Echo Truth

Sarron Tadesse, Kalif Byrd, and Dev Raiyani

What is Echo Truth?

- Echo Truth is an innovative technology project
  - Focused on distinguishing fact from fiction in today's fast-paced information landscape.
  - Leverages advanced machine learning techniques to develop a reliable fake news detection system.

Problem statement

- Tackling the Fake News Challenge:
  - It's damaging public trust,
  - Twisting the truth, and
  - Affecting politics around the world

Solution statement

- Create Echo Truth!
  - Uses machine learning and natural language processing techniques.
  - It's designed to label news as being fake or real
- Our intended audience is for anyone who reads the news

Why this project?

- Motivated by Real-World Impact
- Leveraging CS Skills and AI Learning
- Learn new Technologies and add to our skillset

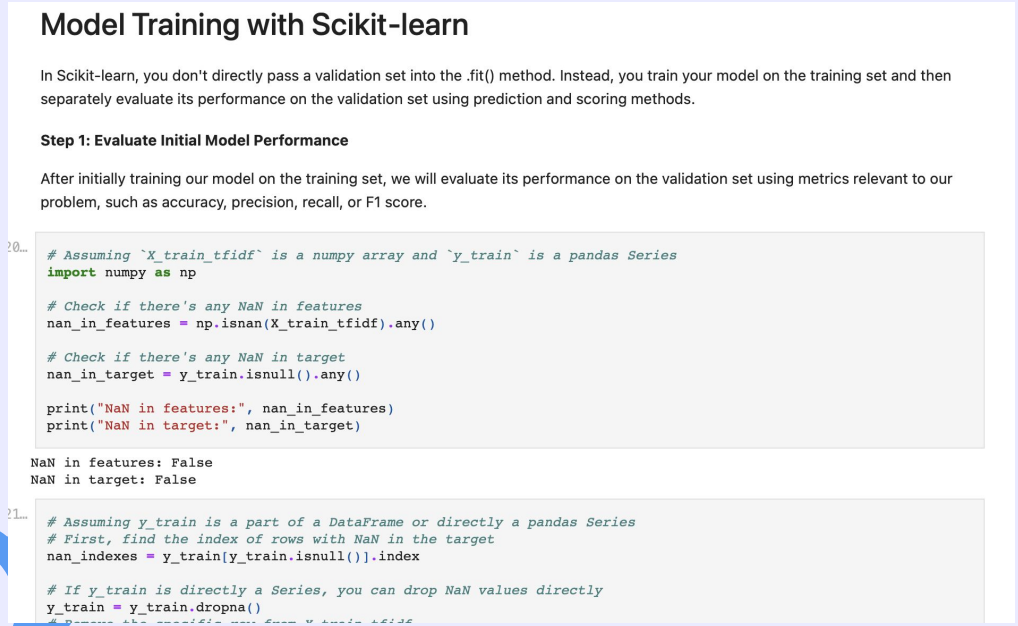
User Interface

- As of now, we are working on making the model but our stretch goal is to deploy it on the cloud (depends on how much time we have and how easily we can figure it out)

Challenges

- ➔ Underscored how hard it would be to sanitize data so it took us a bit of time to do that.
- ➔ Machine learning is a new concept to us, so getting over the learning curve is a bit challenging, but we are progressing so we are happy :)

Status Update



Model Training with Scikit-learn

In Scikit-learn, you don't directly pass a validation set into the .fit() method. Instead, you train your model on the training set and then separately evaluate its performance on the validation set using prediction and scoring methods.

Step 1: Evaluate Initial Model Performance

After initially training our model on the training set, we will evaluate its performance on the validation set using metrics relevant to our problem, such as accuracy, precision, recall, or F1 score.

```
# Assuming `X_train_tfidf` is a numpy array and `y_train` is a pandas Series
import numpy as np

# Check if there's any NaN in features
nan_in_features = np.isnan(X_train_tfidf).any()

# Check if there's any NaN in target
nan_in_target = y_train.isnull().any()

print("NaN in features:", nan_in_features)
print("NaN in target:", nan_in_target)
```

```
NaN in features: False
NaN in target: False
```

```
# Assuming y_train is a part of a DataFrame or directly a pandas Series
# First, find the index of rows with NaN in the target
nan_indexes = y_train[y_train.isnull()].index

# If y_train is directly a Series, you can drop NaN values directly
y_train = y_train.dropna()
```

★ We are in the model training and optimization process!

➔ We have achieved a significant milestone by developing three distinct models, each demonstrating varying levels of accuracy.

➔ The comparative analysis of these models is underway, with the objective of selecting the most effective model for deployment.

Thank you, question?