Aprendizado de Supervisionado

Fabrício Olivetti de França

Universidade Federal do ABC

Tópicos

1. Padronizando e Normalizando os Atributos
2. Tipos de Atributos
3. Representação Textual

UFABC

Padronizando e Normalizando os Atributos

Padronização

Muitos algoritmos de Aprendizado de Máquina supõem que os valores dos atributos seguem $N(0, 1)$.

Se um atributo não segue esse padrão, pode dominar a função-objetivo e se tornar importante demais.

Padronização

Dado uma matriz de dados X, podemos padronizar os valores de cada um de seus elementos como:

$$\hat{X}_{i,j} = \frac{X_{i,j} - \bar{X}_{i,j}}{\sigma_j}$$

Escalonamento

Algoritmos baseados em métodos de gradiente tendem a se beneficiar quando os atributos estão entre $[0, 1]$.

$$\hat{X}_{i,j} = \frac{X_{i,j} - \min X_{:,j}}{\max X_{:,j} - \min X_{:,j}}$$

UFABC

Normalização

Finalmente podemos normalizar cada amostra da base utilizando a normalização de vetores:

$$\hat{x}_{i,j} = \frac{X_{i,j}}{\|X_i\|_p}$$

UFABC

Tipos de Atributos

Tipos de Atributos

- Categóricos:
  - Nominal
  - Binário
  - Ordinal
- Numéricos:
  - Intervalar
  - Razão

UFABC

Atributos Nominais

Símbolos ou nomes descritivos:

- Descreve uma característica
- Não apresentam relação de ordem, apenas igualdade

Ex.: ocupação, cor dos olhos, etc.

Atributos Binários

Atributo nominal com apenas dois valores possíveis: 0 ou 1.

Ex.: resultado de um exame, gênero, etc.

UFABC

Atributos Ordinais

Valores que apresentam ordem, mas não possuem relação de magnitude.

Ex.: {péssimo, ruim, regular, bom, ótimo}

UFABC

Atributos Intervalares

Atributo numérico com relação de ordem e magnitude.

Ex.: Temperatura de $20°C$ é 10 unidades mais quente do que $10°C$.

Não podemos dizer que $20°C$ é duas vezes mais quente do que $10°C$.

UFABC

Atributos Intervalares

O **0** absoluto em Celsius está em $-273.15°C$.

$20°C$ está a 293.15 de nenhum calor, o dobro desse calor seria 586.30 ou $313.15°C$.

UFABC

Atributos de Razões

Atributos numéricos com a mesmas propriedades do intervalar, mas com um ponto-zero.

- Altura
- Contagem de palavras
- Temperatura em Kelvin

Representando Objetos

Precisamos de uma representação numérica.

- Dados numéricos: ok! (ou quase sempre ok)
- Dados nominais: vetor binário
- Dados ordinais: rank

UFABC

Representando Objetos

Tabela 1: Base de Dados.

Profissão	**Conceito**	**Nota**
Engenheiro	A	9.5
Professor	B	8.4
Engenheiro	A	7.3
Gerente	D	9.1

UFABC

Atributos Categóricos $\rightarrow$ Binários

Tabela 2: Base de Dados.

Engenheiro	Professor	Gerente	Conceito	Nota
1	0	0	A	9.5
0	1	0	B	8.4
1	0	0	A	7.3
0	0	1	D	9.1

UFABC

Atributos Categóricos → Binários - Scikit-Learn

```
from sklearn.feature_extraction import DictVectorizer

enc = DictVectorizer()
D = [{'Profissao': 'Engenheiro'}, {'Profissao':'Professor'}, {'P
enc.fit(D)
xi = enc.transform([{'Profissao' : 'Engenheiro'}])

xi.toarray() # [1,0,0]
```

UFABC

Atributos Ordinais $\rightarrow$ Numéricos

$\#$ conceitos $= 5$

Rank: $5, 4, 3, 2, 1$

$z = \frac{(rank - 1)}{\# - 1}$

Tabela 3: Base de Dados.

Engenheiro	Professor	Gerente	Conceito	Nota
1	0	0	1	9.5
0	1	0	0.75	8.4
1	0	0	1	7.3
0	0	1	0.25	9.1

Representação Textual

Mineração de Textos

Documentos de textos:

- Não possuem representação vetorial
- Interdependência dos atributos
- Tamanho variável

UFABC

Textos como conjuntos

Se representarmos os documentos de textos como o conjunto de suas palavras:

D1 = "Estou assistindo a uma aula de Big Data, mas tudo que aprendi foi Haskell durante a aula!"

D2 = "Hoje aprendi Haskell na aula, será que o que aprendi será útil na minha vida?"

UFABC

Textos como conjuntos

Se representarmos os documentos de textos como o conjunto de suas palavras:

$$D1 = \{Estou, assistindo, a, uma, aula, de, Big, Data, mas, tudo, que, aprendi, foi, Haskell, durante, aula!\}$$
$$D2 = \{Hoje, aprendi, Haskell, na, aula, será, que, o, útil, minha, vida?\}$$

UFABC

Textos como conjuntos

Calculando a similaridade de Jaccard, temos:

$$D1 \cap D2 = \{que, aprendi, Haskell\}$$

$$D1 \cup D2 = \{Estou, assistindo, a, uma, aula, de, Big, Data, mas, tudo, que, aprendi, foi, Haskell, durante, aula!, Hoje, na, aula, será, o, útil, minha, vida?\}$$

$$J(D1, D2) = \frac{3}{24} = 0.125$$

Essa representação é conhecida como Bag-of-Words, em que geramos os atributos do texto como atributos categóricos.

Normalização do Texto

Pode ser interessante padronizar a forma do texto para termos serem considerados como um elemento único do conjunto independente de como é escrito.

Por exemplo: Estou, estou, esTou, aula!, aula, aula?, útil, util.

$$
\begin{aligned}
D1 &= \{estou, assistindo, a, uma, aula, de, big, data, mas, tudo, que, aprendi, \\
&\quad foi, haskell, durante, aula\} \\
D2 &= \{hoje, aprendi, haskell, na, aula, sera, que, o, util, minha, vida\} \\
J(D1, D2) &= \frac{4}{23} = 0.17
\end{aligned}
$$

UFABC

Eliminação de atributos irrelevantes

Podemos eliminar palavras que não apresentam significado sozinhas:

$$D1 = \{estou, assistindo, aula, big, data, tudo, aprendi, haskell, durante, aula\}$$
$$D2 = \{hoje, aprendi, haskell, aula, sera, util, minha, vida\}$$
$$J(D1, D2) = \frac{4}{14} = 0.28$$

UFABC

Term-Frequency

Se um termo aparece repetidas vezes em um documento, isso significa que ele pode ter uma importância maior do que os outros termos.

No nosso exemplo, a repetição do termo *Haskell* indica um dos temas dos nossos documentos.

Term-Frequency

A informação de frequência pode ser importante para a representação de nossos documentos. Podemos fazer então:

$$fn(t, d) = \frac{f(t, d)}{|d|},$$

com $f(t, d)$ sendo a frequência do termo t no documento d e $|d|$ a quantidade de termos no documento d.

Inverse Document Frequency

Algumas palavras aparecem com uma frequência muito superior as demais, como: e, que, ou, etc.

Essas palavras não costumam apresentar um significado discriminatório e, portanto, podem ter um peso menor. Para isso podemos multiplicar o TF por:

$$idf(t) = \log \frac{|D|}{|\{d \in D : t \in D\}|}$$

UFABC

TFIDF - Scikit-Learn

```
from sklearn.feature_extraction.text import TfidfVectorizer

D = ["Essa aula é muito legal",
     "Gosto muito da aula de Aprendizado",
     "Aprendizado é importante e muito legal como aula"]

tf = TfidfVectorizer(use_idf=False)
tfidf = TfidfVectorizer(use_idf=True)
X = tf.fit_transform(D)
print(X.toarray())

X = tfidf.fit_transform(D)
print(X.toarray())
```

UFABC

Próxima Aula

Na próxima aula aprenderemos sobre:

- Regressão Linear
- Ordinary Least Square
- Gradiente Descendente

UFABC

Atividade 02

Complete o Laboratório
Supervised_Learning_and_K_Nearest_Neighbors.ipynb.

UFABC

Atividade 02

Dicas de funções, métodos, bibliotecas:

```
import pandas as pd

df = pd.read_csv('arquivo.csv') # lê arquivo CSV

# Escalonamento
from sklearn.preprocessing import StandardScaler, MinMaxScaler

stdSc = StandardScaler()
X_scaled = stdSc.fit_transform(X)

minMax = MinMaxScaler(feature_range=(0,1))
X_minmax = minMax.fit_transform(X)
```

UFABC

Atividade 02

Dicas de funções, métodos, bibliotecas:

```
# kNN
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=3,
                           metric='minkowski',
                           p=2)
knn.fit(X_scaled, y)
yi = knn.predict(xi)
```