

# Data Mining (Week 1)

## MSc - Data Mining

### Topic 06 : Classification

#### Part 03 : Naive Bayes

Dr Bernard Butler and Dr Kieran Murphy

Department of Computing and Mathematics, WIT.  
(bbutler@tssg.org; kmurphy@wit.ie)

Spring Semester, 2021

#### Foundation

Data Handling

Exploratory Data  
Analysis

Data Modelling  
Fundamentals

Data Modelling  
Advanced

#### Rule Based

Association Rules

Clustering

#### Supervised

Regression

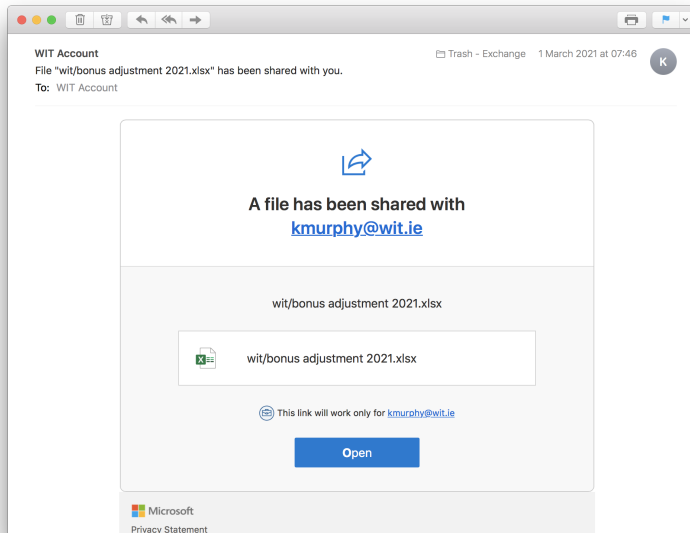
#### Outline

- Motivation behind naive Bayes
- Sample application
- Summary of properties

Deep Learning

# Spam Filtering

## Reality



## Simplified Problem

Assume that we have the following set of messages previously classified as **spam** or **ham**.

Message	Class
"send us your password"	spam
"send us your review"	ham
"password review"	ham
"review us"	spam
"send your password"	spam
"send us your account"	spam

We are interested in classifying the following new message as **spam** or **ham**:

Message	Class
"review us now"	?

# Count occurrences ... compute probabilities

Message	Class
“send us your password”	spam
“send us your review”	ham
“password review”	ham
“review us”	spam
“send your password”	spam
“send us your account”	spam

Word	# in spam	# in ham	$\Pr(\bullet \text{spam})$	$\Pr(\bullet \text{ham})$
account	1	0	1/4	0/2
password	2	1	2/4	1/2
review	1	2	1/4	2/2
send	3	1	3/4	1/2
us	3	1	3/4	1/2
your	3	1	3/4	1/2

Class	Count	Probability
spam	4	$\Pr(\text{spam}) = 4/6$
ham	2	$\Pr(\text{ham}) = 2/6$

prior probabilities

## What we have

The probability that a message contains, say the word **review**, among our message classed as **spam**, i.e.,

$$\Pr(\text{review}|\text{spam}) = 1/4$$

## What we want

The probability that a message is **spam** given that it contains, say the word **review**, i.e.,

$$\Pr(\text{spam}|\text{review}) = ?$$

posterior probabilities

## Aside: Probability Laws

### Bayes Rule

$$\overbrace{\Pr(A|B) \Pr(B)}^{\text{conditional} \times \text{marginal}} = \overbrace{\Pr(A \text{ AND } B)}^{\text{joint}} = \overbrace{\Pr(B|A) \Pr(A)}^{\text{conditional} \times \text{marginal}}$$

$$\Pr(A|B) \Pr(B) = \Pr(B|A) \Pr(A) \longrightarrow \Pr(B|A) = \frac{\Pr(A|B) \Pr(B)}{\Pr(A)}$$

### Complementary Rule

$$\Pr(\text{NOT } A) = 1 - \Pr(A)$$

### Independent Events

If events  $A$  and  $B$  are independent, then  $\Pr(A \text{ AND } B) = \Pr(A) \Pr(B)$

### Law of total of probability

Let  $A$  be any event. Let  $\{B_1, B_2, B_3, \dots, B_n\}$  be a set of events, exactly one of which must occur, then

$$\Pr(A) = \Pr(A|B_1) \Pr(B_1) + \Pr(A|B_2) \Pr(B_2) + \dots + \Pr(A|B_n) \Pr(B_n)$$

## Classifying a Message Using a Single Word

Applying the law of total probability, with  $A = \text{review}$ , and  $B_1 = \text{spam}$  and  $B_2 = \text{ham}$ , then we have

$$\Pr(\text{review}) = \Pr(\text{review}|\text{spam}) \Pr(\text{spam}) + \Pr(\text{review}|\text{ham}) \Pr(\text{ham}) = \frac{1}{4} \cdot \frac{4}{6} + \frac{2}{2} \cdot \frac{2}{6} = \frac{3}{6}$$

Now we can apply Bayes rule, with  $A = \text{review}$  and  $B = \text{spam}$  we have

$$\Pr(\text{spam}|\text{review}) = \frac{\Pr(\text{review}|\text{spam}) \Pr(\text{spam})}{\Pr(\text{review})} = \frac{\frac{1}{4} \cdot \frac{4}{6}}{\frac{3}{6}} = \frac{1}{3} = 33.3\%$$

And we can apply Bayes rule, with  $A = \text{review}$  and  $B = \text{ham}$  to get

$$\Pr(\text{ham}|\text{review}) = \frac{\Pr(\text{review}|\text{ham}) \Pr(\text{ham})}{\Pr(\text{review})} = \frac{\frac{2}{2} \cdot \frac{2}{6}}{\frac{3}{6}} = \frac{2}{3} = 66.7\%$$

So on receiving a message containing the word **review** we can classify it a **spam** with probability 33.3% and **ham** with probability 66.7%.

# Classifying a Message Using Multiple Words

I

- We can now (hopefully) do this for each word in our test message (“review us now”), but what about classifying using multiple words? ... here comes the naïve bit ...

Naïve Bayes assumes that presence of each word are independent events

Recall: independent events means can multiply to get joint probabilities.

- We are interested in classifying the message

review us now

- We don't have data on the word now so only looking at messages containing review and us and not containing account, password, send, or your.

$$\Pr(\{\text{review, us}\}|\text{spam}) = \underbrace{\left(1 - \frac{1}{4}\right)}_{\text{account}} \underbrace{\left(1 - \frac{2}{4}\right)}_{\text{password}} \underbrace{\left(\frac{1}{4}\right)}_{\text{review}} \underbrace{\left(1 - \frac{3}{4}\right)}_{\text{send}} \underbrace{\left(\frac{3}{4}\right)}_{\text{us}} \underbrace{\left(1 - \frac{3}{4}\right)}_{\text{your}} = 0.0044$$

and

$$\Pr(\{\text{review, us}\}|\text{ham}) = \underbrace{\left(1 - \frac{0}{2}\right)}_{\text{account}} \underbrace{\left(1 - \frac{1}{2}\right)}_{\text{password}} \underbrace{\left(\frac{2}{2}\right)}_{\text{review}} \underbrace{\left(1 - \frac{1}{2}\right)}_{\text{send}} \underbrace{\left(\frac{1}{2}\right)}_{\text{us}} \underbrace{\left(1 - \frac{1}{2}\right)}_{\text{your}} = 0.0625$$

# Classifying a Message Using Multiple Words

Now we can apply the law of total probability as before

$$\begin{aligned}\Pr(\{\text{review}, \text{us}\}) &= \Pr(\{\text{review}, \text{us}\}|\text{spam}) \Pr(\text{spam}) + \Pr(\{\text{review}, \text{us}\}|\text{ham}) \Pr(\text{ham}) \\ &= 0.0044 \left(\frac{4}{6}\right) + 0.0625 \left(\frac{2}{6}\right) = 0.0237\end{aligned}$$

And finally Bayes rule

$$\Pr(\text{spam}|\{\text{review}, \text{us}\}) = \frac{\Pr(\{\text{review}, \text{us}\}|\text{spam}) \Pr(\text{spam})}{\Pr(\{\text{review}, \text{us}\})} = \frac{0.0044 \left(\frac{4}{6}\right)}{0.0237} = 0.123$$

and

$$\Pr(\text{ham}|\{\text{review}, \text{us}\}) = \frac{\Pr(\{\text{review}, \text{us}\}|\text{ham}) \Pr(\text{ham})}{\Pr(\{\text{review}, \text{us}\})} = \frac{0.0625 \left(\frac{2}{6}\right)}{0.0237} = 0.877$$

Hence, the probability that the message is **spam** is 12.3%, and **ham** is 87.7%.

# Classifying a Message Using Multiple Words

A final comment:

- The classifier picks the class with the largest probability, so in this case

$$\Pr(\text{spam}|\{\text{review}, \text{us}\}) = \frac{\Pr(\{\text{review}, \text{us}\}|\text{spam}) \Pr(\text{spam})}{\Pr(\{\text{review}, \text{us}\})} = \frac{0.0044 \left(\frac{4}{6}\right)}{0.0237} = 0.123$$

and

$$\Pr(\text{ham}|\{\text{review}, \text{us}\}) = \frac{\Pr(\{\text{review}, \text{us}\}|\text{ham}) \Pr(\text{ham})}{\Pr(\{\text{review}, \text{us}\})} = \frac{0.0625 \left(\frac{2}{6}\right)}{0.0237} = 0.877$$

- We can skip the computation of  $\Pr(\{\text{review}, \text{us}\})$  involving the law of total probability step since that is a common denominator in above formula, And instead look are relative likelihood

$$\frac{\Pr(\text{spam}|\{\text{review}, \text{us}\})}{\Pr(\text{ham}|\{\text{review}, \text{us}\})} = \frac{\Pr(\{\text{review}, \text{us}\}|\text{spam}) \Pr(\text{spam})}{\Pr(\{\text{review}, \text{us}\}|\text{ham}) \Pr(\text{ham})} \implies \begin{cases} > 1 & \text{pick spam} \\ < 1 & \text{pick ham} \end{cases}$$



# Trump's Claims — Dataset

The Washington Post has released a curated list of Donald Trump's false claims while in office. The claim's have been categorised and cross linked. We will see how well a naïve Bayes classifier will do on this dataset.

```
df = pd.read_csv("wapo_trumpclaims_export-012021.csv.gz")
print(df.shape)
df.head(5)
```

```
(30573, 9)
```

	id	location	claim	analysis	pinocchios	category	repeated_ids	repeated_count	date
0	31608.0	Remarks	"We also got tax cuts, the largest tax cut and...	This is Trump's second favorite falsehood, and...	4.0	Taxes	31608, 31581, 31305, 31183, 31530, 30920, 3085...	296	01/20/2021
1	31609.0	Remarks	"We just got seventy five million votes. And t...	When the counting was finished, Trump had rece...	NaN	Election	31609, 31292, 31155, 31016, 31082, 30992, 3156...	19	01/20/2021
2	31610.0	Remarks	"One of the things we're very, very proud of i...	Contrary to his boasts, Trump did not achieve ...	NaN	Miscellaneous	31610, 31598, 31187, 30918, 30374, 29845, 2922...	84	01/20/2021
3	31611.0	Remarks	"Our first lady has been a woman of great grac...	In reality, Melania Trump leaves the White Hou...	NaN	Miscellaneous	NaN	0	01/20/2021
4	31612.0	Remarks	"That's why [regulation cuts] we have such goo...	Leaving aside Trump's claim about the impact o...	NaN	Jobs	NaN	0	01/20/2021

# Trump's Claims — Dataset

```
df.category.value_counts(dropna=False)
```

Immigration	3225
Foreign policy	3165
Election	3037
Miscellaneous	2767
Coronavirus	2521
Trade	2513
Economy	2475
Russia	1838
Jobs	1732
Health care	1629
Ukraine probe	1377
Environment	1065
Biographical record	963
Taxes	857
Crime	852
NaN	169
Guns	165
Education	151
Terrorism	72

Name: category, dtype: int64

General observations ...

- Some classes are relatively rare.
- Some classifications would appear to overlap Russia ↔ Election.
- There are 169 unclassified observations, missing values ... **these need** to be removed/encoded before model building.
- Total of 30,573 false claims! He was busy between golf sessions ... sorry, I could not resist.

# Trump's Claims — Data Preprocessing

## II

We perform our standard train-test split

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    df.claim, df.category, test_size=0.2, random_state=42)
```

Next we need to reproduce the word extraction and counts that we did for the simple example earlier:

```
from sklearn.feature_extraction.text import CountVectorizer
count_vectorizer = CountVectorizer()

X_train_counts = count_vectorizer.fit_transform(X_train)
X_test_counts = count_vectorizer.transform(X_test)
```

Use the train dataset to determine parameters for the operation (**fit**). Then, apply (**transform**) to both train and test

Notice the function calls `fit_transform` on `X_train` and `transform` on `X_test`. The function `fit_transform` is two separate steps: `fit` and `transform` and could be written as

```
count_vectorizer.fit(X_train)
X_train_counts = count_vectorizer.transform(X_train)
```

or

```
X_train_counts = count_vectorizer.fit(X_train).transform(X_train)
```

## How many features?

```
X_test_counts.shape
```

 $(6081, 12718)$ 

```
print(count_vectorizer.get_feature_names()[:1000])
```

0th', '000', '005', '006', '00pm', '00pme', '024', '024th', '028', '03', '036', '067', '085', '09', '0ver', '10', '100', '1000', '100th', '101', '102', '1024', '1024th', '103', '104', '1040', '105', '106', '107', '108', '109', '10s', '10th', '11', '110', '112', '113', '115', '116', '117', '118', '119', '11th', '12', '120', '121', '122', '123', '125', '126', '127', '128', '129', '12th', '13', '130', '132', '133', '134', '135', '135th', '138', '139', '13th', '14', '140', '141', '142', '143', '144', '144th', '145', '147', '149', '1492', '14th', '15', '150', '150th', '151', '154', '155', '157', '158', '15th', '16', '160', '161', '162', '164', '168', '169', '16th', '17', '170', '171', '172', '175', '176', '1776', '17b', '18', '180', '1807', '181', '182', '1824', '183', '184', '185', '1860', '187', '188', '1888', '1890', '1890s', '18s', '18th', '19', '190', '1900s', '191', '1912', '1913', '1917', '1918', '192', '1928', '1929', '1930s', '195', '1940s', '195', '1950', '1950s', '1951', '1952', '196', '1960', '1960s', '1962', '197', '1972', '1974', '1976', '1977', '198', '1980', '1980s', '1982', '1986', '1990', '1992', '1994', '1996', '1997', '19th', '1k', '1liskyevng', '1st', '20', '200', '2000', '2001', '2006', '2008', '2009', '201', '2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019', '2020', '2020election', '2021', '2024', '2025', '2026', '2028', '2030', '2032', '2035', '204', '205', '206', '207', '209', '20k', '20s', '20th', '21', '210', '2100', '212', '215', '216', '218', '21st', '22', '220', '221', '223', '224', '225', '227', '228', '22nd', '23', '230', '232', '235', '237', '238', '24', '240', '243', '245', '25', '250', '251', '252', '253', '256', '257', '259', '25th', '26', '266', '27', '270', '273', '275', '276', '277', '279', '28', '280', '281', '285', '287', '289', '29', '290', '292', '293', '295', '297', '29th', '2a', '2k', '2nd', '2t', '2go40', '2yrs', '30', '300', '3000', '302', '304', '306', '307', '30s', '30th', '31', '311', '312', '315', '31am', '31st', '32', '320', '325', '326', '33', '330', '331', '337', '33rd', '34', '340', '341', '342', '346', '348', '35', '350', '3500', '355', '356', '358', '35s', '36', '360', '368', '37', '375', '379', '38', '380', '39', '392', '3c3rnj4pwt', '3rd', '40', '400', '4000', '401', '401k', '401ks', '40m', '40s', '41', '410', '412', '415', '42', '420', '424', '426', '42am', '43', '435', '44', '440', '45', '450', '46', '462', '464', '468', '47', '470', '475', '48', '480', '49', '490', '495', '4s', '4th', '50', '500', '5000', '500th', '502', '504', '505', '506', '507', '51', '510', '517', '52', '520', '525', '527', '529', '53', '530', '533', '537', '539', '54', '546', '55', '550', '552', '556', '56', '565', '566', '57', '572', '575', '58', '59', '593', '5g', '5l', '5th', '5x', '60', '600', '6000', '603', '608', '60minutes', '60s', '6', '618', '61st', '62', '620', '623', '63', '635', '64', '640', '643', '65', '650', '6500', '654', '658', '66', '663', '67', '670', '675', '68', '69', '690', '6th', '70', '700', '7000', '701', '70s', '71', '716', '72', '722', '725', '73', '730', '738', '739', '74', '747', '747s', '75', '750', '76', '77', '772', '774', '78', '79', '790', '792', '7yrs', '80', '800', '8000', '807', '80s', '81', '817', '82', '83', '84', '85', '850', '85th', '86', '863', '87', '876', '88', '880', '886', '89', '8th', '8x', '90', '900', '9000', '904', '90s', '91', '92', '922', '923', '93', '94', '940', '941', '945', '95', '959', '96', '97', '972', '975', '978', '98', '99', '991', '9th', 'a17', 'a35k9g3d', 'abandon', 'aband ned', 'abbott', 'abc', 'abcworldnews', 'abe', 'abedin', 'abeyance', 'abiding', 'ability', 'abject', 'ablaze', 'able', 'abolish', 'abolished', 'abolishice', 'abolishing', 'abortion', 'abortions', 'abound', 'about', 'above', 'abraham', 'a rams', 'abroad', 'abruptly', 'absence', 'absentee', 'absentia', 'absolute', 'absolutely', 'absurd', 'abu', 'abundant', 'abuse', 'abused', 'abuser', 'abuses', 'abusing', 'abusive', 'aca', 'academy', 'accelerate', 'accept', 'acceptable', 'acceptance', 'accepted', 'accepting', 'access', 'accessible', 'accident', 'accidentally', 'accidents', 'acclaimed', 'accompanied', 'accompanying', 'accomplish', 'accomplished', 'accomplishment', 'accomplishments', 'accord', 'according', 'accounts', 'account', 'accountability', 'accountable', 'accountants', 'accounted', 'accounting', 'accounting', 'accounts', 'accumulated', 'accumulating', 'accuracy', 'accurate', 'accurately', 'accusation', 'accusations', 'accused', 'accuser', 'accuse s', 'acusing', 'aced', 'achieve', 'achieved', 'achievement', 'achievements', 'achieving', 'acid', 'acknowledge', 'acknowledged', 'acknowledgment', 'aclu', 'acosta', 'acquire', 'acquiring', 'acquitall', 'acres', 'across', 'act', 'acted', 'acting', 'action', 'actions', 'activate', 'activated', 'active', 'actively', 'activism', 'activist', 'activists', 'activities', 'activity', 'acts', 'actual', 'actuality', 'actually', 'acu', 'acuity', 'ad', 'adam', 'adamaxalt', 'add', 'added', 'addict', 'addicted', 'addiction', 'adding', 'addition', 'additional', 'additionally', 'additions', 'address', 'addressed', 'addresses', 'addressing', 'adept', 'adhere', 'adjoirn', 'adjoirnement', 'adjudged', 'adjudicate', 'adjud cation', 'adjust', 'adjusted', 'admin', 'administer', 'administering', 'administra', 'administration', 'administrations', 'admins', 'admiral', 'admirer', 'admission', 'admissions', 'admit', 'admits', 'admitted', 'admitting', 'admonished', 'adolph', 'adopted', 'adopting', 'adoption', 'ads', 'adult', 'adults', 'advance', 'advanced', 'advances', 'advancing', 'advantage', 'advantageous', 'advantages', 'adversaries', 'adversely', 'advertisements', 'advertising', 'advic', 'advise', 'advised', 'advisers', 'advising', 'advisor', 'advisors', 'advisory', 'advocate', 'advocated', 'advocating', 'aeroplanes', 'aetna', 'af', 'affair', 'affairs', 'affect', 'affected', 'affecting', 'affection', 'affectionately', 'af ects', 'affh', 'affidavits', 'affirm', 'affirmative', 'affirmatively', 'affirmed', 'afflicted', 'afford', 'affordable', 'affront', 'afghanistan', 'afghans', 'afraid', 'africa', 'african', 'after', 'aftermath', 'afternoon', 'afterwards', 'ag', 'again', 'against', 'age', 'agencies', 'agency', 'agenda', 'agent', 'agents', 'ages', 'aggression', 'aggressive', 'aggressively', 'agitated', 'agitators', 'ago', 'agony', 'agree', 'agreed', 'agreeing', 'agreement', 'agreements', 'agrees', 'agricultural', 'agriculture', 'ah', 'ahead', 'aid', 'aide', 'aides', 'aids', 'allment', 'allments', 'aim', 'aimed', 'aiming', 'ain', 'ainsleyehardt', 'aint', 'air', 'airborne', 'airbus', 'aircraft', 'airlifted', 'airline', 'a rlines', 'airplane', 'airplanes', 'airport', 'airports', 'airtime', 'airwaves', 'ajitpaifcc', 'ak', 'al', 'alabama', 'alan', 'alandersh', 'alarmed', 'alaska', 'albaghaddi', 'alcindor', 'alcohol', 'alec', 'alert', 'alerted', 'alerting', 'alex', 'alexander', 'alexandria', 'alfonse', 'alfred', 'ali', 'alice', 'alien', 'aliens', 'alike', 'alito', 'alive', 'all', 'allegation', 'allegations', 'allege', 'alleged', 'allegedly', 'allegheiny', 'allegiance', 'alleging', 'allentown', 'alli', 'alliance', 'alliances', 'allies', 'alligators', 'allocated', 'allow', 'allowed', 'allowing', 'allows', 'alluding', 'ally', 'allyn', 'almost', 'alone', 'along', 'alongside', 'aloud', 'already', 'also', 'altar', 'alter', 'alter d', 'alternative', 'alternatives', 'although', 'altogether', 'aluminum', 'always', 'am', 'amash', 'amased', 'amateur', 'amateurs', 'amaze', 'amazed', 'amazing', 'amazingly', 'amazon', 'amazons', 'ambassador', 'ambassadors', 'amberjack', 'ambition', 'ambitious', 'amend', 'amendment', 'amer', 'america', 'american', 'americans', 'amicable', 'amid', 'ammunition', 'amnesty', 'among', 'amorphous', 'amount', 'amounts', 'amt', 'amuck', 'amy', 'amylklobuchar', 'an', 'analogous', 'analysis', 'analyst', 'analyze', 'analyzed', 'analyzing', 'anarchist', 'anarchists', 'anarchy', 'anchor', 'anchors', 'ancient', 'and', 'andrew', 'andy', 'angela', 'angeles', 'angels', 'anger', 'angered', 'angle', 'angriest', 'angrily', 'angry', 'anguish', 'animal', 'animals', 'animated', 'animosity', 'annapolis', 'anncoultter', 'annie', 'annihilate', 'annihilated', 'announce', 'announced', 'announcement', 'announcements', 'announcers', 'announces', '

# Trump's Claims — Model Fit, Prediction and Evaluation

I

As before\* we import our model and create an instance of it

```
from sklearn.naive_bayes import MultinomialNB  
model = MultinomialNB()
```

Fit our model using the train dataset

```
model.fit(X_train_counts, y_train)
```

Generate predictions using the test dataset

```
y_pred = model.predict(X_test_counts)
```

How well did it do? ... using accuracy score we have

```
from sklearn.metrics import accuracy_score  
accuracy_score(y_test, y_pred)
```

```
0.7763525735898701
```

This is not bad ... there are lots of class levels and some overlap between classes.

---

\*This is the beauty of sklearn — consistent interface to all models, pre-/post- processing steps.

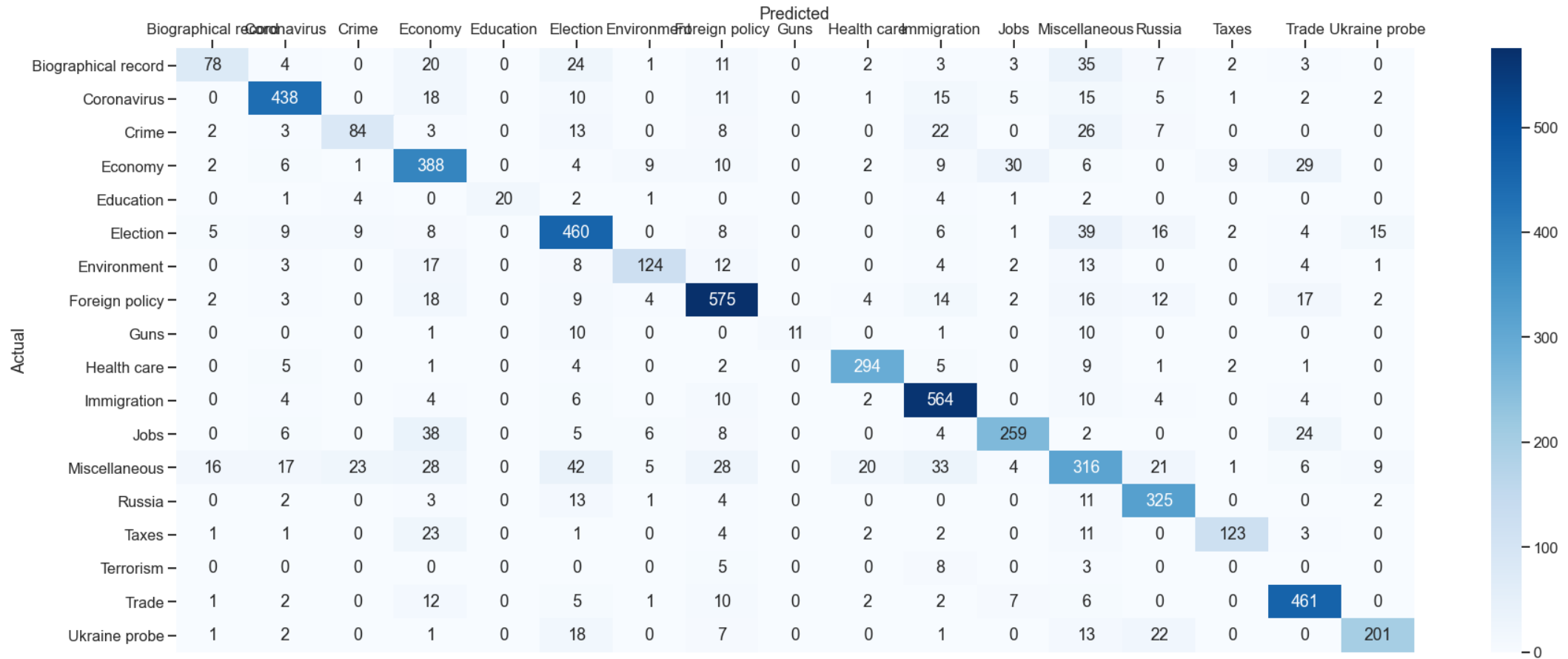
# Trump's Claims —Model Fit, Prediction and Evaluation

II

Predicted	Biographical record	Coronavirus	Crime	Economy	Education	Election	Environment	Foreign policy	Guns	Health care	Immigration	Jobs	Miscellaneous	Russia	Tax
Actual															
Biographical record	78	4	0	20	0	24	1	11	0	2	3	3	35	7	2
Coronavirus	0	438	0	18	0	10	0	11	0	1	15	5	15	5	1
Crime	2	3	84	3	0	13	0	8	0	0	22	0	26	7	0
Economy	2	6	1	388	0	4	9	10	0	2	9	30	6	0	9
Education	0	1	4	0	20	2	1	0	0	0	4	1	2	0	0
Election	5	9	9	8	0	460	0	8	0	0	6	1	39	16	2
Environment	0	3	0	17	0	8	124	12	0	0	4	2	13	0	0
Foreign policy	2	3	0	18	0	9	4	575	0	4	14	2	16	12	0
Guns	0	0	0	1	0	10	0	0	11	0	1	0	10	0	0
Health care	0	5	0	1	0	4	0	2	0	294	5	0	9	1	2
Immigration	0	4	0	4	0	6	0	10	0	2	564	0	10	4	0
Jobs	0	6	0	38	0	5	6	8	0	0	4	259	2	0	0
Miscellaneous	16	17	23	28	0	42	5	28	0	20	33	4	316	21	1
Russia	0	2	0	3	0	13	1	4	0	0	0	0	11	325	0
Taxes	1	1	0	23	0	1	0	4	0	2	2	0	11	0	123
Terrorism	0	0	0	0	0	0	0	5	0	0	8	0	3	0	0
Trade	1	2	0	12	0	5	1	10	0	2	2	7	6	0	0
Ukraine probe	1	2	0	1	0	18	0	7	0	0	1	0	13	22	0

# Trump's Claims — Model Fit, Prediction and Evaluation

III



- Look at the largest, off main-diagonal entries.



# Naïve Bayes Classifier — Review

## When to Consider

- Assumption of independence holds
- Categorical features.
- Spam filtering, Sentiment Analysis, and Recommendation Systems (with collaborative filtering).
- Variants exist for numeric (Gaussian-), binary (Bernoulli-) and multi-class (multinomial-) featured Naïve Bayes.

## Advantages

- It is easy and fast to predict class. It also perform well in multi-class prediction.
- When assumption of independence holds, performs better compare to other models like logistic regression and needs less training data.

## Disadvantages

- Ignores feature relationships
- If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as “Zero Frequency”. To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is Laplace estimation.
- If continuous features, then assumes normality conditions — often too restrictive.



## Resources

---

- A Gentle Introduction to Bayes Theorem for Machine Learning

<https://machinelearningmastery.com/bayes-theorem-for-machine-learning/>

This is well worth a read over a cup of coffee. The author, Jason Brownlee, is worth following.