## **Yelp Review Rating Prediction – ML System Design Document**

## **Problem Statement**

Yelp reviews hold significant business value for consumer analytics and reputation management. The project's core goal is to automate the prediction of Yelp business review ratings (1-5 stars) from text and metadata, leveraging Amazon SageMaker for scalability and reproducibility. Manual assessment of massive review datasets is impractical, so this solution enables real-time or batch inference with explainable machine learning models. The business need is to extract actionable insights for business owners and analysts, supporting an improved customer experience and operational efficiency.

## **Measuring Impact**

Impact is two-fold. Technical impact: Evaluate star prediction accuracy and macro-F1 score, confusion matrix, and per-class precision/recall—these metrics are tracked and reported for every model version. Business metrics: Reduction of manual review cost, faster review moderation, and richer analytics for stakeholders. Inference is validated by sampled reviews; endpoints are monitored for drift; and stakeholder KPIs are linked to accuracy over time for ongoing improvement.

## **Security, Bias, and Ethical Considerations**

All operations reside in secure AWS infrastructure, using IAM and S3 privacy policies. Sensitive identifiers are always anonymized. Communication is encrypted (TLS). Bias is tackled by balancing rating categories during training and text normalization. Model interpretability is available using feature importances and optional SHAP. Ethical use includes not censoring or manipulating reviews, only providing analytics. Monitoring tracks drift and ensures ongoing fairness.

## **Solution Overview**

## **Data Sources**

Yelp Open Dataset (reviews.csv, business fields) stored in S3. Fields include reviewid, userid, businessid, stars, and text. Subsampling is used for experimentation; full-scale runs possible for production. Processing jobs ingest the S3 data directly into SageMaker pipelines.

## **Data Engineering**

Raw review texts are loaded from S3, cleaned (lowercase, punctuation/stopword removal, tokenization), and split 80/10/10 for train/valid/test using stratified sampling. Artifacts are saved to S3 as sparse matrices. Helper scripts automate processing logic and support version-controlled execution and reuse on new incoming batches.

## **Feature Engineering**

TF-IDF vectorization maps text into sparse matrices of ngram counts, with max 30,000 features and (1,2)-gram range. Infrequent terms are dropped to avoid outlier bias. All vectorizers and parameters (e.g., min-df, stopwords) are persisted for reproducibility and used for inference on new review data. KMeans clustering is optionally performed for unsupervised insight into review structure (food types, sentiments). Feature Store integration is available for further real-time analytics.

## **Model Training & Evaluation**

Logistic Regression (multiclass, multinomial) is trained using Scikit-learn on CPU SageMaker instances (ml.m5.large/xlarge). Key hyperparameters: C=1.0 (regularization), max\_iter=1000. Evaluation uses macro-F1 and accuracy on held-out test data. Confusion matrix is computed for per-class analysis; and classification report tracks all precision/recall scores. Helper scripts automate training and ensure artifact versioning and reproducibility.

## **Model Deployment**

Endpoint is deployed using SageMaker SKLearnModel API, with both model and TF-IDF vectorizer. Endpoint takes JSON text, outputs star ratings (1–5), and persistence/scaling managed automatically. Deployment supports real-time scoring and integrates with Model Monitor for data drift detection.  
Sample predictions:

* *"Amazing service and delicious food. Will come back!"* → 5 stars
* *"The burger was dry and the fries were soggy. Not impressed."* → 2 stars

## **Model Monitoring**

Model Monitor baseline statistics/constraints are established on initial training set. Production data is monitored for drift and cloud metrics (latency, accuracy) are tracked via CloudWatch. Alerts are triggered for abnormal drift or constraint violations. Monitoring schedules run hourly and logs are kept for traceability and debugging.

## **CI/CD**

AWS CodePipeline/CodeBuild orchestrate end-to-end ML process from notebook to S3, building, training, deploying, and monitoring with each code or dataset update. CICD scripts auto-trigger retraining and endpoint updates and validate all artifacts. This minimizes manual intervention and supports scalable, reproducible ML life cycle.

## **Prediction Results & Conclusion**

Test Example 1:  
*"Amazing service and delicious food. Will come back!"*Predicted Rating: 5 stars, Probability: 82% for 5 stars

Test Example 2:  
*"The burger was dry and the fries were soggy. Not impressed."*Predicted Rating: 2 stars, Probability: 54% for 2 stars

Conclusion:  
The designed system delivers automated textual review analytics at production scale, with robust security and ethical guards. The ML pipeline enables rapid deployment and high-throughput scoring for Yelp consumer analytics. Model accuracy and macro-F1 (see notebook for details) are strong, and confusion matrix reveals balanced performance. End-to-end monitoring and CI/CD ensure responsible adaptation to new data. Prediction examples highlight the interpretability and usefulness for business owners.