06_column-transformer-text-feats

June 8, 2022

CPSC 330
Applied Machine Learning

1 Lecture 6: `sklearn ColumnTransformer` and Text Features

UBC 2022 Summer

Instructor: Mehrdad Oveisi

1.1 Imports

[1]:
```
import os
import sys

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from IPython.display import HTML

sys.path.append("code/.")
from plotting_functions import *
from utils import *

pd.set_option("display.max_colwidth", 200)

from sklearn.compose import ColumnTransformer, make_column_transformer
from sklearn.dummy import DummyClassifier, DummyRegressor
from sklearn.impute import SimpleImputer
from sklearn.model_selection import cross_val_score, cross_validate,␣
 ↪train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline, make_pipeline
from sklearn.preprocessing import OneHotEncoder, OrdinalEncoder, StandardScaler
from sklearn.svm import SVC
```

```
from sklearn.tree import DecisionTreeClassifier
```

1.2 Learning outcomes

From this lecture, you will be able to

- use `ColumnTransformer` to build all our transformations together into one object and use it with `sklearn` pipelines;

- define `ColumnTransformer` where transformers contain more than one steps;
- explain `handle_unknown="ignore"` hyperparameter of `scikit-learn`'s `OneHotEncoder`;
- explain `drop="if_binary"` argument of `OneHotEncoder`;
- identify when it's appropriate to apply ordinal encoding vs one-hot encoding;
- explain strategies to deal with categorical variables with too many categories;
- explain why text data needs a different treatment than categorical variables;
- use `scikit-learn`'s `CountVectorizer` to encode text data;
- explain different hyperparameters of `CountVectorizer`.

1.3 sklearn's `ColumnTransformer`

- In most applications, some features are categorical, some are continuous, some are binary, and some are ordinal.
- When we want to develop supervised machine learning pipelines on real-world datasets, very often we want to apply different transformation on different columns.
- Enter `sklearn`'s `ColumnTransformer`!!
- Let's look at a toy example:

[2]:
```
df = pd.read_csv("data/quiz2-grade-toy-col-transformer.csv")
df
```

[2]:

	enjoy_course	ml_experience	major	class_attendance	\
0	yes	1	Computer Science	Excellent	
1	yes	1	Mechanical Engineering	Average	
2	yes	0	Mathematics	Poor	
3	no	0	Mathematics	Excellent	
4	yes	0	Psychology	Good	
5	no	1	Economics	Good	
6	yes	1	Computer Science	Excellent	
7	no	0	Mechanical Engineering	Poor	
8	no	0	Linguistics	Average	
9	yes	1	Mathematics	Average	
10	yes	0	Psychology	Good	
11	yes	1	Physics	Average	
12	yes	1	Physics	Excellent	
13	yes	0	Mechanical Engineering	Excellent	
14	no	0	Mathematics	Poor	
15	no	1	Computer Science	Good	

```
16          yes              0        Computer Science          Average
17          yes              1               Economics          Average
18           no              1                 Biology             Good
19           no              0              Psychology             Poor
20          yes              1             Linguistics        Excellent
```

	university_years	lab1	lab2	lab3	lab4	quiz1	quiz2
0	3	92	93.0	84	91	92	A+
1	2	94	90.0	80	83	91	not A+
2	3	78	85.0	83	80	80	not A+
3	3	91	NaN	92	91	89	A+
4	4	77	83.0	90	92	85	A+
5	5	70	73.0	68	74	71	not A+
6	4	80	88.0	89	88	91	A+
7	3	95	93.0	69	79	75	not A+
8	2	97	90.0	94	82	80	not A+
9	4	95	82.0	94	94	85	not A+
10	3	98	86.0	95	95	78	A+
11	1	95	88.0	93	92	85	A+
12	2	98	96.0	96	99	100	A+
13	4	95	94.0	96	95	100	A+
14	3	95	90.0	93	95	70	not A+
15	3	92	85.0	67	94	92	not A+
16	5	75	91.0	93	86	85	A+
17	3	86	89.0	65	86	87	not A+
18	2	91	NaN	90	88	82	not A+
19	2	77	94.0	87	81	89	not A+
20	4	96	92.0	92	96	87	A+

[3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21 entries, 0 to 20
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   enjoy_course      21 non-null     object
 1   ml_experience     21 non-null     int64
 2   major             21 non-null     object
 3   class_attendance  21 non-null     object
 4   university_years  21 non-null     int64
 5   lab1              21 non-null     int64
 6   lab2              19 non-null     float64
 7   lab3              21 non-null     int64
 8   lab4              21 non-null     int64
 9   quiz1             21 non-null     int64
 10  quiz2             21 non-null     object
dtypes: float64(1), int64(6), object(4)
```

```
memory usage: 1.9+ KB
```

1.3.1 Transformations on the toy data

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

[4]:
```
  enjoy_course  ml_experience                major class_attendance  \
0          yes              1     Computer Science        Excellent
1          yes              1  Mechanical Engineering          Average
2          yes              0          Mathematics             Poor
3           no              0          Mathematics        Excellent
4          yes              0           Psychology             Good

   university_years  lab1  lab2  lab3  lab4  quiz1   quiz2
0                 3    92  93.0    84    91     92      A+
1                 2    94  90.0    80    83     91  not A+
2                 3    78  85.0    83    80     80  not A+
3                 3    91   NaN    92    91     89      A+
4                 4    77  83.0    90    92     85      A+
```

- Scaling on numeric features
- One-hot encoding on the categorical feature `major` and binary feature `enjoy_course`
- Ordinal encoding on the ordinal feature `class_attendance`
- Imputation on the `lab2` feature
- None on the `ml_experience` feature

1.3.2 `ColumnTransformer` example

Data

[5]:
```
X = df.drop(columns=["quiz2"])
y = df["quiz2"]
X.columns
```

[5]:
```
Index(['enjoy_course', 'ml_experience', 'major', 'class_attendance',
       'university_years', 'lab1', 'lab2', 'lab3', 'lab4', 'quiz1'],
      dtype='object')
```

Identify the transformations we want to apply

[6]:
```
X.head()
```

[6]:
```
  enjoy_course  ml_experience                major class_attendance  \
0          yes              1     Computer Science        Excellent
1          yes              1  Mechanical Engineering          Average
2          yes              0          Mathematics             Poor
3           no              0          Mathematics        Excellent
4          yes              0           Psychology             Good

   university_years  lab1  lab2  lab3  lab4  quiz1
```

```
0                3    92  93.0    84    91     92
1                2    94  90.0    80    83     91
2                3    78  85.0    83    80     80
3                3    91   NaN    92    91     89
4                4    77  83.0    90    92     85
```

[7]:
```
numeric_feats = ["university_years", "lab1", "lab3", "lab4", "quiz1"]  # apply␣
 ↪scaling
categorical_feats = ["major"]  # apply one-hot encoding
passthrough_feats = ["ml_experience"]  # do not apply any transformation
drop_feats = [
    "lab2",
    "class_attendance",
    "enjoy_course",
]  # for now, do not include these features in modeling
```

For simplicity, let's only focus on scaling and one-hot encoding first.

Create a column transformer

- Each transformation is specified by a name, a transformer object, and the columns this transformer should be applied to.

[8]:
```
from sklearn.compose import ColumnTransformer
```

[9]:
```
ct = ColumnTransformer(
    [
        ("scaling", StandardScaler(), numeric_feats),
        ("onehot", OneHotEncoder(sparse=False), categorical_feats),
    ]
)
```

Convenient `make_column_transformer` syntax

- Similar to `make_pipeline` syntax, there is convenient `make_column_transformer` syntax.
- The syntax automatically names each step based on its class.
- We'll be mostly using this syntax.

[10]:
```
from sklearn.compose import make_column_transformer

ct = make_column_transformer(
    (StandardScaler(), numeric_feats),  # scaling on numeric features
    (OneHotEncoder(), categorical_feats),  # OHE on categorical features
    ("passthrough", passthrough_feats),  # no transformations on the binary␣
 ↪features
    ("drop", drop_feats),  # drop the drop features
)
```

[11]:
```
ct
```

[11]:
```
ColumnTransformer(transformers=[('standardscaler', StandardScaler(),
                                 ['university_years', 'lab1', 'lab3', 'lab4',
                                  'quiz1']),
                                ('onehotencoder', OneHotEncoder(), ['major']),
                                ('passthrough', 'passthrough',
                                 ['ml_experience']),
                                ('drop', 'drop',
                                 ['lab2', 'class_attendance', 'enjoy_course'])])
```

[12]:
```
transformed = ct.fit_transform(X)
```

- When we `fit_transform`, each transformer is applied to the specified columns and the result of the transformations are concatenated horizontally.
- A big advantage here is that we build all our transformations together into one object, and that way we're sure we do the same operations to all splits of the data.
- Otherwise we might, for example, do the OHE on both train and test but forget to scale the test data.

Let's examine the transformed data

[13]:
```
transformed[:2]
```

[13]:
```
array([[-0.09345386,  0.3589134 , -0.21733442,  0.36269995,  0.84002795,
         0.        ,  1.        ,  0.        ,  0.        ,  0.        ,
         0.        ,  0.        ,  0.        ,  1.        ],
       [-1.07471942,  0.59082668, -0.61420598, -0.85597188,  0.71219761,
         0.        ,  0.        ,  0.        ,  0.        ,  0.        ,
         1.        ,  0.        ,  0.        ,  1.        ]])
```

[14]:
```
type(transformed)
```

[14]:
```
numpy.ndarray
```

Note that the returned object is not a dataframe. So there are no column names.

Viewing the transformed data as a dataframe

- How can we view our transformed data as a dataframe?
- We are adding more columns.
- So the original columns won't directly map to the transformed data.
- Let's create column names for the transformed data.

[15]:
```
ct.named_transformers_
```

[15]:
```
{'standardscaler': StandardScaler(),
 'onehotencoder': OneHotEncoder(),
 'passthrough': 'passthrough',
 'drop': 'drop'}
```

[16]:
```
# Here are the new columns created by OneHotEncoder
ct.named_transformers_["onehotencoder"].get_feature_names_out()
```

[16]:
```
array(['major_Biology', 'major_Computer Science', 'major_Economics',
       'major_Linguistics', 'major_Mathematics',
       'major_Mechanical Engineering', 'major_Physics',
       'major_Psychology'], dtype=object)
```

[17]:
```
column_names = (
    numeric_feats
    + ct.named_transformers_["onehotencoder"].get_feature_names_out().tolist()
    + passthrough_feats
)
column_names
```

[17]:
```
['university_years',
 'lab1',
 'lab3',
 'lab4',
 'quiz1',
 'major_Biology',
 'major_Computer Science',
 'major_Economics',
 'major_Linguistics',
 'major_Mathematics',
 'major_Mechanical Engineering',
 'major_Physics',
 'major_Psychology',
 'ml_experience']
```

Note that the order of the columns in the transformed data depends upon the order of the features we pass to the `ColumnTransformer` and can be different than the order of the features in the original dataframe.

[18]:
```
pd.DataFrame(transformed, columns=column_names).head()
```

[18]:

	university_years	lab1	lab3	lab4	quiz1	major_Biology	major_Computer Science	major_Economics	major_Linguistics
0	-0.093454	0.358913	-0.217334	0.362700	0.840028	0.0	1.0	0.0	0.0
1	-1.074719	0.590827	-0.614206	-0.855972	0.712198	0.0	0.0	0.0	0.0
2	-0.093454	-1.264480	-0.316552	-1.312974	-0.693936	0.0	0.0	0.0	0.0
3	-0.093454	0.242957	0.576409	0.362700	0.456537	0.0	0.0	0.0	0.0
4	0.887812	-1.380436	0.377973	0.515034	-0.054784	0.0			

```
4                     0.0              0.0               0.0

   major_Mathematics  major_Mechanical Engineering  major_Physics  \
0                0.0                           0.0            0.0
1                0.0                           1.0            0.0
2                1.0                           0.0            0.0
3                1.0                           0.0            0.0
4                0.0                           0.0            0.0

   major_Psychology  ml_experience
0               0.0            1.0
1               0.0            1.0
2               0.0            0.0
3               0.0            0.0
4               1.0            0.0
```

`ColumnTransformer`: Transformed data Adapted from here

Training models with transformed data

- We can now pass the `ColumnTransformer` object as a step in a pipeline.

[19]:
```
# Make a pipeline that applies ct to columns and then SVC to the resulting table
pipe = make_pipeline(ct, SVC())
pipe.fit(X, y)
pipe.predict(X)
```

[19]:
```
array(['A+', 'not A+', 'not A+', 'A+', 'A+', 'not A+', 'A+', 'not A+',
       'not A+', 'A+', 'A+', 'A+', 'A+', 'A+', 'not A+', 'not A+', 'A+',
       'not A+', 'not A+', 'not A+', 'A+'], dtype=object)
```

1.3.3 Questions for you

True/False: `ColumnTransformer`

1. You could carry out cross-validation by passing a `ColumnTransformer` object to `cross_validate`.
2. After applying column transformer, the order of the columns in the transformed data has to be the same as the order of the columns in the original data.
3. After applying a column transformer, the transformed data is always going to be of different shape than the original data.
4. When you call `fit_transform` on a `ColumnTransformer` object, you get a numpy ndarray.

What transformations on what columns? Consider the feature columns below.

- What transformations would you apply on each column?

colour	location	shape	water_content	weight
red	canada	NaN	84	100
yellow	mexico	long	75	120
orange	spain	NaN	90	NaN
magenta	china	round	NaN	600
purple	austria	NaN	80	115
purple	turkey	oval	78	340
green	mexico	oval	83	NaN
blue	canada	round	73	535
brown	china	NaN	NaN	1743
yellow	mexico	oval	83	265

1.4 More on feature transformations

1.4.1 Multiple transformations in a transformer

- Recall that `lab2` has missing values.

[20]:
```
X.head()
```

[20]:
```
  enjoy_course  ml_experience                 major class_attendance  \
0          yes              1       Computer Science        Excellent
1          yes              1  Mechanical Engineering          Average
2          yes              0             Mathematics             Poor
3           no              0             Mathematics        Excellent
4          yes              0              Psychology             Good

   university_years  lab1  lab2  lab3  lab4  quiz1
0                 3    92  93.0    84    91     92
1                 2    94  90.0    80    83     91
2                 3    78  85.0    83    80     80
3                 3    91   NaN    92    91     89
4                 4    77  83.0    90    92     85
```

- So we would like to apply more than one transformations on it: imputation and scaling.

- We can treat `lab2` separately, but we can also include it into `numeric_feats` and apply both transformations on all numeric columns.

[21]:
```
numeric_feats = [
    "university_years",
    "lab1",
    "lab2",
    "lab3",
    "lab4",
    "quiz1",
]  # apply scaling
categorical_feats = ["major"]  # apply one-hot encoding
```

```
passthrough_feats = ["ml_experience"]  # do not apply any transformation
drop_feats = ["class_attendance", "enjoy_course"]
```

- To apply more than one transformations we can define a pipeline inside a column transformer to chain different transformations.

[22]:
```
ct = make_column_transformer(
    (
        make_pipeline(SimpleImputer(), StandardScaler()),
        numeric_feats,
    ),  # scaling on numeric features
    (OneHotEncoder(), categorical_feats),  # OHE on categorical features
    ("passthrough", passthrough_feats),  # no transformations on the binary␣
 ↪features
    ("drop", drop_feats),  # drop the drop features
)
```

[23]:
```
X_transformed = ct.fit_transform(X)
```

[24]:
```
column_names = (
    numeric_feats
    + ct.named_transformers_["onehotencoder"].get_feature_names_out().tolist()
    + passthrough_feats
)
column_names
```

[24]:
```
['university_years',
 'lab1',
 'lab2',
 'lab3',
 'lab4',
 'quiz1',
 'major_Biology',
 'major_Computer Science',
 'major_Economics',
 'major_Linguistics',
 'major_Mathematics',
 'major_Mechanical Engineering',
 'major_Physics',
 'major_Psychology',
 'ml_experience']
```

[25]:
```
pd.DataFrame(X_transformed, columns=column_names).head()
```

[25]:

	university_years	lab1	lab2	lab3	lab4	quiz1	\
0	-0.093454	0.358913	0.893260	-0.217334	0.362700	0.840028	
1	-1.074719	0.590827	0.294251	-0.614206	-0.855972	0.712198	
2	-0.093454	-1.264480	-0.704099	-0.316552	-1.312974	-0.693936	

```
3         -0.093454  0.242957  0.000000  0.576409  0.362700  0.456537
4          0.887812 -1.380436 -1.103439  0.377973  0.515034 -0.054784

   major_Biology  major_Computer Science  major_Economics  major_Linguistics  \
0            0.0                     1.0              0.0                0.0
1            0.0                     0.0              0.0                0.0
2            0.0                     0.0              0.0                0.0
3            0.0                     0.0              0.0                0.0
4            0.0                     0.0              0.0                0.0

   major_Mathematics  major_Mechanical Engineering  major_Physics  \
0                0.0                           0.0            0.0
1                0.0                           1.0            0.0
2                1.0                           0.0            0.0
3                1.0                           0.0            0.0
4                0.0                           0.0            0.0

   major_Psychology  ml_experience
0               0.0            1.0
1               0.0            1.0
2               0.0            0.0
3               0.0            0.0
4               1.0            0.0
```

1.4.2 sklearn set_config

- With multiple transformations in a column transformer, it can get tricky to keep track of everything happening inside it.
- We can use `set_config` to display a diagram of this.

[26]:
```
from sklearn import set_config

set_config(display="diagram")
```

[27]:
```
ct
```

[27]:
```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
                                                  SimpleImputer()),
                                                 ('standardscaler',
                                                  StandardScaler())]),
                                 ['university_years', 'lab1', 'lab2', 'lab3',
                                  'lab4', 'quiz1']),
                                ('onehotencoder', OneHotEncoder(), ['major']),
                                ('passthrough', 'passthrough',
                                 ['ml_experience']),
                                ('drop', 'drop',
```

```
                                  ['class_attendance', 'enjoy_course'])])
```

[28]:
```
print(ct)
```

```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
                                                  SimpleImputer()),
                                                 ('standardscaler',
                                                  StandardScaler())]),
                                 ['university_years', 'lab1', 'lab2', 'lab3',
                                  'lab4', 'quiz1']),
                                ('onehotencoder', OneHotEncoder(), ['major']),
                                ('passthrough', 'passthrough',
                                 ['ml_experience']),
                                ('drop', 'drop',
                                 ['class_attendance', 'enjoy_course'])])
```

1.4.3 Incorporating *ordinal* feature `class_attendance`

- The `class_attendance` column is different than the `major` column in that there is some ordering of the values.
  - Excellent > Good > Average > Poor

[29]:
```
X.head()
```

[29]:

	enjoy_course	ml_experience	major	class_attendance	university_years	lab1	lab2	lab3	lab4	quiz1
0	yes	1	Computer Science	Excellent	3	92	93.0	84	91	92
1	yes	1	Mechanical Engineering	Average	2	94	90.0	80	83	91
2	yes	0	Mathematics	Poor	3	78	85.0	83	80	80
3	no	0	Mathematics	Excellent	3	91	NaN	92	91	89
4	yes	0	Psychology	Good	4	77	83.0	90	92	85

Let's try applying `OrdinalEncoder` on `class_attendance` column.

[30]:
```
X_toy = X[["class_attendance"]]
enc = OrdinalEncoder()
enc.fit(X_toy)
X_toy_ord = enc.transform(X_toy)
X_toy_ord_df = pd.DataFrame(
    data=X_toy_ord,
    columns=["class_attendance_ord"],
    index=X_toy.index,
```

```
)
```

[31]:
```
X_toy.join(X_toy_ord_df).head(10)
```

[31]:
```
  class_attendance  class_attendance_ord
0        Excellent                   1.0
1          Average                   0.0
2             Poor                   3.0
3        Excellent                   1.0
4             Good                   2.0
5             Good                   2.0
6        Excellent                   1.0
7             Poor                   3.0
8          Average                   0.0
9          Average                   0.0
```

- What's the problem here?
  - The encoder doesn't know the order.
- We can examine unique categories **manually, order them based on our intuitions**, and then provide this human knowledge to the transformer.

What are the unique categories of `class_attendance`?

[32]:
```
X_toy["class_attendance"].unique()
```

[32]:
```
array(['Excellent', 'Average', 'Poor', 'Good'], dtype=object)
```

Let's order them manually.

[33]:
```
class_attendance_levels = ["Poor", "Average", "Good", "Excellent"]
```

Note that if you use the reverse order of the categories, it wouldn't matter.

Let's make sure that we have included all categories in our manual ordering.

[34]:
```
assert set(class_attendance_levels) == set(X_toy["class_attendance"].unique())
```

[35]:
```
oe = OrdinalEncoder(categories=[class_attendance_levels], dtype=int)
oe.fit(X_toy[["class_attendance"]])
ca_ord = oe.transform(X_toy[["class_attendance"]])
ca_ord_df = pd.DataFrame(
    data=ca_ord, columns=["class_attendance_ord"], index=X_toy.index
)
print(oe.categories_)
X_toy.join(ca_ord_df).head(10)
```

```
[array(['Poor', 'Average', 'Good', 'Excellent'], dtype=object)]
```

[35]:
```
  class_attendance  class_attendance_ord
0        Excellent                     3
1          Average                     1
```

```
2              Poor                    0
3         Excellent                    3
4              Good                    2
5              Good                    2
6         Excellent                    3
7              Poor                    0
8           Average                    1
9           Average                    1
```

The encoded categories are looking better now!

More than one ordinal columns?

- We can pass the manually ordered categories when we create an `OrdinalEncoder` object as a list of lists.
- If you have more than one ordinal columns
  - manually create a list of ordered categories for each column
  - pass a list of lists to `OrdinalEncoder`, where each inner list corresponds to manually created list of ordered categories for a corresponding ordinal column.

Now let's incorporate ordinal encoding of `class_attendance` in our column transformer.

[36]:
```
numeric_feats = [
    "university_years",
    "lab1",
    "lab2",
    "lab3",
    "lab4",
    "quiz1",
]  # apply scaling
categorical_feats = ["major"]  # apply one-hot encoding
ordinal_feats = ["class_attendance"]  # apply ordinal encoding
passthrough_feats = ["ml_experience"]  # do not apply any transformation
drop_feats = ["enjoy_course"]  # do not include these features
```

[37]:
```
ct = make_column_transformer(
    (
        make_pipeline(SimpleImputer(), StandardScaler()),
        numeric_feats,
    ),  # scaling on numeric features
    (OneHotEncoder(), categorical_feats),  # OHE on categorical features
    (
        OrdinalEncoder(categories=[class_attendance_levels], dtype=int),
        ordinal_feats,
    ),  # Ordinal encoding on ordinal features
    ("passthrough", passthrough_feats),  # no transformations on the binary␣
↪features
    ("drop", drop_feats),  # drop the drop features
```

```
)
```

[38]:
```
ct
```

[38]:
```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
                                                  SimpleImputer()),
                                                 ('standardscaler',
                                                  StandardScaler())]),
                                 ['university_years', 'lab1', 'lab2', 'lab3',
                                  'lab4', 'quiz1']),
                                ('onehotencoder', OneHotEncoder(), ['major']),
                                ('ordinalencoder',
                                 OrdinalEncoder(categories=[['Poor', 'Average',
                                                             'Good',
                                                             'Excellent']],
                                                dtype=<class 'int'>),
                                 ['class_attendance']),
                                ('passthrough', 'passthrough',
                                 ['ml_experience']),
                                ('drop', 'drop', ['enjoy_course'])])
```

[39]:
```
X_transformed = ct.fit_transform(X)
```

[40]:
```
column_names = (
    numeric_feats
    + ct.named_transformers_["onehotencoder"].get_feature_names_out().tolist()
    + ordinal_feats
    + passthrough_feats
)
column_names
```

[40]:
```
['university_years',
 'lab1',
 'lab2',
 'lab3',
 'lab4',
 'quiz1',
 'major_Biology',
 'major_Computer Science',
 'major_Economics',
 'major_Linguistics',
 'major_Mathematics',
 'major_Mechanical Engineering',
 'major_Physics',
 'major_Psychology',
 'class_attendance',
```

```
 'ml_experience']
```

[41]:
```
pd.DataFrame(X_transformed, columns=column_names)
```

[41]:

	university_years	lab1	lab2	lab3	lab4	quiz1	\
0	-0.093454	0.358913	0.893260	-0.217334	0.362700	0.840028	
1	-1.074719	0.590827	0.294251	-0.614206	-0.855972	0.712198	
2	-0.093454	-1.264480	-0.704099	-0.316552	-1.312974	-0.693936	
3	-0.093454	0.242957	0.000000	0.576409	0.362700	0.456537	
4	0.887812	-1.380436	-1.103439	0.377973	0.515034	-0.054784	
5	1.869077	-2.192133	-3.100139	-1.804821	-2.226978	-1.844409	
6	0.887812	-1.032566	-0.105089	0.278755	-0.094302	0.712198	
7	-0.093454	0.706783	0.893260	-1.705603	-1.465308	-1.333088	
8	-1.074719	0.938697	0.294251	0.774844	-1.008306	-0.693936	
9	0.887812	0.706783	-1.303109	0.774844	0.819702	-0.054784	
10	-0.093454	1.054653	-0.504429	0.874062	0.972036	-0.949597	
11	-2.055985	0.706783	-0.105089	0.675627	0.515034	-0.054784	
12	-1.074719	1.054653	1.492270	0.973280	1.581372	1.862671	
13	0.887812	0.706783	1.092930	0.973280	0.972036	1.862671	
14	-0.093454	0.706783	0.294251	0.675627	0.972036	-1.972240	
15	-0.093454	0.358913	-0.704099	-1.904039	0.819702	0.840028	
16	1.869077	-1.612349	0.493921	0.675627	-0.398970	-0.054784	
17	-0.093454	-0.336826	0.094581	-2.102474	-0.398970	0.200876	
18	-1.074719	0.242957	0.000000	0.377973	-0.094302	-0.438275	
19	-1.074719	-1.380436	1.092930	0.080319	-1.160640	0.456537	
20	0.887812	0.822740	0.693590	0.576409	1.124370	0.200876	

	major_Biology	major_Computer Science	major_Economics	major_Linguistics	\
0	0.0	1.0	0.0	0.0	
1	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.0	
3	0.0	0.0	0.0	0.0	
4	0.0	0.0	0.0	0.0	
5	0.0	0.0	1.0	0.0	
6	0.0	1.0	0.0	0.0	
7	0.0	0.0	0.0	0.0	
8	0.0	0.0	0.0	1.0	
9	0.0	0.0	0.0	0.0	
10	0.0	0.0	0.0	0.0	
11	0.0	0.0	0.0	0.0	
12	0.0	0.0	0.0	0.0	
13	0.0	0.0	0.0	0.0	
14	0.0	0.0	0.0	0.0	
15	0.0	1.0	0.0	0.0	
16	0.0	1.0	0.0	0.0	
17	0.0	0.0	1.0	0.0	
18	1.0	0.0	0.0	0.0	

19	0.0	0.0	0.0	0.0
20	0.0	0.0	0.0	1.0

	major_Mathematics	major_Mechanical Engineering	major_Physics	\
0	0.0	0.0	0.0	
1	0.0	1.0	0.0	
2	1.0	0.0	0.0	
3	1.0	0.0	0.0	
4	0.0	0.0	0.0	
5	0.0	0.0	0.0	
6	0.0	0.0	0.0	
7	0.0	1.0	0.0	
8	0.0	0.0	0.0	
9	1.0	0.0	0.0	
10	0.0	0.0	0.0	
11	0.0	0.0	1.0	
12	0.0	0.0	1.0	
13	0.0	1.0	0.0	
14	1.0	0.0	0.0	
15	0.0	0.0	0.0	
16	0.0	0.0	0.0	
17	0.0	0.0	0.0	
18	0.0	0.0	0.0	
19	0.0	0.0	0.0	
20	0.0	0.0	0.0	

	major_Psychology	class_attendance	ml_experience
0	0.0	3.0	1.0
1	0.0	1.0	1.0
2	0.0	0.0	0.0
3	0.0	3.0	0.0
4	1.0	2.0	0.0
5	0.0	2.0	1.0
6	0.0	3.0	1.0
7	0.0	0.0	0.0
8	0.0	1.0	0.0
9	0.0	1.0	1.0
10	1.0	2.0	0.0
11	0.0	1.0	1.0
12	0.0	3.0	1.0
13	0.0	3.0	0.0
14	0.0	0.0	0.0
15	0.0	2.0	1.0
16	0.0	1.0	0.0
17	0.0	1.0	1.0
18	0.0	2.0	1.0
19	1.0	0.0	0.0

```
20              0.0              3.0              1.0
```

1.4.4 Dealing with unknown categories

How does `OneHotEncoder` deal with unknown categories? Let's see an example:

[42]:
```
X_toy = [['science', 10], ['arts', 30], ['arts', 20]]
columns=['subject', 'group']
pd.DataFrame(X_toy, columns=columns)
```

[42]:

	subject	group
0	science	10
1	arts	30
2	arts	20

[43]:
```
ohe = OneHotEncoder(handle_unknown='error')  # default value for handle_unknown␣
↪is 'error'
ohe.fit(X_toy);
```

[44]:
```
columns_ohe = ohe.get_feature_names_out(['subject', 'group']).tolist()
columns_ohe
```

[44]:
```
['subject_arts', 'subject_science', 'group_10', 'group_20', 'group_30']
```

[45]:
```
ohe.categories_
```

[45]:
```
[array(['arts', 'science'], dtype=object), array([10, 20, 30], dtype=object)]
```

[46]:
```
ex1 = ohe.transform([['arts', 10], ['science', 30]]).toarray()
ex1
```

[46]:
```
array([[1., 0., 1., 0., 0.],
       [0., 1., 0., 0., 1.]])
```

[47]:
```
pd.DataFrame(ex1, columns=columns_ohe)
```

[47]:

	subject_arts	subject_science	group_10	group_20	group_30
0	1.0	0.0	1.0	0.0	0.0
1	0.0	1.0	0.0	0.0	1.0

[48]:
```
# ex2 = ohe.transform([['arts', 10], ['science', 4]]).toarray()

# This would give an error:
# ValueError: Found unknown categories [4] in column 1 during transform
```

[49]:
```
ohe = OneHotEncoder(handle_unknown='ignore')  # now use 'ignore' instead of␣
↪'error'
ohe.fit(X_toy);
```

[50]:
```
ex2 = ohe.transform([['arts', 10], ['science', 4]]).toarray()
ex2
```

[50]:
```
array([[1., 0., 1., 0., 0.],
       [0., 1., 0., 0., 0.]])
```

[51]:
```
pd.DataFrame(ex2, columns=columns_ohe)  # all "group" columns are 0 for value 4
```

[51]:

	subject_arts	subject_science	group_10	group_20	group_30
0	1.0	0.0	1.0	0.0	0.0
1	0.0	1.0	0.0	0.0	0.0

[52]:
```
ex3 = ohe.inverse_transform([[0, 1, 1, 0, 0], [1, 0, 0, 0, 0]])
ex3
```

[52]:
```
array([['science', 10],
       ['arts', None]], dtype=object)
```

[53]:
```
pd.DataFrame(ex3, columns=columns)
```

[53]:

	subject	group
0	science	10
1	arts	None

[54]:
```
ex4 = ohe.inverse_transform([[0, 1, 0, 0, 1], [0, 0, 0, 1, 0]])
ex4
```

[54]:
```
array([['science', 30],
       [None, 20]], dtype=object)
```

[55]:
```
pd.DataFrame(ex4, columns=columns)
```

[55]:

	subject	group
0	science	30
1	None	20

What if we know the possible categories beforehand? We can **specify categories** ahead of time.

[56]:
```
ohe = OneHotEncoder(handle_unknown='error', categories=[['arts', 'science'],␣
↪[10, 20, 30, 4]])
ohe.fit(X_toy);
```

Even though `handle_unknown='error'`, `ex2` does not give error anymore because `categories` are known.

[57]:
```
ex2 = ohe.transform([['arts', 10], ['science', 4]]).toarray()
ex2
```

[57]:
```
array([[1., 0., 1., 0., 0., 0.],
       [0., 1., 0., 0., 0., 1.]])
```

[58]:
```
columns_ohe = ohe.get_feature_names_out(['subject', 'group']).tolist()
pd.DataFrame(ex2, columns=columns_ohe)
```

[58]:

	subject_arts	subject_science	group_10	group_20	group_30	group_4
0	1.0	0.0	1.0	0.0	0.0	0.0
1	0.0	1.0	0.0	0.0	0.0	1.0

Dealing with unknown categories in `cross_validate` Let's create a pipeline with the column transformer and pass it to `cross_validate`.

[59]:
```
ct
```

[59]:
```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
                                                  SimpleImputer()),
                                                 ('standardscaler',
                                                  StandardScaler())]),
                                 ['university_years', 'lab1', 'lab2', 'lab3',
                                  'lab4', 'quiz1']),
                                ('onehotencoder', OneHotEncoder(), ['major']),
                                ('ordinalencoder',
                                 OrdinalEncoder(categories=[['Poor', 'Average',
                                                             'Good',
                                                             'Excellent']],
                                                dtype=<class 'int'>),
                                 ['class_attendance']),
                                ('passthrough', 'passthrough',
                                 ['ml_experience']),
                                ('drop', 'drop', ['enjoy_course'])])
```

[60]:
```
pipe = make_pipeline(ct, SVC())
```

[61]:
```
scores = cross_validate(pipe, X, y, return_train_score=True)
pd.DataFrame(scores)
```

```
/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/model_selection/_validation.py:770: UserWarning: Scoring
failed. The score on this train-test partition for these parameters will be set
to nan. Details:
Traceback (most recent call last):
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/model_selection/_validation.py", line 761, in _score
    scores = scorer(estimator, X_test, y_test)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/metrics/_scorer.py", line 418, in _passthrough_scorer
```

```
    return estimator.score(*args, **kwargs)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/utils/metaestimators.py", line 113, in <lambda>
    out = lambda *args, **kwargs: self.fn(obj, *args, **kwargs)  # noqa
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/pipeline.py", line 707, in score
    Xt = transform.transform(Xt)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/compose/_column_transformer.py", line 748, in transform
    Xs = self._fit_transform(
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/compose/_column_transformer.py", line 606, in _fit_transform
    return Parallel(n_jobs=self.n_jobs)(
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/joblib/parallel.py", line 1044, in __call__
    while self.dispatch_one_batch(iterator):
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/joblib/parallel.py", line 859, in dispatch_one_batch
    self._dispatch(tasks)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/joblib/parallel.py", line 777, in _dispatch
    job = self._backend.apply_async(batch, callback=cb)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/joblib/_parallel_backends.py", line 208, in apply_async
    result = ImmediateResult(func)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/joblib/_parallel_backends.py", line 572, in __init__
    self.results = batch()
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/joblib/parallel.py", line 262, in __call__
    return [func(*args, **kwargs)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/joblib/parallel.py", line 262, in <listcomp>
    return [func(*args, **kwargs)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/utils/fixes.py", line 216, in __call__
    return self.function(*args, **kwargs)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/pipeline.py", line 876, in _transform_one
    res = transformer.transform(X)
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/preprocessing/_encoders.py", line 509, in transform
    X_int, X_mask = self._transform(
  File "/home/moveisi/miniconda3/envs/cpsc330/lib/python3.10/site-
packages/sklearn/preprocessing/_encoders.py", line 142, in _transform
    raise ValueError(msg)
ValueError: Found unknown categories ['Biology'] in column 0 during transform
```

```
  warnings.warn(
```

[61]:

	fit_time	score_time	test_score	train_score
0	0.008816	0.005662	1.00	0.937500
1	0.008196	0.004937	1.00	0.941176
2	0.007402	0.003597	0.50	1.000000
3	0.006360	0.004390	0.75	0.941176
4	0.006452	0.006439	NaN	1.000000

- What's going on here??
- Let's look at the error message:
  - `ValueError: Found unknown categories ['Biology'] in column 0 during transform`
  - Same error that we got in the `OneHotEncoder` example above (`ex2`)

[62]:
```
X["major"].value_counts()
```

[62]:
```
Computer Science          4
Mathematics               4
Mechanical Engineering    3
Psychology                3
Economics                 2
Linguistics               2
Physics                   2
Biology                   1
Name: major, dtype: int64
```

- There is only one instance of Biology.
- During cross-validation, this is getting put into the validation split.
- By default, `OneHotEncoder` throws an error because you might want to know about this.

Simplest fix: - Pass `handle_unknown="ignore"` argument to `OneHotEncoder` - It creates a row with all zeros (as we saw in the example above)

[63]:
```
ct = make_column_transformer(
    (
        make_pipeline(SimpleImputer(), StandardScaler()),
        numeric_feats,
    ),  # scaling on numeric features
    (
        OneHotEncoder(handle_unknown="ignore"),
        categorical_feats,
    ),  # OHE on categorical features
    (
        OrdinalEncoder(categories=[class_attendance_levels], dtype=int),
        ordinal_feats,
    ),  # Ordinal encoding on ordinal features
    ("passthrough", passthrough_feats),  # no transformations on the binary␣
↪features
```

```
    ("drop", drop_feats),  # drop the drop features
)
```

[64]:
```
ct
```

[64]:
```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
                                                  SimpleImputer()),
                                                 ('standardscaler',
                                                  StandardScaler())]),
                                 ['university_years', 'lab1', 'lab2', 'lab3',
                                  'lab4', 'quiz1']),
                                ('onehotencoder',
                                 OneHotEncoder(handle_unknown='ignore'),
                                 ['major']),
                                ('ordinalencoder',
                                 OrdinalEncoder(categories=[['Poor', 'Average',
                                                             'Good',
                                                             'Excellent']],
                                                dtype=<class 'int'>),
                                 ['class_attendance']),
                                ('passthrough', 'passthrough',
                                 ['ml_experience']),
                                ('drop', 'drop', ['enjoy_course'])])
```

[65]:
```
pipe = make_pipeline(ct, SVC())
```

[66]:
```
scores = cross_validate(pipe, X, y, cv=5, return_train_score=True)
pd.DataFrame(scores)
```

[66]:

	fit_time	score_time	test_score	train_score
0	0.008285	0.006410	1.00	0.937500
1	0.008545	0.004508	1.00	0.941176
2	0.006607	0.003485	0.50	1.000000
3	0.006461	0.005872	0.75	0.941176
4	0.007575	0.004189	0.75	1.000000

- With this approach, all unknown categories will be represented with all zeros and cross-validation is running OK now.

Ask yourself the following questions when you work with categorical variables
- Do you want this behaviour? - Are you expecting to get many unknown categories? Do you want to be able to distinguish between them?

Cases where it's OK to break the golden rule We saw above that if we know categories beforehand we can specify them to `OneHotEncoder` to avoid errors during `cross_over` assessments. However, wouldn't tha be ***breaking the golden*** rule?

When we know the categories in advance and this is one of the cases where it might be OK to violate the golden rule and get a list of all possible values for the categorical variable.

For example, if it's some fix number of categories. E.g., if it's something like: - provinces in Canada or - majors taught at UBC.

1.4.5 Categorical features with only two possible categories

- Sometimes you have features with only two possible categories.
- If we apply `OheHotEncoder` on such columns, it'll **create two columns, which seems wasteful**, as we could represent all information in the column in just one column with say 0's and 1's with presence of absence of one of one of the categories.
- You can pass `drop="if_binary"` argument to `OneHotEncoder` in order to create only one column in such scenario.

[67]:
```
X["enjoy_course"].head()
```

[67]:
```
0    yes
1    yes
2    yes
3     no
4    yes
Name: enjoy_course, dtype: object
```

[68]:
```
ohe_enc = OneHotEncoder(drop="if_binary", dtype=int, sparse=False)
ohe_enc.fit(X[["enjoy_course"]])
transformed = ohe_enc.transform(X[["enjoy_course"]])
df = pd.DataFrame(data=transformed, columns=["enjoy_course_enc"], index=X.index)
X[["enjoy_course"]].join(df).head(10)
```

[68]:

	enjoy_course	enjoy_course_enc
0	yes	1
1	yes	1
2	yes	1
3	no	0
4	yes	1
5	no	0
6	yes	1
7	no	0
8	no	0
9	yes	1

[69]:
```
numeric_feats = [
    "university_years",
    "lab1",
    "lab2",
    "lab3",
    "lab4",
    "quiz1",
```

```
]  # apply scaling
categorical_feats = ["major"]  # apply one-hot encoding
ordinal_feats = ["class_attendance"]  # apply ordinal encoding
binary_feats = ["enjoy_course"]  # apply one-hot encoding with drop="if_binary"
passthrough_feats = ["ml_experience"]  # do not apply any transformation
drop_feats = []
```

[70]:
```
ct = make_column_transformer(
    (
        make_pipeline(SimpleImputer(), StandardScaler()),
        numeric_feats,
    ),  # scaling on numeric features
    (
        OneHotEncoder(handle_unknown="ignore"),
        categorical_feats,
    ),  # OHE on categorical features
    (
        OrdinalEncoder(categories=[class_attendance_levels], dtype=int),
        ordinal_feats,
    ),  # Ordinal encoding on ordinal features
    (
        OneHotEncoder(drop="if_binary", dtype=int),
        binary_feats,
    ),  # OHE on categorical features
    ("passthrough", passthrough_feats),  # no transformations on the binary␣
 ↪features
)
```

[71]:
```
ct
```

[71]:
```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
                                                  SimpleImputer()),
                                                 ('standardscaler',
                                                  StandardScaler())]),
                                 ['university_years', 'lab1', 'lab2', 'lab3',
                                  'lab4', 'quiz1']),
                                ('onehotencoder-1',
                                 OneHotEncoder(handle_unknown='ignore'),
                                 ['major']),
                                ('ordinalencoder',
                                 OrdinalEncoder(categories=[['Poor', 'Average',
                                                             'Good',
                                                             'Excellent']],
                                                dtype=<class 'int'>),
                                 ['class_attendance']),
                                ('onehotencoder-2',
```

```
                                 OneHotEncoder(drop='if_binary',
                                               dtype=<class 'int'>),
                                 ['enjoy_course']),
                                ('passthrough', 'passthrough',
                                 ['ml_experience'])])
```

[72]:
```
pipe = make_pipeline(ct, SVC())
```

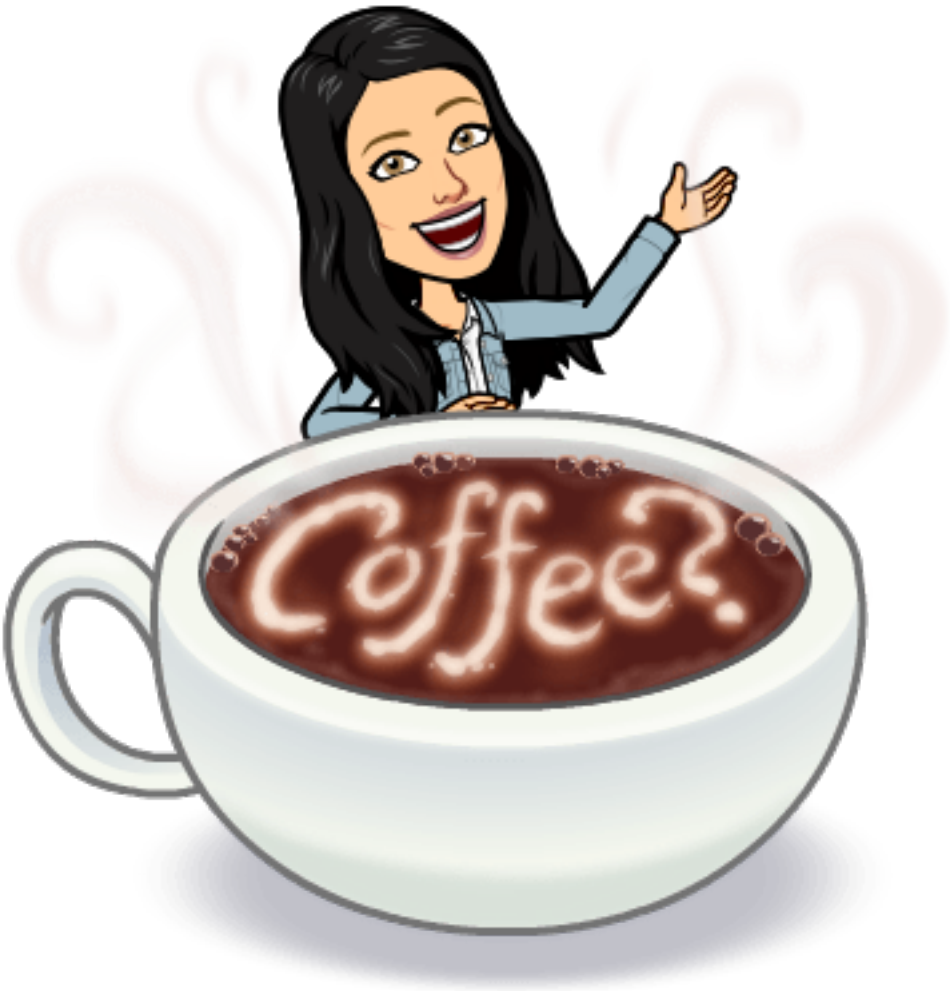
[73]:
```
scores = cross_validate(pipe, X, y, cv=5, return_train_score=True)
pd.DataFrame(scores)
```

[73]:

	fit_time	score_time	test_score	train_score
0	0.010822	0.005830	1.00	1.000000
1	0.008248	0.004571	1.00	0.941176
2	0.008762	0.005081	0.50	1.000000
3	0.008046	0.005495	1.00	0.941176
4	0.007485	0.004781	0.75	1.000000

Note Do not read too much into the scores, as we are running cross-validation on a very small dataset with 21 examples. The main point here is to show you how can we use `ColumnTransformer` to apply different transformations on different columns.

1.5 Break (5 min)



1.6 `ColumnTransformer` on the California housing dataset

[74]:
```
housing_df = pd.read_csv("data/housing.csv")
train_df, test_df = train_test_split(housing_df, test_size=0.1,␣
 ↪random_state=123)

train_df.head()
```

[74]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
6051	-117.75	34.04	22.0	2948.0	636.0	
20113	-119.57	37.94	17.0	346.0	130.0	
14289	-117.13	32.74	46.0	3355.0	768.0	
13665	-117.31	34.02	18.0	1634.0	274.0	
14471	-117.23	32.88	18.0	5566.0	1465.0	

```
       population  households  median_income  median_house_value  \
6051       2600.0       602.0         3.1250            113600.0
20113        51.0        20.0         3.4861            137500.0
14289      1457.0       708.0         2.6604            170100.0
13665       899.0       285.0         5.2139            129300.0
14471      6303.0      1458.0         1.8580            205000.0

      ocean_proximity
6051           INLAND
20113          INLAND
14289      NEAR OCEAN
13665          INLAND
14471      NEAR OCEAN
```

Some column values are mean/median but some are not.

Let's add some new features to the dataset which could help predicting the target: `median_house_value`.

[75]:
```
train_df = train_df.assign(
    rooms_per_household=train_df["total_rooms"] / train_df["households"]
)
test_df = test_df.assign(
    rooms_per_household=test_df["total_rooms"] / test_df["households"]
)

train_df = train_df.assign(
    bedrooms_per_household=train_df["total_bedrooms"] / train_df["households"]
)
test_df = test_df.assign(
    bedrooms_per_household=test_df["total_bedrooms"] / test_df["households"]
)

train_df = train_df.assign(
    population_per_household=train_df["population"] / train_df["households"]
)
test_df = test_df.assign(
    population_per_household=test_df["population"] / test_df["households"]
)
```

[76]:
```
train_df.head()
```

[76]:
```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
6051     -117.75     34.04                22.0       2948.0           636.0
20113    -119.57     37.94                17.0        346.0           130.0
14289    -117.13     32.74                46.0       3355.0           768.0
13665    -117.31     34.02                18.0       1634.0           274.0
14471    -117.23     32.88                18.0       5566.0          1465.0
```

```
       population  households  median_income  median_house_value  \
6051       2600.0       602.0         3.1250            113600.0
20113        51.0        20.0         3.4861            137500.0
14289      1457.0       708.0         2.6604            170100.0
13665       899.0       285.0         5.2139            129300.0
14471      6303.0      1458.0         1.8580            205000.0

      ocean_proximity  rooms_per_household  bedrooms_per_household  \
6051           INLAND             4.897010                1.056478
20113          INLAND            17.300000                6.500000
14289      NEAR OCEAN             4.738701                1.084746
13665          INLAND             5.733333                0.961404
14471      NEAR OCEAN             3.817558                1.004801

       population_per_household
6051                   4.318937
20113                  2.550000
14289                  2.057910
13665                  3.154386
14471                  4.323045
```

[77]:
```
# Let's keep both numeric and categorical columns in the data.
X_train = train_df.drop(columns=["median_house_value"])
y_train = train_df["median_house_value"]

X_test = test_df.drop(columns=["median_house_value"])
y_test = test_df["median_house_value"]
```

[78]:
```
from sklearn.compose import ColumnTransformer, make_column_transformer
```

[79]:
```
X_train.head(10)
```

[79]:
```
       longitude  latitude  housing_median_age  total_rooms  total_bedrooms  \
6051     -117.75     34.04                22.0       2948.0           636.0
20113    -119.57     37.94                17.0        346.0           130.0
14289    -117.13     32.74                46.0       3355.0           768.0
13665    -117.31     34.02                18.0       1634.0           274.0
14471    -117.23     32.88                18.0       5566.0          1465.0
9730     -121.74     36.79                16.0       3841.0           620.0
14690    -117.09     32.80                36.0       2163.0           367.0
7938     -118.11     33.86                33.0       2389.0           410.0
18365    -122.12     37.28                21.0        349.0            64.0
10931    -117.91     33.74                25.0       4273.0           965.0

       population  households  median_income ocean_proximity  \
6051       2600.0       602.0         3.1250          INLAND
```

```
20113        51.0        20.0         3.4861          INLAND
14289      1457.0       708.0         2.6604      NEAR OCEAN
13665       899.0       285.0         5.2139          INLAND
14471      6303.0      1458.0         1.8580      NEAR OCEAN
9730       1799.0       611.0         4.3814       <1H OCEAN
14690       915.0       360.0         4.7188      NEAR OCEAN
7938       1229.0       393.0         5.3889       <1H OCEAN
18365       149.0        56.0         5.8691       <1H OCEAN
10931      2946.0       922.0         2.9926       <1H OCEAN

       rooms_per_household  bedrooms_per_household  population_per_household
6051              4.897010                1.056478                  4.318937
20113            17.300000                6.500000                  2.550000
14289             4.738701                1.084746                  2.057910
13665             5.733333                0.961404                  3.154386
14471             3.817558                1.004801                  4.323045
9730              6.286416                1.014730                  2.944354
14690             6.008333                1.019444                  2.541667
7938              6.078880                1.043257                  3.127226
18365             6.232143                1.142857                  2.660714
10931             4.634490                1.046638                  3.195228
```

[80]:
```
X_train.columns
```

[80]:
```
Index(['longitude', 'latitude', 'housing_median_age', 'total_rooms',
       'total_bedrooms', 'population', 'households', 'median_income',
       'ocean_proximity', 'rooms_per_household', 'bedrooms_per_household',
       'population_per_household'],
      dtype='object')
```

[81]:
```
# Identify the categorical and numeric columns
numeric_features = [
    "longitude",
    "latitude",
    "housing_median_age",
    "total_rooms",
    "total_bedrooms",
    "population",
    "households",
    "median_income",
    "rooms_per_household",
    "bedrooms_per_household",
    "population_per_household",
]

categorical_features = ["ocean_proximity"]
target = "median_house_value"
```

- Let's create a `ColumnTransformer` for our dataset.

[82]:
```
X_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 18576 entries, 6051 to 19966
Data columns (total 12 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   longitude                 18576 non-null  float64
 1   latitude                  18576 non-null  float64
 2   housing_median_age        18576 non-null  float64
 3   total_rooms               18576 non-null  float64
 4   total_bedrooms            18391 non-null  float64
 5   population                18576 non-null  float64
 6   households                18576 non-null  float64
 7   median_income             18576 non-null  float64
 8   ocean_proximity           18576 non-null  object
 9   rooms_per_household       18576 non-null  float64
 10  bedrooms_per_household    18391 non-null  float64
 11  population_per_household  18576 non-null  float64
dtypes: float64(11), object(1)
memory usage: 1.8+ MB
```

[83]:
```
X_train["ocean_proximity"].value_counts()
```

[83]:
```
<1H OCEAN     8221
INLAND        5915
NEAR OCEAN    2389
NEAR BAY      2046
ISLAND           5
Name: ocean_proximity, dtype: int64
```

[84]:
```
numeric_transformer = make_pipeline(SimpleImputer(strategy="median"),␣
 ↪StandardScaler())
categorical_transformer = OneHotEncoder(handle_unknown="ignore")

preprocessor = make_column_transformer(
    (numeric_transformer, numeric_features),
    (categorical_transformer, categorical_features),
)
```

[85]:
```
preprocessor
```

[85]:
```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
SimpleImputer(strategy='median')),
                                                 ('standardscaler',
```

```
                                        StandardScaler())]),
                                 ['longitude', 'latitude', 'housing_median_age',
                                  'total_rooms', 'total_bedrooms', 'population',
                                  'households', 'median_income',
                                  'rooms_per_household',
                                  'bedrooms_per_household',
                                  'population_per_household']),
                                ('onehotencoder',
                                 OneHotEncoder(handle_unknown='ignore'),
                                 ['ocean_proximity'])])
```

[86]:
```
X_train_pp = preprocessor.fit_transform(X_train)
```

- When we `fit` the preprocessor, it calls **`fit` on *all*** the transformers
- When we `transform` the preprocessor, it calls **`transform` on *all*** the transformers.

We can get the new names of the columns that were generated by the one-hot encoding:

[87]:
```
preprocessor
```

[87]:
```
ColumnTransformer(transformers=[('pipeline',
                                 Pipeline(steps=[('simpleimputer',
SimpleImputer(strategy='median')),
                                                 ('standardscaler',
                                                  StandardScaler())]),
                                 ['longitude', 'latitude', 'housing_median_age',
                                  'total_rooms', 'total_bedrooms', 'population',
                                  'households', 'median_income',
                                  'rooms_per_household',
                                  'bedrooms_per_household',
                                  'population_per_household']),
                                ('onehotencoder',
                                 OneHotEncoder(handle_unknown='ignore'),
                                 ['ocean_proximity'])])
```

[88]:
```
preprocessor.named_transformers_["onehotencoder"].get_feature_names_out(
    categorical_features
)
```

[88]:
```
array(['ocean_proximity_<1H OCEAN', 'ocean_proximity_INLAND',
       'ocean_proximity_ISLAND', 'ocean_proximity_NEAR BAY',
       'ocean_proximity_NEAR OCEAN'], dtype=object)
```

Combining this with the numeric feature names gives us all the column names:

[89]:
```
column_names = numeric_features + list(
    preprocessor.named_transformers_["onehotencoder"].get_feature_names_out(
        categorical_features
    )
```

```
)
column_names
```

[89]:
```
['longitude',
 'latitude',
 'housing_median_age',
 'total_rooms',
 'total_bedrooms',
 'population',
 'households',
 'median_income',
 'rooms_per_household',
 'bedrooms_per_household',
 'population_per_household',
 'ocean_proximity_<1H OCEAN',
 'ocean_proximity_INLAND',
 'ocean_proximity_ISLAND',
 'ocean_proximity_NEAR BAY',
 'ocean_proximity_NEAR OCEAN']
```

Let's visualize the preprocessed training data as a dataframe.

[90]:
```
pd.DataFrame(X_train_pp, columns=column_names)
```

[90]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	\
0	0.908140	-0.743917	-0.526078	0.143120	0.235339	
1	-0.002057	1.083123	-0.923283	-1.049510	-0.969959	
2	1.218207	-1.352930	1.380504	0.329670	0.549764	
3	1.128188	-0.753286	-0.843842	-0.459154	-0.626949	
4	1.168196	-1.287344	-0.843842	1.343085	2.210026	
…	…	…	…	…	…	
18571	0.733102	-0.804818	0.586095	-0.875337	-0.243446	
18572	1.163195	-1.057793	-1.161606	0.940194	0.609314	
18573	-1.097293	0.797355	-1.876574	0.695434	0.433046	
18574	-1.437367	1.008167	1.221622	-0.499947	-0.484029	
18575	0.242996	0.272667	-0.684960	-0.332190	-0.353018	

	population	households	median_income	rooms_per_household	\
0	1.026092	0.266135	-0.389736	-0.210591	
1	-1.206672	-1.253312	-0.198924	4.726412	
2	0.024896	0.542873	-0.635239	-0.273606	
3	-0.463877	-0.561467	0.714077	0.122307	
4	4.269688	2.500924	-1.059242	-0.640266	
…	…	…	…	…	
18571	-0.822136	-0.966131	-0.118182	0.063110	
18572	0.882438	0.728235	0.357500	0.235096	
18573	0.881563	0.514155	0.934269	0.211892	
18574	-0.759944	-0.454427	0.006578	-0.273382	

```
18575   -0.164307   -0.396991      -0.711754             0.025998

       bedrooms_per_household  population_per_household  \
0                   -0.083813                  0.126398
1                   11.166631                 -0.050132
2                   -0.025391                 -0.099240
3                   -0.280310                  0.010183
4                   -0.190617                  0.126808
…                         …                         …
18571               -0.099558                  0.071541
18572               -0.163397                  0.007458
18573               -0.135305                  0.044029
18574               -0.149822                 -0.132875
18575                0.042957                  0.051269

       ocean_proximity_<1H OCEAN  ocean_proximity_INLAND  \
0                            0.0                     1.0
1                            0.0                     1.0
2                            0.0                     0.0
3                            0.0                     1.0
4                            0.0                     0.0
…                            …                       …
18571                        1.0                     0.0
18572                        1.0                     0.0
18573                        1.0                     0.0
18574                        0.0                     0.0
18575                        0.0                     1.0

       ocean_proximity_ISLAND  ocean_proximity_NEAR BAY  \
0                         0.0                       0.0
1                         0.0                       0.0
2                         0.0                       0.0
3                         0.0                       0.0
4                         0.0                       0.0
…                         …                         …
18571                     0.0                       0.0
18572                     0.0                       0.0
18573                     0.0                       0.0
18574                     0.0                       1.0
18575                     0.0                       0.0

       ocean_proximity_NEAR OCEAN
0                              0.0
1                              0.0
2                              1.0
3                              0.0
4                              1.0
```

```
…                           …
18571                         0.0
18572                         0.0
18573                         0.0
18574                         0.0
18575                         0.0

[18576 rows x 16 columns]
```

[91]:
```
y_train.to_frame().head()
```

[91]:
```
       median_house_value
6051             113600.0
20113            137500.0
14289            170100.0
13665            129300.0
14471            205000.0
```

[92]:
```
results_dict = {}
dummy = DummyRegressor()
results_dict["dummy"] = mean_std_cross_val_scores(
    dummy, X_train, y_train, return_train_score=True
)
pd.DataFrame(results_dict).T
```

[92]:
```
                fit_time         score_time          test_score  \
dummy  0.001 (+/- 0.000)  0.000 (+/- 0.000)  -0.001 (+/- 0.001)

             train_score
dummy  0.000 (+/- 0.000)
```

[93]:
```
from sklearn.svm import SVR

knn_pipe = make_pipeline(preprocessor, KNeighborsRegressor())
```

[94]:
```
knn_pipe
```

[94]:
```
Pipeline(steps=[('columntransformer',
                 ColumnTransformer(transformers=[('pipeline',
Pipeline(steps=[('simpleimputer',
SimpleImputer(strategy='median')),
('standardscaler',
StandardScaler())]),
                                                 ['longitude', 'latitude',
                                                  'housing_median_age',
                                                  'total_rooms',
                                                  'total_bedrooms',
```

```
                                                  'population', 'households',
                                                  'median_income',
                                                  'rooms_per_household',
                                                  'bedrooms_per_household',
                                                  'population_per_household']),
                                                ('onehotencoder',
OneHotEncoder(handle_unknown='ignore'),
                                                 ['ocean_proximity'])])),
                ('kneighborsregressor', KNeighborsRegressor())])
```

[95]:
```
results_dict["imp + scaling + ohe + KNN"] = mean_std_cross_val_scores(
    knn_pipe, X_train, y_train, return_train_score=True
)
```

[96]:
```
pd.DataFrame(results_dict).T
```

[96]:
```
                                  fit_time         score_time  \
dummy                       0.001 (+/- 0.000)  0.000 (+/- 0.000)
imp + scaling + ohe + KNN   0.024 (+/- 0.002)  0.807 (+/- 0.136)

                                 test_score        train_score
dummy                       -0.001 (+/- 0.001)  0.000 (+/- 0.000)
imp + scaling + ohe + KNN    0.721 (+/- 0.012)  0.816 (+/- 0.006)
```

[97]:
```
svr_pipe = make_pipeline(preprocessor, SVR())
results_dict["imp + scaling + ohe + SVR (default)"] = mean_std_cross_val_scores(
    svr_pipe, X_train, y_train, return_train_score=True
)
```

[98]:
```
pd.DataFrame(results_dict).T
```

[98]:
```
                                           fit_time         score_time  \
dummy                                0.001 (+/- 0.000)  0.000 (+/- 0.000)
imp + scaling + ohe + KNN            0.024 (+/- 0.002)  0.807 (+/- 0.136)
imp + scaling + ohe + SVR (default)  12.143 (+/- 1.702)  3.046 (+/- 0.437)

                                          test_score        train_score
dummy                               -0.001 (+/- 0.001)  0.000 (+/- 0.000)
imp + scaling + ohe + KNN            0.721 (+/- 0.012)  0.816 (+/- 0.006)
imp + scaling + ohe + SVR (default) -0.049 (+/- 0.012) -0.049 (+/- 0.001)
```

The results with `scikit-learn`'s default SVR hyperparameters are pretty bad.

[99]:
```
svr_C_pipe = make_pipeline(preprocessor, SVR(C=10000))
results_dict["imp + scaling + ohe + SVR (C=10000)"] = mean_std_cross_val_scores(
    svr_C_pipe, X_train, y_train, return_train_score=True
)
```

[100]:
```
pd.DataFrame(results_dict).T
```

[100]:

	fit_time	score_time	test_score	train_score
dummy	0.001 (+/- 0.000)	0.000 (+/- 0.000)	-0.001 (+/- 0.001)	0.000 (+/- 0.000)
imp + scaling + ohe + KNN	0.024 (+/- 0.002)	0.807 (+/- 0.136)	0.721 (+/- 0.012)	0.816 (+/- 0.006)
imp + scaling + ohe + SVR (default)	12.143 (+/- 1.702)	3.046 (+/- 0.437)	-0.049 (+/- 0.012)	-0.049 (+/- 0.001)
imp + scaling + ohe + SVR (C=10000)	11.154 (+/- 1.224)	3.165 (+/- 0.114)	0.721 (+/- 0.007)	0.726 (+/- 0.007)

With a **bigger value for `C`** the results are much **better**. We need to carry out systematic **hyperparameter optimization** to get better results. (Coming up next week.)

- Note that categorical features are different than free text features. Sometimes there are columns containing free text information and we we'll look at ways to deal with them in the later part of this lecture.

1.6.1 OHE with many categories

- Do we have enough data for rare categories to learn anything meaningful?
- How about grouping them into bigger categories?
  - Example: country names into continents such as "South America" or "Asia"
- Or having "other" category for rare cases?

1.6.2 Do we actually want to use certain features for prediction?

- Do you ***want*** to use certain features such as **gender** or **race** in prediction?
- Remember that the systems you build are going to be used in some applications.
- It's extremely important to be mindful of the consequences of including certain features in your predictive model.

1.6.3 Preprocessing the targets?

- Generally no need for this when doing classification.
- In regression it makes sense in some cases. More on this later.
- `sklearn` is fine with categorical labels (y-values) for classification problems.

1.6.4 Questions for you

True/False: Categorical features

1. `handle_unknown="ignore"` would treat all unknown categories equally.
2. Creating groups of rarely occurring categories might overfit the model.

1.7 Encoding text data

[101]:
```
toy_spam = [
    [
        "URGENT!! As a valued network customer you have been selected to␣
 ↪receive a £900 prize reward!",
        "spam",
    ],
    ["Lol you are always so convincing.", "non spam"],
    ["Nah I don't think he goes to usf, he lives around here though", "non␣
 ↪spam"],
    [
        "URGENT! You have won a 1 week FREE membership in our £100000 prize␣
 ↪Jackpot!",
        "spam",
    ],
    [
        "Had your mobile 11 months or more? U R entitled to Update to the␣
 ↪latest colour mobiles with camera for Free! Call The Mobile Update Co FREE␣
 ↪on 08002986030",
        "spam",
    ],
    ["Congrats! I can't wait to see you!!", "non spam"],
]
toy_df = pd.DataFrame(toy_spam, columns=["sms", "target"])
```

1.7.1 Spam/non spam toy example

- What if the feature is in the form of raw text?
- The feature **sms** below is **neither categorical nor ordinal**.
- How can we encode it so that we can pass it to the machine learning algorithms we have seen so far?

[102]:
```
toy_df.style.set_properties(**{"text-align": "left"})
```

[102]:
```
<pandas.io.formats.style.Styler at 0x7f7caae7fcd0>
```

1.7.2 What if we apply OHE?

[103]:
```
### DO NOT DO THIS.
enc = OneHotEncoder(sparse=False)
transformed = enc.fit_transform(toy_df[["sms"]])
pd.DataFrame(transformed, columns=enc.categories_)
```

[103]:

	Congrats! I can't wait to see you!!	\
0	0.0	
1	0.0	
2	0.0	

```
3                                   0.0
4                                   0.0
5                                   1.0

  Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030  \
0
0.0
1
0.0
2
0.0
3
0.0
4
1.0
5
0.0

  Lol you are always so convincing.  \
0                                 0.0
1                                 1.0
2                                 0.0
3                                 0.0
4                                 0.0
5                                 0.0

  Nah I don't think he goes to usf, he lives around here though  \
0                                                         0.0
1                                                         0.0
2                                                         1.0
3                                                         0.0
4                                                         0.0
5                                                         0.0

  URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!  \
0                                                                          0.0
1                                                                          0.0
2                                                                          0.0
3                                                                          1.0
4                                                                          0.0
5                                                                          0.0

  URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
1.0
```

```
1
0.0
2
0.0
3
0.0
4
0.0
5
0.0
```

- We do **not have a fixed number** of categories here.
- Each "category" (feature value) is likely to **occur only once** in the training data and we won't learn anything meaningful if we apply one-hot encoding or ordinal encoding on this feature.
- How can we encode or represent **raw text data into fixed number of features** so that we can learn some useful patterns from it?

- This is a well studied problem in the field of ***Natural Language Processing (NLP)***, which is concerned with giving computers the ability to understand written and spoken language.
- Some popular representations of raw text include:
  - **Bag of words**
  - TF-IDF
  - Embedding representations

1.7.3 Bag of words (BOW) representation

- One of the most popular representation of raw text
- Ignores the syntax and word order
- It has two components:
  - The vocabulary (all unique words in all documents)
  - A value indicating either the presence or absence or the count of each word in the document.

Source

1.7.4 Extracting BOW features using `scikit-learn`

- `CountVectorizer`
  - Converts a collection of text documents to a matrix of word counts.
  - Each row represents a "document" (e.g., a text message in our example).
  - Each column represents a word in the vocabulary (the set of unique words) in the training data.
  - Each cell represents how often the word occurs in the document.

Note In the Natural Language Processing (NLP) community text data is referred to as a **corpus** (plural: corpora).

[104]:
```
from sklearn.feature_extraction.text import CountVectorizer

vec = CountVectorizer()
X_counts = vec.fit_transform(toy_df["sms"])
bow_df = pd.DataFrame(
    X_counts.toarray(), columns=vec.get_feature_names_out(), index=toy_df["sms"]
)
bow_df
```

[104]:
```
08002986030  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

100000  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                        11
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
```

```
prize reward!                                                                0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030   1
Congrats! I can't wait to see you!!
0

                                                                           900
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

always  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                           are
\
```

```
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                  0
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

around  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
1
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                            as
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                  1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030   0
Congrats! I can't wait to see you!!
0
```

been
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

call
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

…
\
sms
…
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!
…
Lol you are always so convincing.
…
Nah I don't think he goes to usf, he lives around here though
…
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!

```
…
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030  …
Congrats! I can't wait to see you!!
…

update  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
2
Congrats! I can't wait to see you!!
0

urgent  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                          usf
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                               0
Lol you are always so convincing.
0
```

```
Nah I don't think he goes to usf, he lives around here though
1
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

valued  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                          wait
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
1

                                                                          week
\
sms
```

```
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                          with
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

                                                                          won
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                               0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
```

```
0

                                                                                you
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                    1
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
1

                                                                               your
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

[6 rows x 61 columns]
```

1.7.5 Input to `CountVectorizer.fit_transform`

[105]:
```
type(toy_df["sms"])
```

[105]:
```
pandas.core.series.Series
```

Important Note that unlike other transformers we are **passing a `Series`** object to `fit_transform`. For other transformers, you can define one transformer for more than one columns. But with `CountVectorizer` you need to define **separate `CountVectorizer` transformers for each text column**, if you have more than one text columns.

1.7.6 Output of `CountVectorizer.fit_transform`

`fit_transform` has returned a sparse matrix:

[106]:
```
X_counts
```

[106]:
```
<6x61 sparse matrix of type '<class 'numpy.int64'>'
        with 71 stored elements in Compressed Sparse Row format>
```

Why sparse matrices?

- Most words do not appear in a given document.
- We get massive computational savings if we **only store the nonzero elements**.
- There is a bit of overhead, because we also need to store the locations:
  - e.g. "location (3,27): 1".
- However, if the fraction of nonzero is small, this is a huge win.

[107]:
```
print("The number of rows and columns: ", *X_counts.shape)
print("The total number of elements: ", np.prod(X_counts.shape))
print("The number of non-zero elements: ", X_counts.nnz)
print(
    "Proportion of non-zero elements: %0.4f" % (X_counts.nnz / np.prod(X_counts.
 ↪shape))
)
print(
    "The value at cell (4,%d) is: %d"
    % (vec.vocabulary_["update"], X_counts[4, vec.vocabulary_["update"]])
)
```

```
The number of rows and columns:  6 61
The total number of elements:  366
The number of non-zero elements:  71
Proportion of non-zero elements: 0.1940
The value at cell (4,51) is: 2
```

Question for you - What would happen if you apply `StandardScaler` on sparse data?

1.7.7 `OneHotEncoder` and sparse features

- By default, `OneHotEncoder` also creates sparse features.
- You could set `sparse=False` to get a regular `numpy` array.
- If there are a huge number of categories, it may be beneficial to keep them sparse.
- For smaller number of categories, it doesn't matter much.

1.7.8 Important hyperparameters of `CountVectorizer`

- `binary`
  - whether to use absence/presence feature values or counts
- `max_features`
  - only consider top `max_features` ordered by frequency in the corpus
- `max_df`

  - max document frequency, ignore features which occur in more than `max_df` documents
- `min_df`
  - min document frequency, ignore features which occur in less than `min_df` documents
- `ngram_range`
  - consider word sequences in the given range

Let's look at all features, i.e., words (along with their frequencies).

[108]:
```
vec = CountVectorizer()
X_counts = vec.fit_transform(toy_df["sms"])
bow_df = pd.DataFrame(
    X_counts.toarray(), columns=vec.get_feature_names_out(), index=toy_df["sms"]
)
print("Max value: ", bow_df.max().max())
bow_df
```

```
Max value:  2
```

[108]:
```
08002986030  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

100000  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
```

```
0
Congrats! I can't wait to see you!!
0

                                                                      11
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                          0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030   1
Congrats! I can't wait to see you!!
0

                                                                     900
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                          1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

always  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
```

Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

are
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward! 0
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030 0
Congrats! I can't wait to see you!!
0

around \
sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
1
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

as
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward! 1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though

0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030   0
Congrats! I can't wait to see you!!
0

been
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

call
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

…
\
sms

```
…
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                 …
Lol you are always so convincing.
…
Nah I don't think he goes to usf, he lives around here though
…
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
…
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030  …
Congrats! I can't wait to see you!!
…

update  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
2
Congrats! I can't wait to see you!!
0

urgent  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0
```

```
                                                                                usf
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                    0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
1
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

valued  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                               wait
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
```

mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
1

week
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

with
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

won
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward! 0
Lol you are always so convincing.

```
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

                                                                              you
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                 1
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
1

                                                                              your
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

[6 rows x 61 columns]
```

When we use `binary=True`, the representation uses presence/absence of words instead of word counts.

[109]:
```
vec_binary = CountVectorizer(binary=True)
X_counts = vec_binary.fit_transform(toy_df["sms"])
bow_df = pd.DataFrame(
    X_counts.toarray(), columns=vec_binary.get_feature_names_out(),␣
 ↪index=toy_df["sms"]
)
print("Max value: ", bow_df.max().max())
bow_df
```

```
Max value:  1
```

[109]:
```
08002986030  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

100000  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                        11

\
```

```
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030   1
Congrats! I can't wait to see you!!
0

                                                                          900
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

always  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0
```

```
                                                                                are
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                    0
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

around  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
1
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                                as
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                                    1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030   0
Congrats! I can't wait to see you!!
```

0

been
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

call
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

…
\
sms
…
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!
…
Lol you are always so convincing.
…
Nah I don't think he goes to usf, he lives around here though

```
…
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
…
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030  …
Congrats! I can't wait to see you!!
…

update  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

urgent  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                          usf
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                               0
```

```
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
1
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

valued  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                          wait
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
1

                                                                          week
```

\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

with
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

won
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour

```
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
0

                                                                          you
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                               1
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
1

                                                                         your
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
1
Congrats! I can't wait to see you!!
0

[6 rows x 61 columns]
```

We can control the size of X (the number of features) using `max_features`.

[110]:
```
vec8 = CountVectorizer(max_features=8)
X_counts = vec8.fit_transform(toy_df["sms"])
bow_df = pd.DataFrame(
    X_counts.toarray(), columns=vec8.get_feature_names_out(),␣
 ↪index=toy_df["sms"]
)
```

```
print("Max value: ", bow_df.max().max())
print(bow_df.max())
bow_df
```

```
Max value:  2
free      2
have      1
mobile    2
the       2
to        2
update    2
urgent    1
you       1
dtype: int64
```

[110]:

	free	\	sms	have	\	sms
URGENT!! As a valued network customer you have been selected to receive a £900 prize reward!	0			1		
Lol you are always so convincing.	0			0		
Nah I don't think he goes to usf, he lives around here though	0			0		
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!	1			1		
Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030	2			1		
Congrats! I can't wait to see you!!	0					

```
0
Congrats! I can't wait to see you!!
0

mobile  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
2
Congrats! I can't wait to see you!!
0

                                                                          the
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                               0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    2
Congrats! I can't wait to see you!!
0

                                                                          to
\
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                               1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
1
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
```

```
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030   2
Congrats! I can't wait to see you!!
1

update  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
0
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
0
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
2
Congrats! I can't wait to see you!!
0

urgent  \
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!
1
Lol you are always so convincing.
0
Nah I don't think he goes to usf, he lives around here though
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
0
Congrats! I can't wait to see you!!
0

                                                                          you
sms
URGENT!! As a valued network customer you have been selected to receive a £900
prize reward!                                                               1
Lol you are always so convincing.
1
Nah I don't think he goes to usf, he lives around here though
```

```
0
URGENT! You have won a 1 week FREE membership in our £100000 prize Jackpot!
1
Had your mobile 11 months or more? U R entitled to Update to the latest colour
mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030    0
Congrats! I can't wait to see you!!
1
```

Note (Optional)

Notice that `vec8` and `vec8_binary` have different vocabularies, which is kind of unexpected behaviour and doesn't match the documentation of `scikit-learn`.

Here is the code for `binary=True` condition in `scikit-learn`. As we can see, the binarization is done before limiting the features to `max_features`, and so now we are actually looking at the document counts (in how many documents it occurs) rather than term count. This is not explained anywhere in the documentation.

The ties in counts between different words makes it even more confusing. I don't think it'll have a big impact on the results but this is good to know! Remember that `scikit-learn` developers are also humans who are prone to make mistakes. So it's always a good habit to question whatever tools we use every now and then.

[111]:
```
vec8 = CountVectorizer(max_features=8)
X_counts = vec8.fit_transform(toy_df["sms"])
bow_df = pd.DataFrame(
    X_counts.toarray(), columns=vec8.get_feature_names_out(),␣
 ↪index=toy_df["sms"]
)
bow_df.sum().sort_values(ascending=False).rename("counts").to_frame()
```

[111]:
```
        counts
to           5
you          4
free         3
have         2
mobile       2
the          2
update       2
urgent       2
```

[112]:
```
vec8_binary = CountVectorizer(binary=True, max_features=8)
X_counts = vec8_binary.fit_transform(toy_df["sms"])
bow_df = pd.DataFrame(
    X_counts.toarray(), columns=vec8_binary.get_feature_names_out(),␣
 ↪index=toy_df["sms"]
)
bow_df.sum().sort_values(ascending=False).rename("counts").to_frame()
```

[112]:
```
          counts
to             4
you            4
free           2
have           2
prize          2
urgent         2
mobiles        1
months         1
```

1.7.9 Preprocessing

- Note that `CountVectorizer` is carrying out some preprocessing such as the following because of the default argument values:
  - Converting words to lowercase (`lowercase=True`)
  - getting rid of punctuation and special characters (`token_pattern ='(?u)\\b\\w\\w+\\b'`)

[113]:
```
pipe = make_pipeline(CountVectorizer(), SVC())
```

[114]:
```
pipe.fit(toy_df["sms"], toy_df["target"])
```

[114]:
```
Pipeline(steps=[('countvectorizer', CountVectorizer()), ('svc', SVC())])
```

[115]:
```
pipe.predict(toy_df["sms"]).tolist()
```

[115]:
```
['spam', 'non spam', 'non spam', 'spam', 'spam', 'non spam']
```

[116]:
```
toy_df["target"].tolist()
```

[116]:
```
['spam', 'non spam', 'non spam', 'spam', 'spam', 'non spam']
```

1.7.10 Is this a realistic representation of text data?

- Of course this is not a great representation of language
  - We are throwing out everything we know about language and losing a lot of information.
  - It assumes that there is **no syntax and compositional meaning** in language.

- But it **works surprisingly well** for many tasks.
- We will learn more expressive representations in the coming weeks.

1.7.11 Questions for you

`CountVectorizer`: True or False

1. As you increase the value for `max_features` hyperparameter of `CountVectorizer` the training score is likely to go up.
2. Suppose you are encoding text data using `CountVectorizer`. If you encounter a word in the validation or the test split that's not available in the training data, we'll get an error.

3. `max_df` hyperparameter of `CountVectorizer` can be used to get rid of most frequently occurring words from the dictionary.
4. In the code below, inside `cross_validate`, each fold might have slightly different number of features (columns) in the fold.

```
pipe = (CountVectorizer(), SVC())
cross_validate(pipe, X_train, y_train)
```

Identify column transformations Consider the restaurant data from the survey you did a few weeks ago.

[117]:
```
restaurant_data = pd.read_csv("data/cleaned_restaurant_data.csv")
restaurant_data.head(10)
```

[117]:

	north_america	eat_out_freq	age	n_people	price	food_type	noise_level	good_server	comments
0	Yes	3.0	29	20	10	Other	no music	Yes	NaN
1	Yes	2.0	23	10	20	Chinese	low	Yes	food tastes good
2	Yes	2.0	21	10	40	Other	low	Yes	good food
3	No	2.0	24	15	40	Other	medium	Yes	My love for Korean food
4	Yes	5.0	23	20	10	Canadian/American	low	Yes	Good food
5	Yes	2.0	22	60	20	Chinese	medium	Yes	food tasted great!
6	Yes	2.0	23	10000000	1000000	Canadian/American	crazy loud	Yes	NaN
7	Yes	4.0	20	20	40	Chinese	high	Yes	NaN
8	Yes	1.0	21	40	60	Italian	high	Yes	NaN
9	No	2.0	20	50	45	Canadian/American	medium	Yes	

```
9  Steak was good, ambiance was nice and server was helpful.
```

```
    restaurant_name target
0               NaN   like
1             Midam   like
2         pear tree   like
3              Dami   like
4               NaN   like
5               NaN   like
6  CACTUS CLUB CAFE   like
7               NaN   like
8         Frankie’s   like
9   Hy's steakhouse   like
```

What all feature transformations you would apply on this dataset?

1.8 What did we learn today?

- Motivation to use `ColumnTransformer`
- `ColumnTransformer` syntax
- Defining transformers with multiple transformations
- How to visualize transformed features in a dataframe
- More on ordinal features
- Different arguments `OneHotEncoder`
  - `handle_unknow="ignore"`
  - `if_binary`
- Dealing with text features
  - Bag of words representation: `CountVectorizer`


TALK
soon