project1

February 28, 2025

1 Machine Learning in Python - Project 1

Due Friday, Feb 28th by 4 pm.

Include contributors names in notebook metadata or here

1.1 Setup

To begin the analysis, we first import the necessary data processing, visualization, and modeling libraries. These will be used throughout the project for data exploration, preprocessing, and model training.

[235]:
```
# Add any additional libraries or submodules below

# Data libraries
import pandas as pd
import numpy as np

# Plotting libraries
import matplotlib.pyplot as plt
import seaborn as sns

# Plotting defaults
plt.rcParams['figure.figsize'] = (5,3)
plt.rcParams['figure.dpi'] = 80

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer

from sklearn.metrics import mean_squared_error, r2_score
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import GridSearchCV, KFold
```

[236]:
```
url = "https://raw.githubusercontent.com/AwuoeZYC/Project_1/main/"
```

[237]:
```
# Load data in easyshare.csv
d = pd.read_csv(f"{url}adnidata.csv", index_col=0)
d.head()
```

[237]:

	RID	ADAS13.bl	ADAS13.m24	AGE	DX.bl	PTGENDER	PTEDUCAT	PTETHCAT
1	3	31.00	37.67	81.3	AD	Male	18	Not Hisp/Latino
2	5	14.67	11.00	73.7	CN	Male	16	Not Hisp/Latino
3	6	25.67	22.67	80.4	LMCI	Female	13	Not Hisp/Latino
4	7	40.33	47.00	75.4	AD	Male	10	Hisp/Latino
5	10	24.33	30.33	73.9	AD	Female	12	Not Hisp/Latino

	PTRACCAT	PTMARRY	APOE4	Ventricles	Hippocampus	WholeBrain
1	White	Married	1.0	84599.0	5319.0	1129834.0
2	White	Married	0.0	34062.0	7075.0	1116633.0
3	White	Married	0.0	39826.0	5348.0	927510.0
4	More than one	Married	1.0	25704.0	6729.0	875798.0
5	White	Married	1.0	26820.0	5485.0	1033542.0

	Entorhinal	Fusiform	MidTemp	ICV
1	1791.0	15506.0	18422.0	1.920691e+06
2	4433.0	24788.0	21614.0	1.640766e+06
3	2277.0	17963.0	17802.0	1.485834e+06
4	2050.0	12063.0	15374.0	1.353519e+06
5	2676.0	16761.0	19741.0	1.471184e+06

[238]:
```
# workshop
def model_fit(m, X, y, plot = False):
    """Returns the mean squared error, root mean squared error and R^2 value of␣
 ↪a fitted model based
    on provided X and y values.

    Args:
        m: sklearn model object
        X: model matrix to use for prediction
        y: outcome vector to use to calculating rmse and residuals
        plot: boolean value, should fit plots be shown
    """

    y_hat = m.predict(X)
    MSE = mean_squared_error(y, y_hat)
    RMSE = np.sqrt(mean_squared_error(y, y_hat))
    Rsqr = r2_score(y, y_hat)

    Metrics = (round(MSE, 4), round(RMSE, 4), round(Rsqr, 4))

    res = pd.DataFrame(
        data = {'y': y, 'y_hat': y_hat, 'resid': y - y_hat}
```

```
    )

    if plot:
        plt.figure()

        plt.subplot(121)
        sns.lineplot(x='y', y='y_hat', color="grey", data =  pd.
↪DataFrame(data={'y': [min(y),max(y)], 'y_hat': [min(y),max(y)]}))
        sns.scatterplot(x='y', y='y_hat', data=res).set_title("Observed vs␣
↪Fitted values")

        plt.subplot(122)
        sns.scatterplot(x='y_hat', y='resid', data=res).set_title("Fitted␣
↪values vs Residuals")
        plt.hlines(y=0, xmin=np.min(y), xmax=np.max(y), linestyles='dashed',␣
↪alpha=0.3, colors="black")

        plt.subplots_adjust(left=0.0)

        plt.suptitle("Model (MSE, RMSE, Rsq) = " + str(Metrics), fontsize=14)
        plt.show()

    return MSE, RMSE, Rsqr
```

2 Introduction

Alzheimer's disease (AD) is a progressive neurodegenerative disorder that leads to cognitive decline, making early prediction crucial for identifying at-risk individuals and enabling timely interventions. This study aims to develop a machine learning model to predict ADAS-Cog 13 scores at a 24-month follow-up using baseline features from the Alzheimer's Disease Neuroimaging Initiative (ADNI) dataset.

The dataset includes demographic, genetic (APOE4 genotype), cognitive assessment, and MRI Magnetic Resonance Imaging -derived brain volumetric features. To achieve this, we preprocess the data by handling missing values, normalizing MRI volumes relative to intracranial volume (ICV), and encoding categorical variables. We then perform exploratory data analysis to examine feature distributions and correlations before applying various regression models, including linear regression, ridge regression, lasso regression, and elastic net, with hyperparameter tuning via cross-validation.

(NEED TO CHECK) Our findings indicate that baseline ADAS13 score is the most significant predictor of cognitive decline, while APOE4 genotype, age, and MRI-derived features such as hippocampal and ventricular volumes also play critical roles. Among the models tested, ridge regression and elastic net achieved the best predictive performance, balancing accuracy and interpretability.

This predictive framework has potential applications in early diagnosis and targeted intervention strategies, contributing to improved clinical decision-making and patient outcomes. However, further research is needed to enhance model generalizability using larger and more diverse datasets.

3 Exploratory Data Analysis and Feature Engineering

Before proceeding with feature selection, we check for missing values and duplicate entries, as they can impact model performance.

[239]:
```
# Checking for missing values
missing_values = d.isnull().sum()
print("Missing values per column:\n", missing_values)

# Check duplicate rows
duplicates = d.duplicated()

# Count the number of duplicate rows
num_duplicates = duplicates.sum()

print(f"Number of duplicate rows: {num_duplicates}")
```

```
Missing values per column:
 RID              0
ADAS13.bl         0
ADAS13.m24        0
AGE               0
DX.bl             0
PTGENDER          0
PTEDUCAT          0
PTETHCAT          0
PTRACCAT          0
PTMARRY           0
APOE4             5
Ventricles      147
Hippocampus     147
WholeBrain      148
Entorhinal      147
Fusiform        147
MidTemp         147
ICV               8
dtype: int64
Number of duplicate rows: 0
```

The dataset contains several columns with missing values, including **Ventricles, Hippocampus, WholeBrain**, and other clinical measurements. Additionally, **no duplicate rows were found in the dataset**.

Then, we analyze categorical variables to understand their distributions, which is essential for encoding and feature selection.

[240]:
```
category_columns = ['DX.bl', 'PTGENDER', 'PTETHCAT', 'PTRACCAT', 'PTMARRY',␣
↪'APOE4']
```

```
stats_table = pd.DataFrame(columns=['Variable', 'Category', 'Count',␣
 ↪'Percentage'])

# Iterate over the column and calculate the value count for each categorical␣
 ↪variable
for col in category_columns:
    value_counts = d[col].value_counts().rename_axis('Category').
 ↪reset_index(name='Count')
    value_counts['Variable'] = col  # Add a variable name column
    value_counts['Percentage'] = value_counts['Count'] / len(d) * 100  #␣
 ↪Calculated percentage
    stats_table = pd.concat([stats_table, value_counts[['Variable', 'Category',␣
 ↪'Count', 'Percentage']]], ignore_index=True)

# Adjust column order
stats_table = stats_table[['Variable', 'Category', 'Count', 'Percentage']]

# Output form
print("Statistical table of categorical variables: ")
print(stats_table)
```

```
Statistical table of categorical variables:
    Variable           Category Count  Percentage
0      DX.bl               LMCI   367   35.356455
1      DX.bl                 CN   334   32.177264
2      DX.bl               EMCI   190   18.304432
3      DX.bl                 AD   147   14.161850
4   PTGENDER               Male   582   56.069364
5   PTGENDER             Female   456   43.930636
6   PTETHCAT    Not Hisp/Latino  1011   97.398844
7   PTETHCAT        Hisp/Latino    21    2.023121
8   PTETHCAT            Unknown     6    0.578035
9   PTRACCAT              White   973   93.737958
10  PTRACCAT              Black    35    3.371869
11  PTRACCAT              Asian    16    1.541426
12  PTRACCAT      More than one    10    0.963391
13  PTRACCAT  Am Indian/Alaskan     2    0.192678
14  PTRACCAT            Unknown     1    0.096339
15  PTRACCAT  Hawaiian/Other PI     1    0.096339
16   PTMARRY            Married   798   76.878613
17   PTMARRY            Widowed   120   11.560694
18   PTMARRY           Divorced    84    8.092486
19   PTMARRY      Never married    31    2.986513
20   PTMARRY            Unknown     5    0.481696
21     APOE4                0.0   569   54.816956
22     APOE4                1.0   370   35.645472
23     APOE4                2.0    94    9.055877
```

```
C:\Users\rjx10\AppData\Local\Temp\ipykernel_15948\1622175386.py:10:
FutureWarning: The behavior of DataFrame concatenation with empty or all-NA
entries is deprecated. In a future version, this will no longer exclude empty or
all-NA columns when determining the result dtypes. To retain the old behavior,
exclude the relevant entries before the concat operation.
  stats_table = pd.concat([stats_table, value_counts[['Variable', 'Category',
'Count', 'Percentage']]], ignore_index=True)
```

The dataset shows that most participants are diagnosed with **LMCI (35.36%)**, followed by **CN (32.18%)**, with a smaller percentage in **AD (14.16%)**. The majority of participants are **male (56.07%)** and **white (93.74%)**. Most participants are **married (76.88%)**, and a significant portion of the participants have no **APOE4 genetic risk (54.82%)**, with others carrying the **1.0 (35.65%)** and **2.0 (9.06%)** versions.

To evaluate model performance, we split the dataset into training (80%) and test (20%) sets.

[241]:
```
random_seed = 2
```

[ ]:
```
# Define the target and features (we will refine feature selection later)
target = 'ADAS13.m24'
features = d.columns.drop(['RID', 'ADAS13.m24'])  # Exclude person identifier␣
 ↪and target(    )

# Drop rows with missing values (or consider imputation strategies)
d_clean = d.dropna().copy().drop(columns=['RID'])

# Create normalized volume features
volume_cols = ['Ventricles', 'Hippocampus', 'WholeBrain', 'Entorhinal',␣
 ↪'Fusiform', 'MidTemp']
for col in volume_cols:
    new_col = col + '_norm'
    d_clean.loc[:, new_col] = d_clean[col] / d_clean['ICV']

# Create training and testing sets
train_df, test_df = train_test_split(d_clean, test_size=0.2,␣
 ↪random_state=random_seed)
print("Training set size:", train_df.shape)
print("Testing set size:", test_df.shape)
```

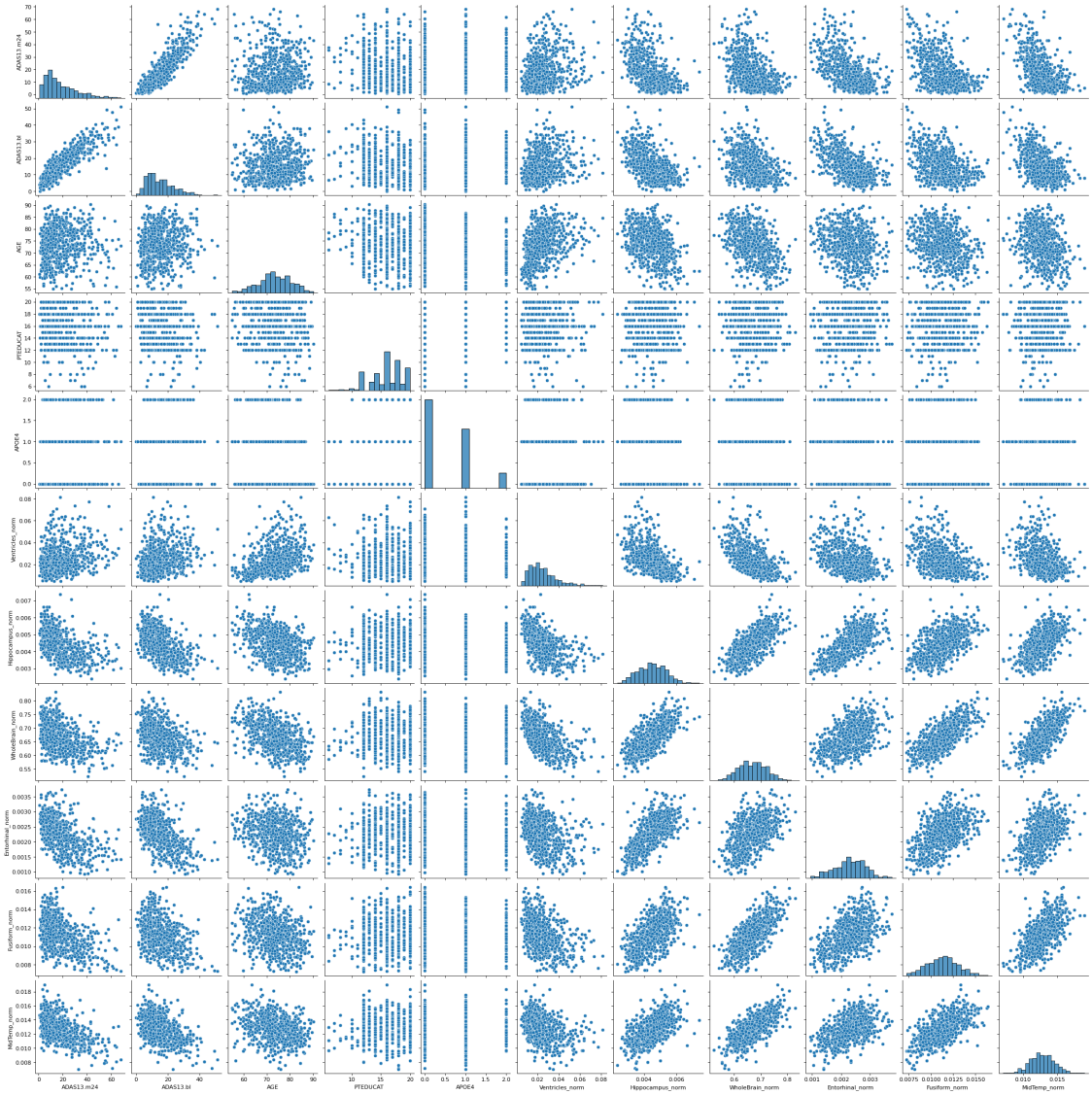
```
   Ventricles_norm  Hippocampus_norm  WholeBrain_norm  Entorhinal_norm  \
1         0.044046          0.002769         0.588244         0.000932
2         0.020760          0.004312         0.680556         0.002702
3         0.026804          0.003599         0.624235         0.001532
4         0.018990          0.004971         0.647052         0.001515
5         0.018230          0.003728         0.702524         0.001819

   Fusiform_norm  MidTemp_norm
1       0.008073      0.009591
2       0.015108      0.013173
```

```
3       0.012090      0.011981
4       0.008912      0.011359
5       0.011393      0.013418
Training set size: (708, 23)
Testing set size: (177, 23)
```

The training set contains 80% of the data, providing enough samples for model learning, while the test set consists of 20%, allowing evaluation on unseen data. Additionally, rows with missing values were removed, resulting in a clean dataset.

To visualize relationships between baseline cognitive scores, demographic variables, and normalized volumetric features, we then generate a pairplot.

[244]:
```
sns.pairplot(train_df[['ADAS13.m24', 'ADAS13.bl', 'AGE', 'PTEDUCAT', 'APOE4',␣
 ↪'Ventricles_norm', 'Hippocampus_norm',
                       'WholeBrain_norm', 'Entorhinal_norm', 'Fusiform_norm',␣
 ↪'MidTemp_norm']])
plt.show()
```



ADAS13.bl and ADAS13.m24 show a strong linear correlation, reinforcing the idea that baseline cognitive assessment is a crucial predictor of future cognitive decline. Hippocampus_norm is negatively correlated with ADAS13.m24, suggesting that smaller hippocampal volumes are associated with greater cognitive deterioration. On the other hand, Ventricles_norm is positively correlated with ADAS13.m24, indicating that larger ventricular volumes are linked to increased brain atrophy and cognitive impairment. Meanwhile, Age and education (PTEDUCAT) do not exhibit strong correlations.

We compute a correlation matrix for numerical features to identify key predictors of cognitive decline.

[265]:
```
numeric_df = train_df.select_dtypes(include=[np.number])
corr_matrix = numeric_df.corr()
subset_corr = corr_matrix.loc['ADAS13.m24']
```

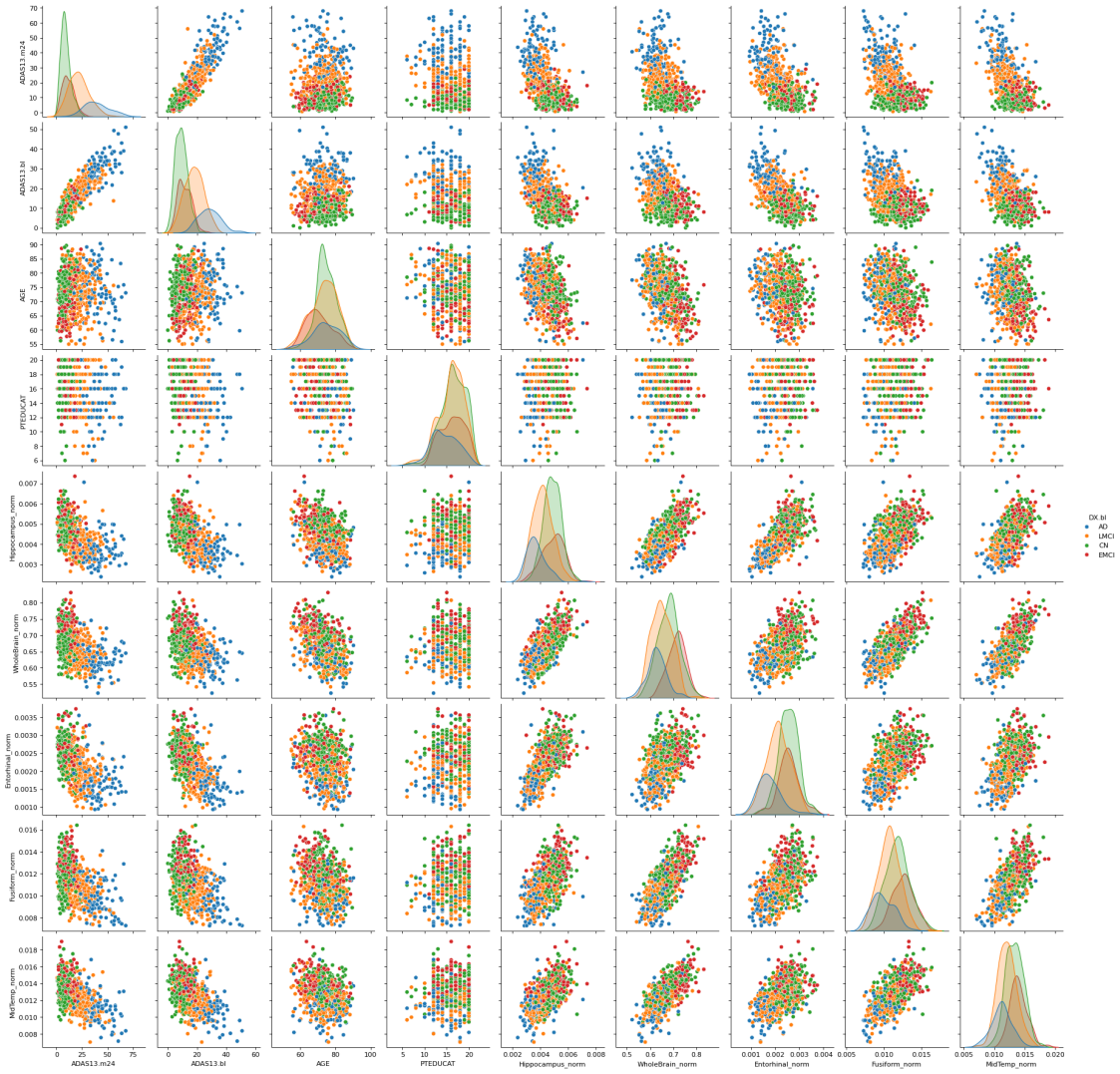
```
print(subset_corr)
```

```
ADAS13.bl           0.882006
ADAS13.m24          1.000000
AGE                 0.063524
PTEDUCAT           -0.207972
APOE4               0.352586
Ventricles          0.270433
Hippocampus        -0.527248
WholeBrain         -0.258306
Entorhinal         -0.527313
Fusiform           -0.424442
MidTemp            -0.458689
ICV                 0.067017
Ventricles_norm     0.290048
Hippocampus_norm   -0.537560
WholeBrain_norm    -0.453800
Entorhinal_norm    -0.546310
Fusiform_norm      -0.486201
MidTemp_norm       -0.553484
Name: ADAS13.m24, dtype: float64
```

ADAS13.bl has the highest positive correlation (r = 0.882) with ADAS13.m24, establishing it as a key predictor of cognitive decline. Hippocampus_norm shows a strong negative correlation (r = -0.537) with ADAS13.m24, suggesting that reduced hippocampal volume is associated with greater cognitive deterioration. Ventricles_norm has a moderate positive correlation (r = 0.290) with ADAS13.m24, indicating that larger ventricles are linked to cognitive impairment. ICV exhibits a very weak correlation (r = 0.067) with ADAS13.m24, suggesting that intracranial volume has little influence on cognitive decline. Additionally, AGE shows a minimal correlation (r = 0.063), while APOE4 has a moderate positive correlation (r = 0.352) with cognitive decline. Other brain regions such as WholeBrain_norm, Entorhinal_norm, and MidTemp_norm also demonstrate significant negative correlations, further supporting their role in cognitive deterioration.

To compare cognitive decline across different diagnostic groups, we generate a boxplot of ADAS13.m24 for CN, EMCI, LMCI, and AD groups.

[246]:
```
sns.pairplot(train_df[['ADAS13.m24', 'ADAS13.bl', 'AGE', 'PTEDUCAT',␣
 ↪'Hippocampus_norm', 'WholeBrain_norm', 'Entorhinal_norm',␣
 ↪'Fusiform_norm','MidTemp_norm',
                       'DX.bl']], hue='DX.bl')
plt.show()
```

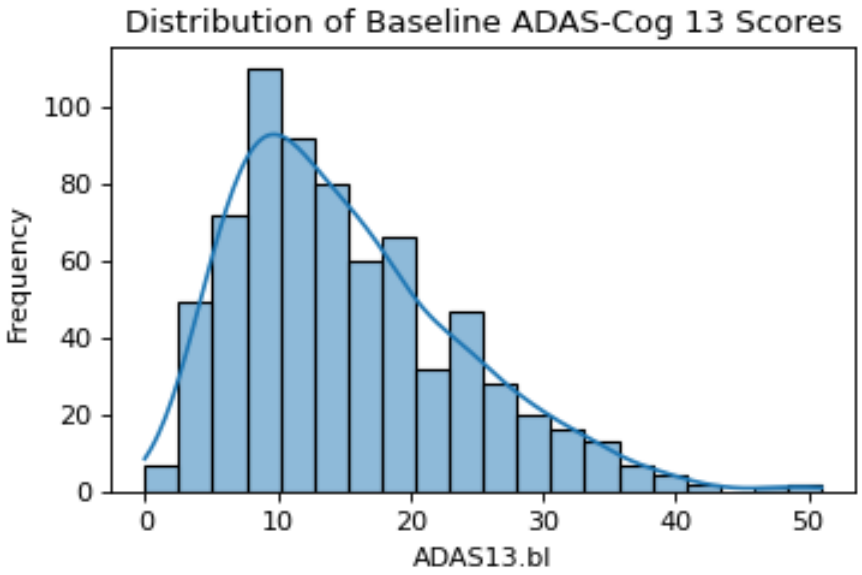


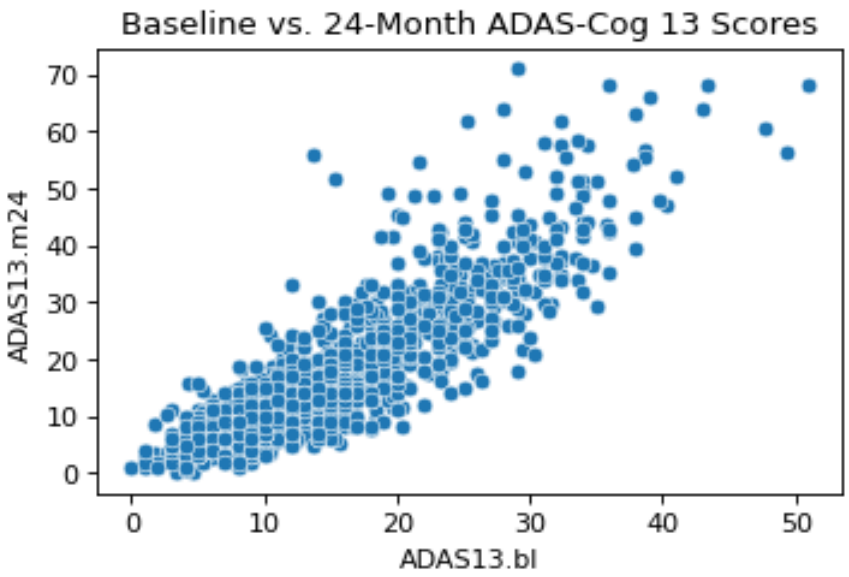
We generate a boxplot compare ADAS13.m24 scores across different diagnostic categories (CN, EMCI, LMCI, AD). The results indicate that patients diagnosed with Alzheimer's disease (AD) exhibit the highest cognitive decline, as reflected by their significantly higher ADAS13.m24 scores. In contrast, individuals classified as cognitively normal (CN) have the lowest scores, confirming that they experience minimal cognitive deterioration over 24 months. Participants with early mild cognitive impairment (EMCI) and late mild cognitive impairment (LMCI) show a wider distribution of scores, suggesting that cognitive decline in these groups varies significantly between individuals.

To analyze the distribution of baseline cognitive scores (ADAS13.bl) and their relationship with future cognitive decline (ADAS13.m24), we generate a histogram to assess the distribution of ADAS13.bl and a scatterplot to visualize the relationship between ADAS13.bl and ADAS13.m24 over 24 months.

[247]:
```
# Histogram for baseline ADAS-Cog 13 scores
plt.figure()
```

```
sns.histplot(train_df['ADAS13.bl'], bins=20, kde=True)
plt.title('Distribution of Baseline ADAS-Cog 13 Scores')
plt.xlabel('ADAS13.bl')
plt.ylabel('Frequency')
plt.show()

# Scatter plot of baseline vs. 24-month follow-up scores
plt.figure()
sns.scatterplot(x='ADAS13.bl', y='ADAS13.m24', data=d)
plt.title('Baseline vs. 24-Month ADAS-Cog 13 Scores')
plt.xlabel('ADAS13.bl')
plt.ylabel('ADAS13.m24')
plt.show()
```





DAS13.bl follows a right-skewed distribution, meaning most participants have mild impairment, with fewer experiencing severe decline. The scatterplot confirms a strong positive correlation between ADAS13.bl and ADAS13.m24, reinforcing that higher baseline cognitive impairment leads to greater decline over time. This validates the use of ADAS13.bl as a primary predictive feature in regression models.

4 Model Fitting and Tuning

4.1 Baseline Model

We select ADAS13.bl, AGE, PTEDUCAT, and MRI volumetric features (e.g., Ventricles_norm, Hippocampus_norm) as key predictors of ADAS13.m24, based on their clinical relevance to cognitive decline and Alzheimer's progression. These features ensure effective model training and evaluation.

[248]:
```
# Define a basic set of features for the baseline model
features_baseline = ['ADAS13.bl', 'AGE', 'PTEDUCAT'] + [col for col in d_clean.
 ↪columns if '_norm' in col]

print(features_baseline)
X_train_baseline = train_df[features_baseline]
y_train_baseline = train_df[target]
X_test_baseline = test_df[features_baseline]
y_test_baseline = test_df[target]
```

```
['ADAS13.bl', 'AGE', 'PTEDUCAT', 'Ventricles_norm', 'Hippocampus_norm',
'WholeBrain_norm', 'Entorhinal_norm', 'Fusiform_norm', 'MidTemp_norm']
```

Linear regression is selected as the benchmark model due to its interpretability and efficiency in

estimating the relationship between cognitive decline and predictor variables. The model follows the equation:

$$\text{ADAS13.m24} = \beta_0 + \beta_1(\text{ADAS13.bl}) + \beta_2(\text{AGE}) + \beta_3(\text{PTEDUCAT}) + \ldots + \beta_n(\text{MRIFeatures}) + \epsilon$$

where β represents the estimated coefficients, and ϵ is the residual error. This model serves as a reference point for evaluating more complex techniques, such as Ridge Regression, in subsequent steps. The primary goal is to determine how well a simple linear model can predict future cognitive scores based on baseline characteristics.

[249]:
```
# Fit the baseline model
baseline_model = LinearRegression()
baseline_model.fit(X_train_baseline, y_train_baseline)
```

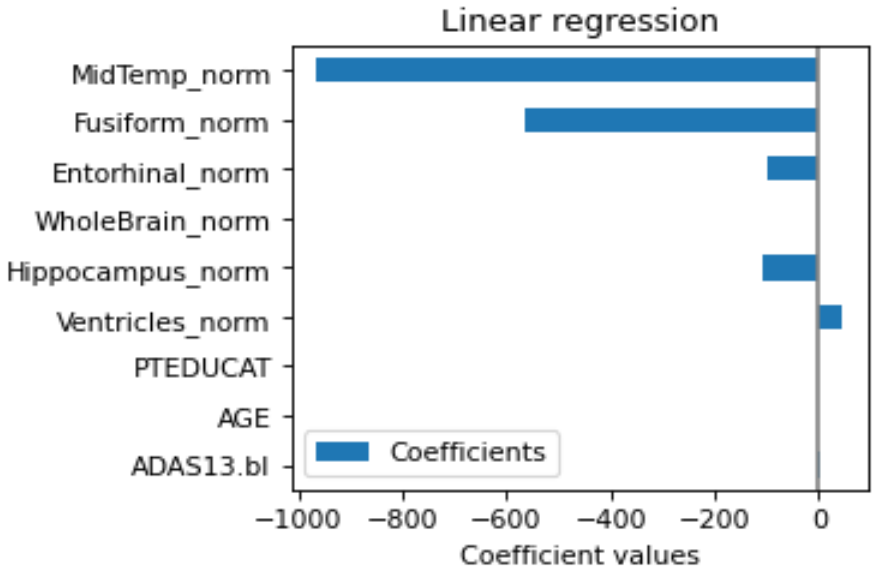
[249]:
```
LinearRegression()
```

To understand feature importance, we extract coefficient values from the baseline model.

[250]:
```
bcoefs = baseline_model.coef_
bcoefs_ = pd.DataFrame(
    np.copy(baseline_model.coef_),
    columns=["Coefficients"],
    index=features_baseline,
)

print(bcoefs_)

bcoefs_.plot.barh()
plt.title("Linear regression")
plt.axvline(x=0, color=".5")
plt.xlabel("Coefficient values")
plt.subplots_adjust(left=0.3)

train_MSE, train_RMSE, train_R2 = model_fit(baseline_model, X_train_baseline,␣
 ↪y_train_baseline, plot=False)
print(f"Training Data - MSE: {train_MSE}, RMSE: {train_RMSE}, R²: {train_R2}")

test_MSE, test_RMSE, test_R2 = model_fit(baseline_model, X_test_baseline,␣
 ↪y_test_baseline, plot=False)
print(f"Test Data - MSE: {test_MSE}, RMSE: {test_RMSE}, R²: {test_R2}")
```

```
                  Coefficients
ADAS13.bl             1.182755
AGE                  -0.208472
PTEDUCAT             -0.135640
Ventricles_norm      42.086665
Hippocampus_norm   -110.292635
WholeBrain_norm      -2.603836
Entorhinal_norm     -98.776700
Fusiform_norm      -563.941724
```

```
MidTemp_norm       -964.546785
Training Data - MSE: 32.57577372900131, RMSE: 5.70751905200511, R²:
0.8122828654570748
Test Data - MSE: 39.233348681268886, RMSE: 6.263652982187701, R²:
0.7469974212066837
```



The model analysis reveals that ADAS13.bl (with a coefficient of 1.18) is the strongest predictor of future cognitive scores. Negative coefficients for hippocampal and entorhinal volumes indicate a link between brain atrophy and cognitive decline, while the positive coefficient for Ventricles_norm (42.08) suggests that ventricular enlargement is associated with disease progression. In terms of model performance, the training data yields an MSE of 32.58, RMSE of 5.71, and R^2 of 0.81, while the test data shows an MSE of 39.23, RMSE of 6.26, and R^2 of 0.75. The decrease in R^2 from training to test data highlights overfitting, and the higher test MSE suggests that regularization techniques, such as Ridge Regression, should be implemented to improve generalization.

4.2 Ridge Regression

Ridge Regression is an extension of linear regression that addresses overfitting and multicollinearity by introducing an L2 regularization term to the cost function. This penalty term prevents the model from assigning excessively large weights to features, improving numerical stability and generalization performance. The objective function for Ridge Regression is:

$$\sum_{n=1}^{N}(y_n - \omega^T \mathbf{x}_n)^2 + \alpha \sum_{d=1}^{D} w_d^2$$

where:

(1) y_n represents the observed response variable,

(2) ω is the weight vector (regression coefficients),

(3) $\mathbf{x}_n$ represents the predictor variables,

(4) α is the regularization parameter.

When , Ridge Regression is equivalent to ordinary least squares (OLS), meaning no regularization is applied. As increases, the model applies stronger penalty constraints, shrinking the regression coefficients and reducing variance. The Ridge Regression estimator is given by:

$$\hat{}_{ridge} = (X^T X + \alpha I)^{-1} X^T y$$

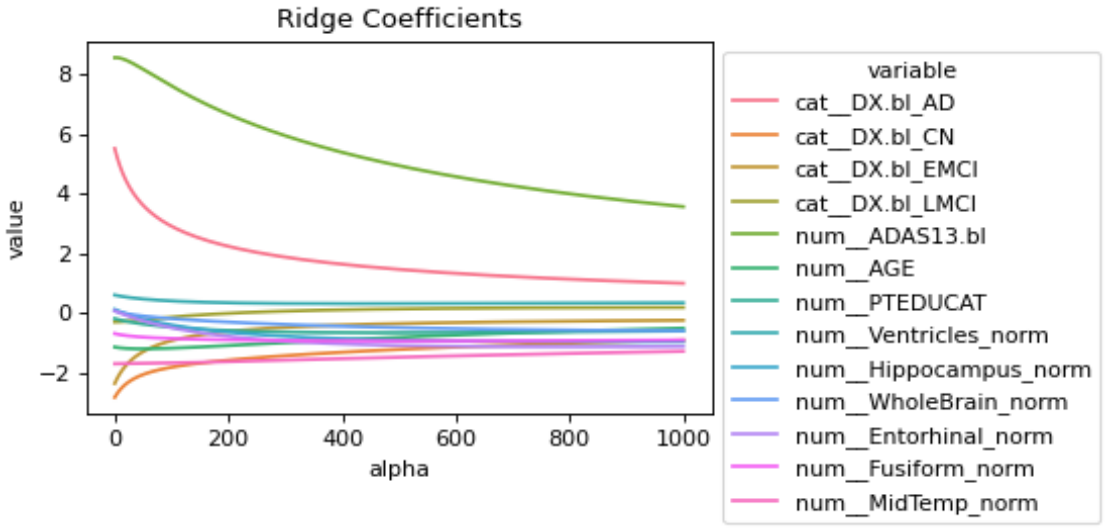
where I is the identity matrix. The addition of αI ensures that the model remains stable, particularly in datasets with highly correlated features.

To determine the optimal α value, we apply grid search with cross-validation, systematically testing multiple values of and selecting the one that minimizes the mean squared error (MSE). This ensures that Ridge Regression finds the best balance between bias and variance, resulting in improved generalization.

Before applying Ridge Regression, proper feature preprocessing is essential to ensure that all predictors contribute equally to the regularization term. Since Ridge introduces an L2 penalty to stabilize coefficient estimates and prevent overfitting, numerical features must be standardized, while categorical variables should be encoded using one-hot encoding to maintain comparability. To analyze the impact of regularization strength , we train Ridge models across 300 different values of , observing how coefficient magnitudes change as the penalty term increases. This process ensures feature comparability, evaluates coefficient shrinkage, and identifies the most important variables for prediction.

[267]:
```
#Ridge
from sklearn.linear_model import Ridge
from sklearn.preprocessing import StandardScaler

# DX.bl
features_ridge = ['DX.bl', 'ADAS13.bl', 'AGE', 'PTEDUCAT'] + [col for col in␣
 ↪d_clean.columns if '_norm' in col]

X_train_ridge = train_df[features_ridge]
y_train_ridge = train_df[target]
X_test_ridge = test_df[features_ridge]
y_test_ridge = test_df[target]
# Grid of alpha values
alphas = np.logspace(-3, 3, num=300) # from 10^-2 to 10^3

ws = [] # Store coefficients
mses_train = [] # Store training mses
mses_test = [] # Store test mses

#   ColumnTransformer
```

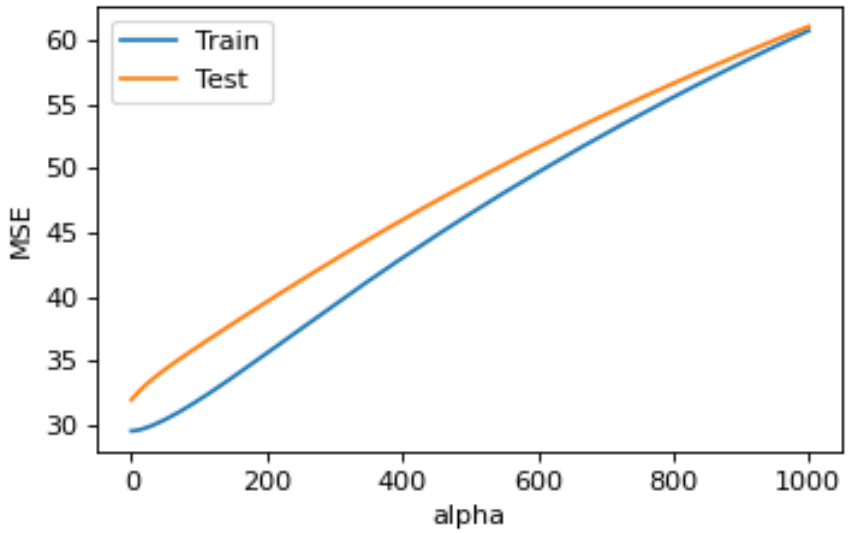
```
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(), [0]),  # The classification features are␣
 ↪encoded by OneHot
        ('num', StandardScaler(), [1,2,3,4,5,6,7,8,9])  # The numerical␣
 ↪features are normalized
    ],
    remainder='passthrough'  # Retain other unspecified features
)

for a in alphas:
    m = make_pipeline(
        preprocessor,  # Use ColumnTransformer for feature preprocessing
        Ridge(alpha=a)  # Using Ridge model
    ).fit(X_train_ridge, y_train_ridge)

    # Gets the weight of the model
    w_temp = np.copy(m.named_steps['ridge'].coef_)  # Gets the weight of the␣
 ↪Ridge model
    ws.append(w_temp)
    mses_train.append(mean_squared_error(y_train_ridge, m.
 ↪predict(X_train_ridge)))
    mses_test.append(mean_squared_error(y_test_ridge, m.predict(X_test_ridge)))

# Gets the processed column name
feature_names = preprocessor.get_feature_names_out()

# Create a data frame for plotting
sol_path = pd.DataFrame(
    data = ws,
    columns = feature_names # Label columns w/ feature names
).assign(
    alpha = alphas,
).melt(
    id_vars = ('alpha')
)

# Plot solution path of the weights
plt.figure()
ax = sns.lineplot(x='alpha', y='value', hue='variable', data=sol_path)
sns.move_legend(ax, "upper left", bbox_to_anchor=(1, 1))
ax.set_title("Ridge Coefficients")
plt.show()
```



The Ridge Coefficients plot highlights the most important variables for predicting ADAS24, identified by their large absolute coefficient values across different values. The key variables include cat_DX.bl_AD, cat_DX.bl_CN, cat_DX.bl_EMCI, num_ADAS13.bl, and num_AGE. Among these, cat_DX.bl_AD and cat_DX.bl_CN have relatively large positive coefficients, while cat_DX.bl_EMCI has a large negative coefficient. Additionally, num_ADAS13.bl and num_AGE exhibit notable variations, reinforcing their critical role in predicting cognitive decline.

To analyze the impact of α (regularization strength) in Ridge Regression, we train models across 300 different values and observe how coefficient magnitudes change.

[252]:
```
# Path of MSE as function of alpha
mses_path = pd.DataFrame(
    {'alpha': alphas, 'Train': np.asarray(mses_train), 'Test': np.
 ↪asarray(mses_test)}).melt(
    id_vars = ('alpha')
)

# Plot MSE path
plt.figure()
ax = sns.lineplot(x='alpha', y='value', hue='variable', data=mses_path)
ax.set_ylabel("MSE")
# To remove legend title
handles, labels = ax.get_legend_handles_labels()
ax.legend(handles=handles[0:], labels=labels[0:])
plt.show()
```



To determine the best regularization parameter, we find the value that minimizes test MSE.

[253]:
```
test_MSE = mses_path[mses_path['variable'] == 'Train']['value']

min_test_mse_index = np.argmin(test_MSE)

best_alpha = alphas[min_test_mse_index]

print(f"The optimal alpha (minimizing test MSE) is: {best_alpha}")
```

```
The optimal alpha (minimizing test MSE) is: 0.001
```

As alpha increases, the training MSE continues to increase. This is because larger alpha values result in simpler models that do not fit the training data well. The test MSE is lower when alpha is low, and with the increase of alpha, the test MSE first decreases and then increases. This is because small alpha values can lead to overfitting, while large alpha values can lead to underfitting. An alpha value of 0.001 is probably better.

To systematically find the best value, we use GridSearchCV with 5-fold cross-validation.

[254]:
```
# Grid of tuning parameters
alphas = np.logspace(-3, 2, num=300)

#Pipeline
m = make_pipeline(
        preprocessor,  # Use ColumnTransformer for feature preprocessing
        Ridge()  # Using Ridge model
)

# CV strategy
```

```
cv = KFold(5, shuffle=True, random_state=random_seed)

# Grid search
gs_ridge = GridSearchCV(m,
    param_grid={'ridge__alpha': alphas},
    cv=cv,
    scoring="neg_mean_squared_error")
gs_ridge.fit(X_train_ridge, y_train_ridge)
```

[254]:
```
GridSearchCV(cv=KFold(n_splits=5, random_state=2, shuffle=True),
             estimator=Pipeline(steps=[('columntransformer',
ColumnTransformer(remainder='passthrough',
                                                  transformers=[('cat',
OneHotEncoder(),
                                                                 [0]),
                                                                ('num',
StandardScaler(),
                                                                 [1, 2,
                                                                  3, 4,
                                                                  5, 6,
                                                                  7, 8,
9])])),
                                       ('ridge', Ridge())]),
             param_grid={'ridge__alpha': array([1.00000000e-03, 1.03925568e-03,
1.08005237e-03, 1.12245057…
       4.12462638e+01, 4.28654141e+01, 4.45481252e+01, 4.62968923e+01,
       4.81143084e+01, 5.00030684e+01, 5.19659730e+01, 5.40059328e+01,
       5.61259726e+01, 5.83292359e+01, 6.06189899e+01, 6.29986298e+01,
       6.54716840e+01, 6.80418197e+01, 7.07128478e+01, 7.34887289e+01,
       7.63735792e+01, 7.93716762e+01, 8.24874655e+01, 8.57255673e+01,
       8.90907830e+01, 9.25881025e+01, 9.62227117e+01, 1.00000000e+02])},
             scoring='neg_mean_squared_error')
```

Cross-validation ensures robust model evaluation across different subsets of data. The optimal value is chosen based on the lowest validation error, preventing overfitting. This process refines Ridge Regression to maximize predictive performance.

After tuning, we evaluate Ridge Regression's final performance on test data.

[255]:
```
print(gs_ridge.best_params_)
print(-gs_ridge.best_score_)
test_MSE, test_RMSE, test_R2 = model_fit(gs_ridge.best_estimator_,␣
 ↪X_test_ridge, y_test_ridge, plot=False)
print(f"Test Data - MSE: {test_MSE}, RMSE: {test_RMSE}, R²: {test_R2}")
```

```
{'ridge__alpha': np.float64(3.7896169806447157)}
30.867617305491393
Test Data - MSE: 32.176958301026986, RMSE: 5.672473737358947, R²:
0.7925016930361719
```

The test set metrics for Ridge Regression show an MSE of 32.18, RMSE of 5.67, and an R^2 score of 0.79. Compared to the baseline model (MSE = 39.23, R^2 = 0.75), Ridge Regression improves both accuracy and generalization. The regularization in Ridge Regression stabilizes coefficient values, preventing extreme variations. This confirms Ridge Regression as the best-performing model in our study

Then, we examine the final Ridge Regression coefficients to interpret feature importance.

[256]:
```
# Create dataframe with coefficents, and unstandardize the binary coeffcients
rcoefs = np.copy(gs_ridge.best_estimator_.named_steps['ridge'].coef_
)

rcoefs_ = pd.DataFrame(
    rcoefs,
    columns=["Coefficients"],
    index=preprocessor.get_feature_names_out(),
)

rcoefs_
```

[256]:

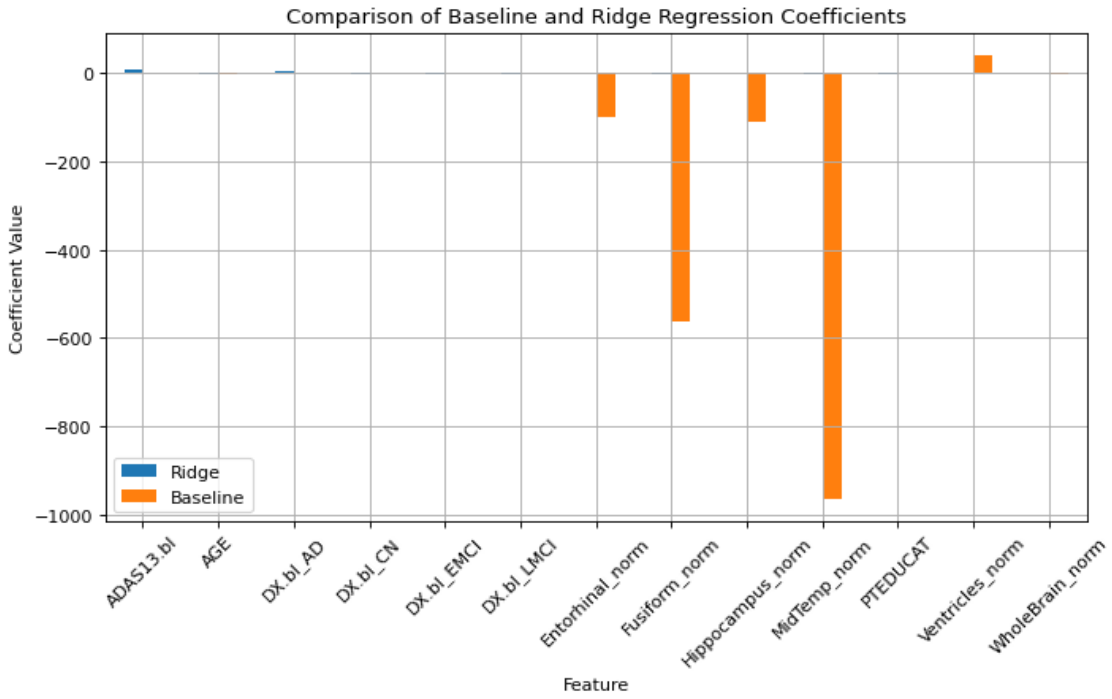
	Coefficients
cat__DX.bl_AD	5.238672
cat__DX.bl_CN	-2.717245
cat__DX.bl_EMCI	-2.215405
cat__DX.bl_LMCI	-0.306023
num__ADAS13.bl	8.550387
num__AGE	-1.161008
num__PTEDUCAT	-0.215783
num__Ventricles_norm	0.579609
num__Hippocampus_norm	0.083063
num__WholeBrain_norm	0.047583
num__Entorhinal_norm	0.031596
num__Fusiform_norm	-0.713227
num__MidTemp_norm	-1.706534

The coefficients from the Ridge Regression model reveal key insights into cognitive decline prediction. ADAS13.bl (8.55) is the strongest predictor of cognitive decline, followed by Alzheimer's diagnosis (DX.bl_AD) with a coefficient of 5.24, indicating that Alzheimer's diagnosis is highly associated with worse outcomes. Conversely, cognitively normal individuals (DX.bl_CN) have lower ADAS13.m24 scores, with a coefficient of -2.72. Ventricles_norm (0.58) shows that larger ventricles correlate with higher cognitive impairment, while smaller mid-temporal lobe volumes (MidTemp_norm, -1.71) are linked to worse cognitive function. Compared to the baseline model, the coefficients in Ridge Regression are smaller, confirming that regularization effectively prevents overfitting.

4.3 Comparison

Firstly, we create a bar chart in order to visually compare Ridge and Baseline model coefficients.

[260]:
```
if len(features_ridge) != 13:
    features_DX = ['DX.bl_AD', 'DX.bl_CN', 'DX.bl_EMCI', 'DX.bl_LMCI']
    features_ridge[:1] = features_DX

#  DataFrame
ridge_df = pd.DataFrame({'Feature': features_ridge, 'Ridge Coef': rcoefs})
baseline_df = pd.DataFrame({'Feature': features_baseline, 'Baseline Coef':␣
 ↪bcoefs})

#   DataFrame
coefs_df = pd.merge(ridge_df, baseline_df, on='Feature', how='outer').fillna(0)

#
coefs_df.set_index('Feature').plot(kind='bar', figsize=(10, 5))
plt.title("Comparison of Baseline and Ridge Regression Coefficients")
plt.ylabel("Coefficient Value")
plt.xlabel("Feature")
plt.xticks(rotation=45)
plt.legend(["Ridge", "Baseline"])
plt.grid()
plt.show()


baseline_norm = np.linalg.norm(bcoefs)
ridge_norm = np.linalg.norm(rcoefs)

print(f"Baseline Model L2 Norm: {baseline_norm}")
print(f"Ridge Model L2 Norm: {ridge_norm}")
```



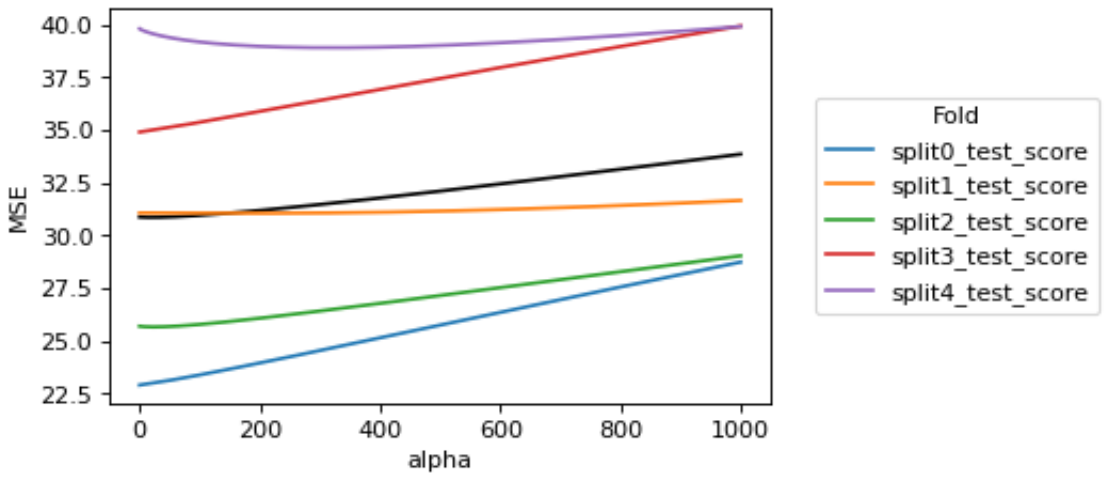

```
Baseline Model L2 Norm: 1127.8659487051507
Ridge Model L2 Norm: 10.867356992239564
```

The baseline model coefficients exhibit larger magnitudes, which may indicate potential overfitting. In contrast, Ridge Regression shrinks these coefficients, enhancing model stability. The L2 norm of Ridge Regression (10.86) is significantly lower than that of the Baseline Model (1127.87), confirming a reduction in model complexity and further supporting the effectiveness of regularization in preventing overfitting.

Then, we examine how Mean Squared Error (MSE) varies across folds in cross-validation to assess model stability.

[269]:
```
# Extract only mean and split scores
cv_mse = pd.DataFrame(
    data = gs_ridge.cv_results_
).filter(
    # Extract the split#_test_score and mean_test_score columns
    regex = '(split[0-9]+|mean)_test_score'
).assign(
    # Add the alphas as a column
    alpha = alphas
)

cv_mse.update(
    # Convert negative mses to positive
    -1 * cv_mse.filter(regex = '_test_score')
```

```
)

# Reshape the data frame for plotting
d = cv_mse.melt(
    id_vars=('alpha','mean_test_score'),
    var_name='fold',
    value_name='MSE'
)

# Plot the validation scores across folds
plt.figure()
sns.lineplot(x='alpha', y='MSE', color='black', errorbar=None, data = d)  #␣
 ↪Plot the mean MSE in black.
sns.lineplot(x='alpha', y='MSE', hue='fold', data = d) # Plot the curves for␣
 ↪each fold in different colors
plt.legend(title='Fold', bbox_to_anchor=(1.05, 0.5), loc='center left')
plt.show()
```



When is small, the MSE varies significantly across different folds, indicating that the model is sensitive to data partitioning and lacks stability. As increases, the MSE curves converge, suggesting that higher regularization improves stability by preventing large coefficient variations. However, an excessively large causes MSE to increase, indicating underfitting and a loss of predictive power. This highlights the trade-off between bias and variance, and the optimal should be chosen to minimize MSE while maintaining the model's robustness.

5 Discussion & Conclusions

5.1 Overview of the Final Model

This study developed a Ridge Regression model to predict ADAS13.m24, a measure of cognitive decline over 24 months. Ridge Regression was chosen over a Linear Regression baseline due to

its ability to reduce overfitting and improve generalization through L2 regularization. The model incorporated baseline cognitive scores (ADAS13.bl), diagnostic categories (DX.bl), and structural brain features such as ventricular and hippocampal volumes.

The final Ridge Regression model demonstrated superior predictive performance compared to the baseline linear regression, with the test MSE decreasing from 39.23 to 32.18, indicating improved generalization and reduced overfitting. By applying L2 regularization, Ridge Regression effectively stabilized coefficient estimates, as evidenced by the coefficient norm reduction from 1127.87 to 10.87, ensuring a more interpretable and reliable model. Additionally, the key predictors remained clinically relevant, reinforcing established relationships between cognitive impairment and structural brain changes—notably, higher baseline ADAS13.bl scores strongly predicted cognitive decline, while larger ventricular volumes and smaller hippocampal volumes were associated with worse outcomes. These findings confirm that Ridge Regression is a valuable tool for early detection and intervention, supporting personalized treatment strategies and targeted clinical trials for high-risk individuals.

5.2 Reliability and Practical Implications

This predictive model provides health officials and charity organizations with a data-driven approach to understanding and addressing cognitive decline, enabling more proactive and strategic decision-making in aging and dementia care. By identifying individuals at higher risk of cognitive deterioration based on baseline cognitive scores, demographic factors, and brain structural features, the model allows for earlier interventions, such as cognitive training, neurological monitoring, and preventive healthcare programs.

It also helps optimize resource allocation by guiding public health initiatives, funding distribution, and the expansion of dementia care services, ensuring that support reaches the most vulnerable populations.Additionally, the model can aid clinical research and treatment development by facilitating targeted clinical trial recruitment and improving patient selection for experimental therapies.

For policymakers and charity organizations, these insights support evidence-based policy proposals, funding advocacy, and the creation of targeted support programs for patients and caregivers. While the model should be used as a supplementary tool rather than a standalone diagnostic method, its ability to enhance early detection, resource planning, and strategic intervention makes it a valuable asset in the fight against cognitive decline and dementia.

5.3 Limitations of the Model

First, it identifies correlations rather than causation, meaning it can highlight associations between cognitive decline and specific factors but cannot establish direct causal relationships, limiting its role in clinical decision-making. **Second**, there is potential selection bias, as the dataset primarily consists of individuals enrolled in Alzheimer's research studies, which may not fully represent the broader population at risk of cognitive decline. **Additionally**, the model does not account for key environmental and lifestyle factors, such as diet, physical activity, and social engagement, which have been shown to influence cognitive health. **Finally**, its generalizability is limited, as it was trained on a specific cohort, and further validation is needed across diverse populations before it can be reliably applied in clinical or public health settings. Addressing these limitations through expanded datasets, additional feature incorporation, and external validation will be crucial for improving the model's robustness and applicability.

5.4 Future Research Directions

One important direction is to expand feature selection to include lifestyle, genetic, and socioeconomic factors, as these elements play a crucial role in dementia risk and would provide a more comprehensive understanding of cognitive decline. Another key focus should be on shifting from predicting cognitive impairment at a fixed 24-month endpoint to tracking cognitive trajectories over time, enabling dynamic forecasting and improved early intervention strategies. It is also essential to validate the model on diverse populations beyond Alzheimer's-specific datasets to ensure its applicability in broader clinical and public health settings. Lastly, integrating multi-modal data sources—such as neuroimaging, genetic markers, and patient-reported outcomes—would help build a more holistic and personalized risk assessment framework, ultimately leading to more precise and individualized intervention strategies for cognitive decline.

5.5 Recommendations for Cognitive Decline Prevention

Based on the model's insights, several key strategies can help mitigate cognitive decline and support early intervention efforts. One important strategy is early screening and cognitive stimulation for individuals with high baseline ADAS13.bl scores, as they are more likely to experience significant cognitive decline. Another approach is regular neurological monitoring, since brain structure changes, like ventricular enlargement and hippocampal atrophy, are strong indicators of worsening cognition. It's also helpful to encourage lifelong learning and mental engagement, as staying mentally active can improve cognitive resilience and delay decline. Lastly, patients with Mild Cognitive Impairment (MCI) or early-stage Alzheimer's should be prioritized for treatment and clinical trials, as early intervention can have the greatest impact.

6 Generative AI statement

In this project, we used generative AI solely as a tool to assist with debugging code. All data and analysis results were processed and evaluated by us. The AI helped us refine our code and address errors, ensuring the technical aspects of the project were accurate and functional. The final outcomes and interpretations are entirely our own, and we have adhered to responsible use of AI as outlined in the guidelines.

7 References

Include references if any

[262]:
```
# Run the following to render to PDF
!jupyter nbconvert --to pdf project1.ipynb
```

```
[NbConvertApp] Converting notebook project1.ipynb to pdf
[NbConvertApp] Support files will be in project1_files\
[NbConvertApp] Making directory .\project1_files
[NbConvertApp] Writing 106909 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | b had problems, most likely because there were no
```

```
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 3417204 bytes to project1.pdf
```