Análise dos dados das campanhas do Kickstarter

Aprendizado de Máquina - 2018.2

Prof. João Paulo Pordeus

Armando Soares
Erick Barros
Fabiano Gadelha
Lucas Mapurunga

Sumário

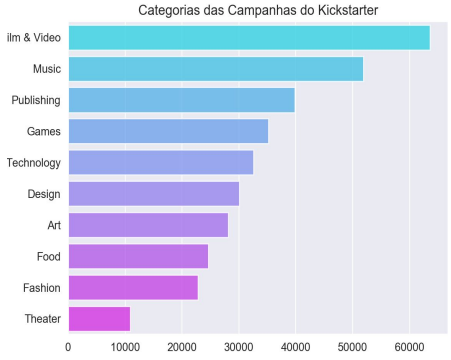
1. Problema: Campanhas do Kickstarter
2. Analisar o dataset das campanhas de 2009 até 2018 do Kickstarter
3. Metodologia
4. Modelo proposto
5. K-Means
6. MLP
7. Conclusões

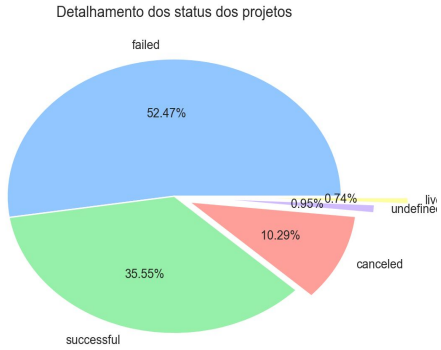
1. Problema: Campanhas do *Kickstarter*

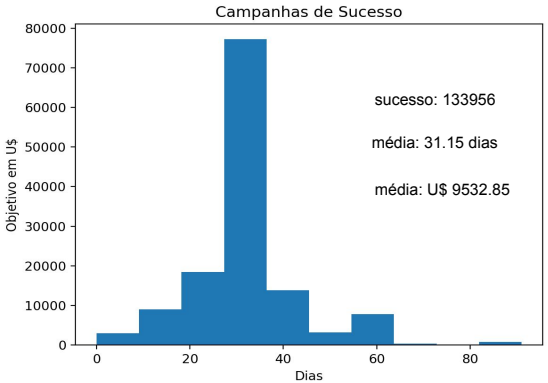
Kickstarter *(Product): a global crowdfunding platform where "creators" can run "campaigns" for people to fund and "back" their project.*

$1.9 billion in pledges from 9.4 million backers to fund 257,000 creative projects

Objetivo da análise: auxiliar os criadores de campanhas na escolha das características de suas propostas através de modelos de aprendizagem automática para então tentar aumentar sua chance de sucesso.







2. Análise do Dataset utilizado

Dados obtidos via Kaggle[1] e Webrobots[2]

Dataset com 372300 amostras sobre as campanhas de maio/2009 a março/2018 do Kickstarter.

15 features disponíveis.

6 features escolhidas: *main_category*, *category*, *backers*, *country*, *usd_pledged_real*, *usd_goal_real*

1 feature criada: *running_days* (A partir de deadline e launched)

1 Label: state

Tabela 1 - Atributos do Conjunto de Dados

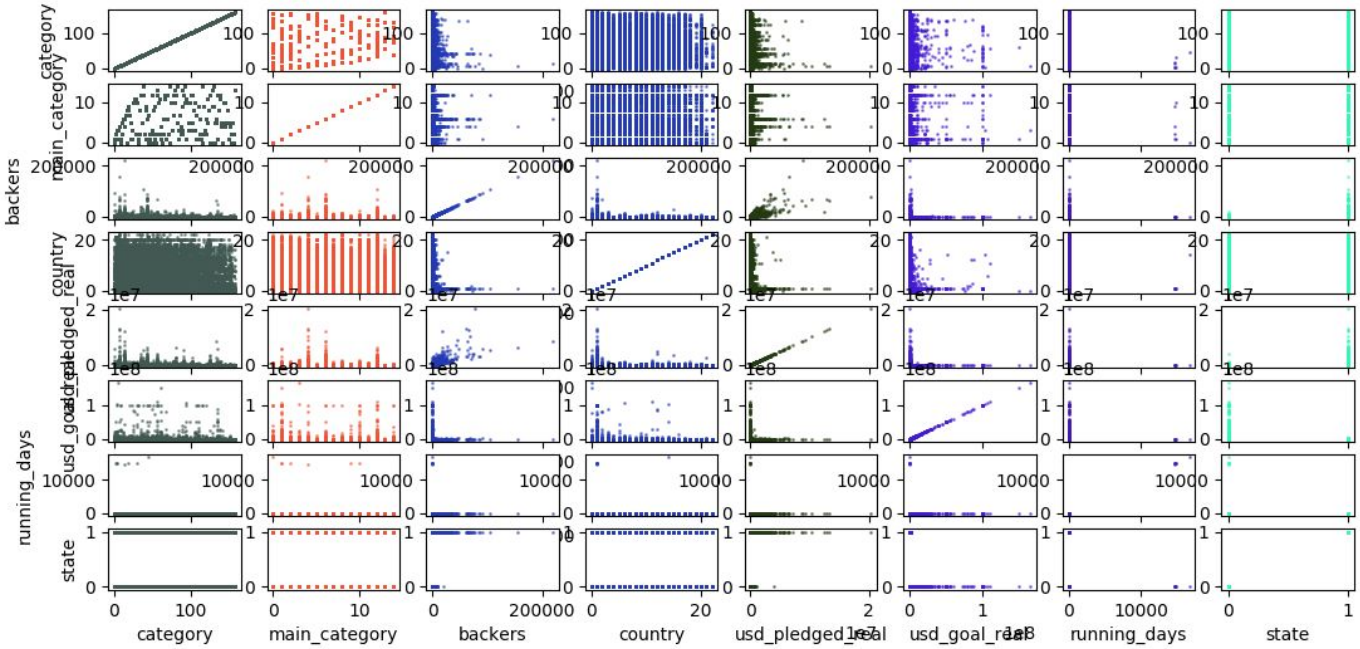
Nome do Atributo	Descrição
ID	Identificador do projeto
name	Nome do projeto
category	Sub-categoria da campanha
main_category	Categoria da campanha
currency	A moeda utilizada (ex: USD)
deadline	Prazo final para o *crowdfunding* do projeto
goal	Montante de dinheiro necessário para o projeto
launched	Dia de lançamento da campanha
pledged	Montante de dinheiro que os apoiadores colaboraram para a campanha
backers	Quantidade de apoiadores do projeto
country	País de origem
usd_pledged	Montante de dinheiro que os apoiadores colaboraram para a campanha em USD
state	Estado final do projeto

2. Análise do Dataset utilizado

Correlação, usando método Spearman, entre as features do dataset:

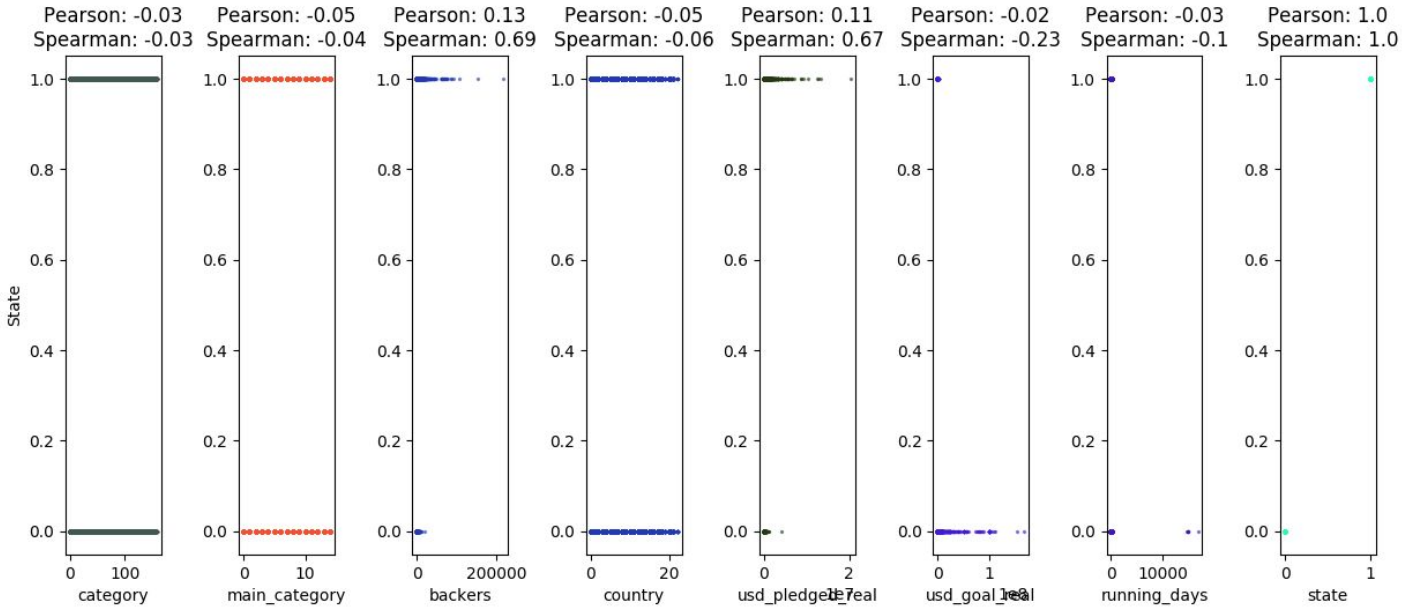
	category	main_category	backers	country	usd_pledged_real	usd_goal_real	running_days	state
category	1.000	0.261	-0.100	0.023	-0.104	-0.081	-0.049	-0.034
main_category	0.261	1.000	0.006	0.052	-0.001	0.040	-0.023	-0.036
backers	-0.100	0.006	1.000	-0.045	0.959	0.106	-0.001	0.691
country	0.023	0.052	-0.045	1.000	-0.040	0.044	0.041	-0.058
usd_pledged_real	-0.104	-0.001	0.959	-0.040	1.000	0.181	0.018	0.672
usd_goal_real	-0.081	0.040	0.106	0.044	0.181	1.000	0.214	-0.228
running_days	-0.049	-0.023	-0.001	0.041	0.018	0.214	1.000	-0.101
state	-0.034	-0.036	0.691	-0.058	0.672	-0.228	-0.101	1.000

2. Análise do Dataset utilizado



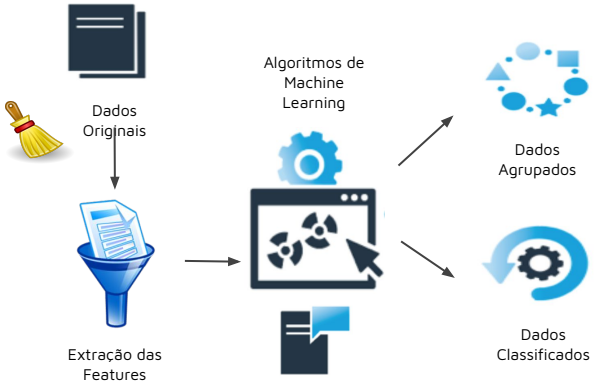
2. Análise do Dataset utilizado

Correlação, usando métodos Pearson e Spearman, entre as features e o status:



3. Metodologia

1. Analisar as informações mais impactantes nas campanhas
2. Fazer a limpeza dos dados do dataset
3. Definir um modelo de clusterização
4. Definir um modelo de classificação
5. Executar os algoritmos de *machine learning*
6. Analisar os dados obtidos



4. Modelo proposto

1. K-means para clusterização
    a. Capacidade de identificar a existência de alguns agrupamentos relacionados às categorias dos projetos.
    b. Uso do *Elbow Method* para escolha do K.

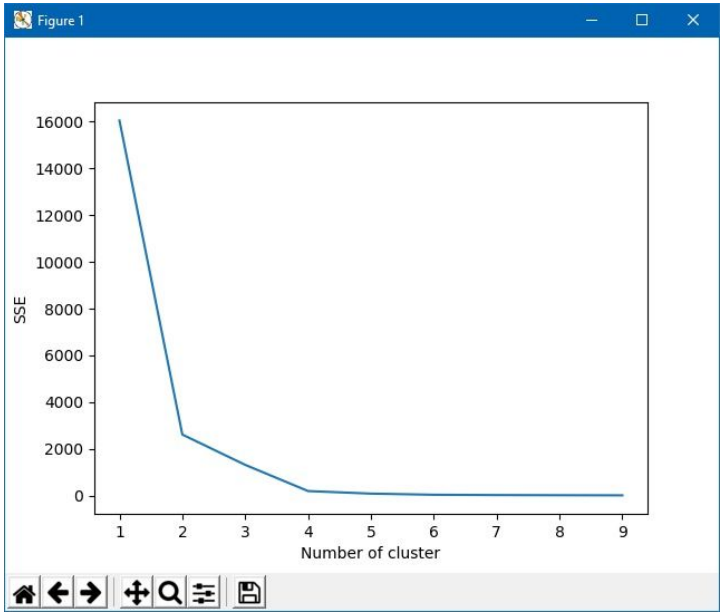
2. Multilayer Perceptron para classificação
    a. Capacidade de lidar com a modelagem de problemas complexos de classificação e dados não linearmente separáveis.
    b. Definição dos números de camadas e critérios de convergência.

3. Foi utilizada a biblioteca Scikit Learn do Python
    a. sklearn.cluster.KMeans
    b. sklearn.neural_network.MLPClassifier

5. K-means

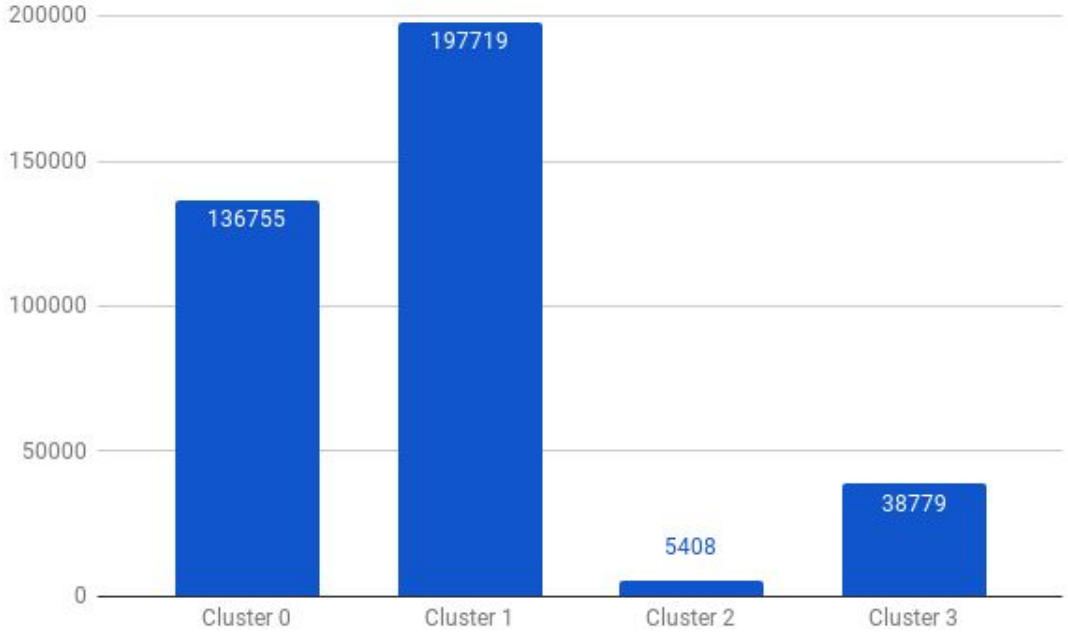
1. Natureza dos dados e tratamento
2. Quantos clusters utilizar?
3. Agrupamento gerado

5. K-means



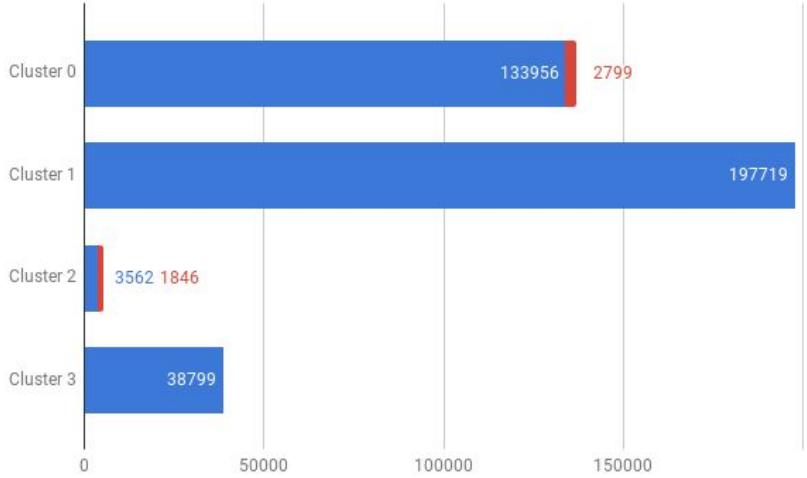
Erro médio por clusters

5. K-means


200000
150000
100000
50000
0
136755
197719
5408
38779
Cluster 0
Cluster 1
Cluster 2
Cluster 3

5. K-means

```
38779
{'failed': 0, 'canceled': 1, 'successful': 2, 'live': 3, 'undefined': 4, 'suspended': 5}
---- States in cluster 0 ----
2    133956
3      2799
Name: state, dtype: int64
---- States in cluster 1 ----
0    197719
Name: state, dtype: int64
---- States in cluster 2 ----
4    3562
5    1846
Name: state, dtype: int64
---- States in cluster 3 ----
1    38779
Name: state, dtype: int64
```



5. K-means

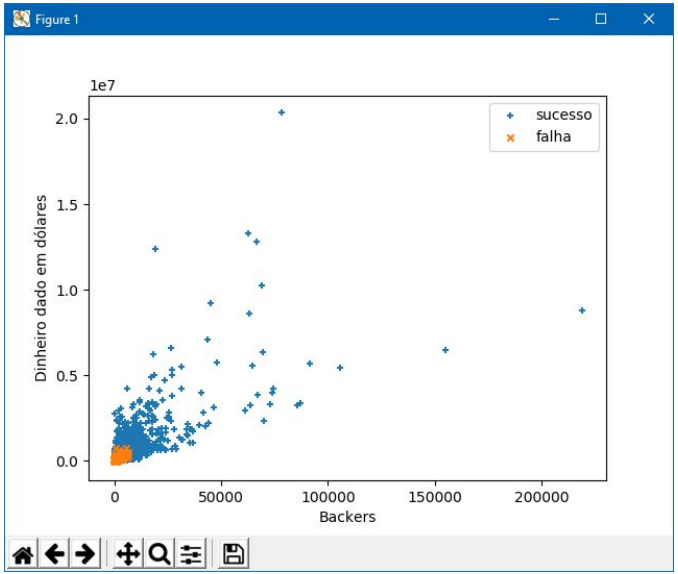


Figure 1
1e7
2.0
1.5
1.0
0.5
0.0
Dinheiro dado em dólares
0
50000
100000
150000
200000
Backers
sucesso
falha

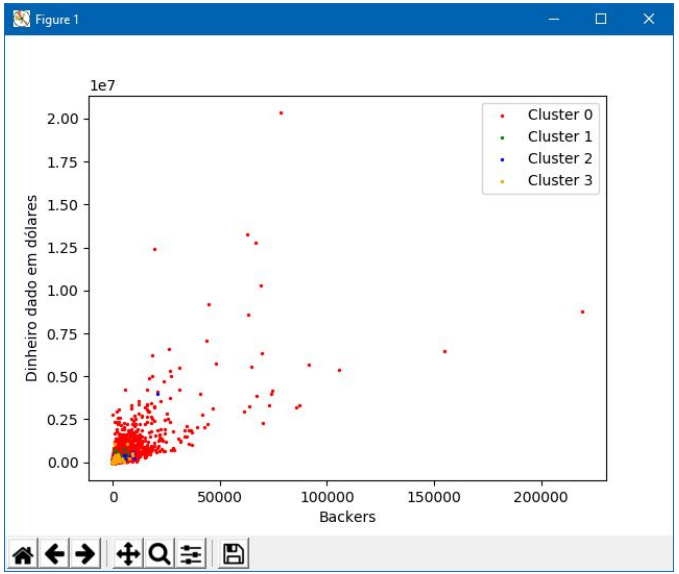


Figure 1
1e7
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Dinheiro dado em dólares
0
50000
100000
150000
200000
Backers
Cluster 0
Cluster 1
Cluster 2
Cluster 3

5. K-means

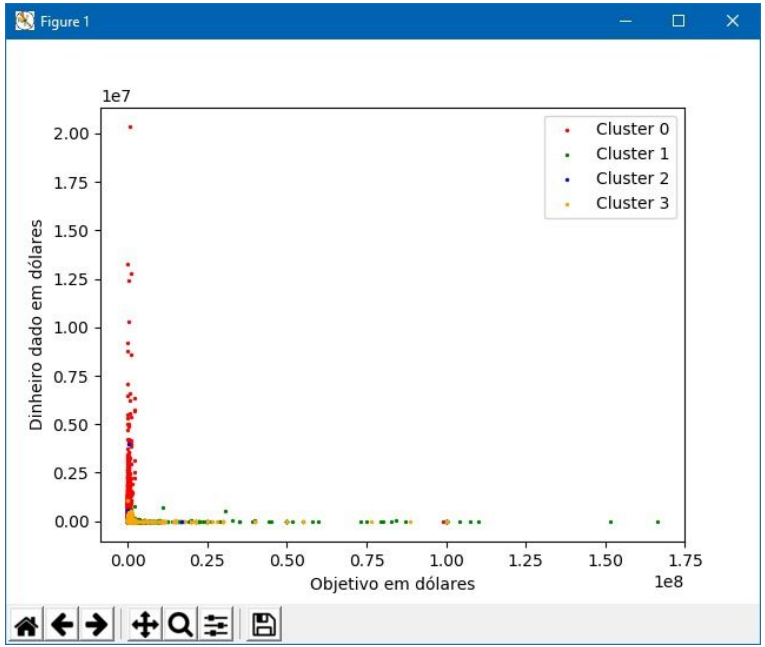


Figure 1
1e7
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Dinheiro dado em dólares
Cluster 0
Cluster 1
Cluster 2
Cluster 3
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
1e8
Objetivo em dólares

6. Multilayer Perceptron

Modelo Genérico:

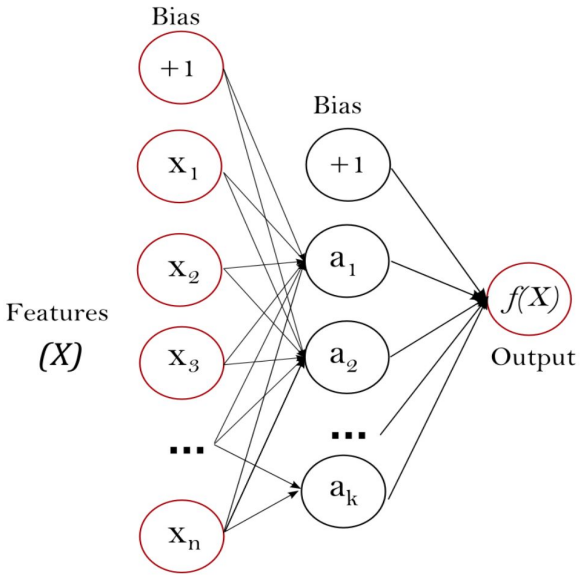


Figure 1 : One hidden layer MLP.

Modelo proposto:

Input: 7 *Features*

x1, x2, x3, x4, x5, x6, x7 (1ª camada)

main_category, *category*, *backers*, *country*, *usd_pledged_real*, *usd_goal_real*, *running_days*

Duas **camadas ocultas** com 5 **neurônios** cada:

a_1^2, a_2^2, a_3^2, a_4^2, a_5^2 (2ª camada)

a_1^3, a_2^3, a_3^3, a_4^3, a_5^3 (3ª camada)

Output

f(X) = a_1^4, a_2^4 (4ª camada)

successful, *others*

6. Multilayer Perceptron

Modelagem dos dados:

- Conjuntos de treino (70%) e teste (30%)
- Normalização dos dados de entrada

Parâmetros utilizados:

- Camadas Ocultas: 2
- Neurônios Ocultos: 10 (5 na 1ª e 5 na 2ª)
- Alpha: 0.01 (taxa de regularização)
- Máximo de Iterações: 100
- Tolerância: 0.0001
- Solver: adam (Versão otimizada do Gradiente Estocástico)

6. Matriz de Confusão

	Positive	Negative
True	70707	1394
False	922	811

6. Métricas

	Acurácia	Precisão	Recall	F1 Score
0	0.98448	0.99	0.99	0.99
1		0.98	0.98	0.98
micro avg		0.98	0.98	0.98
macro avg		0.98	0.98	0.98
weighted avg		0.98	0.98	0.98

7. Conclusões

- Projetos com objetivos mais realistas tem mais chance de sucesso.

- O classificador MLP obteve uma acurácia bastante elevada, provavelmente indicando uma grande separação entre as classes de sucesso e “outros”

Dúvidas?

Referências

1. Repositório do Github com a implementação dos códigos
2. Scikit
3. Dados no Kaggle
4. Dataset do Webrobots

Apêndice. Regressão Logística

Parâmetros utilizados:

- C (parâmetro de regularização): 1.0
- Máximo de Iterações: 100
- Tolerância: 1e-4
- Solver: liblinear

Apêndice. Matriz de Confusão

	Positive	Negative
True	69888	11775
False	1741	28286

Apêndice. Métricas

	Acurácia	Precisão	Recall	F1 Score
0	0.87899	0.86	0.98	0.91
1		0.94	0.71	0.81
micro avg		0.88	0.88	0.88
macro avg		0.90	0.84	0.86
weighted avg		0.89	0.88	0.87