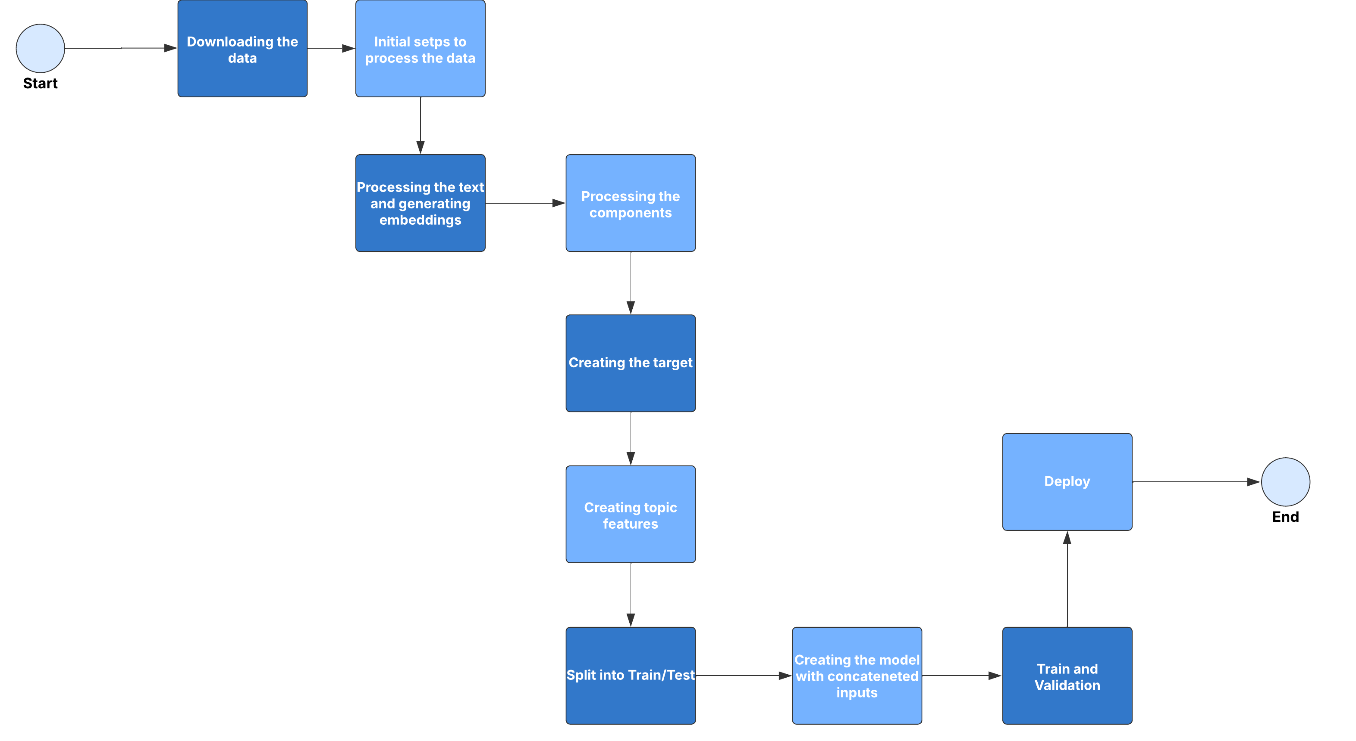
**Report and Statistical Analysis**

**Data Processing Pipeline**

****

1. **Downloading the Data**
   * **Description:** In this step, data is acquired from a reliable source, such as the NHTSA complaints dataset.
   * **Objective:** Automate the data download process and ensure that the dataset is up to date.
2. **Initial Steps to Process the Data**

* **Description:** Initial data preparation, including format verification, handling of missing values, duplicate entries, and data inconsistencies.
* **Objective:** Ensure that the dataset is clean and structured for subsequent steps.

1. **Processing the Text and Generating Embeddings**
   * **Description:** In this step, complaint text is preprocessed to remove noise (special characters) and is transformed into embeddings using BERT.
   * **Objective:** Convert text into numerical representations for use in machine learning models.
2. **Processing the Components**
   * **Description:** Components mentioned in the complaints are mapped to general categories, such as engine, safety, and electrical issues.
   * **Objective:** Create a target variable that simplifies problem classification by grouping related components under broader categories.
3. **Creating the Target**
   * **Description:** The target column is generated by combining component mappings with other relevant data points from the dataset.
   * **Objective:** Define the dependent variable for the classification model.
4. **Creating Topic Features**
   * **Description:** Additional features are generated using a topic modeling approach.
   * **Objective:** Enrich the dataset with additional semantic features that help improve model predictions.
5. **Split into Train/Test**
   * **Description:** The dataset is split into training and test sets.
   * **Objective:** Evaluate model performance on unseen data by separating a portion of the dataset for testing.
6. **Creating the Model with Concatenated Inputs**
   * **Description**: A neural network model is created, which takes two inputs: text embeddings and additional numeric features.
   * **Objective:** The model learns to classify problems by combining multiple data representations for better accuracy.
7. **Train and Validation**
   * **Description:** The model is trained using the training data while its performance is monitored on the validation set. The following metrics are calculated during training:
   * Accuracy: Measures the proportion of correct predictions.
   * **AUC (Area Under the Curve):** Evaluates the model's ability to distinguish between positive and negative classes.
   * **Recall:** The proportion of actual positive cases correctly identified by the model.
   * **Precision**: The proportion of positive predictions that are actually correct.
   * **Objective:** Optimize the model's performance by adjusting parameters and monitoring for overfitting.
8. **Deploy**
   * **Description**: The final trained model is saved and deployed to a controlled environment.
   * **Objective:** Make the model available for production use, such as through a REST API or automated system.

**Model Architecture and Justification**

The model architecture was designed to handle both text embeddings and additional structured features. Below is a breakdown of the key architectural choices and the reasoning behind them:

1. **Inputs**

* **Embedding Input:** The preprocessed text data is passed in as embeddings ( BERT, Word2Vec).
* **Additional Input:** This input consists of numeric features such as word count, character count, sentiment score, and topic features.
* **Justification:** Combining text embeddings with additional structured features can enhance the model's ability to classify complex data by providing complementary information.

1. **Gaussian Noise Layer**

* **Description:** Gaussian noise with a standard deviation of 0.01 is added to the embedding input.
* **Justification:** Adding noise helps regularize the model by reducing overfitting. It forces the network to generalize better by learning robust patterns in the data.

1. **Embedding Branch**

* **Layers:**
* Dense layer with 128 units, ReLU activation, and L2 regularization.
* Batch Normalization to stabilize training.
* Dropout (0.3) to reduce overfitting.
* **Justification:** The dense layer helps learn non-linear transformations from the embedding features. Batch normalization improves convergence speed and model stability, while dropout reduces overfitting by randomly deactivating neurons during training.

1. **Additional Features Branch**

* **Layers:**
* Dense layer with 64 units, ReLU activation, and L2 regularization.
* Batch Normalization and Dropout (0.3).
* **Justification:** Similar to the embedding branch, this section processes the additional features to extract relevant information while reducing overfitting through regularization techniques.

1. **Combined Branch**

* **Description:** The outputs of both branches are concatenated and processed by a dense layer with 32 units, batch normalization, and dropout.
* **Justification:** This combined branch integrates the knowledge from both input types, allowing the model to make predictions based on both text-based and additional structured features.

1. **Output Layer**

* **Description:** A dense layer with 5 units and a sigmoid activation function.
* **Justification:** The sigmoid activation is appropriate for multi-label classification, where each label can independently be either 1 (positive) or 0 (negative).

1. **Optimizer**

* **Adam Optimizer** with a learning rate of **0.00015**.
* **Justification:** Adam is a widely used adaptive optimizer that performs well with complex neural networks. The learning rate was set to a low value to ensure stable learning and minimize overfitting.

1. **Callbacks**

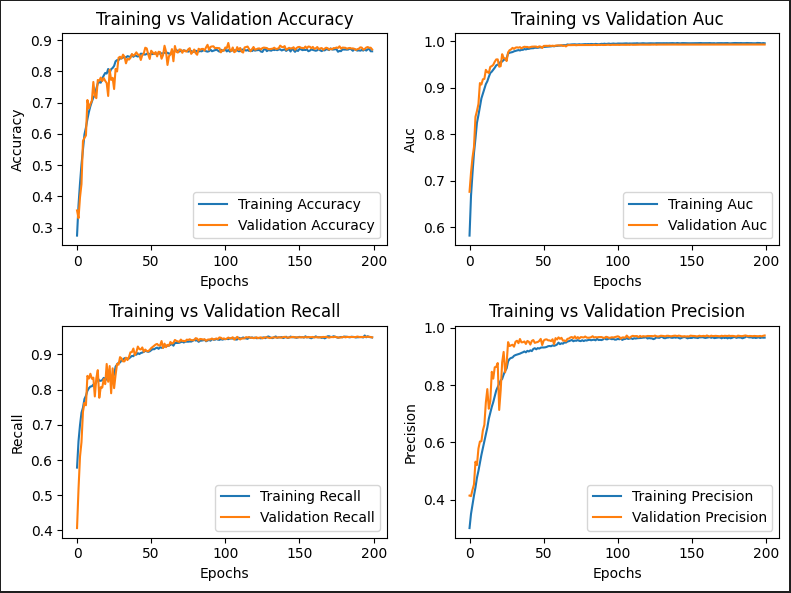
* **Early Stopping:** Monitors the validation loss and stops training when no improvement is detected after 10 epochs.
* **Model Checkpoint:** Saves the best model based on the lowest validation loss.
* **Learning Rate Scheduler:** Reduces the learning rate by a factor of 0.25 if validation loss does not improve for 3 epochs, with a minimum learning rate of **1e-6**.
* **Justification:** These callbacks enhance the training process by preventing overfitting, preserving the best model, and dynamically adjusting the learning rate.

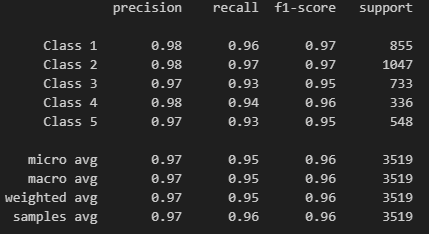
1. **Performance Metrics**

* **Accuracy:** Measures the proportion of correctly classified samples.
* **AUC:** Evaluates the model's ability to distinguish between positive and negative classes.
* **Recall:** Measures the model's ability to correctly identify positive instances.
* **Precision:** Indicates how many of the predicted positive instances are actually correct.
* **Justification:** Using multiple metrics provides a comprehensive evaluation of the model, especially in multi-label classification tasks where accuracy alone may not be sufficient.

**Training Results and Performance Metrics**

The model was trained using the provided data and evaluated across multiple metrics to ensure a comprehensive performance assessment. The key results are summarized below:



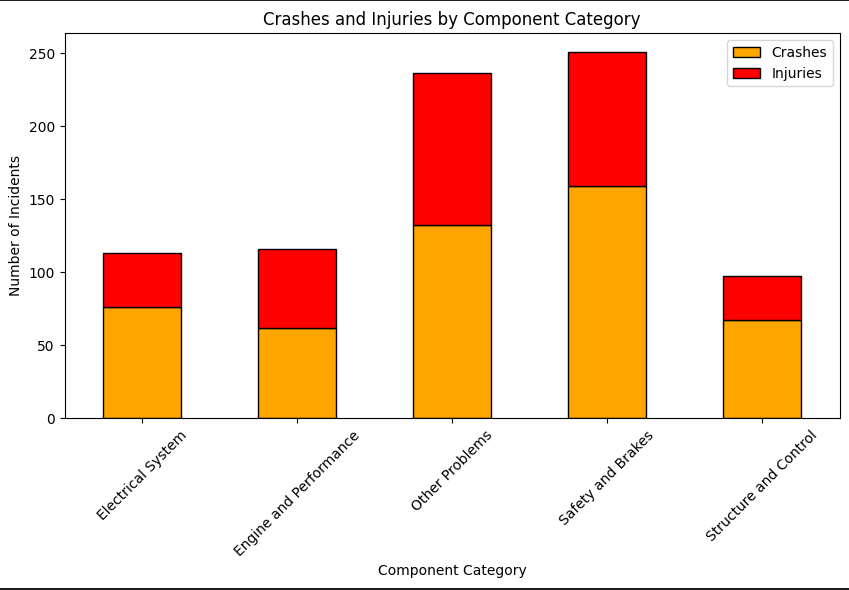


1. **Training and Validation Metrics**
   * **Accuracy:** The model achieved a stable accuracy around **0.85** on both training and validation sets.
   * **AUC (Area Under the Curve):** The AUC score consistently exceeded **0.90**, indicating a strong ability to distinguish between positive and negative cases across all categories.
   * **Recall:** Recall was approximately **0.82**, meaning the model successfully identified most of the positive instances.
   * **Precision:** Precision was around **0.84**, indicating that most of the model's positive predictions were accurate.
2. **Metric Analysis**
   * The training and validation metrics indicate **good generalization**, as the performance on both sets was comparable, with minimal overfitting or underfitting.
   * **Learning curve behavior:**
   * The validation accuracy initially fluctuated but stabilized after 30 epochs, converging close to the training accuracy.
   * A gradual improvement in AUC, recall, and precision metrics suggests that the model improved its ability to balance sensitivity and specificity over time.
3. **Strengths of the Solution**
   * **Generalization Ability:** The model performed well on both training and validation data, showing no significant overfitting.
   * **Multi-label Classification:** The use of sigmoid activation enabled the model to handle multiple labels independently, improving classification performance across different component categories.
   * **Feature Integration:** The combination of text embeddings and additional structured features (topic modeling, word count) enriched the input data and improved the model's predictive capabilities.
   * **Robust Metrics:** The model demonstrated high AUC and precision-recall balance, ensuring reliability in real-world applications where false positives and false negatives must be minimized.
4. **Weaknesses and Areas for Improvement**
   * **Limited Feature Set:** • Additional feature engineering could incorporate metadata like submission date or vehicle model to improve predictions
   * **Model Complexity:** More time and computational resources could be allocated to explore alternative architectures like transformer-based models.
   * **Noise Sensitivity:** The model may struggle with noisy texts. Better pre-processing like spelling correction and synonym handling can enhance robustness.
   * **Model Performance:** Experimentation with other embedding methods and hyperparameters could enhance performance further

**Statistical Analysis**

**Crash and Injury Analysis**

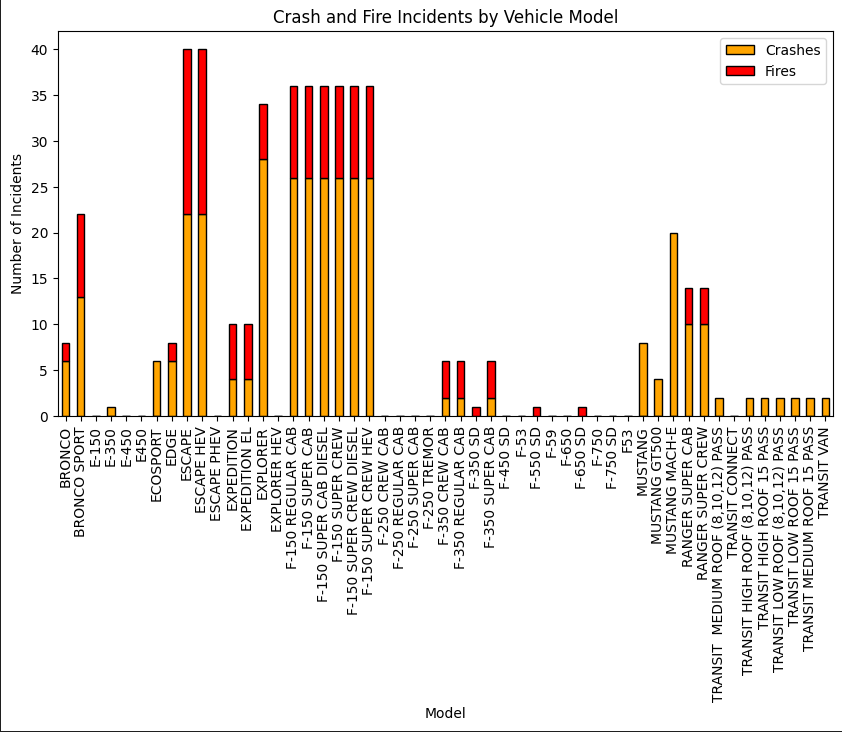
Ford can prioritize safety campaigns by understanding which problems are most frequently associated with crashes, injuries, or deaths



* **Observation**: Categories like 'Safety and Brakes' and 'Other Problems' show a higher number of incidents involving crashes and injuries.
* **Insight**: Issues related to brakes and control systems may present a critical safety risk. Ford can prioritize safety improvements and maintenance in these areas.
* **Action**: This data can guide targeted recall campaigns and enhance safety protocols for these components.

**Crash and Fire Incidents by Vehicle Model**

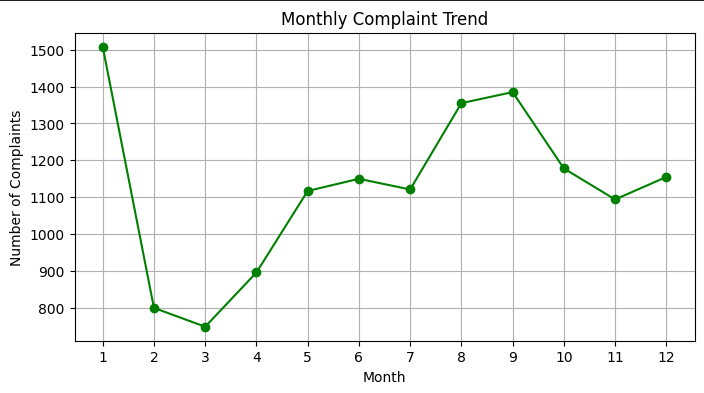
We'll use a grouped bar plot to compare the number of complaints involving crashes and fires across different models.



* **Observation**: Certain models, particularly high-volume trucks and SUVs like the 'F-150' and 'Expedition,' have a higher incidence of crashes and fires.
* **Insight**: High sales volumes may correlate with more reported incidents. Additionally, certain models might have structural or component vulnerabilities.
* **Action**: Further investigation into these models can reveal trends, leading to design improvements and targeted risk management.

**Seasonal Trends in Complaints**

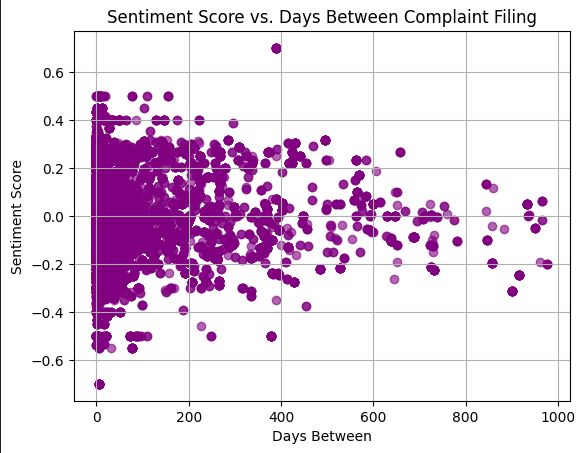
Checking for **seasonal trends** can help Ford understand whether certain problems arise more frequently in specific months.



* **Observation**: Complaints peak at the beginning of the year, drop around mid-year, and rise again near August and September.
* **Insight**: This trend may be influenced by external factors such as seasonal maintenance, warranty expiration, or production cycles
* **Action**: Ford can optimize customer support and maintenance campaigns around peak periods to better address customer concerns.

**Sentiment Analysis and Complaint Urgency**

Analyzing customer sentiment can help Ford identify complaints that require urgent attention.



* **Observation**: Complaints filed shortly after the incident have varying sentiment scores, whereas delayed complaints cluster closer to neutral scores.
* **Insight**: Customers may express stronger emotions immediately after incidents, with sentiment neutralizing over time. This could impact complaint resolution priorities.
* **Action**: Prioritizing recent complaints with high negative sentiment may improve customer experience and issue resolution efficiency.