853.2]
[ 305915.4]
[  85290.2]
[  46808.4]
[ 127545.6]]
59890802493.224045
```

```
[1004366.4]
[ 181126.8]
[ 296324.4]
[  92921.6]
[ 113483. ]
[  51536.4]
[ 306992. ]
[ 529988.6]
[  44212.6]
[ 126639.4]
[ 173289. ]
[  28486.8]
[  69226.6]
[ 170839.8]
[ 124976.8]
[ 125462. ]
[ 448450.4]
[ 355270.2]
[ 119628.6]
[ 110312. ]
[ 125455. ]
[  62815.4]
[ 256258.6]
[ 199833. ]
[  82031. ]
[  91287.6]
[ 312426.6]
[ 156884.4]
[  75098.6]
[  80214.2]
[  38021.4]
[  85793.6]
[  95028.2]
[ 108238. ]
[  41347. ]
[  95974.6]
[ 248457.2]
[ 173289. ]
[ 203113.6]
[ 214108. ]
[ 140153. ]
[ 196099.2]
[ 108238. ]
[ 207101.2]
[ 314851.8]
[ 255190.6]
[ 125760. ]
[ 140775.6]
[ 578116.6]
[ 125446.4]
[ 133525.2]
[ 513173.4]
[ 392253.8]
[ 211729.2]
[ 213920.4]
[ 309512.2]
[  89523.6]
[ 575398.2]
[ 164478.8]
[ 104598.8]
[ 496827.2]
[ 314851.8]
[  56083. ]
[ 164478.8]
```

Super vector machine:

```
from sklearn.svm import SVR
model=SVR()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
print(y_pred)
from sklearn.metrics import mean_squared_error
print(mean_squared_error(y_test,y_pred))
```

output:

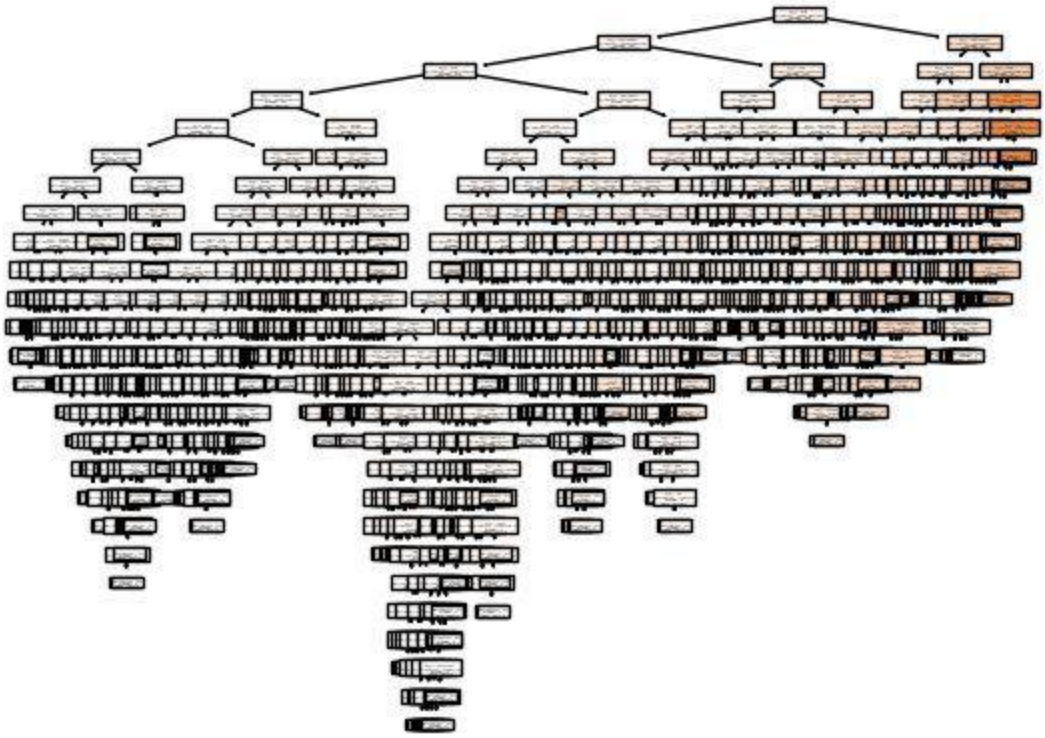
```
[104428.25568324 104374.71779793 104487.83462339 104497.76180899
 104334.22866284 104346.40993691 104333.90588857 104344.69698401
 104348.19717252 104333.83219235 104333.71506085 104510.9551125
 104333.71492256 104338.71038578 104479.08832574 104333.96688321
 104374.50115258 104390.92608357 104474.06620158 104340.13579692
 104396.94237116 104335.20944031 104393.0950223  104348.91936287
 104336.56568592 104341.44324202 104355.84949128 104333.79088394
 104333.71659477 104335.91423543 104336.73410931 104333.73501703
 104333.86523375 104418.83152537 104516.39611388 104344.74849784
 104474.0308931  104408.06859432 104335.15047944 104413.21932002
 104420.29329016 104522.62738047 104498.84167665 104356.21106265
 104334.74704135 104490.52643131 104383.02230635 104333.86087018
 104366.8712742  104334.50355905 104333.74573519 104408.44263162
 104342.84685894 104335.70520037 104521.64608062 104386.75929951
 104510.03413771 104333.85507224 104353.38302672 104336.77550285
 104522.06691711 104483.79193689 104333.71608952 104392.70218518
 104378.88946625 104334.92398707 104333.7151176  104333.83167617
 104333.92523892 104344.57687363 104449.9121075  104449.04894869
 104333.95816781 104333.85017489 104348.41169485 104333.96157046
 104367.13442079 104346.06631925 104342.86983436 104358.27534595
 104439.77974167 104333.81826122 104338.21918597 104339.42692046
 104336.70104564 104349.30639505 104337.41235009 104355.47540678
 104334.13411013 104338.85630489 104522.88243238 104379.36830359
 104333.72660335 104333.71687871 104513.48312285 104341.18471288
 104355.45097805 104428.31496618 104369.35681458 104484.40686163
```

```
 104333.84577015 104333.84341452 104452.63662321 104340.13579703
 104333.91719622 104505.28253033 104405.80596599 104413.97384621
 104420.45317385 104515.00814982 104335.65088765 104517.75943316
 104373.00731395 104346.86867285 104454.69708084 104369.22844356
 104333.93616433 104372.84888403 104363.22239323 104434.33063476
 104334.32784333 104430.38224278 104491.18752646 104341.10572494
 104350.34776182 104333.71508923 104333.80357405 104387.79115898
 104336.04850947 104341.57219955 104333.99059569 104333.71506085
 104333.96966924 104435.77642704 104333.96980179 104333.96917683
 104333.72376099 104463.68575332 104381.23386287 104465.00778564
 104341.57340671 104335.55253913 104333.71523775 104334.54466277
 104364.66544809 104522.87973971 104336.25103078 104518.02276915
 104336.5601012  104333.71503248 104333.73389861 104382.69226053
 104415.26304736 104335.33558233 104340.02789139 104334.64024859
 104518.73687377 104333.71609233 104333.9697497  104427.90442841
 104344.3686069  104365.62704909 104333.72910751 104335.0615367
 104335.14140347 104335.87373311 104333.71659749 104333.71498187
 104457.57093175 104484.42464958 104349.52590751 104516.64497406
 104518.47519905 104335.55185503 104351.34689414 104480.93213631
 104335.77095367 104342.90542975 104361.08298477 104364.94425315
 104339.46826856 104333.71508923 104369.22044567 104333.7158055
 104333.73640028 104333.83219235 104344.48250822 104348.29633508
 104344.76032558 104340.66722577 104349.7900424  104359.54423833
 104348.30865599 104370.72677885 104420.77077902 104338.19344775
 104333.71498187 104346.63103457 104370.39770681 104334.53529212
 104342.56719868 104333.71631065 104336.14671292]
70616071257.95703
```

Decision tree:

```
from sklearn.tree import DecisionTreeRegressor
model=DecisionTreeRegressor()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
print(y_pred)
from sklearn.metrics import mean_squared_error
print(mean_squared_error(y_test,y_pred))
from sklearn import tree
tree.plot_tree(model,filled=True)
```

```
[1.216830e+05 7.213200e+04 2.385370e+05 4.662470e+05 2.986190e+05
 2.731400e+04 2.240800e+04 5.579000e+04 2.626560e+05 8.207900e+04
 2.459300e+04 4.912640e+05 3.653200e+04 1.463000e+04 6.986060e+05
 3.461170e+05 2.693910e+05 1.775990e+05 2.382130e+05 2.608100e+04
 7.534700e+04 8.309400e+04 1.763750e+05 1.370920e+05 5.371700e+04
 3.461170e+05 1.083266e+06 9.164200e+04 1.497500e+04 3.851200e+04
 1.199540e+05 6.531800e+05 6.540300e+04 1.044020e+05 3.042400e+05
 4.139750e+05 2.751040e+05 7.534700e+04 1.037640e+05 1.044020e+05
 2.140350e+05 2.803920e+05 2.657100e+05 8.027700e+04 2.134860e+05
 1.068128e+06 3.498530e+05 9.371000e+03 5.859100e+04 6.506600e+04
 5.693000e+03 1.674580e+05 2.608100e+04 5.246300e+04 9.690290e+05
 2.740460e+05 7.779690e+05 1.053720e+05 1.083984e+06 4.177900e+04
 3.240970e+05 5.280700e+05 1.250400e+04 1.556850e+05 2.693240e+05
 5.888400e+04 2.925300e+04 3.112900e+04 1.056300e+04 1.037640e+05
 2.610750e+05 7.295090e+05 8.994000e+03 6.260200e+04 1.706500e+04
 2.134860e+05 2.986190e+05 1.287540e+05 7.905440e+05 1.145080e+05
 5.600260e+05 4.448410e+05 2.344100e+04 2.712800e+04 4.083800e+04
 9.589500e+04 1.342080e+05 5.313750e+05 1.830600e+04 2.117900e+04
 7.779690e+05 1.083984e+06 4.279900e+04 2.117900e+04 1.562060e+05
 3.024040e+05 2.608100e+04 2.817300e+04 7.849800e+04 2.047850e+05
 8.795900e+04 1.813770e+05 2.815500e+04 8.916800e+04 3.229000e+04
 2.231000e+05 2.493010e+05 5.785510e+05 6.995650e+05 5.280700e+05
 2.653710e+05 2.285410e+05 5.427500e+04 2.182100e+05 2.123010e+05
 1.693250e+05 3.653200e+04 7.213200e+04 4.688700e+04 6.140700e+04
 9.126300e+04 1.801020e+05 5.957900e+04 6.823600e+04 3.928740e+05
 1.497500e+04 5.693000e+03 1.801020e+05 6.530300e+04 9.619000e+04
 1.830600e+04 1.648450e+05 8.360800e+04 1.726800e+05 5.888400e+04
 6.462700e+04 3.762100e+04 7.101800e+05 4.983000e+04 6.986060e+05
 5.371700e+04 2.240800e+04 4.914000e+03 2.101300e+04 1.830550e+05
 2.269080e+05 1.986900e+04 2.225810e+05 4.083800e+04 4.947200e+04
 7.352700e+04 1.616580e+05 1.414650e+05 7.185900e+04 1.765050e+05
 2.880000e+02 7.101800e+05 1.918100e+04 5.640620e+05 2.635670e+05
 3.843700e+04 2.115970e+05 5.262100e+04 1.015190e+05 1.053720e+05
 1.840000e+02 3.022600e+04 5.246300e+04 3.881700e+04 2.946000e+05
 1.261070e+05 3.240970e+05 2.405190e+05 2.880000e+02 7.167000e+04
 2.556835e+06 8.944900e+04 6.531800e+05 3.168700e+04 2.372070e+05
 3.077300e+04 8.022800e+04 1.145080e+05 6.531800e+05 3.022600e+04
 1.648450e+05 5.313750e+05 2.473000e+04 6.294700e+04 7.708500e+04
 5.465200e+04 7.213200e+04 2.257700e+04 4.030400e+04 6.995650e+05
 1.987100e+04 2.411500e+04 1.329240e+05 7.905440e+05 9.825200e+04
 7.905440e+05 9.914500e+04 3.175300e+04]
38616395657.881775
```



6. CONCLUSION

In conclusion, predicting box office collections using machine learning (ML) is a complex and challenging task that involves analyzing a wide range of variables such as historical box office data, movie genre, release date, cast and crew, marketing efforts, and social media sentiment analysis. However, by leveraging the power of ML algorithms such as regression analysis, decision trees, and neural networks, it is possible to accurately predict box office collections for a given movie.

One of the main advantages of using ML for box office prediction is the ability to process vast amounts of data quickly and efficiently, allowing studios and investors to make informed decisions about which movies to invest in and how to market them. ML also enables the identification of patterns and trends that might not be immediately apparent to human analysts, thereby providing insights into the key factors that drive box office success.

Despite the advantages, it's important to note that predicting box office collections using ML is not a foolproof method. There are still many variables that can impact the success of a movie, and unexpected events such as the COVID-19 pandemic can disrupt traditional movie release patterns and make predictions difficult. Nonetheless, the use of ML for box office prediction represents an important tool for the film industry, and advances in technology and data analysis will continue to improve the accuracy of these predictions in the future.

7.FUTURE SCOPE

The future scope of box office prediction in machine learning (ML) is quite promising. With the advancement of ML algorithms and techniques, it has become easier to analyze large amounts of data and make accurate predictions. Here are some possible areas where ML could be applied to predict box office performance:

1. Predicting movie revenue:

ML algorithms can analyze various factors such as genre, cast, release date, marketing budget, and social media buzz to predict the box office revenue of a movie. This can help studios make informed decisions about movie production, marketing, and distribution.

2. Optimizing marketing campaigns:

ML algorithms can analyze audience behavior and preferences to help studios tailor their marketing campaigns to specific demographics. This can help studios make more efficient use of their marketing budget and increase the chances of a movie's success.

3. Identifying potential blockbusters:

ML algorithms can analyze historical box office data to identify patterns and trends that can predict the success of a movie. This can help studios identify potential blockbusters early on and invest in them accordingly.

4. Forecasting the impact of external factors: ML algorithms can analyze external factors such as weather, economic conditions, and competition to predict how they might affect box office performance. This can help studios make informed decisions about when to release a movie and how to position it in the market.

Overall, the future of box office prediction in ML is bright, with potential applications in movie production, marketing, and distribution.

REFERENCES

[1] https://www.kaggle.com/code/akkibansal/box-office-prediction

[2] http://www.ijraset.com/research-paper/imdb-box-office-prediction-using-ml-algorithms

[3] https://github.com/BrianRuizy/box-office-prediction

[4] http://www.researchgate.net/publication/348120181_Prediction_of_Box-office_Success_A_Review_of_Trends_and_Machine_Learning_Computational_Models

[5] https://link.springer.com/article/10.1007/s10796-016-9689-z

[6] https://www.ijraset.com/research-paper/imdb-box-office-prediction-using-ml-algorithms