Emre Aktas & Joseph Tschopp

03/24/2024

DSCC 265 - Problem Set 4

1. Q1

Edmond Awad et al.'s experiment with Moral Machine answers a pressing challenge: how should an autonomous vehicle decide what to do when it finds itself in a lose-lose accident scenario? The experiment seeks to enlist the help of an internet-based platform, the Moral Machine, to collect huge-scale data on societal preferences about the ethical principles that should guide dilemmas in autonomous vehicle behavior. Participating in 233 countries and territories, 40 million decisions were received over 40 million.

This complete dataset is in the public domain to analyze and discuss the topic further. It outlines global moral preferences, documents the variations in such preferences by demographic factors, reports cross-cultural ethical variations, and explores the correlation of such variations with modern institutions and deep cultural traits.

In a very strong fashion, the survey found that people from all walks of life around the world agree that "people's lives should be saved before animals" and "more people's lives should be saved rather than fewer," with a more substantial bias toward younger people more than the elderly. Still, for such similarities, the differences ranged across the individual demographics and cultural backgrounds to some reasonably large extent. It finds that three main clusters of countries have homogeneous moral preferences, and this gives the hint that although global principles for machine ethics can be pursued, at the same time, cultural differences should be acknowledged and given respect.

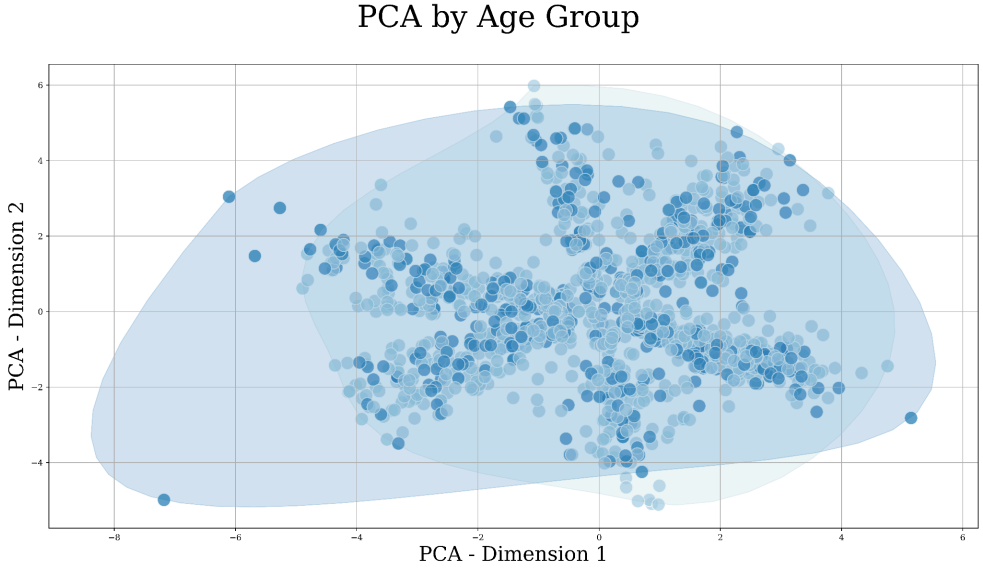
Such findings help policymakers, automakers, and the general public frame ethical considerations. This may help in building socially informed and globally acceptable machine ethics principles. It draws attention to the need to incorporate different voices when discussing the ethics of driverless cars and simultaneously points to public morality's role in deciding the outlines of a future in an AI and automation society.

2. Q2

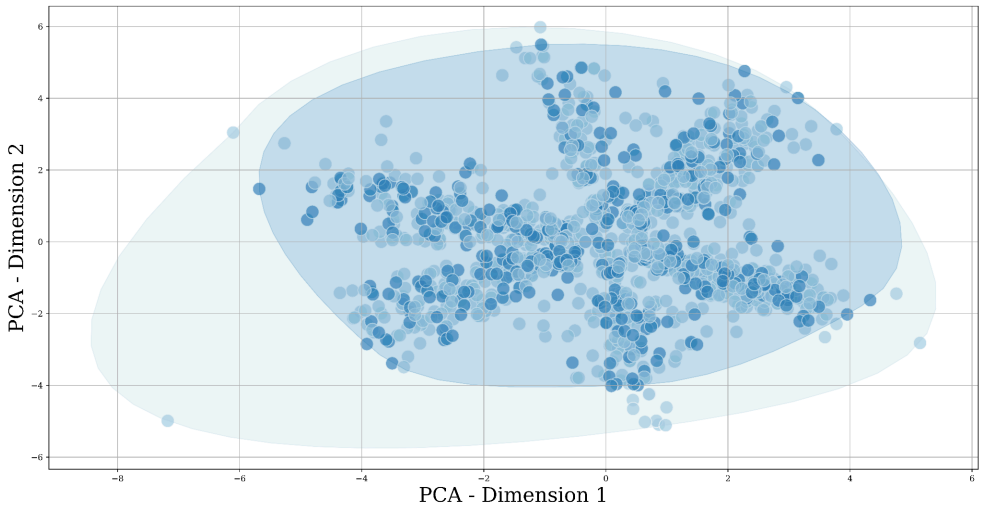
```
# Set random seed for reproducibility
np.random.seed(265)

def pca(X, num_components):
    # Standardize the dataset
    X_standardized = (X - np.mean(X, axis=0)) / np.std(X, axis=0)
    # Compute covariance matrix
    covariance_matrix = np.cov(X_standardized.T)
    # Compute eigenvalues and eigenvectors
    eigenvalues, eigenvectors = np.linalg.eig(covariance_matrix)
    # Sort eigenvectors by eigenvalues in descending order
    idx = eigenvalues.argsort()[::-1]
    sorted_eigenvectors = eigenvectors[:, idx]
    # Select the first num_components eigenvectors
    projection_matrix = sorted_eigenvectors[:, :num_components]
    # Project the data onto the selected eigenvectors
    X_pca = np.dot(X_standardized, projection_matrix)
    return X_pca

# Specify the path to your Excel file
file_path = 'moral_machine_data.xlsx'

# Read the Excel file
df = pd.read_excel(file_path, engine='openpyxl')
results_columns = [...

X = df[results_columns].values

# Perform PCA with 2 components
X_pca = pca(X, num_components=2)

print(X_pca)
```

3. Q3

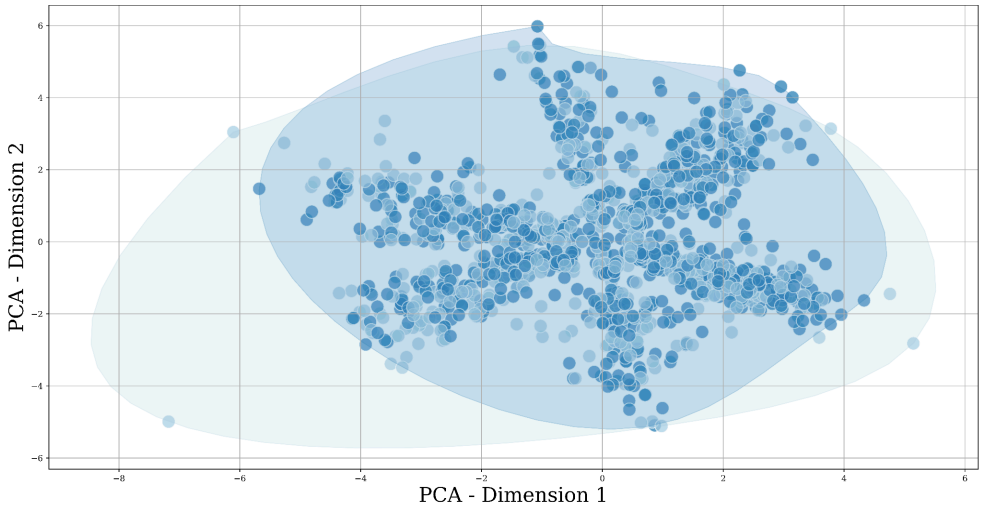
```
# Prepare the DataFrame with PCA components
pca_df = pd.DataFrame(X_pca, columns=['pca_dim_1', 'pca_dim_2'])
df = pd.concat([df.reset_index(drop=True), pca_df], axis=1)

# Add a column for the total number of siblings
df['no_of_siblings'] = df['no_of_sisters'] + df['no_of_brothers']

# Bin 'age' into categorical age groups
df['age_group'] = pd.cut(df['age'], bins=[0, 18, 30, 40, 50, 60, np.inf], labels=['0-18', '19-30', '31-40', '41-50', '51-60', '60+'])

# Plotting settings
mpl.rcParams['figure.dpi'] = 600
plt.rcParams['figure.figsize'] = (20.0, 10.0)
plt.rcParams['font.family'] = "serif"
plt.rcParams['font.size'] = 10

# Function to plot PCA with labels and optional text labels
def plot_pca_with_labels(df, label_column, text_column=None, title=''):
    # Obtain unique labels and corresponding colors
    unique_labels = df[label_column].unique()
    pal = sns.color_palette("Paired", n_colors=len(unique_labels))
    # Map labels to a corresponding numeric value
    label_mapping = {label: i for i, label in enumerate(unique_labels)}
    df['mapped_label'] = df[label_column].map(label_mapping)

    # Create scatter plot
    p1 = sns.scatterplot(x="pca_dim_1", y="pca_dim_2", hue='mapped_label', palette=pal, data=df, s=250, alpha=0.7, legend=False)

    # Add text labels if a column for them is provided
    if text_column:
        for line in range(0, df.shape[0]):
            p1.text(df.pca_dim_1[line], df.pca_dim_2[line], df[text_column][line], horizontalalignment='left', size='medium', color='black', weight='semibold')

    # Set the plot title and axes labels
    plt.suptitle(title, fontsize=36)
    plt.xlabel('PCA - Dimension 1', fontsize=24)
    plt.ylabel('PCA - Dimension 2', fontsize=24)

    # Draw convex hulls for clusters
    for i in unique_labels:
        mapped_label = label_mapping[i]
        points = df[df['mapped_label'] == mapped_label][['pca_dim_1', 'pca_dim_2']].values
        if points.shape[0] > 2:  # Convex hull requires at least 3 points
            hull = ConvexHull(points)
            x_hull = np.append(points[hull.vertices,0], points[hull.vertices,0][0])
            y_hull = np.append(points[hull.vertices,1], points[hull.vertices,1][0])

            # Interpolate
            dist = np.sqrt((x_hull[:-1] - x_hull[1:])**2 + (y_hull[:-1] - y_hull[1:])**2)
            dist_along = np.concatenate(([0], dist.cumsum()))
            spline, u = interpolate.splprep([x_hull, y_hull], u=dist_along, s=0)
            interp_d = np.linspace(dist_along[0], dist_along[-1], 50)
            interp_x, interp_y = interpolate.splev(interp_d, spline)

            # Plot shape
            plt.fill(interp_x, interp_y, '--', c=pal[mapped_label], alpha=0.2)

    plt.grid()
    plt.show()

# Create plots for each of the categories
plot_pca_with_labels(df, 'age_group', title='PCA by Age Group')
plot_pca_with_labels(df, 'gender ', title='PCA by Gender')
plot_pca_with_labels(df, 'grown_up_in_US ', title='PCA by Grown Up in US')
plot_pca_with_labels(df, 'country_of_origin', text_column='country_of_origin', title='PCA by Country of Origin')
plot_pca_with_labels(df, 'no_of_siblings', title='PCA by Number of Siblings')
```



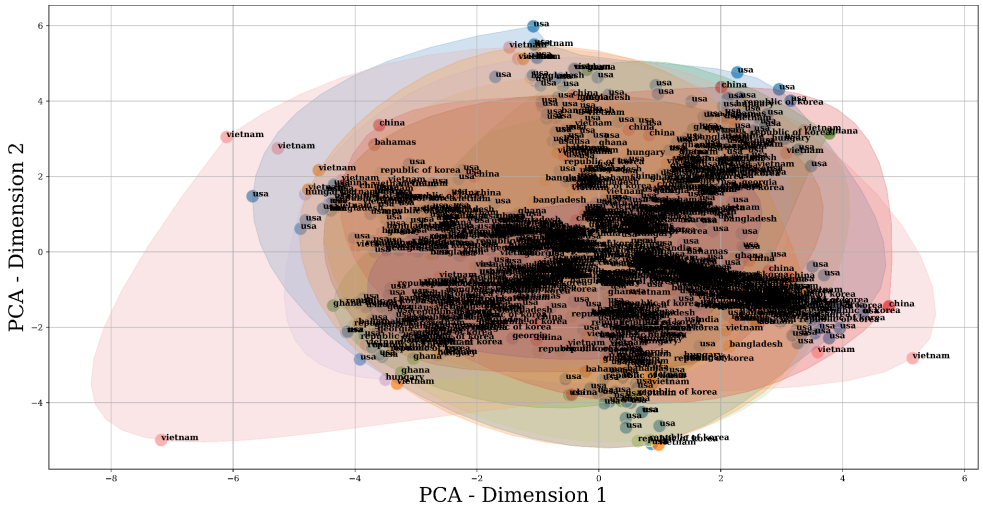
PCA by Gender

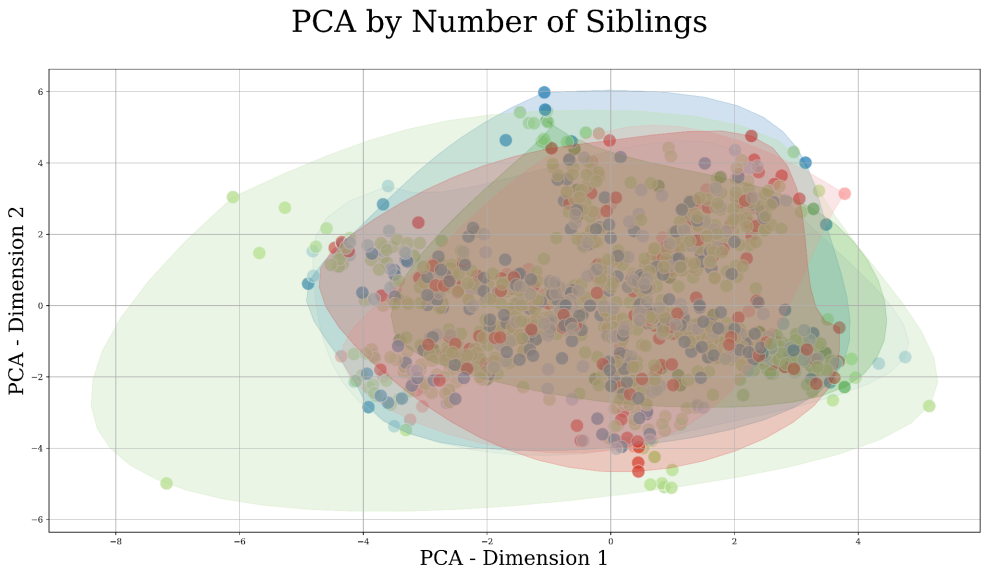


PCA by Grown Up in US



PCA by Country of Origin





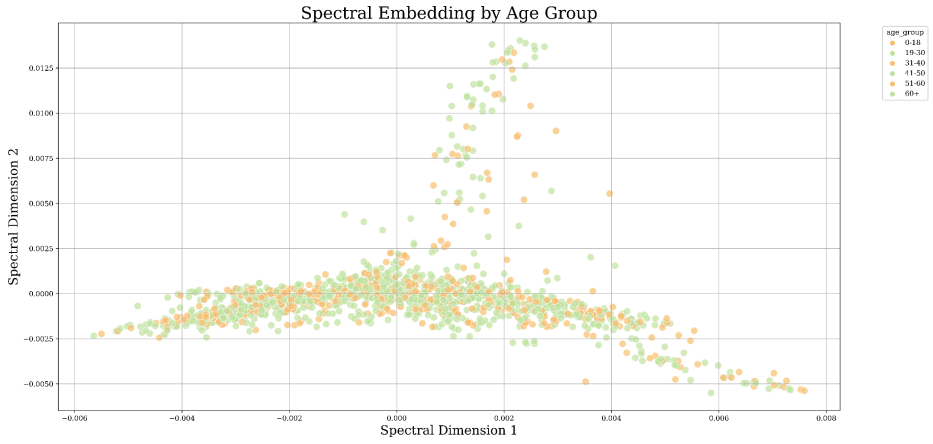
f.

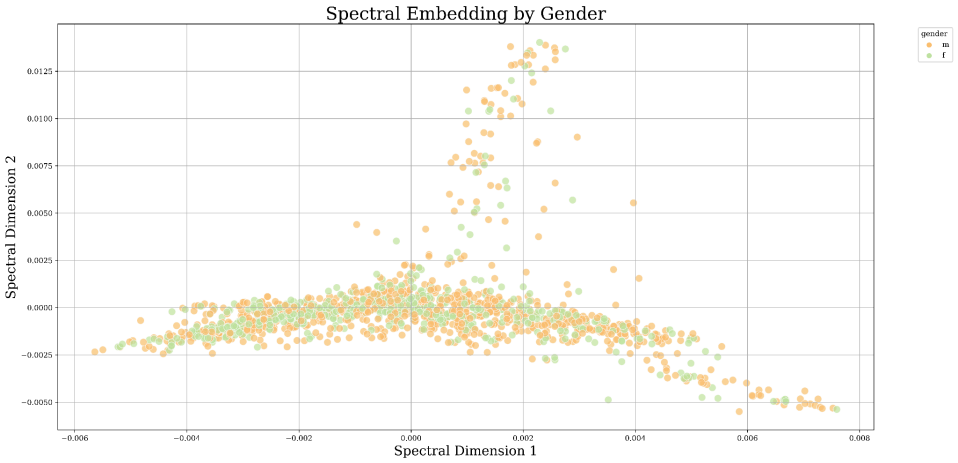
A scatter plot showing a star shape with six edges across the principal components indicates a consistent distribution pattern of data for different types of clustering based on various columns such as age, gender, grown_up_in_US, country_of_origin, and no_of_siblings. This pattern is typical and suggests that the principal components effectively capture the structural characteristics of the data. The variability in the dataset is visually represented and clear, thereby helping to differentiate between values within each column. The star-shaped pattern appearing in scatterplots for different columns indicates that the principal components capture a significant amount of variance and correlations that are not entirely associated with the columns used for clustering. For example, clustering based on age or number of siblings may reveal clusters for demographic groups with similar characteristics or behaviors, while clustering based on country of origin or whether raised in the US may show clusters representing cultural or geographical effects on the variables included in the dataset. The quality of the clustering pattern depends on how well-separated the clusters are and whether or not they yield meaningful

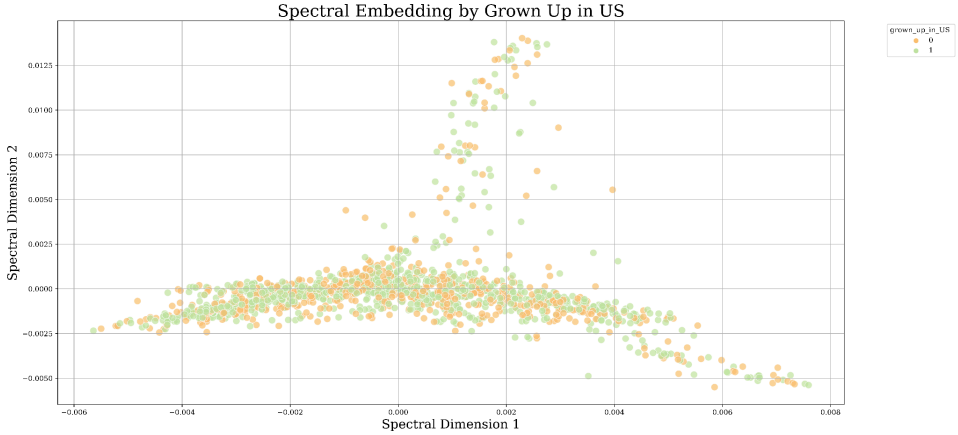
separations, given the context of the data. In the scatter plots, it is evident that country_of_origin generates the best clustering pattern, although this is mostly a matter of opinion.

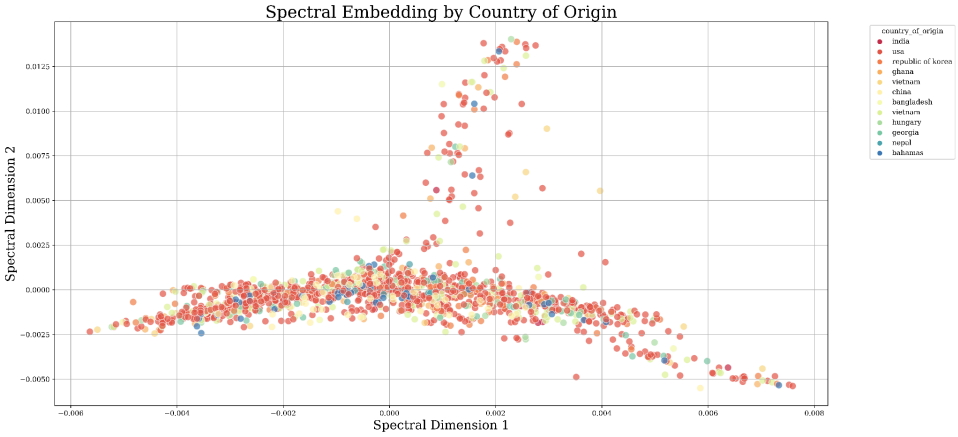
4. Q4

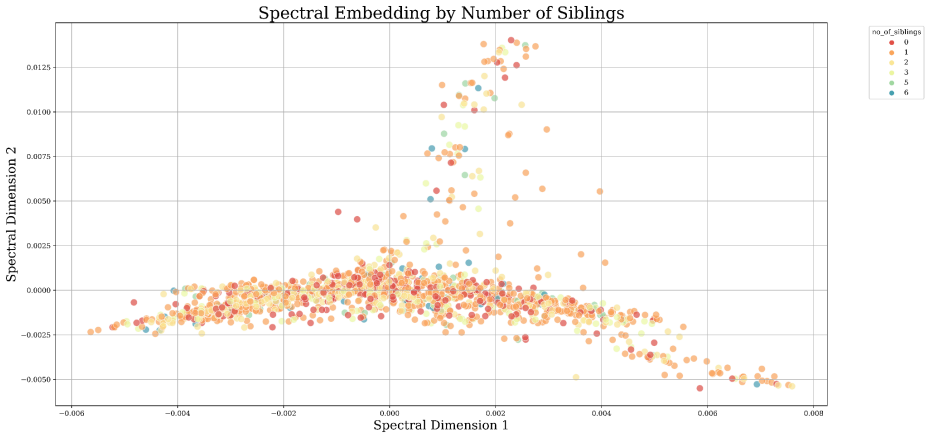
```python
# Apply Spectral Embedding for dimensionality reduction
embedding = SpectralEmbedding(n_components=2, random_state=0)
X_transformed = embedding.fit_transform(X)

# Create a DataFrame for the transformed data
spectral_df = pd.DataFrame(X_transformed, columns=['spectral_dim_1', 'spectral_dim_2'])

# Combine with original data
df = pd.concat([df.reset_index(drop=True), spectral_df], axis=1)

# Function to plot the results of spectral embedding
def plot_spectral_with_labels(df, label_column, title):
    plt.figure(figsize=(20, 10))
    plt.rcParams['font.size'] = 10

    # Determine the number of unique labels for the color palette
    unique_labels = df[label_column].unique()
    palette = sns.color_palette("Spectral", n_colors=len(unique_labels))

    # Set plot title, labels, and legend
    sns.scatterplot(data=df, x='spectral_dim_1', y='spectral_dim_2', hue=label_column, palette=palette, s=100, alpha=0.7)

    # Set plot title, labels, and legend
    plt.title(title, fontsize=24)
    plt.xlabel('Spectral Dimension 1', fontsize=18)
    plt.ylabel('Spectral Dimension 2', fontsize=18)
    plt.legend(title=label_column, bbox_to_anchor=(1.05, 1), loc='upper left')
    plt.grid(True)
    plt.show()

# Plot spectral embedding results for different categorical variables
plot_spectral_with_labels(df, 'age_group', 'Spectral Embedding by Age Group')
plot_spectral_with_labels(df, 'gender ', 'Spectral Embedding by Gender')
plot_spectral_with_labels(df, 'grown_up_in_US ', 'Spectral Embedding by Grown Up in US')
plot_spectral_with_labels(df, 'country_of_origin', 'Spectral Embedding by Country of Origin')
plot_spectral_with_labels(df, 'no_of_siblings', 'Spectral Embedding by Number of Siblings')
```




Spectral Embedding by Gender
gender
m
f
Spectral Dimension 2
Spectral Dimension 1


Spectral Embedding by Grown Up in US
grown_up_in_US
0
1
Spectral Dimension 2
Spectral Dimension 1


Spectral Embedding by Country of Origin
country_of_origin
india
usa
republic of korea
ghana
vietnam
china
bangladesh
vietnam
hungary
georgia
nepal
bahamas
Spectral Dimension 2
Spectral Dimension 1

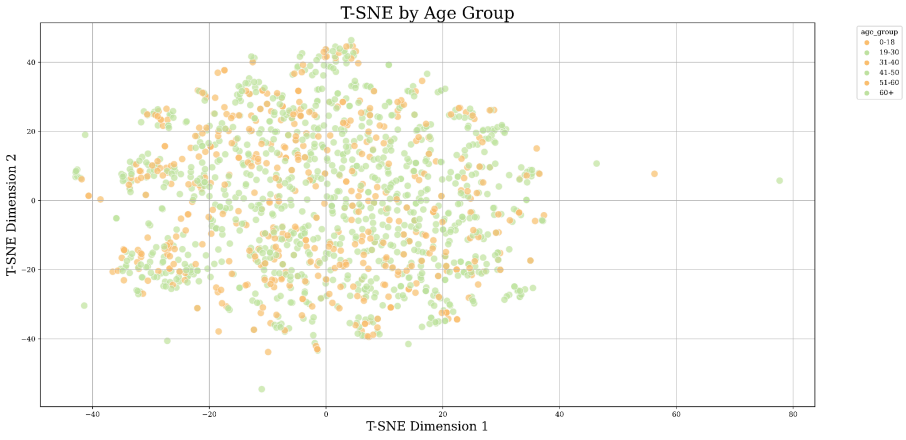


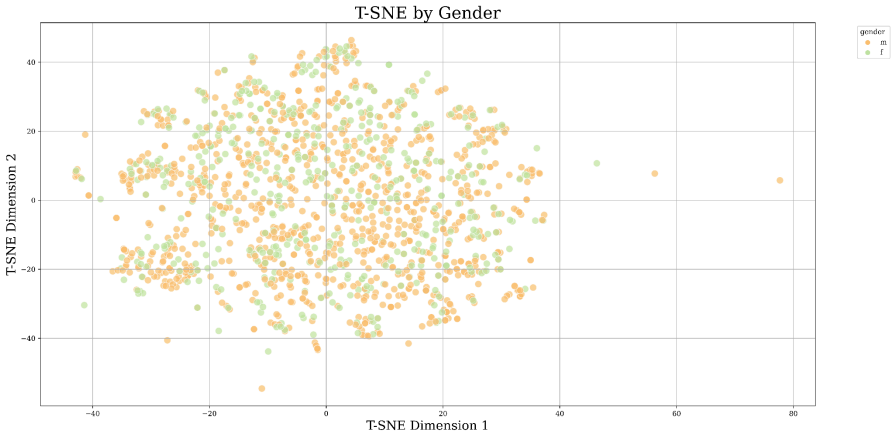
b.

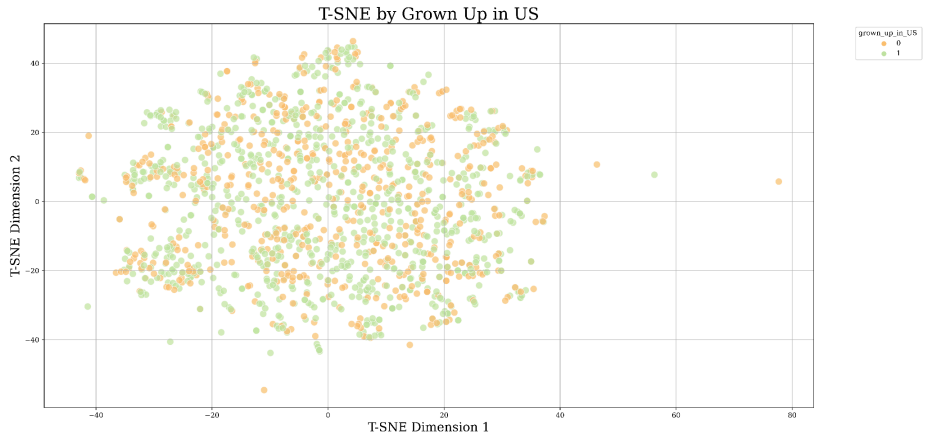
The spectral embedding visualization suggests subtle patterns for age groups and the number of siblings; however, they do not form distinct clusters, indicating a lack of strong differentiation in the dataset's lower-dimensional space. Gender and whether one grew up in the US appear largely indistinct, with a broad overlap across the spectral dimensions. The 'Country of Origin' visualization shows the most potential for clustering, with some data points concentrated by nationality. This suggests that country-specific characteristics influence the dataset's structure more significantly than other variables. Comparatively, while PCA is a linear dimensionality reduction technique that captures global structure, spectral embedding focuses on preserving neighborhood relationships, which might be better at uncovering local clusters. However, in this case, spectral embedding does not outperform PCA noticeably. Neither method results in clear, well-separated clusters for the variables examined. Still, PCA slightly edges forward with a more discernible pattern, particularly for the 'Number of Siblings' and 'Country of Origin,' where some grouping is apparent.

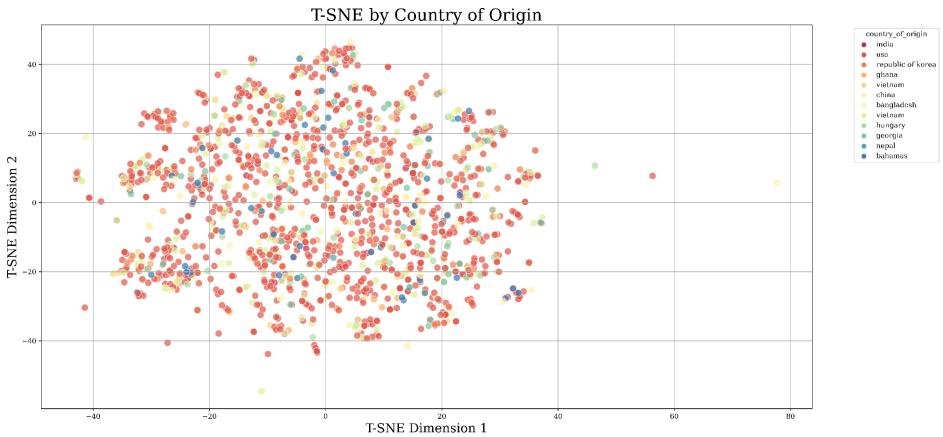
5. Q5

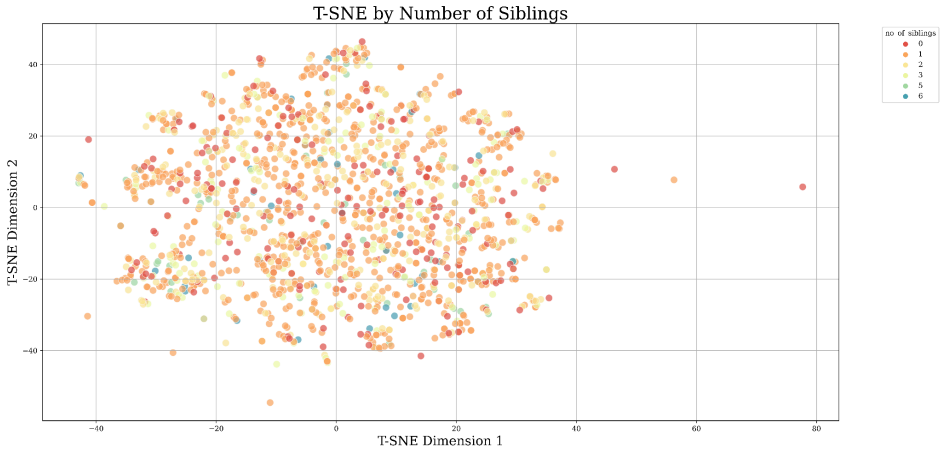
```python
# Apply t-SNE to the dataset for dimensionality reduction
tsne = TSNE(n_components=2, random_state=42)
X_tsne = tsne.fit_transform(X)

# Create a DataFrame from the t-SNE components
tsne_df = pd.DataFrame(X_tsne, columns=['tsne_dim_1', 'tsne_dim_2'])

# Merge the t-SNE dimensions into the original DataFrame
df = pd.concat([df.reset_index(drop=True), tsne_df], axis=1)

# Define a function to create scatter plots using t-SNE dimensions
def plot_tsne_with_labels(df, label_column, title):
    # Configure the figure size and font size
    plt.figure(figsize=(20, 10))
    plt.rcParams['font.size'] = 10

    unique_labels = df[label_column].unique()
    palette = sns.color_palette("Spectral", n_colors=len(unique_labels))

    # Plot t-SNE results with labels
    sns.scatterplot(data=df, x='tsne_dim_1', y='tsne_dim_2', hue=label_column, palette=palette, s=100, alpha=0.7)

    # Set title and labels for the axes
    plt.title(title, fontsize=24)
    plt.xlabel('T-SNE Dimension 1', fontsize=18)
    plt.ylabel('T-SNE Dimension 2', fontsize=18)
    plt.legend(title=label_column, bbox_to_anchor=(1.05, 1), loc='upper left')
    plt.grid(True)
    plt.show()

# Plot the t-SNE results for various demographic categories
plot_tsne_with_labels(df, 'age_group', 'T-SNE by Age Group')
plot_tsne_with_labels(df, 'gender ', 'T-SNE by Gender')
plot_tsne_with_labels(df, 'grown_up_in_US ', 'T-SNE by Grown Up in US')
plot_tsne_with_labels(df, 'country_of_origin', 'T-SNE by Country of Origin')
plot_tsne_with_labels(df, 'no_of_siblings', 'T-SNE by Number of Siblings')
```




T-SNE by Gender
T-SNE Dimension 2
T-SNE Dimension 1
gender
m
f


T-SNE by Grown Up in US
T-SNE Dimension 2
T-SNE Dimension 1
grown_up_in_US
0
1


T-SNE by Country of Origin
T-SNE Dimension 2
T-SNE Dimension 1
country_of_origin
india
usa
republic of korea
ghana
vietnam
china
bangladesh
vietnam
hungary
georgia
nepal
bahamas



b.

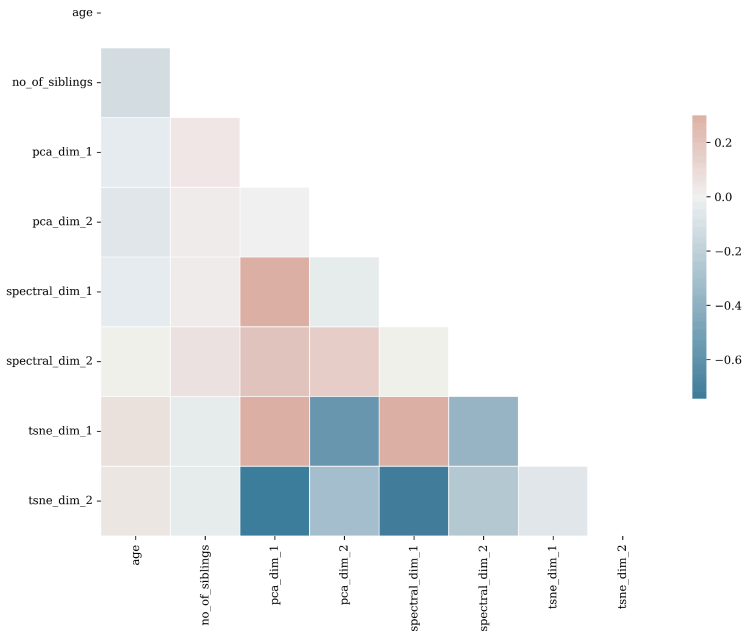
Upon analyzing the t-SNE visualizations, distinct patterns become apparent across various demographic segments of the data. The 'Age Group' visualization displays some clustering, particularly in the youngest and oldest segments, although the middle age groups tend to overlap. This suggests that age may have a non-linear impact on the data structure. 'Country of Origin' produces the clearest clustering, indicating distinct groupings that likely reflect cultural or regional differences. On the other hand, the 'Gender' visualization lacks clear separation, hinting that gender may not be a strong distinguishing factor in the dataset. For 'Number of Siblings,' individuals without siblings form a slightly more distinct cluster while others largely overlap, indicating only a weak pattern. Those who grew up in the US also appear to form clusters, implying that cultural factors may be at play.

When compared to PCA and Spectral Embedding, t-SNE often creates more defined clusters due to its non-linear nature, which can reveal complex structures not visible in PCA's linear projections. For this dataset, 'Country of Origin' and 'Grown Up in US' are standout variables in t-SNE visualizations, forming more visible clusters than what spectral embedding suggests. This implies that t-SNE is more effective at capturing and illustrating the subtle

distinctions of this dataset, especially where intrinsic groupings are concerned. Therefore, while PCA and Spectral Embedding provide valuable insights, t-SNE's ability to highlight granular distinctions may offer better clarity for specific analytical purposes, especially when understanding demographic clustering is crucial.

6. Q6

```
# Columns related to dimensionality reduction methods and original features
dim_reduction_cols = ['age',
                      'no_of_siblings',
                      'pca_dim_1', 'pca_dim_2',
                      'spectral_dim_1', 'spectral_dim_2',
                      'tsne_dim_1', 'tsne_dim_2']

# Subset the DataFrame to include only the specified columns for correlation analysis
df_selected = df[dim_reduction_cols]

# Calculate the correlation matrix for the selected columns
corr = df_selected.corr()

# Create a boolean mask for the upper triangle of the correlation matrix
mask = np.triu(np.ones_like(corr, dtype=bool))

# Initialize a matplotlib figure with a specified size
f, ax = plt.subplots(figsize=(11, 9))

# Define the color map for the heatmap
cmap = sns.diverging_palette(230, 20, as_cmap=True)

# Draw the heatmap with the mask and aspect ratio settings
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})

plt.show()
```



b.

The heatmap visualization reveals interesting correlations between the original dataset variables and the ones obtained through dimensionality reduction techniques. The variable 'age' displays a negative correlation with the second dimension from t-SNE (tsne_dim_2), which implies that as age increases, the values in tsne_dim_2 tend to decrease. This could suggest that tsne_dim_2 captures some age-related variance in the data. On the other hand, 'Number of siblings' does not have a strong correlation with any of the dimension reduction variables, which implies that sibling count may not be a significant factor in the variance these techniques aim to capture.

From the PCA results, pca_dim_1 and pca_dim_2 don't display any strong correlation with 'age' or 'number of siblings', which indicates that the principal components may not strongly represent these particular variables. However, there is a mild correlation between 'number of siblings' and pca_dim_2, which suggests that the second principal component may capture some aspects related to family size.

In Spectral Embedding, spectral_dim_1 and spectral_dim_2 also do not show any strong correlations with 'age' or 'number of siblings'. This lack of strong correlation suggests that these spectral dimensions may capture other variances within the dataset that are not directly related to these specific original variables.

Overall, the dimensionality reduction variables do not seem to strongly represent 'age' or 'number of siblings' from the original dataset, based on the observed correlations. This could mean that the most significant variances captured by PCA, Spectral Embedding, and t-SNE are likely related to other unobserved variables in the dataset or these demographic variables are distributed in a complex way that is not linearly separable. While the dimensionality reduction

techniques are useful for visualizing high-dimensional data, the heatmap indicates that the primary demographic variables 'age' and 'number of siblings' are not predominantly represented in the reduced dimensions, at least not linearly. Therefore, these findings can inform further analysis, suggesting a need to either apply different dimensionality reduction methods that may better capture the essence of these variables or to investigate other variables in the dataset that contribute more to the observed variance.