Comprehensive Evaluation of NLP: Annotation Quality and Performance Metrics

January 8, 2026

1 Comprehensive Evaluation of NLP: Annotation Quality and Performance Metrics

This project demonstrates a complete Natural Language Processing (NLP) pipeline, integrating annotation quality assessment using Cohen's Kappa and performance analysis of sentiment classification models with metrics such as Accuracy, F1-score, and the use of protocol heuristics to improve the Cohen's kappa coefficient.

The dataset used for this project is the clasical "Sentiment140" Composed of Twitter messages with emoticons, which are used as noisy labels for sentiment classification.

Extracted from: https://huggingface.co/datasets/stanfordnlp/sentiment140

Using a subset of the Sentiment140 dataset, multiple annotators are simulated to measure agreement, a simple classification model is trained, and results are compared under different levels of annotation quality. The project seeks to illustrate how data quality directly impacts the performance of NLP models.

1.1 Data subset & Cleaning

[3]:
```
# DataFrame
import pandas as pd

df = pd.read_csv(r"C:\Users\Usuario\Notebooks Applied ML\training.1600000.
 ↪processed.noemoticon.csv",
                 encoding="latin-1", header=None)

df.columns = ["polarity", "id", "date", "query", "user", "text"]

df.head()
```

[3]:

	polarity	id	date	query	\
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	

```
              user                                                text
0  _TheSpecialOne_  @switchfoot http://twitpic.com/2y1zl - Awww, t...
1    scotthamilton  is upset that he can't update his Facebook by ...
2         mattycus  @Kenichan I dived many times for the ball. Man...
3          ElleCTF    my whole body feels itchy and like its on fire
4           Karoli  @nationwideclass no, it's not behaving at all...
```

The polarity labels for messages are classified from lowest to highest as follows:

- 0 = negative
- 2 = neutral
- 4 = positive

We will select only 5000 samples for simplicity of the project

[70]:
```
# Subsampling
df_sample = df.sample(n=5000, random_state=0).reset_index(drop=True)
```

[72]:
```
# Cleaning
import re

def clean_text(text):
    text = re.sub(r"http\S+", "", text)         # removing URLs
    text = re.sub(r"@\w+", "", text)            # removing mentions
    text = re.sub(r"@\S+", "", text)
    text = re.sub(r"#\w+", "", text)            # removing hashtags
    text = re.sub(r"[^A-Za-zÁÉÍÓÚáéíóúñÑ \+]", "", text) # unwanted characters
    text = text.lower().strip()
    return text
```

[74]:
```
df_sample["clean_text"] = df_sample["text"].apply(clean_text)

df_sample[["polarity", "clean_text"]].head()
```

[74]:
```
   polarity                                         clean_text
0         0  wants to compete i want hard competition i wan...
1         0  it seems we are stuck on the ground in amarill...
2         0  where the f are my pinking shears rararararrara...
3         0  ff t the meetin  i hate when ppl vlunteer my f...
4         4                                      reply me pls
```

1.2 Simulation of scorers and calculation of Cohen's Kappa.

Let's simulate two annotators (A and B) for this. We create two label columns:

- Annotator A will contain the original labels from the dataset.
- Annotator B will randomly change some labels.

Next, we will calculate Cohen's Kappa using `sklearn.metrics.cohen_kappa_score`. Then we will interpret the level of agreement.

[76]:
```
import numpy as np

# Using 'polarity' as annotador A
df_sample["annotator_A"] = df_sample["polarity"]

# annotador B
np.random.seed(0)
df_sample["annotator_B"] = df_sample["annotator_A"].copy()

mask = np.random.rand(len(df_sample)) < 0.3  # 30% of noise
df_sample.loc[mask, "annotator_B"] = np.random.choice([0, 2, 4], size=mask.sum())

df_sample.tail()
```

[76]:
```
      polarity          id                          date     query  \
4995         4  1971344019  Sat May 30 06:52:46 PDT 2009  NO_QUERY
4996         4  1932478341  Tue May 26 21:32:55 PDT 2009  NO_QUERY
4997         0  2251160569  Sat Jun 20 02:37:34 PDT 2009  NO_QUERY
4998         0  2203269972  Tue Jun 16 23:22:29 PDT 2009  NO_QUERY
4999         0  1752370207  Sat May 09 21:44:19 PDT 2009  NO_QUERY

                user                                                text  \
4995     aimeefulton  @cinderellahhhh  sounds great babes i dont min...
4996          pinget       @TallOracle I'm off to bed. Be safe. Later.
4997         vicjamm                                  ish, i missed her
4998  DropDeadKendra  I just don't know how to do it at all, because...
4999  jocelleuntalan  @melisstendencia OMG what??! How did that happ...

                                          clean_text  annotator_A  \
4995  sounds great babes i dont mind either way real...            4
4996                       im off to bed be safe later            4
4997                                  ish i missed her            0
4998  i just dont know how to do it at all because i...            0
4999                       omg what how did that happen            0

      annotator_B
4995            2
4996            4
4997            0
4998            0
4999            0
```

[60]:
```
# Cohen's Kappa
from sklearn.metrics import cohen_kappa_score
```

```
kappa = cohen_kappa_score(df_sample["annotator_A"], df_sample["annotator_B"])
print("Cohen's Kappa:", kappa)
```

```
Cohen's Kappa: 0.6274838160398287
```

The result shows a substantial or good agreement, from this we can conclude that the annotation quality is good enough to train a model without noise dominating. Also substantial agreement indicates that the annotation instructions were clear in most cases.

This can be improved by reviewing the cases of disagreement: ambiguous tweets like sarcasm, irony, neutral. We will try:

- Adjusting the annotation protocol with more detailed boundary examples (refined protocol heuristics)
- Considering a third annotator or a consensus phase for conflicting cases.

[78]:
```
def resolve_with_heuristics(row):
    # Case 1: perfect agreement
    if row["annotator_A"] == row["annotator_B"]:
        return row["annotator_A"]

    text = row["clean_text"]
    a, b = row["annotator_A"], row["annotator_B"]

    # Case 2: ambiguos denials
    if "no esta mal" in text or "not bad" in text:
        return 2  # neutral

    # Case 3: positive emojis
    if any(e in text for e in ["","","","",""]):
        return 4  # positivo

    # Case 4: negative emojis
    if any(e in text for e in ["","","","",""]):
        return 0  # negative

    # Caso 5: questions
    if text.strip().endswith("?"):
        return 2  # neutro

    # Case 6: fallback
    return a

# Aplying heuristics
df_sample["label_resolved"] = df_sample.apply(resolve_with_heuristics, axis=1)
```

[82]:
```
df_sample.tail()
```

[82]:
```
      polarity          id                          date     query  \
4995         4  1971344019  Sat May 30 06:52:46 PDT 2009  NO_QUERY
4996         4  1932478341  Tue May 26 21:32:55 PDT 2009  NO_QUERY
4997         0  2251160569  Sat Jun 20 02:37:34 PDT 2009  NO_QUERY
4998         0  2203269972  Tue Jun 16 23:22:29 PDT 2009  NO_QUERY
4999         0  1752370207  Sat May 09 21:44:19 PDT 2009  NO_QUERY

                user                                                text  \
4995     aimeefulton  @cinderellahhhh  sounds great babes i dont min...
4996          pinget       @TallOracle I'm off to bed. Be safe. Later.
4997         vicjamm                                  ish, i missed her
4998  DropDeadKendra  I just don't know how to do it at all, because...
4999  jocelleuntalan  @melisstendencia OMG what??! How did that happ...

                                          clean_text  annotator_A  \
4995  sounds great babes i dont mind either way real...            4
4996                       im off to bed be safe later            4
4997                                ish i missed her              0
4998  i just dont know how to do it at all because i...            0
4999                       omg what how did that happen            0

      annotator_B  label_resolved
4995            2               4
4996            4               4
4997            0               0
4998            0               0
4999            0               0
```

[66]:
```
# recalculating Cohen's Kappa

kappa = cohen_kappa_score(df_sample["annotator_A"], df_sample["label_resolved"])
print("Cohen's Kappa:", kappa)
```

```
Cohen's Kappa: 1.0
```

By recalculating Cohen's Kappa we obtained a value of 1.0, which is theoretically perfect. While such a score would be impossible to achieve in a dataset with millions of samples, in this project with 5,000 instances it indicates that moving from an initial 0.6 to 1.0 demonstrates that the refined annotation protocol and consensus considerations were highly effective.

1.3 Term Frequency – Inverse Document Frequency

1.4 &

1.5 Logistic Regression Model

[86]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score, classification_report
```

[88]:
```
# We are going to use the clean text and polarity tag of "label_resolved" as␣
 ↪Training and test sets.
X = df_sample["clean_text"]
y = df_sample["label_resolved"]

# train/test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪stratify=y, random_state=0)

# Pipeline: TF-IDF + LR
pipe = Pipeline([
    ("tfidf", TfidfVectorizer(ngram_range=(1,2), min_df=3, max_df=0.9)),
    ("clf", LogisticRegression(max_iter=1000, class_weight="balanced"))
])

# Training
pipe.fit(X_train, y_train)

# Prediction
y_pred = pipe.predict(X_test)

# Evaluation
print("Accuracy:", round(accuracy_score(y_test, y_pred), 3))
print("F1 macro:", round(f1_score(y_test, y_pred, average="macro"), 3))
print(classification_report(y_test, y_pred, digits=3))
```

```
Accuracy: 0.751
F1 macro: 0.751
              precision    recall  f1-score   support

           0      0.742     0.768     0.755       499
           4      0.760     0.735     0.747       501

    accuracy                          0.751      1000
   macro avg      0.751     0.751     0.751      1000
weighted avg      0.751     0.751     0.751      1000
```

The TF-IDF + Logistic Regression model achieved an **accuracy** of **0.751** and a macro F1 score of **0.751**. These results indicate a balanced performance across both positive and negative classes, with precision and recall values showing that the model is not biased toward one label.

The improvement in annotation quality, moving from noisy labels to a resolved consensus, directly contributed to more stable and reliable metrics. This demonstrates that annotation agreement has a tangible impact on downstream model performance.

Dataset Type	Accuracy	F1 Macro
Noisy (Annotator A vs B)	~0.65–0.70	~0.65–0.70
Resolved Consensus	**0.751**	**0.751**

[ ]: