Phoenix NDVI and EPA landcove Deeplearning model and ermutation_importance

In [ ]:
```
import numpy as np
import tensorflow as tf
np.random.seed(42)
tf.random.set_seed(42)
import random
random.seed(42)
```

In [ ]:
```
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
```

In [ ]:
```
# Load the data
file_path = r"C:\Users\nahid\Desktop\USFS\Project_USFS\Data\PHOENIX_AZ_2010_ndvi_me
df = pd.read_csv(file_path)
```

In [ ]:
```
df.columns
```

Out[ ]:
```
Index(['NDVIMAX', 'NDVIMEAN', 'NDVIMEDIAN', 'X', 'Y', 'WATER', 'IMPERVIOUS',
       'SOIL_BARREN', 'TREES_FOREST', 'SHRUBS', 'GRASS_HERBACEOUS',
       'AGRICULTURE'],
      dtype='object')
```

In [ ]:
```
# Select the dependent variable (NDVI)
y = df['NDVIMEAN']

# Select the independent variables (land cover classifications)
X = df[['WATER', 'IMPERVIOUS', 'SOIL_BARREN', 'TREES_FOREST', 'SHRUBS',
        'GRASS_HERBACEOUS', 'AGRICULTURE']]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta

# Standardize the data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [ ]:
```
# Build the model
model = Sequential()

# Input layer
model.add(Dense(64, input_dim=X_train_scaled.shape[1], activation='relu'))
```

```
# Hidden layers
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.3))

# Output layer
model.add(Dense(1, activation='linear'))

# Compile the model
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
```

```
c:\Users\nahid\Desktop\USFS\Project_USFS\venv\lib\site-packages\keras\src\layers\cor
e\dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a l
ayer. When using Sequential models, prefer using an `Input(shape)` object as the fir
st layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

In [ ]:
```
# Train the model
history = model.fit(X_train_scaled, y_train, epochs=100, batch_size=32, validation_
```

```
Epoch 1/100
91848/91848 - 73s - 797us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0055 - val
_mae: 0.0527
Epoch 2/100
91848/91848 - 72s - 782us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0055 - val
_mae: 0.0529
Epoch 3/100
91848/91848 - 70s - 764us/step - loss: 0.0057 - mae: 0.0536 - val_loss: 0.0055 - val
_mae: 0.0529
Epoch 4/100
91848/91848 - 70s - 765us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0055 - val
_mae: 0.0530
Epoch 5/100
91848/91848 - 70s - 764us/step - loss: 0.0057 - mae: 0.0536 - val_loss: 0.0055 - val
_mae: 0.0527
Epoch 6/100
91848/91848 - 70s - 766us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0055 - val
_mae: 0.0527
Epoch 7/100
91848/91848 - 70s - 766us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0056 - val
_mae: 0.0533
Epoch 8/100
91848/91848 - 68s - 744us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0055 - val
_mae: 0.0530
Epoch 9/100
91848/91848 - 70s - 760us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0056 - val
_mae: 0.0531
Epoch 10/100
91848/91848 - 71s - 776us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0056 - val
_mae: 0.0530
Epoch 11/100
91848/91848 - 68s - 742us/step - loss: 0.0058 - mae: 0.0537 - val_loss: 0.0055 - val
_mae: 0.0529
Epoch 12/100
91848/91848 - 70s - 760us/step - loss: 0.0058 - mae: 0.0537 - val_loss: 0.0056 - val
_mae: 0.0532
Epoch 13/100
91848/91848 - 68s - 744us/step - loss: 0.0058 - mae: 0.0536 - val_loss: 0.0055 - val
_mae: 0.0527
Epoch 14/100
91848/91848 - 69s - 748us/step - loss: 0.0058 - mae: 0.0537 - val_loss: 0.0055 - val
_mae: 0.0529
Epoch 15/100
91848/91848 - 69s - 750us/step - loss: 0.0058 - mae: 0.0537 - val_loss: 0.0056 - val
_mae: 0.0531
Epoch 16/100
91848/91848 - 68s - 746us/step - loss: 0.0058 - mae: 0.0537 - val_loss: 0.0056 - val
_mae: 0.0532
Epoch 17/100
91848/91848 - 68s - 744us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0056 - val
_mae: 0.0529
Epoch 18/100
91848/91848 - 125s - 1ms/step - loss: 0.0058 - mae: 0.0538 - val_loss: 0.0056 - val_
mae: 0.0531
Epoch 19/100
91848/91848 - 79s - 858us/step - loss: 0.0058 - mae: 0.0538 - val_loss: 0.0056 - val
```

```
_mae: 0.0533
Epoch 20/100
91848/91848 - 71s - 772us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0534
Epoch 21/100
91848/91848 - 72s - 788us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0535
Epoch 22/100
91848/91848 - 70s - 767us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0538
Epoch 23/100
91848/91848 - 70s - 765us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0535
Epoch 24/100
91848/91848 - 71s - 768us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0537
Epoch 25/100
91848/91848 - 71s - 770us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0056 - val
_mae: 0.0536
Epoch 26/100
91848/91848 - 72s - 782us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0536
Epoch 27/100
91848/91848 - 69s - 749us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0057 - val
_mae: 0.0534
Epoch 28/100
91848/91848 - 69s - 751us/step - loss: 0.0059 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0536
Epoch 29/100
91848/91848 - 70s - 759us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0540
Epoch 30/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0540
Epoch 31/100
91848/91848 - 70s - 758us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0056 - val
_mae: 0.0535
Epoch 32/100
91848/91848 - 69s - 751us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0056 - val
_mae: 0.0535
Epoch 33/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0538 - val_loss: 0.0057 - val
_mae: 0.0535
Epoch 34/100
91848/91848 - 69s - 748us/step - loss: 0.0058 - mae: 0.0538 - val_loss: 0.0057 - val
_mae: 0.0542
Epoch 35/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0538 - val_loss: 0.0056 - val
_mae: 0.0533
Epoch 36/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0057 - val
_mae: 0.0543
Epoch 37/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0056 - val
_mae: 0.0530
Epoch 38/100
```

```
91848/91848 - 70s - 758us/step - loss: 0.0058 - mae: 0.0538 - val_loss: 0.0056 - val
_mae: 0.0536
Epoch 39/100
91848/91848 - 70s - 760us/step - loss: 0.0058 - mae: 0.0538 - val_loss: 0.0056 - val
_mae: 0.0530
Epoch 40/100
91848/91848 - 69s - 753us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0057 - val
_mae: 0.0543
Epoch 41/100
91848/91848 - 69s - 752us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0057 - val
_mae: 0.0546
Epoch 42/100
91848/91848 - 69s - 750us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0057 - val
_mae: 0.0541
Epoch 43/100
91848/91848 - 68s - 744us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0057 - val
_mae: 0.0539
Epoch 44/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0057 - val
_mae: 0.0536
Epoch 45/100
91848/91848 - 69s - 754us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0056 - val
_mae: 0.0532
Epoch 46/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0057 - val
_mae: 0.0540
Epoch 47/100
91848/91848 - 69s - 753us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0056 - val
_mae: 0.0535
Epoch 48/100
91848/91848 - 69s - 749us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0056 - val
_mae: 0.0533
Epoch 49/100
91848/91848 - 68s - 744us/step - loss: 0.0058 - mae: 0.0539 - val_loss: 0.0056 - val
_mae: 0.0531
Epoch 50/100
91848/91848 - 69s - 749us/step - loss: 0.0058 - mae: 0.0540 - val_loss: 0.0056 - val
_mae: 0.0534
Epoch 51/100
91848/91848 - 70s - 758us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0056 - val
_mae: 0.0534
Epoch 52/100
91848/91848 - 69s - 749us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0546
Epoch 53/100
91848/91848 - 69s - 749us/step - loss: 0.0059 - mae: 0.0542 - val_loss: 0.0058 - val
_mae: 0.0548
Epoch 54/100
91848/91848 - 69s - 747us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0545
Epoch 55/100
91848/91848 - 69s - 748us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0057 - val
_mae: 0.0542
Epoch 56/100
91848/91848 - 69s - 748us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0540
```

```
Epoch 57/100
91848/91848 - 69s - 749us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0545
Epoch 58/100
91848/91848 - 69s - 751us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0541
Epoch 59/100
91848/91848 - 69s - 753us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0543
Epoch 60/100
91848/91848 - 69s - 751us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0538
Epoch 61/100
91848/91848 - 69s - 750us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0541
Epoch 62/100
91848/91848 - 70s - 757us/step - loss: 0.0058 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0539
Epoch 63/100
91848/91848 - 70s - 759us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0058 - val
_mae: 0.0541
Epoch 64/100
91848/91848 - 70s - 762us/step - loss: 0.0058 - mae: 0.0541 - val_loss: 0.0057 - val
_mae: 0.0539
Epoch 65/100
91848/91848 - 69s - 750us/step - loss: 0.0058 - mae: 0.0541 - val_loss: 0.0057 - val
_mae: 0.0543
Epoch 66/100
91848/91848 - 69s - 752us/step - loss: 0.0058 - mae: 0.0541 - val_loss: 0.0056 - val
_mae: 0.0537
Epoch 67/100
91848/91848 - 70s - 759us/step - loss: 0.0059 - mae: 0.0541 - val_loss: 0.0056 - val
_mae: 0.0537
Epoch 68/100
91848/91848 - 69s - 750us/step - loss: 0.0058 - mae: 0.0541 - val_loss: 0.0057 - val
_mae: 0.0538
Epoch 69/100
91848/91848 - 69s - 746us/step - loss: 0.0059 - mae: 0.0542 - val_loss: 0.0056 - val
_mae: 0.0534
Epoch 70/100
91848/91848 - 70s - 761us/step - loss: 0.0059 - mae: 0.0542 - val_loss: 0.0057 - val
_mae: 0.0542
Epoch 71/100
91848/91848 - 69s - 753us/step - loss: 0.0059 - mae: 0.0544 - val_loss: 0.0058 - val
_mae: 0.0544
Epoch 72/100
91848/91848 - 70s - 757us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0059 - val
_mae: 0.0556
Epoch 73/100
91848/91848 - 69s - 757us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0058 - val
_mae: 0.0541
Epoch 74/100
91848/91848 - 70s - 759us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0058 - val
_mae: 0.0542
Epoch 75/100
91848/91848 - 70s - 757us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0058 - val
```

```
_mae: 0.0543
Epoch 76/100
91848/91848 - 69s - 754us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0058 - val
_mae: 0.0547
Epoch 77/100
91848/91848 - 72s - 782us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0060 - val
_mae: 0.0547
Epoch 78/100
91848/91848 - 69s - 755us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0058 - val
_mae: 0.0545
Epoch 79/100
91848/91848 - 70s - 757us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0058 - val
_mae: 0.0542
Epoch 80/100
91848/91848 - 69s - 756us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0058 - val
_mae: 0.0549
Epoch 81/100
91848/91848 - 71s - 778us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0059 - val
_mae: 0.0553
Epoch 82/100
91848/91848 - 70s - 757us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0059 - val
_mae: 0.0548
Epoch 83/100
91848/91848 - 69s - 755us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0059 - val
_mae: 0.0548
Epoch 84/100
91848/91848 - 70s - 758us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0059 - val
_mae: 0.0553
Epoch 85/100
91848/91848 - 69s - 750us/step - loss: 0.0060 - mae: 0.0546 - val_loss: 0.0059 - val
_mae: 0.0558
Epoch 86/100
91848/91848 - 69s - 747us/step - loss: 0.0060 - mae: 0.0546 - val_loss: 0.0057 - val
_mae: 0.0540
Epoch 87/100
91848/91848 - 69s - 749us/step - loss: 0.0060 - mae: 0.0546 - val_loss: 0.0059 - val
_mae: 0.0552
Epoch 88/100
91848/91848 - 69s - 750us/step - loss: 0.0060 - mae: 0.0546 - val_loss: 0.0057 - val
_mae: 0.0544
Epoch 89/100
91848/91848 - 70s - 757us/step - loss: 0.0059 - mae: 0.0545 - val_loss: 0.0057 - val
_mae: 0.0539
Epoch 90/100
91848/91848 - 69s - 750us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0059 - val
_mae: 0.0548
Epoch 91/100
91848/91848 - 69s - 756us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0057 - val
_mae: 0.0545
Epoch 92/100
91848/91848 - 69s - 747us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0057 - val
_mae: 0.0537
Epoch 93/100
91848/91848 - 69s - 751us/step - loss: 0.0061 - mae: 0.0548 - val_loss: 0.0059 - val
_mae: 0.0547
Epoch 94/100
```

```
91848/91848 - 69s - 750us/step - loss: 0.0060 - mae: 0.0547 - val_loss: 0.0059 - val
_mae: 0.0556
Epoch 95/100
91848/91848 - 69s - 756us/step - loss: 0.0060 - mae: 0.0548 - val_loss: 0.0059 - val
_mae: 0.0560
Epoch 96/100
91848/91848 - 69s - 751us/step - loss: 0.0060 - mae: 0.0546 - val_loss: 0.0058 - val
_mae: 0.0544
Epoch 97/100
91848/91848 - 69s - 749us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0057 - val
_mae: 0.0541
Epoch 98/100
91848/91848 - 69s - 750us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0058 - val
_mae: 0.0548
Epoch 99/100
91848/91848 - 69s - 749us/step - loss: 0.0060 - mae: 0.0546 - val_loss: 0.0059 - val
_mae: 0.0558
Epoch 100/100
91848/91848 - 69s - 753us/step - loss: 0.0059 - mae: 0.0546 - val_loss: 0.0058 - val
_mae: 0.0546
```

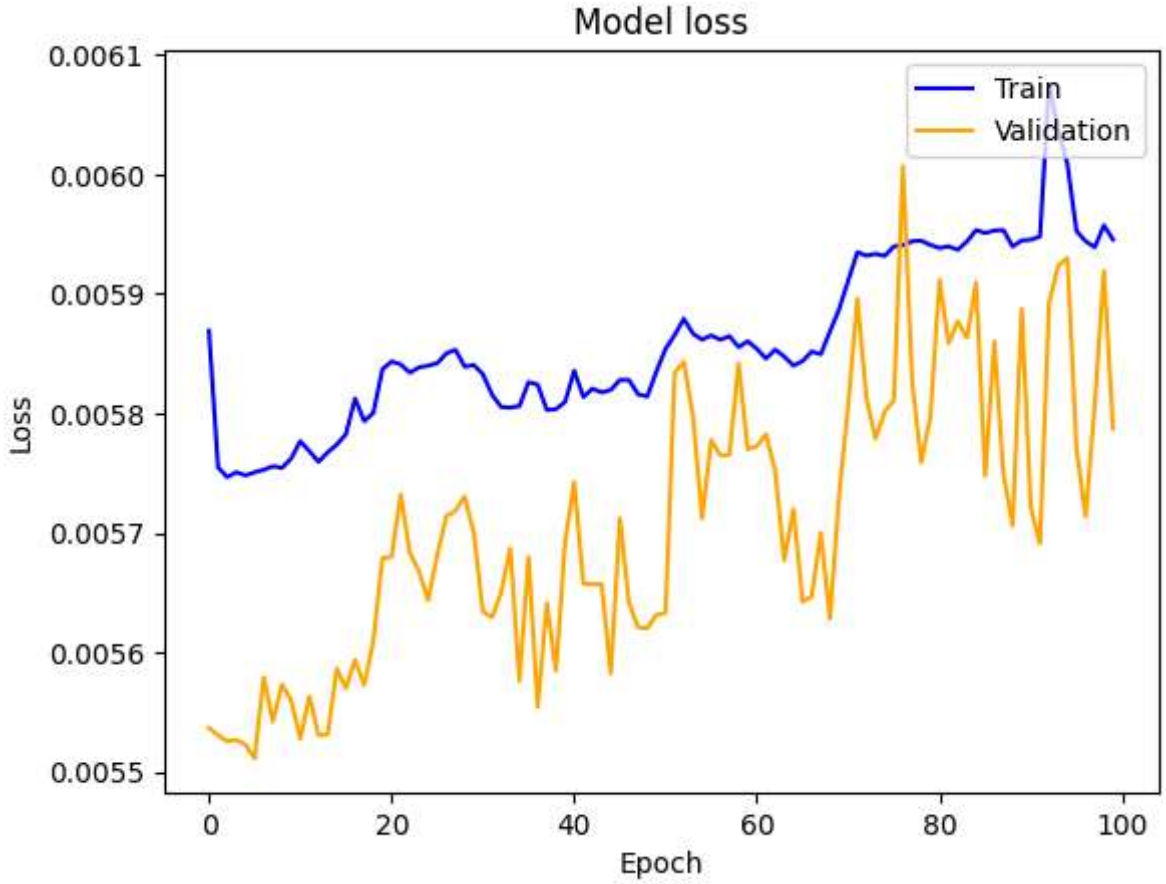
In [ ]:

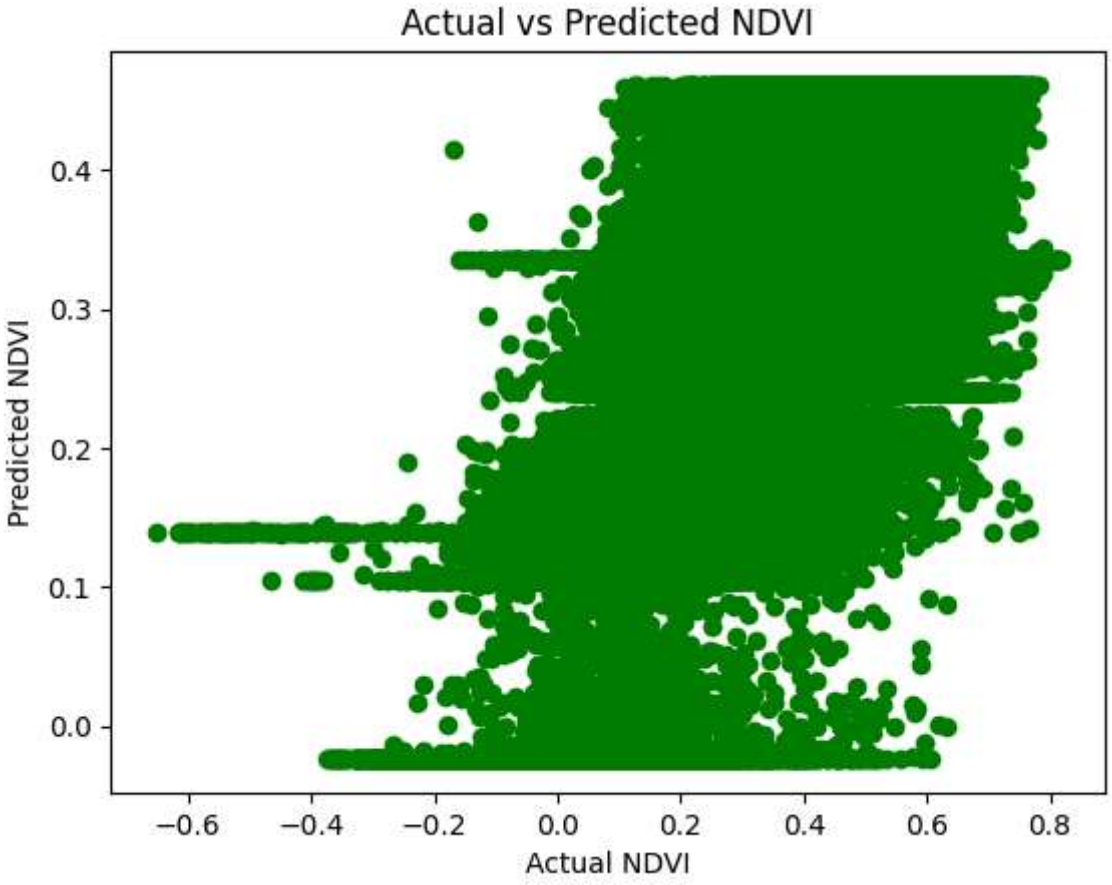
```python
# Evaluate the model on the test data
test_loss, test_mae = model.evaluate(X_test_scaled, y_test)

# Predict on test data
y_pred = model.predict(X_test_scaled)

# Calculate RMSE
rmse = np.sqrt(mean_squared_error(y_test, y_pred))

print(f"Test MAE: {test_mae}")
print(f"Test RMSE: {rmse}")
```

```
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 24s 422us/step - loss: 0.0058 - mae: 0.0545
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 24s 431us/step
Test MAE: 0.054502684623003006
Test RMSE: 0.07600250618852315
```

In [ ]:

```python
import matplotlib.pyplot as plt

# Plot training & validation loss values
plt.plot(history.history['loss'], color='blue')
plt.plot(history.history['val_loss'], color='orange')
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper right')
plt.show()

# Plot predictions vs actual with different colors
plt.scatter(y_test, y_pred, c='green')  # Use green color for the scatter plot
plt.xlabel('Actual NDVI')
plt.ylabel('Predicted NDVI')
plt.title('Actual vs Predicted NDVI')
plt.show()
```

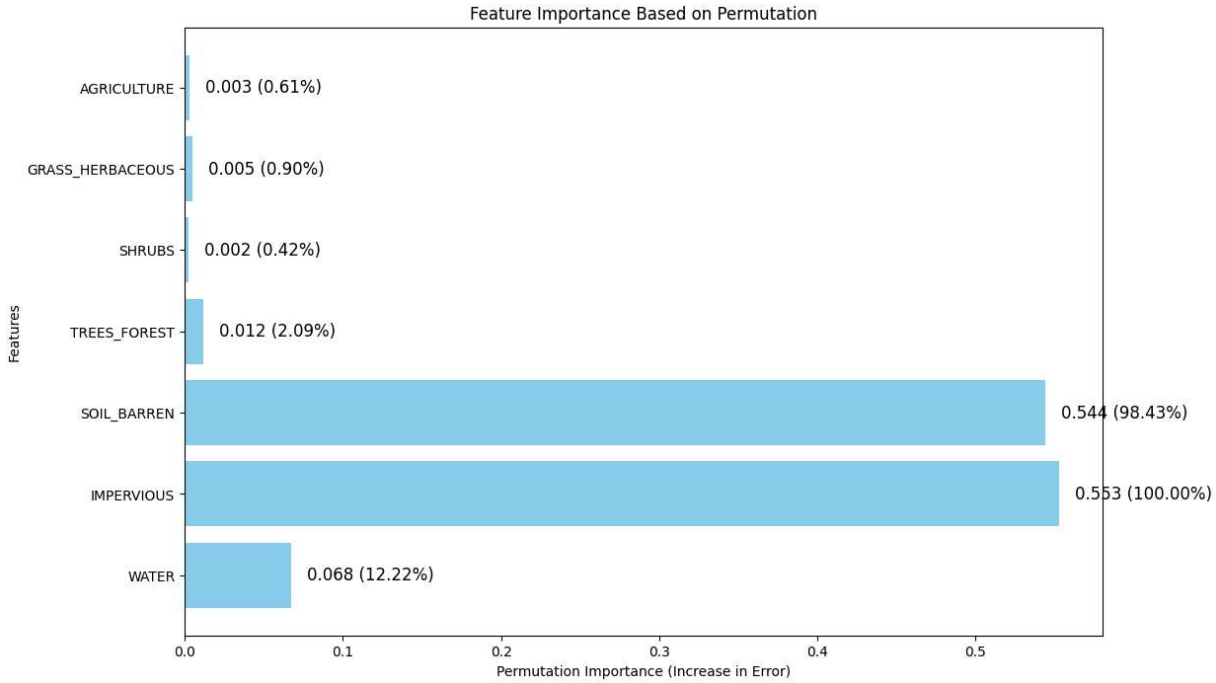

Model loss
Train
Validation
Loss
Epoch
0.0061
0.0060
0.0059
0.0058
0.0057
0.0056
0.0055
0
20
40
60
80
100


Actual vs Predicted NDVI
Predicted NDVI
Actual NDVI
0.4
0.3
0.2
0.1
0.0
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8

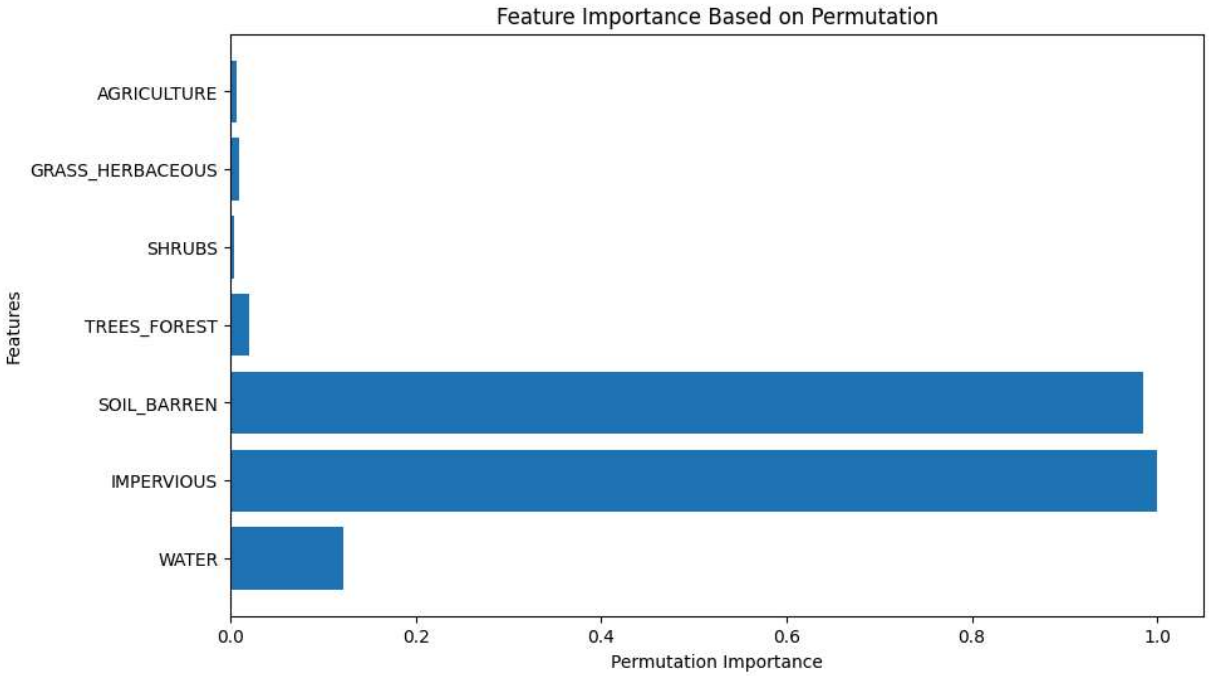
In [ ]:
```
from sklearn.inspection import permutation_importance
import matplotlib.pyplot as plt
import numpy as np

# Compute permutation importance
result = permutation_importance(model, X_test_scaled, y_test, n_repeats=10, random_

# Get the importance of each feature
importance = result.importances_mean

# Normalize importance by the maximum value to show relative importance
relative_importance = importance / np.max(importance)

# Plot the feature importance
plt.figure(figsize=(12, 8))
bars = plt.barh(X.columns, importance, color='skyblue')

# Add exact values on the bars
for bar, val, rel_val in zip(bars, importance, relative_importance):
    plt.text(bar.get_width() + 0.01, bar.get_y() + bar.get_height()/2,
             f'{val:.3f} ({rel_val:.2%})',
             va='center', ha='left', color='black', fontsize=12)

plt.xlabel('Permutation Importance (Increase in Error)')
plt.ylabel('Features')
plt.title('Feature Importance Based on Permutation')
plt.show()
```

```
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 24s 423us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 24s 430us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 402us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 24s 433us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 405us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 385us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 388us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 395us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 398us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 391us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 394us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 396us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 393us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 390us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 396us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 388us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 388us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 388us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 391us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 394us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 390us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 389us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 393us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 390us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 391us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 392us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 396us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 394us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 395us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 388us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 386us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 393us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 404us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 402us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 407us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 400us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 403us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 404us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 405us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 404us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 406us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 403us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 406us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 403us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 399us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 412us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 408us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 403us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 409us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 404us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 406us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 404us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 402us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 394us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 393us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 398us/step
```

```
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 397us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 396us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 398us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 396us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 389us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 395us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 399us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 397us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 398us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 395us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 396us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 393us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 22s 389us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 402us/step
56234/56234 ━━━━━━━━━━━━━━━━━━━━ 23s 403us/step
```



In [ ]:
```python
# Get the importance of each feature
importance = result.importances_mean
relative_importance = importance / np.max(importance)
# Plot the feature importance
plt.figure(figsize=(10, 6))
plt.barh(X.columns, relative_importance)
plt.xlabel('Permutation Importance')
plt.ylabel('Features')
plt.title('Feature Importance Based on Permutation')
plt.show()
```



In [ ]:
```
# Define a function to compute the gradient of the output with respect to the input
def compute_saliency(model, X):
    with tf.GradientTape() as tape:
        tape.watch(X)
        predictions = model(X)
    gradient = tape.gradient(predictions, X)
    return tf.reduce_mean(tf.abs(gradient), axis=0).numpy()

# Compute saliency for the test set
X_test_tensor = tf.convert_to_tensor(X_test_scaled)
saliency = compute_saliency(model, X_test_tensor)

# Normalize the saliency scores
normalized_saliency = saliency / np.max(saliency)

# Display the saliency scores for each feature
feature_importance = pd.Series(normalized_saliency, index=X.columns)
feature_importance.sort_values(ascending=False, inplace=True)
print(feature_importance)
```

```
SOIL_BARREN         1.000000
AGRICULTURE         0.635308
IMPERVIOUS          0.573039
SHRUBS              0.383332
GRASS_HERBACEOUS    0.310391
TREES_FOREST        0.297086
WATER               0.125982
dtype: float64
```

In [ ]: