Relatório de Análise de Dados – PicMoney

Objetivo: O propósito deste relatório é realizar uma análise completa sobre a base de dados da PicMoney para **entender o comportamento dos usuários e o desempenho do negócio**. O processo abrange desde a coleta e verificação da qualidade dos dados até a exploração e a geração de insights acionáveis.

Etapa 1: Coleta e Descrição dos Dados

Propósito

A primeira fase do projeto consiste em carregar os conjuntos de dados no ambiente de análise (Google Colab). Utilizamos a biblioteca pandas para importar os quatro arquivos CSV disponibilizados (Players, Transações, Pedestres e Lojas), que servem como a base para todo o estudo.

Descrição dos Dados

- **Players:** Contém informações cadastrais dos usuários do aplicativo, como idade, sexo e localização.
- **Transações:** Registra todas as transações de cupons capturados, incluindo valores, estabelecimentos e datas.
- **Pedestres:** Uma base simulada com dados de potenciais usuários em uma localização específica (Avenida Paulista).
- **Lojas:** Massa de teste com informações sobre os estabelecimentos parceiros e os valores dos cupons oferecidos.

Código 1: Configuração e Carregamento

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Configurações de visualização para os gráficos e tabelas
sns.set_style('whitegrid')
pd.set_option('display.float_format', lambda x: '%.2f' % x)

# Carregando os quatro datasets a partir dos arquivos CSV
try:
    df_players = pd.read_csv('PicMoney-Base_Cadastral_de_Players-10_000 linhas (1).csv',
delimiter=';')
    df_transacoes =
pd.read_csv('PicMoney-Base_de_Transa__es_-_Cupons_Capturados-100000 linhas (1).csv',
```

```
delimiter=';')
    df_pedestres = pd.read_csv('PicMoney-Base_Simulada_-_Pedestres_Av__Paulista-100000 linhas (1).csv', delimiter=';')
    df_lojas = pd.read_csv('PicMoney-Massa_de_Teste_com_Lojas_e_Valores-10000 linhas (1).csv', delimiter=';')

    print("Todos os arquivos foram carregados com sucesso!")
except FileNotFoundError as e:
    print(f"Erro: Arquivo não encontrado. Verifique o nome e o caminho do arquivo: {e}")
except Exception as e:
    print(f"Ocorreu um erro ao carregar os arquivos: {e}")
```

Etapa 2: Verificação da Qualidade e Limpeza dos Dados

Propósito

Esta é uma etapa crítica para garantir a confiabilidade da análise. O objetivo é identificar e corrigir inconsistências, erros de formatação e valores ausentes nos dados. Uma base de dados limpa e padronizada é fundamental para evitar conclusões equivocadas.

Processo de Limpeza

O script a seguir executa um tratamento completo, que inclui:

- **Padronização de Categorias:** Unificação dos nomes de estabelecimentos para evitar duplicidade.
- **Correção de Tipos de Dados:** Conversão de textos para formatos adequados (datas, números).
- **Limpeza de Campos:** Remoção de caracteres inválidos de colunas numéricas (ex: celular).
- **Tratamento de Valores Ausentes:** Preenchimento de campos vazios para não comprometer os cálculos.
- **Correção de Coordenadas:** Ajuste de um erro de formatação nos dados de latitude e longitude.

Código 2: Limpeza e Tratamento

```
import pandas as pd
import re
import os

# --- Mapeamentos para Padronização de Categorias ---
estabelecimento_para_categoria = {
    "Habib's": "Fast Food & Lanchonetes", 'Subway': "Fast Food & Lanchonetes", 'Burger King': "Fast Food & Lanchonetes",
```

```
    "McDonald's": "Fast Food & Lanchonetes", 'Açaí no Ponto': "Fast Food & Lanchonetes",
'Outback': "Restaurantes & Gastronomia",
    'Octavio Café': "Restaurantes & Gastronomia", 'Madero': "Restaurantes & Gastronomia",
'Café Cultura': "Restaurantes & Gastronomia",
    'Churrascaria Boi Preto': "Restaurantes & Gastronomia", 'Ráscal': "Restaurantes &
Gastronomia", 'Smart Fit': "Academias",
    'Selfit': "Academias", 'Just Run': "Academias", 'Forever 21': "Moda & Varejo", 'Renner': "Moda
& Varejo",
    'Riachuelo': "Moda & Varejo", 'Lojas Americanas': "Moda & Varejo", 'Havaianas': "Moda &
Varejo", 'Sabin': "Saúde & Bem-estar",
    'Lavoisier': "Saúde & Bem-estar", 'Fleury': "Saúde & Bem-estar", 'Clube Pinheiros': "Saúde &
Bem-estar",
    'Droga Raia': "Farmácias", 'Drogasil': "Farmácias", 'Drogaria São Paulo': "Farmácias", 'Extra':
"Supermercados & Mercados",
    'Carrefour Express': "Supermercados & Mercados", 'Pão de Açúcar': "Supermercados &
Mercados", 'Extra Mercado': "Supermercados & Mercados",
    'Starbucks': "Cafeterias", 'Ponto': "Lojas de Departamento & Eletrodomésticos", 'Casas
Bahia': "Lojas de Departamento & Eletrodomésticos",
    'Magazine Luiza': "Lojas de Departamento & Eletrodomésticos", 'Fast Shop': "Lojas de
Departamento & Eletrodomésticos",
    'Ponto Frio': "Lojas de Departamento & Eletrodomésticos", 'Sesc Paulista': "Cultura & Lazer",
'Sesc Carmo': "Cultura & Lazer",
    'Livraria Cultura': "Cultura & Lazer", 'Kalunga': "Papelaria e Escritório", 'Daiso Japan': "Lojas
de Variedades",
}

tipo_loja_pedestres_map = {
    'mercado express': 'Supermercados & Mercados', 'outros': 'Outros', 'restaurante':
'Restaurantes & Gastronomia',
    'esportivo': 'Artigos Esportivos', 'farmácia': 'Farmácias', 'eletrodoméstico': 'Lojas de
Departamento & Eletrodomésticos',
    'vestuário': 'Moda & Varejo', 'móveis': 'Móveis e Decoração', 'N/A': 'Não informado'
}


def corrigir_coordenadas(coord):
    """Corrige coordenadas com múltiplos pontos decimais."""
    if isinstance(coord, str):
        parts = coord.split('.')
        if len(parts) > 1:
            return parts[0] + '.' + ''.join(parts[1:])
    return coord
```

```
# --- Processo de Limpeza ---

# 1. Limpeza da Base Cadastral de Players
df_players['celular'] = df_players['celular'].str.replace(r'\D', '', regex=True)
df_players['data_nascimento'] = pd.to_datetime(df_players['data_nascimento'],
format='%d/%m/%Y', errors='coerce')
location_cols = ['cidade_trabalho', 'bairro_trabalho', 'cidade_escola', 'bairro_escola',
'categoria_frequentada']
for col in location_cols:
    df_players.loc[:, col] = df_players[col].fillna('Não informado')
df_players.to_csv('players_cleaned.csv', index=False)

# 2. Limpeza da Base de Transações
df_transacoes['celular'] = df_transacoes['celular'].str.replace(r'\D', '', regex=True)
df_transacoes['data'] = pd.to_datetime(df_transacoes['data'], format='%d/%m/%Y',
errors='coerce')
df_transacoes['hora'] = df_transacoes['hora'].astype(str)
df_transacoes['produto'].fillna('N/A', inplace=True)
df_transacoes['valor_cupom'] = pd.to_numeric(df_transacoes['valor_cupom'], errors='coerce')
df_transacoes['repasse_picmoney'] = pd.to_numeric(df_transacoes['repasse_picmoney'],
errors='coerce')
df_transacoes.dropna(subset=['valor_cupom', 'repasse_picmoney'], inplace=True)
df_transacoes['categoria_estabelecimento'] =
df_transacoes['nome_estabelecimento'].map(estabelecimento_para_categoria).fillna('Outros')
df_transacoes.to_csv('transacoes_cleaned.csv', index=False)

# 3. Limpeza da Base de Pedestres
df_pedestres['celular'] = df_pedestres['celular'].str.replace(r'\D', '', regex=True)
df_pedestres['latitude'] = df_pedestres['latitude'].astype(str).apply(corrigir_coordenadas)
df_pedestres['longitude'] = df_pedestres['longitude'].astype(str).apply(corrigir_coordenadas)
df_pedestres['latitude'] = pd.to_numeric(df_pedestres['latitude'], errors='coerce')
df_pedestres['longitude'] = pd.to_numeric(df_pedestres['longitude'], errors='coerce')
df_pedestres['data'] = pd.to_datetime(df_pedestres['data'], format='%d/%m/%Y',
errors='coerce')
df_pedestres['data_ultima_compra'] = pd.to_datetime(df_pedestres['data_ultima_compra'],
format='%d/%m/%Y', errors='coerce')
df_pedestres['possui_app_picmoney'] = df_pedestres['possui_app_picmoney'].apply(lambda
x: True if x == 'Sim' else False)
df_pedestres['ultimo_tipo_cupom'].fillna('N/A', inplace=True)
df_pedestres['ultimo_valor_capturado'].fillna(0, inplace=True)
df_pedestres['ultimo_tipo_loja'].fillna('N/A', inplace=True)
df_pedestres['ultimo_tipo_loja'] =
df_pedestres['ultimo_tipo_loja'].map(tipo_loja_pedestres_map).fillna('Não informado')
```

```
df_pedestres.to_csv('pedestres_cleaned.csv', index=False)

# 4. Limpeza da Massa de Teste de Lojas
df_lojas['numero_celular'] = df_lojas['numero_celular'].str.replace(r'\D', '', regex=True)
df_lojas['latitude'] = df_lojas['latitude'].astype(str).apply(corrigir_coordenadas)
df_lojas['longitude'] = df_lojas['longitude'].astype(str).apply(corrigir_coordenadas)
df_lojas['latitude'] = pd.to_numeric(df_lojas['latitude'], errors='coerce')
df_lojas['longitude'] = pd.to_numeric(df_lojas['longitude'], errors='coerce')
df_lojas['data_captura'] = pd.to_datetime(df_lojas['data_captura'], format='%d/%m/%Y',
errors='coerce')
df_lojas['valor_compra'] = pd.to_numeric(df_lojas['valor_compra'], errors='coerce')
df_lojas['valor_cupom'] = pd.to_numeric(df_lojas['valor_cupom'], errors='coerce')
df_lojas.dropna(subset=['valor_compra', 'valor_cupom'], inplace=True)
df_lojas.to_csv('lojas_cleaned.csv', index=False)

print("Processo de limpeza concluído e arquivos *_cleaned.csv gerados!")
```

Etapa 3: Exploração dos Dados e Análise de KPIs

Propósito

Com os dados limpos, iniciamos a Análise Exploratória de Dados (EDA). O objetivo é investigar os dados para descobrir padrões, identificar anomalias, testar hipóteses e extrair os principais indicadores de desempenho (KPIs) através de resumos estatísticos e visualizações gráficas.

Código 3: Análise Exploratória de Dados (EDA)

```
# Carregando os dataframes limpos para a análise
df_players_cleaned = pd.read_csv('players_cleaned.csv')
df_transacoes_cleaned = pd.read_csv('transacoes_cleaned.csv')

print("--- Análise Exploratória da Base de Players ---")
print("\nDistribuição por sexo:")
print(df_players_cleaned['sexo'].value_counts())

plt.figure(figsize=(8, 6))
sns.histplot(df_players_cleaned['idade'], bins=20, kde=True)
plt.title('Distribuição de Idade dos Usuários')
plt.xlabel('Idade')
plt.ylabel('Contagem')
plt.savefig('distribuicao_idade_usuarios.png')
plt.show()
```

```
print("\n--- Análise Exploratória da Base de Transações ---")
print("\nTop 10 Categorias de Estabelecimento por Transação:")
print(df_transacoes_cleaned['categoria_estabelecimento'].value_counts().head(10))

# Gráfico de barras das top 10 categorias
plt.figure(figsize=(12, 8))
sns.countplot(y='categoria_estabelecimento', data=df_transacoes_cleaned, order =
df_transacoes_cleaned['categoria_estabelecimento'].value_counts().index[:10])
plt.title('Top 10 Categorias de Estabelecimento Mais Populares')
plt.xlabel('Número de Transações')
plt.ylabel('Categoria')
plt.savefig('top_10_categorias.png')
plt.show()

print("\nEstatísticas do Repasse para a PicMoney:")
print(df_transacoes_cleaned['repasse_picmoney'].describe())
```

Código 4: Análise Aprofundada e de KPIs

```
# Carregar os dataframes limpos
df_players = pd.read_csv('players_cleaned.csv')
df_transacoes = pd.read_csv('transacoes_cleaned.csv')

# 1.1: Distribuição de usuários por bairro em São Paulo
plt.figure(figsize=(10, 8))
top_10_bairros = df_players['bairro_residencial'].value_counts().nlargest(10)
sns.barplot(x=top_10_bairros.values, y=top_10_bairros.index, palette='viridis', orient='h')
plt.title('Top 10 Bairros com Mais Usuários Cadastrados')
plt.xlabel('Número de Usuários')
plt.ylabel('Bairro')
plt.savefig('analise_bairros_usuarios.png')

# 1.2: Distribuição de usuários por faixa etária
bins = [15, 25, 35, 45, 55, 65, 100]
labels = ['15-25', '26-35', '36-45', '46-55', '56-65', '65+']
df_players['faixa_etaria'] = pd.cut(df_players['idade'], bins=bins, labels=labels, right=False)
plt.figure(figsize=(10, 6))
faixa_etaria_counts = df_players['faixa_etaria'].value_counts().sort_index()
sns.barplot(x=faixa_etaria_counts.index, y=faixa_etaria_counts.values, palette='magma')
plt.title('Distribuição de Usuários por Faixa Etária')
```

```
plt.xlabel('Faixa Etária')
plt.ylabel('Número de Usuários')
plt.savefig('analise_faixa_etaria.png')

# 2.2: Análise de transações por dia da semana
df_transacoes['data'] = pd.to_datetime(df_transacoes['data'])
df_transacoes['dia_da_semana'] = df_transacoes['data'].dt.day_name()
dias_ordem = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
transacoes_por_dia = df_transacoes['dia_da_semana'].value_counts().reindex(dias_ordem)
plt.figure(figsize=(10, 6))
sns.barplot(x=transacoes_por_dia.index, y=transacoes_por_dia.values, palette='cividis')
plt.title('Número de Transações por Dia da Semana')
plt.xlabel('Dia da Semana')
plt.ylabel('Número de Transações')
plt.savefig('analise_transacoes_dia_semana.png')

# 4.1: KPI - Evolução da Receita x Usuários Ativos (Mensal)
df_transacoes['mes_ano'] = df_transacoes['data'].dt.to_period('M')
receita_mensal = df_transacoes.groupby('mes_ano')['repasse_picmoney'].sum().reset_index()
receita_mensal['mes_ano'] = receita_mensal['mes_ano'].astype(str)
usuarios_ativos_mensal = df_transacoes.groupby('mes_ano')['celular'].nunique().reset_index()
usuarios_ativos_mensal.rename(columns={'celular': 'usuarios_ativos'}, inplace=True)
usuarios_ativos_mensal['mes_ano'] = usuarios_ativos_mensal['mes_ano'].astype(str)
df_evolucao = pd.merge(receita_mensal, usuarios_ativos_mensal, on='mes_ano')

fig, ax1 = plt.subplots(figsize=(14, 7))
ax1.set_xlabel('Mês/Ano')
ax1.set_ylabel('Receita Total (R$)', color='tab:red')
ax1.bar(df_evolucao['mes_ano'], df_evolucao['repasse_picmoney'], color='tab:red', alpha=0.6,
label='Receita (R$)')
ax1.tick_params(axis='y', labelcolor='tab:red')

ax2 = ax1.twinx()
ax2.set_ylabel('Usuários Ativos', color='tab:blue')
ax2.plot(df_evolucao['mes_ano'], df_evolucao['usuarios_ativos'], color='tab:blue', marker='o',
label='Usuários Ativos')
ax2.tick_params(axis='y', labelcolor='tab:blue')

plt.title('Evolução Mensal da Receita Total e Usuários Ativos')
fig.tight_layout()
plt.savefig('analise_evolucao_receita_usuarios.png')

plt.show() # Mostra todos os gráficos gerados
```

Etapa 4: Análise dos Resultados e Recomendações

Resultados da Análise

- **Perfil dos Usuários:**
  - **Faixa Etária:** A base de usuários é predominantemente jovem, com forte concentração na faixa de **26 a 35 anos**, seguida pelo grupo de 36 a 45 anos.
  - **Localização:** Existe uma alta densidade de usuários em bairros centrais de São Paulo, com destaque para **Sé, República, Bela Vista e Consolação**.
  - **Poder de Compra:** Embora o grupo de 26 a 35 anos seja o mais numeroso, a faixa etária de **36 a 45 anos** representa o maior volume de gastos totais, indicando um ticket médio mais alto.
- **Padrões de Transação:**
  - **Categorias Mais Populares: Restaurantes e Gastronomia** é a categoria líder, dominando tanto em volume de transações quanto em receita gerada para a PicMoney, seguida por Saúde/Bem-estar e Farmácias.
  - **Pico de Uso:** A atividade no aplicativo é maior nos finais de semana, especialmente às **sextas-feiras e sábados**.
  - **Valor dos Cupons:** A maioria dos cupons utilizados possui valor baixo, o que sugere um uso frequente para compras cotidianas e de pequeno valor.
- **Desempenho do Negócio (KPIs):**
  - **Receita e Usuários Ativos:** A análise mensal mostra uma forte correlação positiva entre o aumento do número de usuários ativos e o crescimento da receita. Isso valida a eficácia das estratégias de aquisição e engajamento.
  - **Performance por Região:** Os bairros com a maior concentração de usuários (Sé, República, Bela Vista) são também os que mais geram receita, confirmando que a presença de uma base de usuários local impulsiona o consumo.

Conclusões e Recomendações

- **Conclusões:**
  1. O público-alvo principal da PicMoney é o **jovem adulto (26-45 anos)** residente ou trabalhador de áreas centrais de São Paulo.
  2. O setor de **alimentação** é o principal motor de receita e engajamento da plataforma.
  3. O comportamento de uso concentrado no **fim de semana** revela uma oportunidade para estimular a atividade em dias de menor movimento.
- **Recomendações Estratégicas:**
  1. **Marketing Focado:** Direcionar campanhas de aquisição e anúncios geolocalizados para os bairros de maior performance (Sé, República, Bela Vista) e perfis demográficos semelhantes em outras áreas.
  2. **Expansão de Parcerias:** Fortalecer e expandir parcerias com restaurantes, bares e cafés. Criar campanhas de cashback mais agressivas para a faixa etária de 36-45 anos, que possui maior poder de compra.

3. **Incentivo ao Engajamento:** Lançar promoções e cupons especiais para os dias de menor movimento (de segunda a quarta-feira) para criar um hábito de uso contínuo e aumentar o volume de transações na semana.
4. **Melhoria da Coleta de Dados:** Implementar validações no momento do cadastro e da transação para garantir maior qualidade e consistência dos dados, o que facilitará análises futuras mais complexas e precisas.