Shallow Neural Network with the Sequential API: Takeaways

Syntax

- List the datasets available within the `seaborn` library:

```
import seaborn as sns
sns.get_dataset_names()
```

- Load the `mpg` dataset available within `seaborn` and checking the columns:

```
df = sns.load_dataset('mpg')
df.columns
```

- Summary statistics of the variable `mpg` :

```
df['mpg'].describe()
```

- The first and last five rows of the dataframe:

```
df.head()
df.tail()
```

- Visualize the categorical variables with the `countplot()` function:

```
sns.countplot(x='variable', data=df)
```

- Visualize the numerical variables with the `histplot()` function:

```
sns.histplot(data=df, x='variable', bins=50).set(title='Title of Chart', ylabel='Title of Y-
axis')
```

- Generate a boxplot for `horsepower` :

```
sns.boxplot(data=df, y='horsepower')
plt.title('Boxplot for horsepower')
plt.show()
```

- Generate a scatterplot for `displacement` with respect to `weight` :

```
plt.scatter(data=df, x='weight', y='displacement')
plt.title('displacement with respect to weight')
plt.ylabel('displacement')
plt.xlabel('weight')
plt.show()
```

- Generate a boxplot for `mpg` , grouped by `origin` :

```
sns.boxplot(data=df, x='origin', y='mpg')
plt.title('Boxplot of mpg grouped by origin')
plt.show()
```

- Print the total missing values for each variable:

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

- Missing value imputation of a numerical variable `horsepower` with its mean:

```
df['horsepower'].fillna(df['horsepower'].mean(), inplace=True)
df.isna().sum()
```

- Drop an irrelevant variable:

```
df = df.drop(['name'], axis=1)
```

- Print the skewness value of a variable, `mpg` :

```
print(df['mpg'].skew())
```

- Replace the outliers in `variablename` above the cutoff value `c` , with the value of `d` ; otherwise keep the original `df['variablename']` value:

```
df['variablename'] = np.where(df['variablename'] > c, d, df['variablename'])
```

- Perform dummy encoding of a given variable, for example `origin` :

```
df = pd.get_dummies(df, columns=['origin'], drop_first=True, prefix='Origin')
```

- The lines of code below create an object of the target variable, `mpg` , as well as the list of all the features, excluding the target variable:

```
target_variable = ['mpg']
predictors = list(set(list(df.columns)) - set(target_variable))
```

- Normalize the predictors and scale them to values between `0` and `1` :

```
df[predictors] = df[predictors] / df[predictors].max()
```

- Create arrays for predictor and target variables, and store them as `X` and `y` , respectively:

```
X = df[predictors].values
y = df[target_variable].values
```

- Define a shallow neural network containing one hidden layer and one ourput layer using the Sequential API:

```
model = tf.keras.Sequential()
model.add(tf.keras.layers.Dense(32, input_shape=(X_train.shape[1],), activation='relu'))
model.add(tf.keras.layers.Dense(1))
```

- Compile a model with the Adam optimizer and mean absolute error as the loss function:

```
optimizer = tf.keras.optimizers.Adam(learning_rate=0.01)
model.compile(loss='mae', metrics=['mae'], optimizer=optimizer)
```

- Fit a model on the train set:

```
model.fit(X_train, y_train)
```

- Model evaluation on train and test datasets:

```
model.evaluate(X_train, y_train)
model.evaluate(X_test, y_test)
```

- Use a model to make predictions on the train and test datasets and compute the R-squared value:

```
pred_train = model.predict(X_train)
r2_score(y_train, pred_train)
pred_test = model.predict(X_test)
r2_score(y_test, pred_test)
```

Concepts

- Sequential API is used to build machine learning models in a sequential way using Keras, which is a high-level API written in Python that allows you to easily construct and train deep learning models in TensorFlow 2.0. Keras has a simple, consistent interface that makes it easy to use, while still allowing you to build complex models.
- In Sequential API, we first add a layer to the model using the `Sequential.add()` method, then another layer, and so on, until we have built our desired model architecture. Once all the layers have been added, we can then compile the model and start training it on our data.
- Seaborn comes with a number of important datasets in its library. Once `seaborn` is imported, these datasets are automatically available and ready for use!
- Exploratory data analysis starts with descriptive statistics, a powerful technique for understanding the key statistical measures of the variables.
- A histogram is a graphical representation of numerical data that shows the distribution of values in a dataset. It is a type of bar chart that shows the number of occurrences of each value in a dataset.
- Outlier identification is a method used in exploratory analysis that deals with finding unusual or unexpected patterns in data.
- A scatterplot is a type of chart used to show the relationship between two numerical variables. Scatterplots are often used to show how one variable (the dependent variable) is affected by another variable (the independent variable).
- Missing value treatment is important because if not performed, it can introduce bias into the machine learning model. This can impact the accuracy of predictions and the interpretability of results.
- The columns that do not add predictive power to machine learning models are referred to as irrelevant columns, and should be dropped from the machine learning process.
- Skewness is a statistical measure of the presence of outliers. The skewness value, when it's between -1 and +1, indicates that the variable does not have significant outliers to worry about.
- For modeling purposes, all the variables should be numeric. One way we can convert categorical variables into numeric features is using a technique called dummy encoding.
- Since the scale of the independent variables can often differ, it is a good idea to bring them to a uniform scale. One such technique is called normalization, which scales the independent variables to values between `0` and `1`.
- Backpropagation is a popular algorithm used in deep neural networks to adjust weights between layers of the network.
- For each layer, backpropagation calculates the error of the output compared to the expected result and uses that information to adjust the weights for future iterations.
- Holdout validation approach in machine learning modeling is a type of model validation technique used to estimate the accuracy of a model on unseen data. The holdout validation approach works by splitting the dataset into two parts: a training set and a test set.
- Adam is a popular optimizer used in many neural networks. Adam is an extension of the well-known stochastic gradient descent (SGD) algorithm.

- SGD is a simple and efficient approach to minimizing an objective function that has smooth, bounded gradients. However, SGD can be difficult to use when training deep neural networks because it requires careful tuning of the learning rate.
- Mean absolute error is a measure of how far predictions or estimates are from the actual value. It is the average of the absolute errors. Error is defined as the difference between the predicted value and the actual value.
- The mean squared error is a statistical measure which represents the average of the squares of the errors, or deviations, between the estimated values and what is actually observed.
- R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.
- The R-squared value ranges from 0 to 1, with 0 indicating that the model does not explain any of the variability in the target variable, and 1 indicating that the model explains all of the variability in the target variable.

Resources

- Countplot
- Sequential Model
- Keras Model Training APIs