Sentiment analysis of text

Introduction:

With all of the tweets, reviews and feedbacks circulating every second it is hard to tell whether the sentiment behind a specific tweet will **impact a company, or a person's, brand for being viral or devastate profit** because it strikes a negative tone. Capturing sentiment in language is important in these times where decisions and reactions are created and updated in seconds. But, which words actually lead to the sentiment description.

Importance of Sentiment Analysis:

In today's digital age, where information flows ceaselessly through social media, online reviews, and customer feedback, Sentiment analysis empowers us to harness the wealth of unstructured text data to gain insights into public opinion, customer satisfaction, and emotional responses ,**Impact on Decision-Making.**

"Today, I'll be discussing a machine learning task I have done on DEPI internship . We'll go from uploading data to visualizing it, and finally building a full pipeline using Python libraries . the model selected which is SVM(support vector machine). The key tools I'll use include **pandas** for data handling, **NumPy** for numerical operations, and **matplotlib** for visualizations. Everything will be implemented in a **Jupyter Notebook** to ensure a clear and interactive workflow.

The key steps we'll cover are:

- **Exploratory Data Analysis (EDA)**: Understanding the data we have.
- **Data Clean and Preprocessing**: Preparing the data for analysis.
- **Feature Engineering**: Creating features that improve model performance.
- **Model Architecture**: Defining the structure of our machine learning model.
- **Training the Model**: Fitting our model to the training data.
- **Model Evaluation**: Assessing the model's performance.
- **Generating the Full Pipeline**: Integrating all steps into a seamless workflow.

First we load the(train,test) data:-

```
tweets_df = pd.read_csv("train.csv")
tweets_df = tweets_df[['text', 'sentiment']]

test_df = pd.read_csv("test.csv")
test_df = test_df[['text', 'sentiment']]

# Remove empty or NaN rows from training data
tweets_df.dropna(subset=['text'], inplace=True)
tweets_df = tweets_df[tweets_df['text'].str.strip() != '']
```

then we visualize it using a simple bar chart:-

```
]: plt.figure(figsize=(10, 6))
   sns.countplot(x='sentiment', data=tweets_df, palette='viridis')
   plt.title('Sentiment Distribution in Training Data')
   plt.xlabel('Sentiment')
   plt.ylabel('Count')
   plt.show()
```

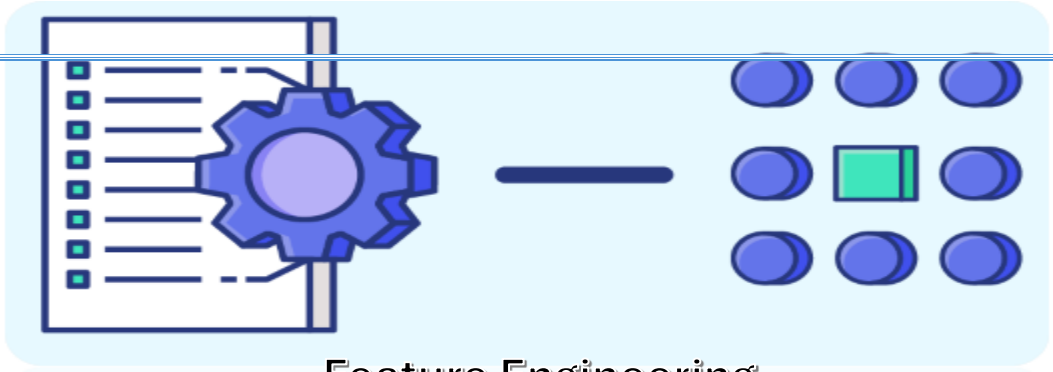
Data Clean and Preprocessing

Define a function textClean to preprocess text :-

```
]: def textClean(text):
       nopunc = [char.lower() for char in text if char not in string.punctuation]
       nopunc = ''.join(nopunc)
       tokens = word_tokenize(nopunc)
       nohttp = [word for word in tokens if word[0:4] != 'http']
       nostop = [word for word in nohttp if word not in stopwords.words('english')]
       return nostop
```

This function do:

- Remove **punctuation**
- **Lowercasing characters**
- **Tokenization**
- **Removing URLs**
- **Removing Stopwords**



Feature Engineering

Vectorize the cleaned text using CountVectorizer to transform it into a numerical format for machine learning

Analyzer purpose is to use “textClean” on all text that will be converted to vector to clean them to improve accuracy

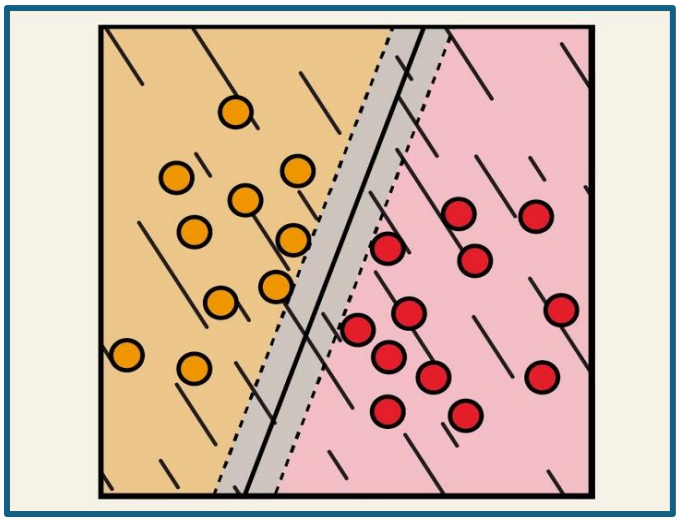
This what vectorizer do:

```
vectorizer = CountVectorizer(analyzer=textClean)

message = vectorizer.fit_transform(tweets_df['text'])

doc = pd.DataFrame(message.toarray(), columns=vectorizer.get_feature_names_out())
```

SMS
SECRET PRIZE! CLAIM SECRET PRIZE NOW!!
Coming to my secret party?
Winner! Claim secret prize now!

secret	prize	claim	now	coming	to	my	party	winner
2	2	1	1	0	0	0	0	0
1	0	0	0	1	1	1	1	0
1	1	1	1	0	0	0	0	1

Model
Architecture



Support Vector Machine (SVM):

SVM is a supervised learning algorithm used for classification and regression tasks.It works by finding the hyperplane that best separates data points belonging to different classes while maximizing the margin between classes.

Key Characteristics:

Non-probabilistic binary linear classifier

Effective in high-dimensional spaces

Use Cases:

Highlight specific use cases where SVMs excel, such as text classification, image classification, and bioinformatics.

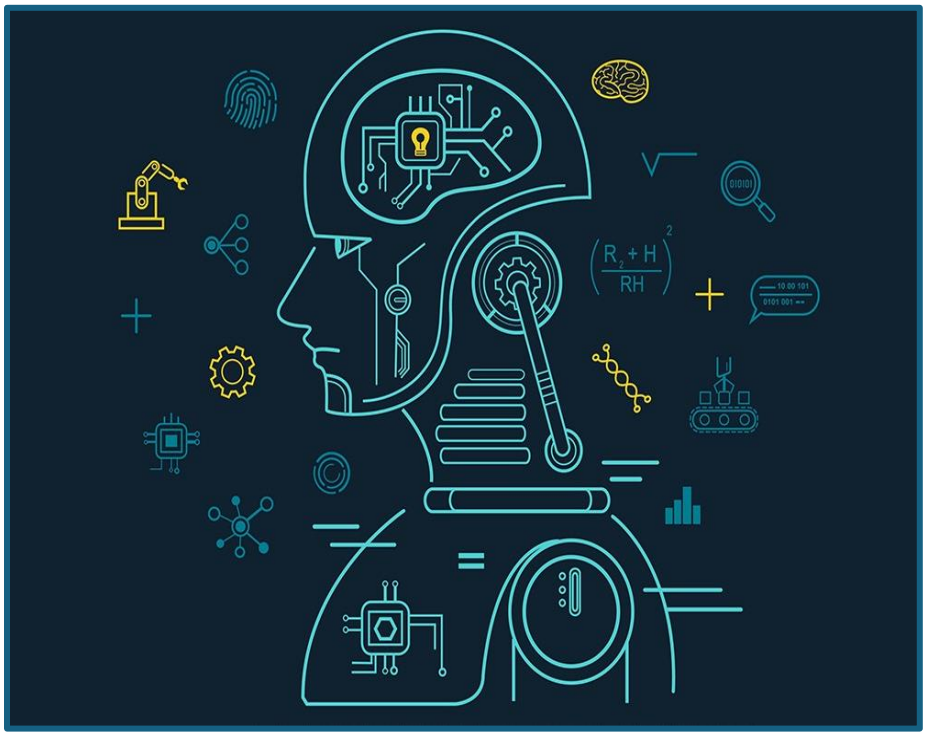
SVM Architecture:

- SVM constructs a hyperplane in a multidimensional space that separates data points into different classes.
- The choice of kernel function (linear, radial basis function, polynomial) influences the decision boundary Advantages.

Model Training:

- During training, the SVM learns the optimal hyperplane that maximizes the margin between classes.
- The choice of kernel and hyperparameter values significantly influences the model's performance.

Training the Model



Spilting data:

To data without target value and target value data to make model

Train on it

```
#split the data into 80% training and 20% testing
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(message,tweets_df.sentiment,test_size=0.20,random_state=42)
```

Kernel Functions:

Linear Kernel:

Description: A linear kernel assumes a linear decision boundary.

Use Case: Suitable for linearly separable data where classes can be cleanly divided by a straight line or hyperplane.

RBF (Radial Basis Function) Kernel:

Description: Radial Basis Functions create a non-linear decision boundary by mapping data into a higher-dimensional space.

Use Case: Effective for capturing complex relationships and handling non-linear separations. Widely used for its versatility.

Polynomial Kernel:

Description: Polynomial kernels create decision boundaries with polynomial shapes.

Use Case: Useful for data that exhibits polynomial relationships.

```
from sklearn.svm import SVC
```

```
svc_model = SVC(C= .2, kernel='linear', gamma= .8)
svc_model.fit(xtrain, ytrain)
```

```
SVC(C=0.2, gamma=0.8, kernel='linear')
```

We used linear because our data is in a straight line (each line don't depend on other lines)

Model Evaluation

Evaluate the SVM model on the training data:

```
train_pred = svc_model.predict(xtrain)
print("Training Data Classification Report:\n", classification_report(ytrain, train_pred))
print("Training Data Accuracy:", accuracy_score(ytrain, train_pred))
```

```
Training Data Classification Report:
               precision    recall  f1-score   support

    negative       0.88      0.75      0.81      6253
     neutral       0.76      0.88      0.82      8863
    positive       0.88      0.82      0.85      6868

    accuracy                           0.82     21984
   macro avg       0.84      0.82      0.82     21984
weighted avg       0.83      0.82      0.82     21984

Training Data Accuracy: 0.8245542212518195
```

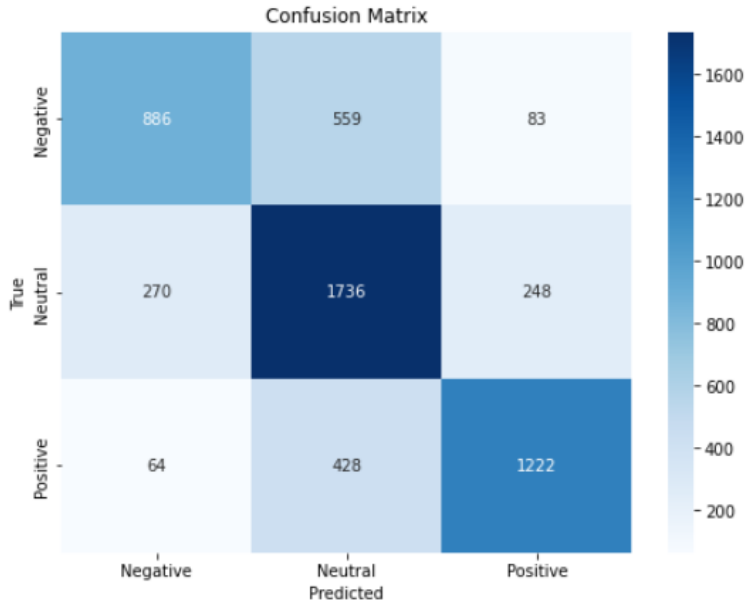
Evaluate the SVM model on the test data:

```
#test from train.csv
test_pred = svc_model.predict(xtest)
print("Testing Data Classification Report:\n", classification_report(ytest, test_pred))
print("Testing Data Accuracy:", accuracy_score(ytest, test_pred))
```

```
Testing Data Classification Report:
               precision    recall  f1-score   support

    negative       0.73      0.58      0.64      1528
     neutral       0.64      0.77      0.70      2254
    positive       0.79      0.71      0.75      1714

    accuracy                           0.70      5496
   macro avg       0.72      0.69      0.70      5496
weighted avg       0.71      0.70      0.70      5496

Testing Data Accuracy: 0.6994177583697234
```

Visualize the confusion matrix for the model's predictions:

```
conf_matrix = confusion_matrix(ytest, test_pred, labels=['negative', 'neutral', 'positive'])

plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Negative', 'Neutral', 'Positive'],
            yticklabels=['Negative', 'Neutral', 'Positive'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```



Make predictions on new entries text samples:

```
new_text = ["I love this movie!", "This product is terrible.", "The food was delicious.", "This product is alright"]
new_features = vectorizer.transform(new_text)
new_predictions = svc_model.predict(new_features)

for i in new_predictions:
    if i == 'positive':
        print(i, "😀")
    elif i == 'negative':
        print(i, "😞")
    elif i == 'neutral':
        print(i, "😐")
```

```
positive 😀
negative 😞
positive 😀
neutral 😐
```

Making a full pipeline by loading preprocessing and analyzing the data(top 500 rows of TeePublic_review.csv) then saving the new file with the predicted sentiment as a new column.

```
df = pd.read_csv('TeePublic_review.csv', encoding='latin-1', nrows=5000)

df = df[['review']].dropna()

new_message = vectorizer.transform(df['review'])

df['predicted_sentiment'] = svc_model.predict(new_message)

full_df = pd.read_csv('TeePublic_review.csv', encoding='latin-1')

full_df.loc[full_df.index[:len(df)], 'predicted_sentiment'] = df['predicted_sentiment']

full_df.to_csv('TeePublic_review_with_sentiments.csv', index=False)

full_df.head()
```

	reviewer_id	store_location	latitude	longitude	date	month	year	title	review	review-label	predicted_sentiment
0	0.0	US	37.090240	-95.712891	2023	6	2015 00:00:00	Great help with lost order	I had an order that was lost in transit. When ...	5	negative
1	1.0	US	37.090240	-95.712891	2023	6	2024 00:00:00	I ordered the wrong size tee and hadï¿½ï¿½ï¿½	I ordered the wrong size tee and had difficult...	5	negative
2	2.0	US	37.090240	-95.712891	2023	6	2017 00:00:00	These guys offer the best customerï¿½ï¿½ï¿½	These guys offer the best customer service in ...	5	positive
3	3.0	US	37.090240	-95.712891	2023	6	2024 00:00:00	Good Stuff	Looked for an obscure phrase on a shirt. Teepu...	5	positive
4	4.0	CA	56.130366	-106.346771	2023	6	2023 00:00:00	My order arrived in a good timelyï¿½ï¿½ï¿½	My order arrived in a good timely fashion & th...	4	positive

Future work:-

- ❑ use it in ecommerce website .
- ❑ building it in form of webpage where you write text and return sentiment maybe.
- ❑ build an API to allow other websites to use it.
- ❑ improve model and train it on more data.
- ❑ improve vectorizer.

Thanks

For reading

Feel free to send me any question?

Abdelrahmanmoataz888@gmail.com