Project CS 556

The project is worth 100 points in total. All teams need to have 3 members.

What to submit:

1. You will be provided with a jupyter notebook to give your requisite code / analysis.

2. In addition, you can use this word document to answer all the non-coding questions (inline) and submit a single document for all the analysis. (Questions will mostly be localized to the "Discussion" section of each question.

Submission format: You need to submit a zipped archive containing the jupyter notebook with your code and a pdf file named *project_questions_and_solutions.pdf* (i.e., the current file with the solutions that you will fill in and convert to PDF format).

Each of your jupyter notebooks should be directly executable without any further modifications (i.e., if your assignment submission is downloaded from canvas and unzipped), any of your submitted notebooks if executed using a jupyter server should be directly runnable without additional data downloads etc.

Housing Price Prediction

a. **Dataset Description:** The data pertains to the houses found in each California district and some summary statistics about them based on the 1990 census data. It contains one instance per district block group. A block group is the smallest geographical unit for which the U.S. Census Bureau publishes sample data (a block group typically has a population of 600 to 3,000 people).
The goal of this task is to design a regression model to predict the *median house value* conditioned upon a set of input attributes corresponding to a particular California district block.
The attributes in the dataset are as follows; their names are self-explanatory:

   i. longitude (continuous): One of the coordinates that are used to identify the California district block

   ii. latitude (continuous): One of the coordinates that are used to identify the California district block

   iii. housing_median_age (continuous): Average age of the house in California district block

   iv. total_rooms (continuous): Total number of rooms of all the houses in the California district block

   v. total_bedrooms (continuous): Total number of bedrooms of all the houses in the California district block

   vi. population (continuous): Number of people residing in the district block

   vii. households (continuous): Number of families in the district block

   viii. median_income (continuous): Median income for households in the district block of houses (measured in tens of thousands of US Dollars)

ix. ocean_proximity (categorical): Location of the house. Is it inland, near the bay, near the ocean, etc.

x. median_house_value.(continuous): Median house value within a district block (measured in US Dollars)

Our target variable will be *median_house_value*. Use the rest of the fields mentioned above to predict the *median_house_value*.

b. **Data Loading / Preprocessing:**

i. Loading:

1. Load the California housing dataset using pandas.read_csv() function and store it in the variable (i.e., a pandas dataframe) named `df'.

2. The resulting data frame should have the shape (20640, 10) indicating that there are 20640 rows and 10 columns.

3. Find the missing values in the data frame. If any (i.e., even if one column in each instance / row has a missing value), drop the row using pandas.DataFrame.dropna() function. The resulting data frame should have the shape (20433, 10) indicating that there are 20433 rows and 10 columns.

4. Create a data frame 'corr_df' by dropping the columns *latitude*, *longitude*, and *ocean_proximity* using the pandas.DataFrame.drop() function. Use the *Pearson correlation* to find the correlation of each remaining feature in the 'corr_df' with the target variable '*median_house_value*' using the function pandas.DataFrame.corrwith(). Report results in the following table.

Feature Name	**Pearson Correlation**
housing_median_age	0.106432
total_rooms	0.133294
total_bedrooms	0.049686

population	-0.025300
households	0.064894
median_income	0.688355

5. Create a data frame X of features (by dropping the column '*median_house_value*'' from the original data frame) using the pandas.DataFrame.drop() function. Create a Series object of targets Y (by only considering the '*median_house_value*' column from the original data frame (Do NOT use the 'corr_df' data frame in this step. Use the data frame which was obtained as a result of step b.i.3 above).

ii. Data Visualization:

1. Use pandas.DataFrame.hist(bins = 50) function for visualizing the variation on the columns *housing_median_age*, *total_rooms*, *total_bedrooms*, *population*, *household*, *median_income* and *median_house_value*. Plot each histogram as a separate subplot.

2. Use pandas.dataframe.describe() function to find the mean, median and standard deviations for each feature and report in the jupyter notebook.

3. Use pandas.get_dummies to convert categorical variables into dummy /one-hot encoding. In this case the categorical column is *ocean_proximity*

iii. Data Splitting:

1. Split data into training and test sets using the sklearn train_test_split() function. Perform 70-30 distribution i.e. 70% training and 30% testing. The result of your data split should yield 4 separate data frames X_train, X_test, y_train, y_test. (respectively, the training features, testing features, training targets and testing target).

iv. Data Scaling:

1. Use the StandardScaler() to instantiate the standard scaler class. **Note**: You will need two separate scaler objects, one to scale the features, another to scale the target values.

2. For each scaler, employ the `fit_transform()' function (only on the training features, training targets) of the scaler to retrieve the new (scaled) version of the data. Store them in *X_train*, and *y_train* again.

3. Scale the *X_test* and *y_test* as well and store the scaled values back in X_test and y_test. (i.e., use the appropriate "fitted" scaler above to "transform" the test data. Note: the function to be employed in this case is `transform()` as opposed to `fit_transform()`).
   Henceforth, X_train, y_train, X_test, y_test will refer to the scaled data unless stated otherwise.

4. Use pandas.DataFrame.hist(bins = 50) function for visualizing the variation of numerical attributes *housing_median_age*, *total_rooms*, *total_bedrooms*, *population*, *household*, *median_income* and *median_house_value* for the *X_train* and *y_train* dataset (similar to step b.ii.1 above). Once again, plot each histogram as a separate subplot.

c. Modelling:

i. Employ Linear Regression from sklearn.linear_model, and instantiate the model.

ii. Once instantiated, `fit()` the model using the *scaled* X_train, y_train data.

iii. Employ the `predict()` function to obtain predictions on X_test. Store the predictions in a variable named `y_preds`. Note: Since the model has been trained on scaled data (i.e., both features and targets, the predictions will also be in the "scaled" space. We need to transform

the predictions back to the original space).

iv. Use inverse_transform() function to convert the normalized data (`y_preds`) to original scale. Store the transformed values back into `y_preds`.

v. Perform PCA on the features (*X_train*) and set *n_component* as 2.
   1. Show a scatter plot where on the x-axis we plot the first PCA component and second component on the y-axis.

   2. Calculate the total percentage of variance captured by the 2 PCA components using pca.explained_variance_ratio_. Also, report the strength of each PCA component using *pca.singular_values_*.

d. Evaluation:

i. Plot a scatter plot using matplotlib.pyplot.scatter function. Plot the predicted median house values on the y-axis vs the actual median house values on the x-axis.

ii. Calculate MAPE, RMSE and R^2 for the model and report them in the following table.
Hint for RMSE set the *squared* parameter to False.

Model	MAPE	RMSE	R^2
Linear Regression	0.28735	67927.63804	0.65388

e. Discussion:

i. Based on the weights of the linear regression model, rank the features (note: only continuous features) in decreasing order of influence on the predictions.

The features were ranked based on the weight(coefficients) of the linear regression model. Since we are investigating it's influence, the absolute value (i.e. magnitude) of the weights were taken. As a result, the obtained rank is presented below. According to the rank, we can see that median_income had the largest influence on the predictions, then latitude and longitude, which was similar, and then total_bedrooms, and population. Then the features followed with less influence on the predictions, in the order of households, total_rooms, and housing_median_age.

1. median_income: 0.6478797663330628
2. latitude: -0.4647311592885787
3. longitude: -0.4579543578585768
4. total_bedrooms: 0.3848627669893396
5. population: -0.3501098134268003
6. households: 0.1277073146615151
7. total_rooms: -0.12262560565532601
8. housing_median_age: 0.12042159589273575

ii. Discuss how the influence of the features (obtained in question e.1) relates to the pair-wise correlation results calculated above i.e., are the features that are highly correlated with the output also the most influential or is there some other phenomenon being observed?

The pairwise correlation results that are calculated above are as follows (and also in the table above) and listed in a descending order of magnitude to observe the influence. Both pair-wise correlation results and weights of the linear regression model showed median income as the most influential feature with similar coefficients (`0.688355 for pair-wise correlation and 0.647879 for linear regression coefficients`). However, for the remaining less-influential features, the coefficients obtained through each of the methods did not show significant correlation. For example, the order of coefficients after median_income is different. This result shows that although the linear regression coefficients show a general picture of the significant feature(s), it is not all comprehensive. Although there will be many reasons as to why this is true, one possible explanation could be because of the complex feature correlation that could not be well captured through a linear method.

```
median_income 0.688355
total_rooms 0.133294
housing_median_age 0.106432
households 0.064894
```

```
total_bedrooms 0.049686
population -0.025300
```

iii. Comment about the MAPE, RMSE, R^2 results. What can we learn from each of these results about the model prediction performance?

First of all, MAPE is the mean absolute percentage error. It gives us the average percentage difference between the prediction and the actual values. We obtained a MAPE value of about 0.287. In other words, the prediction value is 28.7% different from the actual value. In addition, RMSE also measures error, a root mean squared error. We obtained a RMSE of about 67927.64, which means that the predictions are off by about the RMSE value. Lastly the R-squared measure is the proportion of the variance in the y that is explained by the x. We obtained a value of about 0.654, which means the model is explaining 65.4% of variance in the y. Therefore, through the R-squared result, we can say that the model is doing a good job in explaining the variance in the house price, but since the error values such as MAPE and RMSE is not very low, we could try to further improve the model by tuning hyperparameters, etc.

iv. Why is centering and scaling the data important before performing PCA?

Centering and scaling the data is important before performing PCA because since PCA is based on variance of the data, it helps to remove bias in the mean of the data and ensure that the first principal component to be in the direction of highest variability and other components to be orthogonal for easier computation for covariance matrix. Scaling helps to give each feature equal importance so that very large numbers in a feature would not bias the variance calculations and be comparable across different features.