Task: Given Information about title and description of a service or product we need to classify them. So the problem is Binary classification.

The Training set contains 12,00,000 Examples.
The Test set contains 92 Examples.

Also given that sometimes only one column could be present.

We need to take classification decision based on both title and description.

On seeing the titles and descriptions the following things are **my observations.**

1. Titles are mostly short and descriptions are longer in word length.
2. Titles contain mostly unique strings with more product or service information and descriptions contains mostly naturally used language describing the product or service.
3. Titles contain the product/service identification. For this features like TF-IDF would be suitable as they can pick the distinguishing patterns clearly. Whereas descriptions are mostly semantical and I think some word-embeddings models might better understand the meaning behind them.
4. Also, overall in descriptions we can more natural language sentences containing more lengthy sequences. These kind of things can be modelled with LSTM and BERT. Also because of the natural words and sentences in descriptions pretrained model of BERT would give better value.

Importance of Preprocessing in textual data:

Before converting text into features(vectorized format) one has to clean the data by using preprocessing steps like n-grams, removing stop words, removing punctuations, and other techniques like stemming and lemmatization. Stemming and lemmatization prune the word to its root. Plural becomes singular, different tense variants reduce to their simplest present form.

Feature Extraction:

In Text classification, Feature extraction is key in getting better score in classification.
After checking at the data the following feature extraction techniques were used.

Word2vec: They are the vector representations of each word. They are capable of capturing context of a word in a document, semantic and syntactic similarity, relation with other words. Semantic similar words are mapped to proximate points in geometric space.

TF-IDF: Term Frequency(TF) - Inverse Document Frequency(IDF) used to classify the text without understanding the meaning of sentence.

Training and Testing models

I have experimented with different feature extraction techniques like word2vec and TF-IDF and used different classifiers like SVM-linear, Logistic Regression, Random Forest for training the models.

First I have trained different models for title and descriptions and finally combining their predictions using ensemble approach by combing the best title model and the best description model.

Feature concatenation approaches are not suitable in the given problem because it was mentioned that some times either title or description would be absent. In such case we need to see what we can predict individually and combine them effectively if both exists.

I have used average f1 score for both classes to report performance of model and for comparison and results were predicted on test1.csv provided.

Word2vec + Linear svm : Title: f1 score: 0.78
Descrp: f1 score: 0.53

Word2vec + Logistic: Title : 0.77
Descrp 0.57

TF-IDF + Linear svm: Title f1 score: 0.77
Descrp f1 score: 0.53

TF-IDF + Logistic : Title: f1 score: 0.72
Descrp f1 score: 0.59

From TF-IDF results we can observe that for both Title and description classification, the results were slightly inferior to word2vec. My assumption that TF-IDF would work at least for title is proven wrong. Semantic embeddings are still valid and are more valuable in the product/service titles which had more unique strings and patterns.

Word2vec + RandomForest: Title : 0.77
Description: 0.58

LSTM: Given problem has sequences of inputs and for sequence classification we can use RNNs particularly LSTM which are good at modeling and understanding the patterns in sequences. LSTMs are better than RNNs that they solve the problem of vanishing gradients.

In this task we used LSTMs to model the classification as we observe pattern in both title and description.

LSTM on Title : F1 score: 0.717

LSTM on description: F1 score: 0.609

For description results the LSTM worked better than other feature extraction and classifier methods.
Finally I tried with ensemble models and took the ensemble of word2vec-title-logistic model and lstm-description model with a combination of probabilistic scores to make the model better. This ensemble model hasn't improved the accuracy and when we use both description model and title model to get the predictions.
Particularly if the descriptions are unavailable as stated in the problem for some of the inputs then we can only use title model to make predictions.

Ensemble of (word2vec+title+ logistic) +(LSTM+description) = 0.77

Confusion matrix on testset by the above model

	Product	**Service**
Product	**21**	**14**
Service	**7**	**50**

The ensemble model did not improve accuracy because the description model was weaker in predicting the category of descriptions. If the description model is improved then some combination of title model and description model would give better results than the single model alone.

Other Methods and Models to try:

Description classification performance can be improved by better models which are trained on larger corpus or natural sentences which also models context. Some of them would be BERT, ELMO.

More preprocessing of the title information can help improve performance. Many examples contain date and other product id information which if can be processed and identified then it would be easier for model to understand. More study on patterns in title information should be needed to preprocess them.

Non linear classifiers or parameter tuning in classifiers will improve performance. Because of time constraints modeling of parameters couldn't be done. Neural networks with dense layers can also be used as classifiers. That can also be experimented.

Word2vec Embeddings with LSTM: In the above LSTM experiments, we have learnt LSTM using embeddings learned with words, but we can also use word2vec embeddings and use the embedding matrix as pretrained weights for the LSTM. That can improve the results. Loading weights is explained here. Also, to improve the classification of description sentences we need more pretrained models on larger corpus or finetuned on the given dataset.

CNNs for text classification: Convolutional neural networks can also be used for text classification. After obtaining embeddings like word2vec sentences are padded to sequences and passed through convolution layers, Global max pool, dense and dropout layers to predict the labels. This article explains clearly about the use of cnns for text classification.

FastText for text classification: FastText uses a simple and efficient baseline for sentence classification(represent sentences as bag of words (BoW) and train a linear classifier). It uses negative sampling, hierarchical softmax and N-gram features to reduce computational cost and improve efficiency. This was introduced by facebook. Many people have reported better accuracies by using FastText embeddings. That can also use character level information to improve performance.

References:

http://jalammar.github.io/illustrated-word2vec/

https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa
https://towardsdatascience.com/how-i-improved-my-text-classification-model-with-feature-engineering-98fbe6c13ef3

https://towardsdatascience.com/cnn-sentiment-analysis-1d16b7c5a0e7

https://towardsdatascience.com/super-easy-way-to-get-sentence-embedding-using-fasttext-in-python-a70f34ac5b7c
https://towardsdatascience.com/fasttext-under-the-hood-11efc57b2b3