Question 3

COVID - Drug Discovery for COVID19

Given the dataset containing drug molecules (encoded as SMILES) and their binding affinities. The task is to use this dataset to make a regression model for binding affinity prediction.

SMILES Representation of Molecules -

SMILES are character strings to represent drug molecules. For example, a carbon atom can be represented as “C”, an oxygen atom can be represented as “O”, double bond by “=”. The molecule Carbon dioxide is represented as “C(=O)=O”.

For this question, I have used the **rdkit module.**

Installation of rdkit:

```
#install rdkit

!wget
"https://gist.githubusercontent.com/philopon/a75a33919d9ae41dbed5bc6a39f5ede2/raw/5bb62e381123558f2cc3149f9a7baeb84c90ba03/rdkit_installer.py"

!python3 rdkit_installer.py
```

Preprocessing

```
#We're going to settle the function that searches patterns and use it for a list of most common atoms only
def number_of_atoms(atom_list, df):
    for i in atom_list:
        df['num_of_{}_atoms'.format(i)] = df['mol'].apply(lambda x: len(x.GetSubstructMatches(Chem.MolFromSmiles(i))))

number_of_atoms(['C','O', 'N', 'Cl'], df)
```

rdkit.Chem.Descriptors provides a number of general molecular descriptors that can also be used to featurize a molecule. Most of the descriptors are straightforward to use from Python.

Using this package we can add some useful features to our model:

- rdkit.Chem.Descriptors.TPSA() - the surface sum over all polar atoms or molecules also including their attached hydrogen atoms;
- rdkit.Chem.Descriptors.ExactMolWt() - exact molecural weight;

I have used mol to vector model and pretrained data.

#commands to install pretrained model

!pip install git+https://github.com/samoturk/mol2vec;

!ls

!wget https://github.com/samoturk/mol2vec/blob/master/examples/models/model_300dim.pkl?raw=true

Converting mol to vector:

```
#Pre training

from gensim.models import word2vec

model = word2vec.Word2Vec.load('model_300dim.pkl?raw=true')

from mol2vec.features import mol2alt_sentence, mol2sentence, MolSentence,
DfVec, sentences2vec
```

```
#Constructing sentences

mdf['sentence'] = mdf.apply(lambda x: MolSentence(mol2alt_sentence(x['mol'],
1)), axis=1)

mdf['mol2vec'] = [DfVec(x) for x in sentences2vec(mdf['sentence'], model,
unseen='UNK')]

X = np.array([x.vec for x in mdf['mol2vec']])

y = target.values



test['sentence'] = test.apply(lambda x: MolSentence(mol2alt_sentence(x['mol'],
1)), axis=1)

test['mol2vec'] = [DfVec(x) for x in sentences2vec(test['sentence'], model,
unseen='UNK')]

X_test = np.array([x.vec for x in test['mol2vec']])
```

Models used :

1. Ridge model.

```
X_train, X_t, y_train, y_test = train_test_split(X, y, test_size=.1,
random_state=1)

ridge = RidgeCV(cv=5)

ridge.fit(X, y)

evaluation(ridge, X_t, y_test)
```

2. SVR regression model.

```
from sklearn.svm import SVR

clf = SVR(C=200,epsilon=0.5)

clf.fit(new_df, y)

evaluation(clf, X_t, y_test)
```

Metric Used : MSE (Mean Squared Error)

Observations:

1. *From ridge model, I got mse = 5.645*
2. *From SVR model,I got mse =5.234*
3. *Used pre trained model.*
4. *Checked for different values of the C for SVR.*
5. *Increasing C, Decreases the mse.*

Conclusion:

Hence, For test data, I have used SVR to train the model and I got 2.27 rmse.