**Designing a Supervised Learning Model**

**Problem Statement:** The sentiment of the news articles related to Pakistan is being determined as biased or unbiased. Given an input news article, we need to output a set of articles on the relevant topic but which express a view different than the viewpoint stated in the input article.

**Problem Analysis:** We need to think of the following questions while performing the exploratory data analysis:

* What will be the type of input and output to our program?
* Is it a problem of classification or regression?
* What are the features we should take into account?
* What are the limitations?

**Dataset**

We took the dataset from Pakistan today using web scraping libraries regarding the topic of elections 2018. This topic was chosen since this was one of the most controversial topics and the probability of biased news articles on this topic is extremely high. Due to time limitation only one topic was chosen and a small data was gathered which consisted of 85 articles in total since only these articles spanned the time period of elections 2018.

**Web scraping:** While performing web scraping, we used beautiful soup to return the html page from the specified link. The html page was parsed and the html tags were used to retrieve the links. The Article module was used to download the articles from the retrieved links.

**Labels:** The labels were generated manually by reading the articles and using the knowledge about various forms of bias present in media.

**0:** Unbiased

**1:** Biased

**Model Development**

**Data Cleaning**

* Converting to lowercase
* Remove non asci
* Remove punctuation
* Remove stop words
* Remove digits
* Remove commonly occurring words

**Data splitting:** Data was divided into training and testing since we want to predict the performance of our model on unseen data. The data was divided into 80:20 ratios.

**Feature Engineering**

**Bag of words**

We segment each article into words(tokens) and counted the count of each word and assign it an integer id. However, this leads to one issue: it will give more weightage to longer documents than shorter documents. Moreover, this method also gives equal weightage to all the words. Hence we also try some other feature vectors.

**TFIDF** (**Term Frequency times inverse document frequency)**

To deal with the issue of BOW model we use this feature vector. Through this we reduced the weightage of the more common words like (a, this, is) which occurs in the articles. This approach tells that how important a particular word is to a particular document with respect to the corpus.

**Bag of words with N gram model**

Adding features of higher n-grams can be helpful in identifying that a certain sequence of word occurs in the text. Hence we also used n gram model with bag of words. This approach did not work out for our data. We tried this with bigrams N= 2 and trigrams N =3 feature vectors.

**Model selection**

The following classification models were used with various features:

* **Naïve Bayes:** We used this model since this needs less training data.
* **SVM:** We used this because of faster prediction compared to Naïve Bayes algorithm.
* **Logistic regression:** We used this since it usually performs better and has a lower error rate than Naïve Bayes.

**Hyper parameter tuning**

We used Grid Search for hyper parameter tuning in all the models. The final model used was SVM. A linear kernel function was used. The value of gamma was adjusted since a lower value of gamma leads to bias and a high value leads to overfitting. We used regularization to prevent overfitting. Both Lasso and ridge regularization were tested on the dataset. The best model used Lasso regularization.

**Optimization Technique**

**MOTE (Synthetic Minority Over-Sampling Technique)**

Since the classes in our data were imbalanced we used an over-sampling approach in which the minority class is over-sampled by creating “synthetic” examples rather than by over-sampling with replacement. The minority class in our dataset was “Un-Biased” class. This improved the accuracy of our model by 7%.

**Results**

* Without oversampling and using TFIDF as feature vector we achieved best results with logistic regression and ridge regularization. The accuracy was 70%.
* With oversampling and using bag of words as feature vector we achieved best results with SVM and lasso regularization. The accuracy was 77%.

**Final Model (With Oversampling)**

**Feature vector =** Bag of words

**Classifier =** SVM

**Regularization =** Lasso

**SVM Accuracy:** 77%

**MSE:** 0.22

**Precision =** 0.78

**Recall =** 0.78

Recall expresses that the ability of the model to find all relevant instances (biased/unbiased) in

our dataset is 78% of the times while precision expresses the proportion of the data points our

model says was relevant actually were relevant and that is also 78% for both the classes.

**Limitation:** We assumed that the manual labelling is correct in tagging an article as biased or unbiased. It is possible that the tagging is not done correctly so human bias is present in our training data.

**Conclusion: Due to limited data we cannot generalize the results of our model. However, we were able to extract important features from the given data which are used to differentiate among the 2 classes present in dataset.**

**Tools Used**

* Language: Python
* Scraping library: BeautifulSoup and Newspaper
* Machine learning library: scikit-learn
* Data wrangling: Pandas and Numpy
* Plotting: matplotlib and Seaborn

All the visualizations and step by step detail is mentioned in the attached notebook.