Project 1 – Red Wine Classification

Task : Implement a Red wine quality classifier using a Neural network

In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

In [2]:
```
# Read Data
wine=pd.read_csv('winequality-red.csv')
wine
```

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20	0.68	9.8	5
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26	0.65	9.8	5
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16	0.58	9.8	6
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51	0.56	9.4	5
...	...	...	...	...	...	...	...	...	...	...	...	...
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58	10.5	5
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3.52	0.76	11.2	6
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3.42	0.75	11.0	6
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3.57	0.71	10.2	5
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3.39	0.66	11.0	6

1599 rows × 12 columns

1. Perform EDA, Explore the features using histograms, any data preprocessing required.

In [3]:
```
wine.shape
```

Out[3]:
```
(1599, 12)
```

In [4]:
```
wine.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

In [5]:
```
wine.describe()
```

Out[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	46.467792	0.996747	3.311113
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	32.895324	0.001887	0.154386
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000	0.990070	2.740000
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	22.000000	0.995600	3.210000
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	38.000000	0.996750	3.310000
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	62.000000	0.997835	3.400000
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	289.000000	1.003690	4.010000

In [6]:
```
# check for nulls
wine.isnull().sum()
```

Out[6]:
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

In [7]:
```
# Rename Column name with no space as it will be needed while applying DNN
wine.columns = ['fixed_acidity', 'volatile_acidity', 'citric_acid', 'residual_sugar', 'chlorides', '
wine
```

Out[7]:

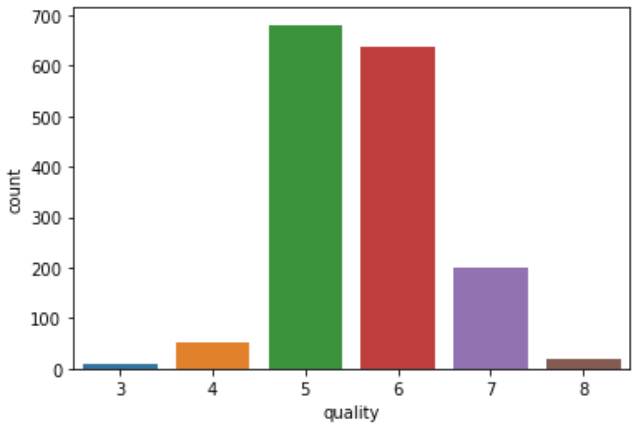
	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	free_sulfur_dioxide	total_sulfur_dioxide	density	
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3
...	...	...	...	...	...	...	...	...	
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3
1595	5.9	0.550	0.10	2.2	0.062	39.0	51.0	0.99512	3
1596	6.3	0.510	0.13	2.3	0.076	29.0	40.0	0.99574	3
1597	5.9	0.645	0.12	2.0	0.075	32.0	44.0	0.99547	3
1598	6.0	0.310	0.47	3.6	0.067	18.0	42.0	0.99549	3

1599 rows × 12 columns

In [8]:
```
wine.value_counts("quality")
```

Out[8]:
```
quality
5    681
6    638
7    199
4     53
8     18
3     10
dtype: int64
```

In [9]:
```
# Visualize the value counts of quality
sns.countplot(x='quality', data=wine, )
plt.show()
```



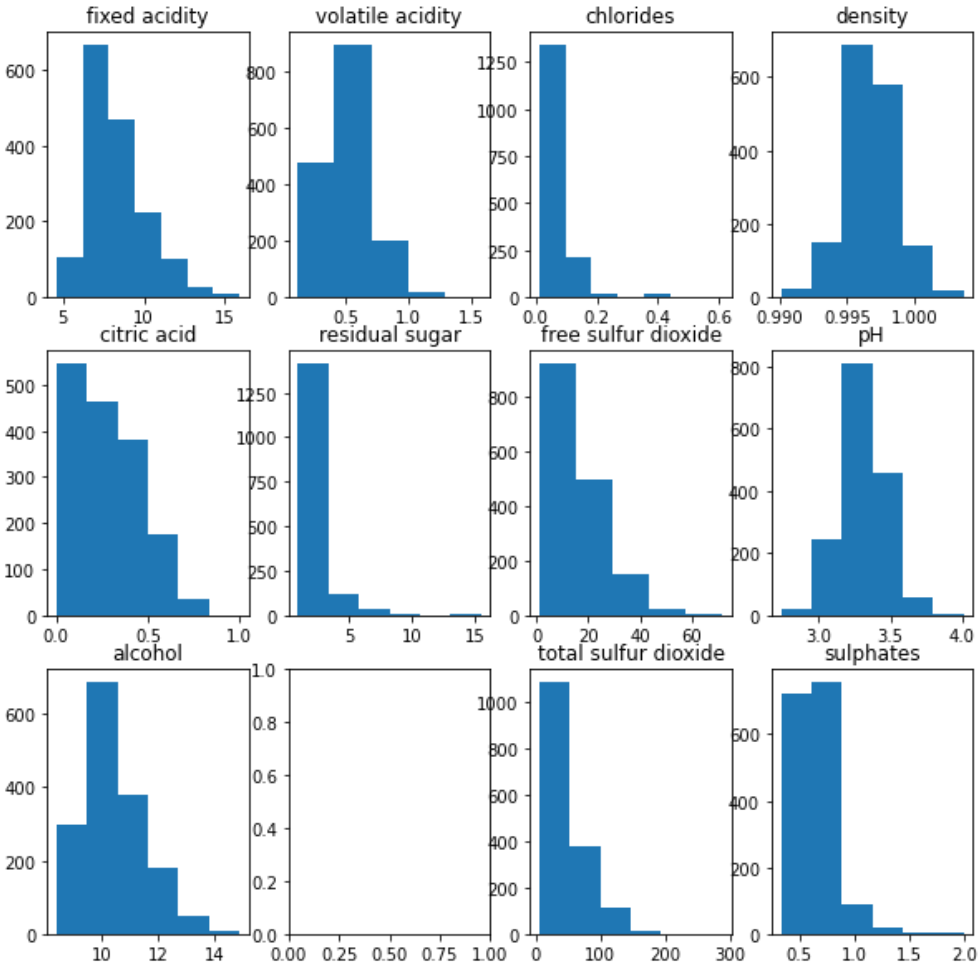
We can see that majority of values are either 5 or 6

In [10]:
```
# mean value of each feature for quality
wine.groupby('quality').mean()
```

Out[10]:

quality	fixed_acidity	volatile_acidity	citric_acid	residual_sugar	chlorides	free_sulfur_dioxide	total_sulfur_dioxide	density
3	8.360000	0.884500	0.171000	2.635000	0.122500	11.000000	24.900000	0.997464
4	7.779245	0.693962	0.174151	2.694340	0.090679	12.264151	36.245283	0.996542
5	8.167254	0.577041	0.243686	2.528855	0.092736	16.983847	56.513950	0.997104
6	8.347179	0.497484	0.273824	2.477194	0.084956	15.711599	40.869906	0.996615
7	8.872362	0.403920	0.375176	2.720603	0.076588	14.045226	35.020101	0.996104
8	8.566667	0.423333	0.391111	2.577778	0.068444	13.277778	33.444444	0.995212

In [11]:
```
# Plot Histogram to check for normal distribution
fig, axes = plt.subplots(3, 4, figsize=(10,10))

axes[0,0].set_title("fixed acidity")
axes[0,0].hist(wine['fixed_acidity'], bins=7)

axes[0,1].set_title("volatile acidity")
axes[0,1].hist(wine['volatile_acidity'], bins=5);

axes[1,0].set_title("citric acid")
axes[1,0].hist(wine['citric_acid'], bins=6);

axes[1,1].set_title("residual sugar")
axes[1,1].hist(wine['residual_sugar'], bins=6);

axes[0,2].set_title("chlorides")
axes[0,2].hist(wine['chlorides'], bins=7)

axes[1,2].set_title("free sulfur dioxide")
axes[1,2].hist(wine['free_sulfur_dioxide'], bins=5);

axes[2,2].set_title("total sulfur dioxide")
axes[2,2].hist(wine['total_sulfur_dioxide'], bins=6);

axes[0,3].set_title("density")
axes[0,3].hist(wine['density'], bins=6);

axes[1,3].set_title("pH")
axes[1,3].hist(wine['pH'], bins=6);

axes[2,3].set_title("sulphates")
axes[2,3].hist(wine['sulphates'], bins=6);

axes[2,0].set_title("alcohol")
axes[2,0].hist(wine['alcohol'], bins=6);
```



2. Implement a Neural Network using TF Estimator DNN Classifier

In [12]:
```
!pip install tensorflow pandas
```

```
Requirement already satisfied: tensorflow in c:\users\eshita gupta\anaconda3\lib\site-packages (2.1
2.0)
Requirement already satisfied: pandas in c:\users\eshita gupta\anaconda3\lib\site-packages (1.3.4)
Requirement already satisfied: tensorflow-intel==2.12.0 in c:\users\eshita gupta\anaconda3\lib\site
-packages (from tensorflow) (2.12.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\eshita gupta\anaconda3\lib\site-packages (fr
om tensorflow-intel==2.12.0->tensorflow) (3.2.1)
Requirement already satisfied: six>=1.12.0 in c:\users\eshita gupta\anaconda3\lib\site-packages (fr
om tensorflow-intel==2.12.0->tensorflow) (1.16.0)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\eshita gupta\anaconda3\lib\site-packag
es (from tensorflow-intel==2.12.0->tensorflow) (3.3.0)
Requirement already satisfied: flatbuffers>=2.0 in c:\users\eshita gupta\anaconda3\lib\site-package
s (from tensorflow-intel==2.12.0->tensorflow) (23.5.26)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\eshita gupta\anaconda3\lib\site-packag
es (from tensorflow-intel==2.12.0->tensorflow) (1.6.3)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\eshita gupta\anaconda3\lib\site
-packages (from tensorflow-intel==2.12.0->tensorflow) (3.10.0.2)
Requirement already satisfied: setuptools in c:\users\eshita gupta\anaconda3\lib\site-packages (fro
m tensorflow-intel==2.12.0->tensorflow) (58.0.4)
Requirement already satisfied: jax>=0.3.15 in c:\users\eshita gupta\anaconda3\lib\site-packages (fr
om tensorflow-intel==2.12.0->tensorflow) (0.4.11)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\eshita gupta\anaconda3\lib\site-pack
ages (from tensorflow-intel==2.12.0->tensorflow) (1.54.2)
Requirement already satisfied: tensorflow-estimator<2.13,>=2.12.0 in c:\users\eshita gupta\anaconda
3\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow) (2.12.0)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0
dev,>=3.20.3 in c:\users\eshita gupta\anaconda3\lib\site-packages (from tensorflow-intel==2.12.0->t
ensorflow) (4.23.2)
Requirement already satisfied: libclang>=13.0.0 in c:\users\eshita gupta\anaconda3\lib\site-package
s (from tensorflow-intel==2.12.0->tensorflow) (16.0.0)
Requirement already satisfied: keras<2.13,>=2.12.0 in c:\users\eshita gupta\anaconda3\lib\site-pack
ages (from tensorflow-intel==2.12.0->tensorflow) (2.12.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\eshita gupta\anaconda3\lib\site-packages
(from tensorflow-intel==2.12.0->tensorflow) (1.4.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\eshita gupta\anaconda3\lib\site-package
s (from tensorflow-intel==2.12.0->tensorflow) (2.3.0)
Requirement already satisfied: packaging in c:\users\eshita gupta\anaconda3\lib\site-packages (from
tensorflow-intel==2.12.0->tensorflow) (21.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in c:\users\eshita gupta\anacon
da3\lib\site-packages (from tensorflow-intel==2.12.0->tensorflow) (0.31.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in c:\users\eshita gupta\anaconda3\lib\site-pack
ages (from tensorflow-intel==2.12.0->tensorflow) (1.12.1)
Requirement already satisfied: numpy<1.24,>=1.22 in c:\users\eshita gupta\anaconda3\lib\site-packag
es (from tensorflow-intel==2.12.0->tensorflow) (1.22.4)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in c:\users\eshita gupta\anaconda3\lib\site-pack
ages (from tensorflow-intel==2.12.0->tensorflow) (0.4.0)
Requirement already satisfied: tensorboard<2.13,>=2.12 in c:\users\eshita gupta\anaconda3\lib\site-
packages (from tensorflow-intel==2.12.0->tensorflow) (2.12.3)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\eshita gupta\anaconda3\lib\site-pack
ages (from tensorflow-intel==2.12.0->tensorflow) (0.2.0)
Requirement already satisfied: pytz>=2017.3 in c:\users\eshita gupta\anaconda3\lib\site-packages (f
rom pandas) (2021.3)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\eshita gupta\anaconda3\lib\site-p
ackages (from pandas) (2.8.2)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\eshita gupta\anaconda3\lib\site-packa
ges (from astunparse>=1.6.0->tensorflow-intel==2.12.0->tensorflow) (0.37.0)
Requirement already satisfied: scipy>=1.7 in c:\users\eshita gupta\anaconda3\lib\site-packages (fro
m jax>=0.3.15->tensorflow-intel==2.12.0->tensorflow) (1.7.1)
Requirement already satisfied: importlib-metadata>=4.6 in c:\users\eshita gupta\anaconda3\lib\site-
packages (from jax>=0.3.15->tensorflow-intel==2.12.0->tensorflow) (4.8.1)
Requirement already satisfied: ml-dtypes>=0.1.0 in c:\users\eshita gupta\anaconda3\lib\site-package
s (from jax>=0.3.15->tensorflow-intel==2.12.0->tensorflow) (0.1.0)
Requirement already satisfied: zipp>=0.5 in c:\users\eshita gupta\anaconda3\lib\site-packages (from
importlib-metadata>=4.6->jax>=0.3.15->tensorflow-intel==2.12.0->tensorflow) (3.6.0)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\eshita gupta\anaconda3\lib\site-pa
ckages (from tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (2.19.1)
Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in c:\users\eshita gupta\anaconda3\li
b\site-packages (from tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (1.0.0)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\eshita gupta\anaconda3\lib\site-pack
```

```
ages (from tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (2.26.0)
Requirement already satisfied: markdown>=2.6.8 in c:\users\eshita gupta\anaconda3\lib\site-packages
(from tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (3.4.3)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\eshita gupta\anaconda3\lib\site-packages
(from tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (2.0.2)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\users\eshita gupta\anaco
nda3\lib\site-packages (from tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (0.7.0)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\eshita gupta\anaconda3\lib\site-packages
(from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (4.9)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\eshita gupta\anaconda3\lib\site-pa
ckages (from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow)
(0.3.0)
Requirement already satisfied: urllib3<2.0 in c:\users\eshita gupta\anaconda3\lib\site-packages (fr
om google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (1.26.7)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\eshita gupta\anaconda3\lib\site-p
ackages (from google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow)
(5.3.1)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\eshita gupta\anaconda3\lib\site
-packages (from google-auth-oauthlib<1.1,>=0.5->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->
tensorflow) (1.3.1)
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in c:\users\eshita gupta\anaconda3\lib\site-pac
kages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.13,>=2.12->tensorflow-intel
==2.12.0->tensorflow) (0.5.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\eshita gupta\anaconda3\lib\site-packa
ges (from requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (202
1.10.8)
Requirement already satisfied: idna<4,>=2.5 in c:\users\eshita gupta\anaconda3\lib\site-packages (f
rom requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflow) (3.2)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\users\eshita gupta\anaconda3\lib\sit
e-packages (from requests<3,>=2.21.0->tensorboard<2.13,>=2.12->tensorflow-intel==2.12.0->tensorflo
w) (2.0.4)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\eshita gupta\anaconda3\lib\site-packages
(from requests-oauthlib>=0.7.0->google-auth-oauthlib<1.1,>=0.5->tensorboard<2.13,>=2.12->tensorflow
-intel==2.12.0->tensorflow) (3.2.2)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\eshita gupta\anaconda3\lib\site-package
s (from packaging->tensorflow-intel==2.12.0->tensorflow) (3.0.4)
```

In [13]:
```
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

In [14]:
```
#Splitting Features and Target in dataset
features = wine.drop('quality', axis=1)
target = wine['quality']
```

In [15]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2, random_state=42)
```

In [16]:
```
# Scale the features using StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [18]:
```
# Create input functions
train_input_fn = tf.compat.v1.estimator.inputs.numpy_input_fn(
    x={"x": X_train_scaled},
    y=y_train.values,
    batch_size=32,
    num_epochs=None,
    shuffle=True
)

eval_input_fn = tf.compat.v1.estimator.inputs.numpy_input_fn(
    x={"x": X_test_scaled},
    y=y_test.values,
    num_epochs=1,
    shuffle=False
)
```

In [21]:
```
# Define the feature columns
feature_columns = [tf.feature_column.numeric_column("x", shape=[X_train_scaled.shape[1]])]
```

In [24]:
```
# Create the DNN Classifier
estimator = tf.estimator.DNNClassifier(
    feature_columns=feature_columns,
    hidden_units=[128, 64],
    n_classes=10,
    model_dir='wine_model'
)
```

```
INFO:tensorflow:Using default config.
INFO:tensorflow:Using config: {'_model_dir': 'wine_model', '_tf_random_seed': None, '_save_summary_
steps': 100, '_save_checkpoints_steps': None, '_save_checkpoints_secs': 600, '_session_config': all
ow_soft_placement: true
graph_options {
  rewrite_options {
    meta_optimizer_iterations: ONE
  }
}
, '_keep_checkpoint_max': 5, '_keep_checkpoint_every_n_hours': 10000, '_log_step_count_steps': 100,
'_train_distribute': None, '_device_fn': None, '_protocol': None, '_eval_distribute': None, '_exper
imental_distribute': None, '_experimental_max_worker_delay_secs': None, '_session_creation_timeout_
secs': 7200, '_checkpoint_save_graph_def': True, '_service': None, '_cluster_spec': ClusterSpec
({}), '_task_type': 'worker', '_task_id': 0, '_global_id_in_cluster': 0, '_master': '', '_evaluatio
n_master': '', '_is_chief': True, '_num_ps_replicas': 0, '_num_worker_replicas': 1}
```

In [26]:
```
# Train the model
estimator.train(input_fn=train_input_fn, steps=1000)
```

```
INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Create CheckpointSaverHook.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from wine_model\model.ckpt-1000
WARNING:tensorflow:From C:\Users\Eshita Gupta\anaconda3\lib\site-packages\tensorflow\python\trainin
g\saver.py:1176: get_checkpoint_mtimes (from tensorflow.python.checkpoint.checkpoint_management) is
deprecated and will be removed in a future version.
Instructions for updating:
Use standard file utilities to get mtimes.
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Calling checkpoint listeners before saving checkpoint 1000...
INFO:tensorflow:Saving checkpoints for 1000 into wine_model\model.ckpt.
INFO:tensorflow:Calling checkpoint listeners after saving checkpoint 1000...
INFO:tensorflow:loss = 1.0303125, step = 1000
INFO:tensorflow:global_step/sec: 498.522
INFO:tensorflow:loss = 1.3093879, step = 1100 (0.201 sec)
INFO:tensorflow:global_step/sec: 916.844
INFO:tensorflow:loss = 1.2990022, step = 1200 (0.109 sec)
INFO:tensorflow:global_step/sec: 648.616
INFO:tensorflow:loss = 1.3165629, step = 1300 (0.156 sec)
INFO:tensorflow:global_step/sec: 560.744
INFO:tensorflow:loss = 1.1810496, step = 1400 (0.178 sec)
INFO:tensorflow:global_step/sec: 504.883
INFO:tensorflow:loss = 1.364764, step = 1500 (0.196 sec)
INFO:tensorflow:global_step/sec: 459.826
INFO:tensorflow:loss = 1.0783223, step = 1600 (0.220 sec)
INFO:tensorflow:global_step/sec: 596.638
INFO:tensorflow:loss = 0.88589084, step = 1700 (0.166 sec)
INFO:tensorflow:global_step/sec: 1033.3
INFO:tensorflow:loss = 0.8144909, step = 1800 (0.097 sec)
INFO:tensorflow:global_step/sec: 1122.28
INFO:tensorflow:loss = 0.98174155, step = 1900 (0.088 sec)
INFO:tensorflow:Calling checkpoint listeners before saving checkpoint 2000...
INFO:tensorflow:Saving checkpoints for 2000 into wine_model\model.ckpt.
INFO:tensorflow:Calling checkpoint listeners after saving checkpoint 2000...
INFO:tensorflow:Loss for final step: 1.0910778.
```

Out[26]:
```
<tensorflow_estimator.python.estimator.canned.dnn.DNNClassifierV2 at 0x1bd96c547c0>
```

In [27]:
```
# Evaluate the model
eval_result = estimator.evaluate(input_fn=eval_input_fn)
```

```
INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Starting evaluation at 2023-06-04T22:44:14
WARNING:tensorflow:From C:\Users\Eshita Gupta\anaconda3\lib\site-packages\tensorflow\python\trainin
g\evaluation.py:260: FinalOpsHook.__init__ (from tensorflow.python.training.basic_session_run_hook
s) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.keras instead.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from wine_model\model.ckpt-2000
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Inference Time : 0.32884s
INFO:tensorflow:Finished evaluation at 2023-06-04-22:44:15
INFO:tensorflow:Saving dict for global step 2000: accuracy = 0.56875, average_loss = 1.0994965, glo
bal_step = 2000, loss = 1.0936989
INFO:tensorflow:Saving 'checkpoint_path' summary for global step 2000: wine_model\model.ckpt-2000
```

In [28]:
```
# Print the evaluation metrics
print("Evaluation results:")
for key, value in eval_result.items():
    print(f"{key}: {value}")
```

```
Evaluation results:
accuracy: 0.5687500238418579
average_loss: 1.0994964838027954
loss: 1.0936988592147827
global_step: 2000
```

3. Implement a Neural Network using TF.Keras:

a. Plot training loss and validation loss

b. Plot training accuracy and validation accuracy

In [40]:
```
# Normalize the features
X_train = (X_train - X_train.mean()) / X_train.std()
X_test = (X_test - X_test.mean()) / X_test.std()
```

In [41]:
```
# Convert target to categorical
y_train_cat = tf.keras.utils.to_categorical(y_train, num_classes=10)
y_test_cat = tf.keras.utils.to_categorical(y_test, num_classes=10)
```

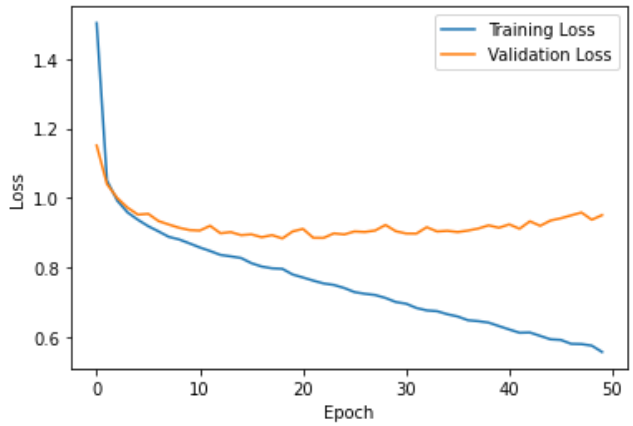
In [42]:
```
# Define the model
model = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='relu', input_shape=(11,)),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(10, activation='softmax')
])
```

In [43]:
```
# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

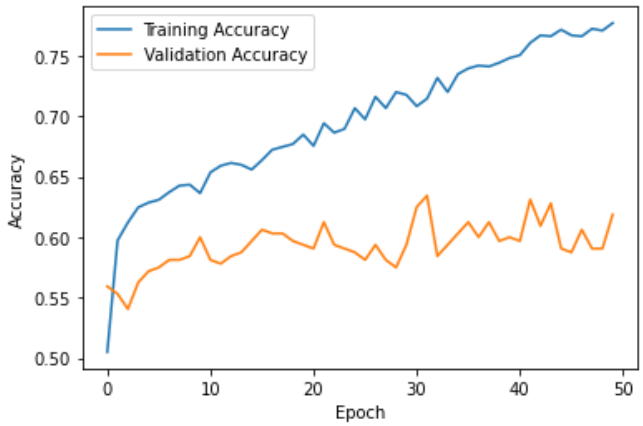
In [44]:
```
# Train the model
history = model.fit(X_train, y_train_cat,
                    validation_data=(X_test, y_test_cat),
                    epochs=50, batch_size=32)
```

```
40/40 [==============================] - 0s 3ms/step - loss: 0.6997 - accuracy: 0.7177 - val_los
s: 0.9038 - val_accuracy: 0.5938
Epoch 31/50
40/40 [==============================] - 0s 3ms/step - loss: 0.6949 - accuracy: 0.7084 - val_los
s: 0.8973 - val_accuracy: 0.6250
Epoch 32/50
40/40 [==============================] - 0s 4ms/step - loss: 0.6825 - accuracy: 0.7146 - val_los
s: 0.8970 - val_accuracy: 0.6344
Epoch 33/50
40/40 [==============================] - 0s 3ms/step - loss: 0.6756 - accuracy: 0.7318 - val_los
s: 0.9154 - val_accuracy: 0.5844
Epoch 34/50
40/40 [==============================] - 0s 3ms/step - loss: 0.6733 - accuracy: 0.7201 - val_los
s: 0.9031 - val_accuracy: 0.5938
Epoch 35/50
40/40 [==============================] - 0s 3ms/step - loss: 0.6645 - accuracy: 0.7349 - val_los
s: 0.9050 - val_accuracy: 0.6031
Epoch 36/50
40/40 [==============================] - 0s 3ms/step - loss: 0.6580 - accuracy: 0.7396 - val_los
s: 0.9015 - val_accuracy: 0.6125
```

In [45]:
```
# Plot training and validation loss
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()
```



In [46]:
```
# Plot training and validation accuracy
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



4. Conclusion:

After Implementing the neural network classification on given red-wine dataset we can conclude,

The training loss and validation loss plots demonstrate the decreasing loss across the epochs, demonstrating the model's ability to learn and enhance its predictions. The training and validation losses both reduced, indicating that the model does not overfit the training set and is capable of generalisation.

Whereas, the training accuracy and validation accuracy plots show a rising tendency. This shows that on both the training and validation datasets, the model was able to recognise patterns and generate precise predictions. The model appears to be operating well on unobserved data because the validation accuracy roughly tracks the training accuracy.

However, to make more robust conclusions about its effectiveness, further analysis and comparison with alternative approaches will be necessary.

In [ ]: