Problem:

Choose an appropriate dataset of your choice such that every record (example) has at least 5 features which are numeric in nature and there is at least one attribute (feature) which is binary in nature. You can use the binary attribute as the binary target label to be predicted.
Split your dataset into a training set and a test set. You can try different splits: 70:30 (70% training, 30% testing), 80:20 or 90:10 split.
On the training set, train the following classifiers:
1. Half Space
2. Logistic Regression (using inbuilt function)
3. SVM classifier (using a linear kernel)
4. SVM classifier (using a Polynomial kernel and a Gaussian kernel)
5. Logistic Regression using the SGD procedure.
If your data is not linearly separable, then you will be required to use the soft SVM formulation. You can use the inbuilt implementation of logistic regression and SVM in SciKit Learn.
Compare and analyze the results obtained by using the different classifiers. For the soft SVM formulation, you should compare the performance with the different values of the regularization parameter. Report the number of support vectors obtained for every dataset split. You should submit a report along with the code.

Solution:

For the given problem on linear classifiers, I have used the following data:
Occupancy Detection Data Set training data.
(source: https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+)

The relevant features are: temperature, humidity, light, CO2, humidity ratio and occupancy.

Libraries used for python coding:

Pandas, numpy, sklearn.linear_model, sklearn.model_selection, sklearn.svm, sklearn.metrics

Linear Classifiers

To prove a hypothesis, we train our data into a training model, which computes loss while defining a learning rule that acts as a feedback system for our model. Linear predictors are one such type of model. Our goal is to correctly classify all the training data points in our training set such that we can classify points other than the training set as well.
Linear predictors are efficient, intuitive, and fit the data reasonably well in many natural learning problems. They are used in several hypothesis classes like halfspaces, linear regression and logistic regression.

Splitting the dataset

Splitting the dataset into training and testing set can be achieved by train_test_split function of ScikitLearn library. Training set is used to train the algorithm or model, Test set is used to check the accuracy of the model. In this assignment, two splits have been used:
(1) training set:test set = 70:30,
(2) training set:test set = 80:20

1. Halfspace

This hypothesis class is designed for binary classification problems where, X = $\mathbf{R}^d$ (d-dimensional real vector) and y = {-1, +1} (classification labels).

The hypothesis class can be stated as:

$$H_{\text{halfspace}} = \{x \rightarrow sign(\langle w, x\rangle + b) : w \in \mathbf{R}^d\}$$

Thus the hypothesis forms a hyperplane, which is perpendicular to vector w, divides the training examples into two halves. The side that hyperplane shares an acute angle with w is labeled positively (+1), and the side where it makes obtuse angle is labeled negatively (-1).

For separable data, we can use Perceptron algorithm to find an ERM halfspace:

```
Input: training set: $(x_1, y_1), \ldots, (x_m, y_m)$
Initialize: $w(1) = (0,\ldots,0)$
  for $t = 1, 2, \ldots$
    if (for every i such that $y_i \langle w^{(t)}, x_i\rangle \leq 0$) then
        $w^{(t+1)} = w^{(t)} + y_i x_i$
    else
        output $w^{(t)}$
```

The perceptron finds an example i which is mislabeled i.e. $sign(\langle w^{(t),} x_i\rangle) \neq y_i$, then it updates $w^{(t)}$ by adding it to x_i scaled by label y_i. Hence, $w^{(t+1)} = w^{(t)} + x_i y_i$. Goal is to have $\forall$ i, $y_i \langle w^{(t)}, x_i\rangle > 0$.

For python code, perceptron function can be found in ScikitLearn library.

2. Logistic Regression

Logistic regression is a classification algorithm. It is used when the data is non-separable or non-realizable. For binary classification, y = {-1, +1} which is determined by feeding the input to the sigmoid function.

The hypothesis class can be written as:

$$H_{\text{sigmoid}} = \{x \rightarrow \varphi_{\text{sig}}(\langle w, x \rangle) : w \in \mathbf{R}^d \}$$

When $\langle w, x\rangle$ is very large, the sigmoid function $\varphi_{\text{sig}}(\langle w, x\rangle)$ is close to 1. While if $\langle w,x\rangle$ is very small $\varphi_{\text{sig}}(\langle w, x\rangle)$ is close to 0. But when $|\langle w, x\rangle|$ is near 0, the logistic hypothesis cannot predict. So, to find the accuracy of the model we calculate logistic loss function: $1 + exp(-y\langle w, x\rangle)$. Thus, ERM problem for logistics

Input: training set: $(x_1, y_1), \ldots,(x_m, y_m)$

Thus, ERM problem for logistics regression is:

$$\text{argmin}(w \in \mathbf{R}^d)\ (1/m) \sum \log(1 + exp(-y\langle w, x\rangle))$$

3. SVM Classifier

Support vector machine paradigm uses high dimensional feature space for learning linear predictors. There are two kinds of linear SVMs namely hard SVM and soft SVM. We use hard SVM for linearly separable data while soft SVM for non-separable data. However, mapping into high dimensional space increases the computational complexity of the model, which is why we use method of kernels. A kernel is a type of similarity function that takes two inputs and outputs their similarity.

Kernels can be linear or non-linear, for example: polynomial, Gaussian, etc. The feature space corresponding with a linear kernel is the original feature space. Linear kernel support vector machines have good performance only on very simple problems. The feature space corresponding with a polynomial kernel is the same as in polynomial regression. For Gaussian kernel, RBF (radial basis function) type kernel function is used.

Kernel function can be written as:

$$K(x,x') = \langle \varphi(x)\ \varphi(x') \rangle \qquad \text{where } x \text{ and } x' \text{ are in the input space X}$$

Comparison of the models:

For train:test = 70:30,

Halspace score = 0.980
Logistic regression score = 0.988
Linear kernel (SVM) score = 0.988
Polynomial kernel (SVM) score = 0.985
Gaussian kernel (SVM) score = 0.991
Logistic regression using SGD score = 0.989

For train:test = 80:20,

Halspace score = 0.970
Logistic regression score = 0.986
Linear kernel (SVM) score = 0.984
Polynomial kernel (SVM) score = 0.981
Gaussian kernel (SVM) score = 0.987
Logistic regression using SGD score = 0.982

Caution:

Regularisation and scaling are not used in this assignment problem because the accuracy of all the models is in the range of 0.9 to 1.0 which is considered good.

Conclusion:

The non-linear learning models are more better compared to linear models with respect to computational speed and accuracy both. Among all the 6 models Linear kernel (SVM) took the longest to compute.