Theodore Koby-Hercsky

01/29/2023

DSC680-T301 2233-1

Professor Catie Williams

Final White Paper

Github: Sentiment Analysis Car Brand Reviews

Presentation Link: https://youtu.be/XXjVLOkLeXE

Business Problem

When it comes time to choosing a car it can be a big decision that can be overwhelming for some as now a days there are so many car brands to choose from. Problems customers are running into are finding a way to filter out what you are looking for and not looking for in a car as the choice you make can be a large financial impact that can last for several years. The business problem this report will be aiming to solve is provide customers with ease of mind by deciphering reviews from each brand to determine the best option when purchasing a vehicle. The use of sentiment analysis will allow customers to analyze mass amounts of reviews to determine the true feedback for each brand. While car brands can also take advantage of sentiment analysis to determine what is working for their company and what they need to improve on. Overall, the goal for this report is to help customers and or brands to decipher the reviews given to provide a true depiction of what you are signing up for when you purchase one a vehicle.

Background/History

The data set that I will be utilizing contains review data for five different car brands Audi, Lexus, Infiniti, BMW, and Mercedes-Benz. The data set includes five variables which include the rating, car ear, brand name, date, and review that can be seen below with a brief description of each in the data dictionary. The data was pulled from Kaggle and the date ranges from 1997 to 2018 for the year the car was built. The five car brands that are used are all in the luxury brand field with Mercedes, Audi, BMW being on the pricier end for the most part. While some customers might have done some research on car rating given J. D. Powers as they focus on "quality and reliability ratings are a combination of quality and dependability scores." (J.D.) While the quality scores are based "on initial owner response and feedback of their new purchase whereas Dependability scores focus on longer-term ownership experiences of 3 years." (J.D.) Overall by reviewing each car brand and performing sentiment analysis on all reviews the goal is to gain a deeper understanding of the positive and negative feedback of each brand to determine the optimal choice for all customers.

Data Explanation (Data Prep/Data Dictionary/etc)

The data I went with focuses on the reviews of five car brands from Kaggle for luxury car brands such as Mercedes-Benz and BMW. The variables that are included in the data set consist of the rating which is displayed as a numeric value that ranges from one to five while another variable is the review which is written by the customer that is regarding the car they purchased. When viewing the dataset, it was determined that all personal identifiable information was excludes in the dataset which allows for accurate reviews of each brand an no way to track a review back to a single user.

Data preparation was needed for further exploration of the date set using describe a "method returns description of the data in the DataFrame" which indicated that the mean rating came in a 4.47% which can be seen below in figure one. (W3Schools) The use of function isnull() was also used to "detect missing values for an array-like object" in the DataFrame which indicated that no missing values have been detected as seen in figure two. (Pandas) The use of unique also came in handy as it allowed me to

view each brand that was listed in the DataFrame which indicated the five brands that would be reviewed seen in figure three. Next I decided to create a quick sentiment rating just to see what I am dealing with by using an if and elif statement. This statement separated the reviews to either a one or a zero depending on the rating seen in figure four. Next, I will take you through the methods I used to conduct further analysis on the reviews provided.

Methods

The use of natural language processing is the first step I will take before conducting any sentiment analysis. These steps include but is not limited to tokenizing sentences, removing stop words, normalizing words, and vectorizing text. The first step that was implemented was lowercase which applied lambda to the reviews to implement lowercase to all letters. Next, I removed all punctuation through the use of strip, split, and apply as seen below in figure five. Once all punctuation was removed I continued by removing all imbedded numbers by using isdigit() and apply which is also seen below in figure five.

Another important process is tokenizing a sentence which is the breakdown of text into sentences, words, and other units. Next is the removal of stop words such as “if”, “but”, and “or” while indicating for the function to keep brand names such as Audi, Lexus, INFINITI, BMW, Mercedes, and Benz seen in figure six. Vectorizing text is also used which is the process of “turning the text into a numerical representation for consumption by your classifier.” (Stratis) Last, I will use VADER sentiment analysis which is a “lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media.” (Ankthon) Seen in figure seven which was used to create polarity and rating that will produce a in depth score rating and a finer scale indicating positive, negative, or neutral for the Vader polarity.

The creation of visualizations was the next step which started with a seaborn bar plot that used the variables brand name, rating, and car reviews. Which can be seen in figure eight that

shows the average rating received for each car brand that showed all brands with an average rating of around four out of five with Lexus with a little higher of an average. A countplot was also created using seaborn that showed the number of reviews by year of the car model seen below in figure nine. Showing that car models from 2004 had over 3,000 reviews while cars from 2012 had just over 500 reviews. Another visualization that was utilized was from plotly express and was a scatter plot that showed Vader rating by brand and date. Seen in figure fifteen it shows that mass amounts of reviews by ratings throughout the years for all brands. Which shows that some brands like Infiniti, Lexus, and Audi had the least number of reviews compared to BMW and Mercedes. Last I created two bar plots with matplotlib to show the rating count percent with the original rating variable and the polarity Vader review rating. Figure ten shows the bar chart for the original variable rating shows 54.36% of all reviews being a rating of 4 while ratings one, two and three add up to only 15.01%. While in figure eleven shows that all positive reviews add up to 86.69% while negative and neutral reviews only add up to 13.31%.

Analysis

Modeling is the next step in this report that starts with the creation of X and Y for the target and predict variables. As the target variable for this report is the updated review which is being used as the X while the Y is the predict which uses the review polarity as Y. Next, I printed the shape of the X and y for both test and train. The x and y train has 19,162 values with one variable while the x and y test shows 12,776 values with one variable. Next I went on to creating TF-IDF which stands for "Term Frequency – Inverse Document Frequency and is a statistic that aims to better define how important a word is for a document." (Borcan) Seen in figure twelve the creation of the x train vector which has a shape of 19,162 and 21,980 that was created with fit transform.

The first model that was used is the Logistic Regression that aims to "solve classification problems by predicting categorical outcomes." (W3Schools) Seen in figure thirteen it shows with the use of logistic regression and the x train vector created an accuracy of 91.49% which is a great start.

The next model I used was the Random Forest Analysis a “supervised learning algorithm that can be used both for classification and regression and is flexible and easy to use.” (Navlani) The creation of the random forest classifier model showed an accuracy of 99.95% that can be seen n figure fourteen. Overall, it is seen that the random forest model had the highest accuracy compared to the logistic regression model. Next I will walk you through the additional models that have been created that focused on each brand.

The use of get dummies was used on the on the brand name variable to create separate variables for each brand to models for each. Next, I split the data into training and test sets and used the Audi dummy variable as the predict data as seen in figure sixteen below. Which moved on to create the tfidvectorizer for the X Train and then the logistic regression that concluded an accuracy of 90.61% for Audi. This process was used on all five brands and showed an average of upper eighties to low nineties in accuracy for all models as seen in figure seventeen. These models indicated that Infiniti had the highest accuracy coming in at 91.41%, Audi 90.61%, Lexus 90.10%, Mercedes 88.43%, and BMW with 87.77% in accuracy for the Logistic Regressions. Last I wanted to go over a seaborn count plot that shows the count of each brand and the positive, neutral, and negative review count received by each brand. Indicating BMW had the most positive reviews with 6,983 while Infiniti had the lowest positive reviews coming in at 3,850. While Mercedes had 959 negative reviews which was the highest wile Infiniti only had 380 negative reviews. Overall, the findings have been very interesting and point to BMW with the highest positive feedback while Mercedes has shown the largest number of negative reviews.

Conclusion

The two models had great findings that both landed in the 90 percentile for accuracy while the Random Forest mode had an accuracy of 99.95% while the Logistic Regression had an accuracy of only 91.49%. All logistic regressions that have been created for each individual brand came back in the low nineties for accuracy. Indicating that Infiniti had the highest accuracy while BMW showed

the lowest accuracy for logistic regression but only had a 2% difference meaning all brands have a great opportunity to provide accurate feedback. In regards to the brand rating visualization it was determined that BMW had the highest rate of positive reviews while Infiniti showed the lowest rate of positive, negative, and neutral reviews. In astonishment the brand that showed the highest count of negative reviews came from Mercedes which had the second highest in positive reviews as well. Overall, each brand has seen its strengths and weaknesses, but I believe the two brands that I would recommend researching further to pick for your next car would be BMW and Lexus as these both had positive feedback and are great options.

Assumptions

The assumptions that can come from the findings in this project is that the largest number of reviews come from cars that are from 2004 with over 3,000 reviews while cars created in 2012 showed the least amount of reviews with only a little over 500 being submitted for the brands from this data set. An assumption that can be seen is that as the years go on the amount of reviews have decrease and there is less issues customers are running into. While most ratings range from a four to five with only 15% being neutral or negative for reviews. Meaning that car brands are seeing good reviews but need to read between the lines to find what customers are looking for and want to stay around in the new cars to come.

Limitations

The limitations that I have ran into and noticed with the data set is that there is only luxury car brand which indicates reviews from wealthier consumers. While the reviews given are few in comparison to what should be seen for the year range that the cars are coming from. While the reviews seem to mostly be positive which limits the number of negative reviews seen within the data. Overall, the data limits users to knowing all facts given on the car brands over the years which could hinder their choice in companies to go with for their next vehicle.

Challenges

The challenges I see when it comes to sentiment analysis for car brand reviews is that users will see positive and negative reviews for each company but in the end, they will need to weigh the options and decide on what they can and cannot live with in their next vehicle. While on the car brand side the companies that use sentient analysis will need to view their reviews and determine the big picture behind them such as outdated technology, engine issues, and more. As BMW has seen reviews regarding outdated technology which is a great start for where they can implement improvements to raise customer satisfaction. The challenge with this is companies take several years to implement changes in their cars as it takes time, testing, and resources to make changes.

Future Uses/Additional Applications

Future use for this project can be used on any reviews one might come across being in the car brand area or even other areas such as book reviews, product reviews, and more. While this report can also go into mor details on each car brand and would have better results if more reviews were acquired. While for use on the car company side managers and analysts can use this project to further evaluate their cars and determine what features they are missing and what they can do without.

Recommendations

The recommendations that can be given are that all car brands have around an average rate of a four out of five. While Lexus has seen the highest ratings while Infiniti and Mercedes seem to have the lowest rating due to a larger number of lower ratings. While in recent years cars that where built in 2016 showed the highest numbers of rating within the last ten years with over 1,000 given. With that being said I would steer clear of cars from 2016 due to the large number of reviews within the years. When researching further I have also indicated that two great brands would be BMW and Lexus that a customer would want to consider with their next purchase.

Implementation Plan

The plan for implementation on the car company side would be for analysts and managers to use sentiment analysis on the reviews the receive and determine what the can work on fixing and what the need to keep to retain current customers and gain new ones. While customers will need to conduct sentiment analysis on car company reviews that they are interested in purchasing to determine the best brand to go with.

Ethical Assessment

An ethical assessment can be conducted on each car brand by analyzing the reviews they receive ethically and determine the best course of action for each department within the company. While auditing their reviews on a monthly basis and follow up with departments to verify improvements are being made. Hopefully by analyzing reviews and auditing each department companies can have a chance to improve on processes and their company.

Data Dictionary

- **Rating** - A numeric variable that gives the car brand a rating of one to five.
- **Car_Year** – The year the car was built.
- **Brand_Name** – Is the brand such as Audi, Lexus, Infiniti, BMW, or Mercedes-Benz
- **Date** – Is the date the owner of the car created the review.
- **Review** – The review given by the costumer regard the car they purchased.

Figures

- Figure 1:

```
# Use describe to see different statistics for the data set
Car_Reviews.describe()
```

	Unnamed: 0	Rating	car_year
count	31938.000000	31938.000000	31938.000000
mean	15968.500000	4.476331	2005.959296
std	9219.850785	0.783148	5.101718
min	0.000000	1.000000	1997.000000
25%	7984.250000	4.375000	2002.000000
50%	15968.500000	4.750000	2005.000000
75%	23952.750000	5.000000	2009.000000
max	31937.000000	5.000000	2018.000000

- Figure 2:

```
# I will now use isnull and sum to check for missing data
Car_Reviews.isnull().sum()
```

```
Unnamed: 0    0
Rating        0
car_year      0
brand_name    0
date          0
review        0
dtype: int64
```

- Figure 3:

```
# View how many unique values the brand_name variable has
Car_Reviews['brand_name'].unique()
```

```
array(['Audi', 'Lexus', 'INFINITI', 'BMW', 'Mercedes-Benz'], dtype=object
```

- Figure 4:

```
# Next I will convert my ratings to be replaced with a 0 or 1
def create_sentiment(rating):

    if rating<=3.99:
        return 0
    elif rating>=4:
        return 1

Car_Reviews['Updated_Rating'] = Car_Reviews['Rating'].apply(create_sentiment)
```

- Figure 5:

Implementing Lowercase

```
# Next I will use Natural language processing to edit the reviews for each car brand.By
Car_Reviews['Updated_Review']= Car_Reviews['review'].apply(lambda x: x.lower())
```

Punctuation Removal

```
# Next remove punctuation
def punctuation_removal(text):
    text = " ".join([word.strip(string.punctuation) for word in text.split(" ")])
    return text
Car_Reviews['Updated_Review'] = Car_Reviews['Updated_Review'].apply(punctuation_removal)
```

Remove Imbedded Numbers

```
# Next I will use isdigit and apply to remove imbedded numbers
def remove_number(text):
    text = "".join([word for word in text if not any(c.isdigit() for c in word)])
    return text
Car_Reviews['Updated_Review'] = Car_Reviews['Updated_Review'].apply(remove_number)
```

- Figure 6:

```
# Set stop words to english
brand_stop_words = set(stopwords.words('english'))
brand_names = ['Audi', 'Lexus', 'INFINITI', 'BMW', 'Mercedes', 'Benz']
punctuation_string = string.punctuation
```

```
# Next I will remove the stopwords from my Review_update
def remove_stopword(text):
    stop = stopwords.words('english')
    text = [x for x in text if x not in brand_stop_words and x not in brand_names and x not in punctuation_string]
    return text
Car_Reviews['Updated_Review'] = Car_Reviews['Updated_Review'].apply(remove_stopword)
```

```
# Import needed package
import nltk
nltk.download('omw-1.4')
```

- Figure 7:

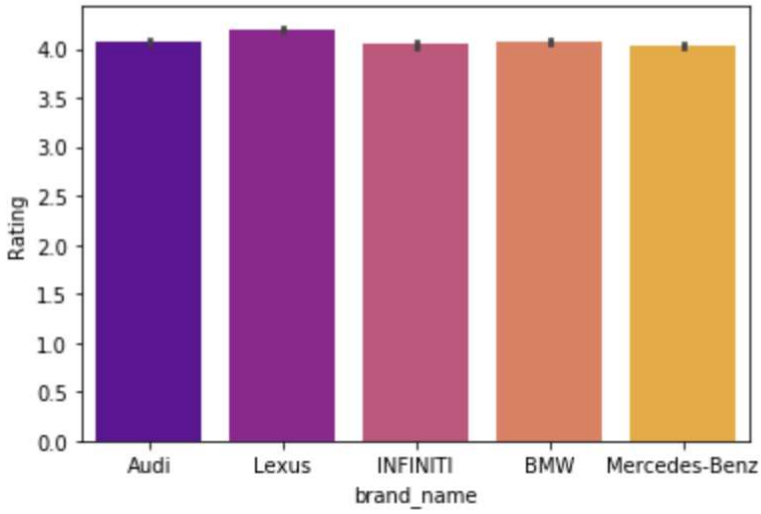
```
# I will use SentimentIntensityAnalyzer to calculat the vader rating
vader_analyzer=SentimentIntensityAnalyzer()
vader = []

for i in Car_Reviews['Updated_Review']:
    score = vader_analyzer.polarity_scores(i)
    vader.append(score['compound'])
Car_Reviews['Rating_Vader'] = vader
```

```
# I will also determine the rating of each value by setting the vader to positive, negative, or neutral
Car_Reviews['Polarity_Vader_Review'] = Car_Reviews['Rating_Vader'].apply(lambda x: 'Positive' if x > 0
                                                                        else('Neutral' if x == 0 else 'Negative'))
Car_Reviews.head()
```

- Figure 8:

```
# First I will create a barplot for each brand that shows the average rating
import seaborn as sns
sns.barplot(x ='brand_name', y ='Rating', data = Car_Reviews,
            palette ='plasma')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x272a09469e8>
```

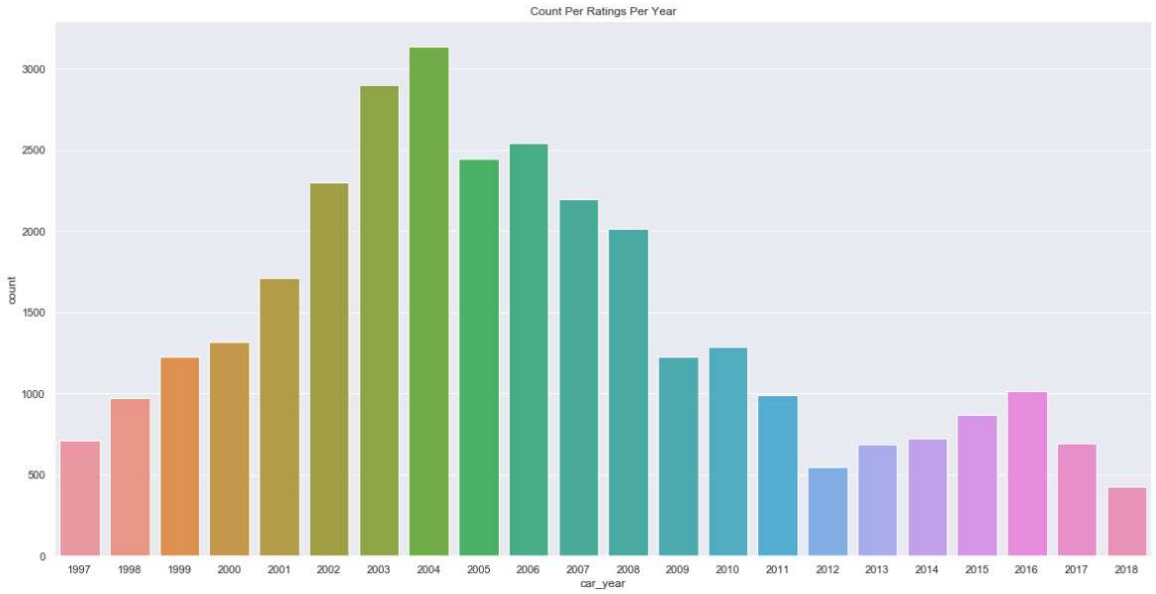


- Figure 9:

Number of Reviews by Car Model Year

```
# Next I will create a plot that shows the number of reviews per car model ye
sns.set(rc={'figure.figsize':(20,10)})
sns.countplot(x ='car_year', data = Car_Reviews).set(title='Count Per Ratings
```

7]: [Text(0.5, 1.0, 'Count Per Ratings Per Year')]

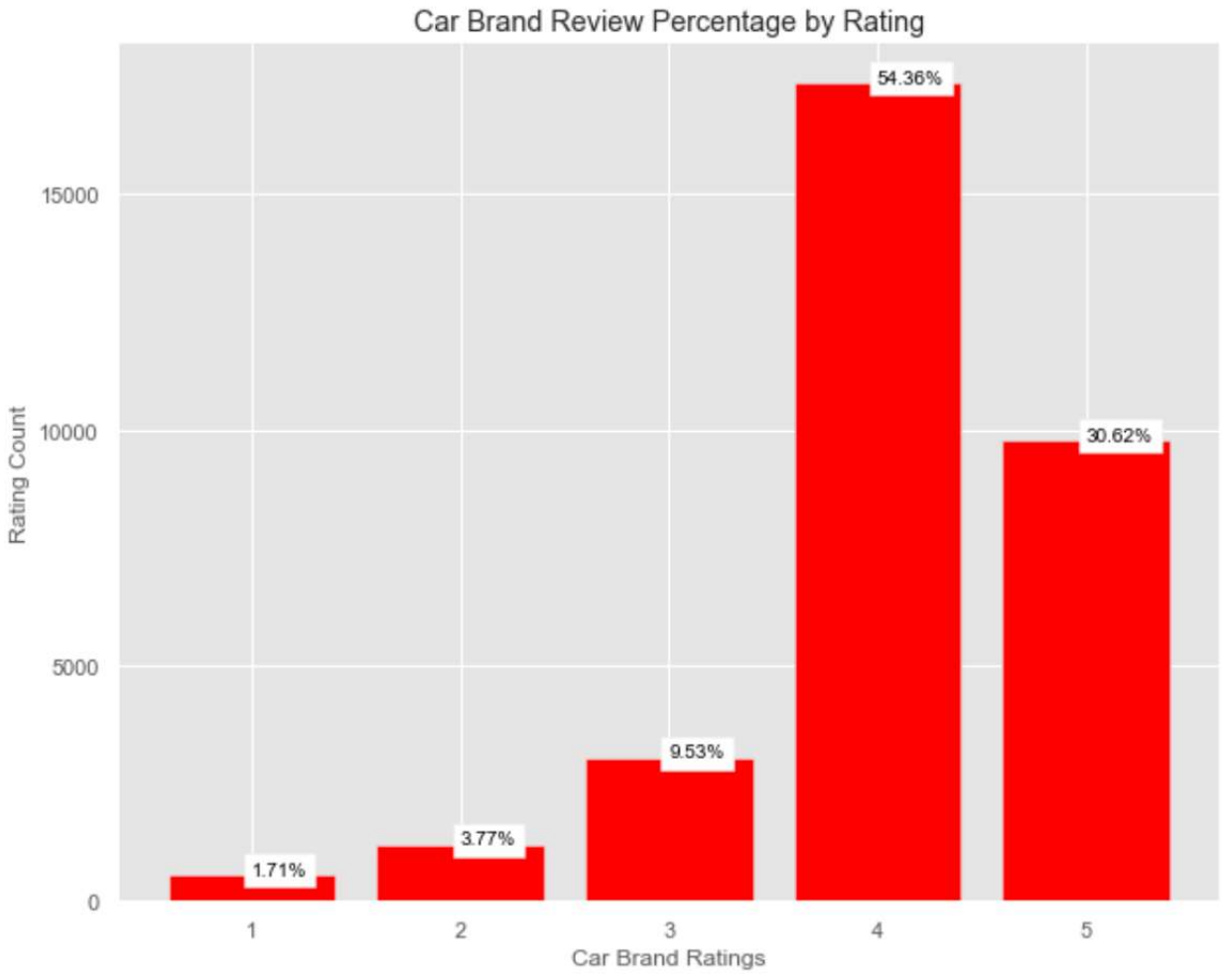


- Figure 10:

Rating Count Percentage

```
# First I will create rate value for the car reviews.
rate_value = Car_Reviews.Rating.value_counts()
```

```
# Create a bar plot that shows the percentage of ratings
with plt.style.context('ggplot'):
    plt.figure(figsize=(10, 8))
    plt.bar(rate_value.index, rate_value.values, color='red')
    for rating, value in zip(rate_value.index, rate_value.values):
        plt.text(rating, value, str(round(value/sum(rate_value.values)*100,
                 color='black', bbox=dict(facecolor='white'))
    plt.title('Car Brand Review Percentage by Rating')
    plt.xlabel('Car Brand Ratings')
    plt.ylabel('Rating Count')
    plt.yticks(np.arange(0, 20000, 5000))
```

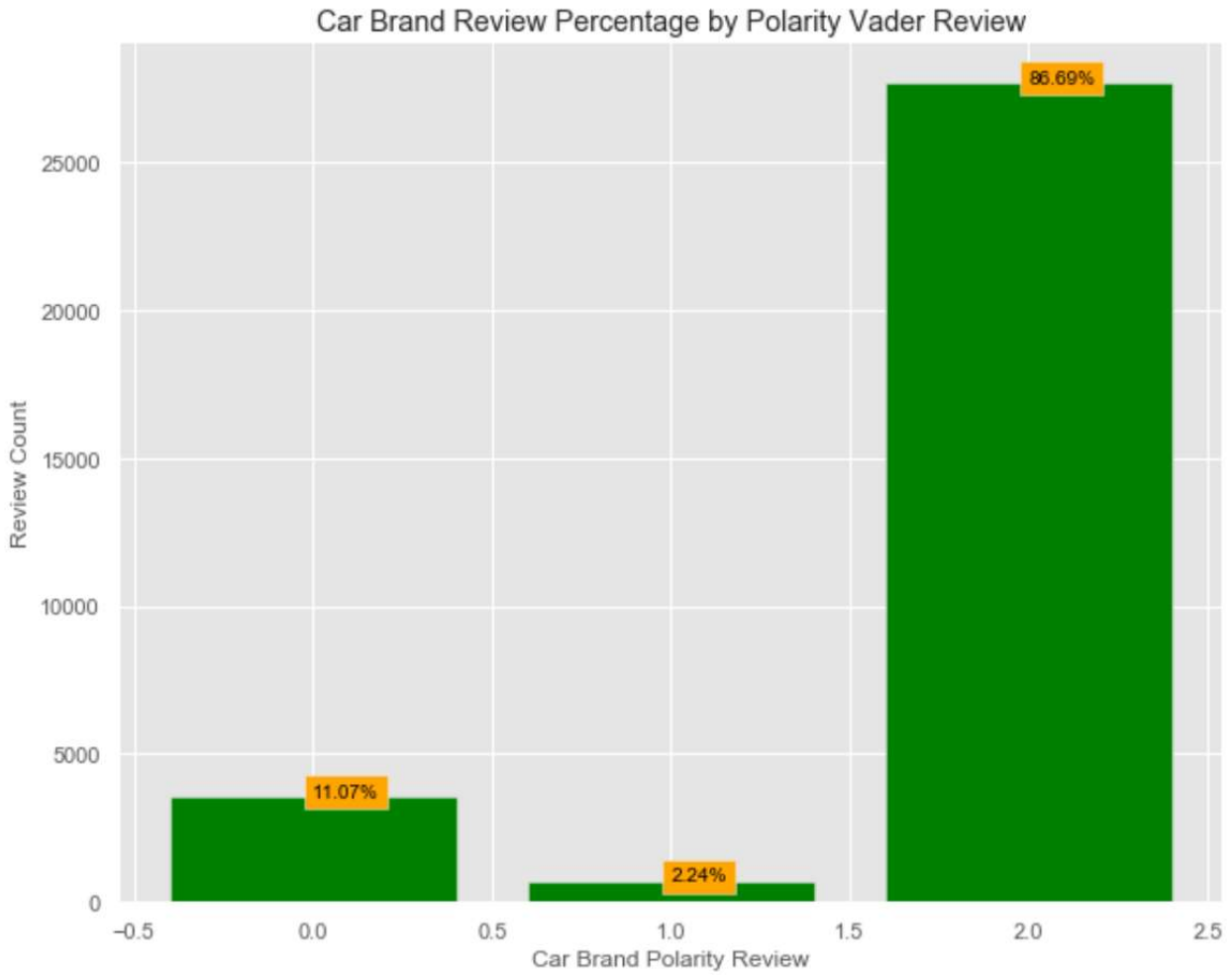


- Figure 11:

Car Brand Review Percentage by Polarity Vader Review

```
# First I will create rate value for the car reviews.
rate_value = Car_Reviews.Polarity_Vader_Review.value_counts()
```

```
# Create a bar plot that shows the percentage of ratings
with plt.style.context('ggplot'):
    plt.figure(figsize=(10, 8))
    plt.bar(rate_value.index, rate_value.values, color='green')
    for rating, value in zip(rate_value.index, rate_value.values):
        plt.text(rating, value, str(round(value/sum(rate_value.values)*100,
                 color='black', bbox=dict(facecolor='orange'))
    plt.title('Car Brand Review Percentage by Polarity Vader Review')
    plt.xlabel('Car Brand Polarity Review')
    plt.ylabel('Review Count')
    plt.yticks(np.arange(0, 30000, 5000))
```



- Figure 12:

TFIDF Training Set Data

```
# Create TfidfVectorizer feature for the x_train
tfidf_train_data = TfidfVectorizer()
x_train_vector = tfidf_train_data.fit_transform(x_train.ravel())
```

```
# Pull TfidfVectorizer train shape
x_train_vector.shape
```

```
]: (19162, 21980)
```

- Figure 13:

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
```

```
# logistic regression model with train data
logistic_model = LogisticRegression()
logistic_model = logistic_model.fit(x_train_vector, y_train)
print(logistic_model.score(x_train_vector, y_train))
```

```
0.9149358104581985
```

- Figure 14:

Random Forest Analysis

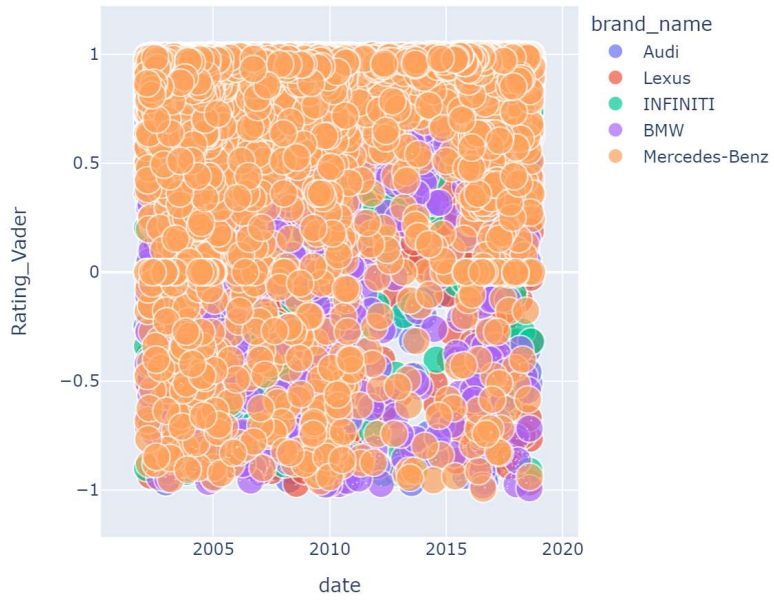
```
from sklearn.ensemble import RandomForestClassifier
```

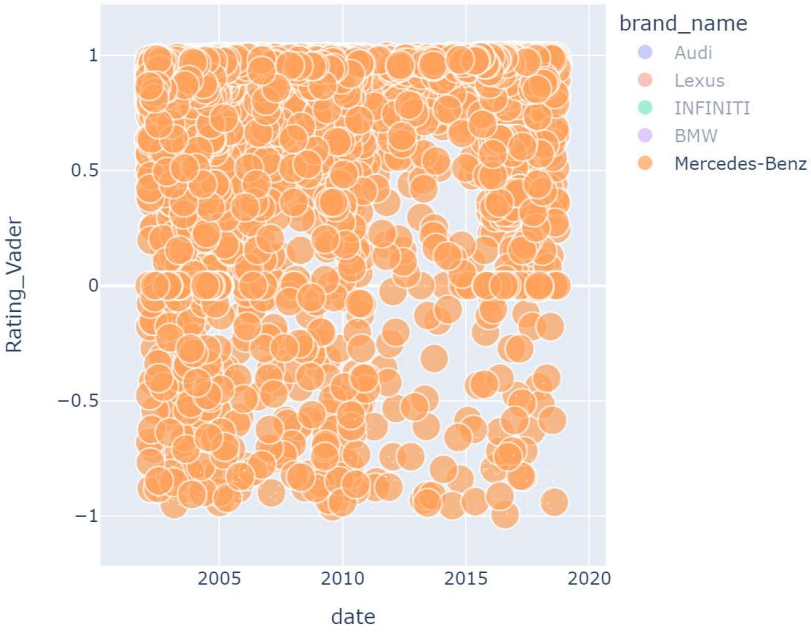
```
# Random Forest Classifier accuracy on the train set
Random_Forest = RandomForestClassifier(max_depth = 300, random_
Random_Forest.fit(x_train_vector, y_train.ravel())
print(Random_Forest.score(x_train_vector, y_train.ravel()))
```

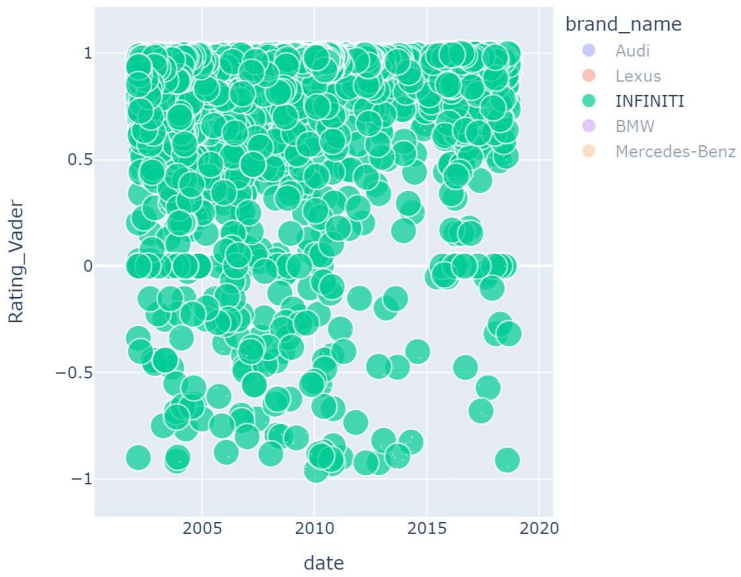
```
0.9995303204258428
```

- Figure 15:

```
# I will create a scatter that shows the Vader rating throughout the years
fig = px.scatter(Car_Reviews,
          x='date',
          y='Rating_Vader',
          color='brand_name',
          size='Updated_Rating')
fig.show()
```


brand_name
Audi
Lexus
INFINITI
BMW
Mercedes-Benz
Rating_Vader
1
0.5
0
−0.5
−1
2005
2010
2015
2020
date


brand_name
Audi
Lexus
INFINITI
BMW
Mercedes-Benz
Rating_Vader
1
0.5
0
−0.5
−1
2005
2010
2015
2020
date


brand_name
Audi
Lexus
INFINITI
BMW
Mercedes-Benz
Rating_Vader
1
0.5
0
−0.5
−1
2005
2010
2015
2020
date

- Figure 16:

```
# Now I will split the Car review data into a training and test sets for Audi
Target_data = ['Updated_Review']
Predict_data= ['Audi']
X=Car_Reviews[Target_data].values
Y=Car_Reviews[Predict_data].values
x_train, x_test, y_train, y_test = train_test_split(X,Y, stratify=Y, test_size=.4, random_state=0)
```

```
# View the shape for my training and test sets for Audi
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
np.unique(y_train, return_counts=True)
```

```
(19162, 1)
(12776, 1)
(19162, 1)
(12776, 1)

(array([0, 1], dtype=uint8), array([15536,  3626], dtype=int64))
```

```
# Create TfidfVectorizer feature for the x_train
tfidf_train_data = TfidfVectorizer()
audi_x_train_vector = tfidf_train_data.fit_transform(x_train.ravel())
```

```
# Pull TfidfVectorizer train shape
audi_x_train_vector.shape
```

```
(19162, 22105)
```

```
# View the Audi logistic regression model with train data
audi_logistic_model = LogisticRegression()
audi_logistic_model = logistic_model.fit(audi_x_train_vector, y_train)
audi_score = audi_logistic_model.score(audi_x_train_vector, y_train)
audi_score
```

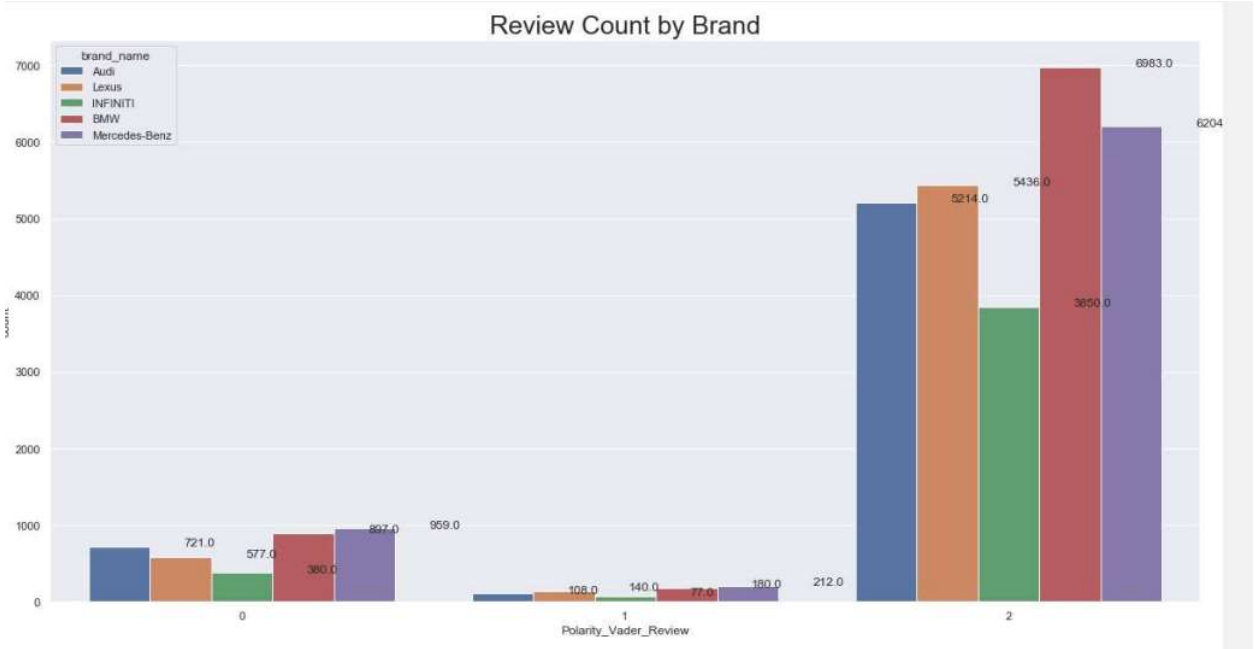
```
0.9061162717879135
```

```
print("Audi:", audi_score)
print("BMW:", BMW_score)
print("Infiniti:", Infiniti_score)
print("Lexus:", Lexus_score)
print("Mercedes-Benz:", Mercedes_score)
```

```
Audi: 0.9061162717879135
BMW: 0.8776745642417284
Infiniti: 0.9141008245485858
Lexus: 0.9010019830915353
Mercedes-Benz: 0.8843022648992799
```

- Figure 17:

- Figure 18:



Citations

J.D. Powers. (n.d.). *Car ratings and awards: J.D. Power Awards and ratings*. Car Ratings and Awards | J.D. Power Awards and Ratings. Retrieved January 22, 2023, from https://www.jdpower.com/cars/ratings

Parida, A. (2021, August 24). *Reviews of 5 car brands*. Kaggle. Retrieved January 22, 2023, from https://www.kaggle.com/datasets/ashisparida/reviews-of-5-car-brands

W3Schools. (n.d.). *Pandas DataFrame describe() Method*. Pandas dataframe describe() method. Retrieved January 22, 2023, from https://www.w3schools.com/python/pandas/ref_df_describe.asp

Pandas isnull() function. w3resource. (n.d.). Retrieved December 6, 2022, from https://www.w3resource.com/pandas/isnull.php#:~:text=The%20isnull()%20function%20is,arrays%2C%20NaT%20in%20datetimelike).&text=Object%20to%20check%20for%20null%20or%20missing%20values

Stratis, Kyle. "Use Sentiment Analysis with Python to Classify Movie Reviews." Real Python, Real Python, 10 Nov. 2022, https://realpython.com/sentiment-analysis-python/.

Ankthon. "Python: Sentiment Analysis Using Vader." GeeksforGeeks, 7 Oct. 2021, https://www.geeksforgeeks.org/python-sentiment-analysis-using-vader/.

Borcan, M. (2020, June 8). TF-IDF explained and python sklearn implementation. Medium. Retrieved January 25, 2023, from https://towardsdatascience.com/tf-idf-explained-and-python-sklearn-implementation-b020c5e83275

W3Schools. (n.d.). Machine learning - logistic regression. Python Machine Learning - Logistic Regression. Retrieved January 25, 2023, from https://www.w3schools.com/python/python_ml_logistic_regression.asp

Navlani, A. (2018, May 16). Sklearn Random Forest classifiers in python tutorial. DataCamp. Retrieved January 25, 2023, from https://www.datacamp.com/tutorial/random-forests-classifier-python

Iriondo, R. (2021, October 22). Natural language processing (NLP) with python - tutorial . Towards AI. Retrieved January 29, 2023, from https://towardsai.net/p/nlp/natural-language-processing-nlp-with-python-tutorial-for-beginners-1f54e610a1a0