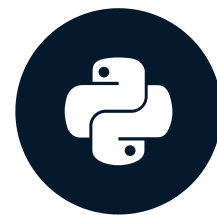


Feature engineering

PREPROCESSING FOR MACHINE LEARNING IN PYTHON



What is feature engineering?

- Creation of new features based on existing features
- Insight into relationships between features
- Extract and expand data
- Dataset-dependent

Feature engineering scenarios

Id	Text
1	"Feature engineering is fun!"
2	"Feature engineering is a lot of work."
3	"I don't mind feature engineering."

user	fav_color
1	blue
2	green
3	orange

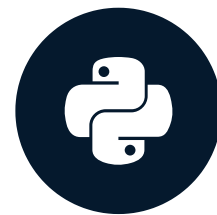
Feature engineering scenarios

Id	Date
4	July 30 2011
5	January 29 2011
6	February 05 2011

user	test1	test2	test3
1	90.5	89.6	91.4
2	65.5	70.6	67.3
3	78.1	80.7	81.8

Encoding categorical variables

PREPROCESSING FOR MACHINE LEARNING IN PYTHON



Categorical variables

```
   user subscribed fav_color
0     1         y    blue
1     2         n   green
2     3         n  orange
3     4         y   green
```

Encoding binary variables - Pandas

```
print(users["subscribed"])
```

```
0    y
1    n
2    n
3    y
Name: subscribed, dtype: object
```

```
print(users[["subscribed", "sub_enc"]])
```

	subscribed	sub_enc
0	y	1
1	n	0
2	n	0
3	y	1

```
users["sub_enc"] = users["subscribed"].apply(lambda val:
                                              1 if val == "y" else 0)
```

Encoding binary variables - scikit-learn

```
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()
```

```
users["sub_enc_le"] = le.fit_transform(users["subscribed"])
```

```
print(users[["subscribed", "sub_enc_le"]])
```

	subscribed	sub_enc_le
0	y	1
1	n	0
2	n	0
3	y	1

One-hot encoding

fav_color
blue
green
orange
green

fav_color_enc
[1, 0, 0]
[0, 1, 0]
[0, 0, 1]
[0, 1, 0]

Values: [blue, green, orange]

- blue: [1, 0, 0]
- green: [0, 1, 0]
- orange: [0, 0, 1]

```
print(users["fav_color"])
```

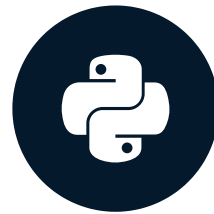
```
0    blue
1   green
2  orange
3   green
Name: fav_color, dtype: object
```

```
print(pd.get_dummies(users["fav_color"]))
```

	blue	green	orange
0	1	0	0
1	0	1	0
2	0	0	1
3	0	1	0

Engineering numerical features

PREPROCESSING FOR MACHINE LEARNING IN PYTHON



```
print(df)
```

```
   city  day1  day2  day3
0   NYC  68.3  67.9  67.8
1    SF  75.1  75.5  74.9
2    LA  80.3  84.0  81.3
3 Boston  63.0  61.0  61.2
```

```
columns = ["day1", "day2", "day3"]
df["mean"] = df.apply(lambda row: row[columns].mean(), axis=1)
print(df)
```

```
   city  day1  day2  day3  mean
0   NYC  68.3  67.9  67.8  68.00
1    SF  75.1  75.5  74.9  75.17
2    LA  80.3  84.0  81.3  81.87
3 Boston  63.0  61.0  61.2  61.73
```

Dates

```
print(df)
```

```
      date purchase
0  July 30 2011  $45.08
1 February 01 2011  $19.48
2 January 29 2011  $76.09
3  March 31 2012  $32.61
4 February 05 2011  $75.98
```

Dates

```
df["date_converted"] = pd.to_datetime(df["date"])
```

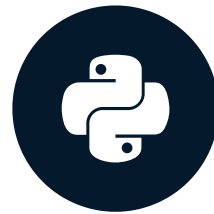
```
df["month"] = df["date_converted"].apply(lambda row: row.month)
```

```
print(df)
```

			date	purchase	date_converted	month
0	July	30	2011	\$45.08	2011-07-30	7
1	February	01	2011	\$19.48	2011-02-01	2
2	January	29	2011	\$76.09	2011-01-29	1
3	March	31	2012	\$32.61	2012-03-31	3
4	February	05	2011	\$75.98	2011-02-05	2

Engineering features from text

PREPROCESSING FOR MACHINE LEARNING IN PYTHON



EXtraction

```
import re
```

```
my_string = "temperature:75.6 F"
```

```
pattern = re.compile("\d+\.\d+")
```

```
temp = re.match(pattern,  
                my_string)
```

```
print(float(temp.group(0)))
```

```
75.6
```

- `\d+`
- `\.`
- `\d+`

Vectorizing text

- tf = term frequency
- idf = inverse document frequency

vectorizing text

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
print(documents.head())
```

```
0    Building on successful events last summer and ...  
1           Build a website for an Afghan business  
2    Please join us and the students from Mott Hall...  
3    The Oxfam Action Corps is a group of dedicated...  
4    Stop 'N' Swap reduces NYC's waste by finding n...
```

```
tfidf_vec = TfidfVectorizer()  
text_tfidf = tfidf_vec.fit_transform(documents)
```

Text Classification

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$