Task Report: Medical Insurance Cost Prediction

1. Overview

The main goal of this analysis is to predict individual medical costs billed by health insurance based on various factors such as age, sex, BMI, number of children, smoking status, and region. The dataset is derived from the book *Machine Learning with R* by Brett Lantz.

Key Objectives:

- Perform exploratory data analysis (EDA).
- Build and evaluate multiple regression models to predict insurance charges.
- Compare model performance and select the best-performing model.
- Fine-tune the best-performing model.

2. Exploratory Data Analysis (EDA)

Dataset Information:

- **Dataset Size:** 1338 rows and 7 columns.
- **Features:** Age, sex, BMI, number of children, smoker status, region, and charges.

```
df.info()
```

```
<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
```

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
age	1338.0	39.207025	14.049960	18.0000	27.00000	39.000	51.000000	64.00000
bmi	1338.0	30.663397	6.098187	15.9600	26.29625	30.400	34.693750	53.13000
children	1338.0	1.094918	1.205493	0.0000	0.00000	1.000	2.000000	5.00000
charges	1338.0	13270.422265	12110.011237	1121.8739	4740.28715	9382.033	16639.912515	63770.42801

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

```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

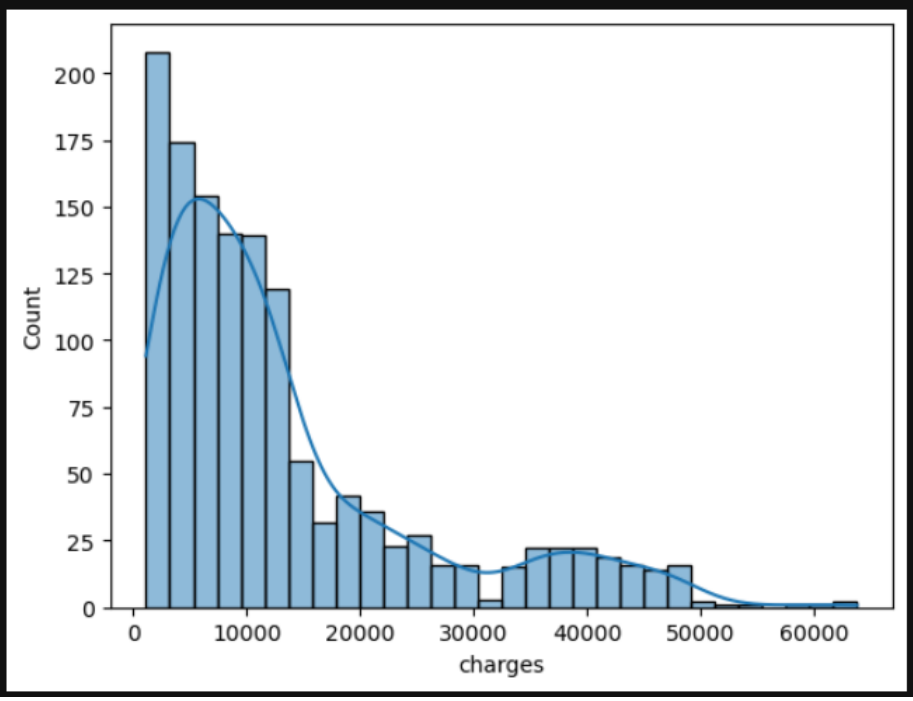
- **No missing data**: The dataset is complete with no null values.

Target Variable: charges

The target variable charges represent the medical costs billed by health insurance. The variable is continuous, and the distribution is right-skewed, which is common for cost-related data.

```
[28]: sns.histplot(data=df_viz, x='charges', kde=True)

C:\Users\Vivek.Nakka\AppData\Local\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1
moved in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
[28]: <Axes: xlabel='charges', ylabel='Count'>
```



Categorical Variables

Several categorical variables, such as sex, smoker, and region, were encoded to numerical values for modeling purposes.

```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
```

```
# Apply LabelEncoder to categorical columns
df['sex'] = label_encoder.fit_transform(df['sex'])  # female -> 0, male -> 1
df['smoker'] = label_encoder.fit_transform(df['smoker'])  # smoker yes -> 1, no -> 0
```

Categorical Variable Counts:

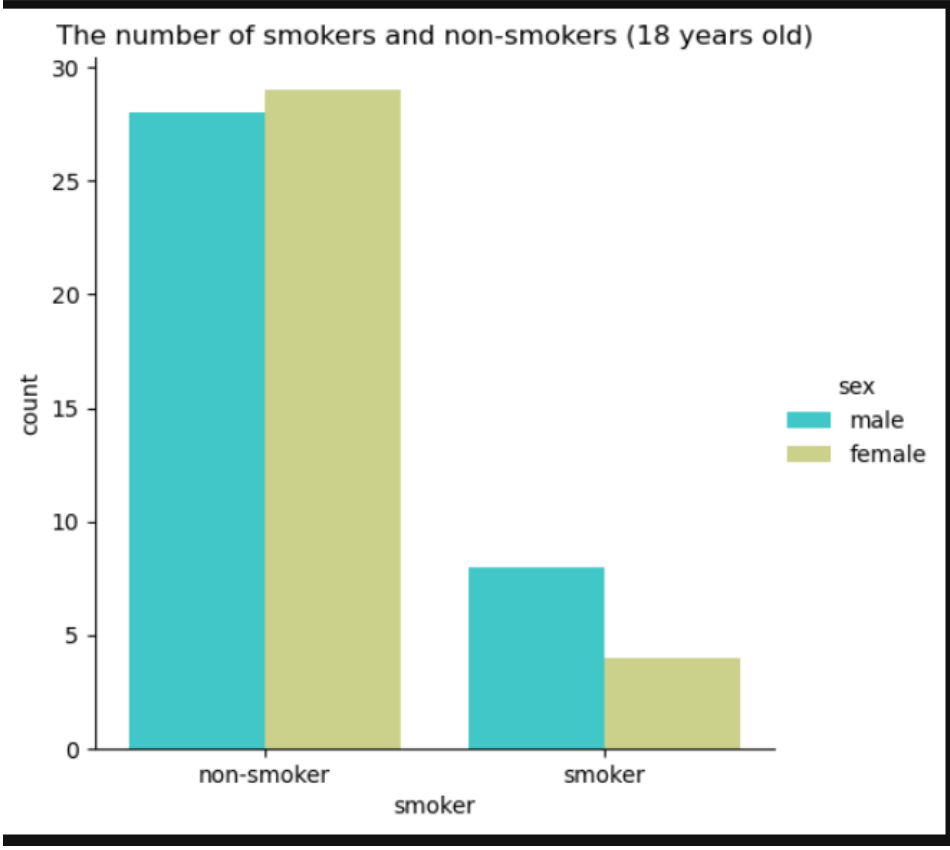
- sex: Male: 51%, Female: 49%
- smoker: Yes: 20%, No: 80%
- region: Distributed across four regions in the US (Northeast, Southeast, Northwest, Southwest).

3. Visualizations

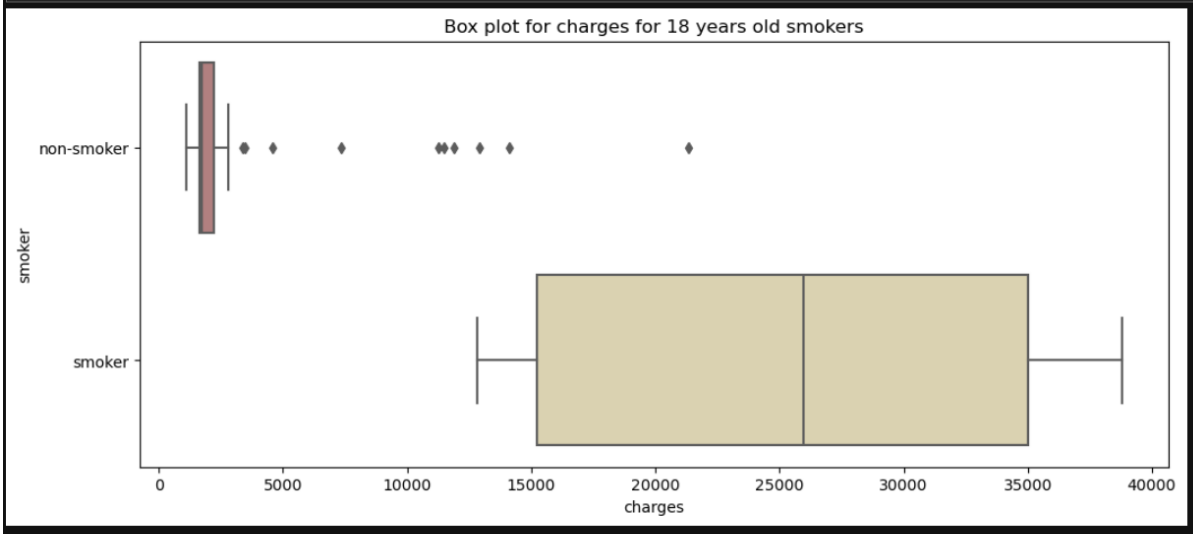
Relationship Between Smoking and Medical Charges

- **Insight:** Smoking significantly increases medical charges. Even at a young age (18 years), smokers incur higher medical costs than non-smokers.

```
sns.catplot(x="smoker", kind="count", hue='sex', palette="rainbow", data=df_viz[df_viz['age'] == 18])
plt.title("The number of smokers and non-smokers (18 years old)")
plt.show()
```



```
plt.figure(figsize=(12,5))
plt.title("Box plot for charges for 18 years old smokers")
sns.boxplot(y="smoker", x="charges", data=df_viz[df_viz['age'] == 18], orient="h", palette="pink")
plt.show()
```

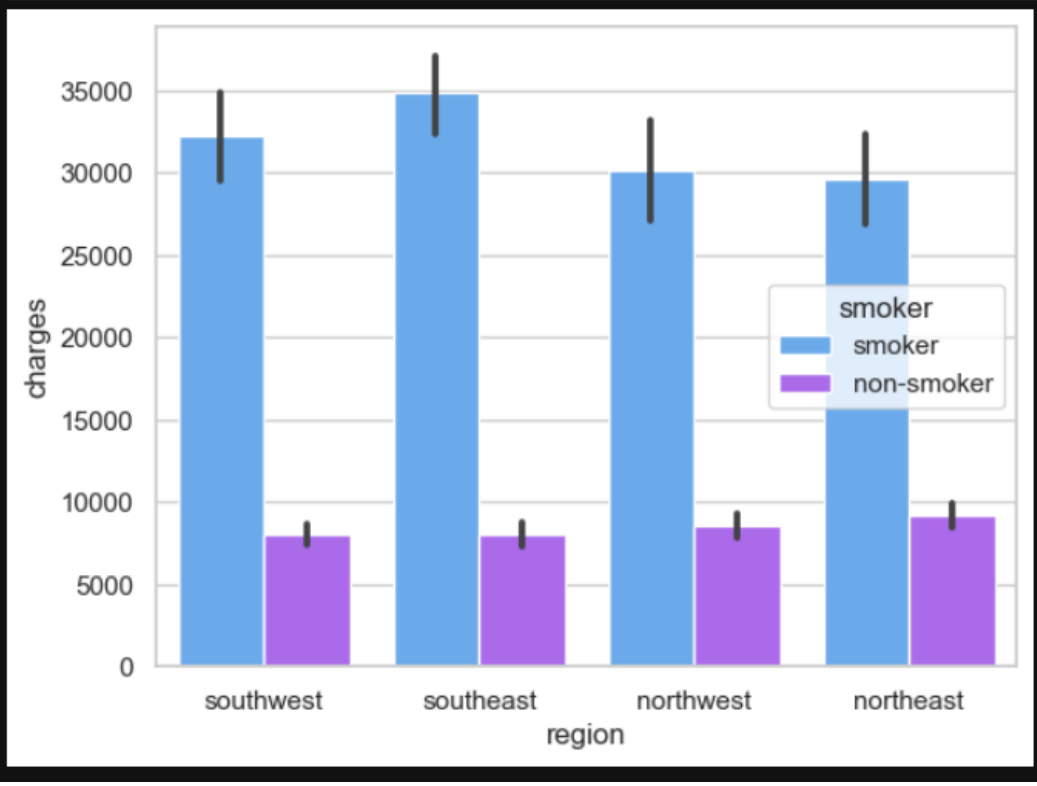


Oh. As we can see, even at the age of 18 smokers spend much more on treatment than non-smokers. Among non-smokers we are seeing some " tails." I can assume that this is due to serious diseases or accidents. Now let's see how the cost of treatment depends on the age of smokers and non-smokers patients.

Medical Charges by Region

- **Insight:** The highest medical charges are observed in the Southeast, while the lowest charges are in the Southwest.

```
[38]: ax = sns.barplot(x='region', y='charges', hue='smoker', data=df_viz, palette='cool')
```



4. Model Building

Data Preparation

- **Feature Selection:** The dataset was split into features (X) and the target variable (y). The categorical columns such as sex, smoker, and region were label-encoded for model training.

```
from sklearn.model_selection import train_test_split
x = df.drop(['charges'], axis = 1)
y = df['charges']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

Model Selection

Several regression models were tested, including:

- **Linear Regression (LR)**
- **Random Forest Regressor (RF)**
- **Decision Tree Regressor (DT)**
- **Gradient Boosting Regressor (GBR)**
- **K-Neighbors Regressor (KNN)**
- **Support Vector Regressor (SVR)**

Model Comparison

For each model, training and testing accuracies were evaluated using R-squared, Mean Squared Error (MSE), and Mean Absolute Error (MAE).

```
models = {
    "LR": LinearRegression(),
    "RF": RandomForestRegressor(n_estimators=100, max_depth=7),
    "DT": DecisionTreeRegressor(),
    "GradientBoosting": GradientBoostingRegressor(n_estimators=100, max_depth=7),
    "KNN": KNeighborsRegressor(),
    "SVR": SVR()
}

for name, model in models.items():
    print(f'Training Model {name} \n-------------------------------------------')
    model.fit(x_train, y_train)
    y_pred = model.predict(x_test)
    print(f'Score is {model.score(x_test, y_test)}')


    # Use appropriate regression metrics
    print(f'Training R-squared: {r2_score(y_train, model.predict(x_train))}')
    print(f'Testing R-squared: {r2_score(y_test, y_pred)}')

    print(f'Mean Squared Error: {mean_squared_error(y_test, y_pred)}')
    print(f'Mean Absolute Error: {mean_absolute_error(y_test, y_pred)}')
```

Model	Training R-squared	Testing R-squared	Mean Squared Error (MSE)	Mean Absolute Error (MAE)
Linear Regression	0.7417	0.7833	33,635,210.43	4,186.51
Random Forest	0.9278	0.8768	19,130,167.40	2,428.20
Decision Tree	0.9983	0.7433	39,858,781.72	2,898.69
Gradient Boosting	0.9928	0.8419	24,543,702.50	2,609.51
K-Nearest Neighbors	0.3938	0.1445	132,814,646.70	7,953.21
Support Vector Reg.	-0.0977	-0.0723	166,474,492.54	8,592.79

Best Model: Random Forest Regressor

- The **Random Forest Regressor** outperformed all other models with a testing R-squared score of 0.8768 and the lowest Mean Squared Error (19,130,167.40).

5. Model Fine-Tuning

Hyperparameter Tuning: Max Depth

The performance of the Random Forest model was optimized by tuning the maximum depth of the trees. Various depths from 1 to 8 were tested to identify the optimal depth.

```
max_depth_values = [1,2,3,4,5,6,7,8]
train_accuracy_values =[]
for max_depth_val in max_depth_values:
    model = RandomForestRegressor(max_depth=max_depth_val,random_state = 2)
    model.fit(x_train, y_train)
    y_pred =model.predict(x_train)
    acc_train=model.score(x_test,y_test)
    train_accuracy_values.append(acc_train)
```

```
train_accuracy_values
```

```
[0.6608048922770777,
 0.8417933495550463,
 0.8675746825596274,
 0.872677170478934,
 0.8753239247794342,
 0.8755309603734038,
 0.8729311378096285,
 0.8711685231482463]
```

Final Model Evaluation

The final Random Forest model with a depth of 6 was selected. The R-squared score on the test set was 0.872, and the MSE was 19845744.977, indicating strong predictive power.

```
forest_train_pred = final_model.predict(x_train)
forest_test_pred = final_model.predict(x_test)

print('MSE train data: %.3f, MSE test data: %.3f' % (
mean_squared_error(y_train,forest_train_pred),
mean_squared_error(y_test,forest_test_pred)))
print('R2 train data: %.3f, R2 test data: %.3f' % (
r2_score(y_train,forest_train_pred),
r2_score(y_test,forest_test_pred)))
```

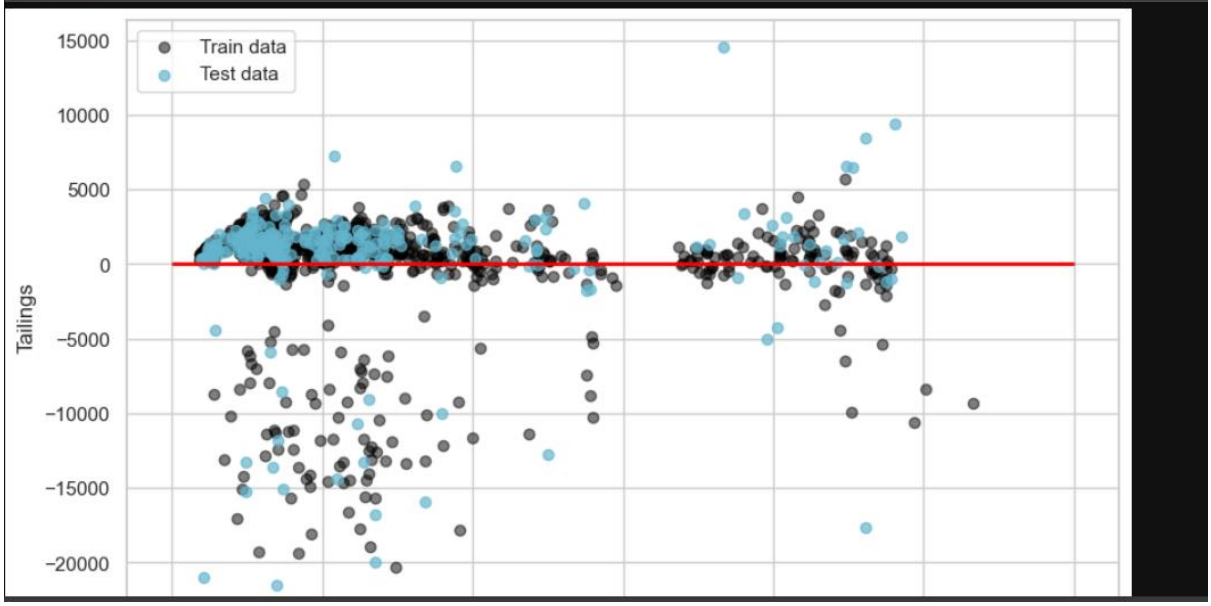
```
MSE train data: 13740918.931, MSE test data: 19845744.977
R2 train data: 0.905, R2 test data: 0.872
```

6. Visualization of Model Predictions

A scatter plot of predicted values against residuals (actual - predicted) shows that the model's residuals are randomly scattered, indicating a good fit without major bias.

```
plt.figure(figsize=(10,6))

plt.scatter(forest_train_pred,forest_train_pred - y_train,
          c = 'black', marker = 'o', s = 35, alpha = 0.5,
          label = 'Train data')
plt.scatter(forest_test_pred,forest_test_pred - y_test,
          c = 'c', marker = 'o', s = 35, alpha = 0.7,
          label = 'Test data')
plt.xlabel('Predicted values')
plt.ylabel('Tailings')
plt.legend(loc = 'upper left')
plt.hlines(y = 0, xmin = 0, xmax = 60000, lw = 2, color = 'red')
plt.show()
```



7. Conclusion

- **Best Model:** The **Random Forest Regressor** performed the best with the highest R-squared value and lowest MSE.
- **Smoker Impact:** Smoking significantly increases medical costs, even at younger ages.
- **Regional Impact:** The Southeast region has the highest medical costs, while the Southwest has the lowest.