Relatório 2 – Aprendizado de Máquina por Regressão

EEL891 – 2022.1 – Prof. Heraldo Almeida

Aluna: Martina Marques Jardim

DRE: 121078124

Id Kaggle: 10340340

1 – Introdução:

O seguinte relatório tem como intuito descrever o código utilizado no segundo trabalho da disciplina, cujo objetivo é estimar o preço de um imóvel na cidade do Recife. Serão relatados o pré-processamento e tratamento de dados, bem como a implementação do método regressor KNN e a verificação da eficiência desse método.

2 – Pré-processamento e tratamento de dados:

O *dataframe* “dadosTreino” refere-se ao conteúdo do documento *.csv* “conjunto_de_treinamento.csv” e o *dataframe* “dadosTeste” refere-se ao documento *.csv* “conjunto_de_teste.csv”.

```
# Será utilizado o regressor KNN

from sklearn.preprocessing import LabelBinarizer
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors     import KNeighborsRegressor
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from scipy.stats import pearsonr
import pandas as pd
import numpy as np
import math

dadosTreino = pd.read_csv('conjunto_de_treinamento.csv')
dadosTeste = pd.read_csv('conjunto_de_teste.csv')
```

Inicialmente, é analisada a cardinalidade das variáveis categóricas do *dataframe* “dadosTreino”. Encontra-se quatro variáveis categóricas, em que ‘tipo’ possui quatro categorias, ‘tipo_vendedor’ tem duas categorias, ‘bairro’ possui 66 categorias, ‘diferenciais’ tem 83 categorias.

A variável ‘tipo_vendedor’ será binarizada, ou seja, suas duas categorias do tipo *string* serão codificadas em 1 e 0. Além disso, será aplicado o *One-Hot Encoding* na variável ‘tipo’, que se refere ao tipo de imóvel.

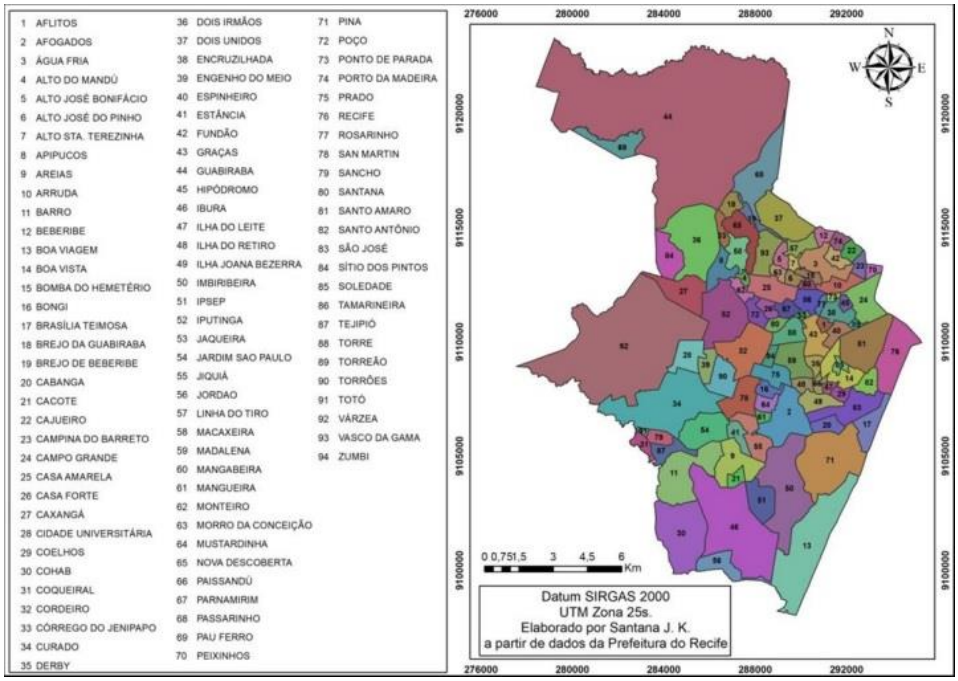
```
# Binarizando
binarizador = LabelBinarizer()

for contador in ['tipo_vendedor']:
    dadosTreino [contador] = binarizador.fit_transform(dadosTreino[contador])
    dadosTeste [contador] = binarizador.fit_transform(dadosTeste[contador])

# One Hot Encoding #
dadosTreino = pd.get_dummies(dadosTreino, columns = ['tipo'])
dadosTeste = pd.get_dummies(dadosTeste, columns = ['tipo'])
```

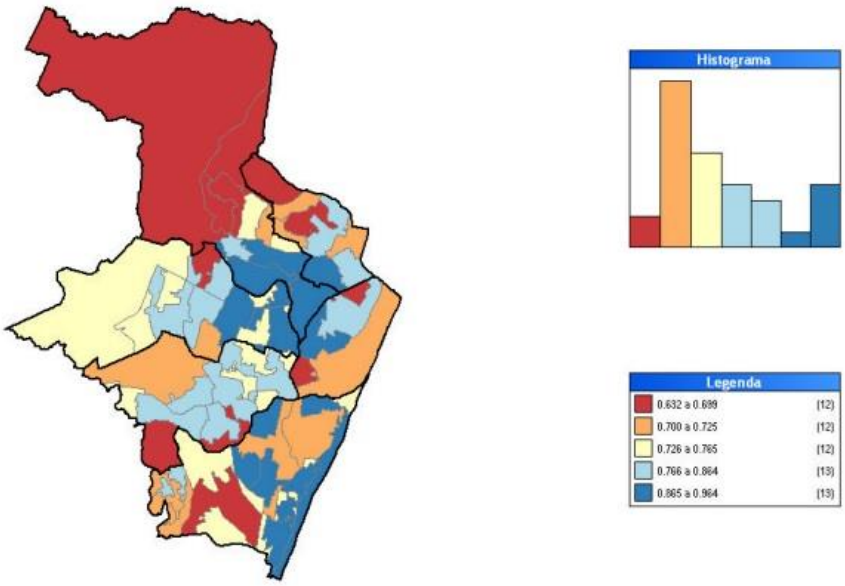
É necessário também retirar os *outliers* do *dataframe* de treinamento. Esse conjunto alguns valores para o preço dos apartamentos que foram digitados incorretamente, tornando-se muito altos ou muito baixos. É necessário retirar essas informações pois isso dificulta o aprendizado e deturpa o resultado final.

Em seguida, serão analisadas as variáveis ‘bairro’ e ‘diferenciais’. Para codificar as categorias da variável ‘bairro’, os nomes dos bairros de Recife foram substituídos pelos seus respectivos IDHs. Essas informações foram coletadas a partir dos seguintes mapas:



Mapa 1 – Localização geográfica dos mapas de Recife

FIG. 1 - RECIFE - Índice de Desenvolvimento Humano Municipal, por Unidades de Desenvolvimento Humano - 2000



Fonte: RECIFE.Prefeitura; PNUD. Atlas do Desenvolvimento Humano no Recife. 2005.

Mapa 2 – IDH de Recife por localização geográfica

Verificou-se, no mapa 1, a localização dos bairros informados nos *dataframes,* e, a partir de sua localização geográfica, determina-se a faixa a qual seu IDH pertence a partir do mapa 2. Como o mapa 2 não informa o valor exato do IDH de cada bairro, foi considerado o valor médio de cada um dos intervalos.

```
dadosTreino = dadosTreino[(dadosTreino['preco']>= 50000) & (dadosTreino['preco']<= 5000000

# Classificando os bairros com base em seu IDH, com base em dados de 2005
# IDH Muito Baixo = 0.666
# IDH Baixo = 0.7125
# IDH Médio = 0.7455
# IDH Alto = 0.815
# IDH Muito Alto = 0.9145

bairros = {'Imbiribeira' : 0.815, 'Casa Amarela' : 0.9145, 'Encruzilhada' : 0.9145,
           'Boa Viagem' : 0.666, 'Rosarinho' : 0.9145, 'Boa Vista' : 0.9145,
           'Espinheiro' : 0.9145, 'Tamarineira' : 0.9145, 'Gracas' : 0.9145,
           'Madalena' : 0.9145, 'Parnamirim' : 0.9145, 'S Jose' : 0.7125,
           'Setubal' : 0.9145, 'Arruda' : 0.9145, 'Pina' : 0.7125, 'Beira Rio' : 0.7125,
           'Caxanga' : 0.7455, 'Casa Forte' : 0.7455, 'Prado' : 0.7455,
           'Iputinga' : 0.7125, 'Campo Grande' : 0.7455, 'Dois Irmaos': 0.666,
           'Torreao' : 0.9145, 'Ilha do Retiro' : 0.7455, 'Areias' : 0.815,
           'Varzea' : 0.7455, 'Cordeiro' : 0.9145, 'Santana': 0.9145, 'Torre' : 0.9145,
           'Barro' : 0.7125, 'Poco da Panela' : 0.9145, 'Ipsep' : 0.9145,
           'Apicupos' : 0.666, 'Aflitos' : 0.9145, 'Poco' : 0.9145, 'Apipucos' : 0.666,
           'Derby' : 0.9145, 'Cid Universitaria' : 0.7455, 'Bongi' : 0.7455, 'Jaqueira' :
           'Sto Amaro' : 0.7455, 'Tejipio' : 0.815, 'Recife' : 0.7125,
           'Monteiro' : 0.815, 'Macaxeira' : 0.666, 'Sancho' : 0.7455,
           'Afogados' : 0.7455, 'San Martin' : 0.815, 'Cajueiro' : 0.815,
           'Hipodromo' : 0.9145, 'Guabiraba' : 0.666, 'Engenho do Meio' : 0.815,
           'Piedade' : 0.7455 , 'Jd S Paulo' : 0.815, 'Lagoa do Araca' : 0.815 ,
           'Ponto de Parada' : 0.9145, 'Ilha do Leite' : 0.9145, 'Estancia' : 0.815,
           'Paissandu' : 0.9145, 'Zumbi' : 0.9145, 'Agua Fria' : 0.7455,
           'Benfica' : 0.9145, 'Soledade' : 0.815, 'Centro' : 0.7455 , 'Sto Antonio' : 0.7
           'Coelhos' : 0.9145, 'Cohab' : 0.7125, 'Ibura' : 0.7125, 'Beberibe' : 0.7125, 'F

dadosTreino = dadosTreino.replace(bairros)
dadosTeste = dadosTeste.replace(bairros)
```

Em seguida, deve-se analisar a variável ‘diferenciais’. Essa possui 83 categorias.

```
diferenciais = {'campo de futebol e copa':'futebol e outros',
 'campo de futebol e esquina':'futebol e outros',
 'campo de futebol e estacionamento visitantes':'futebol e outros',
 'campo de futebol e playground':'futebol e outros',
 'campo de futebol e quadra poliesportiva':'futebol e outros',
 'campo de futebol e salao de festas':'futebol e outros',
 'campo de futebol e sala de ginastica':'futebol e outros',
 'children care':'apenas children care',
 'children care e playground':'children care e outros',
 'churrasqueira':'apenas churrasco',
 'churrasqueira e campo de futebol':'churrasco e outros',
 'churrasqueira e copa':'churrasco e outros',
 'churrasqueira e esquina':'churrasco e outros',
 'churrasqueira e estacionamento visitantes':'churrasco e outros',
 'churrasqueira e frente para o mar':'churrasco e outros',
 'churrasqueira e playground':'churrasco e outros',
 'churrasqueira e sala de ginastica':'churrasco e outros',
 'churrasqueira e salao de festas':'churrasco e outros',
 'churrasqueira e sauna':'churrasco e outros',
 'churrasqueira e children care':'churrasco e outros',
 'copa':'apenas copa',
 'copa e esquina':'copa e outros',
 'copa e estacionamento visitantes':'copa e outros',
 'copa e playground':'copa e outros',
 'copa e quadra poliesportiva':'copa e outros',
 'copa e sala de ginastica':'copa e outros',
 'copa e salao de festas':'copa e outros',
 'copa e hidromassagem':'copa e outros',
 'esquina':'apenas esquina',
 'esquina e estacionamento visitantes':'esquina e outros',
 'esquina e playground':'esquina e outros',
 'esquina e quadra poliesportiva':'esquina e outros',
 'esquina e sala de ginastica':'esquina e outros',
 'esquina e salao de festas':'esquina e outros',
```

```
'estacionamento visitantes':'apenas estacionamento visitantes',
'estacionamento visitantes e playground':'estacionamento visitantes e outros',
'estacionamento visitantes e sala de ginastica':'estacionamento visitantes e outros',
'estacionamento visitantes e salao de festas':'estacionamento visitantes e outros',
'estacionamento visitantes e hidromassagem':'estacionamento visitantes e outros',
'estacionamento visitantes e salao de jogos':'estacionamento visitantes e outros',
'frente para o mar':'apenas frente para o mar',
'frente para o mar e campo de futebol':'frente para o mar e outros',
'frente para o mar e copa':'frente para o mar e outros',
'frente para o mar e esquina':'frente para o mar e outros',
'frente para o mar e playground':'frente para o mar e outros',
'frente para o mar e quadra poliesportiva':'frente para o mar e outros',
'frente para o mar e salao de festas':'frente para o mar e outros',
'frente para o mar e children care':'frente para o mar e outros',
'frente para o mar e hidromassagem':'frente para o mar e outros',
'nenhum':'nenhum',
'piscina':'apenas piscina',
'piscina e campo de futebol':'piscina e outros',
'piscina e children care':'piscina e outros',
'piscina e churrasqueira':'piscina e outros',
'piscina e copa':'piscina e outros',
'piscina e esquina':'piscina e outros',
'piscina e estacionamento visitantes':'piscina e outros',
'piscina e frente para o mar':'piscina e outros',
'piscina e hidromassagem':'piscina e outros',
'piscina e playground':'piscina e outros',
'piscina e quadra de squash':'piscina e outros',
'piscina e quadra poliesportiva':'piscina e outros',
'piscina e sala de ginastica':'piscina e outros',
'piscina e salao de festas':'piscina e outros',
'piscina e salao de jogos':'piscina e outros',
'piscina e sauna':'piscina e outros',
'playground':'apenas playground',
'playground e quadra poliesportiva':'playground e outros',
'playground e sala de ginastica':'playground e outros',
```

```
  'piscina e salao de festas':'piscina e outros',
  'piscina e salao de jogos':'piscina e outros',
  'piscina e sauna':'piscina e outros',
  'playground':'apenas playground',
  'playground e quadra poliesportiva':'playground e outros',
  'playground e sala de ginastica':'playground e outros',
  'playground e salao de festas':'playground e outros',
  'playground e salao de jogos':'playground e outros',
  'quadra poliesportiva':'apenas quadra poliesportiva',
  'quadra poliesportiva e salao de festas':'quadra e outros',
  'sala de ginastica':'apenas sala de ginastica',
  'sala de ginastica e salao de festas':'ginastica e outros',
  'sala de ginastica e salao de jogos':'ginastica e outros',
  'salao de festas':'apenas salao de festas',
  'salao de festas e salao de jogos':'festa e outros',
  'salao de festas e vestiario':'festa e outros',
  'salao de jogos':'apenas salao de jogos',
  'sauna':'apenas sauna',
  'sauna e campo de futebol':'sauna e outros',
  'sauna e copa':'sauna e outros',
  'sauna e esquina':'sauna e outros',
  'sauna e frente para o mar':'sauna e outros',
  'sauna e playground':'sauna e outros',
  'sauna e quadra poliesportiva':'sauna e outros',
  'sauna e sala de ginastica':'sauna e outros',
  'sauna e salao de festas':'sauna e outros',
  'vestiario':'apenas vestiario',
  'futebol e sala de ginastica':'futebol e outros',
  'mar e hidromassagem':'frente para o mar e outros',
  'hidromassagem e salao de festas':'festa e outros'}

dadosTreino = dadosTreino.replace(diferenciais)
dadosTeste = dadosTeste.replace(diferenciais)
```

Os diferenciais foram codificados de forma que se encaixassem em categorias que destacassem, principalmente, a presença de piscinas, saunas, vista para o mar, sala de ginástica, *playground,* salão de festas, salão de jogos, estacionamento, copa, esquina, churrasqueira e *children care.*

Após a substituição, é aplicado o *One-Hot Encoding,* com o intuito de criar novas variáveis binárias que representassem as categorias substituídas. Com a criação dessas novas variáveis, outras variáveis que representam as mesmas informações devem ser excluídas do *dataframe.*

```
dadosTreino = dadosTreino.replace(diferenciais)
dadosTeste = dadosTeste.replace(diferenciais)

#One hot encoding
dadosTreino = pd.get_dummies(dadosTreino,columns = ['diferenciais'])
dadosTeste = pd.get_dummies(dadosTeste,columns = ['diferenciais'])

#Dropando repetições
dadosTreino = dadosTreino.drop (['Id', 'churrasqueira', 'piscina', 'playground', 'sauna',
                                 'quadra', 's_festas', 's_jogos', 's_ginastica', 'vista_mar'], axis = 1)
dadosTeste = dadosTeste.drop (['Id', 'churrasqueira', 'piscina', 'playground', 'sauna',
                               'quadra', 's_festas', 's_jogos', 's_ginastica', 'vista_mar'], axis = 1)

dadosTreino = dadosTreino.replace(diferenciais)
dadosTeste = dadosTeste.replace(diferenciais)
```

A fim de verificar a eficiência e a importância de cada uma das variáveis nos *dataframes,* serão calculados os coeficientes de Pearson de cada uma.

```
colunasNovas = dadosTreino.columns
for col in colunasNovas:
  print('%10s = %6.3f' % (col, pearsonr(dadosTreino[col],dadosTreino['preco'])[0]))
```

Obteve-se os seguintes valores:

```
    bairro = -0.219
tipo_vendedor = -0.029
   quartos =  0.589
    suites =  0.710
     vagas =  0.559
 area_util =  0.605
area_extra =  0.041
estacionamento = -0.046
     preco =  1.000
tipo_Apartamento = -0.116
 tipo_Casa =  0.122
 tipo_Loft = -0.016
tipo_Quitinete = -0.019
diferenciais_apenas children care = -0.009
diferenciais_apenas churrasco = -0.016
diferenciais_apenas copa =  0.007
diferenciais_apenas esquina = -0.026
diferenciais_apenas estacionamento visitantes = -0.049
diferenciais_apenas frente para o mar =  0.025
diferenciais_apenas piscina = -0.023
diferenciais_apenas playground = -0.035
diferenciais_apenas quadra poliesportiva = -0.010
diferenciais_apenas sala de ginastica =  0.085
diferenciais_apenas salao de festas = -0.046
diferenciais_apenas salao de jogos =  0.014
diferenciais_apenas sauna = -0.009
```

```
diferenciais_apenas salao de jogos =  0.014
diferenciais_apenas sauna = -0.009
diferenciais_apenas vestiario =  0.007
diferenciais_children care e outros = -0.014
diferenciais_churrasco e outros =  0.037
diferenciais_copa e outros =  0.063
diferenciais_esquina e outros = -0.001
diferenciais_estacionamento visitantes e outros = -0.029
diferenciais_festa e outros = -0.013
diferenciais_frente para o mar e outros =  0.119
diferenciais_futebol e outros = -0.015
diferenciais_ginastica e outros = -0.003
diferenciais_nenhum = -0.115
diferenciais_piscina e outros =  0.091
diferenciais_playground e outros = -0.037
diferenciais_quadra e outros =  0.015
diferenciais_sauna e outros =  0.027
```

A fim de melhorar a eficiência do modelo preditivo, foram excluídas as variáveis com uma ou duas casas decimais acima ou abaixo de zero.

```
matrizXTreino = dadosTreino.drop(['preco','tipo_Quitinete','diferenciais_apenas children care',
                                  'area_extra', 'estacionamento', 'area_extra', 'estacionamento',
                                  'tipo_Loft','diferenciais_children care e outros',
                                  'diferenciais_apenas churrasco', 'diferenciais_churrasco e outros',
                                  'diferenciais_apenas copa', 'diferenciais_copa e outros', 'diferenciais_apenas esquina',
                                  'diferenciais_esquina e outros', 'diferenciais_apenas estacionamento visitantes',
                                  'diferenciais_estacionamento visitantes e outros', 'diferenciais_festa e outros',
                                  'diferenciais_apenas frente para o mar', 'diferenciais_futebol e outros',
                                  'diferenciais_ginastica e outros', 'diferenciais_apenas piscina',
                                  'diferenciais_piscina e outros', 'diferenciais_apenas playground',
                                  'diferenciais_playground e outros', 'diferenciais_apenas quadra poliesportiva',
                                  'diferenciais_quadra e outros', 'diferenciais_apenas sala de ginastica',
                                  'diferenciais_apenas salao de festas', 'diferenciais_apenas salao de jogos',
                                  'diferenciais_apenas sauna', 'diferenciais_sauna e outros', 'diferenciais_apenas vestiario'
matrizYTreino = dadosTreino['preco']
matrizXTeste = dadosTeste.drop(['area_extra', 'estacionamento', 'area_extra',
                                'tipo_Loft', 'diferenciais_apenas churrasco', 'diferenciais_churrasco e outros',
                                'diferenciais_apenas copa', 'diferenciais_copa e outros', 'diferenciais_apenas esquina',
                                'diferenciais_esquina e outros', 'diferenciais_apenas estacionamento visitantes',
                                'diferenciais_estacionamento visitantes e outros', 'diferenciais_festa e outros',
                                'diferenciais_apenas frente para o mar', 'diferenciais_futebol e outros',
                                'diferenciais_ginastica e outros', 'diferenciais_apenas piscina',
                                'diferenciais_piscina e outros', 'diferenciais_apenas playground',
                                'diferenciais_playground e outros', 'diferenciais_quadra e outros', 'diferenciais_apenas sa
                                'diferenciais_apenas salao de festas', 'diferenciais_apenas salao de jogos',
                                'diferenciais_apenas sauna', 'diferenciais_sauna e outros', 'diferenciais_apenas vestiario'
```

3 – Aplicação do Modelo Preditivo de Regressão

Foi aplicado o modelo de regressão pelos K vizinhos mais próximos (KNN regression). Houve a tentativa de aplicar outros modelos de

regressão, como a Regressão Polinomial, contudo, essa sobrecarregou a memória RAM do computador.

Como o KNN precisa da distância entre os pontos para efetuar o aprendizado, é interessante introduzir uma escala.

```
#Escalando
scaler = StandardScaler()
scaler.fit(matrizXTreino)

matrizXTreino = scaler.transform(matrizXTreino)
matrizXTeste = scaler.transform(matrizXTeste)
```

Por inspeção, verificou-se que o *StandardScaler* possuía a melhor eficiência.

Em seguida, treinou-se o regressor KNN, utilizando o conjunto de dados ‘MatrizXTreino’:

```
#Regressor KNN
regressorKNN = KNeighborsRegressor (n_neighbors = 5)
regressorKNN = regressorKNN.fit(matrizXTreino, matrizYTreino)
```

Para observar a eficiência do modelo, foram aplicados dois testes de medir a acurácia: o *Root Mean Square Percentage Error* (RMSPE) e a validação cruzada.

```
print ('AVALIAÇÃO DE RESULTADOS POR VALIDAÇÃO CRUZADA')

kfold  = KFold(n_splits=5, shuffle=True)
resultado = cross_val_score(regressorKNN, matrizXTreino, matrizYTreino, cv = kfold)
print("K-Fold (R^2) Scores: {0}".format(resultado))
print("Média dos R^2 para Cross-Validation K-Fold: {0}".format(resultado.mean()))

y_resposta_treino = regressorKNN.predict(matrizXTreino)
mse_treino = mean_squared_error(matrizYTreino, y_resposta_treino)
rmse_treino = math.sqrt(mse_treino)
r2_treino = r2_score(matrizYTreino, y_resposta_treino)
rmspe_treino = (np.sqrt(np.mean(np.square((matrizYTreino - y_resposta_treino) / matrizYTreino))))
print (f' MSE Treino: {mse_treino}, RMSE Treino: {rmse_treino}, R2 Treino: {r2_treino}, RMSPE Treino: {rmspe_treino}')
```

Obteve-se os seguintes resultados:

```
AVALIAÇÃO DE RESULTADOS POR VALIDAÇÃO CRUZADA
K-Fold (R^2) Scores: [0.77926372 0.80829004
0.77065762 0.81870434 0.69685798]
Média dos R^2 para Cross-Validation K-Fold:
0.7747547407990549
 MSE Treino: 42145574278.95879, RMSE Treino:
205293.87296984484, R2 Treino:
0.8673732610221505, RMSPE Treino:
0.25273408633897104
```

Por fim, salva-se o vetor de estimativas obtidos em um arquivo .csv:

```
y_resposta_teste = regressorKNN.predict(matrizXTeste)
print (f'Resposta Teste : {y_resposta_teste}')
Id = pd.read_csv('conjunto_de_teste.csv')
respostaKNNMartina2 = pd.DataFrame({'Id':Id.pop('Id'), 'preco':np.squeeze((y_resposta_teste))})
respostaKNNMartina2.to_csv("respostaKNNMartina2.csv", index=False)
```

4 – Conclusão:

Com o presente relatório, é possível observar a aplicação de diversos conceitos de *Machine Learning,* bem como suas aplicações por meio de diversos pacotes da linguagem Python, como *NumPy, ScikitLearn* e *SciPy.* Através de um modelo de regressão por *K-Nearest Neighbors*, foi possível estimar o preço de um imóvel na cidade de Recife com, aproximadamente, 30% de erro.

5- Referências:

Fonte dos IDHs da cidade de Recife: http://www.recife.pe.gov.br/pr/secplanejamento/pnud2005/7.%20IDH-M%20DENTRO%20DO%20RECIFE%20VAI%20DA%20%C3%81FRICA%20%C3%80%20NORUEGA.pdf

Fonte do mapa da cidade de Recife:

DE SANTANA, John Kennedy Ribeiro. ANÁLISE EVOLUTIVA DA OCUPAÇÃO DOS MORROS DA CIDADE DO RECIFE. **Simpósio Nacional de Geografia Urbana**, Universidade Federal do Espírio Santo, 14 nov. 2019. Disponível em: https://periodicos.ufes.br. Acesso em: 4 ago. 2022.

Informações sobre a regressão KNN:

https://towardsdatascience.com/the-basics-knn-for-classification-and-regression-c1e8a6c955

Informações sobre a Validação Cruzada:

https://drigols.medium.com/introdu%C3%A7%C3%A3o-a-valida%C3%A7%C3%A3o-cruzada-k-fold-2a6bced32a90