import os

print(os.listdir("../input"))

# Any results you write to the current directory are saved as output.

### car fuel consumption

import pandas as pd

df=pd.read\_excel(r"../input/measurements2.xlsx")

x = df.iloc[:,[0,2,3,4,5,6,7,8,9]]

y = df.iloc[:,1]

# =============================================================================

# Found missing values in temp\_inside and Specials

# Imputing values of temp\_inside

#=============================================================================

from sklearn.preprocessing import Imputer

imputer = Imputer()

x.loc[:,['temp\_inside']] = imputer.fit\_transform(x.loc[:,['temp\_inside']])

# =============================================================================

# Since NaN of specials cannot be imputed hence replacing all NaN by 0

# =============================================================================

x.loc[:,['specials']] = x.loc[:,['specials']].fillna(' ')

# =============================================================================

# Label Encoding

# =============================================================================

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

encoder = LabelEncoder()

x = pd.get\_dummies(x, prefix=['specials', 'gas\_type'], drop\_first=True)

# =============================================================================

# Splitting train test

# =============================================================================

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y, random\_state=9, train\_size = 0.8)

# =============================================================================

# Feature Scaling ----

# =============================================================================

from sklearn.preprocessing import StandardScaler

scale\_x = StandardScaler()

x\_train.iloc[:,0:4] = scale\_x.fit\_transform(x\_train.iloc[:,0:4])

x\_test.iloc[:,0:4] = scale\_x.fit\_transform(x\_test.iloc[:,0:4])

# =============================================================================

# Modelliong

# =============================================================================

from sklearn.linear\_model import LinearRegression

lin\_reg = LinearRegression()

lin\_reg.fit(x\_train, y\_train)

# =============================================================================

# predicting

# =============================================================================

y\_pred = lin\_reg.predict(x\_test)

# =============================================================================

# Metrics

# =============================================================================

from sklearn.metrics import mean\_squared\_error

rmse = mean\_squared\_error(y\_test, y\_pred)

# =============================================================================

# average of car when using SP98

# =============================================================================

y\_sp98 = lin\_reg.predict(x\_test[x\_test.loc[:,'gas\_type\_SP98']==1])

y\_E10 = lin\_reg.predict(x\_test[x\_test.loc[:,'gas\_type\_SP98']==0])

# =============================================================================

# This shows that E10 takes 0.33l more per 100kms than sp98

# =============================================================================

y\_sp98.mean()

y\_E10.mean()

diff\_in\_fuel\_100km = y\_E10.mean() - y\_sp98.mean()

# =============================================================================

# money matter

# =============================================================================

# per 100km on e10 will cost

e10\_100km = y\_E10.mean() \* 1.38

sp98\_100km = y\_sp98.mean() \* 1.46

# =============================================================================

# difference shows that for a 100km drive e10 costs more than sp98

# =============================================================================

cost\_diff = e10\_100km - sp98\_100km