ΣΕΠΤ.
2022

# ΑΝΑΓΝΩΡΙΣΗ ΠΡΟΤΥΠΩΝ

ΕΥΣΤΡΑΤΙΟΣ ΚΑΡΚΑΝΗΣ – Π19064
ΧΑΡΑΛΑΜΠΟΣ ΧΡΙΣΤΟΦΟΡΙΔΗΣ – Π19188
ΓΕΩΡΓΙΟΣ ΣΕΪΜΕΝΗΣ – Π19204

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

# Εισαγωγή

Η ανάπτυξη της εφαρμογής έγινε με τη γλώσσα Python, με το interface του Jupyter, χρησιμοποιώντας το Visual Studio Code και διάφορες χρήσιμες βιβλιοθήκες, που βοήθησαν πολύ με την οπτικοποίηση και την παλινδρόμηση των δεδομένων. Παρακάτω θα εξηγηθούν αναλυτικά οι χρήσεις των βιβλιοθηκών και η σημασία τους στη διευκόλυνση πολλών διαδικασιών. Συγκεκριμένα, οι βιβλιοθήκες που χρησιμοποιούνται είναι οι:

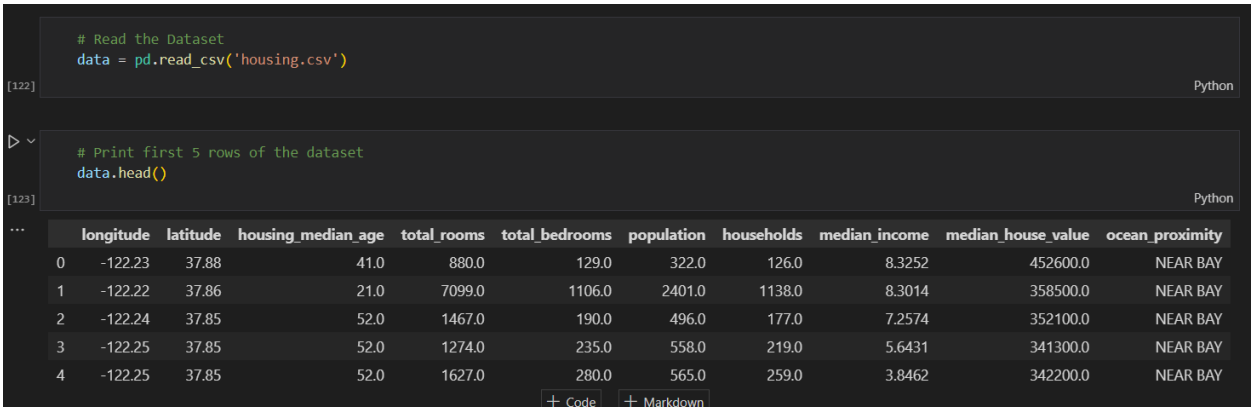
- numpy
- pandas
- matplotlib
- seaborn
- sklearn

Όλες οι παραπάνω είναι απαραίτητες για την εκτέλεση της εφαρμογής.

# Προ-επεξεργασία Δεδομένων

## ΜΕΡΟΣ 1ο: Αναγνώριση της Πληροφορίας

Το αρχείο CSV, που περιέχει τα δεδομένα, διαβάστηκε με τη χρήση της βιβλιοθήκης pandas, ώστε να έχουμε πλήρη εικόνα των μετρήσεων των δεδομένων. Όπως φαίνεται και παρακάτω, τα δεδομένα που έχουμε είναι 8 (οκτώ) στήλες τύπου αριθμού κινητής υποδιαστολής και μίας στήλης τύπου αλφαριθμητικού.



```
# Read the Dataset
data = pd.read_csv('housing.csv')
```

[122] Python

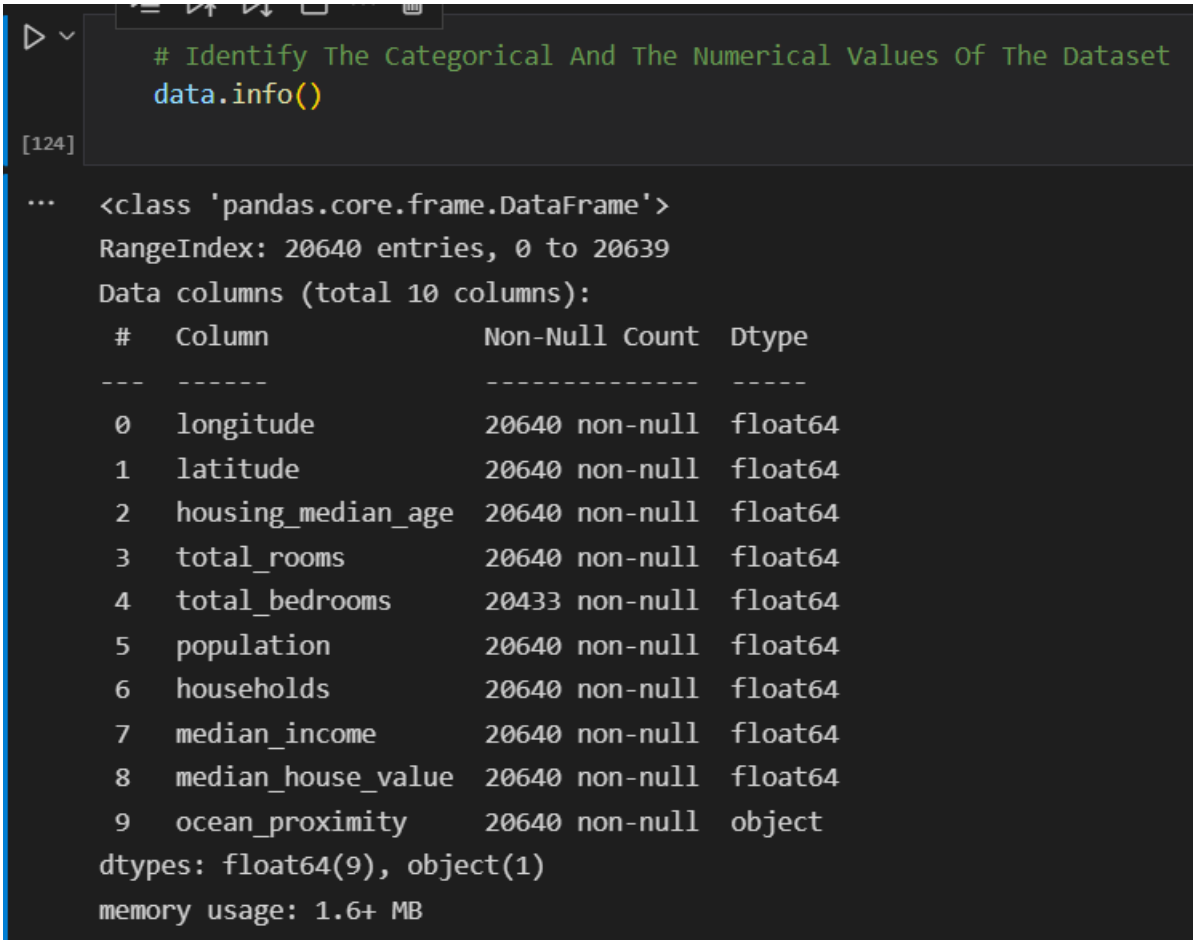
```
# Print first 5 rows of the dataset
data.head()
```

[123] Python

| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms | population | households | median_income | median_house_value | ocean_proximity |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | 322.0 | 126.0 | 8.3252 | 452600.0 | NEAR BAY |
| 1 | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | 2401.0 | 1138.0 | 8.3014 | 358500.0 | NEAR BAY |
| 2 | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | 496.0 | 177.0 | 7.2574 | 352100.0 | NEAR BAY |
| 3 | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | 558.0 | 219.0 | 5.6431 | 341300.0 | NEAR BAY |
| 4 | -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | 565.0 | 259.0 | 3.8462 | 342200.0 | NEAR BAY |

+ Code + Markdown

**Εικόνα 1: Τα πρώτα 5 (πέντε) δεδομένα του αρχείου CSV.**



```
# Identify The Categorical And The Numerical Values Of The Dataset
data.info()
```

[124]

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object 
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

**Εικόνα 2: Αναγνώριση των τύπων δεδομένων των στηλών.**

## Μέρος 2ο: Κλιμάκωση Δεδομένων

Εφ' όσον έχουμε οκτώ στήλες στις οποίες εμπεριέχονται αριθμοί, θα πρέπει να κάνουμε κλιμάκωση. Δηλαδή, θα πρέπει όλες οι τιμές να ανήκουν στο ίδιο διάστημα. Όσο για την άλλη στήλη, που δεν έχει από μόνη της αριθμητικά δεδομένα, θα ακολουθήσουμε άλλη μεθοδολογία, ώστε να έχουμε αριθμητικές τιμές. Παρατηρούμε ότι έχει 5 (πέντε) διαφορετικές τιμές:

- <1H OCEAN
- INLAND
- ISLAND
- NEAR BAY
- NEAR OCEAN

Άρα, θα διαγράψουμε τη στήλη που έχει αλφαριθμητικές τιμές και αντ' αυτής, θα βάλουμε πέντε επιπλέον στήλες. Αυτές οι πέντε στήλες θα αντιστοιχούν στα παραπάνω ονόματα.

Για αρχή, κλιμακώνουμε τα δεδομένα μας, ώστε όλα να ανήκουν στο σύνολο [-3, 3].

```
# Normalize the dataset, in order for all data to be in the same scale

sc = StandardScaler() # instantiate a scaler's object

# Column "ocean_proximity" is not scaled because it contains categorical values
data_scaled = sc.fit_transform(data.drop("ocean_proximity",axis=1))



# print the scaled dataset
data_scaled = pd.DataFrame(data=data_scaled,columns=data.columns[0:-1])
data_scaled.head()
```

**Εικόνα 3: Κλιμάκωση των στηλών με τις αριθμητικές τιμές.**

Ύστερα, «μαζεύουμε» τα δεδομένα της στήλης στην οποία εμπεριέχονται τα αλφαριθμητικά και τα κάνουμε όλα πέντε ξεχωριστές στήλες.

```
# One - Hot encoding of the only categorical attribute 'ocean_proximity'
vector = pd.get_dummies(data['ocean_proximity'])



# Adding the new columns to the dataset
data = pd.concat([data_scaled,vector], axis=1)
```

**Εικόνα 4: Συμμάζεμα των δεδομένων της στήλης με τα αλφαριθμητικά.**

Οπότε, η τελική μορφή του πίνακα, μετά την κλιμάκωση των δεδομένων, είναι η εξής:

```
# print new dataset (which is scaled and one-hot encoded)
data.head()
```

Python

| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms | population | households | median_income | median_house_value | <1H OCEAN | INLAND | ISLAND | NEAR BAY | NEAR OCEAN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | -1.327835 | 1.052548 | 0.982143 | -0.804819 | -0.970325 | -0.974429 | -0.977033 | 2.344766 | 2.129631 | 0 | 0 | 0 | 1 | 0 |
| 1 | -1.322844 | 1.043185 | -0.607019 | 2.045890 | 1.348276 | 0.861439 | 1.669961 | 2.332238 | 1.314156 | 0 | 0 | 0 | 1 | 0 |
| 2 | -1.332827 | 1.038503 | 1.856182 | -0.535746 | -0.825561 | -0.820777 | -0.843637 | 1.782699 | 1.258693 | 0 | 0 | 0 | 1 | 0 |
| 3 | -1.337818 | 1.038503 | 1.856182 | -0.624215 | -0.718768 | -0.766028 | -0.733781 | 0.932968 | 1.165100 | 0 | 0 | 0 | 1 | 0 |
| 4 | -1.337818 | 1.038503 | 1.856182 | -0.462404 | -0.611974 | -0.759847 | -0.629157 | -0.012881 | 1.172900 | 0 | 0 | 0 | 1 | 0 |

**Εικόνα 5: Τα δεδομένα του πίνακα μετά την κλιμάκωση.**

## Μέρος 3ο: Καθαρισμός των Δεδομένων

Στο τελικό στάδιο της προ-επεξεργασίας δεδομένων, αποφασίσαμε να ασχοληθούμε και με τις τιμές που είτε **δεν υπήρχαν** (NaN τιμές σε κάποιο πεδίο εγγραφών), είτε ήταν σε κάποια **ακραία μεγάλη ή μικρή τιμή** (outliers). Αναλυτικότερα:

- Για τις κενές τιμές, **αποφασίσαμε να αντικαταστήσουμε την NaN τιμή με την μέση τιμή (median value)** της εκάστοτε στήλης. Ο λόγος που δεν επιλέξαμε να διαγράψουμε ολόκληρη τη στήλη, ήταν επειδή υπήρχε ο κίνδυνος μεγάλης απώλειας πληροφορίας, συνεπώς οδηγούσε στην πιθανή αλλοίωση των αποτελεσμάτων μας.

```
# Subtitute the NaN values with the median-value of the column (in our case only for the column "total_bedrooms")
data.fillna(data.median(), inplace=True)
```

Python

**Εικόνα 6: Συμπλήρωση τιμών στο πεδίο "total_bedrooms", το οποίο είχε NaN τιμές.**

- Για τις **ακραίες τιμές** που συναντήσαμε (outliers), **αποφασίσαμε να διαγράψουμε οριστικά τις γραμμές στις οποίες βρίσκονταν**. Σε αυτή την περίπτωση, δεν υπήρχε ο κίνδυνος μεγάλης απώλειας, καθώς τέτοιου είδους «ανωμαλίες» είναι εξαιρετικά σπάνιες σε ένα σύνολο δεδομένων (σε εμάς ήταν κάπου στις 300 εγγραφές). Για την διαδικασία εξάλειψης χρησιμοποιήσαμε την μέθοδο IQR.

```
# Delete outliers from the dataset using the IQR method
for x in data.columns[0:9]:
    q75,q25 = np.percentile(data.loc[:,x],[75,25])
    intr_qr = q75-q25

    max = q75+(1.5*intr_qr)
    min = q25-(1.5*intr_qr)

    data.loc[data[x] < min,x] = np.nan
    data.loc[data[x] > max,x] = np.nan


# Delete null values
data.dropna(inplace=True)
```

**Εικόνα 7: Μέθοδος IQR για την εξάλειψη ακραίων τιμών.**

- Τέλος, για να αποφύγουμε τυχόν προβλήματα με τους δείκτες (index) του dataframe, λόγω των πολλών διαγραφών, κάναμε re-index.

```
# Reindex the dataframe after deletion of rows
data.reset_index(drop=True, inplace=True)
```

**Εικόνα 8: Μέθοδος Re-Index μετά την διαγραφή των δεδομένων.**

# Οπτικοποίηση Δεδομένων

## Μέρος 1ο: Ιστογράμματα Συχνοτήτων

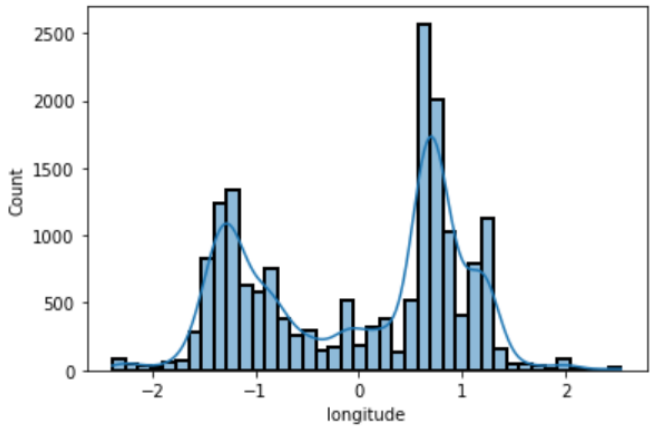
Στο πρώτο κομμάτι της οπτικοποίησης, κληθήκαμε να οπτικοποιήσουμε τα **ιστογράμματα συχνοτήτων** για κάθε μία από τις στήλες/μεταβλητές μας. Χρησιμοποιήσαμε την βιβλιοθήκη *seaborn* και *matplotlib* της Python, που έχουν ειδική συνάρτηση δημιουργίας ιστογραμμάτων.

```
# Histogram of column 'longitude'
sns.histplot(data[data.columns[0]],bins=40,kde=True,lw=2)
```

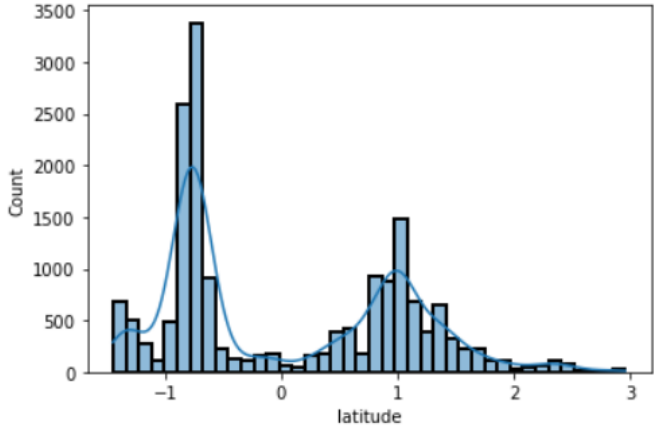
**Εικόνα 9: Παράδειγμα για το πώς παράγουμε ένα ιστόγραμμα με την βιβλιοθήκη.**

Τα ιστογράμματα που παρήχθησαν είναι τα εξής:

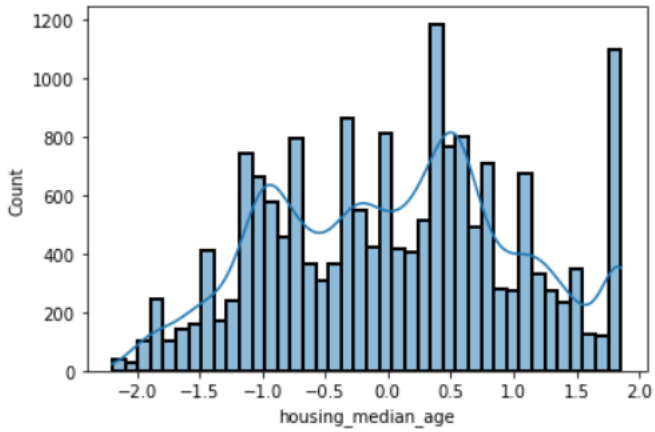
LONGITUDE:



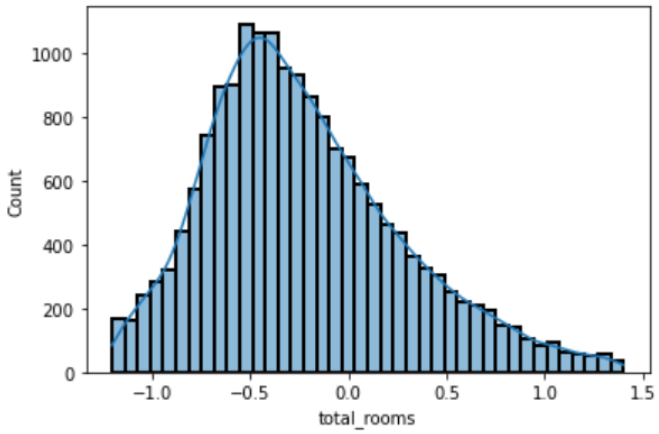

LATITUDE:



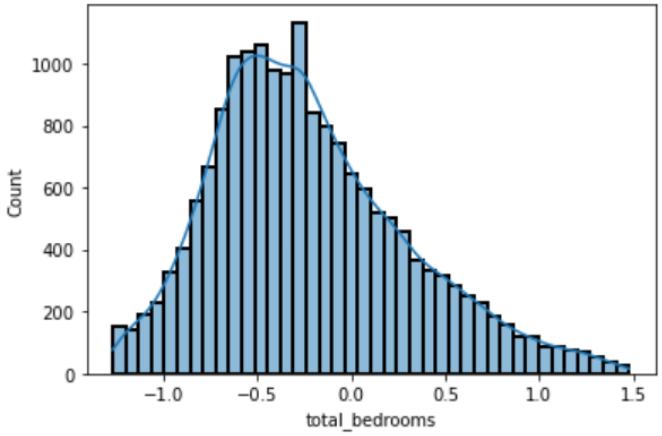
HOUSING MEDIAN AGE:



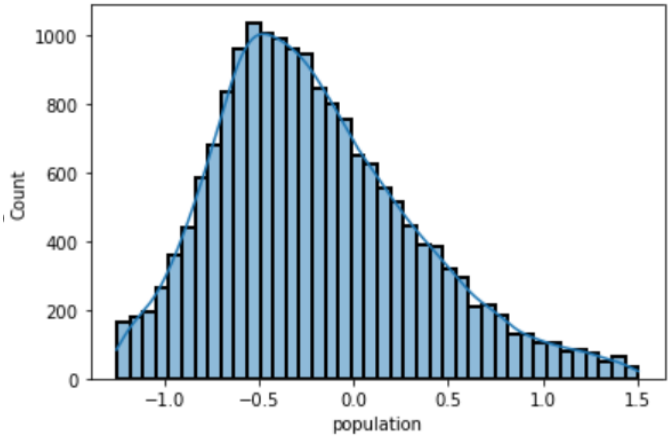

TOTAL ROOMS:



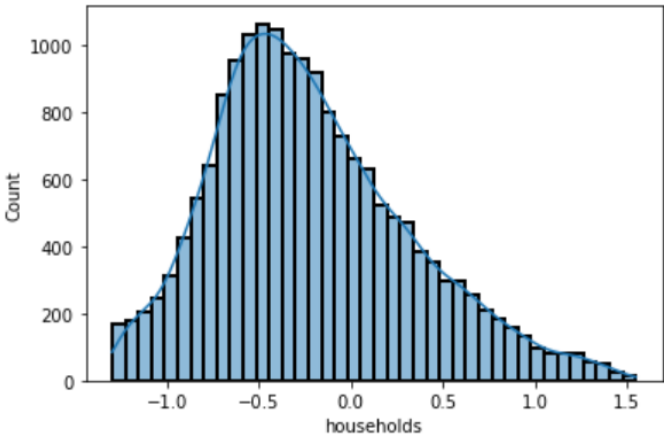

TOTAL BEDROOMS:



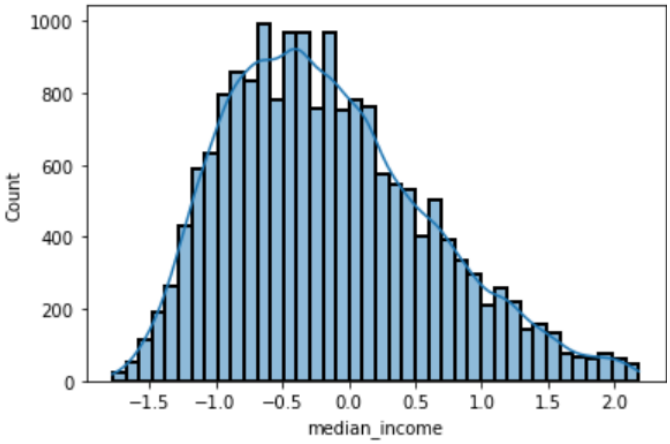
POPULATION:



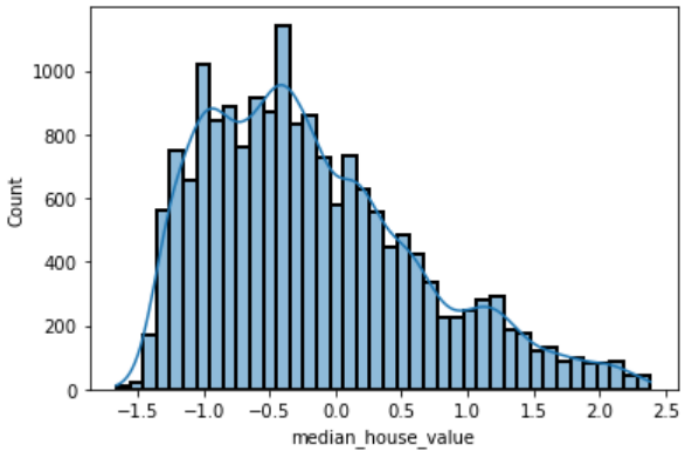

HOUSEHOLDS:



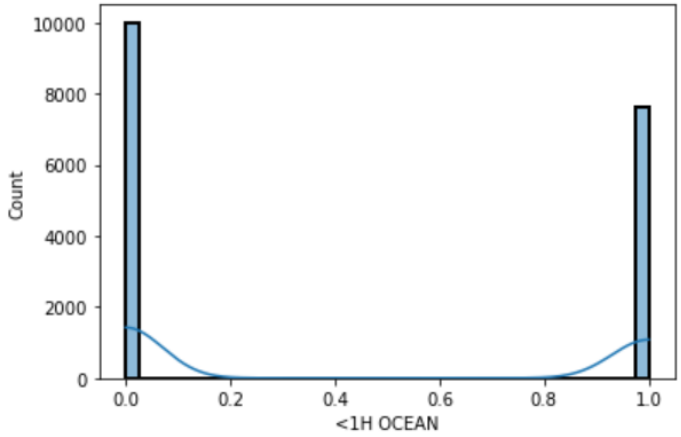

MEDIAN INCOME:



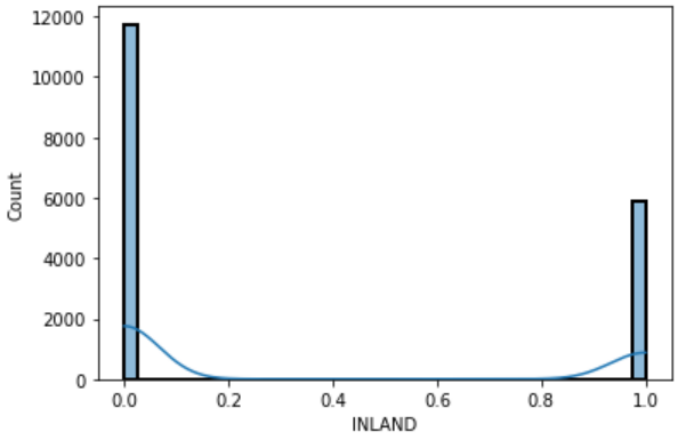
MEDIAN HOUSE VALUE:



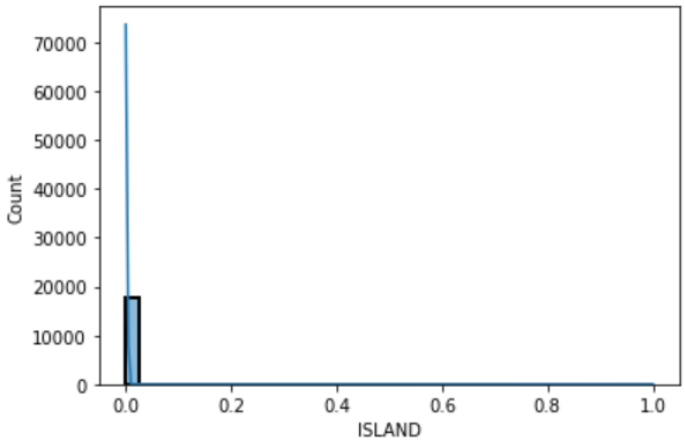

<1H OCEAN:



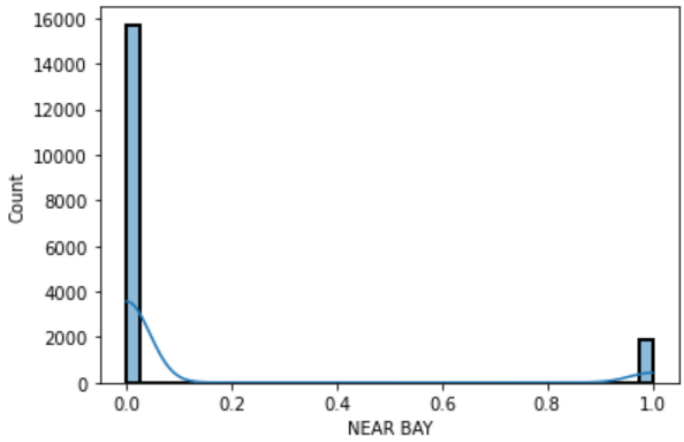

INLAND:



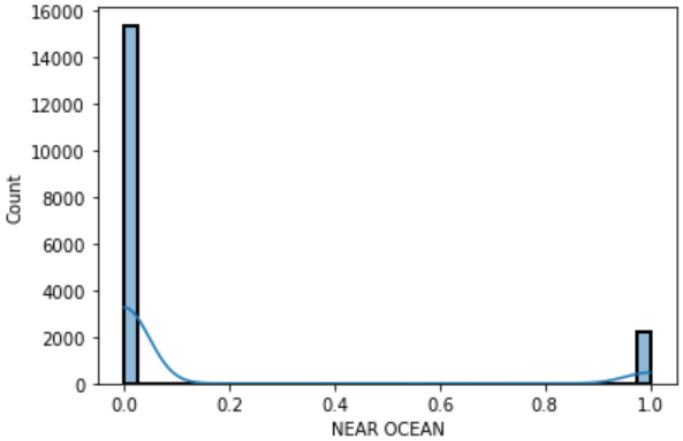
ISLAND:




NEAR BAY:

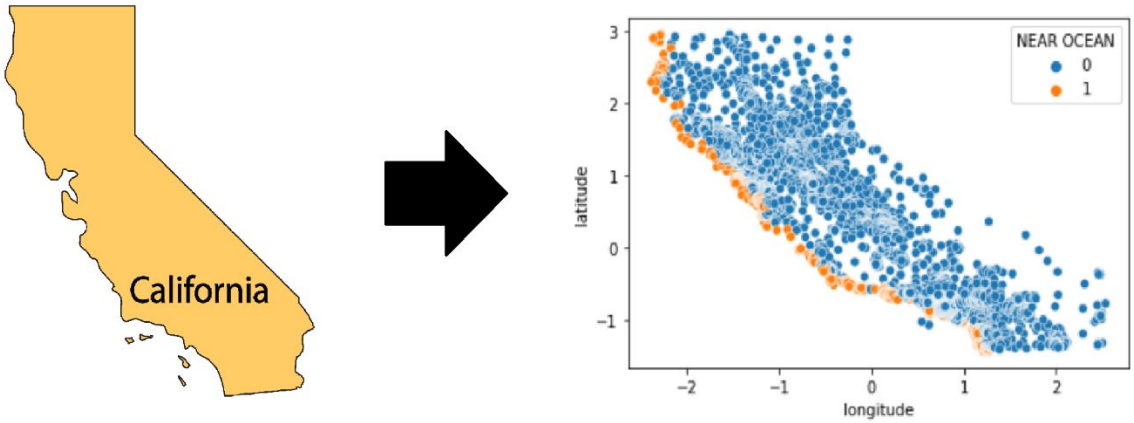



NEAR OCEAN:



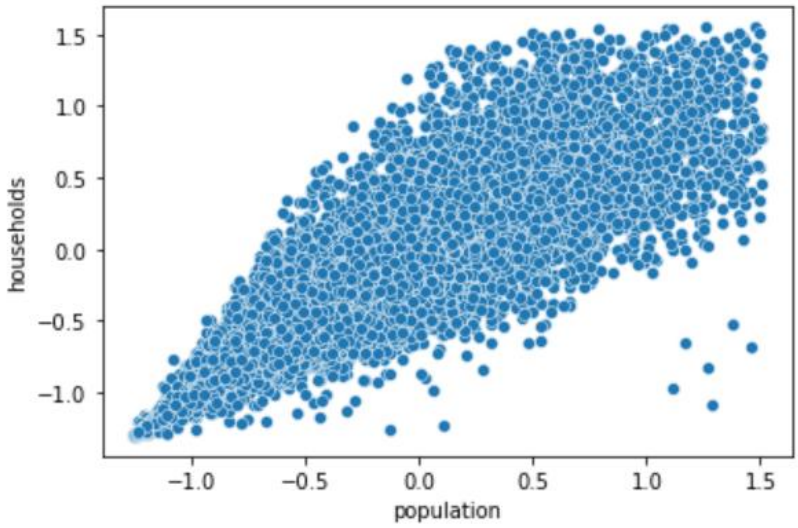
## Μέρος 2ο: Δισδιάστατα Γραφήματα

Ιδιαίτερο ενδιαφέρον προσφέρει το γράφημα του **γεωγραφικού πλάτους (latitude) συναρτήσει του γεωγραφικού μήκους (longitude).** Το συγκεκριμένο γράφημα θυμίζει πολύ τον χάρτη της Καλιφόρνια. Για να επιβεβαιωθεί αυτό, μπορούμε να βάλουμε ως **hue** την στήλη «NEAR OCEAN». Με αυτόν τον τρόπο, με το μπλε χρώμα θα αναπαρασταθούν τα σπίτια που δεν είναι κοντά στον ωκεανό, σε αντίθεση με τα πορτοκαλί, τα οποία είναι κοντά στον ωκεανό.



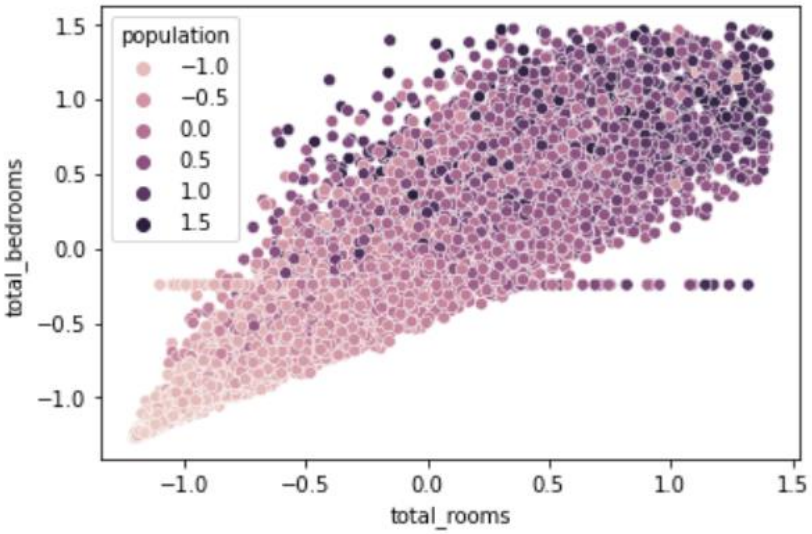

**Εικόνα 10: Γεωγραφικό Πλάτος συναρτήσει του Γεωγραφικού Μήκους.**

Επιπλέον, με τα παρακάτω γραφήματα, **μπορούμε να δούμε το τι γίνεται με την αύξηση του πληθυσμού στα νοικοκυριά**. Συγκεκριμένα, ένα γεγονός, το οποίο βγάζει νόημα, **είναι η αναλογική αύξηση των νοικοκυριών με τον πληθυσμό** της εκάστοτε περιοχής. Όπως φαίνεται και στο επόμενο γράφημα, η παραπάνω πρόταση επιβεβαιώνεται.




**Εικόνα 11: Νοικοκυριά συναρτήσει του Πληθυσμού.**

Παρόμοιο, μπορούμε να δούμε ότι γίνεται με τα **συνολικά δωμάτια και με τα υπνοδωμάτια σε κάθε σπίτι**. Δηλαδή, **όσο αυξάνονται τα συνολικά δωμάτια ενός νοικοκυριού, αυξάνονται και τα υπνοδωμάτια**. Βέβαια, μπορούμε να παρατηρήσουμε, παράλληλα, το γεγονός ότι **κάποια νοικοκυριά έχουν σταθερό αριθμό υπνοδωματίων, όσο κι αν αυξηθούν τα συνολικά δωμάτια.** Για να μπλέξουμε και μία τρίτη μεταβλητή στο γράφημα, όπως και στο πρώτο, θα βάλουμε ως **hue** τον πληθυσμό.




**Εικόνα 12: Συνολικά Υπνοδωμάτια συναρτήσει των Συνολικών Υπνοδωματίων.**

# Παλινδρόμηση Δεδομένων

## Μέρος 2ο: Αλγόριθμο Ελάχιστου Τετραγωνικού Σφάλματος (Least Squares)

Σε αυτό το κομμάτι της εργασίας χρησιμοποιήσαμε την βιβλιοθήκη *LinearRegression* της Python. Αυτή η βιβλιοθήκη **εκτελούσε αυτόματα τη διαδικασία εύρεσης ευθείας γραμμής,** η οποία απέχει την μικρότερη δυνατή απόσταση από τα σημεία των δεδομένων εκπαίδευσης (δηλαδή τα σφάλματα ελαχιστοποιούνται). **Η ρουτίνα αυτή εκτελείται 10 φορές συνολικά,** όπως μας υποδεικνύεται από την μέθοδο της 10-πλής διεπικύρωσης (10 fold cross validation). Αναλυτικότερα, σε κάθε επανάληψη δημιουργούσαμε ένα καινούριο μοντέλο, το οποίο και εκπαιδεύαμε πάνω στα δεδομένα εκπαίδευσης, και ύστερα ελέγχαμε τις προβλέψεις μας πάνω στα δεδομένα ελέγχου. Τα τελικά σφάλματα που προέκυψαν δεν ήταν ιδιαίτερα ικανοποιητικά, παρ' όλα αυτά ήταν τα ελάχιστα δυνατά.

```
Average Mean Absolute Error on training 0.3718958448829153
Average Mean Absolute Error on training 0.24918398096423905


Average Mean Absolute Error on testing 0.372261243711287 55
Average Mean Absolute Error on testing 0.24976751457919652
```

**Εικόνα 13: Το αποτέλεσμα του Ελάχιστου Τετραγωνικού Σφάλματος.**

## Μέρος 3ο: Πολυστρωματικό Νευρωνικό Δίκτυο

Σε αυτό το κομμάτι της εργασίας χρησιμοποιήσαμε τις βιβλιοθήκες Sequential και Dense της Tensorflow/Keras. Οι συγκεκριμένες βιβλιοθήκες μας επέτρεψαν να φτιάξουμε το μοντέλο του νευρωνικού δικτύου μας. Η ρουτίνα αυτή εκτελείται 10 φορές συνολικά όπως μας υποδεικνύεται από την μέθοδο της 10-πλής διεπικύρωσης

(10 fold cross validation). Μέσα στην επανάληψη κατασκευάζουμε **δύο διαφορετικά νευρωνικά δίκτυα.** Το ένα υπολογίζει το μέσο απόλυτο σφάλμα και το άλλο υπολογίζει το μέσο τετραγωνικό σφάλμα, τόσο κατά το στάδιο της εκπαίδευσης όσο και κατά το στάδιο του ελέγχου. Το πρόβλημα, που μας δημιουργήθηκε κατά τη διάρκεια του ερωτήματο,ς είναι οι μεγάλοι χρόνοι που χρειάζονταν για να τρέξουν τα μοντέλα μας. Τέλος, τα σφάλματα κατά τη διάρκεια της εκπαίδευσης φαίνονται με την παράμετρο *"verbose = 1"* στην συνάρτηση *fit()*, ενώ τα σφάλματα κατά το στάδιο του ελέγχου εμφανίζονται στο τέλος.

```
Epoch 1/25
634/634 [==============================] - 3s 3ms/step - loss: 0.3560
Epoch 2/25
634/634 [==============================] - 2s 4ms/step - loss: 0.3205
Epoch 3/25
634/634 [==============================] - 2s 3ms/step - loss: 0.3111
Epoch 4/25
634/634 [==============================] - 3s 4ms/step - loss: 0.3037
```

**Εικόνα 14: Ενδεικτικό παράδειγμα του πολυστρωματικού νευρωνικού δικτύου.**

# Βιβλιογραφικές Πηγές

Οι παρακάτω πηγές βοήθησαν στην συγγραφή και ανάπτυξη του κώδικα της εργασίας. Αναλυτικότερα, φαίνονται παρακάτω με σχετική αναφορά στους υπερσυνδέσμους τους:


1. Data Preprocessing in Data Mining -A Hands On Guide - Analytics Vidhya.
2. (73) Outlier detection and removal using IQR | Feature engineering tutorial python #4 - YouTube.
3. Ελάχιστο Τετραγωνικό Σφάλμα.
4. Κανόνας της κ-πλής επικύρωσης.
5. Least Squares Linear Regression — ML From Scratch (Part 1).
6. How to use k-fold Validation with Keras.