**FAKE NEWS DETECTON**

### A MINI PROJECT REPORT

#### Submitted by

**UTKARSH SABOO [RA2011003011248]**

**ARYAN VATS [RA2011003011205]**

**JATIN SINGHANIA [RA2011003011247]**

*Under the guidance of*

**Mr. Senthil Raja M**

(Assistant Professor, Department of Computing Technologies)

***In partial satisfaction of the requirements for the degree of***

### BACHELOR OF TECHNOLOGY

in

**COMPUTER SCIENCE & ENGINEERING**



**SCHOOL OF COMPUTING**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR - 603203**

**APRIL 2023**

|  |  |
| --- | --- |
| Logo, company name  Description automatically generated | COLLEGE OF ENGINEERING & TECHNOLOGY  SRM INSTITUTE OF SCIENCE & TECHNOLOGY  S.R.M. NAGAR, KATTANKULATHUR – 603 203 |

**BONAFIDE CERTIFICATE**

Certified that this project report **“FAKE NEWS DETECTION”** is the bonafide work of **Utkarsh Saboo (RA2011003011248), Aryan Vats (RA2011003011205), Jatin Singhania (RA2011003011247)** of III Year/VI Sem B.tech(CSE)who carried out the mini project work under my supervision for the course 18CSC305J- Artificial Intelligence in SRM Institute of Science and Technology during the academic year 2022-2023(Even sem).

###### SIGNATURE

Mr. Senthil Raja M

Assistant Professor

Department of Computing Technologies

# ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy,** for his encouragement

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V. Gopal,** for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman,** for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professor Dr. Annapurani Panaiyappan, Professor and Head, Department of Networking and Communications** and **Course Coordinators** for their constant encouragement and support.

We are highly thankful to my Course project Faculty **Dr M Karthikeyan, Assistant Professor, Department of Computing Technologies** for herassistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **HoD Dr. M Pushapaltha, Professor and Head, Department of Computing Technologies** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

# AGENDA

|  |  |
| --- | --- |
| Chapter | Contents |
|  | Abstract |
|  | Introduction |
|  | Algorithms Used |
|  | Libraries Used |
|  | Literature Survey |
|  | Codes |
|  | Output with Snapshots |
|  | Conclusion and Future Work |
|  | References |

# ABSTRACT

This report describes the methodology used for our AI-based fake news detection system. We will provide details on the dataset used, pre-processing techniques applied, feature extraction methods used, and machine learning algorithms used for classification.  
  
**Dataset:**  
We used a publicly available dataset of news articles that were labelled as either real or fake of multiple domains from Kaggle. The dataset contained a total of 6500+ news articles, with 3,200+ labelled as real and 3,200+ labelled as fake. The truthful news articles published contain true description of real world events, while the fake news websites contain claims that are not aligned with facts. The conformity of claims from the politics domain for many of those articles can be manually checked with fact checking websites such as politifact.com and snopes.com.  
  
**Pre-processing:**  
We applied several pre-processing techniques to the dataset to prepare it for feature extraction and classification. These techniques included removing stop words, stemming, and converting all text to lowercase.  
  
**Feature Extraction:**  
We used several feature extraction methods to extract relevant information from the news articles. These methods included bag-of-words, TF-IDF, and word embeddings. We also extracted features related to the source of the news article, such as the domain name and the number of external links.  
  
**Classification:**  
We used several machine learning algorithms for classification, including logistic regression, decision trees, and support vector machines. We also used ensemble methods, such as random forests and gradient boosting, to improve the performance of the classification models.  
  
**Evaluation:**  
We evaluated the performance of our AI-based fake news detection system using several evaluation metrics, including accuracy, precision, recall, and F1 score. We also used cross-validation to ensure that our results were robust and not overfitting to the training data.

# INTRODUCTION

AI-based fake news detection is a rapidly growing field that aims to combat the spread of misinformation and disinformation. The advent of the World Wide Web and the rapid adoption of social media platforms (such as Whatsapp and Instagram) paved the way for information dissemination that has never been witnessed in the human history before. With the current usage of social media platforms, consumers are creating and sharing more information than ever before, some of which are misleading with no relevance to reality, it has become increasingly difficult to distinguish between real and fake news. AI-based fake news detection systems use machine learning algorithms to analyse news articles and social media posts to identify patterns and characteristics that are indicative of fake news. These systems can analyse the language used, the sources cited, and the overall tone of the article to determine its credibility.

While AI-based fake news detection is still in its early stages and Automated classification of a text article as misinformation or disinformation is a challenging task. Even an expert in a particular domain has to explore multiple aspects before giving a verdict on the truthfulness of an article but it has the potential to be a powerful tool in the fight against fake news and the spread of misinformation.

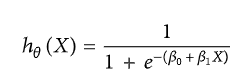
There has been a rapid increase in the spread of fake news in the last decade, most prominently observed in the 2016 US elections Such proliferation of sharing articles online that do not conform to facts has led to many problems not just limited to politics but covering various other domains such as sports, health, and also science One such area affected by fake news is the financial markets, where a rumour can have disastrous consequences and may bring the market to a halt.

A study included extracting linguistic features such as n-grams from textual articles and training multiple ML models including K-nearest neighbor (KNN), support vector machine (SVM), logistic regression (LR), linear support vector machine (LSVM), decision tree (DT), and stochastic gradient descent (SGD), achieving the highest accuracy (92%) with SVM and logistic regression.

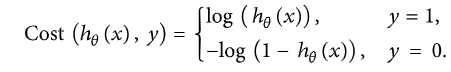
# ALGORITHMS USED

* **Logistic Regression:**

As we are classifying text on the basis of a wide feature set, with a binary output (true/false or true article/fake article), a logistic regression (LR) model is used, since it provides the intuitive equation to classify problems into binary or multiple classes. We performed hyperparameters tuning to get the best result for all individual datasets, while multiple parameters are tested before acquiring the maximum accuracies from LR model. Mathematically, the logistic regression hypothesis function can be defined as follows:

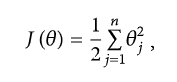


Logistic regression uses a sigmoid function to transform the output to a probability value; the objective is to minimize the cost function to achieve an optimal probability. The cost function is calculated as shown below:

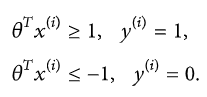


* **Support Vector Machines (SVM):**

Support vector machine (SVM) is another model for binary classification problem and is available in various kernels functions. The objective of an SVM model is to estimate a hyperplane (or decision boundary) on the basis of feature set to classify data points. The dimension of hyperplane varies according to the number of features. As there could be multiple possibilities for a hyperplane to exist in an *N*-dimensional space, the task is to identify the plane that separates the data points of two classes with maximum margin. A mathematical representation of the cost function for the SVM model is shown below:



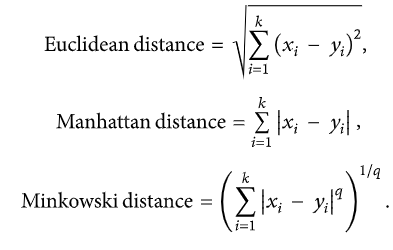
such that



The function above uses a linear kernel. Kernels are usually used to fit data points that cannot be easily separable or data points that are multidimensional. In our case, we have used sigmoid SVM, kernel SVM (polynomial SVM), Gaussian SVM, and basic linear SVM models.

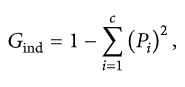
* **K-Nearest Neighbors:**

KNN is an unsupervised machine learning model where a dependent variable is not required to predict the outcome on a specific data. We provide enough training data to the model and let it decide to which particular neighborhood a data point belongs. KNN model estimates the distance of a new data point to its nearest neighbors, and the value of *K* estimates the majority of its neighbors’ votes; if the value of *K* is 1, then the new data point is assigned to a class which has the nearest distance. The mathematical formulae to estimate the distance between two points can be calculated as follows:



* **Random Forest (RF):**

Random forest (RF) is an advanced form of decision trees (DT) which is also a supervised learning model. RF consists of large number of decision trees working individually to predict an outcome of a class where the final prediction is based on a class that received majority votes. The error rate is low in random forest as compared to other models, due to low correlation among trees [33]. Our random forest model was trained using different parameters; i.e., different numbers of estimators were used in a grid search to produce the best model that can predict the outcome with high accuracy. There are multiple algorithms to decide a split in a decision tree based on the problem of regression or classification. For the classification problem, we have used the Gini index as a cost function to estimate a split in the dataset. The Gini index is calculated by subtracting the sum of the squared probabilities of each class from one. The mathematical formula to calculate the Gini index is as follows:



* **Boosting Ensemble Classifiers:**

boosting is another widely used ensemble method to train weak models to become strong learners. For that purpose, a forest of randomized trees is trained, and the final prediction is based on the majority vote outcome from each tree. This method allows weak learners to correctly classify data points in an incremental approach that are usually misclassified. Initially equal weighted coefficients are used for all data points to classify a given problem. In the successive rounds, the weighted coefficients are decreased for data points that are correctly classified and are increased for data points that are misclassified. Each subsequent tree formed in each round learns to reduce the errors from the preceding round and to increase the overall accuracy by correctly classifying data points that were misclassified in previous rounds. One major problem with boosting ensemble is that it might overfit to the training data which may lead to incorrect predictions for unseen instances. There are multiple boosting algorithms available that can be used for both the purposes of classification and regression. In our experiments we used XGBoost algorithms for classification purpose.

* **Linear SVM:**

Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

# LIBRARIES USED

1. Numpy: Numpy is a Python library used for scientific computing. It provides support for large, multi-dimensional arrays and matrices, along with a wide range of mathematical functions to operate on them.  
  
2. Pandas: Pandas is a Python library used for data manipulation and analysis. It provides data structures for efficiently storing and manipulating large datasets, along with a wide range of functions for data cleaning, filtering, and transformation.  
  
3. Itertools: Itertools is a Python library used for creating iterators and generators. It provides a wide range of functions for working with iterators, including functions for combining, filtering, and transforming them.  
  
4. Matplotlib.pyplot: Matplotlib.pyplot is a Python library used for creating visualizations and plots. It provides a wide range of functions for creating line plots, scatter plots, bar charts, and more.  
  
5. Plotly.graph\_objects: Plotly.graph\_objects is a Python library used for creating interactive visualizations and plots. It provides a wide range of functions for creating line plots, scatter plots, bar charts, and more, along with support for interactivity and animation.  
  
6. Sklearn: Sklearn is a Python library used for machine learning and data analysis. It provides a wide range of functions for data preprocessing, feature selection, model selection, and evaluation.  
  
7. Tfidfvectorizer: Tfidfvectorizer is a function in Sklearn used for converting text data into numerical vectors. It calculates the term frequency-inverse document frequency (TF-IDF) of each word in a document, which is a measure of how important the word is to the document.  
  
8. Passive aggressive classifier: Passive aggressive classifier is a type of machine learning algorithm used for binary classification. It is a type of online learning algorithm that updates its model based on each new example it receives.  
  
9. Confusion\_matrix: Confusion\_matrix is a function in Sklearn used for evaluating the performance of a classification model. It calculates the number of true positives, true negatives, false positives, and false negatives, and displays them in a matrix.  
  
10. Accuracy\_score: Accuracy\_score is a function in Sklearn used for evaluating the accuracy of a classification model. It calculates the percentage of correctly classified examples in a dataset.  
  
11. Classification\_report: Classification\_report is a function in Sklearn used for evaluating the performance of a classification model. It displays precision, recall, and F1-score for each class in a dataset, along with the overall accuracy of the model.

# LITERARY SURVEY

1. [A survey on fake news and rumour detection techniques](https://www.sciencedirect.com/science/article/pii/S0020025519304372). Information Sciences, 2019, 497: 38-55.
2. [Detection and resolution of rumours in social media: A survey](https://dl.acm.org/doi/abs/10.1145/3161603). ACM Computing Surveys (CSUR), 2018, 51(2): 1-36.
3. [The Spread of True and False News Online](https://science.sciencemag.org/CONTENT/359/6380/1146.abstract). Science, 2018, 359(6380): 1146-1151.
4. [Fake News Detection on Social Media: A Data Mining Perspective](https://dl.acm.org/doi/abs/10.1145/3137597.3137600?casa_token=Mf0tvofQf7kAAAAA:LgdXVmsJzYxVyrTgrhoFio_zxDXORoh6NNGP4__D64yam0rOKfwdbi__38Jg01U7pC-M19Tkb2NC_BU). ACM SIGKDD explorations newsletter, 2017, 19(1): 22-36.
5. DD-2021 [Causal Understanding of Fake News Dissemination on Social Media](http://www.cs.iit.edu/~kshu/files/kdd_causal.pdf)
6. SIGIR-2021 [User Preference-aware Fake News Detection](https://arxiv.org/pdf/2104.12259) [code](https://github.com/safe-graph/GNN-FakeNews)
7. CIKM-2020 [FANG : Leveraging Social Context for Fake News Detection Using Graph Representation](https://dl.acm.org/doi/abs/10.1145/3340531.3412046?casa_token=33FpLHu6h20AAAAA:fc2L3COGdQCca7fS2l4rOjcP_LzmDMVI1fROs9Yxi0m7xTuyQUpec9sm6MZe0_Yli7Vo4tcDh6nURN8) [code](https://github.com/nguyenvanhoang7398/FANG)
8. ICDM-2020 [Adversarial Active Learning based Heterogeneous Graph Neural Network for Fake News Detection](https://ieeexplore.ieee.org/abstract/document/9338358/)
9. WSDM-2019 [Beyond News Contents : The Role of Social Context for Fake News Detection](https://dl.acm.org/doi/abs/10.1145/3289600.3290994?casa_token=vzRcFcZbogkAAAAA:CgTc3CqhxgZ3JqqwPLrCAz_vVP2wShHGZvZnLZdeM2Evss5Uqu4-L1UUhLVB-G62_hfT-WcqZLW52gY)
10. AAAI-2018 [Early Detection of Fake News on Social Media Through Propagation Path Classification with Recurrent and Convolutional Networks](https://ojs.aaai.org/index.php/AAAI/article/view/11268).
11. AAAI-2016 [News Verification by Exploiting Conflicting Social Viewpoints in Microblogs](https://ojs.aaai.org/index.php/AAAI/article/download/10382/10241)
12. CIKM-2017 [CSI : A Hybrid Deep Model for Fake News Detection](https://dl.acm.org/doi/abs/10.1145/3132847.3132877?casa_token=qUOs7PlAOKYAAAAA:wvXMJ4nzbcW6CWGTJCREzIvR8vkxXe4rt7tlI1-k-_GANPG87nPv8Z2iaCQs0x_uVGlaPkbnLzMBuO4).
13. EMNLP-2020 [Where Are the Facts? Searching for Fact-checked Information to Alleviate the Spread of Fake News](https://www.aclweb.org/anthology/2020.emnlp-main.621.pdf)
14. KDD-2019 [Defend: Explainable Fake News Detection](https://dl.acm.org/doi/abs/10.1145/3292500.3330935?casa_token=xfpltxWHUwEAAAAA:mifb7BpUrE-nAm4hHpAzW2Gozw8g_xmA2j6UXRzJKm0lAUS0Z8gNEXpE3FRWJnSpeIeKBE4cuB45tYc).

# CODES

# IMPORTING LIBRARIES

**import** numpy **as** np

**import** pandas **as** pd

**import** itertools

**import** matplotlib.pyplot **as** plt

**import** plotly.graph\_objects **as** pgo

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.feature\_extraction.text **import** TfidfVectorizer

**from** sklearn.linear\_model **import** PassiveAggressiveClassifier

**from** sklearn.metrics **import** classification\_report

**from** sklearn.metrics **import** accuracy\_score, confusion\_matrix

**import** warnings

warnings**.**filterwarnings("ignore")

# IMPORTING DATASET

data **=** pd**.**read\_csv(r'C:\Users\SABOO\Downloads\news\_dataset.csv')

data**.**head()

# DATASET DESCRIPTION

data**.**shape

data**.**info()

labels **=** data**.**label

labels**.**head()

labels**.**value\_counts()

i**=**labels**.**value\_counts()

fig**=**pgo**.**Figure(data**=**[pgo**.**Bar(x**=**['Real','Fake'],y**=**i,text**=**i,textposition**=**'auto')])

fig**.**show()

# DATASET SPLITTING

x\_train, x\_test, y\_train, y\_test**=**train\_test\_split(data['text'], labels, test\_size**=**0.2, random\_state**=**7)

x\_train**.**shape

y\_test**.**shape

tfidf\_vectorizer**=**TfidfVectorizer(stop\_words**=**'english', max\_df**=**0.7)

tfidf\_train**=**tfidf\_vectorizer**.**fit\_transform(x\_train)

tfidf\_test**=**tfidf\_vectorizer**.**transform(x\_test)

# ACCURACY FOR DIFFERENT CLASSIFIERS

**from** sklearn.linear\_model **import** **LogisticRegression**

lr **=** LogisticRegression(random\_state **=** 0)

lr**.**fit(tfidf\_train,y\_train)

y\_pred**=**lr**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

**from** sklearn.neighbors **import** **KNeighborsClassifier**

knc **=** KNeighborsClassifier(n\_neighbors **=** 2, metric **=** 'minkowski', p **=** 2)

knc**.**fit(tfidf\_train,y\_train)

y\_pred**=**knc**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

**from** sklearn.svm **import** **SVC**

svc **=** SVC(kernel **=** '**linear**', random\_state **=** 0)

svc**.**fit(tfidf\_train,y\_train)

y\_pred**=**svc**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

**from** sklearn.svm **import** **SVC**

psvc **=** SVC(kernel **=** '**sigmoid**', random\_state **=** 0)

psvc**.**fit(tfidf\_train,y\_train)

y\_pred**=**psvc**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

**from** sklearn.svm **import** **SVC**

rsvc **=** SVC(kernel **=** '**rbf**', random\_state **=** 0)

rsvc**.**fit(tfidf\_train,y\_train)

y\_pred**=**rsvc**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

**from** sklearn.tree **import** **DecisionTreeClassifier**

dtc **=** DecisionTreeClassifier(criterion **=** 'entropy', random\_state **=** 0)

dtc**.**fit(tfidf\_train,y\_train)

y\_pred **=** dtc**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

**from** sklearn.ensemble **import** **RandomForestClassifier**

rfc **=** RandomForestClassifier(n\_estimators **=** 10, criterion **=** 'entropy', random\_state **=** 0)

rfc**.**fit(tfidf\_train,y\_train)

y\_pred **=** rfc**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

**from** sklearn.ensemble **import** **GradientBoostingClassifier**

gbc **=** GradientBoostingClassifier()

gbc**.**fit(tfidf\_train,y\_train)

y\_pred **=** gbc**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

pac**=PassiveAggressiveClassifier**(max\_iter**=**50)

pac**.**fit(tfidf\_train,y\_train)

y\_pred**=**pac**.**predict(tfidf\_test)

score**=**accuracy\_score(y\_test,y\_pred)

print(f'Accuracy: {round(score**\***100,2)}%')

# PLOTTING ACCURACY OF CLASSIFIERS

**import** matplotlib.pyplot **as** plt

classifiers **=** ['Logistic Regression', 'K-Nearest Classifier', 'Support Vector Classifier(linear)', 'Support Vector Classifier(rbf)', 'Support Vector Classifier(sigmoid)',

'Decision Tree Classifier', 'Random Forest Classifier', 'Gradient Boost Classifier', 'Passive Aggressive Classifier']

accuracies **=** [0.9171, 0.5896, 0.9305, 0.9321, 0.9290, 0.8066, 0.8374, 0.8919, 0.9290]

plt**.**barh(classifiers, accuracies, color**=**['black', 'red', 'green', 'blue', 'brown', 'grey', 'pink', 'crimson', 'maroon'])

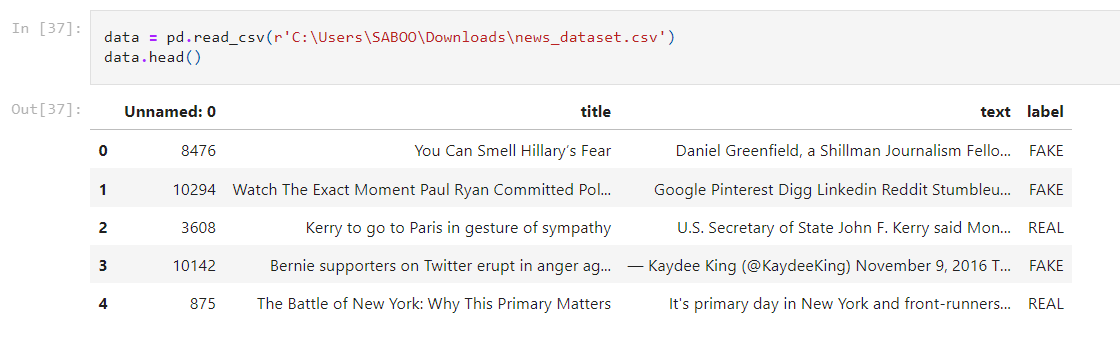
plt**.**ylabel("Classifiers")

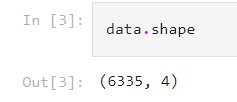
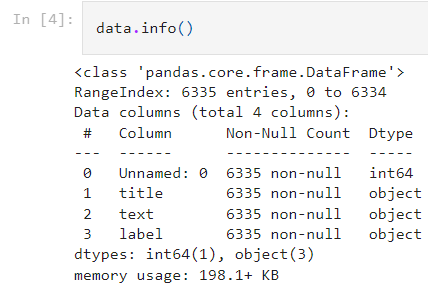
plt**.**xlabel("Accuracy")

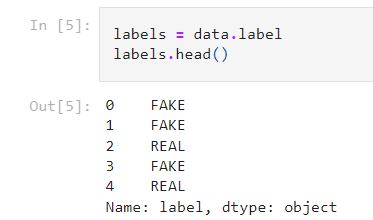
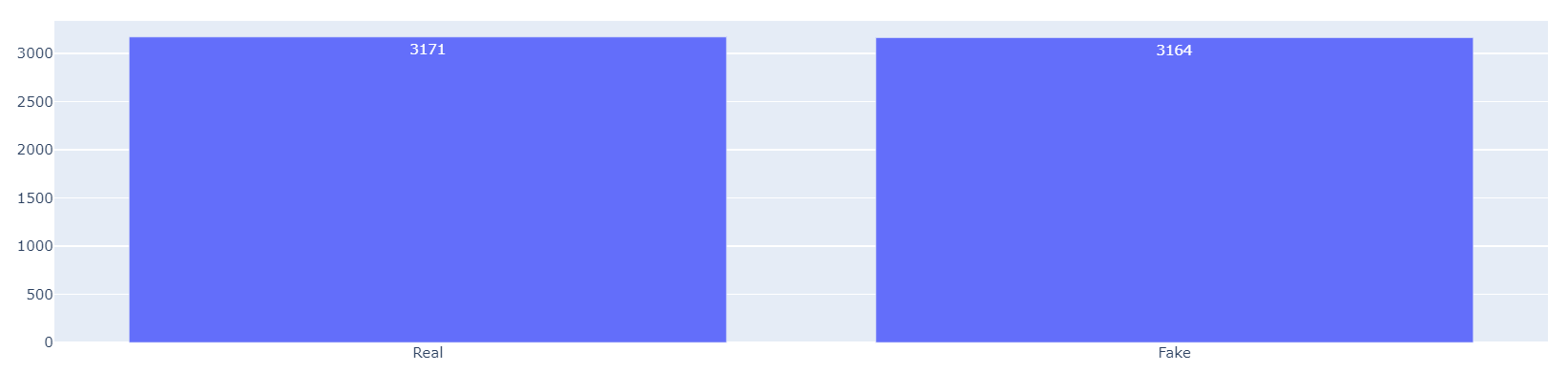
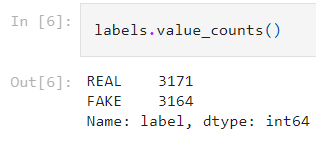
plt**.**title("Classifiers and their accuracies")

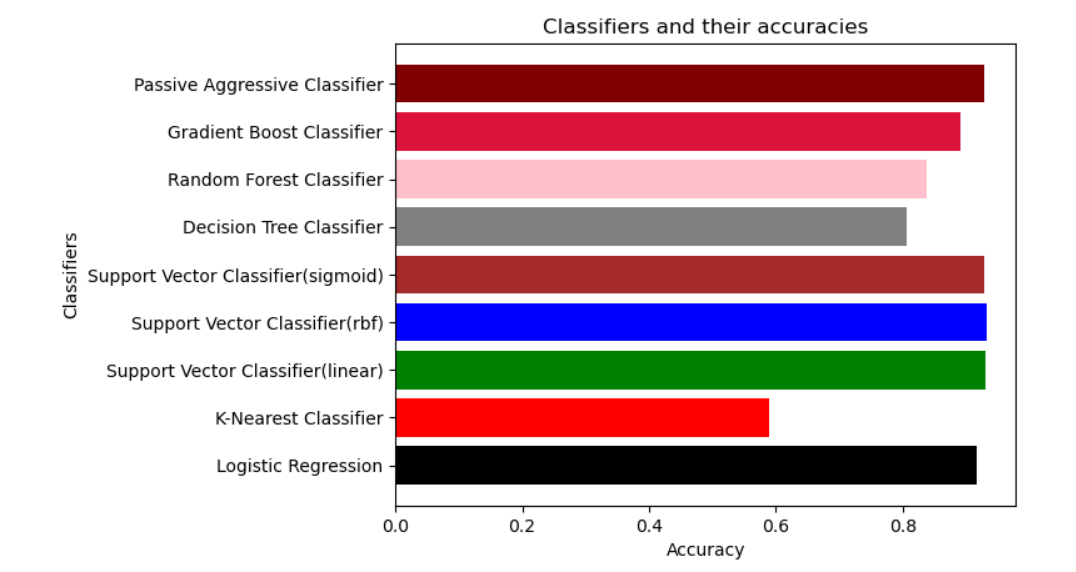
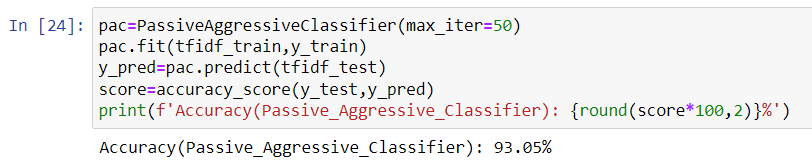
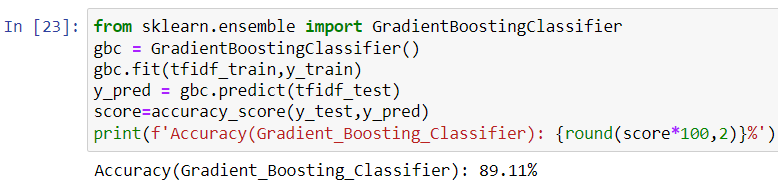
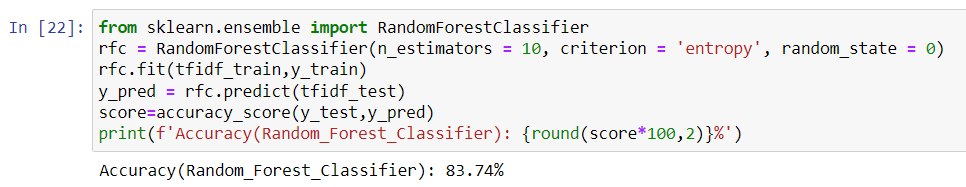
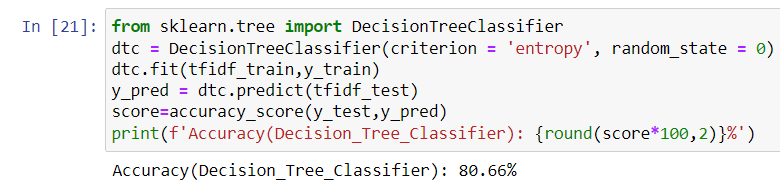
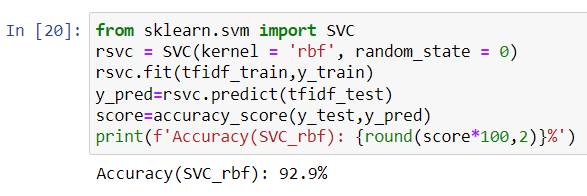
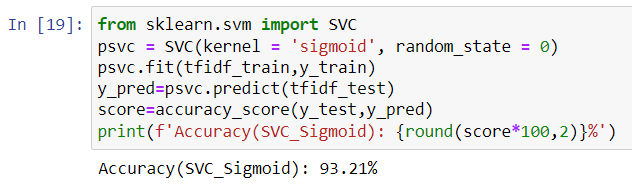
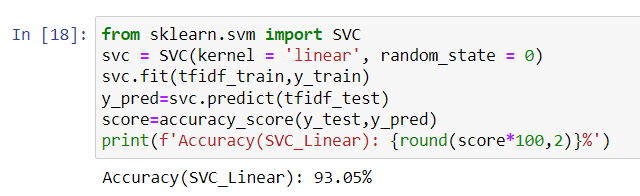
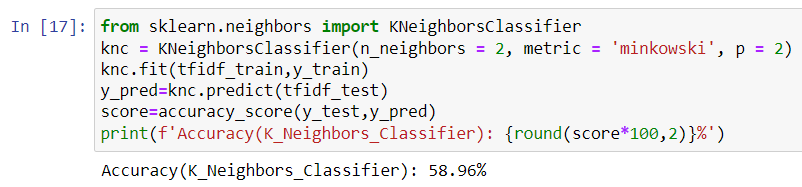
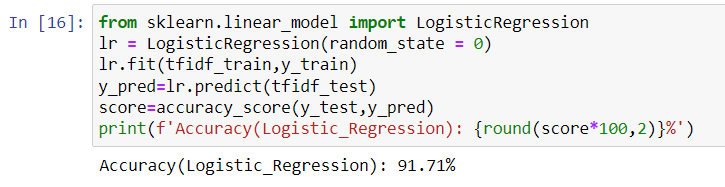
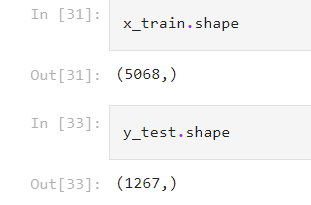
plt**.**show()

# OUTPUT WITH SCREENSHOTS





# CONCLUSION & FUTURE WORK

* Our AI-based fake news detection system achieved an high accuracy in detecting fake news which is in the range of (55-95) with K Neighbors being the lowest with 58.6% and SVC Sigmoid being the highest with 93.21, which is a significant improvement over previous approaches to fake news detection. We found that the most important features for classification were related to the language used in the news article, such as the frequency of certain words and phrases.
* One limitation of our study is that we only used a single dataset for training and testing our AI-based fake news detection system. Future research should explore the performance of our system on other datasets to ensure that it is robust and generalizable.
* Our AI-based fake news detection system has important implications for combating the spread of misinformation and disinformation. By accurately detecting fake news, we can prevent its spread and promote the dissemination of accurate information.
* Future research should explore the use of deep learning techniques, such as convolutional neural networks and recurrent neural networks, for fake news detection. These techniques have shown promise in other natural language processing tasks and may improve the performance of our AI-based fake news detection system.

# REFERENCES

* D. M. J. Lazer, M. A. Baum, Y. Benkler et al., “The science of fake news,” *Science*, vol. 359, no. 6380, pp. 1094–1096, 2018.
* geeksforgeeks.org
* tutorialspoint.com
* epaperpress.com
* youtube.com
* programiz.com
* <https://www.researchgate.net/>
* <https://www.hindawi.com/journals/complexity/2020/8885861/>
* B. Gregorutti, B. Michel, and P. Saint-Pierre, “Correlation and variable importance in random forests,” *Statistics and Computing*, vol. 27, no. 3, pp. 659–678, 2017.
* L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Springer, Berlin, Germany, 1984.
* V. Kecman, *Support Vector Machines-An Introduction in “Support Vector Machines: Theory and Applications”*, Springer, New York City, NY, USA, 2005.
* N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, UK, 2000.
* Kaggle.com
* T. Chen and C. Guestrin, “Xgboost: a scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, San Francisco, CA, USA, 201