Answer the following questions:

- Which techniques you have used while cleaning the data if you have cleaned it?

First detect the number of duplicates rows in dataset using **pandas.DataFrame.duplicated**

```
#Detect duplicate in data 4618 row duplicates in job title per industry
dataset[dataset.duplicated()].head()
```

	job title	industry
7	devops engineers x 3 - global brand	IT
10	devops engineers x 3 - global brand	IT
11	devops engineers x 3 - global brand	IT
26	business analyst	IT
36	.net developer	IT

Second drop duplicates in job title column using **pandas.DataFrame.drop_duplicates**

```
[384] #Drop Duplicated rows
      dataset_cleaned = dataset.drop_duplicates(subset="job title")
      # Show Industry Frequency count after remove duplicates
      plt.figure(figsize=(8,4))
      plt.title("Industry Frequency count")
      sns.countplot(x = 'industry', data = dataset_cleaned)
```

Last preprocessing text like removing stop words (thus,and,a,...), removing words less than 3 letters, punctations and using lower case for all the text like in **clean_text(text)** that I used for do that and I used map to map method for all rows in dataset.

```
[385] # take text and preprocess 'remove stopwords [a, the, and, thus, ... etc] and punctations[,%$ ..etc] and len of text less than 3'
      def clean_text(text):
          """
              text: a string
              return: cleaned string
          """
          result = []
          for token in simple_preprocess(text):
              if token not in STOPWORDS and token not in punctation and  len(token) >= 3 :
                  token = token.lower()
                  result.append(token)
          return " ".join(result)
```

```
[386] dataset_cleaned['job title'] = dataset_cleaned['job title'].map(clean_text)
      dataset_cleaned.head()
```

- Why have you chosen this classifier?

I used both Multinomial Naive Bayes and SGDClassifier both get good predictions and accuracy and good fit for text classification.

- How do you deal with (Imbalance learning)?

There are several ways to deal with Imbalance learning such as **upsampling, downsampling and adding class weights into consideration when training.**

when I drop the duplicates rows it turned out to be most of them were belongs to class ("IT"). This semi-solved the imbalance problem at least between three classes, and one class was yet too small relative to them. So, I also decided to add sample weights to the classifier when training to further solve this issue of imbalance.

- How can you extend the model to have better performance?

i tried 2 types of models: SGDClassifier and Multi Nominal NB

- I also tried to use the following for **text vectorization:**
- Simple features out of text (CountVectorizier and TF-IDF), and
- complex features (word2vec)

- How can you How do you evaluate your model?

I chose my final model to be the highest in measure (accuracy) of those three classifiers which was SGDClassifier which I choosed with accuracy on test set `0.89545844044`

- What are the limitations of your methodology or Where does your approach fail?

My limitations are in the amount data, although I tried to handle class imbalance with sample weights and removing those duplicates, misclassifying for classes with lower relative samples still occurs (My model would fail for this case).

Normally The text classification problem is a problem that requires a lot of data.

Those pretrained models achieving state of the art performance have millions and billions of words to play with.