Categorization of News Articles based on Industry

Objective: Classify the News articles based on Industry: Business, Sports, Tech, Entertainment, and Politics
Data Source: BBC
Tools used: Python
Solution:

1. Using the glob library of Python, I created a data set from the text documents.
   The dataset has 2225 rows and 2 columns: ["Doc Text", "Topic"]
2. **Text Preprocessing:**
   a. Removed all punctuations and new line characters: "\n\n" using regular expressions "[^\w\s]" and "\n\n" respectively.
   b. Removed single characters resulting from the removal of punctuations using the expression "\s+[a-zA-Z]\s+".
   c. Replaced single characters from the beginning of the document with a single space using "\^[a-zA-Z]\s+".
   d. Replaced one or more spaces with a single space using the regular expression "\s+".
   e. Converted all the words to lowercase and removed all the stop words using nltk library
   f. Stemmed all the words using PorterStemmer () of nltk library
3. **Tf-Idf Matrix:**
   Converted the dataset into a tf-idf matrix using TfidfVectorizer () 0f sklearn.feature_extraction.text
4. Split the dataset into training, validation, and test datasets.
5. **Data Modeling**: **(used 95% percent confidence interval (mean + 2*Std) on Cross-Validation scores to pick the best model)**
   I. Built general classifiers, such as SVM (used LinearSVC since it's a large dataset), Naïve Bayes and Decision tree, with all variables. But apart from SVM (LinearSVC) which got mean F1 test score of 0.98, other's performance is not that good with mean F1 test scores of 0.87 and 0.83 respectively
   II. Feature Engineering:
      a. Used Feature selection techniques, such as SelectKBest, RFECV, feature_importances_, etc. But they took a lot of time to run and classifiers built on the output of some techniques performed not that well.
   III. Feature Extraction:
      a. Performed Linear Discriminant Analysis (LDA) to find the best discriminants and used Random Forest Classifier on the variables of LDA and obtained a mean test F1 score of 0.46.
      b. Performed PCA (n_component =3) on the dataset and developed Naïve Bayes, Decision Tree, and SVM (LinearSVC) classifiers. The respective mean test F1 scores are 0.77, 0.87, and 0.75.
   IV. Feature Extraction with Bayesian Optimization:
      a. Used Bayesian Optimization on PCA and classifiers, such as KNN, Decision Tree, SVM (LinearSVC) classifiers. I observed the mean F1 test scores of 0.96, 0.93, and 0.95 respectively.
   V. Ensemble Methods with Hyperparameter Tuning and PCA:
      b. Built Random Forest classifier with all variables of the training data set and obtained an accuracy of 0.95 and a mean test F1 score of 0.95.
      c. Did Bayesian optimization to tune hyperparameters of both PCA and Random Forest and got an accuracy of 0.96 and mean F1 test score of 0.95.
      d. Performed PCA (n_component =3) on the dataset and used Random search to tune the hyperparameters of Light Gradient Boosting which resulted in the accuracy of 0.89 and mean test F1 score of 0.89
      e. Performed Bayesian optimization to tune hyperparameters of both PCA and Light Gradient Boosting and obtained an accuracy of 0.95 and mean test F1 score of 0.95.

Results:

Classifier	Mean test F1 Score	Accuracy
SVM (LinearSVC) with all variables	0.98	0.98
Naive Bayes with all variables	0.88	0.88
Decision Tree with all variables	0.83	0.84

SVM (LineatSVC) with SelectKBest (k=20)	0.66	0.69
Naive Bayes with SelectKBest (k=20)	0.46	0.42
Decision Tree with SelectKBest (k=20)	0.69	0.70
Random Forest with LDA and Bayesian Optimization	0.46	0.49
SVM (LinearSVC) with PCA (n_components = 3)	0.75	0.75
Naïve Bayes with PCA (n_components = 3)	0.77	0.77
Decision Tree with PCA (n_components = 3)	0.87	0.88
SVM (LinearSVC) with Bayesian Optimization (on PCA as well)	0.95	0.96
KNN with Bayesian Optimization (on PCA as well)	0.96	0.96
Decision Tree with Bayesian Optimization (on PCA as well)	0.93	0.93
Random Forest with all variables	0.95	0.95
Random Forest with Bayesian Optimization (on PCA as well)	0.95	0.96
Light Gradient Boosting with PCA and Random Search	0.89	0.89
Light Gradient Boosting with Bayesian Optimization (on PCA as well)	0.95	0.95

Conclusion:

a. I obtained the best accuracy of 0.98 and an F1 score of 0.98 using SVM (LinearSVC) on all variables but the model has high complexity.
b. Bayesian optimization on PCA and Random Forest gave an accuracy of 0.95 and an F1 score of 0.96, and this model is not complex like the best model above but took more time.
c. Bayesian optimization on PCA and Light Gradient Boosting gave an accuracy of 0.95 and an F1 score of 0.95, and this model has low complexity like the above model and took lesser time compared to the above model.

Observations:

a. The given dataset is linearly separable as LinearSVC gave the best performance in terms of F1 score and accuracy.
b. Obtained better models when I used 95% percent confidence interval (mean + 2*Std) on Cross-Validation scores to pick the best model.
c. Performing hyperparameter tuning on both PCA and Ensemble methods/Standard classifiers gave me higher accuracy and F1 score, but took more time, than just using hyperparameter optimization on Ensemble methods/standard Classifiers.
d. When the variables are collinear, it's better to use PCA rather than LDA because with LDA gives only a maximum of min (n_classes, n_features-1) variables.
e. Classifiers, when tuned with Bayesian Optimization, don't give the same results on every instance
f. Sometimes, the standard classifiers, such as Decision Tree, Naïve Bayes, KNN, and SVM, would give the best results when tuned using Bayesian Optimization
g. When the dataset is large, it's better to use Light GB if time is the constraint or else you can use Random Forest or XGBoost which give better results
h. In the case of high dimensional data, it's better to use feature extraction techniques rather than feature selection techniques. You can use feature selection techniques like wrapper methods (RFECV), but RFECV takes a lot of time.