Random Forest Regressor

In [ ]:
```
# Import libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

In [ ]:
```
# Load data
df = sns.load_dataset('car_crashes')
df.head()
```

Out[ ]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08	AL
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93	AK
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35	AZ
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39	AR
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63	CA

In [ ]:
```
# Drop last columns
df.drop(columns='abbrev', inplace=True)
```

In [ ]:
```
df.head()
```

Out[ ]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63

Let's predict the insurance losses `ins_losses` based on the given samples.

In [ ]:
```
# Split data
X = df.iloc[:, :-1]
y = df.iloc[:, -1:]
```

In [ ]:
```
X.shape, y.shape
```

Out[ ]:
```
((51, 6), (51, 1))
```

In [ ]:
```
# Train model and predict on unknown data
model = RandomForestRegressor(n_estimators=100)
model.fit(X, y)
model.predict([[18.7, 7.784, 5.601, 18.778, 15.010, 784.25]])
```

```
C:\Users\awon\AppData\Local\Temp/ipykernel_4784/968989654.py:3: DataConversionWarning: A column-vector y was passed w
hen a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  model.fit(X, y)
c:\users\awon\miniconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but
RandomForestRegressor was fitted with feature names
  warnings.warn(
```

Out[ ]:
```
array([137.8755])
```

What is n_estimators?

n_estimators is the hyperparameter of Random Forest Classifier/Regressor which means **how many number to trees are in the forest**. This can be tuned to improve model results.

Let's split the data into 80-20% of training and test data and check the models performance.

In [ ]:
```
# 80/20 split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2,
                                                    random_state=42)
```

In [ ]:
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[ ]:
```
((40, 6), (11, 6), (40, 1), (11, 1))
```

In [ ]:
```
# Train model and check results
rf_reg = RandomForestRegressor(n_estimators=100, random_state=0)
rf_reg.fit(X_train, y_train)

# R2 score
r2_score = rf_reg.score(X_test, y_test)
print(f'R-squared: {r2_score:.3f}')
```

```
C:\Users\awon\AppData\Local\Temp/ipykernel_4784/2823997653.py:3: 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().
  rf_reg.fit(X_train, y_train)
R-squared: 0.491
```

In [ ]:
```
# Make predictions and accuracy check
y_pred = rf_reg.predict(X_test)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
print(f'RMSE: {rmse:.3f}')
```

```
RMSE: 15.777
```