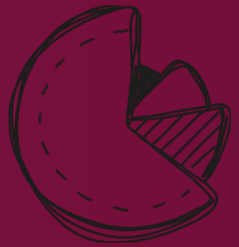
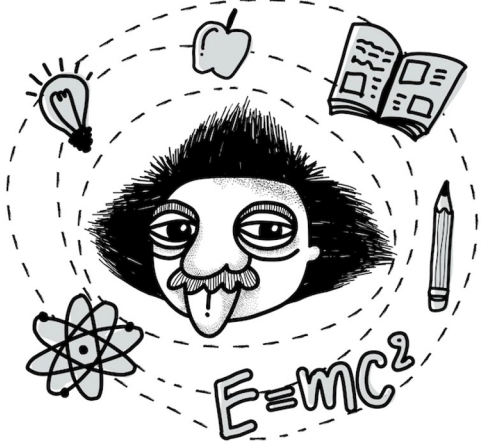
BASIL PRESENTATION

Sklearn Pipelines & SuperLearners

- Nidhi Jyani



Today we'll learn about...

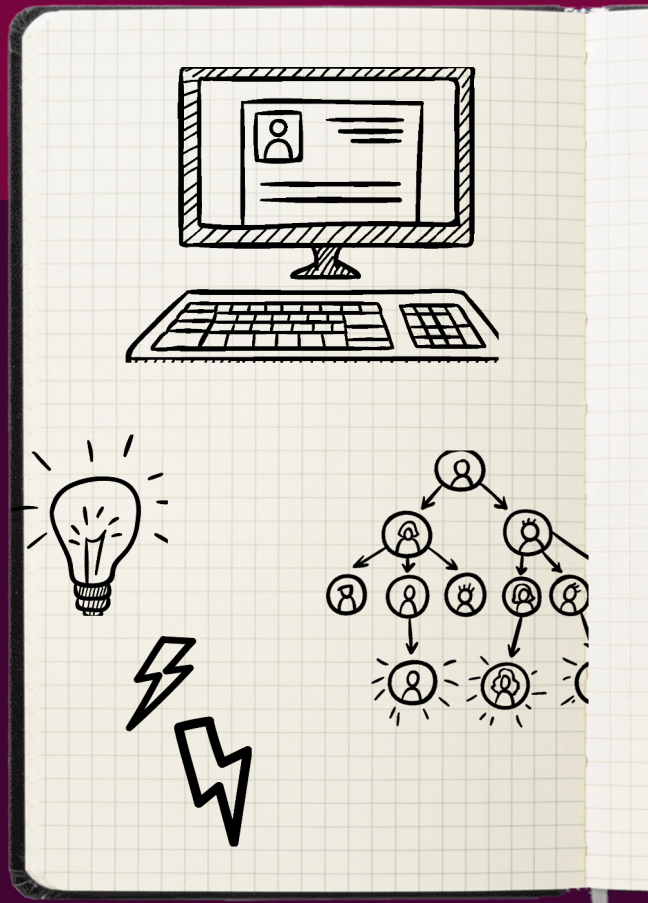


Sklearn pipelines

What are Superlearners

Implementation in Python

Sklearn : Pipeline



Benefits

- They make our workflow much easier to **read and understand**.
- They enforce the implementation and **order of steps** in our project.
- These in turn make our work much more **reproducible**.

sklearn.pipeline.Pipeline

class sklearn.pipeline.**Pipeline**(*steps*, *, *memory=None*, *verbose=False*)

```
>>> from sklearn.svm import SVC
>>> from sklearn.preprocessing import StandardScaler
>>> from sklearn.datasets import make_classification
>>> from sklearn.model_selection import train_test_split
>>> from sklearn.pipeline import Pipeline
>>> X, y = make_classification(random_state=0)
>>> X_train, X_test, y_train, y_test = train_test_split(X, y,
...                                                     random_state=0)
>>> pipe = Pipeline([('scaler', StandardScaler()), ('svc', SVC())])
>>> # The pipeline can be used as any other estimator
>>> # and avoids leaking the test set into the train set
>>> pipe.fit(X_train, y_train)
Pipeline(steps=[('scaler', StandardScaler()), ('svc', SVC())])
>>> pipe.score(X_test, y_test)
0.88
```

.predict
.score

Grid Search CV and Sklearn Pipelines

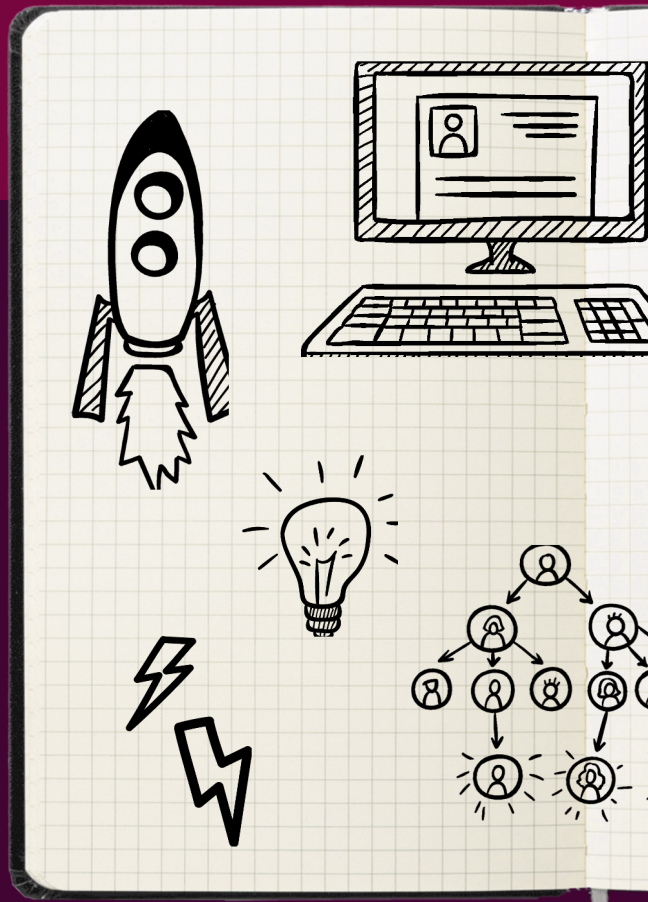
`sklearn.model_selection`.GridSearchCV

Hyperparameter Tuning

Cross Validation

Super Learners

Why use one ML algorithm when you could use all of them!!!



It’s all the same !!!!

Superlearning

- involves combining many individual statistical algorithms to create a new, single prediction algorithm that is expected to perform at least as well as any of the individual algor

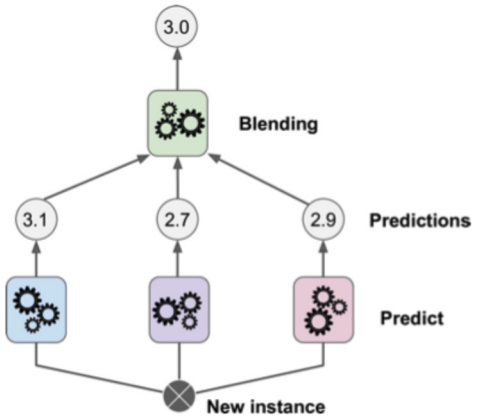


Figure: Illustration on a Stacking model.

Superlearning

The model should have (atleast) 2 layers:

- First layer - Multiple weak learners
- Second layer - A blender. A model to blend previous results to form final prediction.

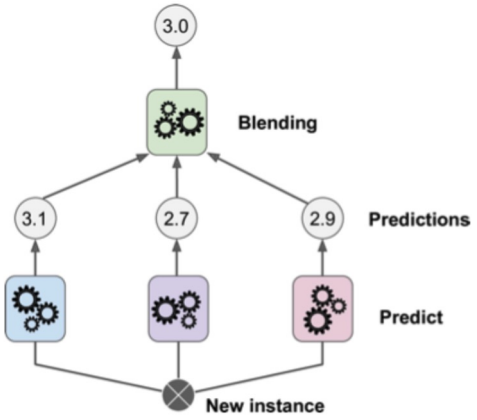


Figure: Illustration on a Stacking model.

Model setting

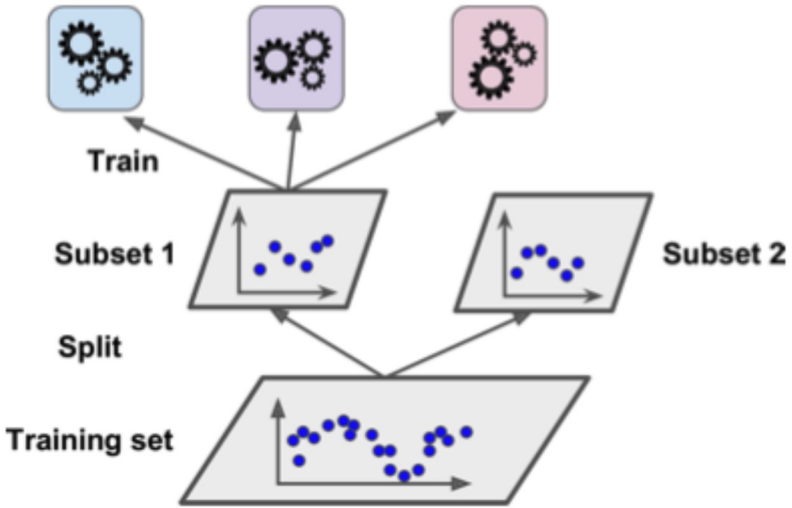


Figure: Training of base learners.

1. Split the data into training and testing set. Further split the training set into two equal parts (T1 and T2).

1. Using T1, fit different weak learners (they can be the same model with different hyperparameters)

Model setting

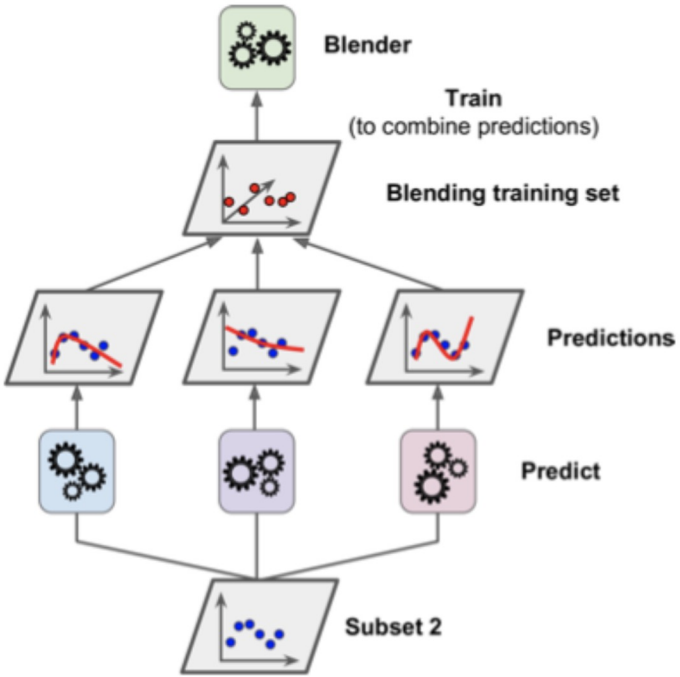


Figure: Training of blender.

3. Perform prediction on these weak/base learners using T2.

4. Train a blender, i.e. final regressor / classifier.

Evaluate model on **Test set**

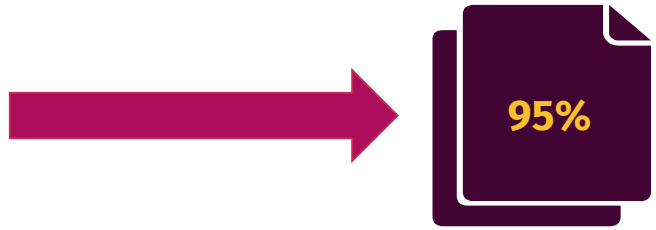
Demo Time

Results

Individual Model	Score
Support Vector Machine	94%
Logistic Regression	88%
Decision Tree	93%

Better Prediction Accuracy

Stacking Model



Improvements

1 Prediction Form

2 Model Diversity

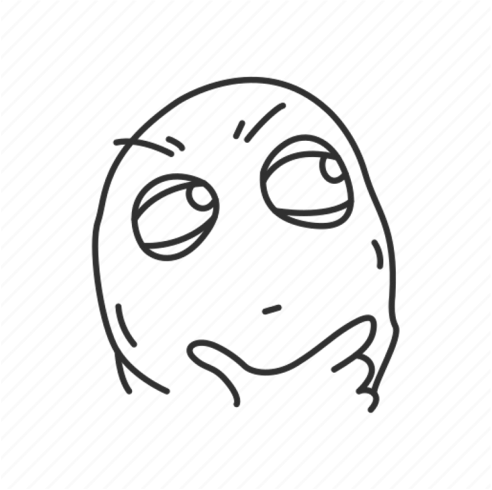
- Linear/Non-linear
- Input Transformation

Some limitations

- Little or no improvement
- Time-consuming
- Expensive to deploy and maintain
- Not useful with Small Data
- Interpretation complexity

“

Do you have any questions?



Thanks!

Any questions?

You can find me at:

- njyani@levi.com