



**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO À DISTÂNCIA**

Analytics & Business Intelligence - Pós-graduação Lato Sensu

RELATÓRIO TÉCNICO
ANÁLISE DE ACIDENTES EM RODOVIAS FEDERAIS BRASILEIRAS

Kaio Oliveira Peixoto

Belo Horizonte
2022

Sumário

1. Introdução	
1.1 Contexto	3
1.2 Objetivos	3
1.3 Público Alvo	3
2. Modelos de Dados	
2.1 Modelo Dimensional	4
3. Integração, Tratamento e Carga	
3.1 Fonte de Dados	5
3.2 Processo de ETL	5
3. Integração, Tratamento e Carga	
3.1 Fonte de Dados	5
3.2 Processo de ETL	5
3.2 Extração	6
3.2 Transformação	6
3.2 Carga	8
4. Camada de Apresentação	
4.1 Métricas	9
4.2 Painel Estratégico	9
4.3 Painel Tático	11
4.4 Painel Operacional	13
4.5 Análises Avançadas	14
5. Registros de Homologação	17
6. Conclusão	20
7. Links	21
Referências	22

1. Introdução

1.1. Contexto

A malha rodoviária brasileira é uma das maiores do mundo. Isso não é novidade quando trata-se de interligar um país de dimensões continentais, o quinto maior do planeta. No entanto, diferentemente de outros países com proporções parecidas como Estados Unidos, Canadá, Rússia e China, o Brasil depende quase que exclusivamente de sua malha rodoviária para transportar pessoas, bens e serviços.

Sendo assim, o fluxo diário de milhões de pessoas e enorme pressão num único modal, aumenta a preocupação da sociedade sobre a ocorrência de acidentes e a segurança como um todo. Neste cenário, surgem questões como quais fatores contribuem para mais acidentes? Quais rodovias e quais trechos são os mais perigosos? Quais cidades e estados tem os maiores números de vítimas fatais?

Para responder a estas e outras perguntas, bem como fazer um diagnóstico da segurança do transporte na malha rodoviária nacional, o presente trabalho analisou a base de dados de acidentes ocorridos nas estradas brasileiras entre os anos de 2007 a 2020, disponibilizada pela Polícia Rodoviária Federal.

1.2. Objetivos

O objetivo principal da análise é trazer compreensão para que medidas eficientes de prevenção possam ser tomadas pelos agentes públicos. No setor logístico, é de grande valia para o planejamento de rotas e elucidação para tomada de decisões.

1.3. Público alvo

Agentes públicos, estudantes e profissionais da área de dados do setor logístico. É também de interesse de qualquer cidadão brasileiro como principal usuário direta ou indiretamente da malha rodoviária federal nacional.

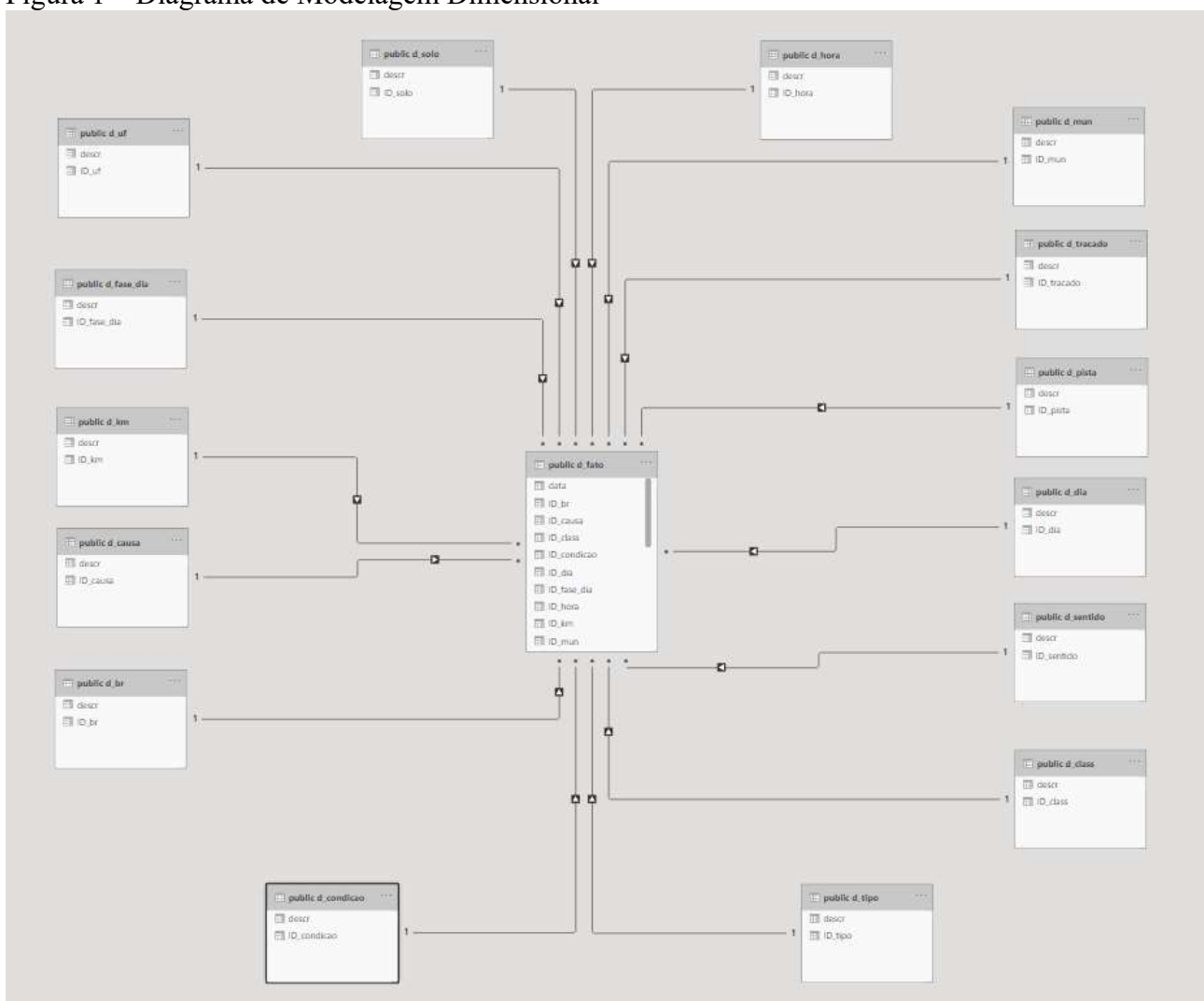
2. Modelo de Dados

2.1. Modelo Dimensional

Para relacionarmos as informações de maneira que seja combinada como um cubo a modelagem desse projeto foi feita com base no modelo *Star Schema* e nas quatro etapas de desenvolvimento: “Selecione o processo de negócio, Declare a granularidade, Identifique as dimensões e Identifique os fatos.” (KIMBALL, 2021).

A análise do arquivo resultou em uma tabela fato e 15 tabelas dimensão. Algumas colunas com ocorrência de muitos valores nulo ou que não eram importantes para a análise foram excluídas do modelo. Veja na figura abaixo como ficou o modelo final em star schema.

Figura 1 – Diagrama de Modelagem Dimensional



3. Integração, Tratamento e Carga de Dados

3.1. Fonte de Dados


A fonte de dados deste estudo é um arquivo csv onde está unificada toda a base coletada pela Polícia Rodoviária Federal entre os anos de 2007 e 2020. O arquivo está no repositório deste projeto no GitHub e o link para pode ser encontrado na seção 6 deste documento. Para facilitar a manipulação e distribuição do arquivo, o mesmo foi reduzido para o formato tar.xz.

3.2. Processo de ETL

O banco de dados escolhido foi o PostgreSQL pelas suas conhecidas vantagens, como economia e alto desempenho. Este SGBD suporta um intenso fluxo de dados com garantia de estabilidade e segurança.

Para o fluxo de ETL foi utilizada a linguagem de programação Python além das bibliotecas Pandas, Numpy e SQLAlchemy para manipulação e análise de dados. O SQLAlchemy permite transformar o banco de dados num objeto manipulável em Python, facilitando assim nossa integração e trabalho de ETL. O repositório encontra-se público, disponível no Link 2. Neste repositório do GitHub encontra-se o notebook onde consta o código Python criado para esta etapa do projeto.

Figura 2 – Jupyter Notebook



```
Importando bibliotecas a serem utilizadas

In [4]: import os
import pandas as pd
import numpy as np
from sqlalchemy import create_engine
import psycopg2
from time import sleep

Leitura do arquivo

In [6]: df = pd.read_csv('acidentes2007-2020.tar.xz', compression='xz', usecols=[
'id', 'id_unico', 'dia_semana', 'horario', 'uf', 'br', 'km', 'municipio',
'causa_acidente', 'tipo_acidente', 'classificacao_acidente', 'fase_dia',
'sentido_via', 'condicao_meteorologica', 'tipo_pista', 'tracado_via',
'uso_solo', 'pessoas', 'mortos', 'feridos_leves', 'feridos_graves', 'ilesos',
'ignorados', 'feridos', 'veiculos'], dtype={'br': 'str', 'km': 'str'})

In [7]: df.columns

Out[7]: Index(['id', 'id_unico', 'dia_semana', 'horario', 'uf', 'br', 'km',
'municipio', 'causa_acidente', 'tipo_acidente',
'classificacao_acidente', 'fase_dia', 'sentido_via',
'condicao_meteorologica', 'tipo_pista', 'tracado_via', 'uso_solo',
'pessoas', 'mortos', 'feridos_leves', 'feridos_graves', 'ilesos',
'ignorados', 'feridos', 'veiculos'],
dtype='object')

In [8]: df.head()

Out[8]:
```

id	id_unico	dia_semana	horario	uf	br	km	municipio	causa_acidente	tipo_acidente	...	tracado_via	uso_solo	pessoas	mortos	feric
----	----------	------------	---------	----	----	----	-----------	----------------	---------------	-----	-------------	----------	---------	--------	-------

3.2.1. Extração

Através da biblioteca pandas descompactamos o arquivo e já fazemos a leitura do mesmo. Veja que também já é possível indicar os tipos de dados de algumas colunas, como 'br' e 'km' que, apesar de dados numéricos, trataremos como strings.

Figura 2 – Leitura do arquivo no Jupyter notebook

```
Leitura do arquivo

In [6]: df = pd.read_csv('acidentes2007-2020.tar.xz', compression='xz', usecols=[
    'id', 'id_unico', 'dia_semana', 'horario', 'uf', 'br', 'km', 'municipio',
    'causa_acidente', 'tipo_acidente', 'classificacao_acidente', 'fase_dia',
    'sentido_via', 'condicao_meteorologica', 'tipo_pista', 'tracado_via',
    'uso_solo', 'pessoas', 'mortos', 'feridos_leves', 'feridos_graves', 'ileso',
    'ignorado', 'feridos', 'veiculos'], dtype={'br': 'str', 'km': 'str'})

In [7]: df.columns

Out[7]: Index(['id', 'id_unico', 'dia_semana', 'horario', 'uf', 'br', 'km',
    'municipio', 'causa_acidente', 'tipo_acidente',
    'classificacao_acidente', 'fase_dia', 'sentido_via',
    'condicao_meteorologica', 'tipo_pista', 'tracado_via', 'uso_solo',
    'pessoas', 'mortos', 'feridos_leves', 'feridos_graves', 'ileso',
    'ignorado', 'feridos', 'veiculos'],
    dtype='object')
```

3.2.2. Transformação

Essa é a parte que ocupa a maior parte do tempo e de processamento. Veja abaixo como iniciamos esta fase conhecendo quais são os tipos de dados de cada coluna.

Figura 3 – Avaliando tipos de dados de cada coluna

```
Conhecendo tipos de dados de cada coluna da nossa base

In [10]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1851866 entries, 0 to 1851865
Data columns (total 25 columns):
#   Column              Dtype
---  ---
0   id                   float64
1   id_unico             object
2   dia_semana           object
3   horario              object
4   uf                   object
5   br                   object
6   km                   object
7   municipio            object
8   causa_acidente       object
9   tipo_acidente        object
10  classificacao_acidente object
11  fase_dia             object
12  sentido_via          object
13  condicao_meteorologica object
14  tipo_pista           object
15  tracado_via          object
16  uso_solo             object
17  pessoas              float64
18  mortos               float64
19  feridos_leves        float64
20  feridos_graves       float64
21  ileso                float64
22  ignorado             float64
23  feridos              float64
24  veiculos              float64
dtypes: float64(9), object(16)
memory usage: 353.2+ MB
```

Depois fazemos uma checagem da qualidade dos dados de todas as colunas. O método `isna()` de pandas nos permite visualizar quais colunas possuem valores nulos e qual a quantidade. Dessa forma podemos ver rapidamente quais colunas precisam de uma revisão.

Figura 4 – Checando qualidade das colunas

Vamos checar quantos e onde estão os valores nulos

```
In [11]: df.isna().sum()

Out[11]: id                1
id_unico                1
dia_semana              1
horario                 1
uf                     1
br                    514
km                    514
municipio              1
causa_acidente         1
tipo_acidente          13
classificacao_acidente 25
fase_dia                2
sentido_via            1
condicao_meteorologica   4
tipo_pista             11
tracado_via            1
uso_solo               1
pessoas               1
mortos                1
feridos_leves          1
feridos_graves         1
ilesos                 1
ignorados              1
feridos                1
veiculos               1
dtype: int64
```

A partir daí iniciamos a limpeza propriamente dita. Abaixo tem um exemplo do primeiro passo dado, que foi excluir colunas com quantidades significativa de valores nulos.

Figura 5 – Iniciando o tratamento

No código abaixo vamos excluir as linhas de colunas que tiveram alguns valores nulos

```
In [12]: df.dropna(subset=['br', 'km', 'tipo_acidente', 'classificacao_acidente',
                        'fase_dia', 'condicao_meteorologica', 'tipo_pista'], axis=0, inplace=True)
```

Como ficou nossa base

```
In [13]: df.head()
```

```
Out[13]:
```

	id	id_unico	dia_semana	horario	uf	br	km	municipio	causa_acidente	tipo_acidente	...	tracado_via	uso_solo	pessoas	mortos	feri
1	1032898.0	10328982007-08-13	segunda	14:25:00	MG	40	585.5	ITABIRITO	outras	Saída de Pista	...	Reta	Rural	3.0	0.0	
2	1051130.0	10511302007-02-12	segunda	02:10:00	MA	135	11.0	SAO LUIS	animais na pista	Atropelamento de Animal	...	Reta	Urbano	5.0	2.0	
3	1066824.0	10668242007-11-20	terça	05:30:00	CE	222	30.8	CAUCAIA	defeito mecânico no veículo	Capotamento	...	Reta	Rural	1.0	0.0	
4	1069918.0	10699182007-12-16	domingo	17:40:00	MA	230	14.0	BARAO DE GRAJAU	outras	Capotamento	...	Curva	Rural	1.0	0.0	
5	1070971.0	10709712007-03-05	segunda	08:10:00	PR	277	584.4	CASCADEL	outras	Colisão Lateral	...	Curva	Urbano	2.0	0.0	

5 rows × 17 columns

Para demais detalhes e informações desta etapa, o link 3 da seção 6 deste projeto oferece acesso ao notebook completo com todas as etapas e descrições, não só da fase de tratamento, mas de todo o processo de ETL.

3.2.2. Carga

Concomitante ao processo de transformação e limpeza foi construída a estrutura física de cada tabela dimensão que compõe este projeto. Abaixo, por exemplo, é mostrada a criação da tabela dimensão d_solo.

Logo após criamos uma função python que modela cada tabela dimensão e já salva na estrutura de tabela correspondente criada no banco de dados PostgreSQL. Daí basta colocar esta função num loop para percorrer por todas as tabelas dimensão da lista campos e concluir o processo de carga.

Figura 6 – Linguagem SQL para criação das tabelas no banco de dados



```
postgres/postgres@PostgreSQL 13 ▾
Query Editor  Query History
1 CREATE TABLE solo (
2   id_solo PRIMATY KEY,
3   descr VARCHAR(50)
4 );
```

Figura 7 – Código criado em Python para carga das tabelas direto para o banco de dados



```
Código para transformar colunas dimensão em tabelas e já salvar para o banco

In [48]: def criar_dim(coluna, salva=False):
          unicos = df_dim[coluna].unique().copy()
          tabela_dim = pd.DataFrame({'ID_{coluna[3:]}':range(1, len(unicos)+1), 'descr':unicos})
          if salva:
              tabela_dim.to_sql(f'd_{coluna[3:]}', con=con, index=False, if_exists='replace')
          return tabela_dim

In [49]: campos
Out[49]: ['ID_dia',
          'ID_hora',
          'ID_uf',
          'ID_br',
          'ID_km',
          'ID_mun',
          'ID_causa',
          'ID_tipo',
          'ID_class',
          'ID_fase_dia',
          'ID_sentido',
          'ID_condicao',
          'ID_pista',
          'ID_tracado',
          'ID_solo']

Aqui vamos criar um laço que vai percorrer por toda a lista acima e criar as tabelas no banco uma a uma

In [50]: for dimensao in campos:
          criar_dim(dimensao, True)
          sleep(5)
```


4. Camada de Apresentação

Após a limpeza, preparação e posterior carregamento do dataset no banco PostgreSQL, o programa Power BI foi escolhido para a fase de visualização dos dados e criação de dashboards. Neles estarão representados números e indicadores que embasarão as estratégias de curto, médio e longo prazos.

Para otimizar a análise dos dados e facilitar na tomada de decisões foram criados painéis do tipo estratégico, tático e operacional.

4.1. Métricas

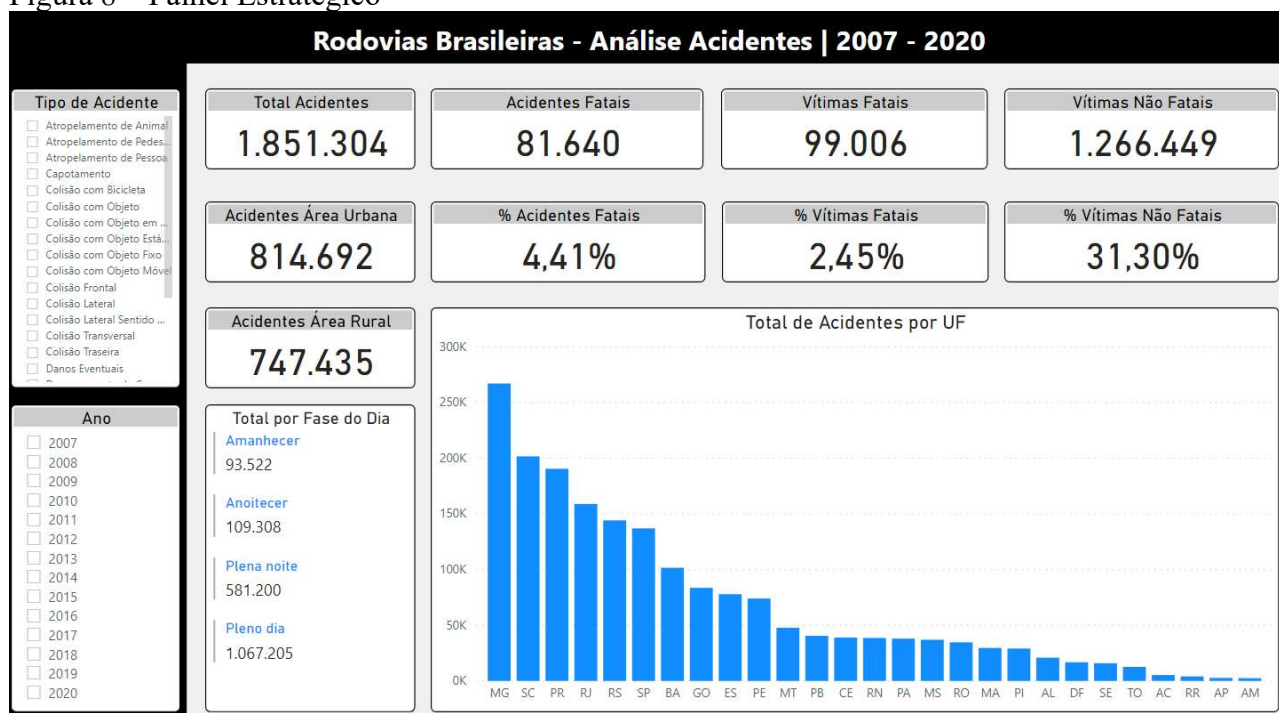
Métricas são medidas utilizadas para analisar e acompanhar o desempenho de um processo, estratégias ou ações de projetos e negócios das mais variadas naturezas, sejam estes com fins lucrativos ou não.

A seguir no projeto serão apresentados e descritos, junto com cada tipo de painel, suas respectivas métricas e filtros relacionados.

4.2. Painel Estratégico

Este tipo de dashboard apresenta indicadores relevantes para a tomada de decisão no longo prazo. O painel abaixo tem o objetivo de apresentar uma visão macro da situação, sem aprofundamento em detalhes. Por exemplo, abaixo pode-se ver total de acidentes, total de vítimas fatais e não fatais e a proporção de acidentes desta natureza em relação ao todo. Para uma análise rápida da situação geral, o trabalho abaixo também traz um ranking na forma de um gráfico de barras verticais por estado da federação. Este ranking também pode servir como segmentador para uma visão estratégica por estado.

Figura 8 – Painel Estratégico



Na figura 9 temos exemplo de métrica calculada a partir da linguagem DAX (*Data Analysis Expression*) do Power BI para aferirmos o total de acidentes com ocorrência de vítimas fatais.

Figura 9 – Métrica para cálculo de acidentes fatais

```
1 acidentes_fatais = CALCULATE([total_acidentes], 'public d_fato'[Total_mortos] <> 0)
```

4.2.1. Filtros no Painel

- **Ano:** segmentação baseada na coluna data da tabela fato.
- **Tipo de Acidente:** filtro baseado na tabela dimensão d_tipo

4.2.2 Indicadores

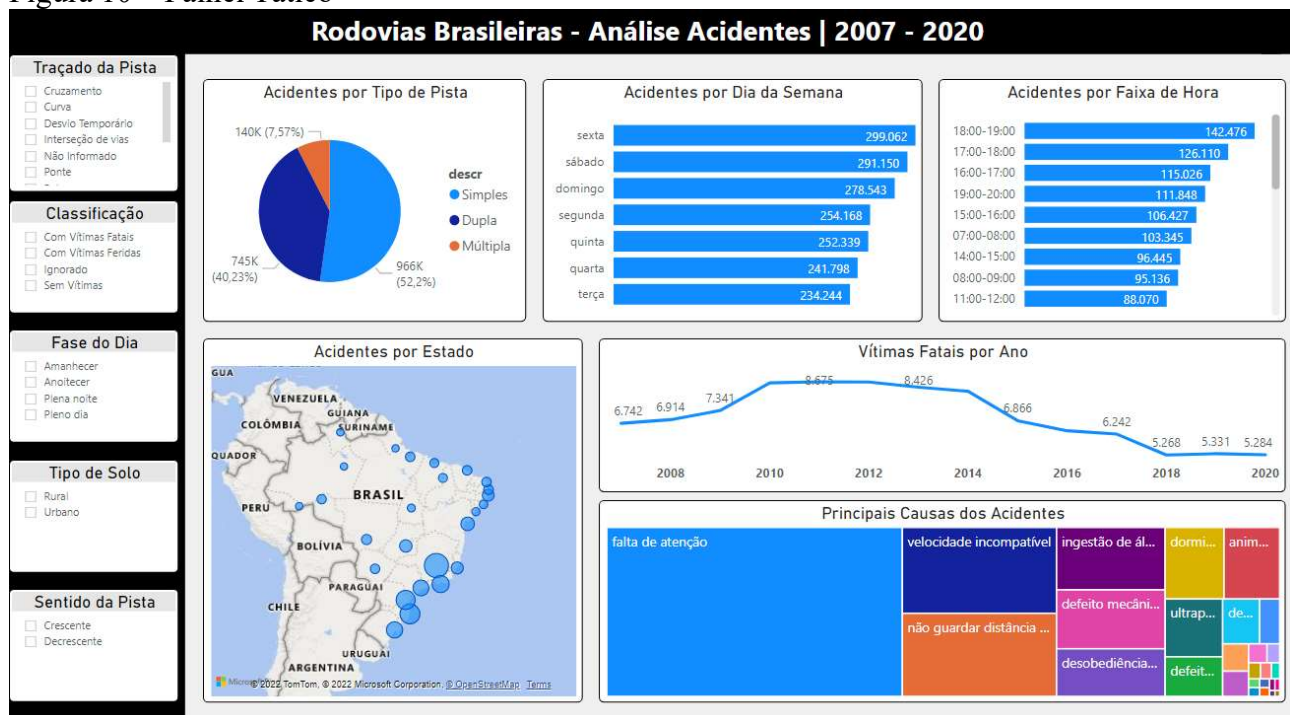
- **Total Acidentes:** contador de total de acidentes de acordo com o filtro utilizado. Foi utilizada a contagem distinta da coluna id_unico da tabela fato. Se cada id único representa um acidente diferente então basta fazemos uma contagem dos mesmos.
- **Acidentes Área Urbana:** reaproveitando a métrica total de acidentes, mas com filtro de visual da tabela dimensão d_tipo, que descreve o tipo de solo.
- **Acidentes Área Rural:** mesmo princípio acima, mas agora com outro tipo de terreno.
- **Acidentes Fatais:** contador do total de acidentes fatais. Neste aqui utilizamos a função CALCULATE do DAX e reutilizamos a medida total_acidentes como primeiro argumento e o filtro da coluna de total de mortos no segundo argumento (ver figura 9).
- **%Acidentes Fatais:** proporção de acidentes fatais em relação à totalidade de acidentes.
- **Vítimas Fatais:** total de vítimas fatais fazendo uma soma simples da coluna Total_mortos da tabela fato.
- **%Vítimas Fatais:** proporção de vítimas fatais em relação ao total de pessoas envolvidas nos acidentes do período.
- **Vítimas Não Fatais:** total de vítimas não fatais fazendo uma soma simples da coluna Total_feridos da tabela fato.
- **%Vítimas Não Fatais:** proporção de vítimas não fatais em relação ao total de pessoas envolvidas nos acidentes do período.
- **Total de Acidentes por UF:** contador acidentes segmentados por estados da federação.

4.3. Painel Tático

Com um nível de detalhe maior, o dashboard tático apresenta indicadores que auxiliam no alcance de objetivos de médio prazo. Para maior descrição, neste painel são combinadas uma maior quantidade de dimensões para melhor visualização do panorama analisado.

Aqui temos uma exposição de acidentes por tipo de pista e principais causas. Também é mostrado um mapa indicando os estados com maior número de acidentes, sendo o tamanho dos círculos proporcionais ao que representa cada unidade. Ainda numa camada maior de detalhe é possível verificar as maiores ocorrências de acidentes por dia e faixa de hora, bem como um indicador de vítimas fatais por ano.

Figura 10 – Painel Tático



4.3.1. Filtros no Painel

- **Traçado da Pista:** segmentação baseada na tabela dimensão d_tracado.
- **Classificação:** baseado na tabela dimensão d_class.
- **Fase do Dia:** baseada na tabela dimensão d_fase_dia.
- **Tipo de Solo:** baseada na tabela dimensão d_solo.
- **Sentido da Pista:** baseada na tabela dimensão d_sentido.

4.3.2. Indicadores

- **Acidentes por Tipo de Pista:** foi utilizada a dimensão d_pista para esta análise.
- **Acidentes por Dia da Semana:** através de código python/pandas foi possível extrair o dia de cada acidente através da coluna data. A partir daí foi criada a tabela dimensão d_dia, de onde foi baseada essa visão.
- **Acidentes por Faixa de Hora:** também com a ajuda do python/pandas, foi possível classificar os acidentes por faixa de hora e posteriormente construir este gráfico com o indicador. Todo o código está disponível no arquivo jupyter notebook que faz parte deste projeto.
- **Acidentes por Estado:** o tamanho de cada círculo está diretamente ligado à quantidade de acidentes naquela localidade.
- **Vítimas Fatais por Ano:** para um maior detalhamento, foi criado um gráfico de linha com o total de vítimas fatais por ano.
- **Principais Causas dos Acidentes:** por se tratar de classificação de uma categoria, foi utilizado um mapa de árvore para descrever as principais causas de acidente.

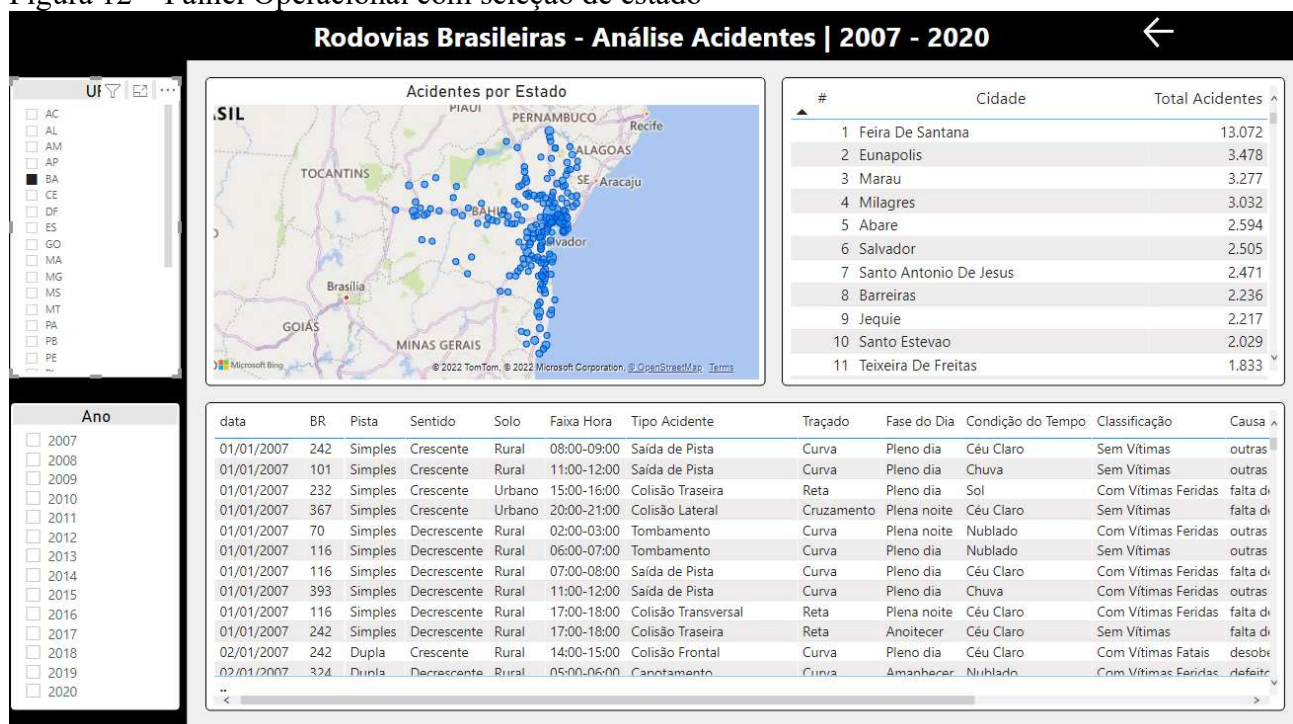
4.4. Painel Operacional

O dashboard operacional fornece indicadores para tomadas de decisão de curto prazo. O painel abaixo propõe uma visão detalhada por cidade de cada estado. Primeiramente, sem nenhuma seleção, todas as cidades do país são mostradas juntamente com um ranking dos maiores números de acidentes por município. Na figura 12, fizemos uma seleção de uma unidade da federação, Bahia, e o painel nos levou diretamente para detalhes de cada cidade deste estado.

Figura 11 – Painel Operacional



Figura 12 – Painel Operacional com seleção de estado



4.5. Análises Avançadas

Esta parte do projeto foi desenvolvido de forma independente ao dashboard, fora do Power BI. Aqui temos que utilizar uma ferramenta de Machine Learning para fazer recomendações e previsões. Para isso foi utilizado o auxílio da biblioteca Phophet da linguagem Python. Prophet é um algoritmo de código aberto criado pelo Facebook para gerar modelos de previsões baseados em séries temporais. Este algoritmo considera sazonalidades e feriados em suas modelagens e, por esse motivo, é a ferramenta de Machine Learning ideal para este trabalho, já que os dados estudados são de ocorrências de acidentes desde o ano de 2007.

Devido às enormes proporções do território brasileiro e as diferentes características que influenciam as regiões, aqui foi feito um recorte do estado de São Paulo somente. Este estado foi o escolhido por concentrar boa parte da população, das ocorrências e oferecer um volume de dados interessante para o treinamento do modelo de Machine Learning.

Favor notar abaixo código utilizado para geração do modelo. Para acesso completo basta acessar o repositório do projeto no Github. O link encontra-se na seção 7.

Figura 12 – Importando biblioteca Prophet e preparando os dados
Módulo C - Análise Avançada

Importando a biblioteca de previsão

```
In [20]: from prophet import Prophet
```

Fazendo uma cópia do conjunto de dados

```
In [137]: data_previsao = df.copy()
```

```
In [151]: # transformando a coluna de data para o tipo de dados datetime
data_previsao['data'] = pd.to_datetime(data_previsao['data'])
```

```
In [152]: # transformando a coluna de total de vítimas para o tipo inteiro
data_previsao['mortos'] = data_previsao['mortos'].astype('int64')
```

```
In [153]: # selecionando somente dados do estado de São Paulo para uma previsão mais precisa
previsao_sp = data_previsao[data_previsao['uf']=='SP']
previsao_sp = data_previsao[['data', 'mortos']].copy()
```

```
In [154]: previsao_sp.rename(columns={'data':'ds', 'mortos':'y'}, inplace=True)
```

```
In [155]: previsao
```


Figura 13 – Treinamento dos dados e rodando as previsões
Modelo de treino

```
In [157]: # aqui vamos instanciar para a classe Prophet
m = Prophet(interval_width=0.95, daily_seasonality=True)

In [158]: # fornecendo os dados para treino
model = m.fit(previsao_sp)
```

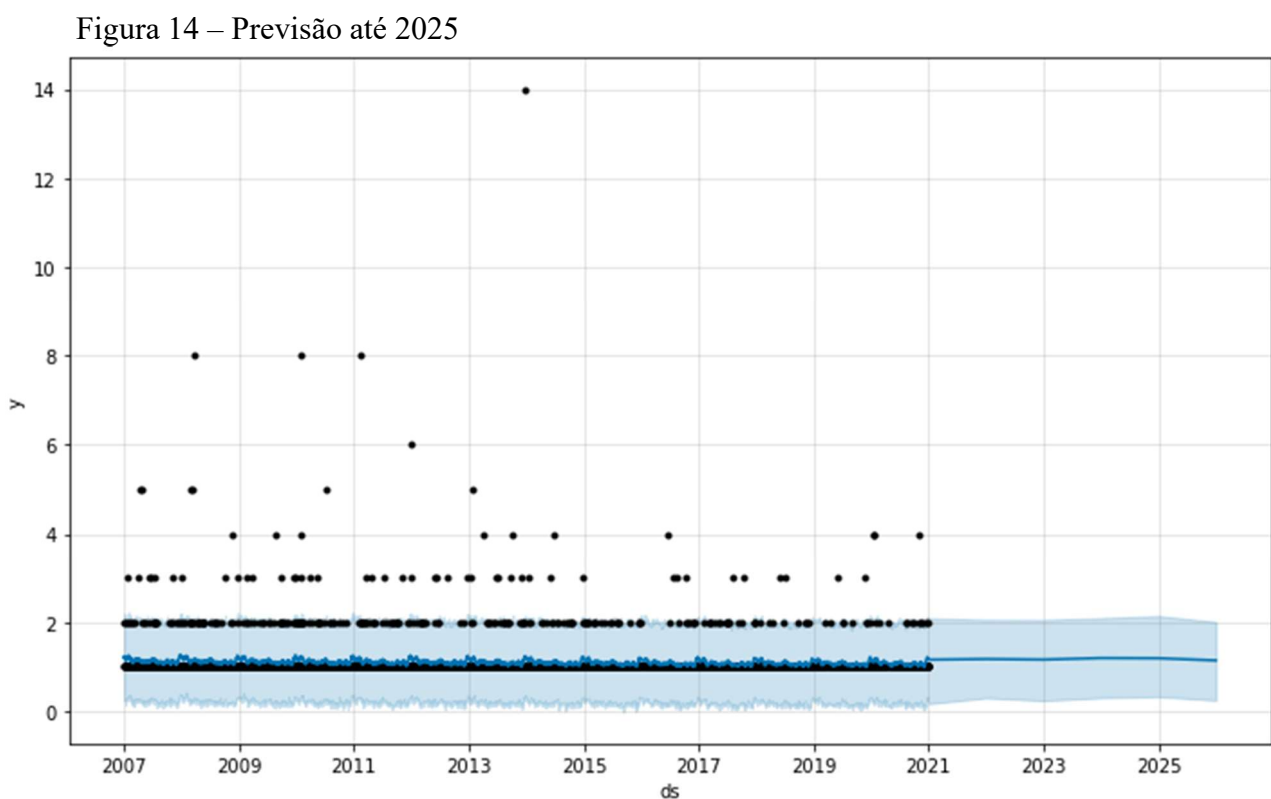
Initial log joint probability = -4.40061

Iter	log prob	dx	grad	alpha	alpha0	# evals	Notes
99	11693.4	4.82125e-06	61.903	1.458	0.1458	136	
103	11693.4	2.91206e-07	78.4726	0.1693	1	143	

Optimization terminated normally:
Convergence detected: relative gradient magnitude is below tolerance

Previsões

```
In [174]: future = m.make_future_dataframe(periods=6, freq='Y')
forecast = m.predict(future)
forecast.tail()
```



Na figura 15, é interessante notar que existe uma tendência de queda até o ano de 2025. Semanalmente os picos permanecem na terça-feira e nos finais de semana, sendo das 20h30 às 03h30 aproximadamente os horários de maior ocorrência.

Figura 15 – Previsões em detalhes



5. Registros de Homologação

Para garantir a integridade e qualidade dos dados, foram feitos testes de homologação com consulta diretamente no jupyter notebook, onde foi feito todo o processo de ETL, comparando com filtros feitos no Power BI. Dessa forma podemos fazer um confronto dos resultados e garantir que se tratam das mesmas informações.

Figura 16 – Contagem de Registros no Power BI

ID_unico	data	ID_dia	ID_hora	ID_uf	ID_br	ID_km	ID_mun	ID_causa	ID_tipo	ID_class	ID_fase_dia	ID_sento	ID_condicao	ID_pista	ID_tracado	ID_solo	Total_pessoas	Total_mortos	Total_feridos leves	Total_feridos grav
1848432007-02-02	02/02/2007	6	23	11	6	1445	130	4	6	3	1	2	1	2	1	2	2	0	0	0
2560942007-07-20	20/07/2007	6	20	11	6	6464	130	4	6	3	1	1	1	2	1	2	2	0	0	0
2600512007-07-27	27/07/2007	6	20	11	6	1259	130	4	6	3	1	2	1	2	1	2	2	0	0	0
2749622007-09-07	07/09/2007	6	16	11	6	7676	130	4	6	3	1	1	1	2	1	2	2	0	0	0
2845592007-10-26	26/10/2007	6	4	11	6	525	130	4	6	3	1	1	1	2	1	2	2	0	0	0
3082412007-11-30	30/11/2007	6	15	11	6	1558	130	4	6	3	1	2	1	2	1	2	2	0	0	0
3090572007-11-30	30/11/2007	6	4	11	6	3533	130	4	6	3	1	1	1	2	1	2	2	0	0	0
3209602007-12-21	21/12/2007	6	4	11	6	1481	130	4	6	3	1	1	1	2	1	2	2	0	0	0
3258842007-12-28	28/12/2007	6	22	11	6	2590	130	4	6	3	1	1	1	2	1	2	2	0	0	0
3423242008-02-19	19/02/2008	6	22	11	6	2113	130	4	6	3	1	2	1	2	1	2	2	0	0	0
3447772008-02-22	22/02/2008	6	14	11	6	399	130	4	6	3	1	1	1	2	1	2	2	0	0	0
3671282008-04-18	18/04/2008	6	21	11	6	2977	130	4	6	3	1	1	1	2	1	2	2	0	0	0
4012582008-07-11	11/07/2008	6	20	11	6	2608	130	4	6	3	1	1	1	2	1	2	2	0	0	0
4066442008-07-25	25/07/2008	6	18	11	6	3254	130	4	6	3	1	2	1	2	1	2	2	0	0	0
4185572008-08-29	29/08/2008	6	5	11	6	4412	130	4	6	3	1	1	1	2	1	2	2	0	0	0
4459962008-10-31	31/10/2008	6	22	11	6	502	130	4	6	3	1	1	1	2	1	2	2	0	0	0
4624112008-12-05	05/12/2008	6	5	11	6	399	130	4	6	3	1	1	1	2	1	2	2	0	0	0
4895612008-12-26	26/12/2008	6	22	11	6	849	130	4	6	3	1	1	1	2	1	2	2	0	0	0
4897582009-01-16	16/01/2009	6	22	11	6	5088	130	4	6	3	1	1	1	2	1	2	2	0	0	0
4840602009-01-30	30/01/2009	6	21	11	6	1558	130	4	6	3	1	2	1	2	1	2	2	0	0	0
5156972009-04-17	17/04/2009	6	17	11	6	1481	130	4	6	3	1	1	1	2	1	2	2	0	0	0
5173542009-04-17	17/04/2009	6	5	11	6	1481	130	4	6	3	1	1	1	2	1	2	2	0	0	0
520082009-04-24	24/04/2009	6	20	11	6	1132	130	4	6	3	1	1	1	2	1	2	2	0	0	0
5428902009-06-19	19/06/2009	6	14	11	6	1674	130	4	6	3	1	1	1	2	1	2	2	0	0	0
5872532009-09-25	25/09/2009	6	21	11	6	1481	130	4	6	3	1	2	1	2	1	2	2	0	0	0
6034382009-10-30	30/10/2009	6	21	11	6	502	130	4	6	3	1	2	1	2	1	2	2	0	0	0
6054442009-10-30	30/10/2009	6	20	11	6	2279	130	4	6	3	1	2	1	2	1	2	2	0	0	0
6053652009-10-30	30/10/2009	6	22	11	6	2113	130	4	6	3	1	1	1	2	1	2	2	0	0	0
6234612009-12-04	04/12/2009	6	20	11	6	2240	130	4	6	3	1	1	1	2	1	2	2	0	0	0
6515112010-01-29	29/01/2010	6	14	11	6	7026	130	4	6	3	1	2	1	2	1	2	2	0	0	0
6516002010-02-12	12/02/2010	6	20	11	6	2977	130	4	6	3	1	2	1	2	1	2	2	0	0	0
7291592010-07-02	02/07/2010	6	1	11	6	4041	130	4	6	3	1	2	1	2	1	2	2	0	0	0
7658262010-09-10	10/09/2010	6	20	11	6	1860	130	4	6	3	1	2	1	2	1	2	2	0	0	0
7710822010-09-17	17/09/2010	6	1	11	6	3088	130	4	6	3	1	1	1	2	1	2	2	0	0	0
7733422010-09-17	17/09/2010	6	1	11	6	1860	130	4	6	3	1	1	1	2	1	2	2	0	0	0
7746532010-09-19	19/09/2010	6	21	11	6	502	130	4	6	3	1	1	1	2	1	2	2	0	0	0
8011052010-11-17	17/11/2010	6	20	11	6	502	130	4	6	3	1	2	1	2	1	2	2	0	0	0

Table: public_d_fato (1,851,304 rows)

Figura 17 – Contagem de Registros no Jupyter Notebook

Registro de Homologação

```
In [89]: # contagem dos registros
df_fato.shape[0]
```

```
Out[89]: 1851304
```

Figura 18 – Acidentes por dia da semana no painel tático



Figura 19 – Acidentes por dia da semana no Jupyter Notebook

Acidentes por dia da semana

```
In [108]: df['ID_dia'].value_counts()
```

```
Out[108]: sexta      299062
sábado      291150
domingo     278543
segunda     254168
quinta      252339
quarta      241798
terça       234244
Name: ID_dia, dtype: int64
```

Figura 20 – Acidentes por faixa hora no painel tátil



Figura 21 – Acidentes por faixa hora no Jupyter Notebook

Acidentes por faixa hora

```
In [109]: df['ID_hora'].value_counts()

Out[109]: 18:00-19:00    142476
          17:00-18:00    126110
          16:00-17:00    115026
          19:00-20:00    111848
          15:00-16:00    106427
          07:00-08:00    103345
          14:00-15:00     96445
          08:00-09:00     95136
          11:00-12:00     88070
          10:00-11:00     86329
          09:00-10:00     86115
          13:00-14:00     83138
          20:00-21:00     79009
          12:00-13:00     78184
          06:00-07:00     70910
          21:00-22:00     67474
          22:00-23:00     59477
          05:00-06:00     49345
          23:00-00:00     48230
          00:00-01:00     36413
          04:00-05:00     36100
          01:00-02:00     30496
          03:00-04:00     28200
          02:00-03:00     27001
          Name: ID_hora, dtype: int64
```

6. Conclusão

A ocorrência de acidentes nas estradas é um fenômeno cheio de particularidades envolvendo motivos desde falha dos indivíduos até problemas de nível macro como a omissão do governo em investimentos em infraestrutura. Detalhar cada um desses motivos eleva ainda mais o nível de complexidade já que podem ser enumerados problemas de ordem pessoal-psicológica no caso dos indivíduos à problemas político-econômicos do lado do governo. No entanto, baseando-se aos fatos e dados apresentados no presente estudo, felizmente foi possível verificar uma queda de vítimas fatais no histórico dos registros e também uma tendência de queda na previsão para os próximos anos.

É interessante destacar que de Minas Gerais vem se mantendo na dianteira de total de acidentes por UF, se consolidando como o estado com as estradas mais perigosas do país. Surpreende também como Santa Catarina, um dos menores estados em território físico, está na segunda posição de total de acidentes, estando à frente de unidades com território e população significativamente maiores como Bahia, São Paulo e Rio Grande do Sul. Ainda fazendo um comparativo entre as diferentes regiões do país, a região Sul, apesar de ser a menor em território e terceira menor em população, está com todos seus estados no top 5 de estradas mais perigosas. Isso ainda chama mais a atenção pelo fato de a região ter uma boa infraestrutura se comparada com demais localidades do Brasil.

Sobre as principais causas de acidente, a falta de atenção ocupa um primeiro lugar, seguido distante por velocidade incompatível e distanciamento incorreto. Isso nos faz pensar como a maioria dos acidentes são bastante "evitáveis" já que esses fatores citados estão sob total controle do condutor do veículo. Motivos que poderíamos dizer como "externos" como defeitos mecânicos ou problemas na pista só aparecem em quinto e nono lugares respectivamente.

A maioria dos acidentes ocorreu nos finais de semana, com sexta-feira, sábado e domingo estando nas primeiras posições. E os horários mais fatais estão entre as 20:30 e 03:30. Ou seja, maior número de acidentes com vítimas fatais nos finais de semana e no período da noite-madrugada pode estar associado à irresponsabilidade de motoristas que saem à lazer para se divertir e "esquecem" que conduzir uma máquina com o cansaço físico e mental coloca em risco vidas de pessoas inocentes.

Nesse sentido, uma sugestão para as autoridades é aumentar ainda mais a fiscalização e tornar mais rígidas iniciativas como a lei seca, principalmente nos dias e horários citados no parágrafo anterior. Ações de conscientização e melhores condições de transporte público também estão diretamente ligadas ao objetivo de mitigar o número de vítimas fatais no trânsito brasileiro, que infelizmente ainda permanece como um dos mais perigosos do mundo.

7. Links

Link 1 [Fonte de Dados Arquivo CSV] Disponível em:

https://github.com/KPxto/pucminas_tcc_mba/blob/master/data/acidentes2007-2020.tar.xz

Link 2 [Repositório Github] Disponível em:

https://github.com/KPxto/tcc_mba

Link 3 [Jupyter Notebook] Disponível em:

https://github.com/KPxto/pucminas_tcc_mba/blob/master/tratamento_tcc_completo.ipynb

REFERÊNCIAS:

G1. Disponível em:

<https://g1.globo.com/economia/noticia/por-que-o-brasil-depender-tanto-do-transporte-rodoviario.ghtml>

PANDAS

<https://pandas.pydata.org/>