Aluno: Revely Macauly de Lima Santana.

Matrícula: 202301516773

Tutora: CARLA C.

Acidentes de Trânsito

Será que existe uma forma de podermos mitigar problemas dos acidentes de trânsito, baseado em dados?

Projetos Sociais

Projetos sociais, foi definido como o tema macro da nossa proposta de projeto de extensão da matéria de **Análise de Dados**. Por ser uma temática ampla, diversos subtemas foram sugeridos: Análise dos Gastos Públicos Locais, Qualidade da Água nos Bairros, Acidentes de Trânsito na Região entre outros.

A escolha pelo tema final se deu pela grande disponibilidade de informações, já que a **Polícia Rodoviária Federal** divulga os dados de acidentes rodoviários anualmente.

Esse problema **custa** para a **sociedade brasileira** cerca de 50 **bilhões** de reais por ano. São cerca de 40 **Bilhões** com acidentes em **rodovias e 10 bilhões em áreas urbanas** (IPEA Junho/2020).

Uma possível solução poderia interessar a diversos grupos, como por exemplo: motoristas, órgãos públicos, concessionárias de rodovias e empresas do segmento logístico. A partir disso, algumas perguntas aparecem dentro deste contexto:

° Onde podemos alocar investimentos e recursos de forma a aumentar a segurança nas estradas?

° Neste momento, quais os pontos mais prováveis de ocorrer um acidente?

° Dado um acidente qualquer, podemos prever a gravidade do acidente?

A base de dados

Desde 2007, a Polícia Rodoviária Federal (PRF) realiza a coleta de dados de acidentes em rodovias federais e as disponibiliza de forma aberta em seu site.

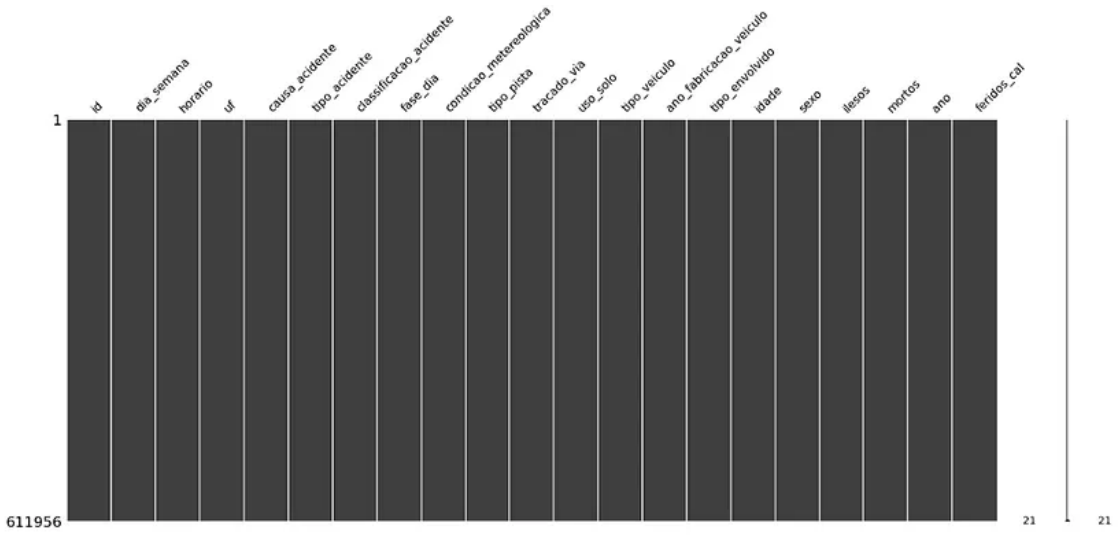
Após analisar as informações em cada formato, escolhi seguir com os dados agrupados por pessoa que possui uma gama maior de informações do que no formato com dados agrupados por ocorrência.

E nessa grande base de dados, veremos dados específicos do acidente, como por exemplo: a causa (motivo do acidente), o tipo de acidente (como ocorreu), a classificação do acidente (com vítimas feridas, mortas e ilesos), a fase do dia (amanhecer), o uso do solo (asfalto), o tipo de pista (dupla, simples etc.), tipo do veículo (carro, moto etc.) e dentre outras.

Assim, olhando para os dados, percebemos nossas possibilidades e decidimos por responder a terceira pergunta apresentada anteriormente, já que no nosso dataset temos a classificação do acidente da PRF.

Antes de iniciarmos a exploração dos dados, direcionamos o olhar ao dataset de forma a identificar quais colunas mais importantes para dar continuidade ao seu tratamento. Durante essa escolha decidimos criar um dataset somente com essas colunas, e em decisão conjunta pelo grupo, eliminamos algumas colunas, como por exemplo: ‘nome da rodovia’(BR), ‘marca’, ‘latitude’, ‘longitude’, ‘regional’, ’delegacia’ e ‘uop’. Com isso, poderíamos direcionar nossos esforços na correção dos dados em cima das colunas que iremos utilizar.

Decidimos avaliar se no nosso novo dataset havia valores que deveriam ser corrigidos. Durante essa análise, foram identificados que em algumas colunas numéricas havia alguns problemas em relação à idade da vítima e ao ano do veículo, onde era possível encontrar valores fora do intervalo estabelecido, assim sendo, os outliers foram substituídos pelas suas respectivas médias da coluna. Em relação às nossas colunas categóricas, havia valores “Não Informados” e “Ignorados”, assim sendo, decidimos substituir esses valores por valores nulos. A questão é que não poderíamos substituir esses valores de um modo normal, como as numéricas, e não poderíamos simplesmente eliminá-las, já que não havia relação entre a presença de dados com outras colunas, conforme vemos abaixo no gráfico:

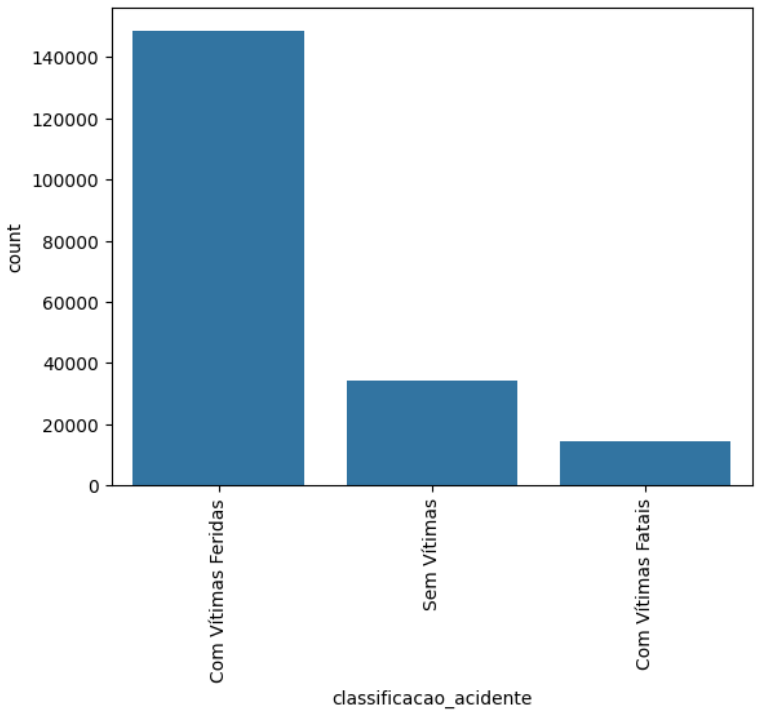


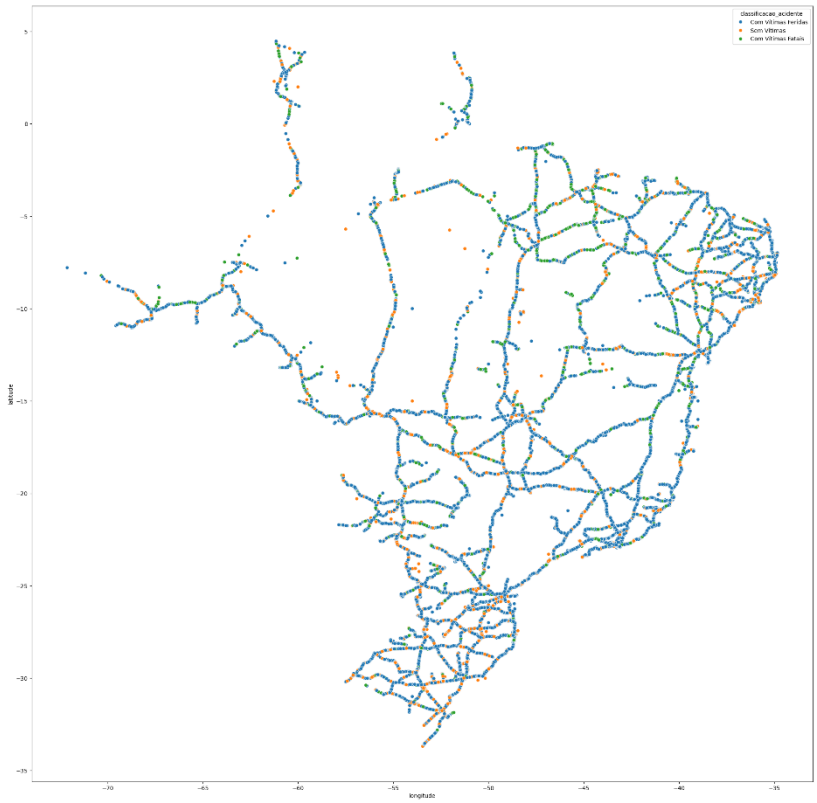

```
def replace_randomly_nan(df_to_replace,column,new_value,size_to_replace):

    df_sample_na = df_to_replace.loc[df_to_replace[column].isnull(),column].sample(size_to_replace).copy()

    for i in df_sample_na.index:
        #print(i)
        df_to_replace.loc[df_to_replace.index == i,column] = new_value
```

Essa fórmula foi utilizada na substituição das variáveis categóricas que apresentavam um número elevado de valores nulos. Na variável que apresentou poucos valores nulos, como podemos ver na coluna ‘Condição meteorológica’ na imagem anterior, decidir em mantê-la já que estamos com a margem de erro quase zerada. Com os dados corrigidos, dataset está pronto para ser utilizado.

Variáveis resposta

A nossa variável resposta ‘classificação acidente’ possui 3 tipos principais: com vítimas fatais, com vítimas feridas e sem vítimas, como pode ser observado abaixo

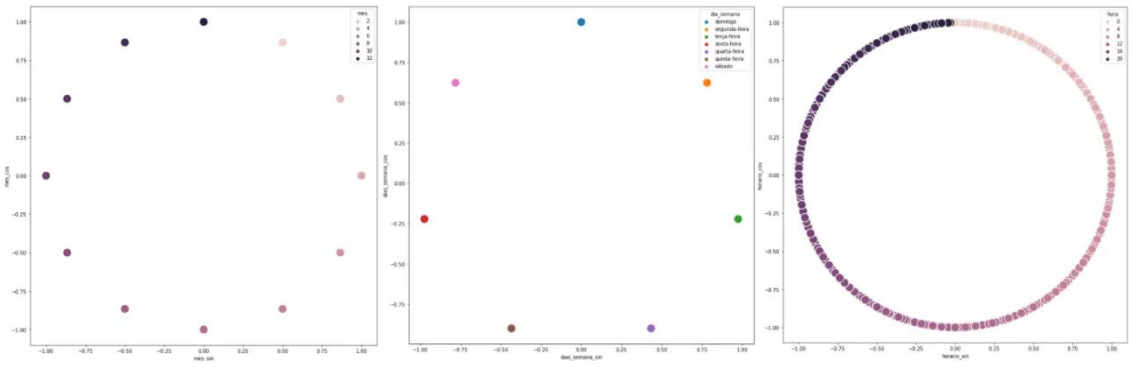




Variáveis Cíclicas

Nossas variáveis cíclicas basicamente serão: ‘dia_semana’, ‘horário’ e ‘data_inversa’

Para cada uma dessas colunas, foram criadas duas colunas: uma com seno e cosseno. Ambas juntas nos passam a ideia do ciclo que queremos que o modelo capture. Uma forma de avaliar essa transformação é plotar essa modelagem graficamente:



Variáveis Contínuas

No nosso Dataset, possuímos as seguintes variáveis contínuas: ‘ano_fabricacao_veiculo’ e ‘ano’, e sendo elas contínuas, não será preciso realizar transformações.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 611956 entries, 0 to 611955
Data columns (total 30 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   id                      611956 non-null  int64
 1   mes                     611956 non-null  int64
 2   mes_sin                 611956 non-null  float64
 3   mes_cos                 611956 non-null  float64
 4   dia                     611956 non-null  int64
 5   dia_semana              611956 non-null  object
 6   dias_semana_sin         611956 non-null  float64
 7   dias_semana_cos         611956 non-null  float64
 8   hora                    611956 non-null  float64
 9   minuto                  611956 non-null  float64
 10  horario_sin             611956 non-null  float64
 11  horario_cos             611956 non-null  float64
 12  uf                      611956 non-null  object
 13  causa_acidente          611956 non-null  object
 14  tipo_acidente           611956 non-null  object
 15  fase_dia                611956 non-null  object
 16  condicao_metereologica  611956 non-null  object
 17  tipo_pista              611956 non-null  object
 18  tracado_via             611956 non-null  object
 19  uso_solo                611956 non-null  object
 20  tipo_veiculo            611956 non-null  object
 21  ano_fabricacao_veiculo  611956 non-null  float64
 22  tipo_envolvido          611956 non-null  object
 23  idade                   611956 non-null  float64
 24  sexo                    611956 non-null  object
 25  ano                     611956 non-null  int64
 26  ilesos                  611956 non-null  int64
 27  feridos_cal             611956 non-null  int64
 28  mortos                  611956 non-null  int64
 29  classificacao acidente  611956 non-null  object
```

Revisão dos dados filtrados

Como apresentamos anteriormente, para a construção dos modelos iniciais, retiramos diversas colunas que em um primeiro momento não pareciam necessárias para o nosso projeto. Para esse novo modelo, decidimos voltar com todas as colunas e deixar o modelo avaliar quais seriam as melhores através do RFECV (Recursive Feature Elimination with Cross-Validation).

Mas antes de obtermos as recomendações das principais variáveis, precisamos ajustar algumas delas, que são os dados de latitude e longitude do acidente.

No primeiro momento, o que conseguimos identificar é que os dados referentes ao ano de 2017 possuem alguns valores discrepantes. Uma vez que estamos falando de acidentes no Brasil, podemos verificar que este ano possuem dados que fogem da zona máxima de latitude e longitude.

Para a correção dos dados de latilon, separamos os dados com problemas em um dataset mantendo os indexes do dataset principal, adicionamos uma nova coluna com o endereço na composição rodovia (BR-000) + km da rodovia + município + UF (exemplo: BR-101, Km 51 — Timbó, Abreu e Lima — PE), e utilizamos a biblioteca **geopy** para obter os valores de latitude e longitude.

```
from geopy.geocoders import Nominatim
geolocator = Nominatim(user_agent="Acidente")
location = geolocator.geocode('Description_Location')
```

```
# Funções que serão chamadas que recebem uma string contendo descrição da localização e retornar latitude ou longitude ;

def latitude(x):
    try:
        x=str(x)
        x = geolocator.geocode(x)
        return x.latitude
    except:
        return np.nan


def longitude(x):
    try:
        x=str(x)
        x = geolocator.geocode(x)
        return x.longitude
    except:
        return np.nan
```

```
# Responsável por iterar nos endereços

for index, item in df_problema_17['endereco'].iteritems():

    try:

        df_problema_17['latitude'][index] = latitude(item)
        df_problema_17['longitude'][index] = longitude(item)

        #print(latitude(item),longitude(item))

    except:

        df_problema_17['latitude'][index] =np.nan
        df_problema_17['longitude'][index]=np.nan
```

Mesmo aplicando as correções nos dados de latilon, apenas uma parte foi corrigida e os dados não recuperados foram retirados do dataset pois apresentavam inconsistências nas informações necessárias para a correção.

Uma vez corrigidos os dados de latitude e longitude no dataset, iniciamos a segunda fase dos modelos, onde obtivemos os seguintes valores:

	Acurácia %	Classificação	Precision	Recall	F1	AUC
XGBClassifier	67.79	Sem vítimas	0.26	0.13	0.18	0.56
		Com vítimas	0.73	0.91	0.81	0.55
		Fatais	0.46	0.02	0.03	0.66
DecisionTreeClassifier	56.59	Sem vítimas	0.23	0.24	0.24	0.52
		Com vítimas	0.73	0.71	0.72	0.52
		Fatais	0.11	0.13	0.12	0.52
RandomForestClassifier	**70.39**	**Sem vítimas**	**0.25**	**0.04**	**0.07**	**0.54**
		Com vítimas	**0.72**	**0.97**	**0.83**	**0.52**
		Fatais	**0.27**	**0.00**	**0.01**	**0.57**
GaussianNB	62.93	Sem vítimas	0.14	0.07	0.09	0.55
		Com vítimas	0.71	0.85	0.77	0.48
		Fatais	0.10	0.05	0.06	0.49
KNeighborsClassifier	68.32	Sem vítimas	0.30	0.11	0.16	0.56
		Com vítimas	0.73	0.91	0.81	0.56
		Fatais	0.24	0.10	0.14	0.60

Modelo Final

Com base nas informações retornadas pelo RFECV, criamos uma lista com apenas as features definidas para rodar novamente o RandomForestClassifier com o objetivo principal de aumentar e equilibrar o recall para as classes desbalanceadas. Outro ponto foi que priorizamos acertos em mortos e feridos, sem vítimas poderia ter menor recall e precisão.

Ponto importante, os balanceamentos foram realizados apenas para os dados de teste, sendo que o de treino se manteve intacto para efeito de validação.

Após a execução do modelo, obtivemos os resultados abaixo:

	Acurácia %	Classificação	Precision	Recall	F1	AUC
Normal	**89.56**	**Sem vítimas**	**0.86**	**0.66**	**0.74**	**0.92**
		Com vítimas	**0.90**	**0.97**	**0.93**	**0.93**
		Fatais	**0.91**	**0.86**	**0.88**	**0.96**
Undersampling	60.63	Sem vítimas	0.36	0.63	0.46	0.75
		Com vítimas	0.88	0.57	0.69	0.78
		Fatais	0.39	0.90	0.54	0.94
Oversampling	88.87	Sem vítimas	0.79	0.70	0.74	0.92
		Com vítimas	0.91	0.94	0.93	0.94
		Fatais	0.88	0.86	0.87	0.96
SMOT	87.33	Sem vítimas	0.76	0.72	0.74	0.92
		Com vítimas	0.93	0.91	0.92	0.94
		Fatais	0.72	0.88	0.79	0.97
ADASYN	87.23	Sem vítimas	0.76	0.72	0.74	0.92
		Com vítimas	0.93	0.91	0.92	0.94
		Fatais	0.70	0.88	0.78	0.97

Conclusão

Conseguimos chegar em um modelo que acerta mais do que o puro acaso (33%) e que classifica de forma satisfatória cada label, apesar de existir um desbalanceamento de classes, importante notar que a acurácia não é uma métrica boa para avaliarmos nosso modelo desbalanceado por isso optamos pelo F1 que se mostrou bem mais coerente para as avaliações.

Confirmamos a importância de definir a pergunta de negócio antes mesmo de começarmos a EDA, isso direciona as análises de forma mais eficiente. Um ponto de atenção é a eliminação das variáveis isso nos direcionou inicialmente a um modelo não satisfatório, por isso é importante que deixemos os dados falarem por si, ao invés de eliminá-los conforme acreditamos ser melhor.

Quando falta uma variável determinante apesar dos esforços para melhoria do modelo, muitas vezes o modelo se torna inaplicável como aconteceu no nosso primeiro caso, o que nos direciona a voltar a discussão dos dados e modelos a procurar de dados mais relevantes.

Importante ressaltar que o modelo funciona para acidente rodoviários e isso deve estar claro para os usuários, prever acidentes fora de rodovia estaria fora do escopo onde o modelo foi treinado.

Próximos passos

Durante o projeto observamos alguns pontos que podem ser otimizados futuramente caso o projeto continue em produção.

Um primeiro ponto seria adicionar o modelo em servidor, para isso precisamos estudar formas alternativas uma vez que o modelo apresenta um tamanho considerável e até o momento não foi colocado em servidor.

Sobre a aplicabilidade seria interessante testar o modelo em campo, de forma a entender sua real aplicabilidade e eficiência.

Entender mais profundamente quais são os vieses de seleção dos acidentes que hoje são registrados, será que existe subnotificação de acidentes sem vítimas, já que alguns acidentes sem vítimas são resolvidos sem a presença da polícia?

https://github.com/Macaulylimacode/projetoestacio