Marble Runs & Cat Brains

How Neural Networks Learn

Prompts

- Create a new column "yammy", where “star_rating” >= 4.5, it's a 1 , otherwise 0.
- Count 0s and 1s in the "yammy" column
- Scale the data
- Build a feed-forward neural network with "yammy" as the target

Strategy

- Prepare the data
- Build a simple neural network, compile the model, train it, evaluate, and predict
- Check the Accuracy and Read the Confusion Matrix
- Visualize training

Code & Results

1. Prepare the data

This code separates features and target.

```python
import pandas as pd

# Assuming you already have a DataFrame named df
data['yammy'] = (data['star_rating'] >= 4.5).astype(int)
data.head()
```

This code scales numeric data.

```python
from sklearn.preprocessing import MinMaxScaler

# Select only numeric columns (excluding 'yammy' if it's a label)
features = data.drop(columns=['yammy'])  # Drop target column if needed
numeric_cols = features.select_dtypes(include=['float64', 'int64']).columns

scaler = MinMaxScaler()
data[numeric_cols] = scaler.fit_transform(data[numeric_cols])

data[numeric_cols].head()
```

This code train-test splits.

```python
from sklearn.model_selection import train_test_split

# Separate features and target
X = data.drop(columns=['yammy','star_rating'])
y = data['yammy']

# Split into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

2. Build and evaluate the Model

This code builds a simple neural network.

```python
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.optimizers import Adam

# 5. Build a simple neural network
model = Sequential([
    Dense(32, input_dim=X_train.shape[1], activation='relu'),
    Dropout(0.2),
    Dense(16, activation='relu'),
    Dense(1, activation='sigmoid')  # binary output
])

# 6. Compile the model
model.compile(
    optimizer=Adam(learning_rate=0.001),
    loss='binary_crossentropy',
    metrics=['accuracy']
)

# 7. Train
history = model.fit(
    X_train, y_train,
    epochs=30,
    batch_size=16,
    validation_split=0.2,
    verbose=1
)

# 8. Evaluate
loss, accuracy = model.evaluate(X_test, y_test)
print(f"Test Accuracy: {accuracy:.2f}")

# 9. Predict
preds = (model.predict(X_test) > 0.5).astype(int)
```

This code evaluates accuracy and the confusion matrix.

```python
y_pred = preds

# Evaluation metrics
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

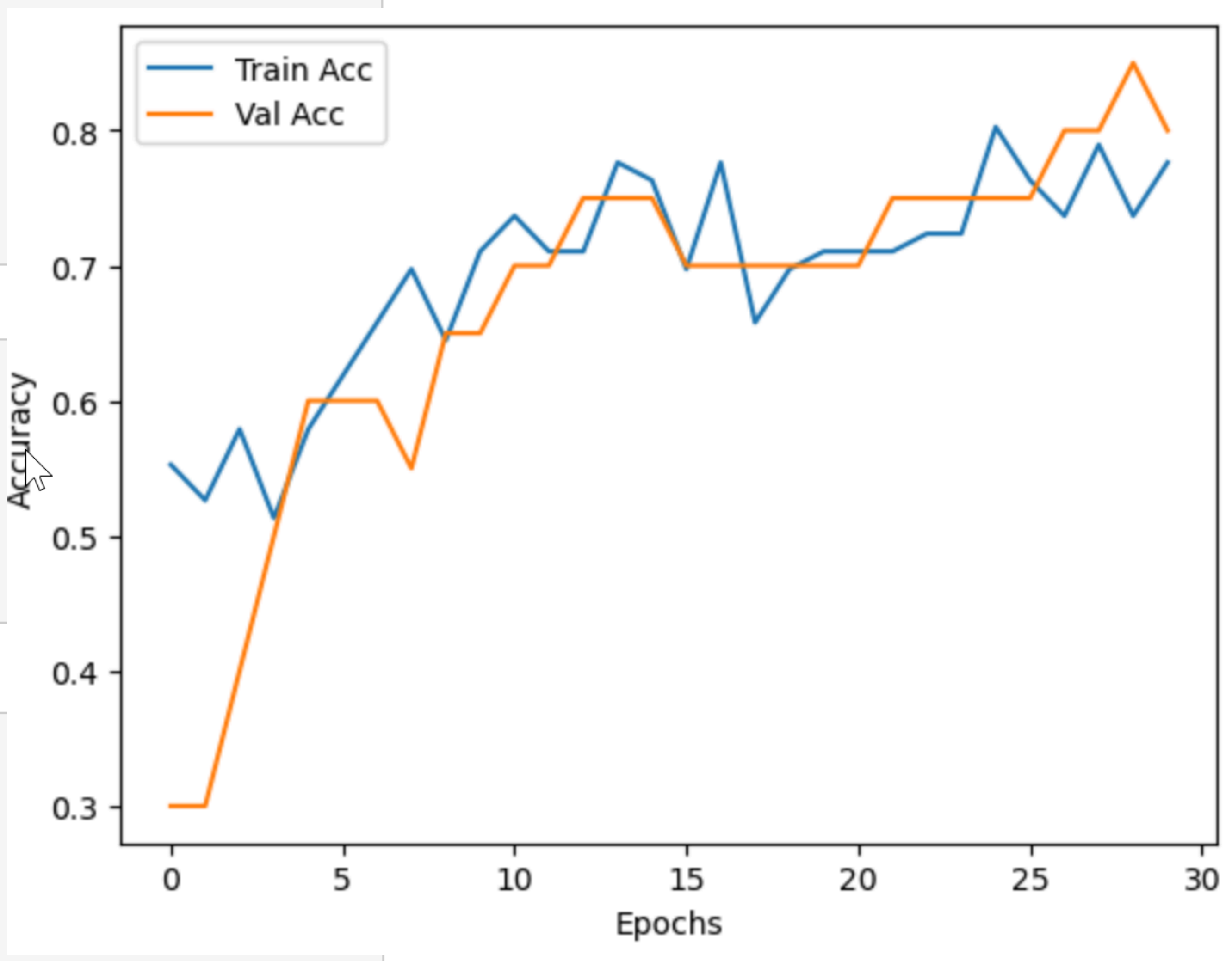
```
Accuracy: 0.84
Confusion Matrix:
 [[12  3]
 [ 1  9]]
```

✅The model got 84% of the tins right.
❌It mislabeled 3 out of 15 meh tins as yummy
It mislabeled 1 out of 10 yammy tins as meh

This code visualizes training.

```python
import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'], label='Train Acc')
plt.plot(history.history['val_accuracy'], label='Val Acc')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Made by: @katerynakononova
Website: https://katerynakononova.github.io/meowlearning/
Book: https://www.amazon.com/dp/B0CW9SFYXF