ex5

July 24, 2024

[ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import BaggingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
import numpy as np
from sklearn.tree import DecisionTreeClassifier
from sklearn.utils import resample
```

[ ]:
```
titanic_df = pd.read_csv('datasets/titanic dataset.csv')

titanic_df.head()
```

[ ]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th…	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

[ ]:
```
titanic_df.isnull().sum()
```

[ ]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

[ ]:
```
titanic_df['Age'].fillna(titanic_df['Age'].median(), inplace=True)
titanic_df['Embarked'].fillna(titanic_df['Embarked'].mode()[0], inplace=True)

titanic_df.drop(['Name', 'Ticket', 'Cabin'], axis=1, inplace=True)
```

[ ]:
```
label_encoders = {}
for column in ['Sex', 'Embarked']:
    le = LabelEncoder()
    titanic_df[column] = le.fit_transform(titanic_df[column])
    label_encoders[column] = le
```

[ ]:
```
X = titanic_df.drop('Survived', axis=1)
y = titanic_df['Survived']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[ ]:
```
scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[ ]:
```
base_estimators = {
    'KNN': KNeighborsClassifier(n_neighbors=5),
    'SVM': SVC(kernel='linear', probability=True)
}

results = {}
for name, base_estimator in base_estimators.items():
    bagging_clf = BaggingClassifier(estimator=base_estimator, n_estimators=60,␣
 ↪random_state=42)
    bagging_clf.fit(X_train, y_train)
    y_pred = bagging_clf.predict(X_test)
```

```
    accuracy = accuracy_score(y_test, y_pred)
    results[name] = accuracy

results
```

[ ]:
```
{'KNN': 0.8212290502793296, 'SVM': 0.7821229050279329}
```

[ ]:
```
X_scratch = X_train
y_scratch = y_train

n_estimators = 60

estimators = []

for _ in range(n_estimators):
    X_resampled, y_resampled = resample(X_scratch, y_scratch, random_state=42)
    estimator = DecisionTreeClassifier(max_depth=3)
    estimator.fit(X_resampled, y_resampled)
    estimators.append(estimator)

predictions = np.zeros((X_scratch.shape[0], n_estimators))
for i, estimator in enumerate(estimators):
    predictions[:, i] = estimator.predict(X_scratch)

final_predictions = (np.sum(predictions, axis=1) >= (n_estimators / 2)).
 ↪astype(int)


final_predictions[:10]
```

[ ]:
```
array([0, 0, 0, 0, 0, 1, 0, 0, 0, 0])
```