

Model Development Phase Template

Date	7 July 2024
Team ID	team-739757
Project Title	Medical Cost Prediction
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshot

Initial model training code

```

Linear Regression

[ ] from sklearn.linear_model import LinearRegression

▶ lr= LinearRegression()
  lr.fit(X_train,y_train)

LinearRegression
LinearRegression()

[ ] y_pred1=lr.predict(X_test)

▶ from sklearn import metrics

[ ] score1=metrics.r2_score(y_test,y_pred1)
  print(score1)

0.7837015388200166

[ ] s1=metrics.mean_absolute_error( y_test,y_pred1)
  print(s1)

3320.557034987548

[ ] rmse_lr=np.sqrt(metrics.mean_squared_error(y_test,y_pred1))
  
```

```
rmse_lr=np.sqrt(metrics.mean_squared_error(y_test,y_pred1))
print("mean_squared_error",rmse_lr)
```

```
mean_squared_error 4845.6792366495965
```

```
accuracy=lr.score(X_test,y_test)
print("-----Linear Regression-----")
print("model accuracy \t\t",accuracy)
print(f'Accuracy in percentage\t:{accuracy:.1%}')
```

```
-----Linear Regression-----
model accuracy      0.7837015388200166
Accuracy in percentage :78.4%
```

Support Vector MachineRegressor

```
[ ] from sklearn.svm import SVR
```

```
[ ] svm= SVR()
    svm.fit(X_train,y_train)
```

SVR

```
[ ] y_pred2=svm.predict(X_test)
```

```
score2=metrics.r2_score(y_test,y_pred2)
print(score2)
```

```
-0.057306433750309305
```

```
[ ] s2=metrics.mean_absolute_error( y_test,y_pred2)
    print(s2)
```

```
7754.513457705959
```

```
[ ] rmse_svm=np.sqrt(metrics.mean_squared_error(y_test,y_pred2))
    print("root_mean_squared_error",rmse_svm)
```

```
root_mean_squared_error 10713.4262641038
```

```
accuracy=svm.score(X_test,y_test)
print("-----Support Vector Machine-----")
print("model accuracy \t\t",accuracy)
print(f'Accuracy in percentage\t:{accuracy:.1%}')
```

```
-----Support Vector Machine-----
model accuracy      -0.057306433750309305
Accuracy in percentage :-5.7%
```

RandomForestRegressor

```
[ ] from sklearn.ensemble import RandomForestRegressor

[ ] rf= RandomForestRegressor()
    rf.fit(X_train,y_train)

[ ] y_pred3=rf.predict(X_test)

[ ] score3=metrics.r2_score(y_test,y_pred3)
    print(score3)
0.8302918166174308

[ ] s3=metrics.mean_absolute_error( y_test,y_pred3)
    print(s3)
2158.311786770744

[ ] rmse_rf=np.sqrt(metrics.mean_squared_error(y_test,y_pred3))
    print("root_mean_squared_error",rmse_rf)
root_mean_squared_error 4292.193966762153
```

```
accuracy=rf.score(X_test,y_test)
print("-----RandomForestRegressor-----")
print("model accuracy \t\t",accuracy)
print(f'Accuracy in percentage\t:{accuracy:.1%}')
```

```
-----RandomForestRegressor-----
model accuracy      0.8302918166174308
Accuracy in percentage :83.0%
```

GradientBoostingRegressor

```
[ ] from sklearn.ensemble import GradientBoostingRegressor

[ ] gb= GradientBoostingRegressor()
    gb.fit(X_train,y_train)

[ ] y_pred4=gb.predict(X_test)

[ ] score4=metrics.r2_score(y_test,y_pred4)
    print(score4)
0.8451154840835637

[ ] s4=metrics.mean_absolute_error( y_test,y_pred4)
    print(s4)
2174.9371457221414

[ ] rmse_gb=np.sqrt(metrics.mean_squared_error(y_test,y_pred4))
    print("root_mean_squared_error",rmse_gb)
root_mean_squared_error 4100.4540432147405
```

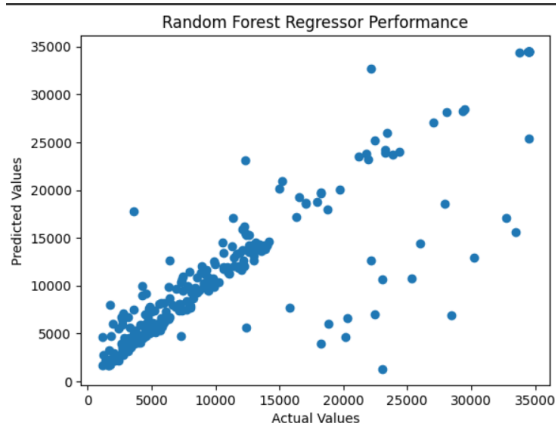
```
accuracy=gb.score(X_test,y_test)
print("-----GradientBoostingRegressor-----")
print("model accuracy \t\t",accuracy)
print(f'Accuracy in percentage\t:{accuracy:.1%}')

-----GradientBoostingRegressor-----
model accuracy      0.8451154840835637
Accuracy in percentage :84.5%
```

Model Validation and Evaluation Report:

Model	Performance Report	Accuracy	Regression matrix
Linear Regression	 <p>Linear Regression Performance</p> <p>The scatter plot shows Predicted Values on the y-axis (0 to 35000) and Actual Values on the x-axis (0 to 35000). The data points are blue dots, showing a positive correlation between predicted and actual values, with some scatter at higher values.</p>	78.4%	<pre>from sklearn.metrics import mean_squared_error, r2_score from sklearn.model_selection import train_test_split print("Regression Metrics:") print("Mean absolute Error:", s1) print("Root Mean Squared Error:", rmse_lr) score1=metrics.r2_score(y_test,y_pred1) print("R-squared:",score1) Regression Metrics: Mean absolute Error: 3320.557034987548 Root Mean Squared Error: 4845.6792366495965 R-squared: 0.7837015388200166</pre>
Support Vector Machine Regressor	 <p>Support Vector Machine Regressor Performance</p> <p>The scatter plot shows Predicted Values on the y-axis (9450 to 9650) and Actual Values on the x-axis (0 to 35000). The data points are blue dots, showing a positive correlation between predicted and actual values, with some scatter at higher values.</p>	-5.7%	<pre>from sklearn.svm import SVR from sklearn.metrics import mean_squared_error, r2_score from sklearn.model_selection import train_test_split print("Regression Metrics:") print("Mean absolute Error:", s2) print("Root Mean Squared Error:", rmse_svm) score2=metrics.r2_score(y_test,y_pred2) print("R-squared:",score2) Regression Metrics: Mean absolute Error: 7754.513457705959 Root Mean Squared Error: 10713.4262641038 R-squared: -0.057306433750309305</pre>

Random
Forest
Regressor

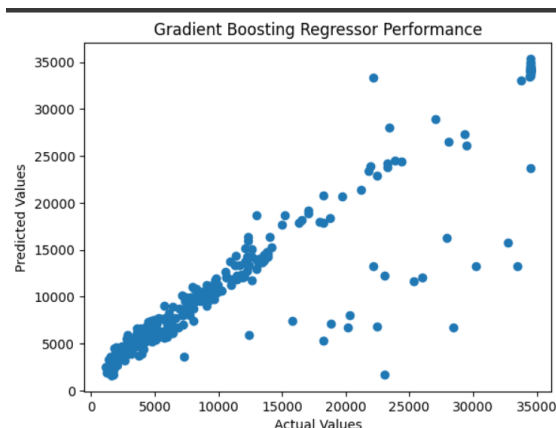


83%

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
print("Regression Metrics:")
print("Mean absolute Error:", s3)
print("Root Mean Squared Error:", rmse_rf)
score3=metrics.r2_score(y_test,y_pred3)
print("R-squared:", score3)
```

Regression Metrics:
Mean absolute Error: 2158.311786770744
Root Mean Squared Error: 4292.193966762153
R-squared: 0.8302918166174308

Gradient
Boosting
Regressor



84.5%

```
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split
print("Regression Metrics:")
print("Mean absolute Error:", s4)
print("Root Mean Squared Error:", rmse_gb)
score4=metrics.r2_score(y_test,y_pred4)
print("R-squared:", score4)
```

Regression Metrics:
Mean absolute Error: 2174.9371457221414
Root Mean Squared Error: 4100.4540432147405
R-squared: 0.8451154840835637