FDA Recall Health Impact Severity

Moulya Naveena Choday

Daniel Rimdans

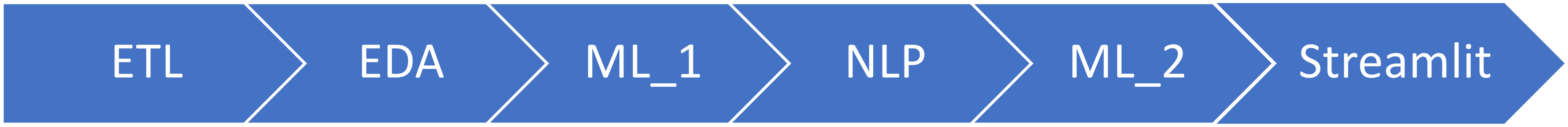
FDA Background

- US FDA -> United States Food & Drug Administration
- Founded 1906
- Agency under Department of Health & Human Services (HHS)
- The FDA regulates food, cosmetics, drugs, medical devices, tobacco, veterinary products, and biologics.
- A primary tool of regulation is **recalls**.

Why Should You Care?

- Do you live in the US and consume food, medication, cosmetics, tobacco, or utilize medical devices?
- If you live outside the US, in 2021 alone, the FDA oversaw **48.1 million imported products**.

Project Structure



ETL

EDA

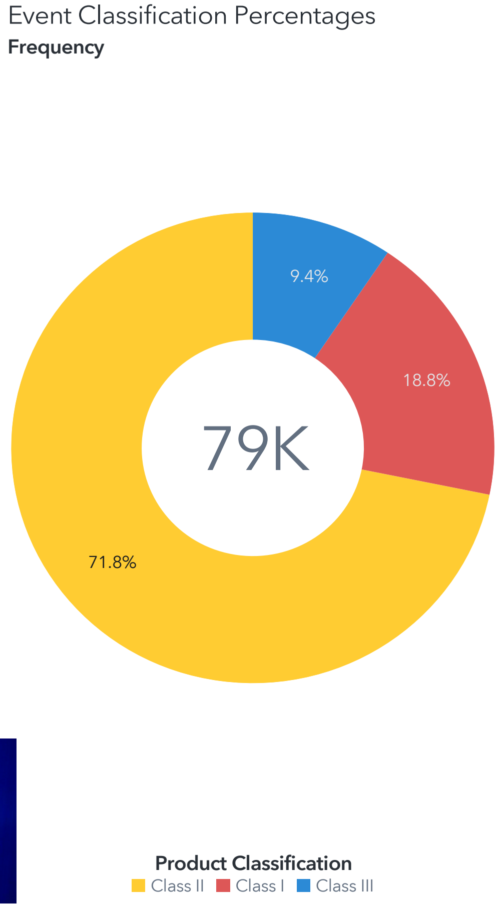
ML_1

NLP

ML_2

Streamlit

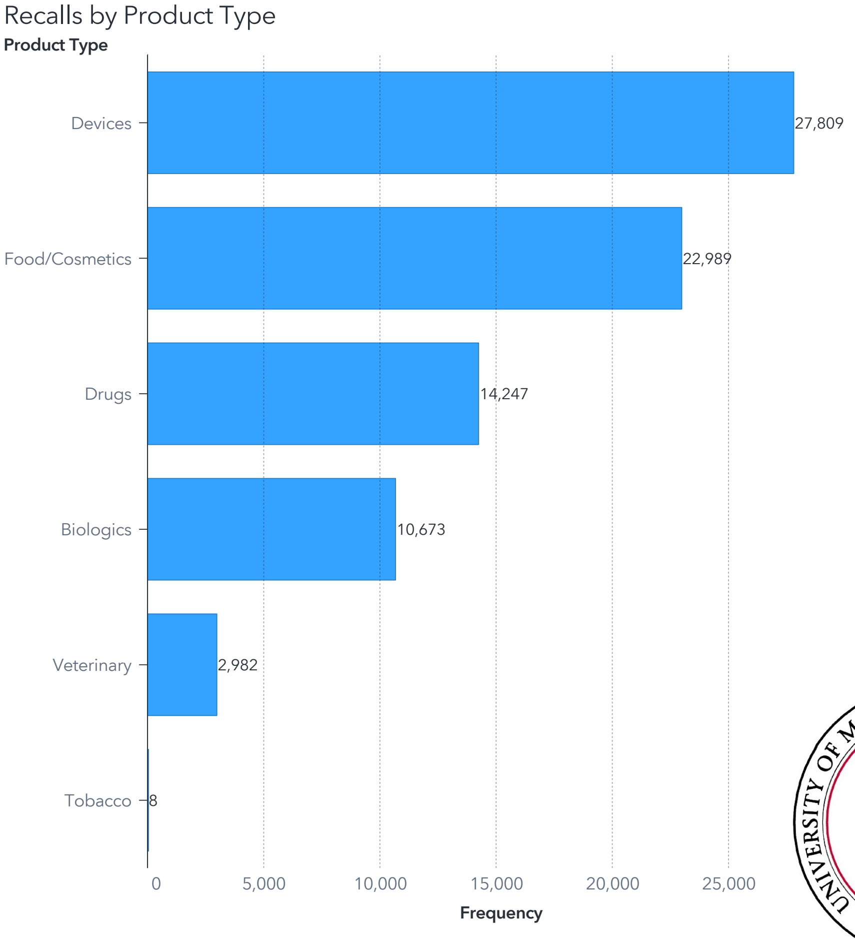
Recalls Data



Event Classification Percentages

Frequency

A1.1

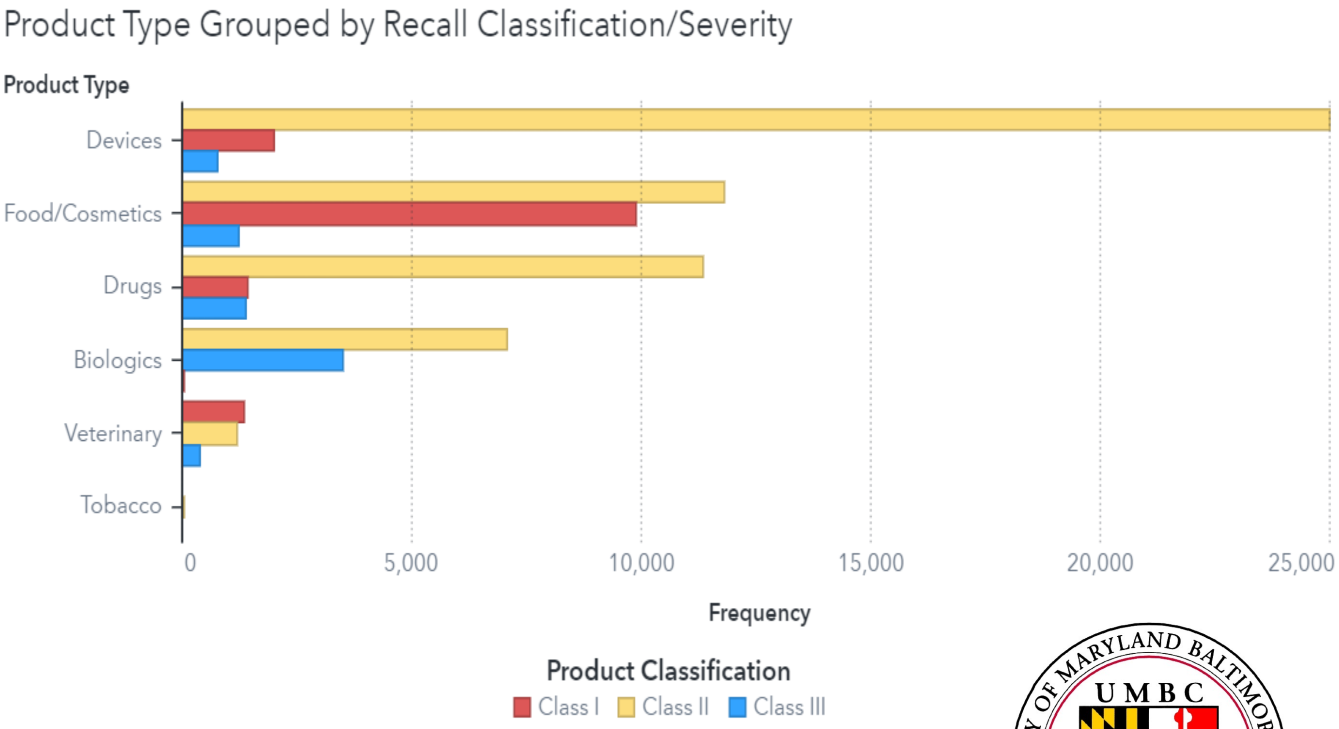


Recalls by Product Type

Product Type

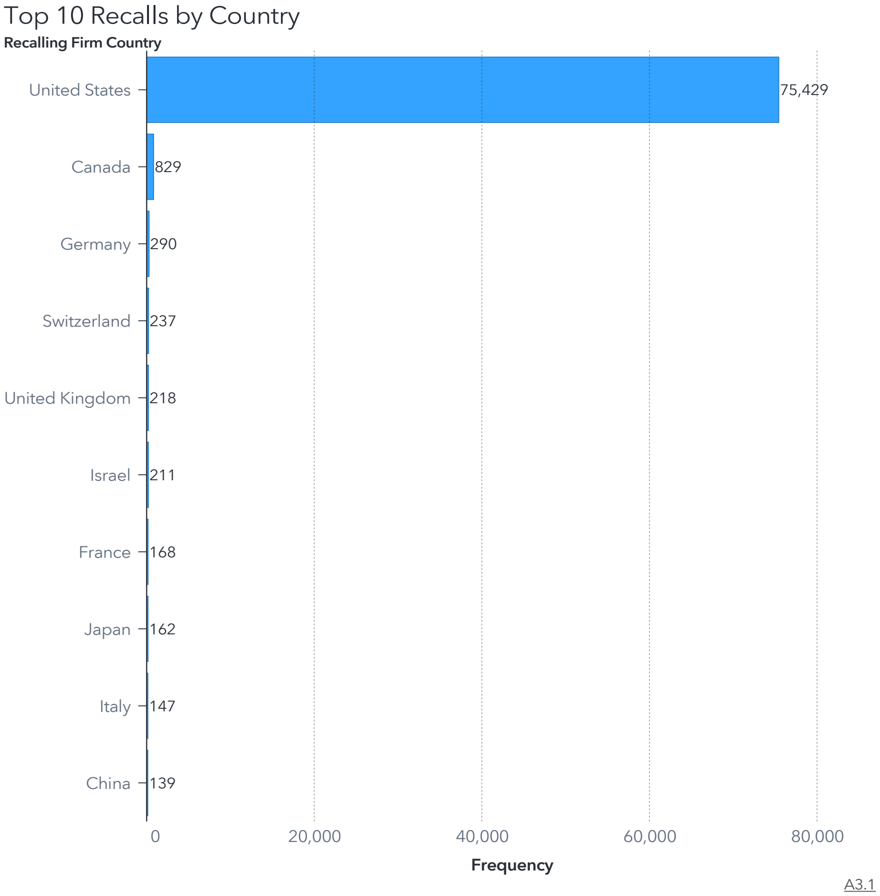
Data EDA Questions

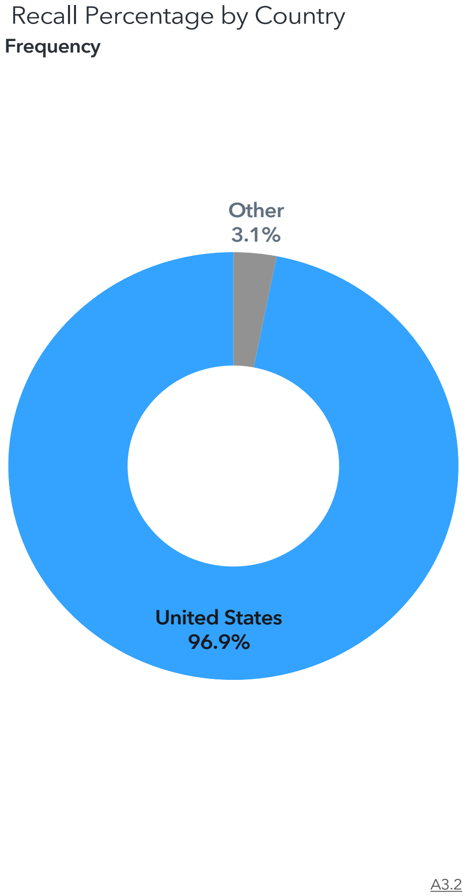
- What product class is being recalled most frequently?
- What class of recalled products has the most severe health impacts (Class I & II)?



Data EDA Questions

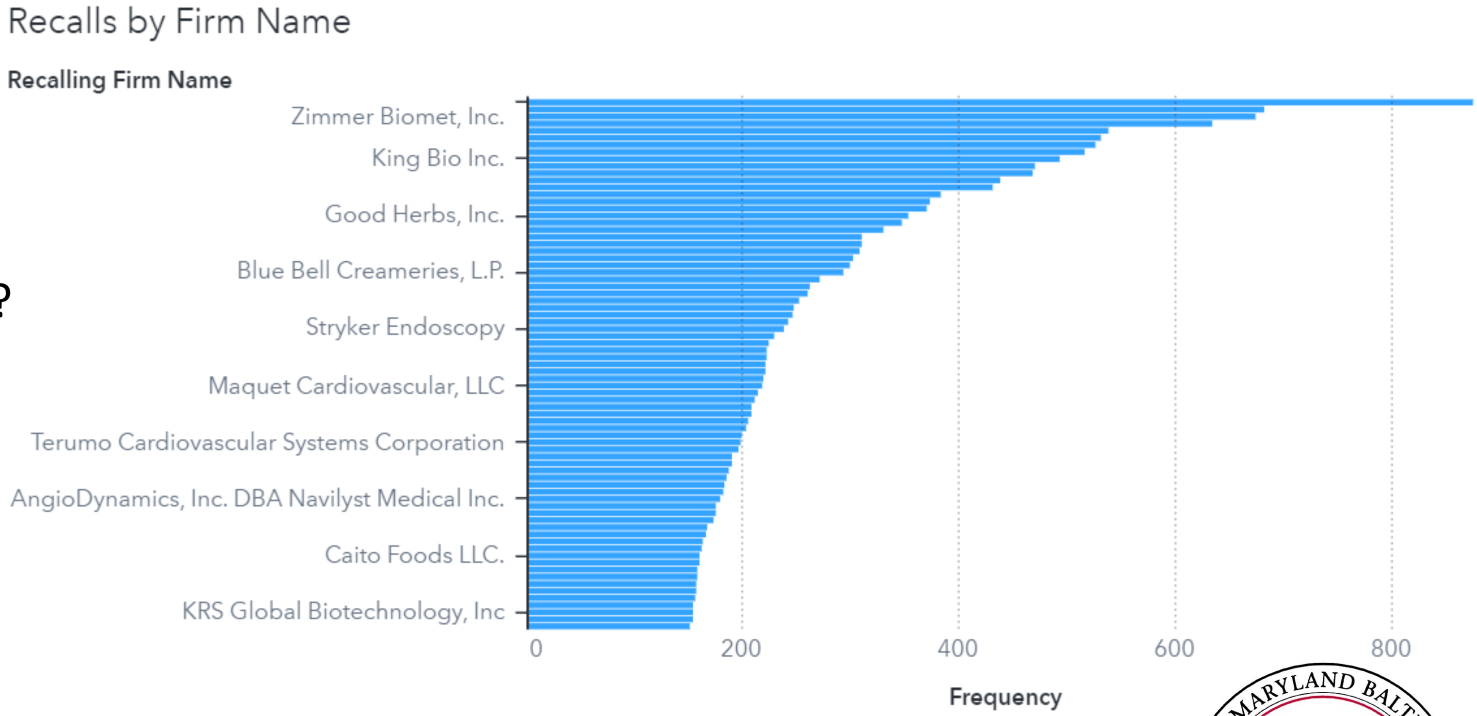
- What countries have the highest recalls?



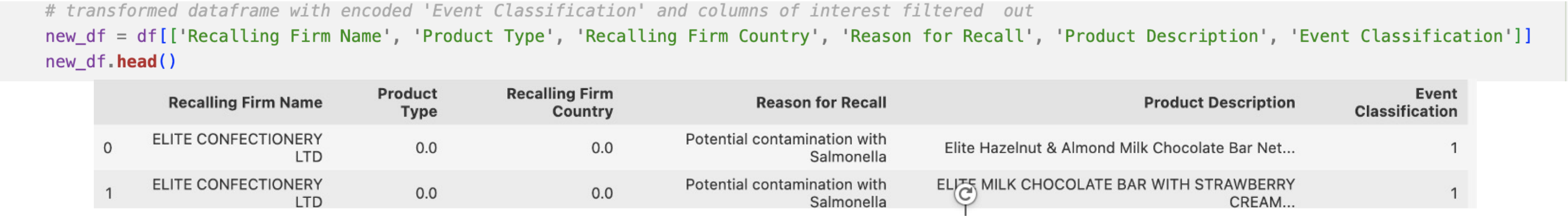


Data EDA Questions

- What manufacturing firms have the highest recalls?

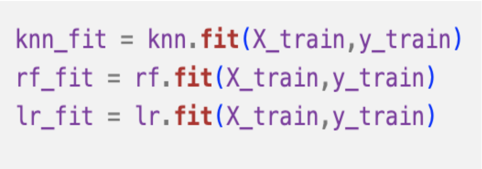


Machine Learning (1)

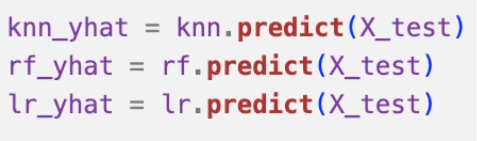


```
# transformed dataframe with encoded 'Event Classification' and columns of interest filtered  out
new_df = df[['Recalling Firm Name', 'Product Type', 'Recalling Firm Country', 'Reason for Recall', 'Product Description', 'Event Classification']]
new_df.head()
```

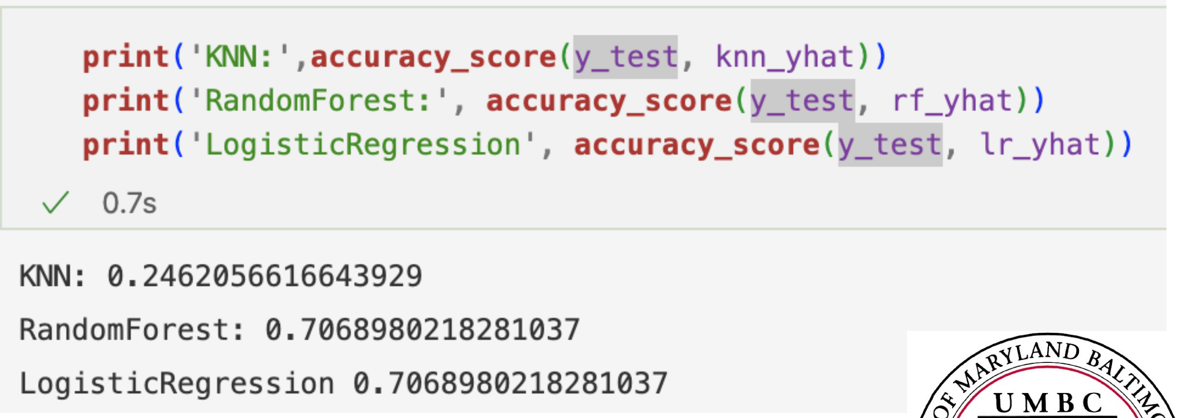
	Recalling Firm Name	Product Type	Recalling Firm Country	Reason for Recall	Product Description	Event Classification
0	ELITE CONFECTIONERY LTD	0.0	0.0	Potential contamination with Salmonella	Elite Hazelnut & Almond Milk Chocolate Bar Net...	1
1	ELITE CONFECTIONERY LTD	0.0	0.0	Potential contamination with Salmonella	ELITE MILK CHOCOLATE BAR WITH STRAWBERRY CREAM...	1



```
knn_fit = knn.fit(X_train,y_train)
rf_fit = rf.fit(X_train,y_train)
lr_fit = lr.fit(X_train,y_train)
```



```
knn_yhat = knn.predict(X_test)
rf_yhat = rf.predict(X_test)
lr_yhat = lr.predict(X_test)
```



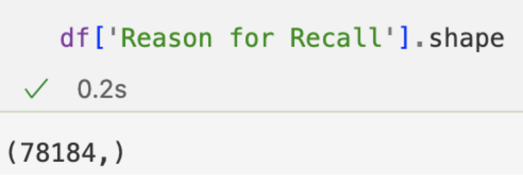
```
print('KNN:',accuracy_score(y_test, knn_yhat))
print('RandomForest:', accuracy_score(y_test, rf_yhat))
print('LogisticRegression', accuracy_score(y_test, lr_yhat))
```

✓ 0.7s

```
KNN: 0.2462056616643929
RandomForest: 0.7068980218281037
LogisticRegression 0.7068980218281037
```

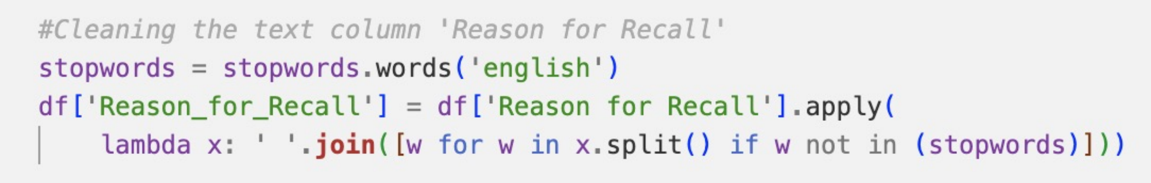
Natural Language Processing (NLP)

df shape before NLP



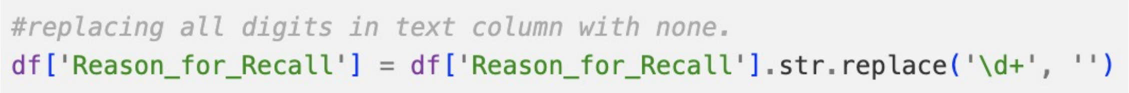
```
df['Reason for Recall'].shape
✓ 0.2s
(78184,)
```

Removing stopwords



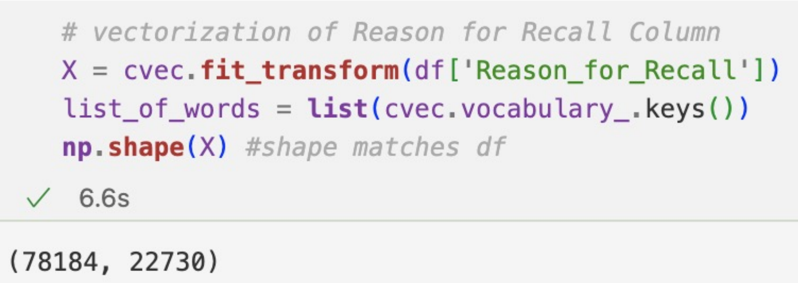
```
#Cleaning the text column 'Reason for Recall'
stopwords = stopwords.words('english')
df['Reason_for_Recall'] = df['Reason for Recall'].apply(
    lambda x: ' '.join([w for w in x.split() if w not in (stopwords)]))
```

Removing digits



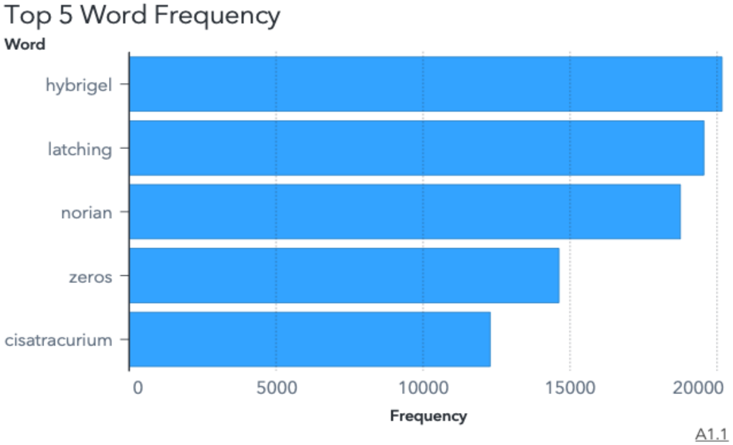
```
#replacing all digits in text column with none.
df['Reason_for_Recall'] = df['Reason_for_Recall'].str.replace('\d+', '')
```

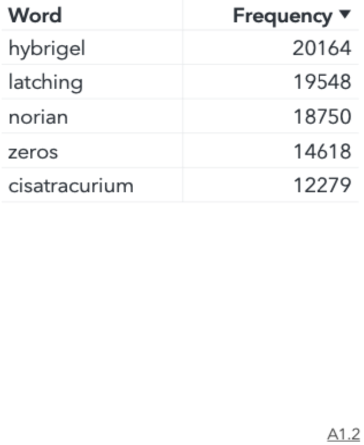
Vectorization



```
# vectorization of Reason for Recall Column
X = cvec.fit_transform(df['Reason_for_Recall'])
list_of_words = list(cvec.vocabulary_.keys())
np.shape(X) #shape matches df
✓ 6.6s
(78184, 22730)
```

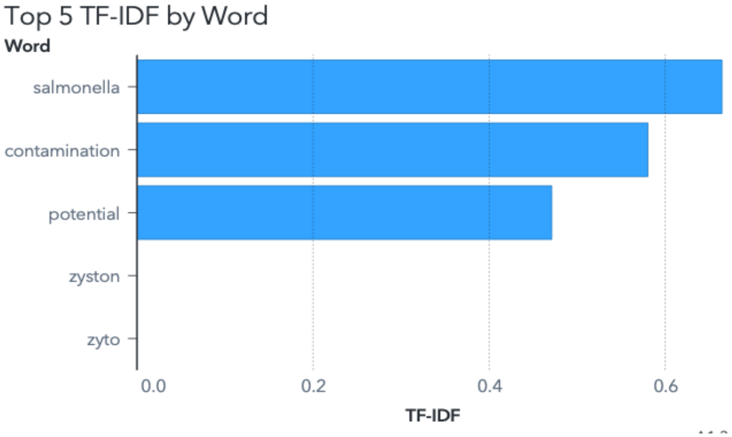
Top 5 Word/Feature Importance

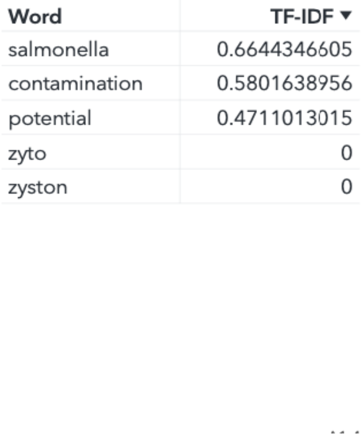




Word	Frequency ▼
hybrigel	20164
latching	19548
norian	18750
zeros	14618
cisatracurium	12279

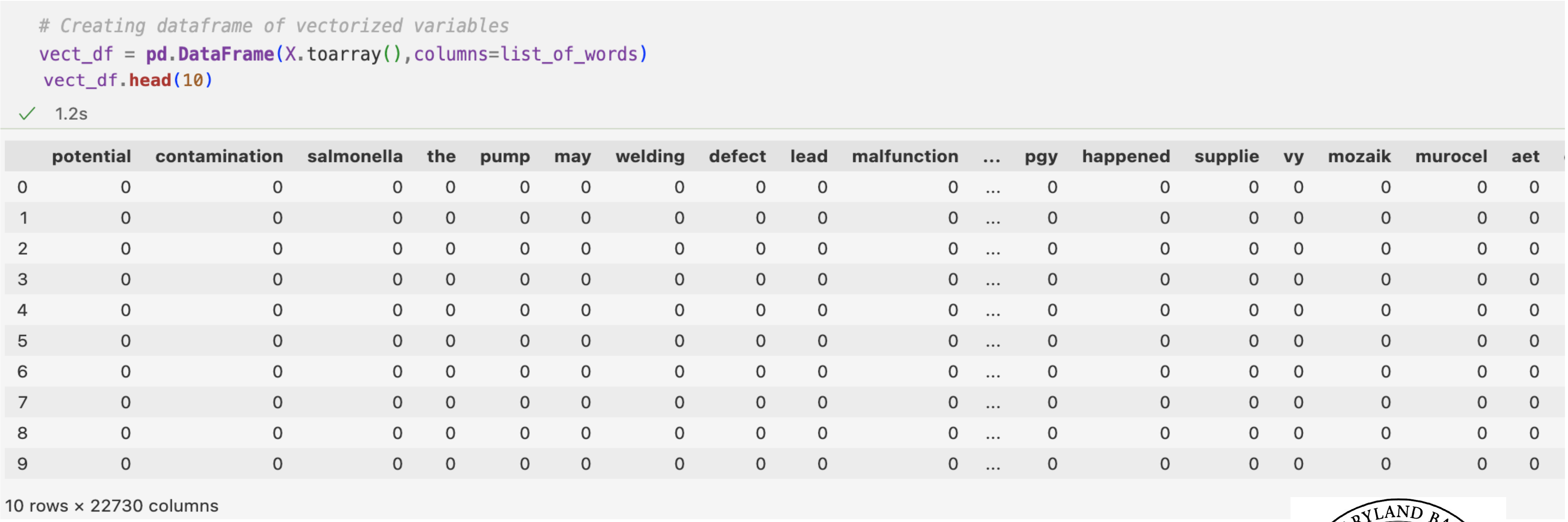
A1.2





Word	TF-IDF ▼
salmonella	0.6644346605
contamination	0.5801638956
potential	0.4711013015
zyto	0
zyston	0

NLP Output Dataframe



```
# Creating dataframe of vectorized variables
vect_df = pd.DataFrame(X.toarray(),columns=list_of_words)
 vect_df.head(10)
```

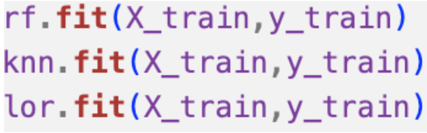
✓ 1.2s

	potential	contamination	salmonella	the	pump	may	welding	defect	lead	malfunction	...	pgy	happened	supplie	vy	mozaik	murocel	aet
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0

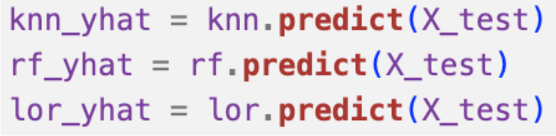
10 rows × 22730 columns

X data

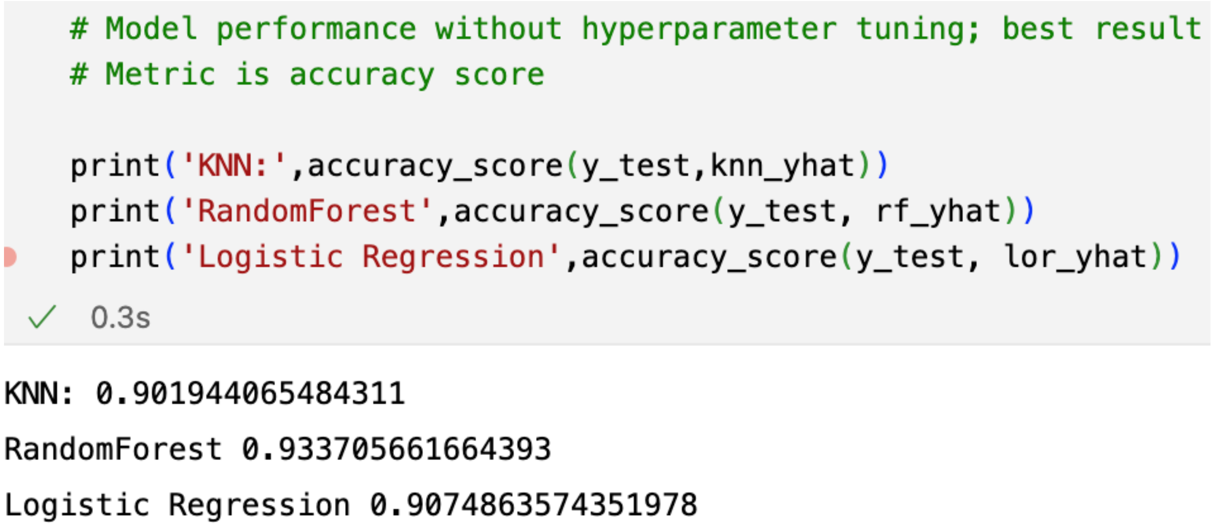
Machine Learning…….again (2) after NLP



```
rf.fit(X_train,y_train)
knn.fit(X_train,y_train)
lor.fit(X_train,y_train)
```



```
knn_yhat = knn.predict(X_test)
rf_yhat = rf.predict(X_test)
lor_yhat = lor.predict(X_test)
```



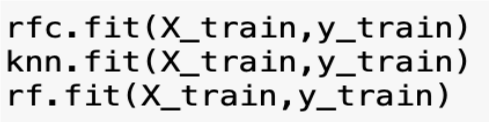
```
# Model performance without hyperparameter tuning; best result
# Metric is accuracy score

print('KNN:',accuracy_score(y_test,knn_yhat))
print('RandomForest',accuracy_score(y_test, rf_yhat))
print('Logistic Regression',accuracy_score(y_test, lor_yhat))
```

✓ 0.3s

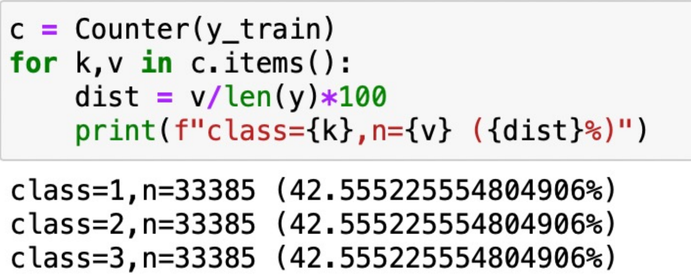
```
KNN: 0.901944065484311
RandomForest 0.933705661664393
Logistic Regression 0.9074863574351978
```

Hyperparameter Tuning

- Grid search with cross validation.
- Balancing the data using Synthetic Minority Over-sampling Technique
  (SMOTE)

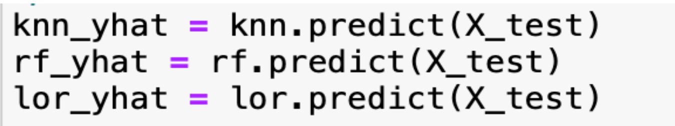


```
rfc.fit(X_train,y_train)
knn.fit(X_train,y_train)
rf.fit(X_train,y_train)
```

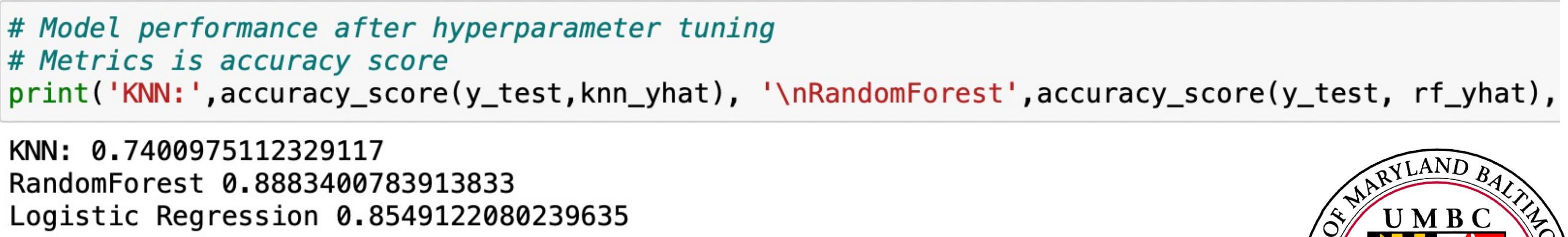


```
c = Counter(y_train)
for k,v in c.items():
    dist = v/len(y)*100
    print(f"class={k},n={v} ({dist}%)")
```

```
class=1,n=33385 (42.555225554804906%)
class=2,n=33385 (42.555225554804906%)
class=3,n=33385 (42.555225554804906%)
```



```
knn_yhat = knn.predict(X_test)
rf_yhat = rf.predict(X_test)
lor_yhat = lor.predict(X_test)
```



```
# Model performance after hyperparameter tuning
# Metrics is accuracy score
print('KNN:',accuracy_score(y_test,knn_yhat), '\nRandomForest',accuracy_score(y_test, rf_yhat),
```

```
KNN: 0.7400975112329117
RandomForest 0.8883400783913833
Logistic Regression 0.8549122080239635
```

Streamlit Web Application

- Creating simple web applications using python programming language.
- Fastest way to create and deploy Web applications for Machine Learning and data science projects.

https://naveenachodayy-team-e-data606-streamlitproject-1kygzy.streamlitapp.com/

Github Repository Walkthrough

- https://github.com/aminrimdans/Naveena_Daniel_Data606
- https://github.com/NaveenaChodayy/TEAM_E_Data606

References

- U.S. Food & Drug Administration. (n.d.). FDA Dashboards—Recalls. Compliance Dashboards. Retrieved June 12, 2022, from https://datadashboard.fda.gov/ora/cd/recalls.htm
- Wikimedia Foundation. (2022, May 11). Food and Drug Administration. Wikipedia. Retrieved June 12, 2022, from https://en.wikipedia.org/wiki/Food_and_Drug_Administration
- https://docs.streamlit.io/library/get-started
- https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

Questions