Milestone 3 Submission

Caroline Amsbary [*1] **Emily Friedman** [*1] **Suhanee Singh** [*1]
[2] [3 4]

1. Milestone 1

1.1. Dataset Overview

Local county health departments conduct inspections of restaurants and other retail food establishments to ensure that employees practice safe food handling and that kitchen facilities meet required standards. The data we will be using is **Restaurant Inspection Scores in the state of Washington**. We have chosen to focus on **King County**, as this is the county where Seattle is located, yielding results from a higher population.

The dataset contains Health Department inspection results for food service establishments in King County from 2006 to the present. The information is organized by Business, Inspection Date, and Violation. Each row represents a single inspection.

Source: `https://kingcounty.gov/en/dept/dph/health-safety/food-safety/search-restaurant-safety-ratings#/`

1.2. Variable Description

To clean this data in preparation for analysis, we can start by creating a new table with only the relevant variables. As of right now, here are the variables provided in the dataset:

- **Name:** Name of the restaurant
- **Program Identifier:** Name of the restaurant (same as Name)
- **Inspection Date:** Date that inspection took place
- **Description:** Describes the restaurant's seating capacity and Risk Category, which is associated with the level of food safety risk (I being lowest risk, IV being highest)
- **Address:** Restaurant's address
- **City:** Restaurant's city
- **Zip Code:** Restaurant's zip code
- **Phone:** Restaurant's phone number
- **Longitude:** Restaurant's longitude
- **Latitude:** Restaurant's latitude
- **Inspection Business Name:** Name of the restaurant (same as Name)
- **Inspection Type:** What kind of inspection was performed
- **Inspection Score:** The score given by the inspector (0 is ideal; higher is worse)
- **Inspection Result:** For routine inspections, either "Satisfactory" or "Unsatisfactory"; for consultation/education inspections, either "Complete" or "Incomplete"
- **Inspection Closed Business:** "True" if the inspection caused closure; "False" if the restaurant remained open
- **Violation Type:** "Red" for high-risk violations and "Blue" for low-risk violations
- **Violation Description:** Detailed explanation of the food safety issue identified
- **Violation Points:** Discrete points (0–30) indicating severity of violation
- **Business_ID:** Unique identifier for each establishment
- **Inspection_Serial_Num:** Unique tracking number for each inspection
- **Violation_Record_ID:** Unique identifier for a specific violation
- **Grade:** Numeric code for severity: 1 = Critical, 2 = Intermediate, 3 = Core

1.3. Selected Variables for Analysis

For our analysis, we have determined that we only need the following variables:

- Name
- Inspection Date
- Description

- Inspection Type
- Inspection Score
- Inspection Result
- Inspection Closed Business
- Violation Type
- Violation Description
- Violation Points
- Grade

1.4. Data Cleaning Process

To clean the data, we will replace missing data with `NaN`, although as far as we have seen, there is no missing data. This is hard to verify just from visual inspection, since there are over 277,000 rows. Additionally, we will ensure that numerical variables are stored as `int` or `float`, which is essential for statistical summaries and plotting.

1.5. Project Goal

We aim to estimate the probability that a restaurant's next inspection in King County, Washington will include at least one **critical ("red") violation**. The model is intended to support risk-based scheduling and targeted education by identifying establishments most likely to require intervention.

2. Milestone 2: Preliminary Methods

2.1. Method Overview

Per Milestone 1, the aim of this project is to examine the restaurants in King County, Washington, in particular their previous inspection results, to predict the likelihood that the next inspection will include at least one critical violation. We plan to achieve this by using the past inspection results to explore patterns and write predictive models using some of the key methods covered in class, specifically kNN, linear models, and decision trees. Our approach will follow this general overview, all of which are discussed in more detail below:

1. Step 1: Data wrangling and cleaning
2. Step 2: EDA and visualization of cleaned data
3. Step 3: Feature engineering and selection
4. Step 4: Model training
5. Step 5: Model validation

2.2. Data Wrangling and Cleaning

Again, as mentioned in Milestone 1, we will begin by creating a new dataset that includes only the relevant variables for our analysis:

- Name
- Inspection Date
- Description (includes Risk Category)
- Inspection Type
- Inspection Score
- Inspection Result
- Inspection Closed Business
- Violation Type
- Violation Description
- Violation Points
- Grade
- Longitude
- Latitude

**Note that we added longitude and latitude to the list of relevant variables, as we decided this is something we'd like to take into consideration.*

Cleaning steps will include:

- Replacing missing data with NaN values and confirming completeness.
- Converting numeric variables such as Inspection Score, Violation Points, and Grade to numeric data types (int or float).
- Ensuring that Inspection Date is properly formatted as a datetime object.
- Removing duplicate or irrelevant records (for example, entries where inspection data are incomplete).
- Encoding categorical variables (for example, Inspection Type, Violation Type, Inspection Result) using one-hot encoding.

This cleaned dataset will serve as the foundation for exploratory analysis and model development.

2.3. Exploratory Data Analysis and Visualization

We will use EDA techniques to understand the distribution and discover which relationships, trends, or anomalies exist in the data. Specifically, we plan to:

- Plot the distribution of Inspection Scores and Violation Points to assess their skewness and range.
- Examine how Risk Category (from Description) relates to the frequency of red violations.
- Use bar charts to show the proportion of inspection results ("Satisfactory" vs. "Unsatisfactory") by restaurant type.
- Explore geographic patterns using longitude and latitude to see whether certain areas of King County experience more frequent red violations.

These visualizations will help identify potential predictive features and highlight patterns that could inform our model choices.

2.4. Feature Engineering

In this step, we will perform feature engineering to simplify the model, reduce overfitting, and speed up our computations. From the raw inspection records, we will generate higher-level features such as:

- Average past inspection score: captures a restaurant's overall performance trend
- Number of previous inspections: reflects inspection frequency, which may correlate with compliance consistency
- Proportion of inspections with red violations: indicates the severity or recurrence of health risks
- Average violation points per inspection: provides a measure of violation intensity
- Time since last inspection: represents temporal risk, as longer gaps may be associated with increased likelihood of violations

These features will help capture each restaurant's inspection history and risk level, providing more in-depth and relevant information for our predictive models.

POTENTIAL CODE for feature engineering:

```
import pandas as pd

# Assume df is the raw inspection dataset
df['Inspection Date'] = pd.to_datetime(df['
    ↪ Inspection Date'])

# Sort by date to ensure correct temporal
    ↪ ordering
df = df.sort_values(['Name', 'Inspection
    ↪ Date'])

# Group by restaurant name
features = df.groupby('Name').apply(lambda
    ↪ x: pd.Series({
    'avg_past_score': x['Inspection Score'
        ↪ ].mean(),
    'num_prev_inspections': x['Inspection
        ↪ Date'].nunique(),
    'prop_red_violations': (x['Violation
        ↪ Type']
        .str.contains('red', case=False, na
            ↪ =False)).mean(),
    'avg_violation_points': x['Violation
        ↪ Points'].mean(),
    'time_since_last_insp': (x['Inspection
        ↪ Date'].max()
        - x['Inspection Date'].min()).days
})).reset_index()
```

2.5. Modeling Approach

In order to predict the likelihood of future red violations, we will set up a binary classification problem, where the response variable Y, is defined as:

$$Y = \begin{cases} 1, & \text{if the next inspection contains} \geq 1 \text{ violation} \\ 0, & \text{otherwise} \end{cases}$$

We plan to train and compare the following models:

- Linear Models (Logistic Regression): To serve as a simple and easily interpretable baseline for predicting the probability of a red violation. Logistic regression will allow us to examine the weight of each feature and identify the most significant predictors.
- k-Nearest Neighbors (k-NN): To predict a restaurant's risk by comparing it to other restaurants that are most similar in the data. In other words, it looks at the "k" closest restaurants and uses their inspection outcomes to make a prediction. We will try different values of k to see how changing the number of neighbors affects the model's accuracy and recall.
- Decision Trees: Predict outcomes by splitting the data into branches based on different features, like inspection type or risk category. This will show how certain combinations of factors can lead to red violations. Decision trees are also easy to understand because they create a clear visual diagram that shows how decisions are made step by step.

Additionally, we will use unsupervised clustering (such as k-means) as an exploratory step to see whether restaurants naturally form groups with distinct inspection patterns (like low-risk vs. high-risk clusters).

2.6. Model Training and Validation

We will split the data into training (70%) and testing (30%) groups. To avoid data leakage, all inspections from the same restaurant will be kept in only one of these sets. For each model, we will:

- Use cross-validation on the training data to confirm the best settings (for example, the number of neighbors in k-NN or the maximum depth in a decision tree)
- Test how well the model performs on the test set using accuracy, precision, recall, F1-score, and ROC-AUC

F1-score = balance between precision and recall

ROC-AUC = Receiver Operating Characteristic and Area Under Curve, tell us how well the model separates the two possible outcomes

After this, we will compare all models to find which one provides the best balance between accuracy and ease of interpretation.

2.7. Preliminary Validation Plan

Some ways we will visualize performance is through confusion matrices to show how many inspections were correctly and incorrectly classified. Another way is through feature importance plots (for decision trees and linear models) to interpret which variables contribute most strongly to predictions. We also hope to test the models on data from different time periods (e.g., pre-2018 vs. post-2018) to evaluate their stability over time.

2.8. Future Work

Looking forward to the Final Report we plan to refine our models by:

- Incorporating temporal trends (seasonality or time since last inspection).
- Testing models that combine multiple techniques.
- Building visual dashboards to display predicted risk levels by restaurant or neighborhood.

2.9. Summary

Our preliminary methodology integrates the data wrangling, EDA, visualization, and many other methods introduced in class. By combining interpretable models such as logistic regression with flexible approaches like k-NN and decision trees, we aim to identify reliable predictors of critical violations and support food safety efforts in King County, Washington.

3. Milestone 3: Results

3.1. Data Wrangling and Cleaning

We first loaded the raw food inspection dataset and converted key fields into appropriate formats. The *Inspection Date* was transformed into a datetime type, and the numeric fields such as *Inspection Score*, *Violation Points*, *Grade*, *Longitude*, and *Latitude* were mapped to numeric for analysis.

Missing violation fields typically indicate that no violation occurred, so we filled these with `"None"` and set missing violation points to `0`. We also created a binary feature, `has_red_violation`, to capture whether an inspection included any red violations. Risk Category was extracted from the text *Description* using a regular expression, and inspections without an explicitly listed category were labeled `"Unknown"`. Missing grades were replaced with `-1`, as a lack of grade usually reflects that none were issued (rather than poor data).

To ensure unique and valid records, we removed rows missing a *Business_ID* or *Inspection Date*, and dropped columns not relevant for modeling (*Phone*, *Address*, etc.). Because the original data set contains one row per violation, we aggregated the data so that each row represents an inspection, retaining relevant details while summing violation points and indicating whether any red violations occurred.

Finally, we consolidated establishment types into clearer categories (e.g., grouping different seating capacities together as `"Restaurant (Seating)"` and mapping vague `"Non"` entries to `"Non-Seating"`). We also simplified inspection outcomes into two categories — `"Satisfactory"` and `"Unsatisfactory/Other"` — to better support classification modeling.

This process produced a cleaned inspection-level dataset (`df_inspections`) ready for exploratory analysis and predictive modeling.

3.2. Exploratory Data Analysis and Visualization

We conducted exploratory analysis to understand the distribution of inspection outcomes and to identify predictors that might be useful for modeling. Figure 1 shows that inspection scores are extremely right-skewed, meaning that most restaurants receive scores near zero, with a long tail of higher-score inspections. This indicates that most establishments perform well on routine inspections.

We also examined how the county's risk classification relates to severe violations. As shown in Figure 2, higher risk categories (Risk Categories I–III) exhibit substantially higher proportions of red violations, suggesting that the official risk category is a strong predictor of future critical violations.

Finally, we explored geographic patterns across King County. Figure 3 visualizes the spatial distribution of inspections. Red violations occur throughout the county but cluster more densely in urban commercial areas such as central Seattle. This cluster supports the inclusion of latitude and longitude in the predictive model.

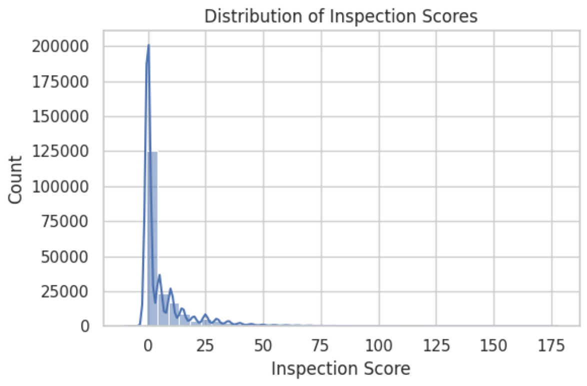

Figure 1. Distribution of inspection scores across all inspections.

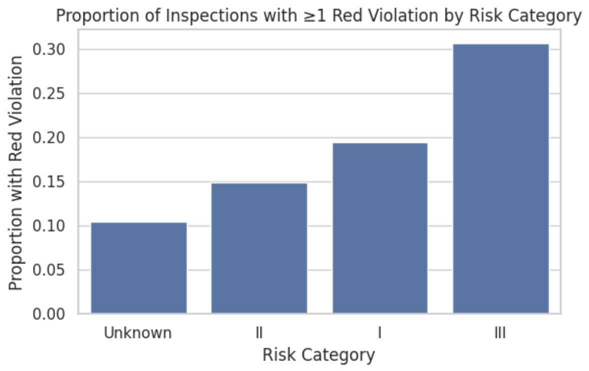

Figure 2. Proportion of inspections with at least one red violation by official risk category.

3.3. Feature Engineering

To build a restaurant-level prediction dataset, we aggregated all historical inspections for each establishment and constructed a set of summary features designed to capture past performance and violation patterns. For each restaurant, we computed: (1) the average past inspection score (`avg_past_score`), (2) the number of previous inspections (`num_prev_inspections`), (3) the

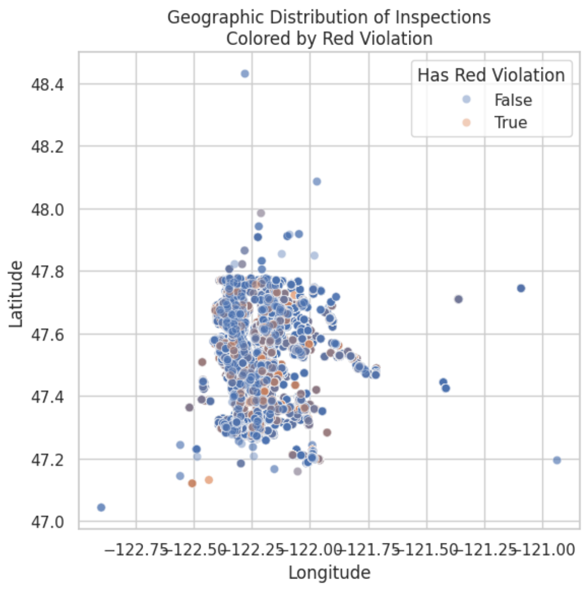

Figure 3. Geographic distribution of restaurant inspections in King County, colored by presence of a red violation.

proportion of inspections with at least one red violation (`prop_red_violations`), (4) the average violation points per inspection (`avg_violation_points`), and (5) the total timespan between the first and most recent inspection (`time_since_last_insp`). These features summarize inspection frequency, typical violation severity, and the historical consistency of critical violations.

We also defined the target variable `Next Red Violation`, which indicates whether the restaurant's next recorded inspection (after its history) contained at least one red violation. This transforms the problem into a binary classification task at the restaurant level, where each row represents the restaurant's entire inspection history up to but not including its next inspection.

After constructing the feature matrix X and target vector y, we handled any remaining missing values using mean imputation. We then performed a 70/30 train–test split to evaluate model performance on unseen restaurants. Finally, features were standardized using a `StandardScaler` for models sensitive to feature magnitude (e.g., logistic regression and k-NN).

3.4. Modeling Approach

To predict whether a restaurant's next inspection will contain at least one red violation, we implemented a set of baseline machine learning classifiers commonly used in binary classification tasks.

We evaluated four supervised learning models:

- **Logistic Regression**: a linear and highly interpretable baseline model, trained on standardized features to account for scale differences.
- **k-Nearest Neighbors (k=5)**: a non-parametric classifier that predicts labels based on the majority class among the 5 most similar restaurants.
- **Decision Tree (depth=5)**: a simple tree classifier capable of capturing nonlinear relationships and threshold effects in inspection history.
- **Random Forest (100 trees)**: an ensemble of decision trees that aggregates predictions across multiple bootstrap samples to improve stability and accuracy.

For models sensitive to feature scale (logistic regression and k-NN), we trained and evaluated using standardized versions of the features. Decision trees and random forests were trained on unscaled data because they are invariant to monotonic transformations.

Each model was trained on 70% of the restaurants and evaluated on the remaining 30%. Performance was measured using accuracy, recall, precision, F1-score, and confusion matrices. Recall is important in this context because missing a potential red violation (a false negative) carries greater public health risk than incorrectly flagging a low risk restaurant.

Across models, we observed consistently strong performance, with accuracy values near 95%–96% and F1-scores above 0.95 for most classifiers. Logistic Regression, k-NN, and Random Forest each achieved high recall (≈0.96), indicating that they successfully identified restaurants that later received critical violations. The Decision Tree model performed slightly worse, likely reflecting its lower capacity compared to the ensemble.

In addition to the supervised models, we also explored unsupervised structure in the data using a 2-cluster k-means algorithm. The resulting clusters aligned loosely with inspection history severity. This shows that restaurants naturally group into lower-risk and higher-risk profiles even without labels. This supports the idea that historical violation patterns contain meaningful predictive signal.

3.5. Model Training and Validation

To ensure that model performance was evaluated fairly across restaurants, we used a group-wise train/test split in which all inspections from the same restaurant were kept together. We applied a 70/30 split using `GroupShuffleSplit`, with restaurant name serving as the grouping variable. This prevents information leakage from multiple inspections of the same establishment appearing in both sets.

For models sensitive to feature magnitude (logistic regression and k-NN), we standardized all features using a `StandardScaler` fitted on the training data. Tree-based models (decision tree and random forest) were trained on unscaled features, since they are invariant to monotonic transformations.

We implemented four supervised learning models and tuned each using a five-fold cross-validated grid search on the training set. Hyperparameter grids included the regularization parameter C for logistic regression, the number of neighbors for k-NN, maximum depth for the decision tree, and both maximum depth and number of estimators for the random forest. Each model was evaluated on the held-out test set using accuracy, precision, recall, F1-score, and ROC-AUC.

Table 1 summarizes our initial results. All models performed strongly, with accuracy near 95% and high recall, indicating that the models were effective at identifying restaurants that later received at least one red violation. The decision tree and random forest models performed particularly well in terms of F1-score and ROC-AUC, suggesting that even shallow tree-based structures capture meaningful nonlinear patterns in inspection history.

Table 1. Comparison of model performance on the test set.

Model	Accuracy	Recall	F1	ROC-AUC
Logistic Regression	0.946	0.963	0.965	0.968
k-NN (best: $k = 9$)	0.948	0.956	0.966	0.969
Decision Tree (depth=3)	**0.954**	0.957	**0.970**	0.971
Random Forest	0.953	**0.958**	0.970	**0.980**

Across models, false negatives (missed red violations) were kept low, as reflected by consistently high recall values (0.955–0.964). This is important in the public health context, where failing to flag a high-risk restaurant poses greater consequences than a false alarm.

3.6. Preliminary Validation

We carried out a preliminary validation step to better understand how the trained models behave on the held-out test set and to check that their predictions are reasonable in the context of public health risk.

First, for each tuned model in our `trained_models` set (logistic regression, k-NN, decision tree, and random forest), we generated confusion matrices and classification reports on the test data. The confusion matrices confirmed the earlier summary metrics: all models correctly identified the vast majority of restaurants whose next inspection contained a red violation, with relatively few false negatives. As expected given the class imbalance, most errors were false positives (flagging a restaurant that did not receive a red violation), which is a safer failure mode in this application.

For models that provide class probabilities, we also recomputed ROC–AUC scores using the predicted probabilities on the test set. These values were consistently high (approximately 0.96–0.98), indicating that the models are able to rank higher- and lower-risk restaurants effectively. The random forest obtained the highest ROC–AUC, reinforcing its strong overall performance.

To interpret the models, we examined feature contributions where possible. For logistic regression, we inspected the learned coefficients; features such as the proportion of past red violations and average violation points had large positive coefficients, indicating that worse historical behavior is strongly associated with a higher probability of a future red violation. For the decision tree and random forest, we plotted feature importances, which again highlighted historical severity (proportion of red violations, average violation points) and temporal coverage (length of inspection history) as key predictors. These patterns are intuitive and suggest that our engineered features are capturing meaningful aspects of restaurant risk.

Overall, the preliminary validation confirms that the models are not only performing well in terms of standard metrics, but are also aligned with domain expectations: restaurants with frequent or severe past violations, and longer or more irregular inspection histories, are more likely to experience critical violations in future inspections.

3.7. Summary

This milestone completes the implementation of our pre-analysis plan and provides an initial evaluation of our restaurant-level risk prediction models. After constructing aggregated inspection features and splitting the data at the restaurant level to avoid leakage, we trained and validated four main classifiers: logistic regression, k-NN, decision trees, and random forests. All models performed strongly, with accuracy and F1 scores around 0.94–0.97 and ROC–AUC values between 0.96 and 0.98. The tuned decision tree and random forest achieved the best overall results, suggesting that nonlinear relationships and interaction effects are important for modeling food safety risk.

Confusion matrices showed that false negatives were relatively rare across all models, meaning high-risk restaurants were rarely missed. Feature importance and coefficient analyses further indicated that historical violation severity (e.g., average past violation points, proportion of red violations) and inspection history length are the strongest predictors of future red violations. These patterns align with public health expectations and validate the relevance of our engineered features.

Overall, our initial results demonstrate that machine learning models can effectively identify restaurants with elevated risk of receiving a future red violation. These findings establish a solid foundation for the final phase of the project, where we will refine models, expand temporal feature engineering, and incorporate additional robustness checks.

COMMITS: Emily Friedman completed code and text for sections 3.1-3.2. Caroline Amsbary completed code and text for sections 3.3-3.5. Suhanee Singh completed code and text for sections 3.6-3.7.