Model Building Part....!

1. Splitting The Dataset Into The Training Set And Test Set & Applying K-Fold Cross Validation

What is Model Building?

- Model building in machine learning is the process of creating and training a mathematical representation that can make predictions or decisions based on input data

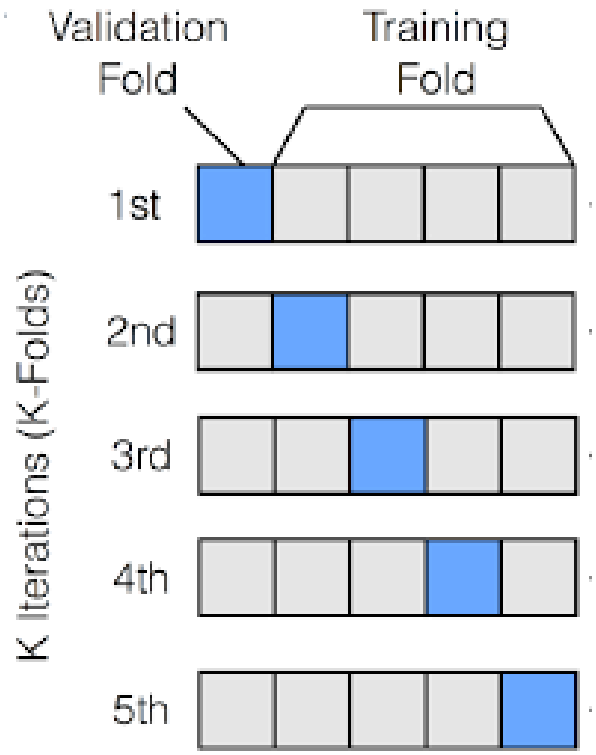
What is Splitting the dataset?

- Splitting dataset is a dividing the available data into two or more subsets for different purposes, typically training and testing.

1. Splitting The Dataset Into The Training Set And Test Set & Applying K-Fold Cross Validation

What is K-Fold Cross Validation?

- K-fold cross-validation is a technique used in machine learning to assess the performance of a model



Code and output

```
model_df={}
def model_val(model,X,y):
    X_train,X_test,y_train,y_test=train_test_split(X,y,
                                                   test_size=0.20,
                                                   random_state=42)
    model.fit(X_train,y_train)
    y_pred=model.predict(X_test)
    print(f"{model} accuracy is {accuracy_score(y_test,y_pred)}")

    score = cross_val_score(model,X,y,cv=5)
    print(f"{model} Avg cross val score is {np.mean(score)}")
    model_df[model]=round(np.mean(score)*100,2)
```

[39] ✓ 0.0s

```
model_df
```

[76] ✓ 0.0s

```
{LogisticRegression(): 80.48,
 SVC(): 79.39,
 DecisionTreeClassifier(): 72.34,
 RandomForestClassifier(): 79.03,
 GradientBoostingClassifier(): 76.86}
```

2. Logistic Regression

What is Logistic Regression?

- Logistic Regression is a statistical method that is used for building machine leaning models where the dependent variable is dichotomous.
- Logistic Regression is used to predict the categorical dependent variable.
- Its used when the prediction is categorical for example ‘yes’ or ‘no’, ‘true’ or ‘false’, ‘0’ or ‘1’.

Code and output

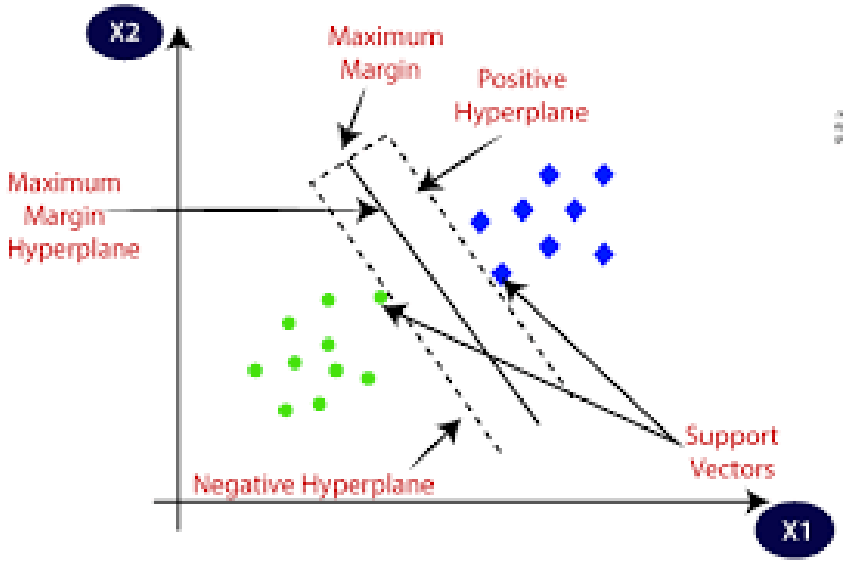
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model_val(model,X,y)
```

[40] ✓ 0.3s

```
LogisticRegression() accuracy is 0.8018018018018018
LogisticRegression() Avg cross val score is 0.8047829647829647
```

3. SVM (Support Vector Machine)

- SVM is a specific implementation of the Support Vector Machine algorithm that is designed specifically for classification tasks.



Code and Output

```
from sklearn import svm
model = svm.SVC()
model_val(model,X,y)
```

✓ 0.5s

```
SVC() accuracy is 0.7927927927927928
SVC() Avg cross val score is 0.7938902538902539
```

Decision Tree Classifier

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node.
- Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions.

Code and output

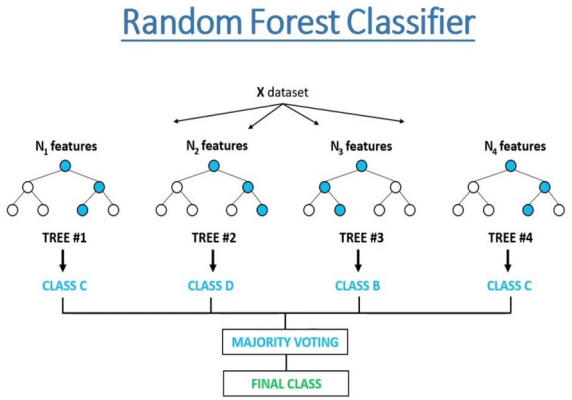
```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model_val(model,X,y)
```

[42] ✓ 0.1s

```
DecisionTreeClassifier() accuracy is 0.7297297297297297
DecisionTreeClassifier() Avg cross val score is 0.708894348894349
```

Random Forest Classifier

- Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique.
- It can be used for both Classification and Regression problems in ML.



Code and output

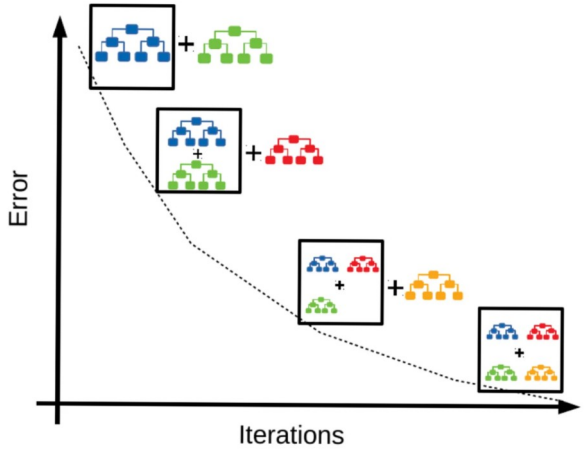
```
from sklearn.ensemble import RandomForestClassifier
model =RandomForestClassifier()
model_val(model,X,y)
```

[43] ✓ 4.8s

```
RandomForestClassifier() accuracy is 0.7567567567567568
RandomForestClassifier() Avg cross val score is 0.7830958230958232
```

Gradient Boosting Classifier

- Gradient Boosting classifier is one of the most popular forward learning ensemble methods in machine learning.
- It is a powerful technique for building predictive models for regression and classification tasks.



Code and output

```
[44]  from sklearn.ensemble import GradientBoostingClassifier
      model =GradientBoostingClassifier()
      model_val(model,X,y)
      ✓ 2.3s

...   GradientBoostingClassifier() accuracy is 0.7927927927927928
      GradientBoostingClassifier() Avg cross val score is 0.7722031122031121
```

Save the model

```
[63]
X = data.drop('Loan_Status',axis=1)
y = data['Loan_Status']
```
✓ 0.0s

```
[64]
rf = RandomForestClassifier(n_estimators=270,
                            min_samples_split=5,
                            min_samples_leaf=5,
                            max_features='sqrt',
                            max_depth=5)
```
✓ 0.0s

```
[65]
rf.fit(X,y)
```
✓ 1.5s

```
RandomForestClassifier(max_depth=5, min_samples_leaf=5, min_samples_split=5,
                       n_estimators=270)
```

Save the model

```
[66]  import joblib
      ✓ 0.0s

[67]  joblib.dump(rf,'loan_status_predict')
      ✓ 0.2s

...   ['loan_status_predict']

      + Code   + Markdown

[68]  model = joblib.load('loan_status_predict')
      ✓ 0.1s
```