New Section

```
#Textblob is an open-source python library for processing textual data.
!pip install textblob

!pip install nltk

!pip install swifter
```

```
Requirement already satisfied: textblob in /usr/local/lib/python3.7/dist-packages (0
Requirement already satisfied: nltk>=3.1 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from n
Requirement already satisfied: nltk in /usr/local/lib/python3.7/dist-packages (3.2.5
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from n
Collecting swifter
  Downloading swifter-1.0.9-py3-none-any.whl (14 kB)
Requirement already satisfied: parso>0.4.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: bleach>=3.1.1 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: dask[dataframe]>=2.10.0 in /usr/local/lib/python3.7/d
Requirement already satisfied: pandas>=1.0.0 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: tqdm>=4.33.0 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: ipywidgets>=7.0.0 in /usr/local/lib/python3.7/dist-pa
Collecting psutil>=5.6.6
  Downloading psutil-5.8.0-cp37-cp37m-manylinux2010_x86_64.whl (296 kB)
     |████████████████████████████████| 296 kB 47.5 MB/s
Requirement already satisfied: cloudpickle>=0.2.2 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: webencodings in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: numpy>=1.13.0 in /usr/local/lib/python3.7/dist-packag
Collecting partd>=0.3.10
  Downloading partd-1.2.0-py3-none-any.whl (19 kB)
Collecting fsspec>=0.6.0
  Downloading fsspec-2021.10.1-py3-none-any.whl (125 kB)
     |████████████████████████████████| 125 kB 50.2 MB/s
Requirement already satisfied: toolz>=0.7.3 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: ipykernel>=4.5.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: traitlets>=4.3.1 in /usr/local/lib/python3.7/dist-pac
```

```
import pandas as pd
import json
import re
import nltk
import string
import swifter

from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer


from textblob import TextBlob

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline


from sklearn.preprocessing import FunctionTransformer
from sklearn.base import BaseEstimator, TransformerMixin
from sklearn.pipeline import FeatureUnion
from sklearn.feature_extraction import DictVectorizer
```

```
#filename="News_Category_Dataset_v2.json"

filename="/content/capstoneproject2/part-00000-f48b9de8-0feb-4e1d-9943-df7213008926-c000.c

nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
True
```

```
from google.colab import drive


csv_dtype_dict={"id": "string",
                "published_time": "string",
                "title":"string",
                "summary":"string",
                "source":"string",
                "category":"string",
                "text":"string"}


# Read CSV
df_news = pd.read_csv(filename, error_bad_lines=False, dtype=csv_dtype_dict)


# Check Columns
df_news.columns
```

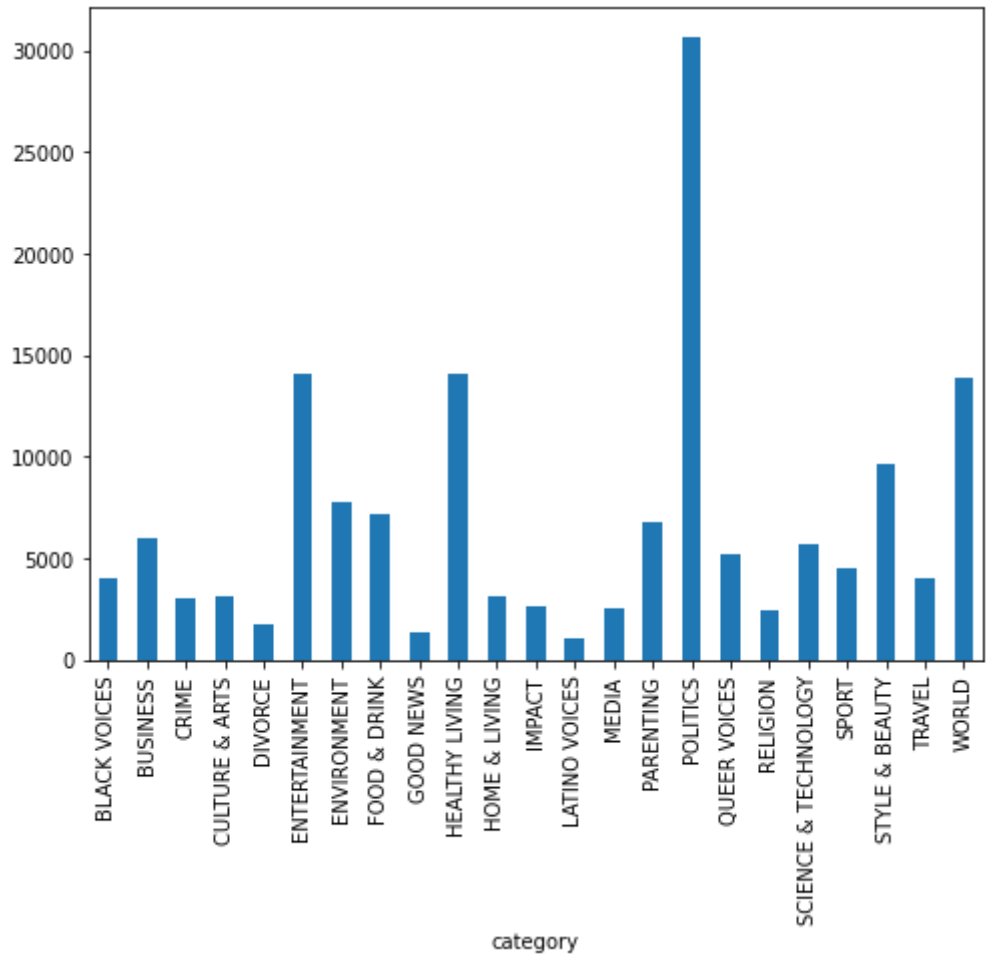
```
Index(['id', 'published_time', 'title', 'summary', 'source', 'category',
       'text'],
      dtype='object')
```

```
from io import StringIO
col = ['category', 'text']
df = df_news[col]
df = df[pd.notnull(df['text'])]
df.columns = ['category', 'text']
df['category_id'] = df['category'].factorize()[0]
category_id_df = df[['category', 'category_id']].drop_duplicates().sort_values('category_i
category_to_id = dict(category_id_df.values)
id_to_category = dict(category_id_df[['category_id', 'category']].values)
df.head()
```

	category	text	category_id
0	WORLD	How rich people could help save the planet fro...	0

```
#Imbalanced class

import matplotlib.pyplot as plt
fig = plt.figure(figsize=(8,6))
df.groupby('category').text.count().plot.bar(ylim=0)
plt.show()
```




```
#Preprocessing


#1. remove the punctuation from text (ex: .,:)
#2. make lowercase because we assume that punctuation and letter case don’t influence the
#3. use NLTK package remove the called stop_word, i.e frecuent words that doesn’t add infc
#    example of stop word are: our, you, yourself, he, his, she,them etc. you can review t
#4. make lemmatization to words, lemmatization is a process of extracting a root word by c
#    For example, “good”, “better”, or “best” is lemmatized (changed) into “good”.



# Text Representation

#limiting the data for first 10k rows due to the memory issue
df1 = df.head(10000)
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(sublinear_tf=True, min_df=5, norm='l2', encoding='latin-1', ngram_
features = tfidf.fit_transform(df1.text).toarray()
labels = df1.category_id
features.shape
```

```
(10000, 6395)
```

```
#Model Selection

from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC
from sklearn.model_selection import cross_val_score
models = [
    RandomForestClassifier(n_estimators=200, max_depth=3, random_state=0),
    LinearSVC(),
    MultinomialNB(),
    LogisticRegression(random_state=0),
]
CV = 5
cv_df = pd.DataFrame(index=range(CV * len(models)))
entries = []
for model in models:
  model_name = model.__class__.__name__
  accuracies = cross_val_score(model, features, labels, scoring='accuracy', cv=CV)
  for fold_idx, accuracy in enumerate(accuracies):
    entries.append((model_name, fold_idx, accuracy))
cv_df = pd.DataFrame(entries, columns=['model_name', 'fold_idx', 'accuracy'])
import seaborn as sns
sns.boxplot(x='model_name', y='accuracy', data=cv_df)
sns.stripplot(x='model_name', y='accuracy', data=cv_df,
              size=8, jitter=True, edgecolor="gray", linewidth=2)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: Conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: Conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: Conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: Conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: Conver
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

```
#Accuracy comparison for each model
cv_df.groupby('model_name').accuracy.mean()
```

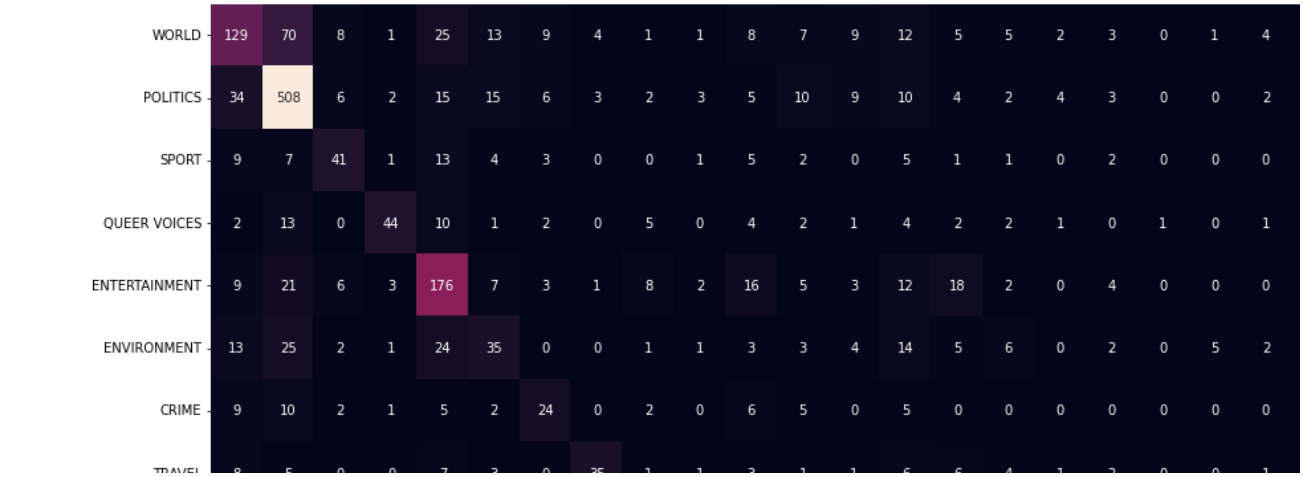
```
model_name
LinearSVC                 0.5367
LogisticRegression        0.5116
MultinomialNB             0.4226
RandomForestClassifier    0.2060
Name: accuracy, dtype: float64
```

```
# LinearSVC and Logistic Regression perform better than the other two classifiers,
# with LinearSVC having a slight advantage with a median accuracy of around 66%.
# Lets use Linear SVC model to train and predict the data
```

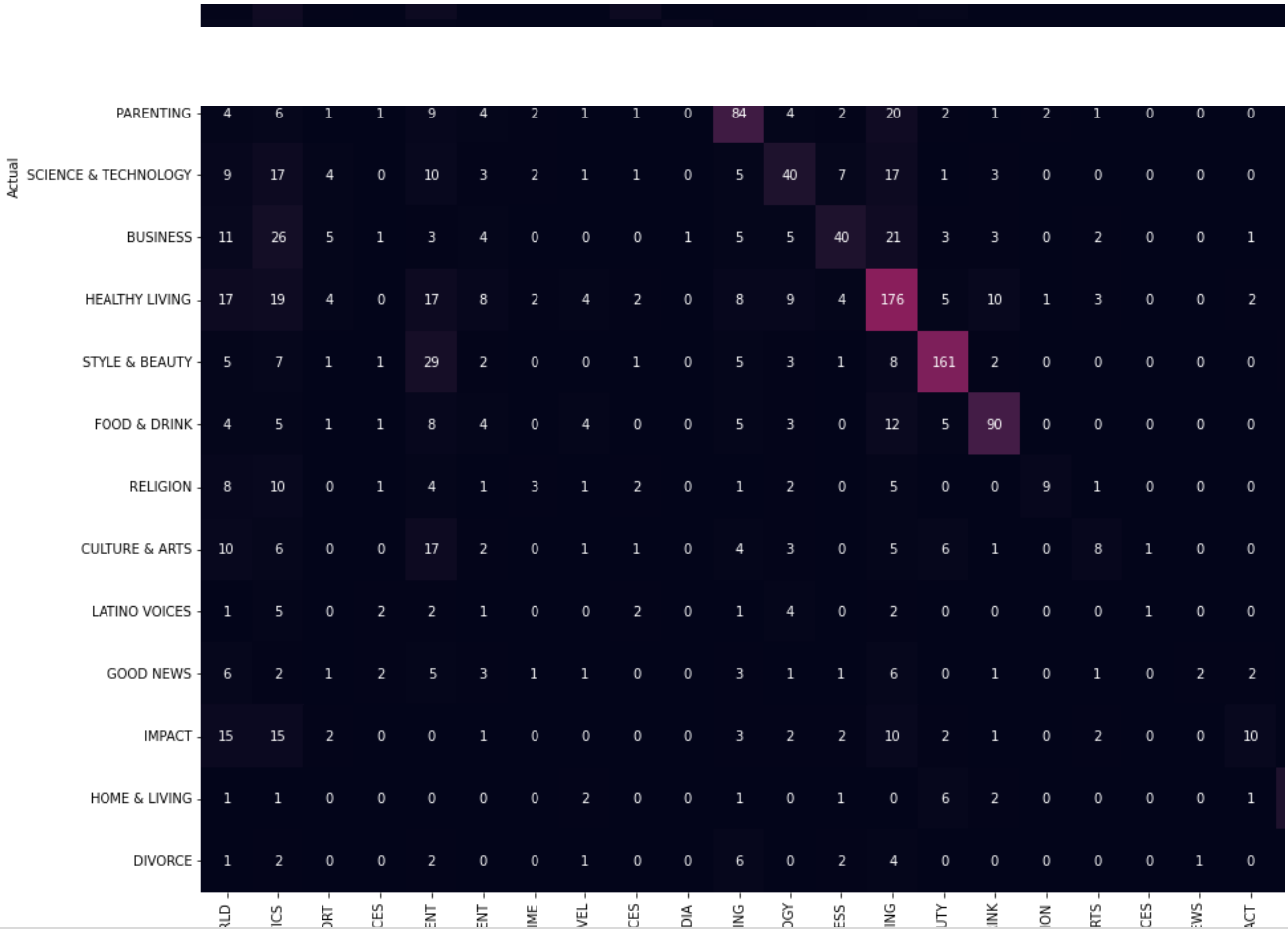
```
#Model Evaluation

model = LinearSVC()
X_train, X_test, y_train, y_test, indices_train, indices_test = train_test_split(features,
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
from sklearn.metrics import confusion_matrix
conf_mat = confusion_matrix(y_test, y_pred)
fig, ax = plt.subplots(figsize=(20,20))
```

```
    fig, ax = plt.subplots(figsize=(20,20))
    sns.heatmap(conf_mat, annot=True, fmt='d',
                xticklabels=category_id_df.category.values, yticklabels=category_id_df.categor
    plt.ylabel('Actual')
    plt.xlabel('Predicted')
    plt.show()
```




```
# Most of the predictions ended up on the diagonal which means the model predictions are g
```



✓ 3s completed at 10:22 AM