OPERADORA MEGALINE

O OBJETIVO DESTE PROJETO É DESENVOLVER UM MODELO QUE POSSA ANALISAR O COMPORTAMENTO DO CLIENTE E RECOMENDAR UM DOS PLANOS MAIS RECENTES DA EMPRESA: SURF OU ULTIMATE. SERÁ DESENVOLVIDO UM MODELO COM A MAIOR ACURÁCIA POSSÍVEL

DESCRIÇÃO DO PROJETO

A operadora de celular Megaline está insatisfeita com o fato de muitos de seus clientes estarem usando planos antigos. Eles querem desenvolver um modelo que possa analisar o comportamento do cliente e recomendar um dos planos mais recentes da Megaline: Smart ou Ultra.
Você tem acesso a dados de comportamento dos assinantes que já mudaram para os novos planos (do projeto do curso de Análise de Dados Estatísticos). Para esta tarefa de classificação, você precisa desenvolver um modelo que escolherá o plano certo. Como você já executou a etapa de pré-processamento de dados, pode ir direto para a criação do modelo.
Desenvolva um modelo com a maior acurácia possível. Neste projeto, o limite para acurácia é 0,75. Verifique a acurácia usando o conjunto de dados de teste.

IMPORTAÇÃO DE BIBLIOTECAS

In [16]:

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
```

CARREGAMENTO DOS DADOS

In [17]:

```
df= pd.read_csv('/datasets/users_behavior_upd.csv')
df.head()
```

Out[17]:

	calls	minutes	messages	mb_used	is_ultimate
0	40.0	311.90	83.0	19915.42	0
1	85.0	516.75	56.0	22696.96	0
2	77.0	467.66	86.0	21060.45	0
3	106.0	745.53	81.0	8437.39	1
4	66.0	418.74	1.0	14502.75	0

OS DADOS JÁ FORAM TRATADOS PREVIAMENTE, PORTANTO NÃO SERÁ NECESSÁRIO VERIFICAR VALORES AUSENTES,DUPLICADOS OU ERROS

DIVISÃO DOS DADOS DE ORIGEM

In [18]:

```
df_train, df_valid_and_test = train_test_split(df, test_size=0.4, random_state=42)
df_valid, df_test = train_test_split(df_valid_and_test, test_size=0.5, random_state=42)

features_train = df_train.drop(['is_ultimate'], axis=1)
target_train = df_train['is_ultimate']

features_valid = df_valid.drop(['is_ultimate'], axis=1)
target_valid = df_valid['is_ultimate']

features_test = df_test.drop(['is_ultimate'], axis=1)
target_test = df_test['is_ultimate']

print(features_train.shape)
print(target_train.shape)
print(features_valid.shape)
print(target_valid.shape)
print(features_test.shape)


print(target_test.shape)
```

```
(1928, 4)
(1928,)
(643, 4)
(643,)
(643, 4)
(643,)
```

AJUSTE DA QUALIDADE DOS MODELOS

ÁRVORE DE DECISÃO

In [19]:

```
decision_tree_cols = ['depth', 'acc_train', 'acc_valid']
decision_tree_list = []

for depth in range(1, 11):
    model_dt = DecisionTreeClassifier(max_depth=depth, random_state=42)
    model_dt.fit(features_train, target_train)
    decision_tree_list.append([depth,
                               model_dt.score(features_train, target_train),
                               model_dt.score(features_valid, target_valid)
                               ])

decision_tree = pd.DataFrame(decision_tree_list, columns=decision_tree_cols)
decision_tree
```

Out[19]:

	depth	acc_train	acc_valid
0	1	0.747925	0.730949
1	2	0.781639	0.782271
2	3	0.797199	0.791602
3	4	0.808610	0.780715
4	5	0.815353	0.772939
5	6	0.822614	0.776050
6	7	0.838693	0.780715
7	8	0.852178	0.796267
8	9	0.863589	0.780715
9	10	0.878631	0.794712

PODE-SE OBSERVAR QUE, EXISTE UMA RELAÇÃO DIRETA E POSITIVA ENTRE A PROFUNDIDADE DA ÁRVORE E A PRECISÃO DO CONJUNTO DE TREINAMENTO, QUANTO MAIOR A PROFUNDIDADE, MAIOR É A PRECISÃO NO CONJUNTO. TODAVIA, A PRECISÃO NO CONJUNTO DE VALIDAÇÃO MELHORA ATÉ O DEPTH 3, E EM SEGUIDA PERDE A QUALIDADE. NO DEPTH 3, É O ÚLTIMO MODELO EM QUE A QUALIDADE DA PRECISÃO EM AMBOS SÃO SEMELHANTES. POR ISSO,A CONCLUSÃO É QUE OS MODELOS MAIS PROFUNDOS SÃO AFETADOS.

FLORESTA ALEATÓRIA

In [20]:

```
random_forest_cols = ['estimator', 'acc_train', 'acc_valid']
random_forest_list = []

for estimator in range(10, 101, 10):
    model_rf = RandomForestClassifier(n_estimators=estimator, random_state=42)
    model_rf.fit(features_train, target_train)
    random_forest_list.append([estimator,
                               model_rf.score(features_train, target_train),
                               model_rf.score(features_valid, target_valid)
                               ])

random_forest = pd.DataFrame(random_forest_list, columns=random_forest_cols)
random_forest
```

Out[20]:

	estimator	acc_train	acc_valid
0	10	0.980290	0.786936
1	20	0.991701	0.791602
2	30	0.994813	0.793157
3	40	0.997925	0.796267
4	50	0.997925	0.796267
5	60	0.999481	0.796267
6	70	0.998963	0.797823
7	80	1.000000	0.800933
8	90	1.000000	0.802488
9	100	1.000000	0.802488

CONFORME AUMENTA O ESTIMATOR, TAMBÉM AUMENTA-SE A PRECISÃO NO CONJUNTO DE TREINAMENTO. A PRECISÃO NO CONJUNTO DE TREINAMENTO ATINGE O LIMITE QUANDO O ESTIMATOR ESTÁ EM 80. NO CONJUNTO DE VALIDAÇÃO, A PRECISÃO PARA DE CRESCER QUANDO ATINGE 90 ESTIMATOR

REGRESSÃO LOGÍSTICA

In [21]:

```
solver_list = ['lbfgs', 'liblinear', 'newton-cg','sag', 'saga']
logistic_regression_cols = ['solver', 'acc_train', 'acc_valid']
logistic_regression_list = []

for solver_item in solver_list:
    model_lr = LogisticRegression(random_state=42, solver=solver_item)
    model_lr.fit(features_train, target_train)
    logistic_regression_list.append([solver_item,
                                model_lr.score(features_train, target_train),
                                model_lr.score(features_valid, target_valid)
                                ])

logistic_regression = pd.DataFrame(logistic_regression_list, columns=logistic_regression_
logistic_regression
```

```
5: LineSearchWarning: The line search algorithm did not converge
  warn('The line search algorithm did not converge', LineSearchWarning)
/opt/conda/lib/python3.9/site-packages/scipy/optimize/_linesearch.py:45
6: LineSearchWarning: The line search algorithm did not converge
  warn('The line search algorithm did not converge', LineSearchWarning)
/opt/conda/lib/python3.9/site-packages/scipy/optimize/_linesearch.py:30
5: LineSearchWarning: The line search algorithm did not converge
  warn('The line search algorithm did not converge', LineSearchWarning)
/opt/conda/lib/python3.9/site-packages/scipy/optimize/_linesearch.py:45
6: LineSearchWarning: The line search algorithm did not converge
  warn('The line search algorithm did not converge', LineSearchWarning)
/opt/conda/lib/python3.9/site-packages/scipy/optimize/_linesearch.py:30
5: LineSearchWarning: The line search algorithm did not converge
  warn('The line search algorithm did not converge', LineSearchWarning)
/opt/conda/lib/python3.9/site-packages/sklearn/linear_model/_sag.py:328:
ConvergenceWarning: The max_iter was reached which means the coef_ did n
ot converge
  warnings.warn("The max_iter was reached which means "
/opt/conda/lib/python3.9/site-packages/sklearn/linear_model/_sag.py:328:
ConvergenceWarning: The max_iter was reached which means the coef_ did n
```

A PRECISÃO É MENOR, PORÉM MAIS RÁPIDA. OS QUE FORCECEM A MELHOR PRECISÃO TANTO NO CONJUNTO DE TREINAMENTO, QUANTO NO CONJUNTO DE VALIDAÇÃO É O LBFGS E NEWTON-CG. E TODOS OS FICAM ABAIXO DE 75%.

CONCLUSÃO DOS MODELOS

FLORESTA ALEATÓRIA: ELA É SUPERAJUSTADA,PORÉM TEM MAIOR PRECISÃO POR UTILIZAR UM CONJUNTO DE ÁRVORES AO EM VEZ DE APENAS UMA.

ÁRVORE DE DECISÃO: A ÁRVORE ESTÁ SUBAJUSTADA QUANDO A PROFUNDIDADE ESTÁ ABAIXO DE 2, E ACIMA DE 3 ESTÁ SOBREAJUSTADA.

REGRESSÃO LOGÍSTICA: ELA NÃO É SUPERAJUSTADA, PORÉM TEM A MENOR QUALIDADE DE PREVISÃO.

MODELO DE TESTE

UTILIZAREMOS O MELHOR MODELO DE TREINO DE OBTIVEMOS: O MODELO DE **FLORESTA ALEATÓRIA** COM ESTIMATOR DE 90, QUANDO ELE ATINGE O AUGE.

In [22]:

```
model = RandomForestClassifier(n_estimators=90, random_state=42)
model.fit(features_train, target_train)

print('Acurácia do melhor modelo do conjunto de treino é:', model.score(features_train, t
print('Acurácia do melhor modelo do conjunto de validação é:', model.score(features_valid
print('Acurácia do melhor modelo do conjunto de teste é:', model.score(features_test, tar
```

```
Acurácia do melhor modelo do conjunto de treino é: 1.0
Acurácia do melhor modelo do conjunto de validação é: 0.80248833592535
Acurácia do melhor modelo do conjunto de teste é: 0.8133748055987559
```

CONCLUSÃO

NESTE PROJETO, FORAM VERIFICADOS OS MODELOS: ÁRVORE DE DECISÃO, FLORESTA ALEATÓRIA E REGRESSÃO LOGÍSTICA. DESTAS, UTILIZAMOS A FLORESTA ALEATÓRIA PARA O TESTE, POIS APRESENTOU O MELHOR PRECISÃO.

O MODELO TESTADO APRESENTOU A PRECISÃO DE 81,33% DE ACURÁCIA NO CONJUNTO DE TESTE.