SPRINT 3: MANIA & TITANIC

Lab. Extensão IFES 2020/2 - EAD

Eduarda Simões, Serenna Ferrari e Thais de Souza

SELEÇÃO DE DADOS

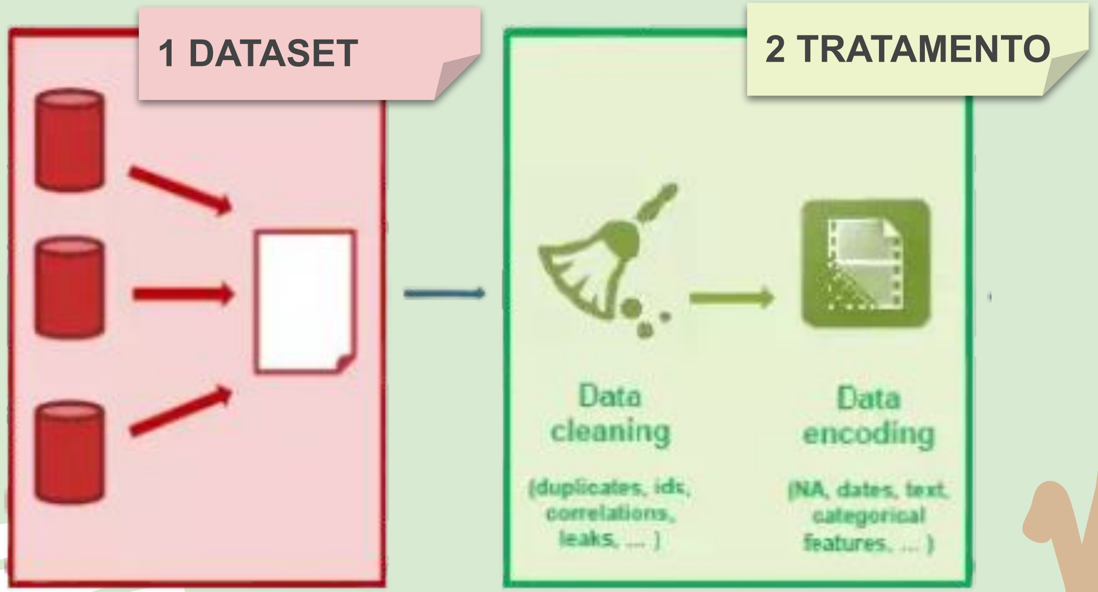
TITANIC

- Informações relacionadas aos passageiros do navio
- Target: survived (0, 1)
- 11 colunas

MANIA

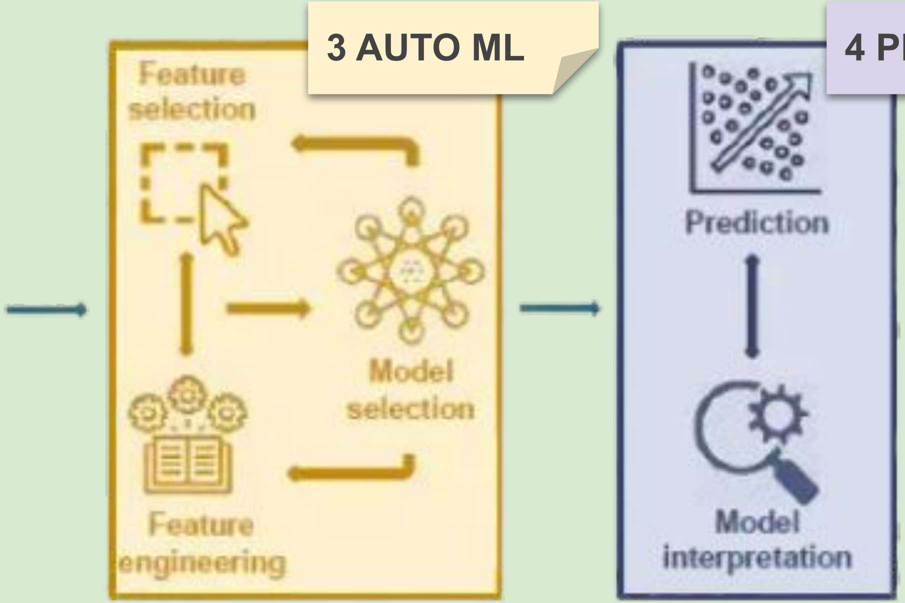
- Informações sobre pacientes de um estudo sobre mania
- Target: dsm_man (5, 1)
- 229 colunas

O PROCESSO



1 DATASET
2 TRATAMENTO
Data cleaning
(duplicates, ids, correlations, leaks, ...)
Data encoding
(NA, dates, text, categorical features, ...)

O PROCESSO


3 AUTO ML
Feature selection
Model selection
Feature engineering
4 PREDIÇÃO
Prediction
Model interpretation

TITANIC

PRÉ PROCESSAMENTO

- Média para preencher dados faltantes;
- Remoção de colunas que não contribuem a análise;
- Label encoder para dados categóricos;
- Encaixotamento para lidar com a alta variação de preços de passagens;
- Exclusão dos outliers;

TITANIC

PRÉ PROCESSAMENTO

NÃO REALIZADO

- Tratamento dos dados duplicados após remoção de colunas
- Balanceamento

MANIA

PRÉ PROCESSAMENTO

- Remoção de colunas nulas ou que não sabíamos as respostas
- Remoção de colunas com correlação maior que 0.85 absoluto
- Preenchimento de dados nulos pelos métodos: ffill e bfill
- Remoção de linhas que não possuíam nenhuma resposta de Mania

MANIA

PRÉ PROCESSAMENTO

- Seleção das 30 melhores características
- Balanceamento por oversampling
- Não houve tratamento de outliers

TITANIC - PRÉ TRATAMENTO
ANÁLISE EXPLORATÓRIA

Dataset statistics

Number of variables	12
Number of observations	891
Missing cells	866
Missing cells (%)	8.1%
Duplicate rows	0
Duplicate rows (%)	0.0%

Variable types

Numeric	5
Categorical	7

Warnings

`name` has a high cardinality: 891 distinct values
`ticket` has a high cardinality: 681 distinct values
`cabin` has a high cardinality: 147 distinct values
`age` has 177 (19.9%) missing values
`cabin` has 687 (77.1%) missing values
`PassengerId` is uniformly distributed
`name` is uniformly distributed
`ticket` is uniformly distributed
`cabin` is uniformly distributed
`PassengerId` has unique values
`name` has unique values
`siblingsSpousesOnboard` has 608 (68.2%) zeros
`parentsChildrenOnboard` has 678 (76.1%) zeros
`fareTicket` has 15 (1.7%) zeros

TITANIC - PÓS TRATAMENTO
ANÁLISE EXPLORATÓRIA

Dataset statistics

Number of variables	7
Number of observations	891
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	280
Duplicate rows (%)	31.4%

Variable types

Categorical
Numeric

Warnings

Dataset has 280 (31.4%) duplicate rows
`siblingsSpousesOnboard` has 608 (68.2%) zeros
`parentsChildrenOnboard` has 678 (76.1%) zeros

MANIA - PRÉ TRATAMENTO
ANÁLISE EXPLORATÓRIA

Dataset statistics

Number of variables	229
Number of observations	5037
Missing cells	613493
Missing cells (%)	53.2%
Duplicate rows	0
Duplicate rows (%)	0.0%

Variable types

Categorical	141
Numeric	79
Unsupported	9

Overview | Warnings 181

Warnings

Missing values warnings	149
Unsupported type warnings	9
Zeros percentage warnings	17
Highly skewed warnings	5
Constant value warnings	1

MANIA - PÓS TRATAMENTO
ANÁLISE EXPLORATÓRIA

Dataset statistics

Number of variables	153
Number of observations	1346
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	31
Duplicate rows (%)	2.3%

Variable types

Categorical	111
Numeric	42

Warnings

Dataset has 31 (2.3%) duplicate rows	Duplicates
`M30G` is highly skewed (γ1 = 25.87298081)	Skewed
`CC32` is highly skewed (γ1 = 25.82124453)	Skewed
`CC49B` is highly skewed (γ1 = 35.74166925)	Skewed
`CC49D` is highly skewed (γ1 = 22.47488804)	Skewed
`M20` has 137 (10.2%) zeros	Zeros
`M21` has 349 (25.9%) zeros	Zeros

COMPARANDO
“ANÁLISE EXPLORATÓRIA

→ Dataset Mania possui aproximadamente 4,5x mais registros que o de Titanic;

→ Mais problemas identificados no Dataset Mania;

→ Pré processamento distinto em cada dataset;

TITANIC
MACHINE LEARNING

```
solvers = ['liblinear', 'newton-cg', 'lbfgs', 'saga']
c_values = [1.99, 1.9, 1.0, 0.1, 0.01]
```

```
grid = dict(solver = solvers, C = c_values)
cv = RepeatedStratifiedKFold(n_splits = 10,
n_repeats = 3, random_state = 1)
grid_search = GridSearchCV(estimator = model,
param_grid = grid, n_jobs = -1, cv = cv,
scoring = 'accuracy', error_score = 0)
grid_result = grid_search.fit(x,y)
```

```
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
for mean, stdev, param in zip(means, stds, params):
  print("%f (%f) with: %r" % (mean, stdev, param))
```

```
Best: 0.822703 using {'C': 1.99, 'solver': 'liblinear'}
0.822703 (0.053345) with: {'C': 1.99, 'solver': 'liblinear'}
0.819747 (0.052302) with: {'C': 1.99, 'solver': 'newton-cg'}
0.819747 (0.052302) with: {'C': 1.99, 'solver': 'lbfgs'}
0.799056 (0.042214) with: {'C': 1.99, 'solver': 'saga'}
0.822703 (0.053345) with: {'C': 1.9, 'solver': 'liblinear'}
0.819747 (0.052302) with: {'C': 1.9, 'solver': 'newton-cg'}
0.819747 (0.052302) with: {'C': 1.9, 'solver': 'lbfgs'}
0.799056 (0.041525) with: {'C': 1.9, 'solver': 'saga'}
0.822703 (0.053345) with: {'C': 1.0, 'solver': 'liblinear'}
0.819747 (0.052302) with: {'C': 1.0, 'solver': 'newton-cg'}
0.819747 (0.052302) with: {'C': 1.0, 'solver': 'lbfgs'}
0.799056 (0.041525) with: {'C': 1.0, 'solver': 'saga'}
0.813835 (0.053079) with: {'C': 0.1, 'solver': 'liblinear'}
0.821232 (0.049097) with: {'C': 0.1, 'solver': 'newton-cg'}
0.821232 (0.049097) with: {'C': 0.1, 'solver': 'lbfgs'}
0.798076 (0.040416) with: {'C': 0.1, 'solver': 'saga'}
```

TITANIC
MACHINE LEARNING

```
rf = RandomForestClassifier(random_state = 0)
print('Parâmetros em uso: \n')
print(rf.get_params())
```

```
param_grid = {
    'bootstrap' : [True, False],
    'max_depth' : [10, 15, 20],
    'n_estimators' : [200, 300, 400]
}
```

```
cv = RepeatedStratifiedKFold(n_splits = 5, n_repeats = 3, random_state = 1)
grid_search = GridSearchCV(estimator = rf, param_grid = param_grid, n_jobs = -1,
                           cv = cv, scoring = 'accuracy', error_score = 0)
grid_result = grid_search.fit(x_train,y_train)
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
for mean, stdev, param in zip(means, stds, params):
  print("%f (%f) with: %r" % (mean, stdev, param))
```

```
Best: 0.799622 using {'bootstrap': True, 'max_depth': 15, 'n_estimators': 400}
```

```
automl = autosklearn.classification.AutoSklearnClassifier(
    time_left_for_this_task = 120,
    per_run_time_limit = 30
    #include_estimators = ["decision_tree", "random_forest", "extra_trees"]
    , tmp_folder = '/tmp/autosklearn_classification_example_tmp'
    , output_folder = '/tmp/autosklearn_classification_example_out',
)
automl.fit(X_train, Y_train)
```

MANIA

MACHINE LEARNING

NÃO REALIZADO

- Uso de hiperparâmetros
- Aplicação de cross validate

MANIA
MACHINE LEARNING

```
automl = autosklearn.classification.AutoSklearnClassifier(
    time_left_for_this_task = 120,
    per_run_time_limit = 30,
    tmp_folder='/tmp/autosklearn_classification_example_tmp',
    output_folder='/tmp/autosklearn_classification_example_out'
)
automl.fit(X_train, Y_train)
```

```
'/tmp/autosklearn_classification_example_out'
```

```
predictions = automl.predict(X_test)


#CRIANDO A MATRIZ DE CONFUSÃO E REPORT
matrix    = confusion_matrix(Y_test, predictions)

print('==== Conf. Matrix ====')
print(matrix)

report = classification_report(Y_test, predictions)
print('\n==== Report ====')
print(report)
```

```
print('Accuracy score:  ', sklearn.metrics.accuracy_score(Y_test, predictions))
print('Accuracy AUC:    ', sklearn.metrics.roc_auc_score(Y_test, predictions))
print('Precision score: ', sklearn.metrics.precision_score(Y_test, predictions))
print('Recall score:    ', sklearn.metrics.recall_score(Y_test, predictions))
print('F1 score:        ', sklearn.metrics.f1_score(Y_test, predictions))
```

TITANIC
RESULTADOS

MATRIZ DE CONFUSÃO

102	5
24	39

REPORT

	PRECISION	RECALL	F1-SCORE
0	0.81	0.95	0.88
1	0.89	0.62	0.73

ACURÁCIAS

SCORE	0.82
AUC	0.72

MANIA

RESULTADOS

MATRIZ DE CONFUSÃO

198	0
0	206

REPORT

	PRECISION	RECALL	F1-SCORE
1	1.0	1.0	1.0
5	1.0	1.0	1.0

ACURÁCIAS

SCORE	1.0
AUC	1.0

COMPARANDO RESULTADOS

TITANIC
REPORT

	PRECISION	RECALL	F1-SCORE
0	0.81	0.95	0.88
1	0.89	0.62	0.73

MANIA
REPORT

	PRECISION	RECALL	F1-SCORE
1	1.0	1.0	1.0
5	1.0	1.0	1.0