VEHICLE SAFETY RECALL

Speaker:

Kevalkumar
Kento
Jason
Muthukrishnan
Taylor
Paulin

Last 10 Years of Vehicle Safety Recalls

1. Top 10 Recalls – Affected Vehicle Components
2. Top 10 Recalls – Category by Year
3. Time Series by Vehicle Components
4. Recalls by Manufacturers
5. Major Components Recalls

Technologies Overview

Pick/Clean Data

csv
Kaggle Data Source
pandas

Database

Postgresql

ETL

SQLAlchemy
Import csv to SQL

Technologies Overview

API Connection

SQLAlchemy
Flask

Visualizations

D3.js
Plotly.js
Chart.js

- Dropped Nulls
- Dropped Columns
- Group Similar Unique Component
- Reduced Data to Last 10yr

```
# Import dependencies
import pandas as pd
# Read in file
df = pd.read_csv('./resources/recalls.csv')
```

[1]

```
# Information of dataframe
df.info()
```

[2]

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26590 entries, 0 to 26589
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Report Received Date  26590 non-null  object
 1   NHTSA ID              26590 non-null  object
 2   Recall Link           26590 non-null  object
 3   Manufacturer          26590 non-null  object
 4   Subject               26590 non-null  object
 5   Component             26590 non-null  object
 6   Mfr Campaign Number   26561 non-null  object
 7   Recall Type           26590 non-null  object
 8   Potentially Affected  26548 non-null  float64
 9   Recall Description    24189 non-null  object
 10  Consequence Summary   21702 non-null  object
 11  Corrective Action     24202 non-null  object
dtypes: float64(1), object(11)
memory usage: 2.4+ MB
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10128 entries, 0 to 10127
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Report Received Date  10128 non-null  datetim
 1   NHTSA ID              10128 non-null  object
 2   Recall Link           10128 non-null  object
 3   Manufacturer          10128 non-null  object
 4   Subject               10128 non-null  object
 5   Component             10128 non-null  object
 6   Mfr Campaign Number   10128 non-null  object
 7   Recall Type           10128 non-null  object
 8   Potentially Affected  10128 non-null  float64
 9   Recall Description    10128 non-null  object
 10  Consequence Summary   10128 non-null  object
 11  Corrective Action     10128 non-null  object
dtypes: datetime64[ns](1), float64(1), object(10)
```

```
components_list = list(last10yr_df['Component'].unique())
components_list.sort()
components_list
```

```
['AIR BAGS',
 'BACK OVER PREVENTION',
 'CHILD SEAT',
 'COMMUNICATION',
 'ELECTRICAL SYSTEM',
 'ELECTRONIC STABILITY CONTROL',
 'ENGINE',
 'ENGINE AND ENGINE COOLING',
 'EQUIPMENT',
 'EQUIPMENT ADAPTIVE/MOBILITY',
 'EXTERIOR LIGHTING',
 'FORWARD COLLISION AVOIDANCE',
 'FUEL SYSTEM, DIESEL',
 'FUEL SYSTEM, GASOLINE',
 'FUEL SYSTEM, OTHER',
 'HYBRID PROPULSION SYSTEM',
 'INTERIOR LIGHTING',
 'LANE DEPARTURE',
 'LATCHES/LOCKS/LINKAGES',
 'OTHER',
 'PARKING BRAKE',
 'POWER TRAIN',
 'SEAT BELTS',
 'SEATS',
 'SERVICE BRAKES',
...
 'UNKNOWN OR OTHER',
 'VEHICLE SPEED CONTROL',
 'VISIBILITY',
 'VISIBILITY/WIPER',
```

```
# Count of all values in 'Component' column (cleaned)
last10yr_df['Component'].value_counts().sort_index(as
```

```
AIR BAGS                          720
BACK OVER PREVENTION               97
CHILD SEAT                         61
COMMUNICATION                       9
ELECTRICAL SYSTEM                1204
ELECTRONIC STABILITY CONTROL       53
ENGINE                            457
EQUIPMENT                        2047
EXTERIOR LIGHTING                 425
FORWARD COLLISION AVOIDANCE        20
FUEL SYSTEM                       559
HYBRID PROPULSION SYSTEM           39
INTERIOR LIGHTING                   6
LANE DEPARTURE                      8
LATCHES/LOCKS/LINKAGES            177
PARKING BRAKE                     126
POWER TRAIN                       523
SEAT BELTS                        326
SEATS                             270
SERVICE BRAKES                    702
STEERING                          499
STRUCTURE                         505
SUSPENSION                        515
TIRES                             278
TRACTION CONTROL SYSTEM             2
...
UNKNOWN OR OTHER                   21
VEHICLE SPEED CONTROL              39
VISIBILITY                        250
WHEELS                            113
```

1.DATA CLEANING

- Create Postgres Schema
- Connect/Import df to Database with SQLAlchemy

Postgres Table Schema

```
create table recalls (
Report_Received_Date date not null,
NHTSA_ID varchar(10) primary key,
        Manufacturer varchar(50) not null,
        Subject varchar(50) not null,
        component varchar(30) not null,
        Mfr_Campaign_Number varchar(30) not null,
        Recall_Type varchar(20) not null,
        Potentially_Affected varchar(10) not null,
        Recall_Description varchar(2000) not null,
        consequence_Summary varchar(800) not null,
        corrective_action varchar(2000) not null
);
```

```
# Review column names
df.columns
```

```
Index(['Report Received Date', 'NHTSA ID', 'Manufactu
       'Component', 'Mfr Campaign Number', 'Recall Ty
       'Potentially Affected', 'Recall Description',
       'Corrective Action'],
      dtype='object')
```

```
# Rename columns to match postgresql table schema
df.columns.str.replace(" ","_").str.lower()
```

```
Index(['report_received_date', 'nhtsa_id', 'manufactu
       'component', 'mfr_campaign_number', 'recall_ty
```

```
from sqlalchemy import create_engine

from config import username, password, hostname, port, db

# username = "postgres"
# password = unique
# hostname = "localhost"
# port = unique
# db = unique

engine = create_engine(f'postgresql+psycopg2://{username}:{passwo
```

```
#with engine.connect() as conn:
df.to_sql("recalls", con=engine, if_exists="append", index=False)
```

```
# Import dependencies
import pandas as pd

# Read in file
df = pd.read_csv('last10yr_df_cleaned.csv')
df.head()
```

	Report Received Date	NHTSA ID	Manufacturer	Subject	Com
0	2023-01-06	23V002000	Triple E Recreational Vehicles	Battery Disconnect Switch May Short	ELE

2.DATABASE – SQLALCHEMY – CSV/SQL

app.py > home

```python
# Import dependencies
import pandas as pd
from sqlalchemy import create_engine
from flask import Flask, render_template, jsonify

from config import username, password, hostname, port, db


app = Flask(__name__)

engine = create_engine(f'postgresql+psycopg2://{username}:{pa
conn = engine.connect()

@app.route("/")
def home():

    return render_template("index.html")
```

```python
@app.route('/api/v1.0/q1')
def q1():
    query = "select component, sum(cast(potentially_affected as float)) as sum_pa \
        from vehiclerecalls group by component order by sum_pa desc"
    df = pd.read_sql(query, conn)
    # returns a dictionary of lists
    return df.to_dict(orient='list')
    #return df.to_json(orient='split')
```

```python
@app.route('/api/v1.0/q2')
def q2():
    query2 = "select manufacturer, sum(cast(potentially_affected as float)) as sum_pa \
                from vehiclerecalls \
                where manufacturer in ('Mercedes-Benz USA, LLC', \
                'BMW of North America, LLC',\
                'Porsche Cars North America, Inc.',\
                'Honda (American Honda Motor Co.)', \
                'Volkswagen Group of America, Inc.') \
                group by manufacturer order by sum_pa desc"
    df2 = pd.read_sql(query2, conn)
    # returns a dictionary of lists
    return df2.to_dict(orient='list')
    #return df.to_json(orient='split')
```

```python
@app.route('/api/v1.0/q3')
def q3():
    # Sum of recalls (desc) grouped by 'Year' (desc) then 'Component'
    query = "select date_part('Year', report_received_date) as year, component,\
    sum(cast(potentially_affected as float)) as sum_pa from vehiclerecalls \
    group by year, component order by year desc, sum_pa desc"
    df = pd.read_sql(query, conn)
    # returns a dictionary of lists
    #return df.to_dict(orient='list')
    return df.to_json(orient='records')
```

3.API CONNECTION

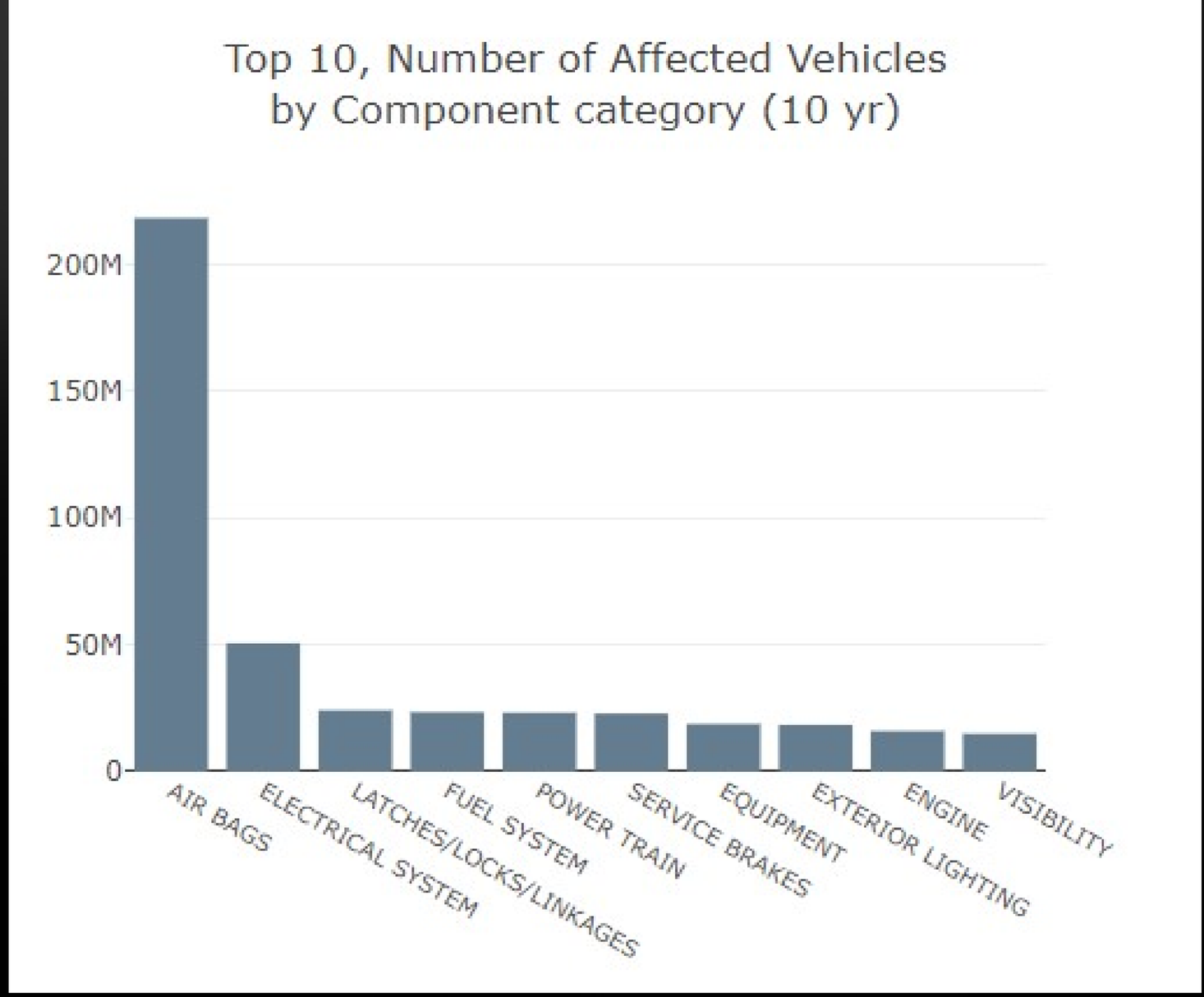
4.VISUALIZATION

```
const url = 'http://127.0.0.1:5000';

function init(){
    // Calls first query
    d3.json(`${url}/api/v1.0/q1`).then(response => {
        //console.log(response);
        let components = response.component;
        //console.log('Component: ', components);
        let totalAffected = response.sum_pa;

        // Data info
        let trace1 = {
            x: components.slice(0,10),
            y: totalAffected.slice(0,10),
            type: 'bar'
        };
        let data = [trace1];

        // Layout info
        let layout = {
            title: 'Top 10, Number of Affected Vehicles
            //height: 500,
            //width: 1200,
            margin: {b:100}
        };

        Plotly.newPlot('bar',data,layout);

    });
```

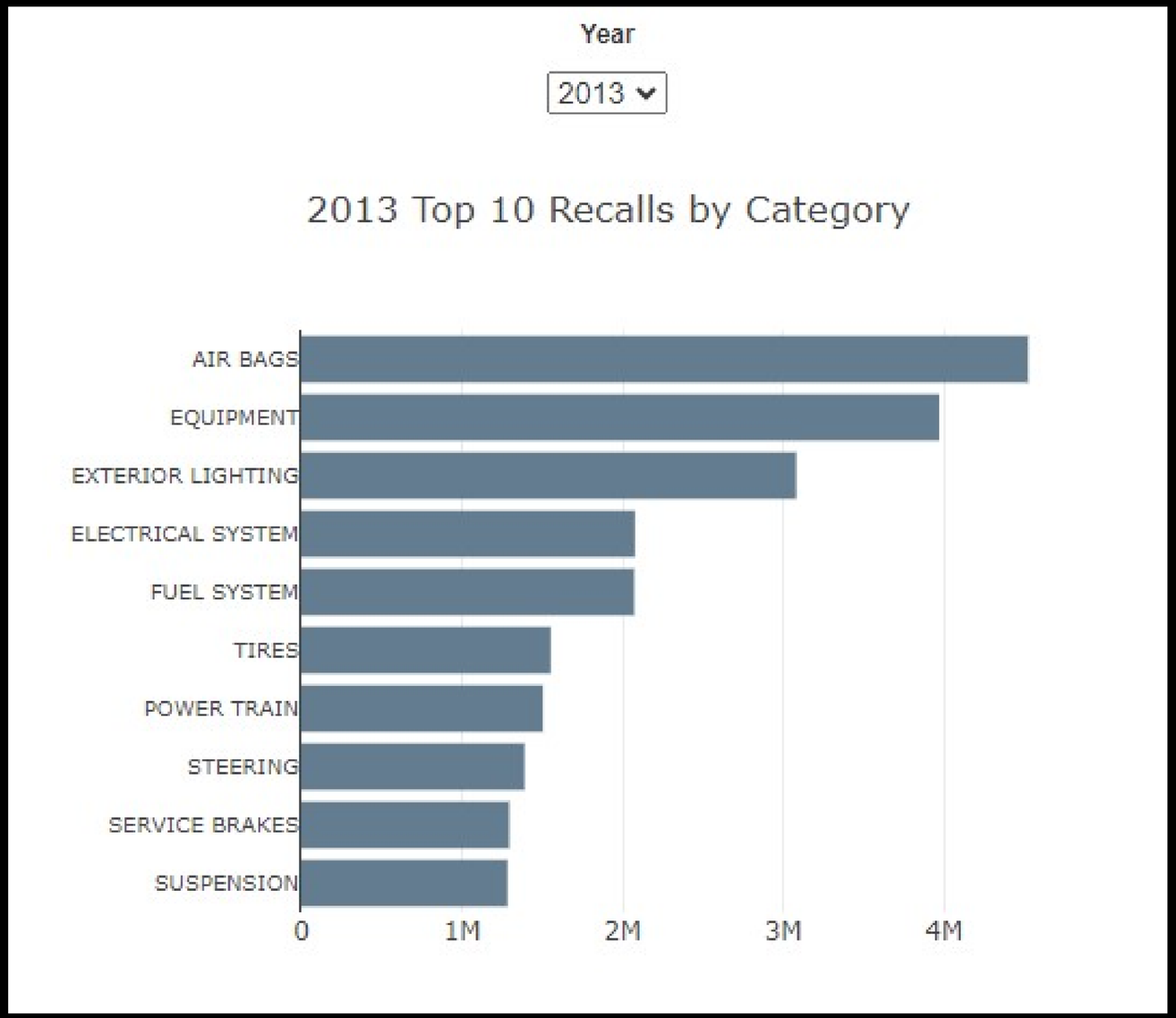
```
// Second query and Pie chart using chart.js
d3.json(`${url}/api/v1.0/q2`).then(response => {
    //console.log(response);
    let manufacturers = response.manufacturer;
    let totalAffected = response.sum_pa;
    console.log('manufacturers: ', manufacturers,totalAffected);
    //const DATA_COUNT = 5;


    // Data info
    const customColors = [
        'rgba(255, 99, 132, 0.8)',
        'rgba(54, 162, 235, 0.8)',
        'rgba(255, 206, 86, 0.8)',
        'rgba(75, 192, 192, 0.8)',
        'rgba(153, 102, 255, 0.8)',
        'rgba(255, 159, 64, 0.8)',
    ];
    const data = {
    labels: manufacturers,
    datasets: [
        {
        label: 'Potentially affected Manufacturers',
        data: totalAffected,
        backgroundColor: customColors ,
        borderColor: customColors,
        borderWidth: 1
        }
    ]
    };

    var ctx = document.getElementById('pie').getContext('2d');
    var myPieChart = new Chart(ctx, {
        type: 'pie',
        data: data,
        options: {
            plugins: {
              title: {
```

```
// Calls third query
d3.json(`${url}/api/v1.0/q3`).then(response => {
    // get unique year values from response for dropdown list
    let yearArray = response.map(response => response.year);
    let yearUnique = yearArray.filter((value, index) => yearArray.in
    console.log('Years: ', yearUnique);

    // Create selector1 options and properties for first dropdown me
    let selector1 = d3.select('#selDataset1');
    for (let i = 0; i < yearUnique.length; i++){
        selector1.append('option').text(yearUnique[i]).property('val
    };
    let firstYear = yearUnique[0];
    buildChart1(firstYear);

});
```

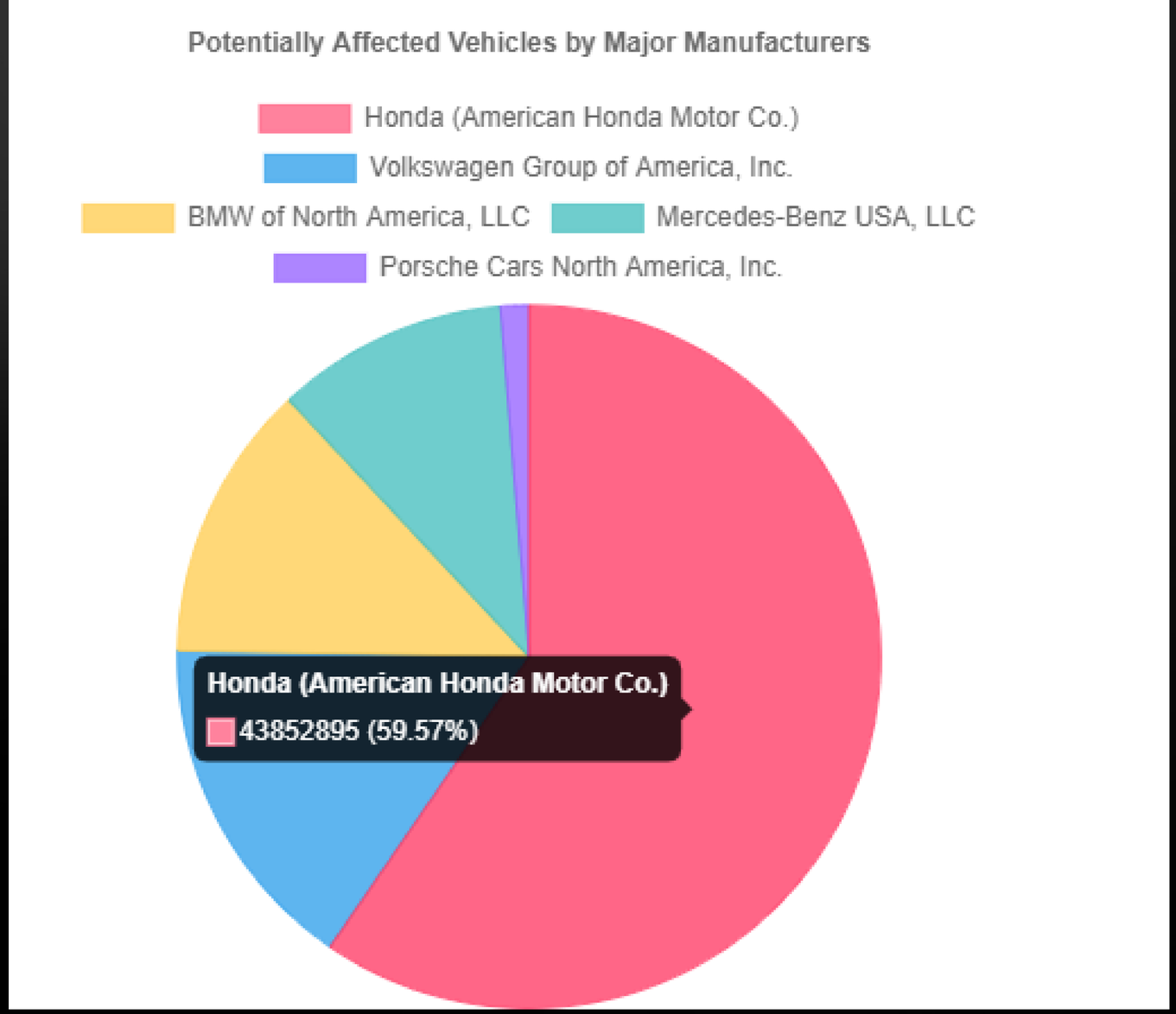
```
function buildChart1(value){
    d3.json(`${url}/api/v1.0/q3`).then(response => {
        let results = response.filter(response => response.year == value);
        let components = results.map(result => result.component);
        let totalAffected = results.map(result => result.sum_pa);

        // Horizontal bar chart info
        let bar_yticks = components.slice(0,10).reverse();
        let bar_xticks = totalAffected.slice(0,10).reverse();

        let barData = [
            {
                y: bar_yticks,
                x: bar_xticks,
                type: 'bar',
                orientation: 'h'
            }
        ];
        let barLayout = {
            title: `${value} Top 10 Recalls by Category`,
            margin: {l:150},
            yaxis: {
                tickfont: {size: 10}
            }
        };

        Plotly.newPlot('bar1', barData, barLayout);
    });
};

function optionChanged(value){
    buildChart1(value);
}

init();
```



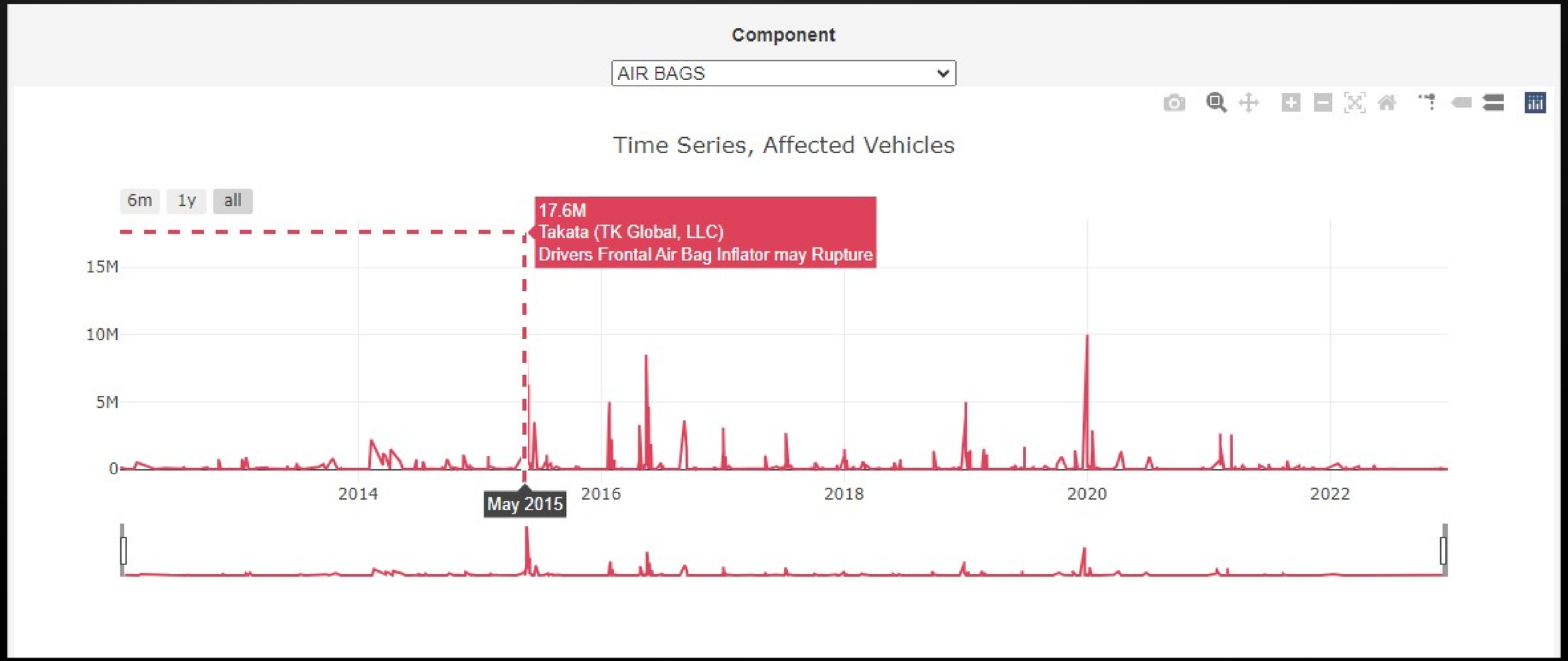
TOP 10, NUMBER OF AFFECTED VEHICLES



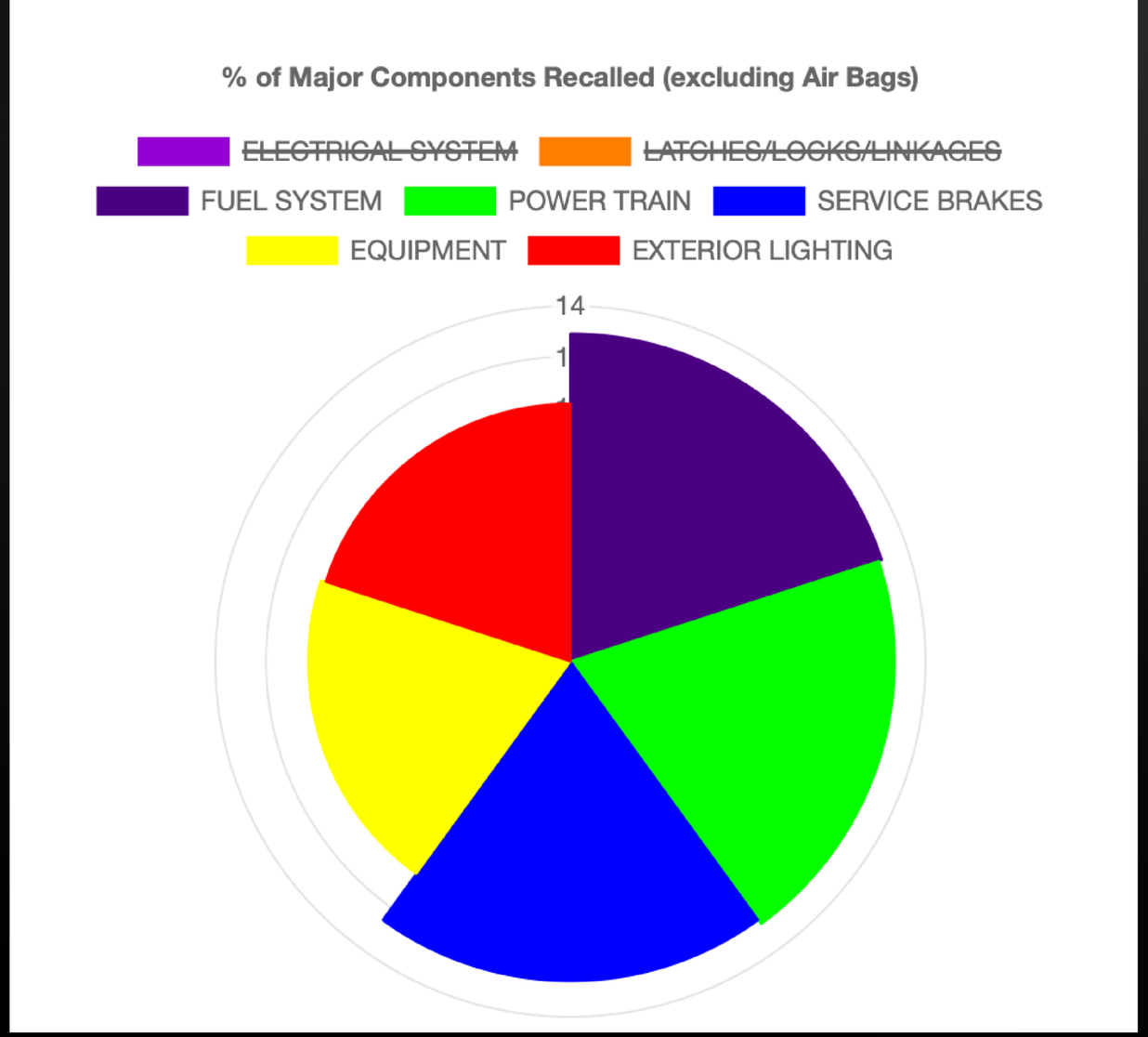
TOP 10 ANNUAL RECALLS

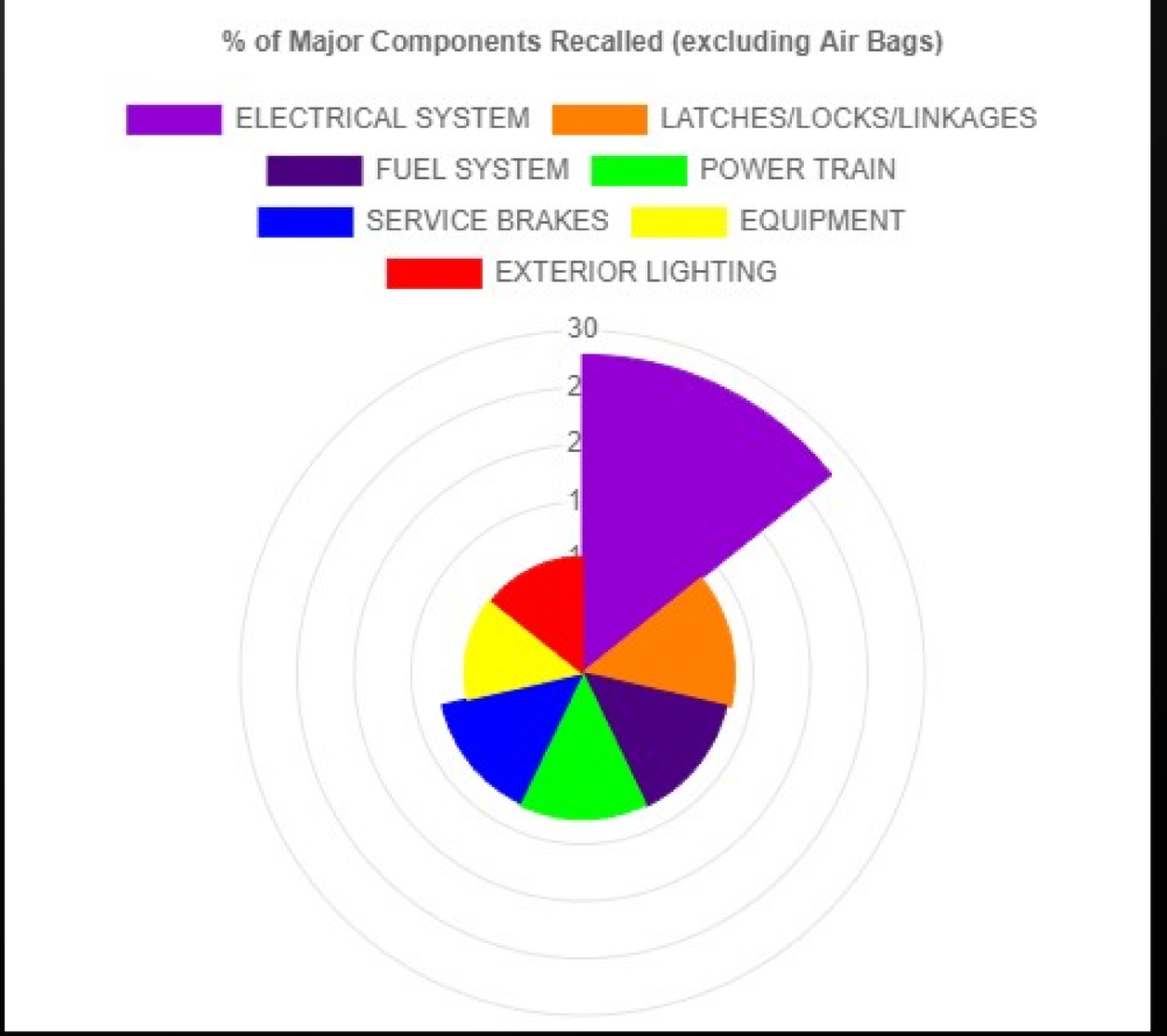


POTENTIALLY AFFECTED VEHICLES BY MAJOR MANUFACTURERS



TIME SERIES, AFFECTED VEHICLES





PERCENTAGE OF MAJOR COMPONENTS RECALLED

THANK YOU