**Objective:**

As the requirement for completion of ‘***Python with Data Science*** ‘course of this institution the objective is to create an end-to-end project of Data Science from scratch. As per that, I have a created a ‘**Nepali News Classifier Model**’ that will classify the news article.

In this project I try different ML algorithm and best accuracy score is obtain by **Random Forest** with accuracy of 92.88%.

**Phase 1:**

Now below is the step-by-step process of the project:

1. **Dataset description:**

**Data Collection**:

Data is collected from <https://github.com/amitness/ml-datasets> where we can find the lots of Nepal datasets. Under Text there is a corpus of ‘16NepaliNews Corpus’. This project uses this dataset.

**Data Description:**

Data is in separate folder for train and test dataset. I use train dataset folder and convert to csv file and build process from the Google Collab. While reading csv file, there are some bad rows which isn’t convertible so I skip that article and now the total news articles are 12198.

Dataset has 2 rows. One is article named as ***text*** and another is ***target*** from 0 to 15.

Data has major sixteen categories:

  0: "Auto",

    1: "Bank",

    2: "Blog",

    3: "Business Interview",

    4: "Economy",

    5: "Education",

    6: "Employment",

    7: "Entertainment",

    8: "Interview",

    9: "Literature",

    10: "National News",

    11: "Opinion",

    12: "Sports",

    13: "Technology",

    14: "Tourism",

    15: "World"

1. **Data Cleaning and Transformation Steps**:

**Handle Missing Values**: Since there is no missing value, we don’t have to handle it.

**Handle Duplicated value**: There are 3 duplicate values which we drop from the Dataset.

**Outliers Detection and Treatment**: Since, it is a NLP project, news can’t be an outlier.

1. **Exploratory Data Analysis (EDA):**

While observing below is the weights of each news articles in datasets:

|  |  |
| --- | --- |
| Category | Proportion (%) |
| National News | 51.908633 |
| Sports | 15.919473 |
| Economy | 8.269454 |
| Entertainment | 8.176539 |
| Opinion | 4.103755 |
| Bank | 2.439024 |
| Tourism | 1.486643 |
| World | 1.471158 |
| Blog | 1.455672 |
| Employment | 1.068525 |
| Business Interview | 0.983353 |
| Technology | 0.758808 |
| Auto | 0.658149 |
| Interview | 0.603949 |
| Education | 0.588463 |
| Literature | 0.108401 |

We can simply observe that the dataset is ***imbalance*** as the National News is 52%, Sports 16%, Economy and Entertainment 8% and rest of other articles are below 5%.

So, I think that keeping these categories will increase the complexity, so I merge the categories and make only 5 categories as below:

1. **National News** – 52%
2. **Sports**- 16%
3. **Economy + Bank + Business Interview** – Business/Economy- 12%
4. **Entertainment**- 8%
5. **Others** - Tourism + World + Blog + Employment + Opinion + Technology + Interview +Education +Literature – 12%
6. **Data Transformation:**

**Target column:**

For transforming a ***target*** column, I use ***LabelEncoder*** to transform the target features from 0 to 4.

sports - 4

others -3

National News -2

Entertainment -1

Business/Economy-0

**Text Preprocessing:**

1. Removing HTML tags
2. Remove URLs
3. Remove punctuations
4. Removing Special Characters
5. Removing stop words
6. Applying Nepali Stemmer

We apply the Text Preprocessing techniques before passing to the model.

**Word Cloud:**

We are creating the word cloud in order to know the which words are repeated the most.

**Applying SMOTE:**

Before building the model, we apply the SMOTE since our target feature is imbalance. Means, 52% articles are of National News, 16% Sports so our model will be more bias towards this news and predict incorrectly. So, by apply SMOTE, we will generate the fake data by SMOTE which will helps to balance the minority articles. So, our model will not be bias.

1. **Model Building:**

**TF-IDF vectorizer:**

We apply TF-IDF vectorizer method in order to convert the articles into meaningful vectors. It uses vocabulary from your training dataset, so it uses only those words it has seen in training time. So new words that have never been appeared in training time are ignore. Also, this vectorizer doesn’t care about semantic relationship of sentence.

**ML Models:**

I try below ML models in this project:

1. Gaussian Naïve Bayes
2. Multinomial Naïve Bayes
3. Bernoulli Naïve Bayes
4. Support Vector Machine Classifier
5. Random Forest Classifier
6. XG Boost
7. Logistic Regression

For hyper parameter tuning, we use the RandomizedSearchCV which will random select the parameters do cross validation. Since, applying GridSearchCV takes longer time so I apply RandomizedSearchCV method for hyper parameter tuning.

1. **Model Dumps:**

It means we will only download the model by pickling so that we can use it while building the application. Here we dump vector i.e., TF-IDF and random forest models

**Phase 2:**

**Application Building:**

For application, I use streamlit and vs code platform for the application. Below files are created in the application process.

.gitignore

docs

app.py

readme.md

requirements.txt

nepali\_news\_classifier.ipynb(for pushing to github)

random\_model.pkl

For the details explanation check the **docs** folder and readme file.

Now, I push the code to the github.

**Phase 3:**

I make account in streamlit cloud. Then create app by linking github. Then deploy it.

Please check in this link: <https://share.streamlit.io/>

Github Repository link: <https://github.com/itsmedipen/nepali_news_classifier>

Limitation:

Since we use TF-IDF as a vectorizer, it use vocabulary from your training dataset, so it uses only that words it has seen in training time. So new words that have never been appeared in training time are ignore, so it can sometimes predict incorrectly for new news.