Projeto Investigando dados Titanic

CONTROLE DE DOCUMENTO

Autor	Janaina Liziane Ferreira
Criado	1/6/2019 8:13
Ultima edição	1/8/2019 9:26

CONTEÚDO

1. Objetivo do Documento

O presente documento descreve os procedimentos realizados na análise da base de dados titanic-data.

A base de dados foi fornecida pela equipe da Udacity como objeto de análise para Projeto final do curso Fundamentos Data Science I. Trata-se de uma amostra dos dados demográficos de cerca de 891 passageiros da viagem inaugural do navio RMS Titanic, viagem essa que culminou em uma das maiores tragédias marítimas da história.

Entre passageiros e tripulação havia a bordo cerca de 2.224 pessoas.

O Titanic e seus passageiros vem gerando curiosidade no passar dos anos e com este projeto esperamos responder algumas perguntas e adentrar nessa fascinante história através da análise de dados.

Incluem os detalhes das atividades:

- Análise e exploração dos dados
- Brainstorm
- Wrangle – Limpeza-preparação-arrumação dos dados
- Demonstração das conclusões

2. Atividades Realizadas

2.1. Importação bibliotecas

- Numpy: pacote que suporta arrays e matrizes multidimensionais e possui uma imensa quantidade funções matemáticas para trabalhar com estas estruturas tornando o trabalho de análise de dados menos moroso.

- Seaborm e Matplotlib: basicamente um pacote para visualização de gráficos que facilitam a comunicação dos resultados obtidos nas análises dos dados.

- Pandas: pacote para análise e manipulação de dados. Possui ferramentas para ler e gravar dados entre estruturas de dados de memoria e diferentes formatos de arquivo.

2.1. Importação bibliotecas

In [40]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib as plt
```

2.2. Carregando os dados e verificando seu conteúdo

Para carregar os dados vamos utilizar a função read_csv do pacote Pandas e criar uma estrutura chamada Data Frame.

Vamos utilizar a função head() para verificar o conteúdo carregado. Esta função retorna os 5 primeiros registros carregados ou podemos passar como argumento a quantidade de linhas desejadas na visualização.

Para verificar o tipo vamos utilizar dtypes.

Usamos () quando utilizamos uma função como o caso de read_csv(), head(). No caso de dtypes trata-se de atributo do dataframe, portanto não existe o ().

2.2. Carregando os dados e verificando seu conteúdo

In [41]:
```
df=pd.read_csv("D:/Python/Projeto 2 Udacity/titanic-data-6.csv")
df.head()
```

Out[41]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [42]:
```
df.dtypes
```

Out[42]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

O shape é outro atributo interessante e nos permite ver a quantidade de linhas e colunas presentes no data frame criado.

In [43]:
```
df.shape
```

Out[43]: (891, 12)

Um resumo estatístico pode ser visto com a função describe()

Com isto já podemos ter algumas informações:

- Total de passageiros: 891
- Idade média: passageiros com aproximadamente 30 anos
- Ticket (tarifa de embarque) médio 32
- O passageiro mais jovem tinha menos de um ano: uma criança do sexo masculino
- O passageiro mais velho tinha 80 anos, também do sexo masculino

In [44]:
```
df.describe()
```

Out[44]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
In [7]: df["Age"].min()

Out[7]: 0.42

In [8]: df["Age"].max()
Out[8]: 80.0
```

Podemos discriminar por sexo utilizando a função group by e aplicando as funções de min e max para obter a estatística por sexo.

```
In [9]: df.groupby(by="Sex")["Age"].min()
Out[9]: Sex
        female    0.75
        male      0.42
        Name: Age, dtype: float64
```

Ou para dados mais detalhados utilizar a função describe()

```
In [12]: df.groupby(by="Sex")["Age"].describe()
```

Out[12]:

Sex	count	mean	std	min	25%	50%	75%	max
female	261.0	27.915709	14.110146	0.75	18.0	27.0	37.0	63.0
male	453.0	30.726645	14.678201	0.42	21.0	29.0	39.0	80.0

2.3. Formulando as perguntas

Em um acidente da proporção e repercussão do naufrágio do Titanic é comum o interesse da população em geral. E este desastre em particular causa curiosidade mesmo nos dias atuais.

Com base nas informações presentes no arquivo construímos as seguintes perguntas:

- Fator sobrevivência foi influenciado por:
  - Valor ticket/classe
  - Idade
  - Sexo
  - Viajar sozinho ou em família

2.4. Qualidade de dados

Para nossas análises e importante verificar a qualidade dos dados e para tal vamos validar se existem dados faltantes ou nulos.

2.4. Qualidade de dados

```
In [45]: df.isnull().sum()
```

```
Out[45]: PassengerId      0
         Survived         0
         Pclass           0
         Name             0
         Sex              0
         Age            177
         SibSp            0
         Parch            0
         Ticket           0
         Fare             0
         Cabin          687
         Embarked         2
         dtype: int64
```

Temos a informação cabine com dados nulos ou faltantes, mas neste caso em específico cabine provavelmente deve estar associada ao valor do ticket e um possível motivo para este campo vir vazio seria que tickets de valores menores não contavam com este recurso.

2.5. Limpeza dos dados

Como a informação da cabine não impacta na resposta das perguntas formuladas no passo anterior optamos por eliminar esta coluna.

A idade dos passageiros que está nula tratamos com a média da idade informada dos demais passageiros e preenchemos as lacunas.

Temos também a coluna Embarked, mas como foram somente 2 registros com este dado incompleto optamos por usar a moda. A moda consiste em repetir o valor que mais aparece no restante do conjunto de dados.

```
In [79]: df["Age"]=df["Age"].fillna(df["Age"].mean())
         df.drop("Cabin",axis=1,inplace=True)
         df.Embarked.fillna(df["Embarked"].mode()[0],inplace=True)
         df.isnull().sum()
```

```
Out[79]: PassengerId    0
         Survived       0
         Pclass         0
         Name           0
         Sex            0
         Age            0
         SibSp          0
         Parch          0
         Ticket         0
         Fare           0
         Embarked       0
         dtype: int64
```

O arquivo não possui dados duplicados e, portanto, está consistente sem ser necessário tratamento.

```
dtype: int64
```

```
In [46]: df.duplicated().sum()
```

```
Out[46]: 0
```

2.6. Criando colunas

Para colaborar com a resposta das perguntas formuladas iremos criar duas colunas adicionais: QtdeFamilia: para indicar a quantidade de pessoas que o tripulante/passageiro viajava em sua companhia e a coluna Faixa etária classificando o tripulante/passageiro de acordo com sua idade em um grupo.

2.6. Criando colunas

In [18]:
```
df["QtdeFamilia"]=df.SibSp+df.Parch+1
df.Age=df.Age.astype(int)
df["FaixaEtaria"] = pd.cut(df.Age,range(0,90,15))
df.head()
```

Out[18]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Embarked	QtdeFamilia	FaixaEtaria
0	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.2500	S	2	(15, 30]
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38	1	0	PC 17599	71.2833	C	2	(30, 45]
2	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.9250	S	1	(15, 30]
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1000	S	2	(30, 45]
4	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.0500	S	1	(30, 45]

2.7. Verificando correlações

As correlações é uma relação linear entre duas variáveis quantitativas, não nos serve para responder perguntas.

É interpretado como o quociente entre a covariância entre duas variáveis e o produto de seus desvios-padrão.

Uma correlação pode resultar valores entre -1 e 1. E quando valor está mais próximo desses extremos, o relacionamento será mais forte. Normalmente, valores entre 0,3 e -0,3 são considerados muito baixos, e de 0,6 ou 0,7 em qualquer um dos dois sinais, quando estamos falando de correlações fortes.

```
In [87]: df.corr()
```

Out[87]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	QtdeFamilia
PassengerId	1.000000	-0.005007	-0.035144	0.033741	-0.057527	-0.001652	0.012658	-0.040143
Survived	-0.005007	1.000000	-0.338481	-0.067809	-0.035322	0.081629	0.257307	0.016639
Pclass	-0.035144	-0.338481	1.000000	-0.335071	0.083081	0.018443	-0.549500	0.065997
Age	0.033741	-0.067809	-0.335071	1.000000	-0.232743	-0.176744	0.093856	-0.247370
SibSp	-0.057527	-0.035322	0.083081	-0.232743	1.000000	0.414838	0.159651	0.890712
Parch	-0.001652	0.081629	0.018443	-0.176744	0.414838	1.000000	0.216225	0.783111
Fare	0.012658	0.257307	-0.549500	0.093856	0.159651	0.216225	1.000000	0.217138
QtdeFamilia	-0.040143	0.016639	0.065997	-0.247370	0.890712	0.783111	0.217138	1.000000

```
In [88]: corr=round(df.corr(),3)
         corr.style.background_gradient()
```

Out[88]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	QtdeFamilia
PassengerId	1	-0.005	-0.035	0.034	-0.058	-0.002	0.013	-0.04
Survived	-0.005	1	-0.338	-0.068	-0.035	0.082	0.257	0.017
Pclass	-0.035	-0.338	1	-0.335	0.083	0.018	-0.549	0.066
Age	0.034	-0.068	-0.335	1	-0.233	-0.177	0.094	-0.247
SibSp	-0.058	-0.035	0.083	-0.233	1	0.415	0.16	0.891
Parch	-0.002	0.082	0.018	-0.177	0.415	1	0.216	0.783
Fare	0.013	0.257	-0.549	0.094	0.16	0.216	1	0.217
QtdeFamilia	-0.04	0.017	0.066	-0.247	0.891	0.783	0.217	1

```
In [ ]:
```

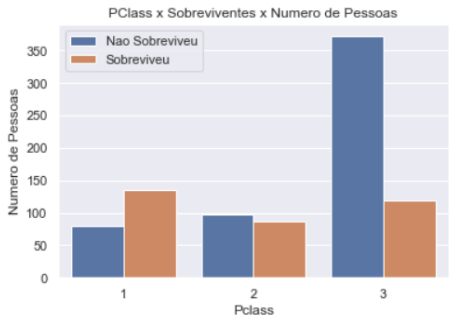
Toda variável e altamente correlacionada com ela própria, por isso temos na diagonal os valores 1.

Os quadrantes em cores mais fortes mostram as correlações mais fortes e as cores claras a menor correlação.

2.8. Exploração

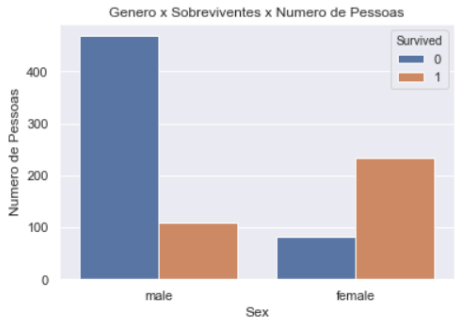
Verificando se a classe influenciou no número de sobreviventes

```
In [96]: ax=sns.countplot(x="Pclass", hue="Survived",data=df)
         ax.set_ylabel("Numero de Pessoas")
         ax.set_title("PClass x Sobreviventes x Numero de Pessoas")
         plt.legend(["Nao Sobreviveu","Sobreviveu"])
         plt.show()
```



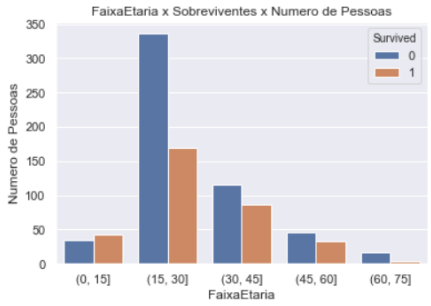
Verificando se o gênero influenciou no número de sobreviventes

```
In [112]: ax=sns.countplot(x="Sex", hue="Survived",data=df)
          ax.set_ylabel("Numero de Pessoas")
          ax.set_title("Genero x Sobreviventes x Numero de Pessoas")
          plt.legend(["Nao Sobreviveu","Sobreviveu"])

Out[112]: <matplotlib.legend.Legend at 0x1c8be0ddeb8>
```



Verificando a influência de faixa etária na sobrevivência do passageiro

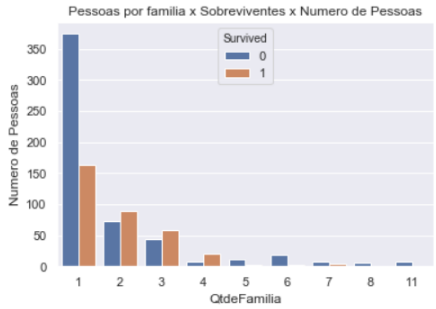
```
In [113]: ax=sns.countplot(x="FaixaEtaria", hue="Survived",data=df)
          ax.set_ylabel("Numero de Pessoas")
          ax.set_title("FaixaEtaria x Sobreviventes x Numero de Pessoas")
          plt.legend(["Nao Sobreviveu","Sobreviveu"])

Out[113]: <matplotlib.legend.Legend at 0x1c8be0fe048>
```




```
In [ ]:
```

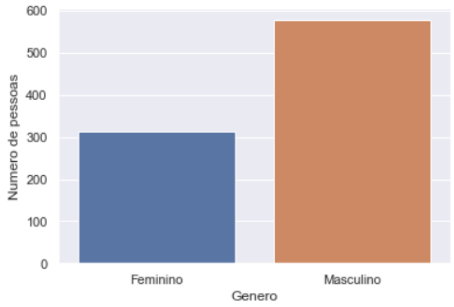
In [115]:
```
ax=sns.countplot(x="QtdeFamilia", hue="Survived",data=df)
ax.set_ylabel("Numero de Pessoas")
ax.set_title("Pessoas por familia x Sobreviventes x Numero de Pessoas")
plt.legend(["Nao Sobreviveu","Sobreviveu"])
```

Out[115]: <matplotlib.legend.Legend at 0x1c8bcb2a0b8>



In [119]:
```
data=np.unique(df.Sex,return_counts=True)
labs=["Feminino", "Masculino"]

plt=sns.barplot(x=labs,y=data[1])
plt.set(xlabel="Genero")
plt.set(ylabel="Numero de pessoas")
```

Out[119]: [Text(0,0.5,'Numero de pessoas')]



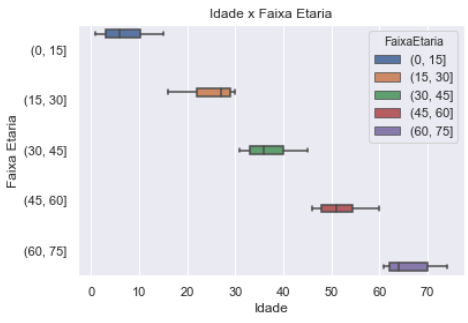
Distribuição dos passageiros por classe

In [29]:
```
fig, ax = plt.subplots(subplot_kw=dict(aspect="equal"))

labels=["1 Classe","2 Classe", "3 Classe"]

ax.legend(wedges, labels,
          title="Ingredients",
          loc="center left",
          bbox_to_anchor=(1, 0, 0.5, 1))

plt.setp(autotexts, size=8, weight="bold")

ax.set_title("Numero de Pessoas x Classe")

plt.pie(x=dataC[1],
       labels=dataC[0],
       shadow=True,
       startangle=90,
       radius=0.9)
plt.show()
```

In [50]:
```
def box(x,y,hue,data_frame):
    """
    Função que que retorna um grafico baseado nos dados informados.
      Parametros:
          data: Data frame base para construcao do grafico.
          x: eixo X do grafico, considerado o valor independente.
          y: eixo y do grafico, considerado o valor dependente
      Retorna:
          Nao possui retorno
    """
    ax=sns.boxplot(x=x, y=y, hue=hue, data=data_frame)
    ax.set_ylabel("Faixa Etaria")
    ax.set_xlabel("Idade")
    ax.set_title("Idade x Faixa Etaria")
```

In [48]:
```
sns.set()
dependent = "FaixaEtaria"
independent = "Age"
box(independent,"FaixaEtaria",dependent,df)
```

Out[48]: <matplotlib.axes._subplots.AxesSubplot at 0x29de10d7fd0>



3. Conclusões

- Classe da Passagem: o fator de sobrevivência é influenciado pela classe, as melhores classes têm um melhor índice de sobrevivência.
- Famílias com tamanhos menores (2-4) tiverem melhor chance comparado com famílias grandes e quem viajou sozinho
- Idade não possui uma correlação forte com sobrevivência, mas foi possível identificar que crianças menores de 5 anos tiveram um índice de sobrevivência considerável.
- Gênero: Mulheres tiveram preferência no uso de botes salva vidas

Em termos de sobrevivência podemos concluir que temos 2 fatores influenciaram:

- Moral: preferência para mulheres, crianças e famílias no uso do bote
- Poder aquisitivo

Podemos concluir que a amostra e significa pois 38% sobreviveram e o no número total temos 30% - valor total do conjunto. Como o percentual da amostra está próximo ao percentual do conjunto podemos dizer que ela é representativa.

3.1. Limitações

Esta base é uma amostra do total de passageiros que estavam no fatídico dia do naufrágio do Titanic.

Dados faltantes ou nulos impactam nas análises.

A coluna Cabin não fazia parte das perguntas de análise e como possuía muito dados nulos optamos por remove-la.

Para tratar idade optamos por media, como a média gera aproximação este dado merece mais cuidado nas interpretações.

Por último e não menos importante usamos o valor moda da coluna para preencher os dados faltantes, como são 2 registros e este não foi usado na análise não se compromete nossas descobertas.

Considerando que existem mais dados envolvidos em um acidente da proporção do naufrágio do Titanic esta analise não é conclusiva.

3.2. Referencias.

https://github.com/dmaiabjj/dsnd-project-titanic-survival

https://paulovasconcellos.com.br/28-comandos-%C3%BAteis-de-pandas-que-talvez-voc%C3%AA-n%C3%A3o-conhe%C3%A7a-6ab64beefa93

https://www.linkedin.com/learning/python-para-data-science-y-big-data-esencial