CROSS VALIDATION

- **Overview**

This document describes the process of saving best machine learning models into single files and evaluating their performance on a test dataset. The models used are logistic regression from balanced data set and support vector machine for imbalanced data set

Saving Multiple Models

To efficiently manage multiple models, we stored them in dictionaries and saved them using **Joblib**, a Python library for serializing objects.

Best Models for Cross Test

The objective of this section is to evaluate and compare the best-performing models obtained from cross-validation under both balanced and imbalanced datasets based on best accuracy select

Dataset Type	Selected Model
Balanced Data	Logistic Regression
Imbalanced Data	Support Vector Machine (SVM)

Model 1: Balanced Dataset Models

CODE:

```
import joblib

# Dictionary of models
all_models = {
   'logistic_regression': log_reg,

}
```

Save models to a single file

```
joblib.dump(all_models, 'model1.pkl')
```

Model 2: Imbalanced Dataset Models

CODE :

Dictionary of models

```
all_models = {

   ‘random_forest’: rf2
}
```

Save models to a single file

```
joblib.dump(all_models, ‘model2.pkl’)
```

model1.pkl contains models trained on balanced data.

model2.pkl contains models trained on imbalanced data.

Model Evaluation

Model Evaluation (Model 1)

Each model was evaluated on the **same test dataset** using **TF-IDF features** (X_test_tfidf1) and the true labels (y_test).

Logistic Regression

Code:

```
y_pred1 = model1['logistic_regression'].predict(X_test_tfidf1)
accuracy1 = accuracy_score(y_test, y_pred1)
```

Accuracy: 0.202

Class	Precision	Recall	F1-Score	Support
1	0.17	0.33	0.22	1375
2	0.10	0.25	0.15	909
3	0.28	0.29	0.28	2246
4	0.24	0.11	0.15	1750
5	0.33	0.10	0.16	2554

Macro Avg: Precision = 0.22, Recall = 0.22, F1 = 0.19
Weighted Avg: Precision = 0.25, Recall = 0.20, F1 = 0.20

Model Evaluation (Model 2)

Random Forest

y_pred2= model2['random_forest'].predict(X_test_tfidf)

accuracy2 = accuracy_score(y_test1, y_pred2)

print(f"Model 2 Test Accuracy: {accuracy2:.4f}")

Accuracy: 0.2063

- Accuracy: 0.2063

Classification Report :

	precision	recall	f1-score	support
1	0.25	0.16	0.19	1831
2	0.07	0.00	0.01	1820
3	0.20	0.67	0.31	1792
4	0.23	0.05	0.08	1755
5	0.19	0.15	0.17	1702
accuracy			0.21	8900
macro avg	0.19	0.21	0.15	8900
weighted avg	0.19	0.21	0.15	8900

- Macro Average: Precision = 0.19, Recall = 0.21, F1 = 0.15
- Weighted Average: Precision = 0.19, Recall = 0.21, F1 = 0.15

Interpretation:

Random Forest achieves a slightly lower accuracy compared to Naive Bayes.

The model performs relatively well for Class 3, but poorly on others. This suggests the model is biased toward the dominant class and struggles with imbalance.

. Conclusion

The Model 2 (Imbalanced) evaluation shows that class imbalance significantly reduces overall performance.
Among all algorithms tested, Naive Bayes continues to demonstrate superior results due to its ability to handle sparse word distributions effectively.

Further optimization of preprocessing and model tuning can lead to notable improvements in accuracy and generalization.

TRAIN AGAIN THE MODEL FOR IMPROVE ACCURACY

Vectorization

tfidf7 = TfidfVectorizer(max_features=5000, ngram_range=(1,2))

- Converts text data into TF-IDF vectors using top **5000** features with **1-gram and 2-gram** combinations.

Model 1 (Balanced Data)

Models Trained

- Logistic Regression

Performance

Model	Accuracy	Key Observations

Logistic Regression **0.4809** Strong precision for classes 1 & 5

Saving Models

```
import joblib
model1 = {
   'logistic_regression': log_reg2,

}
joblib.dump(model1, 'model1.pkl')
```

Model 2 (Imbalanced Data)

Models Trained

- **Random Forest**

Performance

Model	Accuracy	Key Observations
Random Forest	**0.53**	

Saving Models

```
model2 = {

   'random_forest': rf3
}
joblib.dump(model2, 'model2.pkl')
```

Conclusion

- Both models use TF-IDF with bi-grams and balanced class weighting.
- Model 2 performs better overall due to better feature representation and tuned parameters.
- Models are saved as:
    - model1.pkl → Balanced training data
    - model2.pkl → Imbalanced training data