Jeevan Reddy Rachepalli Pavan Kumar Madineni Abhishek Rapelli

Statoil Iceberg challenge:

- First we have downloaded the .7Z files, extracted the json files and loaded them into pandas data frame and found that there are missing values in “inc_angle” column
- So, we replaced the missing(“na”) values with its mean.
- **First method:**
  Used Keras module for building Convolutional Neural Network model for this image classification problem.
  Transformed each array of image(5625 array values) into a feature attribute and added “band_2” column.
  The output variable will be "is_iceberg".
  Split the data in the ratio of 60(train) is to 40(test).
  Built a sequential model of neural net.
  Used different layers of neural nets like 3, 4 and 5 layers.
  Added different neuron counts in each layer with different activation functions.
  As it is a binary classifier we used “sigmoid” activation function in the last level.
  Compiled the model using “adam” optimizer.
  Evaluated the model on the train data. The accuracy level of prediction is estimated to be around 51%.

- **Second method:**
  Observed that the image of the “iceberg” or “ship” is captured only in the 30*30 [20:50,20:50] area of 75*75.
  Extracted only the confined area pixels and used both ”band_1” and ” band_2” images and “inc_angle”(1801 features).
  Followed the first method with different number of activation layers, different activation functions.
  The accuracy level improved to around 53%.

- **Third method:**
  Used the same features from the second method and built a KNeighborsClassifier (n=3) model.
  Split the data in the ratio of 90(train) is to 10(test).
  Fitted the model on training data and predicted the values of test data.
  Found out the confusion matrix.
  Got an accuracy level of 59%
  Used the 10-fold cross validation technique on the whole data and got an average accuracy of 79%.

Currently, we are using Trial and error method to find the optimal solution by varying the model and parameters.

Spooky Author Identification:

- Downloaded the training and test data from Kaggle and observed that the data does not have any missing values.
- Split the sentence of each author into words and stored in three different arrays (one for each author).
- Filtered out the special characters.

- Converted the entire text into lower case alphabets.
- Built a Pandas data frame with columns as words and rows as authors. [word counts are stored in here]

- **First Method:**
  Used Naïve-Bayes theorem to find the probability that a given line is written by a particular author.
  Stored the results in a csv file and uploaded it in Kaggle and found out that our position is in the 30^{th} percentile.

- **Second Method:**
  Deleted all the stopping words from the words data frame.
  Followed the same procedure as the First method and this time our position has improved to 37^{th} percentile.

- **Third Method:**
  This time we have split the data into test and train from the training data itself.
  Log loss error is estimated to be around 0.8.

  The major drawback of this method is that when many new words are encountered from any author, this method works poorly. We are currently trying a new way to tackle this problem.