Shri Vile Parle Kelavani Mandal's
MITHIBAI COLLEGE OF ARTS, CHAUHAN INSTITUTE OF SCIENCE &
AMRUTBEN JIVANLAL COLLEGE OF COMMERCE AND ECONOMICS
(AUTONOMOUS)
NAAC Reaccredited 'A' Grade, CGPA:3.57 (February 2016 to December 2023)
Granted under RUSA for Enhancing Quality & Excellence in select Autonomous Colleges
Granted under FIST-DST & Star College Scheme of DBT, Government of India
Best College (2016–17), University of Mumbai

DEPARTMENT OF STATISTICS

ACADEMICYEAR 2021-22

A Project Report On

"ART OF ARTIFICIAL"

Can You Trust What You See?

Shri Vile Parle Kelavani Mandal's
MITHIBAI COLLEGE OF ARTS, CHAUHAN INSTITUTE OF SCIENCE &
AMRUTBEN JIVANLAL COLLEGE OF COMMERCE AND ECONOMICS
(AUTONOMOUS)
NAAC Reaccredited 'A' Grade, CGPA:3.57 (February 2016 to December 2023)
Granted under RUSA for Enhancing Quality & Excellence in select Autonomous Colleges
Granted under FIST-DST & Star College Scheme of DBT, Government of India
Best College (2016–17), University of Mumbai

DEPARTMENT OF STATISTICS

POST GRADUATION

CERTIFICATE

THIS IS TO CERTIFY THAT,

- *Ms. Gazala Sayyed*
- *Ms. Humaira Khan*
- *Ms. Hunain Shaikh*
- *Ms. Jasmin Shaikh*
- *Ms. Zeenat Khan*
- *Mr. Zishan Sayyed*

HAVE SUCCESSFULLY COMPLETED THE PROJECT ON

"ART OF ARTIFICIAL"
Can You Trust What You See?

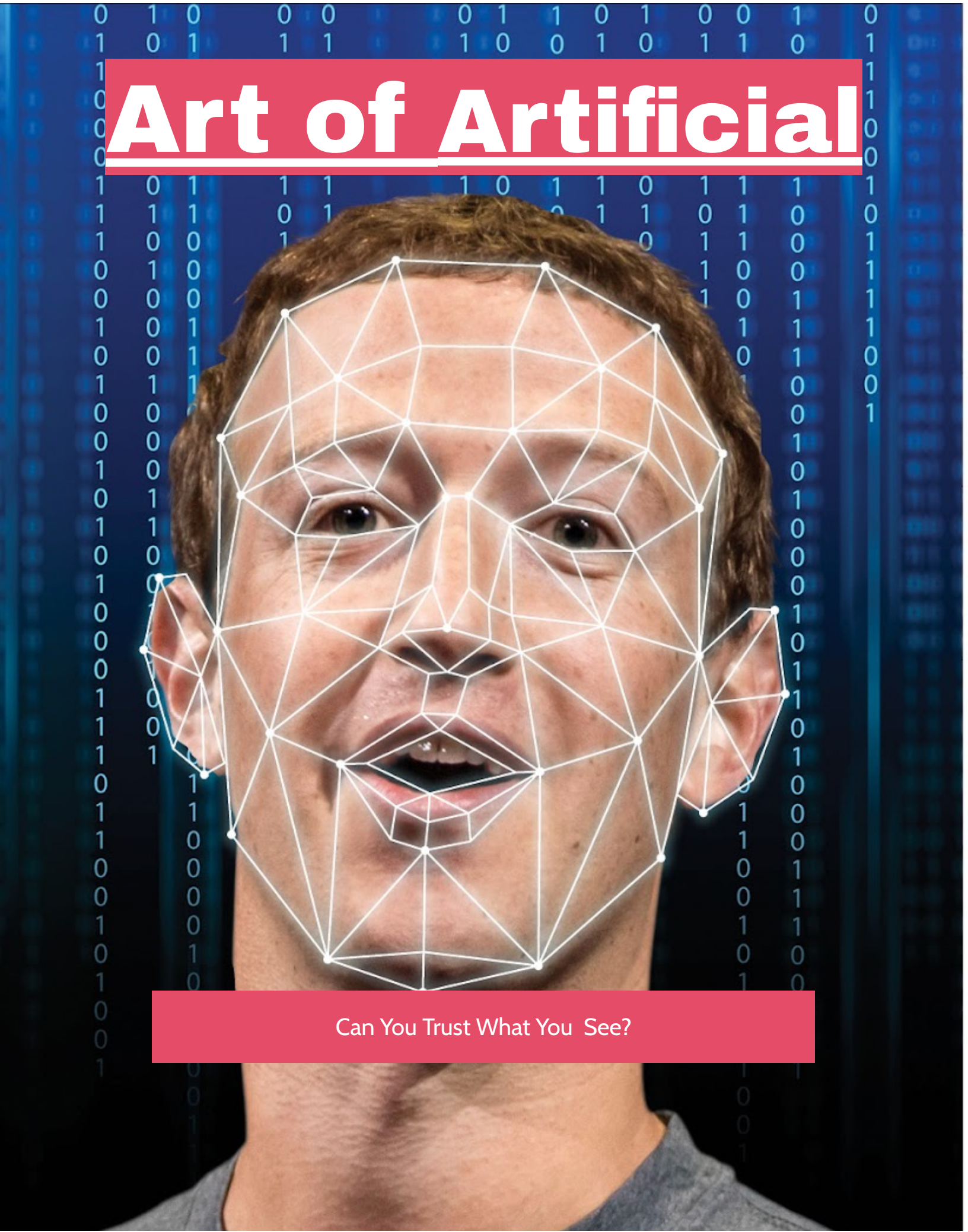
DURING THE ACADEMIC YEAR 2021-22

Mr. AMRIT RAJWADKAR
HEAD,
DEPARTMENT OF STATISTICS

Mr. CHAITANYA ALSHI
INCHARGE
M.Sc. STATISTICS

Mr. ASHISH SHARMA
Chief Manager,
Business Analytics,
Fraud Analytics,
Kotak Bank
EXTERNAL EXAMINER

Art of Artificial



Can You Trust What You See?

Index

Topics:

- Introduction to Deep Fake
- Overview
- Objectives
- **Chapter 1**: Exploratory Data Analysis & Statistical techniques.
  1. Random forest.
  2. Extra tree classifier.
  3. Light Gradient boosting.
  4. Gradient boosting classifier.
  5. Decision tree.
  6.Logistic regression.
  7. Best fit model.
- **Chapter 2**: Creating a Deep fake video.
- **Chapter 3**: Deep Fake Detection
- **Chapter 4**: Use case of Deep fake.
- **Chapter 5**: Important Links
- **Chapter 6**: Resources

Acknowledgement:

We would like to express our gratitude to our coordinator, Mr. Chaitanya Alshi, for supporting and guiding us throughout the project.

We would also like to thank our Professer Priyanka Dangar and one of our MSc friend Mr.Taufique Ahmed Choudhary.

Also towards the principal of Mithibai College,for extending his support and giving us this opportunity.

Also thanking the Department of Statistics,Mithibai College, for providing us the required amenities.

Special thanks to our friends who helped and encouraged us.

Our gratitude towards all the people who helped us by taking some valuable time from their busy schedule and filling our questionnaire.

Introduction:

Deep fake is a technique that aims to replace the face of a targeted person with the face of someone else in a video. It first appeared in the autumn of 2017 as a script used to generate face-swapped adult content. Afterward, this technique was improved by a small community to notably create a user-friendly application called Fake App. The core idea lies in the parallel training of two auto-encoders. Their architecture can vary according to the output size, the desired training time, the expected quality, and the available resources. Traditionally, an auto-encoder designates the chaining of an encoder network and a decoder network. The purpose of the encoder is to perform dimension reduction by encoding the data from the input layer into a reduced number of variables. The goal of the decoder is then to use those variables to output an approximation of the original input. The optimization phase is done by comparing the input and its generated approximation and penalizing the difference between the two, typically using an L^2 distance. In the case of the Deep fake technique, the original auto-encoder is fed with images of resolution 64 × 64 × 3 = 12, 288 variables, encodes those images on 1024 variables, and then generates images with the same size as the input.

The process to generate Deep fake images is to gather aligned faces of two different people A and B, then to train an auto-encoder EA to reconstruct the faces of A from the dataset of facial images of A, and an auto-encoder EB to reconstruct the faces of B from the dataset of facial images of B. The trick consists in sharing the weights of the encoding part of the two auto-encoders EA and EB but keeping their respective decoder separated. Once the optimization is done, any image containing a face of A can be encoded through this shared encoder but decoded with the decoder of EB.

The intuition behind this approach is to have an encoder that privileges encoding general information of illumination, position, and expression of the face and a dedicated decoder for each person to reconstitute constant characteristic shapes and details of the person's face. This may thus separate the contextual information on one side and the morphological information on the other. In practice, the results are impressive, which explains the popularity of the technique. The last step is to take the target video, extract and align the target face from each frame, use the modified auto-encoder to generate another face with the same illumination and expression, and then merge it back in the video.

Fortunately, this technique is far from flawless. Basically, the extraction of faces and their reintegration can fail, especially in the case of face occlusions: some frames can end up with no facial reenactment or with a large blurred area or a doubled facial contour. However, those technical errors can easily be avoided with more advanced networks. More deeply, and this is true for other applications, auto-encoders tend to poorly reconstruct fine details because of the compression of the input data on a limited encoding space, the result thus often appears a bit blurry. A larger encoding space does not work properly since while the fine details are certainly better approximated, on the other hand, the resulting face loses realism as it tends to resemble the input face, i.e. morphological data are passed to the decoder, which is an undesired effect.

Overview

For better understanding, We divided our project in 4 major parts:

1. Statistical analysis:

Involves data collection, data cleaning, exploratory data analysis, and various statistical techniques.

2. Creating deepfake:

Involves creating a deep fake video using DeepFaceLab and exploring it.

3. Detection of deepfake:

This part includes a deep fake video using ML.

4. Future and scope:

What will be the future with deep fake, Pros, cons, and legal aspects you should know.

Objectives

“

•The project aims to understand the audience's behavior towards deepfake .

•Are they able to differentiate between deepfake videos and real videos?

•What are the key parameters to detect a deepfake images/video ?

•Which type of video is most likely to be shared and has an impact in terms of influencing audience perspective?

• Looking at the factors like internet usage platforms used for social media and the probability to be in contact with deep fake videos/images.

•To see the awareness among people with respect to Deepfake.

”

Chapter 1

EXPLORATORY DATA ANALYSIS

COMPUTING VARIOUS DIGITAL PLATFORM USERS IN WORLDWIDE AND IN INDIA:

Internet users worldwide 5,000,000,000		Internet users in India 650,000,000
worldwide **4.3 billion**		In India **500 million**
worldwide **1.40 billion**		In India **231 million**
worldwide **1.6 billion**		In India **487 million**
worldwide **2.9 billion**		In India **240 million**
worldwide **514 billion**		In India **126 million**
worldwide **2.30 billion**		In India **467 million**

STAGE(1):

UNDERSTANDING USAGES(BASED ON GENDER):

NO	QUESTIONS	MALE	FEMALE
1.	**Average Internet Usage**	**4.13**	**4.02**
2.	**Most Used Apps**		
	Youtube	**182**	**205**
	Whatsapp	**174**	**214**
	Instagram	**173**	**193**
	Facebook	**78**	**105**
3.	**Platforms used For News consumption**		
	Google	**152**	**188**
	Youtube	**160**	**181**
	Television(TV)	**77**	**92**
	Others	**73**	**110**
4.	**Type of content consumed**		
	Video	**203**	**223**
	Meme	**115**	**142**
	Audio	**64**	**71**
	Text	**63**	**79**
5.	**Type of medium trusted**		
	Video	**203**	**240**
	Text	**64**	**66**
	Meme	**34**	**24**
	Audio	**43**	**56**

6.	**Among the following whose content is trusted the most?**		

	Sports Person	**160**	**157**
	Celebrity	**93**	**154**
	Religious Leader	**57**	**64**
	Politician	**39**	**53**
7.	**How often do people post videos/images on social media?**		
	Occasionally	**140**	**169**
	Don't Post	**71**	**84**
	Weekly	**15**	**17**
	Daily	**13**	**11**
8.	**Have you ever heard of deepfake?**		
	Yes	**66**	**59**
	No	**173**	**222**

Summary/Conclusions:

- Youtube and whatsapp are the most commonly used apps among people
- For news consumption google and youtube are the top 2 platforms
- Most consumable and trustworthy type of content is Video format
- **75% of the audience is unaware of deep fakes**

STAGE(2):

DEEP FAKE DETECTION(BASED ON GENDER):

NO	QUESTIONS	MALE	FEMALE
1.	**Can you detect a deep fake?**		
	Yes,	**111**	**49**
	No	**62**	**144**
	To some extent	**66**	**88**
2.	**How do you identify a fake video?**		
	Background	**129**	**95**
	Face-edges	**116**	**103**
	Lip-sync	**107**	**83**
	Video Quality	**106**	**82**
3.	**How can you detect a fake image?**		
	Background	**159**	**164**
	Quality	**137**	**174**
	Face-edges	**117**	**103**
	Eyes	**76**	**59**

- **Based on gender males are more confident about detecting deep fake.**
- **Background and face edges are more important features to identify deep fake videos.**
- **Background and quality of image is important aspects to identify deep fake images according to people**

STAGE(3):

TESTING(BASED ON MAJORITY):

NO	QUESTIONS	YES	NO	TO SOME EXTENT
1.	Can you detect a deep fake?	160	206	154
2.	Which one is fake according to you? (tom both fake)			
	Left	41	35	58
	Right	62	41	40
	Both	38	22	33
	None	19	8	23
3.	Which one is fake according to you? (obama rf)			
	Left	51	37	40
	Right	67	140	77
	Both	15	11	10
	None	27	18	27
4.	Which one of them is fake according to you? (obama video 2nd)			
	First	66	35	63
	Second	56	134	40
	Both	24	25	34
	None	14	12	17
5.	Srk_F			
	0% fake	16	24	11
	1%-30%	16	116	23
	31%-60%	16	13	24

	61%-90%	**20**	**15**	**32**
	100% fake	**92**	**38**	**64**
6.	**Nawz_F**			
	0% fake	**20**	**18**	**21**
	1%-30%	**15**	**20**	**25**
	31%-60%	**20**	**16**	**20**
	31%-90%	**10**	**9**	**20**
	100% fake	**95**	**143**	**68**
7.	**Harp_R**			
	0% fake	**78**	**128**	**64**
	1%-30%	**12**	**10**	**25**
	31%-60%	**15**	**24**	**23**
	61%-90%	**16**	**24**	**15**
	100% fake	**39**	**20**	**27**
8.	**Emma_F**			
	0% fake	**72**	**125**	**43**
	1%-30%	**21**	**14**	**31**
	31%-60%	**20**	**27**	**26**
	61%-90%	**14**	**20**	**26**
	100% fake	**33**	**20**	**28**
9.	**Kevin_R**			
	0% fake	**87**	**130**	**76**
	1%-30%	**12**	**16**	**31**
	31%-60%	**21**	**25**	**17**
	61%-90%	**20**	**20**	**14**
	100% fake	**20**	**15**	**16**
10.	**Anush_F**			
	0% fake	**22**	**14**	**18**
	1%-30%	**12**	**18**	**27**

	31%-60%	**21**	**14**	**18**
	61%-90%	**14**	**8**	**22**
	100% fake	**91**	**152**	**69**
11.	**Kajol_F**			
	0% fake	**81**	**150**	**60**
	1%-30%	**16**	**13**	**39**
	31%-60%	**19**	**19**	**22**
	61%-90%	**16**	**9**	**19**
	100% fake	**28**	**15**	**14**
12.	**Will_F**			
	0% fake	**72**	**133**	**59**
	1%-30%	**19**	**22**	**20**
	31%-60%	**21**	**20**	**35**
	61%-90%	**23**	**13**	**20**
	100% fake	**25**	**18**	**20**
13.	**Emma_R**			
	0% fake	**93**	**140**	**84**
	1%-30%	**15**	**12**	**19**
	31%-60%	**13**	**24**	**27**
	61%-90%	**13**	**12**	**8**
	100% fake	**26**	**18**	**16**
14.	**Jony_R**			
	0% fake	**51**	**29**	**46**
	1%-30%	**28**	**114**	**32**
	31%-60%	**38**	**15**	**39**
	61%-90%	**21**	**24**	**22**
	100% fake	**22**	**23**	**16**
15.	**Nasir_F**			
	0% fake	**18**	**37**	**23**

	1%-30%	**15**	**20**	**21**
	31%-60%	**13**	**13**	**15**
	61%-90%	**8**	**14**	**14**
	100% fake	**106**	**130**	**81**

Check Out complete interactive Dashboard:
:https://datastudio.google.com/reporting/b7403197-bbf8-4854-ad8e-da10c07c14da

Statistical Techniques

Before Diving Into Statistical Techniques a Brief about Library Used for Analysis Purpose

Introduction to PyCaret:

What is PyCaret?

PyCaret is an open-source low-code machine learning library in Python that aims to reduce the time needed for experimenting with different machine learning models.
It helps Data scientists to perform any experiments end-to-end quickly and more efficiently.
PyCaret being a low-code library makes you more productive. With less time spent coding, you and your team can now focus on business problems.
PyCaret is a wrapper around many ML models and frameworks such as XGBoost, Scikit-learn, and many more.

Why use PyCaret?

It helps in data preprocessing.
It trains multiple models simultaneously and outputs a table comparing the performance of each model by considering a few performance metrics such as precision, recall, f1-score, and so on.
It is easy to analyze and interpret as it requires minimal codes to run.
In a few lines of code, it increases the productivity

ID of an estimator available in the model library or passes an untrained model object consistent with scikit-learn API.
Estimators available in the model library (ID - Name):

- ‘lr’ - Logistic Regression
- ‘knn’ - K Neighbors Classifier
- ‘nb’ - Naive Bayes
- ‘dt’ - Decision Tree Classifier
- ‘svm’ - SVM - Linear Kernel
- ‘rbfsvm’ - SVM - Radial Kernel

- ‘gpc’ - Gaussian Process Classifier
- ‘mlp’ - MLP Classifier
- ‘ridge’ - Ridge Classifier
- ‘rf’ - Random Forest Classifier
- ‘qda’ - Quadratic Discriminant Analysis
- ‘ada’ - Ada Boost Classifier
- ‘gbc’ - Gradient Boosting Classifier
- ‘lda’ - Linear Discriminant Analysis
- ‘et’ - Extra Trees Classifier
- ‘xgboost’ - Extreme Gradient Boosting
- ‘lightgbm’ - Light Gradient Boosting Machine
- ‘catboost’ - CatBoost Classifier

Understand Different Terms You Should Know:

Accuracy:
Accuracy means that how many data points are predicted correctly. It is one of the simplest forms of evaluation metrics.

Precision:
Precision is a metric for binary classifier which measures the correctness among all positive labels.Precision is the ratio between the number of correct positive answers (true positive) and the sum of correct positive answers (true positive) and wrong but positively labeled answers (false positive).

AUC:
AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1). AUC provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example. AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0.

Recall:
Recall is a metric for binary classifier which measures how many positive labels are successfully predicted amongst all positive labels. Formally, it is the ratio between the number of correct positive answer (true positive) and the sum of correct positive answer (true positive) and wrongly negatively labeled answer (false negative).

F1 Score
F1-Score is the harmonic mean of precision and recall values for a classification problem. The formula for F1-Score is as follows:

MCC:
MCC is computed directly from classification labels, which means that it was used a single threshold value to transform probabilities into classification labels, no matter what the threshold value was. AUC, on the other hand is using the whole range of threshold values.

Model Classifiers:

sr no	description	values
0	session_id	786
1	Target	detect_fake_ video?
2	Target Type	Multiclass
3	Label Encoded	Maybe: 0, No: 1, Yes: 2
4	Original Data	(520, 79)
5	Missing Values	FALSE
6	Numeric Features	12
7	Categorical Features	66
8	Ordinal Features	FALSE
9	High Cardinality Features	FALSE
10	High Cardinality Method	None
11	Transformed Train Set	(363, 122)
12	Transformed Test Set	(157, 122)

1.)-RANDOM FOREST

Random forest classifier creates a set of decision trees from a randomly selected subset of the training set. It then aggregates the votes from different decision trees to decide the final class of the test object.

Random forests are a type of machine learning algorithm that is used for classification and regression tasks. A classifier model takes data input and assigns it to one of several categories. For example, given a set of images consisting of dogs and cats images, a classifier could be used to predict whether each image is of a dog or a cat. In a nutshell, a random forest algorithm works by creating multiple decision trees, each of which is based on a random subset of the data. Decision trees are a type of algorithm that makes predictions by looking at the data inputs and determining which category they belong to. Random forests take this one step further by creating multiple decision trees and then averaging their results. This helps to reduce the chance of overfitting, which is when the algorithm only works well on the training data and not on new data. Random forests are powerful tools for machine learning and can be used for a variety of tasks such as facial recognition, fraud detection, predicting consumer behavior, and stock market predictions.

A random forest can be considered as an ensemble of several decision trees. The idea is to aggregate the prediction outcome of multiple decision trees and create a final outcome based on the averaging mechanism (majority voting). It helps the model trained using the random forest to generalize better with the larger population. In addition, the model becomes less susceptible to overfitting / high variance. Here are the key steps of the random forest algorithm:

Take a random sample of size n (randomly choose n examples with replacement – bootstrap)
Grow the decision tree from the above sample based on the following:
Select m features in a random manner out of all the features
Create the tree by splitting the data using m features based on the objective function (maximizing the information gain)
Repeat the above steps for k number of trees as specified.

Aggregate the prediction outcome of different trees and come up with a final prediction based on majority voting or averaging

Model:

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=100,
                       n_jobs=-1, oob_score=False, random_state=786, verbose=0,
                       warm_start=False)
```

Model evaluation:

Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	
Fold							
0	0.7568	0.9047	0.7348	0.7414	0.7431	0.6304	0.6354
1	0.8649	0.9457	0.8561	0.8667	0.8649	0.7965	0.7974
2	0.8649	0.947	0.8626	0.8826	0.8676	0.7974	0.8035
3	0.8056	0.9652	0.7838	0.8088	0.8032	0.7028	0.7062
4	0.8889	0.9778	0.8808	0.8986	0.8908	0.8312	0.8341
5	0.8056	0.9525	0.7818	0.7986	0.794	0.6993	0.7071
6	0.8333	0.9533	0.8172	0.8443	0.8361	0.7468	0.7494
7	0.75	0.9261	0.7278	0.7525	0.7505	0.6224	0.6231
8	0.8056	0.9164	0.7889	0.7998	0.8018	0.7049	0.7057
9	0.7778	0.9346	0.75	0.7768	0.774	0.662	0.6651
Mean	0.8153	0.9423	0.7984	0.817	0.8126	0.7194	0.7227
Std	0.0448	0.0211	0.0515	0.0514	0.0481	0.0684	0.0685

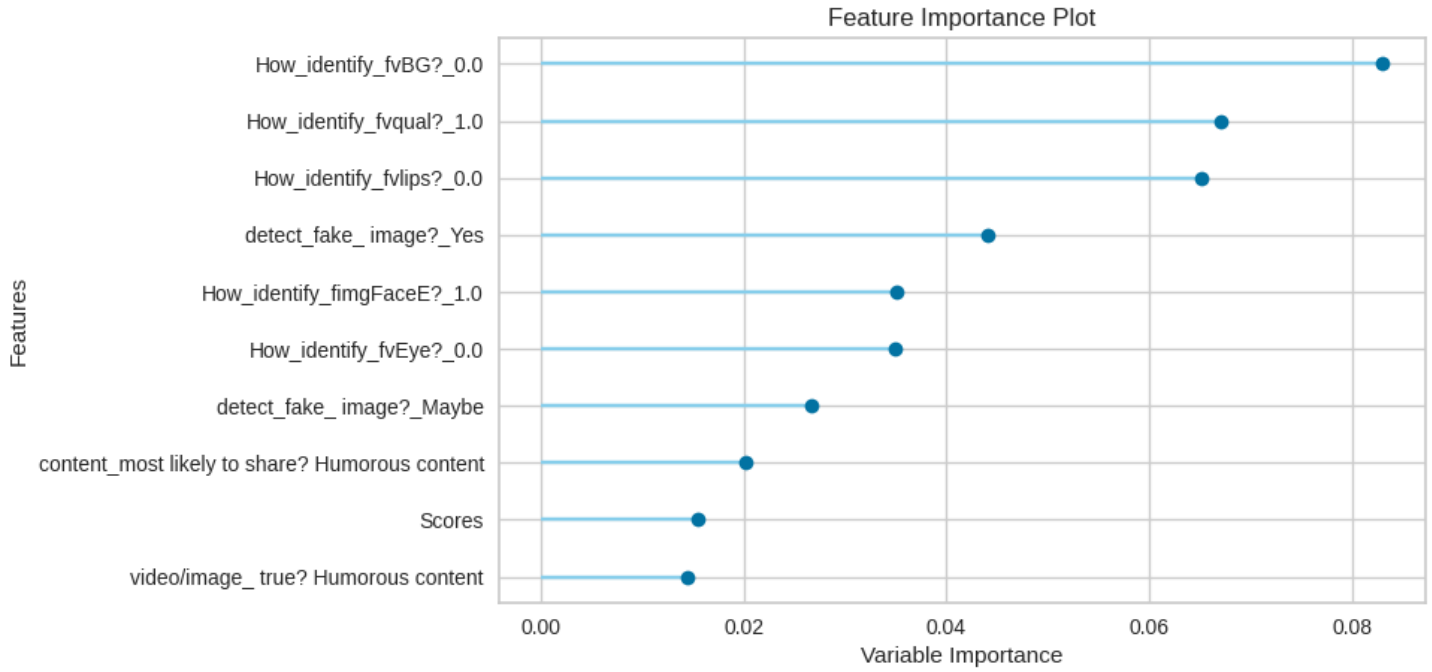
Prediction:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	Random Forest Classifier	0.8471	0.9424	0.8308	0.8449	0.8453	0.7677	0.7682

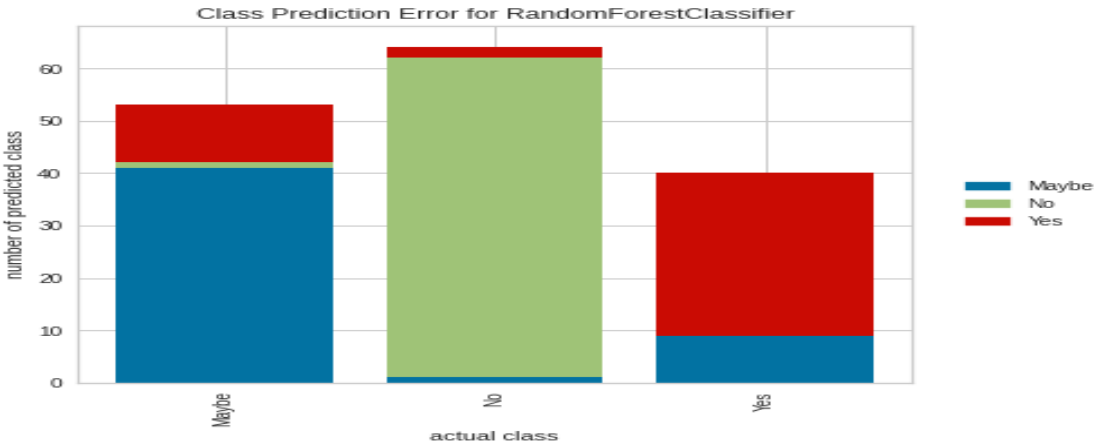
detect_fake_video?	Label	Score
No	No	1
No	No	0.68
Maybe	Maybe	0.49
No	No	1
Yes	Yes	0.51
...	...	...

No	No	0.76
Maybe	Yes	0.37
No	No	0.83
Maybe	Yes	0.37
No	No	0.8

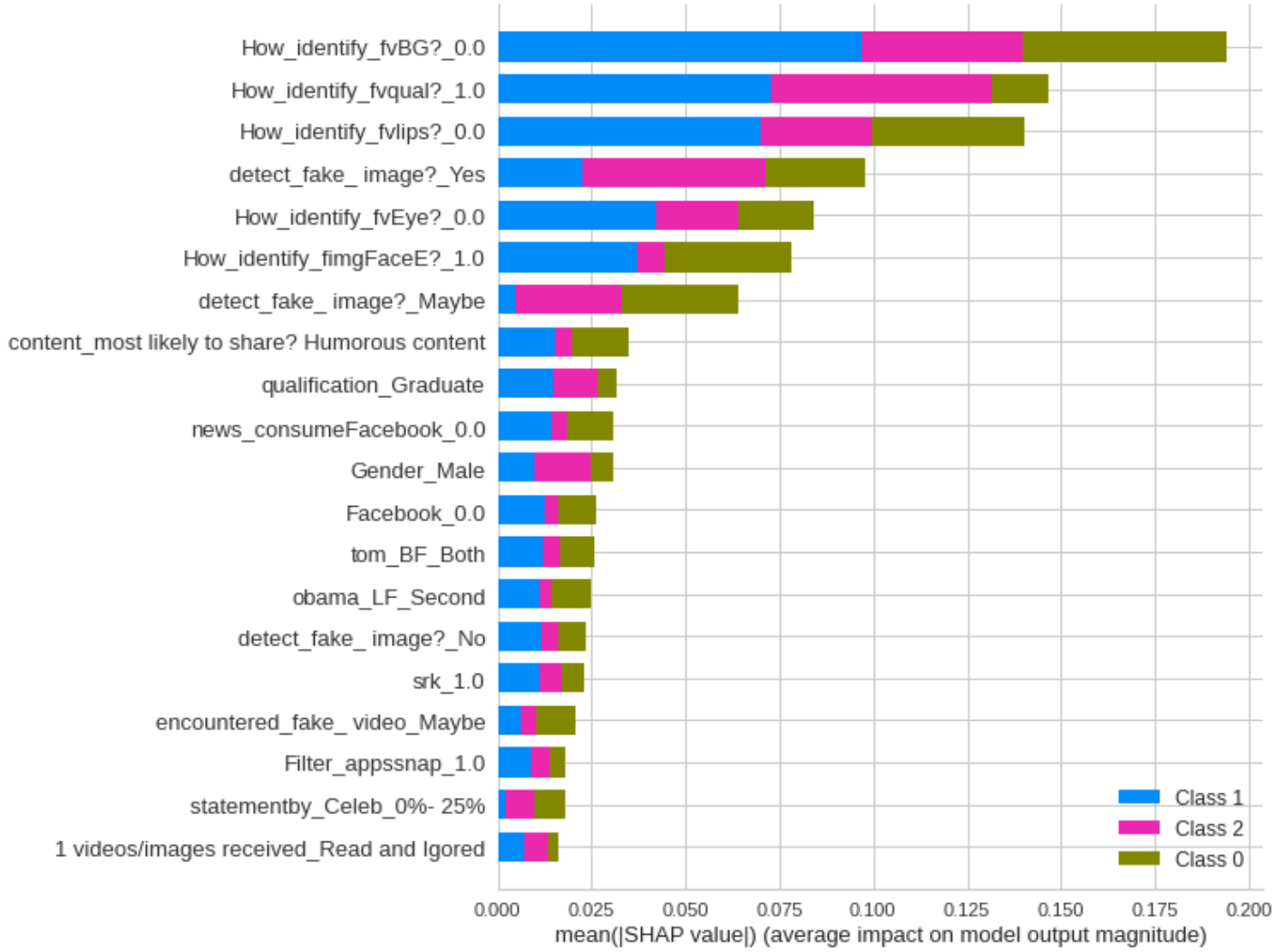
Important Features:



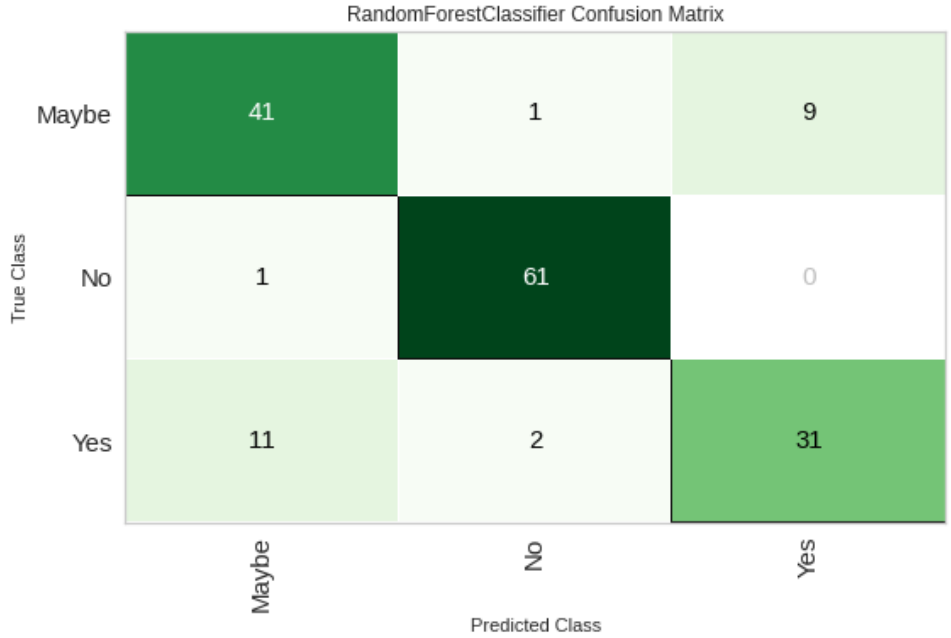
Prediction Error:



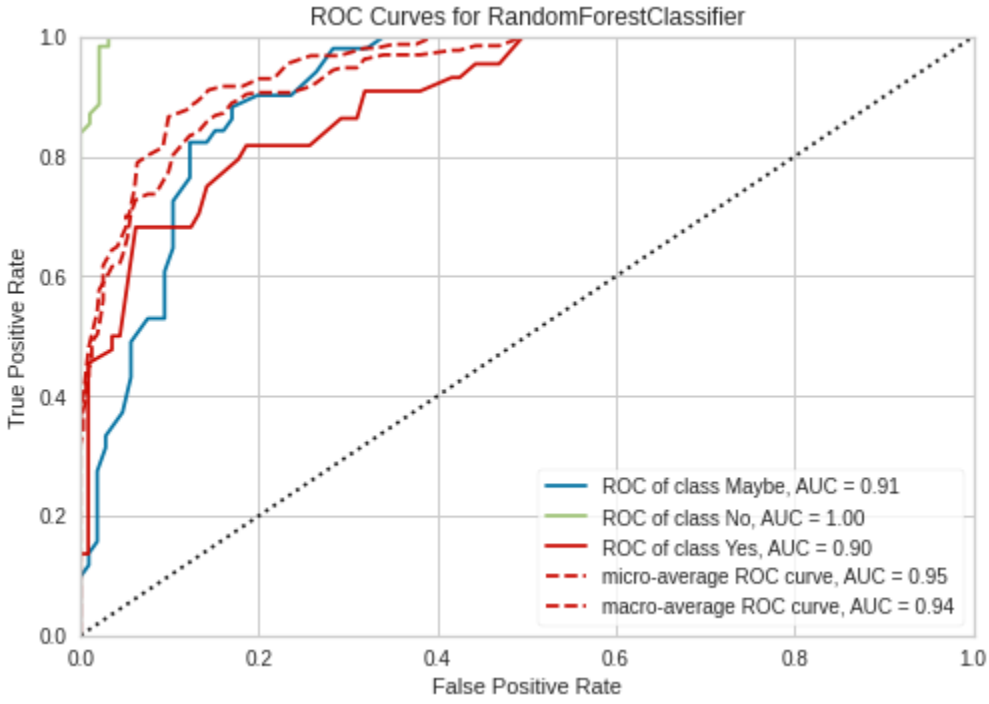
model interpretation:



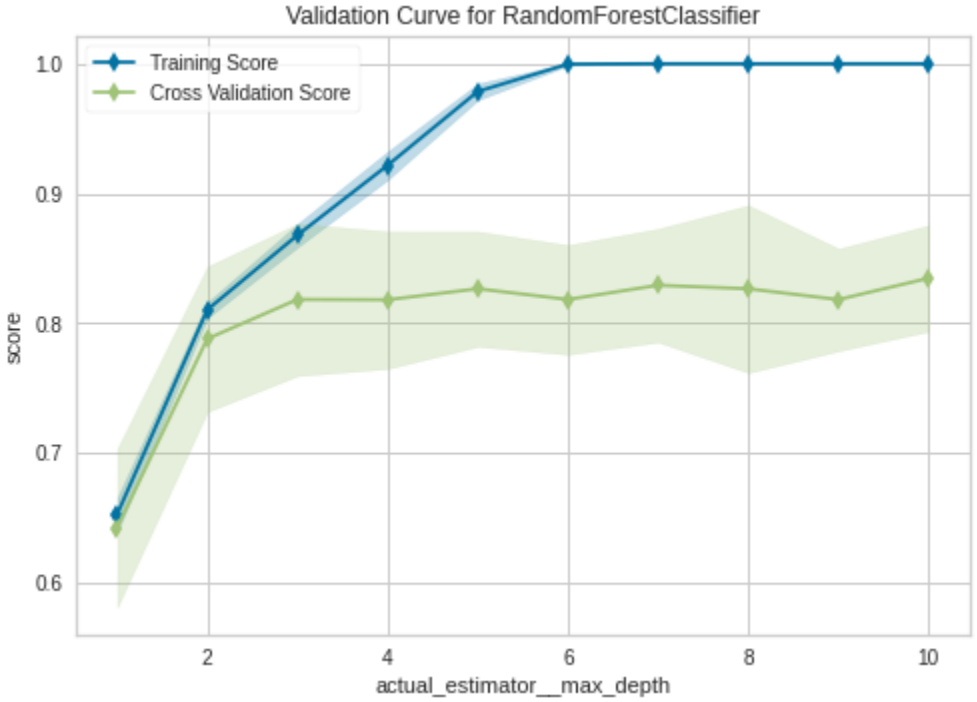
Confusion Matrix:



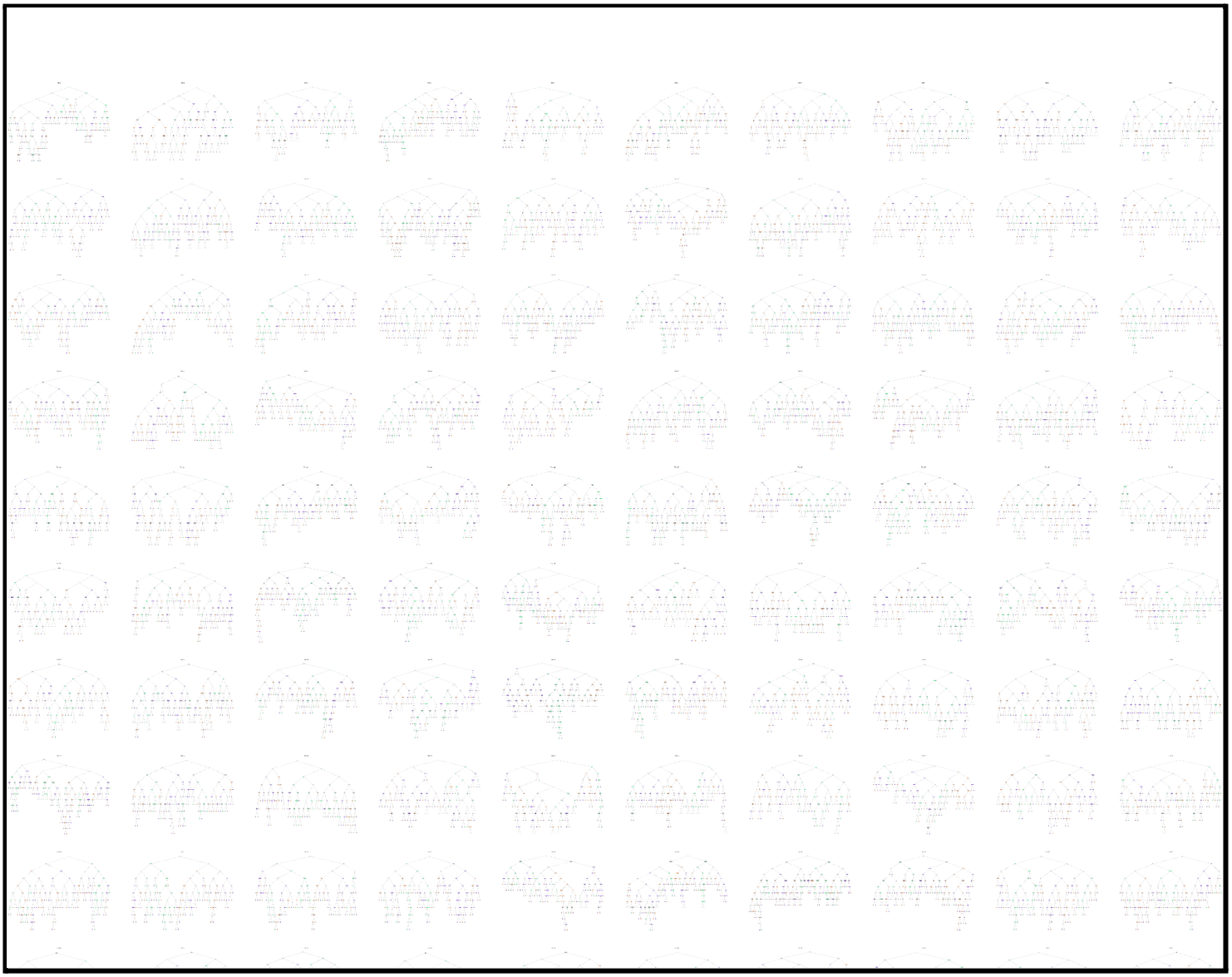
Auc Curve:



validation Curve:



Random Forest:



2.)EXTRA TREE CLASSIFIER

Prerequisites: Decision Tree Classifier
Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique that aggregates the results of multiple de-correlated decision trees collected in a “forest” to output its classification result. In concept, it is very similar to a Random Forest Classifier and only differs from it in the manner of construction of the decision trees in the forest.
Each Decision Tree in the Extra Trees Forest is constructed from the original training sample. Then, at each test node, Each tree is provided with a random sample of k features from the feature-set from which each decision tree must select the best feature to split the data based on some mathematical criteria (typically the Gini Index). This random sample of features leads to the creation of multiple de-correlated decision trees.
To perform feature selection using the above forest structure, during the construction of the forest, for each feature, the normalized total reduction in the mathematical criteria used in the decision of feature of split (Gini Index if the Gini Index is used in the construction of the forest) is computed. This value is called the Gini Importance of the feature. To perform feature selection, each feature is ordered in descending order according to the Gini Importance of each feature and the user selects the top k features according to his/her choice.

model;

```
ExtraTreesClassifier(bootstrap=False, ccp_alpha=0.0, class_weight=None,
                     criterion='gini', max_depth=None, max_features='auto',
                     max_leaf_nodes=None, max_samples=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=-1,
                     oob_score=False, random_state=786, verbose=0,
                     warm_start=False)
```

Model Evaluation:

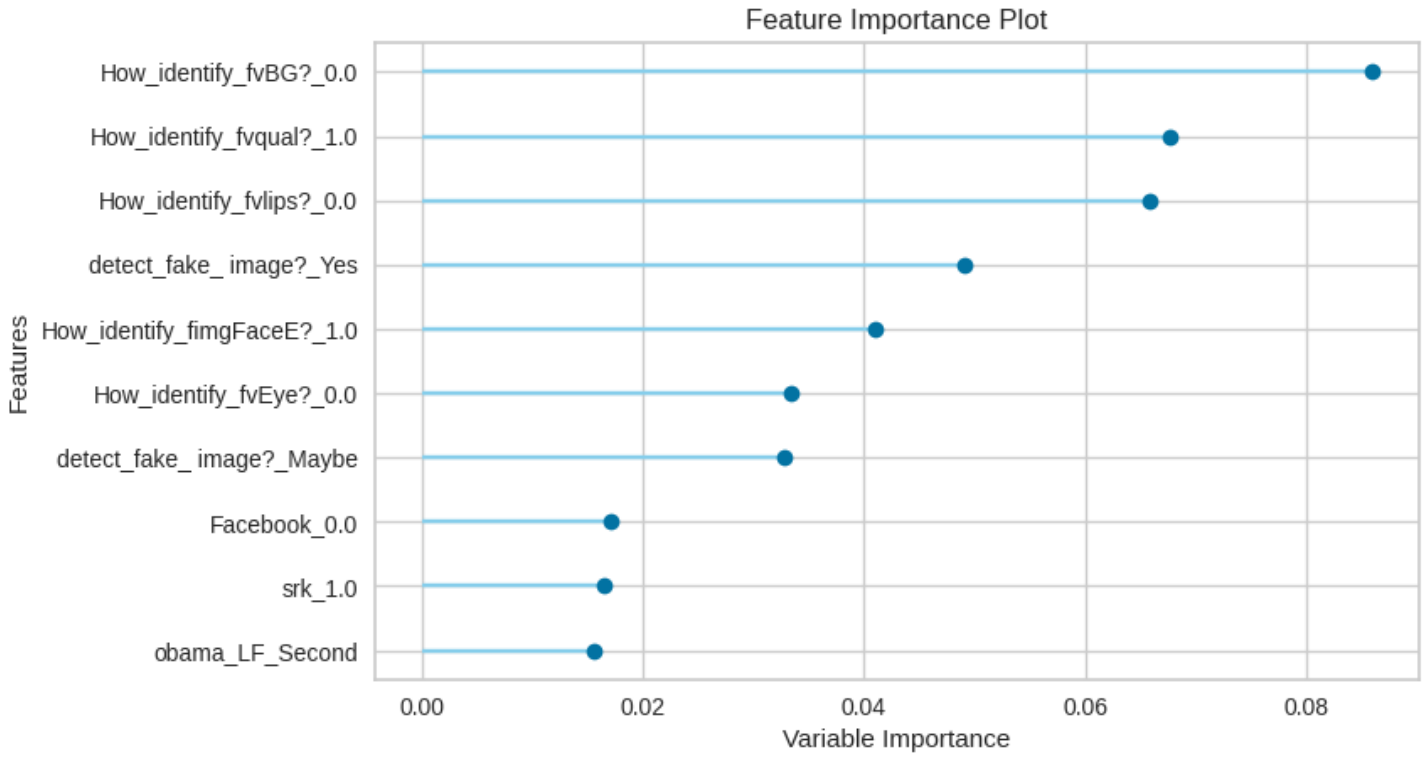
	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Fold							
0	0.7568	0.8913	0.7348	0.7414	0.7431	0.6304	0.6354
1	0.8919	0.9353	0.8838	0.891	0.8904	0.8365	0.8374
2	0.8108	0.9344	0.8149	0.834	0.8141	0.7176	0.7247
3	0.8056	0.956	0.7838	0.8234	0.805	0.7025	0.7101
4	0.9444	0.989	0.9333	0.953	0.9437	0.9151	0.9195
5	0.7222	0.9205	0.6848	0.7119	0.7147	0.575	0.577
6	0.8056	0.9318	0.7838	0.8032	0.8035	0.7032	0.704
7	0.7778	0.9327	0.7611	0.7843	0.7778	0.6651	0.6682
8	0.8333	0.9086	0.8222	0.8316	0.8295	0.7477	0.7503
9	0.7778	0.9159	0.7556	0.7778	0.7778	0.6636	0.6636
Mean	0.8126	0.9316	0.7958	0.8152	0.81	0.7156	0.719
Std	0.0616	0.0254	0.0685	0.0664	0.0636	0.0938	0.094

Prediction:

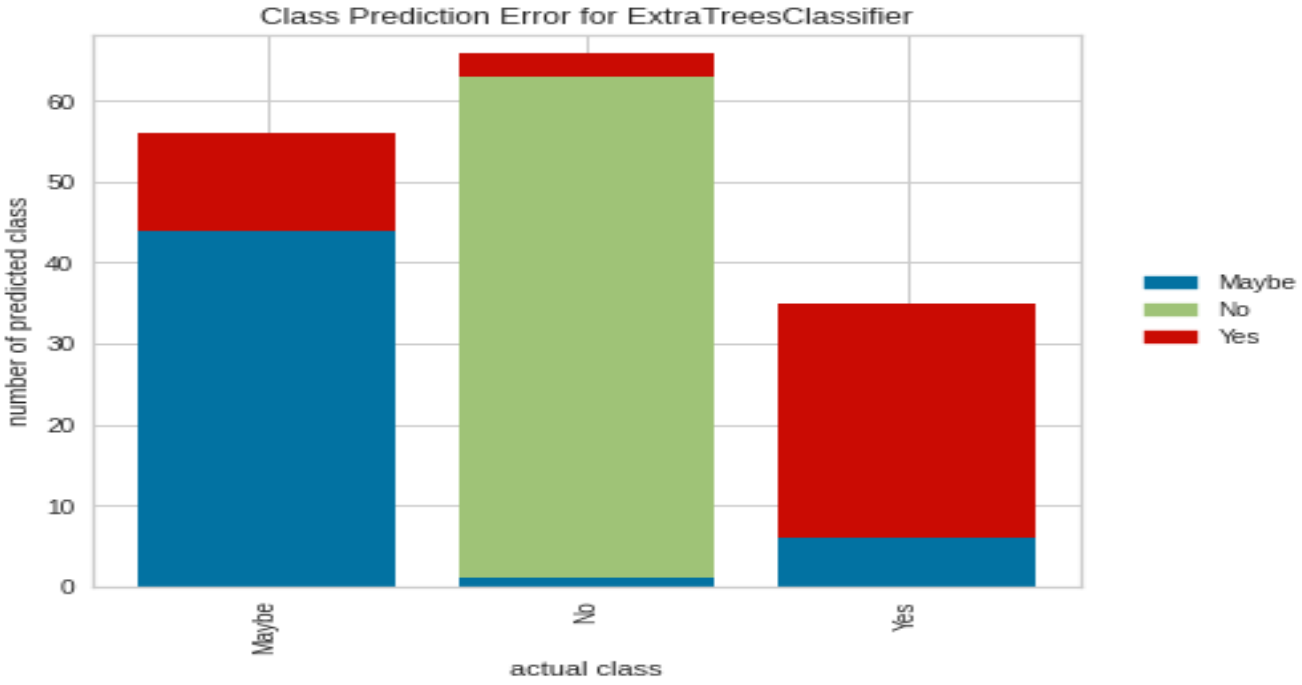
Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Extra Trees Classifier	0.8599	0.9408	0.8406	0.8584	0.8555	0.7863	0.7893

detect_fake_ video?	Label	Score
No	No	1
No	No	0.82
Maybe	Maybe	0.47
No	No	1
Yes	Yes	0.49
...	...	...
No	No	0.72
Maybe	No	0.38
No	No	0.94
Maybe	Maybe	0.35
No	No	0.79

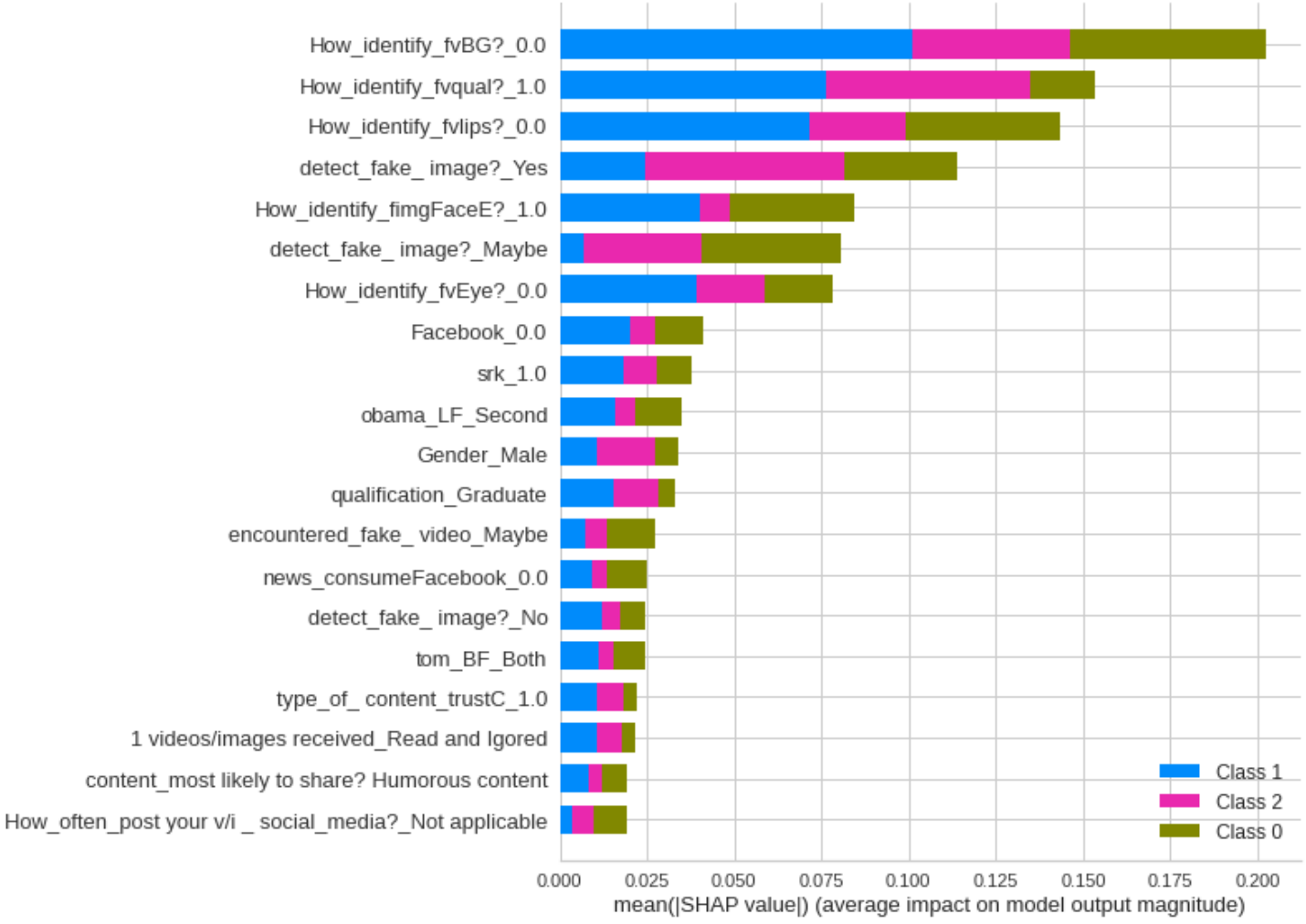
Important Features:



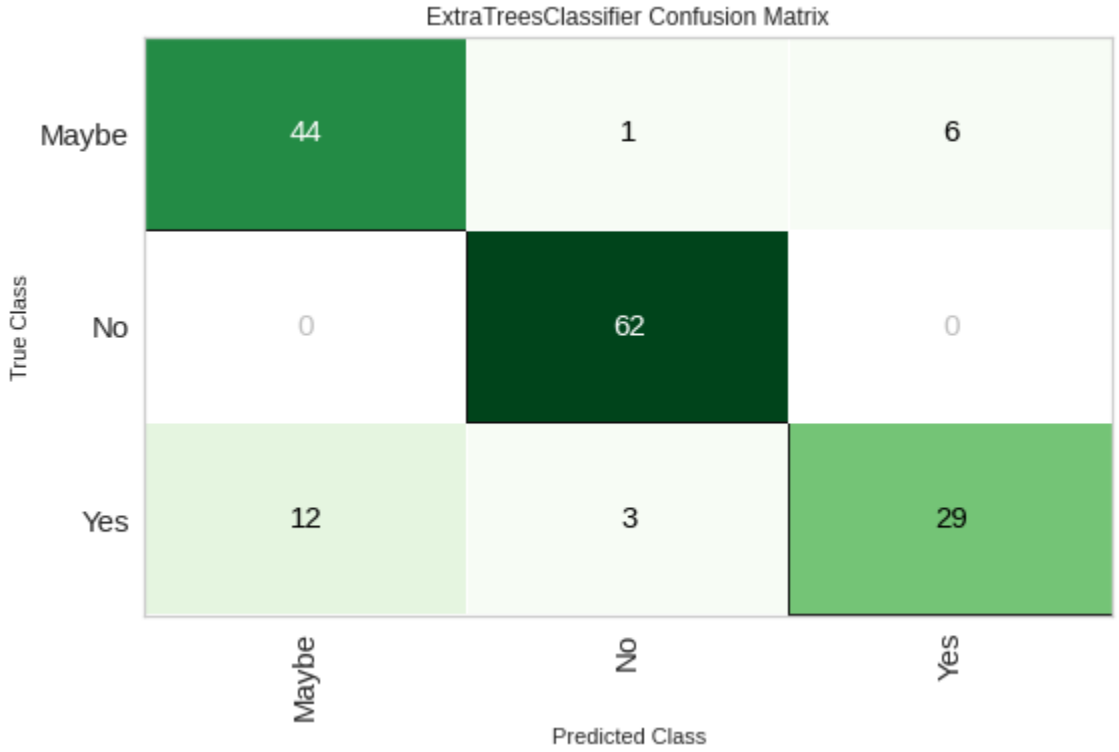
Prediction Error:



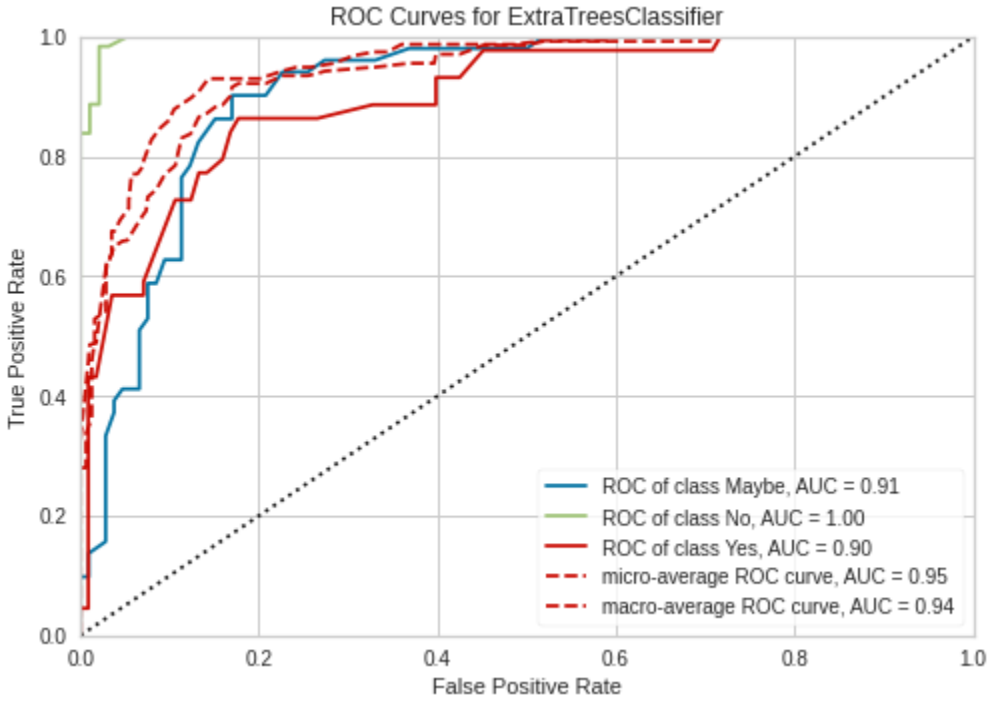
model interpretation:



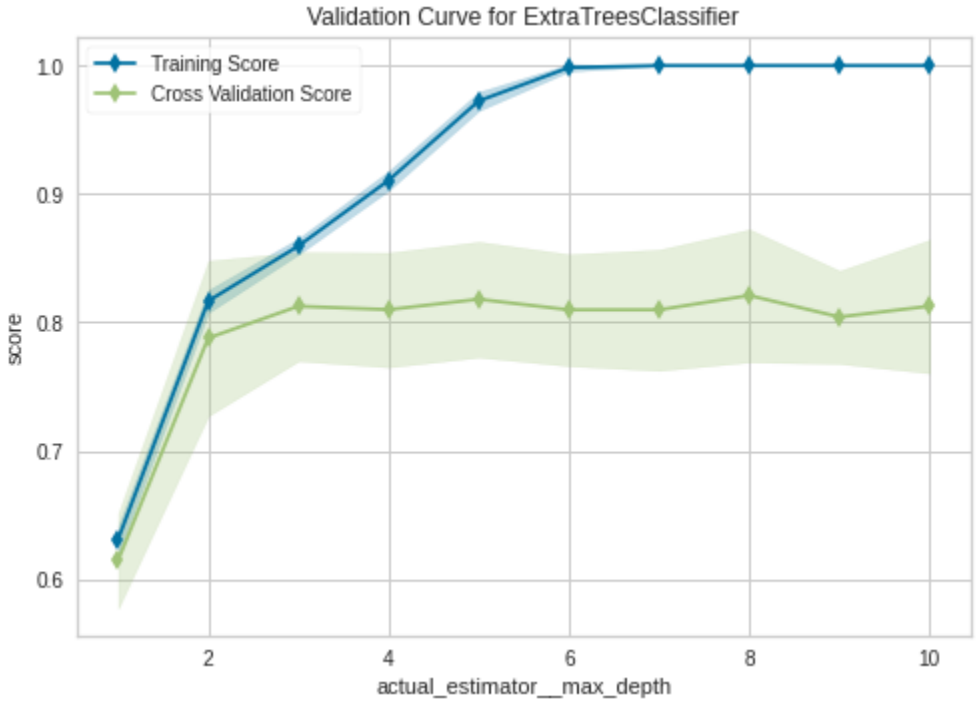
Confusion Matrix:



Auc Curve:



validation Curve:



3.)LIGHT GRADIENT BOOSTING

Light Gradient Boosted Machine, or LightGBM for short, is an open-source library that provides an efficient and effective implementation of the gradient boosting algorithm.

LightGBM extends the gradient boosting algorithm by adding a type of automatic feature selection as well as focusing on boosting examples with larger gradients. This can result in a dramatic speedup of training and improved predictive performance.

As such, LightGBM has become a de facto algorithm for machine learning competitions when working with tabular data for regression and classification predictive modeling tasks. As such, it owns a share of the blame for the increased popularity and wider adoption of gradient boosting methods in general, along with Extreme Gradient Boosting (XGBoost).

Light Gradient Boosted Machine (LightGBM) is 1. an efficient open-source implementation of the stochastic gradient boosting ensemble algorithm.
2. How to develop LightGBM ensembles for classification and regression with the scikit-learn API.
3. How to explore the effect of LightGBM model hyperparameters on model performance.

ARCHITECTURE:

LightGBM splits the tree leaf-wise as opposed to other boosting algorithms that grow tree level-wise. It chooses the leaf with maximum delta loss to grow. Since the leaf is fixed, the leaf-wise algorithm has lower loss compared to the level-wise algorithm. Leaf-wise tree growth might increase the complexity of the model and may lead to overfitting in small datasets.
Below is a diagrammatic representation of leaf-wise tree diagram.

content://com.android.chrome.FileProvider/images/screenshot/16511345880721547509853681454117.png

Code: Python Implementation of LightGBM Model:
The data set used for this example is Breast Cancer Prediction. Click on this to get data set :
https://www.kaggle.com/lbronchal/breast-cancer-dataset-analysis?select=data.csv

Parameter Tuning
Few important parameters and their usage is listed below :

1. max_depth : It sets a limit on the depth of tree. The default value is 20. It is effective in controlling over fitting.
2. categorical_feature : It specifies the categorical feature used for training model.
3. bagging_fraction : It specifies the fraction of data to be considered for each iteration.
4. num_iterations : It specifies the number of iterations to be performed. The default value is 100.
5. num_leaves : It specifies the number of leaves in a tree. It should be smaller than the square of max_depth.
6. max_bin : It specifies the maximum number of bins to bucket the feature values.
7. min_data_in_bin : It specifies minimum amount of data in one bin.
8. task : It specifies the task we wish to perform which is either train or prediction. The default entry is train. Another possible value for this parameter is prediction.
9. feature_fraction : It specifies the fraction of features to be considered in each iteration. The default value is one.

model;

```
LGBMClassifier(boosting_type='gbdt', class_weight=None, colsample_bytree=1.0,
               importance_type='split', learning_rate=0.1, max_depth=-1,
               min_child_samples=20, min_child_weight=0.001, min_split_gain=0.0,
               n_estimators=100, n_jobs=-1, num_leaves=31, objective=None,
               random_state=786, reg_alpha=0.0, reg_lambda=0.0, silent='warn',
               subsample=1.0, subsample_for_bin=200000, subsample_freq=0)
```

Model Evaluation:

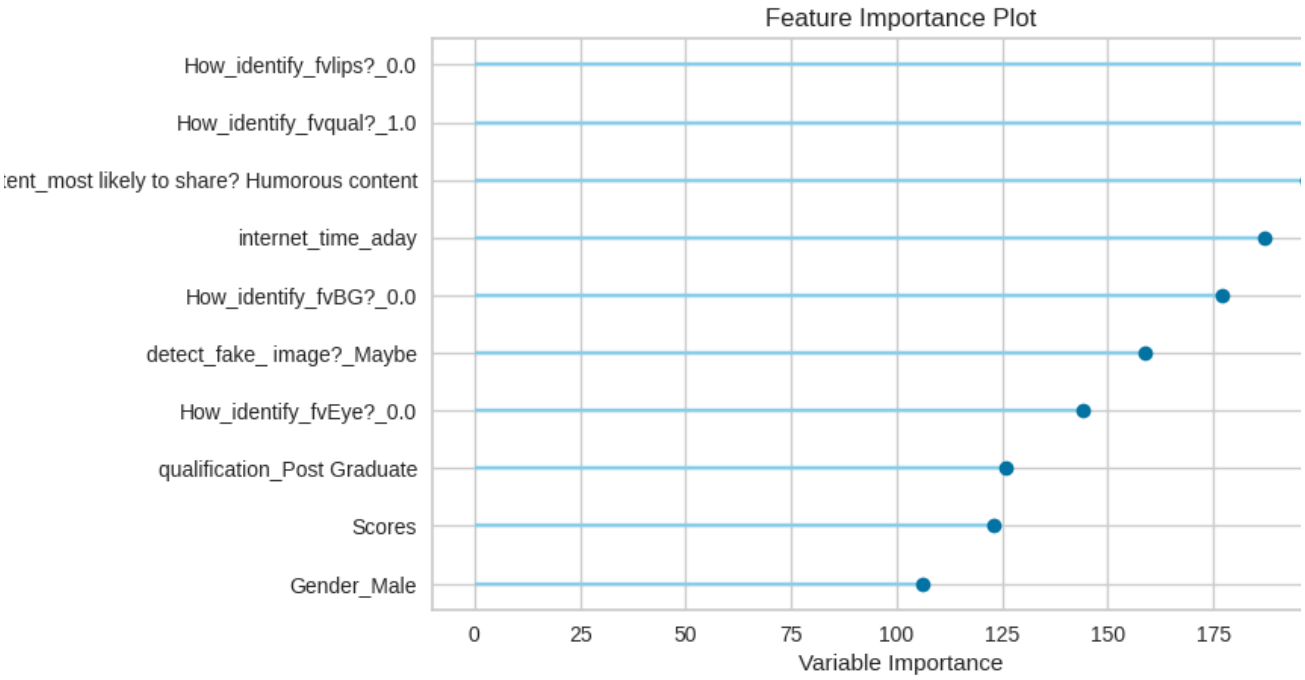
Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7027	0.8929	0.6768	0.7	0.6957	0.5503	0.5559
1	0.8378	0.9301	0.8283	0.8437	0.8372	0.756	0.7594
2	0.8378	0.9141	0.8283	0.8437	0.8372	0.756	0.7594
3	0.8333	0.9509	0.8091	0.8333	0.8333	0.7459	0.7459
4	0.8889	0.9815	0.8667	0.9185	0.8835	0.8298	0.846
5	0.8056	0.927	0.7758	0.8056	0.8047	0.7032	0.704
6	0.8611	0.9291	0.8394	0.8619	0.8605	0.788	0.7889
7	0.8056	0.9022	0.7944	0.8191	0.8043	0.7077	0.7151
8	0.75	0.9265	0.7278	0.7525	0.7505	0.6224	0.6231
9	0.7778	0.9043	0.7556	0.7778	0.7778	0.6636	0.6636
Mean	0.8101	0.9259	0.7902	0.8156	0.8085	0.7123	0.7161
Std	0.0521	0.0245	0.0545	0.0579	0.0526	0.0783	0.08

Prediction:

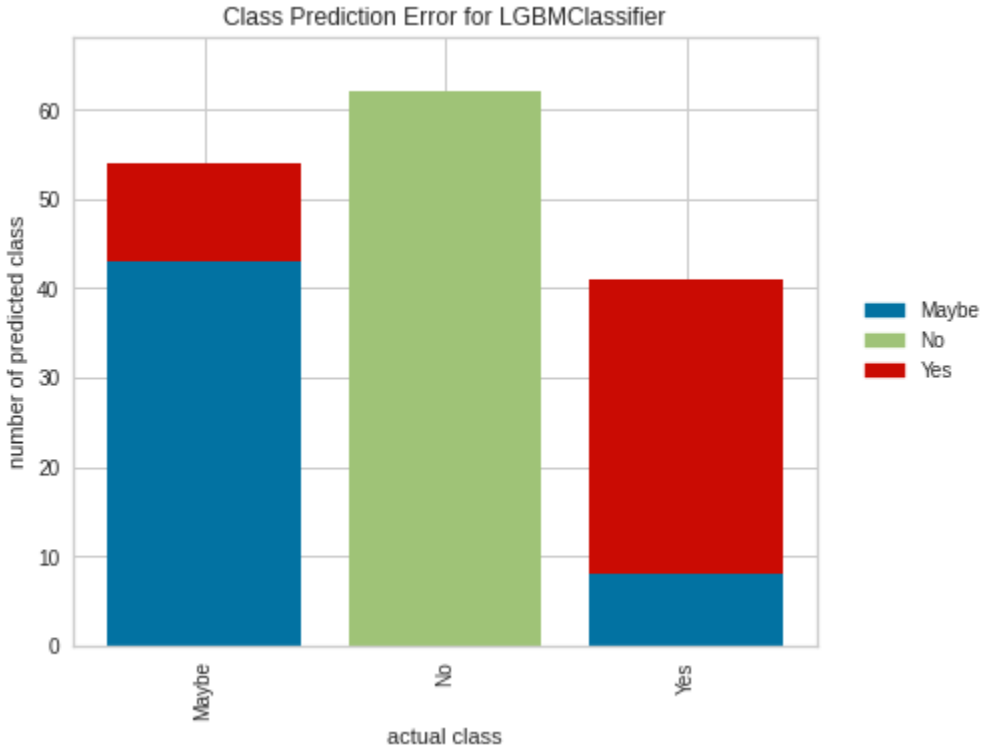
Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Light Gradient Boosting Machine	0.879	0.9527	0.8644	0.8791	0.8786	0.8164	0.8169

detect_fake_ video?	Label	Score
No	No	1
No	No	0.9992
Maybe	Maybe	0.8529
No	No	1
Yes	Yes	0.8042
...	...	...
No	No	0.9998
Maybe	Maybe	0.7901
No	No	0.9998
Maybe	Maybe	0.9626
No	No	0.9997

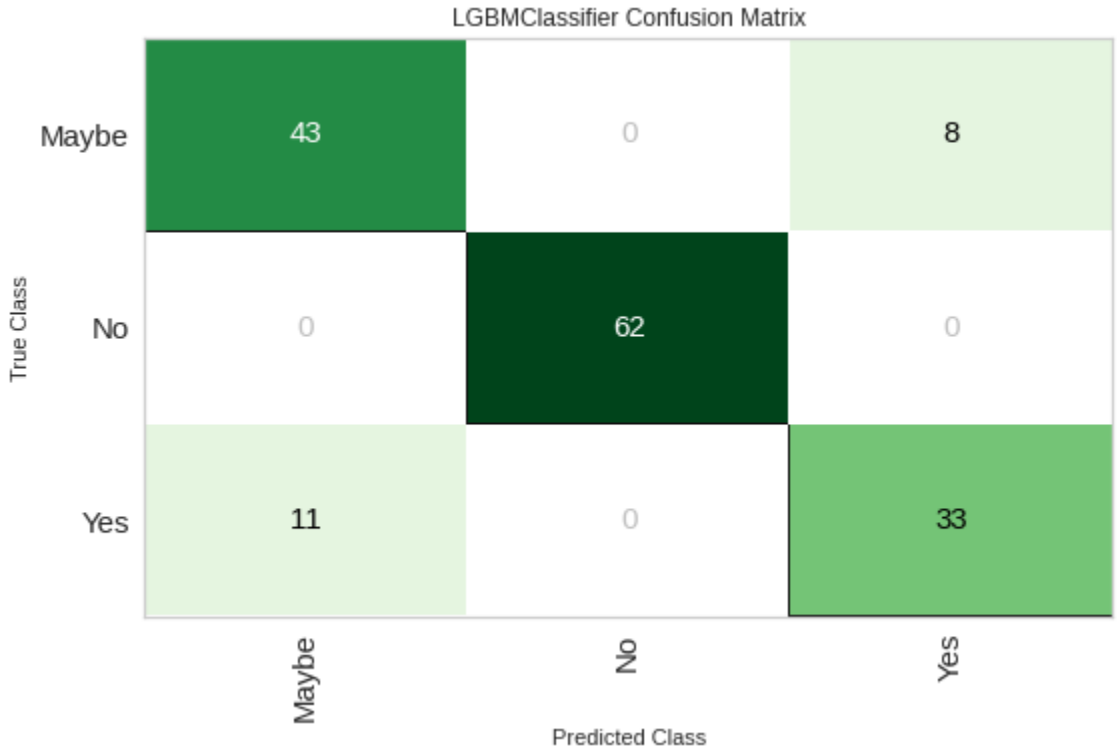
Important Features:



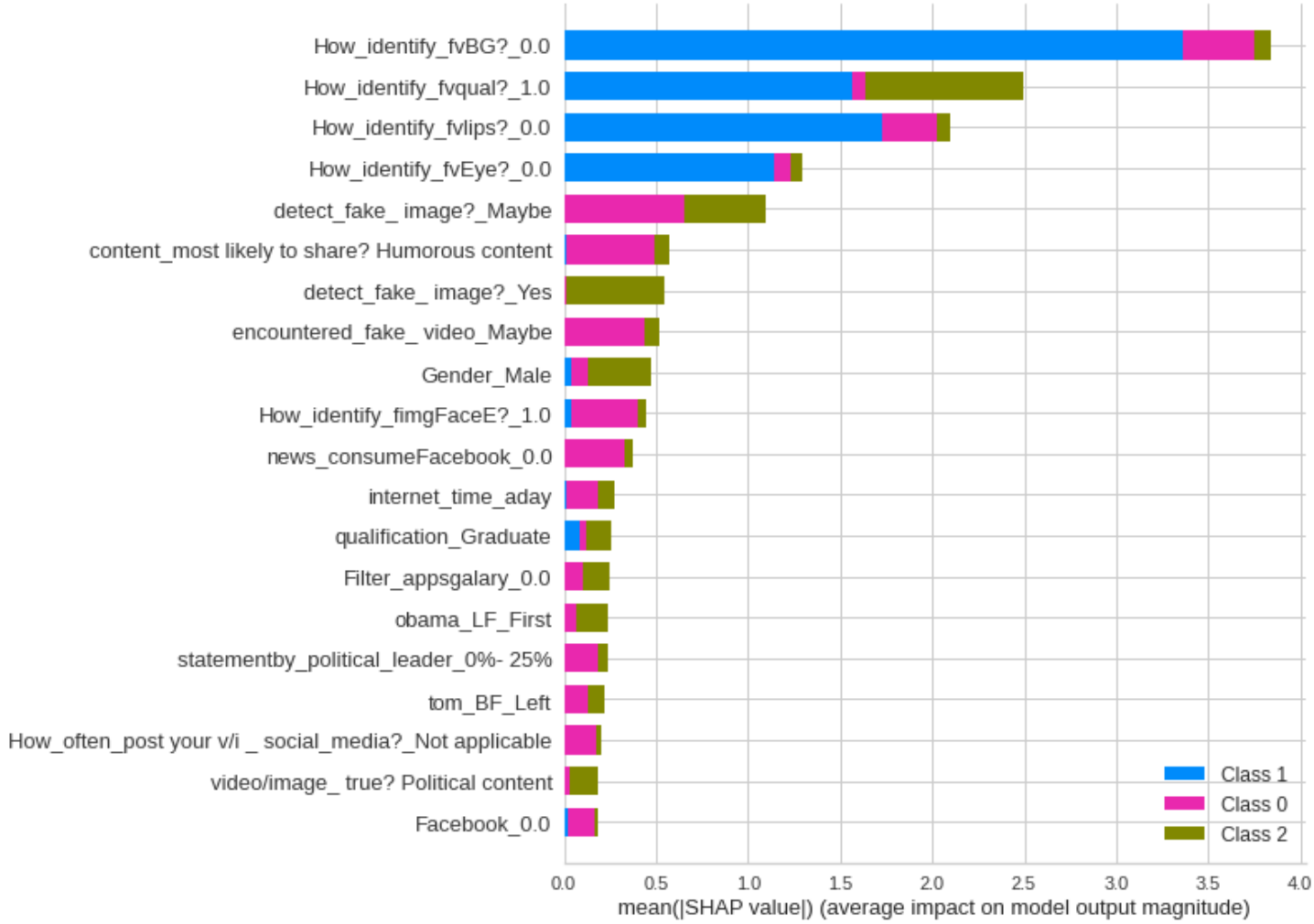
Prediction Error:



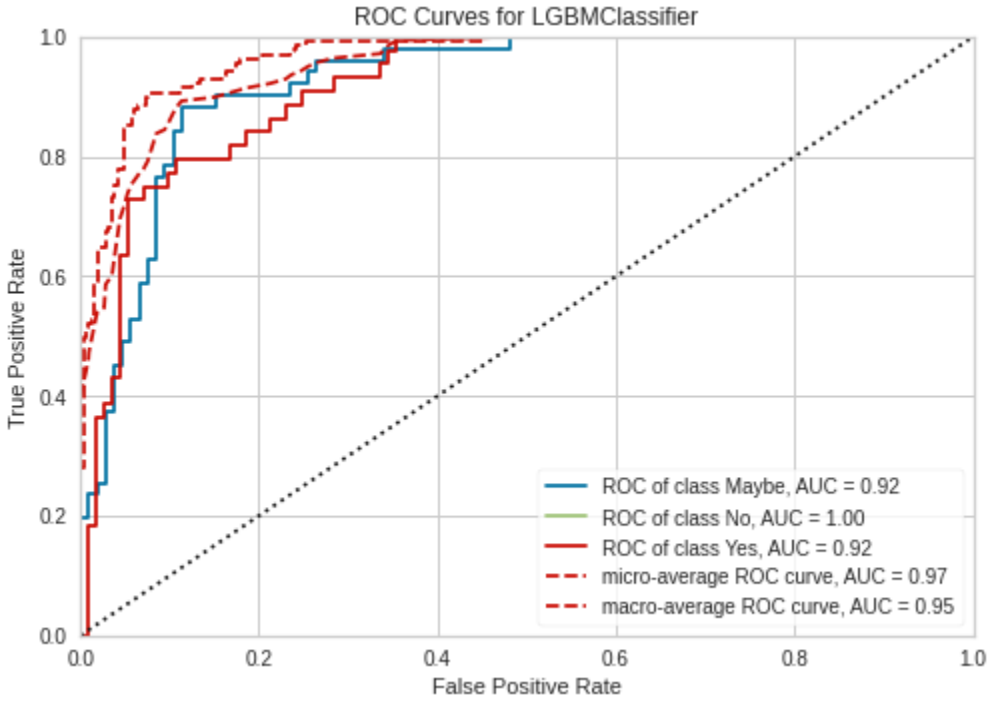
Confusion Matrix:



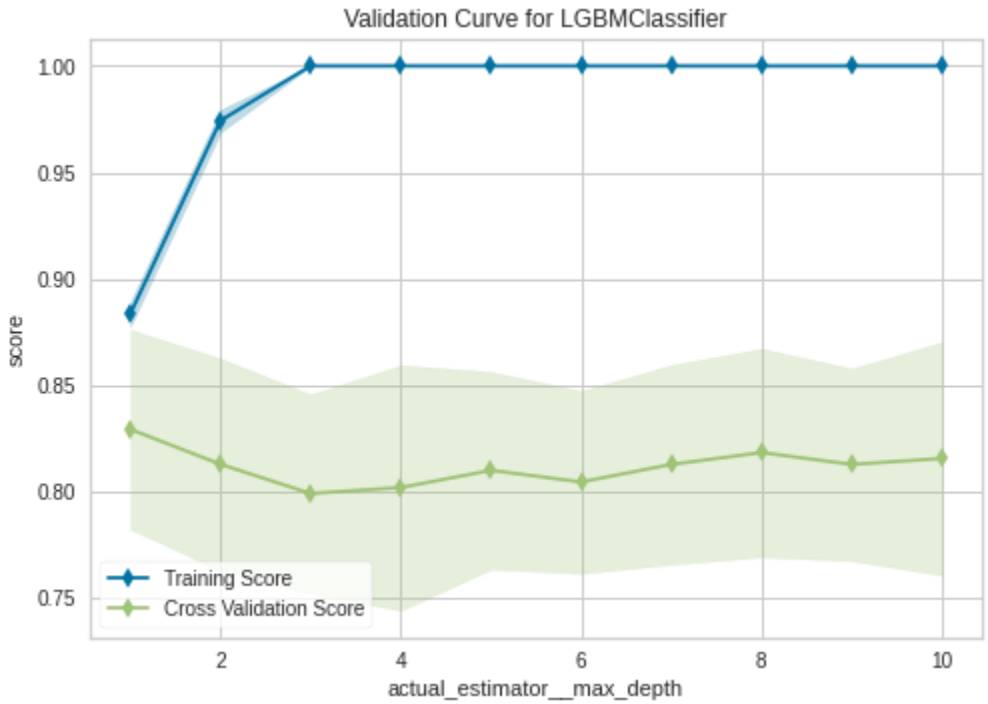
model interpretation:



Auc Curve:



validation Curve:



4.)Gradient Boosting Classifier

Gradient Boosting:

Introduction:

Gradient boosting algorithm is one of the most powerful algorithms in the field of machine learning. As we know that the errors in machine learning algorithms are broadly classified into two categories i.e. Bias Error and Variance Error. As gradient boosting is one of the boosting algorithms it is used to minimize bias error of the model.

Unlike, Adaboosting algorithm, the base estimator in the gradient boosting algorithm cannot be mentioned by us. The base estimator for the Gradient Boost algorithm is fixed and i.e. Decision Stump. Like, AdaBoost, we can tune the n_estimator of the gradient boosting algorithm. However, if we do not mention the value of n_estimator, the default value of n_estimator for this algorithm is 100.

Gradient boosting algorithm can be used for predicting not only continuous target variable (as a Regressor) but also categorical target variable (as a Classifier). When it is used as a regressor, the cost function is Mean Square Error (MSE) and when it is used as a classifier then the cost function is Log loss.

Refer this link for the python code.
https://www.analyticsvidhya.com/blog/2021/04/how-the-gradient-boosting-algorithm-works/

Applications:

i) Gradient Boosting Algorithm is generally used when we want to decrease the Bias error.

ii) Gradient Boosting Algorithm can be used in regression as well as classification problems. In regression problems, the cost function is MSE whereas, in classification problems, the cost function is Log-Loss.

How Gradient Boosting Works:
Gradient boosting involves three elements:

1. A loss function to be optimized.
2. A weak learner to make predictions.
3. An additive model to add weak learners to minimize the loss function.

Advantages of Gradient Boosting are:

• Often provides predictive accuracy that cannot be trumped.
• Lots of flexibility - can optimize on different loss functions and provides several hyper parameter tuning options that make the function fit very flexible.
• No data pre-processing required - often works great with categorical and numerical values as is.
• Handles missing data - imputation not required.

Disadvantages of Gradient Boosting are:

• Gradient Boosting Models will continue improving to minimize all errors. This can overemphasize outliers and cause overfitting.
• Computationally expensive - often require many trees (>1000) which can be time and memory exhaustive.
• The high flexibility results in many parameters that interact and influence heavily the behavior of the approach (number of iterations, tree depth, regularization parameters, etc.). This requires a large grid search during tuning.
• Less interpretative in nature, although this is easily addressed with various tools.

model;

```
GradientBoostingClassifier(ccp_alpha=0.0, criterion='friedman_mse', init=None,
                           learning_rate=0.1, loss='deviance', max_depth=3,
                           max_features=None, max_leaf_nodes=None,
                           min_impurity_decrease=0.0, min_impurity_split=None,
                           min_samples_leaf=1, min_samples_split=2,
                           min_weight_fraction_leaf=0.0, n_estimators=100,
                           n_iter_no_change=None, presort='deprecated',
                           random_state=786, subsample=1.0, tol=0.0001,
                           validation_fraction=0.1, verbose=0,
                           warm_start=False)
```

Model Evaluation:

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7027	0.9078	0.6768	0.7	0.6957	0.5503	0.5559
1	0.8649	0.9321	0.8561	0.8667	0.8649	0.7965	0.7974
2	0.8378	0.9332	0.8308	0.8604	0.8341	0.7566	0.7702

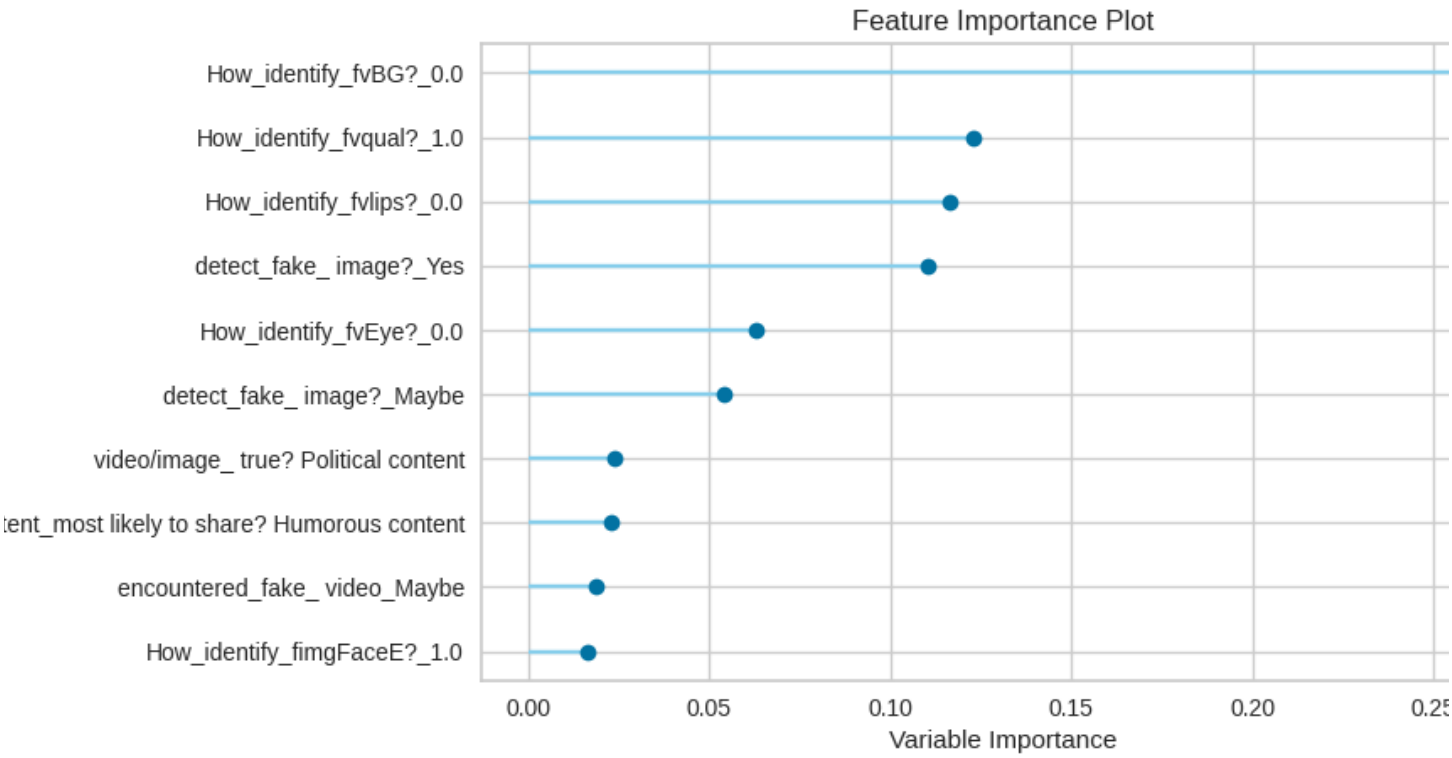
3	0.8333	0.9477	0.8091	0.8333	0.8333	0.7459	0.7459
4	0.8889	0.9847	0.8697	0.8948	0.8873	0.8302	0.8341
5	0.8056	0.9291	0.7758	0.8056	0.8047	0.7032	0.704
6	0.8611	0.9499	0.8364	0.873	0.8572	0.7875	0.796
7	0.75	0.9288	0.7278	0.7525	0.7505	0.6224	0.6231
8	0.7222	0.941	0.7	0.7278	0.7222	0.5814	0.5841
9	0.75	0.9176	0.7222	0.7483	0.7484	0.6206	0.6213
Mean	0.8017	0.9372	0.7805	0.8062	0.7998	0.6994	0.7032
Std	0.0624	0.0199	0.066	0.0658	0.0628	0.0938	0.0948

Prediction:

Model	**Accuracy**	**AUC**	**Recall**	**Prec.**	**F1**	**Kappa**	**MCC**
Gradient Boosting Classifier	0.828	0.9538	0.8079	0.8277	0.8274	0.7391	0.7395

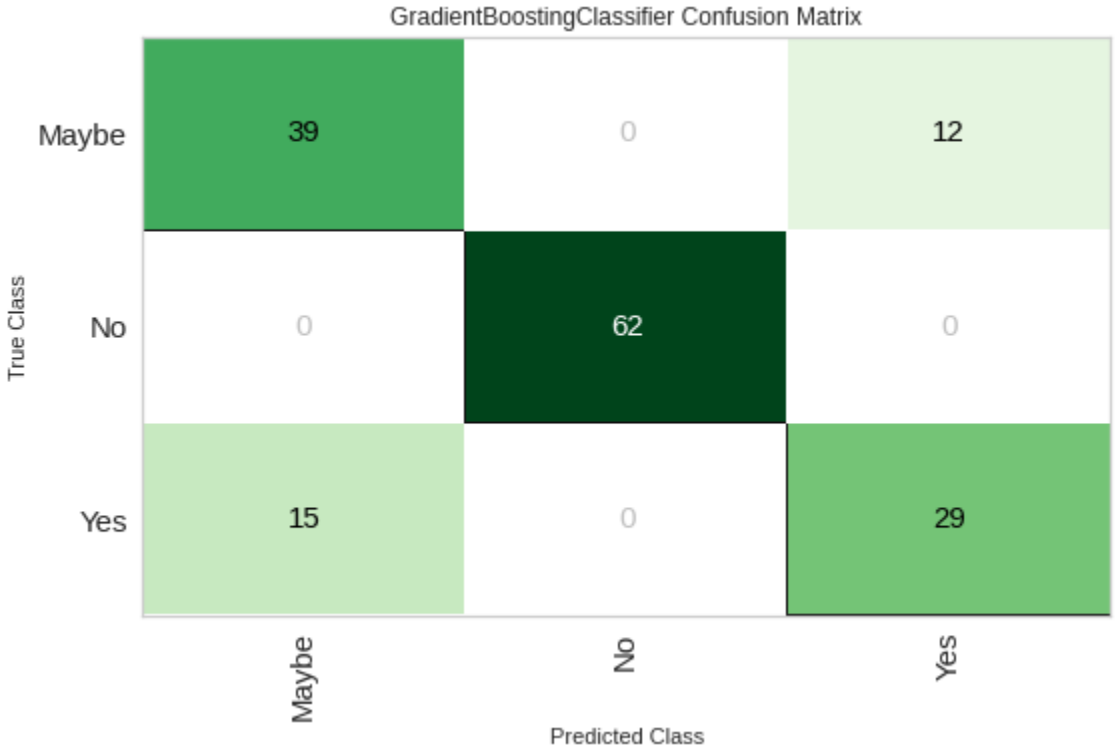
detect_fake_video?	**Label**	**Score**
No	No	0.9991
No	No	0.9977
Maybe	Maybe	0.8905
No	No	0.9991
Yes	Yes	0.5415
...	...	...
No	No	0.9988
Maybe	Maybe	0.6407
No	No	0.9986
Maybe	Maybe	0.8851
No	No	0.997

Important Features:

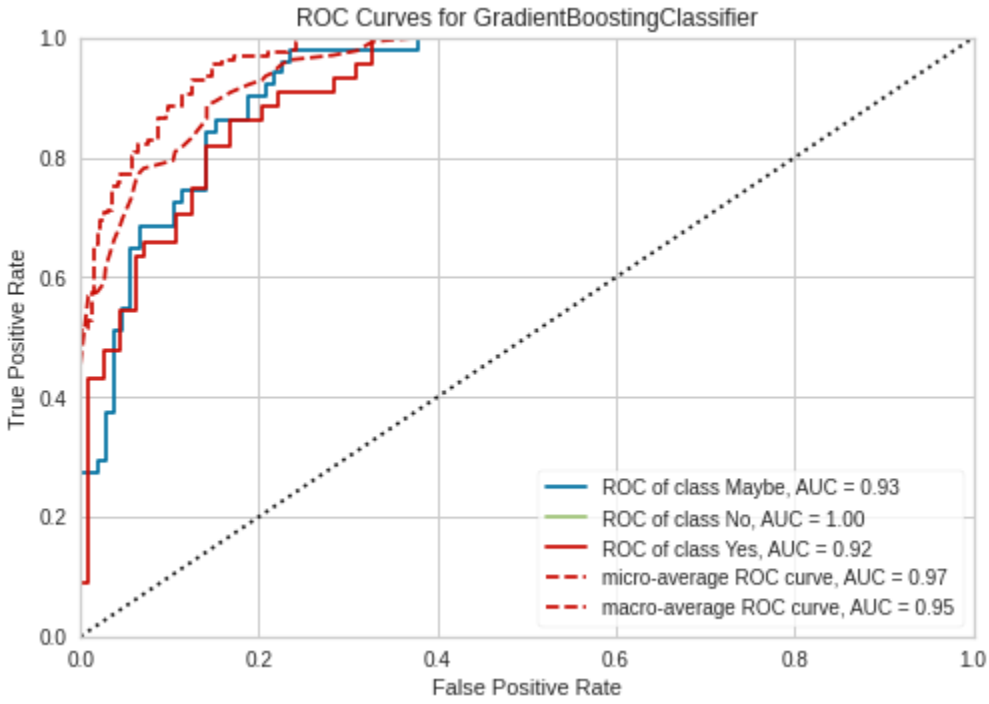


Prediction Error:

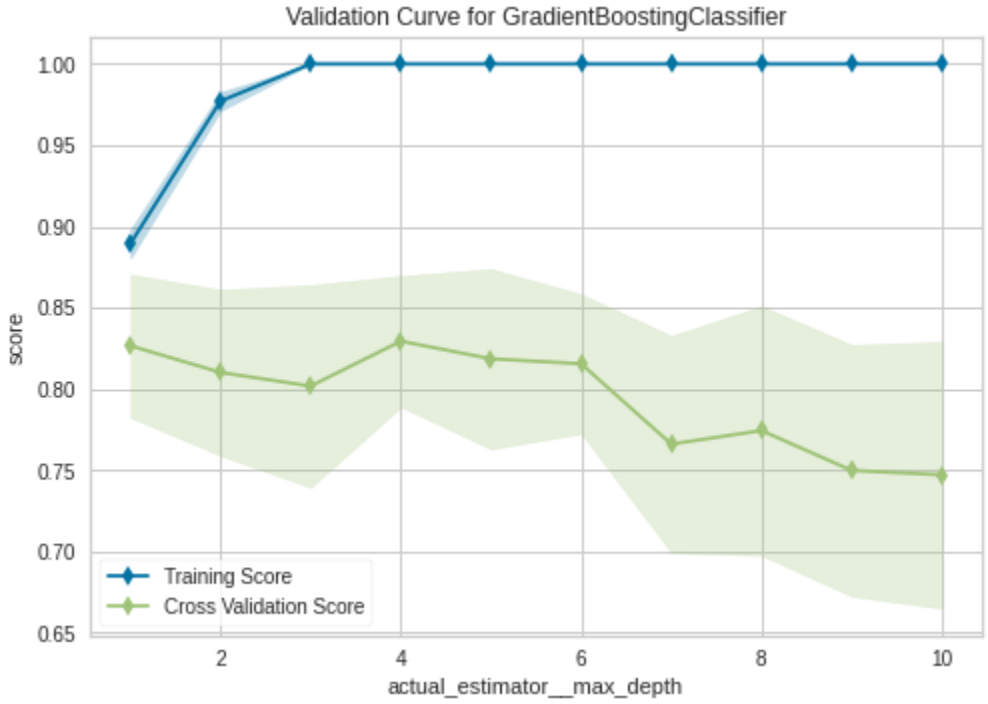
Confusion Matrix:



Auc Curve:



validation Curve:



5.)Decision Tree

Decision Tree algorithm belongs to the family of supervised learning algorithms (pre-defined target variable). Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems. It uses tree-like structures and their possible combinations to solve a particular problem.
The goal of using a Decision Tree is to create a training model that can be used to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).

Types of Decision tree:
Types of decision trees are based on the type of target variable we have. It can be of two types:
1. Categorical Variable Decision Tree: Decision Tree which has a categorical target variable then it is called a Categorical variable decision tree.
2. Continuous Variable Decision Tree: Decision Tree has a continuous target variable then it is called Continuous Variable Decision Tree.

Important Terminologyrelated to Decision Trees:
1. Root Node: It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
2. Splitting: It is a process of dividing a node into two or more sub-nodes.
3. Decision Node: When a sub-node splits into further sub-nodes, then it is called the decision node.
4. Leaf / Terminal Node: Nodes that do not split are called Leaf or Terminal nodes.
5. Pruning: When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
6. Branch / Sub-Tree: A subsection of the entire tree is called branch or sub-tree.

7. Parent and Child Node: A node,which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

Assumptions:
1.At the beginning, the whole training set is considered as the root.
2.Feature values need to be categorical. If the values are continuous then they are discretized prior to building the model.
3.Records are distributed recursively on the basis of attribute values.
4.Order to place attributes as root or internal node of the tree is done by using some statistical approach.

model;

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=786, splitter='best')
```

Model Evaluation:

Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.6486	0.7446	0.6212	0.647	0.6473	0.4697	0.4702
1	0.7838	0.8438	0.7702	0.7884	0.783	0.6747	0.6777
2	0.7568	0.8238	0.7399	0.7582	0.7568	0.6337	0.6344
3	0.8611	0.9023	0.8424	0.8631	0.8611	0.7885	0.7894
4	0.8333	0.8815	0.8061	0.8365	0.831	0.7453	0.7488
5	0.7778	0.8431	0.7455	0.7778	0.7778	0.6612	0.6612
6	0.6944	0.7846	0.6515	0.6957	0.6944	0.5347	0.5353
7	0.6944	0.7821	0.6722	0.7023	0.6925	0.5406	0.5463
8	0.7778	0.8397	0.7556	0.7778	0.7778	0.6636	0.6636
9	0.7222	0.7981	0.6889	0.7183	0.7175	0.5775	0.5802
Mean	0.755	0.8244	0.7293	0.7565	0.7539	0.6289	0.6307
Std	0.0624	0.0455	0.0664	0.0631	0.0628	0.0938	0.0938

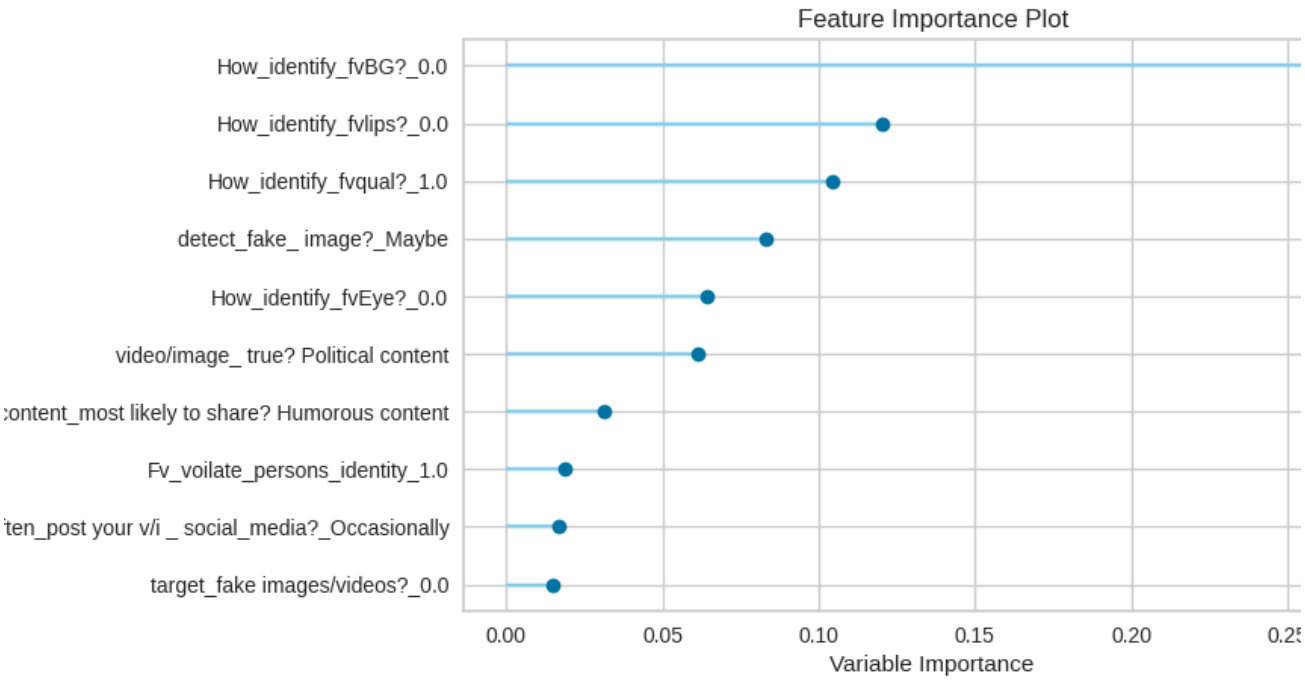
Prediction:

Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC

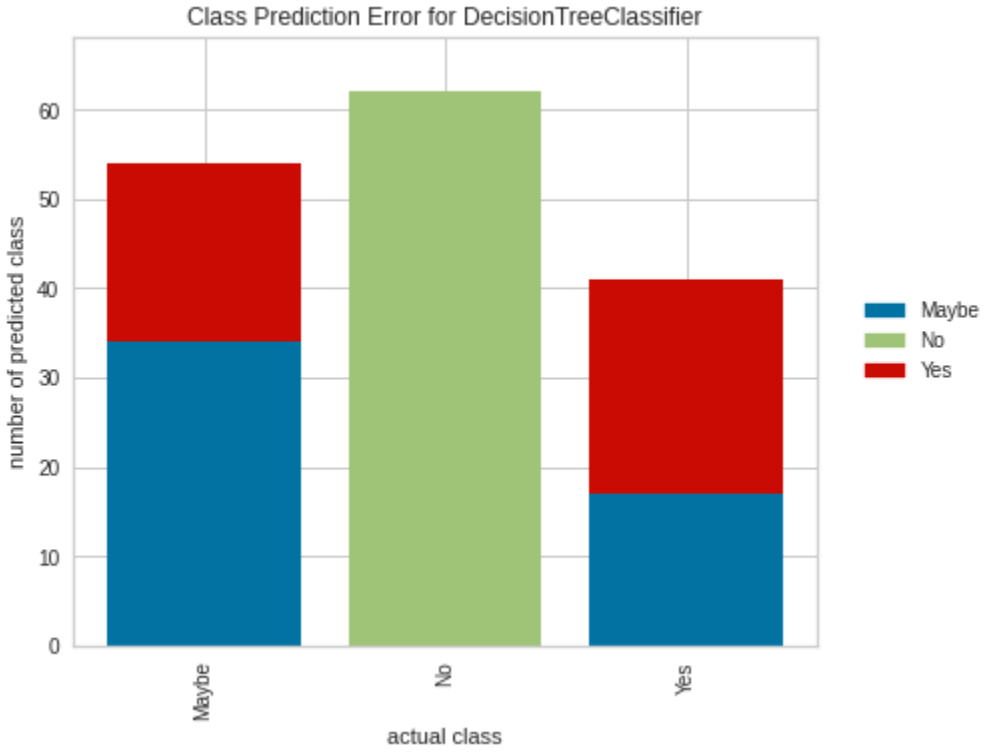
Decision Tree Classifier	0.7643	0.8304	0.7374	0.7635	0.7635	0.6425	0.6428

detect_fake_ video?	Label	Score
No	No	1
No	No	1
Maybe	Maybe	1
No	No	1
Yes	Maybe	1
...	...	...
No	No	1
Maybe	Maybe	1
No	No	1
Maybe	Maybe	1
No	No	1

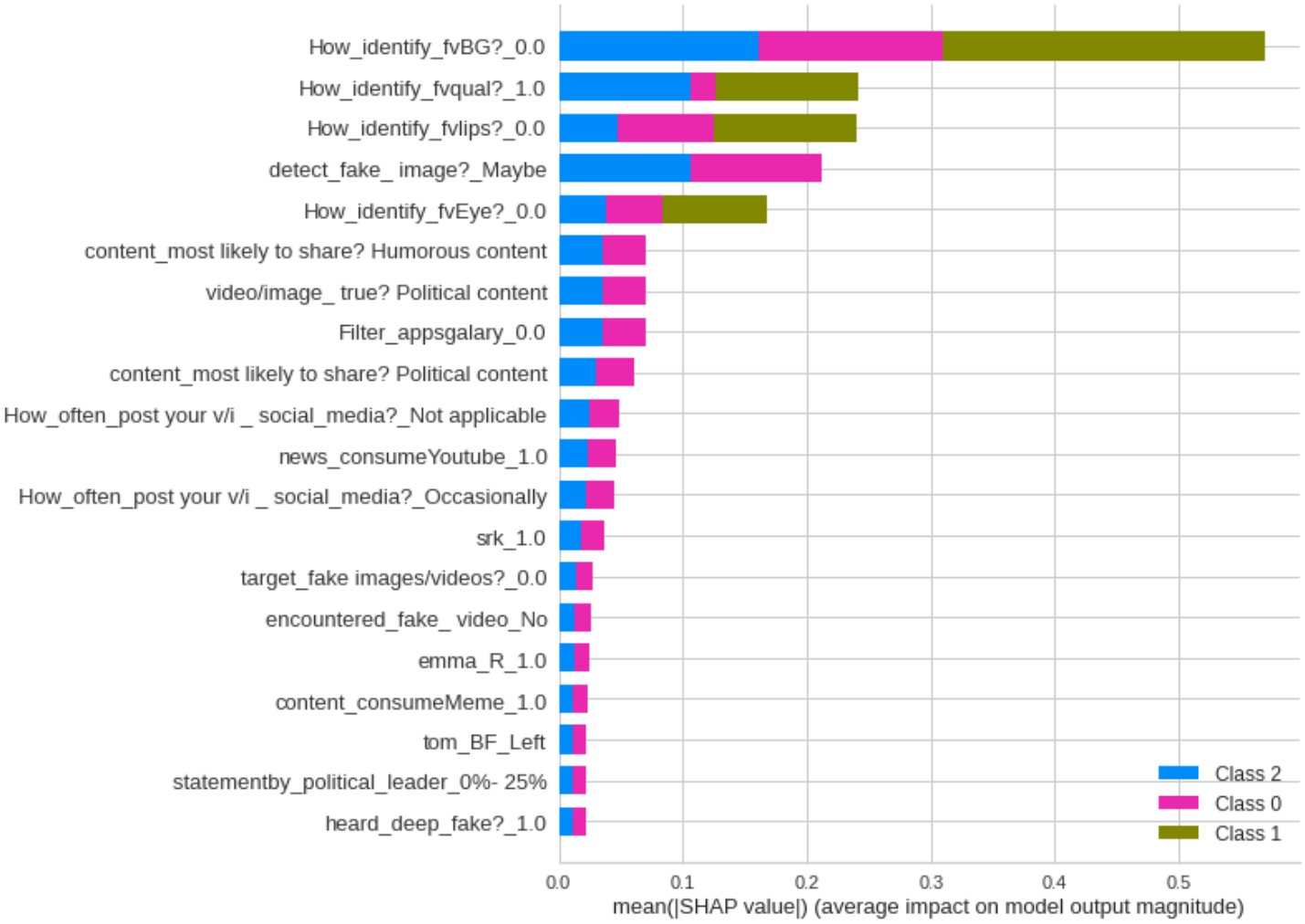
Important Features:



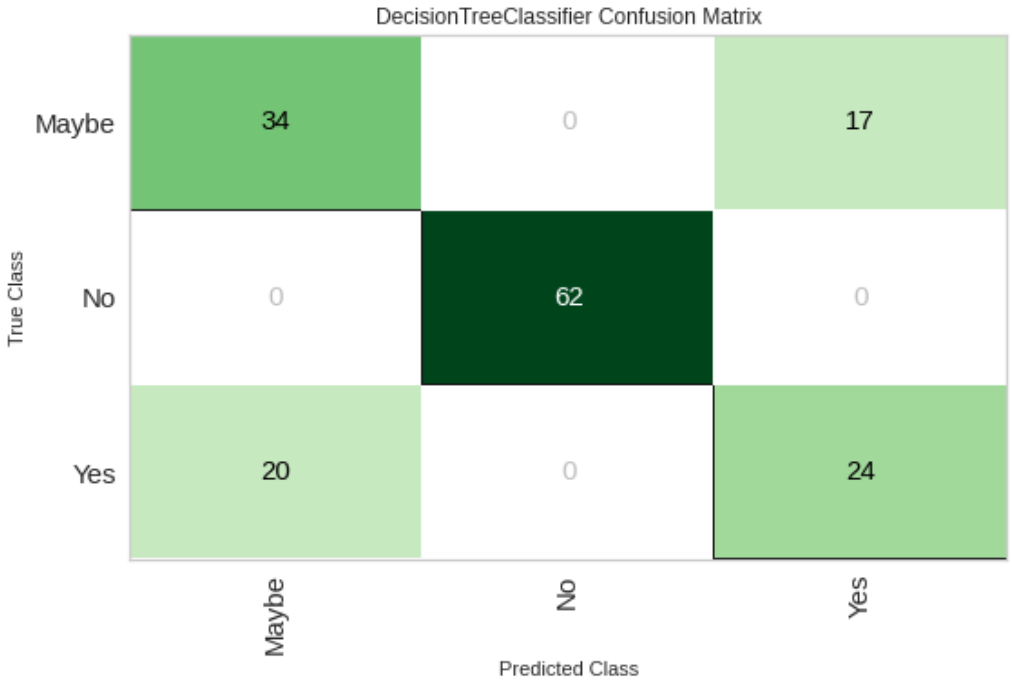
Prediction Error:



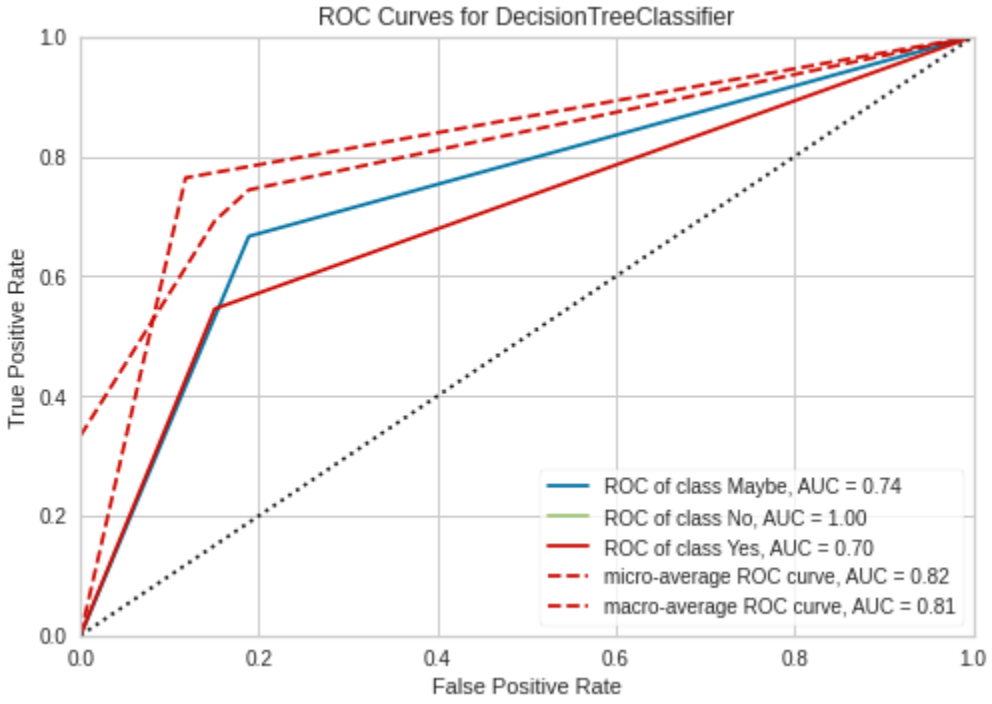
model interpretation:



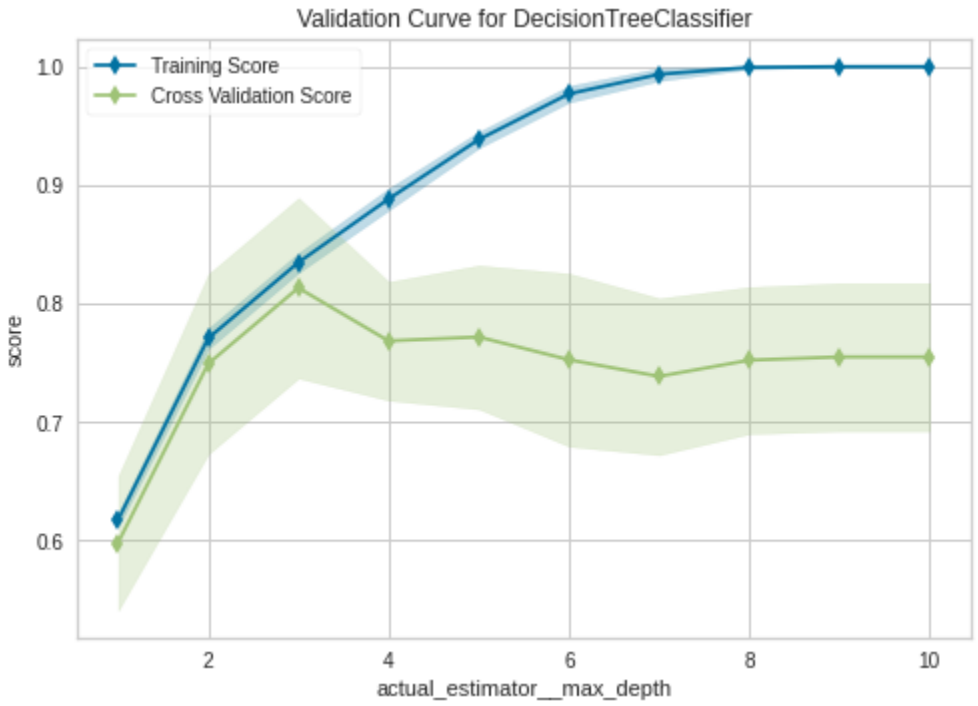
Confusion Matrix:



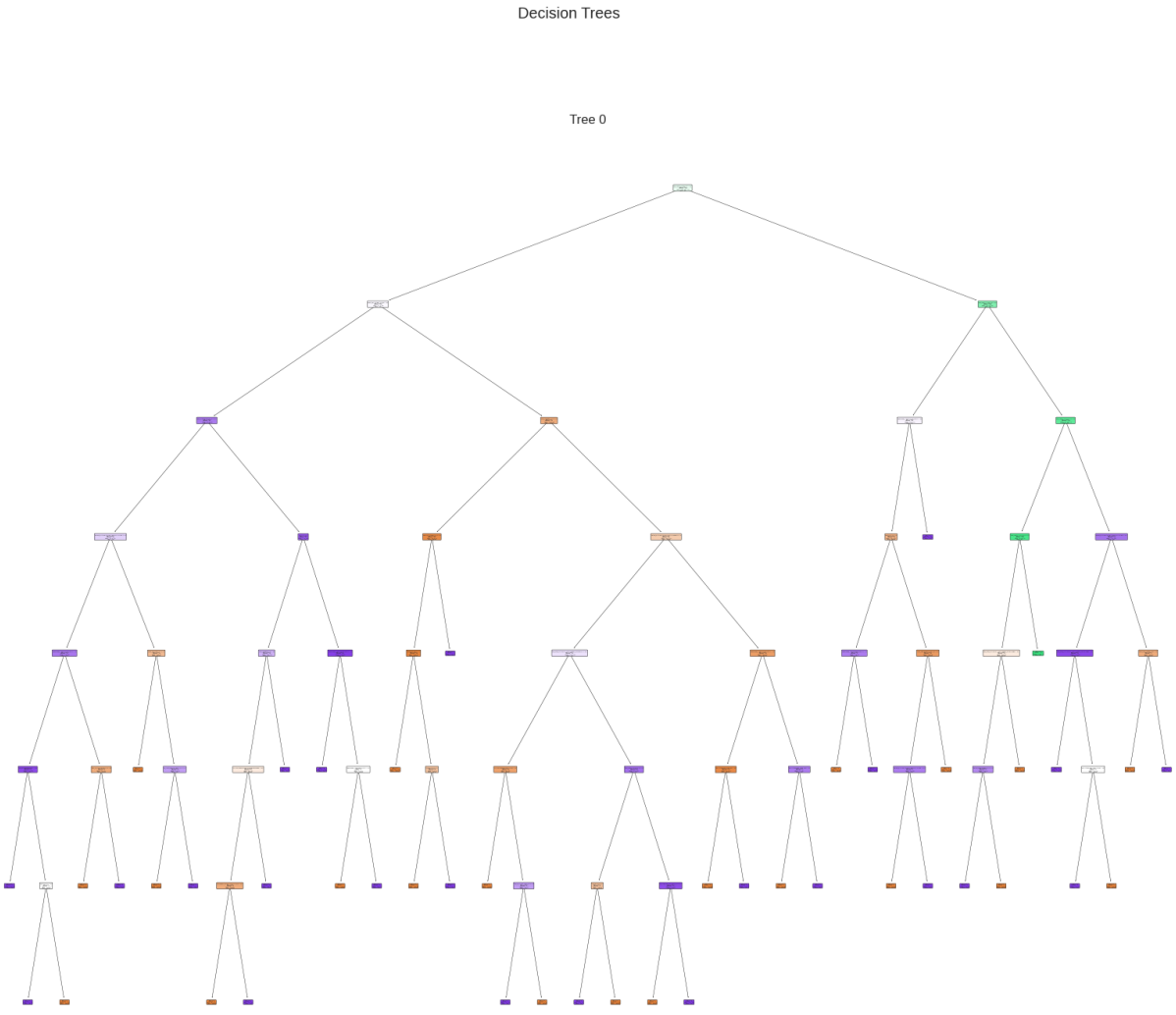
Auc Curve:



validation Curve:



Decision Tree:



6.)Logistic Regression

In statistics, the (binary) logistic model (or logit model) is a statistical model that models the probability of one event (out of two alternatives) taking place by having the log-odds (the logarithm of the odds) for the event be a linear combination of one or more independent variables ("predictors"). In regression analysis, logistic regression[1] (or logit regression) is estimating the parameters of a logistic model (the coefficients in the linear combination). Formally, in binary logistic regression, there is a single binary dependent variable, coded by an indicator variable, where the two values are labeled "0" and "1", while the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling;[2] the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names. See § Background and § Definition for formal mathematics, and § Example for a worked example.

Example graph of a logistic regression curve fitted to data. The curve shows the probability of passing an exam (binary dependent variable) versus hours of studying (scalar independent variable). See § Example for worked details.
Binary variables are widely used in statistics to model the probability of a certain class or event taking place, such as the probability of a team winning, of a patient being healthy, etc. (see § Applications), and the logistic model has been the most commonly used model for binary regression since about 1970.[3] Binary variables can be generalized to categorical variables when there are more than two possible values (e.g. whether an image is of a cat, dog, lion, etc.), and the binary logistic regression is generalized to multinomial logistic regression. If the multiple categories are ordered, one can use the ordinal logistic regression (for example the proportional odds ordinal logistic model[4]). See § Extensions for further extensions. The logistic regression model itself simply models the probability of output in terms of input and does not perform statistical classification (it is not a classifier), though it can be used to make a classifier, for instance by choosing a cutoff value and classifying inputs with probability greater than the cutoff as one class, below the cutoff as the other; this is a common way to make a binary classifier.
Analogous linear models for binary variables with a different sigmoid function instead of the logistic function (to convert the linear combination to a probability) can also be used, most notably the probit model; see § Alternatives. The defining characteristic of the logistic model is that increasing one of the independent variables multiplicatively scales the odds of the given outcome at a constant rate, with each independent variable having its own parameter; for a binary dependent variable, this generalizes the odds ratio. More abstractly, the logistic function is the natural parameter for the Bernoulli distribution, and in this sense is the "simplest" way to convert a real number to a probability. In particular, it maximizes entropy (minimizes added information), and in this sense makes the fewest assumptions of the data being modeled; see § Maximum entropy.
The parameters of a logistic regression are most commonly estimated by maximum-likelihood estimation (MLE). This does not have a closed-form expression, unlike linear least squares; see § Model fitting. Logistic regression by MLE plays a similarly basic role for binary or categorical responses as linear regression by ordinary least squares (OLS) plays for scalar responses: it is a simple, well-analyzed baseline model; see § Comparison with linear regression for discussion. The logistic regression as a general statistical model was originally developed and popularized primarily by Joseph Berkson,[5] beginning in Berkson (1944), where he coined "logit"; see § History.

model;

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=1000,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=786, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

Model Evaluation:

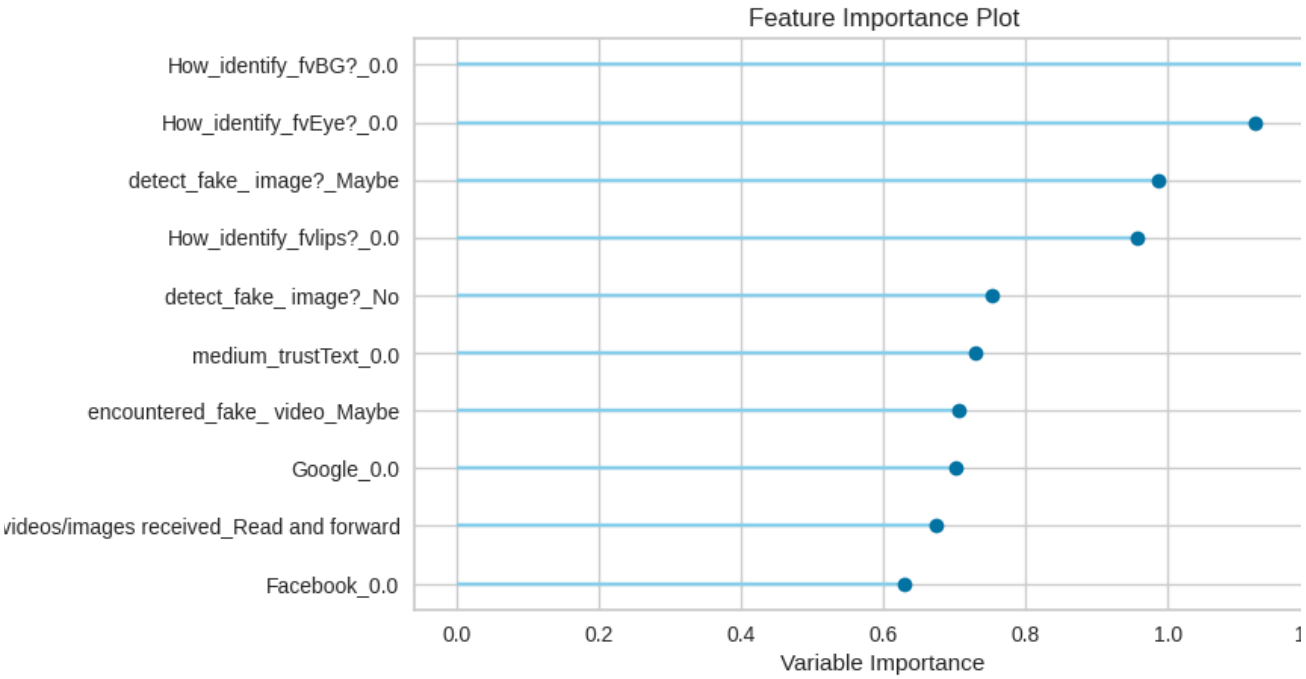
Fold	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
0	0.7568	0.8744	0.7348	0.7588	0.7511	0.632	0.6385
1	0.7838	0.902	0.7781	0.8011	0.7896	0.6758	0.678
2	0.7838	0.8564	0.7767	0.809	0.7864	0.6761	0.6859
3	0.6667	0.8844	0.6374	0.6667	0.6667	0.4918	0.4918
4	0.7778	0.9437	0.7505	0.7745	0.7689	0.6584	0.664
5	0.8056	0.8754	0.7727	0.8185	0.7968	0.7007	0.7119
6	0.6944	0.8951	0.6626	0.6975	0.6932	0.5352	0.5377
7	0.7778	0.888	0.7706	0.8069	0.7819	0.6674	0.6776
8	0.6389	0.8779	0.6246	0.6494	0.6421	0.4571	0.4587
9	0.75	0.8685	0.7373	0.763	0.7542	0.6241	0.6263
Mean	0.7435	0.8866	0.7245	0.7545	0.7431	0.6119	0.617
Std	0.0538	0.0227	0.0569	0.0587	0.0527	0.0814	0.0842

Prediction:

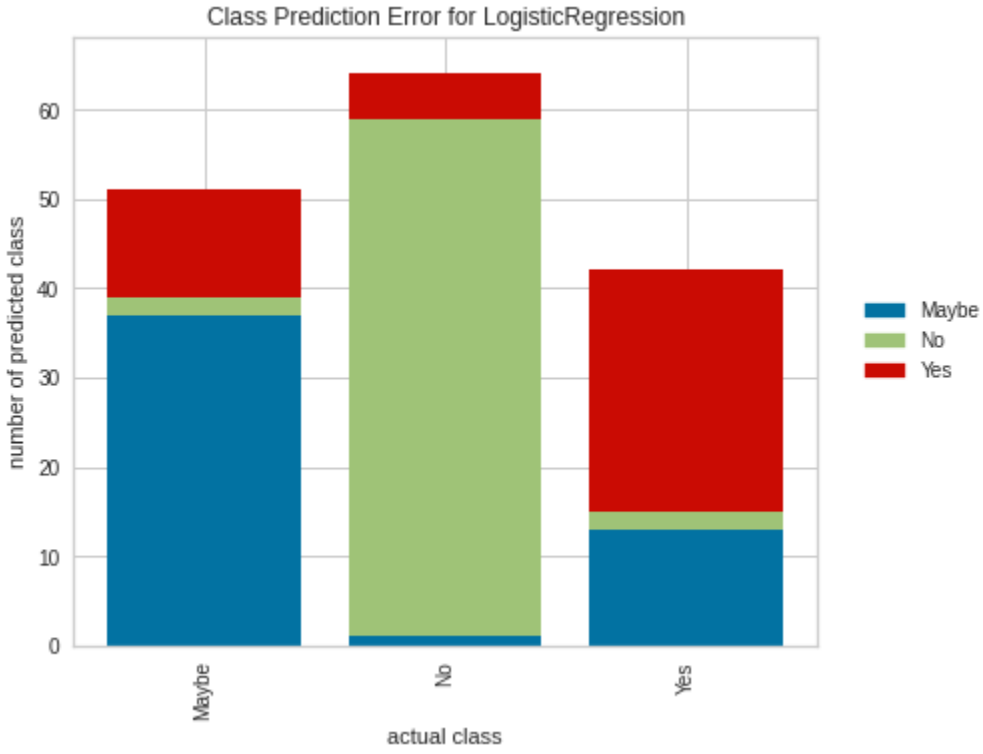
Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Logistic Regression	0.7771	0.9126	0.7582	0.7737	0.7752	0.6615	0.6616

detect_fake_video?	Label	Score
No	No	0.9963
No	No	0.9824
Maybe	Yes	0.8344
No	No	0.9963
Yes	Yes	0.7759
...	...	...
No	No	0.9388
Maybe	Maybe	0.6924
No	No	0.9994
Maybe	Maybe	0.5159
No	No	0.9319

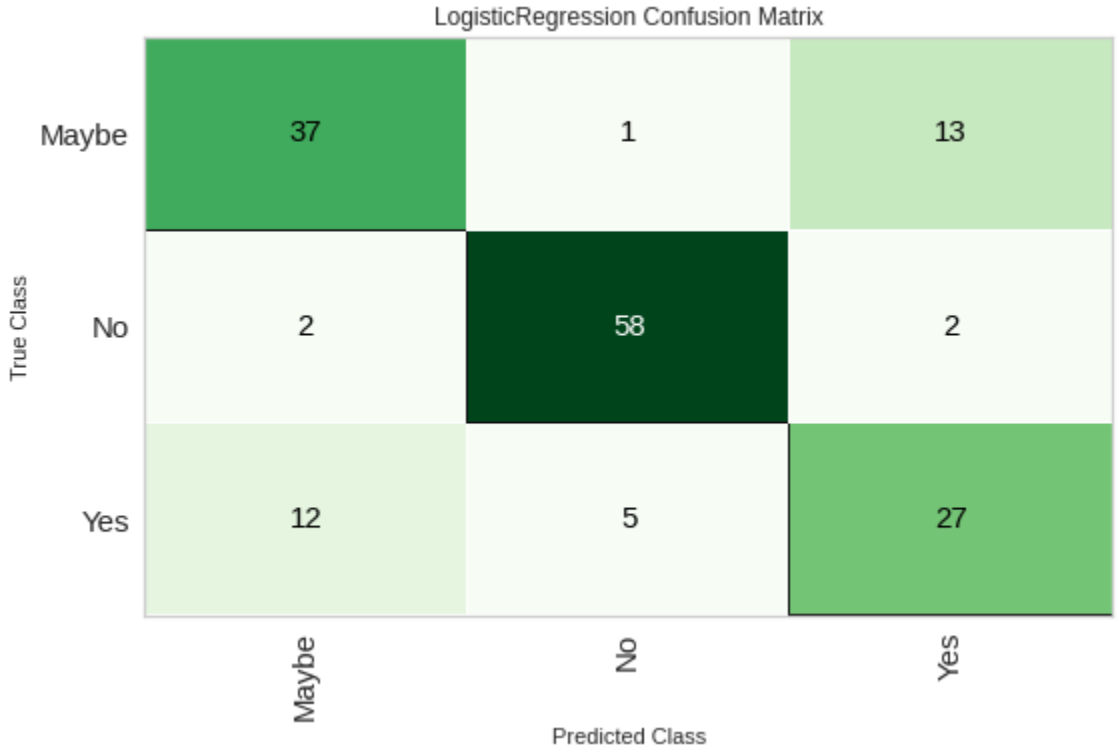
Important Features:



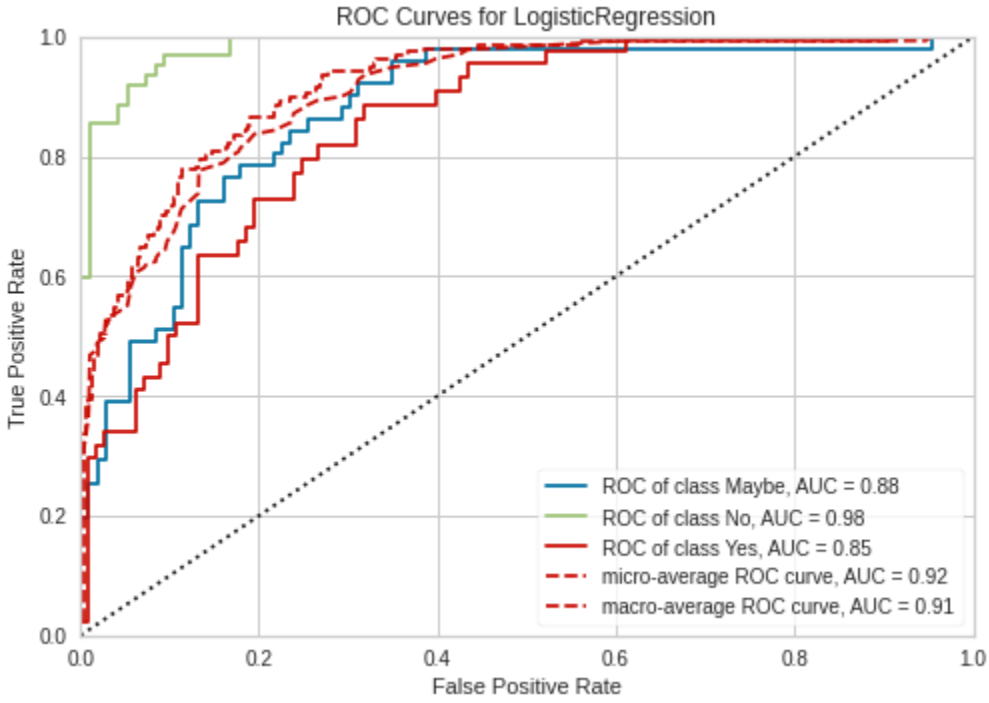
Prediction Error:



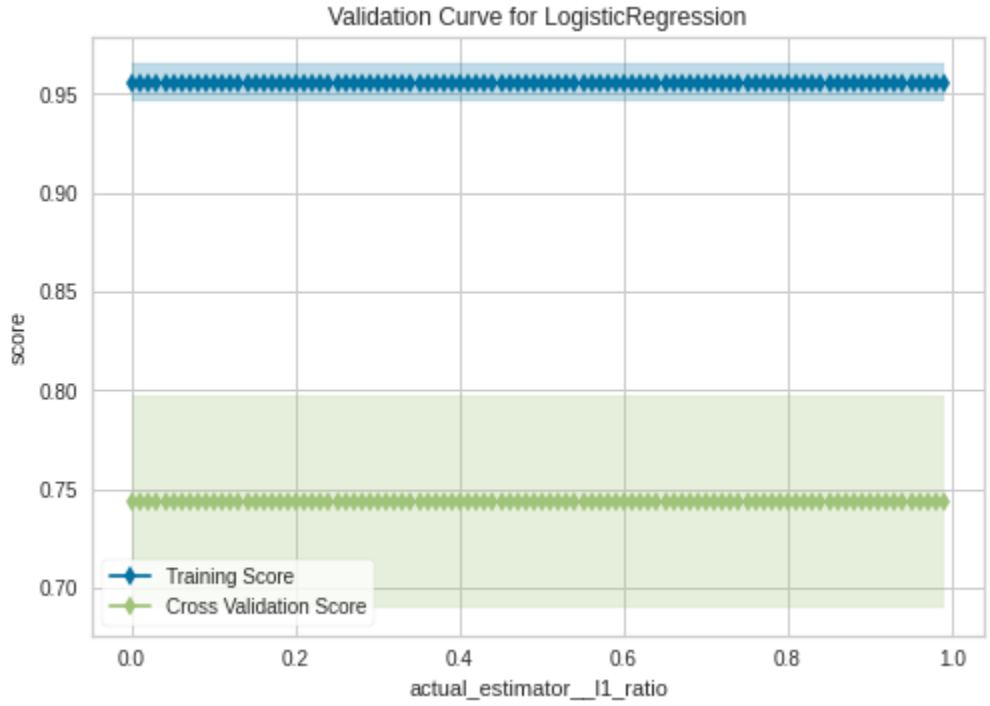
Confusion Matrix:



Auc Curve:



validation Curve:



7.)Best Fit Model

Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
Random Forest Classifier	0.8153	0.9423	0.7984	0.817	0.8126	0.7194	0.7227
Extra Trees Classifier	0.8126	0.9316	0.7958	0.8152	0.81	0.7156	0.719
Light Gradient Boosting Machine	0.8101	0.9259	0.7902	0.8156	0.8085	0.7123	0.7161
Gradient Boosting Classifier	0.8017	0.9372	0.7805	0.8062	0.7998	0.6994	0.7032
Ada Boost Classifier	0.7657	0.913	0.7444	0.775	0.7623	0.646	0.6534
Decision Tree Classifier	0.755	0.8244	0.7293	0.7565	0.7539	0.6289	0.6307
Linear Discriminant Analysis	0.7491	0.8996	0.7279	0.7554	0.7451	0.6186	0.6243
Logistic Regression	0.7435	0.8866	0.7245	0.7545	0.7431	0.6119	0.617
Ridge Classifier	0.7327	0	0.7102	0.7407	0.729	0.5937	0.5997
Naive Bayes	0.7216	0.918	0.7085	0.764	0.6834	0.584	0.6372
SVM - Linear Kernel	0.7215	0	0.7007	0.7413	0.7126	0.5787	0.5945
K Neighbors Classifier	0.6114	0.7903	0.5892	0.6066	0.5981	0.4058	0.4125

Get codes for all models

:https://colab.research.google.com/drive/1FgRpiBR27U992wcC_O4KPcNDvZUGYcs_?usp=sharing

Chapter 2

Creating Deep Fake Video

There are various software and apps that can be used to create Deep Fake videos.

For Example:

1-FaceApp ·
2- Zao ·
3- Reface ·
4- SpeakPic ·
5- DeepFace Lab ·
6- Fake App

For the purpose of our project we used DeepFaceLab.

Deep face lab is an open-source project on GitHub. In recent years, it's been possible to use this software for the creation of deep fakes. The quality of the deep fake that this software creates depends partly on the data available and mostly on the skill of the user.

The process of creating a deep fake is divided into 5 steps. Before going to that 2 things we need to understand

Data source:

In computer programming, source data or data source is the primary location from where data comes. The data source is a database, a dataset, a spreadsheet, or even hard-coded data. When data is displayed, it is retrieved from its data source.

Data destination:

Alternatively referred to as target, a destination is a location where data is sent. For example, when you download something from the internet the website is the source and the computer is your destination.

Step 1: Convert the videos into images.

A- extracting all frames from the video data source.
We used a video of our colleague Zeenat as a data source. The clip was of 18 sec where she was explaining what's deep fake.
you can check out that video:https://youtu.be/c5uJ1P1fhlc
(file used:2) extract images from video data_src)

from that 18 sec, we got a total of 453 frames

	
video length:18 sec	**Frames: 453 images**

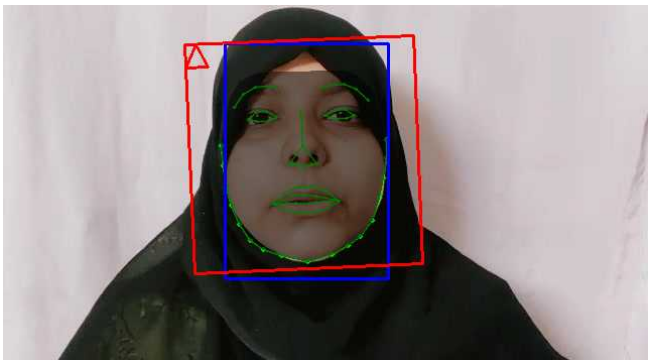
B- extracting all frames from video Data destinations.
We used a video of our colleague's Gazala as a data destination. the clip was of 20 sec where she was explaining what is deep fake.
you can check out that video:https://youtu.be/UHX127ex2sc
(file used :3) extract images from video data_dst FULL FPS)

	
video length:20 sec	**Frames: 627 images**

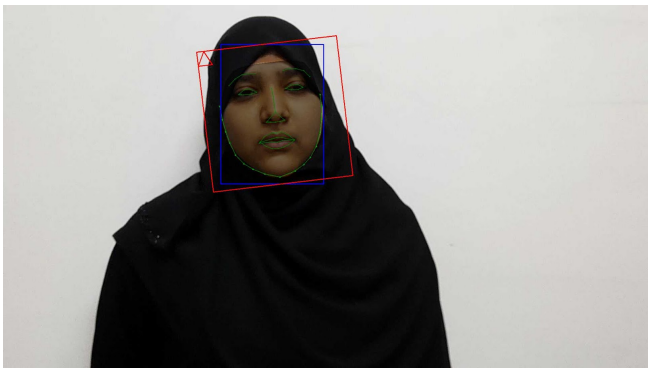
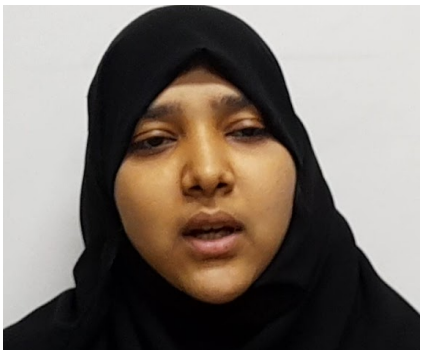
Step 2: Extracting faces from images:

A- extracting faces from frames of Data source:

from all 453 images which were extracted from data source video now we extract all the faces
(file used :4) data_src faceset extract)

	
Frames: 453 images	**Faces: 453**

B- extracting faces from frames of Data destination.
from all 627 images which were extracted from data destination video now we extract all the faces
(file used :5) data_dst faceset extract)

	
Frames: 627 images	**Faces: 627**

Step 3: Model building:

There are different models present in the software. eg:

- AMP SRC-SRC
- AMP
- Quick 96
- SAEHD

For our project we used a Model. SAEHD

We click on (file:6) train SAEHD) and start the training of our model. Make sure to have a minimum of 100000 iterations. The more, the better. If you wish, you can do 500000 or 1000000 iterations. But the more iters, the better the result! When you say “OK, it’s enough,” press the “Enter” key and complete the training session.

<table>
<tr>
<td>

```
================ Model Summary ================
==                                            ==
==            Model name: zeenat saehd_SAEHD  ==
==                                            ==
==     Current iteration: 1994                ==
==                                            ==
==---------------- Model Options ---------------==
==                                            ==
==              resolution: 128               ==
==               face_type: f                 ==
==     models_opt_on_gpu: True                ==
==                   archi: liae-ud           ==
==                 ae_dims: 256               ==
==                  e_dims: 64                ==
==                  d_dims: 64                ==
==             d_mask_dims: 22                ==
==       masked_training: True                ==
==       eyes_mouth_prio: True                ==
==           uniform_yaw: False               ==
==         blur_out_mask: False               ==
==             adabelief: True                ==
==            lr_dropout: n                   ==
==           random_warp: True                ==
==      random_hsv_power: 0.0                 ==
==       true_face_power: 0.0                 ==
==      face_style_power: 0.0                 ==
==        bg_style_power: 0.0                 ==
==               ct_mode: none                ==
==             clipgrad: False                ==
==             pretrain: False                ==
==        autobackup_hour: 0                  ==
== write_preview_history: True                ==
==           target_iter: 0                   ==
==       random_src_flip: False               ==
==       random_dst_flip: True                ==
==            batch_size: 4                   ==
==             gan_power: 0.0                 ==
==        gan_patch_size: 16                  ==
==              gan_dims: 16                  ==
==                                            ==
==---------------- Running On ----------------==
==                                            ==
==          Using device: CPU                 ==
==                                            ==
================================================
```

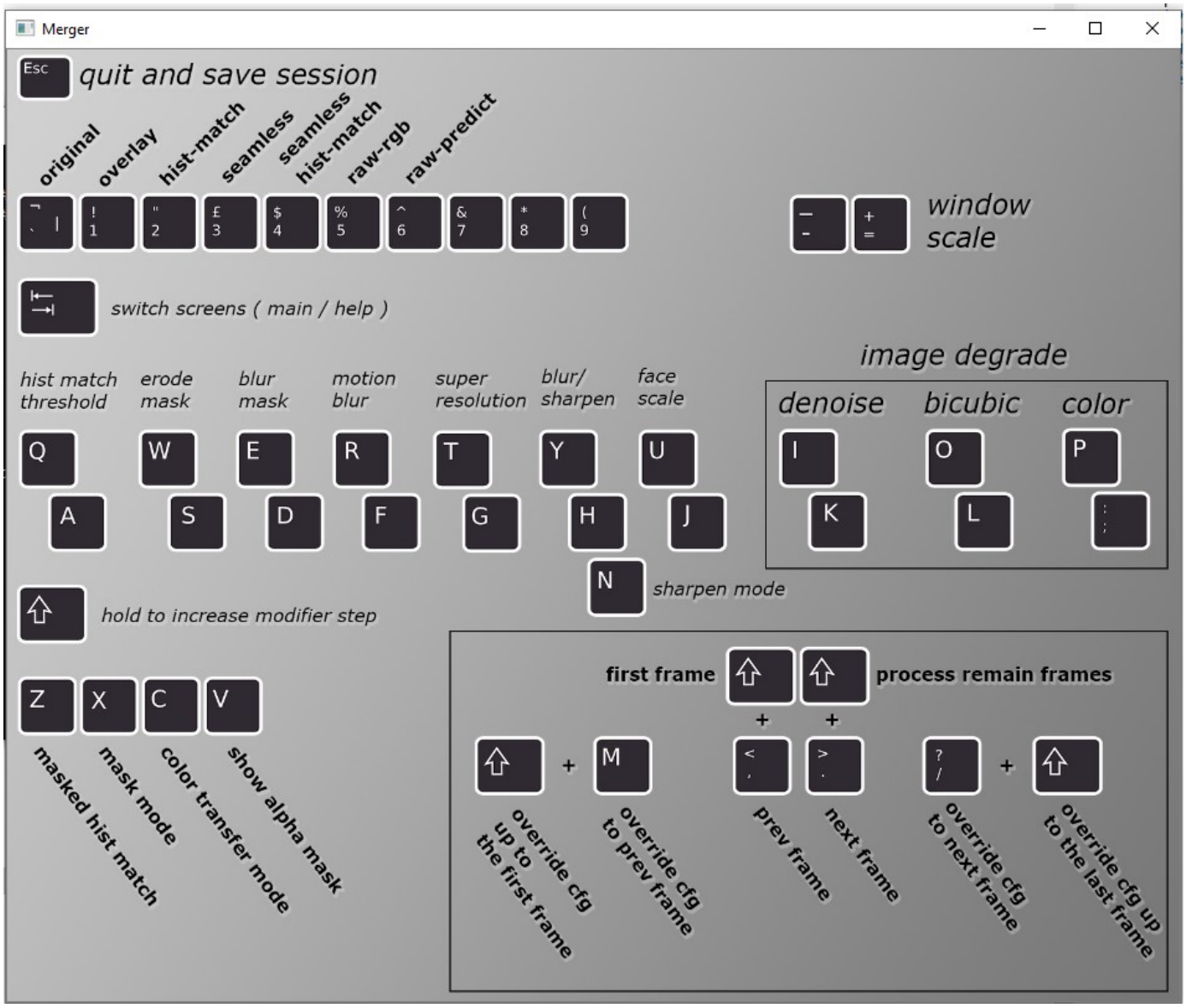
</td>
<td>

</td>
</tr>
<tr>
<td>Model Summary</td>
<td>Model getting Trained</td>
</tr>
</table>

Step 4: Merging and adjusting:

In this step we can do the desired changes in colour, sizes ,shape ,blurriness, brightness of the deep fake image.

The chart for changes are given below



Q:key use to increase hist match threshold
A:key use to decrease hist match threshold

W:key use to increase encord mask
S:key use to decrease encord mask

R:key use to increase blur mask
F:key use to decrease blur mask

T:key use to increase resolution
G:key use to decrease resolution

Y:key use to increase blur/sharpness
H:key use to decrease blur/sharpness

P:key use to increase color

;:key use to decrease color	model image look like this

Step 5: Converting deep fake images to mp4

Convert all the deep fake images into video files. And your deep fake video is ready.

click the image to watch the video:https://youtu.be/EuI7LD_yY6I

Chapter 3

“Deep Fake Identification”

While industry and governments have put forth efforts to limit deepfake use, bans are almost impossible to enforce because, at this time, no one can accurately say whether a video is real or not. The best deep fakes leave no pixelated evidence of a messy edit; their artificiality is virtually undetectable.

ALGORITHMIC DETECTION —

Tech companies, ill-equipped to judge the veracity of videos, need a detection system desperately. So in 2019, some of them teamed up to present their challenge to the world. The competition was called the Deepfake Detection Challenge, an attempt “to spur researchers around the world to build innovative new technologies that can help detect deep fakes and manipulated media."

The Deepfake Detection Challenge released a massive public dataset of 100,000 total clips and sent participants off to the races. After entries by 2114 teams, Belarusian engineer Selim Seferbekov claimed the first prize of $500,000 for his winning model which was 82.56 percent accurate on the public dataset and, when presented with brand new videos, achieved an accuracy of 65.18 percent.

HOW DO HUMANS STACK UP?

The algorithmic detection system’s accuracy score is about the equivalent of a “D” grade in school: certainly not perfect. But what if computers aren’t the sole answer?

“There were all sorts of panic about deep fakes, and we wondered, maybe people are actually decent at detecting digital manipulations,” said Matt Groh, an MIT Ph.D. student who is one of the authors of a recent paper in the Proceedings of the National Academy of Sciences. While his previous research has harnessed AI neural networks for tasks like dermatologist-level classification, he wanted to tap into the facial recognition skills of humans, rather than AI, to see how they stacked up.

“The fascinating thing about deep fake manipulation compared to other forms of manipulation is that it involves faces. And humans — even babies — tend to be really good at

identifying faces," he told me over the phone. So he and other researchers put together a series of media snippets for humans on the internet to judge. Try it out for yourself!

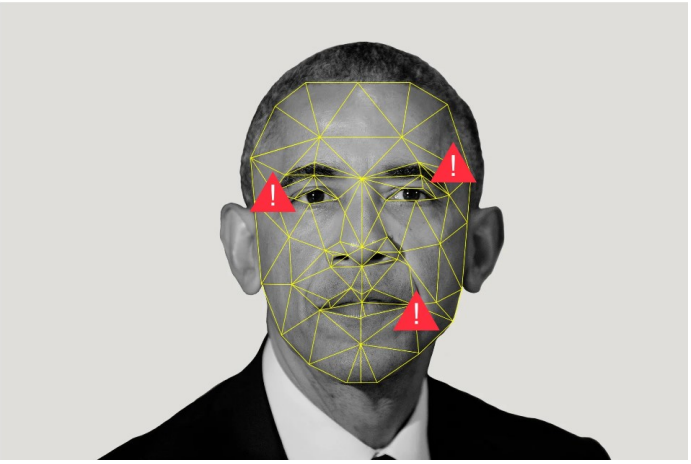
THE FINDINGS —

The research found that people are better than AI — but still not very good — at telling deep fakes from genuine videos. When 882 participants were shown side-by-side videos (one real, one deepfake), 82 percent outperformed the winning AI model. Way to go humans! Interestingly, the research participants' scores didn't improve with more practice or more time spent.

In a more challenging task, participants viewed a single video and guessed whether it was a deep fake or not, moving a slider to report their response ranging from "100% confidence this is NOT a Deep Fake" to "100% confidence this is a Deep Fake." Here, the results were different. Only some people, between 13 percent and 37 percent, performed better than the leading AI model.

We try to create our own model to identify Deep fake Images Using CNN

For The Purpose Of Identification and simplicity of the project, We Stick With Only One person's Face ie Barack Obama



We Created A Binary classification Model Which can Classify images of Barack Obama as Real Or Deep Fake.

Before going to that let's go through a brief about Neural networks and CNN

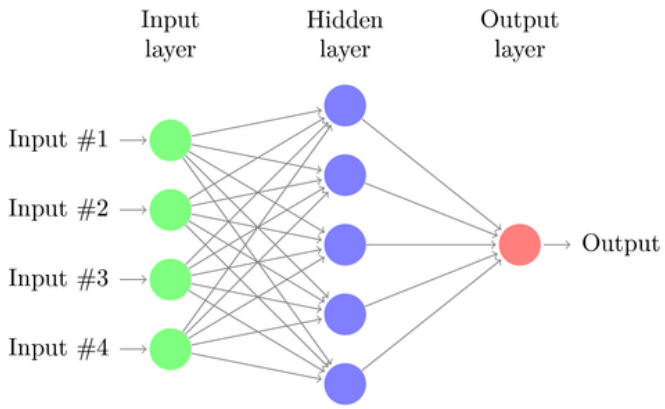
NEURAL NETWORKS

What are neural networks?

Neural networks reflect the behavior of the human brain allowing computer programs to recognize patterns and solve common problems in the fields of AI, machine learning, and deep learning. Neural networks are also known as artificial neural networks (ANNs) or simulated neural networks (SNNs). Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

HOW DO NEURAL NETWORKS WORK?

A neural network has many layers. Each layer performs a specific function, and the complex the network is, the more the layers are, Artificial neural networks are made up of a node layer containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data at a high velocity.



When posed with a request or problem to solve, the neurons run mathematical calculations to figure out if there's enough information to pass on the information to the next neuron. Put more simply, they read all the data and figure out where the strongest relationships exist**.** Neural networks for a computer vision model can be understood using high school math.

STEPS INVOLVED IN WORKING OF NEURAL NETWORK:

1. **Information is fed into the input layer which transfers it to the hidden layer**
2. **The interconnections between the two layers assign weights to each input randomly**
3. **A bias is added to every input after weights are multiplied with them individually**
4. **The weighted sum is transferred to the activation function**
5. **The activation function determines which nodes it should fire for feature extraction**
6. **The model applies an application function to the output layer to deliver the output**
7. **Weights are adjusted, and the output is back-propagated to minimize error**

The model uses a cost function to reduce the error rate. we will have to change the weights with different training models.

The model compares the output with the original result. It repeats the process to improve accuracy. The model adjusts the weights in every iteration to enhance the accuracy of the output.

Convolutional neural network(CNN):

A convolution neural network contains a three-dimensional arrangement of neurons, instead of the standard two-dimensional array. The first layer is called a convolutional layer. Each neuron in the convolutional layer only processes the information from a small part of the visual field. Input features are taken batch-wise like a filter. The network understands the images in parts and can compute these operations multiple times to complete the full image processing. Processing involves the conversion of the image from the HSI scale to grayscale. Furthering the changes in the pixel value will help to detect the edges and images can be classified into different categories.

Applications on Convolution Neural Network:

- **Image processing**
- **Computer Vision**
- **Speech Recognition**
- **Machine translation**

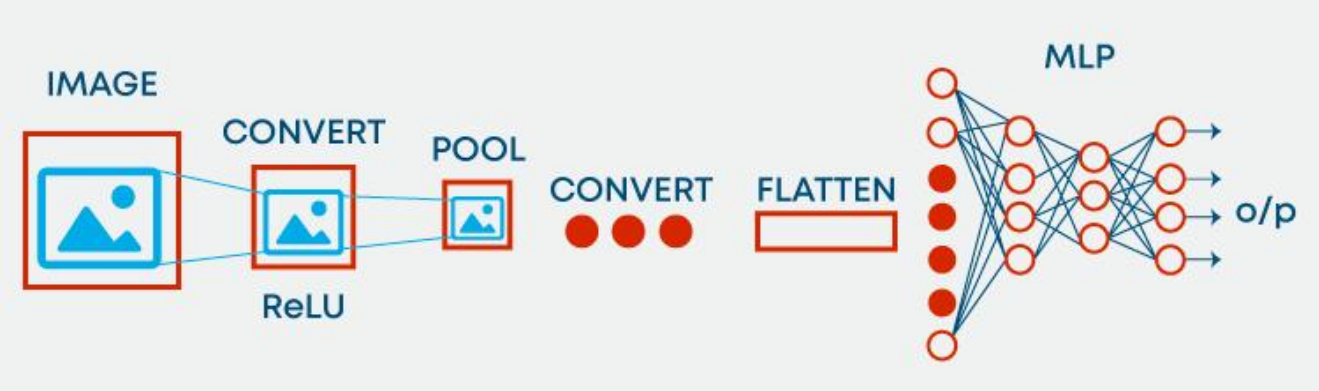


Image Classification Model Of Barack Obama

- **Connecting google colab with google drive:**

```
[ ] from google.colab import drive
drive.mount('/content/drive')
```

- **Unzipping a rare file which was in my google drive**

```
[ ] !unrar x '/content/drive/Shareddrives/Zishan Movies/obama.rar'
```

- **extracting images from videos**

```
[ ] import cv2
import os

vid=cv2.VideoCapture('/content/obama/obamadeepfake/President Obama on Death of Osama bi
n Laden(720P_HD).mp4')

try:
  #creating folder name realkapil
  if not os.path.exists('dfobama'):
    os.makedirs('dfobama')

#if not created then raise error
except OSError:
  print('Error Creating directory of data')
```

```
#frame

currentframe=0

while(True):

  #reading from frame
  ret,frame=vid.read()

  if ret:
    #if video is still left continue creating images
    name='.dfobama/realframe'+str(currentframe)+'.jpg'
    print('Creating..'+name)

    #write the extracted images
    cv2.imwrite(name,frame)



    currentframe+=1

  else:
    break



 vid.release()
cv2.destroyAllWindows()
```

- **Importing Required Libraries for Model Building**

```
Importing Required Libraries for Model Building

1.Libraries:  These are code written by someone else that helps you perform
some common tasks in a less verbose way. An end-to-end accessible platform,
an open-source AI library is a Machine Learning framework that offers
techniques and technologies for software development and the creation of
```

applications. An AI library is empowered by artificial intelligence and is available for commercial and non-commercial uses by the public.

2.NumPy:NumPy can be used to perform a wide variety of mathematical operations on arrays.NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python's built-in sequences.

3.Pandas: Pandas is a Python library. Pandas is used to analyze data. Pandas is a fast, powerful, flexible, and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

4.Matplotlib: Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy.Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

5. Tensorflow: TensorFlow is an open-source library developed by Google primarily for deep learning applications.TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on the training and inference of deep neural networks.

6.Keras: Keras is used for creating deep models which can be productized on smartphones. Keras is also used for distributed training of deep learning models. Keras is a powerful and easy-to-use free open-source Python library for developing and evaluating deep learning models. It wraps the efficient

numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code.

7. OS: The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system dependent functionality.

```
[ ] import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D,MaxPooling2D,Dropout,Flatten,Dense
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import os
```

- Giving the required path address for data training

4.Train & Test: Train/Test is a method to measure the accuracy of your model. It is called Train/Test because you split the data set into two sets: a training set and a testing set. 80% for training, and 20% for testing. You train the model using the training set.

The "training" data set is the general term for the samples used to create the model, while the "test" data set is used to qualify performance. Perhaps traditionally the dataset used to evaluate the final model performance is called the "test set".

5. Validation: In machine learning, model validation is referred to as the process where a trained model is evaluated with a testing data set. The testing data set is a separate portion of the same data set from which the training set is derived.

```
[ ] train='/content/obama/obama database/obama_train'
valid='/content/obama/obama database/obama_vaild'
```

```
train_real=os.path.join(train,"rlobama")
train_fake=os.path.join(train,"dfobama")
```

- **Creating Array of real Images and Deep Fake images:**
  here names for fake images is realframXXXX therefore its showing names in train_fake_names as realframe1671

```
[ ] train_real_names=os.listdir(train_real)
print(train_real_names[:10])
```

```
train_fake_names=os.listdir(train_fake)
print(train_fake_names[:10])
['realframe5264.jpg', 'realframe2307.jpg', 'realframe1341.jpg', 'realframe1671.jpg',
'realframe4536.jpg', 'realframe6152.jpg', 'realframe3530.jpg', 'realframe6073.jpg',
'realframe3564.jpg', 'realframe1920.jpg']
['realframe1671.jpg', 'realframe132.jpg', 'realframe526.jpg', 'realframe23.jpg', 'realframe259.jpg',
'realframe364.jpg', 'realframe1646.jpg', 'realframe415.jpg', 'realframe332.jpg',
'realframe139.jpg']
```

- **Creating a 4x4 graph space to show images in plot for that reason we take 8 real and 8 fake images**

```
[ ] import matplotlib.image as mimg
nrows=4
ncols=4
plt.figure(figsize=(16,9))

real_pic=[]
for i in train_real_names[0:8]:
  real_pic.append(os.path.join(train_real,i))



fake_pic=[]
for i in train_fake_names[0:8]:
  fake_pic.append(os.path.join(train_fake,i))

plt.show()
```

```
[ ] print(real_pic)
print(fake_pic)
['/content/obama/obama database/obama_train/rlobama/realframe5264.jpg',
'/content/obama/obama database/obama_train/rlobama/realframe2307.jpg',
'/content/obama/obama database/obama_train/rlobama/realframe1341.jpg',
'/content/obama/obama database/obama_train/rlobama/realframe1671.jpg',
'/content/obama/obama database/obama_train/rlobama/realframe4536.jpg',
'/content/obama/obama database/obama_train/rlobama/realframe6152.jpg',
'/content/obama/obama database/obama_train/rlobama/realframe3530.jpg',
'/content/obama/obama database/obama_train/rlobama/realframe6073.jpg']
['/content/obama/obama database/obama_train/dfobama/realframe1671.jpg',
'/content/obama/obama database/obama_train/dfobama/realframe132.jpg',
'/content/obama/obama database/obama_train/dfobama/realframe526.jpg',
'/content/obama/obama database/obama_train/dfobama/realframe23.jpg',
'/content/obama/obama database/obama_train/dfobama/realframe259.jpg',
'/content/obama/obama database/obama_train/dfobama/realframe364.jpg',
'/content/obama/obama database/obama_train/dfobama/realframe1646.jpg',
'/content/obama/obama database/obama_train/dfobama/realframe415.jpg']

[ ] pics=real_pic+fake_pic
len(pics)
16
```

- **Showing the plot of 1st 16 images from the data**

```
[ ] for i in range(0,len(pics)):
  data=pics[i].split("/",2)[2] sp=plt.subplot(nrows,ncols,i+1)
  sp.axis('off')
  image=mimg.imread(pics[i])
  plt.imshow(image,cmap="gray")
```

plt.show()

- **Image augmentation is a technique of applying different transformations to original images which results in multiple transformed copies of the same image. Each copy, however, is different from the other in certain aspects depending on the augmentation techniques you apply like shifting, rotating, flipping, etc.**

```
[ ] train_dgen=ImageDataGenerator(rescale=1/255,
                    zoom_range=0.3,
                    rotation_range=50,
                    width_shift_range=0.2,
                    height_shift_range=0.2,
                    horizontal_flip=True)

valid_dgen=ImageDataGenerator(rescale=1/255)
```

- **Dividing Data into batch size of 32**

```
[ ] train_gen=train_dgen.flow_from_directory(train,
                        target_size=(120,120),
                        batch_size=32,
                        class_mode='binary')
Found 6739 images belonging to 2 classes.
```

[ ] valid_gen=valid_dgen.flow_from_directory(valid,
target_size=(120,120),
batch_size=32,
class_mode='binary')

- **Found 1878 images belonging to 2 classes.**

[ ] print(train_gen.class_indices in)
print(train_gen.image_shape)
{'dfobama': 0, 'rlobama': 1}
(120, 120, 3)

[ ] print(valid_gen.class_indices)
print(train_gen.image_shape)
{'fakeobm': 0, 'realobm': 1}
(120, 120, 3)

- **Creating A 2 layered CNN Model**

```
### Creating A 2 **layer** CNN Model
```

CNN:In deep learning, a convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery. A convolutional neural network (CNN) is a type of artificial neural network used in image recognition and processing that is specifically designed to process pixel data.

1.Layers:Keras Layers are the functional building blocks of Keras Models. Each layer is created using numerous layer_() functions. These layers are fed with input information, they process this information, do some computation and hence produce the output. Further, this output of one layer is fed to another layer as its input.

2.Conv2D:
Keras Conv2D is a 2D Convolution Layer, this layer creates a convolution kernel that is wind with layers input which helps produce a tensor of outputs.

3.Kernel:In image processing the kernel is a convolution matrix or mask which can be used for blurring, sharpening, embossing, edge detection, and more by doing a convolution between a kernel and an image.

4.Maxpooling2D:Max pooling operation for 2D spatial data. Down samples the input along its spatial dimensions (height and width) by taking the maximum value over an input window (of size defined by pool_size) for each channel of the input. The window is shifted by strides along each dimension.

5.Dropout: The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Inputs not set to 0 are scaled up by 1/(1 - rate) such that the sum over all inputs is unchanged.

6.Flatten: Flatten is used to flatten the input. For example, if flatten is applied to a layer having input shape as (batch_size, 2,2), then the output shape of the layer will be (batch_size, 4).

7.Dense: Dense layer is the regular deeply connected neural network layer. It is the most common and frequently used layer. Dense layer does the below operation on the input and returns the output. output = activation(dot(input, kernel) + bias) where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer, and bias is a bias vector created by the layer (only applicable if use bias is True). These are all attributes of Dense.

[ ] #performing CNN and Building model

model=Sequential()
#adding 1st layer

```
model.add(Conv2D(32,(3,3),padding="SAME",activation="relu",input_shape=(120,120,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.5))


#adding 2nd layer
model.add(Conv2D(64,(3,3),padding="SAME",activation="relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.5))

#flattening the layer
model.add(Flatten())


model.add(Dense(256,activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(1,activation="sigmoid"))
model.summary()
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 120, 120, 32)	896
max_pooling2d (MaxPooling2D)	(None, 60, 60, 32)	0
dropout (Dropout)	(None, 60, 60, 32)	0
conv2d_1 (Conv2D)	(None, 60, 60, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 30, 30, 64)	0
dropout_1 (Dropout)	(None, 30, 30, 64)	0

flatten (Flatten)	(None, 57600)	0
dense (Dense)	(None, 256)	14745856
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 1)	257

===
Total params: 14,765,505
Trainable params: 14,765,505
Non-trainable params: 0
_____

[ ] import keras
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import Model

[ ]
model.compile(Adam(learning_rate=0.001),loss="binary_crossentropy",metrics=["accuracy"])

- **Training Model for 30 times with 70 steps in each step**

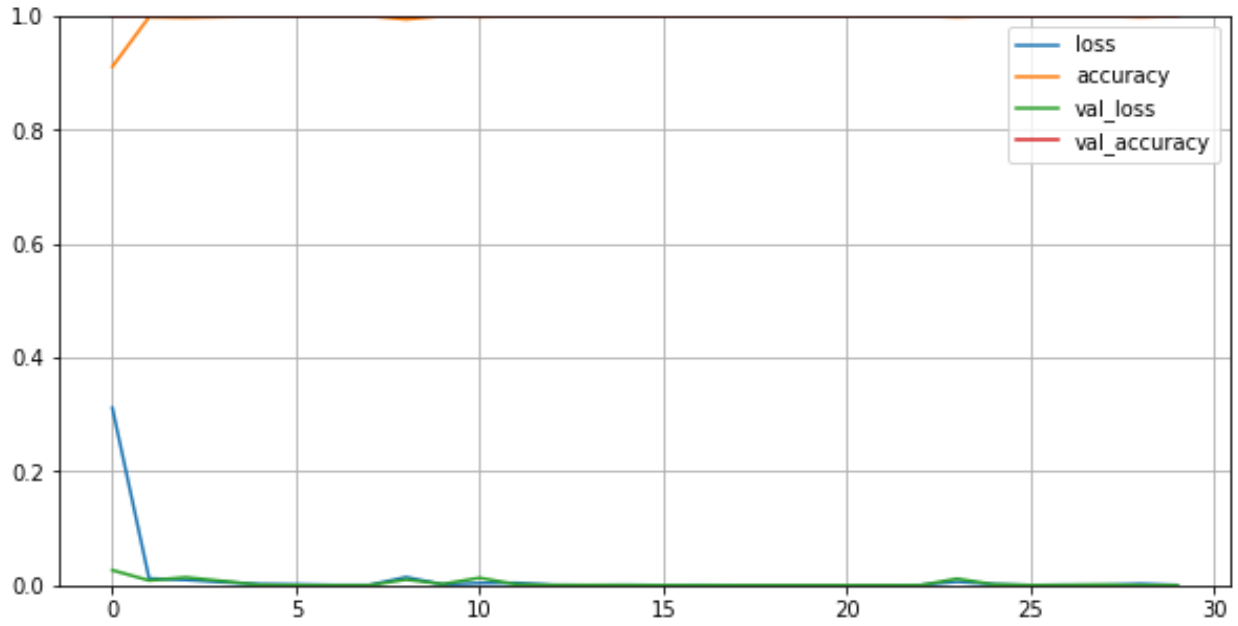
```
[ ] training=model.fit(train_gen,
            steps_per_epoch=70,
            epochs=30,
            validation_data=valid_gen,
            validation_steps=50)
Epoch 1/30
70/70 [==============================] - 133s 2s/step - loss: 0.3116 - accuracy: 0.9111 -
val_loss: 0.0265 - val_accuracy: 1.0000
Epoch 2/30
70/70 [==============================] - 117s 2s/step - loss: 0.0120 - accuracy: 0.9982 -
val_loss: 0.0085 - val_accuracy: 1.0000
Epoch 3/30
70/70 [==============================] - 116s 2s/step - loss: 0.0095 - accuracy: 0.9973 -
val_loss: 0.0140 - val_accuracy: 1.0000
Epoch 4/30
70/70 [==============================] - 116s 2s/step - loss: 0.0054 - accuracy: 0.9987 -
val_loss: 0.0075 - val_accuracy: 1.0000
Epoch 5/30
```

70/70 [==============================] - 115s 2s/step - loss: 0.0021 - accuracy: 0.9996 - val_loss: 6.5389e-04 - val_accuracy: 1.0000
Epoch 6/30
70/70 [==============================] - 116s 2s/step - loss: 0.0016 - accuracy: 0.9996 - val_loss: 2.2925e-04 - val_accuracy: 1.0000
Epoch 7/30
70/70 [==============================] - 116s 2s/step - loss: 5.2662e-04 - accuracy: 1.0000 - val_loss: 2.8022e-05 - val_accuracy: 1.0000
Epoch 8/30
70/70 [==============================] - 116s 2s/step - loss: 5.0778e-04 - accuracy: 1.0000 - val_loss: 4.6269e-05 - val_accuracy: 1.0000
Epoch 9/30
70/70 [==============================] - 115s 2s/step - loss: 0.0143 - accuracy: 0.9942 - val_loss: 0.0101 - val_accuracy: 1.0000
Epoch 10/30
70/70 [==============================] - 115s 2s/step - loss: 0.0014 - accuracy: 1.0000 - val_loss: 0.0022 - val_accuracy: 1.0000
Epoch 11/30
70/70 [==============================] - 115s 2s/step - loss: 0.0037 - accuracy: 0.9987 - val_loss: 0.0131 - val_accuracy: 1.0000
Epoch 12/30
70/70 [==============================] - 115s 2s/step - loss: 0.0042 - accuracy: 0.9996 - val_loss: 0.0016 - val_accuracy: 1.0000
Epoch 13/30
70/70 [==============================] - 114s 2s/step - loss: 6.4122e-04 - accuracy: 1.0000 - val_loss: 3.8156e-04 - val_accuracy: 1.0000
Epoch 14/30
70/70 [==============================] - 117s 2s/step - loss: 1.6162e-04 - accuracy: 1.0000 - val_loss: 4.9777e-04 - val_accuracy: 1.0000
Epoch 15/30
70/70 [==============================] - 127s 2s/step - loss: 5.3711e-04 - accuracy: 1.0000 - val_loss: 4.1546e-04 - val_accuracy: 1.0000
Epoch 16/30
70/70 [==============================] - 115s 2s/step - loss: 2.3820e-04 - accuracy: 1.0000 - val_loss: 5.8384e-05 - val_accuracy: 1.0000
Epoch 17/30
70/70 [==============================] - 115s 2s/step - loss: 4.3288e-04 - accuracy: 1.0000 - val_loss: 4.6348e-05 - val_accuracy: 1.0000
Epoch 18/30
70/70 [==============================] - 115s 2s/step - loss: 1.5277e-04 - accuracy: 1.0000 - val_loss: 7.0785e-05 - val_accuracy: 1.0000
Epoch 19/30
70/70 [==============================] - 115s 2s/step - loss: 1.7928e-05 - accuracy: 1.0000 - val_loss: 4.6433e-05 - val_accuracy: 1.0000

Epoch 20/30
70/70 [==============================] - 115s 2s/step - loss: 1.1446e-04 - accuracy: 1.0000 - val_loss: 6.7816e-05 - val_accuracy: 1.0000
Epoch 21/30
70/70 [==============================] - 115s 2s/step - loss: 2.1377e-05 - accuracy: 1.0000 - val_loss: 3.5402e-05 - val_accuracy: 1.0000
Epoch 22/30
70/70 [==============================] - 115s 2s/step - loss: 2.4409e-05 - accuracy: 1.0000 - val_loss: 1.3763e-05 - val_accuracy: 1.0000
Epoch 23/30
70/70 [==============================] - 116s 2s/step - loss: 3.5818e-04 - accuracy: 1.0000 - val_loss: 2.6019e-06 - val_accuracy: 1.0000
Epoch 24/30
70/70 [==============================] - 115s 2s/step - loss: 0.0063 - accuracy: 0.9982 - val_loss: 0.0116 - val_accuracy: 1.0000
Epoch 25/30
70/70 [==============================] - 116s 2s/step - loss: 0.0018 - accuracy: 1.0000 - val_loss: 0.0016 - val_accuracy: 1.0000
Epoch 26/30
70/70 [==============================] - 116s 2s/step - loss: 1.8411e-04 - accuracy: 1.0000 - val_loss: 2.1455e-04 - val_accuracy: 1.0000
Epoch 27/30
70/70 [==============================] - 115s 2s/step - loss: 5.5819e-04 - accuracy: 1.0000 - val_loss: 5.5557e-04 - val_accuracy: 1.0000
Epoch 28/30
70/70 [==============================] - 115s 2s/step - loss: 4.1677e-04 - accuracy: 1.0000 - val_loss: 0.0013 - val_accuracy: 1.0000
Epoch 29/30
70/70 [==============================] - 115s 2s/step - loss: 0.0026 - accuracy: 0.9982 - val_loss: 6.2620e-09 - val_accuracy: 1.0000
Epoch 30/30
70/70 [==============================] - 115s 2s/step - loss: 3.8189e-04 - accuracy: 1.0000 - val_loss: 1.2747e-04 - val_accuracy: 1.0000

- **plotting a training and accuracy plot**
  pd.DataFrame(trainning.history).plot(figsize=(10,5))

plt.grid(True)
plt.gca().set_ylim(0,1)
plt.show()



[ ] test_loss ,test_acc=model.evaluate(valid_gen)
59/59 [==============================] - 36s 606ms/step - loss: 1.3216e-04 - accuracy: 1.0000

- **Testing Model Accuracy on validation data set here we got 100% Accuracy for validation data**

[ ] model.save("Obama_Deepfake_detection_model.h5")

- **Saving the model**

[ ] modelobama=keras.models.load_model('/content/obama/Obama_Deepfake_detection_model1.h5')

- **Loading the save model for next time**

- **Model Testing For Unseen data (Completely New Dataset)**

[ ] newtestdata='/content/obama/obamadeepfake/Test1'

[ ] newtest_dgen=ImageDataGenerator(rescale=1/255)
newtest_gen=newtest_dgen.flow_from_directory(newtestdata,
target_size=(120,120),
batch_size=32,

class_mode='binary')
Found 240 images belonging to 2 classes.

- **Reshaping the images of new data**

[ ] test_loss ,test_acc=modelobama.evaluate(newtest_gen)
8/8 [==============================] - 7s 837ms/step - loss: 1.6676 - accuracy: 0.8667

- **Here we got accuracy of 86% for unseen data**

[ ] predict=modelobama.predict(newtest_gen)

[ ] predict
[ ] X, y = newtest_gen.next()

- **# Evaluating prediction**

```
print(f"Predicted likelihood: {modelobama.predict(X)[0][0]:.4f}")
print(f"Actual label: {int(y[0])}")
print(f"\nCorrect prediction: {round(modelobama.predict(X)[0][0])==y[0]}")
```

Predicted likelihood: 1.0000
Actual label: 1

Correct prediction: True

```
[ ] correct_real = []
correct_real_pred = []

correct_deepfake = []
correct_deepfake_pred = []

misclassified_real = []
misclassified_real_pred = []

misclassified_deepfake = []
misclassified_deepfake_pred = []
```

- **Creating an array for each predicted image class**
  - **correct_real**
  - **correct_deepfake**

- missclassified_real
- missclassified_fakes

```
[ ] for i in range(len(newtest_gen.labels)):

  # Loading next picture, generating prediction
  X, y = newtest_gen.next()
  pred = modelobama.predict(X)[0][0]

  # Sorting into proper category
  if round(pred)==y[0] and y[0]==1:
    correct_real.append(X)
    correct_real_pred.append(pred)
  elif round(pred)==y[0] and y[0]==0:
    correct_deepfake.append(X)
    correct_deepfake_pred.append(pred)
  elif y[0]==1:
    misclassified_real.append(X)
    misclassified_real_pred.append(pred)
  else:
    misclassified_deepfake.append(X)
    misclassified_deepfake_pred.append(pred)

  # Printing status update
  if i % 1000 == 0:
    print(i, ' predictions completed.')

  if i == len(newtest_gen.labels)-1:
    print("All", len(newtest_gen.labels), "predictions completed")
0  predictions completed.
All 240 predictions completed

[ ] def plotter(images,preds):
  fig = plt.figure(figsize=(16,9))
  subset = np.random.randint(0, len(images)-1, 12)
  for i,j in enumerate(subset):
```

```
    fig.add_subplot(1,1,i+1)
    plt.imshow(np.squeeze(images[j]))
    plt.xlabel(f"Model confidence: \n{preds[j]:.4f}")
    plt.tight_layout()
    ax = plt.gca()
    ax.axes.xaxis.set_ticks([])
    ax.axes.yaxis.set_ticks([])
  plt.show;
  return
```

```
[ ]
len(correct_real)
86

[ ] len(correct_deepfake)
117

[ ] len(misclassified_real)
37

[ ] len(misclassified_deepfake)
0
```

- **Testing Model for random uploaded images**

```
[ ] #now seeing how model works by providing data which was not present previously
import numpy as np
from google.colab import files
from keras.preprocessing import image

uploaded=files.upload()
for frame in uploaded.keys():
 img_path="/content/"+frame
 img= image.load_img(img_path , target_size=(120,120))
 images=image.img_to_array(img)
 images=np.expand_dims(images,axis=0)
 prediction=modelobama.predict(images)
 print(frame)
```

```
if prediction==0:
  print("Real Image Of Barack Obama")

else:
  print("Deepfake Image Of Barack Obama")
```

End

Get Model Codes:
https://colab.research.google.com/drive/1u4tAWvqiBruNUKgfwOPiXT2B6o8mbaWL#scrollTo=FjXudGUxJJrP

Chapter 4

“Use case of Deep Fake”

Positive impact of deepfake technology:

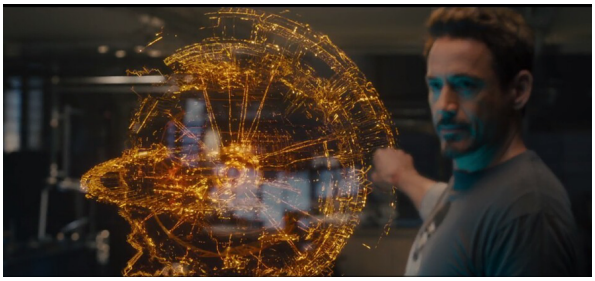
ACCESSIBILITY:
Deepfake uses AI that can make tools that can hear and see with its growing accuracy. It makes media self-sufficient by making accessibility tools smarter and affordable and customized. Thus AI-based tools make much more accessible solutions.

EDUCATION:
Deepfake helps in the visual presentation of the context thereby making it interesting to learn. It can bring the matter under study to life in the classroom. This will create a great impact. It will increase participation and can become an effective teaching-learning tool.

Given the low cost and scale, the use of deep fake can improve success and learning outcomes.

ART :
Deepfake can become an important tool for independent storytellers at the fraction of the cost.

Deepfake can be an excellent alternative for experiencing the realistic feel of comedy and parody in terms of reflection, stretching, contortion and appropriation of real events. As AI generated synthetic media has great potential,it can open up opportunities in the entertainment business. we are seeing many individual creator and YouTubers using this technology.
AI generated graphics and imagery can highly speed up the game development in the video gaming industry. Nvidia demonstrated a hybrid



gaming environment created by deepfake and plans to release it soon.

Another great use of synthetic audio is audio storytelling and book narration. The audio format can be created using the author's voice font or a voice of choice to catch and maintain attention. Business can broaden the reach of their content to a great extent by using synthetic voice-overs for the same actor in different languages.

AUTONOMY AND EXPRESSIONS :

Human rights activists and journalists can use synthetic media to remain anonymous in dictatorial and oppressive regimes. Citizen journalists and activists can also gain a great power by using deepfake technology to report atrocities on traditional and social media. Deepfake , in this case, is mainly used to mask the original voice and face of the individual to maintain their privacy.

Individuals can use deepfake to create avatar realization for self – expression on the internet. They can continue autonomy and expand their purpose, ideas and beliefs by using a personal digital avatar . Synthetic avatar of people with physical or mental disability would help them express themselves online.

Deepfake provides individuals with new tools for self – expression and integration in the digital world.

AMPLIFICATION OF THE MESSAGE AND ITS REACH :

Text-to-speech models can help podcasters to create synthetic audio from text with lesser amount of errors. The process can be sped up using the voice of the podcaster as a font.

Influencers can use deepfake to broaden their reach and expand their audience. A brand could reach a larger no. of customers with targeted and personalized messaging with the help of deepfake. Deepfakes and digital models are becoming a new trend in the fashion and brand marketing.

With influencers and celebrities consent AI Foundation is developing personal and customized AI. This would engage and amplify their audience and would help to create deeper engagement with fans. Moreover , it can also deliver personalized realistic experiences at scale.

DIGITAL RECONSTRUCTION AND PUBLIC SAFETY :

Reconstruction of a crime scene is both an art and a science. It requires both inductive and conductive reasoning along with evidence. AI generated synthetic media can help to reconstruct a crime scene. A team of civil investigators could create a virtual crime scene using cell phone videos. It could be used in drafting autopsy reports and surveillance footage.

INNOVATION :

In many industries ,AI and data are aiding in digital transformation and automation. Deepfake is proving to gain attraction as a way to engage customers and deliver value.

Deepfake can enable customers to convert into models in the fashion retail industry. It allows them to virtually try on the latest clothing and accessories.

It makes online shopping exciting. Data Grid, a Japanese AI Firm , has developed an AI engine. It also generates virtual models for advertising and fashion automatically.

Deepfake allows brands to make a virtual trial

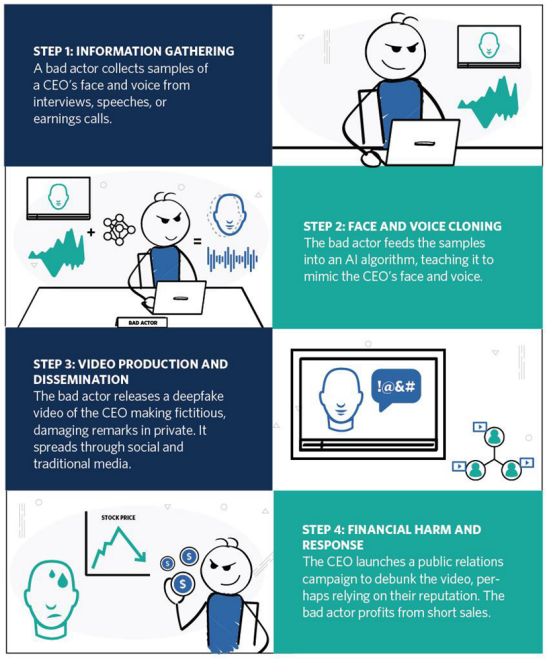
room for their target customers. Users can try product before purchasing them. Retailers can also engage customers at home by creating a mixed virtual world powered by AI.

It would allow them to try on furniture and colors to decorate their space.

Older media can use deepfake technology to enhance the low-resolution images

DANGEROUS APPLICATIONS OF DEEPFAKE TECHNOLOGY:

FIGURE 2
How Deepfake Fabrication of Private Remarks Works

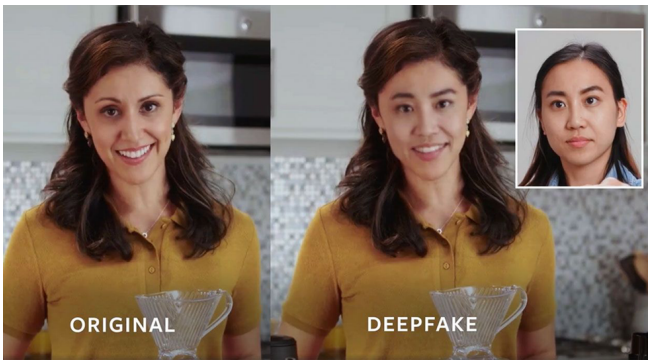


EXTORTION MONEY FROM BUSINESS OR INDIVIDUALS :

Manipulated by deepfake, media file show people making false statements . its now possible to create a video of a CEO making false announcements. An attacker could also blackmail a company by threatening to send the video to the press agencies or post it on social media for money.

FALSE INFORMATION / FALSE NEWS :

Fake news isn't new, and its use is to sow discord and division evident throughout history. It is still in use to deceive the mass and disrupt political , business and social activities. Deepfake can cause doubt and confusion by showing people saying or doing things that they never did.



FAKE VIDEOS :

The first ever version of deepfake was created where a Thai actor replaces President Trump . The skewed video was widely shared on social media and received a lot of attention. Other videos have also gone viral where the actor claims that the President is real but the President claims that he is not.

CONCLUSION:

Deepfake offers a fantastic opportunity to make a positive difference in our lives. AI-generated synthetic media can be a great enabler. From art, expression, and public safety to accessibility and business, new ideas and capabilities for empowerment have emerged from all walks of life. Deep fake has the potential to open doors for everyone regardless of their limitations.

But as synthetic technology becomes more widely available for everyone, the risk of exploitation also rises significantly. Deepfake can taint people's reputations, fabricate and temper with evidence, and defraud in public. One of the significant dangers of deepfake are non-consensual pornography , which accounts to 96% of deepfakes currently found online, done mainly for harassment and abuse.It can erode mass confidence in democratic institutions.

With further improvements and better laws regarding deep fake, it would be a great technology to use.

INDIA'S GROUND ON DEEPFAKES:

Deep fake has deep roots in India where it is majorly used in politics, the film industry, pornography, and even in cases of revenge-defamation. From the cases ranging from deep fake depicting Mr. Manoj Tiwari insulting the state government of Delhi, the Aam Aadmi Party speaking English and Haryanvi, to the generation of pornographic content, an example being that of Ms. Rana Ayyub, it is pertinent that the deepfake can cause major harm to the person himself or herself being targeted as well as the society, by injuring sentiments, thoughts, and perspectives of the

people.

In BJP's Deepfake Video Shared On Wh…

In another case a corporate company named Rephrase. AI made a deepfake of Mr. Hrithik Roshan, showing him congratulating his fans. However, Mr. Hrithik had licensed Rephrase.AI to use his face and make him say anything for confectionery Cadbury. Even if the acts done are legal, there is still a requirement of Laws governing the use of deep fakes. Indian Legislation is still not equipped with provisions that can keep check and balance for deepfakes. However, Indian Laws do provide a block of laws that challenge deepfakes indirectly.

Cadbury Celebration X Hrithik Roshan | …

Cadbury is allowing customers to create an ad for their local stores for free with Bollywood star Shah Rukh Khan on it. The ad, which is in the form of a video, stars the actor, who promotes the local store (it could be Kirana, footwear and electronics) by directly naming the store. The ad is part of the company's "NotJustACadburyAd" campaign that aims to promote local stores. Through a series of videos on its social media channels, Cadbury explains that the company wants to help local stores in India that were heavily impacted during the COVID-19 pandemic.

Supporting Local Retailers This Diwali | …

THE CONSTITUTION OF INDIA:

RIGHT TO PRIVACY:

Deep fakes function by using one's identity, face, or features. This leads to violation and infringement of the right to privacy bestowed by Article 21 of the Constitution of India. Not only in the realm of physical and spatial privacy but also in the protection, preservation, and flow of personal information. Over the years through a series of judgments, the judiciary has shown an attentive inclusive approach in recognizing, protecting, and conserving the right to privacy as a

part and parcel of fundamental rights in the democratic state. In tune with the constitution of India, the concept of privacy has evolved and developed both horizontally as well as vertically. Horizontally (within the individual) it has included sexual autonomy as part of privacy, whereas vertically (state and individual) it imposes an obligation on the state to protect and conserve the right to privacy of every citizen. On the basis of the above discussion, it is clear that the right to privacy is an integral part of the right to life and personal liberty and other freedoms guaranteed in articles 19 and 21 of the Constitution.

THE INDIAN PENAL CODE, 1860 :

DEFAMATION:

Section 500, The Indian Penal Code, 1860 states that a person shall be punished for a term which may extend to two years, or be charged a fine or both if s/he has committed the crime of defamation as mentioned in **Section 499, The Indian Penal Code, 1860.** By using deepfakes, a victim faces harm and damages as he is shown to be involved in obscene acts which s/he never planned or did thus affecting morally, socially, and financially.

FORGERY:

According to **section 463** of **The Indian Penal Code, 1860,** making (a part/ full) of a false document or electronic record for the purpose of causing injury or damages is called forgery**. Section 468 The Indian Penal Code, 1860** states the provision where a forged document is represented as genuine with the intent of cheating. Deepfakes can also be regulated by **section 469 The Indian Penal Code, 1860** as it provides the framework and punishment when a forged document is used to harm one's reputation. Originators and promoters of the deepfakes can be regulated and punished **u/S 471 The Indian Penal Code, 1860** as the section provides the punishment in case where a person has a knowledge/ reason to believe that the said document is forged and still moves forward to use it in a dishonest manner.

SEDITION:

If a person or a group of persons try to/ do use deepfakes to provoke hatred and anti-national sentiments against the rule of law or the presiding Government or bring contempt against the nation and democracy, then they have committed the crime of sedition **u/S 124A The Indian Penal Code, 1860.**

CRIMINAL INTIMIDATION:

Section 506 The Indian Penal Code, 1860 specifies the law and punishment where a person criminally intimidates another person to do/ abstain from doing an act. Hence, if someone is found to threaten somebody via spreading his/ her picture or video then he is liable under **section 506 The Indian Penal Code, 1860**

VOYEURISM:

Section 354C prescribes the punishment for the act of voyeurism where a person is found involved in seeing, capturing any picture of any person for using it in intimate scenes and he/she can be regulated and punished **u/S 354C, The Indian Penal Code, 1860.**

THE COPYRIGHT ACT, 1957:

COPYRIGHT INFRINGEMENT BY DEEPFAKES:

Section 52: The section discusses the concept of fair dealing, though doesn't explain it and further states the areas and scenarios in which there is no infringement of copyright. Here, deepfakes do not come under the umbrella of exceptions mentioned in **section 52 of the Copyright Act, 1957.** Hence, any offence committed via using deepfakes would be termed as copyright infringement.

RIGHTS OF THE OWNER :

Section 57 gives the author of the content the power to restrain or claim damages for any mutilation or distortion of content that can damage the author's reputation. Additionally, the moral right of the author in his work was recognized by the Delhi High Court in Amarnath Sehgal v. Union of India. The author can claim damages for infringement of his moral right in respect of distortion, mutilation, or otherwise modification in his work that would be prejudicial to his honor or reputation.

However, copyright infringement is not the best way to deal with deepfakes as the ownership of the image/ film lies with the producer of the image/ film and thus cannot be challenged.

THE INFORMATION TECHNOLOGY ACT, 2000:

DEEPFAKE INVOLVED WITH COMPUTER-RELATED OFFENCES:

Section 66 D of the Information Technology Act, 2000 holds the provisions for the case where a communication device or computer resource is used mala-fidely for cheating to personate. In other words, a person is made to say and do acts using technology and thus is used for the purpose of cheating. Also, there is a violation of privacy under **section 66 E** of the **Information Technology Act, 2000** when deepfakes are used to invade someone's privacy by capturing, publishing or transmitting someone's intimate pictures or videos.

PUBLISHING SEXUAL CONTENT:

Deep fakes containing pornographic content come under the ambit of section **67A and 66 B, Informational Technology Act, 2000.** The sections outline a fine and appropriate punishment for the publication and transmission of sexual and explicit content involving both

adults and children. Thus, many sites including Pornhub have banned deepfake sexual content.

INTERMEDIARIES' LIABILITY:

The liabilities of intermediaries can also come into question as the intermediaries are the platforms where the deepfake content is posted and thus are regulated by **Section 79** of the **Information Technology Act, 2000.** The section states that the intermediary may take down the content in question after realization/ knowledge of the presence of such content or Court order. However, in the case of Myspace Inc. v Super Cassettes Industries Ltd, the Court held that in circumstances of copyright infringement, the intermediaries are required to take down infringing content upon receiving a notification from private parties without necessarily receiving a Court order.

As per **the Information Technology Rules, 2021,** SSMIs i.e., Social Media Intermediaries (Intermediaries which have registered users above a notified threshold) are required to appoint certain personnel who will need to keep a check and identify the originator of the information and certain types of content. The rules also provide the regulations to the intermediaries to provide a grievance settlement mechanism to solve the issues/complaints/ grievances of the users.

CONCLUSION:

Recently in his response to deep fake technology, India's IT minister had assumed the use of deepfakes to be limited for the circulation of fake news. However, deepfakes can be used solely for entertainment and yet infringe on someone's privacy. Therefore, along with spreading awareness about this novel technology to the masses, adequate attention should be given by the government towards the challenges posed by deepfakes, before they become a menace in India.

Deep fakes have a lot of potential such that they can influence the way we see the world ; however, the negative toll is heavier in this scenario. Thus, there is a major need for laws in India that regulate deepfakes and similar technology to keep checks and controls over any potential nuisance that can cause harm to society and mankind. A committee can also be formed just as proposed in the U.S.A i.e., National Deepfake and Digital Provenance Task Force. Till the time appropriate laws are not implemented , deepfakes should be banned to maintain peace and harmony.

LEGAL ISSUES OF DEEP FAKES:

Deep fake contents initially were only about celebrities, however even ordinary people can create their own deepfake content today. With the widespread use of deepfake content, problems such as manipulation of the public, attacks on personal rights, violations of rights of

intellectual property and personal data protection are becoming more common. Legislators and Big Tech are looking for an effective solution against increasing problems of deep fakes .

The World Intellectual Property Organization (“WIPO”) published the “Draft Issues Paper On Intellectual Property Policy And Artificial Intelligence” on December 2019. The draft holds problems of deepfake contents in terms of intellectual property rights. There are two questions addressed specifically to deepfake issue on the Draft:

“(i) Since deep fakes are created on the basis of data that may be the subject of copyright, to whom should the copyright in a deep fake belong?

(ii) Should there be a system of equitable remuneration for persons whose likenesses and “performances” are used in a deep fake?”

WIPO states that deepfakes may cause more severe problems such as violation of the human rights, right of privacy, personal data protection right, etc. than the copyright infringements. Therefore according to WIPO, the main concern here is whether copyright should even be accorded to deep fake imagery, rather than to whom copyright in a deep fake should belong. As a response, WIPO states that if the deepfake contents are completely contradictory with the victim’s life, the deepfake contents should not be rewarded with copyright protection. For the previous questions, WIPO also mentions that if deepfakes are subject to copyright, it should belong to the inventor of deepfakes. As for the reason that there is no intervention of the source person whose image, sound or other feature is used during the creation of deepfakes but just his/her consent.

Accordingly, copyright is not a good weapon to be put against deepfake, as the victim of deepfakes does not own a copyright interest in their own image. On the other side, the victim of deepfake can take to the right of personal data protection. According to Article 5 (1) of EU General Data Protection Regulation (“GDPR”), “personal data shall be accurate and, where necessary, kept up to date; every reasonable step must be taken to ensure that personal data that are inaccurate, having regard to the purposes for which they are processed, are erased or rectified without delay(‘accuracy’)”.

In the light of the Article 5, if a deepfake content is irrelevant, inaccurate or falsity, they should be erased or rectified without delay. Moreover, even the deepfake content is true or accurate, a data subject —a victim of a deepfake— may exercise the right to be forgotten, granted to European residents in Article 17 of GDPR as the “right to erasure”. In accordance with Article 17, the data subject shall have the right to obtain from the controller the erasure of personal data concerning him or her without undue delay and the data controller shall have the obligation to erase personal data without undue.

CONCLUSION:

Today both states and BigTechs take action against the problems of deepfakes. As an example, BigTechs develop the tools to detect challenges of deepfake contents. Virginia, Texas and California is first states in the US that have regulations against legal issues of deep fakes. The law in Virginia imposes criminal penalties on the distribution of nonconsensual deepfake pornography, whereas the law in Texas prohibits the creation and distribution of deep fake videos intended to harm candidates for public office or influence elections. When current problems are taken into consideration, more legislation and technological tools against deep fakes will probably occur soon.

Chapter 5

“Important Links”

Deepfake Project Resources

Google Form:

https://docs.google.com/forms/d/e/1FAIpQLSe0pHg-eJZmmKPE8KSj2jlibVK6VRttzgXADQTQHcyQaasjiA/viewform

Deep Fake video:

- **Data source** :https://youtu.be/c5uJ1P1fhlc
- **Data destination** :https://youtu.be/UHX127ex2sc
- **Result Deep fake** :https://youtu.be/EuI7LD_yY6I
- **Playlist** :https://youtube.com/playlist?list=PL7dLbJOSm2HbLz4x69k-uXA8kxMqRzcJD

Deepfake detection Model:

Data zip file:
https://drive.google.com/file/d/1FhhFZJxAvKnM9s7Sy1raKPVZ3PPjuYRm/view?usp=sharing

Google colab codes:

Statistical Analysis(models):
https://colab.research.google.com/drive/1FgRpiBR27U992wcC_O4KPcNDvZUGYcs_?usp=sharing

for Obama face detection:
https://colab.research.google.com/drive/1u4tAWvqiBruNUKgfwOPiXT2B6o8mbaWL#scrollTo=FjXudGUxJJrP

GitHub:https://github.com/ZishanSayyed/Art-Of-Artificial-

Presentation:
https://www.canva.com/design/DAE9OR-EXe4/8PPJlMpCO4HPrsszdeGKxg/view?utm_content=DAE9OR-EXe4&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

EDA Dashboard:
https://datastudio.google.com/reporting/b7403197-bbf8-4854-ad8e-da10c07c14da

Raw Data:
Art of AI form (Responses)

Clean Data:
Deepfake data
for model:
Deepfake2

Black Book:
Art of Artificial

Chapter 6

Resources

Reference video:

Eg video:
Tom Cruise as an iron man:
https://youtu.be/A8TmqvTVQFQ

Justice League superman mustache: (positive Use)
https://youtu.be/VnXonpQdOww

Dairy milk ad campaign:
https://youtu.be/5WECsbqAQSk

How to identify deep fake:
https://youtu.be/kYeLBZMTLjk

Images and videos can use in survey
https://www.creativebloq.com/features/deepfake-examples

How to create deep fakes
https://www.youtube.com/watch?v=R-fIN9pkEtc&list=PL7dLbJOSm2Ha24zEAhkx7uXmt6SfkNnRw&index=21&t=830s

How to detect Deep Fake
https://www.youtube.com/watch?v=kYeLBZMTLjk&list=PL7dLbJOSm2Ha24zEAhkx7uXmt6SfkNnRw&index=19

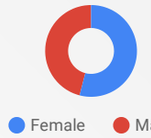
More resources
https://vitalflux.com/how-to-create-detect-deepfakes-deep-learning/

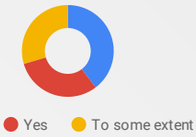
“Meet the group members.”

Name	Connect
Gazala Sayyed	Linkedin: GAZALA SAYED - Mithibai College of Arts Chauhan Institute of Science and AJ College of Commerce and Economics - Mumbai, Maharashtra, India | LinkedIn Gmail: gazala.ed@gmail.com
Hunain Shaikh	Linkedin: Hunain Shaikh - Mithibai College - Mumbai, Maharashtra, India | LinkedIn Gmail: hunainshaik1999@gmail.com
Humaira Khan	Linkedin:Humaira Khan - Mithibai College - Mumbai, Maharashtra, India | LinkedIn Gmail:Humairakhan6868@gmail.com
Jasmin Shaikh	Linkedin: https://www.linkedin.com/in/jasmine-shaikh-aa6a64235 Gmail: jasmineshaikh2812@gmail.com
Zeenat Khan	Linkedin: https://www.linkedin.com/in/zeenat-khan-95b1a7217 Gmail zeenat.j.khan98@gmail.com
Zishan Sayyed	Linkedin:Zishan Sayyed - Mithibai College - Mumbai, Maharashtra, India | LinkedIn Gmail: Zishan Sayyed

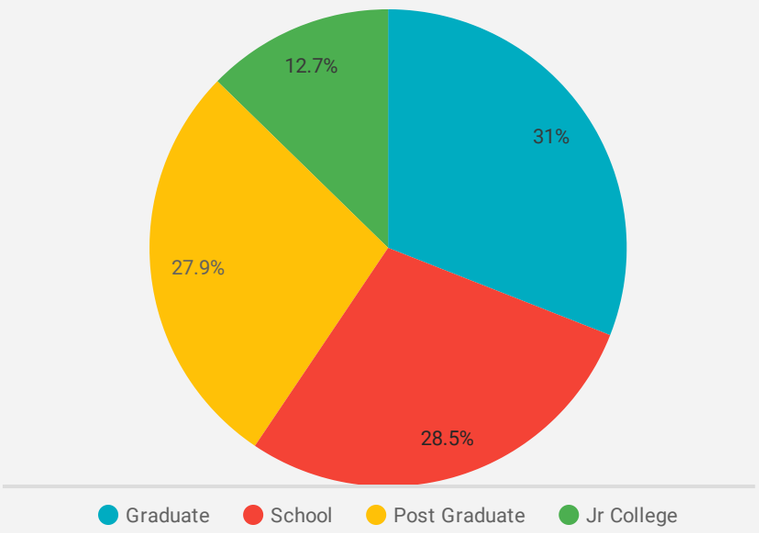
Record Count
520
No data

internet_time_aday
4.07
No data

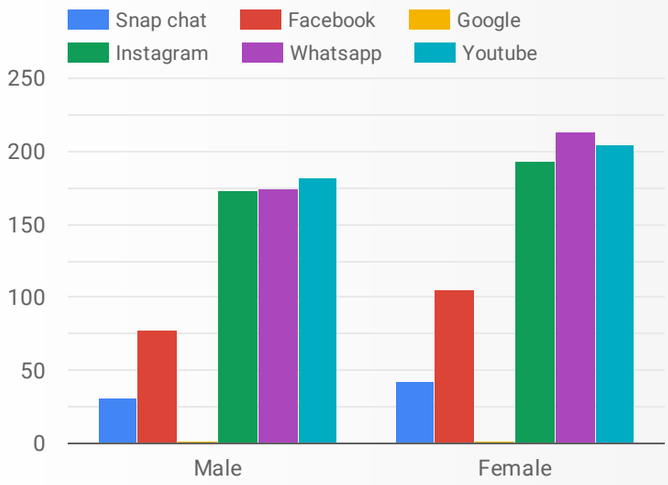




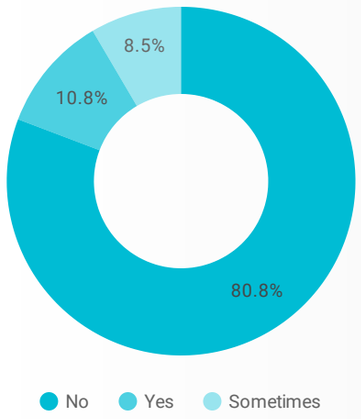
Education



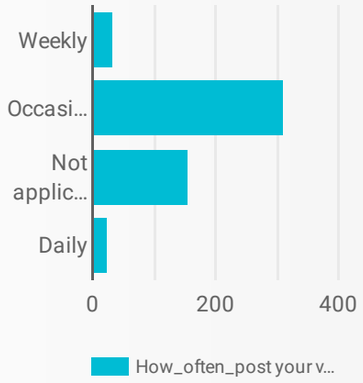
Platform Use



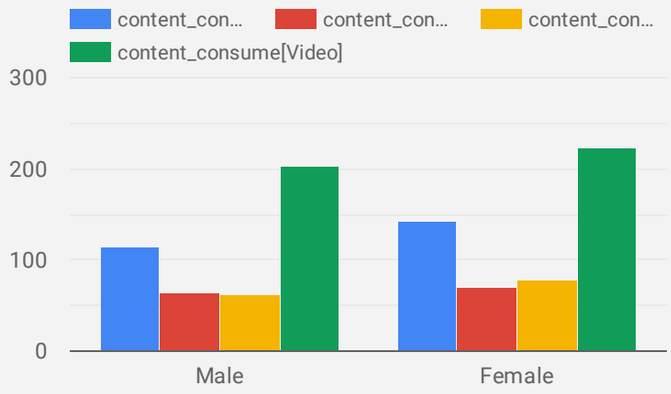
Use Face SwapingApps



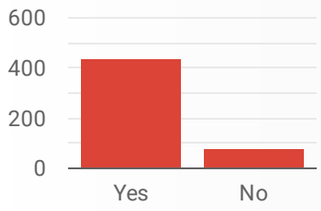
How Often Post On Social Media



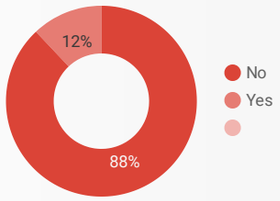
Type Of Content Consume



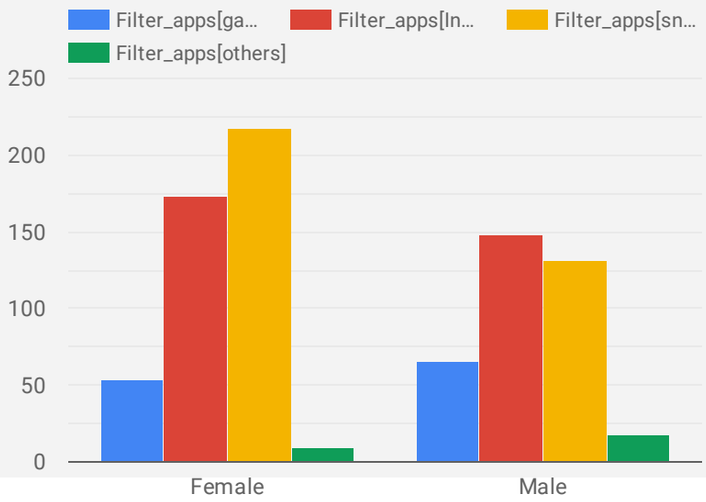
Fv violate persons identity



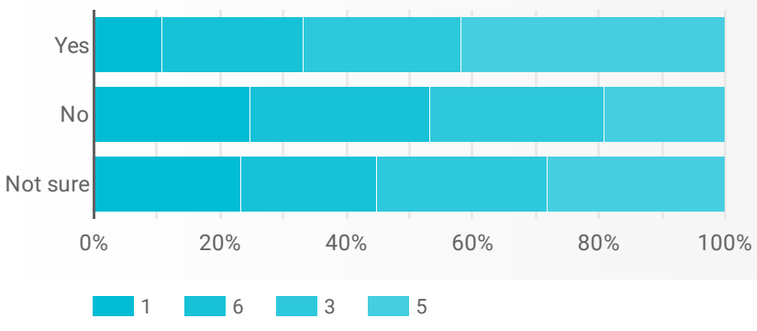
Target of Fake video



Filter App Use

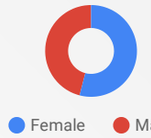


Encountered Fake Video Based on Internet Usage

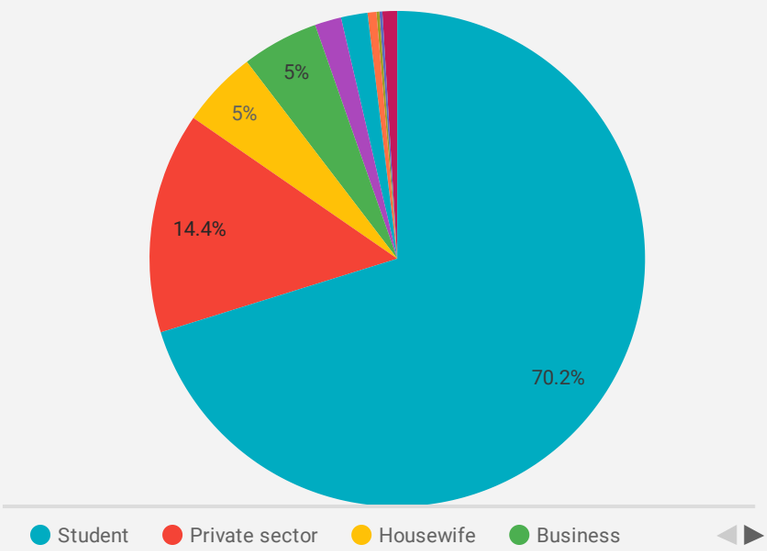


Record Count
520
No data

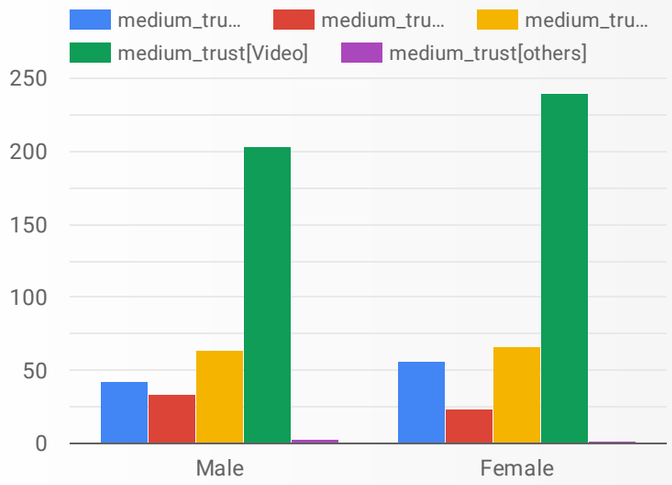
internet_time_aday
4.07
No data



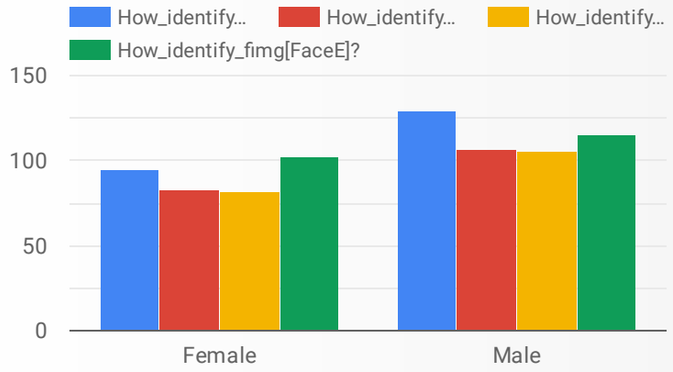
Occupation



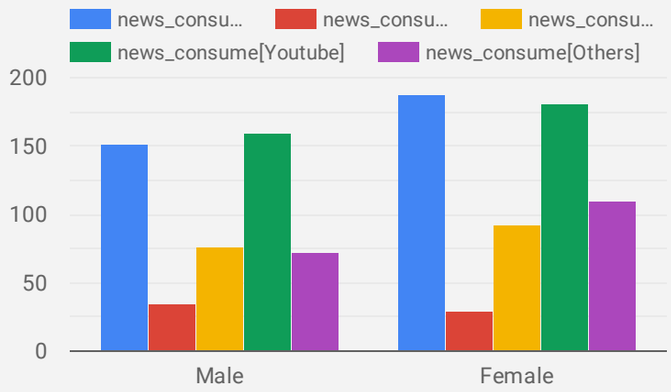
Medium Trust



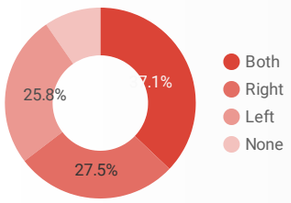
How identify fv



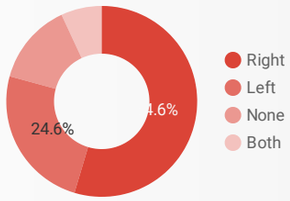
Platform News consume



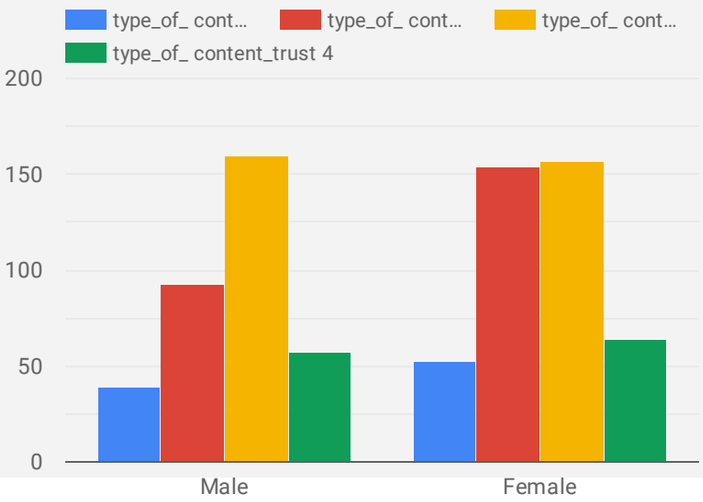
Tom both Fake



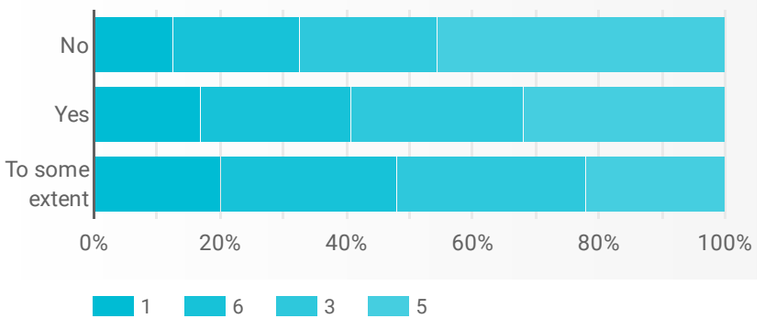
Obama Right Fake



Type Of Content Trust



Detect Fake Video Based on Internet Usage



Art of an artificial

Can You Trust What You See?

* Required

Demographic questions

1. Gender *

Mark only one oval.

- ◯ Male
- ◯ Female

2. What is your educational qualification? *

Mark only one oval.

- ◯ Below SSC
- ◯ SSC
- ◯ HSC/Diploma
- ◯ Graduate
- ◯ Post Graduate
- ◯ Other: _____

3. What is your occupation? *

Mark only one oval.

- ◯ Student
- ◯ Public sector
- ◯ Private sector
- ◯ Business
- ◯ Housewife
- ◯ Retired
- ◯ Unemployed
- ◯ Other: _____

Usage of social media

4. How often do you access the Internet for any purpose? *

Mark only one oval.

- ◯ 0-2 hours a day
- ◯ 2-4 hours a day
- ◯ 4-6 hours a day
- ◯ More than 6 hours

5. Which digital platform do you use the most? *

Check all that apply.

- [ ] Facebook
- [ ] Instagram
- [ ] Youtube
- [ ] Whatsapp
- [ ] Snap chat
- [ ] Tik tok
- [ ] Other: _____

6. Which source of news you consume the most? *

Check all that apply.

- [ ] TV
- [ ] Youtube
- [ ] Facebook
- [ ] Newspaper
- [ ] Google
- [ ] Other: _____

7. Which type of content you consume the most? *

Check all that apply.

- [ ] Audio
- [ ] Video
- [ ] Text
- [ ] Meme
- [ ] Other: _____

8. Which type of medium you trust the most? *

Check all that apply.

- [ ] Audio
- [ ] Video
- [ ] Text
- [ ] Meme
- [ ] Other: _____

9. Among the following whose content you trust the most? *

Check all that apply.

- [ ] Politician
- [ ] Celebrity
- [ ] Sport person
- [ ] Religious leader
- [ ] Other: _____

How much do you believe if you saw a video stating that

10. "The word which is used is 'virgin'. It means unmarried girl, maiden. I don't think these words are not objectionable" - A political leader *

Mark only one oval.

- ◯ 0%- 25%
- ◯ 26%-50%
- ◯ 51%-75%
- ◯ 76%-100%

11. "Shah Rukh is licking my feet and I am feeding him biscuits every now and then. What more can I ask for?" -Amir khan *

Mark only one oval.

- 0%- 25%
- 26%- 30%
- 31%- 75%
- 76%- 100%

12. Corona virus is actually incarnation of god which can save creatures smaller than carnivores - A religious leader. *

Mark only one oval.

- 0%- 25%
- 26%- 50%
- 51%- 75%
- 76%- 100%

*

13. Do you think what you saw in the video/image is always true?(1- being the least and 5 - being the most) *

Mark only one oval per row.

	1	2	3	4	5
Political content					
Educational content					
Religious content					
Humorous content					

14. Which type of content you're most likely to share with others? (1-being the least and 5 being the most) *

Mark only one oval per row.

	1	2	3	4	5
Political content	◯	◯	◯	◯	◯
Religious content	◯	◯	◯	◯	◯
Celebrity content	◯	◯	◯	◯	◯
Humorous content	◯	◯	◯	◯	◯

15. Have you ever encountered a fake video on social media? *

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Not sure

16. Can you detect a fake video? *

Mark only one oval.

- ◯ Yes
- ◯ No *Skip to question 18*
- ◯ To some extent

Identification

17. How do you identify a fake video? *

Check all that apply.

- [ ] Eyes
- [ ] Lip-sync
- [ ] Background
- [ ] Video quality

Detection

18. Can you detect a fake image? *

Mark only one oval.

- ◯ Yes
- ◯ No *Skip to question 20*
- ◯ To some extent

Identification

19. How do you identify a fake photo? *

Check all that apply.

- [ ] Eyes
- [ ] Face edges
- [ ] Quality
- [ ] Background
- [ ] Other: _____

Share content on social media

20. Do you post your videos/images on social media? *

Mark only one oval.

- ◯ Yes
- ◯ No

21. How often do you post your videos/images on social media? *

Mark only one oval.

- ◯ Daily
- ◯ Weekly
- ◯ Occasionally
- ◯ Not applicable

22. Which of the apps you use for filters? *

Check all that apply.

- ☐ Snapchat
- ☐ Instagram
- ☐ B612
- ☐ Phone gallery
- ☐ Other: _____

23. Do you use any face swapping app?

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ Sometimes

24. Have you used any filter which replaces your face with any celebrity? *

Mark only one oval.

- Yes
- No
- Tried, but failed

25. In which type of category you belong to? (with respect to WhatsApp videos/images received) *

Mark only one oval.

- Not believed still shared
- Believed and shared
- Read and ignored
- Read and forward
- Ignored

26. Does fake video violate a person's identity? *

Mark only one oval.

- Yes
- No

27. Have you been a target of fake images/videos?

Mark only one oval.

- Yes
- No

28. Do you want to use fake videos/images? *

Mark only one oval.

- ◯ No
- ◯ Yes, to see celebrity in porn
- ◯ Yes, for pranks
- ◯ Yes, so i can see myself in a movie
- ◯ Yes, for fraud/crime
- ◯ Yes, to take revenge
- ◯ Other: _____

Can you guess?

See carefully!

29. Which one is fake according to you? *

Mark only one oval.

- Left
- Right
- Both
- None

See carefully!

30. Which one is fake according to you? *

Mark only one oval.

- Left
- Right
- Both
- None

Watch carefully!

[redacted]

[redacted] http://youtube.com/watch?v=Vy6Q4wwzz3c

31. Which one of them is fake according to you? *

Mark only one oval.

- First
- Second
- Both
- None

32. Have you ever heard of deep fake? *

Mark only one oval.

- Yes
- No

How much do you think the above images are fake?

1.

33. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%- 60%
- 61%- 90%
- 100% fake

2.

34. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

3.

35. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

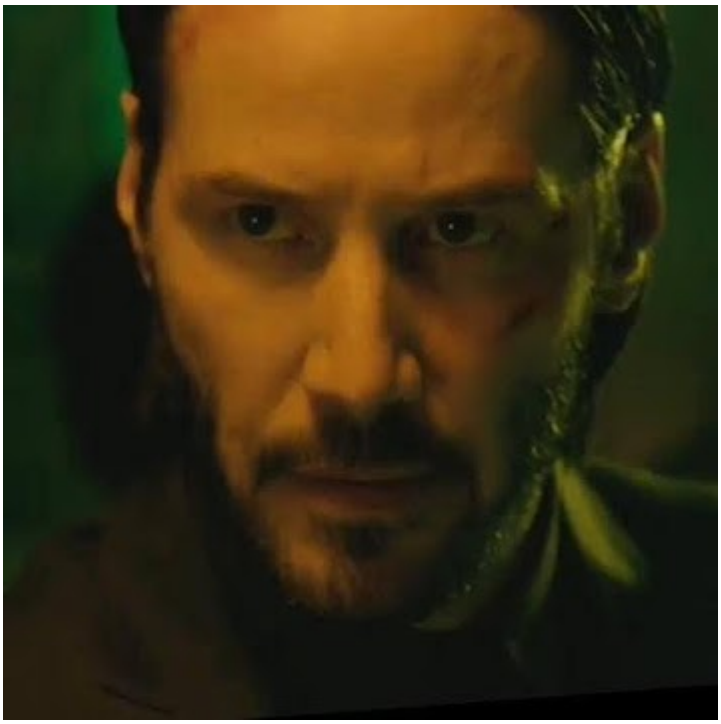
4.

36. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

5.



37. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

6.

38. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

7.

39. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

8.

40. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

9.

41. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

10.

42. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

11.

43. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

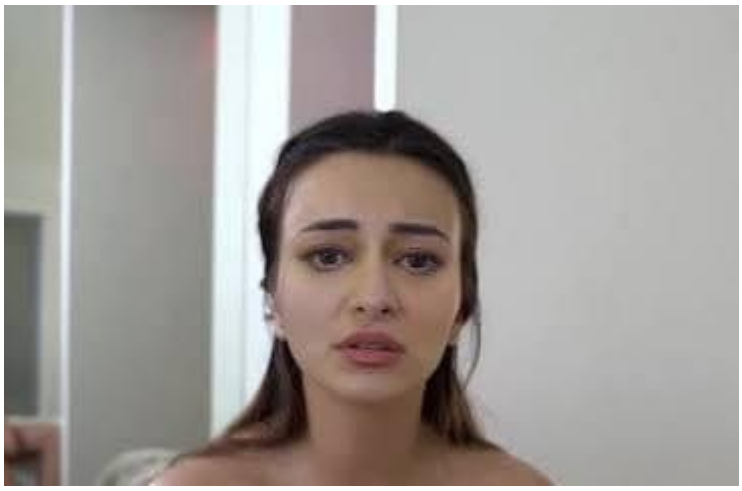
12.

44. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

13.



45. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake

14.

46. *

Mark only one oval.

- 0% fake
- 1%-30%
- 31%-60%
- 61%-90%
- 100% fake