Model Development Phase Template

Date	10 July 2024
Team ID	865503
Project Title	Frappe Activity:Mobile Phone Activity Classification
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

This initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Model Training Code :

```
# Feature Selection
features = ['item','daytime','weekday','isweekend','cost','weather','name']
X = df[features]
y = df[["homework"]]
```

```
print(df['name'])
```

```
0         260
1        3217
2         698
3        1489
4        2523
        ...
96198     260
96199    1037
96200    1994
96201    1307
96202    2429
Name: name, Length: 96203, dtype: int32
```

```
from imblearn.over_sampling import SMOTE
```


```
# Feature Selection
features = ['item','daytime','weekday','isweekend','cost','weather','name']
X = df[features]
y = df[["homework"]]
```

```
print(df['name'])
```

```
0         260
1        3217
2         698
3        1489
4        2523
        ...
96198     260
96199    1037
96200    1994
96201    1307
96202    2429
Name: name, Length: 96203, dtype: int32
```

```
from imblearn.over_sampling import SMOTE
```


```
df['homework'].value_counts()
```

```
homework
1    75670
0    15771
2     4762
Name: count, dtype: int64
```

```
smote_sampler=SMOTE(random_state=42)
X_smote,y_smote = smote_sampler.fit_resample(X,y)
smote_data = pd.concat([X_smote,y_smote],axis=1)
smote_data.shape
```

```
(227010, 8)
```

```
smote_data['homework'].value_counts()
```

```
homework
1    75670
0    75670
2    75670
Name: count, dtype: int64
```

```
X_train, X_test, y_train, y_test = train_test_split(X_smote, y_smote, test_size=0.8, random_state=42)
```

```
# Scaling Data so that all the features has similar weight.
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import AdaBoostClassifier
```

```
models=[]
models = [('AdaBoost', AdaBoostClassifier(algorithm='SAMME'))]
models.append(('KNeighborsClassifier',KNeighborsClassifier()))
models.append(('DecisionTreeClassifier',DecisionTreeClassifier()))
models.append(('RandomForestClassifier',RandomForestClassifier()))
models.append(('XGBClassifier',XGBClassifier()))
models.append(('BaggingClassifier',BaggingClassifier()))
```

```
from sklearn.metrics import precision_score, recall_score, accuracy_score, f1_score

model_perform = {}

for name, model in models:
    # Fit the model
    model.fit(X_train, y_train.values.ravel())

    # Make predictions
    y_pred = model.predict(X_test)

    # Calculate performance metrics
    p = precision_score(y_test, y_pred, average='micro')
    r = recall_score(y_test, y_pred, average='micro')
    a = accuracy_score(y_test, y_pred)
    f = f1_score(y_test, y_pred, average='micro')

    # Store the performance metrics
    s = {'Precision': p, 'Recall': r, 'Accuracy': a, 'F1 Score': f}
    model_perform[name] = s
```

```
for model in model_perform:
    print(model)
    print("Precision: ",model_perform[model]['Precision'])
    print("Recall: ",model_perform[model]['Recall'])
    print("Accuracy: ",model_perform[model]['Accuracy'])
    print("F1 Score: ",model_perform[model]['F1 Score'])
    print()
```

Model Valuation And Evalution Report

Model	Classification Report	F1 Score
Bagging Classifier	BaggingClassifier Precision: 0.635522664199815 Recall: 0.635522664199815 Accuracy: 0.635522664199815 F1 Score: 0.635522664199815	63%
Random Forest	RandomForestClassifier Precision: 0.6382538214175587 Recall: 0.6382538214175587 Accuracy: 0.6382538214175587 F1 Score: 0.6382538214175587	63%
Decision Tree	DecisionTreeClassifier Precision: 0.6193945200651954 Recall: 0.6193945200651954 Accuracy: 0.6193945200651954 F1 Score: 0.6193945200651954	61%
KNeighbors Classifier	KNeighborsClassifier Precision: 0.5377571472622351 Recall: 0.5377571472622351 Accuracy: 0.5377571472622351 F1 Score: 0.5377571472622351	53%
XGB Classifier	XGBClassifier Precision: 0.64430531694639 Recall: 0.64430531694639 Accuracy: 0.64430531694639 F1 Score: 0.64430531694639	64%

AdaBoost	AdaBoost Precision: 0.4572430729923792 Recall: 0.4572430729923792 Accuracy: 0.4572430729923792 F1 Score: 0.4572430729923792	45%