Problem Statement: I have chosen to work for a product development company, which collects social media data from various sources like Amazon, Google, Facebook, Twitter("x") reviews. the main task of this company is to evaluate product and services based on social media reviews to better learn customer feedback, which helps the company understand consumer behavior on various product and services.

As a data scientist i have been given the task to evaluate Twitter("x") reviews, the task is to catogrice the reviews into positive, negative, neutral and irrevelent.

Formulating the problem as a machine learning task: I aim to classify the sentiment of tweets as negative, positive, or neutral. This involves cleaning and preprocessing a chosen dataset, training machine learning algorithms on the prepared data, and selecting the top-performing model based on various evaluation metrics.

Importing the libraries

```
from google.colab import drive
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax
import re
import pandas as pd
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
import re
from nltk.corpus import stopwords
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('maxent_ne_chunker')
nltk.download('words')
nltk.download('vader_lexicon')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping chunkers/maxent_ne_chunker.zip.
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Unzipping corpora/words.zip.
[nltk_data] Downloading package vader_lexicon to /root/nltk_data...
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
True
```

Loading the Dataset

- I have used google drive to access my dataset for easy convinence
- The dataset was taken from Kaggle for this project https://www.kaggle.com/datasets/jp797498e/twitter-entity-sentiment-analysis

```
# Mounts the google grive
drive.mount('/content/drive')
#Defines the column heading
column_names = ['Id_old','companies','feedback type','review']
# the dataset twitter_training.csv is stored as pandas dataframe in
        in_x
in_x = pd.read_csv('/content/drive/MyDrive/Twitter
        Dataset/twitter_training.csv', header=None,
        names=column_names)
in_v = pd.read_csv('/content/drive/MyDrive/Twitter
        Dataset/twitter_validation.csv', header=None,
        names=column_names)
in_x = in_x.sample(n=20000, random_state=15)
in_x.shape
```

```
Mounted at /content/drive
```

```
(20000, 4)
```

Data Preprocessing and Text Cleansing

renaming the old id to new id with

```
in_x =in_x.reset_index(drop=True)
in_x.index += 1  # Starts the new ID from 1 instead of 0
in_x.reset_index(inplace=True)
in_x.rename(columns={'index': 'Id'}, inplace=True)
in_x= in_x.drop('Id_old', axis=1)
in_x.head(10)
```

	Id	companies	feedback type	review
0	1	TomClancysGhostRecon	Positive	Ghost Recon Breakpoint download update 1.09 wi...
1	2	NBA2K	Negative	RT Where would he go? GOAT Larry Bird
2	3	Dota2	Neutral	@gameflip RE: Someone abusing discount code in...
3	4	Google	Neutral	Buy Remove Reviews from Google - Get Bad Revie...
4	5	AssassinsCreed	Positive	It is not the first time that the EU Commissio...
5	6	CS-GO	Irrelevant	THE BEST rising CS:GO Team. : LIVE NOW. :. : t...
6	7	LeagueOfLegends	Neutral	heaps of fan sketches
7	8	Battlefield	Negative	Battlefield 4 literally had no free maps on la...
8	9	Hearthstone	Positive	us
9	10	Microsoft	Neutral	Are you looking for a new job? Check out this ...

Define cleaning functions

```
# Define cleaning functions
in_x['review'] = in_x['review'].fillna('').astype(str)
def remove_urls(text):
    return re.sub(r'https?://\S+|www\.\S+', '', text)
def remove_mentions(text):
    return re.sub(r'@\w+', '', text)
def remove_hashtags(text):
    return re.sub(r'#\w+', '', text)
```

```python
def remove_rt(text):
    return re.sub(r'\brt\b', '', text, flags=re.I)
def remove_special_characters(text):
    return re.sub(r'[^a-zA-Z\s]', '', text)
def remove_extra_spaces(text):
    return re.sub(r'\s+', ' ', text).strip()
def to_lower(text):
    return text.lower()
def remove_stop_words(text):
    stop_words = set(stopwords.words('english'))
    return ' '.join([word for word in text.split() if word not in
        stop_words])
```

Data Exploration

Checking the Shape of our dataframe from a sample of 'review' column.

```python
text = in_x['review'][79]
text
```

{"type":"string"}

we can see that the review text contains unwanted characters which has to be preprocessed

Checking the data-types and whether null values exist

```python
print(in_x.dtypes)
print(in_x.isnull().sum())
```

```
Id                int64
companies        object
feedback type    object
review           object
dtype: object
Id               0
companies        0
feedback type    0
review           0
dtype: int64
```

we can see that tere are no null values

checking the spread of Sentiment in our dataset

```python
sns.countplot(x='feedback type', data=in_x)
plt.title("Sentiment Spread")
plt.show()
```

comparing various companies feedback types which is alredy labeled in this dataset

```python
compare = in_x.pivot_table(index='companies', columns='feedback type',
        aggfunc='size', fill_value=0)
compare
```

feedback type	Irrelevant	Negative	Neutral	Positive
companies				
Amazon	57	156	336	82
ApexLegends	54	139	236	182
AssassinsCreed	82	110	43	392
Battlefield	247	136	85	147
Borderlands	69	108	159	275

feedback type	Irrelevant	Negative	Neutral	Positive
companies				
CS-GO	183	101	141	187
CallOfDuty	160	235	111	124
CallOfDutyBlackopsColdWar	162	173	92	246
Cyberpunk2077	126	113	138	261
Dota2	102	210	152	156
FIFA	134	317	25	134
Facebook	186	177	218	46
Fortnite	201	201	45	155
Google	140	175	213	96
GrandTheftAuto(GTA)	215	169	69	174
Hearthstone	67	143	202	202
HomeDepot	91	243	86	201
LeagueOfLegends	78	173	234	161
MaddenNFL	29	468	50	103
Microsoft	50	201	225	158
NBA2K	46	400	76	98
Nvidia	32	136	262	244
Overwatch	184	155	76	200
PlayStation5(PS5)	107	122	157	263
PlayerUnknownsBattlegrounds(PUBG)	238	201	66	103
RedDeadRedemption(RDR)	51	85	221	254
TomClancysGhostRecon	8	247	217	193
TomClancysRainbowSix	25	288	155	127
Verizon	44	267	146	128
WorldOfCraft	53	77	276	182
Xbox(Xseries)	202	99	109	228
johnson&johnson	54	233	273	69

split our data into train and test sets

We can now split our data into train and test sets, also performing splitting of target labels.

```
from sklearn.model_selection import train_test_split
X = in_x.drop(columns=['Id', 'feedback type'])
y = in_x['feedback type']
X_train, X_test, y_train, y_test = train_test_split(X, y,
        test_size=0.33, random_state=125)
X_train
```

	companies	review
9733	johnson&johnson	Johnson & Johnson will stop selling its talc-b...
7340	PlayStation5(PS5)	i swear i would buy a mf a ps5 baby worth gett...
12486	RedDeadRedemption(RDR)	Listen, you have great takes here and there bu...
14414	RedDeadRedemption(RDR)	This is fantastic!
14863	johnson&johnson	So to @realDonaldTrump... another shared failure

	companies	review
...	...	...
19414	NBA2K	@NBA2K . . okamiStar most original,okamiStar...
18243	Borderlands	There are tons and tons other quality of life ...
5375	Battlefield	Enjoy the power of Frostbite 2 technology in i...
10397	TomClancysRainbowSix	...
19389	LeagueOfLegends	Jinx is a nutjob but she's also really fun. Al...

13400 rows × 2 columns

Applying cleaning function to the x_train not applying clean function on test data

```
X_train['clean_tweet'] =
        X_train['review'].apply(remove_urls).apply(remove_mentions).apply(remove_hashtags).apply(remove_rt).ap
X_train= X_train.drop(columns=['review'])
X_train.head()
```

	companies	clean_tweet
9733	johnson&johnson	johnson johnson stop selling talcbased baby po...
7340	PlayStation5(PS5)	swear would buy mf ps baby worth getting punch...
12486	RedDeadRedemption(RDR)	listen great takes still stinks also opinion w...
14414	RedDeadRedemption(RDR)	fantastic
14863	johnson&johnson	another shared failure

one hot encoding the test and train of companies

```
X_train = pd.get_dummies(X_train, columns=['companies'])
X_test = pd.get_dummies(X_test, columns=['companies'])
```

text vectorization using TF-IDF vectorization and removing the text column 'clean_tweet'

```
from sklearn.feature_extraction.text import TfidfVectorizer
from imblearn.over_sampling import SMOTE
import scipy.sparse
vectorizer = TfidfVectorizer()

X_train_ve = vectorizer.fit_transform(X_train['clean_tweet'])
X_test_ve = vectorizer.transform(X_test['review'])

X_train_numerical = X_train.drop('clean_tweet', axis=1)
X_train_combined = scipy.sparse.hstack((X_train_numerical,
        X_train_ve))

X_test_numerical = X_test.drop('review', axis=1)
X_test_combined = scipy.sparse.hstack((X_test_numerical, X_test_ve))
```

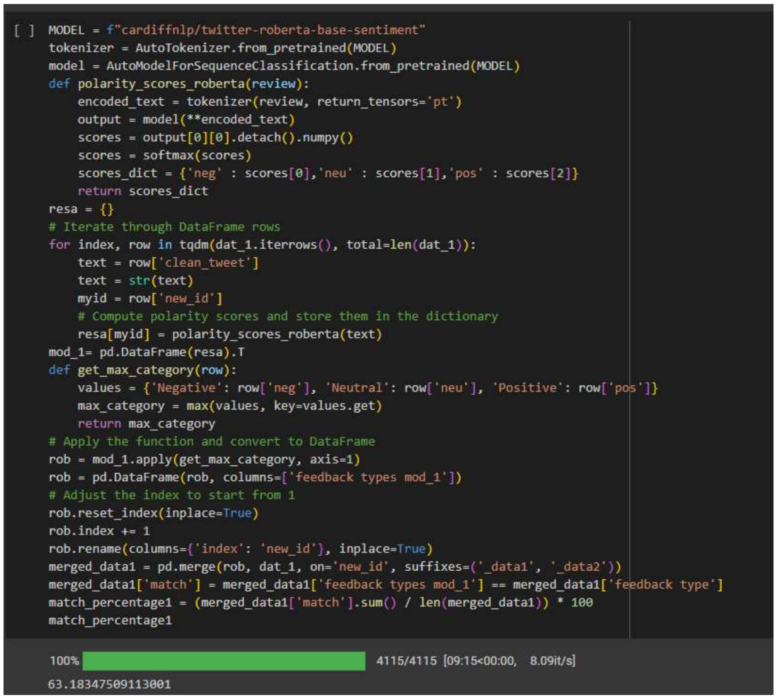
Applying SMOTE because from data exploration we found that we had uneven spread of feedback type

```
smote = SMOTE(random_state=42)
X_train_res, y_train_res = smote.fit_resample(X_train_combined,
        y_train)
```

Model selection

Going with the traditional machine learning models to achieve our classification task because, when i tried using pre trained model like roberta, the issue is that roberta only has three sentimental class positive, negative and neutral, but my feedback type has a 4th type irrevelent, after trying various methode roberta was not able to effectively differentiate irrevelent class. this is the reason for using traditional approch model.

here is the image of the results from the roberta model without the irrevelent class (for reference only)



```
[ ] MODEL = f"cardiffnlp/twitter-roberta-base-sentiment"
    tokenizer = AutoTokenizer.from_pretrained(MODEL)
    model = AutoModelForSequenceClassification.from_pretrained(MODEL)
    def polarity_scores_roberta(review):
        encoded_text = tokenizer(review, return_tensors='pt')
        output = model(**encoded_text)
        scores = output[0][0].detach().numpy()
        scores = softmax(scores)
        scores_dict = {'neg' : scores[0],'neu' : scores[1],'pos' : scores[2]}
        return scores_dict
    resa = {}
    # Iterate through DataFrame rows
    for index, row in tqdm(dat_1.iterrows(), total=len(dat_1)):
        text = row['clean_tweet']
        text = str(text)
        myid = row['new_id']
        # Compute polarity scores and store them in the dictionary
        resa[myid] = polarity_scores_roberta(text)
    mod_1= pd.DataFrame(resa).T
    def get_max_category(row):
        values = {'Negative': row['neg'], 'Neutral': row['neu'], 'Positive': row['pos']}
        max_category = max(values, key=values.get)
        return max_category
    # Apply the function and convert to DataFrame
    rob = mod_1.apply(get_max_category, axis=1)
    rob = pd.DataFrame(rob, columns=['feedback types mod_1'])
    # Adjust the index to start from 1
    rob.reset_index(inplace=True)
    rob.index += 1
    rob.rename(columns={'index': 'new_id'}, inplace=True)
    merged_data1 = pd.merge(rob, dat_1, on='new_id', suffixes=('_data1', '_data2'))
    merged_data1['match'] = merged_data1['feedback types mod_1'] == merged_data1['feedback type']
    match_percentage1 = (merged_data1['match'].sum() / len(merged_data1)) * 100
    match_percentage1

    100% 4115/4115 [09:15<00:00,  8.09it/s]
    63.18347509113001
```

here is a compar of Lightgbm and MultinomialNB

Lightgbm

```
import sklearn.model_selection
import sklearn.feature_extraction.text
from sklearn.model_selection import train_test_split
import tensorflow as tf
import sklearn.ensemble
import lightgbm
parameters_grid = {
    "n_estimators" : [100,150],
    "learning_rate" : [0.07,0.1],
    "num_leaves" : [10,20]
}

model_lgbm =
        sklearn.model_selection.GridSearchCV(lightgbm.LGBMClassifier(objective='multiclass',
        force_col_wise=True, class_weight='balanced', random_state=0), parameters_grid,
        cv=5, scoring='f1_macro',n_jobs=-1)
model_lgbm.fit(X_train_res, y_train_res)
print("f1_macro score LightGBM Classifier: ",model_lgbm.best_score_)
print("Hyperparameters for training LightGBM Classifier:
        ",model_lgbm.best_params_)
```

```
/usr/local/lib/python3.10/dist-
packages/joblib/externals/loky/backend/fork_exec.py:38:
RuntimeWarning: os.fork() was called. os.fork() is incompatible with
multithreaded code, and JAX is multithreaded, so this will likely lead
to a deadlock.
  pid = os.fork()
/usr/local/lib/python3.10/dist-
packages/joblib/externals/loky/backend/fork_exec.py:38:
RuntimeWarning: os.fork() was called. os.fork() is incompatible with
multithreaded code, and JAX is multithreaded, so this will likely lead
```

```
to a deadlock.
  pid = os.fork()

[LightGBM] [Info] Total Bins 38866
[LightGBM] [Info] Number of data points in the train set: 16200,
number of used features: 1439
[LightGBM] [Info] Start training from score -1.386294
[LightGBM] [Info] Start training from score -1.386294
[LightGBM] [Info] Start training from score -1.386294
[LightGBM] [Info] Start training from score -1.386294
f1_macro score of the best LightGBM Classifier is:  0.6214848869119279
Best Hyperparameters for training LightGBM Classifier are:
{'learning_rate': 0.1, 'n_estimators': 150, 'num_leaves': 20}
```

```
lgbm_clf = lightgbm.LGBMClassifier(n_estimators=150,
        learning_rate=0.1, num_leaves=20 ,objective='multiclass',
        class_weight='balanced',
        force_col_wise=True,random_state=0,header=True)
lgbm_clf.fit(X_train_res, y_train_res)
```

```
[LightGBM] [Info] Total Bins 38866
[LightGBM] [Info] Number of data points in the train set: 16200,
number of used features: 1439
[LightGBM] [Info] Start training from score -1.386294
[LightGBM] [Info] Start training from score -1.386294
[LightGBM] [Info] Start training from score -1.386294
[LightGBM] [Info] Start training from score -1.386294
```

```
▾ LGBMClassifier
LGBMClassifier(class_weight='balanced', force_col_wise=True, header=True,
               n_estimators=150, num_leaves=20, objective='multiclass',
               random_state=0)
```

Checking predictions various metrics of performance:

```
y_pred = lgbm_clf.predict(X_test_combined)
clf_report = sklearn.metrics.classification_report(y_test, y_pred)
print(clf_report)
```

```
/usr/local/lib/python3.10/dist-packages/lightgbm/basic.py:1073:
UserWarning: Converting data to scipy sparse matrix.
  _log_warning('Converting data to scipy sparse matrix.')

              precision    recall  f1-score   support

  Irrelevant       0.49      0.51      0.50      1169
    Negative       0.69      0.62      0.65      2008
     Neutral       0.51      0.63      0.56      1594
    Positive       0.62      0.56      0.59      1829

    accuracy                           0.58      6600
   macro avg       0.58      0.58      0.58      6600
weighted avg       0.59      0.58      0.59      6600
```

MultinomialNB

```
from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
model.fit(X_train_res, y_train_res)
```

```
▾ MultinomialNB
MultinomialNB()
```

```
from sklearn.metrics import
        (accuracy_score,confusion_matrix,ConfusionMatrixDisplay,
        f1_score,classification_report,)
y_pred = model.predict(X_test_combined)
```

```
accuray = accuracy_score(y_pred, y_test)
f1 = f1_score(y_pred, y_test, average="weighted")
print("Accuracy:", accuray)
print("F1 Score:", f1)
```

```
Accuracy: 0.5859090909090909
F1 Score: 0.5833632180510573
```

Discussion:

we can see that pre trained model is able to predect sentimental classes more effectivly but was not able to predict the class irrevelent.

Weaknesses of the solution used: the ML model did moderately perform well than pre trained model, this is because the model is trained with the same revelent datasets but does not have weights as pretrained model because the pretrained model is trained on more realtime data than our ML model

solution is to have a hybrid model which combains the advantages of our ML model for classifying irrevelnt model and using pretrained sentimental model like roberta for classifying positive, negative, and neutral class. this would would better of both model.