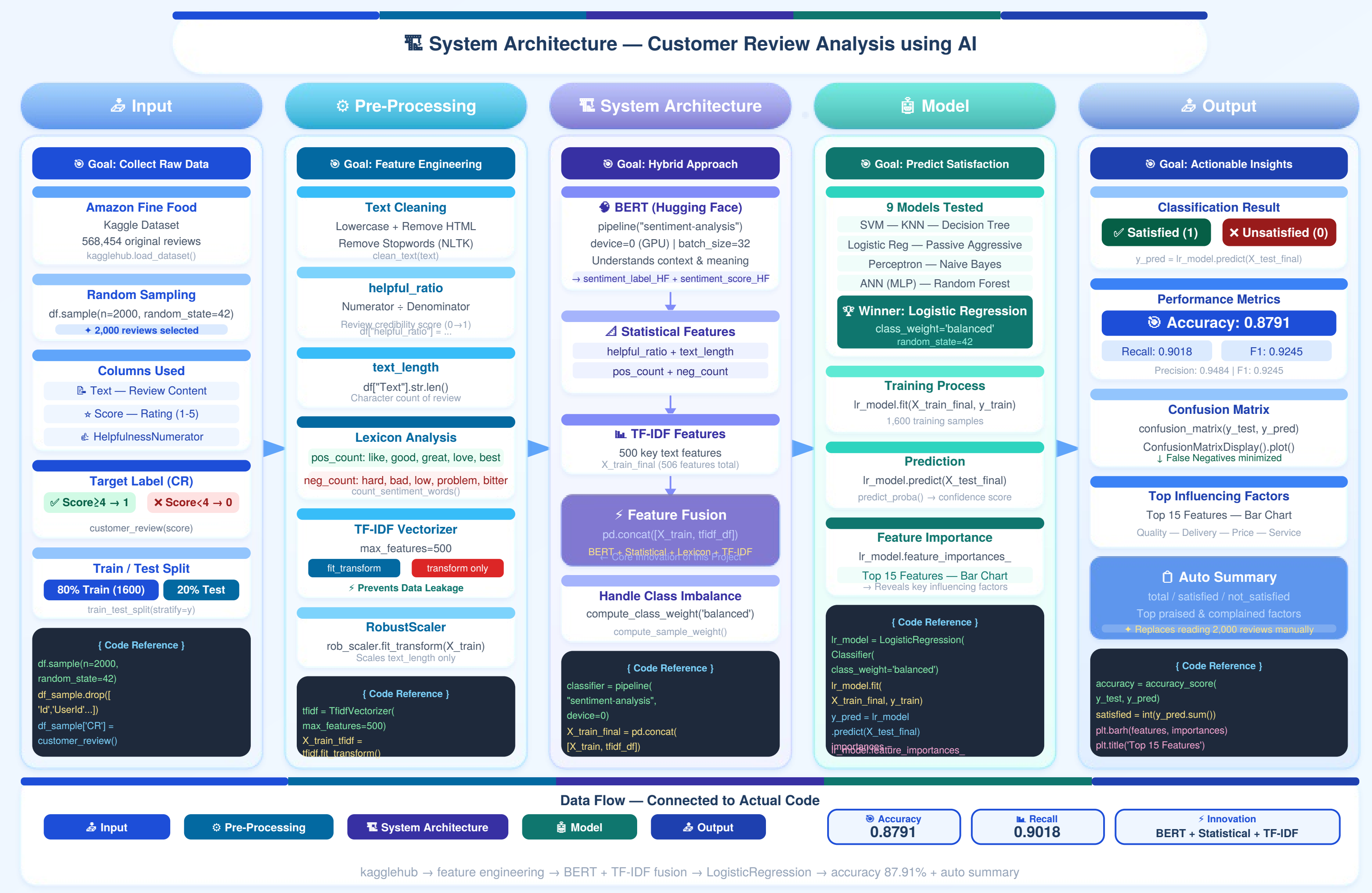

System Architecture — Customer Review Analysis using AI
Input
Pre-Processing
System Architecture
Model
Output
Goal: Collect Raw Data
Amazon Fine Food
Kaggle Dataset
568,454 original reviews
kagglehub.load_dataset()
Random Sampling
df.sample(n=2000, random_state=42)
✦ 2,000 reviews selected
Columns Used
Text — Review Content
Score — Rating (1-5)
HelpfulnessNumerator
Target Label (CR)
Score≥4 → 1
Score<4 → 0
customer_review(score)
Train / Test Split
80% Train (1600)
20% Test
train_test_split(stratify=y)
{ Code Reference }
df.sample(n=2000,
random_state=42)
df_sample.drop([
'Id','UserId'...])
df_sample['CR'] =
customer_review()
Goal: Feature Engineering
Text Cleaning
Lowercase + Remove HTML
Remove Stopwords (NLTK)
clean_text(text)
helpful_ratio
Numerator ÷ Denominator
Review credibility score (0→1)
df["helpful_ratio"] = ...
text_length
df["Text"].str.len()
Character count of review
Lexicon Analysis
pos_count: like, good, great, love, best
neg_count: hard, bad, low, problem, bitter
count_sentiment_words()
TF-IDF Vectorizer
max_features=500
fit_transform
transform only
Prevents Data Leakage
RobustScaler
rob_scaler.fit_transform(X_train)
Scales text_length only
{ Code Reference }
tfidf = TfidfVectorizer(
max_features=500)
X_train_tfidf =
tfidf.fit_transform()
Goal: Hybrid Approach
BERT (Hugging Face)
pipeline("sentiment-analysis")
device=0 (GPU) | batch_size=32
Understands context & meaning
→ sentiment_label_HF + sentiment_score_HF
Statistical Features
helpful_ratio + text_length
pos_count + neg_count
TF-IDF Features
500 key text features
X_train_final (506 features total)
Feature Fusion
pd.concat([X_train, tfidf_df])
BERT + Statistical + Lexicon + TF-IDF
← Core Innovation of this Project
Handle Class Imbalance
compute_class_weight('balanced')
compute_sample_weight()
{ Code Reference }
classifier = pipeline(
"sentiment-analysis",
device=0)
X_train_final = pd.concat(
[X_train, tfidf_df])
Goal: Predict Satisfaction
9 Models Tested
SVM — KNN — Decision Tree
Logistic Reg — Passive Aggressive
Perceptron — Naive Bayes
ANN (MLP) — Random Forest
Winner: Logistic Regression
class_weight='balanced'
random_state=42
Training Process
lr_model.fit(X_train_final, y_train)
1,600 training samples
Prediction
lr_model.predict(X_test_final)
predict_proba() → confidence score
Feature Importance
lr_model.feature_importances_
Top 15 Features — Bar Chart
→ Reveals key influencing factors
{ Code Reference }
lr_model = LogisticRegression(
Classifier(
class_weight='balanced')
lr_model.fit(
X_train_final, y_train)
y_pred = lr_model
.predict(X_test_final)
importances =
lr_model.feature_importances_
Goal: Actionable Insights
Classification Result
Satisfied (1)
Unsatisfied (0)
y_pred = lr_model.predict(X_test_final)
Performance Metrics
Accuracy: 0.8791
Recall: 0.9018
F1: 0.9245
Precision: 0.9484 | F1: 0.9245
Confusion Matrix
confusion_matrix(y_test, y_pred)
ConfusionMatrixDisplay().plot()
↓ False Negatives minimized
Top Influencing Factors
Top 15 Features — Bar Chart
Quality — Delivery — Price — Service
Auto Summary
total / satisfied / not_satisfied
Top praised & complained factors
✦ Replaces reading 2,000 reviews manually
{ Code Reference }
accuracy = accuracy_score(
y_test, y_pred)
satisfied = int(y_pred.sum())
plt.barh(features, importances)
plt.title('Top 15 Features')
Data Flow — Connected to Actual Code
Input
Pre-Processing
System Architecture
Model
Output
Accuracy
0.8791
Recall
0.9018
Innovation
BERT + Statistical + TF-IDF
kagglehub → feature engineering → BERT + TF-IDF fusion → LogisticRegression → accuracy 87.91% + auto summary