In [1]:

```
# Numpy for matrix operations
import numpy as np

# Pandas files input operations
import pandas as pd

#sklearn imports
# Fitting Random Forest Regression to the dataset

from sklearn.ensemble import RandomForestRegressor

# Using Skicit-learn to split data into training and testing sets
from sklearn.model_selection import train_test_split
```

In [2]:

```
kMeansOut = pd.read_csv("dataSet.csv")
kMeansOut = kMeansOut.drop(['Unkown','SESSION_ID','PLAN_ID','SQL_TEXT_TYPE','newlin
kMeansOut.head()
```

Out[2]:

	DATABASE_NAME	USER_NAME	TOTAL_SECONDS	SNIPPETS	THROUGH_PUT_ROWS	T
0	CIW_STAGE	CIW_ETLUSER	0	1	0	
1	CIW_STAGE	CIW_ETLUSER	4	4	0	
2	CIW_STAGE	CIW_ETLUSER	0	1	0	
3	CIW_STAGE	CIW_ETLUSER	0	1	0	
4	CIW_STAGE	CIW_ETLUSER	0	1	0	

In [3]:

```
X = pd.get_dummies(kMeansOut[kMeansOut.columns[0:7]])
X.head()
```

Out[3]:

	TOTAL_SECONDS	SNIPPETS	THROUGH_PUT_ROWS	THROUGH_PUT_SIZE	SQL_TEXT_HA
0	0	1	0	0	15806264
1	4	4	0	0	14638245
2	0	1	0	0	19536645
3	0	1	0	0	8450982
4	0	1	0	0	8838048

5 rows × 56 columns

In [4]:

```
Y = kMeansOut[['Clusters']]
Y.head()
```

Out[4]:

	Clusters
0	4
1	4
2	4
3	4
4	4

In [5]:

```
# Split the data into training and testing sets
train_features, test_features, train_labels, test_labels = train_test_split(X, Y, t

test_labels = np.array(test_labels)
```

In [6]:

```
print('Training Features Shape:', train_features.shape)
print('Training Labels Shape:', train_labels.shape)
print('Testing Features Shape:', test_features.shape)
print('Testing Labels Shape:', test_labels.shape)
```

```
Training Features Shape: (36771, 56)
Training Labels Shape: (36771, 1)
Testing Features Shape: (12257, 56)
Testing Labels Shape: (12257, 1)
```

In [7]:

```
# Instantiate model with 10000 decision trees
rf = RandomForestRegressor(n_estimators = 10000, random_state = 42)
# Train the model on training data
rf.fit(train_features, train_labels)
```

```
/usr/local/lib/python3.5/dist-packages/ipykernel_launcher.py:4: DataC
onversionWarning: A column-vector y was passed when a 1d array was ex
pected. Please change the shape of y to (n_samples,), for example usi
ng ravel().
  after removing the cwd from sys.path.
```

Out[7]:

```
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=Non
e,
                      max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=N
one,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=1000
0,
                      n_jobs=None, oob_score=False, random_state=42,
verbose=0,
                      warm_start=False)
```

In [8]:

```
# Use the forest's predict method on the test data
predictions = rf.predict(test_features)
predictions = np.array(predictions)
```

In [9]:

```
accuracy_test = 100 - np.mean(np.abs(predictions - test_labels)) * 100

print("test accuracy: {} %".format(accuracy_test))
```

```
test accuracy: 94.59893069659576 %
```

In [11]:

```
from sklearn.externals import joblib

# Save the model as a pickle in a file
joblib.dump(rf, 'randomForest.pkl')
```

Out[11]:

```
['randomForest.pkl']
```

In [ ]:

```

```