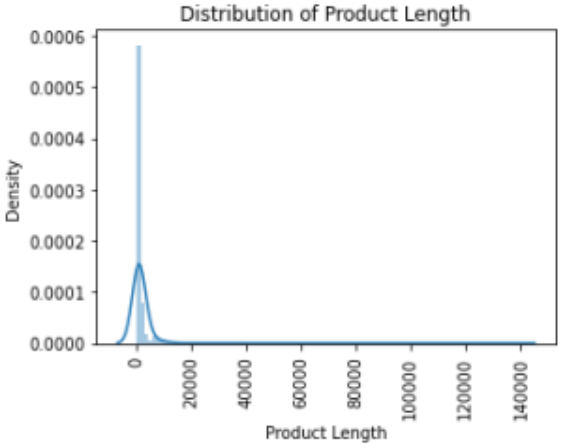
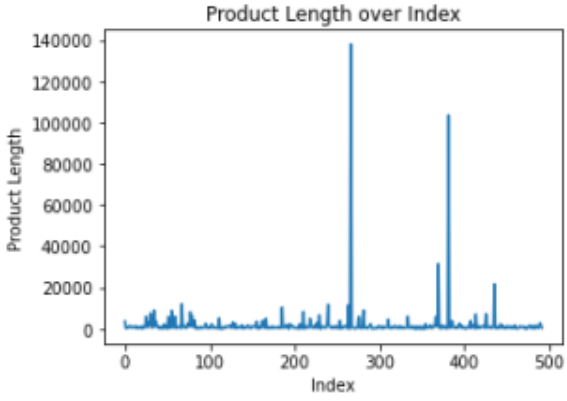
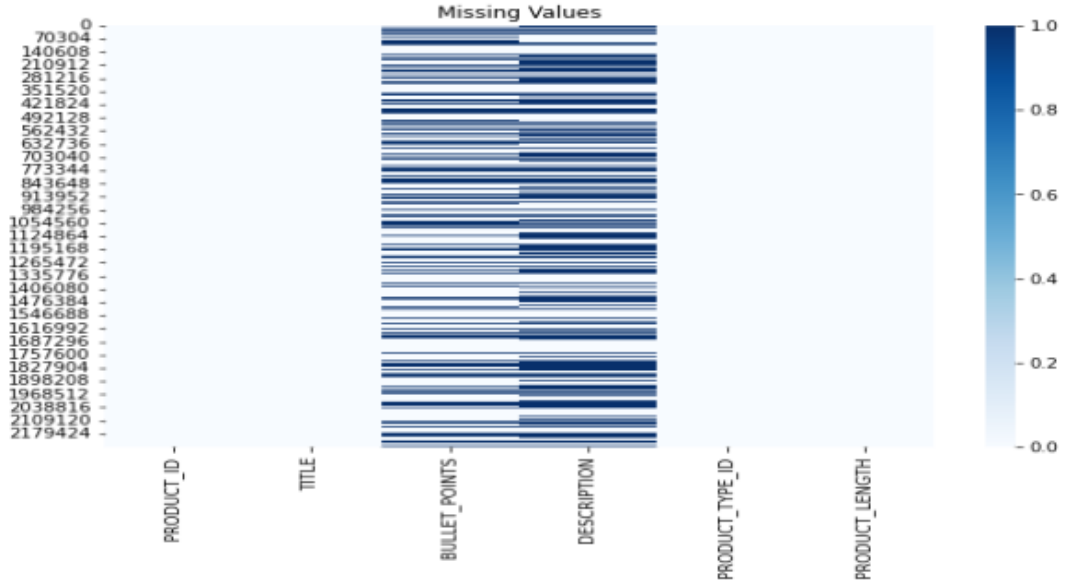
Using NLP to anticipate Amazon Product Length

Data Analysis







Preprocessing the data and storing the vectors

```
import re
nlp = spacy.load("en_core_web_lg")
import gensim.downloader as api
wv = api.load('glove-twitter-200')
```

```
def preprocessing(text):

    text = text.lower()
    text = re.sub(r'<[^>]*>', '', text)
    text = re.sub(r'[^\w+.]',' ', text)
    text = text.strip()
    text = re.sub(r'\s{2,}', ' ', text)

    doc = nlp(text)
    filtered_token = []

    for token in doc:
        if token.is_punct or token.is_stop:
          continue
        filtered_token.append(token.lemma_)

    return(wv.vectors_for_all(filtered_token))
```

```
for i in range(0, len(amazon), 10000):
    start_time = time.time()
    amazon_df = amazon.copy()
    amazon_df = amazon_df[i:i+10000]
    amazon_df = amazon_df.dropna()
    amazon_df.reset_index(drop= True, inplace = True)
    amazon_df = amazon_df.drop(columns = ['PRODUCT_ID','PRODUCT_TYPE_ID'])
    for j in range(3):
        amazon_df.iloc[:,j] = amazon_df.iloc[:,j].apply(lambda x: preprocessing(x))
    amazon_df_len = amazon_df.copy()
    for k in range(3):
        amazon_df_len.iloc[:,k] = amazon_df.iloc[:,k].apply(lambda x: len(x))
    amazon_df = amazon_df[amazon_df_len != 0].dropna()
    for l in range(3):
        amazon_df.iloc[:,l] = amazon_df.iloc[:,l].apply(lambda x: x[0])
    file_name = 'amazon_df_csvs' + '/' + 'amazon_df' + str(i) + '.csv'
    amazon_df.to_csv(file_name)
    end_time = time.time()
    execution_time = end_time - start_time
    print(f"Execution time for {i} to {i+10000}: {execution_time} seconds")
```

```
Execution time for 1550000 to 1560000: 727.2574422359467 seconds
Execution time for 1560000 to 1570000: 680.5712716579437 seconds
Execution time for 1570000 to 1580000: 622.1952383518219 seconds
Execution time for 1580000 to 1590000: 666.6947605609894 seconds
Execution time for 1590000 to 1600000: 680.5627112388611 seconds
Execution time for 1600000 to 1610000: 673.3680160045624 seconds
Execution time for 1610000 to 1620000: 698.5366544723511 seconds
Execution time for 1620000 to 1630000: 698.7736251354218 seconds
```

Fitting the Model

```
X = amazon_df.drop(columns = ['PRODUCT_LENGTH'])
Y = amazon_df.PRODUCT_LENGTH
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size= 0.2, random_state = 2)
```

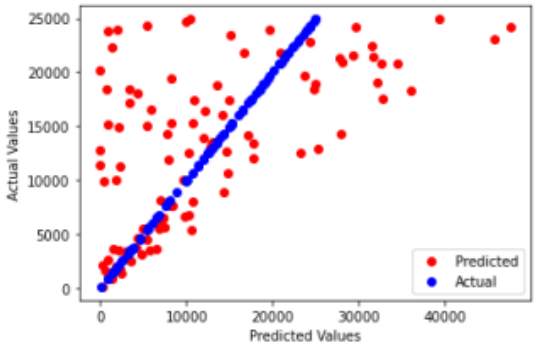
```
reg = Pipeline([
    ('Linear Reg' , LinearRegression())
])
```

```
reg.fit(np.stack(X_train.values.flatten()).reshape(len(X_train),-1), Y_train.values)
```

```
Pipeline(steps=[('Linear Reg', LinearRegression())])
```

```
Y_pred = reg.predict(np.stack(X_test.values.flatten()).reshape(len(X_test), -1))
```

Evaluation




```
Mean Absolute Percentage Error  0.4785999999999999
Mean Square Error  85784670.48944399
```