Trabalho Prático: Introdução à Ciência dos Dados

Entrega 4 - Análise Preditiva

Integrantes: Aryel Penido - 3500 Claudio Barbosa - 3492 Isabela Ramos - 3474

Tema: Análise da população em situação de rua em BH

Dados: https://dados.pbh.gov.br/dataset/populacao-de-rua

Perguntas a serem respondidas:

↳ *1 cell hidden*

Importantando as bibliotecas:

[ ] ↳ *3 cells hidden*

Agrupando todos os dados em um único Dataframe

```
df=pd.concat([data9_20,data10_20, data11_20, data12_20, data1_21, data2_21, data3
```

Agrupando os dados por ano

[ ] ↳ *1 cell hidden*

Tratamento dos dados

[ ] ↳ *14 cells hidden*

Tratamento dos dados - Previsão

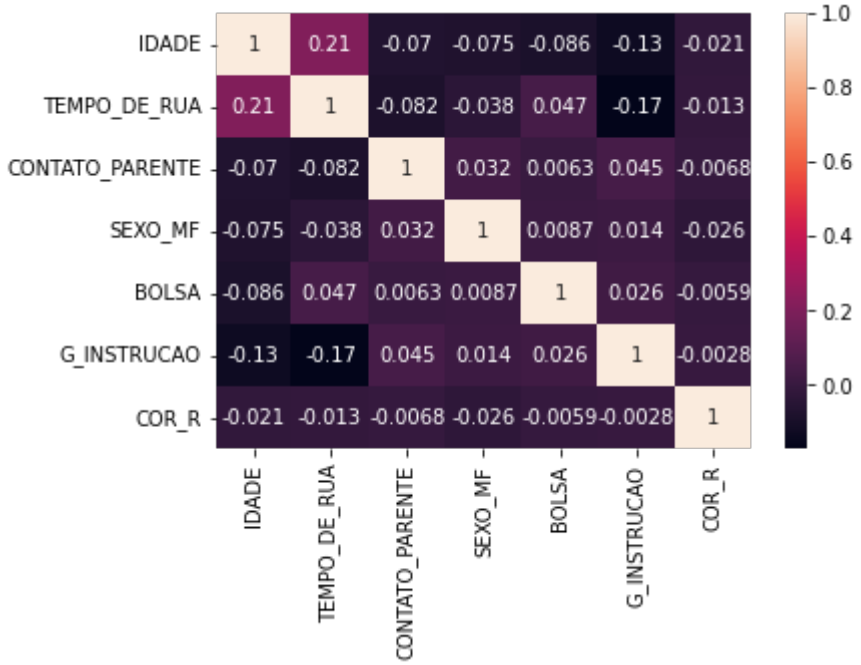
[ ] ↳ *21 cells hidden*

Análise Preditiva

```
       'IDADE', 'SEXO', 'BOLSA_FAMILIA', 'POP_RUA', 'GRAU_INSTRUCAO',
       'COR_RACA', 'Faixa da renda familiar per capita', 'CRAS', 'REGIONAL'
       'FAIXA_DESATUALICACAO_CADASTRAL', 'MES_ANO_REFERENCIA', 'TEMPO_DE_RU
       'CONTATO_PARENTE', 'SEXO_MF', 'BOLSA', 'G_INSTRUCAO', 'COR_R'],
      dtype='object')
```

```
sns.heatmap(df.corr(),annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5717f04c90>
```

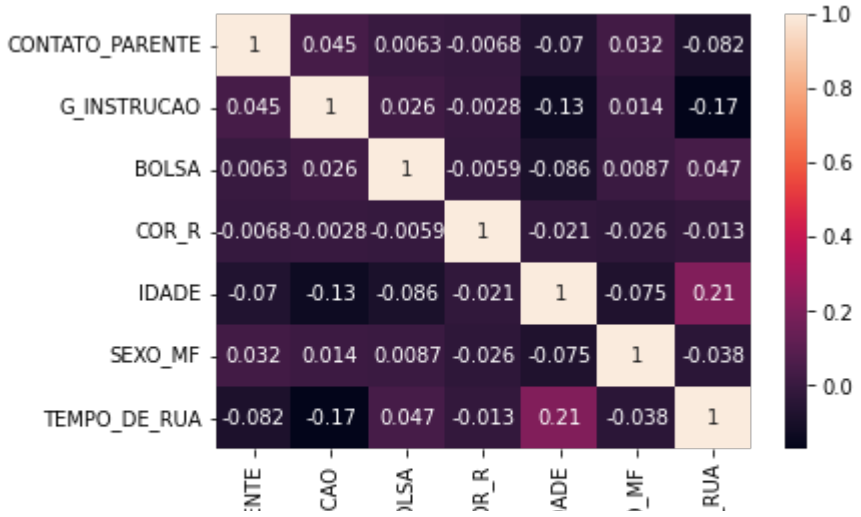



```
df_pred = df[{'IDADE','G_INSTRUCAO','BOLSA','SEXO_MF','CONTATO_PARENTE','COR_R',
df_pred.keys()
```

```
Index(['CONTATO_PARENTE', 'G_INSTRUCAO', 'BOLSA', 'COR_R', 'IDADE', 'SEXO_M
       'TEMPO_DE_RUA'],
      dtype='object')
```

```
sns.heatmap(df_pred.corr(),annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f570fda9590>
```



```
corr = df_pred.corr('spearman')
corr = corr[['IDADE']].sort_values(by = ['IDADE'],ascending = False)
corr
```

	IDADE
IDADE	1.000000
TEMPO_DE_RUA	0.217237
COR_R	-0.019675
CONTATO_PARENTE	-0.057731
SEXO_MF	-0.059367
BOLSA	-0.065517
G_INSTRUCAO	-0.174055

```
corr = df_pred.corr('spearman')
corr = corr[['SEXO_MF']].sort_values(by = ['SEXO_MF'],ascending = False)
corr
```

	SEXO_MF
SEXO_MF	1.000000
CONTATO_PARENTE	0.029739
BOLSA	0.008719
G_INSTRUCAO	0.007300
COR_R	-0.025996
TEMPO_DE_RUA	-0.037482
IDADE	-0.059367

```
corr = df_pred.corr('spearman')
corr = corr[['TEMPO_DE_RUA']].sort_values(by = ['TEMPO_DE_RUA'],ascending = Fals
corr
```

	COR_R
COR_R	1.000000
G_INSTRUCAO	0.001240
CONTATO_PARENTE	-0.005328
BOLSA	-0.006119
TEMPO_DE_RUA	-0.012315
IDADE	-0.019675
SEXO_MF	-0.025996

TEMPO DE RUA	
Ate seis meses	0
Entre seis meses e um ano	1
Entre um e dois anos	2
Entre dois e cinco anos	3
Entre cinco e dez anos	4
Mais de dez anos	5
CONTATO PARENTE	
Nunca	0
Quase nunca	1
Todo ano	2
Todo mes	3
Toda semana	4
Todo dia	5
SEXO	
masculino	0
feminino	1
BOLSA FAMILIA	
sim	1
não	0
GRAU DE INSTRUÇAO	
Nao Informado	0
Sem instrucao	1
Fundamental incompleto	2
Fundamental completo	3
Medio incompleto	4
Medio completo	5
Superior incompleto ou mais	6
COR RAÇA	
Nao Informado	0
Preta	1

PREDIÇÃO:

```
#IMPORTS
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

```
df_pred.head()
```

	CONTATO_PARENTE	G_INSTRUCAO	BOLSA	COR_R	IDADE	SEXO_MF	TEMPO_DE_RUA
0	3	2	1	1	63	1	2
1	1	4	0	4	35	0	0
2	0	2	1	4	58	1	4
3	5	2	0	4	63	1	0
4	2	2	1	4	61	0	4

Double-click (or enter) to edit

```
X = df_pred.drop(columns=['TEMPO_DE_RUA'])
y = df_pred['TEMPO_DE_RUA']
```

```
model = DecisionTreeClassifier()
model.fit(X_train,y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=No
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

```
p = model.predict(X_test)
accuracy_score(y_test, p)
```

```
0.7787991805431528
```

```
#Criar subconjunto para IDADE
X = df_pred.drop(columns=["IDADE"])
y = df_pred["IDADE"]
X_treino, X_teste,y_treino,  y_teste = train_test_split(X,y, test_size = 0.1)
```

```
#Criar modelo
```

```
modelo = DecisionTreeClassifier()
modelo.fit(X_treino,y_treino)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=No
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

```
# [bolsa, idade, tempo de rua, contato com parente, sexo, cor, grau de instrução]
#prever a idade de uma pessoa que
previsao = modelo.predict([[1,60,3,1,1,2]])
previsao
```

```
array([2])
```

prever esse valor mesmo que não seja o caso (e quase sempre ele irá acertar)