ELG 5255 Applied Machine Learning Project

Submission

Please note that this project report should be in a presentation format (i.e., around 20 slides/PPT/PPTX), and for each step (question), you need to show the improvement (in terms of accuracy) by plotting a bar chart containing the baseline accuracy and all previous steps accuracy.

Slides and document Organization (10 Marks)

Presentations should include introduction, agenda, conclusion, etc.
A conceptual image should be at the beginning to explain problem.

Presentation (10 Marks)

Each group member should spend equal time and efforts.
You should prepare a recording of your presentation (maximum 10 minutes).

Please do not include codes in slides and your codes should be submitted as separate files.

For every figure, basic figure elements are expected such as title, x/y-axis, legend, labels (the numbers) for bars and points, and so on. For each step, detailed explanation should be discussed. Please write conclusion for each part and the whole project.

Dataset

Mobile CrowdSensing Dataset is used in this project. Dataset can be downloaded here.

> Dataset is generated by CrowdSenSim simulation tool. Dataset contains legitimate tasks and fake tasks. The task attributed are as follows: {'ID', 'latitude', 'longitude', 'day', 'hour', 'minute', 'duration', 'remaining time', 'battery requirement %', 'Coverage', 'legitimacy', 'GridNumber', 'OnpeakHour'}. Location of tasks are specified by 'latitude' and 'longitude' together. Furthermore, 'day', 'hour' and 'minute' describe the task publish time. 'Duration' denotes task active duration in terms of minutes. 'Remaining time' denotes the residual time of a sensing task till its completion. 'Battery requirement' is percentage of battery required to complete a task. 'Coverage' denotes task sensing distance. 'Legitimacy' describes whether a task is illegitimate one or legitimate one. This feature is used only in training of the machine learning models as the MCS platform is unaware of task legitimacy when a task is submitted. 'GridNumber' is obtained by splitting sensing city map to small grids with numbers beginning at 1. 'OnpeakHour' is a binary flag to indicate if task start time occurs during 7am to 11am. We define 7am to 11am as the peak hour and other hours are non-peak for the sake of simplicity in simulations. Based on the configuration of task generation in Table 1, the dataset is created including total 14,484 tasks, with 12,587 legitimate tasks and 1,897 fake tasks, respectively.

Features:

	Fake Tasks	Legitimate Tasks
Day	Uniformaly distributedly in [1, 6]	Uniformaly distributedly in [1, 6]
Hour	80%: 7am to 11am; 20%: 12pm to 5 pm	8%: 0am to 5am; 92%: 6pm to 23pm
Duration (min)	70% in {40, 50, 60}; 30% in {10, 20, 30}	Uniformly distributed over {10, 20, 30, 40, 50, 60}
Battery usage	80% in {7%-10%}; 20% in {1%-6%}	Uniformly distributed in {1%-10%}
Recruitment Radius	Uniformly distribute in 30m to 100m	Uniformly distribute in 30m to 100m
Movement Radius	[10m, 80m]	[10m, 80m]
Number of Tasks	1,897	12,578

The dataset is splitted according to date. Please use the following code to load the dataset

```
rs = 0
df = pd.read_csv(f'MCSDatasetNEXTCONLab.csv')
df['Ligitimacy'] = df['Ligitimacy'].replace(0,-1)

trDf = df[:10139][:]
teDf = df[10139:][:]

trDf = trDf.apply(lambda x: (x-45)*10000 if x.name == 'Latitude' else x)
trDf = trDf.apply(lambda x: (x+75)*10000 if x.name == 'Longitude' else x)
teDf = teDf.apply(lambda x: (x-45)*10000 if x.name == 'Latitude' else x)
teDf = teDf.apply(lambda x: (x+75)*10000 if x.name == 'Longitude' else x)

trY = trDf['Ligitimacy'].values
teY = teDf['Ligitimacy'].values
trX = trDf.drop(columns=['ID', 'Day', 'Ligitimacy']).values
teX = teDf.drop(columns=['ID', 'Day', 'Ligitimacy']).values
```

Part 1

This part is to develop machine learning pipelines.

NOTE: You can use your preferred library e.g., Scikit-learn, TensorFlow/Keras, PyTorch.

Q1 (10 Marks)

Apply only three of the following ML methods on the provided dataset to obtain baseline performance. Plot confusion matrix and calculate the accuracy for each methods, and plot them in a bar-chart as baseline.

- KNN
- LogisticRegression
- SVM
- DecisionTreeClassifier
- AdaBoostClassifier

Q2 (10 Marks)

To study the impact of over/under-sampling, try the following over/under-sampling methods for the 3 ML methods selected from Q1

- SMOTE
- ADASYN
- Random Undersampling

For each ML method, a `accuracy` vs `over/under-sampling methods` figure is expected.

In this step, the most suitable over/under-sampling method is selected for each ML method

Plot a bar chart to show the changes in performance after previous step and this step with baseline performances.

Q3 (10 Marks)

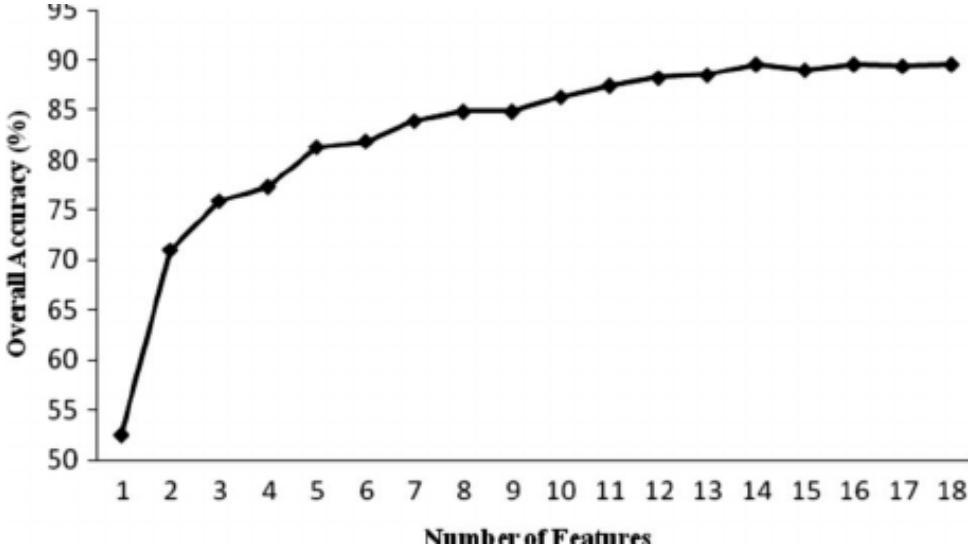
To improve model performances, apply one of the following feature selection and dimension reduction methods as a preprocessing step. By using the chosen method, obtain best 3~10 feature/dimension combinations.

Feature selection methods:

- Recursive Feature Elimination (RFE)
- Feature Importance from Random Forest
- PCA

1. Plot the accuracy score of each method vs number of features.

   E.g.,



2. Select the number of features with highest score, and plot bar-chart to show the changes of performance after previous step and this step with baseline performances.
3. Plot t-SNE 2D figures before and after applying the feature selection/dimension reduction that provide the best performance.

Q4 (10 Marks)

Apply stacking strategy to combine the fine-tuned models from previous step. Compare this results with baseline results and all results obtained from previous steps. Plot the confusion matrix and the bar-chart to show the improvement after each question in terms of accuracy.

Part 2

In this part you will practise the MLP classifier along with clustering approach.

Q5

Apply scaling to features (e.g. MinMaxScaler) before training your models.

Some of the hyperparameters are given below for MLPclassifier. You should find the suitable values for other important parameters (e.g. learning rate, momentum).

Each MLPclassifier should be run 10 times (trained and tested 10 times) and average accuracy should be provided to eliminate the effect of random initial conditions in MLP.

```
clf = MLPClassifier(activation="tanh", solver='sgd', hidden_layer_sizes=(15,15))
```

1. Apply MLP classifier to original data,provide the accuracy and plot the confusion matrix. This is your baseline performance for MLP classifier. (**10 Marks**)

2. Find the best feature combination for MLP model by using wrapper method from Q1 based on the testing accuracy of MLP. Train an MLP classifier on this dataset. Plot the number of features (X-axis) versus accuracy (Y-axis) graph with baseline performance. Provide the accuracy and plot the confusion matrix for the best results. This is your first step to improve baseline performance.(**10 Marks**)
3. In this step, you will utilize PKI approach to improve performance. Cluster dataset (both training and test set) by using SOM(m=k, n=k) where $2 \leq k \leq 8$. Add the obtained cluster labels as additional feature to selected features (best feature combination) in both training and test sets. Train your MLP classifier with the extra knowledge. Plot **k** (X-axis) versus **average accuracy** of 10 runs (Y-axis). Show your final accuracy from the previous question as the updated baseline.(**10 Marks**)

   Warning! (Avoid fitting tarining and testing sets separetly. Predict the cluster labels of test set with the SOM that fitted on the training set)
4. Tune the number of hidden layers(3-10) and number of neurons(10-30) using given values with the latest version of the processed dataset. Plot the **tuned parameters** versus **average testing accuracy**. You should make sure that the maximum accuracy is seen in the figure. Show your final accuracy from the previous question as the updated baseline.(**10 Marks**)