Processamento de dados em Python

Parte 1/3 do curso de visualização computacional

Estagiário PAE: Eric Macedo Cabral
cabral.eric@usp.br

Docente: Maria Cristina Ferreira de Oliveira
cristina@icmc.usp.br

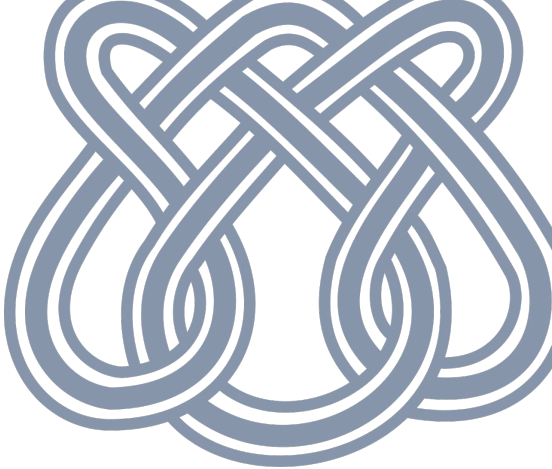
Motivação

- A análise de grandes volumes de dados pode trazer grandes *insights*
- Dados coletados, naturalmente possuem ruído
  - A tarefa de limpar os dados é inviável de ser realizada manualmente (na grande maioria dos casos)
- Uma única instância pode ser descrita por dois ou até milhares de atributos
  - Maldição da dimensionalidade

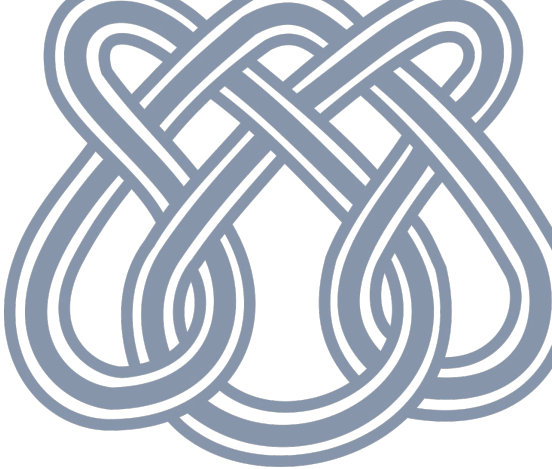
O que aprenderemos neste modulo?

- Coletar conjuntos de dados disponíveis gratuitamente
- Explorar e manipular estes dados
- Limpeza de dados
- Transformação de dados
- Técnicas básicas de agrupamento de dados

Sumário



1 Coleta



1. Seaborn
2. Scikit-Learn
3. Arquivos locais

Seaborn

```
pip install seaborn
```

Jupyter Notebook:

```
import seaborn as sns
sns.load_dataset(dataset)
```

Os dados são carregados no formato pd.Dataframe

Site:

- https://seaborn.pydata.org/

Datasets:

- https://github.com/mwaskom/seaborn-data

Scikit-Learn

```
pip install scikit-learn
```

Jupyter Notebook:

```
from sklearn.datasets import dataset
```

Os dados são carregados no formato sklearn.utils.Bunch, uma espécie de dicionário

Site:

- https://scikit-learn.org/

Datasets:

- https://scikit-learn.org/stable/datasets/index.html

Arquivos locais CSV

```
pip install pandas
```

Jupyter Notebook:

```
import pandas as pd
pd.read_csv(file_path)
```

Os dados são carregados no formato pd.Dataframe

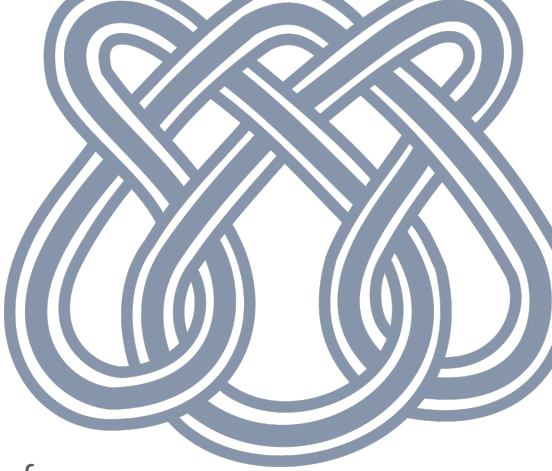
Site:

- https://pandas.pydata.org/

Documentação

- https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html

2 Gerenciamento e manipulação



1. Pandas Dataframe
2. Seleção de dados
3. Agrupamento
4. Funções

Pandas Dataframe I

- Matriz bidimensional de dados rotulados por linhas e colunas
- Suporta dados heterogêneos
- Pode ser vista como um dicionário de `pd.Series`
  - A estrutura de dados sequencial base do Pandas

```
>>> import pandas as pd

>>> df = pd.DataFrame(data=np.array([
…      [1, 2, 3],
…      [4, 5, 6],
…      [7, 8, 9]]), columns=['a', 'b', 'c'])
>>> df
   a  b  c
0  1  2  3
1  4  5  6
2  7  8  9
```

Pandas Dataframe II

- `pd.Dataframe.shape`: a dimensionaliade do Dataframe
- `pd.Dataframe.head(n)`: n primeiras instâncias
- `pd.Dataframe.tail(n)`: n últimas instâncias
- `pd.Dataframe.values`: representação `np.ndarray` do Dataframe
- `pd.Dataframe.to_pickle()`: salva o DataFrame em disco (formato binário)
- `pd.Dataframe.read_pickle()`: carrega o DataFrame do disco (formato binário)

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html

https://docs.python.org/3/library/pickle.html

Seleção de dados I

- Seleção simples de colunas:

```
pd.Dataframe[“col_label”]
```

- Seleção de múltiplas colunas:

```
pd.Dataframe[[“col_label_1”, “col_label_2”]]
```

Seleção de dados II

- Seleção por correspondência de índice (*hashing*):

```
pd.Dataframe.loc[[row_labels], [column_labels]]
```

- Seleção por posição sequencial (lista ordenada):

```
pd.Dataframe.iloc[[row_labels], [column_labels]]
```

Seleção de dados III

- Seleção por condicional simples:

```
pd.Dataframe[condicional]
```

- Seleção por múltiplas condicionais:

```
pd.Dataframe[
        (condicional_1)|
        (condicional_2)]
```

- Operadores condicionais:
  - Negação: ~
  - E lógico: &
  - OU lógico: |

Agrupamento

- Agrupar dados por coluna:

```
pd.Dataframe.groupby(["column_label"])
```

Funções

- Funções agregadas:

```
pd.Dataframe.agg([func])
```

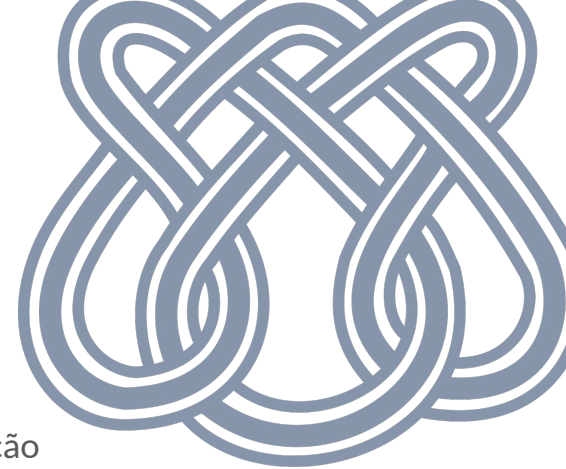
- Broadcasting:

```
pd.Dataframe.apply([func_1, func_2])
```

```
pd.Dataframe[“numeric_col”] * 2
```

exemplos/01.ipynb

3 Processamento e transformações



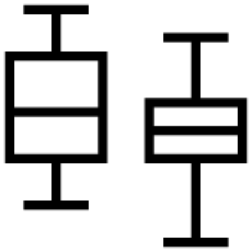
1. Sumarização
2. Dados ausentes
3. Escalonamento
4. Detecção de *outliers*
5. Redução de dimensionalidade

Conheça seus dados!

- Qual o domínio do seu conjunto de dados?
  - Que tipo de tratamento esse conjunto de dados deve receber
- Como seus dados estão estruturados?
- Existem metadados?
- Existem valores ausentes?
  - Como tratá-los sem causar ruídos?
- Existem anômalos?
- Existem atributos correlacionados ou redundantes?

Sumarização

- Medidas estatísticas:
  - Média
  - Desvio padrão
  - Mediana
  - etc...
- Recursos visuais
  - Box-plot
  - Histograma
  - WordCloud (dados textuais)



Dados ausentes

- Descarta linhas com valores ausentes

```
pd.DataFrame.dropna()
```

- Valor sentinela e imputação por média

```
pd.DataFrame.fillna()
```

- Imputação por interpolação

```
pd.DataFrame.interpolate()
```

- Imputação por vizinhos mais próximos

```
from sklearn.impute import KNNImputer
```

Documentação KNNImputer scikit-learn

Escalonamento I

- Trazer os dados para a mesma ordem de grandeza
- Modifica os valores originais

Escalonamento II
Uniformização (z-score)

- *"Standardization"*
- Média = 0
- Desvio padrão = 1

$$x = \frac{x - \mu}{\sigma}$$

```
from sklearn import preprocessing
scaler = preprocessing.StandartScaler()
X_scaled = scaler.fit_transform(X)

ou

from scipy.stats import zscore
pd.DataFrame.apply(zscore)
```

Escalonamento III
Mínimo e máximo

$$X\_mm = \left(\frac{x - x_{min}}{x_{max} - x_{min}} \right) * (max - min) + min$$

- Leva os dados para um novo intervalo
- Definição de mínimo e máximo

```
from sklearn import preprocessing
scaler = preprocessing.MinMaxScaler()
X_scaled = scaler.fit_transform(X)
```

Escalonamento IV
Normalização (l^1, l^2)

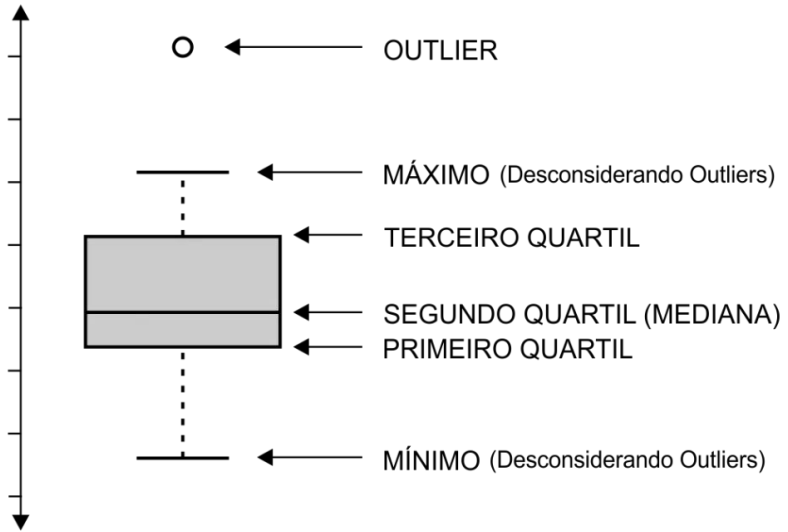
- Escalona as instâncias para a norma unitária
  - É executado linha a linha
- Útil para casos onde a forma quadrática pode ser empregada

```
from sklearn import preprocessing
normalizer = preprocessing.Normalizer()
X_norm = normalizer.fit_transform(X)
```

Detecção de Outliers

Outlier é uma instância de dado significativamente distante das demais instâncias (Wikipédia)

- Visualização
  - Scatter plot
  - Boxplot
- Z-score
- Interquartile Range (IQR)
  - A diferença entre o quartil superior e o quartil inferior
  - É uma medida de dispersão robusta contra outliers



Fonte: abgconsultoria.com.br

Redução de dimensionalidade I

- Centenas ou milhares de atributos
  - Maldição da dimensionalidade
- Representação visual de dados normalmente é bi ou tridimensional
- Preservação de informação
  - Relação e/ou correlação entre as instâncias
  - Vizinhança

Redução de dimensionalidade II
Principal Component Analysis (PCA)

- Encontrar combinações lineares chamadas de **componentes principais**
  - Capturar variância nos dados
- O primeiro componente tem maior variância
- Os demais são ortogonais ao anterior e apresentam variância decrescente

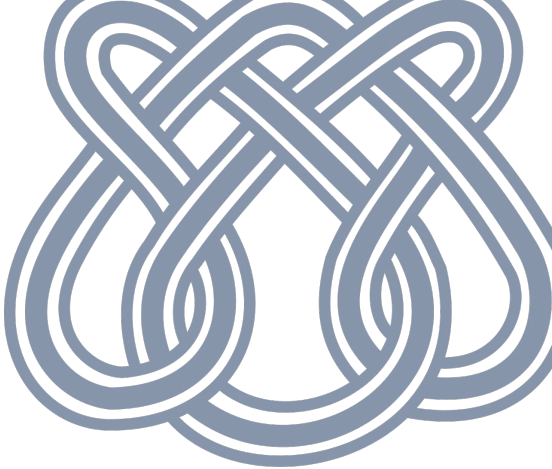
Redução de dimensionalidade III
t-distributed Stochastic Neighbor Embedding (t-SNE)

- Otimização da dissimilaridade entre o espaço projetado e o espaço original
- Sensível a estruturas locais (vizinhanças de dados)
    - Útil para capturar informações de grupos (*clusters*)

How to Use t-SNE Effectively

exemplos/02.ipynb

4 Pipelines



1. Como fazer um Pipeline?
2. Exemplo: Pipeline de texto

Como fazer um Pipeline?

- Código sequencial e estruturado
- Separa as responsabilidades em etapas
- Reuso de código
- Abstração de modelos complexos
    - No Sklearn: `make_pipeline()`

Exemplo: Pipeline de texto I

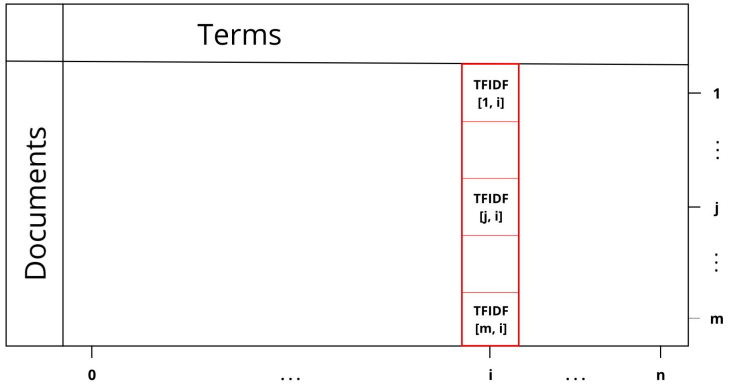
Pré-processamento de texto

1. Transforma para minúsculo
2. Remove símbolos
3. Remove pontuações
4. Remove números
5. Remove espaços em branco
6. Lematização (*WordNet Lemmatizer*)
7. Tokenização
8. Remove palavras vazias (*stopwords*)

Exemplo: Pipeline de texto II

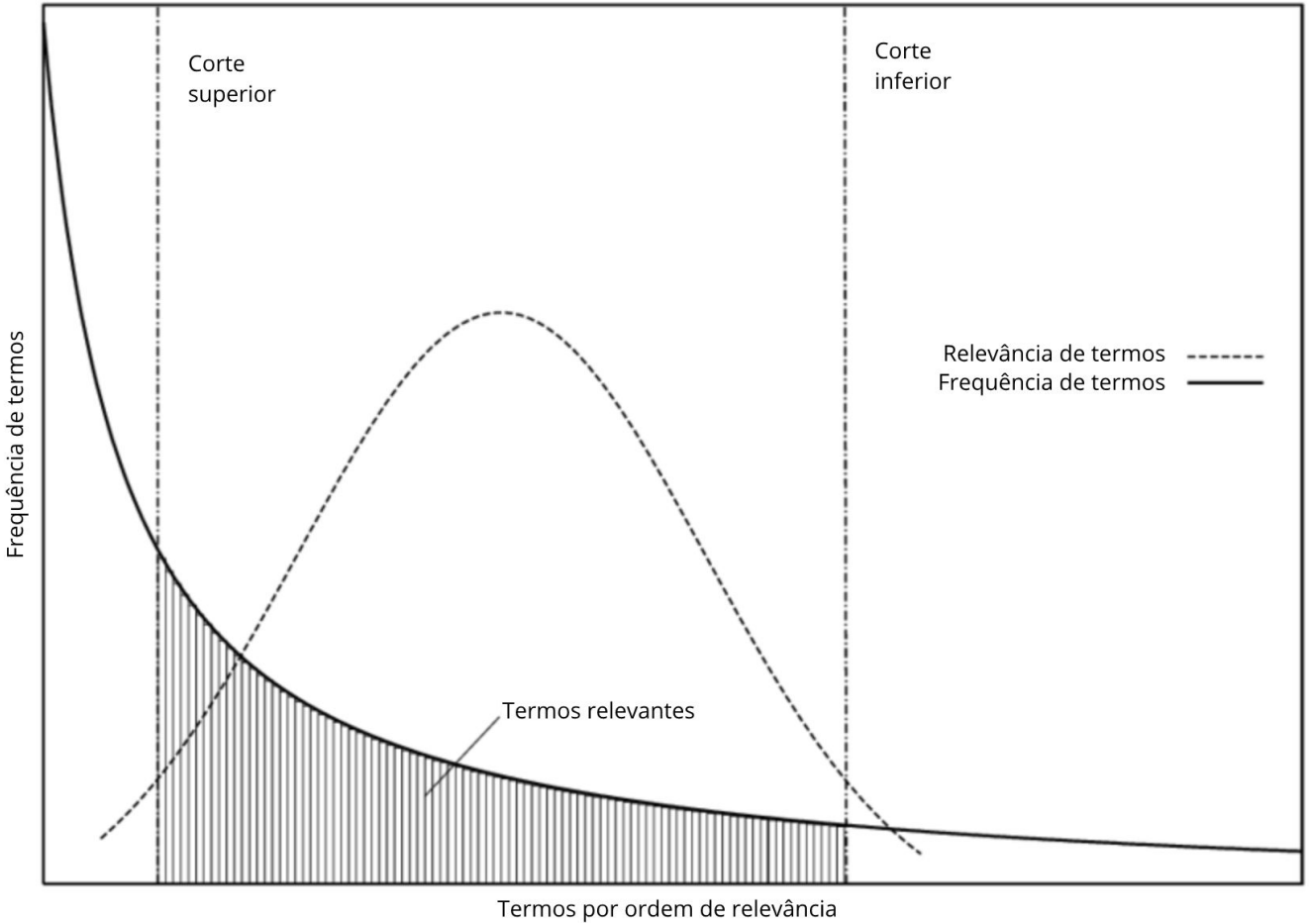
Vetorização dos documentos

- Bag-of-Words
  - Histograma de frequência de palavras no documento
  - n-grams
- Term Frequency - Inverse Document Frequency (TFIDF)
  - Função de peso para o modelo vetorial
- Regra de Zipf e corte de Luhn



https://mungingdata.wordpress.com/2017/11/25/episode-1-using-tf-idf-to-identify-the-signal-from-the-noise/

$$TFIDF_{t,d} = TF_{t,d} \times \log \frac{|D|}{DF_t}$$



Fonte: ResearchGate.net

Exemplo: Pipeline de texto III
Clustering

- KMeans
    - Encontrar os K grupos mais significativos

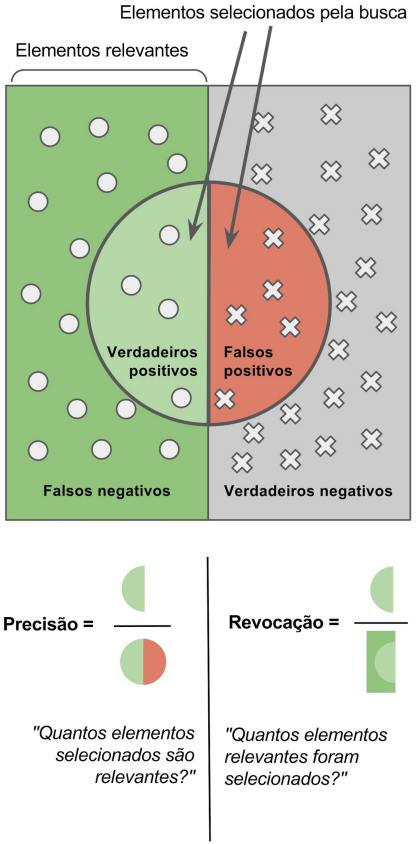
$$\sum_{i=0}^{|X|} \min_{\mu_j \in C} (||x_i - \mu_j||^2)$$

https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

Exemplo: Pipeline de texto IV

Métricas de comparação

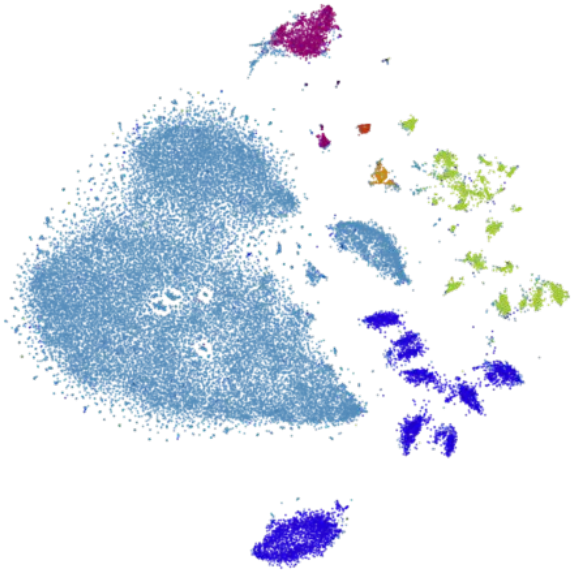
- Silhouette Coefficient[-1, 1]
- Adjusted Rand Index (ARS) [-1, 1]
- Adjusted Mutual Information (AMI) [-1, 1]
- V-Measure (V) [-1, 1]
    - Média harmônica entre homogeneidade e completude
- Fowlkes-Mallows Index (FMI) [0, 1]
    - Média geométrica entre precisão e revocação de pares



Exemplo: Pipeline de texto V

Redução de dimensionalidade

- Norma l^2 + t-SNE



Fonte: OpenTSNE

exemplos/03.ipynb

Projeto etapa 1

Descrição

1. Colete um conjunto de dados de sua preferência
    a. Pode utilizar os conjuntos de dados disponíveis nos repositórios apresentados (Scikit-Learn ou Seaborn)
    b. Dica: a plataforma Kaggle possui uma grande variedade de conjuntos de dados publicamente disponíveis
2. Identifique quais etapas de processamento de dados são necessárias e as execute
3. Descreva os tratamentos executados no seu conjunto de dados e porque eles foram necessários
    a. Demonstre como o tratamento melhorou a análise dos dados
    b. Pode utilizar visualizações ou métricas para as demonstrações

Projeto etapa 1

Organização

Arquivo ZIP contendo:

- Jupyter notebook (Python Versão 3.*) - Código e documentação
- Arquivos externos necessários (.csv, .py, .json, etc...)

Aproveite as funcionalidades do Jupyter para enriquecer e organizar a documentação com fórmulas, tabelas e figuras. Lembre-se que você está entregando um relatório!

Projeto etapa 1

Entrega

- Até 10/10/2021 às 23:55
  - No eDisciplinas