For the Short Data we are testing the for creating the Dictionary for the overall data which is present in both the file. this will create the categories by them self which is present in the dataset in dictionary format which is nothing but the JSON file where data is storing in key value format

In [ ]:
```python
import pandas as pd
import re
from fuzzywuzzy import process
from sentence_transformers import SentenceTransformer
import numpy as np

# Load data
df1 = pd.read_csv('surveydatashort.csv')
df2 = pd.read_csv('garageshortdata.csv')

def preprocess(text):
    if isinstance(text, str):  # Check if the input is a string
        text = text.lower().replace("|", " ").replace("-", " ")
        text = re.sub(r'\b(rh|r)\b', 'right', text)
        text = re.sub(r'\b(lh|l)\b', 'left', text)
        return text
    return ""  # Return an empty string if text is not a string

df1['clean_key'] = df1['TXT_PARTS_NAME'].apply(preprocess)
df2['clean_value'] = df2['PARTDESCRIPTION'].apply(preprocess)

# Fuzzy Matching to get candidates
key_list = df1['clean_key'].tolist()
value_list = df2['clean_value'].tolist()

matches_dict = {}
for key in key_list:
    candidates = process.extract(key, value_list, limit=5)
    matches_dict[key] = [cand[0] for cand in candidates if cand[1] > 80]

# Semantic Similarity to refine matches
model = SentenceTransformer('paraphrase-MiniLM-L6-v2')
key_embeddings = model.encode(key_list)
value_embeddings = model.encode(value_list)

from sklearn.metrics.pairwise import cosine_similarity
similarity = cosine_similarity(key_embeddings, value_embeddings)

final_mapping = {}
for idx, key in enumerate(key_list):
    best_match_idx = np.argmax(similarity[idx])
    final_mapping[key] = value_list[best_match_idx]

# Save results
pd.DataFrame(final_mapping.items(), columns=['TXT_PARTS_NAME', 'PARTDESCRIPTION']).to_csv('sh
```

In [2]:
```python
import pandas as pd
import re
import json
from fuzzywuzzy import process
from sentence_transformers import SentenceTransformer
import numpy as np

# Load data
df1 = pd.read_csv('surveydatashort.csv')
df2 = pd.read_csv('garageshortdata.csv')

def preprocess(text):
    if isinstance(text, str):
```

```
        text = text.lower().replace("|", " ").replace("-", " ")
        text = re.sub(r'\b(rh|r)\b', 'right', text)
        text = re.sub(r'\b(lh|l)\b', 'left', text)
        text = re.sub(r'[^a-z0-9\s]', '', text)
        return ' '.join(text.split()).strip()
    return ""

# Preprocess both datasets
df1['clean_key'] = df1['TXT_PARTS_NAME'].apply(preprocess)
df2['clean_value'] = df2['PARTDESCRIPTION'].apply(preprocess)

# Create mapping dictionaries
original_key_mapping = dict(zip(df1['clean_key'], df1['TXT_PARTS_NAME']))
original_value_mapping = dict(zip(df2['clean_value'], df2['PARTDESCRIPTION']))

# Get unique cleaned values
key_list = df1['clean_key'].unique().tolist()
value_list = df2['clean_value'].unique().tolist()

# Semantic similarity setup
model = SentenceTransformer('paraphrase-MiniLM-L6-v2')
key_embeddings = model.encode(key_list)
value_embeddings = model.encode(value_list)
similarity = cosine_similarity(key_embeddings, value_embeddings)

# Build the final mapping dictionary
result_dict = {}
SIMILARITY_THRESHOLD = 0.65  # Adjust this based on your validation

for key_idx, clean_key in enumerate(key_list):
    matches = []
    for value_idx, score in enumerate(similarity[key_idx]):
        if score > SIMILARITY_THRESHOLD:
            original_value = original_value_mapping[value_list[value_idx]]
            matches.append((original_value, score))

    # Sort matches by similarity score descending
    matches = sorted(matches, key=lambda x: x[1], reverse=True)

    # Get original key and store matches
    original_key = original_key_mapping[clean_key]
    result_dict[original_key] = [match[0] for match in matches]

# Add fuzzy matches for low-confidence items
FUZZY_THRESHOLD = 75
for original_key, clean_key in zip(df1['TXT_PARTS_NAME'], df1['clean_key']):
    if not result_dict.get(original_key):
        fuzzy_matches = process.extract(clean_key, value_list, limit=3)
        result_dict[original_key] = [
            original_value_mapping[match[0]]
            for match in fuzzy_matches
            if match[1] >= FUZZY_THRESHOLD
        ]

# Save to JSON
with open('parts_mapping.json', 'w') as f:
    json.dump(result_dict, f, indent=2)

print("JSON mapping created successfully!")
```

```
JSON mapping created successfully!
```

Now we are testing for the Big data for creating the dictionary.this will create the categories by them self which is present in the dataset in dictionary format which is nothing but the JSON file where data is storing in key value format

In [3]:

```python
import pandas as pd
import re
import json
from fuzzywuzzy import process
from sentence_transformers import SentenceTransformer
import numpy as np

# Load data
df1 = pd.read_csv('../surveyor_data_cleanedd.csv')
df2 = pd.read_csv('../garagedata_cleaned_data.csv')

def preprocess(text):
    if isinstance(text, str):
        text = text.lower().replace("|", " ").replace("-", " ")
        text = re.sub(r'\b(rh|r)\b', 'right', text)
        text = re.sub(r'\b(lh|l)\b', 'left', text)
        text = re.sub(r'[^a-z0-9\s]', '', text)
        return ' '.join(text.split()).strip()
    return ""

# Preprocess both datasets
df1['clean_key'] = df1['TXT_PARTS_NAME'].apply(preprocess)
df2['clean_value'] = df2['PARTDESCRIPTION'].apply(preprocess)

# Create mapping dictionaries
original_key_mapping = dict(zip(df1['clean_key'], df1['TXT_PARTS_NAME']))
original_value_mapping = dict(zip(df2['clean_value'], df2['PARTDESCRIPTION']))

# Get unique cleaned values
key_list = df1['clean_key'].unique().tolist()
value_list = df2['clean_value'].unique().tolist()

# Semantic similarity setup
model = SentenceTransformer('paraphrase-MiniLM-L6-v2')
key_embeddings = model.encode(key_list)
value_embeddings = model.encode(value_list)
similarity = cosine_similarity(key_embeddings, value_embeddings)

# Build the final mapping dictionary
result_dict = {}
SIMILARITY_THRESHOLD = 0.65  # Adjust this based on your validation

for key_idx, clean_key in enumerate(key_list):
    matches = []
    for value_idx, score in enumerate(similarity[key_idx]):
        if score > SIMILARITY_THRESHOLD:
            original_value = original_value_mapping[value_list[value_idx]]
            matches.append((original_value, score))

    # Sort matches by similarity score descending
    matches = sorted(matches, key=lambda x: x[1], reverse=True)

    # Get original key and store matches
    original_key = original_key_mapping[clean_key]
    result_dict[original_key] = [match[0] for match in matches]

# Add fuzzy matches for low-confidence items
FUZZY_THRESHOLD = 75
for original_key, clean_key in zip(df1['TXT_PARTS_NAME'], df1['clean_key']):
    if not result_dict.get(original_key):
        fuzzy_matches = process.extract(clean_key, value_list, limit=3)
        result_dict[original_key] = [
            original_value_mapping[match[0]]
            for match in fuzzy_matches
            if match[1] >= FUZZY_THRESHOLD
```

```
        ]

# Save to JSON
with open('Big_parts_mapping.json', 'w') as f:
    json.dump(result_dict, f, indent=2)

print("JSON mapping created successfully!")
```

```
WARNING:root:Applied processor reduces input query to empty string, all comparisons will have
score 0. [Query: '']
WARNING:root:Applied processor reduces input query to empty string, all comparisons will have
score 0. [Query: '']
WARNING:root:Applied processor reduces input query to empty string, all comparisons will have
score 0. [Query: '']
JSON mapping created successfully!
```

As we where unsuccessful above because the data is to much unorganise and its label data but its different all cross the different dataset no consistent dataset we got something around 50k categories from the 90k and 30k row dataset its doesn't making any sense

As the previous approach was unsuccessful, we decided to attempt a different approach.

1. Input Dataset:
    - We were provided with a primary dataset (3rd dataset) containing around 30 rows and one column.
    - This column contains 2-3 categories in one cell, with categories separated by custom delimiters (such as `||` or space `' '`).
2. New Approach:
    - Treat the categories in the specified column as the main categories.
    - Match these categories with rows from two large datasets:
        - Dataset 1: 30k rows.
        - Dataset 2: 90k rows.
3. Planned Outcome:
    - Create a dictionary:
        - Matched Data: Categories from small dataset matched across larger datasets.
        - Unmatched Data: For rows where categories could not be matched, create a different category or secondary dataset.

Initially, we tested the above approach using a smaller dataset (Shortdata).

In [15]:
```
import pandas as pd
import re
import json
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity

def preprocess(text):
    if isinstance(text, str):
        text = text.lower()
        text = re.sub(r'[^a-z0-9\s]', '', text)
        text = re.sub(r'\b(assembly|comp|assy|unit)\b', '', text)
        text = re.sub(r'\s+', ' ', text).strip()
        return text
    return ""

# Load data
surveyor_df = pd.read_csv('Primary_Parts_Code.csv')
```

```
garage_df = pd.read_csv('garageshortdata.csv')

# Preprocess and clean data
surveyor_df['clean_name'] = surveyor_df['Surveyor Part Name'].apply(preprocess)
garage_df['clean_desc'] = garage_df['PARTDESCRIPTION'].apply(preprocess)

# Remove duplicate garage descriptions while keeping first occurrence
garage_df = garage_df.drop_duplicates(subset=['clean_desc'], keep='first')

# Encode texts
model = SentenceTransformer('paraphrase-MiniLM-L6-v2')
category_embeddings = model.encode(surveyor_df['clean_name'].tolist())
desc_embeddings = model.encode(garage_df['clean_desc'].tolist())

# Calculate similarity
similarity_matrix = cosine_similarity(category_embeddings, desc_embeddings)

# Create optimized mapping with duplicate removal
category_mapping = {}
SIMILARITY_THRESHOLD = 0.65

for idx, row in surveyor_df.iterrows():
    seen = set()
    unique_matches = []

    # Get sorted matches with scores
    matches = sorted(
        [(garage_df.iloc[desc_idx]['PARTDESCRIPTION'], score)
         for desc_idx, score in enumerate(similarity_matrix[idx])
         if score > SIMILARITY_THRESHOLD],
        key=lambda x: x[1],
        reverse=True
    )

    # Deduplicate while preserving order
    for desc, _ in matches:
        if desc not in seen:
            seen.add(desc)
            unique_matches.append(desc)

    # Create category key
    category_key = f"{row['Surveyor Part Name']}"
    category_mapping[category_key] = unique_matches

# Add empty arrays for unmatched categories
for _, row in surveyor_df.iterrows():
    category_key = f"{row['Surveyor Part Name']}"
    if category_key not in category_mapping:
        category_mapping[category_key] = []

# Save to JSON
with open('deduplicated_mapping.json', 'w') as f:
    json.dump(category_mapping, f, indent=2)

print("Clean JSON mapping with unique values created!")
```

```
Clean JSON mapping with unique values created!
```

Following the earlier approach, which yielded great results, we are now experimenting with a larger dataset.

In [19]:

```
import pandas as pd
import re
import json
from sentence_transformers import SentenceTransformer
```

```python
from sklearn.metrics.pairwise import cosine_similarity

def preprocess(text):
    if isinstance(text, str):
        text = text.lower()
        text = re.sub(r'[^a-z0-9\s]', '', text)
        text = re.sub(r'\b(assembly|comp|assy|unit)\b', '', text)
        text = re.sub(r'\s+', ' ', text).strip()
        return text
    return ""

# Load data
surveyor_df = pd.read_csv('Primary_Parts_Code.csv')
garage_df = pd.read_csv('../garagedata_cleaned_data.csv')

# Preprocess and clean data
surveyor_df['clean_name'] = surveyor_df['Surveyor Part Name'].apply(preprocess)
garage_df['clean_desc'] = garage_df['PARTDESCRIPTION'].apply(preprocess)

# Remove duplicate garage descriptions while keeping first occurrence
garage_df = garage_df.drop_duplicates(subset=['clean_desc'], keep='first')

# Encode texts
model = SentenceTransformer('paraphrase-MiniLM-L6-v2')
category_embeddings = model.encode(surveyor_df['clean_name'].tolist())
desc_embeddings = model.encode(garage_df['clean_desc'].tolist())

# Calculate similarity
similarity_matrix = cosine_similarity(category_embeddings, desc_embeddings)

# Create optimized mapping with duplicate removal
category_mapping = {}
SIMILARITY_THRESHOLD = 0.65

for idx, row in surveyor_df.iterrows():
    seen = set()
    unique_matches = []

    # Get sorted matches with scores
    matches = sorted(
        [(garage_df.iloc[desc_idx]['PARTDESCRIPTION'], score)
         for desc_idx, score in enumerate(similarity_matrix[idx])
         if score > SIMILARITY_THRESHOLD],
        key=lambda x: x[1],
        reverse=True
    )

    # Deduplicate while preserving order
    for desc, _ in matches:
        if desc not in seen:
            seen.add(desc)
            unique_matches.append(desc)

    # Create category key
    category_key = f"{row['Surveyor Part Name']}"
    category_mapping[category_key] = unique_matches

# Add empty arrays for unmatched categories
for _, row in surveyor_df.iterrows():
    category_key = f"{row['Surveyor Part Name']}"
    if category_key not in category_mapping:
        category_mapping[category_key] = []

# Save to JSON
with open('Bigdeduplicated_mapping.json', 'w') as f:
    json.dump(category_mapping, f, indent=2)
```

```
print("Clean JSON mapping with unique values created!")
```

```
Clean JSON mapping with unique values created!
```

The two datasets currently have no clear similarities to define a meaningful relationship between them,

except for the "Parts Name". However, using "Parts Name" for creating a relationship does not provide much utility.

Upon further observation, we discovered another similarity between the two datasets: the "Vehicle Model Number".

To address this, we created a new dataset that associates both datasets by adding the "Vehicle Model Number".

This approach allows us to establish a relationship between them, which is crucial for generating invoices

specific to the datasets.

Finally, we compared the data based on the "Vehicle Model Number" from the garage dataset.

In [101…

```
import pandas as pd

input_csv = '../garagedata_cleaned_data.csv'
output_csv = 'filtered_garage_by_Vehicle_Number.csv'
target_codes = ['23103', '30135']  # Must be strings

# 1. Use COMMA delimiter and correct column names
cols_to_load = ['VEHICLE_MODEL_CODE', 'PARTDESCRIPTION','TOTAL_AMOUNT']

# 2. Optimized processing with correct separator
chunk_size = 50_000
first_chunk = True

for chunk in pd.read_csv(
    input_csv,
    sep=',',  # Changed to comma delimiter
    usecols=cols_to_load,
    dtype={'VEHICLE_MODEL_CODE': 'str'},
    chunksize=chunk_size
):
    filtered = chunk[chunk['VEHICLE_MODEL_CODE'].isin(target_codes)]

    filtered.to_csv(
        output_csv,
        mode='a' if not first_chunk else 'w',
        header=first_chunk,
        index=False
    )
    first_chunk = False

print("Filtering complete!")
```

```
Filtering complete!
```

Standardize the garage dataset by mapping inconsistent part names to their standardized versions. We previously defined a `part_mapping` dictionary that contains key-value pairs for this purpose: the keys represent standardized names, and the values represent part names as they appear in the dataset.

This process involves:

1. Identifying part names in the dataset that match the values in the `part_mapping` dictionary.
2. Replacing these names in the dataset with their corresponding keys from the dictionary.

In simpler terms, the dataset and the mapping dictionary are aligned such that part names in the dataset are replaced with their standardized equivalents according to the dictionary.

In [44]:
```python
import pandas as pd
import json

# Load mappings
with open('Bigdeduplicated_mapping.json') as f:
    mapping_data = json.load(f)

# Create reverse lookup
reverse_map = {}
for key, aliases in mapping_data.items():
    for alias in aliases:
        reverse_map[alias.strip().lower()] = key

# Process CSV with CORRECT DELIMITER
input_csv = 'filtered_garage_by_Vehicle_Number.csv'
output_csv = 'GarageModel_Invoice_Data(Replaced).csv'
chunk_size = 10000

# 1. Split columns properly using comma delimiter
first_chunk = True

for chunk in pd.read_csv(
    input_csv,
    sep=',',  # Changed to comma delimiter
    header=0,
    names=['VEHICLE_MODEL_CODE', 'PARTDESCRIPTION', 'TOTAL_AMOUNT'],  # Explicit column names
    chunksize=chunk_size
):
    # Normalize part descriptions
    chunk['clean_part'] = chunk['PARTDESCRIPTION'].str.strip().str.lower()

    # Map to new descriptions
    chunk['PARTDESCRIPTION'] = chunk['clean_part'].map(reverse_map).fillna(chunk['PARTDESCRIP

    # Save results
    chunk[['VEHICLE_MODEL_CODE', 'PARTDESCRIPTION', 'TOTAL_AMOUNT']] \
        .to_csv(
            output_csv,
            mode='a',
            header=first_chunk,
            index=False
        )
    first_chunk = False

print("Processing complete!")
```

```
Processing complete!
```

Above we did for the garage data set now below we doing for servey dataset the same thing Now Working with Servey Data set

In [57]:
```python
import pandas as pd

input_csv = '../surveyor_data_cleanedd.csv'
output_csv = 'filtered_garage_by_Vehicle_Number_of_Servey.csv'
target_codes = ['23103', '30135']  # Keep as strings

# 1. Add data validation and type conversion
```

```
chunk_size = 50_000
first_chunk = True
found_count = 0

for chunk in pd.read_csv(
    input_csv,
    sep=',',
    usecols=['VEHICLE_MODEL_CODE', 'TXT_PARTS_NAME', 'TOTAL_AMOUNT'],
    chunksize=chunk_size,
    dtype={
        'VEHICLE_MODEL_CODE': 'string',  # Force string type
        'TXT_PARTS_NAME': 'string',
        'TOTAL_AMOUNT': 'float64'
    }
):
    # Convert to string and clean vehicle codes
    chunk['VEHICLE_MODEL_CODE'] = chunk['VEHICLE_MODEL_CODE'].str.split('.').str[0]

    # Filter using cleaned codes
    filtered = chunk[chunk['VEHICLE_MODEL_CODE'].isin(target_codes)]

    # Debug: Show found matches
    if not filtered.empty:
        print(f"Found {len(filtered)} matches in chunk:")
        print(filtered[['VEHICLE_MODEL_CODE', 'TXT_PARTS_NAME']].head())
        found_count += len(filtered)

    # Write to CSV if any matches
    if not filtered.empty:
        filtered.to_csv(
            output_csv,
            mode='a',
            header=first_chunk,
            index=False
        )
        first_chunk = False

print(f"\nTotal matches found: {found_count}")
print("Filtering complete!" if found_count > 0 else "No matching records found!")
```

```
Found 831 matches in chunk:
    VEHICLE_MODEL_CODE           TXT_PARTS_NAME
70                23103  tail light housing left
71                23103                bracket 2
85                30135  horn assembly low pitch
86                30135        radiator assembly
149               23103     bumper rear assembly
Found 824 matches in chunk:
      VEHICLE_MODEL_CODE                      TXT_PARTS_NAME
50026               23103                       tank assembly
50027               23103            member comp front bumper
50173               23103  panel comp wheel house inner right
50248               23103                     garnish rdtr upr
50329               30135             fender panel front left
Found 801 matches in chunk:
       VEHICLE_MODEL_CODE              TXT_PARTS_NAME
100041              30135        runfront door glassl
100245              23103    member comp front bumper
100253              23103   stabilizer bar joint 2nos
100333              23103  bracket front bumper right
100361              30135  reinf cowl upper left hand
Found 843 matches in chunk:
       VEHICLE_MODEL_CODE                  TXT_PARTS_NAME
150361              23103                     holder bump
150427              23103       moulding front windshield
150504              23103               radiator assembly
150505              23103  lining comp front fender right
150506              23103           grille radiator lower
Found 716 matches in chunk:
       VEHICLE_MODEL_CODE      TXT_PARTS_NAME
200282              30135   radiator assembly
200367              23103         emblem mark
200449              30135   radiator assembly
200450              30135  bumper front lower
200451              30135  bumper front upper
Found 743 matches in chunk:
       VEHICLE_MODEL_CODE                   TXT_PARTS_NAME
250010              23103            bumper front assembly
250011              23103        holder front right side l
250101              23103  holder front right bumper right
250477              23103      emblem logo rear left top 1
250599              23103      tow hook cover bumper front
Found 772 matches in chunk:
       VEHICLE_MODEL_CODE                    TXT_PARTS_NAME
300050              30135                hose radiator inlet
300192              23103                          tape door
300223              23103  sealant front windshield glass 1
300409              23103           fender panel front left
300441              23103           fender panel front left
Found 813 matches in chunk:
       VEHICLE_MODEL_CODE                          TXT_PARTS_NAME
350096              23103                          tape rear door
350171              23103  strut assembly front suspension right
350401              30135                   grille radiator upper
350571              23103                                  holder
350702              23103                   165 80r14 duraplus gy
Found 863 matches in chunk:
       VEHICLE_MODEL_CODE            TXT_PARTS_NAME
400112              30135   hinge bonnet hood left
400113              30135  garnish rr bumper lower
400244              23103    cover fog light right
400245              23103         head light right
400255              23103    bumper front assembly
Found 750 matches in chunk:
       VEHICLE_MODEL_CODE      TXT_PARTS_NAME
450020              23103      hose discharge
```

```
450031              23103       tank assembly
450060              23103           bracket 1
450074              30135     guard rear left
450313              30135  bumper front lower
Found 776 matches in chunk:
       VEHICLE_MODEL_CODE                            TXT_PARTS_NAME
500086              23103                                 holder rh
500151              23103  fender wheel well lining guard cover right
500237              23103                          emblem maruti suzuk
500255              23103                              tail light left
500427              23103                         holder rr bumper rh
Found 824 matches in chunk:
       VEHICLE_MODEL_CODE            TXT_PARTS_NAME
550145              23103          tail light right
550312              23103  fender panel front right
550317              30135        engine guard cover
550412              23103      bumper rear assembly
550449              23103           head light left
Found 760 matches in chunk:
       VEHICLE_MODEL_CODE          TXT_PARTS_NAME
600649              30135    guard assembly rr rh
600833              23103    bumper rear assembly
601075              23103   bumper front assembly
601230              23103  windshield glass front
601231              23103    bonnet hood assembly
Found 811 matches in chunk:
       VEHICLE_MODEL_CODE          TXT_PARTS_NAME
650046              23103   bumper front assembly
650348              23103         handle rr dr rh
650404              23103   grille radiator upper
650407              23103  windshield glass front
650509              23103              emblem vvt
Found 778 matches in chunk:
       VEHICLE_MODEL_CODE                 TXT_PARTS_NAME
700075              23103                       bezel rh
700247              30135  absorber comp frt bumper lower
700398              23103         left hand fender lining
700482              30135              bumper front lower
700534              30135         emblem logo fender left
Found 797 matches in chunk:
       VEHICLE_MODEL_CODE               TXT_PARTS_NAME
750102              23103  emblem logo rear left top 1
750103              23103                garnish dicky
750105              23103      fender panel front left
750232              30135       tail light right dicky
750233              30135      hinge dicky trunk right
Found 742 matches in chunk:
       VEHICLE_MODEL_CODE                   TXT_PARTS_NAME
800062              23103                             tyre
800063              23103                             bulb
800077              23103            bumper front assembly
800140              23103           bezel front fog lamp l
800217              23103  sealant front windshield glass 1
Found 824 matches in chunk:
       VEHICLE_MODEL_CODE          TXT_PARTS_NAME
850099              30135                 tie rod
850100              30135  disc front right brake
850101              30135  frame suspension front
850102              30135     hose radiator lower
850103              30135                   guard
Found 700 matches in chunk:
       VEHICLE_MODEL_CODE                       TXT_PARTS_NAME
900036              23103                          front glass
900044              23103             moulding front windshield
900062              30135                  bonnet hood assembly
900064              30135                   cowl panel assembly
```

```
900065              30135  panel front right pillar inner right

Total matches found: 14968
Filtering complete!
```

In [58]:
```python
import pandas as pd
import json

# Load mappings
with open('Bigdeduplicated_mapping.json') as f:
    mapping_data = json.load(f)

# Create reverse lookup
reverse_map = {}
for key, aliases in mapping_data.items():
    for alias in aliases:
        reverse_map[alias.strip().lower()] = key

# Process CSV with CORRECT DELIMITER
input_csv = 'filtered_garage_by_Vehicle_Number_of_Servey.csv'
output_csv = 'ServeyModel_Invoice_Data(Replaced).csv'
chunk_size = 10000

# 1. Split columns properly using comma delimiter
first_chunk = True

for chunk in pd.read_csv(
    input_csv,
    sep=',',  # Changed to comma delimiter
    header=0,
    names=['VEHICLE_MODEL_CODE', 'TXT_PARTS_NAME', 'TOTAL_AMOUNT'],  # Explicit column names
    chunksize=chunk_size
):
    # Normalize part descriptions
    chunk['clean_part'] = chunk['TXT_PARTS_NAME'].str.strip().str.lower()

    # Map to new descriptions
    chunk['TXT_PARTS_NAME'] = chunk['clean_part'].map(reverse_map).fillna(chunk['TXT_PARTS_NAI

    # Save results
    chunk[['VEHICLE_MODEL_CODE', 'TXT_PARTS_NAME', 'TOTAL_AMOUNT']] \
        .to_csv(
            output_csv,
            mode='a',
            header=first_chunk,
            index=False
        )
    first_chunk = False

print("Processing complete!")
```

```
Processing complete!
```

Generating the Invoice for the garage based on the dataset created and the model number.

In [69]:
```python
import pandas as pd
from datetime import datetime

def generate_price_range_invoice():
    try:
        # Load and preprocess data
        df = pd.read_csv('GarageModel_Invoice_Data(Replaced).csv',
```

```
                dtype={'VEHICLE_MODEL_CODE': str},
                converters={'TOTAL_AMOUNT': float})

    # Clean data and handle duplicates
    df['PARTDESCRIPTION'] = df['PARTDESCRIPTION'].str.strip().str.title()

    # Group by vehicle model and part description to find price ranges
    grouped = df.groupby(['VEHICLE_MODEL_CODE', 'PARTDESCRIPTION']).agg(
        min_price=('TOTAL_AMOUNT', 'min'),
        max_price=('TOTAL_AMOUNT', 'max'),
        occurrence_count=('TOTAL_AMOUNT', 'count')
    ).reset_index()

    # Create price range strings
    grouped['price_range'] = grouped.apply(
        lambda x: f"${x['min_price']:.2f} - ${x['max_price']:.2f}"
        if x['min_price'] != x['max_price']
        else f"${x['min_price']:.2f}",
        axis=1
    )


    # Calculate vehicle-wise subtotals
    vehicle_totals = df.groupby('VEHICLE_MODEL_CODE').agg(
        total_spent=('TOTAL_AMOUNT', 'sum'),
        part_count=('PARTDESCRIPTION', 'count')
    ).reset_index()

    # Generate HTML invoice
    invoice_date = datetime.now().strftime('%Y-%m-%d')
    invoice_number = f"INV-{datetime.now().strftime('%Y%m%d-%H%M%S')}"

    html_content = f"""
    <html>
    <head>
        <style>
            body {{ font-family: Arial, sans-serif; margin: 2cm; }}
            .header {{ text-align: center; border-bottom: 2px solid #000; }}
            table {{ width: 100%; border-collapse: collapse; margin-top: 20px; }}
            th, td {{ padding: 12px; text-align: left; border-bottom: 1px solid #ddd; }}
            .price-range {{ color: #2c3e50; font-weight: bold; }}
            .total-row {{ background-color: #f8f9fa; }}
        </style>
    </head>
    <body>
        <div class="header">
            <h1>Vehicle Parts Price Analysis By the Garage Agent</h1>
            <p>Report Number: {invoice_number}<br>
            Date: {invoice_date}</p>
        </div>
    """

    for vehicle in vehicle_totals['VEHICLE_MODEL_CODE'].unique():
        vehicle_data = grouped[grouped['VEHICLE_MODEL_CODE'] == vehicle]
        total_info = vehicle_totals[vehicle_totals['VEHICLE_MODEL_CODE'] == vehicle].iloc

        html_content += f"""
        <div style="margin-top: 30px;">
            <h3>Vehicle Model: {vehicle}</h3>
            <table>
                <tr>
                    <th>Part Description</th>
                    <th>Price Range</th>
                    <th></th>
                </tr>"""
```

```
            for _, row in vehicle_data.iterrows():
                html_content += f"""
                    <tr>
                        <td>{row['PARTDESCRIPTION']}</td>
                        <td class="price-range">{row['price_range']}</td>
                        <td>{row['occurrence_count']}</td>
                    </tr>"""

            html_content += f"""
                    <tr class="total-row">
                        <td colspan="2"><strong>Total Parts Purchased</strong></td>
                        <td><strong>{total_info['part_count']}</strong></td>
                    </tr>
                    <tr class="total-row">
                        <td colspan="2"><strong>Total Amount Spent</strong></td>
                        <td><strong>${total_info['total_spent']:.2f}</strong></td>
                    </tr>
                </table>
            </div>
            """

        # Add grand totals
        grand_total = vehicle_totals['total_spent'].sum()
        html_content += f"""
            <div style="margin-top: 40px; border-top: 2px solid #000;">
                <h2>Grand Total Across All Vehicles: ${grand_total:.2f}</h2>
            </div>
        </body>
        </html>
        """

        # Save report
        with open('vehicle_parts_analysis_BY_Garage_Agent.html', 'w') as f:
            f.write(html_content)

        print("Report generated successfully as vehicle_parts_analysis BY Garage Agent.html")

    except Exception as e:
        print(f"Error generating report: {str(e)}")

if __name__ == "__main__":
    generate_price_range_invoice()
```

```
Report generated successfully as vehicle_parts_analysis BY Garage Agent.html
```

Generating the Invoice for the survey based on the dataset created and the model number.

In [72]:

```
import pandas as pd
from datetime import datetime


def generate_price_range_invoiceByServey():
    try:
        # Load data and skip the repeated header row
        df = pd.read_csv('ServeyModel_Invoice_Data(Replaced).csv', dtype={'VEHICLE_MODEL_CODE

        # Ensure TOTAL_AMOUNT is numeric
        df['TOTAL_AMOUNT'] = pd.to_numeric(df['TOTAL_AMOUNT'], errors='coerce')  # Convert in
        df = df.dropna(subset=['TOTAL_AMOUNT'])  # Drop rows with NaN in TOTAL_AMOUNT
        df['TOTAL_AMOUNT'] = df['TOTAL_AMOUNT'].astype(float)  # Ensure TOTAL_AMOUNT is a flo

        # Clean data and handle duplicates
```

```
df['TXT_PARTS_NAME'] = df['TXT_PARTS_NAME'].str.strip().str.title()

# Group by vehicle model and part description to find price ranges
grouped = df.groupby(['VEHICLE_MODEL_CODE', 'TXT_PARTS_NAME']).agg(
    min_price=('TOTAL_AMOUNT', 'min'),
    max_price=('TOTAL_AMOUNT', 'max'),
    occurrence_count=('TOTAL_AMOUNT', 'count')
).reset_index()

# Create price range strings
grouped['price_range'] = grouped.apply(
    lambda x: f"${x['min_price']:.2f} - ${x['max_price']:.2f}"
    if x['min_price'] != x['max_price']
    else f"${x['min_price']:.2f}",
    axis=1
)

# Calculate vehicle-wise subtotals
vehicle_totals = df.groupby('VEHICLE_MODEL_CODE').agg(
    total_spent=('TOTAL_AMOUNT', 'sum'),
    part_count=('TXT_PARTS_NAME', 'count')
).reset_index()

# Generate HTML invoice
invoice_date = datetime.now().strftime('%Y-%m-%d')
invoice_number = f"INV-{datetime.now().strftime('%Y%m%d-%H%M%S')}"

html_content = f"""
<html>
<head>
    <style>
        body {{ font-family: Arial, sans-serif; margin: 2cm; }}
        .header {{ text-align: center; border-bottom: 2px solid #000; }}
        table {{ width: 100%; border-collapse: collapse; margin-top: 20px; }}
        th, td {{ padding: 12px; text-align: left; border-bottom: 1px solid #ddd; }}
        .price-range {{ color: #2c3e50; font-weight: bold; }}
        .total-row {{ background-color: #f8f9fa; }}
    </style>
</head>
<body>
    <div class="header">
        <h1>Vehicle Parts Price Analysis By the Servey Agent</h1>
        <p>Report Number: {invoice_number}<br>
        Date: {invoice_date}</p>
    </div>
"""

for vehicle in vehicle_totals['VEHICLE_MODEL_CODE'].unique():
    vehicle_data = grouped[grouped['VEHICLE_MODEL_CODE'] == vehicle]
    total_info = vehicle_totals[vehicle_totals['VEHICLE_MODEL_CODE'] == vehicle].iloc

    html_content += f"""
    <div style="margin-top: 30px;">
        <h3>Vehicle Model: {vehicle}</h3>
        <table>
            <tr>
                <th>Part Description</th>
                <th>Price Range</th>
                <th>Occurrences</th>
            </tr>"""

    for _, row in vehicle_data.iterrows():
        html_content += f"""
            <tr>
                <td>{row['TXT_PARTS_NAME']}</td>
                <td class="price-range">{row['price_range']}</td>
```

```
                        <td>{row['occurrence_count']}</td>
                    </tr>"""

            html_content += f"""
                    <tr class="total-row">
                        <td colspan="2"><strong>Total Parts Purchased</strong></td>
                        <td><strong>{total_info['part_count']}</strong></td>
                    </tr>
                    <tr class="total-row">
                        <td colspan="2"><strong>Total Amount Spent</strong></td>
                        <td><strong>${total_info['total_spent']:.2f}</strong></td>
                    </tr>
                </table>
            </div>
            """

        # Add grand totals
        grand_total = vehicle_totals['total_spent'].sum()
        html_content += f"""
            <div style="margin-top: 40px; border-top: 2px solid #000;">
                <h2>Grand Total Across All Vehicles: ${grand_total:.2f}</h2>
            </div>
        </body>
        </html>
        """

        # Save report
        with open('vehicle_parts_analysis_BY_Servey_Agent.html', 'w') as f:
            f.write(html_content)

        print("Report generated successfully as vehicle_parts_analysis_BY_Servey_Agent.html")

    except Exception as e:
        print(f"Error generating report: {str(e)}")


if __name__ == "__main__":
    generate_price_range_invoiceByServey()
```

```
Report generated successfully as vehicle_parts_analysis_BY_Servey_Agent.html
```

This script compares two invoices (Garage invoice and Survey invoice) to create a comparison invoice. The comparison details include:

1. Matched parts.
2. Mismatched parts.
3. Differences in the cost of parts.

In [111…

```
import pandas as pd
import numpy as np
from datetime import datetime


def compare_invoices(file1_path, file2_path):
    try:
        # Load datasets
        df1 = pd.read_csv(
            file1_path,
            dtype={'VEHICLE_MODEL_CODE': str},
            usecols=['VEHICLE_MODEL_CODE', 'PARTDESCRIPTION', 'TOTAL_AMOUNT']
        ).rename(columns={'PARTDESCRIPTION': 'PART_NAME'})

        df2 = pd.read_csv(
            file2_path,
            dtype={'VEHICLE_MODEL_CODE': str},
```

```
        usecols=['VEHICLE_MODEL_CODE', 'TXT_PARTS_NAME', 'TOTAL_AMOUNT']
    ).rename(columns={'TXT_PARTS_NAME': 'PART_NAME'})

    # Data cleaning function
    def clean_data(df):
        # Clean vehicle codes
        df['VEHICLE_MODEL_CODE'] = (
            df['VEHICLE_MODEL_CODE']
            .astype(str)
            .str.replace(r'\D', '', regex=True)
            .str.strip()
        )

        # Clean part names
        df['PART_NAME'] = (
            df['PART_NAME']
            .astype(str)
            .str.strip()
            .str.lower()
            .replace('nan', np.nan)
        )

        # Clean amounts
        df['TOTAL_AMOUNT'] = (
            df['TOTAL_AMOUNT']
            .astype(str)
            .str.replace(',', '')
            .str.replace(r'[^\d.]', '', regex=True)
            .replace(r'^\.$', np.nan, regex=True)
            .replace('', np.nan)
            .astype(float)
        )

        return df.dropna(subset=['PART_NAME', 'TOTAL_AMOUNT'])

    # Clean and aggregate data
    df1_clean = clean_data(df1)
    df2_clean = clean_data(df2)

    # Aggregate data with occurrence count
    def aggregate_data(df):
        aggregated = df.groupby(['VEHICLE_MODEL_CODE', 'PART_NAME']).agg(
            min_price=('TOTAL_AMOUNT', 'min'),
            max_price=('TOTAL_AMOUNT', 'max'),
            occurrences=('TOTAL_AMOUNT', 'count')
        ).reset_index()

        # Convert occurrences to integer type
        aggregated['occurrences'] = aggregated['occurrences'].astype(int)
        return aggregated

    # Aggregate and rename columns to avoid full prefixes for common fields (VEHICLE_MODEL
    agg1 = aggregate_data(df1_clean).rename(columns={
        'min_price': 'Garage_min_price',
        'max_price': 'Garage_max_price',
        'occurrences': 'Garage_occurrences'
    })

    agg2 = aggregate_data(df2_clean).rename(columns={
        'min_price': 'Servey_min_price',
        'max_price': 'Servey_max_price',
        'occurrences': 'Servey_occurrences'
    })

    # Merge aggregated data
    comparison = pd.merge(
```

```
            agg1,
            agg2,
            on=['VEHICLE_MODEL_CODE', 'PART_NAME'],
            how='outer',
            indicator=True
        )

        # Generate report
        generate_html_report(comparison)
        print("Report generated successfully: invoice_comparison.html")

    except Exception as e:
        error_html = f"""
        <html><body>
            <h1>Error</h1><p>{str(e)}</p>
        </body></html>
        """
        with open('invoice_comparison.html', 'w', encoding='utf-8') as f:
            f.write(error_html)
        print(f"Error: {str(e)}")


def generate_html_report(comparison):
    # Create content sections
    matched_content = create_matched_table(comparison)
    garage_unique = create_unique_table(comparison, 'left')
    servey_unique = create_unique_table(comparison, 'right')

    html_content = f"""
    <html>
    <head>
        <meta charset="UTF-8">
        <title>Invoice Comparison</title>
        <style>
            body {{ font-family: Arial, sans-serif; margin: 2em; }}
            table {{ border-collapse: collapse; width: 100%; }}
            th, td {{ padding: 8px; text-align: left; border-bottom: 1px solid #ddd; }}
            .match {{ background-color: #e8f5e9; }}
            .garage-only {{ background-color: #fff3e0; }}
            .servey-only {{ background-color: #fbe9e7; }}
        </style>
    </head>
    <body>
        <h1>Invoice Comparison Report</h1>
        <p>Generated: {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}</p>

        <h2>Matching Parts in Both Files</h2>
        {matched_content if matched_content else "<p>No matching parts found</p>"}

        <h2>Unique to Garage Data</h2>
        {garage_unique if garage_unique else "<p>No unique parts in Garage data</p>"}

        <h2>Unique to Servey Data</h2>
        {servey_unique if servey_unique else "<p>No unique parts in Servey data</p>"}
    </body>
    </html>
    """

    with open('invoice_comparison.html', 'w', encoding='utf-8') as f:
        f.write(html_content)


def create_matched_table(comparison):
    matched = comparison[comparison['_merge'] == 'both']
    if matched.empty:
        return ""
```

```
    return matched[[
        'VEHICLE_MODEL_CODE', 'PART_NAME',
        'Garage_min_price', 'Garage_max_price', 'Garage_occurrences',
        'Servey_min_price', 'Servey_max_price', 'Servey_occurrences'
    ]].to_html(index=False, classes='match', border=0)


def create_unique_table(comparison, side):
    prefix = 'Garage' if side == 'left' else 'Servey'
    unique = comparison[comparison['_merge'] == f'{side}_only']
    if unique.empty:
        return ""

    return unique[[
        'VEHICLE_MODEL_CODE', 'PART_NAME',
        f'{prefix}_min_price', f'{prefix}_max_price', f'{prefix}_occurrences'
    ]].to_html(index=False, classes=f'{prefix}-only', border=0)


# Execute comparison
compare_invoices('GarageModel_Invoice_Data(Replaced).csv', 'ServeyModel_Invoice_Data(Replaced
```

```
Report generated successfully: invoice_comparison.html
```

here we categories the primary parts and secondary parts on both data set

In [113…

```
import pandas as pd
import json

# Load data
df = pd.read_csv("../garagedata_cleaned_data.csv")

# Load part mapping JSON
with open("Bigdeduplicated_mapping.json") as f:
    part_mapping = json.load(f)

# Create reverse mapping and primary part set
reverse_map = {}
primary_part_set = set()

for primary_part, secondary_list in part_mapping.items():
    normalized_primary = primary_part.lower().strip()
    primary_part_set.add(normalized_primary)

    for secondary in secondary_list:
        normalized_secondary = secondary.lower().strip()
        reverse_map[normalized_secondary] = normalized_primary

# Process parts data
def categorize_part(part_description):
    normalized_part = part_description.lower().strip()

    if normalized_part in reverse_map:
        return reverse_map[normalized_part]
    elif normalized_part in primary_part_set:
        return normalized_part
    else:
        return None

df['primary_part'] = df['PARTDESCRIPTION'].apply(categorize_part)
df['secondary_part'] = df.apply(
    lambda x: x['PARTDESCRIPTION'] if pd.isnull(x['primary_part']) else None,
    axis=1
)
```

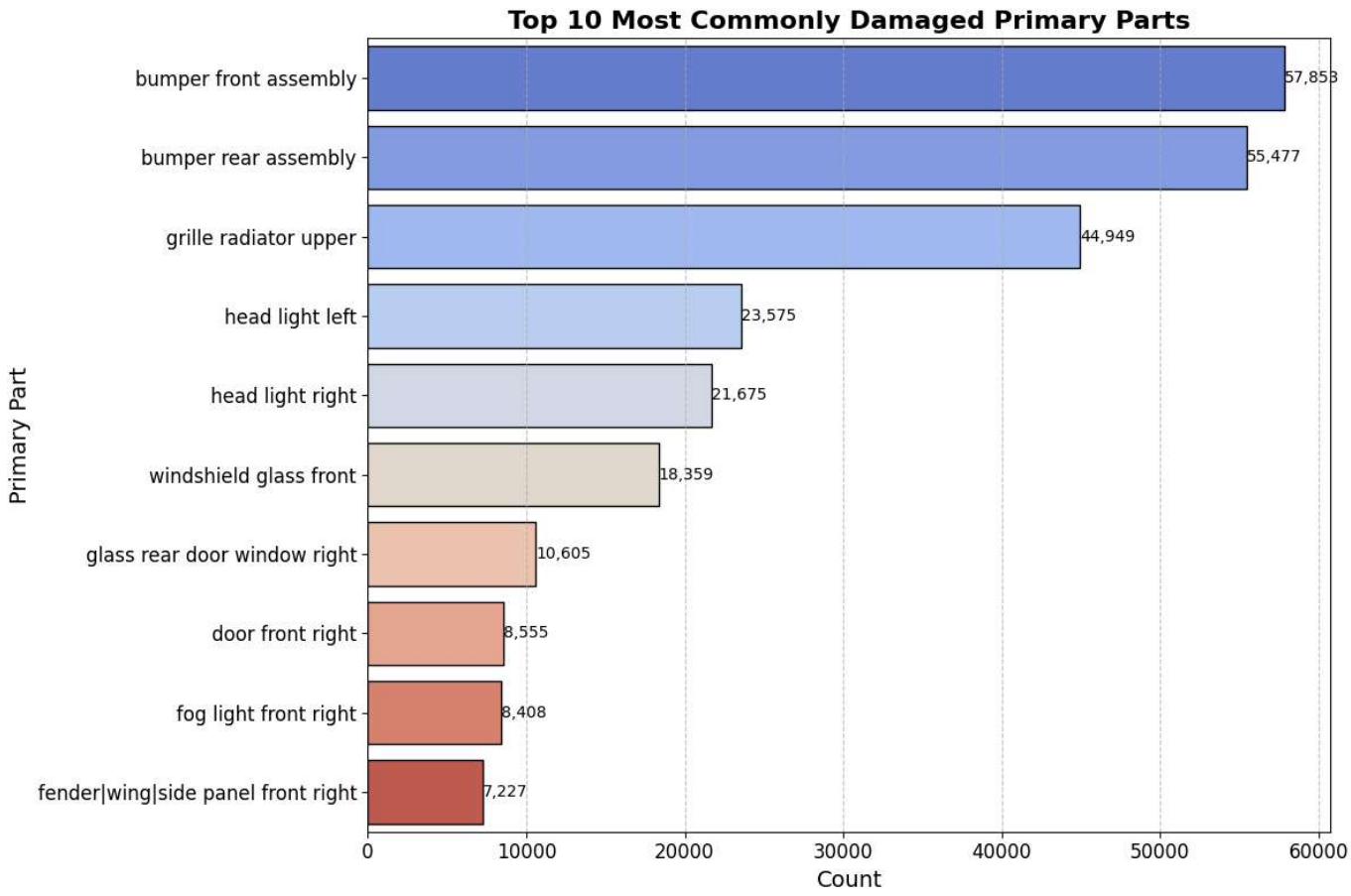
Displaying the Most Commonly Damaged Primary Parts:

In [142…

```
import matplotlib.pyplot as plt
import seaborn as sns

# Calculate primary part analysis
primary_analysis = df['primary_part'].value_counts().reset_index()
primary_analysis.columns = ['Primary Part', 'Count']
total = primary_analysis['Count'].sum()
primary_analysis['Percentage'] = (primary_analysis['Count'] / total * 100).round(2)

# Display most commonly damaged primary parts
print("Most Commonly Damaged Primary Parts:")
print(primary_analysis.head(10))

# Enhanced visualization: Horizontal bar plot for better readability
plt.figure(figsize=(12, 8))
sns.barplot(
    x='Count',
    y='Primary Part',
    data=primary_analysis.head(10),  # Display top 10
    palette='coolwarm',
    edgecolor='black'
)

# Add labels and title
plt.title("Top 10 Most Commonly Damaged Primary Parts", fontsize=16, fontweight='bold')
plt.xlabel("Count", fontsize=14)
plt.ylabel("Primary Part", fontsize=14)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Add value annotations on the bars
for index, value in enumerate(primary_analysis['Count'][:10]):
    plt.text(value + 1, index, f"{value:,}", va='center', fontsize=10, color='black')

plt.tight_layout()
plt.grid(axis='x', linestyle='--', alpha=0.7)  # Add gridlines for better readability
plt.show()
```

```
Most Commonly Damaged Primary Parts:
                        Primary Part  Count  Percentage
0               bumper front assembly  57853       20.07
1                bumper rear assembly  55477       19.24
2               grille radiator upper  44949       15.59
3                     head light left  23575        8.18
4                    head light right  21675        7.52
5             windshield glass front  18359        6.37
6       glass rear door window right  10605        3.68
7                    door front right   8555        2.97
8               fog light front right   8408        2.92
9  fender|wing|side panel front right   7227        2.51
```

```
C:\Users\Aman Jaiswal\AppData\Local\Temp\ipykernel_11004\1810664673.py:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(
```



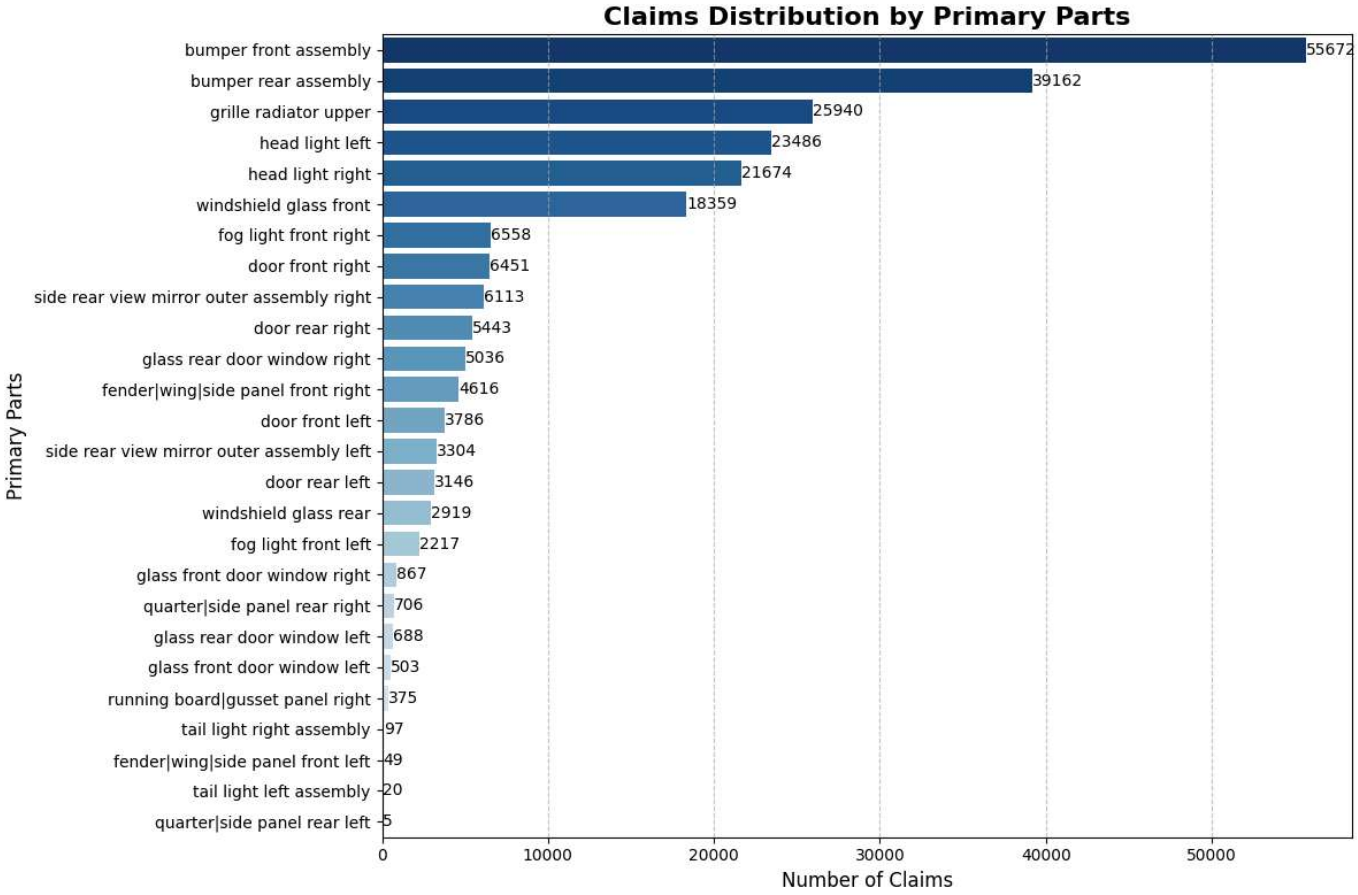
Here are the Claims Distribution Visuals on primary parts

In [141…

```python
# Improved visualization for claims distribution
import matplotlib.pyplot as plt
import seaborn as sns

# Data processing
claims_distribution = df.groupby('primary_part')['CLAIMNO'].nunique().reset_index()
claims_distribution.columns = ['Primary Part', 'Claim Count']
claims_distribution = claims_distribution.sort_values('Claim Count', ascending=False)

# Plot the data
plt.figure(figsize=(12, 8))
sns.barplot(
    x='Claim Count',
    y='Primary Part',
    data=claims_distribution,
    palette='Blues_r'
)

# Add labels and title
plt.title("Claims Distribution by Primary Parts", fontsize=16, fontweight='bold')
plt.xlabel("Number of Claims", fontsize=12)
plt.ylabel("Primary Parts", fontsize=12)
plt.xticks(fontsize=10)
plt.yticks(fontsize=10)
plt.grid(axis='x', linestyle='--', alpha=0.7)

# Add value annotations on bars
for index, value in enumerate(claims_distribution['Claim Count']):
    plt.text(value + 1, index, str(value), va='center', fontsize=10, color='black')

plt.tight_layout()
plt.show()

# Print the data (optional)
```

```
print("\nClaims Distribution by Primary Parts:")
print(claims_distribution)
```

```
C:\Users\Aman Jaiswal\AppData\Local\Temp\ipykernel_11004\3577289864.py:12: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(
```



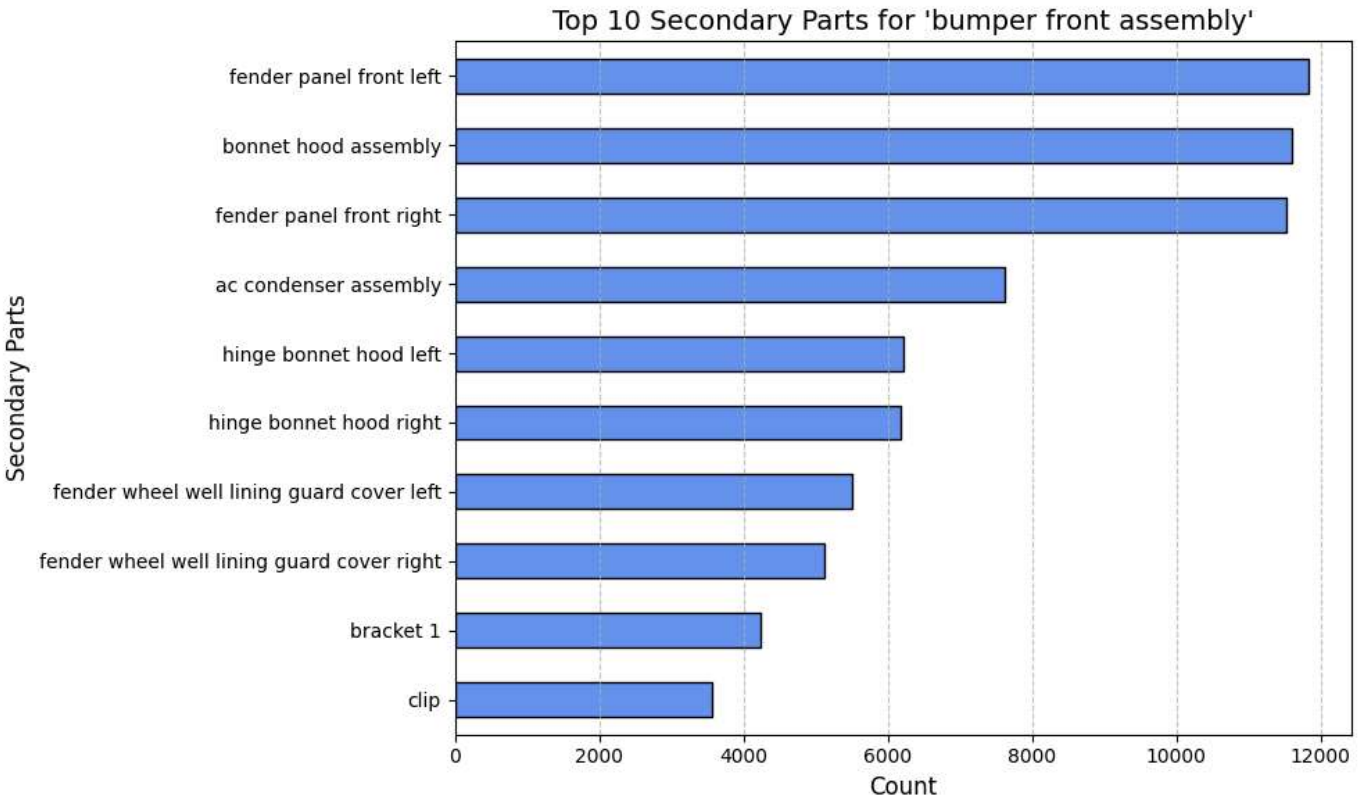

```
Claims Distribution by Primary Parts:
                                  Primary Part  Claim Count
0                         bumper front assembly        55672
1                          bumper rear assembly        39162
14                        grille radiator upper        25940
15                              head light left        23486
16                             head light right        21674
24                       windshield glass front        18359
9                         fog light front right         6558
3                              door front right         6451
21  side rear view mirror outer assembly right         6113
5                              door rear right         5443
13                 glass rear door window right         5036
7            fender|wing|side panel front right         4616
2                              door front left         3786
20   side rear view mirror outer assembly left         3304
4                               door rear left         3146
25                        windshield glass rear         2919
8                          fog light front left         2217
11                glass front door window right          867
18                quarter|side panel rear right          706
12                  glass rear door window left          688
10                glass front door window left          503
19             running board|gusset panel right          375
23                    tail light right assembly           97
6             fender|wing|side panel front left           49
22                     tail light left assembly           20
17                quarter|side panel rear left            5
```

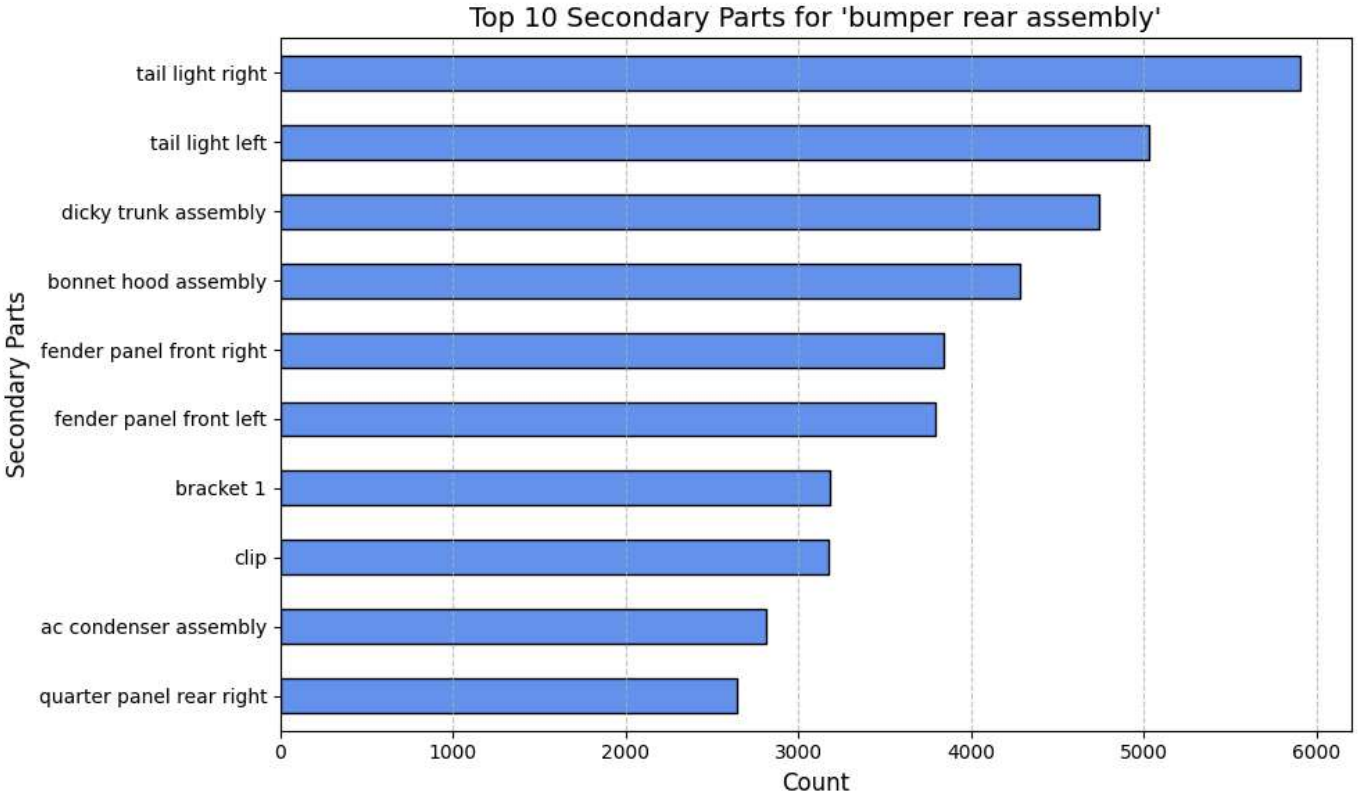
Displaying the Top 10 Secondary Parts on the primary parts:

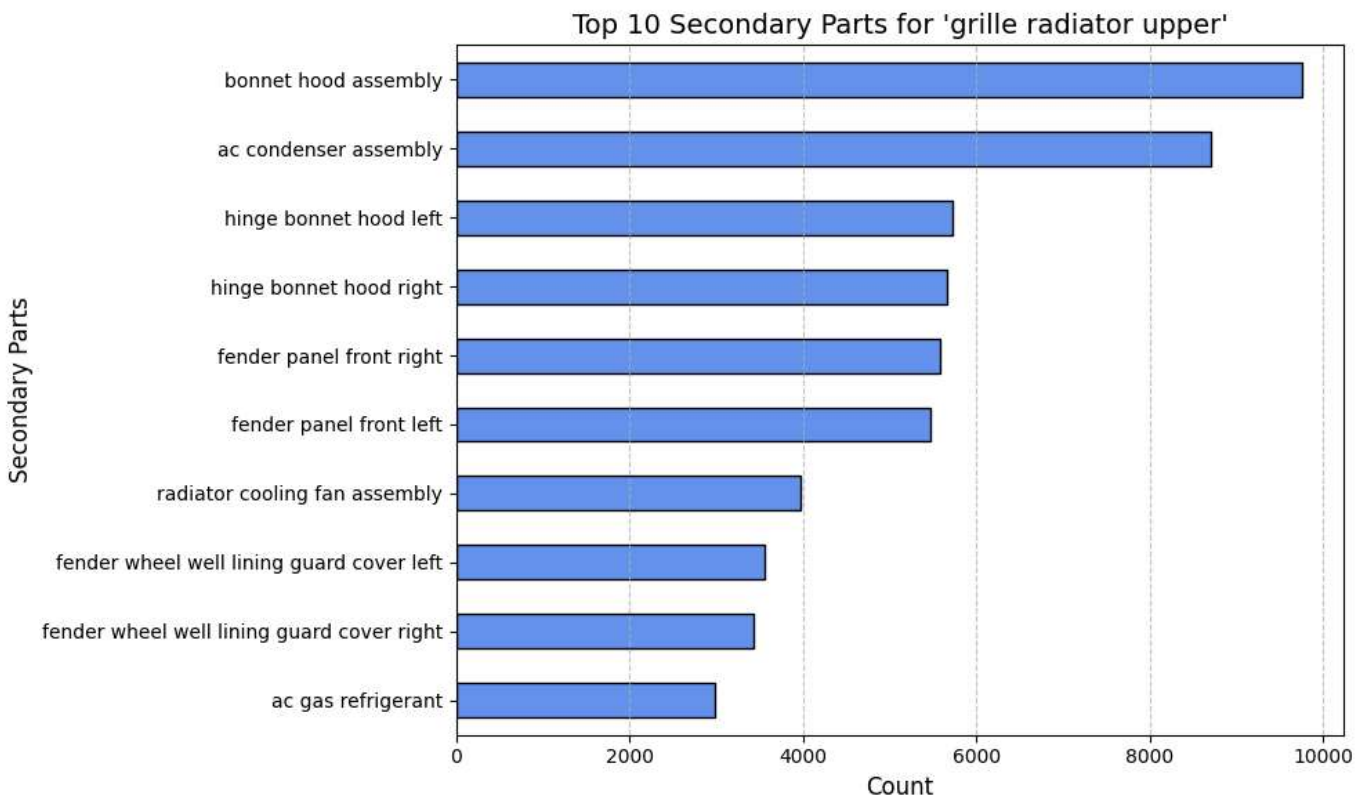
In [137…

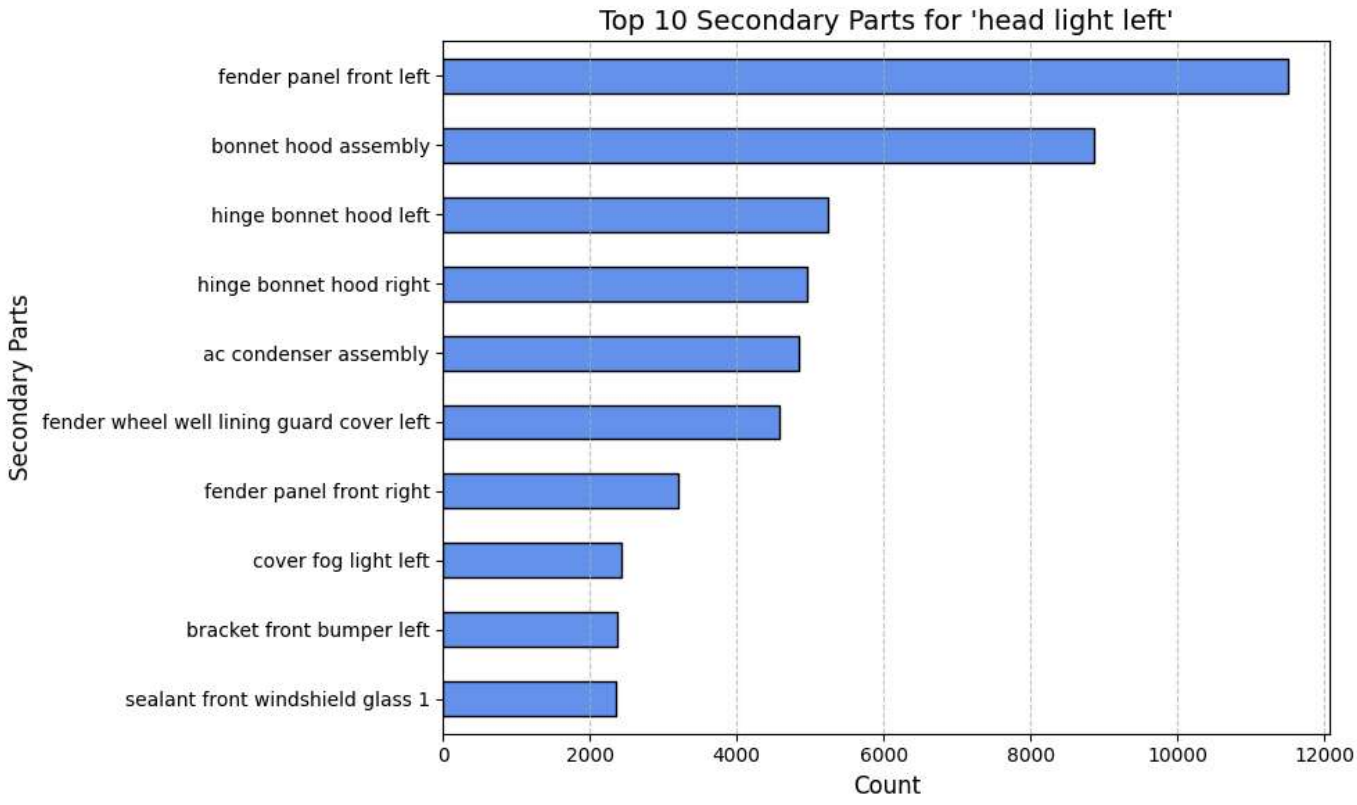
```
# Create claim-part matrix
claim_groups = df.groupby('CLAIMNO').agg({
    'primary_part': list,
    'secondary_part': list
}).reset_index()

# Analyze secondary associations
secondary_associations = {}

# Collect data for all top associations for heatmap creation
heatmap_data = []

for primary in primary_analysis['Primary Part'][:10]:  # Limit to top 10 for better readabili
    # Get claims containing this primary part
    relevant_claims = claim_groups[
        claim_groups['primary_part'].apply(lambda x: primary in x if x else False)
    ]

    # Collect all secondary parts from these claims
    all_secondary = [
        part for sublist in relevant_claims['secondary_part']
        for part in sublist if part
    ]

    # Count occurrences
    counter = pd.Series(all_secondary).value_counts().head(10)  # Limit to top 10 secondary a
    secondary_associations[primary] = counter

    # Prepare data for heatmap (fill NaN with 0 for missing keys)
    heatmap_data.append(counter.reindex(secondary_associations.keys(), fill_value=0))

    # Plotting bar chart for each primary part and its top secondary associations
    plt.figure(figsize=(10, 6))
    counter.sort_values(ascending=True).plot(kind='barh', color='cornflowerblue', edgecolor='
    plt.title(f"Top 10 Secondary Parts for '{primary}'", fontsize=14)
    plt.xlabel("Count", fontsize=12)
    plt.ylabel("Secondary Parts", fontsize=12)
    plt.tight_layout()
    plt.grid(axis='x', linestyle='--', alpha=0.7)
    plt.show()

# Combine heatmap data into a DataFrame
heatmap_df = pd.DataFrame(heatmap_data).fillna(0)

# Plot heatmap for secondary associations using seaborn
import seaborn as sns

plt.figure(figsize=(12, 8))
sns.heatmap(
    heatmap_df,
    annot=False,
    cmap="Blues",
    cbar_kws={'label': 'Count'},
    yticklabels=secondary_associations.keys()  # Primary parts as Y-axis labels
)
plt.title("Heatmap of Top Secondary Associations Across Primary Parts", fontsize=14)
plt.xlabel("Secondary Parts", fontsize=12)
plt.ylabel("Primary Parts", fontsize=12)
plt.tight_layout()
plt.show()

# Print the top associations (optional)
for primary, secondaries in secondary_associations.items():
```

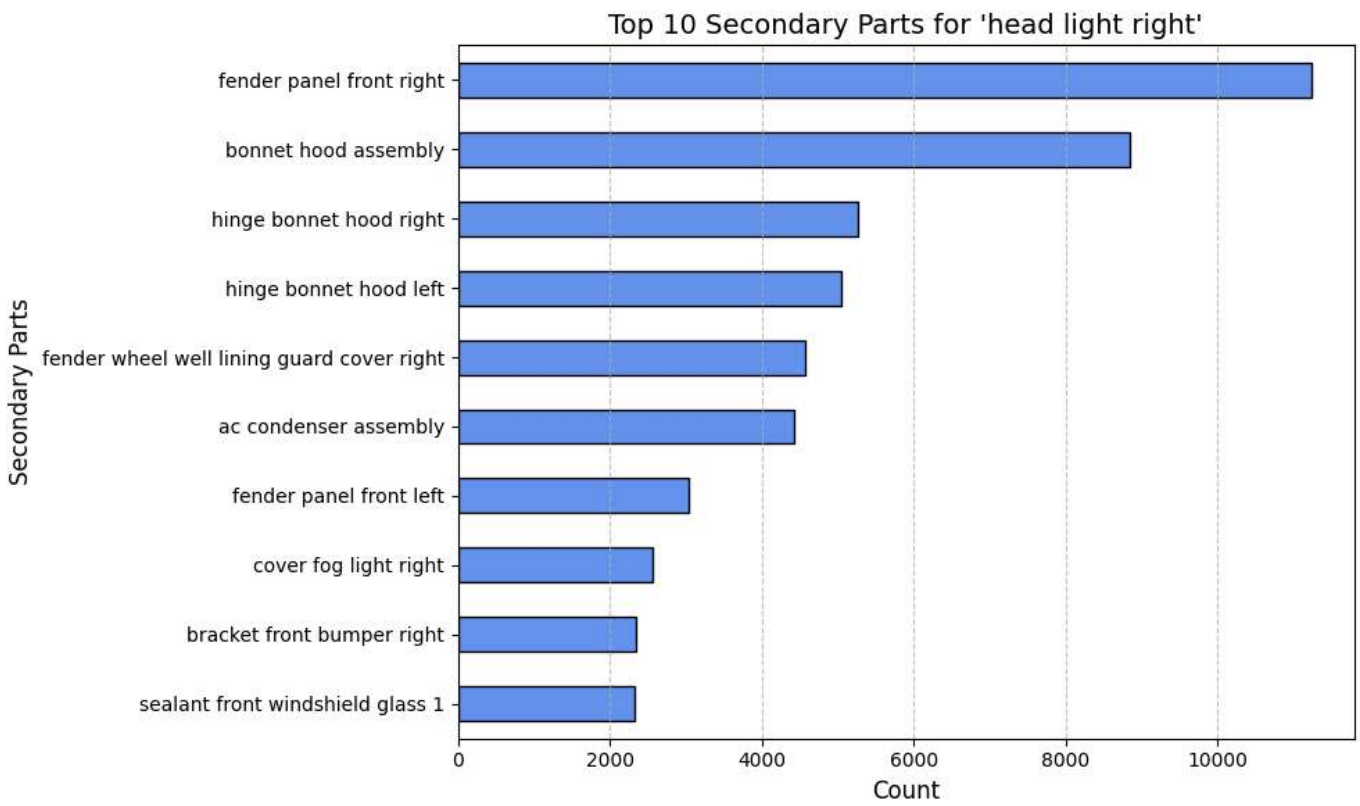
```
    print(f"\nTop 10 Secondary Parts for '{primary}':")
    print(secondaries)
```

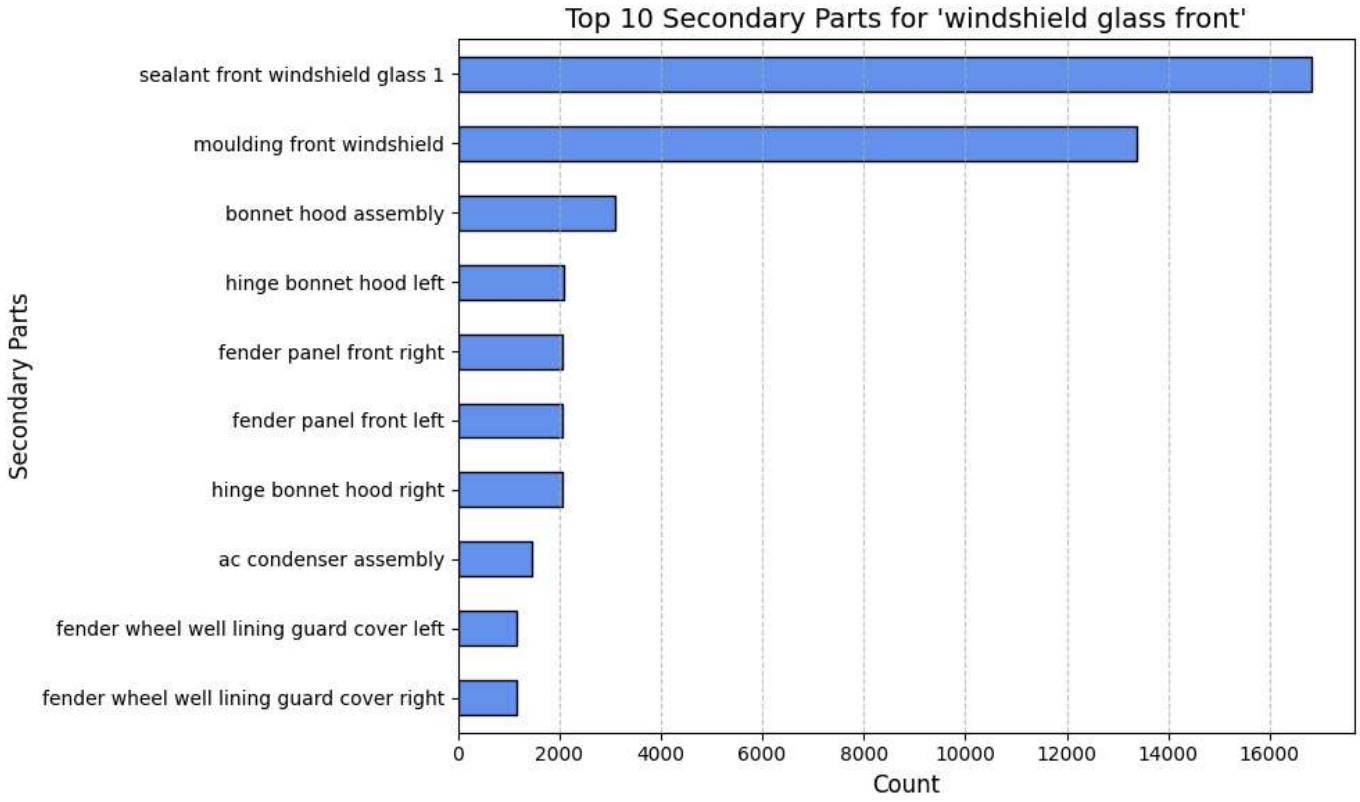


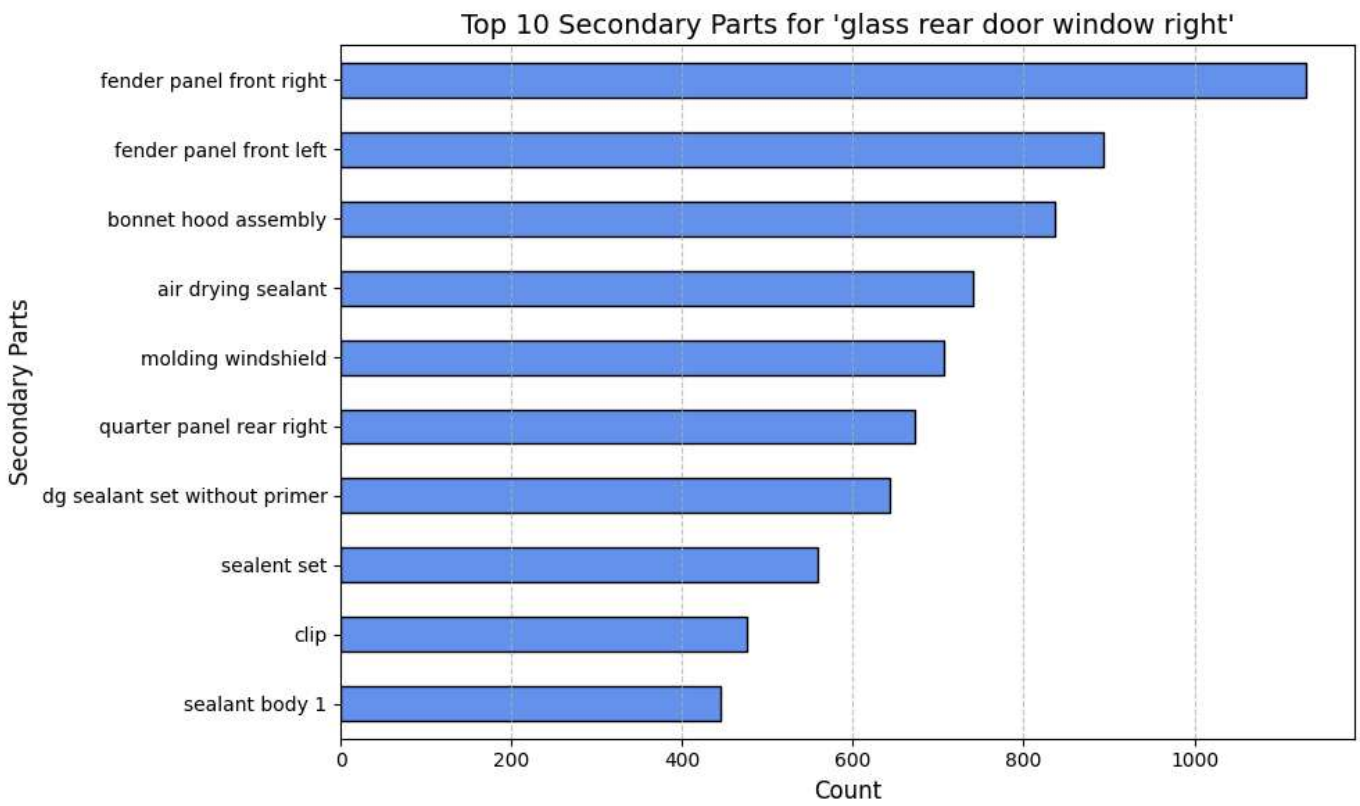


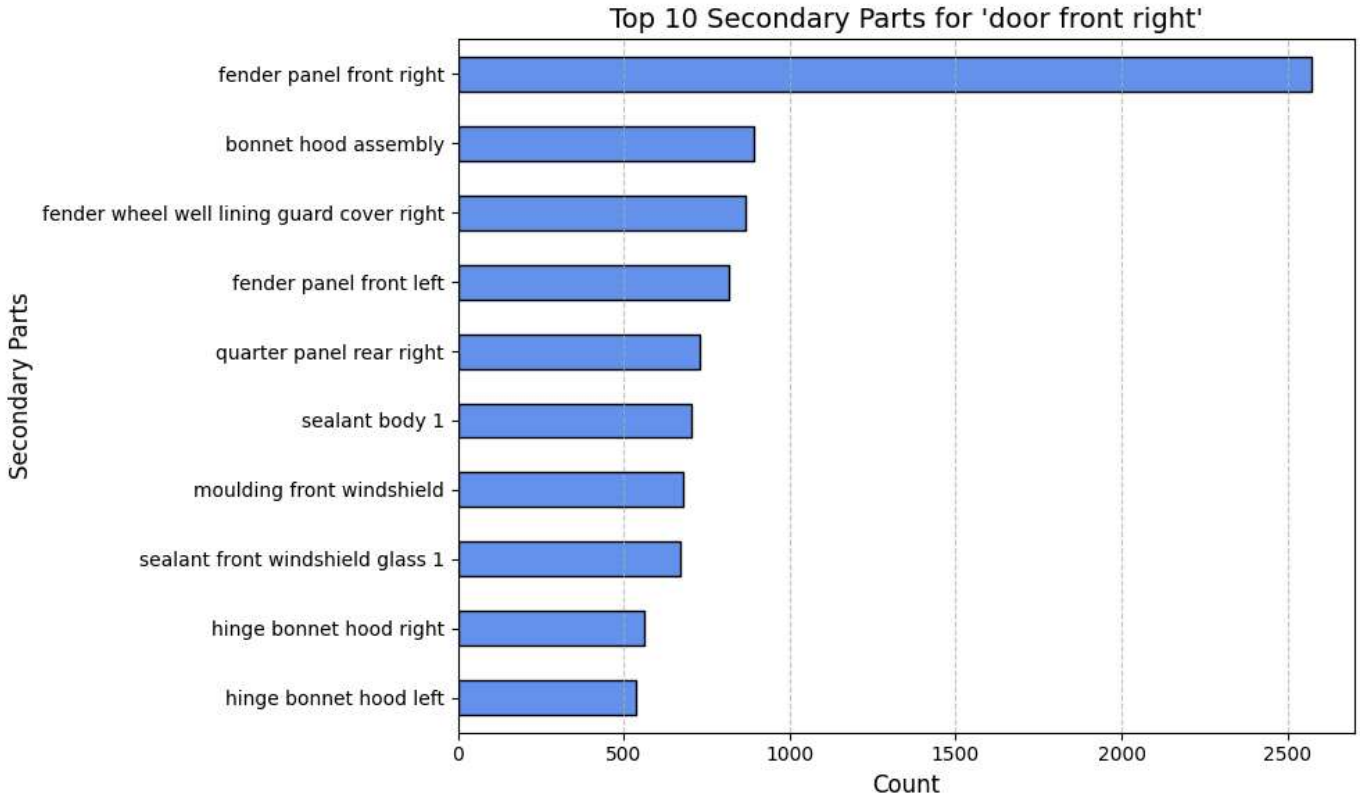

Top 10 Secondary Parts for 'grille radiator upper'
bonnet hood assembly
ac condenser assembly
hinge bonnet hood left
hinge bonnet hood right
fender panel front right
fender panel front left
radiator cooling fan assembly
fender wheel well lining guard cover left
fender wheel well lining guard cover right
ac gas refrigerant
Secondary Parts
0
2000
4000
6000
8000
10000
Count

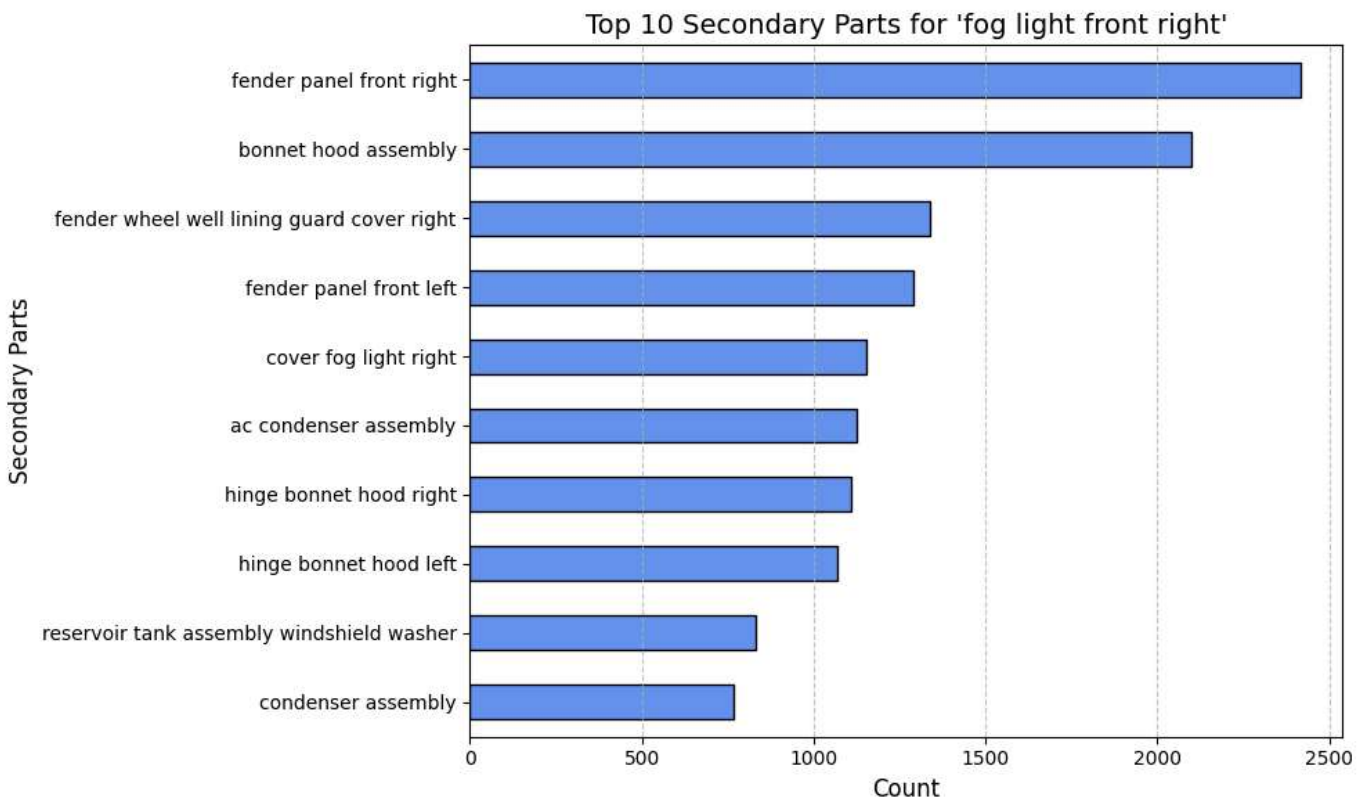

Top 10 Secondary Parts for 'head light left'
fender panel front left
bonnet hood assembly
hinge bonnet hood left
hinge bonnet hood right
ac condenser assembly
fender wheel well lining guard cover left
fender panel front right
cover fog light left
bracket front bumper left
sealant front windshield glass 1
Secondary Parts
0
2000
4000
6000
8000
10000
12000
Count

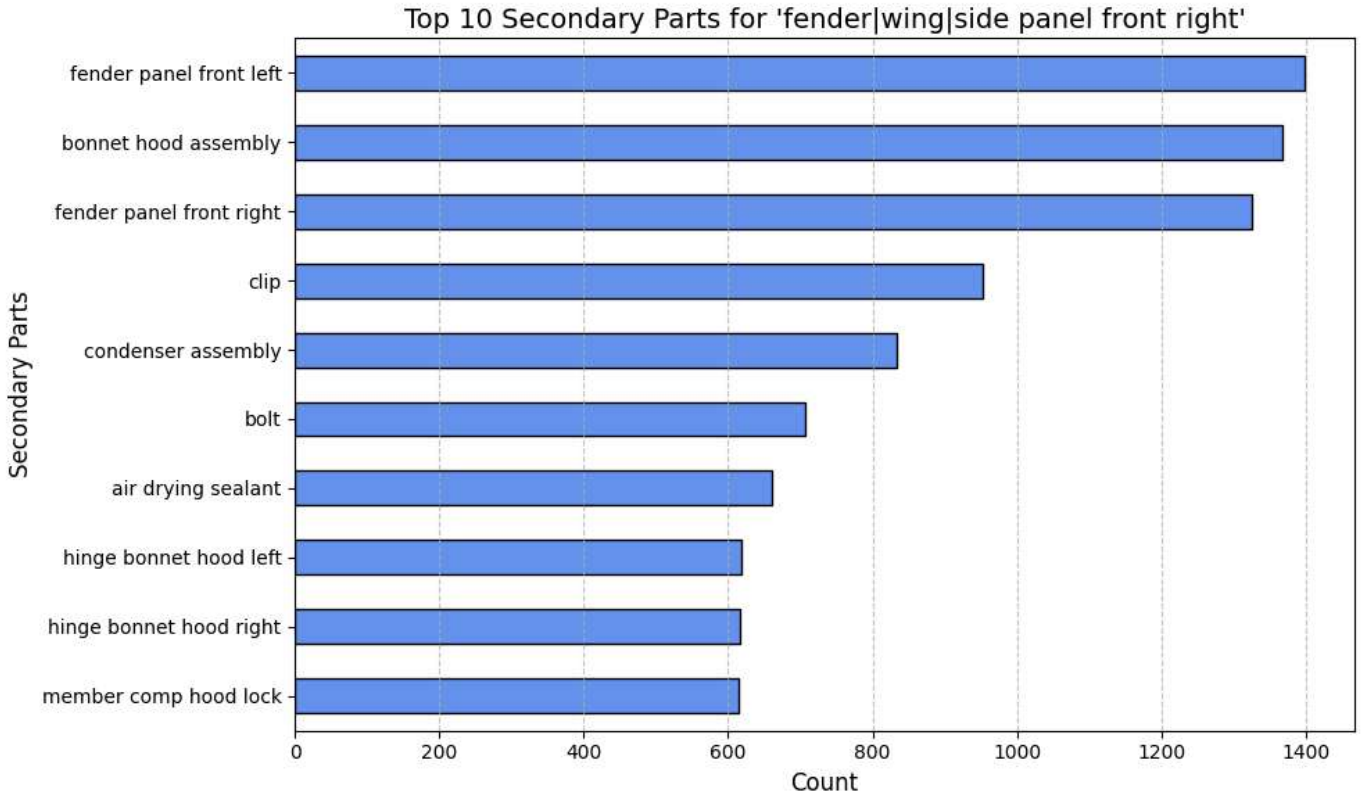

Top 10 Secondary Parts for 'head light right'
fender panel front right
bonnet hood assembly
hinge bonnet hood right
hinge bonnet hood left
fender wheel well lining guard cover right
ac condenser assembly
fender panel front left
cover fog light right
bracket front bumper right
sealant front windshield glass 1
Secondary Parts
0
2000
4000
6000
8000
10000
Count


Top 10 Secondary Parts for 'windshield glass front'
sealant front windshield glass 1
moulding front windshield
bonnet hood assembly
hinge bonnet hood left
fender panel front right
fender panel front left
hinge bonnet hood right
ac condenser assembly
fender wheel well lining guard cover left
fender wheel well lining guard cover right
Secondary Parts
0
2000
4000
6000
8000
10000
12000
14000
16000
Count


Top 10 Secondary Parts for 'glass rear door window right'
Secondary Parts
fender panel front right
fender panel front left
bonnet hood assembly
air drying sealant
molding windshield
quarter panel rear right
dg sealant set without primer
sealent set
clip
sealant body 1
0
200
400
600
800
1000
Count


Top 10 Secondary Parts for 'door front right'
Secondary Parts
fender panel front right
bonnet hood assembly
fender wheel well lining guard cover right
fender panel front left
quarter panel rear right
sealant body 1
moulding front windshield
sealant front windshield glass 1
hinge bonnet hood right
hinge bonnet hood left
0
500
1000
1500
2000
2500
Count






```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In[137], line 46
     43 heatmap_df = pd.DataFrame(heatmap_data).fillna(0)
     45 # Plot heatmap for secondary associations using seaborn
---> 46 import seaborn as sns
     48 plt.figure(figsize=(12, 8))
     49 sns.heatmap(
     50     heatmap_df,
     51     annot=False,
   (...)
     54     yticklabels=secondary_associations.keys()  # Primary parts as Y-axis labels
     55 )

ModuleNotFoundError: No module named 'seaborn'
```

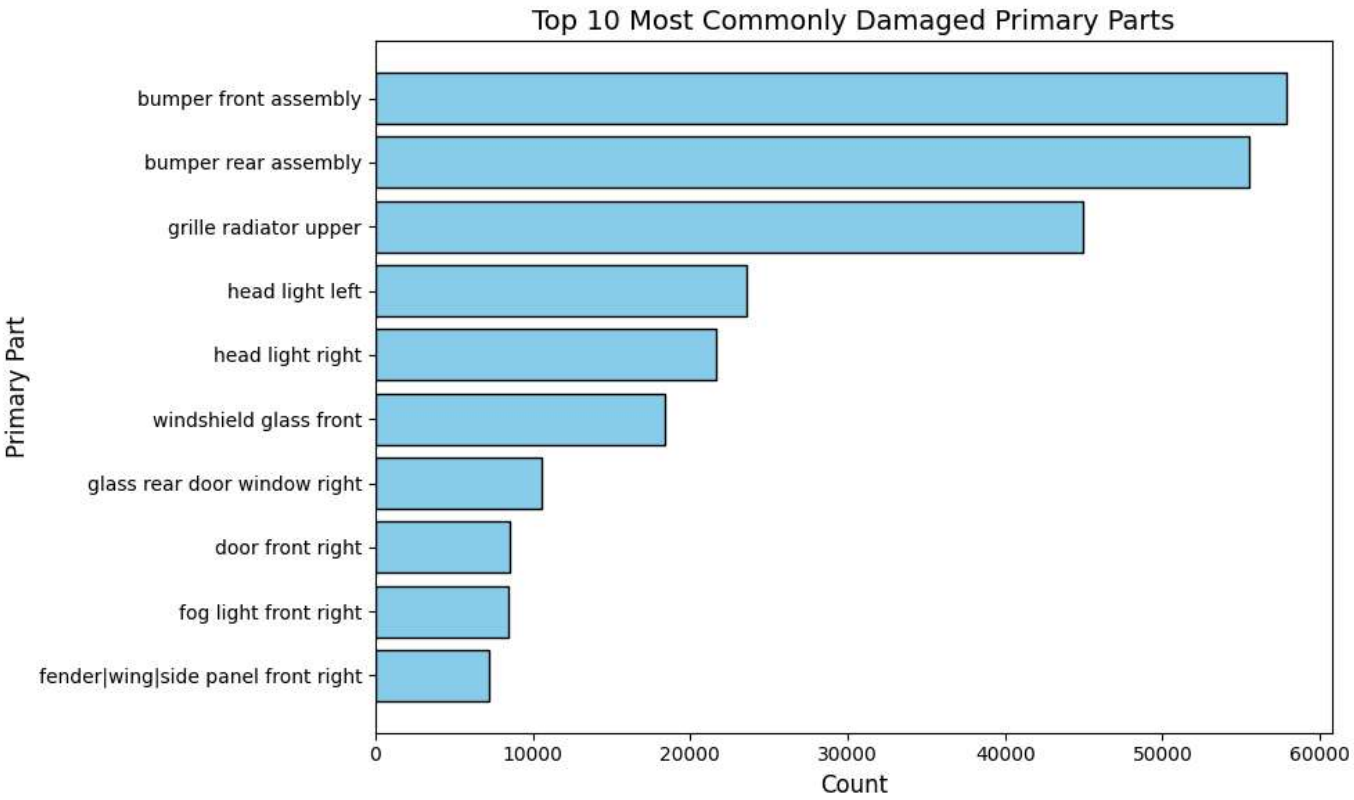
Now on Survey data Most 10 Most common damage parts on primary and secondary

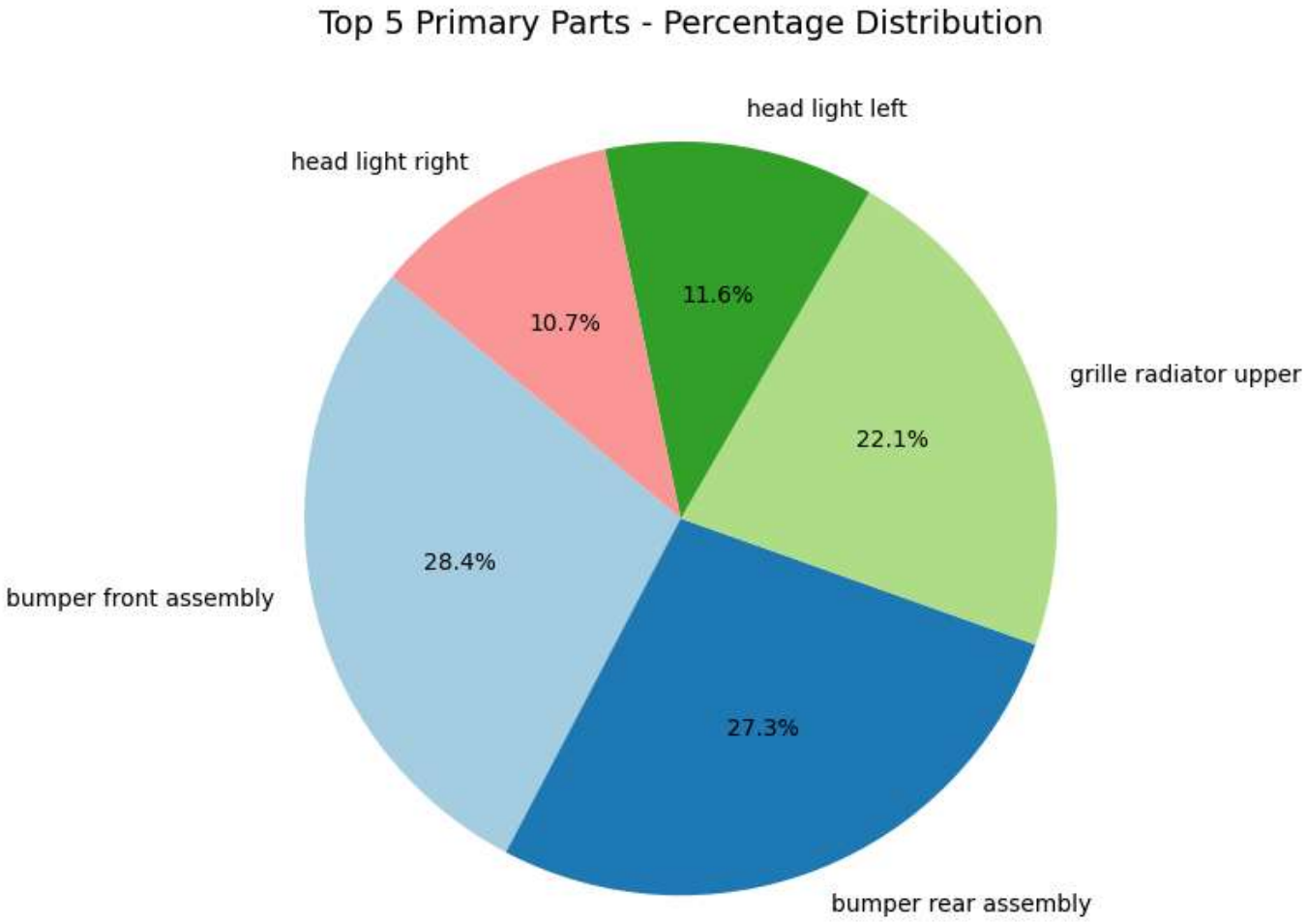
In [131…

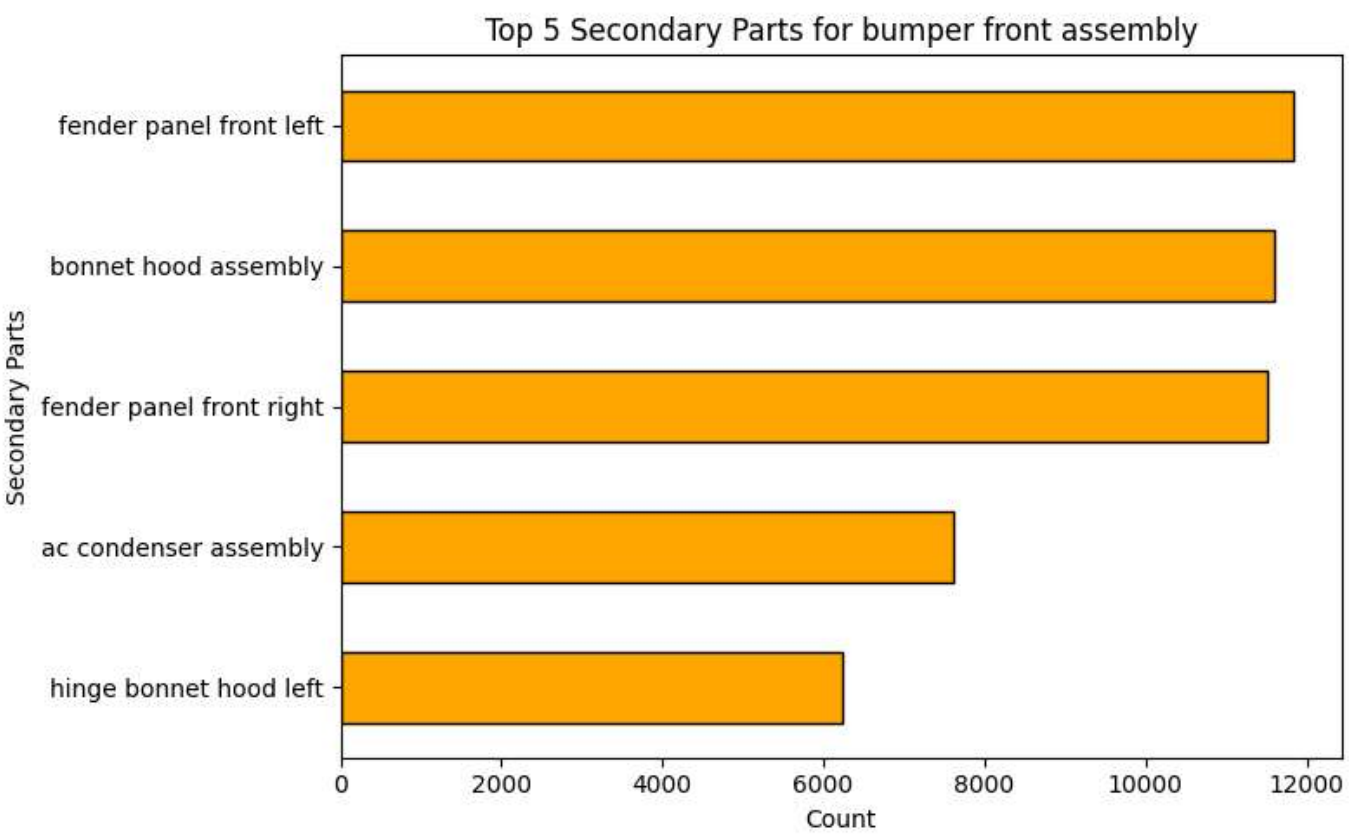
```python
import pandas as pd
import json
import numpy as np
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv("../surveyor_data_cleanedd.csv")

# Load part mapping JSON file
with open("Bigdeduplicated_mapping.json") as f:
    part_mapping = json.load(f)

# Create reverse mapping and primary part set
reverse_map = {}
primary_part_set = set()

for primary_part, secondary_list in part_mapping.items():
    # Normalize primary parts
    normalized_primary = primary_part.lower().strip()
    primary_part_set.add(normalized_primary)

    for secondary in secondary_list:
        # Normalize secondary parts
        normalized_secondary = secondary.lower().strip()
        reverse_map[normalized_secondary] = normalized_primary


# Function to categorize parts based on mappings
def categorize_part(part_description):
    # Handle missing/null part descriptions
    if pd.isnull(part_description):
        return None

    # Normalize the part description
    normalized_part = part_description.lower().strip()

    # Check primary or secondary part mappings
    if normalized_part in reverse_map:
        return reverse_map[normalized_part]  # Map to the primary part
    elif normalized_part in primary_part_set:
        return normalized_part  # It's already a primary part
    else:
        return None  # No match found


# Apply categorization to the `TXT_PARTS_NAME` column
df['primary_part'] = df['TXT_PARTS_NAME'].apply(categorize_part)

# Create a secondary part column for unmapped records
df['secondary_part'] = df.apply(
    lambda x: x['TXT_PARTS_NAME'] if pd.isnull(x['primary_part']) else None,
    axis=1
)

# Ensure `VEHICLE_MODEL_CODE` is converted to integers if appropriate
# and handle potential nulls in `VEHICLE_MODEL_CODE`
if df['VEHICLE_MODEL_CODE'].notnull().all():
    df['VEHICLE_MODEL_CODE'] = df['VEHICLE_MODEL_CODE'].astype(int, errors='ignore')
else:
    df['VEHICLE_MODEL_CODE'] = df['VEHICLE_MODEL_CODE'].fillna(0).astype(int)

# Analyze primary parts
primary_analysis = df['primary_part'].value_counts().reset_index()
```

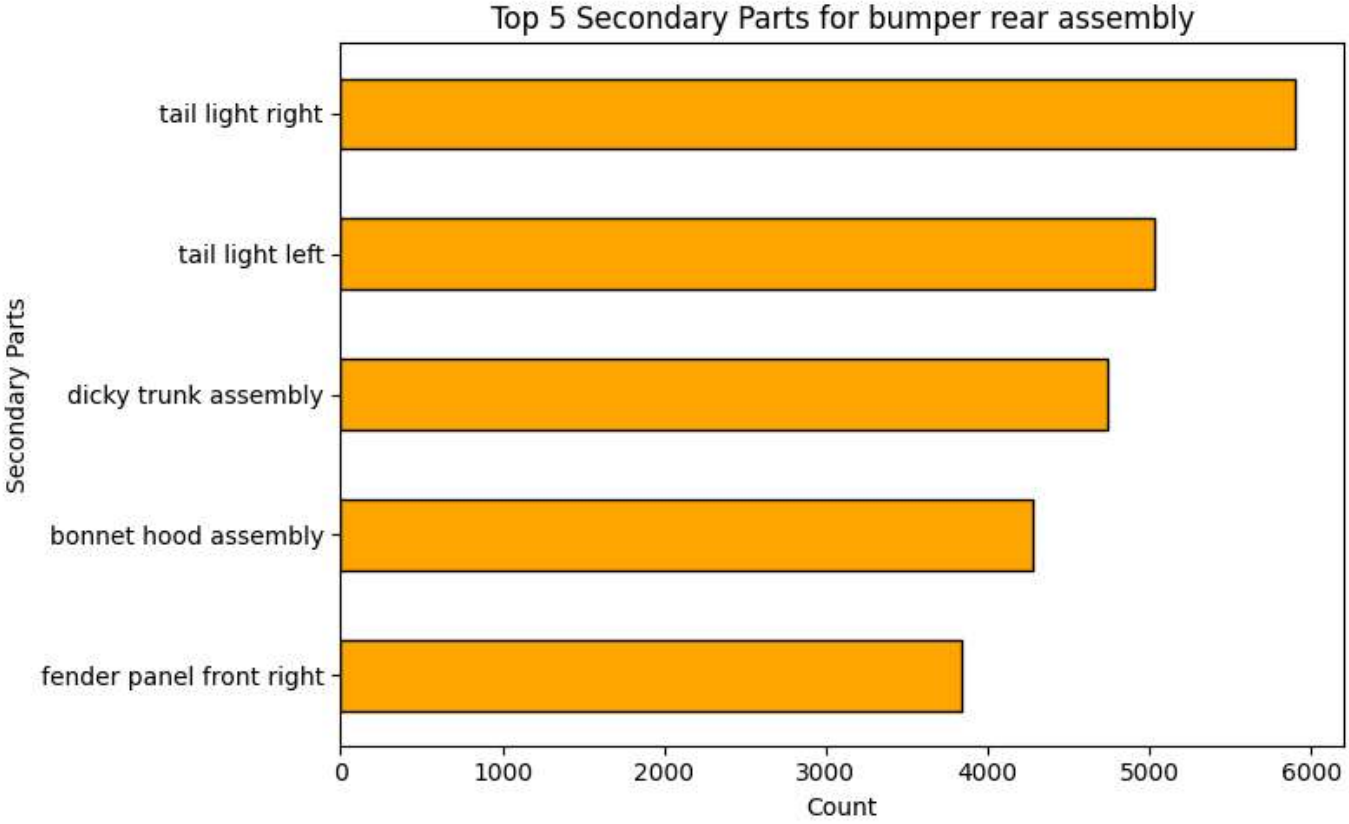
```python
primary_analysis.columns = ['Primary Part', 'Count']
total = primary_analysis['Count'].sum()
primary_analysis['Percentage'] = (primary_analysis['Count'] / total * 100).round(2)

# Visualization: Horizontal bar chart for top 10 most commonly damaged parts
plt.figure(figsize=(10, 6))
plt.barh(primary_analysis['Primary Part'][:10], primary_analysis['Count'][:10], color='skyblu
plt.gca().invert_yaxis()  # Largest value on top
plt.title("Top 10 Most Commonly Damaged Primary Parts", fontsize=14)
plt.xlabel("Count", fontsize=12)
plt.ylabel("Primary Part", fontsize=12)
plt.tight_layout()
plt.show()

# Visualization: Pie chart for percentage distribution of top 5 primary parts
plt.figure(figsize=(8, 8))
plt.pie(
    primary_analysis['Percentage'][:5],
    labels=primary_analysis['Primary Part'][:5],
    autopct='%1.1f%%',
    colors=plt.cm.Paired.colors,  # Use a color palette for better visuals
    startangle=140
)
plt.title("Top 5 Primary Parts - Percentage Distribution", fontsize=14)
plt.tight_layout()
plt.show()

# Create a secondary part analysis visualization
# Analyze secondary associations
claim_groups = df.groupby('CLAIMNO').agg({
    'primary_part': list,
    'secondary_part': list
}).reset_index()

secondary_associations = {}

for primary in primary_analysis['Primary Part'][:5]:  # Limit to top 5 for visualization
    # Get claims containing this primary part
    relevant_claims = claim_groups[
        claim_groups['primary_part'].apply(lambda x: primary in x if x else False)
    ]

    # Collect all secondary parts from these claims
    all_secondary = [
        part for sublist in relevant_claims['secondary_part']
        for part in sublist if part
    ]

    # Count occurrences
    counter = pd.Series(all_secondary).value_counts().head(5)  # Top 5 secondary associations
    secondary_associations[primary] = counter

    # Plot top secondary associations for the primary part
    plt.figure(figsize=(8, 5))
    counter.sort_values(ascending=True).plot(kind='barh', color='orange', edgecolor='black')
    plt.title(f"Top 5 Secondary Parts for {primary}", fontsize=12)
    plt.xlabel("Count", fontsize=10)
    plt.ylabel("Secondary Parts", fontsize=10)
    plt.tight_layout()
    plt.show()

# Output processed DataFrame (optional)
print(df[['primary_part', 'secondary_part']].head())
```

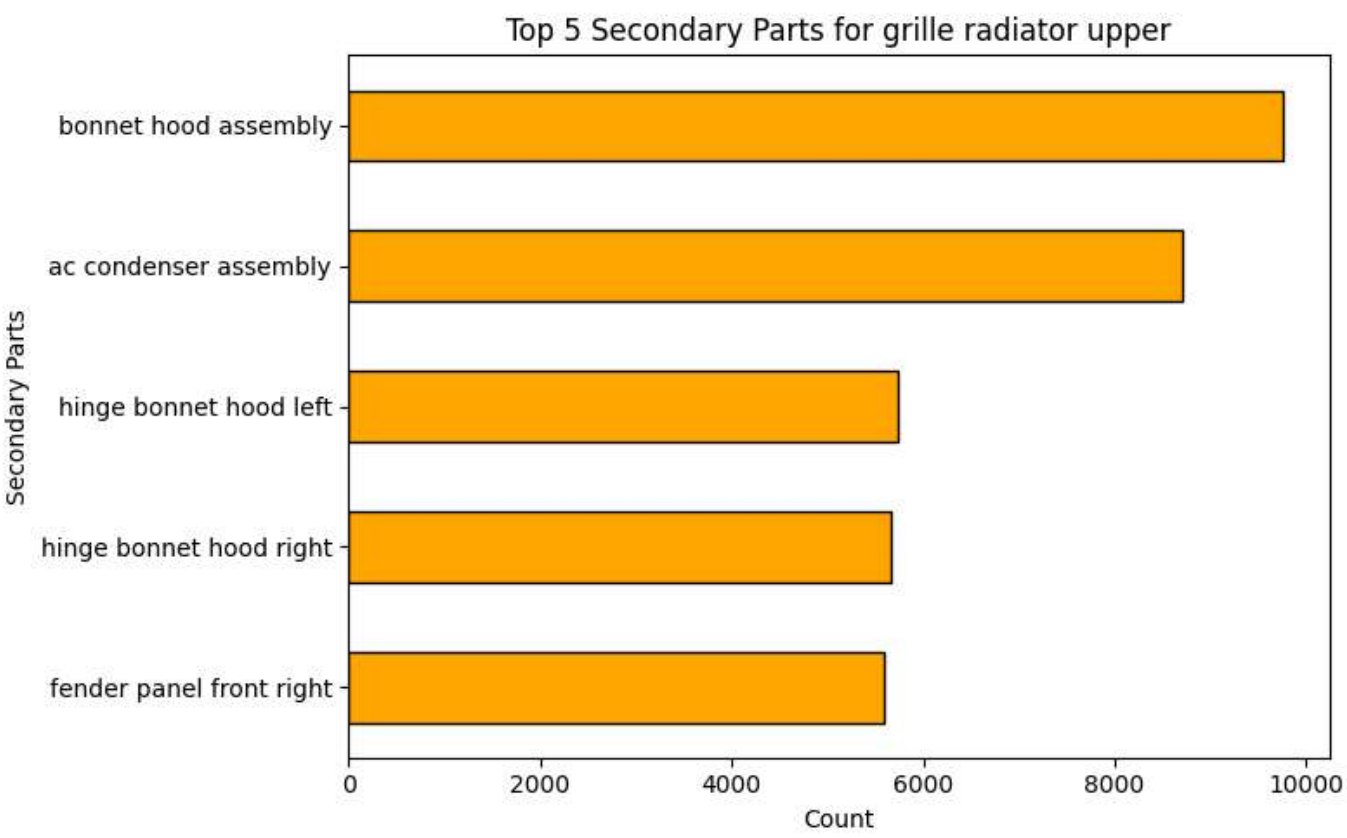
```
# Save the updated DataFrame to a new file (optional)
df.to_csv("processed_surveyor_data.csv", index=False)
```

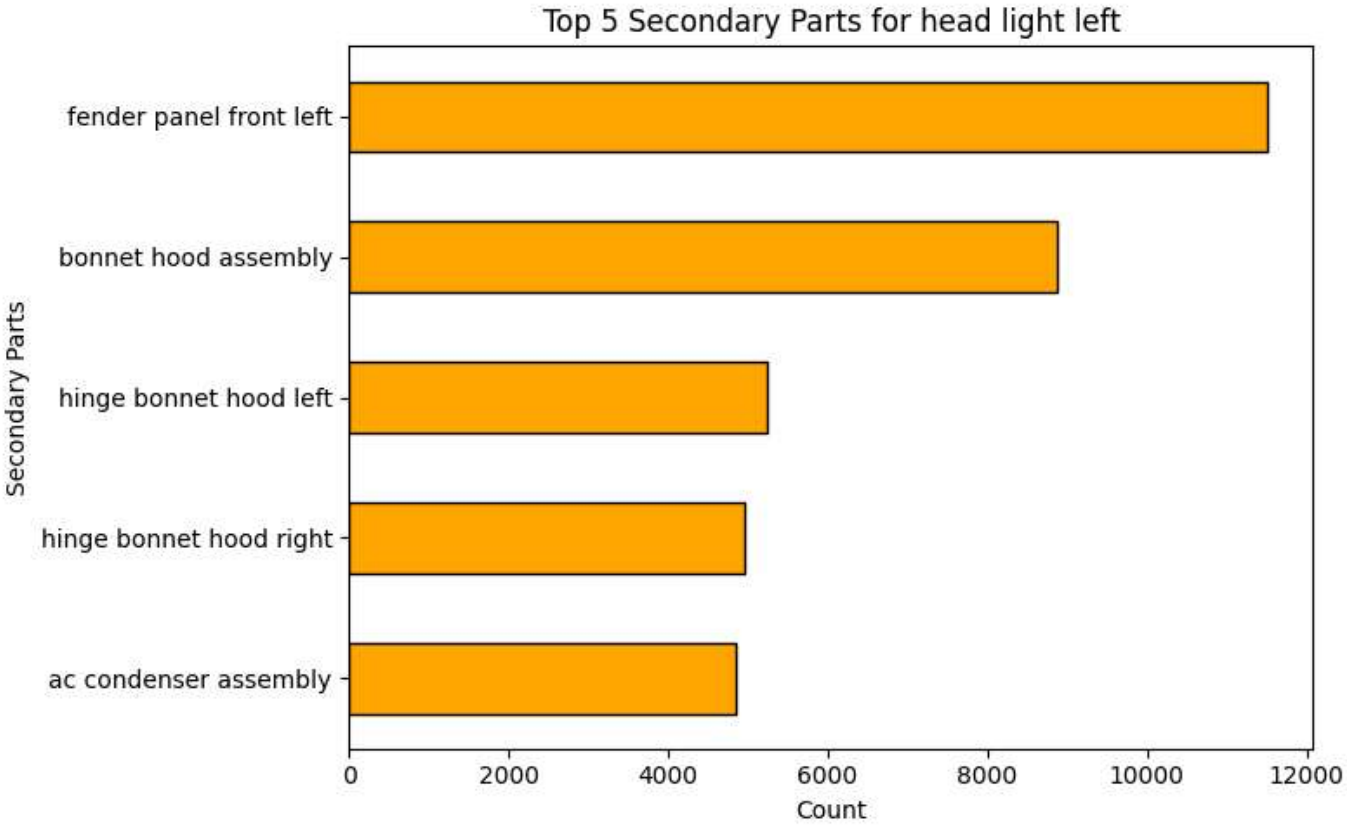


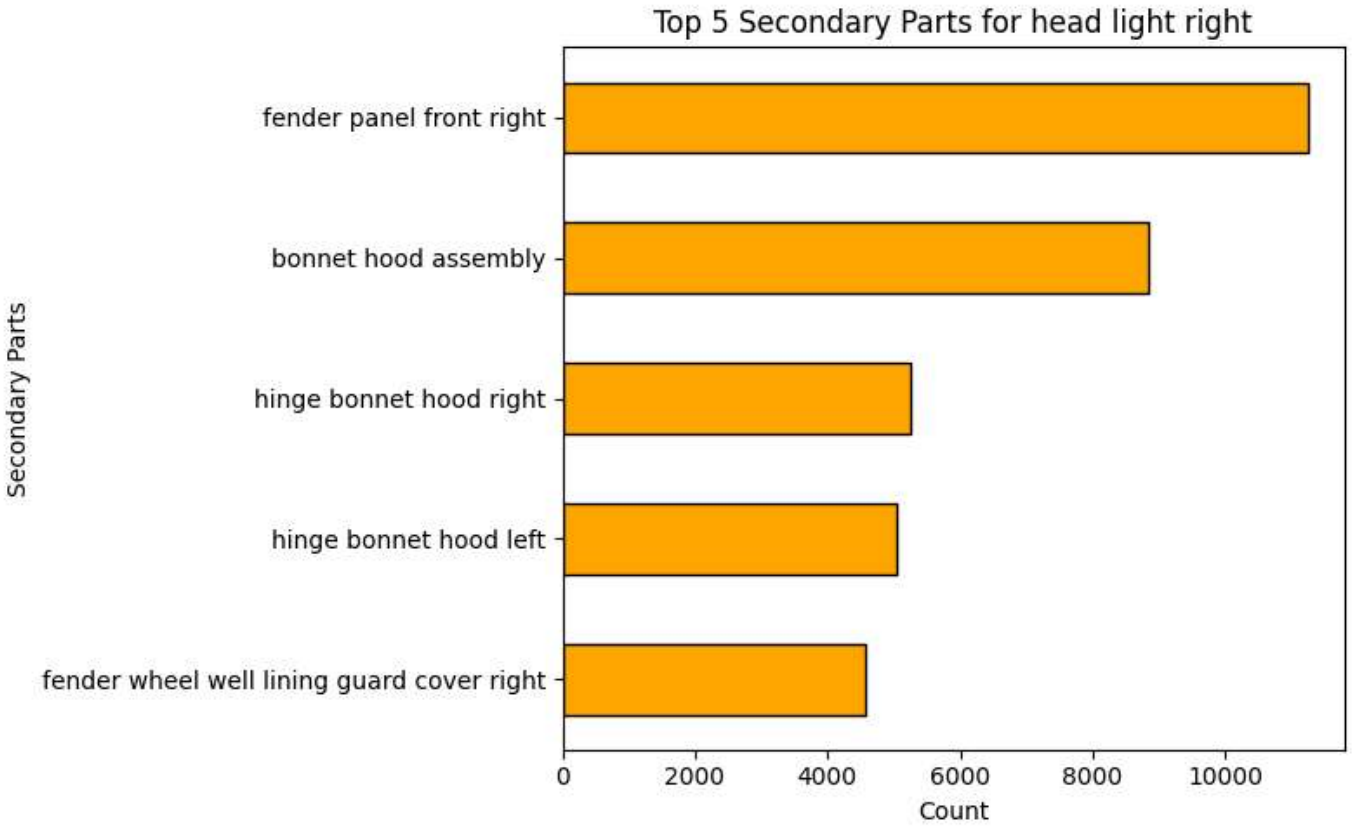



Top 5 Secondary Parts for bumper front assembly
fender panel front left
bonnet hood assembly
fender panel front right
ac condenser assembly
hinge bonnet hood left
Secondary Parts
0
2000
4000
6000
8000
10000
12000
Count


Top 5 Secondary Parts for bumper rear assembly
tail light right
tail light left
dicky trunk assembly
bonnet hood assembly
fender panel front right
Secondary Parts
0
1000
2000
3000
4000
5000
6000
Count


Top 5 Secondary Parts for grille radiator upper
bonnet hood assembly
ac condenser assembly
hinge bonnet hood left
hinge bonnet hood right
fender panel front right
Secondary Parts
0
2000
4000
6000
8000
10000
Count


Top 5 Secondary Parts for head light left
fender panel front left
bonnet hood assembly
hinge bonnet hood left
hinge bonnet hood right
ac condenser assembly
Secondary Parts
0
2000
4000
6000
8000
10000
12000
Count



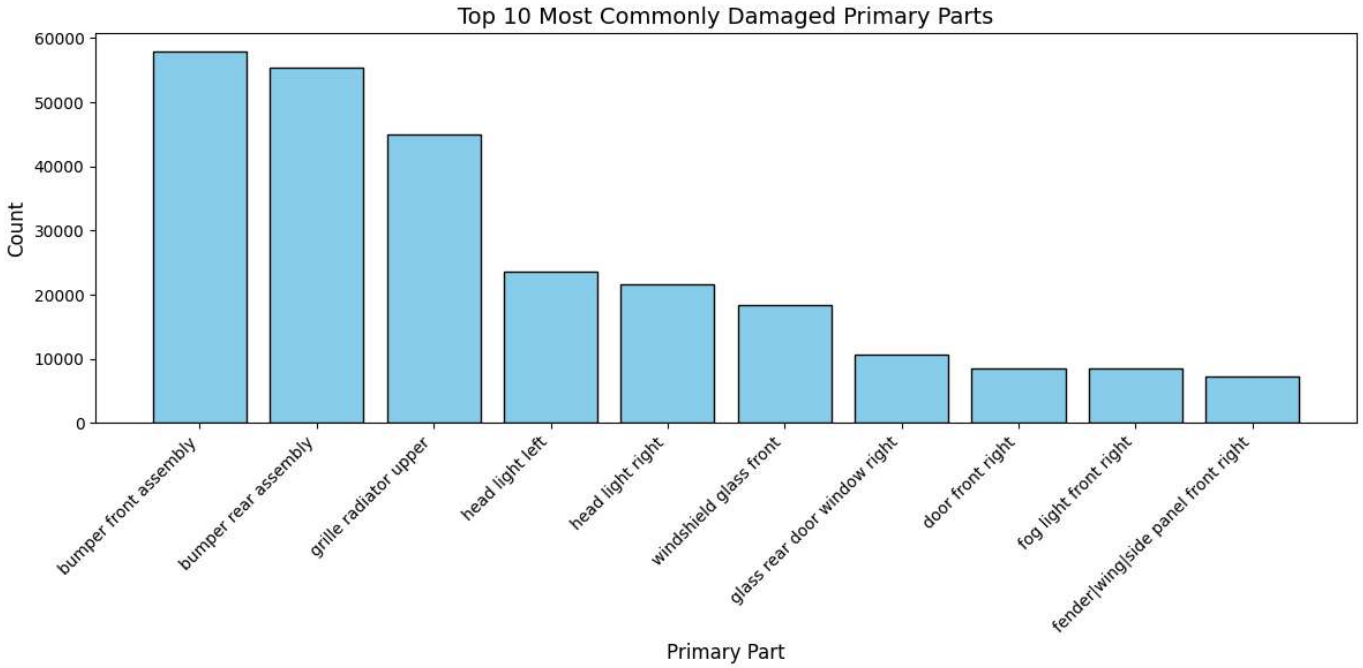
	primary_part	secondary_part
0	None	emblem logo rear middle
1	bumper rear assembly	None
2	windshield glass front	None
3	None	moulding front windshield
4	None	bonnet hood assembly

Most Commonly Damaged Primary Parts: on Survey dataset

In [130…

```
import matplotlib.pyplot as plt

primary_analysis = df['primary_part'].value_counts().reset_index()
primary_analysis.columns = ['Primary Part', 'Count']
total = primary_analysis['Count'].sum()
primary_analysis['Percentage'] = (primary_analysis['Count'] / total * 100).round(2)

# Print most commonly damaged primary parts
print("Most Commonly Damaged Primary Parts:")
print(primary_analysis.head(10))

# Visualization: Bar chart for top 10 most commonly damaged parts
plt.figure(figsize=(12, 6))
plt.bar(primary_analysis['Primary Part'][:10], primary_analysis['Count'][:10], color='skyblue
plt.title("Top 10 Most Commonly Damaged Primary Parts", fontsize=14)
plt.xlabel("Primary Part", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.tight_layout()
plt.show()
```

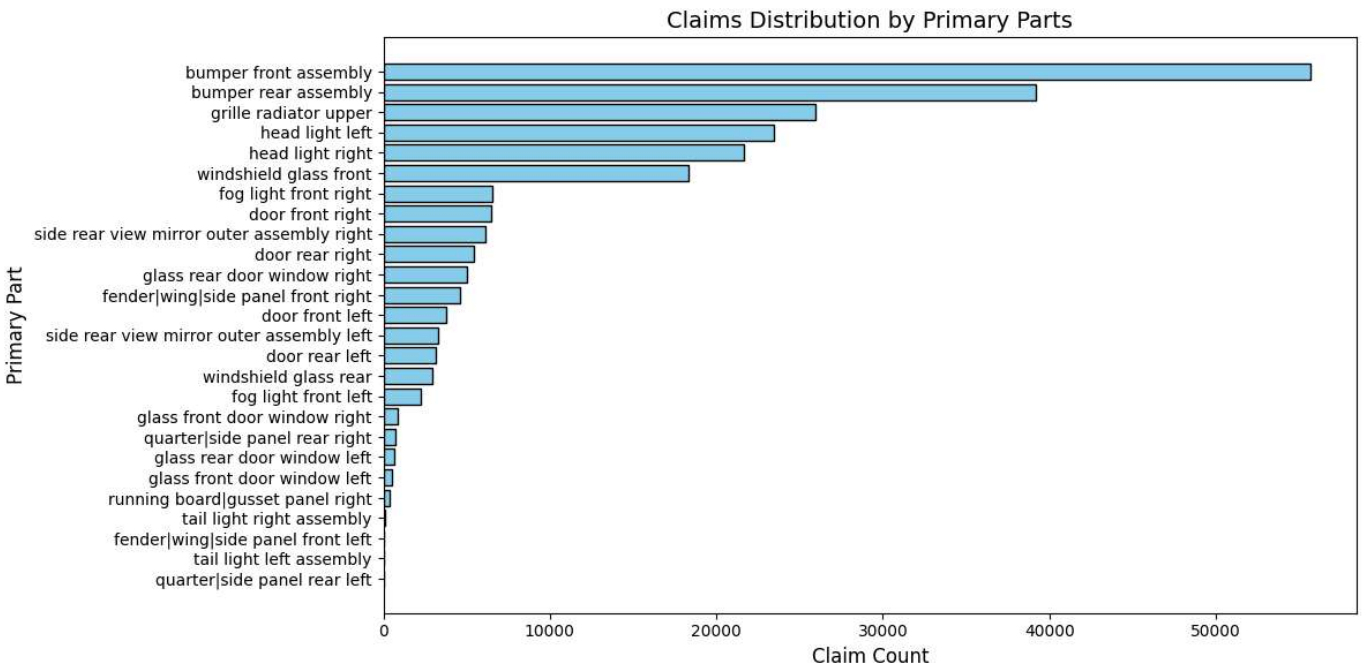
```
Most Commonly Damaged Primary Parts:
                        Primary Part  Count  Percentage
0                bumper front assembly  57853       20.07
1                 bumper rear assembly  55477       19.24
2                grille radiator upper  44949       15.59
3                      head light left  23575        8.18
4                     head light right  21675        7.52
5               windshield glass front  18359        6.37
6         glass rear door window right  10605        3.68
7                      door front right   8555        2.97
8                 fog light front right   8408        2.92
9  fender|wing|side panel front right   7227        2.51
```



Claims Distribution by Primary Parts on Survey data

In [129…

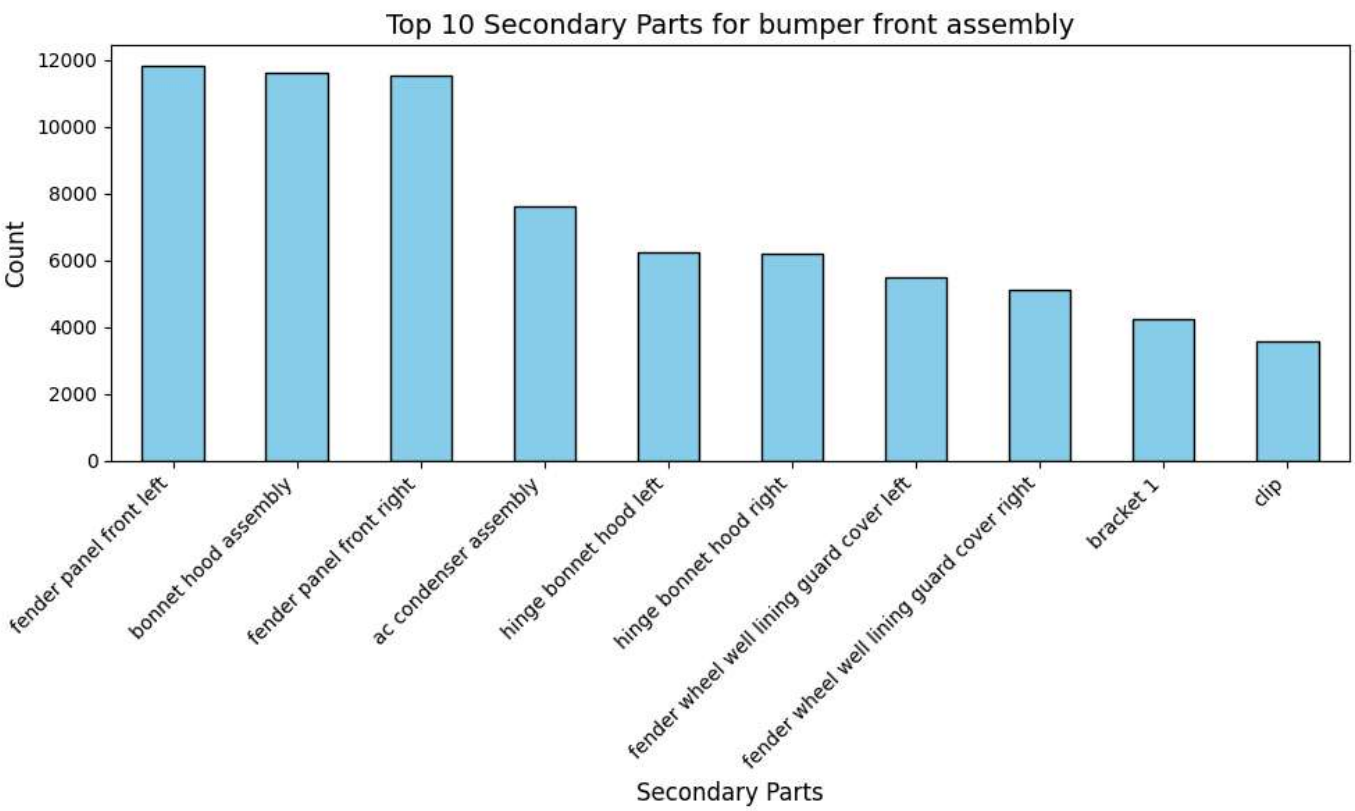
```python
import matplotlib.pyplot as plt

claims_distribution = df.groupby('primary_part')['CLAIMNO'].nunique().reset_index()
claims_distribution.columns = ['Primary Part', 'Claim Count']

# Sort the data by 'Claim Count' in descending order for better visualization
claims_distribution = claims_distribution.sort_values('Claim Count', ascending=False)

# Display the data as a bar chart
plt.figure(figsize=(12, 6))
plt.barh(claims_distribution['Primary Part'], claims_distribution['Claim Count'], color='skyb
plt.title("Claims Distribution by Primary Parts", fontsize=14)
plt.xlabel("Claim Count", fontsize=12)
plt.ylabel("Primary Part", fontsize=12)
plt.gca().invert_yaxis()  # Invert y-axis to show the largest values first
plt.tight_layout()
plt.show()
```

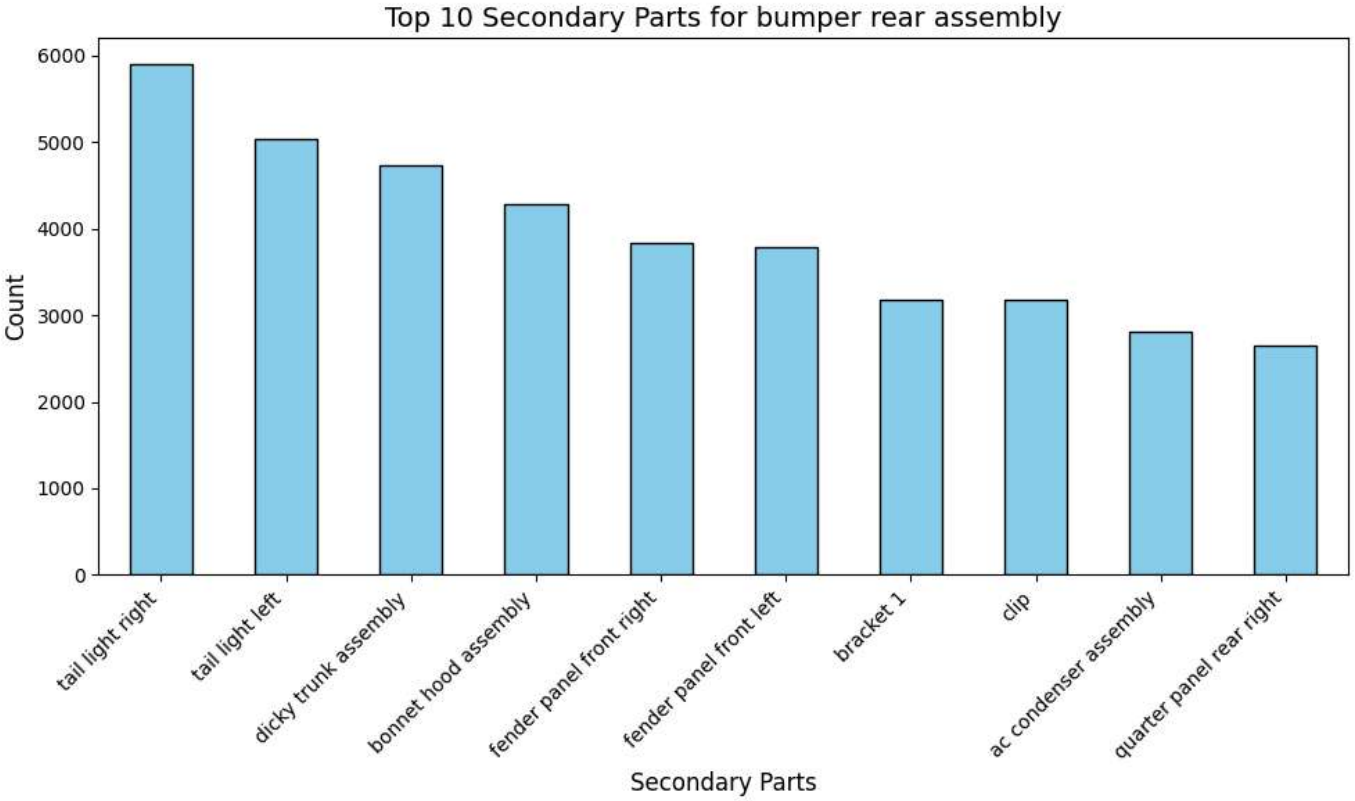


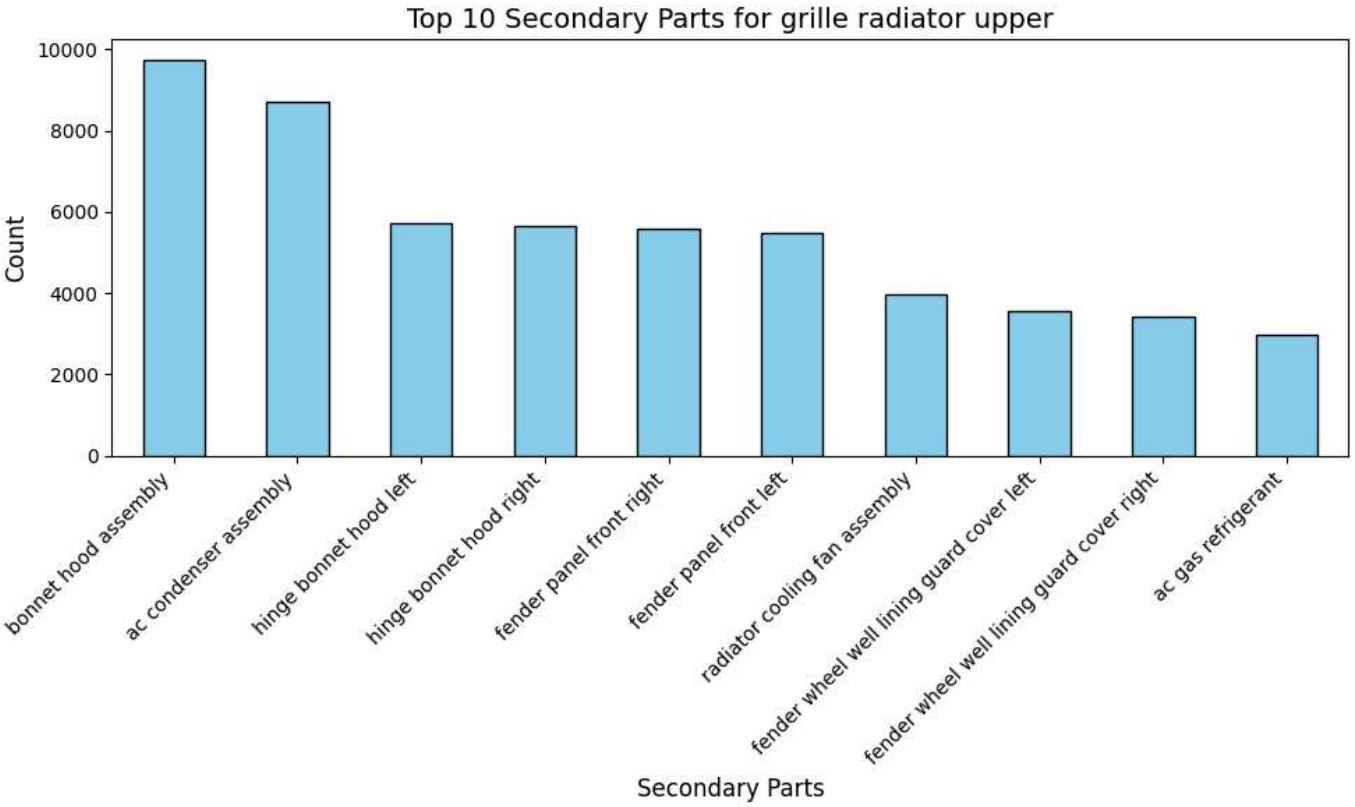
Top 10 Secondary Parts for primary dataset

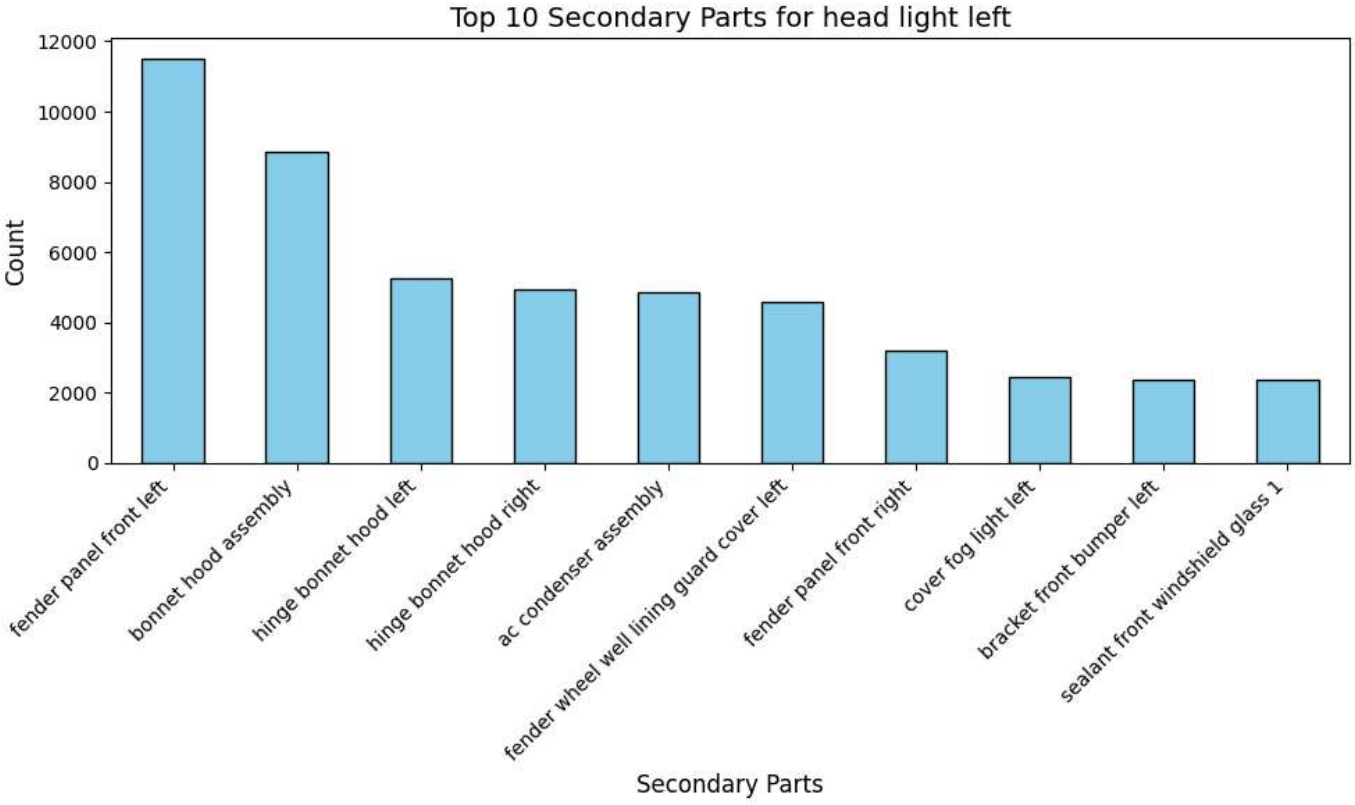
In [128…

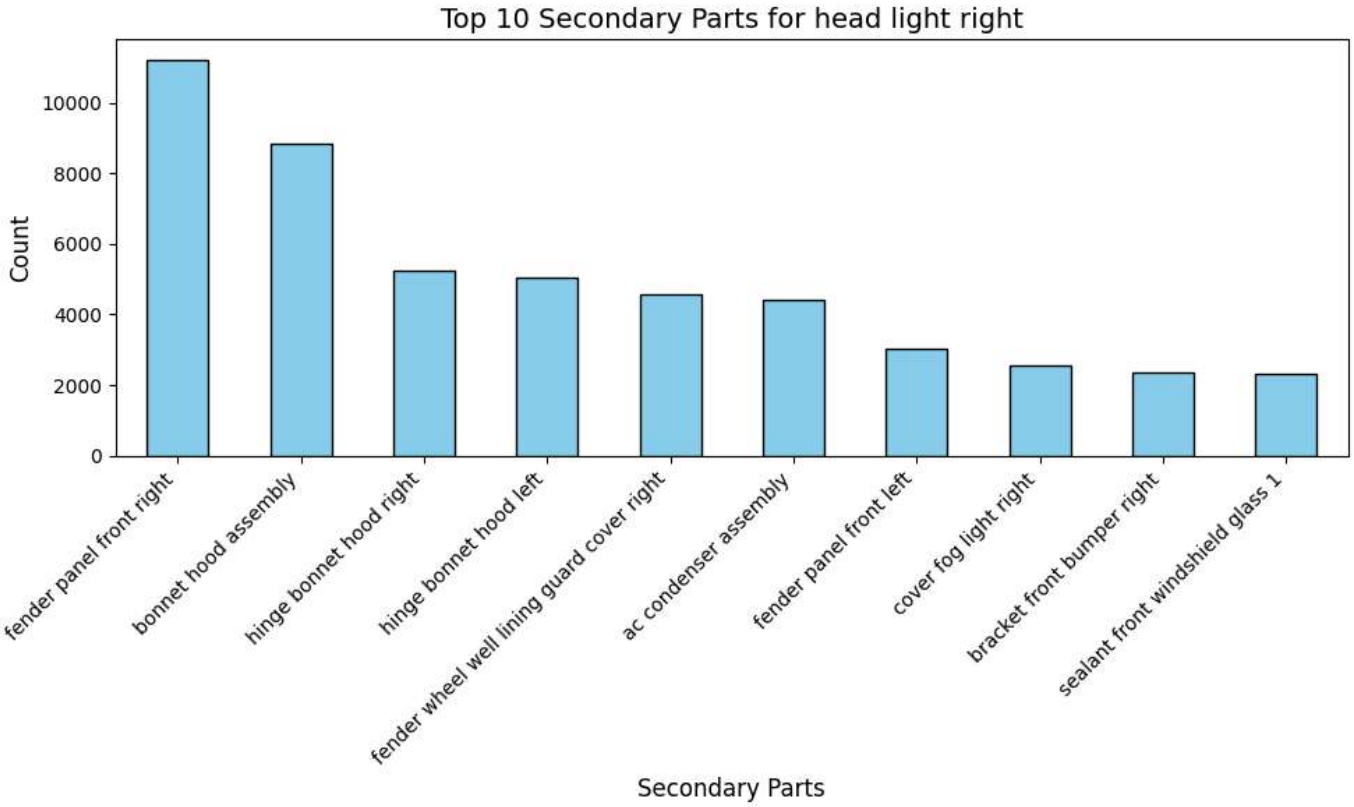
```python
import pandas as pd
import matplotlib.pyplot as plt

# Create claim-part matrix
claim_groups = df.groupby('CLAIMNO').agg({
    'primary_part': list,
    'secondary_part': list
}).reset_index()

# Analyze secondary associations
secondary_associations = {}

for primary in primary_analysis['Primary Part']:
    # Get claims containing this primary part
    relevant_claims = claim_groups[
        claim_groups['primary_part'].apply(lambda x: primary in x if x else False)
    ]

    # Collect all secondary parts from these claims
    all_secondary = [
        part for sublist in relevant_claims['secondary_part']
        for part in sublist if part
    ]

    # Count occurrences
    counter = pd.Series(all_secondary).value_counts().head(10)
    secondary_associations[primary] = counter

    # Plot top secondary associations for the primary part
    plt.figure(figsize=(10, 6))
    counter.sort_values(ascending=False).plot(kind='bar', color='skyblue', edgecolor='black')
    plt.title(f"Top 10 Secondary Parts for {primary}", fontsize=14)
    plt.ylabel("Count", fontsize=12)
    plt.xlabel("Secondary Parts", fontsize=12)
    plt.xticks(rotation=45, ha='right', fontsize=10)
    plt.tight_layout()
    plt.show()

# Print results
for primary, secondaries in secondary_associations.items():
    print(f"\nTop 10 Secondary Parts for {primary}:")
    print(secondaries)
```

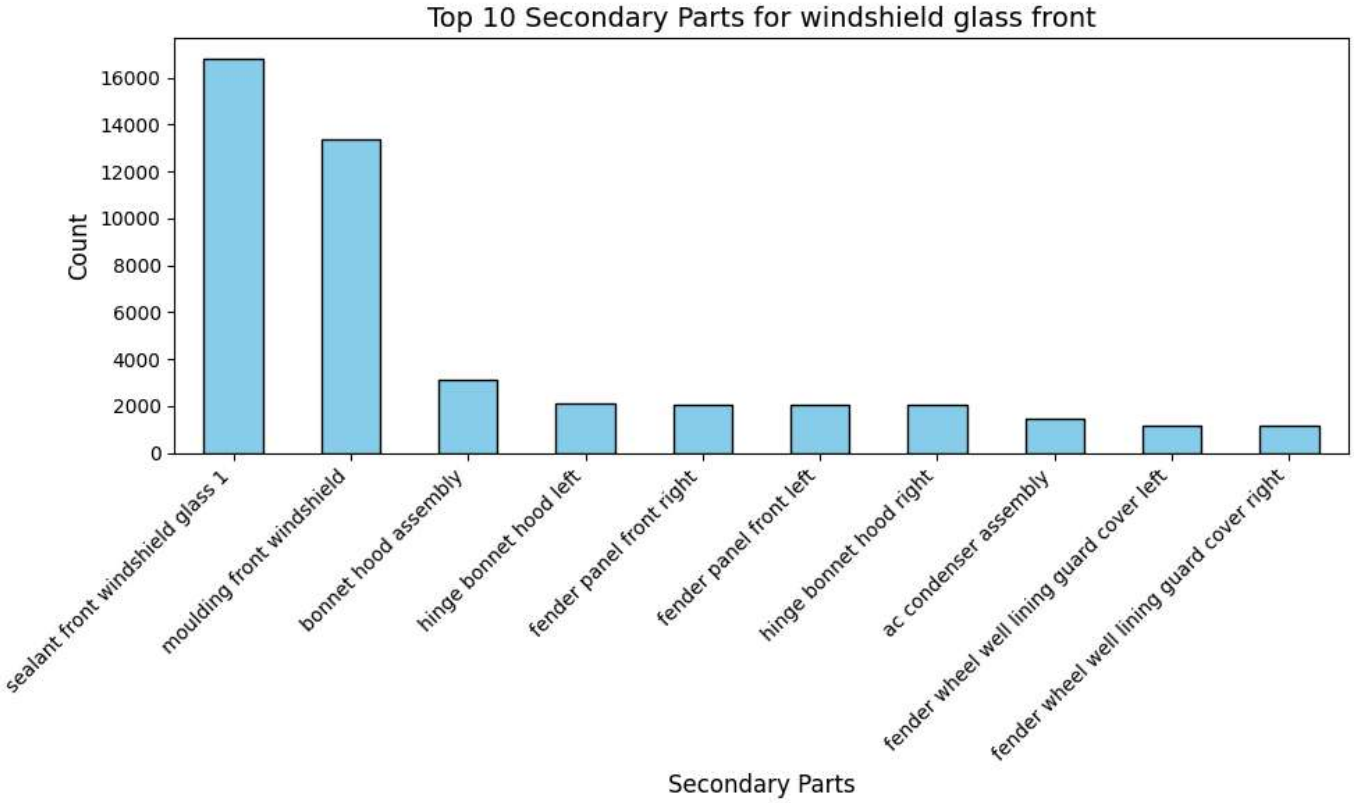

Top 10 Secondary Parts for bumper front assembly
Count
0
2000
4000
6000
8000
10000
12000
fender panel front left
bonnet hood assembly
fender panel front right
ac condenser assembly
hinge bonnet hood left
hinge bonnet hood right
fender wheel well lining guard cover left
fender wheel well lining guard cover right
bracket 1
clip
Secondary Parts

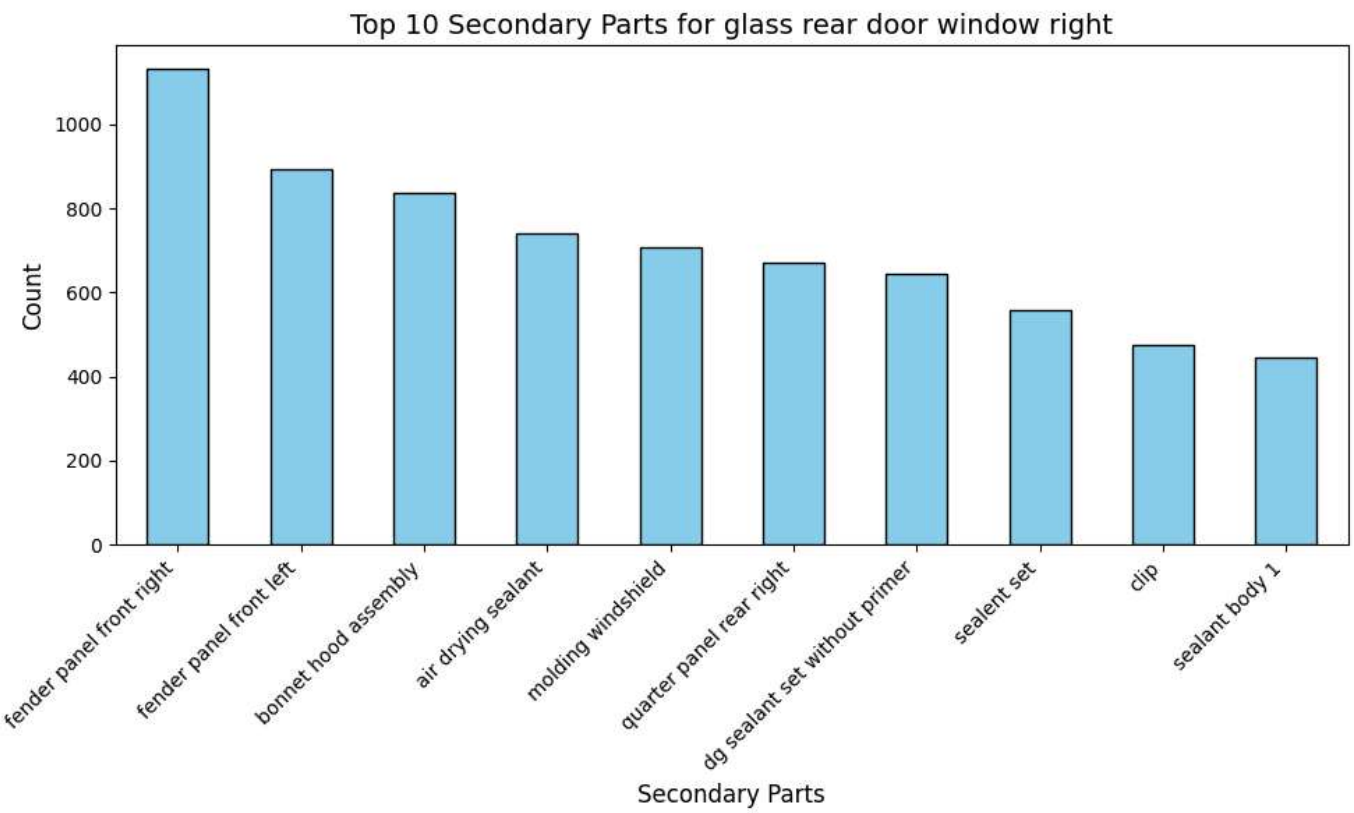

Top 10 Secondary Parts for bumper rear assembly
Count
0
1000
2000
3000
4000
5000
6000
tail light right
tail light left
dicky trunk assembly
bonnet hood assembly
fender panel front right
fender panel front left
bracket 1
clip
ac condenser assembly
quarter panel rear right
Secondary Parts

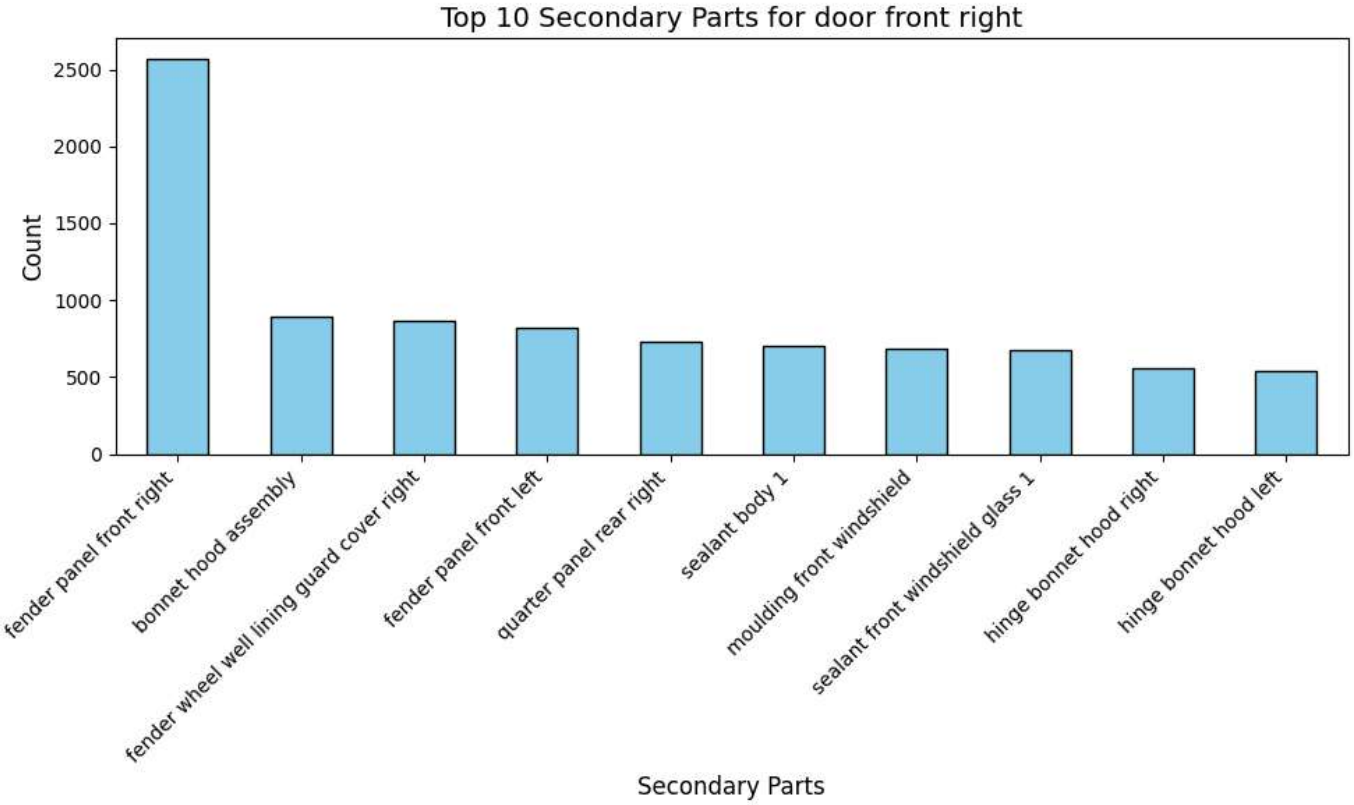

Top 10 Secondary Parts for grille radiator upper
Count
0
2000
4000
6000
8000
10000
bonnet hood assembly
ac condenser assembly
hinge bonnet hood left
hinge bonnet hood right
fender panel front right
fender panel front left
radiator cooling fan assembly
fender wheel well lining guard cover left
fender wheel well lining guard cover right
ac gas refrigerant
Secondary Parts

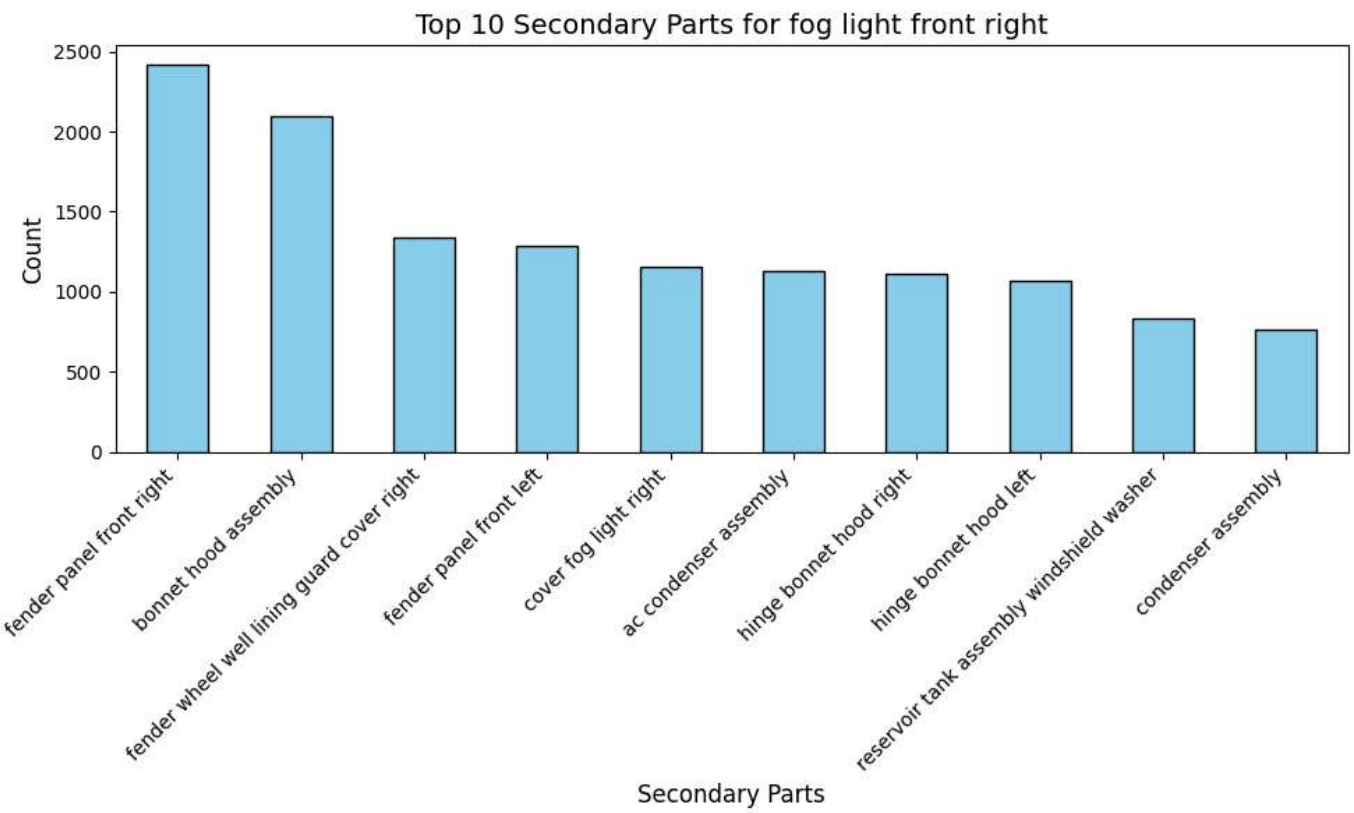

Top 10 Secondary Parts for head light left
Count
0
2000
4000
6000
8000
10000
12000
fender panel front left
bonnet hood assembly
hinge bonnet hood left
hinge bonnet hood right
ac condenser assembly
fender wheel well lining guard cover left
fender panel front right
cover fog light left
bracket front bumper left
sealant front windshield glass 1
Secondary Parts

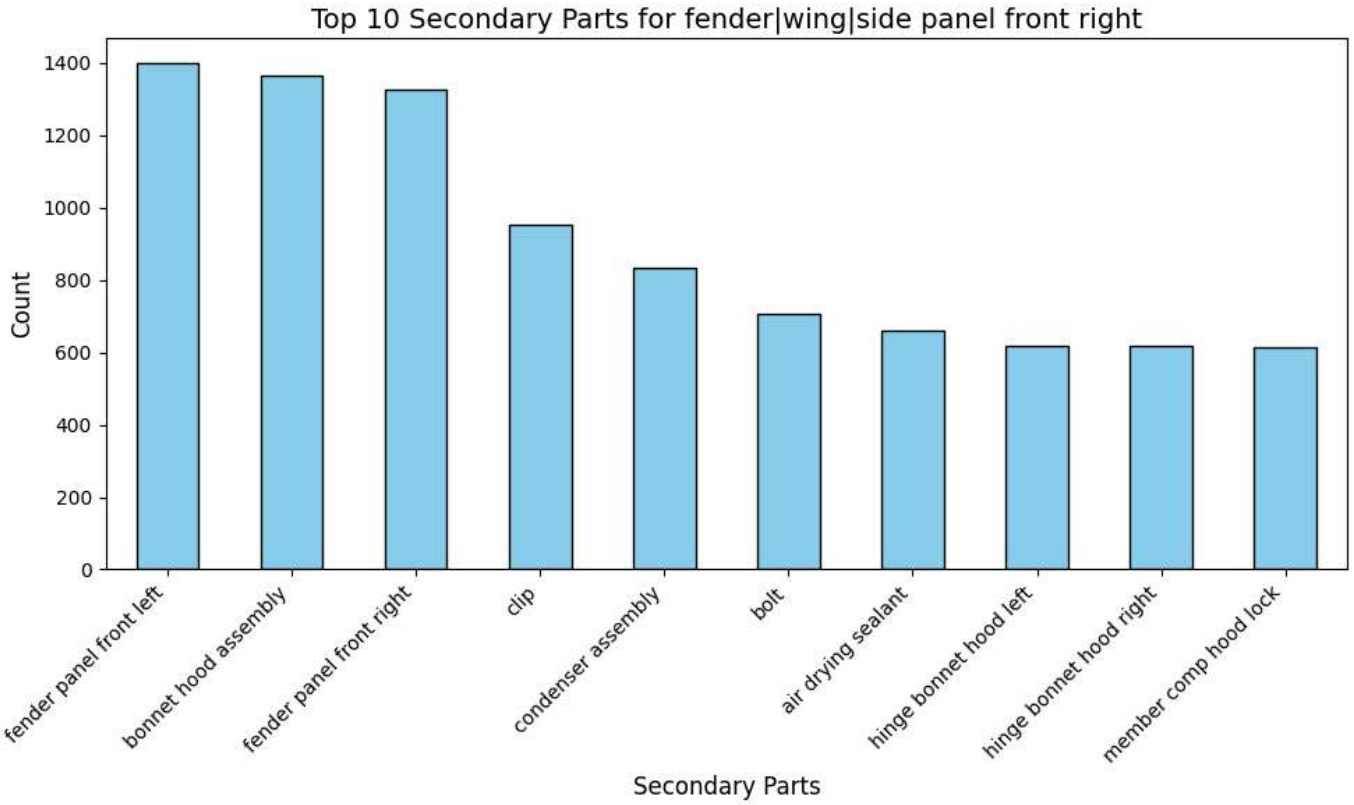

Top 10 Secondary Parts for head light right
Count
0
2000
4000
6000
8000
10000
fender panel front right
bonnet hood assembly
hinge bonnet hood right
hinge bonnet hood left
fender wheel well lining guard cover right
ac condenser assembly
fender panel front left
cover fog light right
bracket front bumper right
sealant front windshield glass 1
Secondary Parts

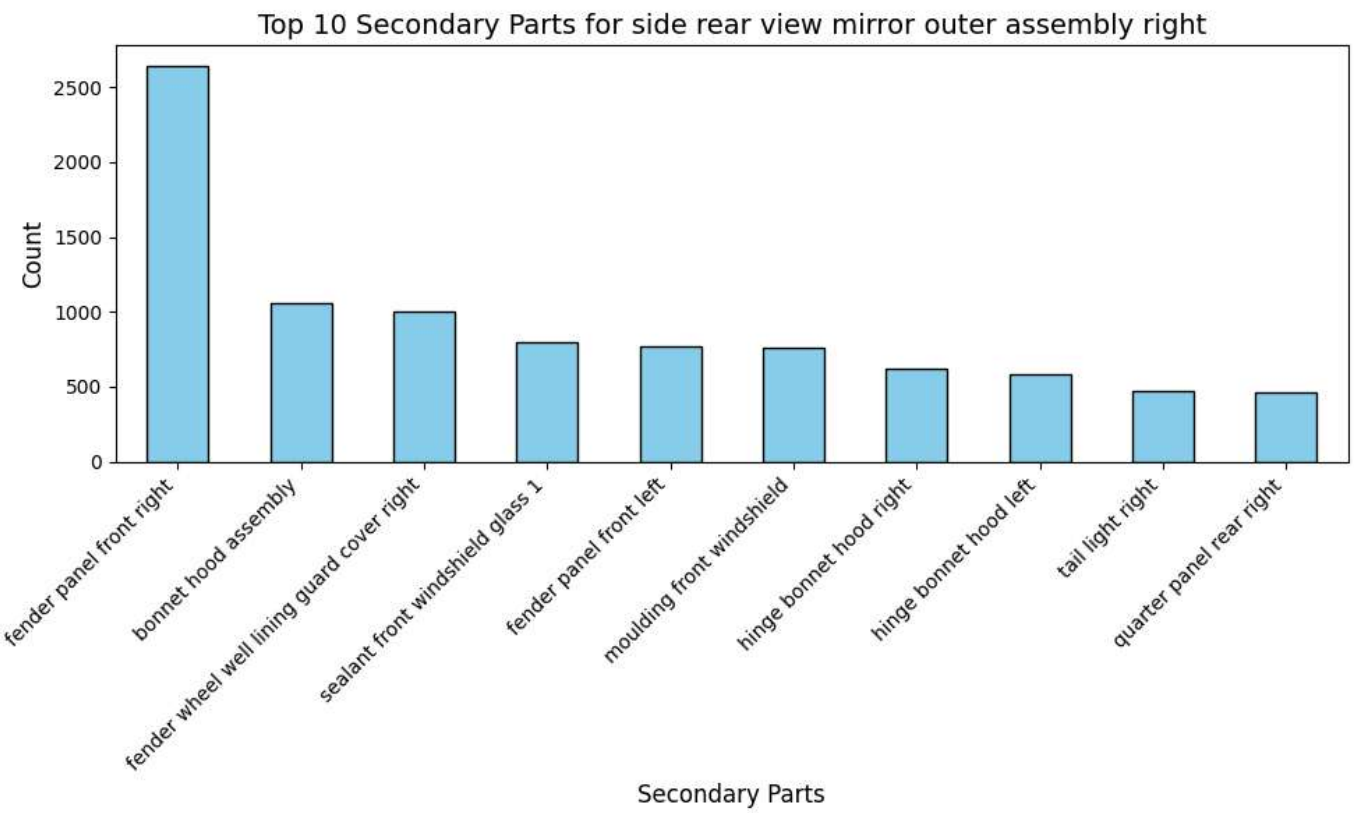

Top 10 Secondary Parts for windshield glass front
Count
0
2000
4000
6000
8000
10000
12000
14000
16000
sealant front windshield glass 1
moulding front windshield
bonnet hood assembly
hinge bonnet hood left
fender panel front right
fender panel front left
hinge bonnet hood right
ac condenser assembly
fender wheel well lining guard cover left
fender wheel well lining guard cover right
Secondary Parts

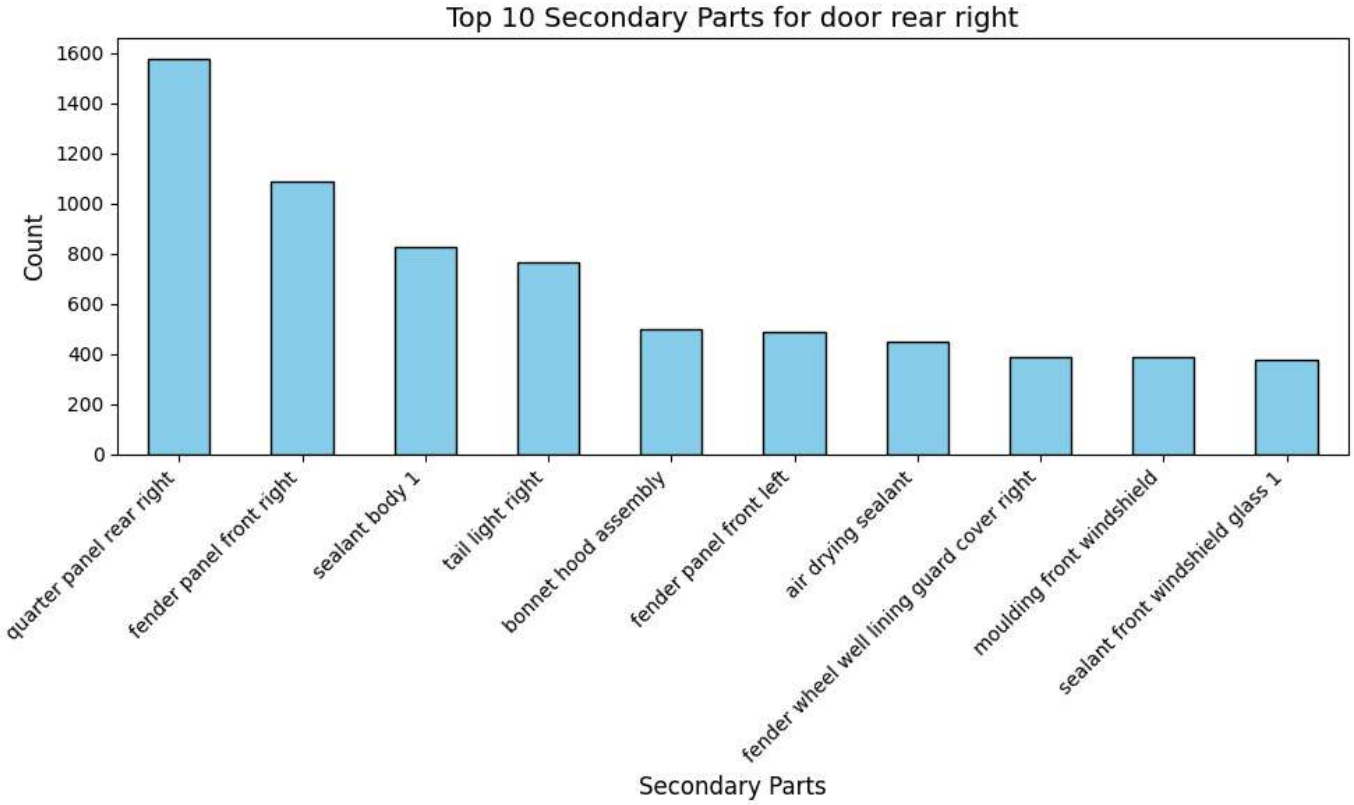

Top 10 Secondary Parts for glass rear door window right
Count
0
200
400
600
800
1000
fender panel front right
fender panel front left
bonnet hood assembly
air drying sealant
molding windshield
quarter panel rear right
dg sealant set without primer
sealant set
clip
sealant body 1
Secondary Parts

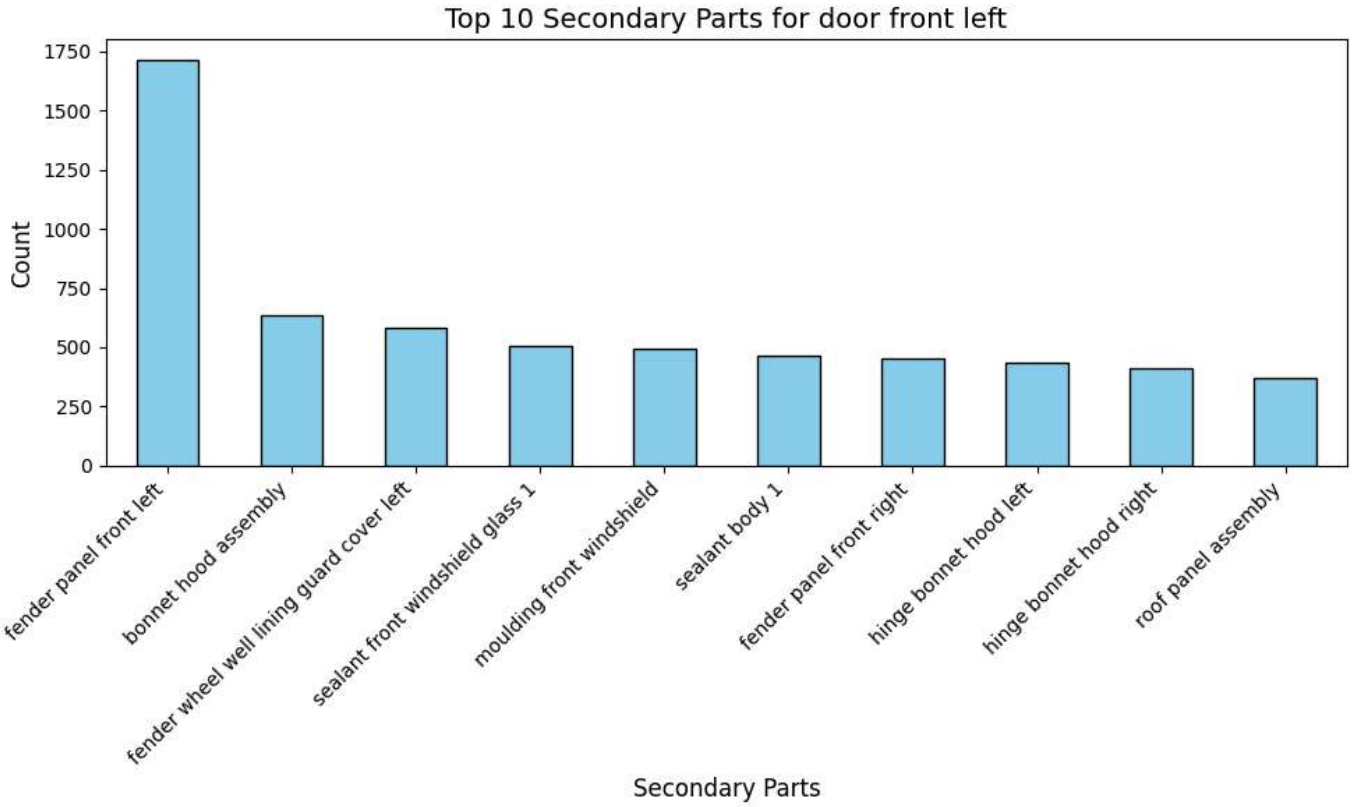

Top 10 Secondary Parts for door front right
Count
0
500
1000
1500
2000
2500
fender panel front right
bonnet hood assembly
fender wheel well lining guard cover right
fender panel front left
quarter panel rear right
sealant body 1
moulding front windshield
sealant front windshield glass 1
hinge bonnet hood right
hinge bonnet hood left
Secondary Parts

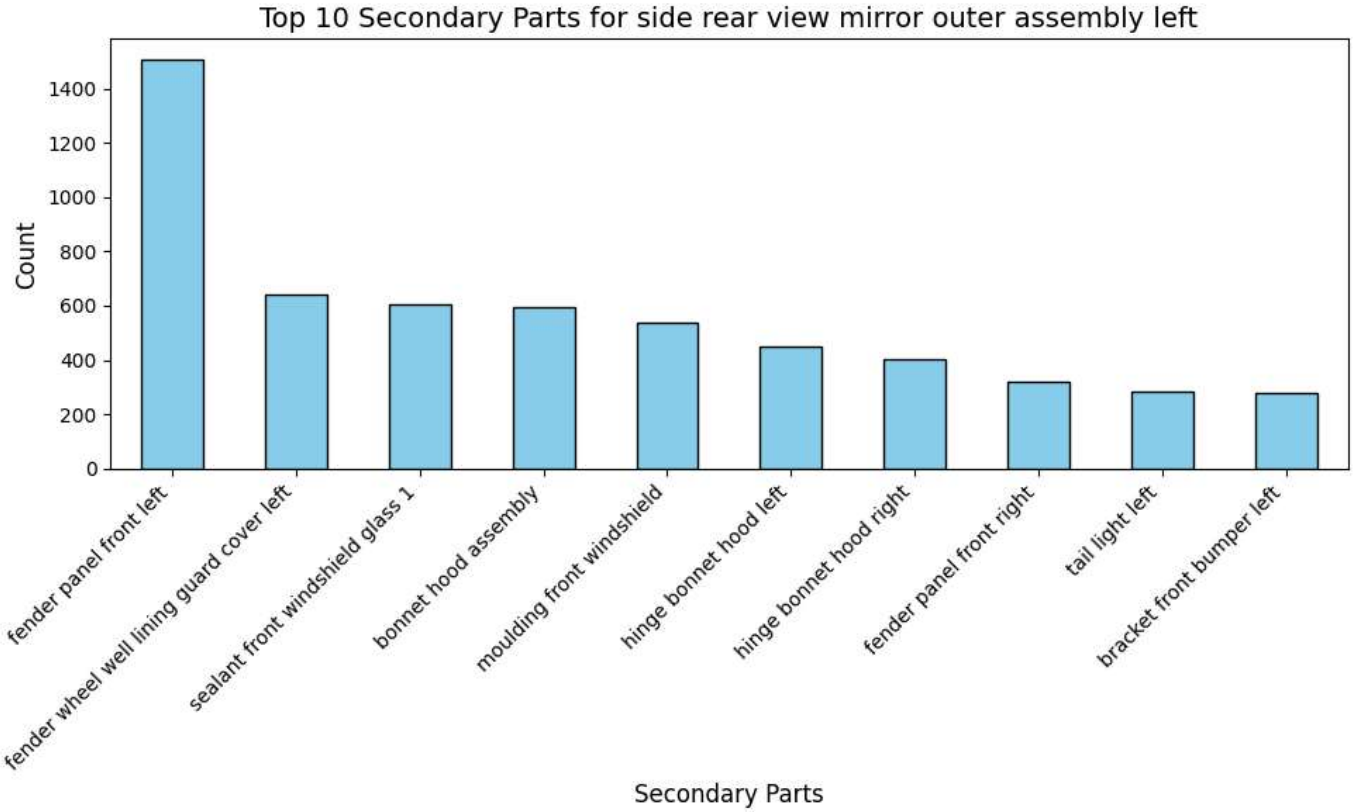

Top 10 Secondary Parts for fog light front right
Count
2500
2000
1500
1000
500
0
fender panel front right
bonnet hood assembly
fender wheel well lining guard cover right
fender panel front left
cover fog light right
ac condenser assembly
hinge bonnet hood right
hinge bonnet hood left
reservoir tank assembly windshield washer
condenser assembly
Secondary Parts

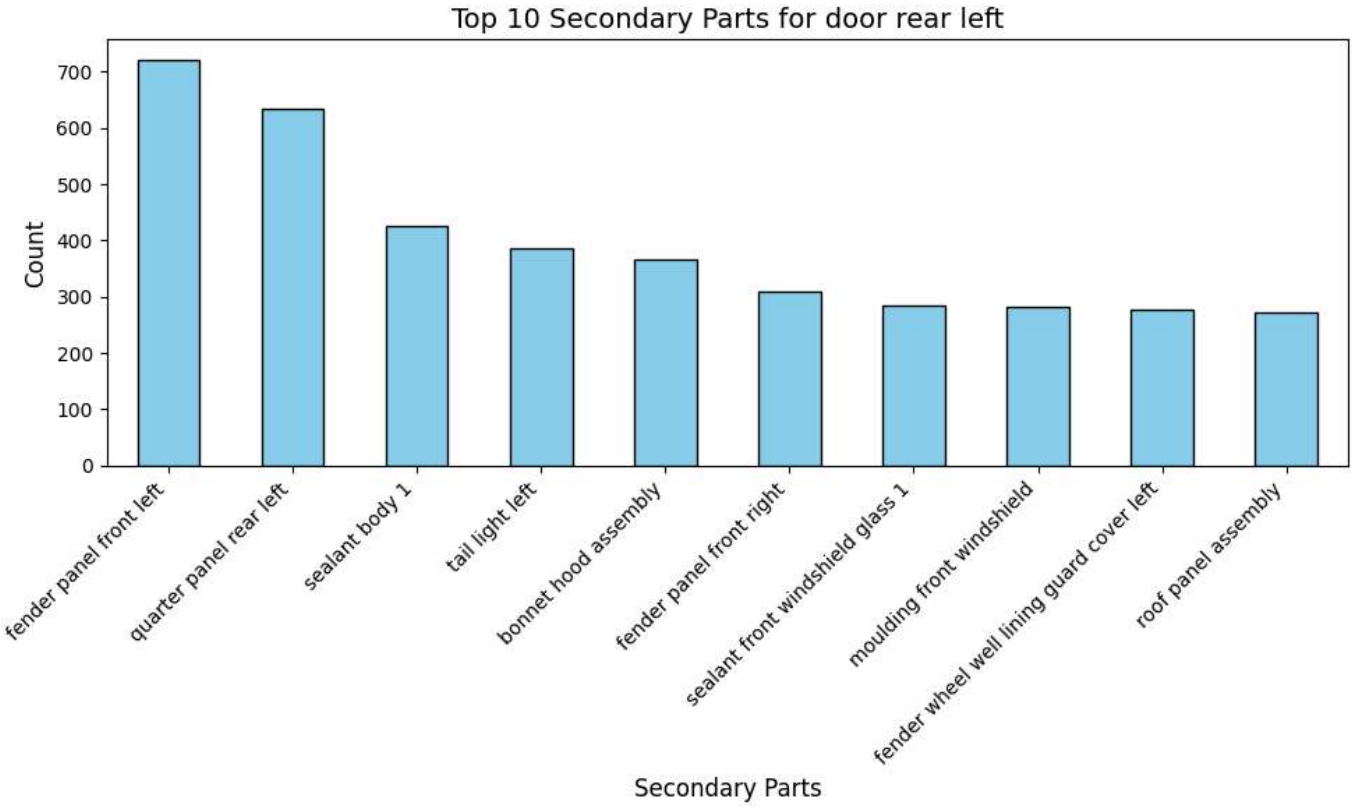

Top 10 Secondary Parts for fender|wing|side panel front right
Count
1400
1200
1000
800
600
400
200
0
fender panel front left
bonnet hood assembly
fender panel front right
clip
condenser assembly
bolt
air drying sealant
hinge bonnet hood left
hinge bonnet hood right
member comp hood lock
Secondary Parts

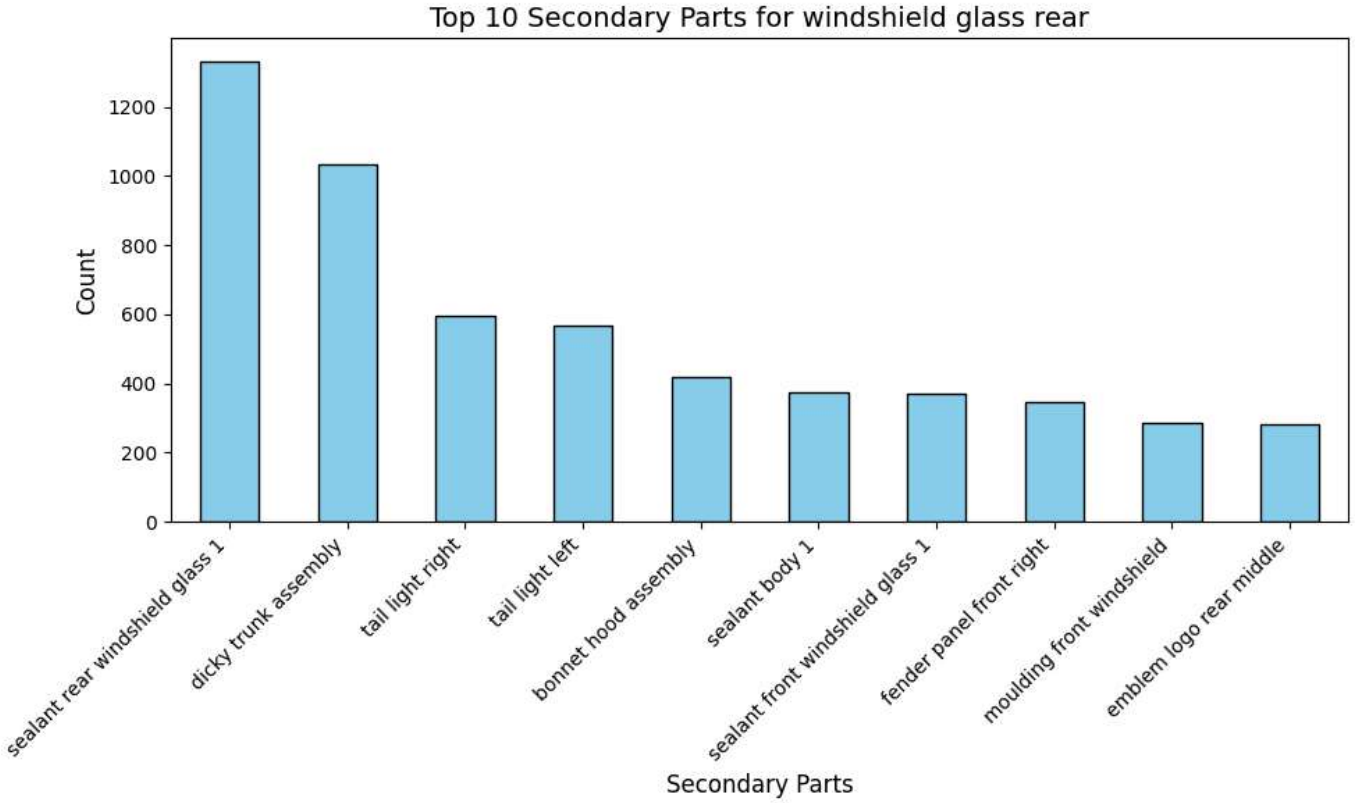

Top 10 Secondary Parts for side rear view mirror outer assembly right
Count
2500
2000
1500
1000
500
0
fender panel front right
bonnet hood assembly
fender wheel well lining guard cover right
sealant front windshield glass 1
fender panel front left
moulding front windshield
hinge bonnet hood right
hinge bonnet hood left
tail light right
quarter panel rear right
Secondary Parts

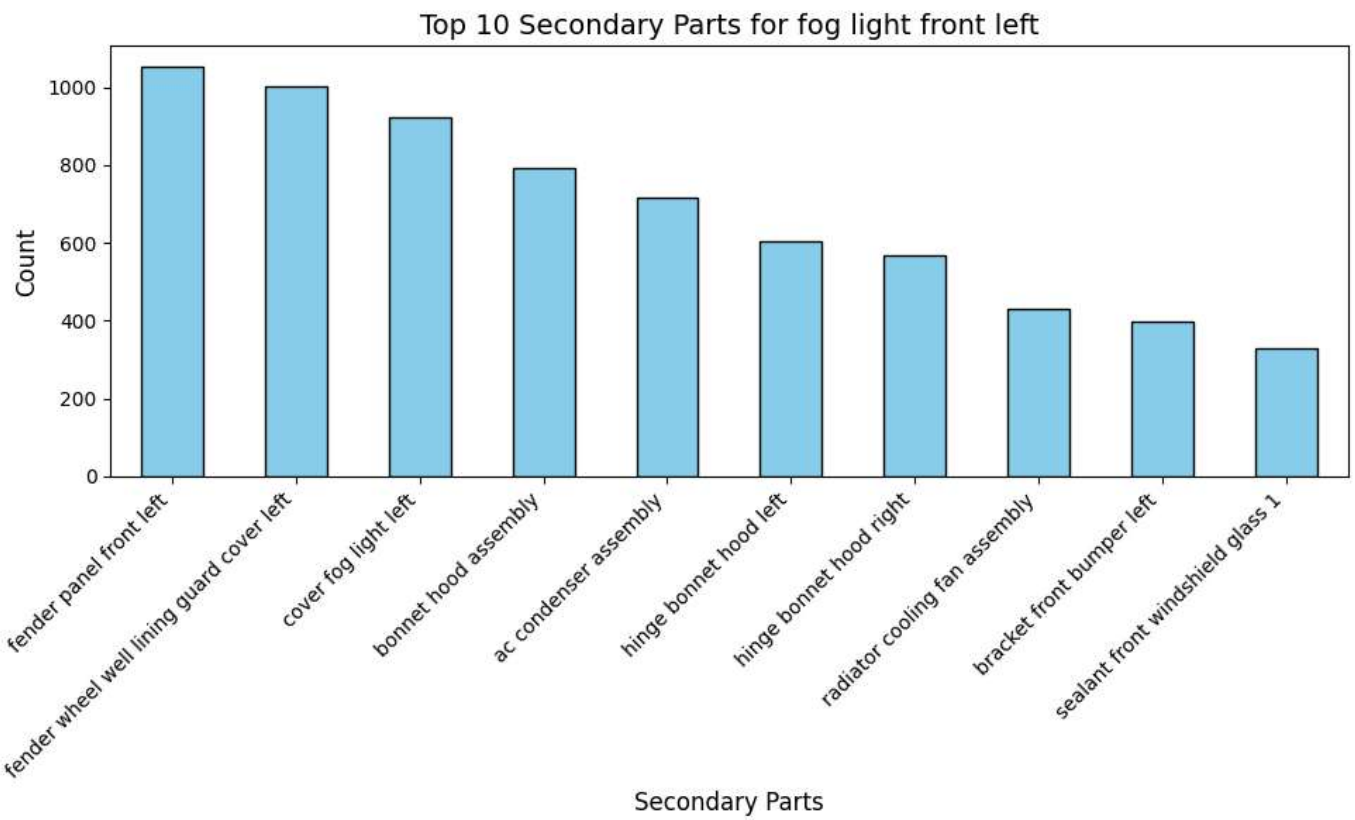

Top 10 Secondary Parts for door rear right
Count
1600
1400
1200
1000
800
600
400
200
0
quarter panel rear right
fender panel front right
sealant body 1
tail light right
bonnet hood assembly
fender panel front left
air drying sealant
fender wheel well lining guard cover right
moulding front windshield
sealant front windshield glass 1
Secondary Parts

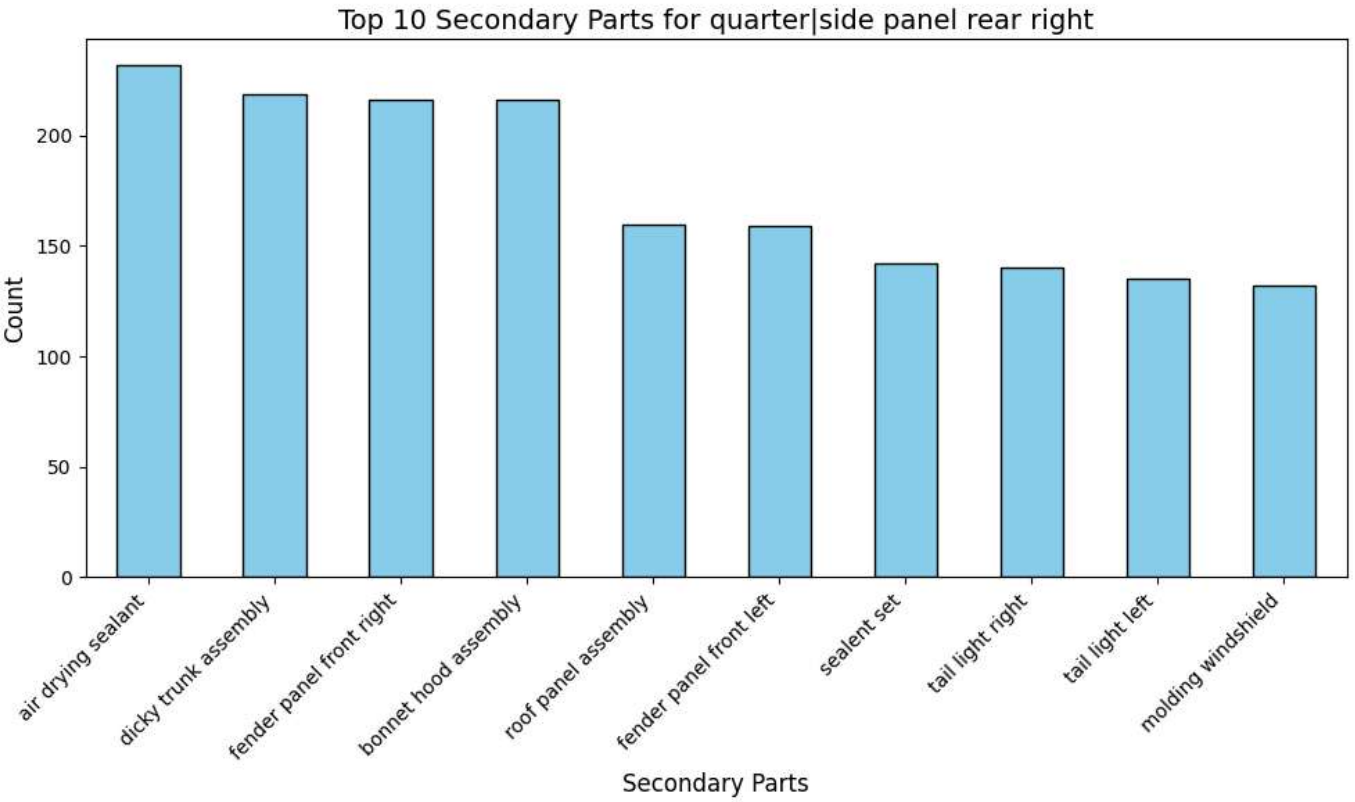

Top 10 Secondary Parts for door front left
Count
1750
1500
1250
1000
750
500
250
0
fender panel front left
bonnet hood assembly
fender wheel well lining guard cover left
sealant front windshield glass 1
moulding front windshield
sealant body 1
fender panel front right
hinge bonnet hood left
hinge bonnet hood right
roof panel assembly
Secondary Parts

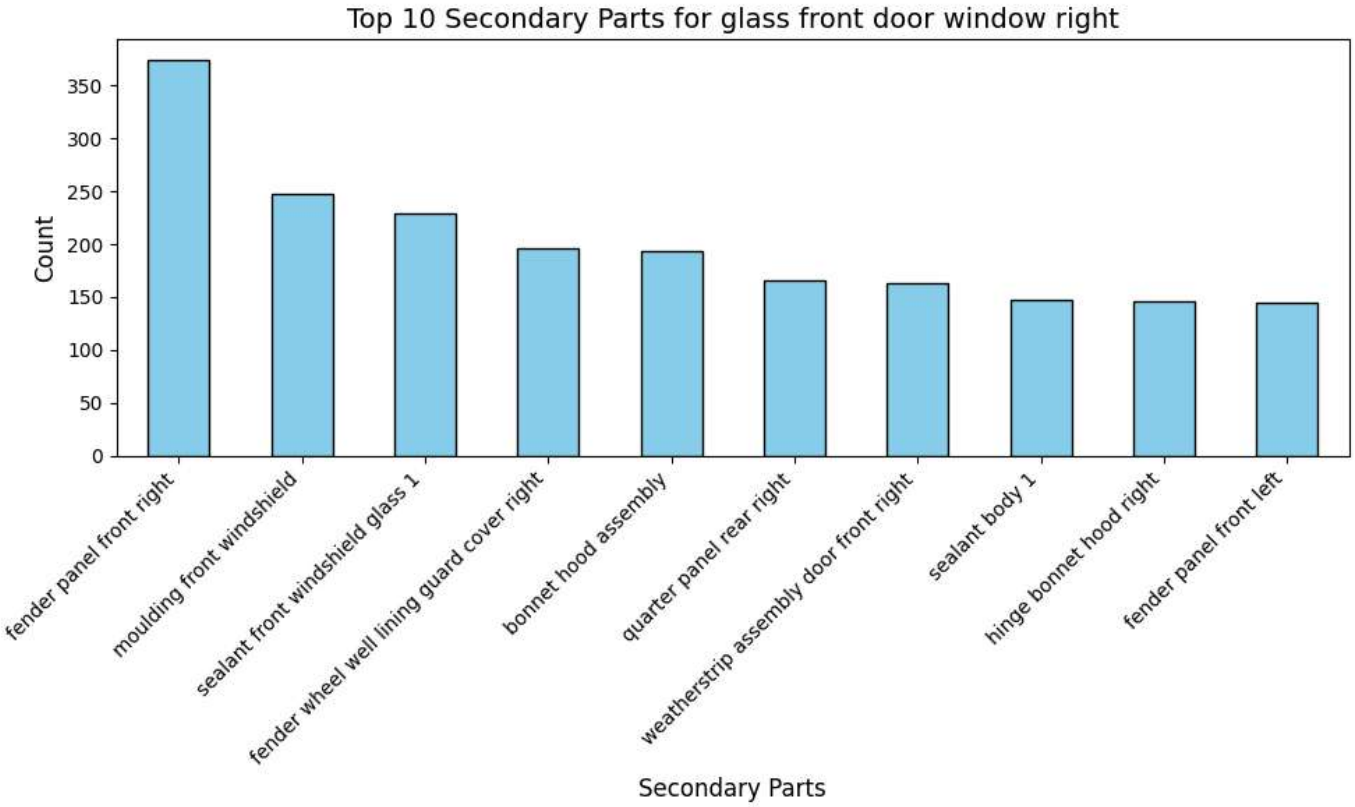

Top 10 Secondary Parts for side rear view mirror outer assembly left
Count
1400
1200
1000
800
600
400
200
0
fender panel front left
fender wheel well lining guard cover left
sealant front windshield glass 1
bonnet hood assembly
moulding front windshield
hinge bonnet hood left
hinge bonnet hood right
fender panel front right
tail light left
bracket front bumper left
Secondary Parts

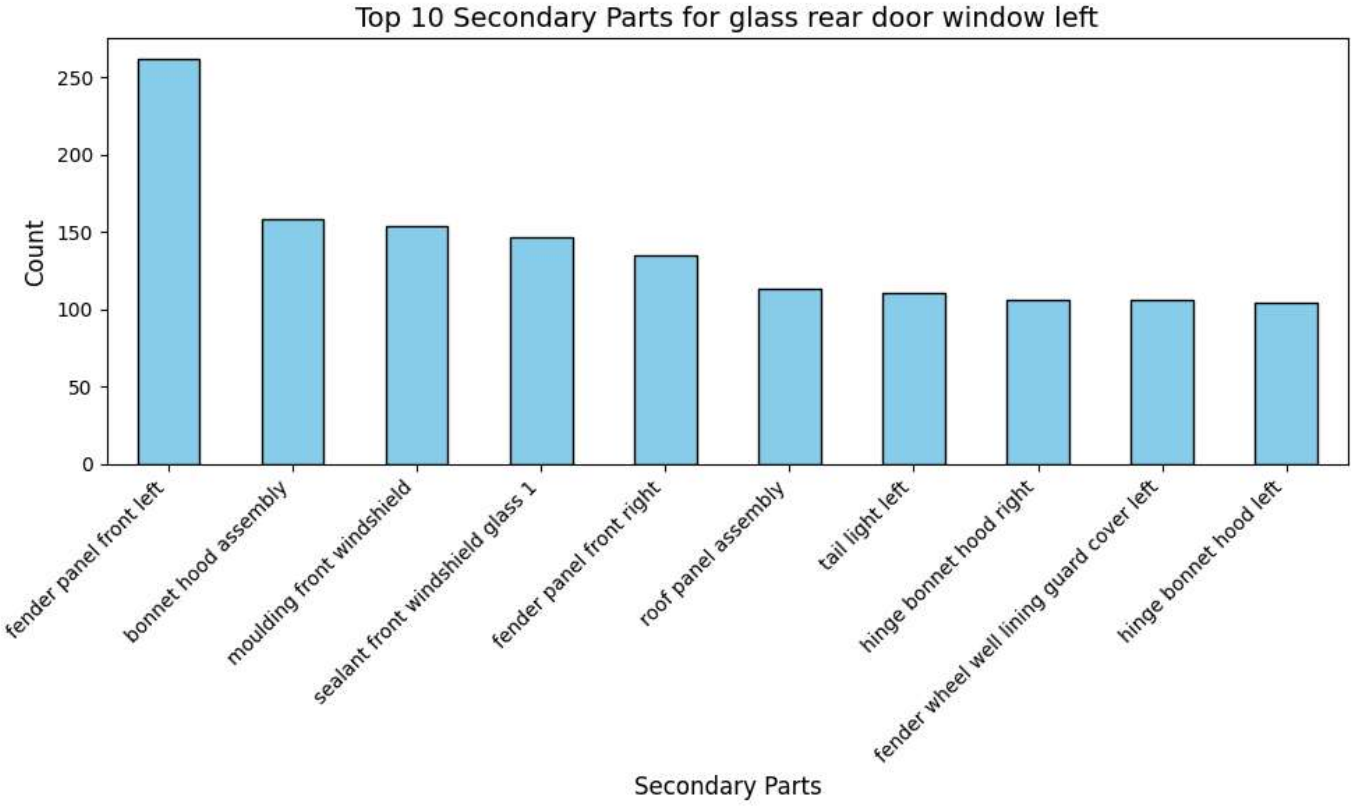

Top 10 Secondary Parts for door rear left
Count
0
100
200
300
400
500
600
700
fender panel front left
quarter panel rear left
sealant body 1
tail light left
bonnet hood assembly
fender panel front right
sealant front windshield glass 1
moulding front windshield
fender wheel well lining guard cover left
roof panel assembly
Secondary Parts

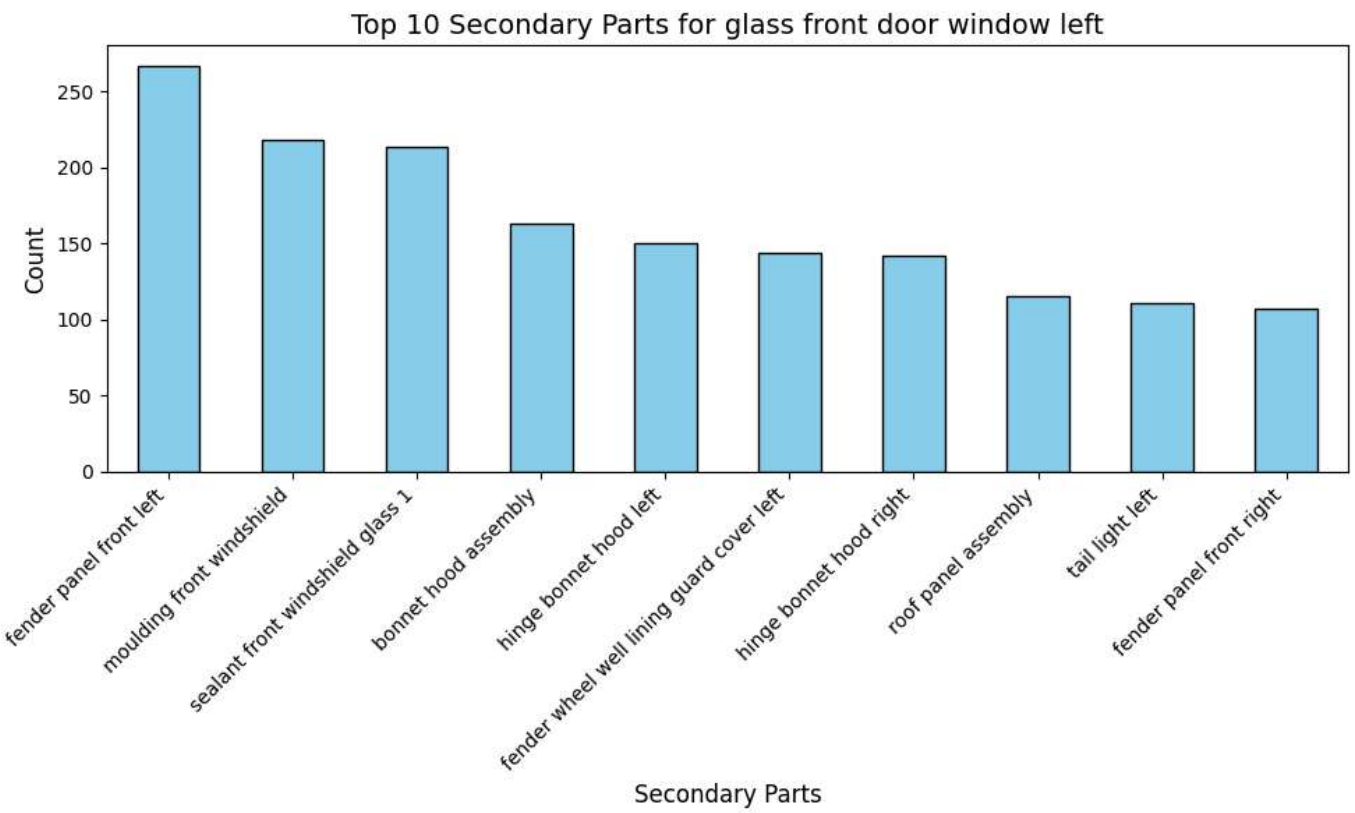

Top 10 Secondary Parts for windshield glass rear
Count
0
200
400
600
800
1000
1200
sealant rear windshield glass 1
dicky trunk assembly
tail light right
tail light left
bonnet hood assembly
sealant body 1
sealant front windshield glass 1
fender panel front right
moulding front windshield
emblem logo rear middle
Secondary Parts

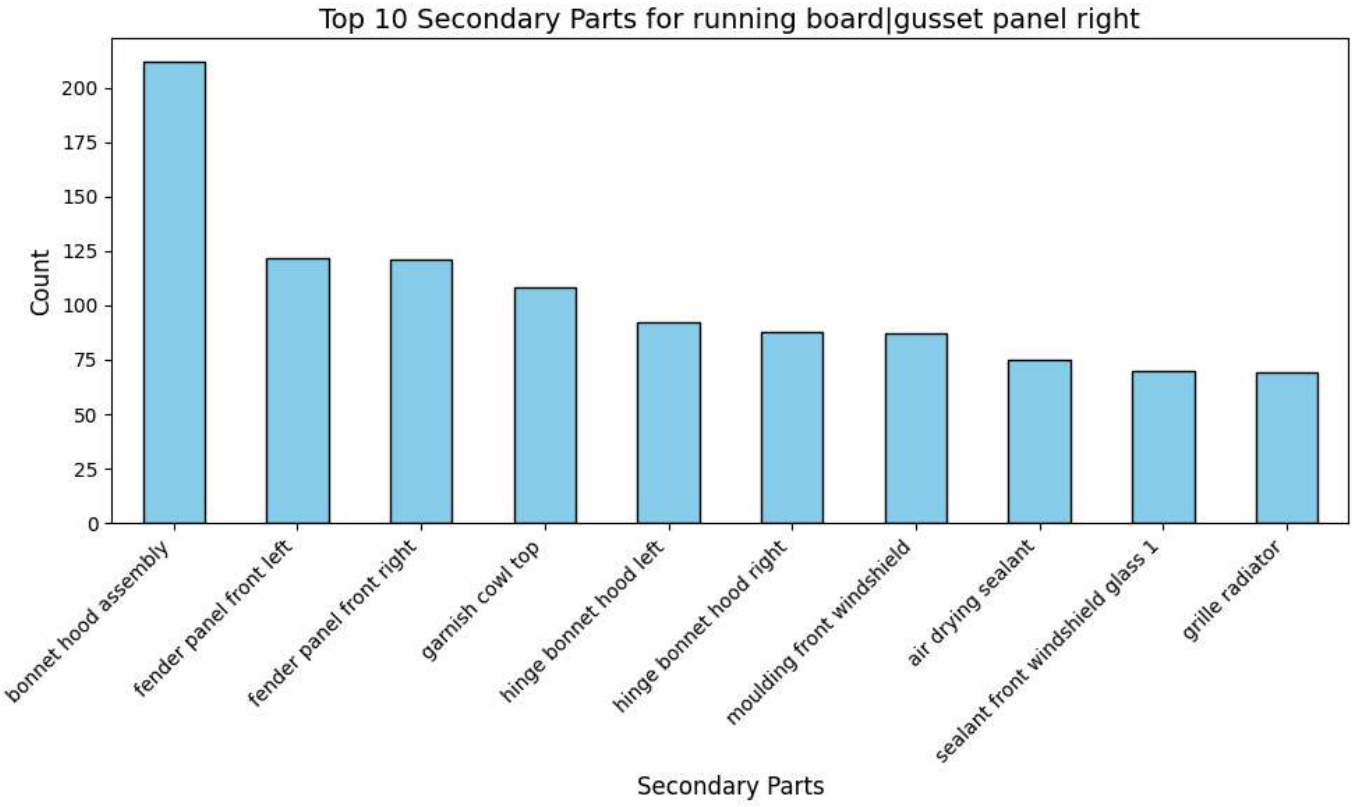

Top 10 Secondary Parts for fog light front left
Count
0
200
400
600
800
1000
fender panel front left
fender wheel well lining guard cover left
cover fog light left
bonnet hood assembly
ac condenser assembly
hinge bonnet hood left
hinge bonnet hood right
radiator cooling fan assembly
bracket front bumper left
sealant front windshield glass 1
Secondary Parts

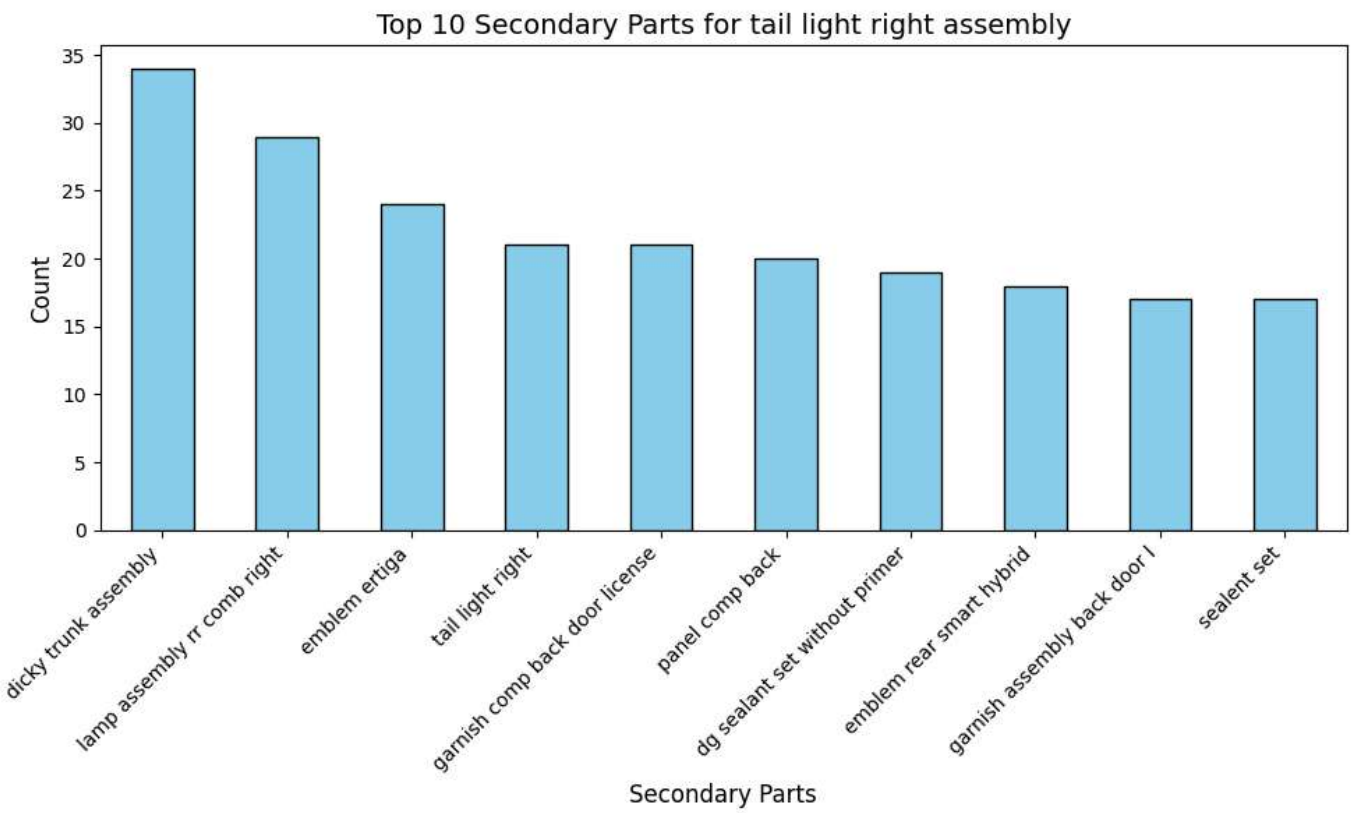

Top 10 Secondary Parts for quarter|side panel rear right
Count
0
50
100
150
200
air drying sealant
dicky trunk assembly
fender panel front right
bonnet hood assembly
roof panel assembly
fender panel front left
sealent set
tail light right
tail light left
molding windshield
Secondary Parts

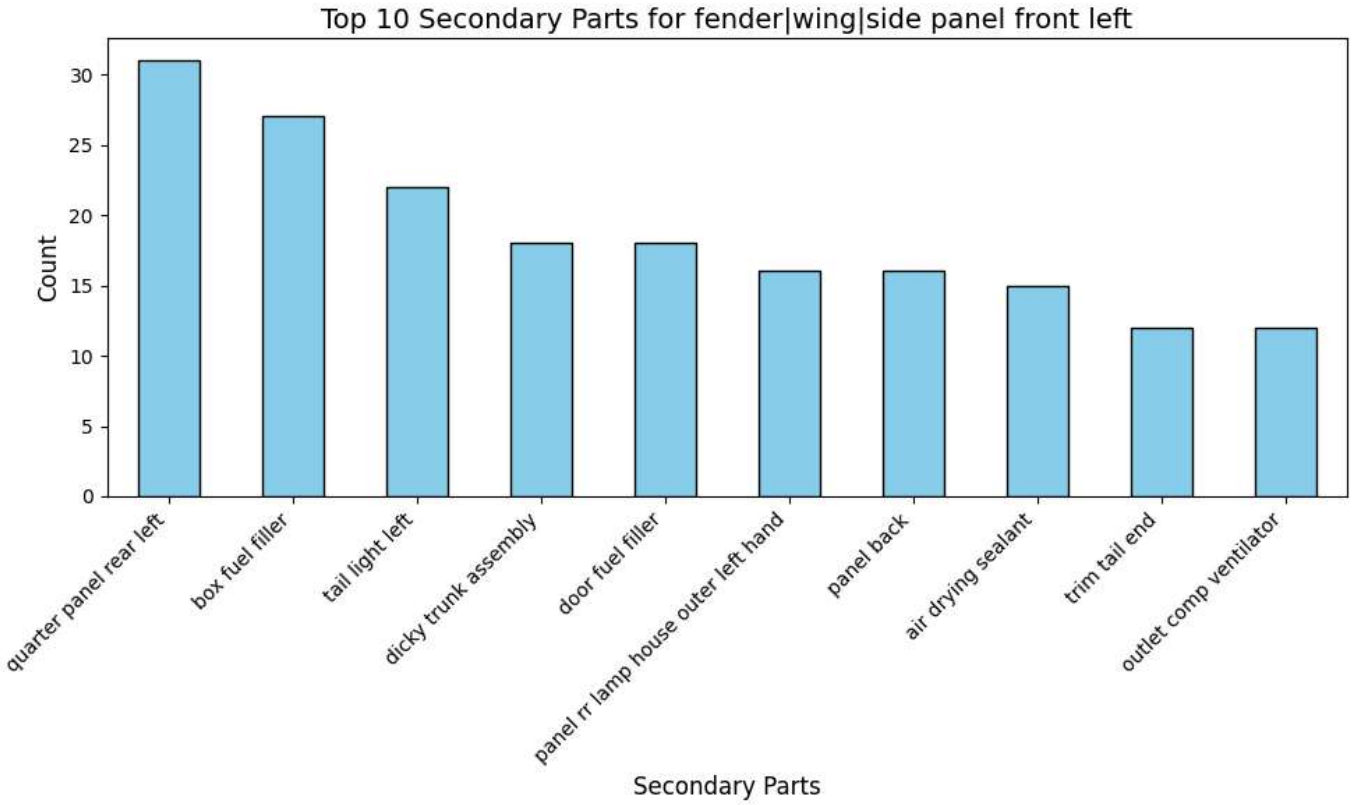

Top 10 Secondary Parts for glass front door window right
Count
0
50
100
150
200
250
300
350
fender panel front right
moulding front windshield
sealant front windshield glass 1
fender wheel well lining guard cover right
bonnet hood assembly
quarter panel rear right
weatherstrip assembly door front right
sealant body 1
hinge bonnet hood right
fender panel front left
Secondary Parts

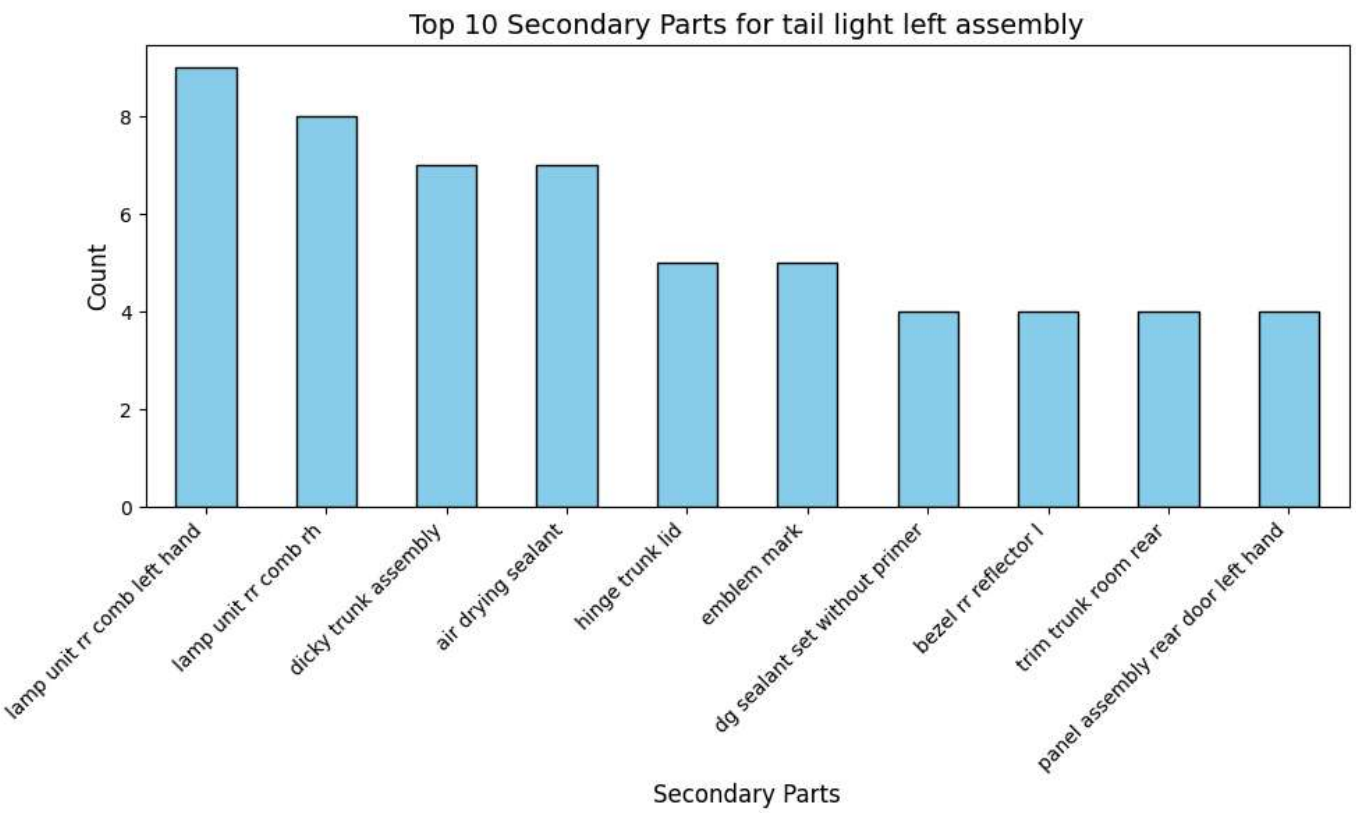

Top 10 Secondary Parts for glass rear door window left
Count
0
50
100
150
200
250
fender panel front left
bonnet hood assembly
moulding front windshield
sealant front windshield glass 1
fender panel front right
roof panel assembly
tail light left
hinge bonnet hood right
fender wheel well lining guard cover left
hinge bonnet hood left
Secondary Parts

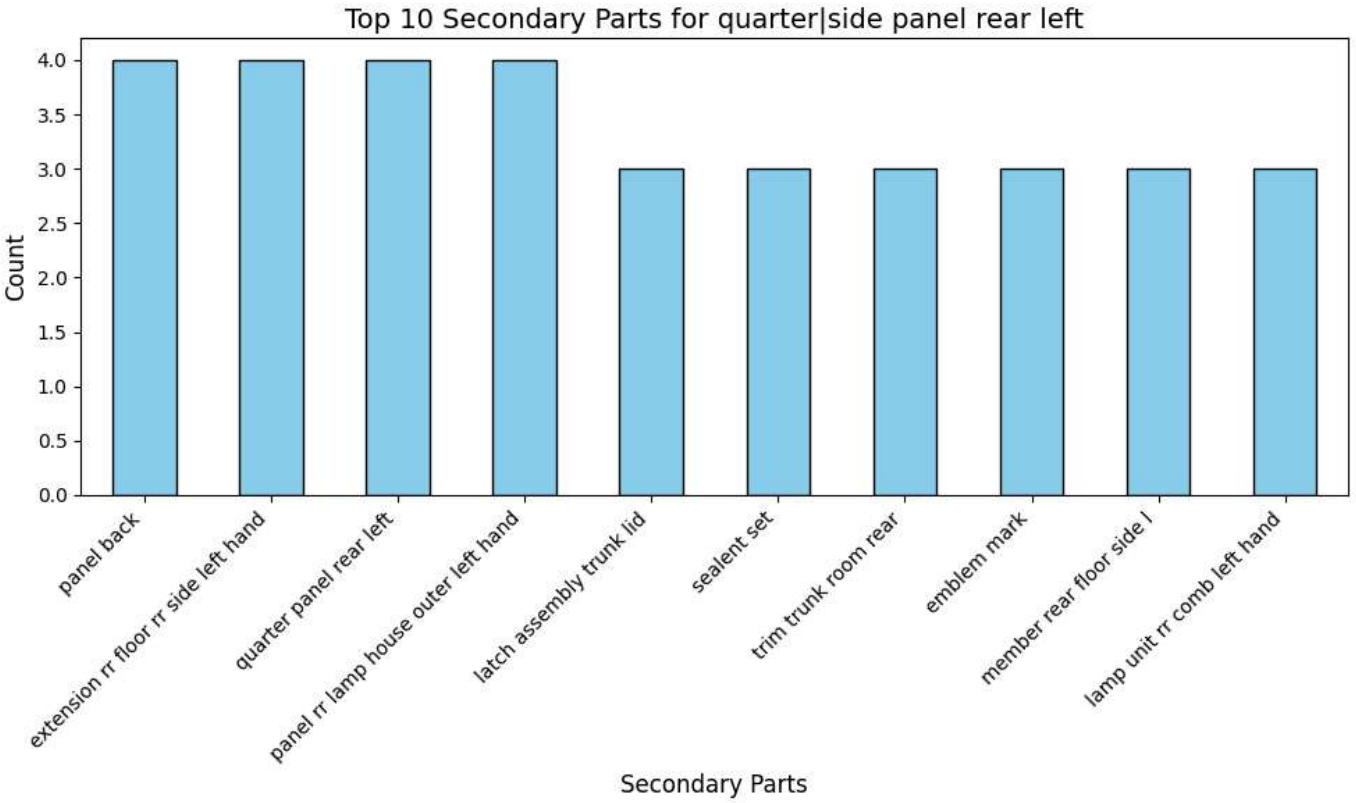

Top 10 Secondary Parts for glass front door window left
Count
0
50
100
150
200
250
fender panel front left
moulding front windshield
sealant front windshield glass 1
bonnet hood assembly
hinge bonnet hood left
fender wheel well lining guard cover left
hinge bonnet hood right
roof panel assembly
tail light left
fender panel front right
Secondary Parts


Top 10 Secondary Parts for running board|gusset panel right
Count
0
25
50
75
100
125
150
175
200
bonnet hood assembly
fender panel front left
fender panel front right
garnish cowl top
hinge bonnet hood left
hinge bonnet hood right
moulding front windshield
air drying sealant
sealant front windshield glass 1
grille radiator
Secondary Parts


Top 10 Secondary Parts for tail light right assembly
Count
0
5
10
15
20
25
30
35
dicky trunk assembly
lamp assembly rr comb right
emblem ertiga
tail light right
garnish comp back door license
panel comp back
dg sealant set without primer
emblem rear smart hybrid
garnish assembly back door l
sealent set
Secondary Parts


Top 10 Secondary Parts for fender|wing|side panel front left
Count
0
5
10
15
20
25
30
quarter panel rear left
box fuel filler
tail light left
dicky trunk assembly
door fuel filler
panel rr lamp house outer left hand
panel back
air drying sealant
trim tail end
outlet comp ventilator
Secondary Parts


Top 10 Secondary Parts for tail light left assembly
Count
0
2
4
6
8
lamp unit rr comb left hand
lamp unit rr comb rh
dicky trunk assembly
air drying sealant
hinge trunk lid
emblem mark
dg sealant set without primer
bezel rr reflector l
trim trunk room rear
panel assembly rear door left hand
Secondary Parts


Top 10 Secondary Parts for quarter|side panel rear left
Count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
panel back
extension rr floor rr side left hand
quarter panel rear left
panel rr lamp house outer left hand
latch assembly trunk lid
sealent set
trim trunk room rear
emblem mark
member rear floor side l
lamp unit rr comb left hand
Secondary Parts

```
Top 10 Secondary Parts for bumper front assembly:
fender panel front left                        11840
bonnet hood assembly                           11611
fender panel front right                       11524
ac condenser assembly                           7620
hinge bonnet hood left                          6228
hinge bonnet hood right                         6176
fender wheel well lining guard cover left       5503
fender wheel well lining guard cover right      5124
bracket 1                                       4231
clip                                            3571
Name: count, dtype: int64

Top 10 Secondary Parts for bumper rear assembly:
tail light right            5906
tail light left             5032
dicky trunk assembly        4740
bonnet hood assembly        4279
fender panel front right    3838
fender panel front left     3795
bracket 1                   3183
clip                        3175
ac condenser assembly       2808
quarter panel rear right    2645
Name: count, dtype: int64

Top 10 Secondary Parts for grille radiator upper:
bonnet hood assembly                           9755
ac condenser assembly                          8700
hinge bonnet hood left                         5734
hinge bonnet hood right                        5657
fender panel front right                       5591
fender panel front left                        5467
radiator cooling fan assembly                  3970
fender wheel well lining guard cover left      3564
fender wheel well lining guard cover right     3436
ac gas refrigerant                             2989
Name: count, dtype: int64

Top 10 Secondary Parts for head light left:
fender panel front left                       11509
bonnet hood assembly                           8870
hinge bonnet hood left                         5246
hinge bonnet hood right                        4953
ac condenser assembly                          4847
fender wheel well lining guard cover left      4576
fender panel front right                       3200
cover fog light left                           2426
bracket front bumper left                      2377
sealant front windshield glass 1               2348
Name: count, dtype: int64

Top 10 Secondary Parts for head light right:
fender panel front right                       11236
bonnet hood assembly                            8845
hinge bonnet hood right                         5263
hinge bonnet hood left                          5040
fender wheel well lining guard cover right      4566
ac condenser assembly                           4416
fender panel front left                         3036
cover fog light right                           2564
bracket front bumper right                      2346
sealant front windshield glass 1                2334
Name: count, dtype: int64

Top 10 Secondary Parts for windshield glass front:
```

```
sealant front windshield glass 1              16830
moulding front windshield                     13377
bonnet hood assembly                           3104
hinge bonnet hood left                         2101
fender panel front right                       2073
fender panel front left                        2054
hinge bonnet hood right                        2050
ac condenser assembly                          1450
fender wheel well lining guard cover left      1157
fender wheel well lining guard cover right     1148
Name: count, dtype: int64

Top 10 Secondary Parts for glass rear door window right:
fender panel front right         1131
fender panel front left           893
bonnet hood assembly              836
air drying sealant                740
molding windshield                706
quarter panel rear right          672
dg sealant set without primer     643
sealent set                       559
clip                              476
sealant body 1                    445
Name: count, dtype: int64

Top 10 Secondary Parts for door front right:
fender panel front right                      2573
bonnet hood assembly                           894
fender wheel well lining guard cover right     866
fender panel front left                        817
quarter panel rear right                       728
sealant body 1                                 704
moulding front windshield                      681
sealant front windshield glass 1               672
hinge bonnet hood right                        561
hinge bonnet hood left                         537
Name: count, dtype: int64

Top 10 Secondary Parts for fog light front right:
fender panel front right                      2417
bonnet hood assembly                          2098
fender wheel well lining guard cover right    1340
fender panel front left                       1290
cover fog light right                         1154
ac condenser assembly                         1127
hinge bonnet hood right                       1111
hinge bonnet hood left                        1070
reservoir tank assembly windshield washer      833
condenser assembly                             765
Name: count, dtype: int64

Top 10 Secondary Parts for fender|wing|side panel front right:
fender panel front left     1398
bonnet hood assembly        1367
fender panel front right    1325
clip                         953
condenser assembly           833
bolt                         706
air drying sealant           661
hinge bonnet hood left       618
hinge bonnet hood right      617
member comp hood lock        614
Name: count, dtype: int64

Top 10 Secondary Parts for side rear view mirror outer assembly right:
fender panel front right                      2647
```

```
bonnet hood assembly                          1055
fender wheel well lining guard cover right     998
sealant front windshield glass 1               802
fender panel front left                        765
moulding front windshield                      762
hinge bonnet hood right                        623
hinge bonnet hood left                         587
tail light right                               470
quarter panel rear right                       464
Name: count, dtype: int64

Top 10 Secondary Parts for door rear right:
quarter panel rear right                      1578
fender panel front right                      1086
sealant body 1                                 827
tail light right                               762
bonnet hood assembly                           497
fender panel front left                        486
air drying sealant                             446
fender wheel well lining guard cover right     385
moulding front windshield                      384
sealant front windshield glass 1               378
Name: count, dtype: int64

Top 10 Secondary Parts for door front left:
fender panel front left                       1715
bonnet hood assembly                           637
fender wheel well lining guard cover left      585
sealant front windshield glass 1               505
moulding front windshield                      496
sealant body 1                                 467
fender panel front right                       454
hinge bonnet hood left                         435
hinge bonnet hood right                        410
roof panel assembly                            369
Name: count, dtype: int64

Top 10 Secondary Parts for side rear view mirror outer assembly left:
fender panel front left                       1507
fender wheel well lining guard cover left      643
sealant front windshield glass 1               606
bonnet hood assembly                           596
moulding front windshield                      535
hinge bonnet hood left                         447
hinge bonnet hood right                        402
fender panel front right                       318
tail light left                                285
bracket front bumper left                      281
Name: count, dtype: int64

Top 10 Secondary Parts for door rear left:
fender panel front left                       721
quarter panel rear left                       635
sealant body 1                                425
tail light left                               386
bonnet hood assembly                          367
fender panel front right                      309
sealant front windshield glass 1              285
moulding front windshield                     281
fender wheel well lining guard cover left     278
roof panel assembly                           271
Name: count, dtype: int64

Top 10 Secondary Parts for windshield glass rear:
sealant rear windshield glass 1     1331
dicky trunk assembly                1034
```

```
tail light right                    596
tail light left                     566
bonnet hood assembly                418
sealant body 1                      373
sealant front windshield glass 1    370
fender panel front right            348
moulding front windshield           285
emblem logo rear middle             281
Name: count, dtype: int64

Top 10 Secondary Parts for fog light front left:
fender panel front left                       1054
fender wheel well lining guard cover left     1003
cover fog light left                           924
bonnet hood assembly                           791
ac condenser assembly                          716
hinge bonnet hood left                         605
hinge bonnet hood right                        567
radiator cooling fan assembly                  429
bracket front bumper left                      396
sealant front windshield glass 1               330
Name: count, dtype: int64

Top 10 Secondary Parts for quarter|side panel rear right:
air drying sealant          232
dicky trunk assembly        219
fender panel front right    216
bonnet hood assembly        216
roof panel assembly         160
fender panel front left     159
sealent set                 142
tail light right            140
tail light left             135
molding windshield          132
Name: count, dtype: int64

Top 10 Secondary Parts for glass front door window right:
fender panel front right                      375
moulding front windshield                     248
sealant front windshield glass 1              229
fender wheel well lining guard cover right    196
bonnet hood assembly                          194
quarter panel rear right                      166
weatherstrip assembly door front right        163
sealant body 1                                147
hinge bonnet hood right                       146
fender panel front left                       145
Name: count, dtype: int64

Top 10 Secondary Parts for glass rear door window left:
fender panel front left                       262
bonnet hood assembly                          158
moulding front windshield                     154
sealant front windshield glass 1              147
fender panel front right                      135
roof panel assembly                           113
tail light left                               111
hinge bonnet hood right                       106
fender wheel well lining guard cover left     106
hinge bonnet hood left                        104
Name: count, dtype: int64

Top 10 Secondary Parts for glass front door window left:
fender panel front left                       267
moulding front windshield                     218
sealant front windshield glass 1              214
```

```
bonnet hood assembly                          163
hinge bonnet hood left                        150
fender wheel well lining guard cover left     144
hinge bonnet hood right                       142
roof panel assembly                           115
tail light left                               111
fender panel front right                      107
Name: count, dtype: int64

Top 10 Secondary Parts for running board|gusset panel right:
bonnet hood assembly                 212
fender panel front left              122
fender panel front right             121
garnish cowl top                     108
hinge bonnet hood left                92
hinge bonnet hood right               88
moulding front windshield             87
air drying sealant                    75
sealant front windshield glass 1      70
grille radiator                       69
Name: count, dtype: int64

Top 10 Secondary Parts for tail light right assembly:
dicky trunk assembly               34
lamp assembly rr comb right        29
emblem ertiga                      24
tail light right                   21
garnish comp back door license     21
panel comp back                    20
dg sealant set without primer      19
emblem rear smart hybrid           18
garnish assembly back door l       17
sealent set                        17
Name: count, dtype: int64

Top 10 Secondary Parts for fender|wing|side panel front left:
quarter panel rear left                31
box fuel filler                        27
tail light left                        22
dicky trunk assembly                   18
door fuel filler                       18
panel rr lamp house outer left hand    16
panel back                             16
air drying sealant                     15
trim tail end                          12
outlet comp ventilator                 12
Name: count, dtype: int64

Top 10 Secondary Parts for tail light left assembly:
lamp unit rr comb left hand           9
lamp unit rr comb rh                  8
dicky trunk assembly                  7
air drying sealant                    7
hinge trunk lid                       5
emblem mark                           5
dg sealant set without primer         4
bezel rr reflector l                  4
trim trunk room rear                  4
panel assembly rear door left hand    4
Name: count, dtype: int64

Top 10 Secondary Parts for quarter|side panel rear left:
panel back                              4
extension rr floor rr side left hand    4
quarter panel rear left                 4
panel rr lamp house outer left hand     4
```

```
latch assembly trunk lid                3
sealent set                             3
trim trunk room rear                    3
emblem mark                             3
member rear floor side l                3
lamp unit rr comb left hand             3
Name: count, dtype: int64
```