07

Faça o que eu fiz na aula

Agora vamos copiar a variável resposta para um variável "y"

```
y = df_quant.transactionRevenue.copy()
```

Também vamos criar um variável X, que vai ter todas as variáveis do df_quant, com exceção da variável resposta.

```
X = df_quant.drop('transactionRevenue',axis=1)
```

Utilizaremos o método train_test_split do sklearn para dividir a base em treino e teste.

```
from sklearn.model_selection import train_test_split
```

Vamos dividir a base em treino e teste, deixando 30% para teste e 70% para treino.

```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, random_state=42)
```

Ao verificar o X_train, a saída deve ser igual a:

```
X_train.head()
```

	fullVisitorId	bounces	hits	newVisits	pageviews
9066	9160012321913348805	0.0	16	1.0	12
2603	3214687528122048939	1.0	1	1.0	1
8832	8936592721963873590	0.0	13	1.0	13
1579	2277422319005476191	1.0	1	1.0	1
9887	989458850703391704	1.0	1	1.0	1

Ao verificar o y_train, a saída deve ser igual a:

```
y_train.head()
```

```
9066    144.97
2603      0.00
8832      6.99
1579      0.00
9887      0.00
```

Agora vamos tentar prever o valor gasto por usuário utilizando uma regressão linear.

```
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
```

```
reg.fit(X_train, y_train)
reg_predict = reg.predict(X_test)
```

Ao verificar o reg_predict, a saída deve ser igual a:

```
reg_predict
```

```
array([1.55166047, 1.55520911, 1.68772647, ..., 1.61310964, 1.52678998,
       1.58119137])
```

Podemos criar um dataframe para avaliar para cada linha da base teste o valor real, previsão e erro.

```
resultados = pd.DataFrame()
resultados['revenue'] = y_test
resultados['predict'] = reg_predict
resultados['erro'] = reg_predict - y_test
```

Ao verificar o resultados, a saída deve ser igual a:

```
resultados.head()
```

	revenue	predict	erro
4122	0.0	1.551660	1.551660
4065	0.0	1.555209	1.555209
1731	0.0	1.687726	1.687726
4740	0.0	1.515492	1.515492
6391	0.0	1.422655	1.422655

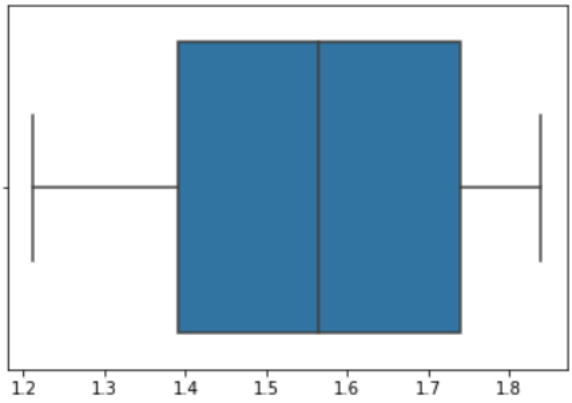
Para avaliar a performance do nosso modelo vamos utilizar a raiz do erro médio quadrático.

```
from sklearn.metrics import mean_squared_error
import numpy as np
print(np.sqrt(mean_squared_error(y_test, reg_predict)))
```

28,41207366

Agora, vamos montar alguns gráficos para analisar as predições.

```
import seaborn as sns
sns.boxplot(reg_predict)
```




```
sns.boxplot(y_test)
```