INTERNSHIP REPORT

A report submitted in partial fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING(AIR)

By

Aakash R P

UID: 258792

Under Supervision of

Mr. Ashok Nair,

US Technology International Private Limited,

Thiruvananthapuram.

(Duration: 14th August, 2023 to 30th November, 2023)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VELLORE INSTITUTE OF TECHNOLOGY, CHENNAI

ACKNOWLEDGEMENT

First, I would like to thank Mr.Ashok Nair, HR, Head, of G&A - Innovation, US Technology International Private Limited, Trivandrum for giving me the opportunity to do an internship within the organization.

I also would like all the people that worked along with me at UST, with their patience and openness they created an enjoyable working environment. It is indeed with a great sense of pleasure and immense sense of gratitude that I acknowledge the help of these individuals. I am highly indebted to my incharge head Mrs. Gopika Bindhu, for the facilities provided to accomplish this internship.

I would like to thank my mentors Mr. Anandhu Bhaskar, for their constructive criticism throughout my internship. I am extremely great full to my USTmembers and friends who helped me in successful completion of this internship.

Aakash R P

(258792)

ABSTRACT

This internship report investigates the convergence of Machine Learning (ML), Natural Language Processing (NLP), and 2D to 3D rendering, with a focus on practical applications across diverse industries. The internship, situated in the context of cutting-edge technological advancements, involves hands-on engagement with ML algorithms and NLP techniques to address complex issues in data analysis and language understanding. A significant portion of the internship is dedicated to the challenges and advancements in 2D to 3D rendering, exploring the transformation of two-dimensional representations into immersive three-dimensional visualizations.

The report highlights the potential of integrating ML and NLP with 2D to 3D rendering, showcasing the development of intelligent, interactive systems capable of interpreting natural language inputs and generating immersive 3D visualizations. Case studies illustrate practical implementations across virtual reality, augmented reality, and computer-aided design. Overall, the findings emphasize the transformative impact of these integrated technologies, offering innovative solutions with broad applications. The report concludes with reflections on encountered challenges, lessons learned, and recommendations for future research in this burgeoning field.

Organization Information:

UST is a global digital transformation solutions provider. For more than 20 years, UST has worked side by side with the world's best companies to make a real impact through transformation. Poweredby technology, inspired by people and led by purpose, UST partners with their clients from design tooperation. With deep domain expertise and a future-proof philosophy, UST embeds innovation and agility into their clients' organizations. With over 30,000 employees in 30 countries, UST builds for boundless impact—touching billions of lives in the process.

Methodologies:

We follow a structured methodology for our projects which starts from designing the solution to the implementation phase. Well planned Project reduces the time to deliver the project and any additional ad-hoc costs to our clients, hence we dedicate majority of our time understanding our clients business and gather requirements. This ground up approach helps us deliver not only the solution to our clients but also add value to your investments.

INTRODUCTION

This internship report delves into a comprehensive exploration of the dynamic realms of Machine Learning (ML), Natural Language Processing (NLP), and the transformative field of 2D to 3D rendering. Undertaken in the context of advancing technological landscapes, this internship aimed to contribute valuable insights and hands-on experience at the intersection of these cutting-edge technologies. The report offers an in-depth analysis of the theoretical foundations, practical applications, and collaborative synergies of ML, NLP, and 2D to 3D rendering, highlighting the interdisciplinary nature of this immersive learning experience.

As technology continues to evolve, the integration of ML and NLP has become pivotal in enhancing our ability to process, understand, and derive meaningful insights from vast datasets and natural language inputs. Simultaneously, the fascinating domain of 2D to 3D rendering has evolved to redefine visual experiences, with applications ranging from virtual reality environments to computer-aided design. This internship sought to bridge these domains, exploring how the convergence of ML, NLP, and 2D to 3D rendering can unlock new dimensions of innovation and efficiency in diverse industries.

The subsequent sections of this report will provide a detailed account of the theoretical foundations and practical applications encountered during the internship. From delving into the intricacies of ML algorithms and NLP techniques to navigating the challenges and advancements in 2D to 3D rendering, this report aims to offer a comprehensive understanding of the synergies and transformative potential of these technologies. Through this internship, the goal was not only to gain practical skills but also to contribute to the ongoing discourse in the fields of ML, NLP, and 2D to 3D rendering, paving the way for future innovations and applications in the ever-evolving landscape of technology.

Objectives

Plot Prediction using Machine Learning

The primary objective of the project on plot prediction using machine learning is to develop and implement accurate predictive models that can effectively analyze diverse datasets and predict plot prices. Through the application of machine learning algorithms, the project aims to uncover intricate patterns and relationships within the data, considering various features such as location, size, economic indicators, and market trends. The overarching goal is to enhance the precision of real estate valuation, providing stakeholders with valuable insights for informed decision-making, optimizing pricing strategies, and identifying investment opportunities in the dynamic real estate market.

Sentiment Analysis using NLP

The main objective of the sentiment analysis project using Natural Language Processing (NLP) is to design and implement robust models that can accurately discern and categorize sentiments expressed in textual data. By leveraging NLP techniques such as tokenization, sentiment lexicons, and machine learning algorithms, the project aims to develop a nuanced understanding of the emotional tone within diverse contexts. The overarching goal is to provide valuable insights into public opinion, customer feedback, or user sentiment, enabling informed decision-making for businesses and organizations. This project seeks to demonstrate the practical application of NLP in extracting and analyzing sentiments, thereby contributing to advancements in text analysis and enhancing the ability to respond effectively to varying emotional expressions in textual content.

2D TO 3D Rendering using Neural Radiance Field

The primary objective of the 2D to 3D rendering project using Neural Radiance Fields (NeRF) is to employ state-of-the-art techniques to transform two-dimensional representations into highly detailed and immersive three-dimensional renderings. Specifically, the project aims to implement NeRF-based models, capitalizing on their ability to capture intricate scene details and lighting effects. Through this, the project seeks to advance the quality and realism of 3D renderings, particularly in applications such as virtual reality, augmented reality, and computer-aided design. The overarching goal is to explore and demonstrate the capabilities of NeRF in revolutionizing the process of 2D to 3D rendering, contributing to advancements in realistic scene reconstruction and visualization technologies.

PROJECT DEVELOPMENT JOURNEY

Preliminary Learning Phase

Understanding Machine Learning

In the project's early phases, I initiated my journey into the realm of machine learning by delving into Python. Focused on gaining practical insights and expertise, I embarked on a project centered around predicting plot prices. This hands-on experience significantly contributed to my understanding of machine learning principles and methodologies.

1.Prediction of Plot Prices using Machine Learning

Steps Involved :

DATA COLLECTION : Collecting data from websites here i have taken my dataset from Kaggle

DATA PRE-PROCESSING: Clean the dataset by handling missing values, outliers, and inconsistencies.Here have used r studios and python to do data preprocessing.

DATA SPLITTING: Split the dataset into training and testing sets to assess the model's performance. A common split ratio is 80% for training and 20% for testing, but this can vary depending on the dataset size.

MODEL SELECTION: Choose a suitable machine learning algorithm for regression tasks. Common choices include linear regression, decision trees, random forests, gradient boosting, and neural networks.Here i have used random forest regression.

MODEL TRAINING: Train the selected model using the training dataset. The model learns the relationships between the features and the target variable (house prices).

MODEL EVALUATION: Evaluate the model and check its accuracy.If the accuracy is less try to improve the accuracy.

CODE:

```r
install.packages("dplyr")
install.packages("data.table")
library(data.table)
library(dplyr)
data=Bangalore
data
#Checking for missing values
missing_values=is.na(data)
missing_values
missing_count <- colSums(is.na(data))
print(missing_count)
#No missing values were found
```

```
In [1]: import pandas as pd
        import numpy as np

In [2]: df=pd.read_csv("Bangalore.csv")

In [3]: df
```

Out[3]:

	Price	Area	Location	No. of Bedrooms	Resale	MaintenanceStaff	Gymnasium	SwimmingPool	LandscapedGardens	JoggingTrack	...	LiftAvailable	Bl
0	30000000	3340	JP Nagar Phase 1	4	0	1	1	1	1	1	...	1	
1	7888000	1045	Dasarahalli on Tumkur Road	2	0	0	1	1	1	1	...	1	
2	4866000	1179	Kannur on Thanisandra Main Road	2	0	0	1	1	1	1	...	1	
3	8358000	1675	Doddanekundi	3	0	0	0	0	0	0	...	1	
4	6845000	1670	Kengeri	3	0	1	1	1	1	1	...	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
6202	5364000	590	Chandapura	1	0	9	9	9	9	9	...	9	
6203	8716000	1179	Kasavanahalli	2	0	9	9	9	9	9	...	9	
6204	7373000	1143	Kasavanahalli	2	0	9	9	9	9	9	...	9	
6205	4985000	1680	Kasavanahalli	3	0	9	9	9	9	9	...	9	
6206	10900000	1162	Kasavanahalli	2	0	9	9	9	9	9	...	9	

6207 rows × 40 columns

```
In [4]: print(df['Location'].unique())

['JP Nagar Phase 1' 'Dasarahalli on Tumkur Road'
 'Kannur on Thanisandra Main Road' 'Doddanekundi' 'Kengeri' 'Horamavu'
 'Thanisandra' 'Ramamurthy Nagar' 'Whitefield Hope Farm Junction'
 'Electronic City Phase 1' 'Yelahanka' 'Anjanapura' 'Jalahalli'
 'Kasavanahalli' 'Bommasandra' 'Bellandur' 'RR Nagar' 'Begur' 'Hosa Road'
 'Sahakar Nagar' 'Kadugodi' 'Jakkur' 'Jigani' 'Krishnarajapura'
 'Brookefield' 'Banashankari' 'Nelamangala' 'Attibele' 'Banaswadi'
 'Kodigehalli' 'ITPL' 'Uttarahalli Hobli'
 'Chikkagubbi on Hennur Main Road' 'Varthur' 'Vidyaranyapura'
 'Electronic City Phase 2' 'J. P. Nagar' 'K. Chudahalli' 'Narayanaghatta'
 'Anekal City' 'Sarjapur' 'Koramangala' 'Hebbal' 'Budigere Cross'
 'Bommanahalli' 'Electronics City' 'Chikkalasandra' 'Kogilu'
 'Nayandahalli' 'Bilekahalli' 'Muneshwara Nagar' 'Junnasandra'
 'Narayanapura on Hennur Main Road' 'Kothanur' 'Kadugodi Industrial Area'
 'Sarjapur Road Wipro To Railway Crossing' 'RMV Extension Stage 2' 'Kudlu'
 'Talaghattapura' 'Kumbalgodu' 'Carmelaram' 'Uttarahalli'
 'Anagalapura Near Hennur Main Road' 'Avalahalli Off Sarjapur Road'
 'R T  Nagar' 'JP Nagar Phase 7' 'Subramanyapura' 'JP Nagar Phase 4'
 'JP Nagar Phase 8' 'Amruthahalli' 'Nagarbhavi' 'Chandapura' 'Marsur'
 'JP Nagar Phase 3' 'JP Nagar Phase 9' 'Gottigere'
 'Kanakapura Road Beyond Nice Ring Road' 'Harlur' 'Konanakunte'
 'Richmond Town' 'Jayanagar' 'Domlur' 'Devanahalli' 'Hulimavu'
 'Kumaraswamy Layout' 'Bikasipura' 'Singasandra' 'JP Nagar Phase 6'
 'Sanjaynagar' 'CV Raman Nagar' 'Padmanabhanagar' 'Hennur' 'KPC Layout'
 'R.K. Hegde Nagar' 'Kannamangala' 'Yerthiganahalli' 'Badamanavarthekaval'
 'Kanakapura' 'Bannerughatta' 'BTM Layout'
 'Kuvempu Layout on Hennur Main Road' 'Marathahalli' 'Rajajinagar'
 'Whitefield' 'RMV' 'Bileshivale' 'Nagawara' 'HSR Layout'
 'Anjanapura Township' 'Hurulichikanahalli' 'Devarachikkana Halli'
 'Sampangiram Nagar' 'Armane Nagar' 'Basavanagudi' 'Yeshwantpur'
 'Battarahalli' 'K C Krishna Reddy Layout' 'Adugodi' 'Kammanahalli'
```

In [5]:
```
#Location numbers
df['Location'].value_counts()
```

Out[5]:
```
Electronic City Phase 2    232
RR Nagar                   217
Begur                      186
Varthur                    168
Kumaraswamy Layout         154
                          ...
Domlur Layout                1
Shivaji Nagar                1
Ashwathkatte Road            1
Kambipura                    1
HAL                          1
Name: Location, Length: 302, dtype: int64
```

In [6]:
```
#Using one hot encoder convert string categorical variable to numerical variable
one_hot_encoded_data = pd.get_dummies(df, columns = ['Location'])
print(one_hot_encoded_data)
```

```
          Price  Area  No. of Bedrooms  Resale  MaintenanceStaff  Gymnasium  \
0      30000000  3340                4       0                 1          1
1       7888000  1045                2       0                 0          1
2       4866000  1179                2       0                 0          1
3       8358000  1675                3       0                 0          0
4       6845000  1670                3       0                 1          1
...         ...   ...              ...     ...               ...        ...
6202    5364000   590                1       0                 9          9
6203    8716000  1179                2       0                 9          9
6204    7373000  1143                2       0                 9          9
6205    4985000  1680                3       0                 9          9
6206   10900000  1162                2       0                 9          9

      SwimmingPool  LandscapedGardens  JoggingTrack  RainWaterHarvesting  ...  \
0                1                  1             1                    1  ...
1                1                  1             1                    1  ...
```

In [7]:
```
df=one_hot_encoded_data
df
```

Out[7]:

	Price	Area	No. of Bedrooms	Resale	MaintenanceStaff	Gymnasium	SwimmingPool	LandscapedGardens	JoggingTrack	RainWaterHarvesting	...	Location
0	30000000	3340	4	0	1	1	1	1	1	1	...	
1	7888000	1045	2	0	0	1	1	1	1	1	...	
2	4866000	1179	2	0	0	1	1	1	1	1	...	
3	8358000	1675	3	0	0	0	0	0	0	0	...	
4	6845000	1670	3	0	1	1	1	1	1	1	...	
...	...	...	...	...	...	...	...	...	...	...	...	...
6202	5364000	590	1	0	9	9	9	9	9	9	...	
6203	8716000	1179	2	0	9	9	9	9	9	9	...	
6204	7373000	1143	2	0	9	9	9	9	9	9	...	
6205	4985000	1680	3	0	9	9	9	9	9	9	...	
6206	10900000	1162	2	0	9	9	9	9	9	9	...	

6207 rows × 341 columns

In [8]:
```
mask = df.apply(lambda row: any(row == 9), axis=1)
df = df[~mask]
```

In [9]:
```
df
df.to_csv('/home/user/Documents/bangloreclean.csv')
```

In [10]:
```
from sklearn.preprocessing import OneHotEncoder
#Create X(featrue matrix)
X=df.drop("Price",axis=1)
#Create Y(labels)
Y=df["Price"]
```

In [11]:
```
# 3.Fit the model to the training data
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=0)
```

In [12]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [13]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components = 4)
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
```

In [14]:
```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators = 250, random_state = 0)
regressor.fit(X_train, y_train)
```

Out[14]:
```
RandomForestRegressor
RandomForestRegressor(n_estimators=250, random_state=0)
```

```
In [15]: regressor.score(X_train,y_train)
Out[15]: 0.9755961615626692

In [16]: regressor.score(X_test,y_test)
Out[16]: 0.9450697151838621
```

```
In [17]: #5.Improve model

np.random.seed(42)
for i in range(250,400,10):
    print(f"Truing model with {i} estimators...")
    clf=RandomForestRegressor(n_estimators=i).fit(X_train,y_train)
    print(f"Accuracy: ",clf.score(X_test,y_test))

Truing model with 250 estimators...
Accuracy:  0.9416167064114455
Truing model with 260 estimators...
Accuracy:  0.9435980379501294
Truing model with 270 estimators...
Accuracy:  0.943619515116332
Truing model with 280 estimators...
Accuracy:  0.9455764434148505
Truing model with 290 estimators...
Accuracy:  0.9462148850720317
Truing model with 300 estimators...
Accuracy:  0.9430786718604119
Truing model with 310 estimators...
Accuracy:  0.946006934437218
Truing model with 320 estimators...
Accuracy:  0.9451445823697107
Truing model with 330 estimators...
Accuracy:  0.9448819745112254
Truing model with 340 estimators...
Accuracy:  0.945704510681658
Truing model with 350 estimators...
Accuracy:  0.9421224763225744
Truing model with 360 estimators...
Accuracy:  0.9438691665190989
Truing model with 370 estimators...
Accuracy:  0.9447015431697064
Truing model with 380 estimators...
Accuracy:  0.94492139666988
Truing model with 390 estimators...
Accuracy:  0.946404977356727
```

OUTPUT:

```
In [18]: X1=[["3701","4","0","0","1","1","0","1","1","1","0","0","1","0","1","0","1","1","1","1","1","1","0","0","0","0","0","
         ]]
         blah = sc.transform(X1)
         data=pca.transform(blah)
         regressor.predict(data)

         /home/user/Downloads/PREFIX=/home/user/anaconda3/lib/python3.11/site-packages/sklearn/base.py:464: UserWarning: X do
         es not have valid feature names, but StandardScaler was fitted with feature names
           warnings.warn(

Out[18]: array([36420535.992])
```

CONCLUSION

To summarize, the project I have done on plot pricing prediction using machine learning has produced an effective tool for the real estate sector. House prices can be precisely estimated by using machine learning algorithms and historical data analysis. The model is useful, but it has limits and needs to be updated frequently. This project might give the real estate business a competitive edge and signal a significant advancement in the field of real estate price prediction.

2.SENTIMENT ANALYSIS USING NLP

Steps Involved:

1. **Data Collection :** Gather textual data from various sources, such as social media, customer reviews, news articles, or any text that contains sentiment.
Define the target text or document that you want to analyze for sentiment.

2. **Data Pre-Processing :**

- Convert text to lowercase to ensure consistency.
- Remove special characters, punctuation, and irrelevant symbols
- Tokenize the text into words or subword units (e.g., word segmentation).
- Remove stopwords (common words like "the," "and," "is") that do not carry sentiment information.
- Stemming or Lemmatization: Reduce words to their base or root form. For example, "running" becomes "run."
- Handle negation: Identify and flag negations like "not happy" to properly represent sentiment.

3. **Feature Extraction :** Convert text data into numerical representations (vectors) that machine learning models can understand. Common methods include:
Bag of Words (BoW): Represent each document as a vector of word counts.

4**. Model Selection & Training :**

- Rule-based models: Define rules to assign sentiment based on keywords and patterns.
- Choose an appropriate sentiment analysis model or algorithm.
- Machine learning models: Use algorithms such as Naive Bayes, Support Vector Machines (SVM), or deep learning methods (e.g., Recurrent Neural Networks, Transformers).
- Train the selected model on labeled data, where each text is associated with a sentiment label (e.g., positive, negative, neutral).

5**. Model Evaluation :** Assess the model's performance using metrics like accuracy, precision, recall, F1-score, and confusion matrices.
Cross-validation and hyperparameter tuning may be performed to optimize the model.

6**. Sentiment Evaluation :** Apply the trained model to unseen text data to predict sentiment labels.Generate sentiment scores or labels for each piece of text (e.g., positive, negative, neutral) based on model output.

CODE:

```
importing the dataset
cleaning the texts
creating the bag of words model
splitting the dataset into the training set and test set
training the naive bayes model on the taining set
predicting the test set results
making the confusion matrix
```

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
dataset = pd.read_csv('Restaurant_Reviews.tsv', delimiter = '\t', quoting = 3)
```

In [3]:
```
dataset
```

Out[3]:

	Review	Liked
0	Wow... Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture was just nasty.	0
3	Stopped by during the late May bank holiday of...	1
4	The selection on the menu was great and so wer...	1
...	...	...
995	I think food should have flavor and texture an...	0
996	Appetite instantly gone.	0
997	Overall I was not impressed and would not go b...	0
998	The whole experience was underwhelming, and I ...	0
999	Then, as if I hadn't wasted enough of my life ...	0

In [4]:
```
# Cleaning the texts
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
corpus = []
for i in range(0, 1000):
  review = re.sub('[^a-zA-Z]', ' ', dataset['Review'][i])
  review = review.lower()
  review = review.split()
  ps = PorterStemmer()
  all_stopwords = stopwords.words('english')
  all_stopwords.remove('not')
  review = [ps.stem(word) for word in review if not word in set(all_stopwords)]
  review = ' '.join(review)
  corpus.append(review)
print(corpus)
```

```
[nltk_data] Downloading package stopwords to /home/user/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
['wow love place', 'crust not good', 'not tasti textur nasti', 'stop late may bank holiday rick steve recommend love
', 'select menu great price', 'get angri want damn pho', 'honeslti tast fresh', 'potato like rubber could tell made
ahead time kept warmer', 'fri great', 'great touch', 'servic prompt', 'would not go back', 'cashier care ever say st
ill end wayyy overpr', 'tri cape cod ravoli chicken cranberri mmmm', 'disgust pretti sure human hair', 'shock sign i
ndic cash', 'highli recommend', 'waitress littl slow servic', 'place not worth time let alon vega', 'not like', 'bur
ritto blah', 'food amaz', 'servic also cute', 'could care less interior beauti', 'perform', 'right red velvet cake o
hhh stuff good', 'never brought salad ask', 'hole wall great mexican street taco friendli staff', 'took hour get foo
d tabl restaur food luke warm sever run around like total overwhelm', 'worst salmon sashimi', 'also combo like burge
r fri beer decent deal', 'like final blow', 'found place accid could not happier', 'seem like good quick place grab
bite familiar pub food favor look elsewher', 'overal like place lot', 'redeem qualiti restaur inexpens', 'ampl porti
on good price', 'poor servic waiter made feel like stupid everi time came tabl', 'first visit hiro delight', 'servic
suck', 'shrimp tender moist', 'not deal good enough would drag establish', 'hard judg whether side good gross melt s
tyrofoam want eat fear get sick', 'posit note server attent provid great servic', 'frozen puck disgust worst peopl b
```

In [5]:
```
# Creating the Bag of Words model
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features = 124)
X = cv.fit_transform(corpus).toarray()
y = dataset.iloc[:, -1].values
```

In [6]:
```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)
```

In [7]:
```
# Training the Naive Bayes model on the Training set
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)
```

Out[7]:
```
GaussianNB()
```

OUTPUT:

```
In [10]: df=pd.read_csv("amazon_product_reviews15.csv")
         df
```

Out[10]:

	review_title	review_body
0	4.0 out of 5 stars Average gaming laptop.	It has various modes on which it operates to s...
1	3.0 out of 5 stars Battery backup very less	Battery backup is less.I was charged 100% and ...
2	4.0 out of 5 stars Experience	Laptop is good and smooth it's working nicely ...
3	4.0 out of 5 stars Review	Seller only send laptop and charging adaptor n...
4	3.0 out of 5 stars Laptop has heating issue an...	Not happy with battery power and have heating ...
5	5.0 out of 5 stars Best laptop	Great display with 17.3 inchBest laptop for Bi...
6	5.0 out of 5 stars Awesome looks	TUF stickers are not available inside the box
7	5.0 out of 5 stars Good	Good laptop. Good performance
8	5.0 out of 5 stars Very Nice	Very fast, nice feature
9	2.0 out of 5 stars Armour crate software has i...	Many times armour crate software stops fan eve...

```
In [11]: corpus = []
         for i in range(len(df)):
           review = re.sub('[^a-zA-Z]', ' ', df['review_body'][i])
           review = review.lower()
           review = review.split()
           ps = PorterStemmer()
           all_stopwords = stopwords.words('english')
           all_stopwords.remove('not')
           review = [ps.stem(word) for word in review if not word in set(all_stopwords)]
           review = ' '.join(review)
           corpus.append(review)
```

```
In [12]: # Creating the Bag of Words model
         from sklearn.feature_extraction.text import CountVectorizer
         cv = CountVectorizer(max_features = 1500)
         X = cv.fit_transform(corpus).toarray()
         classifier.predict(X)
```

```
Out[12]: array([1, 0, 1, 0, 1, 1, 1, 0, 1, 0])
```

CONCLUSION :

In summary, the sentiment analysis project using Natural Language Processing (NLP) has successfully delved into the intricate task of interpreting human emotions from textual data. Through the implementation of NLP techniques and machine learning models, the project demonstrated its effectiveness in classifying and analyzing sentiments, overcoming challenges related to linguistic nuances. The findings underscore the practical relevance of sentiment analysis, particularly in areas like social media monitoring and customer feedback analysis. The adaptability of NLP models and the continuous evolution of the field position sentiment analysis as a valuable tool for informed decision-making across diverse domains.

3. 2D to 3D Rendering Using Neural Radiance Field

NeRF is a fully-connected neural network that generates novel views of complex 3D scenes based on a partial set of 2D images.

Benefits of NeRF:

- Achieves photo-realistic rendering
- Allows fast reconstruction
- Provides compact modeling

Reconstructing 3D Scenes

- NeRF allows reconstruction of 3D scenes from 2D images
- Enables synthesis of new views of the scene

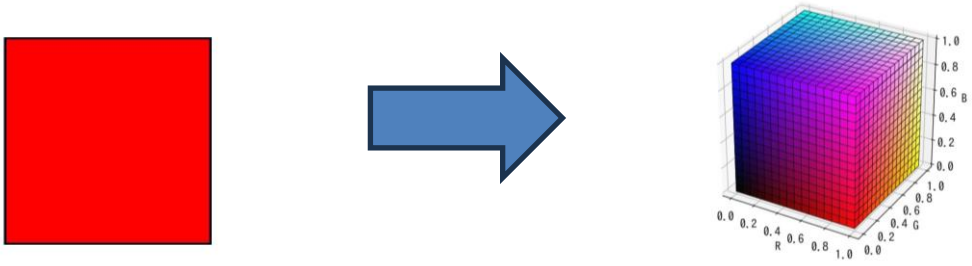
Future Possibilities

- Exploration of potential applications in various fields
- The impact of advancements in NeRF technology

2D to 3D conversion using Python

In [16]:
```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Step 1: Load a 2D image
image_path = "C:/Users/aakas/Documents/square.jpg"
image = plt.imread(image_path)

# Step 2: Convert the image to grayscale
gray_image = np.mean(image, axis=-1)

# Step 3: Extrude the image based on pixel intensity with a lower factor
intensity_factor = 1.0  # Adjust this value to increase or decrease the intensity
extruded_image = intensity_factor * gray_image

# Step 4: Create a 3D meshgrid
x, y = np.meshgrid(np.arange(extruded_image.shape[1]), np.arange(extruded_image.shape[0]))
z = extruded_image

# Step 5: Create a 3D plot
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

# Step 6: Plot the 3D surfaces with different colors
ax.plot_surface(x, y, z, cmap='viridis', edgecolor='k', alpha=0.8)  # Side surface in viridis colormap

# Plot top surface in blue
ax.plot_surface(x, y, z.max(), color='blue', alpha=0.5)

# Plot bottom surface in red
ax.plot_surface(x, y, 0, color='red', alpha=0.5)

# Step 7: Customize the plot
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title('Multicolored 3D Cube from 2D Image')

# Adjust aspect ratio for a proper cube representation
ax.set_box_aspect([np.ptp(coord) for coord in [x, y, z]])

# Step 8: Show the plot
plt.show()
```

OUTPUT:



2D to 3D rendering using Blender:

Transforming a 2D image into a 3D rendering using Blender is a multi-step process that begins with the importation of the 2D image, usually in formats like PNG, JPEG, or SVG. After importing, the image serves as a reference backdrop, aiding in the creation of corresponding 3D geometry through Blender's modeling tools. Essential modeling techniques involve extrusion, where flat surfaces are extended into three-dimensional forms, and scaling to adjust proportions. The meticulous application of these tools enables the recreation of the elements present in the 2D image within a 3D space.

- Once the 3D geometry is established, the next phase involves enhancing the realism of the scene. This is achieved by applying textures and materials to the 3D model. Blender provides a Material Editor where artists can define the visual characteristics of surfaces, including color, reflectivity, and transparency. Lighting plays a crucial role in creating a convincing 3D environment, and Blender offers various light sources, such as point lights and spotlights, allowing artists to experiment with different setups to achieve the desired mood and atmosphere. Additionally, a careful camera setup is necessary to capture the scene from the preferred perspective, with considerations for focal length and depth of field.
- The final stages of the process involve rendering and potential post-processing. Rendering settings, including resolution and output format, are configured in Blender before initiating the rendering process. For those seeking animated outcomes, Blender's animation tools can be employed to keyframe movements and transformations. Post-processing steps, though optional, allow for further refinement of the final output. Blender's Compositor provides a platform for color correction, effects, and additional adjustments. Throughout the entire process, Blender's open-source nature and extensive community support offer a wealth of resources and tutorials to guide artists through the intricacies of 2D to 3D rendering.


U
S T


PS2

REFERENCES

WEBLINKS

https://www.geeksforgeeks.org/machine-learning/

https://www.kaggle.com/

https://www.udemy.com/course/complete-machine-learning-and-data-science-zero-to-mastery/?kw=machine+lea&src=sac

https://youtu.be/W-KZDpaMl3o?si=PQXEojGeetYMXeZC

https://youtube.com/watch?v=8wQGbmLulBw&si=H6eeHWwB5hs8Pdxji

https://youtube.com/playlist?list=PLjEaoINr3zgEPv5y--4MKpciLaoQYZB1Z&si=kGKxdHtKsWV7TMxC

https://www.udemy.com/course/deeplearning/