# ML System Design Document – Yelp Review Rating Prediction

## 1. Problem Statement

The goal of this project is to design and implement an end-to-end machine learning pipeline using Amazon SageMaker to predict Yelp business review ratings based on textual reviews and metadata. Yelp reviews play a major role in influencing consumer decisions and business reputation. However, manual moderation or analysis of large-scale review data is infeasible. This project aims to automate the process of predicting sentiment and star ratings from customer reviews using scalable ML solutions.  
  
The problem specifically focuses on developing a robust, scalable ML pipeline that preprocesses text data, extracts meaningful features, and trains a classification model capable of predicting Yelp review ratings accurately. The system should be deployable in real-time or batch inference environments, enabling business intelligence and customer experience analytics at scale.

## 2. Measuring Impact

The impact of the project will be measured in two dimensions—model performance and business utility.  
  
From a technical standpoint, the primary metric is accuracy in predicting Yelp review ratings (1–5 stars). Supplementary metrics include precision, recall, F1-score, and confusion matrix to assess balanced performance across all classes. During model training, performance improvements were tracked across multiple model iterations and feature extraction techniques.  
  
From a business perspective, impact is measured by how well the model improves insight extraction and decision-making efficiency. By accurately predicting customer satisfaction from text, the model helps identify service strengths and weaknesses, enhances review analytics, and supports automated moderation. Over time, the measurable KPIs will include reduced manual review costs and improved satisfaction analytics for business stakeholders.

## 3. Security, Bias, and Ethical Considerations

Data Security:  
All data operations are performed within the AWS ecosystem, leveraging Amazon S3, AWS IAM, and SageMaker Studio. Access to data is controlled through IAM roles with least-privilege policies. Sensitive user identifiers are excluded or anonymized before model training. Communication between SageMaker components is secured via TLS.  
  
Bias and Fairness:  
Yelp reviews may contain linguistic or demographic biases. The model could inherit such biases if not mitigated. Preprocessing steps include text normalization and balancing of training samples across rating categories to prevent skewed learning toward dominant classes (e.g., 4–5-star reviews).  
  
Ethical Concerns:  
Ethical considerations include avoiding misuse of the model for manipulating or filtering user opinions. Transparency and explainability are encouraged by analyzing model feature importance and ensuring that predictions are used for insights, not censorship.

## 4. Solution Overview

### A. Data Sources

The dataset originates from Yelp Open Dataset, containing millions of business reviews with fields such as review\_id, user\_id, business\_id, stars, and text. For this project, a subset was used for efficient experimentation.  
Data was uploaded to an S3 bucket and accessed directly from SageMaker Processing jobs. Each record includes the review text (input feature) and the corresponding star rating (label).

### B. Data Engineering

Data preprocessing was automated using a SageMaker Processing Job powered by a Scikit-learn container. The steps included:  
- Loading raw data from S3.  
- Cleaning the text by removing stopwords, punctuation, and special symbols.  
- Converting text to lowercase and performing tokenization.  
- Splitting the dataset into training, validation, and test subsets (80/10/10).  
- Persisting processed data as sparse matrices in S3 for downstream tasks.  
  
This stage ensures a consistent, version-controlled data pipeline that can be re-run with new data using the same preprocessing logic.

### C. Feature Engineering

Feature extraction was implemented using TF-IDF vectorization to convert textual data into numerical representations.  
The Scikit-learn TfidfVectorizer was configured to extract the top N terms based on frequency and inverse document frequency to reduce dimensionality while retaining semantic richness.  
  
Key configurations included:  
- max\_features=5000 to balance performance and interpretability.  
- ngram\_range=(1,2) to capture both unigrams and bigrams.  
- min\_df=5 to remove infrequent tokens.  
  
The resulting sparse matrices were serialized using Joblib and stored alongside metadata for reproducibility.

### D. Model Training & Evaluation

The training phase utilized Amazon SageMaker Training Jobs running a Logistic Regression classifier for multi-class prediction of Yelp review stars.  
Parameters included:  
- C=1.0 (regularization strength)  
- max\_iter=1000  
- Solver = liblinear  
  
Training leveraged CPU instances (e.g., ml.m5.xlarge), and models were serialized as .tar.gz artifacts stored in S3.  
Evaluation metrics were computed using the test dataset and included accuracy, precision, recall, and F1-score. Results showed that the Logistic Regression model achieved high interpretability and consistent performance across categories, making it suitable for production deployment.

### E. Model Deployment

Deployment was implemented using a SageMaker Endpoint configured for real-time inference.  
The trained model artifact and TF-IDF vectorizer were deployed as an inference pipeline.  
Requests are handled via SageMaker Runtime using JSON input:  
  
{  
 "texts": [  
 "Amazing service and delicious food. Will come back!",  
 "The burger was dry and the fries were soggy. Not impressed."  
 ]  
}  
  
The endpoint returns predicted ratings (1–5). This real-time deployment supports low-latency predictions and scales automatically with traffic.

### F. Model Monitoring

Continuous monitoring was set up using SageMaker Model Monitor.  
Metrics include:  
- Request/response latency  
- Data drift (comparing live data with training data distributions)  
- Model accuracy on sampled ground-truth feedback  
  
Alerts are triggered when data drift exceeds a defined threshold. Logs are stored in CloudWatch for traceability and debugging.

### G. CI/CD

A full CI/CD workflow was implemented using AWS CodePipeline, CodeBuild, and SageMaker Projects.  
Pipeline stages:  
1. Source: Pulls notebook and script updates from GitHub.  
2. Build: Executes tests, packaging, and container validation.  
3. Train: Launches SageMaker Training Job with new data.  
4. Deploy: Automates endpoint update upon model approval.  
5. Monitor: Integrates with CloudWatch for automated metric tracking.  
  
This ensures consistent, version-controlled ML releases, minimizing manual intervention and risk of model drift.

## 5. Conclusion

The Yelp Review Rating Prediction System demonstrates an end-to-end ML solution from raw data ingestion to deployment and monitoring using Amazon SageMaker. The design emphasizes scalability, automation, and responsible AI practices.  
By combining TF-IDF feature extraction, Logistic Regression modeling, and AWS-native CI/CD orchestration, the system provides a repeatable and production-ready framework for large-scale text classification tasks.