Lab_12_EmsembleTechnique

KUMAR GAURAV 20122065

```
import pandas as pd
import numpy as np
```

```
train = pd.read_csv("train-1.csv")
```

```
train.head()
```

	id	url_legal	license	excerpt	target	standard_error
0	c12129c31	NaN	NaN	When the young people returned to the ballroom...	-0.340259	0.464009
1	85aa80a4c	NaN	NaN	All through dinner time, Mrs. Fayre was somewh...	-0.315372	0.480805
2	b69ac6792	NaN	NaN	As Roger had predicted, the snow departed as q...	-0.580118	0.476676
3	dd1000b26	NaN	NaN	And outside before the	-1.054013	0.450007

```
import seaborn as sns
```

```
train.shape
```

```
(2834, 6)
```

```
test = pd.read_csv('test.csv')
test.head()
```

	id	url_legal	license	excerpt
0	c0f722661	NaN	NaN	My hope lay in Jack's promise that he would ke...
1	f0953f0a5	NaN	NaN	Dotty continued to go to Mrs. Gray's every nig...
2	0df072751	NaN	NaN	It was a bright and cheerful scene that greete...
3	04caf4e0c	https://en.wikipedia.org/wiki/Cell_division	CC BY-SA	Cell division is the process

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7 entries, 0 to 6
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
```

```
---  ------     --------------  -----
 0   id         7 non-null      object
 1   url_legal  3 non-null      object
 2   license    3 non-null      object
 3   excerpt    7 non-null      object
dtypes: object(4)
memory usage: 352.0+ bytes
```

```
test.shape
```

```
(7, 4)
```

```
x = train.drop(['target'],axis=1)
```

```
x.head() # drop target variable
```

	id	url_legal	license	excerpt	standard_error
0	c12129c31	NaN	NaN	When the young people returned to the ballroom...	0.464009
1	85aa80a4c	NaN	NaN	All through dinner time, Mrs. Fayre was somewh...	0.480805
2	b69ac6792	NaN	NaN	As Roger had predicted, the snow departed as q...	0.476676
3	dd1000b26	NaN	NaN	And outside before the palace a great	0.450007

```
y = train['target']
```

```
y.head()
```

```
0   -0.340259
1   -0.315372
2   -0.580118
3   -1.054013
4    0.247197
Name: target, dtype: float64
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test = train_test_split(x,y ,test_size =0.2,random_state= 40)
```

```
from sklearn.tree import DecisionTreeClassifier
```

```
# Model Class to be used for different ML algorithms
class ClassifierModel(object):
    def __init__(self, clf, params=None):
        self.clf = clf(**params)
def train(self, x_train, y_train):
        self.clf.fit(x_train, y_train)
```

```
def fit(self,x,y):
    return self.clf.fit(x,y)

    def feature_importances(self,x,y):
        return self.clf.fit(x,y).feature_importances_

    def predict(self, x):
        return self.clf.predict(x)
def trainModel(model, x_train, y_train, x_test, n_folds, seed):
    cv = KFold(n_splits= n_folds, random_state=seed)
    scores = cross_val_score(model.clf, x_train, y_train, scoring='accuracy', cv=cv, n_job
    return scores


from sklearn.ensemble import RandomForestClassifier


# Random Forest parameters
rf_params = {
    'n_estimators': 400,
    'max_depth': 5,
    'min_samples_leaf': 3,
    'max_features' : 'sqrt',}
rfc_model = ClassifierModel(clf=RandomForestClassifier, params=rf_params)
print(rfc_model)
```

```
<__main__.ClassifierModel object at 0x7fa83fa75e50>
```

```
# AdaBoost parameters
ada_params = {
    'n_estimators': 400,
    'learning_rate' : 0.65
}
ada_model = ClassifierModel(clf=AdaBoostClassifier, params=ada_params)
ada_scores = trainModel(ada_model,x_train, y_train, x_test, 5, 0) # Random Forest
ada_scores
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-51-9bcc502e4e6c> in <module>()
      4     'learning_rate' : 0.65
      5 }
----> 6 ada_model = ClassifierModel(clf=AdaBoostClassifier, params=ada_params)
      7 ada_scores = trainModel(ada_model,x_train, y_train, x_test, 5, 0) #
Random Forest
      8 ada_scores

NameError: name 'AdaBoostClassifier' is not defined
```

SEARCH STACK OVERFLOW

```
# Gradient Boosting parameters
gb_params = {
    'n_estimators': 400,
    'max_depth': 6,
}
```

```
gbc_model = ClassifierModel(clf=GradientBoostingClassifier, params=gb_params)
gbc_scores = trainModel(gbc_model,x_train, y_train, x_test, 5, 0) # Random Forest
gbc_scores
```

```
def trainStackModel(x_train, y_train, x_test, n_folds, seed):
    cv = KFold(n_splits= n_folds, random_state=seed)
    gbm = xgb.XGBClassifier(
     n_estimators= 2000,
     max_depth= 4,
     min_child_weight= 2,
     gamma=0.9,
     subsample=0.8,
     colsample_bytree=0.8,
     objective= 'binary:logistic',
     scale_pos_weight=1).fit(x_train, y_train)

    scores = cross_val_score(gbm, x_train, y_train, scoring='accuracy', cv=cv)
    return score
```

0s completed at 7:59 PM