Desxifrant el Mercat Immobiliari: House Prices - Advanced Regression Techniques

Jan Jové Famadas

Abstract— L'habitatge és una necessitat bàsica de l'ésser humà, i és crucial desenvolupar un model precís per predir-ne els preus, tant per a negociacions de compra com per avaluar la conveniència d'adquirir una propietat. Per a aquest projecte, hem utilitzat el dataset 'Advanced Regression Techniques' de la competició de Kaggle. El nostre objectiu és predir els preus d'un conjunt de dades on les solucions no són conegudes prèviament. Aquest objectiu l'abordarem mitjançant l'exploració de diversos models, incloent estratègies on es prediuen les cases conjuntament i d'altres on es prediuen separadament.

Keywords—Tecniques avançades de regressió, feature engineering, selescció de característiques.

1 Introducció

He desenvolupat un model per predir el preu de cases en una competició de kaggle permenent, anomenada House Prisces: advanced regression techniques. Aquest dataset es composa per cases venudes a california entre 2006 i 2010. Aquest està compost per més de 80 columnes, com per exemple a quin any es va construir, l'àrea de la casa o el nombre de lavabos. La nostre feina serà trobar el millor model per predir i quines son les característiques més rellevants.

2 Proposta

La nostra metodologia que seguirem és provar diversos models per veure quin és el que millor prediu el preu de les cases, per ferho començarem amb un model simple i cada vegada l'intentarem fer més complex. Per comprovar quin és el millor farem servir tant l'R2 com el RMSE.

Abans de començar a provar qualsevol model, hem realitzat un anàlisis inicial de cada feature en el dataset. En aquest hem pogut veure com les característiques més importants serien: la qualitat de la casa, l'àrea de la casa, l'àrea del solar i l'any de construcció.

En cas de tenir nans, en aquest dataset, vol dir que el valor és 0 o inexistent. Per tant, a l'hora de treure els nans, ompliré el valor amb un 0.

En la diferent tria de models com hem dit podem separar-ho en quatre etapes:

Models Simples

- No superen les 15 característiques.
- *Feature engineering* molt pobre, com a molt crear booleà: si hi ha soterrani, venuda abans de 2008 o si té aparcament.
- Models bàsics: Regressió lineal i, en el més avançats, random forest.

Models Complexos

- No superen les 20 característiques.
- *Feature engineering* més elaborat: si està a prop d'una estació o d'un carrer principal, com de lluny està de la crisi del 2008 o quins són els danys de la casa.
- Models més avançats: Lasso, random forest amb cross validation o XGBoost.

Model stacking

- No superen les 25 característiques.
- *Feature engineering* elaborat, combinant característiques, per exemple, l'àrea total de la casa sumant la primera planta, la segona i el soterrani.
- Models avançats: Ridge, random forest i XGBoost amb gridsearch, i models d'stacking amb una xarxa neuronal d'SciKit-Learn.

Dos Models Separats

- En aquesta etapa els models tenen la mateixa complexitat que en l'anterior, però separant les dades en dues: intentant predir les cases barates i les cares per separat.Aquesta separació es farà a traves d'un model per predir els errors, a partir d'aquest veurem quina és la característica que marca l'error.

A tots els models he utilitzat un preprocessing similar. Les variables que marcaben notes: bé, notable, excelent... Les he considerat numèriques per tant les he transformat a un numero del 1 al 5. He tret els nans amb un 0, normalitzat les dades numèriques, fet un TargetEncoding (ja que amb altres encodings obtenia masses columnes). La única diferència és que a partir de models més complexos he començat a fer servir una pipeline.

Per mesurar com funcionen els models en general, utilitzarem el RMSE ja que és com kaggle avalua el nostre model. També farem servir la R2 per valorar com van avançat els diversos models.

3 Experiments, resultats i anàlisi

Com hem explicat podem separar entre diverses etapes el procés que hem seguit per trobar el millor model, a part entre les seccions hem provat diversos experiments per provar de millorar els models. Els resultats en cada fase, i els resultats finals provats tant amb el test visible creat aleatòriament amb dades del train, com amb el test proporcionat del kaggle on no sabem els resultats.

3.1 Model simple

Els models simples son els compresos entre els models 1 i el 4. El primer model per començar hem aconseguit una r2 de 0'75 i un RMSE de 38000, agafant les 7 característiques més importants. En el millor model d'aquesta fase és el model 3 amb una r2 de 0,84 i RMSE 30 000 utilitzant random forest amb cross validation. On ja hem creat variables de posició com podries ser si esta aprop d'una estació o del carrer principal.

La informació més rellevant d'aquesta fase són quines les característiques més importants: el barri, l'àrea de la primera planta i la de la segona. També hem pogut veure com el random forest funciona millor que una regressió lineal.

3.2 Model complex

Els models complexes son entre el 5 i el 7, en aquesta fase aconseguim millorar tant la R2 com el RMSE aconseguint pujar del 0'85 i baixar de 30.000 respectivament. En part gràcies a noves característiques creades com si està al costat d'una estació o del carrer principal, els danys de la casa amb valor logarítmic o si tenen soterrani.

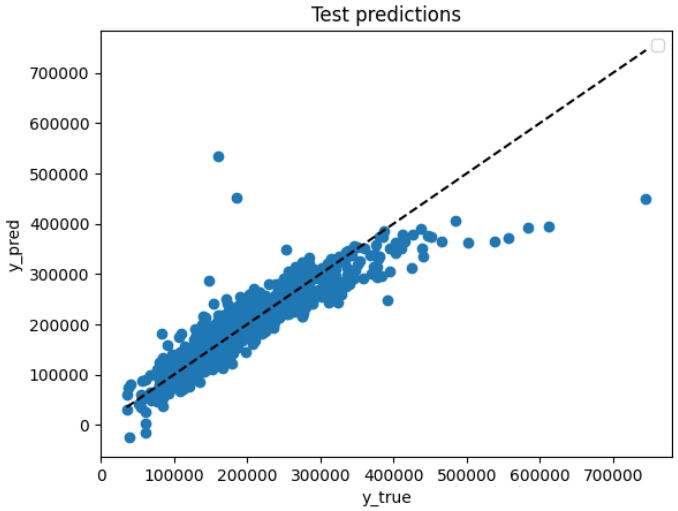

Fig. 1: Plot x=yreal y=ypred

Si fem el plot de les y reals amb les y predites podem veure com infravalorem les cases cares i sobrevalorem akgunes cases barates.

3.3 Model stacking

Aquest apartat compren des del model 8 al 14. En aquest cas el millor model és el 9,12 i 13, i el, amb tots dos fent stacking o XGBoost aconseguim un 0'87 de R2 i 27000 de RMSE.

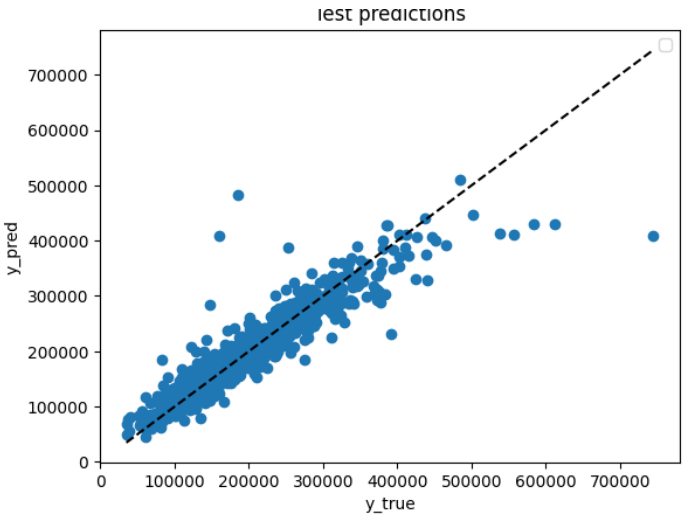

Fig. 2: Plot x=yreal y=ypred

Com abans, veiem que seguim infravalorant les cases més cares, i sobrevalorant algunes cases barates. És cert que obtenim més precisió, en general, però seguim tenint els mateixos problemes que abans, per tant provarem de canviar el punt de vista.

3.4 Lime explainability

Per torbar com arreglar els errors dels models, vam provar de fer feature importance dels punts cars i els punts que els models sobrevaloren. Primer farem un lime explainability amb les característiques del test 12, per veure quines son les més importans en les cases cares.

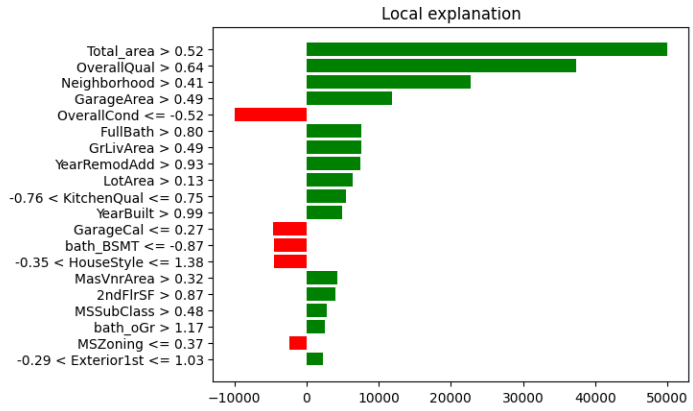

Fig. 3: Lime explainability de les cases cares

Com podem veure l'àrea total, la qualitat de la casa, el barri i l'àrea del garatge son molt importants. La condició global és una variable que ens afecta negativament en les cases cares per tant pot provocar aquesta infravaloració. Segon hem fet el mateix en les cases sobrevalorades:

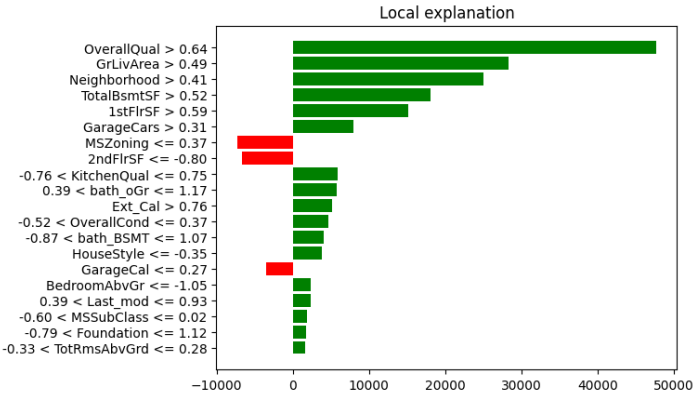

Fig. 4: Lime explainability de les cases sobrevalorades

En aquest cas podem veure com abans que les mateixes característiques son les mes importants nomes es diferencia el metres en el soterrani. També cal remarcar que a diferència de les cases la condició/estat de la casa és positiva.

Tercer hem provat de fer un model amb totes les característiques disponibles, per veure les característiques més rellevants en els punts cars:

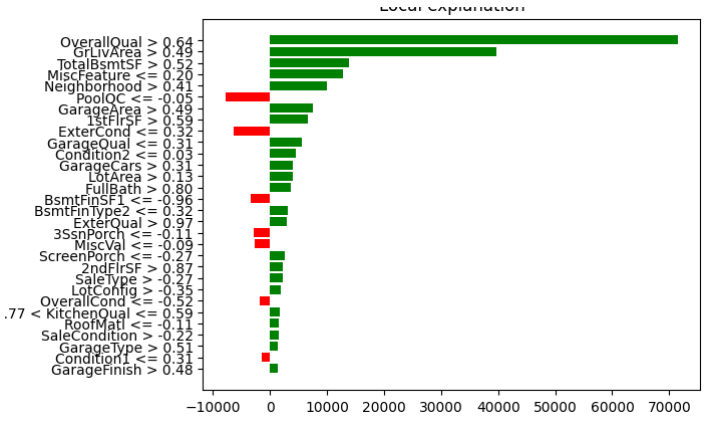

Fig. 5: Lime explainability de les cases cares amb totes les caracrterístiques

Com hem vist abans, podem veure que les features més rellevants son la qualitat ¡ l'àrea de la casa, en aquest cas podem veure que pot ser interessant la Miscelaneous Feature.

3.5 Model per trobar la separació entre cases

Per trobar com separar les cases barates de les cares sense saber quina és la variable target hem fet un model per predir l'error de cada punt, aquest model amb la y predita del 12, hem mirat quines són les features més importants que en el nostre cas, son totes referents a l'àrea de la casa. Per tant buscarem quina pot ser una possible separació entre les cases.

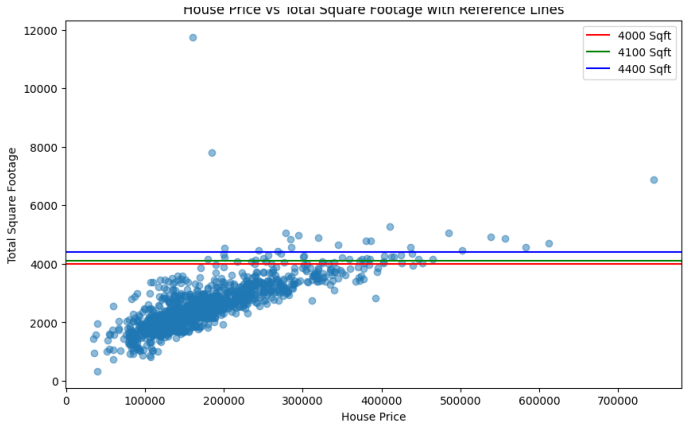

Fig. 6: x=Preu de la casa y=Metres de la casa

Com podem veure el 4400 marca una bona separació, de fet en cas de fer un plot amb les y pred i les y reals on les vermelles son les que tenen una area més gran de 4400(fig 7).

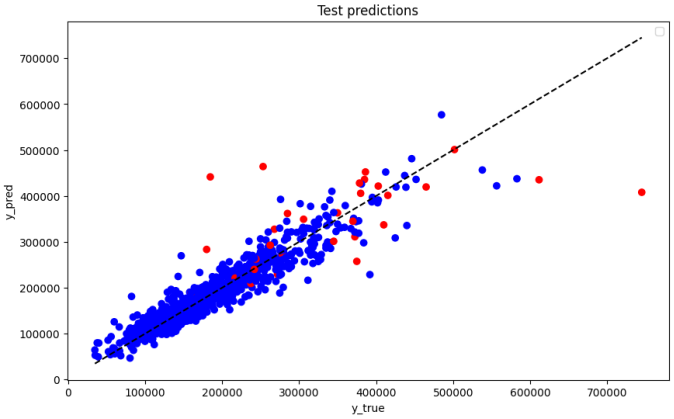

Fig. 7: x=Preu real y=Preu predid, vermell les cases +4400

3.6 Model separats

Un cop trobada la separació, hem provat dos models diferents per cada una de les separacions. És a dir no només fem models de predicció diferents sinó que també escollim característiques diferents.

Primer hem provat un model amb la mateixa selecció de característiques per tenir un punt de partida, al qual hem obtingut: en les mes barates hem aconseguit 0,9 R2 i 21500 RMSE, en les cares 83700 RMSE i no hem pogut computar R2, en total hem obtingut un R2 0,8999 i un RMSE de 24600. Millorant així el model anterior.

Després hem provat quin era el model que prediu millor les dues classes per separat, sobretot ens hem centrat en millorar el model per predir les cases cares, ja que en tenir poques dades és tot molt més difícil. Després de les millores hem obtingut que les cases barates hem aconseguit una petita millora arribant a un RMSE de 21400, en canvi, no hem pogut millorar el model inicial en les cases cares. Assolint un total de 0,9 de R2 i un RMSE de 24300.

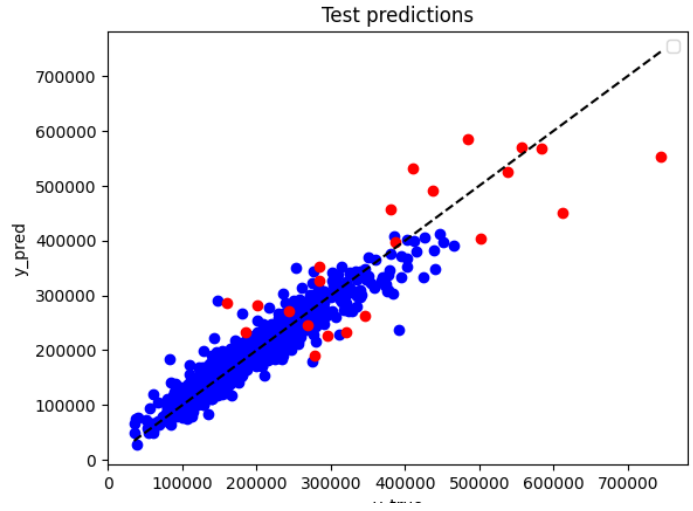

Fig. 8: x=Preu real y=Preu predid, vermell les cases +4400

Com podem veure ara hem aconseguit solucionar els problemes que teníem ja no sobreestimem algunes cases ni infravalorem tant les cases cares, per poder millorar hauríem d'intentar predir millor les cases cares.

En els diferents intents que hem provat de canviar el número de separació entre classes. Un dels problemes principals que tenim,

és que en cas d'una separació marcada ens quedarem amb masses poques dades per poder predir el preu correctament. En canvi, si volem més dades pot passar que no separem prou bé les classes, per tant, tornem a caure en infravalorar les cases cares.

3.7 Solució total

Els resultats de tots els models son aquest:

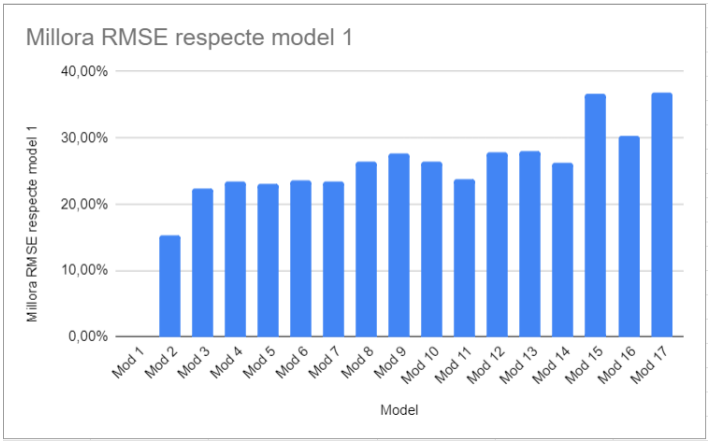

Fig. 9: Millora de tots els models respecte el primer intent

3.8 Comparació de model junt i separats

Com podem veure el model on separem el train en cases cares i barates, arreglem el problema tant de la infravaloració de cases cares com la sobrevaloració de les cases amb una area més gran pel preu que tenen.

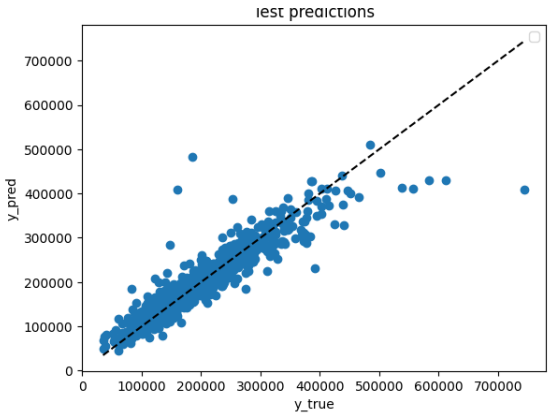

Fig. 10: Model sense separació

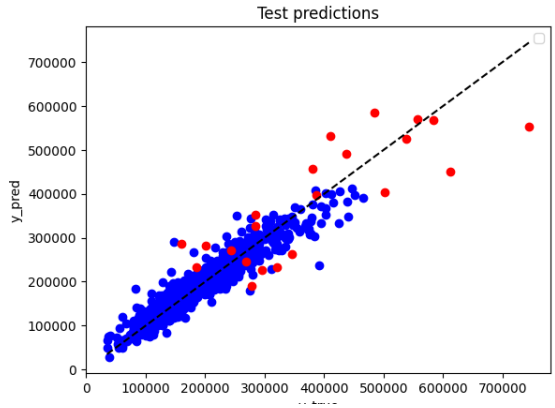

Fig. 11: Model amb separació

Per tant podem veure que el millor model és quan separem entre les cases. S'observa com els valors sobrevalorats a la figura 9 eren cases grans per tant al fer la comparació global les consideràvem igual de bones que les cases més cares.

3.9 Model final

Com hem pogut veure el millor model és el 17: Cases barates

1. Característques:'MSSubClass','LotArea','Neighborhood', 'OverallQual','OverallCond','GrLivArea',BedroomAbvGr', 'TotRmsAbvGrd','GarageCars','CentralAir','1stFlrSF', '2ndFlrSF','TotalBsmtSF','FullBath','YearBuilt', 'YearRemodAdd','GarageArea','MasVnrArea','KitchenQual'
2. Feature engineering: bona posició, area total, nombre de lavabos i si havia estat una venta normal
3. Model: Stacking Regressor

Casses cares:

1. Característques:'MSSubClass','LotArea','Neighborhood', 'OverallQual','OverallCond','GrLivArea','BedroomAbvGr', 'TotRmsAbvGrd','GarageCars','CentralAir','1stFlrSF', '2ndFlrSF','TotalBsmtSF','MSZoning','HouseStyle', 'FullBath','YearBuilt','YearRemodAdd','GarageArea', 'MasVnrArea',"KitchenQual'
2. Feature engineering: bona posició, area total, nombre de lavabos
3. Model: XGBoost

3.10 Test

Abans de començar qualsevol model hem fet una separació entre train i test. Ara utilitzem aquest test per provar com va. Com hem explicat farem servir dos models diferents, entrenats amb dades diferents. Els resultats que hem obtingut son:

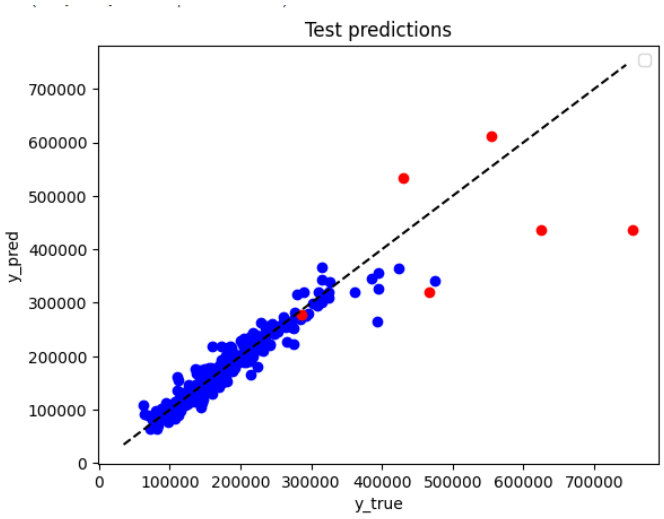

Fig. 12: Test:x=Preu real y=Preu predid, vermell les cases +4400

Com podem veure els resultats del test son bastant més dolents que el train, sobretot en el RMSE que és de 31000. Aquest descens el podem veure en la baixada de les cases cares, ja que en aquest cas obtenim un RMSE de 169000, a diferència de l'anterior que obtenim un RMSE de 85000. En canvi a la part de les cases barates en general sí que és manté el observat en el train. Tot i que es pot veure tornem a infravalorar les cases cares, en aquest cas en el grup de les cases més petites de 4400.

3.10.1 Analisi d'errors

En els errors a les cases grans, si agafem la casa amb més preu en el test i la comparem amb la casa amb més preu en el train veiem que son casi iguals, només canvia en que un mur i l'altre no.Que ho marca la característica MasVnrArea.

En el cas de les cases infravalorades no hem estat capaços de trobar cap diferència amb les casses que hem predit amb el mateix preu, segurament si les haguéssim classificat com a cares hauríem pogut millorar la predicció. Hauríem d'haver intentat incloure a la categoria de cases cares o si més no trobar una característica que expliqui la diferència de preu.

3.11 Test Kaggle

Hem fet diverses submissions al test kaggle, curiosament la millor ha estat la del model 14, quan, en el train, en cap cas era el model amb millors resultats. Aquesta submissió en ha donat una puntuació 0.129

Com era d'esperar amb l'error en el test el nostre model final dona certs errors, amb una puntuació de 13.1. En cas de voler millorar les posicions al cas kaggle seguiria amb el model de dues classes separades.

4 Conclusió

En aquest treball, hem desenvolupat un model per predir el preu de les cases utilitzant dades de Kaggle, amb l'objectiu de predir el valor de les propietats amb la màxima precisió possible. Hem aplicat enginyeria de característiques i hem provat diversos models, des de models simples fins a altres més complexos com el random forest, XGBoost, i models d'stacking. Hem observat que els models més complexos i la separació del conjunt de dades en grups diferents de cases (barates i cares) han proporcionat resultats més precisos i fiables.

La metodologia de començar amb un model simple ha estat clau en l'anàlisi d'un conjunt de dades amb moltes columnes, per evitar obviar característiques importants. Això ens ha guiat constantment en la millora del model, com podem veure en el següent gràfic.

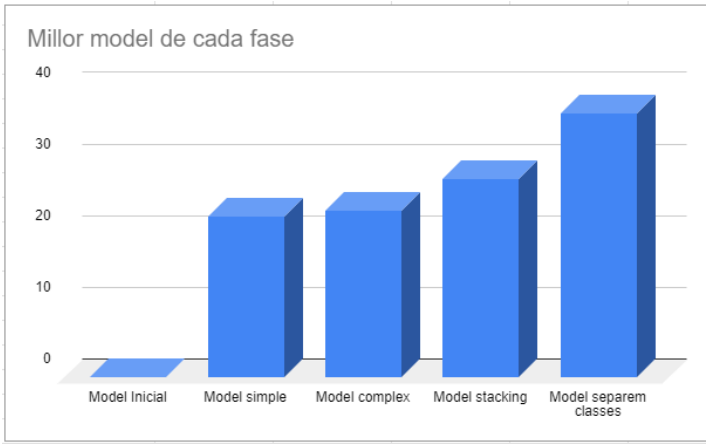

Fig. 13: x=Preu real y=Preu predid, vermell les cases +4400

Hem identificat que les característiques més importants per determinar el preu d'una casa són les seves dimensions, la mida de la parcel·la, la capacitat del garatge, el nombre de banys, la qualitat de la construcció i la ubicació de la propietat.

El feuture engineering és una eina molt eficaç, a moltes de les columnes la informació és dispersa, per exemple, en la columna de localització tenim si estan aprop de l'estació, d'un carrer principal... Aquesta columna és pot convertir en 3 o 4 boolenas, o en cas de voler fer un més general podem fer una boolear de si és bona la localització. Aquest és un exemple de com hauriem convertir una variable poc útil, amb una o algunes que ens aporta bastanta informació.

Entre les possibles millores a considerar, destaquen: primer, el desenvolupament d'un model específic per millorar la predicció de la classe de cases cares; segon, la recerca de més d'un criteri de separació. Fins ara, hem separat les cases grans de les petites, quan l'objectiu hauria de ser distingir entre propietats cares i barates. Això podria incloure l'ús d'un model de classificació.

Un repte en el context de Kaggle ha estat la incapacitat de veure els resultats directament, el que impedeix comprendre per què uns models funcionen millor que altres. En el nostre cas, el millor model a Kaggle era diferent del que esperàvem, destacant la possible discrepància entre un model teòricament perfecte i la seva aplicabilitat real.

En conclusió, la millor manera d'abordar aquest problema és mitjançant un model que separi les cases. No obstant això, hem trobat dificultats degut a la implementació deficitària del model, a una estratègia de separació inadequada de les cases, o a la falta de dades sobre les propietats de preu més elevat.