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

1. **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:
2. longitude (continuous): One of the coordinates that are used to identify the California district block
3. latitude (continuous): One of the coordinates that are used to identify the California district block
4. housing\_median\_age (continuous): Average age of the house in California district block
5. total\_rooms (continuous): Total number of rooms of all the houses in the California district block
6. total\_bedrooms (continuous): Total number of bedrooms of all the houses in the California district block
7. population (continuous): Number of people residing in the district block
8. households (continuous): Number of families in the district block
9. median\_income (continuous): Median income for households in the district block of houses (measured in tens of thousands of US Dollars)
10. ocean\_proximity (categorical): Location of the house. Is it inland, near the bay, near the ocean, etc.
11. 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*.

1. **Data Loading / Preprocessing:**
   1. Loading:
2. Load the California housing dataset using pandas.read\_csv() function and store it in the variable (i.e., a pandas dataframe) named `df’.
3. The resulting data frame should have the shape (20640, 10) indicating that there are 20640 rows and 10 columns.
4. 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()](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html) function. The resulting data frame should have the shape (20433, 10) indicating that there are 20433 rows and 10 columns.
5. Create a data frame ‘corr\_df’ by dropping the columns *latitude*, *longitude*, and *ocean\_proximity* using the [pandas.DataFrame.drop()](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html) 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().](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.corrwith.html) Report results in the following table.

|  |  |
| --- | --- |
| **Feature Name** | **Pearson Correlation** |
| housing\_median\_age |  |
| total\_rooms |  |
| total\_bedrooms |  |
| population |  |
| households |  |
| median\_income |  |

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()](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop.html) 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).

* 1. Data Visualization:

1. Use [pandas.DataFrame.hist(bins = 50)](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.hist.html) 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()](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.describe.html) function to find the mean, median and standard deviations for each feature and report in the jupyter notebook.
3. Use [pandas.get\_dummies](https://pandas.pydata.org/docs/reference/api/pandas.get_dummies.html) to convert categorical variables into dummy /one-hot encoding. In this case the categorical column is *ocean\_proximity*
   1. Data Splitting:
4. Split data into training and test sets using the sklearn [train\_test\_split()](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html) 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).

* 1. Data Scaling:

1. Use the [StandardScaler()](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html) 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()](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html)’ 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)](https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.hist.html) 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.
5. **Modelling:**
6. Employ Linear Regression from [sklearn.linear\_model](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html), and instantiate the model.
7. Once instantiated, `fit()` the model using the *scaled* X\_train, y\_train data.
8. 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).
9. Use [inverse\_transform](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html)() function to convert the normalized data (`y\_preds` ) to original scale. Store the transformed values back into `y\_preds`.
10. Perform [PCA](https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html) 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\_](https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html). Also, report the strength of each PCA component using *[pca.singular\_values\_](https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html)*.
11. **Evaluation:**
    1. Plot a scatter plot using [matplotlib.pyplot.scatter](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.scatter.html) function. Plot the predicted median house values on the y-axis vs the actual median house values on the x-axis.
    2. Calculate [MAPE](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_percentage_error.html), [RMSE](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html) and [R](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html)[2](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html) for the model and report them in the following table.   
       **Hint** for RMSE set the *squared* parameter to False.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | MAPE | RMSE | R2 |
| Linear Regression |  |  |  |

1. **Discussion:**
   1. Based on the weights of the linear regression model, rank the features (note: only continuous features) in decreasing order of influence on the predictions.
   2. 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?
   3. Comment about the MAPE, RMSE, R2 results. What can we learn from each of these results about the model prediction performance?
   4. Why is centering and scaling the data important before performing PCA?