```
!pip install pandas numpy scikit-learn tabulate

#PREDICAO DEFINITIVA
import pandas as pd
import ast
import numpy as np
from IPython.display import display
import re
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
from math import radians, cos, sin, asin, sqrt

# Carregar a planilha
df = pd.read_csv('dados.csv')

# Filtrar dados nulos em 'Estimatives'
df = df[df['Estimatives'].notnull()].copy()

# Verificar se 'Estimatives' está em string ou dict
if isinstance(df['Estimatives'].iloc[0], str):
    df['Estimatives'] = df['Estimatives'].apply(ast.literal_eval)

# Extração dos valores de 'Estimatives' para novas colunas
df['valor_uber'] = df['Estimatives'].apply(lambda x: x.get('UberX') if
isinstance(x, dict) else np.nan)
df['valor_pop99'] = df['Estimatives'].apply(lambda x: x.get('pop99')
if isinstance(x, dict) else np.nan)
df['valor_poupa99'] = df['Estimatives'].apply(lambda x:
x.get('poupa99') if isinstance(x, dict) else np.nan)
df['valor_Comfort'] = df['Estimatives'].apply(lambda x:
x.get('Comfort') if isinstance(x, dict) else np.nan)

# Remover linhas com valores nulos em qualquer uma das colunas de
valor
df = df.dropna(subset=['valor_uber', 'valor_pop99', 'valor_poupa99',
'valor_Comfort'])

# Mapear os dias da semana
dias_semana = {
    'Monday': 0,
    'Tuesday': 1,
    'Wednesday': 2,
    'Thursday': 3,
    'Friday': 4,
    'Saturday': 5,
    'Sunday': 6
}
df['dia_semana'] = df['Dia'].map(dias_semana)
```

```
# Converter horário e extrair a hora
df['Horario'] = pd.to_datetime(df['Horario'], format='%H:%M:%S',
errors='coerce')
df['hora'] = df['Horario'].dt.hour

def safe_convert_to_lat_long(value):
    # Remover pontos
    value = str(value).replace('.', '')

    # Verificar se o valor não é vazio e se pode ser convertido
    if value and value.strip() != '':
        numeros_sem_ponto = value.replace('.', '')
          # Coloca o último ponto no lugar certo (antes dos 3 últimos
dígitos)
        return float(re.sub(r'(\d+)(\d{3})$', r'\1.\2',
numeros_sem_ponto))
    else:
        # Retornar um valor padrão, como 0, caso a célula esteja vazia
ou inválida
        return '0.000000'

# Aplicar a função de conversão para todas as colunas de latitude e
longitude
df.loc[:, 'LatOrigem'] =
df['LatOrigem'].apply(safe_convert_to_lat_long)
df.loc[:, 'LongOrigem'] =
df['LongOrigem'].apply(safe_convert_to_lat_long)
df.loc[:, 'LatDestino'] =
df['LatDestino'].apply(safe_convert_to_lat_long)
df.loc[:, 'LongDestino'] =
df['LongDestino'].apply(safe_convert_to_lat_long)
print(display(df.head(10)))

# Selecionar colunas de entrada (X) e saída (y)
X = df[['Distancia_KM', 'hora', 'dia_semana', 'LatOrigem',
'LongOrigem', 'LatDestino', 'LongDestino', 'valor_Comfort',
'valor_poupa99', 'valor_pop99']]
y = df['valor_uber']

# Dividir os dados em treinamento e teste
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2)

# Treinar o modelo
modelo = RandomForestRegressor()
modelo.fit(X_train, y_train)

# Fazer previsões
y_pred = modelo.predict(X_test)
```

```python
# Calcular métricas
mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

# Exibir resultados
print(f"MAE (Erro Absoluto Médio): {mae:.2f}")
print(f"RMSE (Raiz do Erro Quadrático Médio): {rmse:.2f}")
print(f"R² (Coeficiente de Determinação): {r2:.4f}")
```

```
     RideID              LatOrigem             LongOrigem
CityOrigem  \
0   1183201            -274919.788 -48528287999999.898438         NaN

1   1183202            -198495.799 -44019915999999.898438         NaN

3   1183204            -109198.019 -37077441799999.898438         NaN

6   1183208            -198495.393            -440199.286         NaN

9   1183211 -23743931699999.898438            -466083.953         NaN

12  1183214 -20559277599999.898438            -546231.586         NaN

19  1183221             -23708.786            -465267.639         NaN

20  1183223            -160386.142            -480188.949   Brasília

21  1183224            -227786.199            -433193.931         NaN

22  1183225            -237088.701             -46526.664         NaN


    LatDestino           LongDestino          CityDestino
Distancia_KM  \
0  -274371.486 -4839824309999.990234                  NaN
18.19
1   -19936.899           -439401.603       Belo Horizonte
17.15
3  -109071.288           -370877.194              Aracaju
3.22
6   -19936.899           -439401.603                  NaN
17.15
9  -236896.422 -4659155519999.990234              Diadema
14.78
12 -204786.441           -546261.535                  NaN
13.22
19 -237047.254           -466175.221              Diadema
13.02
20  -16068.436           -479852.555  Valparaíso de Goiás
```

```
10.19
21 -227923.134                -433.0                   NaN
3.93
22 -237047.254           -466175.221                   NaN
13.03

                                             Estimatives  \
0   {'Flash': 31.5, 'Comfort': 33.5, 'poupa99': 26...
1   {'Moto': 25.5, 'Comfort': 44.0, 'UberFlash': 4...
3   {'poupa99': 7.88, 'pop99': 7.88, 'turbo-taxi':...
6   {'poupa99': 71.3, 'pop99': 71.3, 'turbo-taxi':...
9   {'poupa99': 40.43, 'pop99': 40.43, 'turbo-taxi...
12  {'Flash': 38.5, 'poupa99': 18.1, 'turbo-taxi':...
19  {'poupa99': 31.98, 'turbo-taxi': 57.46, 'regul...
20  {'poupa99': 17.74, 'pop99': 17.74, 'turbo-taxi...
21  {'poupa99': 10.76, 'pop99': 10.76, 'turbo-taxi...
22  {'UberX': 48.5, 'poupa99': 33.84, 'pop99': 33....

                          Updated      Dia              Horario
valor_uber  \
0   2021-08-17 10:10:06.735871100  Tuesday 1900-01-01 10:10:06
31.5
1   2021-08-17 10:10:19.660182900  Tuesday 1900-01-01 10:10:19
42.0
3   2021-08-17 10:10:37.528390200  Tuesday 1900-01-01 10:10:37
7.5
6   2021-08-17 10:11:09.415673800  Tuesday 1900-01-01 10:11:09
42.0
9   2021-08-17 10:11:25.021341600  Tuesday 1900-01-01 10:11:25
27.5
12  2021-08-17 10:11:36.487865700  Tuesday 1900-01-01 10:11:36
38.5
19  2021-08-17 10:13:14.316337800  Tuesday 1900-01-01 10:13:14
49.5
20  2021-08-17 10:13:49.998431700  Tuesday 1900-01-01 10:13:49
16.0
21  2021-08-17 10:13:49.951794400  Tuesday 1900-01-01 10:13:49
10.5
22  2021-08-17 10:13:53.799629800  Tuesday 1900-01-01 10:13:53
48.5

    valor_pop99  valor_poupa99  valor_Comfort  dia_semana  hora
0         29.14          26.91           33.5           1    10
1         74.80          74.80           44.0           1    10
3          7.88           7.88            9.5           1    10
6         71.30          71.30           44.0           1    10
9         40.43          40.43           40.0           1    10
12        18.10          18.10           38.0           1    10
19        31.98          31.98           74.5           1    10
20        17.74          17.74           22.0           1    10
```

```
21         10.76           10.76            13.5            1    10
22         33.84           33.84            73.5            1    10

None
MAE (Erro Absoluto Médio): 1.95
RMSE (Raiz do Erro Quadrático Médio): 4.02
R² (Coeficiente de Determinação): 0.9651
```