#importing some important libraries and get data import pandas as pd

import numpy as np

import matplotlib.pyplot as plt import seaborn as sns

calories = pd.read\_csv('/content/calories.csv') exercise=pd.read\_csv('/content/exercise.csv')

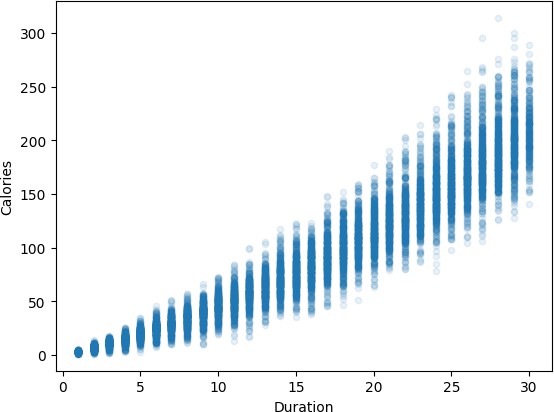
df=exercise.merge(calories,on='User\_ID')

Exploratory Data Analysis

 Data Visualization

#Visualizing Duration and Calories

df.plot(kind='scatter',x ="Duration" , y = 'Calories' ,alpha=0.1)

 <Axes: xlabel='Duration', ylabel='Calories'>

df["Gender"] = df["Gender"].astype("category").cat.codes

cor\_df = df.corr()

cor\_df['Calories'].sort\_values(ascending=False)

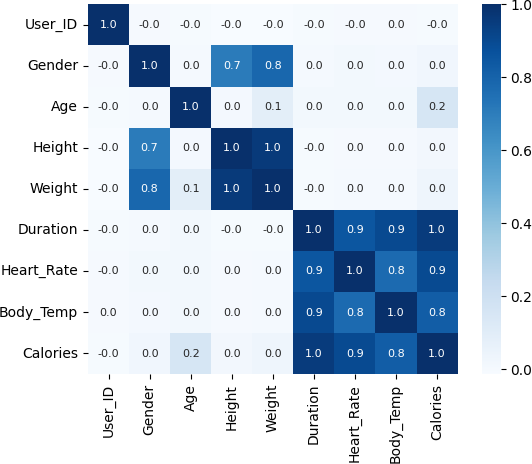


|  |  |
| --- | --- |
| Calories | 1.000000 |
| Duration | 0.955421 |
| Heart\_Rate | 0.897882 |
| Body\_Temp | 0.824558 |
| Age | 0.154395 |
| Weight | 0.035481 |
| Gender | 0.022357 |
| Height | 0.017537 |
| User\_ID | -0.001661 |

Name: Calories, dtype: float64

corr=df.corr()

sns.heatmap(corr,cbar=True,square=True,fmt='.1f',annot=True,annot\_kws={'size':8},cmap='Blues')

 <Axes: >

#  Splitting The Data

x = df.drop(columns=['User\_ID','Calories']) y = df['Calories']

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

X\_train,X\_test,y\_train,y\_test = train\_test\_split(x,y,test\_size=0.2,random\_state=1)

#  LABEL ENCODING

#Encoding

from sklearn.preprocessing import LabelEncoder le = LabelEncoder()

X\_train['Gender'] = le.fit\_transform(X\_train['Gender']) X\_test['Gender'] = le.fit\_transform(X\_test['Gender'])

X\_test



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Gender** | **Age** | **Height** | **Weight** | **Duration** | **Heart\_Rate** | **Body\_Temp** |
| **7576** | 0 | 74 | 165.0 | 68.0 | 29.0 | 106.0 | 41.0 |
| **10509** | 1 | 43 | 186.0 | 93.0 | 13.0 | 97.0 | 39.9 |
| **4253** | 0 | 43 | 148.0 | 52.0 | 29.0 | 108.0 | 40.5 |
| **5150** | 1 | 62 | 182.0 | 84.0 | 4.0 | 83.0 | 38.9 |
| **506** | 0 | 37 | 173.0 | 67.0 | 15.0 | 94.0 | 40.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **9533** | 1 | 34 | 182.0 | 86.0 | 25.0 | 101.0 | 41.3 |
| **13457** | 0 | 77 | 152.0 | 61.0 | 4.0 | 91.0 | 38.9 |
| **14764** | 0 | 24 | 165.0 | 65.0 | 6.0 | 94.0 | 39.5 |
| **8375** | 1 | 35 | 172.0 | 75.0 | 22.0 | 98.0 | 40.9 |
| **8500** | 0 | 29 | 178.0 | 68.0 | 5.0 | 77.0 | 39.2 |

3000 rows × 7 columns

from sklearn.metrics import mean\_squared\_error,mean\_absolute\_error,r2\_score

from sklearn.model\_selection import cross\_val\_score

##  LINEAR REGRESSION

from sklearn.linear\_model import LinearRegression model1 = LinearRegression()

model1.fit(X\_train,y\_train) model1.score(X\_test,y\_test)

 0.9655977245826504

y\_pred=model1.predict(X\_test) y\_pred

 array([198.81182363, 80.43555305, 194.40940033, ..., 22.14745631,

118.63504926, -11.98134672])

model1.score(X\_train,y\_train)  0.967592555473578

print('MSE is:',mean\_squared\_error(y\_test,y\_pred)) print('MAE is:',mean\_absolute\_error(y\_test,y\_pred))

print('RMSE is:',np.sqrt(mean\_squared\_error(y\_test,y\_pred))) print('R2 score is:',r2\_score(y\_test,y\_pred))

 MSE is: 138.12408611460899

MAE is: 8.479071745987955

RMSE is: 11.752620393538157

R2 score is: 0.9655977245826504

cv\_scores = cross\_val\_score(model1, x, y, cv=5) print('CV scores:', cv\_scores)

print('Mean CV score:', cv\_scores.mean())

 CV scores: [0.96712832 0.96658977 0.96769213 0.96828562 0.96606908] Mean CV score: 0.967152984018283

##  DECISION TREE REGRESSION

from sklearn.tree import DecisionTreeRegressor model2=DecisionTreeRegressor()

model2.fit(X\_train,y\_train) model2.score(X\_train,y\_train)

 1.0

y\_pred2=model2.predict(X\_test) model2.score(X\_train,y\_train)

 1.0

print('MSE is:',mean\_squared\_error(y\_test,y\_pred2)) print('MAE is:',mean\_absolute\_error(y\_test,y\_pred2))

print('RMSE is:',np.sqrt(mean\_squared\_error(y\_test,y\_pred2))) print('R2 score is:',r2\_score(y\_test,y\_pred2))

 MSE is: 30.542333333333332

MAE is: 3.541

RMSE is: 5.526511859512593

R2 score is: 0.9923928853194367

cv\_scores = cross\_val\_score(model2, x, y, cv=5) print('CV scores:', cv\_scores)

print('Mean CV score:', cv\_scores.mean())

 CV scores: [0.99315318 0.99259894 0.99318227 0.99267099 0.99299414] Mean CV score: 0.992919901338472

##  RANDOM FOREST REGRESSION

from sklearn.ensemble import RandomForestRegressor model3= RandomForestRegressor()

model3.fit(X\_train,y\_train) model3.score(X\_test,y\_test)

0.9976628512632606

y\_pred3=model3.predict(X\_test) model3.score(X\_train,y\_train)

 0.9996820747191102

print('MSE is:',mean\_squared\_error(y\_test,y\_pred3)) print('MAE is:',mean\_absolute\_error(y\_test,y\_pred3))

print('RMSE is:',np.sqrt(mean\_squared\_error(y\_test,y\_pred3))) print('R2 score is:',r2\_score(y\_test,y\_pred3))

 MSE is: 9.383580866666668

MAE is: 1.8174799999999998

RMSE is: 3.0632631076462675

R2 score is: 0.9976628512632606

cv\_scores = cross\_val\_score(model3, x, y, cv=5) print('CV scores:', cv\_scores)

print('Mean CV score:', cv\_scores.mean())

 CV scores: [0.99809395 0.99794617 0.99798817 0.99769319 0.99802519] Mean CV score: 0.9979493362465822

##  SUPPORT VECTOR REGRESSION

from sklearn import svm

model4 = svm.SVR(kernel='linear') model4.fit(X\_train, y\_train)

model4.score(X\_test, y\_test)  0.9617112879997057

y\_pred4 = model4.predict(X\_test) model4.score(X\_train, y\_train)

 0.9653749757500488

print('MSE is:', mean\_squared\_error(y\_test, y\_pred4)) print('MAE is:', mean\_absolute\_error(y\_test, y\_pred4))

print('RMSE is:', np.sqrt(mean\_squared\_error(y\_test, y\_pred4))) print('R2 score is:', r2\_score(y\_test, y\_pred4))

 MSE is: 153.72801041174648

MAE is: 8.372907296378877

RMSE is: 12.398710030150172

R2 score is: 0.9617112879997057

cv\_scores = cross\_val\_score(model4, x, y, cv=5) print('CV scores:', cv\_scores)

print('Mean CV score:', cv\_scores.mean())

 CV scores: [0.96472358 0.96452881 0.96588848 0.96604203 0.96345961] Mean CV score: 0.9649285030548607

##  RIDGE REGRESSION

from sklearn.linear\_model import Ridge model5 = Ridge(alpha=0.1)

model5.fit(X\_train, y\_train) model5.score(X\_test, y\_test)

 0.9655977082522759

y\_pred5 = model5.predict(X\_test) model5.score(X\_train, y\_train)

 0.9675925554241426

print('MSE is:', mean\_squared\_error(y\_test, y\_pred5)) print('MAE is:', mean\_absolute\_error(y\_test, y\_pred5))

print('RMSE is:', np.sqrt(mean\_squared\_error(y\_test, y\_pred5))) print('R2 score is:', r2\_score(y\_test, y\_pred5))

MSE is: 138.12415168056407

MAE is: 8.479061921349464

RMSE is: 11.752623182956393

R2 score is: 0.9655977082522759

cv\_scores = cross\_val\_score(model5, x, y, cv=5) print('CV scores:', cv\_scores)

print('Mean CV score:', cv\_scores.mean())

 CV scores: [0.96712827 0.96658973 0.96769217 0.96828561 0.96606915] Mean CV score: 0.9671529854344781

##  LASSO REGRESSION

from sklearn.linear\_model import Lasso model6 = Lasso(alpha=0.1)

model6.fit(X\_train, y\_train) model6.score(X\_test, y\_test)

 0.9655110582038617

y\_pred6 = model6.predict(X\_test) model6.score(X\_train, y\_train)

 0.9675318037824249

print('MSE is:', mean\_squared\_error(y\_test, y\_pred6)) print('MAE is:', mean\_absolute\_error(y\_test, y\_pred6))

print('RMSE is:', np.sqrt(mean\_squared\_error(y\_test, y\_pred6))) print('R2 score is:', r2\_score(y\_test, y\_pred6))

 MSE is: 138.472048981071

MAE is: 8.485678477262354

RMSE is: 11.767414711017496

R2 score is: 0.9655110582038617

cv\_scores = cross\_val\_score(model6, x, y, cv=5) print('CV scores:', cv\_scores)

print('Mean CV score:', cv\_scores.mean())

 CV scores: [0.96701829 0.96646919 0.96764276 0.96825862 0.9660856 ]

Mean CV score: 0.9670948936292485

model\_names=["Linear Regression","Decision Tree","Random Forest","SVR","Ridge Regression","Lasso Regression"] mse\_scores=[138.12,30.04,9.35,153.73,138.12,138.47]

mae\_scores=[8.48,3.48,1.81,8.37,8.48,8.49]

rmse\_scores=[11.75,5.48,3.06,12.40,11.75,11.77] r\_sq\_values=[0.97,0.99,0.99,0.96,0.96,0.97]

cv\_scores=[0.97,0.99,0.99,0.96,0.96,0.97]

x=np.arange(len(model\_names)) width=0.2

fig, ax = plt.subplots(figsize=(10, 6))

rects1=ax.bar(x - width, mse\_scores,width,label='MSE') rects2=ax.bar(x, mae\_scores,width,label='MAE')

rects3=ax.bar(x + width, rmse\_scores,width,label='RMSE')

rects4=ax.bar(x + 2\*width, r\_sq\_values,width,label='R2 Score') rects5=ax.bar(x + 2\*width, cv\_scores,width,label='CV Score') ax.set\_ylabel('Scores')

ax.set\_title('Model Comparison') ax.set\_xticks(x + width/2)

ax.set\_xticklabels(model\_names) ax.legend()

ax.bar\_label(rects1, padding=6) ax.bar\_label(rects2, padding=6) ax.bar\_label(rects3, padding=6) ax.bar\_label(rects4, padding=6) ax.bar\_label(rects5, padding=6)

fig.tight\_layout() plt.show()