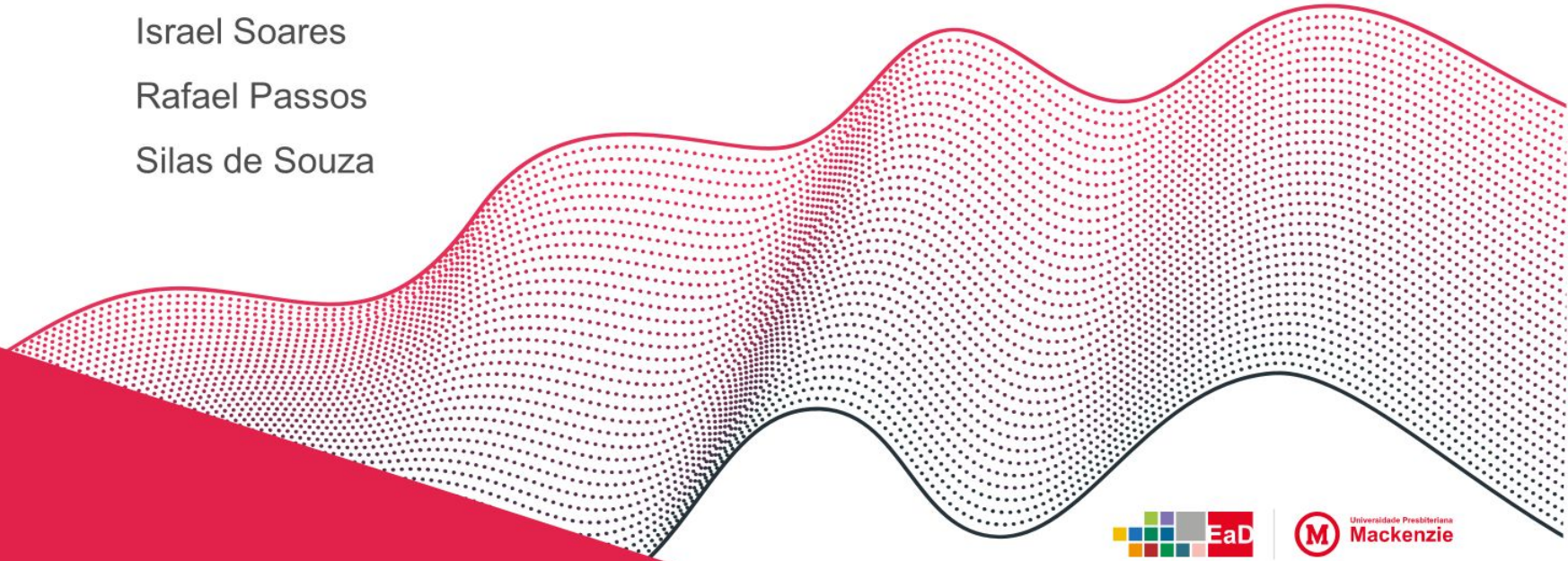
Gustavo Rios
Israel Soares
Rafael Passos
Silas de Souza



EaD
Universidade Presbiteriana Mackenzie

STORYTELLING

- **No cenário atual é evidente a abundância de informações, porém, nem sempre adaptadas às necessidades individuais.**
- **Desafio: Como oferecer aos usuários uma experiência informativa alinhada com suas preferências individuais?**

- **Exemplos:**



STORYTELLING

- **O projeto está relacionado ao desenvolvimento do sistema de classificação e recomendação de notícias.**
- **Objetivo principal: automatizar a classificação de conteúdo informativo e proporcionar uma experiência personalizada para cada usuário.**

STORYTELLING

- **Personalização da Experiência do Usuário.**
- **Preferências individuais como: entretenimento, política, estilo, esportes.**
- **Uso do Feedback do usuário na melhoria contínua do sistema.**

- **Conclusão**:

A Ciência de Dados é o guia para a jornada pela informação personalizada, revelando novas perspectivas e horizontes inexplorados no vasto universo da informação.

O futuro é de evolução constante, de desbravamento de novas fronteiras para as informações personalizadas no meio digital.

Entendimento dos Dados

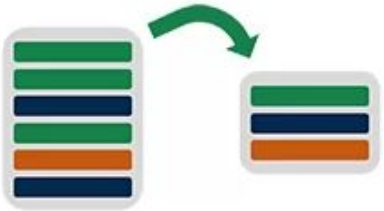
```
[2]: df = pd.read_json('News_Category_Dataset.json', lines=True)
```

```
[4]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209527 entries, 0 to 209526
Data columns (total 6 columns):
 #   Column             Non-Null Count   Dtype         
---  ------             --------------   -----         
 0   link               209527 non-null  object        
 1   headline           209527 non-null  object        
 2   category           209527 non-null  object        
 3   short_description  209527 non-null  object        
 4   authors            209527 non-null  object        
 5   date               209527 non-null  datetime64[ns]
dtypes: datetime64[ns](1), object(5)
memory usage: 9.6+ MB
```

- Os dados com as informações das notícias foram lidos em formato JSON;
- As *Features* são as informações descritivas das notícias e o *Label* é a categoria em que elas pertencem;
- O objetivo final deste projeto será a modelagem de algoritmos de classificação que consigam a partir das informações das notícias prever em quais categorias elas pertencem.

Tratamento dos Dados



- Foi realizado o tratamento dos dados nulos e duplicados;

Substituir strings vazias ou espaços em branco por 'Not Specified' usando replace

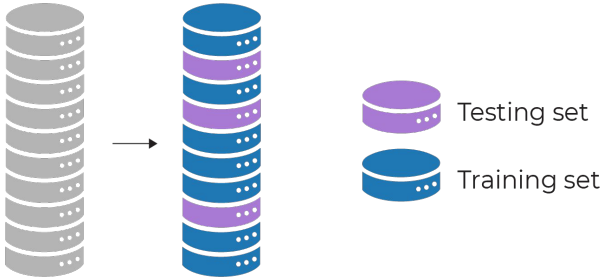
```
In [14]: df.replace({'': 'Not Specified', ' ': 'Not Specified'}, inplace=True)

         # Verificando se as mudanças ocorreram corretamente.
         for col in df.columns:
             # Verifica se a coluna é do tipo object, o que normalmente indica strings
             if df[col].dtype == 'object':
                 # Conta quantas vezes os valores são strings vazias ou espaços em branco
                 empty_values = df[df[col].str.strip() == ''].shape[0]
                 print(f"Número de '{col}' com valores não especificados (strings vazias ou espaços): {empty_values}")

         Número de 'link' com valores não especificados (strings vazias ou espaços): 0
         Número de 'headline' com valores não especificados (strings vazias ou espaços): 0
         Número de 'category' com valores não especificados (strings vazias ou espaços): 0
         Número de 'short_description' com valores não especificados (strings vazias ou espaços): 0
         Número de 'authors' com valores não especificados (strings vazias ou espaços): 0
         Número de 'website' com valores não especificados (strings vazias ou espaços): 0
```

- Foi realizado uma técnica de imputação com um valor fixo ‘Not Specified’ nos valores que estavam em branco.

Divisão em Treino e Teste



4. Divisão dos dados em Treino e Teste;

In [27]:
```
# Vamos usar apenas as colunas 'headline' e 'short_description'
# como features e 'category' como target
df['text'] = df['headline'] + ' ' + df['short_description']

# Limpeza básica do texto pode ser expandida conforme necessário
df['text'] = df['text'].str.lower().str.replace('[^\w\s]', '')
```

In [28]:
```
from sklearn.model_selection import train_test_split
```

In [29]:
```
# Variáveis Independentes
X = df['text']

# Variáveis Dependentes
y = df['category']
```

In [30]:
```
# Dividindo em Treino e Teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- Foram instanciadas as variáveis dependentes e as independentes;
- Os dados foram divididos em treino e teste;
- 80% dos dados para treino e 20% para teste.

Random Forest Classifier

In [39]:
```
from sklearn.ensemble import RandomForestClassifier

# Criando e treinando o modelo Random Forest
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train_vect, y_train)
```

Out[39]:
```
RandomForestClassifier(random_state=42)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

- O Random Forest é um algoritmo de Machine Learning supervisionado;
- É baseado na construção de múltiplas árvores de decisão em subconjuntos aleatórios na busca de reduzir a correlação entre elas e maximizar o ganho de informação.
- Foram criadas 100 árvores *(n_estimators=100)* como as da imagem ao lado.

Random Forest Classifier

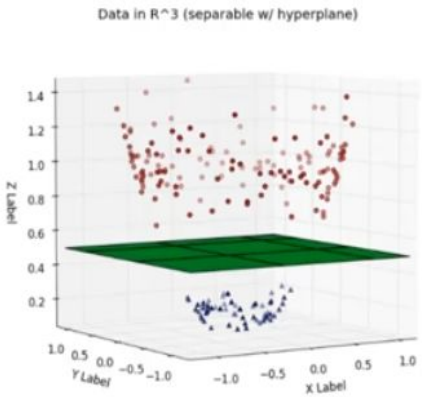
Fazendo previsões e avaliando o modelo

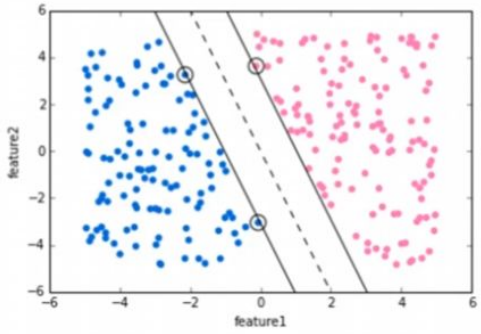
```
In [40]: rf_y_pred = rf_model.predict(X_test_vect)
         print(classification_report(y_test, rf_y_pred))
```

	precision	recall	f1-score	support
ARTS	0.54	0.16	0.25	310
ARTS & CULTURE	0.31	0.10	0.15	255
BLACK VOICES	0.45	0.22	0.30	911
BUSINESS	0.49	0.35	0.41	1176
COLLEGE	0.48	0.27	0.35	226
COMEDY	0.45	0.32	0.37	1044
CRIME	0.45	0.44	0.44	725
CULTURE & ARTS	0.86	0.21	0.34	285
DIVORCE	0.82	0.67	0.74	689
EDUCATION	0.36	0.13	0.19	196
ENTERTAINMENT	0.51	0.71	0.59	3499
ENVIRONMENT	0.89	0.09	0.16	279
FIFTY	0.46	0.07	0.12	256
FOOD & DRINK	0.51	0.74	0.61	1290
GOOD NEWS	0.40	0.07	0.11	260
GREEN	0.40	0.16	0.23	536
HEALTHY LIVING	0.32	0.23	0.26	1342
HOME & LIVING	0.66	0.61	0.63	875
IMPACT	0.46	0.07	0.12	711
LATINO VOICES	0.95	0.08	0.15	220
MEDIA	0.64	0.24	0.35	586
MONEY	0.56	0.15	0.24	356
PARENTING	0.48	0.61	0.53	1733
PARENTS	0.36	0.24	0.29	778
POLITICS	0.56	0.90	0.69	6992
QUEER VOICES	0.79	0.61	0.69	1242
...				
accuracy			0.55	41903
macro avg	0.55	0.34	0.38	41903
weighted avg	0.55	0.55	0.51	41903

- Após treinado, o modelo foram criadas previsões com os dados de teste;
- Na previsão das categorias algumas se destacaram como '*DIVORCE'* com uma precisão de 0.82 e recall de 0.67 e '*STYLE & BEAUTY*' com precisão de 0.70 e recall de 0.77;
- Porém a acurácia geral do modelo foi de 0.55 indicando uma alta variação de desempenho em determinadas categorias que não são facilmente distinguíveis com base nas features utilizadas no treino.

Support Vector Machines






```
In [41]: from sklearn.feature_extraction.text import TfidfVectorizer
         # Vetorização dos textos usando TF-IDF
         vectorizer = TfidfVectorizer(stop_words='english', max_features=1000)
         X_train_tfidf = vectorizer.fit_transform(X_train)
         X_test_tfidf = vectorizer.transform(X_test)

In [42]: from sklearn.svm import SVC
         # Criando e treinando o modelo SVM
         svm_model = SVC(kernel='linear', C=1.0)
         svm_model.fit(X_train_tfidf, y_train)

Out[42]: SVC(kernel='linear')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

- É um modelo supervisionado;
- Os dados de treinamento são representados como pontos em um espaço multidimensional e o objetivo é encontrar um hiperplano que maximize a margem entre as diferentes categorias;
- Os pontos que tocam as margens são chamados ‘*support vectors*’ e definem a margem de separação das classes;
- O modelo classifica novos dados com base em qual lado do gap eles estão localizados.

Support Vector Machines

In [43]:
```
from sklearn.metrics import classification_report

# Fazendo previsões e avaliando o modelo
svm_y_pred = svm_model.predict(X_test_tfidf)
print(classification_report(y_test, svm_y_pred))
```

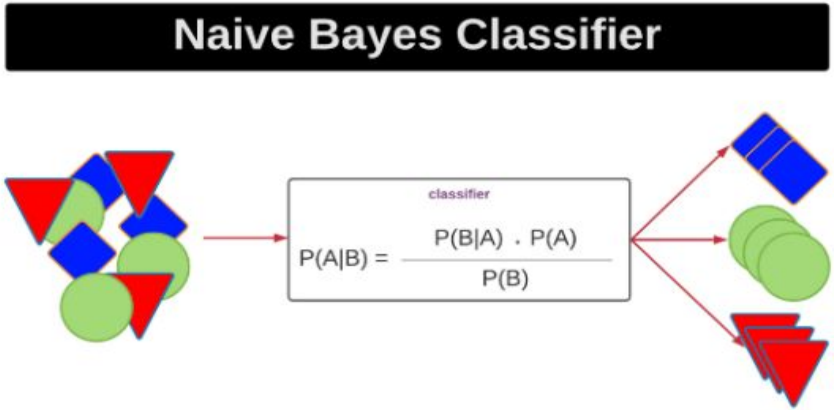
	precision	recall	f1-score	support
ARTS	0.33	0.09	0.14	310
ARTS & CULTURE	0.26	0.13	0.17	255
BLACK VOICES	0.40	0.24	0.30	911
BUSINESS	0.38	0.34	0.36	1176
COLLEGE	0.38	0.36	0.37	226
COMEDY	0.42	0.24	0.30	1044
CRIME	0.41	0.44	0.43	725
CULTURE & ARTS	0.34	0.13	0.19	205
DIVORCE	0.74	0.61	0.67	689
EDUCATION	0.32	0.31	0.32	196
ENTERTAINMENT	0.38	0.61	0.47	3499
ENVIRONMENT	0.45	0.15	0.22	279
FIFTY	0.34	0.05	0.08	256
FOOD & DRINK	0.48	0.56	0.52	1290
GOOD NEWS	0.24	0.08	0.12	260
GREEN	0.29	0.17	0.21	536
HEALTHY LIVING	0.35	0.13	0.19	1342
HOME & LIVING	0.56	0.52	0.54	875
IMPACT	0.29	0.13	0.18	711
LATINO VOICES	0.33	0.00	0.01	220
MEDIA	0.43	0.20	0.27	586
MONEY	0.32	0.24	0.27	356
PARENTING	0.44	0.67	0.53	1733
PARENTS	0.40	0.08	0.14	778
POLITICS	0.60	0.81	0.69	6992

...

accuracy			0.49	41903
macro avg	0.42	0.31	0.33	41903
weighted avg	0.48	0.49	0.46	41903

- Obtivemos uma acurácia geral de 0.49, o que indica que cerca de metade das previsões do modelo estavam corretas;
- Algumas categorias tiveram uma melhor performance como por exemplo *‘POLITICS’* com precisão de 60% e recall de 81%;
- Já a categoria ‘*U.S News’* teve a pior performance com 20% de precisão e recall quase nulo, indicando uma falta de características distintivas para esta categoria.

Naive Bayes




```
In [35]: from sklearn.naive_bayes import MultinomialNB

In [36]: # Treinando o modelo Naive Bayes
         naive_model = MultinomialNB()
         naive_model.fit(X_train_vect, y_train) # (X_train vetorizado, true label)

Out[36]: MultinomialNB()
         In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
         On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [37]: from sklearn.metrics import classification_report

In [38]: # Fazendo previsões no conjunto de teste
         y_pred = naive_model.predict(X_test_vect)

         # Avaliando o modelo
         print(classification_report(y_test, y_pred, zero_division=0))
```

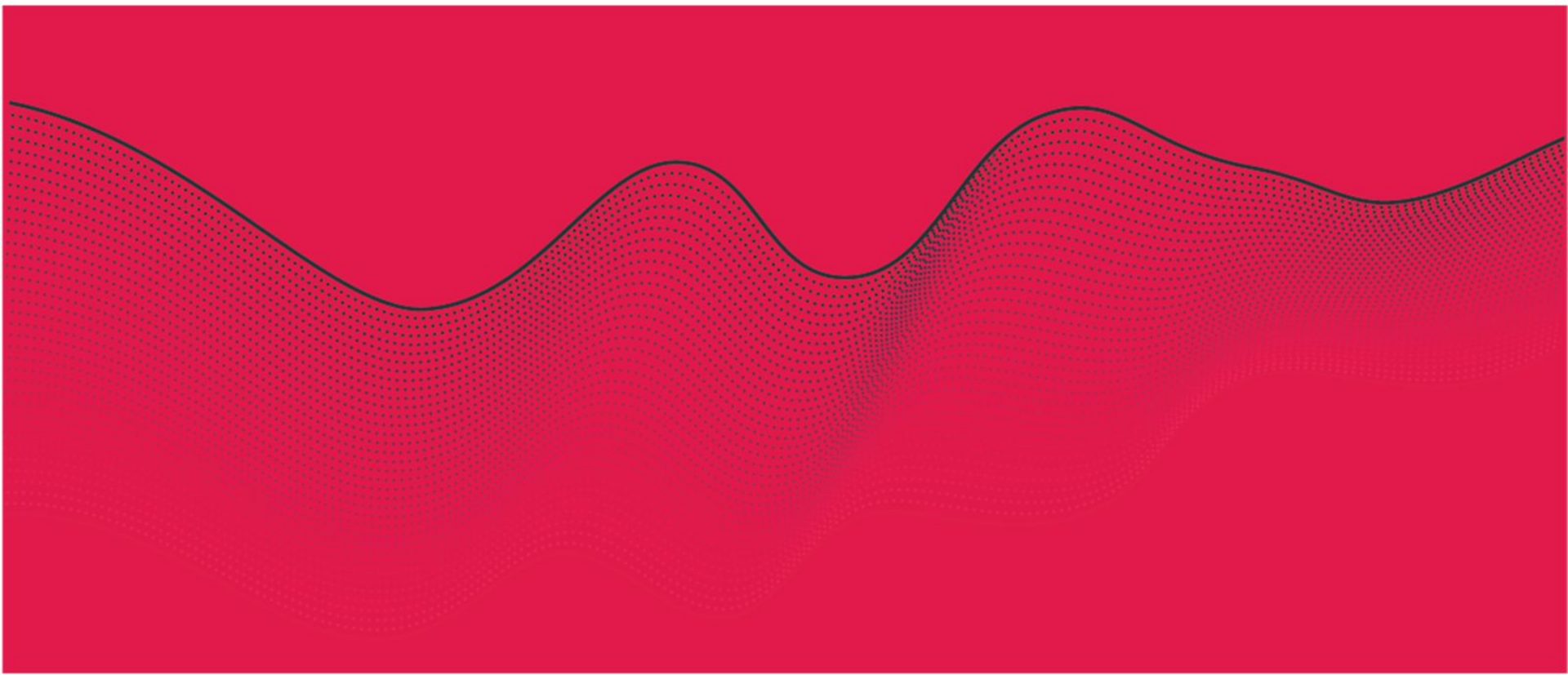
- Naive Bayes é um algoritmo criado para classificar os dados e assim torná-los disponíveis para otimizar o processo de tomada de decisão.
- Consiste em calcular a probabilidade de um evento ocorrer, e após calculada essa probabilidade, ela é relacionada a outro evento.
- O algoritmo apenas classifica em duas classes de probabilidade, alta ou baixa, e a partir daí é gerada uma tabela de probabilidades, onde o modelo separa e classifica essas informações.

Naive Bayes

	precision	recall	f1-score	support
ARTS	0.71	0.05	0.10	310
ARTS & CULTURE	0.67	0.02	0.03	255
BLACK VOICES	0.60	0.20	0.30	911
BUSINESS	0.47	0.41	0.44	1176
COLLEGE	0.73	0.04	0.07	226
COMEDY	0.61	0.23	0.33	1044
CRIME	0.50	0.52	0.51	725
CULTURE & ARTS	0.76	0.11	0.19	205
DIVORCE	0.88	0.51	0.65	689
EDUCATION	0.67	0.03	0.06	196
ENTERTAINMENT	0.50	0.81	0.62	3499
ENVIRONMENT	0.76	0.09	0.16	279
FIFTY	0.00	0.00	0.00	256
FOOD & DRINK	0.62	0.74	0.67	1290
GOOD NEWS	0.62	0.03	0.06	260
GREEN	0.45	0.15	0.23	536
HEALTHY LIVING	0.56	0.18	0.28	1342
HOME & LIVING	0.82	0.57	0.67	875
IMPACT	0.49	0.17	0.25	711
LATINO VOICES	1.00	0.02	0.04	220
MEDIA	0.74	0.13	0.22	586
MONEY	0.60	0.10	0.18	356
PARENTING	0.41	0.68	0.51	1733
PARENTS	0.60	0.07	0.12	778
POLITICS	0.57	0.92	0.70	6992
QUEER VOICES	0.72	0.51	0.60	1242
RELIGION	0.80	0.22	0.34	533
SCIENCE	0.73	0.27	0.40	400
SPORTS	0.73	0.61	0.67	976
STYLE	0.78	0.01	0.03	501
STYLE & BEAUTY	0.67	0.83	0.74	1978
TASTE	0.43	0.01	0.01	437
TECH	0.81	0.17	0.27	411
THE WORLDPOST	0.60	0.38	0.46	760
TRAVEL	0.57	0.81	0.67	2072
U.S. NEWS	0.80	0.01	0.03	269
WEDDINGS	0.87	0.58	0.70	756
WEIRD NEWS	0.66	0.09	0.16	564
WELLNESS	0.47	0.87	0.61	3634
WOMEN	0.69	0.12	0.20	712
WORLD NEWS	0.58	0.24	0.34	665
WORLDPOST	0.59	0.16	0.26	543
accuracy			0.56	41903
macro avg	0.64	0.30	0.33	41903
weighted avg	0.59	0.56	0.50	41903

- O modelo em geral apresentou uma acurácia média, cerca de 0.56 de acertos, havendo algum destaque para categorias em específico

  - Em destaque ficam as categorias ***Entertainment*** com precisão de 0.50 e recall de 0.81, ***Politics*** precisão de 0.57 e recall de 0.92
    e por fim a categoria ***Style e Beauty*** que possui maior equilíbrio com precisão de 0.67 e recall de 0.83



EaD

Universidade Presbiteriana
Mackenzie