Universidade Presbiteriana Mackenzie

Regressão Logistica - LAB

```
import pandas                  as pd
import numpy                   as np
import matplotlib.pyplot       as plt
import seaborn                 as sns
import statsmodels.formula.api as sm
import warnings
warnings.filterwarnings("ignore")

from sklearn.model_selection import train_test_split #para facilitar a separação dos dados..
from sklearn.linear_model import LogisticRegression  #para configurar o modelo..
from sklearn import metrics                          #para obter métricas de análise..
```

Relembrando..

```
#separando variáveis independentes das dependentes
x = df.drop('Y',axis=1)
y = df['Y']

#montando todos os df necessários
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=1)

#criando o modelo
logreg = LogisticRegression()

#treinando o modelo
logreg.fit(X_train,y_train)

#utilizando o modelo
y_pred=logreg.predict(X_test)

#visualizando os resultados
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix

#verificando com qual probabilidade os dados foram classificados
y_pred_prob = logreg.predict_proba(df.drop('Y',axis=1))
```

CASO: Consumo de combustível em veículos

Resumo: Informações sobre o consumo de combustível em veículos, medido em milhas por galão (mpg)

Para descrição completa dos dados acesse https://archive.ics.uci.edu/ml/datasets/auto+mpg.

```
mpg = sns.load_dataset('mpg')

mpg.head()
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	3449	10.5	70	usa	ford torino

```
#Qual o tamanho do df?
mpg.shape

#existem valores que devem ser excluídos?
mpg.isna().sum()

#se existirem, exclua..
mpg = mpg.dropna()
```

```
mpg.head()
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	3449	10.5	70	usa	ford torino

```
mpg[mpg.mpg > 25]['mpg'].count()/len(mpg)
```

```
0.3979591836734694
```

```
mpg.origin.value_counts()
```

```
usa       245
japan      79
```

```
europe     68
Name: origin, dtype: int64
```

Vamos alterar a coluna `origin` de modo que ela represente se um veículo foi ou não produzido nos EUA. Dica: utilize:

```
# df.coluna.replace('antigo','novo')
```

```
#trocando os dados
mpg.origin = mpg.origin.replace('europe','non-usa')
mpg.origin = mpg.origin.replace('japan','non-usa')
```

```
#quantos veículos da base foram produzidos nos EUA e quantos não?
mpg.origin.value_counts()

#qual a porcentagem de veículos dos EUA?
#mpg.origin.value_counts()[0] / len(mpg)
```

```
usa        245
non-usa    147
Name: origin, dtype: int64
```

O consumo em mpg e o número de cilindros, conseguem classificar se um veículo foi produzido nos EUA ou não?

```
x = mpg[['mpg','cylinders']]
y = mpg['origin']
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=1)
```

```
#criando o modelo
logreg = LogisticRegression()

#treinando o modelo
logreg.fit(X_train,y_train)

#utilizando o modelo
y_pred=logreg.predict(X_test)

#visualizando os resultados
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

```
array([[36,  4],
       [15, 63]])
```

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

```
0.8389830508474576
```

Construa um modelo de regressão logistica para prever se um veículo foi produzido nos EUA ou não, considerando todas as variáveis possíveis.

```
x = mpg.drop(['origin','name'], axis=1)
y = mpg['origin']
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=1)
```

```
#criando o modelo
logreg = LogisticRegression()

#treinando o modelo
logreg.fit(X_train,y_train)

#utilizando o modelo
y_pred=logreg.predict(X_test)

#visualizando os resultados
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

```
array([[37,  3],
       [ 4, 74]])
```

```
accuracy_score(y_test,y_pred)
```

```
0.940677966101695
```