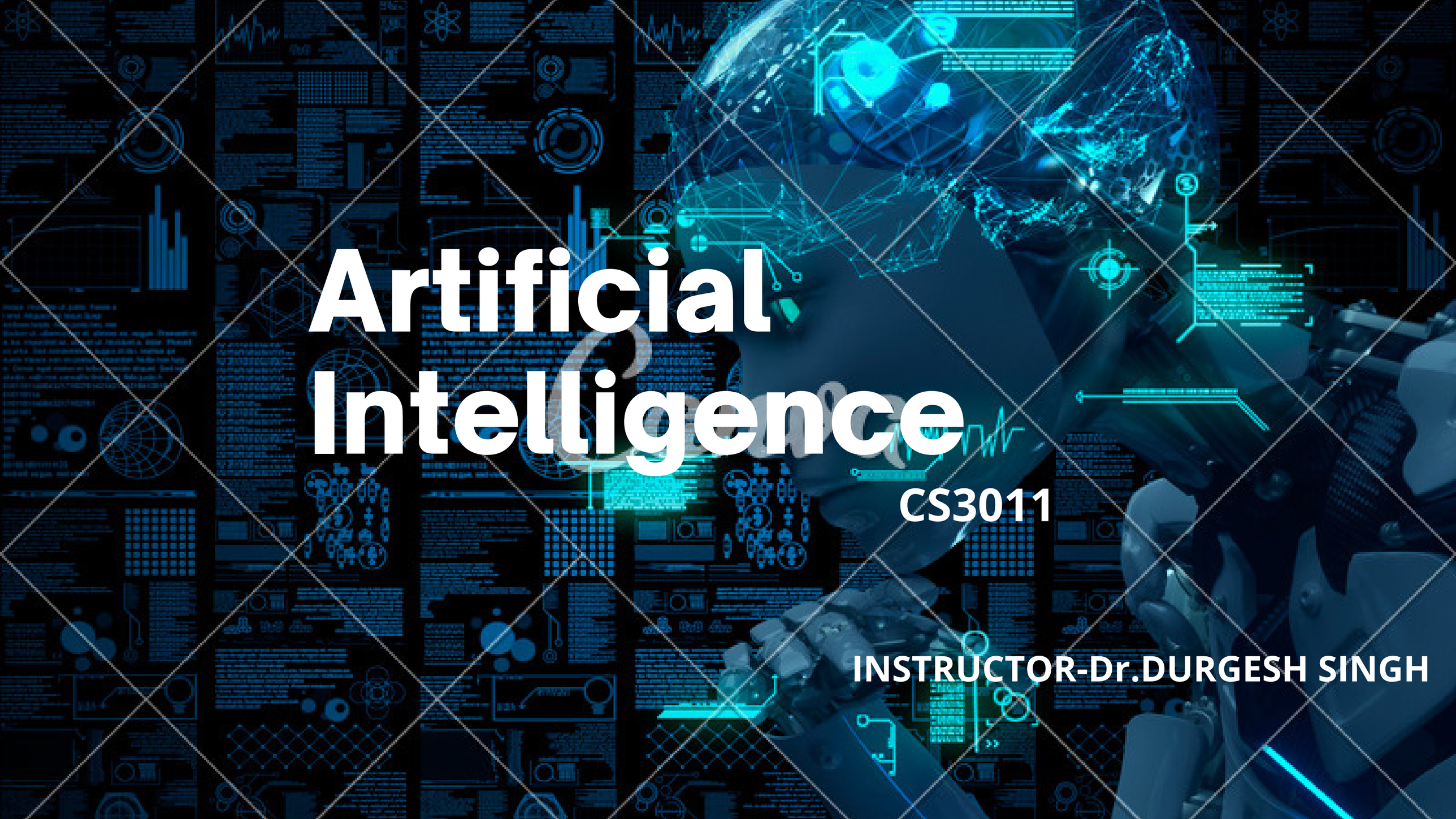


Artificial Intelligence

CS3011

INSTRUCTOR-Dr.DURGESH SINGH

Problem Statement

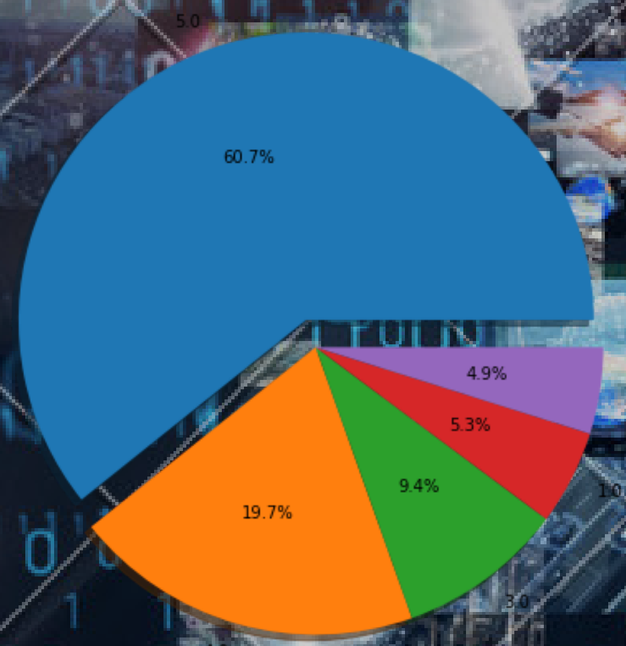
Detection of fake reviews out of a massive collection of reviews having various distinct categories like Home and Office, Sports, etc. with each review having a corresponding rating, label i.e. CG(Computer Generated Review) and OR(Original Review generated by humans) and the review text.

Main task is to detect whether a given review is fraudulent or not. If it is computer generated, it is considered fake otherwise not.

Data Set

The generated fake reviews dataset, containing 2k fake reviews and 2k real product reviews. OR = Original reviews (presumably human created and authentic); CG = Computer-generated fake reviews.

	category	rating	label	text_
0	Home_and_Kitchen_5	5	CG	love well made sturdi comfort i love veri pretti
1	Home_and_Kitchen_5	5	CG	love great upgrad origin i 've mine coupl year
2	Home_and_Kitchen_5	5	CG	thi pillow save back i love look feel pillow
3	Home_and_Kitchen_5	1	CG	miss inform use great product price i
4	Home_and_Kitchen_5	5	CG	veri nice set good qualiti we set two month



Packages used

- Numpy
- Pandas
- Matplotlib.pyplot
- Seaborn
- Warnings
- nltk
- nltk.corpus
- String
- sklearn.naive_bayes
- sklearn.feature_extraction
- sklearn.model_selection
- sklearn.ensemble
- sklearn.tree
- sklearn.linear_model
- sklearn.svc
- sklearn.neighbors

Techniques Used for Text Preprocessing

- Removing punctuation character
- Transforming text to lower case
- Eliminating stopwords
- Stemming
- Lemmatizing
- Removing digits

Transformers Used for Text Vectorization, Weighting and Normalization

- CountVectorizer Bag of Words Transformer
- TFIDF(Term Frequency-Inverse Document Frequency) Transformer

Methodology

we will outline the methodology employed for classifying the dataset using three distinct classifiers: Random Forest, Support Vector Machine (SVM), and Logistic Regression. The objective is to assess the performance of these classifiers in categorizing our dataset.

Data Preprocessing:

- Data Cleaning: We started by cleaning the dataset to handle missing values, outliers, and inconsistencies, ensuring data quality.

Data Splitting:

- We divided the dataset into training and testing sets to evaluate classifier performance. Common ratios include 75% for training and 25% for testing.

Classifier Selection

- Random Forest: This ensemble method combines multiple decision trees to enhance accuracy. We used the Random Forest classifier due to its versatility and robustness.
- Support Vector Machine (SVM): SVM is a powerful tool for binary and multiclass classification. It excels in finding optimal decision boundaries.
- Logistic Regression: A simple yet effective linear model suitable for binary classification. We included it for baseline performance comparison.

Model Training:

- We trained each classifier on the training set using the respective algorithm. This involved fitting the model to the data.

Model Evaluation:

- To assess classifier performance, we used a range of evaluation metrics such as accuracy, precision, recall..

Results and Analysis:

- We present the results in terms of model accuracy, precision, recall..
- Comparative analysis among classifiers helps us identify the most suitable algorithm for our dataset.

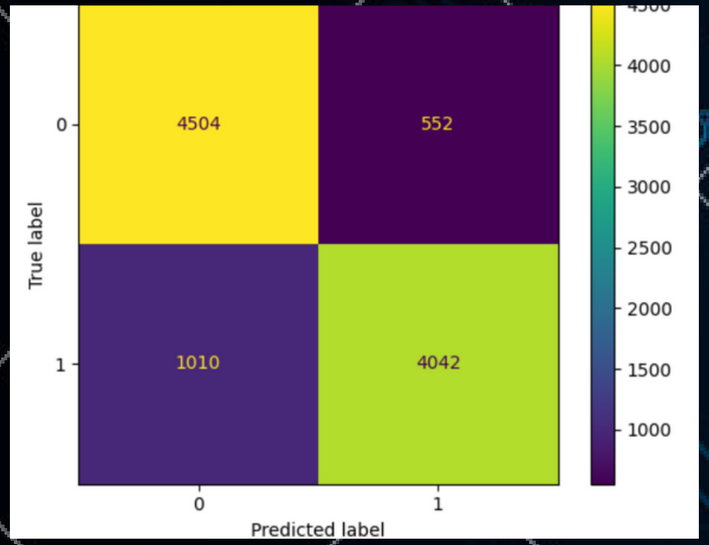
Model Deployment:

- If one classifier outperforms the others significantly, we consider deploying it for real-world applications.

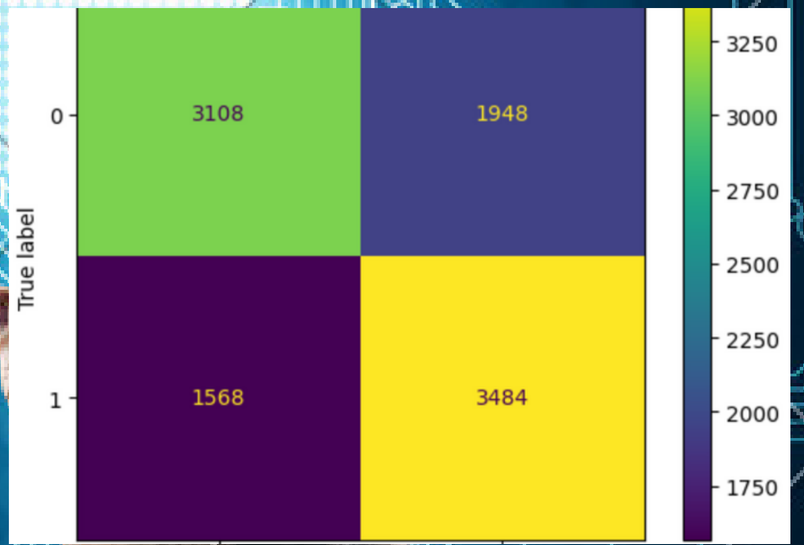
Conclusion:

- The methodology employed in this classification task allows for a systematic assessment of three classifiers' performance, aiding in data-driven decision-making and model selection.

Results


4504
552
1010
4042
0
1
True label
Predicted label
4000
3500
3000
2500
2000
1500
1000


4411
645
502
4550
0
1
True label
4000
3500
3000
2500
2000
1500
1000


3108
1948
1568
3484
0
1
True label
3250
3000
2750
2500
2250
2000
1750

```
print('accuracy of the using logistic regression model:',str(np.round(accuracy_score(y_test,logisticRegression)*100,2)
```

```
accuracy of the model: 65.22%
```

```
print('accuracy of the model:',str(np.round(accuracy_score(y_test,supportVectorClassifier)*100,2)) + '%')
```

```
accuracy of the model: 88.65%
```

```
'Accuracy of the model: ',str(np.round(accuracy_score(y_test,randomForestClassifier)*100,2)) + '%'
```

```
cy of the model:  84.55%
```


Thank You!