📊 Notebook 2: EDA & Visualization Final Project - Ordinal vs Nominal Sentiment Analysis Atharv Chaudhary

---

Purpose: Exploratory Data Analysis and create visualizations for report.

Input: `amazon_electronics_cleaned.csv`

Output: `class_distribution.png`

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call dr
```

```
# Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

# Plot settings
plt.style.use('seaborn-v0_8-whitegrid')
sns.set_palette("husl")
plt.rcParams['figure.dpi'] = 150

print("✅ Libraries imported")
```

```
✅ Libraries imported
```

Step 1: Load Cleaned Data

```
# Load cleaned data from Notebook 1
df = pd.read_csv(r'/content/drive/MyDrive/Fall 2025/Foundations of Artifici

print(f"✅ Loaded {len(df):,} reviews")
print(f"   Columns: {list(df.columns)}")
df.head()
```

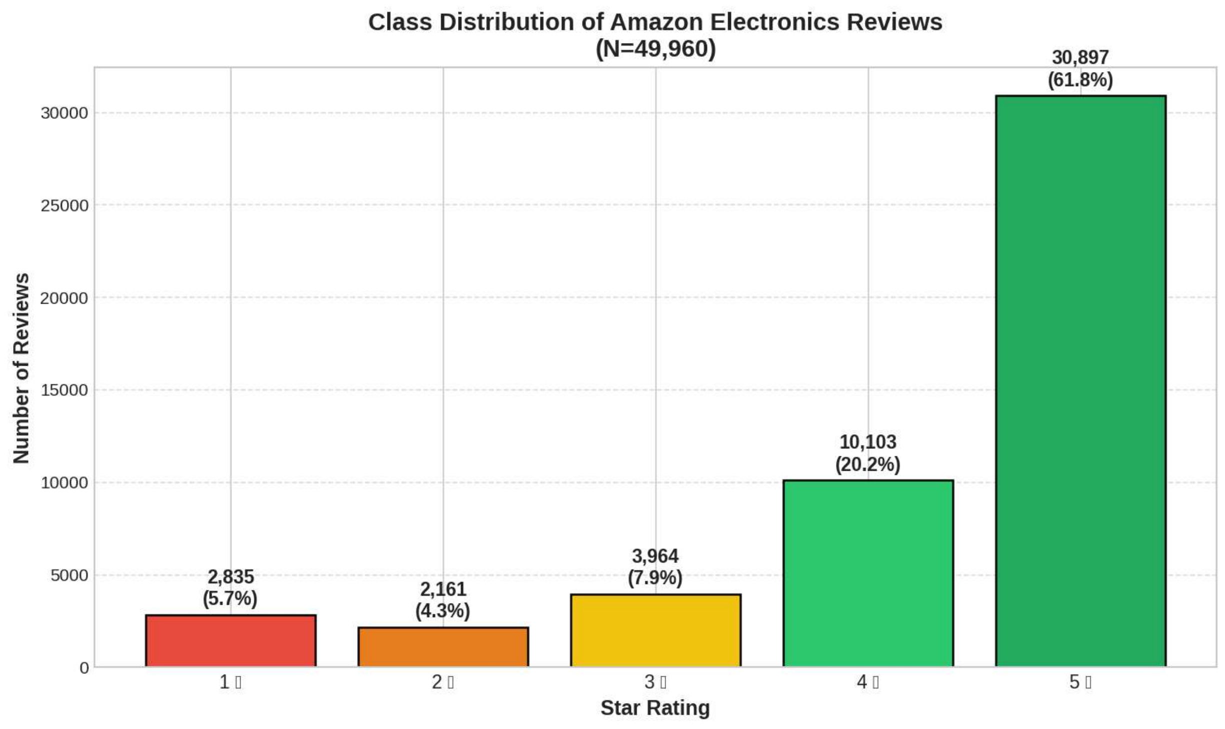
```
✅ Loaded 49,960 reviews
  Columns: ['text', 'rating']
```

	text	rating
0	We got this GPS for my husband who is an (OTR)...	5
1	I'm a professional OTR truck driver, and I bou...	1
2	Well, what can I say. I've had this unit in m...	3
3	Not going to write a long review, even thought...	2
4	I've had mine for a year and here's what we go...	1

Next steps: Generate code with df New interactive sheet

Step 2: Class Distribution Visualization

```
# ==========================================================================
# CLASS DISTRIBUTION PLOT (For Report)
# ==========================================================================

fig, ax = plt.subplots(figsize=(10, 6))

rating_counts = df['rating'].value_counts().sort_index()
colors = ['#e74c3c', '#e67e22', '#f1c40f', '#2ecc71', '#27ae60']

bars = ax.bar(rating_counts.index, rating_counts.values, color=colors,
              edgecolor='black', linewidth=1.2)

# Add value labels on bars
for bar, count in zip(bars, rating_counts.values):
    height = bar.get_height()
    ax.text(bar.get_x() + bar.get_width()/2, height + max(rating_counts)*(
            f'{count:,}\n({count/len(df)*100:.1f}%)',
            ha='center', va='bottom', fontsize=11, fontweight='bold')

ax.set_xlabel('Star Rating', fontsize=12, fontweight='bold')
ax.set_ylabel('Number of Reviews', fontsize=12, fontweight='bold')
ax.set_title(f'Class Distribution of Amazon Electronics Reviews\n(N={len(
             fontsize=14, fontweight='bold')
ax.set_xticks([1, 2, 3, 4, 5])
ax.set_xticklabels(['1 ⭐', '2 ⭐', '3 ⭐', '4 ⭐', '5 ⭐'], fontsize=1

# Add grid
ax.yaxis.grid(True, linestyle='--', alpha=0.7)
ax.set_axisbelow(True)

plt.tight_layout()
plt.savefig('/content/drive/MyDrive/Fall 2025/Foundations of Artificial I
                  facecolor='white', edgecolor='none')
plt.show()

print("\n✅ Saved: /content/drive/MyDrive/Fall 2025/Foundations of Artifi
```



```
✅ Saved: /content/drive/MyDrive/Fall 2025/Foundations of Artificial Intelligen
```

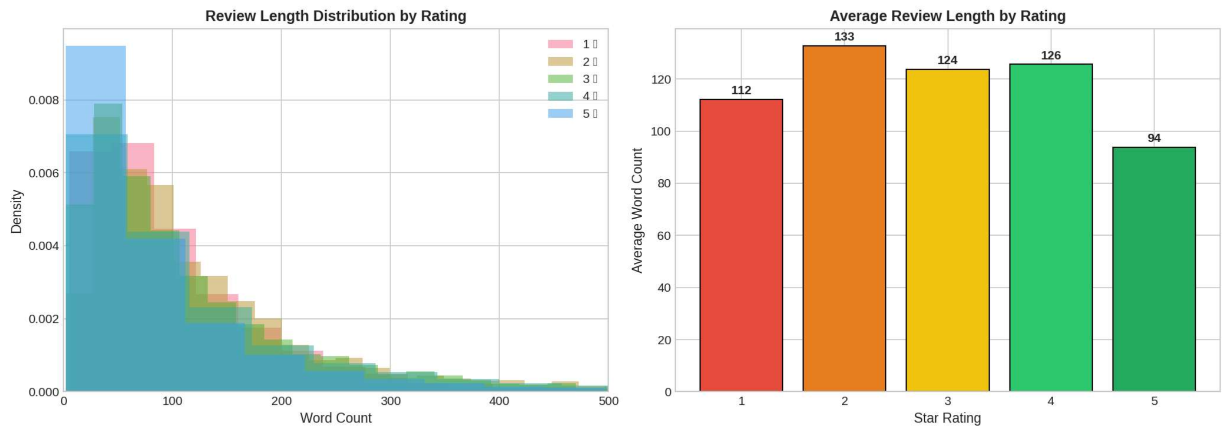
Step 3: Review Length Analysis

```
# Calculate text statistics
df['text_length'] = df['text'].str.len()
df['word_count'] = df['text'].str.split().str.len()

print("📏 Review Length Statistics:")
print(df[['text_length', 'word_count']].describe())
```

```
📏 Review Length Statistics:
        text_length    word_count
count  49960.000000  49960.000000
mean     577.486930    105.335028
```

```
std      698.808926   124.742919
min       10.000000     2.000000
25%      173.000000    32.000000
50%      337.000000    62.500000
75%      698.000000   128.000000
max    15567.000000  2845.000000
```

```
# ==============================================================================
# REVIEW LENGTH BY RATING
# ==============================================================================

# Ensure 'word_count' is available in case it was dropped prematurely
if 'word_count' not in df.columns:
    df['text_length'] = df['text'].str.len()
    df['word_count'] = df['text'].str.split().str.len()

fig, axes = plt.subplots(1, 2, figsize=(14, 5))

# Word count distribution
for rating in [1, 2, 3, 4, 5]:
    subset = df[df['rating'] == rating]['word_count']
    axes[0].hist(subset, bins=50, alpha=0.5, label=f'{rating} ⭐', density

axes[0].set_xlabel('Word Count', fontsize=11)
axes[0].set_ylabel('Density', fontsize=11)
axes[0].set_title('Review Length Distribution by Rating', fontsize=12, font
axes[0].legend()
axes[0].set_xlim([0, 500])

# Average word count by rating
avg_words = df.groupby('rating')['word_count'].mean()
bars = axes[1].bar(avg_words.index, avg_words.values, color=colors, edgecol
axes[1].set_xlabel('Star Rating', fontsize=11)
axes[1].set_ylabel('Average Word Count', fontsize=11)
axes[1].set_title('Average Review Length by Rating', fontsize=12, fontweigh
axes[1].set_xticks([1, 2, 3, 4, 5])

for bar, val in zip(bars, avg_words.values):
    axes[1].text(bar.get_x() + bar.get_width()/2, val + 2, f'{val:.0f}',
                 ha='center', fontsize=10, fontweight='bold')

plt.tight_layout()
plt.savefig('/content/drive/MyDrive/Fall 2025/Foundations of Artificial Int
plt.show()

print("\n✅ Saved: /content/drive/MyDrive/Fall 2025/Foundations of Artific
```



```
✅ Saved: /content/drive/MyDrive/Fall 2025/Foundations of Artificial Intelligen
```

Step 4: Sample Reviews

```
# Show sample reviews for each rating
print("=" * 70)
print("SAMPLE REVIEWS BY RATING")
print("=" * 70)

for rating in [1, 2, 3, 4, 5]:
    sample = df[df['rating'] == rating]['text'].iloc[0][:200]
    print(f"\n{'⭐' * rating} ({rating}-star):")
    print(f"   {sample}...")
```

```
======================================================================
SAMPLE REVIEWS BY RATING
======================================================================

⭐ (1-star):
   I'm a professional OTR truck driver, and I bought a TND 700 at a truck stop h

⭐⭐ (2-star):
   Not going to write a long review, even thought this unit deserves one. I've d

⭐⭐⭐ (3-star):
   Well, what can I say.  I've had this unit in my truck for about four days now

⭐⭐⭐⭐ (4-star):
   This adapter easily connects my Nook HD 7&#34; to my HDTV through the HDMI ca

⭐⭐⭐⭐⭐ (5-star):
    We got this GPS for my husband who is an (OTR) over the road trucker.  Very I
```

Step 5: Key Insights

```python
# ===============================================================================
# KEY INSIGHTS FOR REPORT
# ===============================================================================

print("=" * 70)
print("📋 KEY INSIGHTS FOR REPORT")
print("=" * 70)

rating_counts = df['rating'].value_counts().sort_index()

print(f"""
1. DATASET SIZE:
   - Total reviews: {len(df):,}
   - Source: Amazon Electronics Reviews (McAuley Lab, UCSD)

2. CLASS IMBALANCE:
   - 5-star reviews: {rating_counts[5]:,} ({rating_counts[5]/len(df)*100:.1
   - 1-star reviews: {rating_counts[1]:,} ({rating_counts[1]/len(df)*100:.1
   - Imbalance ratio: {rating_counts[5]/rating_counts.min():.1f}:1

3. REVIEW LENGTH:
   - Average words: {df['word_count'].mean():.0f}
   - Median words: {df['word_count'].median():.0f}
   - Negative reviews tend to be longer (more detail about complaints)

4. CHALLENGE FOR CLASSIFICATION:
   - Adjacent ratings (4★ vs 5★) use similar vocabulary
   - Class imbalance affects minority class performance
   - Ordinal structure: 1 < 2 < 3 < 4 < 5
""")
```

```
======================================================================
📋 KEY INSIGHTS FOR REPORT
======================================================================

1. DATASET SIZE:
   - Total reviews: 49,960
   - Source: Amazon Electronics Reviews (McAuley Lab, UCSD)

2. CLASS IMBALANCE:
   - 5-star reviews: 30,897 (61.8%)
   - 1-star reviews: 2,835 (5.7%)
   - Imbalance ratio: 14.3:1

3. REVIEW LENGTH:
   - Average words: 105
   - Median words: 62
   - Negative reviews tend to be longer (more detail about complaints)

4. CHALLENGE FOR CLASSIFICATION:
   - Adjacent ratings (4★ vs 5★) use similar vocabulary
   - Class imbalance affects minority class performance
   - Ordinal structure: 1 < 2 < 3 < 4 < 5
```

```
# Clean up helper columns
df = df.drop(columns=['text_length', 'word_count'], errors='ignore')

# Images are now saved directly to Google Drive, no need for separate downl
print("Images have been saved directly to Google Drive.")
```

```
Images have been saved directly to Google Drive.
```

```
# This cell is no longer needed, as images are now saved directly to Google Drive by the plt.savefig commands.
print("Images have been saved directly to Google Drive, no further move operation is required.")
```

```
Images have been saved directly to Google Drive, no further move operation is re
```

✅ Summary

Visualizations created:

- `class_distribution.png` - For Dataset section of report
- `review_length_analysis.png` - Additional analysis

Key findings:

- Severe class imbalance (5-star dominant)
- Negative reviews are longer on average
- Ordinal structure should be leveraged

Next: Run `3_Models_Nominal.ipynb`