Project : Youtube Adview Prediction

6. Use linear regression, support vector regressor, random forest and for training and get errors.

Different machine learning models can be used for training out of which we can choose whichever gives the best result. We will use the scikitlearn libraries for importing these models and the training them use the fit method and providing necessary labelled data (input and output). We are optimising for mean square error here because it's a regression problem after all. The metrics that we can use for us to compare different model can be mean square error and mean absolute error.

- [ ] Import scikitlearn library
- [ ] Import the model and define
- [ ] Use .fit method with data as arguments to train
- [ ] Calculate errors

```
# Evaluation Metrics
from sklearn import metrics
def print_error(X_test, y_test, model_name):
    prediction = model_name.predict(X_test)
    print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, prediction))
    print('Mean Squared Error:', metrics.mean_squared_error(y_test, prediction))
    print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, prediction)))


# Linear Regression
from sklearn import linear_model
linear_regression = linear_model.LinearRegression()
linear_regression.fit(X_train, y_train)
print_error(X_test,y_test, linear_regression)

# Support Vector Regressor
from sklearn.svm import SVR
supportvector_regressor = SVR()
supportvector_regressor.fit(X_train,y_train)
print_error(X_test,y_test, linear_regression)
```

7. Use Decision Tree Regressor and Random Forest Regressors.

Another class of machine learning algorithms include decision trees and random forests. We can import these models from sklearn.tree and then again use the .fit function. We need to give appropriate hyper parameters for them. These are something we can experiment with to achieve better results.

- [ ] Import models
- [ ] Assign hyperparameters for random forest
- [ ] Train the models
- [ ] Calculate errors

You can also use ensemble methods such as xgboost to get a better performance (bonus)

```
# Decision Tree Regressor
from sklearn.tree import DecisionTreeRegressor
decision_tree = DecisionTreeRegressor()
decision_tree.fit(X_train, y_train)
print_error(X_test,y_test, decision_tree)

# Random Forest Regressor
from sklearn.ensemble import RandomForestRegressor
n_estimators = 200
max_depth = 25
min_samples_split=15
min_samples_leaf=2
random_forest = RandomForestRegressor(n_estimators = n_estimators, max_depth = max_depth, min_samples_split=min_samples_spli
random_forest.fit(X_train,y_train)
print_error(X_test,y_test, random_forest)
```