EMOTION TRACKING IN UKRAINE-RUSSIA WAR TWEETS: A COMPARISON OF MACHINE LEARNING AND VADER

Presented by : Sanjil Kokare (775482)
Professor : Dr. Michael Colombo
Subject : Analysis of Semi-Structured & Unstructured Data
University : Fom University of Applied Sciences

Content

INTRODUCTION

The Ukraine - Russia war, one of the most geopolitically impactful events of the decade. Many countries support Ukraine by imposing economic sanctions against Russia.
It generated millions of tweets reflecting fear, anger, hope, and global solidarity.
Russia on 24 February 2022 launched a wave of missile attacks across many of Ukraine's biggest cities.
At least seven civilians were killed and 33 injured between Wednesday and Thursday, Ukraine's presidential office said, including one person killed and 23 wounded when four Kalibr cruise missiles hit the southern city of Mykolaiv.

Aim

This presentation aims to analyze that public emotion using natural language processing specifically, by comparing a machine learning model trained on tweet data with VADER, a rule-based sentiment analyzer designed for social media.

DATASET OVERVIEW

Source: Kaggle – Amir Motefaker's Ukraine–Russia War Tweets Dataset
~10,000+ tweets collected during the early phase of the conflict
Key columns used:

- tweet (text)
- language (filter for English)
- username (user frequency, optional)

Preprocessing:

- Removed URLs, hashtags, mentions, special characters
- Converted to lowercase
- Stored cleaned version in 'cleaned_tweet'

```
data.head()
```

	username	tweet	language
0	tomasliptai	@nazijaeger__ @derwener @Anonymous9775 Russia ...	en
1	paperfloure	The Russia HAARP which could destroy USA in on...	en
2	katetbar1	Putin gives Steven Seagal Russia’s O...	en
3	jlhrdhmom	@MainelifeR @BaddCompani It's ALWAYS PROJECTIO...	en
4	phemikali	@Pottingpinks @mfa_russia @mod_russia @mil_his...	en

TEXT PREPROCESSING

- Removed: URLs, mentions, hashtags, special characters
- Lowercased and cleaned: e.g., 'WAR!!!' → 'war'
- Result saved as: 'tweet'
- Cleaned text enables consistent vectorization and model performance.

Cleaning

```
[ ]  nltk.download('stopwords')
     stemmer = nltk.SnowballStemmer("english")
     stopword=set(stopwords.words('english'))

     def clean(text):
         text = str(text).lower()
         text = re.sub('\[.*?\]', '', text)
         text = re.sub('https?://\S+|www\.\S+', '', text)
         text = re.sub('<.*?>+', '', text)
         text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
         text = re.sub('\n', '', text)
         text = re.sub('\w*\d\w*', '', text)
         text = [word for word in text.split(' ') if word not in stopword]
         text=" ".join(text)
         text = [stemmer.stem(word) for word in text.split(' ')]
         text=" ".join(text)
         return text
     data["tweet"] = data["tweet"].apply(clean)
```

Most Frequent Word

MACHINE LEARNING MODEL UNDERSTANDING PUBLIC EMOTION

Vectorized text using TF-IDF (max 5000 features)
Trained Logistic Regression model
Labels:

- Positive → Encouragement, support
- Neutral → News, factual updates
- Negative → Fear, violence, outrage


```
[ ]
    # Step 2: TF-IDF vectorization
    vectorizer = TfidfVectorizer(max_features=5000)
    X = vectorizer.fit_transform(data['cleaned_tweet'])
    y = data['label']
```

```
# Step 3: Train model
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
LogisticRegression
LogisticRegression()
```

```
from sklearn.metrics import classification_report, accuracy_score

y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.8227658512231653
              precision    recall  f1-score   support

          -1       0.89      0.59      0.71       447
           0       0.78      0.98      0.87       940
           1       0.89      0.76      0.82       616

    accuracy                           0.82      2003
   macro avg       0.85      0.77      0.80      2003
weighted avg       0.84      0.82      0.82      2003
```

VADER LEXICON-BASED EMOTION DETECTION

- VADER (Valence Aware Dictionary for sEntiment Reasoning)
- Pre-trained on social media text
- Better at catching emotionally charged or context-rich tweets
- Labels tweets as Positive, Neutral, or Negative based on rule-based scoring

	tweet	Positive	Negative	Neutral
0	nazijaeg derwen russia place satan rule well	0.259	0.000	0.741
1	russia haarp could destroy usa one fell swoop ...	0.000	0.280	0.720
2	putin give steven seagal order friendship	0.367	0.000	0.633
3	mainelif baddcompani it alway project russia	0.000	0.000	1.000
4	pottingpink mfarussia modrussia milhistrf muze...	0.068	0.078	0.854

```python
def vader_sentiment(text):
    score = analyzer.polarity_scores(str(text))['compound']
    if score >= 0.05:
        return "Positive"
    elif score <= -0.05:
        return "Negative"
    else:
        return "Neutral"

# Apply to DataFrame
data['vader_sentiment'] = data['tweet'].apply(vader_sentiment)
data['vader_sentiment'].value_counts()
```

vader_sentiment	count
Negative	4199
Positive	3145
Neutral	2670

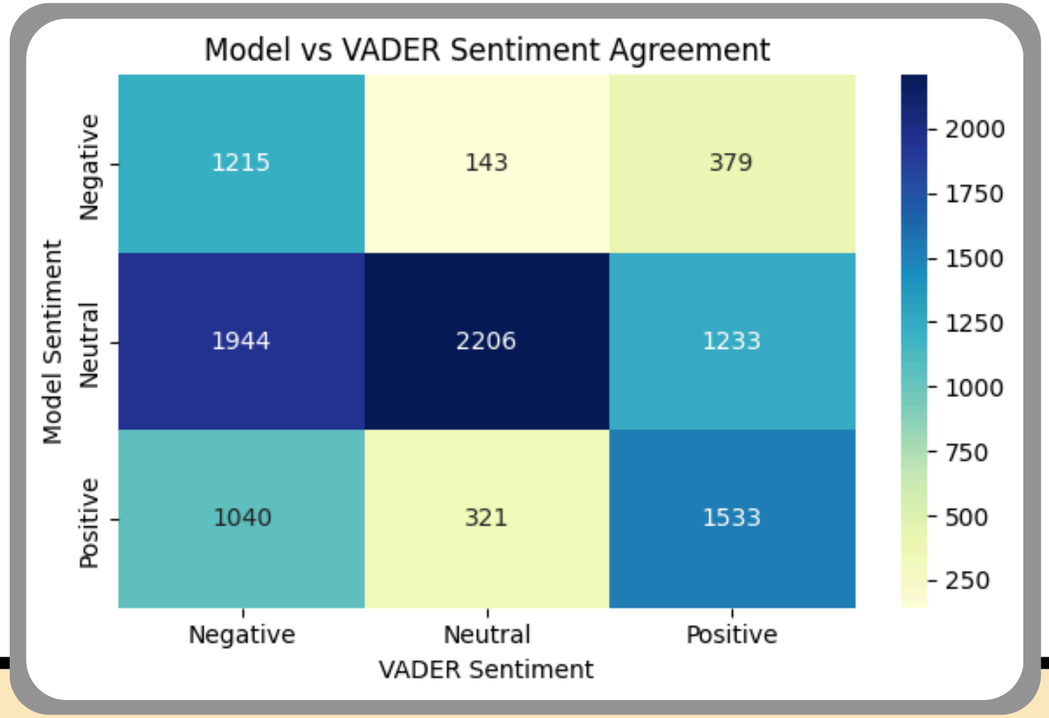
dtype: int64

MODEL VS VADER: SENTIMENT COMPARISON

- Applied both ML model and VADER on the same tweets
- Created crosstab and heatmap to compare predictions
- Findings:

  VADER detects more negative sentiment in emotional or violent contexts

  ML model misclassifies due to lack of semantic understanding



CONCLUSION & FUTURE SCOPE

- Sentiment analysis reveals how society reacts to war events
- ML and VADER each offer value — together they offer deeper insight
- Future improvements:

  Use BERT for better contextual accuracy

  Build a real-time emotional monitoring dashboard for crisis response

TOOLS OR LIBRARIES

Data Handling & Preprocessing
pandas – for reading, filtering, and managing tweet data
re – for regex-based text cleaning
numpy – for numerical operations (if used implicitly)

Natural Language Processing (NLP)
TextBlob – to calculate sentiment polarity (ML baseline)
vaderSentiment – rule-based sentiment analysis for social media
nltk – if used for tokenization or stopword filtering (optional)

Machine Learning & Modeling
scikit-learn (sklearn):
TfidfVectorizer – to convert tweet text into numeric features
LogisticRegression – sentiment classification model
train_test_split – to split the dataset
accuracy_score, classification_report – evaluation metrics

Development Environment
Google Colab – for coding, visualizations, and live demo
Canva Editor – to generate your slides

Visualization
matplotlib – for bar and pie charts
seaborn – for heatmaps and enhanced visual styling
wordcloud – for visualizing high-frequency emotional terms

REFERENCES

Dataset Source

Ukraine–Russia Twitter Sentiment Dataset

https://www.kaggle.com/code/amirmotefaker/ukraine-russia-war-twitter-sentiment-analysis

VADER Sentiment Lexicon

Official GitHub (by C.J. Hutto and Eric Gilbert, Georgia Tech):

https://github.com/cjhutto/vaderSentiment

Research Paper PDF:

https://ojs.aaai.org/index.php/ICWSM/article/view/14550

TF-IDF & Logistic Regression Concepts

Scikit-learn TF-IDF Vectorizer:

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

Logistic Regression in NLP:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

Background & News Sources

Reuters Ukraine Crisis Coverage:

https://www.reuters.com/places/ukraine/

Twitter Blog: Role in Crisis Communication:

https://blog.twitter.com/en_us/topics/company/2022/supporting-the-people-of-ukraine


DANKE