**Project deliverables :**

1. Participate in Kaggle competition using the link: <https://www.kaggle.com/t/bd97585cca2a49f9b1701160047e989b>
2. Submit your jupyter notebooks and pdf files used to generate a solution for Kaggle Competition. Please submit files on eLearning.

Grading Criterion:

The project will be graded based on following three parameters:

1. **Completeness and thoroughness of your solutions. (Weight – 60%).**

This will be judged based on the submitted jupyter notebooks. **Please do not ignore warnings from your notebooks.** There are some warnings that cannot be avoided but some can be resolved by modifying code. If your notebook has warnings that can be resolved, you will lose points. For Example, in HW3 many students use model LogisticRegression(solver= lbfgs). This solver only supports l2 regularization. In the hyper parameters students then used both l1 and l2 penalty. The system gave you warning about this. In this case students should have changed their solver.

You will also be evaluated based on hyperparameter tuning. Please try to choose the best hyperparameters for your model (not just the default values taken from class templates).

Here is the breakdown for this component :

**Preprocessing Step (10%)** – Run the EDA (explanatory data analysis) e.g. examine distributions, missing values, correlations etc. Please add appropriate pre-processing steps based on your EDA. You need to provide some explanation of your pre-processing step.

**Models (50%)** - Here is the sample list of models you can try. If you think any model in the list below is not suitable for the data, please provide your justification of excluding the model:

1. **Basic Algorithms (5%)**
   1. Logistic Regression
   2. Decision Tree
   3. k-Nearest Neighbors
   4. Support Vector Machine (optional as it may take long time to run)
   5. Random Forest
   6. Extra Trees
   7. Gradient Boosting
   8. XgBoost
2. **Cost Sensitive Algorithms (15%)**
   1. Logistic Regression
   2. Decision Trees
   3. Support Vector Machines (optional as it may take long time to run)
   4. Random Forest
   5. XGBoost
   6. Extra Trees
   7. Bagging decision tree with under sampling
3. **Data Sampling Algorithms (15%)**
   1. Logistic Regression
   2. Decision Tree
   3. k-Nearest Neighbors
   4. Support Vector Machine (optional as it may take long time to run)
   5. Random Forest
   6. Easy Ensemble Classifier
   7. XgBoost
   8. Neural Network (scikit learn MLPClassifier)
4. **Stacking Classifiers (15%):** Try to combine various models to create stacking models. Generally diverse models gives you better results. As a rule of thumb, you need one final estimator for every 7 base estimators. If you have more than 7 base estimators in your stack, then the final estimator should itself be a stacking model with more than one estimator. Combining models trained on different data (one way to create different data set is using different pre-processing steps) can also sometimes gives better predictions. **As a minimum you need to try at least three different combinations for stacking.**
5. **Try at least three things which we have not covered in the class (20%)**

Students should be able to learn new things and apply them. Here is the sample of things we have not covered in class :

* + Hyper parameter tuning methods other than GridSeachCV e.g. RandomSerachCV.
  + Saving and Uploading models (pickle and/or Joblib)
  + Explore More Models/Libraries (some examples below)
    - Naïve Bayes
    - One Class Classification for imbalanced data (Isolation forest)
    - LDA
    - QDA
    - GaussianProcessClassifier
    - CatBoost
    - Light GBM

1. **Your rank in Kaggle Competition (20%)**

The 20% of assignment grade will be based on ranking in Kaggle Competition. This will be based on ranking on Private leaderboard and not Public leaderboard.