Machine Learning Notes

This document provides basic knowledge about machine learning, particularly applications to Global Change Ecology Research. I will recursively update this document in order to help myself summarize what I have learned from reading books and articles and from conducting research.

Feature importance

It is noted that not all features are equally important. Some of them are either redundant or unessential. Thus, we need to discard some features in order to

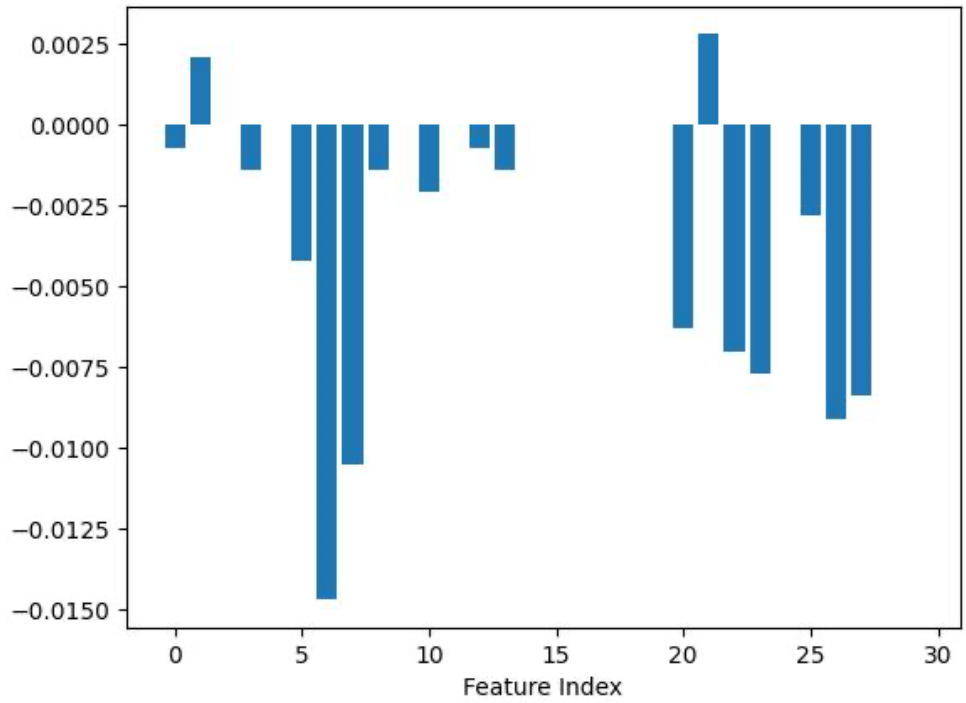
- improve the ability of a trained model
- reduce the over-fitting
- increase the speed of training and deduction /dɪˈdʌkʃ(ə)n/
- enhance the explainability

Methods of analyzing feature importance

Permutation importance

```
from sklearn.datasets import load_breast_cancer
from sklearn.ensemble import RandomForestClassifier
from sklearn.inspection import permutation_importance
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

 cancer = load_breast_cancer()

 X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer.target,
random_state=1)

 rf = RandomForestClassifier(n_estimators=100, random_state=1)
 rf.fit(X_train, y_train)

 baseline = rf.score(X_test, y_test)
 result = permutation_importance(rf, X_test, y_test, n_repeats=10,
random_state=1, scoring='accuracy')

 importances = result.importances_mean

 # Visualize permutation importances
```

```
plt.bar(range(len(importances)), importances)
plt.xlabel('Feature Index')
plt.ylabel('Permutation Importance')
plt.show()
```



Coef_feature_importances

```
def coef_feature_importances():
    from sklearn.datasets import load_breast_cancer
    from sklearn.ensemble import RandomForestClassifier

    X, y = load_breast_cancer(return_X_y=True)

    rf = RandomForestClassifier(n_estimators=100, random_state=1)
    rf.fit(X, y)

    importances = rf.feature_importances_

    # Plot importances
    plt.bar(range(X.shape[1]), importances)
    plt.xlabel('Feature Index')
    plt.ylabel('Feature Importance')
    plt.savefig('coef_feature_importances.jpg')
```

```
    plt.show()
```

Leave-one-out

```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import numpy as np

# Load sample data
X, y = load_breast_cancer(return_X_y=True)

# Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=1)

# Train a random forest model
rf = RandomForestClassifier(n_estimators=100, random_state=1)
rf.fit(X_train, y_train)

# Get baseline accuracy on test data
base_acc = accuracy_score(y_test, rf.predict(X_test))

# Initialize empty list to store importances
importances = []

# Iterate over all columns and remove one at a time
for i in range(X_train.shape[1]):
    X_temp = np.delete(X_train, i, axis=1)
    rf.fit(X_temp, y_train)
    acc = accuracy_score(y_test, rf.predict(np.delete(X_test, i, axis=1)))
    importances.append(base_acc - acc)

# Plot importance scores
plt.bar(range(len(importances)), importances)
plt.savefig('leave_one_out.jpg')
plt.show()
print("yes")
```

Correlation analysis

```
def correlation_analysis():
    import pandas as pd
    from sklearn.datasets import load_breast_cancer

    X, y = load_breast_cancer(return_X_y=True)
    df = pd.DataFrame(X, columns=range(30))
    df['y'] = y

    correlations = df.corrwith(df.y).abs()
    correlations.sort_values(ascending=False, inplace=True)

    correlations.plot.bar()
```

Recursively eliminate features

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import RFE
import pandas as pd
from sklearn.datasets import load_breast_cancer
import matplotlib.pyplot as plt

X, y = load_breast_cancer(return_X_y=True)
df = pd.DataFrame(X, columns=range(30))
df['y'] = y

rf = RandomForestClassifier()

rfe = RFE(rf, n_features_to_select=10)
rfe.fit(X, y)

print(rfe.ranking_)
```

XGBoost

```
import xgboost as xgb
import pandas as pd
from sklearn.datasets import load_breast_cancer
import matplotlib.pyplot as plt

X, y = load_breast_cancer(return_X_y=True)
df = pd.DataFrame(X, columns=range(30))
df['y'] = y
```

```
model = xgb.XGBClassifier()
model.fit(X, y)

importances = model.feature_importances_
importances = pd.Series(importances, index=range(X.shape[1]))
importances.plot.bar()
```

PCA

```
from sklearn.decomposition import PCA
import pandas as pd
from sklearn.datasets import load_breast_cancer
import matplotlib.pyplot as plt

X, y = load_breast_cancer(return_X_y=True)
df = pd.DataFrame(X, columns=range(30))
df['y'] = y

pca = PCA()
pca.fit(X)

plt.bar(range(pca.n_components_), pca.explained_variance_ratio_)
plt.xlabel('PCA components')
plt.ylabel('Explained Variance')
```

Variance analysis

```
## variance analysis
## the higher a f, the strong importance a feature has ##
from sklearn.feature_selection import f_classif
import pandas as pd
from sklearn.datasets import load_breast_cancer
import matplotlib.pyplot as plt

X, y = load_breast_cancer(return_X_y=True)
df = pd.DataFrame(X, columns=range(30))
df['y'] = y

fval = f_classif(X, y)
fval = pd.Series(fval[0], index=range(X.shape[1]))
fval.plot.bar()
```

Chi2

```
from sklearn.feature_selection import chi2
import pandas as pd
from sklearn.datasets import load_breast_cancer
import matplotlib.pyplot as plt

X, y = load_breast_cancer(return_X_y=True)
df = pd.DataFrame(X, columns=range(30))
df['y'] = y

chi_scores = chi2(X, y)
chi_scores = pd.Series(chi_scores[0], index=range(X.shape[1]))
chi_scores.plot.bar()
```

Warming Tips

1. use multiple methods
2. focus on ensemble methods of aggregated results
3. pay more attention to relative order instead of absolute values
4. differences do not mean problems, indicating we should check the reasons behind