Final Project Implementation Plan

Ordinal vs Nominal Sentiment Analysis

Team: Atharv Chaudhary, Zijie Liu (Scott), Kien Nguyen

MSAI Class of 2027

1. Project Deliverables

Deliverable	Due Date	Format	Status
Proposal	Oct 27	1 page PDF	✓ Done
Pre-recorded Presentation	Dec 1	7 ± 1 min video (MP4/MOV)	Pending
Project Report	Dec 1	8 pages max, scientific paper	Pending
GitHub Repository	Dec 1	Public repo with all code	Pending

2. Professor Feedback (Must Address)

Feedback 1: Specify concrete ordinal methods — use ordinal logistic regression (proportional odds model), threshold-based approaches, or ranking loss functions.

Feedback 2: Remove fake review detection — it's scope creep that dilutes the main contribution.

3. Core Research Question

Do the performance gains from ordinal treatment justify the increased model complexity across different classification algorithms?

Key comparison: Treat 5-star ratings as ORDINAL (1 < 2 < 3 < 4 < 5) vs NOMINAL (unordered categories)

4. Work Division

Team Member	Models	Report Sections	Presentation
Scott (Zijie)	LLM / GenAI approach	Methods (GenAI) Related Works	~2 min: GenAI methodology + results

Kien	SVM or Deep Neural Network	Dataset section Results (figures/tables)	~2 min: Data processing + SVM/DNN results
Atharv	Naive Bayes + Ordinal Logistic Reg	Introduction Methods (ordinal) Discussion	~2 min: Intro + ordinal vs nominal

Shared Tasks (Everyone contributes):

- Data preprocessing pipeline (TF-IDF vectorization)
- GitHub repository setup and code integration
- Final report proofreading and formatting

5. Two-Day Implementation Timeline

DAY 1 — Implementation (Today)

Time	Task	Owner
Morning	Set up shared GitHub repo, establish folder structure	All
Morning	Sample 10,000-50,000 reviews from dataset	Kien
Morning	Create shared preprocessing pipeline (TF-IDF)	Kien
Afternoon	Implement Naive Bayes + Ordinal Logistic Regression	Atharv
Afternoon	Implement SVM or DNN with ordinal/nominal comparison	Kien
Afternoon	Implement LLM/GenAI approach	Scott
Evening	Generate confusion matrices, accuracy, MAE for all models	All
Evening	Create visualizations (class distribution, performance plots)	Kien

DAY 2 — Report & Presentation (Tomorrow)

Time	Task	Owner
Morning	Write Introduction section	Atharv
Morning	Write Dataset section + create result figures	Kien
Morning	Write Methods section (GenAI + Related Works)	Scott
Afternoon	Write Methods section (Ordinal encoding approach)	Atharv
Afternoon	Write Results section with tables	Kien
Afternoon	Finalize all code and push to GitHub	Scott
Evening	Write Discussion + Conclusion	Atharv
Evening	Proofread and format final report	All
Evening	Record individual presentation segments (~2 min each)	All
Night	Merge video segments, final submission	Scott

6. Report Structure (8 Pages Max)

Section	Content	Owner	Pages
1. Introduction	Background, research question, why ordinal matters	Atharv	~1
2. Methods	Mathematical formulas for all models • Ordinal Logistic Regression formula • Naive Bayes formula • SVM/DNN architecture • LLM approach	All	~2
3. Dataset	Amazon Reviews description, preprocessing, class distribution, train/test split	Kien	~1
4. Results	Tables: Accuracy, F1, MAE for each model Figures: Confusion matrices, performance plots	Kien	~2
5. Discussion	SOTA comparison, what worked/failed, lessons learned, future work	Atharv	~1
6. References	All citations used in text	All	~0.5
7. Appendix	GitHub link, additional plots	Scott	~0.5
Contributions	Each member's contributions listed	All	-

7. Presentation Requirements

Format: 7 ± 1 minutes pre-recorded video (MP4 or MOV)

Recording: Use Zoom or similar — must see and hear you present

Attendance: Must attend classmate presentations on Dec 9 (Zoom)

Content Structure (~2 min each person):

Person	Content	Duration
Atharv	Introduction + Research Question + Ordinal vs Nominal explanation	~2 min
Kien	Dataset + Data processing + SVM/DNN results	~2-3 min
Scott	LLM/GenAI methodology + Results + Conclusions	~2-3 min

8. Implementation Guide

SHARED: Data Loading & Preprocessing

Save this as `shared_pipeline.py` — everyone imports from here:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

# Load sampled data
df = pd.read_csv('amazon_reviews_sample.csv')

# Keep only text and rating
df = df[['text', 'rating']].dropna()
df = df[df['text'].str.len() > 10]  # Remove very short reviews

# Create labels
# NOMINAL: treat as separate classes (1, 2, 3, 4, 5)
y_nominal = df['rating'].astype(int)

# ORDINAL: same values but will be treated ordinally by specific models
y_ordinal = df['rating'].astype(int)

# TF-IDF Features
vectorizer = TfidfVectorizer(max_features=5000, stop_words='english')
X = vectorizer.fit_transform(df['text'])

# Train/Test Split (stratified)
X_train, X_test, y_train, y_test = train_test_split(
    X, y_nominal, test_size=0.2, random_state=42, stratify=y_nominal
)
```

ATHARV: Naive Bayes + Ordinal Logistic Regression

```
# pip install mord  (for ordinal regression)
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, mean_absolute_error, confusion_matrix
import mord  # Ordinal regression library

# 1. NAIVE BAYES (Nominal treatment - baseline)
nb_model = MultinomialNB()
nb_model.fit(X_train, y_train)
nb_pred = nb_model.predict(X_test)
print(f"Naive Bayes Accuracy: {accuracy_score(y_test, nb_pred):.4f}")
print(f"Naive Bayes MAE: {mean_absolute_error(y_test, nb_pred):.4f}")

# 2. LOGISTIC REGRESSION - Nominal (multinomial)
lr_nominal = LogisticRegression(multi_class='multinomial', max_iter=1000)
lr_nominal.fit(X_train, y_train)
lr_nom_pred = lr_nominal.predict(X_test)
print(f"LR Nominal Accuracy: {accuracy_score(y_test, lr_nom_pred):.4f}")
print(f"LR Nominal MAE: {mean_absolute_error(y_test, lr_nom_pred):.4f}")

# 3. ORDINAL LOGISTIC REGRESSION (Proportional Odds Model)
olr_model = mord.LogisticAT(alpha=1.0)  # All-Threshold variant
olr_model.fit(X_train.toarray(), y_train)
olr_pred = olr_model.predict(X_test.toarray())
print(f"Ordinal LR Accuracy: {accuracy_score(y_test, olr_pred):.4f}")
print(f"Ordinal LR MAE: {mean_absolute_error(y_test, olr_pred):.4f}")

# Generate confusion matrices for report
import matplotlib.pyplot as plt
import seaborn as sns

fig, axes = plt.subplots(1, 3, figsize=(15, 4))
for ax, pred, title in zip(axes,
    [nb_pred, lr_nom_pred, olr_pred],
    ['Naive Bayes', 'LR Nominal', 'LR Ordinal']):
```

```
    cm = confusion_matrix(y_test, pred)
    sns.heatmap(cm, annot=True, fmt='d', ax=ax, cmap='Blues')
    ax.set_title(title)
    ax.set_xlabel('Predicted')
    ax.set_ylabel('Actual')
plt.tight_layout()
plt.savefig('confusion_matrices_atharv.png', dpi=150)
```

KIEN: SVM Implementation

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, mean_absolute_error, classification_report

# SVM with different strategies
# 1. One-vs-Rest (OvR) - Nominal treatment
svm_ovr = SVC(kernel='linear', decision_function_shape='ovr')
svm_ovr.fit(X_train, y_train)
svm_ovr_pred = svm_ovr.predict(X_test)
print(f"SVM OvR Accuracy: {accuracy_score(y_test, svm_ovr_pred):.4f}")
print(f"SVM OvR MAE: {mean_absolute_error(y_test, svm_ovr_pred):.4f}")

# 2. One-vs-One (OvO) - Also nominal but different strategy
svm_ovo = SVC(kernel='linear', decision_function_shape='ovo')
svm_ovo.fit(X_train, y_train)
svm_ovo_pred = svm_ovo.predict(X_test)
print(f"SVM OvO Accuracy: {accuracy_score(y_test, svm_ovo_pred):.4f}")
print(f"SVM OvO MAE: {mean_absolute_error(y_test, svm_ovo_pred):.4f}")

# Class distribution plot
import matplotlib.pyplot as plt
plt.figure(figsize=(8, 5))
df['rating'].value_counts().sort_index().plot(kind='bar', color='steelblue')
plt.title('Class Distribution of Star Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.savefig('class_distribution.png', dpi=150)
```

SCOTT: LLM/GenAI Approach

Scott will implement the LLM/GenAI approach. Options include:

- Fine-tuning a pre-trained model (DistilBERT, RoBERTa)
- Using OpenAI API for few-shot classification
- Prompt engineering for ordinal prediction

9. Required Evaluation Metrics

Metric	What It Measures	Why Important
Accuracy	Overall correct predictions	Basic performance
MAE (Mean Absolute Error)	Average distance between predicted and true rating	KEY for ordinal! Lower = better ordinal treatment
F1-Score (per class)	Balance of precision/recall for each rating	Shows class imbalance issues
Confusion Matrix	Error patterns between classes	Shows adjacent rating confusion

10. Results Table Template (For Report)

Model	Encoding	Accuracy	MAE	F1 (macro)
Naive Bayes	Nominal	___	___	___
Logistic Regression	Nominal	___	___	___
Logistic Regression	Ordinal	___	___	___
SVM (OvR)	Nominal	___	___	___
SVM (OvO)	Nominal	___	___	___
LLM/GenAI	TBD	___	___	___

11. Key Hypothesis to Test

If ordinal encoding works:

- MAE should be LOWER for ordinal models
- Adjacent rating confusion (4↔5 stars) should decrease
- Accuracy might be similar, but error severity should improve

Your preliminary results showed: 62% of errors were between adjacent ratings → this supports investigating ordinal treatment!

12. Final Submission Checklist

Item	Format	Check
Report PDF	8 pages max, IEEE-style formatting	■
Video (MP4/MOV)	7 ± 1 minutes, all members visible	■
GitHub Repository	Public, contains all code + README	■
Confusion Matrices	One per model in report	■
Results Table	Accuracy, MAE, F1 for all models	■
Class Distribution Plot	Bar chart of 1-5 star counts	■

References	Properly formatted citations	■
Contributions Statement	Each member's role listed	■

13. Communication

Keep in constant contact over the next 2 days. Suggested check-ins:

- **Day 1 Evening:** Share code progress, confirm all models running
- **Day 2 Morning:** Share draft report sections for review
- **Day 2 Afternoon:** Record presentation segments
- **Day 2 Night:** Final merge and submission