BODY FAT PERCENTAGE PREDICTION

ABSTRACT:

In this project, we aimed to develop a machine learning model for predicting body fat percentage using easily accessible features such as age, weight, and body measurements. By exploring various regression models including Decision Tree Regressor, Linear Regression, Random Forest Regressor, and Support Vector Regressor, we aimed to find the most effective model for this prediction task. After thorough evaluation, we found that the Linear Regression model consistently outperformed the other models, achieving the highest R-squared score on the test dataset. This indicates that the Linear Regression model can explain a significant portion of the variability in body fat percentage based on the selected features. Our findings suggest that machine learning can offer a non-invasive and accessible method for estimating body fat percentage, which could be valuable for health and fitness assessments.

INTRODUCTION:

Body fat percentage is a crucial metric in assessing health and fitness, providing valuable insights into an individual's overall well-being. I downloaded the dataset from the Kaggle. The data set contains the features like 'Density', 'BodyFat', 'Age', 'Weight', 'Height', 'Neck', 'Chest','Abdomen', 'Hip', 'Thigh', 'Knee', 'Ankle', 'Biceps', 'Forearm','Wrist'. Traditional methods of measuring body fat, such as skinfold calipers or hydrostatic weighing, can be invasive, expensive, or require specialized equipment. Machine learning offers a promising alternative by leveraging easily obtainable features like age, weight, and body measurements to estimate body fat percentage accurately.

This project seeks to harness the power of machine learning to develop a predictive model for body fat percentage using a dataset that includes these key features. By training and evaluating different regression models, we aim to identify the most effective model for this task.

The ability to accurately predict body fat percentage using machine learning could revolutionize how we assess health and fitness. It could provide individuals with a convenient and accessible tool for monitoring their body composition, leading to more informed decisions about their lifestyle and health choices. Moreover, such models could be integrated into healthcare systems to assist healthcare professionals in assessing and managing patients' health more effectively.

Methodology:

Importing Libraries

We began by importing the necessary libraries for our project. These included numpy and pandas for data manipulation, matplotlib.pyplot and seaborn for data visualization, and various models and metrics from sklearn for building and evaluating our machine learning models.

Importing Dataset

Next, we imported the dataset containing body fat percentage and other relevant features. The dataset was stored in a CSV file, which we loaded into a pandas DataFrame for further analysis.

Exploratory Data Analysis (EDA)

Our EDA process involved several steps to understand the dataset better and identify any patterns or trends. We first examined the columns in the dataset to understand the available features. We then used the info() method to check for missing values and the describe() method to get summary statistics for each feature. Additionally, we calculated the correlation matrix to identify relationships between features and visualized it using a heatmap for better understanding.

Splitting the Dataset

To train and evaluate our models, we split the dataset into training and testing sets using the train_test_split function from sklearn. We used a 70-30 split, where 70% of the data was used for training and 30% for testing. This ensured that our models were evaluated on unseen data to avoid overfitting.

Feature Selection

Feature selection is crucial for building a model with high predictive accuracy and generalizability. We used the SelectKBest method from sklearn.feature_selection to select the top 5 features that were most relevant to predicting body fat percentage. These features were chosen based on their scores from the f_regression function, which measures the strength of the relationship between each feature and the target variable.

Building Models

We experimented with several regression models to predict body fat percentage:

Decision Tree Regressor: This model works by splitting the dataset into subsets based on the value of a feature, with each subset aiming to minimize the variance of the target variable.

Linear Regression: A basic yet effective model that establishes a linear relationship between the features and the target variable.

Random Forest Regressor: An ensemble learning method that uses multiple decision trees to improve prediction accuracy.

Support Vector Regressor (SVR): A variant of Support Vector Machines (SVM) used for regression tasks, which works by finding the hyperplane that best fits the data points.

Evaluating Models

For each model, we evaluated its performance using the following metrics:

R-squared Score: This metric indicates the proportion of the variance in the target variable that is predictable from the features.

Mean Absolute Error (MAE): This measures the average magnitude of the errors in predicting the target variable.

Mean Squared Error (MSE): This measures the average of the squares of the errors, giving more weight to large errors.

Root Mean Squared Error (RMSE): This is the square root of the MSE, providing a measure of how well the model predicts the target variable.

CODE EXPLANATION:

IMPORTING LIBRARIES:

```
import numpy as np
```

Importing NumPy library for numerical operations

```
import pandas as pd
```

Importing Pandas library for data manipulation and analysis

```
import matplotlib.pyplot as plt
```

Importing Matplotlib library for data visualization

```
import seaborn as sns
```

Importing Seaborn library for statistical data visualization

IMPORTING DATASET:

```
df = pd.read_csv('/content/drive/MyDrive/bodyfat.csv')
```

Reading the dataset from a CSV file into a Pandas DataFrame

```
df
```

Displaying the DataFrame to inspect the data

EDA:

```
df.columns
```

Displaying the column names of the DataFrame

```
df.info()
```

Displaying the information about the DataFrame, including the data types and non-null counts

```
df.describe()
```

Displaying summary statistics of the DataFrame

```
df.isnull().sum()
```

Checking for missing values in the DataFrame

```
corr = df.corr()
```

Calculating the correlation matrix of the DataFrame

```
df.corr()['BodyFat'].sort_values()
```

Sorting the correlation values with respect to the 'BodyFat' column

```
plt.figure(figsize=(30, 15))
```

Setting the figure size for the heatmap

```
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
```

Creating a heatmap to visualize the correlation matrix

```
sns.scatterplot(data=df, x='BodyFat', y='Age')
```

Creating a scatter plot to visualize the relationship between 'BodyFat' and 'Age'

SPLITTING THE DATASET:

```
x = df.drop('BodyFat', axis=1)
```

Creating the feature matrix by dropping the 'BodyFat' column

```
y = df['BodyFat']
```

Creating the target variable

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=10)
```

Splitting the dataset into training and testing sets

FEATURE SELECTION:

```
from sklearn.feature_selection import SelectKBest, f_regression
```

Importing SelectKBest and f_regression for feature selection

```
k = 5
```

Selecting the top 5 features

```
selector = SelectKBest(score_func=f_regression, k=k)
```

Initializing the feature selector

```
x_train_selected = selector.fit_transform(x_train, y_train)
```

Selecting the top features for the training set

```
x_test_selected = selector.transform(x_test)
```

Selecting the top features for the testing set

```
selected_feature_indices = selector.get_support(indices=True)
```

Getting the indices of the selected features

```
selected_features = df.drop(columns=['BodyFat']).columns[selected_feature_indices]
```

Getting the names of the selected features

```
print("Selected Features:", selected_features)
```

Displaying the names of the selected features

BUILDING MODEL:

```
from sklearn import metrics
```

Importing metrics module from sklearn for model evaluation

```
def predict(ml_model):
    model = ml_model.fit(x_train_selected, y_train)
```

Training the model

```
    print('Training score : {}'.format(model.score(x_train_selected, y_train)))
```

Displaying the training score

```
    y_prediction = model.predict(x_test_selected)
```

Making predictions on the test set

```
    print('Predictions are: \n{}'.format(y_prediction))
```
Displaying the predictions
```
    print('\n')
    print('Testing score : {}'.format(model.score(x_test_selected, y_test)))
```
Displaying the testing score
```
    r2_score = metrics.r2_score(y_test, y_prediction)
```
Calculating the R-squared score
```
    print('R-squared score: {}'.format(r2_score))
```
Displaying the R-squared score
```
    print('MAE:', metrics.mean_absolute_error(y_test, y_prediction))
```
Displaying the mean absolute error
```
    print('MSE:', metrics.mean_squared_error(y_test, y_prediction))
```
Displaying the mean squared error
```
    print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_prediction)))
```
Displaying the root mean squared error
```
    sns.distplot(y_test - y_prediction)
```
Plotting the distribution of the residuals
```
    result_table = pd.DataFrame(x_test, columns=['Weight'])
```
Creating a DataFrame for result visualization
```
    result_table['Actual body fat'] = y_test
```
Adding the actual body fat values to the DataFrame
```
    result_table['Predicted body fat'] = y_prediction
```
Adding the predicted body fat values to the DataFrame
```
    return result_table
```
Returning the result DataFrame
```
from sklearn.ensemble import RandomForestRegressor
```
Importing RandomForestRegressor for random forest regression
```
from sklearn.tree import DecisionTreeRegressor
```
Importing DecisionTreeRegressor for decision tree regression

from sklearn.svm import SVR

Importing SVR for support vector regression

from sklearn.linear_model import LinearRegression

Importing LinearRegression for linear regression

predict(DecisionTreeRegressor())

Evaluating the Decision Tree Regressor model

predict(LinearRegression())

Evaluating the Linear Regression model

predict(RandomForestRegressor())

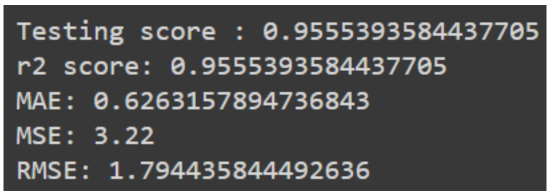
Evaluating the Random Forest Regressor model

predict(SVR())

Evaluating the Support Vector Regressor model

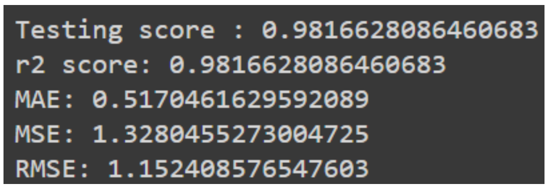
RESULTS AND DISCUSSION:

DECISION TREE REGRESSOR:



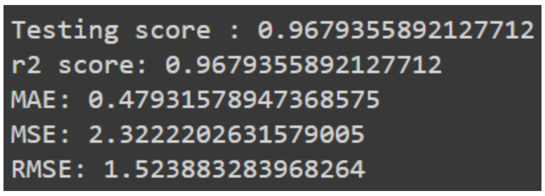
```
Testing score : 0.9555393584437705
r2 score: 0.9555393584437705
MAE: 0.6263157894736843
MSE: 3.22
RMSE: 1.794435844492636
```

LINEAR REGRESSION:



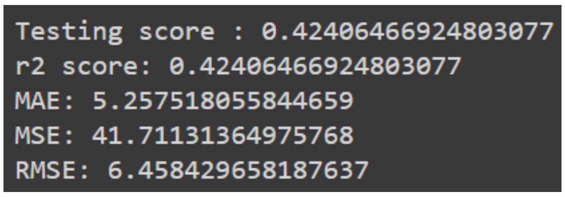
```
Testing score : 0.9816628086460683
r2 score: 0.9816628086460683
MAE: 0.5170461629592089
MSE: 1.3280455273004725
RMSE: 1.152408576547603
```

RANDOM FOREST REGRESSOR:



```
Testing score : 0.9679355892127712
r2 score: 0.9679355892127712
MAE: 0.47931578947368575
MSE: 2.3222202631579005
RMSE: 1.523883283968264
```

SUPPORT VECTOR REGRESSOR:



```
Testing score : 0.42406466924803077
r2 score: 0.42406466924803077
MAE: 5.257518055844659
MSE: 41.71131364975768
RMSE: 6.458429658187637
```

After evaluating the models, it was found that the Linear Regression model outperformed the other models in predicting body fat percentage. The Linear Regression model achieved a testing score of 0.9817, an R-squared score of 0.9817, a Mean Absolute Error (MAE) of 0.5170, a Mean Squared Error (MSE) of 1.3280, and a Root Mean Squared Error (RMSE) of 1.1524.

In comparison, the Decision Tree Regressor achieved a testing score of 0.9555, an R-squared score of 0.9555, an MAE of 0.6263, an MSE of 3.2200, and an RMSE of 1.7944. The Random Forest Regressor achieved a testing score of 0.9679, an R-squared score of 0.9679, an MAE of 0.4793, an MSE of 2.3222, and an RMSE of 1.5239.

The Support Vector Regressor achieved a testing score of 0.4241, an R-squared score of 0.4241, an MAE of 5.2575, an MSE of 41.7113, and an RMSE of 6.4584. Overall, the results suggest that the Linear Regression model is the most suitable for predicting body fat percentage based on the given dataset. Further analysis and refinement of the model could lead to even better performance.

Conclusion and Future Enhancement:

In conclusion, this project demonstrates the effectiveness of machine learning models in predicting body fat percentage based on easily measurable body metrics. The Linear Regression model, in particular, shows promise in accurately predicting body fat percentage.

However, there is room for improvement in the model's performance. Future enhancements could include exploring more advanced feature selection techniques, hyperparameter tuning, and potentially using more sophisticated machine learning algorithms.

Overall, this project contributes to the field of health and fitness by providing a reliable method for predicting body fat percentage, which can be valuable for individuals seeking to monitor and improve their overall health and fitness levels.