Building Your Own AI model

Building an AI tool, that can take file containing 1000's of articles, and tells you what each article is about, using ml & python

Block Chain Domain: https://unstoppabledomains.com/?ref=aniakubow Tabnine Copilot: http://bit.ly/tabnine-top-tool

Sklearn - machine learning library also contains tools

- To build a model, first train a model with more amount of data, in python using the **gravity ai** package.
- Pip install gravityai
- Pip install numpy
- Pip install pandas
- Pip install sklearn

Pickle module - contains functions for **serializing & desterilizing python objects** , **Pickling** is the process where **python object is converted into a byte stream and stored in a binary file.**

tfidf - time frequency inverse document frequency it helps in determining the weighted count of each word in our vocabulary. It is basically a tool that converts a collection of raw documents into a matrix of TF-IDX features.

Location Every Single Category in picture (Excel):

```python
from gravityai import gravityai as grav
import pickle
import pandas as pd

model = pickle.load(open())
tfidf_vectorise = pickle.load(open(''))
lael_encoder = pickle.load(open(''))

#Reading of the csv data
def process(inPath,outPath):
    input_df = pd.read_csv(inPath)
    #term frequency vectoriser to vectorise the data
    feature_vector = tfidf_vectorise.transform(input_df['body'])  #uses t
    predicting_classes = model.predict(feature_vector) #predicts the clas
    input_df['predicted_class'] = label_encoder.inverse_transform(predict
    #save the results to a csv file
    output_df = input_df[['id','category']]
    output_df.to_csv(outPath, index=False)
```

Using Colab: (
https://colab.research.google.com/drive/1pnosax3sbSbBztDL979WD8G4Tuf4WuFY#scrollTo=C_-pwKbrctEU)

```python
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn import preprocessing
import pandas as pd
import json
import pickle
```

Random Forest Classifier is a machine learning algorithm, that is used in the classification of tasks. It is a collection of decision trees that work together to make predictions on root models. Each decision tree is constructed using random dataset and training data and random subset of features, which pulls in randomness and diversity into the model. The final prediction is then made by taking a majority vote among all the individual trees.

Unique() - this provides the category without any duplicate data marks

Now building a machine learning model that can reject the article category by the given content:

- **The data must be transformed into a format that can be understood by the machine.**
  Firstly converting categories into numeric values which is done by the LabelEncoder() module

Creating the array of labels:

```
[8]  financial_corpus_df['category'].unique()

     array(['International_Finance', 'Earning_Reports', 'Commo
            'Economy', 'Fraud', 'Mergers_Acquisitions', 'Polic
            'Capital', 'Litigation', 'Real_Estate'], dtype=obj
```

```
[9]  label_encoder = preprocessing.LabelEncoder()
     label_encoder.fit(financial_corpus_df['category'])
     financial_corpus_df['label'] = label_encoder.transform(fi
```

```
[10] financial_corpus_df['label'].unique()

     array([ 5,  2,  1,  3,  4,  7,  9,  8,  0,  6, 10])
```

Here we are converting the based on weight on the csv file and assign significant unique values, here the new column is created "Label" & each category is labelled.

Now using the tokensier we can tokenise the entire dataset using the pytorch library (converting numbers to names)
Link for reference - https://neptune.ai/blog/vectorization-techniques-in-nlp-guide

Tokenising the articles, breaking down the articles body in a list of tokens, and converting it to the lower case, and remove the stop words (and, or, the) therefore removing low level information, which helps the algorithm to really focus on the words significantly, removing the punctuations, creating a

bag of words which is a vector representation of a document based on repetition of the number of word in the file.

This can be done by using the skitlearn model TfidfVectorizer

Vectorization is the process of converting input from the raw format into vectors of real numbers where the ml models support on, it is a step in the feature of extraction

Creating 2 variables for the body of the corpus:

```
xnum = financial_corpus_df['body']
ynum = financial_corpus_df['label']
```

```
pickle.dump(rf_clf, open('financial_text_classifier.pkl', 'wb
pickle.dump(vectorizer, open('financial_text_vectoriser.pkl',
```

wb- write mode in binary
rb - read in binary

Using Gravity AI create a project -https://www.gravity-ai.com/dashboard/team/a34615ed-6448-4c25-9778-eff181d2dc00/my-projects

Using the docer container to run the ai - model

Docker is used to build, test, deploy applications quickly