Medical Cost Prediction

Presented by
Nina Aulia N. Azhary

Presentation Outline

Today's Topics

NEXT

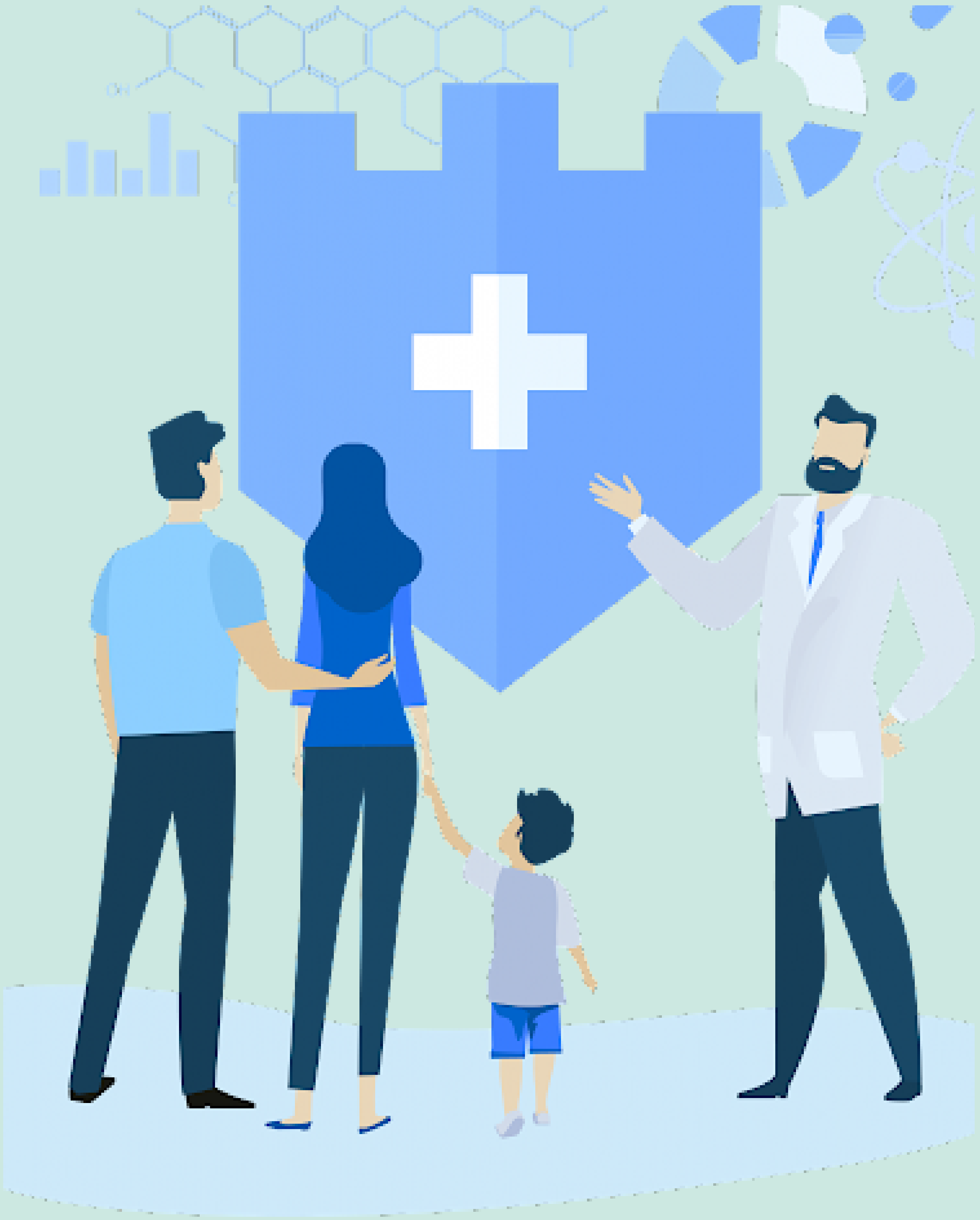
Problem Formulation

Business Problem

→

During the spread of COVID-19, the number of patients in the hospital has increased in the last two years. Hence, hospitals need technology to predict the cost of health that can give more efficient, helpful, and faster analysis for patients' convenience.

Problem Statement



●○○

What factors have contributed to the high cost of healthcare?

●●○

how to classify patients based on their insurance costs?

●●●

Is the 'region' a factor in the high cost of health insurance?

Dataset :

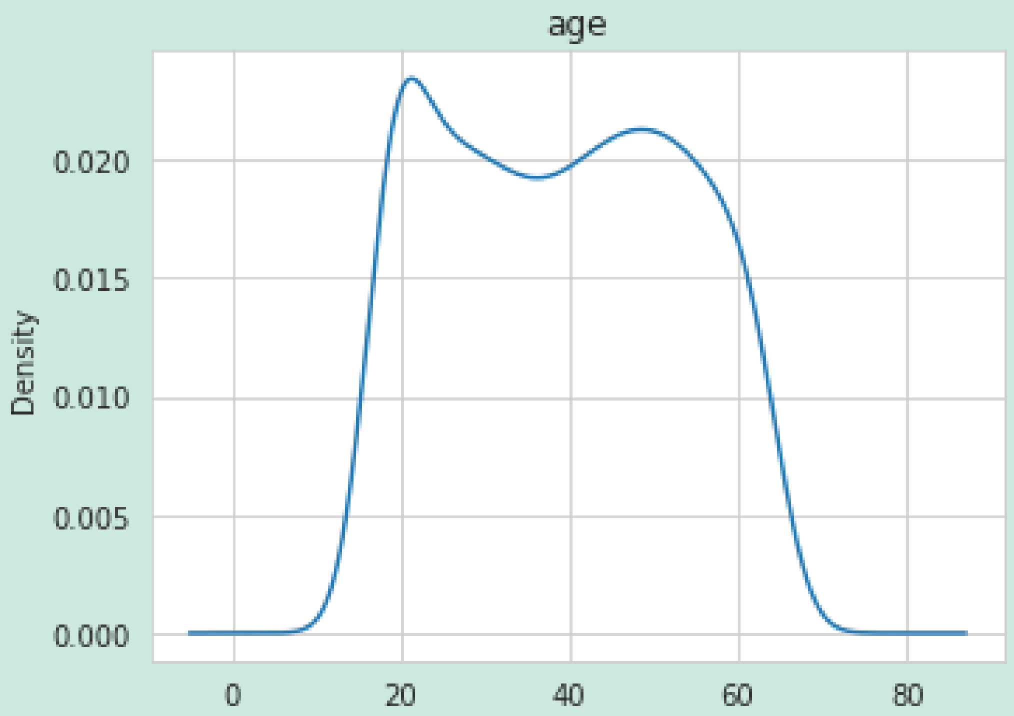
Insurance dataset has 1338 rows and 6 features :

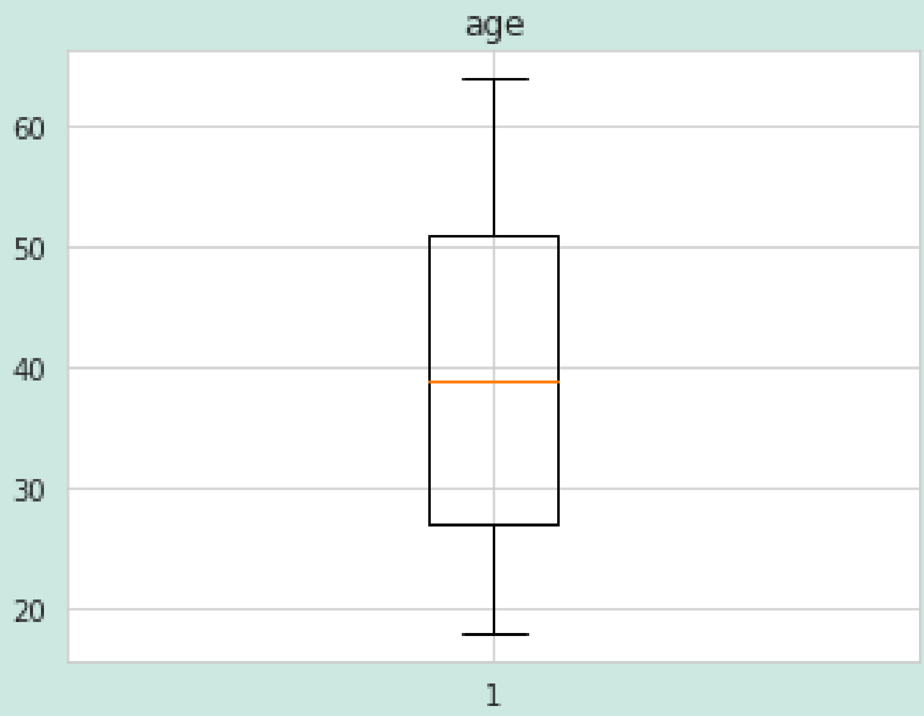
- Age : age of primary beneficiary
- Sex: insurance contractor gender, female, male
- BMI: Body mass index, providing an understanding of body, weights that are relatively high or low relative to height,objective index of body weight (kg / m ^ 2) using the ratio of height to weight, ideally 18.5 to 24.9
- Children: Number of children covered by health insurance / Number of dependents
- Smoker: Smoking, yes / no
- Region: the beneficiary's residential area in the US, northeast, southeast, southwest, northwest.
- Charges: Individual medical costs billed by health insurance

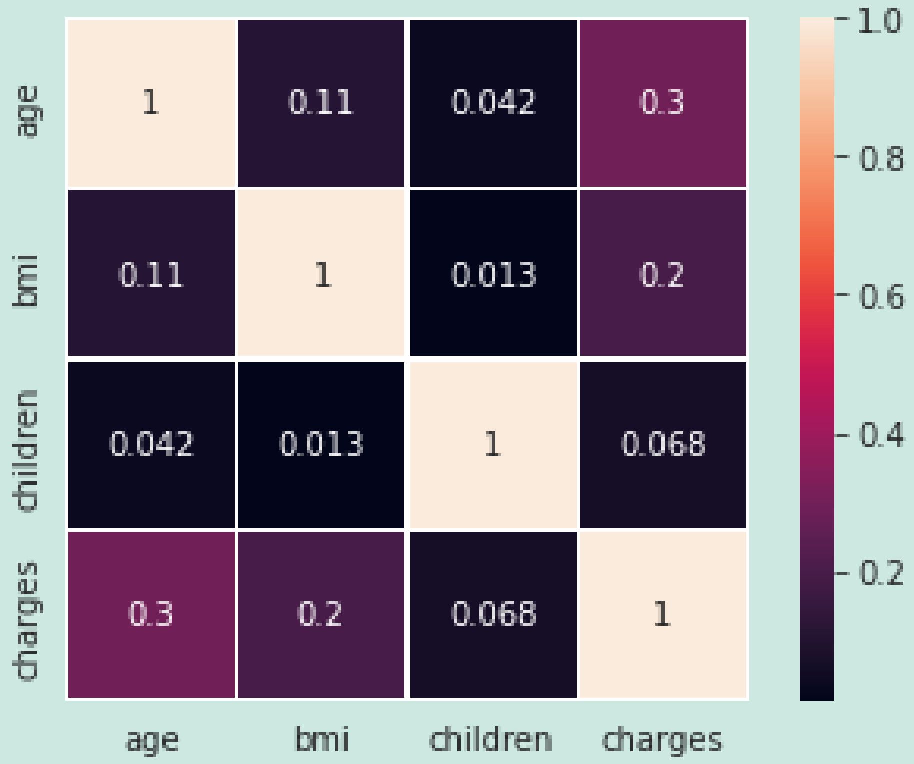
	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.693750	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

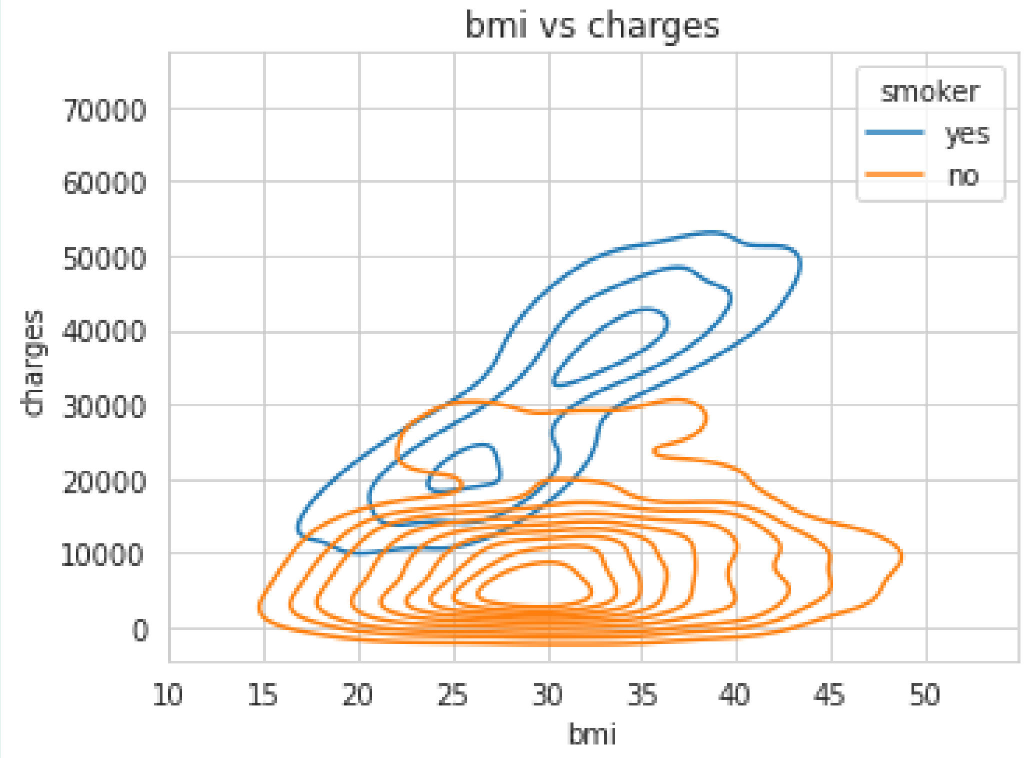
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

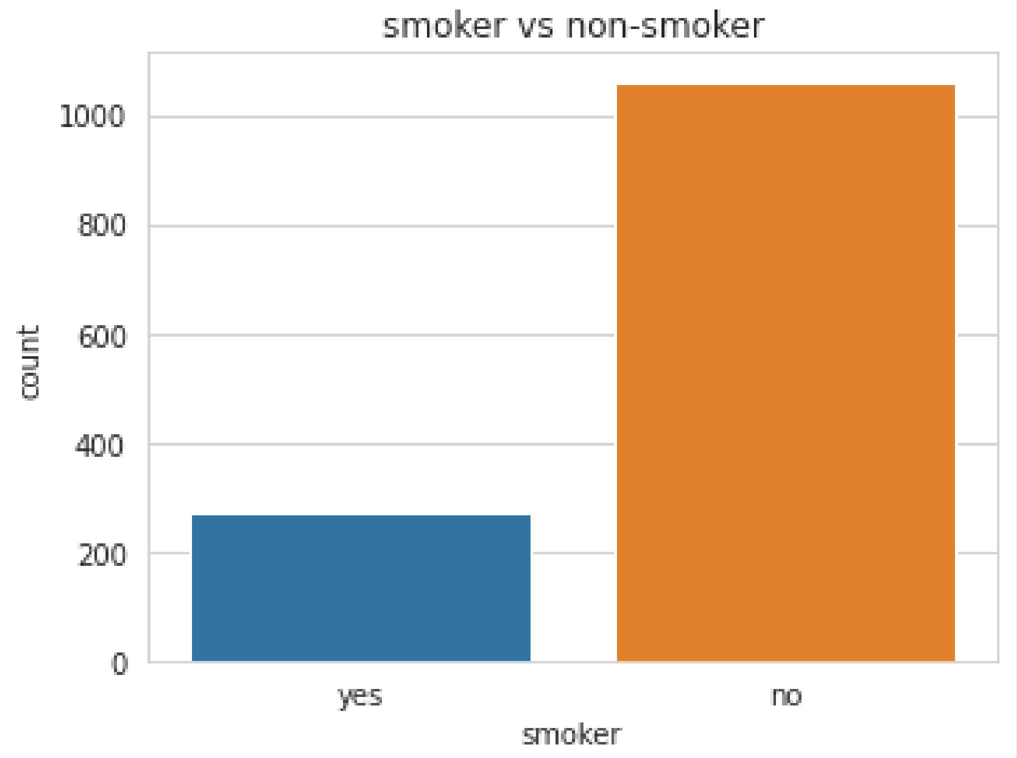
Exploratory data analysis





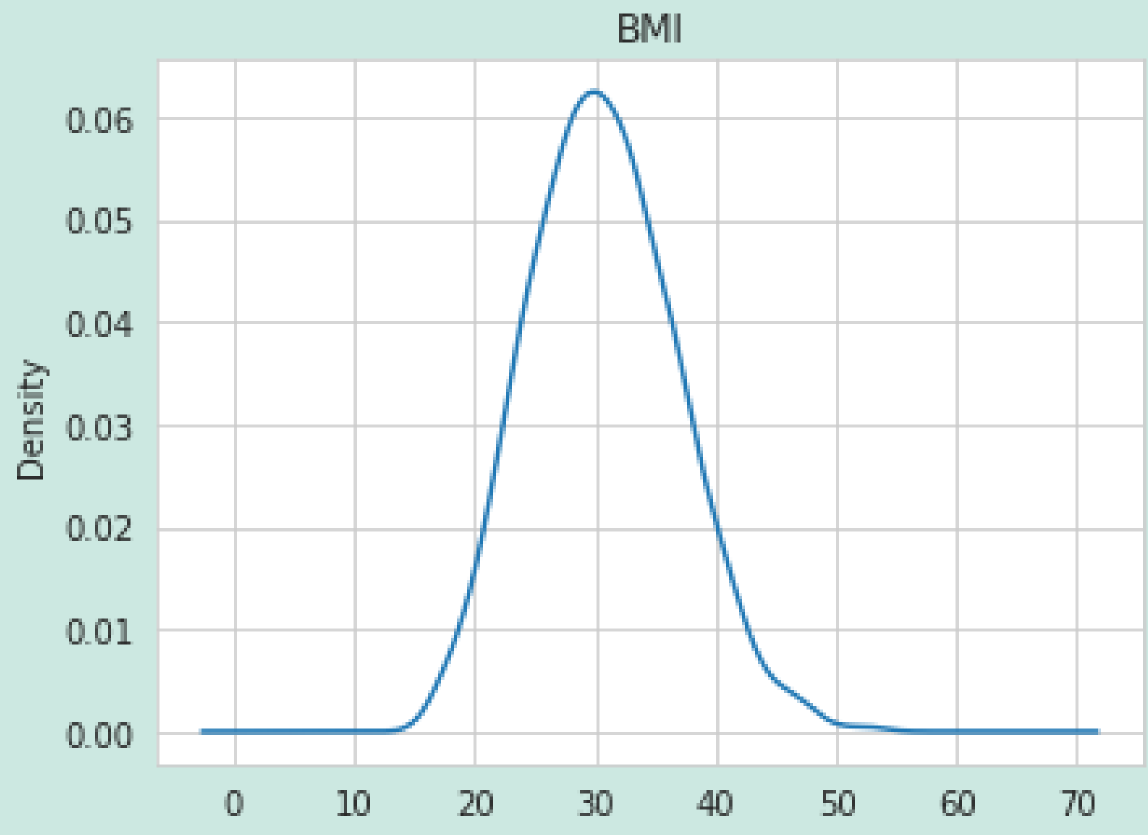


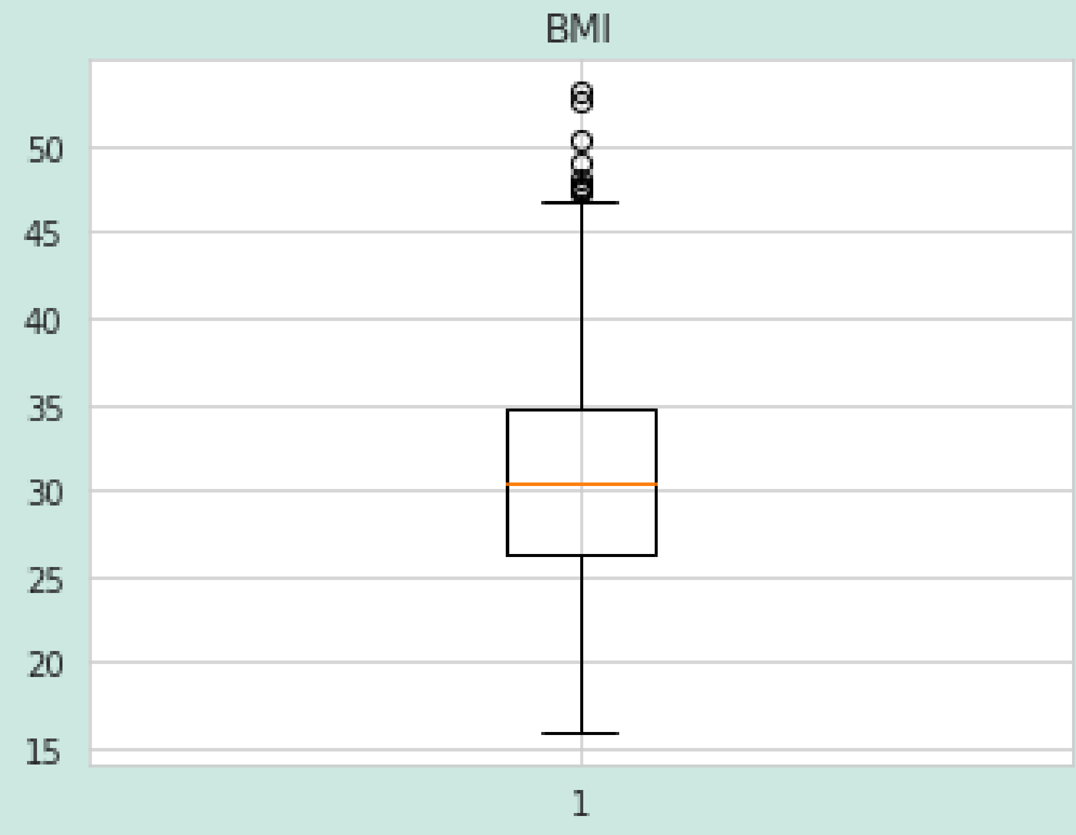




Finding :

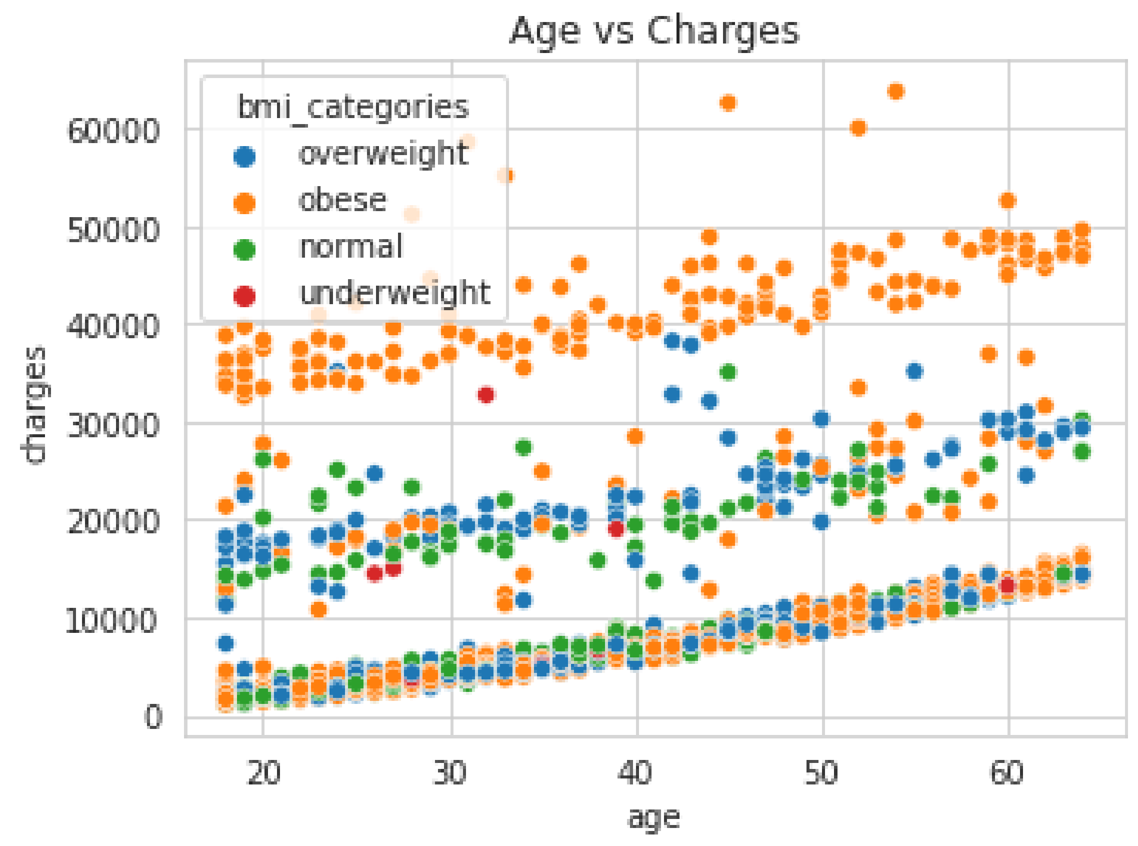
- charges and age have the highest correlation.
- Age has no outliers.
- smoker has higher cost.
- our dataset are mostly non smokers but they have higher BMI than smoker.

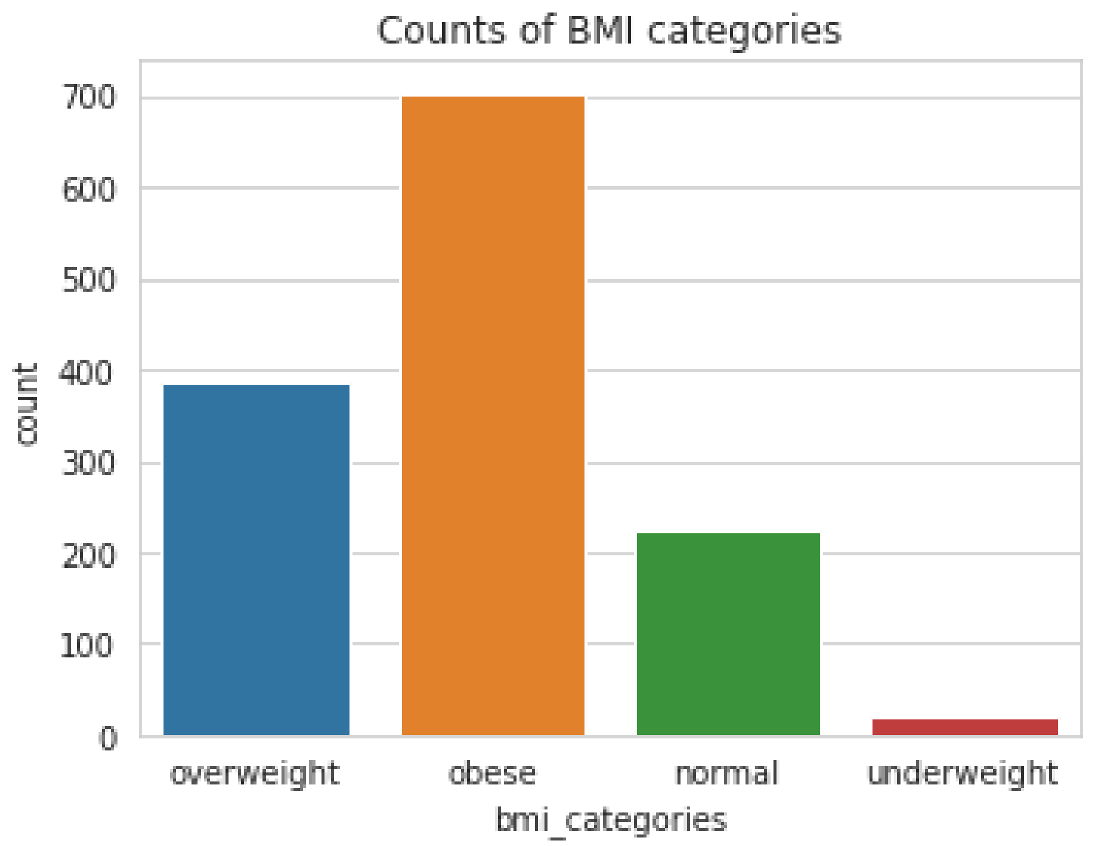





```python
conditions = [
    (df['bmi'] <= 18.5),
    (df['bmi'] >= 18.5) & (df['bmi'] <= 24.986),
    (df['bmi'] >= 25) & (df['bmi'] <= 29.926),
    (df['bmi'] >= 30)
    ]

values = ['underweight', 'normal', 'overweight', 'obese']

df['bmi_categories'] = np.select(conditions, values)
df.head()
```

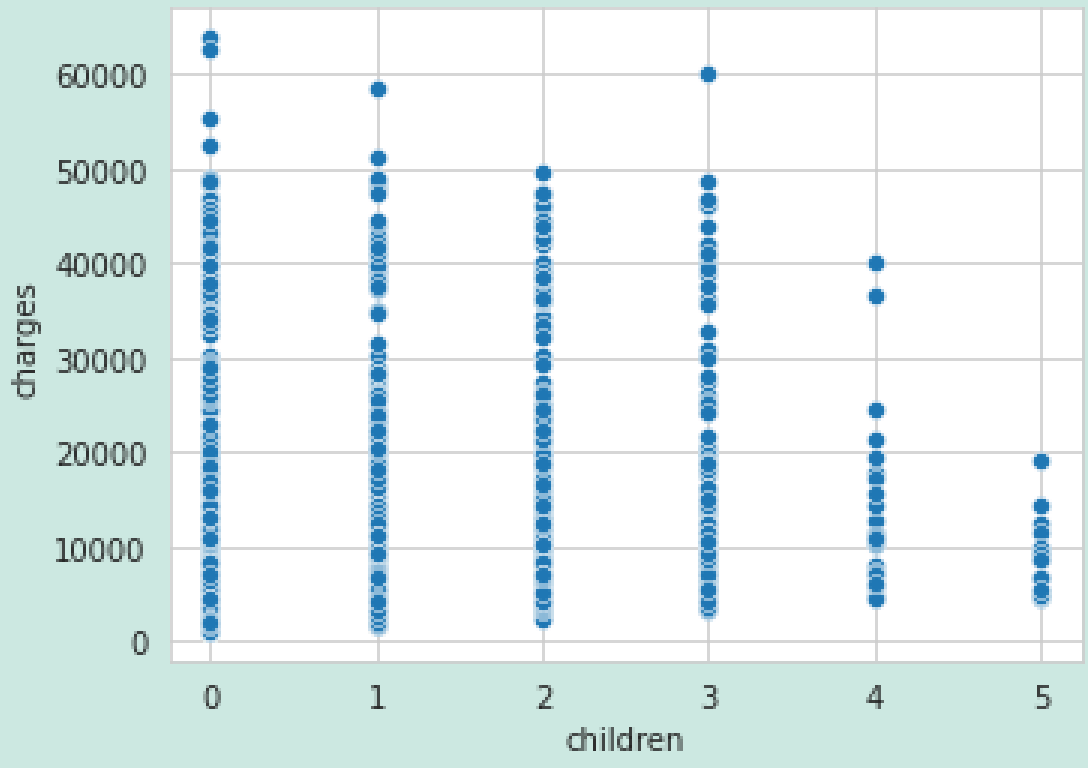
	age	sex	bmi	children	smoker	region	charges	zscore	bmi_categories
0	19	female	27.900	0	yes	southwest	16884.92400	-0.453151	overweight
1	18	male	33.770	1	no	southeast	1725.55230	0.509431	obese
2	28	male	33.000	3	no	southeast	4449.46200	0.383164	obese
3	33	male	22.705	0	no	northwest	21984.47061	-1.305043	normal
4	32	male	28.880	0	no	northwest	3866.85520	-0.292447	overweight

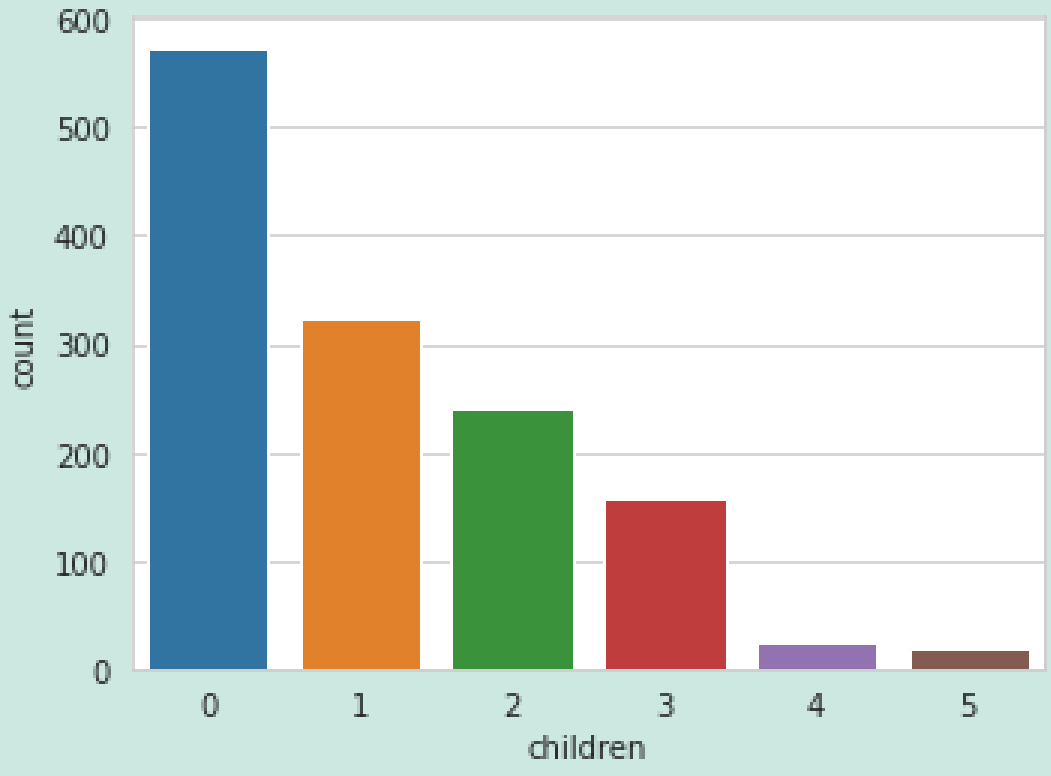


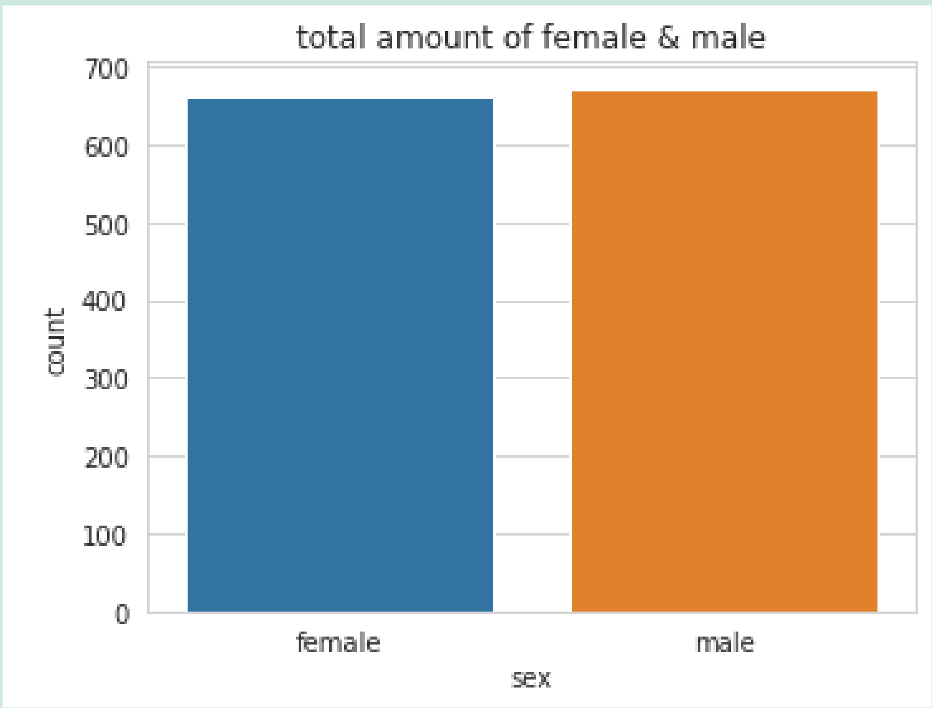


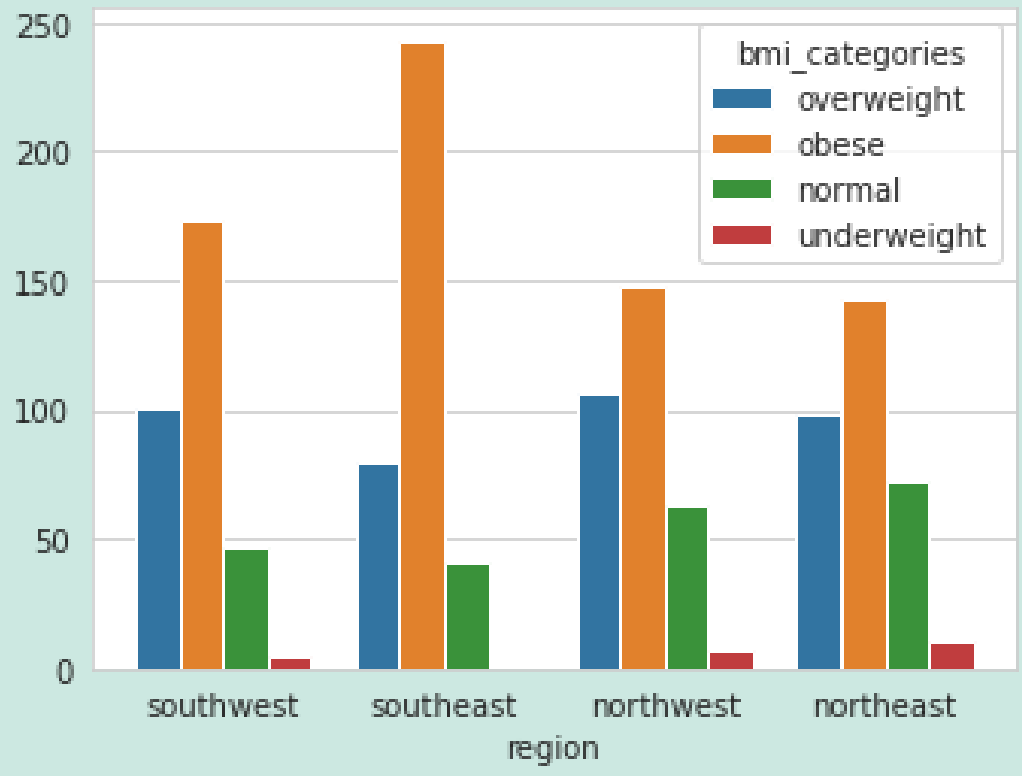
Finding :

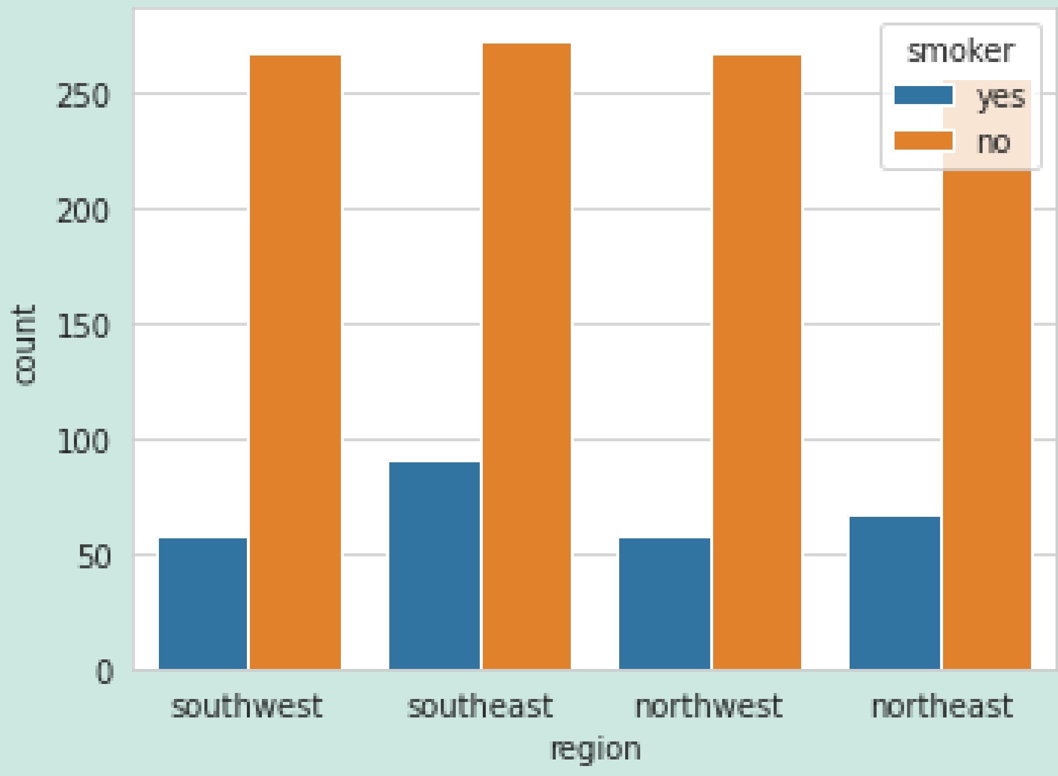
- BMI has a normal distribution with outliers
- most people in our data is obese and they tend to pay more for the medical cost











Finding :

- people with no children has higher medical cost.
- most peoplein each region are non smoker and obese.

Data Preprocessing

Removing outliers using Z score

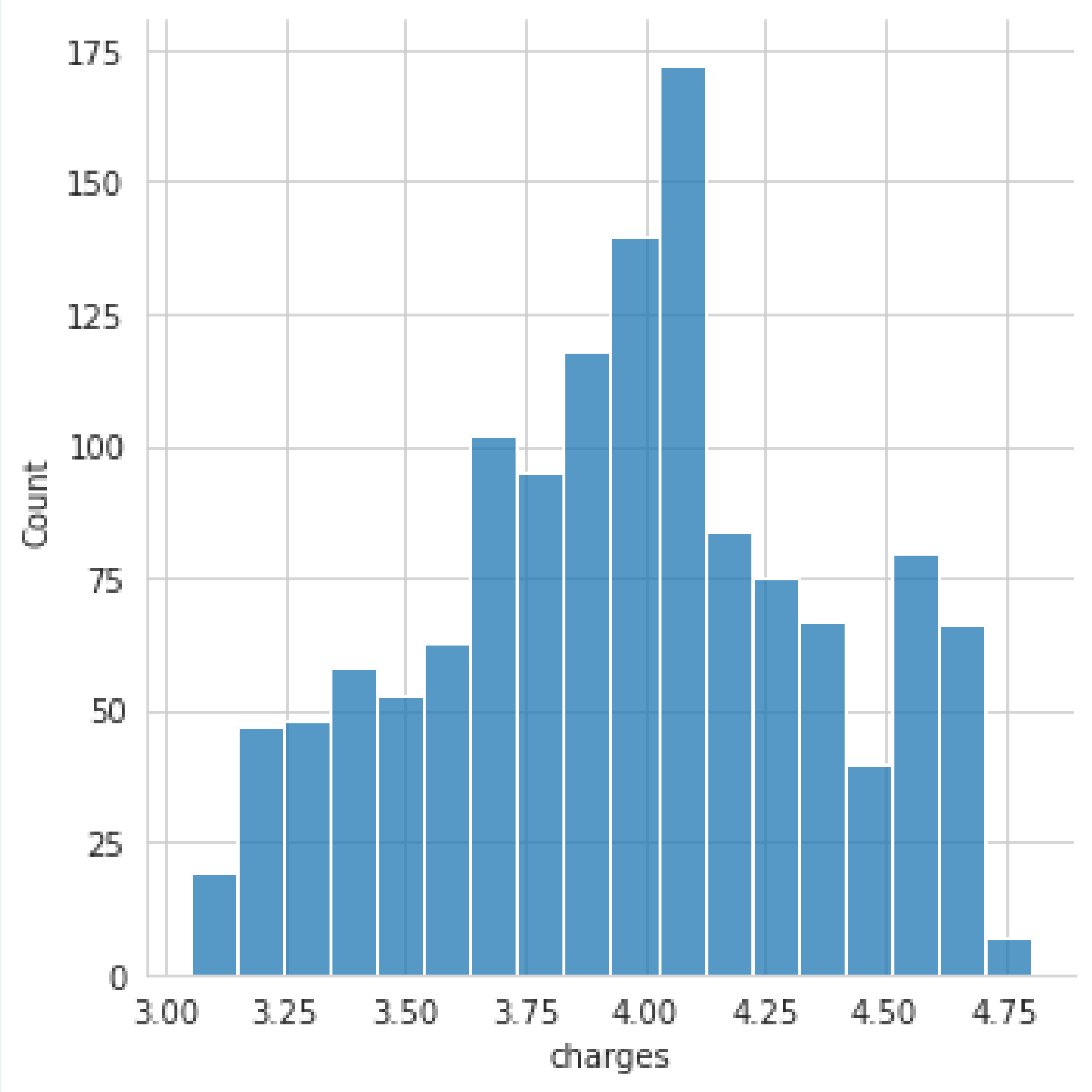
we using Z score for bmi because bmi follow the normal distribution and Zscore can quantify the unusualness of an observation in our data.

Log transformation for data target

Because charges have a right distribution and a lot of the outliers can't be filtered out so we use log transformation for helps reducing skewness.

Label encoding

Implement label encoding for categorical variable like sex, smoker, region and bmi_categories



Log transformation for target.

Modeling

- Separate the data into train and test set:

```python
y= df['charges']
X = df.drop(['charges'], axis = 1)


X_train, X_test, y_train, y_test = train_test_split(X,y, test_size= 0.2, random_state = 42)
X_train.shape, X_test.shape
```

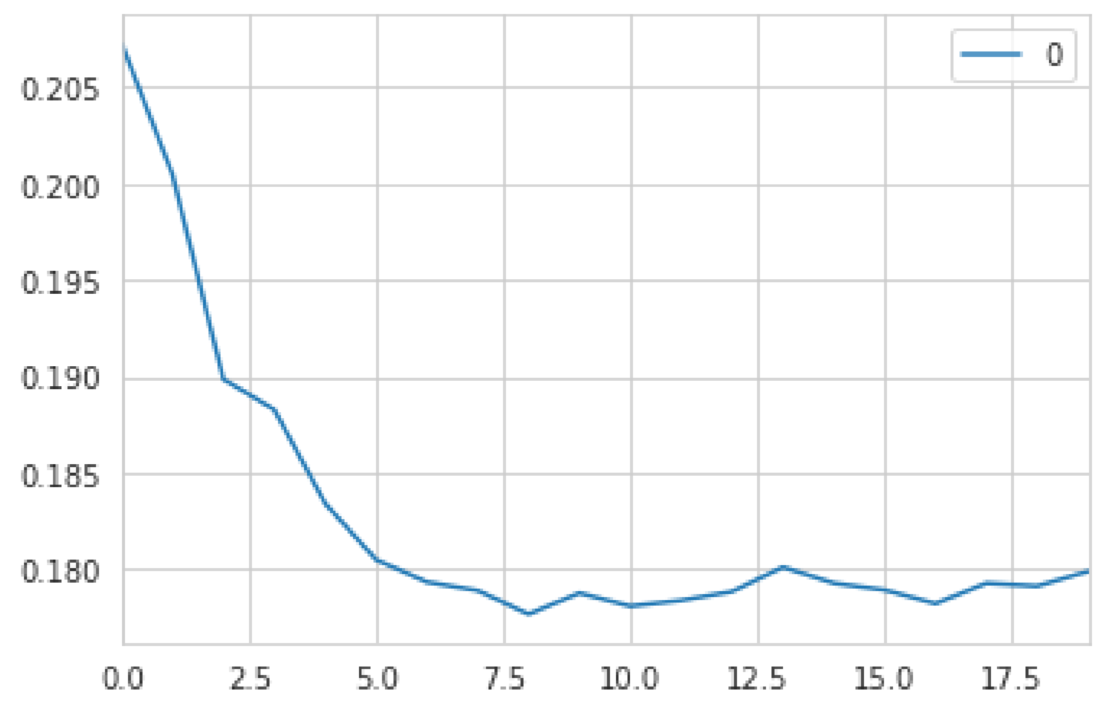
```
((1067, 7), (267, 7))
```

```python
from sklearn import neighbors

rmse_val = []
for K in range(20):
    K = K+1
    steps = [('scaler', StandardScaler()),
             ('knn', KNeighborsRegressor(n_neighbors = K))]
    model = Pipeline(steps)

    model.fit(X_train, y_train)
    pred=model.predict(X_test)
    error = sqrt(mean_squared_error(y_test,pred))
    rmse_val.append(error)
#     print('RMSE value for k= ' , K , 'is:', error)

curve = pd.DataFrame(rmse_val) #elbow curve
curve.plot()
```

- Model implementation :

we use pipelines for modeling and standard scaling for data scale.

1. Liniear regression
2. Polynomial regression (degree = 4)
3. Knn (n_neighbors = 8)
4. Decision tree (max_depth = 3)



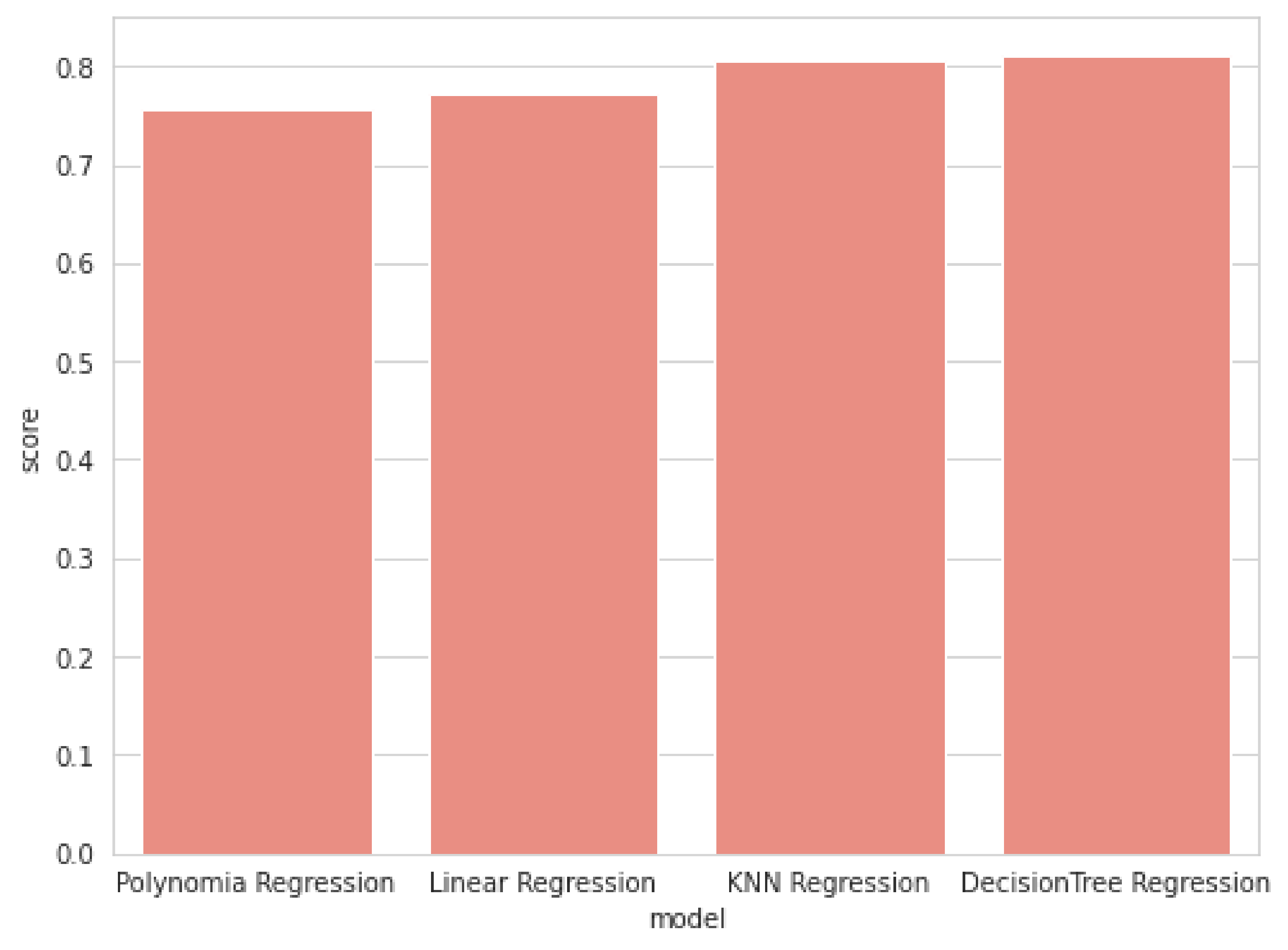
Find the best K for KNN

Models comparison

```
score = [['Linear Regression', linreg_score, mse_linreg, rmse_linreg]
         ['Polynomia Regression', poly_score, mse_poly, rmse_poly],
         ['KNN Regression', knn_score, mse_knn, rmse_knn],
         ['DecisionTree Regression', dt_score, mse_dt, rmse_dt]]

models = pd.DataFrame(score)
models.columns = ['model', 'score', 'MSE', 'RMSE']
print (models,'\n')

plt.figure(figsize=(8,6))
sns.barplot(x = 'model',
            y = 'score',
            data = models,
            color = 'salmon',
            order=models.sort_values('score').model)
plt.show()
```

```
                     model   score       MSE      RMSE
0        Linear Regression  0.7721  0.037439  0.193491
1     Polynomia Regression  0.7552  0.040222  0.200554
2           KNN Regression  0.8052  0.031998  0.178879
3  DecisionTree Regression  0.8105  0.031124  0.176421
```

Decsion tree has high score with small MSE or RMSESo we'll take decision tree for our model!



Tuning hyperparameters & saving model.

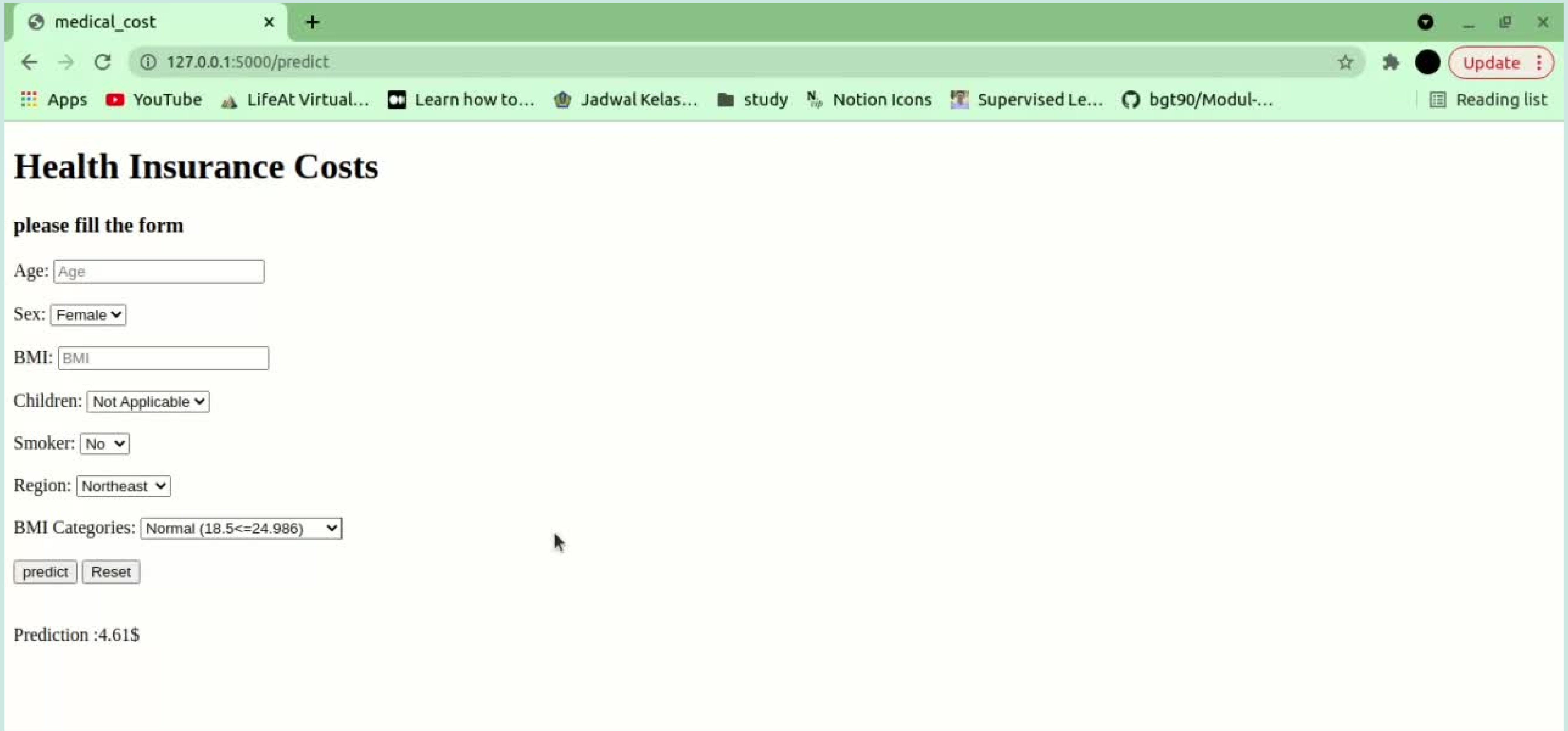
After looking for best hyperparameters with 5 k-fold validation and gridsearch then training the dataset and tsave our model with pickle

```
steps = [('scaler', StandardScaler()),
         ('dt', DecisionTreeRegressor(criterion = 'friedman_mse',
                                      max_depth =  4,
                                      max_features = None,
                                      min_samples_leaf =  4,
                                      min_weight_fraction_leaf =  0.1,
                                      splitter =  'best'))]
pipe = Pipeline(steps)
pipe.fit(X_train, y_train)

y_pred = pipe.predict(X_test)

score = pipe.score(X_test, y_test)
print('score : ', round(score, 4)*100)
print('MSE :', mean_squared_error(y_test, y_pred))
print('RMSE :', np.sqrt(mean_squared_error(y_test,y_pred)))
```

```
score :  80.4
MSE : 0.03220440783374481
RMSE : 0.17945586597752888
```

```
[ ] #final model
    tuning.best_estimator_.fit(X_train,y_train)
    filename = 'insurance.pkl'
    pickle.dump(tuning.best_estimator_,open(filename,'wb'))
```

Deploying a machine learning model to the web



https://www.canva.com/design/DAEvqbq4EI8/LnYLF7sbpefaRbexYwb77A/view?utm_content=DAEvqbq4EI8&utm_campaign=designshare&utm_medium=link&utm_source=publishsharelink

Contact Information

FULL CODE ON

https://github.com/ninaaulia

EMAIL ADDRESS

ninaaulia6652@gmail.com

WEBSITE

https://nninaulia.medium.com/