Project Name : Concrete Compressive Strength Prediction using DL Auto Keras(Auto ML)

- To predict and analysis concrete compressive strength using Machine Learning techniques and auto ML

Abstract

Concrete is the most important material in civil engineering. Theconcrete compressive strength is a highly nonlinear function of age andingredients. These ingredients include cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, and fine aggregate

Data Characteristics

The actual concrete compressive strength (MPa) for a given mixture under a specific age (days) was determined from laboratory.

Time Line o fthe Project:

- Data Analysis
- Data Preprocessing
- Feature Engineering
- Model Building using DL
- Model Building using Auto Keras

Importing Libraries

```
import pandas as pd
import numpy as np
```

```
from matplotlib import pyplot as plt
%matplotlib inline
import matplotlib
```

```
from google.colab import drive
drive.mount('/content/drive')
df = pd.read_csv("/content/drive/MyDrive/concrete_data.csv")
df.head()
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True)
```

```
df.shape
```

```
(1030, 9)
```

Data Analysis

```
df.describe()
```

```
df.info()
```

```
df.isna().sum()
```

Heatmap

```
import seaborn as sns
plt.figure(figsize=(10,10))
sns.heatmap(df.corr(),annot=True)
```

```
for i in df.columns:
    for j in df.columns:
        plt.figure(figsize=(9,7))
        sns.scatterplot(x=i,y=j,hue="concrete_compressive_strength",data=df)
        plt.show()
```

Outlier Analysis

```
def outlier(data,column):
    plt.figure(figsize=(5,3))
    sns.boxplot(data[column])
    plt.title("{} distribution".format(column))
```

```
for i in df.columns:
    outlier(df,i)
```

Findind the min and max value for every feature

```
def end_value_show(data,column):
    print("min value of {} is {} \nmax value of {} is {}".format(column,data[column].min(),column,data[column].max()))
```

```
for i in df.columns:
    end_value_show(df,i)
```

Replacing the Outliers

```
df=df[df["blast_furnace_slag"]<350]
df=df[(df["water"]<246) & (df["water"]>122)]
df=df[df["superplasticizer"]<25]
df=df[df["age"]<150]
```

Feature Engineering

```
df.columns
```

```
Index(['cement', 'blast_furnace_slag', 'fly_ash', 'water', 'superplasticizer',
       'coarse_aggregate', 'fine_aggregate ', 'age',
       'concrete_compressive_strength'],
      dtype='object')
```

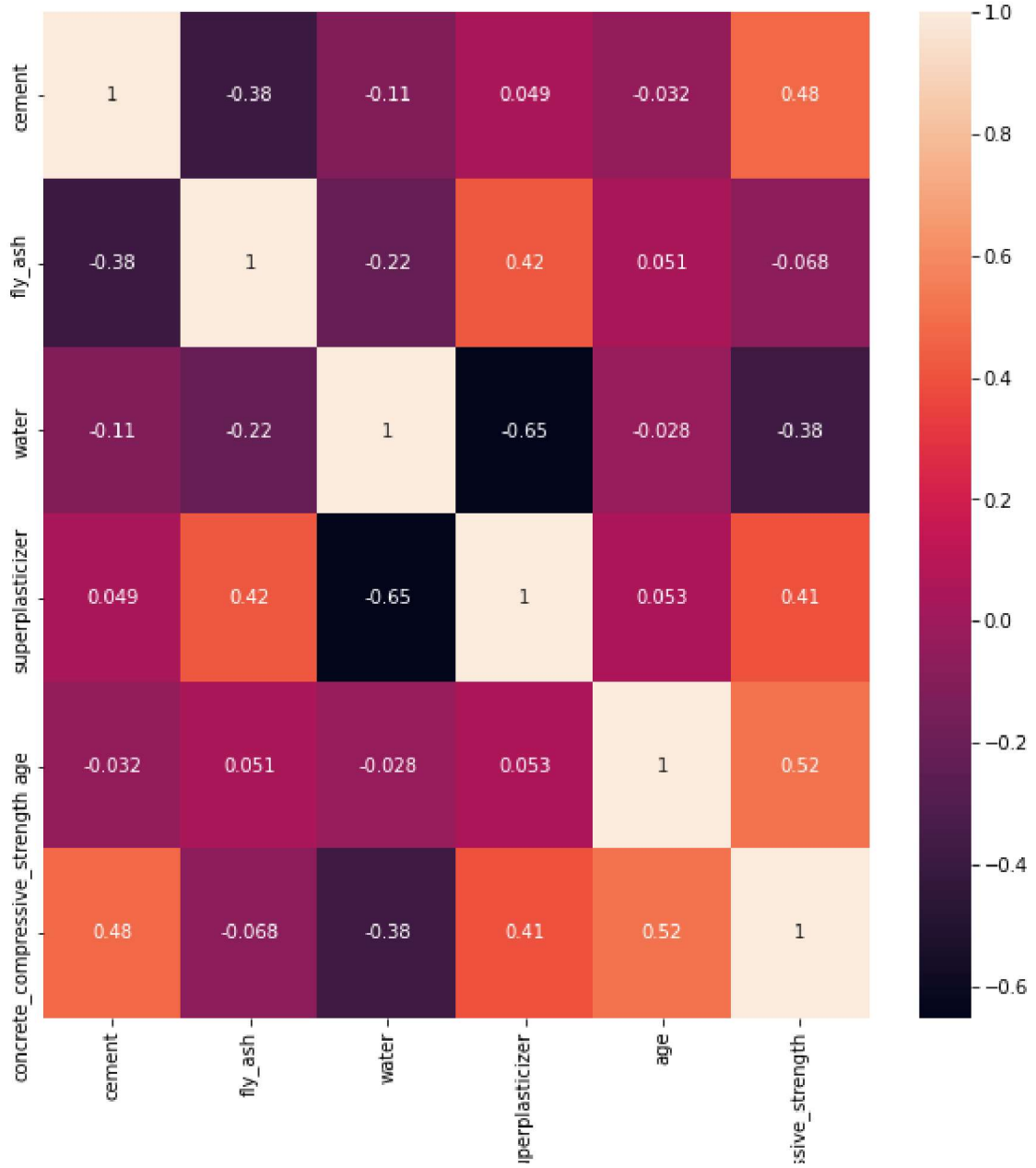
```
df.drop(["blast_furnace_slag"],axis=1,inplace=True)
df.drop(["coarse_aggregate"],axis=1,inplace=True)
df.drop(["fine_aggregate "],axis=1,inplace=True)
```

```
df.columns
```

```
Index(['cement', 'fly_ash', 'water', 'superplasticizer', 'age',
       'concrete_compressive_strength'],
      dtype='object')
```

```
plt.figure(figsize=(10,10))
sns.heatmap(df.corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f58211ba5d0>
```



Splitting the Data

```
x=df.drop(["concrete_compressive_strength"],axis=1)
y=df["concrete_compressive_strength"]
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=42)
```

```
x_train.shape
```

```
(666, 5)
```

Model Building using DL

We will be using Keras Sequential Model for this project

```
from tensorflow.keras import models,layers
```

```
model=models.Sequential()
model.add(layers.Dropout(0.1))
model.add(layers.Dense(100,activation='relu',input_shape=(x_train.iloc[1].shape)))
model.add(layers.Dropout(0.7))
model.add(layers.Dense(5,activation='tanh'))
model.add(layers.Dropout(0.2))

model.add(layers.Dense(1))
model.compile(optimizer='rmsprop',loss='mse',metrics=['mae'])
```

```
model.fit(x_train,y_train,epochs=100,batch_size=1,validation_data=(x_test,y_test))
```

```
model.evaluate(x_test,y_test)
```

```
9/9 [==============================] - 0s 3ms/step - loss: 276.4749 - mae: 13.1592
[276.4749450683594, 13.159246444702148]
```

```
pred=model.predict(x_test)
pred[4]
```

```
array([34.20076], dtype=float32)
```

Using Auto Keras

AutoKeras: An AutoML system based on Keras. It is developed by DATA Lab at Texas A&M University

Installing Auto Keras

```
!pip install git+https://github.com/keras-team/keras-tuner.git@1.0.2rc1
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting git+https://github.com/keras-team/keras-tuner.git@1.0.2rc1
  Cloning https://github.com/keras-team/keras-tuner.git (to revision 1.0.2rc1) to /tmp/pip-req-build-c5lmgiak
  Running command git clone -q https://github.com/keras-team/keras-tuner.git /tmp/pip-req-build-c5lmgiak
  Running command git checkout -q 0fb69434a132093518e0e53d40020145ae192629
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (21.3)
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (0.16.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (1.21.6)
Requirement already satisfied: tabulate in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (0.8.10)
Collecting terminaltables
  Downloading terminaltables-3.1.10-py2.py3-none-any.whl (15 kB)
Collecting colorama
  Downloading colorama-0.4.5-py2.py3-none-any.whl (16 kB)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (4.64.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (2.23.0)
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (1.7.3)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.7/dist-packages (from keras-tuner==1.0.2rc1) (1.0.2)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging->keras-tuner=
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests->keras-tuner==1.0.2rc1) (2
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests->keras-tuner==1.0.2r
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests->keras-tuner==1.0.2rc
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn->keras-tuner==
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn->keras-tuner==1.0.2rc1
Building wheels for collected packages: keras-tuner
  Building wheel for keras-tuner (setup.py) ... done
  Created wheel for keras-tuner: filename=keras_tuner-1.0.2rc1-py3-none-any.whl size=85445 sha256=24dfc93fbf97a8e12ccf43e7a4f6e
  Stored in directory: /tmp/pip-ephem-wheel-cache-7jiogi8s/wheels/44/e5/92/e83049ca00432aec622a4fa0200e254d88aefae9d74aa86941
Successfully built keras-tuner
Installing collected packages: terminaltables, colorama, keras-tuner
Successfully installed colorama-0.4.5 keras-tuner-1.0.2rc1 terminaltables-3.1.10
```

```
!pip install autokeras
```

```
!pip show autokeras
```

```
Name: autokeras
```

```
     Version: 1.0.19
     Summary: AutoML for deep learning
     Home-page: http://autokeras.com
     Author: DATA Lab, Keras Team
     Author-email: jhfjhfj1@gmail.com
     License: Apache License 2.0
     Location: /usr/local/lib/python3.7/dist-packages
     Requires: pandas, keras-tuner, tensorflow, packaging
     Required-by:
```

```
import numpy as np
import pandas as pd
import tensorflow as tf


import autokeras as ak


reg = ak.StructuredDataRegressor(
    overwrite=True, max_trials=3
)


reg.fit(x=x_train, y=y_train, verbose=0)


# evaluate the model
mae, _  = reg.evaluate(x_test, y_test, verbose=0)
#print('MAE: %.3f' % mae)
# use the model to make a prediction
yhat_test = reg.predict(x_test)

# get the best performing model
model = reg.export_model()
```

```
     9/9 [==============================] - 0s 2ms/step
```

```
# summarize the loaded model
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 5)]               0

 multi_category_encoding (Mu  (None, 5)                0
 ltiCategoryEncoding)

 normalization (Normalizatio  (None, 5)                11
 n)

 dense (Dense)               (None, 256)               1536

 re_lu (ReLU)                (None, 256)               0

 dense_1 (Dense)             (None, 32)                8224

 re_lu_1 (ReLU)              (None, 32)                0

 regression_head_1 (Dense)   (None, 1)                 33

=================================================================
Total params: 9,804
Trainable params: 9,793
Non-trainable params: 11
_________________________________________________________________
```

```
yhat_test
```

```
y_test
```

```
248    44.30
469    44.28
757    18.13
826    24.39
```

```
557    17.24
       ...
862    35.23
513    40.29
939    32.72
454    39.64
277    36.97
Name: concrete_compressive_strength, Length: 286, dtype: float64
```