Homework 3

Answer 2

a) Steps: -

- **Downloaded the data**
- **Wrote a code in Spyder for the initial analysis:**
  - Imported the numpy and pandas libraries in python.
  - Uploaded the training dataset.
  - Made a subset of data taking relevant columns.
  - Set x_train and y from the dataset.
  - Imported LabelEncoder and OneHotencoder from sklearn.
  - Applied LabelEncoder on the categorical string columns for data preprocessing.
  - Imported re, nltk to create corpus of the text containing columns.
  - Applied OneHotEncoder on all the columns for x_train and y.
  - Imported train_test_split from sklearn to apply cross validation on data by splitting the data into training and testing set.
  - Imported StandardScaler from sklearn for feature scaling.
  - Imported LogisticRegression from sklearn to build a model.
  - Imported confusion_matrix from sklearn to build an error metrics.
  - Imported accuracy_score from sklearn to find the accuracy of the hence built model.
  - Imported DecisionTreeClassifier from sklearn to build a decision tree model.
  - Found the accuracy again for this model.
  - Imported SVC from sklearn to build an SVM model using 'rbf' kernel.
  - Found the accuracy of the model.
  - Repeated all the steps till best features are identified to build the model and by the best model means the best accuracy.

Features	**Logistic Regression**	**SVM**	**Decision Tree**
All the columns	54.26%	46.53%	57.54%
Leaving columns page, answer and text	66.70%	66.77%	72.77%

After a round of hit and trial I found that the accuracy is nearly equal while including page, answer or text column either one of them or all of them. This means that these columns are not making a good model. However, excluding these columns gave the accuracy of decision tree equivalent to 72.77% which was the best among the rest.

b) The patterns that I came across here was the columns "page", "answer" and "text" are not making any sense as they are decreasing the accuracy. I tried using each one of these columns separately but came to a conclusion that the accuracy is almost same in all the cases. This was a pattern for all the models. None of the models gave a good accuracy model if these features were included.

Answer 3

For this part, I tried using different approaches one by one on the columns particularly 'page' and 'text' so that the accuracy of my model could be increased. I created some plots to find the useful parameters. One approach that I thought of was to take the length of the text and create a new column for that. The thought process behind this approach was that the more the length of the text the more are the chances of the answer to be wrong. Also, I tried to separate the year of the tournament from the column 'tournament' and noticed that in years 2007, 2008 and 2009, there are more answers that are false. I also noticed that if the answer type is Work, then there are more chances for the answer to be correct. I have attached all the plots in a separate document. Thus, I have used all these parameters to create a new model so that the accuracy of the model should increase.

Steps: -

- Imported the numpy and pandas libraries in python.
- Uploaded the training dataset.
- Import re and create a new column year by separating the yeqar from the existing 'tournament' column.
- Create a new column to calculate the length of the text in the column 'Text'.
- Made a subset of data taking relevant columns.
- Set x_train and y from the dataset.
- Imported LabelEncoder and OneHotencoder from sklearn.
- Applied LabelEncoder on the categorical string columns for data preprocessing.
- Imported re, nltk to create corpus of the text containing columns.
- Applied OneHotEncoder on all the columns for x_train and y.
- Imported train_test_split from sklearn to apply cross validation on data by splitting the data into training and testing set with 20% of the data as test data.
- Imported StandardScaler from sklearn for feature scaling.
- Imported LogisticRegression from sklearn to build a model.
- Imported confusion_matrix from sklearn to build an error metrics.
- Imported accuracy_score from sklearn to find the accuracy of the hence built model.
- Imported DecisionTreeClassifier from sklearn to build a decision tree model.
- Found the accuracy again for this model.
- Imported SVC from sklearn to build an SVM model using 'rbf' kernel.
- Found the accuracy of the model.
- Printed the confusion matrix for the model with best accuracy.

a) Attached plot.pdf
b) The table below shows the improved efficiencies from each model.

Features	Logistic Regression	SVM	Decision Tree
Leaving columns row, page, answer and text and taking length and year instead.	68.5%	68.87%	75.93%