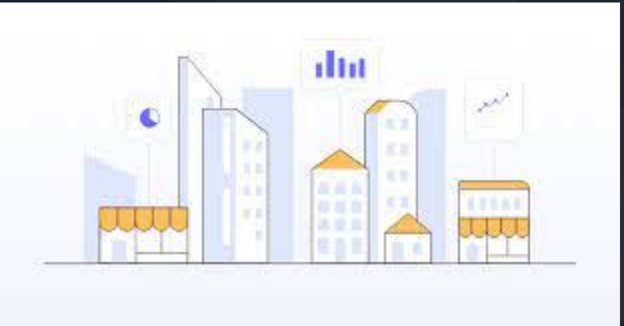
# Housing Price Prediction model



Machine Learning with Random Forest

By: Yug Sharma

# Introduction

**Objective**: To predict Housing prices from a synthetic dataset unique to this project

**Target**: Achieve a regression model with an R^2 score of at least 80.

**Methodology:** Machine Learning using Random Forest

# Data Overview

**Source of Data:** Synthetic Dataset

**Key Features:** Area, Bedrooms, Bathrooms etc.

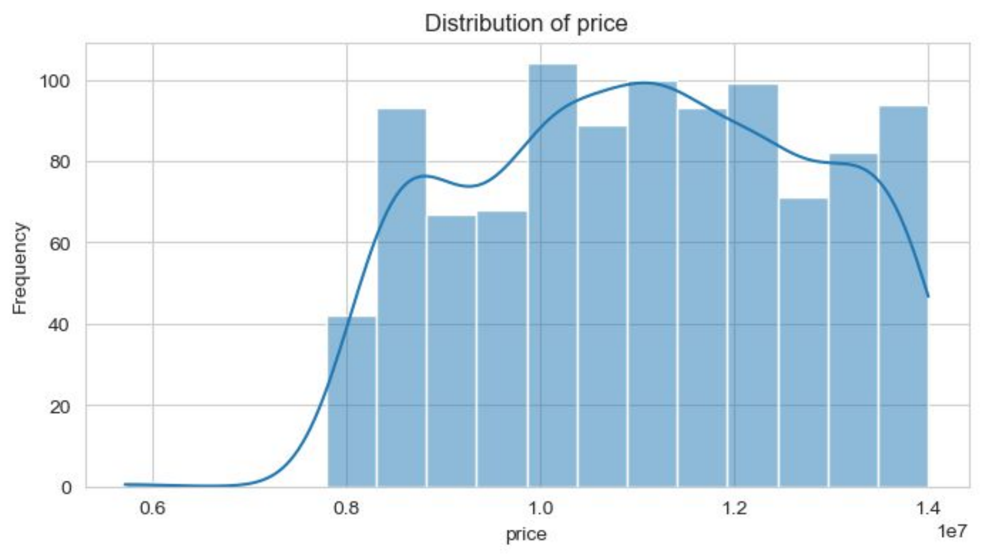
Out[3]:

| | price | area | bedrooms | bathrooms | stories | parking |
|---|---|---|---|---|---|---|
| **count** | 9.910000e+02 | 988.000000 | 990.000000 | 985.000000 | 983.000000 | 1003.000000 |
| **mean** | 1.105174e+07 | 8957.272267 | 3.978788 | 2.608122 | 2.423194 | 1.448654 |
| **std** | 1.742147e+06 | 32301.033484 | 0.816014 | 4.030213 | 1.114591 | 1.214560 |
| **min** | -1.120000e+02 | -998877.000000 | 3.000000 | 1.000000 | 1.000000 | -12.000000 |
| **25%** | 9.709294e+06 | 6776.250000 | 3.000000 | 2.000000 | 1.000000 | 0.000000 |
| **50%** | 1.106197e+07 | 10077.500000 | 4.000000 | 2.000000 | 2.000000 | 1.000000 |
| **75%** | 1.243362e+07 | 13006.250000 | 5.000000 | 4.000000 | 3.000000 | 2.000000 |
| **max** | 1.399908e+07 | 16196.000000 | 5.000000 | 124.000000 | 4.000000 | 3.000000 |

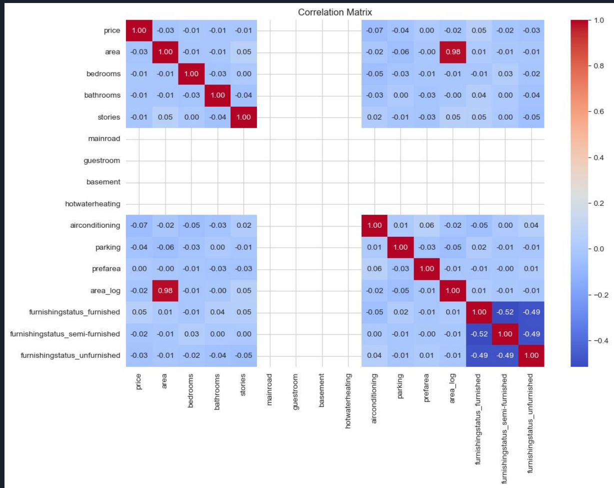
| price | area | bedrooms | bathrooms | stories | mainroad | guestroom | basement | hotwaterhea | airconditioni | parking | prefarea | furnishingstatus |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 11991472 | 14139 | 3 | 1 | 4 | FALSE | TRUE | FALSE | FALSE | FALSE | 1 | TRUE | unfurnished |
| 11991472 | 14139 | 3 | 1 | 4 | FALSE | TRUE | FALSE | FALSE | FALSE | 1 | TRUE | unfurnished |
| 8632694 | 4182 | 3 | 3 | 1 | TRUE | FALSE | FALSE | TRUE | TRUE | 3 | FALSE | unfurnished |
| 8481911 | 10645 | 4 | 4 | | FALSE | TRUE | TRUE | FALSE | FALSE | 0 | FALSE | semi-furnished |
| 13556045 | 6415 | 4 | 2 | 1 | FALSE | TRUE | FALSE | TRUE | FALSE | 0 | TRUE | furnished |
| 9925983 | 9504 | 3 | 2 | 2 | FALSE | TRUE | FALSE | FALSE | TRUE | 1 | FALSE | furnished |
| 9101598 | 5752 | 3 | 1 | 1 | FALSE | FALSE | TRUE | FALSE | TRUE | 1 | TRUE | semi-furnished |
| 9979630 | 7420 | 3 | 3 | 1 | FALSE | FALSE | TRUE | TRUE | FALSE | 0 | TRUE | furnished |
| 9979630 | 7420 | 3 | 3 | 1 | FALSE | FALSE | TRUE | TRUE | FALSE | 0 | TRUE | furnished |
| 10179634 | 7400 | 4 | 1 | 1 | TRUE | FALSE | FALSE | TRUE | FALSE | 0 | TRUE | furnished |
| 10161840 | 8918 | 4 | 2 | 4 | FALSE | TRUE | TRUE | FALSE | TRUE | 0 | FALSE | unfurnished |
| 12045879 | 15243 | 4 | 4 | 1 | FALSE | TRUE | FALSE | FALSE | TRUE | 3 | FALSE | unfurnished |
| -112 | 8687 | 3 | 4 | 1 | TRUE | TRUE | FALSE | TRUE | TRUE | 3 | FALSE | semi-furnished |
| 11420591 | 12697 | 3 | 3 | 3 | TRUE | TRUE | TRUE | TRUE | TRUE | 1 | FALSE | unfurnished |
| 12047335 | 5109 | 3 | 3 | 3 | FALSE | FALSE | FALSE | TRUE | TRUE | 2 | FALSE | unfurnished |
| 8277674 | 13807 | 3 | 2 | 4 | FALSE | FALSE | TRUE | TRUE | TRUE | 3 | FALSE | furnished |
| 8666997 | 9556 | 3 | 1 | 4 | TRUE | FALSE | TRUE | FALSE | FALSE | 3 | TRUE | unfurnished |
| 12352573 | 6792 | 5 | 4 | 4 | TRUE | FALSE | FALSE | TRUE | FALSE | 0 | FALSE | furnished |
| 10357357 | 5033 | 4 | 3 | 3 | FALSE | TRUE | TRUE | FALSE | FALSE | 0 | FALSE | unfurnished |
| 12284409 | 5175 | 4 | 1 | 2 | FALSE | TRUE | TRUE | TRUE | TRUE | 3 | FALSE | unfurnished |
| 13023367 | 6476 | 5 | 1 | 2 | TRUE | FALSE | TRUE | FALSE | FALSE | 1 | TRUE | semi-furnished |
| 10013209 | 5745 | 4 | 2 | 1 | TRUE | TRUE | TRUE | FALSE | FALSE | 1 | FALSE | unfurnished |
| 10741460 | 9204 | 5 | 4 | 2 | TRUE | TRUE | TRUE | TRUE | FALSE | 0 | TRUE | furnished |
| | 13148 | 4 | 3 | 2 | FALSE | TRUE | FALSE | FALSE | FALSE | 1 | FALSE | unfurnished |
| 11443414 | 10566 | 5 | 2 | 2 | FALSE | FALSE | TRUE | TRUE | TRUE | 3 | TRUE | semi-furnished |
| 10251397 | 12243 | 3 | 2 | | TRUE | TRUE | FALSE | FALSE | TRUE | 1 | TRUE | semi-furnished |
| 8257015 | 16112 | 3 | 4 | 3 | FALSE | TRUE | TRUE | FALSE | TRUE | 1 | TRUE | semi-furnished |
| 8646789 | 5370 | 3 | 4 | 3 | TRUE | FALSE | TRUE | TRUE | TRUE | 1 | FALSE | furnished |
| 8805447 | 8206 | 3 | 3 | 2 | TRUE | FALSE | FALSE | TRUE | FALSE | 0 | TRUE | semi-furnished |
| 10552257 | 7270 | 5 | 4 | 3 | FALSE | TRUE | FALSE | TRUE | FALSE | 1 | FALSE | furnished |
| 12507071 | 14748 | 4 | 1 | 3 | FALSE | TRUE | TRUE | FALSE | TRUE | 1 | TRUE | furnished |
| 10056972 | 5399 | 4 | 4 | 2 | TRUE | TRUE | FALSE | TRUE | TRUE | 1 | TRUE | unfurnished |
| 12904621 | -998877 | 4 | 2 | 3 | FALSE | TRUE | TRUE | TRUE | TRUE | 0 | FALSE | furnished |
| 13353346 | 4146 | 5 | 1 | 3 | TRUE | FALSE | FALSE | TRUE | FALSE | 1 | FALSE | semi-furnished |
| 13501570 | 5429 | 3 | 3 | 3 | FALSE | TRUE | TRUE | TRUE | FALSE | 1 | FALSE | semi-furnished |
| 13986836 | 7732 | | 3 | 1 | FALSE | FALSE | FALSE | TRUE | TRUE | 3 | FALSE | semi-furnished |

# Exploratory Data Analysis (EDA)

Libraries used: Pandas, Matplotlib

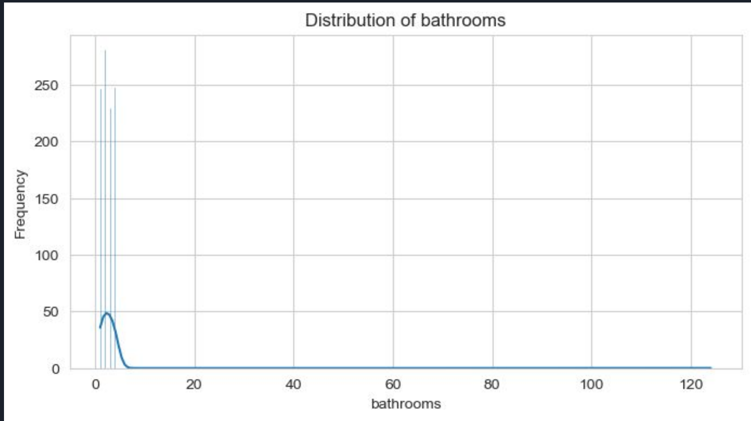


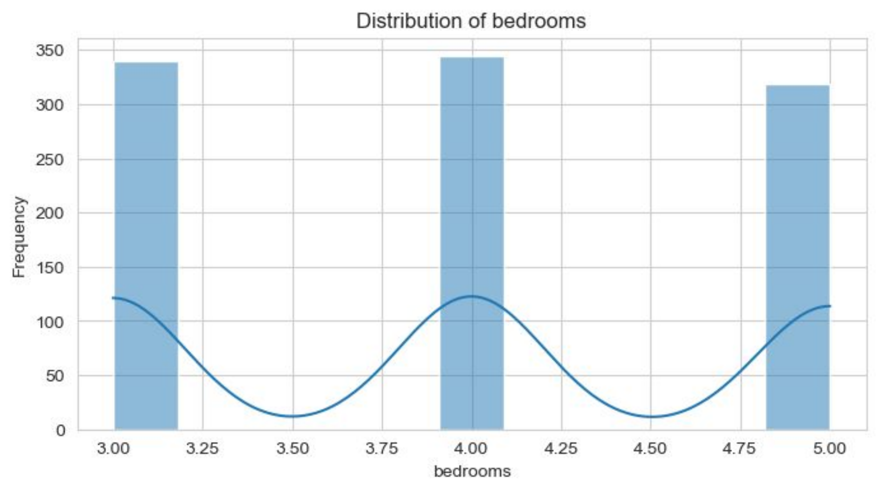


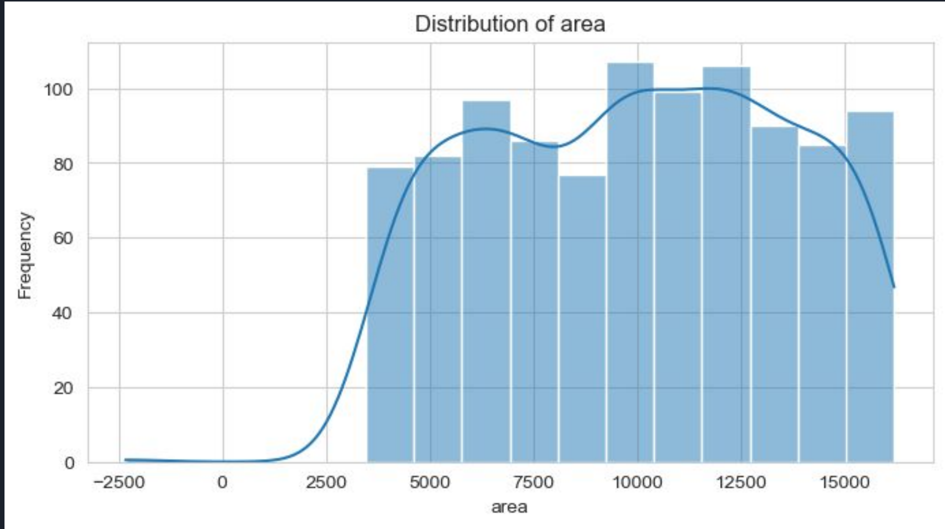


# Data Cleaning and Preprocessing

- Cleaned through Pandas in python



# Model Building and Selection

```
#Perform cross-validated feature importance evaluation
best_rf_model = RandomForestRegressor(**best_params)
cv_feature_importances = []
for train_idx, test_idx in KFold(n_splits=5).split(X_train):
    fold_X_train, fold_X_test = X_train.iloc[train_idx], X_train.iloc[te
    fold_y_train, fold_y_test = y_train.iloc[train_idx], y_train.iloc[te
    best_rf_model.fit(fold_X_train, fold_y_train)
    cv_feature_importances.append(best_rf_model.feature_importances_)

# Average feature importances across folds
avg_feature_importances = np.mean(cv_feature_importances, axis=0)
```

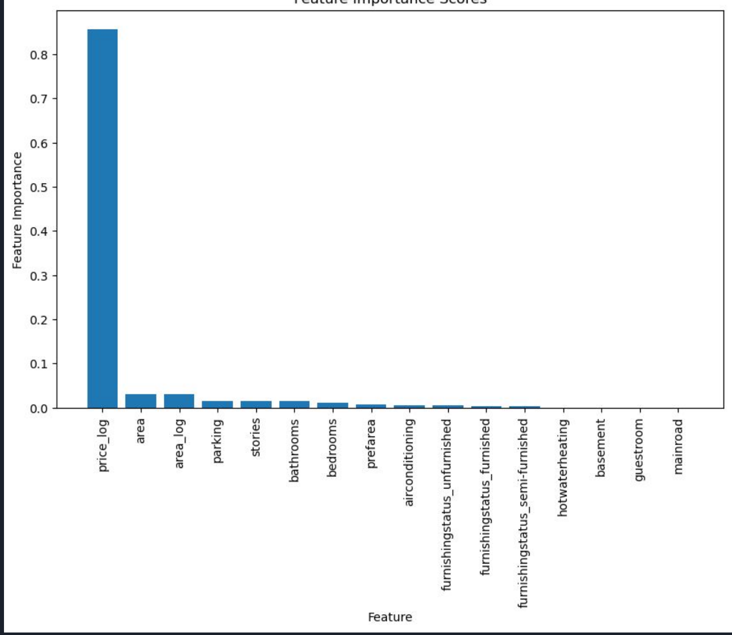
# Hyperparameter Tuning

```
#Define a hyperparameter distribution and initialize RandomizedSearchCV
param_dist = {
    'n_estimators': range(100, 601, 100),
    'max_depth': [None] + list(range(10, 31, 10)),
    'min_samples_split': range(2, 11, 2),
    'min_samples_leaf': range(1, 11, 2),
    'max_features': ['sqrt', 'log2']  # Correct options
}

random_search = RandomizedSearchCV(
    estimator=RandomForestRegressor(random_state=42),
    param_distributions=param_dist,
    n_iter=10,
    cv=3,
    scoring='r2',
    random_state=42,
    n_jobs=-1
)

# fit random search to data
random_search.fit(X_train, y_train)
best_params = random_search.best_params_
print("Best Hyperparameters:", best_params)
```

```
Best Hyperparameters: {'n_estimators': 500, 'min_samples_split': 6, 'mi
n_samples_leaf': 1, 'max_features': 'log2', 'max_depth': 10}
```

# Model Evaluation

**Final Results**: R^2 of ~90%




```
In [10]: # Instead of selecting the top features, use all features for training a
X_train_all_features = X_train
X_test_all_features = X_test

# Retrain the model on all features with cross-validation
cv_scores_all_features = cross_val_score(best_rf_model, X_train_all_feat
print("Cross-Validation R² Scores on All Features:", cv_scores_all_featu
print("Mean CV R² Score on All Features:", cv_scores_all_features.mean()

# Refit the model on the entire training set with all features
best_rf_model.fit(X_train_all_features, y_train)

# Make predictions and evaluate on the test set with all features
y_pred_test_all_features = best_rf_model.predict(X_test_all_features)
test_r2_all_features = r2_score(y_test, y_pred_test_all_features)
print("Test R² Score on All Features:", test_r2_all_features)
```

```
Cross-Validation R² Scores on All Features: [0.9431587  0.95546198 0.95
083541 0.93929218 0.93964358]
Mean CV R² Score on All Features: 0.9456783700068371
Test R² Score on All Features: 0.9352216263387702
```

```
# Make predictions and evaluate on the test set
y_pred_test = best_rf_model.predict(X_test)
test_r2 = r2_score(y_test, y_pred_test)
print("Test R² Score on Selected Features:", test_r2)


# Evaluate with additional metrics
test_mae = mean_absolute_error(y_test, y_pred_test)
test_mse = mean_squared_error(y_test, y_pred_test)
print("Test MAE:", test_mae)
print("Test MSE:", test_mse)
```

```
Test R² Score on Selected Features: 0.9259274500968896
Test MAE: 0.11033008092490257
Test MSE: 0.022965514293246492
```

# Conclusions

**Takeaways**:

- Random Forest is a robust algorithm suitable for complex regression tasks.
- Feature importance analysis is crucial for understanding model behavior.
- Hyperparameter tuning significantly improves model performance.
- Cross-validation is essential for assessing model generalizability

**Conclusions**:

- The model exceeded the target $R^2$ score, demonstrating high predictive accuracy.
- The synthetic dataset provided a controlled environment for model training and evaluation.
- The project showcases the effectiveness of machine learning in real estate price prediction.

**Limitations**:

- The synthetic nature of the data may not capture all real-world complexities.
- The model's performance on actual market data is yet to be tested.
- There's a trade-off between model complexity and interpretability with Random Forest.
- The current model may not account for time-series trends in the housing market.

# Thanks for watching

Any Questions?