**Vini Report – Version 4 – Data Exploration & Preprocessing**

**Data Exploration**

**1. Loading and Inspecting the Data, Analysis of Datatypes & Univariate Feature Analysis**

We began by loading the dataset and inspecting its structure using functions like df.info() and df.describe(). This provided an overview of column types, missing values, and statistical metrics such as mean, median, and standard deviation. For example, *Average Weekly Wage* displayed significant skewness, and categorical features like *District Name* showed dominant categories such as “NYC.” [Image to illustrate what was said]

During this analysis, we identified discrepancies in datatypes that required adjustments. Columns like *Zip Code* and *District Name*, although appearing numeric, were treated as categorical variables, while *Industry Code*, stored as a float, was better represented as an integer to reflect its discrete nature. These corrections ensured compatibility with preprocessing and machine learning algorithms. [Image to illustrate what was said]

Additionally, univariate analysis helped further evaluate individual features. For numerical variables, histograms and density plots revealed skewness and outliers. For example, *Average Weekly Wage* required scaling due to its wide range, while categorical features, such as *District Name* and *County of Injury*, were visualized with bar charts to understand distribution patterns. Redundant features like *OIICS Nature of Injury Description* were removed, and other variables were flagged for scaling and encoding during preprocessing. This analysis provided key insights into the dataset’s structure and guided subsequent steps.

[Image to illustrate what was said]

**2. Comparing Train and Test Data & Examining Missing Values (NaN)**

The train and test datasets were compared to ensure consistency and identify patterns of missing values. The training dataset had a higher proportion of missing values (10%) compared to the test set (4.5%), with one-third of incomplete training records missing geographical information. In contrast, the test set always retained at least one geographical feature per record. Features such as *District Name* and *County of Injury* exhibited consistent distributions across both datasets, ensuring that transformations applied to the training data could generalize effectively. [Image to illustrate what was said]

A deeper examination of missing values revealed that *Zip Code* had the highest proportion of missing data (8.1% in training, 4.9% in testing). Additionally, a dependency was observed between *Industry Code* and *Industry Code Description*, where missing values in one column often coincided with the other. These findings informed imputation strategies and guided decisions to drop rows with excessive missing values while preserving useful data for analysis. [Image to illustrate what was said]

**3. Outlier Analysis**

Outliers were identified using the IQR method, where bounds were calculated as 1.5 times the interquartile range below the 1st quartile and above the 3rd quartile. For instance, *Age at Injury* included erroneous values of 0, which were corrected. Extreme but valid values, such as high wages, were retained to maintain real-world variability. This approach ensured the dataset remained robust and informative without compromising its integrity. [Image to illustrate what was said]

**Preprocessing**

**1. Changing the Datatypes**

Correcting datatypes was a critical preprocessing step to ensure accurate feature interpretation:

• **Datetime Conversion**: Columns like *Accident Date*, *C-2 Date*, and *First Hearing Date* were converted to datetime format, enabling easy computation of intervals or extraction of time-based features.

• **Integer Conversion**: Features such as *Age at Injury* and *Industry Code* were converted to Int64 to standardize numerical operations while handling missing values gracefully.

• **Categorical Conversion**: Variables like *District Name*, *Claim Injury Type*, and *County of Injury* were categorized to reflect their non-numeric nature, reduce memory usage, and facilitate efficient encoding.

[Image to illustrate what was said]

**2. Handling Missing Values**

Rows with excessive missing data (more than 29 NaNs) were dropped from the training set. Other missing values were addressed using imputation strategies:

• *Industry Code*: Missing values were replaced with 0.

• Temporal Features (*Accident Date Year*, *Accident Date Month*): Median values from the training set were used.

Additionally, KNN imputation was applied to numerical features, leveraging patterns from other observations to fill gaps effectively. [Image to illustrate what was said]

**3. Scaling of Numerical Features**

Numerical features were scaled using MinMaxScaler to bring all variables into a comparable range, preventing larger features from dominating optimization processes. This scaling was particularly beneficial for distance-based and gradient-based algorithms. [Image to illustrate what was said]

**4. Handling Class Imbalance**

Class imbalance in the target variable was addressed using SMOTE (Synthetic Minority Over-sampling Technique) to generate synthetic samples for underrepresented classes. Undersampling was applied for overrepresented classes. This ensured that the dataset was balanced, reducing prediction bias and improving fairness. [Image to illustrate what was said]

**5. Treating Outliers**

Erroneous outliers, such as *Age at Injury* = 0, were corrected, while valid extreme values were retained to preserve variability. This balance maintained the dataset’s realism and helped avoid overfitting to a limited range. [Image to illustrate what was said]