Wine Quality Predict Analysis

Vivek Mondal

Data Used: **https://drive.google.com/file/d/1tTKVw6qFzXDTtuDGvuHMe-QkmA-WZhp_/view?usp=drive_link**

Tool Used: Anaconda Jupyter Notebook.
Language used: Python.

Library used: Pandas, Numpy, Matplotlib, Seaborn, Tensorflow,Keras, Sckt-learn,Statmodels.

In Wine_Quality_Data_Visualization_by_Vivek_Mondal.ipynb file.
The dataset has been imported using Pandas Dataset, the dataset has a total of 1599 rows and 12 columns and it contains 'fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar','chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density','pH', 'sulphates', 'alcohol', ‘quality’. The target column of this dataset is ‘quality’.
From the dataset we can cruelly see that the data is not balanced, number of average quality is higher than the poor and high quality.
I have noticed that the dataset has a very less number of null value, so I have replaced that null value with the average of that column.
I have plotted some bar plots and for observing the relation of each columns with the quality, and also plot a box-plot using seabird library , to watch the distribution of the data of each column. I have also Plotted an Heatmap using the seaboarn library to see the co-relation between the datset.

In Prediction_Of_Wine_Quality_by_Vivek_Mondal.ipynb file,

I have created a class named ‘Wine_Quality_Prediction’ to predict the quality of the wine using Machine Learning Model.

Functions :

- **read_data()** ->Read the dataset.
- **dataset_shape()** ->Returns the shape of the dataset.
- **dataset_info()** -> Return the information about the dataset.
- **dataset_round()** -> Round all the values to 3 decimals places and return the dataset.
- **dataset_null_check()** -> It check if there is any ‘Null’ value in the dataset.
- **dataset_null_value_set()** -> It fill the Null value.
- **dataset_corelation_heatmap()** -> It show the HeatMap of the dataset to the users.
- **dataset_pair_plot()** -> It show the PairPlot of the dataset to the users.
- **outlier()** -> It removes the outliers from the dataset.
- **dataset_split_Xy()** -> It has splitted the dataset into Target variable and Dependendent variables. And return Dependent variables as X and targeted variable as y.
- **oversampler_dataset()** -> Since the dataset is very small so I have used RandomOverSampler library which is coming from imblearn.over_sampling, to increase the dataset.
- **scaled_data()** -> With this function I have scaled the dataset. Using StandardScaler from sklearn.preprocessing library it returns the scaled dataset.
- **dataset_train_validation_split()** -> This function break the dataset into Train dataset and validation dataset . It takes 80% of the dataset as train dataset , and 20 % of the

dataset as validation dataset. It returns train dataset as X_train and y_train , and the validation dataset as X_test, y_test.

- **rfe_cols_name()** -> using the RFE from sklearn.feature_selection library it removes the weak features from the dataset and returns the X_train with strong features.
- **model_vif()** -> It returns a data frame which shows the Variance influence factors of the X_train dataset.

Since from the dataset it was not easy to say that it can be a classification model or a regression model , so I have done bot Classification Model and Regression model.

- **quality_predict_regression()** -> After using scaled_data() and **dataset_train_validation_split()** , **rfe_cols_name(),** I have got a scaled X_train data , and then I have build an ANN model using Sequential() coming from 'tensor flow.keras.models' . In the model first I have added a Dense layer with value 128 , and activation is "relu".
  Then add another Dense layers value 256 with same activation._Then add another Dense layers value 256 with same activation.
  Then add another 2 Dense layers value 512 with same activation.
  Then add another 3 Dense layers value 1024 with same activation.
  At last I have added Dense layers with value 1 and activation 'linear'.
  I have compile the model using 'Adam' optimizer , with learning_rate=0.0004 , and use 'mean_squared_error' as loss. Then fitted the model using batch_size=42, epochs=200 and validation_split=0.1.

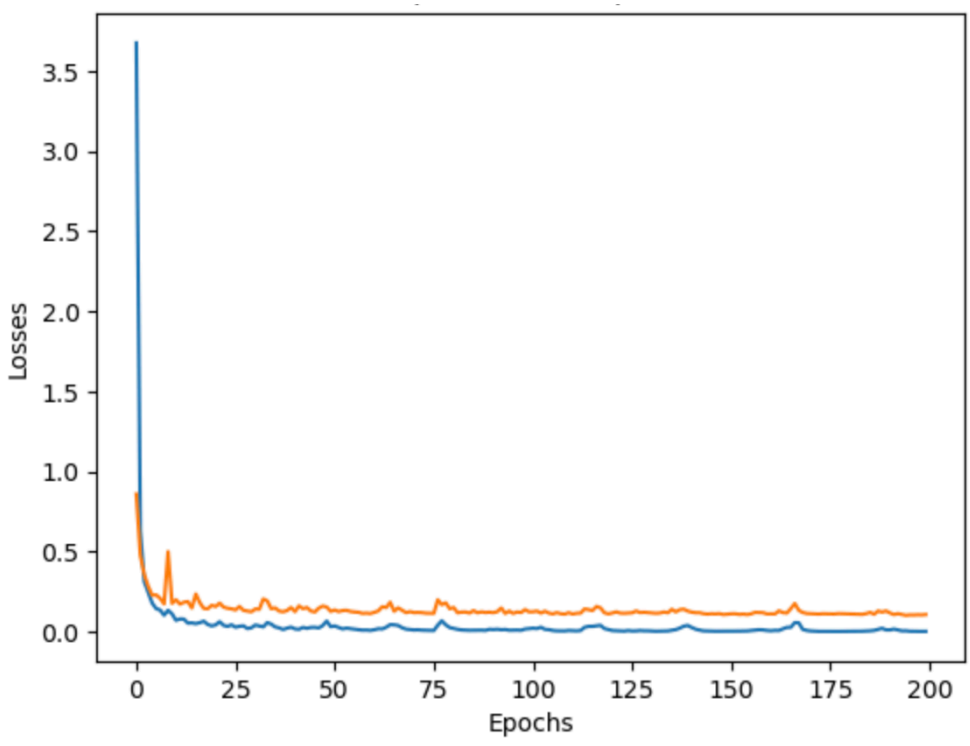
After completion I have predicted the value using model.predict and it return me the predicted value using the model .

Then I have calculated the r2_score for the target value and the predicted target value and got 0.99 for train dataset and 0.96 of validation dataset.

```
w.quality_predict()
92/92 [==============================] - 1s 9ms/step - loss: 0.0096 - val_loss: 0.1113
Epoch 194/200
92/92 [==============================] - 1s 9ms/step - loss: 0.0038 - val_loss: 0.1063
Epoch 195/200
92/92 [==============================] - 1s 9ms/step - loss: 0.0043 - val_loss: 0.1008
Epoch 196/200
92/92 [==============================] - 1s 9ms/step - loss: 0.0015 - val_loss: 0.1034
Epoch 197/200
92/92 [==============================] - 1s 9ms/step - loss: 8.8490e-04 - val_loss: 0.1029
Epoch 198/200
92/92 [==============================] - 1s 9ms/step - loss: 8.5888e-04 - val_loss: 0.1044
Epoch 199/200
92/92 [==============================] - 1s 9ms/step - loss: 5.5918e-04 - val_loss: 0.1036
Epoch 200/200
92/92 [==============================] - 1s 9ms/step - loss: 7.2277e-04 - val_loss: 0.1057
102/102 [==============================] - 0s 2ms/step
26/26 [==============================] - 0s 2ms/step
r2_score for train data: 0.9962151395229902
r2_score for validation data 0.9618155021823702
```

Using quality_predict_regression()

And using **Regression_training_graph_show()** I have got the graph of loss vs epochs of train and validation data.



Using quality_predict_regression()

- **quality_predict_classification()** ->_After using scaled_data() and **dataset_train_validation_split()** , **rfe_cols_name(), set_data_for_classification() ->** used to transform the quality data to [0,1,2] . I have got a scaled X_train data , and then I have build an ANN model using Sequential() coming from 'tensor flow.keras.models' . In the model first I have added a 1 Dense layer with value 1024 , and activation is "relu".
  Then add 1 Dense layers value 2 with sigmoid activation.
  I have compile the model using 'Adam' optimizer , with learning_rate=0.0004 , and use 'keras.losses.categorical_crossentropy' as loss.
Then fitted the model using batch_size=42, epochs=200 and validation_split=0.1. After completion I have predicted the value using model.predict and it return me the predicted value using the model .Then I have calculated the r2_score for the target value and the predicted target value and got 0.93 for train dataset and 0.87 of validation dataset.

```
In [48]: w.quality_predict_classification()
82/82 [==============================] - 0s 4ms/step - loss: 0.0438 - val_loss: 0.2216
Epoch 194/200
82/82 [==============================] - 0s 4ms/step - loss: 0.0419 - val_loss: 0.2199
Epoch 195/200
82/82 [==============================] - 0s 4ms/step - loss: 0.0515 - val_loss: 0.2060
Epoch 196/200
82/82 [==============================] - 0s 4ms/step - loss: 0.0313 - val_loss: 0.2126
Epoch 197/200
82/82 [==============================] - 0s 4ms/step - loss: 0.0153 - val_loss: 0.1899
Epoch 198/200
82/82 [==============================] - 0s 4ms/step - loss: 0.0149 - val_loss: 0.2302
Epoch 199/200
82/82 [==============================] - 0s 4ms/step - loss: 0.0190 - val_loss: 0.1883
Epoch 200/200
82/82 [==============================] - 0s 4ms/step - loss: 0.0229 - val_loss: 0.2503
103/103 [==============================] - 0s 1ms/step
26/26 [==============================] - 0s 1ms/step
r2_score for train data: 0.9384372996241357
r2_score for validation data 0.8740744554906491
```

Using quality_predict_classification()

- I have used another model using XGBoost using the Xgboost classifier I have got 90.3 % accuracy on validation . That means this model can predict the quality of the Wine 90.3 %

```
In [115]:
xgclf = xgb.XGBClassifier()

xgclf.fit(X_train, y_train)
```

```
Out[115]:
XGBClassifier
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              n_estimators=100, n_jobs=None, num_parallel_tree=None,
              objective='multi:softprob', predictor=None, ...)
```

```
In [116]:
metrics.accuracy_score(y_train,xgclf.predict(X_train))
```

```
Out[116]:
1.0
```

```
In [117]:
metrics.accuracy_score(y_test,xgclf.predict(X_test))
```

```
Out[117]:
0.903125
```

For XGBClassifier I have got accuracy of 90.3% for validation data .thats mean this model can detect the quality of the wine 90.3 % correctly.

Using XGBoost

correct.

- All the Models made by me can Predict the quality of the wine with a very good accuracy. As per user choice any model can be used to predict the quality of the Wine .

* Tested the XGBoost model on random data.

```
testing_data={"fixed acidity":10.2,"volatile acidity":0.5,"citric acid":0,"residual sugar":2.5,"chlorides":0.045,"free
test_data=pd.DataFrame(testing_data,index=[0])
test_data
predicted=xgclf.predict(test_data)
if predicted==1:
    print("Average quality")
elif predicted==0:
    print("Poor quality")
else:
    print("Excelent Quality")
```

```
Average quality
```

```
testing_data={"fixed acidity":9.1,"volatile acidity":0.4,"citric acid":0.5,"residual sugar":1.8,"chlorides":0.071,"fre
test_data=pd.DataFrame(testing_data,index=[0])
test_data
predicted=xgclf.predict(test_data)
if predicted==1:
    print("Average quality")
elif predicted==0:
    print("Poor quality")
else:
    print("Excelent Quality")
```

```
Excelent Quality
```

Random data test

End of The Report.