Git Hub Link: https://raw.githubusercontent.com/RamishaRaniK/dataset/main/insurance_pre.csv

Problem Statement or Requirement:

A client’s requirement is, he wants to predict the insurance charges based on the several parameters. The Client has provided the dataset of the same. As a data scientist, you must develop a model which will predict the insurance charges.

1.) Identify your problem statement

2.) Tell basic info about the dataset (Total number of rows, columns)

3.) Mention the pre-processing method if you’re doing any (like converting string to number – nominal data)

4.) Develop a good model with r2_score. You can use any machine learning algorithm; you can create many models. Finally, you have to come up with final model.

5.) All the research values (r2_score of the models) should be documented. (You can make tabulation or screenshot of the results.)

6.) Mention your final model, justify why u have chosen the same.

1.) Identify your problem statement The problem statement is to predict the insurance charges based on various parameters such as age

Domain is ML

Learning Type is Supervised Learning

Problem is Regression as they have clear requirement of predicting the insurance charges

The dataset contains information about individuals, including their age and many more features, so it is a multileaner regression problem.

2.) Tell basic info about the dataset (Total number of rows, columns)

In [96]:

```
import pandas as pd
# Load the dataset
data = pd.read_csv("insurance_pre.csv")

# The dataset contains the following number of rows and columns
rows, columns = data.shape
print(f"The dataset contains {rows} rows and {columns} columns.")
print(data)
```

```
The dataset contains 1338 rows and 6 columns.
      age     sex     bmi  children smoker      charges
0      19  female  27.900         0    yes  16884.92400
1      18    male  33.770         1     no   1725.55230
2      28    male  33.000         3     no   4449.46200
3      33    male  22.705         0     no  21984.47061
4      32    male  28.880         0     no   3866.85520
...   ...     ...     ...       ...    ...          ...
1333   50    male  30.970         3     no  10600.54830
1334   18  female  31.920         0     no   2205.98080
1335   18  female  36.850         0     no   1629.83350
1336   21  female  25.800         0     no   2007.94500
1337   61  female  29.070         0    yes  29141.36030

[1338 rows x 6 columns]
```

3.) Mention the pre-processing method if you’re doing any (like converting string to number – nominal data)

In [97]:

```
# The dataset contains categorical variables
# Convert categorical variables to numerical using one-hot encoding as they are nominal variables.
data = pd.get_dummies(data, drop_first=True) #as no comes 1st in alphabetical order - it will be dropped
# The dataset now contains only numerical values
# Check the first few rows of the dataset
print(data.head())
# Check the columns of the dataset
print(data.columns)
# Check the data types of the columns
print(data.dtypes)
```

```
   age     bmi  children      charges  sex_male  smoker_yes
0   19  27.900         0  16884.92400     False        True
1   18  33.770         1   1725.55230      True       False
2   28  33.000         3   4449.46200      True       False
3   33  22.705         0  21984.47061      True       False
4   32  28.880         0   3866.85520      True       False
Index(['age', 'bmi', 'children', 'charges', 'sex_male', 'smoker_yes'], dtype='object')
age             int64
bmi           float64
children        int64
charges       float64
sex_male         bool
smoker_yes       bool
dtype: object
```

4.) Develop a good model with r2_score. You can use any machine learning algorithm; you can create many models. Finally, you have to come up with final model.

In [98]:

```python
# Split the dataset into independent and dependent variables
independent_variables = data[['age', 'bmi', 'children', 'sex_male', 'smoker_yes']]

dependent_variables = data[['charges']]
# Split the dataset into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(independent_variables, dependent_variables, test_size=0.2, random_state=0)
```

In [99]:

```python
# Standardize the features
###################### Inorder to improve the performance of the model, lets used "Standardization Method", - we will scale the features usi
from sklearn.preprocessing import StandardScaler
# Create a StandardScaler object
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

LinearRegression Model

In [100…

```python
# Import the LinearRegression from sklearn.model_selection
from sklearn.linear_model import LinearRegression
# Create the Linear Regression model
regressor = LinearRegression()
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)
from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.7978644236809905
```

SVR Model

Using the "linear" kernel

In [101…

```python
#SVM - Support Vector Machine, regression method
from sklearn.svm import SVR
# Create the SVR model
regressor = SVR(kernel='linear', C=100, gamma='scale', epsilon=.1)
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)
from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}\n")
```

```
R2 Score: 0.6423323553032765
```

```
/home/deehub/JoinDeeHub/.venv/lib/python3.10/site-packages/sklearn/utils/validation.py:1408: DataConversionWarning: A column-vector y was p
assed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Using the "rbf" kernel

In [102…

```python
#SVM - Support Vector Machine, regression method
from sklearn.svm import SVR
# Create the SVR model
regressor = SVR(kernel='rbf', C=100000, gamma='scale', epsilon=.1)
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)
from sklearn.metrics import r2_score
# Calculate the R2 score
```

```
r2 = r2_score(y_test, y_pred)
print(f"\n R2 Score: {r2}\n")
```

```
/home/deehub/JoinDeeHub/.venv/lib/python3.10/site-packages/sklearn/utils/validation.py:1408: DataConversionWarning: A column-vector y was p
assed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
 R2 Score: 0.8870555499325374
```

Using the "poly" kernel

In [103…

```
#SVM - Support Vector Machine, regression method
from sklearn.svm import SVR
# Create the SVR model
regressor = SVR(kernel='poly', C=100, gamma='scale', epsilon=.1)
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)
from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}\n")
```

```
R2 Score: 0.6591216048381952
```

```
/home/deehub/JoinDeeHub/.venv/lib/python3.10/site-packages/sklearn/utils/validation.py:1408: DataConversionWarning: A column-vector y was p
assed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Using the "sigmod" kernel

In [104…

```
#SVM - Support Vector Machine, regression method
from sklearn.svm import SVR
# Create the SVR model
regressor = SVR(kernel='sigmoid', C=100, gamma='scale', epsilon=.1)
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)
from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}\n")
```

```
R2 Score: 0.5353164568930158
```

```
/home/deehub/JoinDeeHub/.venv/lib/python3.10/site-packages/sklearn/utils/validation.py:1408: DataConversionWarning: A column-vector y was p
assed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Using the "precomputed" kernel

```
                ###########################   ValueError: Precomputed matrix must be a square matrix. Input is a 40x6 matrix.
```

Decision Tree Regressor Model

criterion='poisson', splitter='random'

In [105…

```
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='poisson', splitter='random')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.788969844221026
```

In [106…

```
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='poisson', splitter='best')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.7831900397331205
```

criterion='squared_error', splitter='best'

In [107…

```python
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='squared_error', splitter='best')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.7487994461010957
```

In [108…

```python
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='squared_error', splitter='random')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.7136881660147943
```

criterion='friedman_mse', splitter='best'

In [109…

```python
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='friedman_mse', splitter='best')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.7232010258509102
```

In [110…

```python
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='friedman_mse', splitter='random')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.7631996241425044
```

criterion='absolute_error', splitter='best'

In [111…

```python
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='absolute_error', splitter='best')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.683460593261744
```

In [112…

```python
# Import the Decision Tree Regressor from sklearn.tree
from sklearn.tree import DecisionTreeRegressor
# Create the DecisionTreeRegressor with specific criterion and splitter
regressor = DecisionTreeRegressor(criterion='absolute_error', splitter='random')
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)

from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"R2 Score: {r2}")
```

```
R2 Score: 0.7177715716332422
```

Random Forest Regressor Model

In [113…

```
# Import the Random Forest Regressor from sklearn.ensemble
###### We know under Random Forest Regressor, have sklearn.ensemble, sklearn.bagging, sklearn.boostrapping, sklearn.randomfeaturesselection
from sklearn.ensemble import RandomForestRegressor
# Create the Random Forest Regressor model
regressor = RandomForestRegressor(n_estimators=100, random_state=42)
# Fit the model to the training data
regressor.fit(X_train, y_train)
# Predict the results
y_pred = regressor.predict(X_test)
from sklearn.metrics import r2_score
# Calculate the R2 score
r2 = r2_score(y_test, y_pred)
print(f"\n R2 Score: {r2}")
```

```
/home/deehub/JoinDeeHub/.venv/lib/python3.10/site-packages/sklearn/base.py:1389: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  return fit_method(estimator, *args, **kwargs)
 R2 Score: 0.8751002051529456
```

5.) All the research values (r2_score of the models) should be documented.

(You can make tabulation or screenshot of the results.)

Model	Kernel/Criterion	R² Score
Linear Regression	-	**0.80**
SVR	linear	0.65
SVR	rbf	0.89
SVR	poly	0.66
SVR	sigmoid	0.54
Decision Tree Regressor	poisson, random	0.79
Decision Tree Regressor	poisson, best	0.79
Decision Tree Regressor	squared_error, best	0.75
Decision Tree Regressor	squared_error, random	0.72
Decision Tree Regressor	friedman_mse, best	0.73
Decision Tree Regressor	friedman_mse, random	0.77
Decision Tree Regressor	absolute_error, best	0.69
Decision Tree Regressor	absolute_error, random	0.72
Random Forest Regressor	n_estimators=100, random_state=42	**0.88**

6.) Mention your final model, justify why u have chosen the same.

Final Model Justification Chosen Model: Random Forest Regressor Why?

Achieved the highest R² score of 0.88

Robust to overfitting compared to decision trees

Handles feature importance well

Works well with both categorical and numerical features

Conclusion :

The Random Forest Regressor was selected as the final model due to its superior accuracy and robustness in handling the data. With an R² score of 0.88, it provides the most reliable predictions for insurance charges based on client attributes.