Stacking Regression

+ Code + Text

```
# from google.colab import files
# up = files.upload()
```

import dataset

```
import pandas as pd
df = pd.read_csv('df.csv')
df.head()
```

	f1	f2	f3	f4	T
0	16.5	202.0	865.500000	1880.0	50.000000
1	18.0	204.0	688.000000	1738.5	44.000000
2	18.0	203.0	583.666667	1470.0	66.666667
3	17.0	201.5	892.500000	1484.0	43.000000
4	31.5	218.0	1059.500000	2065.0	38.500000

cleaning

```
# clean the data
```

encoding

```
# encode the data
```

define x , y

```
import numpy as np
x = df[['f1', 'f2', 'f3']].values
y = df['T'].values
```

spliting

```
### finding best random state
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=42)
```

scaleing

```
# scaling depends on the base models used in the stacking ensemble.
# If any base model requires feature scaling, then the data should be scaled accordingly.
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler().fit(x_train)
x_train = sc.transform(x_train)
x_test = sc.transform(x_test)
```

Define base learners

```
from sklearn.ensemble import StackingRegressor
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR

base_learners = [
    ('lr', LinearRegression()),
    ('rf', RandomForestRegressor(n_estimators=100, random_state=1)),
    ('svr', SVR())
]
```

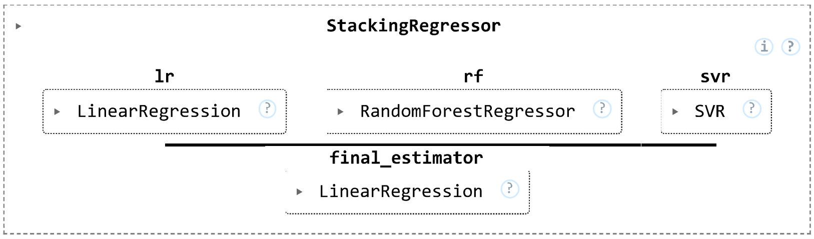
Initialize the Stacking Regressor

```
sr= StackingRegressor(
    estimators=base_learners,
    final_estimator=LinearRegression()
)
```

fit the model

```
### K-fold cross validation
```

```
sr.fit(x_train, y_train)
```



predict test data

```
yhat_test = sr.predict(x_test)
```

evaluate the model

```
from sklearn.metrics import r2_score
print("r2-score (train data): %0.4f" % r2_score(y_train, sr.predict(x_train)))
print("r2-score (test data): %0.4f" % r2_score(y_test, yhat_test))
```

```
r2-score (train data): 0.6894
r2-score (test data): 0.3418
```

```
from sklearn.metrics import mean_squared_error, mean_absolute_error
print(f"MSE (train data): {mean_squared_error(y_train, sr.predict(x_train))}")
print(f"MAE (train data): {mean_absolute_error(y_train, sr.predict(x_train))}")
print(f"MSE (test data): {mean_squared_error(y_test, yhat_test)}")
print(f"MAE (test data): {mean_absolute_error(y_test, yhat_test)}")
```

```
MSE (train data): 58.64858046241831
MAE (train data): 6.108113666632705
MSE (test data): 96.37577162614235
MAE (test data): 7.9064710609055
```

save the model

```
# import joblib
# joblib.dump(sr, 'sr_model.pkl')
```

load the model

```
# import joblib
# sr = joblib.load('sr_model.pkl')
```